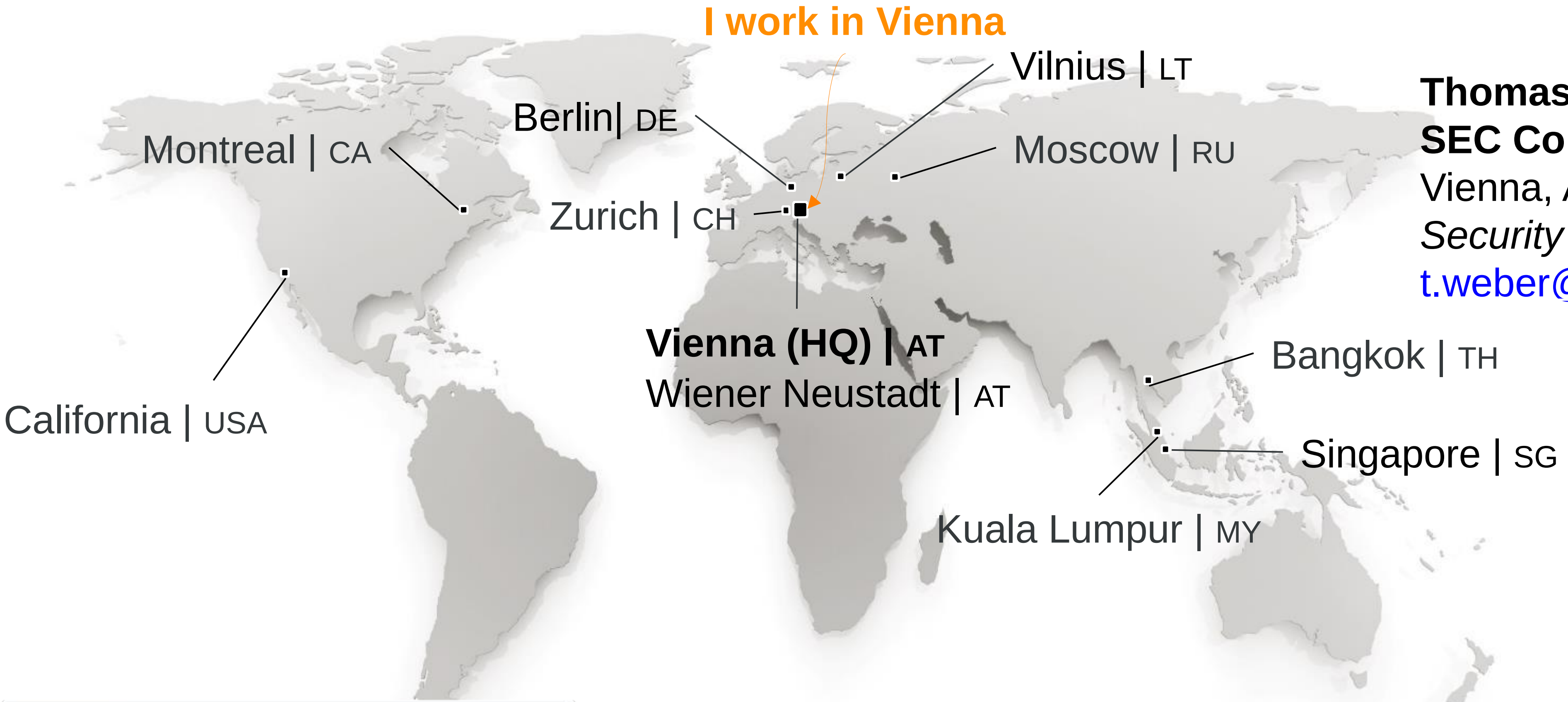
Reverse Engineering Custom ASICs by Exploiting Potential Supply-Chain Leaks

IT-SECX2019

© fotolia 41706530

$ whoami



I work in Vienna

Thomas Weber
SEC Consult Group
Vienna, Austria
Security Researcher & Consultant
t.weber@sec-consult.com

SEC Consult

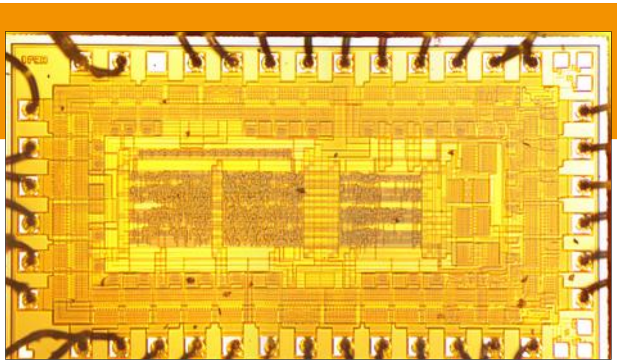
My employer since 2015

Outline

At a glance:

- Introduction & motivation – important notes
- Dangers of supply-chains
- Reverse engineering methods
  - Deductive reasoning - probing methods
  - Deeper insights
- Live debugging & demo
- Fun fact
- Conclusion

Introduction



What is an ASIC?

→ Application Specific Integrated Circuit

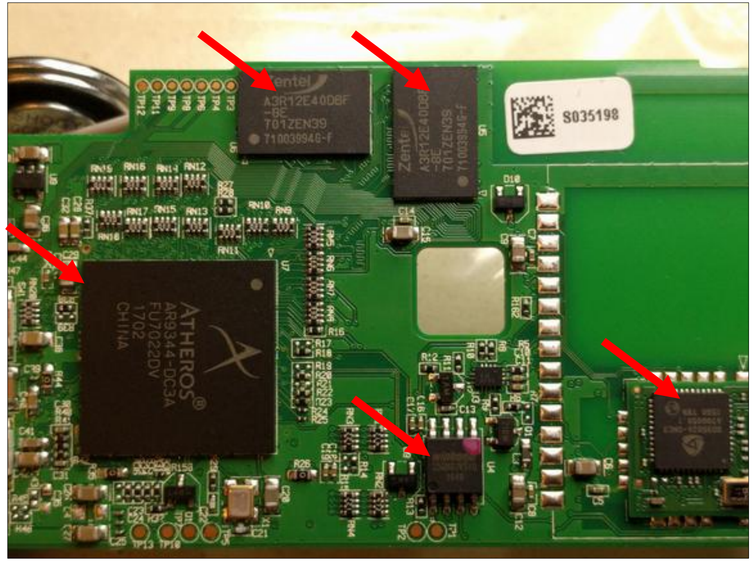
→ Can also be a System on Chip solution with customized peripherals (theoretically everything)

Who cares?

→ Vendors, security researchers, blackhat hackers...

Where is it used?

→ In every (embedded) computer system. There are more precise names for the specific applications like SoC, ASIP, NoC and so on.



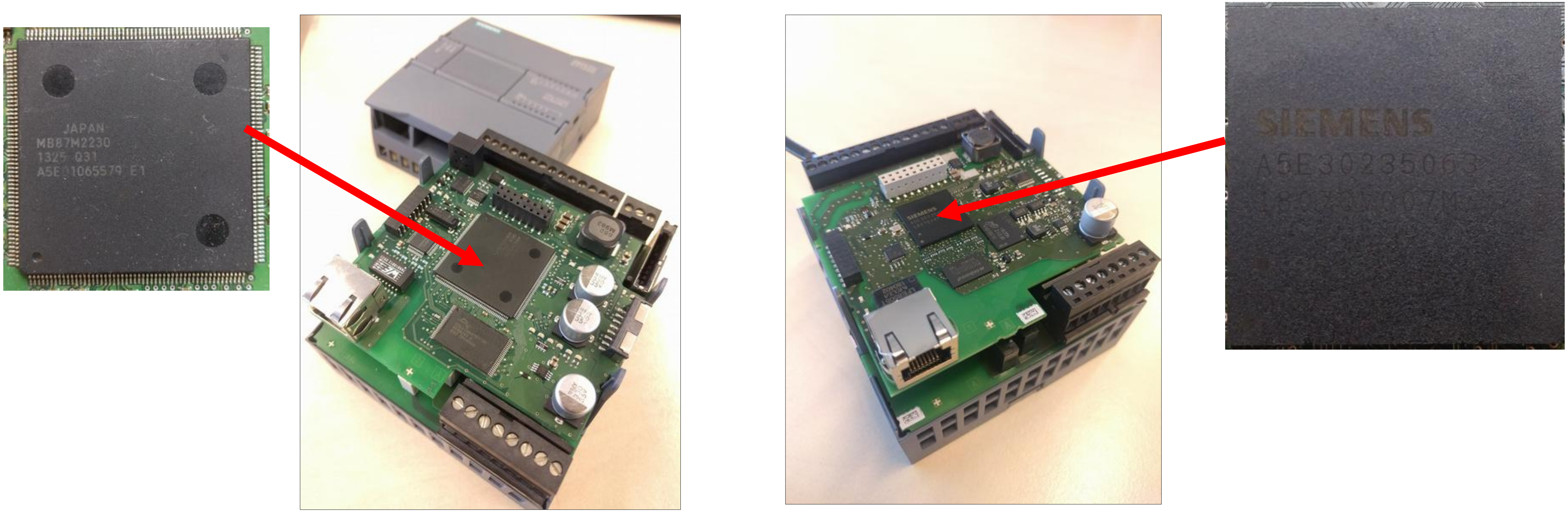
Introduction

It gets hard when there are **complete custom** chips without public documentation.

This means:
→ Architecture is **unknown**
→ Pinout is **unknown**
→ I/O memory map is **unknown**
→ Additional constraints are **unknown**
→ Sometimes, even the vendor is **unknown**

Motivation

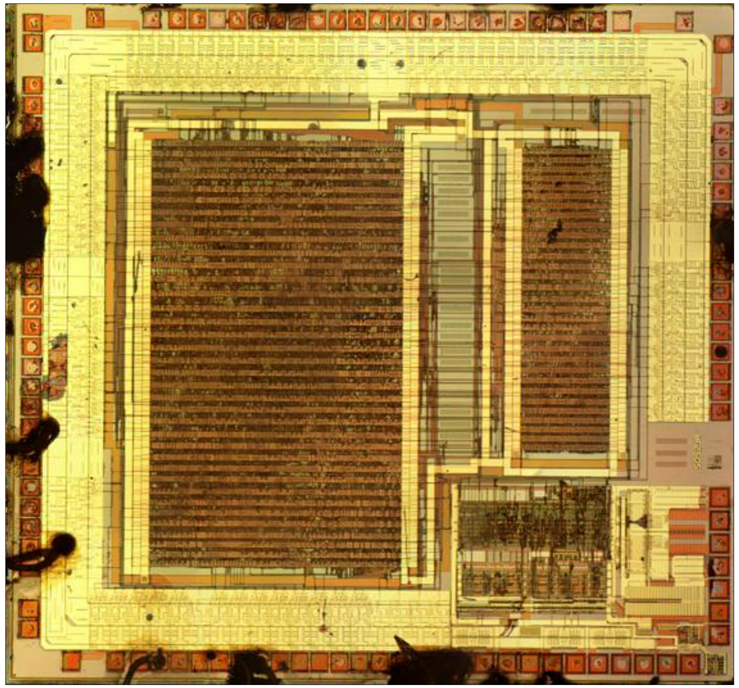
A textbook example for custom ASICs can be found inside of industrial products like the PLC series S7-1200. There are even different hardware versions of this PLC series, and two different main chips. **Can we identify the JTAG port?**



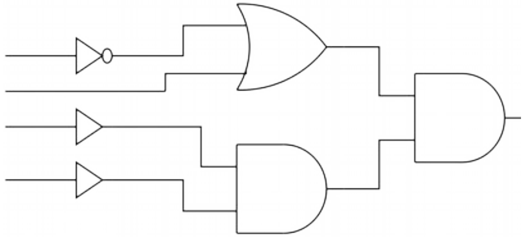
Motivation

Expensive option: Decapping, FIB or SEM and delayering, recover the hardware.

From silicon die… *… to hardware description.*



Source: https://www.capovani.com



Motivation

Cheap option: Search for similar hardware with the **same chip** on the internet.
Good sources are: strange online shops, eBay, AliExpress and Taobao (淘宝网)
Multiple PCBs with the **same chip** are even better to reverse engineer each pin functionality. The possibility to identify debug ports by having multiple different PCBs with the same chip is higher.

Bad:

Not all secrets of the hardware can be revealed in that way.

Good:

No need for super expensive equipment!

Dangers of supply-chains

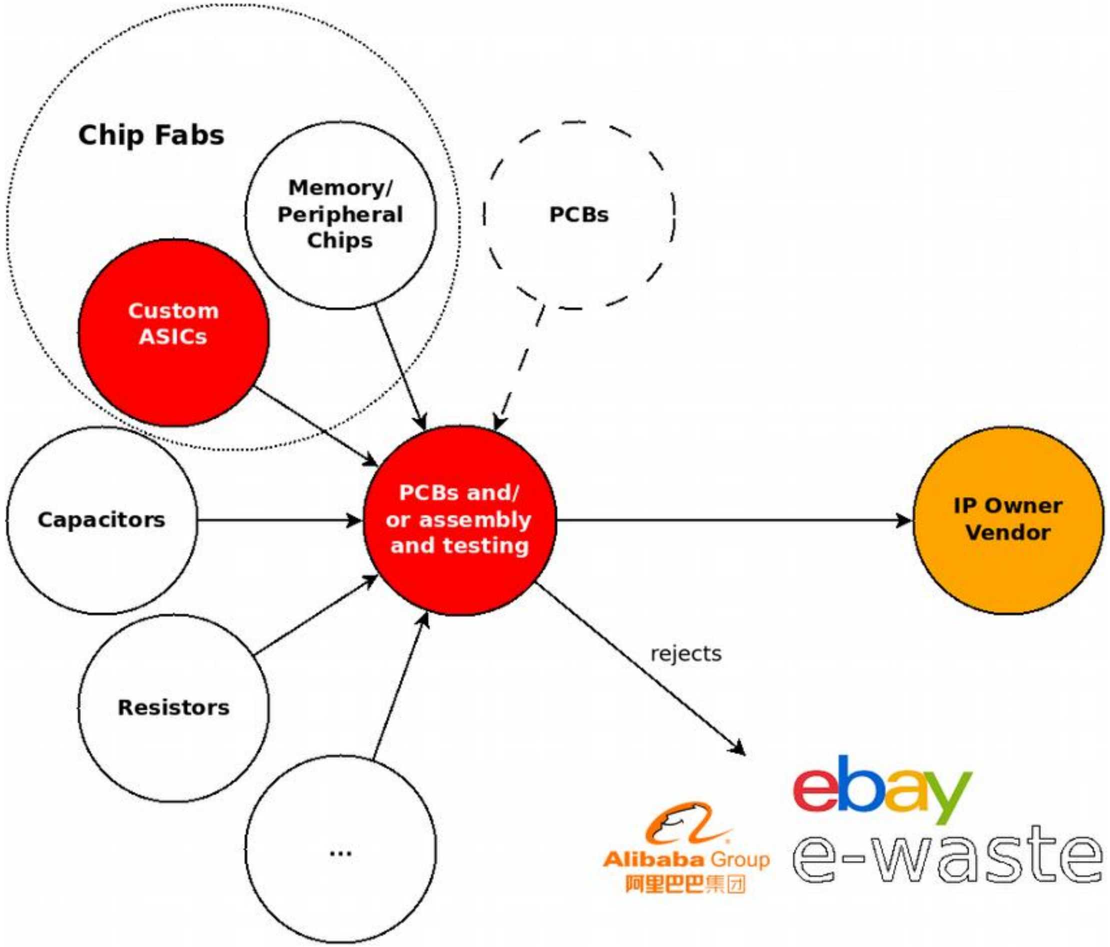
Where are the dangers of loosing IP?

Rejects:

- Leaks of prototypes
- Leaks of original boards
- Leaks of dev. boards

Espionage:

- Leaks of blueprints

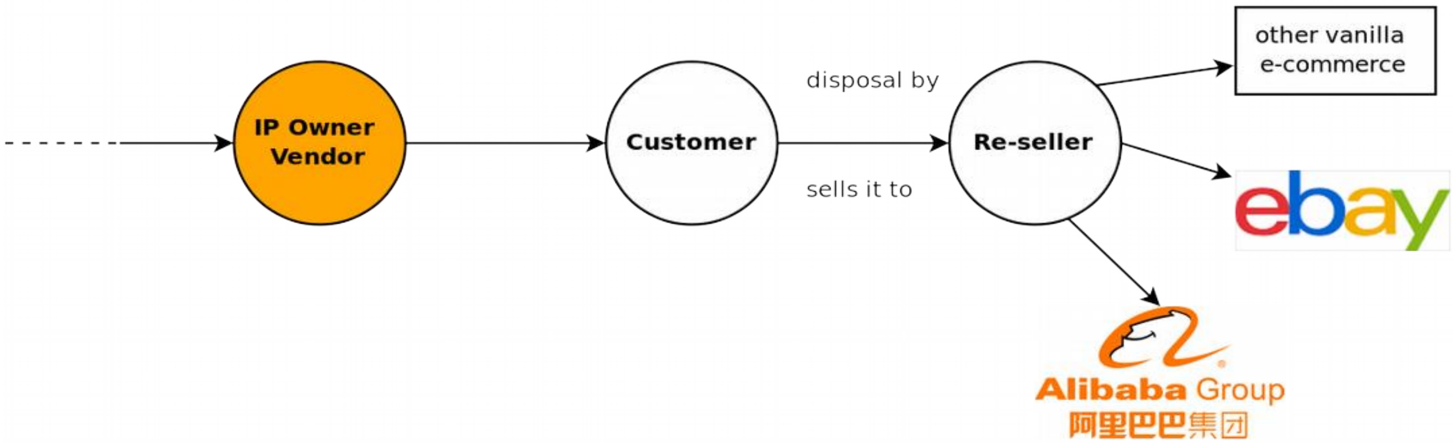


Dangers of supply-chains

Where are the dangers of losing IP?

Aftermarket issues:

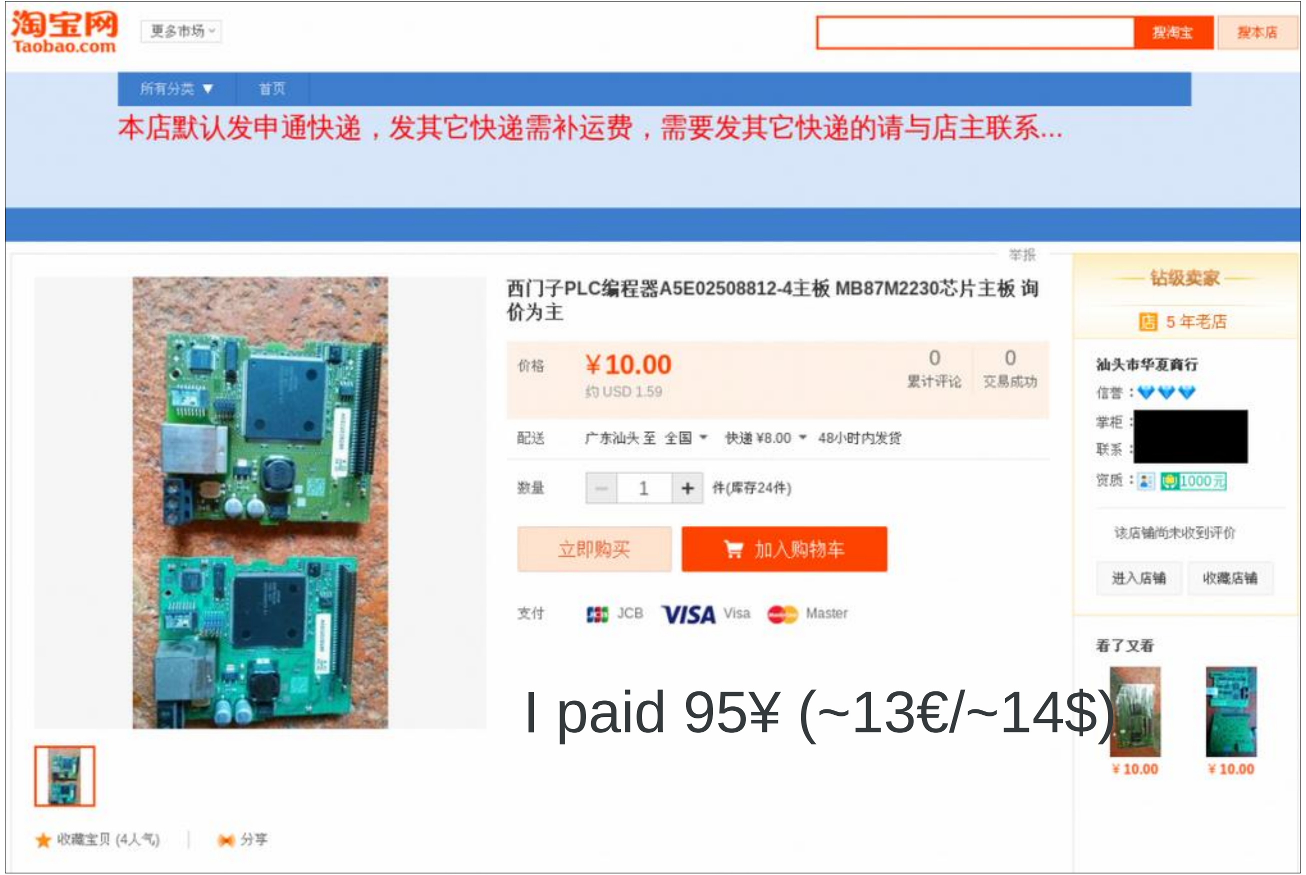
A product, which is hard to unearth (very expensive or just available when you have a contract with the vendor) is available in big cheap batches from a re-seller. This enables you to do reverse engineering even with a small budget.



→ Cheap option from previous slide!

Dangers of supply-chains – Material Chasing

Searching for the label of the ASIC used in the older S7 1200v1 on Google gave some results, one of them was Taobao:



本店默认发申通快递，发其它快递需补运费，需要发其它快递的请与店主联系...

举报

西门子PLC编程器A5E02508812-4主板 MB87M2230芯片主板 询价为主

价格 ¥10.00
约 USD 1.59

0 累计评论 | 0 交易成功

配送 广东汕头 至 全国 ▾ 快递 ¥8.00 ▾ 48小时内发货

数量 − 1 + 件(库存24件)

立即购买 | 加入购物车

支付 JCB VISA Visa Master

— 钻级卖家 —

5 年老店

汕头市华夏商行
信誉：💎💎💎
掌柜：[redacted]
联系：
资质：1000元

该店铺尚未收到评价

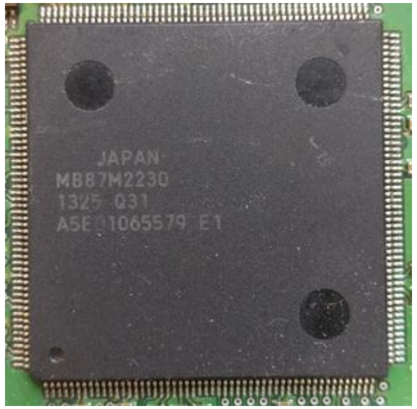
进入店铺 | 收藏店铺

看了又看

¥10.00 | ¥10.00

★ 收藏宝贝 (4人气) | 分享

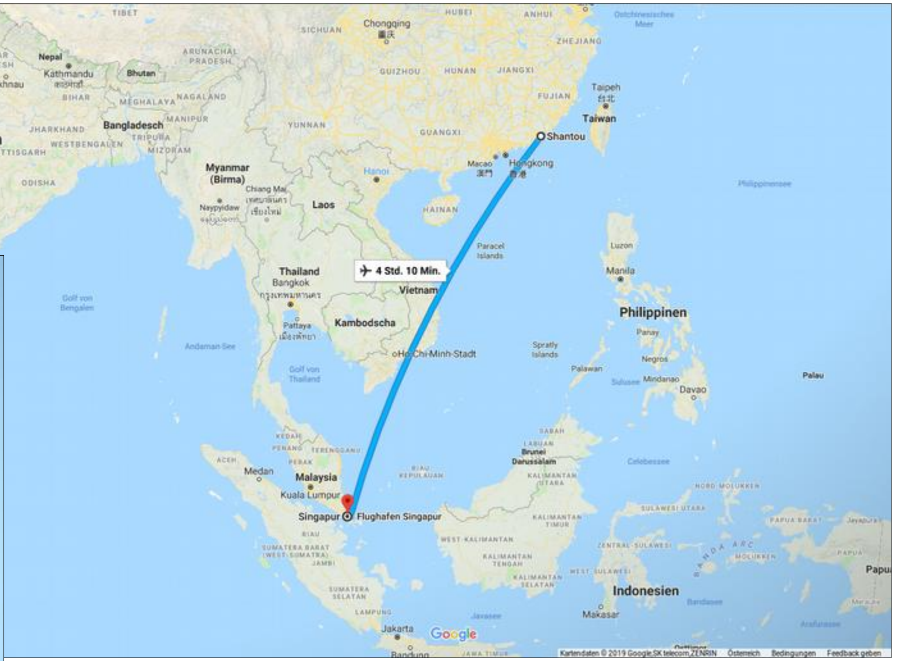
I paid 95¥ (~13€/~14$)

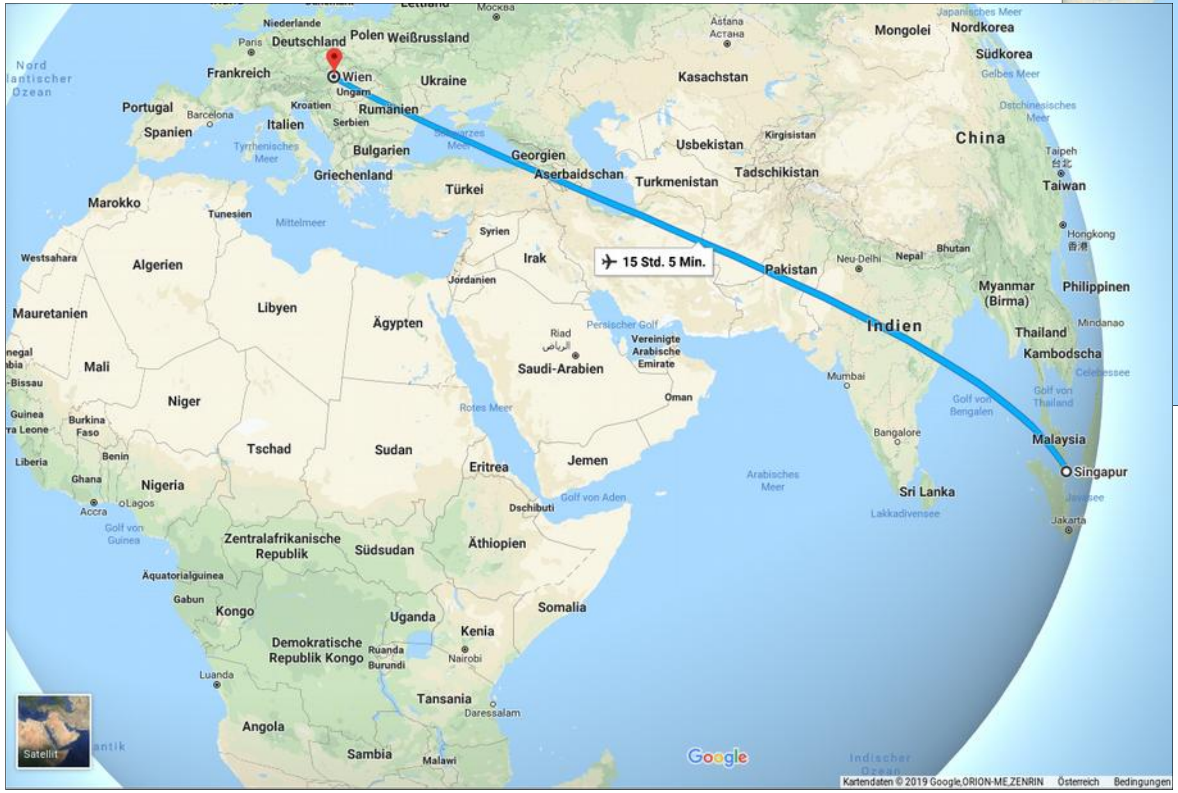


MB87M2230

Dangers of supply-chains – Material Chasing

Taobao just sells stuff inside China. Colleagues and friends from Singapore and China came to the rescue!

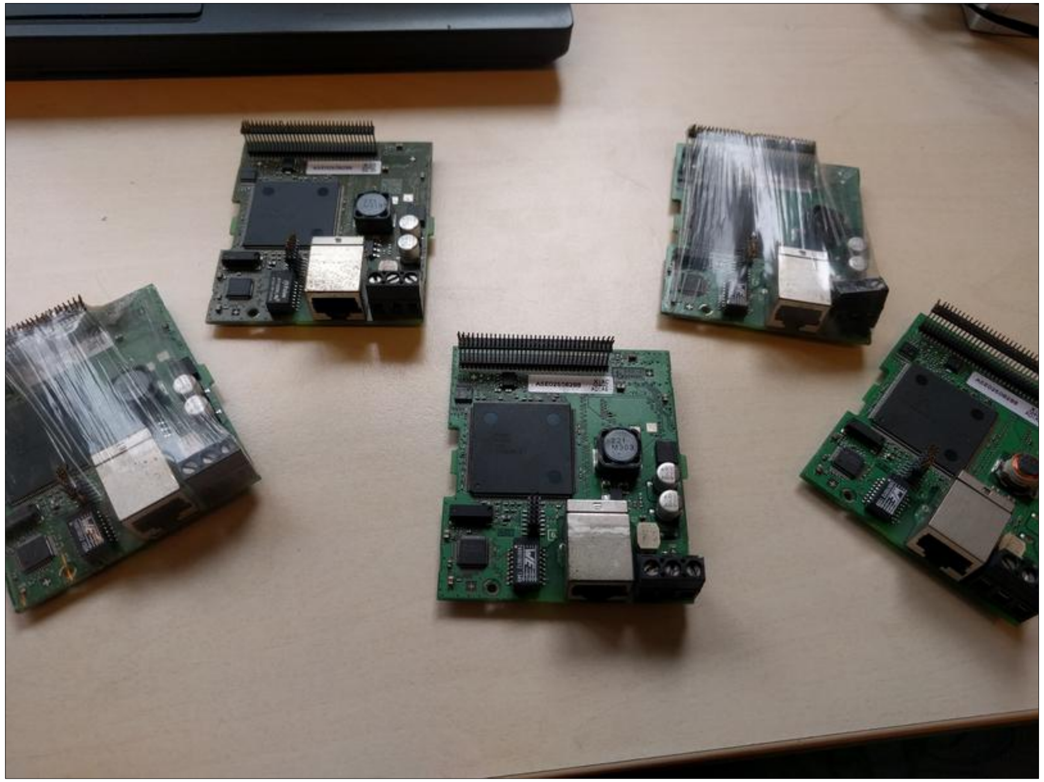




Two batches were ordered one after the other.

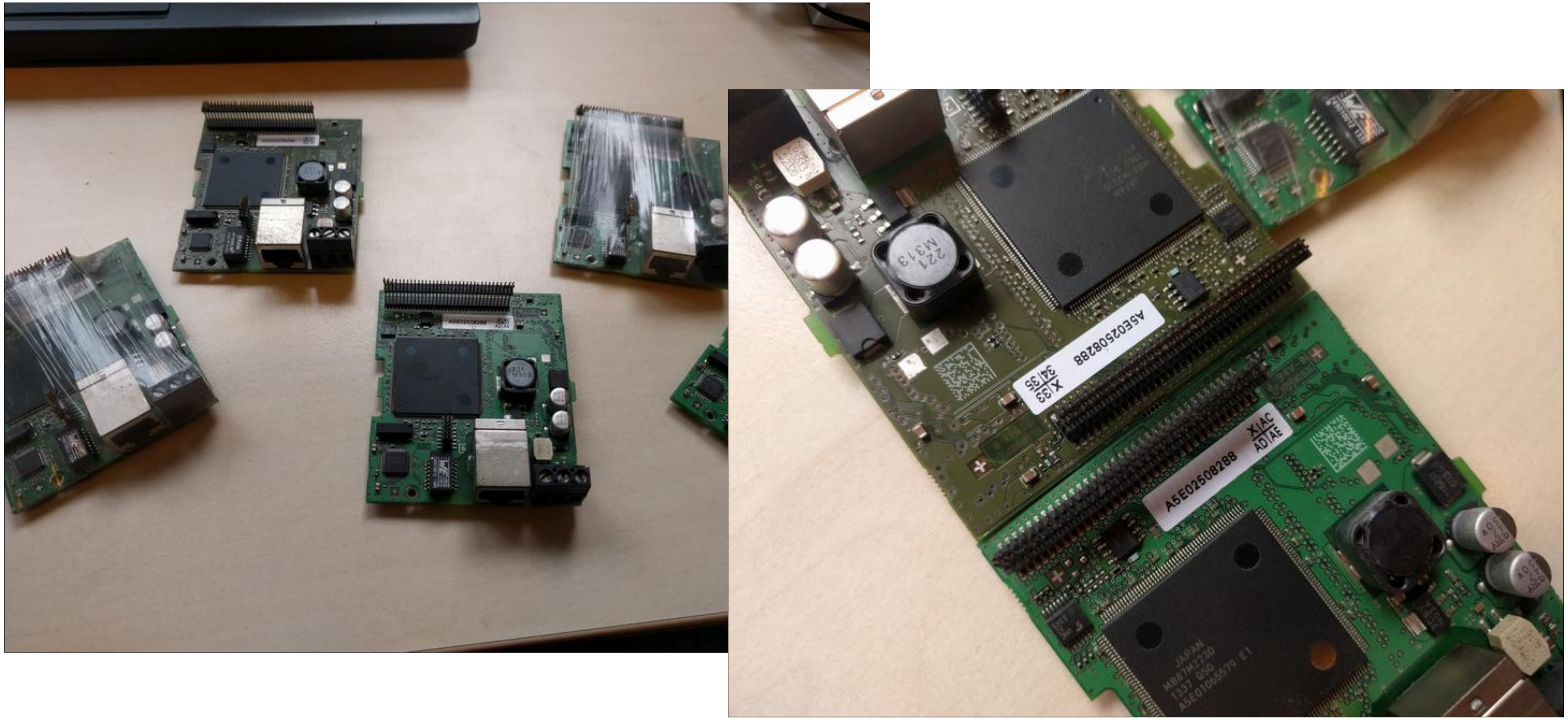
Dangers of supply-chains – Material Chasing

The first batch (MB87M2230)



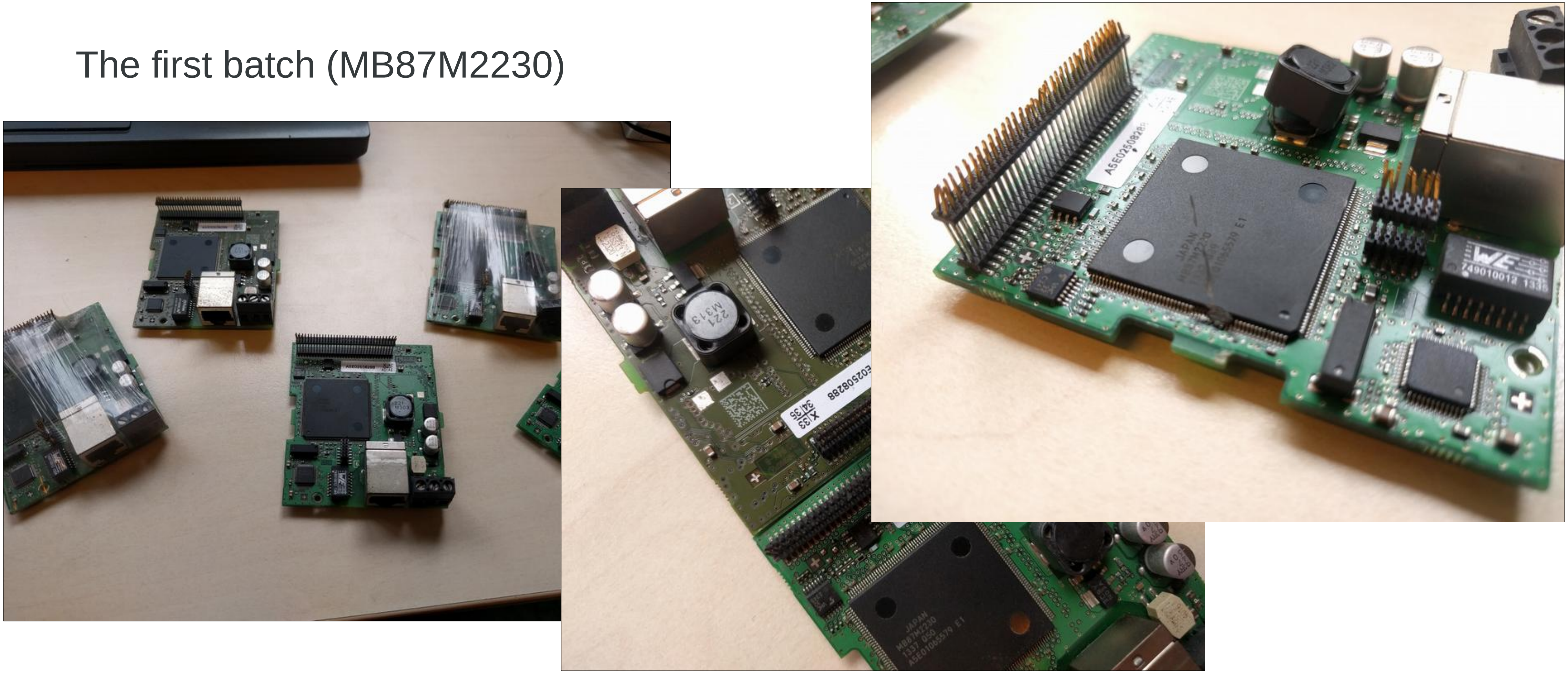
Dangers of supply-chains – Material Chasing

The first batch (MB87M2230)



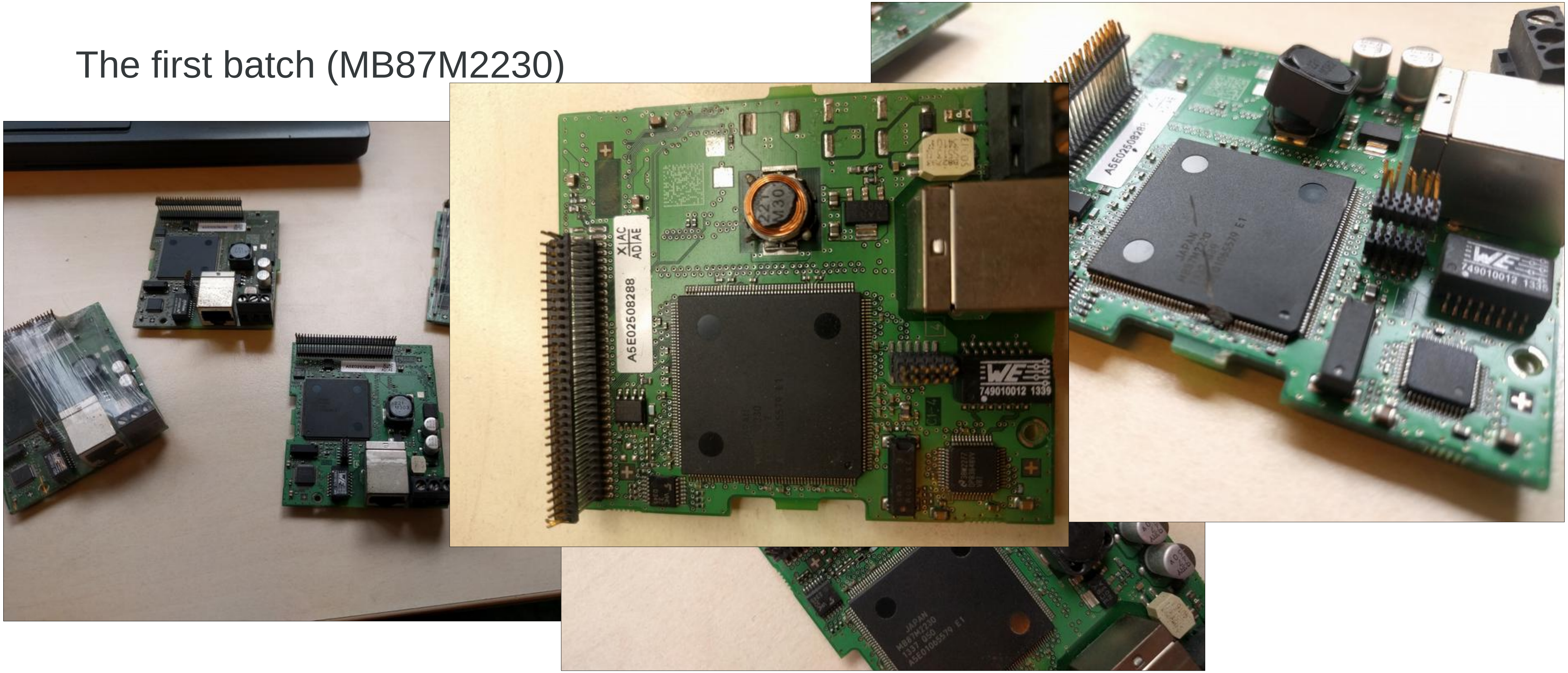
Dangers of supply-chains – Material Chasing

The first batch (MB87M2230)



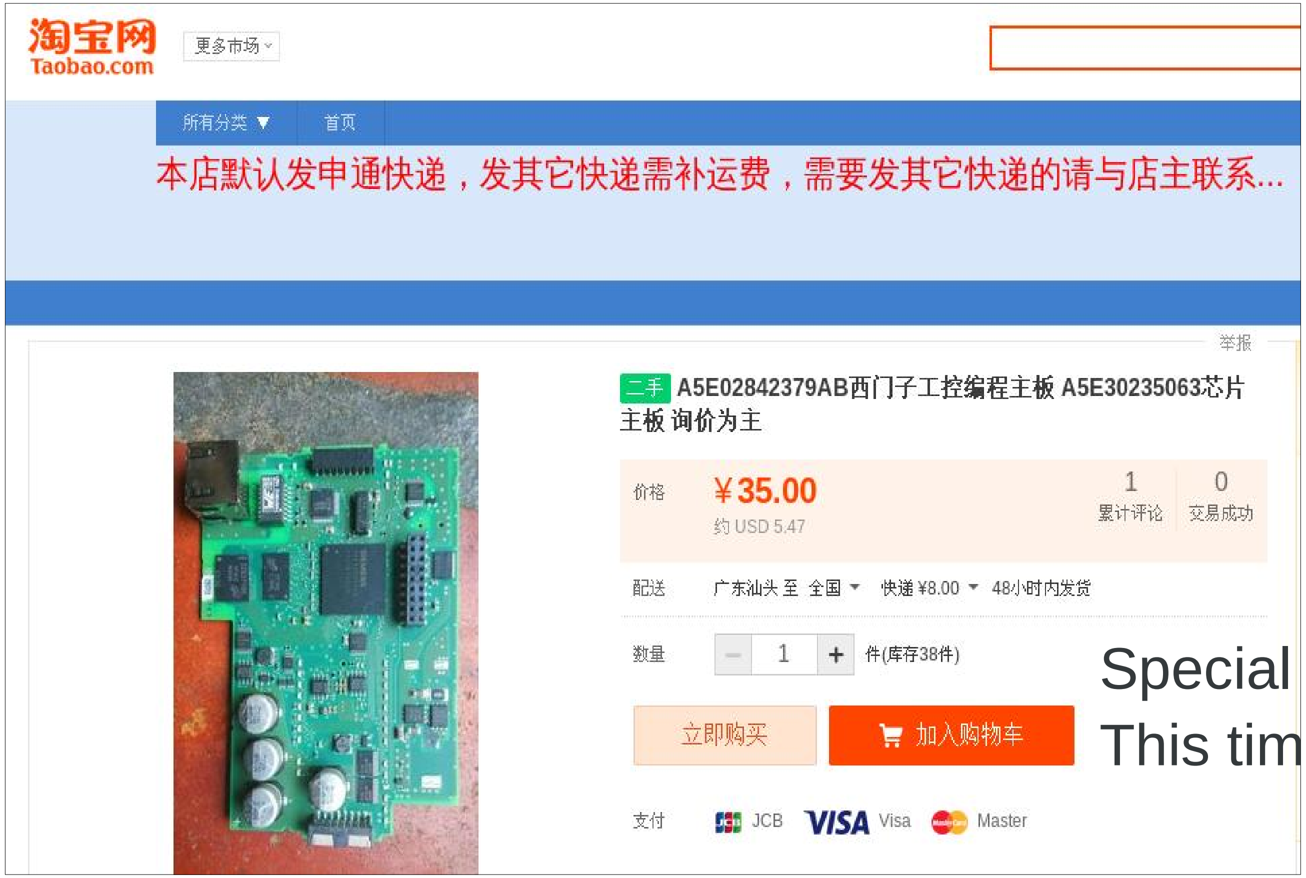
Dangers of supply-chains – Material Chasing

The first batch (MB87M2230)



Dangers of supply-chains – Material Chasing

Searching for the label of the ASIC used in the newer S7 1200v4 on Google gave some results, one of them was Taobao, again from the same seller:



淘宝网 Taobao.com 更多市场

所有分类 ▼ 首页

本店默认发申通快递，发其它快递需补运费，需要发其它快递的请与店主联系...

举报

二手 A5E02842379AB西门子工控编程主板 A5E30235063芯片主板 询价为主

价格 ¥35.00 约 USD 5.47

1 累计评论 0 交易成功

配送 广东汕头 至 全国 快递 ¥8.00 48小时内发货

数量 − 1 + 件(库存38件)

立即购买 加入购物车

支付 JCB VISA Visa Master

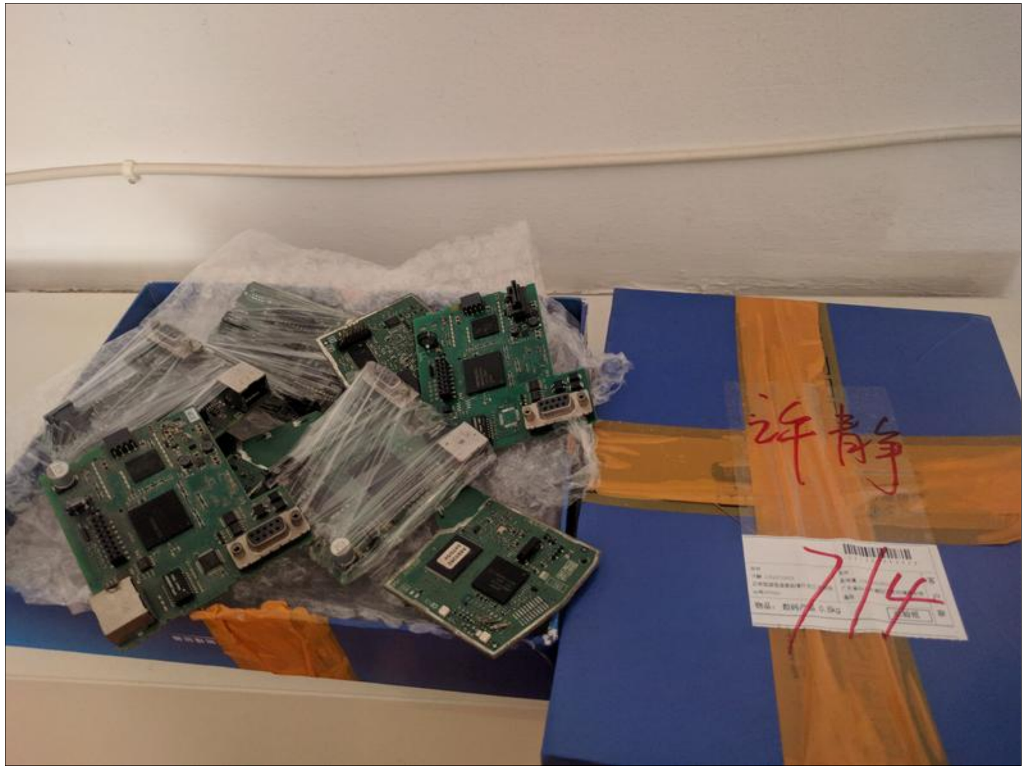
A5E30235063

Special Price!

This time just 80¥ (~11€/~12$)

Dangers of supply-chains – Material Chasing

The second batch (A5E30235063)

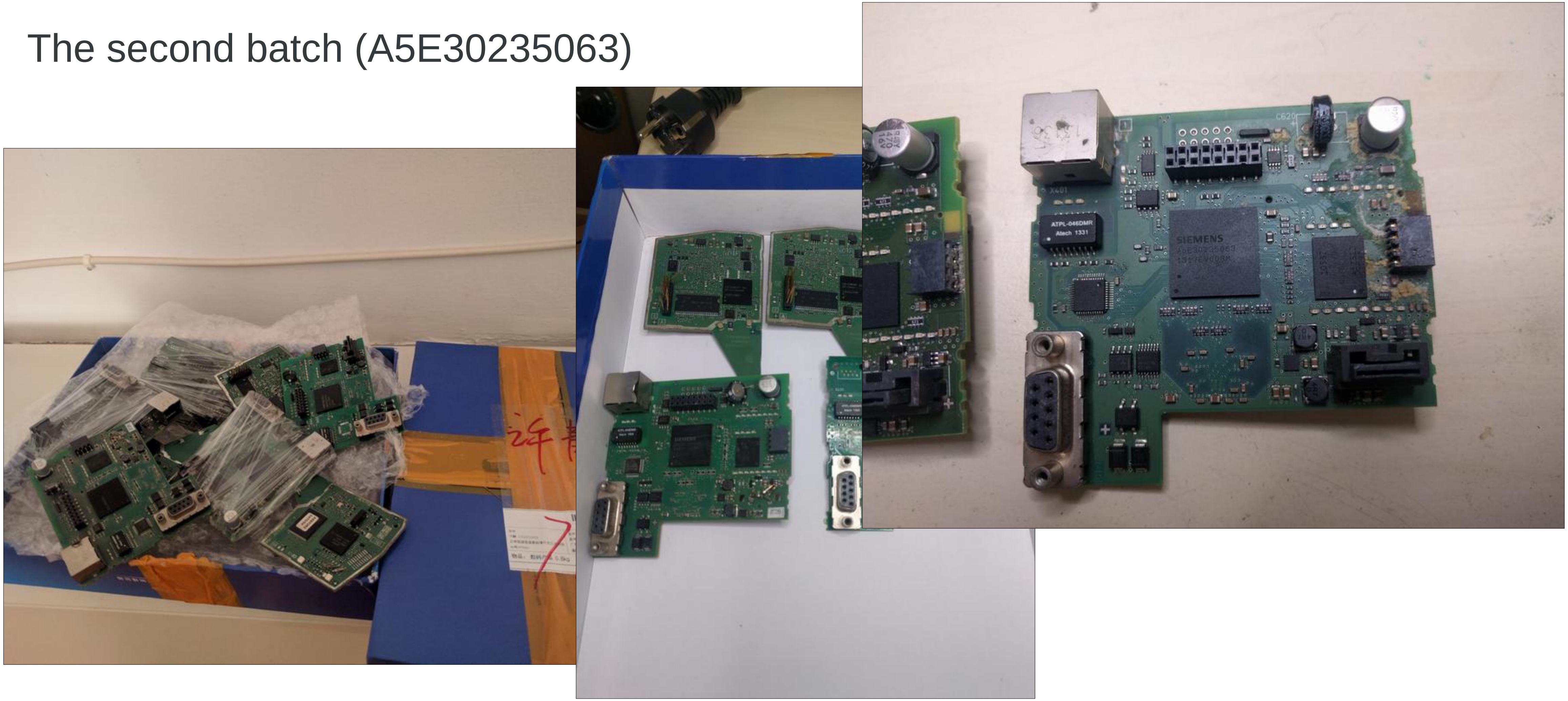


Dangers of supply-chains – Material Chasing

The second batch (A5E30235063)

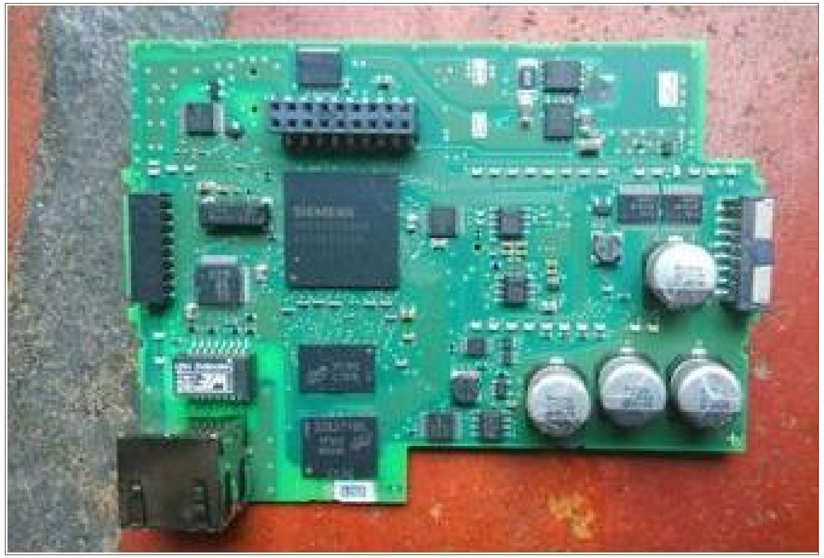
Dangers of supply-chains – Material Chasing

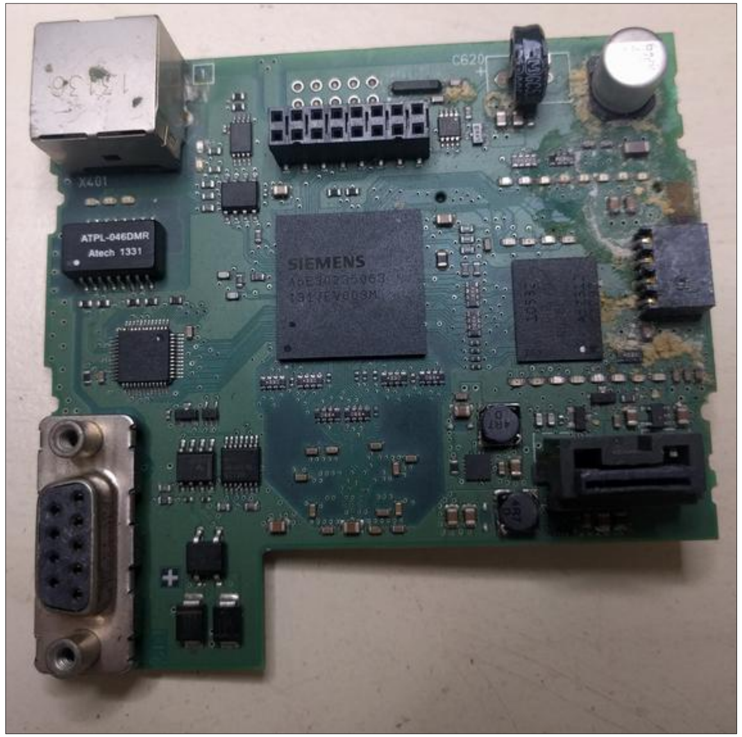
The second batch (A5E30235063)



Dangers of supply-chains – Material Chasing

Looks not like the PCB I ordered…

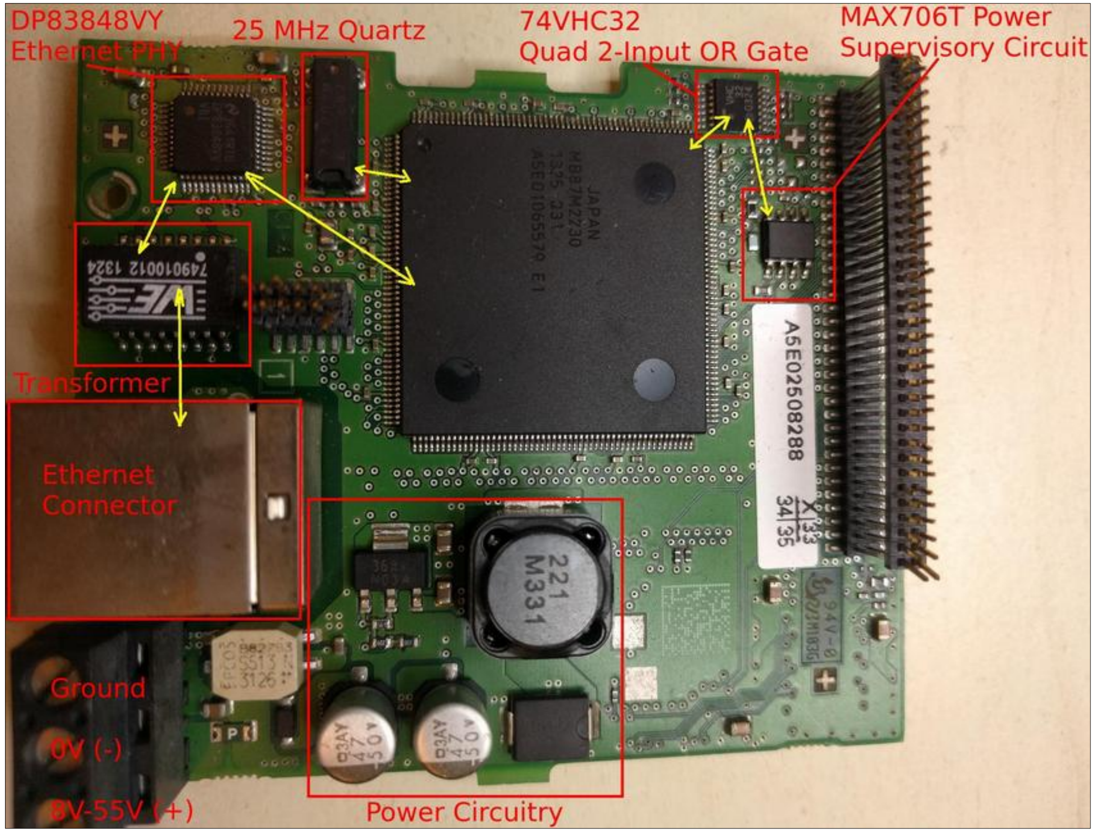


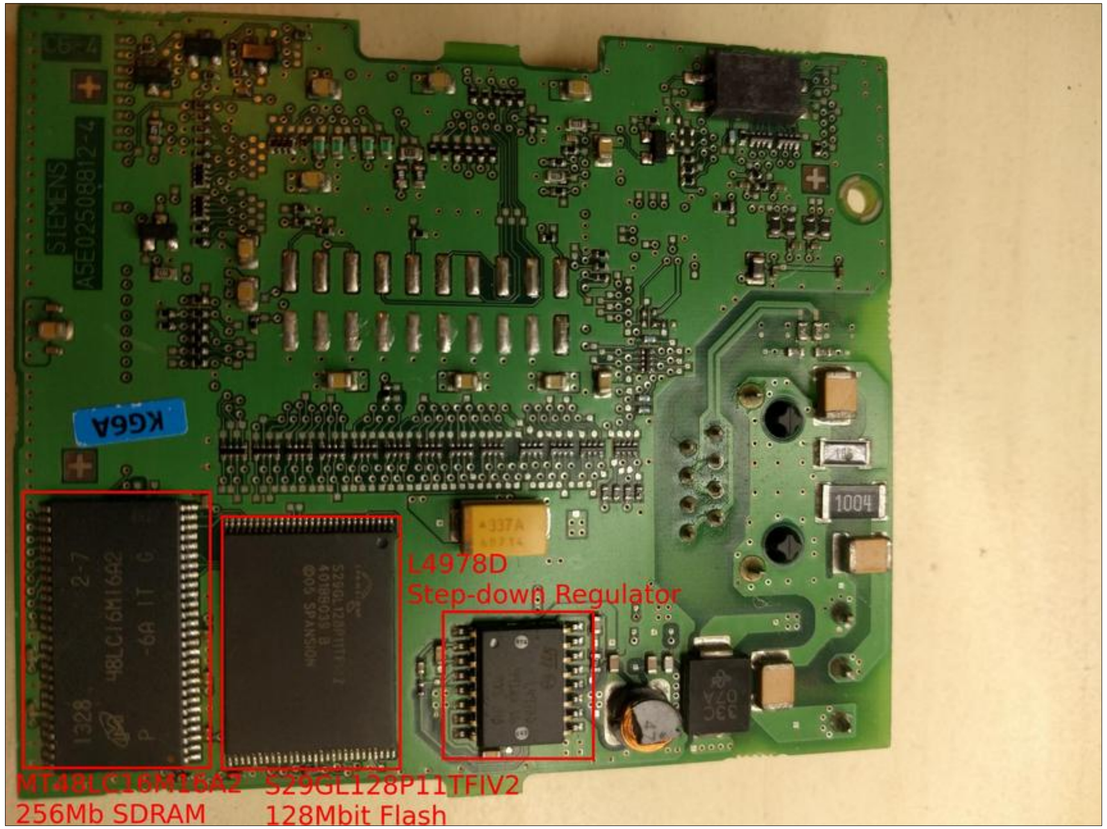


…which is similar to some profile pics on dating websites.

Reverse engineering methods – First batch of PCBs

Collecting datasheets by looking at the PCB:

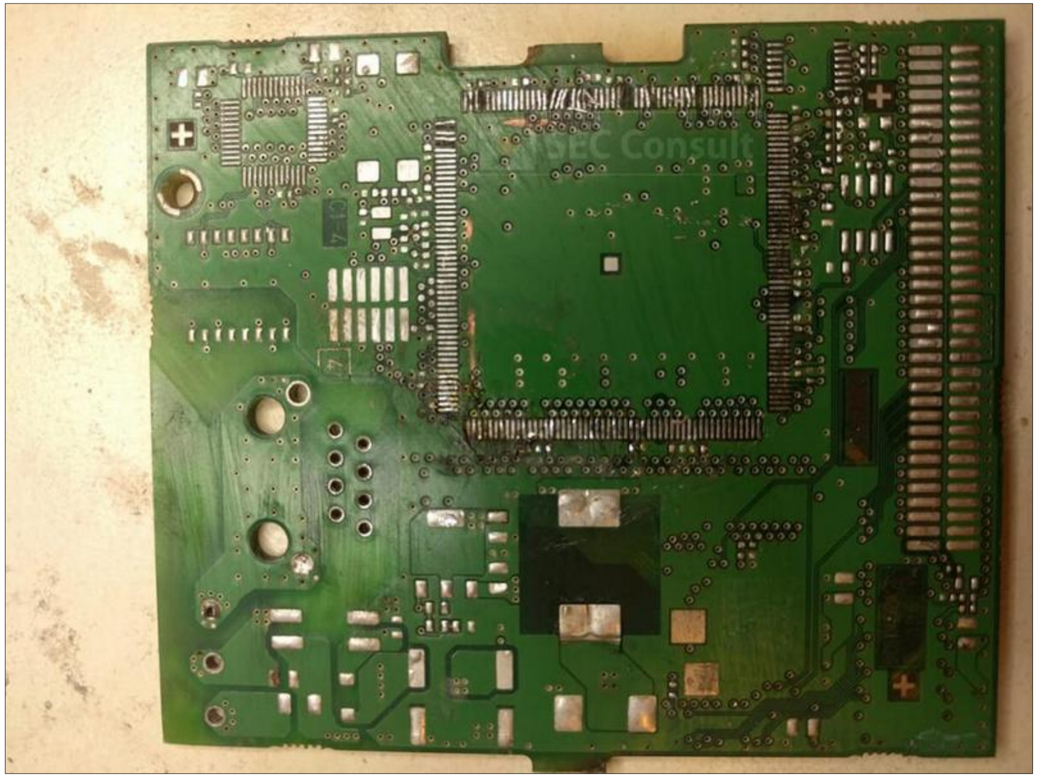


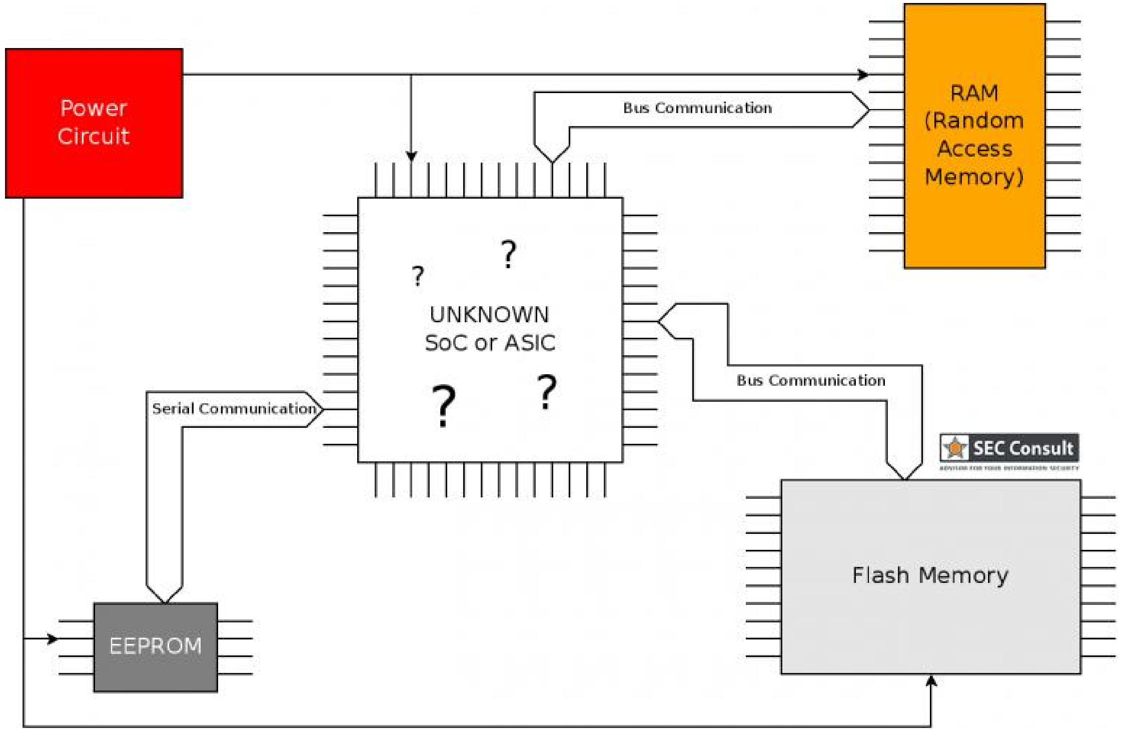


Reverse engineering methods – Deductive reasoning

Remove all parts from one PCB to be able to track all connections.

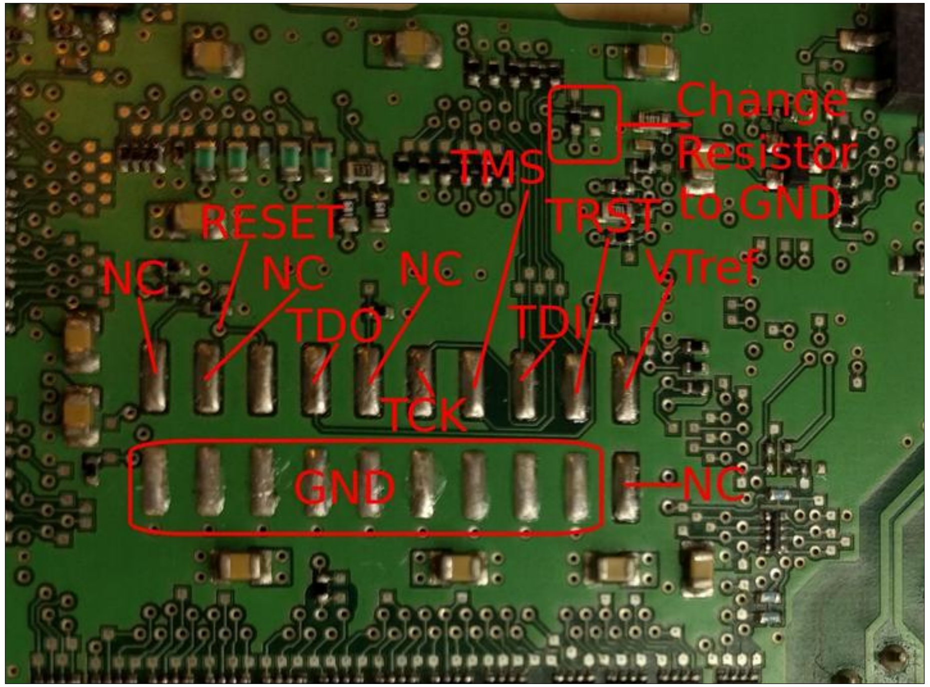
Determining the obvious Vdd pins.

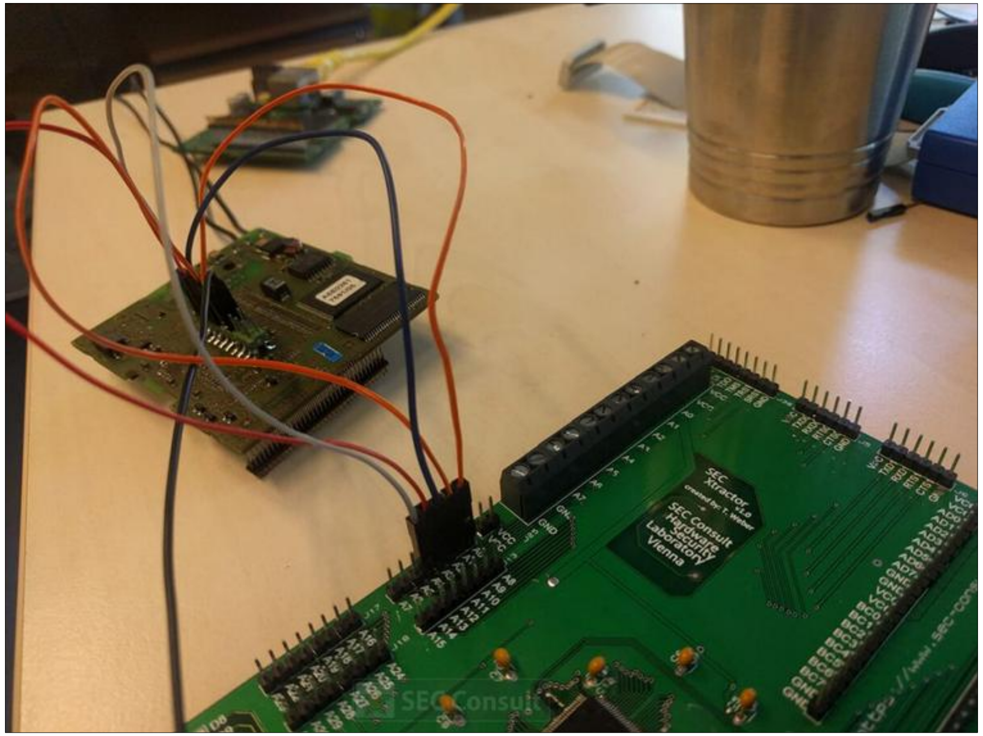




Reverse engineering methods – Deductive reasoning

Actively probing for debug interfaces, in this case for JTAG. Some pins were excluded from this test because of the prior step.



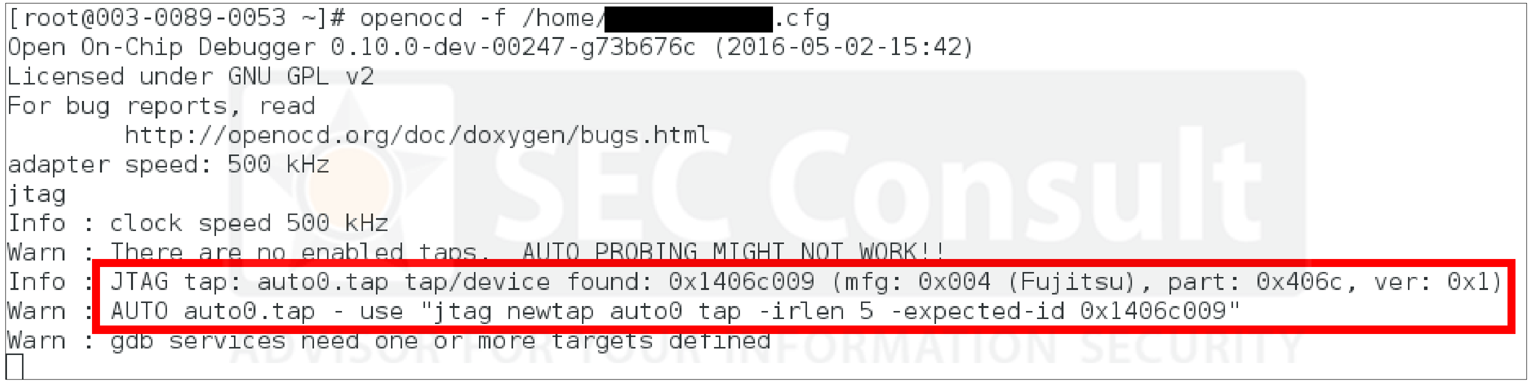


These pins are often pulled to Vdd by using a pull-up resistor!

They may be close to SPI or UART!

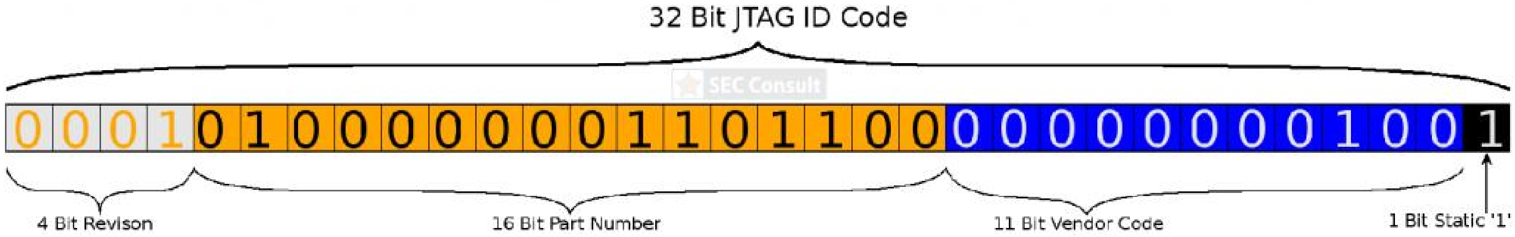
Reverse engineering methods – Deductive reasoning

After finding such a JTAG port, the ID-code can be fetched and interpreted:



```
[root@003-0089-0053 ~]# openocd -f /home/[redacted].cfg
Open On-Chip Debugger 0.10.0-dev-00247-g73b676c (2016-05-02-15:42)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
adapter speed: 500 kHz
jtag
Info : clock speed 500 kHz
Warn : There are no enabled taps.  AUTO PROBING MIGHT NOT WORK!!
Info : JTAG tap: auto0.tap tap/device found: 0x1406c009 (mfg: 0x004 (Fujitsu), part: 0x406c, ver: 0x1)
Warn : AUTO auto0.tap - use "jtag newtap auto0 tap -irlen 5 -expected-id 0x1406c009"
Warn : gdb services need one or more targets defined
```

Refer to JEDEC “JEP106AV”!



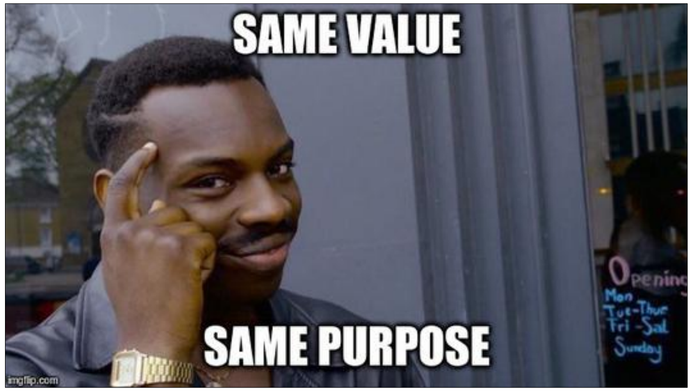
Reverse engineering methods – Deductive reasoning

Besides JTAG, another challenging task is the detection of reset pins (**SRST** not **TRST**).

Common design patterns can help here, e.g.:

- The reset pin might be **bound to Vdd by** the **same** pull-up **resistor value** like all other ICs.

- The reset pin might be **switched** from Vdd to GND **by using a transistor**.



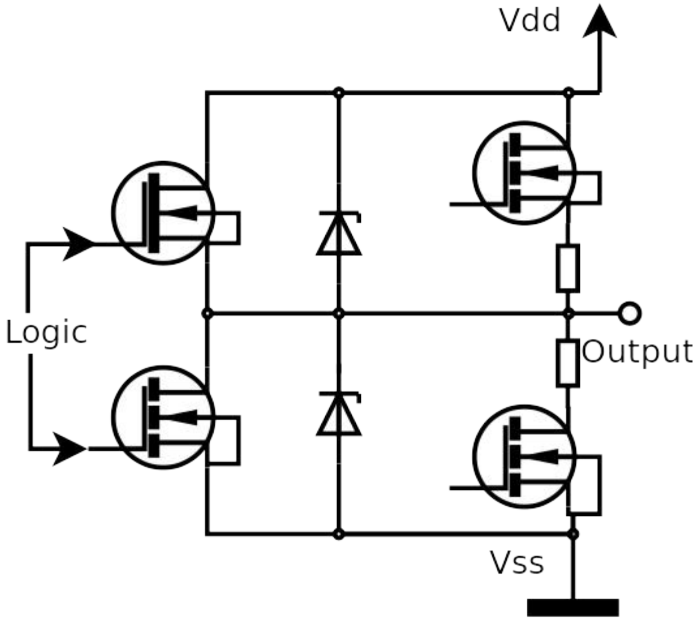
→ These two cases are very likely!

Quick test: Short circuit the pin to GND
(be sure to not kill the power IC)
BINGO! → When the CPU jumps to its reset vector!

Reverse engineering methods – Deductive reasoning

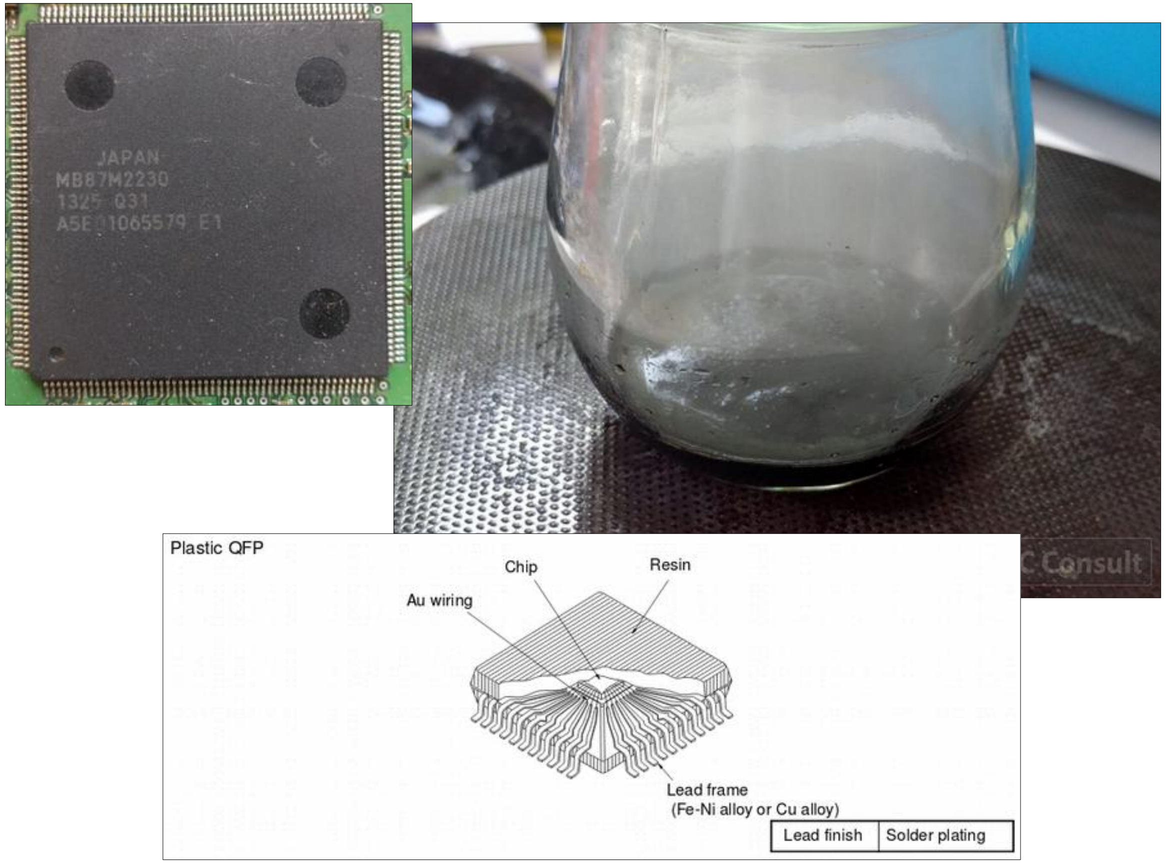
Whether a pin is an input, output or inout pin, can be determined by measuring the resistance of a pin. This is different from chip to chip and can be used as last step to identify the possible purpose of a pin.

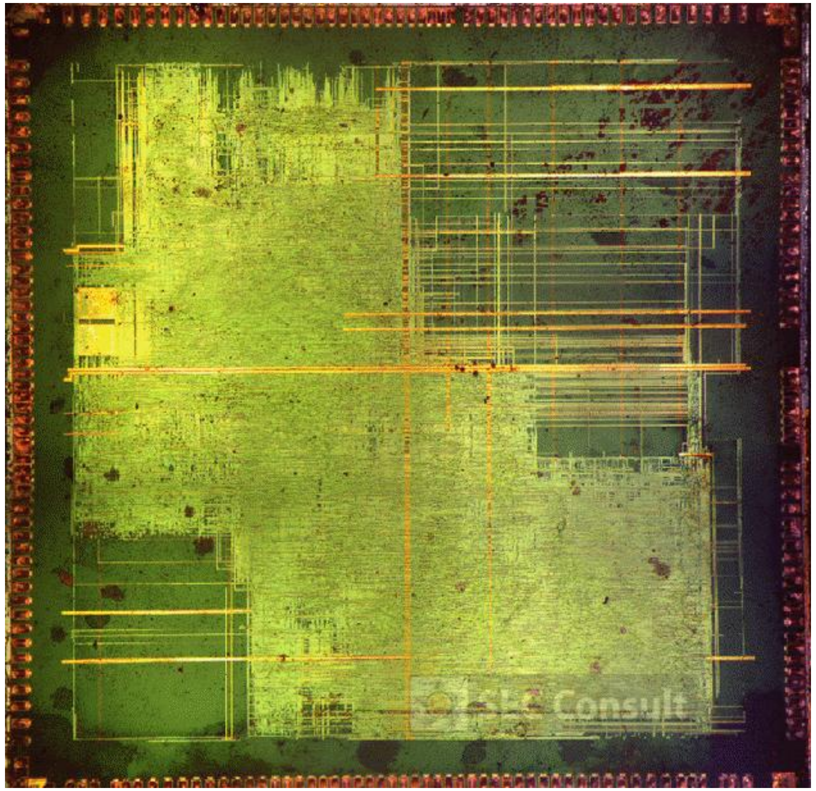
For example: Output logic



Deeper insights

Delicious cooking in sulfuric acid!





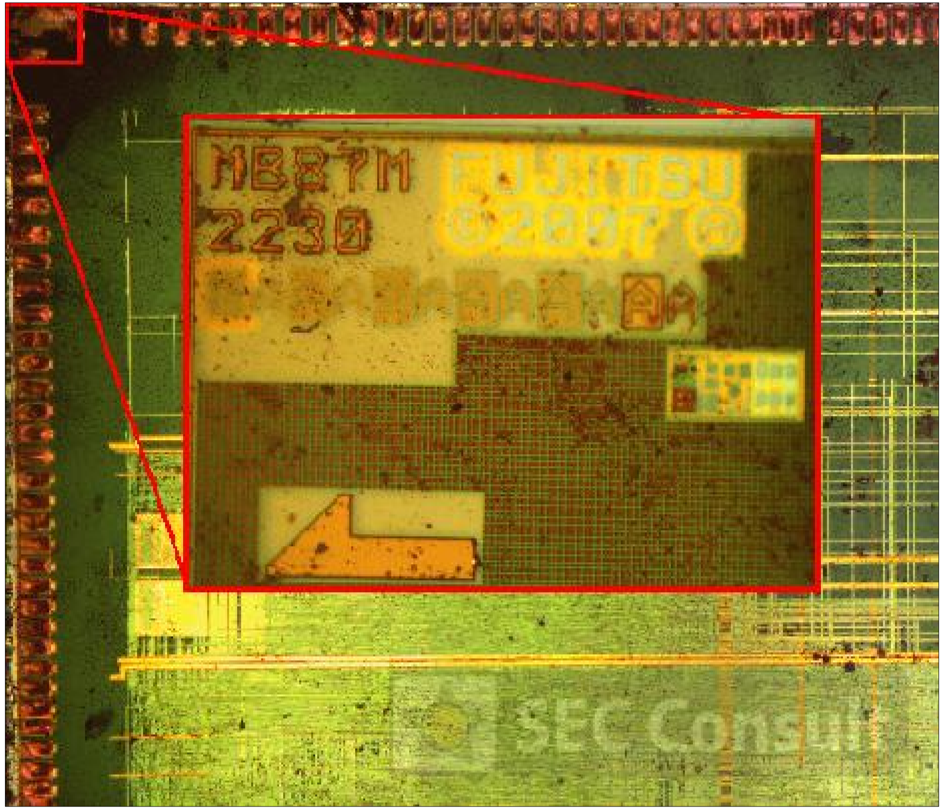
Don’t do that at home!

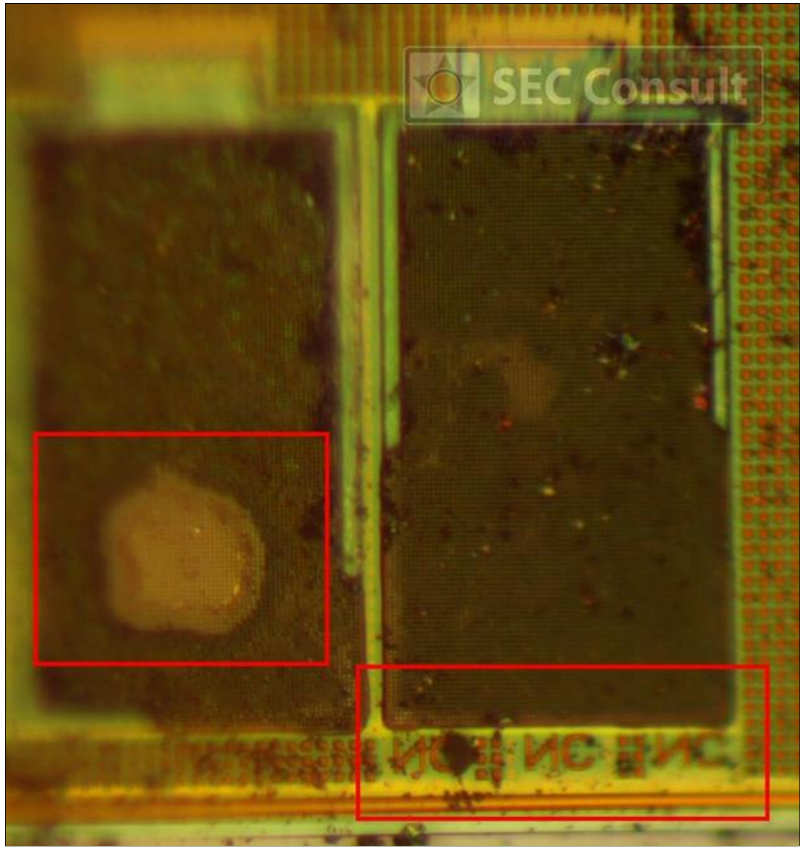
https://www.fujitsu.com/downloads/MICRO/fma/pdfmcu/packageguide-contents-x1.pdf

Deeper insights

The labels on the bare die sometimes reveal important information.

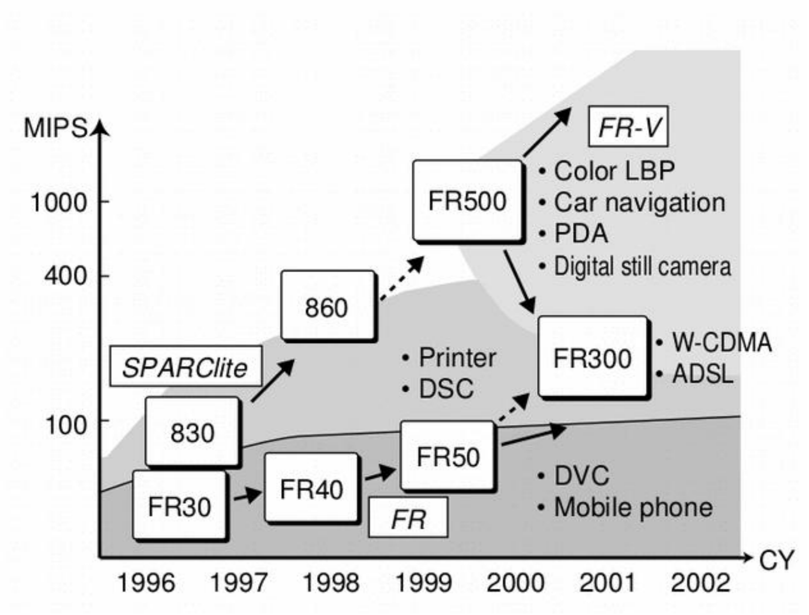
For this chip, it was good to verify the JTAG output – it was designed by Fujitsu.



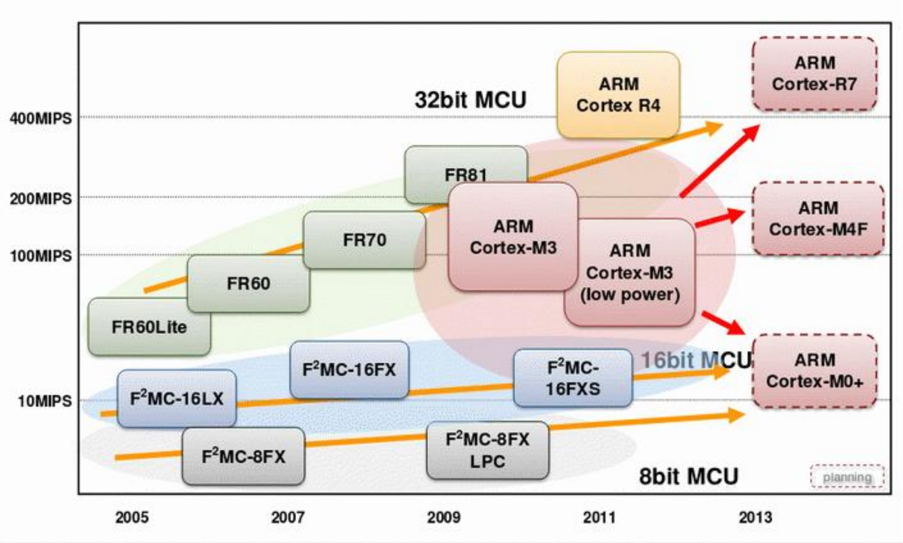


Deeper insights – Digging through the literature

So many possibilities!



https://www.fujitsu.com/global/documents/about/resources/publications/fstj /archives/vol36-1/paper06.pdf

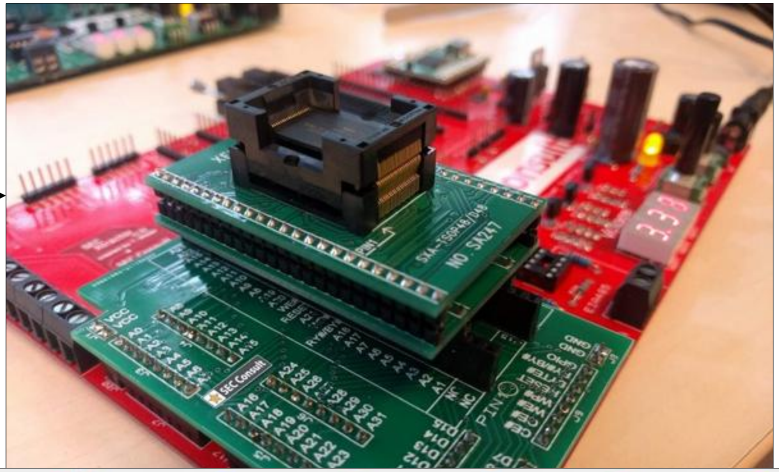


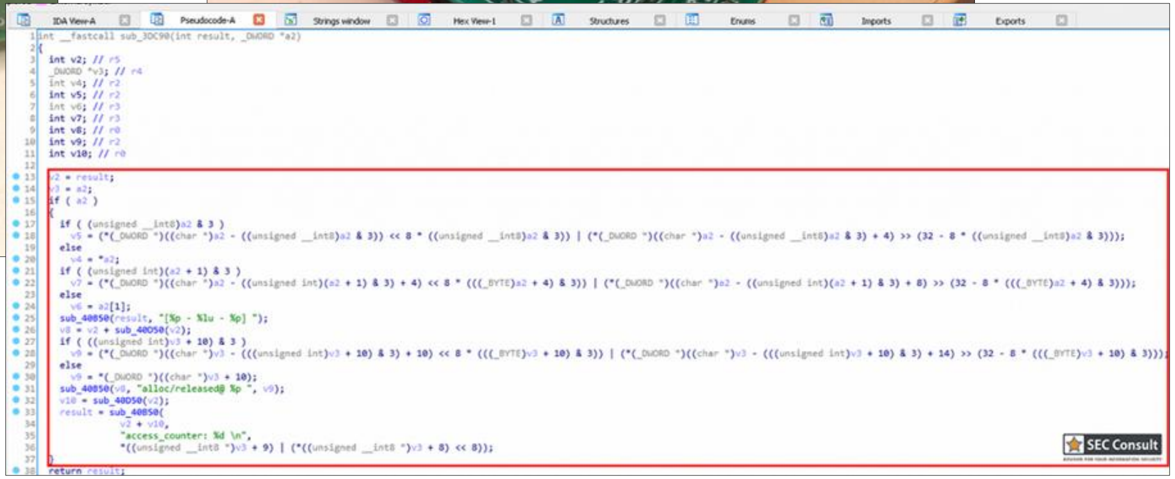
http://docplayer.net/4207609-Right-sized-solutions-for-embedded-applications.html

Other MB8xMxxxx chips have ARC Tangent processors, or Fujitsu RISC (FR). It can also be F²MC….

Deeper insights

Removing the flash memory and reading out its content always helps.

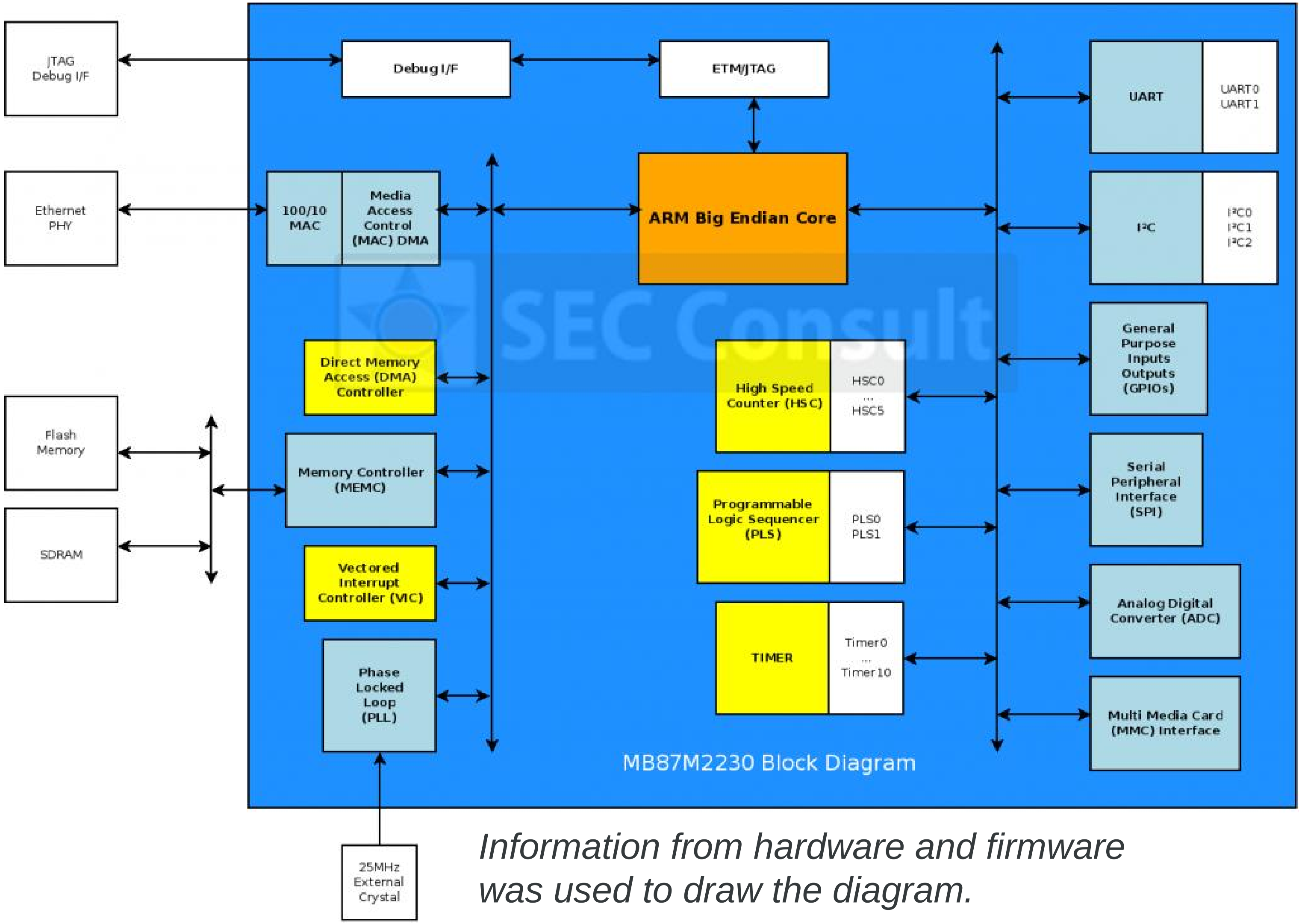




```
int __fastcall sub_3DC90(int result, _DWORD *a2)
{
  int v2; // r5
  _DWORD *v3; // r4
  int v4; // r2
  int v5; // r2
  int v6; // r3
  int v7; // r3
  int v8; // r0
  int v9; // r2
  int v10; // r0

  v2 = result;
  v3 = a2;
  if ( a2 )
  {
    if ( (unsigned __int8)a2 & 3 )
      v5 = (*(_DWORD *)((char *)a2 - ((unsigned __int8)a2 & 3)) << 8 * ((unsigned __int8)a2 & 3)) | (*(_DWORD *)((char *)a2 - ((unsigned __int8)a2 & 3) + 4) >> (32 - 8 * ((unsigned __int8)a2 & 3)));
    else
      v4 = *a2;
    if ( (unsigned int)(a2 + 1) & 3 )
      v7 = (*(_DWORD *)((char *)a2 - ((unsigned int)(a2 + 1) & 3) + 4) << 8 * (((_BYTE)a2 + 4) & 3)) | (*(_DWORD *)((char *)a2 - ((unsigned int)(a2 + 1) & 3) + 8) >> (32 - 8 * (((_BYTE)a2 + 4) & 3)));
    else
      v6 = a2[1];
    sub_40850(result, "[%p - %lu - %p] ");
    v8 = v2 + sub_40D50(v2);
    if ( ((unsigned int)v3 + 10) & 3 )
      v9 = (*(_DWORD *)((char *)v3 - (((unsigned int)v3 + 10) & 3) + 10) << 8 * (((_BYTE)v3 + 10) & 3)) | (*(_DWORD *)((char *)v3 - (((unsigned int)v3 + 10) & 3) + 14) >> (32 - 8 * (((_BYTE)v3 + 10) & 3)));
    else
      v9 = *(_DWORD *)((char *)v3 + 10);
    sub_40850(v8, "alloc/released@ %p ", v9);
    v10 = sub_40D50(v2);
    result = sub_40850(
               v2 + v10,
               "access_counter: %d \n",
               *((unsigned __int8 *)v3 + 9) | (*((unsigned __int8 *)v3 + 8) << 8));
  }
  return result;
```

Deeper insights

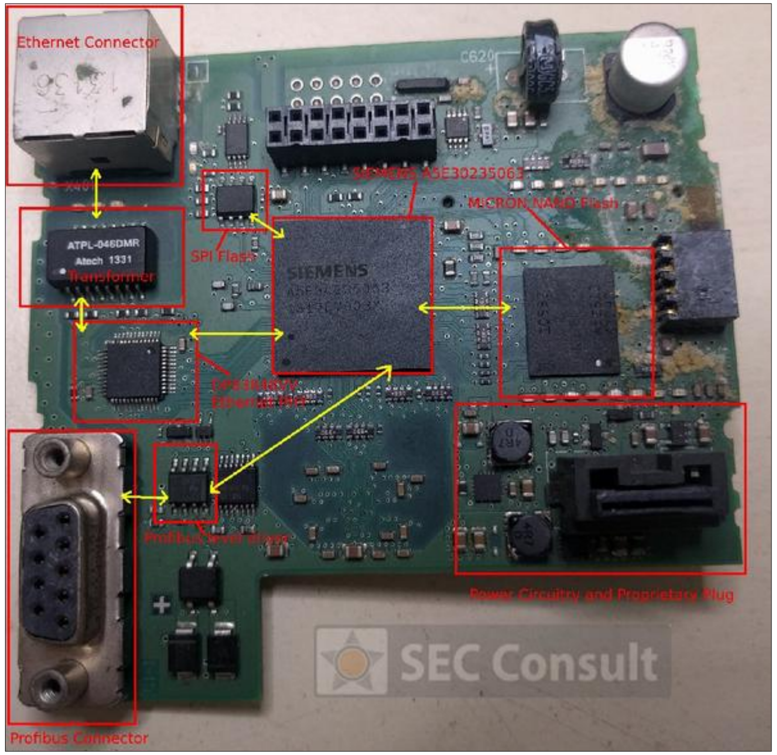


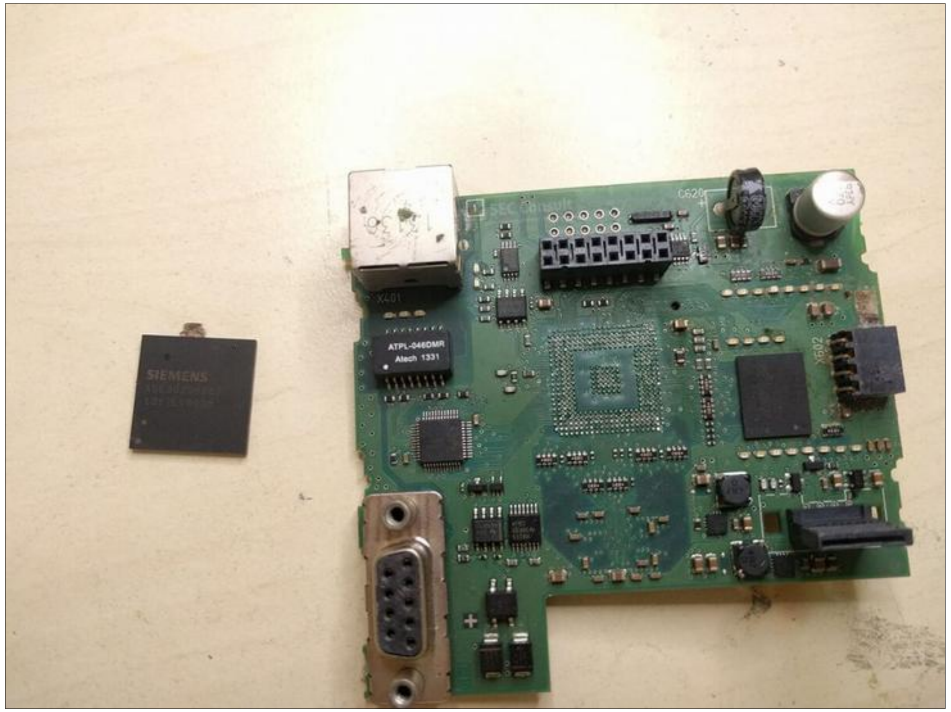
Information from hardware and firmware was used to draw the diagram.

By combining the information of the used CPU core, the year and the available IP cores from Fujitsu at that time we can be pretty sure that ARM926/ARM946 is used.

Deeper insights – Second batch

The second batch of PCBs can be analyzed in the same way as the first one.

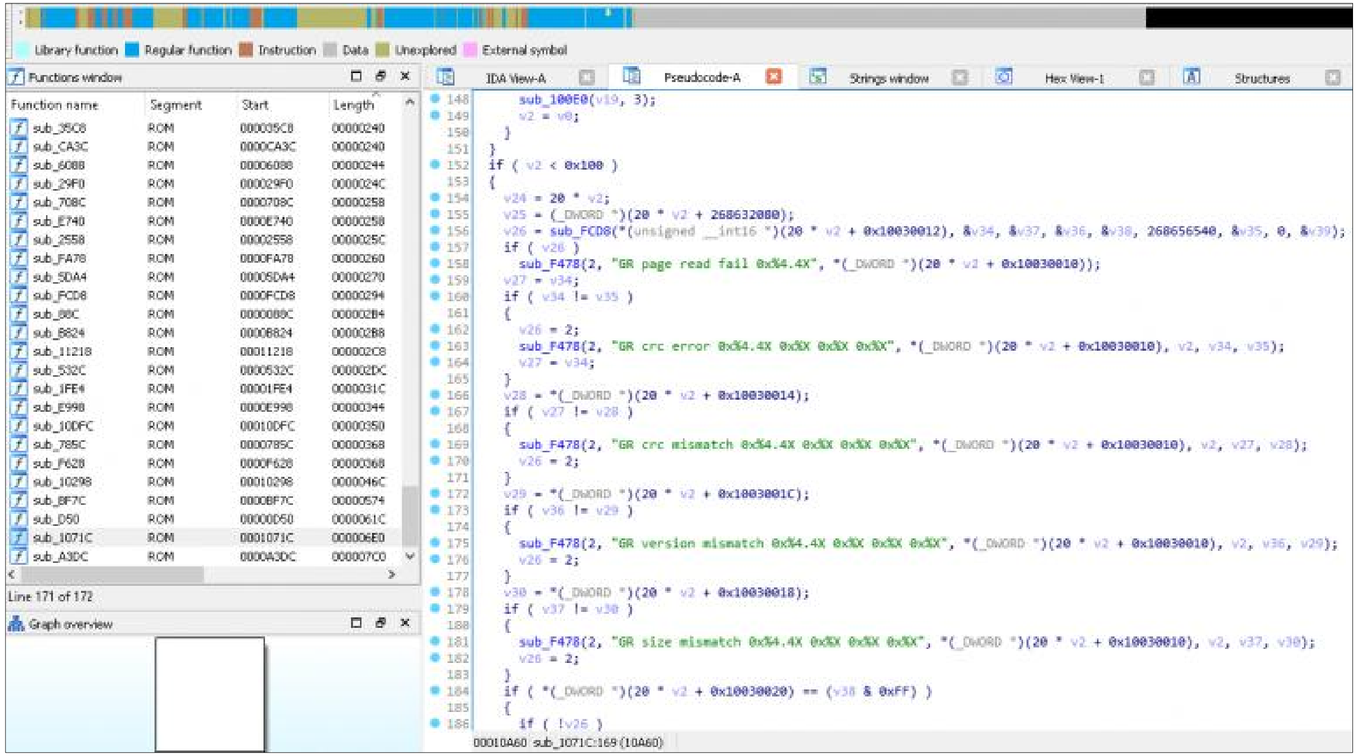




The different architectures of SPI flash + NAND flash were one of the first observations.

Deeper insights – Second batch

The bootloader, which is located at the SPI flash memory was dumped and loaded into IDA Pro:



Function name	Segment	Start	Length
sub_35C8	ROM	000035C8	00000240
sub_CA3C	ROM	0000CA3C	00000240
sub_6088	ROM	00006088	00000244
sub_29F0	ROM	000029F0	0000024C
sub_708C	ROM	0000708C	00000258
sub_E740	ROM	0000E740	00000258
sub_2558	ROM	00002558	0000025C
sub_FA78	ROM	0000FA78	00000260
sub_5DA4	ROM	00005DA4	00000270
sub_FCD8	ROM	0000FCD8	00000294
sub_88C	ROM	0000088C	000002B4
sub_B824	ROM	0000B824	000002B8
sub_11218	ROM	00011218	000002C8
sub_532C	ROM	0000532C	000002DC
sub_1FE4	ROM	00001FE4	0000031C
sub_E998	ROM	0000E998	00000344
sub_10DFC	ROM	00010DFC	00000350
sub_785C	ROM	0000785C	00000368
sub_F628	ROM	0000F628	00000368
sub_10298	ROM	00010298	0000046C
sub_BF7C	ROM	0000BF7C	00000574
sub_D50	ROM	00000D50	0000061C
sub_1071C	ROM	0001071C	000006E0
sub_A3DC	ROM	0000A3DC	000007C0

Line 171 of 172

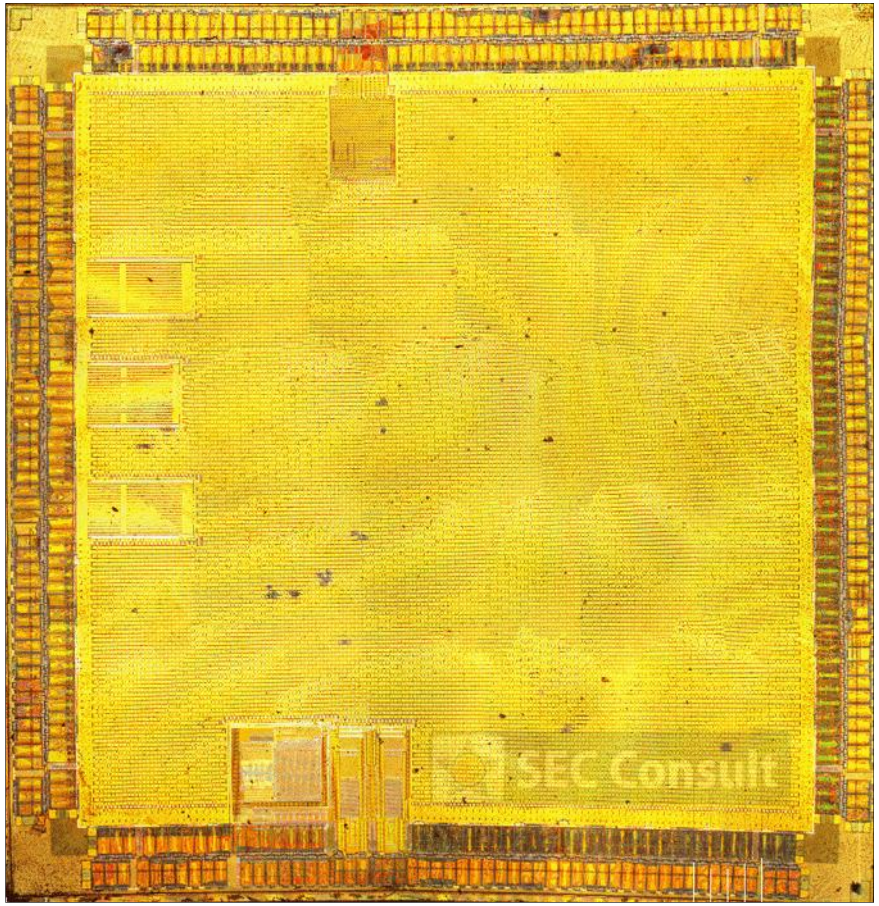
```
      sub_100E0(v19, 3);
      v2 = v0;
    }
  }
  if ( v2 < 0x100 )
  {
    v24 = 20 * v2;
    v25 = (_DWORD *)(20 * v2 + 268632080);
    v26 = sub_FCD8(*(unsigned __int16 *)(20 * v2 + 0x10030012), &v34, &v37, &v36, &v38, 268656540, &v35, 0, &v39);
    if ( v26 )
      sub_F478(2, "GR page read fail 0x%4.4X", *(_DWORD *)(20 * v2 + 0x10030010));
    v27 = v34;
    if ( v34 != v35 )
    {
      v26 = 2;
      sub_F478(2, "GR crc error 0x%4.4X 0x%X 0x%X 0x%X", *(_DWORD *)(20 * v2 + 0x10030010), v2, v34, v35);
      v27 = v34;
    }
    v28 = *(_DWORD *)(20 * v2 + 0x10030014);
    if ( v27 != v28 )
    {
      sub_F478(2, "GR crc mismatch 0x%4.4X 0x%X 0x%X 0x%X", *(_DWORD *)(20 * v2 + 0x10030010), v2, v27, v28);
      v26 = 2;
    }
    v29 = *(_DWORD *)(20 * v2 + 0x1003001C);
    if ( v36 != v29 )
    {
      sub_F478(2, "GR version mismatch 0x%4.4X 0x%X 0x%X 0x%X", *(_DWORD *)(20 * v2 + 0x10030010), v2, v36, v29);
      v26 = 2;
    }
    v30 = *(_DWORD *)(20 * v2 + 0x10030018);
    if ( v37 != v30 )
    {
      sub_F478(2, "GR size mismatch 0x%4.4X 0x%X 0x%X 0x%X", *(_DWORD *)(20 * v2 + 0x10030010), v2, v37, v30);
      v26 = 2;
    }
    if ( *(_DWORD *)(20 * v2 + 0x10030020) == (v38 & 0xFF) )
    {
      if ( !v26 )
```

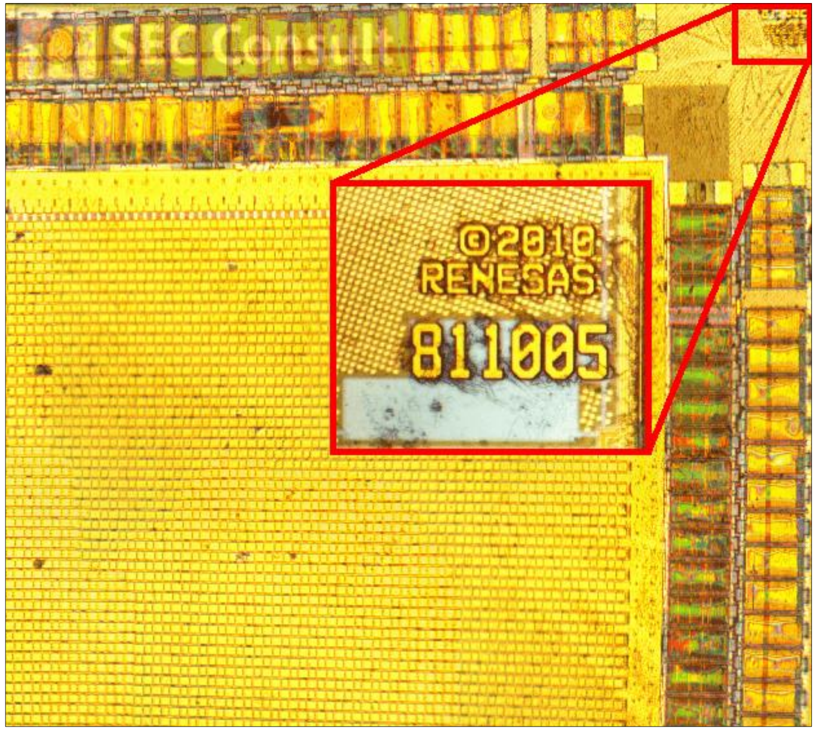
00010A60 sub_1071C:169 (10A60)

Most strings were referenced immediately, ARM big endian was used here too.

Deeper insights – Second batch

It turned out that the newer chip (A5E30235063) was designed by Renesas.

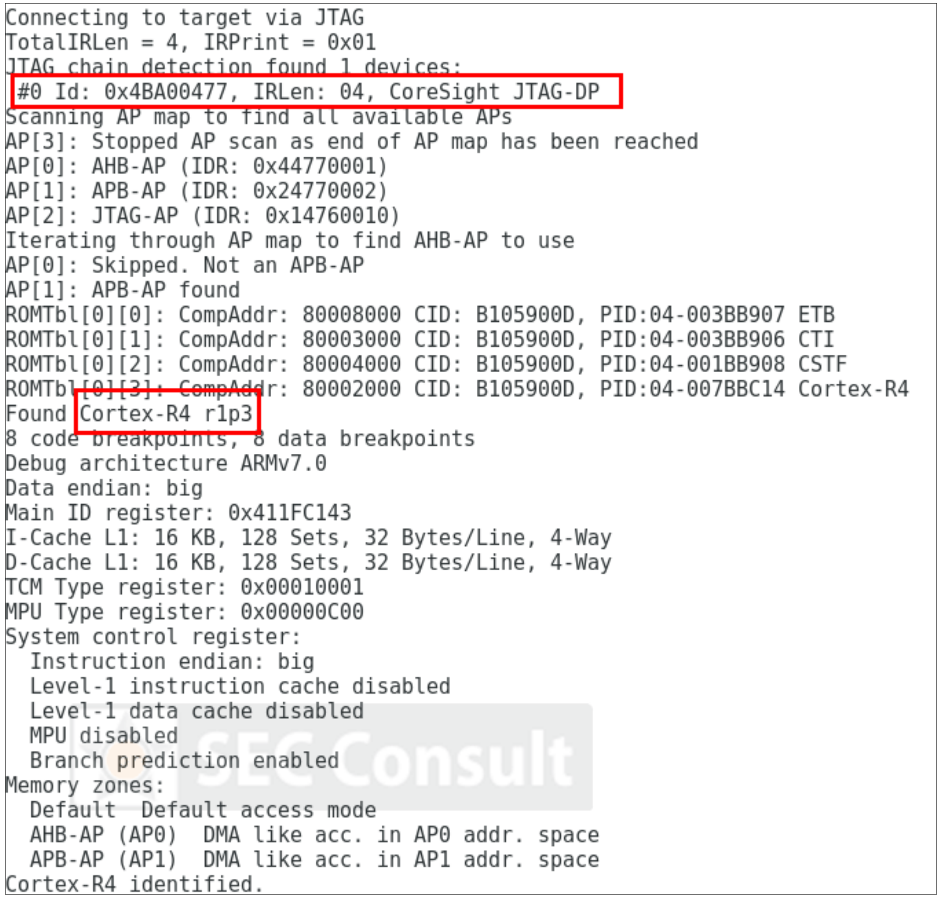




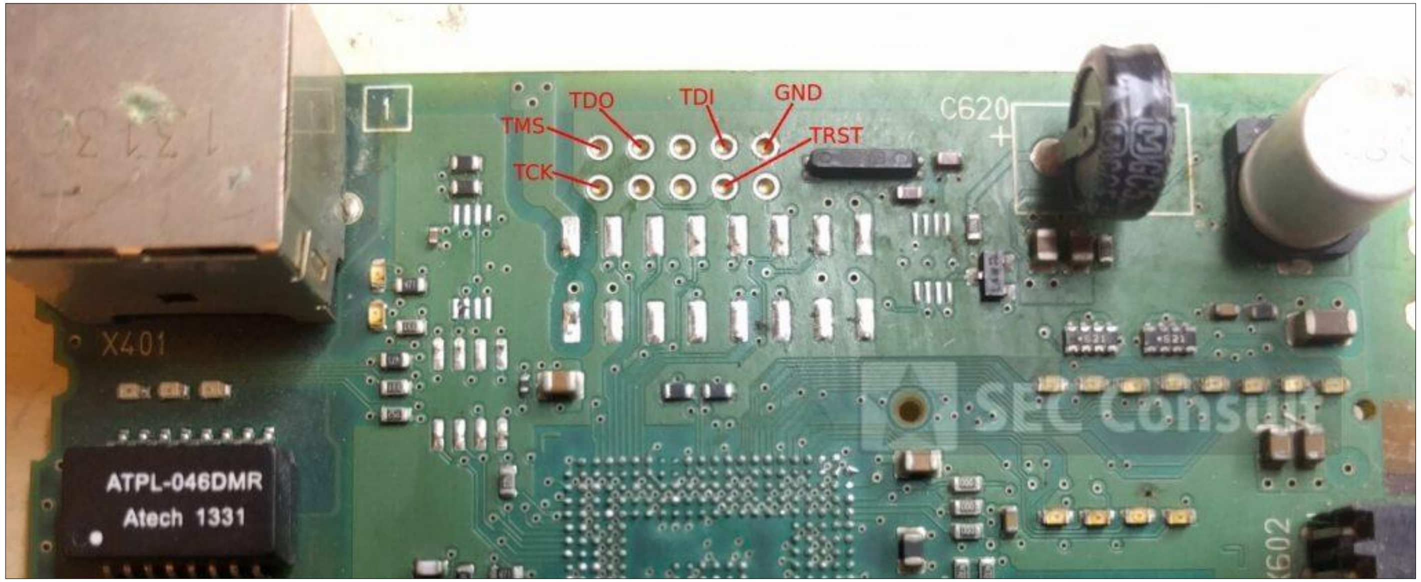
There are some similarities to ERTEC 200P/400 (Siemens/Renesas).

Deeper insights – Second batch

By brute forcing the 10-pin header of the PCB a JTAG port was found!



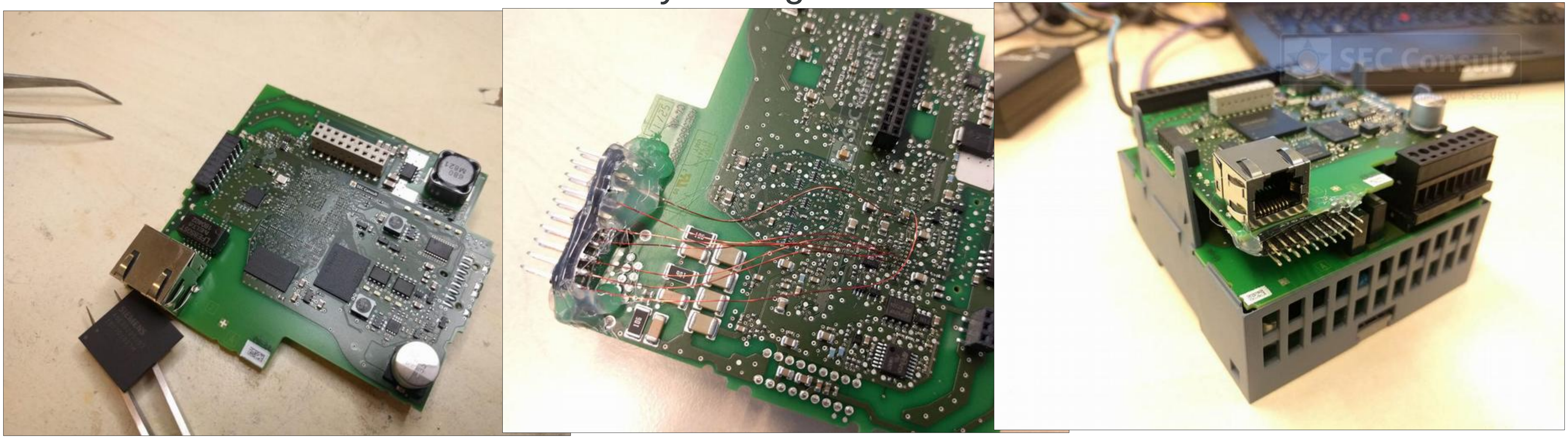
```
Connecting to target via JTAG
TotalIRLen = 4, IRPrint = 0x01
JTAG chain detection found 1 devices:
 #0 Id: 0x4BA00477, IRLen: 04, CoreSight JTAG-DP
Scanning AP map to find all available APs
AP[3]: Stopped AP scan as end of AP map has been reached
AP[0]: AHB-AP (IDR: 0x44770001)
AP[1]: APB-AP (IDR: 0x24770002)
AP[2]: JTAG-AP (IDR: 0x14760010)
Iterating through AP map to find AHB-AP to use
AP[0]: Skipped. Not an APB-AP
AP[1]: APB-AP found
ROMTbl[0][0]: CompAddr: 80008000 CID: B105900D, PID:04-003BB907 ETB
ROMTbl[0][1]: CompAddr: 80003000 CID: B105900D, PID:04-003BB906 CTI
ROMTbl[0][2]: CompAddr: 80004000 CID: B105900D, PID:04-001BB908 CSTF
ROMTbl[0][3]: CompAddr: 80002000 CID: B105900D, PID:04-007BBC14 Cortex-R4
Found Cortex-R4 r1p3
8 code breakpoints, 8 data breakpoints
Debug architecture ARMv7.0
Data endian: big
Main ID register: 0x411FC143
I-Cache L1: 16 KB, 128 Sets, 32 Bytes/Line, 4-Way
D-Cache L1: 16 KB, 128 Sets, 32 Bytes/Line, 4-Way
TCM Type register: 0x00010001
MPU Type register: 0x00000C00
System control register:
  Instruction endian: big
  Level-1 instruction cache disabled
  Level-1 data cache disabled
  MPU disabled
  Branch prediction enabled
Memory zones:
  Default  Default access mode
  AHB-AP (AP0)  DMA like acc. in AP0 addr. space
  APB-AP (AP1)  DMA like acc. in AP1 addr. space
Cortex-R4 identified.
```



ARM Cortex R4 was identified. Now it was easy to trace the connections back to the chip!

Deeper insights – Second batch

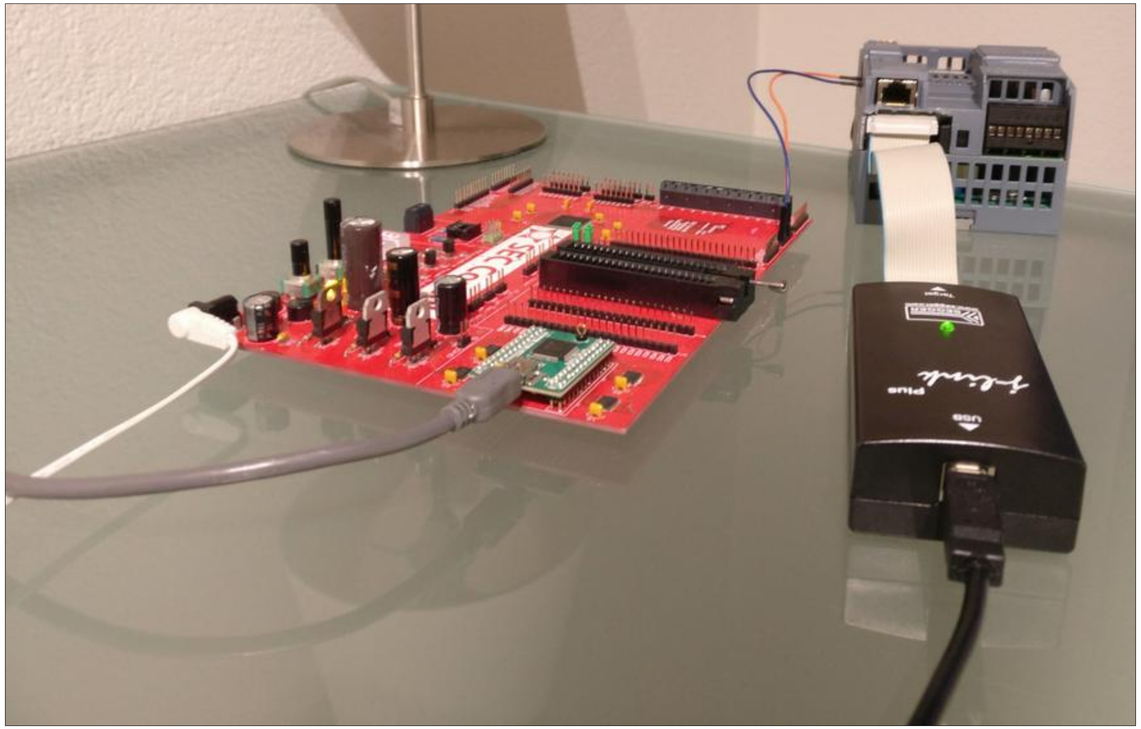
After removing the chip of an original S7 1211C, the traces can be followed back to the backside. JTAG can be enabled by adding an additional header to the PCB.

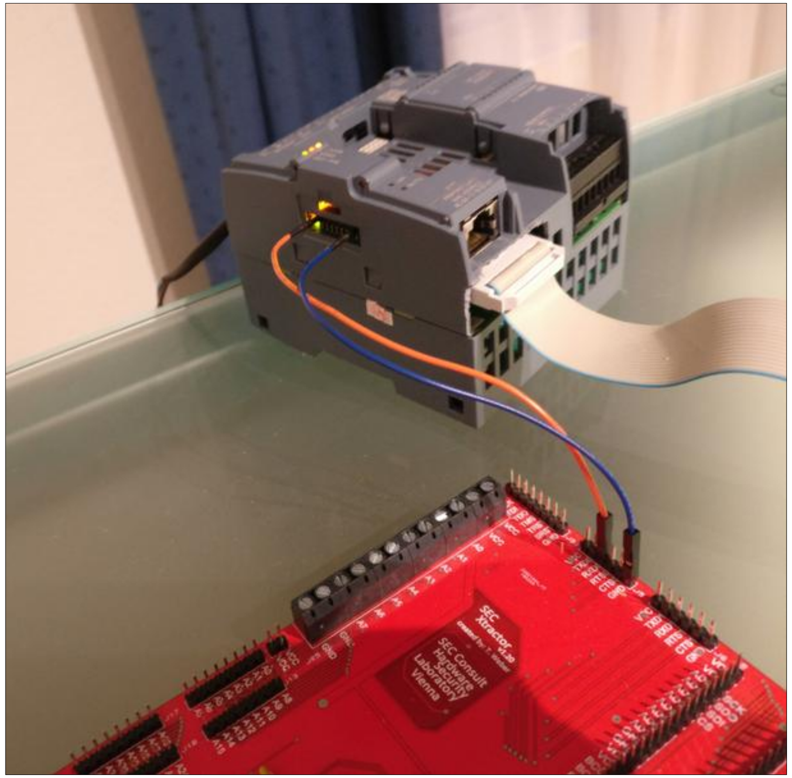


Beware, when you attach the debugger! It seems that Siemens have implemented a hardware module for deleting the flash memory when the CPU is stopped!!!

Demo time!

To provide a proof of concept, a small assembly program was written and uploaded to the PLC via the JTAG interface.





Special thanks goes to Dr. Ali Abbasi for providing me the UART MMIO address.

https://www.syssec.ruhr-uni-bochum.de/chair/staff/aliabbasi/

Fun Fact

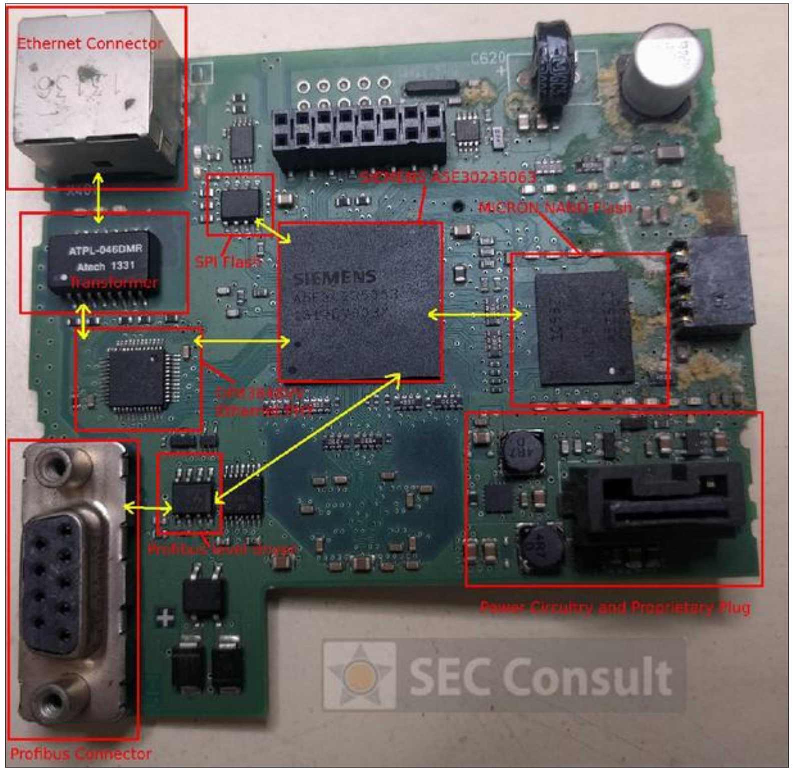
Few days before publishing our research, we received the following statement from Siemens:

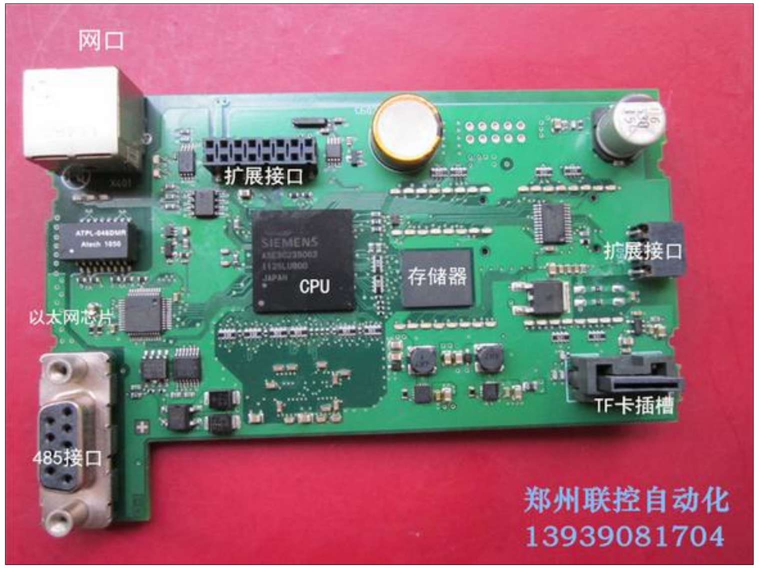
> *“The boards purchased by SEC Consult were not development boards but previously used or refurbished boards from Siemens devices. Siemens does not see a supply chain leak.”*

As it turns out, I was looking at boards from another series. The seller from Taobao fooled me. He offered boards from the older **S7-200 SMART** series labeled as **S7-1200** series … but no bad feelings: the board had **JTAG**!

Fun Fact

Can you spot the similarities?





S7-200 SMART http://www.plcweixiu.com/news/html/390.html

Q&A

Thank you!

Find the full blogpost here:

https://sec-consult.com/en/blog/2019/02/reverse-engineering-architecture-pinout-plc/