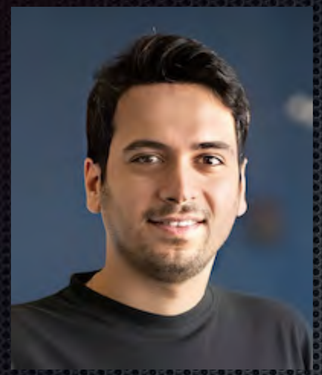
Doors of Durin:
The Veiled Gate to Siemens S7 Silicon

Ali Abbasi, Tobias Scharnowski, Thorsten Holz

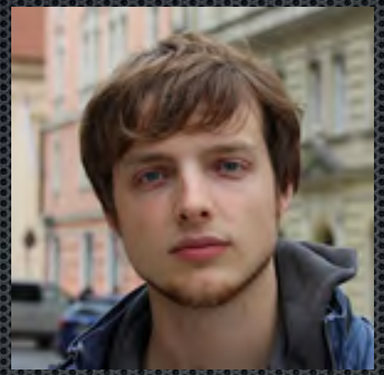
Chair for Systems Security
Ruhr-University Bochum, Germany

Who We Are?



Ali Abbasi
Researcher at SYSSEC RUB

404 Visa Not Found



Tobias Scharnowski
PhD Student at SYSSEC RUB

Thorsten Holz
Professor at SYSSEC RUB

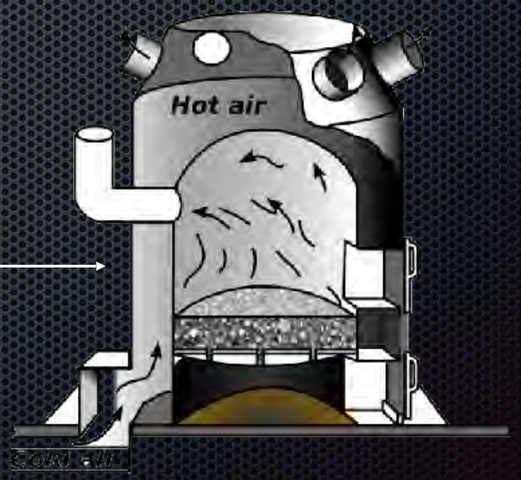
ENOTIME

Plan of Talk

- Background on Process Automation
- Overview of Siemens PLCs
- Siemens Startup Process
- Siemens Firmware Components
- The Special Access Feature
- Demos
- Conclusion and Future Works

Process Automation

Set point

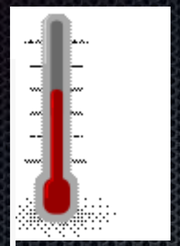


What we do with much more complex control loops?

- We use PLCs.

Programmable Logic Controllers

Sensors



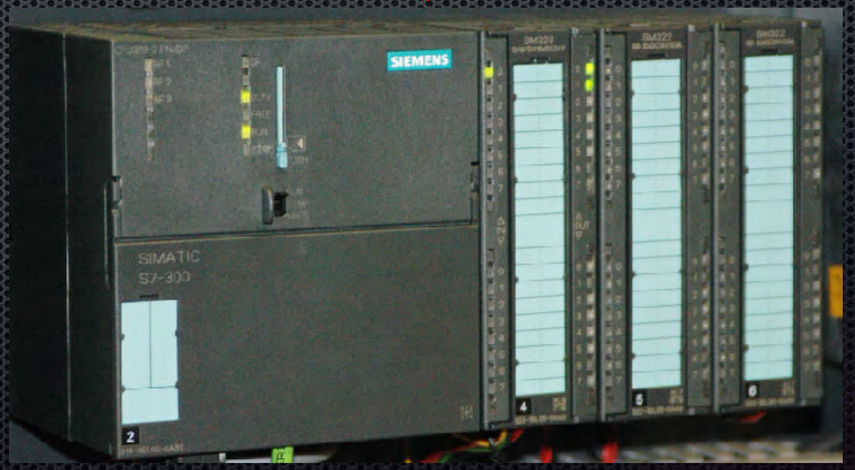
Inputs

PLC

Control Logic Executed by The Firmware

Outputs

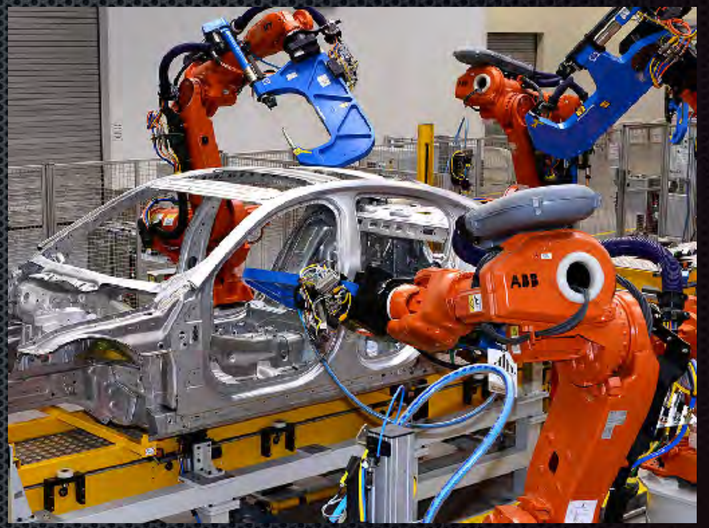
Actuators



Programmable Logic Controller

https://vecer.mk/files/article/2017/05/02/485749-saudiska-arabija-ja-kupi-najgolemata-naftena-rafinerija-vo-sad.jpg

http://www.jfwhite.com/Collateral/Images/English-US/Galleries/middleboro9115kvbreakers.jpg



https://www.roboticsbusinessreview.com/wp-content/uploads/2016/05/jaguar-factory.jpg

https://www.oilandgasproductnews.com/files/slides/locale_image/medium/0089/22183_en_16f9d_8738_honeywell-process-solutions-rtu2020-process-controller.jpg

https://selinc.com/uploadedImages/Web/Videos/Playlists/Playlist_RTAC_1280x720.png?n=63584758126000

http://www02.abb.com/global/seitp/seitp202.nsf/0/0601d25ed243cfb0c1257d7e0043e50e/$file/7184_lvl2.jpg

Recent Attacks Against ICS

Reconnaissance and weaponization of capabilities

It's happening: Publicly known cyber-physical attacks

1999

First active recon & initial intrusion attempts

2010

Planned operation to hinder Iran's enrichment program (Stuxnet)

2013

First publicly known OT recon activities (HAVEX)

2015

Ukraine power grid attack (BlackEnergy)

2016

Ukraine power grid attack (Industroyer)

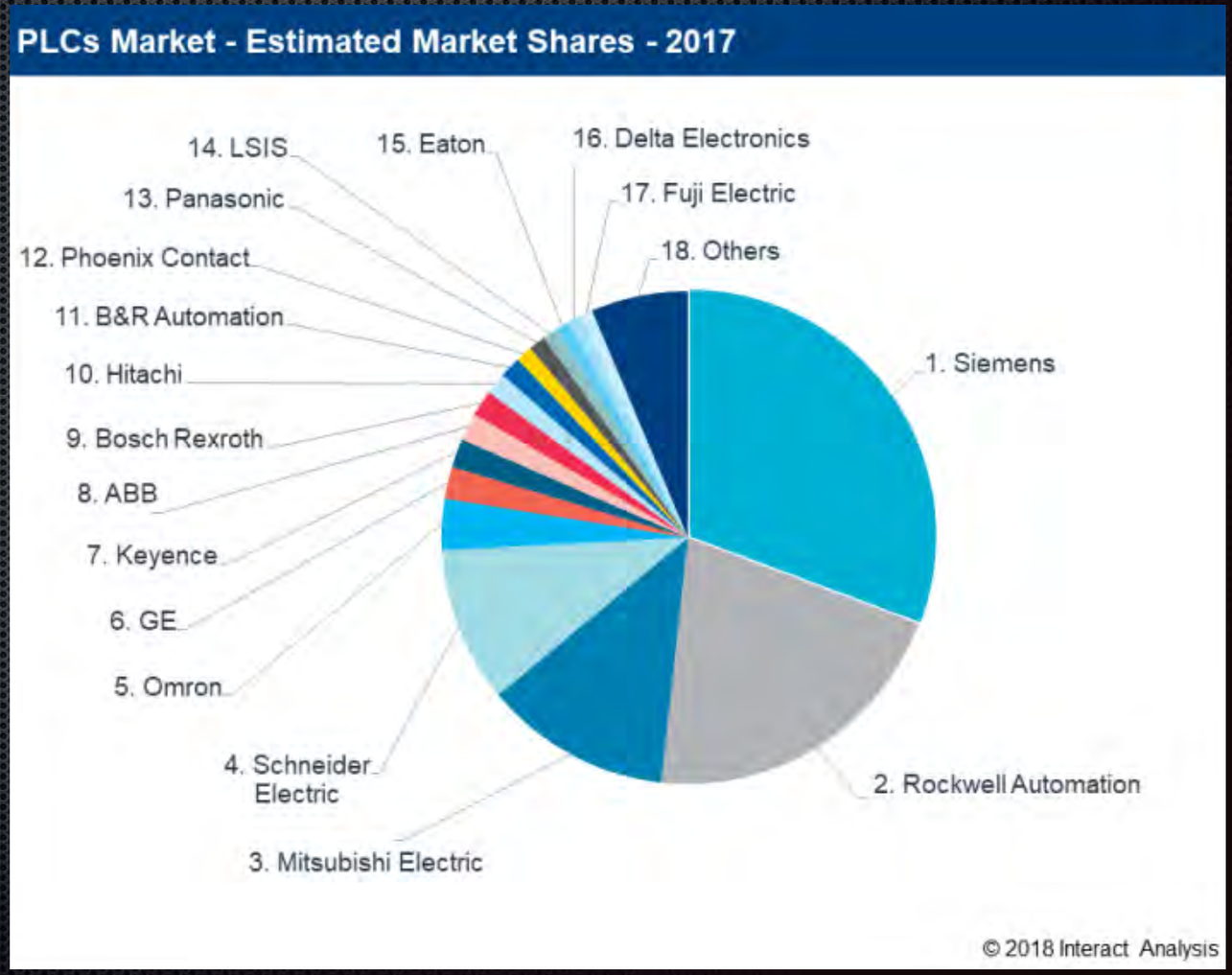
2017

TRITON

https://www.schneider-electric.com/ww/en/Images/tricon-IC-654x654.jpg

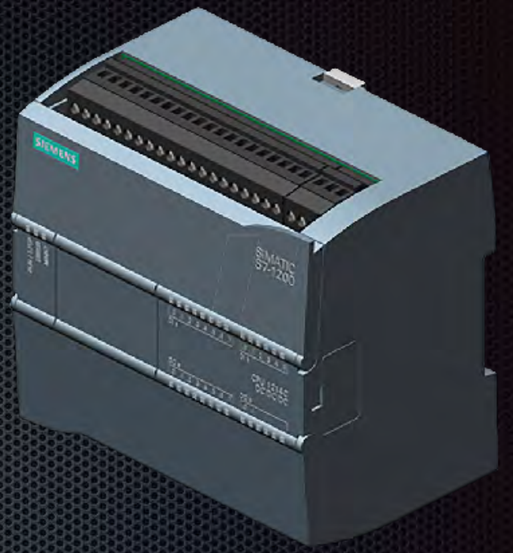
Background on Siemens PLCs Market Share

- Siemens and Rockwell Automation are PLC Market Leaders.
- Security issues in their product can have significant impact.



Source: https://www.interactanalysis.com/who-were-the-leading-vendors-of-industrial-controls-plcs-and-dcs-in-2017/

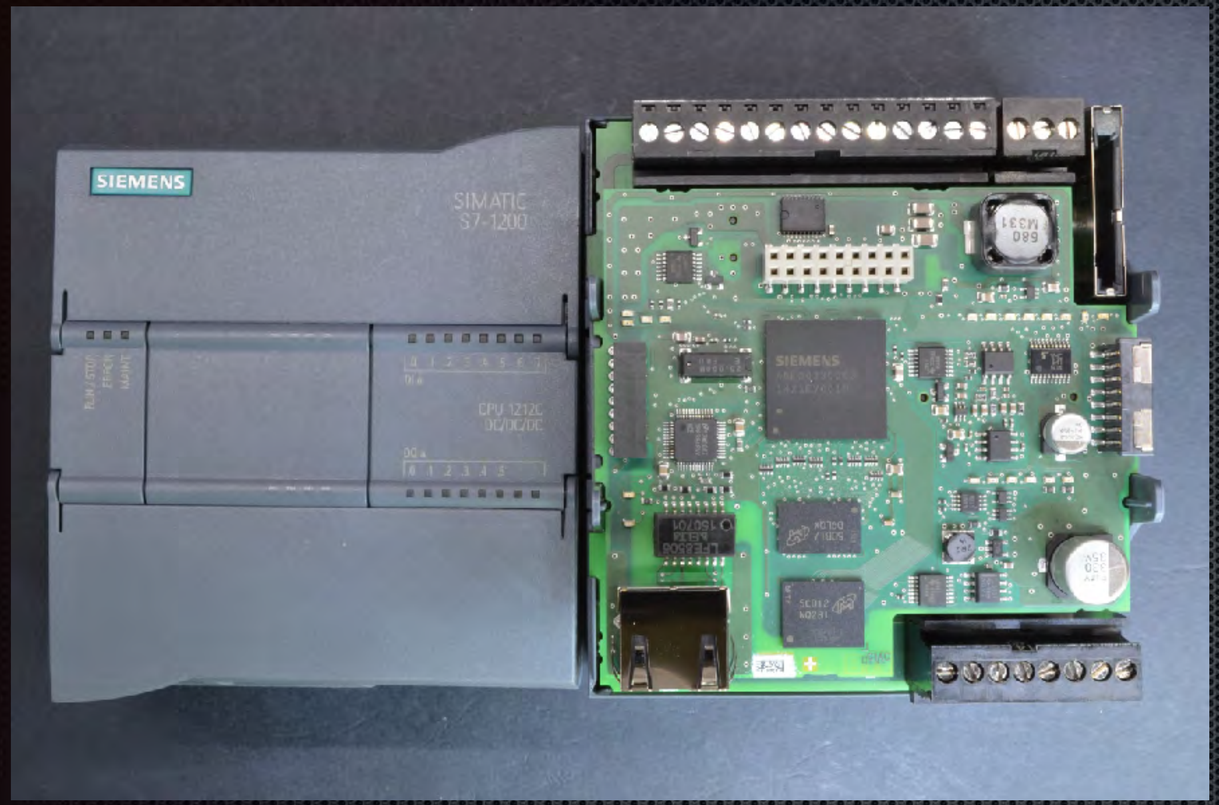
Siemens S7-1200

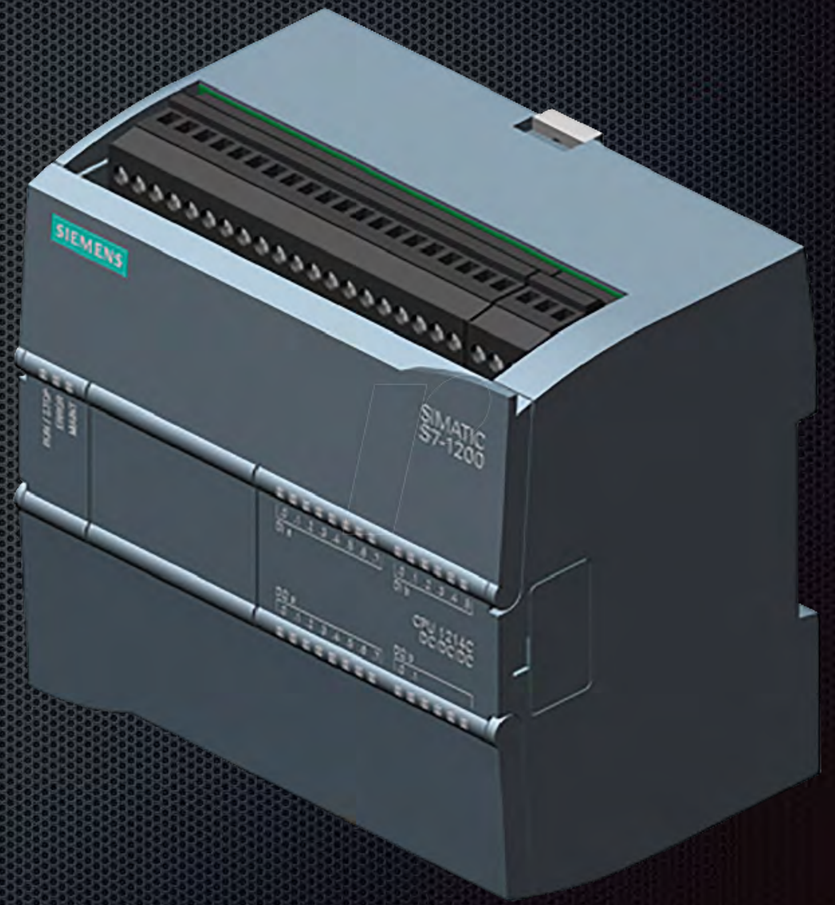


- Entry PLC from Siemens (excluding Logo!)
- Easily accessible to researchers
- Applications*:
  - Chemical
  - Critical Manufacturing
  - Energy
  - Food and Agriculture
  - Water and Wastewater Systems

*Source: https://www.us-cert.gov/ics/advisories/icsa-19-318-02

S7-1200 PLC





Source: http://s7detali.narod.ru/S7_1200/S7_1212C.html

S7-1200 V4 PLC

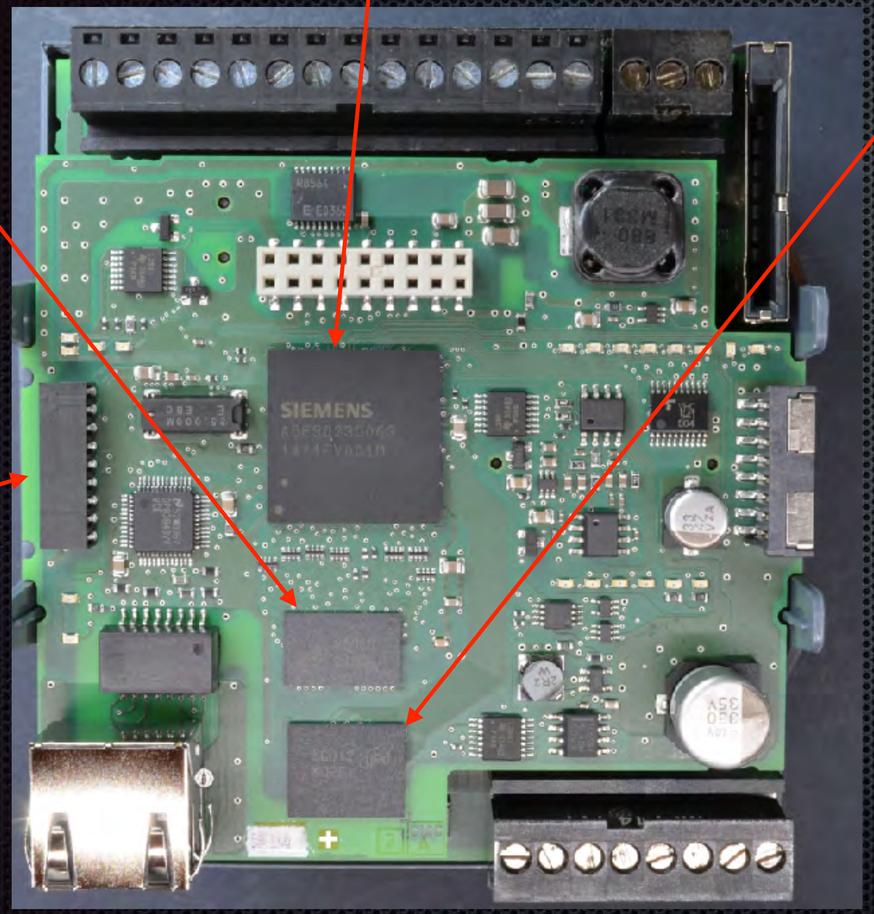
1GB Winbond W94AD2KB
LPDDR1 SDRAM

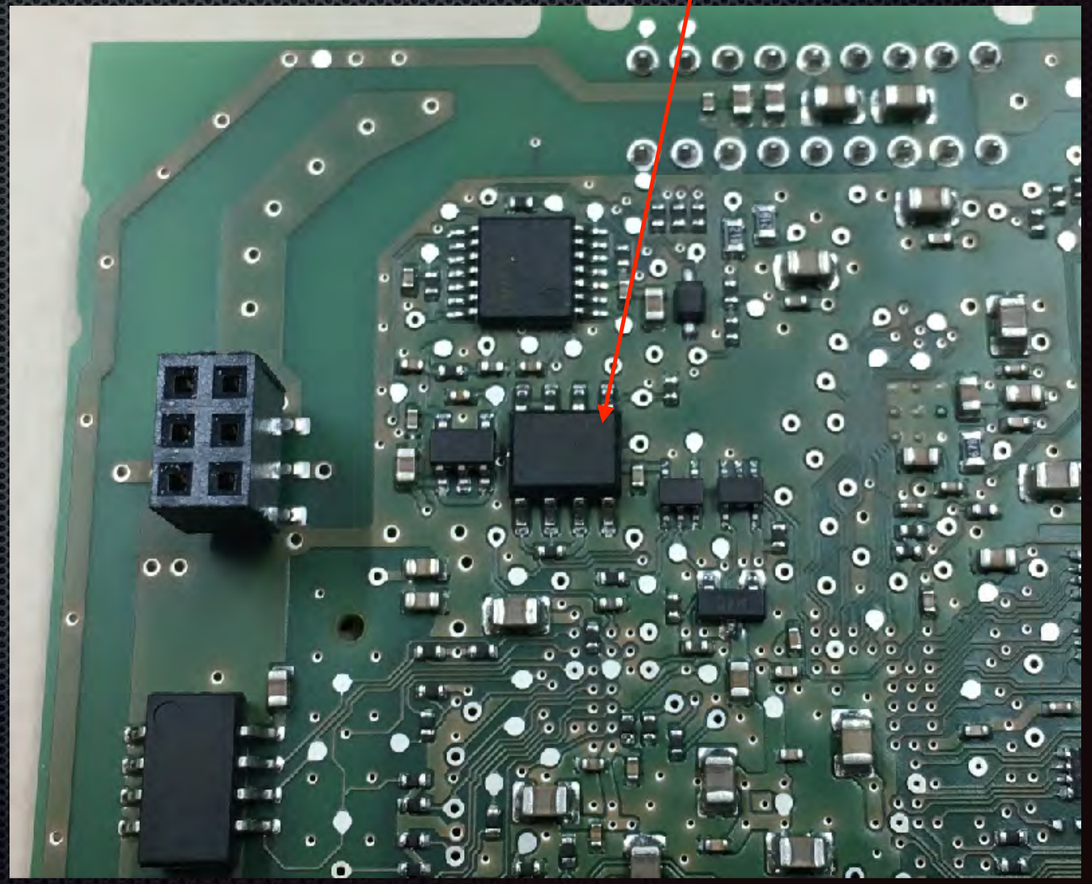
ARM CPU

128MB Micron Technologies
MT29F1G16ABBDAHC-IT:D
63-ball VFBGA NAND Flash for Firmware.

STMicroelectronics M25P40/MX25L4005
4MB SPI Flash for bootloader

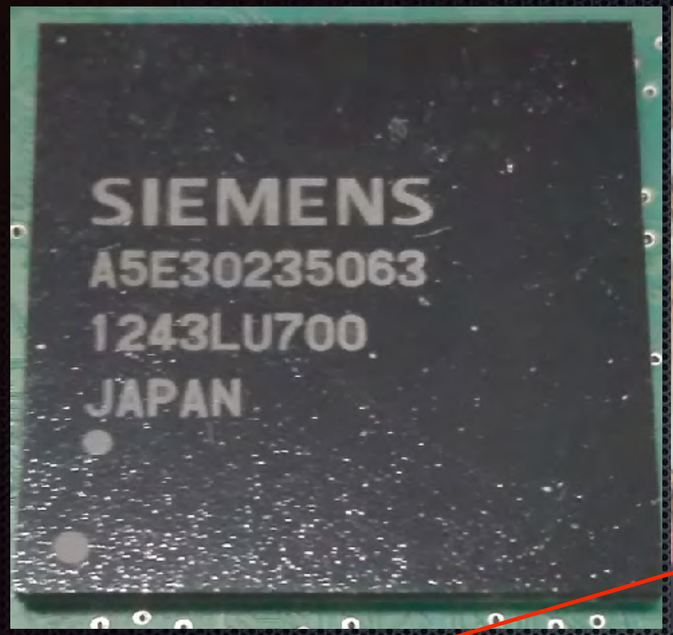
UART/Port for
RS232 extension
(CM 1241)

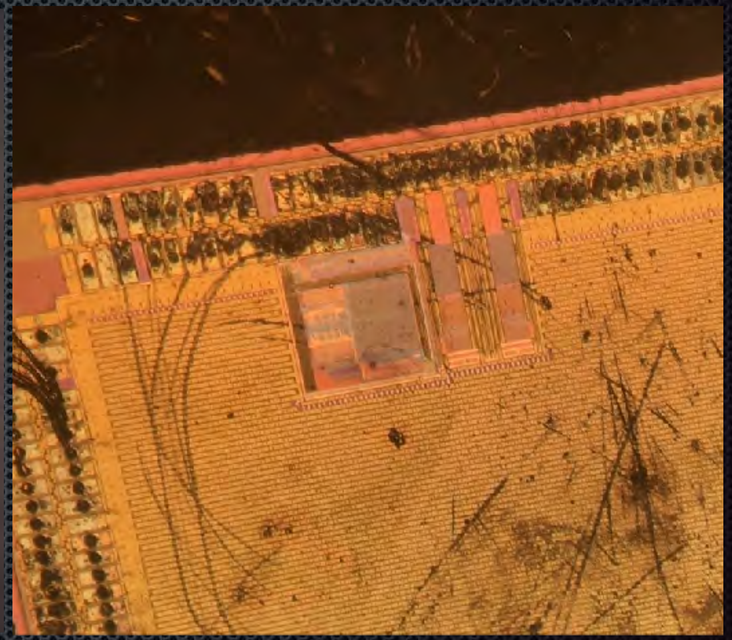




Source: http://s7detali.narod.ru/S7_1200/S7_1212C.html

S7-1200 V4 PLC HARDWARE - SOC DECAP





- Renesas 811005 Manufactured, Siemens A5E30235063 ARM Cortex-R4 (Big-Endian), 2010.
- Instruction Set/Read Main ID Register: Running CP15 command ***mrc p15, 0, r0, c0, c0, 0*** instruction inside PLC yields **0x411fc143** response.

To access the Main ID Register, read CP15 with:

```
MRC p15, 0, <Rd>, c0, c0, 0 ; Read Main ID Register
```

Cortex-R4 and Cortex-R4F Technical Reference Manual Revision: r1p

Home > System Control Coprocessor > System control coprocessor registers > c0, Main ID Register

4.2.2. c0, Main ID Register

The Main ID Register returns the device ID code that contains information about the processor.

The Main ID Register is:

- a read-only register
- accessible in Privileged mode only.

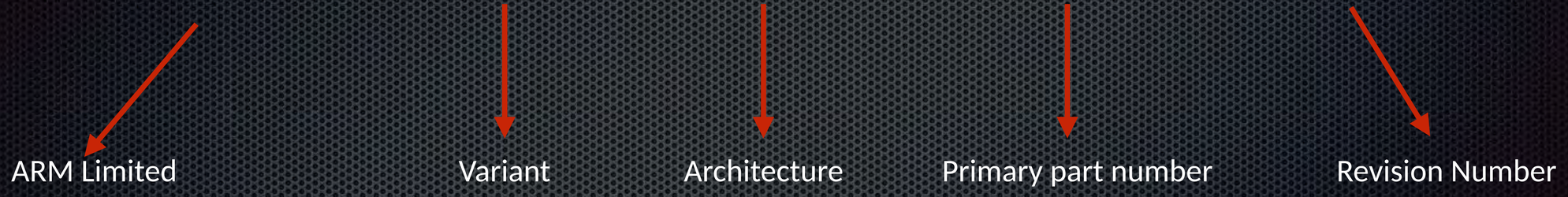
Source:http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0360f/I65012.html

Decoding 0x411fc143

MRC p15, 0, r0, c0, c0, 0 instruction

10000010001111111100000101000011

0x41 - 1 - f - c14 - 3



Implementer

Hex	Implementer
0x00	Reserved for software use
0xC0	Ampere Computing
0x41	**Arm Limited**
0x42	Broadcom Corporation
0x43	Cavium Inc.
0x44	Digital Equipment Corporation
0x49	Infineon Technologies AG
0x4D	Motorola or Freescale Semiconductor Inc.
0x4E	NVIDIA Corporation
0x50	Applied Micro Circuits Corporation
0x51	Qualcomm Inc.
0x56	Marvell International Ltd.
0x69	Intel Corporation

Architecture

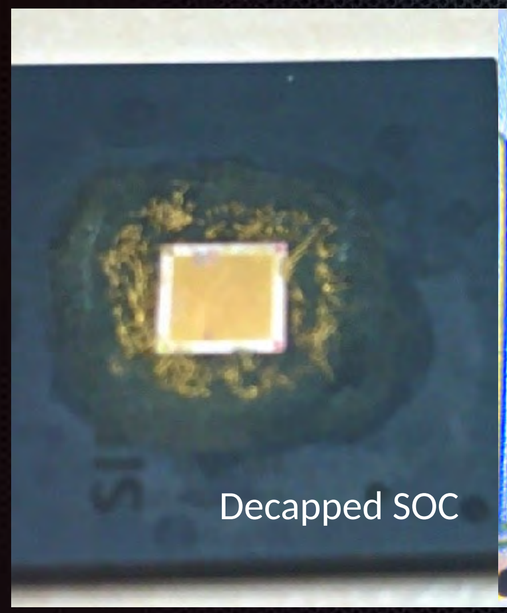
Architecture	Meaning
0b0001	Armv4.
0b0010	Armv4T.
0b0011	Armv5 (obsolete).
0b0100	Armv5T.
0b0110	Armv5TEJ.
0b0111	Armv6.
0b1111	**Architectural features are individually identified in the**

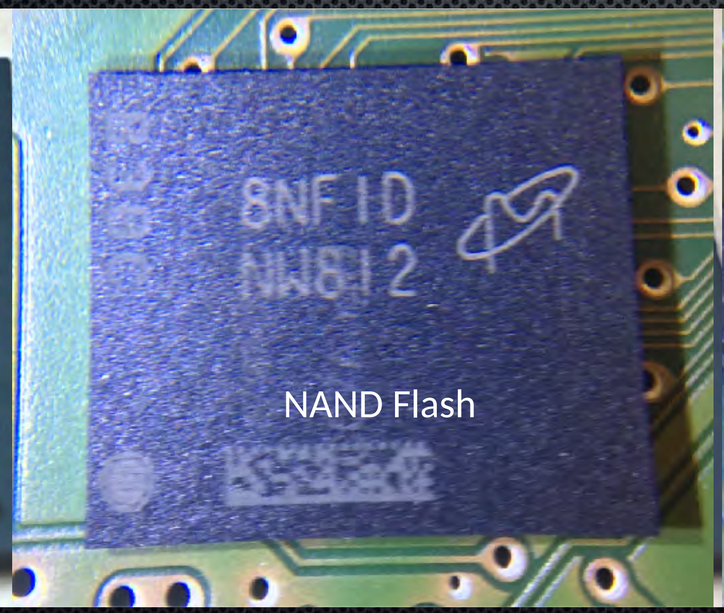
Primary Part Number List

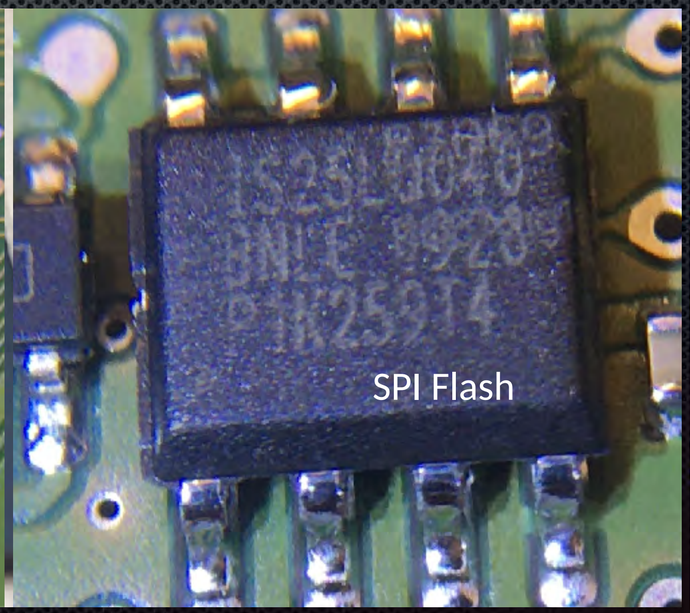
Core	Primary Part Number
Cortex-A5	0xc05
Cortex-A7	0xc07
Cortex-A8	0xc08
Cortex-A9	0xc09
Cortex-A12	0xc0d
Cortex-A15	0xc0f
Cortex-A57	0xd07
Cortex-r4	**0xc14**
Cortex-r5	0xc15

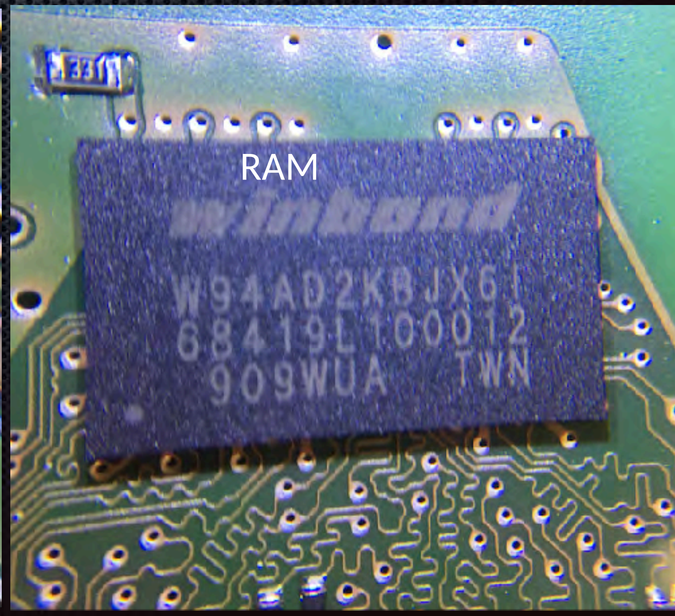
S7-1200 v4 Closer Look

- Cortex R4 Revision 3 ARMv7 R, Thumb 2: Real-Time profile Protected Memory System Architecture (PMSA), based on a **Memory Protection Unit (MPU)**
- Multiple RAM Sizes
- SPI Flash (multiple type)
- NAND



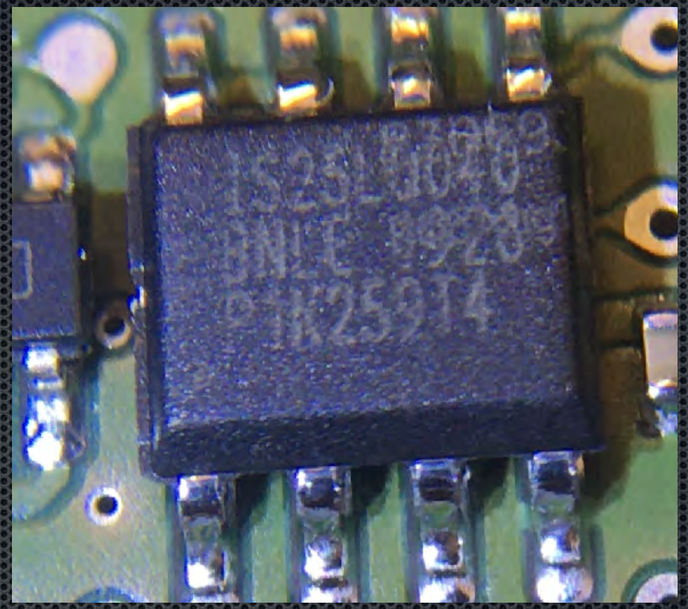


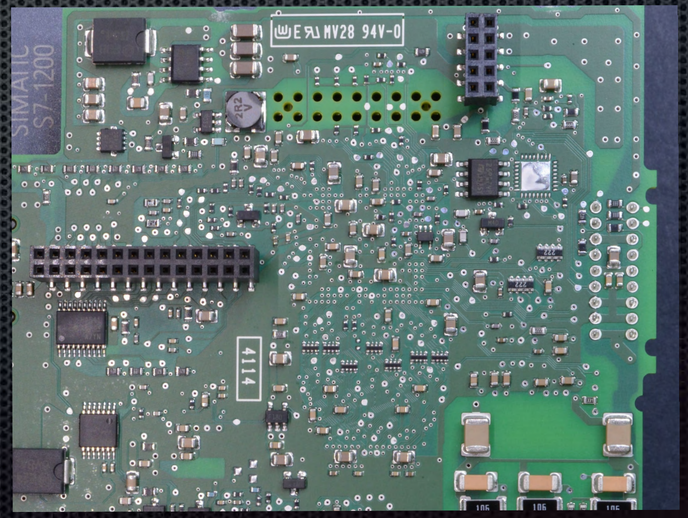


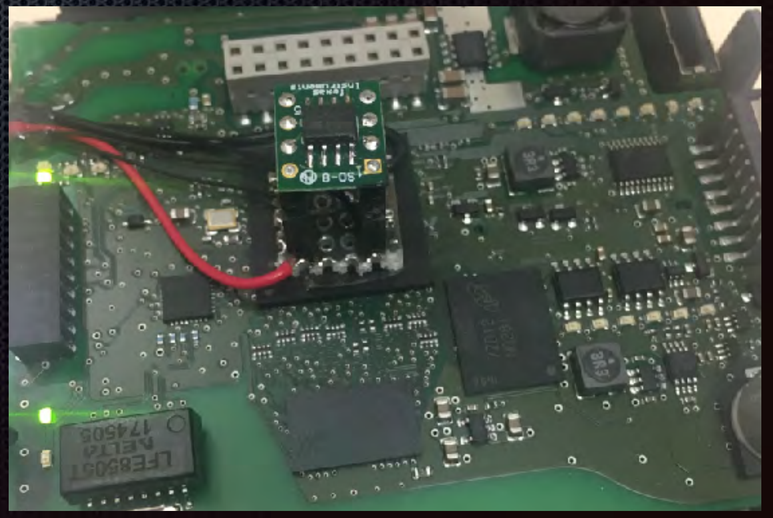


M25P40/ Serial Flash Embedded Memory (bootloader)

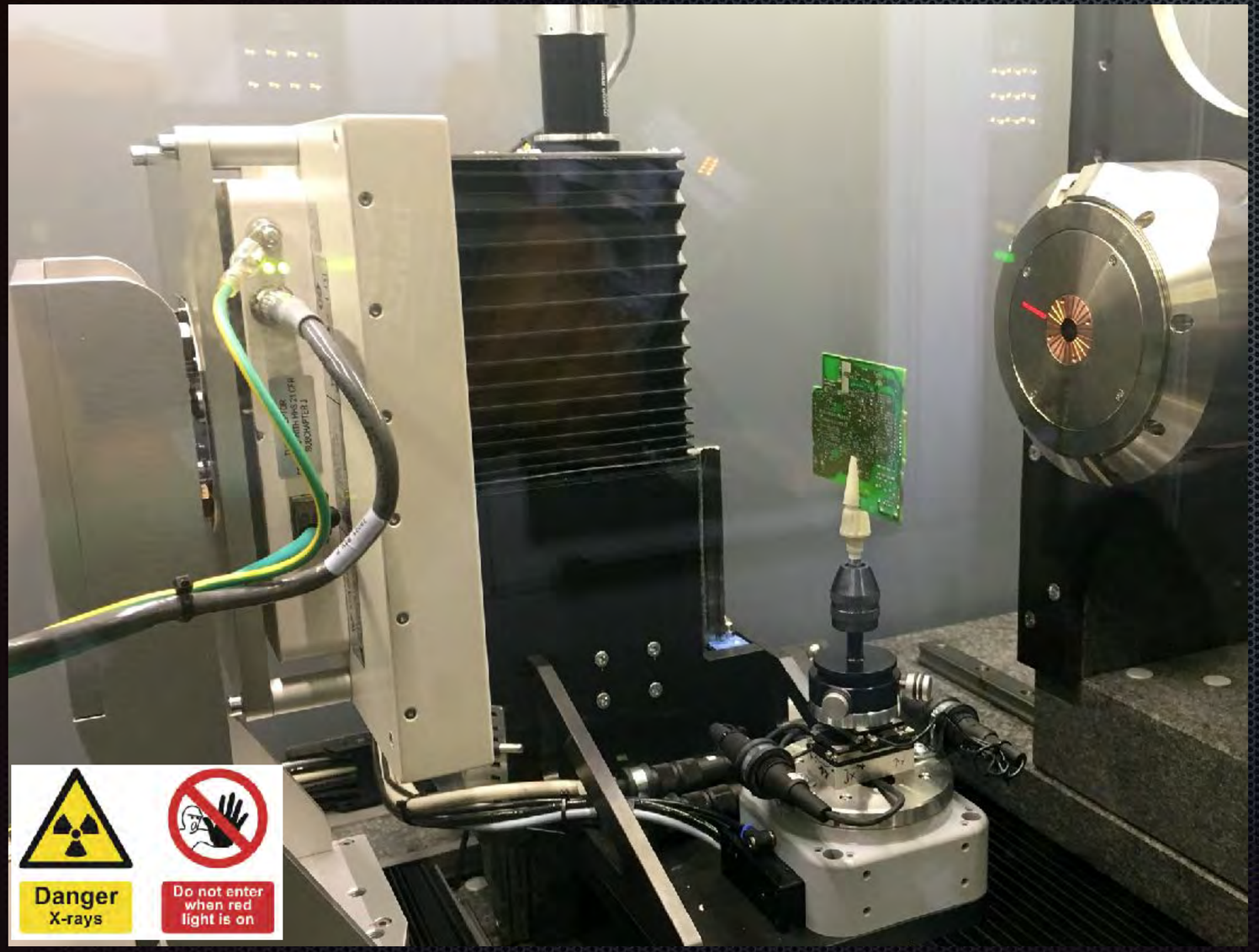
- SPI Flash with 1 to 4MB Low Voltage Paged Flash Memory.
- Contains banks with the size of 512KB.
- Bootloader verifies the integrity of the firmware and loads it to memory

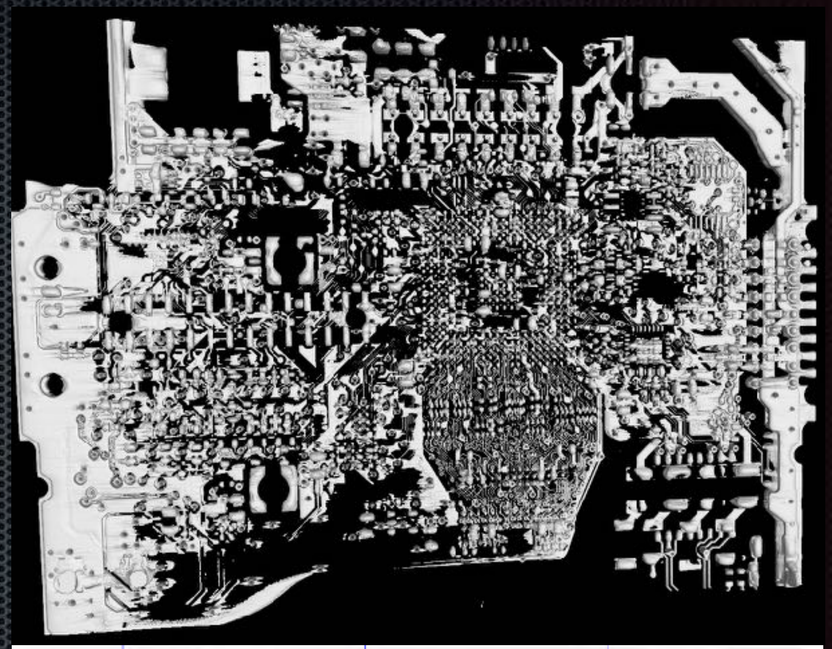


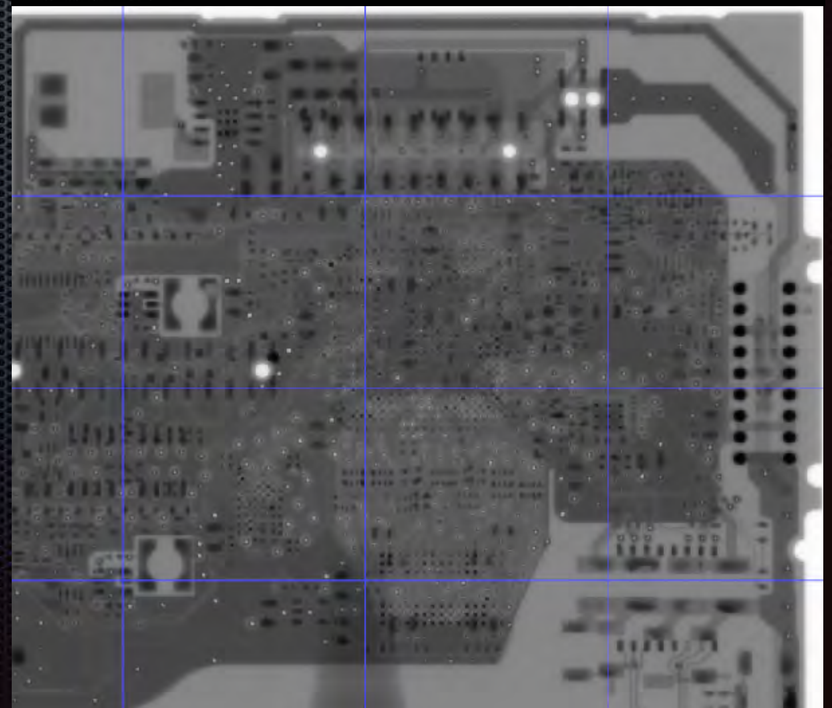




S7-1200 Specs, 3D X-Ray Tomography

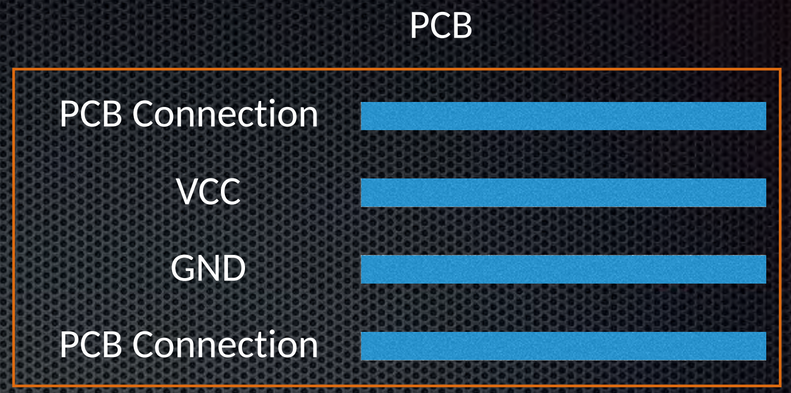


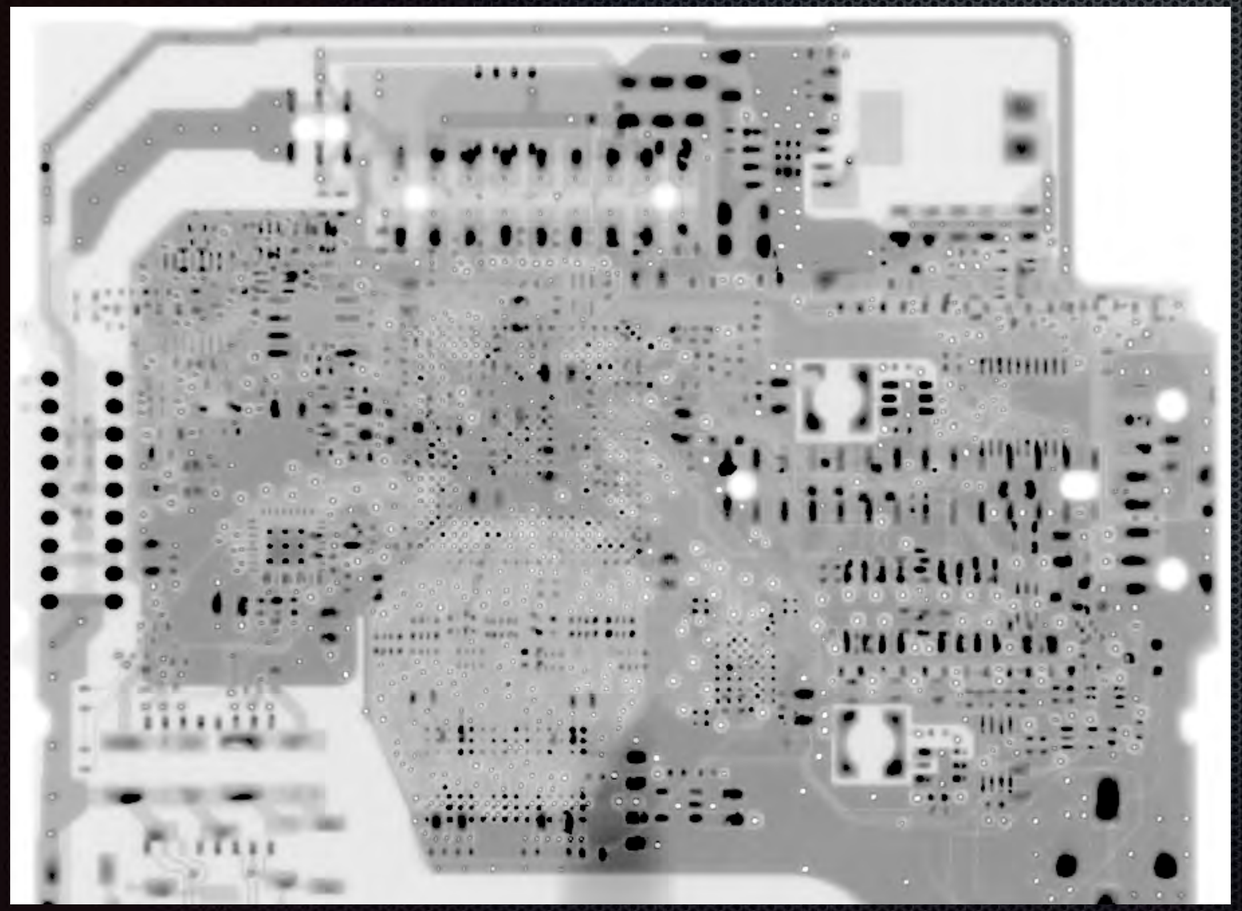


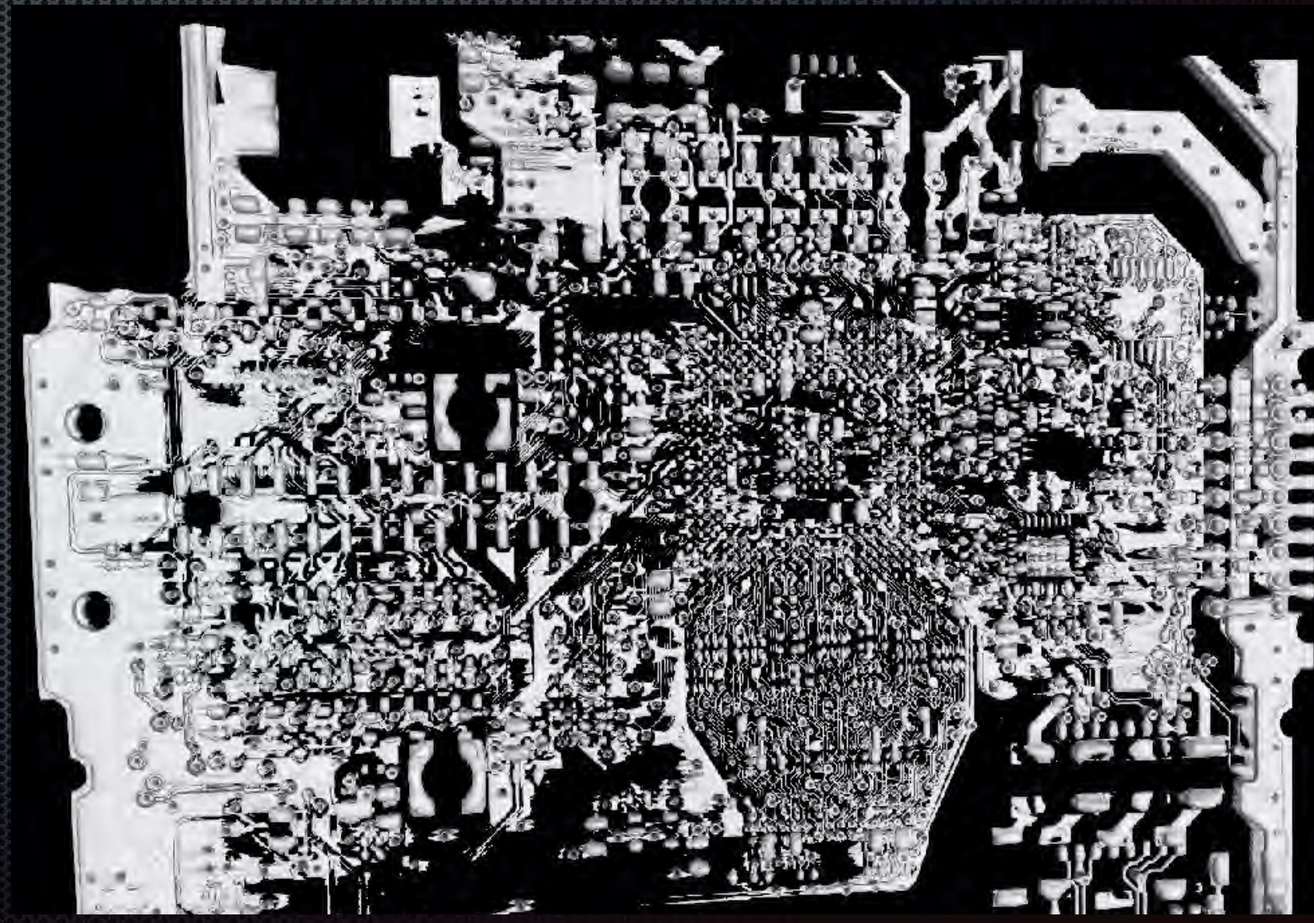


3D X-Ray Tomography

- 4 layered PCB design

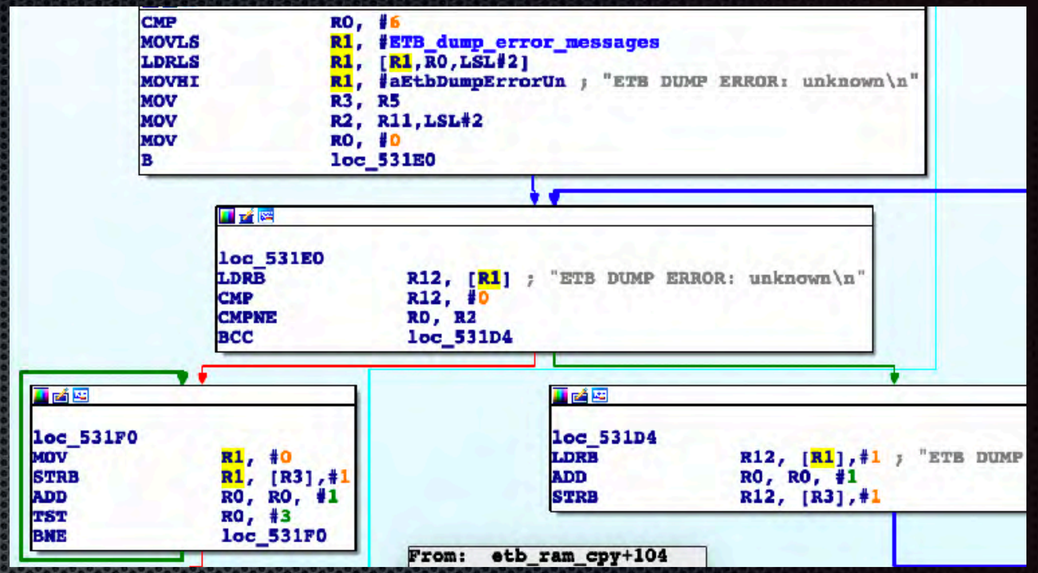


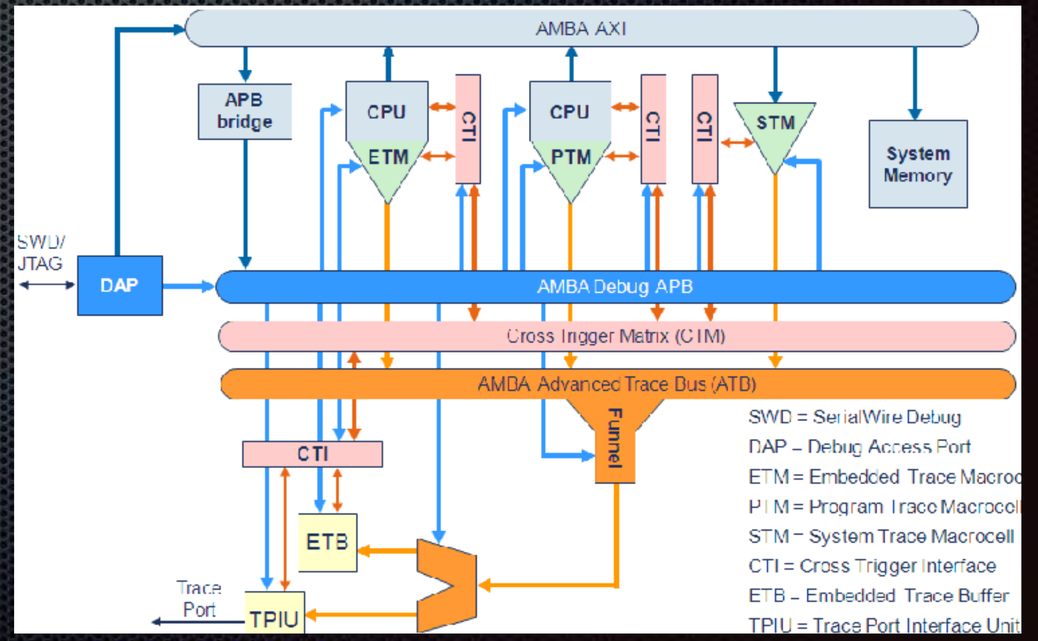




CoreSight in Siemens PLCs

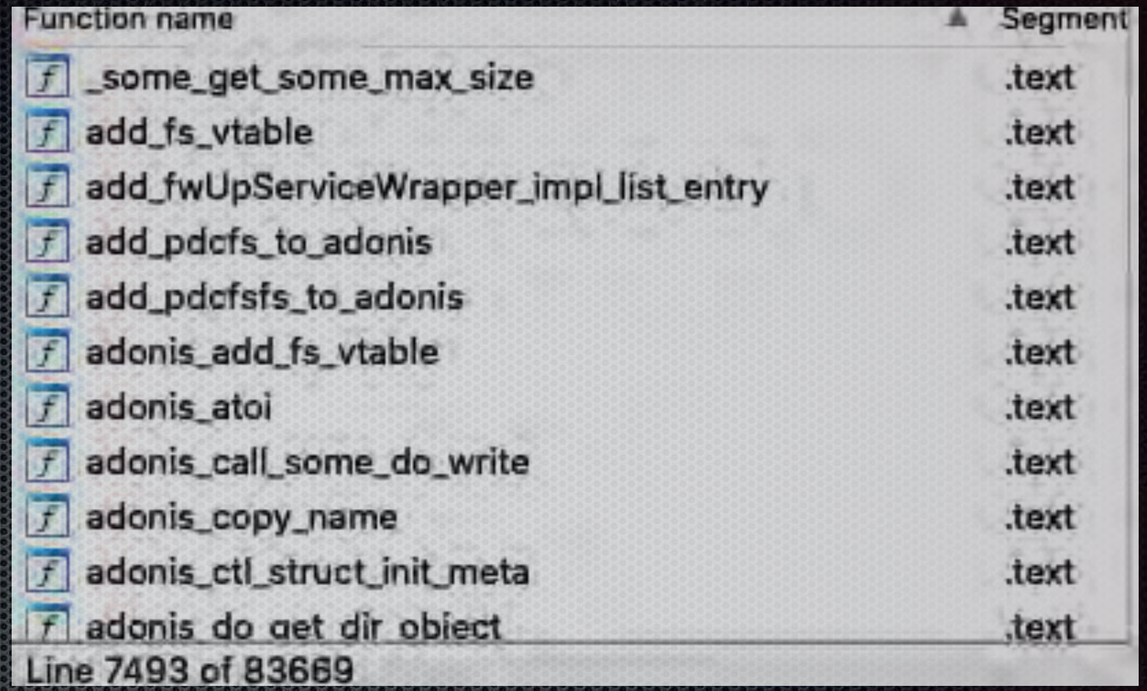
- An ARM hardware tracing feature to trace programs.
- Equivalent to Intel Process Trace (Intel PT) in ARM world.
- Can be used for getting code-coverage from programs.
- Hardware part is well documented*
- Siemens PLCs detect being debugged via a watchdog and brick the PLC upon using CoreSight.





*Reverse Engineering Custom ASICs by Exploiting Potential Supply-Chain Leaks, Thomas Weber, BlackHat Asia 2019.

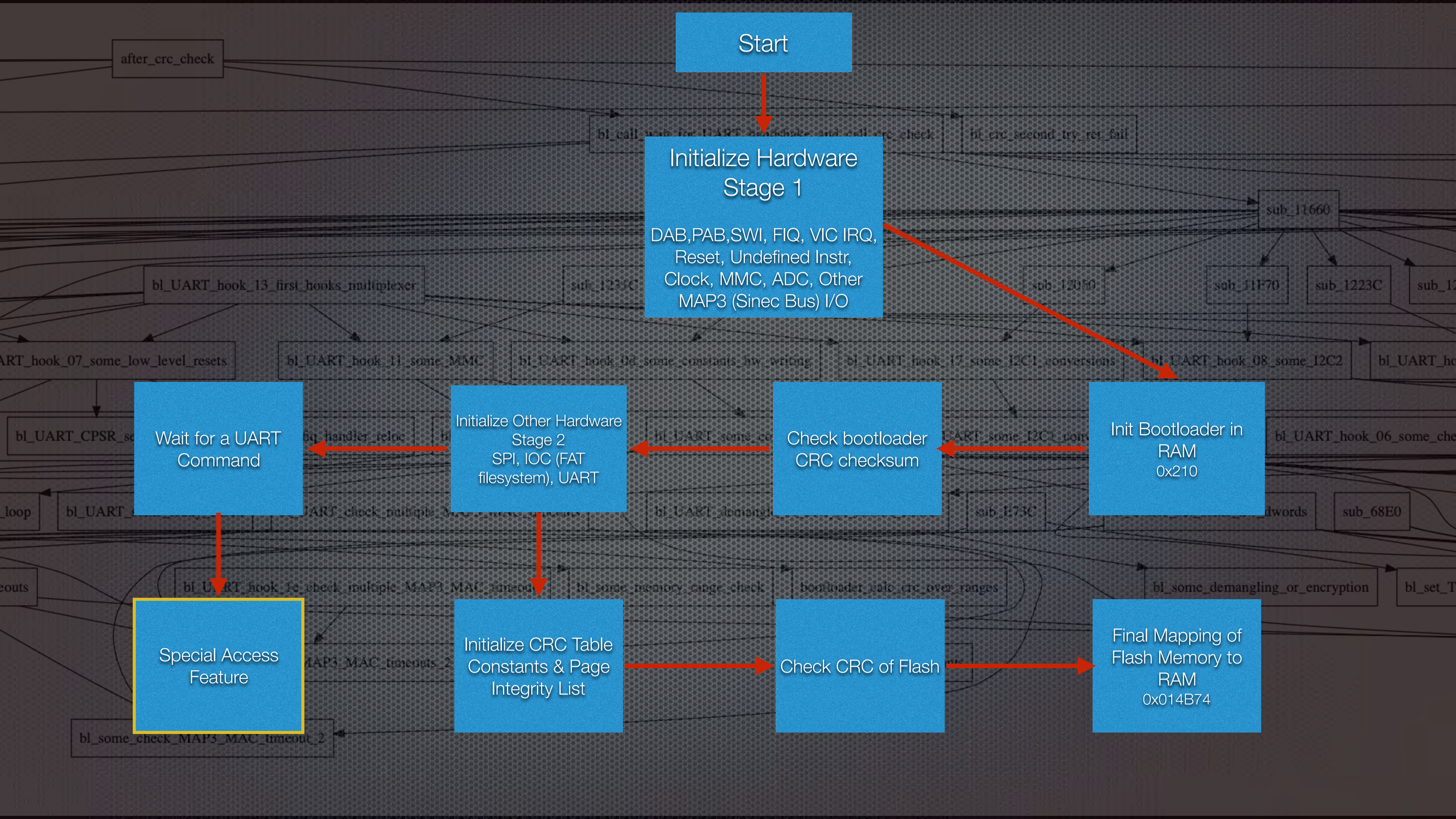
Siemens Firmware



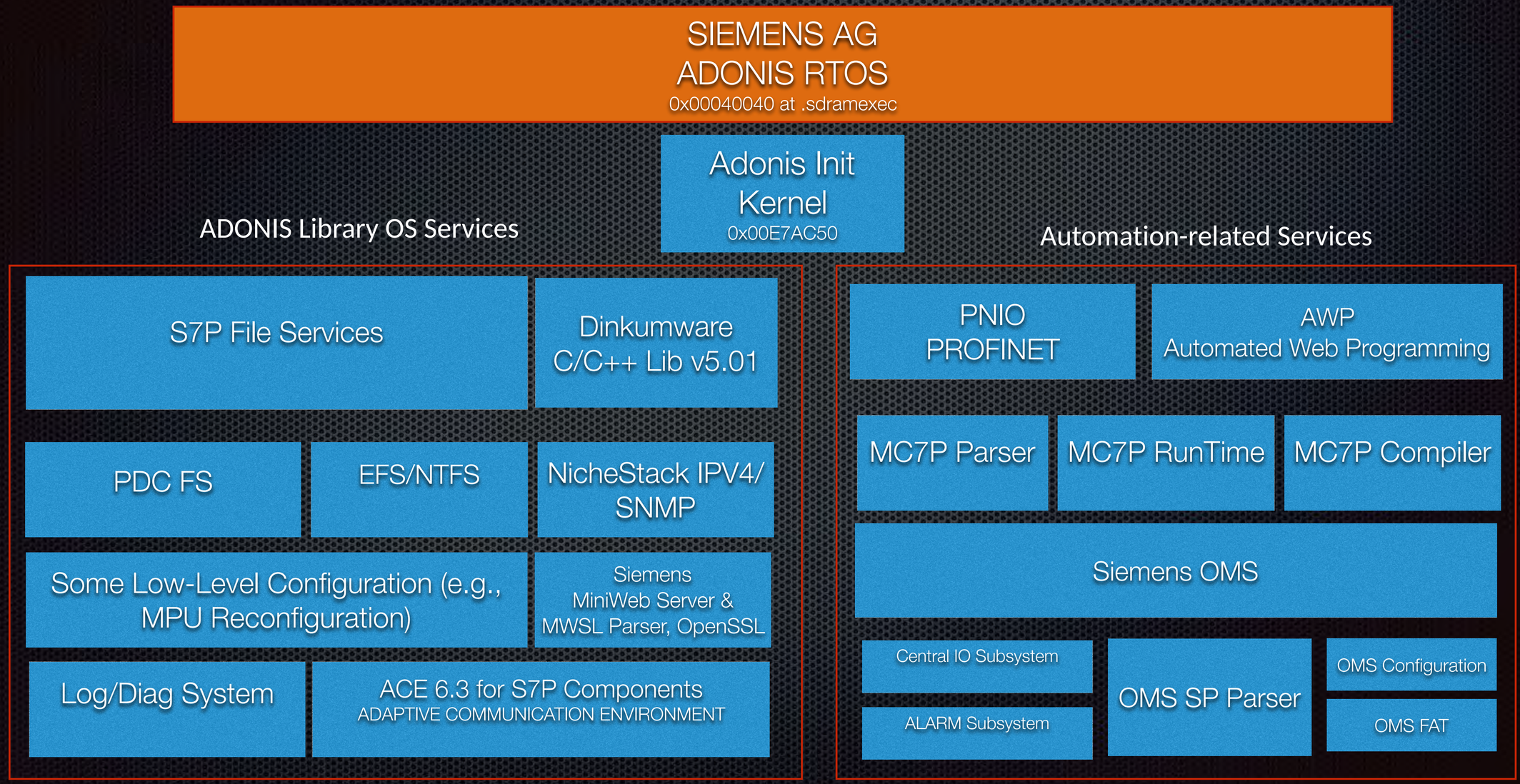
- ~13MB binary.
- LZP3 compressed when downloaded from Siemens website.
- We got a dump of the RAM for both .text and dynamic memory area.
- ~84000 functions identified.

Siemens Bootloader, Startup Process

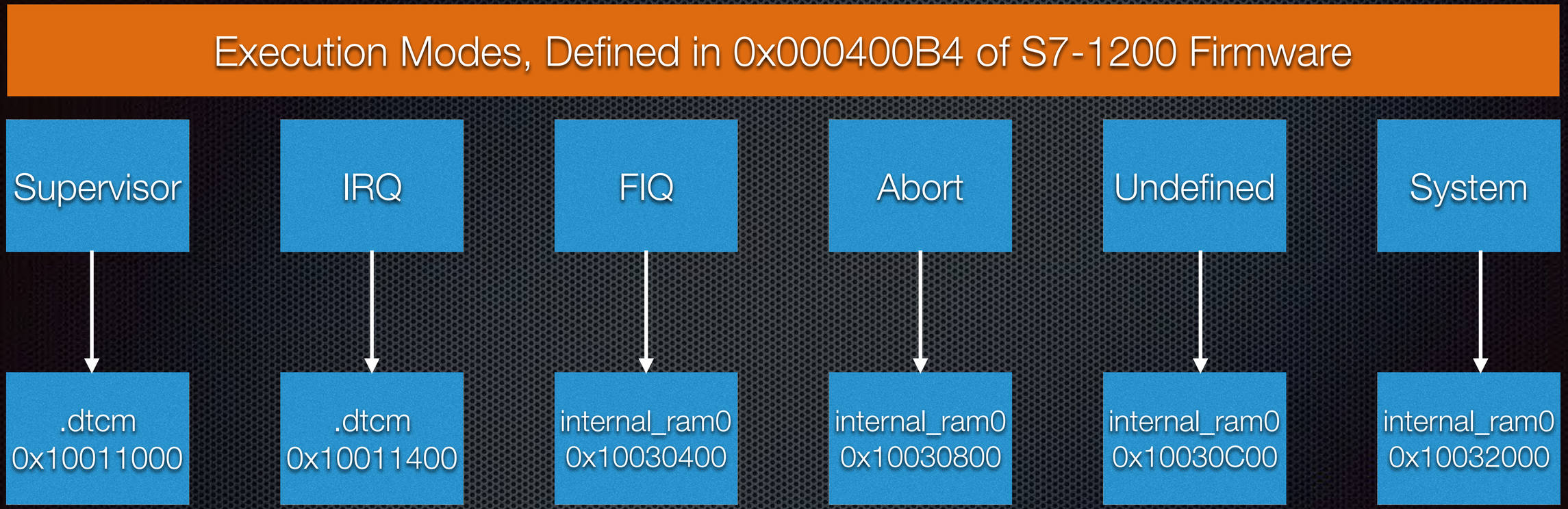
- Setup low level hardware configuration.
  - Vectored Interrupt Controller (VIC)
  - Software Interrupts, Prefetch-Abort Handlers, Data-Abort Handlers,
  - Reset/undefined instruction and FIQ handlers.
- Checking CRC32 checksums of bootloader
  - The last 4 bytes of the bootloader is the CRC checksum for it.
- Check the CRC32 checksums of the firmware before passing control to it.
- Less than 128KB.
- Siemens changed the bootloader SPI Flash in 2018.


Start
Initialize Hardware Stage 1
DAB,PAB,SWI, FIQ, VIC IRQ, Reset, Undefined Instr, Clock, MMC, ADC, Other MAP3 (Sinec Bus) I/O
Init Bootloader in RAM
0x210
Check bootloader CRC checksum
Initialize Other Hardware Stage 2
SPI, IOC (FAT filesystem), UART
Wait for a UART Command
Special Access Feature
Initialize CRC Table Constants & Page Integrity List
Check CRC of Flash
Final Mapping of Flash Memory to RAM
0x014B74

Siemens Firmware Components



Execution Mode Stack in S7-1200 v4



- Some additional interesting code regions:
  - 0x80080000 Flash related MMIO.
  - 0x03c41040 to 0x06fac934: used as heap memory returned by malloc().

FW Memory Mapping on S7-1200 v4, From 0x000439C0

Segment Name	Range	Size	Flags
.exec_in_lomem	0x00000000-0x000075b4	000075b4	1
.bitable	0x00040000-0x00040040	00000040	1
.sdramexec	0x00040040-0x00040510	000004d0	1
.syscall	0x00040540-0x00040548	00000008	1
.th_initial	0x00041040-0x00043998	00002958	33
.secinfo	0x000439c0-0x00043cfc	0000033c	34
.fixaddr	0x00043d00-0x00043d00	00000000	4
.fixtype	0x00043d00-0x00043d00	00000000	4
.text	0x00043d00-0x00defda0	00dac0a0	33
.rodata	0x00defdc0-0x011a116c	003b13ac	34
.data	0x011a1180-0x011c1d14	00020b94	42
.bss	0x01e01040-0x02620f58	0081ff18	12
.cc_memory	0x03641040-0x03641040	00000000	4
.uninitialized	0x03c41040-0x06fac934	0336b8f4	12
CLSI_CACHED_MEM_POOL	0x06fac940-0x06fac940	00000000	4
.dram_uncache	0x07ff0000-0x07ff0000	00000000	4
MAP_MAC_MEM	0x07ff0000-0x07ff0494	00000494	12
.iram0	0x10030000-0x10037aa0	00007aa0	12
.iram1	0x10040000-0x1004c35c	0000c35c	12
.crctable	0x1004f400-0x1004f800	00000400	12
.softboot	0x1004f800-0x1004ff00	00000700	12
.bootinfo	0x1004ff00-0x1004ff1c	0000001c	12
.dtcm	0x10010000-0x10012c70	00002c70	12

I/O Memory Mapping on A5E30235063, From 0x00df4a80

Segment Name	Range	Size	Segment Name	Range	Size
itcm	0x00000000-0x00008000	0x00008000	MAP3_OUTPUTS	0xfffba000-0xfffba218	0x00000218
ddram	0x00008000-0x04000000	0x03ff8000	MAP3_ITIMER0	0xfffbb000-0xfffbb010	0x00000010
configured_dtcm	0x10010000-0x10014000	0x00004000	MAP3_ITIMER1	0xfffbb010-0xfffbb020	0x00000010
internal_ram0	0x10030000-0x10040000	0x00010000	MAP3_ITIMER2	0xfffbb020-0xfffbb030	0x00000010
internal_ram1	0x10040000-0x10050000	0x00010000	MAP3_ITIMER3	0xfffbb030-0xfffbb040	0x00000010
MAP3_PWRSTK	0xfffb0000-0xfffb003c	0x0000003c	MAP3_ITIMER4	0xfffbb040-0xfffbb050	0x00000010
MAP3_SPI0	0xfffb1000-0xfffb1018	0x00000018	MAP3_ITIMER5	0xfffbb050-0xfffbb060	0x00000010
MAP3_SPI1	0xfffb2000-0xfffb2018	0x00000018	MAP3_ITIMER6	0xfffbb060-0xfffbb070	0x00000010
MAP3_I2C0	0xfffb3000-0xfffb306c	0x0000006c	MAP3_ITIMER7	0xfffbb070-0xfffbb080	0x00000010
MAP3_I2C1	0xfffb4000-0xfffb406c	0x0000006c	MAP3_ITIMER8	0xfffbb080-0xfffbb090	0x00000010
MAP3_I2C2	0xfffb5000-0xfffb506c	0x0000006c	MAP3_ITIMER9	0xfffbb090-0xfffbb0a0	0x00000010
MAP3_ADC	0xfffb6000-0xfffb6024	0x00000024	MAP3_ITIMER10	0xfffbb0a0-0xfffbb0b0	0x00000010
MAP3_UART0	0xfffb7000-0xfffb709c	0x0000009c	MAP3_ITIMER11	0xfffbb0b0-0xfffbb0c0	0x00000010
MAP3_UART1	0xfffb8000-0xfffb809c	0x0000009c	MAP3_ITIMER12	0xfffbb0c0-0xfffbb0d0	0x00000010
MAP3_HSC0	0xfffb9100-0xfffb9180	0x00000080	MAP3_ITIMER13	0xfffbb0d0-0xfffbb0e0	0x00000010
MAP3_HSC1	0xfffb9180-0xfffb9200	0x00000080	MAP3_TIMERS	0xfffbb000-0xfffbb15c	0x0000015c
MAP3_HSC2	0xfffb9200-0xfffb9280	0x00000080	MAP3_IOC	0xfffbc000-0xfffbc02c	0x0000002c
MAP3_HSC3	0xfffb9280-0xfffb9300	0x00000080	MAP3_FL_MEMCTL	0xfffbd000-0xfffbe000	0x00001000
MAP3_HSC4	0xfffb9300-0xfffb9380	0x00000080	MAP3_VIC	0xfffffc00-0xfffffe00	0x00000200
MAP3_HSC5	0xfffb9380-0xfffb9400	0x00000080	MAP3_EMB0	0xfff50000-0xfff50048	0x00000048
MAP3_INPUTS	0xfffb9000-0xfffb9400	0x00000400	MAP3_EMB1	0xfff51000-0xfff51048	0x00000048
MAP3_PLS0	0xfffba000-0xfffba080	0x00000080	MAP3_DDR_MEMCTL	0xfff52000-0xfff5208c	0x0000008c
MAP3_PLS1	0xfffba080-0xfffba100	0x00000080	MAP3_MMC	0xfff60000-0xfff60104	0x00000104
MAP3_PLS2	0xfffba100-0xfffba180	0x00000080	MAP3_LCD	0xfff70000-0xfff70ff8	0x00000ff8
MAP3_PLS3	0xfffba180-0xfffba200	0x00000080	MAP3_MAC	0xfff90000-0xfff900a4	0x000000a4
NOT USED	NOT USED	NOT USED	MAP3_BOOL_HELPER	0xfffa0000-0xfffa4000	0x00004000

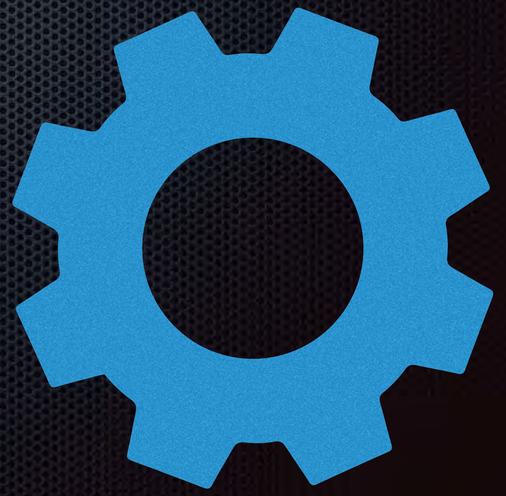
ADONIS MPU Configuration at 0x000400B4

RG-NR	base_addr	region_end	region_size	region_access (XN=Execute Never, Read/Write, Type Extension, Cacheable, Bufferable)
B	FFF00000	100000000	1MB	XN=1 Privilege Mode: RW, user mode: RW, TEX: 0 Share: 0 C: 0 B: 1
A	7FF0000	8000000	64KB	XN=0 Privilege mode: RW, User mode: RW, TEX: 4 Share: 0 C: 0 B: 0
9	0	8000000	128MB	XN=0, Privilege mode: RW, User mode: RW, TEX: 4 Share: 0 C: 0 B: 1
8	80000000	84000000	64MB	XN=1, Privilege mode: RW, User mode: RW, TEX: 0 Share: 0 C: 0 B: 0
7	10030000	10040000	64KB	XN=0, Privilege mode: RW, User mode: RW, TEX: 4 Share: 0 C: 0 B: 0
6	10040000	10050000	64KB	XN=0, Privilege mode: RW, User mode: RW, TEX: 4 Share: 0 C: 0 B: 1
4	10010000	10014000	16KB	XN=0, Privilege mode: RW, User mode: RW, TEX: 4 Share: 0 C: 0 B: 0
3	10020000	10022000	8KB	XN=0, Privilege mode: RO, User mode: RO, TEX: 4 Share: 0 C: 0 B: 1

TEX: Type extension field, allows you to configure memory access type, for example strongly ordered, peripheral.

Siemens Firmware Boot Process

- **0x00040040**: Enable instruction cache and some system specific coprocessor register
- **0x000400B4**: Set up stack pointers for different execution modes.
- **0x000402A4**: Set up the Vectored Interrupt Controller (VIC)
- **0x000400B4**: Configure MPU via MPU coprocessor instructions.
- **0x000403E4**: Zero out the .bss section
- **0x00567698**: Set some IOC config and run ADONIS boot function



ADONIS Kernel

- ADONIS boot at 0x002E098C
- An array of initialization functions are called one by one.
  - OMS and XML serialize parser initialize here.
  - HTTP Handlers, MiniWeb.
- Setup management structures
  - LTRC Subsystem initialize here.
- Setup file system entries in .bss
- Threading is supported in the OS.
- Memory Management:chunks, malloc, free and techniques such as reference counting available.

ADONIS File System

- We identified four different file system devices
- FAT file system (and NTFS) is supported.
- Proprietary PDCFS (Power Down Consistency Files System)

Embedded Software Development Engineer
Siemens
Jul 2014 – Dec 2016 · 2 yrs 6 mos

Responsible for design and development of software for industrial automation product called PROFINET Driver. The driver, delivered as a source code, simplifies individual Profinet communications on a standard PC or embedded board.

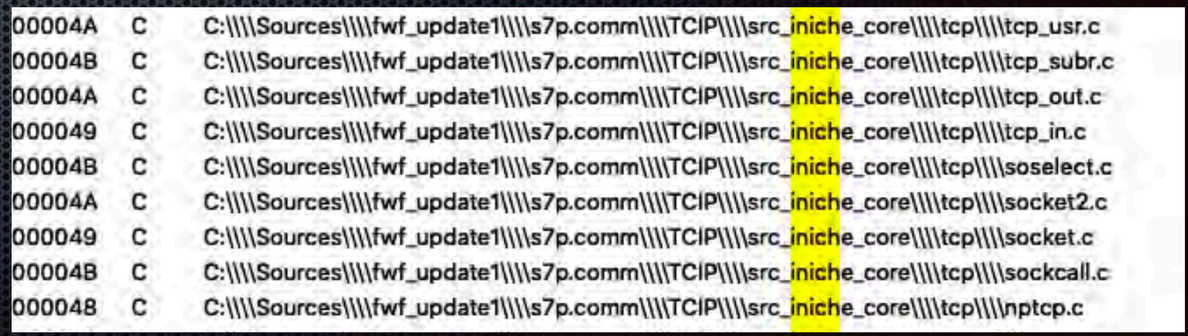
- Platform Independent application
- Portable application
- C/C++ project
- Object Oriented Design
- Test Driven Development
- Google Test/Mock Framework
- Profinet
- TIA Portal
- Windows, Linux and Adonis OS
- Embedded SoC card (ASIC)

Intern
Siemens
May 2012 – Aug 2012 · 4 mos
Princeton, NJ

Identified and implemented critical features on FAT32 compliant proprietary filesystem (PDCFS) on Adonis and WinCE6.0.
Debugged and fixed filesystem bugs on these platforms.

ADONIS TCP/IP Stack

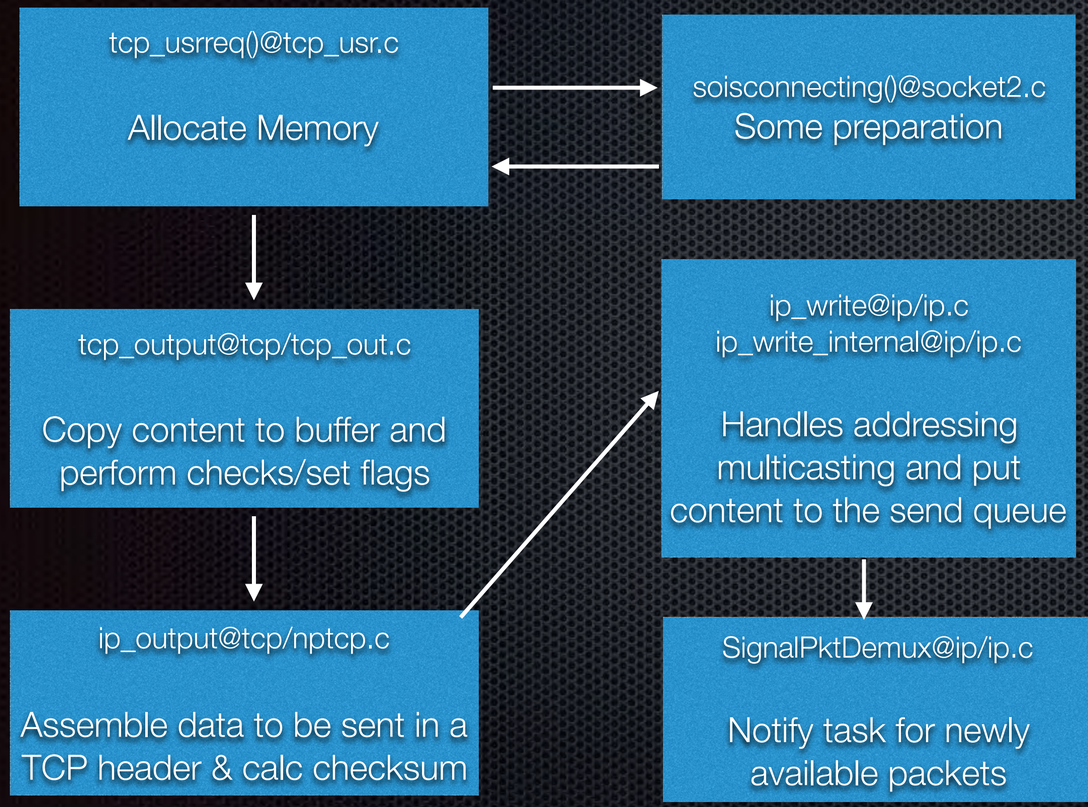
- ADONIS utilizes the InterNiche Technologies TCP/IP Stack v3.1
- IP, TCP, UDP, ARP, SNMP, NTP are supported.
- TCP/IP stack is already leaked via some OEM of the stack. Not all components are used in S7 PLCs.
- We were able to independently compile the leaked stack.



ADONIS TCP/IP Stack, InterNiche Technologies INET Module

- Several wrapper functions for socket creation, closing, binding and receiving in iniche/tcp/**sockall.c**.
- Based on functions on tcip/tcp/socket.c (soclose, sobind, soreceive)
- Handling Struct:
  - struct protosw * so_proto; /* protocol handle */
- In our version, the structs are defined in tcip/tcp/protosw.h
  - int (*pr_usrreq) __P ((struct socket *, struct mbuf *, struct mbuf *));

Example TCP Connect() Invocation in PLC



Symbol	Description
INICHE_tcip_proto_SOCK_STREAM	The struct of type `struct protosw` as defined in `tcip/tcp/protosw.h` for the socket type SOCK_STREAM (tcp socket).
tcp_usrreq	The handler function for serving user requests for TCP sockets.
INICHE_tcip_proto_SOCK_DGRAM	The struct of type `struct protosw` as defined in `tcip/tcp/protosw.h` for the socket type SOCK_DGRAM (udp socket).
udp_usrreq	The handler function for serving user requests for UDP sockets.

Firmware Update Process On S7 PLC

- Updates available via Webserver and SDCard **(24MB,** costs a whopping **~250** Euros!)
- Siemens S7-1200 firmware (.upd) are compressed update structure, contains:
  - Constant size metadata (44 bytes)
  - Headers describing contents of the file
  - Contents of headers
    - Types seen so far:**BG_ABL**: descriptor, **FW_SIG**: firmware signature, **A00000**: Main update contents, **B00000**: metadata

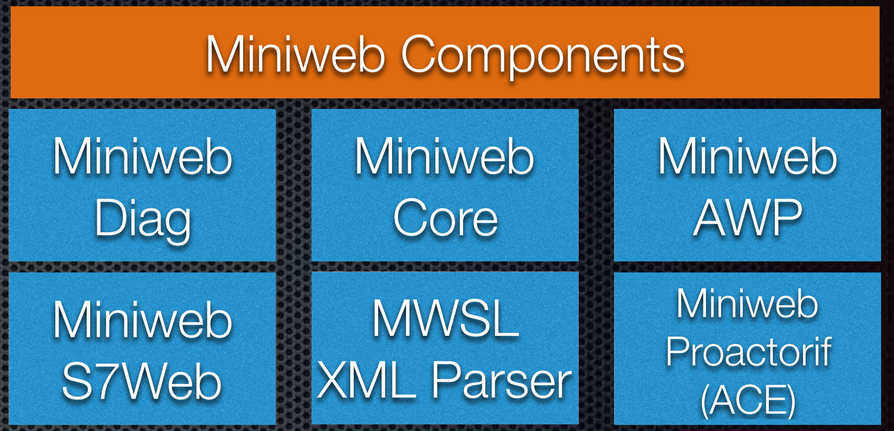
Decompressed Firmware Update File Structure

- Decompressing a `.upd` file results in the raw structure of the flash contents (has its own format)
- Actual firmware update occurs at 0x0059CAE0.
  - Bootloader has its own update function, skipping signature verification (at 0x000137E8).
- LZP3 decompress routine available at 0x00BCEAA8 of V04.02.01 firmware.
- Firmware signature is based on ECDSA signature over the SHA256 hash of compressed firmware contents.
- Public key available at 0x010D1EE8 of V04.02.01 firmware

Firmware Structure
Size 0x40

Offset	Name
00	Magic Header 0x5D1B4153
04	Entry Point 0x00040040
08	File offset of end of IRAM/Start FW contents , 0x00040040
0C	Unknown
10	Address of Firmware Section 0x00040000
14	Header Size (0x40) 0x00040000
18	(File offset end of IRAM/Start FW)-1 0x00040000
1C	Address of IRAM (0x10000000)
20	Version DWORD
??	Unknown
34	File offset or end of FW contents
…	Unknown
3C	CRC over header

MiniWeb, Siemens S7 PLCs Webserver



- Miniweb starts at 0x0063BE20 as a service in main firmware initialization routine of ADONIS
- HTTP login mechanism:
  - Initial checks are called from FormLogin routine and some undocumented HTTP request handler.
  - Actual password check happens at 0x007CCBA0.
- Miniweb seems like using jQuery version 1.11 which has some known XSS issues.
- Part of Miniweb code base used for Windows-based webserver from Siemens.

Undocumented HTTP Handlers

HTTP GET Request Handler	Description
/appapiappa/vvvvvvvvv	version
/appapiappa/vvvvvvvvvv	version
/appapiappa/lililililil	log in
/appapiappa/lololololo	log out
/appapiappa/cmcmcmcmcm	change CPU mode
/appapiappa/flflflflfl	flash LEDs
/appapiappa/gbpigbpigb	get station info as json (name, mac, mode)
/appapiappa/gmigmigmig	get module info (list): get name, serial, FW version, HW version, status
/appapiappa/galegalega	unknown
/appapiappa/galeldgale	get AS log entry
/appapiappa/gvigvigvig	unknown
/appapiappa/svsvsvsvsv	unknown

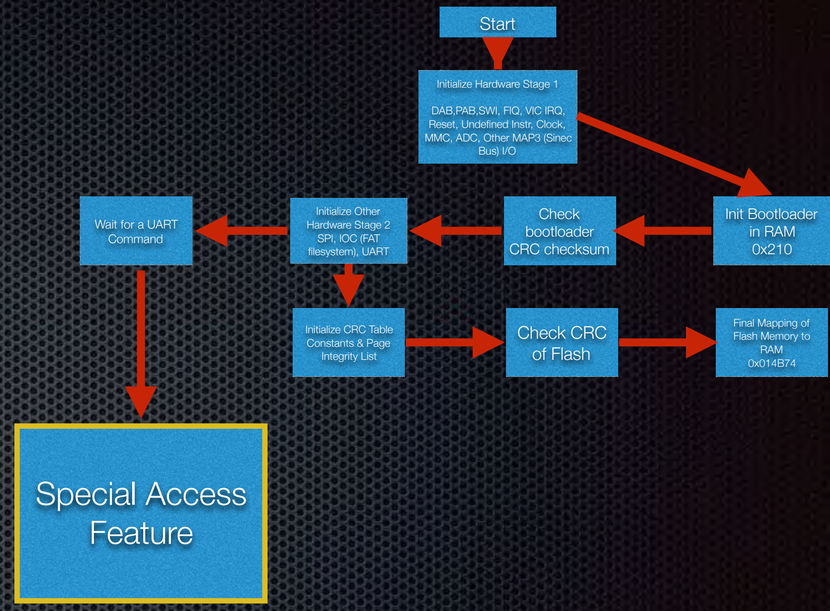
MiniWeb Scripting Language (MWSL)

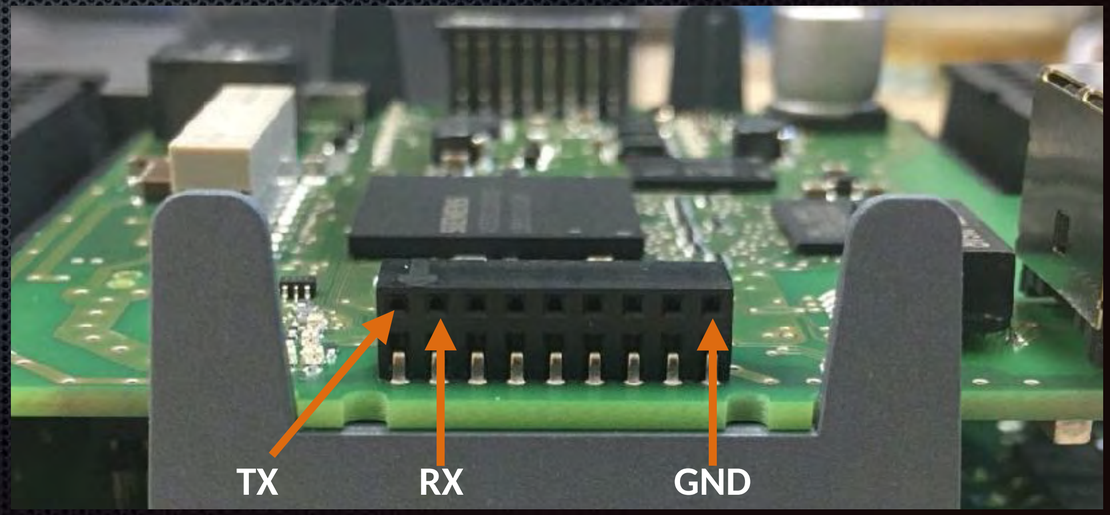
```
<MWSL>

    //MWSL code to be executed

    WriteVar("CPULoad.Percent", "PROCESS", "%e");

    GetVar("var/systemClock", "PROCESS", "%d");

</MWSL>
```

- Users can create custom web pages using MWSL.
- MWSL is somewhat resembling PHP with JavaScript Syntax.
- Errors in MWSL functions result in error messages in output HTML (possible leakage)
- No output encoding apart from url encoding and decoding functions
  - Raises the risk of trivial XSS vulnerabilities.
  - GetVar("var/modeOfOperation", "PROCESS", “%s") supports arbitrary-length outputs.
- Compiled MWSL programs consist of a compiled handler function and nested templating chunk structs.

Special Access Feature

- A set of special functionalities within the PLC bootloader
  - Various diagnostics functions, Exposed via UART Access
  - UART port was previously documented*.
- There were other functionalities in it, which we did not investigate.

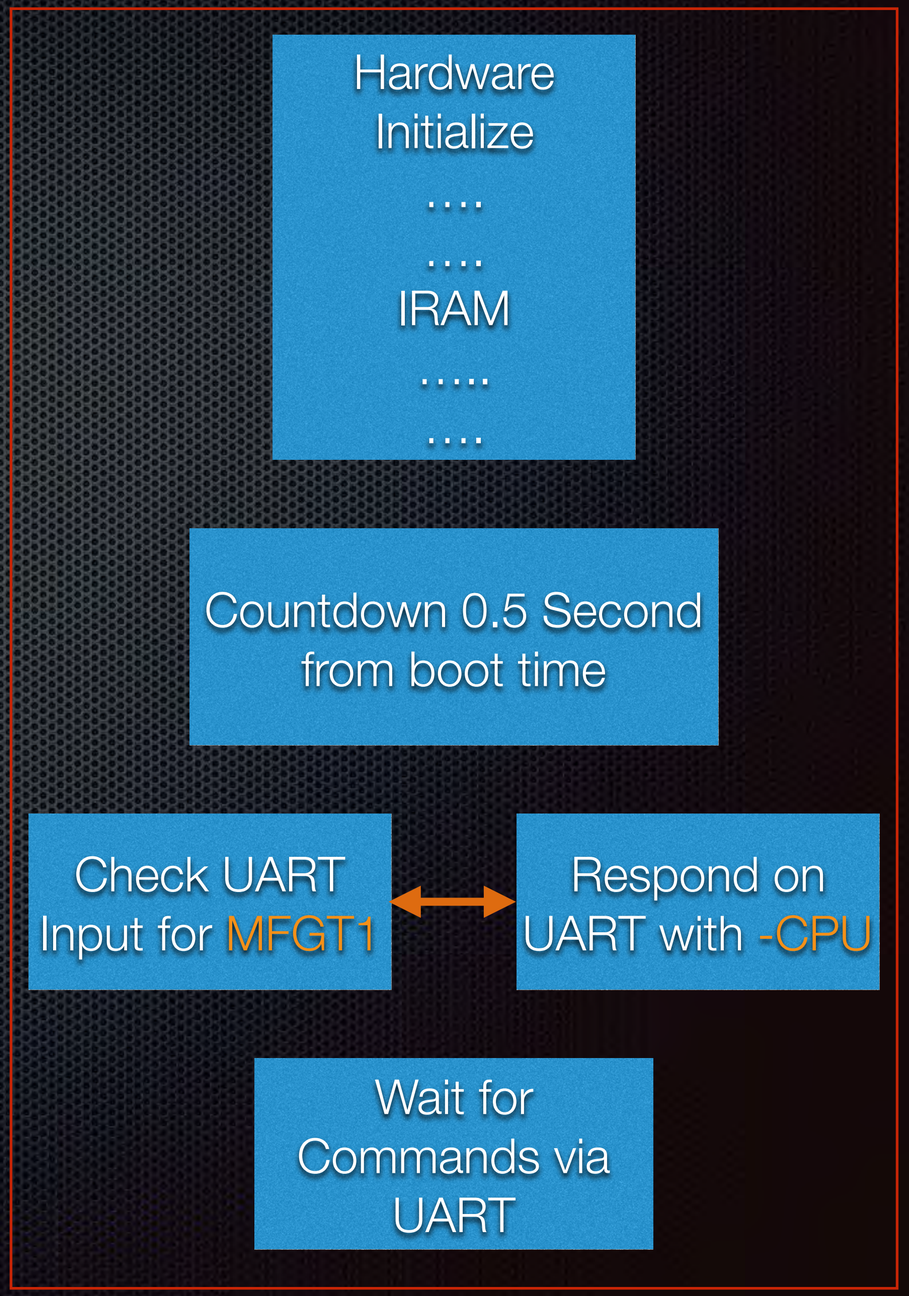




*Cojocar et al. Off-the-shelf Embedded Devices as Platforms for Security Research

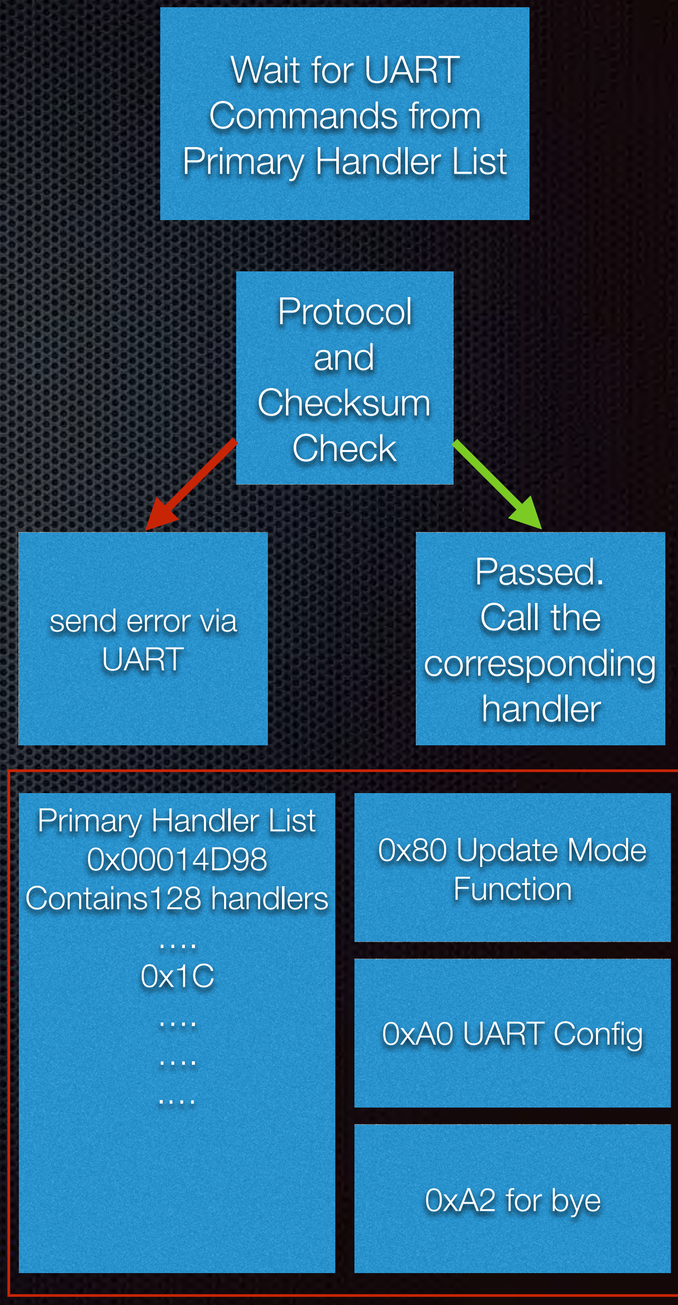
Special Access Feature

- Bootloader initializes the hardware
  - This includes copying some content from bootloader to IRAM.
- Only wait half a second from UART for :
  - **MFGT1** strings, possibly **M**it **F**reundlichen **G**rußen (German Greeting).
  - If PLC received “MFGT1” string it will acknowledge with **-CPU**
  - PLC now waits for Special Access Feature commands at 0xedf8.



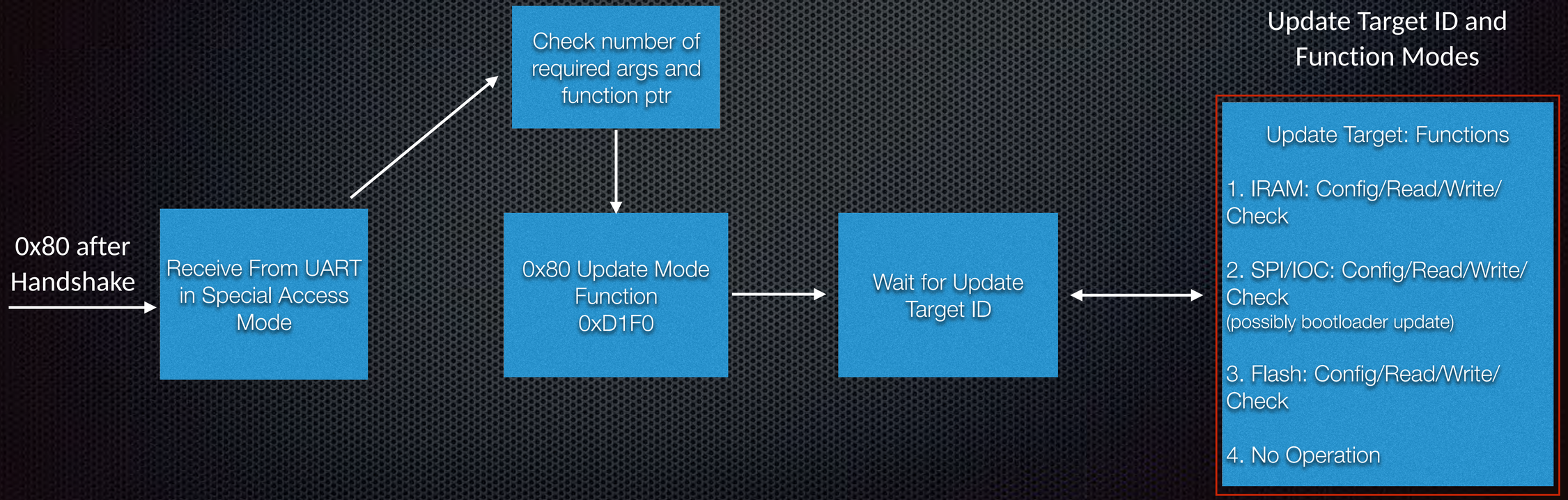
Primary Handlers After Handshake

Index	Input Length	Functionality
0	2	Get Version
1	2	Get firmware version
2	2	Check bootloader CRC
3	2	Check CRC flash part 2 (sectors 0 to 0x203), returns details on failure
4	2	Check IRAM (internal RAM) read/write
5	xx	Check TCM (Tightly Coupled Memory) read/write
6	3	Hardware tests for I2C, IOC, MAC
7	2	Some low level reset involving a temporay fiq handler
8	2	Some functionality related to I2C2
0c	6	Perform some read and crc calculation on flash memory, uses buf also used with ADC IOC
0d	2	Performs reads of static bootloader values and writes to some hardware mapped addresses
0e	3	Outputs some internal RAM values previously retrieved from static bootloader contents
11	2	Something related to MMC looping/checking
12	2	Some activation or waiting for I2C0
14	7	[!] Print (and seemingly update) current flash contents to(/from) UART.
16	2	Queries and returns some MMC bit.
17	2	Performs some lookup table based stuff in I2C1
18	3	Prepare reading flash update via UART to IRAM
19	>6	Read flash update via UART to IRAM
1a	2	[!] Commit flash update
1c	XX	**We will talk about it later**
30	XX	Query some CRC info about flash mode/part 2
31	7	[!] Performs an update of 8 bytes of flash number 2. This may be sending a length/crc pair
32	2	Check flash 2
..	xx	Prepares writes to EMB0 (plus some more ops on state structs).



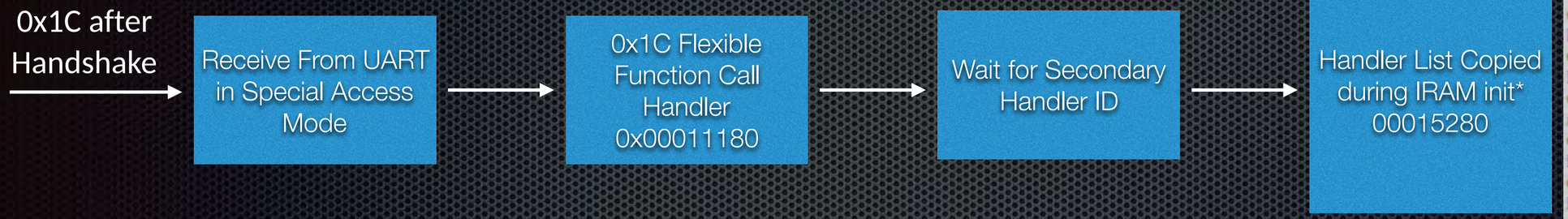
0x80 Handler, Update Mode Function

- **0x80** allows us to write to IRAM in Update Target 1 (IRAM)



0x1C Primary Handler

- This handler allows to call functions from a secondary list.



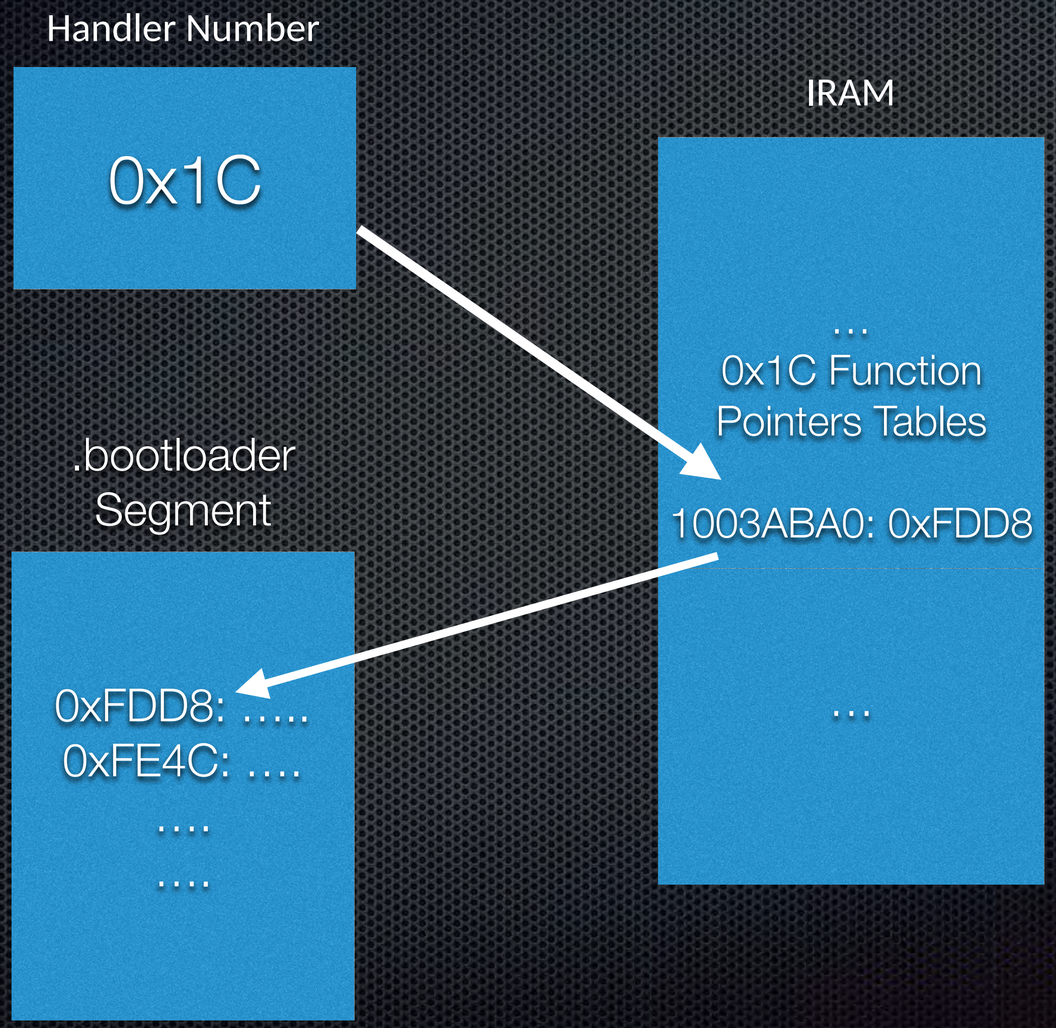
Bootloader: *Copied Function Table for 0x1c from bootloader (00015280) to IRAM 0x00000638

```
                        ; bootloader:off_6E0↑o
bl_UART_add_hook_00_some_setup
0
bl_UART_add_hook_01_perform_flash_write_list
0
bl_UART_add_hook_02_some_fill_arg_list
0
bl_UART_add_hook_03_set_80568300
0
nullsub_152
0
bl_UART_add_hook_05_get_80280000_byte_2
0
bl_UART_add_hook_06_query_some_dw
0
bl_UART_add_hook_07_query_some_dw
0
bl_UART_add_hook_08_wipe_chosen_flash_offset
0
bl_UART_add_hook_09_read_some_chosen_second_byte
0
bl_UART_add_hook_0a_write_crc_or_metadata_word
0
bl_UART_add_hook_0b_count_actual_data_dwords
7
bl_UART_add_hook_0c_count_some_configure_or_erase
0
bl_UART_add_hook_0d_some_find_dword_occurrences
0
bl_UART_add_hook_0e_some_read_flash
0
nullsub_153
0
bl_UART_add_hook_10_write_flash_with_incr_values
0
bl_UART_add_hook_11_read_fw_flash
3
bl_UART_add_hook_12_call_count_flash_occurrences
3
bl_UART_add_hook_13_some_wipe_flash
3
bl_UART_add_hook_14_call_count_flash_occurrences
3
bl_UART_add_hook_15_some_do_flash_update
3
bl_UART_add_hook_16_some_do_multiple_flash_update
3
bl_UART_add_hook_17_call_count_flash_occurrences
3
bl_UART_add_hook_18_multi_flash_reads
0
bl_UART_add_hook_19_nullsub_154
0
bl_UART_add_hook_1a_nullsub_155
```

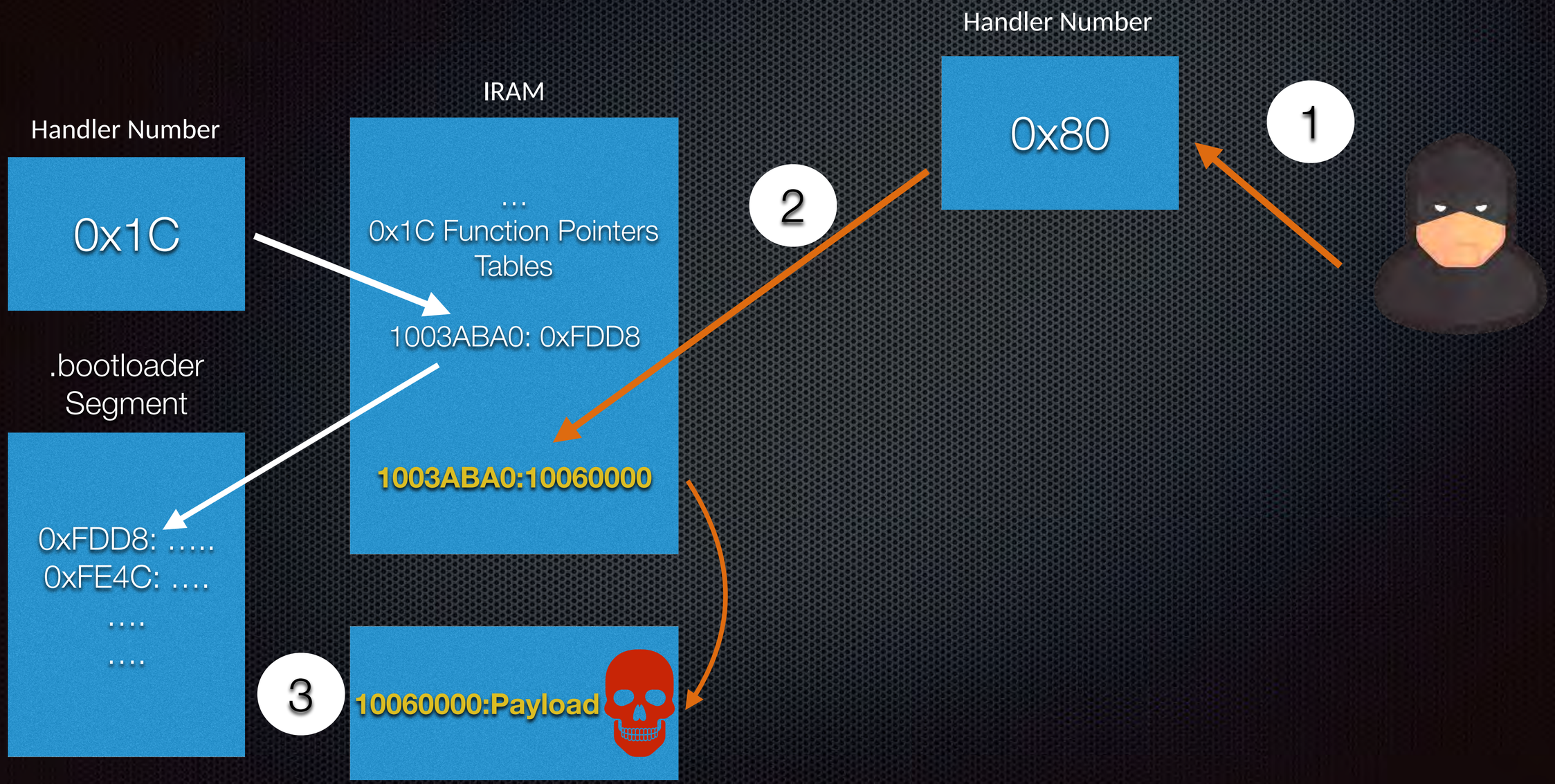
“Tobi, RCE Please!?”

–Ali Abbasi, A Guy Without UK Visa.

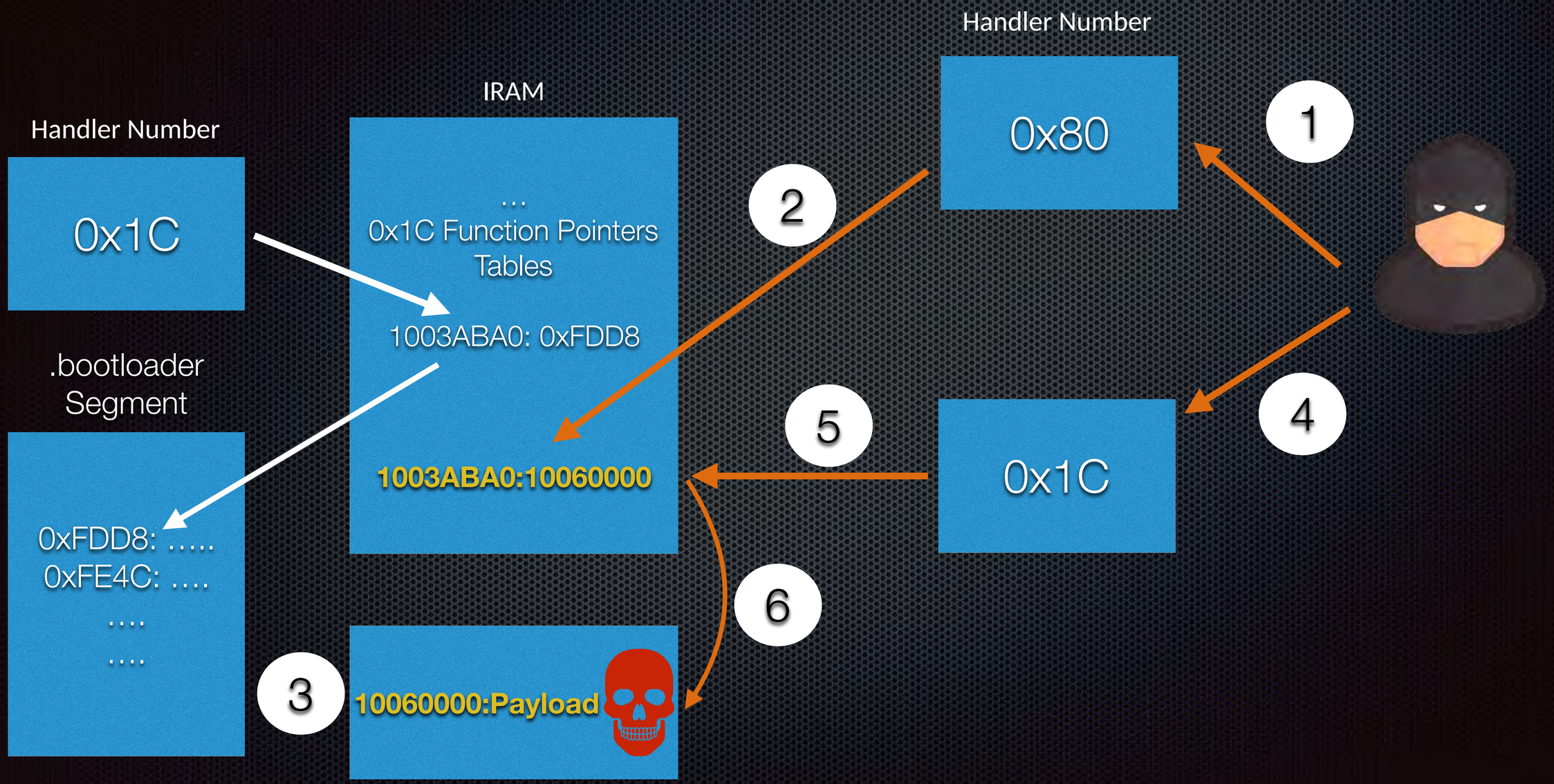
Siemens S7-1200 PLC Bootloader Arbitrary Code Execution



Siemens S7-1200 PLC Bootloader Arbitrary Code Execution



Siemens S7-1200 PLC Bootloader Arbitrary Code Execution



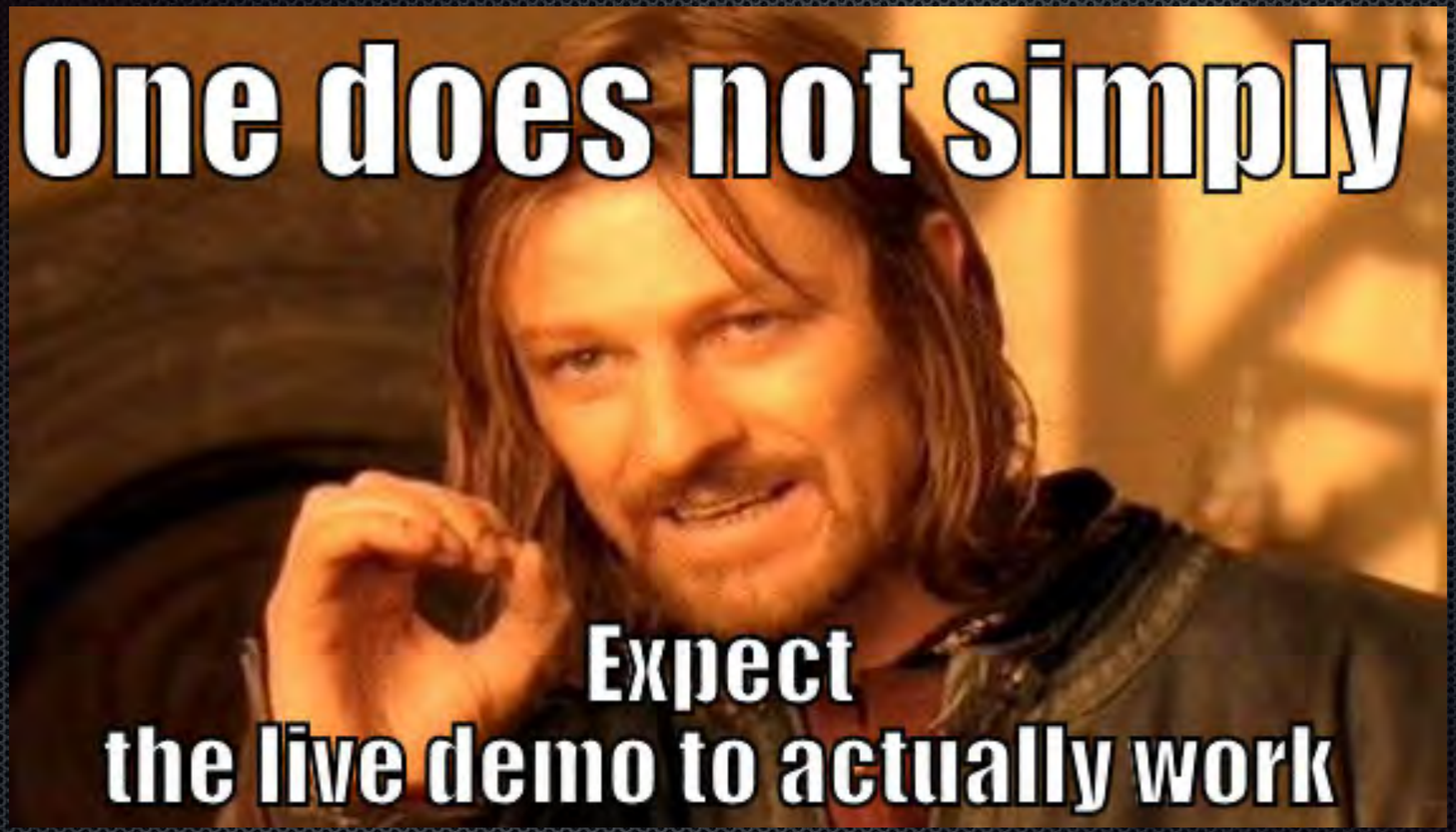
Siemens S7-1200 PLC Bootloader Arbitrary Code Execution

- Chaining everything together.
- Use 0x80 to write stager payload to the IRAM (update target 1, mode 3)
- Use 0x80 to add an additional item to handler list of 0x1C , pointing to our Stager payload.
- Now call 0x1C and….

Stager Payload

- Using Special Access only allows write to IRAM
- The write primitive is slow and error prone.
- During Special Access mode we are in privilege mode.
- Stager payload upgrades this to an arbitrary read/write primitive of the entire PLC memory
- We use this to inject arbitrary payloads.

DEMO Time



Conclusions and Future Works

- Programmable Logic Controllers are becoming more and more complex.
  - Built-in complexities such as the MWSL implementation can lead to security issues.
- Vendors upped their game by introducing certain hardening to their devices.
  - We observed other PLCs which had with similar legacy access features, but they did not have security features.
- We should push vendors to remove legacy access features which undermine their security ecosystem.
- At the same time, vendors should rethink the security via obscurity mindset when considering IP protection approaches.
- There are a lot of things to be done to make PLCs safer.

Special Thanks To

- Thomas Weber, Sec-Consult
- Alexandre Gazet, Airbus Cyber Security
- Marina Krotofil, BASF
- Lucian Cojocar from VU Amsterdam
- Nikita Golovliov, TU Eindhoven

“Everything That Has A Beginning Has An End”

The Matrix Revolutions

.

Looking for more technical details? Attend our talk in:
Chaos Communication Congress (36C3), Leipzig, Germany
SCADA Security Scientific Symposium (S4) in Miami USA


bl4ckic3 ali@ali.re


ScepticCtf tobias.scharnowski@rub.de


ThorstenHolz thorsten.holz@rub.de