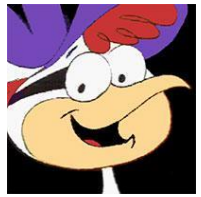
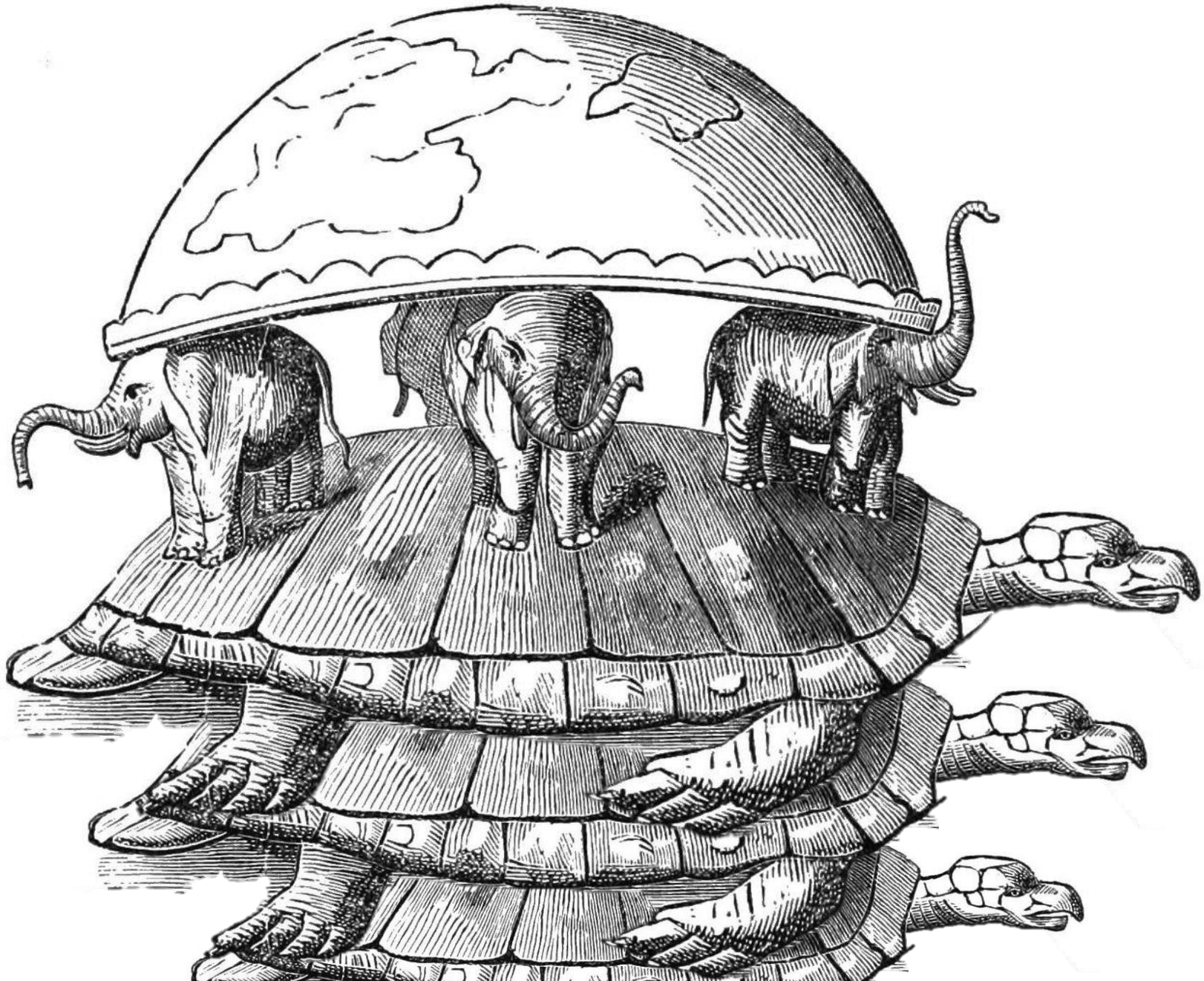
It’s 8051s all the way down

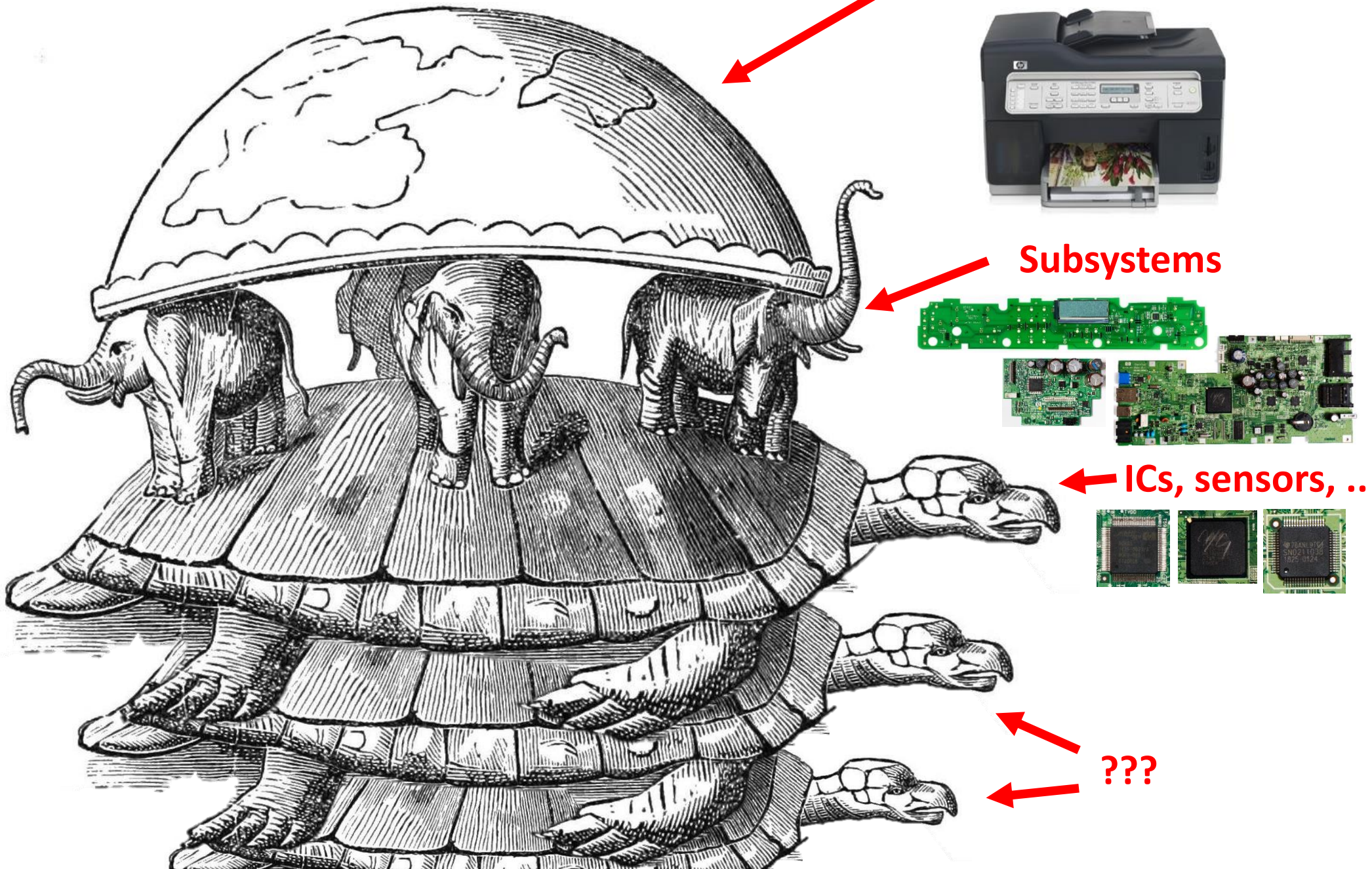
Reverse engineering the SiLabs EZRadioPRO

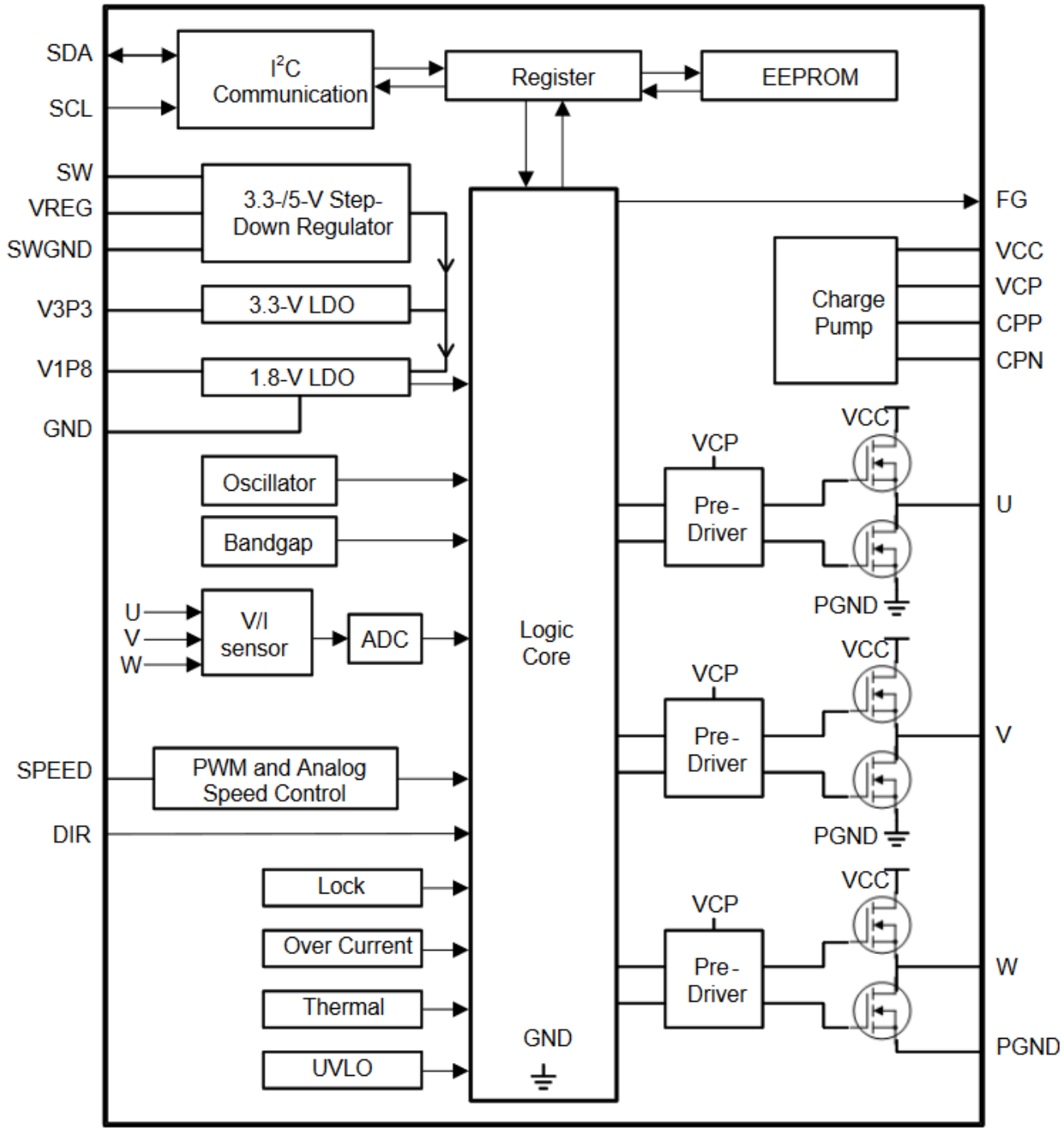


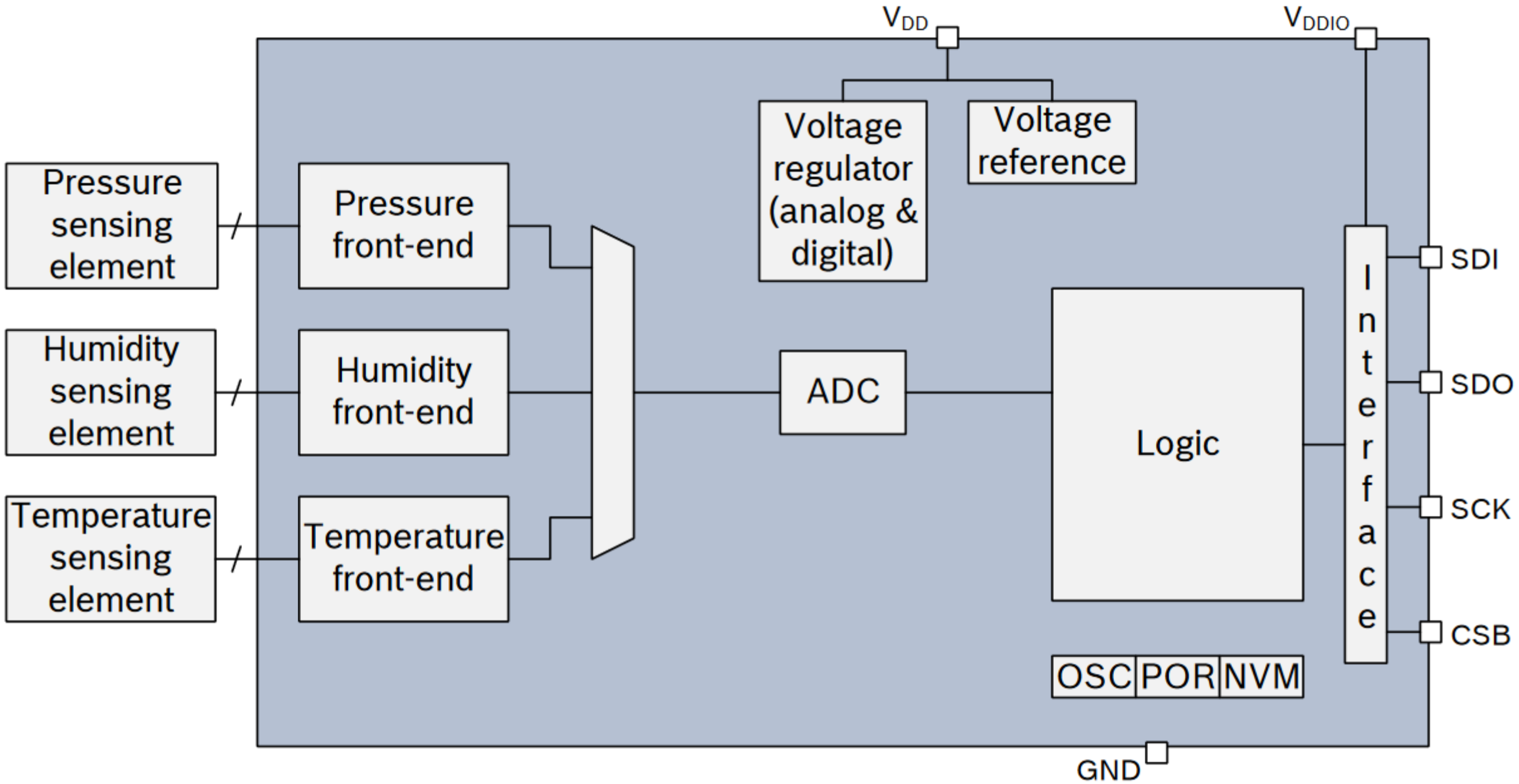
https://github.com/astuder/Inside-EZRadioPRO

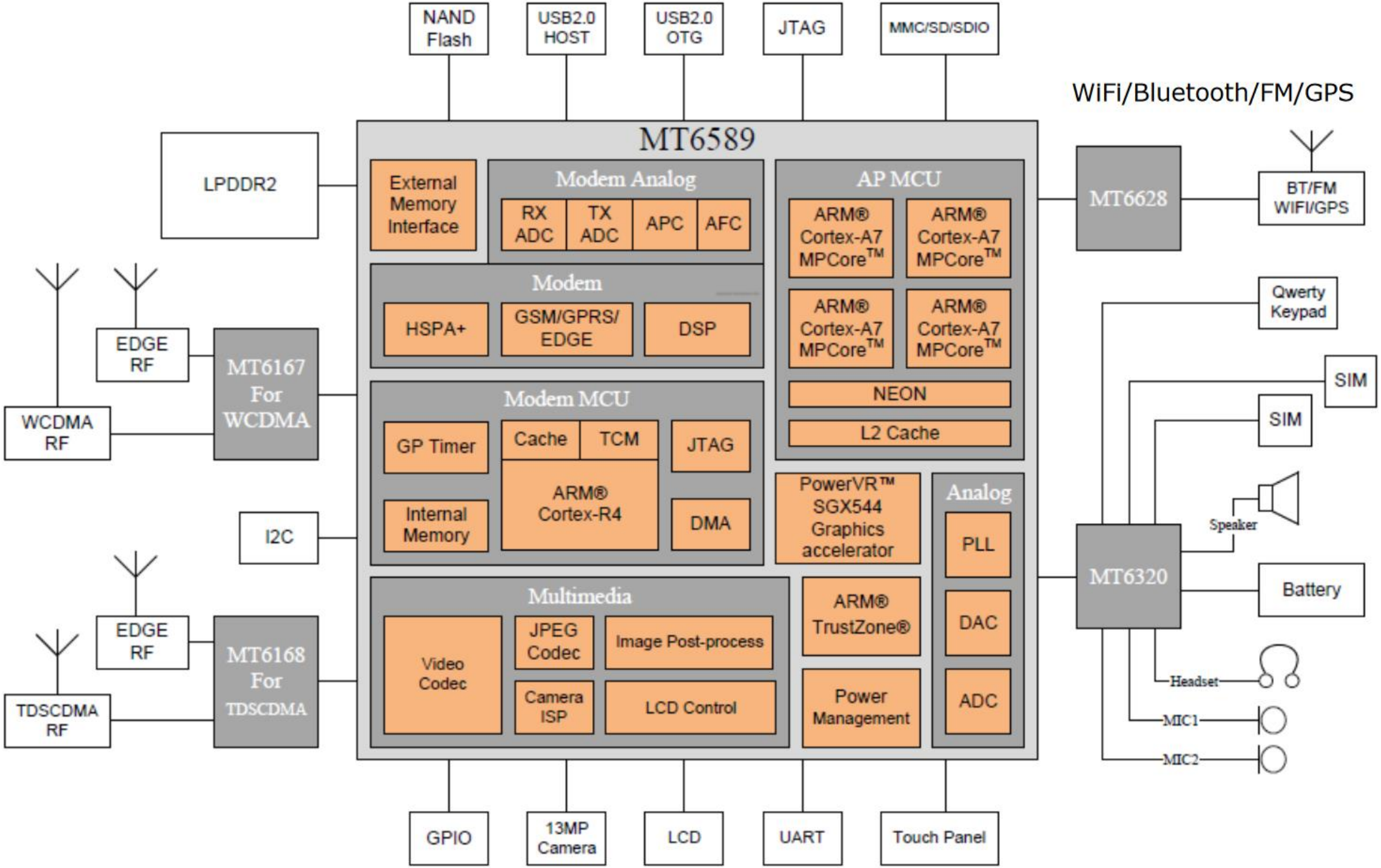
@adistuder




Electronic device
Subsystems
ICs, sensors, ..
???

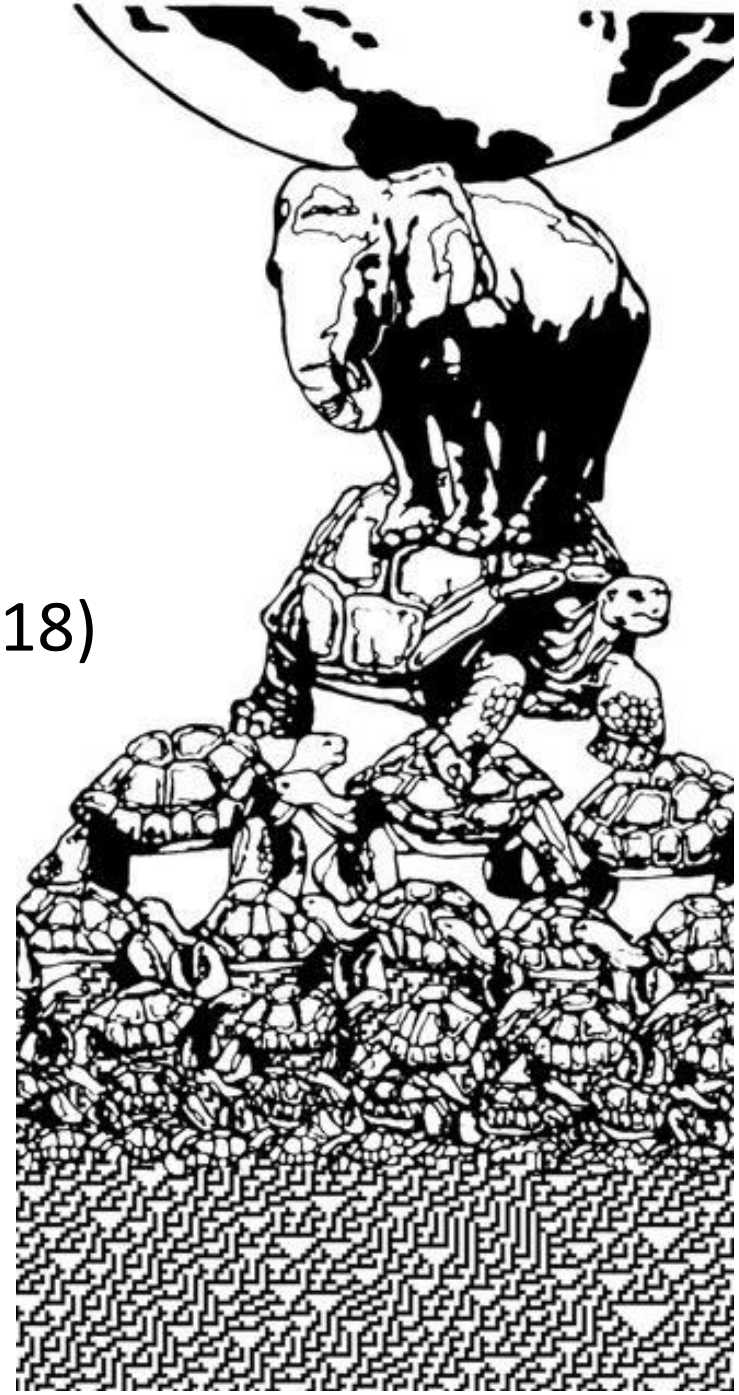

SDA
SCL
I²C Communication
Register
EEPROM
SW
VREG
SWGND
3.3-/5-V Step-Down Regulator
V3P3
3.3-V LDO
V1P8
1.8-V LDO
GND
FG
VCC
VCP
CPP
CPN
Charge Pump
Oscillator
Bandgap
U
V
W
V/I sensor
ADC
Logic Core
VCP
Pre-Driver
VCC
U
PGND
V
SPEED
PWM and Analog Speed Control
DIR
Lock
Over Current
Thermal
UVLO
GND
W
PGND


V_{DD}
V_{DDIO}
Voltage regulator (analog & digital)
Voltage reference
Pressure sensing element
Humidity sensing element
Temperature sensing element
Pressure front-end
Humidity front-end
Temperature front-end
ADC
Logic
Interface
SDI
SDO
SCK
CSB
OSC
POR
NVM
GND


NAND Flash
USB2.0 HOST
USB2.0 OTG
JTAG
MMC/SD/SDIO
WiFi/Bluetooth/FM/GPS
MT6589
LPDDR2
External Memory Interface
Modem Analog
RX ADC
TX ADC
APC
AFC
AP MCU
ARM® Cortex-A7 MPCore™
ARM® Cortex-A7 MPCore™
ARM® Cortex-A7 MPCore™
ARM® Cortex-A7 MPCore™
NEON
L2 Cache
Modem
HSPA+
GSM/GPRS/ EDGE
DSP
Modem MCU
GP Timer
Cache
TCM
JTAG
ARM® Cortex-R4
Internal Memory
DMA
PowerVR™ SGX544 Graphics accelerator
Analog
PLL
DAC
ADC
Multimedia
Video Codec
JPEG Codec
Image Post-process
Camera ISP
LCD Control
ARM® TrustZone®
Power Management
MT6628
BT/FM WIFI/GPS
MT6320
Qwerty Keypad
SIM
SIM
Speaker
Battery
Headset
MIC1
MIC2
EDGE RF
MT6167 For WCDMA
WCDMA RF
I2C
EDGE RF
MT6168 For TDSCDMA
TDSCDMA RF
GPIO
13MP Camera
LCD
UART
Touch Panel

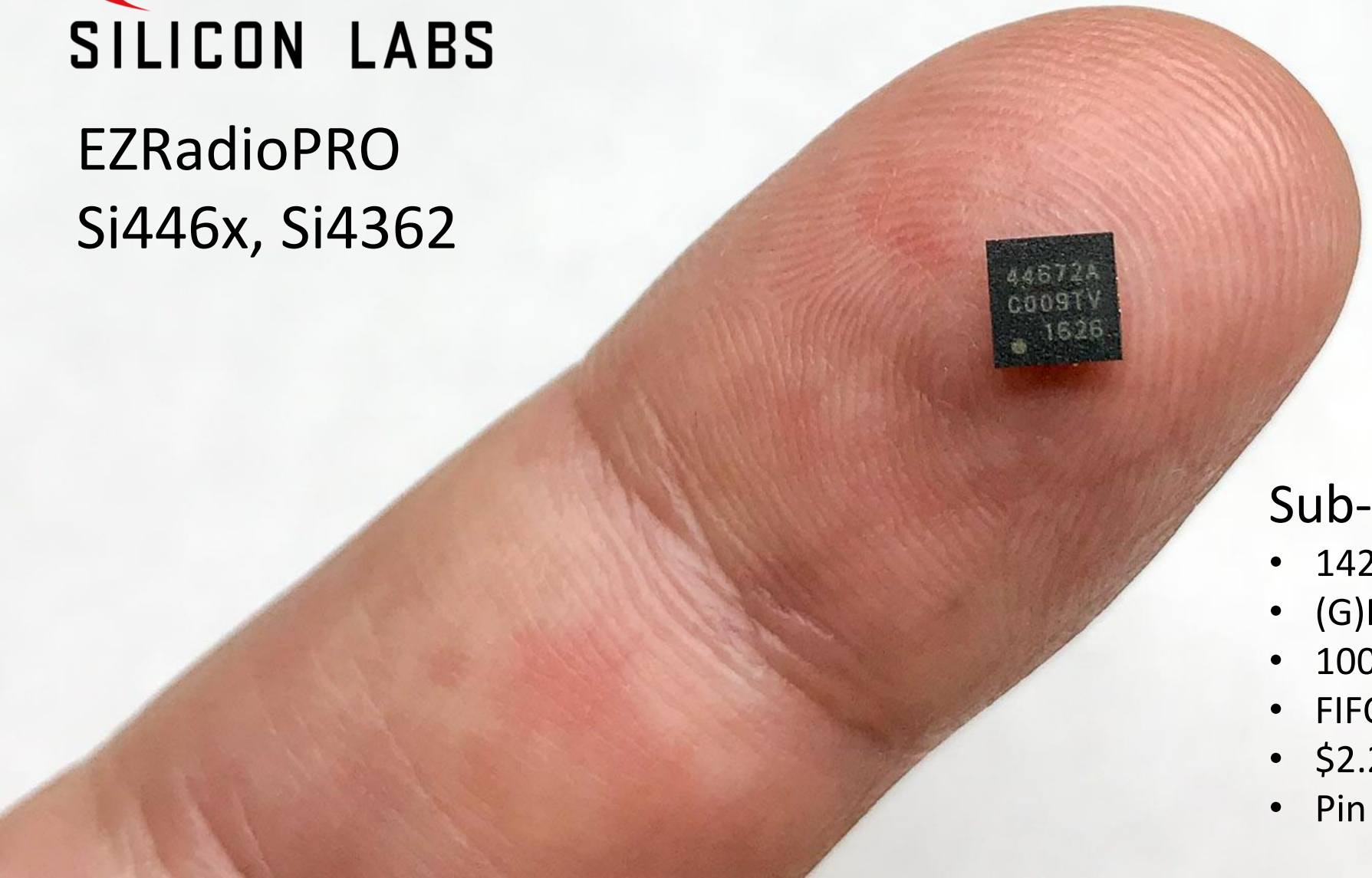
It’s MCUs all the way down

- Intel Management Engine (IME)
- SD Cards (Bunnie & xobs @30c3)
- SD Card w/ WiFi (Guillaume Valadon @Black Hat 2018)
- eMMC (eMMC sudden death @xdadevelopers)
- Radio & MCU combos (WiFi, BT, baseband..)
- …



SILICON LABS

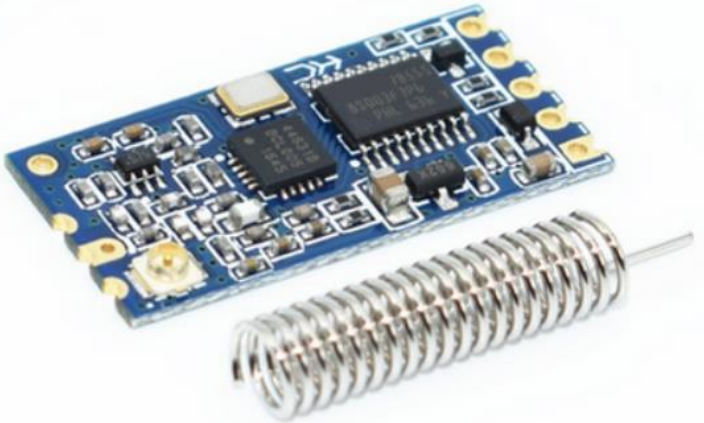
EZRadioPRO
Si446x, Si4362



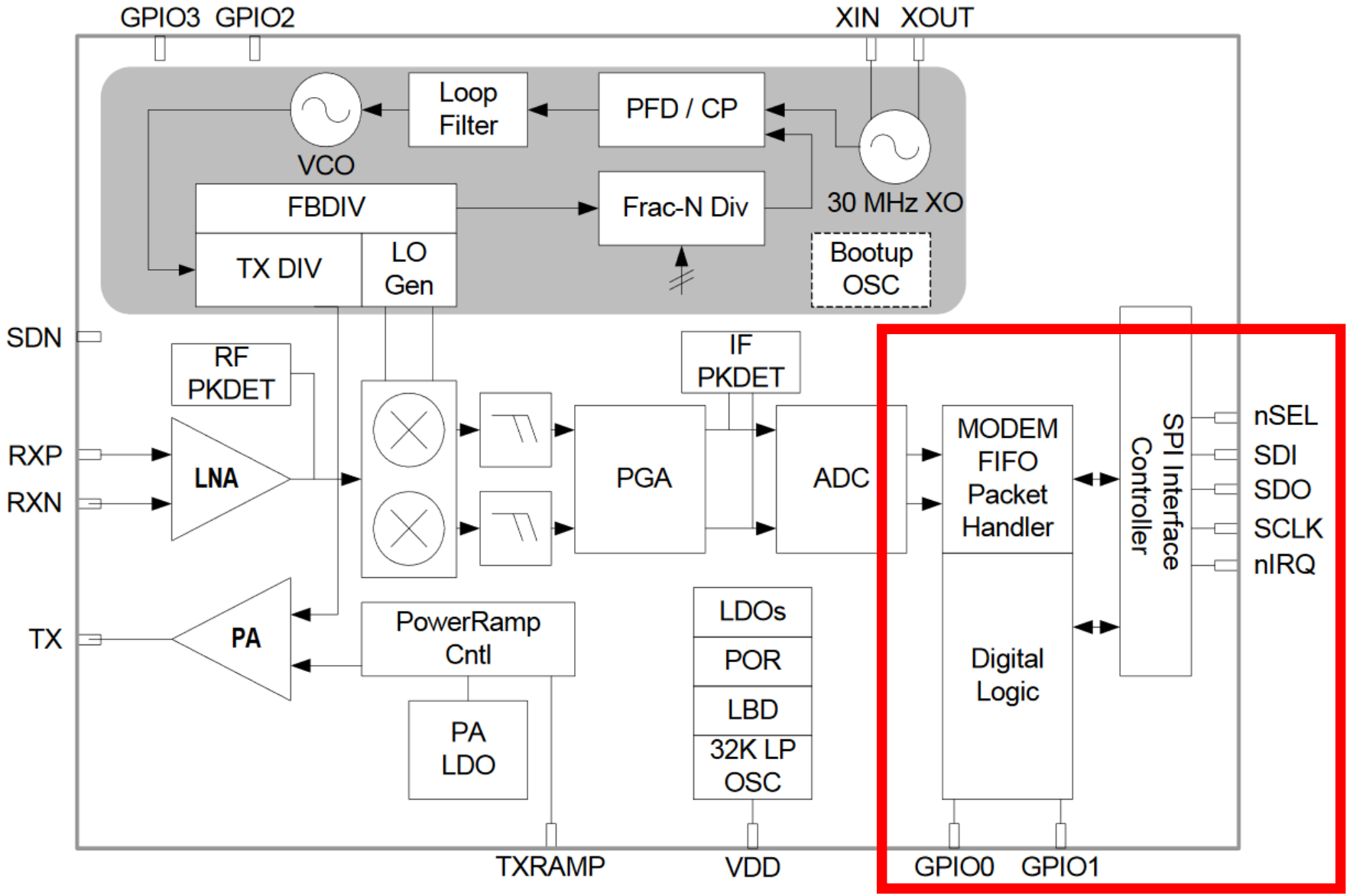
Sub-GHz radios

- 142-1040 MHz
- (G)FSK, 4(G)FSK, (G)MSK, OOK
- 100-1M bps
- FIFO, packet handler, CRC, …
- $2.20-$2.50 in single quantity
- Pin compatible across family

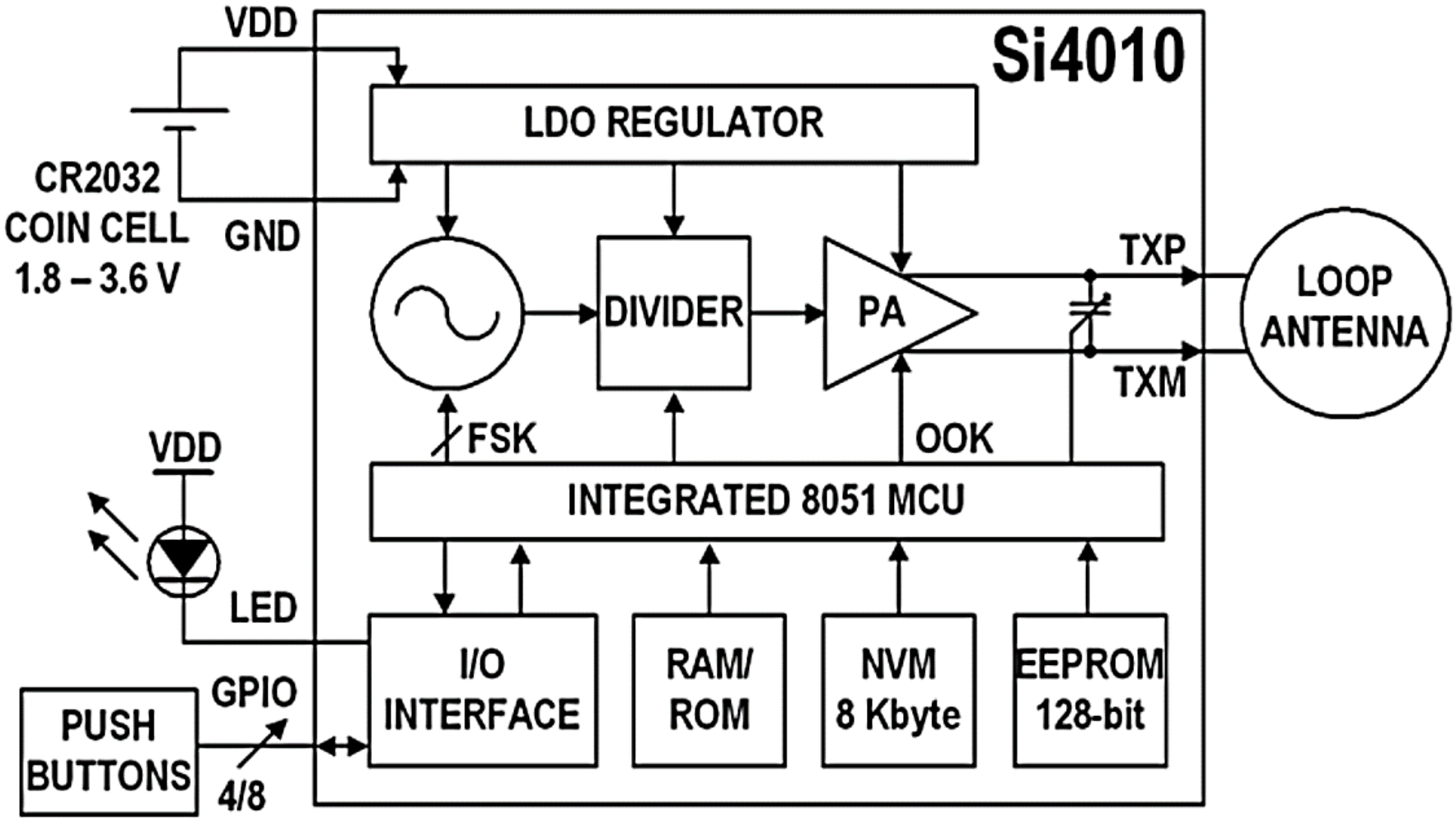
- Xyloband
- goTenna, goTenna MESH
- Sigfox modules (TD next TD120x)
- Cheap AIS receivers
- High-altitude balloon trackers (APRS transmitters)
- 433 MHz modules (“Si4463 module” or HC-12 on Ebay, AliExpress)



Architecture according to the datasheet



Architecture of Si4010 (EZRadio)



Silabs Si4010-C2 datasheet
https://www.silabs.com/documents/public/data-sheets/Si4010.pdf

Google Patents

(radio) (8051) assignee:(Silicon Laboratories)

Sign In

SEARCH TERMS

radio × or + Synonym

8051 × or + Synonym

+ Synonym

SEARCH FIELDS

Date · Priority

YYYY-MM-DD — YYYY-MM-DD

+ Inventor

Silicon Laboratories × or

+ Assignee

Patent Office · Language

Status · Type

Litigation

About 44 results

Download · Side-by-side

Sort by · Relevance · Grouped by · None · Results / page · 10

Method and apparatus for symbol synchronization for an 802.15.4 **radio** platform

US • US8223820B2 • Alexandre Rouxel • **Silicon Laboratories** Inc.

Priority 2007-12-31 • Filing 2008-03-31 • Grant 2012-07-17 • Publication 2012-07-17

A technique for receiving a data stream including a spreading sequence packet of information containing a data payload and, in addition to the data payload, packet overhead including at least periodic information and at least one unique section of known coded information that defines a unique ...

Hardware synchronizer for 802.15.4 **radio** to minimize processing power ...

US • US8023557B2 • Nicolas Constantinidis • **Silicon Laboratories** Inc.

Priority 2007-12-31 • Filing 2007-12-31 • Grant 2011-09-20 • Publication 2011-09-20

FIG. 12 , there is illustrated a diagrammatic view of the MCU system. At the heart of the MCU system is the **8051** processor core 120 . This is connected ... 1. A method for controlling the operation of a low power **radio** platform that realizes the physical layer (PHY) with a software portion and an analog ...

Single chip low power fully integrated 802.15.4 **radio** platform

US • US8050313B2 • Nicolas Constantinidis • **Silicon Laboratories** Inc.

Priority 2007-12-31 • Filing 2007-12-31 • Grant 2011-11-01 • Publication 2011-11-01

At the MAC layer, the processing is less intense and, therefore, a standard architecture such as the **8051** architecture, a well known architecture, ... 8. The **radio** platform of claim 7 , wherein said periodic management circuitry includes calibration circuitry to calibrate the frequency of the clock ...

Home electrical device control within a wireless mesh network

WO US JP • US9166812B2 • Peter Shorty • Sigma Designs, Inc.

Priority 2006-01-31 • Filing 2007-04-19 • Grant 2015-10-20 • Publication 2015-10-20

In embodiments of the present invention improved capabilities are described for associating a first node in a mesh network with an electrical device, wherein the electrical device comprises a home control device, enabling the mesh network to perform channel adjustment, transmitting data through ...

Audio-visual system control using a mesh network

Architecture according to SiLabs patent

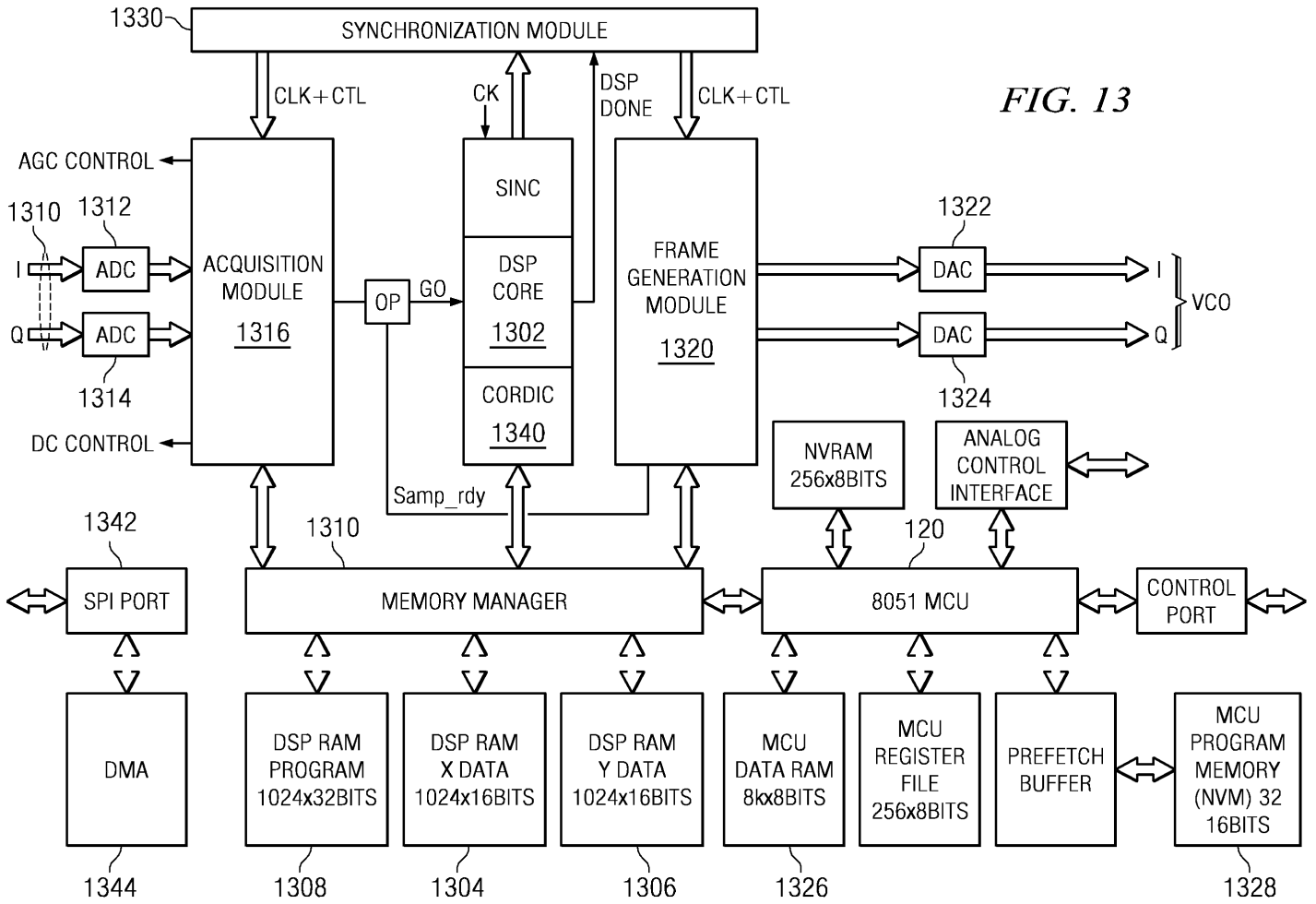


FIG. 13

Single chip low power fully integrated 802.15.4 radio platform
https://patents.google.com/patent/US8050313B2

In a related patent…

image. Note that in various implementations, multiple firmware versions may be generated such that different devices formed of a single wafer or within one or more wafer lots can each be programmed with different firmware versions. These firmware versions may thus provide different functionality, allowing a single mask set to be used to produce devices having different functionalities, which may provide for sales at different price points.

Still referring to FIG. **3**, during marketing of a chip, which may occur over a number of months or years, it may be

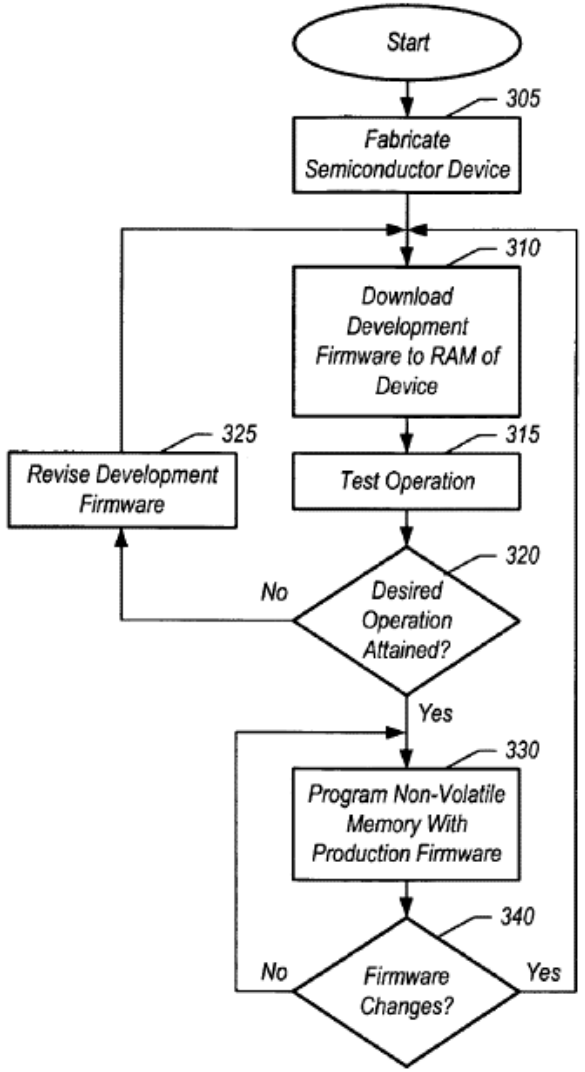


FIG. 3

Digital architecture using one-time programmable (OTP) memory
https://patents.google.com/patent/US7613913B2

There are patches!

"If a bug is identified it is usually fixed with a patch on the current revision, and the change will be implemented on the next revision. Also, we can test future features of the chip by patching the radio."

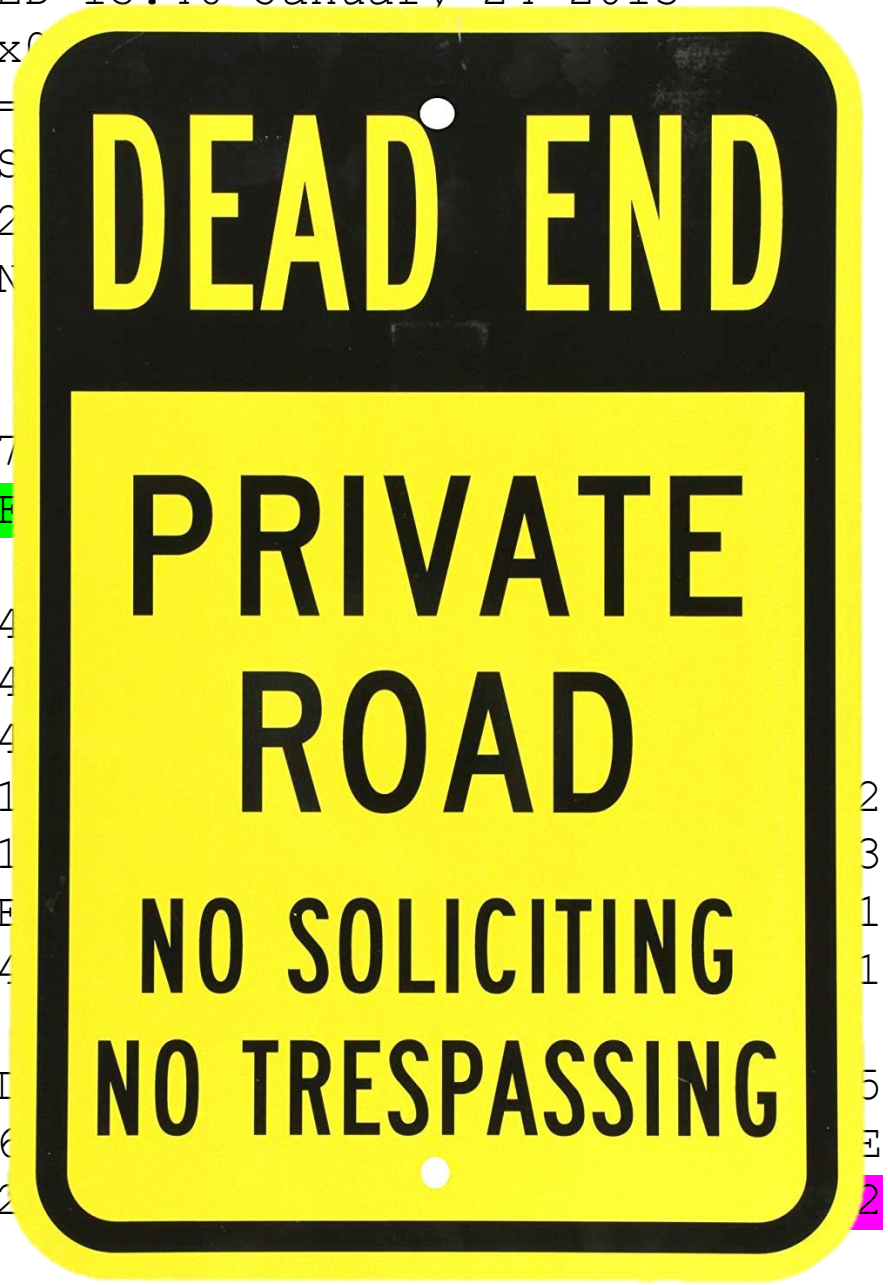
https://www.silabs.com/community/wireless/bluetooth/forum.topic.html/apply_patch_to_si446-c7pE

```
// GENERATED=15:46 January 24 2013
// ROMID=0x03
// PATCHID=0xF692
// REQUIRES=NONE
// SIZE=2128
// FUNCTION=MAIN
// MAJOR=3
// MINOR=0
// BUILD=27
// CRCT=0xF776

#define SI446X_PATCH_ROMID  03
#define SI446X_PATCH_ID     27
#define SI446X_PATCH_CMDS                       \
{ 0x04,0x11,0xF7,0x76,0x00,0x00,0xA6,0x82 }, \
{ 0x05,0x61,0xE6,0x82,0x5E,0xB7,0xFB,0x93 }, \
{ 0x05,0x1E,0x12,0xBD,0x5A,0xC2,0x52,0x41 }, \
{ 0xE7,0xF4,0xDF,0x6A,0x24,0xD9,0xBA,0x31 }, \
…
{ 0xEF,0x7D,0x0D,0xB5,0xCF,0x00,0xC5,0x75 }, \
{ 0xE3,0xC6,0x0E,0x0B,0x10,0x44,0x10,0xEE }, \
{ 0x05,0x12,0x86,0x0D,0xC0,0xA5,0xF6,0x92 }
```

```
// GENERATED=15:46 January 24 2013
// ROMID=0x
// PATCHID=
// REQUIRES
// SIZE=212
// FUNCTION
// MAJOR=3
// MINOR=0
// BUILD=27
// CRCT=0xE

#define SI4
#define SI4
#define SI4                                                    \
{  0x04,0x11                                                2 }, \
{  0x05,0x61                                                3 }, \
{  0x05,0x1E                                                1 }, \
{  0xE7,0xF4                                                1 }, \
...
{  0xEF,0x7D                                                5 }, \
{  0xE3,0xC6                                                E }, \
{  0x05,0x12                                                2 }
```

PATCH_IMAGE

PATCH_ARGS,
PATCH_DATA?



265 lines
= 1-2k of code?

SiLabs knowledge base

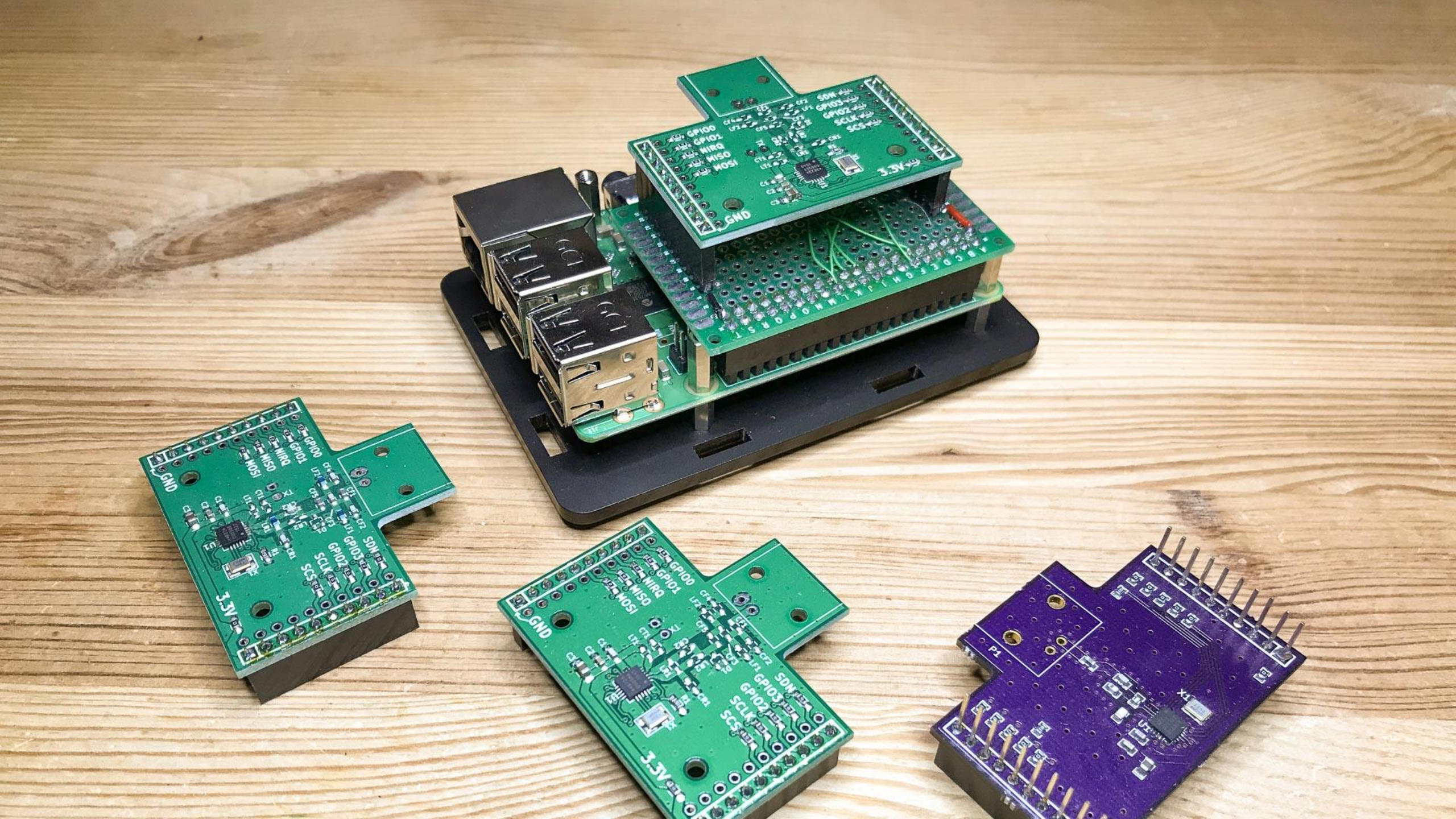
Q: How do I retrieve and re-apply the results of IQ calibration on Si446x?

A: [..] special API command that allows direct register access is referred to as POKE for writing registers and PEEK for reading [..]

The command code for PEEK is 0xF0 and for POKE it is 0xF1. The two byte addresses of the registers of concern are the following: iq_calamp – 0x00 0xD6; iq_calph – 0x00 0xD7.

https://www.silabs.com/community/wireless/proprietary/knowledge-base.entry.html/2014/05/12/si446x_iq_calibratio-ybHg


**** COMMODORE 64 BASIC V2 ****
64K RAM SYSTEM 38911 BASIC BYTES FREE
READY.
10 A=0
20 PRINT PEEK(A)
30 A=A+1
40 IF A<65536 GOTO 20
RUN


GPIO0
GPIO1
NIRQ
MISO
MOSI
GND
SDN
GPIO3
GPIO2
SCLK
SCS
3.3V
P1

Peeking full 16-bit address range

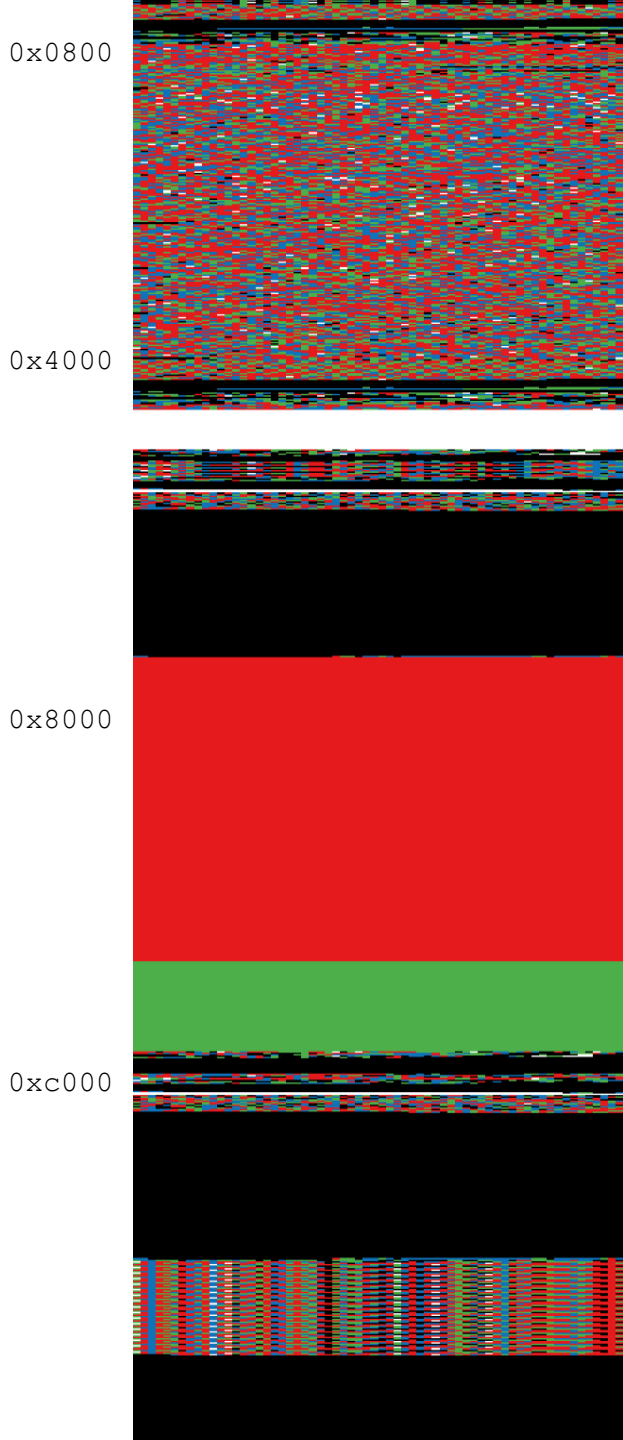
- Raspberry Pi
- Breakout boards for radio ICs
- Python libraries: rpi.gpio, py-spidev

```
resetRadio()
sendCommand(POWER_UP)
for addr in range(0,65535):
    print sendCommand(PEEK,addr)
```

```
00000000: 0000 0000 0000 5000 0000 0000 0000 04ff  ......P.........
00000010: 0000 1e40 4f76 07ff 0008 0000 0000 0000  ...@Ov..........
00000020: 0000 0000 1000 0000 6e00 0032 00bf 0c00  ........n..2....
00000030: 0002 0014 0000 0000 d200 0000 0000 0000  ................
00000040: 9e3c 0000 0000 0014 0015 0000 0000 0000  .<..............
00000050: 0000 0000 0000 0000 0000 000c ff03 e002  ................
00000060: 000f 4240 0406 d300 003b 2800 0080 0800  ..B@.....;(.....
00000070: 0010 004b c6d3 a006 d382 0400 0000 0041  ...K...........A
00000080: 0000 0000 0000 5000 0000 0000 0000 04ff  ......P.........
00000090: 0000 1e40 4f76 07ff 0008 0000 0000 0000  ...@Ov..........
000000a0: 0000 0000 1000 0000 6e00 0032 00bf 0c00  ........n..2....
000000b0: 0002 0014 0000 0000 d200 0000 0000 0000  ................
000000c0: 9e3c 0000 0000 0014 0015 0000 0000 0000  .<..............
000000d0: 0000 0000 0000 0000 0000 000c ff03 e002  ................
000000e0: 000f 4240 0406 d300 003b 2800 0080 0800  ..B@.....;(.....
000000f0: 0010 004b c6d3 a006 d382 0400 0000 0041  ...K...........A
00000100: cda6 02bb a502 cf5b 02ba 2702 c54b 02cd  .......[..'..K..
00000110: 4102 b09b 02d4 1402 a313 02b1 e202 c022  A..............."
00000120: 02c1 8202 0367 0290 0102 0358 02cd b302  .....g.....X....
00000130: aa63 028f e002 ce17 02cd 8602 aaa3 02ce  .c..............
00000140: 5202 bd85 0200 4f02 a020 02d5 f502 a657  R.....O.. .....W
00000150: 02b3 7102 96f4 02b7 c502 d5dc 02b6 ad02  ..q.............
00000160: 98e9 02c0 f502 9736 02bf 1502 c50d 02a7  .......6........
00000170: d702 c5a1 02c0 8002 b92e 02d2 5302 9cfc  ............S...
00000180: 02b6 c502 ba40 02cd b002 cd58 02c3 6102  .....@.....X..a.
00000190: 0046 02cd 3c02 9e09 02b6 1702 94c9 029c  .F..<...........
000001a0: 0102 9ae3 02a9 4902 d74c 02bb ae02 9cd1  ......I..L......
000001b0: 02bd 0402 c683 02b1 6e02 97ff 028f 4302  ........n.....C.
000001c0: b07e 0203 7702 032f 02ab 4302 b560 0203  .~..w../..C..`..
000001d0: 4b02 c000 02be 9502 c7c3 02aa 3202 b3f2  K...........2...
000001e0: 02b7 4a02 a881 02d3 4b02 cdc6 02ce a202  ..J.....K.......
000001f0: c0b5 02b8 0002 b230 0296 6c02 9acf 0298  .......0..l.....
00000200: 9702 9a19 02b0 3e02 b13e 02a9 c402 97ef  ......>..>......
```

Reverse engineering tools

- https://binvis.io
- Disassembler
- RE application
  - IDA Pro
  - radare2
  - Ghidra
  - …



Free reverse engineering tools

radare2

- Command line UX
- Very extendable (Python, gdb..)
- Very active dev community
- GUI project: Cutter
- Weak project management
- Weak decompiler(s)

Ghidra

- Newly open-sourced
- GUI-centric
- Project management features
- Great decompiler
- Do you trust the NSA?

Data

```
0x00000000  0000 0000 0000 5000 0000 0000 0000 04ff  ......P.........
0x00000010  0000 1e40 4f77 07ff 0008 0000 0000 0000  ...@Ow..........
0x00000020  0000 0000 1000 0000 6e00 0032 00bf 0c00  ........n..2....
0x00000030  0002 0014 0000 0000 d200 0000 0000 0000  ................
0x00000040  9e3c 0000 0000 0014 0015 0000 0000 0000  .<..............
0x00000050  0000 0000 0000 0000 0000 000c ff03 e002  ................
0x00000060  000f 4240 0406 d300 003b 2800 0080 0800  ..B@.....;(.....
0x00000070  0010 004b c6d3 a006 d382 0400 0000 0041  ...K...........A
0x00000080  0000 0000 0000 5000 0000 0000 0000 04ff  ......P.........
0x00000090  0000 1e40 4f77 07ff 0008 0000 0000 0000  ...@Ow..........
0x000000a0  0000 0000 1000 0000 6e00 0032 00bf 0c00  ........n..2....
0x000000b0  0002 0014 0000 0000 d200 0000 0000 0000  ................
0x000000c0  9e3c 0000 0000 0014 0015 0000 0000 0000  .<..............
0x000000d0  0000 0000 0000 0000 0000 000c ff03 e002  ................
0x000000e0  000f 4240 0406 d300 003b 2800 0080 0800  ..B@.....;(.....
0x000000f0  0010 004b c6d3 a006 d382 0400 0000 0041  ...K...........A
```

8051 code

```
0x00000800      f2          movx @r0, a
0x00000801      22          ret
0x00000802      a4          mul ab
0x00000803      2582        add a, dpl
0x00000805      f582        mov dpl, a
0x00000807      e5f0        mov a, b
0x00000809      3583        addc a, dph
0x0000080b      f583        mov dph, a
0x0000080d      22          ret
0x0000080e      c2d5        clr psw.5
0x00000810      30f707      jnb b.7, 0x081a
0x00000813      b2d5        cpl psw.5
0x00000815      63f0ff      xrl b, #0xff
0x00000818      05f0        inc b
0x0000081a      30e70c      jnb acc.7, 0x0829
0x0000081d      b2d5        cpl psw.5
0x0000081f      f4          cpl a
0x00000820      04          inc a
0x00000821      84          div ab
```

8051 vector table

```
0x00004000      02cc43          ljmp 0xcc43
0x00004003      02cc4c          ljmp 0xcc4c
0x00004006      00              nop
0x00004007      02cc55          ljmp 0xcc55
0x0000400a      00              nop
0x0000400b      0202eb          ljmp 0x02eb
0x0000400e      22              ret
0x0000400f      02cc5e          ljmp 0xcc5e
0x00004012      22              ret
0x00004013      02cc67          ljmp 0xcc67
0x00004016      22              ret
0x00004017      02cc36          ljmp 0xcc36
0x0000401a      00              nop
0x0000401b      02ccc1          ljmp 0xccc1
0x0000401e      00              nop
0x0000401f      02cc70          ljmp 0xcc70
0x00004022      00              nop
0x00004023      02cc79          ljmp 0xcc79
0x00004026      00              nop
```

Entry point: reset vector

```
0x00004000      02cc43          ljmp 0xcc43
0x00004003      02cc4c          ljmp 0xcc4c
0x00004006      00              nop
0x00004007      02cc55          ljmp 0xcc55
0x0000400a      00              nop
0x0000400b      0202eb          ljmp 0x02eb
0x0000400e      22              ret
0x0000400f      02cc5e          ljmp 0xcc5e
0x00004012      22              ret
0x00004013      02cc67          ljmp 0xcc67
0x00004016      22              ret
0x00004017      02cc36          ljmp 0xcc36
0x0000401a      00              nop
0x0000401b      02ccc1          ljmp 0xccc1
0x0000401e      00              nop
0x0000401f      02cc70          ljmp 0xcc70
0x00004022      00              nop
0x00004023      02cc79          ljmp 0xcc79
0x00004026      00              nop
```

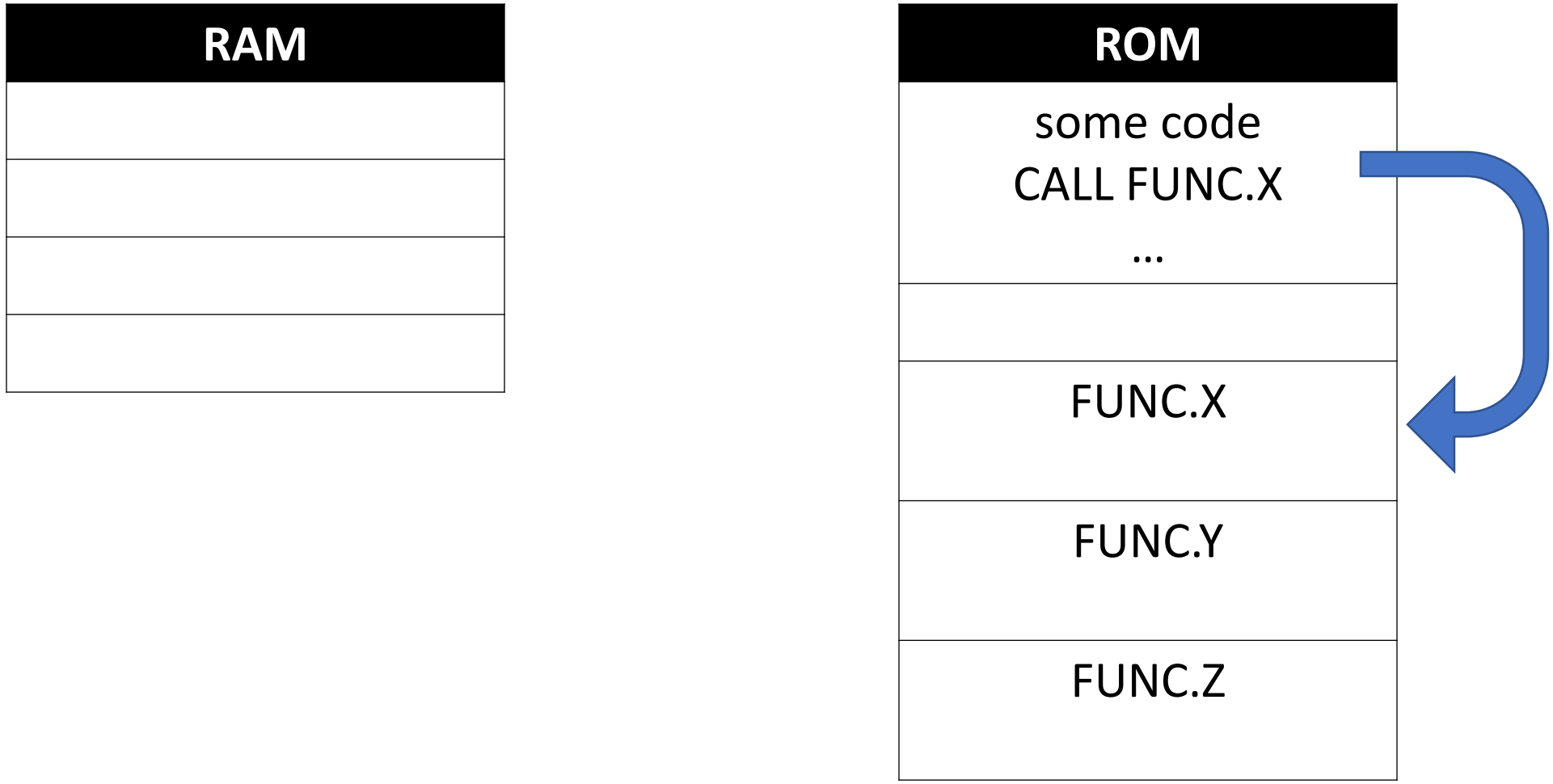
Entry point: stack initialization

```
0x00000b98      75819f          mov sp, #0x9f
0x00000b9b      12028b          lcall 0x028b
0x00000b9e      02024f          ljmp 0x024f
0x00000ba1      75819f          mov sp, #0x9f
0x00000ba4      1201a4          lcall 0x01a4
0x00000ba7      02024f          ljmp 0x024f

0x0000028b      02a97b          ljmp 0xa97b
```

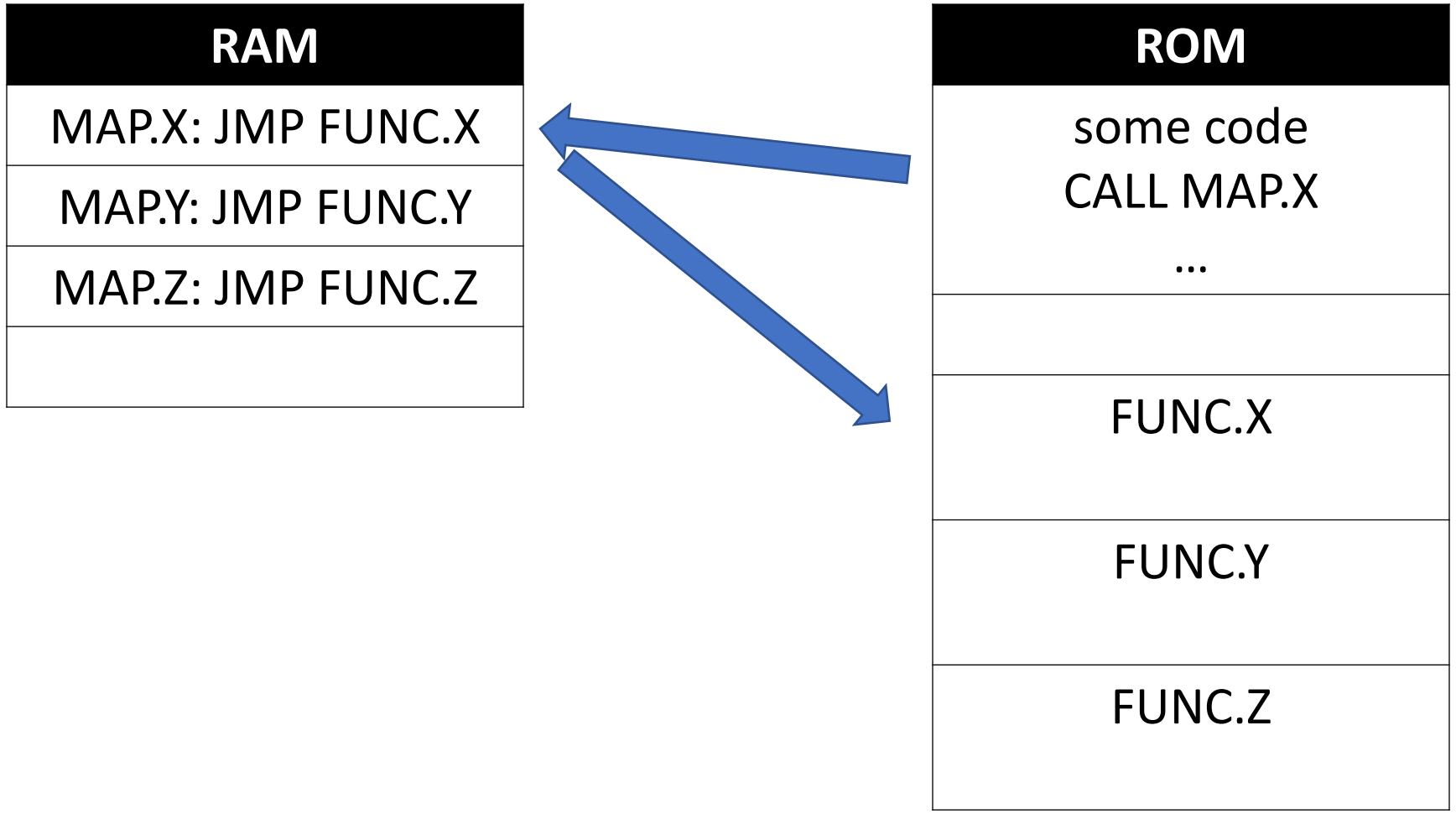
Jump table

```
0x00000108      02ba27          ljmp 0xba27
0x0000010b      02c54b          ljmp 0xc54b
0x0000010e      02cd41          ljmp 0xcd41
0x00000111      02b09b          ljmp 0xb09b
0x00000114      02d414          ljmp 0xd414
0x00000117      02a313          ljmp 0xa313
0x0000011a      02b1e2          ljmp 0xb1e2
0x0000011d      02c022          ljmp 0xc022
0x00000120      02c182          ljmp 0xc182
0x00000123      020367          ljmp 0x0367
0x00000126      029001          ljmp 0x9001
0x00000129      020358          ljmp 0x0358
0x0000012c      02cdb3          ljmp 0xcdb3
0x0000012f      02aa63          ljmp 0xaa63
0x00000132      028fe0          ljmp 0x8fe0
0x00000135      02ce17          ljmp 0xce17
0x00000138      02cd86          ljmp 0xcd86
```

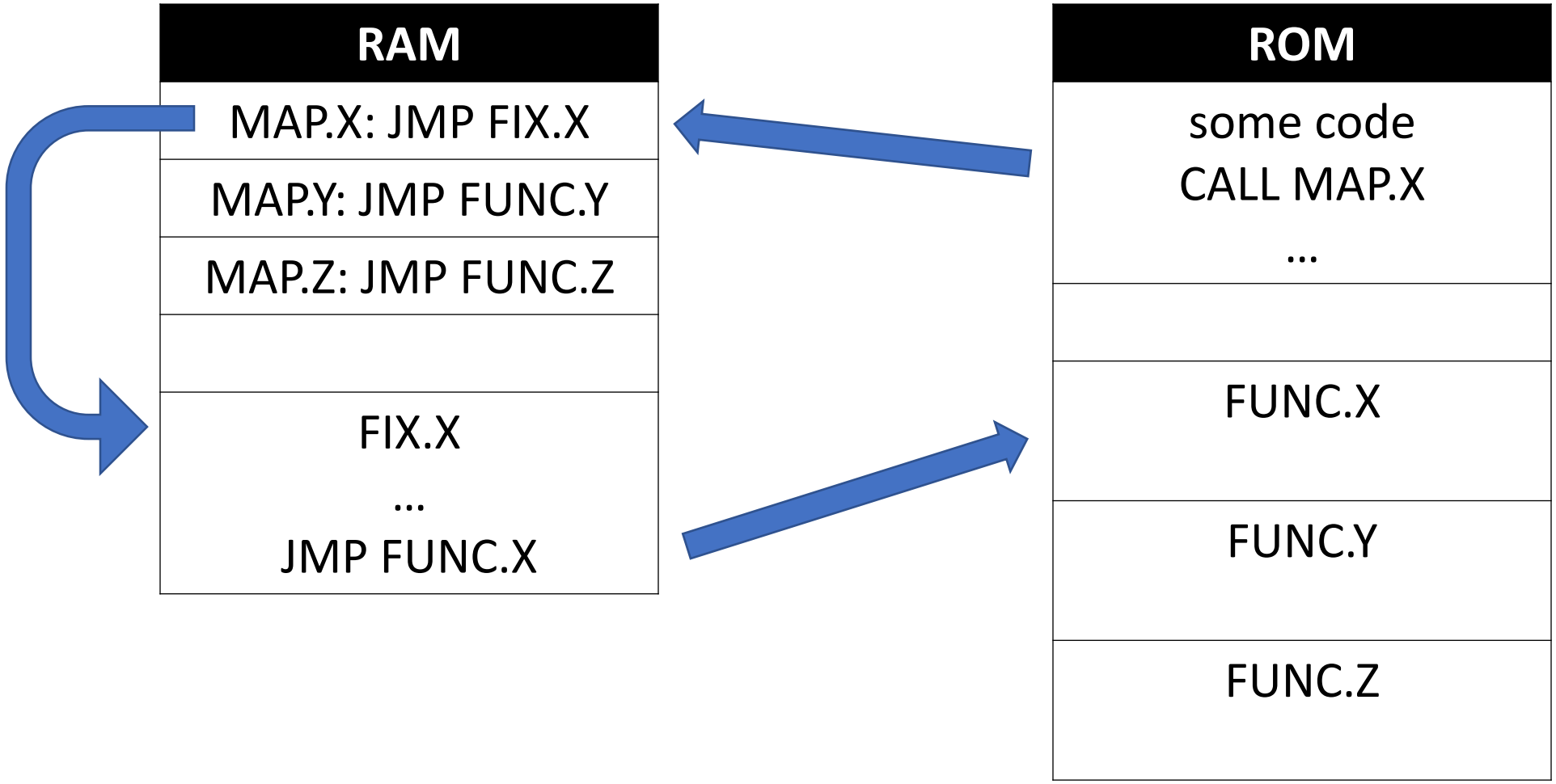
Concept: Jump table



Concept: Jump table



Concept: Jump table



Entry point: API commands

- EZRadioPRO API:

Argument Stream:

FIFO_INFO Argument Stream									
	Index	Name	7	6	5	4	3	2	1	0
	0x00	CMD	*0x15*							
	0x01	FIFO	0	0	0	0	0	0	RX	TX

Reply Stream:

FIFO_INFO Reply Stream										
	Index	Name	7	6	5	4	3	2	1	0
	0x00	CTS	CTS							
	0x01	RX_FIFO_COUNT	RX_FIFO_COUNT							
	0x02	TX_FIFO_SPACE	TX_FIFO_SPACE							

Entry point: API commands

```
0x00001e3c      b4f002          cjne a, #0xf0, 0x1e41
0x00001e3f      a12a            ajmp 0x1d2a
0x00001e41      b4f102          cjne a, #0xf1, 0x1e46
0x00001e44      8064            sjmp 0x1eaa
0x00001e46      b4f202          cjne a, #0xf2, 0x1e4b
0x00001e49      801e            sjmp 0x1e69
0x00001e4b      b41303          cjne a, #0x13, 0x1e51
0x00001e4e      020258          ljmp 0x0258
0x00001e51      b43502          cjne a, #0x35, 0x1e56
0x00001e54      8021            sjmp 0x1e77
0x00001e56      b43803          cjne a, #0x38, 0x1e5c
0x00001e59      02014d          ljmp 0x014d
0x00001e5c      b41a03          cjne a, #0x1a, 0x1e62
0x00001e5f      02019e          ljmp 0x019e
0x00001e62      7002            jnz 0x1e66
0x00001e64      e15b            ajmp 0x1f5b
0x00001e66      020132          ljmp 0x0132
```

0xf0 = PEEK

0xf1 = POKE

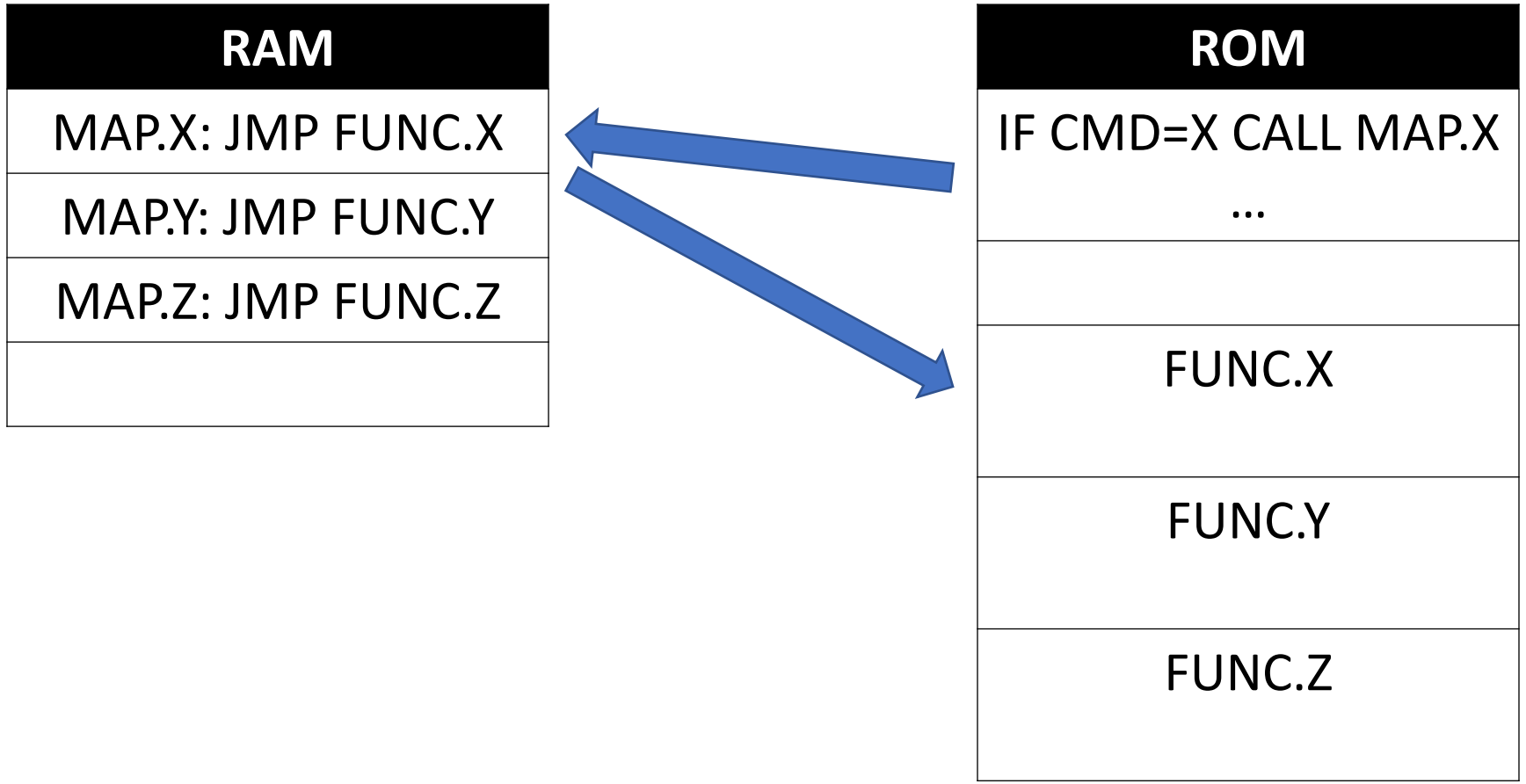
Some cmds go via jump table

Reverse engineering API commands

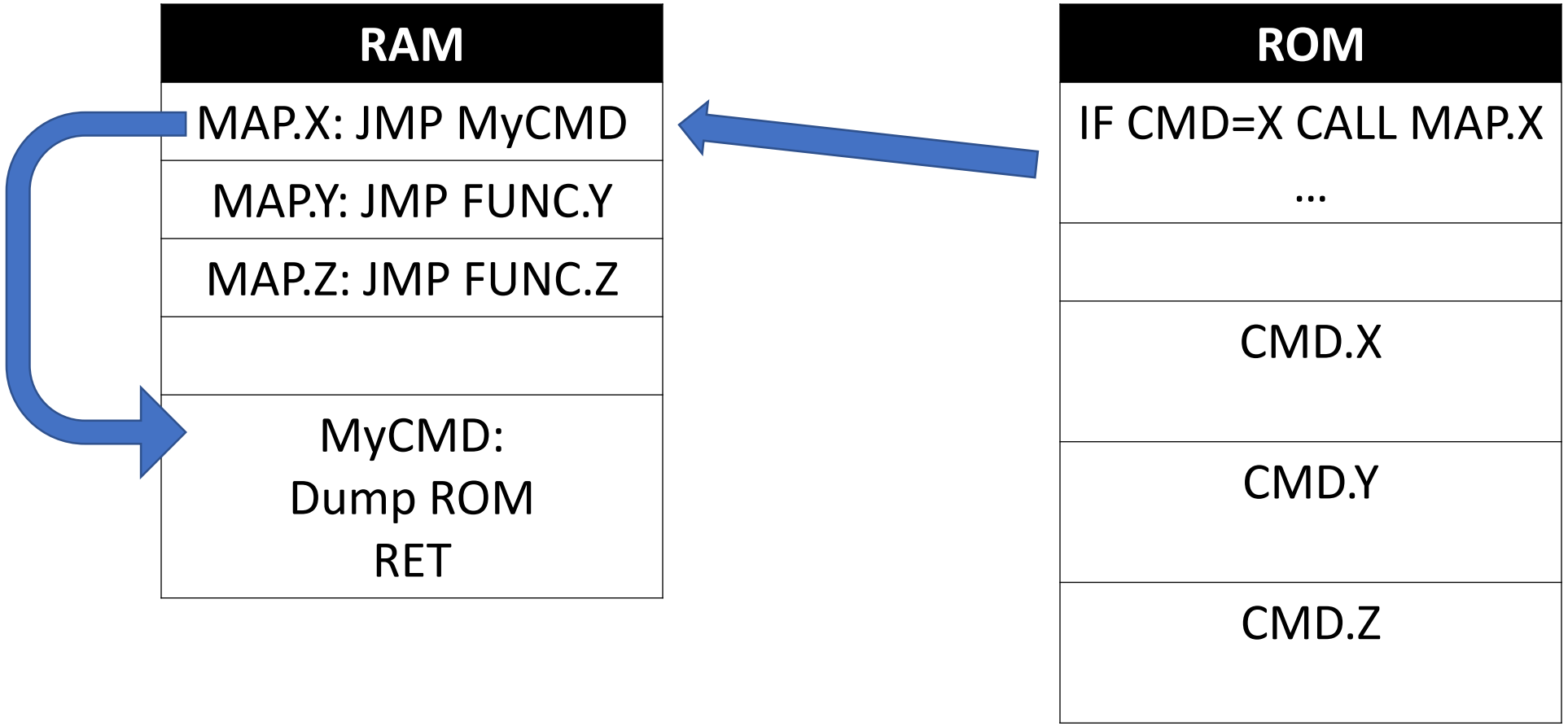
- Argument stream
  - XREG:0x70 Command
  - XREG:0x71-0x7f Arguments
- Reply stream
  - XREG:0x70-0x7f Reply bytes 0x01-0x10
  - CTS byte generated somewhere else
- RET to finish command

- Not all PEEK/POKE addresses translate to RAM/ROM access

Hijack API command to dump firmware



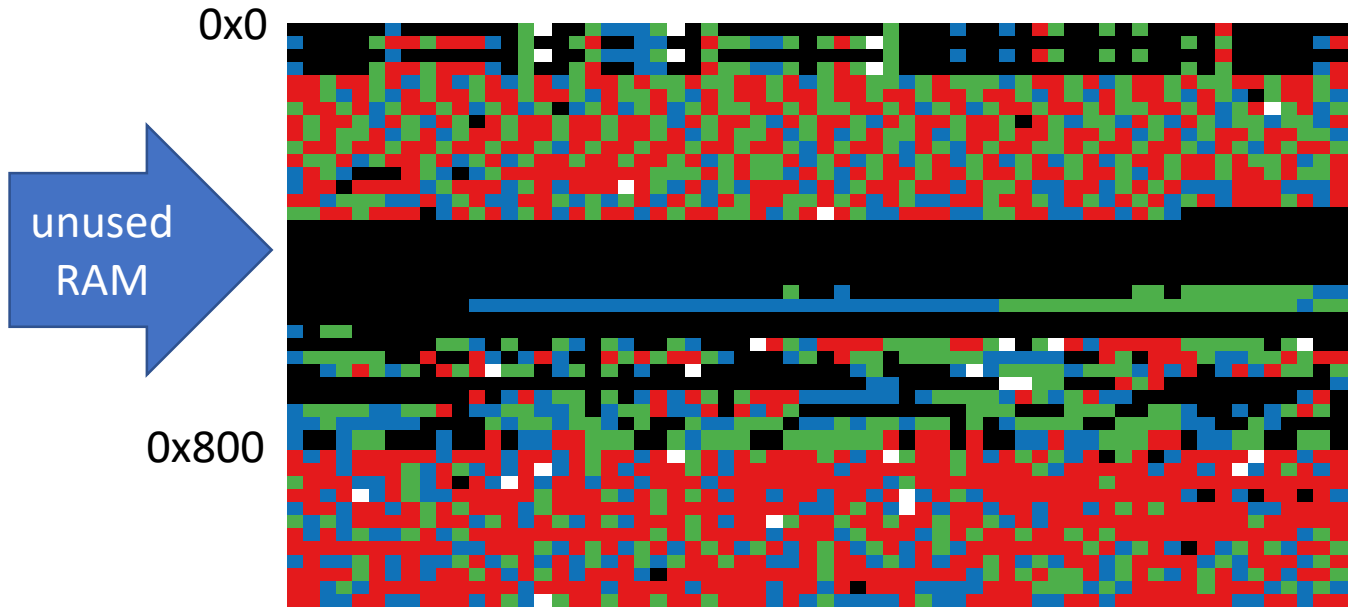
Hijack API command to dump firmware



Hijack API command to dump firmware

- Command to hijack
  - CJNE present in dump
  - Jump table entry above 0x0100
  - ➢ 0x12 **GET_PROPERTY**
- Unused RAM
  - B1B: 0x0484 – 0x0560
  - C2A: 0x03b6 – 0x051e
  - A2A: 0x0466 – 0x051e
  - ➢Write patch at 0x0488
- 96 bytes max

```
cjne a, #0x12, 0x1e24
ljmp 0x0168
```



Dumping complete firmware

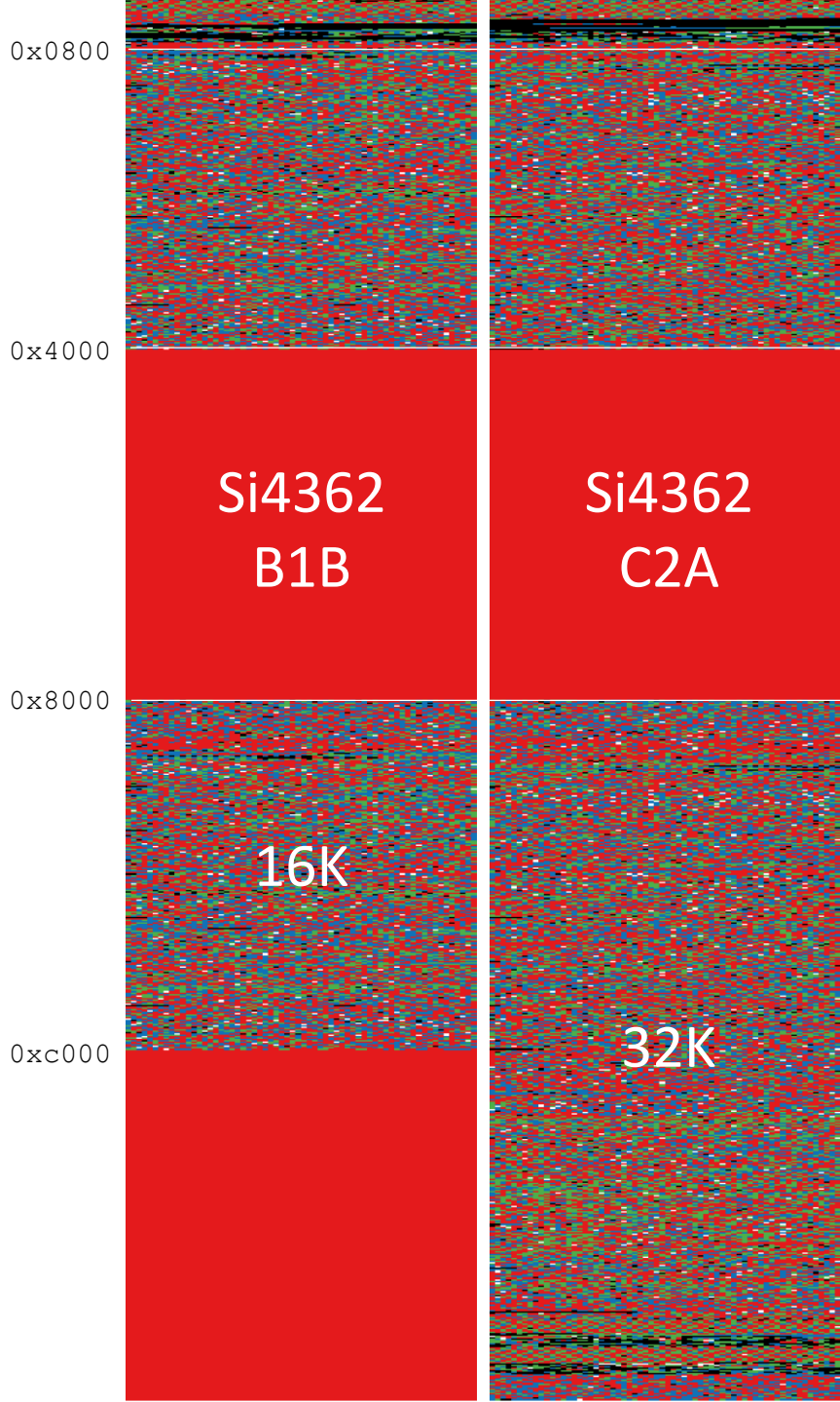
```
hack.memory_dump ();
        ; CODE XREF from map.cmd_get_property (0x168)
        0x00000488              7871            mov r0, #0x71
        0x0000048a              e2              movx a, @r0
        0x0000048b              f583            mov dph, a
        0x0000048d              08              inc r0
        0x0000048e              e2              movx a, @r0
        0x0000048f              f582            mov dpl, a
        0x00000491              7870            mov r0, #0x70
        ; CODE XREF from hack.memory_dump (0x498)
    ┌─> 0x00000493              e4              clr a
    ¦   0x00000494              93              movc a, @a+dptr
    ¦   0x00000495              a3              inc dptr
    ¦   0x00000496              f2              movx @r0, a
    ¦   0x00000497              08              inc r0
    └─< 0x00000498 [1]          b880f8          cjne r0, #0x80, 0x0493
        0x0000049b              22              ret
```

Patch procedure – in pseudo python

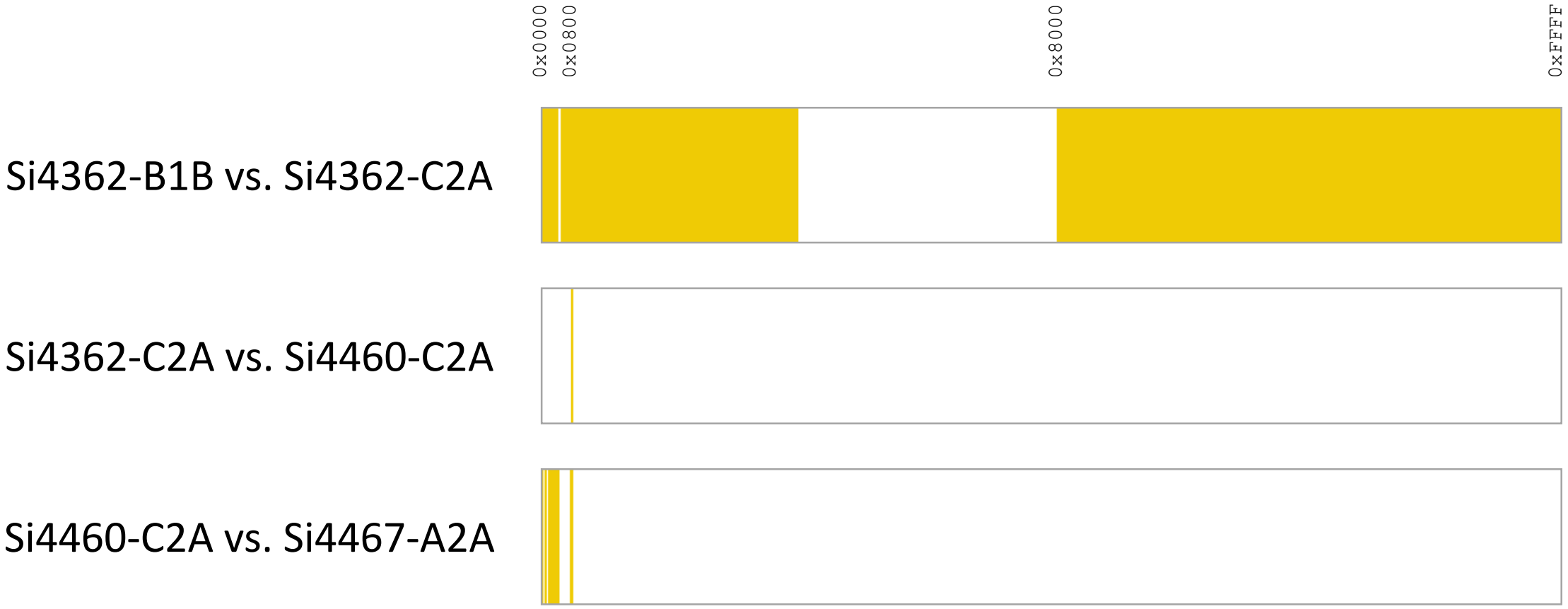
```
patch_code = [0x78,0x71,0xe2,..]
patch_loc = 0x488
patch_jmp = 0x168
patch_cmd = 0x12

resetRadio()
sendCommand(POWER_UP)

addr = patch_loc
for data in patch_code:
     sendCommand(POKE,[addr,data])
     addr = addr+1

sendCommand(POKE,[patch_jmp,patch_loc])

print sendCommand(patch_cmd,[params])
```

Success!

- 0x0000 code – RAM 2K
  - 0x0000 vector table
  - 0x0057 jump table
  - 0x02eb code
  - 0x051e data
- 0x0800 code – mirror of ROM @0x8800
- 0x4000 empty
- 0x8000 code – ROM 16-32K



Comparing chip models and revisions



Strategies for reversing firmware

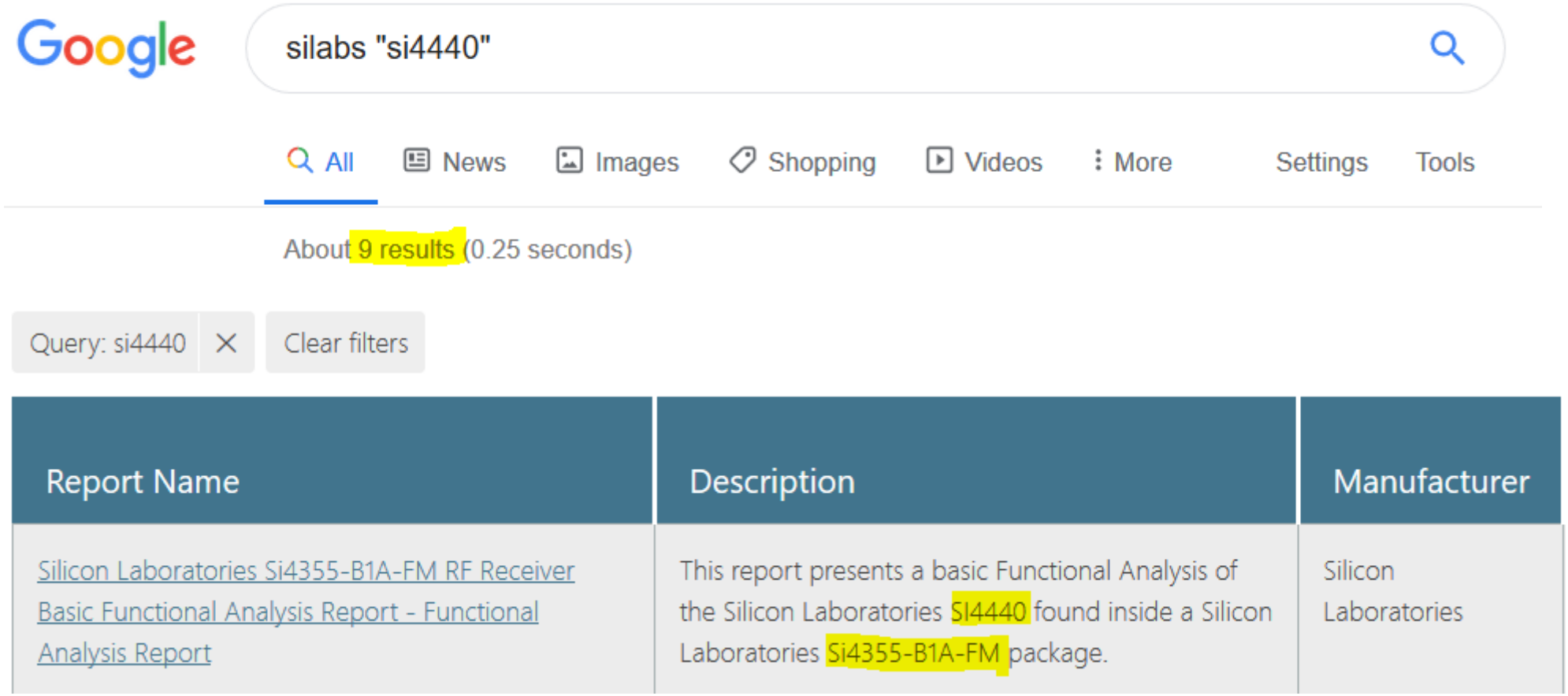
- Strings

```
401792e4 beq     LAB_401792f0
401792e6 lsl     r4,r4,#0x10
401792e8 asr     r4,r4,#0x10
401792ea mov     r0,r4
401792ec blx     SCPI_report_error
401792f0 ldr     r0,["SCPIPrivateSuperMode Test!\n"]
401792f2 blx     SCPI_printf
401792f6 add     sp,#0x14
401792f8 pop     { r4, r5, pc }
```

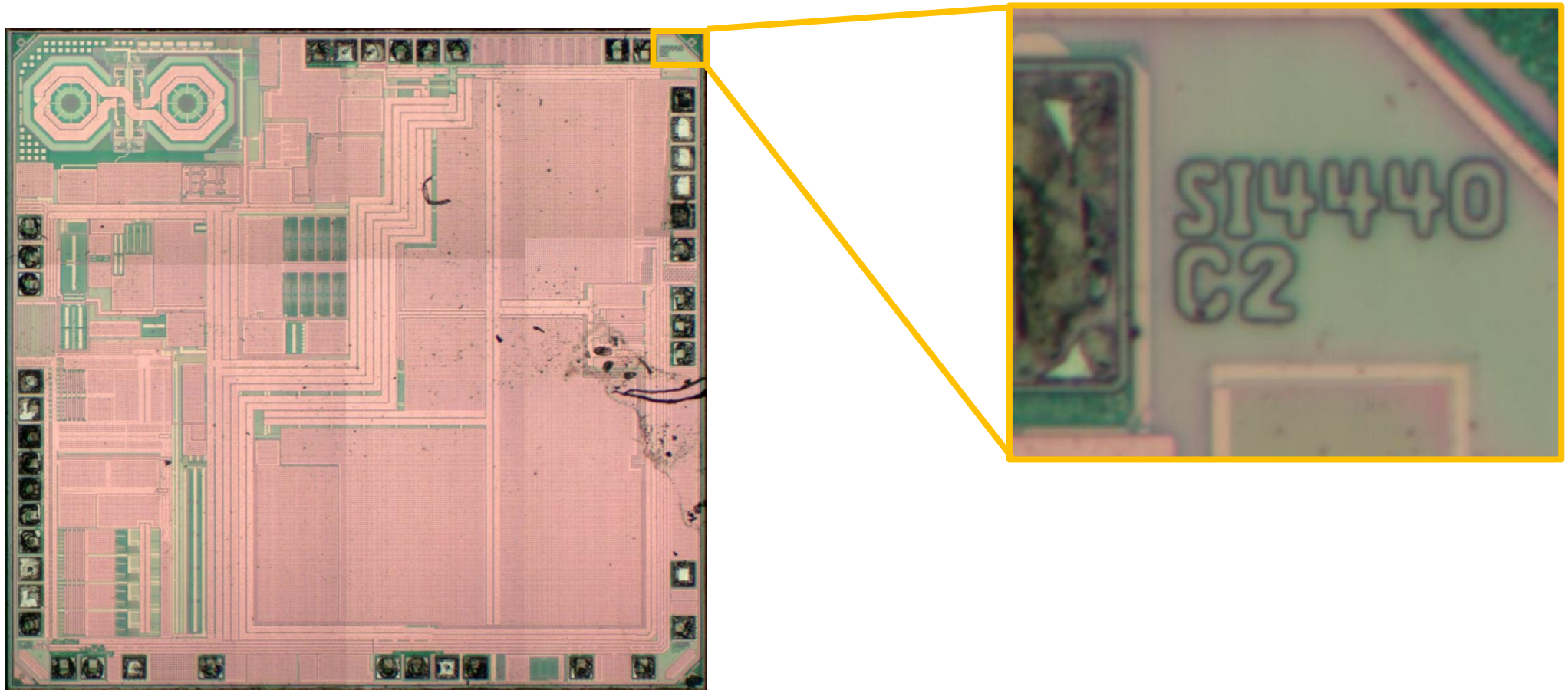
(real example from unnamed test equipment)

Only few strings

Si4440 – LMGTFY



Si4440 – Sending IC and $60 to China



Si4460-C2A

Strategies for reversing firmware

- ~~Strings!~~
- Libraries (signatures)

Strategies for reversing firmware

- ~~Strings!~~
- ~~Libraries (signatures)~~
- Reverse engineer documented API
  - Commands
  - Properties

0x33 REQUEST_DEVICE_STATE

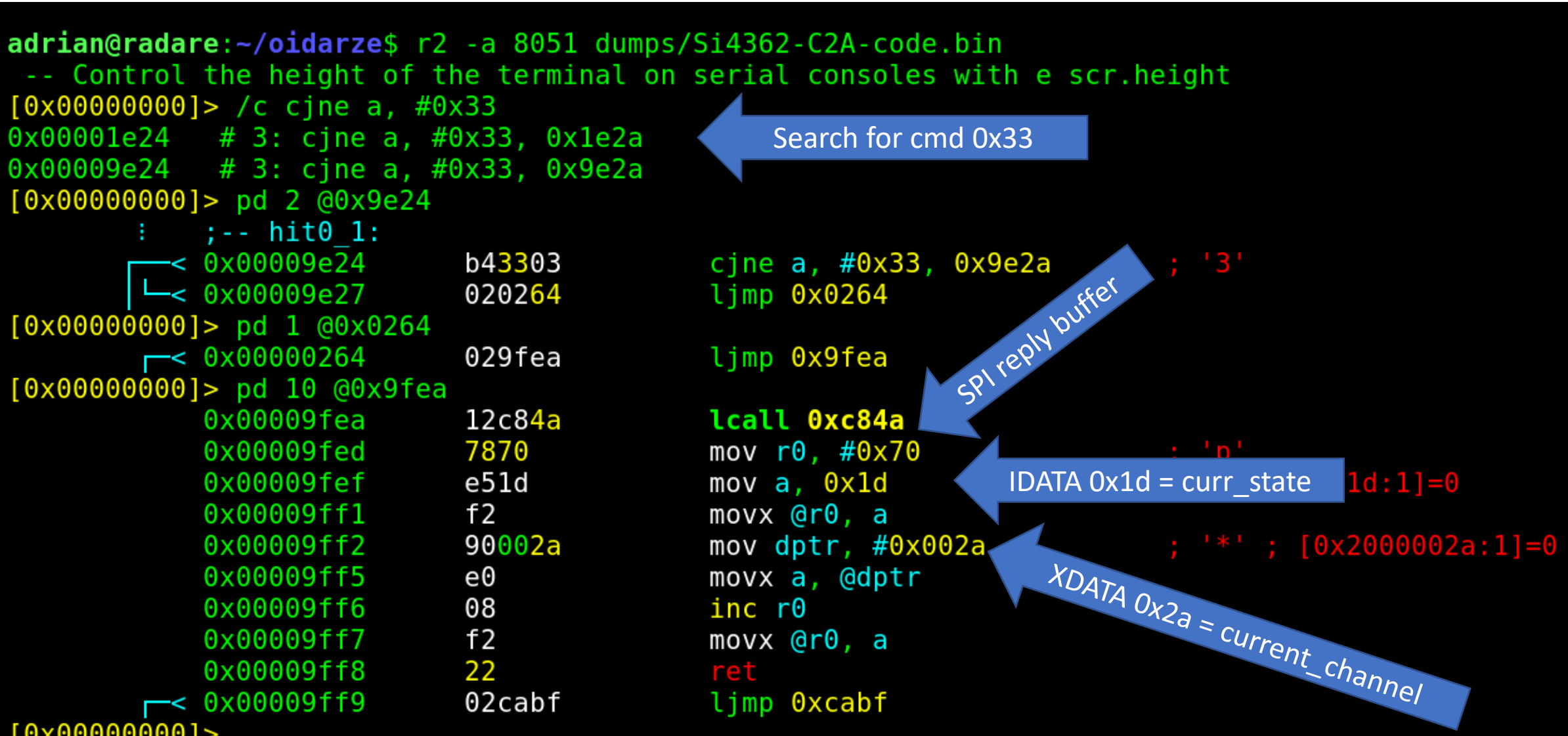
Argument Stream:

REQUEST_DEVICE_STATE Argument Stream										
	Index	**Name**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	0x00	**CMD**	*0x33*							

Reply Stream:

REQUEST_DEVICE_STATE Reply Stream										
	Index	**Name**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	0x00	**CTS**	CTS							
	0x01	**CURR_STATE**	X	X	X	X	MAIN_STATE			
	0x02	**CURRENT_CHANNEL**	CURRENT_CHANNEL							

0x33 REQUEST_DEVICE_STATE

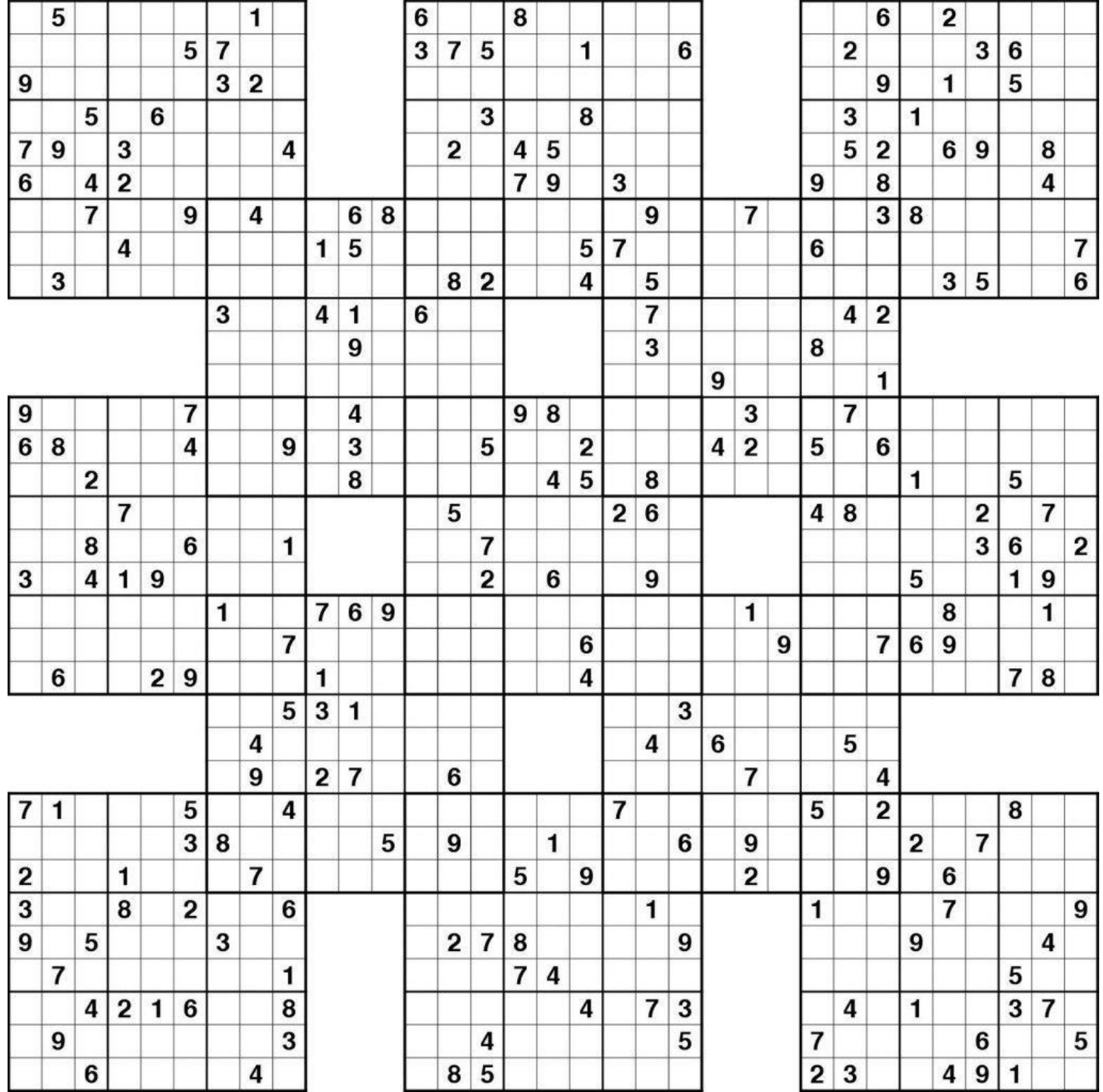


0x33 REQUEST_DEVICE_STATE

- Assign function and variable names
  - CODE:0x9fea rom.cmd_request_device_state()
  - IMEM:0x1d var.current_state
  - XMEM:0x2a var.current_channel
  - Inspect other code that references these variables (XREFs)

```
;-- var.CURRENT_CHANNEL:
; XREFS: DATA 0x00000387  DATA 0x00008a68  DATA 0x00009ff2  DATA 0x0000a6eb  DATA 0x0000bd56
; XREFS: DATA 0x0000d091  DATA 0x0000d10a  DATA 0x0000d3fd  DATA 0x0000d9a7  DATA 0x0000e8df
; XREFS: DATA 0x0000e987  DATA 0x0000ec1a  DATA 0x0000ec76
0x0000002a      .byte 0x00
```

- Rinse & repeat



```
[0x0000cfae [xAdvc] 0% 150 /home/adrian/oidarze/dumps/Si4362-C2A-code.bin]> pd $r @ rom.spi_parse_cmds
 (fcn) rom.spi_parse_cmds 66
   rom.spi_parse_cmds ();
      ; CODE XREF from map.spi_parse_cmds (0x2d0)
      ; CODE XREF from func3.fragment (+0x267)
      0x0000cfae      7870            mov r0, #0x70               ; 0x5070 ; xreg.CMD_BUFF0
      0x0000cfb0      e2              movx a, @r0
   ,=< 0x0000cfb1      b43603          cjne a, #0x36, 0xcfb7       ; RX_HOP
   |,< 0x0000cfb4      0202c7          ljmp map.cmd_rx_hop
   ||      ; CODE XREF from rom.spi_parse_cmds (0xcfb1)
  ,==< 0x0000cfb7      b43703          cjne a, #0x37, 0xcfbd       ; TX_HOP
  |,,< 0x0000cfba      0202b2          ljmp map.cmd_tx_hop
  ||||    ; CODE XREF from rom.spi_parse_cmds (0xcfb7)
  ,===< 0x0000cfbd      b43403          cjne a, #0x34, 0xcfc3       ; CHANGE_STATE
  |||,< 0x0000cfc0      0202df          ljmp map.cmd_change_state
  ||||    ; CODE XREF from rom.spi_parse_cmds (0xcfbd)
  ,===> 0x0000cfc3      b43203          cjne a, #0x32, 0xcfc9       ; START_RX
  |,,,< 0x0000cfc6      0202be          ljmp map.cmd_start_rx
  ||||    ; CODE XREF from rom.spi_parse_cmds (0xcfc3)
  ,===< 0x0000cfc9      b43103          cjne a, #0x31, 0xcfcf       ; START_TX
  |,,,< 0x0000cfcc      0202c1          ljmp map.cmd_start_tx
  ||||    ; CODE XREF from rom.spi_parse_cmds (0xcfc9)
  ,===> 0x0000cfcf      b42003          cjne a, #0x20, 0xcfd5       ; GET_INT_STATUS
  |,,,< 0x0000cfd2      0202b5          ljmp map.cmd_get_int_status
  ||||    ; CODE XREF from rom.spi_parse_cmds (0xcfcf)
  ,===< 0x0000cfd5      b42103          cjne a, #0x21, 0xcfdb       ; GET_PH_STATUS
  |||,< 0x0000cfd8      0202e5          ljmp map.cmd_get_ph_status
  ||||    ; CODE XREF from rom.spi_parse_cmds (0xcfd5)
  ,===> 0x0000cfdb      b42203          cjne a, #0x22, 0xcfe1       ; GET_MODEM_STATUS
  |,,,< 0x0000cfde      0202b8          ljmp map.cmd_get_modem_status
```

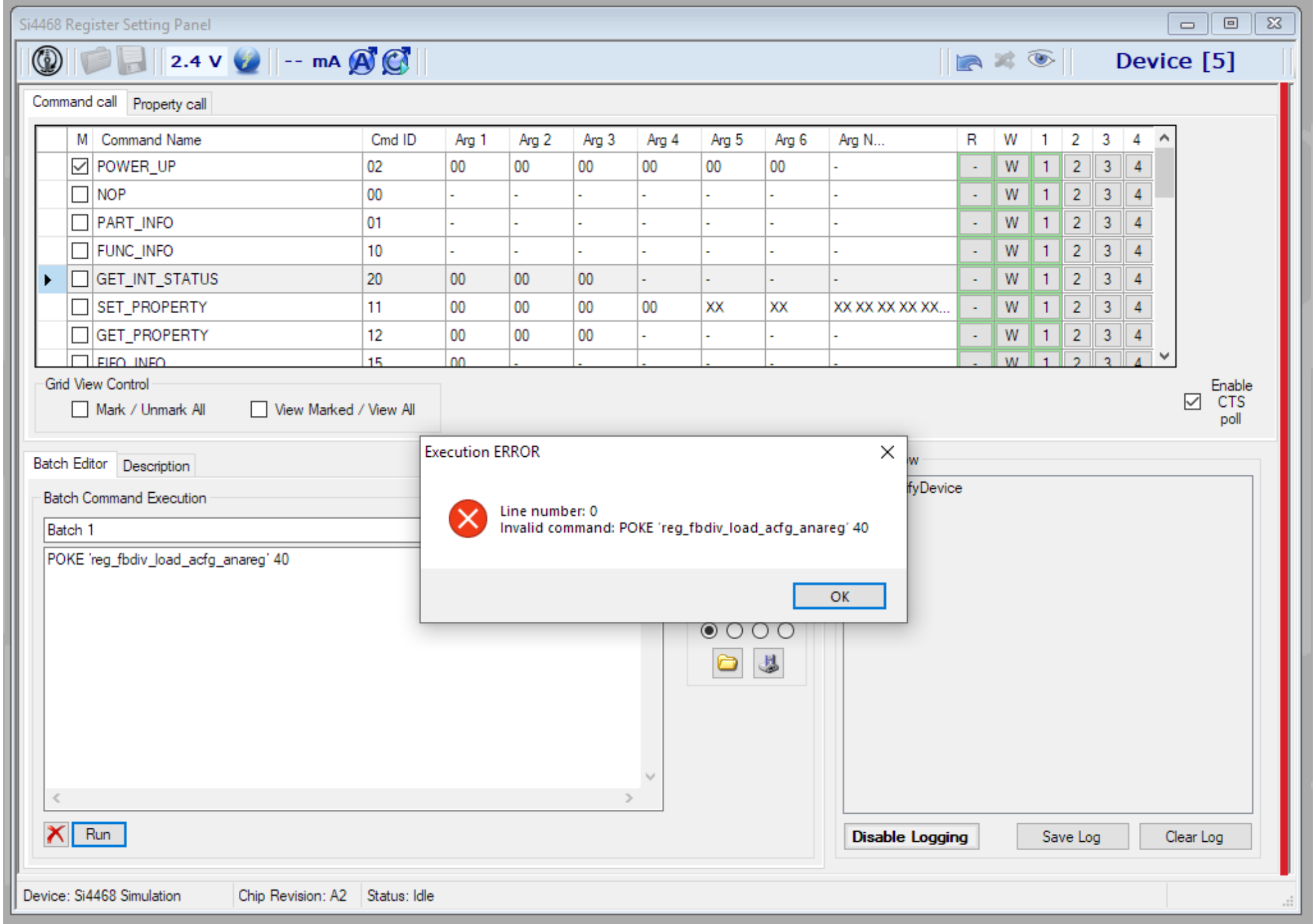
Strategies for reversing embedded firmware

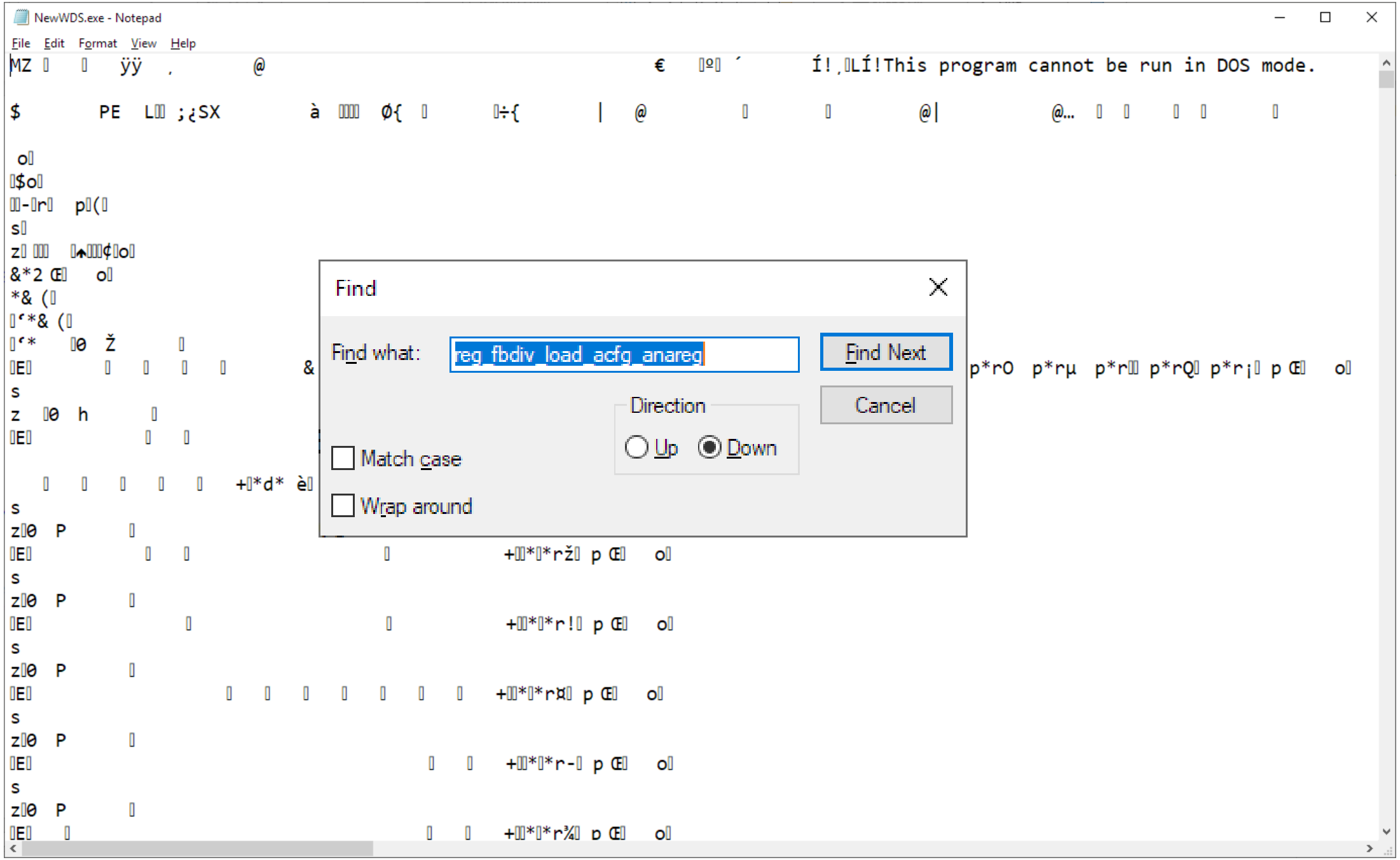
- ~~Strings!~~
- ~~Libraries (signatures)~~
- Reverse documented API
  - Commands
  - Properties
- Cheat by reversing the vendor tool

Magic strings

“A further 2–3 dB improvement can be achieved by applying the following register write command. This value will increase the supply on the PLL feedback divider block for better phase noise performance. ‘POKE’reg_fbdiv_load_acfg_anareg’ 40”

https://www.silabs.com/documents/public/application-notes/AN736.pdf


Si4468 Register Setting Panel
2.4 V
-- mA
Device [5]
Command call
Property call
M Command Name Cmd ID Arg 1 Arg 2 Arg 3 Arg 4 Arg 5 Arg 6 Arg N... R W 1 2 3 4
POWER_UP 02 00 00 00 00 00 00 -
NOP 00 - - - - - - -
PART_INFO 01 - - - - - - -
FUNC_INFO 10 - - - - - - -
GET_INT_STATUS 20 00 00 00 - - - -
SET_PROPERTY 11 00 00 00 00 XX XX XX XX XX XX...
GET_PROPERTY 12 00 00 00 - - - -
Grid View Control
Mark / Unmark All
View Marked / View All
Enable CTS poll
Batch Editor
Description
Batch Command Execution
Batch 1
POKE 'reg_fbdiv_load_acfg_anareg' 40
Run
Execution ERROR
Line number: 0
Invalid command: POKE 'reg_fbdiv_load_acfg_anareg' 40
OK
fyDevice
Disable Logging
Save Log
Clear Log
Device: Si4468 Simulation
Chip Revision: A2
Status: Idle


NewWDS.exe - Notepad
File Edit Format View Help
MZ ÿÿ @ € Í!,LÍ!This program cannot be run in DOS mode.
Find
Find what: reg_fbdiv_load_acfg_anareg
Find Next
Cancel
Direction
Up
Down
Match case
Wrap around

NewWDS.exe - Notepad

File Edit Format View Help

```
      <Bit index="5" name="reserved" />
      <Bit index="4" name="reserved" />
      <Bit index="3" name="reserved" />
      <Bit index="2" name="tempco_f[2]" />
      <Bit index="1" name="tempco_f[1]" />
      <Bit index="0" name="tempco_f[0]" />
      <Description>\\silabs.com\design\projects\si4440\revB0\modules\pro2_is_refs\docs\specs\vtr_hv_stop.docx</Description>
    </Register>
    <Register address="06" name="reg_fbdiv_load_acfg_anareg" size="8" defaultVal="00" isRead="1" isWrite="1" mode="0">
      <Classification isTestReg="0" isRFReg="0" isDigitalReg="0" isMiscReg="1" isStatusReg="0" isAconfig="1" />
      <Bit index="7" name="trim[3]" />
      <Bit index="6" name="trim[2]" />
      <Bit index="5" name="trim[1]" />
      <Bit index="4" name="trim[0]" />
      <Bit index="3" name="en_min_load" />
      <Bit index="2" name="fine[2]" />
      <Bit index="1" name="fine[1]" />
      <Bit index="0" name="fine[0]" />
      <Description>\\silabs.com\design\projects\si4440\revB0\modules\pro2_is_regs\docs\specs\reg_ana_stop.docx</Description
    </Register>
    <Register address="07" name="reg_cplf_load_acfg_anareg" size="8" defaultVal="00" isRead="1" isWrite="1" mode="0">
      <Classification isTestReg="0" isRFReg="0" isDigitalReg="0" isMiscReg="1" isStatusReg="0" isAconfig="1" />
      <Bit index="7" name="trim[3]" />
      <Bit index="6" name="trim[2]" />
      <Bit index="5" name="trim[1]" />
      <Bit index="4" name="trim[0]" />
      <Bit index="3" name="en_min_load" />
      <Bit index="2" name="fine[2]" />
      <Bit index="1" name="fine[1]" />
      <Bit index="0" name="fine[0]" />
      <Description>\\silabs.com\design\projects\si4440\revB0\modules\pro2_is_regs\docs\specs\reg_ana_stop.docx</Description
    </Register>
    <Register address="08" name="reg_clkgen_load_acfg_anareg" size="8" defaultVal="00" isRead="1" isWrite="1" mode="0">
      <Classification isTestReg="0" isRFReg="0" isDigitalReg="0" isMiscReg="1" isStatusReg="0" isAconfig="1" />
```

NewWDS.exe - Notepad

File Edit Format View Help

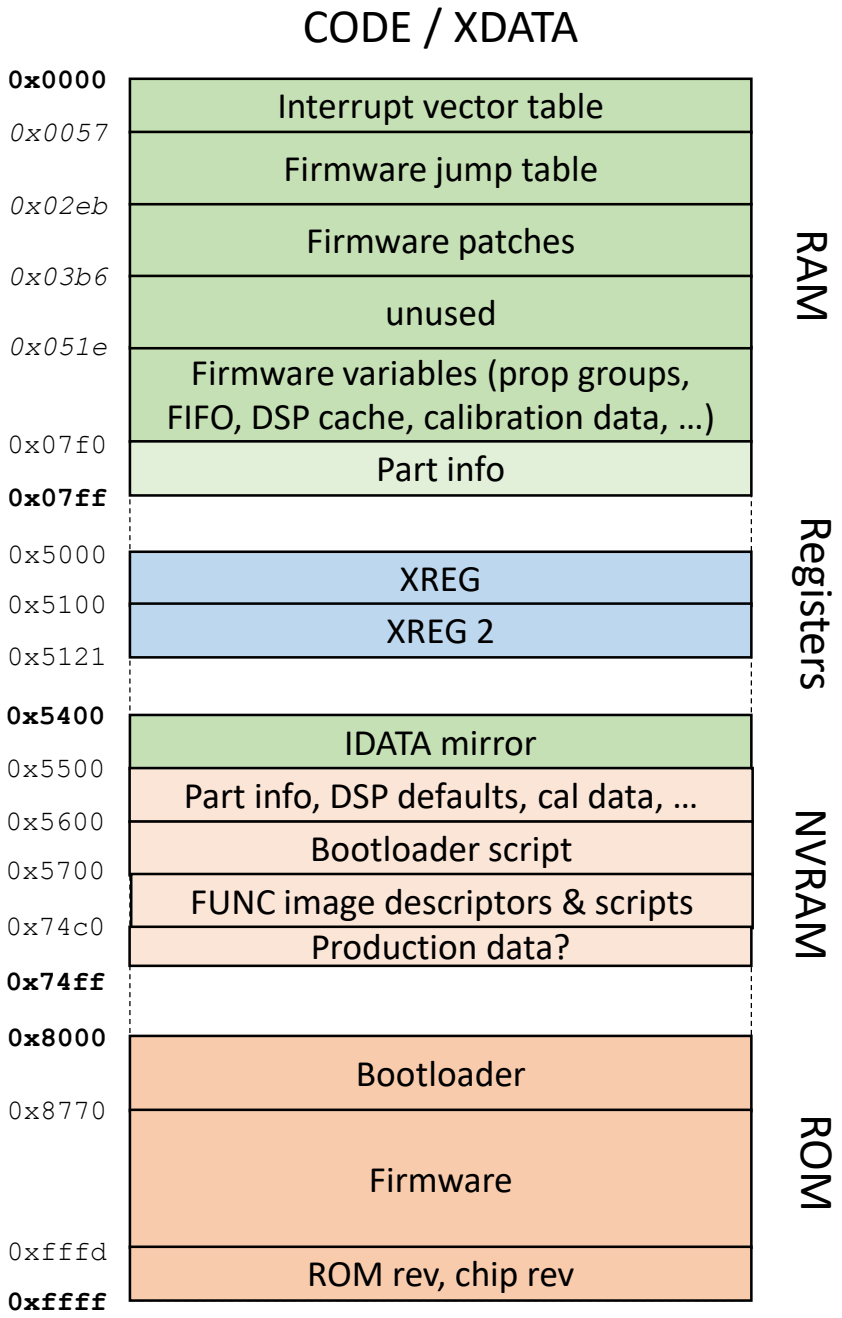
```
            Clear to Send. Indicates that the previous command has completed
            execution and DATA\[1..16\] is valid. The next command may be sent.
          </DESCRIPTION>
        </REPLY>
      </OUTPUT_DETAILS>
    </COMMAND>
    <COMMAND name="PATCH_DATA" value="0xE0" feature="boot" dev="1" wip="0" internal="1">
      <SUMMARY>*ENCRYPTED* Reserved command used for patch file downloads.</SUMMARY>
      <DESCRIPTION>Load patch data. Technically, there are 16 PATCH_DATA commands 0x10-0x1F. CMD[2:0]  is the number of val
      <TARGETRELEASE></TARGETRELEASE>
      <INPUT_DETAILS internal="1">
        <PARAMETER name="DATA0" address="1" type="u8">
          <DESCRIPTION>encrypted patch data byte to load into RAM</DESCRIPTION>
        </PARAMETER>
        <PARAMETER name="DATA1" address="2" type="u8">
          <DESCRIPTION>...</DESCRIPTION>
        </PARAMETER>
        <PARAMETER name="DATA2" address="3" type="u8">
          <DESCRIPTION>...</DESCRIPTION>
        </PARAMETER>
        <PARAMETER name="DATA3" address="4" type="u8">
          <DESCRIPTION>...</DESCRIPTION>
        </PARAMETER>
        <PARAMETER name="DATA4" address="5" type="u8">
          <DESCRIPTION>...</DESCRIPTION>
        </PARAMETER>
        <PARAMETER name="DATA5" address="6" type="u8">
          <DESCRIPTION>...</DESCRIPTION>
        </PARAMETER>
        <PARAMETER name="DATA6" address="7" type="u8">
          <DESCRIPTION>...</DESCRIPTION>
        </PARAMETER>
        <PARAMETER name="DATA7" address="8" type="u8">
          <DESCRIPTION>...</DESCRIPTION>
```

What I learned about the EZRadioPRO

(so far)

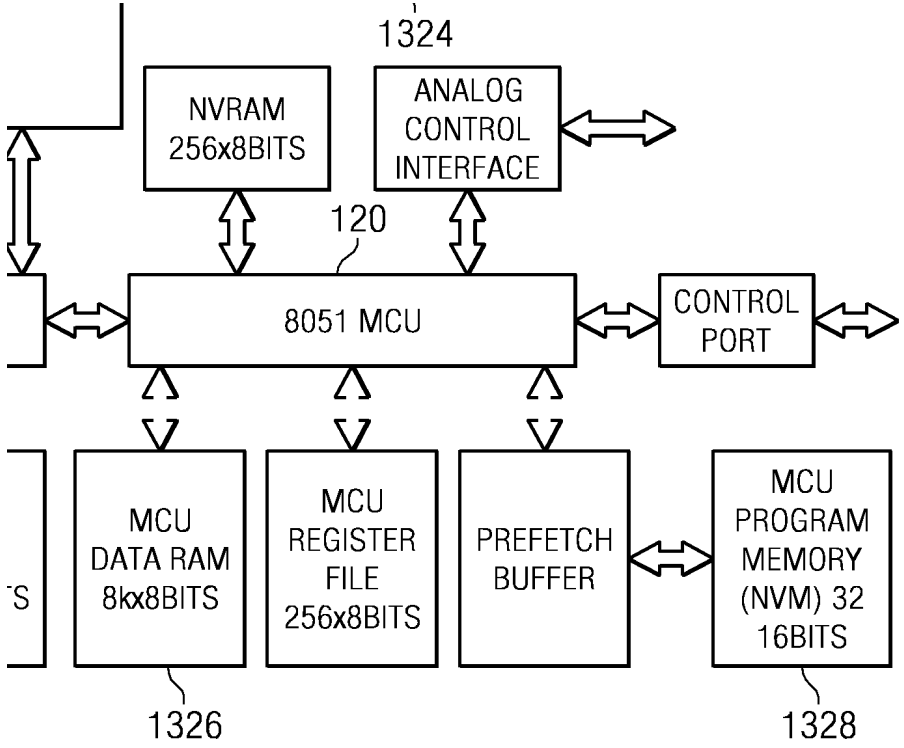
Memory map (C2A, FUNC 1)

- Shared CODE and XDATA address space
- 2K XDATA + 256 bytes IDATA RAM
- 32K ROM
  - Bootloader
  - 3 Firmware images/modes
- 3 sets of registers
  - SFR
  - XREG/XREG2 (XDATA 0x5000-0x51ff)
  - “DSP” via SFR 0x94/0x95
- 4-8K NVRAM
  - Access via DMA



What is done by the 8051?

- Bootloader
- Radio configuration
- Radio state machine
- Interrupts and flags
- Packet handler
- Non-std preamble
- Software CRC & whitening
- Image rejection calibration
- RSSI latch, antenna diversity (RX)
- Power amplifier control (TX)
- ADC for temperature & voltage



What is NOT done by the 8051?

- SPI communication (DMA)
- Some API commands
  - FRR_x_READ, READ_RX_FIFO
- RF ADC & DAC
- Modulation / Demodulation
- Standard preamble
- HW CRC & whitening
- GPIO

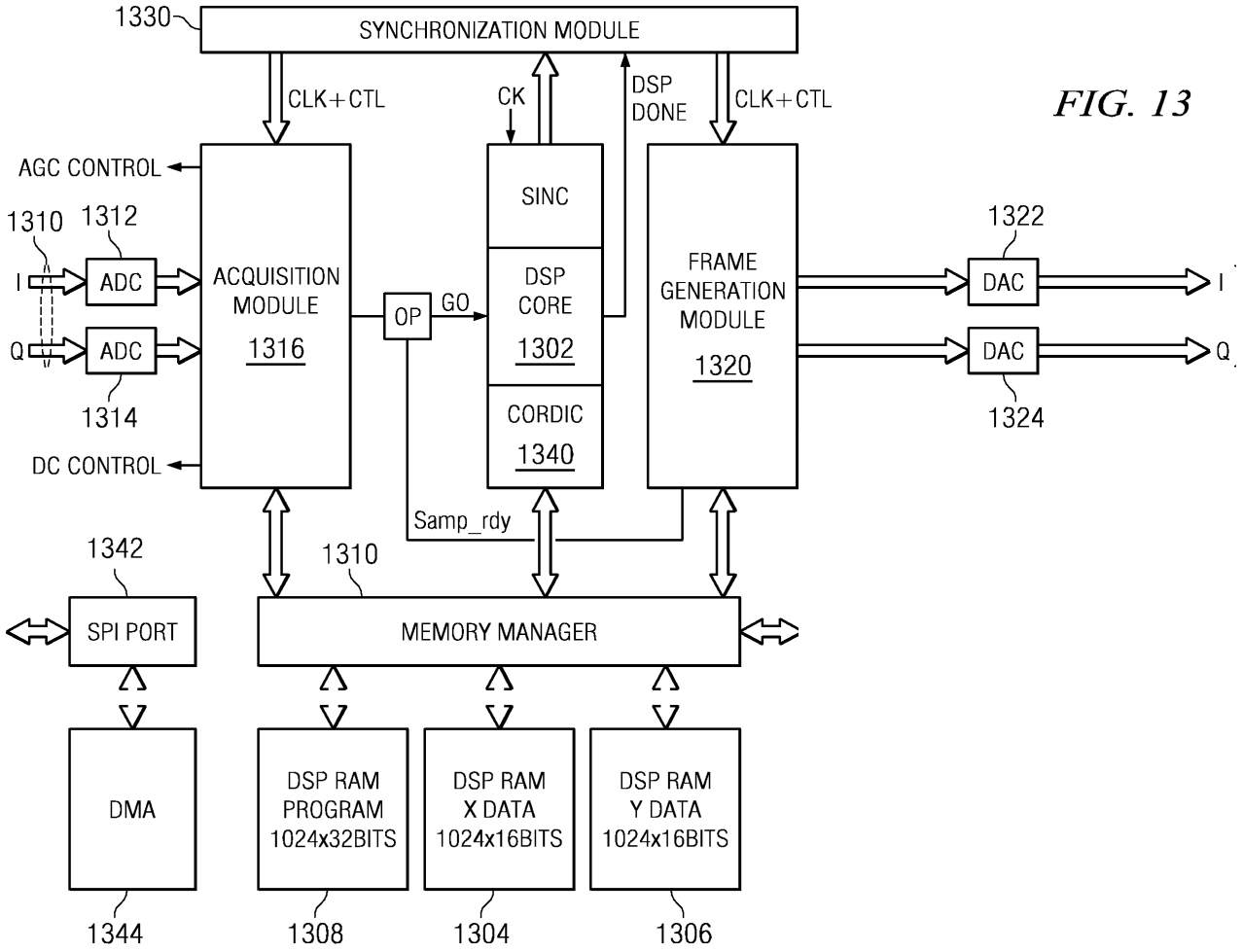


FIG. 13

Future research

- Decrypt and create firmware patches
- Access DSP RAM and ROM
- Modify NVRAM
- Explore other SiLabs radios (e.g. Si46xx/Si47xx FM transceivers)

Project Inside-EZRadioPRO

- https://github.com/astuder/Inside-EZRadioPRO
- What I share:
  - Python code to dump firmware
  - R2 scripts
  - Reversed documentation (like this deck)
- What I don’t share:
  - Firmware binaries (use Python script to dump your own)
  - SiLabs documentation (use Google)
- Pull requests are welcome

Thank you!

Questions?

@adistuder

https://github.com/astuder/