

電子辞書のアイデンティティを消す方法

末田 卓巳 @puhitaku

第54回 情報科学若手の会 #wakate2022

末田 卓巳 Takumi Sueda



@puhitaku



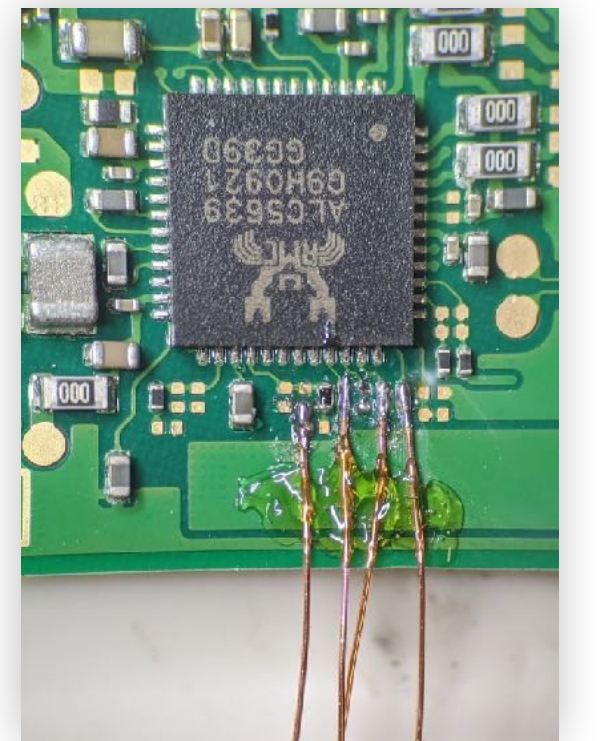
フリー開発者。米住宅ベンチャー HOMMA Inc. などで開発に従事。

好きなもの

- ・ 全レイヤーの技術、特に低いもの
- ・ リバースエンジニアリング
- ・ 3Dプリント
- ・ 音楽

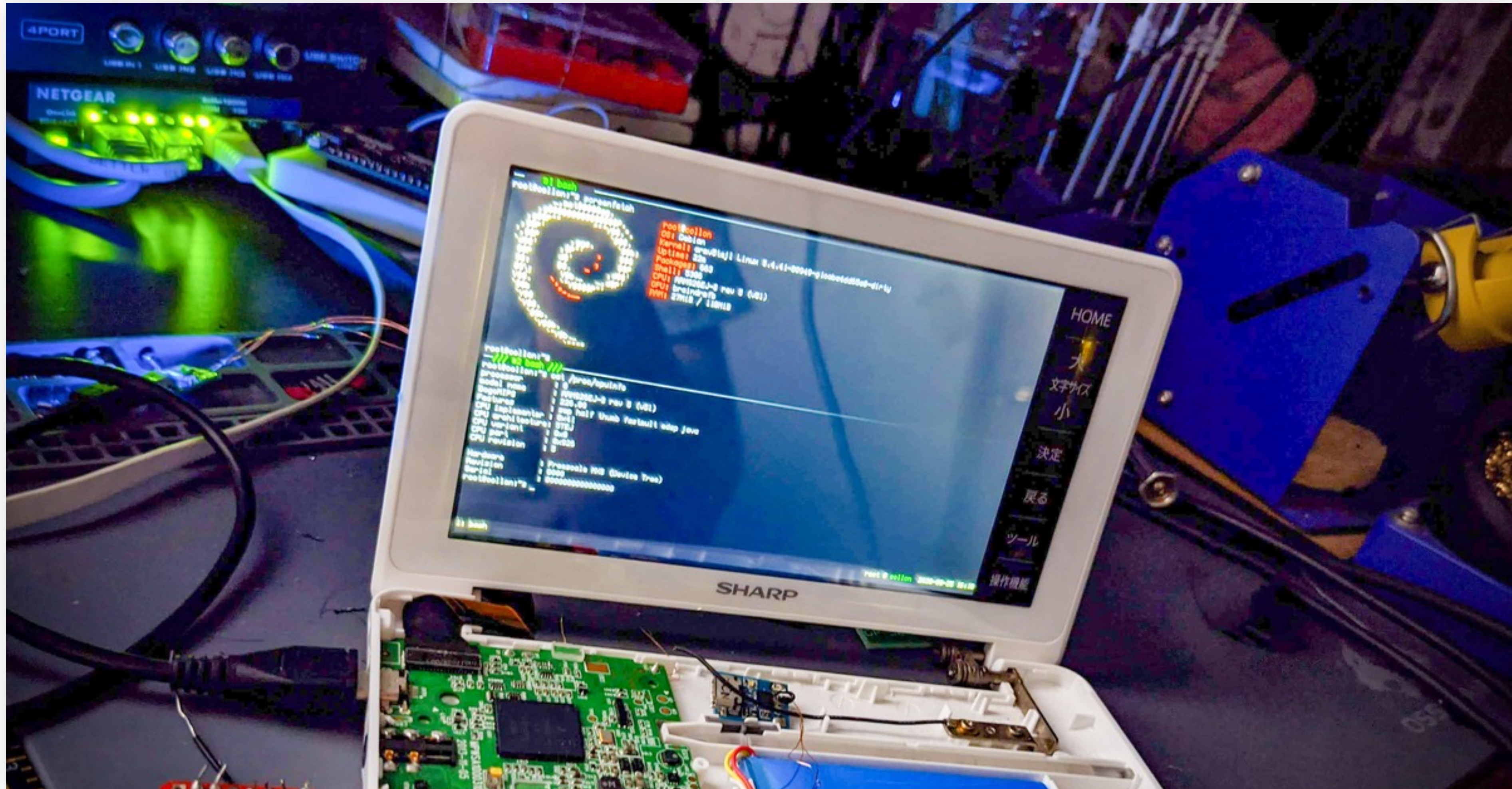
これまでの参加歴

- ・ 2014～2018
- ・ 2020「電子辞書は組み込み Linux の夢を見るか？」
- ・ 2021「TEPRA Lite ではじめる BLE リバースエンジニアリング」



はじめに

本日はご紹介する電子辞書…



SHARP Brain

電子辞書は組み込みLinuxの夢を見るか？

末田 卓巳 @puhitaku

第53回 情報科学若手の会 #wakate2020

去る2020年の若手の会で Linux のポーティングを発表

**前発表のスライドと記事が弊ブログにあるので
後でお読みいただくと
今日のスライドが更に楽しめると思います
(Twitter #wakate2022 で流します)**

SHARP Brain とは？

特徴

- SHARP が2008年に発売した電子辞書ブランド
- Windows CE を搭載（2020年の機種まで）
- CE 向けアプリの一部が動く
- **自分でコンパイルした exe も動く**
- 発売後すぐ Brain ハック界隈が 2ch で成立

Brain で動く Windows CE アプリ

- アプリランチャー
- オフラインで Wikipedia を読むソフト
- matplotlib
- ギャルゲー
などなど...



SHARP Brain PW-SH1

CPU: NXP i.MX283

ARM926EJ-S, armv5tej
454 MHz

DRAM: LPDDR 128MB

LCD: 800x480 など



SD: SDXC

スロットに挿抜可能

eMMC: 8GB

SDの親戚・挿抜不可

その他:

バッテリー、タッチパネル、
磁気センサー（開閉検知）など

ハードウェアの構成としては 初代 Raspberry Pi をもっと古く素朴にして
キーボードと LCD とバッテリーを付けた感じ

Linux のポーティング

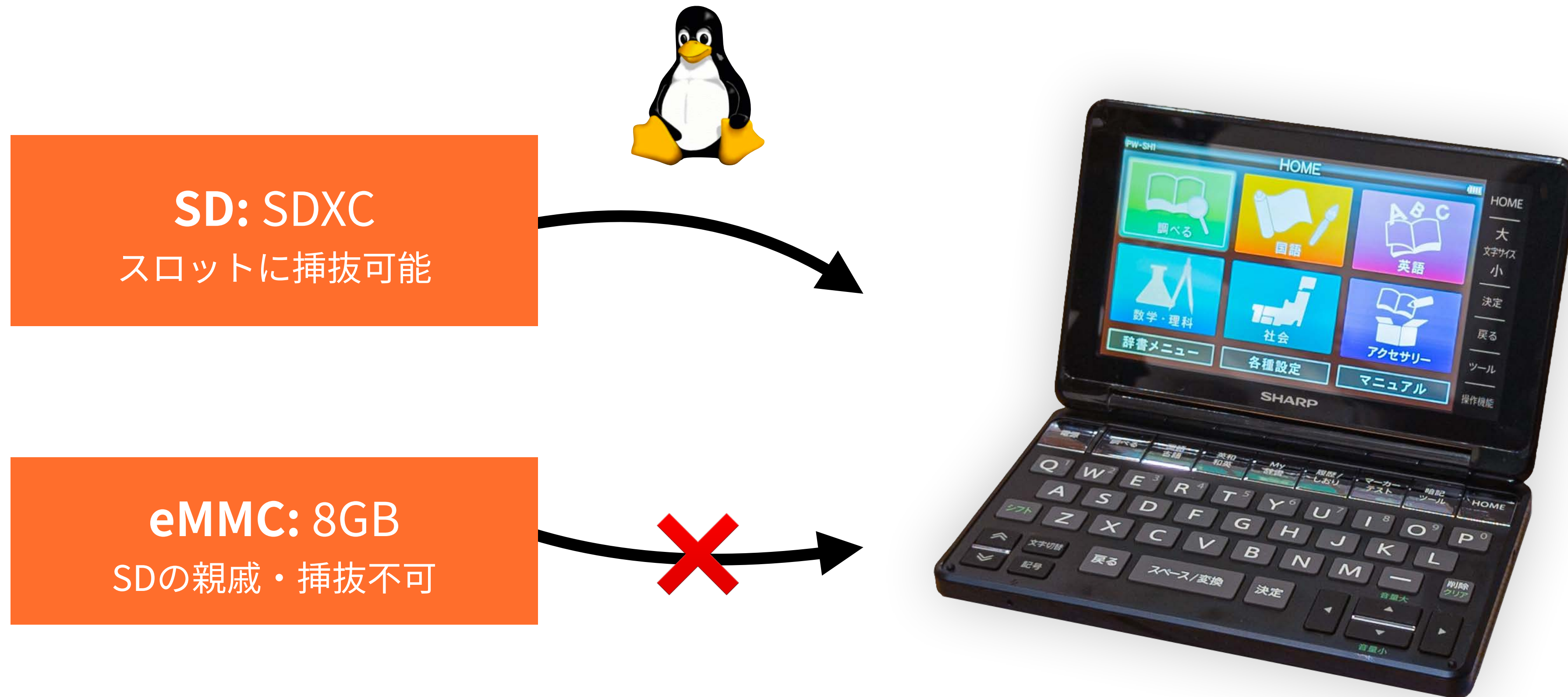
- 2020年 Linux のブートに成功
- コミュニティ "Brain Hackers" を立ち上げ
メンバーの努力でサポート機種を拡大
- Brain 向けにカスタムした Debian ベースの
Linux ディストリ **Brainux** をリリース
- OS イメージをダウンロードして SD カードに書
き込むだけで Brain で Linux が動かせるように



Brainux / Brain Hackers ロゴ

**ポーティングができても
まだやり残していることは山ほどあった**

その1つが「Windows を Linux で上書きインストールすること」



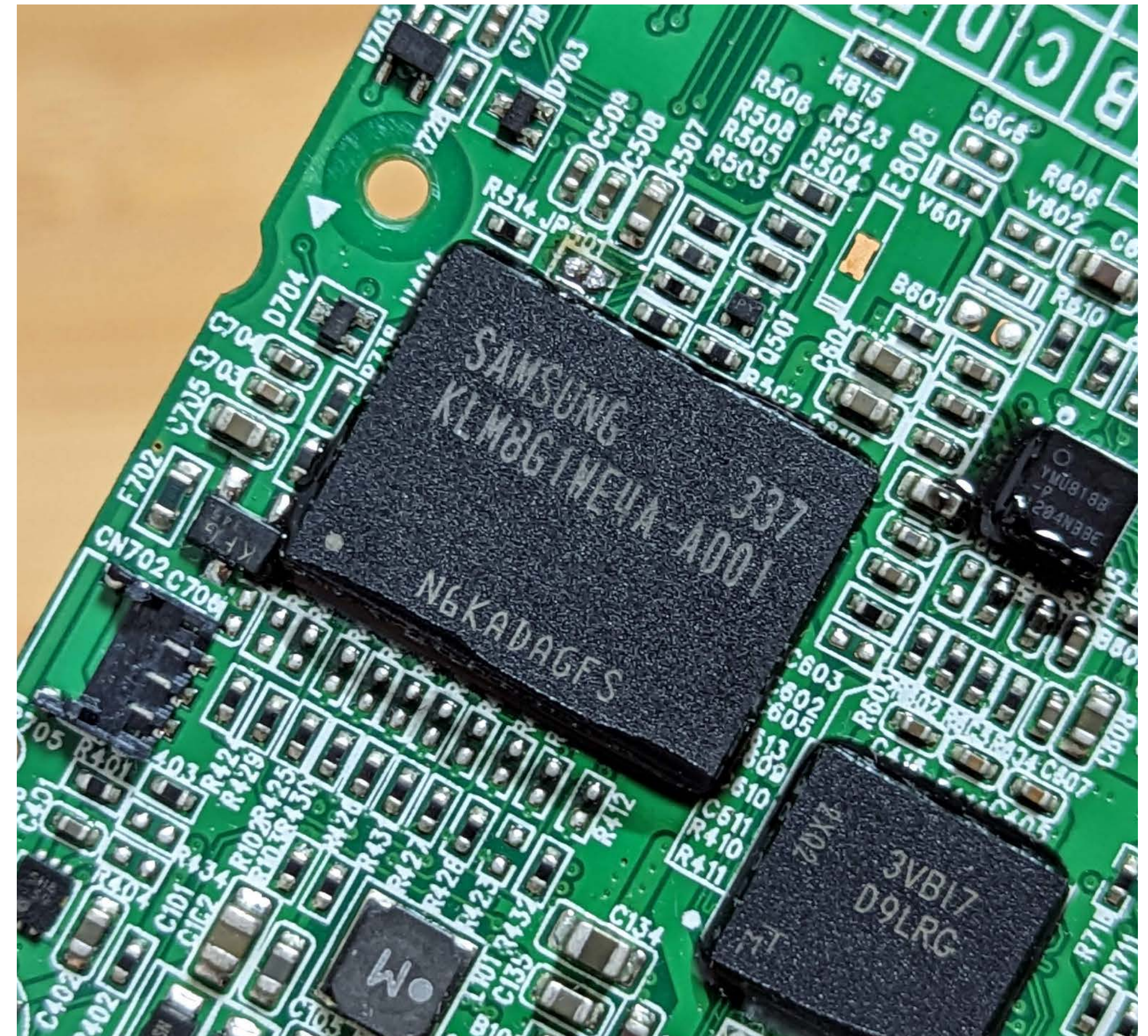
より正確に言うと「SD カードから Linux を起こす手法」は完成していたが
「本体内部の eMMC にインストールする手法」が未着手だった

eMMC に Linux が入ると何が嬉しいのか？

- SD よりもストレージ性能が上がる
- SD スロットが自由に使える

SD スロットの活用例

- SDIO にして Wi-Fi ドングルを接続
 - 単体で技適が通っていて Linux mainline にドライバがある SDIO Wi-Fi モジュールがある！
- GPIO にして任意の回路を接続
 - もはや開発ボード



**Brain 本体に Linux をインストールして
純然たる Linux マシンにしよう！！**

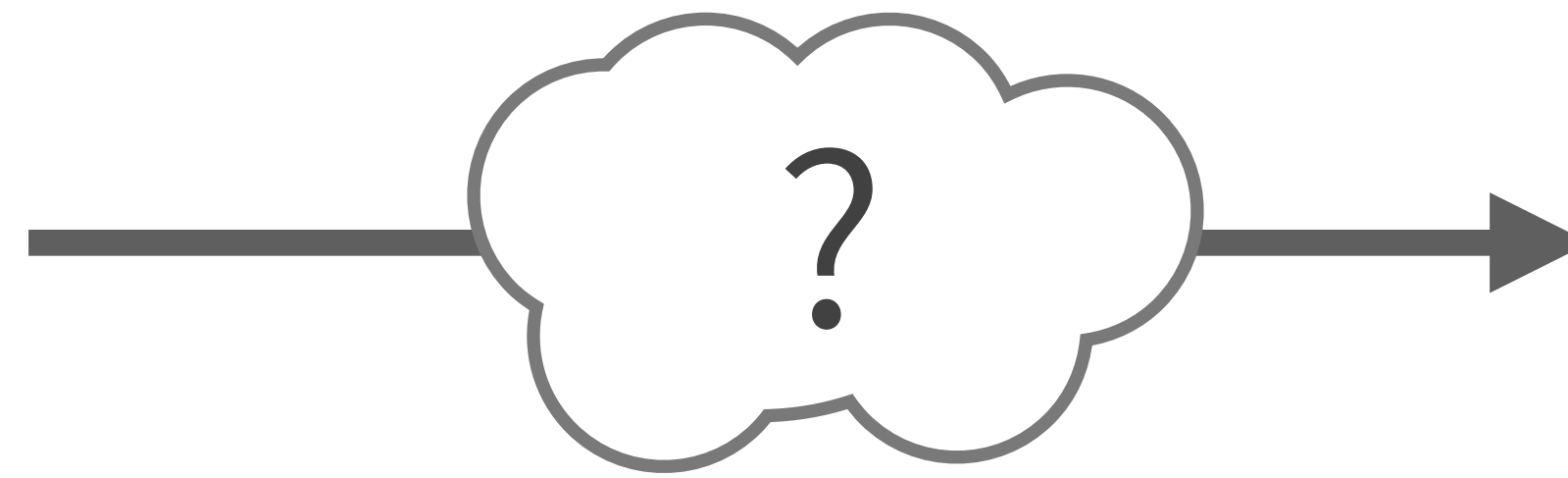
Good-bye, Windows！！ Good-bye, 電子辞書！！

eMMC から Linux を起こすために必要なもの

**電源が投入されたコンピューターは
「ブート」という過程を通じてハードウェアを初期化し
ソフトウェアを動かす準備をする**



Brain の電源を入れると、いくつかの「ブートローダー」によって
ブートが行われて Windows CE が起動する



Windows CE を
ブートする仕組み



リセット直後
すべての始まり



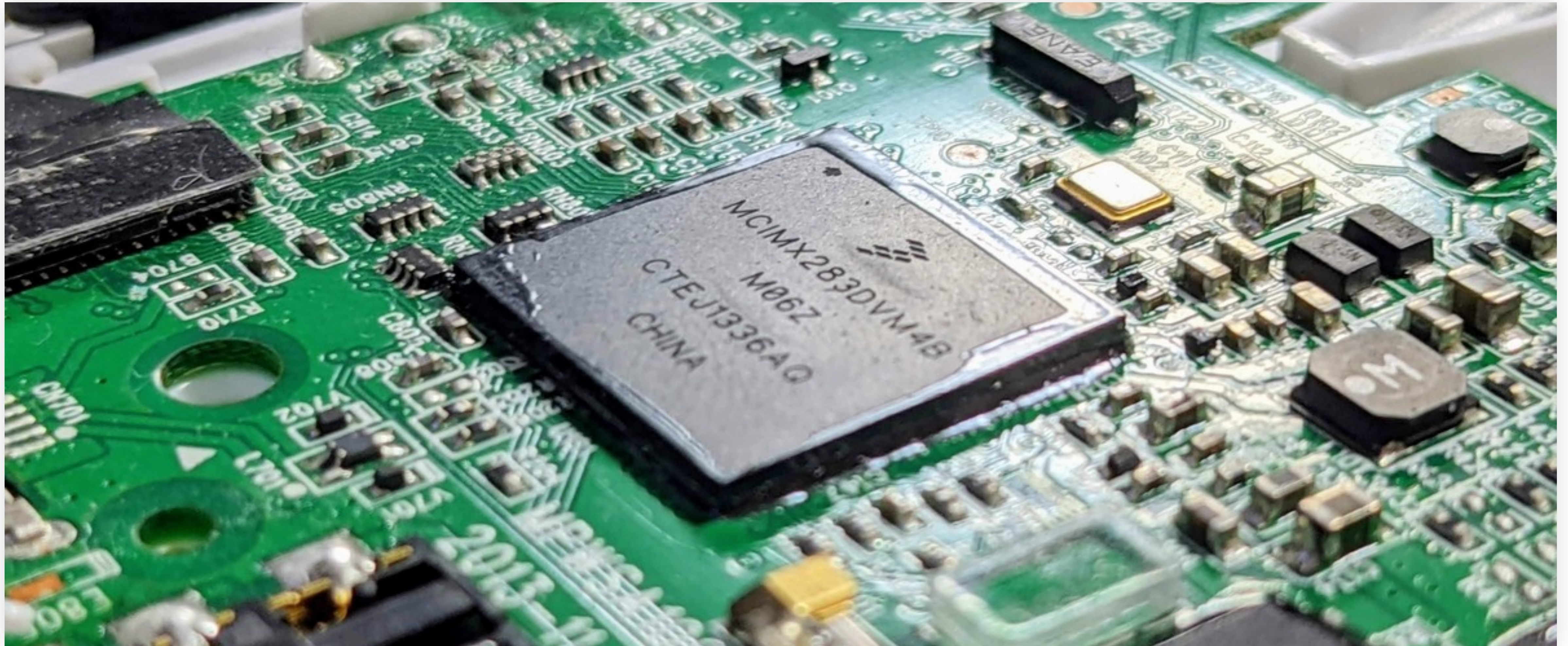
電源が入った瞬間からシステムが起動し終わるまでの流れを**ブートシーケンス**と呼ぶ

**SoC の動きとブートローダーのつながりを
電源投入直後から順番に追い
ブートシーケンスを把握**

**そのシーケンスを適切に上書きすれば
eMMC からのブートが実現できる！**

ブートシーケンスの解説と解析

リセット ～ Boot ROM



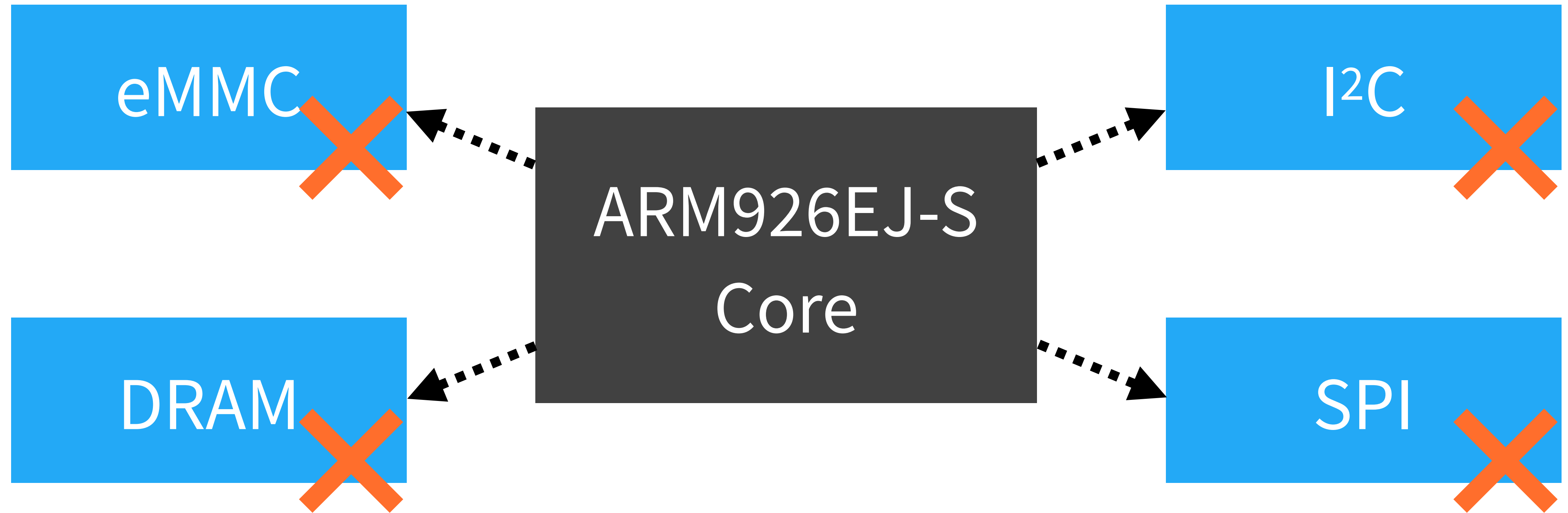
電源が入った直後の Arm SoC は何を実行するのか？

A. Boot ROM

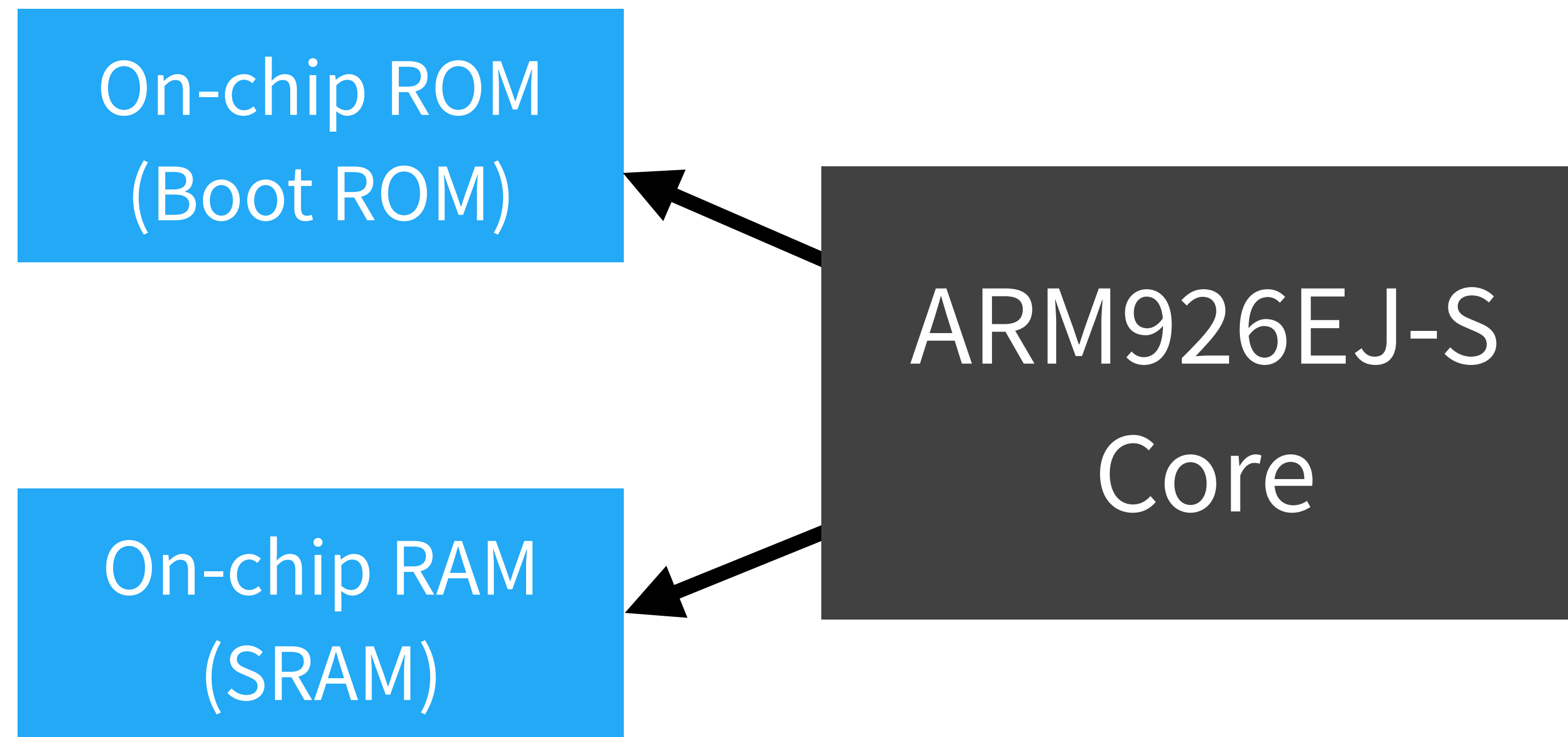
A. Boot ROM

(i.MX28 では On-chip ROM と呼称)

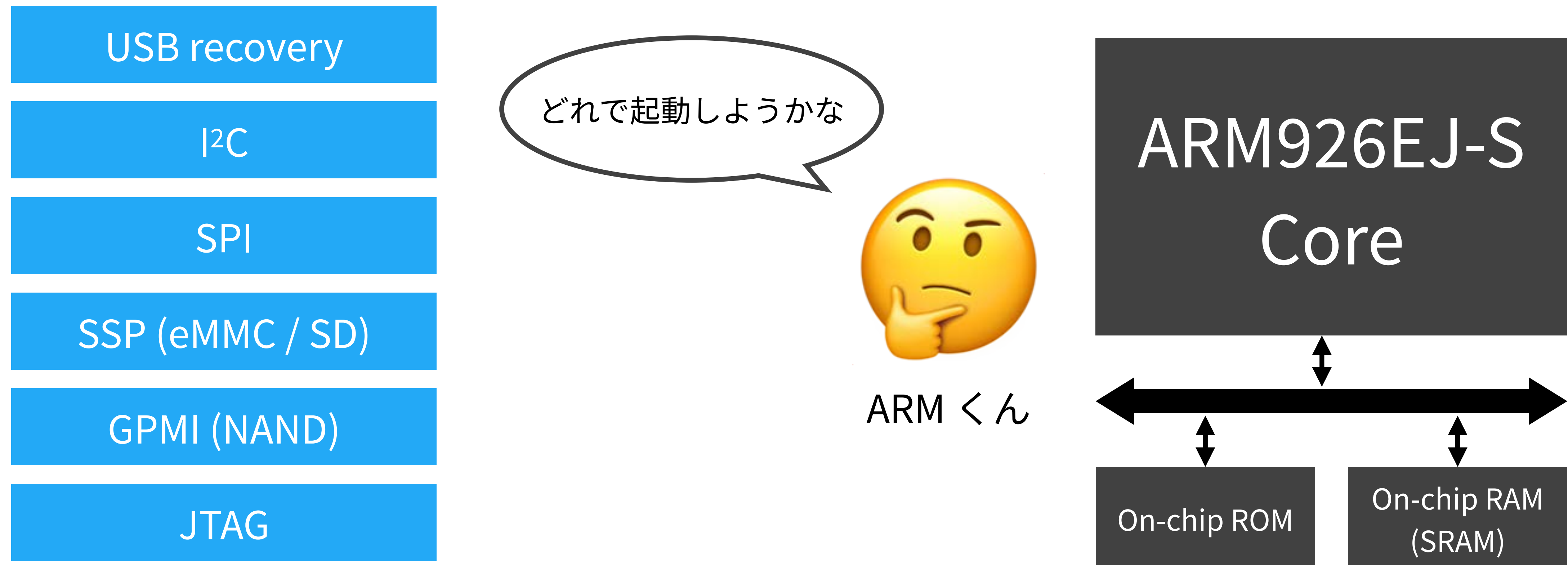
ブートシーケンスの解説と解析 — リセット ～ Boot ROM#5



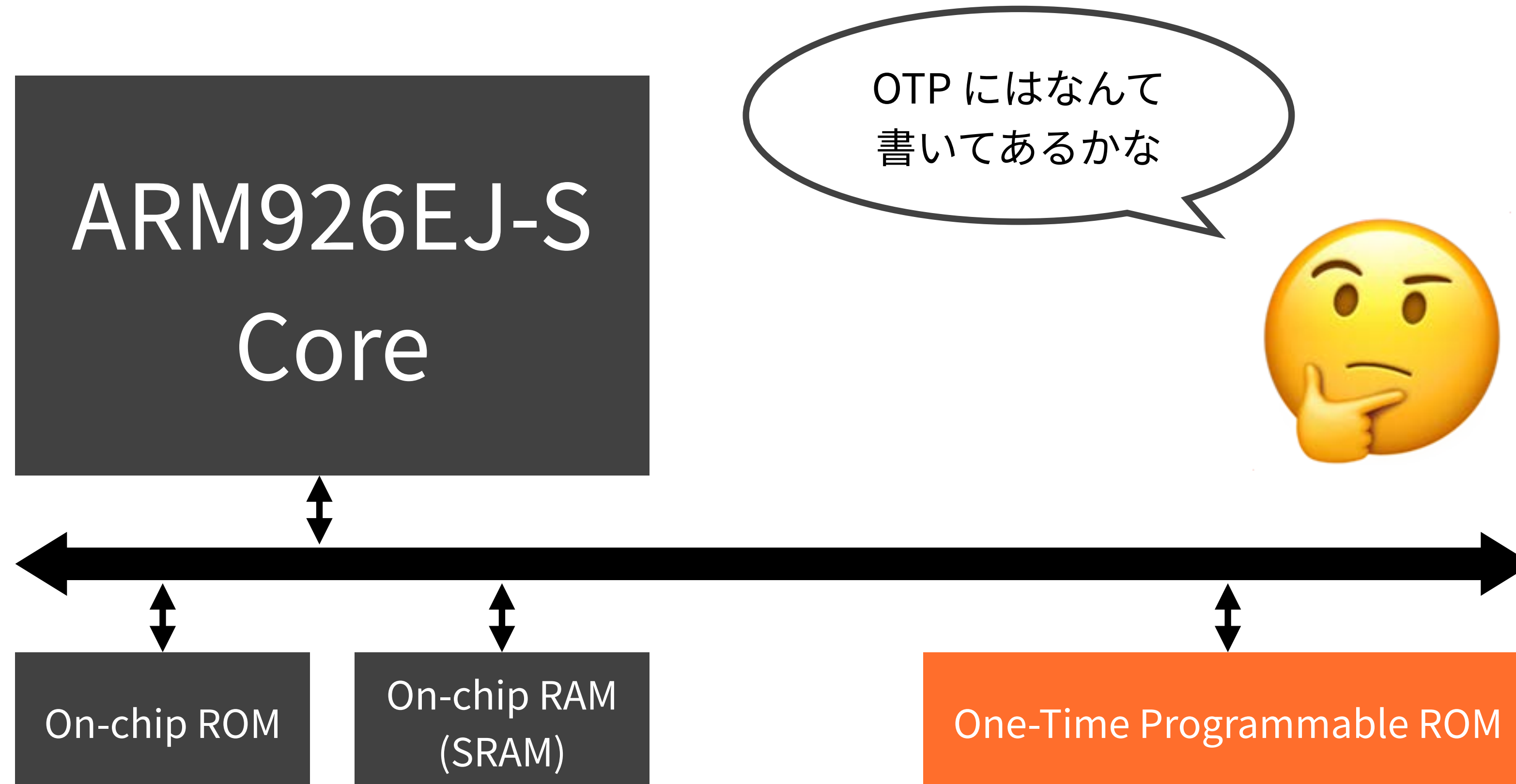
リセット直後は外部ペリフェラルが未初期化で使えないため
CPU は一般にバスで直接繋がった場所しかアクセスできない



On-chip ROM と On-chip RAM はリセット直後からバスでアクセスできるので
On-chip RAM を作業領域にしつつ On-chip ROM の命令列を実行する



CPU は Boot ROM つまり最初のブートローダーを使って
起動デバイス選択・ペリフェラル初期化・次段ブートローダー読み込みを行う

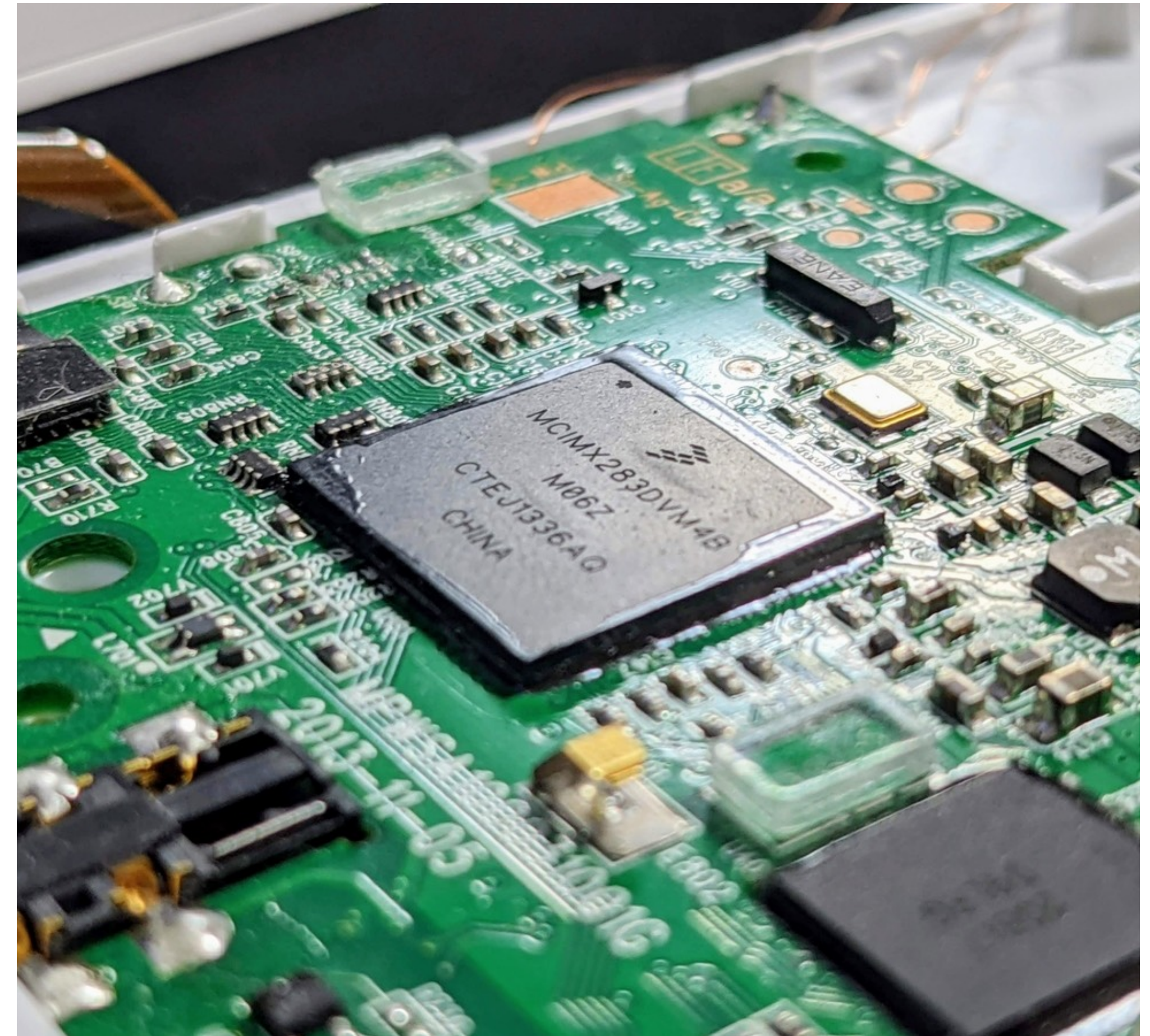


Boot ROM は次に続くブートローダーをどの外部デバイスから読み出すかを "One-Time Programmable ROM" に書いてある "Boot Mode" を見て決定する

One-Time Programmable (OTP) ROM とは？

- "OTP ROM" あるいは単に "OTP" とも
- SoC 中の半導体に実装された一度限りプログラム可能かつ消去不可な ROM
- 開発者の意思を CPU に伝えるのに使える最初の領域
- ブートデバイスによっては固有の設定が必要なのでそういった項目も OTP に書き込まれる

**まずは OTP の中身を見て最初の
ブートデバイスを見つけよう！**



ブートシーケンスの解説と解析

OTP の設定内容を見る

特徴

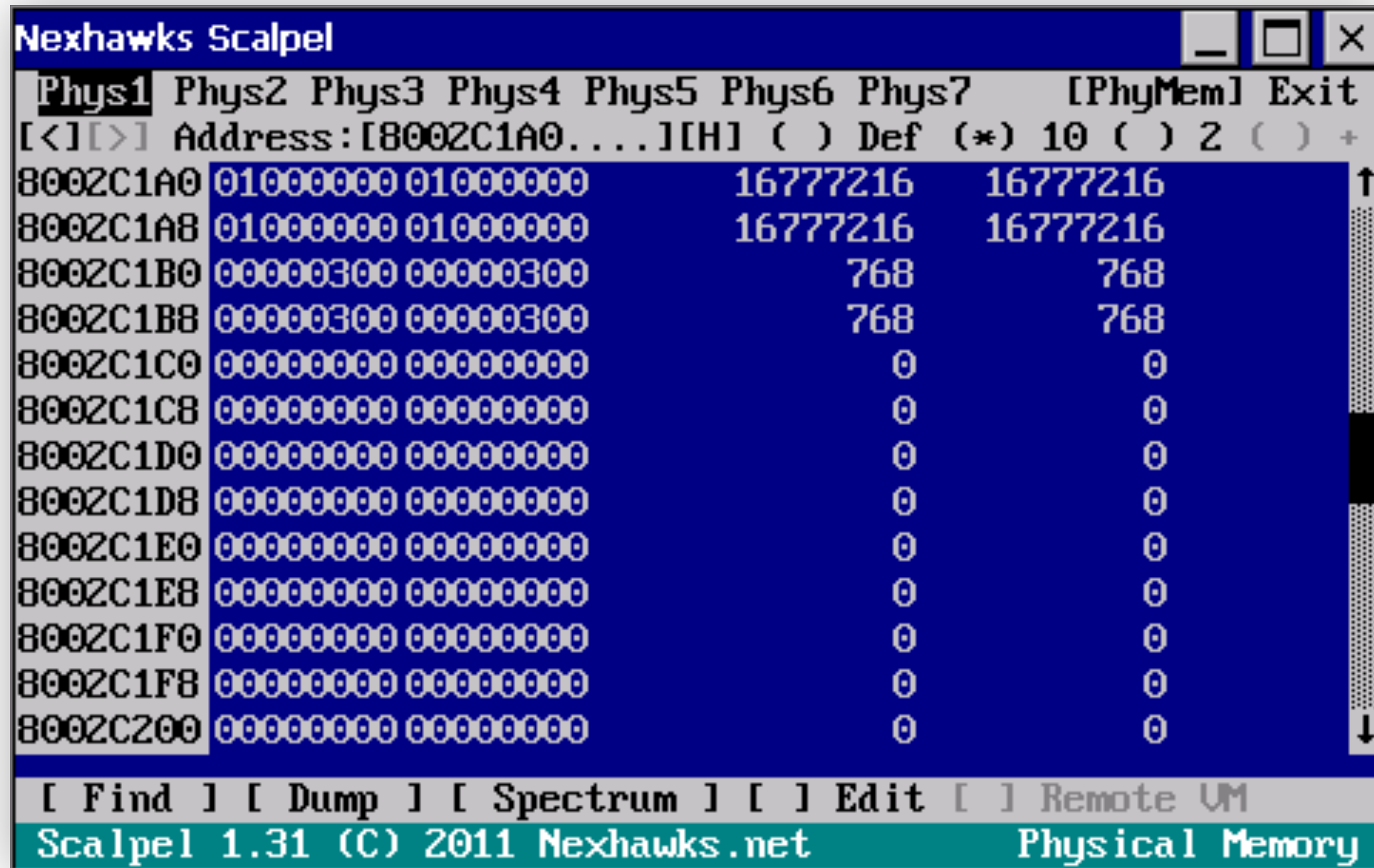
- SHARP が2008年に発売した電子辞書ブランド
- Windows CE を搭載（2020年の機種まで）
- CE 向けアプリの一部が動く
- **自分でコンパイルした exe も動く**



Brain で動く Windows CE アプリ

- アプリランチャー
- オフラインで Wikipedia を読むソフト
- matplotlib
- ギャルゲー
などなど…

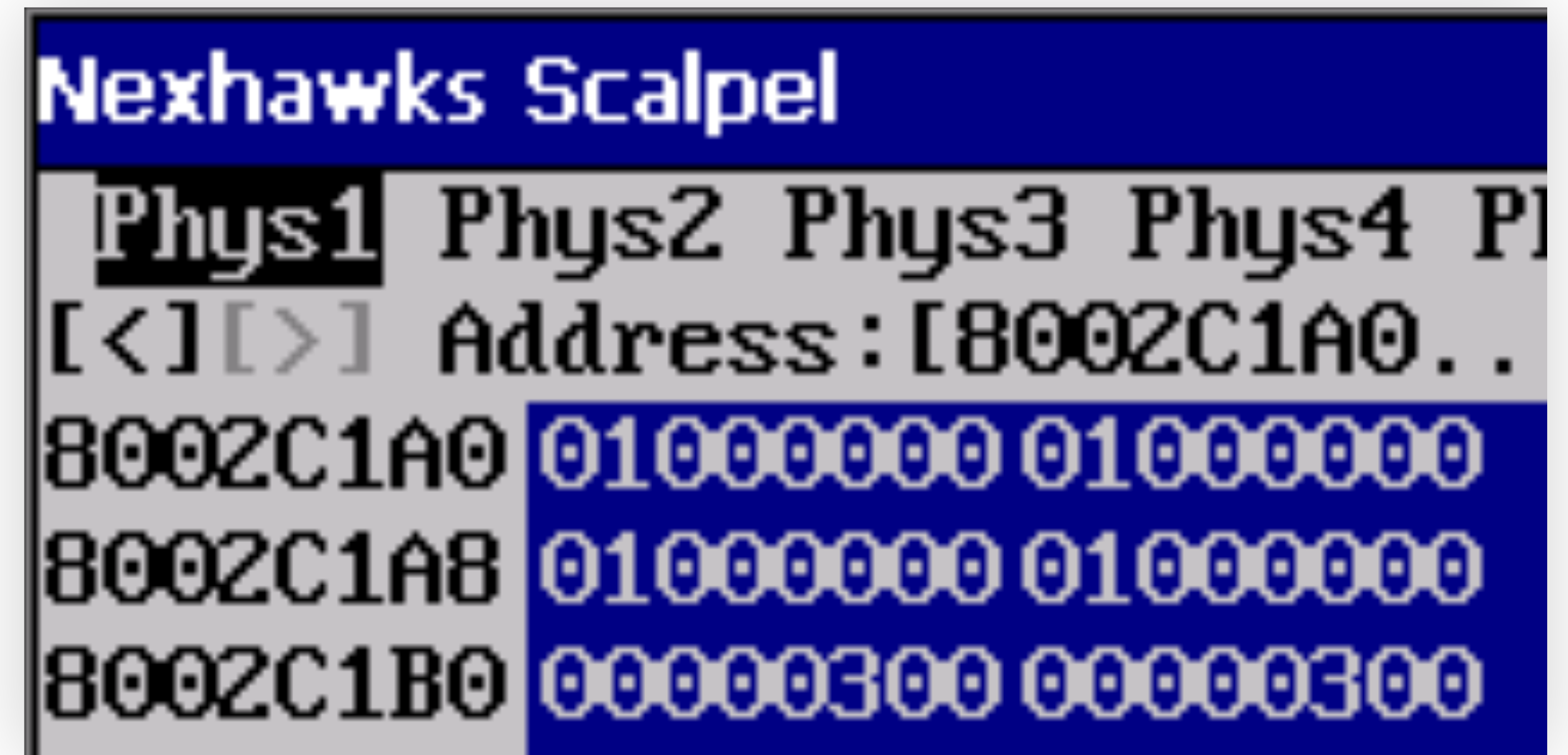
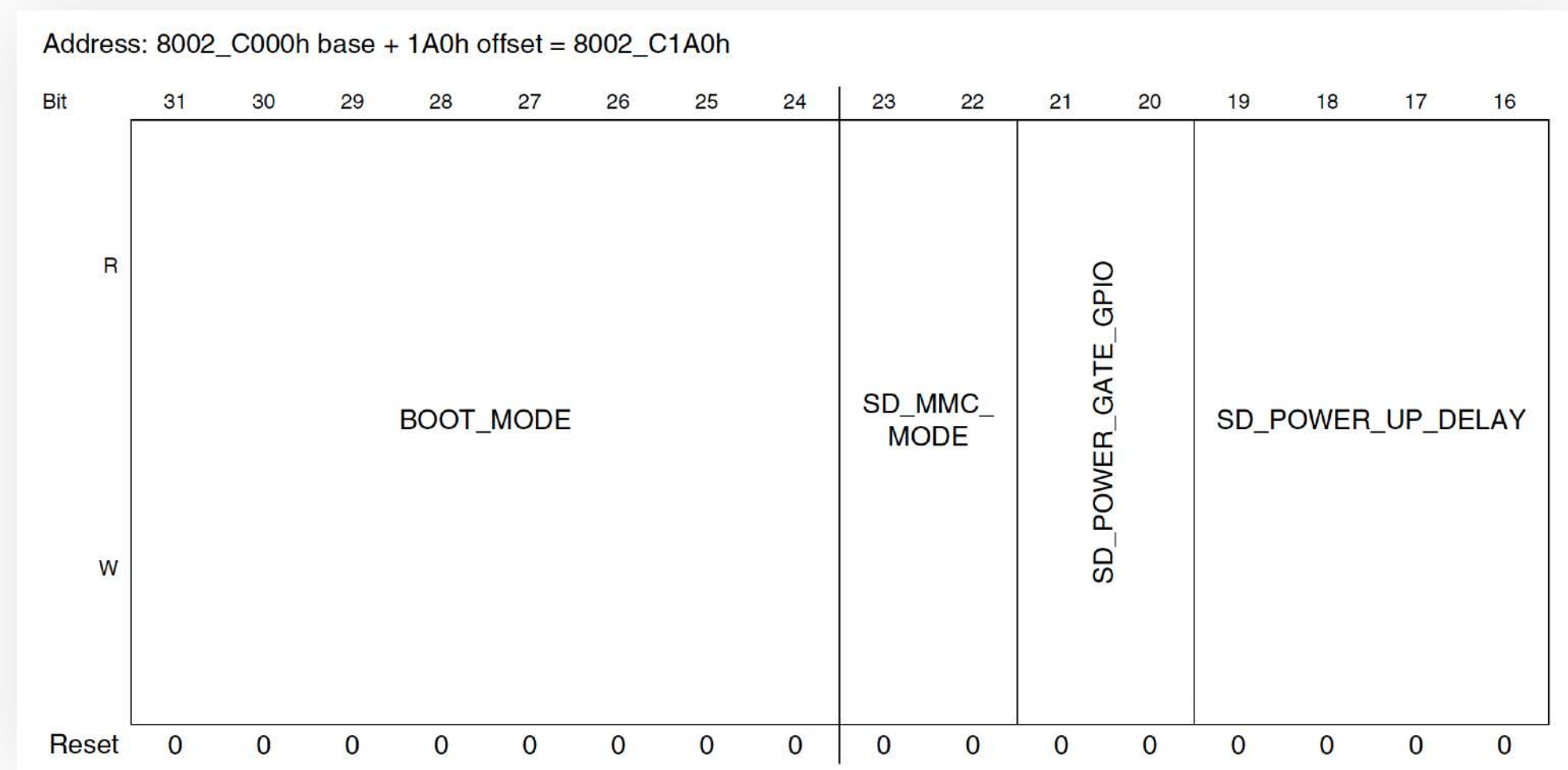
SHARP Brain PW-SH1



The screenshot shows the Nexhawks Scalpel application window. The title bar is 'Nexhawks Scalpel'. The menu bar includes 'Phys1', 'Phys2', 'Phys3', 'Phys4', 'Phys5', 'Phys6', 'Phys7', '[PhyMem]', and 'Exit'. The status bar at the bottom shows 'Scalpel 1.31 (C) 2011 Nexhawks.net' and 'Physical Memory'. The main display area shows a memory dump with columns for address, data, and physical address. The data is organized into rows, with the first two columns showing hexadecimal values and the third column showing decimal values. The address column ranges from 8002C1A0 to 8002C200. The data column shows various hexadecimal values, including 01000000, 00000300, and 00000000. The physical address column shows values like 16777216 and 768. A vertical scrollbar is on the right side of the data area.

Address	Phys1	Phys2	Phys3	Phys4	Phys5	Phys6	Phys7	[PhyMem]
8002C1A0	01000000	01000000				16777216	16777216	
8002C1A8	01000000	01000000				16777216	16777216	
8002C1B0	00000300	00000300				768	768	
8002C1B8	00000300	00000300				768	768	
8002C1C0	00000000	00000000				0	0	
8002C1C8	00000000	00000000				0	0	
8002C1D0	00000000	00000000				0	0	
8002C1D8	00000000	00000000				0	0	
8002C1E0	00000000	00000000				0	0	
8002C1E8	00000000	00000000				0	0	
8002C1F0	00000000	00000000				0	0	
8002C1F8	00000000	00000000				0	0	
8002C200	00000000	00000000				0	0	

Brain のメモリ空間が全部見える神ツール Scalpel を使って Windows CE 上で読む



データシートによると
アドレス 0x8002C1A0 の 32bit 値のうち
MSB 側 8bit が BOOT_MODE

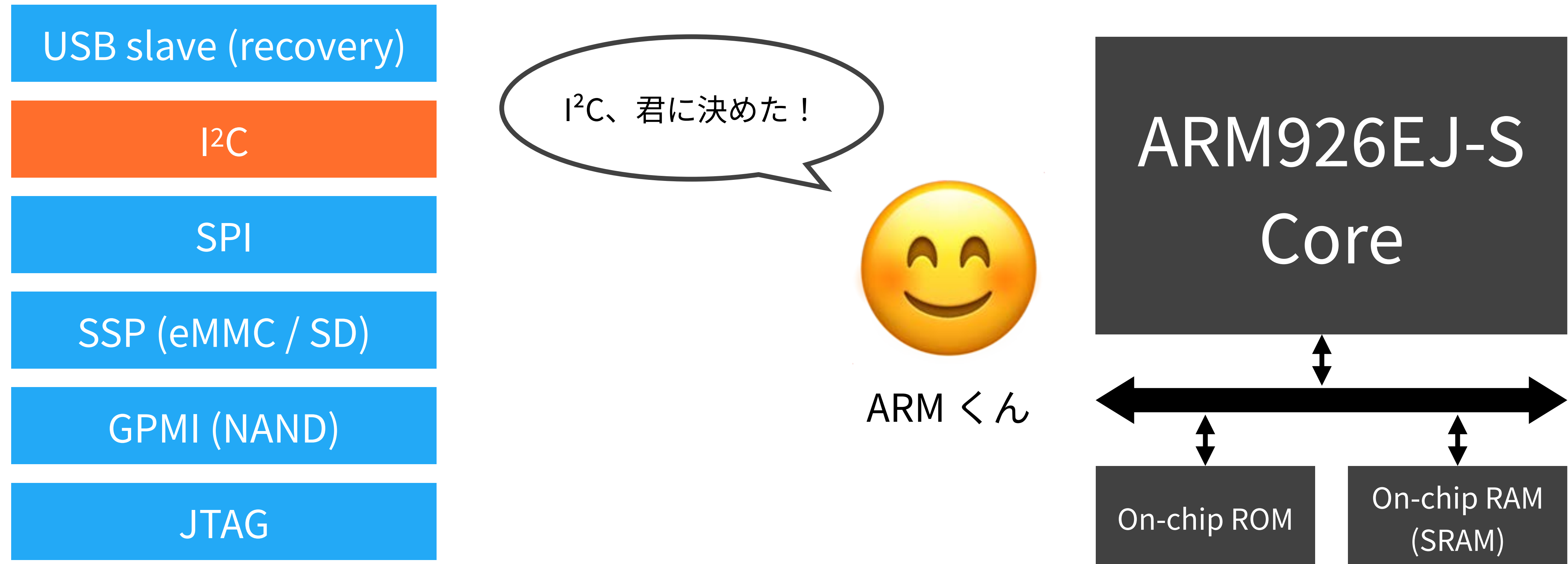
Scalpel で 0x8002C1A0 を読むと
BOOT_MODE は 0x01 (0b00000001)

12.2.2 Boot Mode Selection Map

Table 12-3. Boot Mode Selection Map

LCD_ D05	VOLTAGE SELECT(BM4)/ LCD_ D04	BM3/ LCD_ D03	BM2/ LCD_ D02	BM1/ LCD_ D01	BM0/ LCD_ D00	PORT	BOOT MODE
x	x	0	0	0	0	USB0	USB (unencrypted vs. encrypted is under OTP control)
x	0	0	0	0	1	I2C0	I2C0 master, 3.3 V
x	1	0	0	0	1	I2C0	I2C0 master, 1.8 V

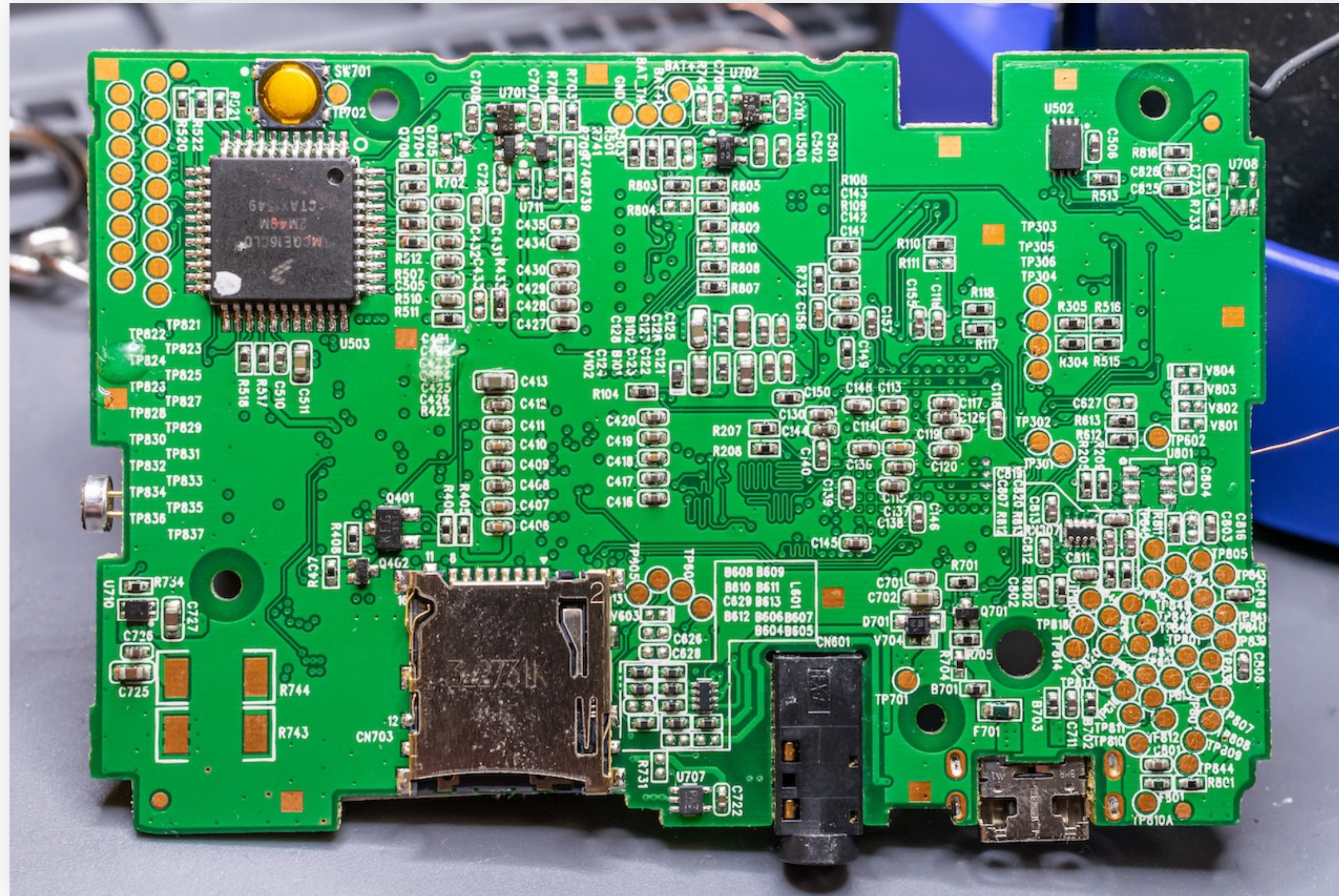
データシートによると BOOT_MODE 0bXXX00001 は I²C 3.3V



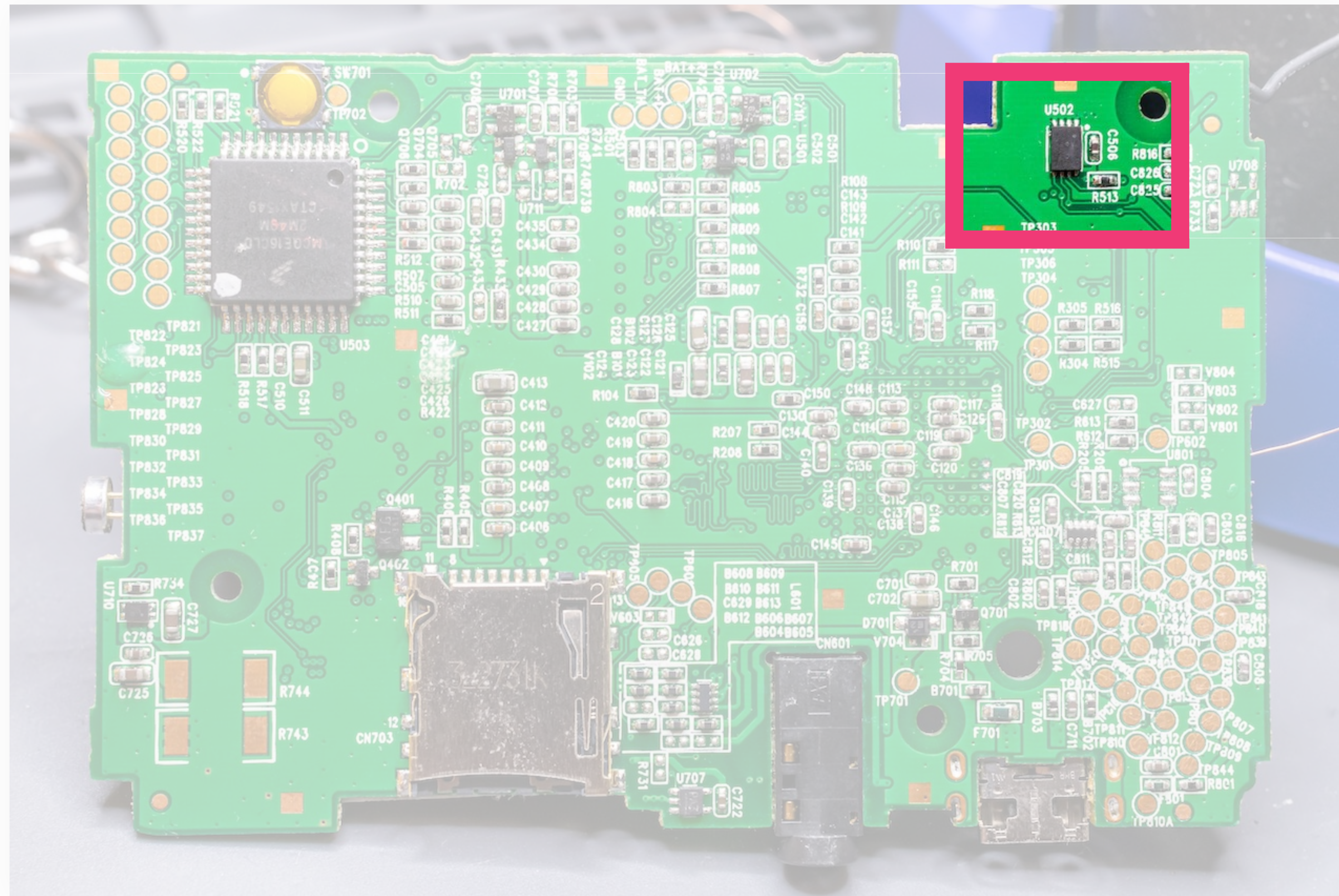
… こうして、CPU は I²C で接続された EEPROM を最初に読みに行くことが判明

ブートシーケンスの解説と解析

EEPROM の中身を吸い出して読む



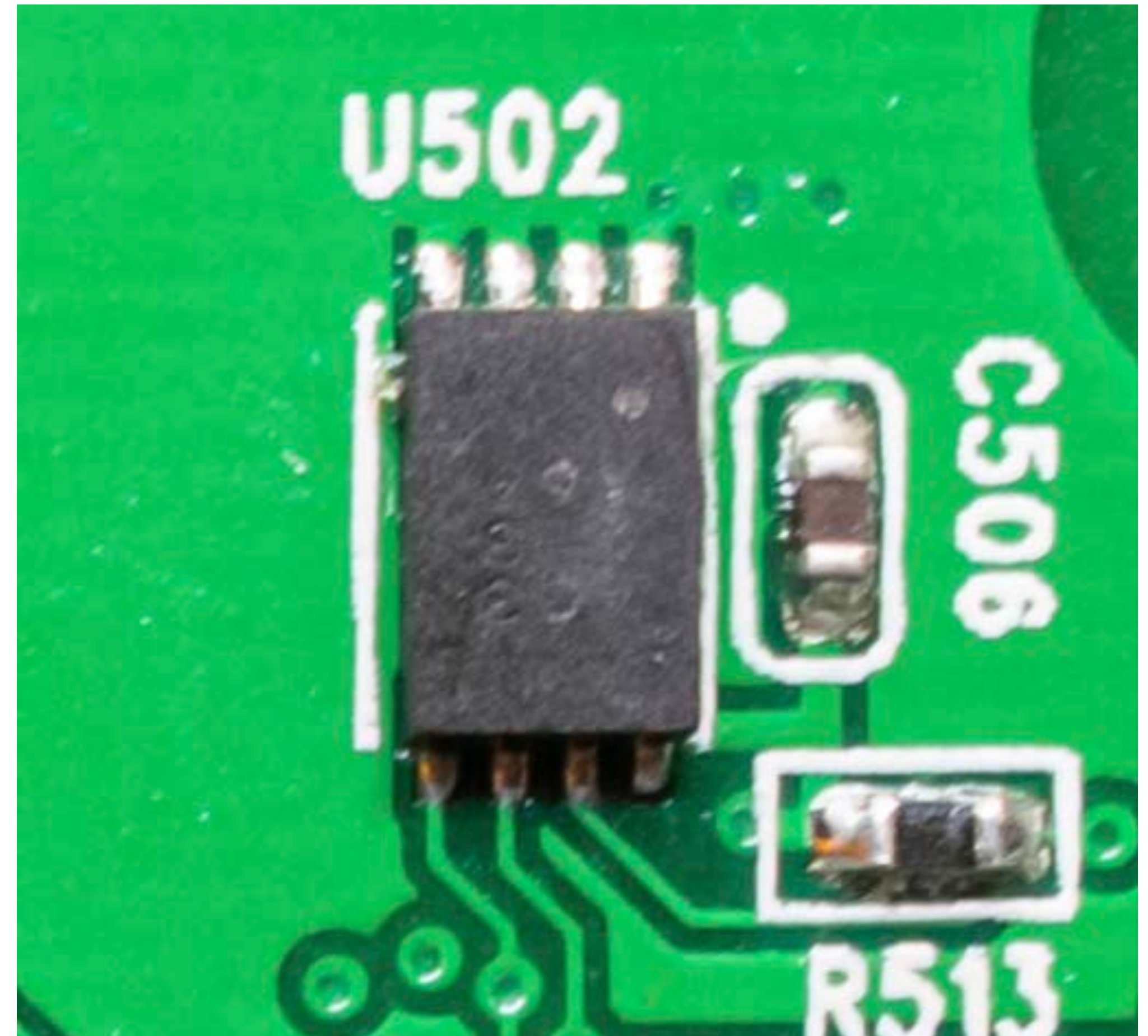
Q. これは Brain の基板画像です。EEPROM はどこにあるでしょう？

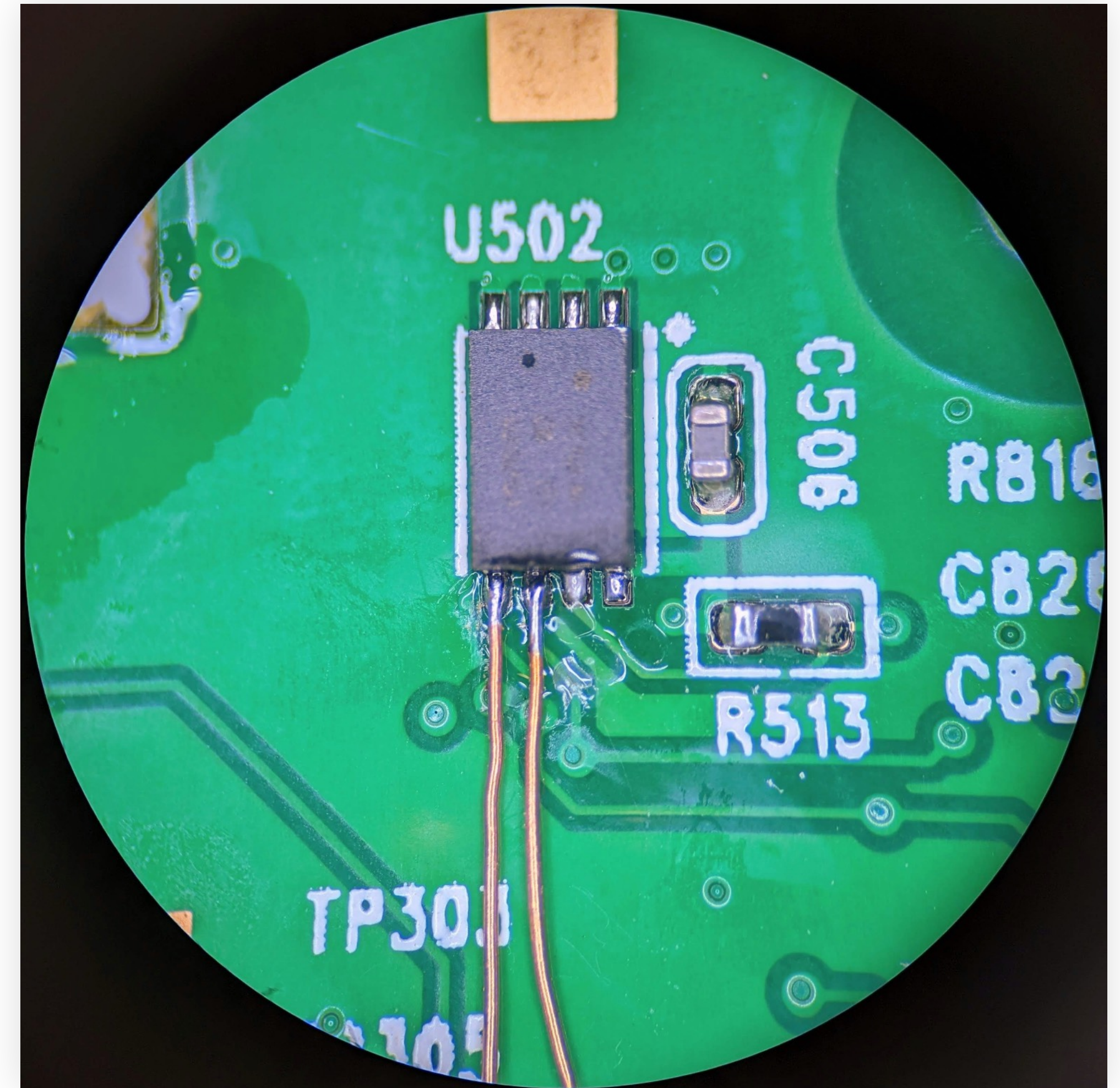


正解はここ

EEPROM を見つけて素性を特定する

- それっぽい部品のピンにオシロスコープのプローブを押し当てて電源を入れる…を繰り返す
- I²C であれば 100kHz か 400kHz のクロックが出る
- EEPROM を見つけたら、部品のフットプリントで Digi-key の在庫を絞り込み型番を特定する
 - 足が出ていない → **SON**
 - 2mm x 3mm → **ON Semi** か **Rohm**
 - 厚み 0.5mm → **Rohm** の **VSON** 形 **EEPROM**
- IC 表面に 4G3 とあるので BR24G32 = **4KB**





実体顕微鏡下で 0.2mm のポリウレタン線をはんだ付け・ロジックアナライザで読む

EEPROM の中身

- 中には構造体 "Boot Stream" (SB) が入っていた
- SB に書かれたコマンドを Boot ROM が逐次実行する
- Rockbox※のリポジトリに置いてあるツール sbtoelf で得たバイナリを Ghidra で disassemble してみる

```
1 DISPLAYPROGRESS
2 SECTION 0x0 BOOTABLE
3 TAG LAST
4 LOAD      0x1000      spl/u-boot-spl.bin
5 LOAD IVT  0x8000      0x1000
6 CALL HAB  0x8000      0x0
7 LOAD      0x40200000  u-boot.bin
8 LOAD IVT  0x8000      0x40200000
9 CALL HAB  0x8000      0x0
```

Boot ROM が実行するコマンドの例 (U-Boot)

※ Rockbox: 携帯プレーヤーを多機能化するオープンな代替ファームウェア

SB Header

Encryption keys

Section 0

Cmd: LOAD @ 0xE440

Cmd: FILL @ 0xE6A0 ~ +0x114

Cmd: LOAD @ 0xE400

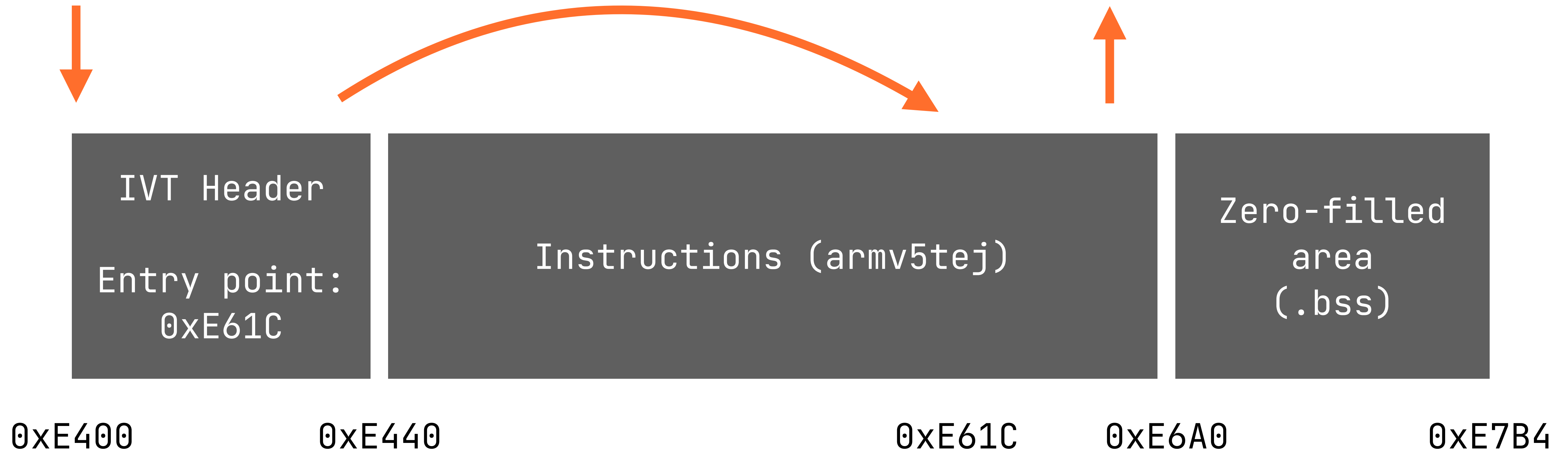
Cmd: CALL @ 0xE400

Cmd: MODE to 0x09

① CALL command

② Jump

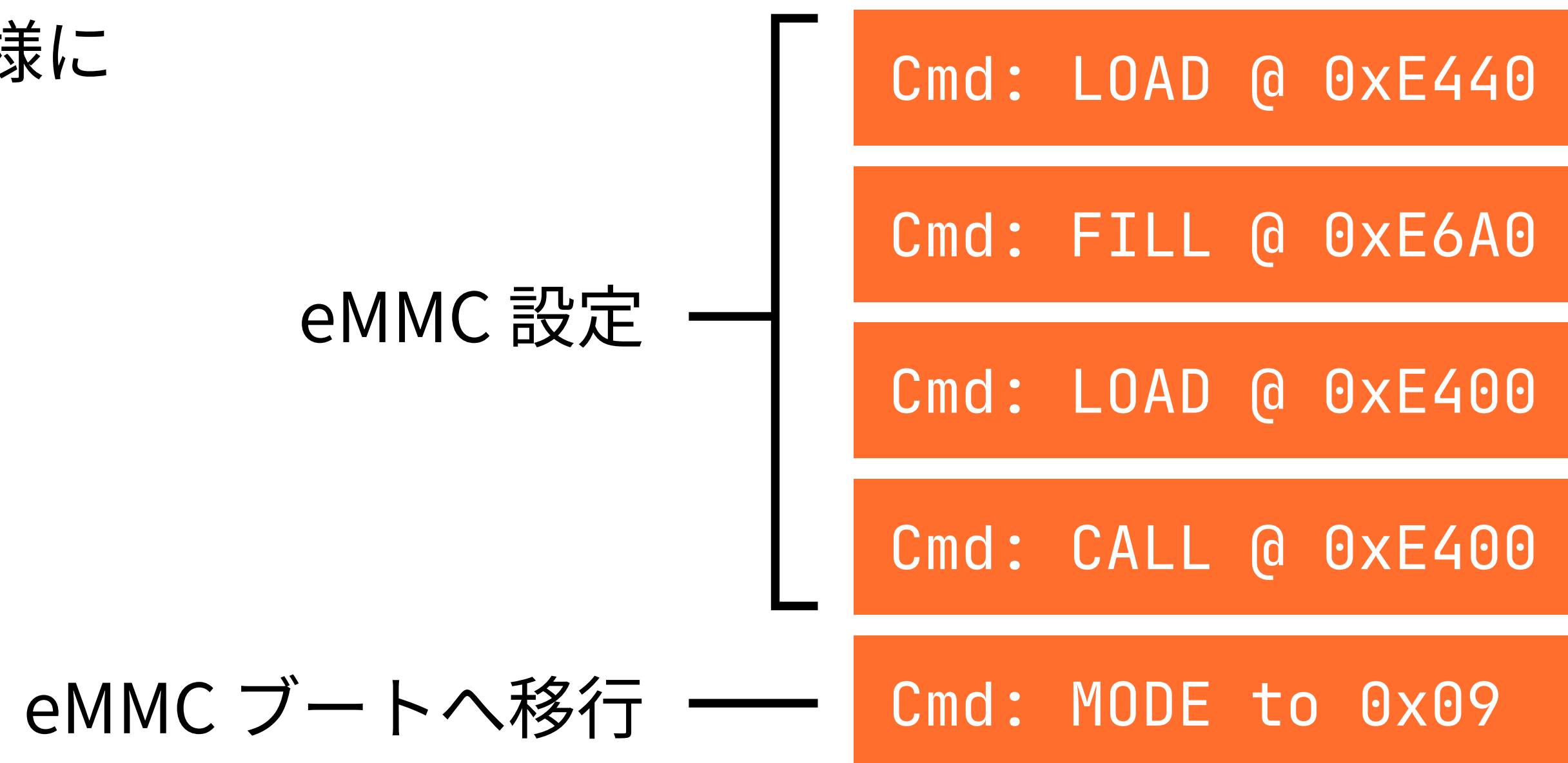
③ Return



SB のコマンドを実行した結果 SRAM に展開されるバイナリの構造とジャンプの流れ

EEPROM を読んで Boot ROM がやること

- eMMC のペリフェラル（それ以外もあるかも）に固有の設定値を書き込んで eMMC ブートへ移行
- eMMC ブートに移行すると EEPROM 同様に eMMC にある SB を解釈する





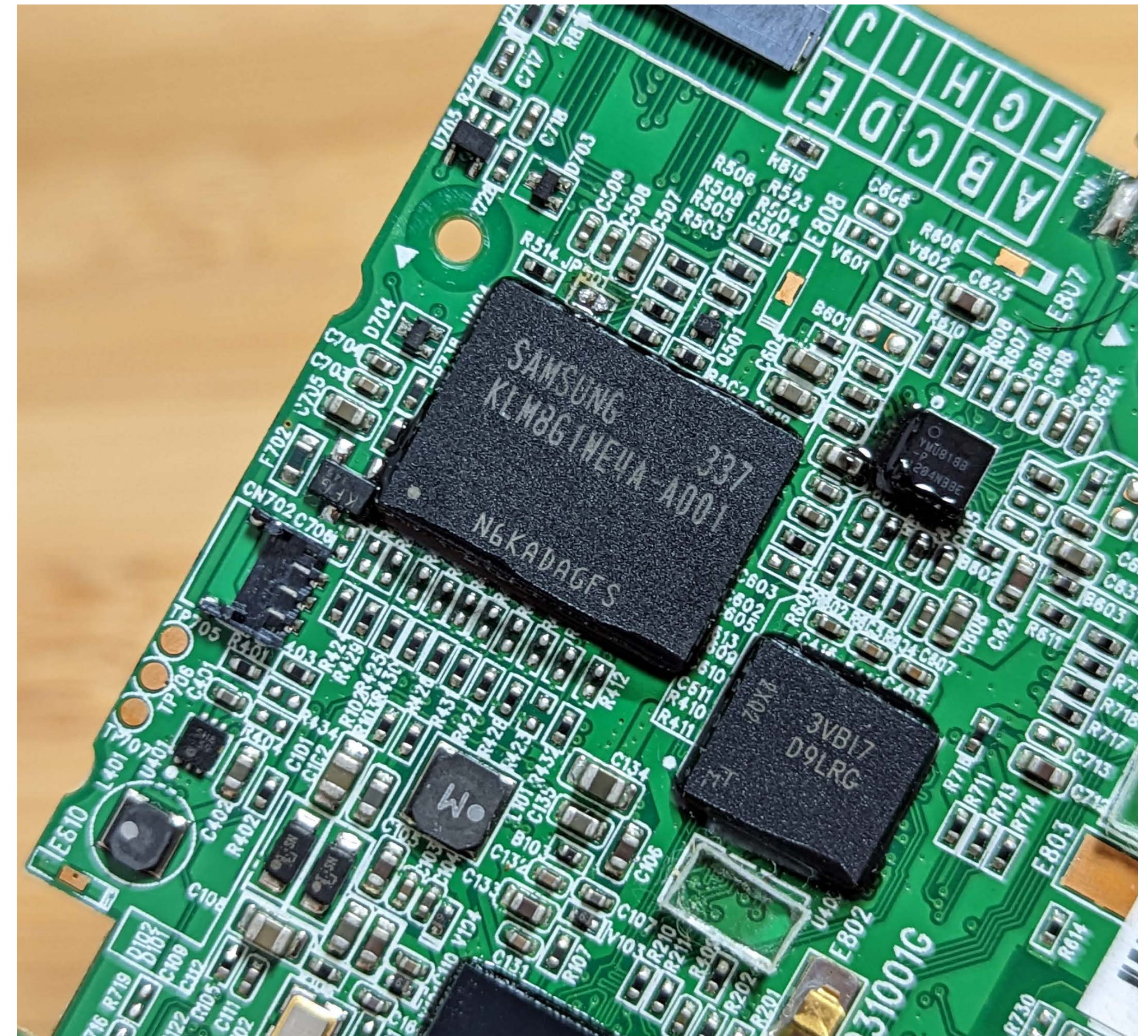
… こうして CPU は I²C EEPROM を読んで eMMC の設定を行った上で eMMC ブートに移行することが判明

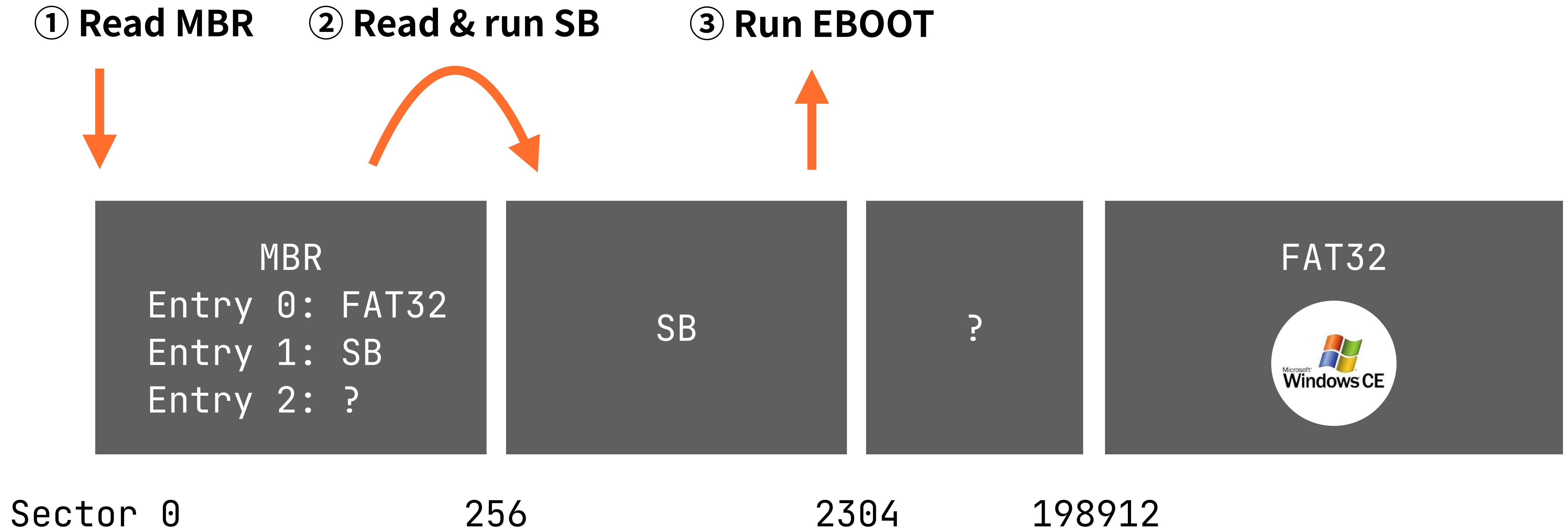
ブートシーケンスの解説と解析

eMMC の中身を吸い出して読む

eMMC の吸い出しは簡単

- Linux を SD から起動して dd で /dev/mmcblk0 を吸うだけ
- Master Boot Record から i.MX28 固有の partition type 0x53 がついたエントリーを探す
- 該当パーティションの冒頭に SB へのポインタ (セクタ番号) がある

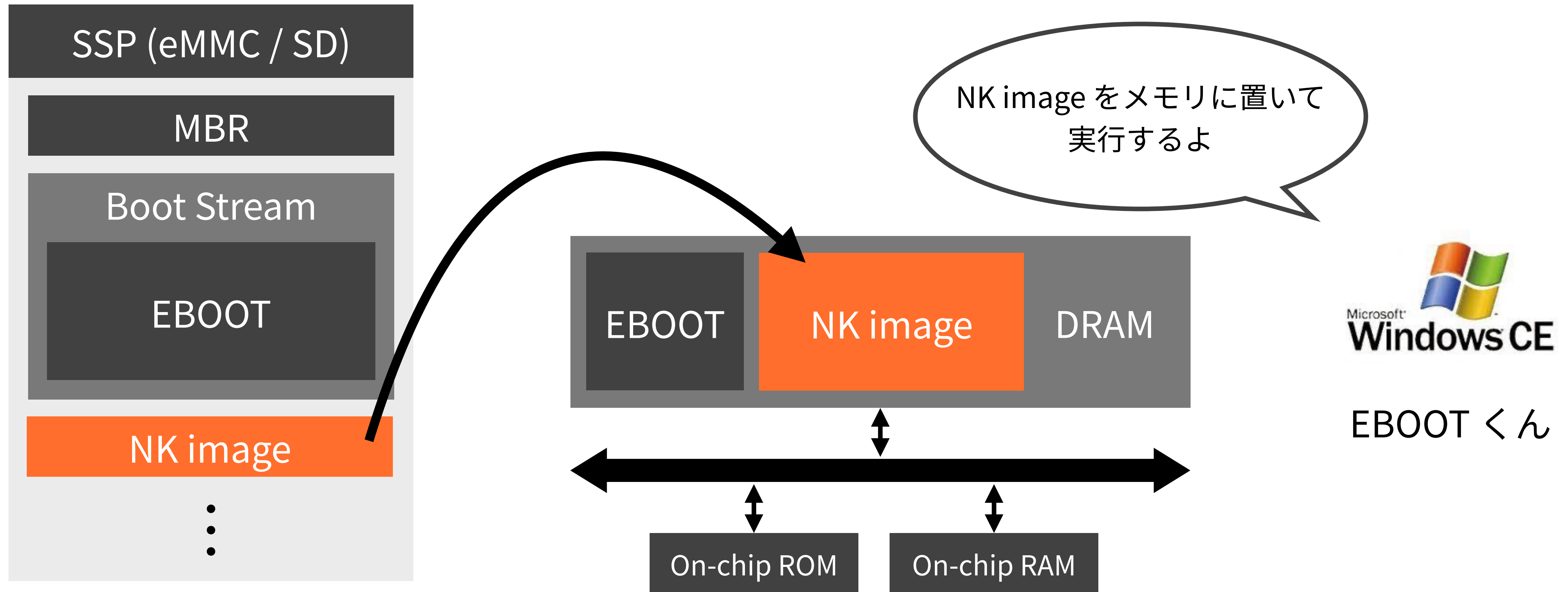




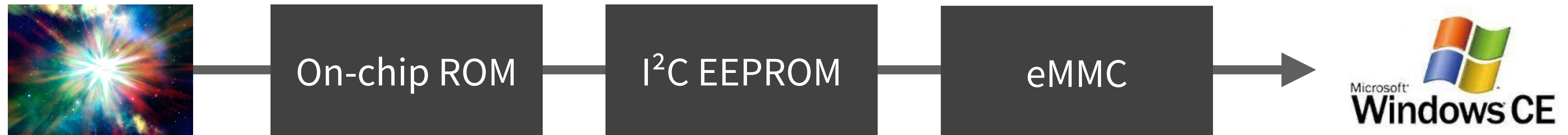
eMMC 上でブートに関わるデータと実行の流れ



… こうして CPU は eMMC を読んで EBOOT (Windows CE のブートローダー) を実行することが判明



あとは EBOOT が NK image (Windows のシステムを固めたもの) を eMMC から DRAM に展開しジャンプすると Windows CE が立ち上がる



リセットから Windows CE までのブートシーケンスが完全に判明！

ブートシーケンスのどこを書き換えるか決める

どこに自作バイナリを入れれば Linux が起こせるか？

I²C EEPROM にある SB は回路起因の調整をするだけで OS 非依存

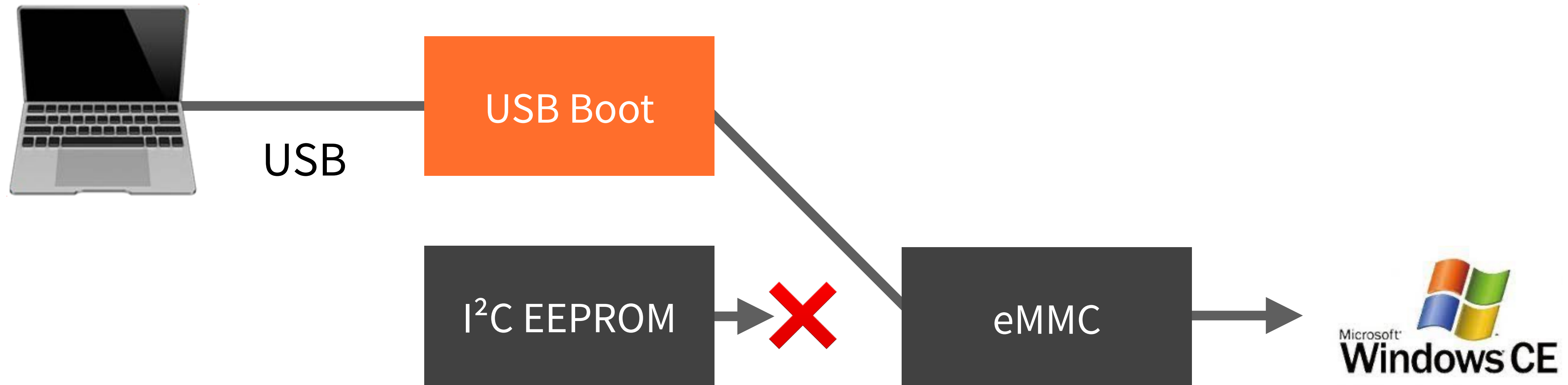
eMMC にある SB は Windows CE の EBOOT が入っている

つまり、eMMC の中身をすべて自作のものに置き換えれば良さそう？



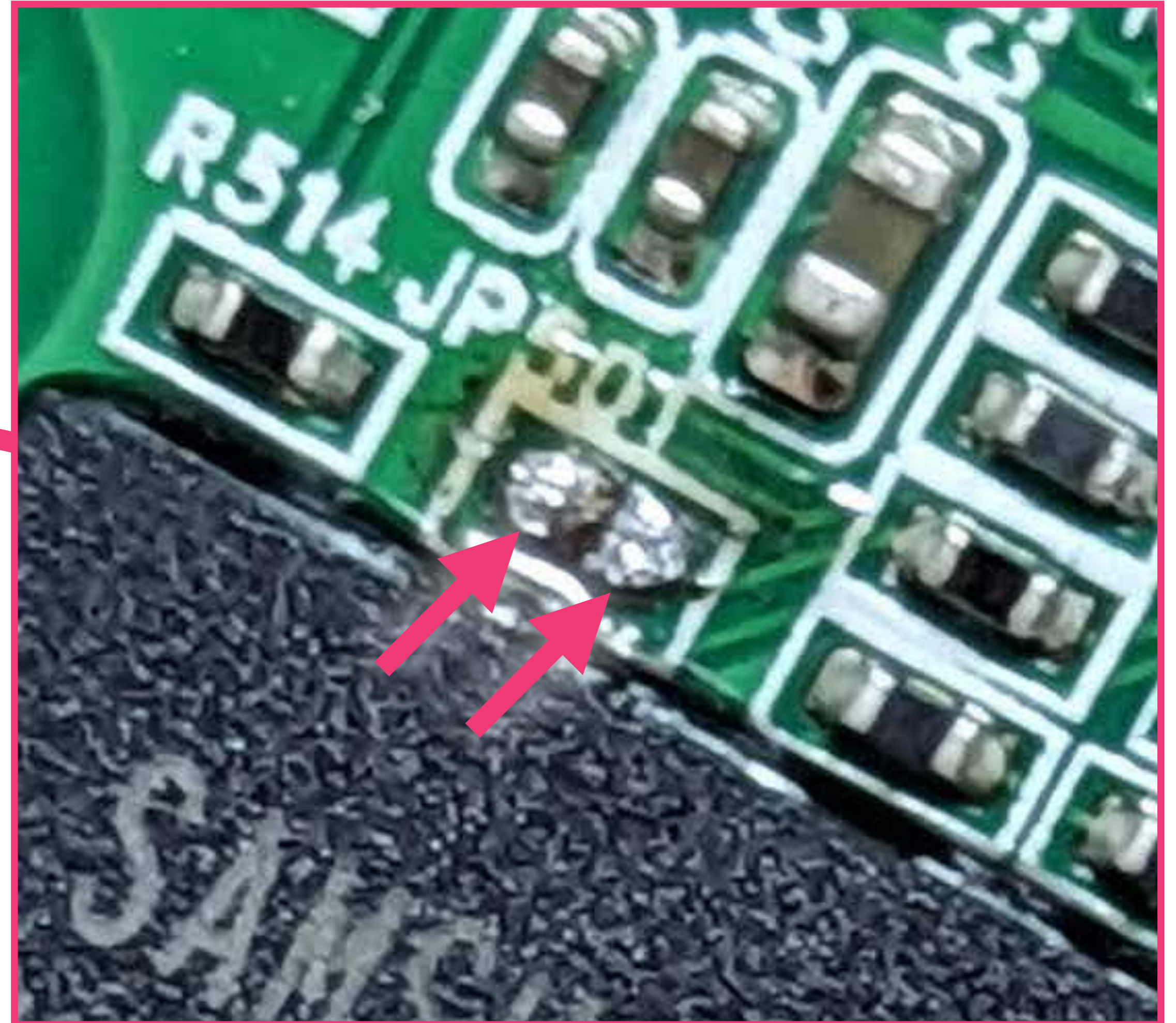
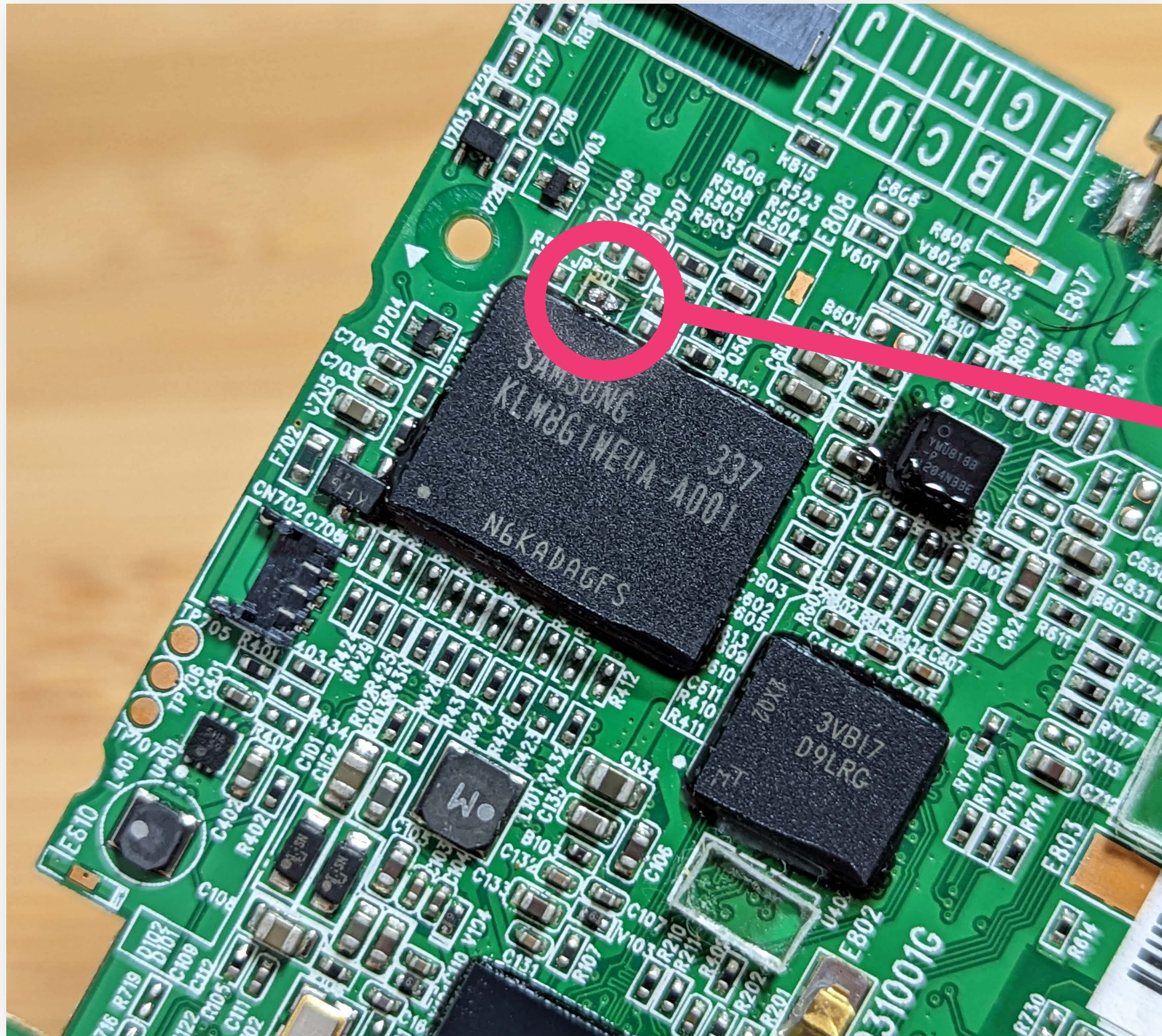
ブートシーケンスのどこを書き換えるか決める

実験: USB boot → eMMC boot で Windows を起こす



"USB boot mode" で PC から SB を流し込むと EEPROM を介さないブートが可能
eMMC ブートに移行するだけの SB を PC から流し込み Windows CE が起動するなら
EEPROM の内容は OS 非依存と確定できる

※SB の生成には U-Boot の mkimage を使用



eMMC 横のランドを導通させて USB 接続すると強制的に USB boot mode に入る

結果: 成功

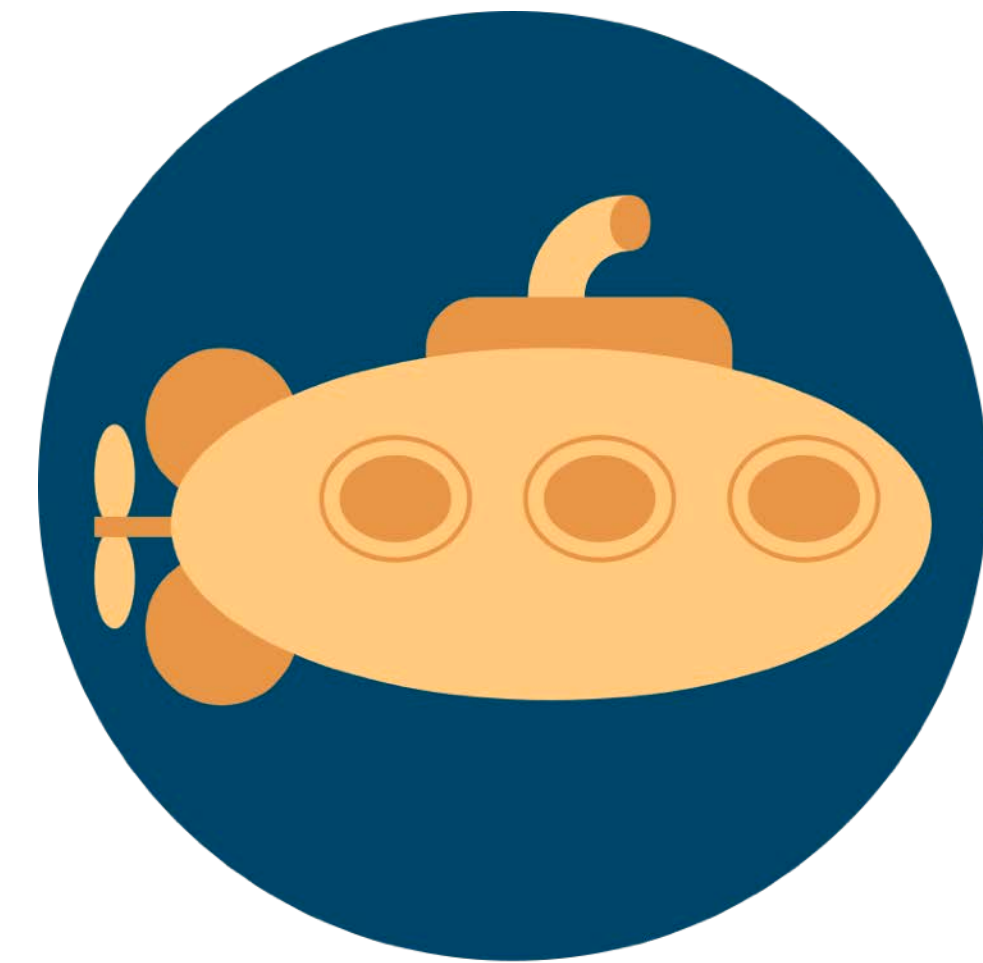
EEPROM はそのままにして eMMC を書き換える方向で確定



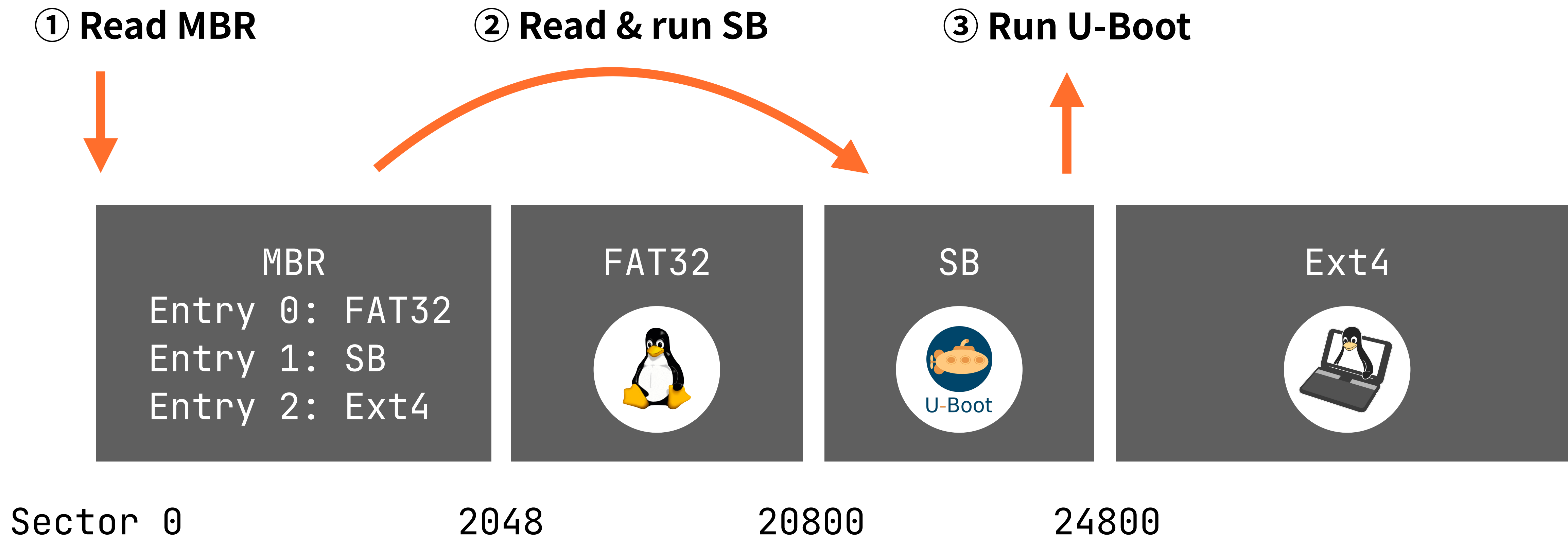
ブートローダー入り eMMC イメージを作る

eMMC に書き込む新たなローダーと OS イメージ

- 組み込み Linux でよく使われる U-Boot (Das U-Boot) を使用
- もともと SD から Linux kernel 等を取り出していた U-Boot に eMMC から取り出すように変更を加えれば準備完了
- SD 用の OS イメージ生成スクリプトに手を加えて SB と U-Boot を書き込んだものを生成



U-Boot

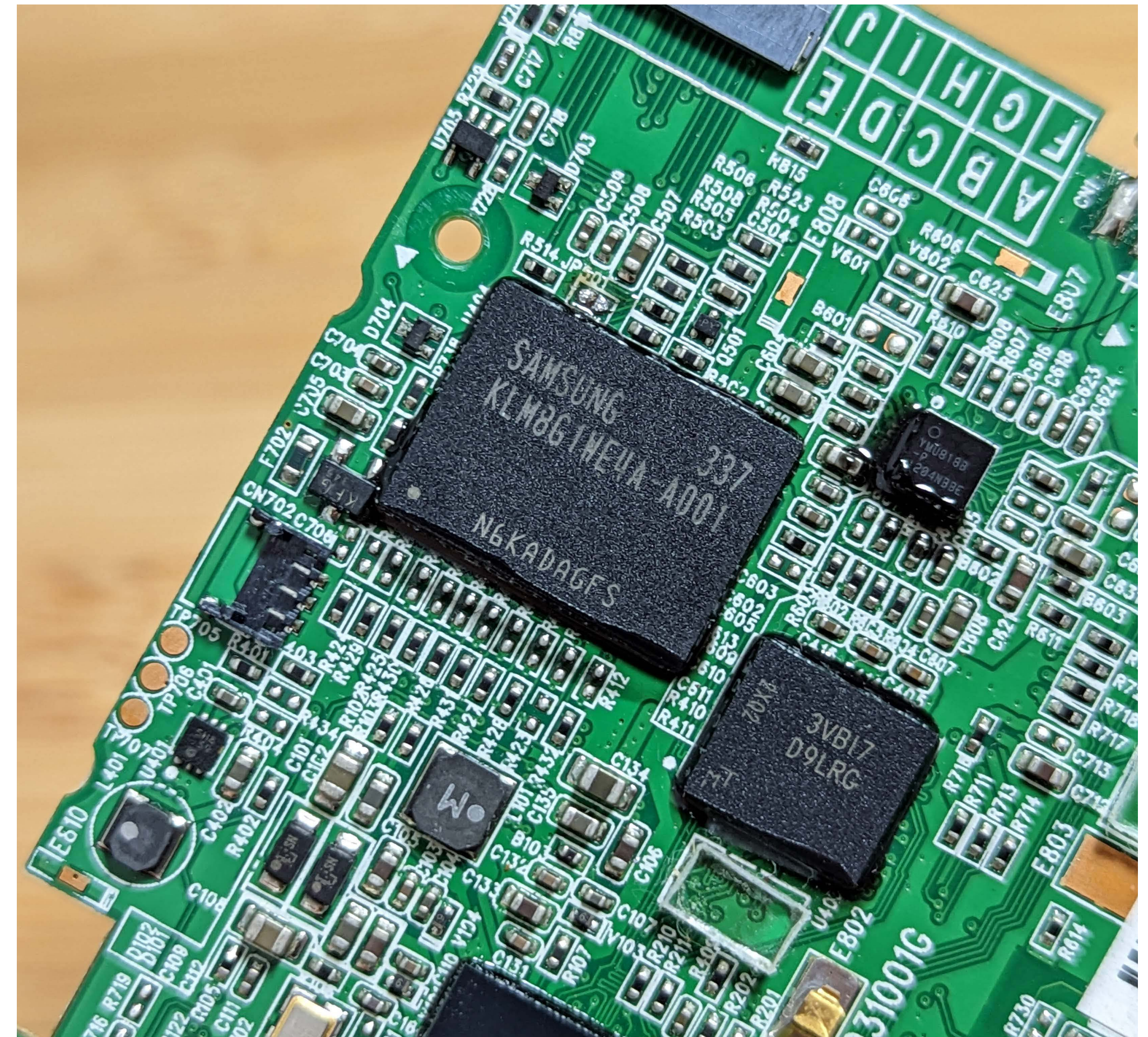


eMMC 向け OS イメージの構造

eMMC イメージを書き込んで起動する

eMMC の書き込み

- Linux 入り SD カードに忍ばせた eMMC イメージを実機で dd で焼くだけ
- バックアップを忘れずに！（電子辞書に未練がない限りは）
- 細かいはんだ付けができるなら書き換えで失敗しても USB boot + シリアル通信 (UART) で復旧可能




```
Last login: Fri Sep 23 04:13:49 JST 2022 on ttyAMA0
Linux brain 5.4.149-06270-g09cc34a56d01 #2 PREEMPT Fri Sep

The programs included with the Debian GNU/Linux system are
the exact distribution terms for each program are describe
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the
permitted by applicable law.
user@brain:~$
user@brain:~$ mount | grep mmcblk0
/dev/mmcblk0p3 on / type ext4 (rw,relatime)
user@brain:~$
```

`/dev/mmcblk0p3 on / type ext4`

`/dev/mmcblk0 = eMMC`



動いた！！！！

**こうして SHARP Brain は電子辞書であったことを忘れ
Linux マシンとして新たな一歩を踏み出すのであった**

まとめ



実装は SoC やアーキテクチャごとに異なっても
ブートローダーとブートシーケンスの存在はすべてのコンピューターに共通

お手持ちのハードの動作を根底から変えたいなら
はんだこてを握りつつブートシーケンスを掘り返していじってみましょう