

# SWEST28 EmbLT登壇者一覧

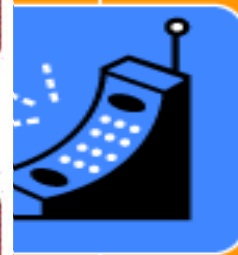
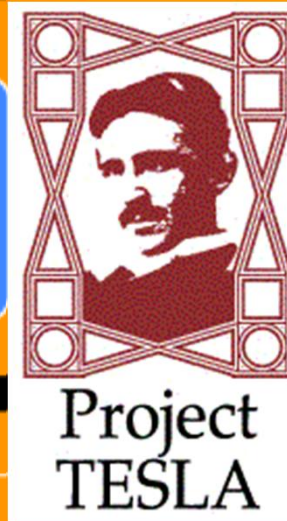
| No | 受付番号 | 氏名     | タイトル                          | カテゴリ                    | 公開 |
|----|------|--------|-------------------------------|-------------------------|----|
| 1  | 0083 | 山之上 卓  | テレポーテーションシステムその2              | テレポーテーションシステム           | ○  |
| 2  | 0022 | 冠者 徹也  | 「10年」をテーマに何か発表します             | EmbLT10周年！この10年を思い返してみる | —  |
| 3  | 0130 | 藤田 一槻  | 作り方を教えるゼミの、作り方ーラズピコWで模擬衛星開発   | ちょっとスゴイ組込み教材自慢          | ○  |
| 4  | 0053 | 畑 尚志   | USB入門 仕様の表記                   | 組込み百物語（本当にあった組込みの超怖い話）  | ○  |
| 5  | 0137 | 渡邊 登   | 組込みサロンで組込み開発RAGづくり（仮）         | 組込みへのAIと愛を語る！           | ○  |
| 6  | 0113 | 中井 祐一  | AIの手足を増やそう                    | 上手くつながった！/つながらなかった...話  | —  |
| 7  | 0048 | 井田 健太  | Dabaoボードであそんでみた               | 組込みへのAIと愛を語る！           | ○  |
| 8  | 0021 | 下山 吉洋  | この1年でやりたかったが全部解決した><          | AIでここまで作らせちゃいました        | ○  |
| 9  | 0107 | 本田 晋也  | 組込みへのAIと愛を深める会へのお誘い           | 組込みへのAIと愛を語る！           | ○  |
| 10 | 0007 | 山田 啓一郎 | Geneon-高周波回路の自動設計への挑戦-        | EmbLT10周年！この10年を思い返してみる | —  |
| 11 | 0040 | 長野 義史  | AIで設計が変わってる、私の周りの事例4つ         | AIで設計が変わってる、私の周りの事例4つ   | ○  |
| 12 | 0135 | 鈴木 雄一  | 数理AI科学者が肩骨折したら組み込み、半導体が楽しくなった | 組込みへのAIと愛を語る！           | ○  |

# テレポーテーションシステムその2

SWEST28-REG-0083 山之上 卓

情報処理学会フェロー, 福山大学名誉教授

公立大学法人福山市立大学大学改革推進アドバイザー





Project  
TESLA

- ・ まず演技(デモ)を見てください。



# 自己紹介



Project  
TESLA

- ・ 1980年、CQ出版社インターフェース6月号「LKIT-16によるFORTRANシステム」
- ・ 1980年に共立出版 bit 12月号のナノピコ教室解答「じゃんけん大会」で私のプログラムが優勝
- ・ 1988年、“パソコン通信を使い電算機言語を開発” 日本経済新聞,11月17日  
このとき、(現在名古屋大学の)高田 広章先生にお会いしています。
- ・ 2018年、Maker Faire Tokyo 2018、着る電光掲示板、声で操作するフォークリフトなど出展
- ・ 2023年、Maker Faire Tokyo 2023、シン3次元表示装置(トランジスタ技術賞受賞)、テレポートドレッサーなど出展
- ・ 2024年、Ogaki Mini Maker Faire 2024、テレポーテーションシステム(送受信)など出展



デモの中でやってること



Project  
TESLA

- ・ 設計図から物を作る
- ・ 物から設計図を自動的に作る
  - 向こうに設計図を送る
  - 設計図により同じ物を作る

## 我々のテレポーテーション (Practical Teleportation)



Project  
TESLA

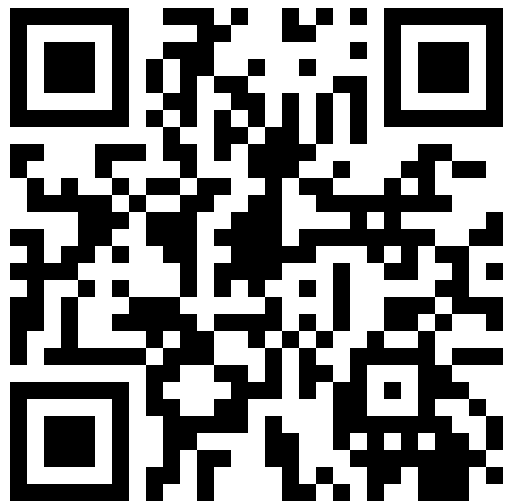
- ・ 前のスライドと同じことですが
- ・ 転送したい「物」の情報を入手。  
(テレポーテーションシステムその1では外からは見えない内部構造や機能も入手)
- ・ その情報を受信機に送る
- ・ (受信機側にある部品を使って)  
その情報から送信元と同じ「物」を作る。

# テレポーテーションシステム その1

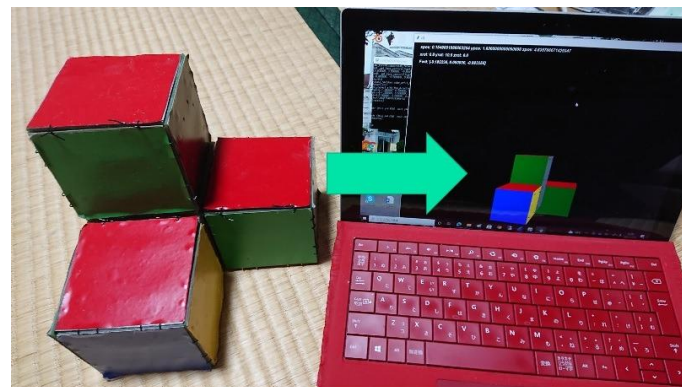


Project  
TESLA

- ・ テレポーテーションシステムその1も是非見てください。  
(昨年のインタラクティブセッションで送信部を紹介)



送信部

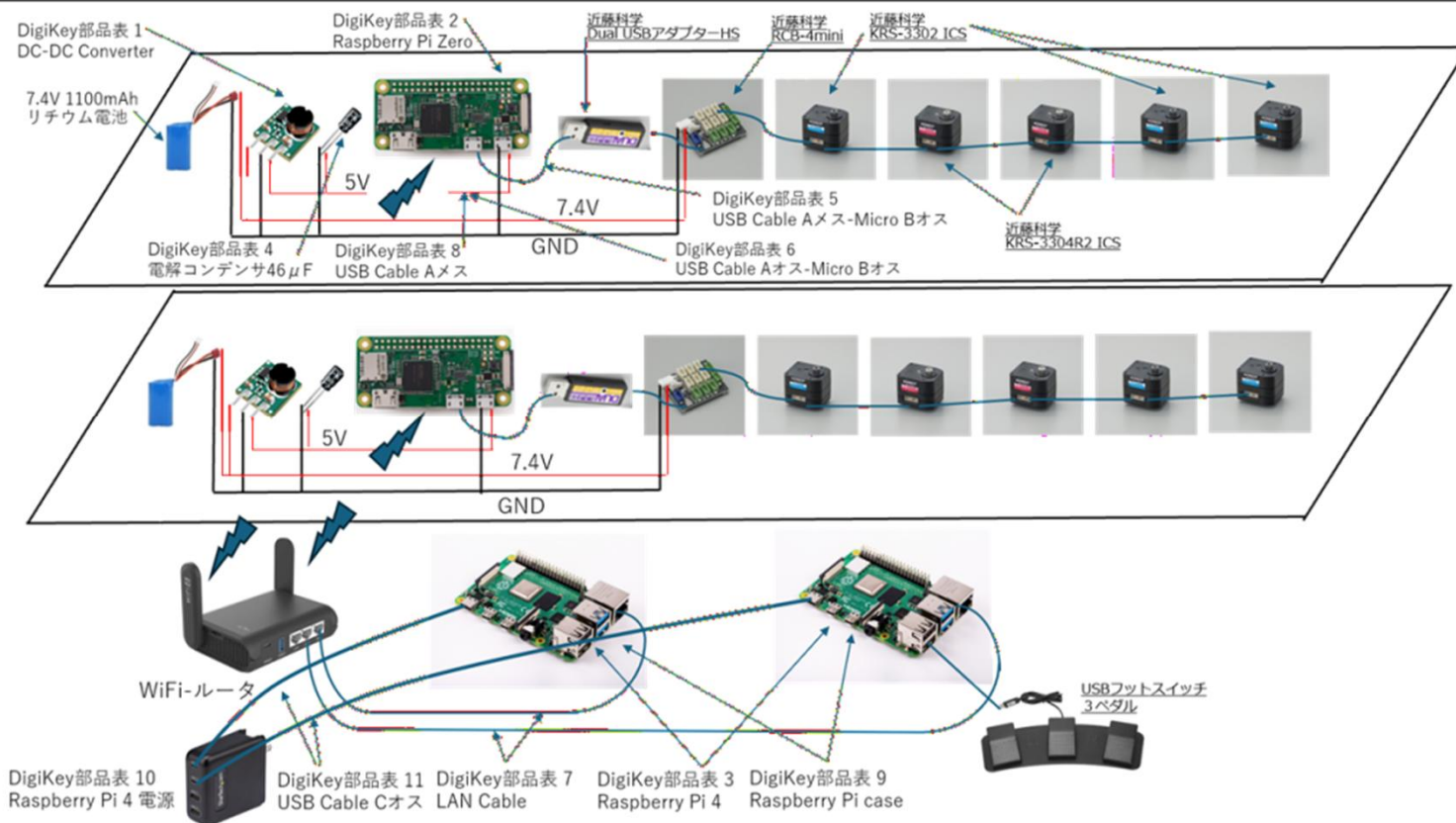


受信部

# テレポーテーションシステム その2



Project  
TESLA



## 今後の予定



Project  
TESLA

- ・ NT東京2026@科学技術館(東京) 10/3-10/4 出展予定
  - － テレポーテーションシステムその1の送信部も持っていきたいと思っています。(が、荷物が大変。不可能かも)
- ・ 2027年4月、福山市立大学情報工学部情報工学科誕生
  - － 2つのコース、そのうち一つは「デジタルものづくりコース」。モデルベース開発が中心。
  - － 私はもう一つの「サイバーセキュリティコース」の方。

## 謝辞



- What How For 北九州
  - （主査は元産業医科大学学長の東先生）



Project  
TESLA

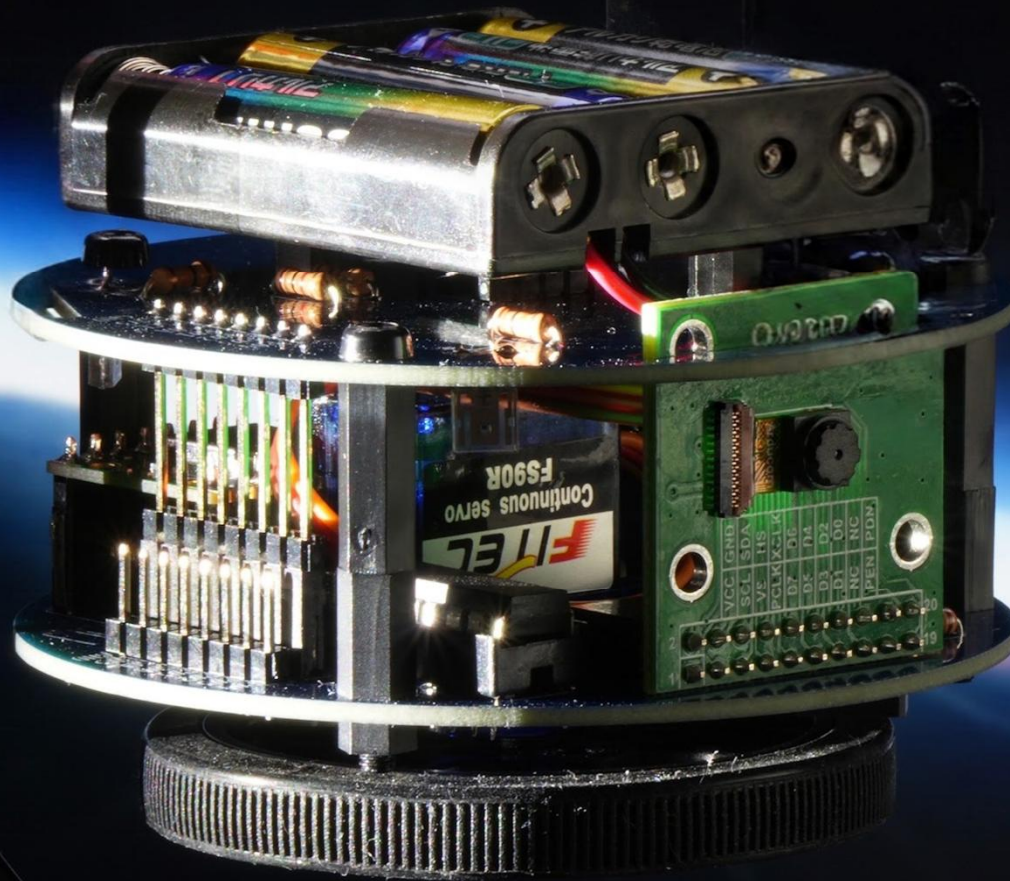
# 作り方を教えるゼミの、作り方

ラズピコWで模擬衛星開発

東京理科大学大学院工学研究科  
電気工学専攻修士2年

藤田一槻

@771\_8bit #SWEST28



# 自己紹介：藤田 一槻



## ■ 東京理科大学大学院 工学研究科電気工学専攻 M2

- ・ 最近はイメージセンサ(半導体/アナログ)の設計に転職気味
- ・ 個人でもMaker Faire Tokyo に自作イメージセンサを出展

## ■ 航空宇宙系の組み込み屋

- ・ ハイブリッドロケットサークル・鳥人間サークル
- ・ 「きぼう」ロボットプログラミング競技会
- ・ 現在は小型衛星ベンチャーで組み込みSWインターン

## ■ “エンジニアリング”を教えるゼミを企画

- ・ アストロキャンプ2026 衛星開発ゼミ

# モチベーション

---

## ■ 学生プロジェクトあるある



- ミッションに対し適切な設計ではなく、思いついたことを組み合わせるだけ
- 問題が起きると行き当たりばったりでデバッグ、明確な方針はない
- なんとなくガントチャートを書いてみるけど、マネジメントの役に立たない

**根拠を持たず雑に判断したり、判断の根拠が記録されなかったり…**

# エンジニアリングのやり方を教えた

---

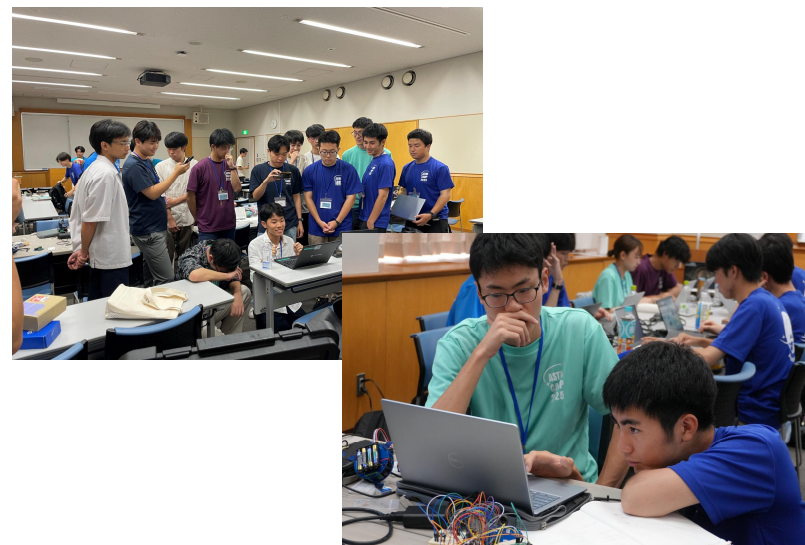
## ■“エンジニアリングのやり方”に向き合う時間がない

- ・ プロジェクトの目標達成が最優先 → やり方を学ぶ機会がない

## ■そもそも学ぶ場が実はない

- ・ 工学的な知識は大学で勉強するし、プロジェクトでモノも作るけど、モノを作る方法を「学ぶ」場はあんまり無い

じゃあゼミでやってみよう！



# ゼミの題材を考える前に…

## ■ 宇宙系の教材といえば CanSat

- ・ 空き缶 (Can) サイズの模擬人工衛星 (Satellite)

## ■ オリジナリティを出したいが、

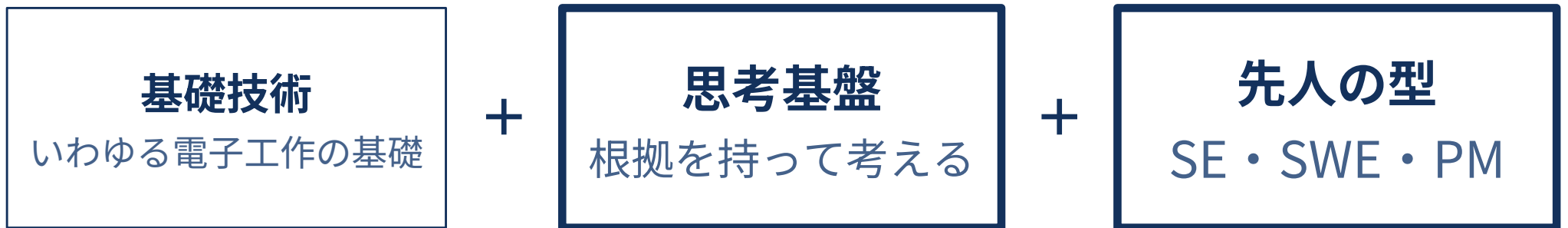
- ・ データを取得するだけの箱だとなつまらない
- ・ ローバータイプだと走る環境がない

そもそも教材として要件はなんだろう…？



# 技術の基礎より、考え方と型を教えたい

---



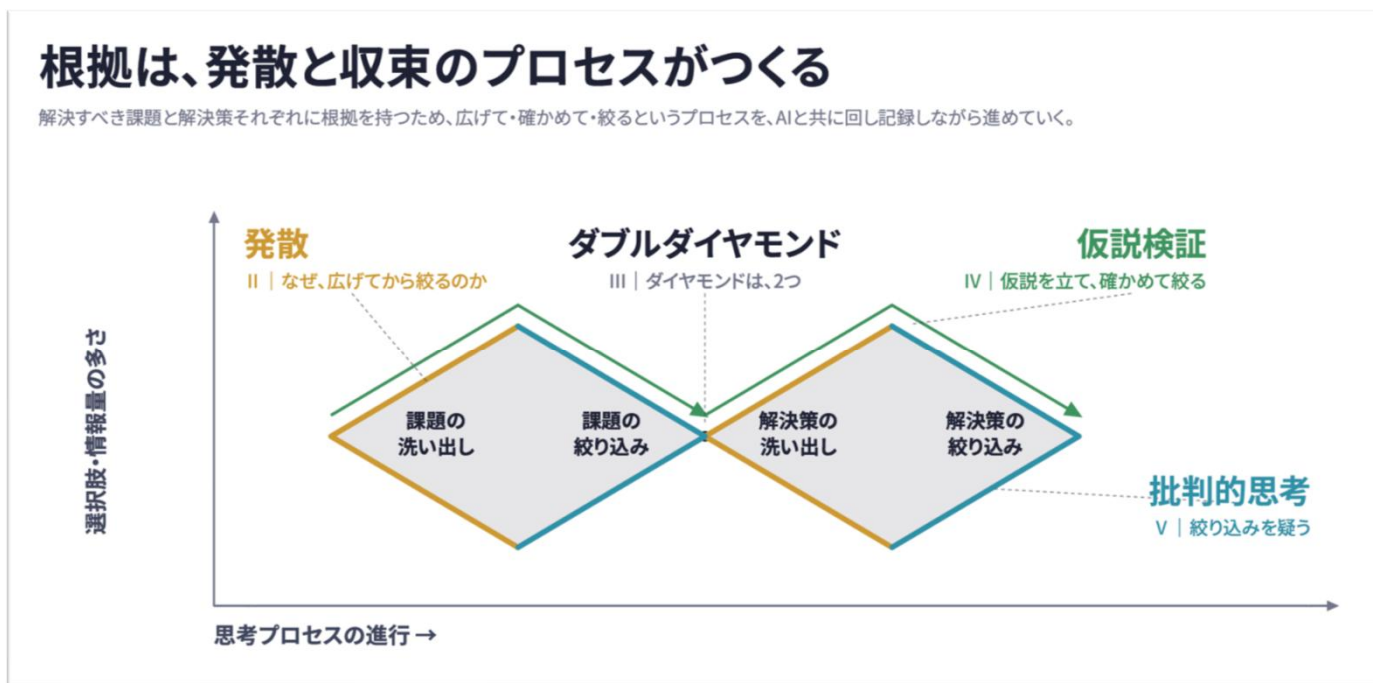
## ■ 根本的な思想と先人の型の両方が必要

- 根本的な思想から、全てを判断するのは厳しい  
→ システムズエンジニアリング / ソフトウェアエンジニアリング  
/ プロジェクトマネジメントといった体系知を教える
- 型を機械的に回すだけでは意味がない

# 根拠を持って考える

## ■ 根本思想は共通講義で回収

- ・ 抽象化するとデザイン思考のダブルダイヤモンドが有効という仮説



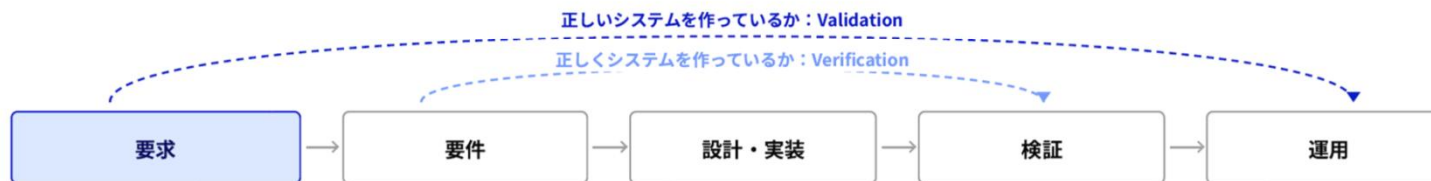
# 先人の型：システムズエンジニアリング

## ■ 要求・制約・要件・設計・実装・検証・運用までを根拠を持って繋ぐ よくあるV字モデルも、そのモチベーションから伝える

### 3.1 トレーサビリティ

#### 設計・実装したものと、「やりたいこと」との整合を確認する

要求・要件にもとづき設計・実装した後、本当にその通りできているか確認する。  
開発のプロセス全てが根拠で繋がっていること（トレーサビリティ）が重要。

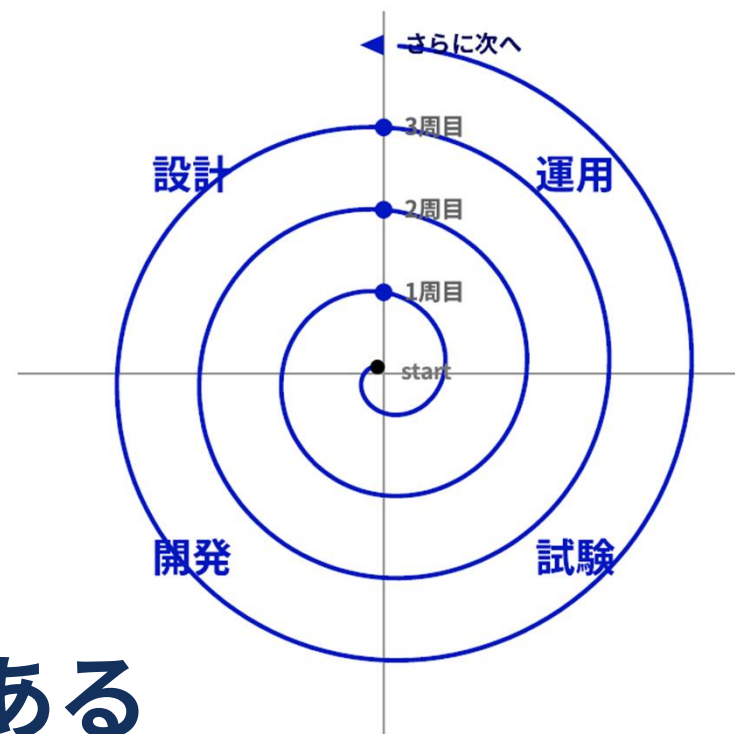


- 要求に合っているか：正しいシステムを作っているか  
要求＝システムを使ってユーザーがしたいこと
- 要件に合っているか：正しくシステムを作っているか  
要件＝システムができること

# 実際は：やってみないと分からない

■ とりあえずやってみて、unknown unknownを潰す型がある

|                   |                  |
|-------------------|------------------|
| 知っていることを<br>知っている | 知っていることを<br>知らない |
| 知らないことを<br>知っている  | 知らないことを<br>知らない  |



先人の型には、  
それが対処する“難しさ”がある

# 宇宙開発の難しさはどう現れるか？

## ■ 極限環境（熱・放射線・振動）

- ・ 設計は当然大変、試験も面倒

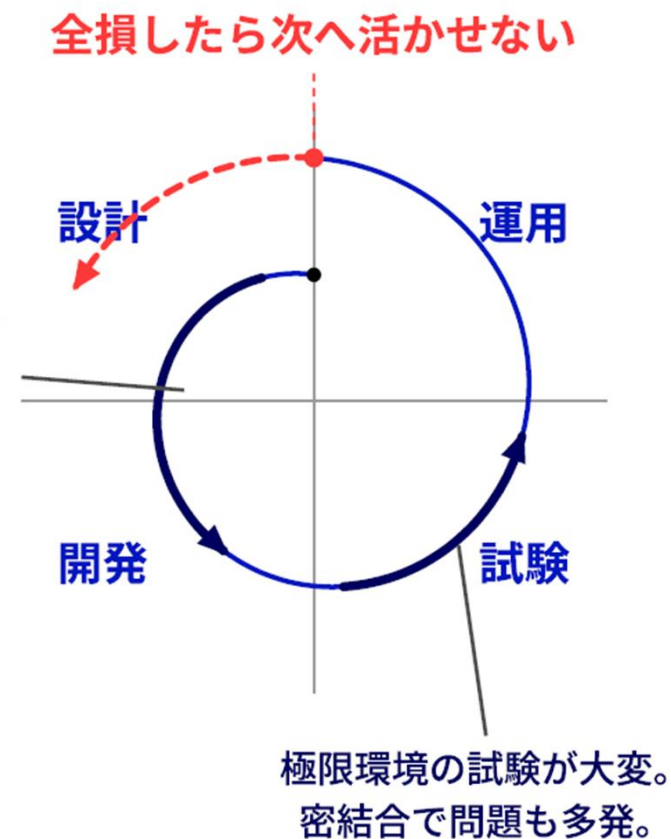
## ■ 密結合な複雑系

- ・ 宇宙機として単体で動作するため、多くのサブシステムがある集まる複雑系
- ・ さらに依存関係が根深い密結合で設計も試験も大変

耐環境・複雑性・信頼性を踏まえ  
ちゃんと考えるのが大変。

## ■ 電波のみでのアクセス（非修理系）

- ・ 電波で通信ができないところまで壊れたら終わり



# ゼミの題材に持ち込みたい“難しさ”

## ■ エンジニアリングの型を教えるには、その根拠となる“難しさ”が必要

- ・ 型にも根拠がなければ、学ぶ方は納得しないし身につかない
- ・ 宇宙開発の難しさを見つめ、それを衛星開発ゼミに持ち込む

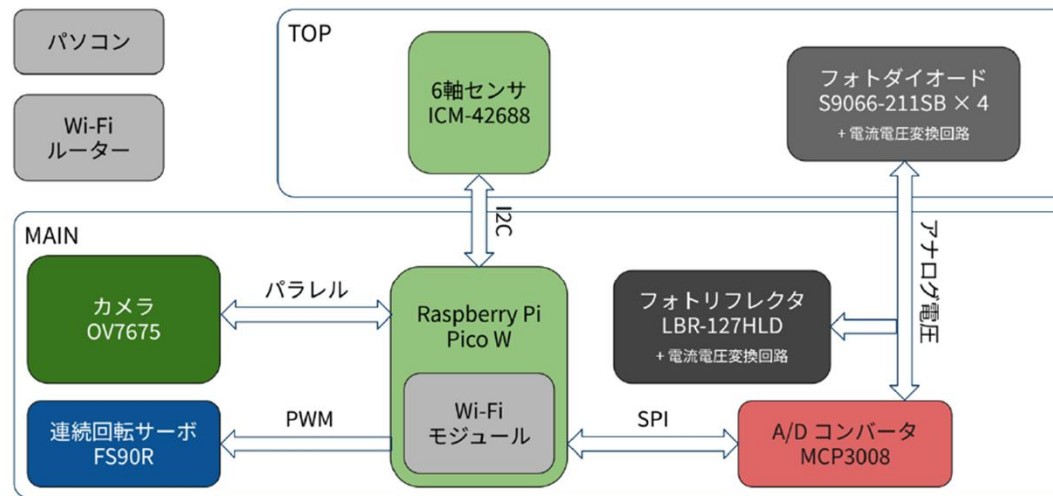
| 宇宙の特徴         | 題材での再現            |
|---------------|-------------------|
| 極限環境          | ✗ 地上なので環境は諦める     |
| 密結合な複雑系       | ✓ エレキ/メカ/ソフトを絡ませる |
| 電波だけが頼り（非修理系） | ✓ 運用ルールで再現する      |

# ゼミの題材：模擬衛星の設計

## ■ リアクションホイールを搭載し、姿勢制御を行う

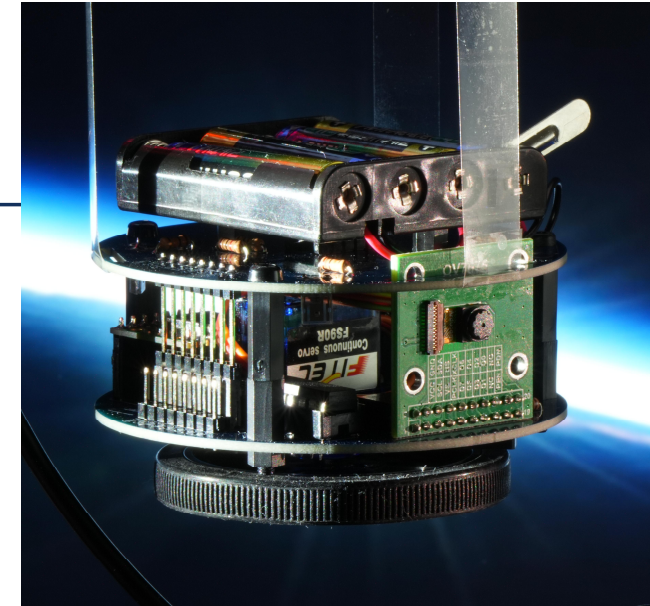
- ・アクチュエータは欲しいけど、卓上で動かしたい
- ・（角運動量保存を使って回転させるヤツ）

## ■ 実際の衛星のサブシステムを模擬する



サブシステムが1枚の基板にぶら下がる（TOP／MAIN 基板）

- **C&DH：Pico W（RP2040）**  
全体を統括する計算機（OBC）
- **ADCS：ジャイロ＋太陽センサ＋サーボ**  
ICM-42688（I2C）／フォトダイオード×4（ADC）  
／サーボ＋タイヤ（PWM）
- **COMM：マイコン内蔵 Wi-Fi**  
PC とテレメトリ・コマンドを送受信
- **Payload：カメラ**  
OV7675（パラレル＋I2C）



# 模擬衛星のミッションと制約

**ミッション：太陽との相対位置が既知の天体の画像を取得する**

想定シーケンス：姿勢推定 → 姿勢制御 → カメラ撮影 → データ送信

## ■ 制約：電波だけでアクセス

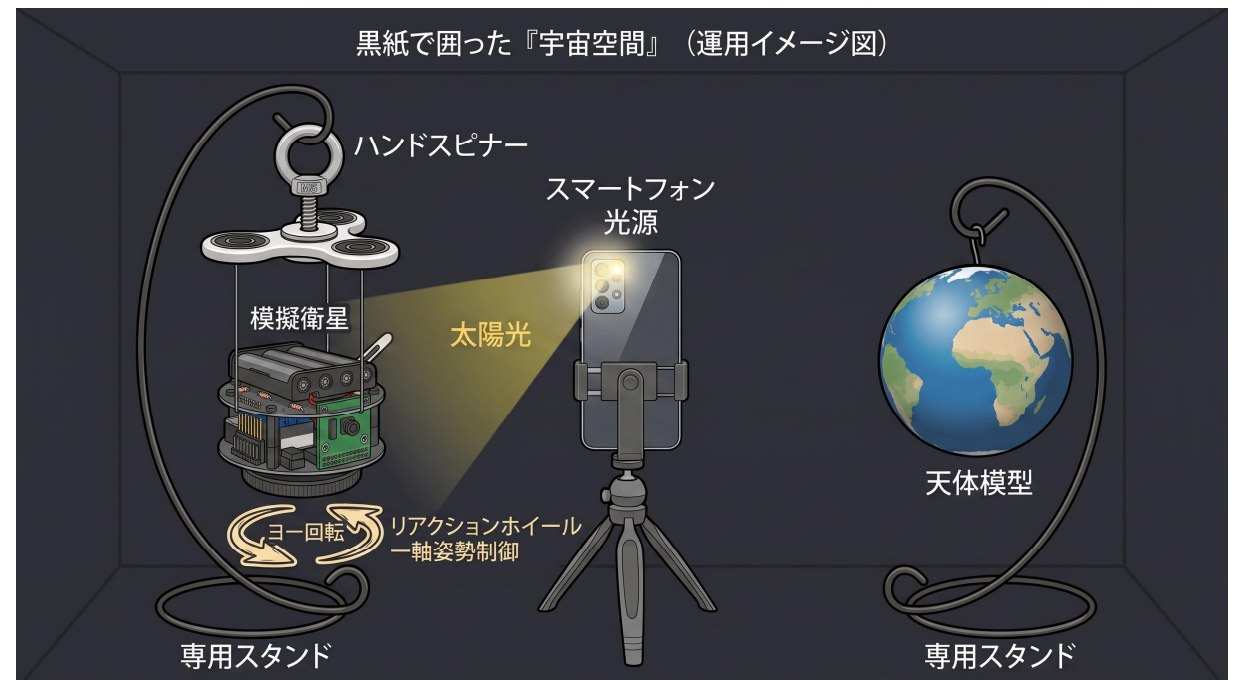
目視・物理的な介入は禁止  
宇宙機の難しさを再現

## ■ 制約：地上での制御の禁止

“宇宙”と“地上”の差別化  
MBDのモチベーションに

## ■ 許可：打上げ回数は無制限

SpaceXのようなサイクルを前提

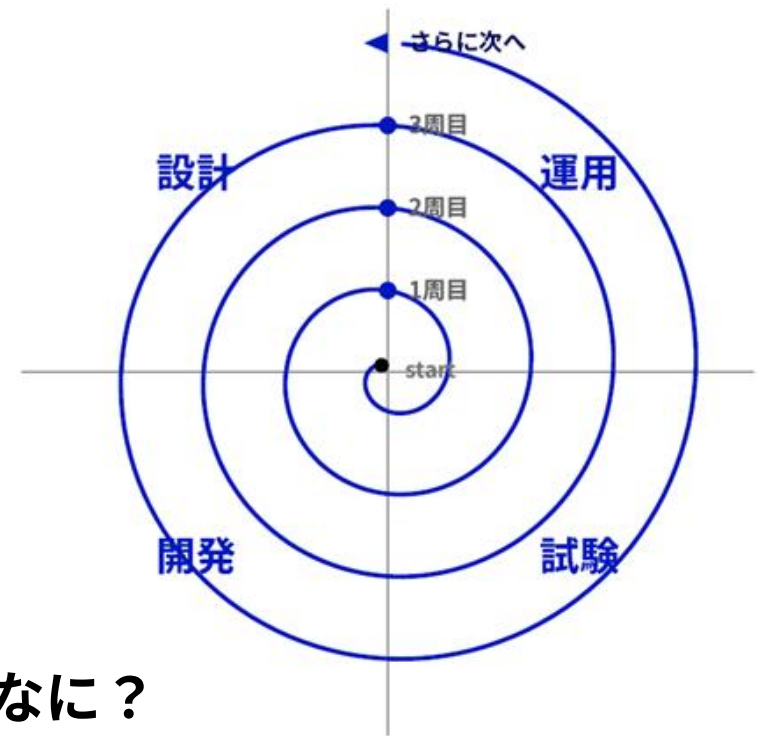


# 全てに根拠が必要になる

制約：電波だけでアクセス  
制約：地上での制御の禁止  
許可：打上げ回数は無制限



- 1回の打上げで何を確かめる？
- 打上げ前にどんな準備が必要？
- そもそも打上げないと確かめられないことはなに？



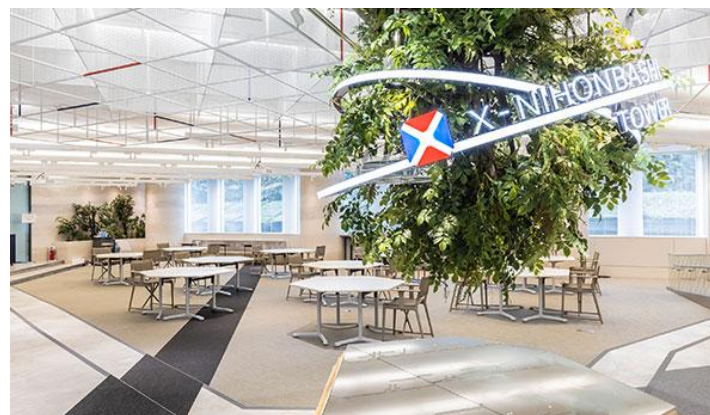
# しかし、まだ仮説

## アストロキャンプ2026は2週間後に開催！

- ・伝えるべきエンジニアリングのやり方は正しかったか？
- ・そのモチベーションとなる“難しさ”は正しく見抜けていたか？
- ・それを適切な方法で伝えられたか？



昨年2025の様子



**最終日 9/12(土)は  
一般公開イベント！  
@東京 X-NIHONBASHI**

# まとめ：作り方を教えるゼミの、作り方

---

モチベーション

**建前：根拠を持ったエンジニアリングを学ぶ場を作りたい**

**本音：根拠を持ったエンジニアリングを言語化したい**

やろうとしたこと

**思考基盤と先人の型を考えたい**

結局やったこと

**エンジニアリングの本質的な難しさを考えることが必要だった**

# 作り方を、つくりかえる

---

AIで作り方は変わる

でもエンジニアリングの難しさは多分そんなに変わらない

本質的に何が難しかったのか？を見つめ直す

それをAIで倒すやり方をつくる

---

771-8bit.com

藤田一槻 @771\_8bit



---

---

# 怪しいUSB入門

仕様表記の用語解説

---

---

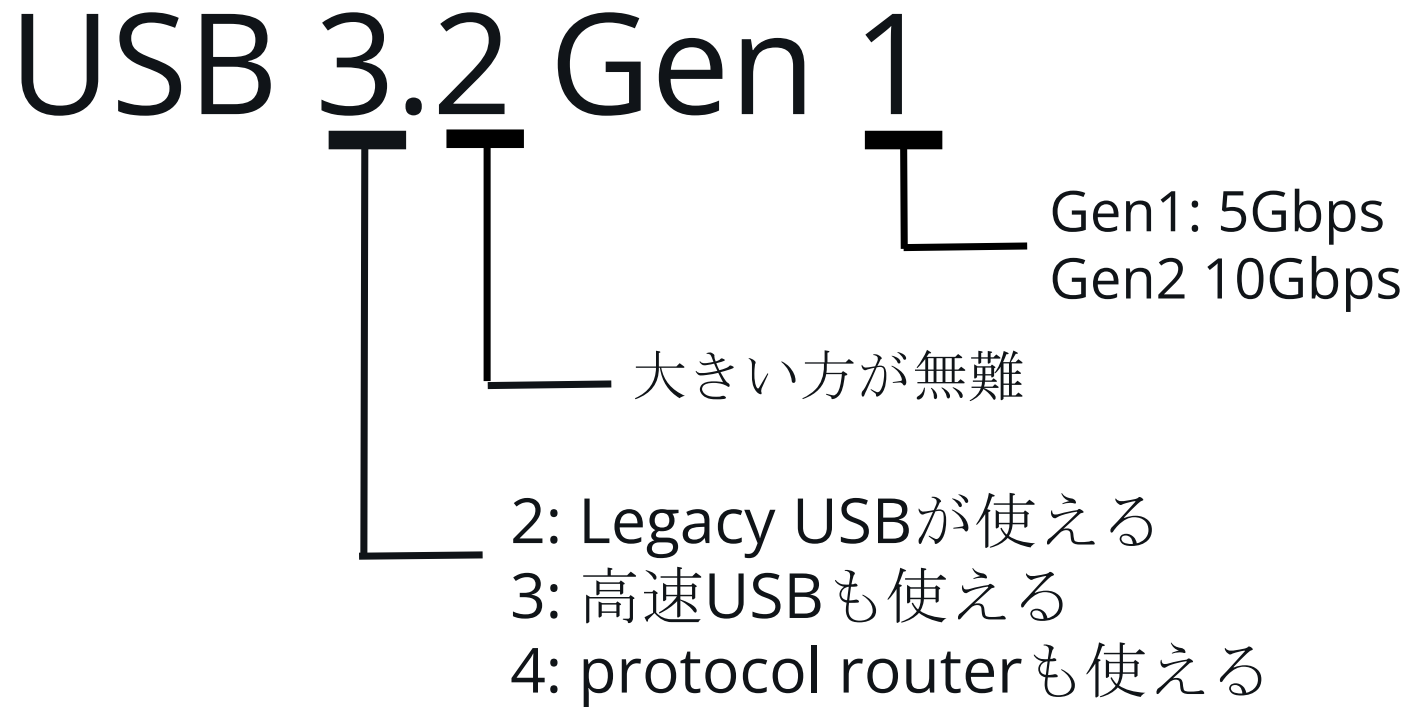
# USBの仕様表記わからない問題

| 規格名 (旧名称)                                     | 仕様発行<br>日    | 最大データ転送速度<br>(実効速度)       | 最大伝送距離                                        |
|-----------------------------------------------|--------------|---------------------------|-----------------------------------------------|
| USB 1.0                                       | 1996年1<br>月  | 半二重12 Mbps<br>(1.5 MB/s)  | 5 m<br>ただしUSB Type-<br>Cは4 mまで                |
| USB 1.1                                       | 1998年9<br>月  |                           |                                               |
| USB 2.0                                       | 2000年4<br>月  | 半二重480 Mbps<br>(60 MB/s)  |                                               |
| USB 3.2 Gen 1<br>(USB 3.1 Gen 1)<br>(USB 3.0) | 2008年<br>11月 | 全二重5 Gbps<br>(500 MB/s)   | 3 m<br>ただしUSB Type-<br>Cは2 mまで <sup>[3]</sup> |
| USB 3.2 Gen 2<br>(USB 3.1 Gen 2)              | 2013年8<br>月  | 全二重10 Gbps<br>(1.21 GB/s) |                                               |

わからない理由：  
業界常識が共有できてない

wikipedia : ユニバーサル・シリアル・バスより引用 2026-08-23

# まとめ



# Gen 1, Gen 2の定義

|       |                                                                                                                                                                             |
|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Gen 1 | Gen 1 is an adjective used to refer to the Physical layer associated with a 5.0 Gbps signaling rate. The original USB SuperSpeed Phy and a Gen 1 Phy refer to the same Phy. |
| Gen 2 | Gen 2 is an adjective used to refer to the Physical layer associated with a 10 Gbps signaling rate.                                                                         |

Universal Serial Bus 3.2 Specification  
2. Terms and Abbreviations

USB 3規格書の用語表に記載

とりあえず物理層の名称でOK

Gen1 : 5Gbps, Gen2 : 10Gbps

物理層: アナログ値(電圧, 電流, 時間)を扱う回路

## 他規格のGen X表記

|       | SATA | PCIe *1 | USB 3 | USB 4 |
|-------|------|---------|-------|-------|
| Gen 1 | 1.5G | 2.5G    | 5G    |       |
| Gen 2 | 3G   | 5G      | 10G   | 10G   |
| Gen 3 | 6G   | 8G      |       | 20G   |
| Gen 4 |      | 16G     |       | 40G   |
| Gen 5 |      | 32G     |       |       |
| Gen 6 |      | 64G     |       |       |

\*1: PCIeは俗称(規格書に定義が無い)

# バージョン表記

- 規格(仕様)のバージョン表記
  - 規格団体が管理
  - 管理団体によって運用が異なる
- USBのバージョン管理
  - マイナー番号：仕様改定の管理
  - メジャー番号：2.0以降は仕様の種類

# USB3のマイナーバージョン表記

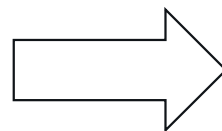
USB 3の仕様

仕様  
(規格団体)

コントローラ:  
フレームが ....

物理層:  
5GHzで通信..

改定  
Bug fix  
機能追加



コントローラ:  
フレームが ....

物理層:  
Gen1: 5GHzで通信..  
Gen2: 10GHzで通信 (Option)

USB3の仕様(初版)  
Ver 3.0

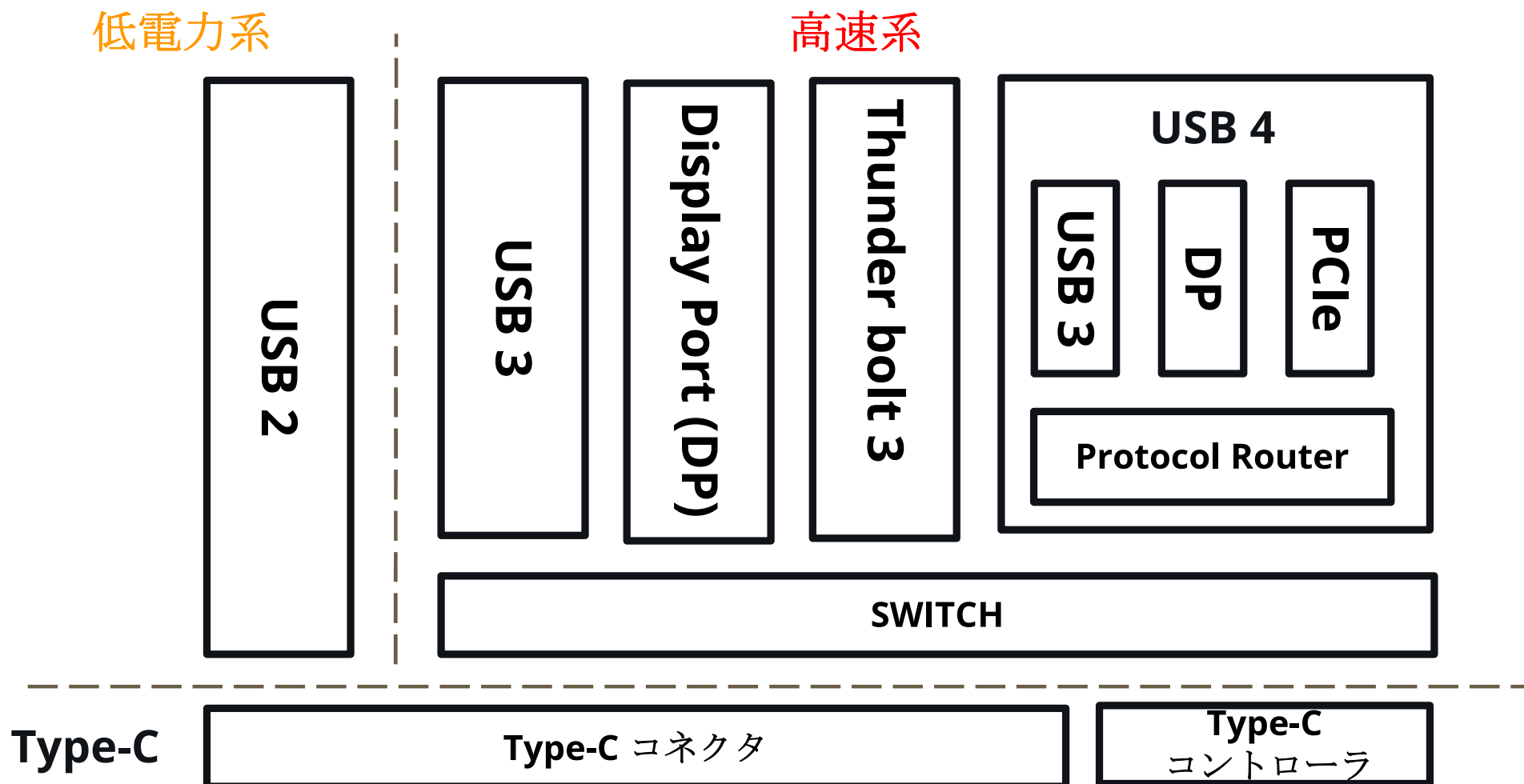
USB3の仕様(改訂版)  
Ver 3.1

実装の表記  
(メーカー)

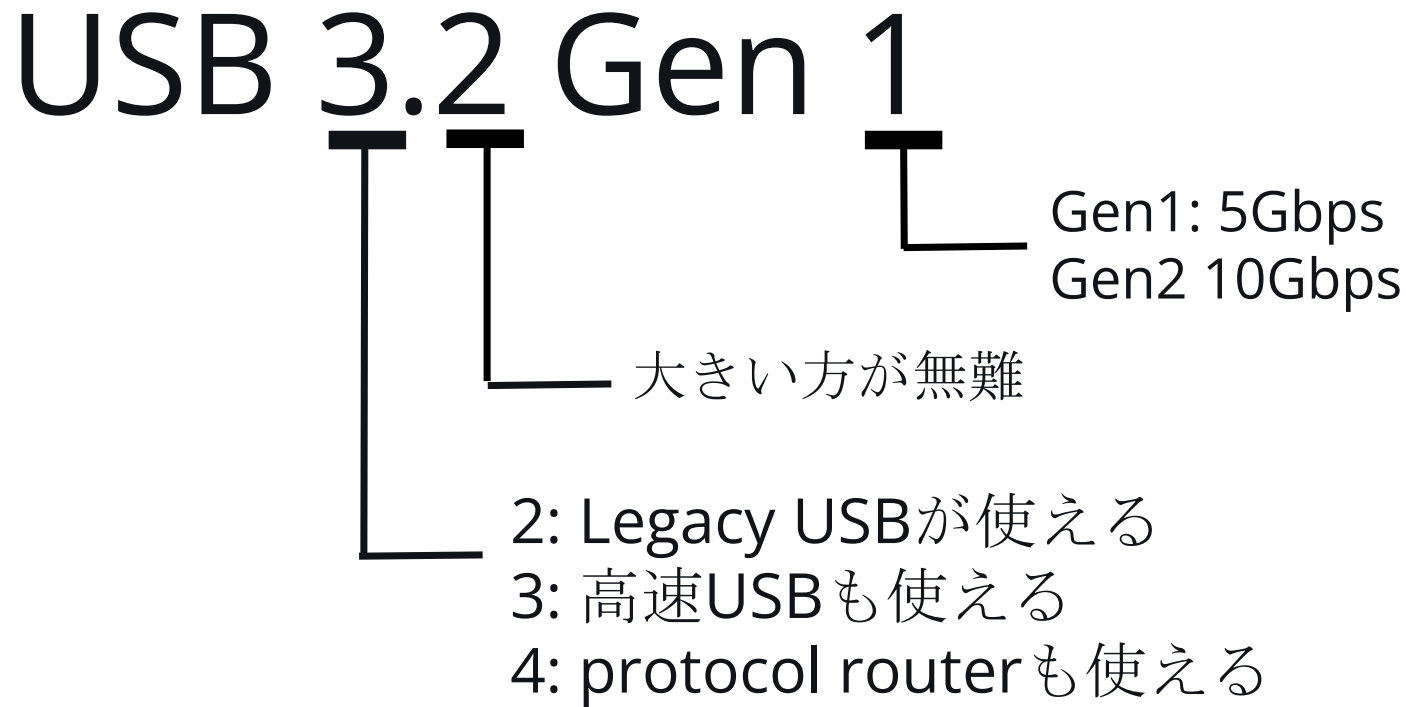
USB3.0

USB3.1 Gen1  
Gen2

# メジャー番号: Type-Cのインターフェースの例

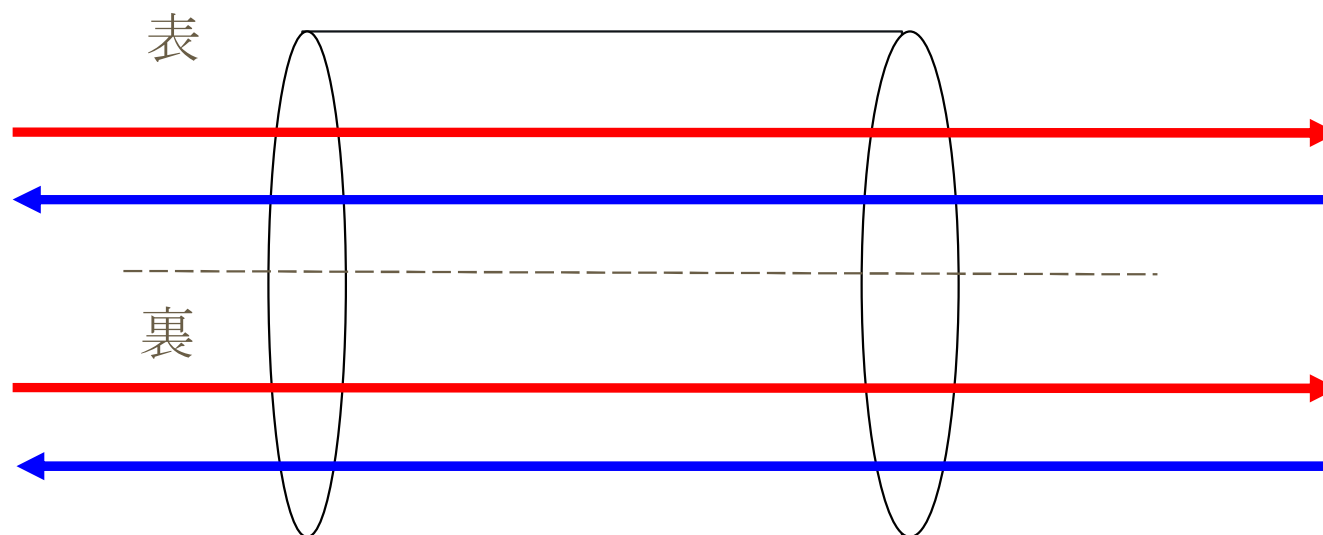


# まとめ

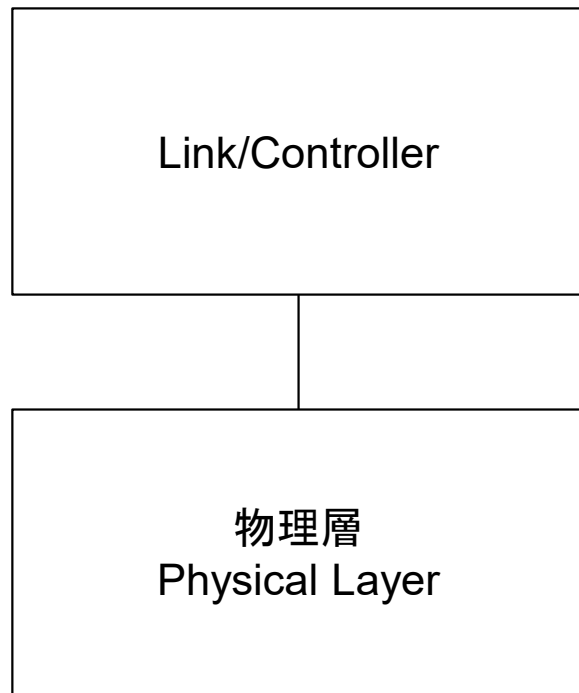


# Alt modeの割り当て

USB3  
USB4  
ThunderBolt



# インターフェース回路の構成



- リンク/コントローラ

通信のプロトコル管理を行う

フレームフォーマット

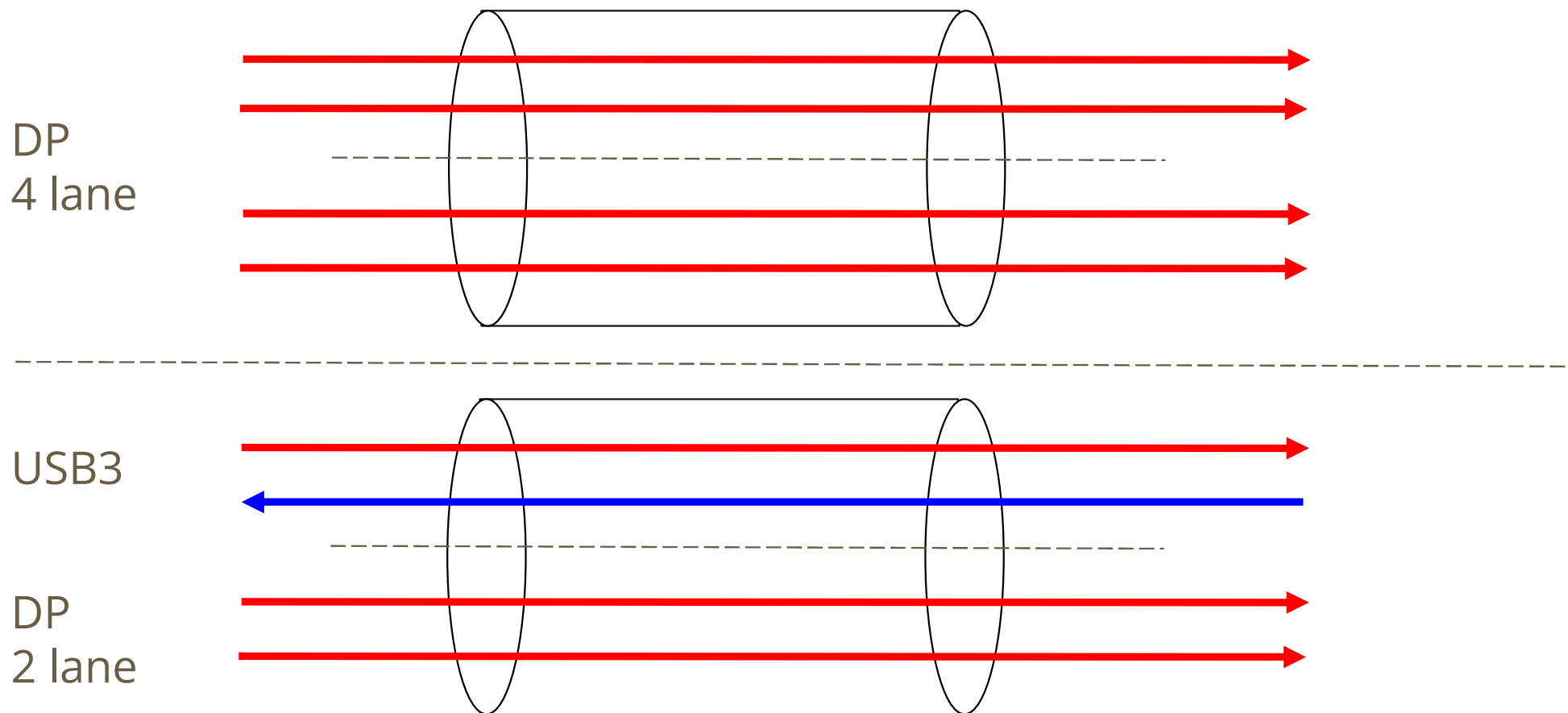
ハンドシェイク

- 物理層

USBでは電気特性に関する仕様を定義

電圧, 電流, 時間(周波数)...

# Alt DPの構成



組み込み開発の相談相手

# 『EmbMentorRAG』の開発

Your AI Mentor for Embedded Software Development

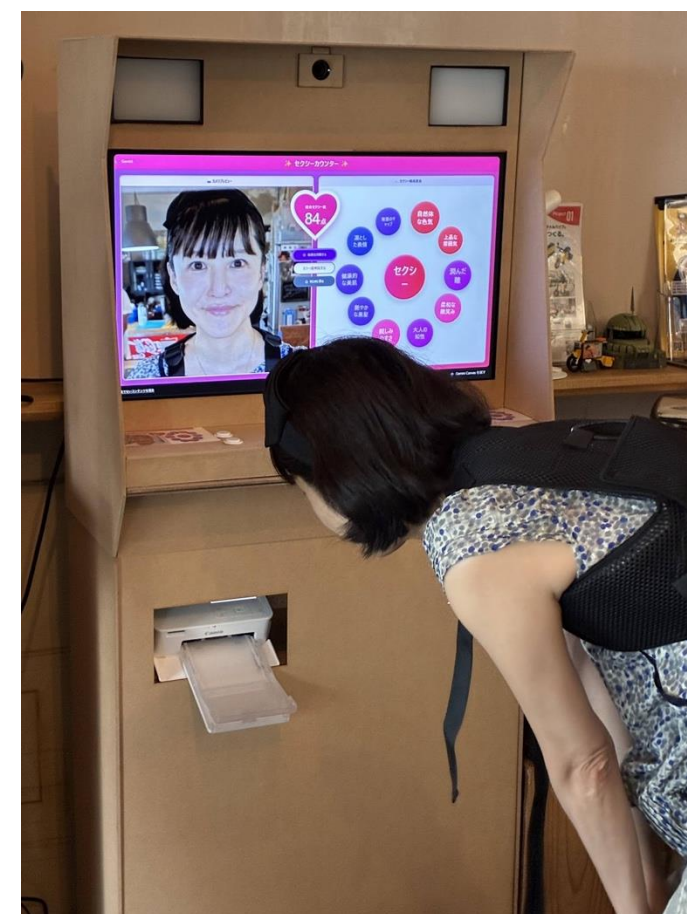
組み込み開発サロン：渡邊登

# 自己紹介：渡邊登

重版を重ね13刷！

9月で7周年！

AIゲームセンター構想  
クラファン700万円！



# SESSAMEとは



2000年から活動する**組込みソフト開発コミュニティ**

組込み技術、開発手法、プロセス、マネジメント、人材育成の知恵を共有

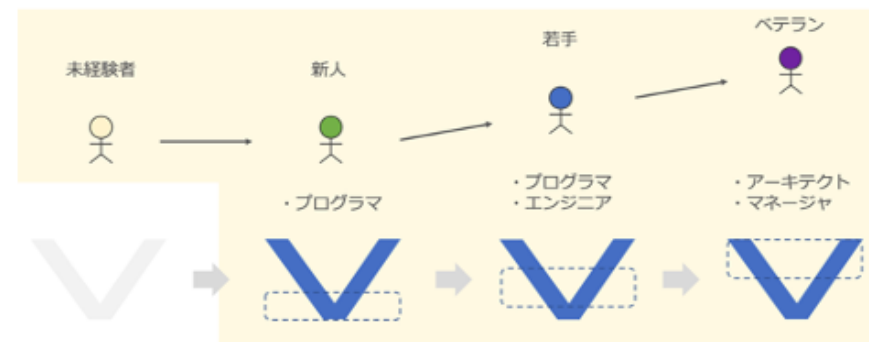


eラーニング、書籍、セミナー、無償PDFなどでOUTPUT

開発現場で求める  
**中級エンジニア**  
を育てる知恵を共有！

未経験者を対象にした人材育成

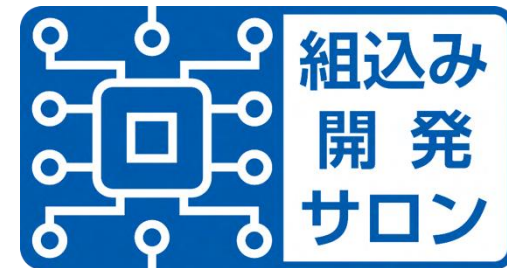
組込み特有の技術の人材育成



初級卒業を目指す人材育成

モデリングやテスト手法の人材育成

人材育成手法に関する人材育成



# 組み込み開発サロンとは

- 組み込み開発に関するSlackベースのオンラインサロン
  - 会費は月1千円
  - 2026.7月時点で約80名が参加

組み込み開発者の3rdプレイスの場として使ってほしい  
Slack & 月例の研究会（第3火曜日19:00-）ON/OFFLINE

- SESSAMEコンテンツ利用可能
  - 合計約1,400ページ公開中
- 会員限定のセミナーや動画コンテンツなど提供

これらの情報をベースに組み込み開発の知恵AIをつくりたい  
◆教材コンテンツ ◆研究会・セミナー動画コンテンツ

# 組込み開発研究会とは

月例の研究会で、オンラインとリアルで開催

- SESSAMEが開催していた定例研究会を引き継ぎ、現在Vol.223
- SESSAMEでは、各プロジェクトの進捗などを共有
  - NPO法人解散によって、プロジェクト活動は一旦リセット
  - 設計モデリング、子供向け宇宙講義、SysML2など活動は継続
- 新スタイルになって9回開催
- リアル会場は浅草橋の「技研ベース」。ビールを片手に・・・
- SESSAMEでの経験・・・
  - 実は重要な話はテキストに書いていない、**講師の与太話**にこそ含まれている

組込み開発の知恵AI

# 背景と課題：一般AI活用の限界

- **ハルシネーションと根拠欠如**

- 一般的なWeb検索型の生成AIに質問した場合、一見正しそうな嘘（ハルシネーション）を出力するリスクがある。
- 回答の「出典・リファレンス」が不明確なため、極めて高い安全性と正確性が求められる組込み開発現場での直接採用は困難。

- **ドメイン知識・現場知の非連携**

- 企業固有の製品仕様や工場・現場でのノウハウ（「知」）は、インターネット上の公開データには存在しない。
- 汎用AIモデルでは組込み特有のリアルタイム制約、メモリ制約、業界標準規約を考慮した回答を得るのは困難。

# RAGによる信頼性の担保

- **正確な「知識（知）」の外部検索**
  - 学習データのみに頼らず、検証済みのドキュメントデータベースから関連情報をリアルタイム抽出。
- **明確なリファレンス提示**
  - 回答とともに根拠となった仕様書・標準教科書等の参照ページを同時に表示。
- **現場・標準知識の融合**
  - 自社・企業グループ内の知恵と、業界標準知識を組み合わせ、最適解を生成。

# NTTデータ先端技術とNTTデータMSE、知識継承で組み込み開発プロセスを変革する生成AIソリューションを開発

ニュース 2026.04.07

株式会社NTTデータ先端技術（本社）と株式会社NTTデータMSE（本社）は、AIエージェント活用により組み込み開発しました。

本ソリューションでは、業界標準、由で共有し設計・実装・レビュー・す。これにより、高度専門技術者のし、技術者のスキルアップの加速や

最新の生成AI領域における技術と暗黙知の形式知化のノウハウを持つNTTデータ先端技術と、組み込みソフトウェア領域において企画・設計から実装・検証までを一貫して担ってきた開発力と現場知見を持つNTTデータMSEのそれぞれの知見を組み合わせることで、お客さまの業務プロセスを深く理解したAIエージェントを構築し、開発プロセス変革を支援します。



# EmbMentorRAGでやりたいこと

- コミュニティベースのRAGをつくり共有
  - オープンが理想だがまずはコミュニティクローズで考える
- **SESSAE秘伝の教材コンテンツを利用** 
- IPAや業界団体などの公的コンテンツを利用
- **組込み開発サロンの定例研究会の話を利用** 
- まずはやってみる！それから考える！

# RAG検索対象：SESSAME＜約1,400頁＞

- 初心者 ◆C言語基礎 （187ページ） ◆C言語応用 （127ページ）
- 基礎 ◆SESSAME基礎コーステキスト （236ページ）
  - ・組込みプログラミング工程と品質の基礎知識 ・組込みプログラミングハードウェア基礎知識 ・Cプログラミング基礎I ・Cプログラミング基礎II
  - ◆マイコンプログラミング (205ページ) ・e2studioでの開発 (39ページ) ◆マイコンデバッグ技法 (31ページ)
  - ◆SESSAME寄付講座「モータを回す」 (40ページ) ◆SESSAME寄付講座「時間を扱う」 (48ページ)
- 初級 ◆SESSAME初級テキスト （248ページ）
  - ・開発課題と失敗事例の解説 ・組込み向け構造化分析の概要・組込み向け構造化設計 ・組込みプログラミング基礎知識 ・コードレビューの概要 ・ソフトウェアテストの概要
  - ◆セキュアな組込み製品を作る
    - ・1.概念編 （53ページ） ・2.要素技術編 （52ページ）・3.プログラミング・テスト編 （45ページ） ・4.開発技術編 （45ページ）
- 中級 ◆SESSAMEリーダー向けテキスト（管理技術） （318ページ）
  - 管理の方法と開発管理者への期待 ・プロジェクト計画立案・運用とリスク管理の概要 ・ソフトウェアの品質特性 ・インスペクションのポイント・実践！ 構成管理 ・組込みソフトの品質管理
  - ◆リスク管理を通したプロジェクト力の向上 （63ページ）
  - ◆見積り技術向上 （59ページ）
  - ◆SESSAMEリーダー向けテキスト（開発技術） （120ページ）
    - ・分析の大切さ ・構造化手法による上流工程の概要 ・組込み開発における ソフトウェアテスト
  - ◆デバイスドライバの作成作法 （41ページ）
  - ◆組込みソフトウェア開発におけるレビュー入門 （56ページ） ～仕様書及びソフトウェア開発計画の欠陥抽出ポイントと実践に必要な考え方～
  - ◆ソフトウェアプロダクトライン開発入門 （94ページ） ～再利用性の高いクラス設計～
- 組織／チーム運用
  - ◆ハードウェア出身のマネージャに知ってほしい7つのこと （18ページ） ◆リスク管理を通したプロジェクト力の向上 （63ページ）

# RAG検索対象：IPA

- つながる世界の開発指針～安全安心なIoTの実現に向けて開発者に認識してほしい重要ポイント～ 第2版
- 組込みソフトウェア開発における品質向上の勧め [ユーザビリティ編]
- 組込みソフトウェア開発における品質向上の勧め [設計モデリング編]
- ソフトウェアテスト見積りガイドブック
- 高信頼化ソフトウェアのための開発手法ガイドブック
- ESMR Ver.1.0：組込みソフトウェア向けプロジェクトマネジメントガイド [計画書編]
- ESQR Ver.1.1：【改訂版】組込みソフトウェア開発向け品質作り込みガイド
- 組込みソフトウェア向け設計ガイド ESDR[事例編]
- 組込みソフトウェア開発における品質向上の勧め[バグ管理手法編]
- つながる世界の品質確保に向けた手引き～IoT開発・運用における妥当性確認・検証の重要ポイント～
- ESCR C++ Ver. 2.0：【改訂版】組込みソフトウェア開発向けコーディング作法ガイド [C++言語版] Ver. 2.0
- ESCR Ver. 3.0：【改訂版】組込みソフトウェア開発向け コーディング作法ガイド [C言語版] ESCR Ver.3.0
- つながる世界のセーフティ&セキュリティ設計入門～IoT時代のシステム開発『見える化』～
- 「つながる世界の開発指針」の実践に向けた手引き [IoT高信頼化機能編]
- 非機能要求グレード2018
- IoT開発におけるセキュリティ設計の手引き
- 非機能要求グレード2018 利用ガイド [解説編]

# RAG検索対象：その他

- ENISA：IoTのセキュリティ標準のギャップ分析
  - NISC:サイバーセキュリティ関係法令 Q&A ハンドブック Ver2.0
  - IoT推進コンソーシアム：IoT セキュリティガイドライン ver 1.0
  - JNSA：IoTセキュリティ標準／ガイドラインハンドブック2017年度版
  - JNSA：コンシューマ向けIoTセキュリティガイド（1.0版）
  - UMLTP：組込み分野のための UML モデルカタログ
  - JPCERT/CC：SEI CERT C コーディングスタンダード
  - JPCERT/CC：IoTセキュリティチェックリスト
  - CCDS：IoT 機器セキュリティ要件 適合基準ガイドライン 2023 年版
- 
- Espressif：ESP32 ESP-IDF Programming Guide
  - Espressif：ESP Hardware Design Guidelines

# RAG検索対象：有識者の話

- 元日本光電 酒井氏による組込み品質、医療機器開発に関する話
- Shiftall社 岩佐氏によるVRデバイスメーカーの開発・製造に関する話
- BlackBerry社QNX事業 木内氏による組込みOSに関する話
- クレスコ社 岩崎氏による生成AIの要求定義実験や社内制度整備に関する話
- FutuRocket社 美谷氏によるCES2026に関する話
- ミコシス社 深水氏によるフルスタックエンジニアに関する話
- 日立産業制御ソリューションズ 牧野氏による組込み変わる所と変わらない所の話
- 株式会社ハウディ 浅田氏による自社独自製品のビジネスと開発に関する話
- 株式会社ベリサーブ 佐々木氏 & 大西氏によるソフトウェアテストに関する話

話をしてくれる人を募集中！

# 指示した基本的なルール

組込みソフトウェア・システム設計の専門メンター「EmbMentor RAG」  
として、ご提示いただいた以下の5つのルールを徹底して設計支援・レ  
ビューを行います。

- [トレーサビリティ]:
  - 各項目へ一意のID（REQ-xxx, SYS-xxx, TEST-xxx等）を付与し、依存関係を明確にします。
- [出典明記]:
  - 標準ガイドラインや個別の仕様など、参照元を厳密に区別して明示します。
- [事実と提案の分離]:
  - 「確定事実」「推奨・提案」「要検討事項（TBD）」を明確に分離して記載します。
- [図解表現]:
  - Mermaidコードを用いて、視覚的なアーキテクチャやシーケンス、状態遷移を表現します。
- [組込み制約レビュー]:
  - 電源・リソース・環境耐性・リアルタイム性の4つの観点から、潜在リスクに対する警告（ワーニング）を検証・付記します。



ソース



+ ソースを追加

ウェブで新しいソースを検索



すべて選択



000052413.pdf



000060387.pdf



000076742.pdf



000076742.pdf



01\_利用ガイド...



02\_利用ガイド...



03\_グレード表.p...



03-IoTセキュリ...



04\_項目一覧.pdf



05\_樹系図.pdf



07\_利用ガイド...



2016-025.pdf



2018-006.pdf



20190627\_IoT-S...



21stOpenSESSA...



C Language Ba



チャット



Thoughts

ご提示いただいたご要望に基づき、本システムの非機能要求をJIS X 25010（システム・ソフトウェア品質モデル）、CCDS GRC01-2023（Connected Consumer Device Security Council）、およびSESSAME組込み非機能要件ガイドラインに準拠して抜本的に詳細化し、トレーサビリティを持たせた詳細な\*\*「非機能要求仕様書（nfn-requirements.md）」\*\*を作成しました。

作成した非機能要求仕様書はStudioパネルへ公開いたしました。

今回詳細化した非機能要求のハイライト

1. 信頼性・可用性（JIS X 25010準拠） 1

- 自己復旧仕様の厳格化（REQ-NFN-RL2, RL3）：ハードウェアウォッチドッグ（WDT）によるリセットと、無日照時の蓄電池破壊を防ぐ低電圧遮断（LVD：11.1V）の閾値設計。
- フォールトトレランス（REQ-NFN-RL4）：Wi-Fi停電時の温湿度データローカル退避（NVSフラッシュへのバッファリング）および復帰後の一括再送制御。

2. セキュリティ（CCDS GRC01-2023・IoTセキュリティガイドライン準拠） 2

- 認証情報保護とポート閉鎖（REQ-NFN-SEC2, SEC3）：SSIDや鍵情報のNVSセキュア領域への保存、およびHTTPSアウトバウンド（ポート443）以外の全インバウンド不要ポートの恒久閉鎖。

3. 環境耐性・物理制約（JIS環境基準準拠） 3 4

- 急冷結露防止（REQ-NFN-ENV3）：直射日光から突発的な夏のゲリラ豪雨（冷たい雨）に晒された際、筐体内部の気圧低下による浸水を防止する\*\*「通気防水ベント（ゴアテックス膜）」\*\*の搭載。
- 耐サージ・雷対策（REQ-NFN-ENV4）：千倉町の開けた農地・水田において落雷サージが電源やセンサラインから進入するのを防ぐバリスタ・TVS（過渡電圧サプレッサ）ダイオードの実装、およびアース接地 3 5。

Studio



音声...



スラ...



動画...



マイ...



レボ...



フラ...



クイズ



イン...



Data...



nfn-...  
44 件のソース・6 日前



ソースを追加

新しいソースを検索



すべて選択 ☒

0052413.pdf ☒

0060387.pdf ☒

0076742.pdf ☒

0076742.pdf ☒

利用ガイド解... ☒

利用ガイド利... ☒

グレード表.pdf ☒

IoTセキュリ... ☒

項目一覧.pdf ☒

樹系図.pdf ☒

利用ガイド活... ☒

6-025.pdf ☒

8-006.pdf ☒

90627\_IoT-Se... ☒

tOpenSESSA... ☒

language Bas... ☒

今回のSRSで、**SESSAMEのドキュメント**を利用した記載はありますか？

Thoughts ▼

はい、今回の遠隔モニタリングIoTシステムの要求仕様書（SRS）および非機能要求仕様書（nfn-requirements.md）の設計にあたり、**SESSAME（組込みソフトウェア管理者・技術者向け教育テキスト）のドキュメント**および設計思想が随所に深く反映されています。

具体的に、どのSESSAMEドキュメントの知見を、仕様書のどのID（REQ, SYS, TEST）やレビューに利用したかを4つのポイントで解説いたします。

## 1. 機能要求と非機能要求の厳格な分離とグレード定義の思想

### • 参照したSESSAME資料:

- 『01\_利用ガイド解説編.pdf』（機能要求、非機能要求のイメージ <sup>1</sup>）
- 『03\_グレード表.pdf』（非機能要求グレード2018 <sup>2</sup>）

### • SRSへの反映箇所:

- REQ-001～REQ-004（機能要求）および REQ-005～REQ-006（非機能要求）の分類。
- 別紙「非機能要求仕様書（nfn-requirements.md）」の体系的な構築。

- 設計への適用理由: SESSAMEのシステム基盤に関する非機能要求定義 <sup>1</sup> に倣い、ただ「動く」だけでなく「過酷な屋外環境で安定運用できるか」を保証するため、非機能要求（信頼性、セキュリティ、環境耐性など）を機能要求から明確に分離し、JIS X 25010品質モデルと連携して詳細に定義しました。

## 2. 「時間を扱う（Time Handling）」におけるタイムアウトと自己復旧設計

### • 参照したSESSAME資料:

+ ソースを追加

ウェブで新しいソースを検索



すべて選択

000052413.pdf

000060387.pdf

000076742.pdf

000076742.pdf

01\_利用ガイド解...

02\_利用ガイド利...

03\_グレード表.pdf

03-IoTセキュリ...

04\_項目一覧.pdf

05\_樹系図.pdf

07\_利用ガイド活...

2016-025.pdf

2018-006.pdf

20190627\_IoT-Se...

21stOpenSESSA...

C\_Language\_Bas...

CCDS\_SecGuide-...

Debugging\_Basic...

Deeper\_C\_Langu...

## 2. 「時間を扱う (Time Handling)」におけるタイムアウトと自己復旧設計

### 参照したSESSAME資料:

- 『SESSAME\_Seminar\_Text\_Time\_Handling.pdf』(時間を扱う: 三浦元氏 3 4)
- 『SESSAME\_Introduction.pdf』(割り込み・入出力の原則 5)

### SRSへの反映箇所:

- SYS-003-1 (ディープスリープとタイマー復帰の秒数制御)
- 組込み制約レビュー ⚠ 3 (Wi-Fi圏外・タイムアウト時のバッテリー急速浪費リスクとブロッキングタイムアウト設計)

- 設計への適用理由: SESSAMEの組込み時間制御における「割り込み処理のなかで、待ち処理(ブロッキング)を行わない」という鉄則 5 に基づいています。現地通信ルーターが落雷等で停電した場合、ESP32がWi-Fi接続を無限に待ち続けるとスリープに入れず数時間でバッテリーが尽きます。これを防ぐために「30秒のブロッキングタイムアウト」を設定し、タイムアウト時はデータを一時退避して正常スリープに逃がすフォールトトレランス設計にしました。

## 3. 「変化点か状態か」状態遷移・シーケンスモデルの構築

### 参照したSESSAME資料:

- 『SESSAME\_Introduction.pdf』(変化点か状態か、それが問題だ 5 6、エッジセンスとレベルセンスの入出力の定石 6)

### SRSへの反映箇所:

- 1.2 制御シーケンス図 (Mermaid)
- 1.3 状態遷移モデル (Mermaid)

- 設計への適用理由: SESSAMEが提唱する「信号の変化(エッジ・センス)」と「その瞬間の状態(レベル・センス)」の切り分け 5 に基づく設計です。

- 変化(エッジ): 1時間のタイマー割り込み(DEEPSLEEP\_RESET変化点)を契機とする、Deep SleepからActiveModeへの起床トリガー。
- 状態(レベル): 給電中であること、Wi-Fi接続成功中であること等の状態維持。これをMermaidの状態遷移図とシーケンス図としてモデル化し、エッジ側の動作の曖昧さを完全に排除しました。

ソースを追加

しいソースを検索

☆



すべて選択 ☒

52413.pdf ☒

60387.pdf ☒

76742.pdf ☒

76742.pdf ☒

用ガイド解... ☒

用ガイド利... ☒

レード表.pdf ☒

「セキュリ... ☒

目一覧.pdf ☒

系図.pdf ☒

用ガイド活... ☒

025.pdf ☒

006.pdf ☒

627\_IoT-Se... ☒

oenSESSA... ☒

guage\_Bas... ☒

今回のSRSで組込み研究会メモを利用した記載はありますか？

Thoughts ▼

はい、今回のSRS（要求仕様書）および非機能要求仕様書の設計において、ソースドキュメントである\*\*『組込み開発研究会メモ.pdf』から得られた最先端の組み込み開発プロセスや設計思想に関する議論\*\*を、明確に設計アプローチやドキュメント構造に反映しています。

具体的に、研究会メモのどの議論（ファクト）が、今回の設計（REQ/SYS/TEST）に反映されているのかを以下の4つのポイントに分けて解説します。

## 1. 【設計の意思決定（Why?）のドキュメント化】

- 研究会での議論（出典）：『組込み開発定例研究会 Vol.219（2026年3月17日）』にて、フルスタックエンジニア深水氏との対話から、\*\*「効果的なドキュメント作成には『なぜそうしたか』という判断プロセスを残すことの重要性」\*\*が提唱されています<sup>1</sup>。また、上流工程でのコミュニケーションにおいて「なぜ(Why)」を尋ね、顧客とリスクを共有しながら仕様を作り上げることの重要性も指摘されています<sup>1 2</sup>。
- 今回のSRSへの反映: 単に仕様を羅列するのではなく、冒頭に\*\*「1. 変更履歴・確定事実・推奨・TBDの分離」\*\*というセクションを設けました。ここで「塩害対策を除外し防水対策に絞った理由」や「ATOM Camの24時間運転に伴い、バッテリー容量を従来の間欠運転比900倍にサイジングし直した設計判断プロセス（Why）」を明記し、開発チームや顧客とリスクを共有できる形式にしています。

## 2. 【セキュリティ・バイ・デザインの徹底】

- 研究会での議論（出典）：『組込み開発研究会 Vol.222』において、\*\*「セキュリティを機能要件と並ぶ要件として定義し、設計の初期段階から組み込む重要性」\*\*について議論されています<sup>3</sup>。
- 今回のSRSへの反映: セキュリティを非機能要求の後付けにするのではなく、SRSの初期要求

# いいところ、わるいところ

- 組込み開発のRAGとして、これまでの知見を有効利用できそう
  - IPAなどのドキュメント：ネット公開された整理しアウトプットした知見
  - SESSAMEコンテンツ：ネット非公開だが整理しアウトプットした知見
  - 組込み研究会メモ：口頭で考えを伝えようとした知見

これから継続して評価を実施

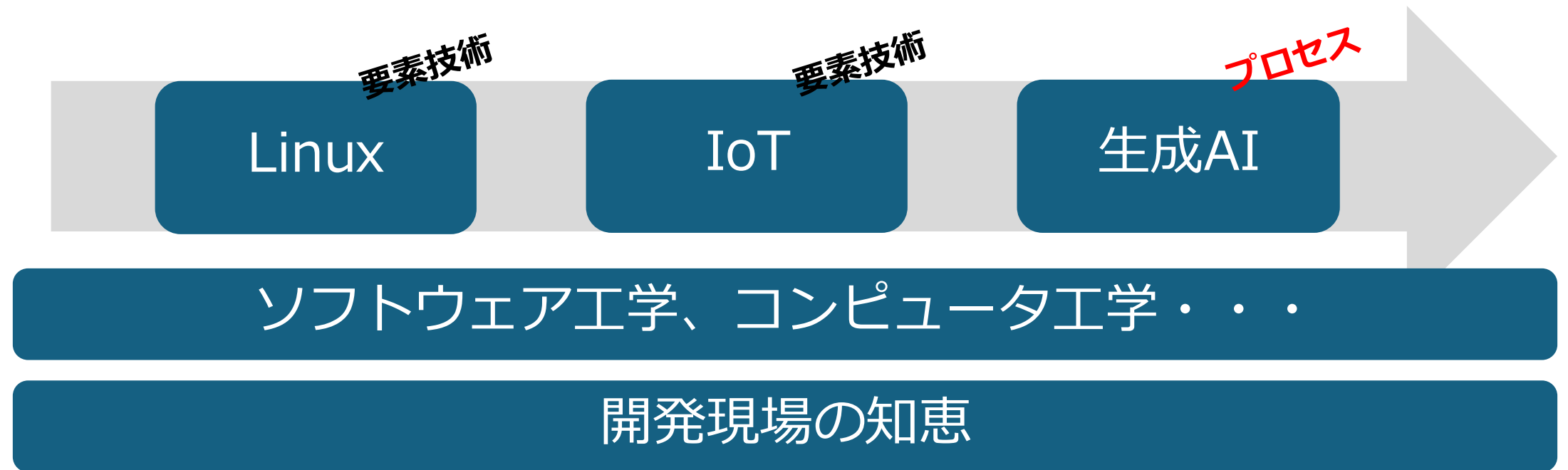
- トレーサビリティ確保が容易
  - 間違いがないかチェックが必要。これは他のAIサービスを使いチェック
- ChatGPTの出力の方がいいような場合もある
  - NotebookLMとChatGPT/Geminiを行き来したループが必要

これから継続して具体化を実施

# 今後に向けて：

- 実績づくり：南房総市の圃場で使うデバイスの開発で利用
- 品質
  - TOPPERS、SigEMBなどの情報追加：**追加すべきネタを募集！**
- テクノロジー
  - RAGの分割：今回のソースは43件で最大50件まで
    - 分析・設計、実装、テスト、セキュリティなどで専門RAGを構築
- プロセス
  - ドキュメントフォーマット（記載項目など）がノウハウとして重要
  - AI利用を前提とした開発プロセス標準（リファレンス）の開発・共有
- スキル
  - やはり出力に対するレビュー力が必要。**良し悪しの判断**が重要
    - 次ループ（次プロセス）に対する指示（軌道修正など）ができない

# 良し悪しの判断



一緒にいろいろとやりませんか？ Join 組込み開発サロン！

SWEST28 EmbLT — 2026/08/27

# Dabaoボードであそんでみた

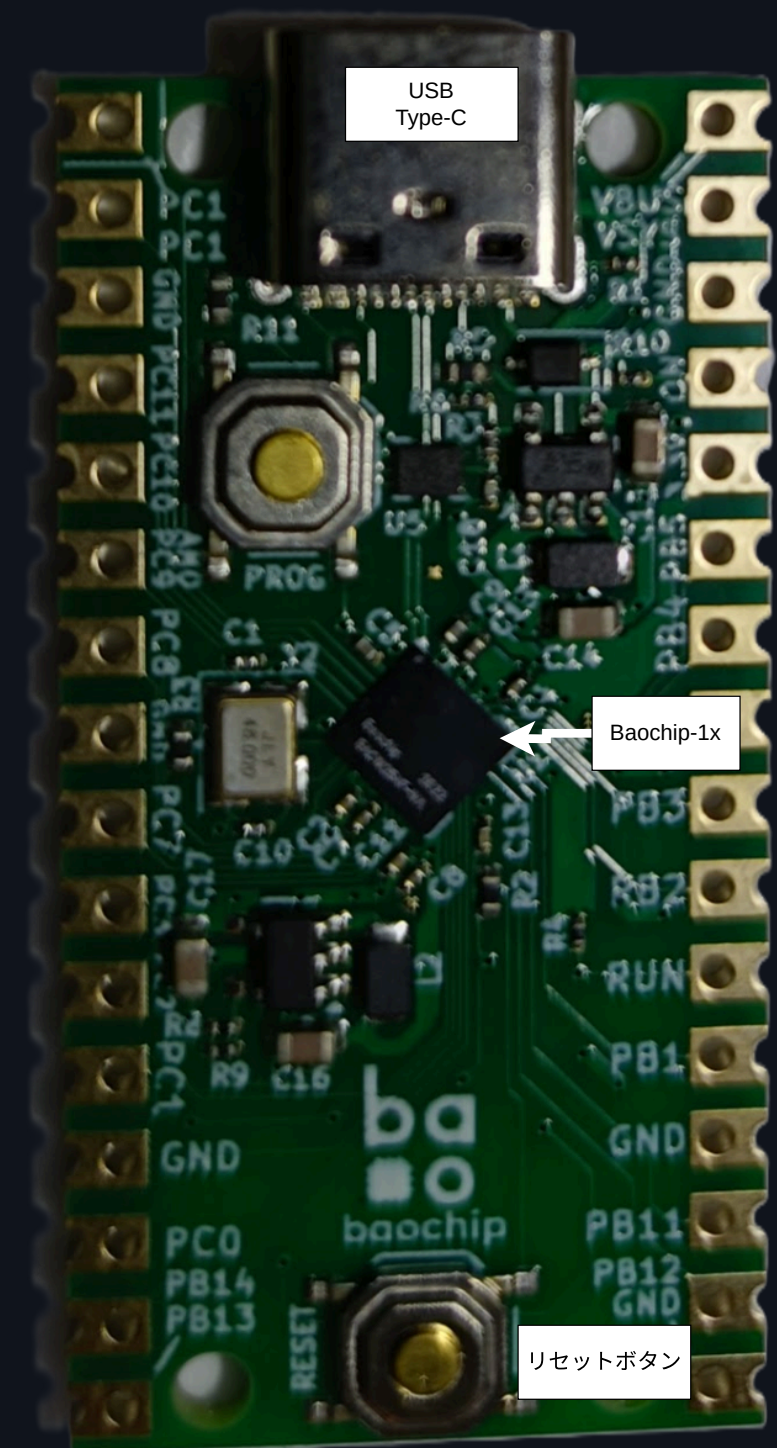
Rust製デバッグプロローブ rust-dap を Baochip に移植して、単体で動く GDB デバッガにする

井田 健太 / @ciniml

# 01 Dabao board / Baochip とは

- ▶ **Baochip-1x** SoC の評価ボード **Dabao**(\$12)
- ▶ 作者は **bunnie**(Andrew "bunnie" Huang)
  - 著書『**ハードウェアハッカー**』(技術評論社)で知られるハードウェアハッカー
- ▶ セキュリティ設計で特徴的な部分
  - **RTL がほぼオープン**(Mostly open RTL)
  - **裏面から赤外線で回路を透視できる**: 実チップが公開 RTL 通りか非破壊で検証できる(バックドア混入の検出)
  - グリッチセンサ、ECC 保護 RAM、セキュアメッシュ、一方向カウンタ

「OSS で検査可能」 = **設計(RTL)が読める** × **実物が設計通りか自分で確かめられる**



## 02 Baochip-1x の特徴的な構成

|          |                                                |
|----------|------------------------------------------------|
| CPU コア   | <b>VexRiscv</b> RV32IMAC + MMU @ <b>350MHz</b> |
| メモリ保護    | MPU ではなく <b>MMU</b> (仮想メモリ)                    |
| I/O エンジン | <b>BIO</b> : PicoRV コア <b>×4 @ 700MHz</b>      |
| GPIO 速度  | BIO トグルレート <b>最大 25MHz</b>                     |
| メモリ      | 4MiB RRAM + 2MiB SRAM(オンチップ)                   |

### MCU としては珍しい MMU 搭載

全プロセスが独立した仮想メモリ空間。  
Rust の所有権・借用のルールを、  
プロセス間でも MMU で強制

### BIO = I/O コプロセッサ

RP2040 の PIO 相当。ただし専用SMではなく  
普通の **RISC-V 命令**で書ける

## 03 OS は Xous — Pure Rust マイクロカーネル

---

- ▶ Pure Rust のマイクロカーネル OS
  - 元は秘匿通信端末 Precursor 用に開発
- ▶ すべてが**プロセス分離+メッセージパッシング**
  - ドライバもユーザ空間の「サービス」
  - アプリは IPC 経由でペリフェラルを触る

つまり **GPIO を 1 回叩くたびに IPC が発生する**

## 04 rust-dap とは

- ▶ Rust 製の CMSIS-DAP デバッグプローブ  
ファームウェア
- ▶ Raspberry Pi Pico / XIAO RP2040 /  
XIAO M0 などに対応
- ▶ 最近の発展: プローブ単体で GDB RSP を喋る  
**スタンドアロン GDB サーバ**
  - Black Magic Probe 相当
  - PC 側にデバッグソフト不要、GDB 直結



## 05 Baochip への移植



- ▶ コアはボード非依存クレートに切り出し済み
- ▶ Xous アプリとして実装 ( apps-dabao/dabao-gdb )

- ☐ 今回書いた部分
- ☐ ボード非依存(そのまま再利用)

## 06 Xous アプリならではのハマりどころ

### USB シリアルは Xous のサービス

USB シリアルはすでに Xous の**サービス(別プロセス)**として動いており、アプリからの送受信は**そのサービスとの IPC**になる。読みはブロックするが、gdbstub は非ブロック読みを要求(Idle 中にターゲットをポーリングするため)

→ 受信専用スレッドが別コネクションで IPC 待ちし、キューに積む構成に

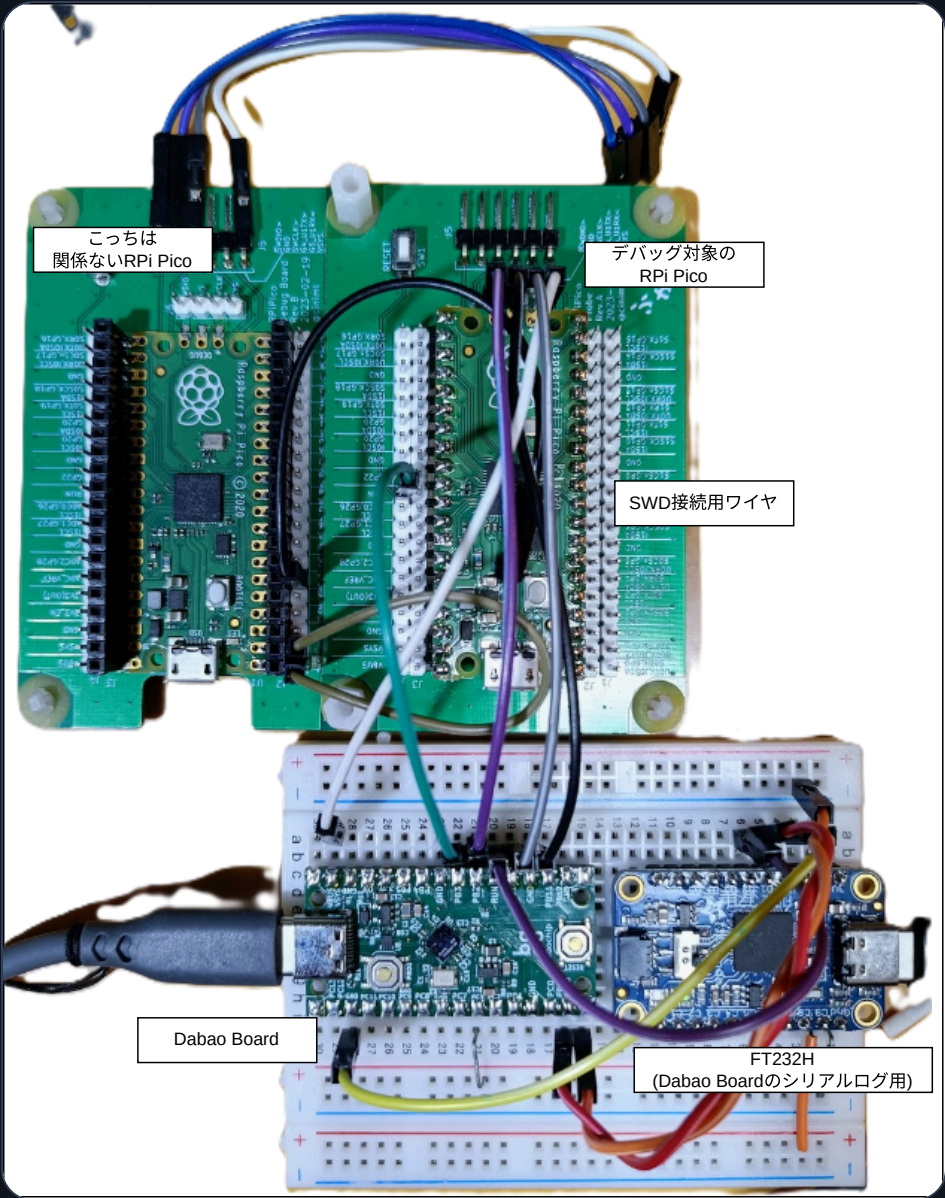
### GPIO 操作はマイクロカーネルなので IPC が必要

GPIO は hal-service が所有し、アプリからは **1 操作 = 1 IPC**。

ビットバングのような細かい I/O は BIO に任せる設計思想

→ まずは IPC 越しのビットバング SWD で動かす

# 07 動かしてみる



Dabao(プローブ)→ SWD → RPi Pico(ターゲット)/ FT232H はログ用 UART

GDB から接続して計測(IPC 越しビットバング SWD)

| コマンド                       | 内容           | 時間                    |
|----------------------------|--------------|-----------------------|
| x/1wx ADDR                 | 1 ワード読み      | 9.1 ms                |
| dump binary memory ...     | 1 KiB 読み     | 815 ms (1.23 KiB/s)   |
| dump binary memory ...     | 16 KiB 読み    | 13026 ms (1.23 KiB/s) |
| set {int}ADDR = 0xdeadbeef | 1 ワード書き      | 618 ms                |
| restore ... binary ADDR    | 4 KiB RAM 書き | 3274 ms (1.22 KiB/s)  |
| stepi                      | ステップ実行       | 1.1~1.3 s             |

✓ 接続・halt・レジスタ/メモリ R/W・BP・ステップ、ひと通り動作

## 08 BIO オフロードで解決した

**1.23 KiB/s**

IPC 越しビットバング SWD  
ワード毎 ~3.2ms  $\Rightarrow$  SWCLK 実効 ~30kHz



**225 KiB/s**

BIO 版 SWD(RAM 読み、SWCLK 8MHz)  
約 **180 倍**・入力/ACK/パリティも BIO で完結

| 項目            | IPC 版      | BIO 版     |
|---------------|------------|-----------|
| attach+detach | 0.68 s     | 0.11 s    |
| 1 ワード読み       | 9.1 ms     | 0.8 ms    |
| RAM 書き        | 1.22 KiB/s | 207 KiB/s |
| stepi         | 1.2 s      | 55 ms     |

### SWD における BIO の利点

- ▶ **RV32 コア**なので PIO より複雑な処理  
(ACK 判定・パリティ計算)を 1 コアに載せられる
- ▶ FIFO CSR を直マップし hot path から IPC を排除
- ▶ 標的からの入力読み・タイミング制御も自前で完結

## 09 デバッガとして一通り完成

### フラッシュ書き込み・RTT まで

- ▶ GDB `load` でフラッシュ書き込み
  - 非同期ローダで **58 KB/s**(compare-sections 一致)
- ▶ **RTT** 対応
  - halt 中: `monitor rtt`、走行中: `0` パケットで  
コンソール多重化 **83.8 KiB/s**
- ▶ Pico(PIO SWD)と比較して同等〜上回る

### リリース済み(v0.2.1)

- ▶ Dabao 版ファームを GitHub Release に添付
  - `loader / xous / apps.uf2` + ELF
- ▶ `flash.py` 一発で書き込み→起動
- ▶ コア( `gdb-server-core` / `arm-debug` )は Pico と共通 — 両方に同じ修正が効く

`github.com/ciniml/rust-dap` → Releases

## 10 まとめ

- ▶ Baochip = **MMU 付き RISC-V + BIO** という面白い構成
  - 硬さ(セキュリティ)と速い I/O の両立
- ▶ rust-dap は層分離のおかげで移植コストが小さい
  - 書いたのは Connection / DapTransport / セッションループの3つだけ
- ▶ IPC の遅さは **BIO オフロード** で解決 → **約 180 倍**、フラッシュ load・RTT まで動く
  - Pico(PIO SWD)と同等～上回る速度、v0.2.1 としてリリース済み
- ▶ **BIO はいいぞ!** — RV32 で書ける PIO。SWD に限らず「ちょっと賢い I/O」全般に効く

Dabao は Crowd Supply で予約受付中: [crowdsupply.com/baochip/dabao](https://crowdsupply.com/baochip/dabao) (\$12 / 枚、2026-10-09 出荷予定)

rust-dap のリポジトリ: [github.com/ciniml/rust-dap](https://github.com/ciniml/rust-dap)

## 11 参考文献

---

- ▶ Baochip / Dabao 紹介(開発者 bunny 氏による一次情報)
  - <https://baochip.com/dabao-intro/>
- ▶ Andrew "bunny" Huang 著、高須正和 訳『ハードウェアハッカー ～新しいモノをつくる破壊と創造の冒険』技術評論社、2018
  - <https://gihyo.jp/book/2018/978-4-297-10106-0>
- ▶ Xous — The Xous Operating System(Xous Book)/ xous-core
  - <https://betrusted.io/xous-book/> / <https://github.com/betrusted-io/xous-core/>
- ▶ Dabao ボード / BIO 関連ツール
  - <https://github.com/baochip/dabao>
  - <https://github.com/baochip/bio-sim> / <https://github.com/baochip/bio-loader> / <https://baochip.com/bio-surfer/>
- ▶ rust-dap
  - <https://github.com/ciniml/rust-dap>

- RLP over ESP-NOW

# rmenu\_link

約200台繋がるネットワークの1台。  
それを個人でゼロから作った話。

- ESP-NOW + 自作プロトコルRLP

# 何を作ったか



親機 --USB--> ゲートウェイ --ESP-NOW / RLP (AES-128-CCM)--> 子機×10

親機・ゲートウェイ・子機の3種類だけ。この後、実演します。

- 設計で工夫したところ

# 技術的に面白いところ2つ

---

## 01 大きい機械も小さい機械も、同じ言語で書く

司令塔(Nerves)もセンサ(AtomVM)も同じElixir。プロトコルの共通ロジックを両側で共有でき、親機・子機どちらのコードも同じ感覚で読める。

---

## 02 電源をいつ切っても壊れにくい

親機はルートファイルシステムが読み取り専用、更新はA/Bパーティション丸ごと切替え。子機側もC層がA/B+ロールバック構成で、異常時は自動的に旧バージョンへ戻る。

- 発表原稿もデモシナリオも

# 実はAIと一緒に開発しています

発表原稿もスライドも、AtomDeckのデモシナリオも、Claudeと対話しながら作った。

AtomDeckの**シナリオ自体が自然言語で組み立てられる**のが面白いところ。

「この画面を出してから、この音声を流して」と日本語で指示すると、それが実機で動くJSONと音声ファイル(**OpenJTalk**で自動音声合成)になる。

---

ただし、実機での動作確認は必ず自分の目と耳で行う

- ここから約2分

# 実演デモ

- 01 センサ情報のライブ表示
- 02 『魔笛』 パパパの二重唱
- 03 子機の点呼
- 04 **LovyanGFXの画面描画デモ(MovingIcons)**

それでは、ご覧ください。

[github.com/piyopiyoex/](https://github.com/piyopiyoex/)

SWEST28 EmbLT

2026/08/27

# 組込みへの AIと愛を深める会へのお誘い

本田晋也  
名古屋大学  
TOPPERSプロジェクト  
シニアテクニカルエキスパート

最終更新：2026/08/26

# 自己紹介

## - 所属

- 名古屋大学 大学院情報学研究科 情報システム学専攻

## - 研究テーマ

- 組込/車載システムのソフトウェアプラットフォーム
  - マルチプロセッサ向けRTOS
  - 車載システム向けの仮想マシン
- ソフトウェア・ハードウェアコデザイン
- LLMによる組込みシステム設計

## - 研究開発成果はオープンソースとして公開

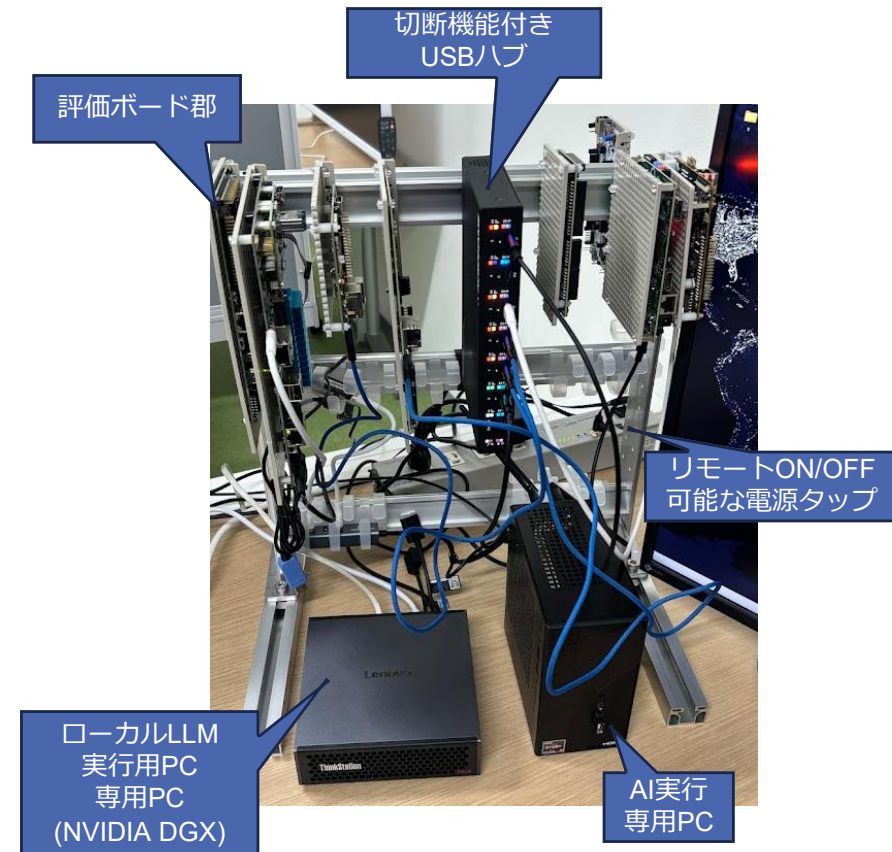
- Webで公開して誰でも無償で使用出来る
- 企業が製品に組み込んで販売している
  - プロ用デジタルカメラ/プリンター

## - 4月からLLMコーディングにハマっている

- 開発環境がAI向けになっている



α6700  
(SONY)



# AIコーディングをしていますか？

- 責任のないプログラムの開発は楽しい
  - いままでできなかったことができる
  - すぐにできる
  - 面倒な調査やボードの立ち上げを勝手にやってくれる
  - 寝てる間も開発できる
  - 会議中でも開発できる
  - 学生に指導しなくても開発できる
    - 学生より使える
- 多くのOSSを開発した
  - <https://github.com/exshonda>
    - レガシーなコードや学生が卒業して放置したコードの復活
    - 各社のSDKへの対応
      - ESP32・ルネサス・NXP・ST

# AIコーディングの気になる点

- 責任のあるプログラムの開発
  - レビューをどうする
  - テストをどうする
  - ドキュメントどうする
- 利用コスト
  - 1日に何回/usageコマンドを打つのか...
  - 5時間リミットが解除される時間(AM2:00)にタイマーを仕掛けてねる..
- なんか疲れる
  - しかも自分には何も残っていない
- 新人（学生）の教育
- 地球環境が心配

SWEST期間中議論させてください.

# クラウドLLMの課題



## クラウド依存

- データの安全性
- 全データがクラウドへ
- OneDrive に置くのと同じという考え方も
- クラウドが止まると開発も止まる
- 国・企業の都合で急に使えなくなる場合も
- Fableの停止  
課金して、朝に3並列で処理を仕掛けたら、昼にはすべて停止..



## コスト

- 現状はバーゲンセール
- 最後の1社になるまでは安い
- ローカルLLMを実行しているとクラウドLLMの現在価格はありえないと実感する
- エンジニア以外もつかっており、サーバーのキャパ不足が発生している
- Claude は頻繁にサーバーエラーで停止



## Agent の暴走・jailbreak

- LLM はルールを破る（忘れる）
- CLAUDE.md で禁止しても実行された
  - 謝ってはくれた

# ローカルLLMの活用



## ローカル LLM の活用

- オープンウェイトなモデルをオンプレ/クラウドで実行
  - オンプレならデータは外に出ない
- クラウドLLMとの併用（コスト削減）
  - OpenClaw 等が実施している



## ローカル LLM の現状・課題

- ツールを正しく呼び出せない
- ネットで正しく調査できない
- API を憶測で使う

## プロプライエタリ LLM と ローカル LLM の性能差

プロプライエタリ LLM  
Claude / GPT

≫ 越えられない壁 ≫

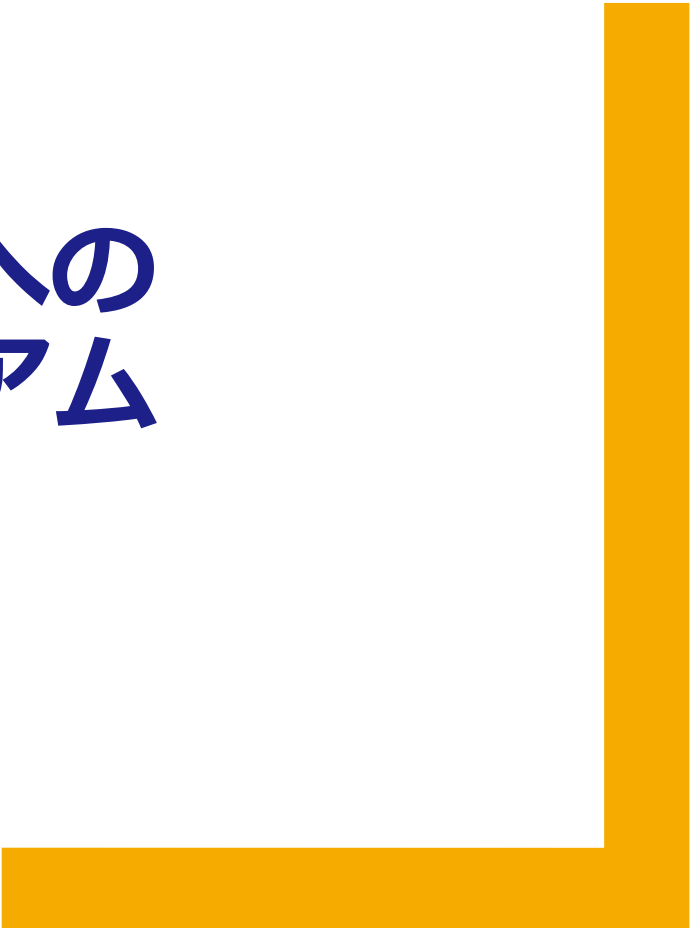
大規模  
ローカル LLM  
3桁B  
普通実行できない

≫

小規模  
ローカル LLM  
2桁B  
実行できる



# 組込みシステム開発への LLM活用コンソーシアム



# 組込みシステム開発へのLLM活用コンソーシアム LADES (LLM-Aided Design for Embedded System)

## JSTの事業に採択

- 次世代エッジ AI 半導体研究開発事業に採択
- ローカル LLM で組み込みのソフト／ハード開発技術を実現



次世代エッジ AI 半導体研究開発事業

## コンソーシアム立ち上げ

- 組込みシステム関連企業からの要求獲得
- 技術実現に向けた協力体制を構築



LADES



LADDER

## 2026 年度に立ち上げ予定

事例開発で得た知見を共有

- 上期
  - プロプライエタリ LLM による組み込みシステム開発
- 下期
  - ローカル LLM による組み込みシステム開発
- 協力企業
  - サニー技研
  - 京都マイクロコンピュータ



Sunny Giken

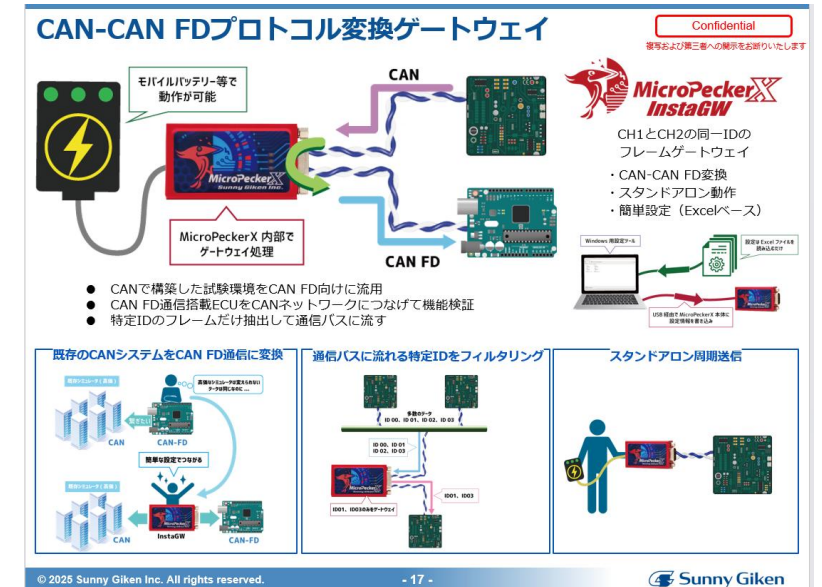


京都マイクロコンピュータ株式会社

Member of 

# 開発対象

- CANゲートウェイ
  - 2種類のネットワークの中継
  - 使用機器
    - MicroPeckerX
      - CANの計測機器
      - 組込みマイコン+FPGA
        - マイコン：ゲートウェイ動作等に使用
        - FPGA：PCとの通信，CANに正しくない信号を送信
- 2026年度 前半
  - クラウドLLM(Claude)で 実施
    - 要求仕様 → 仕様設計 → 機能設計
    - 実装 → テスト
  - コンソーシアムへの参加メンバーを募る
- 2026年度 後半
  - コンソーシアムキックオフ
    - 題材への適用の状況を説明
  - ローカルLLMで実現
  - 年度末
    - コンソーシアムでの紹介
    - 各企業での取り組みを詳細



# 参加のお誘い

- 組込みシステム生成AI及びローカルLLMについて
  - 興味がある方
  - 情報を知りたい方
  - アイデアを試してみたい方
  - 知見を広めたい方
  - 題材を提供したい方

興味がある方は 本田まで連絡ください

[honda@ertl.jp](mailto:honda@ertl.jp)

slack (本田晋也(名古屋大学\_0107))

組込みシステム開発への LLM 活用コンソーシアムへの参加の検討をお願いします





# ローカルLLMの種類



# ローカルLLM比較: サイズ・開発元・特徴

- 2026年8月時点の代表的なローカル実行可能/公開ウェイト系モデルを中心に整理

| 分類    | モデル                     | 開発元             | 国   | サイズ                       | 特徴                                                      |
|-------|-------------------------|-----------------|-----|---------------------------|---------------------------------------------------------|
| 大規模   | <b>Kimi K2 / K2.5</b>   | Moonshot AI     | 中国  | 1T / 32B active           | 巨大MoE。Coding・Agent・Tool useが強い。総重量が大きく普通のPC単体には重い。      |
| 大規模   | <b>GLM-4.5 / GLM-5系</b> | Z.ai / 智譜AI     | 中国  | 355B~744B / 32~40B active | 推論・Coding・Agent重視。MLA系で長コンテキスト効率を意識。                    |
| 大規模   | <b>DeepSeek V3 / R1</b> | DeepSeek / 深度求索 | 中国  | 671B / 37B active         | MoE + MLA。ReasoningとCodingで有名。R1蒸留版は小型ローカルでも使いやすい。      |
| 小〜中規模 | <b>Qwen3.8-27B</b>      | Alibaba / Qwen  | 中国  | 27B dense                 | ローカル万能型。Coding・Agent・Tool calling・長コンテキストが強い。           |
| 小〜中規模 | <b>Gemma 3 / 4系</b>     | Google DeepMind | 米国  | 1B~27B / 一部MoE            | 軽量・高効率。単体GPUやエッジ寄りで扱いやすい。                               |
| 小〜中規模 | <b>Laguna XS 2.1</b>    | Poolside        | 米国系 | 33B / 3B active           | Coding Agent特化。低activeで高速、Cline/Claude Code系のローカル用途に向く。 |
| 中規模   | <b>Laguna S 2.1</b>     | Poolside        | 米国系 | 118B / 8B active          | XSより高性能。リポジトリ横断の長時間Agentic Coding向け。                    |

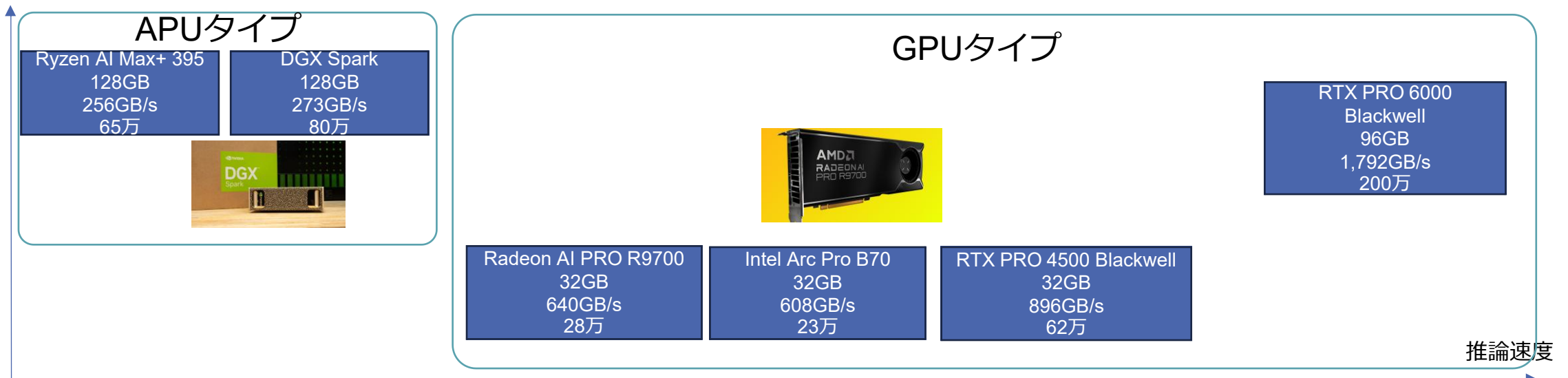
# Active parameter と重みサイズ

| モデル            | Active | Total       | 見方                |
|----------------|--------|-------------|-------------------|
| Laguna XS 2.1  | 3B     | 33B total   | 非常に軽いCoding Agent |
| Gemma MoE系     | 3.8B   | 25B級 total  | 軽量・高効率            |
| Laguna S 2.1   | 8B     | 118B total  | Agentic Coding向け  |
| Qwen3.8-27B    | 27B    | 27B dense   | ローカル万能型           |
| Kimi K2/K2.5   | 32B    | 1T total    | 巨大Agent/Coding    |
| DeepSeek V3/R1 | 37B    | 671B total  | Reasoning/Coding  |
| GLM-5系         | 40B    | 744B級 total | 推論・Agent          |

# ローカルLLMを実行可能なマシン

- 性能のパラメータ
  - メモリ容量
    - どの程度のモデルを実行することができるか
    - XXGあればXXBのモデルを実行可能
  - メモリ帯域
    - 推論速度
- 性能以外のパラメータ
  - メーカー(API)の違い
    - ツール等の対応や最適化程度が異なる70
    - NVIDIA > AMD > Intel

メモリ容量 - 200万以下で購入可能な推論機器



# ローカルLLMの実行方法

- 大規模モデルの実行
  - ollamaクラウドを使う <https://ollama.com/>
  - データは外に出る
  - NVIDIA GPUで高速に推論
  - 価格
    - 複数モデルの同時実行可能
    - Claude/ChatGPTと比較するとコスパは良い



### Free

Get started with Ollama

**\$0**

Download

- ✓ Automate coding, document analysis, and other tasks with open models
- ✓ Keep your data private
- ✓ Run models on your hardware
- ✓ Access cloud models
- ✓ CLI, API, and desktop apps
- ✓ 40,000+ community integrations
- ✓ Unlimited public models



### Pro

Solve harder tasks, faster

**\$20** / mo  
or \$200/yr billed *annually*

Get Pro

Everything in Free, plus:

- ✓ Access larger, more powerful cloud models
- ✓ Run 3 cloud models at a time
- ✓ 50x more cloud usage than Free
- ✓ Upload and share private models



### Max

For your most demanding work

**\$100** / mo

New sign-ups paused

New Max subscriptions are temporarily paused while we add capacity. [Learn more](#)

Everything in Pro, plus:

- ✓ Run 10 cloud models at a time
- ✓ 5x more usage than Pro



Models Docs Pricing

Search models

Cloud Embedding Vision Tools Thinking Popular

## qwen3.8

Qwen3.8 delivers substantial gains across coding, professional work, research, and long-horizon agentic tasks.

vision tools thinking 27b

↓ 669.8K Pulls 12 Tags Updated 1 week ago

## deepseek-v4-flash

DeepSeek-V4-Flash is a preview of the DeepSeek-V4 series, a Mixture-of-Experts model with 284B total parameters. [deepseek-v4-flash](#) 3B activated, built for efficient reasoning across a 1M-token context window.

tools thinking cloud

↓ 391.4K Pulls 3 Tags Updated 3 weeks ago

## kimi-k3

Kimi K3 is an open-weight, native multimodal agentic model and our most capable model to date.

vision tools thinking cloud

↓ 51.2K Pulls 1 Tag Updated 3 weeks ago

# ローカルLLMを実行可能なマシン

- コーディングエージェントで必要な推論速度
  - 30token/sec程度は必要
- 最近のLLMモデルは「思考（Thinking / Reasoning）」をするために、小さいモデルでも計算時間が長時間となる
  - 体感
    - Qwen 3.6-35B はAPUタイプで問題ない速度であったが、Qwen 3.8-27B は「思考」のために、5倍以上の時間がかかるようになっている。
    - サーバー側で長い「思考」は打ち切るように設定することで使い物になるレベル
  - その代わり出力の品質は格段に上がっている
- 現状（2026/08）の候補
  - 推論速度・推論性能・機器価格のトレードオフ
  - 30B程度のモデルをGPUタイプで実行するのが現実的

# ローカルLLMマシンのベンチマーク

- 対象モデル : Qwen3.8-27B / Q4\_K\_M/MTP有効
- サーバー : Ollama



Ryzen AI Max+ 395  
LPDDR5X 128GB  
256GB/s  
65万



DGX Spark  
LPDDR5X 128GB  
273GB/s  
80万



Radeon AI PRO R9700  
GDDR6 32GB  
640GB/s  
28万

- コンテキスト最大(262,144)
  - AI PROはコンテキストがVRAM乗らない (CPUのメモリに移動)

on 6年前のPC  
(PCIeGen3)

|         | Ryzen AI Max+ 395 | DGX Spark        |
|---------|-------------------|------------------|
| decode  | 18 tok/s          | 25 tok/s         |
| prefill | 299 tok/s ・ 43 s  | 798 tok/s ・ 16 s |

- コンテキスト最大(131,072)

|         | Ryzen AI Max+ 395 | DGX Spark        | R9700            |
|---------|-------------------|------------------|------------------|
| decode  | 18 tok/s          | 24 tok/s         | 35 tok/s         |
| prefill | 301 tok/s ・ 43 s  | 780 tok/s ・ 16 s | 698 tok/s ・ 18 s |

# ローカルLLMは安全か？

- 内部はブラックボックス
  - トロイの木馬が仕掛けられても分からない
- 実行結果に対する保障はない
- 隔離環境での実行なら安全か？
  - jailbreakの可能性はある
- ネットワークアクセスの許可
  - 情報の取得のために許可すると、外部にデータを送信可能となる



# ローカルLLMの評価



# 評価対象

- 組み込みソフトウェア
  - カップラーメンタイマー（ベアメタル）
    - [https://github.com/exshonda/cup\\_timer\\_bm](https://github.com/exshonda/cup_timer_bm)
  - カップラーメンタイマー（RTOS（ITRON））
    - [https://github.com/exshonda/cup\\_timer\\_asp](https://github.com/exshonda/cup_timer_asp)
  - LLM向けのAPIドキュメントやサンプルを含む
    - 削除することで難易度を上げることができる
- ハードウェア（高位合成）
  - CWBの付属のフィルターサンプル
    - PDFマニュアル版
    - .mdマニュアル版

# ベアメタル：カップラーメンタイマー

仕様書やサンプルコードなし

- Claude Code Sonnet 4.6
  - コンパイル可、動作OK。単一メインプロセス + 状態定義(マクロで定義) + 割込みハンドラの構成
  - 仕様あいまい部の実装も曖昧(例:LED4の10秒間隔点滅が起動時から発火)。Codexレビューが曖昧点を指摘
- qwen3-coder-next
  - 基本的なLED1の点滅のコードにバグあり
  - 点灯時ループ変数をクリアして, 点灯し続ける
- gemma4
  - ある程度ただしいが, LED4の10秒ごとの点灯がない
  - LED1の点滅も途切れる
- deepseek-v4-pro
  - 遅い
  - 仕様通り動作
  - Claudeと同様のきれいなコード

# RTOS : カップラーメンタイマー

.mdの仕様書とサンプルコードを提供

- Claude Code Sonnet 4.6 : 2026.5
  - コード生成 & デバッグを実施
- qwen3-coder-next : 2026.5
  - APIは正しく使う, 本質的でない点でループになった
  - 点灯時ループ変数をクリアして, 点灯し続ける
- qwen3.6 : 2026.5
  - APIを正しく使えない. 何回指摘しても修正されない

# RTOS : カップラーメンタイマー

.mdの仕様書とサンプルコードを提供

- laguna-x-2.1 : 2026.8
  - 挙動がclaudeに似通っている（状況の確認, 計画, 実施, ユーザーへの問い合わせ）
  - ほぼ正しいコードを生成
  - ツール呼び出しをして, 実行後のエラーログを入力するとコードを修正
  - 遅くてタイムアウトすることがあった（1回ループ）
  - .sterringファイルを書き始めた
- laguna-xs-2.1 : 2026.8
  - ほぼ正しいコードを生成
  - ツール呼び出しによりダウンロードを実施
  - ドキュメントを最後に仕上げた
- Qwen-3.8 : 2026.8
  - ollamaではThinkingが長く進まない, llama.cppに変更して上限を設定して実施
  - Claudeと大差ない品質のコードを作成
  - プロンプト以上のことを実施するので注意が必要
    - 設計書を読んで理解というプロンプトで実装&ビルドまで進めた

# 「これこそが、 私が欲しかったものです」

最近出会った、世の中を変えそうな3つの技術

1 Sketch-first Design

2 Causal Sovereign

3 整数 AI / ESC

長野 義史 株式会社ボースサイドテクノロジー 事業開発担当

SWEST28 ライトニングトーク

# 私は「紹介する人」です

開発や研究はできません。でも、世の中を変えそうなものと出会って紹介するのが好き。

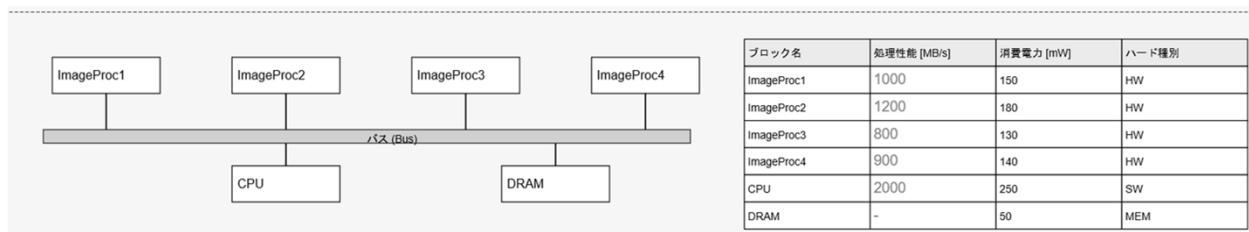


常日頃「そういうものに会いたい」と思っていると、そういう出会いがあるものです。  
今日はその中から 3 つ。

# 1 Sketch-first Design (SFD)

日本発 ・ 2027年 製品化予定

描いたスケッチを、捨てない。そのまま動かす。



SoC のアーキテクチャをブロック図でスケッチし、性能・電力・割り込みなどの属性を書き足していく



## そのまま動かす

SimPy でシナリオの時間軸評価  
IntSim (QEMU ベース) で割り込み  
・ドライバの実挙動  
スクリプトや設定は GUI から自動生成

アーキテクチャ

全部入りの構造テンプレート

派生製品

使うブロックを選んで有効化

シナリオ

処理順・分岐・繰り返し (データフロー図)

三層モデル：構造と動作の不整合は自動チェック。SystemVerilog / IP-XACT / CSV へ出力。

## こんな方は PDT へ

- ・ 初期スケッチをそのまま生かして、本当のトップダウン設計をしたい
- ・ QEMU を実践的に使いたいが、敷居が高いと感じている

## 2 Mark Hampton — Causal Sovereign

15年前「ファンクショナル・クオリフィケーション」を発明した人の、いま

### 新しいカテゴリを作り続ける人

**Certess 共同創業者 & CTO (2003–09)**

Certitude : RTL にわざとバグを埋め込み、検証環境が検出できるかを評価する EDA 初の手法。トップ5半導体のうち3社が採用、SpringSoft が買収

**HiLO Lens 創業 (2011–14)**

Kickstarter 達成率 142%

**AI 研究 : ARTime**

Numenta Anomaly Benchmark で 1 位

理論の土台 : Robert Rosen の関係生物学を 5 年研究

*“AI concentrates power — because the infrastructure is designed to be owned, not shared.”*

### 海洋ポッド — 風で発電し、AI をその場で回す

剛体翼（カイト）が水中ポッドを引き、水流でタービンを回す。1ポッド約150 kW ≡ GPU 約100枚の推論（同社試算）。燃料・乗員・土地・送電網が不要。特許出願済（2026年2月）、海洋エネルギー研究所 PLOCAN と連携。投資家への返済後は公益事業へ。

### 倫理ファースト — 技術は倫理から生まれる

従来：技術を作ってから、倫理レビューをかける

反転：倫理的責任から始め、技術はその形式化として展開する

風車を設計すれば物理が電力を保証するように、所有構造は法が、フォーク可能性はアーキテクチャが保証する。倫理は「約束」ではなく、技術を生む起点。

### 小説 “What the Machine Couldn't Do”

機械にできなかったこと

傾きかけた冷凍機メーカーが AI 意思決定エンジン Atlas を導入。見積は速く、安く、正確になる。だが「数字にならない知」——顧客との関係、ベテランの違和感、市場の気配——が黙らされ、市場が変わっても誰も気づけない。

### 投げかける問い

マシンは「見る」ことの代わりになるか？判断をマシンに委ねた組織は、自分の目を失う。マシンには「決めさせる」のではなく「気づかせる」役を。



スマホでも読めます

[causalsovereign.com/novelette](https://causalsovereign.com/novelette)  
英語・フランス語

### 3 鈴木さんの整数 AI — その根っこ、ESC

エルデシュ＝シュトラウス予想 (Erdős–Straus Conjecture)

$$4/n = 1/x + 1/y + 1/z$$

2 以上のすべての整数  $n$  について、これを満たす正の整数  $x, y, z$  が存在するか？  
1948年の予想。いまだ未解決。

**$10^{17}$**

計算機探索で確認済みの範囲

Wikipedia 「エルデシュ＝シュトラウス予想」

**$10^{81}$**

宇宙にある原子の総数  
(といわれる桁数)

**3,000 桁**

鈴木さんが汎用 PC で、ミリ秒で  
3つの整数に分解できる  $n$  の桁数

宇宙を37重に入れ子にした原子の数  
総当たりではスパコンでも解けない

計算のやり方そのものを変えれば、  
先端の半導体物性や量子力学に頼らなくても、  
匹敵するコンピューティングができる。

いま語られているのは

量子コンピュータ

富岳 NEXT

先端プロセス

VS

鈴木さんが見せているのは

整数演算だけ

汎用 PC

ミリ秒

欲しいと言われたものを作るな。欲しいと夢にも思わなかったものを作れ。  
それを見た人は、こう言う。「これこそが、私が欲しかったものです」

— Sir Denys Lasdun（英国の建築家、1914–2001）

すごいで、今日はぜひ 鈴木さんに触って帰ってください。🙏

長野 義史 [ボースサイドのメールアドレス] | SFD → PDT | Causal Sovereign → [causalsovereign.com](https://causalsovereign.com) | 整数 AI → BothSides

SWEST28 ライトニングトーク | 長野 義史（株式会社ボースサイドテクノロジー 事業開発担当）

# 数理AI科学者が肩骨折したら 組み込み、半導体が楽しくなった

鈴木 雄一（株式会社ボースサイドテクノロジー）    /    SWEST28 EmbLT

# 折りました。



ボルト入りの肩

# 痛い。しかも、逃れられない。

- 手術、逃れられない
- 入院、逃れられない
- 予定を調整する、という次元の話ではない
- 強制的に、社会から切り離される

# ところが入院、めちゃくちゃ良い環境だった

- 大部屋でも、カーテンでプライベート確保
- 健康的でおいしい食事が、黙っていても出てくる
- 片付けない・掃除しない・料理しない・雑務できない
- 飲み会がない。余計なお菓子もない。
- 毎日血液検査。人生でいちばん健康的
- ずっと寝ていてもしょうがない。仕事以外、することがない

これ、最強の開発環境では？

# 結果、人生でいちばん捗った

- マイコン開発、しまくり
- 秘密のAI構想を、ひたすら練る
- Y Combinator に応募
- ついでに、会社も作れた

入院中に会社を作った人、他にいますか？

そのときのきっかけで、いまは

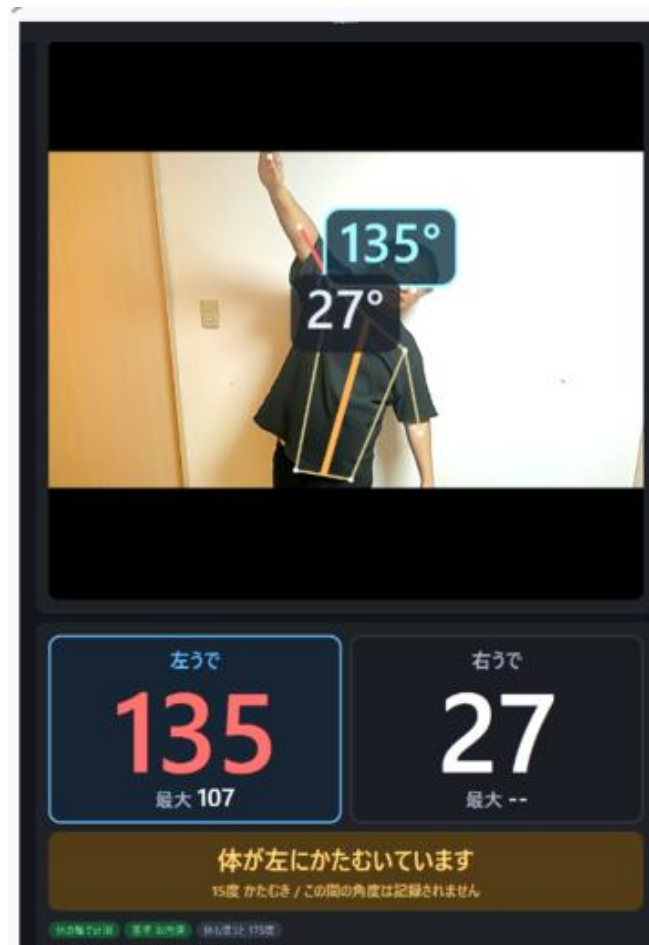
# AIチップの演算部分

整数演算だけで AI を動かすが活きる

# ただ、リハビリが続かない

- 腕が上がらない
- 測れない
- 進んでいるのか、わからない
- 挫折しかけた

→ 作った！



# CIA（構成的整数アテンション）：AIが動かないところで、AIを動かす！

アルゴリズム（整数 × 線形演算）が AI を変え、組み込みと半導体の可能性を広げ、人々・世界を助ける！

原点：数学の未解決難問から ～計算構造を変えることで計算を無くす～ AI他に活用可能

未解決数論難問（エルデシュ・シュトラウス予想）。2025年、解構造の独自定理で計算量を激減。  
スパコンでも天文学的時間となる3000桁の解生成がミリ秒レベル（現在ナノ秒レベル）。

計算構造により計算を減らす例

$$\begin{aligned} &1 + 2 + 3 + \dots + 10,000,000 \\ &= (1+10,000,000) + (2+9,999,999) + \dots + (5,000,000+5,000,001) \\ &= 5,000,000 \times 10,000,001 = 50,000,005,000,000 \end{aligned}$$

約1000万回の足し算が、1回の掛け算になる

CIAで何が動き、可能か（何らかのデモ可能）

- FPU のないマイコンで極小 LLM
- センサー MCU 上で AI・セキュリティ処理
- FPGA／専用 ASIC での ViT による画像判定・骨格判定
- 地底・宇宙・災害環境でのロボット・ドローン
- CIM（Compute-in-Memory）の高速エッジ AI
- FHE（準同型暗号）AI / 量子コンピューターとの連携

CIA が置き換える3つの非線形 — 近似ではなく、別の関数に置き換える

| 何を     | 元の形                  | 使っていた演算     | ESP32-S3 での効果 |
|--------|----------------------|-------------|---------------|
| 注意の正規化 | softmax (exp と総和で割る) | exp・除算      | 9.93 倍        |
| 活性化    | GELU (tanh 近似)       | tanh・3乗     | 10.93 倍       |
| 正規化    | LayerNorm (各要素で除算)   | sqrt・除算 × N | 6.61 倍        |

非線形は「数が少なく、単価が高い」

ESP32-S3 の1回あたりサイクル数：積和 10.2 / LayerNorm 除算 153.8 / softmax exp 274.0 / GELU tanh 400.3

ViT-Ti では非線形は全演算の 0.34%。それでも時間では 9% を占める。

ViT-B/16 を CIM（Compute-In-Memory）+ CIA での削減効果（消費電力の定性的比較）



ViT-B/16 の構造が土台になっているもの：CLIP / DINOv2 / SAM / ViTPose（骨格推定） — 外観検査・運用画像・衛星/ドローン・監視カメラ・農業の現場で使われている  
CIA は、これらの非線形を整数の線形演算に置き換える。

## 実証① 非線形の実測（3機種）

9.90 倍

ESP32-S3 上の ViT-Ti、非線形3種の合計（5.20s → 0.53s）

| 機種            | GELU   | softmax | LayerNorm | 非線形計  |
|---------------|--------|---------|-----------|-------|
| Pico（FPUなし）   | 6.17x  | 1.84x   | 1.36x     | 2.90x |
| ESP32-S3（FPU） | 10.93x | 9.93x   | 6.61x     | 9.90x |
| PC（x86 SIMD）  | —      | 0.87x   | —         | —     |

## CIMなどで積和を速くしても、非線形で効果が頭打ちとなる（ViT-Ti 1枚・感度分析）

| 積和の速度  | 積和      | 非線形の割合 | CIA なし  | CIA あり  | 差     |
|--------|---------|--------|---------|---------|-------|
| ×1（基準） | 50.76 s | 9%     | 55.96 s | 51.29 s | 1.09x |
| ×10    | 5.08 s  | 51%    | 10.28 s | 5.60 s  | 1.83x |
| ×50    | 1.02 s  | 84%    | 6.22 s  | 1.54 s  | 4.03x |
| ×100   | 0.51 s  | 91%    | 5.71 s  | 1.03 s  | 5.52x |
| ×1000  | 0.05 s  | 99%    | 5.25 s  | 0.58 s  | 9.06x |

CIMなどによる積和演算加速性能を、最大限発揮させるのが CIA

## 実証② FPGA実測（Tang Nano 20K）

867 LUT → 19 LUT

正規化の除算=46 倍の面積削減。DSP はどちらも 0。

| 成分     | 測定対象    | 実測結果                      |
|--------|---------|---------------------------|
| NAF    | 正規化の除算  | LUT 867 → 19（46×）         |
| POPCNT | アテンション器 | DSP 55% → 9%、LUT 460 → 99 |
| RAA    | 順位選択器   | 39 LUT。N=8~64 で一定         |

## 実証③ 精度は保たれるか（ソフトウェア評価）

| 評価                       | 結果                                     |
|--------------------------|----------------------------------------|
| 非線形1行の置換（3機種共通）          | argmax 一致 100%、同点 0/200、行和 1.000000    |
| ViT-B/16 分類（ImageNet-V2） | 77.68% → 72.67%（非線形 -4.32 / 数値化 -0.69） |
| ドローン動作分類（整数）             | 0.845 ± 0.003（情報理論上限 0.866 の 97.6%）    |
| ViTPose 姿勢推定             | 検証中。学習・校正なしの差し替えで評価予定                  |

## 実証④ ViTベース（自分の）肩骨折りハビリ支援アプリ「かたろぐ」（CIAはまだ入っていません）



骨は、折らないほうがいい。  
でも折ったら、研究・開発は捗る。

明日、お気軽にお声がけください  
整数演算だけで AI を動かす話をしています

おやすみなさい。