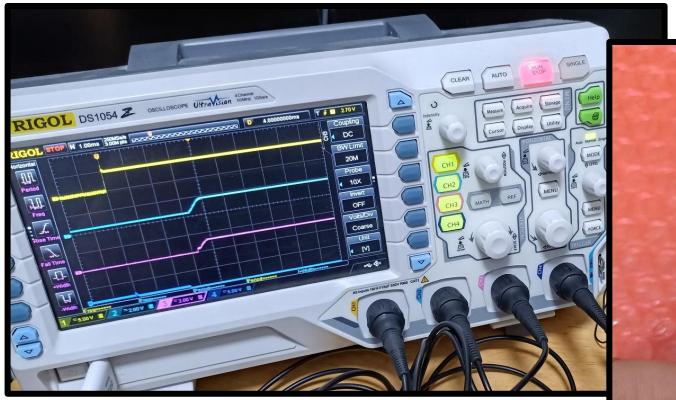
Zephyr(RTOS)でARMとRISC-Vのコア間通信をしてみた

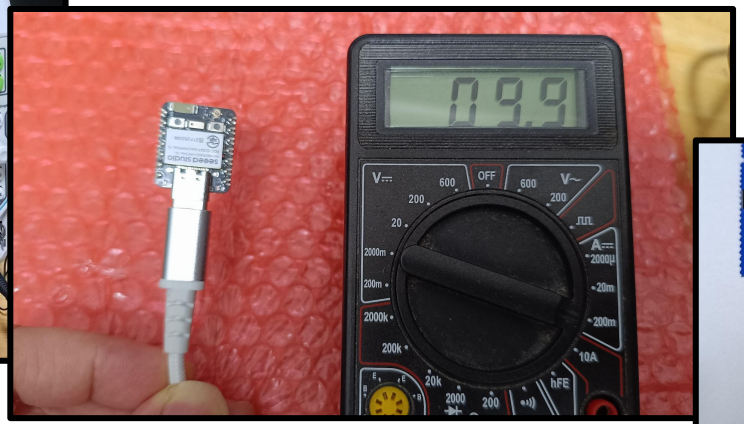
ミソジ
2026/3/27
Zephyr Project meetup: Nagoya, Japan
#ZephyrRTOS

自己紹介

ハードウェアのエンジニアで、趣味でブログとか書いてます

名前:　ミソジ　@misoji engineer
ブログ: エンジニアの電気屋さん(https://misoji-engineer.com/)





アジェンダ

Zephyr(RTOS)でコア間通信をしてみた話

- このテーマを選んだ理由
- Zephyr(RTOS)のトレンド・強いところ
- ARMとRISC-Vでコア間通信したら、面白いのでは？
- 実装+テストしてみた

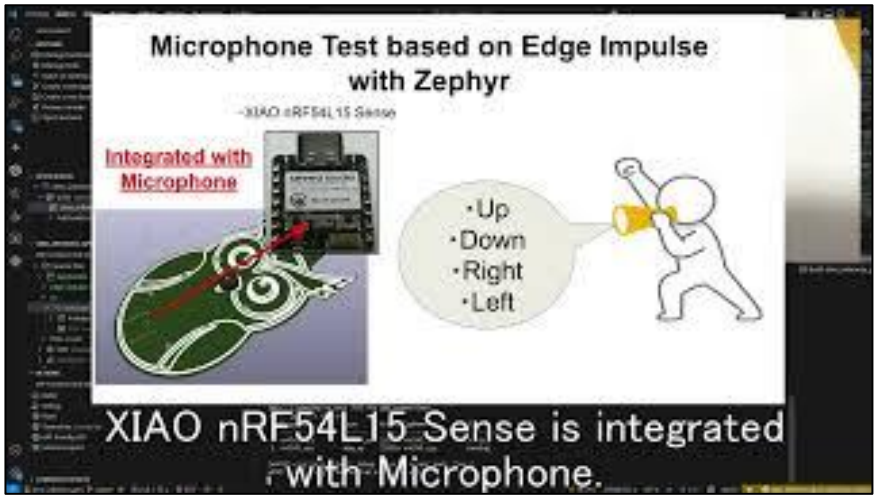
このテーマを選んだ理由

Edge AI

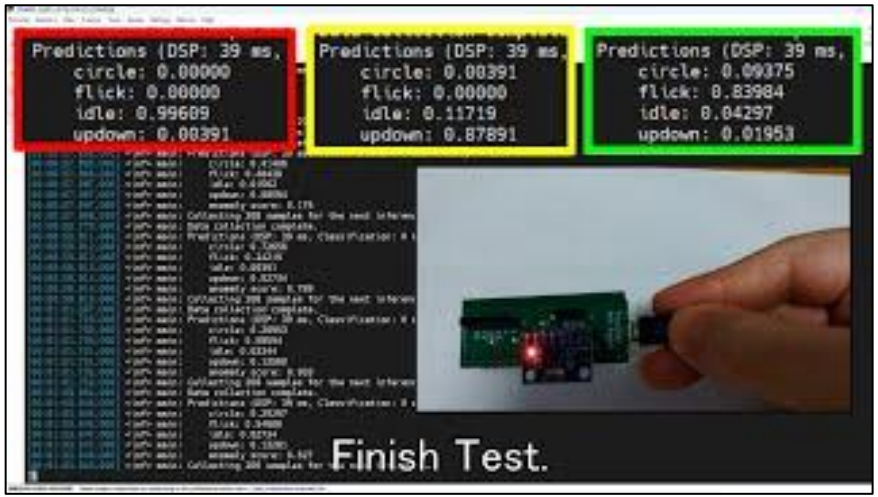
去年はZephyr +Edge AIを多く実装してました

■コンテスト、勉強会、カンファレンスでも良く使っていました
（見栄え良い ＋トレンド性ある ＋　分かりやすい）

・XIAO nRF54L15 Senseでの音声認識



・ラズパイPico2Wでのジェスチャー認識



とあるカンファレンスで...

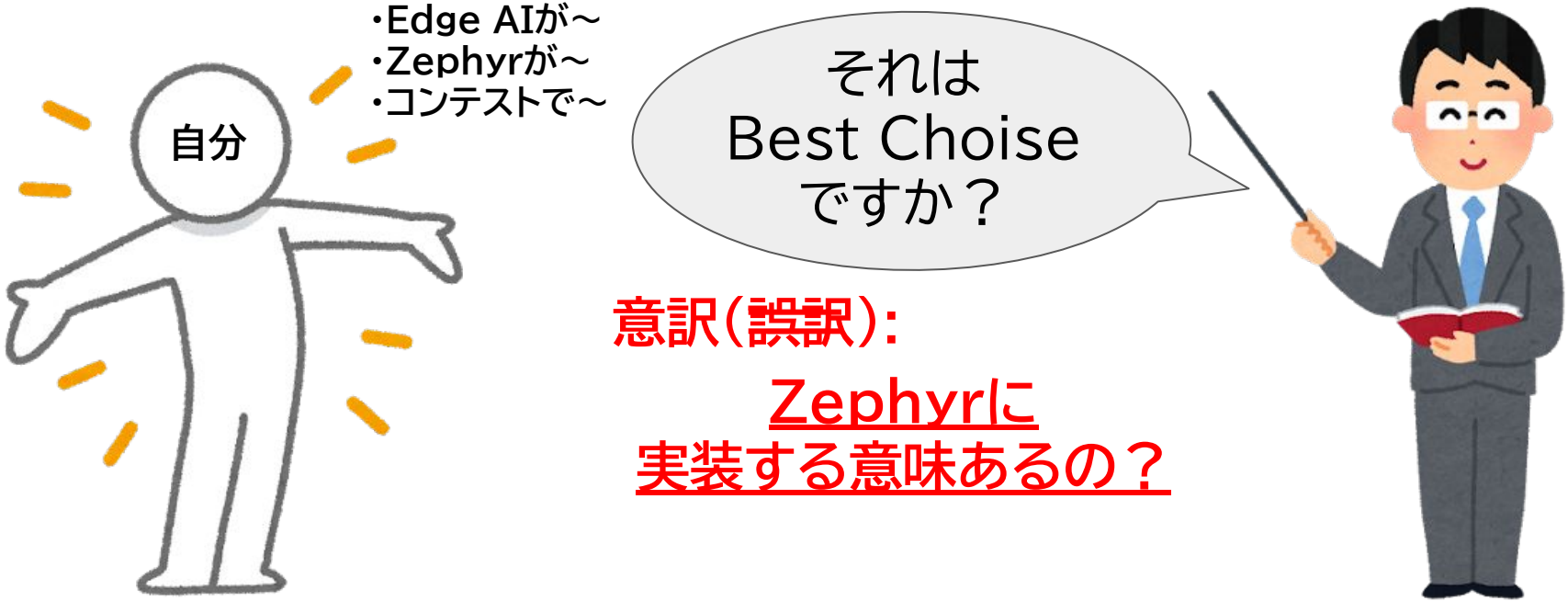
「鉄板」+「良い質問」が来ました

とあるカンファレンスで...

「鉄板」+「良い質問」が来ました

とあるカンファレンスで...

「鉄板」+「良い質問」が来ました



意訳(誤訳):

Zephyrに
実装する意味あるの？

致命傷で済んだ。

~~俺も思っていた~~。マジでクリティカル。

Zephyrである必要は？

正直言うと…無い。 OS無しでもEdge AI処理できる

もちろん軽量・省電力というEdge AIに有効なRTOSですが…
→無理して使う必要は無い。

Zephyr®

https://www.zephyrproject.org/

去年参加したEdge AIのコンペでも
自分以外、ほぼArduino(OS無し)で実装

2025年8月～9月
Edge AI Earth Guardians
https://www.hackster.io/contests/earthguardians

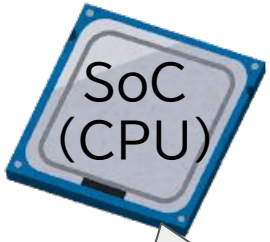
ただ、このままでは悔しいので...

Zephyr + Edge AIが有効であるケースを、考えてみる。

Zephyr(RTOS)の
トレンド・強いところ

Zephyrの強み

SoC内のコア(OS)間の通信・仮想化もトレンドの一つ。

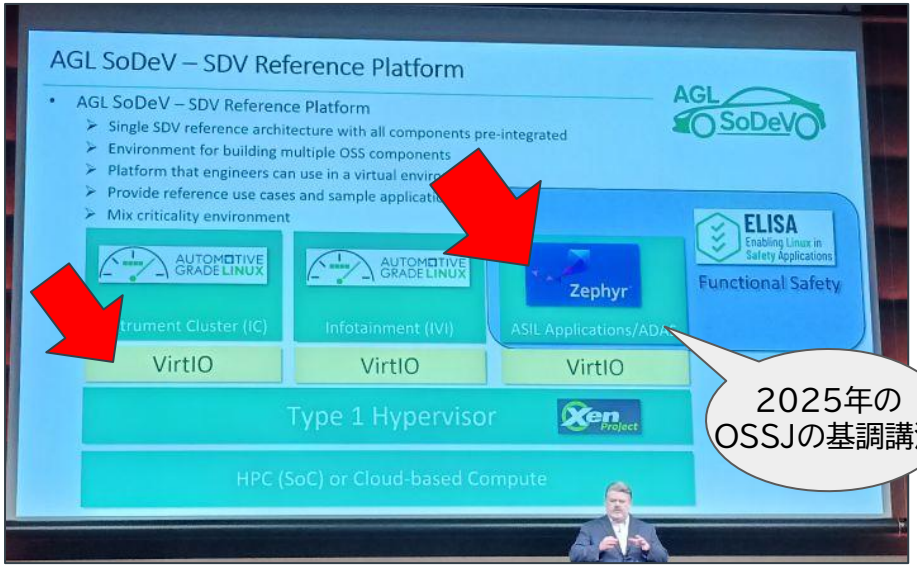


最近はSoC内の
コアの種類・数が多い
（使い切れない…）

ヘテロジニアス
（異種混合）化

サブ_Mコア（RTOS）	メイン_Aコア（Linux）
NPU（AI処理）	機能安全_Mコア（RTOS）

車載系のプラットフォーム例
コア（OS）間の違いを越えた、I/O仮想化（VirtIO）



https://www.youtube.com/watch?v=QjiWMYTrMQU

Zephyrのコア間通信のサポート例

色々あるが、IPCが一番ハード寄りで自分好み

Geminiに聞いてみた

一番分かりやすい例えは、**「物流（ロジスティクス）」**です。

- IPC ＝ 道路や橋（物理的なインフラ）
- VirtIO ＝ 交通ルールとトラックの規格（運び方の作法）
- OpenAMP ＝ 運送会社・物流システム全体（管理と運用）

IPCとEdgeAIを
絡めればメリットあるのでは…

項目	IPC（Inter-Processor Communication）	VirtIO（Virtual I/O）	OpenAMP（Open Asymmetric Multi-Processing）
定義	コア間通信の**総称**	I/O仮想化の**標準規格**	マルチコア連携の**フレームワーク**
レイヤー	物理層・データリンク層（ハード寄り）	トランスポート層（抽象化）	アプリケーション・管理層（ソフト全体）
主な役割	データの通り道を確保する	メモリの使い方の作法を共通化する	コアの起動・停止やメッセージを管理する
具体例	共有RAM、メールボックス、割り込み	vring（リングバッファ）	RemoteProc, RPMsg

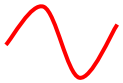
IPC(Inter-Processor Communication)

IPCで「軽量+高速」と「複雑+重い」タスクを分ける

高速な信号のため、
他処理の影響を小さくしたいケース

PWM

ADC
(Analog-
Digital)



ARMコア

・高速モータ制御
・高速ADC
(サンプリング)

軽量+高速 タスク

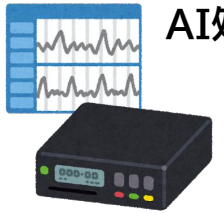
IPC

・共有RAM
・メールボックス
・割り込み

ARMコア

・Edge AI
・ネットワーク

複雑+重い タスク



AI処理

WIFI,LTE,
Bluetooth

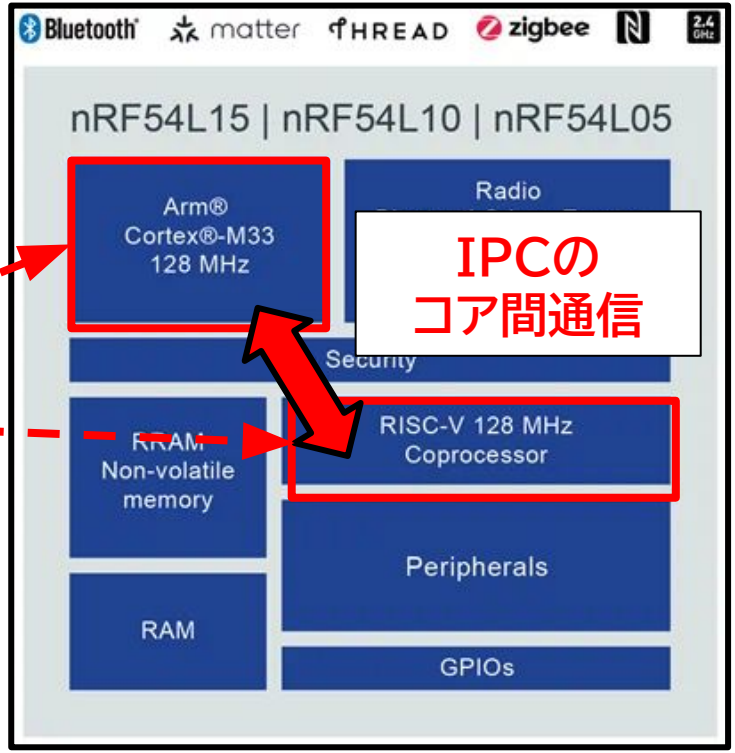
ちょうど良いネタ(SoC)がある

Nordic nRF54L15は「ARM」と「RISC-V」の2つのコア

■Nordic Wireless SOC nRF54L15

ARM_M33

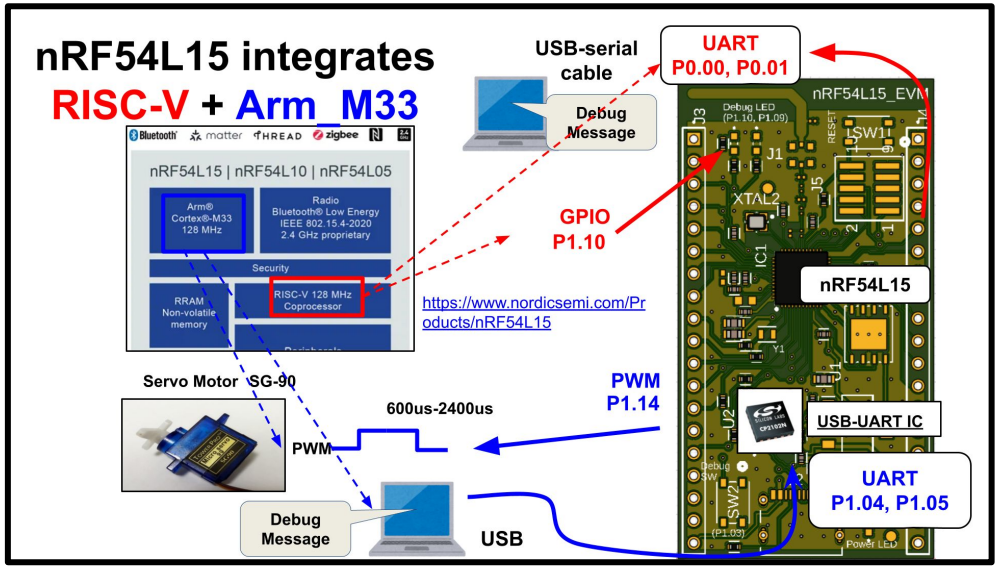
RISC-V

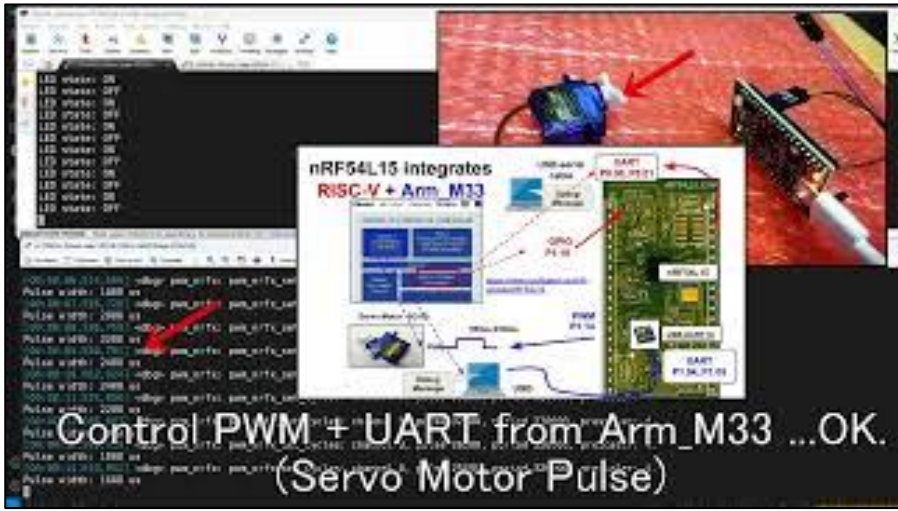


https://www.nordicsemi.com/Products/nRF54L15

ARMもRISC-Vも単独ではテスト済

まだ、IPCのコア間通信はテストしていなかった。





■去年nRF54L15のカスタムボードで
RISC-VとARM_M33の簡単な(GPIO・UART・PWM)動作確認はしていた

プロジェクトURL: https://www.hackster.io/iotengineer22/maker-s-nrf54l15-debug-board-1a6a88

Zephyrの良さが活きる。

RTOSとして、「軽量」かつ「多くのアーキテクチャ」に対応

PWM

ADC
(Analog-
Digital)

RISC-Vコア

・高速モータ制御
・高速ADC
（サンプリング）

軽量＋高速 タスク

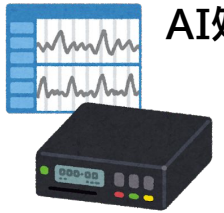
IPC

・共有RAM
・メールボックス
・割り込み

ARMコア

・Edge AI
・ネットワーク

複雑＋重い タスク

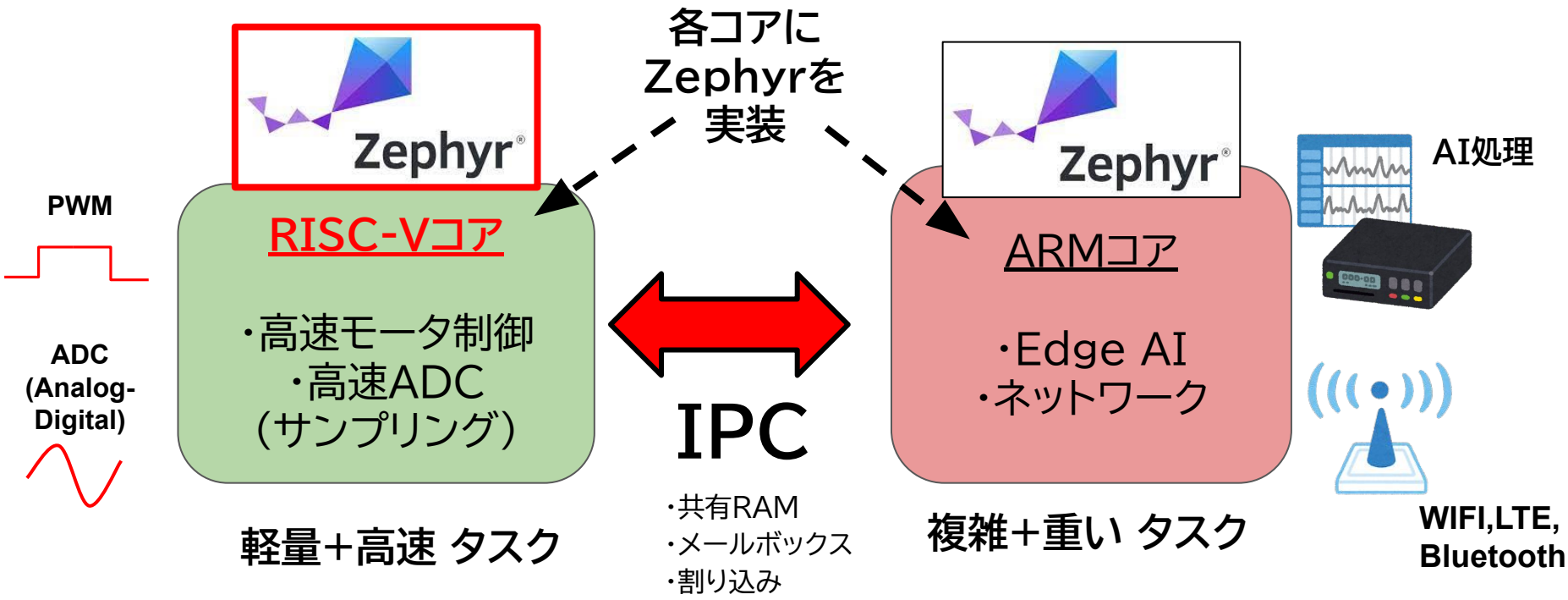


AI処理

WIFI,LTE,
Bluetooth

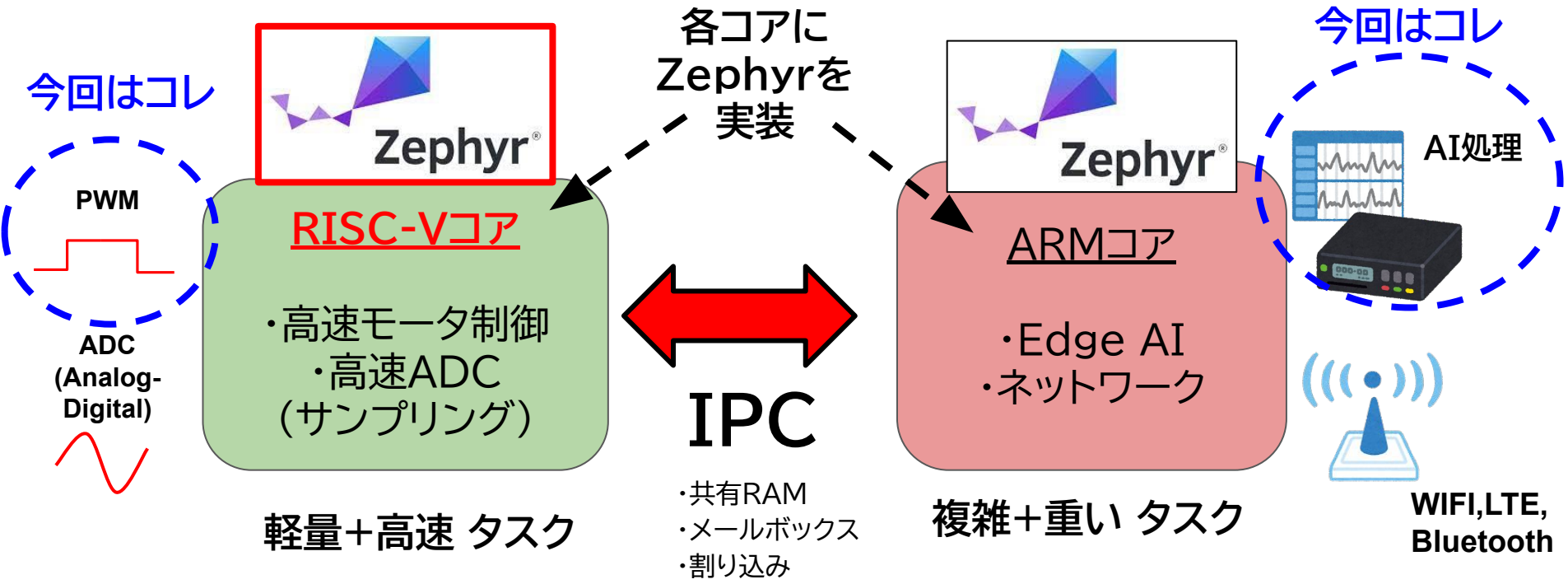
Zephyrの良さが活きる。

RTOSとして、「軽量」かつ「多くのアーキテクチャ」に対応



Zephyrの良さが活きる。

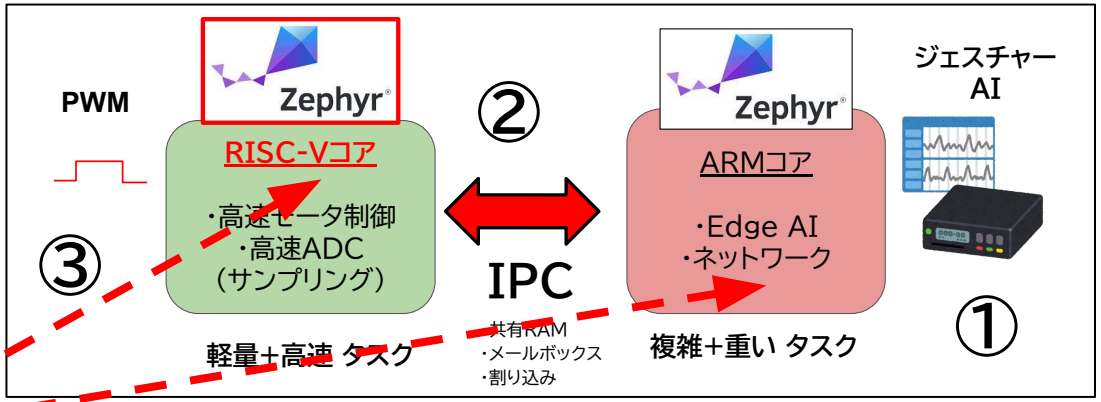
RTOSとして、「軽量」かつ「多くのアーキテクチャ」に対応



テスト実装してみる

シンプルな内容でテスト実装してみた。

■Nordic Wireless
SOC nRF54L15



①ARM側でジェスチャーAI処理
②IPCでデータ共有　*項目とスコア値のみ
③RISC-V側のPWMを変更

実装+テストしてみた

nRF54L15 + IPC

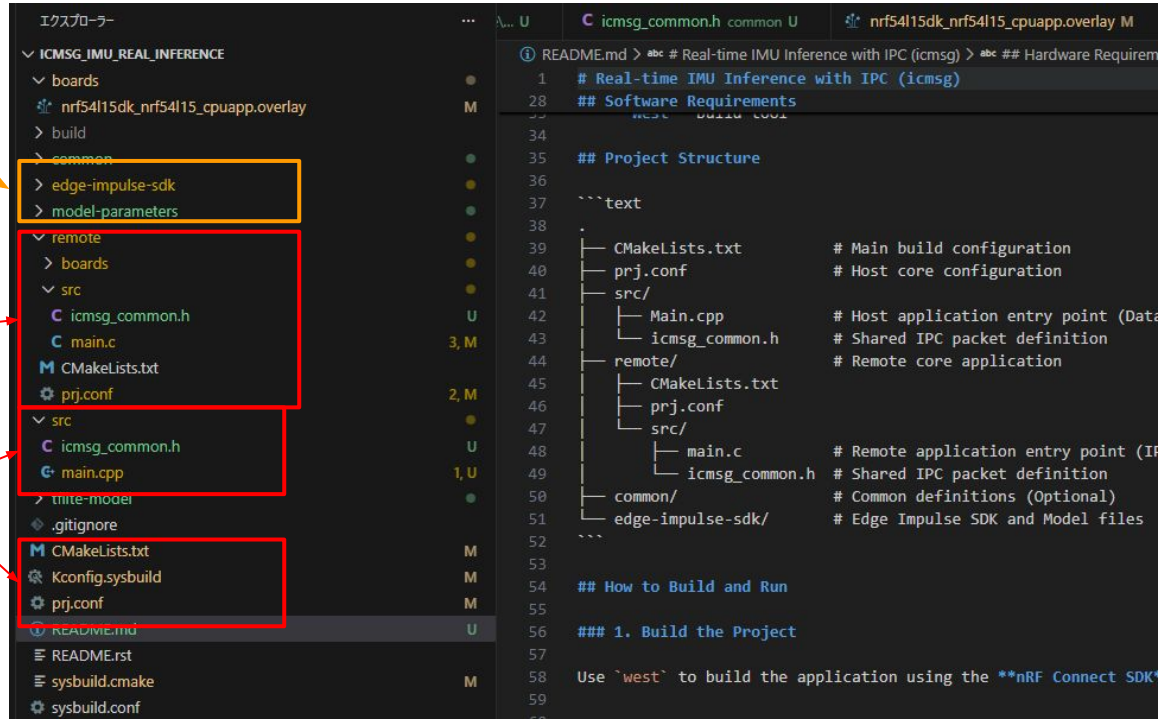
Zephyrのサンプルを調整 ＋　上手いことEdge AI処理を結合

*AIモデルは従来通り
Edge ImpulseのC++出力
（今回は説明省略）

①remoteフォルダにRISC-V側の
main.c、Makelist、prj.conf

②通常通りARM側の
main.c、Makelist、prj.conf

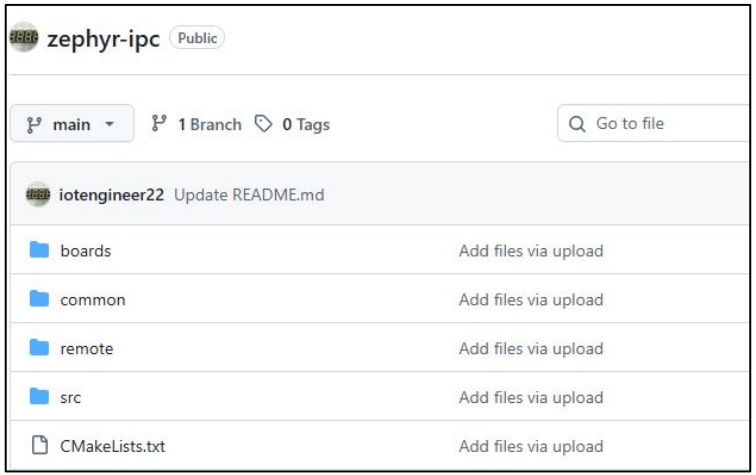
③NordicのRISC-V用の
スニペット（nordic-flpr）を使用



GitHubのリンク先

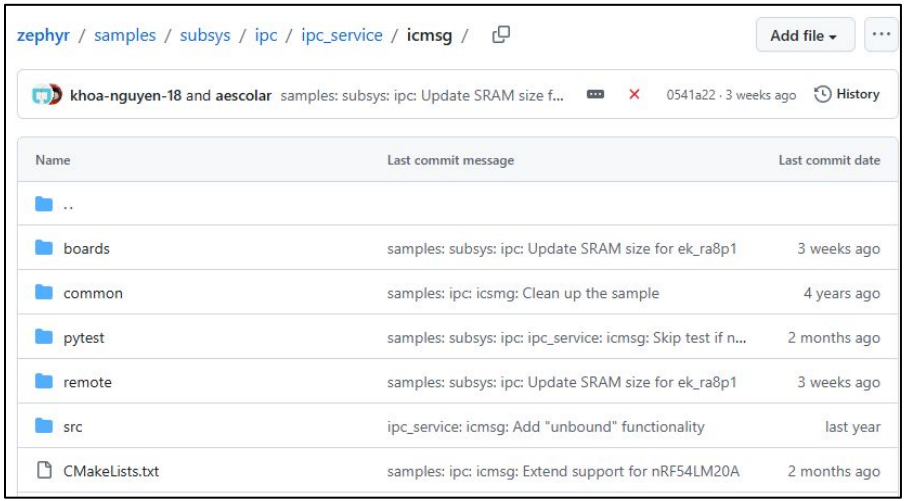
Zephyr公式サンプルの方を見てね。

~~(雑な)~~自分のテストプログラム



https://github.com/iotengineer22/zephyr-ipc

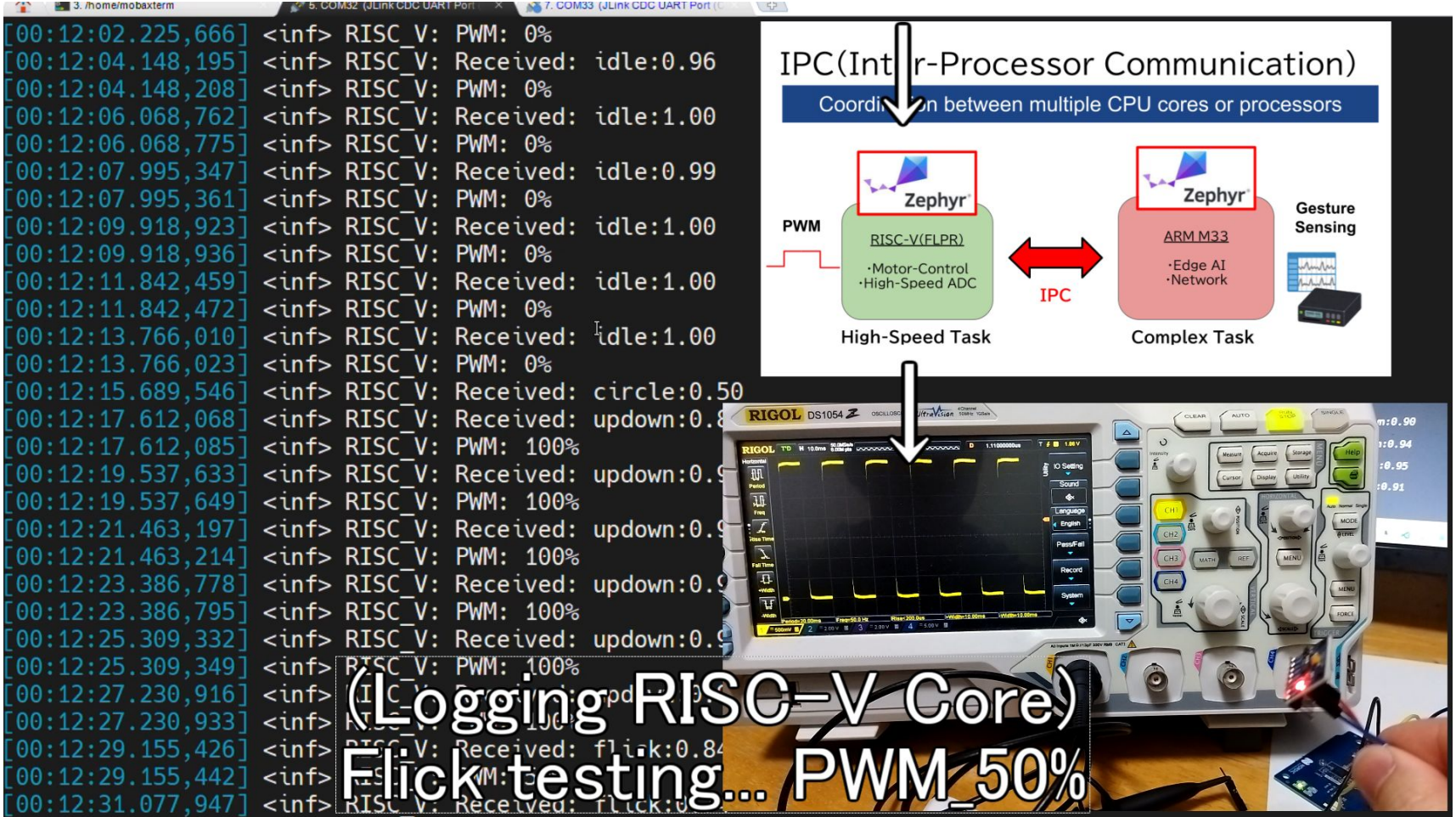
Zephyr公式のIPCのサンプル



https://github.com/zephyrproject-rtos/zephyr/tree/main/samples/subsys/ipc/ipc_service/icmsg

IPCデモ動画(nRF54L15-DK)

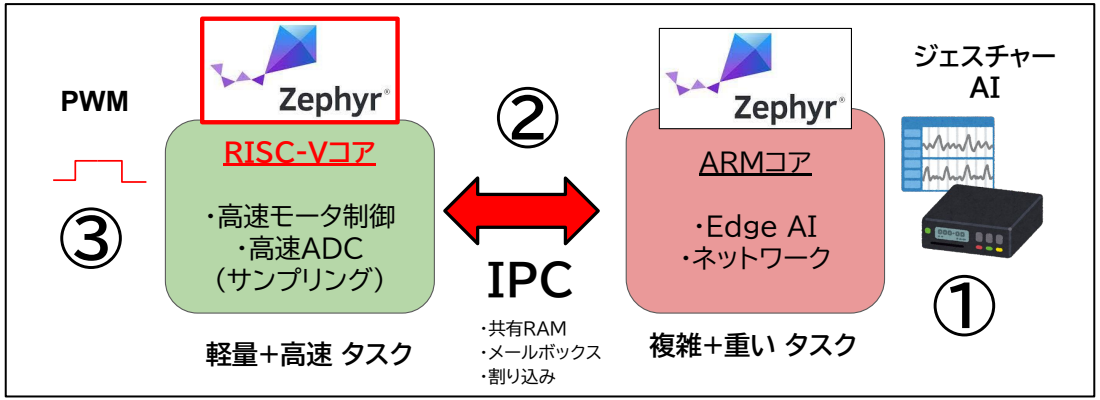
https://youtu.be/ACCoVr_i27I



IPCでテスト成功

ログ見ても、IPCでコア間のデータ転送→OK

①ARM側でジェスチャーAI処理
②IPCでデータ共有
③RISC-V側のPWMを変更




```
[00:12:42.613,297] <inf> RISC_V: Received: updown:0.91
[00:12:42.613,313] <inf> RISC_V: PWM: 100%
[00:12:44.539,900] <inf> RISC_V: Received: updown:0.80
[00:12:44.539,917] <inf> RISC_V: PWM: 100%
[00:12:46.464,435] <inf> RISC_V: Received: idle:0.91
[00:12:46.464,449] <inf> RISC_V: PWM: 0%
[00:12:48.389,982] <inf> RISC_V: Received: idle:0.99
[00:12:48.389,996] <inf> RISC_V: PWM: 0%
```

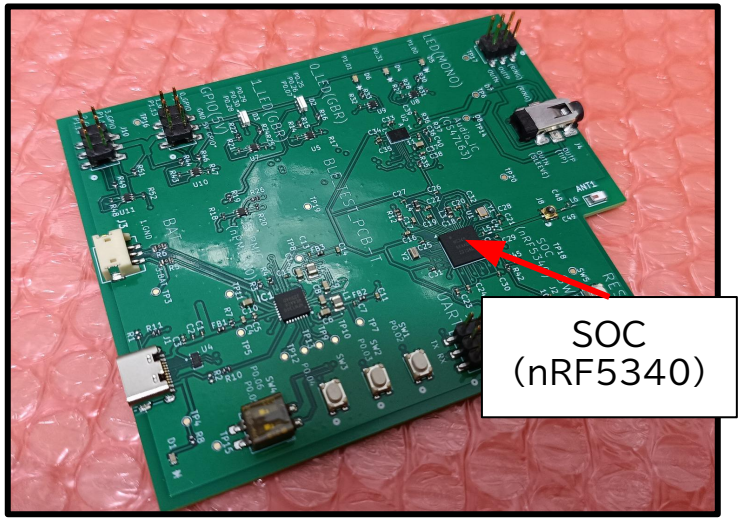
```
[00:12:44.539,739] <inf> host: Predictions (DSP: 40 ms, Classificati
[00:12:44.539,764] <inf> host:     circle: 0.00391
[00:12:44.539,776] <inf> host:     flick: 0.00391
[00:12:44.539,787] <inf> host:     idle: 0.19531
[00:12:44.539,799] <inf> host:     updown: 0.79687
[00:12:44.539,899] <inf> host: Sent inference result: updown:0.80
[00:12:46.464,274] <inf> host: Predictions (DSP: 39 ms, Classificati
[00:12:46.464,299] <inf> host:     circle: 0.00781
[00:12:46.464,311] <inf> host:     flick: 0.02344
[00:12:46.464,323] <inf> host:     idle: 0.91016
[00:12:46.464,335] <inf> host:     updown: 0.05469
[00:12:46.464,435] <inf> host: Sent inference result: idle:0.91
```

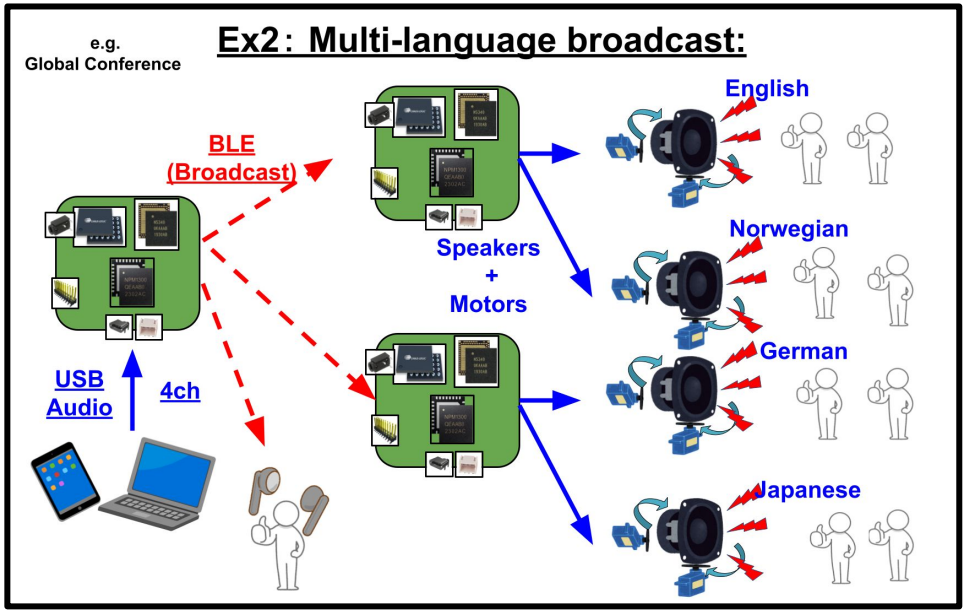
他のIPC応用例

実は知らずにIPC使っていたケース

Bluetooth LE Audio対応のトランスミッター・スピーカー

*BLE…Bluetooth Low energy



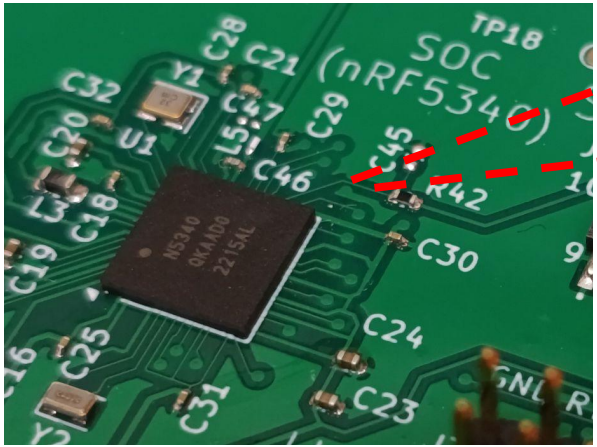


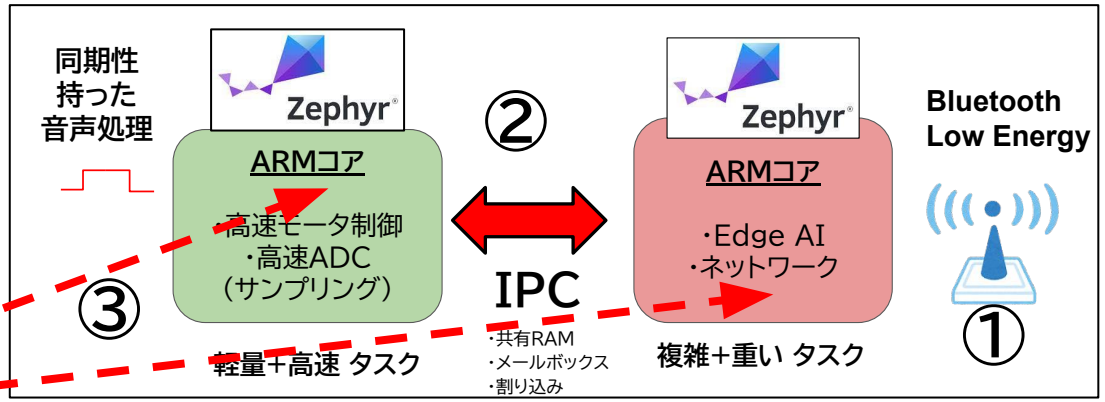
去年BluetoothのAudio開発コンテスト
プロジェクトURL:
https://www.hackster.io/iotengineer22/adaptive-directional-ble-audio-speaker-2d892d

nRF5340(ARM_M33 のデュアル + IPC)

「ネットワーク処理コア」と「音声処理のコア」を分けていた

■Nordic Wireless
SOC nRF5340
*ARM M33が2個内蔵



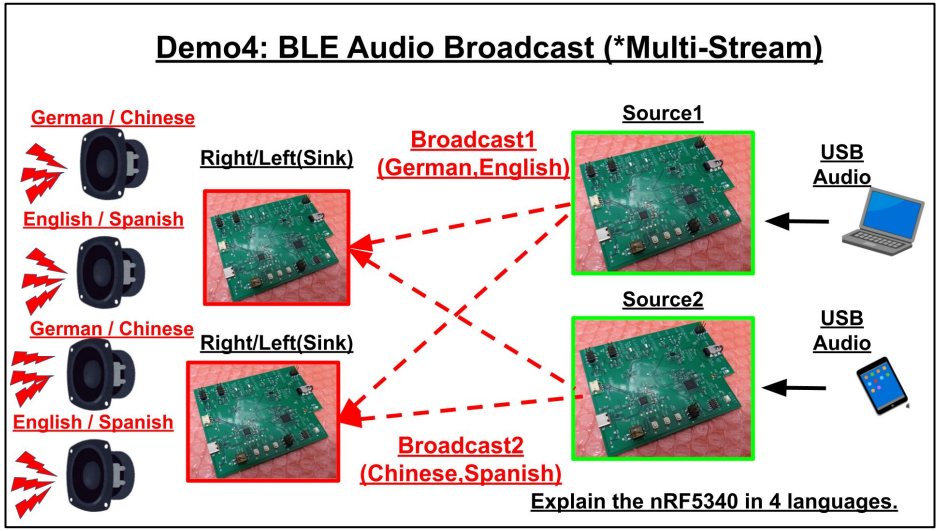


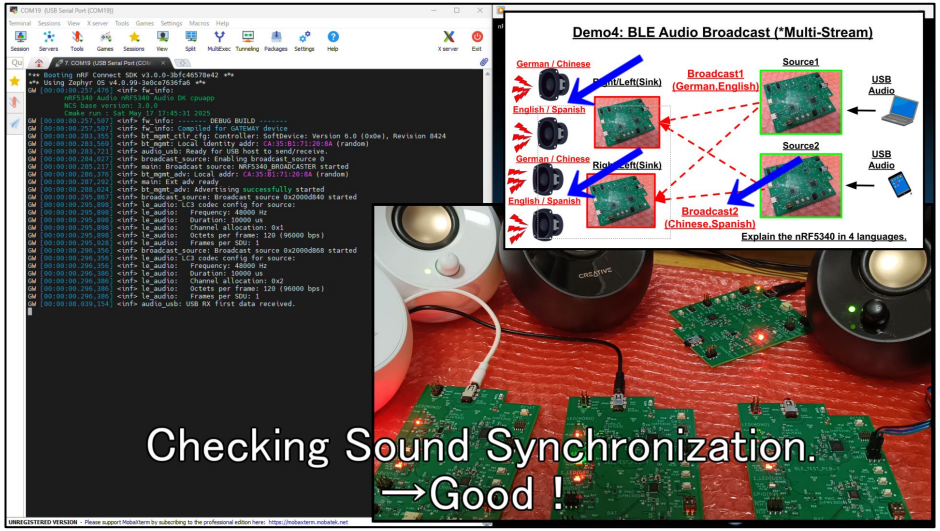
①ARM側でBluetooth LE処理
②IPCで音声データ共有
③ARM側で同期性持った音声出力

Bluetooth LE Audio

高い同期精度が必要　→デュアルコアでの分散処理 + IPC

■今までのBluetoothには無い機能…Unicast/Broadcastができる
→(同期合わせた)特定オーディオの一斉送信





まとめ

まとめ

Zephyr(RTOS)で コア間通信(IPC)を実装+遊べた！

・Zephyrがコア間通信を強くサポートされていたのを学べた
（複数/異種のコアを持つSoCに対応）

・Zephyrが様々なアーキテクチャに対応
（ARM+RISC-Vのようなコア間でも通信できた）

・分散処理したり、厳密なRTOS制御したい場合に使えそう
（EdgeAIや厳密なネットワーク同期制御など）

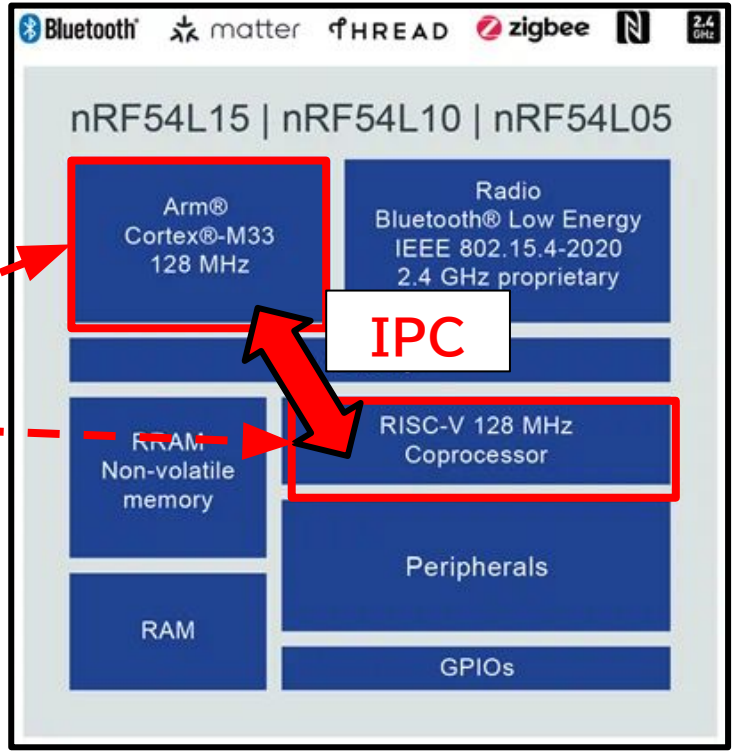
IPC(Inter-Processor Communication)

Coordination between multiple CPU cores or processors

■Nordic Wireless SOC nRF54L15

ARM_M33

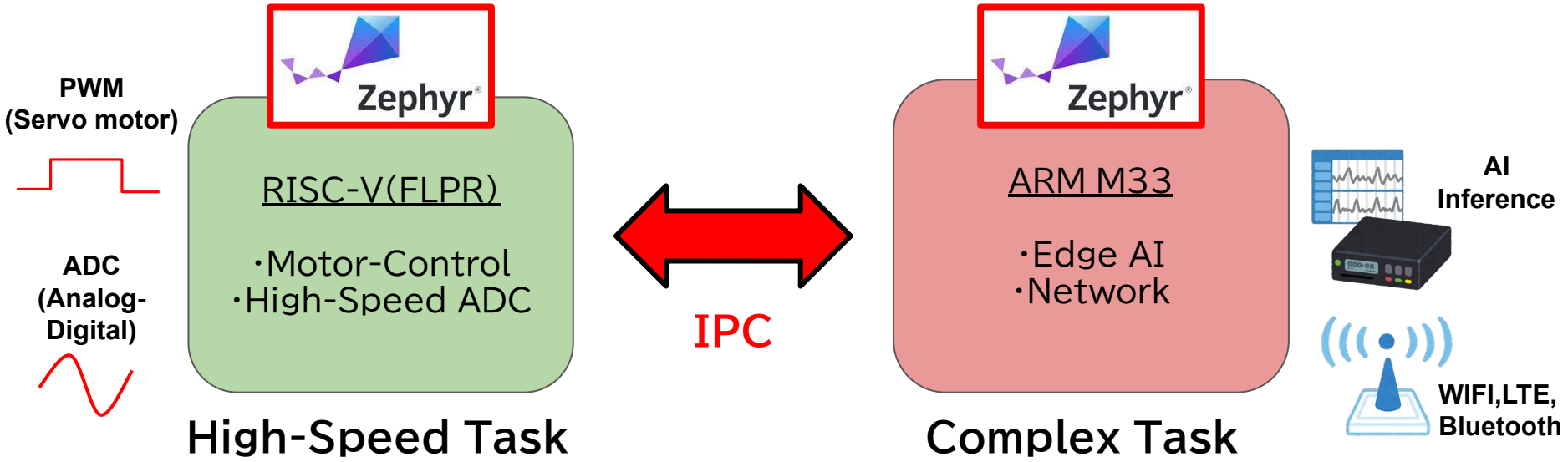
RISC-V



https://www.nordicsemi.com/Products/nRF54L15

IPC(Inter-Processor Communication)

Coordination between multiple CPU cores or processors



IPC(Inter-Processor Communication)

Coordination between multiple CPU cores or processors