マルチコアソフトウェア活用戦略

イー・フォース株式会社　横田　敬久

会社紹介

- **本社　　：東京**
- **開発拠点　　：東京 / インド**
- **事業内容**
  - Embedded事業
    - RTOSやミドルウェアの開発・販売
  - IoT事業
    - IoTプラットフォーム「iot-mos」の開発・販売
    - コンサルタント
- **沿革**

2006　:設立
国内のOSベンダとして初めて、ArmのCortex®-M/Aに対応し
「µC3/Compact」、「µC3/Standard」をリリース
2008　:TCP/IPスタック「µNet3」を開発
2013　:マルチコアデバイスに対応した「µC3/Standard+M」をリリース
2015　:産業用イーサネットへの取り組みをスタート
2016　:無線LANモジュール向けのSWパッケージを開発
2017　:RTOSとLinuxを共存させるソリューションをリリース
2018　:IoTプラットフォーム「iot-mos」を発表
2019　:Arm Cortex®-M33(TrustZone)に正式対応

Embedded事業のプロダクトマップ

Embedded事業

RTOS

- µC3/Compact
- µC3/Standard
- µC3/Standard+M
- µC3/Standard+M with Hetero
- µC3 + Linux

ドライバ

- µC3-SDドライバ
- µC3-BSP
- JSL-ware (JSL)

ミドルウェア

Network

- µNet3
- µNet3-Professional
- µNet3-SSL
- µNet3-IPv6
- µNet3-PPP
- µNet3-websocket
- µNet3-IGMPv3
- µNet3-MQTT

Filesystem

- µC3-Filesystem
- Fugue (KSR)

(Flash Filesystem)

USB

- GRAPEWARE USB (GRAPE SYSTEMS)
- MatrixQuest USB (Uquest)

GUI

NEW

- Qt for MCUs (Qt)
- Storyboard (Uquest)
- GENWARE（ILC）

Industrial Ethernet

NEW

- JS-EtherNet/IP Adapter SDK on µNet3 (JSL)
- JS-PROFINET Device SDK on µNet3 (JSL)

開発環境

統合開発環境

- EWARM (IAR)
- Arm Development Studio(DTSインサイト)

デバッガ

- PALMiCE (Computex)
- advice (DTSインサイト)

静的解析ツール

- C-STAT (IAR)
- PGRelief (DTSインサイト)

動的解析ツール

- C-RUN (IAR)

プログラマ

- Flasher ARM (EmbiTek)

アジェンダ

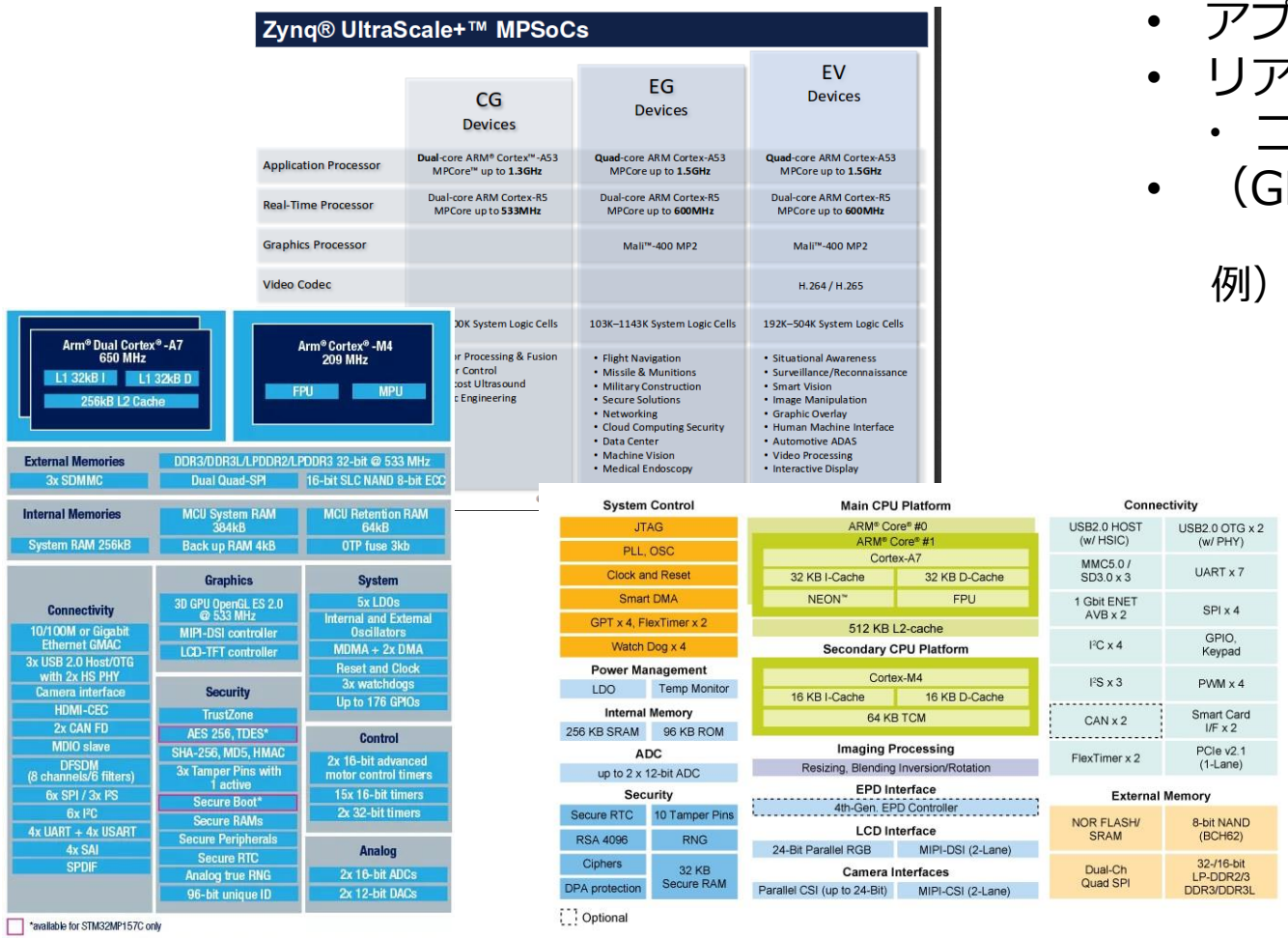
- イー・フォースの紹介
- マルチコア対応の必要性（なぜ積極的なマルチコアへの対応が必要なのか？）
- 具体的な採用事例（パターン）
  - マルチコア対応RTOSのユースケース
  - マルチOSのユースケース
- マルチコア/マルチOS製品の紹介
  - µC3/Standard+Mシリーズ
    - µC3/Standard+M
    - µC3/Standard+M with Hetero
  - µC3+Linux
    - µC3+Linuxの対応状況
    - µC3+Linuxとは
    - OpenAMPのバージョン遷移
    - 今後の展開の可能性

なぜ積極的なマルチコアへの対応が必要なのか？

eForce

さらに促進されるSoCのマルチコア化・ヘテロジニアスマルチコア化

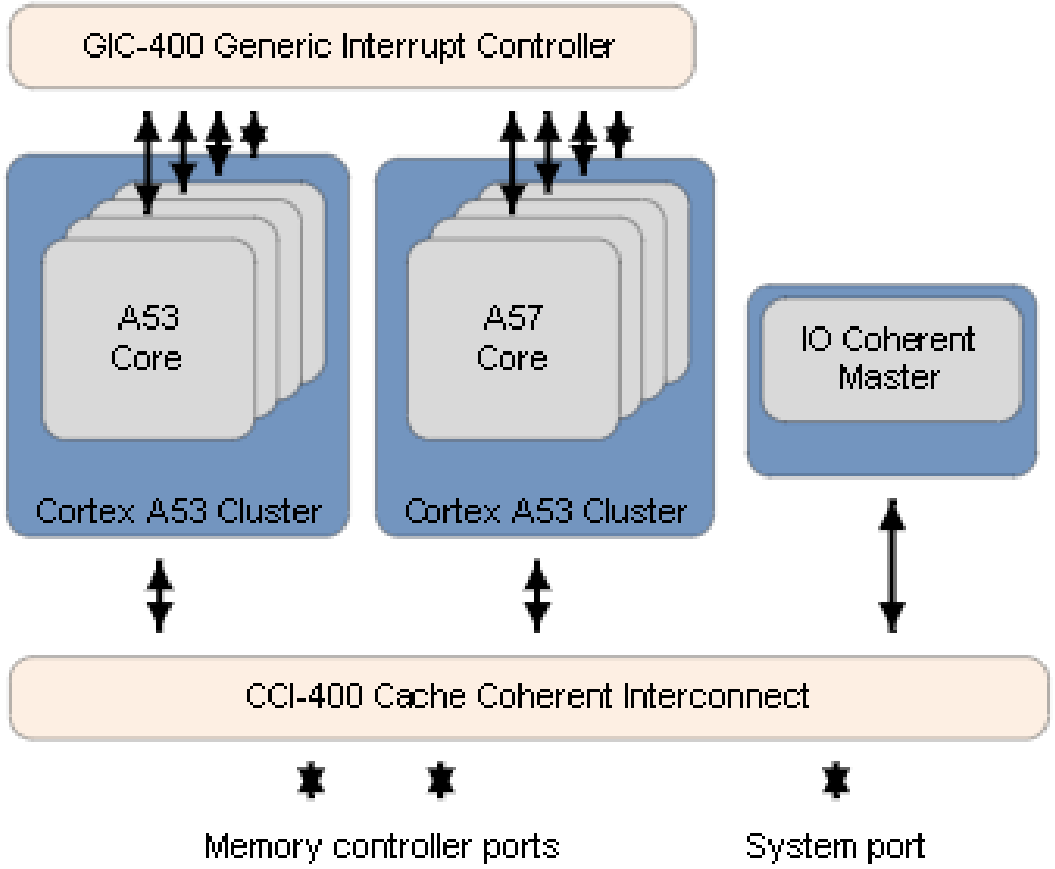
- **SoC内に全く違う目的のCPUコアを搭載**



- アプリケーションプロセッサ
- リアルタイムプロセッサ
  - コントローラプロセッサ
- （GPU・DSP・CODECなど・・・）

例）Xilinx Zynq Ultra Scale+MPSoC
STmicro STM32MP1
NXP i.MX7D

SoC内に同一命令セットヘテロジニアスマルチコア（big.LITTLE）を搭載



https://documentation-service.arm.com/static/5fbd26f271eff94ef49c7004?token=

- big.LITTLEアーキテクチャのコアが出現
  - big Core ・・・ Out-of-Orderのハイエンドコア
  - LITTLE Core ・・・ In-Orderのローエンドコア
- ハイエンドのSoC
  - さらにリアルタイムプロセッサなども含む
    複合的な構成も増えている

例）RENESASRZ/G2H
　　NXP i.MX8 Quad Plus

コア数の増加・構成の複雑化

プロセッサ名	コア数	コア構成
STM32H7	1～2	Cortex-M7 + Cortex-M4
NXP LPC4300	1～2	Cortex-M4 + Cortex-M0
STM 32MP1	2～3	Cortex-A7x2 + Cortex-M4
NXP i.MX7D	2～3	Cortex-A7x2 + Cortex-M4
NXP i.MX8M Mini	2～5	Cortex-A53 (x1~4) + Cortex-M4
NXP i.MX8M Nano	2～5	Cortex-A53 (x1~4) + Cortex-M7
TI AM57x	4	Cortex-A15x2 + Cortex-M4x2
TI AM65x	6	Cortex-A53 (x4) + Cortex-R5Fx2
TI AM64x	3	Cortex-A53 (x2) + Cortex-R5F
TI DRA829	6	Cortex-A72 (x2) + Cortex-R5Fx4
Zynq Ultrascale+MPSoC	4 or 6	Cortex-A53 (x2,4) + Cortex-R5Fx2
RENESAS RZ/G2M	6	Cortex-A57 (x2) + Cortex-A53 (x4)
RENESAS RZ/G2H	8	Cortex-A57 (x4) + Cortex-A53 (x4)

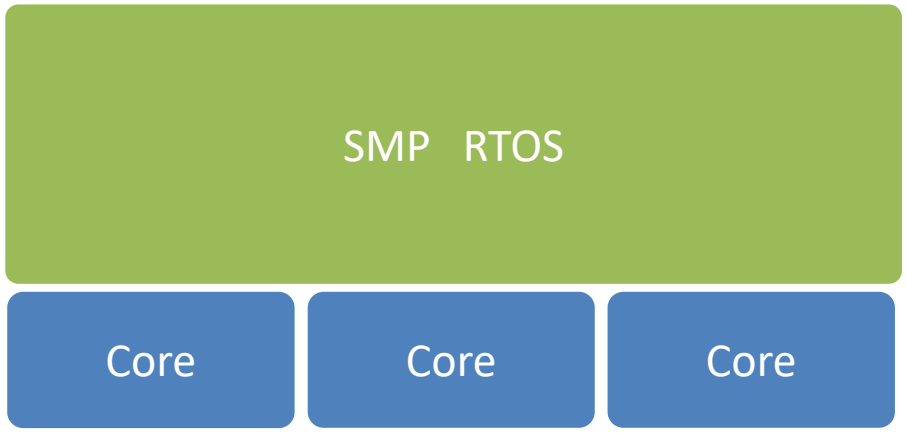
組込SoC向けにマルチコア・ヘテロジニアスマルチコアを活かすソリューションの提供が必須

具体的なユースケース

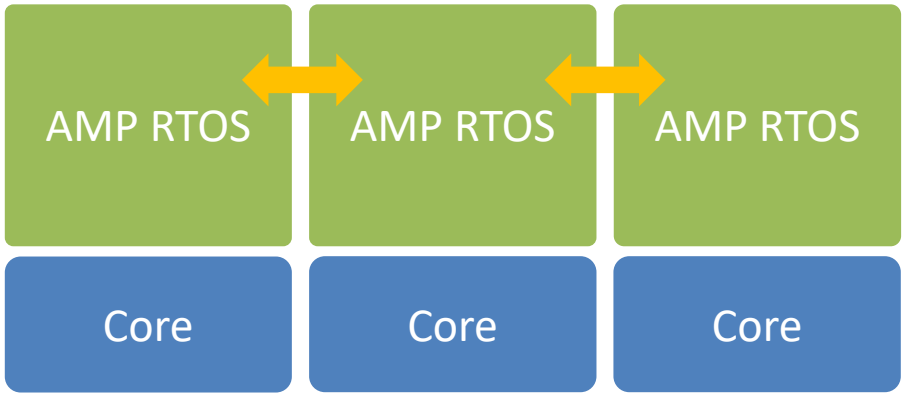
eForce

マルチコア対応のRTOS

SMP方式のRTOS



AMP方式のRTOS



- **SMP方式**
  - マルチコア全体で１つのOSとして動作
  - スケジューリングの理解は難しい
  - タスクなどのリソースの管理が簡単
- **AMP方式**
  - それぞれのコアで独立したOSが動作
    - シングルコアの資産をそのまま活用
  - どのコアでも同じRTOSが動作し活用できる。
  - スケジューリングの理解が簡単

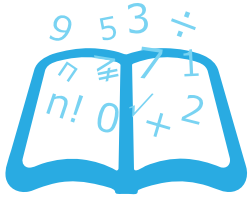
※この場合、ハードウェアのマルチプロセッサとしてのSMP・AMPとは違い、ソフトウェア的な概念

マルチコア対応のRTOSのユースケース

- **交通インフラでの課金システム**（インフラ系メーカ様）

Xilinx Zynq
Ultrascale+MPSoC

- **無線機**（コンシューマメーカ様）



Renesas RZ/G1E

- **車載機**（産業機器メーカ様）

Xilinx Zynq
Ultrascale+MPSoC

eForce

マルチコア対応のRTOSのユースケース

- **紙幣・硬貨鑑別機**

Xilinx Zynq Ultrascale+MPSoC
INTEL SoCFPGA Cyclone V/SoC等

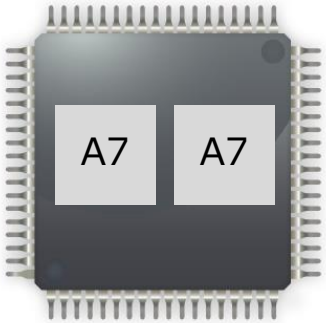
Linuxを用いたOSの共存

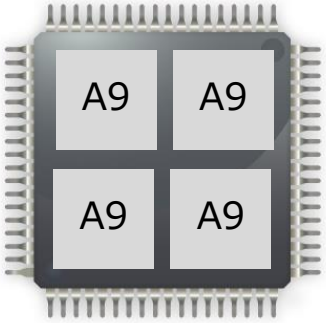
LinuxとRTOSとの共存　（マルチOS）

eForce

マルチコアプロセッサの種類

ホモジニアス・・・同一プロセッサ・コアによる構成





Renesas RZ/N1D
Renesas RZ/G1E, M
NXP i.MX 6 series
INTEL SoC FPGA series
Xilinx Zynq-7000

ヘテロジニアス・・・異なるプロセッサ・コアによる構成

Renesas RZ/N1S
Renesas RZ/G1H
NXP i.MX 6SoloX, 7
Xilinx Zynq Ultrascale+ MPSoC

マルチコアプロセッサの種類

ホモジニアス・・・同一プロセッサ・コアによる構成

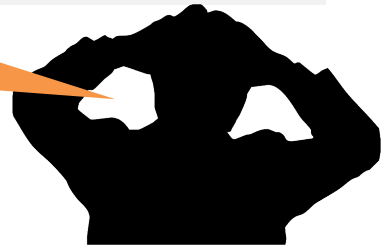
【ユーザ視点】
マルチコア・ヘテロコアを簡単に活用できるソリューションはないの？？

Renesas RZ/N1D
Renesas RZ/G1E, M
NXP i.MX 6 series
INTEL SoC FPGA series
Xilinx Zynq-7000

ヘテロジニアス・・・異なるプロセッサ・コアによる構成

【半導体ベンダ視点】
増えるバリエーションをどうやって活用して頂こう？？

Renesas RZ/N1S
Renesas RZ/G1H
NXP i.MX 6SoloX, 7
Xilinx Zynq Ultrascale+ MPSoC



なぜマルチOSの共存が必要？

LinuxとRTOSの共存の実現が必要

計測器メーカ様

リアルタイム性能を維持したまま、GUIを高機能化したい

産業機器メーカ様

Linuxの豊富なライブラリを活用したい。ただし、リアルタイム処理が必要な機能もある。

セキュリティメーカ様

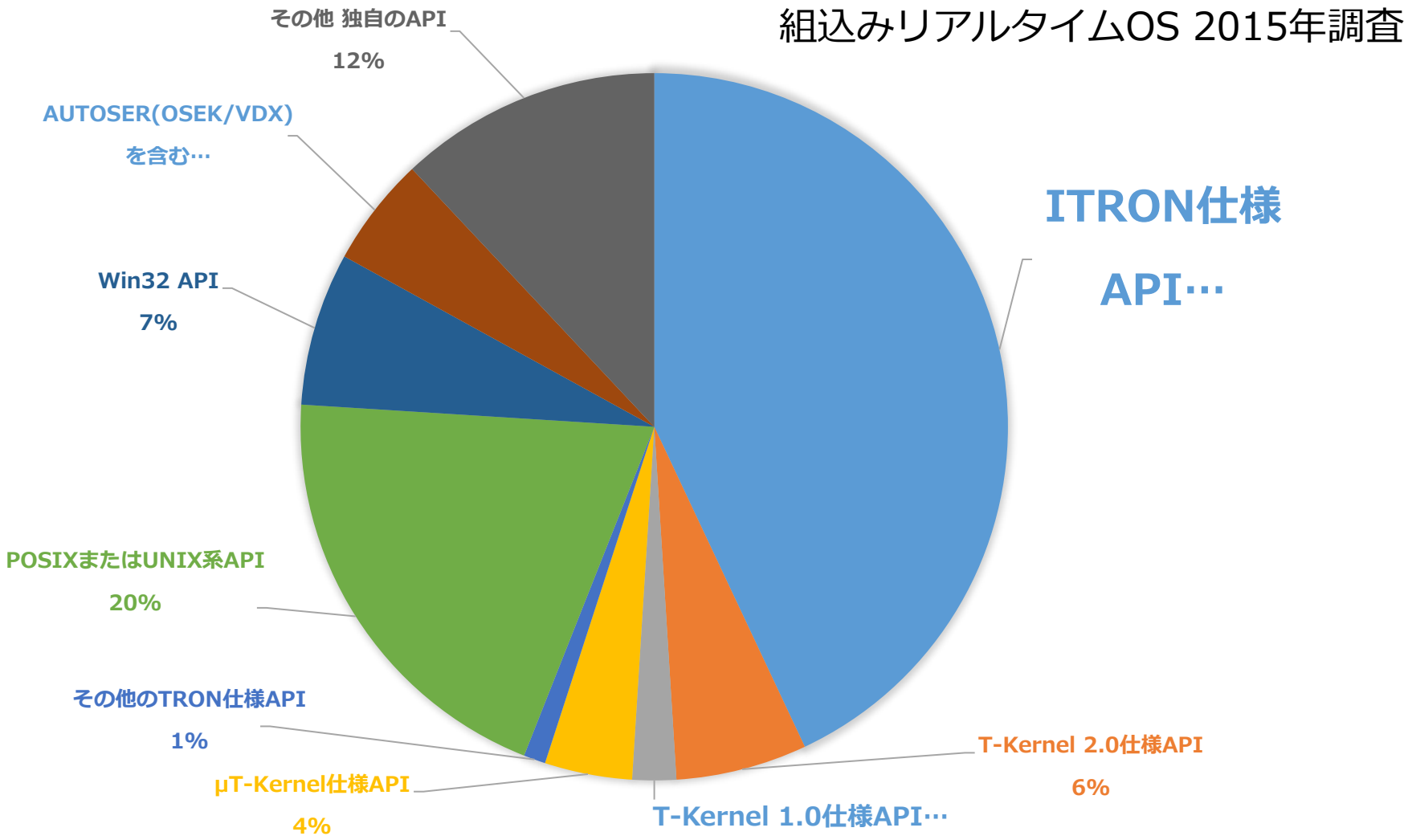
既存のRTOSのSW資産を活かしながら、ミドルウェアをLinux側に集約したい

コンシューマメーカ様

PC（Linux）で開発していたものを、高性能化・低価格化してきたプロセッサで実現したい。

eForce

なぜLinuxとRTOSを共存させるのか？



なぜLinuxとRTOSを共存させるのか？

組込みリアルタイムOS 2015年調査

Linuxのメリット・デメリットは？

メリット

- オープンソース・無償
- ミドルウェアが豊富
- デバイスドライバが豊富（GPU, VPU, Codec, etc.）
- 開発環境が揃っている
- 豊富な情報が公開

デメリット

- リアルタイム性能
- 起動時間が遅い
- 無保証

eForce

マルチOSユースケース（採用事例をもとに）

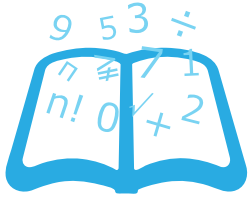
- **GUIの高機能化に活用**（計測器メーカ様・医療機器メーカ様・多機能プリンタメーカ様）

Graphic | Realtime proc

Linux | µC3

Xilinx Zynq
Ultrascale+MPSoC

- **機能配分の研究・開発に活用**（コンシューマメーカ様）

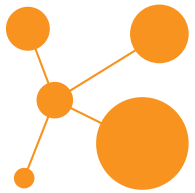


高負荷演算 | Realtime proc

Linux | µC3

Xilinx Zynq
Ultrascale+MPSoC

- **機器のIoT化に活用**（産業機器メーカ様）

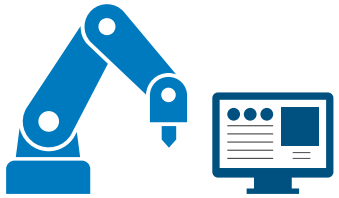


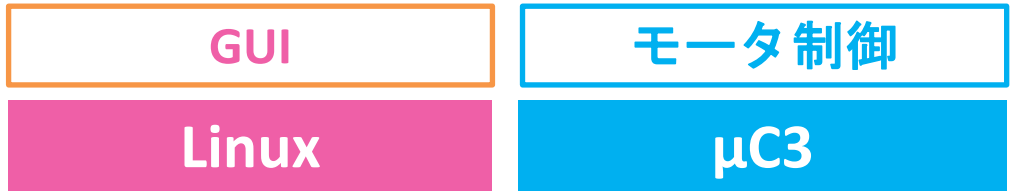
Ethernet | EtherCAT

Linux | µC3

マルチOSユースケース（採用事例をもとに）

- **1CHIP化（工作機械メーカ様）**





NXP i.MX8

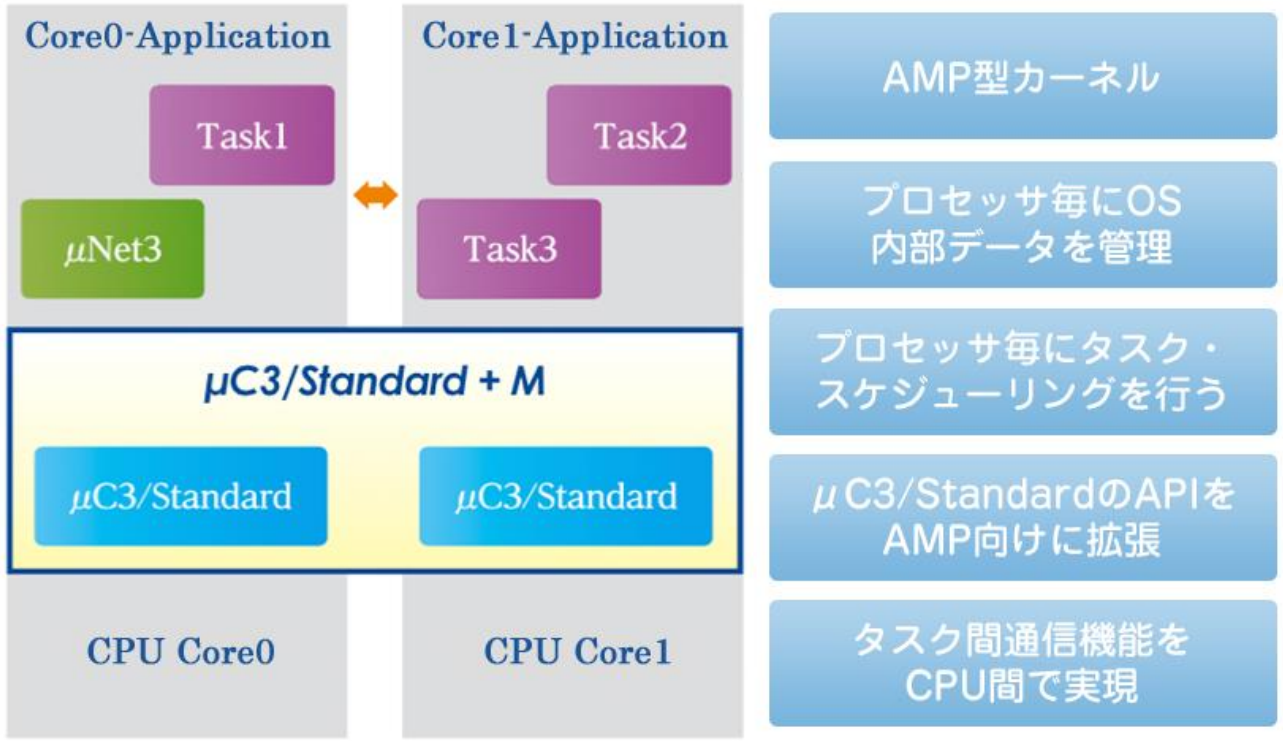
マルチコア対応製品の紹介

eForce

µC3/Standard+M

µC3
micro c cube
Standard+M

- µC3/Standardをマルチコア拡張したもの
- µITRON4.0仕様
- コア毎にµC3/Standardカーネルが動作(AMP方式)
- コア間はコア通信APIで行う



eForce

μC3/Standard+Mのコア間通信API

機能	システムコール名	説明
タスク管理機能	vact_tsk	タスクの起動
	ivact_tsk	
	vsta_tsk	タスクの起動（起動コード指定）
タスク付属同期機能	vwup_tsk	タスクの起床
	ivwup_tsk	
	vrel_wai	待ち状態の強制解除
	ivrel_wai	
セマフォ	vsig_sem	セマフォ資源の返却
	ivsig_sem	
	vpol_sem	セマフォ資源の獲得（ポーリング）

機能	システムコール名	説明
イベントフラグ	vset_flg	イベントフラグのセット
	ivset_flg	
	vclr_flg	イベントフラグのクリア
	vpol_flg	イベントフラグ待ち（ポーリング）
データキュー	vpsnd_dtq	データキューへの送信（ポーリング）
	ivpsnd_dtq	
	vfsnd_dtq	データキューへの強制送信
	ivfsnd_dtq	
	vprcv_dtq	データキューからの受信（ポーリング）
時間管理機能	ivsig_tim	タイムチックの供給
システム状態管理機能	vrot_rdq	タスクの優先順位の回転
	ivrot_rdq	
	get_cid	コアIDの取得

eForce

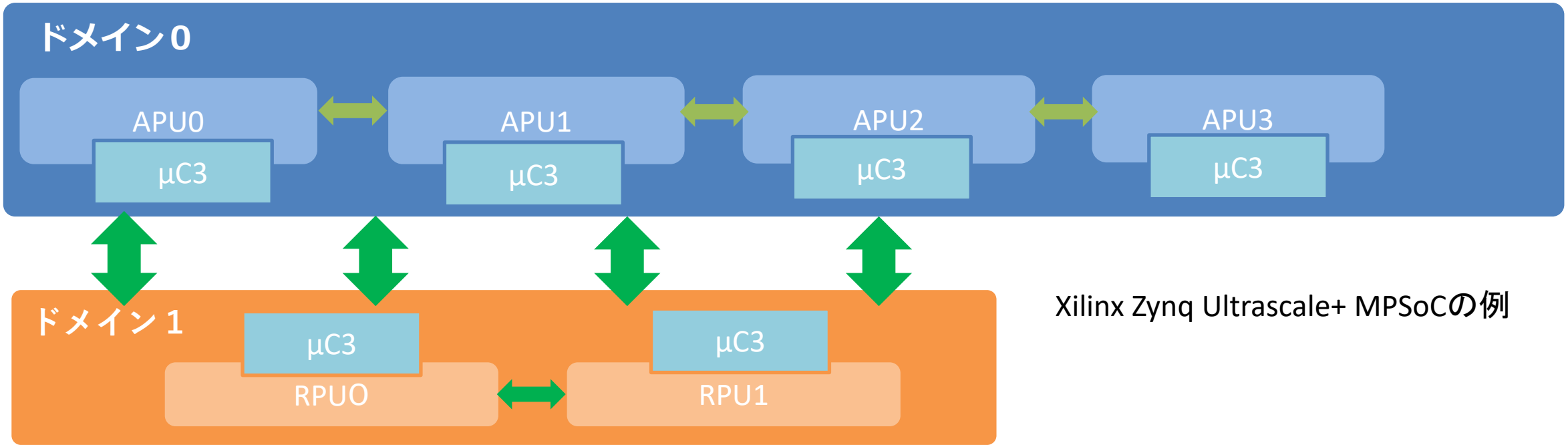
μC3/Standard+M対応SoC（代表的なもの）

プロセッサ名	コア数	コア構成
STMicro STM32H7	1～2	Cortex-M7 + Cortex-M4
STMicro STM32MP1	2～3	Cortex-A7x2
NXP i.MX7D	2～3	Cortex-A7x2 + Cortex-M4
NXP i.MX6	2～4	Cortex-A9x4
TI AM57x	4	Cortex-A15x2 + Cortex-M4x2
Zynq Ultrascale+MPSoC	4 or 6	Cortex-A53 (x2~4
Zynq Ultrascale+MPSoC	4 or 6	Cortex-R5Fx2
INTEL CycloneV/ArriaV SoC	2	Cortex-A9
INTEL Arria10 SoC	2	Cortex-A9
NXP i.MX7ULP	2	Cortex-A7 + Cortex-M4
RENESAS RZ G1M G1N	2	Cortex-A15x2
RENESAS RZ G1E G1H	2	Cortex-A7x2

eForce

µC3/Standard+M with Hetero

- **µC3/Standard+Mをヘテロジニアスマルチコア環境に対応**
- **コアのグループ（＝ドメイン）の概念を導入**
- **基本的な使用方法はµC3/Standard+Mと同じものを実現**



Xilinx Zynq Ultrascale+ MPSoCの例

eForce

μC3/Standard+M with Hetero対応SoC

対応デバイス	対応コア構成
Xilinx Zynq Ultrascale+MPSoC	Cortex-A53 x 4 + Cortex-R5 x 2
TI DRA829	Cortex-A72 (x2) + Cortex-R5Fx4
STMicro STM32MP1	Cortex-A7x2 + Cortex-M4

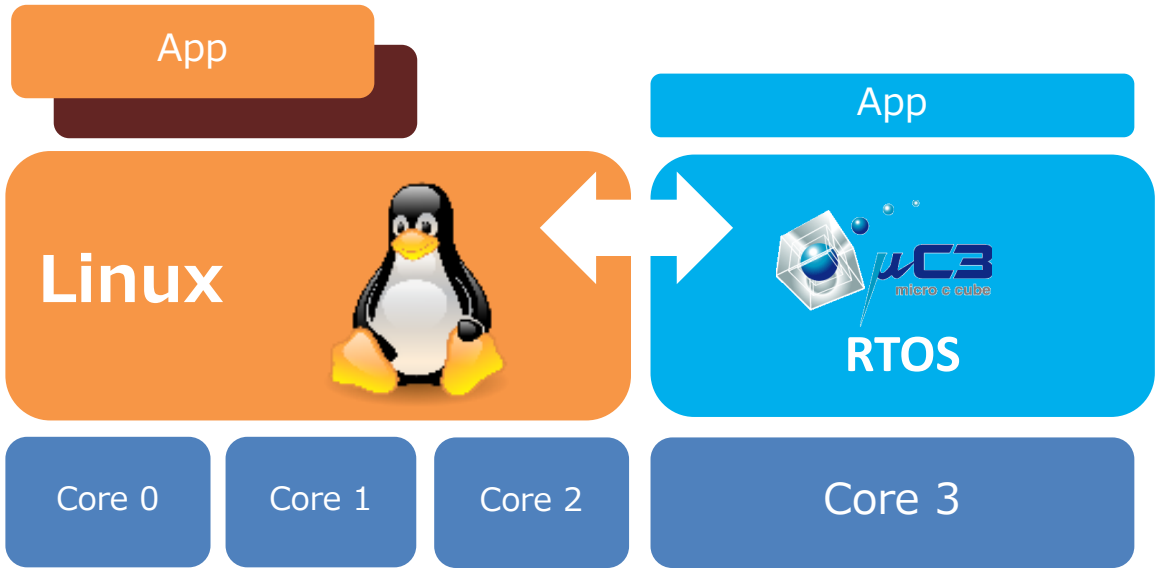
マルチOS対応製品の紹介

OpenAMP方式を採用した
「μC3+Linux」とは？

eForce

µC3+Linux

µC3 micro c cube +Linux



『LinuxとRTOSの共存ソリューション』

- µC3+LinuxはマルチコアCPUに
  LinuxとRTOSを共存させOS間の通信を可能に！
- OSの共存・通信にはOpenAMPを採用
  - Multicore Associationによる国際的な取り決め
- 共有メモリAPI（独自仕様）による
  大規模なメモリの共有を実現

ヘテロコアプロセッサにも対応可能

OpenAMP同士であれば
Linux以外のOSやベアメタルでも通信可能

eForce

µC3+Linuxの対応リスト(一部を抜粋)

対応デバイス	対応コア構成	CPUとOSの構成
Xilinx Zynq Ultrascale+MPSoC	Cortex-A53 x 4 + Cortex-R5F x 2	Cortex-A53 x 4 Linux Cortex-R5Fx2 µC3
	Cortex-A53 x 4	Cortex-A53 x 2~3 Linux Cortex-A53 x 1~2 (NS) µC3
TI AM64x	Cortex-A53 (x2)	Cortex-A53 x 1 Linux Cortex-A53 x 1 (NS) µC3
TI AM65x	Cortex-A53 (x4) + Cortex-R5F x 2	Cortex-A53 x 4 Linux Cortex-R5F µC3
STM32MP1	Cortex-A7x2 + Cortex-M4	Cortex-A7x2 Linux Cortex-M4 µC3
NXP i.MX8MNano	Cortex-A53 (x1~4) + Cortex-M7	Cortex-A53x4 Linux Cortex-M7 µC3
NXP i.MX8MMini, i.MX8MQ	Cortex-A53 (x1~4) + Cortex-M4	Cortex-A53x4 Linux Cortex-M4 µC3
Renesas RZ/G2M	Cortex-A53 (x1~4) + Cortex-M4	Cortex-A53x4 Linux Cortex-M4 µC3

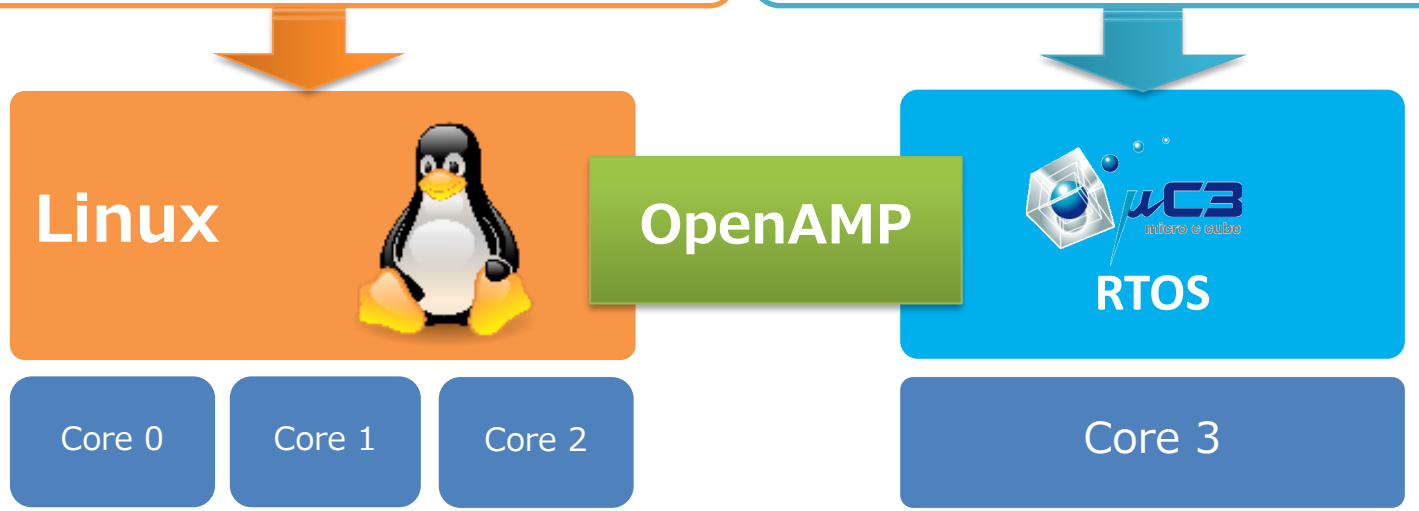
eForce

μC3+Linuxとは？

Linuxの豊富な資産を活用 x RTOSのリアルタイム性能・信頼性担保

オープンソース・無償
ミドルウェアが豊富
デバイスドライバが豊富
豊富な情報が公開

リアルタイム性能
信頼性の担保
高速起動



μC3+Linuxとは？　-OpenAMP-

OpenAMP　：正式名称「OPEN ASYMMETRIC MULTI PROCESSING」

マルチコアで各コアが連携できるようにコア間の通信や
リソースの管理を行うためのMulticore Association (MCA)で規定する標準規格です。

OpenAMPフレームワークが提供する主要なコンポーネント

remoteproc

マスタプロセッサにリモートプロセッサのライフサイクル管理機能を提供します
(ブート・シャットダウン・プログラムのロード等)

RPMsg

RPMsg APIは、AMPシステム内に存在する同種または異種のコア上で動作する、
独立したソフトウェアコンテキスト間のプロセッサ間通信(IPC)を可能とします。

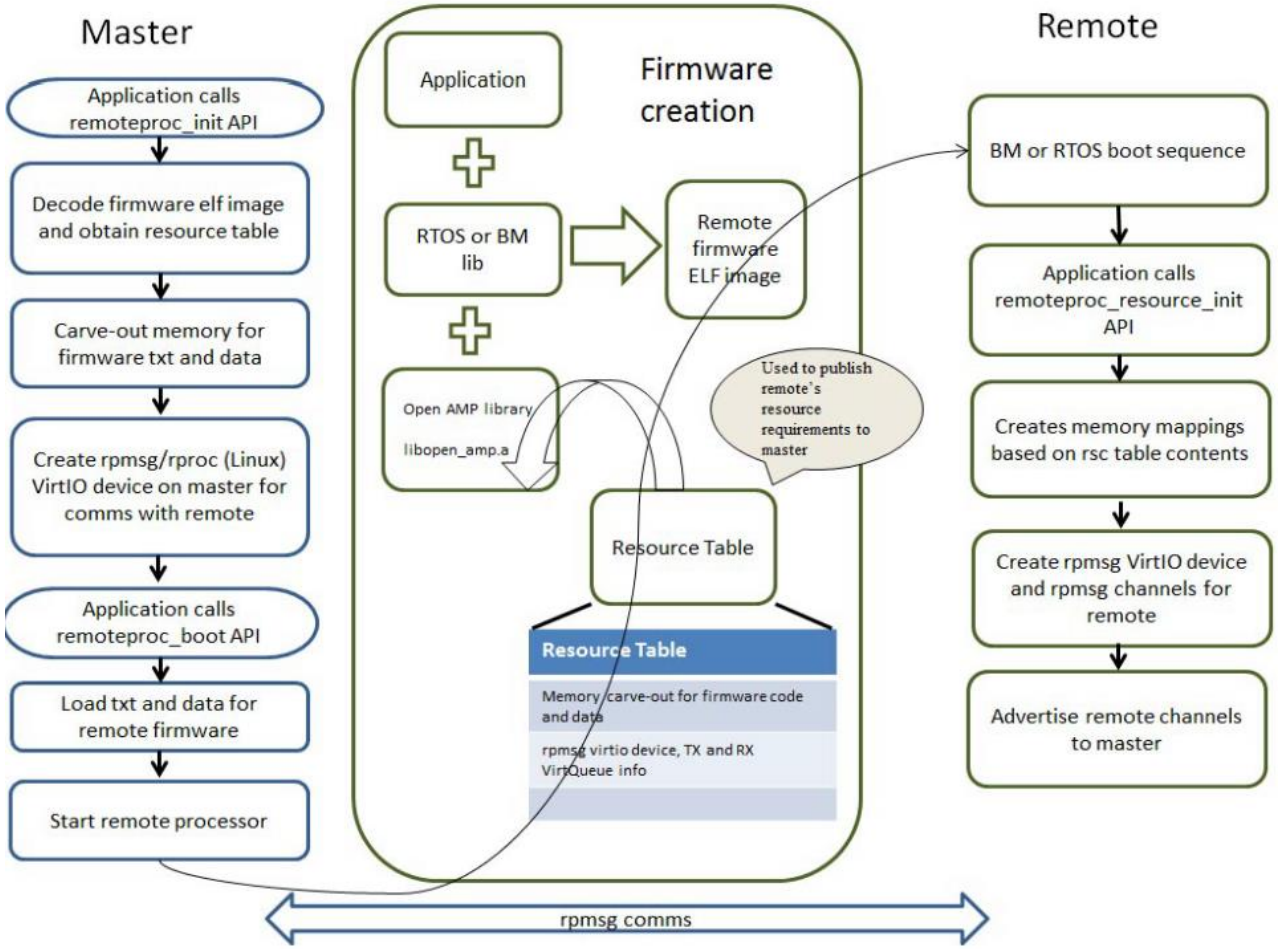
μC3+Linuxとは？　-OpenAMP-

remoteproc APIの重要な機能

① マスタアプリケーションがリモートファームウェアイメージをメモリ内の適切な場所にロードする
② リセットからリモートプロセッサを解放して、リモートファームウェアの実行を開始する
（※場合によっては依存部の実装が必要）
③ リモートコンテキストとのランタイム通信用にRPMsg通信チャネルを確立する
④ リモートプロセッサをシャットダウンする
⑤ リモートアプリケーションがリモート側のリモートシステムをシームレスに初期化し、
マスタコンテキストとの通信チャネルを確立できるようにする

eForce

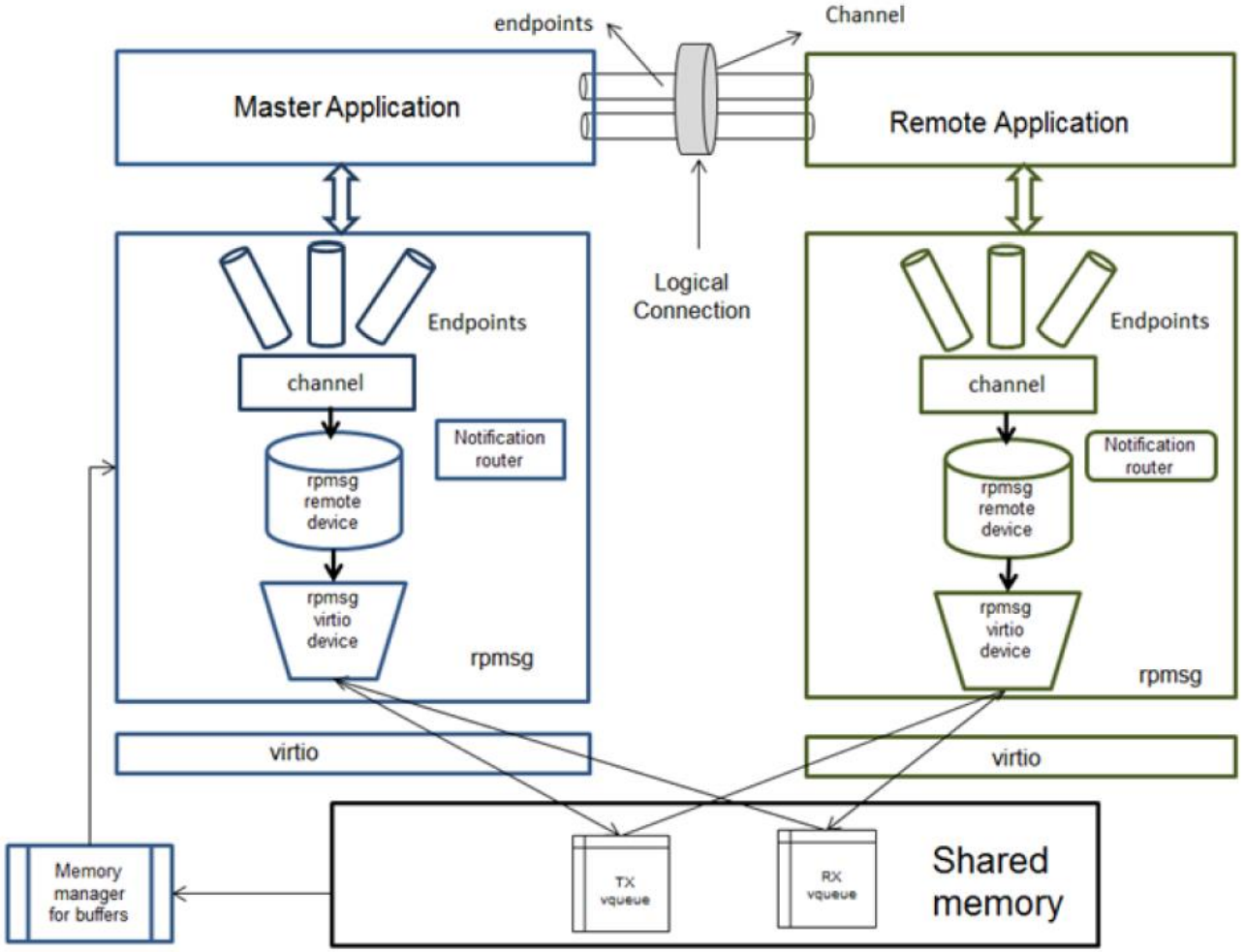
remoteprocの概要



※OpenAMP Framework User Referenceから抜粋

μC3+Linuxとは？　-OpenAMP-

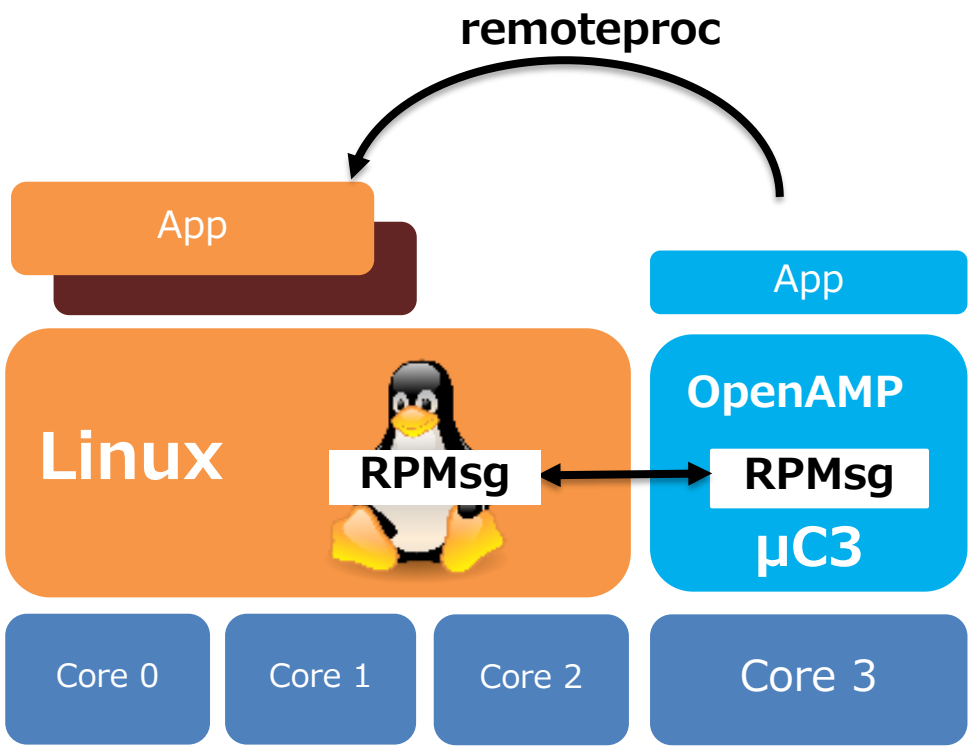
RPMsgの概要



※OpenAMP Framework User Referenceから抜粋

μC3+Linuxとは？

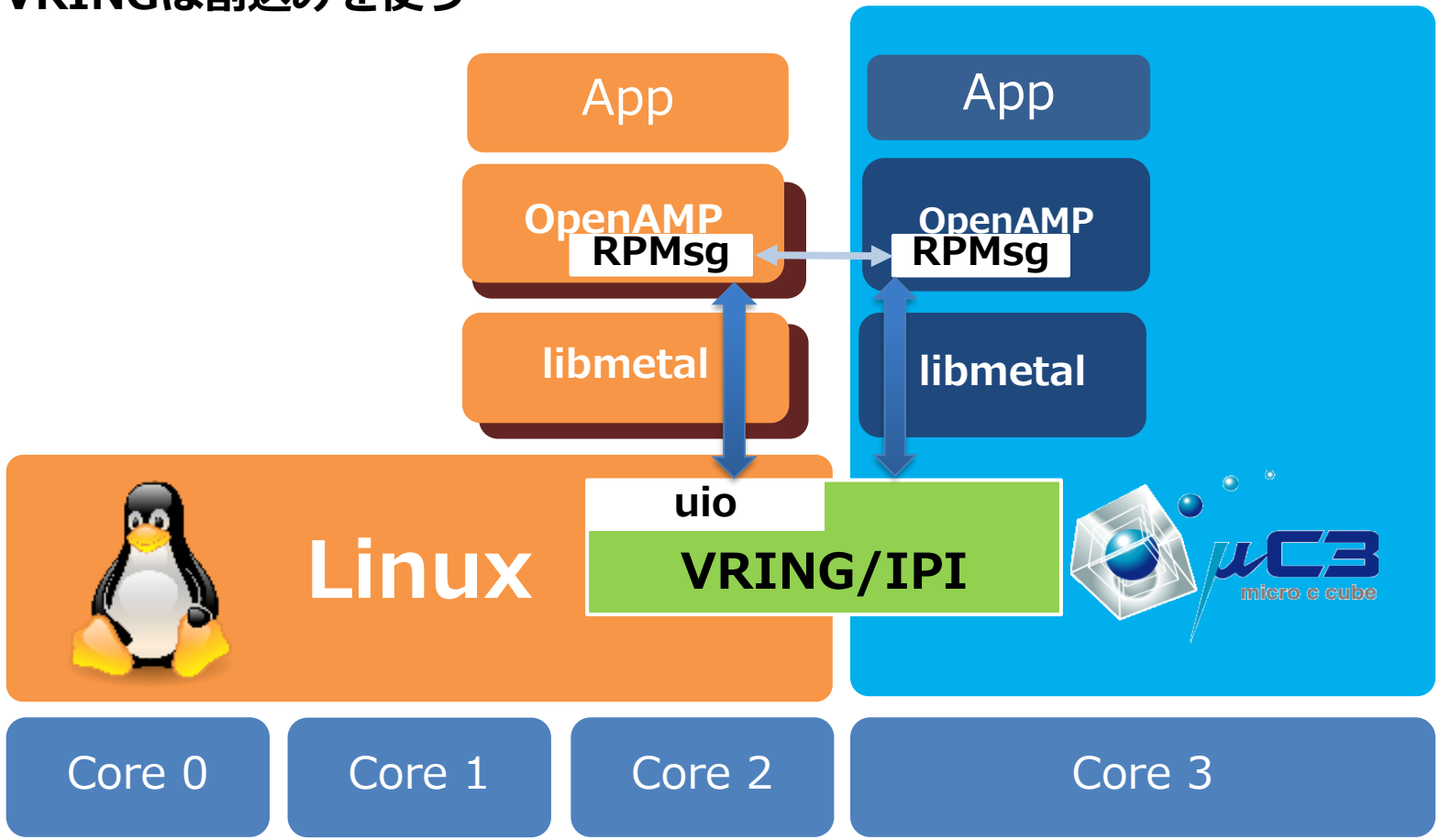
Linuxとコア間連携できるように、μC3にOpenAMPコンポーネントを実装



eForce

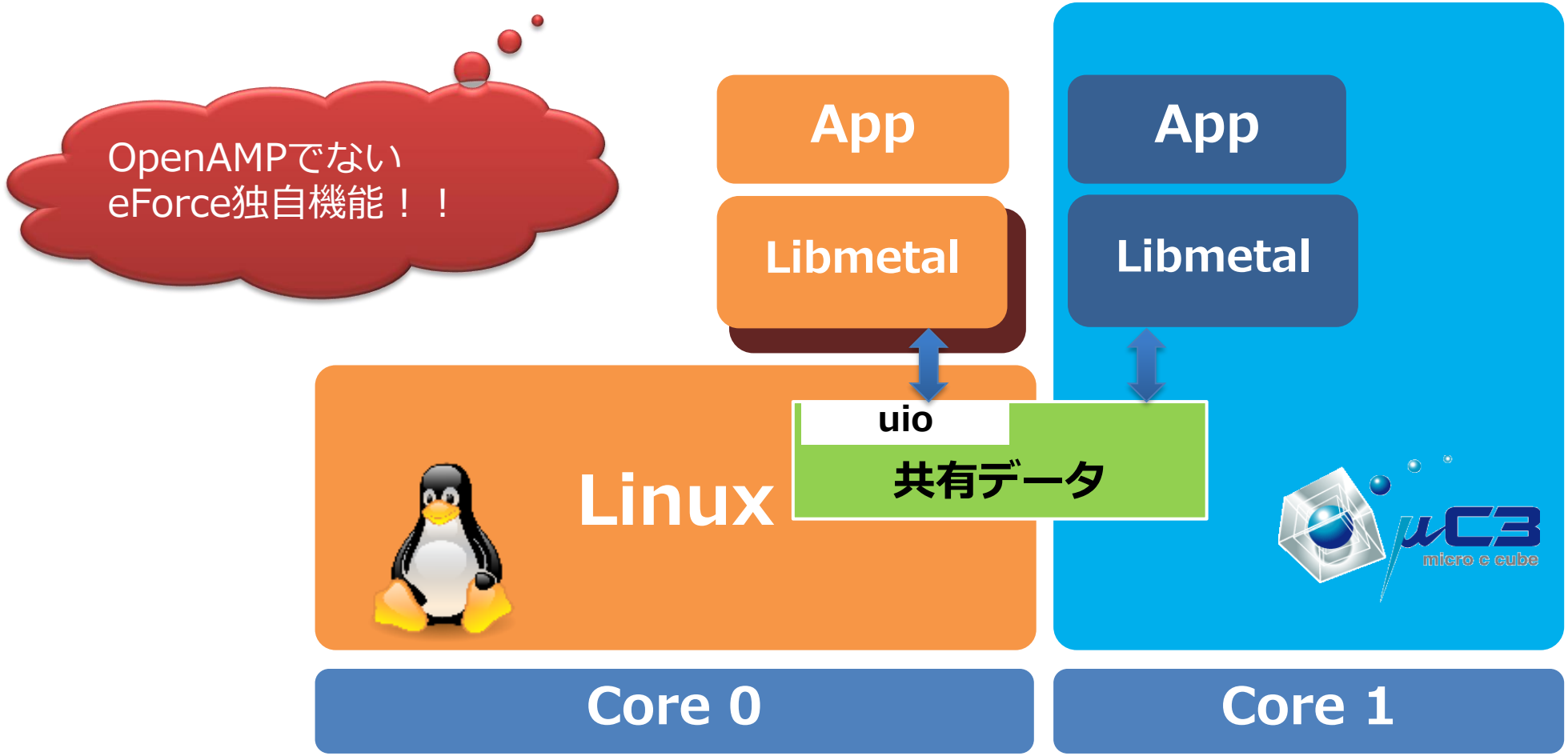
μC3+Linuxとは？

OpenAMPを利用したOS間通信
RPMsgは512バイト（ヘッダを含む）のメッセージ通信
VRINGは割込みを使う



共有データ(大きいデータ）の扱い

・共有メモリの取得・排他制御（スピンロック）が可能に（libmetalを使用）
・共有メモリ専用のAPIを提供



OpenAMPの最近の状況

OpenAMPのリリース履歴

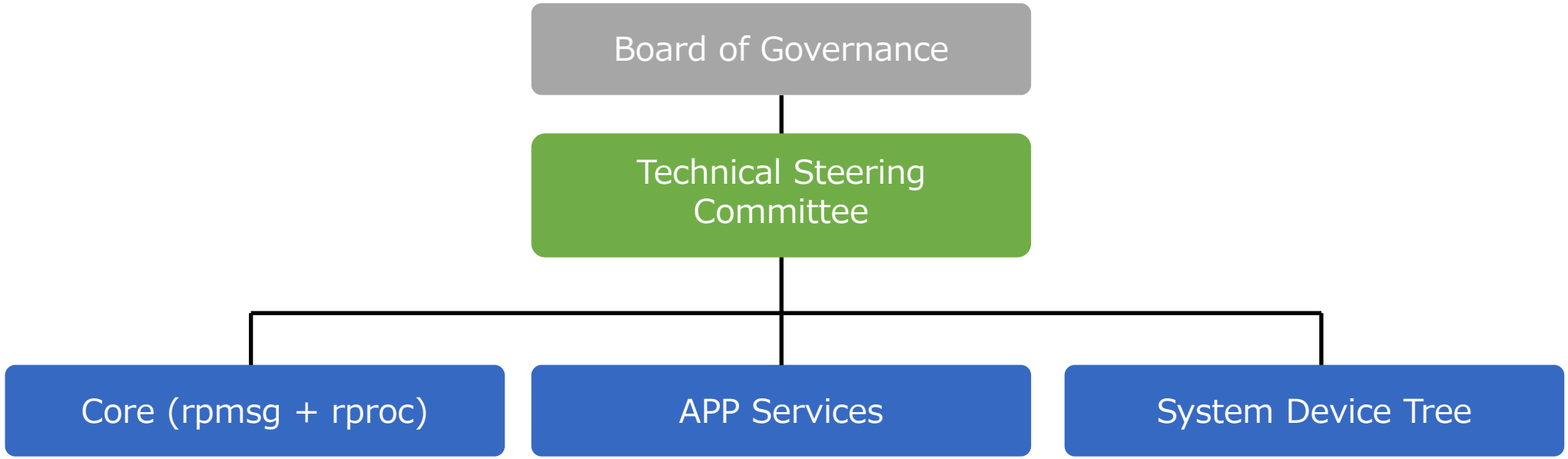
バージョン	内容	備考
2016.04	最初にgithubに登録されたもの	
2016.10	libmetalの導入、Linux ユーザースペース,プロセス間通、 ゼロコピーAPIの追加	HILとlibmetalの抽象化が混在
2017.04	Virtioリセットサポート	
2017.10	バグフィクス	
2018.04	C++サポート,Zepherサポート	従来バージョンの最終版
2018.10	HIL廃止,RPMsgをエンドポイントで通信する方式に変更 RPMsgとRemoteprocの機能を分離	事実上の全面的な作り変え **(μC3+Linuxで採用しているバージョン)**
2020.01	IRQの登録方式の変更	メインメンテナの変更(Xilinx社 → Linaro社)
2020.04	RPMsgのバッファサイズを変更できるように変更	
2020.10	Microblazeのサポート、rpmsgのエンドポイントのアロケーション方式の変更、 バグフィックス、rpmsg_init_eptの廃止	
2021.04	RPMsgゼロコピーAPIの再導入、バグフックス	
2021.10	RPCのサービス化、RPCサンプルの追加	

2018.10以降はIRQの登録方式の変更以外は
大きな変更はいまのところはない

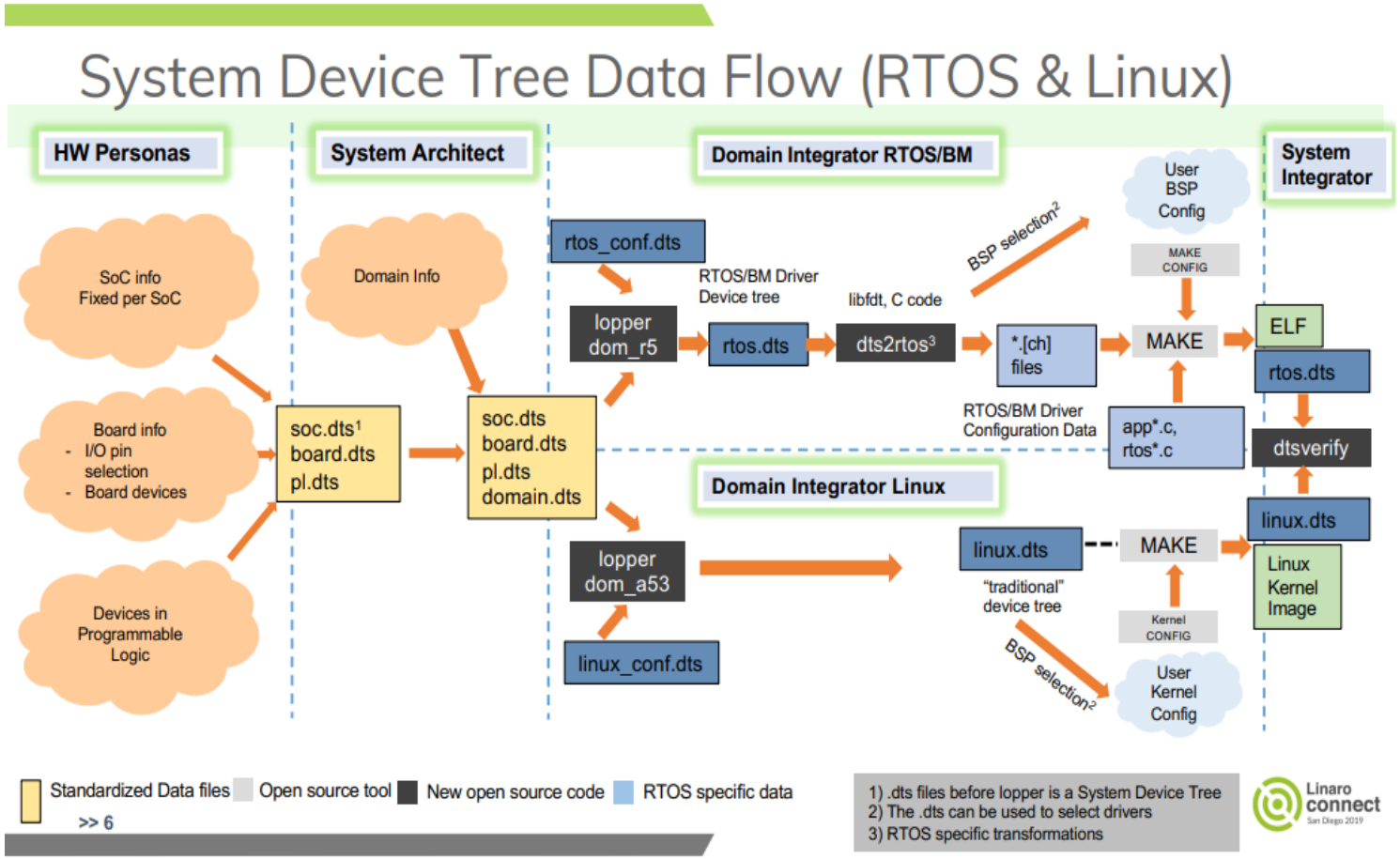
eForce

OpenAMPの今後

- **Linaro社の管理になってからいくつかのサブグループで技術的な議論がされているようです。**
  - Core(rpmsg + rproc)　・・・　いままでのOpenAMP
  - System Device Tree
  - Application Services



System Device Treeへの対応



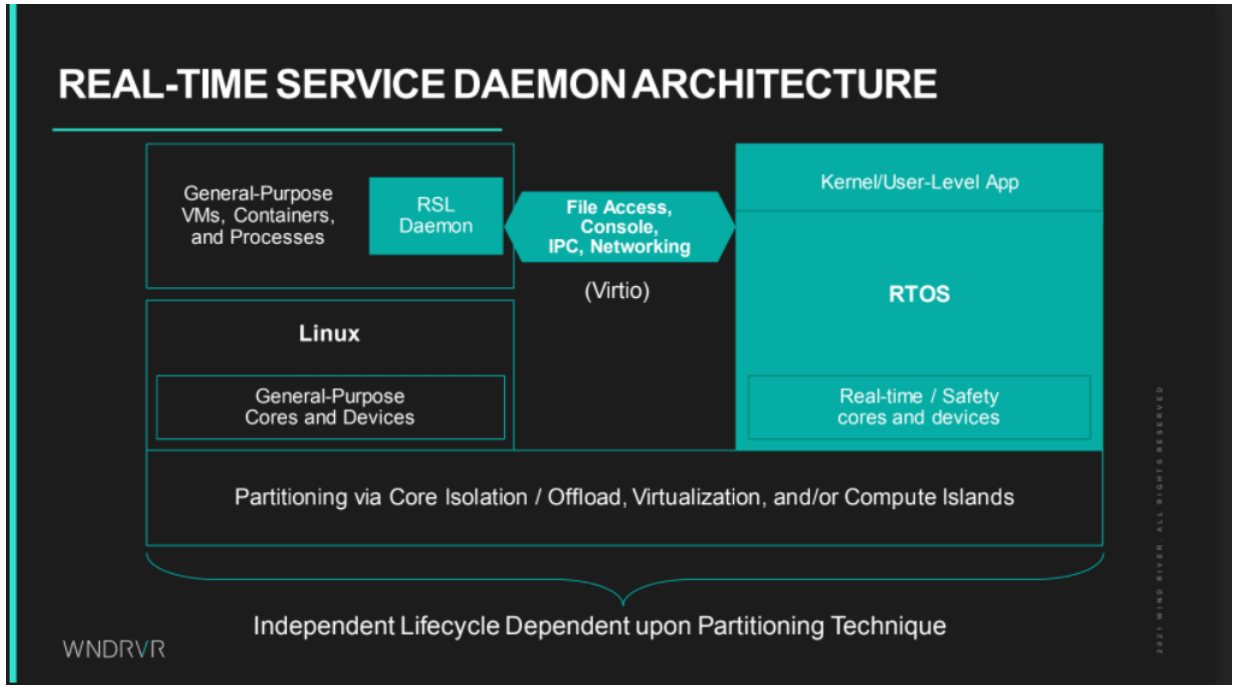
システムデバイスツリー（ドメイン情報）でマルチOSの全ての情報を取りまとめる。

- RTOSやベアメタル用のコードを出力。
- Linux用のデバイスツリーを出力。

https://static.linaro.org/connect/san19/presentations/san19-115.pdf

Application Service（HyperVisor-Less）

- https://drive.google.com/file/d/1GoXWNY-AjjWSrudvljnSaUKTjeK1sHI7/view
- HypervisorlessVirtioBlog_Feb2021.pdf (openampproject.org)



RSLデーモン(kvmtool)経由でRTOSがvirtio-net, 9p, virtio-coloseなどいろいろな機能をつかえるようになる。

※RTOS・ベアメタル側にもvirtio-net, virtio-consoleなどの実装が必要になるが現状のOpenAMPのコアではまだ未実装。

μC3+Linuxの今後

- **新SoCへの積極的な対応**
- **通信部のコンフィギュレーションの容易化**
  - 簡易のコンフィギュレータ等(ET当日ならデモができるかもしれません。)
    将来的にSystem Device Treeに対応するかはまだ未定

- **今後、新しいOpenAMPの新機能への対応**
  - HyperVisorLess等

- **対応するLinux**
  - チップベンダーの提供するもの
    - 各ベンダーによってはLinux側の更新頻度が追いつけないためバージョンはある程度固定になります。
      - バージョンアップに関しては応相談

eForce

まとめ

- **さらに進むSoCのマルチコア・ヘテロジニアスマルチコア化**
  - マルチコア・ヘテロジニアスマルチコア化に対応した課題の解決が必要
- **複雑化するマルチコア・ヘテロジニアスマルチコア向けの課題解決**
  - マルチコア・ヘテロジニアスマルチコア向けのRTOSの活用
    - SMP方式：すべてのコアで１つのOSが動作，スケジューリングの理解が必要。
    - AMP方式：それぞれのコアでOSが動作，シングルコアと同じ要領でアプリケーションが作成できる。
  - マルチOSによるそれぞれのOSの利点を活用
    - GUIなどを中心にLinuxのライブラリを用いて簡単に実装できる。
- **製品紹介**
  - µC3/Standard+M
  - µC3/Standard+M with Hetero
  - µC3+Linux
    - 今後の、「µC3+Linux」の対応
    - OS間通信のコンフィグレーション対応
    - OpenAMPの新機能への対応
- **最後にeForceでは、RTOSの体験セミナーを定期的に開催しております。**

eForce

ご清聴ありがとうございました

eForce