



Embedded  
Multicore  
Consortium

[www.embeddedmulticore.org](http://www.embeddedmulticore.org)

# EMC WG3(マルチコア適用委員会)の活動報告

Activity Report about EMC WG3(Multicore Application Subworking Group)

2017-11

ガイオテクノロジー(株) 岩井 陽二

# EMCで何しよう..

(利用者目線での)

非競争的領域に対し技術トレンドの集約と配信..

実務に転用可能な情報・データ作成と配信.. が良いだろう..と思量

マルチコア移行時のオートコード方法は?

マルチコア移行時の機能分割方法は?

負荷分散設計をする場合の指標は? 設計前の能力測定が大変そう

ツールチェーン統合(設計-評価の統合)は

ハードにばかり目が、意識が、向かってしまう

どうすれば良い?

ソフトウェアのデバッグ方法は? ISO26を想定した場合の設計の考え方の

メリット/デメリット(なぜその方法が良いの?)

並列化すべき方法論は? ツールの選び方は?

OEMとサプライヤの分業境界は?

アーキテクチャ設計指針の目安は?

品確の値ごろ感は?

性能評価方法/不具合検出方法は?

OSの使い方においてマルチコア  
特有の手法に関する情報が乏しい

アプリケーションのシミュレータは?

コア間の実行タイミング評価はどうすれば良いの?

マルチコア(並列動作)対応設計書式の  
標準化は?(設計書レビューが大変そうで心配)

結合評価方法/統合評価方法は?

SMP/AMP/BMPそれぞれの選定目安は?

マルチコア活用のテクニックのメリット/デメリット(なぜその方法が良いの?) RTOS、何を選べは良いの?

シングルコアで発生しない不具合やリスク(として何を想定しておくべきか)の目安は?

コア数や(AMP/SMPなどの)利用方針に影響されないプラットフォームは?



# マルチコア適用委員会を立ち上げました(2017-)

組込みマルチコアコンソーシアム  
(EMC)

関連業界で協力・連携し  
活用支援・ビジネス推進・市場の活性化貢献を実現  
マルチコア技術の組込みシステムでの活用を促進

SHIM  
委員会(WG)

マルチコア向け開発支援ツールのための  
ハードウェア抽象化記述SHIM標準化と導入支援

SHIM: Software-Hardware Interface for Multi-many-core

モデルベース並列化  
委員会(WG)

リファレンスとしてSHIMを利用した  
マルチコア向け設計支援ツール群を開発

マルチコア適用  
委員会(WG3)

この資料で紹介する委員会(WG)

# マルチコア適用委員会（役割と概要）

## 役割

- マルチコア前提の開発が直面する課題を明確にする
  - 課題リストを毎年アップデートし開示する
  - 自動車分野から他の分野に展開する
- 課題のうち、解決できるテーマに対しアウトプットを示す
  - マイルストーンにもとづきアウトプット（成果物）をリリース
- アウトプットを開示し 広く利用を促進する
  - アウトプットは国内外に開示し 広く利用を促進する

## 概要

- WG（委員会）名称
  - マルチコア適用委員会
- WG構成
  - 委員長：岩井（ガイオ・テクノロジー）
  - WGメンバー：18名 9団体 \*\*2017年11月現在
- 活動計画
  - 定例委員会を開催（＋立ち上げ期は集中討議もあり）
- 期間：2017/1～（2020年3月迄の3年間は継続を前提）

# FY17期初の成果物目標はツールチェーン調査&報告書

— フロント転写 → ポジション → (地図)

MBSEの図 → ARコレ → (ARコレ)

EMC MPP本 DM

ポジションを示す「地図」を用意

課題とメソドロジー&ツールを一覧化して議論することに

調査&報告書のテンプレートを合意

EMC Embedded Multicore Consortium  
超込みマルチコア/メニューコア  
ツールチェーン調査&報告書  
Draft 0.1  
Revision 0.01 2017 xx.xx

EMC-WG3-001  
超込みマルチコア/メニューコア  
ツールチェーン調査&報告書

EMC Embedded Multicore Consortium  
超込みマルチコア/メニューコア  
ツールチェーン調査&報告書

Page: 1



# リサーチの方向とテーマ

ジャンル	テーマ・課題	内容
機能安全	ISO26262:2018 DISで追加されたMulti-coreに関する要求事項- ①ISO26262:2018 DIS Part6(ソフトウェア) clause5	モデリングガイドラインとコーディングガイドラインでカバーすべき内容にマルチコアに搭載したソフトウェアの同時処理が追加。 Topics to be covered by modelling and coding guidelines Topics:1i Representation of concurrency aspects ASIL-A + ASIL-B + ASIL-C + ASIL-D + Concurrency of processes or tasks may be a topic when executing software in a multi-core or multiprocessor runtime environment
機能安全	ISO26262:2018 DISで追加されたMulti-coreに関する要求事項- ②ISO26262:2018 DIS Part11 (半導体) clause5	マルチコアに安全機構を搭載した場合のデコンポジションの妥当性説明が追加。 従属故障分析や共通原因故障分析が必要。
機能安全	ISO26262:2018 DISで追加されたMulti-coreに関する要求事項- ③ISO26262:2018 DIS Part11 (半導体) clause5	マルチコアに搭載したソフトウェアに対して、無干渉の妥当性説明が追加。 ソフトウェアの従属故障分析が必要。パーチャル技術の有効性が言及されている。
機能安全	ISO26262:2018 DISで追加されたMulti-coreに関する要求事項- ③ISO26262:2018 DIS Part6,Part11	マルチコアに搭載したソフトウェアコンポーネントのタイミングに関する故障検出。
エンジニアリング現場	組込み市場のほとんどの技術者、管理者について 「そもそも何を課題にすべきかわからない」ことが課題	
エンジニアリング現場	マルチコア移行時の機能分割方法	レガシー資産の機能分割の指針づくり 設計前の能力測定へ 課題:BSWやマルチコアの動作のさせ方を決定と並行し検討するとやり方が定まらない
エンジニアリング現場	不具合検出方法	BSW(プラットフォーム)設計者としての不具合検出 や アプリケーション(制御)設計者としての不具合検出 不具合検出に不慣れな人でもわかり易いものが必要
エンジニアリング現場	性能評価方法	BSW(プラットフォーム)設計者としての性能評価 や アプリケーション(制御設計)者としての性能評価 課題:性能評価の基準をどのように作るか(それぞれの部門ごとに異なる)
エンジニアリング現場	設計前の能力測定	選んだマイコンやBSWなどでどこまで動かせるかの測定(簡易的な考えから)どこまで機能拡張できるか?
マイコンまわり	システム成立性(機能&性能)	
マイコンまわり	システム検証(なのか機能検証なのかは微妙だが)	
マイコンまわり	シミュレーション/検証	
ツール/OSまわり	スレッドへのコア割	
ツール/OSまわり	並列プロセスのス	
ツール/OSまわり	動的なコア割り当	
ツール/OSまわり	可視化 や 何等か	
ツール/OSまわり	デバッグ・結果の	
ツール/OSまわり	ログの取り方、フォ	
ツール/OSまわり	非決定的な挙動の	
ツール/OSまわり	設計フェーズ実装	
ツール/OSまわり	テストケース自動生成 や 形式検証	
ツール/OSまわり	純粹に並行処理ソフトウェアが持つ問題 や デッドロックなど	
ツール/OSまわり	プログラミング言語、設計言語 関数型プログラミング MBD	

機能安全/開発現場/マイコンまわり/ツールOSまわりのジャンルから19テーマをピックアップ

# リサーチの方向とテーマ(ズームアップ)

ジャンル	テーマ・課題
機能安全	ISO26262:2018 DISで追加されたMulti-coreに関する要求事項-①ISO26262:2018 DIS Part6(ソフトウェア) clause5
機能安全	ISO26262:2018 DISで追加されたMulti-coreに関する要求事項-②ISO26262:2018 DIS Part11 (半導体) clause5
機能安全	ISO26262:2018 DISで追加されたMulti-coreに関する要求事項-③ISO26262:2018 DIS Part11 (半導体) clause5
機能安全	ISO26262:2018 DISで追加されたMulti-coreに関する要求事項-③ISO26262:2018 DIS Part6,Part11
エンジニアリング現場	組込み市場のほとんどの技術者、管理者について「そもそも何を課題にすべきかわからない」ことが課題
エンジニアリング現場	マルチコア移行時の機能分割方法
エンジニアリング現場	不具合検出方法
エンジニアリング現場	性能評価方法
エンジニアリング現場	設計前の能力測定
マイコンまわり	システム成立性(機能&性能)
マイコンまわり	システム検証(なのか機能検証なのかは微妙だが)
マイコンまわり	シミュレーション/解析など
ツール/OSまわり	スレッドへのコア割り当て 設計
ツール/OSまわり	並列プロセスのスケジューリングの問題(タイミング要素)
ツール/OSまわり	動的なコア割り当ての設計、実行状況の把握 可視化 や 何等かの指標
ツール/OSまわり	デバッグ・結果の検証 や ログの取り方、フォーマット
ツール/OSまわり	非決定的な挙動の網羅的検証が困難 設計フェーズ実装フェーズでの検証 や テストケース自動生成 や 形式検証
ツール/OSまわり	純粹に並行処理ソフトウェアが持つ問題 や デッドロックなど
ツール/OSまわり	プログラミング言語、設計言語 関数型プログラミング や MBD

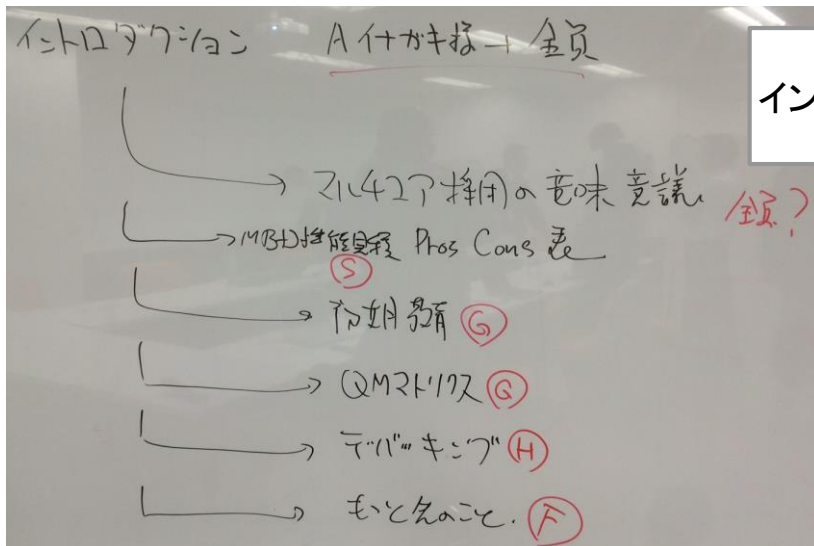
# 期初計画を100とすれば50%程度が本年度の着地点か

ジャンル	テーマ・課題
機能安全	ISO26262:2018 DISで追加されたMulti-coreに関する要求事項-①ISO26262:2018 DIS Part6(ソフトウェア) clause5
機能安全	ISO26262:2018 DISで追加されたMulti-coreに関する要求事項-②ISO26262:2018 DIS Part11 (半導体) clause5
機能安全	ISO26262:2018 DISで追加されたMulti-coreに関する要求事項-③ISO26262:2018 DIS Part11 (半導体) clause5
機能安全	ISO26262:2018 DISで追加されたMulti-coreに関する要求事項-③ISO26262:2018 DIS Part6,Part11
エンジニアリング現場	組込み市場のほとんどの技術者、管理者について「そもそも何を課題にすべきかわからない」ことが課題
エンジニアリング現場	マルチコア移行時の機能分割方法
エンジニアリング現場	不具合検出方法
エンジニアリング現場	性能評価方法
エンジニアリング現場	設計前の能力測定
マイコンまわり	システム成立性(機能&性能)
マイコンまわり	システム検証(なのか機能検証なのかは微妙だが)
マイコンまわり	シミュレーション/解析など
ツール/OSまわり	スレッドへのコア割り当て 設計
ツール/OSまわり	並列プロセスのスケジューリングの問題(タイミング要素)
ツール/OSまわり	動的なコア割り当ての設計、実行状況の把握 可視化 や 何等かの指標
ツール/OSまわり	デバッグ・結果の検証 や ログの取り方、フォーマット
ツール/OSまわり	非決定的な挙動の網羅的検証が困難 設計フェーズ実装フェーズでの検証 や テストケース自動生成 や 形式検証
ツール/OSまわり	純粹に並行処理ソフトウェアが持つ問題 や デッドロックなど
ツール/OSまわり	プログラミング言語、設計言語 関数型プログラミング や MBD



# 今年度アウトプット 目次構成と役割を検討

- WG当初から少し軌道修正
- イン트로ダクションでマルチコアの必要性を説き 課題解決の必要性を挙げ 以降のチャプターにつなげる構成とする
- 各チャプターの原稿ネタは各社大なり小なり準備をすすめているので書き進める



## イントロダクション

マルチコア採用の意味・意義

## 各チャプター

マルチコア利用者向けガイド(教育)

MBDベースの性能見積(ProsCons)

QMマトリクス+ツール一覧

デバッグ・テスト

もっと先のこと

# GQM法を用いた検討アプローチ

## ■ GQM(Goal Question Metrics)法による検討アプローチ

車載ECU マルチコア化による複数ゴールを検討



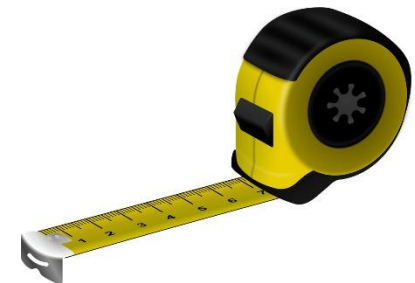
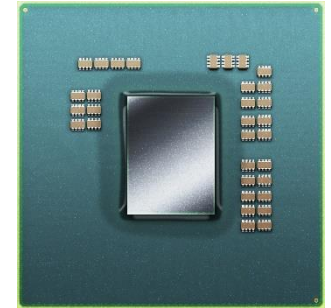
各ゴールを満足させるためには何が求められるかを調査・検討



各クエスションを判定するための定量的な指標を検討



指標(メトリクス)を計測・評価する手段となるツールチェーンを紐づけ



# 車載ECUマルチコア採用のゴールから考察

## ■ 以下のゴールを設定

ただし、検討をシンプルかつ分かりやすくするために  
マイコンはAMP(ホモジーニアス)であることを前提に議論と考察を実施中  
～ゴール設定～

- ① 機能安全上の冗長化機構を実現する
- ② コアのクロックスピードを上げずにパフォーマンスを増加させる
- ③ ECU統合によりECUの総数を減らす
- ④ 搭載機能の増加に対応する
- ⑤ 画像処理(大容量データ／複数データの扱い)※



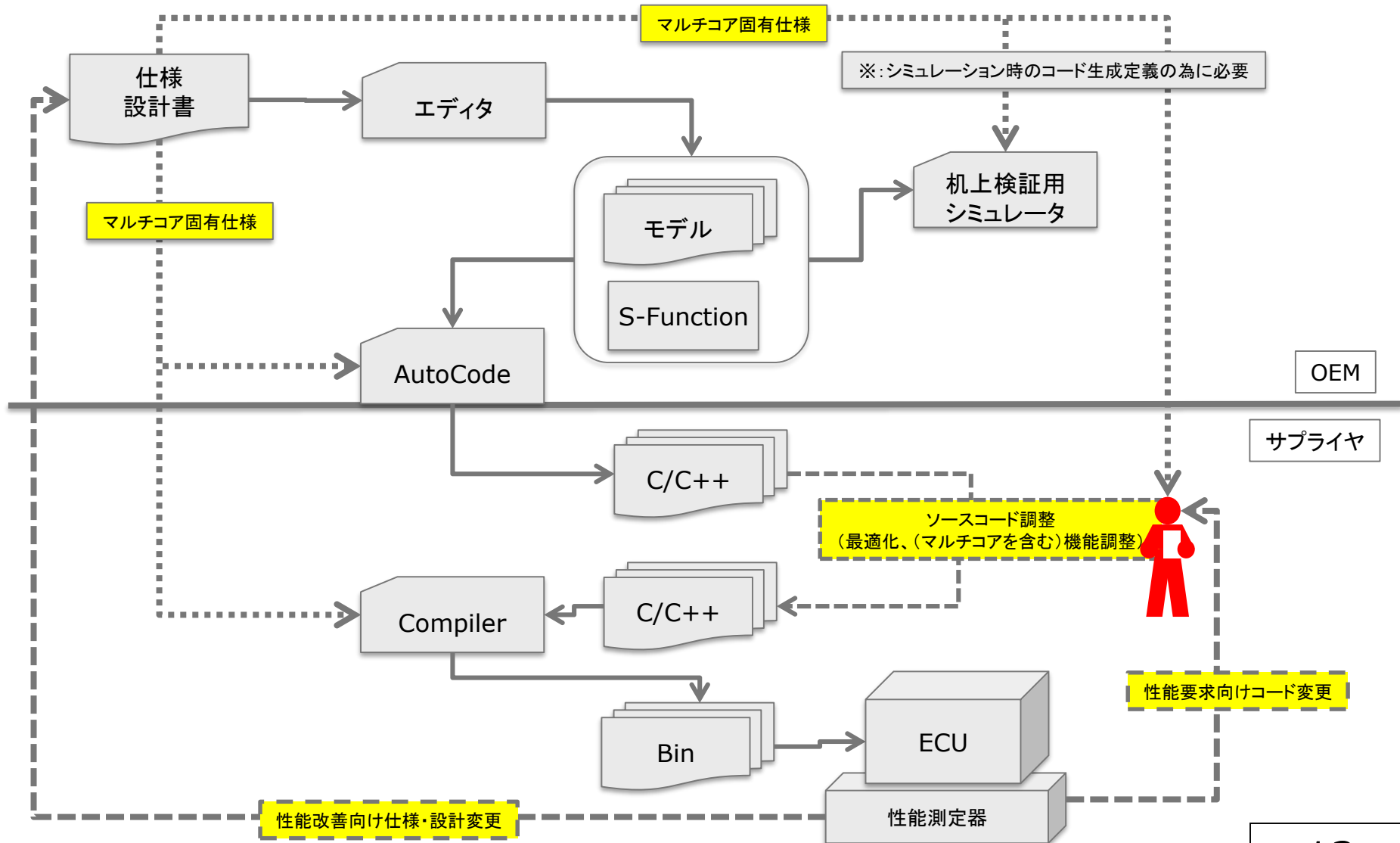
※⑤はADASのような ヘテロジーニアス なドメインとなるため  
次の改版時の検討とし 今回はAMP前提で検討を進めることとした

# 現在 下表のようなシートを作成・検討中

## ■ Q「クエスチョン」、M「メトリクス: 検証評価項目」、検証または評価手段、対応するツールチェーン、といった項目が表に並ぶことに..

No.	クエスション	メトリクス: 検証評価項目	留意点	検証または評価手段	対応するツールチェーン				
1-1	共働れしない冗長構成になっているか	マスターが停止もしくは異常状態になったときにスレーブ側に切り替わるか		故障診断, 故障注入, テスト	SPLS	HLS	ベンチ		
1-2	特定のコアに異常が発生しても他コアの動作に影響をあたえないか	デュアル実行時に片方が停止もしくは異常状態になったとしても他方が動作し続けるか		故障診断, 故障注入, テスト	SPLS	HLS	ベンチ		
2-1	踏込機能を備えているか	特定のタスクに異常が発生しても他タスクの動作に影響をあたえないか		故障診断, 故障注入, テスト	SPLS	HLS	ベンチ		
3	電力消費量の大幅な増加がないか	(マルチコア化で)電力消費量の大幅な増加がないか		ECU内搭電源にて計測 (一々個での測定)	テスター	オシロスコープ/メモリハイコーダー			
4	発生熱量の大幅な増加がないか	(マルチコア化で)発生熱量(または℃)の大幅な増加がないか		温度センサーによる計測	テスター	オシロスコープ/メモリハイコーダー			
5	ワイヤ・ハルネスの削減につながっているか	(マルチコア化で)ワイヤ・ハルネスの削減につながっているか		総延長や総重量の計測	ジョナー				
6	ECU数が削減できたか			-					
7-1	ECU全体のリソース(速度、メモリ)制約を満足しているか	コア毎の、例えば単位時間(10msあたり)の実行時間が設計値以内であるか		計測器もしくは内蔵ソフトウェアで計測する	supplier毎のカスタムソフトウェア	T1	TA	SymTA/S TraceAnalyzer	
7-2		タスク毎の実行時間が設計値以内であるか		計測器もしくは内蔵ソフトウェアで計測する	supplier毎のカスタムソフトウェア	T1	TA	SymTA/S TraceAnalyzer	
7-3		共有メモリからのR/W実行速度が設計値以内であるか		計測器もしくは内蔵ソフトウェアで計測する	supplier毎のカスタムソフトウェア	T1	TA	SymTA/S TraceAnalyzer	
7-4		ROM/RAMの消費量が設計値以内であるか		計測器(コンパイラ)で計測する	ターゲットコンパイラ				
7-5	ECU全体のリソース(速度、メモリ)の拡張性を備えているか	タスク毎のスタック消費量が設計値以内であるか		計測器もしくは内蔵ソフトウェアで計測する	ターゲットコンパイラ				
8-1		コア毎の、例えば単位時間(10msあたり)の実行時間が設計値以内であるか	余裕度を別途定義しておくこと	計測器もしくは内蔵ソフトウェアで計測する	supplier毎のカスタムソフトウェア	T1	TA	SymTA/S TraceAnalyzer	
8-2		タスク毎の実行時間が設計値以内であるか	余裕度を別途定義しておくこと	計測器もしくは内蔵ソフトウェアで計測する	supplier毎のカスタムソフトウェア	T1	TA	SymTA/S TraceAnalyzer	
8-3		共有メモリからのR/W実行速度が設計値以内であるか	余裕度を別途定義しておくこと	計測器もしくは内蔵ソフトウェアで計測する	supplier毎のカスタムソフトウェア	T1	TA	SymTA/S TraceAnalyzer	
8-4	ECU内の搭載機能数が増えたか	ROM/RAMの消費量が設計値以内であるか	余裕度を別途定義しておくこと	計測器(コンパイラ)で計測する	ターゲットコンパイラ				
8-5		タスク毎のスタック消費量が設計値以内であるか	余裕度を別途定義しておくこと	計測器もしくは内蔵ソフトウェアで計測する	ターゲットコンパイラ				
9	ECU内の搭載機能数が増えたか	ECU内の搭載機能数が増えたか	機能数の単位はコンポーネント毎に異なる。 例えば、エンジン系/ボディー系/シャシー系で異なる	数える 動作検証する 設計DRで確認する	HLS	ベンチ	実車		
10-1	通信プロトコルや信号プロトコル体系が最適化されているか	通信のトラフィック量が設計値以内であるか		プロトコルアナライザもしくはOS機能で計測する	OSコマンド	プロトコルアナライザ			
10-2		通信のサイクルが設計値以内であるか		プロトコルアナライザもしくはOS機能で計測する	OSコマンド	プロトコルアナライザ			
10-3		通信毎のTATが設計値以内であるか		プロトコルアナライザもしくはOS機能で計測する	OSコマンド	プロトコルアナライザ			
11	車載制御システムに要求されるリアルタイム性を満たしているか	機能毎に定められたレスポンスタイムが設計値以内であるか	方案はHLS以降 マイコン内のソフトウェアについてはPLSも可能	動作検証する 設計DRで確認する	PLS	HLS	ベンチ	実車	
12	異なるコアに割り付けられたタスク間で、ブロッキングを用いて排他制御する機能を持っているか	排他制御が機能しているか	RTOSの機能が有効に働いているか	動作検証する 設計DRで確認する	T1	TA	SymTA/S TraceAnalyzer		
13	タスクまたはISRから、他のコアに対して割り込みハンドラの起動を要求する機能を持っているか	割り込み制御が機能しているか	RTOSの機能が有効に働いているか	動作検証する 設計DRで確認する	T1	TA	SymTA/S TraceAnalyzer		
14	シングルコア向けに開発した1つのアプリケーションを、性能向上のために複数のコアに跨って動作させる機能を持っているか	横道設計通りコンプリケーションが動作しているか		動作検証する 設計DRで確認する	T1	TA	SymTA/S TraceAnalyzer		
15	信頼系と制御系のコアを分離できているか	横道・上・下層分組が破綻せずに動作しているか	コア分離の結果、特定コアに負荷が集中するようにならないこと 結果的にパフォーマンスが劣化しては本末転倒	動作検証する 設計DRで確認する	PLS	HLS	ベンチ	実車	
16-1	低コストに並列処理が可能か	横道設計通りに並列処理が動作しているか		動作検証する 設計DRで確認する	T1	TA	SymTA/S TraceAnalyzer		
16-2		パフォーマンス劣化が発生していないか		計測器もしくは内蔵ソフトウェアで計測する	T1	TA	supplier毎のカスタムソフトウェア	SymTA/S TraceAnalyzer	
17-1	全体制御CPUのやりとり ECUが受け持つセンサーの処理、演算処理、ECUが受け持つモーターの制御などを並行して実行できるか	横道設計通りに並行に処理が動作しているか		動作検証する 設計DRで確認する	T1	TA	supplier毎のカスタムソフトウェア	SymTA/S TraceAnalyzer	
17-2		パフォーマンス劣化が発生していないか		計測器もしくは内蔵ソフトウェアで計測する	T1	TA	supplier毎のカスタムソフトウェア	SymTA/S TraceAnalyzer	
18	コア間のリソース共有を最小限に抑えられているか	共有リソースの占有が設計値以内であるか		計測器もしくは内蔵ソフトウェアで計測する	T1	TA	supplier毎のカスタムソフトウェア	SymTA/S TraceAnalyzer	
19	将来に増加する(アプリケーション機能)を見込んだ、パフォーマンス予測ができるか	過去値・現在値からパフォーマンスの実測値を計測できるか	過去値、現在値を回る仕組みをもっていないと予測はできない そのうえで、事前に予測できる設計DRとかができるか？ 定常運転タスク、割り込みイベント 同期・非同期 それぞれに対して検証する必要がある	計測器もしくは内蔵ソフトウェアで計測する 設計DRで確認する	T1	TA	supplier毎のカスタムソフトウェア	SymTA/S TraceAnalyzer	
20	定時性を確保できているか	機能毎に求められる、時間防壁性における定時性を満たしているか		計測器もしくは内蔵ソフトウェアで計測する 設計DRで確認する	T1	TA	supplier毎のカスタムソフトウェア	SymTA/S TraceAnalyzer	

## 現場での開発環境とマルチコア概念介在点

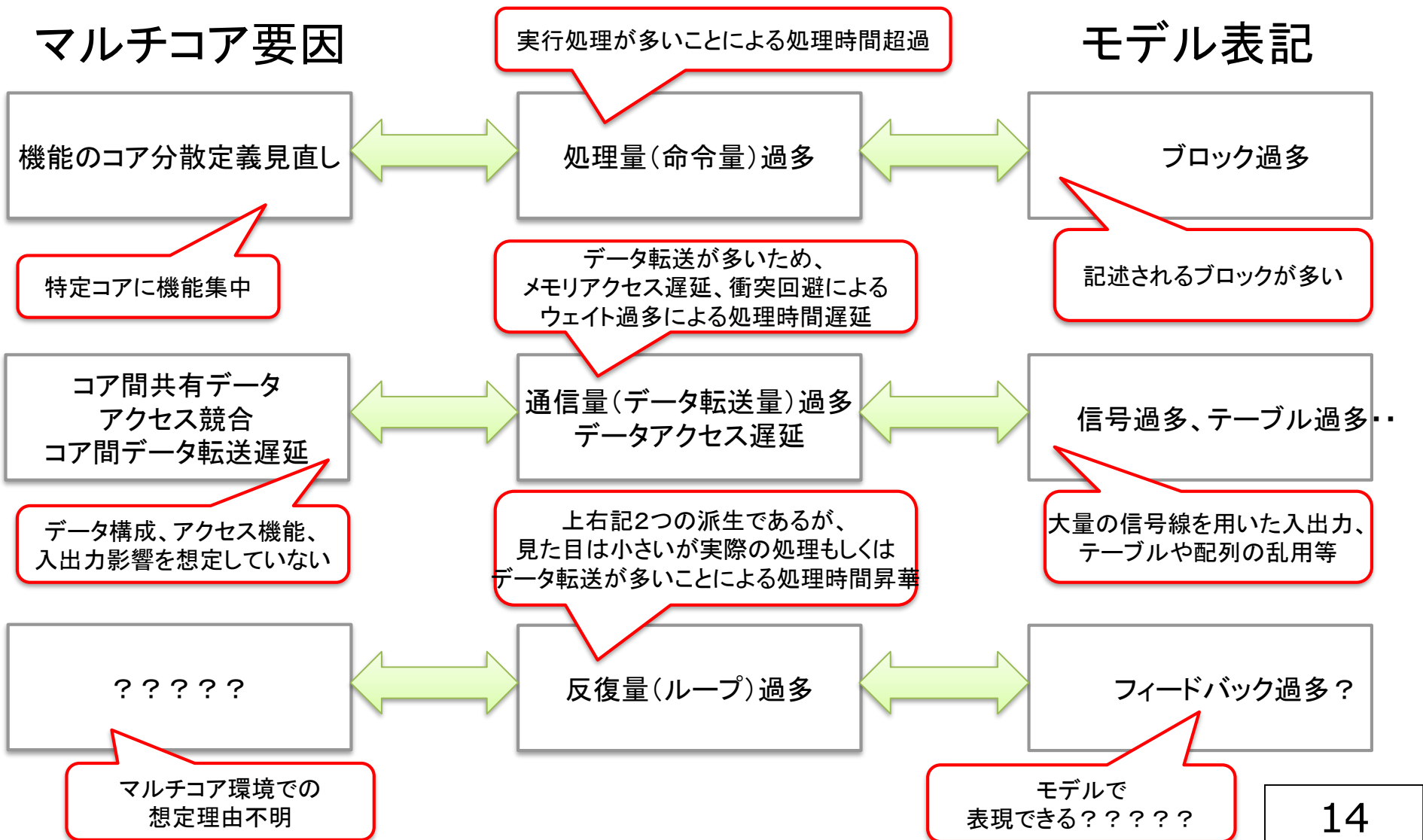


# 処理時間破綻の原因想定とモデル表記との関連性

## 破綻要因

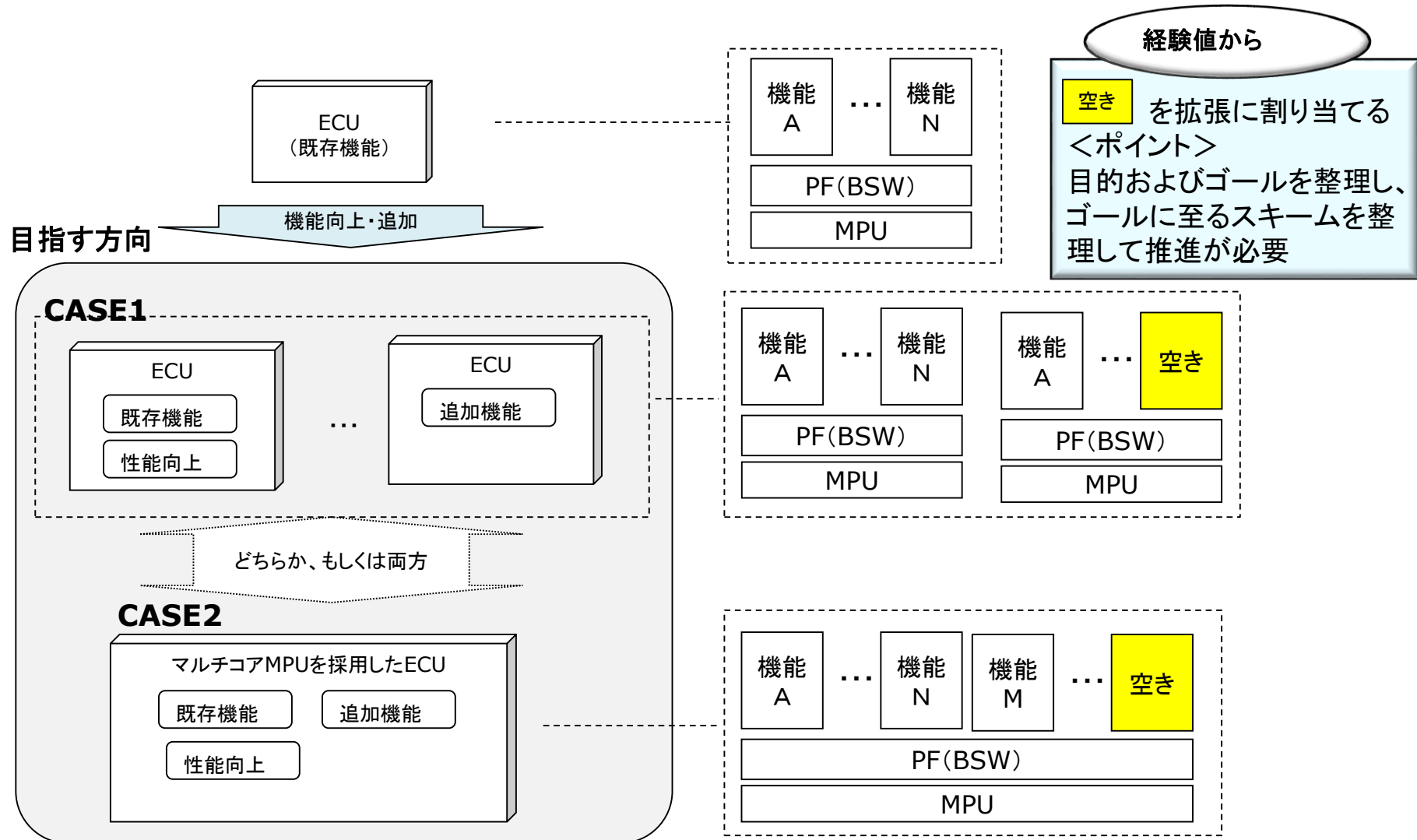
### マルチコア要因

### モデル表記



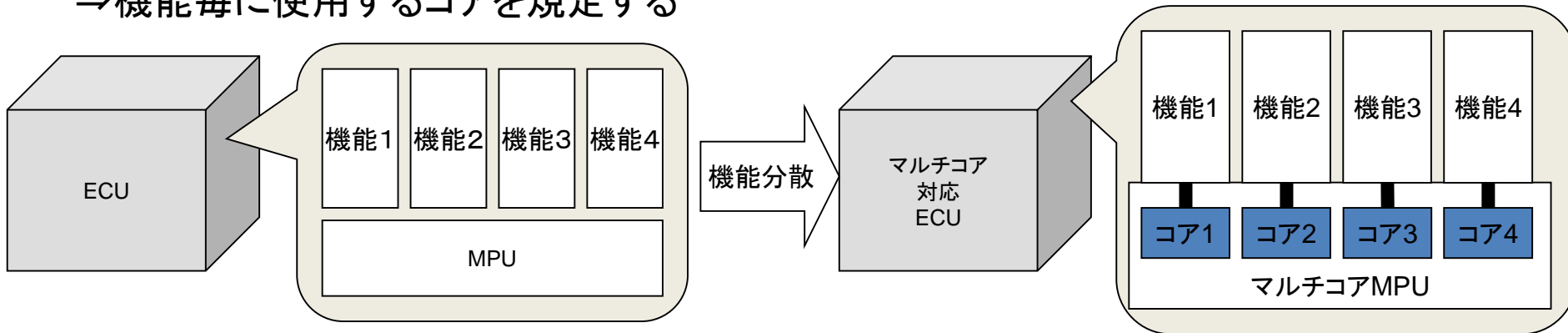


# マルチコアの検討

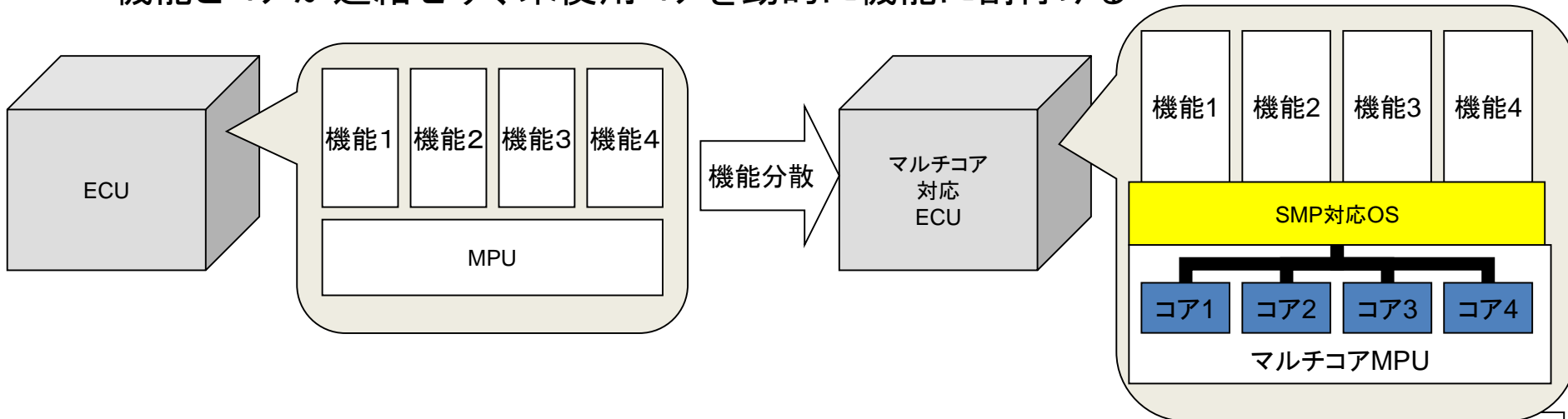


# マルチコア対応ECUの実現形態 1例

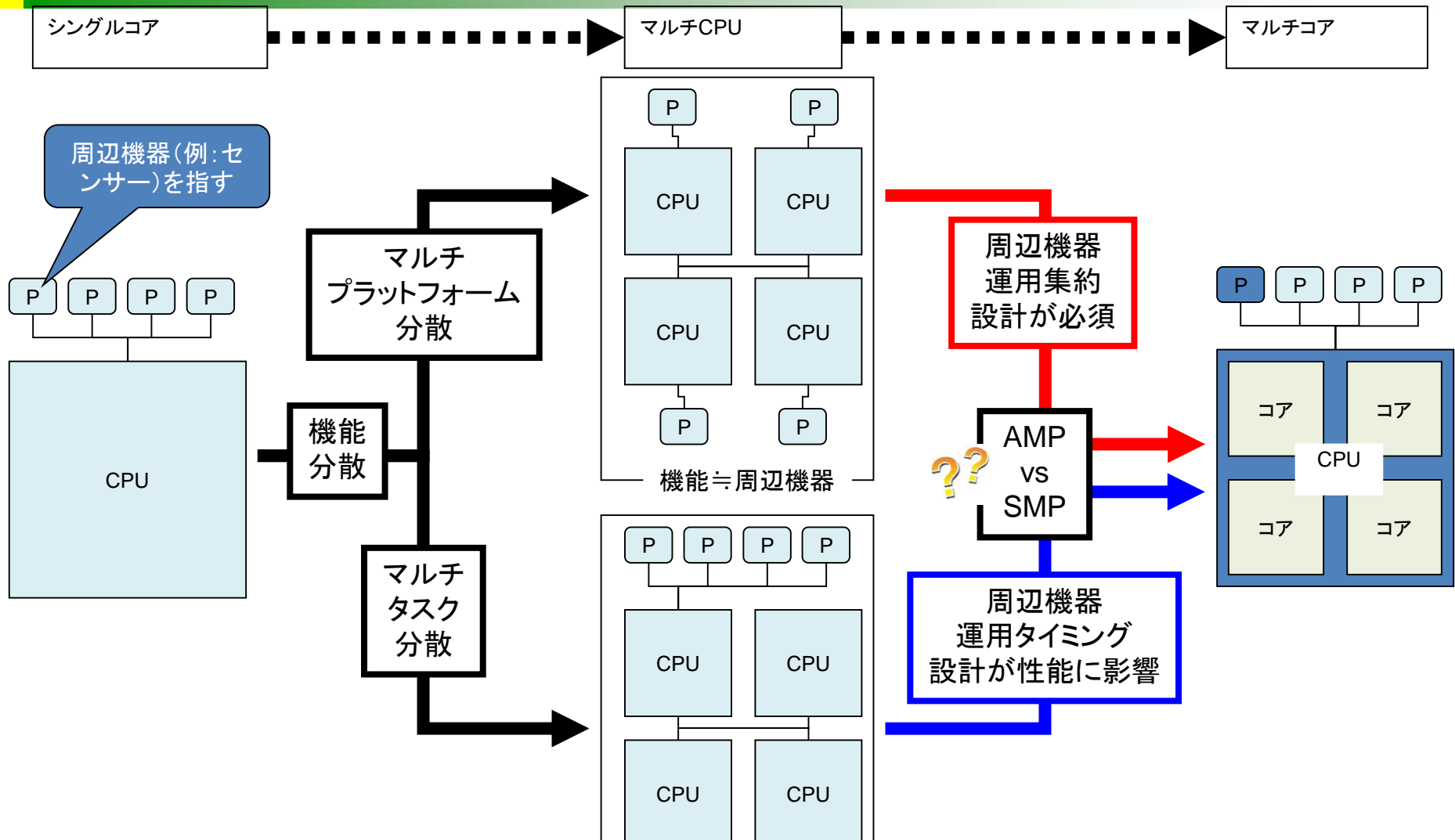
ECU性能向上1(非対称マルチプロセッシング(AMP)方式)  
⇒機能毎に使用するコアを規定する



ECU性能向上2(対称マルチプロセッシング(SMP)方式)  
⇒機能とコアが連結せず、未使用コアを動的に機能に割付ける

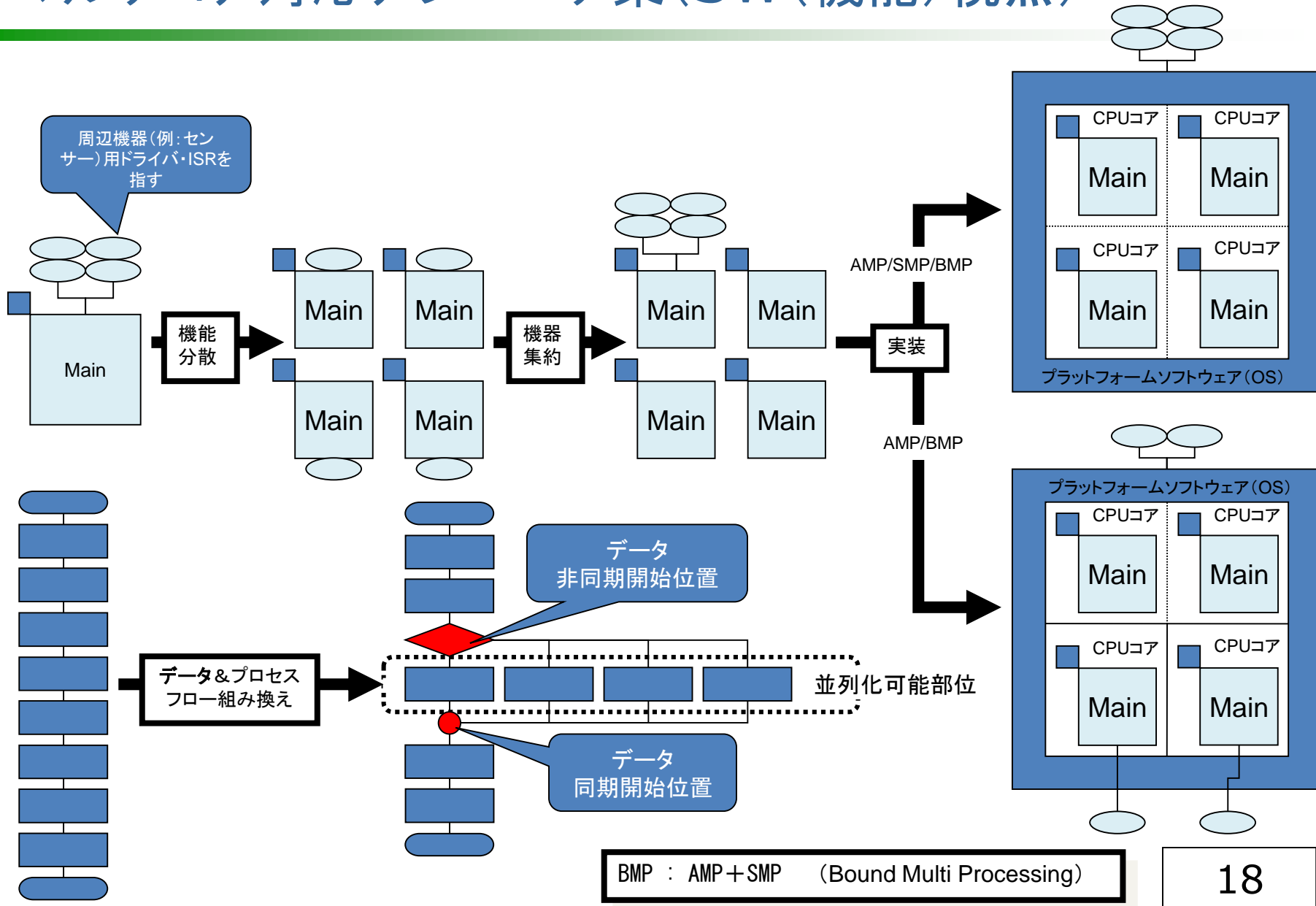


# マルチコア対応アプローチ案(PF/HW視点)



SMP : 対称型マルチプロセッシング (Symmetric Multiprocessing)  
AMP : 非対称型マルチプロセッシング (Asymmetric Multiple Processing)

# マルチコア対応アプローチ案(SW(機能)視点)



## WG3 先々の構想(妄想)・期待

組込みマルチコアコンソーシアム  
(EMC)

SHIM委員会(WG)

モデルベース並列化委員会(WG)

マルチコア適用委員会(WG3)

技術図書制作  
SWG

ツール化の種  
未踏技術研究  
SWG

外部団体協調  
SWG

- 一緒に考え、汗をかく、実業界の会員を増やしたい..
- その先には..WG3からSWGを立ち上げ業界に貢献、その副次物として、個社のビジネスの種を見出したい
- マルチコア・メニーコア時代の開発力涵養に貢献したい

(良い悪いは別にして)ほとんどの方は、ご自身の業務にスグ役に立つ参考書やテンプレートが欲しい、と思われるでしょうから、技術図書制作サブワーキングはお役立ちポイントが高いだろう、と思量。

WG3を推進すると、足りないツール、残念なツールが見えてくる..ここから狭間を埋めるツール化の種が芽生える、と、期待。また、誰も、どこも、挑戦していない技術研究を提議したい。

世の中の、先行する、外部団体、活動体、と協調することで、 $1+1=2$ を創出したい。重複活動を未然に回避し、エコな業界活動をめざす。



Embedded  
Multicore  
Consortium

[www.embeddedmulticore.org](http://www.embeddedmulticore.org)

お問い合わせは

[www.embeddedmulticore.org](http://www.embeddedmulticore.org)