第 3 章 WG1 設計

3-1 概要

国際活動においては、将来の SoC（System on Chip）構造・規模を定量分析可能な形式で予測したモデルとして、モバイル・プラットフォーム SoC モデルを構築しました。本モデルは ITRS Design WG（Working Group）に提示済みであり、2005 年版への掲載に向け賛同と評価を得ました。

国内活動としては、設計技術向上（設計生産性向上）に対する要求値の明確化に着手しました。今年度は、システム・アーキテクチャ設計を注力対象として選定し、中間レベル Canonical Design Flow と、各サブ工程における技術要求ロードマップを検討しました。

補足：Canonical Design Flow（規範［基準］設計フロー）とは、設計技術ロードマップ検討のために、現在及び将来の SoC 設計フローを、定義・明確化した図及び解説です。詳細は、2004 年版 ITRS の Design 章を参照願います。

3-1-1 国際活動の概要（図表 3-1）

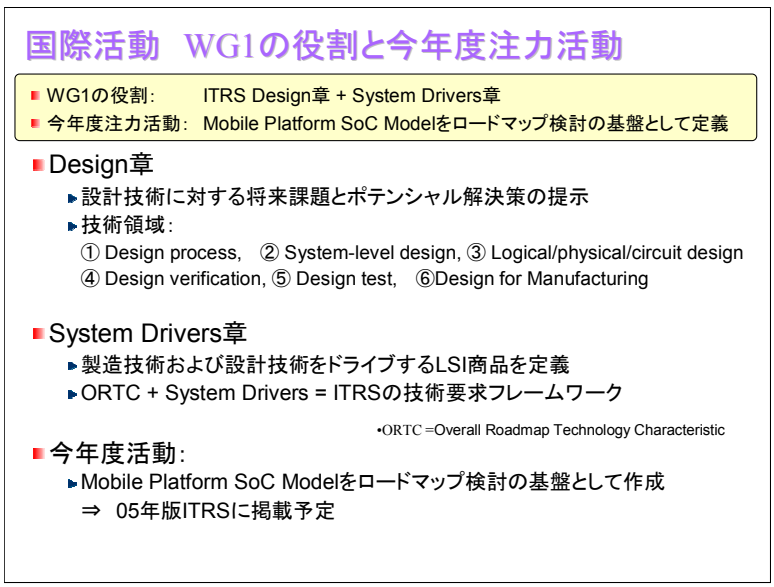
4 月、7 月、12 月にそれぞれ欧州、米国、東京で開催された ITRS の Design WG に参加しました。また、6 月及び 2 月に、米国で臨時会議を開催し、議論の加速を図りました。

WG1 による最大の成果は、モバイル・プラットフォーム SoC モデルです。モバイル・プラットフォーム SoC モデルは、パーソナル・モバイル機器の中核 SoC を想定し、その構造及び規模を数値表現した将来像です。本 SoC モデルを日米欧間の設計技術ロードマップ検討の共通基盤とすることにより、国際ロードマップ検討を加速できると考えております。

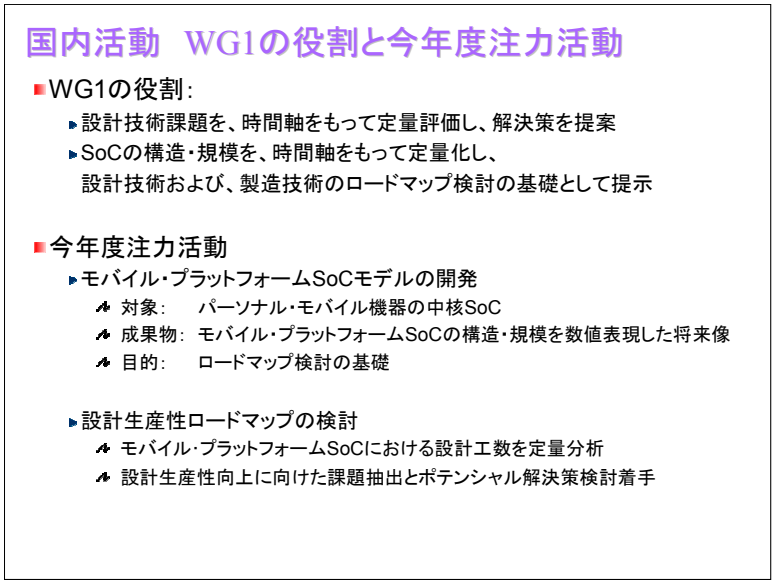
3-1-2 国内活動の概要（図表 3-2）

国内活動においては、設計技術向上（設計生産性向上）に対する要求値の明確化に着手しました。今年度は、システム・アーキテクチャ設計に注力し、同工程における中間レベル・フローの作成及び、必要となる新設計技術とその要求・実現時期を定義しました

システム・アーキテクチャ設計に設計技術ロードマップの検討方法として新たな検討枠組みが完成したと考えており、システム・アーキテクチャ設計に対して適用した検討枠組みを、今後は全設計工程に展開したいと考えています。



図表 3-1 国際活動：WG1 の役割と今年度注力活動

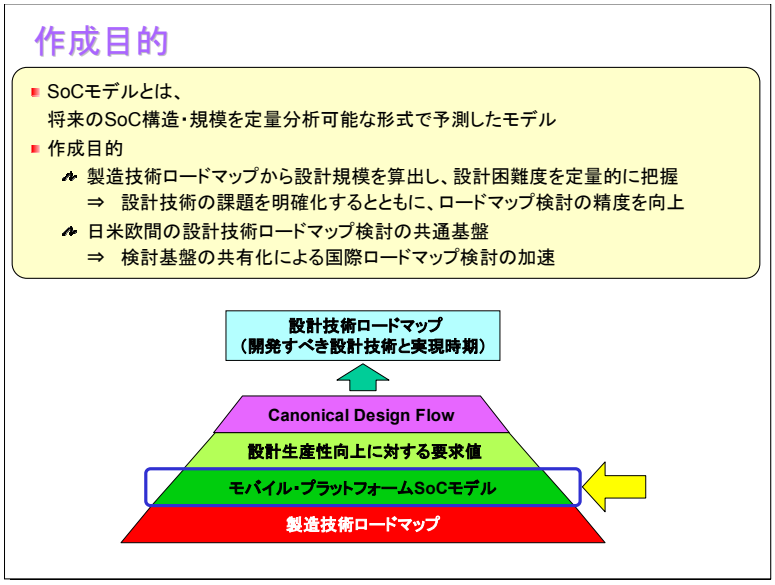


図表 3-2 国内活動:WG1 の役割と今年度注力活動

3-2 モバイル・プラットフォーム SoC モデル

3-2-1 SoC モデルの作成目的（図表 3-3）

SoC モデルとは、将来の SoC 構造・規模を定量分析可能な形式で予測したモデルです。SoC モデルを作成する 1 番目の目的は、製造技術ロードマップから設計規模を算出し、設計困難度を定量的に把握することです。これにより、設計技術の課題を明確化するとともに、ロードマップ検討の精度が向上できます。2 番目の目的は、SoC モデルを日米欧間の設計技術ロードマップ検討の共通基盤とすることにより、国際ロードマップ検討を加速させることです。

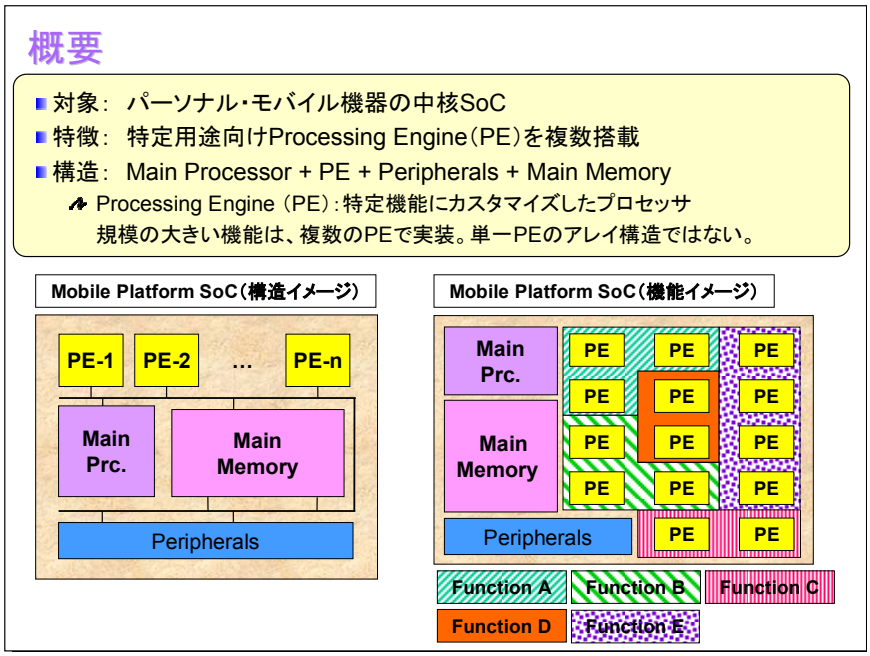


図表 3-3 SoC モデルの作成目的

3-2-2 概要（図表 3-4）

モバイル・プラットフォーム SoC モデルとは、パーソナル・モバイル機器の中核 SoC における将来像です。特徴は、特定用途向けの Processing Engine（PE）を複数搭載している点であり、構成は「Main Processor ＋ PE ＋ Peripheral ＋ Main Memory」としています。Processing Engine（PE）とは、特定機能にカスタマイズしたプロセッサ

であり、規模の大きい機能は複数の PE で実装します。なお、単一 PE のアレイ構造ではありません。



図表 3-4　モバイル・プラットフォーム SoC モデルの概要

3-2-3　構造

モバイル・プラットフォーム SoC モデルでは、構造及び、数値パラメータを明確に定義し、Microsoft Excel に実装しました。

(1)　基礎パラメータ

面積見積りの基礎データであるトランジスタ密度に関しては、2004 年版の ITRS/ORTC 値を使用しています。また、性能見積りの基礎データであるデバイス性能に関しては、ITRS/PIDS における Low Operating Power Logic の値を用いました。

(2)　面積

ダイ・サイズは、Technology Node が進んでも一定であると想定しました。一定とした理由は、製造コストの上昇を抑えるためです。また、面積オーバーヘッドも一定としました。面積オーバーヘッドとは、ダイ・サイズにおいて、I/O、アナログ回路、電源回路（電源配線を含む）が占める割合を意味します。

(3)　Main Processor

Main Processor は、SoC 全体を制御するための汎用プロセッサです。Main Processor の回路規模は配線遅延増加の影響を避けるために、Technology Node が進んでも一定としました。また、搭載個数も Technology Node に依存せず、1 個（一定）であるとしました。

(4)　Processing Engine（PE）

PE は、特定機能にカスタマイズしたプロセッサです。PE の回路規模は配線遅延増加の影響を避けるために、Technology Node が進んでも一定としました。搭載個数は、面積条件が許す最大個数搭載としたので、Technology Node が進むにしたがい、PE の総回路規模は増加します。PE の演算能力を示す指標である動作周波数は、デバイス性能に対して比例で向上するとしました。

(5)　Main Memory

Main Memory の容量は、PE 数に比例して増加するとしました。

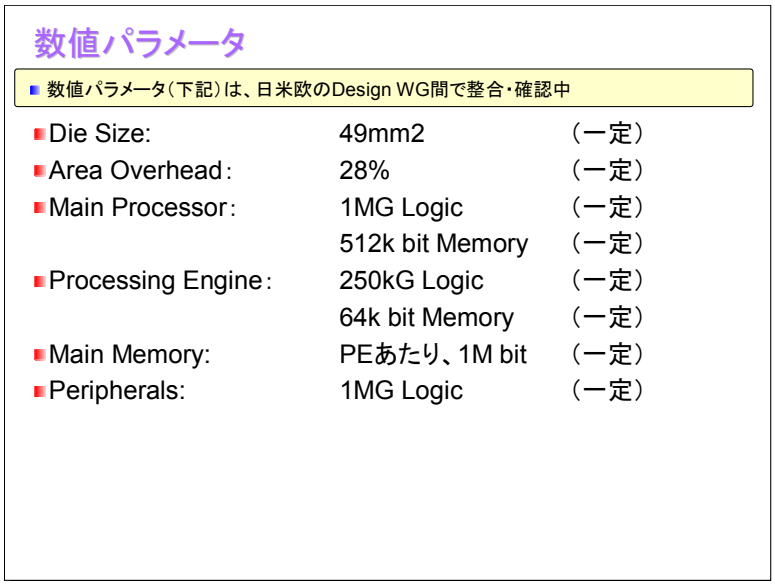
(6)　Peripheral

Peripheral は、SoC 外とのインタフェース用回路です。ただし、I/O 回路は Peripheral ではなく面積オーバーヘッド

に含まれます。Peripheral の回路規模は、Technology Node に依存せず一定であるとしました。

3-2-4 数値パラメータ

SoC モデルにおける数値パラメータを、図表 3-5 に示します。数値パラメータのレベルでは、日本案に基づき、日米欧で整合中です。



図表 3-5 モバイル・プラットフォーム SoC モデルの数値パラメータ

3-2-5 モバイル・プラットフォーム・SoC が示す未来像

(1) PE 数

PE の搭載個数は、面積条件が許す最大個数搭載としたので、下式で算出できます。

$$(\text{PE数}) = \frac{\text{チップ面積} - \text{オーバーヘッド面積} - \text{主プロセッサ面積} - \text{周辺回路面積}}{\text{PE面積} + \text{PEあたりのメモリ面積}}$$

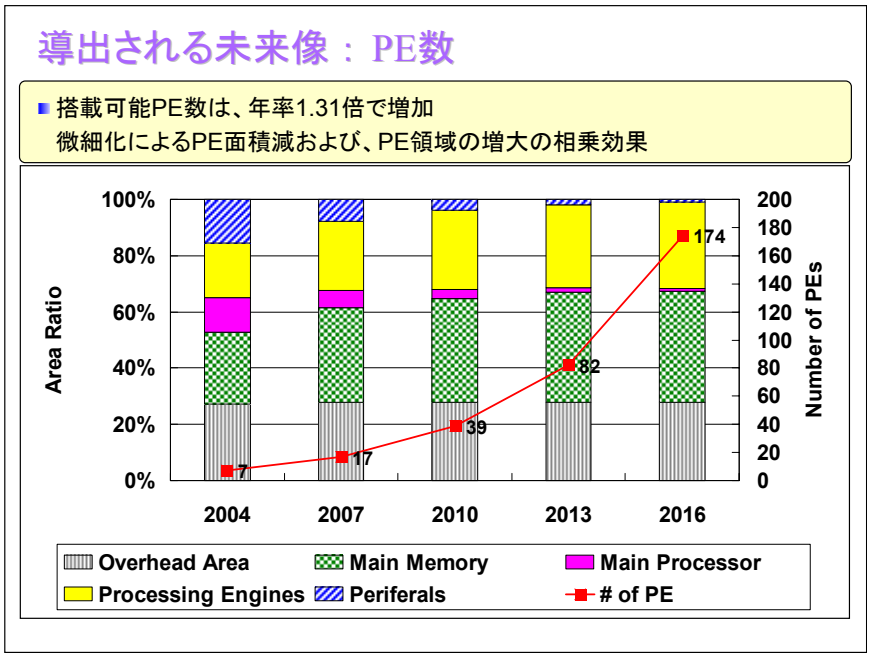
PE の搭載個数は、微細化とともに増加し、2004 年では 7 個ですが、2016 年には 174 個と大幅に増加します。増加率で言えば、年率 1.31 倍となります（図表 3-6）。

(2) 総演算能力

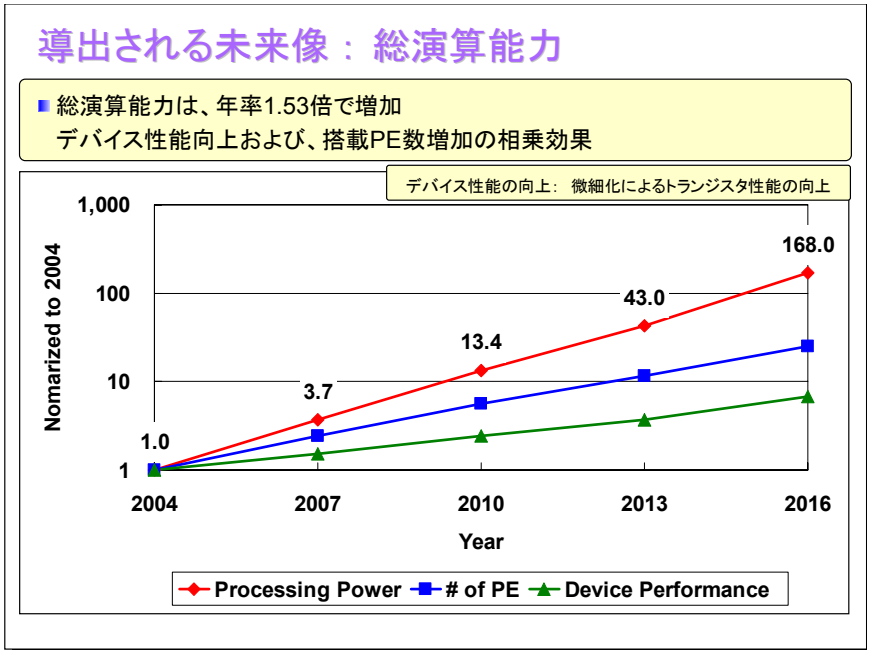
全PEを同時に動作させた場合における演算能力を総演算能力とするならば、総演算能力はPE 数及び、デバイス性能向上に比例増加するとみなせます。計算式で表現するならば、下式のとおりです。

$$(\text{最大演算能力}) \propto (\text{デバイス性能}) * (PE\text{数})$$

総演算能力は、年率 1.53 倍で増加し、2016 年には 2004 年の 168 倍に達します（図表 3 -7）。なお、消費電力の問題を考えると全PE の同時実行は困難ですが、SoC 性能を把握するための指標として「総演算能力」を設定しました。



図表 3-6　導出される未来像:PE 数



図表 3 -7　導出される未来像:総演算能力

3-3　設計生産性要求値

3-3-1　検討対象、方法、最終目標

「モバイル・プラットフォーム SoC モデル」を検討対象とし、「SoC 開発における HW 設計工数 ＝ 一定」を実現するために必要な「設計生産性向上率＋IP 再利用率」を検討しました。本検討の目的は、設計生産性要求値を設計フローにマッピングし、設計技術ロードマップ（開発すべき設計技術と実現時期）の定量的分析の実現にあります。

3-3-2　仮定・前提

(1)　大前提:SoC 開発工数 ＝ 一定

検討の大前提は、「SoC 開発工数 ＝ 一定」としました。ここに、開発工数とは、HW 開発工数のみを含みます。

すなわち、HW/SW 協調検証及び、SW 開発の工数は対象外としました。

(2) 新規設計に要する工数

新規設計に要する工数は、論理回路規模に比例するとしました。すなわち、論理回路以外のメモリやアナログ回路などは、IP として提供されると想定しています。

(3) 設計資産（IP）再利用に必要な工数

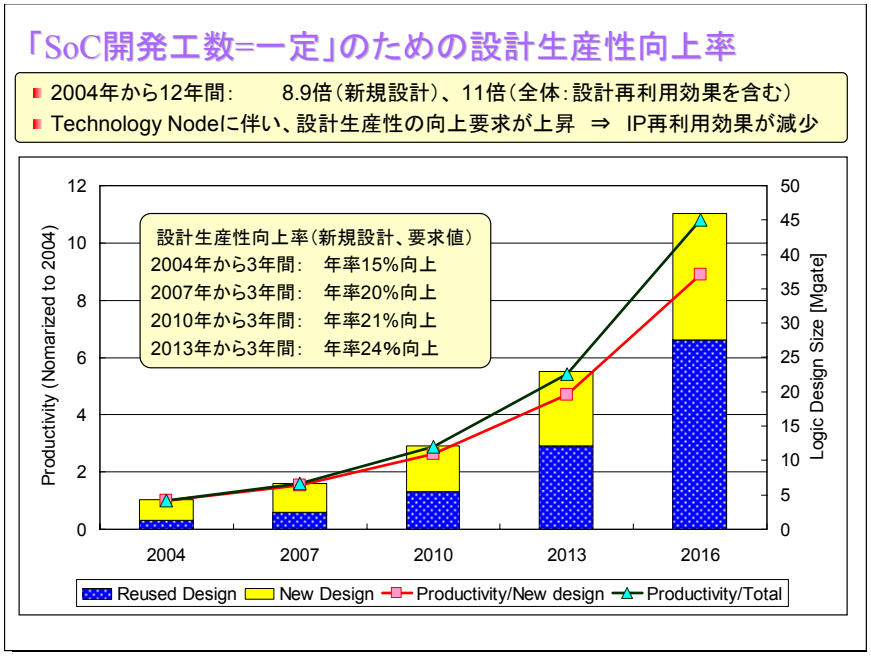
IP 再利用に必要な工数は、同一規模回路の新規設計に要する工数の 50%としました。再利用工数には、テクノロジ・ポーティング（Soft IP の場合）、検証、評価などの工数を含みます。

(4) IP 再利用

IP 再利用率は、2004 年度が 20%であり、2016 年に 60%に達するまでリニアに増加するとしました。

3-3-3 「SoC 開発工数=一定」のための設計生産性向上率

今回の仮定を前提とすると新規設計における設計生産性は、2004 年から 16 年間で 8.9 倍とする必要があります。また、設計再利用効果を含めた場合における必要向上率は、今後 16 年間で 11 倍となります。いずれも、極めてチャレンジングな目標値です（図表 3-8）。



図表 3-8 「SoC 開発工数=一定」のための設計生産性向上率

3-4 設計生産性向上のための解決策検討

3-4-1 検討対象、方法、最終目標

「システム・アーキテクチャ設計」を検討対象とし、「Canonical Design Flow の詳細化（サブ工程への分割）」及び、「各サブ工程の生産性向上のための解決策を検討・定義」を実施しました。

本検討の最終目的は、「設計生産性要求値に基づく、設計技術ロードマップの定量分析」ですが、今年度はシステム・アーキテクチャ設計のみを範囲とする予備的な検討としました。

3-4-2 Canonical Design Flow

(1) 内容

Canonical Design Flow とは、ロードマップ検討のために、現在及び、将来の SoC 設計フローを、定義・明確化し

た図及び、解説です。一般的に理解されている設計フローを、図及び、文書で明確化しています。

(2) 目的

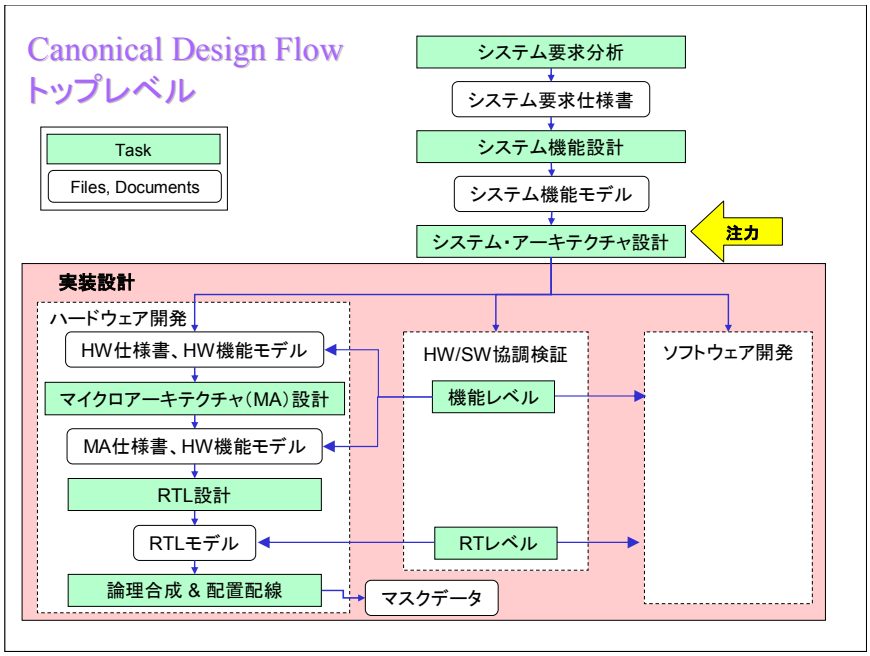
Canonical Design Flow を作成する目的は、設計技術ロードマップの基本背景の提示及び、設計コストに関する議論の基盤提供です。設計技術ロードマップの基本背景提示により、設計技術に関する議論の共有化及び、伝達が容易化できます。

(3) 開発状況

Canonical Design Flow のトップ・レベル・フローは、日本が作成・提案し、2003 年版の ITRS から掲載しています。04 年度では、設計生産性向上策検討の観点から、中間レベル・フローの詳細化に着手しました。今年度は、特にシステム・アーキテクチャ設計に注力しました。

3-4-3 Canonical Design Flow：トップ・レベル

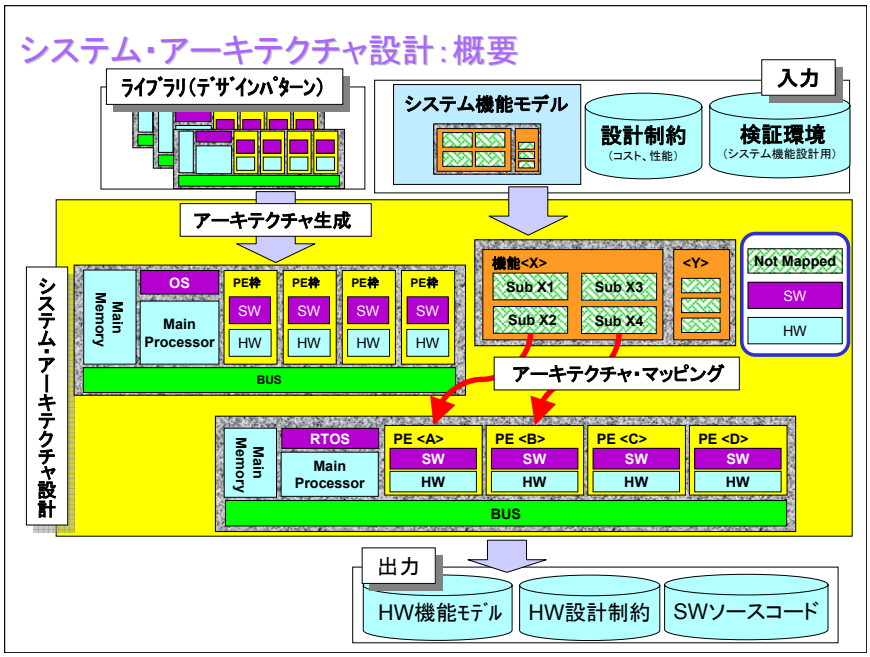
図表 3 -9 は、Canonical Design Flow の全体像を示すトップ・レベル・フロー図です。今年度は、本フロー図におけるシステム・アーキテクチャ設計の詳細化に注力しました。



図表 3 -9 Canonical Design Flow：トップ・レベル

3-4-4 システム・アーキテクチャ設計：概要

図表 3-10 は、システム・アーキテクチャ設計の概要を示します。本工程の入力は、①システム機能モデル、②設計制約、③検証環境及び、④デザイン・パターンです。一方、出力は、⑤HW 機能モデル、⑥HW 設計制約及び、⑦SW ソース・コードです。システム・アーキテクチャ設計では、デザイン・パターンから適切なアーキテクチャを選択し、システム機能を SW/HW に分割して、適切な PE に割り付けます。



図表 3-10　システム・アーキテクチャ設計：概要

(1)　システム機能モデル

システムの機能を記述したモデルであり、システム・アーキテクチャ設計の着手段階ではソフトウェアとハードウェアにまだ分割されていません。

(2)　設計制約

システムの設計制約を記述したモデルであり、システムが満たすべき製造コスト、性能などを規定します。

(3)　検証環境

システム機能モデルを作成・評価した段階で使用した検証環境です。設計成果物がシステム機能モデルと等価であるかを確認するために、システム機能モデルの検証に利用した環境を再利用します。検証環境の再利用は、検証コスト削減の点でも重要です。

(4)　デザイン・パターン

SoC のアーキテクチャを事前に開発・蓄積したデータベースです。システム・アーキテクチャ設計では、デザイン・パターンから適切なアーキテクチャを選択します（図表 3-11）。なお、**図表 3-11** 中の「PE（外枠のみ）」における「外枠のみ」とは、「PE が機能割り付け前の状態である」を意味します。

(5)　HW 機能モデル

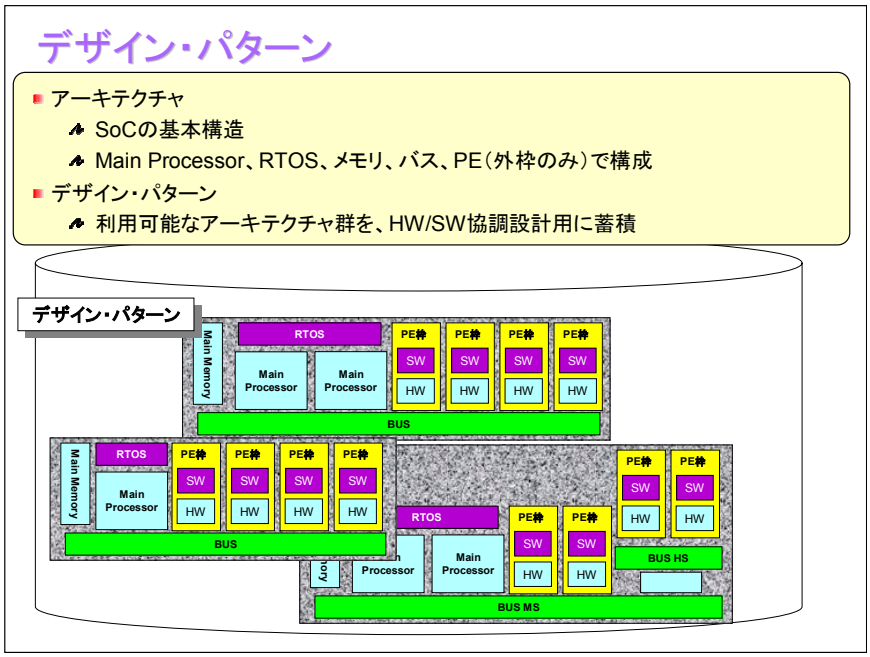
HW として実現すべき機能を記述したモデルです。

(6)　HW 設計制約

HW が満たすべき設計制約を、システム・レベルの機能制約からブレークダウンしたモデルです。

(7)　SW ソース・コード

SW として実現すべき機能を記述したソース・コードです。

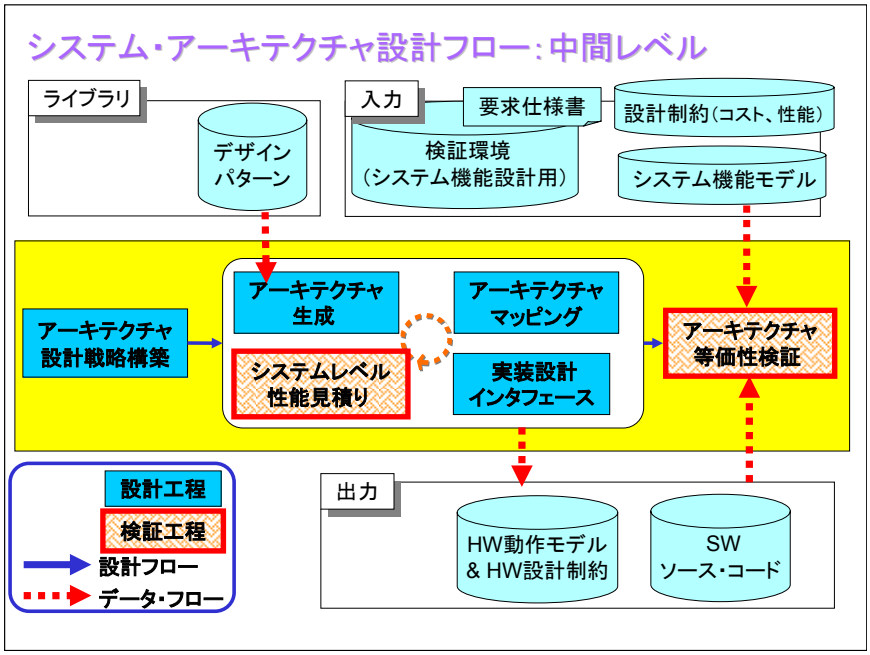


図表 3-11　デザイン・パターン

3-4-5　システム・アーキテクチャ設計：中間レベル・フロー

システム・アーキテクチャ設計を明確化するために、中間レベルのフロー図及び、サブ工程の実施内容の定義を作成しました。図表 3-12 は、システム・アーキテクチャ設計における中間レベル・フローを示しています。中間レベル・フローでは、設計工程と検証工程を区別する形式で表示します。

また、各サブ工程の実施内容を、下記(1)－(6)に示すとおりに明確に定義しました。



図表 3-12　システム・アーキテクチャ設計フロー：中間レベル

(1)　アーキテクチャ設計戦略構築

アーキテクチャ設計戦略構築では、システム・レベル性能見積りにおいて、評価すべき項目と、評価を実施するための手段（シミュレーション環境）を定義します。

(2) アーキテクチャ生成

アーキテクチャ生成では、デザイン・パターン中から要求に合致するアーキテクチャを選定します。

(3) アーキテクチャ・マッピング

アーキテクチャ・マッピングでは、機能を分割し、HW/SW のどちらで実現するかを判断して、適切な PE に割り付けます。

(4) システム・レベル性能見積り

システム・レベル性能見積りでは、「選定したアーキテクチャ＋マッピングが、目標性能を満たすか」を判定します。改善すべき点がある場合は、アーキテクチャ生成、アーキテクチャ・マッピングを再実行します。

(5) 実装設計インタフェース

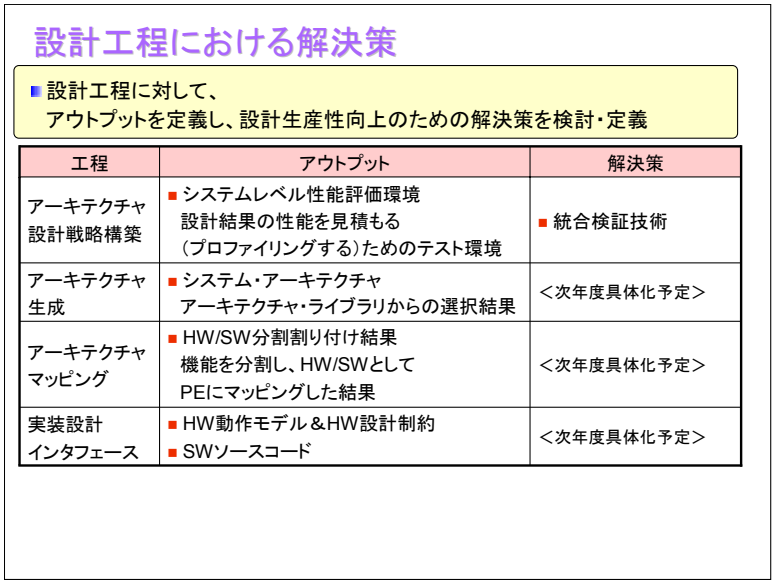
実装設計インタフェースでは、「選定したアーキテクチャ＋マッピング」を、後工程（実装設計）用に変換します。

(6) アーキテクチャ等価性検証

アーキテクチャ等価性検証では、設計結果である HW 動作モデル及び、SW ソース・コードが、システム機能モデルと等価であるかを検証します。

3-4-6 システム・アーキテクチャ設計：設計工程における解決策

中間レベル・フローにおける設計工程に対しては、各工程のアウトプットを明確化し、設計生産性向上のための解決策を検討・定義しました。図表 3-13 は、設計工程に対する検討結果のまとめを示します。例えば、アーキテクチャ設計戦略構築におけるアウトプットは、システム・レベル性能評価環境、すなわち、（プロファイリングする）ためのテスト環境となります。また、設計生産性向上に必要な解決策は、統合検証技術であると考えております。

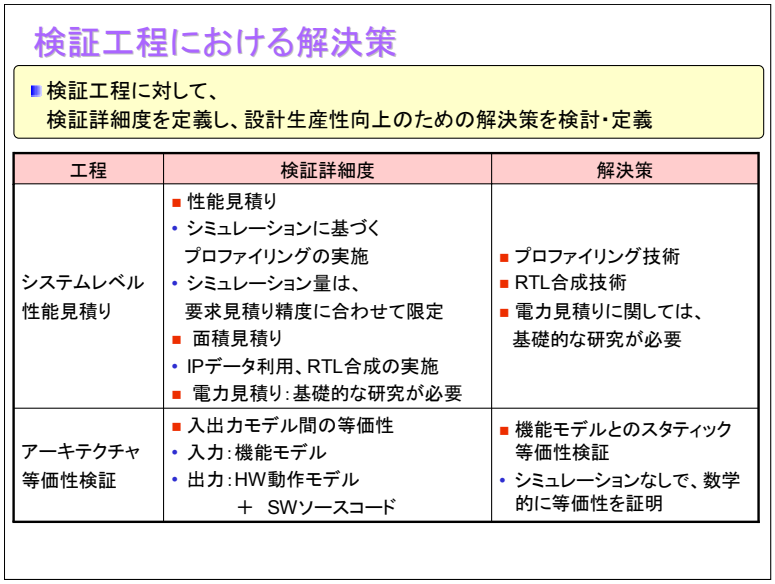


図表 3-13 設計工程における解決策

3-4-7 システム・アーキテクチャ設計：検証工程における解決策

中間レベル・フローにおける検証工程に対しては、各工程の検証詳細度を明確化し、設計生産性向上のための解決策を検討・定義しました。

図表 3-14 は、検証工程に対する検討結果のまとめを示します。例えば、システム・レベル性能見積りにおける性能見積りは、シミュレーションに基づくプロファイリングにより計測するが、詳細度は、要求見積り精度に合わせて限定したシミュレーションで決定します。また、設計生産性向上に必要な解決策は、プロファイリング技術の確立と考えております。



図表 3-14　検証工程における解決策

3-4-8　システム・アーキテクチャ設計：解決策の定義

システム・アーキテクチャ設計における設計生産性向上のための解決策としては、①統合検証技術、②プロファイリング技術、③RTL 合成技術、④機能モデルとのスタティック等価性検証が抽出できました。

(1)　統合検証技術

統合検証技術は、システムからゲート・レベルの全工程を範囲とする検証技術です。設計工程間のモデル等価性を確保し、検証に要する工数を削減するために、検証環境を工程毎の抽象度を越えて再利用する技術が重要となります。検証環境には、テストベンチ、ランダム生成を含むテストパターン、レファレンスモデル、アサーションなどを含みます。

(2)　プロファイリング技術

プロファイリング技術は、シミュレーションによる HW 間の通信量計測によりシステム性能を見積もる技術であり、システム・アーキテクチャ設計において、設計結果の評価に必要な技術です。

(3)　RTL 合成技術

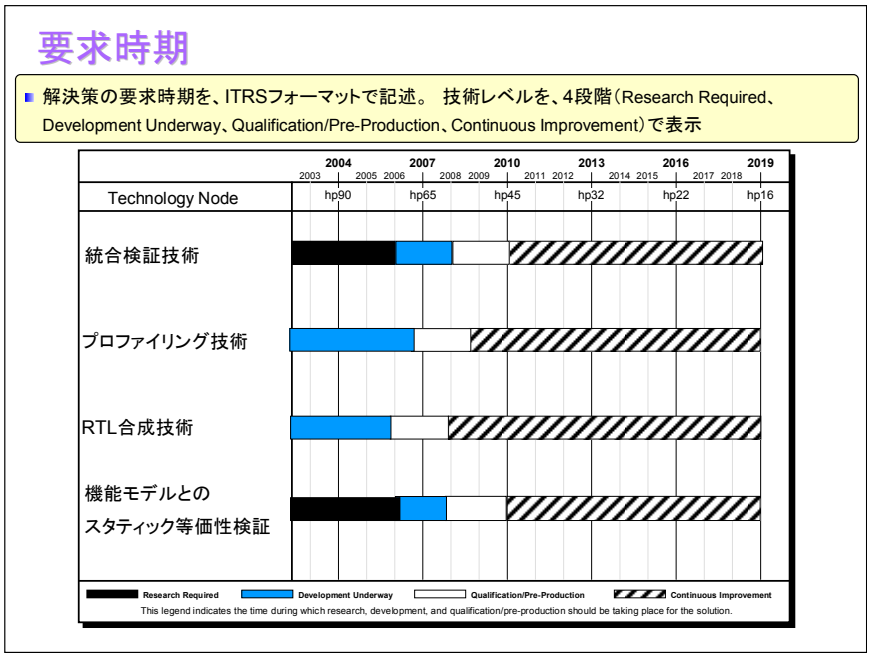
RTL 合成技術は動作記述から RTL を合成する技術であり、システム・アーキテクチャ設計では HW の「動作周波数及び、面積を見積もる手段として利用します。

(4)　機能モデルとのスタティック等価性検証

「機能モデルとのスタティック等価性検証システム」とは、機能モデルに対する設計結果の等価性を検証する技術であり、設計結果である HW 動作モデル＋SW ソース・コードがシステム機能モデルに対する一致を、シミュレーションなしで数学的に証明します。

3-4-9　システム・アーキテクチャ設計：要求時期

図表 3-15 は、各解決策の要求時期を、ITRS のフォーマットで記述した図です。各解決策の技術レベルを、4 段階（Research Required、Development Underway、Qualification/Pre-Production、Continuous Improvement）で示しています。



図表 3-15　解決策の要求時期

3-5　まとめ、今後の計画

3-5-1　まとめ

(1)　モバイル・プラットフォーム SoC モデル

将来の SoC 構造・規模を定量分析可能な形式で予測したモデルとして、モバイル・プラットフォーム SoC モデルを構築しました。本モデルは ITRS Design WG に提示済みであり、2005 年版への掲載に向け賛同と評価を得ております。

(2)　設計技術ロードマップ

設計技術向上（設計生産性向上）に対する要求値の明確化に着手しました。今年度は、システム・アーキテクチャ設計に注力し、中間レベル Canonical Design Flow と、各サブ工程における技術要求ロードマップを検討しました。

3-5-2　今後の計画

(1)　モバイル・プラットフォーム SoC モデル

モバイル・プラットフォーム SoC モデルに関しては、2005 年版 ITRS 掲載に向け、ブラッシュ・アップを図りたいと考えています。

(2)　設計技術ロードマップ

設計技術ロードマップに関しては、検討の枠組みが完成したと考えており、全設計工程に展開する予定です。具体的には、全設計工程における中間レベル・フローを作成し、各サブ工程に対する技術要求と実現時期を定義したいと考えております。