システムレベル設計概説

立命館大学 理工学部　電子情報デザイン学科
(Dept. of VLSI System Design)

福 井 正 博

LSIの設計困難度の増大

ITRS2001: Additional Design Technology Requirements

Year of Production	*2001*	*2002*	*2003*	*2004*	*2005*	*2006*	*2007*	*2010*	*2013*	*2016*	*Driver*
DRAM ½ Pitch (nm)	*130*	*115*	*100*	*90*	*80*	*70*	*65*	*45*	*32*	*22*	
MPU/ASIC ½ Pitch (nm)	*150*	*130*	*107*	*90*	*80*	*70*	*65*	*50*	*35*	*25*	
MPU Printed Gate Length	*90*	*75*	*65*	*53*	*45*	*40*	*35*	*25*	*18*	*13*	
MPU Physical Gate Length	*65*	*53*	*45*	*37*	*32*	*28*	*25*	*18*	*13*	*9*	
SOC new design cycle (months)	12	12	12	12	12	12	11	11	10	9	*SOC*
SOC Logic Mtr per designer-year (10-person team)	1.2			2.6			5.9	13.5	37.4	117.3	*SOC*
SOC dynamic power reduction beyond scaling (X)	0			1.5			2.5	4	7	20	*SOC*
SOC stanby power reduction beyond scaling (X)	2			6			15	30	150	800	*SOC*
% Test covered by BIST	20			30			45	60	75	90	*MPU, SOC*

Mtx --- M transistors

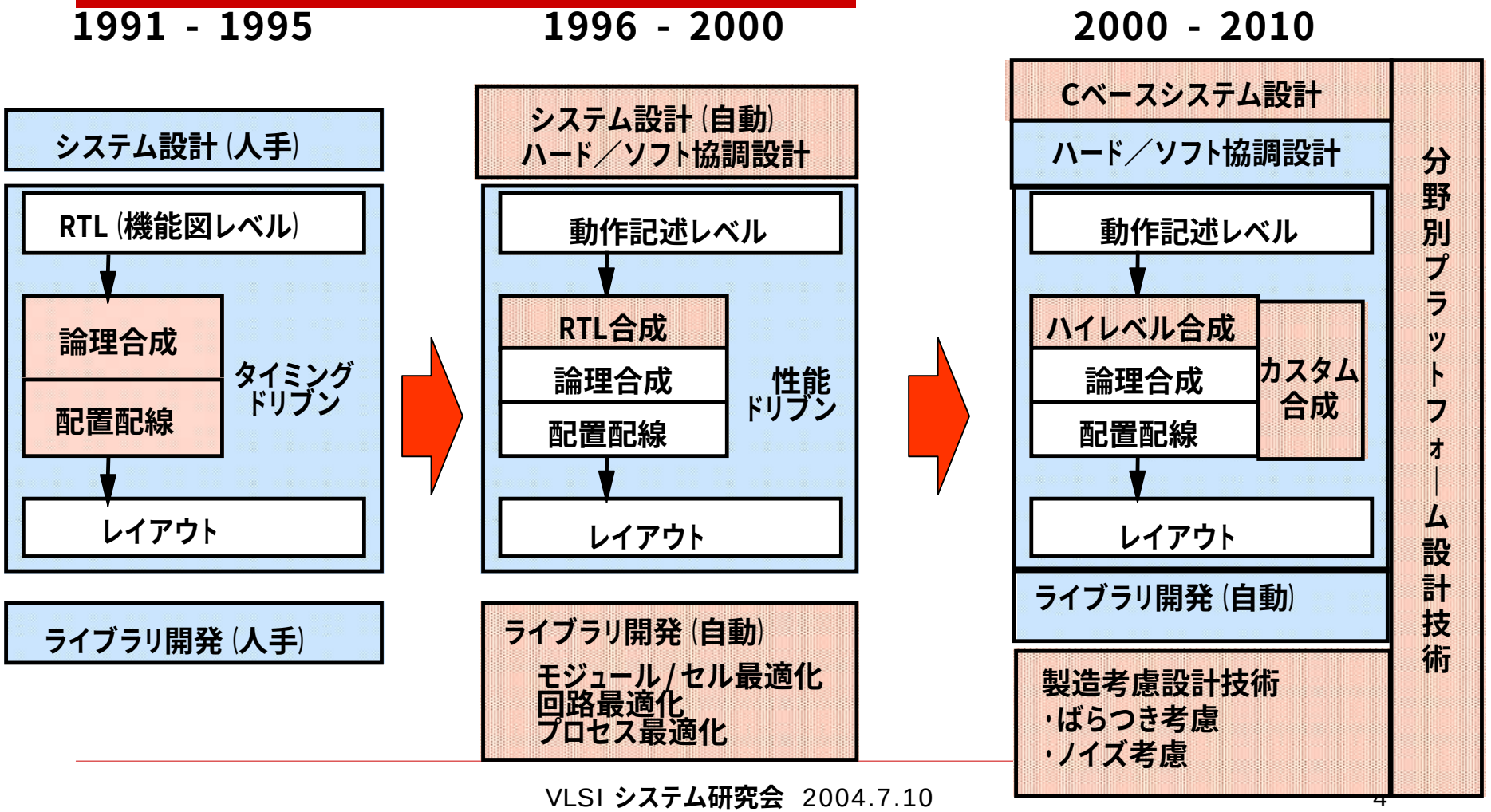
White --- Manufacturable Solutions Exist, and Are Being Optimized
Yellow --- Manufacturable Solutions are Known
Red --- Manufacturable Solution are NOT Known

SOURCE: International Technology Road Map for Semiconductors 2001 - Design

システムレベル設計が要望される背景

- 設計危機
  - LSIがどんどん大規模化しているのに対して、設計の高効率化がついていってない.
    → 高抽象度での設計が求められる.
- 消費電力、機能検証の危機
  - システムの大規模化、高速化に伴い、消費電力の大幅な増大と機能検証の困難化への対応が課題となる.
    →設計の源流での設計最適化が求められる.

設計技術の変遷


1991 - 1995
1996 - 2000
2000 - 2010
システム設計（人手）
RTL（機能図レベル）
論理合成
配置配線
タイミングドリブン
レイアウト
ライブラリ開発（人手）
システム設計（自動）
ハード／ソフト協調設計
動作記述レベル
RTL合成
論理合成
配置配線
性能ドリブン
レイアウト
ライブラリ開発（自動）
モジュール/セル最適化
回路最適化
プロセス最適化
Cベースシステム設計
ハード／ソフト協調設計
動作記述レベル
ハイレベル合成
論理合成
配置配線
カスタム合成
レイアウト
ライブラリ開発（自動）
製造考慮設計技術
・ばらつき考慮
・ノイズ考慮
分野別プラットフォーム設計技術

システムとは？

- 辞書では

System ＝ **組織、系統**

社会システムや教育システムから、機械システム、化学プラントのシステム、ソフトウエアシステムなど各分野で、系統だったものを指す用語として使われる.

LSIが対象とするシステム

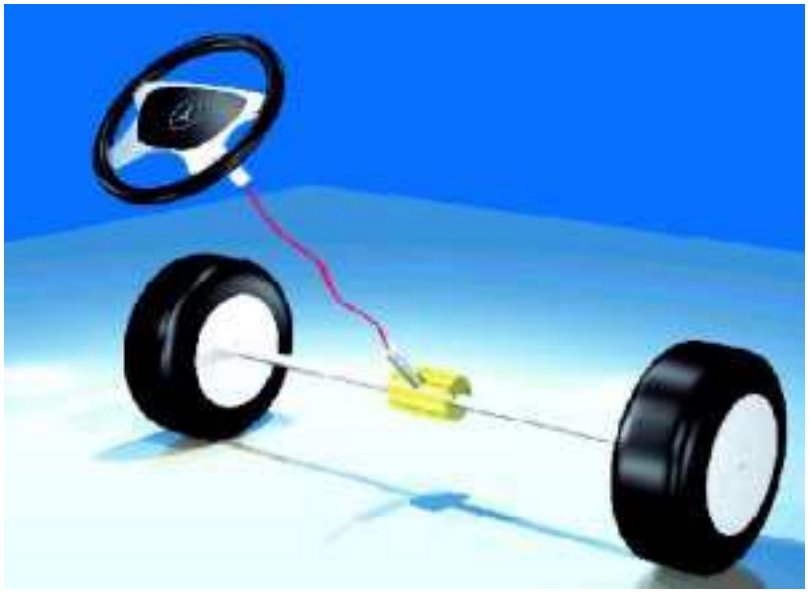
- システムLSIが対象とするもの
  - 情報家電 (携帯電話、デジタルカメラ、…)
  - コンピュータ、通信、セキュリティ、放送、交通
  - 車載機器、ロボット、医療、各種機器制御

→ システムというのはそれぞれ使用者の前に存在する製品そのものを意味する.

(例) 携帯電話システム＝携帯電話そのもの

システムの例

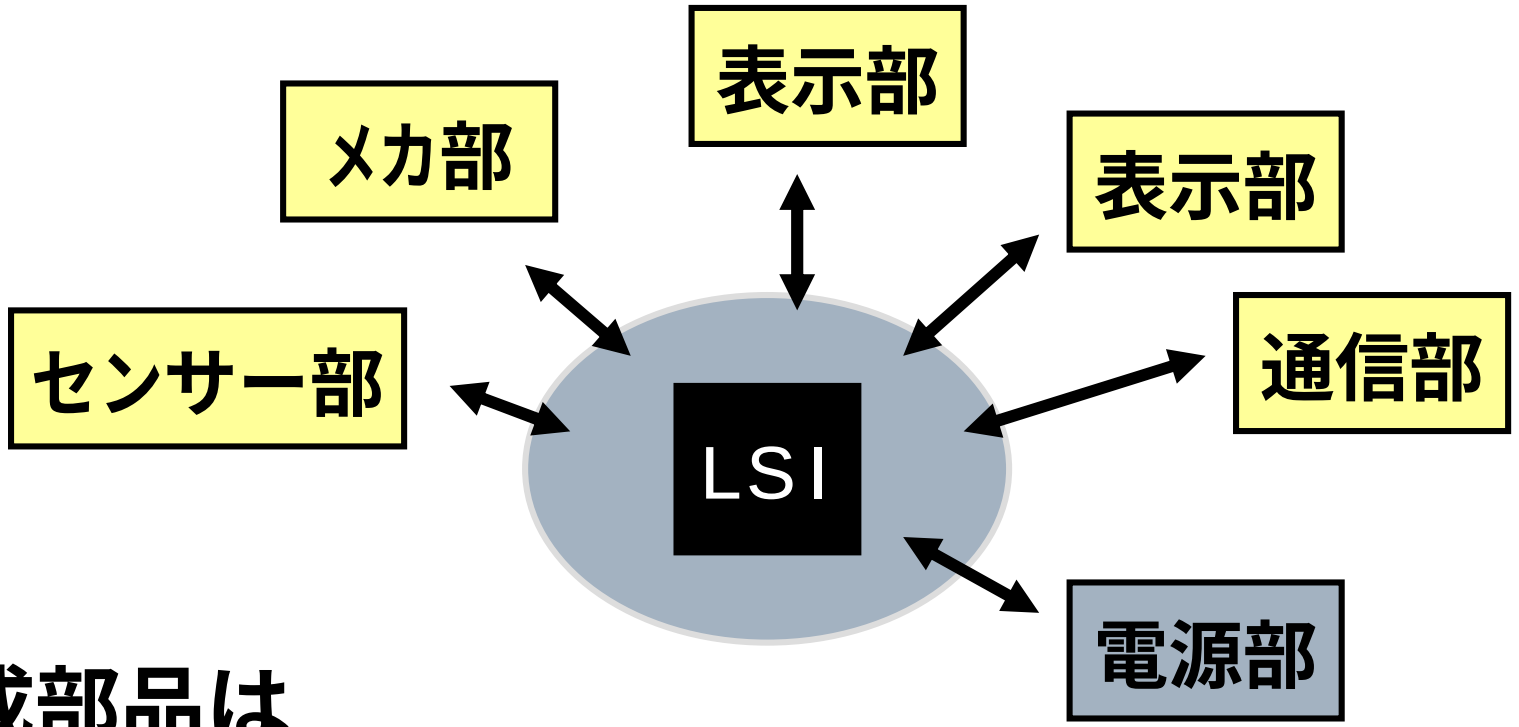
自動車用システムの例



Stair by wire

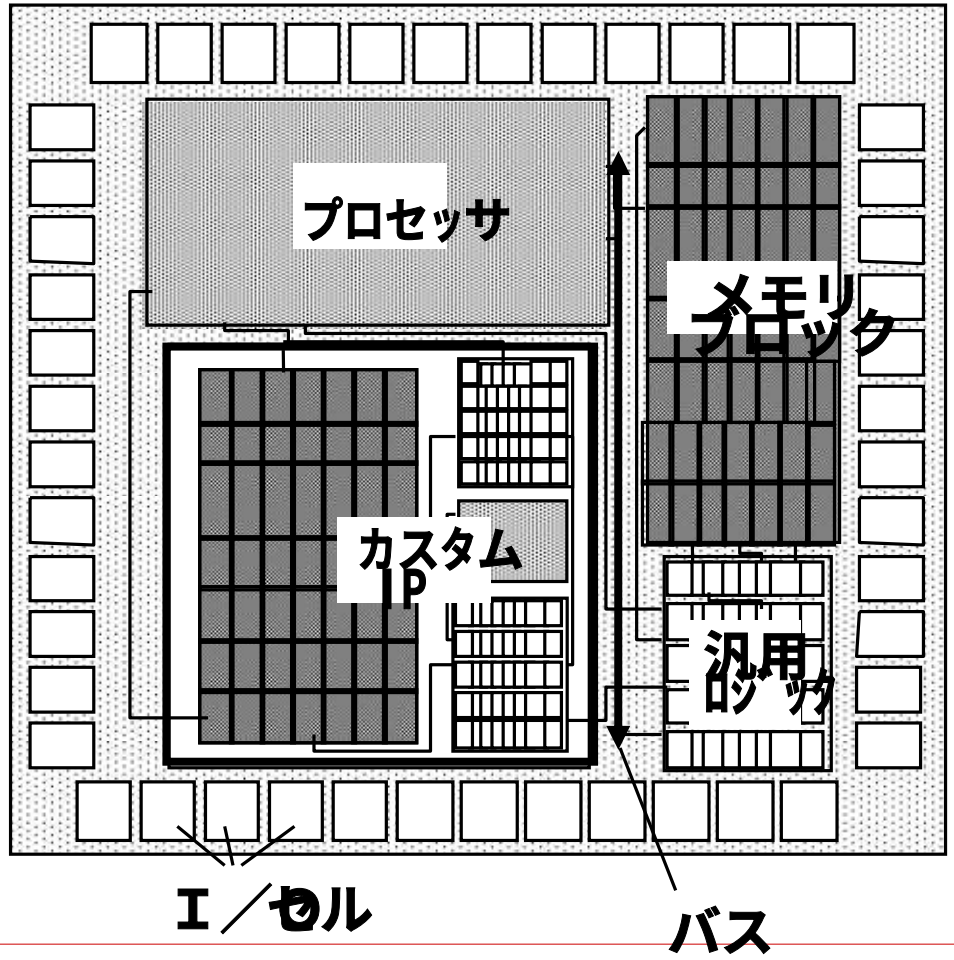
Sensor in tire

システムイメージ

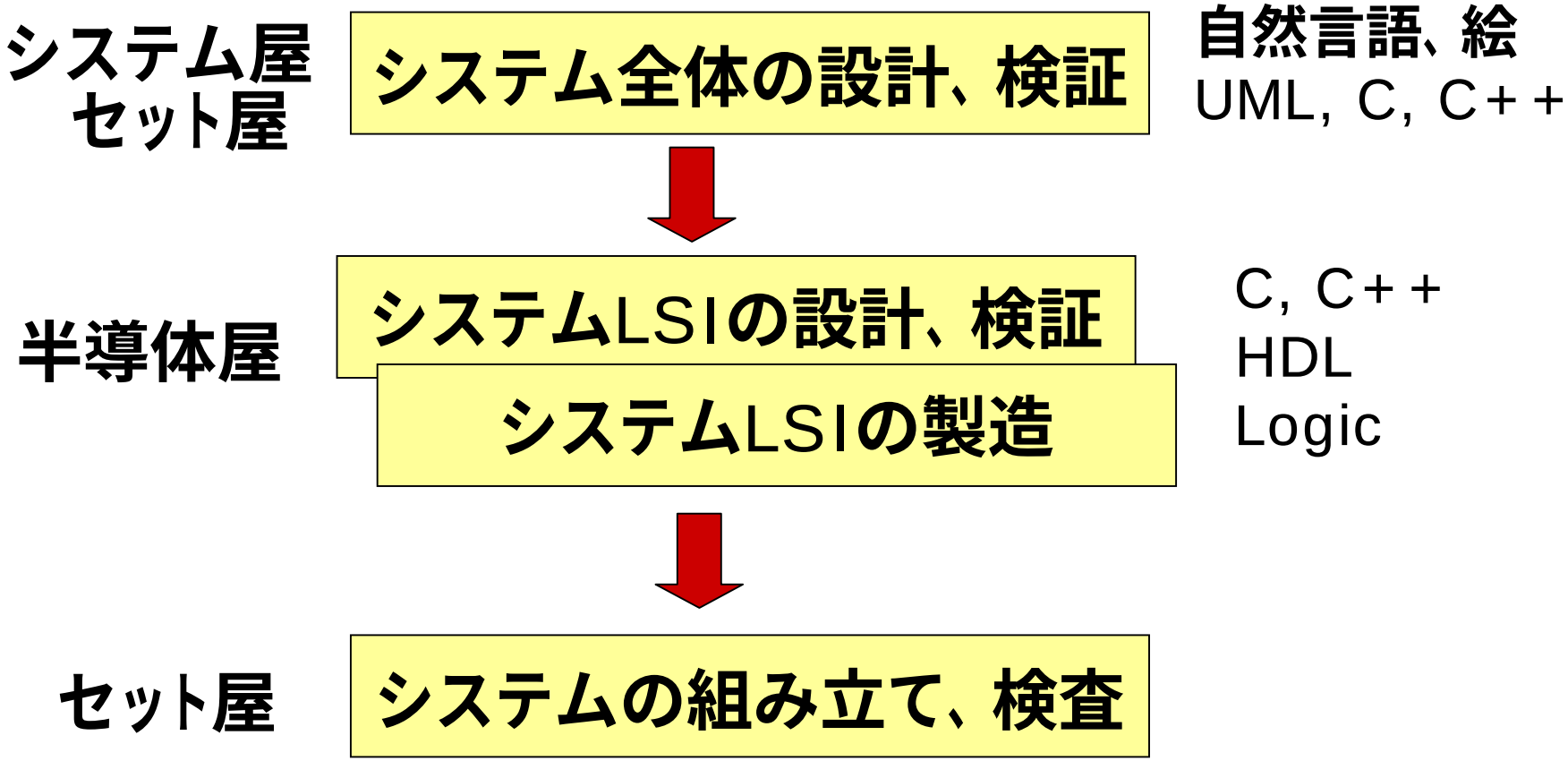


構成部品は
外部装置とシステムLSI

システムLSIのイメージ



システム設計と製造の流れ



システム仕様の受け渡し

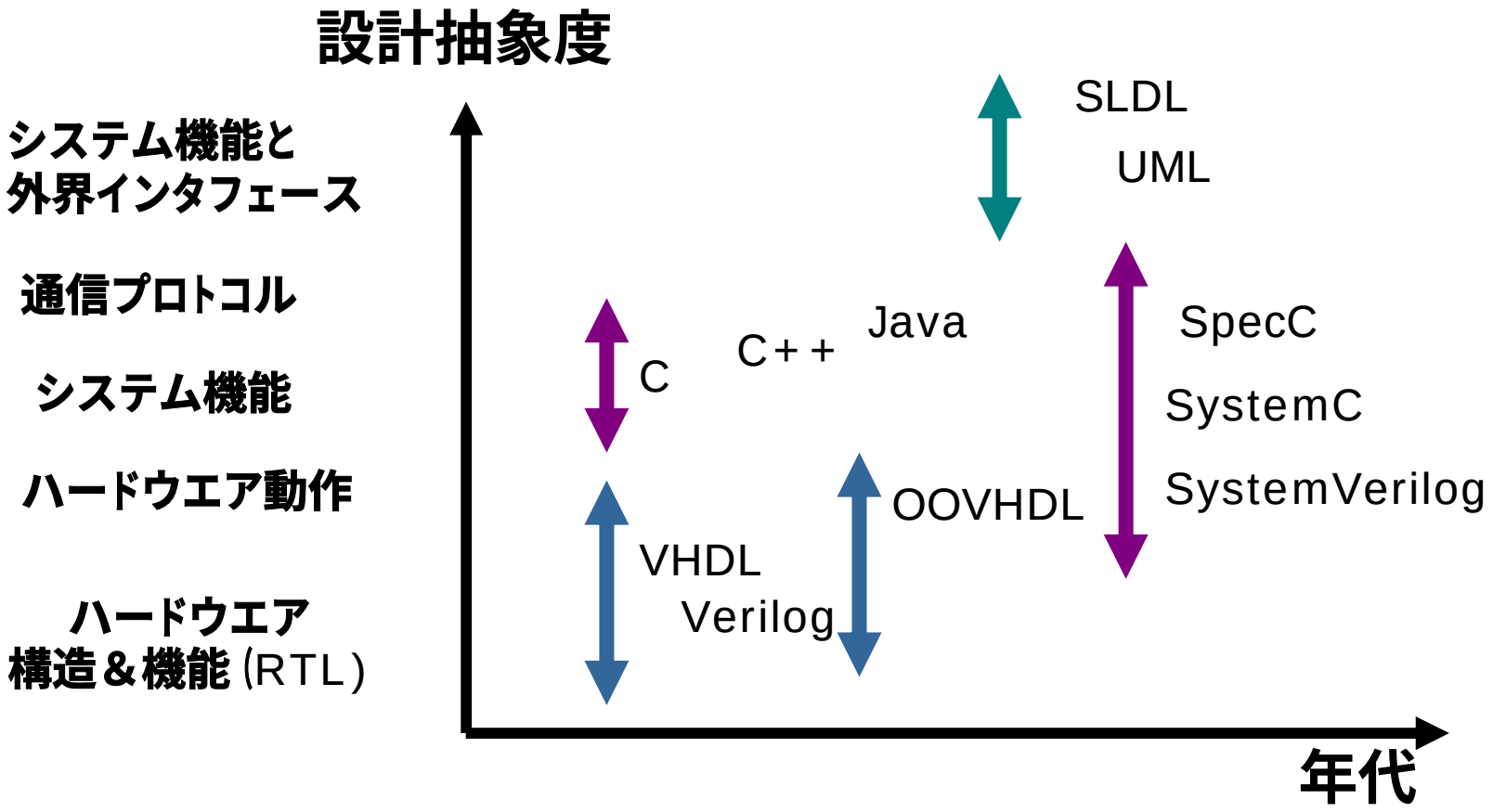
- システム屋：システムの仕様決定
- 半導体屋：仕様を満たすLSIを作る

→　設計言語は、設計、検証、コミュニケーションの手段

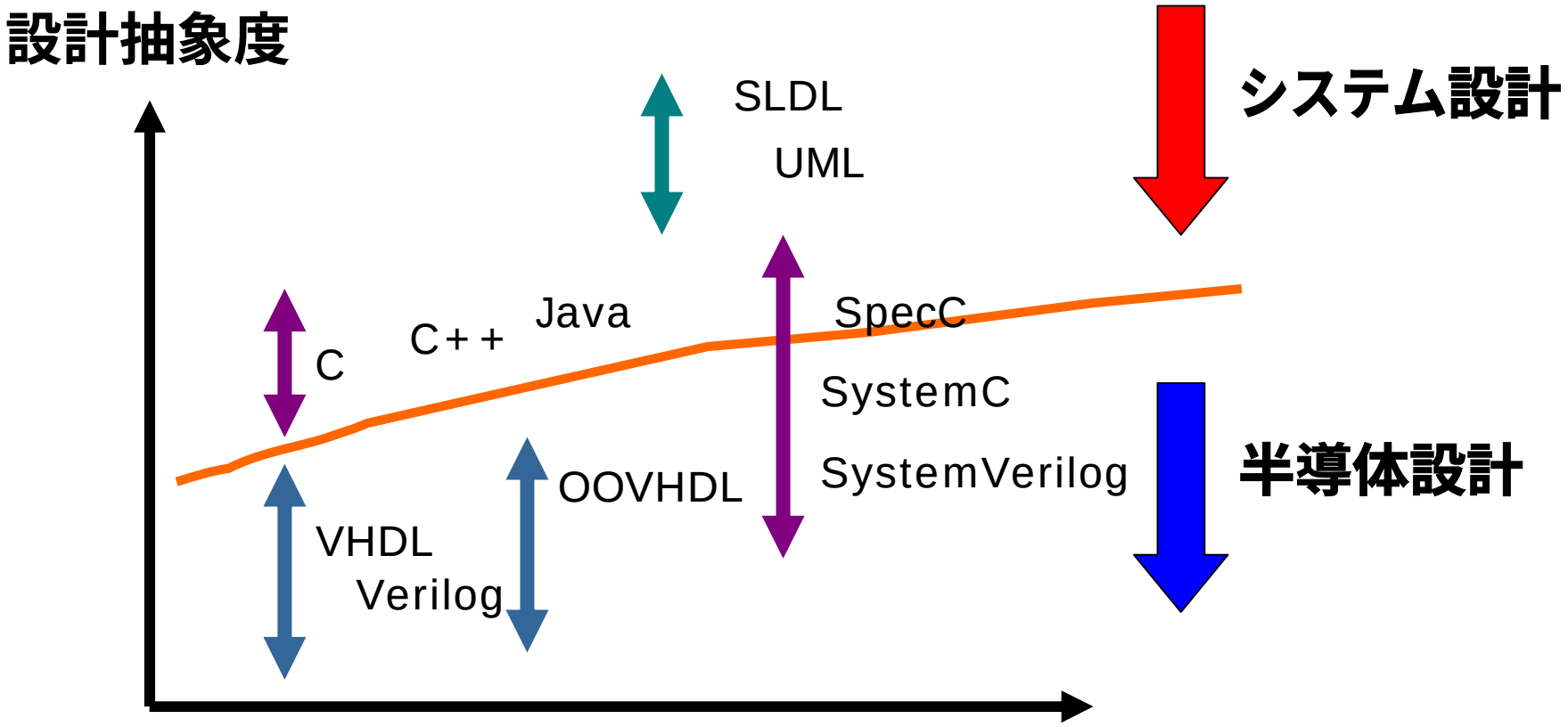
システム仕様の例

- 携帯電話システム
  - 携帯意電話が外部からの入力に対してどのような動作を行うかの取り決め
    - 待機時間の動作
    - メール配信の動作 (エディタ、画像圧縮伸張、カメラ)
    - 住所録
    - 通話品質の管理、制御
    - ゲーム
  - 消費電力や処理スピードに関する制約条件.

システム設計言語


設計抽象度
システム機能と
外界インタフェース
通信プロトコル
システム機能
ハードウエア動作
ハードウエア
構造＆機能 (RTL)
SLDL
UML
C
C++
Java
SpecC
SystemC
SystemVerilog
OOVHDL
VHDL
Verilog
年代

システムレベル設計言語：なぜCか？



SpecC

- システム仕様設計段階では、主にシステムの主要な機能と通信のプロトコルに関しての設計が行われる.
- Gajskiらはシステムレベルにおける動作の記述をプロセスとチャネルという概念を用いて整理した.
- それぞれの動作に関しては、スーパーＦＳＭなどを用いてわかりやすく表現することに成功している

SystemC

- 後にSystemCのコンソシアムが立ち上がり、システム仕様レベルからRTLまでの設計抽象度を扱う実用的な言語システムが開発される
- システムレベルの動作記述に関しては、SpecC に負うところが大きい

Handel C、他

- Celoxica社が開発したRTLでの設計をCで記述できる.
- 残念ながら、広く使われてはないが、HDLよりは記述効率が良いので、使い勝手の良い言語と考えることができる.
- SystemCとのインタフェースも開発している.
- 他に、シャープ他が開発したBach C、NECが開発したCyberなどが有名.

システムレベル設計の位置づけ

- LSIの設計フロー
  - **システムレベル設計**
    - ハード･ソフトの分割
    - アーキテクチャ設計
    - RTL設計
    - 論理設計
    - レイアウト設計
    - マスク設計

･システム仕様の
モデル化、検証
･(性能、電力等の推定)
･(合成)

※ 合成と推定に関してはまだ出始めの段階であり、ますますの研究開発が必要.

システムレベル設計

- エンベディッドプロセッサとソフトウエア、専用ハードウエアのモデル化
- ハードウエア･ソフトウエアの最適分割
- ハードウエアの動作合成

システムの表現モデル

- 機能モデル
  - 通信部分、計算部分のモデル化
- 時間モデル
  - Un-timed --- 順序関係のみを規定
  - Timed --- クロック精度はないが時間的概念がある
  - Cycle Accurate --- クロック精度がある
- プロセス間通信のモデル
  - バッファタイプ --- 同期のみ
  - FIFOタイプ --- 非同期動作可能
  - ブロッキング (読み書き可能になるまで待つ)
  - ノン･ブロッキング (読み書き可能でない場合は他の処理を行う)
- 割り込みの方法、活性化の方法

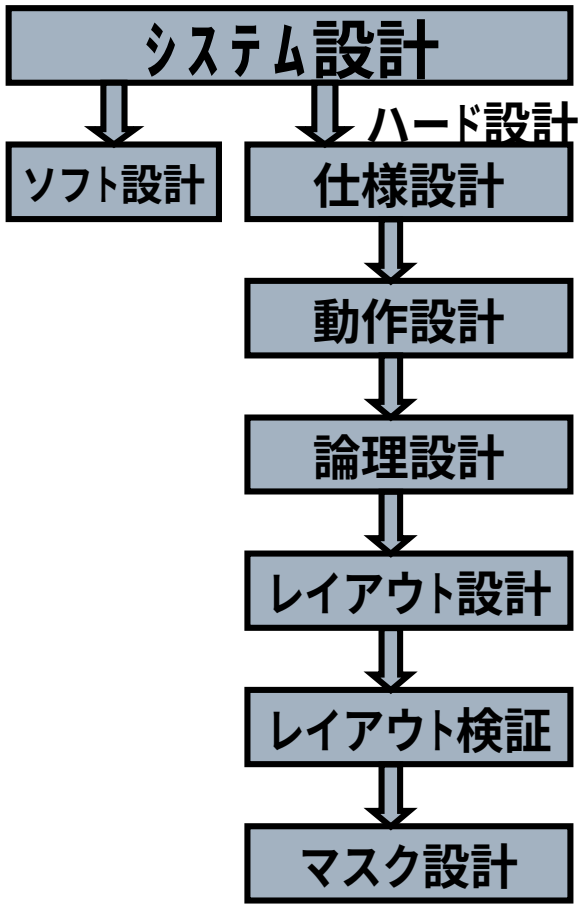
システムの動作モデル

- システムの動作レベルでは、システムを動作モジュールの機能と通信のプロトコルで表現
- それぞれの機能の実現方法に関しては制約がない.

システムレベル設計

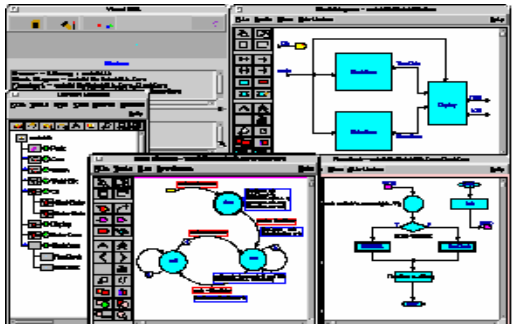
- エンベディッドプロセッサとソフトウエア、専用ハードウエアのモデル化
- ハードウエア・ソフトウエアの最適分割
- ハードウエアの動作合成

LSIの設計フロー

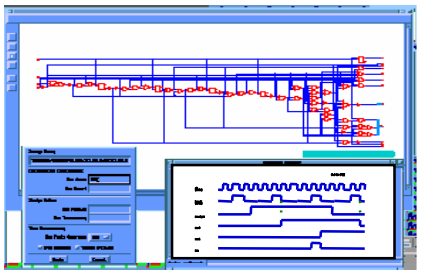



```
module KeyScan(CLOCK,RESET,SIN,SCAN,VAL)
input       [3:0]       SIN;
output      [3:0]       SCAN,VAL;
reg                     clk,rst;
always @(posedge clk or psedge rst) begin
    if(rst) r_scan <= 4'd0;
    else    case(Init)
                1'b0: r_scan <=  4'd8; // Cobstant: r_scan[3:0]
                1'b0: case( Scanning )
                          1'b0: r_scan <= { r_scan[3] , r_scan[2] }
                          1'b0: r_scan <= r_scan;
                          default: r_scan <= 4'bx;
                      endcase
                default: r_scan <= 4'bx;
            endcase
end
```

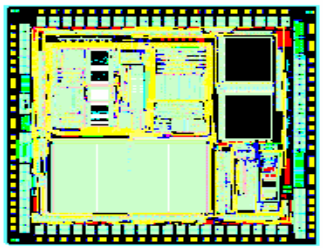
ハードウェア記述言語 (HDL)



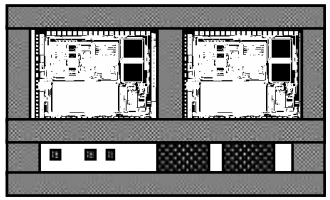
機能図



論理図と論理シミュレーション



レイアウト図



レチクル (ガラス原板)

LSIの上流設計のフロー

□ LSIの設計手順

①システム仕様設計、ハードウエアソフトウエア分割とプロセッサの選択

②ハードウエアの仕様設計 (動作、消費電力、チップサイズ、パーケージ、外部インタフェース等の決定)

③ハードウエアモジュールへの割り当てと動作順序の決定 (アロケーションとスケジューリング) .

④メモリやデータパス、それらを結ぶバス配線など、データの流れを作るハードウエアの合成.

⑤コントロール回路の論理合成.

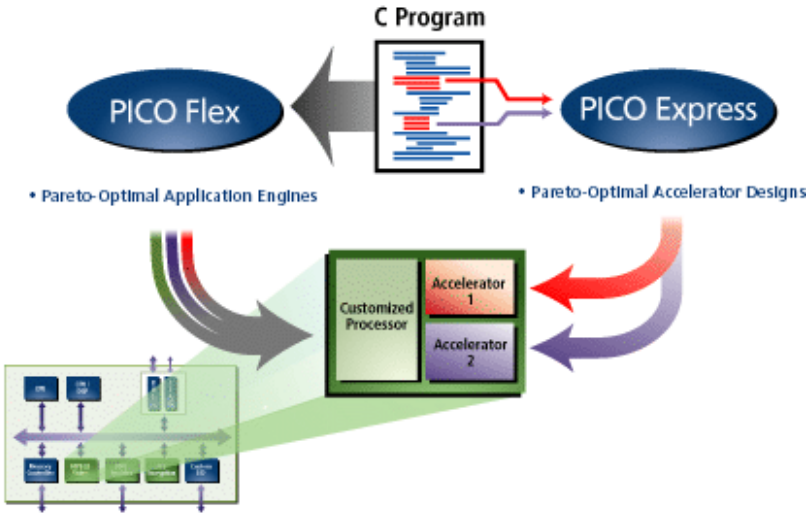
H/Sコデザインと高位合成

- システムをハード／ソフトへの分割
- ハード／ソフトそれぞれの具体化
  - ソフトはインストラクションセットモデルで表現されたプロセッサモデル上での動作シーケンスとして具体化.
  - ハードウエアは、動作レベルのハードウエア記述から、IPへのマッピングや高位合成 (モジュールへの割り当てとスケジューリング)

※ただし、高位合成の実用化はこれから

上流合成ツールの例 (1)

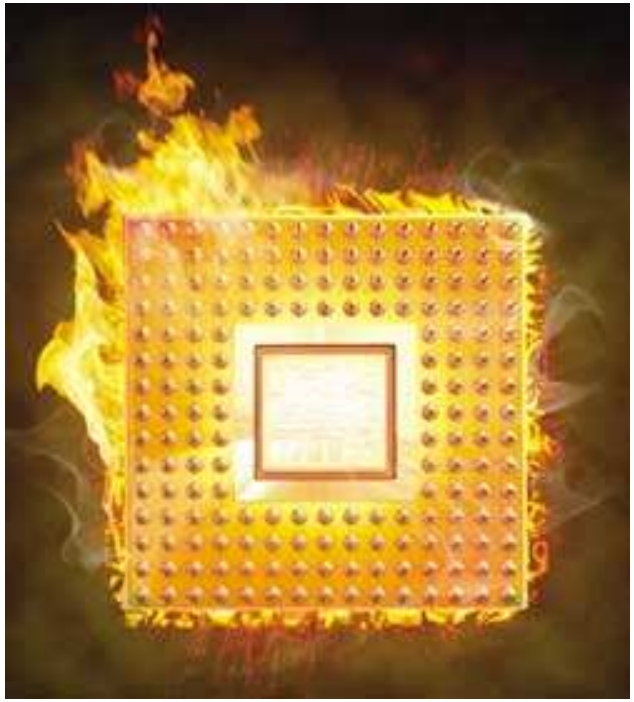
- CoWare/LISA TEK (ASIP設計ツール)
  - 設計の規格が決まる前に設計を開始できるので有利.
  - 23日でRTLとコンパイラの合成、2Wで、4タイプのASIPコアの開発.
- Synfora (高位合成)
  - Cでの設計記述を解析し、頻度高く実行されるところはカスタムで設計.それ以外は、専用プロセッサコアへのマッピング.
  - 充実した実現ライブラリ



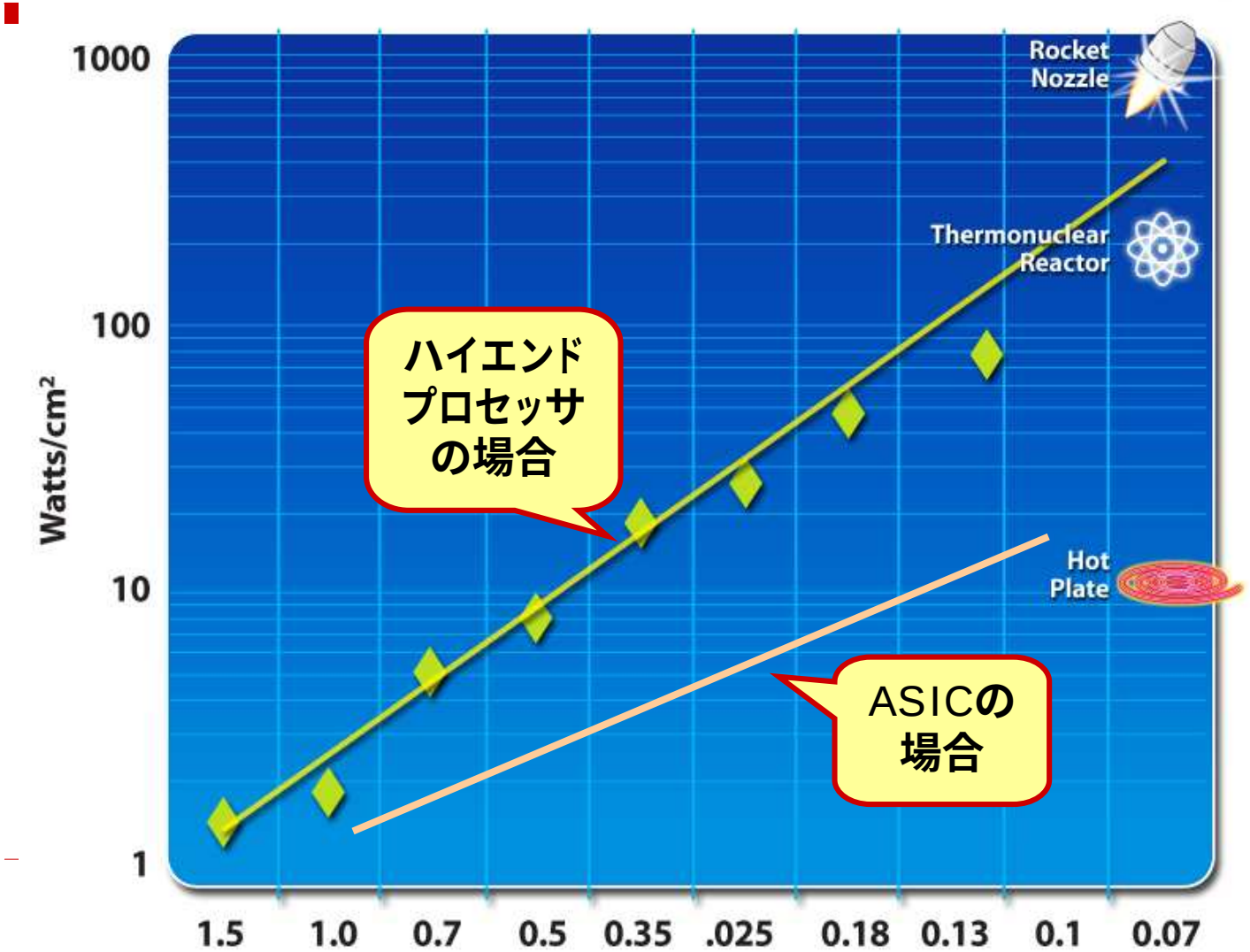
上流合成ツールの例 (2)

- YXI
  - Ansi-C からの合成 (Excite-Pro)
    - パイプライン合成
    - 複雑なラッチ挿入など
  - クロックアキュレイトな最適化(Excite-Expert)
- Forte
  - SystemCからの合成. 10社がすでに導入
  - 一般的なハイレベル合成アルゴリズム

システムレベル消費電力推定

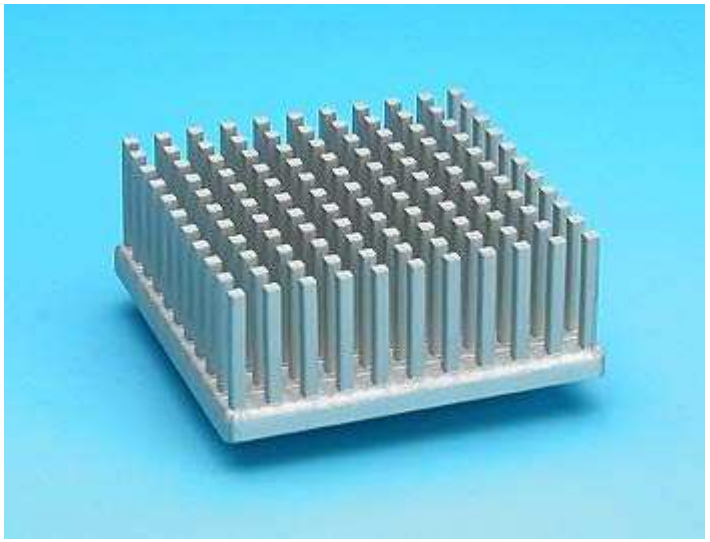


電力密度集中の問題

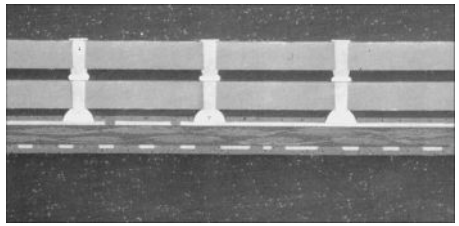


Source: "Circuit, Platform Design and Test Challenges in Technologies Beyond 90nm", Grundmanm, Galivanch, DATE 2003

空冷 (ヒートシンク)



LSIの内部に液体を流す冷却技術をIntel社が初披露（2004.04.23 日経マイクロ）



インテル社の低消費電力戦略

- インテルCTOの話
  - 我々はムーアの法則を守る！
  - 製造ばらつきが顕著に.
  - ゲートリークはHigh－Kで乗り切る.
  - トランジスタはFree(ただ). 面積を増やしても、製造ばらつき削減、電力削減の道を取る.
  - Vtコントロールアダプティブボディバイアスを使う.
  - タイミング設計はデタミニスティックからプロバブリスティックに変化.（2005にはすでに）

IBM Power PCでの低消費電力化

- 電力マネジメント(MVT)
  - 島に分け、MVTを使い、動いてない部分を動的に電源カットしていく.
- Vt, Vdd を細かく変化させる.
  - 基本セルを1000タイプ以上準備
  - Vddで島を形成.途中にレベルシフタ.
- 熱集中を避ける配置.

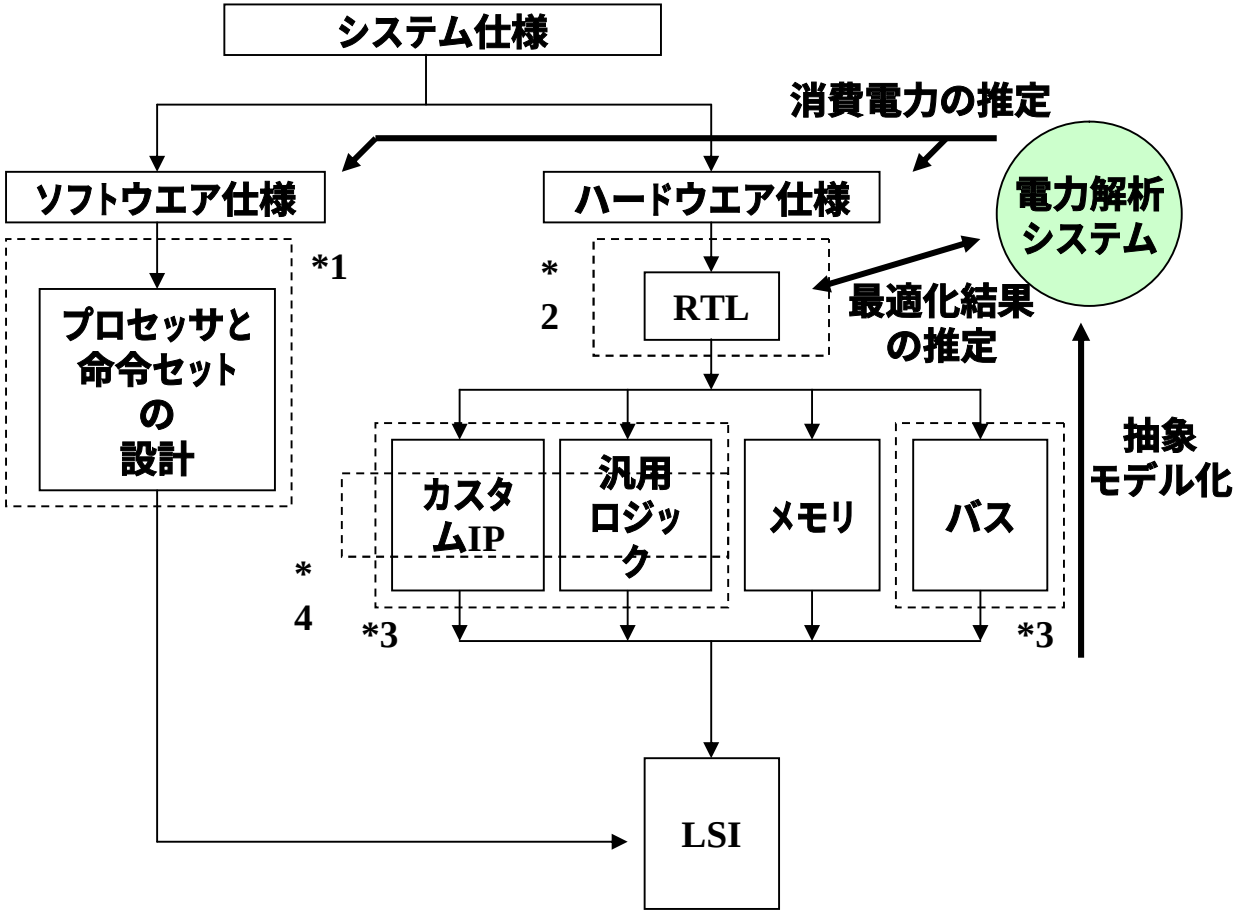
システムレベルでの低電力化で大事なこと

- 最適化のアプローチ
  - アーキテクチャレベルで
  - Vt,Vddをうまくコントロール、動的に制御.
  - ばらつきを無くす工夫も組み合わせる
- 電力解析のアプローチ
  - 精度が高いのはRTL以下
  - システムレベルでは電力削減効果が大
  - システムレベルに高精度モデルを提供する

消費電力解析ツールの例

- PowerEscape (CoWare) メモリアクセス空間や消費電力を解析.
- ORINOCO(ChipVision) 仮高位合成、フロアプランから消費電力を推定.
- Platune(UC Riverside) 各モジュールのパラメタライズされたキャッシュとバスの電力モデルを準備、それを用いて電力解析.

立命館の電力考慮システム設計PJ.



特徴

- 機能部品の詳細設計情報を高精度かつ高速計算可能な形でモデル化
  - 電源電圧、寄生容量、入力信号の特徴を捉え、各々の特徴毎に電力を計算
  - 各機能回路の特徴ごとに高位言語でモデル化.
- RTL設計の最適化結果を高精度に推定する機能を提供
- ソフトウエア設計に対して電力モデルを提供

予想成果

- 電力計算に要する時間を、論理回路シミュレーションを用いた場合に比較して、１０００倍程度高速化する.
- 電力推定の精度として実用に耐えるレベル (誤差10%以下、従来の1/10) の手段を提供する.
- 設計フロー上流のシステム仕様設計レベルでの高精度な電力最適化が可能となるので、システムLSIの大幅な電力削減 (約７０%程度) が図れる.
- 消費電力を考慮したソフトウエア設計を可能とする.

システム設計に関する今後の展望

- 不確実な製造時代への突入
  - ばらつき統計的アプローチ、自己調整回路
- 予想が難しい
  - 予想技術のてこ入れ
- 超短期開発
  - 合成系の充実
- 信頼性、自己修復
  - 新アーキテクチャ、リコンフィギャラブル

まとめ

- システムレベル設計は始まったところ
  - 設計効率の向上が必須
  - 高抽象レベルでの最適化が重要
- システムと半導体の接点ではC言語ベースの設計手法が主流
- 高位合成と推定技術はこれから実用化が進む