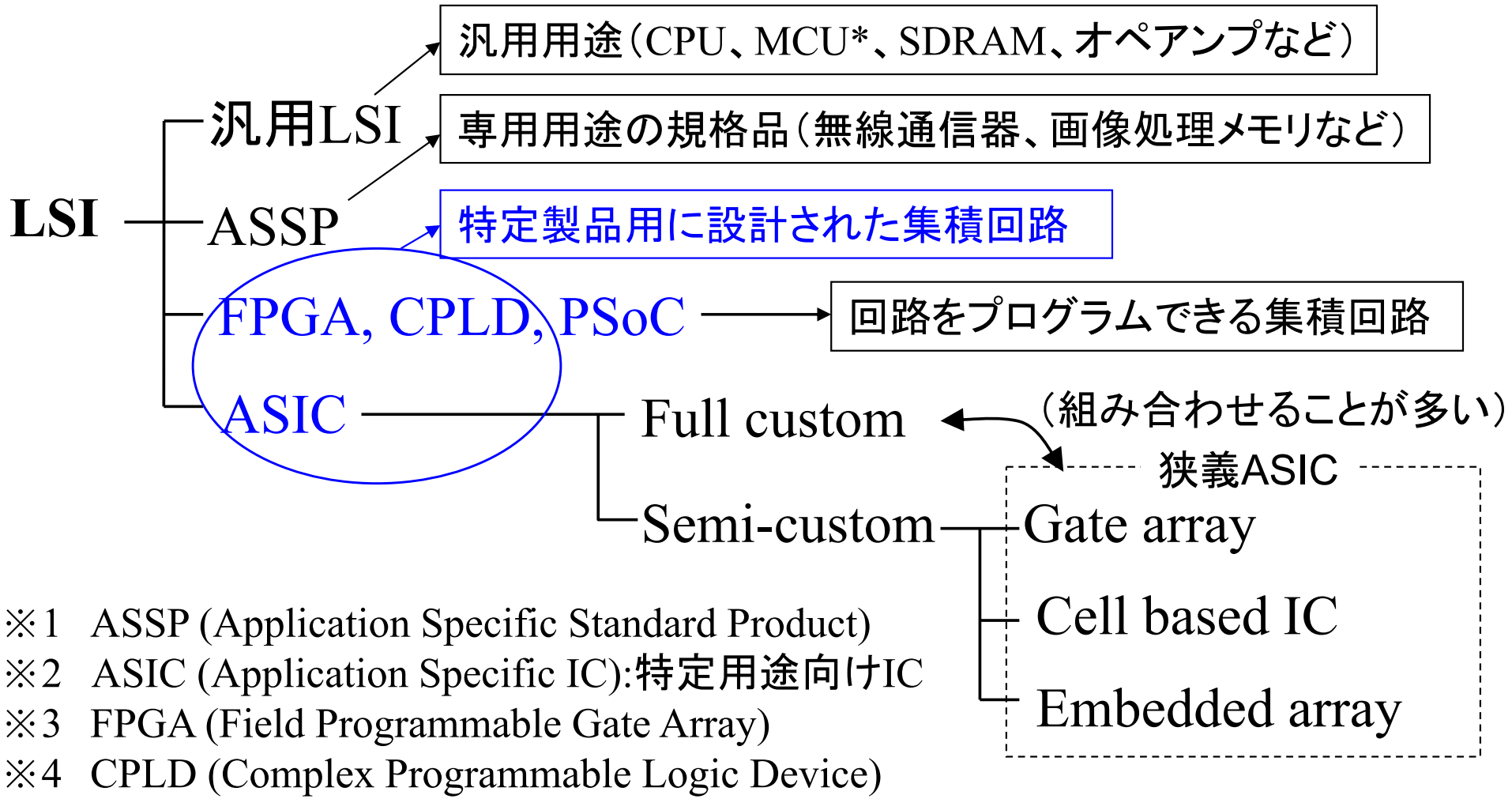
7.1 Design flow

ボトムアップ方式とトップダウン方式

７．１．１　回路実装方式の分類

LSIの実装方式による分類



※1　ASSP (Application Specific Standard Product)
※2　ASIC (Application Specific IC):特定用途向けIC
※3　FPGA (Field Programmable Gate Array)
※4　CPLD (Complex Programmable Logic Device)

＊MCUには不揮発性メモリが集積化されており、プログラム（ファームウエア）を書き込むことができる。
近年はハードウエアの構成自体もプログラムできるMCU(PSoC等)が製品化されており応用分野が広い。

回路実装方式の比較

実装方式	主な利用形態	長所	短所
汎用LSI/ ASSP	1．複数の市販LSIを組み合わせてシステムを構成	1．高信頼性 2．複数品種から選択できる 3．開発期間が短い	1．設計自由度が低い 2．ASICよりは性能が低い
MCU（汎用LSIの一種）	1．システム全体または機能ブロックをソフトウエアとして実装	1．プログラムが可能 2．高信頼性 3．開発費が少ない 4．（優秀なプログラマがいれば）開発期間が短い	1．ASICよりは性能が低い 2．設計自由度が低い(複数品種から選択)
FPGA/ CPLD	1．ディジタルシステムの1チップ化 2．プロセッサ周辺回路の1チップ化	1．回路の書き換えが可能 2．大規模集積化が可能 3．ASICより短期間で開発	1．ASICよりは性能が低いが高性能 2．量産により価格が下がらない
広義ASIC	1．システムの1チップ化 2．市販されていない機能ブロックの1チップ化	1．高性能 2．設計自由度が高い 3．大規模集積化が可能 4．量産により低価格化する	1．初期開発コストが高い

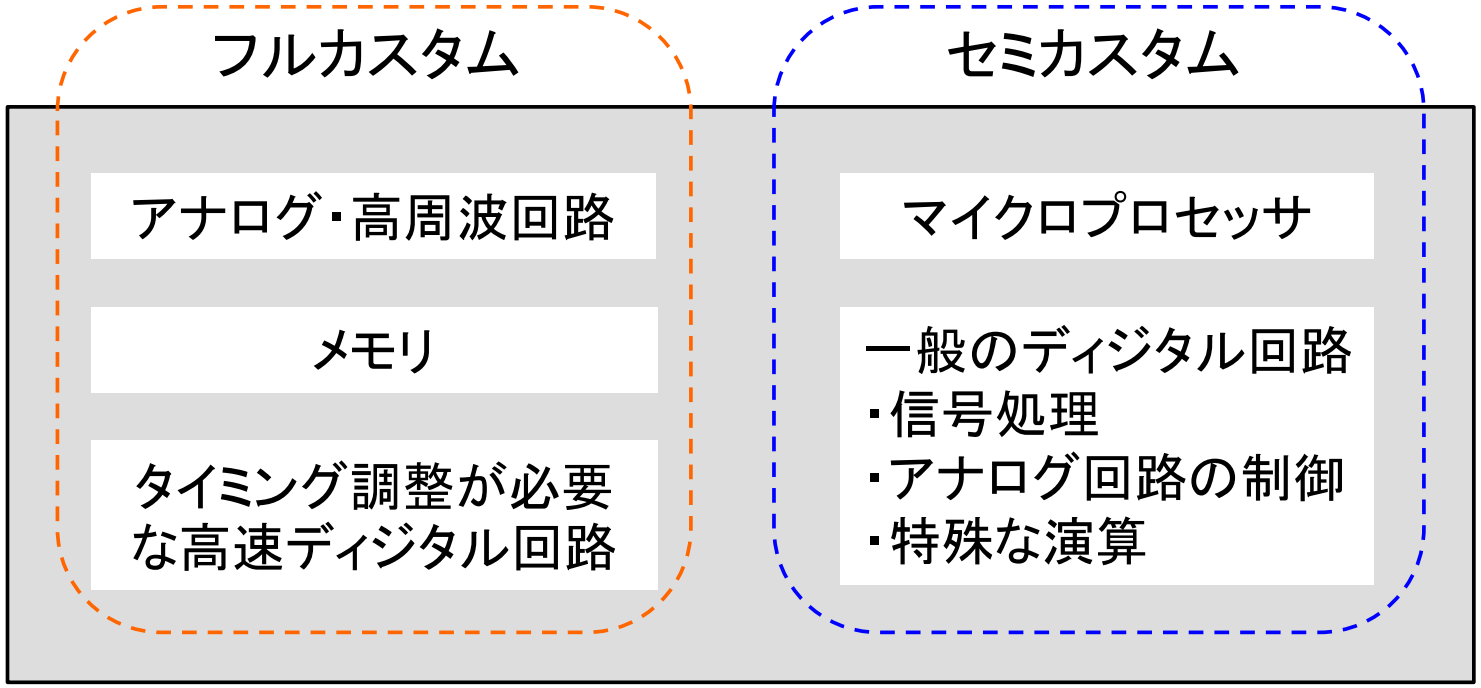
実際には、これらの方式を組み合わせてシステムを構築することが多い。

広義のASICの設計方式

- フルカスタム方式による設計
  - 細部から全体を構築する＝ボトムアップ方式の設計フロー
  - 短所：回路とレイアウトの設計を手動で行うため大規模な設計は困難
  - 長所：トランジスタレベルの最適化により高性能化・小型化が可能
  - 短所：製造メーカやテクノロジを変更すると設計のやり直しが必要
- セミカスタム方式による設計
  - 全体から細部に向かって作り込んでいく＝トップダウン方式の設計フロー
  - 長所：CAD(Computer Aided Design)ツールを用いた自動設計
    - 高位合成、論理合成、自動配置配線技術を使用
  - 短所：詳細に最適化された回路やアナログ回路は設計できない
  - 長所：一度設計すれば製造メーカやテクノロジを変更することが可能

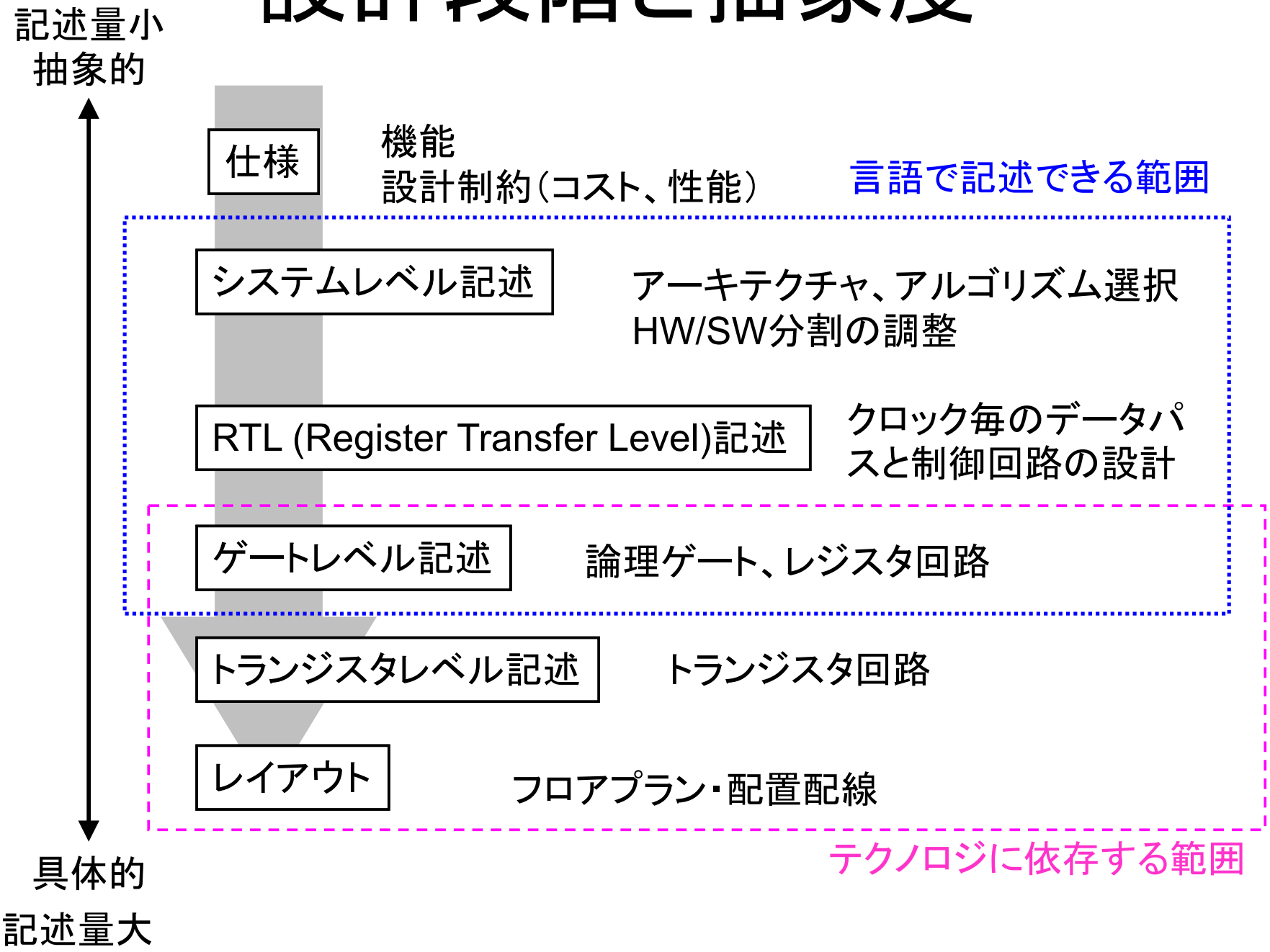
設計方式の併用

- 実際にはフルカスタム設計とセミカスタム設計を併用してLSIが開発される
  - フルカスタム設計済みの回路ブロックがIPとして提供される場合は、セミカスタム設計フローの中で利用できる（例：SRAMはメーカから提供されることが多い）



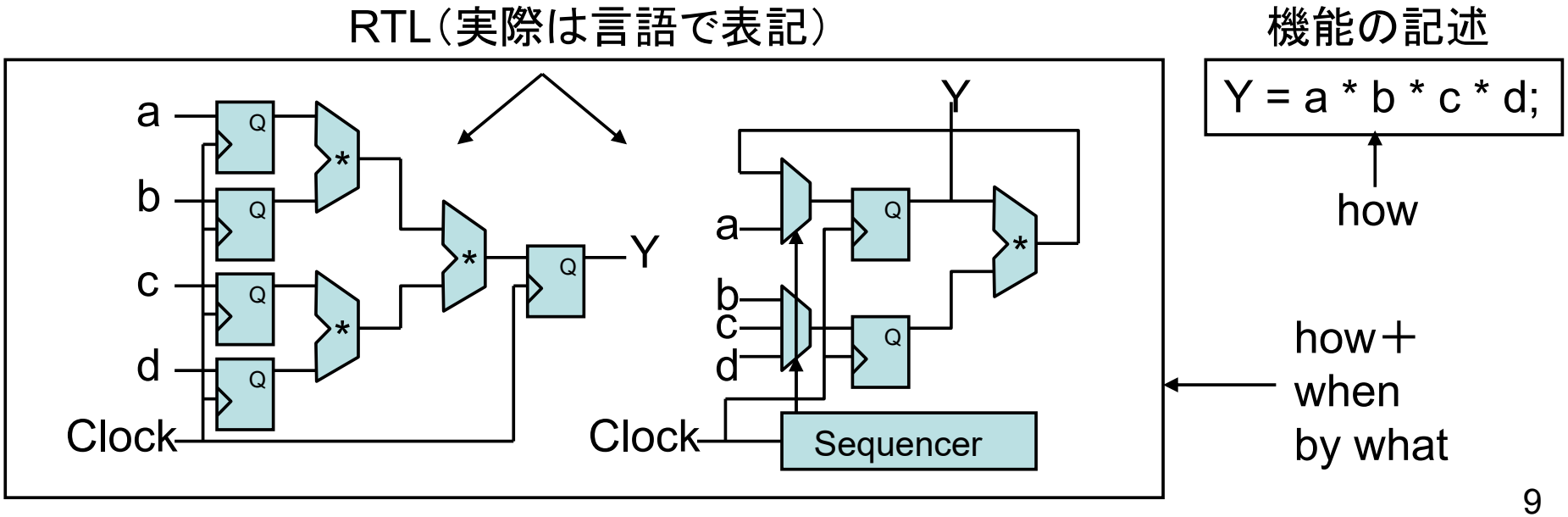
７．１．２　設計の抽象度

設計段階と抽象度


記述量小
抽象的
仕様
機能
設計制約(コスト、性能)
言語で記述できる範囲
システムレベル記述
アーキテクチャ、アルゴリズム選択
HW/SW分割の調整
RTL (Register Transfer Level)記述
クロック毎のデータパスと制御回路の設計
ゲートレベル記述
論理ゲート、レジスタ回路
トランジスタレベル記述
トランジスタ回路
レイアウト
フロアプラン・配置配線
テクノロジに依存する範囲
具体的
記述量大

システムレベルとRTL

- システムレベル（S/W, H/W共通の抽象度）
  - 機能や動作（処理・通信の手順）だけを記述（クロック数やタイミングを気にしない）
  - ハードウエアとソフトウエアの役割分担は意識する必要がない（ただし、この記述レベルでは論理合成できない場合が多い）
- RTL (Register Transfer Level)（H/Wの抽象度）
  - クロック毎の処理内容とレジスタへの代入を記述
  - how と when（どのクロックエッジ？）と by what（どの機能をどう使って？）
  - ハードウエア内の構造をある程度意識（確実に合成可能とするため）



抽象度による記述量の違い

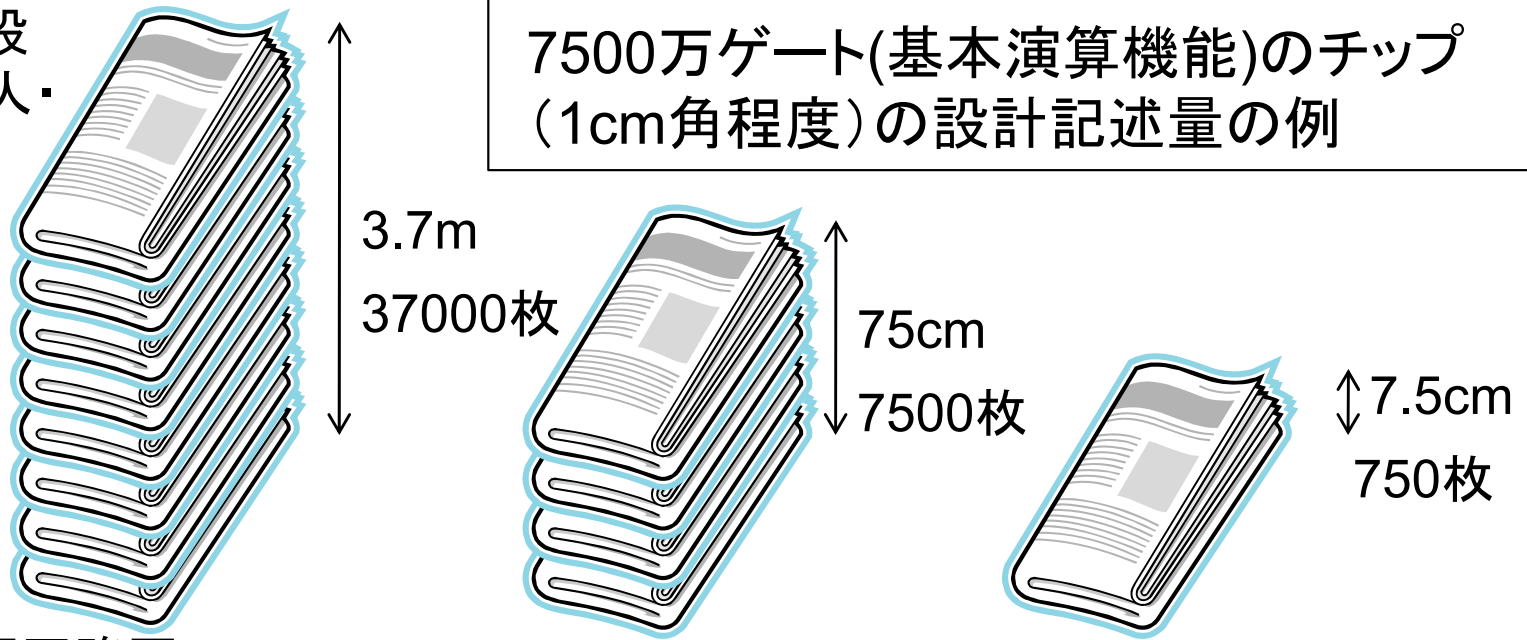
設計記述の抽象度を高めてコンピュータで自動変換すると生産性が増大

抽象度の高い記述から低い記述を自動生成する技術がLSI-CAD技術

論理回路図では
非常に優秀な設計者でも2800人・月が必要
（実質不可能）

記述量の比較

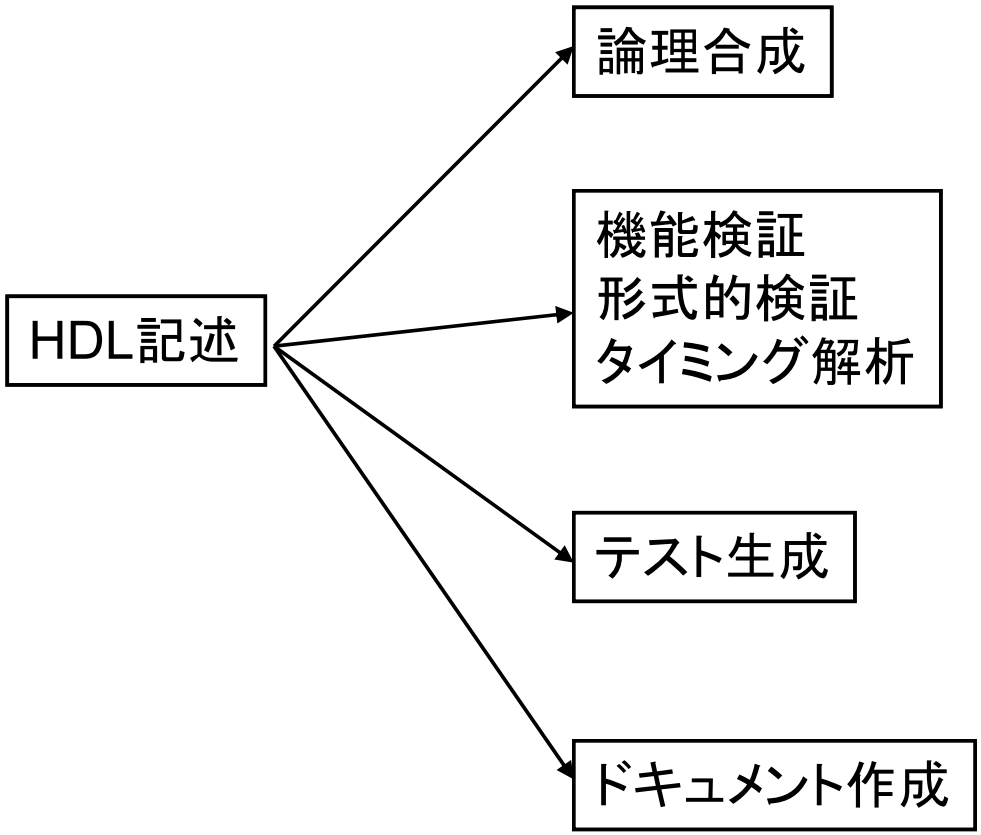
7500万ゲート(基本演算機能)のチップ
（1cm角程度）の設計記述量の例



主な記述言語

記述内容	言語（標準化）	特　長
ハードウエア記述言語（HDL:　Hardware Description Language） ハードウエアの論理機能シミュレーションと論理合成のための言語	Verilog-HDL (IEEE-1364)	C言語に似ていて記述が簡潔（LSI開発者にユーザが多い）
	VHDL: Very High Speed Integration Circuit HDL (IEEE-1164)	記述量が多いが文法が厳密でバグを見つけやすい（ユーザが最も多い）
システムレベル記述言語(System Level Description Language) ソフトウエアを含むシステムのモデリングと動作合成のための言語	SystemC	C/C++のクラスとして提供されるのでシミュレーションに特別な開発環境が不要
	UML	システムのモデリングに使用される図式言語

HDLの役割



論理機能の記述から回路を合成

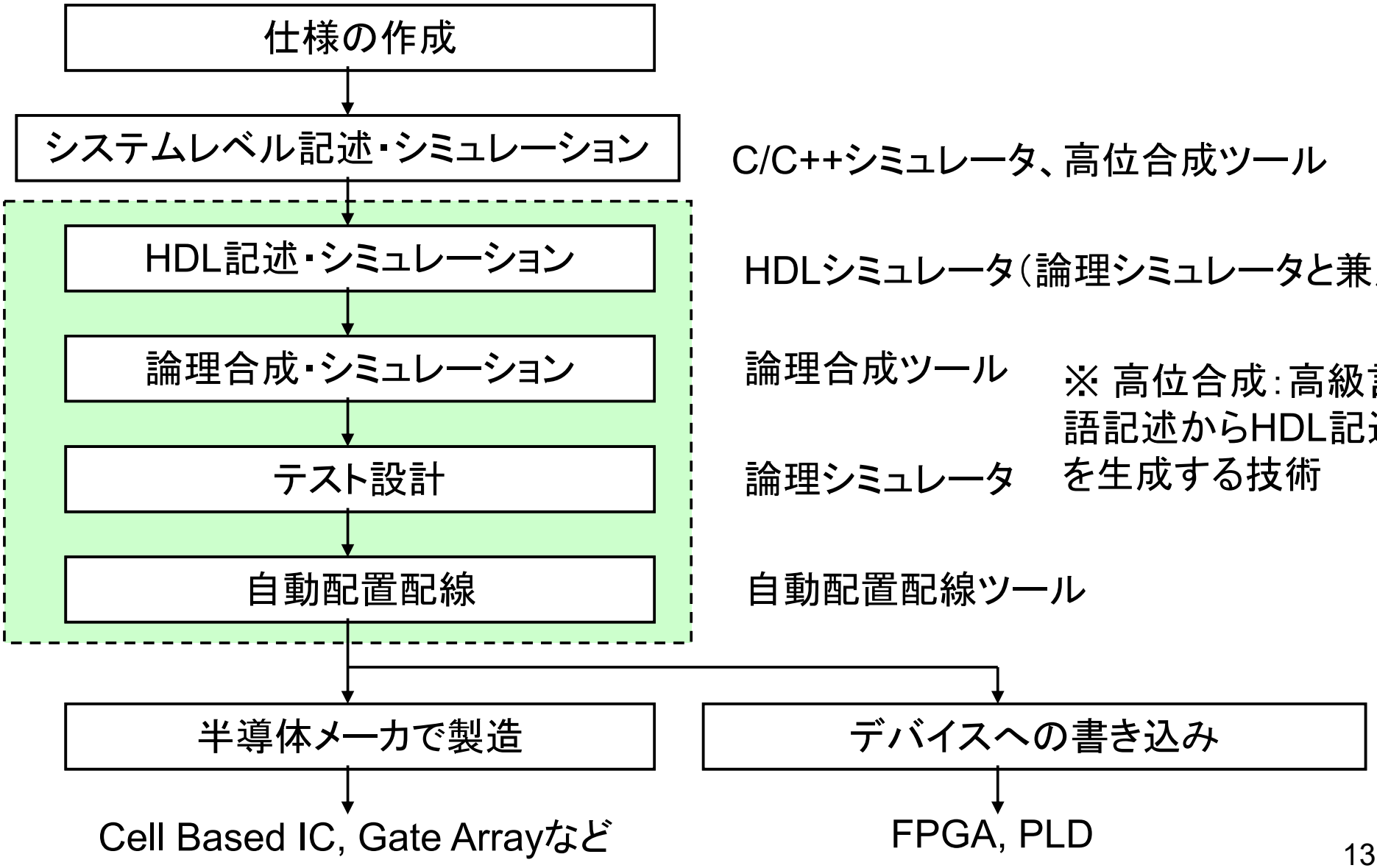
システム検証
論理検証
（設計のバグとり）

テスト設計
（不良品の選別用テスト用
回路とテスト用データの作成）

設計の資産化（IPの説明書）

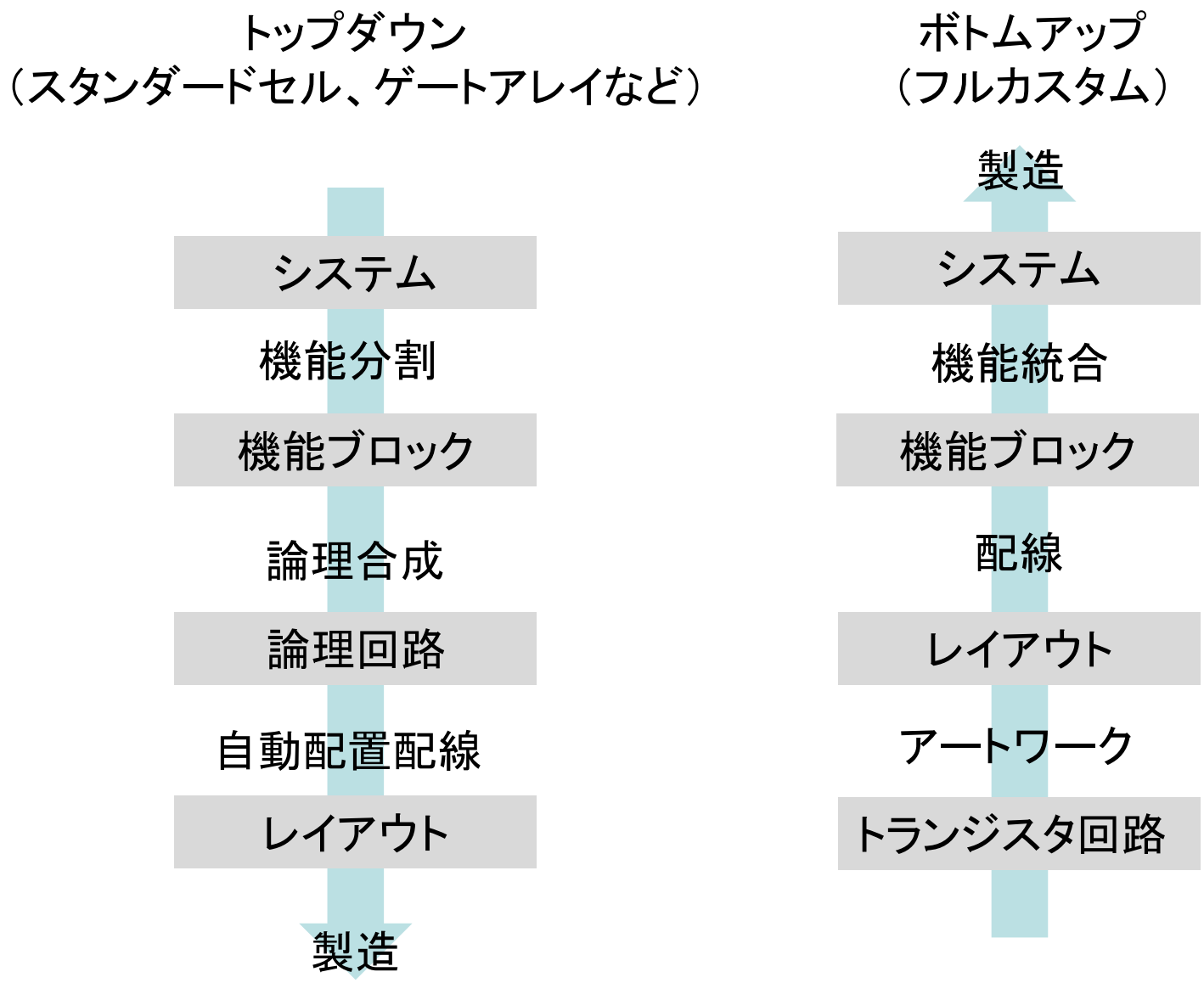
一度HDLで設計しておけば、色々な場面で使える

HDLを用いた設計フローの概要



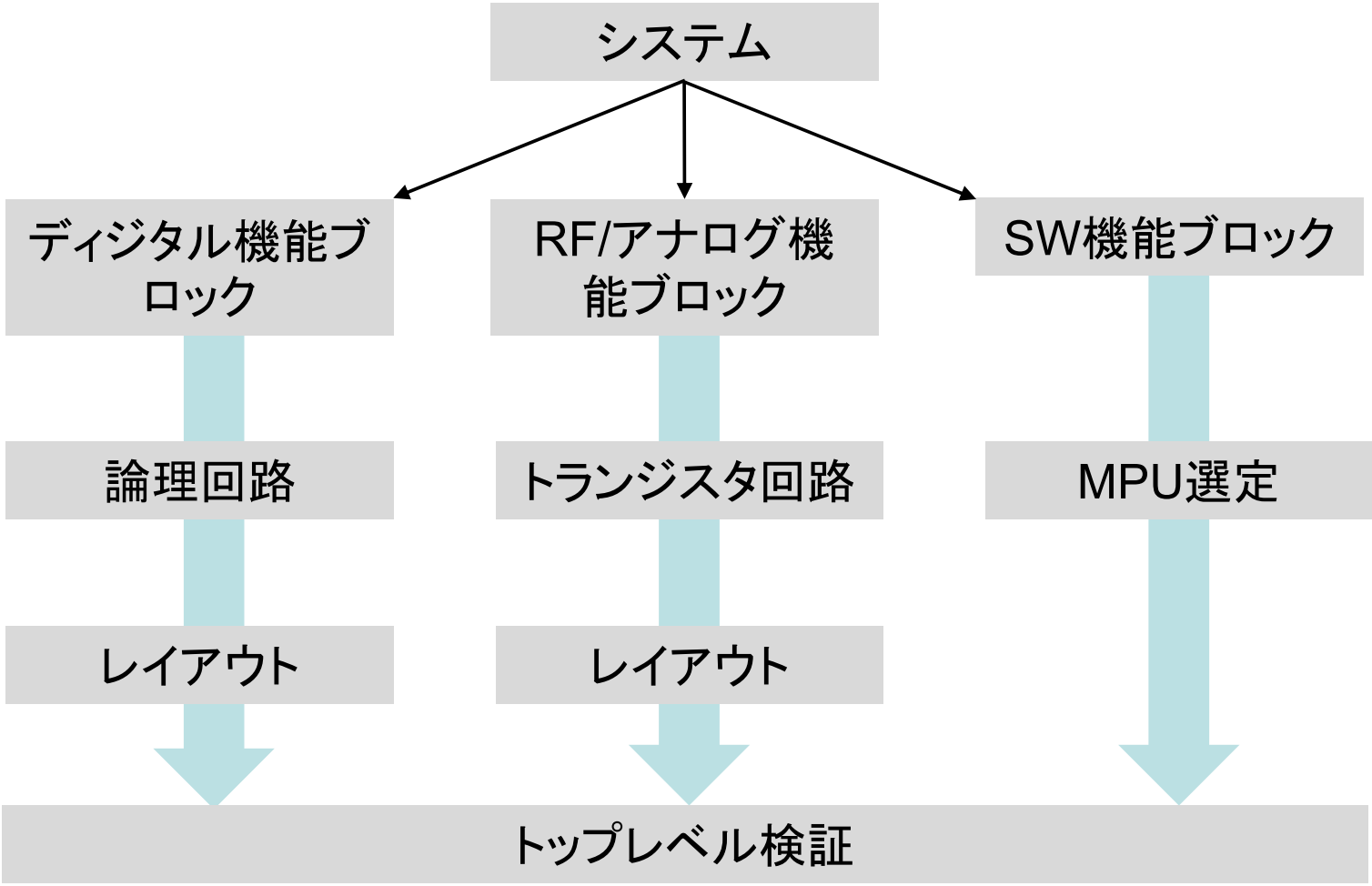
７．１．３　ボトムアップとトップダウン

ボトムアップとトップダウン設計フロー



実際の設計フロー

実際にはトップダウンとボトムアップを併用



ボトムアップとトップダウンの比較

方式	長所	短所
ボトムアップ方式	• 回路・レイアウト上での高性能化が追求できる • 回路面積が小さい • アナログや高周波に対応可能 • 規則性の高い回路が得意	• 見通しが悪い（バグを作りやすい） • 設計が大変 • ランダム構造の回路は苦手
トップダウン方式	• 見通しがよい（バグが発生しにくい） • 設計は比較的楽 • ランダム構造の回路が得意	• 回路・レイアウト上での高性能化はできない • 回路面積が大きい • アナログや高周波への対応は困難 • 規則性の高い回路は苦手

フルカスタム設計CADツール

CADツールの種類	役　割
回路シミュレータ	トランジスタレベルの回路シミュレーションを行う。アナログ回路の設計や自動設計のためのライブラリ構築には必須。LSIは試行錯誤で作れないため、回路シミュレータが最後の命綱となる。
回路図エディタ	回路シミュレーションやLVS（後述）に利用するため、回路図を入力するお絵かきツール。
レイアウトエディタ	チップ上のトランジスタや配線パターンを手入力で描くお絵かきツール。レイアウトデータの半導体業界標準フォーマットであるGDS-IIファイルを出力する機能を持つ。
設計規則チェッカ (DRC, LVS, LPE)	レイアウトが設計規則に違反していないことをチェック（DRC）する。回路とレイアウトの等価性照合（LVS）、配線抵抗や容量などの寄生素子の算出（LPE）などの機能も持つ。

DRC:　Design Rule Check
LVS:　Layout vs Schematic
LPE:　Layout Parasitic Extraction

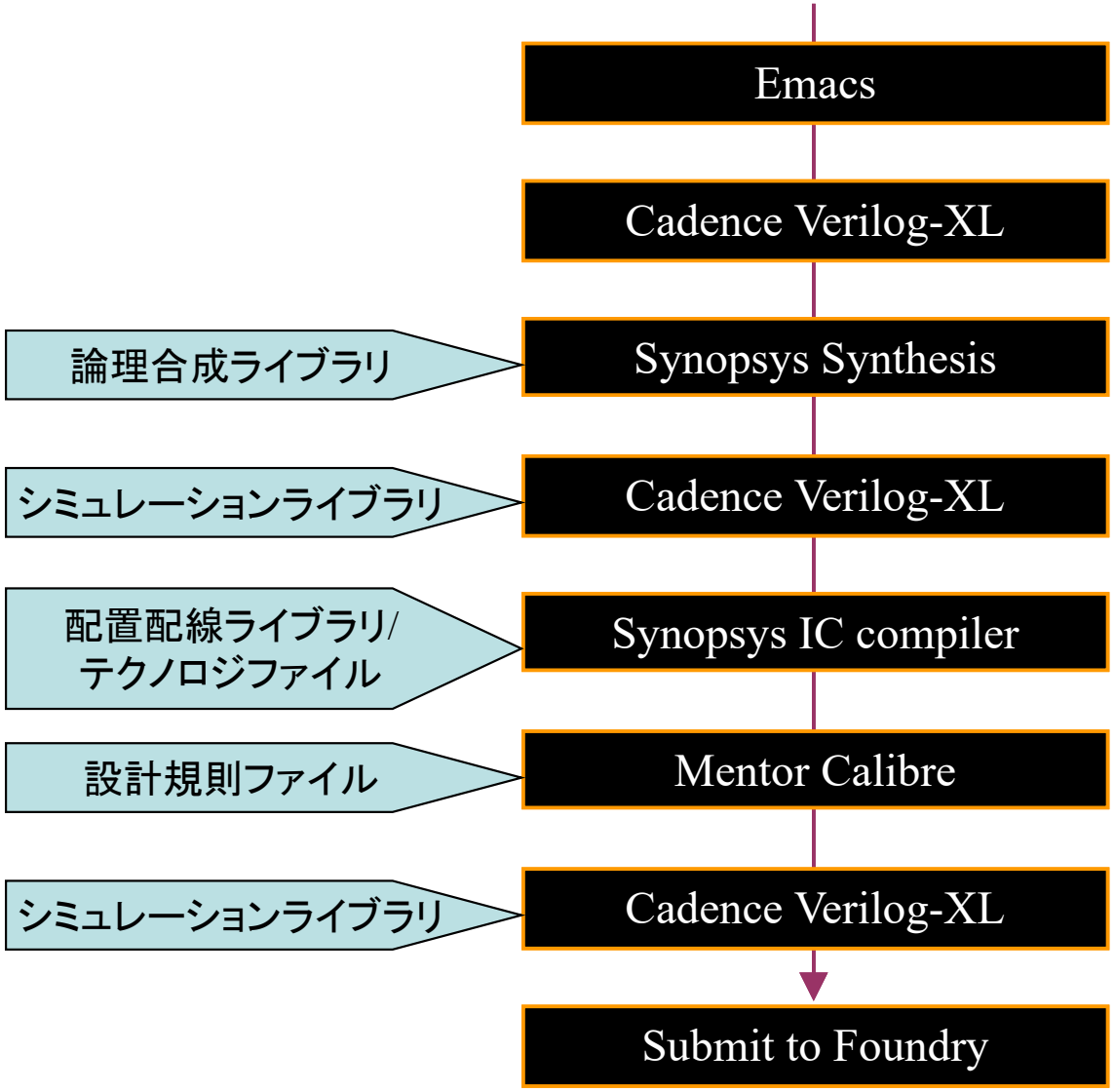
VDECを利用したフルカスタム設計フローの例

セミカスタム設計CADツール

CADツールの種類	役　割
高位合成ツール	システムレベル記述（SystemCなど）からHDL記述を合成する。SW用のコンパイラと異なり、演算器数、クロック周波数などの最適化オプションがある。
C++シミュレータ	システムレベル記述でモデル化されたHW＋SWの動作をシミュレーションするために使用する。
論理合成ツール	HDLで記述から論理回路（ゲートの接続情報）を合成する。
HDLシミュレータ（論理シミュレータ）	HDL記述されたHWの動作をシミュレーションする。HDLは、HWの処理内容だけでなく、ゲートレベルの論理回路も表現できるので、論理シミュレータとしても使用される。
自動配置配線ツール(P&R)	論理回路から集積回路チップ上のトランジスタ配置および配線情報（レイアウト）を作り出す。FPGAでは、機能ブロックの機能と配線情報（Configuration data）を作り出す。
設計規則チェッカ	レイアウトが設計規則に抵触していないことをチェックする。

P&R: Place and Route

VDECを利用したセミカスタム設計フローの例



HDL記述（RTL:Register Transfer Level）

HDLシミュレーション（機能検証）

論理合成

論理シミュレーション（タイミング検証）

フロアプラン・自動配置配線
簡易DRC/LVS

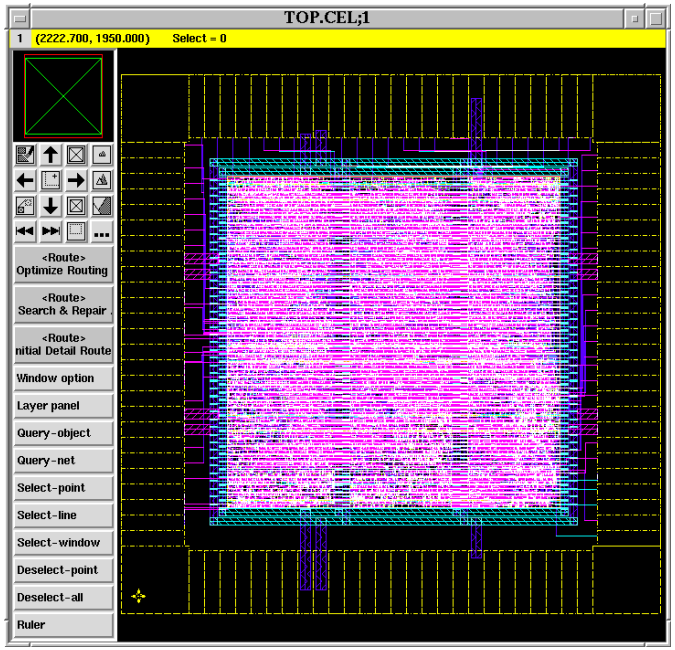
設計規則チェック/LVS
回路パラメータ抽出

実配線シミュレーション

メーカ

設計自動化の条件

- ディジタル回路であること
  - アナログ回路、特に高周波の自動設計は現状では困難
- テクノロジ情報と設計ライブラリが入手できること
  - ライブラリは、有料で提供されるが、自分で作ることも可能



自動配置配線ツールによるレイアウトデータ作成例。自動配置配線でも、各種の設計規則エラーが発生することが多い。これらのエラーの意味を理解するためにはテクノロジやレイアウトの知識が必要なので、フルカスタムの設計経験は必要。

設計自動化に必要なデータ

- 論理シミュレーションモデル
  - 論理ゲートの遅延時間、駆動力などの情報
- 論理合成ライブラリ
  - 論理ゲートの面積、ファンアウト数、遅延時間など
- 自動配置配線ライブラリ
  - 論理ゲートの形状、端子位置など

設計ライブラリと呼ばれる

テクノロジ・ファイルと呼ばれる

- 設計規則
  - レイアウトパターンに関する規則、レイアウト→回路抽出情報など
- トランジスタモデル
  - 回路シミュレーション用デバイスモデルのパラメータ・セット（トランジスタレベルの回路設計をしないなら必要ない）
- レイヤー情報
  - レイアウトデータに含まれるレイヤーの種類やレイヤー番号など