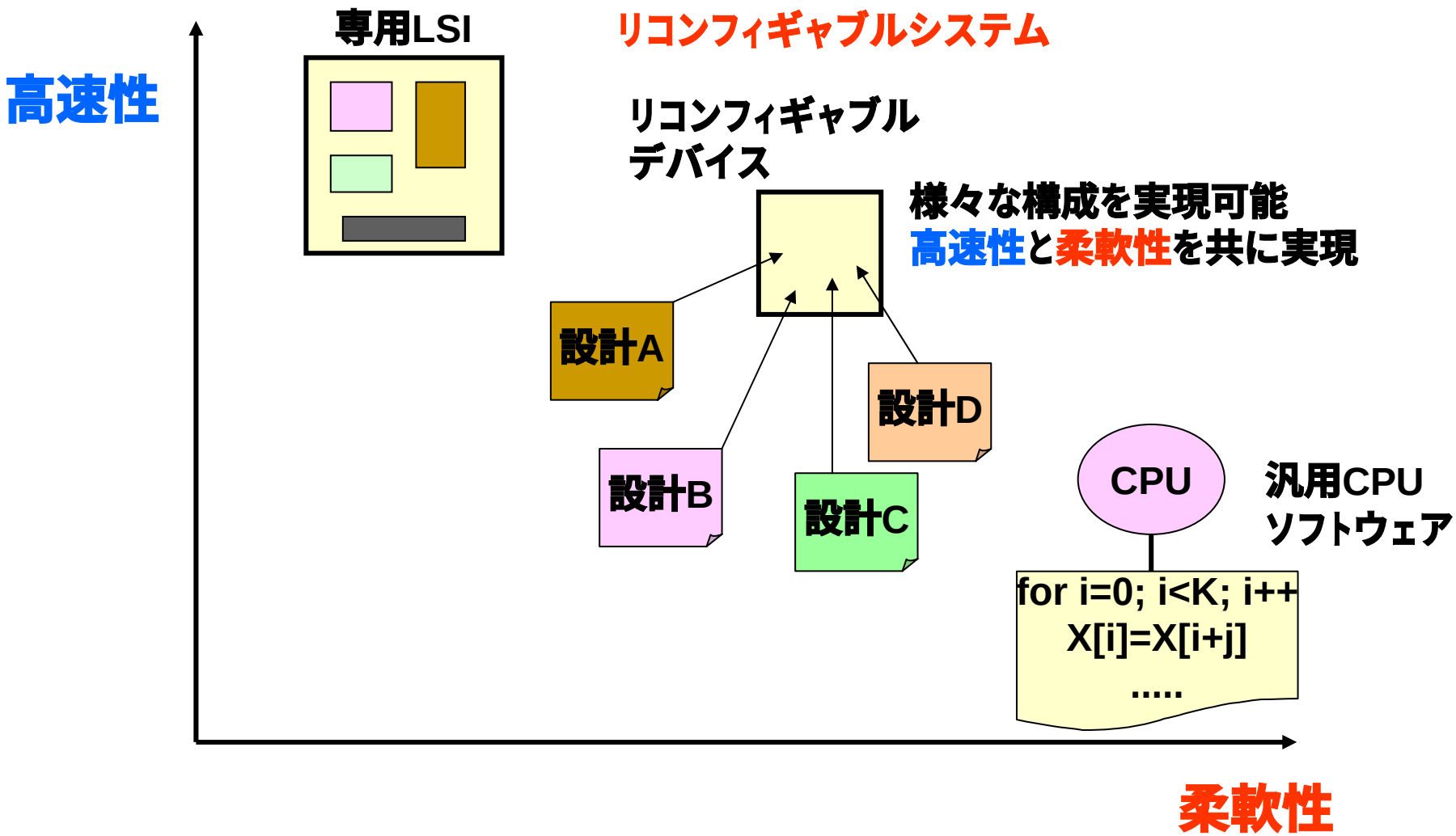
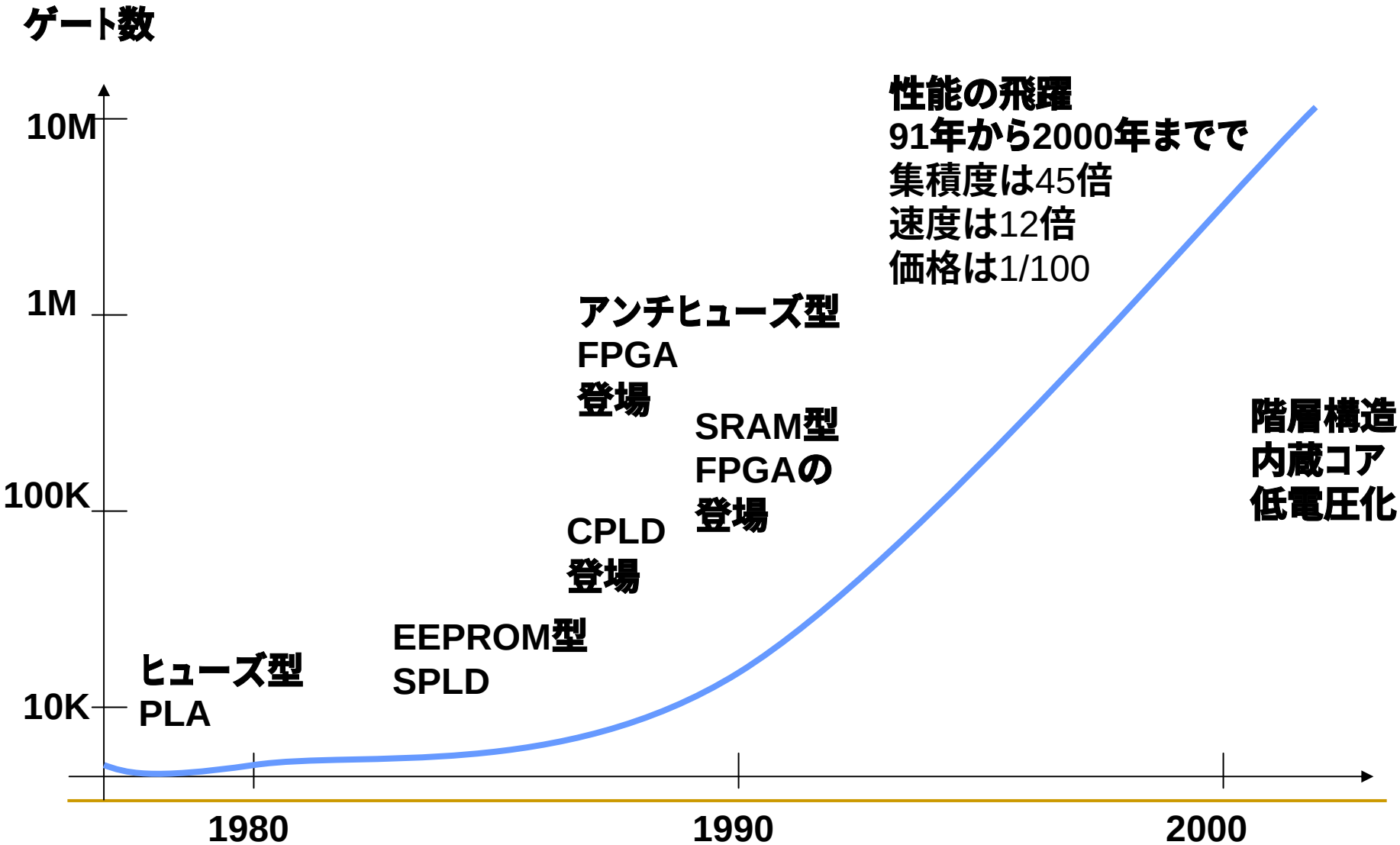
リコンフィギャラブルシステム最新情報

慶應義塾大学理工学部
天野英晴

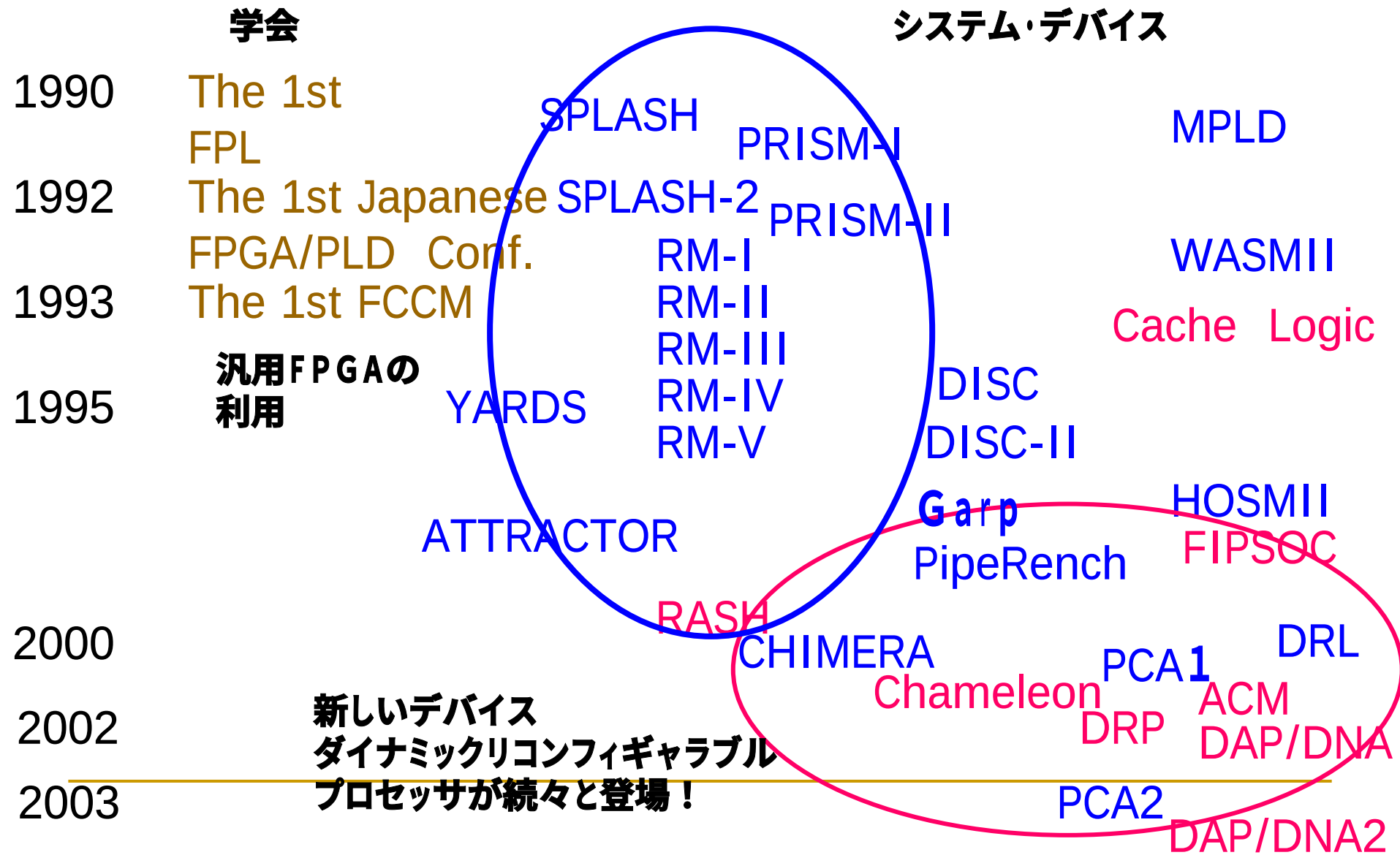
リコンフィギャラブルシステムとは？



PLDの成長


ゲート数
10M
1M
100K
10K
1980
1990
2000
ヒューズ型
PLA
EEPROM型
SPLD
CPLD
登場
アンチヒューズ型
FPGA
登場
SRAM型
FPGAの
登場
性能の飛躍
91年から2000年までで
集積度は45倍
速度は12倍
価格は1/100
階層構造
内蔵コア
低電圧化

Reconfigurable Systemsの発展



リコンフィギャラブルシステムとダイナミックリコンフィギャラブルシステム

- リコンフィギャラブルシステム
  - 柔軟に設計変更可能
  - 汎用FPGAの利用
  - ニッチ市場である程度地位を確立
    - プロトタイピング、基地局、バイオインフォマティクス他用のアクセラレータ
  - CPU内蔵FPGAの登場により、協調設計システム、システムとしての開発環境の研究が進む
- ダイナミックリコンフィギャラブルシステム
  - 動作中に構造を変更することにより、主として面積効率の向上を目指す
  - ニッチ市場ではなく、SoC一般を広くターゲットとする
  - 汎用FPGAは構成の変更時間が長すぎる→専用のデバイスを利用

従来型リコンフィギャラブルシステムの最近の話題

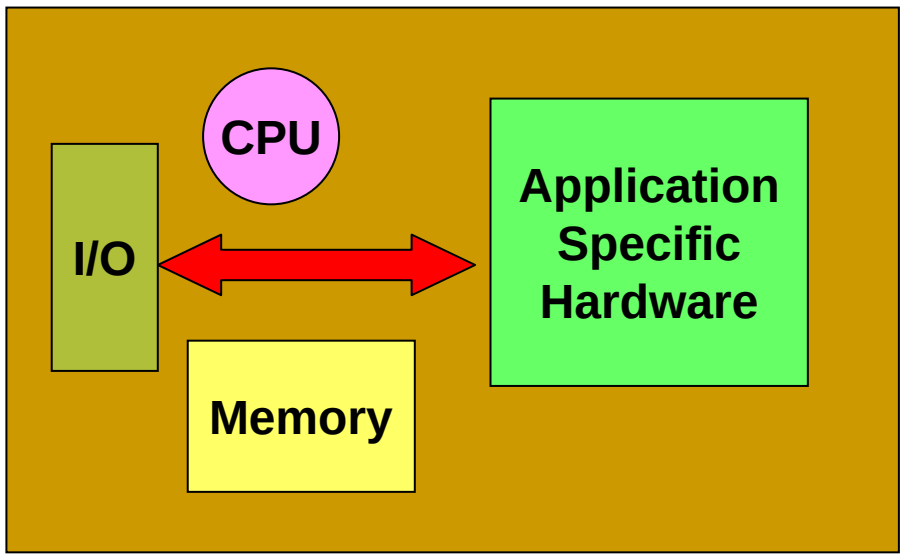
- 大規模SoPD (System on Programmable Device)の登場と普及
  - ネットワークインタフェース、バイオインフォマティクス等の応用分野への本格利用
  - CPU/Reconfigurable Systemのシステム設計、協調設計
  - 組み込みOSとの連携、遠隔再構成、動的再構成
- FPG
  - 組
- 科学技術計
  - 内蔵演算器と分散メモリの利用により、本格的に科学技術演算への応用を目指す
  - 可変長の浮動小数演算
  - 今年のFCCMは浮動小数点系ばかり

リコンフィギャラブルシステム研究会などで、その成果が活発に議論されています。今年は9月15, 16、17日に長崎大であります。ぜひご参加ください。

紹介の流れ

- ダイナミックリコンフィギャラブルプロセッサ導入の動機
  - 単なるFPGA/CPLDの発展形ではない
  - SoCの問題点
  - 組み込みCPU＋FPGA/PLDの問題点
- 3つのポイント
- ダイナミックリコンフィギャラブルプロセッサ紹介
- DRPの世界
- おわりに

ＳｏＣの問題点



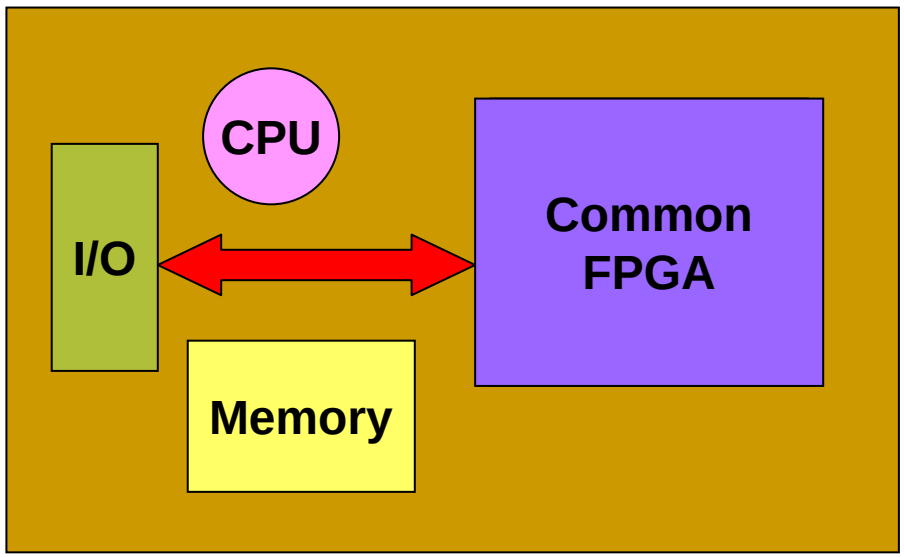
SoC (System-on-a-Chip)
CPU,Memory,I/O,専用Hwを混載したLSI
多様なアプリケーションに利用可能
Cellular Phones,
Network Controllers,
Mobile Terminals…

Problem!

- 性能は専用Hwに寄るところが大
- 新しい技術が次々に登場
  (JPEG2000, AES,Turbo code…..)
- 新しいテクノロジではバックエンドの負荷が増大
- デザインコストの増大!
- ＳｏＣを作っても儲けにつながらない

性能と柔軟性を兼ね備えた手法の必要性!

ＣＰＵ+汎用ＦＰＧＡではどうか?

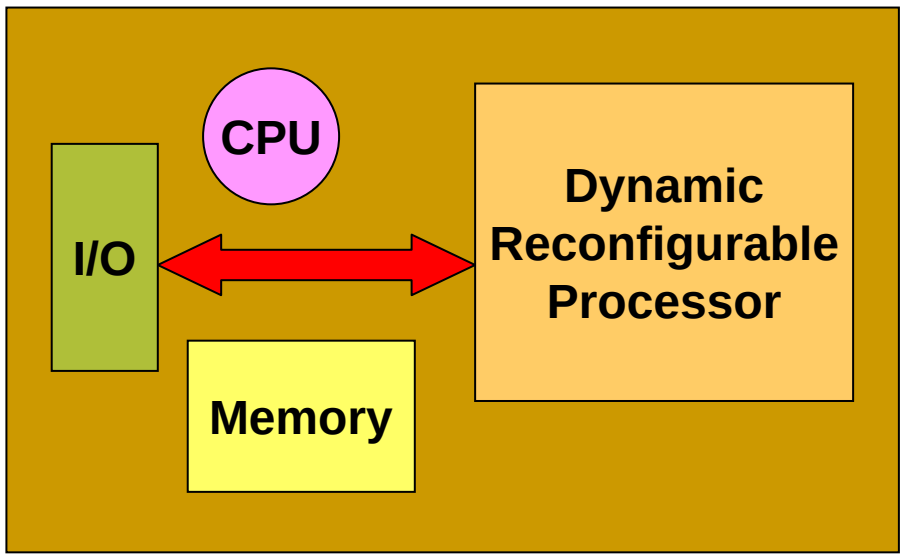


汎用 FPGA は柔軟
Xilinx社　Virtex II Pro　(PowerPC)
Altera社　Excalibur (ARM)　は
既に市場に出回っている

しかし

- **ＬＵＴベース→　面積が大きい**
- **性能も専用ハードウェアに比べて低い**
- **高価！**
- **新規参入が困難**
  - **ＣＡＤの問題**
  - **特許の問題**

CPU+ダイナミックリコンフィギャラブルプロセッサのアプローチ



粗粒度構成
(Course Grain Structure)
→　高速で面積も小さい

ダイナミック
リコンフィギュレーション
→　高い面積効率

C-level programming
→　容易に高い性能が得られる

紹介の流れ

- ダイナミックリコンフィギャラブルプロセッサ導入の動機
- 3つのポイント
  - 粗粒度構成
  - ダイナミックリコンフィギュレーション
  - C-level設計
- ダイナミックリコンフィギャラブルプロセッサ紹介
- DRPの世界
- おわりに

ポイント1：粗粒度構成の採用

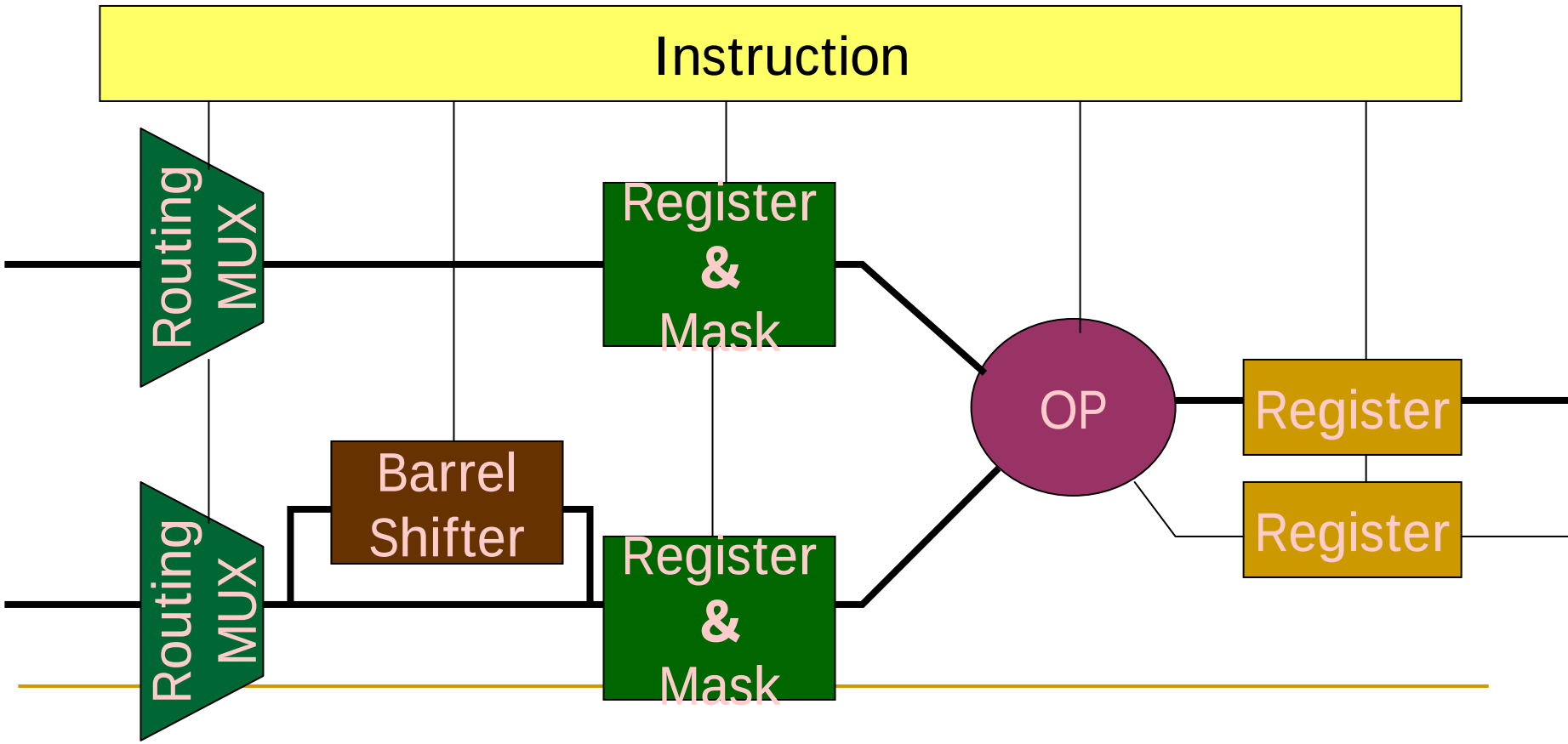
- ＬＵＴやＡＮＤ－ＯＲから演算器を中心とする粗粒度構成の採用
- 構成要素
  - 演算器
  - Ｓｈｉｆｔｅｒ＋マスク
  - マルチプレクサ
  - レジスタ
- 機能、要素間の配線、動作方式を再構成可能にする

Chameleon CS2112のDPU

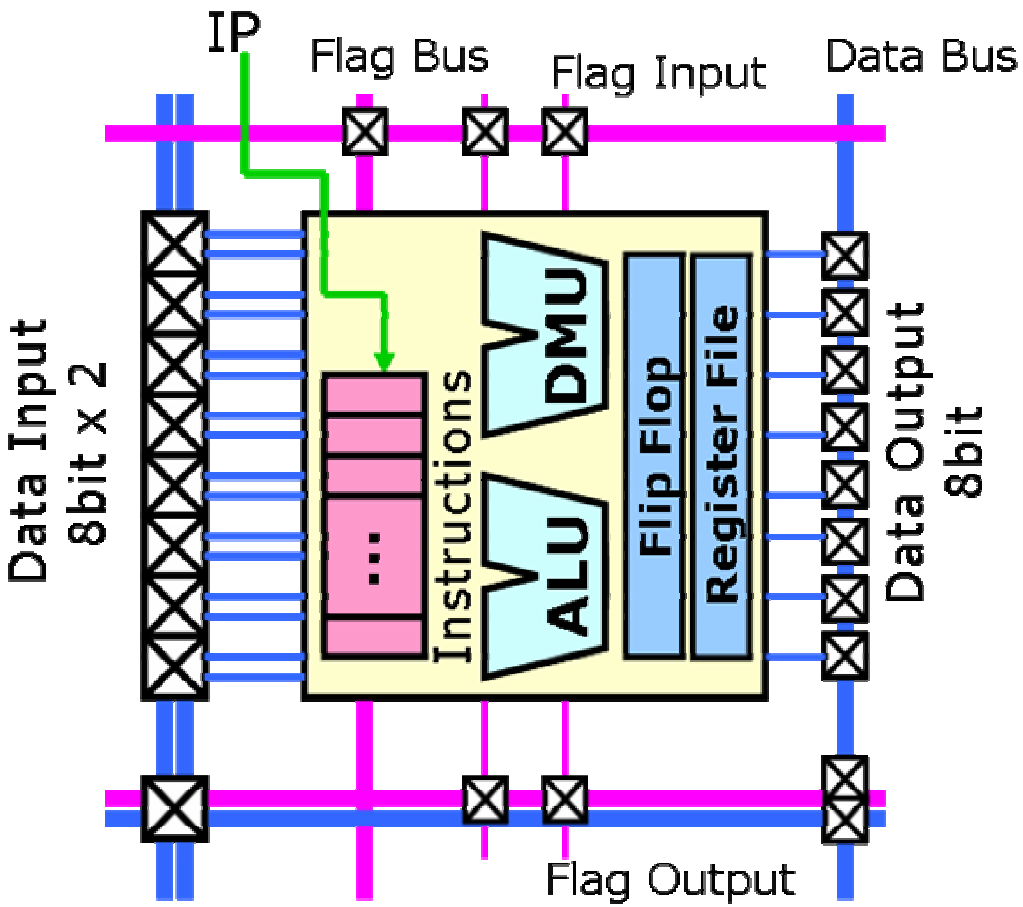
OP：Operations in C or Verilog

SIMD arrays and pipelines are formed with multiple DPUs.

32bit・16bit
構成



NEC社DRP-1の PE (Processing Element構造)



ポイント2：ダイナミックリコンフィギュレーション

- 粗粒度構成により、細粒度構成に比べて演算処理の性能は向上し、面積も小さくなる

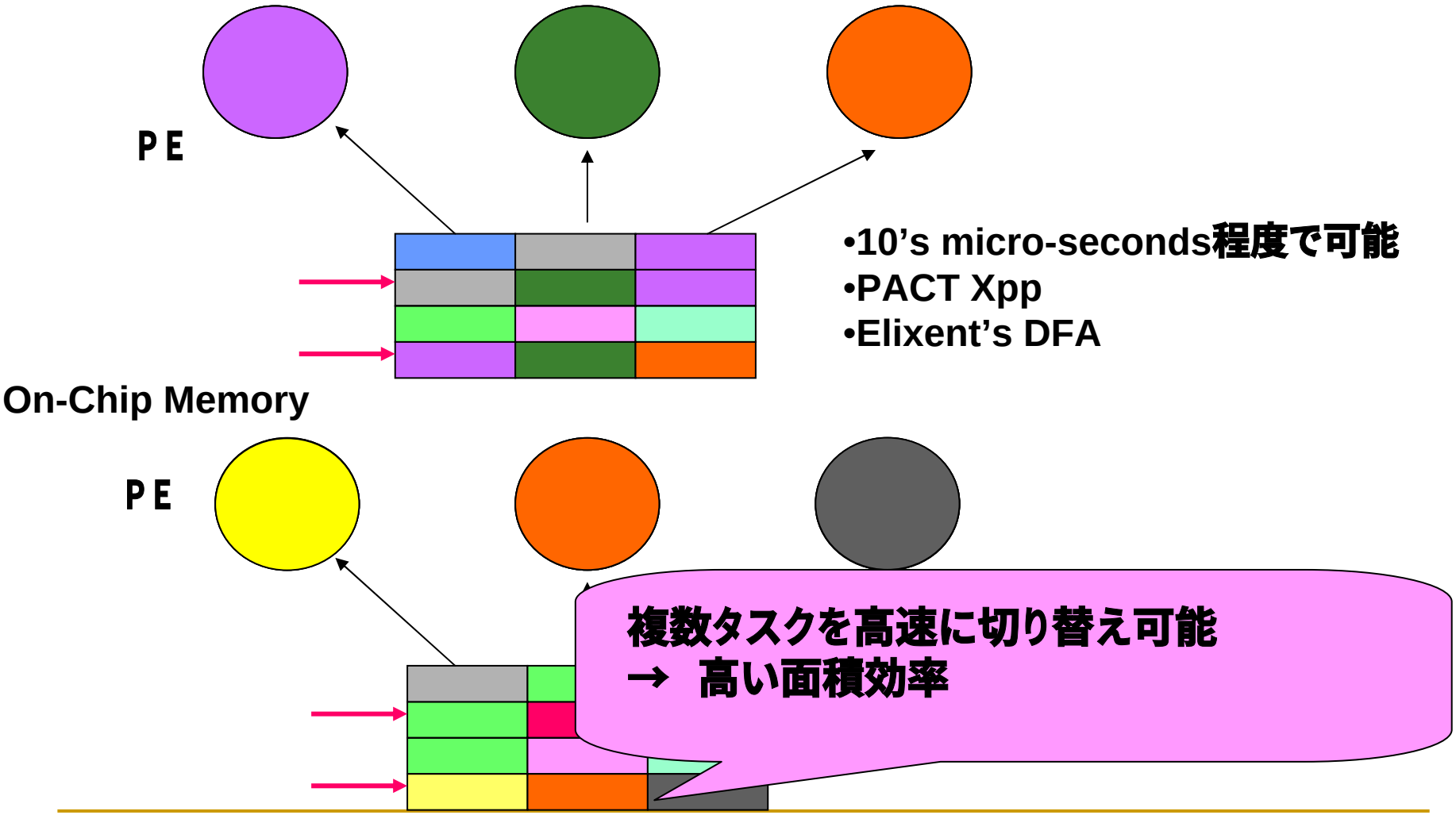
→　しかし、まだ専用ハードウェアには特に面積の点で不利

- 再構成能力を利用して、面積効率を高めることはできないか？

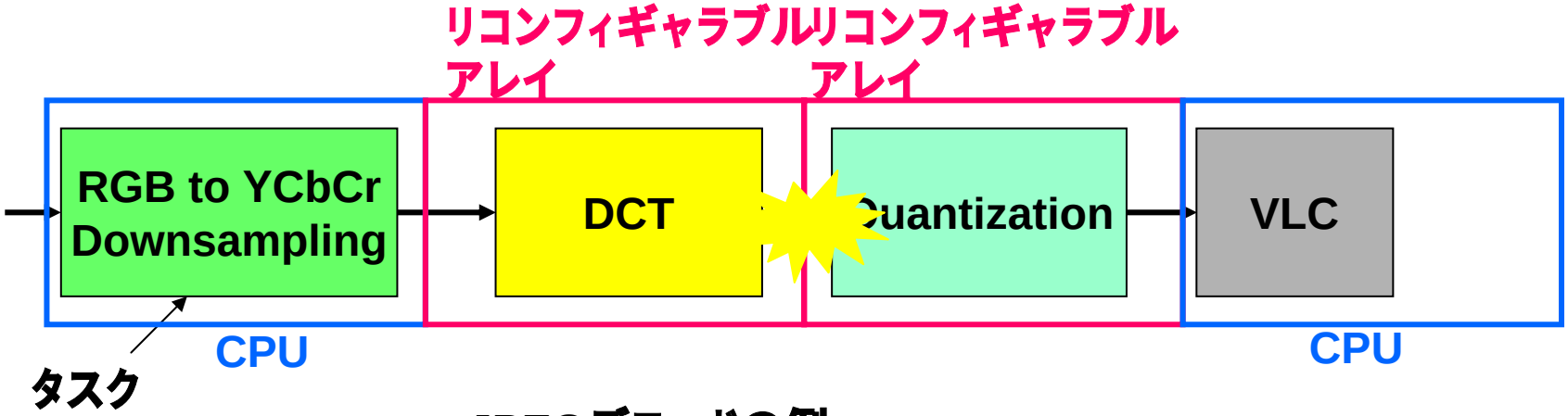
→　ダイナミックリコンフィギュレーション

動作中に構成を書き換えることにより、対象とする処理を時分割多重で実行

命令・構成情報配送方式


PE
On-Chip Memory
•10’s micro-seconds程度で可能
•PACT Xpp
•Elixent’s DFA
PE
複数タスクを高速に切り替え可能
→ 高い面積効率
On-Chip Memory

ストリーム処理とダイナミックリコンフィギャラブルデバイス



JPEGデコードの例

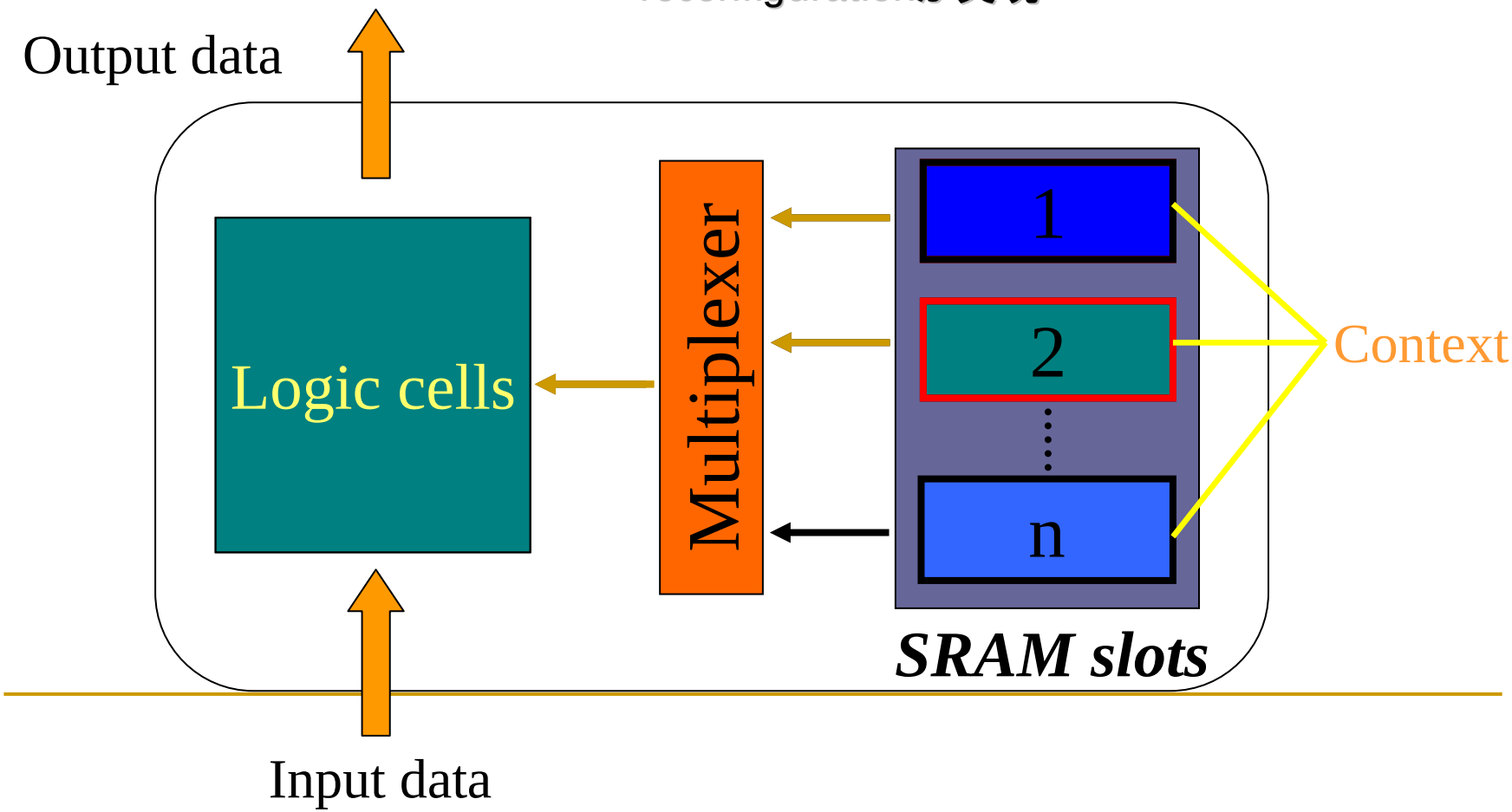
- 一連のタスクを連続してデータストリームに施す
- データストリームは一定の間隔で到着
  - この間隔には絶対に間に合うことが必要
  - 一定以上高速でも意味がない場合もある→それより電力を減らせ！
- 複雑だが計算量の少ないものと、単純で計算量が多いものなどさまざま
- CPUとダイナミックリコンフィギャラブルデバイスの協調処理が有効

→ 構成情報配送方式は、この処理に向いている

マルチコンテキスト方式

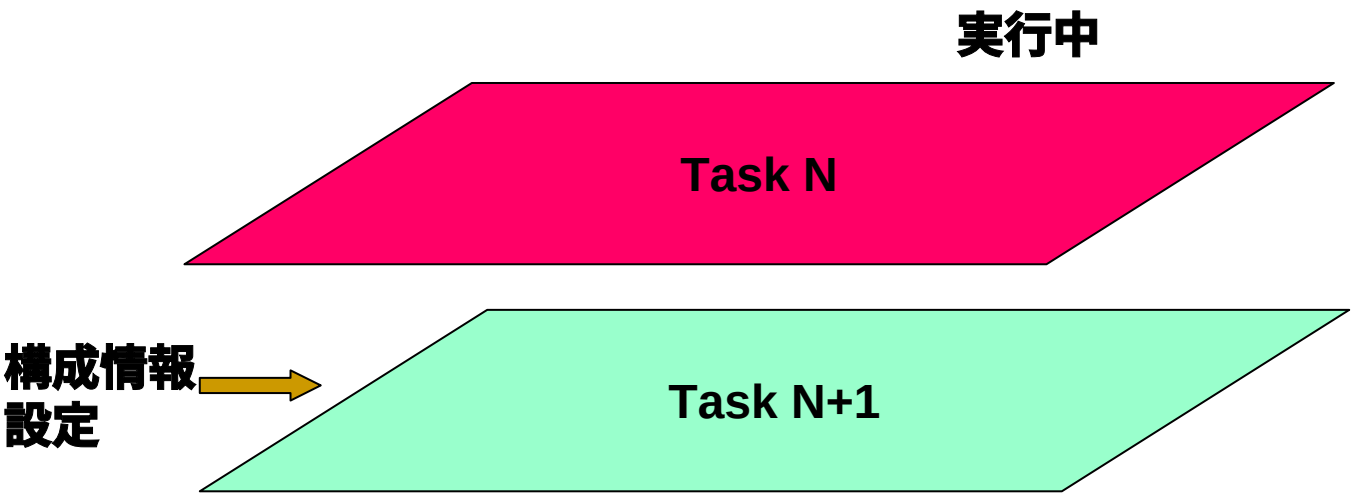
同一チップ上に複数のConfiguration RAMを持ち切り替えて用いる

ROMを使った富士通のMPLD(1990)、
WASMIIIでこれをRAM化(1992)、Xilinxも同様な提案を行う(1997)
NECのDRP(2002)により、Partial Context switch, runtime reconfigurationが実現



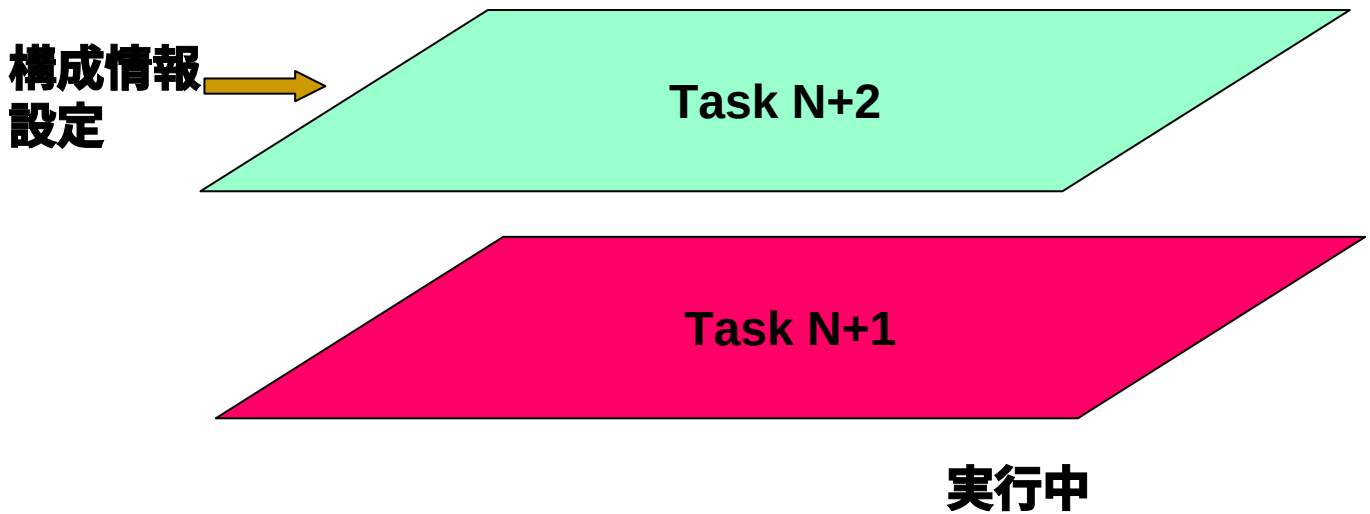
マルチコンテキスト方式の特徴

- タスク間の切り替えが、構成情報の読み込み時間なしで可能



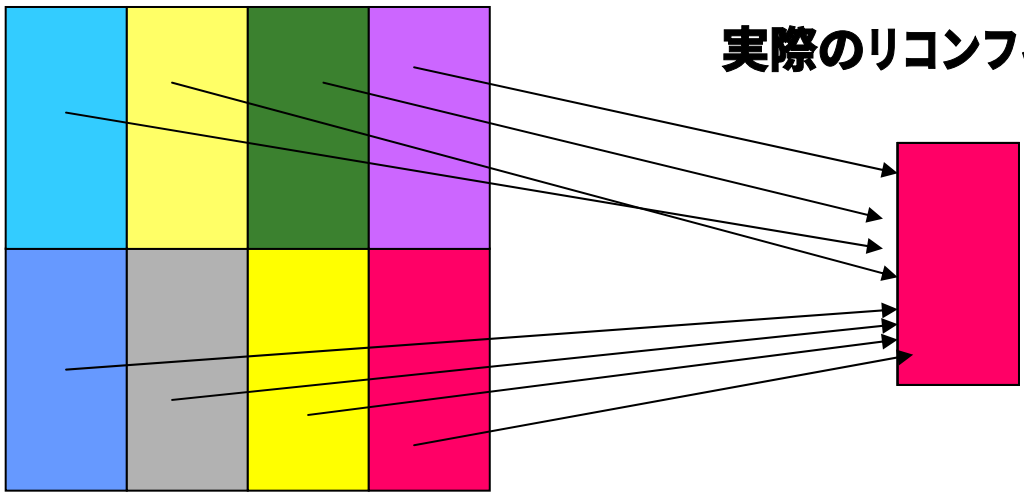
マルチコンテキスト方式の特徴

- タスク間の切り替えが、構成情報の読み込み時間なしで可能



単一タスクの時分割多重実行

実行対象回路

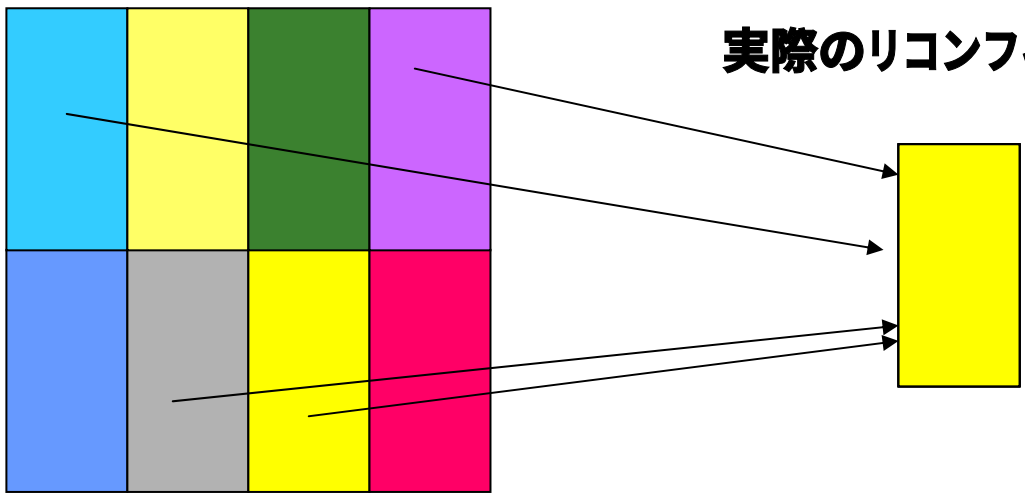


実際のリコンフィギャラブルデバイス

素直に実行するとn時分割多重は面積は1/nになるが、性能も1/nになってしまう

単一タスクの時分割多重実行

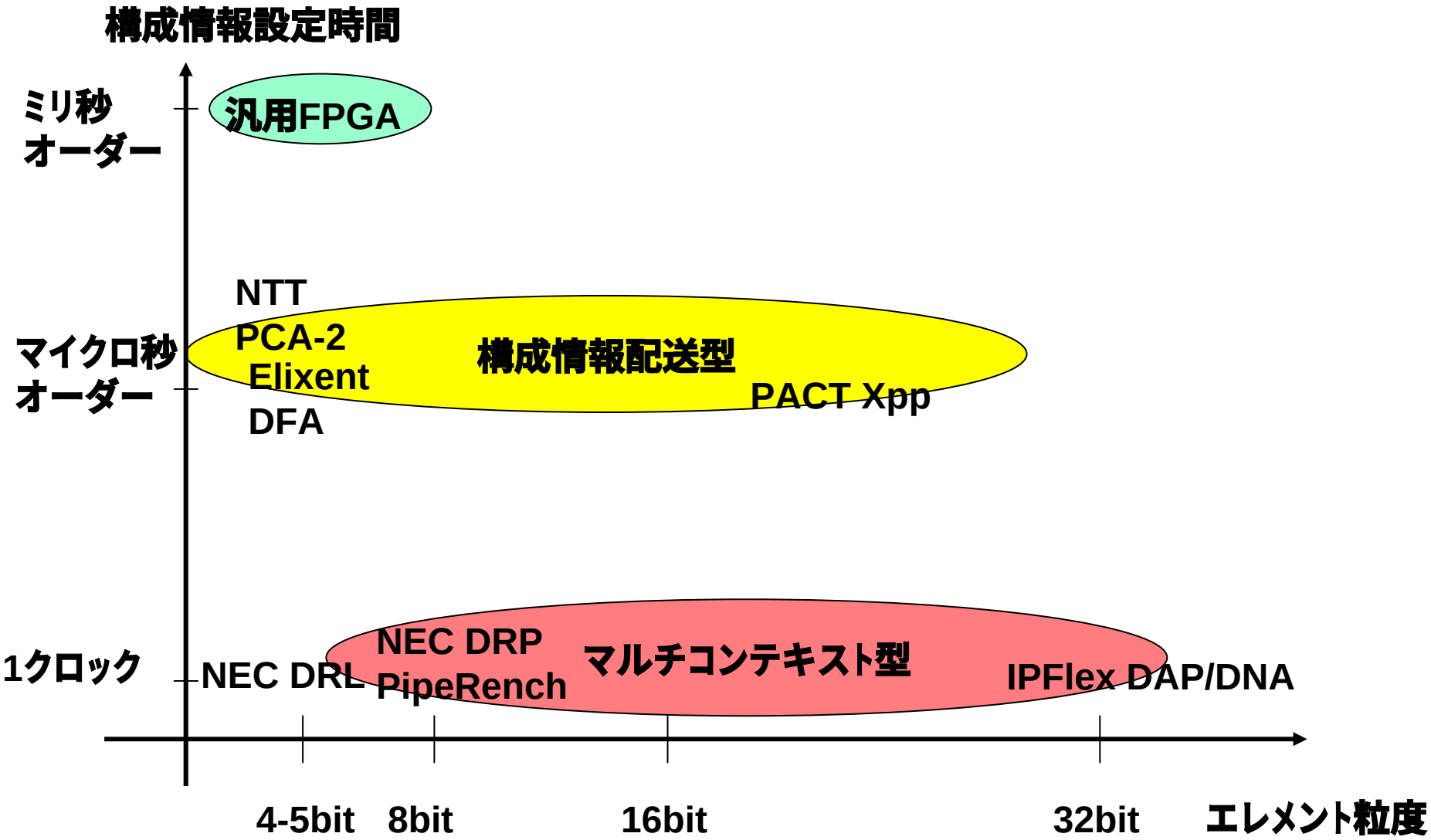
実行対象回路



実際のリコンフィギャラブルデバイス

実際は全回路がフルに動作していることは少ない
→　時分割多重実行は、面積効率が改善される

リコンフィギャラブルデバイスの位置づけ



ポイント3：C-level設計

- 特殊化したC言語(BDL, Dataflow C, Stream Cなど)
  - ビット幅記述の拡張、合成系への指示
  - メモリへのポインタ、関数へのポインタなどの禁止
- 多数の演算器、シフタ、レジスタ、分散メモリ上に処理回路を実現する。
  → 計算機用コンパイラよりもASIC設計用の高位合成技術が用いられる。
- 構成情報配送型

→ 演算アレイ上に演算フローをマップ

- マルチコンテキスト型

→ コンテキストの切り替えによってデータパスを一瞬で切り替える。

- 一般のVerilog HDLやVHDL記述のからの変換よりむしろC-level記述からの変換が楽

→ 演算器ライブラリを組み合わせる方法を取る (これはあんまり書きやすくない)

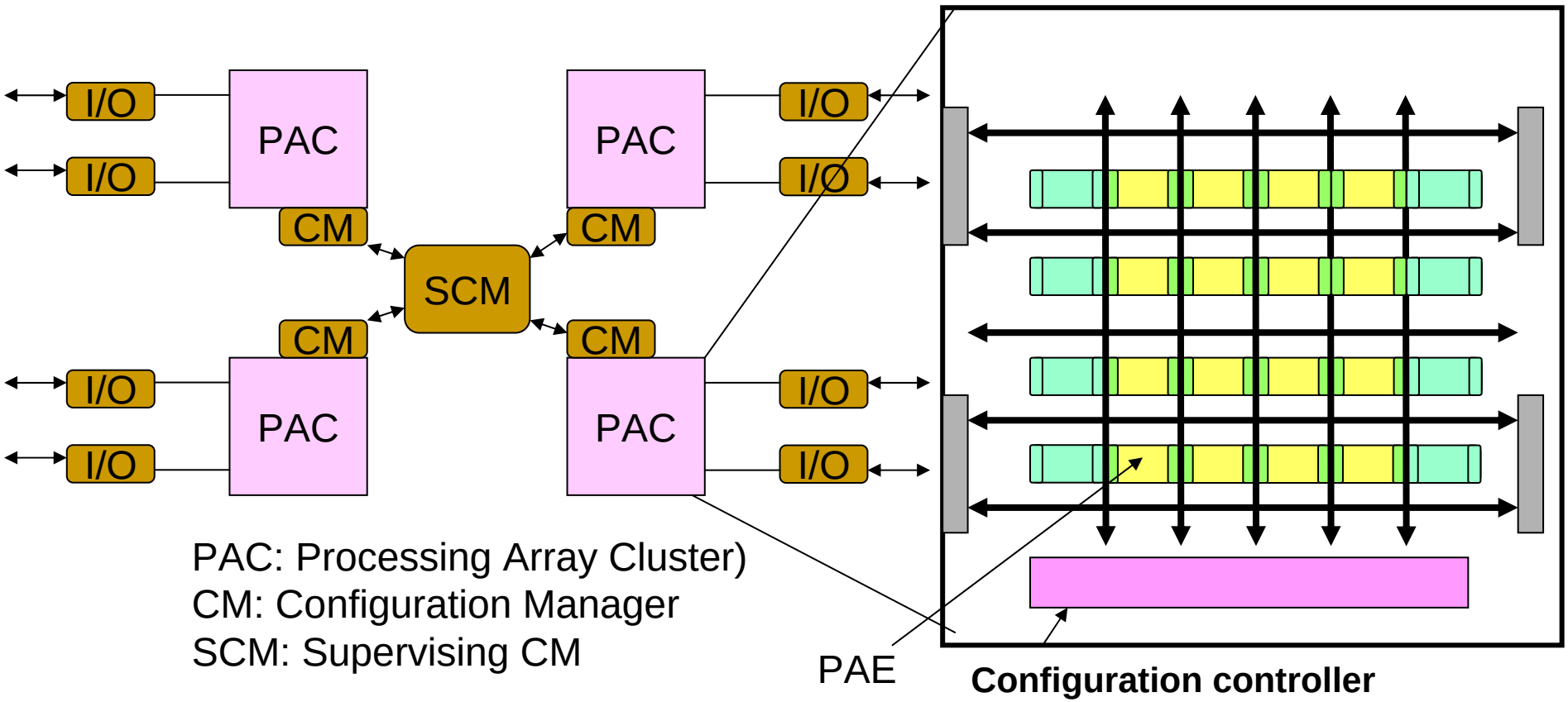
紹介の流れ

- ダイナミックリコンフィギャラブルプロセッサ導入の動機
- 3つのポイント
- ダイナミックリコンフィギャラブルプロセッサ紹介
  - 命令/構成情報配送方式
  - マルチコンテキスト方式
  - チップマルチプロセッサとの境界的デバイス
- DRPの世界
- おわりに

構成情報配送型の紹介

- 粗粒度のリコンフィギャラブルデバイスは90年代後半より様々の研究用のプロトタイプが作成されている。
- 商用の粗粒度構成デバイス
  - PACT社xpp
  - Elixent社DFA1000
  - メディア処理を対象とし、DSPの代替を意識している
- 独自のユニークなデバイス
  - NTT社PCA/PCA-2
  - 京大、長崎大などで類似の方式の研究が行われている。

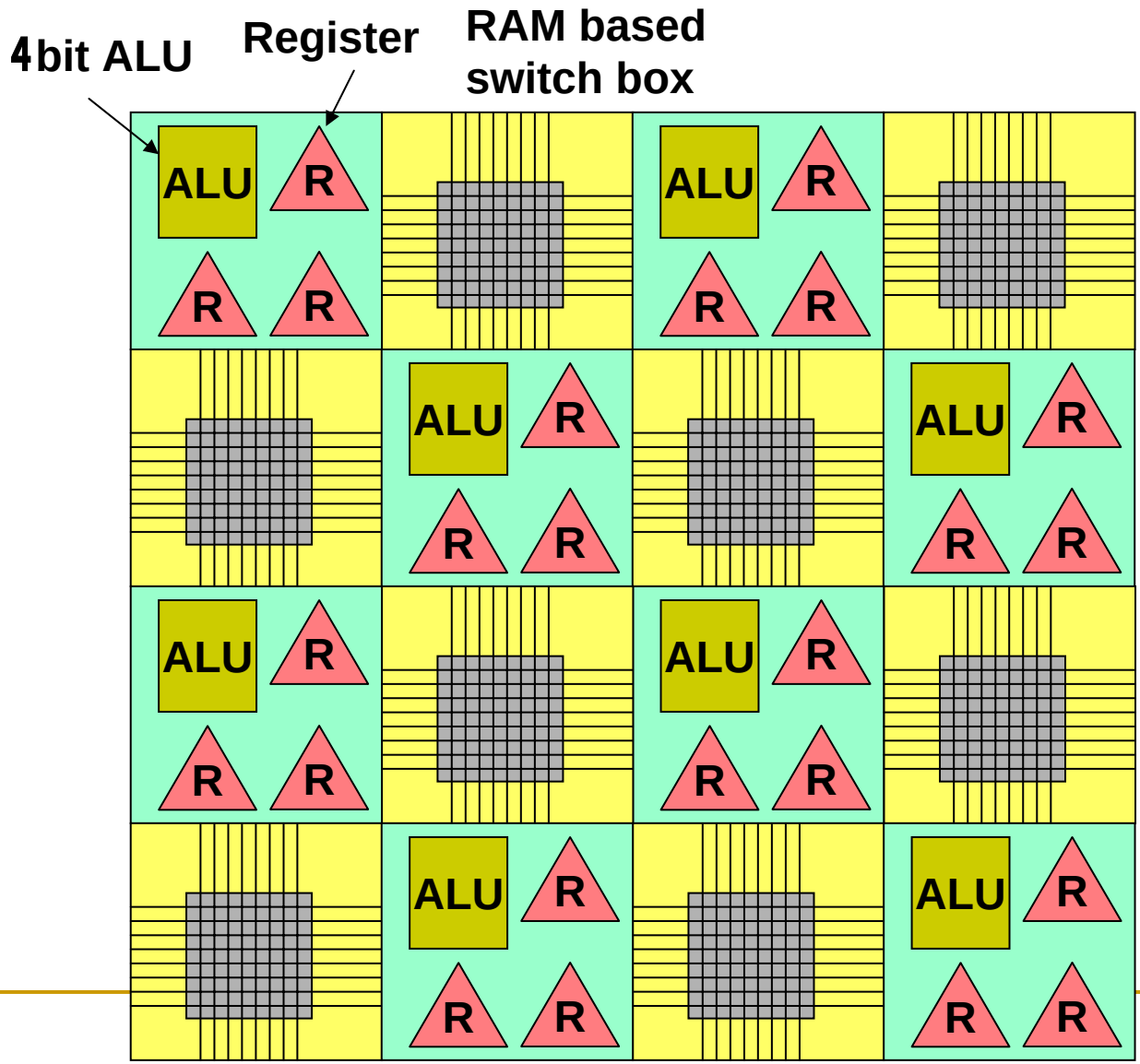
Xpp (PACT Informations technologie)



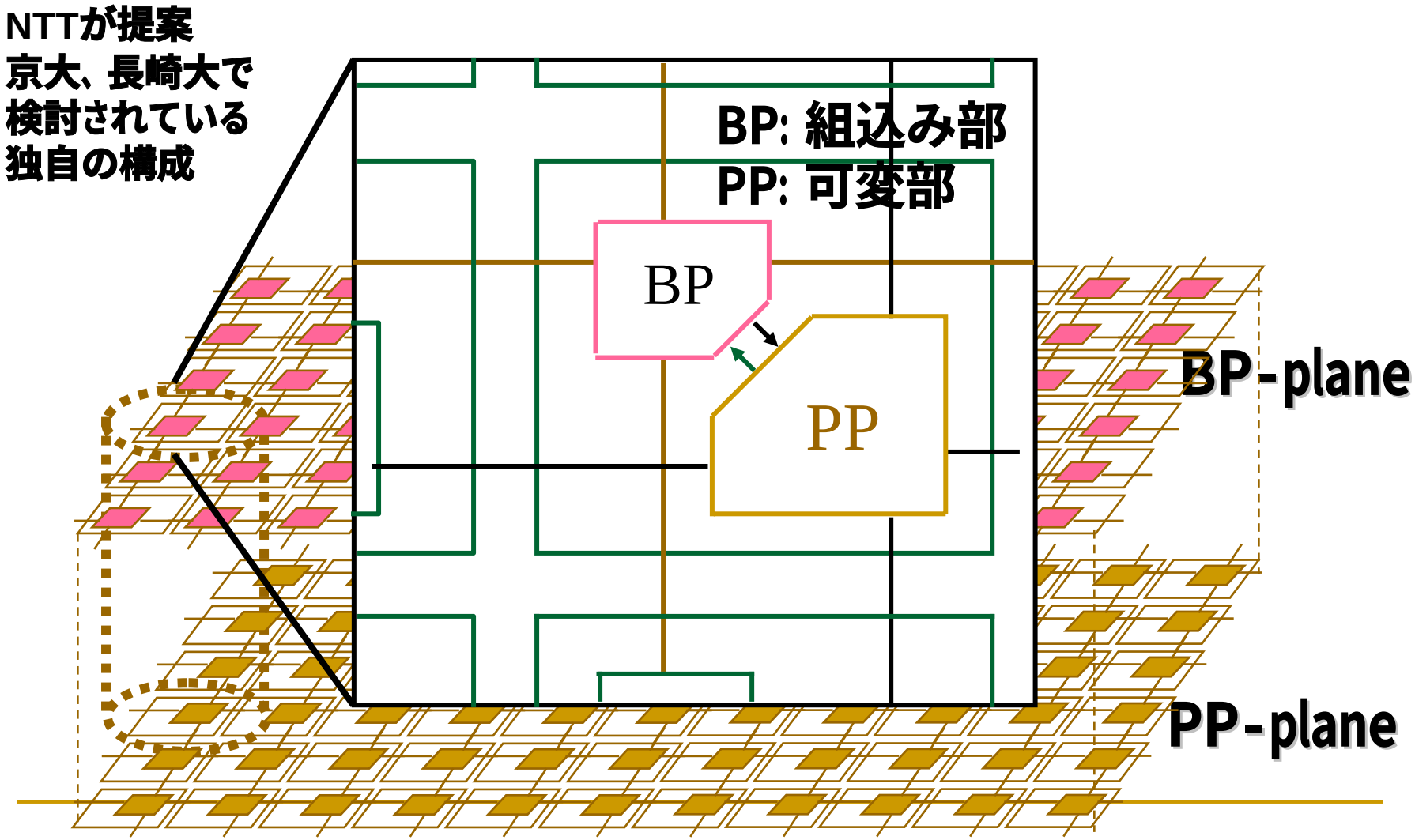
PAC: Processing Array Cluster)
CM: Configuration Manager
SCM: Supervising CM

Xpp64 (8x8 のＰＡＣ) が現在稼動
Configuationには数100クロック要する
ＰＡＥは24ビット幅、クロックは４０ＭＨｚ

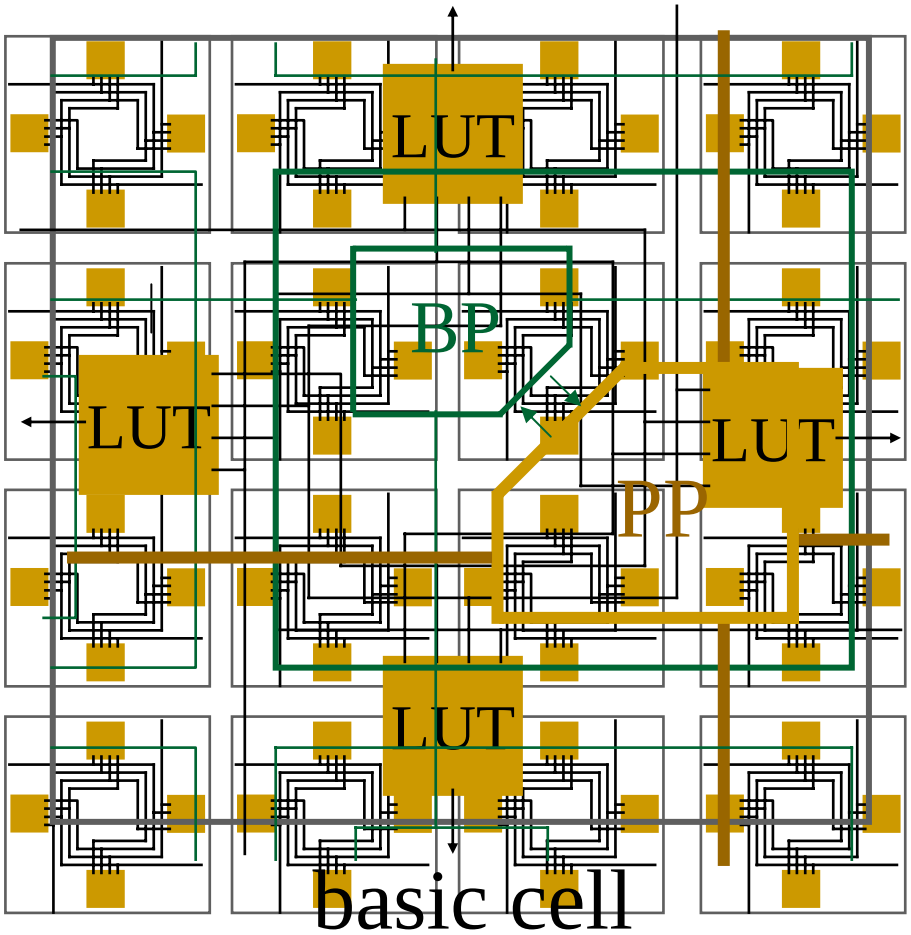
Elixent社DFA1000の演算アレイ



PCA (Plastic Cell Architecture)



Plastic Partの構成



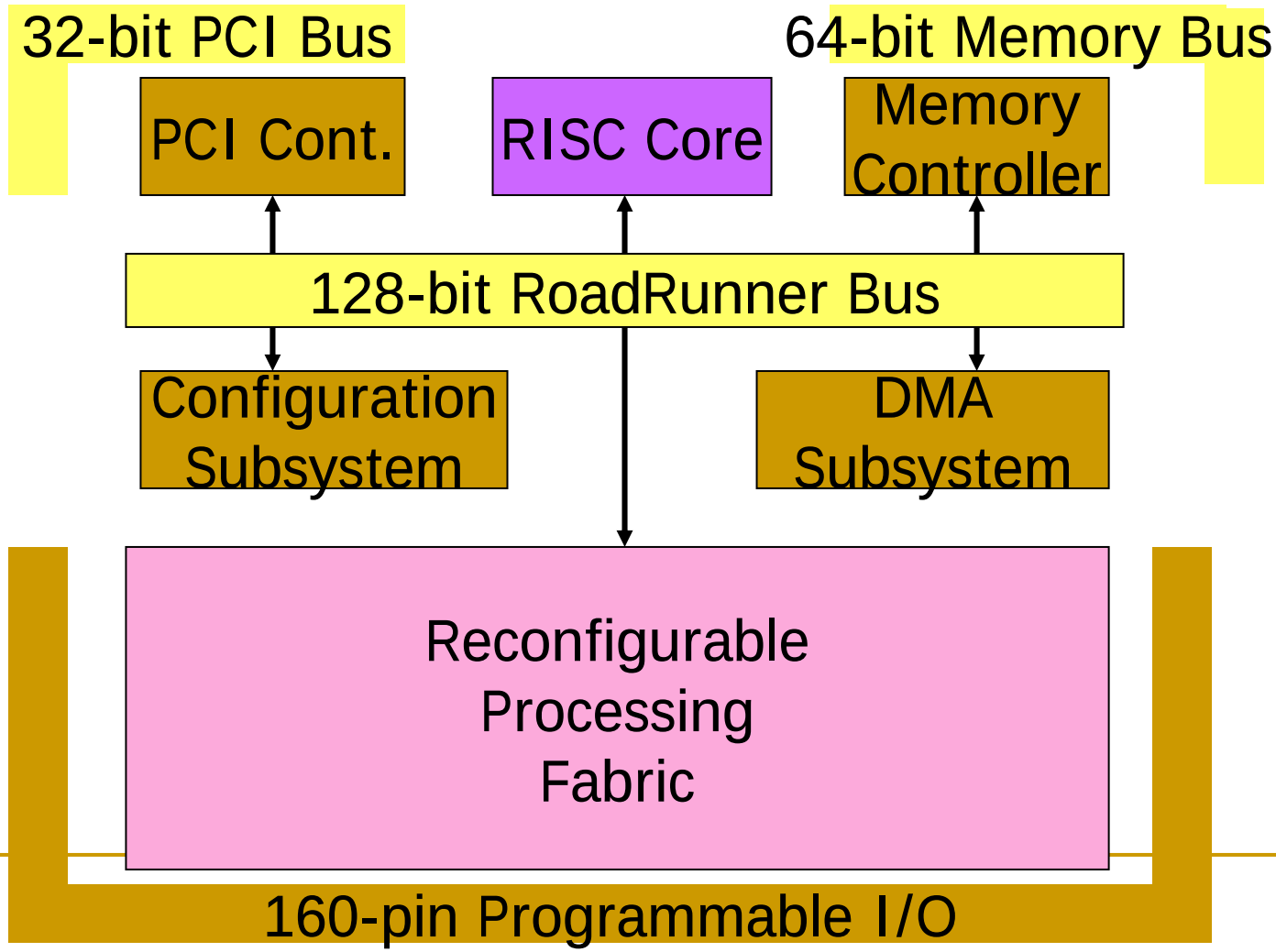
- 4入力1出力のLUT
- オブジェクト単位で構成
  - 機能オブジェクト
  - メモリオブジェクト
- 部分的に再構成可能
- 動的再構成が有利な構成
  - 非同期動作
  - Buit-in-partにおけるパケット転送
  - 動的Hw　forkが可能

マルチコンテキスト型の紹介

- **粗粒度構成の商用機はChameleon社が最初**
  → しかし商業的に失敗してしまった。
- **日本企業ががんばっている**
  - IPFlex社DAP/DNA, DAP/DNA-2
  - NECエレクトロニクス　DRP
- **オンチップマルチプロセッサとの境界タイプの商用化も進む**
  - Quicksilver社ACM
  - MorphTech社rDSP
  - PicoChip社PC101

これは後でゆっくりと

Chameleon CS2112

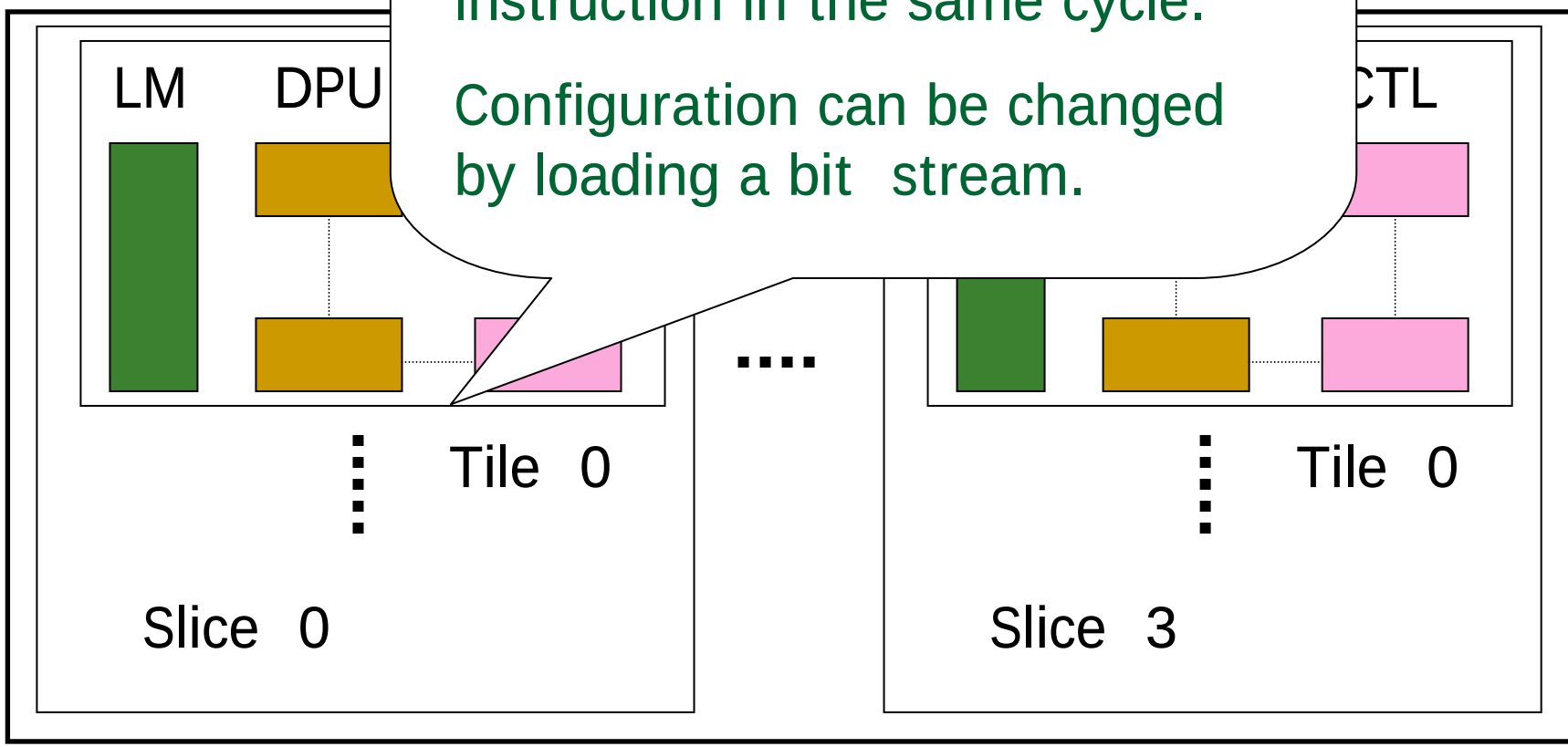


Reconfigura
Chameleon

8 instructions stored in the CTL are executed in the DPU.

The CTL can select the next instruction in the same cycle.

Configuration can be changed by loading a bit stream.

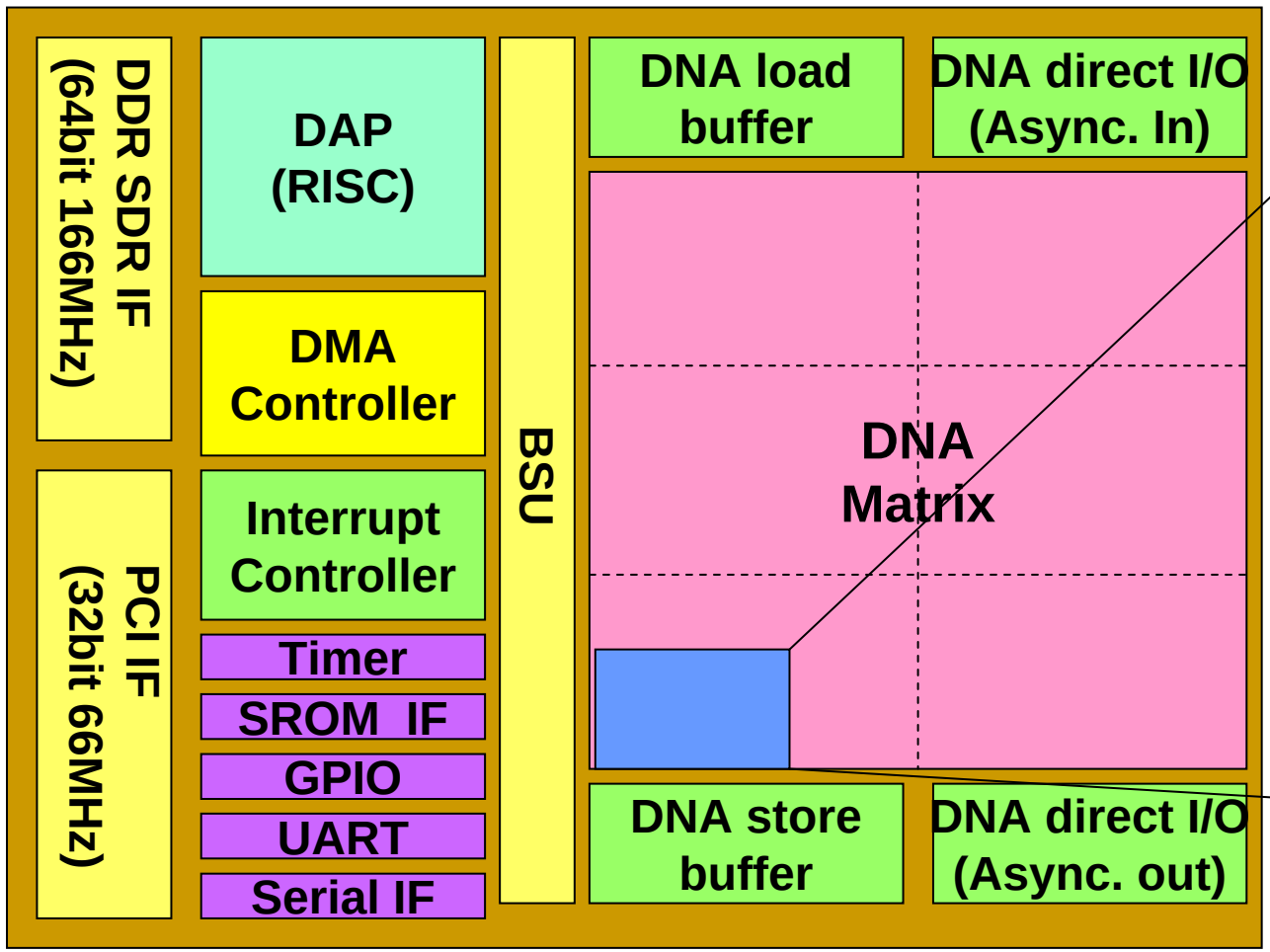


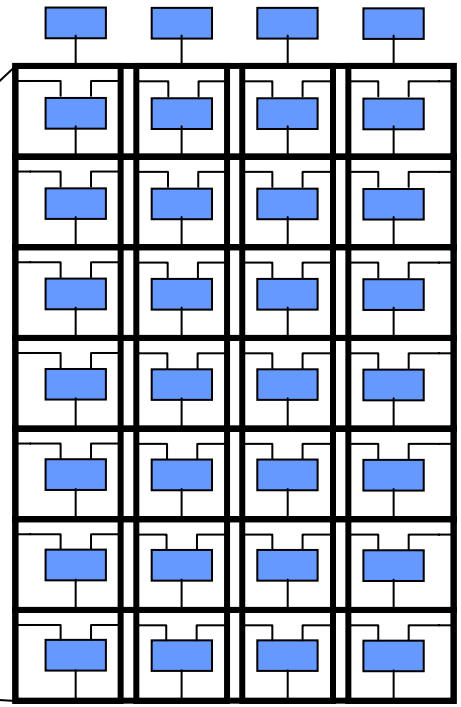
108 DPU(Data Path Unit)s consists 4 Slices (3Tiles each)

1Tile: 9DPU＝32bit ALU X 7 16bit + 16bit multiplier X 2

IPflex社DAP/DNA-2の全体構成

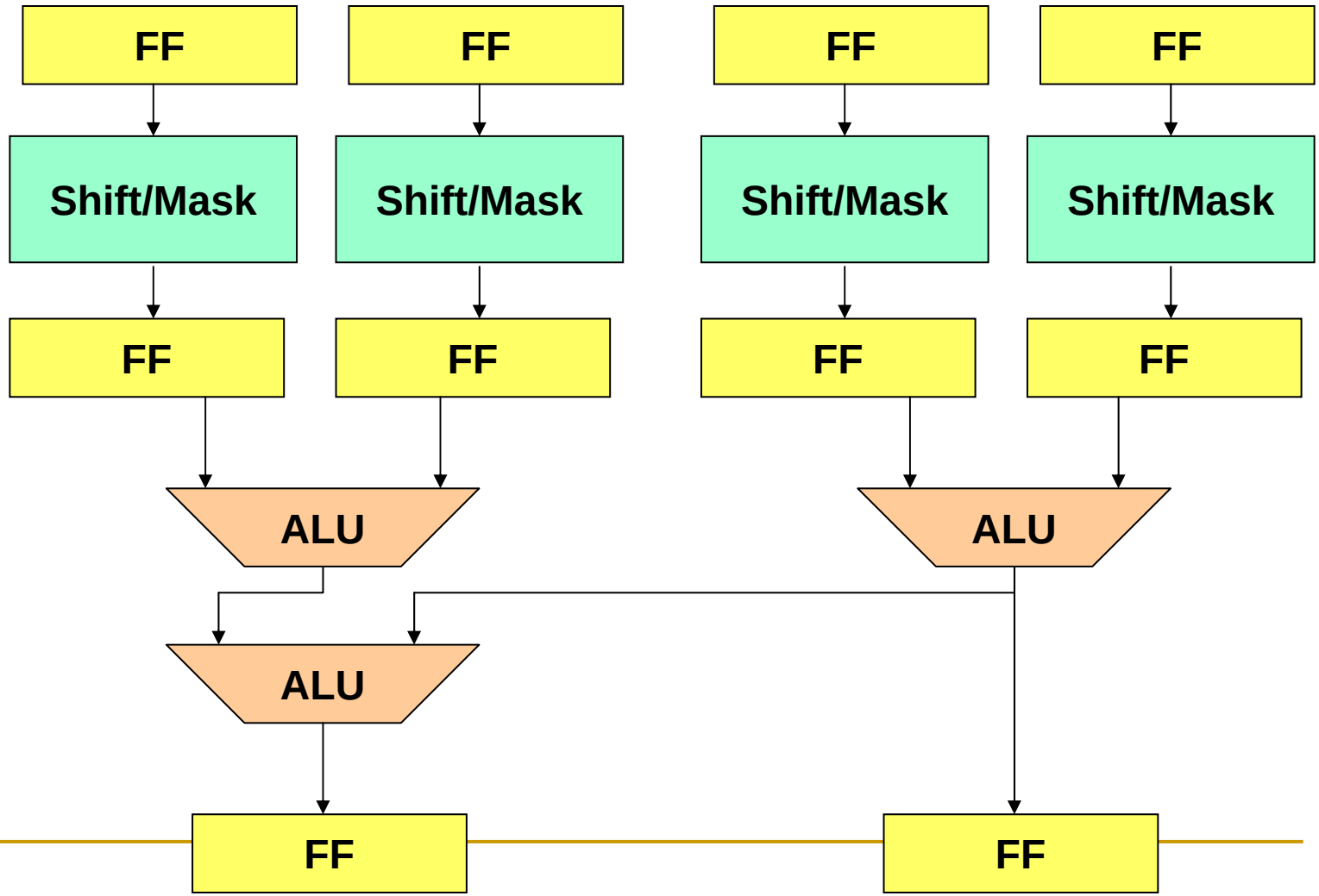
368個の演算アレイ
ALU,メモリ、遅延素子
等ヘテロジーニアス



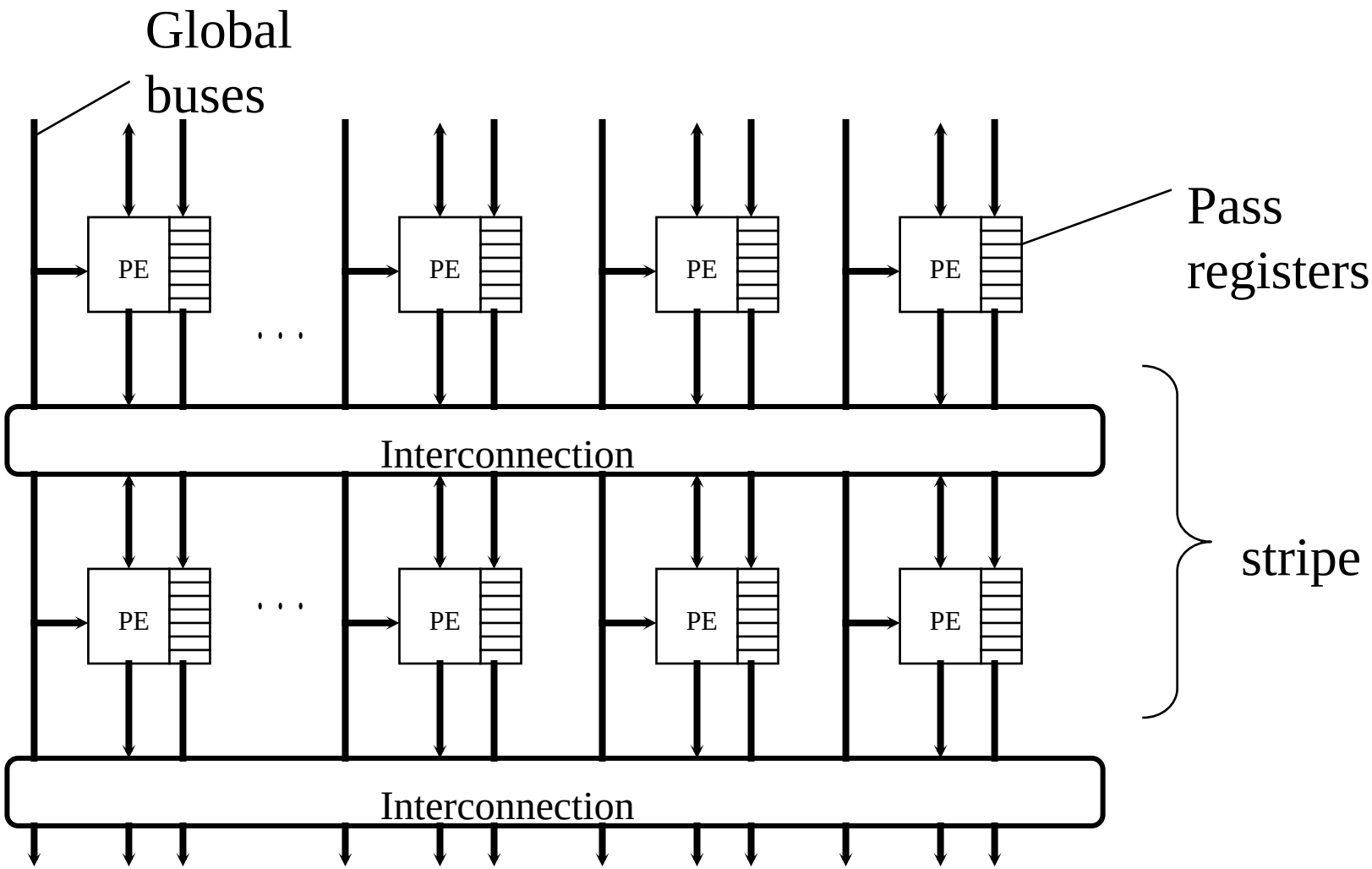


DNAマトリックスのエレメント

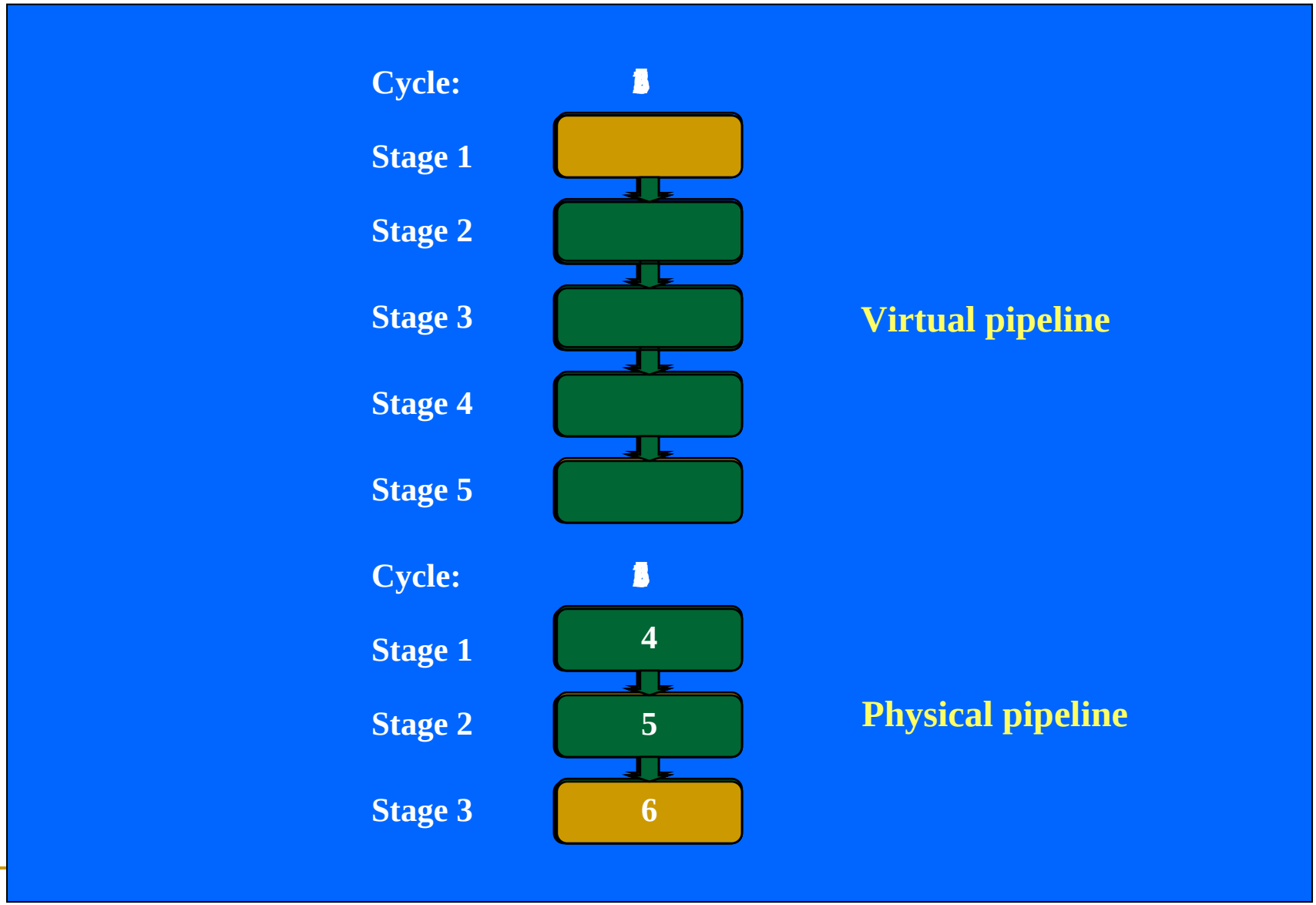
SMAの構造： 遅延素子、RAM、入出力等8種類がある



PipeRench （CMU)



Pipelined Reconfiguration



チップマルチプロセッサとの境界タイプ

- 各PEがPCを持ち、独立して命令をフェッチして分岐する能力を持つ

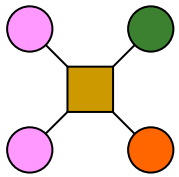
→したがって、分類するとなるとチップマルチプロセッサとなる

- しかし、以下の点で従来のチップマルチプロセッサよりもダイナミックリコンフィギャラブルプロセッサに近い
  - PE構成がヘテロジーニアス
  - 単純なPEを多数用いる
  - プログラムが埋め込み
  - PE間接続は単純
  - コンテキスト切り替えの概念が存在
  - 対象がストリーム処理、メディア処理、信号処理
    - ライバルとしてDSPを意識している
  - Quicksilver社ACM
  - MorphTech社rDSP
  - PicoChip社PC101

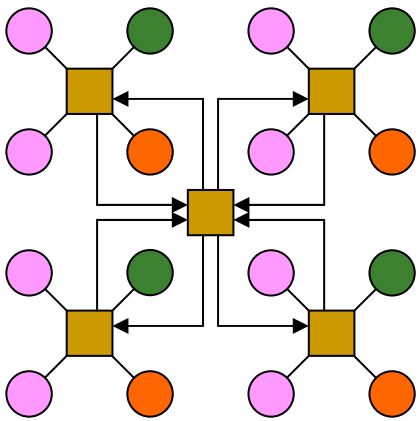
ACM (Quicksilver)

Matrix Interconnect Network

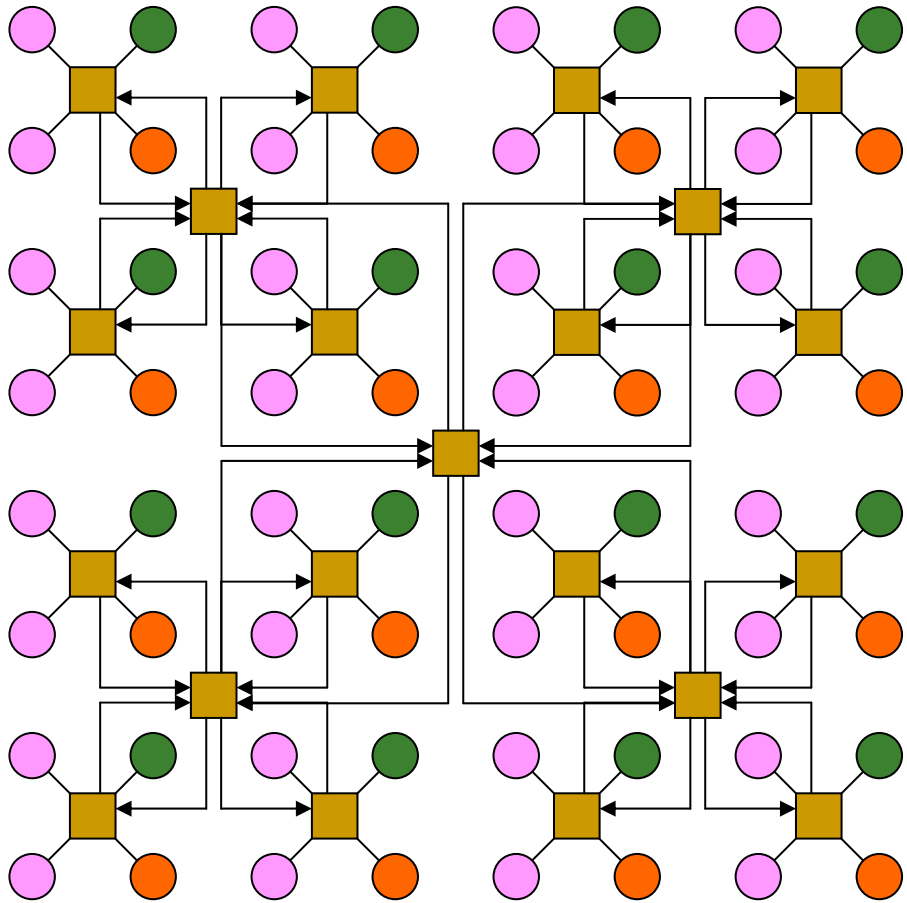
Adaptive Node Programmable Node Domain Node



Level1 Cluster



Level2 Cluster

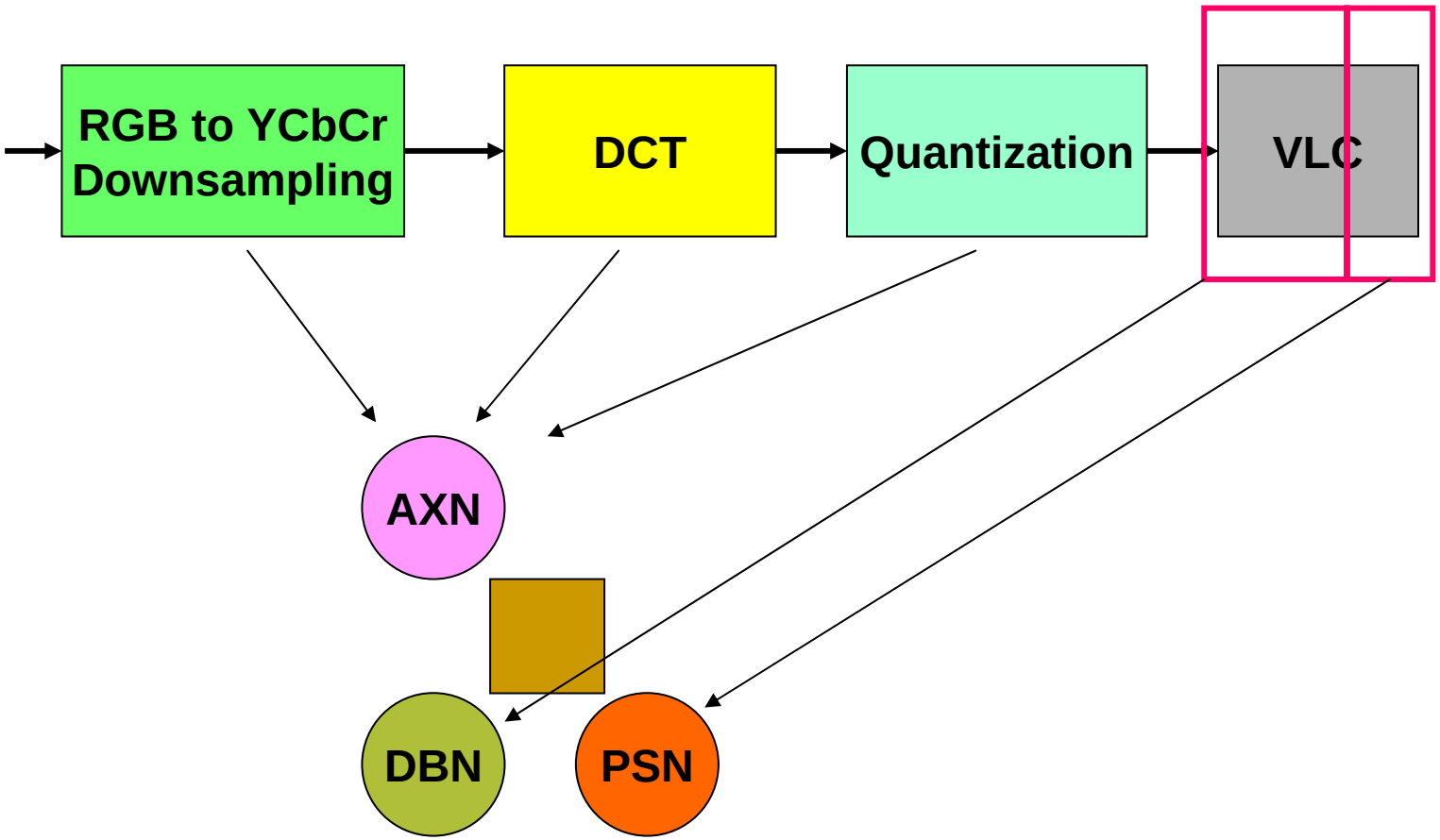


Level3 Cluster

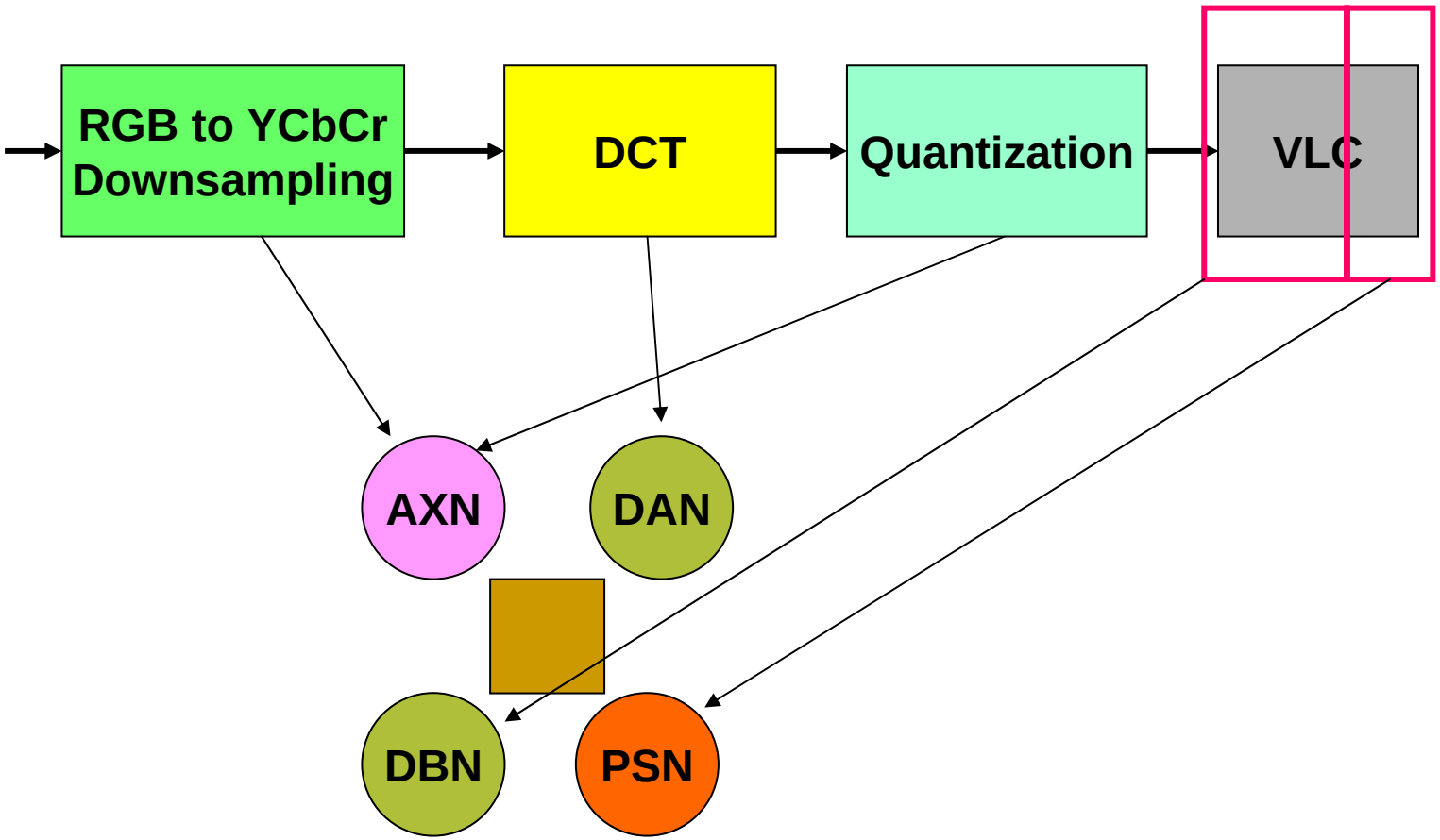
ＡＣＭの特徴

- ダイナミックリコンフィギャラブルプロセッサとは若干違う
- 様々な種類のノード
  - ローカルメモリと自律制御機能を持つ
  - Adaptive Node
    - 強力な演算器、レジスタ＋制御回路
  - Domain Node
    - Filter, Bit制御等の専用演算器
  - Programmable Scalar Node
    - RISC CPU (ARCの拡張)
- 1ワードのパケットを転送するＭＩＮ
- 31コンテキストを高速に切り替え可能
- データ(ストリーム)駆動によりノード間の処理の制御
- Silver-Cによる問題記述(一部はアセンブラ)

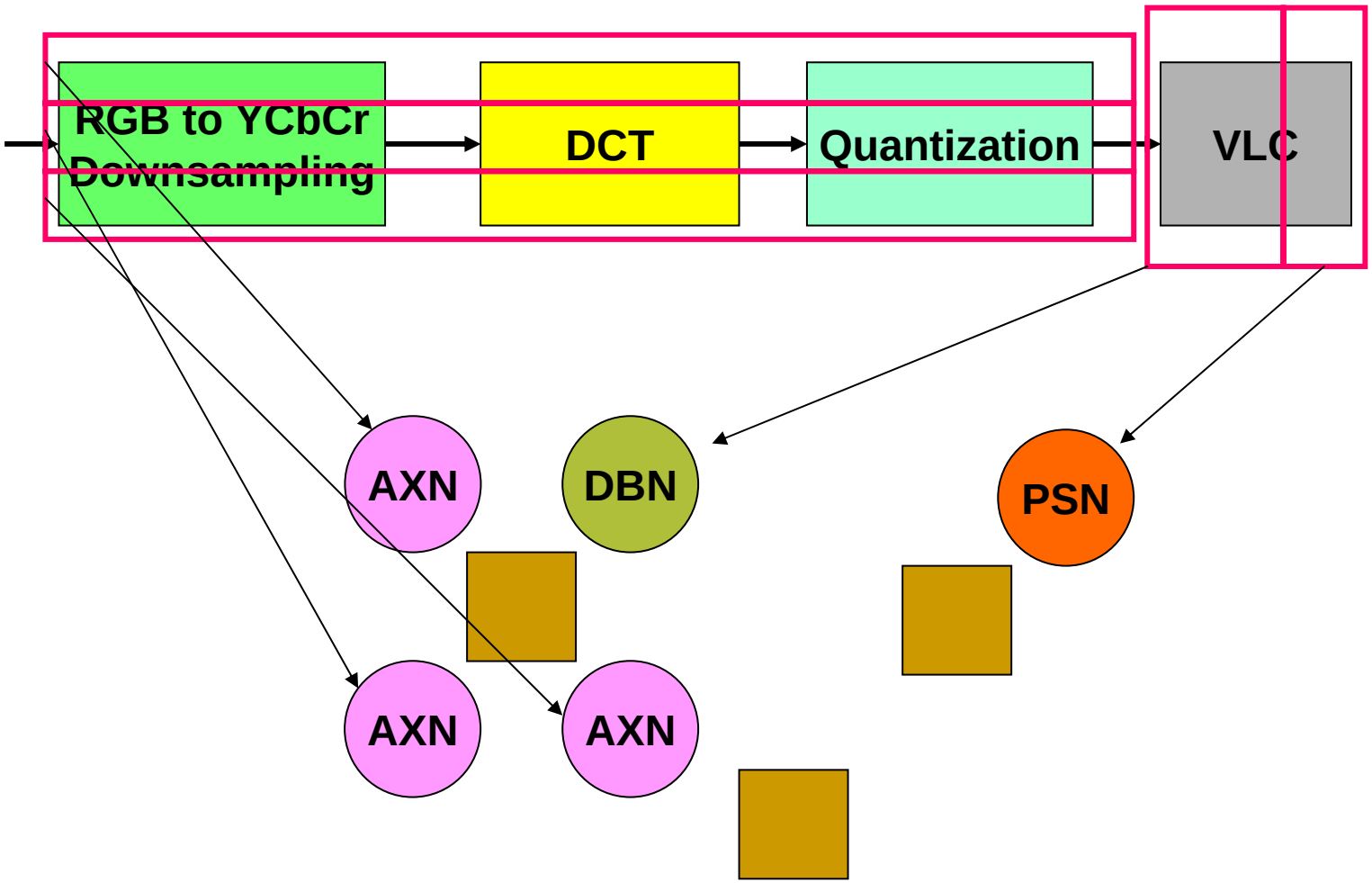
ＡＣＭにおける処理の例（ＪＰＥＧデコード）



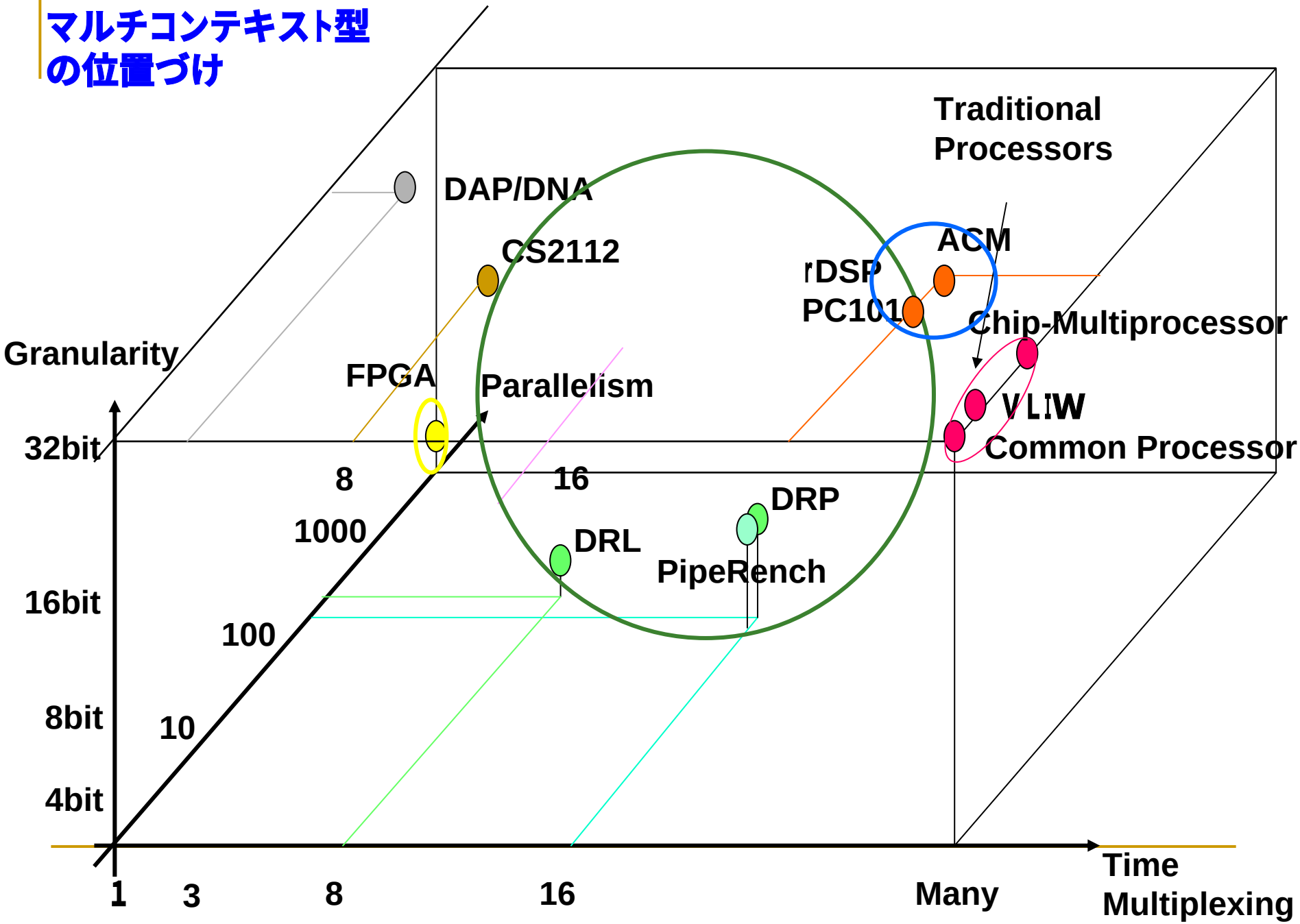
ＡＣＭにおける処理の例（ＪＰＥＧデコード）



ＡＣＭにおける処理の例（ＪＰＥＧデコード）



マルチコンテキスト型の位置づけ



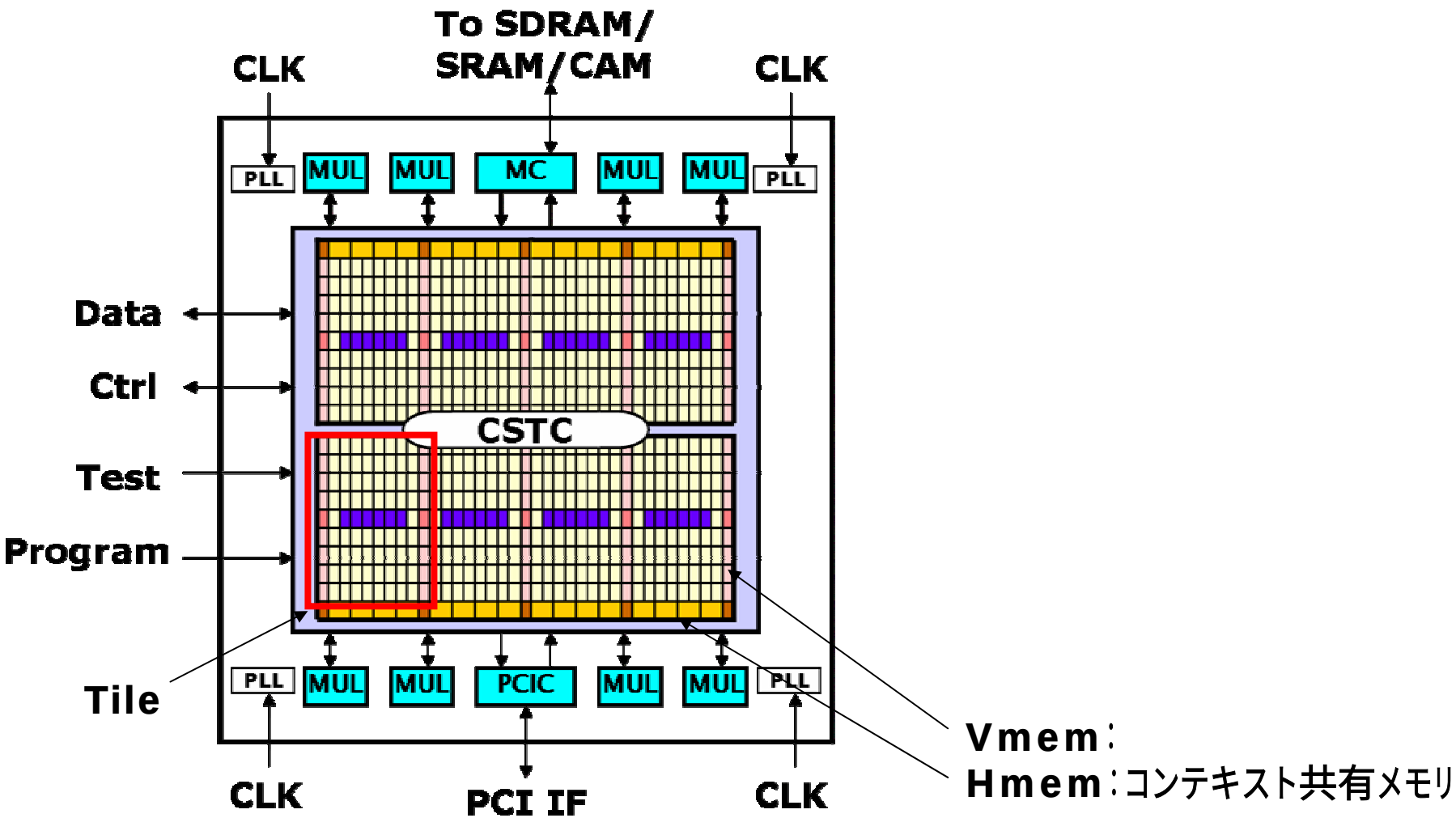
紹介の流れ

- ダイナミックリコンフィギャラブルプロセッサ導入の動機
- 3つのポイント
- ダイナミックリコンフィギャラブルプロセッサ紹介
- DRPの世界
  - デバイスの特徴
  - 使い方
  - 設計例
- おわりに

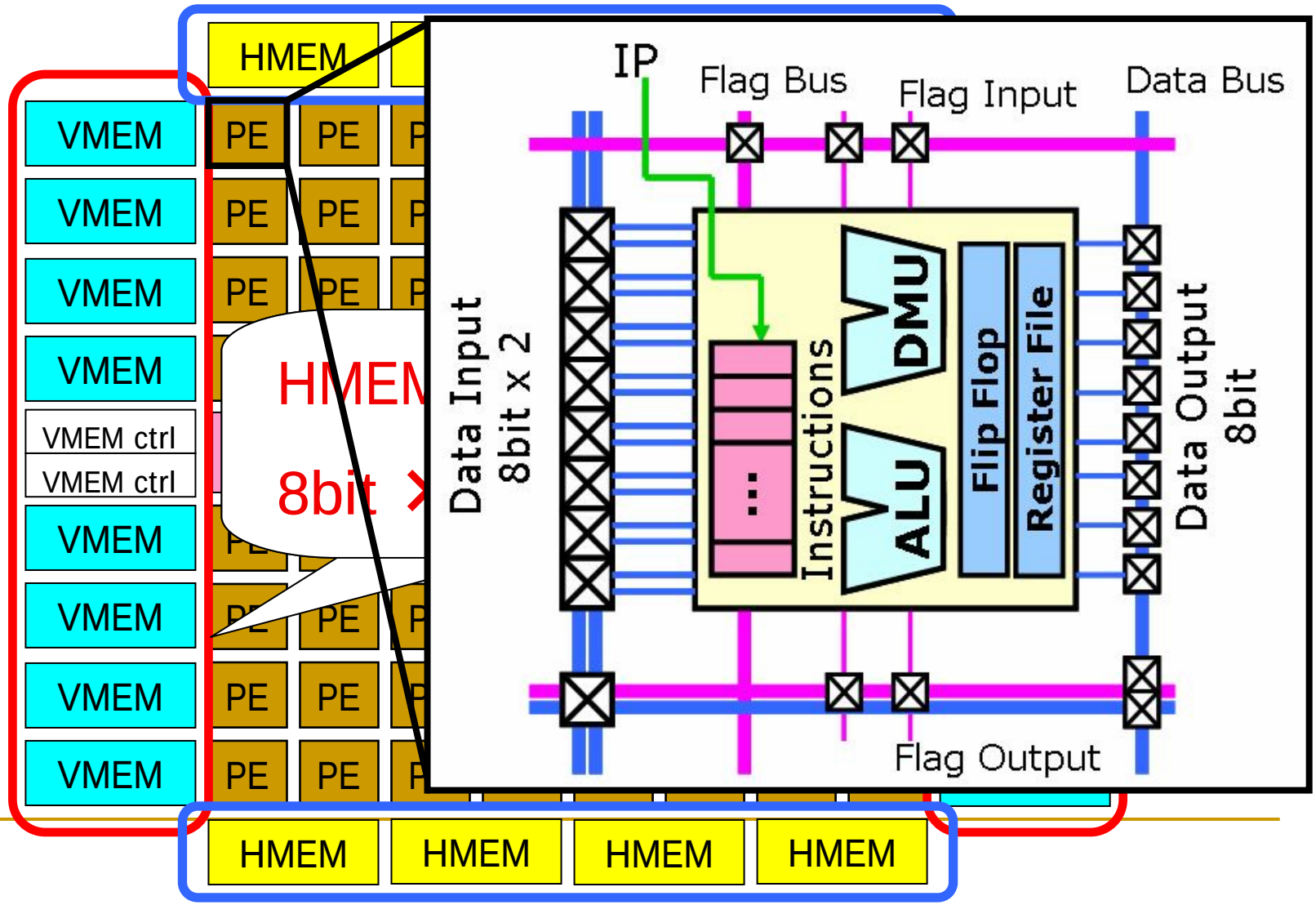
NECエレクトロニクス
DRP (Dynamically Reconfigurable Processor)の特徴

- マルチコンテキスト型
  - 最大で16コンテキスト分の情報を保持
  - 5コンテキストの中の一つに状態遷移が可能
  - 切り替え対象外のコンテキストへのコンフィギュレーションロードに対応
- プロセッサアレイ構成
  - チップ全体に配置されているPEによって並列かつ高速な演算が可能
- 部分再構成可能
- メモリ拡散配置型アーキテクチャ
- 単一タスクの時分割実行、コンテキスト内の並列実行両方が可能なダイナミックリコンフィギャラブルプロセッサ

プロトタイプチップ DRP-1

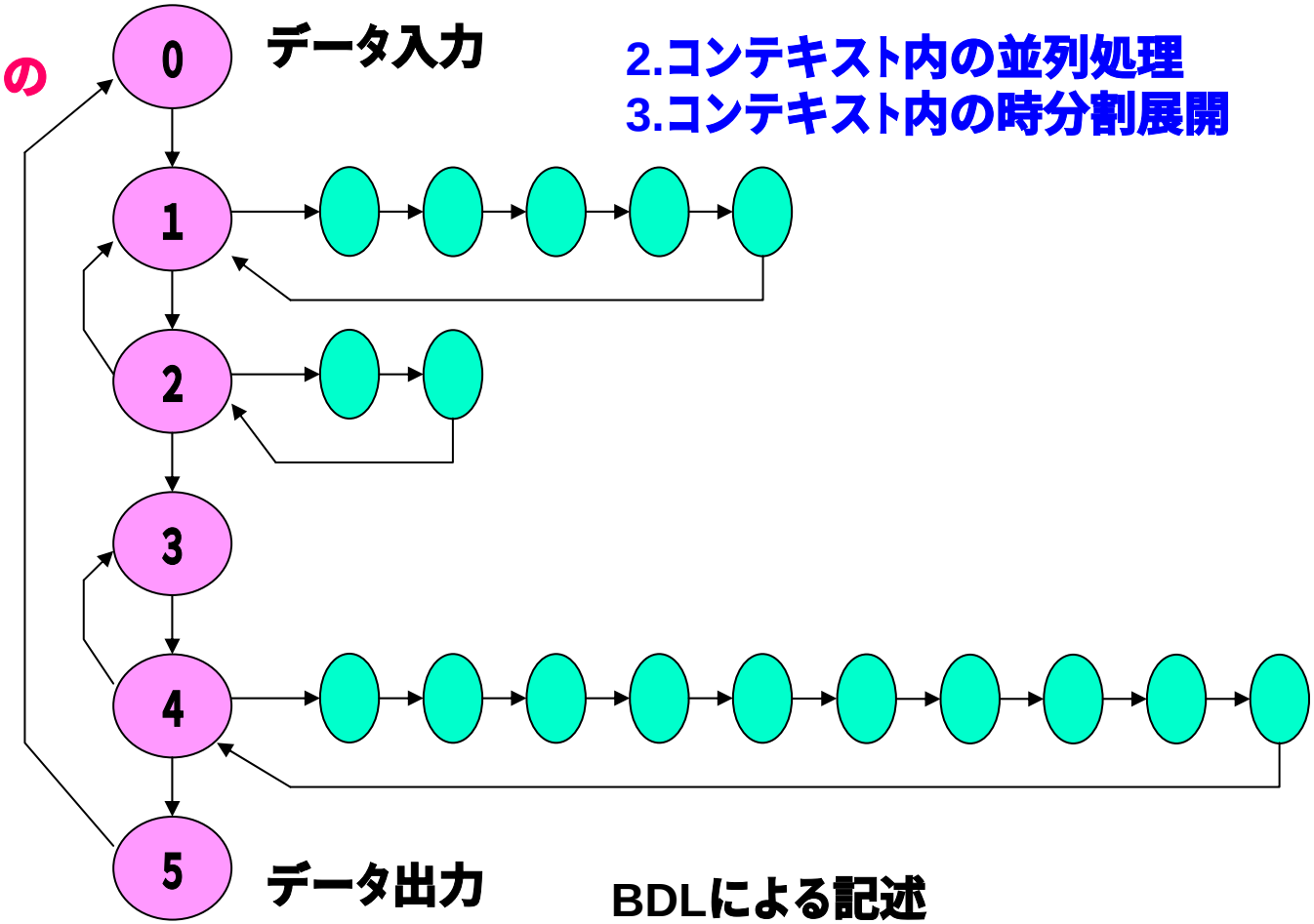


DRP Tileの構造


HMEM
VMEM
PE
VMEM ctrl
HMEM
8bit
IP
Flag Bus
Flag Input
Data Bus
Data Input
8bit x 2
Instructions
DMU
ALU
Flip Flop
Register File
Data Output
8bit
Flag Output
HMEM
HMEM
HMEM
HMEM

ＤＲＰのコンテキスト制御

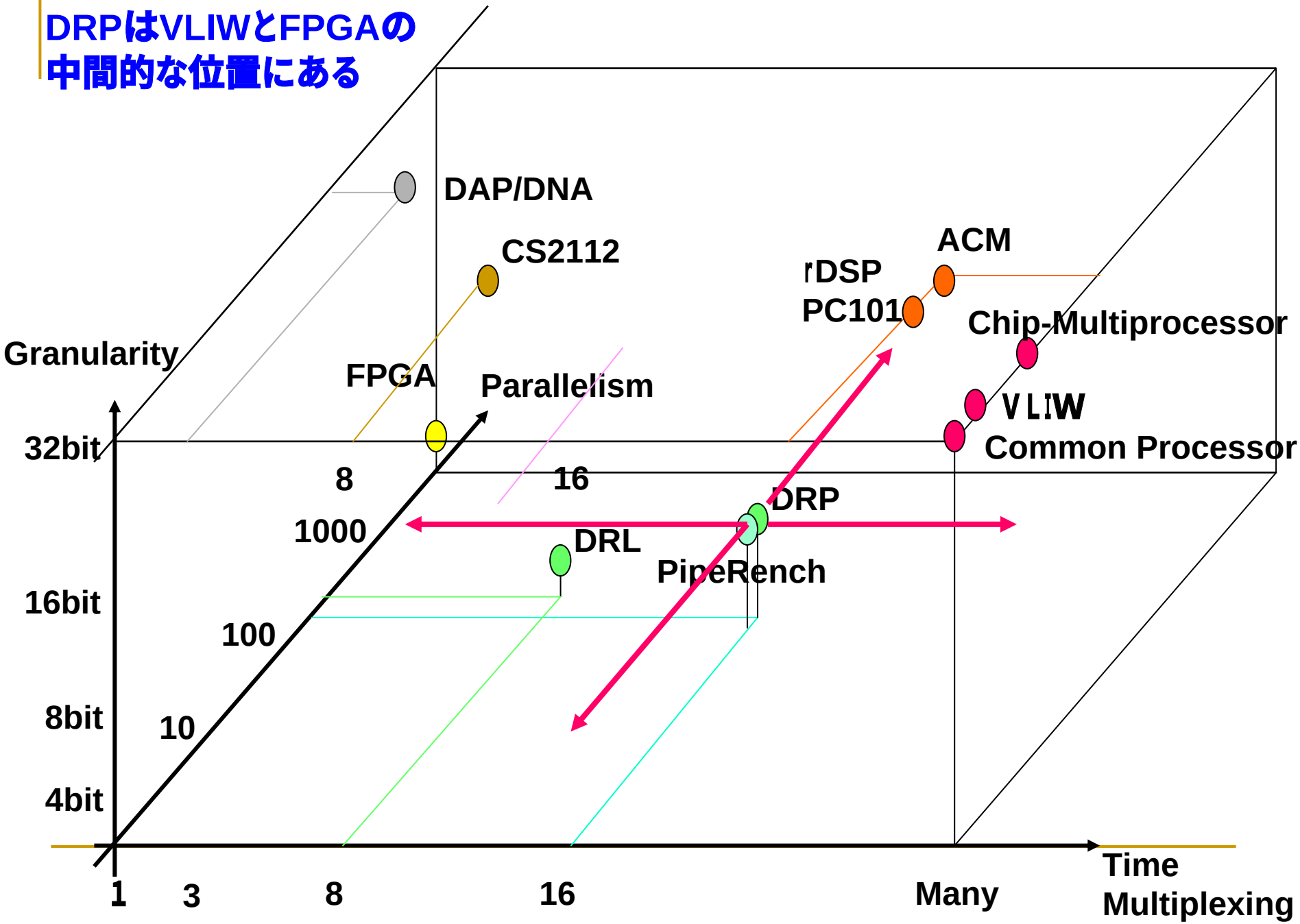
1.コンテキストの
切り替え



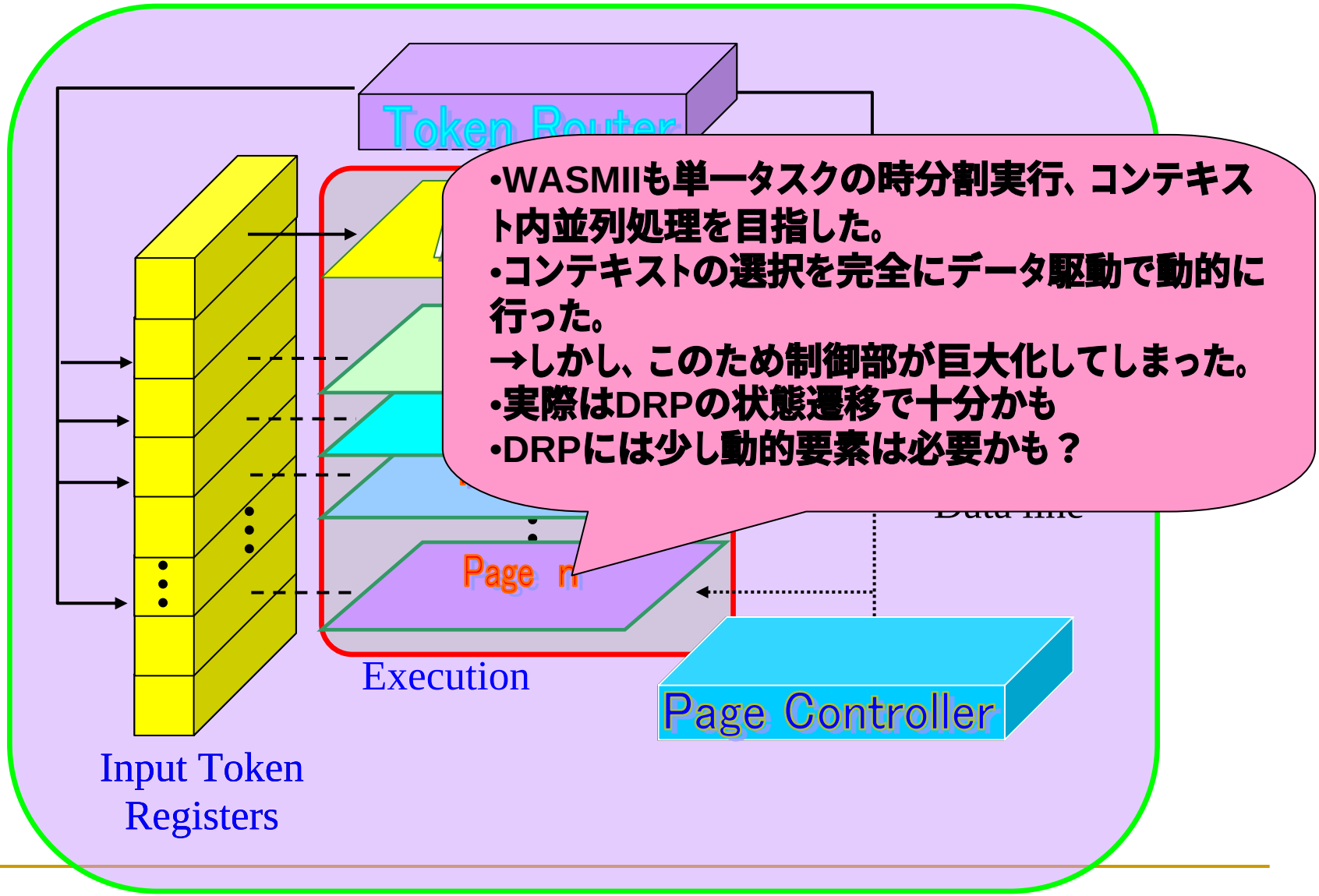
2.コンテキスト内の並列処理
3.コンテキスト内の時分割展開

BDLによる記述
DRPコンパイラが三次元的な展開を制御

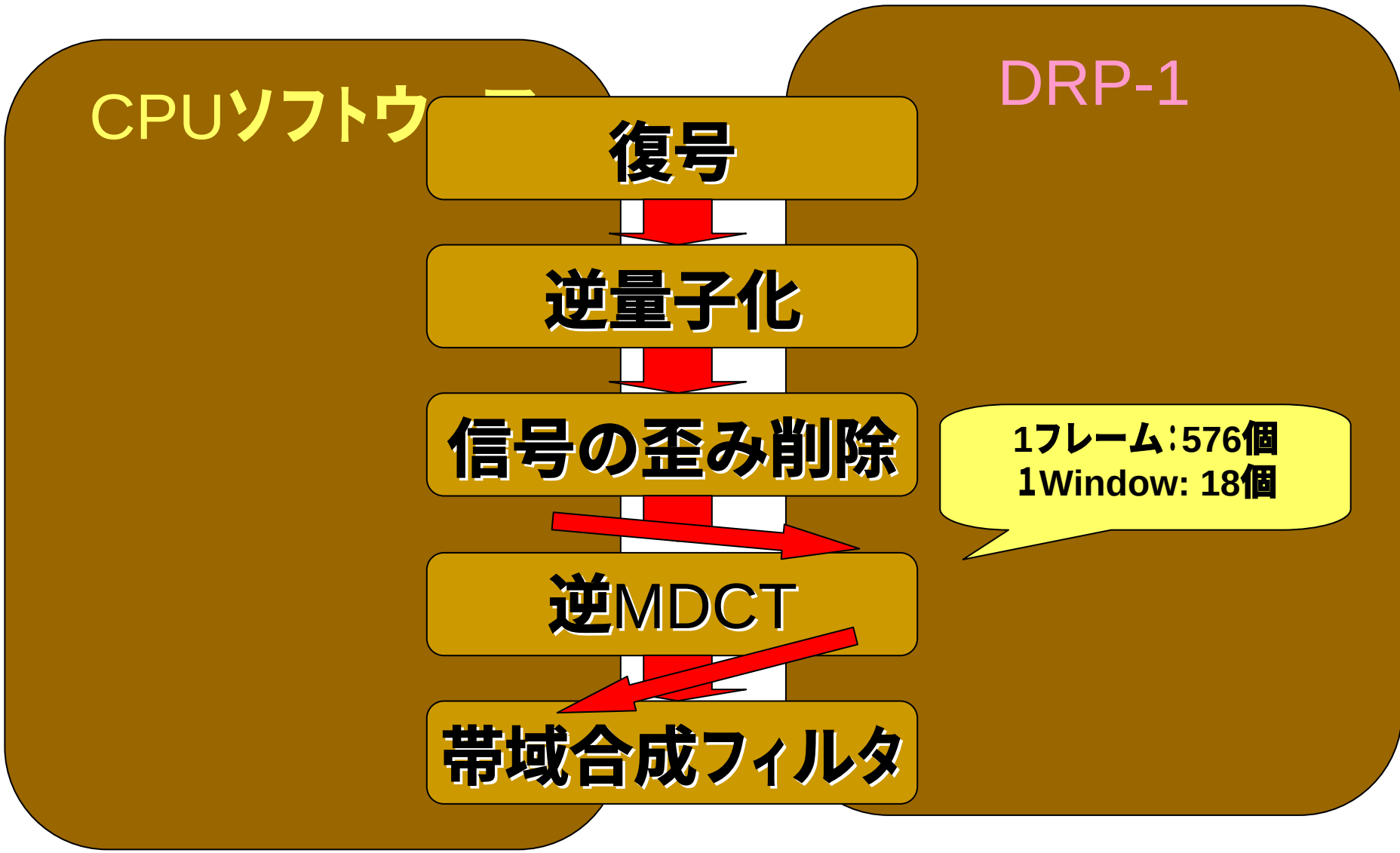
DRPはVLIWとFPGAの中間的な位置にある



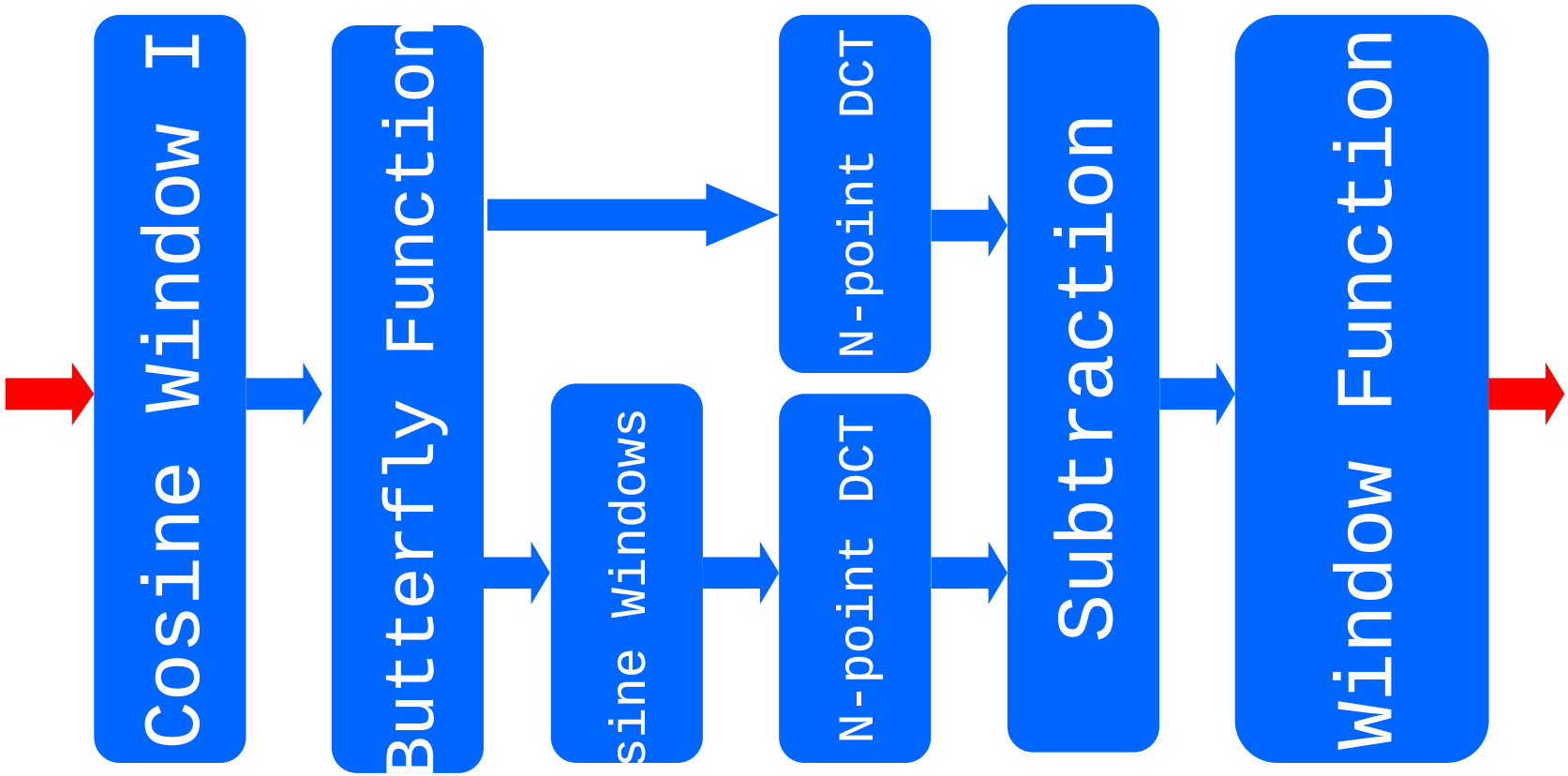
WASMIIとDRP


Token Router
Page n
Execution
Page Controller
Input Token
Registers
Control
•WASMIIも単一タスクの時分割実行、コンテキスト内並列処理を目指した。
•コンテキストの選択を完全にデータ駆動で動的に行った。
→しかし、このため制御部が巨大化してしまった。
•実際はDRPの状態遷移で十分かも
•DRPには少し動的要素は必要かも？

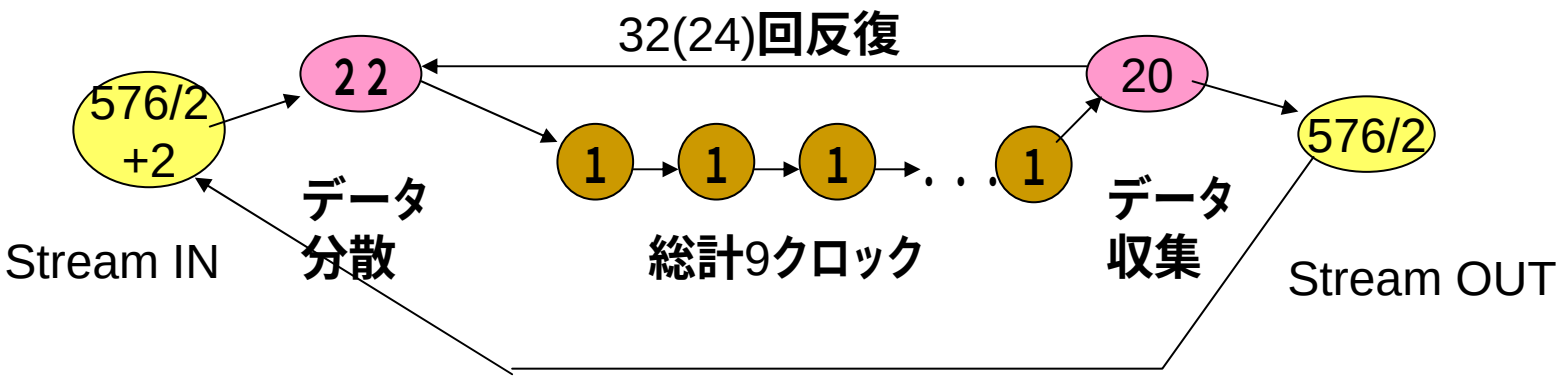
逆MDCTの実装例


CPUソフトウ
DRP-1
復号
逆量子化
信号の歪み削除
逆MDCT
帯域合成フィルタ
1フレーム：576個
1Window: 18個

DRP-1上での逆MDCT処理



DRP-1での処理



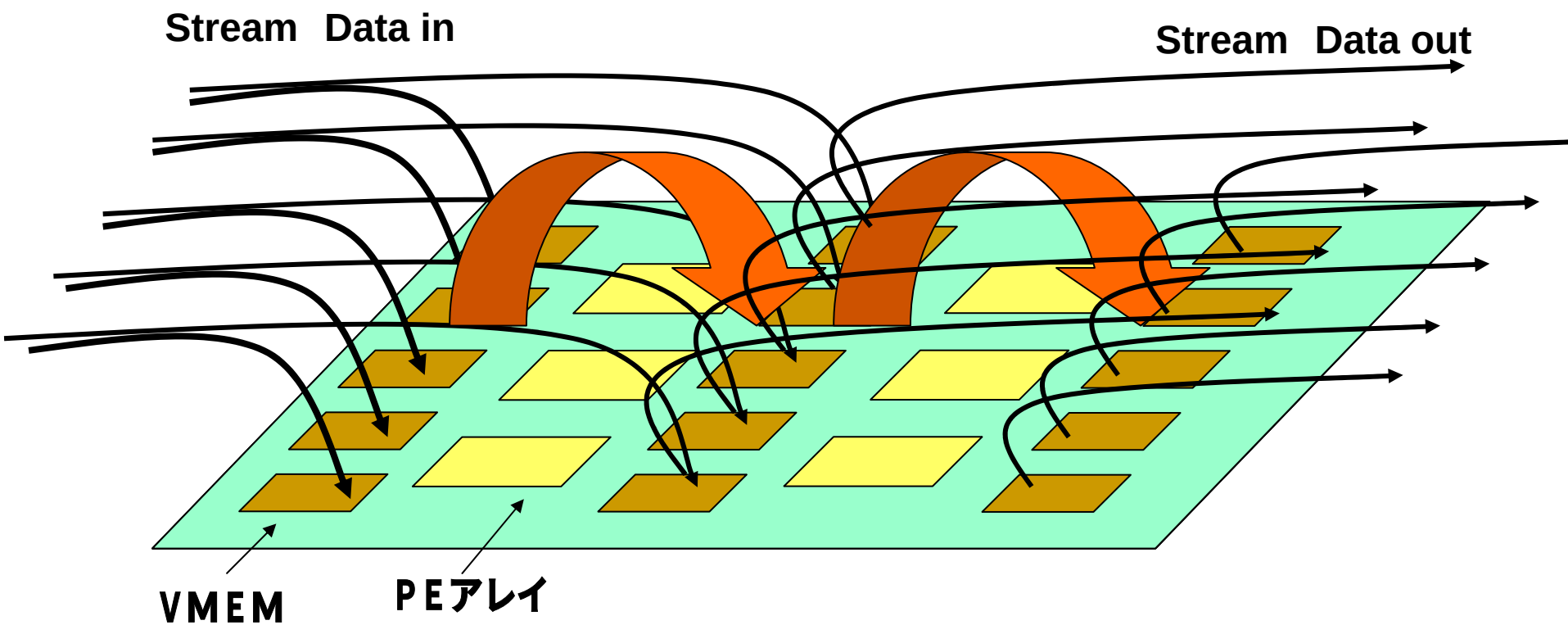
51clk　X　45ns = 2295ns/1回

収集、分散はオーバーラップ可能

入出力を含め13コンテキストを利用
22MHzで動作
演算部はPentium III 600MHzの約1,8倍の性能を達成

ＤＲＰ上での典型的なストリーム処理

異なったデータの集合にそれぞれの演算操作を施す
ＪＰＥＧ，ＭＰＥＧ，ＡＥＳ等様々な応用プログラムが実装されている



統合設計環境によるプログラミング

- **Ｃライクな言語**BDL(Behavior Description Language)**で記述**
  - **ＢＤＬはマクロをうまく使うと**gcc**でコンパイル可能**
  - **人手である程度スケジュールの制御可能**
- **統合設計環境**Musketeer
  - **機能合成 (ＤＲＰコンパイラ)**
  - **テクノロジ マッパ**
  - **配置配線ツール**
  - **デバッグ、設計解析用の表示ツール**
  - **各レベルのシミュレータとリンク**

統合設計環境による高級言語設計

- 一度使うとRTLエントリには戻れない
- 三次元的な展開制御
  - コンテキストの切り替え
  - コンテキスト内の並列処理
  - コンテキスト内の時分割処理
  - これらは機能合成ツールが、最適化してくれる（はず）
- 単一メモリアドレスに基づくC言語のプログラムがそのまま使えるわけではない
  - 分散メモリ、分散入力データ、ストリームの切り分けが性能向上の鍵
  - ここにプログラマは腕の見せ所がある！
- 高級言語設計は圧倒的に優れており、ダイナミックリコンフィギャラブルシステムの普及には不可欠

その他の設計事例

- **RTLエントリ・レイアウトツールを用いたもの**
  - Van del Pol/ Neural Network (FPGA/PLD Conf. 2003年1月)
  - **離散**Wavelet**変換(出口**：CPSY 2003年1月)
  - **逆**MDCT (**山田**：CoolChipsVI 2003年4月)
  - Viterbi**コーダ** (**金子**：Reconf I 2003年9月)
  - **ブロック暗号**RC6 (**長谷川**：Reconf III 2004年1月)
  - **αブレンダ** (**鈴木**：Reconf III 2004年1月)
  - **動的適応型スイッチ** (**天野**：FPL 2003年9月)
  - MIN**シミュレータ** (**天野**：Reconf III 2004年1月)
- Musketeer**利用例**
  - FFT (**黒瀧**：FPGA/PLD Conf. 2004年1月、FCCM　2004年4月)
  - **逆**MDCT (**天野**)

CPSY:電子情報通信学会コンピュータシステム研究会
Reconf:電子情報通信学会リコンフィギャラブルシステム研究会
FPL, COOL Chips:国際学会

FFTの性能比較

	DRP (リソース重視)	DRP (性能重視)	MIPS64	TI DSP
必要サイクル数	45056	11776	248047	83997
最大動作周波数	50MHz	33MHz	500MHz	225MHz
FFT**回数/秒**	1109	2802	2015	2678

これは1Tileしか使ってないことに注意！

アプリケーションの性能

アプリケーション	C数	DRPの動作周波数	性能
DWT	14	61MHz	Pentium III(600M)の2倍以上
逆MDCT	13	22MHz	Pentium III(600M)の約1.8倍
Viterbi	12	33MHz	Pentium IV 2.4Gの5倍
RC6	13	32MHz	MIPS64(500M)の6倍 TMS320C6713(225M)の22倍
αブレンダ	5	38MHz	Pentium IV(2.5G)の3倍 TMS32C6713(225M)の17倍

C数：コンテキスト数

紹介の流れ

- ダイナミックリコンフィギャラブルプロセッサ導入の動機
- 3つのポイント
- ダイナミックリコンフィギャラブルプロセッサ紹介
- DRPの世界
- おわりに
  - まとめ、問題点、Open Problem
  - 我田引水
  - お誘い

ダイナミックリコンフィギャラブルプロセッサのまとめ

- 専用ハードウェアに比べて性能面で匹敵（間に合うという点で）、面積効率で勝つ、消費電力はこれから、、
- 対象は、ストリーム処理を中心に広いがキラーアプリを確保して市場を押さえていない
  - Chameleonの失敗
  - ダイナミックリコンフィギャラブルプロセッサでなければならない、というのが少ない
- 本来、高級言語指向だが現在各社とも、開発ツールは整備中
- 多いライバル
  - 高性能組み込みプロセッサ
  - オンチップマルチプロセッサ
  - 目的用途別Configurable　Processor
  - DSP
  - 標準FPGA/CPLD
  - System On Chip
- しかし、大きな可能性を持つ
  - 新しいアーキテクチャ研究領域

Open Problems

- ストリームの切り分け、分散メモリへの展開手法に一般的な方法論はあるのか？
  - アプリケーション毎にプログラマが考えるのはしんどい
- Configurationはどの程度の時間で切り替わるべきなのか？
  - 毎クロック替わるのは、消費電力の点で明らかに不利
  - しかし、ずーっと同じConfigurationならば、動的再構成機能など必要ない
- Processing Elementの粒度は？
  - 本当に8-32bitの演算器レベルで良いのか？
  - Xilinxの特許逃れのためにアーキテクチャが歪んでいるのでは？
- データパスと制御のバランスは？
  - DRPのように完全に演算器データパスでは制御回路が組み難い
  - ACM,IPFlexなどヘテロな構成のノードバランスはどうするか？

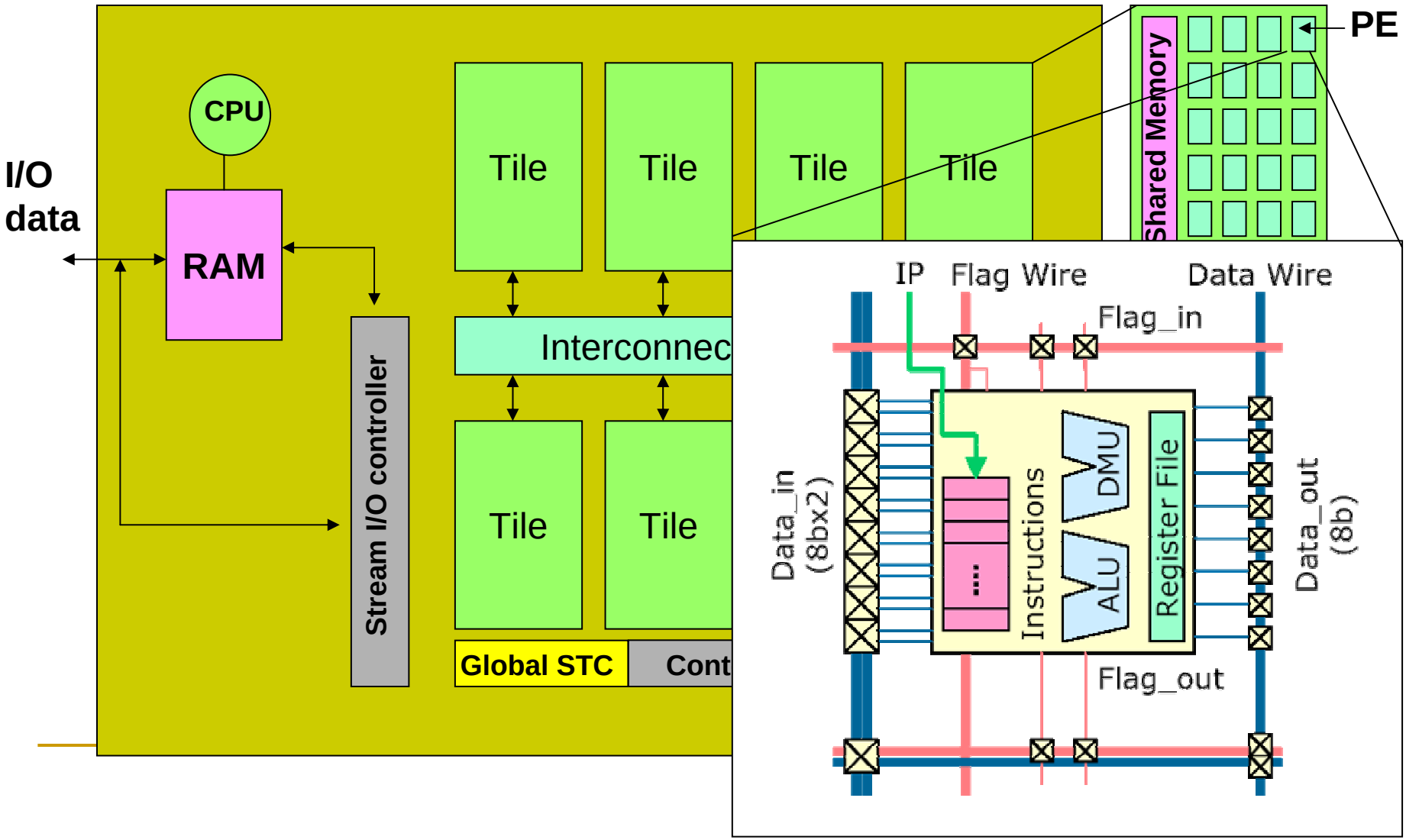
Reconfigurable Architectureの Big Problem

- **プログラム格納型は、**ISA (Instruction Set Architecture)**の形でソフトウェアとハードウェアを分離して発達した。**
- Reconfigurable Architecture**は、どこかのレベルで標準化して分離すべきなのか？**
  - **例えばストリームの入出力、処理手法を標準化**
  - **それとも分離していないことが本質なのだから、混沌をそのままにしておくべきか？**
- **何かとっかかりがないと、アプリケーション開発が非常に苦しい。しかし混沌に穴をあけることにより、死に至らしめるのかも、、、**

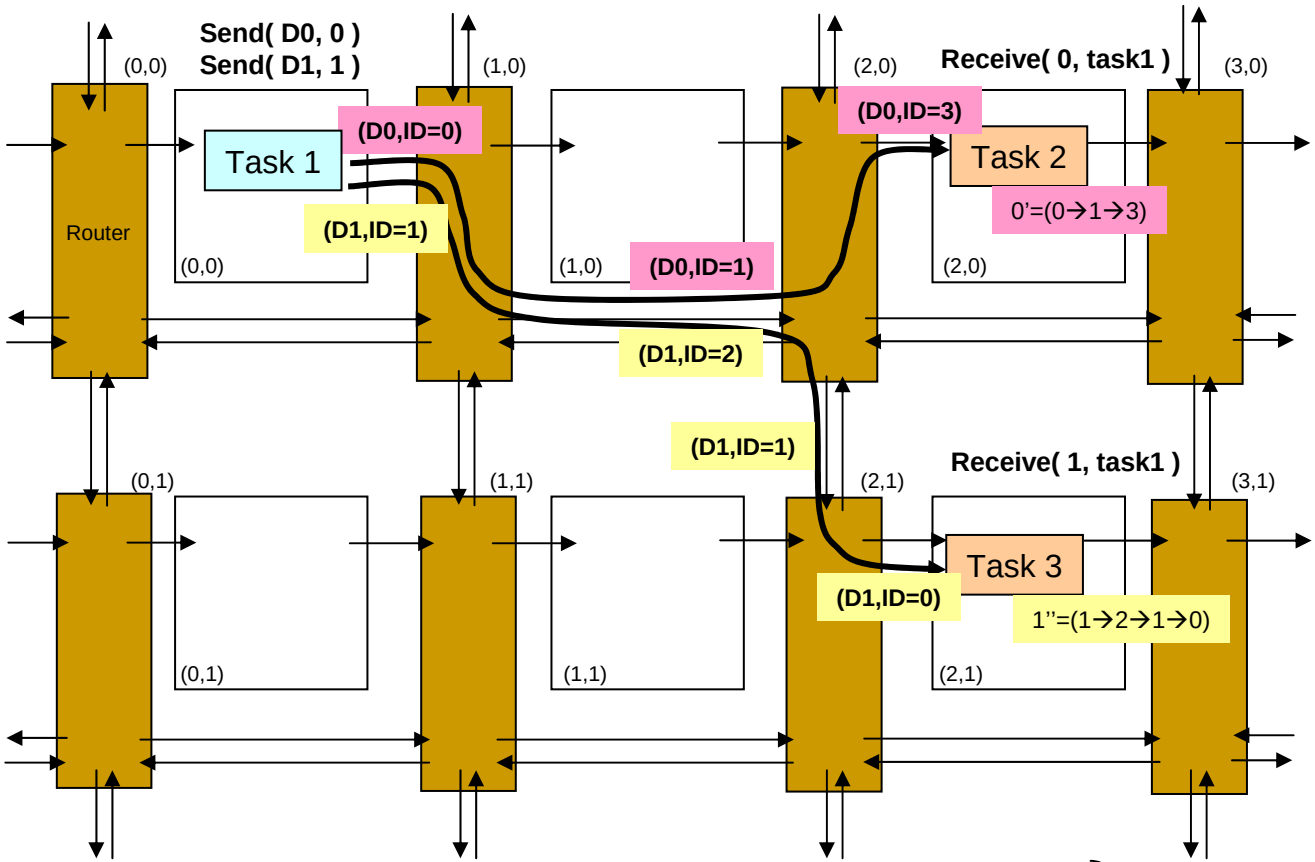
我田引水 (天野研プロジェクト)

- SARA (Stream processing architecture with Reconfigurable processor array)
  - Network-On-Chip: Black-bus
  - **仮想ハードウェア**
- DRP**および**SARA**でのアプリケーション開発**
  - **各種ストリーム処理**
  - MP3、Wavelet**変換**、JPEG2000、Viterbi coder、Alpha blender、**ロボットの音源分離処理等**
  - **動的適応型ハードウェア**
    - **全ての機能を実現する回路と一部の機能に限られるが、性能や消費電力の点で優れた回路を状況に応じて切り替えて用いる**

SARA (Stream processing Architecture with Reconfigurable processor Array)
新しいリコンフィギャラブルアーキテクチャ：DRPのアレイ



Tile間接続結合網Black-bus



- 各配線リンク上を通過できるパケット → ID種類分まで
- 配線リンク間をIDを変化させながら転送
- 受信ノードに到着時のIDから逆探索すれば、送信ノードの特
  → ルーティング経路が静的に定まることを利用
- IDはデータの識別子も兼ねる

プログラマブルスイッチとルータの中間的性質

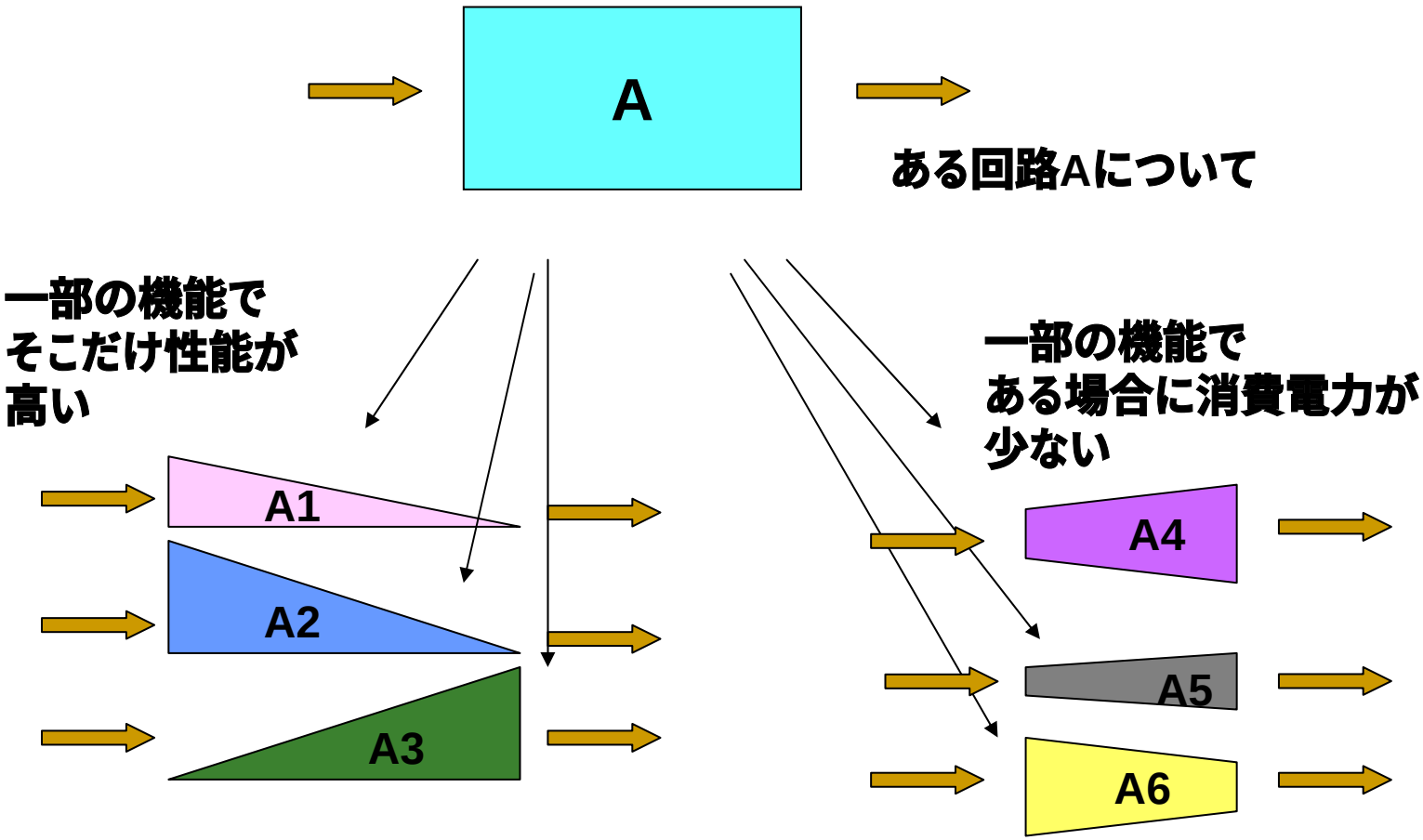
仮想ハードウェア (VHw)

- 動的にハードウェア構成を入れ替えることにより、システムサイズを超えるハードウェアを仮想的に実現すること
- 実現が容易なケースは既に実用的になっている

→ 関数オフローダ(NEC2004)

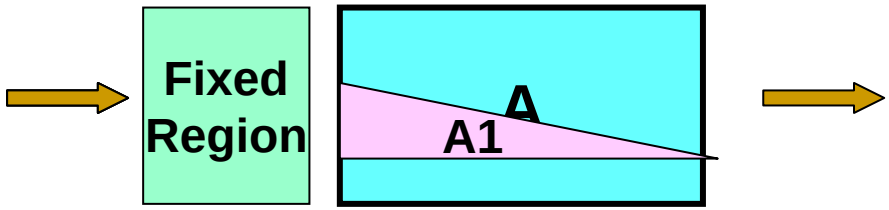
- 単一ジョブVHW
  - チップ内のコンテキストに収容できない巨大な単一ジョブを実行 → ループが複数収容できれば容易に実現可能
- 複数ジョブVHW
  - 大きなアプリケーション実現のため、連続した独立性の強いジョブを連続して実行
  - あるアプリケーションをサスペンドし、別のアプリケーションを実行

動的適応型Hw

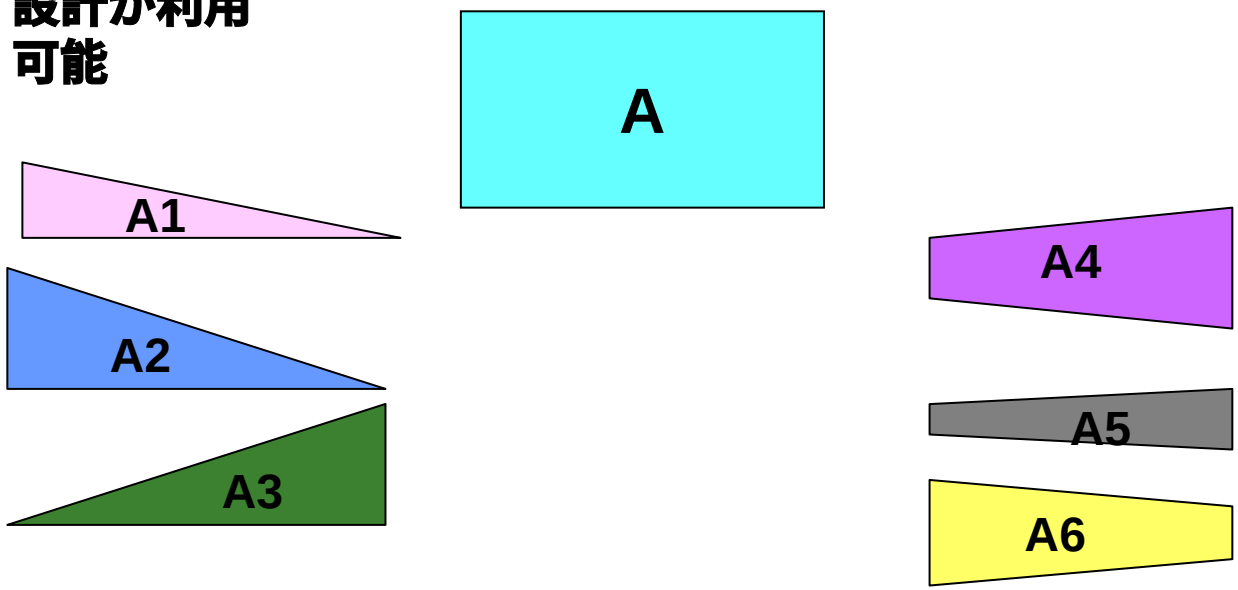


一部の機能で良ければ、数多く設計可能なはず
これを多数用意して切り替えよう

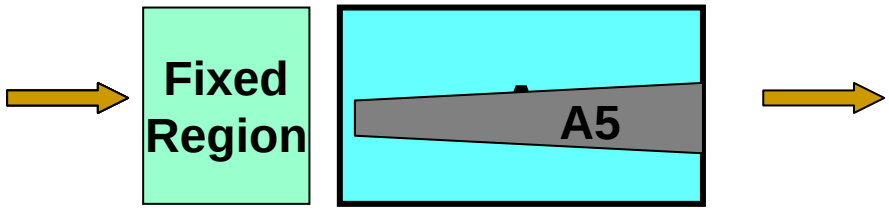
動的適応の原理



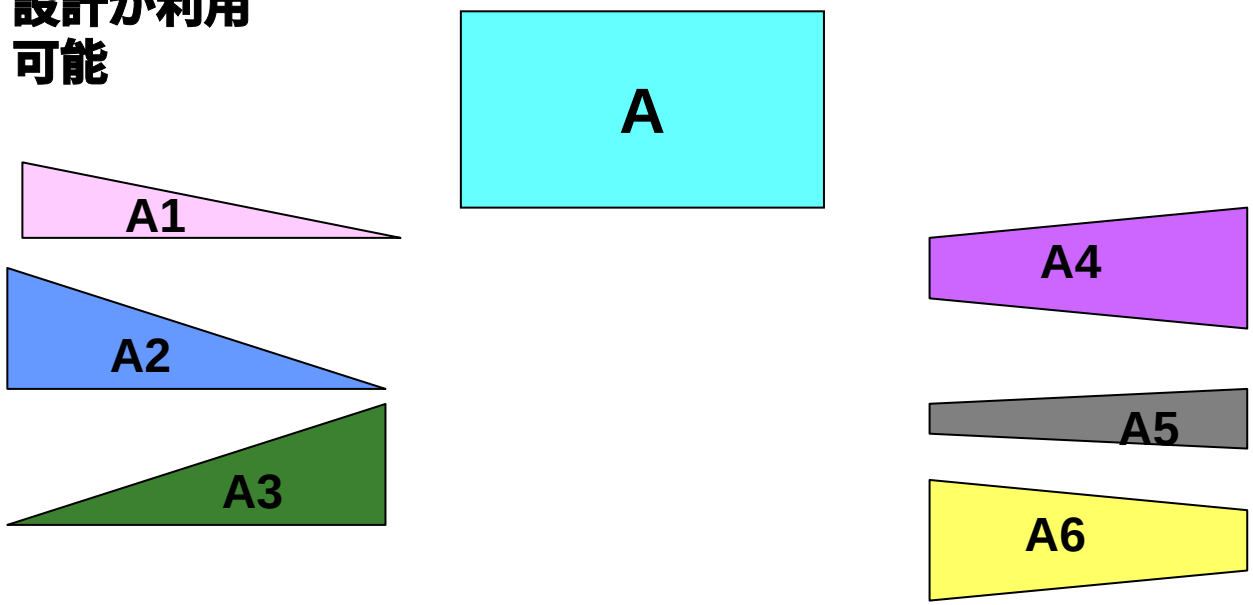
対応する機能的な設計が利用可能



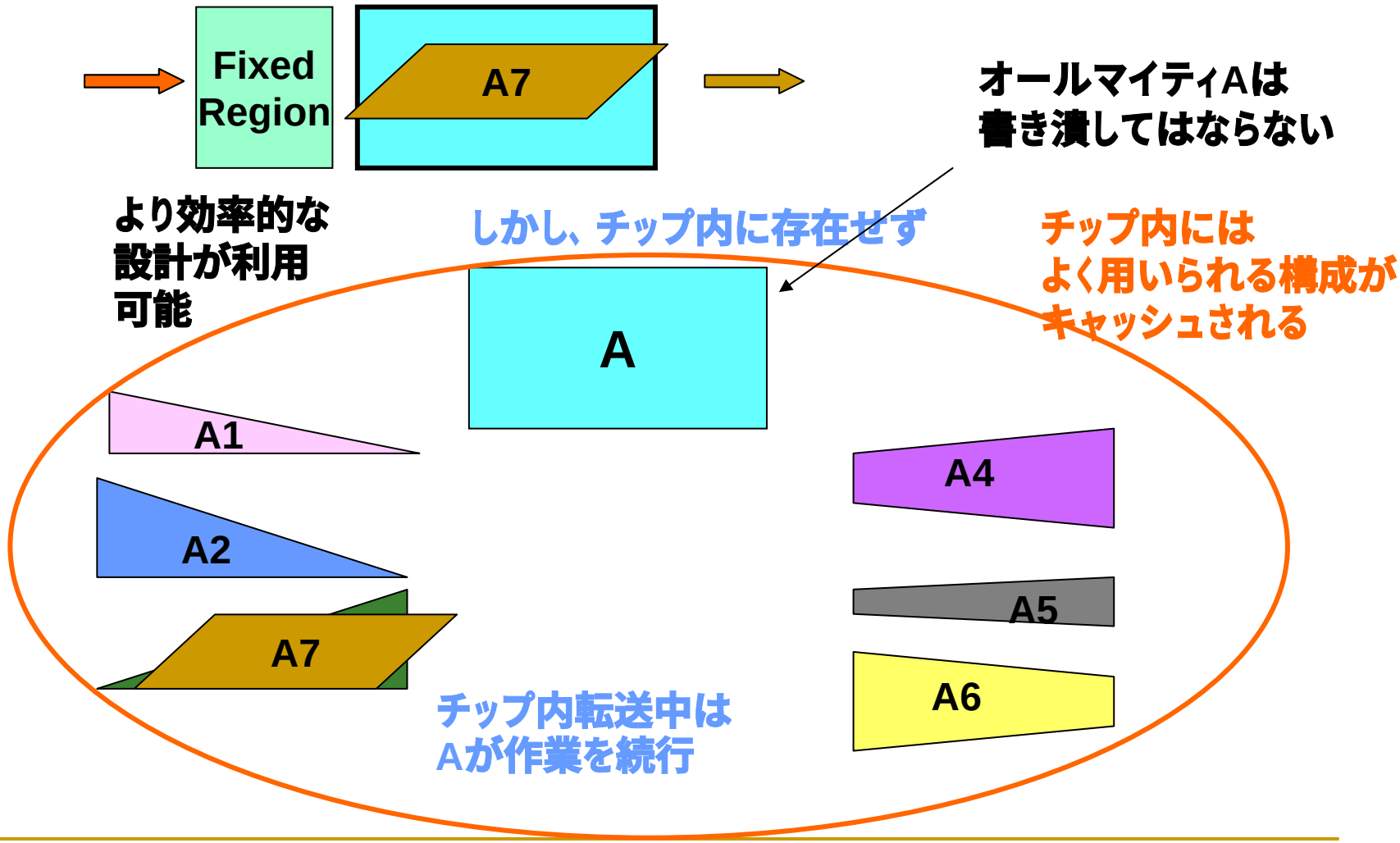
動的適応の原理



対応する機能的な設計が利用可能



動的適応の原理


Fixed Region
A7
オールマイティAは
書き潰してはならない
より効率的な
設計が利用
可能
しかし、チップ内に存在せず
チップ内には
よく用いられる構成が
キャッシュされる
A
A1
A2
A7
A4
A5
A6
チップ内転送中は
Aが作業を続行

ダイナミックリコンフィギャラブルプロセッサ研究へのお誘い

- ダイナミックリコンフィギャラブルプロセッサ/オンチップマルチプロセッサ境界例は、計算機アーキテクトの研究領域なのでは？
- なぜ、みんなこの領域に乗り出してくれないの？
  - 汎用プロセッサではないから
    - しかし、対象アプリは広いぞ
    - 一般的コンピュータ屋は若干の意識変革が必要
      - アルゴリズムは徹底的に最適化して組み込んでしまってよい
      - ストリーム処理は、応答時間が速ければ良いわけではない
      - OSは協調しているCPU上で走る
  - 怪しいから、
    - 怪しいだけに面白いところも多い
    - なんでもあり、だけに、コンピュータサイエンスの手法をきっちり行えば光る研究が可能
- University Programが走りつつある→ DRP, DAP/DNA-2