FPGAを用いた高性能計算の可能性とデータフロープログラミング

佐野　健太郎

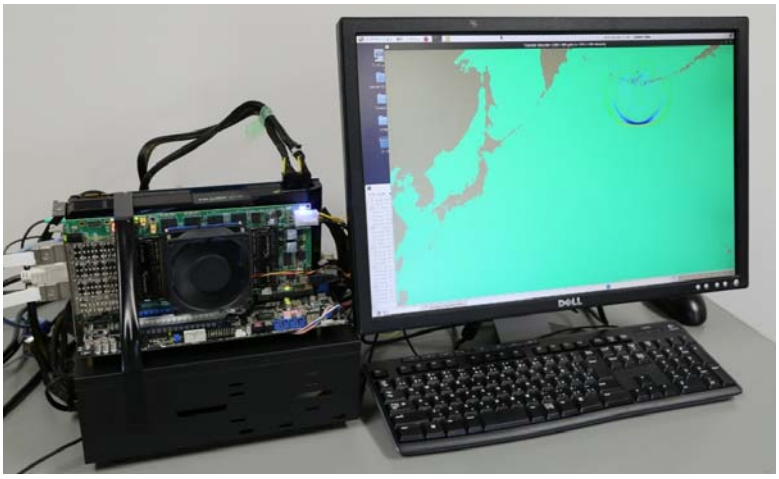
理化学研究所 計算科学研究機構 プロセッサ研究チーム

東北大学 大学院情報科学研究科

15 Dec, 2017

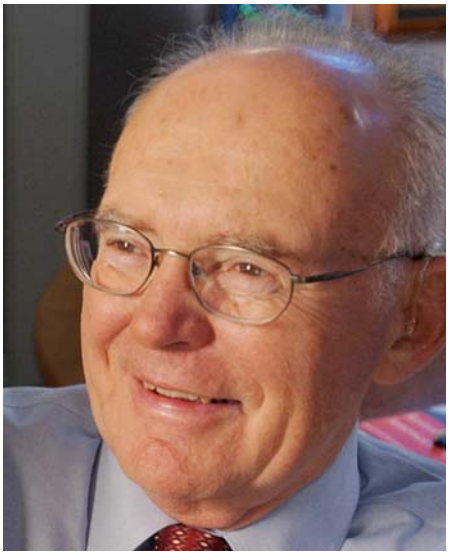
概要

- **マルチコアアーキテクチャの今後**
  - ✓ ポストムーア時代を生き延びる方法とは？
- **FPGAを用いた高性能計算**
  - ✓ 最近の話題
  - ✓ データフロー計算システム
  - ✓ 密結合FPGAクラスタ
- **データフローに基づくプログラミング**
- **事例**
- **まとめ**



ムーアの法則

微細化により 18~24ヶ月ごとに
チップ上のトランジスタは倍増



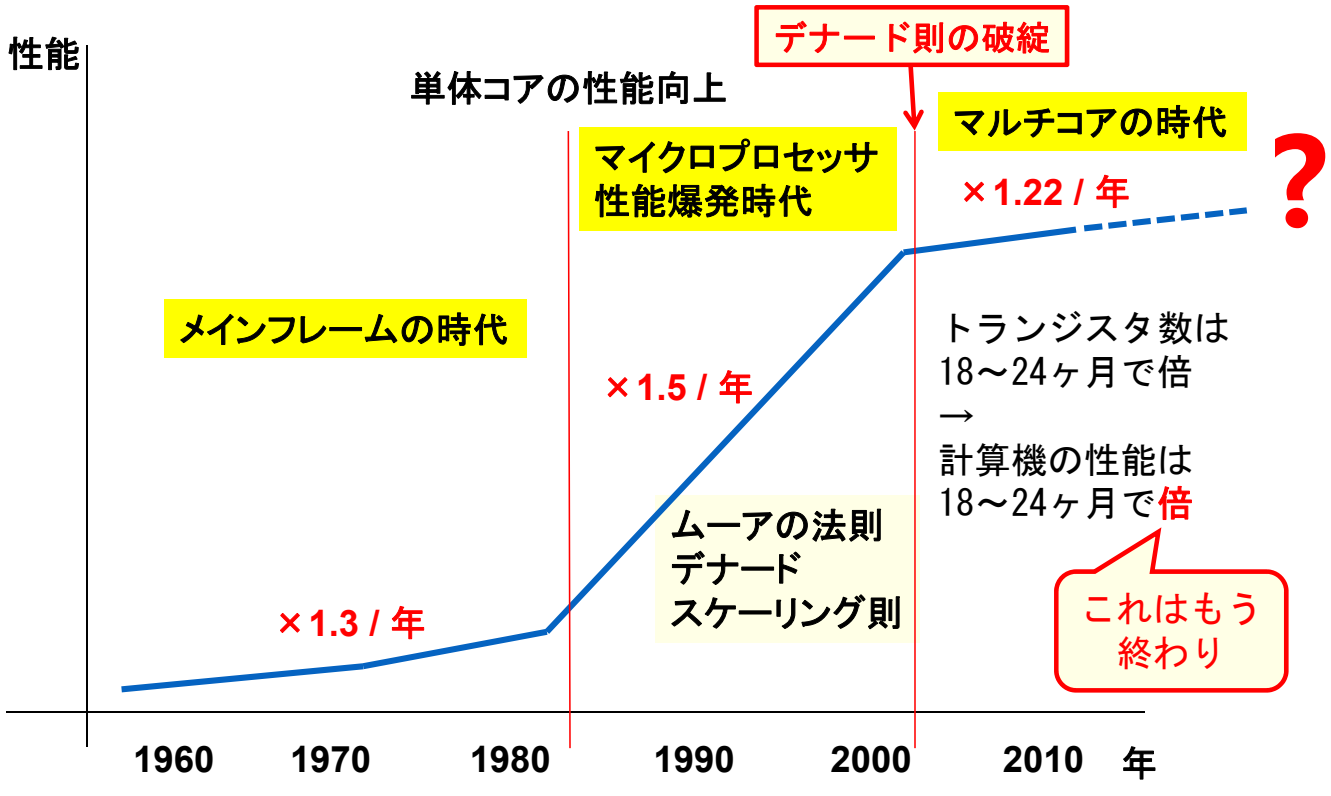
CMOS性能向上の鍵

デナードスケーリング則

トランジスタの寸法に比例して
電圧と電流は低減

4 Dec, 2017

マルチコアスケーリングはいつまで続くか？



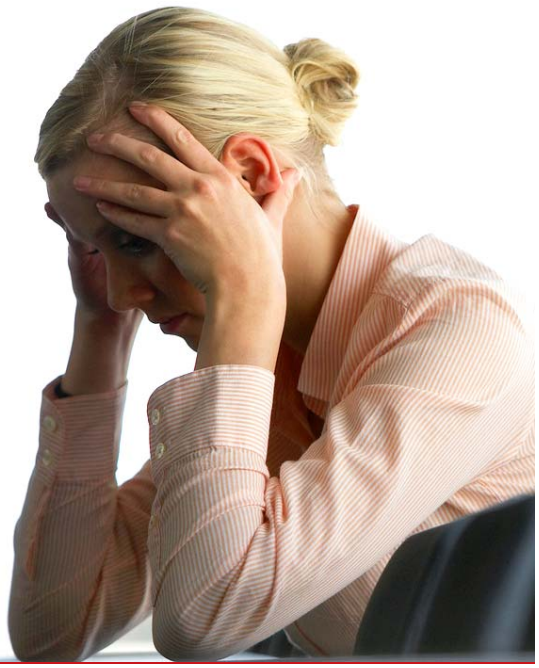
FIT2016 天野英晴「ポストムーア時代の計算プラットフォーム」パネルスライドより

マルチコアスケーリングの終焉

さらに高集積化が進み、コアが多数となったとしても

Q1. 消費電力を一定以下に抑えられるか？

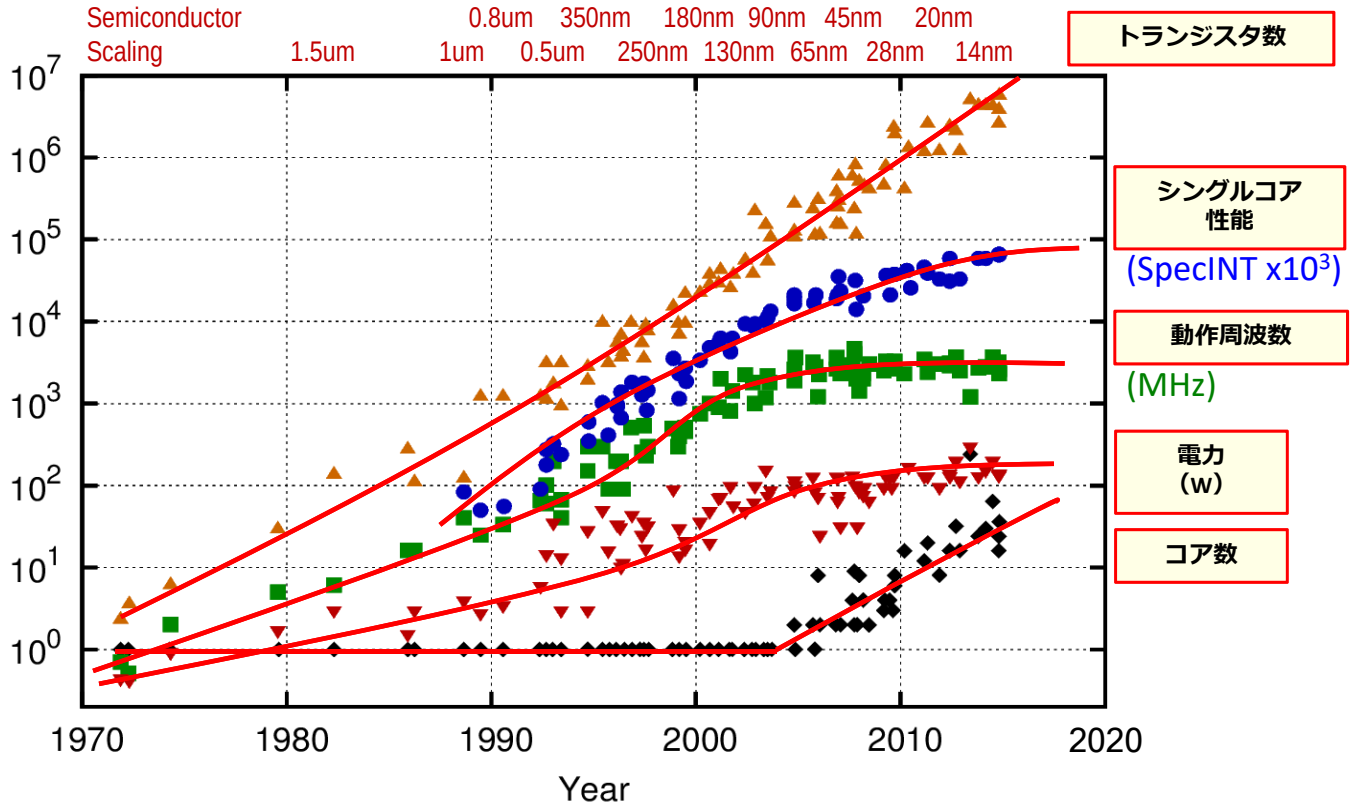
いいえ。　電力制約のために
一部のコアは稼働できなくなります



Q2. 全コアに対し十分な並列性があるか？

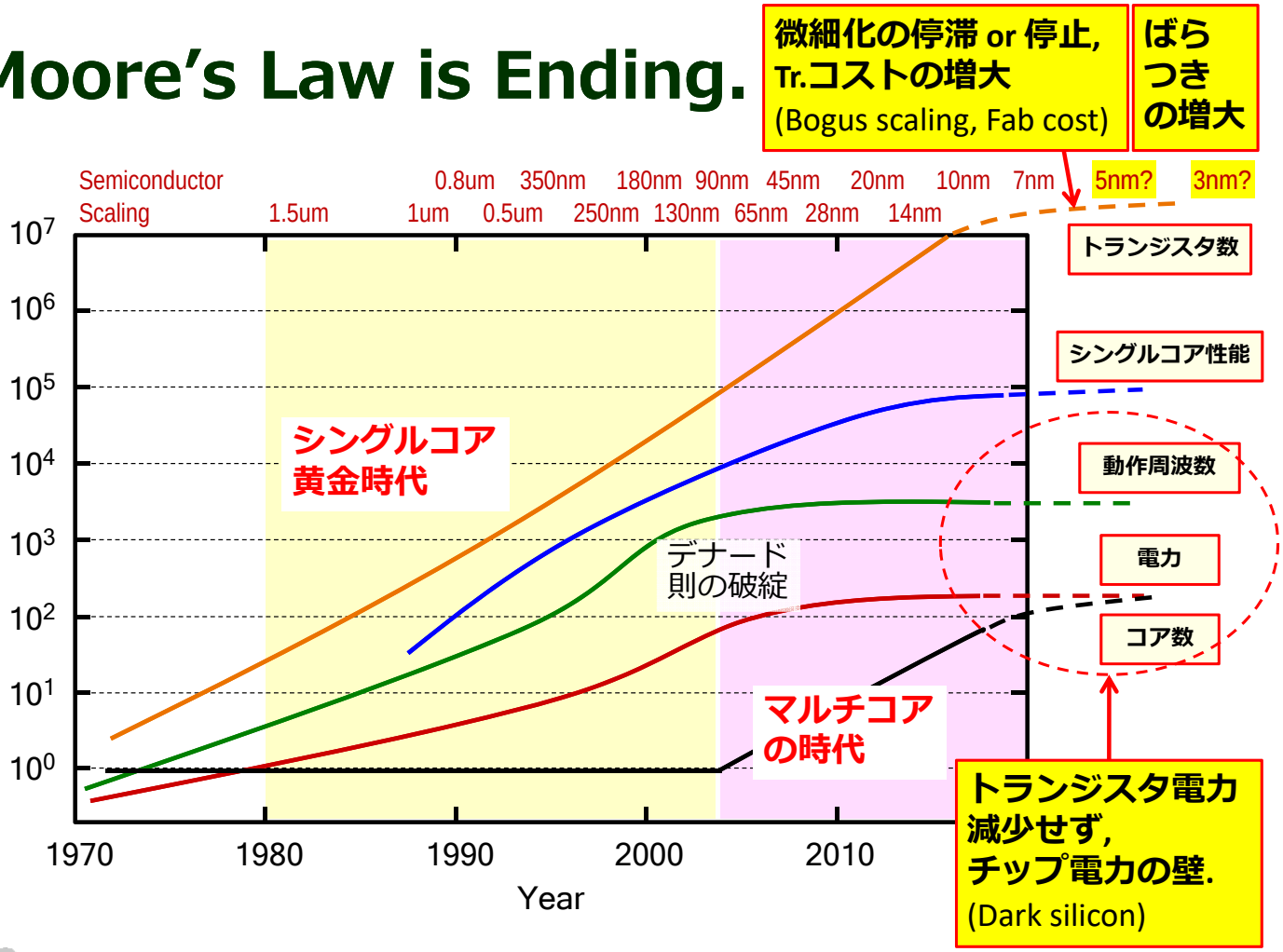
いいえ。　並列性不足のために
一部のコアは性能に寄与しません

Microprocessor Trend in 40 Years



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten,
New plot and data collected for 2010-2015 by K. Rupp, and Trend lines drawn by K.Sano

Moore's Law is Ending.



ポストムーアを生き延びる

限界を迎える技術

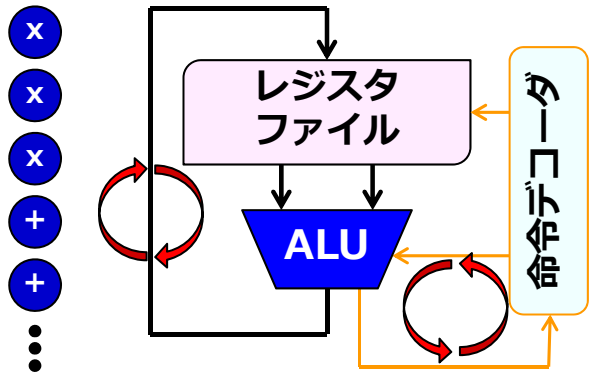
- ✓ 単一チップ上の**集積度**向上 △
- ✓ ダークシリコン ×
- ✓ 信号伝搬やメモリ参照の**遅延** ×

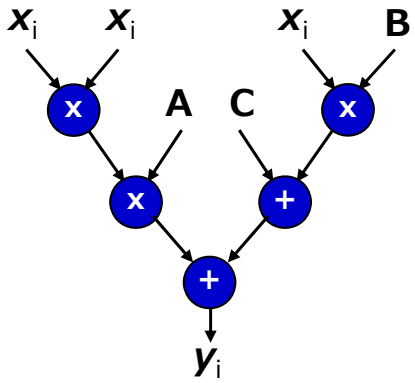
発展が期待できる技術

- ✓ パッケージ・システムレベルの集積 （多チップ） ○
- ✓ チップ内外のI/O帯域 ○
  （2.5D, 3D, 光電子技術, …）

多数のチップにより与えられた**膨大な数のトランジスタ**を
増大する信号遅延の下でも**効率良く使う**アーキテクチャが必要

ノイマン型 v.s. データフロー





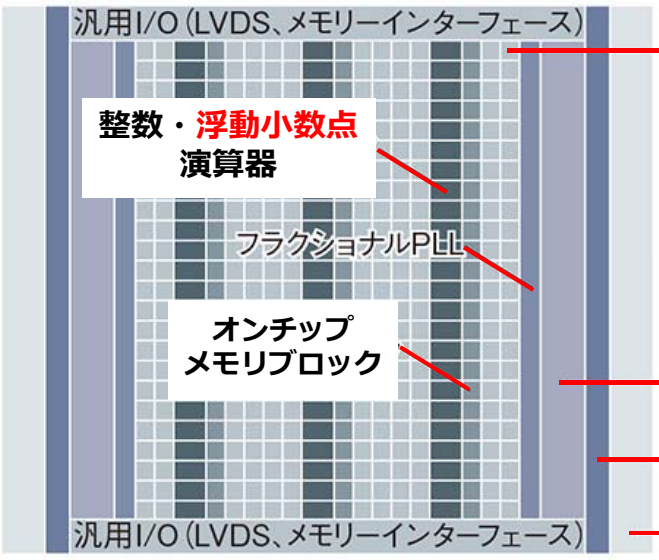
マイクロプロセッサ （ノイマン型）

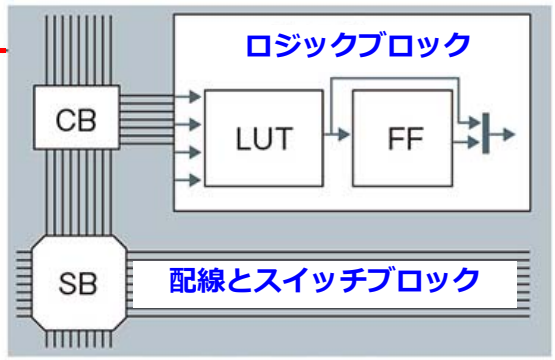
- ✓ メモリ更新サイクル
- ✓ 制御サイクル （実行中の決定）
- ✓ **逐次実行 ⇒ 遅延の影響大**
- ✓ マルチコア：メモリを介した非効率なデータ移動や同期

データフロー （非ノイマン型）

- ✓ サイクル ⇒ フロー
- ✓ 制御 ⇒ 構造 （実行前の決定）
- ✓ 演算とデータ移動の融合
- ✓ **遅延を許容 （パイプライン）**
  規則的な大量データ処理向き

FPGA ： カスタムハードウェアを実現





High-speed I/O
(PCI-Express, 10G Ethernet
Interlaken, serial transceiver)
Memory I/F (DDR4, HBM, etc.)

日経エレクトロニクス2016年月号

ロジックというよりは 演算器・メモリ・高速I/Oの巨大アレイ

浮動小数点に対応、動作周波数はまもなく **1GHz近く** へ

目標・チャレンジ

来るべきポストムーア時代の高性能
計算機アーキテクチャ・並列計算モデル

ポストムーア時代

- ✓ メニーコアのみでは非効率、しかし膨大な数のチップ
- ✓ 「カスタム化の導入、大域的同期を避けた並列計算」が有望

チャレンジとアプローチ

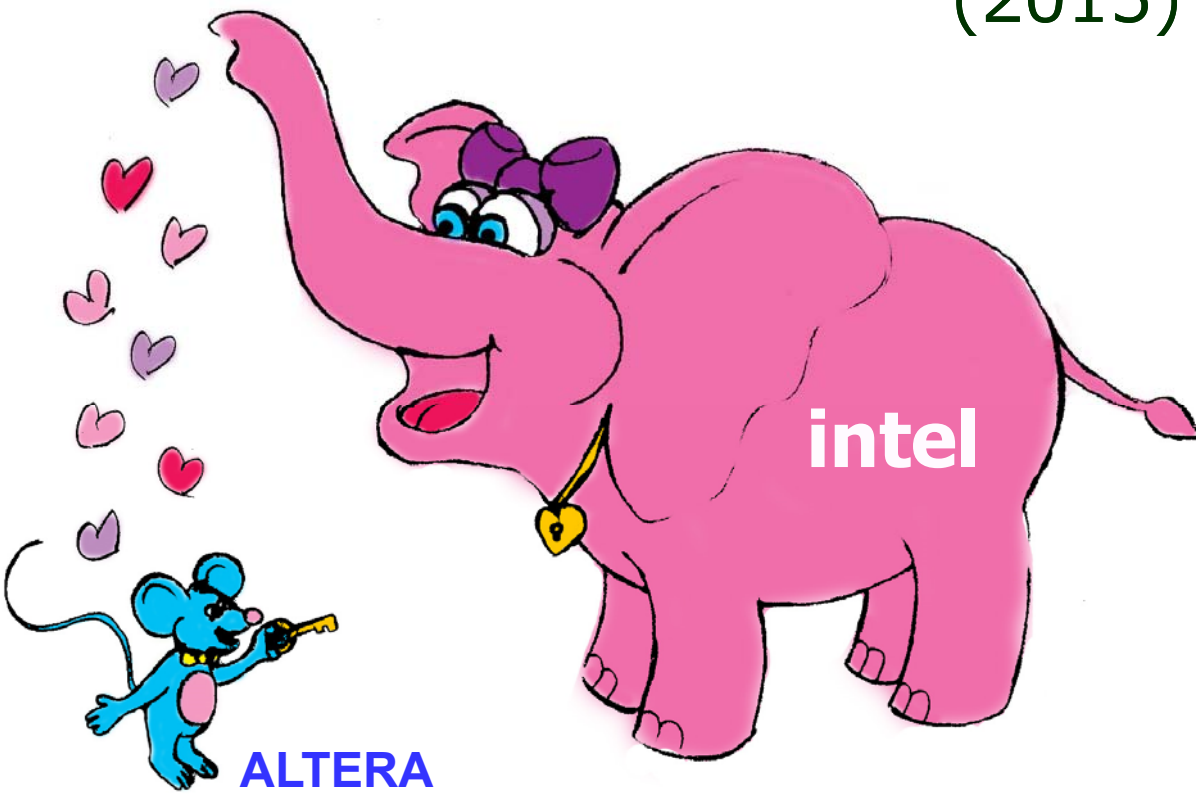
- ✓ 適したアーキテクチャは？　専用＆汎用 ⇒ FPGA
  プログラミングモデルは？　局所的な同期 ⇒ データフロー

研究課題

- ✓ FPGA・カスタムハードウェアを利用した計算の高速化
- ✓ データフロープログラミングモデルと動的実行の処理系

FPGAに関する最近の話題

$16.7B Acquisition of ALTERA by Intel (2015)



FPGAのデータセンター利用の広がり

Microsoft
Search engine w/ FPGAs

J.P.Morgan
High-speed trading w/ FPGAs

Xeon & FPGA with QPI
intel

IBM
Power & FPGA with CAPI

Baidu 百度
Deep learning w/ FPGAs

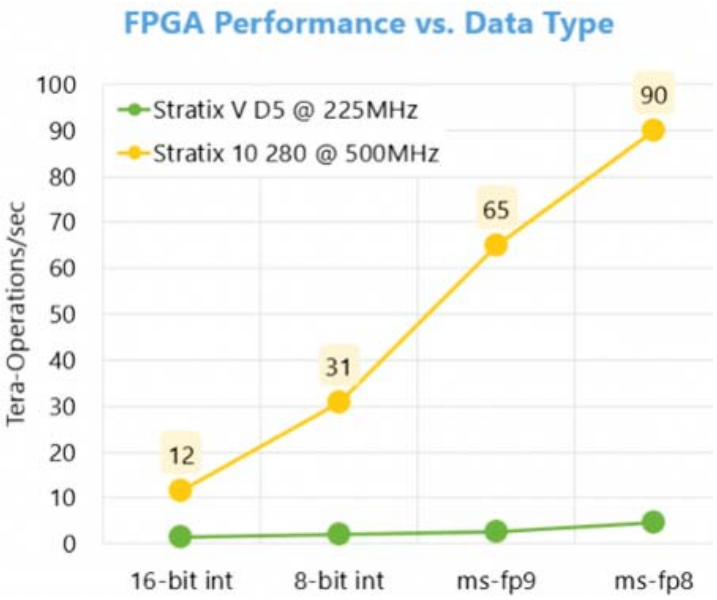
dwango
Data centers w/ FPGAs

FPGAs are now serving applications for end users.

FPGA Wins Versus Google TPUs for AI

Microsoft's Project Brainwave

- ✓ **DNN processor (DPU)** that is programmed/synthesized on 14nm Stratix10 **FPGAs** for deep learning in could services
- ✓ Accelerators linked across high bandwidth, low-latency fabric to **dynamically allocate** the resources to optimize their utilization.
- ✓ Significant performance advantages **over Google's TensorFow (ASIC)**



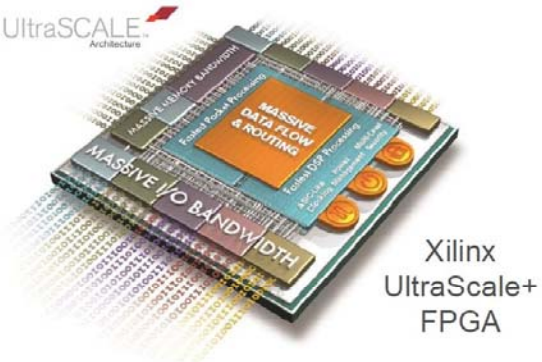
[Microsoft HotChips2017]

https://www-forbes-com.cdn.ampproject.org/c/s/www.forbes.com/sites/moorinsights/2017/08/28/microsoft-fpga-wins-versus-google-tpus-for-ai/amp/

FPGAs in AWS (Amazon Web Services)

GPU and FPGA for Accelerated Computing

NVIDIA GPU



Xilinx UltraScale+ FPGA

P2: GPU-accelerated computing

- Enabling a high degree of parallelism – each GPU has thousands of cores
- Consistent, well documented set of APIs (CUDA, OpenACC, OpenCL)
- Supported by a wide variety of ISVs and open source frameworks

F1: FPGA-accelerated computing

- Massively parallel – each FPGA includes millions of parallel system logic cells
- Flexible – no fixed instruction set, can implement wide or narrow datapaths
- Programmable using available, cloud-based FPGA development tools

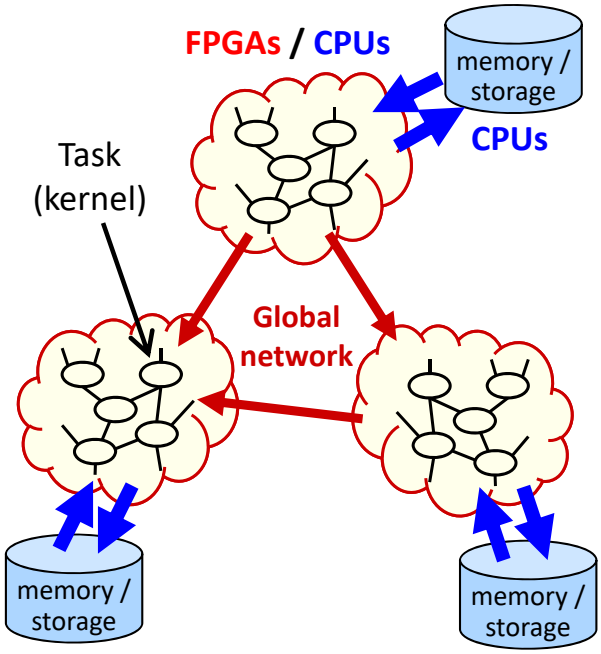
[Amazon HotChips2017]

FPGAのクラウド利用・サービスが進む

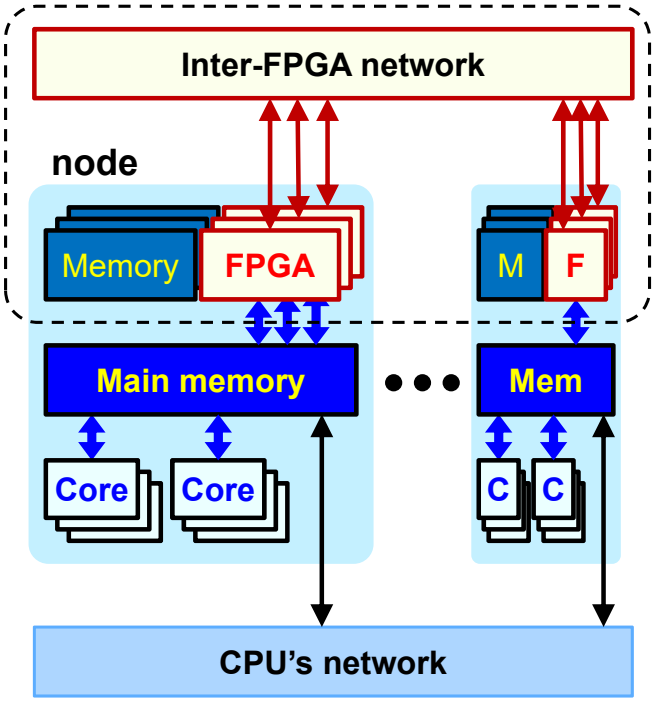
FPGAによる高性能データフロー計算システム

目標とする計算モデルとアーキテクチャ

Allocation, Scheduling > **automated**
Synchronization > **localized**



大域データフロー実行モデル



FPGA拡張部を有する計算ノード

研究開発の進捗

密結合FPGAクラスタの試作

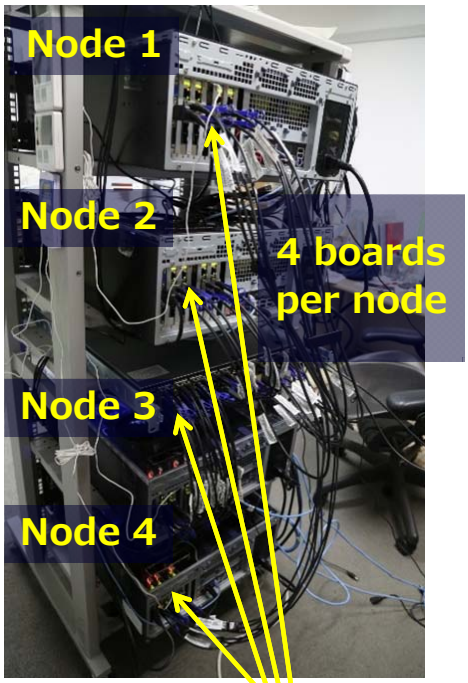
- ✓ システムオンチップデザイン
- ✓ ソフトウェアスタック（driver, API）
- ✓ データフローコンパイラ
- ✓ アプリによる性能評価（津波, 流体）

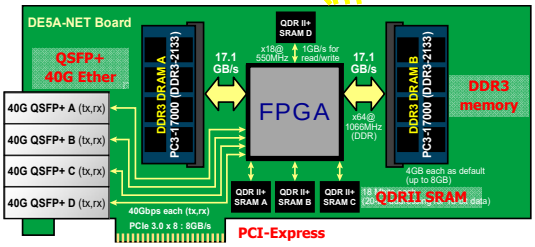
複数FPGAによるスケーラブル計算

- ✓ FPGAパイプライン
- ✓ N-Body（ストリームレジスタファイル）
- ✓ 流体計算（帯域圧縮）
- ✓ 2D トーラス網

データフローコンパイラの改良

- ✓ グラフパッキングの自動探索



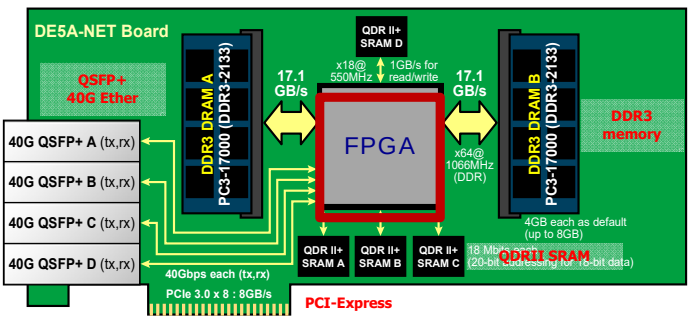


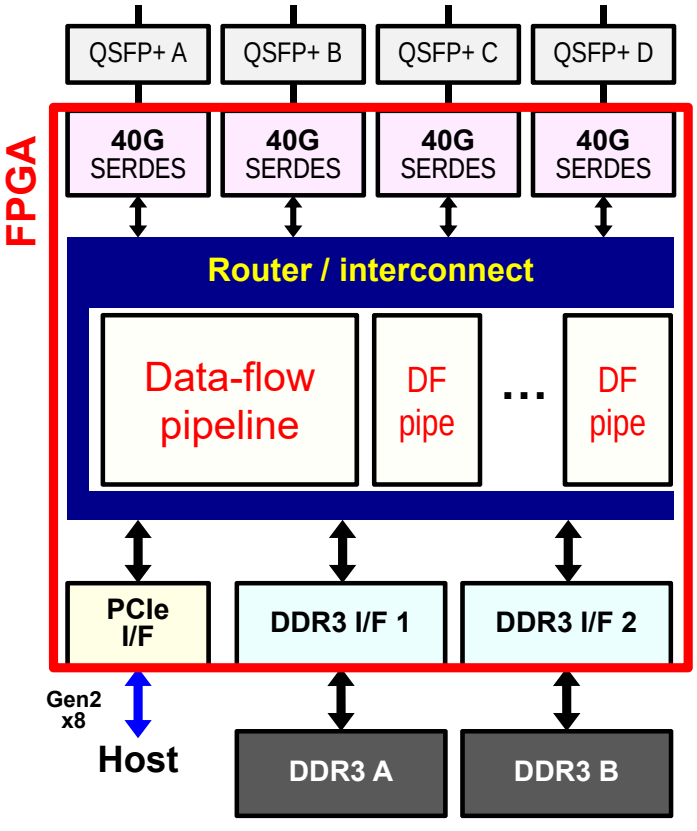
Arria10 FPGA
(20nm, floating-point DSPs)

FPGA Shell

不変のハードウェア部

- ✓ **可変部**：ユーザコア
  - ➢ データフローコンパイラで生成
- ✓ **アプリに共通の機能**を提供
  - ➢ メモリ I/F, PCIe I/F, NW I/F
    オンチップインタコネクト
- ✓ **ホスト側**のシステムソフト
  - ➢ ドライバ, API

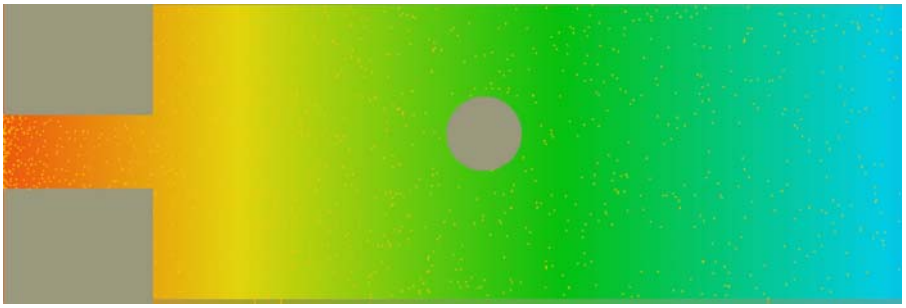




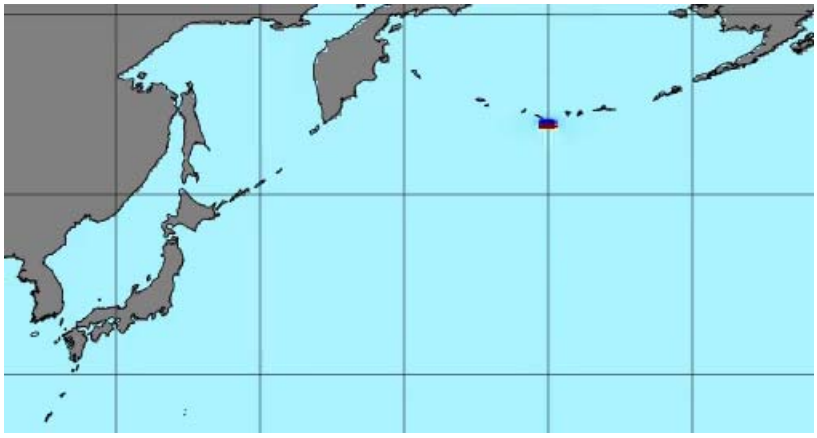
FPGA Shellのブロック図

アプリ

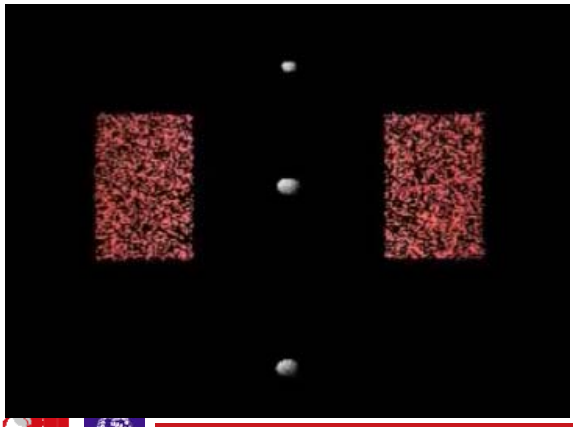
- **ステンシル計算**
  - ✓ LBM, 津波シミュ
- **多体問題**
- **今後の計画**
  - ✓ Deep learning
  - ✓ Spiking neural network



Fluid dynamics simulation (LBM)



Tsunami simulation



N-Body

データフローに基づく プログラミング

データフローに基づく ストリーム計算回路コンパイラ SPGen

既存の高位合成（HLS）コンパイラの問題

いわゆるHLSコンパイラ（Vivado-HLS, CWB）	SW言語でHW（IP core）を記述 計算を即実行できない（システムは別）
システムHLSコンパイラ（OpenCLなど）	システム全体を合成し、計算が即実行可 SWモデルに限定、最適化が困難

いずれにしても、**複数FPGAをプログラムできない！**

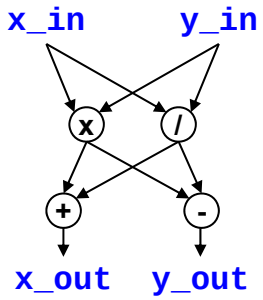
本研究：Stream Processor Generator （SPGen）

- ✓ データフロー計算モデルに特化した高抽象度言語
- ✓ 数式のみの記述で計算を実行可
- ✓ データフローグラフの形状・HW構造を制御可、最適化が容易
- ✓ 将来は、密結合された複数FPGA上のデータフローをプログラム

Stream Processing Description (SPD)

```
Name      PE;     ### Define pipeline "PE"
Main_In   {in:: x_in,  y_in};
Main_Out  {out::x_out, y_out};

EQU  eq1,  t1    = x_in * y_in;
EQU  eq2,  t2    = x_in / y_in;
EQU  eq3,  x_out = t1 + t2;
EQU  eq3,  y_out = t1 - t2;
```

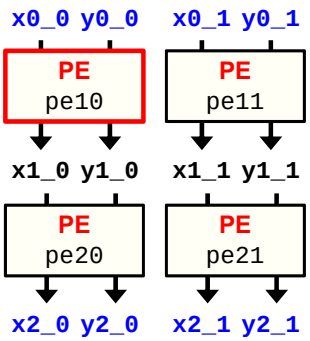
数式記述



計算コア（DFG）

```
Name      Core;         ### Define IP core "Core"
Main_In   {in:: x0_0, x0_1, y0_0, y0_1};
Main_Out  {out::x2_0, x2_1, y2_0, y2_1};

### Description of parallel pipelines for t=0
HDL pe10, 123, (x1_0, y1_0) = PE(x0_0, y0_0);
HDL pe11, 123, (x1_1, y1_1) = PE(x0_1, y0_1);

### Description of parallel pipelines for t=1
HDL pe20, 123, (x2_0, y2_0) = PE(x1_0, y1_0);
HDL pe21, 123, (x2_1, y2_1) = PE(x1_1, y1_1);
```

モジュール呼出し



階層的ハードウェア構造

SPGenツールチェイン

START

SPD ファイル

DFG 最適化ツール
ReAIEx（開発中）
- FP-DSP の最適割当て
- 演算ノードクラスタリング

NEW!

最適化済み
SPD ファイル

FINISH

FPGAコンパイラ
- HW のコンパイル
- ビットストリーム生成

SPGen
- フロー制御ラッパー生成
- パイプライン同士の接続
- スケジューリング

FPGen
- 式からパイプライン生成
- 多入力・多出力
- FP-DSP 利用可

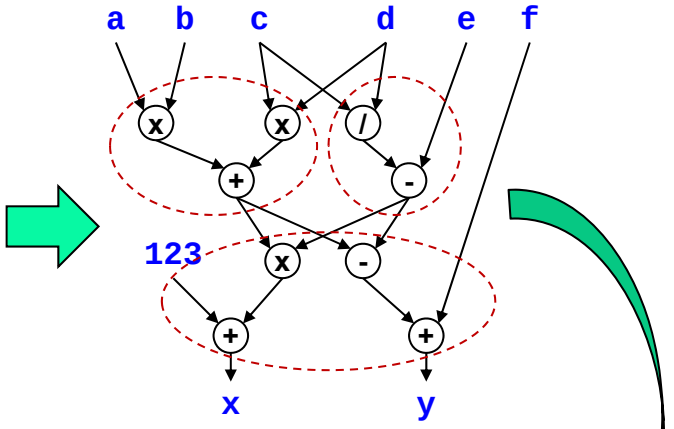
FloPoCo
- 式からパイプライン生成
- 多入力・単出力
- FP-DSP 利用不可

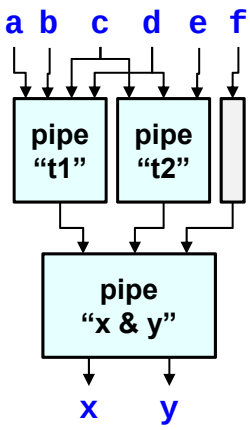
データフロー コンパイルの流れ

DFGのHW資源への最適マッピングを探索

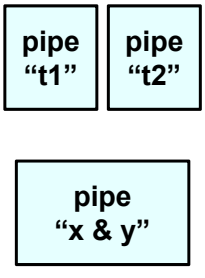
```
Main_In   {in::  a, b, c, d, e, f};
Main_Out  {out:: x, y};

EQU  eq1,  t1  = a*b + c*d;
EQU  eq2,  t2  = c/d – e;
EQU  eq3,  x   = t1*t2 + 123;
EQU  eq4,  y   = t1 - t2 + f;
```

SPD (Stream Processing Description)

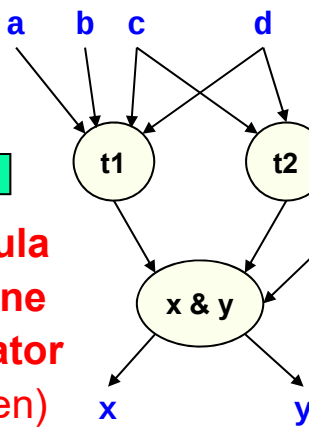




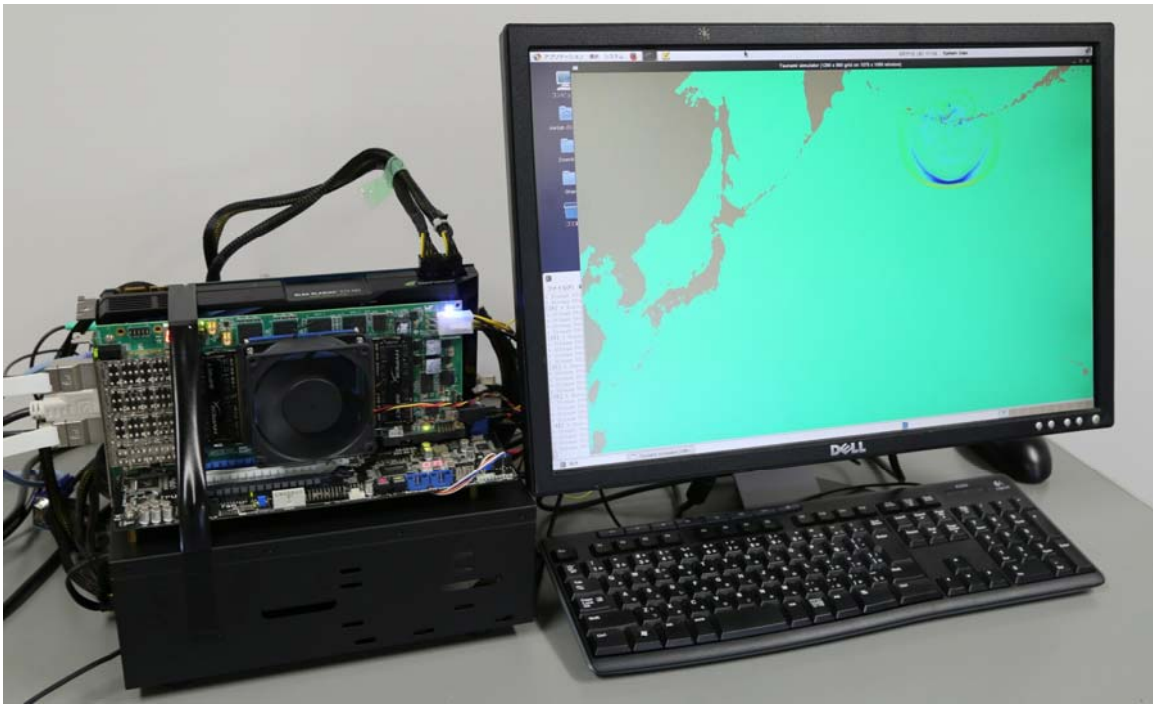
Stream processor generator (SPGen)



Formula pipeline generator (FPGen)



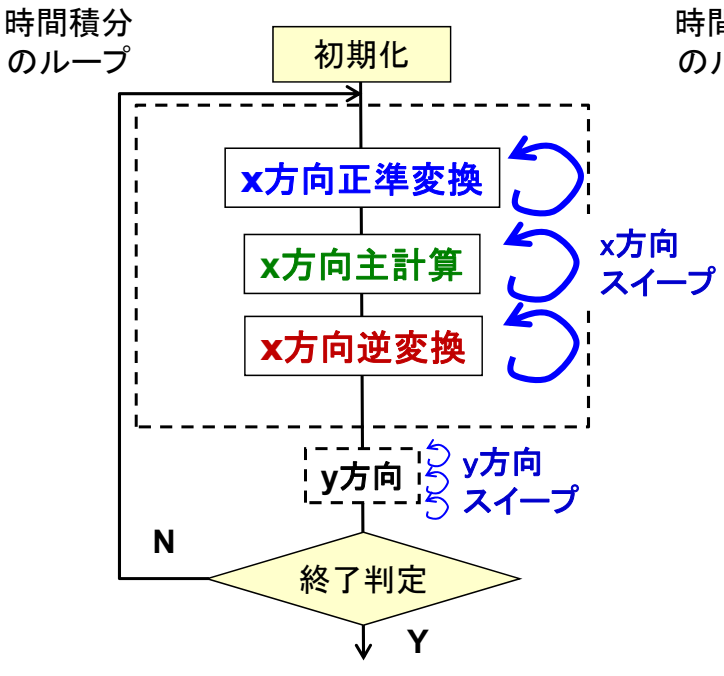
ReAIEx (clustering, resource mapping)



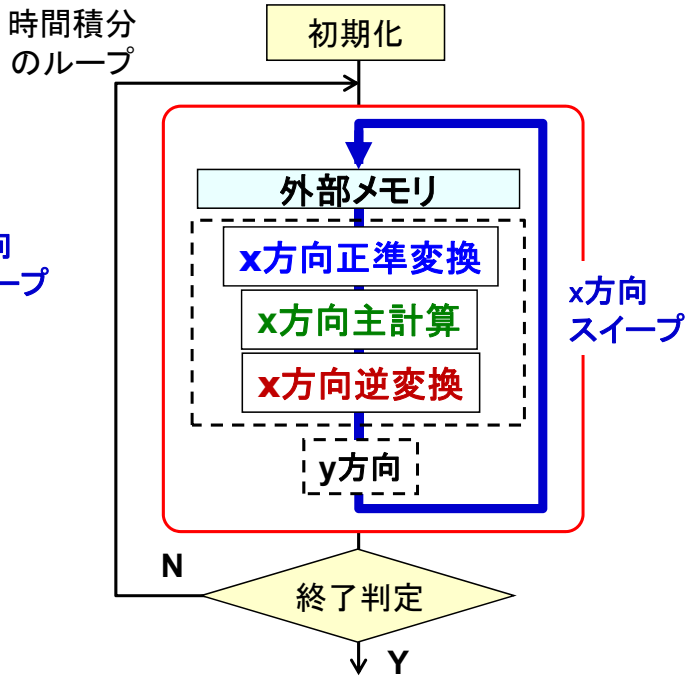
数値流体力学計算への応用事例

Arria10 FPGAによる津波シミュレーション

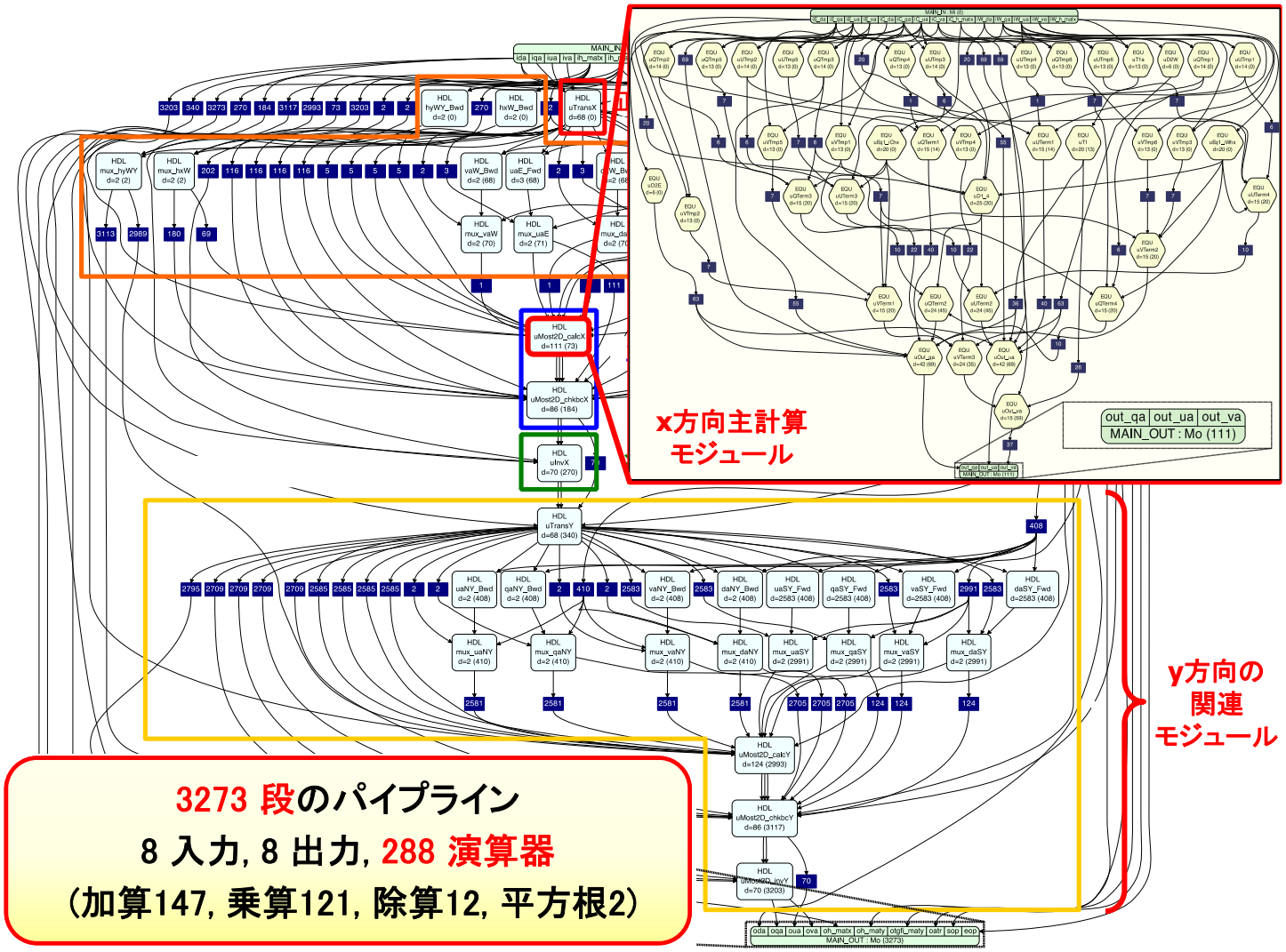
複数ループの融合と単一ストリーム化



元のアルゴリズム
（各ループの度にメモリを参照）

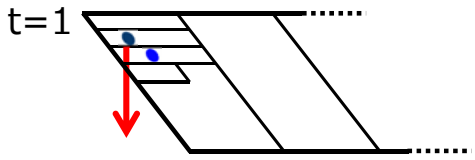


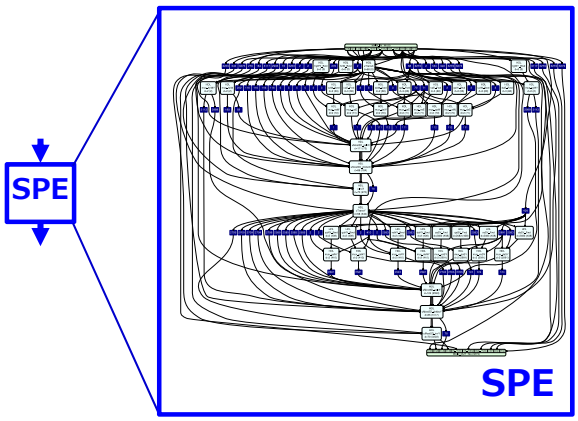
ストリーム化したアルゴリズム
(最初と最後のみメモリ読書き)



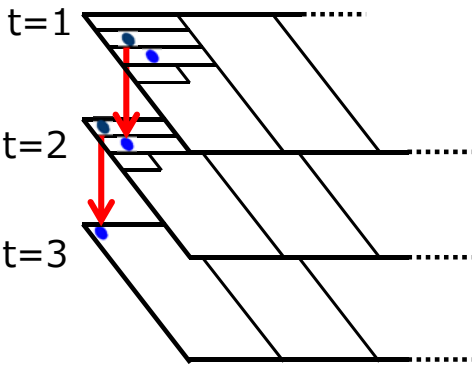
時間並列性による性能向上

時間並列性無し





時間並列性あり

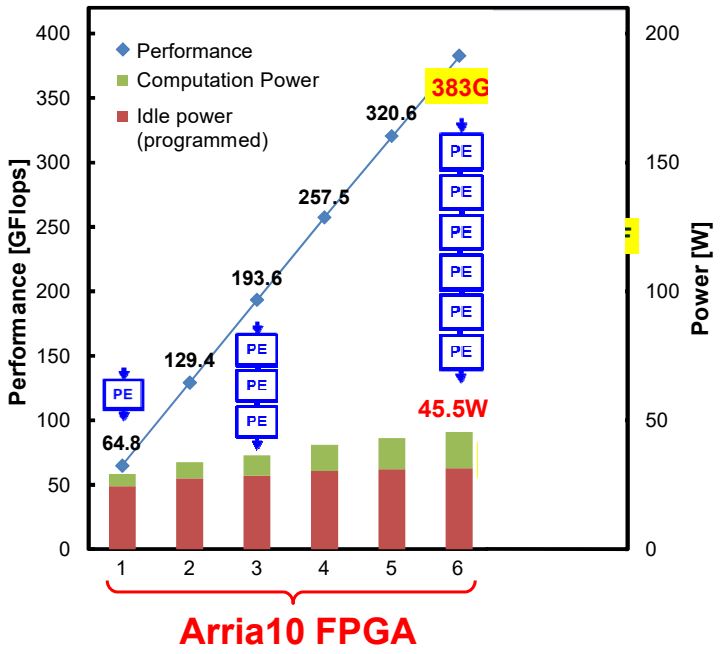


パイプラインの多段化
帯域一定のまま
性能向上

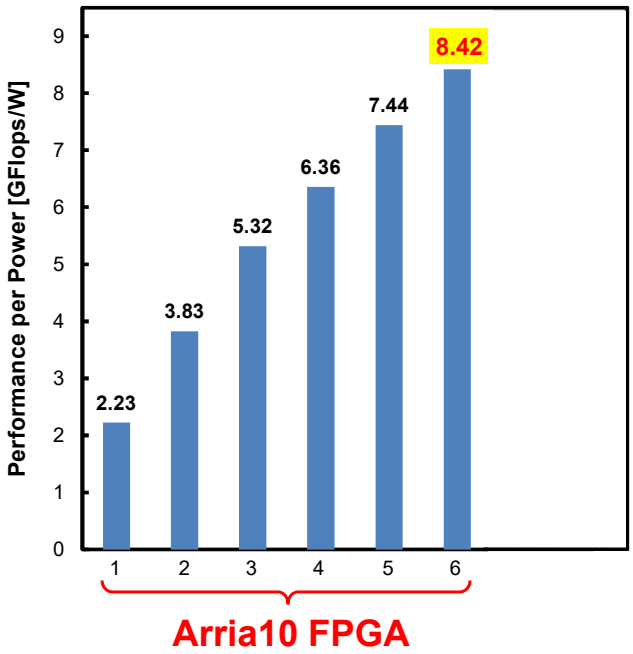
～6 SPEs

連続する複数タイムステップを並列に計算
（粗粒度パイプライン）

単一Arria10 FPGAによる計算性能と電力



Performance and power of board



Performance per power

Arria10 FPGAはGPUよりも優れた性能と電力性能比

おわりに

ポストムーアを生き延びる方法

- ✓ **カスタムハード**ウェアによる特化型計算 ⇒ 高効率化
- ✓ **データフロー**計算 ⇒ 大域的同期を回避

FPGAによるデータフロー計算

- ✓ 既に高性能・低消費電力計算デバイス
- ✓ 産業応用や高性能計算事例
- ✓ **課題**
  - ・プログラミング・最適化
  - ・膨大な配置配線時間（回路のコンパイル=数時間～）
  - ・大規模システムにおけるスケーラビリティ

異アーキテクチャ、異プログラミング手法が生まれる
カンブリア爆発的 多様化 時代 の到来