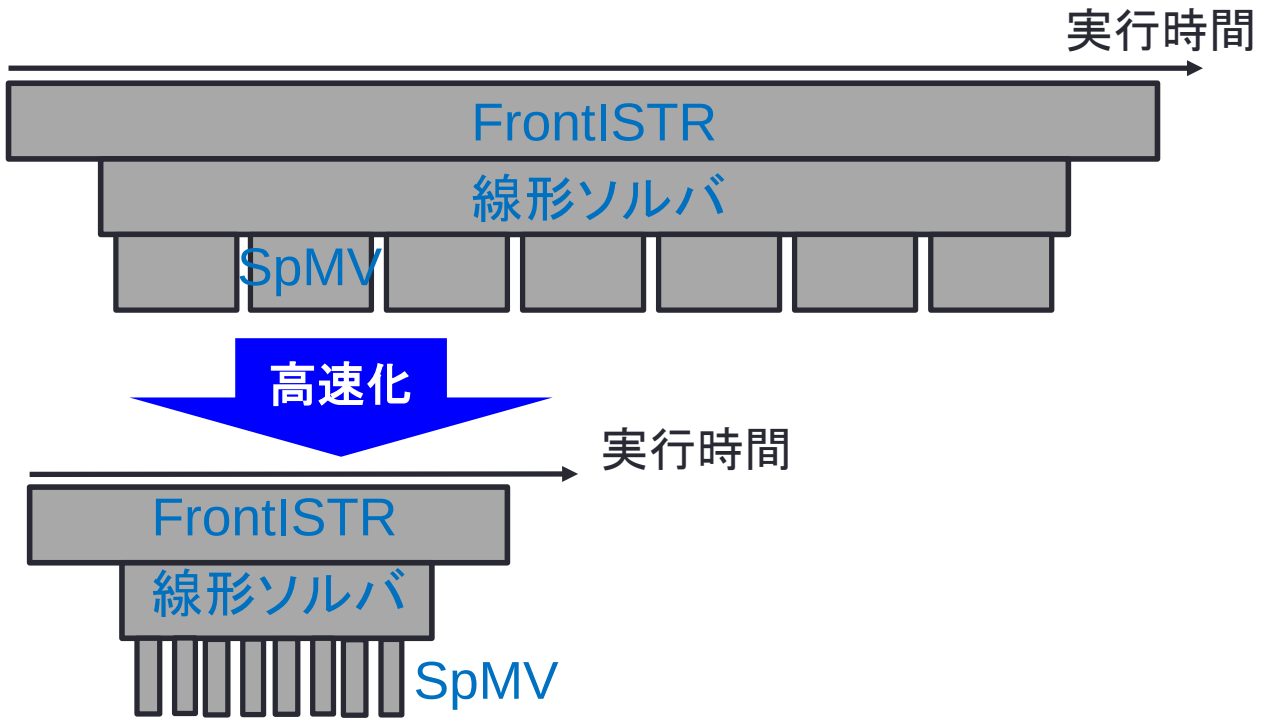
FPGA SoCへのFrontISTRの移植

● **井原遊[1)]　橋本学[1)]　奥田洋司[1)]**

1) 東京大学大学院新領域創成科学研究科　人間環境学専攻

背景

- FrontISTRのうち，線形ソルバがホットスポット
- 線形ソルバ(CG法)のうち，疎行列ベクトル積(SpMV)がホットスポット
- 行列ベクトル積の高速化 → 全体の高速化



背景

- CPUによる高速化
  - クロック数の向上
    - 期待できない．待っていても早くならない（はず）．
  - SIMD命令拡張
    - 並列性のあるデータをうまく持たなければならない．
    - ベクトル長は長くなっていく方向である．
  - メニーコア化（1ノード 100コアが見えてきている）
    - ロードバランスを取らなければならない．
    - 複数コアにデータ分散が必要になる．

CPU・GPU・ASIC・FPGAの特徴

- 適材適所に使う.

CPU	プログラミング言語で記述．開発が容易. 高速なクロック．処理が逐次的．多様な演算命令.
GPU	プログラミング言語で記述．開発は難しくない. SIMD的．メモリとのバス幅の問題．単純な命令.
ASIC	電子回路で記述．低消費電力．量産すれば低コスト. ただし回路修正は出来ない.
FPGA	電子回路的に記述．低消費電力．並列も逐次も回路次第. 回路変更が出来る.

- GPGPUは注目されているが
  - メモリ との バス幅が狭い．(PCI)
  - GPU内のメモリが小さい．（10GB程度まで．）
  - 長いSIMD的演算を活用しなければならない.

CPU演算の特徴

- a = a*b+c を例にすると,

1. メモリ から 必要なデータ を レジスタ に ロード
   - mov rax, a ;raxに変数aを格納
   - mov rbx, b ;rbxに変数bを格納
   - mov rcx, c ;rcxに変数cを格納

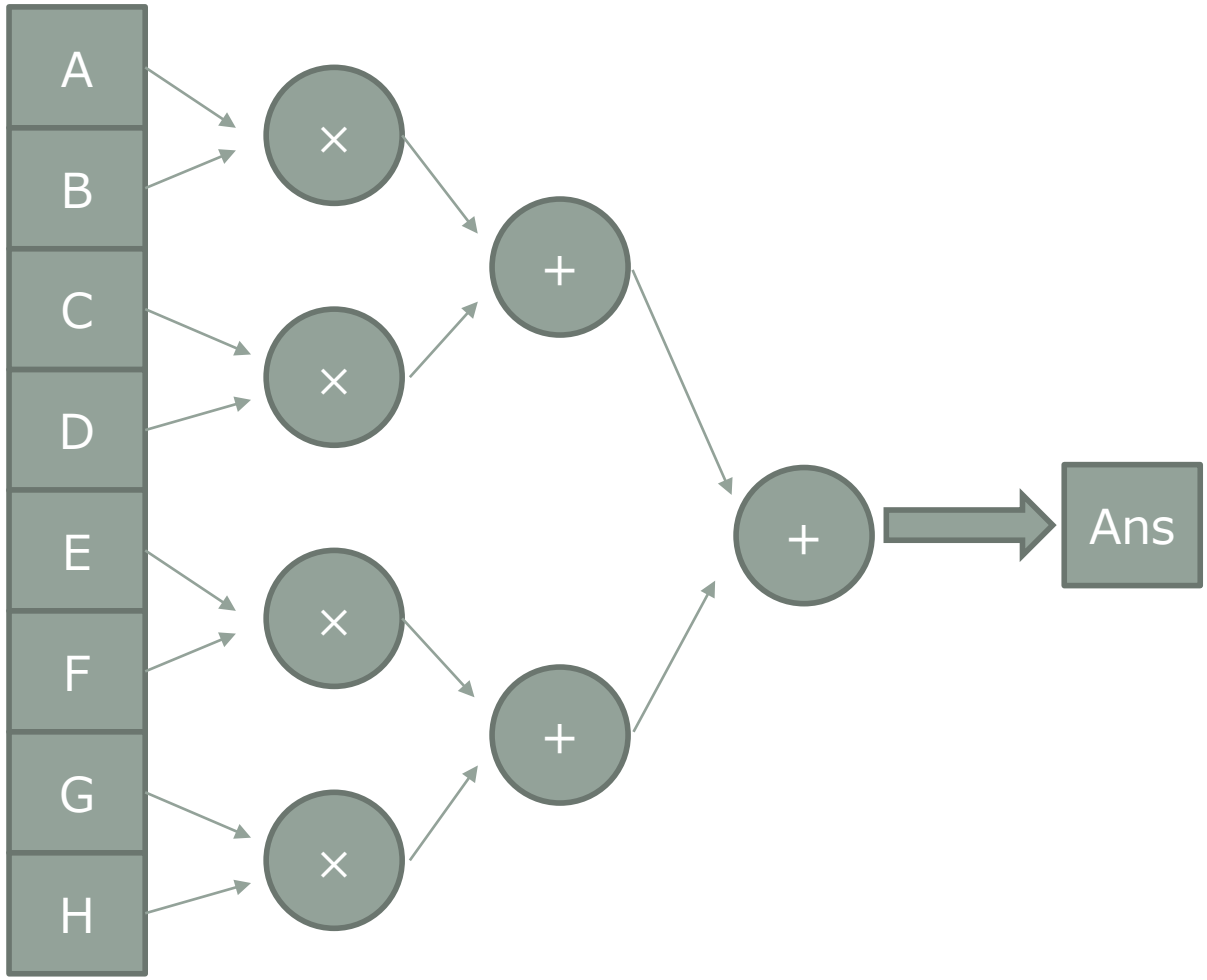
2. レジスタ同士 で 演算
   - mul rcx ;rax*rcx
   - add rax, rbx ;rax+rbx

3. メモリに 計算が終わったデータを ストア
   - mov a, eax ;aにraxレジスタを格納

- 処理を逐次に行う.
- （基本的に）1クロックで1処理をする.
  - ただし，今のCPUでは，積和演算なら拡張命令があるので一発で出来る.
  - 命令にはレイテンシもあるが，パイプライン処理もある.

FPGAを使うと

- 回路で書けるなら，処理は1クロックで終わる．



CPUなら

```
mov rax, a
mov rbx, b
mul rbx
mov rcx, rax
mov rax, c
mov rbx, d
mul rbx
add rax, rcx
mov rax, rdx
mov rax, e
mov rbx, f
mul rbx
mov rcx, rax
mov rax, g
mov rbx, h
mul rbx
add rax, rcx
add rax, rdx
```

CPU と FPGA

- CPU： レジスタの数は決まっている.
  - 他のデータを計算するためには，一旦追い出さないと行けない.
  - キャッシュはあるが，最悪ではメモリも戻す.
- FPGA: 回路設計次第

- CPU：処理するデータ が 固定長.
- FPGA：回路が書ければ任意の長さのデータを処理可能.

- CPU： 複雑な処理が得意.
  - データを追い出して，別の命令を実行して，戻す.
  - いろんな命令が用意されている.
- FPGA： 回路の通りにしか動かない.
  - 一組の回路では，同じことしか出来ない.
  - 場合分けで回路が複雑になる.

目的

- CG法における行列ベクトル積や，直接法のLU分解等の
  高頻度・高負荷部分のハードウェア記述化を目指す.

- ここでは，その前段階として Arria 10 SoC の ARMプロセッサで実行する.

検証用FPGAハードウェア

- インテル Arria 10 SoC 開発キット

Fig.1. FPGA SoC開発キット[1]

[1] https://www.altera.co.jp/products/boards_and_kits/dev-kits/altera/arria-10-soc-development-kit.html

Arria10 SoC 開発キットのインターフェース

- SoC (プロセッサ部)
  - ハード・プロセッサー・システム (HPS)
    - デュアルコア ARM* Cortex*-A9 MPCore
    - CPU キャッシュ
  - FPGA
- PCI Express
- イーサネット・ポート
- SFP
- USB
- DDR4 Memory
- SDフラッシュカード
- 文字LC, Displayport

Arria® SoC シリーズ 10 ハード・プロセッサー・システム

https://www.altera.co.jp/products/soc/portfolio/arria-10-soc/arria10-soc-hps.html

Arria 10 SoC スペック

- HPS memory size
  - 1GB DDR4 (256Mb x 40 x single rank) - *ships with kit*
- FPGA memory size
  - 2GB DDR4 (256Mb x 72 x single rank)
- HPS Boot Flash
  - SD Micro flash card: 4GB (Kingston)

- 評価ボード単体では メモリサイズは大きくない.
  - 1台で実問題を解くには不十分.

開発ボードの詳細

- 1枚でコンピュータとしても完結.
- インターフェースが物理層で存在.

- 制御をFPGA/HPSで実施.
- ボード間の接続も可能.

- インタフェースは様々.
  - PCI
  - SFP
  - Ethernet
  - SDIビデオ
  - DisplayPort
  - USB
  - シリアル
  - sma コネクタ（高周波）

Arria 10 SoC Development Kit User Guide, https://www.altera.co.jp/content/dam/altera-www/global/en_US/pdfs/literature/ug/ug-a10-soc-

Arria 10 SoC Block Diagram

- SoCに様々なデバイスが繋がっている.
- 制御をして様々なデバイスを実現可能.
- FPGA部への実装前に,CPUで演算をする.

SpMV と FPGA

- 密行列
  - 単にベクトルと行列を入力して行列ベクトル積演算をそのまま回路化すれば良い.
  - 回路規模が大きくなるが，実装できると効率よく計算できる.
- 疎行列
  - CG法：反復中で行列の非ゼロプロファイルは変わらない.
  - 演算の順番は同じ．入ってくるデータが違うだけ.
  - この特徴をうまく回路化．特定の問題専用の演算機を毎回作るイメージ.

HPS上での動作検証の報告

- (BootFlashにプリインストールされた)Angstrom linux
  - パッケージからコンパイラのインストール
    - × Fortranコンパイラがない
  - GCC を arm上で コンパイル
    - × エラー対応しても，最終的にコンパイラのバグに当たったり解決せず.
  - 他プラットフォームでクロスコンパイル
    - × GCC ARM Toolchain → Fortranコンパイラがない
    - × GCCクロスコンパイラーを作成してビルド → うまく行っていない.
    - ○ Ubuntu 提供の クロスコンパイラで動作確認.
      - （arm-linux-gnueabihf）

HPS Boot Flashの書換

HPS Boot Flash にOSを入れる

↓

Linuxで起動できる.

- 現在検証中.
  - OSは，HPSのARMアーキテクチャに対応していれば良い.
  - ARM用に用意されたディストリビューションを持ってくる.

HPS から FPGA の操作

- 現在検証中.

今後の検討課題

- SpMVを動かす.
- CG法全体をハードウェア化.
- 直接法での検討
  - フロンタル法，スカイライン法，ブロックLU分解． 他

- FPGA同士での並列演算
  - インターコネクトを含めて記述.
    - 高速な分散メモリハードウェアもできる.
  - メモリ制約も回避.
  - FPGAで完結できれば1クロックでベクトル積が終わる.
    - 高速なソルバも可能.