はじめに

- プロセッサ・シミュレータ
  - プロセッサの挙動を再現するソフトウェア

1. ファンクショナル・シミュレータ（エミュレータ）
   - プログラマから直接見える機能のシミュレーションを行う
   - 例：VMWare，VirtualPC など

2. サイクル・アキュレート・シミュレータ
   - プログラマからは直接見えないマイクロ・アーキテクチャまでも含めて，サイクル・アキュレートに再現
     - キャッシュ，分岐予測，out-of-order 実行 etc…
   - IPC を測定することができる

SimpleScalar

- サイクル・アキュレートなシミュレータとしては
  デファクト・スタンダード

- 問題点：
  1. 性能バグの検出容易性（detectability）
     - エミュレーションを命令フェッチ時にin-order に行う

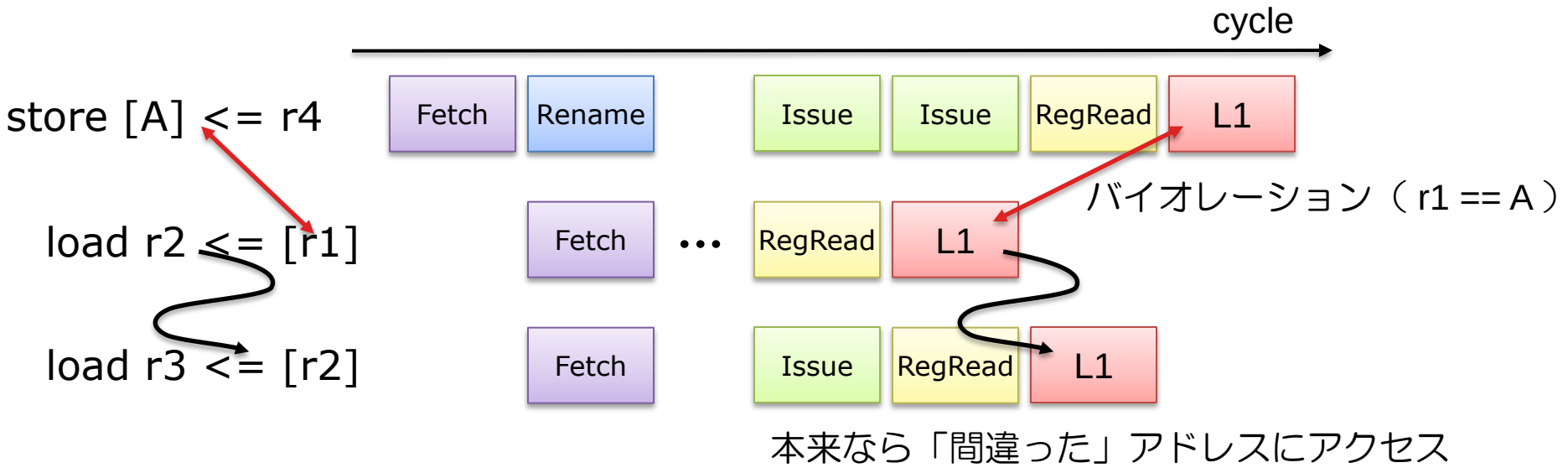
     ⇒バグがあっても，「正しく」終了してしまう

  2. 忠実性（fidelity，再現性）
     - データ投機に失敗した場合の挙動が再現できない

  3. 拡張性（extensibility）、可読性
     - 機能が非常に固定的
     - 一つの関数が数千行とか

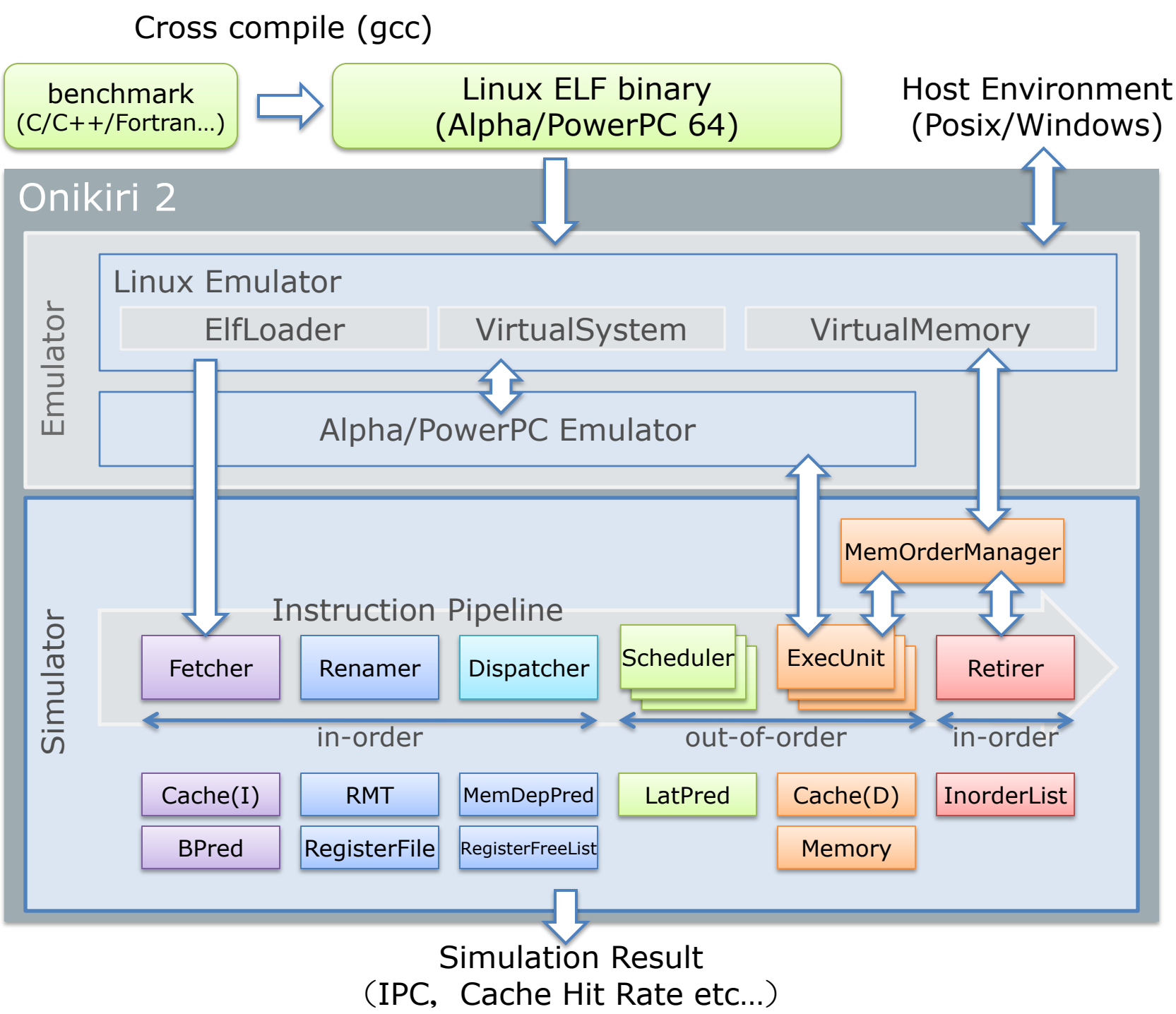
SimpleScalar の問題：データ投機に失敗した場合の挙動

- データ予測ミスした命令の後続の挙動が再現できていない
  - 例：アドレス一致/不一致予測ミス
  - キャッシュ・ヒット率などに影響が出る



目的・方針

- SimpleScalar の問題点の解消
- 当研究室の研究テーマへの対応
  - ツインテール・アーキテクチャ
  - 逆デュアルフロー・アーキテクチャ
  - マルチコア/マルチスレッド への対応
- 最近のターゲット環境への対応
  - SPECCPU2006．最新のgcc…


Cross compile (gcc)
benchmark
(C/C++/Fortran…)
Linux ELF binary
(Alpha/PowerPC 64)
Host Environment
(Posix/Windows)
Onikiri 2
Emulator
Linux Emulator
ElfLoader
VirtualSystem
VirtualMemory
Alpha/PowerPC Emulator
Simulator
MemOrderManager
Instruction Pipeline
Fetcher
Renamer
Dispatcher
Scheduler
ExecUnit
Retirer
in-order
out-of-order
in-order
Cache(I)
RMT
MemDepPred
LatPred
Cache(D)
InorderList
BPred
RegisterFile
RegisterFreeList
Memory
Simulation Result
（IPC，Cache Hit Rate etc…）

鬼斬弐

- 設計の方針：
  - スーパスカラ・プロセッサの基本的・最小限の性質を抜き出して抽象化
- パイプラインのモデル
  1. フェッチ，リネーム　　：in-order
     - 制御とデータに関する投機を行う
  2. 実行　　：out-of-order
     - リネーム時に予測した依存関係を満たすよう，スケジューリング
     - スケジューラはプロセッサ内に1つまたは複数存在
  3. コミット，リタイア　　：in-order

鬼斬弐の設計
データに関する投機

- データに関する投機
  - アドレス一致/不一致予測
  - 値予測
- リネーム時に，命令間に投機的な依存関係を張る事により実現
  - レジスタ＆メモリは統一的に扱う
    - レジスタ・リネーミングも，予測による依存として扱う
    - Wakeup の実装の共通化
- 例：アドレス一致/不一致予測
  - ロード/ストア命令のアクセスするアドレスはリネーム時には不明
  - アドレスが一致すると予測したロード/ストア命令間に，投機的に依存関係を張る

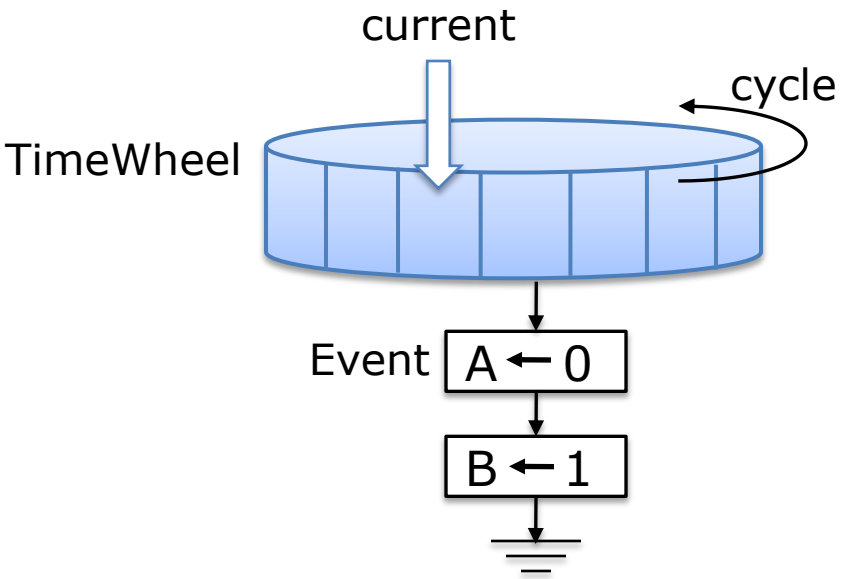
鬼斬弐の設計
エミュレーションの実行タイミング

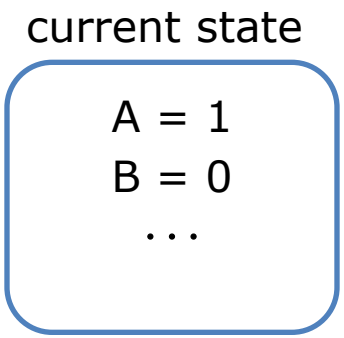
- エミュレーション：
  - 命令実行時に，out-of-order に行う
    - Simple Scalar では，命令フェッチ時にin-order に
  - 実際の物理レジスタ上の値を使用

- 利点：
  1. 性能バグの検出が容易に
     - バグがあった場合，実行結果もおかしくなる
     - 例：依存関係が解決していない命令を実行した場合，結果もおかしくなる
  2. データ系投機の実装が可能に
     - 誤った値を使用しての実行が継続可能

状態変化の計算と更新

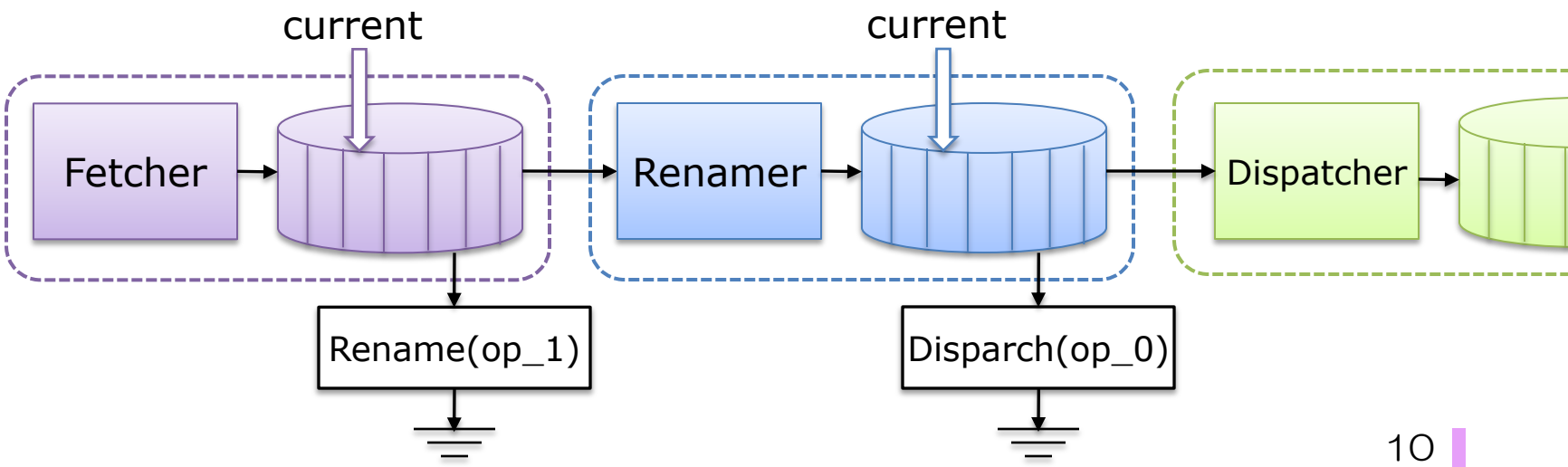
- 状態の変化の計算と，更新をそれぞれ別のフェーズに分離
- 更新をイベントとしてタイムホイールに登録
  1. current サイクルに登録されているイベントの処理
  2. current サイクルの状態から次の状態を計算イベントとして登録
  3. current を進める





パイプラインの表現

- PipelineNode とPipeline の接続によって表現
  - PipelineNode
    - Fetcher，Renamer などの機能を実現
  - Pipeline
    - タイムホイール
- ストールの表現：
  - Pipeline（タイムホイール）のcurrent の更新を停止する事により実現



マルチコア/マルチスレッド（SMT）への対応

1. 論理資源と物理資源のそれぞれを多重化，識別可能に
   - 論理資源：PC，アドレス，命令
     - アドレス空間の識別子を付加
   - 物理資源：コア/スレッド毎に必要なものを分けて多重化
     - コア毎に必要なもの：
       - Fetcher，Renamer，RegisterFIle，PHT･･･
     - スレッド毎に必要なもの：
       - FetchPC，RMT，InoderList，GlobalHistory･･･

2. Fetcher/Retirer のマルチスレッド対応
   - Fetch/Commit 対象スレッドの決定と処理

鬼斬弐の実装

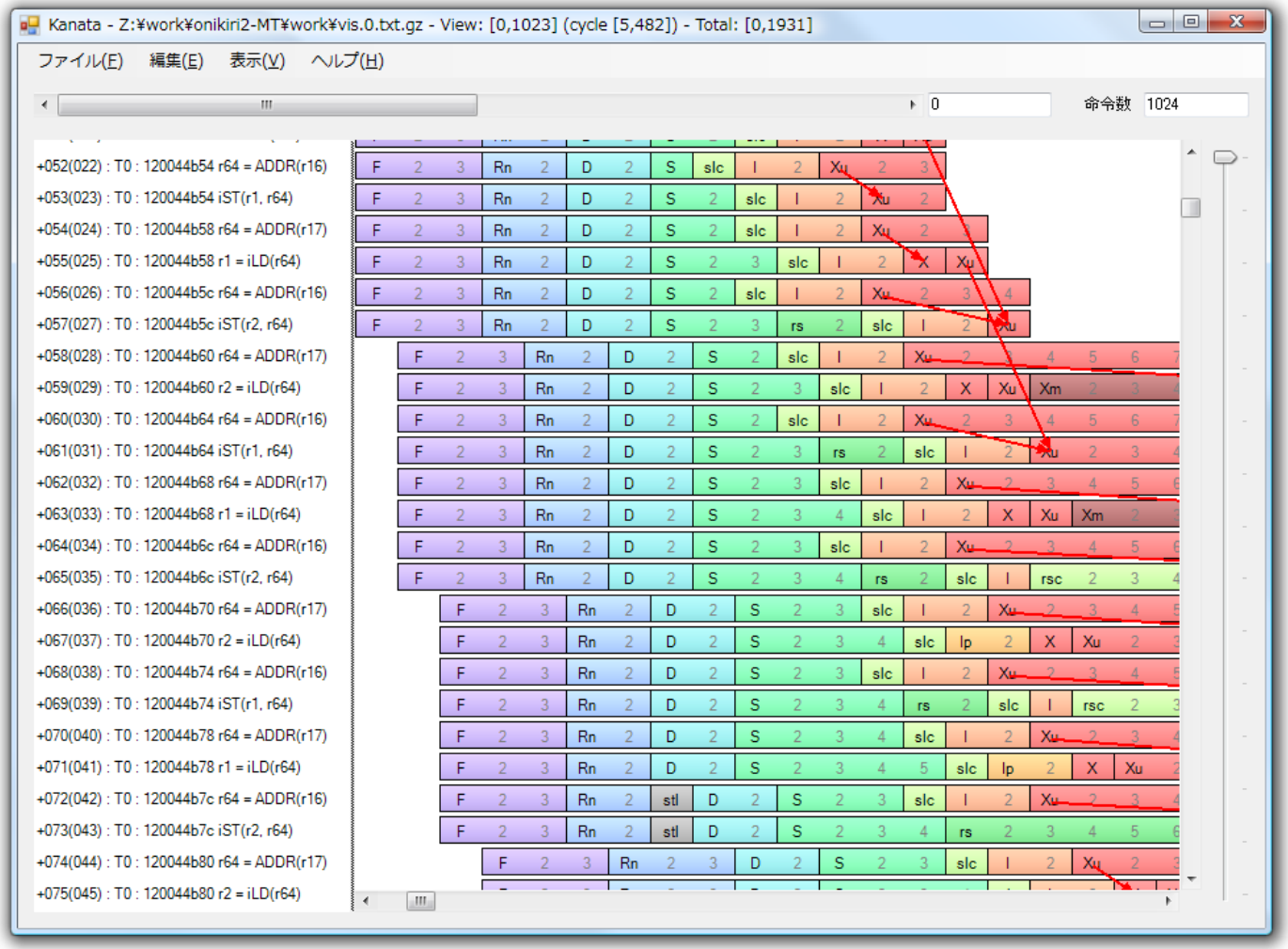
- 方針：
  - C++ で記述
  - オブジェクト指向，デザインパターン，テンプレート…
- 拡張性のための仕組み
  - Hook
    - 各処理に対し，ユーザー定義のフックを仕掛けられるように
  - OpExtraStateTable
    - 命令毎に関連づけられたデータを管理するテーブル

  ⇒ オリジナルのソースを編集することなく，拡張可能に

その他の機能・特徴

- 開発環境：
  - MS Visual Studio/GCC の双方に対応
  - Visual Studio プロジェクト・ファイルと，GCC 用 Makefile を一元管理
- 対応ターゲット：
  - ISA：Alpha/PowerPC 64bit
  - 最新のGCC Ver4 系クロスコンパイラを利用可能
  - SPEC CPU 2000/2006 の全ベンチマークを実行可能
- 周辺ツール等
  - クラスタ投入，データ集計
  - SPECCPU 2000/2006 実行用XML定義
  - パイプライン可視化ツール「Knanata」

パイプライン可視化ツール「Kanata」



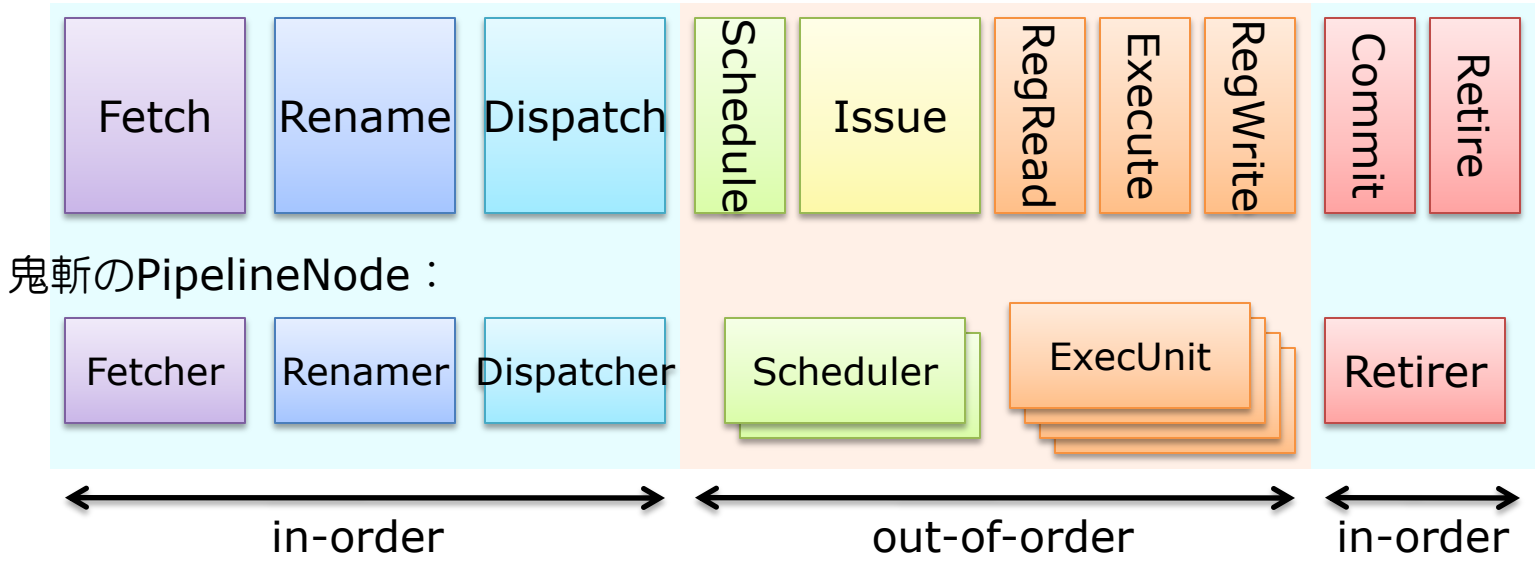
比較と利用例

- 実行速度（@Opteron 2.2GHz）
  - SimpleScalar　：916 K insns/sec
  - 鬼斬弐　：　68 K insns/sec
- コード量
  - SimpleScalar　：40723 lines
  - 鬼斬弐　：35394 lines
- 利用例
  - 非レイテンシ指向レジスタ・キャッシュ
    - 共有部分　：566 lines
    - 通常のレジスタキャッシュ　：481 lines
    - 非レイテンシ指向レジスタ・キャッシュ　：870 lines
  - 全てHookで記述
    - 鬼斬本体側のソースには一切修正を行わず

付録

パイプラインのモデル

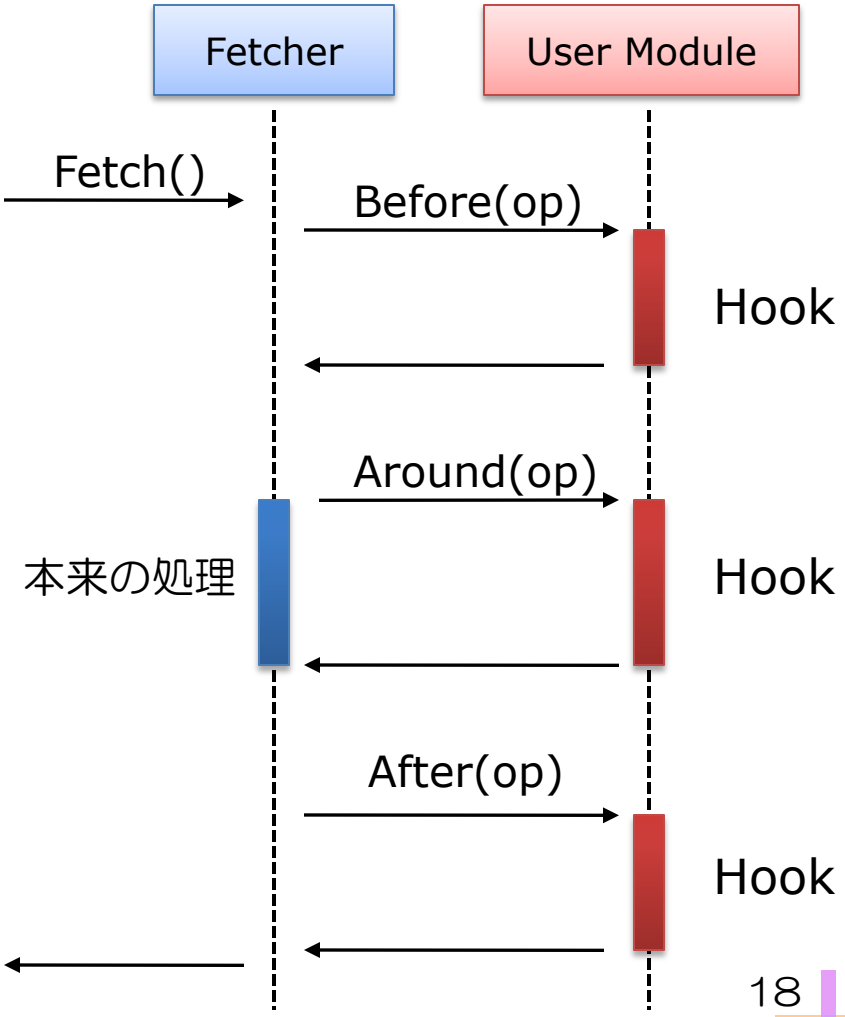
パイプライン：



- フェッチ，リネーム　：in-order
  - 制御とデータに関する投機を行う
- 実行　：out-of-order
  - リネーム時に予測した依存関係を満たすようスケジューリング
- コミット，リタイア　：in-order

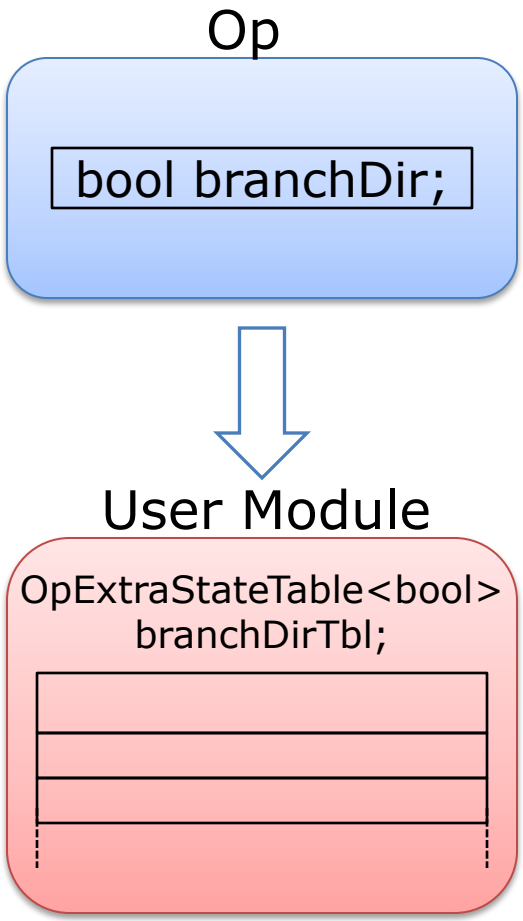
Hook

- シミュレータの各処理に対して，
  - 前後に処理を追加
  - 代替の処理を追加
- 処理の例：
  - フェッチ，リネーム…
- 鬼斬のソース・コードを編集すること無く，ルーチンを追加可能



OpExtraStateTable

- ユーザー・モジュールが命令ごとに情報を記憶したい



今後の課題とQ&A

- 今後の課題：
  - バグ取り
  - マルチプロセッサ周りの実装
    - メモリ・モデル
    - キャッシュ・コヒーレンス・プロトコル

- おまけ
  - Q：名前の由来は？
  - A：刀の名前
    - 平安時代に作られた刀で，渡辺綱と言う人がこの刀を使って鬼の腕を切った…らしい
    - 初代メインプログラマー：渡辺くん

その他の機能・特徴

- パラメータ設定・シミュレーション結果の出力
  - XML を利用
    - 階層化された構造を容易に扱うことが可能
    - SMP/SMT 時のモジュール間接続の指定
- 周辺ツール等
  - クラスタ投入，データ集計
  - SPECCPU 2000/2006 実行用XML定義
  - パイプライン可視化ツール「Knanata」