

計算機アーキテクチャ 第二 (O)

10. アウトオブオーダー実行プロセッサ フロントエンド

東工大 大岡山キャンパスで開催



- 学会の**雰囲気**を知る良いチャンスです.
- 東工大の**先輩**達も発表します.

Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005



学生ボランティア募集

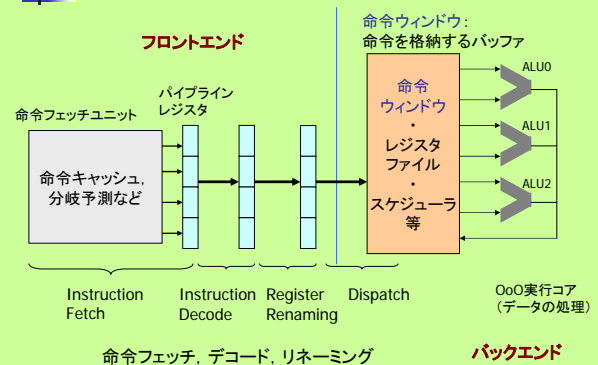
- 〆切: 2011年1月10日(月)17:00
- 募集人数: 57人
- 作業内容: 学会(2011/3/1~4)の開催手伝い
 - 受付, 会場設営, 発表会場のベル係, アンケート収集, ゴミ片付け等.
- 謝礼: 4日間で合計 25,000円.
 - 日給 7,000円, 午後半日 4,000円※ 交通費込み, 昼食なし
- 申込み・質問先: vol@sde.cs.titech.ac.jp



詳しくはこちらにアクセス下さい.
<http://www.sde.cs.titech.ac.jp/vol.html>

Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

アウトオブオーダー実行プロセッサの構成



Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

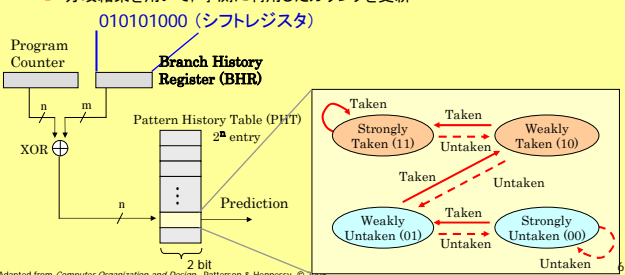
高いバンド幅の命令フェッチ

- パイプラインにバブルを生じさせないためには、条件分岐命令をフェッチした時に、次の3つを予測しなければならない.
 - フェッチしている命令が分岐かどうか
 - 分岐方向
 - 分岐先アドレス

Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

Gshare (TR-DEC 1993)

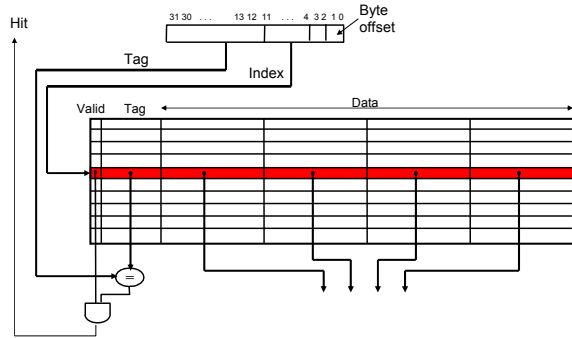
- グローバル分岐履歴と分岐アドレスとの排他的論理和によりパターン履歴表へのインデックスを作成
 - パターン履歴表は2ビット飽和型カウンタの配列で、選択された2ビットカウンタの値により分岐方向を予測 (bimodalと同じ)
 - 分岐結果を用いて、予測に利用したカウンタを更新



Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

命令キャッシュの実装

- ラインサイズ 4ワード (16 Byte)



Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

7

命令キャッシュの実装

```

lcache::lcache(int icache_size, main_memory *m) {
    mem = m;
    size = icache_size;
    buf = (icache_line *)calloc(size, sizeof(iline));
}

struct icache_line {
    int valid;
    int tag;
    int data[4];
} iline;

class lcache {
    main_memory *mem;
    icache_line *buf;
public:
    int size;
    lcache(int, main_memory*);
    int fetch(data_t, data_t*);
};

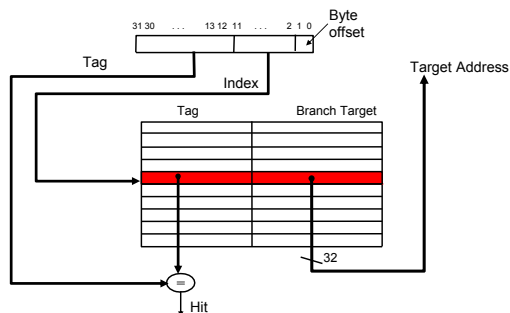
int lcache::fetch(data_t pc, data_t *ir) {
    int index = (pc >> 4) % size;
    data_t tag = (pc >> 4);
    if(buf[index].valid && buf[index].tag==tag) { /** hit **/
        for(int i=0; i<4; i++) ir[i]=buf[index].data[i];
        return 1;
    }
    else { /** cache miss **/
        buf[index].valid = 1;
        buf[index].tag = tag;
        for(int i=0; i<4; i++){
            data_t ir_t;
            mem->ld_4byte(pc+4*i, &ir_t);
            buf[index].data[i] = ir_t;
        }
        return 0;
    }
}
    
```

Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

8

Branch Target Buffer (BTB)の実装

- 分岐成立の場合にのみ、分岐先アドレスを登録する。
- Validビットは利用しない。



Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

9

Branch Target Buffer (BTB)の実装

```

BTB::BTB(int btb_size) {
    size = btb_size;
    buf = (btb_line *)calloc(size, sizeof(btb_line));
}

struct btb_line {
    int tag;
    int data;
};

class BTB {
    btb_line *buf;
public:
    int size;
    BTB(int);
    void fetch(data_t, data_t*);
    void regist(data_t, data_t);
};

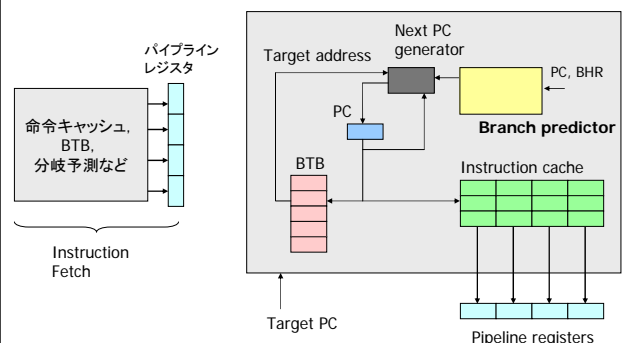
void BTB::fetch(data_t pc, data_t *target) {
    int index = (pc >> 2) % size;
    data_t tag = (pc >> 2);
    if(buf[index].tag==tag) *target=buf[index].data;
    else *target = 0;
}

void BTB::regist(data_t pc, data_t target) {
    int index = (pc >> 2) % size;
    data_t tag = (pc >> 2);
    buf[index].tag = tag;
    buf[index].data = target;
}
    
```

Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

10

命令フェッチユニットの例

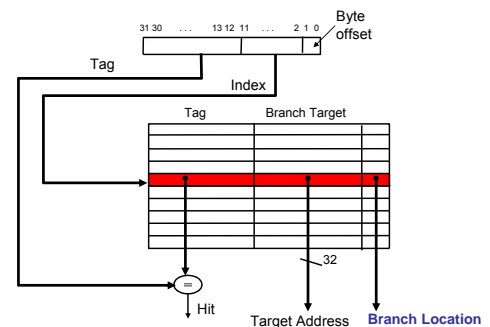


Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

11

Branch Target Buffer (BTB)の改良

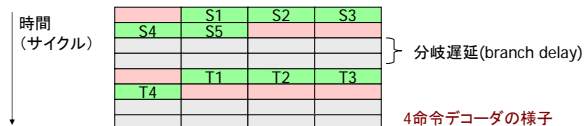
- キャッシュラインに1つの分岐のみを許す



Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

12

分岐命令S5の飛び先をT1とする.

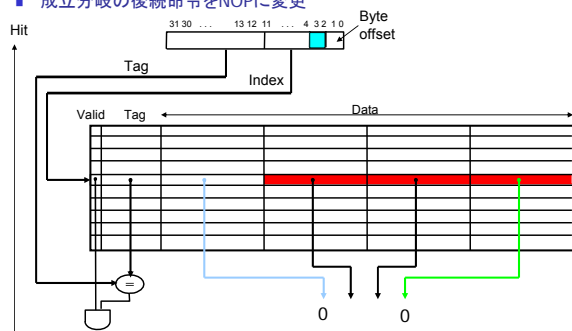


マイク・ジョンソン, スーパースカラプロセッサ 13

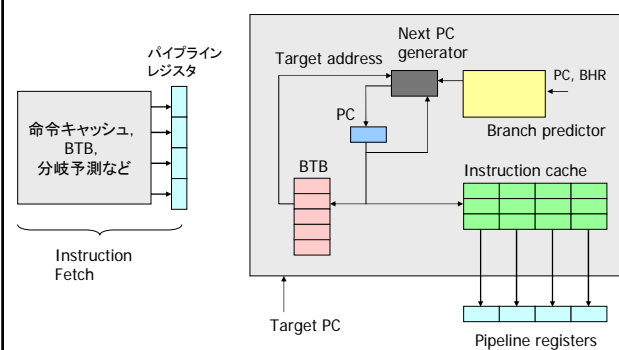


14

PCが指し示す以前の命令をNOPに変更
成立分岐の後続命令をNOPに変更

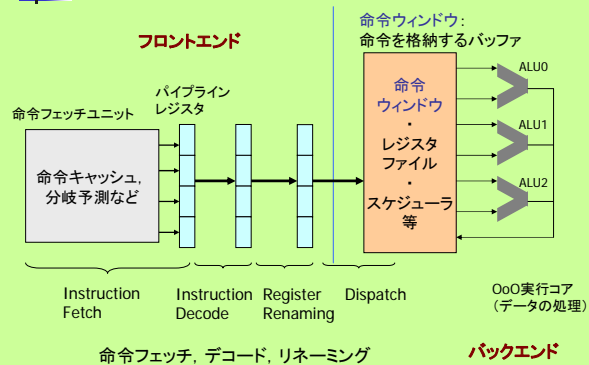


15



16

フロントエンド



17

計算機アーキテクチャ 第二 (0)

10. コンピュータシステム

18

SimMips: 教育・研究に有用な
Linuxが動く5000行のMIPSシステムシミュレータ



○藤枝 直輝(東工大)
渡邊 伸平(東工大)
吉瀬 謙二(東工大)

開発の背景

20

- ◆コンピュータシステムの複雑化
 - ▶システムシミュレータも同様
 - ▶シンプルで扱いやすいシステムシミュレータへの要求
 - ◆プロセッサの高速化
 - ▶コードをシンプルに保ちつつ現実的な速度でシミュレーションが可能
- ↓
- ◆シンプルさと可読性を重視した
MIPSシステムシミュレータSimMips

SimMipsの概要

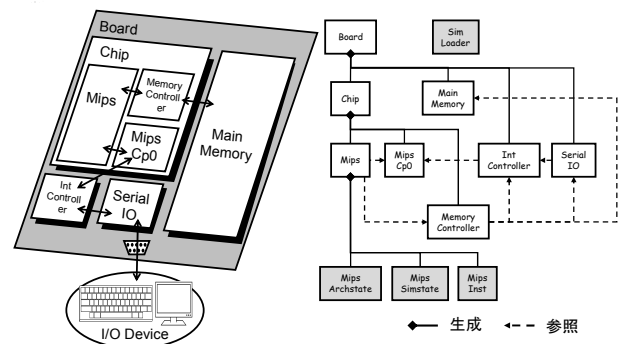
21

- ◆命令セット: MIPS32 R2のサブセット
 - ▶実装命令数は約100種
(浮動小数点命令を除くほぼ全て)
- ◆コード量: 4,422行(v0.5.0)
 - ▶コメント, 空行を含む
- ◆主要部分(~v0.4.0)の開発期間: 約2ヶ月半
 - ▶開発は学生2名
 - ▶休日などを含む

```
744 define.h
21 main.cc
622 board.cc
297 memory.cc
227 simloader.cc
899 mips.cc
767 mipsinst.cc
309 cp0.cc
536 device.cc
4422 total
```

SimMipsのモデル(v0.5.0)

22



SimMipsの2つのモード

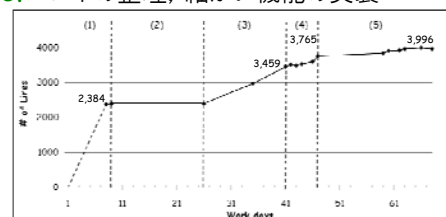
23

- ◆Appモード (Applicationモード)
 - ▶静的リンクされたユーザプログラムを実行
 - ▶プロセッサシミュレータとしての利用
- ◆OSモード
 - ▶OSカーネルを動作させるモード
 - ▶あらかじめカーネルやRAMディスクがロードされた状態を想定
 - ▶システムシミュレータとしての利用

開発の段階

24

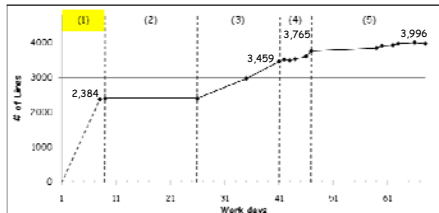
1. 計算コア部分(Appモード)の実装
2. 制御コア(CPO)の機能について調査
3. CPO, 割り込み, シリアル(OSモード)の実装
4. デバッグと検証
5. コードの整理, 細かい機能の実装



開発の段階

25

1. 計算コア部分(Appモード)の実装
2. 制御コア(CPO)の機能について調査
3. CPO, 割り込み, シリアル(OSモード)の実装
4. デバッグと検証
5. コードの整理, 細かい機能の実装

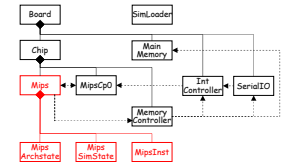


ComSys2008 2008/11/13

Mipsクラス

26

- ◆ MIPSの計算コア部分
- ◆ 可読性のため, ハードウェアにはないクラスを追加
 - ▶ レジスタファイルなどはMipsArchstateクラス
 - ▶ 統計情報(命令ミックスなど)はMipsSimstateクラス
 - ▶ 命令情報(デコード結果など)はMipsInstクラス



ComSys2008 2008/11/13

プロセッサのステージを意識した構成

27

```
int Mips::step_func()
{
    ...
    fetch();
    decode();
    regfetch();
    execute();
    if (inst->attr &
        LOADSTORE) {
        memsend();
        memreceive();
    }
    writback();
    setnpc();
    return ...;
}
```



ComSys2008 2008/11/13

フェッチ, デコード

28

```
inline void Mips::fetch()
{
    ...
    mciid = mc->enqueue(addr, 4, NULL);
    ...
}

void MipsInst::decode()
{
    opcode = (ir >> 26) & 0x3f;
    funct = ir & 0x3f;
    ...
    switch (opcode) {
    case 0:
        ...
        switch (funct) {
        case 33:
            op = ADDU____;
            attr = READ_RS | READ_RT |
                WRITE_RD;
            break;
        ...
        }
    }
}

addu rd, rs, rt
opcode      funct
0  rs  rt  rd  0  33
31 26 25 21 20 16 15 11 10 6 5 0
```

ComSys2008 2008/11/13

レジスタフェッチ, 実行, ライトバック

29

```
inline void Mips::regfetch()
{
    ...
    if (inst->attr & READ_RS)
        rrs = as->r[inst->rs];
    if (inst->attr & READ_RT)
        rrt = as->r[inst->rt];
    ...
}
```

```
inline void Mips::execute()
{
    switch (inst->op) {
    case ADDU____:
        rrd = rrs + rrt;
        break;
    ...
    }
}
```

```
inline void Mips::writback()
{
    ...
    if (inst->attr & WRITE_RD)
        as->r[inst->rd] = rrd;
    ...
}
```

```
addu rd, rs, rt
⇔ r[rd] = r[rs] + r[rt];
```

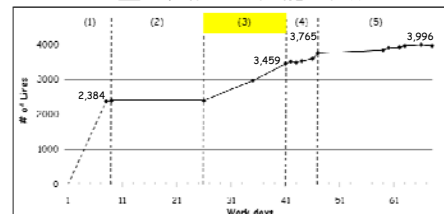
	0	rs	rt	rd	0	33
	31	26	25	21	20	16
	15	11	10	6	5	0

ComSys2008 2008/11/13

開発の段階

30

1. 計算コア部分(Appモード)の実装
2. 制御コア(CPO)の機能について調査
3. CPO, 割り込み, シリアル(OSモード)の実装
4. デバッグと検証
5. コードの整理, 細かい機能の実装



ComSys2008 2008/11/13

Linuxの動くシミュレータにするために

31

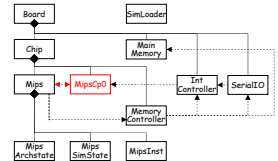
- ◆ 制御コアCPO(Coprocessor Zero)の実装
 - ▶ 例外
 - ▶ TLB(Translation Lookaside Buffer)
 - ◇ MIPSではソフトウェアによる制御
 - ▶ アドレス変換
- ◆ 割り込みコントローラの実装
- ◆ シリアル入出力の実装

ComSys2008 2008/11/13

MipsCp0クラス

32

- ◆ 例外の取り扱い
 - ▶ 制御レジスタに例外に関する情報を記録
 - ▶ 例外ハンドラ(通常0x80000180)から実行再開
- ◆ TLBとアドレス変換
 - ▶ TLB<->制御レジスタ間データ授受命令の実装
 - ▶ 仮想ページ番号(VPN)でTLBエントリを検索
 - ◇ エントリなし→TLBミス例外
 - ◇ 有効ビット無→ページフォルト

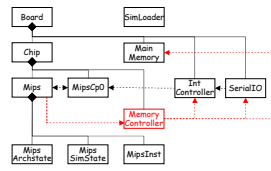


ComSys2008 2008/11/13

MemoryControllerクラス

33

- ◆ ロード・ストアは必ずこのクラス経由
- ◆ アドレスを見て適切なクラスへのリード・ライトを行う

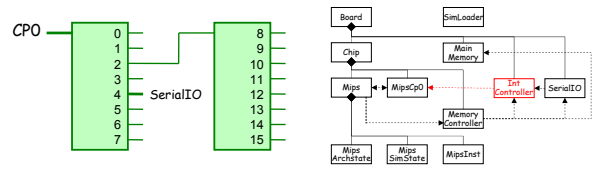


ComSys2008 2008/11/13

IntControllerクラス

34

- ◆ 割り込みコントローラ
 - ▶ Intel 8259相当(x2)
 - ▶ 接続デバイス(現在はSerialIOのみ)から割り込みを受け取り、CPOへ送る

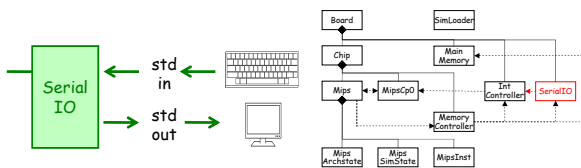


ComSys2008 2008/11/13

SerialIOクラス

35

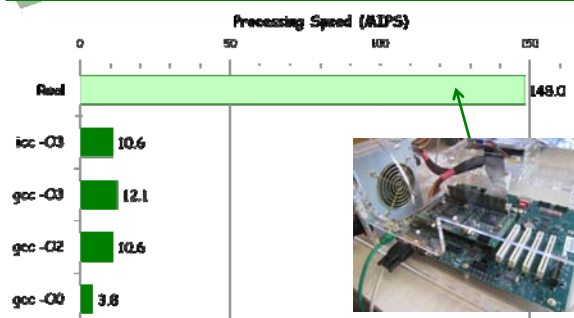
- ◆ シリアル接続のI/Oコントローラ
 - ▶ ns 16550相当
 - ▶ 入力はキーボード(標準入力)から受け取り、出力はコンソール(標準出力)へ



ComSys2008 2008/11/13

SimMipsのシミュレーション時間

36



※Xeon X5365(3.06GHz), gcc 4.1.2, icc 10.1
ベンチマークはクイックソートを使用

ComSys2008 2008/11/13

SimMipsを用いたメニーコア研究

37

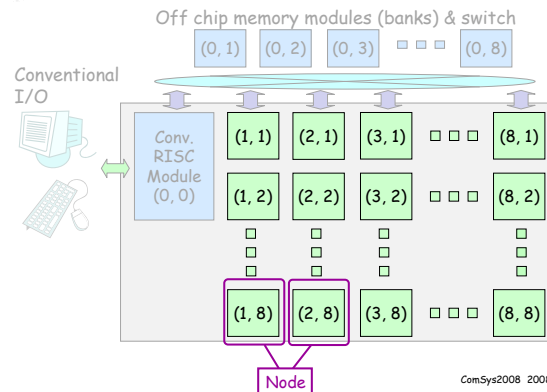
- ◆マルチコア・メニーコア
 - 近年の高性能汎用プロセッサのトレンド
- ◆シミュレータを短期間で構築することが重要
- ◆メニーコアシミュレータSimMc※のコア部分にSimMips(のAppモード)を使用
 - ▶SimMips側への変更は行わず
 - ▶短期に、かつネットワーク部分に注力した開発が可能に

※植原ほか: シンプルで効率的なメニーコアアーキテクチャの開発, 情報処理学会研究報告 2008-ARC-180 (Oct. 2008)

ComSys2008 2008/11/13

アーキテクチャモデル

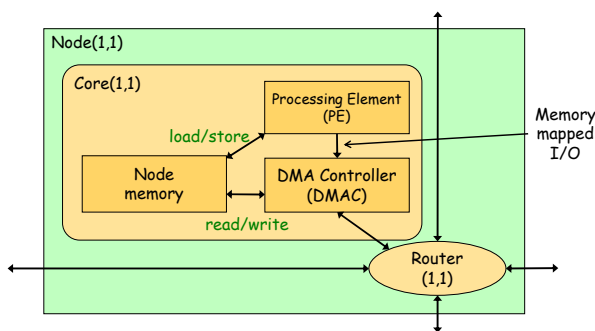
38



ComSys2008 2008/11/13

ノードの構成

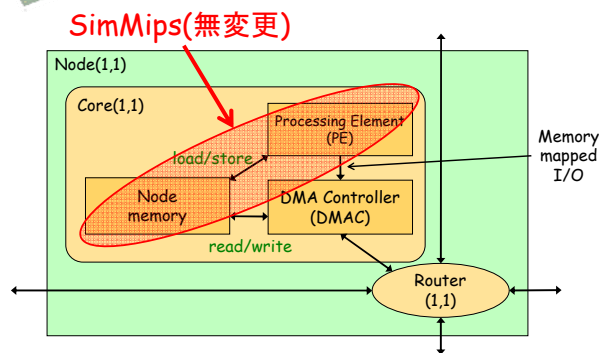
39



ComSys2008 2008/11/13

ノードの構成

40



ComSys2008 2008/11/13

まとめと課題

41

- ◆シンプルで可読性の高いMIPSシステムシミュレータ **SimMips**を開発
 - ▶教育・研究への有用性
- ◆本日使用したバージョンは <http://www.arch.cs.titech.ac.jp/SimMips/>にて公開中
- ◆今後の課題
 - ▶機能追加: ネットワークなど... (→v1.0)
 - ▶OS等を含めた包括的な教育・研究プラットフォームの構築

ComSys2008 2008/11/13

課題: SimMipsプロセッサシミュレータ

- 004_rand, 050_tokei, 090_dhry を動作させ、正しく動作していることを確認せよ。
- データキャッシュのヒット率を測定する仕組みを追加し、ヒット率を測定せよ。(004_rand, 050_tokei, 090_dhry)
 - ダイレクトマップ方式、ラインサイズは4ワードとする。
 - セット数を8, 16, 32, 64, 128, 256に変更した場合のヒット率を示せ。
 - このキャッシュのヒット率を改善する任意の方式を実装し、その効果を示せ。
- データ値予測 (last-value predictor) の予測精度を測定する仕組みを追加し、予測精度を測定せよ。レジスタに値を書く命令が対象。
 - -e40m 004_rand.mex
 - -e40m 050_tokei.mex
 - 090_dhry
- MieruPC用の面白いアプリケーションを作成せよ。その魅力を示せ。
 - ソースコードは各自のディレクトリに格納すること。
 - MieruPC用のアプリケーションとして利用させてください。

Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

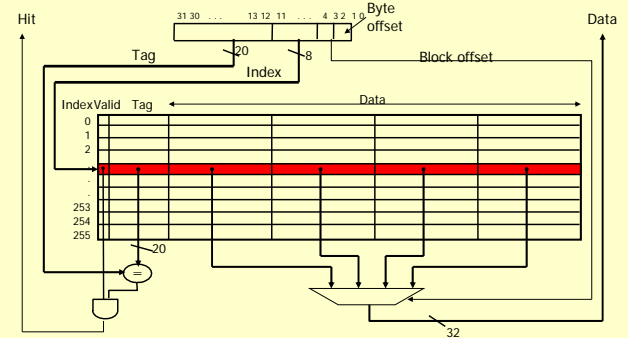
課題: SimMipsプロセッサシミュレータ

- 1月17日の講義開始時にレポートを提出
(Intel系のプロセッサとLinuxの利用を推奨)
 - 追加したコードの主要部分の記述と説明
 - 実行結果は正しいか? その根拠を示せ.
 - それぞれの課題にどれくらいの時間を必要としたか?
 - 感想, 苦労した点など.
 - SimMipsに対する要望など.
- 質問などは
 - [kise at cs.titech.ac.jp](mailto:kise@cs.titech.ac.jp) まで

Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

Multiword Block Direct Mapped Cache

- Four words/ **block**, cache size = 1K words



44

マルチレベル・ストライド値予測機構による
命令レベル並列性の向上
(JSPP 1999)

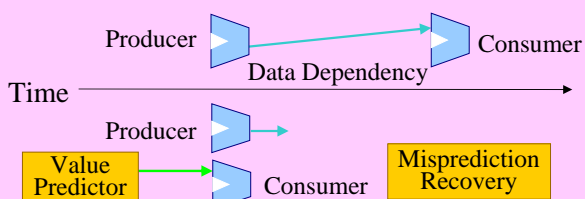
研究の背景

- 真のデータ依存関係が命令レベル並列性を制限
- 生産者から消費者へのデータの流れを解消する技術として値予測

Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

研究の背景

- 真のデータ依存関係が命令レベル並列性を制限
- 生産者から消費者へのデータの流れを解消する技術として値予測



Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

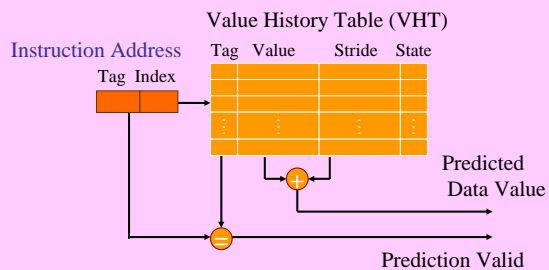
関連研究: 値生成のアルゴリズム

- Last-value予測
 - 最も近い過去に得られた値を予測値
- ストライド値予測
 - 最も近い過去に得られた2回の値の差分 Stride と、Last-value の和を予測値
- 2レベル値予測
 - 過去のn個の履歴の中からひとつを選択
- ハイブリッド値予測
 - 複数のアルゴリズムから選択

Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

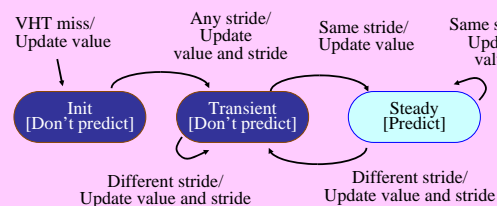
ストライド値予測機構

$$\text{Predicted Value} = \text{Last-value} + \text{Stride}$$



Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

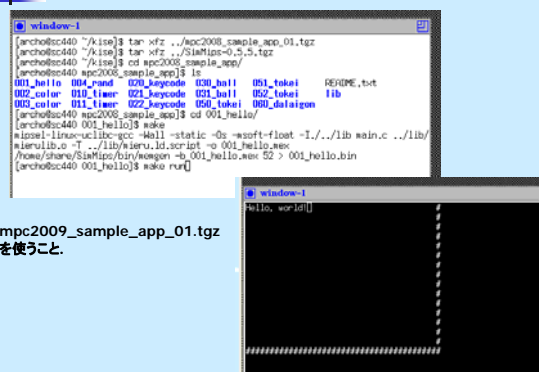
ストライド値予測機構 (cont.)



Stateフィールドの推移と予測アルゴリズム

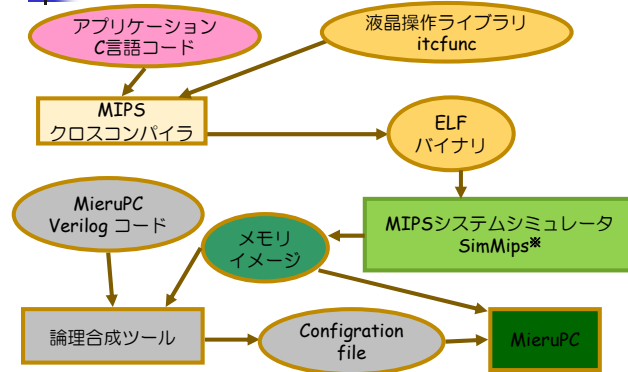
Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

SimMips-0.5.5 の使い方など



Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

アプリケーションの開発フロー



Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005

アナウンス

- 講義スライド, 講義スケジュール
 - www.arch.cs.titech.ac.jp
- 講義用の計算機のIPアドレスが変わりました.
- ユーザ名 archo で serv.arch.cs.titech.ac.jp にログイン
 - linuxなど
 - ssh archo@serv.arch.cs.titech.ac.jp
 - 講義時に伝えたパスワードでログイン

Adapted from Computer Organization and Design, Patterson & Hennessy, © 2005