ループに限定したハードウェアホットパス検出機構

中島 伸吾 † 横田 隆史 † 大津 金光 † 馬場 敬信 †
† 宇都宮大学大学院工学研究科情報工学専攻

1 はじめに

プログラム高速化の手法の一つとして動的最適化が注目されている．プログラム中の実行頻度の高い部分に最適化を適用することで高速化が期待できる．つまり，実行頻度の高いループ (ホットループ) や実行頻度の高いパス (ホットパス) を特定することが高速化へつながると考えられる．

パスの検出方法には，ソフトウェアで計測する方法とハードウェアプロファイラを用いる方法が挙げられる．

我々の研究室では，オーバーヘッドを低く抑えられる特徴をもつハードウェアプロファイラを採用し，ハードウェアによって実現容易な Bit Tracing[2] を用いたホットパス検出機構の開発がシミュレータ上で行われている [1]．本研究では，このホットパス検出機構を実際にハードウェア上で実現した際の有効性及び問題点を明らかにするために，FPGA (Field Programmable Gate Array) を用いてホットパス検出機構のハードウェア設計を行う．本稿では，どのようにハードウェアで構成していくかを検討する．

2 ループパスの提案

最適化対象をループに限定し，そのループ内の繰り返し部分のパス情報を得るためにループパスを提案している．ループパスとは，パスの末尾の後方分岐先のアドレスがパスの先頭アドレスと一致するパスのことである．ループパスは以下の情報を用いて表現する．

- 開始アドレス (bt_start_addr)
  Bit Tracing の開始アドレス，つまりループの先頭のアドレスを格納する．
- 分岐履歴 (bt_history)
  分岐結果である Taken/Not-Taken をそれぞれ“1”と“0”で記録する．
- 継続アドレス (bt_next_addr)
  通常は後方分岐先のアドレス (NPC) を格納するが，分岐履歴オーバーフローや間接分岐との遭遇でパスが終端した際には，次に続くパスの開始アドレスを格納する．
- パス情報 (bt_info)
  最適化を行う際に必要な情報及び Bit Tracing を行う際に必要な情報を格納する．
  - OVF:分岐履歴オーバーフローを示す．
  - CONT:継続パスであることを示す．
  - SRC:サブルーチンコール発生を示す．
  - INDIR:間接分岐が発生したことを示す．
  - SYSC:システムコール発生を示す．

3 ホットループパス検出機構

3.1 検出機構の構成

ホットループパス検出機構は，ループの検出を行う Bit Tracing Stack (BTS)・実行頻度の高いループを検出する Hot Loop Detector (HLD)・実行頻度の高いループパスを累積記録する Hot Path Accumulator (HPA) の 3 つで構成される．

図 1 にその構成図を示す．

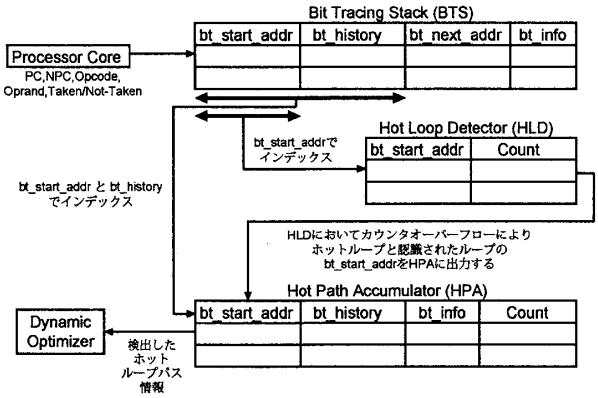

図 1: ホットループパス検出機構

3.2 BTS におけるループ検出

BTS はプロセッサコアとのインターフェース部分になり，命令コード (Opcode)・オペランド・プログラムカウンタ (PC)・次に実行する命令アドレス (NPC)・分岐の成立/不成立 (Taken/Not-Taken) を受け取り，Bit Tracing によってループの検出を行う．

Bit Tracing 中であることを示すために，bt_start フラグを用意し，初期値を False に設定しておく．後方分岐が発生した際にその分岐先 (NPC) を bt_start_addr に格納し，bt_start を True にして Bit Tracing を開始する．次の後方分岐が発生するまで，前方分岐の分岐結果 (Taken/Not-Taken) を bt_history に記録していき，次の後方分岐が発生したら分岐先 (NPC) を bt_next_addr に格納する．そして格納された bt_start_addr と bt_next_addr を比較し，一致したらループと判断し HLD・HPA へそのパスの情報を出力し，不一致なら内容を破棄する．

また Bit Tracing 中にサブルーチンコール・間接分岐が発生した場合は以下のようにして扱う．

- サブルーチンコール
  コール時に一旦 Bit Tracing を中断し，スタックをプッシュするとともに新たなパスの記録を開始する．リターン命令で戻った際にスタックをポップし Bit Tracing を再開する．
- 間接分岐
  間接分岐によってパスが終端するため，bt_next_addr に次の命令アドレス (NPC) を格納し，新たなエントリをプッシュしてパスの記録を再開する．その際，新たなエントリの bt_start_addr には次の命令アドレス (NPC) を格納し，継続パスであることを示すフラグ CONT を格納しておく．

3.3 HLD におけるホットループの検出

BTS においてループと識別されたパスの先頭アドレス (bt_start_addr) をテーブルに格納していく．このとき，継続パスであることを示すフラグ“C”がセットされていた場合は，ループの先頭ではないため格納しない．HLD に格納されているループと同じループが検出された場合は，Count をインクリメントする．この Count が一定値を超えた場合そのループをホットループとする．

A Hardware Hot Path Detector for Loop Optimizations
†Shingo Nakajima,Takashi Yokota, Kanemitsu Ootsu and Takanobu Baba
Department of Information Science, Faculty of Engineering, Utsunomiya University (†)

HLD のエントリ数は有限であるため，置換を行う必要がある．検出数の低いエントリを追い出すため LFU (Least Frequently Used) 置換を行う．

3.4 HPA におけるホットパスの検出

BTS から出力されたすべてのパスがテーブルに格納される．同じパスが検出された場合は，HLD と同様 Count をインクリメントする．また HLD からのホットループの先頭アドレス (bt_start_addr) を受け取り，HPA 内に格納されているパスのうち同一の先頭アドレスをもつパスをホットパスとして動的最適化機構へ出力する．HPA のエントリ数も有限であるため，置換を行う必要がある．ホットループ内のパスが残っていることが求められるため LRU(Least Recently Used) 置換を行う．

4 ハードウェアホットパス検出機構の検討

上記のホットパス検出機構を FPGA 上で実現するために，ハードウェア構成をどのようにするか検討を行った．

4.1 各テーブルについて

BTS・HLD・HPA では，ループ又はパス検出のために一時的にデータを格納する機能が必要となる．そのため，FPGA 上で構成する際にはブロック RAM を用いることとした．ブロック RAM を用いた場合，一度データを読み出して更新を行った後書き込みを行う必要がある．

4.2 BTS の構成

BTS 操作中に新たな分岐命令が発生した場合，一時的にデータを格納しておくバッファが必要となる．プロセッサコアから受け取った順に BTS 操作を行っていくことからバッファには FIFO (First In First Out) を用いることとした．プロセッサコアから受け取った分岐命令・プログラムカウンタ (PC)・次に実行する命令アドレス (NPC)・分岐結果 (Taken/Not-Taken) を一度 FIFO に格納する．そして，BTS が IDLE 状態の時に FIFO の内容を読み出して，BTS 操作に移る．

またBTS のエントリ数は有限であるため，スタックオーバーフローを起こしてしまう可能性がある．そこで，スタックオーバーフローが起きないようにスタックサイズを十分にとることにした．シミュレータでのプロファイリング結果によると，エントリ数を 256 程度にしておけばオーバーフローは発生しない．この結果を受け，エントリ数を 256 に設定することで，オーバーフローを回避する．しかし，エントリ数を 256 にしてもすべてのプログラムでオーバーフローしないとは言い切れないため，更なる検討が必要である．

以上を踏まえ BTS のハードウェア構成を図 2 に示す．

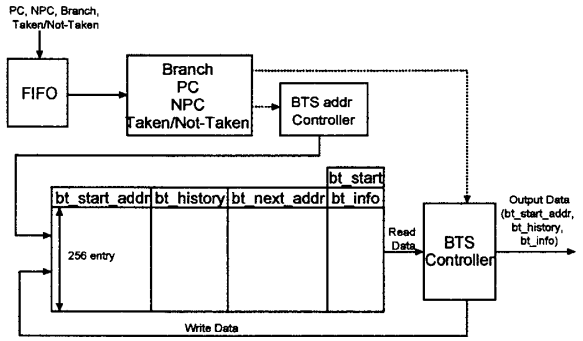

図 2: Bit Tracing Stack の HW 構成

4.3 HLD・HPA の構成

RAM にデータを一意的に格納するために，ハッシングを行う必要がある．ハッシュ関数には以下のものを用いる．

- HLD
  bt_start_addr を Index とし，n ビットずつ区切りそれらに xor 演算を行う．
- HPA
  bt_start_addr と bt_history を Index とし，n ビットずつ区切りそれらに xor 演算を行う．

HLD・HPA のエントリ数は有限であるため，置換を行う必要がある．そのため，図 3 のように構成し，置換を行うことにした．LRU 置換を行うために，各テーブルに Ref という 1 ビットのデータを持たせ，新しいデータには“1”を，古いデータには“0”を書き込むことで，2 つ RAM のうちどちらに新しいデータが格納されているかを判断する．

BTS から受け取ったデータが，HLD もしくは HPA に格納されているデータと同じものなら，Count をインクリメントする．同じものが格納されていない場合は，2 つの RAM の Ref をみて“0”が格納されているほう，つまり古いほうのデータを破棄し上書きを行う．

図 3 に HLD のハードウェア構成を示す．また HPA のハードウェア構成に関しては，HLD とほぼ同様の機構で実現可能なため省略する．

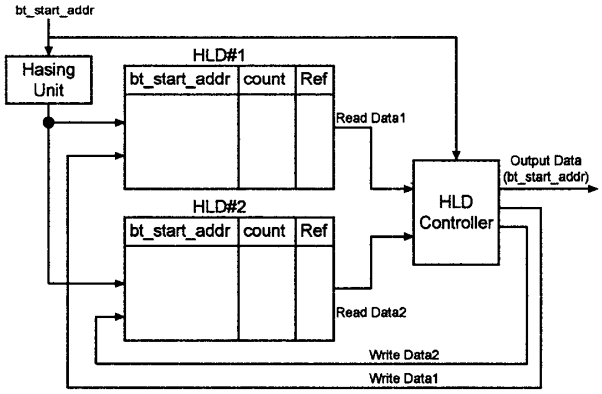

図 3: Hot Loop Detector の HW 構成

5 おわりに

本稿では，ループに限定したホットパス検出機構をどのようにハードウェア上で構成するかの検討を行った．しかし，検出したホットパスを動的最適化機構にどのようにして渡すか，などまだいくつか問題が残されている．残された問題を解決したのち，設計を行っていく．設計は Verilog-HDL を用いて行い，FPGA 上でホットパス検出機構を実現していく予定である．

謝辞

謝辞　本研究は，一部日本学術振興会科学研究費補助金（基盤研究 (B)18300014，同 (C)19500037，若手研究 (B)17700047）および宇都宮大学重点推進研究プロジェクトの援助による．

参考文献

[1] 矢野目 秀人，増保 智久，大津 金光，横田 隆史，馬場 敬信，“ループに限定したハードウェアホットパス検出機構”，電子情報通信学会コンピュータシステム研究会 (CPSY)，信学技報，Vol.107, No.175, pp.89–94, (CPSY2007-21), 2007 年 8 月．

[2] Vasanth Bala, “Low overhead path profiling”, HP Laboratories Technical Report HPL-96-87, 1996.