様式Ｃ－１９

科学研究費助成事業（科学研究費補助金）研究成果報告書

平成２５年　６月　７日現在

機関番号：２５４０３
研究種目：基盤研究（Ｃ）
研究期間：2010～2012
課題番号：２２５０００５１
研究課題名（和文）
演算精度不足によるプログラムの計算誤りを検出する研究
研究課題名（英文）
Detection of miscalculation caused by precision degradation
研究代表者
北村　俊明（KITAMURA TOSHIAKI）
広島市立大学・情報科学研究科・教授
研究者番号：10324683

研究成果の概要（和文）：
効率の良い演算精度低下検出を目的として、精度低下検出機能を追加した浮動小数点演算器を複数搭載したベクトルコプロセッサを開発し、精度低下検出が正しく行えることを確認した。また、実際に精度低下を発生するワークロードを用いて、汎用プロセッサで実行した場合と実行サイクル数を比較することで効率の良さを評価し、ベクトル型を採用することで５分の１程度のサイクル数とすることができ、プロトタイプシステムとしての性能も確保できた。
研究成果の概要（英文）：
To achieve the efficient accuracy degradation detection on floating point number calculation, we developed a vector co-processor that has two or more floating-point arithmetic units with accuracy degradation detection function. And we investigated that it worked correctly.
Moreover, we compared the number of execution cycles of the workload, which actually causes accuracy degradation, with a conventional processor. Because our vector processor can execute by about 1/5 number of execution cycles, we can practically use this system as a prototype system.

交付決定額

（金額単位：円）

	直接経費	間接経費	合　計
2010 年度	900,000	270,000	1,170,000
2011 年度	800,000	240,000	1,040,000
2012 年度	800,000	240,000	1,040,000
年度			
年度			
総　計	2,500,000	750,000	3,250,000

研究分野：総合領域
科研費の分科・細目：情報学、計算機システム・ネットワーク
キーワード：計算機アーキテクチャ、演算精度低下検出、長精度計算

１．研究開始当初の背景

コンピュータによる数値計算では、
- 演算結果の丸め
- オーバフロー／アンダフロー
- 桁落ち
- 情報埋没

などの要因により、計算した結果が正しい答えとなっていない場合がある。従来から、数値計算を専門に行っている研究者は、誤差解析や演算精度を変えて検算をするなど、多大な努力を払って結果の信頼性を確保してきた。また、これらの数値計算で使用される浮

動小数点データ表現も IEEE754 という規格で統一され、表現形式の違いによる結果の異なりが無くなっただけでなく、オーバフロー／アンダフローのハードウェアによる検出、不正確例外による丸め誤差の検出（実態は、ほとんどの演算で検出されるため実用的ではないが）も規格として採用され上記の計算誤りに対して一部は注意が払われるようになった。

しかし、コンピュータの利用が一般的になり、日常生活で欠かせないシステムとなっているがこのような計算誤りに対する認識は、充分に認知されているとは言えず、「計算機の出した結果だから正しい」と思い込んでいる利用者も多い。実際、このような誤差や桁落ちの解析を行うには、充分な知識が必要とされ、一般利用者にそれを期待することは困難である。そこで、桁落ちや情報埋没の発生を検出するハードウェア機構を、浮動小数点演算器に付加することでユーザに通知することが発想される。この機構は浮動小数点演算のハードウェアアルゴリズムの中で簡単な条件を検査することで検出できる。

この桁落ちや情報埋没がどのくらいの頻度で発生するものかは、アプリケーションに依存するが、例えば、素粒子物理の理論数値計算のために KEK 石川らのグループでは、自動計算システムを開発しており、計算コードを生成し、それに必要なライブラリを用意するだけでなく、物理的な性質を利用して検証する機能を有するが、この機能によりパラメータによっては桁落ちが発生することがわかっている。素粒子物理の理論数値計算が特に悪い条件とは言えず、それ以外の分野でも同様の事態が生じている可能性がある。しかもその発生を見いだすためには、プログラムやアルゴリズムの検証では正確に言い当てることはできず、どのようなデータが入力されるかも考慮に入れないと決まらないものであるため、一般的に対応可能とは言いがたい。

２．研究の目的

本研究は、大規模数値計算において、桁落ちによる結果の不正確さを検出し、桁落ちを生じた箇所と生じさせたデータを明確にユーザに通知し、計算を継続した場合の計算結果の不確かさを警告することである。近年、ディペンダブルコンピューティングに対する要求が高まっており、コンピュータシステムとして誤動作や故障による結果の誤りが無いことを保証する、あるいはシステムの可用性を高めることが強調されている。しかし、コンピュータというシステムそのものが持っている正確さの限界という落とし穴を見逃しており、本研究では、この点を補った演算結果に対する信頼性を高めるというディペンダビリティの向上を目指す。また、精度低下を恐れるあまり必要以上の長精度の演算を行うことによって、記憶域や実行時間の増大を招き無駄に電力を消費することを避ける。

３．研究の方法

(1) 浮動小数点演算器の基本設計

ハードウェア規模を推定するために倍精度浮動小数点演算器を設計する。また、FPGA ボードに実装し、基本的な演算機能の確認と桁落ち／情報埋没の検出論理を検証する。

(2) 計算誤り検出機構のソフトウェアインターフェース仕様の策定

計算誤り（桁落ち／情報埋没）検出機構付き演算器を通常のプロセッサの演算器として実装してしまえば、単に割り込み要因の追加等で対ソフトインターフェースは十分であるが、入出力バスを経由して演算機構を組み込むため、各種制御レジスタをメモリマップド I/O レジスタとして定義する。演算器側にローカルメモリを備えメモリとの DMA 転送を行うことで、単体の演算でなくある程度の大きさの粒度で演算を実行するような仕様とする。このような演算機能の定義としては、ちょうどベクトルプロセッサの命令セマンティックスが扱いやすいと考え参考にする。

(3) 計算誤り検出機能付き浮動小数点演算ライブラリの仕様策定

上記のようなシステム構成は、最近広く使用されている大量データ向け演算加速ボードと同様の構成となる。これらのシステムは、バンドルして提供されるソフトウェア・ライブラリを経由して使用するため、同様のライブラリ構成としておくと、これらのライブラリを用いた既存のソフトウェアシステム資産を流用あるいは改造して使用できる可能性が出てくる。このため、上記の様な既存システムのライブラリ仕様を参考として、仕様策定する。

(4) 計算誤り検出機能付き浮動小数点演算器の論理設計

準備計画で設計した浮動小数点演算器に桁落ち検出機構と情報埋没検出機構を付け加える。桁落ち検出機構は、減算後の正規化処理でシフト桁数が設定ビット数を超えた場合が、その条件となる。この設定ビット数は、制御レジスタとして指定できるような仕様としておくが、各種のワークロードで評価した後は固定化できるように適切な条件出しを図りたい。また、情報埋没検出機構は、加減算を行うときにオペランドの指数合わせのスケーリングでの仮数部シフト桁数が設定ビット数を越えた場合がその条件となり、設定ビット数は同様に変更可能としておく。

(5) 上記演算器を組み合わせて長精度演算を行うための論理設計

このフェーズの設計では、多数の演算器を

実装することを考えており、並列に動作させて実行速度を向上させる構成と、複数の演算器を連結するなど特定の結合をさせることで長精度計算を効率よく行う構成に、動的に構成を変化させることを可能とするハードウェア仕様とする。このため、演算器を接続するパス等のアーキテクチャ設計を行い、実際の論理設計を行う。

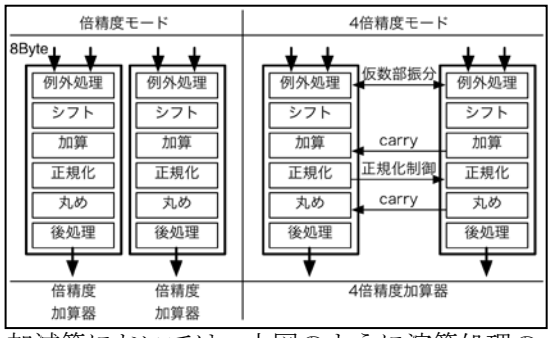
(6) アプリケーションによる検証

最初に対象とするプログラムは、素粒子物理の理論数値計算で現れる多次元積分で、桁落ちが発生する特異性のある場合があることが得られている。開発したシステム上で動作可能なようにライブラリやランタイムシステムを作成するとともに、評価を行う。また、その他の数値計算プログラムでも評価を行うと同時に、より日常的に使われるプログラムも対象とする。

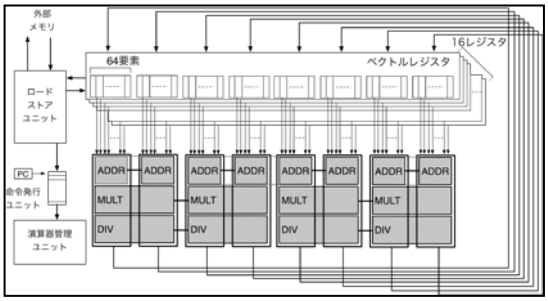
４．研究成果

(1) FPGA を用いたベクトルコプロセッサ型桁落ち検出システム

桁落ち検出、情報埋没検出の機能を搭載した浮動小数点加減算器を設計し、また、浮動小数点乗算器、浮動小数点除算器も設計して、これを８加減算、４乗算、４除算の並列実行可能なベクトルプロセッサ構成を採った。また、計算誤り検出時には、精度を増加させて再実行するものと考え、各演算器を２台連結させて４倍精度の演算もできるように追加の回路を組込んだ。



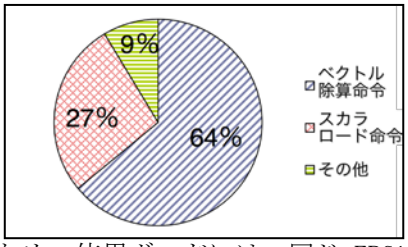
加減算においては、上図のように演算処理の特定ステップで下位と上位の演算器で情報の交換が必要となり、スループットは落としても動作周波数は倍精度演算が最適となるように配慮した。全体構成を下図に示す。



このコプロセッサをインストールした FPGA とローカルメモリを構成する SRAM を実装したボードを、ホストとなる FreeBSD の PC に PCI Express で接続し、また、これを使用するためのソフトウェア・ライブラリを作成した。これにより、KEK 石川・湯浅氏から提供いただいた素粒子に関する数値積分コードが実行できるようになり、評価を行った。このコードの核となるループを示す。

```
for(i1=1;i1<=512;i1++){                                  ループ3
  xx = x30[i1-1]*cnt0;
  by = 1.0 - xx;
  cnt2 = by -ay;
  for(i2=1;i2<=512;i2++){                                ループ2
    yy = x30[i2-1]*cnt2;
    bz = 1.0-xx-yy;
    cnt4 = bz-az;
    for(i3=1;i3<=512;i3++){                              ループ1
      zz=x30[i3-1]*cnt4;
      d=-xx*yy*s
        -tt*zz*(1.0-xx-yy-zz)
        +(xx+yy)*pow(ramda,2)
        +(1.0-xx-yy-zz)*(1.0-xx-yy)*pow(fme,2)
        +zz*(1.0-xx-yy)*pow(fmf,2);
      w3[i3-1] = (cnt0*cnt2*cnt4*gw30[i1-1]
                 *gw30[i2-1]*gw30[i3-1])/ (pow(d,2));
    }
    for(sum = 0.0,i=1;i<=512;i++)  sum = sum + w3[i-1];
    w2[i2-1]=sum*h;
  }
  for(sum = 0.0,i=1;i<=512;i++)  sum = sum + w2[i-1];
  w1[i1-1]=sum*h;
}
for(i=1;i<=512;i++)   sum = sum + w1[i-1];
```

この実行状況を観察すると、ベクトル化とベクトル命令のチェイニング機能により、スカラ実行の PC と比較して実行クロック数は減少させることができた。しかし、FPGA によるインプリメントのため動作周波数を高速化できず、実時間では高速とは言えない。そこで実行命令の占める割合を調査したところ、大半が除算命令で占めており、また、スカラロード命令もそれに続いて大きな割合を占めていることが分かった。

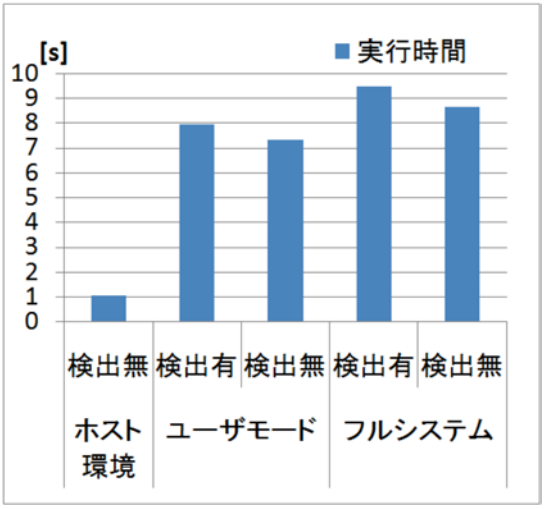


このため、使用ボードには、同じ FPGA チップが２個実装されていることを利用し、チップ間の同期と言う問題点は増えるが、２個使うことで使用できる回路量を増やして除算アルゴリズムを高速のものに変更し、また、２チップに分割して実装することでメモリバスを倍増させてメモリスループットを向上させた。この結果、除算・メモリアクセス命令の実行速度を向上させることができた。実行サイクル数を以下に示す。

	倍精度モード		4 倍精度モード	
	throughput [element/cycle]	latency [cycle]	throughput [element/cycle]	latency [cycle]
加減算	1	6	2	9
乗算	1	4	4	9
除算	27	30	57	60
総和演算	64	66	64	66

(2) 仮想マシン・ソフトウェアによる検出システム

ベクトルコプロセッサ型のシステムでは、演算誤りの検出を行なうには、ソースプログラムを修正して、検出したい箇所をベクトル化してコプロセッサで処理を行う必要があった。しかし、日常使用しているアプリケーションのソースプログラムは、アクセス可能とは限らない。このため、例えば市販の表計算ソフトウェアで、このような計算誤りの検出を行うことを考えた。このため近年その性能の高さやや改造の容易さが充実しているQEMU システムを用い、動的な命令変換システムにより浮動小数点加減算に対して計算誤りの検出コードを付加するシステムを開発した。この方法により、OpenOffice の Calc で標準偏差を求める計算において、データの条件が悪いと桁落ちが発生していることを



確認した。

実行時間については、KEK 提供コードの実行結果(上図)に示すように、QEMU を使わずそのまま実行した場合に比べて 8～10 倍の実行時間がかかる。しかしそのほとんどは QEMU による動的翻訳モードによる遅れであり、計算誤りの検出にかかるオーバヘッドは、コード生成をハンドコーディングで最適化した効果もあり１０％程度に収めることができた。

(3) 今後の発展

本研究に対して、「検出した精度低下は最終結果にどのように影響を及ぼすのか」「一度に精度低下が起るのではなく、少しずつ精度低下が繰り返し起るような場合はどうするのか」と言う質問を良く受けた。これに対して、演算対象となる数値そのものに、その時点での精度情報を持たせる数表現形式を考え、その演算器の構成を考えた。精度の予測を厳密な誤差解析で行なうと、前進誤差解析と同じになり、非常に悲観的な結果となって、ほとんどの計算において有効桁数が無くなってしまう。このため、桁落ちのみに着目した有効桁追跡の方式を考え、今後も発展的に研究を継続していきたい。

５．主な発表論文等

（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計３件）

①安仁屋宗石，吉田浩章，伴野充，北村俊明，SIMD命令セットを用いた浮動小数点演算における精度低下検出，情報処理学会　研究報告「計算機アーキテクチャ(ARC)」，査読無，2013-ARC-205(14)，2013，pp.1-7
②松田稔彦，北村俊明，計算精度低下を検出する PC エミュレータの開発，情報処理学会　研究報告「計算機アーキテクチャ(ARC)」，査読無，2011-ARC-197(24)，2011，pp.1-6
③金子啓太，北村俊明，精度低下検出を行う浮動小数点演算器の検討と評価，情報処理学会研究報告「計算機アーキテクチャ(ARC)」，査読無，2011-ARC-193(16)，2011，PP.1-6

〔学会発表〕（計３件）

① Soseki Aniya, Toshiaki Kitamura, A Performance Improvement for Floating-Point Arithmetic Unit with Precision Degradation Detection, The 17th Workshop on Synthesis And System Integration of Mixed Information Technologies (SASIMI 2012), 2012 年 3 月 9 日，B-Conプラザ(大分県別府市)
②松田稔彦，北村俊明，計算精度低下を検出する PC エミュレータの開発，電気・情報関連学会中国支部第６０回連合大会，2011 年 10 月 22 日，広島工業大学
③ Keita Kaneko, Toshiaki Kitamura, Evaluation of floating-point arithmetic unit with precision degradation detection, 16th Asia and South Pacific Design Automation Conference, Student Forum, 2011 年 1 月 27 日，パシフィコ横浜

〔図書〕（計０件）

〔産業財産権〕

○出願状況（計０件）

○取得状況（計０件）

〔その他〕

ホームページ等

http://www.csl.info.hiroshima-cu.ac.jp

６．研究組織

(1)研究代表者

北村　俊明（KITANURA TOSHIAKI）
広島市立大学・情報科学研究科・教授
研究者番号：10324683

(2)研究分担者

(3)連携研究者