Linux/Alpha活用講座

清水 尚彦　nshimizu@keyaki.cc.u-tokai.ac.jp

第16回
Alphaのパフォーマンス測定(2)

Windows CEを使い始めて約1カ月たちました。その間何度かリセットボタンを押すことがあって、やっぱりWindowsだなあとおかしなところで実感しています。それでも電源を入れてすぐに使えるし、リセットしても立ち上がりが速いのはさすがですね。まめに文書をセーブしながら原稿を書いています。200LXからCE＋アイゲッティに携帯機器を変更して一番戸惑っているのは、スケジュールソフトの使い勝手の悪さです。CEもアイゲッティも200LXで使っていた1カ月分のスケジュール一覧ができないのです。正確には、カレンダーにスケジュールの内容が表示されないのです。HPのCEならソフトが追加されているようですが、CEの標準ではありません。逆に良かった点は、当然ながら大型のキーボードによる入力の便利さと、LANの接続による通常の環境との互換性です。今までファイルの転送に苦労していたのが嘘のように簡単になりました。さらに、電子メールの添付ファイルにWordやExcelをつけてくる人がいるのですが、簡単なものであればCEで見られるのも大きなメリットですね。簡単なものなら添付でなくテキストで送ればいいと思うのですが……。

先月号では、性能測定のための命令「wrperfmon」の説明をしました。この命令はプロセッサのタイプごとに異なるので、今月号でもその続きとして、21264におけるwrperfmon命令の説明をします。また、特権命令なので一般ユーザーから実行できないこの命令を使うためには、カーネルモジュールとしてドライバを登録して利用するのが一般的です。先月号ではCompaqが提供するカーネルモジュールである「Ipr」を使った基本的な性能の測定方法について紹介すると書きましたが、このモジュールはプロセッサタイプに強く依存しているので、今の最新のプロセッサには対応できません。そこで、どのプロセッサにも使用可能なモジュールを作成しようと考えておりますのでカーネルモジュールの利用方法についてはもう少しお待ちください。Linuxでは性能測定のプロセスを選択するPMEビットをサポートしていないのですが、本来はカーネルのパッチとしてPMEのサポートも追加したいところです。しかし、そこまではすぐにはできないので将来への課題とさせてください。

カーネルモジュールにプロセッサタイプの依存性を持たせなくても、どこかで依存性を解消する必要があります。しかし、モジュールで処理するよりは一般のユーザープログラムで処理したほうが、複数の世代で安定してカーネルを使えると私は考えています。モジュールのダイナミックロードがサポートされているLinuxでは大きな問題ではないかもしれませんが、機能の棲み分けについて少し考えなくてはなりません。もっとも、システムに依存する部分はユーザーに見せるべきでないという議論もあるかもしれませんが、測定項目や測定に伴う割り込みインターバルすらことなるプロセッサ世代間で、ユーザー側がプロセッサを意識せずに使える機能はほとんどないと言えます。であるならば、カーネルとプロセッサの両方の世代によって影響されるカーネルモジュールとしての実装よりも、独立したユーザープログラムのところにプロセッサ依存部分を任せたほうが実用性は高いと思います。

Alphaのニュース

AlphaLinuxのWebサイトから話題を何点か紹介します。最新ニュースはhttp://www.alphalinux.org(画面1、2)をご参照ください。

・英語でのディスカッションですが、Alphaの新しいメイリングリストが2つできました。参加したい人は「majordomo@

alphalinux.org」に、本文に「subscribe axphardware」もしくは「subscribe axp-software」と書いたメールを送ってください。

・164LXと164UXのマザーボードの生産をAPIが中止するそうです。これらのマザーボードが必要な人は、生産中止前に入手しましょうとのことです。21164のマザーボードとして活躍してきた164LXがなくなるのは寂しいですね。21164の世代の後継のボードが提供されるのか私はまだ確認していません。もしかしたらこれは21164自体の生産を縮小もしくは中止するための第一歩なのかもしれません。21264がかなりの性能を上げることが分かった以上、21164の生産を続ける必然性は少ないのでしょう。低価格向きには21164PCがあって、こちらの生産中止はアナウンスされていませんから、21264と21164PCの組でIntelに対抗していくつもりなのだと思います。

・「SRM Howto」の新版がでました。今回は新しくIDEのドライブからのブートや、Red Hat Linux 6.0やSuSE 6.1のロードの方法などが追加されました。前にも書いたのですが、NTのサポートが不要になったことで、Alphaの世界ではSRMの重要性はますます大きくなっていきます。MILOの新しいバージョンも発表されていますが、SRMが必要なデバイスにサービスを提供すればMILOを使わなくてもすみます。SRMでIDEなどの安価なデバイスからのブートができれば、多くの用途で十分な機能となるのではないでしょうか。

A Compaqのコンパイラの正式リリース

ついにFortranとCのコンパイラが正式にリリースされました。詳細は

http://www.compaq.com/linux

からたどっていただければいいのですが、本稿でも概略の紹介をしておきたいと思います。β版を使っていたユーザーの関心事は正式版の性能と価格ではないかと思います。性能については私自身がまだインストールしていないので割愛させていただきますが、価格については、特に個人ユーザーにとってはよいニュースがあります。Cコンパイラについては価格は無償になっていて、ライセンス条項を確認すればそのままダウンロードできるようになっています。ライセンス条項も、ソフトウェアの譲渡や貸し出しの禁止などが制約されているだけで、利用分野の制限や商用目的の利用についての制限は示されておりません。ですから、Webからユーザー登録すれば基本的には誰でも使えるようになります。また、Fortranに対して、Compaqは新たに「Technology Enthusiast License Terms」(技術マニアとでもいいますか)という契約条件を設定して、この条件に合致するユーザーに対しては、無償で利用できることにしました。このライセンスの元では、このコンパイラにより生成された成果物が利益を生み出すために用いられてはいけないため、実質上オープンソースのフリーソフトの開発や教育目的の開発だけに利用が制限されることになります。しかしながら、Linuxのユーザーにはこの目的で利用する人も多いと思われるので、無償で使えることに十分な意味を持っていると思います。商用にも使えるライセンスの価格は、1ライセンスで399ドルになっています。大量に導入する人は10ユーザーで1399ドルになっています。ドキュメントだけ必要な人は35ドルで購入できるようです。ただし、日本のCompaqでどのように価格をつけるのか、そもそも販売するのかは確認していません。このように、βテストされていた2つのコンパイラが無事リリースされたことと、それ以上に「太っ腹」なところを見せてくれたCompaqの姿勢に敬意を表したいと思います。

画面1

画面2

EV6(21264)における wrperfmon命令

EV6では、性能カウンタは20ビットのものが2本あります。それぞれ独立に設定したり読み書きできるようになっています。ビット数が増えたためと思いますが、割り込み頻度の設定がなくなり、オーバーフローの割り込みとカウンタの値から性能を抽出するようになります。カウンタの取得するイベントはEV5に比べて大幅に少なくなりました。性能を測定してプロセッサ動作を把握する目的からするとちょっと残念です。しかし、基本動作の把握はできるくらいのイベントは取得できるようです。プロセッサ設計者は、もっと多くの項目を知りたいはずなので、21264が本当に項目を減らしたのかドキュメントから落としただけなのかはこれだけからではよく分かりませんけれど。

wrperfmon命令には、機能コードの異なる7種類の動作があります(表1)。EV4やEV5の命令と同じく、a0のレジスタに機能コードを設定して、引数をa1のレジスタに入れて呼び出します。命令の結果はv0のレジスタに返ってきます。

ここで機能コード0、1の引数はカウンタの番号を引数のビットに対応させたビットマップになっていて、これは表2の通りになっています。このビットが1のカウンタを動作させたり止めたりします。このように、2つのカウンタを独立に制御することができます。

機能コード2の命令は、カウンタのイベントを選択します。引数のビットマップによって次のように2つのカウンタの動作を規定します(表3)。それぞれの値によって表4、5のイベントが選択されます。

機能コード3の引数は、カウンタが動作するプロセスを絞り込むために使います(表6)。ビットマップで表されます。21264はアウトオブオーダーのプロセッサなので、実行途中もしくは実行が終了しても、プロセッサ内に滞留している命令が数多くある場合があります。アウトオブオーダーのプロセッサには、プログラムに記述された順序に従って命令の処理が終了したようにソフトウェアに見せるために、プロセッサに用意されたリオーダーという機構があります。リタイアというのは、このリオーダーバッファに滞留していた命令が本来の実行の順序がきて、アーキテクチャ上定義されているレジスタに結果を書き戻す動作のことをいいます。すなわち、リタイアしたということは、ソフトウェアから見た実行が終了したことを意味しています。DTBはデータアドレスの変換バッファのことで、ITBは命令アドレスの変換バッファのことになります。非アラインとラップは、データ型により規定されたデータのアドレス空間上の配置に合っていないアクセスをしたときに発生する例外で、通常、OSで命令の実行を代行して正しい処理に変換されることが多く、ユーザーの目には見えないものですが(Linuxではコンソールにメッセージが表示されます)、性能上は大きなペナルティになるものです。リプレイトラップは、パイプラインにおいて競合条件を確定できず、予測を元に実行を進める場合に

表1

機能コード	説明
0	性能モニタのディスエーブル
1	性能モニタのイネーブル
2	モニタイベントの選択
3	モニタプロセスの制御
5	カウンタリード
6	カウンタライト
7	性能モニタのイネーブル(カウンタ設定つき)

表2

ビット	カウンタ番号
0	カウンタ0
1	カウンタ1

表3

ビット	動作
4	カウンタ0イベント選択
3:0	カウンタ1イベント選択

表4

イベント選択ビット	取得イベント
0	クロックサイクル
1	リタイア(命令終了)した命令数

表5

イベント選択ビット	取得イベント
0000	クロックサイクル
0001	リタイアした条件分岐命令
0010	リタイアした予測失敗の分岐命令
0011	リタイアしたDTBシングルミス
0100	リタイアしたDTBダブルダブルミス
0101	リタイアしたITBミス
0110	リタイアした非アラインとラップ
0111	リプレイトラップ

表6

引数のビット0の値	動作
0	全プロセスをカウンタの計数対象とする
1	PMEビットが立つプロセスのみを対象とする

表7

引数のビット位置	選択カウンタ
0	カウンタ0
1	カウンタ1

表8

引数のビット位置	選択カウンタ
0	CTR0
1	CTR1

予測が外れたときに、それまでの実行結果をキャンセルするためにプログラムカウンタを確定しているところまで引き戻し、命令読み出しから再実行するものです。21264のようなアウトオブオーダのプロセッサでのリプレイの影響がどれほどであるかは、資料からはよく分からなかったのですが、悪くするとリオーダバッファの中身を含めてキャンセルする必要があるのかもしれませんから、影響は小さいとは言えません。

機能コード5と6は、カウンタのリードライトの命令ですが、どちらもカウンタの本体は表7のビットを用います。また、機能コード7はカウンタのライトと同時にカウンタのイネーブルを行う命令になります。引数はカウンタライトと同じです。

カウンタライトの時に、2つのカウンタの書き込みを独立して行うために書き込み指示のビットを用います。書き込みは表8のビットが立っているカウンタに対してのみ行われます。

EV8の概要

マイクロプロセッサフォーラムで「21464(EV8)」が発表されました。このプロセッサは、「同時マルチスレッド(SMT)」と彼らが呼んでいる機構を実現していて、1チップで複数のプログラムスレッドが動作するようになっています。複数のスレッドには独立してプログラムカウンタやアーキテクチャレジスタが割り当てられるため、結果として1チップのマルチプロセッサのように見えます。キャッシュのアクセスのように、長いストールの間、プロセッサの動作を切り替えるマルチスレッドは、IBMのAS400を始めとしていろいろとやられていますが、EV8は同時にパイプラインに投入するという点が異なっています。これは、パイプライン中の命令スロットには実は依存性などで空いたところが多いという問題点を解決するために、パイプラインの途中のリソースを共用しながら性能を上げることができるというアイデアです。

すぐに気になるのは、マルチプロセッサでキャッシュを共有したら、1プロセッサあたりのキャッシュ容量が少なくなって性能が低下しないかということだと思います。実はキャッシュのミス率は、キャッシュ容量が実質的に半分になったからといって倍になったりはしないので、マルチプロセッサでのキャッシュ共有はそれほど大きなオーバヘッドにならないということと、たとえミス率が多少上がってしまったとしても、ミスのスレッドが待たされるだけで、他のスレッドが動いていれば、プロセッサ全体としては動作していることによって全体性能は向上するものです。なんでわざわざ同時マルチスレッドにするのかというと、実は、アウトオブオーダーのプロセッサであるEV6以降のプロセッサにはレジスタリネーミングの機構が入っています。これには物理レジスタとアーキテクチャレジスタのマッピング機構が必然的に入ってくることと、分岐予測のように投機的に実行する命令列を区別する機構が入っているということで、複数のプログラムカウンタを持つプロセッサを実現しやすい環境にあったといえます。プログラムカウンタの一部としてASN(アドレス空間番号)を扱えば、それほど無理なく複数のスレッドをアウトオブオーダーの枠組みで扱うことができるのではないでしょうか。

マイクロプロセッサフォーラムの発表では、いくつかのベンチマークを合わせて示しています。マルチプログラミング環境とマルチスレッド環境のベンチマークを見ていると、4スレッドの実装が本当に必要なものか疑問を感じる向きも多いと思います。一般に、この手の施策の改善率は導入初期(すなわちこの場合には2スレッド)における改善がもっとも大きいものですが、ご多分にもれず、本ベンチマークでも同じ傾向が見られます。SpecIntのベンチマークでは、2スレッドにした場合に70%程度の性能改善が見られるのに対して、もう1つスレッドを追加しても30%程度しか向上しません。さらにもう1つ追加して4スレッドとしても20%程度の性能改善と、スレッドの本数に比較して性能の改善率は低下していきます。4本全部では120%の向上と十分な性能が得られているように見えても、その分リソースの投入がなされているのでリソースに比較してどうかという議論が必要になります。もっともSMTの実現に必要なハードウェアの量は、アーキテクチャレジスタを除けば3スレッドでも4スレッドでもあまり変わらないので、それなら4スレッド用意すれば良いという議論はありそうです。一方、SpecFPでは、3スレッド以上では性能が飽和しているグラフとなっています。原因は発表を聞いていないので憶測に過ぎませんが、パイプラインのスロットの空きが、そもそも3スレッド程度でほぼ埋まってしまうのか、メモリ転送か命令発行の部分でボトルネックが生じて性能を律速しているものと考えられます。詳細な解析をすれば面白い結果が得られると思います。

おわりに

今回は、先月号でお約束した内容と少し変更したので、がっかりした人もいるかもしれません。ですが、よりよい内容のものを鋭意作成するつもりでおりますのでご勘弁ください。コンパイラも正式リリースされているし、CompaqやAPIが共同でAlphaの技術に対する投資をするとなっているので、今後の展開を楽しみにしているところです。