組込みシステムのための実時間性能計測手法の開発

指導教員：市川 周一　　学籍番号：063724 手塚 康瑛

1　研究背景・目的

組込みシステムとは，各種の機器に組み込まれてその制御を行うコンピュータシステムのことである．近年の組込みシステムでは，マルチタスク OS を搭載するなど多機能化が進んでいる．組込みシステムのプロセッサにはパフォーマンスカウンタを備えた品種があり，キャッシュミス等の CPU イベントを実時間で計測することが出来る．実時間性能計測はシステムの性能や動きを知り，性能向上を図る上で有用である．

本研究では，パフォーマンスカウンタを用いて CPU のイベント数を計測する手法を開発した．例としてキャッシュミス数をカウントした結果を示す．

2　実装

2.1　実装環境

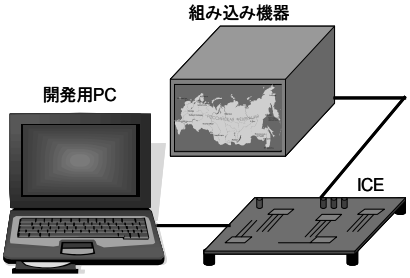

図 1: 組込みシステム開発機概要

本手法の実装には図 1 に示す実用的組込み機器 (カーナビゲーションシステム) を用いた．このシステムには，プロセッサとして MIPS 系 64bitCPU である NEC VR5500 [1] が搭載されている．このシステムに ICE (In-circuit emulator) を接続してデバグと計測を行う．ICE を PC と接続することで，CPU のブレーク，レジスタとメモリの書き換えを行うことが出来る．プログラム開発には T-Kernel [2] 用の統合開発環境を用いた．

このシステムの OS には T-Kernel が採用されている．T-Kernel は T-Engine Forum [2] が開発，配布を行っている組込み機器向けマルチタスクオペレーティングシステムである．

2.2　パフォーマンスカウンタ

パフォーマンスカウンタとは，CPU に内蔵された性能計測用カウンタで，CPU 内部の様々なイベントを計測するハードウェアである．VR5500 プロセッサには独立した 2 組のパフォーマンスカウンタが備わっており，それぞれ個別のイベントを測定することが出来る．

パフォーマンスカウンタはカウンタレジスタ (32bit) と制御レジスタ (32bit) から成り，制御レジスタでカウント対象やカウント方法を設定する．設定可能なイベントとしては，クロックサイクル数，命令実行数，LOAD/STORE 命令実行数，I/D キャッシュミス数などがある．パフォーマンスカウンタの書き込みと読み出しは特権命令によって行う．

2.3　測定手法の概要

本手法では，タスク毎に CPU イベントを計測するために，OS のタスクスイッチ部でカウンタ値を操作する．カーネルのディスパッチャが呼ばれた際に，コンテキストの退避に合わせてその時点のカウンタレジスタの内容を各タスクに対応するメモリ領域に積算する．その後，次に実行されるタスクのコンテキストを復元するのに合わせてカウンタレジスタの初期化を行う．パフォーマンスカウンタは 32bit であるが，イベント値は 64bit の値としてメモリ上の配列に積算する．このとき同時に，各タスクのディスパッチ回数を計測する．タスク毎のディスパッチ回数は，メモリ上の配列に 32bit の値として保存する．これら処理を行うプログラムを T-Kernel 内の cpu_support.S にアセンブリ言語で埋め込んでいる．

測定の開始と終了は ICE を用いてシステムをブレークさせることで行う．測定の初期化と計測値の取得は，ICE によってシステムのメモリを開発用 PC 上から操作することで行う．

3　測定例

開発したツールを用いて試験的測定を行った．ここでは測定の例としてシステムの命令キャッシュミス率を示す．測定条件は次の 4 つである．

状態 A　モード (1) の定常負荷時
状態 B　モード (2) の定常負荷時
状態 C　モード (1) で操作 (処理要求) を行った場合
状態 D　モード (2) で操作 (処理要求) を行った場合

命令キャッシュミス率は，命令実行数と命令キャッシュミス数を計測することによって求められる (式 1).

$$\text{命令キャッシュミス率} = \frac{\text{命令キャッシュミス数}}{\text{命令実行数}} \qquad (1)$$

組込みシステムは外部割り込みによって動作するため，タスクディスパッチの順序や回数が不定である．そのため各状態に対して命令キャッシュミス数，命令実行数の測定を 3 回行い，平均値を採用した．図 2 は状態 A から D の命令キャッシュミス率である．また，図 3 は図 2 の D 状態におけるタスク毎キャッシュミス率である．

このように，タスク毎の実時間性能計測が可能であることを確認した．例えばキャッシュミス率を計算することで，大きく複雑なプログラムをタスク毎に解析し，動作速度に問題のあるタスクを解析することが出来る．更に，動作速度に問題のあるタスクを解析することで効率的なプログラムのチューニングが可能となる．

本手法では図 3 に示した命令キャッシュミス率の他に，データキャッシュミス率や分岐予測ミス率を計測することが可能である．また，クロックサイクル数と命令実行数から CPI 値や CPU のビジー率等を求めることが出来る．

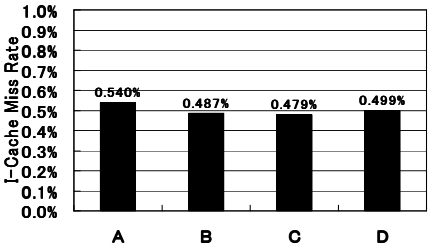

図 2: 状態の違いによる命令キャッシュミス率

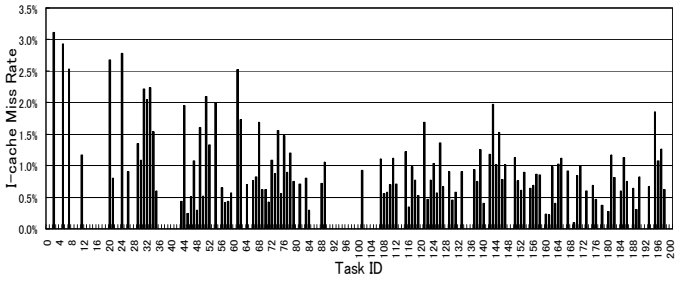

図 3: 状態 D でのタスク毎命令キャッシュミス率

参考文献

[1] NEC: ユーザーズ・マニュアル VR5500A, VR5532A (2003).

[2] T-Engine Forum：T-Engine フォーラム, http://www.t-engine.org