Vol.2014-MBL-72 No.14
Vol.2014-CDS-11 No.10
2014/8/29

CPU 使用率とメモリ帯域使用率を考慮した性能予測手法

若林昇[†1]　　吉岡信和[†2]

デジタルテレビを筆頭にコンシューマ機器は年々高機能化の一途にある．一方で，価格面での競争激化の為，コストダウンは必須となっており，高性能な CPU，潤沢なメモリは期待できない．従って，現行の性能を限界まで引き出す必要があり，その為には，定量的な性能測定と性能予測手法が必要になる．本論文では，CPU 使用率とメモリ帯域使用率を考慮した性能予測手法を提案する．

A Performance prediction technique in consideration of CPU utilization and memory band utilization

NOBORU WAKABAYASHI[†1]　NOBUKAZU YOSHIOKA[†2]

Consumer products, such as Digital-TV, the consumer products have been becoming high functionality year by year. On the other hand, the cost reduction is required because of the competition intensification on the price side, without using rich specification hardware(CPU, memory, etc). Thus, it is needed to draw current performance to the limit. After all, the performance prediction technique is required. This paper proposes a performance prediction technique in consideration of CPU utilization and memory band utilization.

1. はじめに

コンシューマ機器など組込み機器の多くは，年々高機能化の一途にあり，新機能が追加される．一方で，価格面での競争激化のため，コストダウンは必須となっている．ソフトウェアの観点からは, より高性能な CPU やより大容量なメモリであれば，新機能を追加する際に性能面を考慮する必要がなくなる．しかし，コストアップになるこれらのハードウェアの追加をしないために，新機能と既存機能の同時動作時の性能予測値による見積もりが必要になる．また，開発の後工程で性能問題が発覚すると大きな手戻りになるため，性能見積もりは新機能開発時の早い段階で必要になる．新機能開発時の早い段階では，既存の機能と同時に動作させることは困難になるため，実際に同時動作させて性能測定することはできない．

従来, 同時動作時の性能予測値は各機能の CPU 使用率を加算して算出していた．しかし，デジタルテレビなど映像を扱うような機器では高画質映像などの多量のデータを扱う場合，メモリ帯域を多く使用する．このような映像を扱う機器で，バスアービタ等によるバスの調停機構がある機器の場合, デコーダなど CPU 以外の機器がバスを優先使用することがあり. この場合 CPU はメモリアクセスを待たされるため，CPU 使用率が増えるように見える．すなわち，CPU 使用率とメモリ帯域使用率には因果関係があるといえる．性能予測の際は，メモリ帯域使用率も考慮した性能予測が必要になる．

メモリ帯域使用率を考慮した CPU 使用率の算出方法として，メモリストール時間を計測し，CPU 使用率を算出する方法があるが, メモリストール時間を算出するためには, キャッシュミス回数を計測する必要がある．そして，キャッシュミス回数を計測するためには，CPU にパフォーマンスカウンタが必要になる．しかし，低コストを要求されるような機器では，CPU にパフォーマンスカウンタが付いていないことがあり，メモリストール時間を用いたこのような従来手法を用いることができない．

本論文では，パフォーマンスカウンタが付いていない機器でも，チップセレクト信号の計測からメモリ帯域使用率測定し，CPU 使用率との関係を導き出すことで，同時動作時の性能予測を行う手法について提案する．

続く 2 章では，従来手法となる関連研究とその課題について述べ，3 章で本研究の対象となる機器構成について説明する．4 章では，従来手法の課題を解決する手法について提案し, 5 章で提案手法を評価する．6 章で提案手法に関する議論を行い，最後に 7 章でまとめる．

2. 関連研究

システムの性能予測に関連する研究として，ハードウェアとソフトウェアの協調シミュレータを用いたハードウェア開発環境の研究[1][2]等がある．これらの研究により，ハードウェア実装前にシステム全体の性能予測が可能となっ

†1 (株)日立製作所
Hitachi Ltd.
†2 国立情報学研究所
National Institute of Informatics

Vol.2014-MBL-72 No.14
Vol.2014-CDS-11 No.10
2014/8/29

た．しかし，コンシューマ機器では，製品開発中にハードウェアが変更されることが多く，このようなシミュレータは最終的な製品に搭載されるハードウェアをシミュレートすることは少なく，最終製品の性能予測ができるとは言い切れない．

また，性能予測に関連する他の研究として，プログラムコード抽象化手法に基づくシステムの性能評価環境 PSI-NSIM[3]がある．これはインターコネクトのシミュレーションを行うことで，システム全体の性能を高速かつ精度良く見積もり，大規模システムの効果的な性能解析支援や可視化の実現するものである．しかし，このような大規模システムのシミュレーションは，コンシューマ機器のような低コストが要求される機器には適用が難しい．

モデル検査を用いた性能予測に関する研究として，時間制約の検証ツールや確率的モデル検査ツールを用いた研究がある[4][5][6][7]．これらは，現在最も普及している性能モデル検証ツールである UPPAAL[8]や確率的振る舞いを持つシステムに対するモデル検査ツール PRISM[9]を用いて，実時間ネットワークシステムの詳細なモデルからシステム全体の性能解析を行うものである．しかし，このようなモデル検査ツールを用いた性能予測では，どのようにモデル化するかが重要であり，メモリ帯域使用率と CPU 使用率の関係をモデル化する必要があるが，これらの関係については述べていない．

CPU 使用率とメモリ帯域使用率の関係に関する研究として，バス調停に関する研究やメモリウォール問題に関する研究などがある．

バス調停に関する研究では，ラウンドロビン方式によるバス調停方法[10]，静的固定優先度方式によるバス調停方法[11]，TDMA によるバス調停方式[12]，TDMA とラウンドロビンを組み合わせた方法によるバス調停方式[13][14]，LOTTERYBUS 方式によるバス調停方式[13][14]，Slack-based Bus 方式によるバス調停方式[12]等がある．しかし，これらの研究には性能予測に関して，メモリ使用率と CPU 使用率を用いた性能予測方法については提示されていない．

メモリウォール問題に関する研究では，与えられたハードウェア資源制約下においてこれらを最大限に有効活用し，システム全体の性能を向上する提案[15]がある．また，動的にキャッシュラインサイズを変更しシステムの性能を向上する提案[16]がある．これらの提案の中ではメモリストール時間を考慮したプログラムの実行時間（CPU 使用率）の算出方法が提示されている．メモリストール時間を算出するためには，キャッシュミス回数を計測する必要があり，キャッシュミス回数を計測するためには，CPU にパフォーマンスカウンタが必要になる．

また，パフォーマンスカウンタを用いた性能予測手法の研究[17]がある．これは，あらかじめ多数のプログラムを実行し，統計的な学習を行い，あるパフォーマンスカウンタの値と性能の間の関係を回帰分析により求める，定量的な手法である．しかし，コンシューマ機器のような低コストを要求されるような機器では，CPU にパフォーマンスカウンタが付いていないことがある．この場合，このような従来手法を用いることができない．

3. 対象機器構成

本研究における性能評価の対象となる機器構成の例を図 1 に示す．

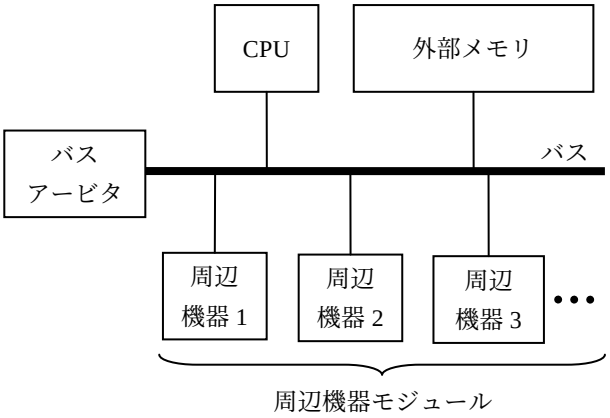

図 1 対象機器構成例

図 1 で示すとおり，RAM など外付けの外部メモリと CPU，それ以外の例えばデコーダやネットワークコントローラ(Network Interface Controller：NIC)，HDD や SSD と言った２次記憶装置など数多くの周辺機器モジュールがバス上で接続されており，各モジュールからのメモリアクセス要求をバスアービタが調停する構成になっているモデルを対象としている．このように周辺機器がバスに接続するモデルは一般的であり，またバス調停を行うバスアービタが接続される構成については，映像などの多量のデータがバス上を通信する機器で多く見ることができる．

4. CPU 使用率とメモリ帯域使用率を考慮した同時動作性能予測手法

4.1 概要

本章では，パフォーマンスカウンタが付いていない機器でも，メモリ帯域使用率を考慮した同時動作時の性能予測ができる手法について述べる．

パフォーマンスカウンタが付いていない機器でも，図 1 で示した対象モデルの通り，DDR2-SDRAM 等外部メモリは接続されている．

提案手法では，まず，SDRAM のチップセレクト信号の単位時間あたりのパルス数（周波数）を計測し，理論的な

Vol.2014-MBL-72 No.14
Vol.2014-CDS-11 No.10
2014/8/29

最大値との比率からメモリ帯域使用率を算出する．また，メモリ帯域への負荷を変えて，CPU 使用率を計測し，関連性を算出する．更に，各単体機能の CPU 使用率及びメモリ帯域使用率の測定値と，基準点の CPU 使用率及びメモリ帯域使用率の測定値から，算出した CPU 使用率とメモリ帯域使用率の関連性を考慮する．これにより，各機能が同時動作した際の予測値を算出する式を提案する．

4.2 チップセレクト信号を用いたメモリ帯域使用率測定方法

4.1 節で述べたように，性能予測する際は，メモリ帯域使用率まで考慮する必要がある．本節では，メモリ帯域使用率の測定方法について述べる．

本論文の対象機器構成において，DDR2-SDRAM 等外部メモリに繋がるバス上には，CPU の他にもデコーダやネットワークコントローラ(Network Interface Controller : NIC), HDD や SSD と言った２次記憶装置など数多くのモジュールが接続されており，各モジュールからのメモリアクセス要求をバスアービタが調停する構成になっている．しかし，どのモジュールがどれだけバスを占有しているのかリアルタイムに測定する仕組みは用意されていないことが多い．このため，従来は机上で見積った理論的な最大値などを積み重ねることでワーストケースでの性能を予測していた．

しかしこの方法では，アービタの調停動作などを含めて実際にどう動いているのかの挙動把握をすることはできず，したがって，平均値を知ることもできなかった．

そこで，リアルタイムにメモリ帯域使用率を測定する為に，本手法では，DDR2-SDRAM へ出力しているチップセレクト信号の単位時間あたりのパルス数（周波数）によってメモリの負荷（使用率）を近似する．これは，チップセレクト信号を周波数ドメインアナライザなどで観測することにより，周波数の動的な変化と平均値をリアルタイムに計測するとともに，チップセレクト信号の周波数の理論的な最大値との比率からメモリバンド幅の使用率を算出する手法である．図 2 にデジタルテレビにおける視聴機能と録画機能を同時に動作させた時のチップセレクト信号パルスの周波数の変化の一例を示す．横軸は時間(ms)，縦軸は周波数(MHz)である．

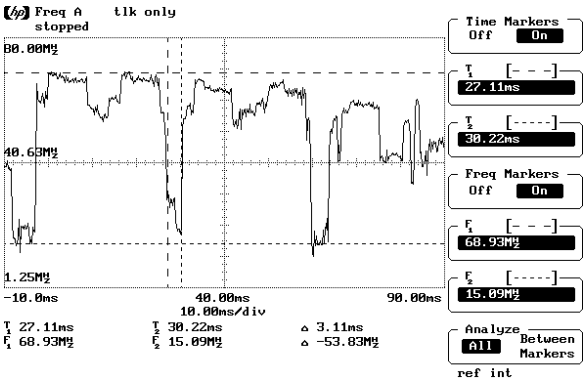

図 2 デジタルテレビにおける視聴と録画の同時動作時のチップセレクト信号パルス周波数変化

DDR メモリでは，周期的なリフレッシュが必要となる．このリフレッシュ期間は他のメモリアクセスは実行できず実質的にはバンド幅負荷となっている．しかしながら，チップセレクトの周波数にはこのリフレッシュの影響はほとんど反映されない為，その分を補正する必要がある．これらを考慮すると，チップセレクトの周波数の平均値を F_{mean}[MHz]，理論的な最大値を F_{max}，単位時間当たりのリフレッシュ期間を R[%]としたときのメモリ帯域使用率 U_m[%]を（数式 1）で算出する．

$$U_m[\%] = (F_{mean}\ [MHz] + F_{max}\ [MHz] \times R[\%]) / F_{max}\ [MHz] \times 100 \quad \ldots \quad \text{（数式 1）}$$

4.3 CPU 使用率とメモリ帯域使用率の関係

本節ではメモリ帯域使用率が CPU 使用率に及ぼす影響について述べる．4.2 節でも述べたように，DDR はリフレッシュ動作を行っている．この周期は設定により変更することができるので，リフレッシュ周期を意図的に変えることにより，ダミーのメモリバンド負荷として利用できる．

このようにしてダミー負荷の量を変えながら CPU 使用率を測定することで，メモリ帯域使用率と CPU 使用率の関係を導き出すことが可能である．なお CPU 使用率の測定は，各種ツール[a]があるのでそのツールを用いればよい．

デジタルテレビに対して，リフレッシュ周期を意図的に変えてメモリ帯域に負荷を与えた時の CPU 使用率の測定結果のグラフを図 3 に例示する．

図 3 のグラフの縦軸は単位時間あたりのリフレッシュ期間 1%の時の CPU 使用率を 1 とした時の CPU 使用率の比率であり，横軸はリフレッシュ頻度を変化させた時に（数式 1）で求められるメモリ帯域使用率である．

a vmstat や top といったオープンソースソフトウェアのコマンドがある

Vol.2014-MBL-72 No.14
Vol.2014-CDS-11 No.10
2014/8/29

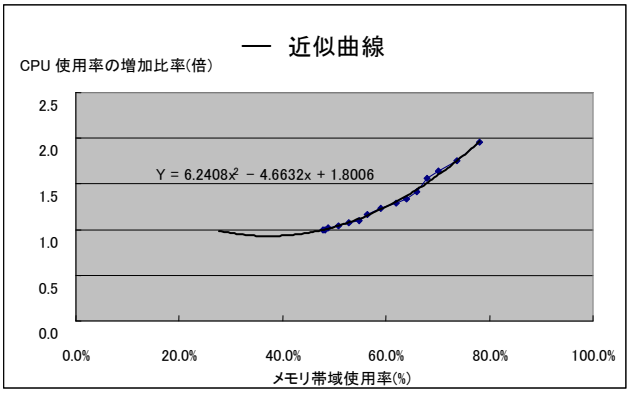

図 3　バンド幅の増加分が CPU 負荷率に与える影響

本手法では，測定した値をグラフにプロットし，近似曲線を求める．図 3 の例では 2 次曲線で近似でき，CPU 使用率の増加比率 Y はメモリ帯域使用率 x [%]から以下の（数式 2）の近似式が得られる．（数式 2）においてそれぞれの係数は，a = 6.2408, b = －4.6632, c = 1.8006 となる．なお，（数式 2）および a,b,c の係数は例である．プロットした値によっては a,b,c の係数は変わり，また 2 次曲線ではなく他の曲線で近似することもあり得る．

$$Y = f(x) = ax^2 + bx + c \quad \ldots \quad \text{（数式 2）}$$

図 3 のグラフからわかるように，同じソフトウェアを同じ CPU で実行していても，メモリ帯域が約 70%の使用率になると CPU 使用率が約 1.5 倍になり，メモリ帯域が約 80%になると CPU 使用率が 2 倍となる．これはメモリ帯域に負荷がかかるため，CPU がメモリへのアクセスを待たされる結果，CPU が動けない，すなわち，ある処理を行うのに時間がかかってしまい，メモリ帯域の負荷が高いほど，CPU 使用率が高くなるということである．

このように，CPU 使用率とメモリ帯域使用率は依存関係にある．その為，純粋な CPU 使用率がどれくらいあるのかは，CPU 使用率測定ツールから得られた値よりメモリ帯域負荷による影響部分を排除しなければならない．

4.4 CPU 使用率予測値とメモリ帯域使用率予測値の算出方法

本節では，複数機能の同時動作時における CPU 使用率及びメモリ帯域使用率の予測値を算出する方法について述べる．

4.3 節でも述べたように，CPU 使用率測定ツールから得られた値よりメモリ帯域負荷による影響部分を排除しなければならない．まずは，このメモリ帯域負荷の影響を取り除いた CPU 使用率の計算方法について述べる．

多くの CPU 使用率測定ツールは，システム全体あるいはプロセス単位の CPU 使用率が測定される．その為，ある単体機能の純粋な CPU 使用率を測定したい場合，まずは基準となる機能の CPU 使用率及びバンド帯域使用率を測定する必要がある．本論文では，デジタルテレビに対して，デジタル番組視聴中でかつ，バスのアービタの設定を CPU 最優先とした場合の値をデジタルテレビの例における基準点とするようにした．ここで，基準点の CPU 使用率を U_{cb}[%]とし，ある単体機能の CPU 使用率を U_{cx} [%]，メモリバンド幅使用率を U_m [%]とすると，該単体機能の純粋な CPU 使用率 U_{cp} [%]は （数式 3）で算出する．第 1 項は CPU 使用率測定ツールから得られた CPU 使用率からメモリ帯域使用率の影響を反映した CPU 使用率になり，これに第 2 項である基準機能の CPU 使用率を差し引く．

$$U_{cp}\,[\%] = U_{cx} / f(U_m) - U_{cb} \quad \ldots \quad \text{（数式 3）}$$

また，基準点のメモリバンド幅使用率を U_{mb} [%]とすると，この単体機能のメモリバンド幅使用率 U_{mp} [%]は，CPU 使用率からの影響はないため，（数式 4）のように単純に差分を取ることで算出する．

$$U_{mp}\,[\%] = U_m - U_{mb} \quad \ldots \quad \text{（数式 4）}$$

次に，これらの数式を用いた複数機能の同時動作時の性能予測値算出方法について述べる．本手法では，以下の手順で算出する．

(1) 単体機能の CPU 使用率及びメモリ帯域使用率を測定（CPU 使用率測定ツール及び 4.2 節のメモリ帯域使用率測定方法にて計測）
(2) 基準点の CPU 使用率及びメモリ帯域使用率を測定（CPU 使用率測定ツール及び 4.2 節のメモリ帯域使用率測定方法にて計測）
(3) (1)(2)で得られた値を，(数式 3)，(数式 4)に代入して，単体機能の純粋な CPU 使用率及びメモリ帯域使用率を算出
(4) (1)の機能の組み合わせについて，(3)で算出した値から，同時動作時の CPU 使用率及びメモリ帯域使用率を算出

(4)において，ある機能 1 の純粋 CPU 使用率を U_{cp1} [%]，単純メモリ帯域使用率を U_{mp1} [%]，別の機能 2 の純粋 CPU 使用率を U_{cp2} [%]， 単純メモリ帯域使用率を U_{mp2} [%]，基準点の CPU 使用率を U_{cb} [%]，メモリ帯域使用率を U_{mb} [%]とすると，機能 1 と機能 2 の同時動作時のメモリ帯域使用率 U_{mt} [%]及び CPU 使用率 U_{ct} [%]は，それぞれ以下の（数式 5），（数式 6）のように算出する．

$$U_{mt}\,[\%] = U_{mb} + U_{mp1} + U_{mp2} \quad \ldots \quad \text{（数式 5）}$$

Vol.2014-MBL-72 No.14
Vol.2014-CDS-11 No.10
2014/8/29

$$U_{ct}\,[\%] = (U_{cb} + U_{cp1} + U_{cp2}) \times f(U_{mt}) \quad \ldots \text{（数式 6）}$$

5. 評価

4 章で提案した同時動作性能予測手法について，妥当性を評価する. HDD への録画機能が搭載されたデジタルテレビの実機に対して，提案手法を用いた同時動作時の性能予測を行った．対象となるデジタルテレビのブロック図の概要を図 4 に示す.

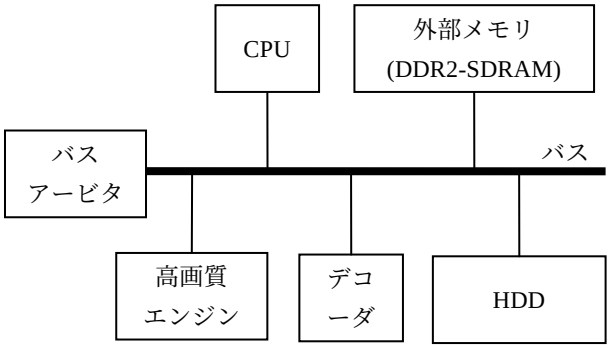

図 4　評価対象となるデジタルテレビのブロック図(概要)

DDR2-SDRAM 等外部メモリに繋がるバス上には，CPU の他に, MPEG2 や H.264 の放送波圧縮画像データをデコードするデコーダ，画質を高画質化する高画質化エンジン，録画データなどを記憶する HDD など，主に画像処理に関わるハードウェアモジュールが接続されており，各モジュールからのメモリアクセス要求をバスアービタが調停する構成になっている.

このような構成のデジタルテレビにおいて，同時動作を行う対象機能を下記に示す.

- 放送波表示
- 放送番組の HDD 録画
- 録画番組の HDD 再生

これらの各単体機能に対して，4.4 節で述べた手順(1)～(3)の結果得られる基準点から比較した純粋 CPU 使用率とメモリ帯域使用率を表 1 に示す．なお，基準点は，4.4 節でも示した通り，デジタル番組表示中でかつ，バスのアービタの設定を CPU 最優先（通常はデコーダ優先）とした場合の値を基準点とするようにした.

表 1　対象機能の CPU 使用率及びメモリ帯域使用率(基準点からの差分)

機能	CPU **使用率** [%]	**メモリ帯域使用率** [%]
放送波表示	3.9	3.0
HDD **録画**	17.7	0.0
HDD **再生**	8.1	1.8

これらの機能について，下記の組合せで同時動作させた場合の，提案手法による予測値と，実際に同時動作させて得られた測定値を比較することで，提案手法の妥当性を検証する.

- 同時動作（1）:「放送波表示」＋「HDD 録画」
- 同時動作（2）:「HDD 録画」＋「HDD 再生」

上記同時動作（1），(2）に対する提案手法（4.4 節で述べた手順(4)）による予測値と，実機による測定値を表 2 に示す.

表 2　同時動作予測方法の検証

同時動作	**提案手法(予測値)**		**測定値**	
	メモリ帯域使用率 [%]	CPU **使用率** [%]	**メモリ帯域使用率** [%]	CPU **使用率** [%]
放送波表示＋ HDD **録画**	49.6	68.4	49.6	65.6
HDD **録画＋** HDD **再生**	48.4	71.5	48.1	72.9

また，提案手法による予測値と実際の測定値との誤差（絶対誤差及び相対誤差）を表 3 に示す.

表 3　同時動作予測方法の検証（誤差）

同時動作	**絶対誤差**		**相対誤差**	
	メモリ帯域使用率 [%]	CPU **使用率** [%]	**メモリ帯域使用率** [%]	CPU **使用率** [%]
放送波表示＋ HDD **録画**	0	2.8	0	4.1
HDD **録画＋** HDD **再生**	0.3	1.4	0.6	2.0

表 3 に示したとおり，絶対誤差で見ると，メモリ帯域使用率及び CPU 使用率ともにほぼ同じ値になっており, 相対誤差で見た場合でも，0～4.1% であり，性能的にはほぼ同等といえる．なお，経験上 10%以上値が離れるとユーザ操作に影響が及ぶ．これにより，提案する同時動作の性能予測手法は妥当であると考える.

6. 議論

6.1 提案手法の一般性

5 章の評価では，同時動作時の実機による測定値をとる必要があったため，既に同時動作ができる実装済みの機能で評価したが，本手法を用いると，新機能開発時などの新機能など，実際に同時動作させることができなくても，既にある環境の測定値と，新機能単体の動作が可能な環境での測定値があれば，4.4 節で述べた手順により，同時動作時の予測値を得ることができる．

また，本論文では，デジタルテレビを例にしたが，図 1 で示した対象モデルで, 下記の特徴を持つ機器に対しては，提案手法は有効であると考える．

- バスアービタ等によるバスの調停機構がある
- バス上で多量のデータを扱う（バスの負荷が高い）
- パフォーマンスカウンタがない
- SDRAM メモリ(チップセレクト信号)を使用

このような特徴を持つ機器としては，映像を扱う機器で多く見ることができる．例えば，評価対象としたデジタルテレビ以外でも，ビデオや液晶プロジェクタ，スキャン機能を有するマルチファンクションプリンタなどのコンシューマ機器に適用できる. また, コンシューマ機器以外でも，CT スキャンや MRI などの高画質データを扱う医療系機器でも適用可能であると考える．また，今後 M2M(Machine to Machine)や IoT(Internet Of Things)と言った世界で用いられる組込み機器でも，ビックデータを扱う場合は多量のデータがバスを使用するため，本提案手法を適用できると考えており，このような組込み機器は今後増えてくると予想している．

図 1 で示した対象機器構成例では, 周辺機器が３つの場合であったが，本手法では，メモリ帯域を実測するため，３つ以上の場合であっても適用できると考えている. また，図 1 で示した対象機器構成例では, CPU が１つの場合であったが，２以上のマルチコア構成でも本手法の基本的な考え方は適用できると考えている．ただし，この場合，各コアからのチップセレクト信号をどのように考慮するかを検討する必要がある．

6.2 提案手法の限界

本提案手法による性能予測値の使い方には留意が必要である．性能予測値が 100%を超えないと問題ないように思えるが, 適用対象の機器によっては，100%に達していなくても，ある閾値を超えると，操作性や応答性に影響が出る場合がある．

例えば，デジタルテレビの場合，CPU 使用率が高くなると，リモコンによるチャンネル切替や音量調整操作に対する反応が遅くなることがあり，また，メモリ帯域使用率が高くなると，ブロックノイズなどの映像破綻が発生する可能性がある．

その為，同時動作の可否を判断する際は，これらの不具合が出ないレベルの閾値を設け，その閾値を超えないことで，判断する必要があると考える．なお，この閾値の設定方法については過去の経験によるところが大きく，人手によって設定する必要がある．

7. おわりに

7.1 結果

パフォーマンスカウンタが付いていない機器でも，メモリ帯域使用率を考慮した同時動作時の性能予測ができるチップセレクト信号を用いた性能予測手法について提案した．

提案手法では，チップセレクト信号からメモリ帯域使用率と CPU 使用率の関連性を算出し，更に，各単体機能の CPU 使用率及びメモリ帯域使用率の測定値と，基準点の CPU 使用率及びメモリ帯域使用率の測定値から，各機能が同時動作した際の予測値を算出する式を提案した．

また, HDD への録画機能が搭載されたデジタルテレビの実機に対して，提案手法を用いた同時動作時の性能予測を行った．提案手法による性能予測値と，実際に測定した値と比較し，ほぼ同じ値になることを確認し，本提案手法の有効性を確認した．これにより，以下の結果を得た．

- チップセレクト信号によるメモリ帯域使用率の算出方法は複数機能同時動作時の性能予測に有効である
- リフレッシュ周期を変えたときのメモリ使用率と CPU 使用率の実測値から得られる近似式で関係性を導き出し, 複数機能同時動作時の性能予測に用いることは有効である
- 基準点の CPU 使用率及びメモリ帯域使用率を計測し，各機能の純粋な CPU 使用率及びメモリ帯域使用率を算出し, 複数機能同時動作時の性能予測に用いることは有効である
- パフォーマンスカウンタが付いていない機器でも，チップセレクト信号を用いることで, メモリ帯域使用率を考慮した同時動作時の性能予測が可能である

7.2 今後の課題

6.2 節で述べた通り，本提案手法による性能予測値の使い方には留意が必要であり，操作性や応答性に影響がないかを判断する閾値を設けることが実際には必要であることを述べた．この閾値は，現状，人手によって設定する必要がある. 経験に基づく人手の設定では, 属人的になるため，自動化することが望ましいと考えているので，閾値設定の自動化が今後の課題である．

デジタルテレビの例では，リモコンの操作キューの蓄積具合等から応答性や操作性を数値化し，性能予測方法に反映させることや，バッファーオーバーフローやアンダーフローのエラー回数などから映像破綻について数値化し，性能予測方法に反映させることが今後の課題である．

また他の方法として，経験による閾値をデータベース化し，類似機能に対する閾値をデータベースから取得するなどの，閾値取得の自動化が今後の課題になる．

参考文献

1) B. Juurlink, I. Antochi, D. Crisu, S. Cotofana, and S. Vassiliadis, GRAAL: A framework for low-power 3D graphics accelerators, IEEE Comput. Graph. Appl., vol.28, no.4, pp.63-73, 2008.
2) D. Crisu, S. Cotofana, and S. Vassiliadis, A hardware / software co-simulation environment for graphics accelerator development in ARM-based SOCs, Proc. 13th Annual Workshop on Circuits, Systems and Signal Processing, ProRISC, 2002.
3) 柴村英智，薄田竜太郎，本田宏明，稲富雄一，于雲青，井上弘士，青柳睦：PSI-NSIM: 大規模並列システムの性能解析に向けた並列相互結合網シミュレータ，信学技報，CPSY2007-32 (2007).
4) Gerd Behrmann, Alexandre David, Kim Guldstrand Larsen, Paul Petterson and Wang Yi, Developping UPPAAL over 15 years, Software: Parctice and Experience, Vol. 41, Issue 2, pp.133-142, 2011.i
5) Sebastian Kupferschmid, Martin Wehrle, Bernhard Nebel and Andreas Podelski, Faster Than Uppaal?, CAV 2008, pp.552-555, 2008.
6) Goran Frehse, Colas Le Guernic, Alexandre Donze', Scott Cotton, Rajarshi Ray, Olivier Lebeltel, Rodolfo Ripado, Antoine Girard, Thao Dang and Oded Maler, SpaceEx: Scalable Verification of Hybrid Systems, CAV 2011, pp.379-395, 2011.
7) 伊藤明彦，長岡武志，岡野浩三，楠本真二：確率的モデル検査ツールを用いた実時間ネットワークシステムの検証手法の提案およびネットワークシミュレータ NS-2 との比較，電子情報通信学会技術研究報告. SS，ソフトウェアサイエンス，109(170) ，pp37-42, 2009 .
8) Alexandre David, Kim Guldstrand, Larsen, Axel Legay, Marius Mikucionis and Zheng Wang, Time for Statistical Model Checking of Realtime Systems, Proceedings of the 23rd International Conference on Computer Aided Verification (CAV'11), 2011.
9) M. Kwiatkowska, G. Norman and D. Parker, PRISM 4.0: Verification of Probabilistic Real-Time Systems, CAV, pp. 585–591, 2011.
10) A.S. タネンバウム：OS の基礎と応用　設計からの実装，DOS から分散 OS Amoeda まで，ピアソン・エデュケーション，東京，1995.
11) C.L. Liu and J.W. Layland, Scheduling algorithms for multiprogramming in a hard-real-time environment, J. Association for Computing Machinery, vol.20, no.1, pp.46-61, Jan. 1973.
12) M. Jun, K. Bang, H. Lee, N. Chang, and E. Chung, Slack-based bus arbitration scheme for soft real-time constrained embedded system, Proc. 12th Asia and South Pacific Design Automation Conference(ASPDAC2007), pp.159-164, 2007.
13) K. Lahiri, A. Raghunathan, and G. Lakshminarayana, LOTTERYBUS: A new high-performance communication architecture for system-on-chip designs, Proc. 38th Design Automation Conference(DAC2001), pp.15-20, 2001.
14) C.H. Pyoun, C.H. Lin, H.S. Kim, and J.W. Chong, The efficient bus arbitration scheme in SoC environment, Proc. 3rd IEEE International Workshop, pp.311-315, June-July 2003.
15) 林徹生，今里賢一，井上弘士，村上和彰：演算/メモリ性能バランスを考慮した CMP 向けオンチップ・メモリ貸与法の提案，情報処理学会研究報告. 組込みシステム，Vol.1，pp.59-64(2008).
16) 井上弘士，甲斐康司，村上和彰： DRAM/ロジック混載 LSI の高オンチップ・メモリバンド巾を活用する動的可変ラインサイズ・キャッシュ方式の提案，信学技報, pp.109-116(1998)..
17) 金井遵，佐々木広，近藤正章，中村宏，天野英晴，宇佐美公良，並木美太郎：性能予測モデルの学習と実行時性能最適化機構を有する省電力化スケジューラ，情報処理学会論文誌，コンピューティングシステム，49，pp.20-36 (2008).