Vol. 22 No. 3　　情報処理学会論文誌　　May 1981

命令語用とデータ語用とに分割したキャッシュの実用性の一検討†

齋藤　将人††

高度にパイプライン化された超高速中央処理装置 (CPU) ではマシンサイクル当り命令とデータフェッチの二つのキャッシュアクセスが必要となる．その実現手段の一つとして命令語用とデータ語用とに分割したキャッシュ (分割型キャッシュ，SCM) が考えられる．本論文は SCM の実現性を検討するため，SCM のミスヒット率と論理遅延時間を検討し，キャッシュの論理遅延時間が CPU マシンサイクル短縮のボトルネックになっているときの CPU 平均命令実行時間を考察した．それによると，SCM はそれと等容量の従来型キャッシュ (共用型キャッシュ，CCM) にくらべ若干のハードウェア量増となるが，命令語用とデータ語用とを等容量にした SCM はソフトウェア特性の広汎な変化に対しても CCM と平均的に遜色のないミスヒット率が得られる一方，キャッシュがマシンサイクル時間短縮のボトルネックになっているとき SCM は CCM よりマシンサイクル時間を短縮できるため CPU 性能が向上する．

1. まえがき

主記憶 (MM) のバッファ記憶としてキャッシュ (CACHE)[3] が IBM 360 に導入されて以来，一般の中央処理装置 (CPU) に広く採用されてきている．キャッシュは，一般に，MM の情報の写しをブロック単位でもっているデータアレー (DA) とそのブロックの MM 上の実番地を示しているアドレスアレー (AA) で構成されている (DA をキャッシュと呼んでいる場合もある)．キャッシュ上のこのブロック収容位置をブロックセルと呼ぶことにする．MM の論理番地を実番地に変換する TLB (Translation Lookaside Buffer)[4] との接続関係も含めて，キャッシュは図1の (a)～(d) のいずれかの論理ブロック構造をとるのが普通である[4]．今日までの実用化されたキャッシュでは，どのブロックセルもそこに収容される情報が命令語かデータ語かを区別していない．このタイプを共用型キャッシュメモリ (Common Cache Memory, CCM) と呼ぶことにする．

制御がパイプライン化された CPU では，命令フェッチ，命令デコード，オペランドアドレス生成・変換，オペランドフェッチ，演算の各動作が並列に実行される．基本命令でのこれらの動作は，たとえば M 200 H[4),5] のような超高速 CPU では1マシンサイクルで行われるため，命令フェッチとオペランドフェッチの2つのキャッシュアクセスが1マシンサイクル内で同時に実行される必要がある．すなわち，超高速 CPU 実現のための一つの基本的要請としてキャッシュサイクル時間を実効的に (1/2) マシンサイクル時間にすることが必要である．この実現手段の一つに M 200 H にみられるように CCM を時分割で使用する方式があるがこれはキャッシュに使用するテクノロ

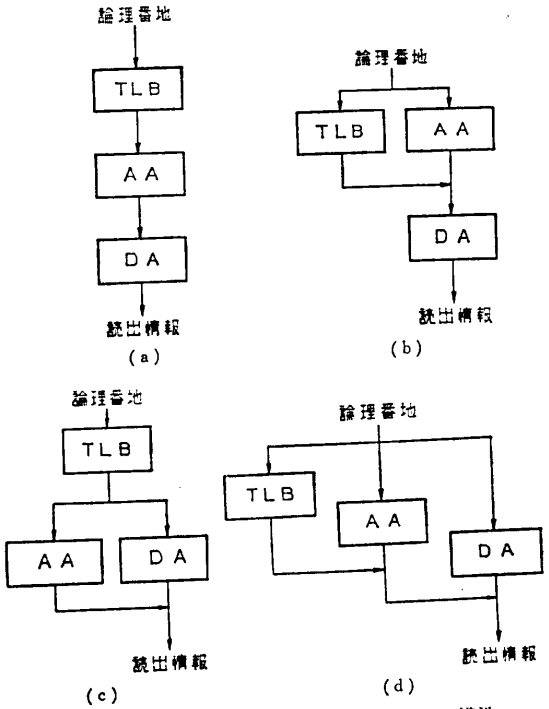

図1　キャッシュ，TLB の論理ブロック構造

Fig. 1　Logical block diagrams of cache memory with TLB.

† A Feasibility Study of a Separate Cache Memory System Divided into Two Parts for Instructions and for Data by Masato Saito (Computer Engineering Division, Nippon Electric Co., Ltd.).

†† 日本電気(株)コンピュータ技術本部

ジに高い性能を要求する．別な手法にキャッシュメモリを命令語専用とデータ語専用とに分割する空間分割方式が考えられ，これはテクノロジに特別高い性能を要求しない．後者の方式を分割型キャッシュメモリ (Separate Cache Memory, SCM) と呼ぶことにする．

飯塚は[6] SCM を用いた場合の主記憶の実効アクセス時間の定式化を試みているがキャッシュのミスヒット率等を未知数としており SCM の有効性に結論を与えていない．筆者は容量が等しい SCM と CCM のミスヒット率を評価するための解析手法を提案した[1]．また，C. V. Freiman[7] は高速プロセッサでキャッシュのバンド幅を広げる一手段として SCM をあげている．

しかし，筆者の知るかぎりでは SCM の実用化された例はない．それは，恐らく，キャッシュの実効サイクル時間を (1/2) マシンサイクル時間にしてまでも CPU 性能を上げねばならない要請が今までなかったからであろう．したがって，SCM の実現性についての検討報告もない．そこで，ここでは，SCM について

- CCM に比べミスヒット率がどうか
- 命令語用とデータ語用との容量配分をどうしたら最適であるか
- CCM の場合に比べ CPU 性能はどうか

等について検討を行い，SCM の効果とその限界について考察する．

2. ミスヒット率

SCM で，命令語用とデータ語用の各キャッシュメモリの容量の和を CCM の容量と同じにしたとき，そのミスヒット率は CCM のそれに比べどうなるかが SCM の実用性評価の上から重要なポイントの一つとなる．また，SCM で命令語用キャッシュメモリ (ICM) とデータ語用キャッシュメモリ (DCM) との容量配分をどうすればミスヒット率の観点から最適であるかを知ることは SCM 設計上重要である．これらの解析手法はすでに提案[1] したので，ここではその解析モデルと結果を示したのち実際の数値例でミスヒット率の評価を行う．

2.1 解析モデル

解析を簡単にするために，ここではアソシアティブ方式[4] (あるいは完全アソシアティブ方式) で考える．実用の大半はセットアソシアティブ方式[4] であるが，SCM と CCM のミスヒット率の比を問題にする範囲ではアソシアティブ方式でその様相をたしかめうる．

キャッシュの置換アルゴリズムは LRU (Least Recently Used) 方式[4] とする．

MM への書込方式は直接ストア方式[4] とし，キャッシュに該当ブロックがあればそれも更新する．

マルチプロセッサでの一致処理の影響は考えない．

ここで扱う命令語とは，ハードウェアがキャッシュアクセスのアルゴリズムに従って命令語と見なした情報であり，データ語についても同様である．また，ICM にデータ語があったり，DCM に命令語があったりすることでの損失もありうるが，ここの解析ではこれを無視する．

ICM と DCM の各ブロックセル数をそれぞれ n_I, n_D とし，SCM の総ブロック数または CCM のブロック数を n とする．

$$n=n_I+n_D \tag{1}$$

任意の MM アクセス要求が命令語の読出，データ語の読出と書込である確率をそれぞれ $\beta_I, \beta_D, \beta_S$ とする．

$$\beta_I+\beta_D+\beta_S=1 \tag{2}$$

タスクの切換え等で MM のアドレス空間が切換わるとキャッシュ内のエントリはほとんど無効になることがある．この事象をキャッシュのクリアと呼びその生起確率を命令読出確率 β_I の部分確率で定義し $1-\gamma$ とする．本解析ではクリアが生起するとキャッシュの全エントリを無効化する．

CCM, SCM のいずれにおいてもキャッシュに有効な命令語ブロックとデータ語ブロックがそれぞれ k と l だけ存在している状態 $s(k,l)$ において，命令あるいはデータの読出でヒットする確率をそれぞれ $\pi_I(k)$ あるいは $\pi_D(l)$ とすると

$$\left.\begin{aligned}&\pi_I(k)=1-(1-\pi_I(1))k-\lambda_I \\ &\qquad(k=1,2,\cdots\ ) \\ &\pi_D(l)=1-(1-\pi_D(1))l-\lambda_D \\ &\qquad(l=1,2,\cdots\ ) \\ &\pi_I(0)=\pi_D(0)=0\end{aligned}\right\} \tag{3}$$

で近似できる[1]．ただし，$\pi_I(1), \pi_D(1), \lambda_I, \lambda_D$ は定数である．式 (3) は中村らの測定[8]，Makino ら[9] と Belady[10] の実験式と同じである．

また，$s(k,l)$ である確率を SCM では $p(k,l)$, CCM では $q(k,l)$ とする．

2.2 ミスヒット率を与える式

SCM と CCM のミスヒット率をそれぞれ α_p と α_q とし，その比を θ とすると，これらは次式で与えられ

る[1].

$$\alpha_p=1-\sum_{k=0}^{n_I}\sum_{l=0}^{n_D}(\beta_I\pi_I(k)+\beta_D\pi_D(l))\cdot p(k,l)/(\beta_I+\beta_D) \quad (4)$$

$$\alpha_q=1-\sum_{k=0}^{n}\sum_{l=0}^{n-k}(\beta_I\pi_I(k)+\beta_D\pi_D(l))\cdot q(k,l)/(\beta_I+\beta_D) \quad (5)$$

$$\theta=\alpha_p/\alpha_q \quad (6)$$

ただし，$p(k,l)$ は

$$\left.\begin{aligned}
&p(1,0)=(a(0,0)/R(1,0))p(0,0)\\
&p(1,l)=(b(l-1)/R(1,l))p(1,l-1)\\
&\qquad(l=1,2,\cdots,n_D)\\
&p(k,0)=(a(k-1)/R(k,0))p(k-1,0)\\
&\qquad(k=2,3,\cdots,n_I)\\
&p(k,l)=(a(k-1)/R(k,l))p(k-1,l)\\
&\quad+(b(l-1)/R(k,l))p(k,l-1)\\
&\qquad(k=2,3,\cdots,n_I,\ l=1,2,\cdots,n_D)\\
&\sum_{k=0}^{n_I}\sum_{l=0}^{n_D}p(k,l)=1\\
&p(0,1)=p(0,2)=\cdots=p(0,n_D)=0
\end{aligned}\right\} \quad (7)$$

で与えられ，$q(k,l)$ は

$$\left.\begin{aligned}
&q(k,l)=t(k,l)\cdot q(0,0)\ (k+l<n)\\
&R(n,0)q(n,0)-d(n-1,1)q(n-1,1)\\
&\quad=a(n-1)q(n-1,0)\\
&-e(k+1,n-k-1)q(k+1,n-k-1)\\
&\quad+R(k,n-k)\cdot q(k,n-k)\\
&\quad-d(k-1,n-k+1)q(k-1,n-k+1)\\
&\quad=a(k-1)q(k-1,n-k)+b(n-k-1)\\
&\qquad\cdot q(k,n-k-1)\ (k=2,3,\cdots,n-1)\\
&-e(2,n-2)q(2,n-2)\\
&\quad+R(1,n-1)q(1,n-1)\\
&\quad-d(0,n)q(0,n)\\
&\quad=b(n-2)q(1,n-2)\\
&-e(1,n-1)q(1,n-1)+R(0,n)q(0,n)=0\\
&q(0,1)=q(0,2)=\cdots=q(0,n-1)=0\\
&\sum_{k=0}^{n}\sum_{l=0}^{n-k}q(k,l)=1
\end{aligned}\right\} \quad (8)$$

で与えられ，ここに，$t(k,l)$ は

$$p(k,l)=t(k,l)p(0,0)\qquad(k\neq n_I,\ l\neq n_D) \quad (9)$$

で与えられ，また，

$$\left.\begin{aligned}
&a(0,0)=\gamma\\
&a(k)=\beta_I\gamma(1-\pi_I(k))\\
&b(l)=\beta_D(1-\pi_D(l))
\end{aligned}\right\} \quad (10)$$

$$r(k,l)=\beta_I\gamma\pi_I(k)+\beta_D\pi_D(l)+\beta s \quad \begin{pmatrix}\text{CCM}:\ 1\leq k+l<n,\ k\geq 1\\ \text{SCM}:\ 1\leq k<n_I,\ 0\leq l<n_D\end{pmatrix} \quad (11)$$

$$\left.\begin{aligned}
&r_P(n_I,l)=\beta_I\gamma+\beta_D\pi_D(l)+\beta s\\
&\qquad(0\leq l<n_D)\\
&r_P(k,n_D)=\beta_I\gamma\pi_I(k)+\beta_D+\beta s\\
&\qquad(1\leq k<n_I)\\
&r_P(n_I,n_D)=\beta_I\gamma+\beta_D+\beta s
\end{aligned}\right\} \quad (12)$$

$$\left.\begin{aligned}
&d(k,n-k)=(1-k/n)\beta_I\gamma(1-\pi_I(k))\\
&\qquad(k=0,1,\cdots,n)\\
&e(k,n-k)=(k/n)\beta_D(1-\pi_D(n-k))\\
&\qquad(k=0,1,\cdots,n)\\
&r_q(k,n-k)=\beta_I\gamma\{\pi_I(k)+(k/n)(1-\pi_I(k))\}\\
&\quad+\beta_D\{\pi_D(n-k)+(1-k/n)(1-\pi_D(n-k))\}\\
&\quad+\beta s\\
&\qquad(k=0,1,\cdots,n)
\end{aligned}\right\} \quad (13)$$

$$\left.\begin{aligned}
&R(k,l)=1-r(k,l)\\
&\qquad\text{SCM}:\ k\neq n_I\ \text{かつ}\ l\neq n_D\\
&\qquad\text{CCM}:\ k+l\neq n\\
&\quad=1-r_P(k,l)\\
&\qquad\text{SCM}:\ k=n_I\ \text{かつ/または}\ l=n_D\\
&\quad=1-r_q(k,l)\\
&\qquad\text{CCM}:\ k+l=n
\end{aligned}\right\} \quad (14)$$

である．ここで，式(13)の導出には，CCM が $s(k,n-k)$ にあるとき，全エントリのうち最も古く参照されたブロックが命令語／データ語のブロックである確率は，全エントリ数に占める命令語／データ語ブロック数に比例すると近似した．

2.3 ミスヒット率の評価

ここでは，SCM のミスヒット率とそれと等容量の CCM のミスヒット率との比 θ を評価する．

ACOS システム 500 上での FORTRAN と COBOL の実測例では

$$\left.\begin{aligned}
&1-\pi_I(1),\ 1-\pi_D(1)=0.2\sim0.8\\
&\lambda_I,\ \lambda_D=1\sim3
\end{aligned}\right\} \quad (15)$$

であった[1].

8バイト単位で MM にアクセスする CPU では，SGM (Scientific Gibson Mix.) で $\beta_I=0.26$, $\beta_D=0.64$, BGM (Business Gibson Mix.) で $\beta_I=0.19$, $\beta_D=0.61$ あるが両者の β_I, β_D にあまり大きな差がない．ここでは

$$\left.\begin{aligned}
&\beta_I=0.3 \qquad \beta_D=0.6\\
&\beta_I=0.47 \qquad \beta_D=0.23
\end{aligned}\right\} \quad (16)$$

の二つのケースで評価する．

図2は，$n_I=(1/2)n$ のとき γ と n が θ に及ぼす影響を評価している．β_I, β_D, λ_I, λ_D, $\pi_I(1)$, $\pi_D(1)$ は式(15)(16)の範囲で θ を最も大きくする値に設定してある．一般に，n は大きい方が，γ は小さい方が θ は1に近づく．32バイト／ブロックのキャッシュでは $n=512$

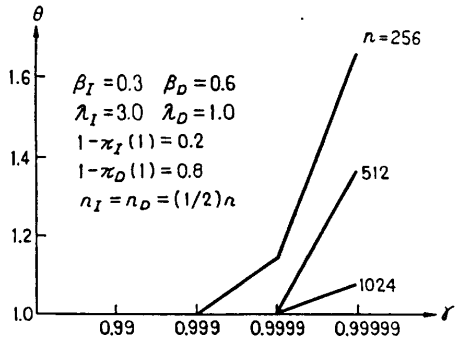

図 2 n と γ の θ に対する影響
Fig. 2 Effects of n and γ on θ.

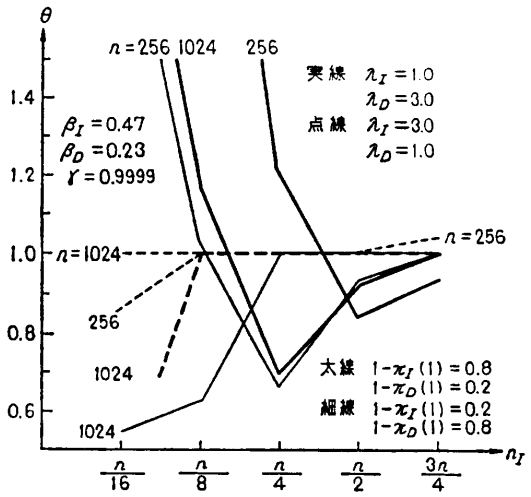

図 3 n_I に対する θ の値 (1)
Fig. 3 Effect of n_I on θ (1).

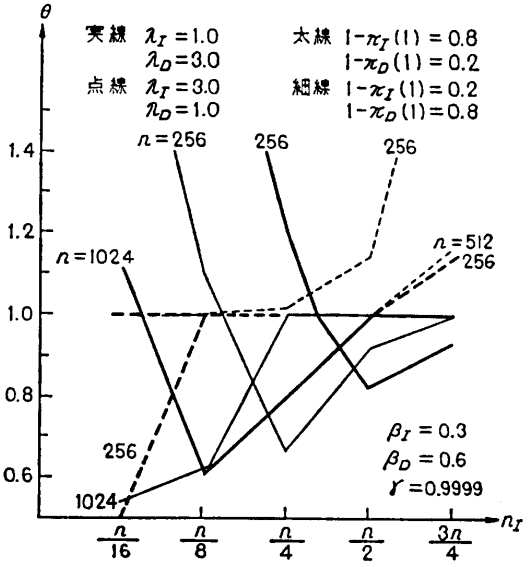

図 4 n_I に対する θ の値 (2)
Fig. 4 Effect of n_I on θ (2).

のとき 16 キロバイト容量になるが，これ以上のキャッシュ容量であれば $\gamma \leqq 0.9999$ (すなわち，おおよそ $1 \sim 3 \times 10^4$ 命令に 1 回以上のクリア) のとき θ は 1 以下である．$\gamma=0.99999$ でもキャッシュの総容量が 32 キロバイトであれば $\theta \leqq 1.07$ である．

図 3 は $\beta_I=0.47$, $\beta_D=0.23$ のとき，**図 4** は $\beta_I=0.3$ $\beta_D=0.6$ のときの n_I に対する θ の変化を示している．ただし $\gamma=0.9999$ に固定している．これから，各パラメータ値の変化に対して平均的に θ を最小に保つには $n_I=(1/2)n$ が最適であることがわかる．特に，$n \geqq 512$ であれば $n_I=(1/2)n$ で $\theta \leqq 1$ である．

また，**図 3～4** で条件によっては θ を 1 より大幅に小さくすること，すなわち SCM のミスヒット率が CCM よりずっと小さい場合があることがわかる．このことはプログラムの特性により—また，その特性が大きく変化しないとき—SCM のミスヒット率が CCM のそれより小さいように n_I を選択できる可能性を示唆している．

3. キャッシュの論理遅延時間とマシンサイクル

普通，CPU の目標性能と使用テクノロジを勘案して CPU のマシンサイクルをあらかじめ定め，これに合せて論理設計を行う．テクノロジがもつ性能にくらべ特にマシンサイクル時間を短くしない限り，クロック間の論理遅延時間のクリティカルパスはキャッシュとコントロールストアレッジ (CS) が主である．すなわち，キャッシュ，CS は論理構造の自由度にとぼしいが，その他の論理ブロックは論理回路の組立て方で論理遅延時間をコントロールできるのが普通である．

CPU が高性能化すると CS は分散化するかワイアド論理化するため一つの CS のワード数は減少する方向にある．一方，キャッシュ容量は高性能 CPU になる程増大する傾向にある．したがって，CPU が高速になる程キャッシュの論理遅延時間がマシンサイクル時間の最小値を決める確度が高い．

図 1 (a) では TLB, AA, DA の各索引を各々 1 マシンサイクルで，(b) では TLB と AA の並列索引を 1 マシンサイクルで実行後 DA の索引を次のマシンサイクルで，(c) では 1 マシンサイクルで TLB 索引後次のマシンサイクルで AA と DA の並列索引を，(d) では TLB, AA と DA のすべての索引を 1 マシンサイクルで並列に行う．一般に，DA, AA, TLB の順に論理遅延時間が大きいから，CPU のマシンサイ

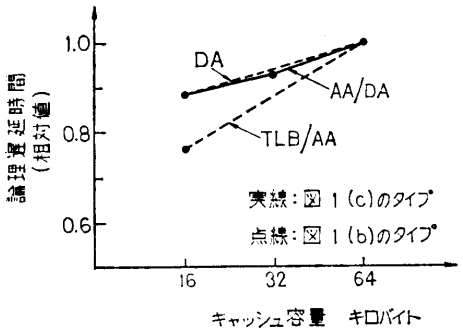

図5 キャッシュの論理遅延時間
Fig. 5 Delay time of cache memory.

クル時間がキャッシュによって決定づけられるとするとそれは(d), (c), (b)と(a)の順に長い.

キャッシュ容量の増大とともに DA, AA の論理遅延時間も増大する. それは, キャッシュ容量の増大が DA と AA におけるメモリアレーの増加, メモリアレー駆動回路の増加, メモリアレー出力の一致回路の増加, メモリアレー出力の論理和回路の増加等をきたすことにより信号の媒体伝播時間の増大と IC チップ段数の増大を生ずるからである. **図5**はキャッシュ容量と論理遅延時間の関係の一例である. ただし, 32バイト／ブロックのセットアソシアティブ方式で論理遅延時間は 64 キロバイトのときを各ケースとも1とした相対値である. 実線は図1(c)のタイプで AA と DA の並列索引の論理遅延時間であり, 64 キロバイトのとき 256 セット×8 コンパートメントで, 16 キロバイトのとき 128 セット×4 コンパートメントであり, 32 キロバイトのときはセット数とコンパートメント数のいずれを変化させても図5では1点に縮退してみえる程度の差である. 点線は図1(b)のタイプで DA の論理遅延時間と TLB, AA の並列索引の論理遅延時間の例である. 64 キロバイトのとき 128 セット×16 コンパートメントで, 16 キロバイトのときコンパートメント数を4にした. 実線と点線は全く異なったテクノロジを使用しているが結果はよく一致している. 図5で DA に着目すると, 64 キロバイトにくらべ 32 キロバイトは 7%, 16 キロバイトは 11% の論理遅延時間の減少になっており, 実線の例でみると 32 キロバイトにくらべ 16 キロバイトは 5% の論理遅延時間の減少になっている.

SCM では, 図1の AA と DA はいずれも命令語用とデータ語用に分割した二組で構成される. これに伴い論理構成上から図1(b)(d)では TLB も命令語用とデータ語用に分割する必要があるが(a)(c)ではその必要はない. ただし, 命令フェッチとデータフェッチの競合を TLB においても完全になくすためには TLB も二重化が必要であるが, ページまたはセグメントをまたがるときのみ TLB 索引を行う方式をとれば TLB の二重化は避けながらも競合を無視できる程度まで小さくできるであろう. キャッシュを二重化すると, すくなくとも

- キャッシュ読出制御
- MM 書込でのキャッシュ一致処理制御

の二重化が必要であり, これによるハードウェア量の増分は, 一試行設計によると, 総容量 128 キロバイトキャッシュをもち M 200 H 程度の性能規模のコンピュータで CPU 全体の 2～3% 程度である.

以上から, ICM と DCM が等容量の SCM は論理遅延時間の観点からは半分の容量をもつ CCM と等価である. したがって, キャッシュが CPU のマシンサイクルを決める論理遅延時間のクリティカルパスであると, $n_I=n_D$ の SCM はこれと等容量の CCM にくらべ図5の例ではマシンサイクル時間を 5～7% 短縮することができるが CPU 全体の 2～3% 相当のハードウェアを余分に投資する必要がある.

更に, CCM を時分割で使用する場合, 伝播信号のレーシングを防止するためクロック間にラッチ回路の挿入が必要となりうるが, この場合更にマシンサイクル時間を長くする.

4. 平均命令実行時間

ここでは時分割方式の CCM をもつ CPU と, それと等容量の SCM をもつ CPU とで平均命令実行時間を比較してみる.

筆者は I_m バンクのインタリーブされた MM と各バンクごとに1段ストアバッファをもちミスヒット率 α のキャッシュをもつ単一プロセッサシステムの平均命令実行時間 $\tilde{t}$ は次式で与えられ, これが実際をよく表わしていることを示した[2].

$$\left.\begin{aligned}&\tilde{t}=\tilde{t}_0+x\beta_R\alpha t_{aR}+(x\beta_s/I_m)H(\tilde{t}_{cs},\alpha,I_m)\\&H(\tilde{t}_{cs},\alpha,I_m)=(\beta_R\alpha+\beta_s)(x/\tilde{t}_0)\\&\quad\cdot\int_0^{\tilde{t}_{cs}}u\left(1+\frac{x\beta_s(\tilde{t}_{cs}-u)}{\tilde{t}_0 I_m}\right)^{I_m-1}\\&\quad\cdot\exp\{-(\beta_R\alpha+\beta_s)(x/\tilde{t}_0)(\tilde{t}_{cs}-u)\}\,du\end{aligned}\right\}\quad(17)$$

ここに,

$\tilde{t}_0$: MM アクセスを考えないときの平均命令実行時間

t_{aR}: キャッシュからみた MM のアクセス時間

$\tilde{t}_{cs}$: MM の書込サイクル時間（平均値）

x: 命令当りの平均 MM アクセス要求回数

β_R: MM アクセスが読出である確率

β_s: MM アクセスが書込である確率

$$\beta_R+\beta_s=1 \tag{18}$$

式(17)の右辺第2項は MM の実効アクセス時間，第3項は MM 上での平均メモリ待ち時間を与えている．後者は I_m を充分大きくすれば無視できる程度に小さくすることができるから，マシンサイクル時間を τ とすると式(17)は

$$\left.\begin{aligned}&\tilde{t}/\tau \fallingdotseq \tilde{r}_0+x\beta_R\alpha r_{cR}\\&\tilde{r}_0=\tilde{t}_0/\tau,\quad r_{cR}=t_{cR}/\tau\end{aligned}\right\} \tag{19}$$

と書くことができる．τ と α の変量 $\Delta\tau$ と $\Delta\alpha$ に対する $\tilde{t}$ の変量 $\Delta\tilde{t}$ は式(19)から

$$\left.\begin{aligned}&\Delta\tilde{t}/\tilde{t}=\Delta\tau/\tau+K\cdot(\Delta\alpha/\alpha)\\&K=x\beta_R\alpha r_{cR}/(\tilde{r}_0+x\beta_R\alpha r_{cR})\end{aligned}\right\} \tag{20}$$

となる．大型機では，ほぼ

$$K=0.1\sim0.2 \tag{21}$$

である．したがって，マシンサイクル時間の変化はそのまま平均命令実行時間の変化になるが，キャッシュのミスヒット率の変化は 1/5～～1/10 に減少して平均命令実行時間にきく．

キャッシュがマシンサイクル時間を決めているとして，図3～5の例でみると，$n\geq512$ (32バイト／ブロックで16キロバイト以上の容量) なら $n_I=(1/2)n$ の点で $\theta\leq1$ であるから32～64キロバイトの CCM を命令語用とデータ語用とに等分割した SCM にすると平均命令実行時間はすくなくとも 5～7% 短縮される．**3** の末尾に述べたクロック間のラッチ回路があるとこの差はさらに大きくなる可能性がある．

一般に，SCM は等容量の CCM より平均的にミスヒット率が $\Delta\alpha$ だけ大きく，マシンサイクル時間は $\Delta\tau$ 短縮されるとしたとき式(20)から

$$\Delta\alpha/\alpha<(1/K)\cdot(-\Delta\tau/\tau) \tag{22}$$

であれば SCM をもつ CPU の方が高い性能をうる．

5. むすび

1マシンサイクルに二つのキャッシュアクセスを必要とする超高速 CPU で，キャッシュの実現手段の一つとして SCM があり，これは等容量の CCM に比べすくなくとも

・キャッシュ読出制御

・MM 書込でのキャッシュ一致処理制御

のハードウェアを ICM と DCM とに対するものとして二重にもつ必要があるが (CPU 全体に対して数パーセントのハードウェア量増)，多くの場合，ICM と DCM を等容量にした SCM は CCM に比べ各パラメータ値の広汎な変化に対して平均的に遜色のないミスヒット率が得られる一方，キャッシュがマシンサイクル時間短縮のボトルネックになっているとき SCM は CCM よりマシンサイクル時間を短縮できるため CPU 性能が向上する．キャッシュ容量が大きくなる程この結論の妥当性の確度は高くなる．このアプローチはハードウェアの投資効率のよさよりも CPU 性能そのものを最大限まで追求するものであり，そこまでは性能を追求せずむしろハードウェアの投資効率を重視しかつキャッシュ設計の煩雑さを回避したいときは CCM を選択することになろう．

実効キャッシュサイクル時間を (1/2) マシンサイクル時間にする時分割方式の CCM がテクノロジまたは論理構成上の制約から困難である場合は超高速 CPU を実現するために SCM の導入は必至となろう．

ここでは単にその可能性を示唆するに止めるが，SCM は前述の効果以外に ICM と DCM に分割されていることにより，命令語シーケンスとデータ語シーケンスの各特性を個別にかつ効果的に利用したキャッシュ設計を可能にするであろう．その一例として，命令の分岐履歴を ICM で管理することがあげられる．超高速 CPU において分岐命令での先取命令キャンセルによるパイプラインの乱れの影響が式 (17) の $\tilde{t}_0$ の半分近くを占めているが，ICM の分岐履歴で分岐命令以後の先取方向を制御することによりパイプラインの乱れを大幅に減少させうる可能性がある．また，**2.3** で述べたように，特別な利用環境として，プログラムの特性が一定であれば，n_I の選び方で CCM よりミスヒット率の小さい SCM を実現しうる場合があるのもその一例である．

参考文献

1) 齋藤将人：キャッシュのノットファウンド確率を求める一つの解析的手法について，信学技報，Vol. 77, No. 129, pp. 7-17 (1977-09).

2) 齋藤将人：単一プロセッサシステムにおけるキャッシュ，ストアバッファとメモリインタリーブの効果と多重プロセッサシステムの性能について，情報処理学会論文誌，Vol. 21, No. 5, pp. 391-401 (1980-09).

3) Liptay, J. S.: Structual aspects of the system/360 model 85. II, The cache, *IBM Systems J.*, Vol. 7, No. 1, pp. 15-21 (1968).

4) 長島重夫，堀越 彌：キャッシュ記憶，情報処理，Vol. 21, No. 4, pp. 333-340 (1980-04).
5) 中澤喜三郎他：LSI 技術の助けを借りてパイプライン方式を強化した最高速の商用汎用コンピュータ，日経エレクトロニクス，No. 228, pp. 104-130, 12-24 (1979).
6) 飯塚 肇：スレーブメモリを持つ計算機システムの簡単な解析，信学論，Vol. 54-C, No. 8, pp. 698-705, '71/8.
7) Freiman, C. V.: VLSI High Performance Processors: How Mixed the Blessing ?, COMPCON SPRING 80, 25-29 (Feb. 1980).
8) 中村奉夫他：バッファメモリ方式のシュミレーション，情報処理，Vol. 15, No. 1, p. 26 (1972).
9) Makino, T. and Ohno, N.: Characteristics of Not Found Probability (NFP) in Memory Hierachy System, NEC R & D, No. 43, pp. 51-58, (Oct. 1976).
10) Belady, L. A. and Kuehner, C. J.: Dynamic Space-sharing in Computer System, CACM, 12, 5, pp. 282-288 (1969).

(昭和54年11月5日受付)
(昭和55年11月20日採録)