00-01075

次世代マルチメディア情報端末を指向する VLSI アーキテクチャの研究

中　村　　宏　　東京大学先端科学技術研究センター助教授

1. はじめに

21世紀においては，高速バックボーン，一般家庭までの高速ネットワークの整備と共に，多様なマルチメディア情報端末が大衆家電として各家庭に入り込むことによって，真の意味での高度情報化社会が達成されると期待されている。その実現のためには，社会インフラであるネットワークの整備，および，高品質なマルチメディア情報端末を廉価に提供することが重要となる。ここで後者に注目すると，マルチメディア情報端末に対する性能向上の要求は，デバイスの微細化による素子の速度向上を凌いでおり，従来のディジタルシステムの VLSI アーキテクチャを踏襲していては，その要求を満たすことができなくなる，という問題が指摘されている。

この問題は具体的には以下のように説明される。大容量のデータを扱うマルチメディア処理においては，システム全体の処理能力は演算処理部の能力だけではなくメモリシステム側のデータ供給能力によっても決定される。半導体技術の進展により，素子の集積度と演算処理の速度は今後も向上すると期待されているが，DRAM，SRAM 等の記憶素子の速度は大きくは向上しないと予測され，また VLSI を実装するパッケージのピン数も物理的な制約によりそれほど増加しないため，プロセッサとメモリ間の転送スループットを向上させるのは困難な状況である。

そこで，本研究では素子集積度の向上を活用すべく，同一 VLSI 上にプロセッサコアと大容量のメモリを搭載する VLSI アーキテクチャを提案する。大容量メモリは，従来型のキャッシュだけでなく，新たにソフトウェアから制御可能なメモリとしても用いる。この可制御メモリは，アドレス指定可能としソフトウェアによるデータ配置と入れ替えを制御可能とする点が従来のキャッシュとは本質的に異なる。また，VLSI 上大容量メモリは，キャッシュと可制御メモリの比率を動的に再構成可能なものを提案する。我々はこのアーキテクチャを SCIMA（Software Controllable Integrated Memory Architecture）と呼んでいる。

本研究では，SCIMA の可制御メモリをソフトウェアから利用するためのコンパイル技術の開発[2][4]，提案するアーキテクチャの VLSI 上での実装可能性とその有効性の評価[1][3]を行った。

2. SCIMA のアーキテクチャ

図1に，SCIMA の構成を示す。

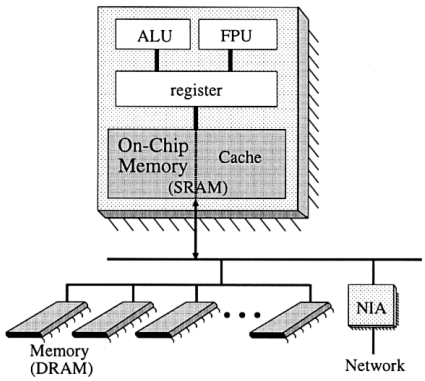


図1　SCIMA の構成

SCIMA では，チップ上のメモリとして，キャッシュに加えオンチップメモリを搭載する。キャッシュはハー

ドウェア制御によりデータ配置・置き換えが行われるのに対し，オンチップメモリは，ソフトウェアでデータ配置・置き換えの指定が可能である。

2.1 アドレス空間

SCIMA では，論理アドレス空間上にオンチップメモリ領域をマップする。オンチップメモリは大きな連続ブロック領域であるため，この管理を TLB ではなく専用レジスタで行ない，TLB ミスの頻発を防ぐ。導入されるレジスタは，オンチップメモリの開始アドレスを保持する ASR（On-Chip Address StartRegister）とオンチップメモリ容量を保持する AMR（On-Chip Address Mask Register）である。

2.2 拡張命令

SCIMA では，page-load/page-store と呼ぶ，オンチップメモリと主記憶間の転送命令を ISA（命令セットアーキテクチャ）上に備える。この命令により，オンチップメモリのデータ配置・置き換えをソフトウェアで行うことが可能となる。また，本命令は，ブロックストライド転送機能を備える。データ転送元の開始番地，データ転送先の開始番地，転送サイズ，ブロック幅，ストライド幅の５オペランドを用いたブロックストライド転送機能により，不連続なデータをオンチップメモリ上の連続領域に転送させることができるため，無駄なデータ転送を省き，チップ内の記憶領域を有効に利用可能である。

オンチップメモリ領域は *page* と呼ぶ複数のブロックに分割され，この *page* を単位として管理する。*page* のサイズは２のべき乗である。page-load/page-store 命令で転送できる最大データサイズはこの *page* のサイズであり，*page* を跨いでの転送はできない。

2.3 キャッシュ・オンチップメモリ統合機構

SCIMA では，キャッシュとオンチップメモリをハードウェア的に統合し，それらに割り振られる容量比を，対象とするアプリケーションの性質に合わせて実行時に再構成できる[4]。本機構では，従来のキャッシュに対し，way 単位でキャッシュ・オンチップメモリの属性を設け，データ参照を制御する。属性は WLR（Way Lock Register）に保持され，参照する way の判別に用いられる。

3. SCIMA 用最適化コンパイルアルゴリズムの検討

SCIMA 用最適化コンパイルアルゴリズムの検討として，SCIMA の特徴であるオンチップメモリをどのように用いれば性能向上が達成できるかを検討した。

我々はアクセスの連続性とデータの再利用性の観点から配列の特徴を整理し（**表１**），その分類に基づいたオンチップメモリの利用法を提案している[4]。具体的には**図２**のようにオンチップメモリを用いることで性能向上を図る。

表１：配列のアクセスの特徴の分類

アクセスの特徴	分類
再利用性	あり / なし
連続性	連続 / ストライド / 不規則

⑴ **再利用性はないが連続アクセスされる配列**：page-load/page-store 命令による大粒度転送を行い，転送回数を減らすことでオフチップメモリレーテンシの影響を減らすことができる。この場合はオンチップメモリ上に *page* サイズのバッファを確保し，その領域をストリームバッファとして用いながらアクセスする。

⑵ **再利用性はないがストライドアクセスされる配列**：page-load/page-store のもつストライド転送機能により無駄なデータ転送を省き効率的なデータ転送を行うことが可能である。⑴と同様オンチップメモリをストリームバッファとして用いる。

⑶ **再利用性がなくアクセスが不規則な配列**：このような配列に関しては SCIMA のオンチップメモリを用いても効果が期待できない。したがってオンチップメモリを用いない。

⑷ **再利用性があり連続アクセスされる配列**：ブロッキング手法を用いるなどオンチップメモリサイズに分割し

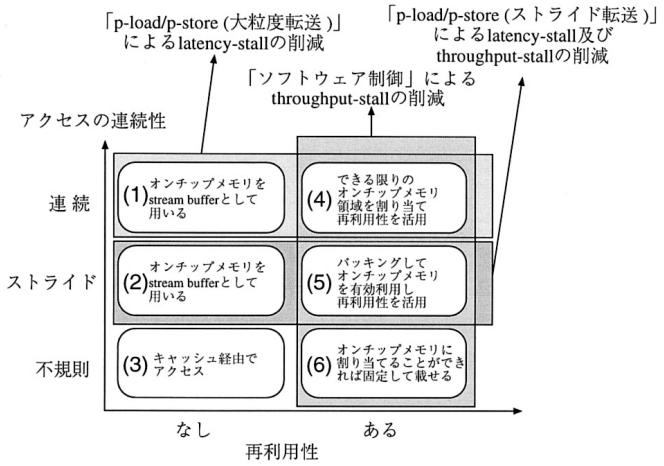


図２：配列の特徴に対するオンチップメモリの利用法

てアクセスすることで他のデータとの干渉を防ぎながら再利用性を最大限活用する。オンチップメモリ上にループ中でアクセスされるワーキングセット分のオンチップメモリ領域を割り当てる。

⑸ **再利用性がありストライドアクセスされる配列**：page-load/page-store 命令のストライド転送機能により不連続なデータをパッキングしてオンチップメモリに載せ，チップ内記憶領域の有効活用を図ると同時に，他のデータとの干渉を防ぎながら再利用性を最大限活用する。⑷と同様にループ中でアクセスされるワーキングセット分のオンチップメモリ領域を割り当てる。

⑹ **再利用性はあるがアクセスが不規則な配列**：アクセスされる範囲があらかじめわかり，かつその範囲がオンチップメモリに載る程度の大きさであればオンチップメモリ上に固定してデータを載せ再利用性を活用する。オンチップメモリに載らない場合はキャッシュを用いてアクセスする。

⑴〜⑹は独立な最適化であるが，まず，⑴，⑵に分類される配列に対する最適化が重要である。これは，再利用性のないデータに対してキャッシュは無力であるため，ストリームバッファを提供してレーテンシの影響を減らすことに効果があると考えられるためである。次に，データの再利用性がある⑷，⑸，⑹に分類される配列に対する最適化を行い，さらなる性能向上を行う。

4. SCIMA のメモリアクセス機構の設計

SCIMA の VLSI 上での実装可能性を検討するため，実際に SCIMA の設計を RTL（レジスタトランスファレベル）で Verilog-HDL を用いて行った。SCIMA は，既存のアーキテクチャに対し，SCIMA 章で述べた拡張を施すことで定義できる。本検討では，MIPS R10000 プロセッサに対して拡張を施したものとして SCIMA の設計を行った。

4.1 SCIMA におけるメモリアクセス命令の動作

R10000 に対し SCIMA 用追加制御を加えた，load/store 命令のパイプライン動作を**図３**に示す。**図３**の斜線部が R10000 に対する追加動作である。OCM test でオンチップメモリアドレスかどうかを判定し，address comp. で先行 page-load/page-store 命令との依存判定を行なう。以上の追加動作は従来のメモリアクセス動作と並列に行なわれる。

page-load/page-store 命令は PLS queue に格納した後，転送元・転送先アドレスのオペランドが利用可能になり次第レジスタファイルを参照し，アドレスを取得して後述する PLS address stack に格納する。その後，先行命令が全て実行完了した後に issue され，全オペランドの情報を取得し転送を開始する。転送終了後，PLS queue のエントリを削除する。

4.2 メモリアクセス機構の設計

本章では，設計した SCIMA のメモリアクセス機構の動作について述べる。設計対象とする範囲を**図４**の斜線部で示す。今回の設計では，SCIMA での追加拡張機構に対象を絞って設計を行なった。設計した各機構の動

作を以下に述べる。

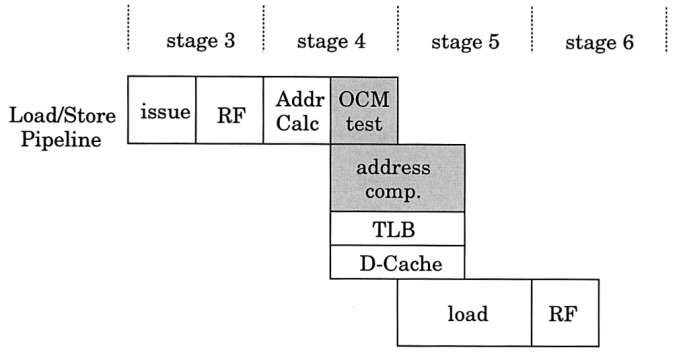


図３：load 命令のパイプライン動作

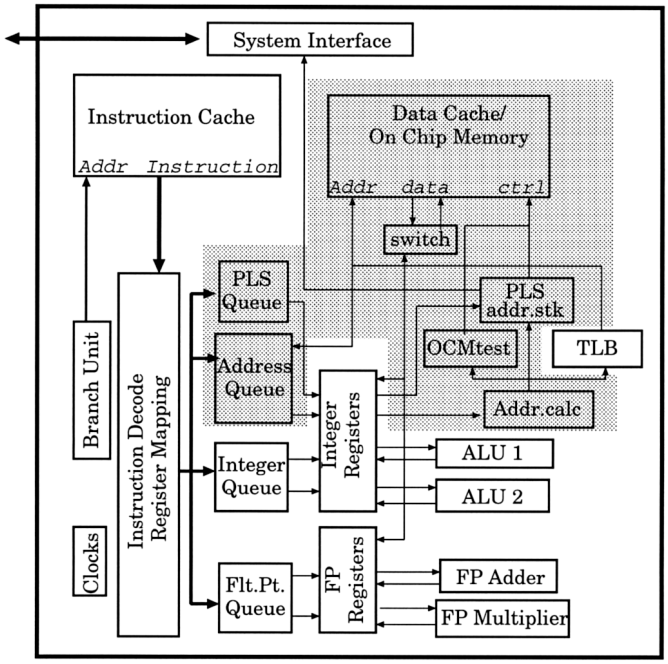


図４：設計範囲

wakeup & select logic

以降では，address queue，PLS queue，および select logic をまとめて wakeup & select logic と表記する。SCIMA では page-load/page-store 命令用の命令 queue として PLS queue を新たに設ける。また，今回の設計では，load/store 命令と page-load/page-store 命令の実行において共有するリソースが多いことから，load/store 命令と page-load/page-store 命令の issue ポートを統合する。

Address queue

Address queue は R10000 に存在する load/store 命令専用の命令 queue であり，16エントリの circular FIFO 構造をしている。Address queue の各エントリにおける制御は，以下の通りである。SCIMA の拡張として，page-load/page-store 命令との依存等，issue の条件が増える。

PLS queue

PLS queue は SCIMA の追加機構であり，page-load/page-store 命令専用の命令 queue である。今回作成した PLS queue は４エントリの circular FIFO 構造をして いる。PLS queue の各エントリにおける制御は，Address queue と同様である。

select logic

select logic は命令 queue の ready 信号を検知し，その中から発行を許可する命令を選択する。今回 SCIMA

への拡張を施した select logic では，前述の issue ポート統合のため, Address queue と PLS queue の両者より ready 信号を受け取り，その中から1つを 選択する構成となっている。

PLS address stack

PLS address stack は，SCIMA で追加される機構で，PLS queue とエントリ数が同じ queue であり，各エントリは PLS queue のエントリと一意に対応している。page-load/page-store 命令は取得した転送元・転送先アドレスを PLS address stack に保持する。保持したアドレスは，転送開始時に利用する。一方，load/store 命令の実行時には，エントリ内のアドレスとのアドレス比較により，page-load/page-store 命令との依存判定（図３の address comp. 動作）を行なう。

OCM test 機構

OCM test では，２章で述べた ASR，AMR および WLR を用いてオンチップメモリアドレスかどうかの判定を行ない,オンチップメモリアドレスであればキャッシュ・オンチップメモリの参照 way を示す信号を作成する。

cache data select

従来のデータキャッシュが持つ data select 部を SCIMA 用に拡張するもので，OCM test および address comp. の結果によりオンチップメモリアドレスであると判定された場合には，タグの判定結果を OCM test 機構より送られた way 選択信号で置き換える制御を追加する。

5. 評価

5.1 面積・遅延の評価

R10000に準じたメモリアクセス機構（cache model），および４章で述べたメモリアクセス機構（SCIMA model）をそれぞれ Verilog での RTL 記述により設計し，その面積と遅延を調べる。評価には VDEC 提供の 0.35μm プロセスライブラリを用いる。論理合成ツール上で，速度を最優先として合成した結果を**表２**に示す。

表２：面積・遅延の評価(速度最優先で合成)

cache model	area [mm^2]	latency [ns]	SCIMA model	area [mm^2]	latency [ns]
wakeup & select logic			wakeup & select logic		
- Address queue	1.64	2.07	- Address queue	1.86	2.27
- select logic	0.03	4.04	- select logic	0.05	4.21
			- PLS queue	0.37	1.80
address calc. logic	0.15	3.21			
			OnChipMemory Control logic		
			- OCM test logic	0.09	1.43
			- PLS address stack	0.49	1.59
cache			cache/OnChipMemory		
- decode logic	0.28	1.82			
- data select logic	2.50	3.31	- data select logic	2.43	3.50

5.1.1 サイクルタイムへの影響

R10000 のパイプライン構成に合わせてサイクルを区切ると，［Address queue/PLS queue］と select logic で１サイクル，address calc. と［cache decode/OCM test/PLSstack］で１サイクル，SRAM macro の参照と cache data select で１サイクルとなる。

SRAM macro は設計対象外であり，**表２**に載っていないが，SRAM macro の遅延が 2.8[ns] 以上であれば，SRAM macro と cache data select がサイクルタイムとなる。一方，**表２**中で最も latency が長いのは wakeup & select logic であり，メモリアクセス機構の中では wakeup & select と SRAM macro & dataselect がクリティカルパスの候補となる。

wakeup & select がクリティカルパスであった場合，サイクルタイムには4.9%程度の影響があり，SRAM macro & data select がクリティカルパスであった場合，SRAM macro を両モデルでともに 2.8[ns] と仮定すると，サイクルタイムには3.1%程度の影響があることになる。よって，SCIMA 用拡張がサイクルタイムに影響を

及ぼしたとしても最大で4.9%である。

5.1.2 面積への影響

R10000 のフロアプランから求めた Address queue の面積は総面積の約2.7%である。**表2**より，SCIMA model における追加拡張機構（**表2**中の Address queue，select，PLS queue，OCM test，PLS address stack が対象）によって，面積は cache model での Address queue の約1.7倍となるが，それでも総面積の4.8%である。すなわち，SCIMA model での増分は約２%であり，面積に大した影響はない。

5.2 総合評価

5.2.1 評価方法

5.2 節で検討した追加拡張機構のサイクルタイムへの影響を考慮し，cache model と SCIMA model でのモデル間の相対評価を行なう。評価にはサイクルレベルシミュレータを用いる。また，評価におけるパラメータは**表3**に従う。PLS queue エントリ数は４である。

評価では，プロセッサの実行時間を CPU-busy time（T_b），latency-stall（T_l），throughput-stall（T_t）の３つに分類する。CPU-busy time とはプロセッサが実際に計算処理を行っている時間であり，latency-stall は主記憶のアクセスレーテンシがもたらすストール時間を，また throughput-stall はオフチップメモリのスループット不足に起因するストール時間を指す。

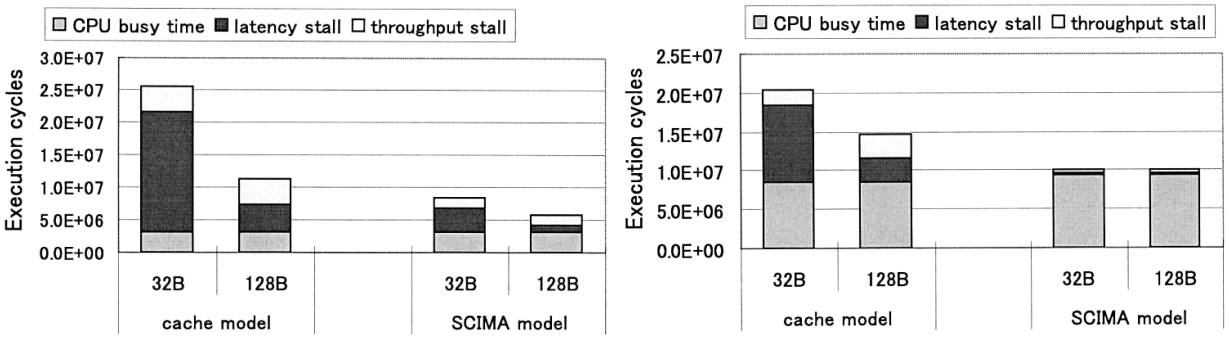
サイクルタイムの延びは CPU の動作速度に影響し，前節の結果の最悪値として SCIMA model での CPU 動作時間が cache model に比べ1サイクルあたり4.9%長くなるとする。一方，メモリ参照のレーテンシ・スループットはメモリの性能によるため，サイクルタイムへの影響はないとした。評価ベンチマークとして行列積と CG を用いる。評価において，オンチップメモリの利用法に関しては，３章で述べた最適化を施し，評価にはサイクルレベルのシミュレータを用いる。評価に用いるパラメータを**表3**に示す。

表3：評価に用いるパラメータ

同時発行命令数	
- 整数演算	2
- 浮動小数点演算(積和)	1
- 浮動小数点演算(除算/平方)	1
- ロード・ストア	1
命令queueサイズ	
- integer，FP，load/store	各16
Active list エントリ	48
キャッシュラインサイズ	32B，128B
*page*サイズ	4KB
オフチップメモリスループット	2B/cycle
オフチップメモリレーテンシ	80cycle
メモリサイズ 合計	64kB
- cache model	キャッシュ 64kB(4-way)
- SCIMA model	キャッシュ16kB(1-way)
(software pipelining)	オンチップメモリ 48kB

5.2.2 評価結果，考察

評価結果を**図5**(a)，**5**(b)に示す。縦軸は cache model のサイクル数を表す。SCIMA model は CPU-busy time が５%増加した結果，増加しないと仮定した場合に比べ，ラインサイズ 128B の場合には行列積で4.5%，CG で2.7%程度実行時間が長くなる。しかし，メモリアクセスの最適化により，latency-stall と throughput-stall で表されるオフチップメモリアクセス時のペナルティが大幅に削減されている。従って，全体性能で見ると，cache model に対して SCIMA model は，ラインサイズが 32B の時に行列積で103%，CG で197%性能が向上し，ラインサイズが 128B の時に行列積で45%，CG で95%性能が向上している。



(a) **総合評価（行列積）**　　(b) **総合評価（CG）**

図５： 評価結果

6. まとめ

次世代マルチメディア情報端末を指向した，同一 VLSI 上にプロセッサコアと大容量のメモリを搭載する VLSI アーキテクチャを提案しそのアーキテクチャ用のコンパイルアルゴリズムの提案と VLSI 実装上の実現可能性と有効性の評価を行った。評価結果より，提案するコンパイルアルゴリズムを用いた場合に，提案するアーキテクチャは従来のものに比べ有効であり，VLSI 実装上の問題も少ないことがわかった。今後の課題としては，コンパイルアルゴリズムの実装，ならびに現実のハードウェアとして実装を通した有効性の実証があげられる。

参考文献

［1］ 大根田拓，近藤正章，中村宏。SCIMA におけるメモリアクセス機構の検討。情報処理学会研究報告，No. ARC-144，pp.165-170，2001.

［2］ 藤田元信，近藤正章，中村宏，千葉滋，佐藤三久。ソフトウェア制御オンチップメモリのための最適化コンパイラの構想。情報処理学会研究報告，No. ARC-146，pp.31-36，2002.

［3］ 大根田拓，近藤正章，中村宏。SCIMA におけるメモリアクセス機構の設計と評価。情報処理学会研究報告，No. ARC-147，pp.79-84，2002.

［4］ 近藤正章，中村宏，朴泰祐。SCIMA における性能最適化手法の検討。情報処理学会論文誌，Vol.42，No. SIG12（HPS4），pp.37-48，2001.

〈発　表　資　料〉

題　　名	掲 載 誌 ・ 学 会 名 等	発 表 年 月
SCIMA におけるメモリアクセス制御機構の検討	情報処理学会研究報告，ARC-144-29，No. 165-170	2001年７月
SCIMA における性能最適化手法の検討	情報処理学会研究会論文誌，Vol.42，No. SIG12（HPS4），pp.37-48	2001年12月
ソフトウェア制御オンチップメモリのための最適化コンパイラの構想	情報処理学会研究報告，ARC-146-6，pp.31-36	2002年２月
SCIMA におけるメモリアクセス機構の設計と評価	情報処理学会研究報告，ARC-147-14，pp.79-84	2002年３月