計算機アーキテクチャ　87－6
（１９９１．３．１１）

３２ビットマイクロプロセッサMN10400の
マイクロプログラムシミュレータ実現方法

田積　誠
松下電器産業株式会社　情報通信関西研究所

本報告は３２ビットマイクロプロセッサMN10400の開発において作成したマイクロプログラム開発用シミュレータについて述べる。マイクロプログラム制御方式をとるプロセッサにおいて、マイクロプログラムをハードウェアの論理設計と並行して開発し開発効率を向上させるためには、マイクロプログラムシミュレータの早期構築が不可欠である。そこで、本プロセッサで用いたマイクロプログラムシミュレータでは、制御部を実際のハードウェア論理設計からボトムアップ的に機能記述し、実行部を仕様設計からトップダウン的に作成することにより、シミュレータの早期開発とハードウェアの設計変更に対する柔軟性を実現することができた。

The Method for Implementing the Microprogram-simulator of a 32-Bit Microprocessor MN10400

Makoto Tazumi
Kansai Information Communications Research Laboratory, Matsushita Electric Industrial Co.,Ltd.

This paper describes the method for implementing the microprogram-simulator of the 32-bit microprocessor MN10400. When we develop the microprocessor, we must implement the microprogram-simulator in it's early stage, and verify the microprogram efficiently. In the case of MN10400, we described the Control-Unit of the simulator based on the logical design of the processor(in the bottom-up fashion), and the Execution-Unit based on the behavioral specification(in the top-down fashion). By using this method, we were able to get the flexible microprogram-simulator, and develop the microprogram efficiently.

1．はじめに

半導体技術の進歩により、数１０万ゲートに及ぶ大規模な論理素子がLSIに搭載可能となるに従って、マイクロプロセッサの機能および性能は著しく向上している。しかしその結果として、プロセッサの開発に要する期間は長期化しており、この問題に対処するために効率的な開発方法を用いる必要がある。

マイクロプログラム制御方式をとるプロセッサでは、ハードウェアの論理設計と並行してマイクロプログラムを開発することによって開発期間を短縮している[1]。この際、パイプライン処理を行なうプロセッサにおいては、プロセッサの備えるハードウェア資源やパイプライン構造を反映しマイクロプログラムの正当性を検証するためのシミュレータ（以後、マイクロシミュレータと呼ぶ）が不可欠である。しかしプロセッサのハードウェア設計と並行してマイクロプログラムを開発するには、ハードウェアの設計終了前にマイクロシミュレータを構築する必要がある。そのため、シミュレータ構築後にもさまざまな設計変更が発生する。従ってマイクロシミュレータを構築する際には、シミュレータ構築の早期化とハードウェアの設計変更に対する柔軟性を兼ね備えた実現方法が求められる。

今回、３２ビットマイクロプロセッサMN10400を開発するにあたり、マイクロシミュレータの新しい実現方法を提案し上記課題を解決した。本報告では、まず３２ビットマイクロプロセッサMN10400の概要を紹介し、次にMN10400の開発において作成したマイクロシミュレータの実現方法を説明する。そして最後に、今回提案したマイクロシミュレータの実現方法の優位性について評価する。

2．パイプライン構造の概要

MN10400は[2]、TRON仕様に基づいた３２ビット実記憶プロセッサであり、高速化のためにパイプライン処理を行なっている。パイプラインの段数は４段で、さらにメモリへのオペランド書き込み用にストアバッファを設けている。このパイプライン構造を図１に示す。パイプラインの第１段は命令プリフェッチを行なうIFステージ、第２段は第１の命令解読を行なうDEC1ステージ、第３段はオペランドアドレス計算を行なうOAステージおよび第２の命令解読を行なうDEC2ステー

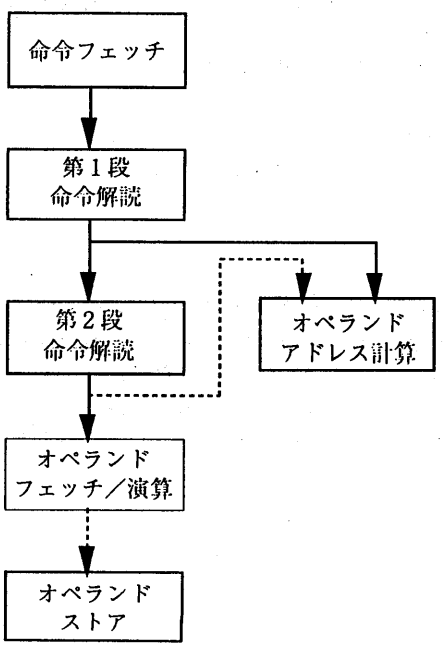

図１　MN10400のパイプライン構造

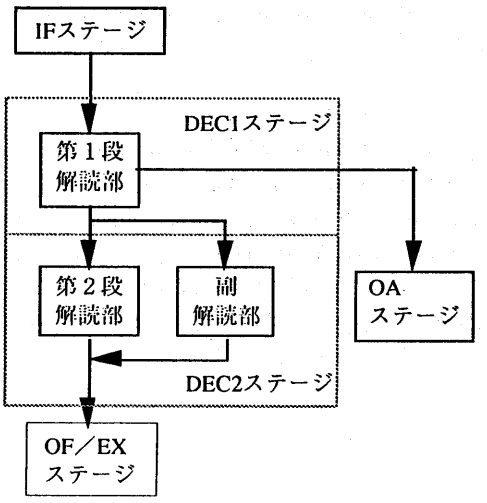

図２　命令解読部の構成

ジ、第4段はオペランドフェッチおよび演算を行なうOF/EXステージである。

このようにMN10400は2段階の命令解読を行い、第1段の解読部はPLA、第2段の解読部はROMで実現している。また、第2段の解読部のROMサイズを縮小するために第2段と並列にPLAによる副解読部を設けている。図2に命令解読部の構成とパイプラインステージの関係を示す。

マイクロプログラムは3つの解読部に分散して格納され、その動作はマイクロ命令による指定やパイプラインの状態[3]によって制御される。従って、このようなマイクロプログラムを検証するためには、マイクロシミュレータが正確なパイプライン制御方式を反映する必要がある。しかし、これらの命令解読部の構造やパイプライン制御方式はMN10400のために新しく考案したものであり、マイクロプログラムと並行して開発されるため、マイクロシミュレータ構築後にも設計が変更される可能性が高い。従って、マイクロシミュレータは、命令解読部やパイプライン制御方式の変更に柔軟に対応できることが必要である。

3. シミュレータ実現方法

3.1 シミュレータ構築の課題

図3にマイクロプロセッサの設計フローの概略を示す。プロセッサを開発するには、まず目標とする性能や機能を満たすようにレジスタなどのハードウェア資源やその制御方法などの大まかな仕様を定義する（仕様設計）。次に、プロセッサをいくつかのサブブロックに分割して（構造設計）、各ブロックごとに仕様を実現するために必要な機能レベルの動作を決定し（機能設計）、さらにその機能を実現する論理回路を設計する（論理設計）。そして最後に、論理設計の結果を反映したゲートレベルのシミュレータなどを用いて詳細な論理検証を行う（詳細検証）。詳細検証の結果により設計変更が発生した場合は、その変更のレベルに従って各設計段階にフィードバックされる。

マイクロシミュレータを実現するためには、上記の設計段階のうちいずれかの設計結果を反映して動作するシミュレータを構築する必要がある。これまでに述べてきたように、マイクロシミュレータ構築における課題は次の3点である。

（1）早期構築
（2）設計変更に対する柔軟性
（3）正確なパイプライン制御の反映

以下においては、マイクロシミュレータを実現するために従来用いられていた2つの方法と、今回MN10400のマイクロシミュレータ実現に用いた方法を説明し、上記課題への対応について考察する。

3.2 仕様設計結果を用いた実現方法

マイクロプログラムを早期に完成するためには、早い段階の設計結果を用いてシミュレータを構築する必要がある。例えば、プロセッサ設計の最初の段階である仕様設計の結果より、ハードウェア設計とは独立にシミュレータを構築する方法[4]がある。この方法によれば、シミュレータはハードウェアの機能設計や論理設計とは独立に構築されるので、詳細検証による設計変更の影響は少なく、シミュレータの修正の柔軟性はあまり要求されない。しかし、プロセッサのパイプライン制御方式を正確に反映することは不可能であり、

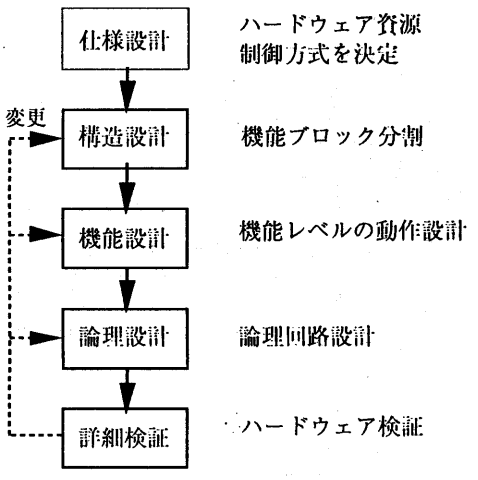

図3　プロセッサ設計フロー

マイクロプログラムの正当性を完全に検証するためには、ハードウェアの論理設計後にパイプライン制御を反映したゲートレベルシミュレータなどを用いてマイクロプログラムを再テストする必要があり、結局はマイクロプログラムの完成が遅くなる。

3.3 機能設計結果を用いた実現方法

大規模論理回路を効率的に設計する手法として階層化設計が注目されている。マイクロシミュレータを構築するには、この階層化設計の過程で作成される機能レベルシミュレータを利用する方法がある。

階層化設計では、プロセッサ全体を機能的にまとまった複数のブロックに分割し、ブロック間のインタフェース信号を定義する。各ブロックを詳細論理設計が可能なサブブロックにまで繰り返し分割した後、各サブブロックごとにその動作を機能レベルで厳密に記述し検証する。そして、機能レベルの検証が終了したブロックは、論理レベルへと順次トップダウン的に設計し、機能レベルと論理レベルの設計間で動作の一致を確認する。

この手法を用いてマイクロプログラムを早期に開発するには、各ブロックの機能設計で作成した機能レベル記述を用いてマイクロシミュレータを構築する。図4に階層化設計でのマイクロシミュレータの実現方法を示す。

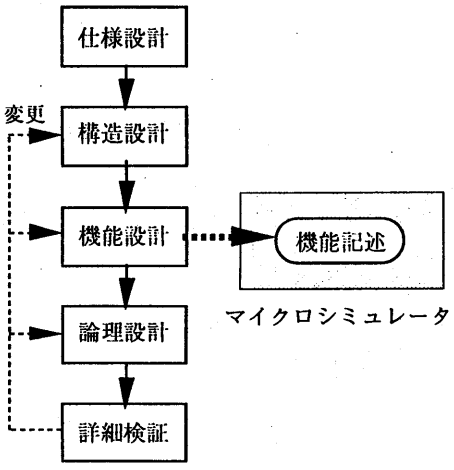

図4　階層化設計による実現方法

この方法では、機能レベルで記述したシミュレータの動作は論理レベルの回路の動作と完全に一致するため、マイクロシミュレータは正確なパイプライン制御を反映する。また、論理設計前に機能レベルの記述を完全に作成するため、マイクロシミュレータを早期に構築できる可能性がある。しかし、回路全体を一度に把握することが困難な大規模論理回路の設計では、設計が完了する以前にサブブロックの詳細な仕様やインタフェースを決定することは不可能に近く、ブロック分割を中心として設計を進める階層化設計の手法を簡単に用いることはできない[5]。特に、プロセッサのパイプライン構造や命令解読方式までを新規に開発する場合、機能レベルから論理レベルに詳細設計を進める際に、ブロックの機能に変更が必要となったり、基本となるブロック分割やブロック間インタフェースにまで変更が生じる可能性が高い。このため、ブロック分割やブロックの機能定義に変更があるたびに再度トップダウン的なシミュレータの修正が必要となり、結果として論理設計完了までシミュレータが完成せず最終的な設計工数の増加を招く。

3.4 MN10400における実現方法

MN10400ではマイクロシミュレータを実現するにあたり、従来の階層化設計のようにシミュレータ全体をハードウェアのブロック分割に対応させながらトップダウン的に作成するという方法ではなく、プロセッサのパイプライン制御に関するブロックを分離し、その部分だけはハードウェアの論理設計の結果からボトムアップ的にシミュレータを作成するという方法を採った。図5に本シミュレータの実現方法を示す。

本実現方法では、プロセッサをパイプライン制御および命令解読を行なう制御部と、制御部から発行されたマイクロ命令を受け取って実行する実行部とに明確に分離し、制御部はパイプライン制御構造とマイクロプログラムの関係を忠実に実現するために、パイプライン制御回路の論理設計後

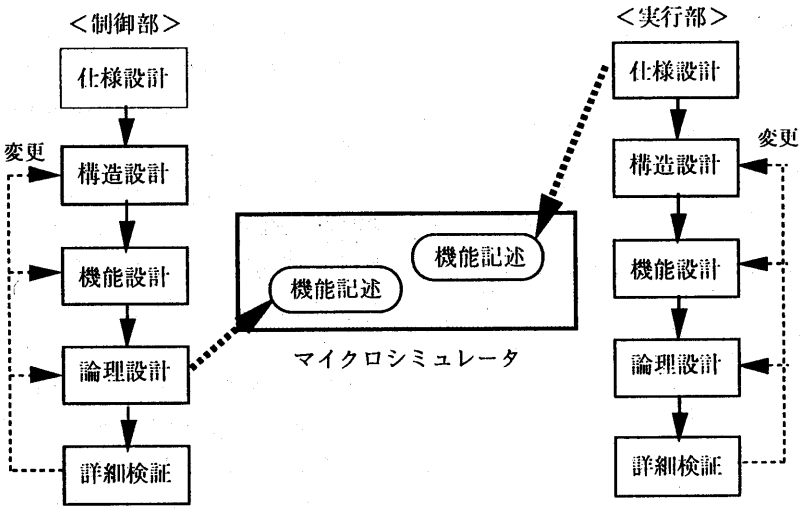

図５　MN10400の実現方法

に論理回路に基づいてマイクロシミュレータの機能記述を作成した。また、マイクロ命令の制御対象となる実行部の機能は仕様設計時にほとんど決定するため、ハードウェアの設計とは独立に仕様設計のみからマイクロシミュレータの記述を作成した。

本実現方法の場合、シミュレータの構築を開始する時に必要となるのは制御部と実行部の基本的なブロック分割のみに限定される。このブロック間インタフェースはプロセッサ設計の基本コンセプトにかかわるものであり、設計初期に確定し、シミュレータ構築後に大幅な変更は有り得ない。それに対し、論理設計途中や詳細検証時に発生するハードウェアの設計変更は、実行部に関してはシミュレータがハードウェア設計とは独立に構築されるためシミュレータの記述に影響を与えず、制御部に関しては論理設計の修正結果を論理回路から直接シミュレータ記述に反映することが可能である。

このように実際のハードウェアからボトムアップ的に構築した制御部と、要求される仕様からトップダウン的に構築した実行部を結合してシミュレータを実現することにより、制御部と実行部の基本的なインターフェースが決定した段階でシミュレータの構築が開始でき、その後のパイプライン制御論理の修正に対しても柔軟に対応可能となる。また、マイクロシミュレータの実行部は、ハードウェアのブロック分割とは関係なく、マイクロ命令で指定される演算ごとに記述する。そして、マイクロプログラムの開発経過に合わせてシミュレータの演算機能を追加して行くことにより、シミュレータによるマイクロプログラムの開発を早めることができる。

４．評価

本章では、３２ビットプロセッサMN10400のマイクロシミュレータ実現方法について、シミュレータ記述量・構築期間・設計変更の影響の３つの観点から評価する。

4.1 記述量

MN10400のマイクロシミュレータの記述量を評価するために、階層化設計で作成される機能レベルシミュレータの記述量と比較した。比較に用いた機能レベルシミュレータの記述量は、MN10400の最終的な論理設計のブロックごとに、動作を機能記述して見積った。表１に各シミュレータの記述量を示す。

MN10400のマイクロシミュレータの記述量はソースプログラムで８８００ステップとなった。こ

れは、シミュレータ全体をハードウェアのブロック分割に対応させて作成した機能レベルシミュレータの約６０％の記述量である。

表１　シミュレータ記述量

	MN10400 シミュレータ (ステップ数)	機能レベル シミュレータ (ステップ数)
制御部	４５００	４５００
実行部	４３００	１００００
合計	８８００	１４５００

4.2 構築期間

今回提案したシミュレータ実現方法では、プロセッサのパイプライン制御回路の論理設計が終了した時点でシミュレータの構築が可能となる。従って、パイプライン制御回路の論理設計期間が長い場合、シミュレータの構築が遅れマイクロプログラムの早期開発は不可能となる。実際には、MN10400のパイプライン制御回路のトランジスタ数は約２６００で、これはプロセッサ全体のトランジスタ数の１％未満であり、その論理設計に要した期間はプロセッサ論理設計の初期の約２ケ月のみである。図６にマイクロシミュレータの構築期間をプロセッサの設計と対応させて示す。

本マイクロシミュレータの基本機能を実現してマイクロプログラムの検証を開始するまでに要した期間は約４ケ月（工数４人月）であり、その以後、機能追加によりシミュレータ全体の完成までに要した期間は約８ケ月（工数８人月）である。MN10400と同程度の機能と回路規模を有するマイクロプロセッサ[6]を階層化設計手法を用いて開発した場合、機能設計を終了して機能レベルシミュレータを構築するまでに要する期間は約１２ケ月（工数３６人月）であり、本実現方法によってマイクロプログラムの検証開始を約８ケ月早めることができた。

4.3 設計変更の影響

本実現方法はプロセッサの制御部と実行部の分割によって決定される主要インタフェースに基づいてシミュレータの実行部を構築するため、その主要インタフェースに変更が多発するとシミュレータ構築が困難になる。また、シミュレータの制御部はハードウェア論理設計に基づいて構築する

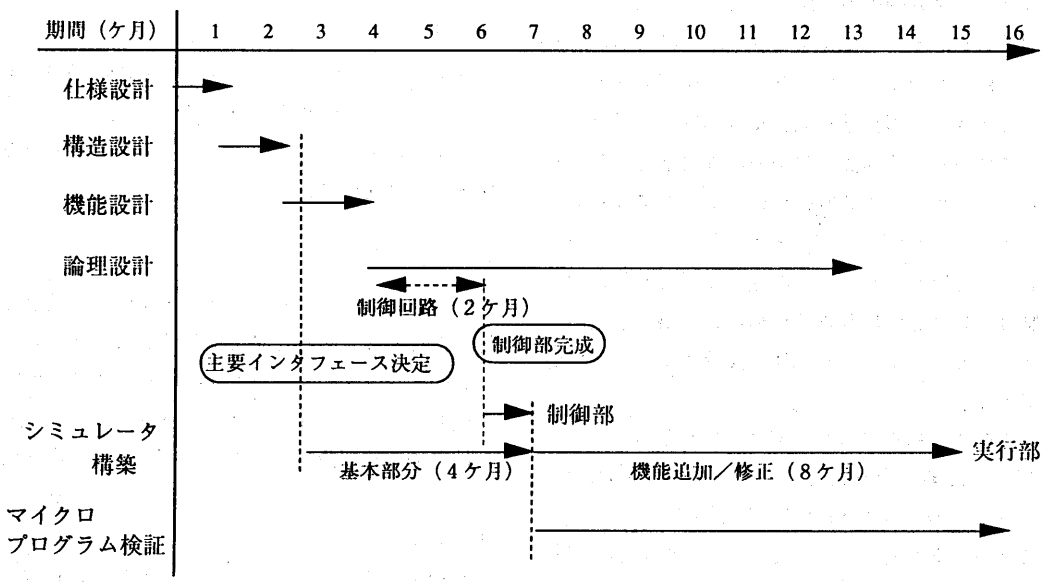

図６　マイクロシミュレータ構築期間

表2　ハードウェア設計変更の発生割合

	制御部（%）	実行部（%）	合計（%）
論理設計ミス	7.7	16.3	24.0
ブロック機能変更	17.6	24.9	42.5
ブロック間 インタフェース変更	6.8	22.2	29.0
制御部・実行部 主要インタフェース変更	4.5		4.5

表3　シミュレータ修正の割合

ハードウェア変更箇所		シミュレータを修正した変更の割合
制御部	論理設計ミス	100%
	ブロック機能変更 インタフェース変更	100%
実行部	論理設計ミス	0%
	ブロック機能変更 インタフェース変更	21%
主要インタフェース変更		100%

ため、制御回路の論理設計ミスもシミュレータに反映される。従って、本実現方法によってマイクロシミュレータを構築するためには、主要インタフェースの変更や制御回路の論理設計ミスが少ないことが前提となる。表2にMN10400の詳細検証によって発生したハードウェアの設計変更の変更箇所別の発生割合を示す。表2よりわかるように、問題となる主要インタフェース変更および制御回路の設計ミスは、それぞれ設計変更全体の4.5%,7.7%と少なく、本実現方法を用いることによる問題は少ないと考えられる。

それに対して、本実現方法ではシミュレータの実行部をハードウェア設計とは独立して構築するため、実行部の設計変更のうちシミュレータの修正が必要なものは一部のみである。表2からわかるように、実行部のブロック機能変更やインタフェース変更は設計変更全体の50%近くを占めるためこの効果は大きい。設計変更箇所ごとの、ハードウェア設計変更のうちシミュレータ修正を要する変更の割合を表3に示す。表3よりわかるように、実行部のブロック機能変更やインタフェース変更のうち、シミュレータに影響するものは20%程度のみである。本実現方法を用いることによって、プロセッサの設計変更の約半分を占める実行部のブロック設計変更の約80%について、シミュレータ修正を減少することができた。

5．まとめ

今回、制御部を実際のハードウェア設計からボトムアップ的に作成し、実行部を機能設計からトップダウン的に作成するマイクロシミュレータの実現方法を提案し、３２ビットマイクロプロセッサMN10400の開発に適用した。

本シミュレータの作成にはハードウェア記述言語を用い、ステップ数は約９０００、シミュレーション実行速度は実時間の０.５×１０^{7}倍となった。また構築期間は基本機能の実現に約４ケ月、全体の完成に約８ケ月であり、本シミュレータを用いてプロセッサのハードウェア論理設計と並行してマイクロプログラムの検証を進めることにより、早期にマイクロプログラムを完成することができた。

謝辞

本研究の機会を与えて頂きました松下電器産業情報通信関西研究所出口室長に感謝します。また、本研究に関して有意義な助言を頂きました同研究所清原主任技師，西川技師に感謝致します。

参考文献

1）追田行介　他,"VLSIにおけるマイクロプログラム設計支援",情報処理学会,Vol.25,No.10,pp.1041-1047,1984

2）T.Kiyohara et al.,"Design Consideration of the Matsushita 32-Bit Microprocessor for Real-Memory System",TRON Project,pp.263-273,1988

3）宮崎雅也　他,"TRON仕様３２ビットマイクロプロセッサMN10400のパイプライン制御方式とテスト手法",電子情報通信学会　ICD90-3,pp.15-22,1990

4）橋本幸治　他,"Gmicro/200におけるマイクロプログラムの評価手法",トロン技術研究会,Vol.2,No.1,pp.1-10,1989

5）宮田操　他,"階層的ハードウェア設計言語H^{2}DLの思想",情報処理学会　設計自動化22-1,pp.1-8,1984

6）宇佐美公良　他,"フルカスタムマイクロプロセッサの設計手法最適化",情報処理学会　設計自動化47-ﾊﾞ1,pp.1-4,1989