TRON仕様32ビットマイクロプロセッサGMICRO/100

2U-6　（2）関係データベースによるマイクロプログラムの管理とマイクロデコーダの設計

渡辺由香里　清水徹　岩田俊一　斎藤祐一　吉田豊彦　宮沢治

（三菱電機株式会社　ＬＳＩ研究所）

1．はじめに

ＴＲＯＮ仕様に基づく３２ビットマイクロプロセッサGMICRO／１００ではマイクロプログラム制御方式を採用している。高機能命令の高速実行がその目的の一つである。またマイクロプログラム制御方式の採用によりアーキテクチャの仕様変更、ハードウェアの改訂をマイクロプログラムで吸収することができる。その結果アーキテクチャ設計、ハードウェア設計、マイクロプログラムの設計を並行に進めることができる。

ここで重要なことはアーキテクチャ、ハードウェアの変更をマイクロプログラムですばやく吸収することである。これはハードウェアの規模が大きくなるにつれて困難になってくる。我々は関係データベースをマイクロプログラムの管理、更にマイクロデコーダの設計に用いることによりこの問題の解決を図った。本稿ではGMICRO／１００のマイクロプログラム、マイクロデコーダの設計手法について述べる。

2．GMICRO／１００開発の流れ

GMICRO／１００の開発では、アーキーキテクチャの設計、ハードウェアの設計、マイクロプログラムの設計をほぼ並行して行った。マイクロプロセッサの基本となるアーキテクチャ設計でハードウェアやマイクロプログラムのアーキテクチャを検討し、仕様がほぼ確定した段階でハードウェアやマイクロプログラムの設計を開始する。アーキテクチャ設計の細部の機能検証も同時に実施され、その過程でアーキテクチャ仕様の変更が発生するとハードウェアやマイクロプログラムに影響を与える。

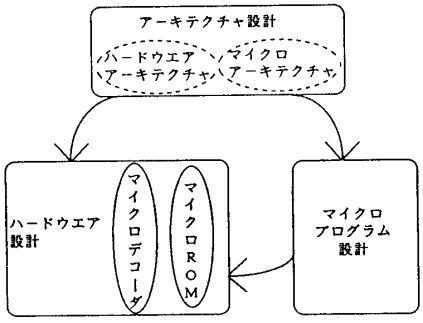

図１　GMICRO/100開発の流れ

この仕様の変更にすばやく柔軟に対応できる開発体制を築くことが、短期開発のために必要となる。我々は、関係データベース管理システム（INGRES）を使ってマイクロプログラムを管理することによってこの問題の解決を図った。

3．関係データベースの利用

マイクロプログラムはハードウェアの各機能ブロックを制御するものであり、２クロックサイクルで実行される各マイクロ命令をシーケンシャルに流してマクロ命令を実行する。このマイクロ命令をいくつかのフィールドに分割し、それらを関係データベースのフィールドとして管理する。

マイクロプログラムを柔軟に管理するには人間が目でみて分かりやすいことが必要である。そのためにマイクロプログラムのフィールドの値をハードウェアの機能を表現するようなニモニックで表し、関係データベースのデータとした。

一方マイクロデコーダの開発のために、各マイクロオペレーションとそれに対応する制御信号線を関係データベースで管理した。マイクロオペレーションの関係と制御信号線の関係をつなげれば、マイクロ命令に対する制御信号の一覧が得られる（図２）。各信号線を関係データベースのフィールドとし、マイクロデコーダの設計に使用した。

operation	ls	f1	f2	cl	ro	se	rs	hb	sh	bs	sb
SFT	0	-	-	-	-	-	-	-	-	0	0
so<<s2->ba2	0	0	1	0	0	0	0	1	0	0	1
so<<s2=>do	0	0	1	0	0	0	0	1	0	1	0
soros2	0	0	1	0	1	0	0	0	0	0	0
soshs2->bb2	0	0	1	0	0	0	0	0	0	0	1

↑ マイクロオペレーション　　↑ 制御信号

図２　マイクロオペレーションと制御信号

4．マイクロプログラムの開発

マイクロプログラムの開発は、まずチップの持つ命令セットに基づきアルゴリズムを考え、これをフィールド毎に関係データベース化する。マイクロ命令に対する最初のフィールド構成は設計が進むにつれて分割、追加等手が加えられていく。これらは関係データベースフィールドの分割、追加そのものであり、データベースの操作によりスムーズにマイクロプログラムの変更を行うことができた。

またマイクロ命令のフィールドの統合にも有効であった。一例を図３に示す。最初、演算のサイズを制御するマイクロプログラムのフィールドは機能的に１１

Implementation of a 32-bit microprocessor GMICRO/100 based on TRON specification (2) Management of microprogram and design of microdecoder by relational database

Yukari WATANABE,Toru SHIMIZU,Shunichi IWATA,Yuichi SAITO,Toyohiko YOSHIDA,Osamu TOMISAWA
MITSUBISHI ELECTRIC CORPORATION

個であった。これを関係データベースの検索機能を利用して最少化した。例えばＡＬＵのSIZE-Aフィールドで指定するサイズと、バレルシフタのSIZE-Dの指定とが常に等しいか否かの検索、アドレスレジスタ制御用のサイズSIZE-Sとメモリのアクセスサイズ SIZE-Uが"don't care"を利用して統合できるか否かの検索などを行うことができる。その結果、機能的には１１個であったサイズフィールドを５個に畳み込むことができた。

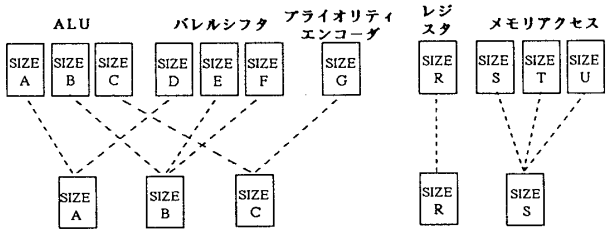

図３　サイズフィールドの統合

更に、関係データベースでは複数のマイクロオペレーションの組合せを検索することができ、それによりハードウェアで同時に起こってはいけない事象を使用していないかどうかを検査することができた。

５．マイクロデコーダの開発

マイクロデコーダはマイクロプログラムと演算器等のハードウェアの間のインターフェースをとるものであり、マイクロプログラムを格納したマイクロＲＯＭのデータを制御信号にデコードする（図４）。

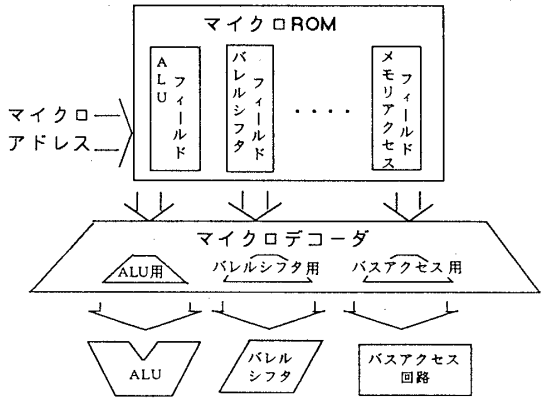

図４　マイクロデコーダの位置づけ

マイクロデコーダの設計に関して、マイクロプログラムのフィールドをそのままデコーダの分割に使うとマイクロＲＯＭのビット数が増えてしまう。そこでマイクロＲＯＭのビット数を削減するため、マイクロプログラムのフィールドをマイクロデコーダからみたフィールド（以下サブフィールドと呼ぶ）に分け直すことにした。サブフィールドに分け直す際、関係データベース内のマイクロプログラムのフィールドを組み合わせて、マイクロオペレーションの組合せを検索し、最適なサブフィールドを決定した。サブフィールドがマイクロのフィールドと一致する場合も、複数のフィールドが一つのサブフィールドになる場合もある。

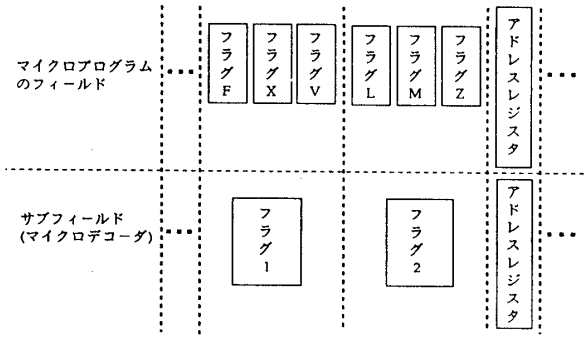

図５　マイクロプログラムのフィールドとサブフィールド

マイクロプログラムのフィールドに対応する制御信号１本づつを関係データベースのフィールドとして入力し制御信号のデータベースを作った。マイクロプログラムのデータベースと制御信号のデータベースをつなぎ合わせることで、各サブフィールドのマイクロオペレーションの組合せに対応する制御信号の組合せのリストをとり、これを元にマイクロデコーダを設計した。制御信号をまとめるときに関係データベースのフィールドを入れ替えることで制御信号の並べ替えを行い、最適なマイクロＲＯＭのビット割付を実現した。その結果マイクロＲＯＭのビット幅を約半分にできた。それにつれて、マイクロデコーダの規模を小さくすることができた。

６．結論

ＴＲＯＮ仕様３２ビットマイクロプロセッサGMICRO／１００の開発に当たり、マイクロ関連の開発に関係データベース管理システム（INGRES）を利用した。

マイクロプログラムの開発に関係データベースを利用したことにより、マイクロオペレーションのフィールド構成の最適化を図り、アーキテクチャの変更、ハードウェアの変更に柔軟に対応することができた。またマイクロデコーダの設計に関係データベースを利用したことで、ハードウェアの機能ブロックとマイクロプログラムの関係を解りやすくし、マイクロＲＯＭのビット幅を約半分にでき、マイクロデコーダの規模を小さくすることができた。

謝辞

ＴＲＯＮプロジェクトをご指導頂いている東京大学坂村助教授をはじめ有意義な討議と助言を頂いたGMICRO／１００の開発に携わっている方々に感謝致します。

（参考文献）

Robert Epstein "A TUTORIAL ON INGRES" December 15,1977 ,University of California, Berkeley