計算機設計評価システム - MPGS／GPMS

藤野喜一　箱崎勝也　服部光宏　山本昌弘
矢野美智子　梅村護　（日電　中研）

1. まえがき

計算機のデザインオートメーションシステムは現在広く使用されており、計算機の設計のために不可欠である。しかしながら、デザインオートメーションシステムが使用されている分野は現在では主として論理設計後のプリント板やバックボードの配線表や診断のためのテストパターンの発生に限られており、設計の質的評価の道具としては用いられていない。設計の評価のために多くのソフトウェアシミュレータ[1]が開発されているが、論理設計レベルのシミュレーションや大型計算機のシミュレーションに用いると非常に時間がかかる。またソフトウェアシミュレータを用いるためには実際の機械の設計以外にモデル化や特別な記述が必要で計算機の設計者に非常に負担になるため充分利用されていない。

ここで述べる計算機設計評価システムは計算機の設計方式を高速かつ定量的評価データに基づいて広範囲に評価しながら、評価が満たされると最後に製造ドキュメントを作る。例えばレジスタやデータパスの最適数，マイクロプログラムの方式、その他各種制御方式等が定量的評価によって決定される。更にこのMPGS／GPMS システムは高級言語計算機，バーチャルマシン，ソフトウェア／ハードウェア トレードオフの評価等の将来の計算機の問題を検討するためにも使用できる。

本システム[2]は２つのサブシステムから成っている。

(1) ＭＰＧＳ（Micro Program Generating System）[3]

ソフトウェアサブシステムで機械の記述をシミュレーション及び製造のために、実行可能なマイクロプログラムへ翻訳する。また機械の変更や評価項目の変更を高速に行う機能をもち、各設計レベルで必要なドキュメントを作る。

(2) ＧＰＭＳ（General Purpose Microprogrammed Simulator）[4]

ハードウェアサブシステムで非常に高速度でシミュレーションを行い、多くの動的な評価情報を収集する。

2. ＭＰＧＳ／ＧＰＭＳシステム

2.1 目　的

計算機の設計は通常実験的モデルの設計，評価及び修正をくり返し行うことによって達成され、満足な評価が行われると製造モデルのための最終ドキュメントが得られる。しかし計算機の設計及び評価を行う道具が少いために、このようなサイクルをくり返すのに非常に時間がかかり、充分な評価がされないまま製造されている。ＭＰＧＳ／ＧＰＭＳシステムの大きな目的は設計評価サイクルに必要な時間を減らし、充分な評価を行った良い設計を達成することである。

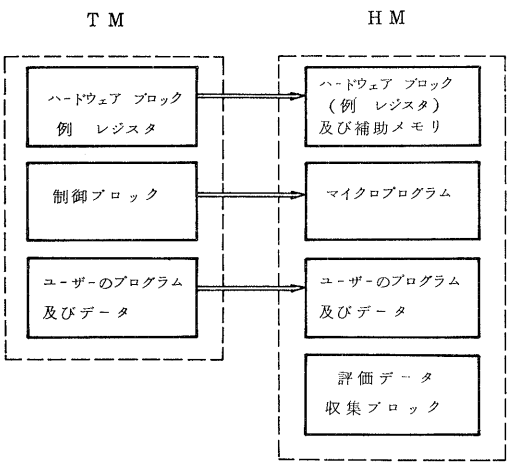
2.2 シミュレーションの方法

今シミュレートされるべき１つの機械であるターゲットマシン（ＴＭ）をシミュレートする時に、ＴＭが持っている機能をホストマシン（ＨＭ）であるＧＰＭＳが持っている機能へ等価的に対応づける。第１図に示すように、

例えばＴＭのハードウェアブロックはＧＰＭＳのレジスタやメモリへ対応づけ、制御ブロックは等価なＧＰＭＳマイクロプログラムへ変換し、ＧＰＭＳのマイクロプログラムメモリへロードする。ＴＭがマイクロプログラム計算機でなくても、制御シークエンスはマイクロプログラムとして等価的に表現できる。ＴＭが持っている加算器，シフト器，論理演算器等はＧＰＭＳの論理演算ユニットを用いてマイクロプログラムへ翻訳される。ＴＭの周辺デバイスはＧＰＭＳの周辺デバイスや擬似入出力動作（周辺デバイスと実際に入出力動作を行わずに、入出力時間だけを制御することによって模擬する）でおきかえられる。この時点でＧＰＭＳはＴＭのバーチャル イメージとして働くことができる。シミュレーションのために使用される評価のためのプログラム及びデータは何ら変換せずにＧＰＭＳの主メモリへロードされる。

ＴＭとして働くと同時に、ＧＰＭＳは評価データ収集ハードウェアによって各種の評価データを収集する。更に多くの詳細な評価情報の収集は変換されたＴＭと等価なマイクロプログラムの中へ収集用の特別なマイクロプログラムを挿入することによって可能である。

シミュレーションの結果，要求がみたされるとＭＰＧＳは製造用ビットパターンとしてＴＭのマイクロプログラムを作る。ＴＭの記述やＧＰＭＳへの変換規則は高級言語であるＭＰＧＳ言語で書かれる。ＭＰＧＳはそれを翻訳し、ＧＰＭＳ用マイクロプログラムを発生する。ＴＭの記述はシミュレーションの目的に依存して、マクロにもミクロにも可能である。ＧＰＭＳはＴＭのシミュレーションと評価データの収集を主なる目的とし、ソフトウェアシミュレータのハードウェア版であるのでそれに比べて高速なシミュレーションができる。



第１図　ＴＭとＨＭの対応づけ

3. ＭＰＧＳ

ＭＰＧＳはマイクロプログラム記述のための高級言語とファイル保守機能を備えたトランスレータから成っている。ＭＰＧＳトランスレータはＭＰＧＳ言語で書かれたＴＭの記述をＨＭ上の等価なマイクロプログラムへ翻訳する。ＭＰＧＳを決定する上で以下の点を考慮した。

(1) ＭＰＧＳはＧＰＭＳ及び他の汎用マイクロプログラム計算機のマイクロプログラムを記述する道具として使用できる。ＧＰＭＳ用の評価情報収集用マイクロコマンドをソースプログラムへ挿入することが容易にできる。

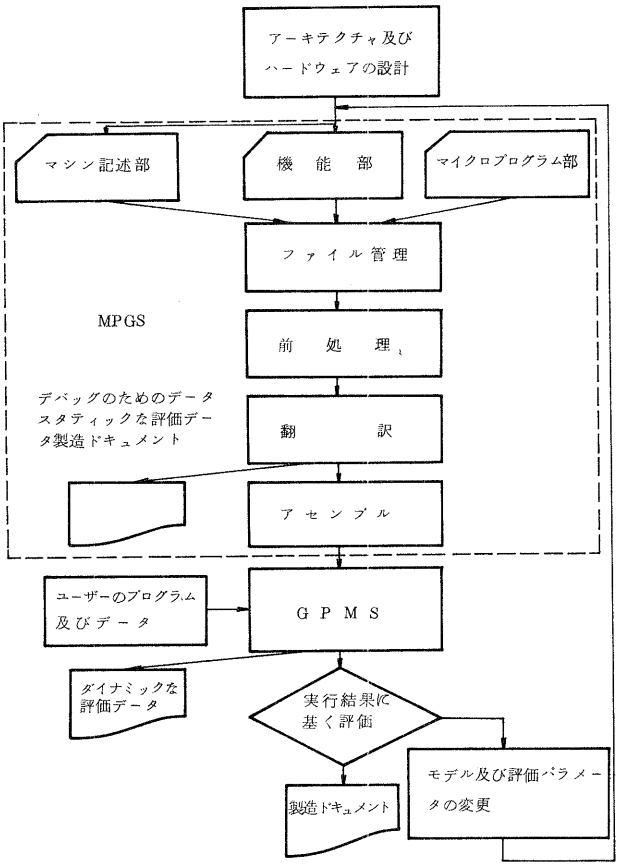
(2) ＭＰＧＳ言語で書かれたプログラムは設計の途中で設計者によってしばしば参照されるので、この言語はドキュメント性がよくなければいけない。

(3) プログラムへの高速なアクセスや修正が設計者自身はもちろん、他の設計者によっても可能である。

(4) 効率よいオブジェクトコードが発生される。ユーザーはオブジェクトコード発生規則を記述することによって、部分的に最適化されたオブジェクトコ

－ドを得ることができる。

上記の点以外に、ＨＭのマイクロ命令セットを充分利用できるようにＭＰＧＳはオブジェクトコードに対して、柔軟なものとした。すなわちＭＰＧＳではＨＭのマイクロ命令のためのシンボリックコードは固定されず、ユーザー自身が定義できる。



第２図　MPGS／GPMS システム

第２図のマシン記述部ではＴＭのハードウェア（例えばレジスタ，サブレジスタ，マスク）が宣言され、且つＴＭとＨＭ間の対応づけが行われる。機能部ではＴＭの制御ブロックとＨＭのマイクロ命令間の変換規則が記述される。マイクロプログラム部ではＴＭの制御ブロックが記述され、マシン記述部と機能部の情報に基いて、ユーザー自身によって定義されたシンボリックコードで書かれたマイクロプログラムへ変換される。このマイクロプログラムはマイクロプログラムアセンブラによってビットパターンへ翻訳される。

3.1　ＭＰＧＳ言語

ＭＰＧＳプログラムはマシン記述部，機能部，マイクロプログラム部から成っている。

3.1.1　マシン記述部

メモリ，レジスタ，サブレジスタ，マスク等のＴＭが持っているハードウェアの宣言が行われる。

```
DECLARE     R(24),
            R0=R(0-3),
            R1=R(4-23);
            X(4);
DEFINE      R=SPM#1-2;
            X=SPM3(4-7);
```

上の例ではＲ及びＸは各々２４，４ビットのＴＭのレジスタで、またR_0,R_1はＲのサブレジスタで各々４，２０ビット長である。マイクロプログラム部

でＲ，Ｘが用いられると、ＤＥＦＩＮＥ部の定義に基いてＲ，ＸはＧＰＭＳの以下で示すＩＣメモリの語と見做される。

Ｒ＝ＳＰＭ１（０－１５），ＳＰＭ２（０－７）；

Ｘ＝ＳＰＭ３（４－７）；

ＴＭのレジスタがＧＰＭＳの１語の一部に対応づけられている時に、そのレジスタを処理する場合にはＧＰＭＳではマスクを用いて扱われるが、このためにトランスレータはマシン記述部で宣言されたマスクパターンをサーチすることによって対応するマスクを自動的に選択する。例えば上例のＸではマスクパターン"０００００１１１００００００００"が選択される。

3.1.2　機能部

機能部はＴＭの制御ブロックとＨＭのマイクロ文間の変換規則を変換制御文を用いて制御ブロックの機能単位でサブルーチンとして定義する。サブルーチン内にはマイクロ文，変換制御文，サブルーチン呼び出し文，シミュレーション制御文が記述される。これらのサブルーチンはマイクロプログラム部や他のサブルーチンによって呼ばれると実パラメータに従って対応するマイクロ文が選択されマイクロプログラムに挿入される。サブルーチンは宣言部とオペレーション部から成り、宣言部はサブルーチン名，パラメータ，パラメータの属性と範囲，そのサブルーチンが使用する作業エリアを定義する。オペレーション部は実パラメータに従ってサブルーチンがどのようなマイクロ文へ変換されるかを変換制御文を用いて記述する。マイクロ文はＴＭのマイクロ命令をシンボリックに表現したもので、単なる文字列でどのような形式も許され、特殊文字"，"，"．"，"／"，"＄"，"＝"で区切られている。ＴＭのレジスタ　Ａ（３２），Ｂ（３２），Ｃ（３２）　が宣言され、ＴＭとＨＭの対応づけが以下のようにされているとする。

```
DEFINE    A＝SPM#1－2；
          B＝SPM#3－4；
          C＝SPM#5－6；
```

ここでマイクロ文　Ｃ＝Ａ／ＡＮＤ／Ｂ；　　と記述されるとＨＭのハードウェアへ変換され、且つ以下のように展開される。

```
SPM6 ＝ SPM2／AND／SPM4；
SPM5 ＝ SPM1／AND／SPM3；
```

変換制御文として代入文，条件文，条件付代入文，ＤＯ-ＬＯＯＰ文，ＥＸＰＡＮＤＩＮＧ-ＤＯ　文等がある。

```
@OPR ＝ ADD0    @IF  X＝0，
     ＝ ADD1    @IF  X＝1，
     ＝ ADDC    @IF  X＝CARRY；
```

上記の例は条件付代入文の一例で、変換制御変数ＯＰＲは変換制御変数Ｘが対応するブール式を満たすようにＡＤＤ０，ＡＤＤ１，ＡＤＤＣ　の内から選ばれる。上の例で示すように条件付代入文を用いることによって、ドキュメント性をよくし、プログラムを書き易く見易くし、ＧＯ-ＴＯ文の少いプログラムを書くことができる。

3.1.3　マイクロプログラム部

ＴＭの制御ブロックの動作はマイクロプログラム部で記述される。マイクロプログラム部の最初の部分でマイクロプログラム名，一時作業用レジ

スタ，評価データ収集のために挿入されるべきマイクロコマンドが宣言される。マイクロプログラム部はシークェンス，ステップ，ステートメントから成っており、シークェンスは制御ブロックの小区分に対応し、名付けられたシークェンス名で互いに参照される。1つのシークェンスは複数個のステップから成り、ステップは少くとも1つのステートメントから成っている。ステップはTM上の通常1クロックサイクルで実行される制御単位で、マイクロプログラム式TMの1つのマイクロ命令に相当する。マイクロプログラム部のステートメントは最小の記述単位で機能部で定義されたサブルーチンを呼び出す文やユーザーによって定義されたHMのマイクロ文である。マイクロプログラム部の1ステートメントはHMの1つまたは複数個のマイクロ文へ拡張される。

上で述べたようにMPGSプログラムはプログラム構造，マイクロプログラム，シークェンス，ステップ，ステートメントという階層をとることによって、変換時に各レベルでのマイクロプログラムに関する統計データを収集することが容易で、またTMを評価するための動的データ収集のために、対応するレベルのソースプログラムへマイクロコマンドを挿入することが容易にできる。

MPGS言語では2つのレベルの相対レーベルを書くことができる。サブルーチン部内で現れた時にはレーベルの領域はサブルーチン内に限定され、アドレス計算はステートメント数に対して行われる。一方マイクロプログラム部内に現れた時にはレーベルの領域はそのマイクロプログラム内に限定され、アドレス計算はステップ数に対して行われる。

3.2 MPGSトランスレータ

このシステムを効率よく使用するために効果的なファイル保守機能とトランスレータオプション機能を備えている。MPGSトランスレータのファイル保守機能を用いて設計者はプログラムを高速に容易に修正することができる。またすでにマスタファイル中に作られているプログラムを適当に選択して各種のプログラムを発生することによって、設計者は多くの異った実験的モデルの設計やシミュレーションを行うことができる。さらにトランスレータオプション機能を用いて、ソース及びオブジェクトプログラムに関する統計的データ（例えばクロスレファランステーブル）を収集したり、出力リストを出したり、レーベルアドレステーブルを作ることができる。

トランスレータはコントローラーと3つの論理的フェーズから成っており各論理的フェーズはいくつかの物理的フェーズから成っている。

MPGSはシステムの記述言語BPL（PL／1のサブセット）で書かれており、BPL言語で約25Kステップである。又これはNEAC2200モデル500で動作する。

4. GPMS

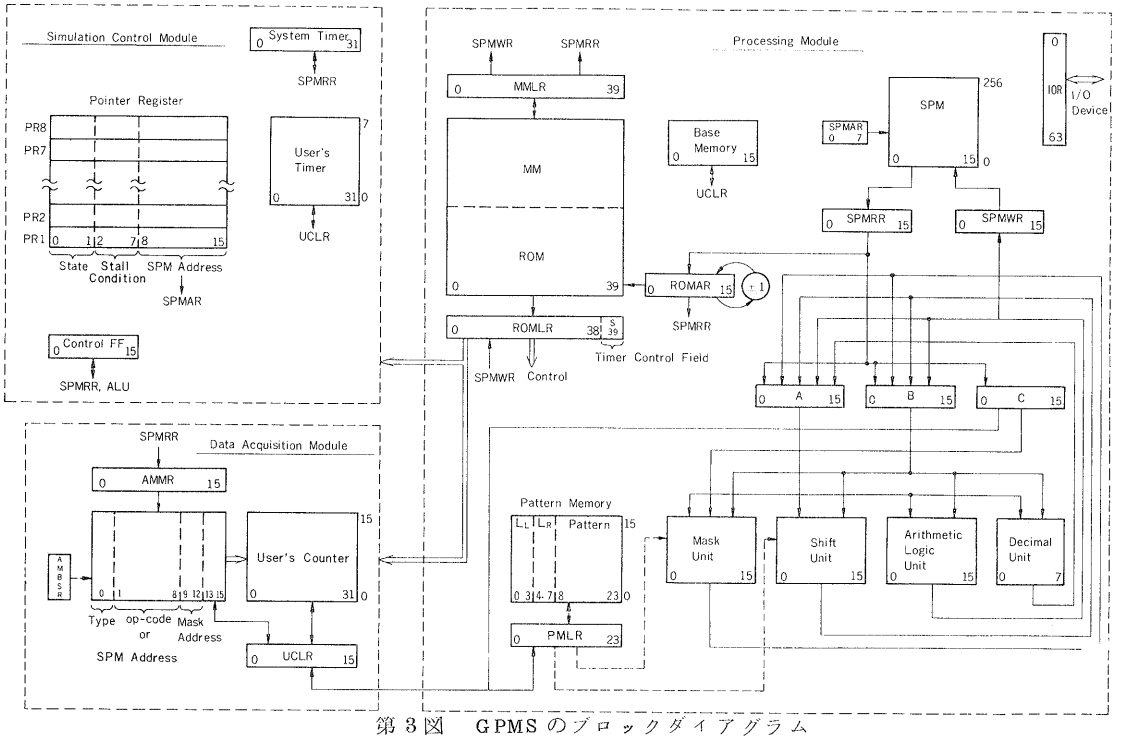
GPMSはMPGS／GPMSシステムのハードウェアサブシステムで、マイクロプログラム制御のハードウェアシミュレータであるため、いかなる種類の計算機でも高速にシミュレートすることができる。GPMSは3つのモジュールから成っており、処理モジュールはTMのデータ処理機能を高速にシミュレートし、シミュレーション制御モジュールはシミュレーションプロセスの制御を

行い、データ収集モジュールはハードウェアリソースの使用率を測定するものである。

シミュレーションに先だって、ＭＰＧＳによって作られたマイクロプログラムはＧＰＭＳのマイクロプログラムメモリへロードされ、ＴＭの評価プログラム及びデータはＧＰＭＳの主メモリ又は入出力デバイスへロードされる。そのマイクロプログラムが評価プログラム及びデータに従ってＧＰＭＳで実行されると最後にはシミュレーション時間やＴＭのハードウェアリソースの使用率が測定される。またＧＰＭＳは汎用計算機としても使用できる。

4.1　処理モジュール

このモジュールは16ビット長のデータ処理ができる演算処理ユニットで、第3図に示すように、主メモリまたはマイクロプログラムメモリとして使用されるサイクル時間1.6マイクロ秒，1語40ビット，容量16K語のコアメモリ，複数個のレジスタ，ＴＭが持っているレジスタや作業用レジスタとして用いられるサイクル時間100ナノ秒，1語16ビット，容量256語のＩＣメモリ，16ビット長の論理演算ユニット，32ビット長のシフトユニット，1語の一部分をぬき出すための16ビット長のマスクユニット等から成っている。マスクユニットを使用するために、サイクル時間100ナノ秒で16ビット長の16種類のマスクを保存するマスクメモリが備えられており、マイクロ命令で適当に選択することによってレジスタやＩＣメモリの1語のいかなる部分でも容易に処理することができる。



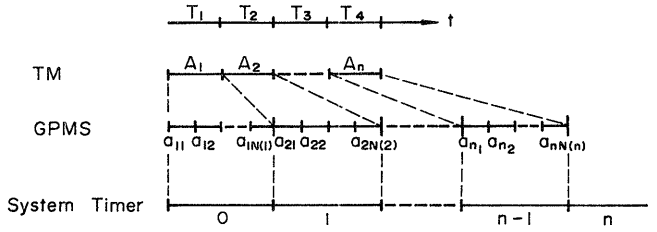
第3図　GPMSのブロックダイアグラム

マイクロプログラムメモリは書き変え可能なため、ＴＭのモデルや評価パラメータを容易に変えることができ、容量はシミュレートされるモデルの大きさに従って主メモリと合わせた合計が１６Ｋ語まで可変である。マイクロ命令は40ビットから成り、データ処理や制御フィールド以外に第3図で示すように1ビットのタイマ制御部を持っている。約100種のマイクロ命令を持

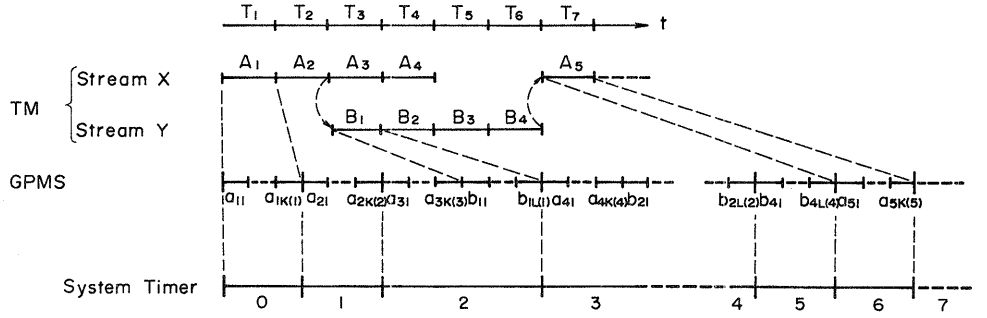
っており、マイクロプログラムメモリが遅いことを利用して効率よいシミュレーションを可能にするように1マイクロ命令で高度な処理ができるように設計されている。主メモリに関係しないマイクロ命令はすべて1.6マイクロ秒で、主メモリに関するマイクロ命令は3.2マイクロ秒で実行される。

4.2　シミュレーション制御モジュール

このモジュールはマイクロプログラムメモリ中のマイクロシーケンスの実行動作を制御するもので、システムタイマ，ポインタレジスタ，ユーザータイマ，シミュレーション制御フリップフロップから成っている。



a.　1つの制御ストリームの時



b.　2つの制御ストリームの時

第4図　タイミング図

システムタイマは8桁のTM系の時計で、マイクロ命令のSフィールドで制御され、コンソール上のINITIALIZEボタンでクリアされる。第4a図は1つの制御ストリームしか同時にアクティブにならないようなTM（例えば小型計算機）をシミュレートする時のシステムタイマの動作を示す。TMの或るデータ処理は基本オペレーションA1, A2, …, Ai, … Anによって達成される。但しAiは1つのマシンサイクルに対応する処理，マイクロ命令または機械語命令で実現されるオペレーションである。TMでのオペレーションAiは通常GPMS上の複数個のマイクロ命令列 $a_{i1}, a_{i2}, \cdots, a_{iN(i)}$ へ翻訳することができる。但しマイクロ命令 $a_{iN(i)}$ のSフィールドは"1"にセットされている。従ってTM上のシーケンスA1, A2, …, AnはGPMSのマイクロ命令列 $a_{11}, \cdots a_{1N(1)}, a_{21}, \cdots, a_{2N(2)}, \cdots, a_{nN(n)}$ を実行することによって、シミュレートすることができる。システムタイムはSフィールドが"1"にセ

ットされているマイクロ命令 $a_{1N(1)}$, $a_{2N(2)}$, … $a_{nN(n)}$ が実行されると1進められ、シミュレーションの最後には第4a図で示されるようにTM上での実行時間を示す。一方第4b図は2つの制御ストリームが同時にアクティブになるTM（例えば命令の先取り部を備えた大型計算機）を実行する時のシステムタイマの動作を示す。第4b図では2つの制御ストリームX，YはクロックT_3及びT_4で同時にアクティブになる。クロックT_1, T_2, T_5, T_6でのシミュレーションは第4a図で示すのと同様に実現される。クロックT_3では制御ストリームXのオペレーションA_3に等価なGPMSのマイクロ命令列a_{31}, a_{32}, …, $a_{3K(3)}$ が先ず実行され、その後制御ストリームYのオペレーションB_1に等価なGPMSのマイクロ命令列b_{11}, b_{12}, … $b_{1L(1)}$を実行した後、システムタイマは1進められる。以下同様にクロックT_4の処理が行われる。

ポインタレジスタは8個までのマイクロシーケンスに関する制御語を持っており、GPMSのシミュレーションプロセスを制御するために用いられる。各制御語は実行状態，ストール条件，マイクロ命令アドレス部から成っている。実行状態はNON-ACTIVE, STALL, NEXT-ACTIVE, ACTIVE の4つの状態を規定する。マイクロ命令アドレス部は実行すべき最初のマイクロ命令が貯蔵されているマイクロプログラムメモリの番地を規定する。第3図に示すように、PR1は第4b図の制御ストリームXの制御状態を示し、PR2は制御ストリームYの制御状態を示す。又ポインタレジスタはBEGIN, STALL 等のシミュレーション制御用マイクロ命令によって変更できる。

ユーザータイマは32ビット長の2進カウンタでそのカウンタからのオーバーフロー条件をポインタレジスタ中のストール条件部で指定できる。各ユーザータイマは各々独立に、

(1) Sフィールドが"1"のマイクロ命令が実行されるごとに、または
(2) Sフィールドが"1"のマイクロ命令がアクティブなシーケンスについてすべて実行されるごとに、

更新される。上記の2つのモードのどちらで更新されるかはユーザータイマ制御レジスタ（第3図のUTC）に従って決定される。ユーザータイマはI/O割り込み，外部割り込み，タイマ割り込み等の各種の割り込み機構を擬似的にシミュレートするために用いることができる。すなわち割り込み処理をシミュレートするためのマイクロシーケンスを作っておき、ユーザータイマに前もってセットされた時間が経過するとそのシーケンスが起動され、必要な割り込み処理を行うようにすることができる。

シミュレーション制御フリップフロップは16ビット長の通常のフリップフロップでポインタレジスタのストール条件部で指定することができ、シーケンス間の制御のためにユーザーが自由に使用できる。

4.3 評価データ収集モジュール

TMが持っているレジスタ，データパス，命令等のハードウェアリソースの使用率を実行中のGPMSのマイクロ命令が持っている情報を用いて等価的に測定する。このモジュールは収集すべきリソースに対応した16ビット長の16個のキーパターンを貯蔵するアソシアティブメモリと使用率を貯蔵する16個の32ビット長の2進カウンタであるユーザーカウンタから成っている。処理モジュールでTMのデータ処理のためにマイクロ命令が実行されると同時に、アソシアティブメモリはそのマイクロ命令から作られたマスクパタ

ーンにより照合され、キーパターンと一致するとアソシアティブメモリの各語から一致信号が発生し、その語に対応するユーザーカウンタが1加えられる。キーパターンは16ビット長から成っており、レジスタか命令かを区別するTYPE部とアドレス部またはOPコード部から成っている。

このアソシアティブメモリとユーザーカウンタを用いて評価データを収集することができる以外に、データ収集のために特別なマイクロシーケンスを挿入することによって可能である。すなわち前もってセットされた条件が生じるとそのシーケンスが動作し、シミュレーション制御フリップフロップやポインタレジスタの内容を主メモリや磁気テープへ保存する。

GPMSの操作卓上にはコンソールデバッグを容易にするために、マイクロプログラムやデータの変更，実行プロセスのダイナミックな表示機能等を持っている。

入出力装置との接続のために64ビット長，最大転送率2MBの汎用インターフェースを持ち、ユーザーによって構成されたマイクロプログラムによってどんな入出力デバイスでも接続できるように構成されている。現在PTR，MT，TW，CRT，Mini-Diskが接続されている。

5. 実施例

本システムを用いて、主として本システムの性能を評価するために、汎用中型計算機をシミュレートした場合について以下に示す。

(1) TMの仕様

事務用ギブソンミックスが約20マイクロ秒の汎用中型計算機でマイクロプログラム制御計算機とした。

(2) シミュレーション結果

事務用ギブソンミックスに関する命令をシミュレーションした時の結果を示す。

- マイクロ命令実行ステップ比

  TMの1マイクロ命令は約15倍のGPMSのマイクロ命令でシミュレートすることができる。

- 実行比

  TMの実行時間の約40倍。

6. 結論

以上述べたMPGS／GPMSシステムは計算機の設計及び評価のために有効であることを示した。このシステムを用いることによって設計，評価，修正サイクルの時間は非常に短縮され、GPMSのデータ収集機能を用いて得られた高精度の情報に基いた設計が可能である。

MPGS言語は非常に柔軟なため、大きな自由度を与え、いかなるレベルでも機械を記述することができ、高度に最適化したオブジェクトコードを発生することができる。しかし余りに柔軟なためにユーザーが発生されるべきオペレーションをすべて定義する必要がある。そのため少くとも標準オペレーションのオブジェクトコードはMPGSで発生されるようにライブラリとして内蔵され、ユーザーは定義しなくても自由に使用でき、ユーザー個有のものだけ定義すればよいようにする必要がある。

ＭＰＧＳのファイル管理機能を用いて、１人のユーザーによって定義されたファンクションを他のユーザーが自由に使用できる。

ＭＰＧＳを会話型で使用できるように、又ＧＰＭＳを汎用計算機システムへ接続することを計画している。その時にはより高速にモデルを修正することができ、接続した計算機が持っているソフトウェア／ハードウェアリソースを有効に利用できる。

ハードウェアシミュレータを用いているのでソフトウェアシミュレータに比べて高速のシミュレーションができる。大型機をミクロなレベルでシミュレートすることはＧＰＭＳを用いてもかなりな時間がかかるが、まずマクロなシミュレーションを行い、必要な部分をミクロにシミュレートすることによって解決することができる。入出力オペレーションのシミュレーションは多くの時間に依存する要素を持っているのでGPMS ではそれ程容易でない。しかしながらマクロな入出力動作はＧＰＭＳ のユーザータイマを用いてシミュレートできる。

シミュレーションのために必要な入力（例えばＴＭの評価プログラムやデータ）を選択することは非常に困難な問題である。例えばリソースの使用率はシミュレーション時に用いられるプログラムの種類に大きく依存する。

7. 謝辞

本システムの研究開発にあたり、その機会を与えて下さった木地，村上部長，三上マネジャ，有意義な討論に参加下さった草鹿調査役，村山リーダー，本システムのソフトウェアの開発に援助下さった方々、ハードウェアの開発に協力下さった伏見氏に感謝します。

8. 参考文献

(1) M.S.ZUCKER,"LOCS:An EDP Machine Logic and Control Simulator",IEEE TRANSACTIONS ON ELECTRONIC COMPUTERS,EC-14,NO-6,PP403-416,June,1965.

(2) M.YAMAMOTO,et,al.,"A Microprogrammed Computer Design and Evaluation System",PROCEEDINGS of FIRST USA-JAPAN Computer Conference,1972,PP139-144.

(3) M.HATTORI,et,al.,"MPGS:A High Level Language for Microprogram Generating System",PROCEEDINGS of National Conference ACM,1972,PP572-581.

(4) 山本他，"GPMS:汎用マイクロプログラムシミュレータ" 電子通信学会全国大会，　昭和４７年，　ＰＰ　１２６３.