ハードウェア・モニタの技法について

箱崎　勝也　小野　隆喜
（日本電気株式会社　中央研究所）

1. まえがき

コンピュータ・システムの主として性能を評価するために、稼動中のシステム動作を測定する手法が比較的多く用いられるようになって来た。システム測定には、そのためのプログラムによってシステム動作データの収集を行うソフトウェア・モニタと、測定のための特別な装置、いわゆるハードウェア・モニタとがあるが、それぞれ長所と短所があり一概に優劣を論ずることはできない。一般に、ソフトウェア・モニタではそれ自身が対象システムの上で動作するために、システムの資源、たとえばCPU時間、主記憶スペース，ファイルなどを使用し、測定動作がシステム動作に対して何等かの影響を及ぼすことは避けられない大きな欠点があるが、一方では、システムの動作状態に関する情報を管理プログラムによって用意されたテーブル類から容易に集めることができる利点を持つ。その反対に、ハードウェア・モニタでは測定動作自体は、それが外部の系によって行われるために、対象とするシステム動作に影響を及ぼすことはないが、本質的に受動的にしか情報を収集できず、ソフトウェア・モニタにとっては容易に収集しうる情報でもハードウェア・モニタにとってはそれが極めて困難なことがある。これらのモニタは相補的な性質を持っており、測定の目的に応じて、どちらかを選択するか、あるいはその両方を併用することが望しい。

しかしながら、現実には対象とするシステムに必要とするソフトウェア・モニタ機能が準備されていないなどの理由により、ハードウェア・モニタにソフトウェア・モニタ並の機能が求められることが多い。ソフトウェア，特にシステム・ソフトウェアはハードウェアよりも"固い"という現象はしばしば経験するところである。

このような状況において、筆者等はシステム動作をいろいろな角度から測定するハードウェア・モニタの研究を行い、主としてプログラムの特性に関するデータを収集するデータ収集システムSYDAS[1][2][3]，システム設計における性能改善に使用することを主目的とした性能評価システムSYDAS-Ⅱ[4][5]の試作を行った。本稿は、それらのハードウェア・モニタで使用した技法を中心に、ハードウェア・モニタによるシステム測定法について述べ、これからハードウェア・モニタを作成する、あるいは、ハードウェア・モニタを使用するときの参考に供したい。

2. システム性能と測定

2.1 システムの階層的とらえ方

コンピュータ・システムは多くのシステム・コンポーネントの複雑な組み合わせによって構成されている。システムを性能評価の観点から見るとき、Fig.1に示されるような階層的なモデルでこれをとらえることができる。システムの性能を示す尺度として、最終的に求められるのは、through-putと応答時間の2つであると考えられる。前者はシステム側から見たとき、特に問題であり、後者は、ユーザ側に立ったときに最も重要なもので、この2つにシステムの性能が集約されていると考えることが出来る。

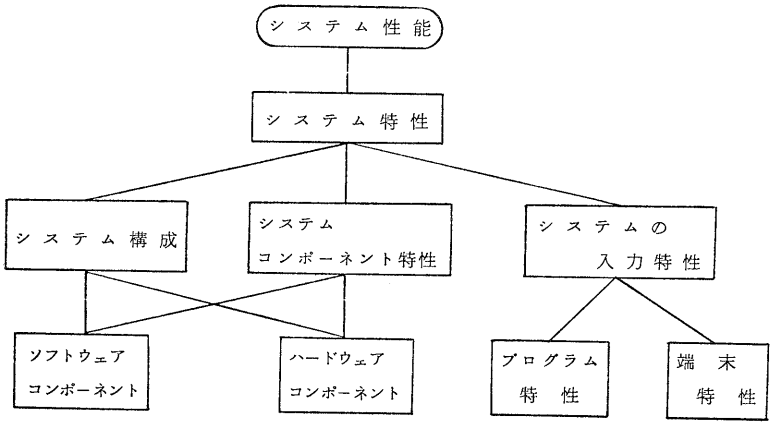

Fig.1　システムの階層的モデル

これらの2つのパラメータで表現されるシステム性能はシステムの特性によって定まる。システムの資源の使用率やシステム各部の queue に関する情報がシステムの特性を定める主なものである。システム特性はそれを構成するシステム・コンポーネントの特性、システムへの入力特性およびシステム構成に依存する。システムを1つの black box として考えるとき、システムの特性はその内部構造を表わすシステム・コンポーネントの持つ特性、たとえば、命令実行速度で示されるCPUの性能、入出力チャネルや入出力装置の性能、システム・プログラムの走行ステップ数、マルチプログラムの数と、CPU，コアメモリ，入出力チャネル，入出力装置の数と同時動作可能性で表わされるシステム構成と、システムへの入力特性で定まると考えることができる。

システムへの入力特性はシステムが提供する各種のサービスに対する要求の特性であり、システムの利用特性である。これはユーザ・プログラムの特性と、TSSやオンライン・システムにおける端末ユーザの特性またはバッチシステムにおける入力特性に相当するユーザ特性によって定められる。

2.2　システム測定の目的

システム測定において、システムの階層のあるレベルに注目し、その特性を定めるパラメータの値を求めることが必要であるが、それらは測定の目的により異る。測定目的は大きく次のように分けることが出来る。

(1)　既存システムの性能評価と改善

既に稼動しているコンピュータ・システムのシステム特性から、システムのボトルネックを見つけたり、現状の負荷では要求される性能を満足していても将来の負荷の増加・変動に於てボトルネックとなる潜在的な要素を検出してこれらの処置を行うような場合がこれに相当する。

(2)　設計段階における性能予測のための基礎データ

システム設計あるいはシステム・コンポーネントの設計において、システム性能の予測が行われることが多いが、システム測定はこのような性能予測における基礎データを得る目的に使用されることがある。

(3) その他

システムの測定の特殊な場合として、プログラムのデバッグや性能予測手法の検定のためのシステム測定がある。

これらの目的のうち、既存システムの評価と性能予測とは密接な関係にある。測定と評価によって判明したボトルネックをなくすために，システムの改善を行うに当って、システムを改善することによる効果の予測が必要であり、測定・評価と改善による性能予測、改善されたシステムの測定と評価がシステム寿命のつきる迄くり返し行われる。従って、既存システムの測定においても、それに続くシステム性能予測手法で使用しやすい形でデータを求められることが望しい。測定の目的はそれを行なう者の置かれている立場に依存し、次のような要因の組み合わされたものである。

・評価者の属性：システム供給者か／システム管理者か／ユーザか
・システムの性質：バッチ・システム／ＴＳＳ／オンライン・システム
・システムの段階：設計段階（基本設計／詳細設計）／稼動段階
・改善の自由度：システム構成／ハードウェア／ソフトウェア／運用形態
・その他

測定システムはこれらによって決まる多様な目的をカバーする機能を備えることが要求される。システムの性能評価はいろいろな立場と背景を持つ人々で構成されたチームによって実行されることが多いため、大きい目的は一つであっても、多くの測定項目が何段かの階層に分かれて出てくるのである。

3. ハードウェア・モニタの基本構成と機能

ハードウェア・モニタは、一般にFig.2に示されるように、プローブ、測定信号抽出論理、データ処理部と記録器によって構成される。プローブは対象システムの中からシステムの動作を反映する状態信号をシステムの正常な動作に影響を及ぼさないように検出し、それをハードウェア・モニタの論理信号レベルに再生・増幅する。この信号をもとに、測定信号抽出論理で測定のための信号が抽出され、データ処理部でカウンタやバッファー・メモリを介して処理された測定データが記録され、解析プログラムにより図表やレポートが出力される。

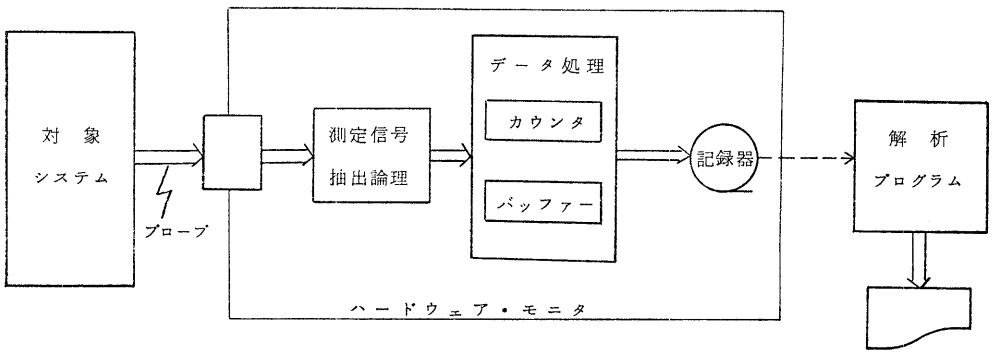

Fig.2 ハードウェア・モニタの基本構成

3.1 測定信号の抽出

プローブによって検出される信号はシステムの動作のある側面を反映しているものであって、システムの完全な動作はシステムに含まれる全要素の状態の遷移列としてとらえることができる。しかし、測定の対象となるシステムの動作はその測定の目的に応じて定められるシステム・モデル上の状態の遷移としてとらえる必要がある。すなわち、システムの動作を測定モデルの動作に写像し、写像されたモデルの動作を測定することにより、測定の目的に応じたレベルでシステムを測定すると考えられる。測定信号抽出の過程は実システムの状態の部分集合から、測定モデル上の状態の集合への写像を行うものであり、ハードウェア・モニタの機能はこの写像がどの程度可能かに大きく依存する。測定信号はその後の処理が行いやすいように、次の3つの形態で求められることが普通である。

一つは、測定信号として、システムが測定の対象となる状態に含まれているか否かに対応する2値をとる測定状態信号としてとりだすもので、たとえば、システムの特性を求めるのに、次のようなシステム状態を持つモデルについて測定するような場合がこれに相当する。

- CPUがbusyである状態：M_1
- いずれかのI/Oチャネルがbusyである状態：M_2
- CPUとI/Oチャネルが共にbusyの状態：$M_3 = M_1 \cap M_2$
- CPUのみがbusyである状態：$M_4 = M_1 \cap \overline{M}_2$
- I/Oチャネルのみbusyの状態：$M_5 = \overline{M}_1 \cap M_2$
- 共にbusyでない状態：$M_6 = \overline{M}_1 \cap \overline{M}_2$

上の例では、対象となるシステムの中にM_1，M_2に対応する状態信号がプローブによって検出しうるとき、簡単な論理回路によって、$M_1 \sim M_6$の状態信号をとりだすことができる。

第2の形態は、測定の対象とする状態に変化をもたらすきっかけ（イベント）の発生を示す信号を測定信号として使う場合である。たとえば、あるプログラムが実行時に使用される回数が測定の対象であるとき、そのプログラムの入口が1つしかないなら、その入口アドレスを通過するものがあったというイベントの発生が検出されれば充分である。

第3の形態としてあるイベントが発生した時点におけるシステムの状態、たとえば特定のレジスタの内容を測定信号として使用するような場合がある。これは第一の形態の測定状態信号へ分解される前の状態信号とも考えられるが、特に、特定のメモリ・アドレスへの書き込みまたは読み取りデータを記録することによって、タスクやジョブに関する動作を測定したり、あるいはqueueの長さを測定するような場合に使用される。

3.2 測定信号抽出の技法

ハードウェア・モニタでは測定信号をとりだすために、いろいろな論理回路が組み込まれているが、汎用ハードウェア・モニタとして市販されているものは、AND，OR，INVERTER，フリップ・フロップ、遅延素子、比較回路などの論理素子をパッチボードを利用して回路を構成し、必要な測定信号をとりだすような構造になっているのが普通である。たとえば、3.1節の例で、M_1，M_2に対応する信号が得られているとき、$M_3 \sim M_6$はFig.3.a)に示すような回路で求められるし、特定のアドレスaに対するアクセスが行われたことを示すイベントEはFig.3.b)の回路によって得られる。このような隔通性は測定者にとって一つのハードウェア・

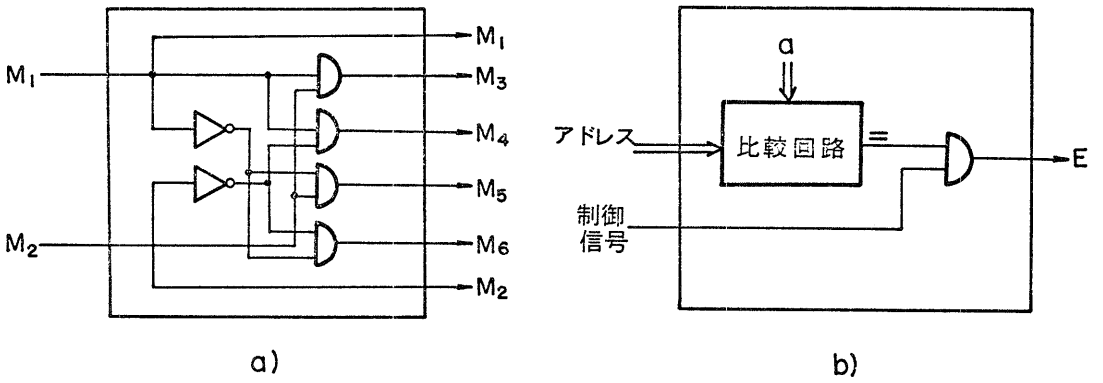

Fig. 3　測定信号抽出論理の例

モニタで多様な測定が出来る利点があるが、一方、測定対象となるマシンのハードウェアについて充分な知識が必要とされる。対象となるシステムが限定され、かつ測定目的がある範囲内に規定しうる場合には、あらかじめ数種の予想される抽出論理を組み込んでおき、使用時にそれを選択するような方式をとることも出来る。このような方式で汎用性を充分に残すための一つの方法は、比較回路を多数用意しておき、比較回路に与えるパラメータを変えることによって、多くの目的に合致する測定信号を得られるようにすることである。たとえば、Fig. 3.b) に示される回路において、入力信号をアドレスの代りに命令コードに切り換え、それに対応する制御信号をとれば、aというパターンと一致する命令の実行というイベントが検出される。

SYDAS , SYDAS-Ⅱ では比較回路を多数用意する代りに、連想メモリが使用されている。連想メモリの各ビットは比較機能と記憶機能を持っているので、比較の対象とするパターンを連想メモリに書き込んでおき、測定時に入力パターンとの比較を各語毎に行わせて、一致したパターンが存在したか否かでイベントを検出しうる。これを用いることにより、メモリに常駐しているプログラムの動きを各プログラムの入口または出口を通過したというイベントの列として追跡することが、連想メモリにそれらのアドレスを書き込むという操作だけで可能であり、固定されたハードウェア・モニタの構成でも、多くのイベントを検出することができる。SYDAS には 128 語 24bit , SYDAS-Ⅱ では16語 18bit の連想メモリがあるが、システム・プログラムの追跡によって、システムの特性を求めるには NEAC2200シリーズの MOD Ⅳ のOSで約32～64語程度の連想メモリを要した。

最近の集積回路技術の発展により、コンピュータの集積度が向上しており、直接プローブによって検出しうる信号が少くなって来ているし、また、マイクロプログラム方式の採用によって、システムの状態がマイクロプログラム・シーケエンスの中に埋もれてしまう傾向にあるが、連想メモリによって間接的に測定信号をとりだすアプローチはハードウェア・モニタでは重要になる。

3. 3　測定信号の処理

測定信号は最終的に測定の目的に沿った形の情報として得られるように処理される必要があるが、その過程はハードウェア・モニタ内部で行われる測定時の処理と、ハードウェア・モニタにより処理され記録されたデータをもとにモニタの

外部で行われる後処理の2つに分けることが出来る。測定時の処理が多ければ多い程、記録されるデータ量が少くなり、後処理に要する手間と時間は少くてすむが、測定データから得られる情報は測定時の処理に依存してしまい、多様な情報を引き出すことが困難になる。一方、測定時の処理が少ければ、データ量が増大し、後処理が大変であるが、逆に、一回の測定によって得られるデータから後処理を変えることにより多様な情報を得られる利点がある。ハードウェア・モニタの設計において、これらの処理のトレード・オフを測定目的との関連において充分考慮する必要がある。

ハードウェア・モニタの中で行われる処理の代表的なものは、2進または10進カウンタを用いて、測定状態信号が"1"(または"0")である時間を累算するか、あるいはイベントの発生回数を求め、その計数結果を適当な時間々隔ごとに磁気テープ、紙テープあるいはプリンタなどに記録するものである。この処理では出力されるデータ量はイベントの発生頻度には関係なく、出力の時間々隔とカウンタの数とビット数によってのみ決まる。これは長期間にわたって、システムの特性を統計的に測定するような場合に適しており、多くのハードウェア・モニタはカウンタによる時間または回数の計数を基本にしている。

このようにして求められるものは、時間または回数の時間々隔ごとの総和であり、たとえばある状態信号の接続時間の分布に関する情報は失われてしまう。勿論、ある状態への遷移回数とその状態の時間とから平均値を各時間々隔毎に求めることが出来、時間々隔毎の平均値の分布は求められるが、正確な分布を求めたことにはならない。そのために、分布を求めるカウンタ群を別に用意し、第一のカウンタで求めた状態時間を、その状態が終ったときにその状態時間に対応するカウンタで発生回数を求める処理を行うものもある。

もう一つの代表的な処理はトレースと呼ばれるもので、イベントの発生時にイベントの識別子、発生時刻、その時の状態信号などを時系列データとして逐次記録する。これは後処理に処理の大部分を委ねる測定法の典型的なものであるが、イベントの発生間隔が比較的長い場合や、短時間に多くのデータを得る必要がある場合などに使用される。SYDAS-Ⅱのように、ミニコンピュータが測定システムに組み込まれている場合には、処理の大部分をミニコンピュータにまかせ、モニタ側はミニコンピュータでは処理が間に合わないような頻度の高いイベントに対して、カウンタで処理することが考えられる。特に最近かなりの性能を持ったマイクロプロセッサが安価に入手しうるようになって来ているので、マイクロプロセッサの利用も充分考えられる。

ミニコンピュータやマイクロプロセッサを組み込むことのもう一つの利点は、測定の制御をプログラムすることが出来ることである。SYDAS-Ⅱでもミニコンピュータ(NEAC-M4)が測定のプログラム制御、測定時の処理のために用いられているが、ハードウェア・モニタの操作性の向上させる上で有益であることが確認されている。

4. システム・モデルとハードウェア・モニタによる測定

システム測定を行うにあたって、まず第一に測定の目的が明確にされることが重要であるが、その次には、その目的に沿ったシステムのモデル化[6]を行い、そのモデルを表現するパラメータを測定によって求めるアプローチが必要である。モデルは測定目的に沿ったもので、かつ、実際のシステムとの対応が容易につけられるものであることが望しい。モデルを考えずにシステムを測定しても、いたず

らにデータの山を築くだけでそこから何の情報も得られないことになりかねない。そのようなモデルは現在のところあまり確立されているとはいえないが、いくつかの例とそれに対するハードウェア・モニタによる測定法について述べる。

4.1 システム・プロフィール

システム特性を表わすモデルとして、システムを構成する主要なコンポーネントの使用率を、各コンポーネントの同時動作を考慮に入れて表現するシステム・プロフィールがシステム構成のバランスを調べるために用いられる。Fig.4 にその一例を示す。

システム・プロフィールの測定は対象となるシステム・コンポーネントの使用状態を示す信号から論理回路により、それらのオーバーラップ状態を求め、各状態の時間をカウンタにより求めることによって容易に行うことが出来、ハードウェア・モニタにとって最も得意な測定の一つである。このようなシステム・プロフィールからボトルネックの可能性が高いシステム資源を見つけることができる。[10]

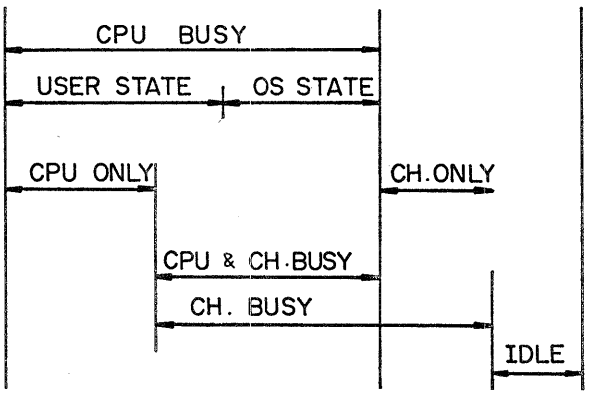

Fig.4 システム・プロフィール

システム・プロフィールの一つの表現法として、Kiviat 等の提案した Kiviat Graph[7] がある。これは8つの軸にシステム特性の良性因子と悪性因子を交互に配置し、その各軸上における使用率をプロットしたもので、理想的なバランスで動作しているシステムは Fig.5 に示すような星型になる。このように表現することにより、視覚的にシステムのアンバランスを見つけやすくしている。

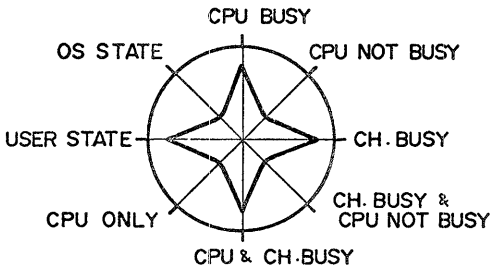

Fig.5 Kiviat Graph

4.2 SVC モデル[8]

システムへの入力特性として、システムの提供するサービスに対するユーザ・プログラム(またはシステム・プログラム)からの要求(スーパーバイザ・コール(SVC))の列と相隣接する SVC 間の時間々隔の列としてとらえるモデルで表現することがある。システム性能予測を行うための比較的詳細なレベルのシミュレーションあるいは解析的手法においても、SVC 列でプログラムの特性を表現するモデルは有効である。

SVC のようにプログラムに関連するイベントを検出することは、ハードウェア・モニタにとってかなり苦手なものであり、ソフトウェア・モニタの方が得意な測定項目の一つである。SYDAS では連想メモリに OS の各 SVC に対応する処理ルーチンの入口アドレスと、ユーザ・プログラムを識別するために、ジョブ、タスクの制御ブロックのアドレスとを設定しておき、それらのアドレスに対するアクセ

スが行われたというイベントを時系列として求め、それを解析することにより、SVC 列を求めている。このようにして得られたデータから、ユーザ・プログラム毎に SVC 発生の時間々隔の分布、SVC の種類別の発生頻度および総 SVC 発生個数の形に集約してプログラムの特性を求められる。

SVC 列はシステムから見たプログラムの動作を反映しているわけで、この測定データを編集することにより、Fig.6 に示すようなタイム・チャートにシステム動作を再現することが出来る。更に、これを集約することから、Fig.4 に示されたシステム・プロフィールを求められる。このような測定法はデータの量が多くなり、後処理も面倒ではあるが、一つの測定データから多くの情報をとりだすことができ、システム設計のための基礎データとして有効である。

4.3 Working Set モデル[9]

SVC モデルではシステムの主として時間的要素に注目しているが、記憶空間上の動作を表わすモデルとして、Denning の Working Set モデルがある。Working Set はプログラムのメモリに対する要求特性であり、ある時刻 t から、それ以前の t 時間の間にプログラムが実際に使用したメモリ・ブロック（ページ）の集合として定義される。

SYDAS の Working Set の測定では、連想メモリを用いることにより、プログラムがアクセスしたメモリ・ブロックを検出し、連想メモリに結合されたカウンタで各メモリ・ブロックに対するアクセス回数を求めている。このカウンタを一定時間毎に記録したデータを解析することによって、Working Set に関する情報が求められる。

4.4 その他の測定モデル

システム測定のためのモデルとしては、まだ確立されているものの方が少いが、以下に示すような測定が行われることが多い。これらをある意味ではモデル化が行われていると考えられる。

(1) 命令使用頻度：CPU の設計において、実際のアプリケーションにおける命令使用頻度は重要である。特にマイクロプログラム方式では設計の自由度がかなりあり、そのアーキテクチャーにおけるコンパイラの特性も含んだ命令使用頻度が必要とされる。ハードウェア・モニタでは比較回路または連想メモリとカウンタによって測定される。

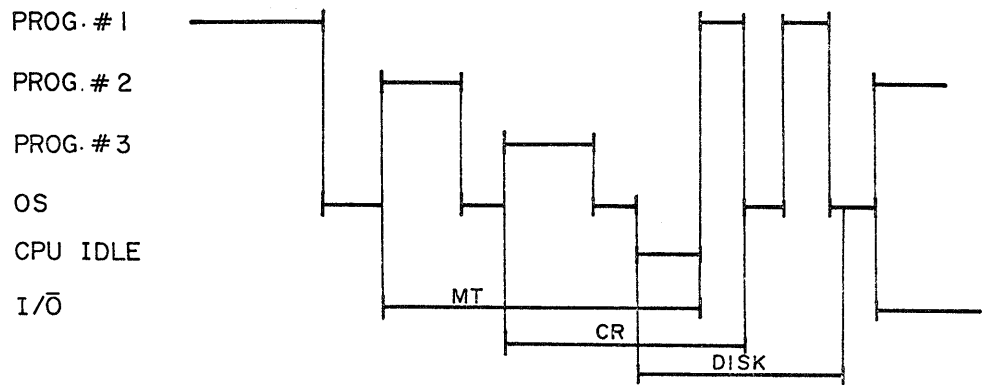

Fig. 6　システムの動作チャート

(2) プログラムの使用頻度と処理時間：命令使用頻度と同様に、システム・ソフトウェアにおいて、プログラム・モジュールの使用頻度とその処理時間の測定が重要である。ハードウェア・モニタよりもプログラムに手を加えて測定する方が簡単かも知れないが、比較回路または連想メモリとカウンタの組合せで測定される。

(3) Queue：特にオンライン・システムにおいて、システム各部の queue に関する情報は大切であるが、一般にハードウェア・モニタによる測定は困難である。SYDAS-Ⅱ では queue の待ち個数をその管理を行うプログラムがメモリに書き込むデータをとらえることにより測定している。

(4) ファイルへのアクセス特性：コンピュータ・システムの中でファイルは極めて重要な要素の一つであるが、ファイルに関する評価モデルはまだあまりないようである。ファイルの論理的構造はハードウェア・モニタで検知することは極めて困難であり、ハードウェアの動作、たとえばシーク時間、サーチ時間、シリンダ、トラックへのアクセス系列、ファイル装置の使用率などが測定されることが多い。ファイルに関する測定・評価方法は今後の課題の一つであろう。

5. むすび

筆者等の試作したハードウェア・モニタを中心にそこで使用した測定技法、およびハードウェア・モニタの測定法について述べた。ハードウェアの価格は集積回路技術の発展により安くなっており、ハードウェア・モニタが比較的安価に作りうるようになっており、今後ハードウェア・モニタが使用されることが多くなると予想される。ハードウェア・モニタはシステム測定のための有効な方法の一つであるが、必しも万能ではなく、ソフトウェア・モニタやその他の手法と共に用いられるべきものであると考える。これからのコンピュータ・システムにおいて、システムの性能を評価・改善するために測定機能はあらかじめ組み込まれるべきである。仮にそうでないにしても、せめて測定を考慮に入れた設計がなされる必要がある。

謝辞

日頃御指導頂いている三上課長；SYDAS，SYDAS-Ⅱの試作に種々の便宜をはかって頂いた小高部長，木地部長，鍵山部長および大野氏，伏見氏はじめ関係各位に心から感謝いたします。

参照文献

(1) 小野，箱崎，"システム・データ収集装置－SYDAS"
昭和46年，電子通信学会全国大会予稿，№1053

(2) 箱崎，小野，"ハードウェアモニタ（SYDAS）によるシステム性能評価" 昭和46年，情報処理学会第12回大会予稿，№186

(3) 箱崎，小野，"ハードウェアモニタによるシステム測定"
情報処理，Vol.13，№11，1972，PP782-788

(4) 小野，箱崎，宇津木，"性能評価システム－SYDAS-Ⅱ"
昭和47年，情報処理学会第13回大会予稿，№70

(5) 箱崎，小野，宇津木 "コンピュータシステムの測定・解析システム：SYDAS-Ⅱ"，昭和49年，電子通信学会電子計算機研究会資料
資料番号 EC73-61

(6) A.Sekino, "Performance Evaluation of Multiprogrammed Time-Shared Computer Systems", 1972, MIT, Project MAC TR-103

(7) K.W.Kolence, P.J.Kiviat, "Software Unit Profiles & Kiviat Figures", ACM, Performance Evaluation Review, Vol.2, №3, 1973, PP2-12

(8) 三上，久保，高橋他，"コンピュータシステム性能評価シミュレータ PACSS"，情報処理，Vol.12, №1, 1971, PP14-25

(9) P.J.Denning, "The Working Set Model for Program Behavior", Comm.ACM, Vol.11, №5, 1968, PP323-333

(10) J.S.Cockrum, "Interpreting the Result of a Hardware System Monitor", Proc. SJCC, 1971, PP23-38