論　文

制御用マルチコンピュータシステムにおける共有メモリの設計と解析*

坂東忠秋** 川本幸雄** 前島英雄**
小林芳樹** 井手寿之***

Abstract

Recently, multi-minicomputer architecture is being widely adopted in control computer applications because of its higher performance/cost, expandability, responsiveness, and availability. In typical architecture, each processor has its own memory and shares a common memory. As the common memory stores the system data and the global plant data, it plays a key role.

There are three major design considerations. First of all, the large common area must be easy for a 16-bit-minicomputer to access. Secondly, the common memory must be highly reliable. If it has a failure, the entire system operation will shut down. Finally memory conflicts must be kept low so that the performance decrease will be within the limit.

In this paper, the actual design for the first two considerations is described. For the third problem, the new method to calculate the waiting time in the system which has both the common memory and the the processor's own memory is discussed. These designs are realized in HIDIC 80 (Hitachi Digital Control Computer).

1. まえがき

最近の計算機制御システムは，大規模化・高度化の一途をたどっており，このために，制御用計算機は，多様なシステム機能，高いコスト・パフォーマンス，高い拡張性・応答性・信頼性が要求されるようになってきている．このような多様な要求に柔軟に対処するためには，小規模システムから大規模システムまで基本的なモジュールをつみ上げることによって，統一的にシステムを構成できるモジュラーアーキテクチャが望ましい．マルチミニコンピュータシステムは，このような要請に答えうるものと考えられる．

ところで，計算機システムは，いかなるマルチコンピュータシステムを要求しているのであろうか．**Fig. 1**は，マルチコンピュータシステムを用いた計算機制御システムの構成法を示している．まず，プラント業

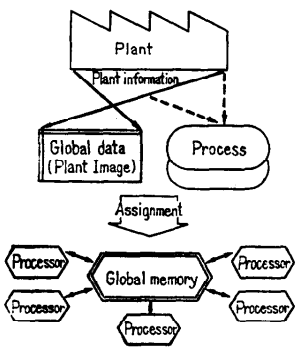

Fig. 1 System design of a computer control system by multi-computer.

* Design and Analysis of the Common Memory in a Multi-computer System for Control Applications by Tadaaki, BANDOH, Yukio KAWAMOTO, Hideo MAEJIMA, Yoshiki KOBAYASHI (Hitachi Research Laboratory, Hitachi, Ltd.), and Jushi IDE (Omika Works, Hitachi, Ltd.).

** (株)日立製作所日立研究所
*** (株)日立製作所大みか工場

務を，プラントの状態を写像するプラント情報と処理を行う複数のプロセスに分解し，次にプラント情報を共有メモリ（グローバルメモリと呼ぶ）に，また幾つかのプロセスをまとまった単位で，コンポーネントコンピュータに割り付ける．これによりいわゆる分散処理システムといわれるマルチコンピュータシステムに再構成でき，システムの拡張が容易になり，大規模システムとなっても高い応答性を維持できる．またプロセッサ間は高速高信頼度のグローバルメモリによって結合されているために，万一あるプロセッサが故障しても他のプロセッサによって制御を継続できる．

このような制御用マルチコンピュータシステムにおいて，システム構成の基幹となるのが，グローバルメモリである．本論文では，このグローバルメモリの具体的設計手法について，特に，制御用として重要な問題点であるアクセス方法と，高信頼化手法について論じ，またメモリ競合によって発生する「待」を解析する新しい手法を呈示している．メモリの競合解析に関しては，これまでに発表されている論文[2)~4)]と異なり，本論文では，グローバルメモリのほかにプロセッサごとに個有メモリを持つ場合を扱っており，また，非同期アクセスを考慮した解析方法を提起し，シミュレーションによって検証している．

2. グローバルメモリの課題と設計手法

2.1 グローバルメモリの使用法と望まれる要件

制御用マルチコンピュータシステムの具体的な構成を **Fig. 2** に示す．各プロセッサ PE は，個有のメモリ PM を持ち，グローバルメモリ GM が，PE 間を結合する．入出力装置 I/O は，スイッチ SW を介して，PE に接続される．PE 間は，GM を介してデータの送受を行うほか，相互に割り込みがかけられるようになっている．

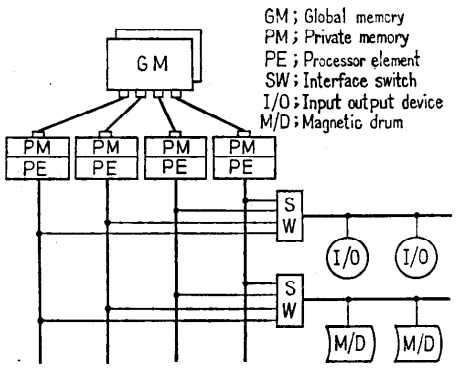

Fig. 2 A multi-minicomputer structure.

GM にはプラント情報のほか，システムのリソースの状態，すなわち，使用中か否か，あるいは，故障中か否かなどを記憶するシステムテーブル類，PE 間の交信エリアなどが設けられる．PM には，ワークエリアや，プラントデータのうちでも，PE が故障した場合に，リカバリが多少遅れても支障のないデータが記憶される．GM では必ずしもプログラムが実行できなくともよい．それはプログラムは不変なために，リカバリが簡単に行えること，及び，プログラムを GM において実行させると，メモリのアクセス頻度が高くなって，競合による性能低下が大きくなるためである．

このような使い方がなされているので，GM は，1 語単位に容易にアクセスできることが望ましく，重要なデータを格納するために高信頼性を必要とし，また多くの PE からアクセスするために競合が起るが，この時の性能低下が許容範囲内であることが望ましい．本章では，このようなグローバルメモリの課題につき，アクセス方法と高信頼度化について具体的設計手法を述べ，メモリ競合については次章で解析方法と結果について述べる．

2.2 GM のアクセス方法

GM には，システムテーブルなどが記憶されるために，1 語単位にアクセスでき，また Compare and Swap 命令によって，セマフォーを実現できなければならない．語長が 16 ビットのミニコンピュータで，大きな空間をとり扱うには，効率の良いアドレス方式が必要である．

このための代表的な手法としては，C.mmp[5)] で採用されているマッピング方式があげられる．これは，マップを書換えることによって，大きな物理空間を参照できるようにするものである．この方式の制御用への応用を考えたときの問題点は，論理空間はやはり 64k 語のために，GM のサイズが大きくなった場合には，マップの切換を多用しなければならず，効率低下

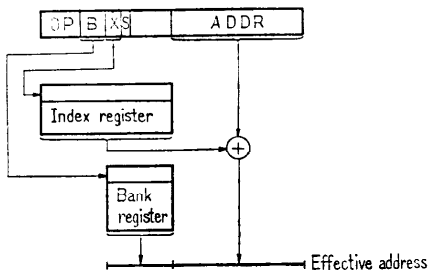

Fig. 3 A method of GM addressing.

を招くことになる点である．このような問題点を解決するものとして，筆者らが具体化した方式について次に説明する．**Fig. 3** は，2重修飾機能を有するミニコンピュータにおける，GM のアクセス方法を示したものである．S ビットが 0 の時には，普通の 2 重修飾であり，(B)+(X)+ADDR が実効アドレスとなるわけである．S ビットが 1 のときには，B パートは Bank register を示し，その出力と (X)+ADDR が並置されて，実効アドレスとなるものである．GM は，Bank register が 0 以外の場合であり，0 の場合は PM を示す．

この方法によれば，最大 448 k 語の GM 空間が直接参照できることになる．大規模システムでは，大きなグローバルエリアを必要とするものがあり，このように広い空間を直接参照できると効果的である．なお，Compare and Swap 命令にも，このようなアクセス方法を与えることによって，セマフォーが実現でき，システムテーブル類を排他的にとり扱うことが可能である．

2.3 GM の信頼性設計

GM には，重要な情報が記憶されているために，GM が故障した場合，PE 間の連絡が不可能となるだけでなく，プラントの状態を示す重要な情報を失ってしまい，リカバリに必要な時間が長くなってしまう．いわば，GM はシステムの中枢ともなる所で，高信頼度設計がなされていなければならない．

高信頼性を実現する一手法として，GM を 2 重化する方法がある．2 重化を行い，単一故障ではシステムダウンとならないようにすると，2 重系システムの MTTF (Mean Time to Failure) は，λ を故障率，μ を修復率としたときに，

$$3/2\lambda+\mu/2\lambda \qquad (2.1)$$

で表わされ[1]，1 桁以上の信頼性改善度が期待できる．ここで，2 重系システムの MTTF を向上させるために重要な点は，μ を大きくすること，すなわち保守性を向上させることと，故障が波及してシステムダウンにならないように，安全性への配慮を行うことである．特に保守を行うときには，システムを停止しなくても良いように，切り離し，個別保守，再起動が容易に行えるように配慮されていなければならない．

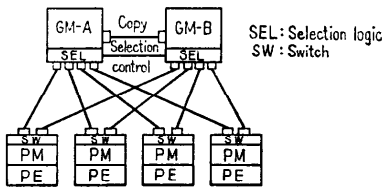

Fig. 4 Dual structure of GM.

Fig. 4 は，このようにして設計された 2 重化 GM の構成を示したものである．GM は，A と B の 2 つの系より成り，SEL は PE からの要求を選択する所，SW は，両系から読出されたデータのうち，パリティエラーのない方を選択する所である．GM-A，GM-B は，各々，別のユニットに実装され，電源も別々に用意されている．

ここで，Selection Control は，GM-A と GM-B の SEL が，同一順序で，PE に対するサービスをするように制御するためのものである．これは，選択の順序が異なると，両系のデータが異なってくるためである．例えば，GM-A では，PE 1, PE 2 の順序で選択し，GM-B では，逆に，PE 2, PE 1 の順序で，同一エリアに書込を行った場合，GM-A では，PE 2 のデータが，GM-B では，PE 1 のデータが最終的に残ることになってしまう．

また，Copy は，単独運転状態からデュアル運転になるときに使われるもので，一方の GM から，他方の GM にデータを転送して，内容の同一化を図るものである．この場合に，PE からの要求を無視して全ワードを一度に複写すれば簡単であるが，サービス停止時間が長くなるために，1 語ずつ PE の要求がないときに複写されるように設計することが望ましい．

以上，GM の設計手法について述べたが，最も問題となる競合による処理性低下に関して，次章以下に解析法とシミュレーション結果を説明する．

3. GM の競合の解析

3.1 GM の性能に関する要件

GM は，PE とは別のユニットに実装されるために，メモリのアクセス時間に加えて，ケーブルによる遅れ，選択回路による遅れなどが加算されて，競合による待のない場合でも，200 ns 以上アクセス時間が長くなってしまう．さらに待による遅れが加わるが，平均的待時間を短縮する手段としては，インターリーブなどの手法が利用される．ただし，インターリーブは，ピン数の制約なども大きいために，インターリーブ数を大きくすることは容易ではない．

一方，GM は，PE 間の共通データだけが記憶されプログラム，ワークエリアなどはすべて PM であるとした場合には，GM に対するアクセス頻度はそれほ

ど高くはない．命令フェッチとオペランドフェッチの割合が 1.5 : 1 程度であり，オペランドフェッチのうち GM にアクセスする割合を考慮すれば，全アクセスのうち，GM のアクセスは高々 30% 程度であると推測される．

従って，このような使用条件下で，性能低下が許容範囲内で収まるように，インターリーブ数を決定すれば良い．

3.2 GM の待行列解析モデリング

PM と GM の2種のメモリを持つマルチコンピュータシステムの GM 待行列モデルを図示すると **Fig. 5** のようになる．Server は，GM でインターリーブを行わない場合には，単一窓口，単位分布のサービスを行い，また，客の数，すなわち，PE の数は，有限の場合の問題となる．ここで PE の GM 要求の分布は，すべての命令実行時間が同一で，GM の要求確率が同一と仮定すれば，PE の Think time (GM のサービスを受け終ってから，次の要求を出すまでの時間) は幾何分布に従う．しかし，実際には命令の実行時間は一定ではないために，指数分布に近いものになると仮定できる．従って，このモデルは，有限個客，Think time が指数分布，単位時間サービスの待行列となる．

なお，メモリ参照には Read と Write があり，おいてきぼり制御をしなければどちらも同じと考えて良く，おいてきぼり制御をする場合には，待時間は両者の要求を合わせて計算し，処理速度への影響を考える場合には，これを Read の割合で乗ずれば良い．

これまでにメモリの競合をとり扱った論文は，多く見うけられる．2)は，プロセッサの台数の少ないときの解析として，メモリ参照要求のタイミングは，全プロセッサが同期して出すことを仮定し，競合が起ったときに，各プロセッサがメモリを占有できる確率を与えて計算する方法を提案している．これは，プロセッサとチャネルなど性質の異なったものが要求を出す場合の解析法として考えられたもので，プロセッサ数が少ないときには有効だが，多くなると解析が困難になる．3)は，やはり同期アクセスを仮定し，プロセッサの性質は同じ場合に，離散マルコフチェーンを使った解法を示している．この方法は，同期アクセスの仮定が許容できれば正確であるが，プロセッサ数が多い場合には計算が大変である．4)は，3)のケースにおいてプロセッサ数が多い場合の近似解法を与えている．本論文では，PM と GM の両者が存在する場合の解析を扱っているので，この点でこれらの論文とは異なる．解析モデルは，3)のモデルにおいてプロセッサの処理時間を幾何分布とした場合に近いが，メモリ参照の要求は同期している必要はなく計算も簡単になる．

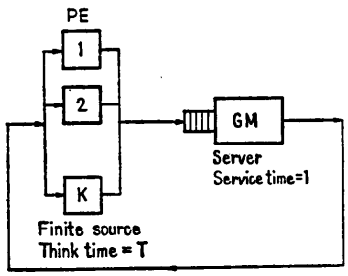

Fig. 5 A queuing system with finite input source.

ところで，各プロセッサのメモリ要求が非同期でも良いとした場合に解析的に直接解けるのは，メモリがただひとつ，すなわち，インターリーブしていない場合であり，この場合については隠れマルコフチェーンを使って待時間が求められる．次に，この解析について述べ，インターリーブをしている場合については等価 Think time という考え方を導入して解けることを明らかにする．

3.3 インターリーブがないときの待行列解法

この問題は，メモリのサービス時間が単位分布であるために連続的なマルコフ過程とはならないが，メモリのサービスが完了した時点 $t_1, t_2, \cdots t_k$ をとり，このときの系の長さを $X_1, X_2, \cdots X_k$ とすると，$\{X_i\}$ はマルコフ連鎖をなしており，これを利用して系の定常確率を求めることが可能である．いわゆる隠れマルコフチェーンである．定常状態において，PE が j 台待状態にある確率を π_j とし，状態確率ベクトルを $\boldsymbol{\Pi}[\pi_0, \pi_1, \cdots, \pi_{n-1}]$ と定義する．ここで n は，PE の台数である．次に遷移確率行列 $\boldsymbol{P}=[P_{ij}]$ は，$P_{ij}=P\{X_{k+1}=j|X_k=i\}$ で定義され，現在状態 i のときに，次のステップで状態 j に移行する確率を示している．PE の Think time を T とし，GM のサービス時間を 1 とすると，GM の1サービス時間中にある PE が GM 参照要求を出す確率 α は，

$$\alpha=1-e^{-1/T}$$

で表わされ，P_{ij} は，

$$P_{ij}=\binom{m-i}{j-i+1}\cdot\alpha^{j-i+1}\cdot(1-\alpha)^{(m-i)-(j-i+1)} \quad (i\neq 0)$$

$$P_{0j}=\binom{m-1}{j}\cdot\alpha^{j}\cdot(1-\alpha)^{m-j} \quad (i=0)$$

で計算される．ただし

$$\binom{k}{j}={}_kC_l=\frac{k!}{l!(k-l)!}$$

である．定常状態では，平衡方程式

$$\boldsymbol{\Pi}=\boldsymbol{\Pi}\cdot\boldsymbol{P} \qquad (3.1)$$

が成立し，これと

$$\sum_{i=0}^{n-1}\pi_i=1$$

を解けば，状態確率 π_i が求められる．

次に，PE の待時間の平均値を求める．上記の計算で求まった値は，GM のサービスが完了した時点における系の状態確率である．これは同時に，PE からの要求が到着したときの系の状態確率でもある[5]．いま，要求が到着したときに，系にサービス中のものも含めて，k 個 $(k\geq 1)$ の要求があったとする．このときの待時間は，

$$W=k-1+Y \qquad (0\leq Y<1)$$

で表わされる．ただし Y は，現在サービス中のものの残りのサービス時間である．Y を求めるには，次のように考えれば良い．いま注目している要求は，現時点のサービスが開始される時点では到着しておらず．完了した時点では到着していたわけであるから．Xをいま注目している要求が発生した時点で，現在のサービス開始時点から測った時間，x を $0\leq x<1$ の変数とすると，X が x より小さい確率は，

$$P\{X<x\,|\,x<1\}=\frac{1-e^{-x/T}}{1-e^{-1/T}}$$

で表わされる．従って X の平均値 EX は，

$$\mathrm{EX}=\int_0^1 x\frac{dP\{X<x\,|\,X<1\}}{dx}\cdot dx$$

$$=-\frac{e^{-1/T}}{1-e^{-1/T}}+T$$

と計算される．従って，待時間の平均 EW は

$$\mathrm{EW}=\sum_{i=1}^{n-1}(i-1+\mathrm{EY})\cdot\pi_i$$

$$=\sum_{i=1}^{n-1}(i-\mathrm{EX})\cdot\pi_i \qquad (3.2)$$

から求められる．このようにして求めた結果を **Fig. 6** に示す．GM へのアクセスは，全アクセスの内，高高 30% 程度であり，従って Think time が 3.3 以下であるといえる．Fig. 6 には，この範囲の結果が図示されている．

3.4 インターリーブがあるときの待行列解法

ここで述べるインターリーブは，アドレス下位ビットのインターリーブであり，PE が要求を出す確率は

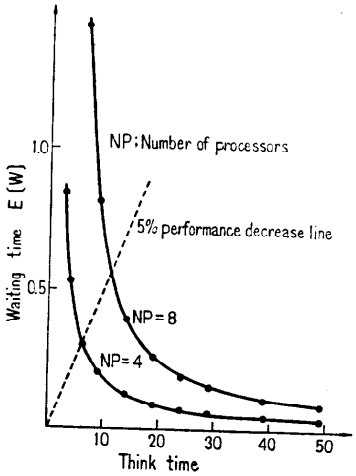

Fig. 6 Waiting time of non-interleaved memory.

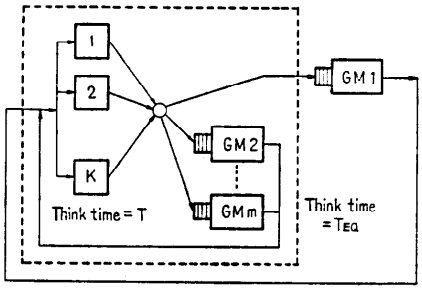

Fig. 7 A model of interleaved memory system.

全ての GM に対して同一と仮定する．GM が複数個非同期で動作するようになると，前節のように GM のサービス完了の時点がマルコフ連鎖とはならなくなる．従って，直接的に解くことはできないが，近似的に次のような仮定をおいて解くことが可能である．

この方法は，複数の GM のうち，**Fig. 7** のようにひとつだけに注目して，等価的な Think time を求めて，解く方法である．ここでは GM 1 のみに注目しており，PE が GM 2～GMm を使用中の場合には，Think time が続いていると見なすものである．GM 1 を選択するまでの GM 要求回数は幾何分布に従うために，ある PE が GM 1 のサービスを受けてから次に GM 1 に要求を出すまでの等価的 Think time はやはり，指数分布に近いと考えられ，この平均値を Teq とおく．この Teq は，本来の Think time T と，待時間の平均値 EW から，

$$\mathrm{Teq}=(m-1)\cdot(\mathrm{T}+\mathrm{EW}+1)+\mathrm{T}$$

で求められる．ここで EW は，Teq が与えられれば，平衡方程式(3.1)，平均を求める式(3.2)から求め

ることができるが，Teq は EW が与えられなければ計算できない．これを求めるには次のような繰返し演算すなわち，

Teq $i=(m-1)\cdot(T+W_i+1)+T$ ただし $W_0=0$

によって，i 回目の等価 Think time Teq i を求め，これより，(3.1), (3.2)を使って W_{i+1} を計算すれば良い．

ここで，W_i の収束性が問題であるが，W_i が収束するならば，その値は EW となることがいえる．この理由は，もし $W_i>EW$ ならば，Teq $i>$Teq となり，W は T の単調減少関数であるから，$W_{i+1}<EW$ となるために，常に $W_i<EW<W_{i+1}$ または $W_i>EW>W_{i+1}$ の関係が成立するからである．有効数字3桁を求める場合，NP=8, NM=4, T=1.5 で，6回で収束しており，T は通常の動作状態では，これより大きくなるが，この範囲ではもっと早く収束している．

このようにして，計算した結果を **Fig. 8** に示す．これは，PE 数と Think time をパラメータとしてインターリーブの効果を示したものである．

3.5 シミュレーション結果との比較

前節では，等価 Think time という考え方を導入し，待時間を算出したが，この結果の妥当性を評価するために，シミュレーションを行い，結果を比較した．このシミュレーションは，Think time とアクセスする GM の番号を乱数を発生して与えるものである．**Table 1** に結果を数値計算と比較して示す．シミュレーションは，GM アクセスを 1,000 回実行させたものを，20 回繰返して，平均値と分散から信頼度区間を求めてある．結果を見て分る通り，数値計算とシミュレーションは，良く一致しており，等価的 Think time の考え方の妥当性が分る．また，数値計算の場

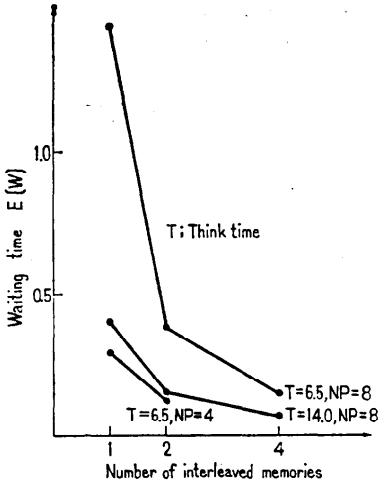

Fig. 8 Effect of interleaved memories.

Table 1 Waiting time comparison between analysis and simulation results.

NP=8, NM=1			NP=8, NM=2		
Think time	Analysis WT	Simulation WT 95% Confidence interval	Think time	Analysis WT	Simulation WT 95% Confidence interval
4.0	3.05	(3.05 , 3.15)	4.0	0.708	(0.717 , 0.779)
9.0	0.807	(0.774 , 0.834)	6.5	0.379	(0.364 , 0.394)
14.0	0.395	(0.383 , 0.413)	9.0	0.251	(0.249 , 0.267)
19.0	0.256	(0.243 , 0.265)	11.5	0.186	(0.178 , 0.194)
24.0	0.188	(0.182 , 0.194)	14.0	0.148	(0.145 , 0.155)
29.0	0.148	(0.143 , 0.159)	19.0	0.104	(0.098 , 0.106)
39.0	0.105	(0.099 , 0.109)	24.0	0.0804	(0.0755, 0.0859)
49.0	0.0806	(0.0756, 0.0856)	29.0	0.0654	(0.0619, 0.0671)
59.0	0.0655	(0.0614, 0.0720)	34.0	0.0550	(0.0511, 0.0575)
NP=8, NM=4			NP=4, NM=2		
Think time	Analysis WT	Simulation WT 95% Confidence interval	Think time	Analysis WT	Simulation WT 95% Confidence interval
1.5	0.594	(0.605 , 0.643)	1.5	0.464	(0.484 , 0.510)
2.75	0.354	(0.349 , 0.375)	2.75	0.278	(0.275 , 0.293)
4.0	0.243	(0.232 , 0.248)	4.0	0.194	(0.186 , 0.198)
5.25	0.183	(0.177 , 0.189)	5.25	0.148	(0.146 , 0.156)
6.5	0.146	(0.138 , 0.152)	6.5	0.119	(0.114 , 0.124)
9.0	0.104	(0.0969, 0.103)	9.0	0.0857	(0.0826, 0.0912)
11.5	0.0801	(0.0749, 0.0819)	11.5	0.0667	(0.0619, 0.0697)
14.0	0.0653	(0.0624, 0.0718)	14.0	0.0546	(0.0502, 0.0572)
16.5	0.0550	(0.0539, 0.0599)	16.5	0.0463	(0.0450, 0.0512)

NP: Number of processors, NM: Number of memories, WT: Waiting time

合には，シミュレーションでは求め難い待時間の分布をも求めることも可能である．

3.6 GM 待時間が性能に与える影響

以上，隠れマルコフチェーンと等価 Think time を使うことによって，GM を参照する際の待時間を求める方法を示した．GM 待時間は，命令実行時間を増加させることになり，その割合は，W/T で表わされる．仮に，待による性能低下を5% 以下におさえようとした場合，Fig. 6 で待時間は，破線より下になければならないが，破線よりも上にある場合には，インターリーブなどの手段を用いて，等価 Think time が許容範囲内に入るようにしなければならない．これは待による性能低下分であるが，さらに GM のアクセス時間が，PM よりも長いことを考慮に入れておかなければならない．このようにして，GM の性能を評価しながら，設計を進められるわけであるが，もうひとつ注意すべき問題点としては，I/O のオーバーランがあげられる．

ディスクなどの回転体メモリは，データが一定の時間内にサービスされないとデータを紛失してしまう，いわゆるオーバーラン現象が起る．従って，GM の競合によって生ずる最大の待時間よりも I/O のバッファサイズを大きくしておかなければならない．これまでの解析では，GM のアクセスは1回だけのサイクルの場合について扱ってきたが，実は Compare and Swap のような，2回のサイクルを要する場合もあり，これも含めて，最大待時間を考慮しなければならない．

また，最大待時間を有限にするために大切なことは，選択制御であり，一定時間内には必ずサービスされるような方法が望ましい．優先サービス方式は，優先度の低いものが長く待たされる可能性があり，オーバーランの要因となる．待時間の分布は FIFO の場合には簡単に求められるが，ハードウェアで FIFO を実現するのは難しく，循還サービスなどの代替法をとることになる．循還サービスとは，選択回路が一定の順序で，要求の有無を調べてゆくやり方であり，この方法でも大差のない待時間分布が得られることが明らかになっている[6]．

4. むすび

制御用マルチコンピュータにおける，グローバルメモリの設計上の諸問題について報告した．グローバルメモリは，マルチコンピュータシステムの中枢ともなる所で，システムテーブル，プラント情報など最も重要なデータが記憶される．グローバルメモリの設計上問題となるのは，語長の短い，ミニコンピュータで，グローバルメモリをアクセスする方法，高信頼度構成法，及び，競合による性能低下を許容できる範囲におさえることである．アクセス法に関しては，Bank Register によってアドレスビット長を増やす方法を示し，高信頼化に関しては，2重化の方法と実現上の諸問題について説明した．競合の問題については，インターリーブのない場合には，隠れマルコフチェーンを使った解法を示し，さらに，インターリーブがある場合には，等価 Think time という考え方を導入した解析手法を示した．さらに，待によって生ずる問題点として，平均命令実行時間の増大とともに，I/O のオーバーラン現象について論述した．

なお，本論文で述べた設計・解析法は，制御用マルチコンピュータシステム，HIDIC 80[7] の開発に際し適用されたものである．

参考文献

1) 猪瀬 博：コンピュータシステムの高信頼化，pp. 44～47，情報処理学会 (1977).
2) C. E. Skinner, J. R. Asher: Effects of storage contention on system performance, IBM Syst. J., Vol. 8, No. 4, pp. 319～333 (1969).
3) Dileep P. Bhandarkar: Analysis of Memory Interference in Multiprocessors, IEEE Trans. on Comput. Vol. C-24, No. 9, pp. 897～907 (1975).
4) Forest Baskett, Alan Jay Smith: Interference in Multiprocessor Computer Systems with Interleaved Memory, Communications of ACM, Vol. 19, No. 6, pp. 327～334 (1976).
5) Leonard Kleinrock: Queueing Systems, pp. 176, Wiley-Interscience, New York (1975).
6) Tadaaki Bandoh, Yukio Kawamoto: Design Considerations in Multi-minicomputer performance, Proceedings of the 1976 International Conference on Parallel Processing, pp. 219 (1976).
7) 桑原 洋：HIDIC 80 処理装置，日立評論，Vol. 58, No. 6, pp. 69～74 (1976).
8) William A. Wulf, C. G. Bell: C. mmp-A Multi-Mini-Processor, Proc. of FJCC, Vol. 41, pp. 765～777 (1972).

(昭和 52 年 8 月 29 日受付)
(昭和 53 年 1 月 19 日再受付)