Vol. 22 No. 3 情報処理学会論文誌 May 1981

# 仮想独立バスインタフェース方式による計算機間結合†

阿草清滋†† 大野 豊††

複数のミニコンピュータを結合する簡便な方式を提案し，その適用例について述べている．ここで提案された手法は中継結合形計算機間結合の一種であり，中継装置とミニコンピュータはバス結合される．このバス結合インタフェース回路は各プロセス間通信路ごとで仮想的に独立している．すなわち，固有の物理装置番号，割り込みベクトル発生回路を持つように制御される．

この仮想独立バスインタフェース方式により，ミニコンピュータから見た論理的なプロセス間通信路は1つの入出力装置に対応させることができ，ミニコンピュータ側に要求されるソフトウェアは非常に簡単なものとなる．多くの計算機間結合においては，複数のプロセス間通信路間で共有される1つの物理通信路を管理するためのソフトウェアはかなり複雑であり，また各計算機ごとに異なる．本方式では中継装置にマイクロプロセッサを用い，ここで多重化を行い，各計算機の制御方式の差異を吸収している．

本方式を MUNPS (Micro processor Used Network/Peripheral System) として実現した．これは4台の異機種ミニコンピュータから成るシステムであり，低速入出力装置の制御をも中継装置に制御させることで，ミニコンピュータの高スループット化も図っている．データ転送能力はあまり高くないが，信頼性，保守性，簡便さなどの本方式の特徴が，実際の運用によって検証された．

## 1. はじめに

最近の LSI 技術の急速な進歩により，計算機システムのコストに占める CPU コストの比率は年々下がってきている．これに対して，機械的な部品を含む入出力装置や，ソフトウェア開発などのコストは増大の傾向にある．このため，複数の計算機を結合して周辺機器やソフトウェアなどのリソース共有を図ることが多い[1),2)]．

しかし，計算機間結合のためのハードウェア，ソフトウェアはそれほど簡単ではなく，リソース共有システムの開発にかかる人手，コストの大きさがその利用の障害となることも多い．とくに計算機がミニコンの場合にはその本体価格が安価であるために，必要とされる機能ごとにミニコンを割り当てることも可能であり，計算機間結合に要求されるハードウェア，ソフトウェアのコストは，その機能をほかのミニコンに移植するコストより小さくなければならない．

我々は計算機間結合装置の中核にマイクロプロセッサを用い，簡便で使い勝手のよい計算機システム構成を設計，開発した．これは中継結合形[3)]の計算機複合体の一種でありながら，チャネル間結合のように相手の計算機を入出力機器としてみなすことができ，各計算機上に必要なソフトウェアは簡単な入出力ハンドラプログラムだけで済む．また，複数のプロセス間通信路を単一のハードウェア上に設定するためには，何らかのソフトウェアにより時間軸上に通信路を多重化しなければならないが，この機能をマイクロプロセッサに負わせることで，ホスト計算機側から見た各プロセス間通信路には各々独立したインタフェース用ハードウェアが用意されていると見なされる．これにより，アプリケーション・レイヤのプロトコルは入出力装置制御のためのプロトコルと一致させることが容易にできる．さらに，計算機の代りに入出力装置を接続することで入出力制御機能の一部をマイクロプロセッサに行わせ，ホスト計算機の入出力制御プロセッサとして働きホスト計算機の負荷軽減も図っている．

我々はこのプロセス間通信路ごとに割り当てられた仮想的なインタフェース回路を仮想独立バスインタフェースと呼び，この方式を用いた計算機結合を MUNPS (Microprocessor Used Network/Peripheral control System)[4)] として実現し，その有効性を確かめた．

## 2. 仮想独立バスインタフェースによる計算機間結合

異機種計算機間の結合には，ある新たな標準インタ

† A Simple Computer Network via Virtually Independent Bus Interfaces by KIYOSHI AGUSA and YUTAKA OHNO (Department of Information Science, Kyoto University).

†† 京都大学工学部情報工学教室

フェースを設定し，各計算機のインタフェースをこれに適合させることが必要となる．バス結合方式やリング形結合方式ではそこに用いられるバスの仕様が，また，中継結合方式では中継装置が標準インタフェースと見なせる．バス結合方式ではバス競合管理が複雑であるが，結合の自由度は大きく，密な結合が可能である．中継結合方式では中継装置として計算機を用いることにより，インタフェースが容易となり拡張性，柔軟性に富んだシステムを構成できる．

我々は中継結合方式に仮想独立バスインタフェースの考えを付加し，簡単なインタフェース回路とソフトウェアで計算機間結合を行うこととした．また，中継結合装置としてマイクロプロセッサを用いることで，その周辺 LSI が利用でき，小型で簡単なハードウェアで実現できる．

### 2.1 仮想独立バスインタフェースの概念

複数のデバイスが1つの信号線を共有するために一般にバス構造が採られる．ほかの計算機上のプロセスとの通信をラインプリンタやディスク装置等の一般の入出力処理と同様に扱うためには，その通信インタフェースを入出力バスに接続し，必要とされる入出力装置ハンドラをオペレーティング・システムに組み込むことで実現される．多くの場合，ある計算機上には複数のプロセスが存在し，これらが互いに独立にほかの計算機上のプロセスと交信を行うため，複数のプロセス間通信路が必要とされる．各プロセス間通信路ごとに独立な物理インタフェースを割り当てることはハードウェアコストの点から望ましくない．このため，通信インタフェースもまた，入出力バスと同様に時分割して複数のプロセス間通信路に共有される．この実現方式は種々あるが，必要とされるソフトウェアは複雑でオペレーティングシステムの変更が要求されることも多い．

我々は中継装置の中枢であるマイクロプロセッサにプロセス間通信路の多重化を行わせる仮想独立バスインタフェース方式を考案した．この方式は，ある計算機上のプロセスから見たプロセス間通信路ごとにその通信路専用の独立したバスインタフェース回路が存在しているかのように見えるもので，単にユーザプログラムレベルだけでなく，オペレーティングシステムに組み込まれる入出力ハンドラルーチンにとっても独立したインタフェースとして扱える点に特徴がある．すなわち，各仮想独立バスインタフェースは固有の物理装置番号が識別のため割り付けられ，実際のハードウェアがあるかのようにオペレーティングシステムにより管理される．**図1**にこの概念図を示す．

一般に計算機の入出力はユーザプロセスと入出力装置の間に，オペレーティングシステムにより管理されるデバイスハンドラを介し行うことで，各種の入出力装置を論理ファイルと見なし，OPEN, CLOSE, READ, WRITE などのマクロで統一的な扱いを可能としている．デバイスハンドラはユーザプロセスからのマクロの発行，あるいは，デバイスからの割込みにより起動され，どちらも割込み禁止状態で実行されるため，同一デバイスに複数のデバイスハンドラを割り付けても，その中の1つしか一時には起動されない．このため，インタフェース回路のレジスタなどの排他アクセスもデバイスハンドラを適当に作成することで容易に実現できる．

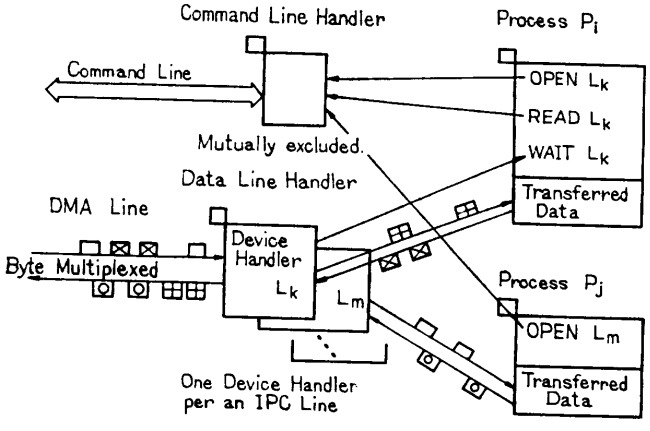


**図 1** 仮想独立バスインタフェースの概念図

Fig. 1 Inter-process communication via virtually independent bus Interface.

### 2.2 中継装置の役割

一般に計算機結合における中継装置の役割りは，プロセス間通信路の設定，解放，同期，エラー処理である．仮想独立バスインタフェース方式では，これらの処理に加えて各計算機の入出力動作の特殊性を吸収するための処理ルーチンが必要となる．このルーチンにより，中継装置に結合される計算機の側ではその計算機の標準的なインタフェース回路，デバイスハンドラを用意するだけで簡単に接続できる．

計算機間通信のオーバヘッドを減らすため，DMA (Direct Memory Access) 方式でデータ転送を行うが，中継装置ではこのためのメモリアドレス，転送語数などの管理を行う．仮想独

立バスインタフェース方式ではそれらの値を保持するレジスタも仮想化し時分割して利用するため，それらの値を中継装置のメモリ内に格約し管理する．さらに，データ幅の変換（バイトまたはワード），特定データの無効化，指定データによるデータ転送の中断などを制御する．

中継装置に結合されるのは計算機のみならず，紙テープ読取機やプリンタなどの各種入出力装置も含まれる．これにより，中継装置はある計算機の入出力制御用計算機とも見なせ，機能の分散化がなされる．ミニコンへの入出力装置の接続インタフェースに比して，マイクロプロセッサへの接続は豊富な LSI ファミリの制御用チップを用いることで容易に安価に実現される．これら入出力装置の制御をも中継装置に組み入れることは有用である．

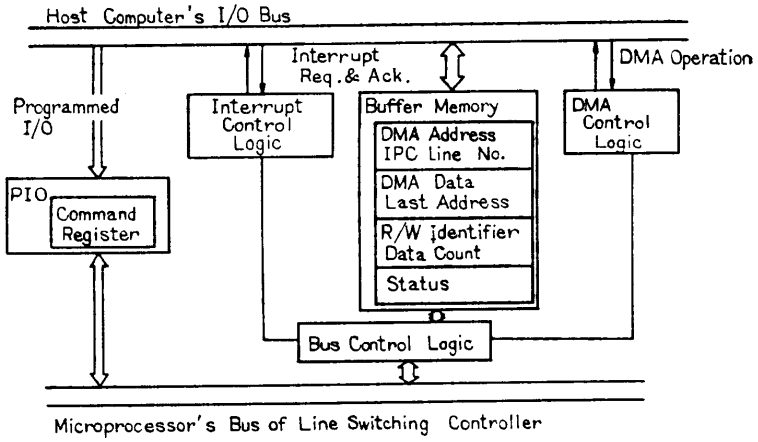


**図 2**　仮想独立バスインタフェースの回路構成

Fig. 2　Typical interface circuit for virtually independent bus interface schema.

### 2.3　中継装置とのインタフェース

#### 2.3.1　物理的インタフェース

計算機と中継装置の間の物理インタフェースは各計算機の入出力バスの仕様により異なるが，通常はコマンドライン制御部とデータ転送制御部からなる．このインタフェース回路の物理装置番号はコマンドライン制御部に1個と $n$ 本の計算機通信路に $n$ 個の合計 $n+1$ の装置番号が割り付けられる．コマンドライン制御部に割り付けられた装置番号は通信路の設定，解放，状態のチェック等の場合に用いられ，計算機側からその装置番号でアキュミュレータ経由で入出力される．計算機間通信路に割り当てられた装置番号は，データ転送の終了，異常などを中継装置が知らせるために用いられ，装置番号をベクトル割込みに用いる計算機では割込み確認応答時にその仮想インタフェース番号を入出力バスに乗せる．また，割り込み要因をフラッグセンスで調べる計算機では対応する装置番号のフラッグセンス時に適当なデータを乗せる．

16 ビットミニコンピュータの計算機と中継装置の間には 16 ビット4ワードの2ポート・バイポーラメモリが置かれ，両者の処理の同期がとられる．すなわち，データ転送時には DMA 制御情報の送り出し，データの受け取りバッファとして，転送終了時には転送終了通信路名（装置番号），データ転送最終アドレス，転送データ数，通信路状態などのレジスタとして用いられる．コマンドラインは中継装置のマイクロプロセッサのプログラマブル入出力制御用 LSI の1つのポートに直接接続することで，回路の簡略化を図っている．**図2**に標準的なインタフェース回路を示す．

#### 2.3.2　論理的インタフェース

プロセス間通信を行う場合には，通信のための何らかの規約，プロトコルを定めなければならない．この規約には各種のレベルがあるが，ユーザプロセスと中継装置ハンドラの間は一般の入出力装置に対する処理手順と同じものとするためここではその説明を省略する．中継装置ハンドラと仮想独立バスインタフェース制御モジュールとの間の規約には，計算機間通信路の設定，データ転送，通信路の解放などに関するものが必要となる．

計算機間通信路の設定，解放，データ転送の開始などを指示できる計算機をマスタと呼び，ほかをスレーブと呼ぶ．受動的にしか動作できない入出力機器や，シングルジョブしか実行できないオペレーティングシステムの下に利用されている計算機で複数の通信路が不用な計算機などはスレーブとなる．スレーブとなる機器の通信路，データ転送などはシステム起動時にオペレータにより指定されているものとし，通常はデータ転送の同期待ち状態とされている．しかし，必要に応じスレーブの通信路の状態をマスタ，またはオペレータにより変更できるようにし，スレーブ間通信も可能とする．

#### 2.3.3　仮想独立バスインタフェース制御コマンド

中継装置を仮想独立バスインタフェースとして機能させるための指令として，**表1**に示す8つのコマンド

表 1 プロセス間通信のためのコマンド一覧
Table 1 Control commands of virtually independent bus contoller.

| Command | Code | Meanings |
|---|---|---|
| BOOT | 00 | Activate MUNPS |
| RELEASE | 01 | Kill MUNPS |
| OPEN | 02 | Open Inter-process Communication Line |
| CLOSE | 03 | Close Inter-process Communication Line |
| READ | 04 | Start Read Data |
| WRITE | 05 | Start Write Data |
| STATUS | 06 | Read Status of Inter-process Communication Line |
| SPECIAL | 80-FF | Special Command for I/O Device Control |

を用意する．これらはマスタである計算機，あるいは中継装置に接続されたキャラクタディスプレイから与えることができる．仮想独立バスインタフェース方式による計算機結合は，一般のコンピュータネットワークにおけるバーチャルコールのように通信に先立ってその通信路を確定する．通信データのブロックの大きさが，バイト単位，あるいはワード単位で転送され，物理インタフェース上では複数の通信路のデータがバイト（ワード）マルチプレックスされる．これにより低速で発生する通信要求に対しても，バッファの管理などのオーバヘッドなく効率よくデータ転送を可能とする．この場合，1つの通信路をコマンドの転送に割り付けるにせよ，この通信路のデータ転送の起動のためには，中継装置に対する割り込みなどで知らせる必要がある．このため，コマンドの転送路は一般の計算機間通信路と区別されたコマンドラインと称するものを設ける．このラインを介して送られるコマンドが仮想独立バスインタフェース制御コマンドである．

（a） BOOT コマンド

中継装置へのプログラムの転送，初期化などを行うコマンドであり，特定の計算機（通常は計算機網の中枢的な計算機）からのみ発せられる．必要な入出力装置の初期化，論理通信路番号としての物理装置番号の割当てなどが行われる．

（b） RELEASE コマンド

中継装置の機能を停止させるコマンドであり，入出力装置の動作終了，すべての通信路は閉鎖される．このコマンドも特定の計算機によってのみ発せられる．

（c） OPEN コマンド

計算機間通信路の設定を指示するコマンドである．このコマンドに対するパラメータリストを図3に示す．このパラメータリストは仮想インタフェース制御テーブル VICT (Virtual Interface Control Table)

| Upper Half Byte | Lower Half Byte |
|---|---|
| Command Code (see Tab. 1) | IPC Line Number |
| Editing Command | End Code |
| Neglected Code-1 | Neglected Code-2 |
| Data Address High | Data Address Low |
| Word Count High | Word Count Low |

注：IPC Line Number はデバイスコードに対応させられる

図 3 プロセス間通信路制御のためのパラメータ・リスト
Fig. 3 Parameter list of inter-process communication line control.

として中継装置内に置かれ，通信路制御に置かれる．各論理通信路ごとに2つの VICT が組として用意される．OPEN パラメータリストには相手計算機を示すパラメータは無く，論理通信路を示す物理装置番号のみでリンクが確立される．このため，ユーザは計算機網内のすべてのプロセスの状況を考慮しなければならないが，同じプログラムで必要に応じて異なる計算機上のプロセスと通信でき，中継装置の制御プログラムの簡略化も図れる．

データ編集パラメータはデータ幅（バイト／ワード），上下バイトの順序，パリティチェック有無，終了コード／無視コードの処理の有無など，中継装置に対する転送データの編集を指示するものである．

ステータスはその通信路の状態を示しており，接続待ち，データ転送中，転送終了，転送エラーなどを示す．

データバッファは転送データの幅が送信側と受信側で異なる場合に用いられる．中継装置の制御プログラムの簡略化のため，データのバッファリングは行わないことを基本とするが，仮想的なDMAインタフェースのレジスタは複数の通信路に共有されるため，転送データ幅が送受信で異なるときのみバッファリングを行う．

データ・エリア・アドレス，ワード・カウントは通常の DMA インタフェース回路のアドレスレジスタ，ワードカウンタに当るもので，カウントが0になれば転送が終了する．送受信側のバッファの大きさの違いにより，先にどちらかのワードカウントが0になっても他方には知らされず，単にデータ転送終了待ちの状態とする．

終了コードはデータ転送要求量に満たない場合にもこのコードが転送されることで転送を終了させるものであり，ソースファイルの行単位の転送，エンドオブファイルコードの処理などに有効である．

無視コードはデータ転送中に無視されるコードであり，入出力装置とのデータ転送の際に主として用いる．

(d) CLOSE コマンド

通信路の閉鎖を指示するコマンドであり，中継装置では対応する VICT の抹消を行い，データの転送途中であれば相手方にそのステータスを返す．

(e) READ コマンド

すでに設定済みの通信路を介してデータの入力を要求するコマンドである．同じ物理装置番号の通信路に READ 要求が出ていると，異常として両端の計算機に知らせる．

(f) WRITE コマンド

READ コマンドに相対するコマンドであり，設定された通信路を経てデータの出力を要求するコマンドである．同一通信路に WRITE コマンドがすでに発せられている場合には，機器異常のステータスがセットされる．

(g) STATUS コマンド

通信路の状態を知るためのコマンドで，通信路がアクティブか同期待ちであるか，データ転送数，データ・アドレスなどが知らされる．

(h) SPECIAL コマンド

接続されている計算機，入出力装置によって，特別なインタフェースが作られることがある．たとえば，スレーブの起動とか，入出力装置の初期化などが必要とされる場合に，それを制御するためのコマンドである．

## 3. 仮想独立バスインタフェースの適用

我々は仮想独立バスインタフェース方式を採用した4台のミニコンピュータからなるインハウス・ネットワークを設計し製作した．その中継装置を MUNPS と呼ぶ．

この背景は年々増大する研究室の計算機への処理要求に応ずるため，演算能力，ファイル管理機能の優れた新しいミニコンピュータの設置と，大型計算機センターの超大型計算機の TSS サービス，RES (Remote Entry System) 機能の使用が必要となったことである．

従来からある2台のミニコンピュータ (HITAC 10/II 64k バイト，IMLAC PDS-1D 32k バイト) と新たに導入される計算機を有機的に結合し，これまでのソフトウェア資源，ハードウェア資源を有効利用する必要があった．また，大型計算機センターの遠隔局のためのソフトウェアはかなり複雑であり，その開発は困難に思えたので，そのソフトウェアを組み込んである LSI ベースのミニコンピュータも購入し，ソフトウェア開発の労力を省いた．このため，4台のミニコンピュータからなるインハウス・ネットワークを構成した．これに仮想独立バス方式を適用した．これは以下の理由による．

(1) オペレーティング・システムへの組み込みやすさ

(2) 拡張性——ミニコンの低価格化に伴い，機能分散形システムの要求も高まると考えられるが，その場合に既存システムのハードウェア／ソフトウェアの変更は必要でなく拡張性が大きい．

(3) オーバ・ヘッドが少ない．——中継装置が低速入出力装置の制御を行う．またプロセス間通信のソフトウェアも簡単である．

(4) 適応性が高い．——中継装置にマイクロ・プロセッサを用いており，環境の変化に対応できる．また，特定のコンピュータ間のデータ転送速度を速めたい場合には，そのサービスの優先度を動的に変更できる．

図4に現在の MUNPS を中心とするシステムの構成を示す．

### 3.1 ハードウェア

#### 3.1.1 MUNPS の制御装置

MUNPS の制御素子として，ザイログ社の Z-80 を採用し，1/4k バイトの ROM と 4k バイトの RAM を持つ．これは Z-80 が，(1)高速割り込み処理，(2)高速入出力制御，(3)高速テーブル操作，などの命令セットを持つことによる．

MUNPS は約150個の IC (MSI, LSI を含む) から成り，システム・モニタリングのために7セグメント LED とブザーが付加されている．

#### 3.1.2 計算機-MUNPS 間インタフェース回路

新たに導入した2台の計算機は2.3.1節で述べたインタフェース回路により MUNPS に結合されている．また，既存の2台のミニコンピュータは図4に示されるようにマイクロプロセッサを介して 19.2kbps の非同期通信回線で結合されている．これは物理的距離の問題と IMLAC, HITAC のオペレーティング・システムはシングル・ジョブしか扱えず，複数のプロセス間通信路を必要としないためである．

Z-80 のブロック入出力命令を用い，ホスト・ミニ

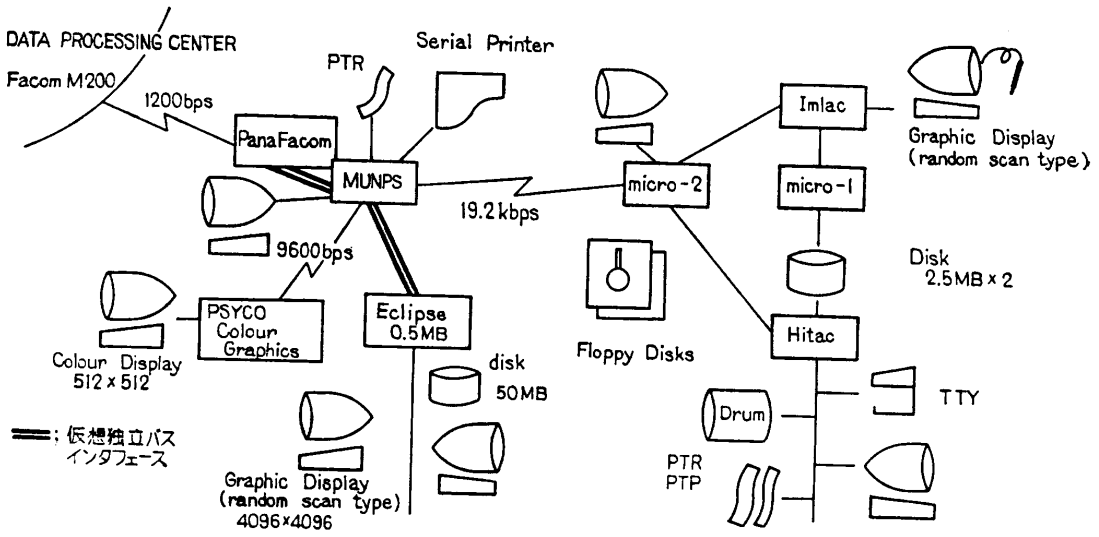


図 4 MUNPS を中心とするシステム構成
Fig. 4 The current system configuration around MUNPS.

コンピュータの DMA 制御に必要な数バイトのパラメータ（メモリ・アドレス，書込みデータなど）が1命令でセットできるようなハードウェアとし，高速データ転送を可能としている．

### 3.2 ソフトウェア

#### 3.2.1 ミニコンピュータのソフトウェア

MUNPS とのインタフェースのためには，2種のデバイスハンドラを用意した．1つはコマンドライン制御用であり，OPEN/CLOSE や READ/WRITE などプロセス間通信路の設定／解除，通信の開始などを司る．ほかの1つは転送終了時の処理用であり，データ転送終了待ちしているプロセスに終了を知らせるものであり，定型的なエラー処理も行う．

仮想独立バスインタフェース方式では各プロセス間通信路は独立しているように扱えるため，OS に組み込むべきソフトウェアは一般の入出力装置制御プログラムと大差なく，コマンドライン制御部で約 600 ステップ，データ転送制御部で約 300 ステップでインプリメントされている．データ転送制御部はリエントラントな構造を持ち，複数の仮想独立バスインタフェースからの割込みの制御要求を受け付けることができる．この点は一般の入出力装置制御プログラムと異なる．

プロセスの生成・消滅に関するコマンドは省略されている．これはオペレーティング・システムの変更を必要とする点，および，研究室内のネットワークでは仕事の内容が定型化していて事前に必要なすべてのプロセスを起動しておける点，物理的距離も離れておらずすべての操作卓に容易にアクセスできる点などの理由による．

エラー対策についてはアプリケーション・レベルにその処置を任せることとし，エラー発生時には通信路閉鎖と同様の処置を行う．エラーはハードウェア・レベルでのパリティ検査，ソフトウェアでのタイマ管理で検出される．研究室内のネットワークではすべての計算機の動作状況のユーザによる把握が可能であり，エラー発生時の処置をユーザの指示に任せることもでき，必ずしも複雑なエラー処理プログラムがアプリケーション・ソフトウェアに要求されない．

#### 3.2.2 MUNPS のソフトウェア

計算機間結合の中心となる MUNPS は3つの動作モードを有する．

（a） ノーマル・モード　MUNPS がプロセス間通信路制御を行うモードである．

（b） IPL モード　MUNPS に接続されている1つのホスト計算機から MUNPS 制御プログラムをロードする．システム・デバッグ，変更を容易にするため，ROM 領域に制御プログラムを置くことを避けている．

（c） PTR モード　MUNPS は単にホスト計算機の紙テープ読取機インタフェースとして働く．これはホスト計算機のディスク内容が壊れるようなパニックに備えたもので，紙テープを用いたシステム・ジェネレーションを可能としている．これは特殊モードであるため，ユーザが MUNPS のパネルのスイッチを

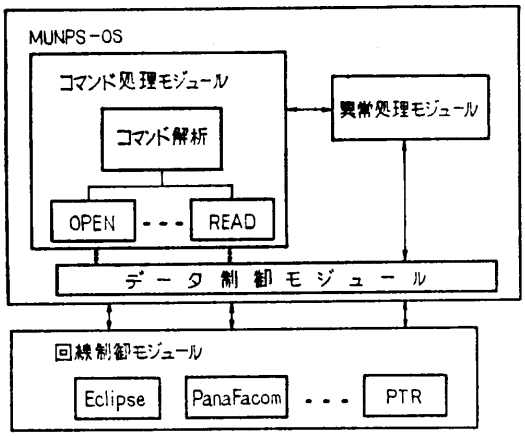


図 5 MUNPS のソフトウェア構成
Fig. 5 The software structure of MUNPS.

押すことでこのモードに入る.

ノーマル・モードにおける MUNPS のソフトウェア構造を**図5**に示す. 仮想独立バスインタフェースの制御の中心となる MUNPS-OS と, 個々の計算機, 装置で異なる制御方式の差異を吸収するための回線制御モジュールから成る. MUNPS-OS はコマンド解析・処理モジュール, データ制御モジュール, 異常処理モジュールから成る.

データ転送は基本的には1語単位の転送とし, ブロック化を避け, MUNPS でデータ・ブロックの管理を行うことはない. また, エラー検出もバイト単位のパリティ検査で済ませ, 巡回符号などによるバースト誤り検出は考慮していない. さらに再転送などのエラー回復も行わない. これによりソフトウェアは簡単化し, 余り高速でない 8 ビット マイクロプロセッサによる通信路制御を可能としている. ソフトウェアの大きさは約 4,000 バイトであり, すべてアセンブリ言語で記述された.

### 3.2.3 データ転送能力

MUNPS は 8 ビット マイクロプロセッサにより構成されており, 余り高速のデータ転送は望めない. 図4 に示したシステムでの最大要求データ転送量は**図6**に示す場合で 3.5 k バイト/秒である. ただし, ミニコン U-100 は大型計算機センターのリモート・ステーション制御のために動作していて, 大型計算機センターとの通信回線の速度 1,200 bps でデータ転送要求が制限されていると仮定する. MUNPS のデータ転送能力は, その実行プログラム・ステップ数から約 6.4 k バイト/秒(ただし, OPEN, CLOSE など通信路設定, 解除などのコマンドが来ない場合) と見積られており, 現在のところその転送能力に問題はない.

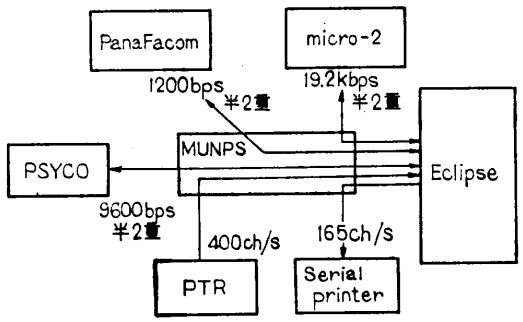


図 6 MUNPS における最大要求データ転送量の見積り
Fig. 6 The case of the maximum rate of data transfer required for MUNPS.

画像データや音声データなどの大量データを実時間で転送する必要が生じた場合には, MUNPS の制御プログラムを入れ替え, ある1つの通信路のみのデータ転送を管理させ, 約 70 k バイト/秒までの転送を可能とさせる.

## 4. おわりに

マイクロプロセッサを用いて, ミニコンピュータの直接メモリアクセス制御回路を制御し, 計算機間の各プロセス間通信路ごとに仮想的に独立なインタフェース回路を提供する計算機間結合方式について述べた. この方式で用いられるマイクロプロセッサは入出力装置も制御し, 一種の入出力制御プロセッサと見なされるが, 各計算機の割り込み機構を巧妙に利用する点でメモリ共有型の入出力プロセッサとは異なる.

本方式では複数のハードウェア論理を中継装置のソフトウェアで実現するため転送速度は余り期待できない. しかしマイクロプロセッサ利用により拡張性, プログラマブルな点などの利点とともに, 各種周辺装置用 LSI を用いてインタフェースも簡単である. また簡単なハードウェアという点で高信頼なシステムを期待できる. データ転送能力は現在のマイクロプロセッサ技術の進歩を考えるとき, かなりの向上が見込まれ広く利用できよう.

本論文でのプロセス間通信路の管理は, 中継装置が 8 ビット マイクロプロセッサで実現されることを仮定して, できるだけ簡略化が図られている. これは処理速度と機能とのバランスで決められる. 研究室内のネットワークではユーザが限定され, しかもシステムを熟知しているので, かなりユーザに負担をかけることで中継装置に必要とされるソフトウェアを軽減する

ことも可能であろう.

## 参考文献

1) Roberts, L. G. and Wessler, B. D.: Computer Network Development to Achieve Resource Sharing, AFIPS, Proc. SJCC, Vol. 36, pp. 543-549 (1970).
2) White, J. E.: A High Level Framework for Networkbased Resource Sharing, Proc. NCC, Vol. 45, pp. 561-569 (1976).
3) 元岡 達: コンピュータ・コンプレックスの展望, 情報処理, Vol. 15, No. 7, pp. 525-533(1974).
4) Kiyoshi Agusa, Tatsunori, Shiomi and Yutaka Ohno: MUNPS; Microcomputer-Used Network/Peripheral Control System, Proc. 12th HICSS, pp. 249-258 (1979).

(昭和55年4月26日受付)
(昭和55年12月18日採録)