電子計算機研究会資料
資料番号EC73-57(1973-12)

研究用電子計算機網TECNET

田中英彦・元岡　達
(東京大学)

1973年12月20日
(於関西)

社団法人 電子通信学会

研究用電子計算機網　TECNET

TOKYO EXPERIMENTAL COMPUTER NETWORK PROJECT

田中英彦　　元岡達
Hidehiko TANAKA　　Tooru MOTOOKA
東京大学工学部
University of Tokyo

1．はしがき

最近，いわゆるComputer Networkと呼ばれるシステムがかなり出現しているが，その目的，システム構成，規模，能力は様々である．我々の所では将来の情報処理システム研究の一環として以前より行なっていた計算機結合研究を電算機網としてまとめあげることにし電算機網プロジェクトTECNET（Tokyo Experimental Computer Network）を開始していたが，この程システム構成が固まって来たので，その概略を御報告する．

TECNETは現在の所，我々研究室の内部的計算機網であるので，その目的は様々なものを含んでいる．すなわち，①計算機網の研究　②マイクロプログラミングの研究　③図形処理及び認識システムの研究　④大型計算機へのリモートジョブエントリ等である．①はいわゆる計算機網一般のシステム構成・利用方式の研究で，数多くの計算機・ファイル等のリソースを有機的に結合して1システムとする為の諸問題研究である．特に，分散形処理システムの基本構造をプロセス間通信としてとらえた場合のプロセス間協調プロトコールを検討すること，複数プロセッサ利用の情報処理能力特性を調べること，計算機網のファイル構成，更に計算機網を意識したオペレーティングシステムの構成，計算機間接続諸方式の比較検討などを目的としている．②は，主要リソースの一つであるWCSを持ったマイクロプログラム計算機の利用方式研究であり，将来エミュレーションを主目的とした計算機が増えてくれば，計算機間のリソース共有を容易にする一道具として期待しているものである．③は研究室の他のプロジェクトとして図形処理があるが，複数計算機を利用して能率良く処理するシステムを開発してゆく上でTECNETを道具とするものである．④は当大学キャンパス内にある大型計算機システムHITAC 8800/8700に対しリモートジョブエントリをおこなうことで，TECNETのどこからでも容易にアクセス出来ることを目的としている．

2．システム構成

2．1　TECNETの網構成

TECNETは図1のような構成となっている．計算機間の接続は，ISO基本伝送制御手順回線接続，拡張モード伝送制御手順回線接続（コードトランスペアレント回線），

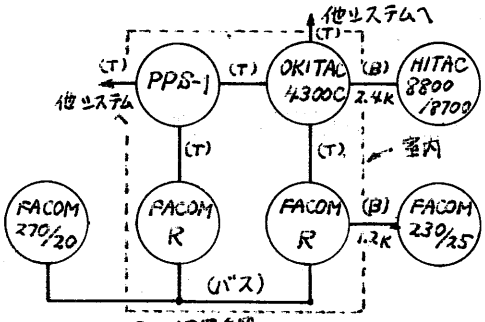


図1 TECNET網

バスによる直接接続の3種を用い，図中ではそれぞれ順に(B)，(T)，(バス)と記されている．また，計算機システムPPS-1は，当研究室と富士通の協同開発機種で現在製作中であり，昭和49年3月頃完成が予定されている．書き替え可能な制御記憶(4kW 25ビット/語)を持つマイクロプログラム制御プロセッサを複数(3台)備えたポリプロセッサ・システムである．Max4台

また，共通バスによる接続システムは以前の計算機研究会[1]で発表したもので，12本の両方向性バス(情報線8本，制御線4本)を張り，それにICC(Inter Communication Controller)を経て計算機が接続される．FACOM-Rにはプログラム・バスに接続されているが，FACOM 270/20にはデータチャネルを通して接続されている．バスの確保・情報転送・バスの解放という手順を取り，非同期でバイト単位で情報が転送される．その速度はバス確保後では20kバイト/秒程度である．金物規模は，IC，MSI等で数えて95ヶ(R用)，132ヶ(270/20用)程度．

全システムの内，破線で囲まれた部分が現在同一室内にあるが，(T)回線は通信回線であるので長距離接続も可能である．

TECNETはARPA網におけるHOST-IMPの厳密な役割分担は無く，OKITAC 4300C，FACOM-R，PPS-1，FACOM 270/20はそれぞれHOSTとIMPの両機能を持ち，HITAC 8800/8700，FACOM 230/25はHOST機能のみを持っている．TECNETの通信制御機能に着目したハードウェア構成は図2の通りである．

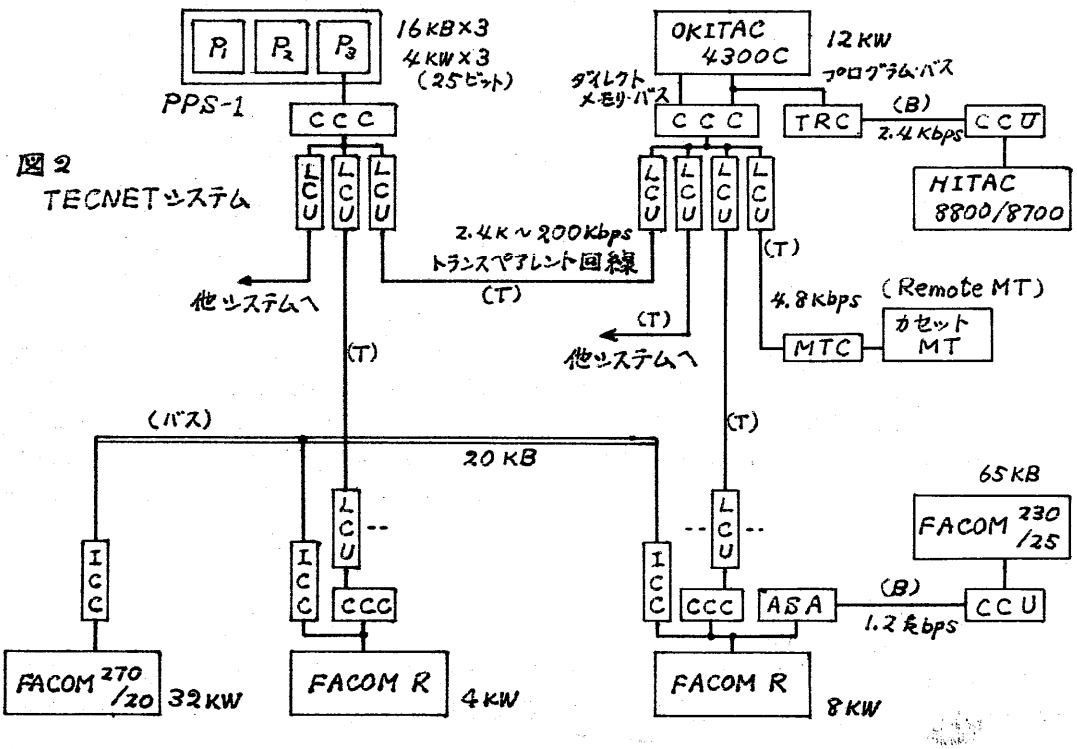


図2 TECNETシステム

2.2 コード・トランスペアレント通信制御装置

計算機相互を結ぶ回線では，主記憶間の直接転送という形を取りうるので，如何なるビットパタンから成る情報であってもそのまま相手迄正しく届く必要があり，伝送コードに何の制約もないこと（コード・トランスペアレンシー）が要求される．TECNETではJIS7ビット・コードを用いた拡張モードの二重DLE法を採用し，伝送路上の単位情報（パケット）のフォーマットを図3(a)のようにした．誤り検出に用いた巡回符号の生成多項式には次のものを用いた．

$$X^{16} + X^{12} + X^{5} + 1$$

伝送情報単位としては最大長を定めたパケットを用いており，長さがその最大値以内の場合はそのまま1つのパケットとしている．研究用であるので，パケット最大長としては（生データ最大長）様々な値が取れるハードウェア構成となっている（32～1024 B）．各パケット内情報の構成は図3(b)のようにヘッダ（6B）とテキストから成り，ヘッダは，発信計算機番号，着信計算機番号，メッセージ番号，パケット番号，制御コード（ACKパケット指定，メッセージACK指定，最終パケット指定等から成り，網内中継交換・リアセンブルに使われる．また各パケットのテキストの始めには通信リンク番号，メッセージバイト総数が入る．これはプロセス間通信制御用である．

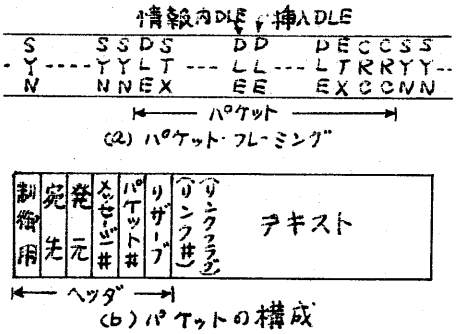


図3 伝送パケット

・ ハードウェア

通信制御装置は図4のような構成になっている．送信回線・受信回線毎にLCU送信部・受信部が付き，これらを組として4組迄接続が可能である．LCUはすべてCCC（共通制御部）内の装置と接続されていて，ここで各回線のマルチプレクス・デマルチプレクスをおこなう．ここまでの装置はすべての計算機に共通な部分である．各計算機に固有なインタフェスと合わせる為のハードウェアがこれに接続され計算機と結合している．これらはIC, MSI等で作られ，その規模はLCU送信部1台，LCU受信部1台，共通制御部ともに，それぞれ100ヶ程度のものになっている．研究室内手製である．

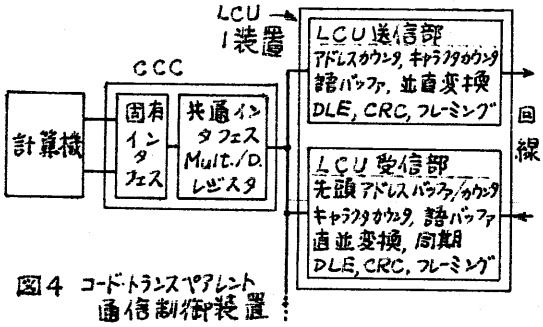


図4 コード・トランスペアレント通信制御装置

3. システム・ソフトウェア

3.1 TECNETのオペレーティング・システム

TECNETのオペレーティングシステムには，次のような特徴がある．

① プロセスの概念によるOSのモジュール構成．
② プロセス間協調方式はセマフォとプロセス間通信プリミティブによる2本立て．
③ ローカル・プロセスとリモート・プロセスの一様制御
④ 網上ファイルシステム

従来迄のOS設計ではプロセス間通信手段として様々な方法を用いており，一様性

3種のデータ記述
（① 言語プロセサや応用プログラムに必要な変数・データ構造についての詳細
② MMと外部記憶間の転送用IOルーチンに使われる情報
③ リソースアロケーションをするシステムルーチンに必要なデータ記述）

がないのが普通である．これはプロセッサを多くのプロセスで能率的に多重利用する場合は非常に勝れているが，プロセス間の構造が明確でなく，又大きなシステム開発の際デバッグや保守が難かしくなる欠点がある．そこでプロセス間協調の手段を一様化しすべてそれを用いることにすればこれらの欠点は補なわれるであろう．それが①である．しかし一様化による能率低下が問題で，特に多くのプロセスが次々と切替わる処理形態ではこの影響が大きい．しかし今後のOS設計，特に計算機網が重要になってくる場合にはこのプロセス一様管理が重要となるであろうと思われる．

②は，TECNET OS内で用いるプロセス間協調手法についてである．一般にプロセス間インタラクションには，共用リソースの相互排除と，プロセス間の情報伝達の2種あると考えられる．前者に役立つプリミティブとしてDIJKSTRAのセマフォー等があり，後者の機能もセマフォーを用いて実現できるしその方がより一様的となるが，ここであえて後者用として通信プリミティブを分離して別個に設けているのは，それぞれの特質を利用して適材適所に利用する為である．

③は計算機網向きOS設計という目的から来るもので，各計算機内で走っている数多くのプロセスを用いながら処理を進める場合に望ましい形態であろう．すなわち，図5のように計算機網上で自由に処理を実行するには，ローカルプロセスに対する協調と（P_1-P_2間），リモートプロセスに対する協調（P_2-P_3間）との扱いが一様であることが望ましい．勿論これは通信プリミティブが同一であるということであって，そのプリミティブの実行がローカルとリモートとでどう実行されるかには関知しない．それぞれに最も能率良いようにすればよい．

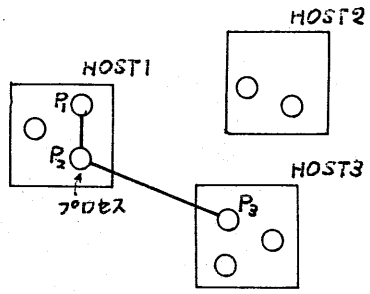


図5 計算機網上の処理形態

④も同様に計算機網上処理が自由である為には重要な要素であると考えられる．網内の全ファイルを各計算機ですべて管理することは実際上困難であるし，又ファイル構造が異なるので難かしい．逆に，全ファイルを一ヶ所に集中しデータ・ベースとして取扱うことも非能率であろう．従って各計算機のユーザが必要に応じて網上各所にあるファイルをアクセス出来，又よく利用するものは利用し易い形に組むことができることが望ましい．そこで単一計算機内のデータ管理システムを拡張し，各HOST内のファイル管理トリーの最上位を計算機名ノードとしてその上に全HOSTを統合するルート・ノードを設けるのが自然であろう．このようなシステムを制御するのが網上ファイル・システムである．勿論，各要素計算機内のデータ・フォーマットが同一でないと容易にはいかずデータ変換の問題が出て来るが，④の研究の第1段階として取りあえず，データ変換は各ユーザに任せる形で構成することにした．第2段階以降では，データ再構成システム，データ記述言語の開発と利用が問題になって来るものと思われる．

3．又　TECNET 制御プログラム

現在の所，HITAC 8800/8700のOS7，FACOM 230/25のBOSは手を加えることができないので，それぞれ通信回線から可能な利用モード（リモート・ジョブ・エントリ，データ伝送）に限られているが，他の5システムの計算機処理プログラムは研究室内で作成又は修正をおこなっている．その網向OSの基本構成は図6のようであって，

OS の核となる部分は，セマフォー，プロセス通信プリミティブ，プロセス生成／消滅を制御する部分で，プロセス・スケジューリングや，端末制御，回線制御，ファイル管理等はすべてこれを基礎にして作られている．

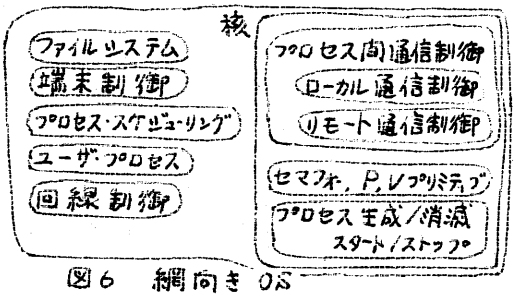


図6　網向き OS

ARPA 網の IMP制御プログラムは，ここでは回線制御に含まれ，NCP（網制御プログラム）はプロセス間通信制御のリモート制御部分に相当する．この OS の回線制御部は拡張モード回線用と基本モード回線用に分かれ，計算機によって多少異なるがバッファを除いて 0.5 kW，1.0 kW 程度である．また，FACOM 270/20 では網 OS の他に，バス結合複合計算機システム全体の制御プログラムが入る予定である．PPS-1 は網内の主要リソースであり，またファイル・システムの中心になる．この OS としては，OKITAC 4300C の網向き OS をエミュレーション・モードで利用することも考えており，更に3台あるプロセッサの1台を他計算機との通信専用とし通信制御装置をマイクロプログラム制御することを考えている．

4．プロセス間通信プロトコール

4．1 プロセス間通信の要素

プロセス間通信手法には様々な方式が利用又は提案されているが，一般にこれらの通信手法を特徴ずける事項は次のようなものである．すなわち，①通信相手の指定と通信要求マッチ（SEND／RECEIVE）の必要性　②待行列　③通信制御と情報転送の区別　④通信契約の設定　⑤伝送メッセージ量と伝送速度の制御　等がある．

②は各プロセス毎にその入力メッセージの待行列を持たせるか否かということで，この待行列は，両プロセスが独立に通信を実行できること，複数プロセスからの通信を容易に扱えること，優先度を付け易いこと等の点で便利である．③，④はメッセージ通信に先立ってまず相互で通信契約を取り交わし，リンクを設定しておいてから実際の情報転送をおこなうか，通信メッセージ毎に通信契約をしてメッセージ（単一）を伝送し終ると自動的にリンクが消滅するかの区別である．リンク方式は多量のデータを連続して伝送する場合リンク設定のオーバヘッドが少なくなるので能率的であるが反面，会話モードの場合はリンク設定と解消のオーバヘッドが無視できなくなる．一方，メッセージ毎の契約方式は，リンク設定と情報伝送が同時におこなえるし，リンク解消が不要，制御メッセージが少なくなるという利点を持つが，連続メッセージ転送の場合，オーバヘッドが問題になろう．ARPA ではファイル転送に重点を置いた形のリンク設定方式を採用している．

⑤は特にリモート通信での問題である．計算機各々側の情報処理速度は異なっているのが普通なので，自分の処理速度に比べてメッセージが早く届きすぎることを規制したり，各メッセージの大きさを相互の取定めで定めてから伝送する等が必要になる．ARPA 網ではこの為に特にバッファ制御コマンドが用意されている．しかし，メッセージ毎の通信契約を行なう方式を取ると，メッセージの伝送速度を相互に自由に制御できるので，これらの制御コマンドが不要になることも考えられる．

一方，計算機網上のプロセス間通信では，それら両プロセスが同一 HOST 内にある場合と互いに離れた HOST 内にある場合の区別なく一様に扱えることが望ましい．

しかしローカル通信とリモート通信とでは，能率・融通性に対する要求，伝送速度，遅延，通信誤りに対する考慮等が異なる．単一レベルの手法ではこれらの両方に合ったものを作るのが困難であるので，利用する上からは一様であるが実際上のオペレーションはローカル／リモートによって区別し，それぞれに合った動作をおこなうことが考えられる．

4．2　TECNET プロセス間通信プロトコール

TECNETでは網上プロセス間通信プロトコールを図7のような構成としている．

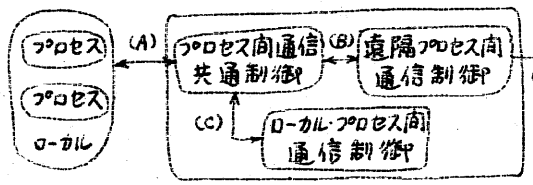


図7　TECNET プロセス間通信制御方式

通信プリミティブは(A)の所からそのオペレーションを呼ぶという形で利用されるが，まずプロセス間通信共通制御に入り相手プロセスがローカルの場合は(C)の所からローカル プロセス間通信を実行し，リモートの場合は(B)のインタフェースを経てリモート・プロセス間通信制御に制御が渡される．これはリモートHOST内の遠隔プロセス間通信制御と交信して実際の伝送を実行する．(A)のプロトコールは，このHOSTから見た網上プロセス間通信プロトコールであるが，これはすなわちローカル・プロセス間通信プロトコール（LCP）自身に等しい．一方遠隔プロセス間通信プロトコール(RCP)はこのLCPに埋め込まれプロセスからは見えないが，各HOSTによってLCPが異なる場合でもRCP自身は同一でなければならず，従って(B)の所がそのインタフェースとなる．

- LCP（Local Communication Protocol）

図7(A)に於けるプロトコールはプロセス間で実際に利用される立場のプロトコールである．ここでは次のような通信プリミティブとした．

SEND（s, r, m, ℓ）　　RECV（s, r, b, r_s, ℓ）
RPLY（p, r_m），　　CLS（s, r）

但し，r：受信プロセス名，　m：メッセージポインタ，　s：送信プロセス名
b：受信バッファポインタ，　p：相手プロセス名，　な：リスタート・ロケーション，　ℓ：リンクモード指定フラグ，　r_m：応答メッセージポインタ

SENDはメッセージ送信要求，RECVは受信要求で，これらのプリミティブが出されるとテーブルに登録される．双方から出された時通信要求は成立しmで示されるメッセージがプロセスsからプロセスrへ送出され，bで示されるメッセージバッファ内に収められる．この通信が終了するとRPLYプリミティブがプリミティブに対する応答としてs, rの双方へそれぞれ返され，r_mで示される応答メッセージが渡されてテーブルはクリアされる．ℓはリンクモード指定に用いるもので，これが立っている(ℓ=1)，リンクモードとなり後程CLS（Close）等によって切られる迄そのリンクは存続する．

- RCP（Remote Communication Protocol）

図7(D)で示される間のコマンドセットが遠隔プロセス間通信プロトコールであり，これは各HOSTのLCPによって起動される．ARPA網に於けるHOST-HOSTプロトコールがこれに相当する．TECNETで用いているコマンド・セットは次の通りである．

SRQ (S, R, i, n_m), RRQ (S, R, n_b)
CLS (S, R), RST, MAK (i, ℓ)

但し, S：送信プロセスの中介ネーム, R：受信プロセスの中介ネーム
i：リンク番号, n_m：メッセージ長, n_b：メッセージ・バッファ長
ℓ：リンクモード指定 (ℓ=1：リンクモード, 0：非リンクモード)

SRQ は Send Request, RRQ は Receive Request で, これら両コマンドが双方の間で交されると通信契約が成立する. 成立すると送信側は自動的にメッセージを送信側に送出する. この場合, i はその契約のIDを示すリンク番号であって, 送信側が管理していると同時に送出するメッセージのヘッダの後には必ずこの番号を入れて送出する. 受信側では, SRQ コマンドによって i を以前に受けているのでプロセス間通信 (S, R 間) を識別することができる. S, R は送受信プロセスの網上に於けるユニークな中介ネームを表わしている. 各HOST 内に於ける実際のプロセス名 (ローカルネーム) S, R とこれらの間のマッピングは各所の遠隔プロセス間通信制御プログラム (RCCP) に於いておこなう. n_m, n_b はそれぞれメッセージ, 受信バッファの長さであるが, 通信契約が交される時 n_m と n_b は異なっていてよい. この場合, $n_m \leq n_b$ の時は1回の伝送で終了するが, $n_m > n_b$ の時は次のようになる. まず第1回目のメッセージ転送では n_b バイトで実行される. これにより RRQ は満されたが SRQ は未だ満されていないので, 受信側 HOST では RRQ を未だ受取らぬ SRQ があるものとみなして RPLY によりもう一度バッファを要求し, プロセスR から RECV が出されてバッファを与えられればMAKを送信側に返し完全にメッセージ転送が終了する迄続けられる.

又, メッセージが受信側に届くと MAK (Message ACK) を送信側に送り返し受信を通知する. これで通信契約は解消しリンクは切断される. これは非リンクモード時の動作であるが, LCP でリンクモードを指定するとリンクモードMAKを受信してもリンクは切れない. リンクモード指定は上コマンドを用いずメッセージ内に含めて伝送する. 即, 通信契約が SRQ-RRQ マッチで成立すると自動的にメッセージが送出されるが, それにはリンク・モード・フラグが立っておりMAK 返送時もフラグを立てて返される. CLS は Close で, リンクモード時では送信側・受信側いずれの側からでもリンクを切断したい時に出し, 非リンクモード時では相手からの要求を断わる場合に用いる. 切断要求を受けると相手に CLS を送り確認してから断となる. RST は Reset であって, あるHOST 間でのプロセス間通信の初期化を行なう為に用いる.

◦ プロセス名とプロセスの網上中介名

プロセス名は各HOST に於いてユニークに管理されているが, LCP で用いるプロセス名は網上でユニークに識別できるネームでなければならない. 例えば, HOST 名とプロセス・ローカル・ネームの結合は一つのグローバル・ネームとなるが, ローカル・ネームの付け方は各HOST によって様々であるから, 網上で唯一に識別するには標準形式の中介プロセス名 (ARPA網のソケットに相当する) に変換し, そのネームで通信する必要がある. 従ってRCP で用いるプロセス識別名は次のようになる.

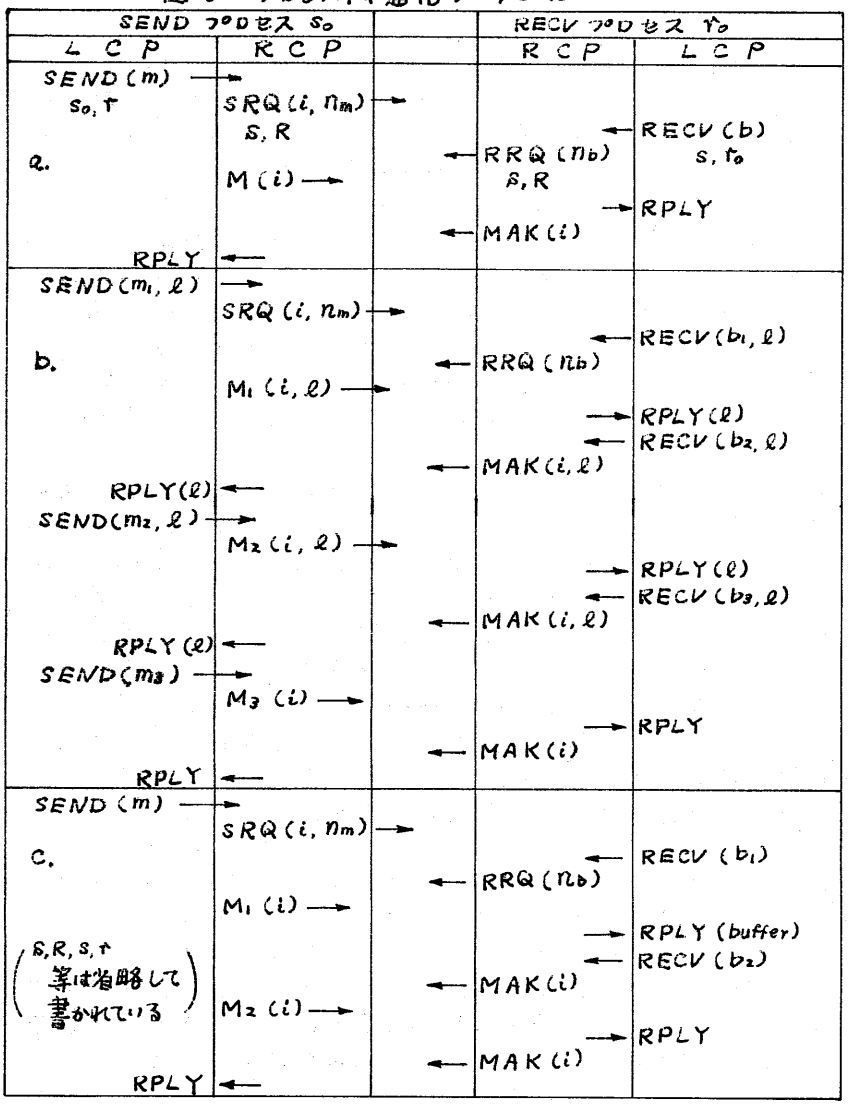
HOST名 + プロセスの中介ネーム

また, あるHOST から見たリモート・プロセスの識別ネームには, ①リモートHOST内のローカル・ネーム ②ローカルHOST 内に於けるその対応ローカル・ネーム ③標準中介ネーム の3種があり, 各HOST ではこれらを適当に対応させて通信をおこなう.

4. プロトコールの使用例

前節で与えたプロトコールをより詳しく説明する意味でその使用例をあげる．これらの例では両HOST内のLCPは同一としておき，HOST1のプロセスs_0が，HOST2のプロセスt_0（それぞれの中介名はS，R）にメッセージM(i,ℓ)を伝送する場合を想定する（i，ℓはリンク番号とフラグ）．この場合，ネームs_0，t_0はそれぞれのHOSTに於けるローカルネームであるが，s_0のHOST2に於けるローカルネームをs，t_0のHOST1に於けるローカルネームをtとしておく．図8に次の3つの場合の使用例を示す．

図8　プロセス間通信プロトコール



a．単一メッセージ伝送（$n_m \leqq n_b$）

b．複数メッセージ伝送（$n_m \leqq n_b$）

c．単一メッセージ伝送（$2n_b \geqq n_m > n_b$）

a.ではSENDが出されるとRCCP（遠隔プロセス間通信プログラム）はそれに対応してSRQを出す．HOST2に於いてプロセスt_0がRECVを出すとRRQが送られ，HOST1で受信するとSEND／RECVのマッチが成立し，SENDで指定されたメッセージが送出される．HOST2で受信するとRPLYがプロセスt_0に返され，同時にMAKがHOST1に返される．HOST1のRCCPはMAKを受信すると送信完了を知りRPLYをプロセスs_0に返して完了を通知する．

b．はリンクモードの場合で，両プロセスがリンクモードを希望することによって成立しリンク・モード・フラグ付のメッセージM_1が送られる．この時，RPLYによってM_1到着を知ったプロセスt_0は次の受信が可能であるとRECVを出し，それに応じてRCCPはMAKを返送する．第2回目のRECVがヌリンクモードを指定するとこのMAKにフラグが立てら

れる．RPLYでそれを知ったプロセスS。は，次のSENDをおこなう（高速化の為に前以って出しておいてもよい）．これによりメッセージが送出される．最後のメッセージ送信時にプロセスS。は非リンクモードのSENDを出し，メッセージのリンク・モードフラグは立てない．これを受けた受信側ではMAKのモードフラグをリセットして返送し，これによって伝送が完了となり，リンクが解消する．

C.は受信バッファより大きなメッセージを送出する場合で，この場合は単一メッセージ送信でも何回かに分けて送出される．例では2回である．1回目の送出がおこなわれるとRPLYがプロセスAに返され再びバッファを要求する．プロセスAがRECVを出すとMAKが返送され，送信側ではこれによって2回目の送出を開始する．これで完送になると，HOST2のRCCPは完了をRPLYによってプロセスAに伝えMAKを送出する．MAKを受けたHOST1のRCCPは完了のRPLYをプロセスS。に返す．

本プロトコールでは，通信の速度調節をSEND／RECVを出すタイミングによっておこなっている為，その為の制御コマンドは不要である．又バッファ割当については SEND／RECVマッチの時に契約を取り交わす他，毎回の伝送毎にその制御が実行される．リンク設定が簡単であるので単一メッセージの転送の場合にオーバヘッドが少なく，又複数メッセージ転送時も毎回リンク設定をする必要はなく能率が高い，又グローバルなプロセス間通信プロトコールが簡単である等の利点がある．

5．むすび

幾つかの計算機を直接に又は通信回線で結合し相互のリソース共有を目的とした電子計算機網構成の研究用として，我々の所ではTECNETシステムを開発し様々な検討をおこなっている．ここで御報告したのはその研究概要であるが，現在の所，基本伝送制御手順回線の部分とバス結合部分のハードウェア及び制御プログラムは完成して動いており，コード・トランスペアレント 回線はハードウェア試作が終り増設をおこなっており，その制御プログラムはデバッグ段階にある．今後の予定としては，昭和49年春に増設が完了し，網上OSもV1版が完成する予定で，マルチ・マイクロプロセッサPPS-1のエミュレータ完成を待って網上OSの網上プロセス間通信を利用した諸実験をおこなってゆく積りである．本文で提案したプロセス間通信プロトコールもそれらの実験を通してより改良されてゆくであろう．

参考文献

1) 勝又裕，元岡達：共通母線による計算機群結合方式，電子計算機研究会資料，EC-71-22（1971-09）．

2) A. Mckenzie : HOST-HOST protocol for the ARPA network, Jan. 1972 NIC 8246.

3) D. C. Walden : A system for interprocess communication in a resource sharing computer network, Comm. ACM, Vol. 15, No. 4, April 1972, pp. 221-230.

4) P. B. Hansen : The nucleus of a multiprogramming system, Comm. ACM, Vol. 13 No. 4, April 1970, pp. 238-250.

（盛文社）