

コンピュータ・ネットワーク JIPNET*

山本 欣子**

1. まえがき

JIPNET (JIPDEC Integrated Project Network) は、実用と実験の両目的を持つ分散型異機種結合のコンピュータ・ネットワークである。第1ステップとしてインハウスの3台のコンピュータ、FACOM 230-75、HITAC 8450および、NEAC システム 700の3台を HOST とし図-1 に示すように3台の IMP および TIP と 48 kBPS の回線により結合されている。

JIPNET は、リモートバッチ、TSS 等の処理形態を含む汎用のリソース・シェアリング・ネットワークのモデルであり、ハード、ソフト、ファイルおよび特殊入出力の共用、TIP 端末からのネットワーク・アクセスなどの実用的目的と同時に、異機種間の分散型データ・ベース・マネージメントあるいは仮想ネットワークの有効性の基礎研究などの実験的目的も持つものである。

このプロジェクトは 1973 年に4年計画で開始され、1974 年末時点で、サブネット、2HOST の NCP および基本的な高位プロトコルのインプリメントを終了し、12月に TIP からの TSS 利用を含む実用プログラムを対象に各種の利用法のデモンストレーションを実施した。今後はより高度なネットワーク利用の実験を行うと共に、基本機能の信頼性、効率などの評価およびその改良を重ねて行く予定である。今回は本プロジェクトの半ばにあたり、現在までの成果および評価の一端を紹介する。

2. サブネットのコントロール

JIPNET におけるネットワーク・コントロール機能の、サブネットと HOST との負荷分担の基本線は、データ伝送の信頼性および効率にかかわる一切の

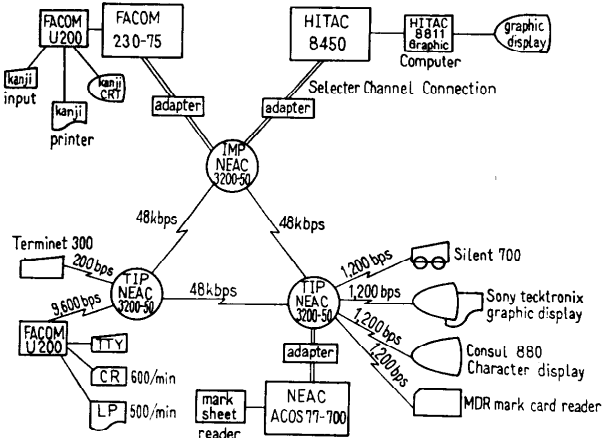


図-1 JIPNET の構成

処理はサブネットの分担とし、逆に HOST オリエンテッドな部分は各 HOST で吸収する。また拡張性の面から、すべての IMP はハード、ソフトとも同一のものとしている。

サブネット・コントロールの設計にあたり、次のような点に特に留意した。

(1) 信頼性: CRC の採用によりメッセージ伝送の誤り率を下げるとともに、フェイルソフト (failsoft) 機能としては、IMP と HOST 間および IMP 同士に互いに依存関係がなく、一部の障害がネットワーク全体に及ぼす影響を最少限におさえる。

(2) 効率性: 会話型処理のレスポンスは十分早く、かつ大量データ転送においては十分なスループットが得られること。また同一 IMP に結合された HOST 間では、特に高速伝送が可能であること。

(3) 拡張性: ネットワークの拡張やポロジの変化に対し容易に追従し得ること、現在のソフトウェアでは、これらの変化に際しての変更は数語以内である。

2.1 SCP

SCP (Subnet Control Program) の主な機能は次の

* Computer Network JIPNET by Kinko YAMAMOTO (Japan Information Processing Development Center)

** (財) 日本情報処理開発センター

ようなものである。

(1) 蓄積交換 (Store & Forward)

発信 HOST からのメッセージは、発信地 IMP でパケット (Packet) 化され、各パケットは任意の経路を経て目的地 IMP に到着し、ここで再びメッセージにリアンブル (reassemble) され、受信 HOST に送られる。受信の確認として隣接 IMP 間では ACK が返され、両 HOST 間では RECEIVE が返送される。ACK の返送は原則として逆方向のパケットに便乗して返送される。ACK, あるいは RECEIVE の返信により、発信地に保管されていたコピーは棄却される。逆にこれらの返送が遅れた場合、原則としてタイマーの管理により再伝送が行なわれる。

伝送効率を向上させるため、隣接 IMP 間を論理的な4つのチャンネルに分割し、1つのパケットは1つのチャンネルを使って伝送される。このチャンネル数は国内で考え得る最長 IMP 間の距離 (例えば、東京—札幌間約 800 km) と 48 kbps の伝送速度に基づき定められている。

(2) ルーティング (Routing)

基本的には適応ルーティングを採用することとしたが、トラフィック・オーバーヘッドを避け得る固定ルーティングのメリットも捨て切れず、両者の折衷的手法として Shortest Queue+Bias (SQ+B) ルーティングにおいて、サブネット内の回線や IMP の up/down にともなう重大な変化の際にのみバイアステーブル (Bias table) を更新する方法を採用することとした。

(3) フロー・コントロール (Flow control)

発信地 IMP と目的地 IMP の間にパイプ (pipe) と呼ぶ論理的なパスを考え、そのパイプの容量を制限することによってフロー制御を行なっている。

シングル・パケット・メッセージ (144 byte 以下) のための S-pipe とマルチ・パケット・メッセージ (1152 bytes 以下) のための L-pipe とを設け、それぞれ独立に管理することによって、短いメッセージの伝送に長いメッセージの伝送に要するオーバーヘッドの影響を与えない配慮をしている。現在 S-pipe の容量は4、L-pipe の容量は1としている。

なお、マルチ・パケット・メッセージは目的地 IMP 内にそのリアンブル用のバッファ・エリアのリザーブが行なわれたという確認を得て始めて発信 HOST から発信地 IMP に送り出される。

(4) シークエンシング (Sequencing)

各メッセージには、0 から 255 の番号が round-ing に付けられており、この番号によって順序づけが行なわれる。2つの HOST 間のコネクションにおいて、1つのコネクションに対しては S-pipe か L-pipe かどちらか一方のパイプのみが対応するため、シークエンシングにおいても両パイプはそれぞれ別々に管理されている。

また、シークエンシングの機能を利用して、メッセージの紛失や重複の発見、あるいは RECEIVE の消滅などのエラー検出も行なっている。

(5) ロックアップ対策

JIPNET では、次の4種類のロック・アップ (lock up) を想定し、対策を考慮した。

- ① Store & forward lock up
- ② Reassemble lock up
- ③ Sequencing lock up
- ④ HOST/IMP lock up

③は1つの目的地にメッセージが集中する場合に発生するものであり、④はHOST-IMP間のアダプタが半二重のために発生するものである。

これらはIMP内のバッファ管理あるいは転送の優先権の設定などによって防御する手段を講じた。

(6) 障害対策

サブネットにおける主な障害はIMP障害、回線障害、ディスコネクション (disconnection) などであり、IMP障害はその回りのすべての回線障害とみなして処理する。

また、ディスコネクションの発見は各IMPが図-2に示すようなコネクション行列を持っており、何らかの障害にともない行列の更新が行なわれ、グラフ理論の利用による行列演算によって各IMPで他のIMPとのディスコネクション状態を検出する。

2.2 サブネットの効率

サブネットの効率測定のため、Artificial traffic ge-

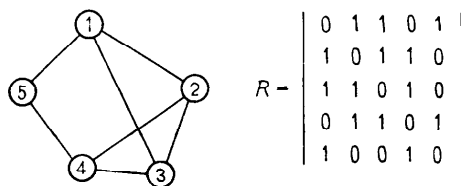


図-2 コネクション行列

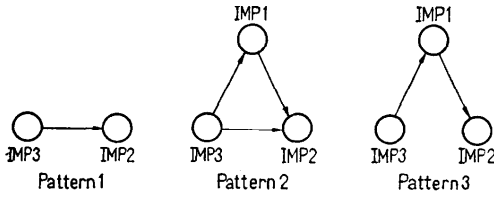


図-3 測定パターン

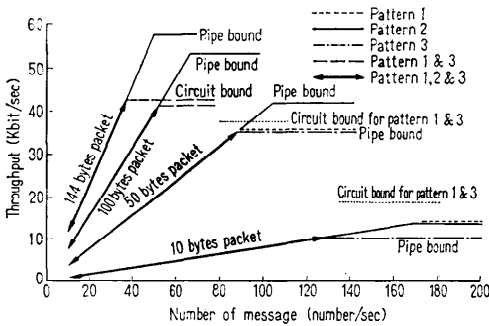


図-4 シングル・パケット・メッセージのスループット

nerator を SCP 内に付け加え、スループット (throughput) とラウンド・トリップ・タイム (round trip time) を測定した。測定にあたっての3つの pattern を図-3 に示す。

図-4 はシングル・パケット・メッセージに対するスループットであり、これによれば短いパケットにおいてはフロー・コントロール機構 (パイプの容量) により伝送量がおさえられ、パケット長が長くなると 48k BPS 回線の伝送容量によっておさえられる事がわかる。また経路が1つの pattern 1 および3と、経路が2つの pattern 2 とを比較すると、後者はルーティ

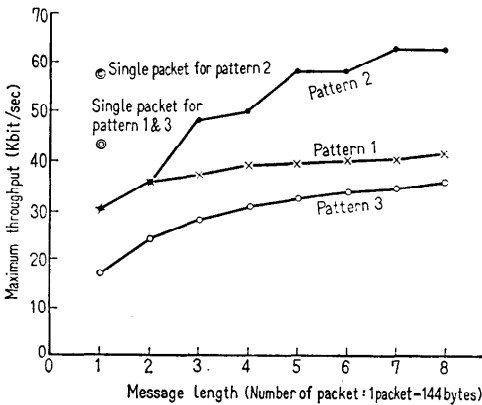


図-5 マルチ・パケット・メッセージのスループット

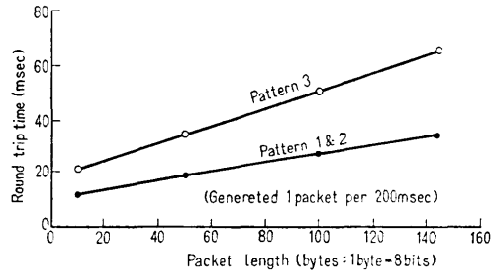


図-6 ラウンド・トリップ時間

ングの効果によってパケットが分流され、回線バンドが生じない事がわかる。

図-5 はマルチ・パケット・メッセージに対する最大スループットを示す。シングル・パケットとしての測定値 (◎印) と、1パケットの場合の差は、前者が容量4の S-pipe を通した結果であるのに対して、後者は容量1の L-pipe を通したことによるものである。pattern 2 においてパケット数が3, 5, 7の場合にスループットが急上昇するのは、ルーティングにより2つの経路に分流されることの効果である。

図-6 はシングル・パケットのラウンド・トリップ・タイムである。この測定値と、SCP 内での各タスク間の遷移経路から、1つの IMP が1パケットの蓄積交換に要する時間は約 1.16 ms であるという結果が得られた。

2.3 検討事項

以上のような測定あるいはインプリメントを通して次のような問題点を指摘することができる。

(1) 割込みの CPU 利用率: 現在の SCP においては割込み処理の CPU 利用率は 30~40% と高率であり、この主原因は、SCP のプログラム構造にコントロール・プログラムによるディスパッチ形式を採用したことによる。この形式はシステムの早期安定化には極めて有効であったが、効率の点からはコントロール・プログラムを持たず、割込み主導により CPU が各タスクにディスパッチされる方式が望ましいと言える。

(2) ルーピング: 適応ルーティング一般に於いては、ルーピング、ピンポン現象などの対策が不充分である。例えば図-7 (次頁参照) において、 i から d へのパケットを i が j に送ったとすれば、必ず再び i に戻って来る (return back path select)。このような無駄なトラフィックの防止は、例えばコネクション行列を利用して i に結合されているすべての回線が故障したもとして j と d とがディスコネクション状態にな

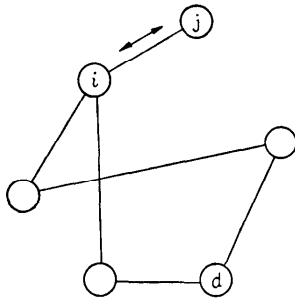


図-7 Return back path select

るかならぬかによって、 j が return back path であるか否かを判断することは可能であるが、全体の効率から見て、どこまでの処理負荷が妥当であるかは今後の検討課題である。

(3) ディスコネクションの検出：前述のようにディスコネクションの検出にはコネクション行列を使用しているが、行列の一要素を 1 bit に対応させているため、1 word の bit 数とノード数との関連が強く、ノード数を 2 倍にすれば処理時間は約 2.5 倍となる。

16 ノードまでの現在の JIPNET では、各 IMP においてディスコネクションを検出するに要する時間は約 3 ms であり、IMP がネットワーク・トポロジ情報を持たぬ ARPANET における手法に比べると、各 IMP が自己に対するディスコネクション IMP をかなり早期に発見し得るが、ノード数が百を越えるネットワークでは、容量および処理負荷は無視できぬものとなる。

(4) 半二重：HOST-IMP 間のインタフェース・アダプタ (Interface adaptor) は全二重にすべきである。半二重の故に HOST/IMP ロック・アップが起るばかりでなく、両方向のコマンドの衝突が、SCP 内、アダプタ内、HOST のモニタ内など各所で発生し、それぞれの処置を行なうのはかなり面倒であった。

3. HOST のネットワーク・コントロール機能

これについては本号の「コンピュータ・ネットワークにおける HOST-HOST プロトコル」で多くの関連する問題につき述べているので、ここではなるべく重複せぬ部分をとり上げる事とする。

3.1 NCP

NCP (Network Control Program) は、HOST-

HOST および HOST-IMP プロトコルにもとづき、サブネットを通しての他 HOST とのコミュニケーションと、各 HOST 内のプロセスに対してサービスを行なうものであり、次のような機能を持つ。

(1) プロセス間のコミュニケーション

プロセス間のコネクションの確立とその閉鎖、コネクションを通してのデータの授受、プロセス間の同期などの処理を行なう。コネクションの確立とは、2つの HOST のそれぞれのプロセス間で、一方の入力が他方の出力となるロジカルなパスを作ることである。各プロセスには HOST 番号とユーザ番号とからなるネットワークに対してグローバルな id がつけられ、JIPNET ではこれを port-id と呼ぶ。1つのコネクションは2つのプロセスのそれぞれの port-id のペアに対して確立される。

(2) HOST 間のフローコントロール

WABT (WAit Before Transmit) 方式を採用している。HOST 間で1つのコネクションが確立すると、最低1個の基底バッファが与えられるが、それ以上は必要に応じてバッファプール (Buffer Pool) から取り出して供給され、使用後はプールに返却される。プールに残ったバッファ数が、ある限度を切った場合、多量のバッファを占有しているコネクションに対しては発信 HOST にデータ送出の一時停止を要求する。このコネクションに対しては、基底バッファ内のデータが処理のため明け渡された時点で送信の再開を許す。

一つの HOST で一時に確立し得るコネクション数は 30 前後であり、バッチ処理の場合は1コネクション当たり 2 バッファ×4 パケット、TSS 処理の場合は 1 バッファ×1 パケットの割合でバッファ数を想定し、100 ないし 150 パケット分のバッファを持つ。

3.2 コントロール・コマンドとサービス・コマンド

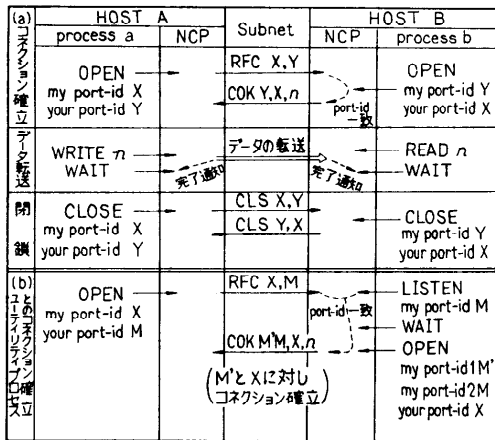
2つの HOST の NCP 間は、コントロール・コマンド、即ち HOST-HOST プロトコル・コマンドにより交信される。一方ユーザ・プロセスと NCP 間にはサービス・コマンドが設けられており、サービス・コマンドには多重コネクション管理、フロー・コントロールなどのマクロな機能を含んでいる。図-8 (次頁参照) に両コマンドの種類を、図-9 (次頁参照) に両者の関連の例を示す。(b)は M がユーティリティ・プロセスの場合である。

3.3 高位プロトコルの処理

現在は、DSP (Demand Service Protocol), RBP (Remote Batch Protocol) および FTP (File Transfer

コントロール・コマンド		サービス・コマンド	
RFC	コネクション確立要求	OPEN	コネクション確立要求
COK	コネクション確立応答	CLOSE	コネクションの閉鎖
CLS	コネクションの閉鎖	READ	コネクションを通してデータを受ける
CEASE	データ送信の一時停止	WRITE	コネクションを通してデータを送る
RESUME	データ送信の再開	LISTEN	コネクション確立要求待ち
POST	コントロール情報の通知	WAIT	イベント待ち
LOADE	プロセスの起動	POST	コントロール情報の通知
DELP	プロセスの消滅		
CNT	追加情報		

図-8 コントロール・コマンドとサービス・コマンド



(nはコネクション番号)

図-9 両コマンドの関連

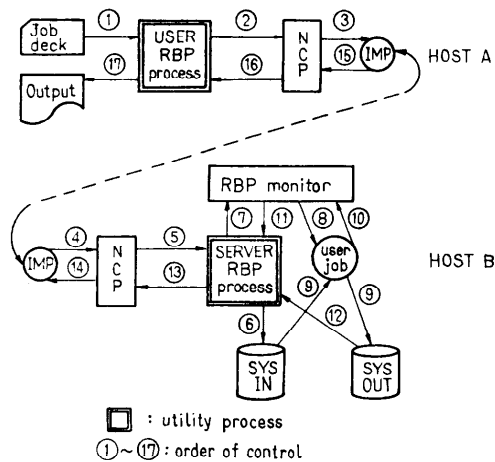


図-10 RBP 処理の概念図

Protocol) がサポートされている。高位プロトコルの処理は拡張性を考慮し、各 HOST のユーティリティ・プロセスの形で実現している。図-10 に RBP プロ

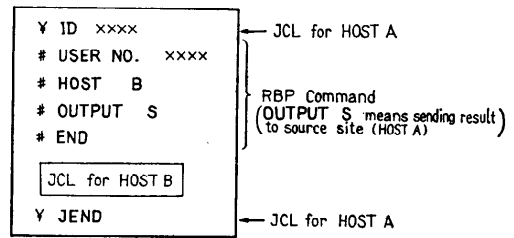


図-11 RBP のジョブ・デック

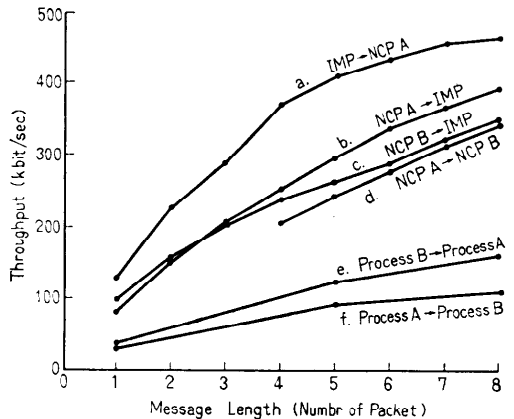


図-12 NCP 効率の測定

セス処理の概念図を示し、図-11 はそれに対応するジョブ・デックの例である。

3.4 NCP 効率

ユーザプロセス、NCP、IMP の3点に、それぞれ Artificial Message Generator/Receiver を挿入して2つの HOST A,B でスループットを測定した。結果の一部を図-12 に示す。これによると NCP-IMP 間の最大スループットは 350~460 kBPS, NCP-NCP 間は 340 k BPS, ユーザ・プロセス間は 110~160 k BPS である。また HOST A は NCP をユーザプログラムとして作成し、HOST B はサブモナとして作成している事による割込み処理のオーバヘッドの多少の差が e, f に表われている。また HOST A のコマンドチェーンの機能が、かなり効果をあげている事も認められた (b と c の差)。

3.5 NCP 作成上の問題点

NCP は各 HOST の制御プログラムの一部、サブモナ、あるいはユーザプログラムなどのいずれかの形で既存 OS に付け加えられることとなるが、ネットワーク形成の基本精神にのっとり、既存 OS にできる限り手を加えず、なおかつコントロール・プログラム

としての効率を保つには、サブモニタ型が原則的には望ましい。しかしサブモニタ作成には、当然既存 OS にその受け入れ機能が備えられている事が前提となり、同時に他機能とのインタフェースを明確に記述したドキュメントが整備されていないとてはならない。将来サブネットが VAN あるいは特殊通信網としてサービスされた際においても、各 HOST の NCP の作成は今後とも残される作業となるであろうから、ドキュメントの整備と公開は今後メーカ(あるいは OS 作成者)の十分な協力を要請したい問題である。

更に一步進んで既存 OS を改造し、NCP を制御プログラムの 1 部とする場合は、ネットワークに無関係なローカル・ジョブに対するオーバーヘッドを増加させぬ配慮が必要であろう。しかし、この形で NCP を作成するには当然、既存 OS 内部の十分な知識が必須条件となる。

一方、ユーザプログラムとして NCP を作成する場合、次のような点が問題となる可能性がある。

- ① 現在の OS ではファイルの創成、抹消はユーザプログラムでは不可能である。
- ② ファイル保護機能のため他人のプログラムに属するファイルの Read, Write が禁止されている。
- ③ 他の 1 つのプロセスとの交信は可能だが、複数のプロセスとの同時交信はむずかしい。
- ④ 割込み禁止モードで動く事がむずかしい。
- ⑤ 自 HOST のリソースの利用情報を取り出せぬ恐れがある。
- ⑥ ms 以下のタイマが利用できぬ場合がある。

いずれにしろ NCP の作成は既存 OS とのかかわり合いが深く、従来の OS は、夫々のシステムが独立して働くことを前提として設計されており、コンピュータ間通信はむしろ異端的な扱いとなっている。NPL の Davies 等は、将来の OS は NCP の下に従来の OS 機能が従属すべきであると提唱しているが、少なくとも今後の OS は他のコンピュータとのコミュニケーションを前提として設計されるべきであろう。NCP 作成上、既存 OS に対する最低限の要求事項については、本号の「HOST プロトコル……」に記述されているので参考にして頂きたい。

3.6 検討事項

サブネットと HOST の分担にかかわる問題も含め、HOST のコントロール機能に関する主な検討事項を挙げる。

- (1) メッセージ交換型とコネクション設定型:

JIPNET においては、汎用ネットワークの性格からコネクション設定型を採用した。これはパケット交換網に回線交換的な論理を持ち込む事により、大容量ファイル転送の効率を上げ、かつコネクション単位のフロー・コントロールを容易にするためである。

(2) 単方向、両方向のコネクション: プロセス間のコネクションにおいて現在は単方向方式をとっているが、コネクション要求の交差によるデッドロックの防止が必要となり、またオーバーヘッドの減少の点からも両方向コネクションがより望ましいのではないかと検討中である。

(3) WABT 方式フロー・コントロール: この方式では受信側がバッファ不足に陥ると当該コネクションのデータ伝送を一時中止するが、この中止指令や再開指令を発する時点と、データ送信側がそれを受ける時点とのタイミングのずれにより、その間に送出あるいは受信されたデータの扱いがノーマルに処理されぬ恐れがある。より高位のプロトコルにおいて、これをカバーする手段が必要となる。

(4) メッセージ長のコントロール: CYCLADES はサブネット上でのメッセージはすべてシングルパケットのみとしている。これはパケット化やリアセンブルを HOST の負担としている、という言い方もできる。これによって当然サブネットの負担はかなり軽くなる。JIPNET ではロング・メッセージとして 8 パケットすなわち 1,152 bytes までのメッセージを許しているが、転送データの実長はそれより遙かに長い(例えば MT 一本)ケースは多く、いずれにしろ高位プロトコル (FTP) においてデータの分割と再編成処理を行わざるを得ない。データ長のコントロールをサブネット、HOST の基本機能あるいは高位プロトコルのいずれでどう分担するかは、そのネットワークで使用されるメッセージ長の要求とからみ、今後の一つの課題であろう。

(5) コネクションの切り換え: ネットワークにおける HOST 間の自動的なロードシェアリングを実現する場合には、コネクションの自動切り換えが発生する可能性がある。この機能を HOST のベーシックレベルでおこなうか、高位プロトコルで行うかは 1 つの問題点である。JIPNET においては一応前者の立場をとり、それに対応するため RECEIVE の確認を HOST 間で行なっている。コンピュータ・ネットワークにおいては基本的に HOST のベーシック機能を大きくする事は極力さけるべきであり、その意味からコネ

クシヨク切り換えは高位プロトコルに、かつ RECEIVE の返送確認はサブネットにと、夫々譲り渡すべきか否かを検討中である。

4. む す び

紙面の都合上、言い尽くせぬ点が多々生じたと同時に、かなりの説明不足もお詫び致したい。JIPNETの

詳細は下記の文献に記述されている。

参 考 文 献

- 1) 日本情報処理開発センター：コンピュータ・ネットワーク JIPNETの研究開発, 49-S001(1975)
- 2) T. Ito, K. Yamamoto: Packet switching subnet of JIPNET system, second UJCC, (1975)

(昭和50年4月17日受付)