≪論　文≫

インハウス・コンピュータ・ネットワークと HOST コンピュータ*

坂井利之** 田畑孝一**
大西廣一*** 北澤茂良***

Abstract

Computer network is defined that several standing-alone computer systems (Hosts) are combined on an equal footing with each other through high-speed communication network.

The KUIPNET (Kyoto University Information Processing NETwork) is an inhouse-oriented computer network developed in accordance with the above philosophy, while most announced inhouse systems are constructed rather on the basis of computer-complex configuration. KUIPNET is also prepared to join a prospective wider-distributed computer network via 48 kbps lines.

In KUIPNET, we can transmit the data at more than 200 kbps (effective continuous transfer rate) between two processes in different Hosts, which allows Hosts to share the resources of intelligent terminals in real-time.

Two Hosts in KUIPNET are minicomputers in which we have been able to implement reasonable-sized Network Control Programs.

In this paper the aspect of Hosts in KUIPNET is mainly described.

1. まえがき

散在するコンピュータ・システム (HOST と呼ばれる) を高速通信回線で結び，互いのシステムの情報資源を共有しようというコンピュータ・ネットワークの重要性が認識されつつある．

一般に，コンピュータ・ネットワークとは独立した HOST を対等の立場で通信網で結ぶもので，ネットワークの加入者となる各 HOST は自立して単独でも運転可能である．そして要求に応じ通信網で互いにメッセージを交換するのであって，管理プログラムの観点からも計算機結合の観点からもその結合はもっともゆるいものである．

一方，従来のインハウス・ネットワーク・システムでは，コンピュータはチャネル結合，共通バス結合，環状バス結合など，その結合はかなり密で，いわゆるコンピュータ・コンプレックスの結合形態をとるものが多い．これらの中にはコンピュータ・システム全体をネットワークの加入者と考えるよりも，むしろコンピュータ・システムをその構成要素であるプロセッサ，メモリ，ディスク・ファイルなどに分解して，これらをネットワークの加入者と考えるものもある．

本論文で報告する KUIPNET (Kyoto University Information Processing NETwork; 1973 年秋，第一期完成，**Fig. 1**(次頁参照)) は前者のコンピュータ・ネットワークの思想に基づいてインハウス向きに構成されたもので，典型的な「インハウス・コンピュータ・ネットワーク」の例としての意義を持つ．

インハウス向きに構成された KUIPNET は，いわば電話の PBX で，また広域コンピュータ・ネットワーク (公衆電話網) に加入可能なように当初から考慮していることもあって，本格的な広域コンピュータ・ネットワークとして知られる米国 ARPANET の構成思想に準じた方式を取り入れているが，それに加えて次の 2 点を新たに検討し，いずれも肯定的な結論を得

* Inhouse Computer Network and HOST Computer by Toshiyuki SAKAI, Koh-ichi TABATA, Hirokazn OHNISHI, and Shigeyoshi KITAZAWA (Faculty of Engineering, Kyoto University)
** 京都大学工学部情報工学教室
*** 京都大学大学院工学研究科

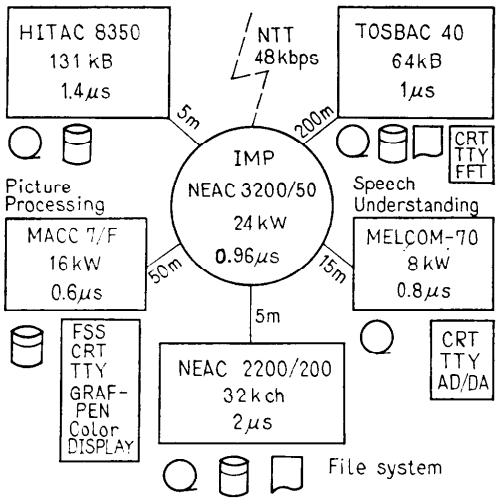

Fig. 1 Organization of KUIPNET

た.

(1) ローカル・ネットワークの増強: KUIPNET の HOST は同一 IMP (Interface Message Processor) につながっていて, 広域ネットワークの観点からはローカル・ネットワークとみなされる. KUIPNET では, 広域的機能を低下させることなく, インハウス (すなわちローカル) の任意の一対の HOST 内のプロセス間で, 高速 (実効連続転送速度 200 kbps 以上) のリアルタイム性の強い通信を他の HOST 対の通信より優先して行なうことが可能になっている. これは音声・画像などの高速大量データを処理するのに必要な機能であるが, 一方これと並行して存在する他の HOST 対での通常のキャラクタ・メッセージの通信に著しい妨害を与えないこと.

(2) ミニコンピュータ HOST: コンピュータ・ネットワークにはいかなる機種の計算機システムも加入可能というが, ミニコンピュータのようなシステムでもその加入は有効かの検討. すなわちミニコンピュータ上に合理的なサイズでネットワーク・コントロール・プログラムを新しく加えてインプリメントが可能かを検討した.

コンピュータ・ネットワークのうち, HOST を除いた通信網の部分を「サブネット」と呼ぶ. HOST に対するサブネットの任務は, ある HOST から指定された他の HOST に任意のビット系列を正しく必要とする時間内に送り届けることである. サブネットの構造はネットワークのノードに HOST が直列に結合されるか, 並列に結合されるかによって2つに分類される. ローカル・ネットワークとして特色のある米国ベル研究所のループ・ネットワークは直列系であるが, KUIPNET では実効連続転送速度を大きくとるために並列系を採用している. その他, KUIPNET と他のネットワークの比較については著者らの文献 6) を参照されたい.

本論文では, KUIPNET のうち主として HOST に関する側面を記述した. サブネットの部分については別稿での報告が予定されている. 一般に HOST とサブネットの分離は肝要であり, その意味からも本論文に述べる HOST に対する要請はサブネットによって影響されない一般性がある.

2. プロセス間通信方式とプロトコル

HOST の観点からいえば, コンピュータ・ネットワークとはネットワーク内の任意の HOST のプロセス (タスク) が, 任意に指定された他の HOST の任意のプロセスと互いに通信網を経て情報交換できる手段を提供するものといえる. プロセス間の通信を司さどるプログラムが各 HOST に用意されるが, これをネットワーク・コントロール・プログラム (NCP) という. NCP のプロセスに対する役割 (NCP-net) は **Fig. 2** のように, ちょうどサブネットの HOST に対する役割と同様である. 各 HOST 内の NCP は互いにコントロール・コマンドを交換し対話を行ないながら, プロセスの間の通信のための制御を行なう.

さて, プロセス間の通信の手順はネットワーク内で統一しなければならないが, この通信規約のことをプロトコルという. プロセス間通信のプロトコルを定めるにあたって2つの側面がある.

(1) 集中型プロトコルか, 分散型プロトコルか.

(2) 接続型プロトコルか, メッセージ単位伝送プロトコルか.

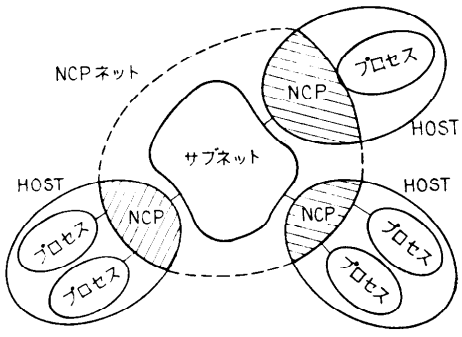

Fig. 2 Subnet and NCP-net

(1)について，集中型の場合サブネットのどこかに中央 NCP センタがあり，これが各 HOST の NCP と連繋して通信制御を行なう．各 HOST の NCP は簡単になるがネットワークの拡張性およびシステム障害に関して難点がある．分散型の場合には NCP の実体はサブネット内には存在せず，すべて HOST 内にあり，その長所短所は集中型の場合と逆になる．どの HOST の NCP も他のどの HOST の NCP とも直接通信が可能である．KUIPNET ではインハウス HOST間の優先順位指定に関する取り決めを除いては ARPANET と同様分散型である．

(2)について，接続型プロトコルでは，実際の情報転送に先立って双方のプロセス間で接続（リンク）を確立し，その後そのリンク上で何回も自由にメッセージを交換し，通信が完了するとそのリンクの解放を行なう．メッセージ単位伝送プロトコルの場合は，リンクはメッセージ単位ごとのもので1単位転送が完了すると自動的にリンクは解消する．これは単発メッセージが主体のプロセス間通信にむいている．KUIPNET の場合，高速大量のデータ伝送を扱うので前者の接続型プロトコル（ARPANET と同じ）が採用され，必要な場合にはこのプロトコルによって形成されるリンク上に優先順位を指定できる．

さて，プロセス間通信を実現するためにはまず自他プロセスの識別を行ない，リンクの確立を行なう必要がある．すなわち相手プロセスが通信可能な状態にあるかを調べ，当該プロセス間に論理的な通信路を確立する（プロセス間の同期という）．通信路の制御を簡単にするため1つのリンクは一方向のみの通信を扱う．送受両方を行なう場合には2本のリンクが必要である．リンクが確立すると，このリンク上で当該プロセス間でメッセージの転送が行なわれるがそのためには受信側にバッファ・スペースを割付けること，および利用可能なスペースの大きさを送信側に知らせる必要がある．これをメッセージの流れの制御という．通信が完了するとリンクは解放される．プロセス間通信に必要な上記の一連の手続きをユーザプロセス自身が行なうのは繁雑なので，HOST ごとに一括して上述の NCP に行なわせるのである．

3. HOST のネットワーク・コントロール・プログラム

プロセス間の通信実現のための論理的接続とメッセージの流れの制御は，HOST の OS の一部として常駐するネットワーク・コントロール・プログラム(NCP)が行なう．NCP はサブネットをその HOST の1つの周辺装置とみた入出力制御を行なうのではなく，論理的な多数のメッセージ転送路を扱い，また原則としてOS管理下の全てのプロセスから使用できる．

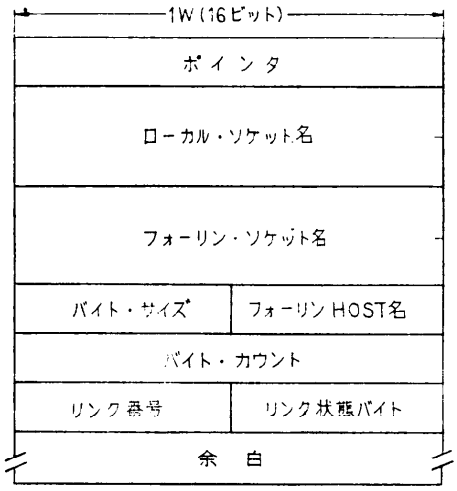

Fig. 3 Configuration of LCB

3.1 KUIPNET の NCP

KUIPNET で実装した NCP はプロセス間接続の状態を Link Control Block (LCB) に記録する．**Fig. 3** に1つの接続に対する LCB の構成を示す．接続は1つのプロセスの出力を他のプロセスの入力とするように結び，ソケット名（すなわち，ネットワークで共通に付けられたプロセスの名前）と HOST 名で規定される両端を1本のリンクで結ぶことである．接続が確立した後は送り手 HOST 名とリンク番号で一意的に区別される．リンク状態バイトは接続要求中，接続確立・切断，メッセージの到着待ちなどのフラグからなる．バイト・サイズ，バイト・カウントはメッセージの流れの制御に使われる．プロセス間の通信に先立つ NCP 間の打合せはネットワークに共通に定められたリンク（コントロール・リンク）を用い，NCP のコントロール・コマンドで行う．

Table 1 NCP system calls

l: local socket name
f: foreign socket name
d: data name

Init (l, f)	: 接続要求
Listen (l)	: 接続待ち
Send (l, d)	: 送　信
Receive (l, d)	: 受　信
Close (l)	: 接続拒否または解放

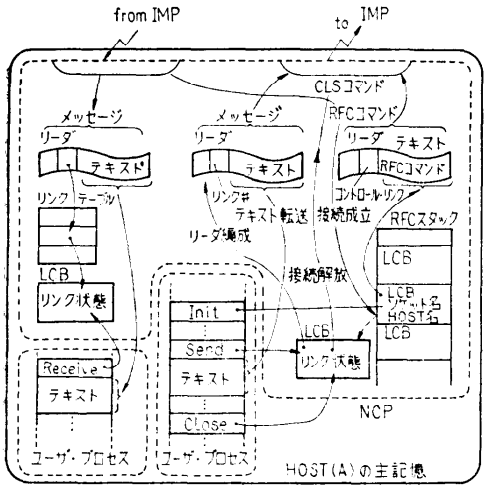

Fig. 4 NCP and user process

ユーザ・プロセスは他 HOST のユーザ・プロセスと通信するために **Table 1** に示す NCP システム・コールを出す. **Fig. 4** によって, HOST の主記憶常駐の NCP の動作を説明する. ある HOST(A) 上のユーザ・プロセスが接続要求 Init を出すものとし, その相手になるべき HOST (B) 上のユーザ・プロセスはそれより前に接続待ち Listen を出しているものとする. HOST (A) では, Init のパラメータである自他ソケット名が LCB に書き込まれる. NCP は LCB の情報からコントロール・コマンドとして RFC (Request For Connection) コマンドを組立て, これを一つのメッセージ中のテキストとする (メッセージにはテキストの前に宛先を示すリーダが付けられコントロール・リンク上で相手 HOST に送られる). この Init は NCP 内の RFC スタックに待ち行列の形で覚えられる. HOST (B) では接続待ち Listen が既に出されているが, NCP はこの待ち要求を RFC スタックに登録するのみで, Init のように特別な RFC コマンドを送ることはしない.

さて, HOST (A) の Init によって出された RFC は, HOST (B) の NCP に受けとられると, 自 NCP 内の RFC スタックと一致をとり, 一致がとれれば, リンク番号を割当て, 接続成立の意味の RFC コマンドを HOST (A) の NCP に返送する. こうして接続が確立する. この接続上で 2 HOST のユーザ・プロセスは Send, Receive のシステム・コールを行なって, 定まった方向にデータをテキストとするメッセージを送ることができる. メッセージが NCP に到着するとリーグ部のリンク番号によって, 使用中のリンクのテーブルを参照し, Recive を出しているユーザ・プロセスにテキスト部を転送する. プロセスが Close を出すと NCP はコントロール・コマンド CLS (close) を相手 NCP に送る. これによってリンクは解放され, その旨が双方プロセスに通知される.

NCP はバッファ約 8000 ビットを含め, MACC 及び MELCOM で 1 kW, NEAC で 4 k 字である.

3.2 プロセス間通信の例

3台の HOST のプロセスを結んで, 一つの協同作業を行なう例を考える. まず, 各 HOST に, NCP とモニタを NEAC のディスク・ファイルからロードし, オペレータの指示待ちになる. オペレータは MACC または MELCOM のコンソール・タイプライタから指示し, 全 HOST に NEAC のファイルから, ある協同作業を分担するユーザ・プロセスをロードする. この作業過程を **Fig. 5** の 1～6 の番号で示した. 1 の段階では, MACC のプロセスは **Table 2** (次頁参照) のごとく, システム・コール系列 Init, Send (作業名), Close を実行して, NEAC のプロセスへ作業名に対応するプログラムをディスクから取出し転送するように指示する. NEAC のプロセスは既に Listen を出してブロックされているが, 接続が確立されたときにブロックが解け, プログラムを転送する (2 の段階). 次に 3 で MELCOM のプロセスに作業

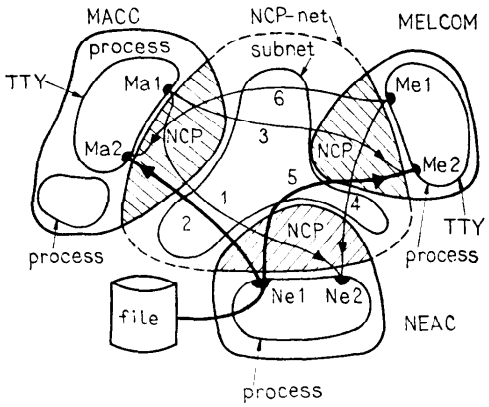

Fig. 5 An example of cooperating processes over three Hosts

Table 2 Sequence of system calls corresponding to Fig. 5.

D: cooperating-process name d: program to be transfered

stage	MACC	NEAC	MELCOM
		Listen(Ne 2)	Listen(Me 2)
1	Init(Ma 1, Ne 2)		
	Send(Ma 1, D)	Receive(Ne 2, D')	
	Close(Ma 1)		
2	Listen(Ma 2)	Init(Ne 1, Ma 2)	
	Receive(Ma 2, d')	Send(Ne 1, d)	
		Close(Ne 1)	
3	Init(Ma 1, Me 2)	Listen(Ne 2)	
	Send(Ma 1, D)		Receive(Me 2, D')
	Close(Ma 1)		
4	Listen(Ma 2)		Init(Me 1, Ne 2)
		Receive(Ne 2, D')	Send(Me 1, D)
			Close(Me 1)
5		Init(Ne 1, Me 2)	Listen(Me 2)
		Send(Ne 1, d)	Receive(Me 2, d')
		Close(Ne 1)	
6			Init(Me 1, Ma 2)
	Receive(Ma 2, D')		Send(Me 1, D)
			Close(Me 1)

名が送られ，4，5では1，2と同様の事が行われる．図中6の過程は全ての HOST に協同作業を分担するプロセスがロードされ終ったことの確認である．

4. KUIPNET の機能

情報処理研究用の KUIPNET は，インハウス・コンピュータ・ネットワークとして次のような機能をもつ．

(i) ミニコンと汎用計算機の機能共有

音声，画像などの情報処理研究に必要な各種入出力装置とソフトウェアをもつミニコンと，汎用計算機をネットワーク化することにより，各 HOST ごとに開発されているリソースが，互いに，即座に利用できる．MACC 7/F は Flying Spot Scanner (FSS), Color-Display, Graf-Pen などをもつ画像処理用の，また MELCOM 70 は AD/DA 変換器をもつ音声情報処理用のそれぞれ特色のある処理能力をもつミニコン知能端末である．ネットワーク化により，例えば，音声情報をカラー表示するなどのリソースの相互利用や，ミニコンと汎用計算機との機能分担による並列処理が可能となる．ミニコンの高速制御能力によりデータの入出力や前処理を行ないながら，NEAC 2200/200 でその抽出パラメータの複雑な処理やデータのファイル化が行われる．この際，TOSBAC 40 の高速フーリエ変換 (FFT) 機能も利用可能となり，数値計算はコンパイラ言語でプログラムを書き HITAC 8350 で処理させることもできる．今まで，断片的にせざるを得なかった一連の仕事が，ネットワーク化により即時にオンラインで可能となり，単なる効率の向上を越えた高次のマン・マシン・システムが期待できる．

(ii) リモート・ファイル

ミニコンでのネットワーク・モードの対話形処理や入出力制御に伴う CPU のアイドルタイムにはアセンブルなどの処理が可能であるが，アセンブルの対象となるプログラムの格納のため NEAC 2200/200 のファイルが利用できる．今日，安価で優れた機能をもつミニコンが多数開発され，一研究室に一台という時代になってきているが，ミニコンにその本体と比べて高価な周辺装置を多く備えることは，ミニコンの低価格性と矛盾する．このためには，周辺装置を完備した計算機とミニコンを結ぶことにより，その周辺装置をミニコンが利用できれば都合が良い．

(iii) 高速通信によるリアルタイム処理

ARPANET では IMP-IMP 間の伝送速度は 48 kbps, IMP-HOST 間のそれは 100 kbps であるが，KUIPNET では IMP-HOST 間の線路伝送速度は 1.6 Mbps である．さらに IMP に特別な優先順位指定回路を設けることにより，任意の一対の HOST 内のプロセス間で実効連続転送速度 300 kbps 以上のメッセージ転送が可能となっている．この機能により，種々のネットワークモードのジョブに混じってリアルタイム性の処理が必要なジョブ，例えば，NEAC 2200/200 のファイル内のディジタル音声データを MELCOM 70 の DA 変換器に送り，とぎれのない音声を合成することも可能である．

5. ミニコン HOST の OS

KUIPNET の諸目的を達成するにはミニコン HOST が重要な役目を果たす．すなわち，ネットワークの HOST と知能端末の両方の機能を所有する OS が必要となる．ここでは各ミニコンに開発したモニタと処理プログラムの概要を述べる．モニタ本体は **Fig. 6** (次頁参照)のような構成をとり，NCP と合わせて約 2 kW (1 W: 16 ビット，バッファ: 500 W) でインプリメントすることができた．

5.1 ミニコン用モニタの設計仕様

(i) マルチジョブとその優先権

HOST の独立性を守る意味で，ミニコン・モード・ジョブと他 HOST からのネットワーク・モード・ジョブが同時に存在し得ることが必要である．そこで，

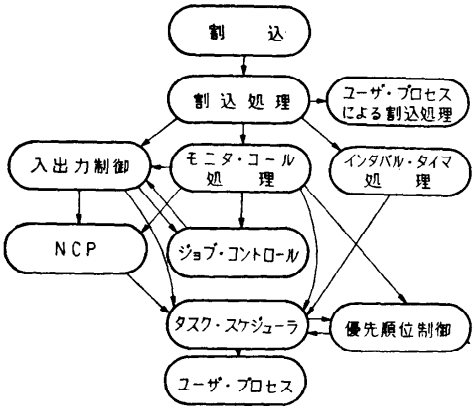

Fig. 6 Organization of monitor for minicomputer-HOST.

タイムスライス (図のインタバル・タイマによる) をもつマルチプログラミングによって任意個のマルチジョブを処理できるようにした. しかし, 主記憶が小容量であるため, 当面2ジョブに制限し, ジョブにはアドレスの最上位と最下位からメモリを割り当てている.

また, ジョブはモニタコールによってサブプロセスを生成することができる. そのため普通はプロセスは先着順に処理されるが, 1つのプロセスだけはモニタコールによって, 優先順位制御部の指示の下に他プロセスに優先して処理されるプロセスになることができる. この優先権を用いることにより, 多数プロセスが共存していても DA 変換による音声合成のようなリアルタイム処理が可能となる. つまり, リアルタイム的なネットワーク利用のために, サブネットに HOST 間通信の優先機能を持たせているのに対応して, HOST の OS でも同様の機能を付与させた.

(ii) 入出力制御

周辺装置の割当てや割込みを伴う入出力制御はモニタで管理しているが, 予め登録することにより, 割込みをユーザプロセスで処理することも可能にした. さらに割込みを伴わない入出力命令はプロセスが直接実行することにより, 入出力のモニタにおけるオーバヘッドが軽減され, 高速のリアルタイム処理が可能となった.

ミニコンの場合の IMP-HOST 通信用ハードウェアは DMA チャネルに接続した. この制御のための割込みはメッセージの送受の開始及び終了の4種類, 入出力命令は 10 個程度である. モニタと NCP のインタフェースは通信ハードウェアからの割込みとプロセスからのシステムコールの二つだけなので, 通常の入出力装置を新しく付加する際の手続きと同様であった.

(iii) 端末からのログイン

各ミニコンには情報処理ソフトウェアと会話形処理プログラムが開発されているので, ユーザは端末からログインできる方が便利である. そこで, TTY またはキャラクタ CRT からジョブの開始, 終了, 処理プログラムのロードと起動, リソースの占有状況の質問応答等を可能にした. この機能を拡張すれば, ネットワークによるリモート・ジョブ・エントリも可能となる.

5.2 会話形の処理プログラム

(i) アセンブラ

これまで, 大容量二次記憶がないため, 小容量のプログラムしか会話形でアセンブルすることができなかった. さらに TTY を使っていたため, リスト出力に長時間を要した. そこで, ネットワーク経由で NEAC 2200 の DISK, MT, LP, HSP などのリソースを利用することにし, ミニコンのアセンブラが NEAC のファイル・コントロール・プログラムと通信するためのプロトコルを定め, その機能をアセンブラにインプリメントした.

(ii) デバッギング・プログラム

KUIPNET のミニコンは, プログラムのエラーに対するハードウェア的防御機能が完全ではないので, ネットワーク化すると他 HOST まで侵害しかねない. モニタでのチェックも限界があるので, 新しくデバッギング・プログラムを開発した. これはデバッグの完了していないプログラムの実効アドレスや, 特権命令のチェックを行ないながらその命令を実行する. 従って, デバッグ中のプログラムがネットワークの利用や入出力を実際に行なえる他, トレースやスナップショットの機能も持つ.

6. むすび

情報処理研究のためにインハウス・コンピュータ・ネットワーク KUIPNET を構成した. これは典型的なコンピュータ・ネットワークの思想に基づきインハウス向きに構成されたもので, 広域ネットワークに加入可能なように考慮されている. 本論文ではこのシステムのうち主として HOST に関する側面を記述した.

現在, KUIPNET は第一期の開発が終了し, 試験

的に運用中である．実験例のネットワーク・モードのジョブとしては，MACC 7/F の Graf-Pen で入力した署名の画数，筆速をカラーディスプレイで表示すると共に画数の合計，所要時間を IMP 経由で NEAC 2200 に送り，対応する音声データが DISK ファイルから編集され，IMP を経て MELCOM 70 へ実効連続転送速度 200 kbps (20 kHz サンプリング，10 ビット）で転送され，DA 変換によって音声となる．この間，MACC において同時にミニコン・モードとして，エディテングが実行されているといったシステムの実験に成功した．

近く，KUIPNET の第2期として NEAC 2200 が4つのマルチジョブが走り，メモリプロテクション，科学演算機構のついたものにレベルアップされ，さらに大形 HOST コンピュータと 48 kbps の公社回線での接続も計画されているので，KUIPNET の機能が著しく拡大，実用化されるものと期待される．最後に，KUIPNET のハードウェア，ソフトウェアの作成に協力して下さった坂井研究室の方々に深く感謝いたします．

参 考 文 献

1) F. E. Heart et al: The interface message processor for the ARPA computer network, AFIPS, SJCC, Proceedings, p. 551.
2) 坂井利之他：インハウス・コンピュータ・ネットワーク KUIPNET について，信学会，電子計算機研究会資料，1973. 12
3) S. M. Ornstein et al: The Terminal IMP for the ARPA computer network, AFIPS, SJCC, 1972, p. 243.
4) T. Sakai et al: Inhouse Computer Network for Information Processing and some Applications, preprint for the seminar on Computer Asisted Chemical Research Design, July, 1973.
5) S. D. Crocker et al: Function-oriented protocols for the ARPA Computer Network, AFIPS, SJCC, 1972, p. 271.
6) 坂井利之他：コンピュータ・ネットワーク，計測と制御，Vol. 12, No. 11, 1973, p. 863.
7) A. Mckenzie et al: Host/Host protocol for the ARPA Network, NIC 8246, Jan. 1972.
8) 坂井利之他：インハウス・コンピュータ・ネットワーク KUIPNET と HOST ソフトウェア，第 14 回情報処理学会全国大会(1973).

(昭和 49 年 4 月 16 日受付)
(昭和 49 年 8 月 30 日再受付)