



卒業研究報告書

平成 4 年度

研 究 課 題	パソコン LAN に関する研究
------------	-----------------

指 導 教 官	成 田 紘一教授
------------------	----------

提 出 者 氏 名	桐 島 俊之
--------------	--------

平成 5 年 2 月 1 日 提出

(奈良工業高等専門学校 電気工学科)

【 要 約 】

本研究は、電話回線・イーサネット回線の2種類の回線にパソコンを接続しメールやファイル転送等の通信機能を手軽に利用することが出来る通信ソフトウェアの開発を行うものである。

本研究で開発したソフトウェアは、電話回線を利用してのパソコン通信に対応する一方、イーサネットを利用してUNIXのTELNETサーバに接続する機能を持つ。

また、初心者も利用できる通信システムにするために、ほとんどの操作をマウス対応にした。各種の機能の選択は、マルチウィンドウ方式を取り入れメニュー形式とした。それにより、従来のものよりも操作が簡単で扱い易い通信システムとなった。

本論文の第2章では使用するイーサネットハードウェアについて述べ、つづいて第3章ではソフトウェアの設計、第4章ではソフトウェアの作成について述べた。第5章では実行結果、第6章では、それらの検討を行った。

【 目 次 】

第 1 章	ま え が き	1
第 2 章	ハ ー ド ウ ェ ア の 概 要	3
2 - 1	L A N コ ン ト ロ ー ラ M B 8 6 9 5 0 B	4
2 - 2	C S M A / C D 方 式	7
第 3 章	ソ フ ト ウ ェ ア の 設 計	8
3 - 1	基 本 設 計	8
3 - 2	モ デ ム 通 信 機 能 の 設 計	8
3 - 3	M S - D O S コ マ ン ド 機 能 の 設 計	1 2
3 - 4	接 続 先 登 録 機 能 の 設 計	1 2
3 - 5	便 箋 機 能 の 設 計	1 5
3 - 6	ヒ ス ト リ 機 能 の 設 計	1 5
3 - 7	L A N 接 続 機 能 の 設 計	1 6
3 - 7 - 1	通 信 プ ロ ト コ ル T C P / I P	1 6
3 - 7 - 2	デ ー タ 入 出 力 プ ロ グ ラ ム の 設 計	1 7
3 - 8	ア イ コ ン エ デ ィ タ の 設 計	2 1
3 - 9	ア イ コ ン メ ニ ュ ー の 設 計	2 2
3 - 1 0	飛 火 野 用 ウ ィ ン ド ウ 関 数 の 設 計	2 3

第 4 章	ソフトウェアの作成	2 5
4 - 1	L A N 端 末 接 続 機 能 の 作 成	2 5
4 - 2	モ デ ム 通 信 機 能 の 作 成	2 7
4 - 3	M S - D O S コ マ ン ド 機 能 の 作 成	2 9
4 - 4	接 続 先 登 録 機 能 の 作 成	2 9
4 - 5	便 箋 機 能 の 作 成	3 3
4 - 6	ヒ ス ト リ 機 能 の 作 成	3 5
4 - 7	ア イ コ ン エ デ ィ タ の 作 成	3 7
4 - 8	ア イ コ ン メ ニ ュ ー の 作 成	3 8
4 - 9	飛 火 野 用 ウ ィ ン ド ウ 関 数 の 作 成	3 8
第 5 章	プ ロ グ ラ ム の 実 行	4 0
5 - 1	モ デ ム 通 信 機 能 の 実 行	4 0
5 - 2	L A N 端 末 接 続 機 能 の 実 行	4 1
5 - 3	M S - D O S コ マ ン ド 機 能 の 実 行	4 2
5 - 4	接 続 先 登 録 機 能 の 実 行	4 2
5 - 5	便 箋 機 能 の 実 行	4 3
5 - 6	ヒ ス ト リ 機 能 の 実 行	4 4
5 - 7	ア イ コ ン エ デ ィ タ の 実 行	4 5
5 - 8	ア イ コ ン メ ニ ュ ー 機 能 の 実 行	4 6
第 6 章	実 行 結 果 と 検 討	4 7

6 - 1	実行結果	4 7
6 - 2	実行結果の検討	4 7
第 7 章	あ と が き	5 0

◆ 謝 辞

◆ 参 考 文 献

◆ 付 録

第 1 章 ま え が き

近年、新しいメディアとしてのパソコン通信が広範囲に利用されてきている。パソコン通信を行う際には、電話回線を利用してホストコンピュータに接続しなくてはならない。このときに、必要となるのが通信ソフトである。現在の主要パソコン通信ソフトの例とそのマンマシンインタフェース環境の採用状況を表 1 に示す。

表 1 主要パソコン通信ソフトの G U I 環境の採用状況

通 信 ソ フ ト 名	マウス対応	アイコン対応	ウィンドウ対応
W T E R M	×	×	○
M y T a l k	×	×	○
C C T 9 8	×	×	△

表 1 よりわかるように操作の際にキーボードの使用を前提としているものが多く、初心者がパソコン通信を始めるのにふさわしいものとなっているとは言い難い。

そこで本研究では、操作が簡単なマウス、一目で何の機能かわかるアイコン、画面上に整然と必要な情報を表示するマルチウィンドウを取り入れ、可能な限りマン・マシンインターフェイスの優れた通信ソフトを開発することにした。

一方で、パソコン通信以外でも企業や学校においてパソコン通信と同様な形態で高速なLANが活用されるようになってきている。そこで、本研究では上記で述べたパソコン通信用としてモデムを使用しての従来の回線をサポートするとともに、このLANにも対応できる通信ソフトを開発することにした。LANとしてはイーサネットを対象として研究した。

これらの研究によりモデム、イーサネットを使用しての通信が可能な通信ソフトウェアを作成することができた。

注) LAN端末接続機能の作成の際に本研究では、株式会社アスキーのInetBIOSを使用した。

第2章 ハードウェアの概要

本システムで使用するネットワークハードウェアは、モデムとイーサネットカードである。

モデムは、オムロンのインテリジェントモデムMD2400Bを使用した。イーサネットカードは富士通LANコントローラMB86950Bを採用した。このイーサネットカードはIEEE802.3（アメリカ電気電子技術者協会802委員会によって標準化された規格）に準拠している。本研究では、UNIXが使用しているアクセス方式がCSMA/CD方式である10BASE5（Ethernet）を使用する。本システムの構成図を図2-1に示す。

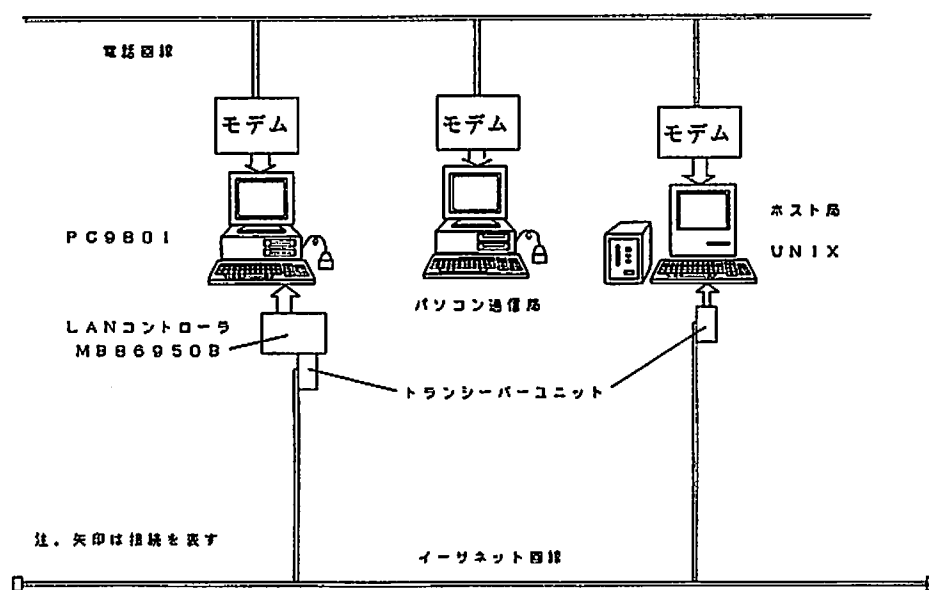


図2-1 本システムのブロック構成図

2 - 1 L A N コントローラ M B 8 6 9 5 0 B

本研究で使用する L A N コントローラは、バッファマネージャ、D R A M コントローラ、データリンクコントローラ、S t a r L A N 用マンチェスタエンコード・デコード、システムインターフェイスの計 5 種類の機能ブロックを 1 チップに集積している。このため、インターフェイス回路、ソフトウェアのオーバーヘッドを抑えることが出来る。

ホストシステムとのインターフェイスは 8 bit、1 6 bit どちらのシステムバスともインターフェイスが可能である。また、ホストシステムとのデータ転送は送信、受信データともバッファメモリポートを通して、ホストシステム側より I / O 命令によるプログラム転送か D M A 転送により行う。

送受信データは最大 6 4 K B の専用バッファメモリにストアされバッファマネージャによりホストシステムからのアクセスと、データリンクコントローラからのアクセスのアービトレーション、バッファ領域の管理が行われる。このため、データ管理のためのローカル C P U を必要とせず、直接ホストシステムバスに接続することが可能である。

図 2 - 2 に L A N コントローラのブロック図を示す。

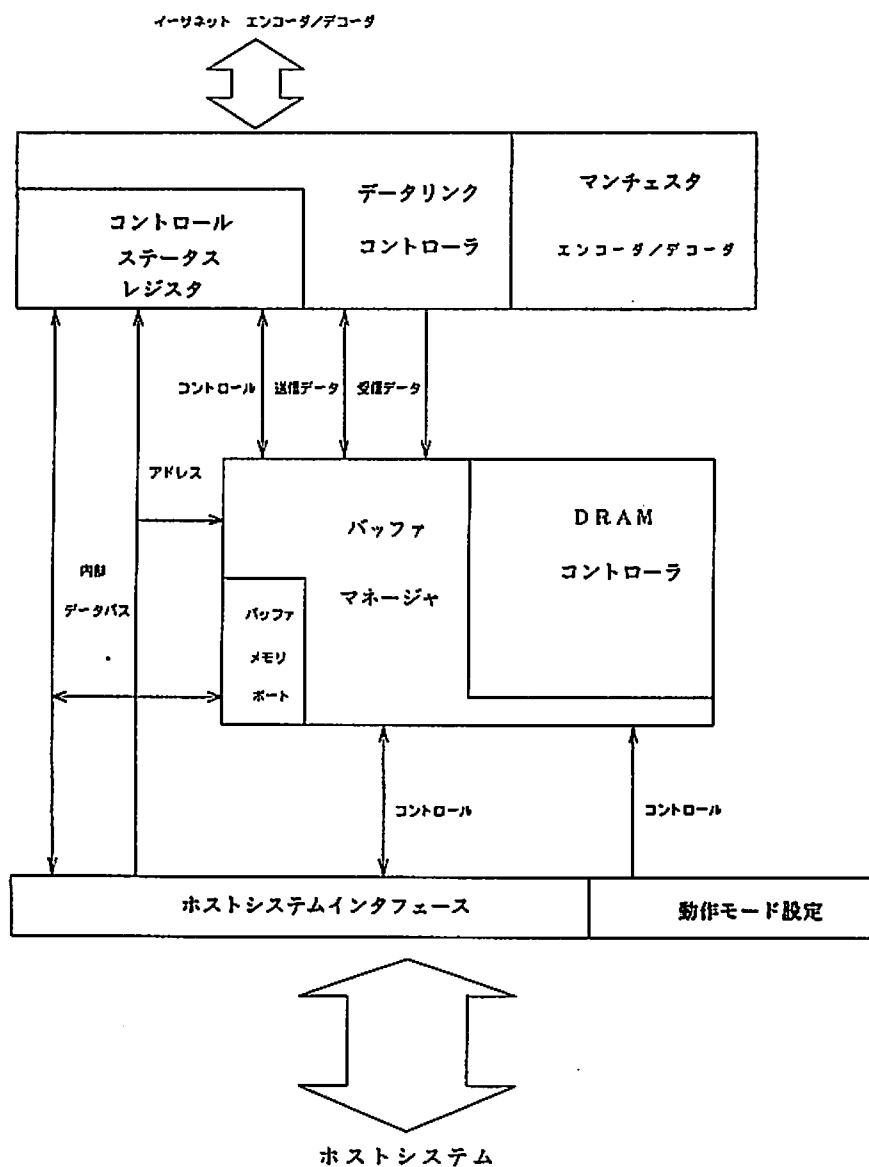


図 2 - 2 L A N コントローラのブロック図

ここで L A N コントローラの各部の働きについて説明する。

・ バッファ マネージャ

バッファ マネージャはホストシステムとバッファメモリ、データリンクコントローラとバッファメモリ間のデータ転送を制御す

る。また、バッファマネージャはアービトレーション機能を持っておりデータリンクコントローラを通して、LANネットワークとパケット送受信動作中でも、ホストシステムとバッファメモリ間のデータ転送を行うことができる。

- ・ D R A M コントローラ

バッファメモリ用 D R A M に対し、タイミング信号の発生、リフレッシュサイクルを実行する。

- ・ データリンクコントローラ

データリンクコントローラは、I E E E 8 0 2 . 3 で規定されたデータリンク層の機能を実行する。

- ・ S t a r L A N マンチェスタエンコード・デコード

本システムではバス型 L A N を採用するためこの機能は、使用しない。

- ・ システムインターフェイス

システムインターフェイス部は、ホストシステムバスとのインターフェイスを行うためのデータバスバッファ及びインターフェイス用制御信号の入出力を行う。データバスは 8 bit、1 6 bit 切り替えが可能である。データ転送レートは、最大 6 . 6 Mbyte/sec である。

2 - 2 C S M A / C D 方式

主にバス型とスター型のLANに採用されている方式で、送信を行いたい端末は、伝送路に搬送波（Carrier）が出ていないかを調べる。伝送路が使用されていなければ搬送波は、出でらず端末はデータの伝送を行うことができる。

C S M A / C D 方式のLANでは、2台以上の端末が同時に伝送路の空き状態を検知してデータ伝送を行うことがある。この時に伝送路でデータの衝突（Collision）が発生する。そこで、各端末は、常時衝突検出を行い、衝突を検出するとデータをすべて無効にし、ランダムな時間後に再送信を行う。C S M A / C D 方式では、送信希望の端末が多いと衝突が多発して伝送効率が著しく悪くなる欠点を有している。しかし、この方式は機能が比較的簡単で、低価格でLANネットワークを実現することが可能である。

第3章 ソフトウェアの設計

本研究で開発する通信ソフトの設計を以下に行う。

3-1 基本設計

これまでの市販やPDSの通信ソフトを参考にして、以下に本研究で開発するプログラムの主な機能を考えた。

1. LAN 端末接続機能
2. モデム通信機能
3. 便箋機能（簡易ワープロ）
4. 接続先登録機能
5. MS-DOS コマンド機能
6. ヒストリ機能（履歴）

またこれらの機能は、相互に使用することができるように、アイコンメニューによりアクセスできるように設計する。また、マウス対応のマルチウィンドウを取り入れることで、一層のマン・マシンインターフェイスの充実を図る。

3-2 モデム通信機能（LAN 端末接続機能も同じ仕様）

本機能には以下に示すサブ機能を持たせることとした。

- ・ 便箋送信機能

通信時に便箋に書かれている内容を相手に送信する。

- ・ 1行送信機能

通信時に相手に短い（１行分）データを送信する。

登録したデータはファイルに保存しておく。

・ファイル転送機能

ここでは以下に示す機能を持たせる。

1. 通信プロトコルの設定
2. 転送方向の設定
3. １行送信字数の設定
4. 送信後待ち時間の設定
5. ラインフィード処理の設定
6. 処理ファイル名の登録

通信プロトコルに Y M O D E M、K E R M I T を選

択した場合マルチファイル転送を可能にする。

・回線切断機能

接続していた回線を切断する。

・諸機能設定

通信状態で必要な以下の機能を持たせることとし、

これらの機能をフラグとしてセットし、通信状態でも

動作できるようにした。

1. ハードコピー
2. タイマー表示

3. 制御コード表示

4. PC9801 / VT100

・予約コマンド機能

接続先に応じてコマンドを登録し、いつでもそのコマンドを送信することが出来るようにする。（各コマンドには注釈を入れることも可能）

・ブレイク信号送信機能

ブレイク信号を送出する。

図3-1にLAN端末接続機能のフローチャート、図3-2にモデム通信機能のフローチャートを示す。

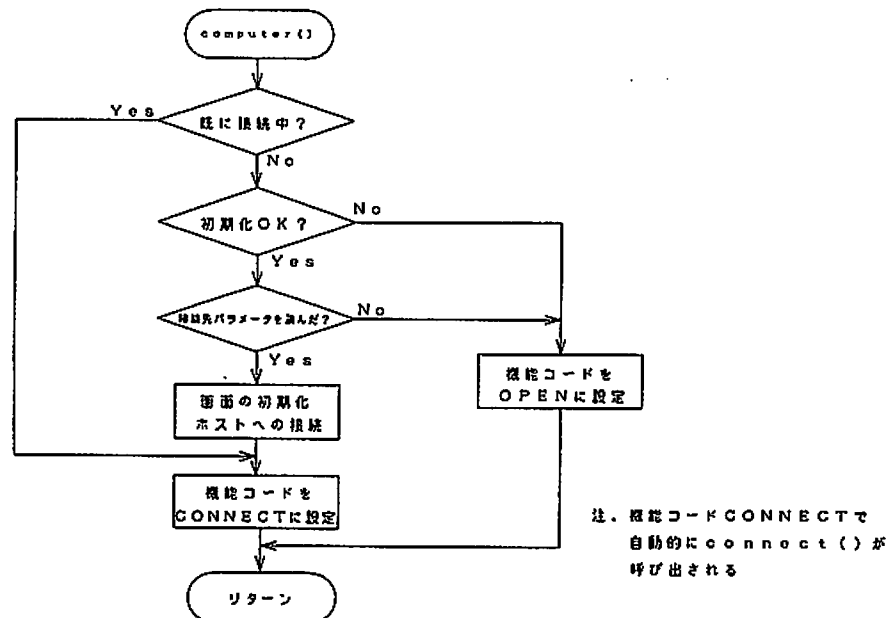


図3-1 LAN端末接続機能のフローチャート

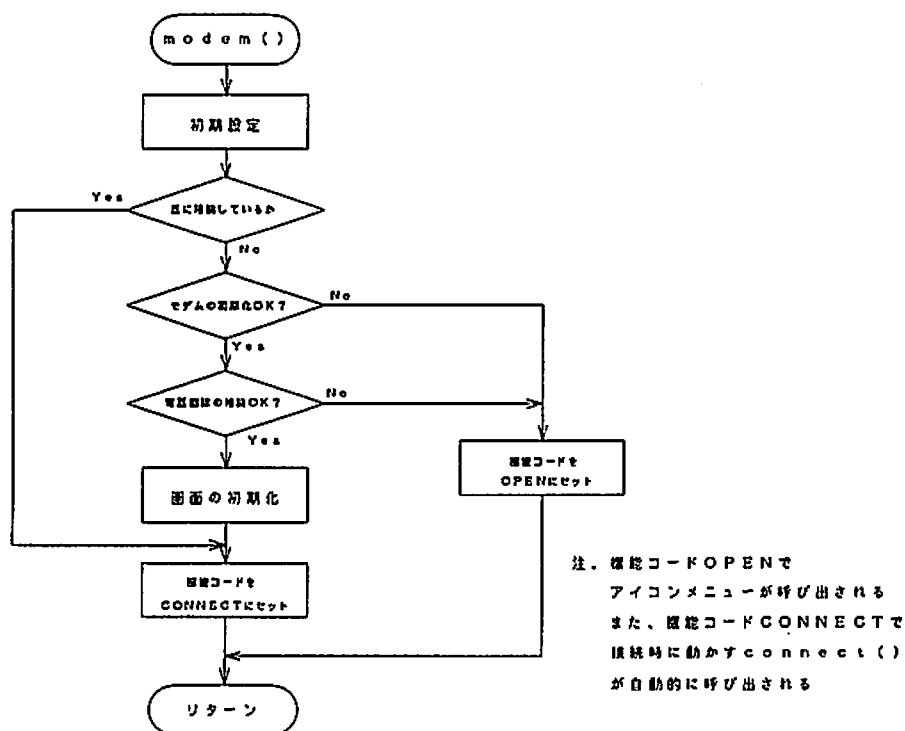


図 3 - 2 モデム通信機能のフローチャート

通信時にLAN接続の場合でもモデム通信の際のサブルーチンを
フラグ切り替えの手法で利用することとする。接続時に動作するプ
ログラムconnect()までの流れ図(図3-3)とその動作プログラ
ムのフローチャートを図3-4に示す。

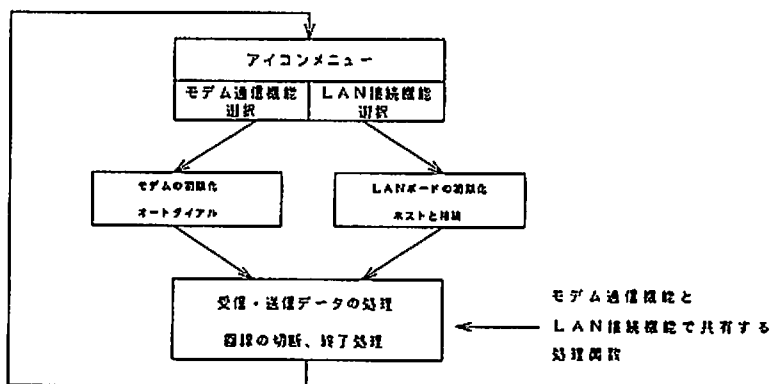


図 3 - 3 通信機能使用時の処理行程

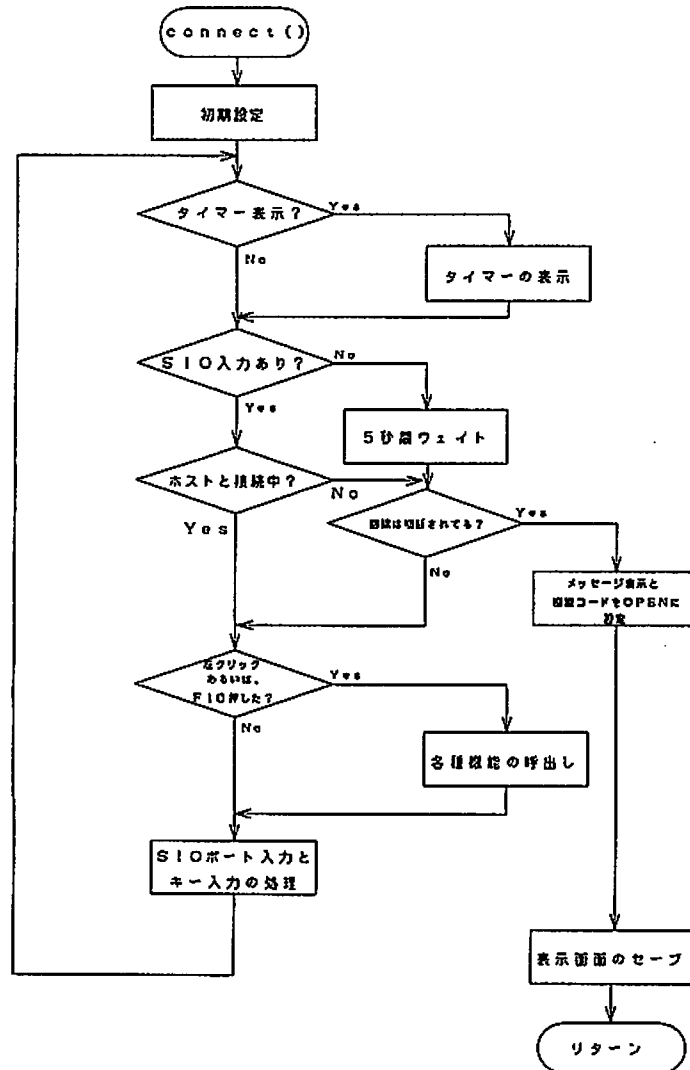


図 3 - 4 通信時のフローチャート

3 - 3 MS-DOS コマンド機能

この機能はMS-DOS命令の登録と実行を行う。空コマンドを入力することによりコマンドライン上でのコマンド実行を可能とする。

図3-5にフローチャートを示す。

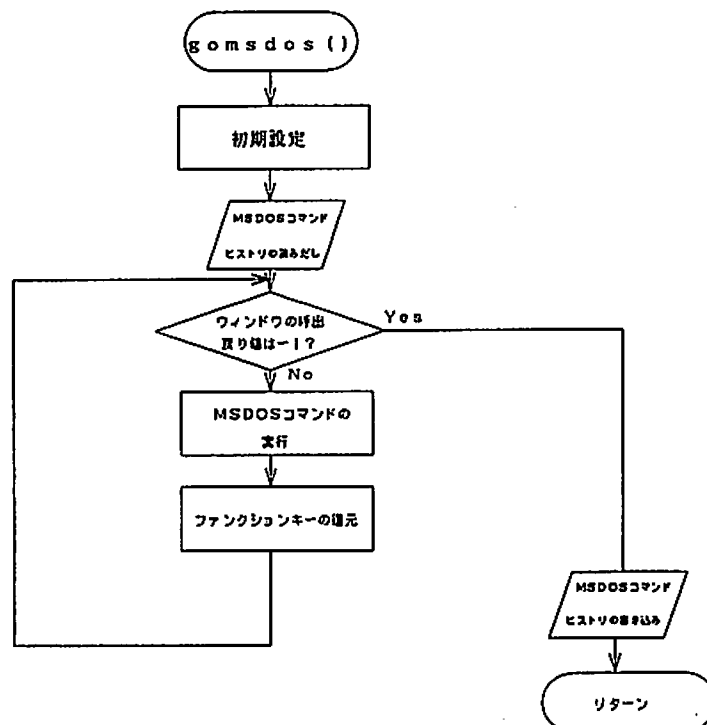


図3-5 MS-DOS コマンド機能のフローチャート

3 - 4 接続先登録機能

このモードでは、通信に必要なパラメータの設定を行う。以下に示す設定項目を設ける。

- ・ N E T 名 登 録
- ・ 電 話 番 号 登 録 (イ ー サ ネ ッ ト ア ド レ ス の 登 録)
- ・ 使 用 回 線 の 設 定
- ・ ボ ー レ ー ト の 設 定
- ・ M N P の 選 択
- ・ 使 用 チ ャ ネ ル の 設 定
- ・ デ ー タ 長 の 設 定
- ・ パ リ テ ィ ー の 選 択
- ・ 通 信 方 式 の 選 択
- ・ ス ト ッ プ ビ ッ ト の 設 定
- ・ X o n / X o f f の 選 択
- ・ デ ー タ コ ー ド の 選 択
- ・ C R 受 信 処 理 時 の 設 定
- ・ C R 送 信 処 理 時 の 設 定
- ・ L F 受 信 時 の 設 定
- ・ L F 送 信 時 の 設 定
- ・ 話 中 待 ち 時 間 の 設 定
- ・ 再 ダ イ ア ル 数 の 設 定

大半の設定項目は、選択式にし容易に設定出来るようにする。

登録したデータを消す際のために登録抹消モードを設ける。

図 3 - 6 にフローチャートを示す。

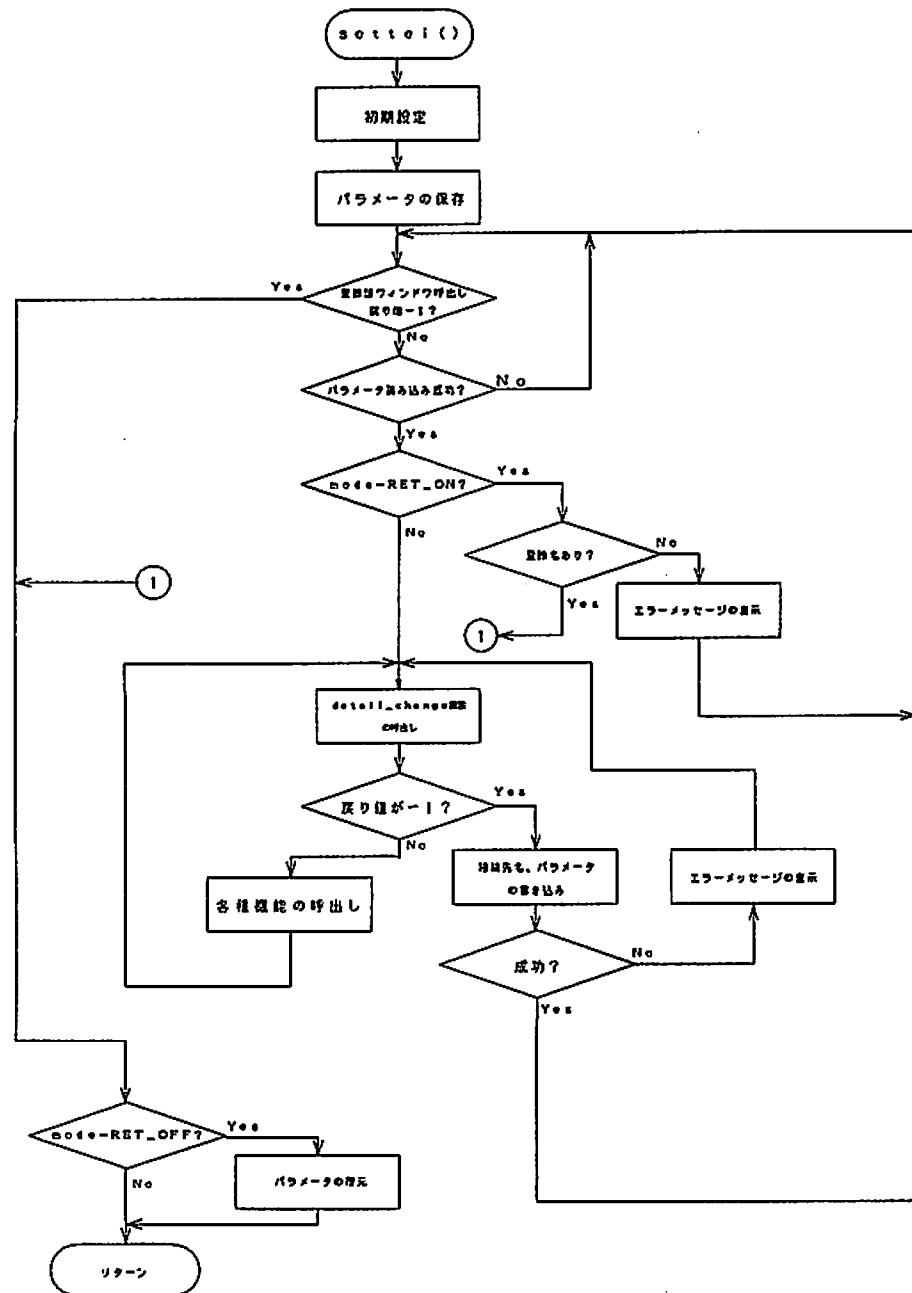


図 3 - 6 接続先登録機能のフローチャート

3 - 5 便箋機能（簡易ワープロ）

このモードでは、以下に示す機能を持たせる。

- ・ ファイルの読みだし
- ・ ファイルの書き込み
- ・ 便箋白紙（エディット内容の初期化）
- ・ 文字列検索
- ・ 範囲削除、範囲複写
- ・ 表示幅の設定

図 3 - 7 にフローチャートを示す。

3 - 6 ヒストリ機能

このモードでは、通信内容の閲覧、編集等の作業をする機能を持たせる。

- ・ 便箋複写機能（エディタに通信内容を転送）
- ・ ファイル記録機能（通信内容を保存）
- ・ プリント機能（プリンターに通信内容を出力）
- ・ 文字検索機能（指定文字列の検索）
- ・ 範囲削除

図 3 - 8 にフローチャートを示す。

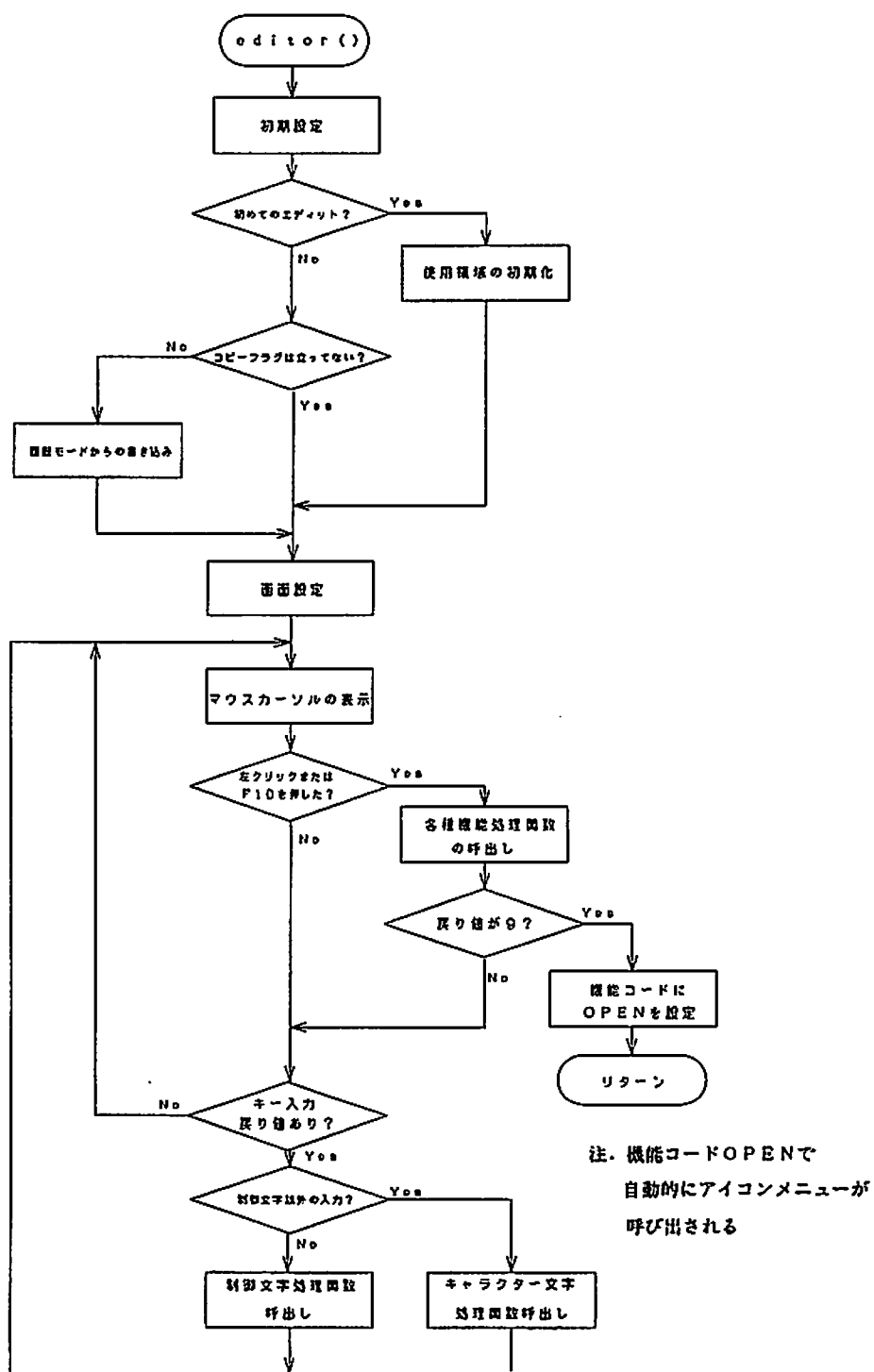


図 3 - 7 便箋機能のフローチャート

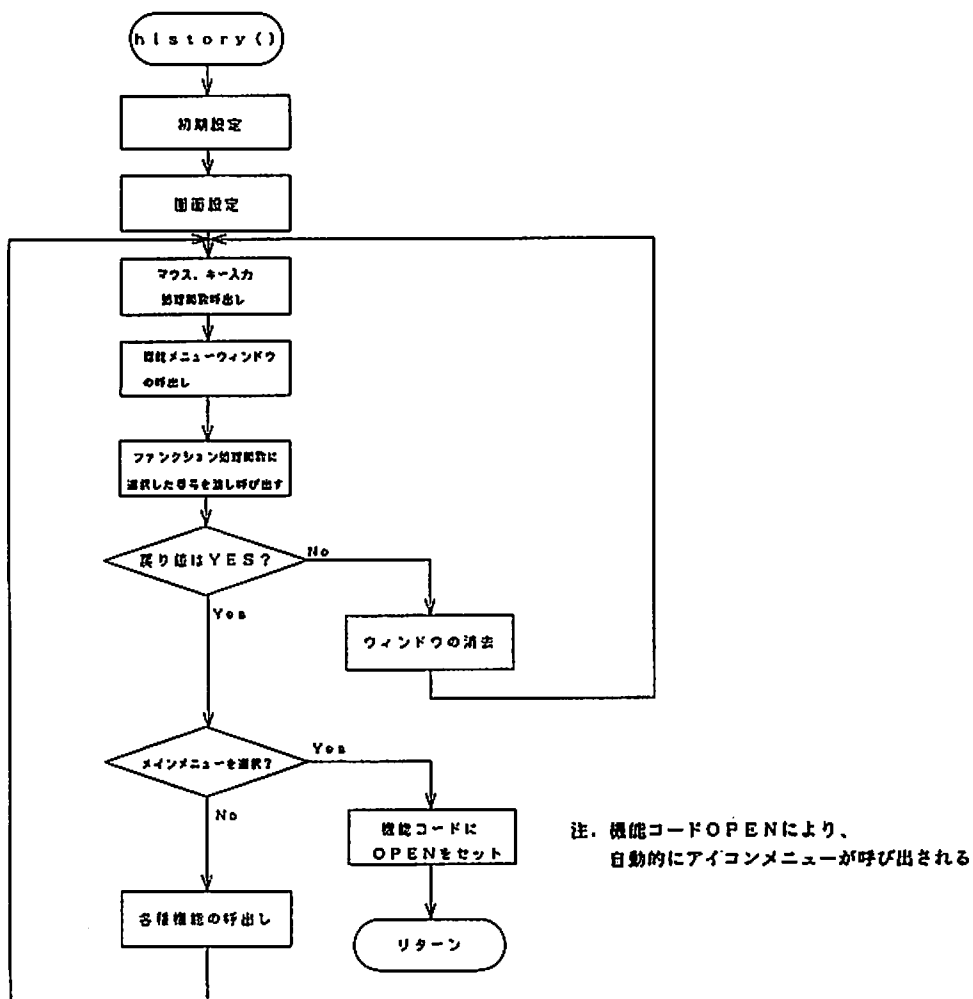


図 3 - 8 ヒストリ機能のフローチャート

3 - 7 LAN 接続機能の設計

3 - 7 - 1 通信プロトコル TCP / IP

本機能では U N I X を接続の対象としているのでイーサネットタイプの T C P / I P パケットフォーマットを使用する。しかし、使用する L A N コントローラは I E E E 8 0 2 . 3 で規定されたパケットに対応している。これら 2 つのパケットフォーマットの違いは、送受信の際に双方のパケットフォーマットを合致させることで無視することができる。

図 3 - 9 にイーサネットタイプのパケットフォーマットを、図 3 - 1 0 に I E E E 8 0 2 . 3 のパケットフォーマットを示す。

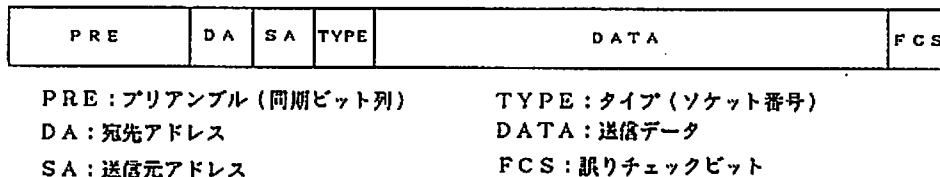


図 3 - 9 イーサネットタイプのパケットフォーマット

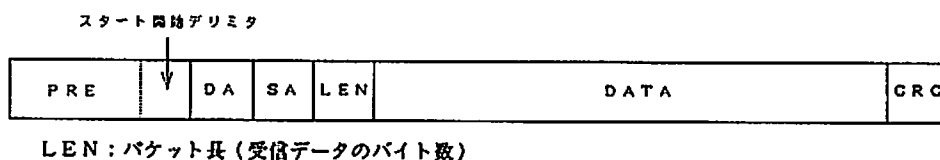


図 3 - 1 0 I E E E 8 0 2 . 3 のパケットフォーマット

3 - 7 - 2 データ入出力プログラムの設計

(1) LANコントローラ使用時のデータ処理プロセス

フレームの受信の際には、LANコントローラが送信されてくるフレームの宛先等を判断してそのデータをバッファRAMに記憶する、その際にハードウェア割り込みが発生し本システムにフレームの到着を知らせる仕組みとなっている。図3-11にフレーム受信の際のフローチャートを示す。

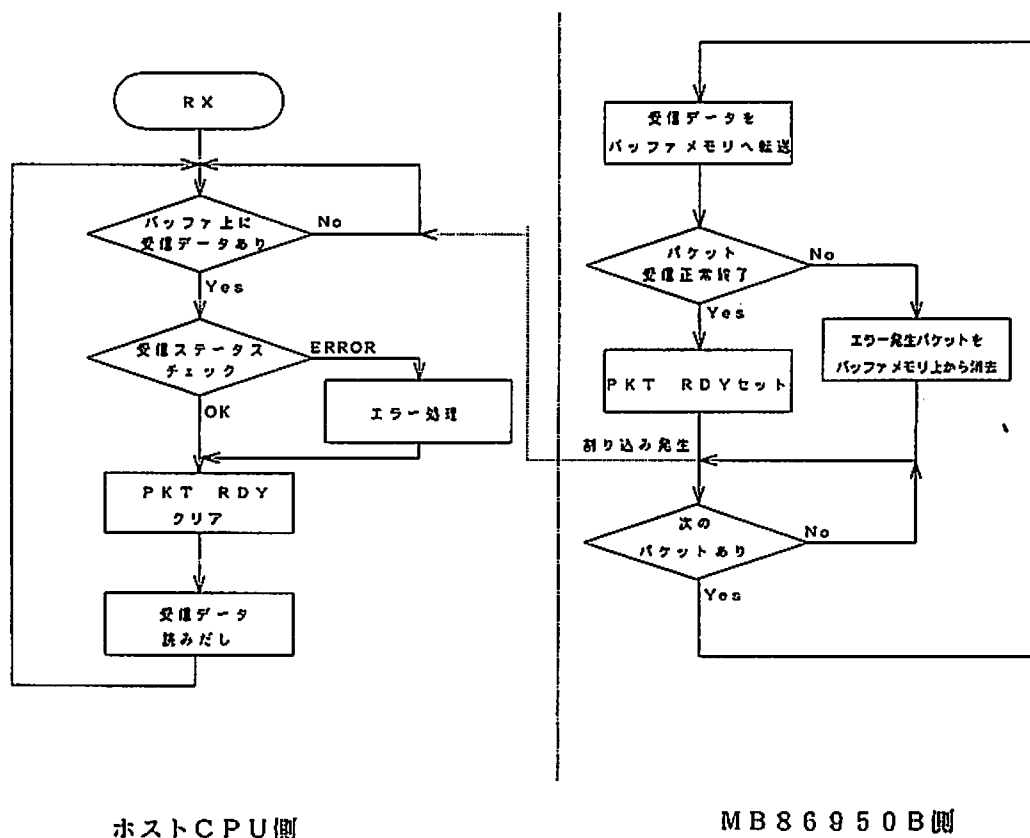


図 3 - 1 1 フレーム受信時のフローチャート

また、送信の際には、LANコントローラ内にあるバッファRAM内の送信バッファ領域にデータを書き込み送信フラグを立てることでイーサネットに送信される。図3-12に送信時のフローチャートを示す。

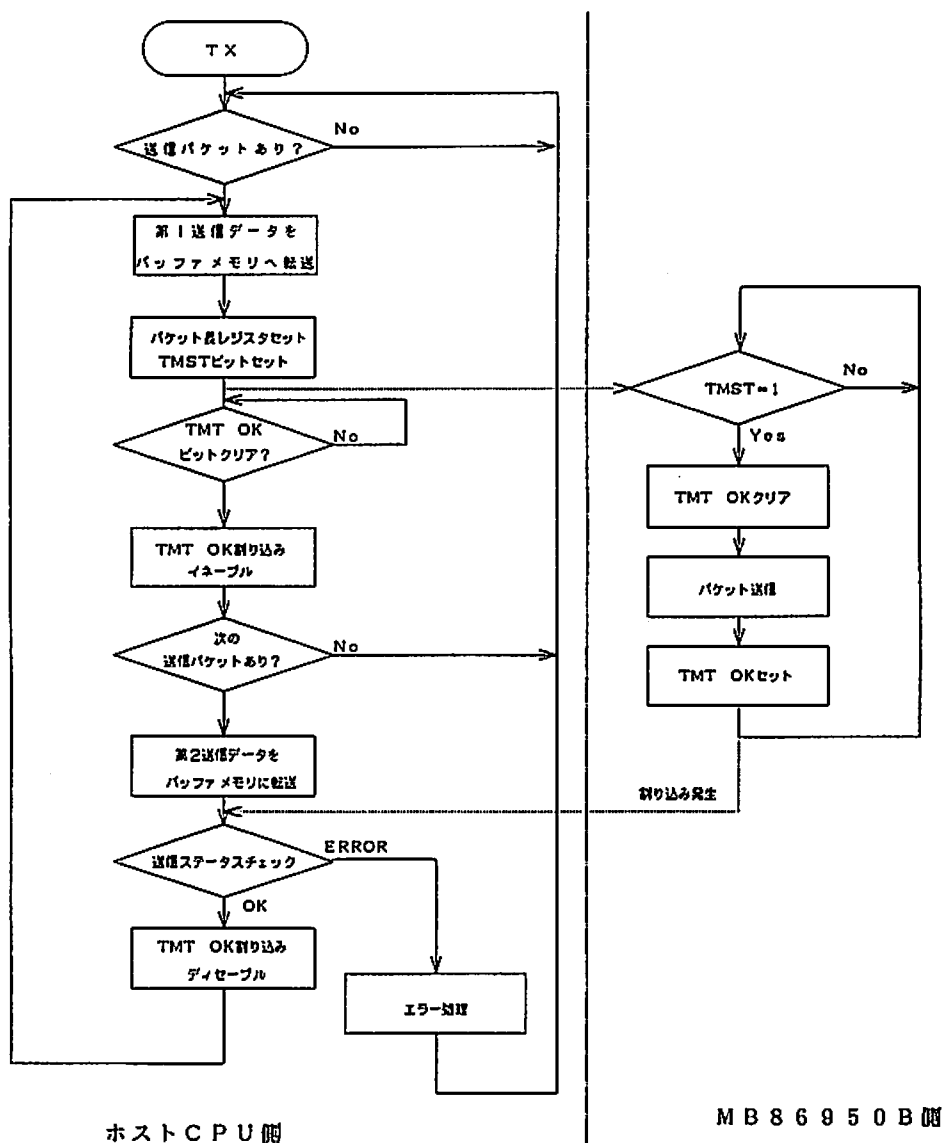


図3-12 フレーム送信時のフローチャート

(2) モデム通信の際のデータ処理プロセス

図3-13にモデム通信におけるデータの流れを示したブロック図を示す。

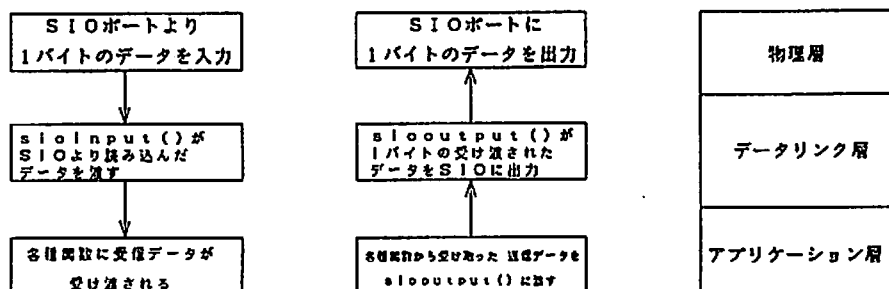


図3-13 モデム通信時のデータの流れ

(3) LAN端末接続時のデータ処理プロセス

モデム通信機能で使用する関数と同様の関数を作成することを選択するために以下に示す手順でLAN端末接続機能を作成する。

- ・ LANコントローラ内のバッファRAMと受信バッファを納めるメインメモリの領域を管理するためにバッファマネージャを作成する。(LANコントローラ内のバッファマネージャとは別)
- ・ 送信の際、まず、モデム通信の際と同様にSIO出力用関数にデータを渡す。そのSIO出力用関数内でLANコントローラに出力するデータかどうかを判別する。LANコントローラに送信するデータであるならばバッファマネージャにそのデータを渡し、LANコントローラ内のバッファRAMにデータを

セットさせ送信させる。

- 受信の際、まずSIO入力用関数が呼び出される。SIO入力用関数内で、LAN接続時であるかどうかを判断し、そうであるときはバッファマネージャより受信データを受け取りそのデータを返す。

図3-14にLAN端末接続時のデータの流れを示したブロック図を示す。

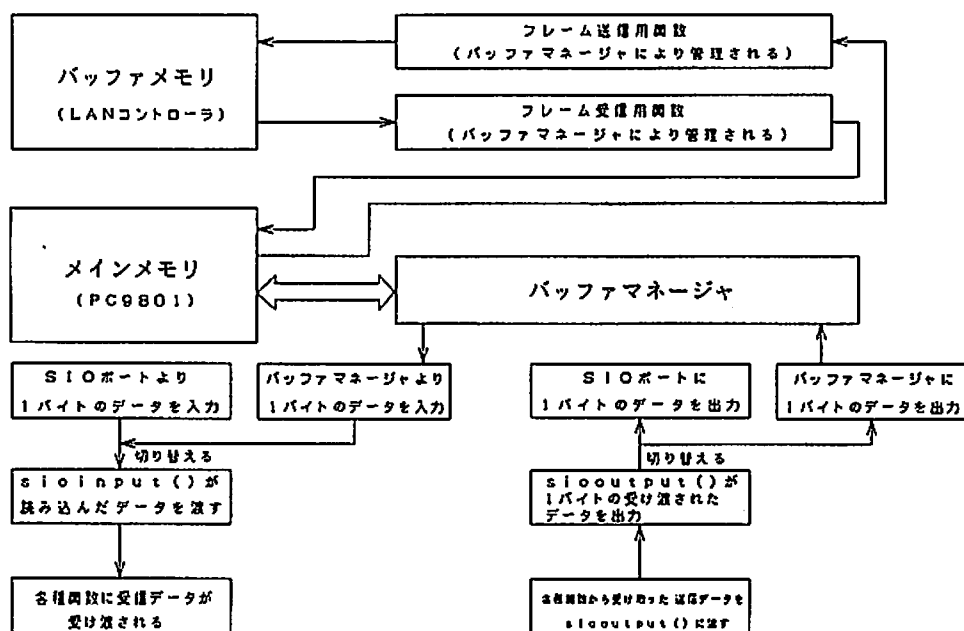


図3-14 LAN端末接続時のデータの流れ

3 - 8 アイコンエディタの設計

見やすいアイコンを短時間で作成しデザインの変更が容易に行えるようにアイコンエディタを作成する。

次に本システムで作成するアイコンエディタの仕様を示す。

- ・ 最大 30 枚のアイコンを 1 画面上で作成可能
- ・ エディット中のアイコンの任意の色を別の色に変えることが可能なパレット機能
- ・ エディット可能なアイコンの大きさ 40 x 40 ドット

アイコンエディタの画面構成図を図 3 - 15、フローチャートを

図 3 - 16 に示す。

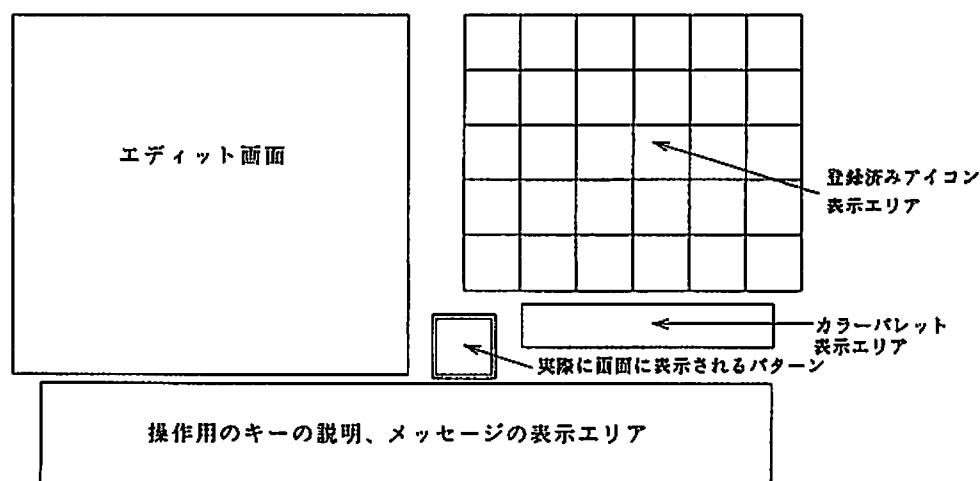


図 3 - 15 アイコンエディタの画面構成

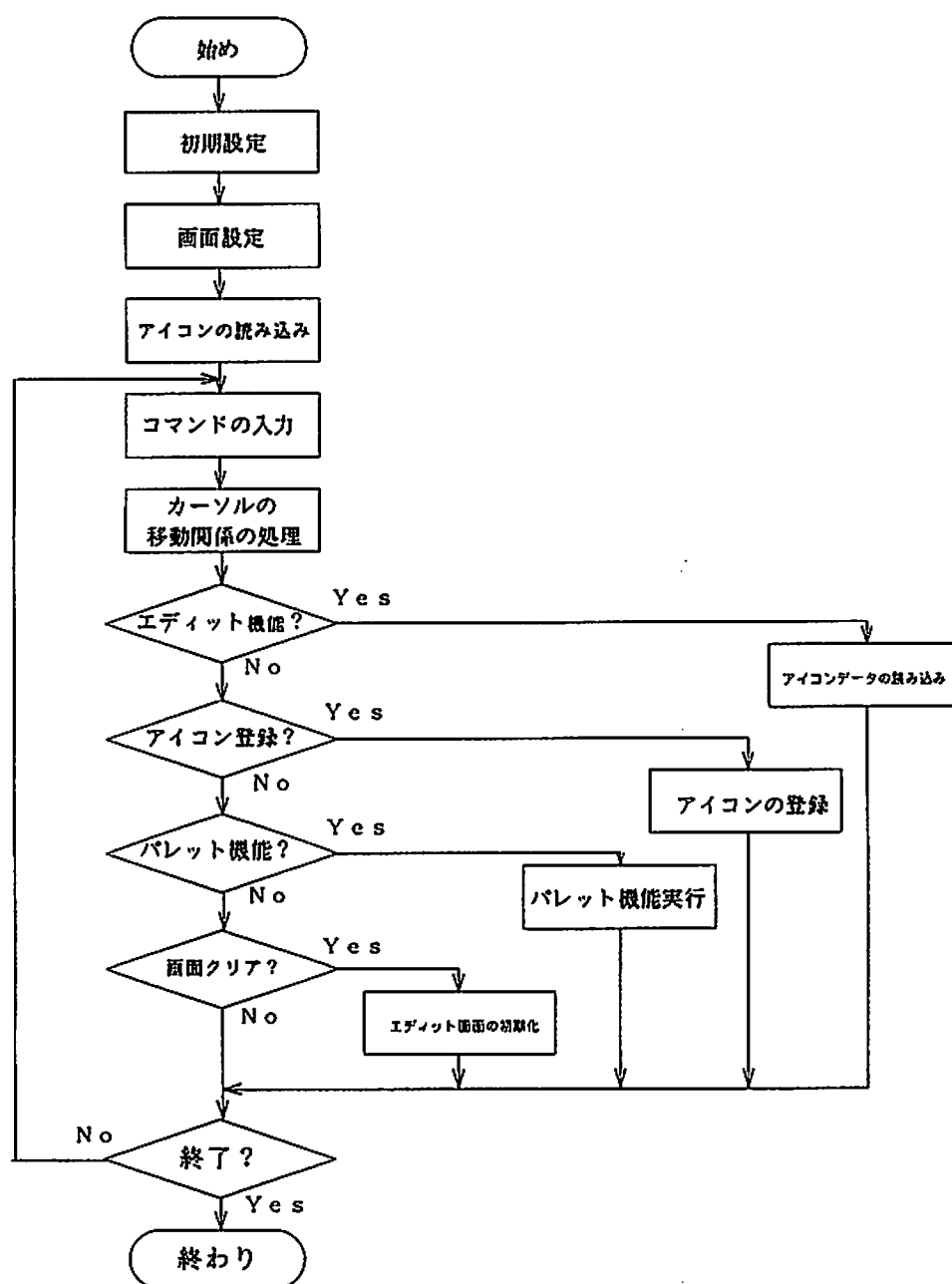


図 3 - 1 6 アイコンエディタのフローチャート

3 - 9 アイコンメニューの設計

本システムでは、アイコンメニューを介して各種機能（モデム通信機能、接続先登録機能、MS-DOSコマンド機能、便箋機能、LAN端末接続機能、ヒストリ機能）をアクセスするようにする。

現在、アイコンメニュー関数により呼び出される機能は、計6種類であるが将来、新たに機能を容易に増設することができるように、上記6種類の機能の他に2つ分、空アイコンとしてサポートしておく。

図3-17にアイコンメニューを介しての各機能の遷移図を、図3-18にアイコンメニュー関数のフローチャートを示す。

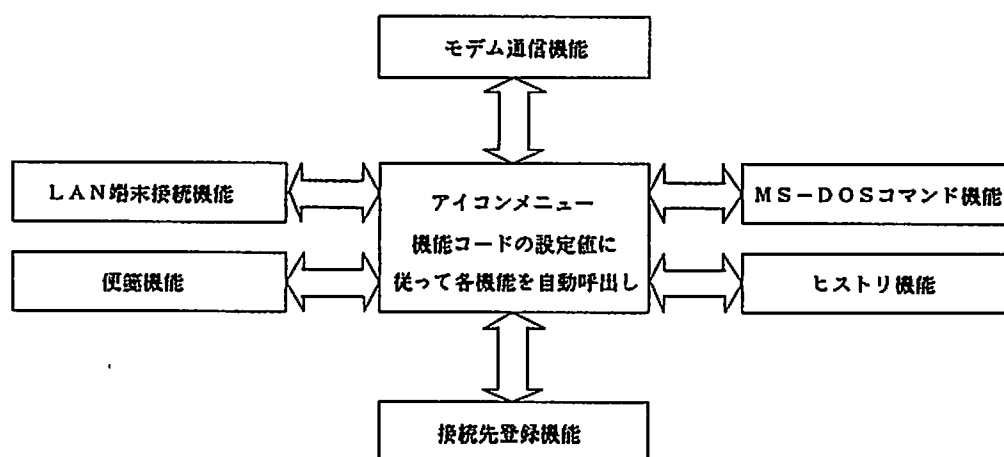


図3-17 アイコンメニューを介しての各機能の遷移図

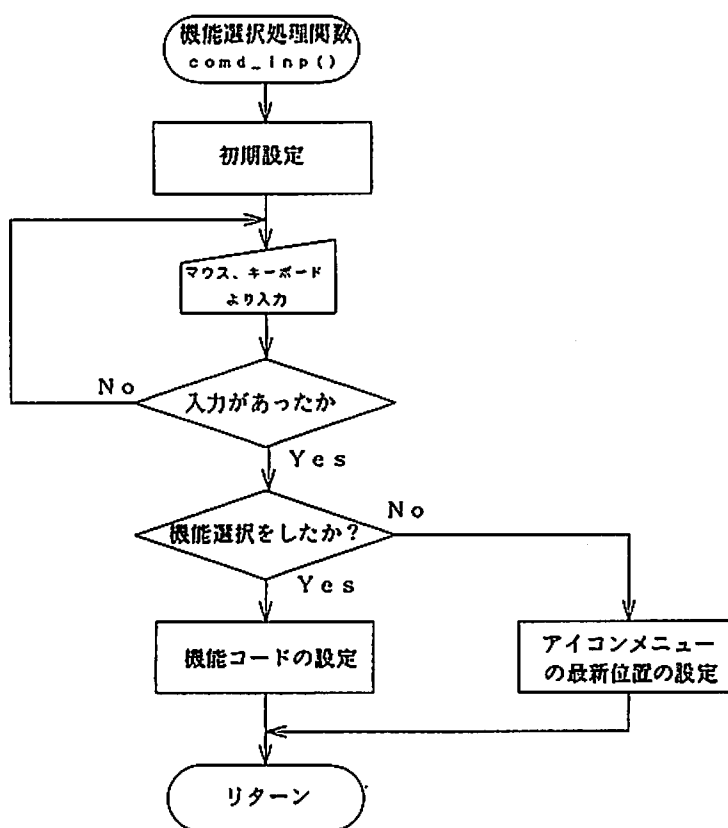


図 3 - 1 8 アイコンメニュー関数のフローチャート

3 - 1 0 飛火野用ウィンドウ関数の設計

本研究で作成するウィンドウ処理関数 `twindow()` には次に示す諸機能を持たせる。

1. 複数のウィンドウを一括して管理
2. 登録簿、ディレクトリ、メニューの表示に対応
3. マウス、キーボードによるウィンドウの操作に対応
4. ウィンドウの表示位置は画面上で自由に変更可能

ウィンドウ関数の動作構造を図3-19、図3-20に示す。

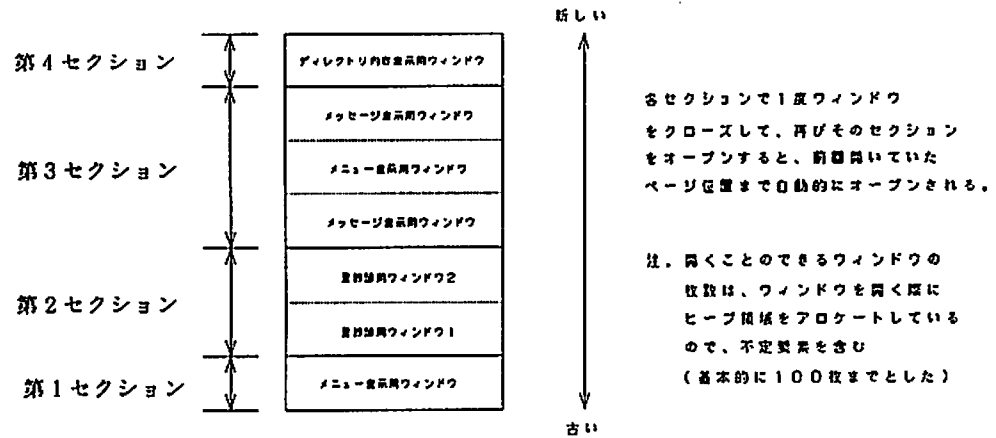
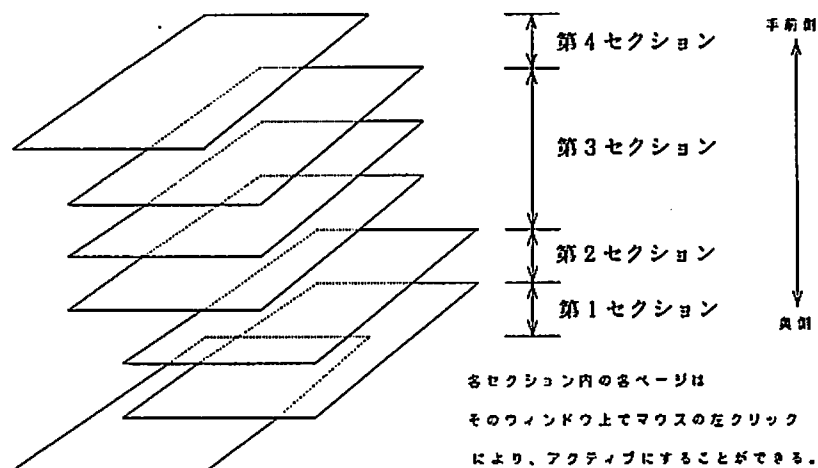


図3-19 各種ウィンドウをセクション化して管理



セクション間の相互関係

図3-20 ウィンドウ関数の動作構造

図3-21(A)(B)(C)にウィンドウ関数 `twindow()` のフローチャートを示す。

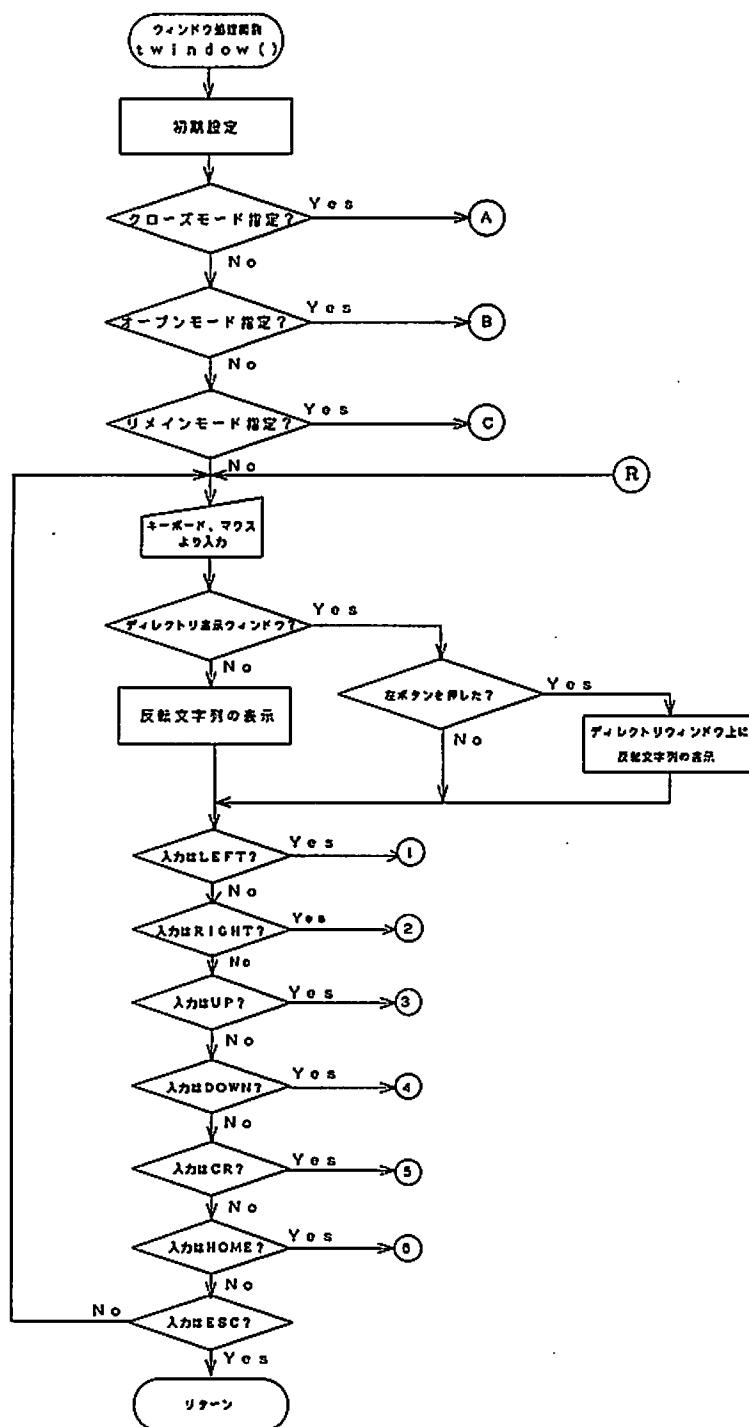


図 3 - 2 1 (A) twindow() のフローチャート

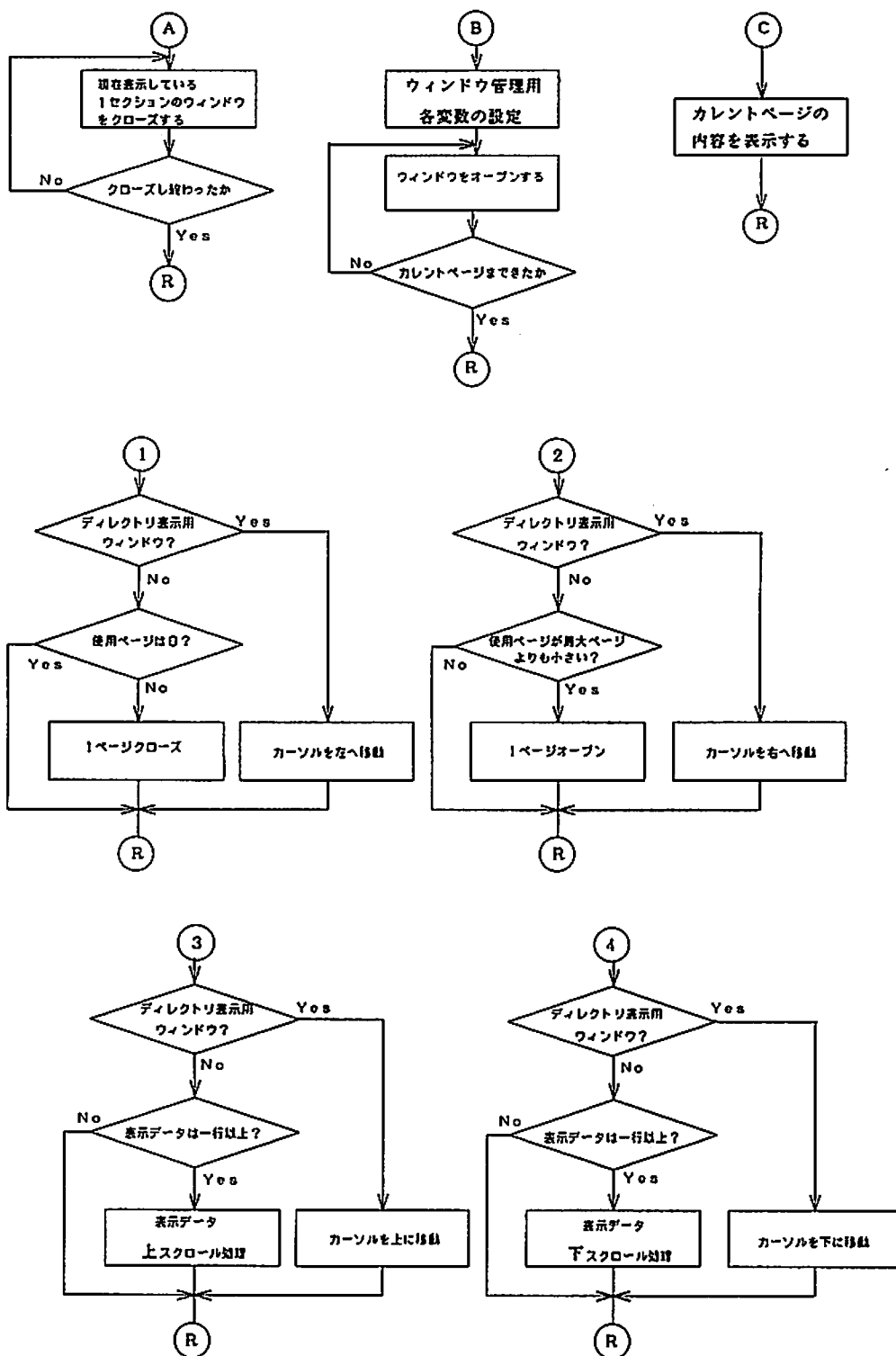


図 3 - 2 1 (B) `twindow()` のフローチャート

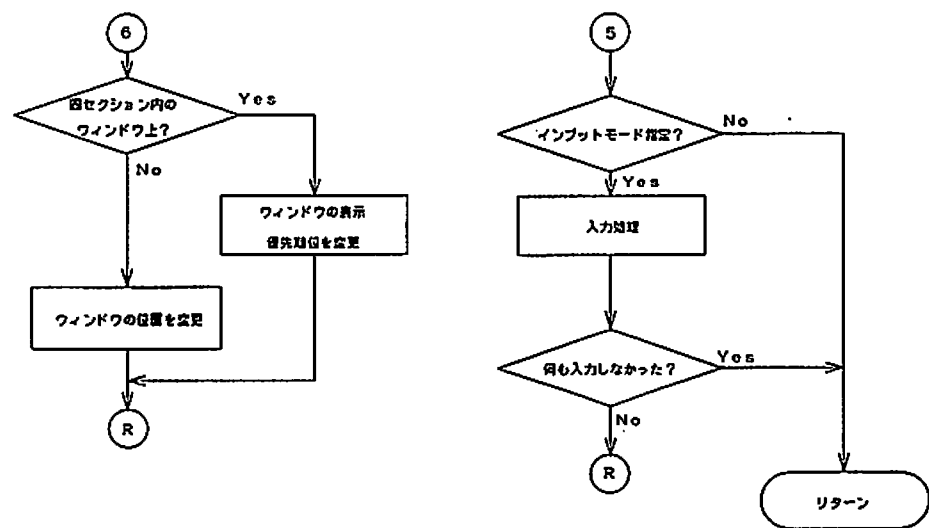


図 3 - 2 1 (C) twindow() のフローチャート

第 4 章 ソフトウェアの作成

4 - 1 LAN 端末接続機能の作成

LAN 端末接続機能を実現するために作成した関数を以下に示していく。

・ TELNET への接続: `telcon()`

コネクション開設用関数 `inittelnet()` を呼出してホストに接続した後、ログイン・データを送信し TELNET に接続する。図 4 - 1 に TELNET への接続手順を示す。

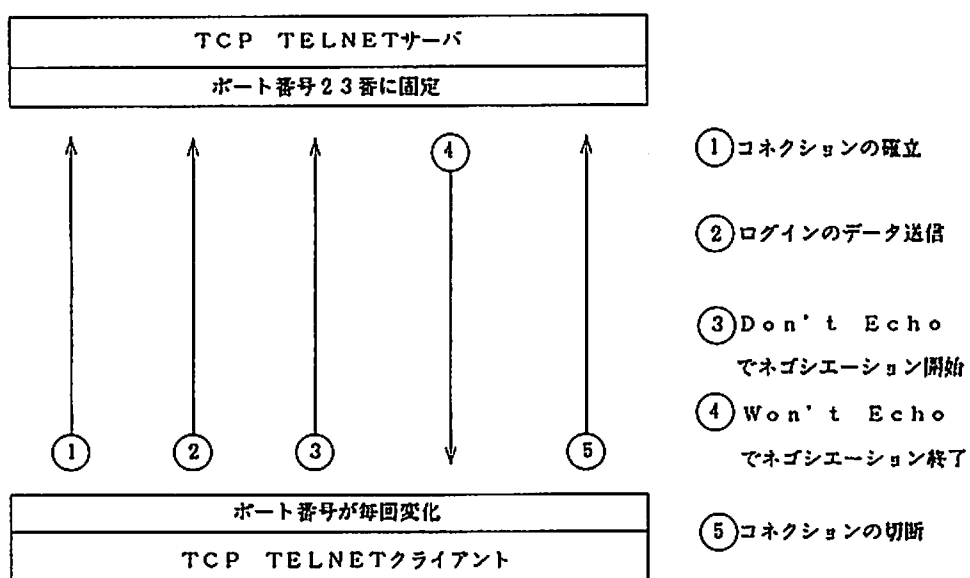


図 4 - 1 TELNET への接続手順

当初、ログイン・データが何であるのか分からなかった。しかし、試行錯誤を繰り返した結果、ホストコンピュータとのコネクションを開設したときに送られてくるデータをクライアント

ト側よりエコーバックすることで、TELNETサーバとの接続が可能であることが分かった。

・コネクションの切断: telcut()

シャットダウン実行用関数 fintelnet() を呼び出して、プログラム間通信を終了し、シャットダウンを行った後、通信を終了する。

・バッファマネージャ: bufman()

フレームの受信状態をチェックし、受信バッファへと転送する。sioinput() 関数より呼び出された時には、受信バッファより1バイトずつ取り出し戻り値として帰す。siooutput() 関数より呼び出された時には、与えられた1バイトのデータを送信バッファにセットし送信する。受信バッファ内の残留データ量の問い合わせに対して残留データ量を戻り値として帰す。

L A N 端末接続時の各種機能は、モデム通信時に使用している関数をフラグ切り替えにより使用しているため、次の4-2に示すモデム通信機能に準ずる。

尚、これらの作成した関数を付録O. に示す。

4 - 2 モデム通信機能の作成

モデム通信機能を実現するために作成した関数を以下に示していく。

- ・ 便箋送信機能: `edsend()`

1 ライン転送関数 `line_trs()` を使用して作成し表示関係にはウィンドウ関数を使用した。

- ・ 1行送信機能: `lineed()`

1 バイト (漢字の場合 2 バイト) 転送関数 `ksiooutput()` を使用して作成し、転送データの入力および選択はウィンドウ関数で作成した。

- ・ 回線切断機能: `conncut()`

ウィンドウ関数を使用して、切断するか否かを選択。切断ならば、YESを返し、そうでないときには、FALSEを返す。

- ・ 予約コマンド機能: `yoyakucom()`

ウィンドウ関数を使用して、予約コマンドファイル名の入力および選択を行い。予約コマンドファイル名が選択されれば、そのファイルを読み込み、ウィンドウ関数を用いて予約コマンドの入力および選択を行う。転送の際には、1 バイト転送関数 `ksiooutput()` を使用した。

- ・ ブレイク信号送信機能: `breakcry()`

ウィンドウ関数のメッセージ表示用ウィンドウを使用して、ブレーク信号を送信していることを画面に表示。ブレーク信号を送出する際に `siobreak()` を使用した。

・ 諸機能設定 : `setflag()`

ウィンドウ関数を使用して、機能設定項目の表示および選択を行い、設定したときには、その項目の先頭に '*' マークが表示される。設定した項目のフラグには -1 をセットする。

・ ファイル転送機能 : `ftrans()`

ウィンドウ関数を使用して設定項目の選択をし、以下のに示す関数を呼び出す。

プロトコルの設定 : `getprotoc()`

転送方向の設定 : `getdirect()`

1 行送信字数の設定 : `getlinecr()`

送信後待ち時間設定 : `getxfwait()`

ラインフィード処理 : `getlfpros()`

処理ファイル名獲得 : `getxfile()`

実行の際には、プロトコルに応じて以下に示す関数を呼び出す。

無手順の場合 : 送信時 `tty_transm()`

受信時 `tty_recive()`

X M O D E M の 場 合 : 送 信 時 xmod_transm()

受 信 時 xmode_receive()

Y M O D E M の 場 合 : 送 信 時 ymod_transm()

受 信 時 ymod_receive()

K E R M I T の 場 合 : 送 信 時 kerm_transm()

受 信 時 kerm_receive()

以上の処理を終えた後ウィンドウをクローズして復帰する。

尚、これらの作成した関数を付録 K.、付録 M. に示す。

4 - 3 M S - D O S コマンド機能の作成

M S - D O S コマンド機能を実現するために作成した関数を以下に示していく。

・ M S - D O S コマンド機能: gomsdos()

ウィンドウ関数を使用して、コマンドの入力および選択を

行う。コマンドの実行はsystem()関数を使用した。

尚、これらの作成した関数を付録 N. に示す。

4 - 4 接続先登録機能の作成

接続先登録機能を実現するために作成した関数を以下に示していく。

- ・ 設定モードメイン: `settei()`

登録簿用ウィンドウにより登録ファイル名の選択を行い、
設定を行う項目の選択およびその関数を呼び出す。

- ・ 設定パラメータ表示: `detail_change()`

ウィンドウ関数を用いて、設定中のパラメータを画面に表示する。電話回線とイーサネット回線では設定するパラメータが異なるので、その際の表示切り替えも行う。

次に各パラメータの設定を行うために作成した関数を示す。

- ・ N E T 名の登録: `setcntname()`

ウィンドウ関数を用いて、ネット名の入力を行う。

- ・ 電話番号の登録: `settelno()`

ウィンドウ関数を用いて、電話番号の登録を行う。

- ・ 回線の設定: `setline()`

ウィンドウ関数を用いて、回線タイプ（電話回線、イーサネット回線）の選択を行う。

- ・ ボーレートの設定: `setboudrate()`

ウィンドウ関数を用いて、ボーレート（300～9600bps）の選択を行う。

- ・ M N P の選択: `setmnp()`

ウィンドウ関数を用いて、M N P の選択（しない、する）を

行う。

- ・ 使用チャネルの選択: `setchannel()`

ウィンドウ関数を用いて、使用する S I O チャネル (1 ~ 3) の選択を行う。

- ・ データ長の設定: `setdatalength()`

ウィンドウ関数を用いて、データ長の選択を行う。

- ・ パリティの設定: `setparity()`

ウィンドウ関数を用いて、偶数・奇数・パリティなしのいずれかの選択を行う。

- ・ 通信方式の設定: `settrsmethod()`

ウィンドウ関数を用いて、全二重および半二重の選択を行う。

- ・ ストップビットの設定: `setstopbit()`

ウィンドウ関数を用いて、ストップビットの選択を行う。

- ・ X o n / X o f f あり、なしの設定: `setxonoff()`

ウィンドウ関数を用いて、X o n / X o f f のあり、なしの選択を行う。

- ・ データコードの設定: `setcode()`

ウィンドウ関数を用いて、データコードのタイプ (シフト J I S、旧 J I S、N E C 漢字、D E C 漢字、J I S 8、J I S 7) の選択を行う。

- ・ C R 受信処理の設定: `setreccr()`

ウィンドウ関数を用いて、C R 受信の際の処理を設定する。

- ・ C R 送信処理の設定: `setsndcr()`

ウィンドウ関数を用いて、C R 送信の際の処理を設定する。

- ・ L F 受信処理の設定: `setreclf()`

ウィンドウ関数を用いて、L F 受信の際の処理を設定する。

- ・ L F 送信処理の設定: `setsndlf()`

ウィンドウ関数を用いて、L F 送信の際の処理を設定する。

- ・ 話中待ち時間の設定: `setringsec()`

ウィンドウ関数を用いて、回線が既に使用されていた場合の待ち時間の設定を行う。

- ・ 再ダイヤル数の設定: `setringno()`

ウィンドウ関数を用いて、回線が既に使用されていた場合の再ダイヤルする回数を設定する。

- ・ 登録抹消: `rskill()`

設定したパラメータを全て消去し、登録内容と登録ファイルを抹消する。

尚、これらの作成した関数を付録 I. に示す。

4 - 5 便箋機能の作成

便箋機能を実現するために作成した関数を以下に示していく。

- ・ 便箋機能メイン: `editor()`

マウスの左ボタンにより機能メニュー処理関数 `editfunc()`

を呼出し、戻り値が 9 のときアイコンメニューに復帰する。

また、キーボードからの入力に応じてファンクションキー処理関数 `edcontrol()`、あるいは、文字処理関数 `edcharkey()` を呼出す。

- ・ 機能メニュー処理関数: `editfunc()`

ウィンドウ関数を使用して、各機能の選択・呼出しを行う。

選択した機能の番号を戻り値として返す。

以下に便箋機能における各種機能を実現するために作成した関数を示していく。

- ・ ファイルの読みだし: `edfilerd()`

ファイル名の入力・選択は `editfile()` を呼び出して行う。

読み込むファイル名が指定された場合、そのファイルをオープンして、内容を便箋のバッファに取り込む。

- ・ ファイルの書き込み: `edfilewr()`

ファイル名が指定されていない場合、ウィンドウ関数を使用してファイル名を入力し、そうでない場合、上書きすることへ

の確認をする。以上の手続きを経たところで、ファイルへの書き込みを行う。

・ 便箋白紙機能: `edclrmail()`

エディット領域を初期化するのに `edbufnew()` を使用し、エディットの際に使用する各種変数の初期設定を行う。

・ 文字列検索機能: `edsearch()`

ウィンドウ関数により検索文字列の入力を行う。その後、現在の表示位置以降の検索文字列を順番に探していき、検索文字列が発見出来なくなったところで終了する。

・ 範囲指定: `edareaset()`

範囲削除、範囲指定の対象となる領域の始点、終点を設定する。

・ 範囲削除: `edareadel()`

範囲指定で設定された領域を削除する。

・ 範囲複写: `edareacpy()`

範囲指定で設定された領域を別領域にコピーする。

・ 表示幅の設定: `edwidth()`

入力する文書 1 ラインの最大表示幅を設定する。

尚、これらの作成した関数を付録 J. に示す。

4 - 6 ヒストリ機能の作成

ヒストリ機能を実現するために作成した関数を以下に示していく。

- ・ ヒストリー機能メイン: `history()`

ヒストリモードで利用できる各種機能の選択、キー入力処理

関数 `hist_keyin()` の呼出しを行う。

ヒストリモードでの各種機能の実現のために作成した関数を以下に示していく。

- ・ 便箋複写機能: `hist_copy()`

ヒストリのバッファ領域の内容を便箋バッファ領域に複写する。

- ・ ファイル記録機能: `hist_file()`

ヒストリの内容をファイルにセーブする。ファイル名は、`hist_func()` で読み込んだものを使用する (`hist_func()`はこの機能の前に呼び出される)。

- ・ プリント機能: `hist_print()`

ヒストリの内容をプリンターに出力する。プリンターに出力する際には、1 バイトプリンタ出力関数 `printer()` を使用する。

- ・ 文字検索機能: `hist_serch()`

ウィンドウ関数により、検索文字列の入力を行う。その後、現在の表示位置以降の検索文字列を順番に探していき、検索文

字列が発見出来なくなったところで終了する。

・履歴先頭: `hist_topgo()`

ヒストリバッファの先頭行に変数を設定し画面を描き直す。

画面の書き直しには、`hist_chardisp()` を使用した。

・履歴末尾: `hist_endgo()`

ヒストリバッファの最終行に変数を設定し画面を描き直す。

・範囲削除: `hist_del()`

リターンキーにより設定していた範囲内のデータを消去する。

リターンキーで範囲が設定されていない場合、全行を処理する

かどうかを `chkallline()` により確認する。

以下にヒストリモードでのキー入力処理を行うために作成した関数
を示していく。

・キー入力処理: `hist_keyin()`

ファンクションキーや、キャラクタキーが入力された場合の

処理を行う、各種関数の呼出しを行う。

キー入力処理の際に使用する関数を以下に示していく。

・リターンの処理: `hist_cr()`

範囲指定の際の始点、終点を設定する際に使用する。

・エスケープの処理: `hist_esc()`

指定した範囲の設定解除を行う際に使用する。

- ・ U P キー入力の際の処理: `hist_up()`

カーソルキーの'上'を押したときの処理を行う。

- ・ D O W N キー入力の際の処理: `hist_down()`

カーソルキーの'下'を押したときの処理を行う。

- ・ R U P キー入力の際の処理: `hist_rup()`

ロールアップキーが押された時の処理を行う。

- ・ R D O W N キー入力の際の処理: `hist_rdn()`

ロールダウンキーが押された時の処理を行う。

尚、これらの作成した関数を付録 L. に示す。

4 - 7 アイコンエディタの作成

アイコンエディタの諸機能を実現するために作成した関数を以下に示していく。

- ・ キー操作の説明を画面下に表示: `menu()`
- ・ 画面に升目を表示する: `flame()`
- ・ 画面設定: `scr_set()`
- ・ アイコンデータをディスクにセーブ: `pat_set()`
- ・ アイコンデータをディスクより読み込む: `scr_load()`
- ・ 指定アイコンデータをディスクより読み込む: `edit()`
- ・ アイコンエディタメイン: `main()`

画面設定、コマンド入力、カーソル移動などの処理を行う。

尚、これらの作成した関数を付録 D. に示す。

4 - 8 アイコンメニューの作成

アイコンメニューによる各種関数の呼出しを行うために以下に示す関数を作成した。

- ・ アイコンデータの読み込み： `icset()`
- ・ アイコンメニューの表示： `icon_disp()`
- ・ アイコンメニューによる機能の選択： `comd_inp()`

尚、これらの作成した関数を付録 N. に示す。

4 - 9 飛火野用ウィンドウの作成

マルチウィンドウを実現するために作成した関数を以下に示す。

- ・ 全表示ウィンドウの消去： `allclose()`
- ・ ウィンドウ用メモリの取得： `getmem()`

ウィンドウを表示する際に下敷きとなる領域の情報を納めるためのバッファを取得する。

- ・ マウス、キーボードからの入力処理： `wdkey_in()`
- ・ データ入力モードでの入力受け付け： `input()`
- ・ ウィンドウのフレーム作成： `flame()`

- ・ ウィンドウ内へのデータの表示: `dspdata()`
- ・ 反転文字列、通常属性文字列の表示: `print()`
- ・ ディレクトリウィンドウにおけるデータ表示: `fdspdata()`
- ・ ディレクトリウィンドウにおける反転文字列、通常属性文字列の表示: `fprint()`
- ・ ウィンドウ関数メイン: `twindow()`

全表示ウィンドウの管理および、各種ウィンドウ（メニュータイプウィンドウ、登録簿タイプウィンドウ、ディレクトリ内容表示用ウィンドウ）を一括して管理する。各種のウィンドウの作成の仕方については付録 A、B、C に示した。

尚、これらの作成した関数を付録 N. に示した。

第 5 章 プログラムの実行

5 - 1 モデム通信機能の実行

図 5 - 1 にアイコンメニューでモデム通信機能を選択し、本校の電子計算機室と接続したときの画面を示す。

図 5 - 2 に 1 行送信機能を選択したときの実行画面を示す。

1 行送信用ウィンドウ上では、送信内容の入力・選択を行うことができる。ここで登録された内容は、自動的に記憶されるので次回接続したときにも再入力する必要はない。

図 5 - 3 に接続モードでファイル転送を選択しマルチファイル名の入力を行っている画面を示す。（マルチファイル転送は複数のファイルの送受信を一度に行うことができる）

図 5 - 4 にファイル転送を行った直後の画面を示す。

図 5 - 5 に接続モードで機能フラグ設定モードを選択した画面を示す。この機能ではリアルタイムの通信内容ハードコピー、画面右上へのタイマー（接続時間と現在時間）の表示設定等の接続時に役立つ機能を設定することができる。

図 5 - 6 に予約コマンド機能を選択して作成するコマンドファイル名を入力し、予約コマンドの内容を入力している画面を示す。予約コマンドファイルは 8 個作成でき、各々のファイルには 18 コマンド分の登録が可能である。

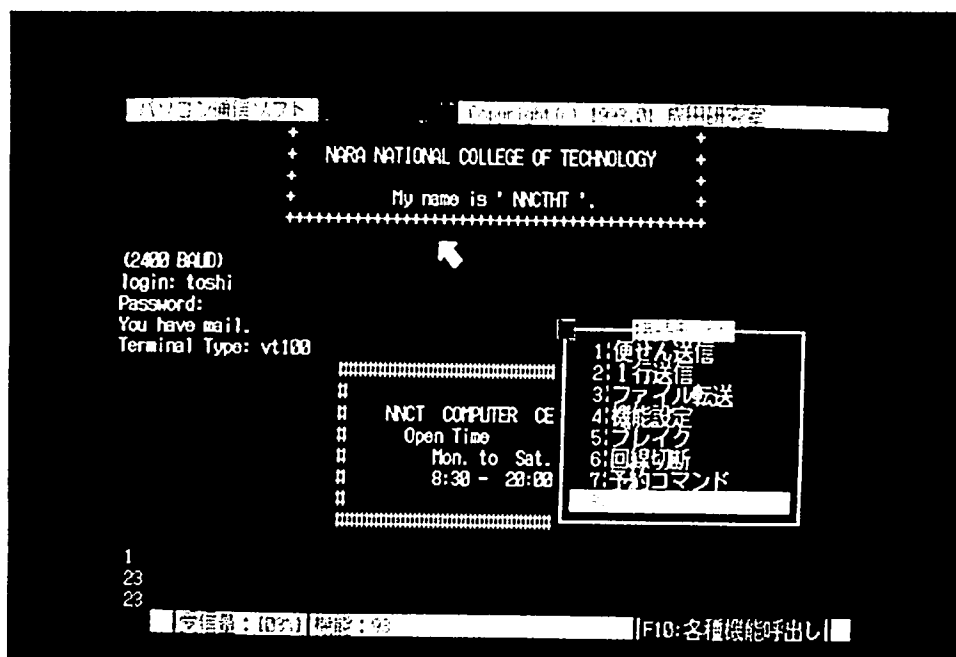


図 5 - 1 本校電子計算機室 A - 4 0 0 と接続した画面

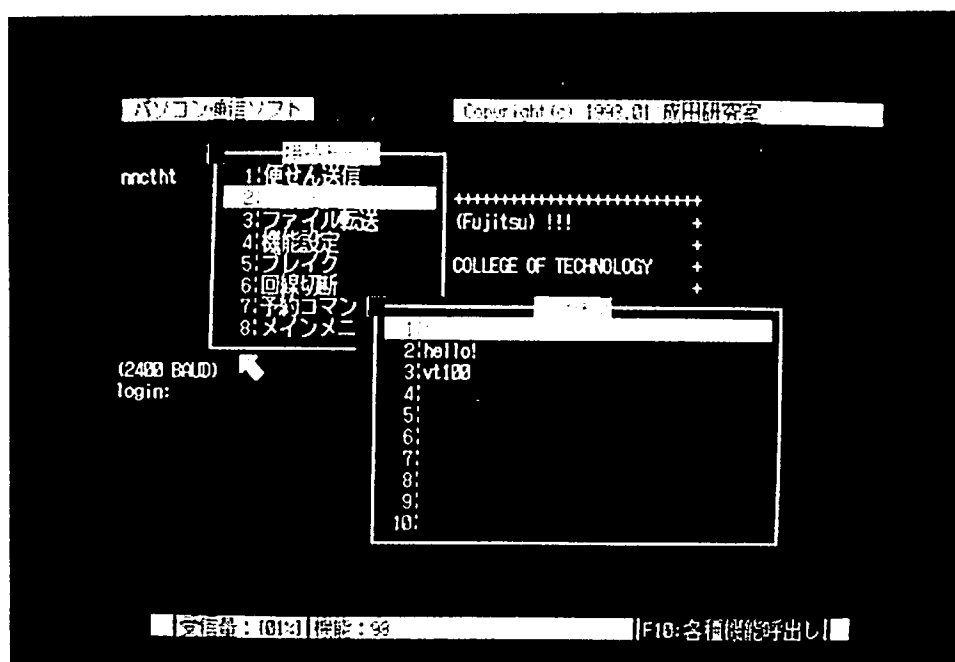


図 5 - 2 接続モードの 1 行送信機能の実行画面

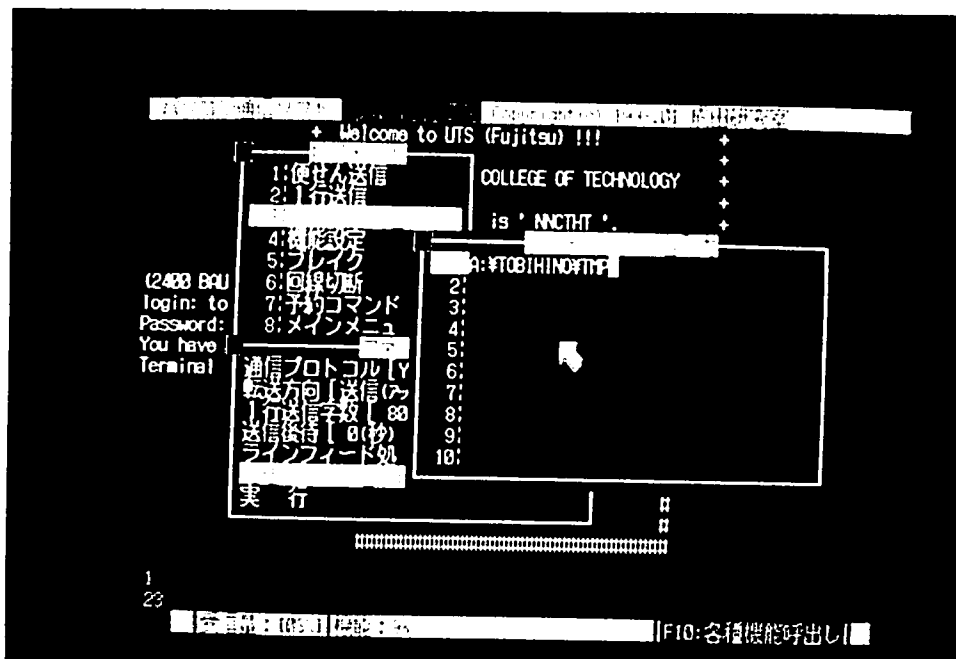


図 5 - 3 ファイル転送でマルチファイル名を入力している画面

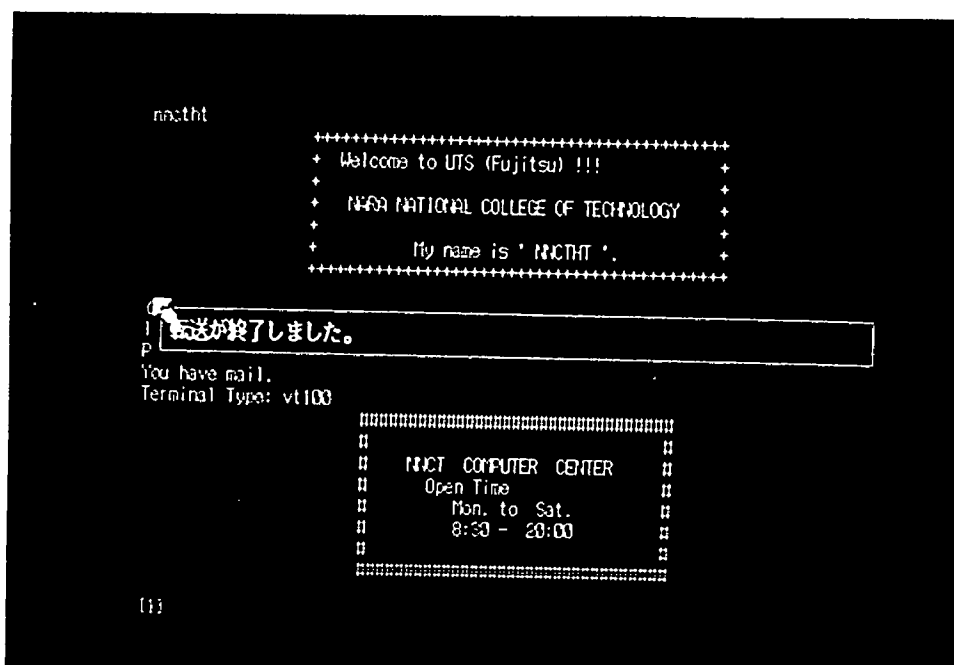


図 5 - 4 ファイル転送機能使用直後の画面

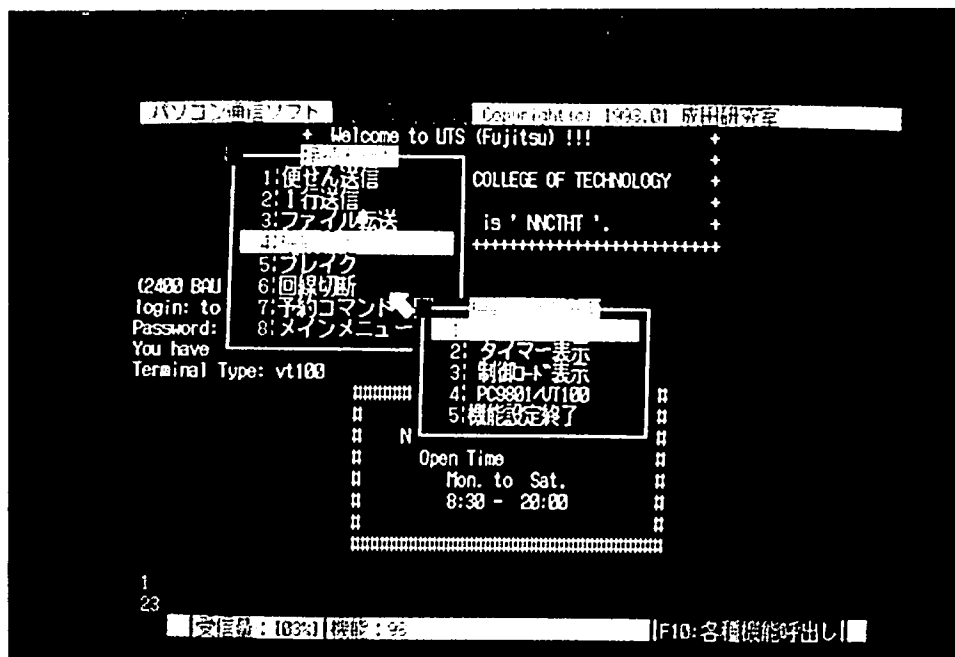


図 5 - 5 機能フラグ設定時の実行画面

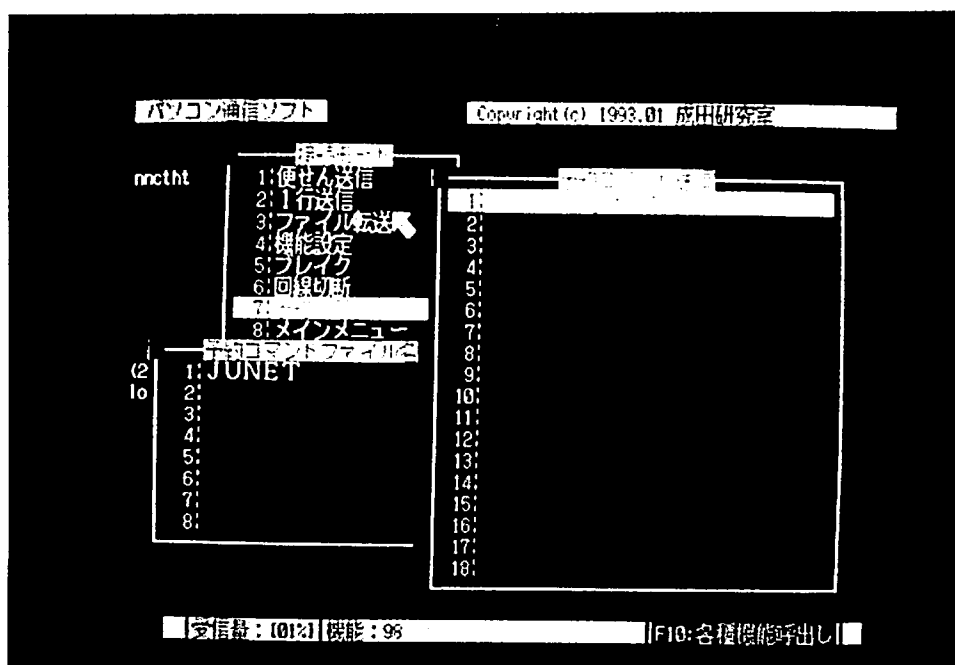


図 5 - 6 予約コマンド機能選択時の実行画面

5 - 2 LAN 端末接続機能の実行

図 5 - 7 に本研究室の S O N Y - N E W S に接続した際の実行画面を示す。さらに、図 5 - 8 に v i エディタを利用している画面を示す。

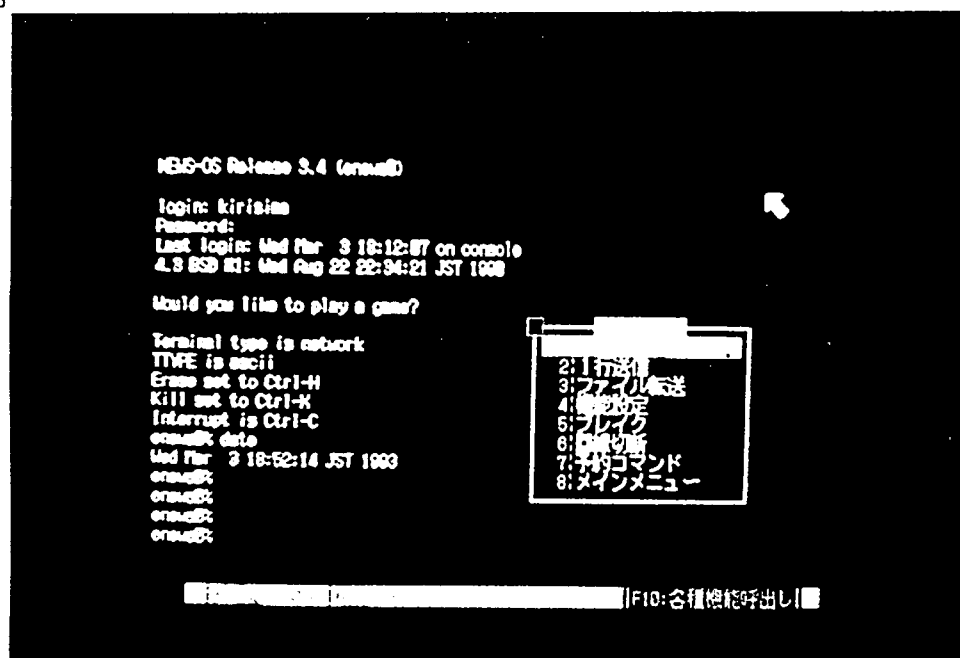


図 5 - 7 S O N Y - N E W S に接続した時の実行画面

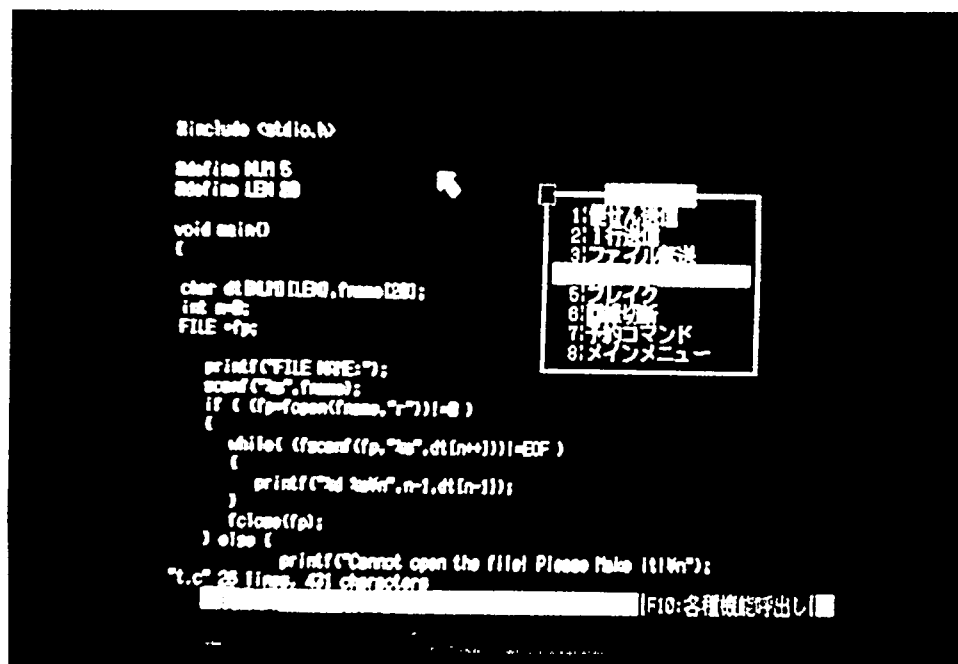


図 5 - 8 v i エディタを利用している画面

5 - 3 M S - D O S コマンド機能の実行

図 5 - 9 にアイコンメニュー画面で M S - D O S 機能を選択した時の画面を示す。図 5 - 9 はコマンドの登録・実行用画面であるが、何も入力せずに実行（空文の実行）するとコマンドライン上で M S - D O S コマンドを使用できる。

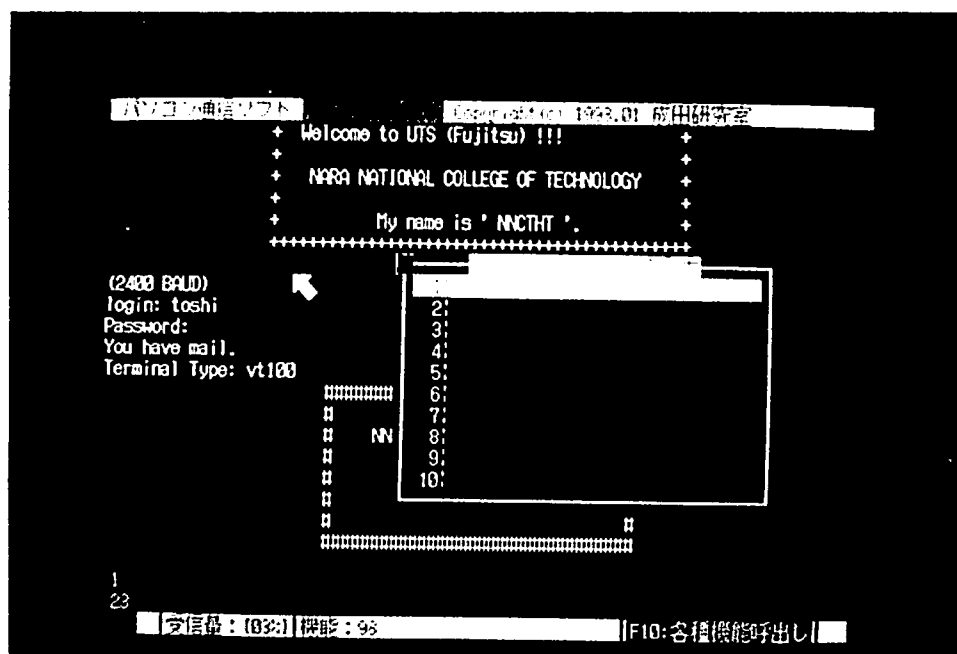


図 5 - 9 M S - D O S コマンド機能の実行画面

5 - 4 接続先登録機能の実行

図 5 - 10 にアイコンメニューで接続先登録機能を選択した画面を示す。図 5 - 10 はネット名の入力中の画面である。

接続先の登録は 38 件まで可能とした。



図 5 - 1 0 接続先登録機能の実行画面

5 - 5 便箋機能（エディタ）の実行

図 5 - 1 1 に便箋機能での各種機能選択のメニュー画面を示す。

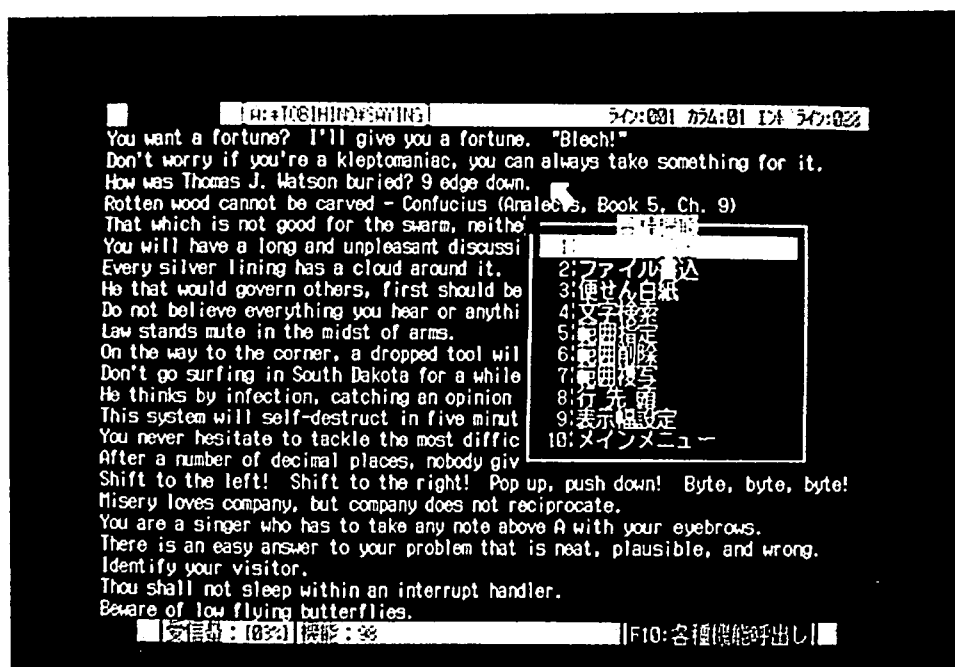


図 5 - 1 1 便箋機能での各種機能選択メニュー

5 - 7 アイコンエディタの実行

図5 - 14 にアイコンエディタの実行画面を示す。図5 - 14 は、モデム通信機能用のアイコンをエディットしている画面である。

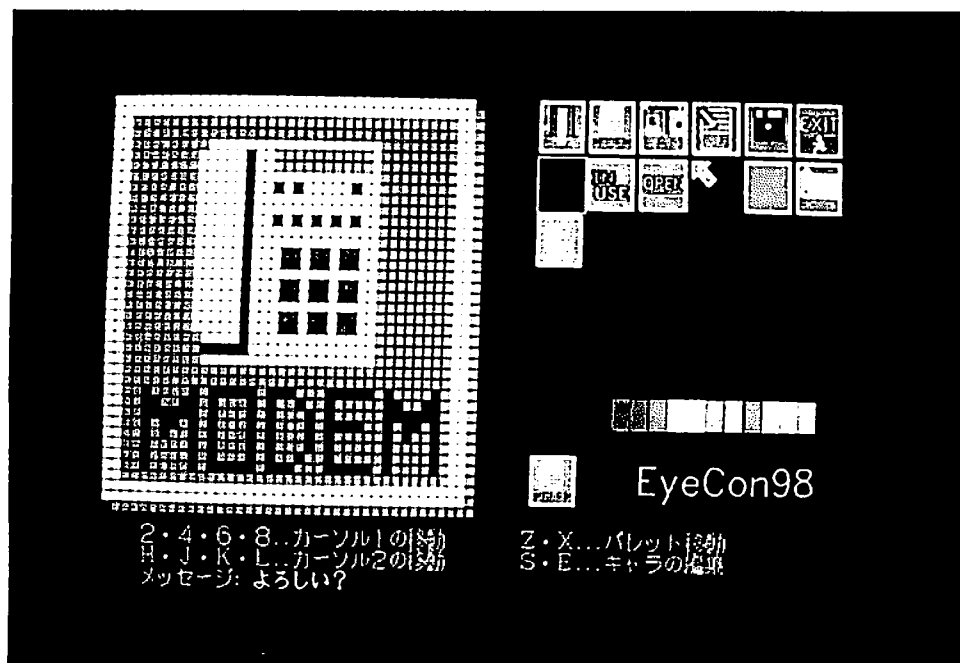


図5 - 14 アイコンエディタの実行画面

5 - 8 アイコンメニュー機能の実行

図 5 - 1 5 にアイコンメニュー関数により呼び出されたアイコンメニューの表示画面を示す。

機能の選択はマウスを使用する代わりにカーソルキーでもできる（カーソルキーでマウスカーソルを移動できる）。また、ファンクションキーにより各機能を選択することも可能とした。

将来の機能拡張のために 2 つ空アイコンを作成しておいた。

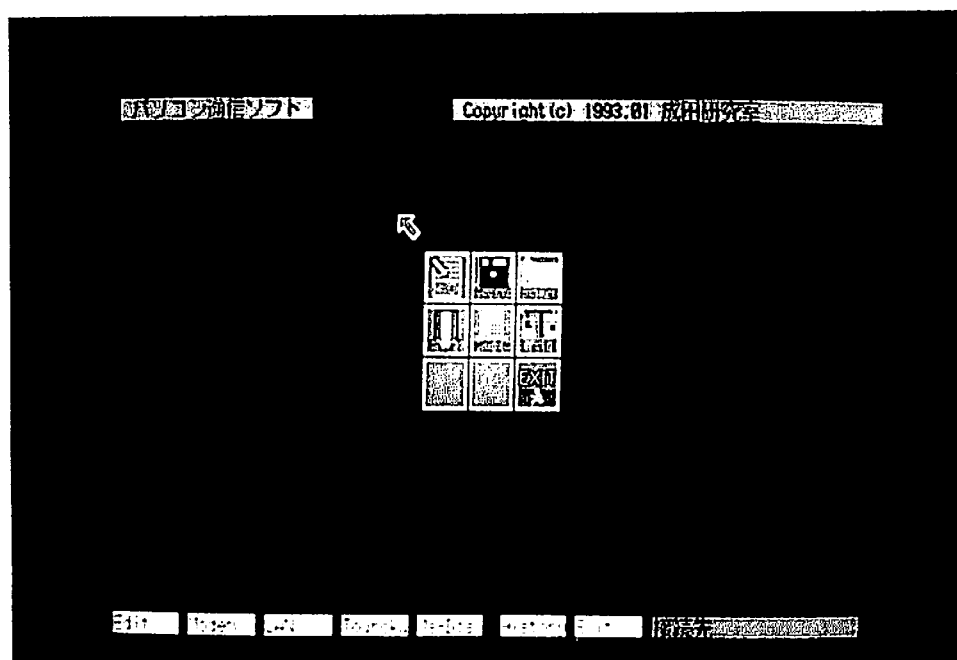


図 5 - 1 5 アイコンメニュー機能の実行画面

第 6 章 実行結果と検討

6 - 1 実行結果

本研究では、画面に表示される機能メニュー等をマウス（キーボードでも可）により選択することでパソコン通信ができる、マンマシンインタフェースの優れたパソコン通信ソフトを作成することができた。さらにイーサネット回線を使用しての LAN 端末接続機能、モデムを使用しての通信機能も設計していた通りに動作させることができた。

その他の便箋機能、MS-DOS コマンド機能、接続先登録機能、ヒストリ（履歴）機能も設計していた通りに動作させることが出来た。

6 - 2 結果の検討

多種多彩の機能を 1 つのソフトウェアに凝縮したために本研究で作成した通信ソフトの実行ファイルは約 230kB という大きなものとなってしまった。本ソフトウェアのモデム通信機能、便箋機能は、本研究室にあるパソコン通信ソフトの関数群を利用して作成したが、この関数群のなかで目についたのが表示関係とキー入力関係に多くの手間暇がかかるということであった。

より一層のプログラム開発効率の向上と操作性の向上、そして無駄なメモリの使用を極力抑えるために、本研究では、マルチウィン

ドウ関数の作成を行った。

実行ファイルの大きさは作成したマルチウィンドウ関数を使用する前は約260kBであったのに対して、使用後は約230kBとやや減少しているものの、依然大きな実行ファイルとなってしまうている。

主に考えられる原因としては、

- ・ 便箋機能、LAN端末接続機能で大きな外部変数領域を使用している。(これは、メモリ管理を省き処理速度を維持するためにやむを得ない。)
- ・ モデム通信機能、LAN端末接続機能では、各種機能をいつでもアクセスできるように設計しているため、通信中の画面をセーブしておかなくてはならないために、比較的大きな画面セーブ領域を必要とする。

が挙げられるがこれらは最低限必要となるものなのでどうしても確保していなくてはならない。

しかしながら、ソースファイルの大きさは、当初約14,000行あったものを約10,000行に抑えることができた。

本研究で作成したLAN端末接続機能は、接続先のホストコンピュータ(UNIX)の提供している数々の機能(例えば、viエディタ等)を利用することができる。パソコン上に大きなシステムを

載せるよりも、ホストコンピュータにLANで接続の方が能率の面でも費用の面でもメリットは大きいことから本研究で作成したLAN端末接続機能は有意義であると思われる。

第7章 あとがき

本研究では、アイコン、マウス、マルチウィンドウといったGUI環境を実現するために欠かせない要素を取り入れ、初心者向けの操作の簡単なパソコン通信ソフトの開発を行った。

アイコン、マウスカーソルのデザインは人それぞれに好みが違うのでアイコンエディタを作成し、いつでも変更可能にしておいた。また、マウスをウィンドウ、アイコンメニューに対応させることで、ソフト全体の操作性を高めることができた。

本年度は、イーサネットを利用してのLAN端末接続を可能にするべく、ソフトウェアの設計を行い、仮想端末としてUNIXへと接続することができた。さらに、モデムを使用しての通信機能等、当初設計していた機能もすべて正常に動作した。

今後は、230KBという本ソフトウェアの実行可能ファイルの縮小化を図り、通信ログの編集の際に欠かすことの出来ない便箋機能のさらなる強化を図り、より実用的な通信ソフトにされることを期待する。

【 謝 辞 】

本研究を遂行するにあたり、常にご指導してくださいました、
成田絃一教授、中村善一助教授に深く感謝いたします。

また、本研究で使用するためのハードウェアの開発に御尽力下さいました、香川県工業技術センターの濱田敏弘さん、本研究で必要な資料・資材を探す際にご協力いただきました、情報工学科の西野貴之技官、電子計算機室の川辺涼子さんに深く感謝いたします。

最後に5年間いろいろとお世話していただきました電気工学科の先生方を始め、一般教科の先生方、研究する際に数々の助言をいただきました成田・中村研究室の皆様には感謝の意を表したいと思えます。

【 参 考 文 献 】

(1) 手 作 り パ ソ コ ン L A N シ ス テ ム

荒 井 文 吉 著

技 術 評 論 社

(2) P C 9 8 0 1 実 践 パ ソ コ ン 通 信

太 平 洋 工 業 株 式 会 社 編

日 刊 工 業 新 聞 社

(3) イ ン タ ー フ ェ イ ス 1 9 9 2 . 1 1 増 刊

オ ー プ ン シ ス テ ム 入 門 教 室

小 倉 裕 明 著

C Q 出 版 社

(4) 異 機 種 接 続 と T C P / I P 絵 と き 読 本

道 下 宣 博 / 本 間 泰 則 共 著

オ ー ム 社

(5) T u r b o C V e r 2 . 0 初 級 プ ロ グ ラ ミ ン グ

河 西 朝 雄 著

技 術 評 論 社

(6) 平成 2 年度電気工学科卒業論文

マルチウィンドウに関する研究

福井 誠之

奈良高専電気工学科

(7) 平成 3 年度電気工学科卒業論文

パソコン LAN に関する研究

前田 行輝

奈良高専電気工学科

(8) I n e t B I O S 対応アプリケーション開発ガイド

株式会社アスキー

【 付 録 】

A. ウィンドウ用構造体 W I N D O W の説明	付 - 1
B. ウィンドウ関数 twindow() の使用方法	付 - 5
C. ウィンドウ関数の使用例とその設定値	付 - 6
D. アイコンエディタのプログラムリスト	付 - 9
E. メイクファイル	付 - 1 4
F. 外部変数宣言ファイル	付 - 1 5
G. ヘッドファイル	付 - 1 6
H. ソースファイル tobihino.c	付 - 2 4
I. ソースファイル set.c	付 - 2 8
J. ソースファイル ed.c	付 - 4 2
K. ソースファイル connect.c	付 - 6 5
L. ソースファイル hist.c	付 - 9 6
M. ソースファイル ftrans.c	付 - 1 0 7
N. ソースファイル hand.c	付 - 1 4 1
O. ソースファイル ether.c	付 - 1 6 9

注. E ~ O は本通信ソフト関連のファイルである。

付録 A. ウィンドウ用構造体 W I N D O W

本研究で作成したウィンドウは、手軽に使用できるように、各機能やパラメータをウィンドウ構造体に代入するだけでよいものとした。次にウィンドウ構造体を示す。

```
typedef struct WD {  
  
    int x;          /* ウィンドウの表示座標 */  
  
    int y;  
  
    int wx;         /* ウィンドウの大きさ */  
  
    int wy;  
  
    int col1;       /* フレームの色 */  
  
    int col2;       /* タイトルの色 */  
  
    int col3;       /* 表示文字列の色 */  
  
    int l;          /* カーソルの座標 */  
  
    char *title;    /* タイトル */  
  
    int inp_mode;   /* ウィンドウ機能 */  
  
    int cmax;       /* カーソル移動上限 */  
  
    int cmin;       /* カーソル移動下限 */  
  
    int button;     /* フレームのタイプ */  
  
    int cursor_on; /* カーソル表示のフラグ */  
  
    int lineno_on; /* ライン番号表示フラグ */  
};
```

```
int inp_line; /* データ入力行の指定 */

} WINDOW;
```

次に各設定項目について説明する。

・ ウィンドウ表示座標

ここでは、指定ウィンドウの画面上での表示座標を定める。

```
int x; ( 1 < x < 80 )
```

```
int y; ( 1 < y < 25 )
```

・ ウィンドウの大きさ

ここでは、指定ウィンドウの大きさを定める。

```
int wx; ( 4 < wx < 80 )
```

```
int wy; ( 1 < wy < 25 )
```

・ フレーム、タイトル、表示文字列の色

ここで指定する色は、Turbo C のヘッダファイル conio.

h で定義されているテキストカラーのデフォルト値を使用する。

・ カーソルの座標

ここでは、ウィンドウ内でのカーソルの座標を設定する。

・ タイトルの設定

ここではウィンドウ最上部に表示するタイトル文字列へのポ

インタを設定する。

・ ウィンドウ機能の設定

ここでは、指定ウィンドウの機能を設定する。

```
int inp_mode;
```

inp_mode=0 の時、データ内容の更新

inp_mode=1 の時、データ内容を表示

inp_mode=2 の時、データ内容の選択

・カーソル移動の上限と下限の設定

ここでは、指定ウィンドウ上でのカーソルの移動範囲を設定する。

・フレームのタイプの設定

ここでは、指定ウィンドウのフレームのタイプを設定する。

```
int button;
```

button=0 の時ウィンドウ左上にクローズボタン表示

button=1 の時ウィンドウ上に上下左右移動用ボタンと

クローズボタンを表示する。

button=2 の時ボタン類は何もウィンドウ上には表示し

ない。

・カーソル表示フラグの設定

ここでは、指定ウィンドウ上でカーソルの表示を行うか否かを指定する。

```
int cursor_on;
```


`cursor_on=1` の時、カーソルを表示

`cursor_on=0` の時、カーソルは表示しない

・ライン番号表示フラグの設定

ここでは、指定ウィンドウ上でライン番号の表示を行うか否かを指定する。

```
int lineno_on;
```

`lineno_on=1` の時、ライン番号を表示する

`lineno_on=0` の時、ライン番号を表示しない

・データ入力行の設定

ここでは、`inp_mode=0` に設定したときの入力行を指定する。

```
int inp_line;
```

`inp_line= -1` の時、入力対象は全行

それ以外は、任意の行

尚、ディレクトリ表示用ウィンドウの場合 `inp_line` の値は、1ライン当たりのファイル名表示個数とみなされ処理される。

付録 B. ウィンドウ関数 twindow() の使用方法

次に twindow() のプロトタイプを示す。

```
int twindow(    int MAXPAGE  
  
               ,int mode  
  
               ,WINDOW wd[]  
  
               ,char wddata[][DATA_WD]    )
```

MAXPAGE は、表示するウィンドウの枚数を表し、mode は使用するウィンドウのタイプを指定する。これは、ヘッダファイル NFIRE.H で宣言されているものを使う。以下にデフォルト値を示す。

mode=WD_OPEN はメニュー選択、登録簿タイプのウィンドウをオープンする時に使用する。

mode=WD_REMAIN は一度使用したウィンドウを活性化する際に使用する。

mode=FW_OPEN はディレクトリ表示タイプのウィンドウをオープンする時に使用する。

mode=FW_REMAIN はディレクトリ表示タイプのウィンドウを活性化する際に使用する。

mode=WD_CLOSE は一番新しいウィンドウ(1セクション全て)を閉じる。

付録 C. ウィンドウ関数の使用例とその設定値

図 8 - 1 にメニュー選択用ウィンドウの実行例を示す。

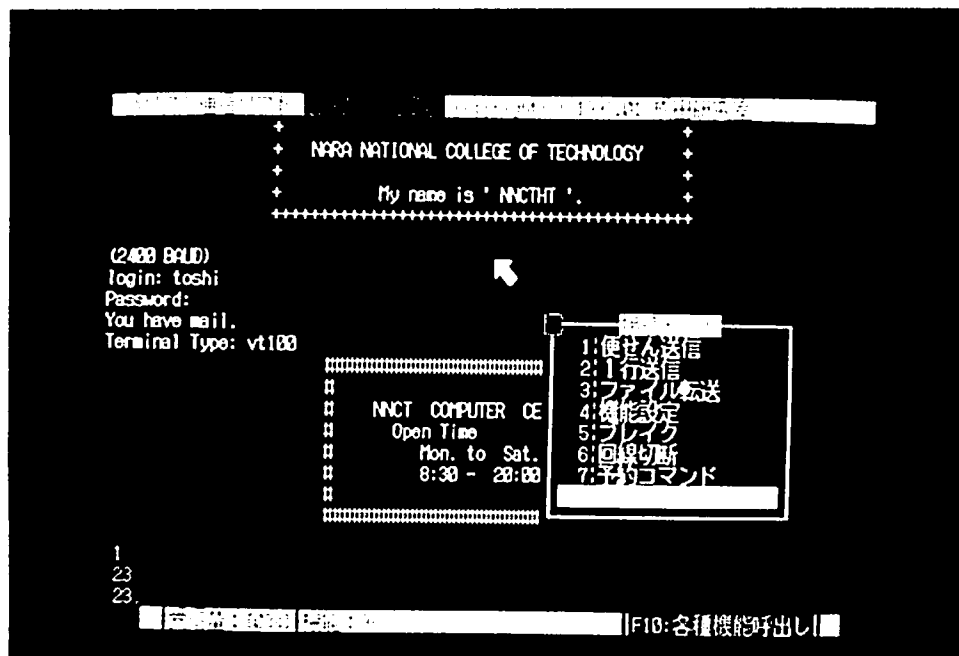


図 8 - 1 メニュー選択用ウィンドウの実行例

次にこのウィンドウの場合の設定方法を示す。

```
WINDOW wd={ x,y,wx,wy,T_CYAN,T_YELLOW,T_WHITE,0,"接続モード"
```

```
,2,8,0,0,1,1,0 };
```

```
static char data[8][DATA_WD]=
```

```
{ "便せん送信", ..., "メインメニュー" };
```

呼出す方法

```
twindow( 1,WD_OPEN,&wd,data );
```

クローズする方法

```
twindow( WD_CLOSE );
```

次に示す図 8 - 2 に登録簿用ウィンドウの実行例を示す。

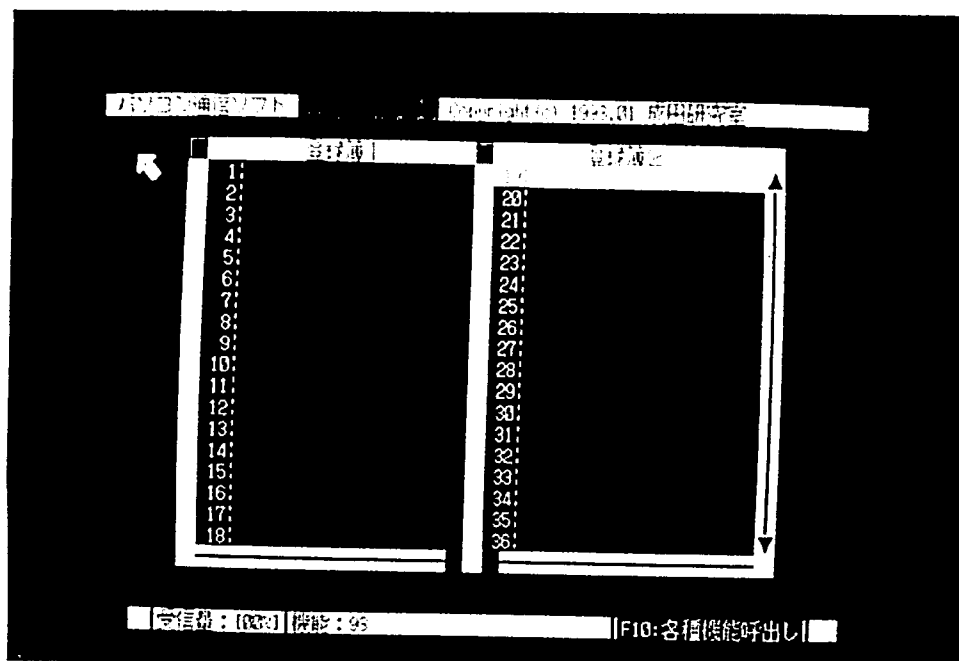


図 8 - 2 登録簿用ウィンドウの実行例

この場合の設定例は図 8 - 1 で説明したときのウィンドウ構造体を配列にして設定し受け渡したものとなる（ページ数設定に注意）。

また、メッセージ表示用ウィンドウの例は、図 8 - 3 にしめす。

この場合は、`wd.button=2` とすれば上記の例と同様に行える。

ディレクトリ表示用ウィンドウの例を図 8 - 4 に示す。この場合は、呼び出すときに、`WD_OPEN` の代わりに `FW_OPEN` を指定すればよい。

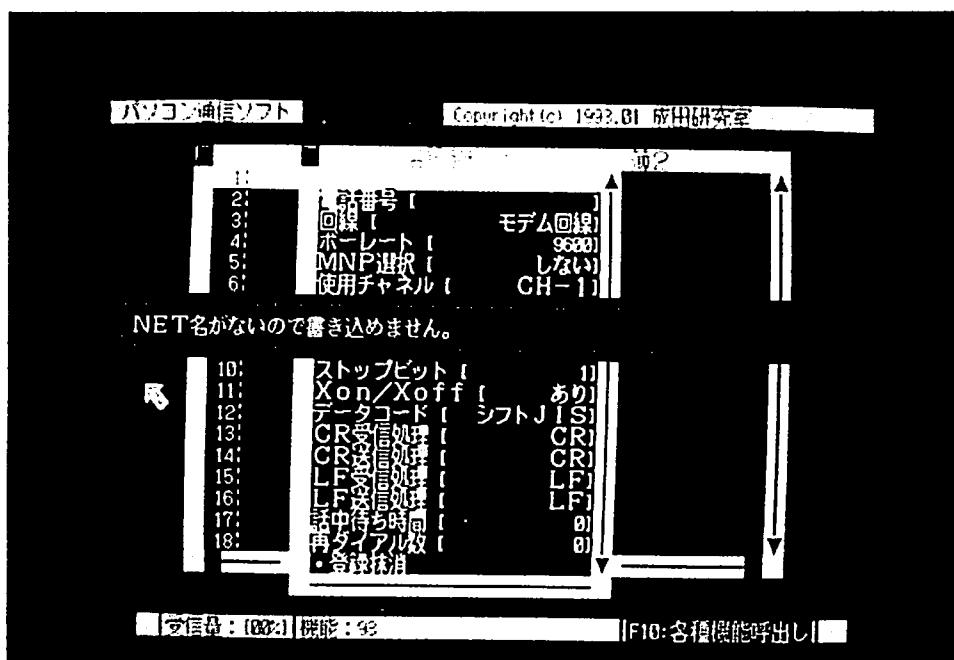


図 8 - 3 メッセージ表示用ウィンドウの実行例

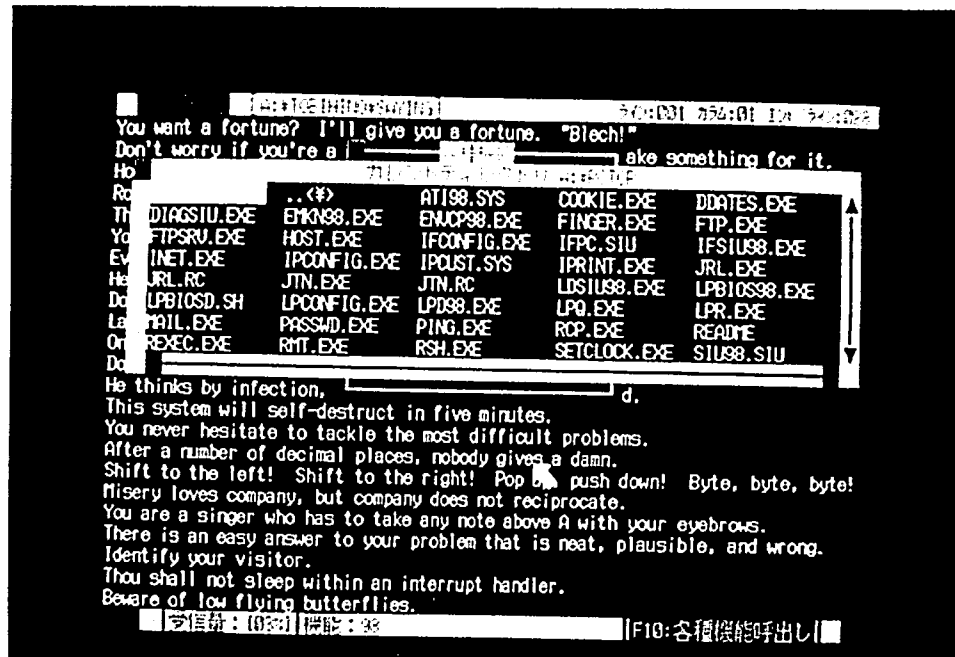


図 8 - 4 ディレクトリ表示用ウィンドウの実行例

付録D. アイコンエディタのプログラムリスト

```

#include <stdio.h>
#include <conio.h>
#include <graphics.h>
#include <bios98.h>
#include <stdlib.h>

#define XMAX 40
#define YMAX 40
#define COLOR_TBLX (XMAX)-8+40+20
#define COLOR_TBLX (YMAX)-8-(YMAX)-2-20
#define FLH_X (XMAX)-8+40
#define FLH_Y (YMAX)-8-YMAX-16
#define PAT_X (XMAX)-8+40
#define PAT_Y 1
#define SPACE "

typedef struct PAT { int x;
                    int y;
} PATXY;
PATXY pat[50];

struct KEY_VECTOR {
    int vx;
    int vy;
};
static struct KEY_VECTOR v[] = { { -1, 1 }, { 0, 1 }, { 1, 1 },
                                  { -1, 0 }, { 0, 0 }, { 1, 0 },
                                  { -1, -1 }, { 0, -1 }, { 1, -1 } };

static char path[20] = "c:\\data\\", path2[20], name[5];

static char path_gd[20] = "b:\\x\\c";

static int col_tbl[] = { BLACK, BLUE, RED, MAGENTA, GREEN, CYAN, YELLOW,
                        WHITE,
                        LIGHTRED, LIGHTGREEN, LIGHTBLUE, LIGHTCYAN,
                        LIGHTMAGENTA, BROWN
                        };

int sc[YMAX][XMAX];

void paint( int x, int y, int c1, int c2 )
{
    setfillstyle( SOLID_FILL, c1 );
    floodfill( x, y, c2 );
}

void menu()
{
    typedef struct col {
        int x;
        int y;
        char *message;
        int attr;
    } COLOR;
    static COLOR tmp[] = { { 45, 21, "Z・X... バレット移動", T_CYAN },
                           { 45, 22, "S・E... キャラの編集", T_CYAN },
                           { 45, 23, "Q... 画面クリア", T_RED },
                           { 5, 21, "2・4・6・8... カーソル1の移動", T_CYAN },
                           { 5, 22, "H・J・K・L... カーソル2の移動", T_CYAN },
                           { 5, 23, "メッセージ:", T_YELLOW }
    };

    int i;
    for(i=0; i<6; ++i) {
        textcolor( tmp[i].attr );
        gotoxy( tmp[i].x, tmp[i].y );
        cprintf( "%s", tmp[i].message );
    }
    textcolor( T_WHITE );
}

void flame()

```

```

{
int i,i2;
for(i=0;i<YMAX;++i) for(i2=0;i2<XMAX;++i2) sc[i][i2]=0;
setcolor( LIGHTGRAY );
rectangle(0,0,XMAX*8,YMAX*8);
paint(1,1,BLACK,LIGHTGRAY );
setcolor( DARKGRAY );
rectangle( 0,0,XMAX*8,YMAX*8 );
paint( FLM_X+10,FLM_Y+10,BLACK,DARKGRAY );
setcolor( DARKGRAY );
    rectangle( 0,0 ,(XMAX*8),(YMAX*8) ); /* 升目を書く */
    for(i=8;i<XMAX*8;i+=8) line(i,1,i,YMAX*8-1);
    for(i=8;i<YMAX*8;i+=8) line(1,i,XMAX*8-1,i);
}

void scr_set()
{
int gd=DETECT,gm,i,i2,n=0; /* グラフィックスの初期化 */
setnewdriver( "PC98",detect98 );
initgraph( &gd,&gm,path_gd );
for(i=0;i<YMAX;++i) for(i2=0;i2<XMAX;++i2) sc[i][i2]=0;
setcolor( DARKGRAY );
    rectangle( 0,0 ,(XMAX*8),(YMAX*8) ); /* 升目を書く */
    for(i=8;i<XMAX*8;i+=8) line(i,1,i,YMAX*8-1);
    for(i=8;i<YMAX*8;i+=8) line(1,i,XMAX*8-1,i);
for(i2=0;i2<YMAX*5+10;i2+=YMAX*2) /* キャラ用の枠を書く */
    for(i=0;i<XMAX*6+12;i+=XMAX*2){
        pat[n].x=PAT_X+i;pat[n].y=PAT_Y+i2;n++;
        rectangle( PAT_X+i-1,PAT_Y+i2-1,PAT_X+i+XMAX,PAT_Y+i2+YMAX );
    }
for(i=1;i<5;i++) rectangle( FLM_X-i-1,FLM_Y-i-1,FLM_X+XMAX+i,FLM_Y+YMAX+i);
n=0; /* パレットを描く */
rectangle(COLOR_TBLX,COLOR_TBLX+15*14,COLOR_TBLX+15,COLOR_TBLX+15);
for(i=COLOR_TBLX+15;i<COLOR_TBLX+15*14+15;i+=15){
    line(i,COLOR_TBLX+15,COLOR_TBLX+15*14+15,i);
    paint(i-2,COLOR_TBLX+15,col_tbl[n++],DARKGRAY);
}
setcolor( LIGHTGREEN );
settextstyle( 3,HORIZ_DIR,4 );
outtextxy(FLM_X+90,FLM_Y,"EyeCon98");
menu();
}

void pat_set(int pc) /* キャラクターの登録 */
{
int i,i2;
char c;
FILE *fp;
gotoxy( 17,23 );
cprintf("よろしい?");
c=getch();
if ( c=='y' || c=='Y' ){
    gotoxy(17,23);
    cprintf("登録中!!");
    itoa( pc.name,10 );strcpy( path2,path );strcat( path2,name );
    if ((fp=fopen(path2,"w"))==NULL){ gotoxy(17,23);
        cprintf("CAN'T OPEN FILE!");
        getch();
        gotoxy(17,23);cprintf("SPACE");
    }
}
else {
    for(i=0;i<YMAX;i++){
        for(i2=0;i2<XMAX;i2++){
            putpixel(pat[pc].x+i2,pat[pc].y+i,col_tbl[sc[i][i2]]);
            fputc( sc[i][i2]+'0',fp );
        }
        fputc( '\n',fp );
    }
    fclose(fp);
}
gotoxy( 17,23 );
cprintf(" ");
}

```

```

)

void scr_load() /* キャラクターのロード */
{
    int i,i2,pc,c;
    FILE *fp;
    for ( pc=0 ; pc<30 ; ++pc ){
        itoa( pc,name,10 );strcpy( path2,path );strcat( path2,name );
        if ((fp=fopen(path2,"r"))!=NULL){
            for(i=0;i<YMAX;i++){
                for(i2=0;i2<XMAX;i2++){
                    do {
                        c=fgetc( fp );
                    } while( c=='\n' );
                    putpixel( pat[pc].x+i2,pat[pc].y+i,col_tbl[c-'0'] );
                }
            }
            fclose(fp);
        }
    }
}

void edit( int pc ) /* キャラクターのエディット */
{
    int i,i2,c;
    FILE *fp;
    itoa( pc,name,10 );strcpy( path2,path );strcat( path2,name );
    if ((fp=fopen(path2,"r")) == NULL ){
        gotoxy( 17,23 );cprintf("空っぽだよ。");
        getch();
        gotoxy( 17,23 );cprintf(" ");
    } else {
        for(i=0;i<YMAX;++i){
            for(i2=0;i2<XMAX;++i2){
                do {
                    sc[i][i2]=fgetc( fp );
                } while( sc[i][i2]!='\n' );
                sc[i][i2] -= '0';
                switch( sc[i][i2] ){
                    case 0:
                        putpixel( FLM_X+i2,FLM_Y+i,BLACK);
                        paint(i2*8+1,i*8+1,BLACK,DARKGRAY);
                        break;
                    default:
                        putpixel( FLM_X+i2,FLM_Y+i,col_tbl[sc[i][i2]]);
                        paint(i2*8+1,i*8+1,col_tbl[sc[i][i2]],DARKGRAY);
                        break;
                }
            }
        }
        fclose(fp);
    }
}

void replace( void )
{
    char k;
    int c1,c2,i,i2;
    gotoxy( 17,23 );cprintf("何色を?");
    c1=color_select();
    gotoxy( 17,23 );cprintf("何色に?");
    c2=color_select();
    gotoxy( 17,23 );cprintf("よろしい?");
    k=getch();
    if ( k == 'y' ;; k == 'Y' ){
        gotoxy( 17,23 );
        cprintf("置換中!!");
        for(i=0;i<YMAX;i++){
            for(i2=0;i2<XMAX;i2++){
                if ( sc[i][i2]==c1 )
                {
                    putpixel( FLM_X+i2,FLM_Y+i,col_tbl[c2]);
                    paint(i2*8+1,i*8+1,col_tbl[c2],DARKGRAY);
                    sc[i][i2]=c2;
                }
            }
        }
    }
}

```



```

    )
    gotoxy( 17,23 );
    cprintf(" ");
    putchar( 7 );
}

int color_select()
{
    char k;
    int c=0;
    while( 1 )
    {
        setcolor( WHITE );
        rectangle(COLOR_TBLX+c*15,COLOR_TBLY,COLOR_TBLX+c*15+15,COLOR_TBLY+25);
        k=getch();
        if ( k==' ' ) break;
        if ( k=='z' && c>0 ) { setcolor( DARKGRAY );
            rectangle(COLOR_TBLX+c*15,COLOR_TBLY,COLOR_TBLX+c*15+15,COLOR_TBLY+25)
        ;
            c--;
        }
        if ( k=='x' && c<13 ) { setcolor( DARKGRAY );
            rectangle(COLOR_TBLX+c*15,COLOR_TBLY,COLOR_TBLX+c*15+15,COLOR_TBLY+25)
        ;
            c++;
        }
    }
    setcolor( DARKGRAY );
    rectangle(COLOR_TBLX+c*15,COLOR_TBLY,COLOR_TBLX+c*15+15,COLOR_TBLY+25);
    return ( c );
}

void main(void) /* メイン関数 */
{
    int x=0,y=0,px=0,py=0,pc=0,
        vx,vy,vpx,vpy,
        i,i2,k,c=0,tmp;

    scr_set(); /* スクリーンのスタンバイ */
    scr_load();

    while(1)
    {
        setcolor( WHITE );
        rectangle(COLOR_TBLX+c*15,COLOR_TBLY,COLOR_TBLX+c*15+15,COLOR_TBLY+25);
        rectangle(pat[pc].x-1,pat[pc].y-1, pat[pc].x+XMAX,pat[pc].y+YMAX);
        rectangle( x*8,y*8,x*8+8,y*8+8 );
        k=getch();
        setcolor( DARKGRAY );
        rectangle( x*8,y*8,x*8+8,y*8+8 );
        if ( (k-'0' > 0) && (k-'0' < 10) ) {
            vx= v[k-'1'].vx;
            vy= v[k-'1'].vy;
        }
        vpx=0;vpy=0;
        if ( k=='k' && py>0 ) vpy= -1;
        if ( k=='j' && py<4 ) vpy= 1;
        if ( k=='h' && px>0 ) vpx= -1;
        if ( k=='l' && px<5 ) vpx= 1;
        if ( vpx || vpy ) { setcolor( DARKGRAY );
            rectangle(pat[pc].x-1,pat[pc].y-1,pat[pc].x+XMAX,pat[pc].y+YMA
X);
            px += vpx;py += vpy;vx=vy=0;
            pc= 6*py+px;
        }
        if ( k=='z' && c>0 ) {
            setcolor( DARKGRAY );
            rectangle(COLOR_TBLX+c*15,COLOR_TBLY,COLOR_TBLX+c*15+15,COLOR_TBLY+25)
        ;
            c--;vx=vy=0;
        }
        if ( k=='x' && c<13 ) {
            setcolor( DARKGRAY );

```

```

rectangle(COLOR_TBLX+c*15,COLOR_TBLY,COLOR_TBLX+c*15+15,COLOR_TBLY+25)
;
c++;vx=vy=0;
)
if ( k==' ' ) {
    putpixel(FLM_X+x,FLM_Y+y,col_tbl[c]);
    paint(x*8+1,y*8+1,col_tbl[c],DARKGRAY);
    sc[y][x]=c;
}
if ( k==27 ) {
    closegraph();
    clrscr();
    exit(0);
}
if ( k=='c' ) {
    putpixel(FLM_X+x,FLM_Y+y,BLACK);
    paint(x*8+1,y*8+1,BLACK,DARKGRAY);
    sc[y][x]=0;
}
if ( k=='q' ) {
    gotoxy( 17,23 );
    cprintf("よろしい?");
    tmp=getch();
    if ( tmp == 'y' || tmp == 'Y' ) {
        flame();x=y=0;vx=vy=0;
    }
    gotoxy( 17,23 );
    cprintf("      ");
}
if ( k=='s' ) pat_set( pc );
if ( k=='e' ) edit( pc );
if ( k=='r' ) {
    setcolor( DARKGRAY );
    rectangle(COLOR_TBLX+c*15,COLOR_TBLY,COLOR_TBLX+c*15+15,COLOR_TBLY+25)
;
    replace();
}
if (x+vx<0 || x+vx>XMAX-1 || y+vy<0 || y+vy>YMAX-1 ) vx=vy=0;
x+=vx;y+=vy;
}

```

付録E. 通信ソフト「飛火野」用メイクファイル

```
FILES1=tobihino.obj ed.obj set.obj ftrans.obj hist.obj hand.obj
FILES2=conn.obj ether.obj

test.exe: $(FILES1) $(FILES2)
    tcc -K -C -O -mh -Lb:%tc%lib $(FILES1) $(FILES2) graphics.lib

tobihino.obj: tobihino.c nfire.h fall.h
    tcc -K -C -c -O -mh -lb:%tc%include tobihino.c

ed.obj: ed.c nfire.h fall.h
    tcc -K -C -c -O -mh -lb:%tc%include ed.c

ftrans.obj: ftrans.c nfire.h fall.h
    tcc -K -C -c -O -mh -lb:%tc%include ftrans.c

hist.obj: hist.c nfire.h fall.h
    tcc -K -C -c -O -mh -lb:%tc%include hist.c

set.obj: set.c nfire.h fall.h
    tcc -K -C -c -O -mh -lb:%tc%include set.c

hand.obj: hand.c nfire.h fall.h
    tcc -K -C -c -O -mh -lb:%tc%include hand.c graphics.lib

conn.obj: conn.c nfire.h fall.h
    tcc -K -C -c -O -mh -lb:%tc%include conn.c

ether.obj: ether.c nfire.h fall.h
    tcc -K -C -c -O -mh -lb:%tc%include ether.c
```

付録F. 通信ソフト「飛火野」用外部変数宣言ファイル

```

/*
.....
.
.
.
外部変数宣言
.
.
.....
*/
extern struct CENTERPARA cp; /* 接続先パラメータ */
extern struct COLOR co; /* カラーとリバース */
extern struct CENTERNAME cn; /* センター・パラメータ */
extern struct UT100 vt; /* VT100コード */
extern struct FUN_CUR_KEY funkey[2];

extern int CDsk; /* カレント・ディスク */
extern char CDir[]; /* カレント・ディレクトリ */
extern int MAXDSK; /* ドライブ数 */
extern int DskErrFlag; /* ディスクのエラーフラグ */
extern int MX,MY; /* マウス座標 MX(MAX=80),MY(MAX=24) */
extern int hist_xp; /* 履歴X座標 */
extern int hist_yp; /* 履歴Y座標 */
extern int xonoff_flag; /* Xon/offフラグ */
extern int AtribNum; /* 画面アトリビュート */
extern int CrtAtrib[]; /* キャラクタのアトリビュート */
extern char XfileNam[]; /* 転送処理ファイル名 */
extern char MultiFile[10][DATA_WD]; /* マルチファイル名バッファ */
extern int ScrollTp; /* スクロールの種類 */
extern int ScrollBm; /* スクロールのボトム */
extern int EditFlag; /* メモ編集オプション */
extern int EdCopyFlag; /* 履歴から便箋へのコピー */
extern char EscBuff[]; /* エスケープバッファ */
extern int ConnectFlag; /* 接続フラグ */
extern int ModemFlag; /* モデム接続フラグ */
extern int TimerFlag; /* タイマー表示フラグ */
extern int TimerSet; /* タイマーセットフラグ */
extern unsigned int TimeOver; /* タイムオーバーフラグ 1分 */
extern int connect_x; /* 通信画面X座標 */
extern int connect_y; /* 通信画面Y座標 */
extern int hist_line; /* 表示ライン番号 */
extern int hist_top; /* 行範囲トップ */
extern int hist_end; /* 行範囲エンド */
extern int hist_flag; /* 行範囲表示フラグ */
extern int k,
MS_X,MS_Y, /* マウス表示位置 */
MS_X2,MS_Y2, /* (旧表示位置) */
MS_LB,MS_RB, /* マウスのボタン状態 */
MS_LB2,MS_RB2; /* (旧状態) */
extern char ENU[]; /* このソフトを動かしているディレクトリ */
extern int rcvp,rcvpmax,rcvlen,rcvflag;
extern int EtherFlag,EtherFin,LagFlag;
extern char rbuf[],sbuf[];

```

付録 G. 通信ソフト「飛火野」用ヘッダファイル

```

/*
.....
*               NFIRE.H               *
*   使用default値の設定および構造体の定義   *
*.....
*/

/* ASCIIコード */

#define BELL 0x07    /* Bell */
#define BS  0x08     /* Back Space */
#define TAB 0x09     /* Tab */
#define LF  0x0A     /* Line Feed */
#define VT  0x0B     /* Vertical Tab */
#define FF  0x0C     /* Home Clear */
#define CR  0x0D     /* Carriage Return */
#define ESC 0x1B     /* Escape */
#define D1  0x11     /* ctrl-Q */
#define D3  0x13     /* ctrl-S */
#define DL  0x7F     /* Delete */

/* PC98スキャンコード */

#define RUP 0x36
#define RDOWN 0x37
#define INS 0x38
#define DEL 0x39

#define UP 0x3a
#define LEFT 0x3b
#define RIGHT 0x3c
#define DOWN 0x3d

#define HOME 0x3e
#define HELP 0x3f

#define FUN1 0x62
#define FUN2 0x63
#define FUN3 0x64
#define FUN4 0x65
#define FUN5 0x66
#define FUN6 0x67
#define FUN7 0x68
#define FUN8 0x69
#define FUN9 0x6a
#define FUN10 0x6b

#define YES 0
#define FALSE -1
#define ETC 1

struct COLOR{
    int title;    /* カラーの構造体 */
    int box;      /* ウィンドウ・タイトル */
    int oh;       /* ボックス・カラー */
    int number;   /* ウィンドウ内キャラクタ・カラー */
    int history;  /* コマンドNoカラー */
    int editor;   /* 履歴キャラクタ・カラー */
    int half;     /* 便せんキャラクタ・カラー */
    int send;     /* 半2重キャラクタ・カラー */
    int recv;     /* 無手順送信文字キャラクタ・カラー */
    int control;  /* 無手順受信文字キャラクタ・カラー */
    int error;    /* 制御文字キャラクタ・カラー */
    int itime;    /* エラー・キャラクタ・カラー */
    int ntime;    /* 接続時間カラー */
    int ltime;    /* 現在時刻カラー */
    int kakuninn; /* ラップタイムカラー */
    int mouse;    /* 確認カラー */
    int R_title;  /* マウス・カラー */
    int R_box;    /* ここからリバーデータ */
    int R_title;  /* ウィンドウ・タイトル */
    int R_box;    /* ボックス・カラー */

```

```

    int R_half;      /* 半2重キャラクタ・カラー */
    int R_send;      /* 無手順送信文字キャラクタ・カラー */
    int R_recv;      /* 無手順受信文字キャラクタ・カラー */
    int R_control;    /* 制御文字キャラクタ・カラー */
    int R_error;      /* エラー・キャラクタ・カラー */
};

#define DATA_WD 41 /* 入力データ最大幅 */

struct CENTERNAME{
    int CenterNo;
    char NAME(40)(DATA_WD);
};

struct CENTERPARA ( /* パラメータ構造体 */
    int sio;          /* SIOのチャネル */
    int boudrate;     /* ボーレート */
    int length;       /* データ長 */
    int parity;       /* パリティ */
    int method;       /* 全/半2重 */
    int stopbit;      /* ストップ・ビット */
    int xonoff;       /* Xon/Xoff */
    int code;         /* 漢字コード */
    int recer;        /* c r 受信処理 */
    int snder;        /* c r 送信処理 */
    int reclf;        /* l f 受信処理 */
    int sndlf;        /* l f 送信処理 */
    int mnp;          /* MNP設定 */
    int ringno;       /* 電話回数 */
    int ringsec;      /* 話中待時間 */
    int ltype;        /* 接続回線のタイプ */
    char telno(DATA_WD); /* 電話番号 */
    char conname(DATA_WD); /* 接続先名 */
);

struct SIOPARA( /* SIOパラメータの構造体 */
    int channel;
    int boudrate;
    int databit;
    int parity;
    int stopbit;
    int xonoff;
    int time; /* unit 500ms */
);

struct FUN_CUR_KEY( /* ファンクションキーと制御キー */
    char func(20)(16);
    char curs(11)(6);
);

struct UT100(
    int KeyPad; /* UT100のキーパッド */
    int CursorKey; /* UT100カーソルキーモード */
    int Origin; /* UT100カーソルオリジンモード */
    int AutoLap; /* UT100オートラップフラグ */
    int CODE; /* 漢字コード */
    int SIS0; /* シフトイン/アウト */
);

#define NOJIS 0 /* 漢字コード無し */
#define NEWJIS 1 /* 新漢字コード */
#define OLDJIS 2 /* 旧漢字コード */
#define NECJIS 3 /* NEC漢字コード */

#define MODEM 2
#define CENTER 3
#define SETTEL 4
#define END 6
#define MSDOS 5
#define OPEN 20
#define CONNECT 7
#define EDITOR 1
#define HISTORY 9

```

```

struct DIRFILE(
    char    name[5][13];    /* ファイル名セーブエリア */
);

#define HISTLIN 720          /* 履歴バッファのライン数 */
#define BUFLIN 100          /* 便箋バッファのライン数 */
#define BUFNUM 83*BUFLIN    /* 便箋バッファの大きさ */
#define SBUFSIZE 10
#define RBUFSIZE (7*1514)

#define ICON_XMAX (640-160)
#define ICON_YMAX (400-32-80-40-16)
#define MAX_PAGE 100        /* オープン出来るウィンドウの枚数 */

#define MERGIN 6             /* データNO表示用 */
#define NUM_MERGIN 4         /* データNOを表示しないとき */

#define MS_LX 200            /* マウスの初期座標 */
#define MS_LY 104
#define MS_PAT_NO 9          /* マウスパターンの番号 */
#define ICON_MAX_NO 11       /* 全登録アイコン数 */
#define COM_NO 9             /* 全機能数 */

#define WD_OPEN 0            /* ウィンドウ関数使用時のdefault値 */
#define WD_CLOSE 0,1,0,0
#define WD_CLOSEU 1
#define WD_REMAIN 2
#define FW_OPEN 3
#define FW_REMAIN 4

#define RET_OFF 0
#define RET_ON 1
#define SND 0
#define RCU 1
#define SIZ 2

struct ICON_POS (
    int x;
    int y;
    int num;
    int com;
);

struct WD_PARA (
    int x1,y1;
    int x2,y2;
    void *adr;
);

typedef struct WD (
    int x;                /* ウィンドウ表示座標 */
    int y;
    int wx;                /* ウィンドウの大きさ */
    int wy;
    int col1;              /* ウィンドウ フレーム 表示色 */
    int col2;              /* ウィンドウ タイトルの色 */
    int col3;              /* ウィンドウ 表示文字列の色 */
    int l;                 /* ウィンドウ カーソル座標(変数) (from 0) */
    char *title;           /* ウィンドウ タイトル */
    int inp_mode;          /* 0..内容更新可能タイプ
                          /* 1..表示のみのウィンドウ
                          /* 2..内容更新不能タイプ(選択のみ)
    int cmax;              /* カーソル上限(この値には、ならない)(from 0)
    int cmin;              /* カーソル下限(この値もで、なれる) (from 0)
    int button;            /* 0..ESC機能をサポート(ボタンあり)
                          /* 1..上下左右ESC機能をサポート
                          /* 2..メッセージのみ表示する(ボタンないがESC可能)
    int cursor_on;         /* カーソルを表示する? 1..YES 0..NO
    int lineno_on;         /* 0..ラインNOを表示しない
                          /* 1..ラインNOを表示する
    int inp_line;          /* インプット行の指定(from 0)
                          /* -1の時全ての行で入力可能
                          /* ディレクトリ表示の際は横方向に
                          /* 表示するファイル数の数
) WINDOW;

```

```

.....
*
*   I n e t B I O S 用   イ ン ク ル ード ファ イ ル
*
.....

#ifndef _INETBIOS_
#define _INETBIOS_

.....
* Network Address Formats *
.....

/*
* ethernet address
*/
struct ether_addr {
    unsigned char ea_chars[6];
};

/*
* machine address
*/
struct ip_addr {
    union {
        unsigned char ia_chars[4];
        unsigned long ia_long;
    } ia_un;
#define ia_l ia_un.ia_long
#define ia_c ia_un.ia_chars
};

/*
* internet address -- identifies an endpoint
*/
struct inet_addr {
    unsigned short ina_port;
    struct ip_addr ina_iaddr;
};

.....
* Module Description Table *
.....

/*
* Common part of module description table
*/
typedef struct _moduletable {
    unsigned char JmpNimonic;
    unsigned char OldVector[4]; /* JMP instruction to previous handler */
    unsigned char Padding;
    struct _moduletable far *NextTable; /* link to next module table */
    char ModuleMagic[4]; /* Here-is-Inet/MLDI-table indicator */
    char ModuleType[4]; /* module type; "INET" or "MLDI" */
    char ModuleName[8]; /* implementation name */
    unsigned char SpecVer; /* specification version */
#define INETVERS 0x20 /* InetBIOS = 2.0 */
#define MLDIVERS 0x10 /* MLDI = 1.0 */
    unsigned char ImplVer; /* implementation version */
    } ModuleTable;

/*
* InetBIOS module description table
*/

typedef struct {
    ModuleTable Common; /* common part of module table */
    unsigned char Protocol; /* protocol */
#define TCPTOID 2 /* TCP/IP */
    unsigned char AddrLen; /* internet address length */
#define TCPADDRLEN 8 /* TCP/IP */
    void (far *EntryPoint)(); /* InetBIOS entry point (for use by protocol manage
r) */
    short MaxDgramSend; /* maximum size of datagram that can be sent */

```



```

    short MaxDgramRecv;          /* maximum size of datagram that can be received */
    unsigned char Services;      /* supported services */
#define TCPSERVICES 0x11        /* TCP/IP; datagram and stream */
    unsigned char NumAdapters;   /* # of adapters present */
    char NumEndpoints;          /* maximum # of open endpoints */
    char NumEPReqs;             /* maximum # of requests per endpoint */
    char NumConnections;        /* maximum # of connections per endpoints */
    char NumConnReqs;           /* maximum # of requests per connections */
    char ModuleIdent[1];         /* identification string or copyright notice;... */

    /* ... zero-terminated variable string */

} InetTable;

#define UNLIMITED -1           /* value -1 for MaxDgramXXX and NumXXX above means
                               there is no fixed limited as far as memory space allows */

/*
 * MLDI module description table (applicable only for portable tcp version)
 */

struct mlditable {
    ModuleTable common;         /* common part of module table */
    void (far *mldi_init)();    /* driver entry points */
    void (far *mldi_send)();
    void (far *mldi_pass)();
    void (far *mldi_stop)();
    void (far *mldi_get_paddr)();
    void (far *mldi_get_stat)();
    unsigned short media;       /* media type */
#define MEDIAETHER 1           /* ethernet */
#define MEDIATOKEN 2           /* IBM token ring */
#define MEDIAAPPLE 3           /* Apple LocalTalk */
#define MEDIAOMNT 4            /* Omnet */
#define MEDIASLIP 5            /* serial IP */
    unsigned short paddrLen;    /* physical address length */
#define ETHERADDRLEN 6         /* ethernet */
    unsigned short minframe;    /* minimum allowable frames size */
#define ETHERMINLEN 64
    unsigned short maxframe;    /* maximum frame size */
#define ETHERMAXLEN 1518
};

/*.....
 * InetBIOS-related definitions
 *.....*/

/*
 * InetBIOS transport control block
 */

typedef struct {
    unsigned short Length;      /* length of endpoint address; 8 for TCP/IP */
    struct inet_addr InetAddr;  /* Internet address */
    char NoetUsed[8];
} EPAddress;

typedef struct {
    char Reserved[16];          /* reserved for private use by InetBIOS module */
    char Command;               /* command code */
    char Result;                /* result code */
    void (far *Async)();        /* asynchronous notification routine */
    char Complete;              /* no-wait command status; complete=0, pending=-1 */
    unsigned char Adapter;      /* adapter number */
    unsigned char Service;      /* service type */
#define SRU_DG 0                /* datagram */
#define SRU_STRM 5              /* stream */
    char Padding;               /* not used */
    unsigned short Flags;       /* command option flags */
#define F_SHUTDOWN 2            /* shutdown on Send (stream) */
#define F_RARP 1                /* RARP on ResetAdapter */
#define F_ICMPMASK 1            /* ICMP subnet mask query on SetSubnetMask */
    unsigned short EndPoint;    /* endpoint identifier (EID) */

```

```

    unsigned short Connection; /* connection identifier (CID) */
    char far *PrmBuf;          /* primary buffer address */
    unsigned short PrmLen;     /* primary buffer length */
    char far *SndBuf;          /* secondary buffer address (used by Send) */
    unsigned short SndLen;     /* secondary buffer length */
    EPAddress LocAddr;         /* local endpoint address */
    EPAddress RemAddr;         /* remote endpoint address */
} TCB;

/*
 * InetBIOS command codes
 */

/* service command */
#define INETB_OPEN 1 /* create a local endpoint */
#define INETB_CLOSE 2 /* destroy an open endpoint */
#define INETB_CALL 3 /* create a connection */
#define INETB_LISTEN 4 /* wait for a connection request */
#define INETB_HANGUP 5 /* destroy an active connection */
#define INETB_SEND 6 /* send information to the peer */
#define INETB_CAPACITY 7 /* return how much data is acceptable */
#define INETB_RECV 8 /* receive information from the peer */
#define INETB_PEEK 9 /* return the number of bytes received */

/* administration command */
#define INETB_ADAPTERINFO 0x41 /* get information about a network adapter */
#define INETB_RESET 0x42 /* reset network adapter */
#define INETB_ADDROUTE 0x43 /* add a piece of routing information */
#define INETB_DELROUTE 0x44 /* delete a piece of routing information */
#define INETB_ROUTEINFO 0x45 /* returns the contents of the routing table */
#define INETB_SETSUBNET 0x46 /* set subnet mask */

/*
 * InetBIOS result codes
 *
 * The InetBIOS result codes are a subset of BSD UNIX error codes.
 * It is recommended that the implementors use these error codes as possible.
 * The implementor must always return the result code if it is explicitly
 * mentioned in the command description.
 */

#define NoError 0 /* successful */
#define EIO 5 /* any error */
#define ENOMEM 12 /* no memory or buffer */
#define ENODEV 19 /* no such adapter */
#define EINVAL 22 /* invalid command or arguments */
#define EMFILE 24 /* too many open endpoints or connections */
#define EMSGSIZE 40 /* message too long */
#define EOPNOTSUPP 45 /* command or option not supported */
#define EADDRINUSE 48 /* address already used */
#define ENETDOWN 50 /* network is down */
#define EUNREACHABLE 51 /* network, machine or endpoint is not reachable */
#define ECONNABORTED 53 /* connection broken locally */
#define ECONNRESET 54 /* connection broken by peer */
#define ESHUTDOWN 58 /* connection carries no more data in that direction */
#define ETIMEOUT 60 /* operation timed out */
#define ECONNREFUSED 61 /* connection request refused */

/*
 * InetBIOS adapter information & routing information
 */

typedef struct {
    unsigned short Length;
    struct ether_addr Machine;
    char Dummy[8];
} EtherAddr;

typedef struct {
    char CardName[8]; /* card name */
    EPAddress ProtAddr; /* protocol address of this adapter */
    EtherAddr PhysAddr; /* physical address of this adapter */
    unsigned short MediaType; /* media type */

```

```

        unsigned short FrameLength;      /* maximum frame length */
        unsigned long NumSends;          /* number of frames sent */
        unsigned long NumSendErrs;       /* number of errors in sending frames */
        unsigned long NumRecvs;          /* number of frames received */
        unsigned long NumRecvErrs;       /* number of errors in receiving frames */
        unsigned long NumCollis;         /* number of collisions */
        unsigned long SubnetMask;        /* subnet mask */
    } AdapterInfo;

typedef struct {
    unsigned char Network[4];            /* destination network or machine address */
    unsigned char Gateway[4];           /* gateway for relaying datagrams for that destina
tion */
} RouteInfo;

/* Service field in InetBIOS module table */
#define DATAGRAM_SUPPORTED 1
#define STREAM_SUPPORTED 0x10

/* default module table chain vector */
#define DFLTVECT 0x7e

extern int _inetvect;

/*
 * MLDI related definitions
 */

/*
 * buffer list
 */
struct bufpair {
    unsigned short bp_len;              /* length of buffer */
    char far *bp_buf;                  /* pointer to buffer */
};

struct buflist {
    unsigned short bl_stat;             /* status indicator */
    struct bufpair bl_pairs[];          /* one or more buffer lists appears */
#define bl_len(n)    bl_pairs[(n)].bp_len
#define bl_buf(n)    bl_pairs[(n)].bp_buf
};

/*
 * statistics
 */

struct netstat {
    unsigned long ipkts;
    unsigned long ierrs;
    unsigned long opkts;
    unsigned long oerrs;
    unsigned long collis;
};

/*
 * protocol-side entry points
 */
struct protoentries {
    void (far *done)();
    void (far *peek)();
};

/*
 * buffer list status codes
 */
#define MS_SUCCESS 0    /* no error */
#define MS_ILLEGAL 1    /* illegal buffer list */
#define MS_NETDOWN 2    /* network is down */
#define MS_CANTSEND 3   /* cannot send frame */

/*
 * command return codes
 */

```

- Several commands may return MR_SUCCESS and MR_ERROR.
- Only SEND command may return either of all return codes below.
-
- Meaning of SEND return codes are:
 -
 - MR_SUCCESS: frame successfully sent; protocol can immediately free buffers
driver will not call DONE entry.
 - MR_ERROR: driver failed to send frame; protocol must free buffers now
 - MR_CALLDONE: frame kept by driver; driver will call DONE entry later,
at that time, protocol can free buffers
 -
-

```
#define MR_SUCCESS 0
#define MR_ERROR -1
#define MR_CALLDONE 1
#define MR_QUEUED 1

#endif
```

```

#define OpenMass3      " Copyright(c) 1993.02      成田 研究室      "

static int STOPOFF;      /* ストップ割り込みエントリ オフセット */
static int STOPSEG;      /* ストップ割り込みエントリ セグメント */
static int EQUIP;        /* 接続装置 */
static int CLOCK;        /* CPUクロック */

extern int Key98Flag;
extern unsigned int      _delaybase;      /* delay関数の値 */
extern unsigned int      _stklen=50000;    /* スタック領域 50KB */

/*
.....
*          飛火野      初期設定用関数群          *
.....
*/
/*
-----
                        最上部へのプログラム名表示
-----
*/
void      openmessage()
{
    window(1,1,80,25);
    gotoxy(1,1);
    textreverse(REVERSE);
    textcolor(co.box);
    cprintf(OpenMass1);
    textcolor(co.title);
    cprintf(OpenMass2);
    textcolor(co.box);
    cprintf(OpenMass3);
    textreverse( NOREVERSE );
    window(1,1,80,25);
}
/*
-----
                        最下位行へ接続先表示
-----
*/
void      dspconnectname()
{
    int x,i,a,b;
    a=wherex();b=wherey();
    window(1,1,80,25);
    textcolor( T_WHITE );
    gotoxy(1,25);
    textreverse( REVERSE );cprintf("Edit      ");
    textreverse( NOREVERSE );cprintf(" ");
    textreverse( REVERSE );cprintf("Modem  ");
    textreverse( NOREVERSE );cprintf(" ");
    textreverse( REVERSE );cprintf("LAN      ");
    textreverse( NOREVERSE );cprintf(" ");
    textreverse( REVERSE );cprintf("Touroku");
    textreverse( NOREVERSE );cprintf(" ");
    textreverse( REVERSE );cprintf("Ms-Dos  ");
    textreverse( NOREVERSE );cprintf(" ");
    textreverse( REVERSE );cprintf("History");
    textreverse( NOREVERSE );cprintf(" ");
    textreverse( REVERSE );cprintf("Exit      ");
    textreverse( NOREVERSE );cprintf(" ");
    textcolor( co.box );
    textreverse( REVERSE );cprintf(" 接続先 %14s",cp.conname);
    x=wherex();
    for( i=x;i<79;i++ ) cprintf(" ");
    textreverse( NOREVERSE );
    window(1,1,80,24);
    gotoxy( a,b );
}
/*
-----
                        ディスクエラー割り込み処理
-----
*/

```

```

int errhandler()
{
    errordisp("ドライブの準備ができていません。");
    DskErrFlag=-1;
    return 0;
}
/*
-----
                        disk error
-----
*/
void setdiskerr()
{
    harderr(errhandler);
}
/*
-----
                        終了処理
-----
*/
void interrupt endfunc()
{
    char NAME[80];
    icon_close();
    closegraph();
    mouseclr();
    siostop();
    setdisk(CDsk);
    chdir(CDir);
    window(1,1,80,25);
    clrscr();
    window(1,1,80,24);
    textcursor(DISP_CURSOR);
    poke(0,0x0010,STOPOFF);
    poke(0,0x001a,STOPSEG);
    putfunkey();
    sprintf( NAME, "%s%CLFIRE", ENU );
    system(NAME);
    normvideo();
    printf("%c[>11", ESC);
    exit(0);
}
/*
.....
.
.
.
.
.
.....
*/
void main( void )
{
    int gd=DETECT,gm;
    char NAME[80],*nm;
    if(setCharBuff()!=YES){
        printf("メモリが足りません。");
        exit(0);
    }
    textmode(UL8025);
    getcwd(CDir,80);
    nm=getenv( "TOBI" );
    if ( nm==NULL ) strcpy( ENU,CDir ); else strcpy( ENU ,nm );
    readenvfile();
    clrcolor();
    edbufnew();
    edinit();
    setfront();
    sprintf( NAME, "%s%TSETFONT", ENU );
    if(system(NAME)!=0){
        resetfront();
        textcursor(DISP_CURSOR);
        printf("外字登録ファイルがありません。%r%n");
        exit(0);
    }
    resetfront();
}

```

/* メニュークローズ */
 /* グラフィックス終了 */
 /* マウス初期化 */
 /* 回線の強制切断 */
 /* 元のディレクトリに戻る */
 /* ウィンドウの初期化 */
 /* ファンクションの復元 */
 /* 終了 */

「飛火野」メイン

/* ヒストリ用のバッファ獲得 */
 /* 環境値の読みだし */
 /* カラーのセット */
 /* エディットバッファのクリア */
 /* 外字登録 */

```

STOPOFF=peek(0,0x0018);      /* STOP 割り込みエントリ */
STOPSEG=peek(0,0x001a);
EQUIP=bios98equip();
CLOCK=(0x000f & (EQUIP>>4));
switch(CLOCK){
    case 0: _delaybase=237; break; /* 8086 5MHz */
    case 1: _delaybase=148; break; /* U30 8MHz */
    case 2: _delaybase=76; break; /* 80286 10MHz */
    case 3: _delaybase=62; break; /* 80286 12MHz */
    case 4: _delaybase=40; break; /* 80386 16MHz */
}
MAXDSK=(char)(0x000f&(EQUIP>>8));
setdskerr(); /* ディスクのエラーベクトルセット */
setvect(0x0006, endfunc);
getfunkey();
setfunkey();

setnewdriver("PC98", detect98); /* グラフィックスの初期化 */
initgraph(&gd, &gm, ENU);
icset(); /* アイコンデータのセット */
mouseinit(); /* マウスの初期化 */
mouseposset(MS_IX, MS_IY); /* マウスの初期設定 */
mouseareaset(); /* マウス移動範囲の設定 */
openmessage();
dspconnectname();

while(!cmd_inp()); /* 機能選択 */
while(1)
{
    switch(k)
    {
        case 1: k=editor(); break; /* ファイル編集機能 */
        case 2: k=modem(); break; /* パソコン通信機能 */
        case 3: k=computer(); break; /* LAN接続機能 */
        case 4: k=settei(RET_OFF); break; /* 接続先登録機能 */
        case 5: gomsdos(); k=20; break; /* MS-DOSコマンド機能 */
        case 6: freeCharBuff(); /* 終了処理 */
                endfunc();
        case 7: k=connect(); break; /* 接続モード */
        case 8: k=20; break; /* 空アイコン1 */
        case 9: k=history(); break; /* 履歴モード */
        case 10: k=20; break; /* 空アイコン2 */
        default: openmessage();
                dspconnectname();
                cmd_inp();
                underlinedisp();
                break;
    }
}
}

```

付録1. 各種パラメータ設定用関数群・ソースファイル (set.c)

```

/*
.....
.      パラメータ設定モード
.
.      [ SETTEL.C ]
.....
*/
#include <stdio.h>
#include <conio.h>
#include <mem.h>
#include <stdlib.h>
#include <alloc.h>
#include <dir.h>
#include <string.h>
#include <jstring.h>
#include <jctype.h>
#include <dos.h>
#include <process.h>
#include <io.h>
#include "nfire.h"
#include "fall.h"

int      settel();
void      setboudrate();
void      setdatalength();
void      setparity();
void      setttrsmethod();
void      setstopbit();
void      setxonoff();
void      setcode();
void      setreccr();
void      setsnder();
void      setreclf();
void      setsndlf();
void      setmnp();
void      setringno();
void      setringsec();
void      settelno();
void      setcntname();
int      change_mode( void );
void      set_name( void );
int      detail_change();
void      setline( void );
void      setchannel();
void      underbardisp();
int      readenvfile( void );
int      getfilnam(int mode,char *buff);
char      *dspfilnam(char *name);
int      getdirfile(int *maxl,char *name);
void      gomsdos( void );
void      setfront( void );
void      resetfront( void );
void      clrcolor( void );
int      rskill( void );
int      parasave( void );
int      paraload( void );
int      rssave( void );
void      rsload(int flg);
void      mesdisp( char *data );
void      errordisp( char *data );

/* 設定モードのメイン */
/* ボーレート */
/* データ長 */
/* パリティ */
/* 全/半2重 */
/* ストップ・ビット */
/* XON/OFF */
/* 漢字コード */
/* 受信CR処理 */
/* 送信CR処理 */
/* 受信LF処理 */
/* 送信LF処理 */
/* MNPの設定 */
/* ダイヤル回数 */
/* 話中待ち時間 */
/* 電話番号入力 */
/* 接続先名入力 */
/* 設定変更モード change_mode */
/* 登録名変更 */
/* 各設定モード */
/* 回線の設定 */
/* チャンネルの設定 */
/* 受信畳、機能コードの表示 */
/* 環境値ファイルの読みだし */
/* ファイル名の読みだし */
/* ファイル名の表示 */
/* ファイル名の獲得 */
/* MS-DOSの実行 */
/* FEP処理 */
/* カラー設定 */
/* 登録名の削除 */
/* パラメータの保存 */
/* パラメータの読み込み */
/* 登録名の保存 */
/* 登録名の読み込み */
/* メッセージの表示 */
/* エラーメッセージの表示 */

/*... 定数領域 ...*/
static char data2[] (DATA_WD)= ( "モデム回線","イーサネット回線" ),
data3[] (DATA_WD)= ( "9600","4800","2400","1200","600","300" ),
data4[] (DATA_WD)= ( "しない","する" ),
data5[] (DATA_WD)= ( "CH-1","CH-2","CH-3" ),
data6[] (DATA_WD)= ( "8ビット","7ビット" ),
data7[] (DATA_WD)= ( "なし","偶数","奇数" ),
data8[] (DATA_WD)= ( "全二重","半二重" ),
data9[] (DATA_WD)= ( "1","1.5","2" ),
data10[] (DATA_WD)= ( "あり","なし" ),

```



```

data11[][DATA_WD]={ "シフトJIS","新JIS","旧JIS","NEC漢字",
                    "DEC漢字","JIS8","ASCII"},
data12[][DATA_WD]={ "CR","CR+LF"},
data13[][DATA_WD]={ "CR","CR+LF"},
data14[][DATA_WD]={ "LF","CR+LF"},
data15[][DATA_WD]={ "LF","無視"};

extern int k;
extern int PhonoType;
static char nam[100][DATA_WD]; /* ディレクトリのファイル名(max 100) */
int FepType=0; /* FEPの形式 */
int s1; /* 選択ナンバー */
/*
.....
* 設定モードメイン
*
.....
*/
int settei(int mode)
{
    int i,i2,flg=1,wflg=WD_OPEN,rsflg,pflg,rflg,s,s2;
    struct CENTERPARA cp2;
    FILE *fp;
    static WINDOW wd2=( 46,15,34,5,T_YELLOW,T_WHITE,T_WHITE,1,"確認モード",2,3,1,0,1,0,
0 );
    static WINDOW rs_wd[]={
        ( 5, 3,32,20,T_CYAN,T_CYAN,T_WHITE, 0, "登録種1",2,18,0,1,1,1,0),
        ( 45, 3,32,20,T_CYAN,T_CYAN,T_WHITE, 0, "登録種2",2,36,18,1,1,1,0),
    };
    static char data[3][DATA_WD]={ "このまま登録していいですか?",
                                    "はい",
                                    "いいえ"};

    memcpy( &cp2,&cp,(2*16+DATA_WD+DATA_WD) ); /* パラメータの保存 */
    while( 1 )
    {
        rsload(flg); /* 接続先名の読み込み */
        flg=0;
        if ( (s1=twindow(2,wflg,rs_wd,cn.NAME) ) == -1 ) break;
        if ( wflg==WD_OPEN ) wflg=WD_REMAIN;
        if ( paraload()==-1 ) continue;
        if ( mode==RET_ON ) /* 接続先名の取得のみ? */
        {
            if ( strlen( cp.conname ) == 0 )
            {
                errordisp("ネット名がないのでアクセス出来ません。");
                continue;
            }
            break;
        }
        rsflg=0;
        while( 1 )
        {
            if ( (s2=detail_change(flg))== -1 ) {
                pflg=rflg=0;
                if ( flg ) {
                    if ( rsflg==0 ) pflg=parasave(); /* パラメータのセーブ */
                    rflg=rssave(); /* 接続先名のセーブ */
                }
                if ( pflg==0 && rflg==0 ) break;
                errordisp("ネット名がないかディスクが異常です。");
                s=twindow(1,WD_OPEN,&wd2,data);
                twindow( WD_CLOSE );
                if ( s==1 ) {
                    setmem( cn.NAME[s1],DATA_WD,0 );
                    break;
                }
            }
            flg=1;
            continue;
        }
        flg=1;
        switch( s2 )
        {
            case 0: setcntname();break;
            case 1: settelno();break;
            case 2: setline();
                    flg=2;
        }
    }
}

```

```

        break;
    case 3: if ( cp.ltype==1 ){
                if ( rskill()==YES ) rsflg=1;
            } else setboudrate();
        break;
    case 4: setmnp();break;
    case 5: setchannel();break;
    case 6: setdatalength();break;
    case 7: setparity();break;
    case 8: settrsmethod();break;
    case 9: setstopbit();break;
    case 10: setxonoff();break;
    case 11: setcode();break;
    case 12: setreccr();break;
    case 13: setsndcr();break;
    case 14: setreclf();break;
    case 15: setsndlf();break;
    case 16: setringsec();break;
    case 17: setringno();break;
    case 18: if ( rskill()==YES ) rsflg=1;
        break;
    default:break;
    }
}

twindow(WD_CLOSE);
}
twindow( WD_CLOSE );
if ( mode==RET_OFF ) memcpy( &cp,&cp2,(2*16+DATA_WD+DATA_WD) );
if ( mode==RET_ON && s1==-1 ) return -1;
return OPEN;
}
/*
-----
          設定パラメータの選択
-----
*/
int detail_change(int flg)
{
    static WINDOW wd={ 21,3,34,21,T_YELLOW,T_YELLOW,T_WHITE,0,"各設定モード",2,19,0,1,
        1,0,0 };
    static char data[19][DATA_WD];
    static int tp;

    sprintf(data[0],"N E T 名 [%19s]",cp.conname);
    if ( cp.ltype == 0 ) /* モデム回線選択? */
    {
        sprintf(data[1],"電話番号 [%19s]",cp.telno);
        sprintf(data[2],"回線 [%23s]",data2[cp.ltype]);
        sprintf(data[3],"ボーレート [%17s]",data3[cp.boudrate]);
        sprintf(data[4],"M N P 選択 [%17s]",data4[cp.mnp]);
        sprintf(data[5],"使用チャネル [%15s]",data5[cp.siol]);
        sprintf(data[6],"データ長 [%19s]",data6[cp.length]);
        sprintf(data[7],"パリティ [%17s]",data7[cp.parity]);
        sprintf(data[8],"通信方式 [%19s]",data8[cp.method]);
        sprintf(data[9],"ストップビット [%13s]",data9[cp.stopbit]);
        sprintf(data[10],"X o n / X o f f [%11s]",data10[cp.xonoff]);
        sprintf(data[11],"データコード [%15s]",data11[cp.code]);
        sprintf(data[12],"C R 受信処理 [%15s]",data12[cp.reccr]);
        sprintf(data[13],"C R 送信処理 [%15s]",data13[cp.sndcr]);
        sprintf(data[14],"L F 受信処理 [%15s]",data14[cp.reclf]);
        sprintf(data[15],"L F 送信処理 [%15s]",data15[cp.sndlf]);
        sprintf(data[16],"話中待ち時間 [%15d]",cp.ringsec);
        sprintf(data[17],"再ダイヤル数 [%15d]",cp.ringno);
        sprintf(data[18],"・登録抹消");
        wd.cmax=19;
        wd.wy=21;
        if ( tp==1 ) { wd.x=21;wd.y=3;wd.l=0; }
        tp=0;
    }
    else { /* イーサネット回線選択 */
        if ( tp==0 ) { wd.x=21;wd.y=8;wd.l=0; }
        tp=1;
        wd.cmax=4;
        wd.wy=6;
        sprintf(data[1],"ネットアドレス[%14s]",cp.telno);
    }
}

```

```

        sprintf(data[2],"回線 [%23s]",data2[cp.ltype]);
        sprintf(data[3],"・登録抹消");
    }
    if ( flg==0 || flg==2 )
    {
        if ( flg==2 ) twindow( WD_CLOSE );
        return twindow(1,WD_OPEN,&wd,data ); /* 設定メニューの選択 */
    }
    return twindow(1,WD_REMAIN,&wd,data ); /* 設定メニューの選択 */
}
/*
-----
                        接続先名設定
-----
*/
void      setcntname()
{
    int i2,flg,wflg=WD_OPEN;
    static WINDOW wd=( 48,3,30,3,T_YELLOW,T_CYAN,T_WHITE,0,"N E T 名 登 録",0,1,0,0,1,0,
    0 );
    static char data[DATA_WD];

    strcpy( data,cn.NAME[s1] );
    while( 1 )
    {
        setmem( data,DATA_WD,0 );
        if ( twindow(1, wflg,&wd,data )== -1 ) break;
        wflg=WD_REMAIN;
        flg=0;
        for( i2=0;i2<40;++i2 )
        {
            if ( strcmp(data,cn.NAME[i2])==0 && i2!=s1 && strlen(data)>0 )
            {
                errordisp("同じ登録名が他にも存在しています。");
                flg=1;
            }
        }
        if ( flg==0 ){
            strcpy( cn.NAME[s1],data );
            strcpy( cp.conname,data );
            break;
        }
    }
    twindow( WD_CLOSE );
}
/*
-----
                        電話番号設定
-----
*/
void      settelno()
{
    static WINDOW wd=( 48,3,30,3,T_YELLOW,T_CYAN,T_WHITE,0,"電話番号登録",0,1,0,0,1,0,
    0 );
    static char data[DATA_WD];
    if ( cp.ltype==1 ) wd.title="ネットアドレス";
    strcpy( data,cp.telno );
    if ( twindow(1, WD_OPEN,&wd,data )!= -1 ) strcpy( cp.telno,data );
    twindow( WD_CLOSE );
}
/*
-----
                        回線の設定
-----
*/
void setline( void )
{
    int i;
    static WINDOW wd=( 48,3,30,4,T_YELLOW,T_CYAN,T_WHITE,0,"回線の設定",2,2,0,0,1,0,0 )
    ;
    i=twindow(1, WD_OPEN,&wd,data2 );
    if ( i != -1 ) cp.ltype=i;
    twindow( WD_CLOSE );
}

```

```

/*
-----
                     ボーレート設定
-----
*/
void      setboudrate()
{
    int i;
    static WINDOW wd=( 48,3,30,8,T_YELLOW,T_CYAN,T_WHITE,0,"ボーレートの設定",2,6,0,0,
1,0,0 );
    i=twindow(1, WD_OPEN,&wd,data3 );
    if ( i != -1 ) cp.boudrate=i;
    twindow( WD_CLOSE );
}
/*
-----
                     M N P 処理設定
-----
*/
void      setmnp()
{
    int i;
    static WINDOW wd=( 48,3,30,4,T_YELLOW,T_CYAN,T_WHITE,0,"M N P 選択",2,2,0,0,1,0,0 )
;
    i=twindow(1, WD_OPEN,&wd,data4 );
    if ( i != -1 ) cp.mnp=i;
    twindow( WD_CLOSE );
}
/*
-----
                     チャンネル設定
-----
*/
void      setchannel()
{
    int i;
    static WINDOW wd=( 48,3,30,5,T_YELLOW,T_CYAN,T_WHITE,0,"使用チャネル",2,3,0,0,1,0,
0 );
    i=twindow(1, WD_OPEN,&wd,data5 );
    if ( i != -1 ) cp.sio=i;
    twindow( WD_CLOSE );
}
/*
-----
                     データ長設定
-----
*/
void      setdatalength()
{
    int i;
    static WINDOW wd=( 48,3,30,4,T_YELLOW,T_CYAN,T_WHITE,0,"データ長の設定",2,2,0,0,1,
0,0 );
    i=twindow(1, WD_OPEN,&wd,data6 );
    if ( i != -1 ) cp.length=i;
    twindow( WD_CLOSE );
}
/*
-----
                     パリティ設定
-----
*/
void      setparity()
{
    int i;
    static WINDOW wd=( 48,3,30,5,T_YELLOW,T_CYAN,T_WHITE,0,"パリティの種類",2,3,0,0,1,
0,0 );
    i=twindow(1, WD_OPEN,&wd,data7 );
    if ( i != -1 ) cp.parity=i;
    twindow( WD_CLOSE );
}
/*
-----
                     接続形式設定
-----

```

```

*/
void    settismethod()
{
    int i;
    static WINDOW wd=( 48,3,30,4,T_YELLOW,T_CYAN,T_WHITE,0,"通信方式",2,2,0,0,1,0,0 );
    i=twindow(1, WD_OPEN,&wd,data8 );
    if ( i != -1 ) cp.method=i;
    twindow( WD_CLOSE );
}
/*
-----
                        ストップビット設定
-----
*/
void    setstopbit()
{
    int i;
    static WINDOW wd=( 38,3,40,5,T_YELLOW,T_CYAN,T_WHITE,0,"ストップビット長設定",2,3,
0,0,1,0,0 );
    i=twindow(1, WD_OPEN,&wd,data9 );
    if ( i != -1 ) cp.stopbit=i;
    twindow( WD_CLOSE );
}
/*
-----
                        X o n / o f f 設定
-----
*/
void    setxonoff()
{
    int i;
    static WINDOW wd=( 48,3,30,4,T_YELLOW,T_CYAN,T_WHITE,0,"X o n / X o f f",2,2,0,0,
1,0,0 );
    i=twindow(1, WD_OPEN,&wd,data10 );
    if ( i != -1 ) cp.xonoff=i;
    twindow( WD_CLOSE );
}
/*
-----
                        漢字コード設定
-----
*/
void    setcode()
{
    int i;
    static WINDOW wd=( 48,3,30,9,T_YELLOW,T_CYAN,T_WHITE,0,"データコード",2,7,0,0,1,0,
0 );
    i=twindow(1, WD_OPEN,&wd,data11 );
    if ( i != -1 ) cp.code=i;
    twindow( WD_CLOSE );
}
/*
-----
                        C R 受信処理設定
-----
*/
void    setreccr()
{
    int i;
    static WINDOW wd=( 48,3,30,4,T_YELLOW,T_CYAN,T_WHITE,0,"C R 受信処理",2,2,0,0,1,0,
0 );
    i=twindow(1, WD_OPEN,&wd,data12 );
    if ( i != -1 ) cp.reccr=i;
    twindow( WD_CLOSE );
}
/*
-----
                        C R 送信処理設定
-----
*/
void    setsndcr()
{
    int i;
    static WINDOW wd=( 48,3,30,4,T_YELLOW,T_CYAN,T_WHITE,0,"C R 送信処理",2,2,0,0,1,0,
0 );
    i=twindow(1, WD_OPEN,&wd,data13 );
    if ( i != -1 ) cp.sndcr=i;
    twindow( WD_CLOSE );
}

```

```

0 );
i=twindow(1, WD_OPEN,&wd,data13 );
if ( i != -1 ) cp.sndcr=i;
twindow( WD_CLOSE );
}
/*
-----
                L F 受信処理設定
-----
*/
void      setrecif()
{
    int i;
    static WINDOW wd=( 48,3,30,4,T_YELLOW,T_CYAN,T_WHITE,0," L F 受信処理",2,2,0,0,1,0,
0 );
    i=twindow(1, WD_OPEN,&wd,data14 );
    if ( i != -1 ) cp.recif=i;
    twindow( WD_CLOSE );
}
/*
-----
                L F 送信処理設定
-----
*/
void      setsndif()
{
    int i;
    static WINDOW wd=( 48,3,30,4,T_YELLOW,T_CYAN,T_WHITE,0," L F 送信処理",2,2,0,0,1,0,
0 );
    i=twindow(1, WD_OPEN,&wd,data15 );
    if ( i != -1 ) cp.sndif=i;
    twindow( WD_CLOSE );
}
/*
-----
                話中待ち時間設定
-----
*/
void      setringsec()
{
    int i;
    static WINDOW wd=( 48,3,30,3,T_YELLOW,T_CYAN,T_WHITE,0," 話中待ち時間設定",0,1,0,0,
1,0,0 );
    static char data[DATA_WD];
    itoa( cp.ringsec,data,10 );
    i=twindow(1, WD_OPEN,&wd,data );
    if ( i != -1 ) {
        cp.ringsec=atoi(data);
        if ( cp.ringsec>999 ) cp.ringsec=999;
    }
    twindow( WD_CLOSE );
}
/*
-----
                再ダイヤル数設定
-----
*/
void      setringno()
{
    int i;
    static WINDOW wd=( 48,3,30,3,T_YELLOW,T_CYAN,T_WHITE,0," 再ダイヤル数設定",0,1,0,0,
1,0,0 );
    static char data[DATA_WD];
    itoa( cp.ringno,data,10 );
    i=twindow(1, WD_OPEN,&wd,data );
    if ( i != -1 ) {
        cp.ringno=atoi(data);
        if ( cp.ringno>999 ) cp.ringno=999;
    }
    twindow( WD_CLOSE );
}
/*
-----
                エラーメッセージの表示
-----

```

```

-----
*/
void errordisp( char *data )
{
WINDOW wd=( 2,22,76,3,T_RED,T_YELLOW,T_YELLOW,0,"",2,1,0,2,0,0,0 );
twindow(1, WD_OPEN,&wd,data );
twindow( WD_CLOSE );
return;
}
/*
-----
メッセージの表示
-----
*/
void mesdisp( char *data )
{
WINDOW wd=( 2,22,76,3,T_BLUE,T_WHITE,T_WHITE,0,"",2,1,0,2,0,0,0 );
twindow(1, WD_OPEN,&wd,data );
twindow( WD_CLOSE );
return;
}
/*
-----
tobihino.namの読み込み
-----
*/
void rload(int flg)
{
FILE *fp;
char NAME[80];

sprintf( NAME,"%s%stobihino.nam",ENU );
if ( (fp=fopen(NAME,"rb"))!=NULL && flg )
{
fread( &cn,(2+40+DATA_WD),1,fp);
fclose(fp);
}
}
/*
-----
tobihino.namへの書き込み
-----
*/
int rsave()
{
FILE *fp;
char NAME[80];

sprintf( NAME,"%s%stobihino.nam",ENU );
if ( (fp=fopen(NAME,"wb"))!=NULL ){
errordisp( "ファイル tobihinio.nam が開けません。" );
fclose(fp);
return(-1);
}
fwrite( &cn,(2+40+DATA_WD),1,fp );
fclose(fp);
return 0;
}
/*
-----
接続先パラメータの読み込み
-----
*/
int paraload( void )
{
FILE *fp;
static char path[DATA_WD];

sprintf(path,"%s%Y%s",ENU,cn.NAME[s1]);
setmem( &cp,(2+16+DATA_WD+DATA_WD),0 ); /* 領域の初期化 */
if ( strlen( cn.NAME[s1] )==0 ) return 0; /* 読み込む必要なし */
if ( (fp=fopen(path,"rb"))!=NULL ){
fclose( fp );
errordisp( "そのような登録ファイルはありません" );
}
}

```

```

    setmem( cn.NAME[s1].DATA_WD,0 );
    rsave();
    return(-1);
}
fread( &cp,(2*16+DATA_WD+DATA_WD),1,fp );
fclose( fp );
return 0;
}
/*
-----
                接続先パラメータの書き込み
-----
*/
int parasave( void )
{
    FILE *fp;
    int i;
    static char path[DATA_WD];
    static WINDOW wd={ 42,15,36,5,T_CYAN,T_CYAN,T_WHITE,1,"書き込みモード",2,3,1,0,1,0,0 };
    static char data1[DATA_WD]={"今のパラメータを登録しますか?"
                                ,"はい"
                                ,"いいえ"
                                };

    if ( strlen(cn.NAME[s1])==0 )
    {
        errordisp("NET名がないので書き込めません。");
        return -1;
    }
    sprintf( path,"%s%%s",ENV,cn.NAME[s1] );
    if ( twindow(1,WD_OPEN,&wd,data)==1 )
    {
        if ( (fp=fopen(path,"wb"))==NULL ) {
            errordisp( "ファイルを書き込むことができません" );
            twindow( WD_CLOSE );
            fclose(fp);
            return(-1);
        }
        fwrite( &cp,(2*16+DATA_WD+DATA_WD),1,fp );
        fclose( fp );
    } else {
        /* パラメータの登録はしない */
        if ( (fp=fopen(path,"rb"))==NULL ) setmem(cn.NAME[s1].DATA_WD,0);
        fclose( fp );
    }
    twindow( WD_CLOSE );
    return 0;
}
/*
-----
                登録抹消
-----
*/
int rskill()
{
    int i,flg;
    static WINDOW wd={ 48,15,30,4,T_RED,T_YELLOW,T_YELLOW,0,"登録抹消",2,2,0,0,1,0,0 };

    if ( twindow(1, WD_OPEN,&wd,data4 )==1 ) {
        setmem( &cp,(2*16+DATA_WD+DATA_WD),0 );
        unlink( cn.NAME[s1] );
        setmem( cn.NAME[s1].DATA_WD,0 );
        setmem( cp.connname,DATA_WD,0 );
        flg=YES;
    } else flg=FALSE;
    twindow( WD_CLOSE );
    return flg;
}
/*
-----
                最下位行表示
-----
*/
void underbardisp(int lfg,int pfg,int tfg,int cfg,int f98)
{

```



```

int x,y,x1,i;

x=wherex();
y=wherey();
window(1,1,80,25);
gotoxy(1,25);
textcolor(co.box);
cprintf(" ");
textreverse(REVERSE);
cprintf(" ");
cprintf("| 受信量 : [00%]");
cprintf("| 機能 : ");
if(lfg!=0) cprintf(" — ");
if(pfg!=0) cprintf(" ");
if(tfg!=0) cprintf(" ");
if(cfg!=0) cprintf(" ");
if(f98!=0) cprintf("UT");
else cprintf("98");
x1=wherex();
for(i=x1;i<78;i++) putch(' ');
gotoxy(78-22,25);
textreverse( NOREVERSE );
cprintf("| F10: 各種機能呼出し|");
window(1,1,80,24);
gotoxy(x,y);
textcolor( T_WHITE );
)
/*
-----
環境値ファイルの読み出し
-----
*/
int readenvfile()
(
FILE fp;
int Fh;
char t,NAME[80];

sprintf(NAME,"%s\\tobihino.env",ENU );
if((fp=fopen(NAME,"rb"))==NULL) return FALSE;
Fh=fopen(fp);
if(read(Fh,&t,1)<0){
fclose(fp);
return FALSE;
}
PhoneType=(0x00ff&t);
if(read(Fh,&t,1)<0){
fclose(fp);
return FALSE;
}
FepType=(0x00ff&t);
fclose( fp );
return YES;
)
/*
.....
- ファイル名取得とディレクトリ表示 -
.....
*/
-----
ファイル名の取得
-----
*/
int getfilnam(int mode,char *buff)
(
static WINDOW wd=( 20,10,40,3,T_CYAN,T_YELLOW,T_WHITE,0,"パスとファイル名入力"
,0,1,0,0,1,0,0 );
char data[DATA_WD],dir[65],fname[13],ext[4],drive[3],*p;
int i,s,cdisk,dsk,wflg=WD_OPEN,jpflg=0;
FILE *fp;

setmem(data,DATA_WD,0); /* 領域の初期化 */
setmem(dir,65,0);

```

```

setmem(fname,13,0);
setmem(ext,4,0);
cdsk=getdisk();
fnsplit(buff,drive,dir,fname,ext);
strncpy(data,buff,DATA_WD);

while( 1 )
{
if ( jpflg==0 ){
if ( (s=twindow(1, wflg,&wd,data ))== -1 ) break; /* 中断 */
wflg=WD_REMAIN;
}
jpflg=0;
fnsplit(data,drive,dir,fname,ext);
dsk=toupper(*drive)-'A'; /* 正しいパスを入力したか? */
if(dsk<0 && dsk>MAXDSK) dsk=0;
if(dsk!=cdsk){
setdisk(dsk);
cdsk=dsk;
}
if(*dir!=0) chdir(dir); /* チェンジ ディレクトリ */
if(strchr(fname,'.')!=0 || strchr(fname,'?')!=0 || *data==0){
setmem(data,DATA_WD,0);
if(*data==0) strcpy(data,".."); /* 何も入力してない時 */
else{
strcpy(data,fname); /* ワイルドカードの使用 */
strcat(data,ext);
}
p=dspfilnam(data); /* ファイル名の表示 */
if(DskErrFlag!=0){ /* ディスクエラー */
setmem(data,DATA_WD,0);
DskErrFlag=0;
errordisp("ディスク関連でエラーが発生しました。");
continue;
}
setmem(data,DATA_WD,0);
getcwd(data,DATA_WD);
if(*p!=0){
if(strlen(data)>3) strcat(data,"*");
strcat(data,p);
}
}
else{
if ( (fp=fopen(data,"r"))==NULL && mode==RET_ON )
{
/* Still Haven't got FileName! */
chdir(data);
setmem( data,DATA_WD,0 );
jpflg=1;
continue;
}
fclose( fp );
setmem(buff,DATA_WD,0); /* そのままでいけるパス名 */
strncpy(buff,data,DATA_WD);
break;
}
}

}
twindow( WD_CLOSE );
return s;
}
/*

```

ディレクトリーの表示

```

/*
char      dspfilnam(char *name)
{
WINDOW wd=( 2,13,76,10,T_CYAN,T_CYAN,T_WHITE,0,"",2,100,0,1,1,0,5 );
char      fname[80];
char      cdir[80];
char      dspdir[80];
int MAXLOW,MAXCOL;
int s,i,ini=1,tmp;

```

```

/* 現在のディレクトリのクリア */
setmem(cdir,80,0);
getcwd(cdir,79);

while(1){
tmp=getdirfile(&MAXLOW,name); /* ファイル名を読み込む */
sprintf(dspdir,"カレントディレクトリ %s",cdir);
if ( tmp != FALSE ) wd.omax=tmp;
wd.title=dspdir;
if ( ini==0 ) twindow( WD_CLOSE ); else ini=0;
s=twindow(1 ,FW_OPEN,&wd,name);
if ( s==1 )
{
setmem(fname,80,0);
break;
}
if(strchr(nam[s],'%')!=NULL) /* ディレクトリ指定 */
{ /* もう一度、初めから */
i=strlen(nam[s]);
nam[s][i-3]='%0';
chdir(nam[s]);
getcwd(cdir,77);
continue;
}
else{ /* ファイル名決定 */
setmem(fname,80,0);
strcpy(fname,nam[s]);
break;
}
}
twindow( WD_CLOSE );
return fname;
}
/*
-----
ファイル名の読み出し
-----
*/
int getdirfile(int *maxl,char *name)
{
struct fblk fb;
FILE *fp;
int lin=0;

if(findfirst(name,&fb,FA_DIREC)==-1){ /* 入力したファイルが存在していない */
*maxl=0;
errordisp("該当するファイルが存在していませんでした。");
return FALSE;
}
setmem(nam,(20*5+DATA_WD),0);
strcpy( nam[0],fb.ff_name );
if ( (fp=fopen(fb.ff_name,"rb"))==NULL )
{
strcat(nam[0],"<%%>");
} else fclose( fp );
lin++;
while(findnext(&fb)!=-1){
strcpy( nam[lin],fb.ff_name );
if ( (fp=fopen(fb.ff_name,"rb"))==NULL )
{
strcat( nam[lin],"<%%>"); /* ディレクトリだ! */
} else fclose( fp );
lin++;
if(lin>100) break;
}
*maxl=lin;
return lin;
}
/*
.....
MS-DOSの実行
.....
*/
void gomsdos( void )

```

```

{
FILE *fp;
unsigned long l;
int s,x1,y1;
static WINDOW wd=( 20,3,40,12,T_CYAN,T_YELLOW,T_WHITE,0
,"MSDOS コマンドの入力",0,10,0,0,1,1,-1 );
static char data[10][DATA_WD];
static char MSCOMD[80],NAME[80];          /* MS-DOS コマンドバッファ */

sprintf( NAME,"%s\\tobihino.dos",ENV );
/* DOS ヒストリーの処理 */
if ( (fp=fopen(NAME,"rb")) != NULL )
{
fread( data,10*DATA_WD,1,fp );
}
fclose( fp );

x1=wherex();
y1=wherey();
while( (s=twindow(1, WD_OPEN,&wd,data)) != -1 )
{
twindow( WD_CLOSE );
setmem( MSCOMD,80,0 );
strcpy( MSCOMD,data[s] );
textcursor(DISP_CURSOR);
putfunkey();
printf("%c[11%c[2J",ESC,ESC);
l=coreleft();
printf("残りのメモリサイズは%dバイトです。%r%n",l);
printf("EXITでTOBIHINOに戻ります。%r%n");
system(MSCOMD);
printf("%c[24;1Hどれかのキーを押して下さい",ESC);
while( key_getc()==0 );
rewind( stdin );
setfunkey();
}
if ( s== -1 ) twindow( WD_CLOSE );
textcursor( NODISP_CURSOR );
gotoxy( x1,y1 );
/* DOS ヒストリーの処理 */
fp=fopen(NAME,"wb");
fwrite( data,10*DATA_WD,1,fp );
fclose( fp );
return;
}
/*

```

----- フロントプロセッサのセット -----

```

/*
void setfront()
{
union REGS regs;

switch(FepType)
{
case 0: /* ATOK6 */
regs.h.ah=0x01;
int86(0x6f,&regs,&regs);
regs.h.ah=0x17; /* Iコー */
int86(0x6f,&regs,&regs);
break;

case 1: /* vje */
regs.h.ah=0x01;
regs.h.al=0x02;
int86(0x70,&regs,&regs);
break;

case 2: /* NEC:A1 */
regs.h.cl=0xef;
int86(0xdc,&regs,&regs);
break;

default:
break;
}
}

```

```

)
/*
-----
      フロントプロセッサのリセット
-----
*/
void    resetfront()
{
    union    REGS    regs;

    switch(FepType)
    {
    case    0:    /* ATOK */
        regs.h.ah=0x0b;
        int86(0x6f,&regs,&regs);
        break;
    case    1:    /* vje */
        regs.h.ah=0x01;
        regs.h.al=0x00;
        int86(0x70,&regs,&regs);
        break;
    case    2:    /* NEC:AI */
        regs.h.cl=0xf0;
        int86(0xdc,&regs,&regs);
        break;
    default:
        break;
    }
}
/*
-----
      カラーの初期化
-----
*/
void    clrcolor()
{
    co.title=T_RED;          /* ウィンドウ・タイトル */
    co.box=T_CYAN;           /* ボックス・カラー */
    co.ch=T_YELLOW;         /* ウィンドウ内キャラクタ・カラー */
    co.history=T_WHITE;      /* 履歴キャラクタ・カラー */
    co.editor=T_WHITE;       /* 便せんキャラクタ・カラー */
    co.half=T_BLUE;         /* 半重キャラクタ・カラー */
    co.send=T_MAGENTA;       /* 無手順送信文字キャラクタ・カラー */
    co.recv=T_GREEN;         /* 無手順受信文字キャラクタ・カラー */
    co.control=T_RED;        /* 制御文字キャラクタ・カラー */
    co.error=T_RED;          /* エラー・キャラクタ・カラー */
    co.itime=T_GREEN;        /* 接続時間カラー */
    co.ntime=T_CYAN;         /* 現在時刻カラー */
    co.ltime=T_WHITE;        /* ラップタイムカラー */
    co.kakuninn=T_MAGENTA;   /* 確認カラー */
    co.R_control=NOREVERSE;  /* 制御文字キャラクタ・カラー */
}

```

付録J. 簡易ワープロ・ソースファイル(ed.c)

```

/*
.....
.      パソコン通信ソフト「飛火野」
.
.      スクリーンエディタ
.....
*/
#include      <stdio.h>
#include      <conio.h>
#include      <string.h>
#include      <jstring.h>
#include      <jctype.h>
#include      <mem.h>
#include      <dos.h>
#include      <process.h>
#include      <io.h>
#include      <stdlib.h>
#include      "nfire.h"
#include      "fall.h"

#define SCRNW1      (SCRNW-1)
#define SCRNW2      (SCRNW-2)
#define SCRNL      24
#define SYSFNMAX      80
#define linepl      (linep-1)

int editor();
int ed_keyin();
int ed_modechg();
int edcontrol(char c);
int editfunc();
int editfile();
void edcharkey(char c);
int edfilerd();
void edfilewr();
void edclrmail();
void edgotop();
int setno();
void edsetcolm();
void edareaset();
void edareadel();
void edareacpy();
int edsearch(int x,int y);
void edins(unsigned short ch);
int edins1();
int edins2();
int edins3();
void edins4(unsigned short ch);
void edins5(unsigned short ch);
int getpos(int x,int line);
void newline();
void inspace();
void condisp(int line);
int condispl(int x,int line);
void setedcolor(int line);
void conclr(int y);
int chkjis(int x,int line);
void ed_haniset();
void ed_hanioff();
void ed_copy();
void ed_left();
void ed_right();
void ed_rup();
void ed_rdown();
void ed_up();
void ed_down();
void ed_back();
void ed_del();
void line_del();
void crret();
void dshtmov(int n,int p);

```

/* 1 行の字数 */
 /* 漢字の字数 */
 /* 画面の総行数 */
 /* ファイルバッファの字数 */
 /* 現在のポインタライン */
 /* エディタのメイン */
 /* キー入力 */
 /* モード変更 */
 /* 外部キー処理 */
 /* 機能処理 */
 /* ファイル名取得 */
 /* キャラクタキー処理 */
 /* ファイルの読み込み */
 /* ファイルの書き込み */
 /* 便箋白紙 */
 /* 先頭行移動 */
 /* 行字数 */
 /* 桁数設定 */
 /* 行範囲設定 */
 /* 行範囲削除 */
 /* 行範囲複写 */
 /* 文字検索 */
 /* 基本部 */
 /* 最終ポインタ書き込み */
 /* 挿入書き込み */
 /* 現在のポジション */
 /* 改行動作 */
 /* スペース出力 */
 /* 画面表示 */
 /* 1行表示 */
 /* エディター用カラーセット */
 /* コンソールクリア */
 /* 位置のコードタイプ */
 /* 範囲セット */
 /* 範囲解除 */
 /* 範囲複写 */
 /* 左移動 */
 /* 右移動 */
 /* ページアップ */
 /* ページダウン */
 /* 上移動 */
 /* 下移動 */
 /* バック */
 /* 1字削除 ^G */
 /* ライン削除 ^Y */
 /* CR表示 */
 /* メモリ移動部 減 */

```

void      ushtmov(int n,int p);          /* 再移動、並び替え */
void      reload(int line);             /* モードコメント */
void      pmtmode(char *s);             /* エディタ表示 */
void      pmtedit();                    /* 最上行の表示 */
void      pmtupperdisp();               /* 行表示 */
void      pmtline();                    /* 力ラム表示 */
void      pmtcol();                     /* 変数のクリア */
void      edbufnew();                   /* エディタの初期化 */
void      edinit();                     /* 表示幅の設定 */
int        edwidth(void);

int        SCRNW=90;                    /* 1行字数 */
char       editbuf[BUFNUM];             /* edit buffers */
int        linenum[BUFLN];              /* display character numbers */
int        linepos[BUFLN];              /* line pointer */
char       edsrbuf[60];                 /* 検索文字のバッファ */
char       EdFile[90];                  /* エディタ・ファイル名 */
int        linep;                       /* current line pointer */
int        endpos;                      /* char end pointer */
int        ins_flag=0;                  /* insert mode flag */
int        x_pos;                       /* 範囲指定 x座標 */
int        y_pos;                       /* 範囲指定 y座標 */
int        AreaFlag;                   /* 範囲設定フラグ */
int        CopyFlag;                   /* コピー設定フラグ */
int        areastrline;                 /* 範囲スタートライン */
int        areaendline;                 /* 範囲エンドライン */
int        editendline;                 /* エディタバッファ最終行 */
int        dispstrline;                 /* 画面表示スタートライン */

/*
-----
                      内部エディタモード
-----
*/

int editor()
{
    int i,s,c;
    char c1,c2;

    textcursor(NOD[SP_CURSOR]);          /* 画面設定 */
    window(1,1,80,25);
    clrscr();
    underlinedisp();
    linebufdisp();
    gotoxy(1,1);
    textcolor(co.box);
    textreverse(REVERSE);
    for(i=0;i<80;i++)    putch(' ');
    window(1,1,80,24);
    if(EditFlag==0){          /* 初めてエディットモードにきた */
        edbufnew();
        edinit();
        endpos=0;             /* end position */
        editendline=0;        /* end line numbers */
        EditFlag=FALSE;
    }
    else {
        if(EdCopyFlag!=0){    /* 履歴モードからの書き込み */
            reload(0);
            EdCopyFlag=0;
        }
        window(1,1,80,24);
        gotoxy(1,2);
        textreverse(NOREVERSE);
        textcolor(co.editor);
        condisp(linep);
    }
    window(1,1,80,24);
    gotoxy(1,2);
    pmtupperdisp();
    textcursor(DISP_CURSOR);
    setfront();

```

```

mouseon();
MS_LB=0;
MS_RB=0;

while(1)
{
    do {
        if ( MS_LB ) { mouseoff();
                        s=editfunc();
                        if ( s==9 ) return OPEN; /* メインメニューに復帰 */
                        mouseon(); }
        mouseput();
    } while ( ( c=key_in() ) == 0 );
    c1=(char)(0x00ff&(c)>8);
    c2=(char)(0x00ff&c);
    if(c2==0) edcontrol(c1);    else    edcharkey(c2);
}

/*
-----
                      コマンドキー処理
-----
*/
int editfunc()
{
    int s,x,y,wflg=WD_OPEN,flg=1;
    static char edmenu[11][DATA_WD]=
    {
        "ファイル読出","ファイル書込","便せん白紙","文字検索",
        "範囲指定","範囲削除","範囲複写","行先頭","表示幅設定",
        "メインメニュー"
    };
};
static WINDOW wd=
{ 25,3,30,12,T_CYAN,T_YELLOW,T_WHITE,0,"各種機能",2,10,0,0,1,1,0 };

x=wherex();
y=wherey();
while( flg )
{
    s=twindow(1,wflg,&wd,edmenu );
    wflg=WD_REMAIN;
    switch(s)
    {
        case 0: if ( edfilerd()==YES ) { flg=0;
                                          x=1;y=2;
                                          }
               break;
        case 1: edfilerw();               break;
        case 2: edclrmall();flg=0;x=1;y=2; break;
        case 3: if ( edsearch(x,y)==YES ) { x=wherex();
                                          y=wherey();
                                          flg=0;
                                          }
               break;
        case 4: edareaset();               break;
        case 5: edareadol();               break;
        case 6: edareacpy();               break;
        case 7: edgotop();
               x=wherex();
               y=wherey();
               break;
        case 8: if ( edwidth()==YES ) { flg=0;
                                          x=1;y=2;
                                          }
               break;
        case 9: /* メインメニューに復帰 */
        default: flg=0;break;
    }
}
allclose();
window( 1,1,80,24 );
gotoxy(x,y);
textcolor(co.editor);

```



```

textcursor( DISP_CURSOR );
return s;
}
/*
-----
表示幅設定
-----
*/
int edwidth( void )
{
static WINDOW wd={ 10,15,30,3,T_YELLOW,T_CYAN,T_WHITE,0,"表示幅設定"
,0,1,0,0,1,0,0 };
static char data[DATA_WD];
int s;

s=twindow( 1,WD_OPEN,&wd,data );
twindow( WD_CLOSE );
if ( s != -1 ) {
    allclose();
    SCRNV=atoi( data );
    reload(0);
    window( 1,2,80,24 );
    clrscr();
    window( 1,1,80,24 );
    linep=1;
    gotoxy( 1,2 );
    condisp( 0 );
    pmtupperdisp();
    return YES;
}
return FALSE;
}
/*
-----
ファイル名入力
-----
*/
int editfile()
{
    if ( getfilnam(RET_ON,EdFile)== -1 ) return FALSE;
    if (*EdFile!=0) return YES;
    else return FALSE;
}
/*
-----
コントロールキー処理
-----
*/
int edcontrol(char c)
{
textcursor(NODISP_CURSOR);
switch(c)
{
case  RUP:      /* ページアップ */
    ed_rup();    pmtline(); break;
case  RDOWN:    /* ページダウン */
    ed_rdown();  pmtline(); break;
case  UP:
    ed_up();     pmtline(); break;
case  DOWN:
    ed_down();   pmtline(); break;
case  LEFT:
    ed_left();   pmtcol(); break;
case  RIGHT:
    ed_right();  pmtcol(); break;
case  INS:      /* INS */
    if(AreaFlag!=0) break;
    if(ins_flag==0) ins_flag=0xff;
    else ins_flag=0;
    pmtedit();  break;
case  DEL:      /* Delete */
    if(AreaFlag!=0) break;
    ed_del();    pmtcol();
    break;
}
}

```

```

        case FUN10: MS_LB=1;break;
        default: break;
    )
    textcursor(DISP_CURSOR);
    return 1;
}

/*
-----
               キ ャ ラ ク タ キ ー 処 理
-----
*/
void edcharkey(char c)
{
    int kc,kc1,kc2;
    char c1;

    textcursor(NODISP_CURSOR);
    switch(c)
    {
        case BS:
            ed_back();
            ed_del();
            pmtcol();
            break;
        case 0x19: /* ^Y */
            if(editendline>=linep1){
                line_del();
                pmtline();
                pmtcol();
            }
            break;
        case 0x07: /* ^G */
            ed_del();
            pmtcol();
            break;
        case CR:
            if(AreaFlag==0){
                edins(0x00ff&c);
                pmtline();
                pmtcol();
            }
            else ed_haniset();
            break;
        case ESC: ed_hanioff(); break;
        default :
            if(iskanji(0x00ff&c)!=0){
                c1=getch();
                kc1=0xff00&((0x00ff&c)<<8);
                kc2=0x00ff&c1;
                kc=kc1|kc2;
            }
            else kc=0x00ff&c;
            edins(kc);
            pmtline();
            pmtcol();
            break;
    }
    textcursor(DISP_CURSOR);
}

/*
-----
               ファ イ ル ・ デ ー タ の 読 み 出 し
-----
*/
int edfilerd()
{
    FILE *file;
    int i,j,q,r,ch;
    long count;
    char c,*p;

    textcursor(NODISP_CURSOR);

```

```

while( 1 )
{
    if ( editfile()==FALSE ) return FALSE; /* ファイル名の読み込み */
    if ( (file=fopen(EdFile,"rb"))==NULL )
    {
        errordisp("そのようなファイルは、存在していません。");
        fclose( file );
        continue;
    }
    break;
}
odbufnew();
i=0;
r=0;
for(count=0;count<BUFNUM;count++){
    ch=fgetc(file);
    if(feof(file)!=0) break;
    else c=(char)ch;
    if(c==TAB){
        q=8-(r-(r/8)*8);
        for(j=0;j<q;j++){
            editbuf[i]=' ';
            i++;
        }
        r+=q;
    }
    else if(c==CR){
        editbuf[i]=CR; r=0; i++;
    }
    else if(c>=' '){
        editbuf[i]=c; r++; i++;
    }
    if(i>BUFNUM) break;
}
fclose(file);
allclose();
endpos=i;
linep=1;
reload(0);
window(1,2,80,24);
clrscr();
window(1,1,80,24);
textreverse(NOREVERSE);
textcolor(co.editor);
gotoxy(1,2);
condisp(linep1);
putupperdisp();
return YES;
}
/*
-----
          ファイルへのデータ保存
-----
*/
void edfilewr()
{
    FILE *fp;
    int i,s;
    char c;
    static WINDOW wd={ 2,6,40,3,T_CYAN,T_YELLOW,T_WHITE,0
        ,"新規登録(ファイル名を入力)",0,1,0,0,1,0,-1 };
    static WINDOW wd2={ 2,12,36,5,T_CYAN,T_YELLOW,T_WHITE,1
        ,"ファイル更新モード",2,3,1,0,1,0,0 };
    static char data[1][DATA_WD];
    static char data2[3][DATA_WD]={ "上書きしてもよろしいでしょうか?",
        "はい",
        "いいえ" };

    if(EdFile[0]==0){ /* ファイル名の登録 */
        s=twindow(1, WD_OPEN,&wd,data );
        twindow( WD_CLOSE );
        if ( s==-1 ) return; /* ESC */
        strcpy( EdFile,data[0] );
    }
}

```

```

if((fp=fopen(EdFile,"rb"))!=NULL){ /* 同じ名前のファイル名が存在 */
    s=twindow(1, WD_OPEN,&wd2.data2 );
    twindow( WD_CLOSE );
    fclose( fp );
    if ( s != 1 ) return;
}
else {fclose(fp);
fp=fopen(EdFile,"wb");
if(fp!=NULL){
    for(i=0;i<BUFNUM;i++){
        if(i==endpos) break;
        c=editbuf[i];
        fputc(0x00ff&c,fp);
        if(c==CR) fputc(0x00ff&LF,fp);
        else if(c==0) break;
    }
    fclose(fp);
}
}
/*

```

便箋白紙

```

/*
void    edclrmail()
{
    allclose();
    edbufnew(); /* クリア */
    edinit();
    endpos=0; /* end position */
    editendline=0; /* end line numbers */
    window(1,2,80,24);
    clrscr();
    window(1,1,80,24);
    pmtupperdisp();
}
/*

```

行範囲指定

```

/*
void    edareaset()
{
    allclose();
    pmtmode("範囲設定");
    AreaFlag=1;
    pmtline();
    pmtcol();
}
/*

```

範囲削除

```

/*
void    edareadel()
{
    unsigned short  p1,p2,n;

    allclose();
    if(AreaFlag!=3) return;
    p1=linepos[areaendline+1];
    p2=linepos[areastrline];
    n=p1-p2;
    dshtmov(n,p2);
    reload(areastrline);
    gotoxy(1,2);
    conclr(wherex());
    AreaFlag=0;
    gotoxy(1,2);
    condisp(dispsrline);
    linep=dispsrline+y_pos-1;
    areaendline=0;
    areastrline=0;
    gotoxy(x_pos,y_pos);
}

```

```

    pmtupperdisp();
    gotoxy(x_pos,y_pos);
}
/*
-----
                      範囲複写
-----
*/
void    edareacpy()
{
    allclose();
    if(AreaFlag==3){
        pmtmode("複写先指定");
        CopyFlag=0xff;
    }
    pmtline();
    pmtcol();
}
/*
-----
                      文字検索
-----
*/
int edsearch(int x,int y)
{
    char    data[81],c,*p1,*p2;
    int d,s,i,lp,cp,ny,nx,x1,y1;
    static WINDOW wd=( 2,2,40,4,T_GREEN,T_YELLOW,T_WHITE,1,"検索文字の入力"
        ,0,2,0,0,1,0,1 );
    static WINDOW wd2=( 2,22,32,3,T_BLUE,T_YELLOW,T_WHITE,0,"",1,1,0,2,1,0,0 );
    static char mdata[2][DATA_WD];
    static char mdata2[1][DATA_WD]={"再検索は(S^*-S) 中止は(ESC)" };
    p1=data;
    nx=1;ny=1;
    strcpy( mdata[0],edsrdbuf );
    s=twindow(1, WD_OPEN,&wd,mdata );
    twindow( WD_CLOSE );
    if ( s != -1 ) strcpy( edsdbuf,mdata[s] );
    if( s== -1 || *edsdbuf==0 ){
        window(1,1,80,24);
        textcolor(co.editor);
        textreverse(NOREVERSE);
        textcursor(DISP_CURSOR);
        gotoxy(x,y);
        return FALSE;
    }
    allclose();
    window(1,1,80,24);
    gotoxy(x,y);
    while(1){
        x1=wherex();
        y1=wherey();
        setmem(p1,81,0);
        if(x1<linenum[1]linep1){
            lp=linep1; cp=x1;
        }
        else{
            lp=linep1+1; cp=0;
        }
        for(i=lp;i<BUFLen;i++){
            setmem(p1,81,0);
            memcpy(p1,(editbuf+linepos[i]),linenum[i]);
            if((p2=jstrchr((p1+cp),edsdbuf))!=NULL) break;
            lp++;
            cp=0;
        }
        if(i>=BUFLen){
            textcursor( NODISP_CURSOR );
            errordisp("検索文字列が見つかりません。");
            window(1,1,80,24);
            textcolor(co.editor);
            textreverse(NOREVERSE);
            textcursor(DISP_CURSOR);
            gotoxy(x1,y1);
        }
    }
}

```

先頭行移動

```
.....
|                                     |
|               エディター             |
|                                     |
|-----
```

```

        mouseoff();
        errordisp("便せんが満杯です。");
        mouseon();
        window(1,1,80,24);
        textcolor(co.editor);
        textreverse(NOREVERSE);
        gotoxy(x,y);
        textcursor(DISP_CURSOR);
        return;
    }
    if (linep1>editendline){
        if (c==CR :: ins_flag!=0)    return;
        p=edins1();
    }
    else if (linep1==editendline && linenum[linep1]<x)    p=edins2();
    else if (linep1<editendline && linenum[linep1]<x){
        if (c==CR :: ins_flag!=0)    return;
        else    p=edins3();
    }
    else    p=getpos(x,linep1);
    if (endpos==p && linep1==editendline)    edins4(ch);
    else    edins5(ch);
}
/*
-----
insert cr and space
-----
*/

int edins1()
{
    int i,x,y,p;

    x=wherex();
    y=wherey();
    for (i=editendline; i<linep1; i++){
        editbuf[endpos]=CR;
        linenum[editendline]++;
        editendline++;
        endpos++;
        linepos[editendline]=endpos;
    }
    linep=editendline+1;
    for (i=0; i<(x-1); i++){
        editbuf[endpos]=' ';
        endpos++;
    }
    linenum[linep1]=(x-1);
    gotoxy(1,2);
    condisp(editendline-(y-2));
    gotoxy(x,y);
    p=getpos(x,linep1);
    return p;
}
/*
-----
insert space in end line
-----
*/

int edins2()
{
    int i,x,y,n,p;

    x=wherex();
    y=wherey();
    n=linenum[linep1]+1;
    gotoxy(n,y);
    for (i=n; i<=(x-1); i++){
        putch(' ');
        editbuf[endpos]=' ';
        linenum[linep1]++;
        endpos++;
    }
    gotoxy(x,y);

```

```

        p=getpos(x,linep1);
        return p;
    }
    /*
    -----
        insert space
    -----
    */
    int edins3()
    {
        int i,x,y,p,n;

        x=wherex();
        y=wherey();
        if(x==SCRNW1 && linenum[linep1]==78){
            gotoxy(1,y);
            ed_down();
            p=getpos(1,linep1);
            return p;
        }
        else{
            n=x-linenum[linep1];
            p=getpos(linenum[linep1],linep1);
            ushtmov(n,p);
            for(i=0;i<n;i++)    editbuf[p+i]=' ';
            reload(linep1);
            gotoxy(1,y);
            condispl(1,linep1);
            textcursor(DISPC_CURSOR);
            gotoxy(x,y);
            p=getpos(x,linep1);
            return p;
        }
    }
    /*

```

エンドポインターへの書き込み

```

    */
    void edins4(unsigned short ch)
    {
        int x;
        char c1,c2;

        x=wherex();
        c1=(char)(0x00ff&_rotr(ch,8));
        c2=(char)(0x00ff&ch);
        if(c1!=0){
            if(x>SCRNW2)    newline();
            editbuf[endpos]=c1;
            linenum[linep1]++;
            endpos++;
            editbuf[endpos]=c2;
            endpos++;
            linenum[linep1]++;
            putch(c1);
            putch(c2);
            x=wherex();
            if(x==SCRNW)    newline();
        }
        else if(c2==CR){
            editbuf[endpos]=c2;
            linenum[linep1]++;
            crret();
            endpos++;
            newline();
        }
        else{
            editbuf[endpos]=c2;
            if(x>SCRNW1)    newline();
            putch(c2);
            linenum[linep1]++;
            endpos++;
            x=wherex();

```



```

        if(x==SCRNW)    newline();
    }
}
/*
-----
      挿入書き込み
-----
*/
void    edins5(unsigned short ch)
{
    int x,y,p,n1,n2,n3;
    char    c1,c2,c3,c4;

    x=whorox();
    y=wherey();
    c1=(char)(0x00ff&_rotr(ch,8));
    c2=(char)(0x00ff&ch);
    p=getpos(x,linep1);
    if(ins_flag==0){
        if(c1!=0){ /* 漢字 */
            ushrtmov(2,p);
            editbuf[p]=c1;
            editbuf[p+1]=c2;
            reload(linep1);
            gotoxy(1,y);
            if(linenum[linep1]<SCRNW2)    condispl(1,linep1);
            else
                condispl(linep1);
            if(x==SCRNW2){
                gotoxy(1,y);
                ed_down();
            }
            else if(x==SCRNW1){
                gotoxy(3,y);
                ed_down();
            }
            else
                gotoxy(x+2,y);
        }
        else if(c2==CR){
            ushrtmov(1,p);
            editbuf[p]=c2;
            reload(linep1);
            gotoxy(1,y);
            condispl(linep1);
            gotoxy(1,y);
            ed_down();
        }
        else if(c2>=' '){
            ushrtmov(1,p);
            editbuf[p]=c2;
            reload(linep1);
            gotoxy(1,y);
            if(linenum[linep1]<SCRNW2)    condispl(1,linep1);
            else
                condispl(linep1);
            if(x==SCRNW1){
                gotoxy(1,y);
                ed_down();
            }
            else
                gotoxy(x+1,y);
        }
    }
    else{
        n1=chkjis(x,linep1);
        n2=chkjis(x+1,linep1);
        if(c1!=0){
            if(editbuf[p]==CR)    return;
            if(editbuf[p+1]==CR)    return;
            if(n1==2)    return;
            if((n1==0 && n2==0) || n1==1){
                editbuf[p]=c1;
                editbuf[p+1]=c2;
                gotoxy(1,y);
                clreol();
                condispl(1,linep1);
                if(x>(SCRNW1-2)){

```

```

        gotoxy(1,y);
        ed_down();
    }
    else gotoxy(x+2,y);
}
}
else if (c2==CR) return;
else{
    if(editbuf[p]==CR) return;
    if(n1==2) return;
    if(n1==1){
        editbuf[p]=c1;
        editbuf[p+1]=c2;
        gotoxy(1,y);
        clrscr();
        condispl(1,linep1);
        if(x>(SCRNW1-1)){
            gotoxy(1,y);
            ed_down();
        }
        else gotoxy(x+2,y);
    }
    else{
        editbuf[p]=c2;
        putch(c2);
        if(x==SCRNW){
            gotoxy(1,y);
            ed_down();
        }
        else gotoxy(x+1,y);
    }
}
}
}
}
/*

```

改行動作

```

/*
void    newline()
{
    putch(CR);
    ed_down();
    editendline++;
    linepos[linep1]=endpos;
}
/*

```

範囲指定解除

```

/*
void    ed_hanioff()
{
    int x,y;

    AreaFlag=0;
    CopyFlag=0;
    dispstrline=0;
    areastrline=0;
    areaendline=0;
    x=wherex();
    y=wherey();
    gotoxy(1,2);
    condispl(linep1-y+2);
    gotoxy(x,y);
    pntedit();
}
/*

```

カーソル左移動

```

/*
void    ed_left()
{

```

```

    int x;

    if(wherex()==1) return;
    putch(BS);
    pmtcol();
}
/*
-----
                カーソル右移動
-----
*/
void    ed_right()
{
    int x,y,p,n;

    x=wherex();
    y=wherey();
    n=chkjis(x,linep1);
    if(x==SCRNW1) return;
    else{
        if((linenum[linep1]<x) gotoxy(x+1,y);
        else if(n==0 || n==2) gotoxy(x+1,y);
        else if(n==1) gotoxy(x+2,y);
        else gotoxy(x+1,y);
    }
}
/*
-----
                カーソル・ページアップ
-----
*/
void    ed_rup()
{
    int x,y;

    y=wherey();
    x=wherex();
    if(AreaFlag!=0) return;
    if((linep1+(SCRNL-y))>editendlino) return;
    else linep+=(SCRNL-y);
    gotoxy(1,2);
    condisp(linep1);
    textcursor(DISP_CURSOR);
    gotoxy(x,y);
    linep+=(y-2);
}
/*
-----
                カーソル・ページダウン
-----
*/
void    ed_rdown()
{
    int x,y;

    if(AreaFlag!=0) return;
    y=wherey();
    x=wherex();
    if(linep1<23) linep=1;
    else linep-=((y-2)+23);
    gotoxy(1,2);
    condisp(linep1);
    textcursor(DISP_CURSOR);
    gotoxy(x,y);
    linep+=(y-2);
}
/*
-----
                カーソル・アップ
-----
*/
void    ed_up()
{
    int x,y,i;

```

```
char c;

x=wherex();
y=wherey();
if((linep!=0){
    linep=1;
    return;
})
else{
    if(y==2){
        if(AreaFlag>=2){
            if(AreaFlag==2){
                if((linep==areastrline){
                    textcolor(co.kakuninn);
                    textreverse(REVERSE);
                }
                else{
                    textcolor(co.editor);
                    textreverse(NOREVERSE);
                }
            }
            else setedcolor(linep);
            gotoxy(1,2);
            condisp1(1,linep);
        }
        linep--;
        gotoxy(1,2);
        insline();
        setedcolor(linep);
        condisp1(1,linep);
        gotoxy(x,2);
    }
    else if(y>2){
        if(AreaFlag>=2){
            if(AreaFlag==2){
                if((linep==areastrline){
                    textcolor(co.kakuninn);
                    textreverse(REVERSE);
                }
                else{
                    textcolor(co.editor);
                    textreverse(NOREVERSE);
                }
            }
            else setedcolor(linep);
            gotoxy(1,y);
            condisp1(1,linep);
        }
        y--;
        linep--;
        if(AreaFlag>=2){
            gotoxy(1,y);
            setedcolor(linep);
            condisp1(1,linep);
        }
        gotoxy(x,y);
    }
}
)
)
/*
-----
カーソル・ダウン
-----
*/
void ed_down()
{
    int x,y;

    x=wherex();
    y=wherey();
    if((linep>=BUFLen) return;
    if(y==SCRNL){
        gotoxy(1,2);
        delline();
    }
}
```

```

        gotoxy(1,SCRNL);
        linep++;
        setedcolor(linep1);
        condispl(1,linep1);
    }
    else(
        y++;
        linep++;
        if(AreaFlag==2){
            gotoxy(1,y);
            setedcolor(linep1);
            condispl(1,linep1);
        }
    )
    gotoxy(x,y);
}
/*
-----
カーソル・バック 1字消去 [BS]
-----
*/
void ed_back()
{
    int x,y;

    x=wherex();
    if(x==1 && linep==1) return;
    else if(x==1){
        ed_up();
        gotoxy(linenum[linep1],wherey());
        pmtline();
    }
    else ed_left();
}
/*
-----
1字消去 [DEL]
-----
*/
void ed_del()
{
    int x,y,p,n;
    char c;

    if(endpos==0){
        gotoxy(1,2);
        clreol();
        return;
    }
    x=wherex();
    y=wherey();
    if(linenum[linep1]<x || linenum[linep1]==0) return;
    p=getpos(x,linep1);
    if(endpos==p){
        editbuf[p]=0;
        endpos--;
        return;
    }
    c=editbuf[p];
    n=chkjis(x,linep1);
    if(n==1 || n==2){
        if(n==2){
            ed_left();
            p-=1;
        }
        dshtmov(2,p);
    }
    else dshtmov(1,p);
    if(c==CR){
        editendline--;
        reload(linep1);
        gotoxy(1,y);
        condispl(linep1);
    }
}

```

```

    else{
        reload(linep1);
        if((linenum[linep1]<SCRNL2)  condispl(x,linep1);
        else condispl(linep1);
    }
    gotoxy(x,y);
}
/*
-----
1 ライン削除
-----
*/
void line_del()
{
    int x,y,p1,p2,n;

    x=wherex();
    y=wherey();
    p1=linepos[linep1];
    p2=linepos[linep1+1];
    if(editendline==linep1) p2=endpos;
    n=p2-p1;
    dshtmov(n,p1);
    reload(linep1);
    delline();
    if((linep1+SCRNL-y)>editendline) return;
    gotoxy(1,SCRNL);
    condispl(1,(linep1+(SCRNL-y)));
    gotoxy(x,y);
}
/*
-----
行範囲複写
-----
*/
void ed_copy()
{
    int i,x,y,p0,p1,p2,n1,n2;

    x=wherex();
    y=wherey();
    p0=linepos[linep1];
    p1=linepos[areastrline];
    p2=linepos[areaendline+1];
    n1=areaendline-areastrline+1;
    if((editendline>=linep1 && (editendline+n1)>=BUFLEN)
        || (editendline<linep1 && (linep1+n1)>=BUFLEN)){
        errordisp("便せんの余りが有りません。");
        window(1,1,80,24);
        gotoxy(x,y);
        ed_hanioff();
        textcursor(DISP_CURSOR);
        gotoxy(x,y);
        return;
    }
    n2=p2-p1;
    if(editendline>linep1){
        ushtmov(n2,p0);
        for(i=0;i<n2;i++){ editbuf[p0+i]=editbuf[p1+i];
            endpos+=n2;
        }
        reload(linep1);
    }
    else{
        for(i=editendline;i<linep1;i++){
            editbuf[endpos]=CR;
            linenum[editendline]++;
            editendline++;
            endpos++;
            linepos[editendline]=endpos;
        }
        for(i=0;i<n2;i++){
            editbuf[endpos]=editbuf[p1+i];
            endpos++;
        }
    }
}

```

```

        reload(B);
    }
    CopyFlag=0;
    AreaFlag=0;
    gotoxy(1,2);
    condisp(line1-(y-2));
    gotoxy(x,y);
}
/*
.....
:
:      各種ユーティリティー
:
:.....
*/

/*
-----
      display console buffer
-----
*/
void    condisp(int line)
{
    int x,y,y1,i;

    x=wherex();
    y=wherey();
    setedcolor(line);
    if(condisp1(x,line)==FALSE){
        if(y<SCRNL) conclr(y+1);
        return;
    }
    line++;
    for(i=y+1;i<=SCRNL;i++){
        gotoxy(1,i);
        setedcolor(line);
        if(condisp1(1,line)==FALSE) break;
        line++;
        if((line>editendline)          break;
    }
    y1=wherey();
    if(y1<SCRNL)    conclr(y1+1);
}
/*
-----
      one line display on console
-----
*/
int condisp1(int x,int line)
{
    char    data[81],c;
    int i,p,n;
    unsigned short ch;

    textcursor(NODISP_CURSOR);
    if(linenum[line]==0){
        gotoxy(1,wherey());
        clreol();
        return FALSE;
    }
    setmen(data,81,0);
    p=getpos(x,line);
    n=linenum[line]-x+1;
    strncpy(data,&editbuf[p],n);
    if(data[n-1]==CR){
        data[n-1]=0;
        cputs(data);
        crret();
    }
    else    cputs(data);
    clreol();
    textreverse(NOREVERSE);
    return YES;
}

```

```

/*
-----
          ライン表示用カラーセット
-----
*/
void      setedcolor(int line)
{
    if(AreaFlag<=1){
        textreverse(NOREVERSE);
        textcolor(co.editor);
    }
    else if(AreaFlag==2){
        if(line>=areastrline){
            textreverse(REVERSE);
            textcolor(co.kakuninn);
        }
        else{
            textreverse(NOREVERSE);
            textcolor(co.editor);
        }
    }
    else if(AreaFlag==3){
        if(line>=areastrline && line<=areaendline){
            textreverse(REVERSE);
            textcolor(co.kakuninn);
        }
        else{
            textreverse(NOREVERSE);
            textcolor(co.editor);
        }
    }
}
/*
-----
          console clear
-----
*/
void      conclr(int y)
{
    int x1,y1;

    x1=wherex();
    y1=wherey();
    textcursor(NODISP_CURSOR);
    window(1,y,80,SCRNL);
    clrscr();
    window(1,1,80,24);
    gotoxy(x1,y1);
    textcursor(DISP_CURSOR);
}
/*
-----
          get the buffer pointer of data on CRT
-----
*/
int getpos(int x,int line)
{
    int p;

    p=linepos(line)+x-1;
    return p;
}
/*
-----
          down shift remove one charactor
-----
*/
void      dshtmov(int n,int p)
{
    memmove(&editbuf[p],&editbuf[p+n],(endpos-p-n));
    endpos-=n;
    setmem(&editbuf[endpos],n,0);
}

```



```

/*
-----
up shift remove n charactors
-----
*/
void    ushtmov(int n,int p)
{
    memmove(&editbuf[p+n],&editbuf[p],(endpos+1-p));
    endpos+=n;
}
/*
.....
:
:          C R T 画 面 出 力 モ ジ ュ ー ル
:
:
:.....
*/
-----
write new mode message on prompt line
-----
*/
void    pmtmode(char *s)
{
    int i,x,y;
    char    buff[31];
    char    *str;

    x=wherex();
    y=wherey();
    window(1,1,80,24);
    gotoxy(1,1);
    textcolor(co.box);
    textcursor(NODISP_CURSOR);
    textreverse(REVERSE);
    for(i=0;i<52;i++)    putchar(' '); /* ライン・クリア */
    gotoxy(3,1);
    textcolor(co.title);
    cprintf("%s",s);
    textcolor(co.box);
    cprintf(" ");
    if(!EdFile==0)    cprintf(" [ ファイル無し ] ");
    else{
        setmem(buff,31,0);
        if(strlen(EdFile)>30){
            strncpy(buff,EdFile,1);
            str=strchr(EdFile,0x00ff&0x5c); /* ¥ */
            strcat(buff,str);
        }
        else    strcpy(buff,EdFile);
        cprintf(" [%s] ",buff);
    }
    window(1,1,80,24);
    gotoxy(x,y);
    textcolor(co.editor);
    textcursor(DISP_CURSOR);
    textreverse(NOREVERSE);
}
/*
-----
change mode on prompt line to edit
-----
*/
void    pmtedit()
{
    if(ins_flag==0)    pmtmode(" 便ㇿン : 挿入 ");
    else                pmtmode(" 便ㇿン : 上書 ");
}
/*
-----
最上行の表示
-----
*/
void    pmtupperdisp()

```

```

{
    pmtedit();
    pmlline();
    pmcol();
    textreverse(NOREVERSE);
    textcolor(co.editor);
    textcursor(DISP_CURSOR);
}
/*
-----
print the line number on the prompt line
-----
*/
void    pmlline()
{
    char    data[10],ch;
    int i;

    setmem(data,10,0);
    sprintf(data,"ライン:%03d",linep);
    for(i=0;i<10;i++){
        ch=data[i];
        if(ch==0)    break;
        pokeb(0xa000,(53+i)*2,ch);
        pokeb(0xa000,(53+i)*2+1,0);
    }
}
/*
-----
print column number of the cursor
-----
*/
void    pmcol()
{
    char    data[20],ch;
    int i,x;

    x=wherex();
    setmem(data,20,0);
    sprintf(data,"カラム:%02d 行:%03d",x,editendline);
    for(i=0;i<20;i++){
        ch=data[i];
        if(ch==0)    break;
        pokeb(0xa000,(61+i)*2,ch);
        pokeb(0xa000,(61+i)*2+1,0);
    }
}
/*
-----
replace line numbers and line data numbers
-----
*/
void    reload(int line)
{
    int i,p;
    char    c;

    p=linepos[line];
    setmem(&linenum[line],(BUFLen-line),0);
    setmem(&linepos[line+1],(BUFLen-line-1),0);
    i=0;
    while(p<endpos){
        c=editbuf[p];
        if(c!=CR && c!=' ')    break;
        else if(iskanji(c)!=0){
            if(i>=SCRNW2){
                if(++line>=BUFLen)    break;
                linepos[line]=p;
                i=0;
            }
            linenum[line]++;
            p+=2;
            i+=2;
        }
    }
}

```

```

    else if (c==CR){
        linenum[line]++;
        if(++line>BUFLen) break;
        linepos[line]=++p;
        i=0;
    }
    else{
        if(i>=SCRNW1){
            if(++line>BUFLen) break;
            linepos[line]=p;
            i=0;
        }
        p++;
        i++;
        linenum[line]++;
    }
}
if(p<endpos){
    setmem(&editbuf[p],(endpos-p),0);
    endpos=p;
}
editendline=line;
}
/*
-----
display carries and return
-----
*/
void crret()
{
    int x;

    x=wherex();
    if(x==80) return;
    textcolor(co.control);
    putch('<');
    textcolor(co.editor);
}
/*
-----
check jis
-----
*/
int chkjis(int x,int line)
{
    char *p;
    unsigned int n,i;
    char buf[81];
    char c;

    setmem(buf,81,0);
    p=&editbuf[linepos[line]];
    for(i=0;i<80;i++){
        c=*(p+i);
        if(c==CR) break;
        buf[i]=c;
    }
    n=athctype(buf,x-1);
    if(n==CT_ANK) return 0;
    else if(n==CT_KJ1) return 1;
    else if(n==CT_KJ2) return 2;
}
/*
-----
行範囲処理
-----
*/
void ed_haniset()
{
    int x,y;

    if(AreaFlag==1){
        x=wherex();
        y=wherey();
        /* 行範囲設定 */
    }
}

```

```

x_pos=x;
y_pos=y;
pmtmode("終行設定");
areastrline=linepl;
dispstrline=linepl-(y-2);
gotoxy(1,y);
textreverse(REVERSE);
textcolor(co,kakuninn);
condispl(1,linepl);
textcursor(DISP_CURSOR);
gotoxy(x,y);
AreaFlag=2;
}
else if (AreaFlag==2) {
    if (areastrline<=linepl) {
        areaendline=linepl;
        AreaFlag=3; /* 終行設定 */
        pmtmode("設定終了");
    }
}
else if (AreaFlag==3 && CopyFlag!=0) {
    ed_copy();
    pmtedit();
    AreaFlag=0;
    CopyFlag=0;
}
}
/*
-----
            エディットバッファのクリア
-----
*/
void    edbufnew()
{
    setmem(editbuf,BUFNUM,0);
    setmem(linenum,BUFLEN,0);
    setmem(linepos,BUFLEN,0);
}
/*
-----
            エディタの初期化
-----
*/
void    edinit()
{
    setmem(edsrdbuf,60,0);
    setmem(EdFile,SYSFNMAX,0); /* ファイル名はクリア */
    linep=1; /* 表示スタートライン */
    ins_flag=0; /* insert mode flag */
    AreaFlag=0; /* 範囲設定フラグ */
    CopyFlag=0; /* コピーフラグ */
    areastrline=0; /* 範囲スタートライン */
    areaendline=0; /* 範囲エンドライン */
    dispstrline=0; /* 画面表示スタートライン */
}

```



```

    gettext(1,1,80,24,ConnectBuff);
    TimeOver=0;
    ConnectFlag=-1;
    connect_x=wherex();
    connect_y=wherey();
    sleep(1);
}
    return CONNECT;
}

/*
.....
*
*           L A N 端 末 接 続
*
*
*.....
*/
int computer()
{
    if( ConnectFlag!=0 || ModemFlag!=0 )    return CONNECT;
    window(1,1,80,24);
    clrscr();
    if ( initialize()!=YES )    return OPEN;
    if ( settei( RET_ON )!=-1 || cp.ltype==0 ) return OPEN;
    timerread(ConnectTime);
    ConnectSec=secread();
    window(1,1,80,24);
    clrscr();
    setatrib();
    gettext(1,1,80,24,ConnectBuff);
    TimeOver=0;
    ConnectFlag=-1;
    connect_x=wherex();
    connect_y=wherey();
    EtherFlag=YES;
    EtherFin=FALSE;
    rcvp=rcvpmx=rcvlen=rcvflg=0;
    setmem( rbuf,RBUFSIZE,0 );
    setmem( sbuf,SBUFSIZE,0 );
    if ( tolcon()==FALSE ){
        ConnectFlag=0;
        ModemFlag=0;
        EtherFlag=FALSE;
        EtherFin=YES;
        return OPEN;
    }
    return CONNECT;
}

/*
.....
*
*           接 続 モ ー ド
*
*
*.....
*/
int connect()
{
    int s,x,y,o,flg;
    unsigned int t;

    BsFlag=0;
    window(1,1,80,25);
    clrscr();
    underlinedisp();
    puttext(1,1,80,24,ConnectBuff);
    window( 1,1,80,24 );
    gotoxy(connect_x,connect_y);
    textcursor(DISP_CURSOR);
    setatrib();
    TimeOver=0;
    s=0;
    ModemFlag=-1;
    mouseon();
    MS_LB=MS_RB=0;
    while( 1 )

```

```

(
    if (TimerFlag!=0 && TimeOver>10000){          /*..... タイマー表示 .....*/
        contimedisp();
        TimeOver=0;
        s=0;
    }
    if ( cp.ltype==0 && ModemFlag!=0 && ConnectFlag!=0
        && (0x0080&siosts())==0){ /*... SIOのステータス・チェック...*/
        t=0;
        while(t<50){
            mouseput();
            delay(100);
            if ((0x0080&siosts())!=0)      break;
            t++;
        }
        if ( t==50 ){
            mouseoff();
            textcursor( NODISP_CURSOR );
            siostop();
            mesdisp(" 電話回線が切断されました。");
            ConnectFlag=0;
            ModemFlag=0;
            c=OPEN;
            break;
        }
    }
    if ( MS_LB ) {          /* 左ボタンを押した? */
        mouseoff();
        textcursor( NODISP_CURSOR );
        flg=con_funkey();
        if ( flg==8 ){
            c=OPEN;          /* メニュー呼出し */
            break;
        }
        mouseon();
        textcursor( DISP_CURSOR );
    }
    if ( EtherFlag != YES && cp.ltype==1 && LagFlag==FALSE
        ;: rcvflg==0 && LagFlag==YES )
    {
        telcut();
        ConnectFlag=0;
        ModemFlag=0;
        mouseoff();
        mesdisp(" 通信終了しました。");
        c=OPEN;
        break;
    }
    mouseput();
    switch( s )          /* ポート入力、キー入力 */
    {
        case 0:s=fun_keyin();break;
        case 1:s=fun_siain();break;
        default:break;
    }
    readxyprt();
    gettext(1,1,90,24,ConectBuff);
    if ( c==0 ) return OPEN;
    return c;          /* セレクタに復帰 */
}
/*
-----
               キー入力処理
-----
*/
int fun_keyin()
{
    int      s,q,c;
    unsigned short  kc1;
    unsigned short  kc2;
    unsigned short  kc;
    char      c1,c2;
    union  REGS  regs;

```



```

if( (c=key_in())==0 ){          /* キー入力あり? */
    if(TimerFlag!=0)    TimeOver++;
    else                TimeOver=0;
    return 1;
}
if(KeyNoFlag!=0)            return 1;
if( c=='z' ) c1=FUN10; else c1=(char)((c>>8)&0x00ff);
c2=(char)(0x00ff&c);
if(c1!=0 && c2==0){          /* 特殊文字だった? */
    switch(c1)
    {
        case FUN10:MS_LB=1;return 1;
        case HOME: /* HOME */
            if(Key98Flag==0){
                ksiooutput(0x000b); return 1;
            }
            else{
                if(vt.KeyPad==0)
                    escsioout("[1~");
                else
                    escsioout("OP");
                return 1;
            }
        case HELP: /* HELP */
            if(Key98Flag==0){
                ksiooutput(0x00ff&'?'); return 1;
            }
            else{
                if(vt.KeyPad==0) escsioout("{4~");
                else
                    escsioout("O0");
                return 1;
            }
        case UP: /* up */
            if(Key98Flag==0){
                ksiooutput(0x001e); return 1;
            }
            else{
                if(vt.CursorKey!=0)
                    escsioout("OA");
                else
                    escsioout("[A");
                return 1;
            }
        case DOWN: /* down */
            if(Key98Flag==0){
                ksiooutput(0x001f); return 1;
            }
            else{
                if(vt.CursorKey!=0)
                    escsioout("OB");
                else
                    escsioout("[B");
                return 1;
            }
        case RIGHT: /* right */
            if(Key98Flag==0){
                ksiooutput(0x001c); return 1;
            }
            else{
                if(vt.CursorKey!=0)
                    escsioout("OC");
                else
                    escsioout("[C");
                return 1;
            }
        case LEFT: /* left */
            if(Key98Flag==0){
                ksiooutput(0x001d); return 1;
            }
            else{
                if(vt.CursorKey!=0)
                    escsioout("OD");
            }
    }
}

```

```

        else
            escsioout("(D");
            return 1;
    }
    case DEL: /* del */
        ksiooutput(0x00ff&DL); return 1;
    case INS:
        if(Key98Flag==0){
            ksiooutput(0x000b); return 1;
        }
        else{
            escsioout("(2~"); return 1;
        }
    case RUP:
        if(Key98Flag!=0) escsioout("(5~");
        return 1;
    case RDOWN:
        if(Key98Flag!=0) escsioout("(6~");
        return 1;
    default: break;
}
return 1;
}
else if(c1==0 && c2!=0){ /* 入力が漢字だったか? */
    if(!iskanji(0x00ff&c2)!=0){ /* うん */
        c1=getch();
        kc1=0xff00&(0x00ff&c2)<<8;
        kc2=0x00ff&c1;
        kc=kc1|kc2;
    }
    else kc=0x00ff&c2; /* いいや */
    ksiooutput(kc); /* 出力 */
    return 1;
}
}
/*

```

シリアルインターフェイスからのデータ入力

```

/*
・ バイト : アスキー、ワード : シフト J I S ・
int fun_sioin()
{
    unsigned short o_data;
    char o_data1;
    char o_data2;
    int x,y;

    if(sioinum()==0){
        if(TimerFlag!=0) TimeOver++;
        else TimeOver=0;
        return 0;
    }
    o_data=ksioinput();
    if(o_data==0) return 0; /* NULL CODE */
    o_data1=(char)(0x00ff&(_rotr(o_data,8)));
    o_data2=(char)(0x00ff&o_data);
    funcoutput(o_data1,o_data2);
    if(o_data1==0){
        if(BsFlag!=0){
            if(o_data2!=BS) BsFlag=0;
        }
        if(o_data2==' '){
            check_pos1(o_data2);
            writeCharBuff(0,o_data2);
        }
    }
    else{
        switch(o_data2)
        {
            case CR:
                putch(CR);
                writeCharBuff(0,CR);
                if(cp.sndcr!=0){
                    outlf();
                }
            }
        }
    }
}

```

```

        linebufdisp();
        writeCharBuff(0,LF);
    }
    break;
case LF:
    if(cp.sndlf!=0){
        putchar(CR);
        writeCharBuff(0,CR);
    }
    outlf();
    linebufdisp();
    writeCharBuff(0,LF);
    break;
case FF:
    x=wherex();
    y=wherey();
    window(1,ScrollTp,80,ScrollBm);
    clrscr();
    window(1,1,80,24);
    gotoxy(x,y);
    break;
case BS:
    if(BsFlag==0){
        putchar(BS);
        writeCharBuff(0,BS);
        x=wherex();
        y=wherey();
        if(peekb(0xa000,BSP0S)==0) BsFlag=0;
        else BsFlag=0xff;
    }
    else BsFlag=0;
    break;
case DL:
    putchar(DL);
    writeCharBuff(0,DL);
    break;
case TAB:
    putchar(TAB);
    writeCharBuff(0,o_data2);
    break;
case BELL: putchar(BELL); break;
case UT: cursorup(1); break;
default: break;
    }
}
else{
    BsFlag=0;
    check_pos2(o_data1,o_data2);
    writeCharBuff(o_data1,o_data);
}
return 0;
}

```

バイトキャラクタのスクロール処理

```

/*
void check_pos1(char ch)
{
    int x,y;

    x=wherex();
    y=wherey();
    if(x==80){
        if(vt.AutoLap==0){
            if(y!=24){
                putchar(ch);
                gotoxy(x,y);
            }
        }
        else{
            window(1,1,80,25);
            putchar(ch);
        }
    }
}

```

```

        window(1,1,80,24);
        gotoxy(x,y);
    }
}
else{
    putch(ch);
    if(y==ScrollBm
    && (ScrollTp!=2 :: ScrollBm!=24)){
        gotoxy(1,ScrollTp);
        delline();
        gotoxy(x,y);
        insline();
    }
}
}
else    putch(ch);
}

```

/*

漢字キャラクタのスクロール処理

*/

```

void    check_pos2(char ch1,char ch2)
{
    int x,y;

    if(vt.AutoLap!=0){
        x=wherex();
        y=wherey();
        if(x>79){
            if(y<=ScrollTp && y>ScrollBm){
                putch(ch1);
                putch(ch2);
                gotoxy(x,y);
            }
            else if(y==ScrollBm){
                window(1,1,80,25);
                putch(ch1);
                putch(ch2);
                window(1,ScrollTp,80,ScrollBm);
                y=y-ScrollBm+1;
                gotoxy(x,y);
            }
        }
        else{
            putch(ch1);
            putch(ch2);
        }
    }
    else{
        putch(ch1);
        putch(ch2);
    }
}

```

/*

ラインフィード出力

*/

```

void    outlf()
{
    int x,y;

    x=wherex();
    y=wherey();
    if(ScrollTp!=2 :: ScrollBm!=24){
        gotoxy(1,ScrollTp);
        delline();
        gotoxy(x,y);
        insline();
    }
    else    putch(LF);
}

```

```

}

/*
-----
各種コマンド入力
-----
*/
int con_funkey()
{
    static char conmenu[9][DATA_WD]={
        "便せん送信","1行送信","ファイル転送","機能設定",
        "ブレイク","回線切断","予約コマンド","メインメニュー"};

    static WINDOW wd=(28,3,26,18,T_CYAN,T_YELLOW,T_WHITE,0,"接続モード",2,8,0,0,1,1,0)
    ;
    int x,y,s,wflg,flg;

    x=wherex();
    y=wherey();
    wflg=WD_OPEN;
    flg=YES;
    while( flg==YES ){
        s=twindow(1, wflg,&wd,conmenu );
        wflg=WD_REMAIN;
        switch( s )
        {
            case 0: window( 1,1,80,24 );
                    gotoxy(x,y);
                    if ( edsend()==YES ) wflg=WD_OPEN;
                    break;
            case 1: window( 1,1,80,24 );
                    gotoxy(x,y);
                    if ( lined()==YES ) flg=FALSE;
                    break;
            case 2: window( 1,1,80,24 );
                    gotoxy(x,y);
                    if ( ftrans()==YES ) flg=FALSE;
                    break;
            case 3: setflag(); break;
            case 4: breakcry(); break;
            case 5: if ( connect()==YES ) flg=FALSE;
                    break;
            case 6: window( 1,1,80,24 );
                    gotoxy(x,y);
                    if ( yoyakucom()==YES ) flg=FALSE;
                    break;
            case 7: flg=8; /* メインメニューに復帰 */
                    break;
            default: flg=FALSE;break;
        }
    }
    textcursor( NODISP_CURSOR );
}
alliclose();
window( 1,1,80,24 );
gotoxy(x,y);
return flg;
}

/*
-----
接続モードにおける時刻表示
-----
*/
void contimedisp()
{
    char TimeCrtSave[1500]; /* 時刻表示部分のセーブエリア */
    int x,y;

    x=wherex();
    y=wherey();
    MS_RB=0;
    MS_LB=0;
    delay( 100 );
    window( 1,1,80,24 );

```

```

    gettext(1,1,80,4,TimeCrtSave);
    while( key_gets()==0 && siornum()==0 && MS_LB==0 && MS_RB==0 ){
        mouseput();
        timerdisp(ConnectTime,ConnectSec);
    }
    textcolor( T_WHITE );
    textcursor( DISP_CURSOR );
    window( 1,1,80,24 );
    puttext(1,1,80,4,TimeCrtSave);
    gotoxy(x,y);
}

/*
-----
                便箋内容送信
-----
*/

extern int      AreaFlag;
extern int      CopyFlag;
extern int      areastrline;
extern int      areasendline;
extern int      editendline;
extern char      editbuff[];
extern int      linepos[];
extern int      linenum[];

/* 範囲設定フラグ */
/* コピー設定フラグ */
/* 範囲スタートライン */
/* 範囲エンドライン */
/* エディタバッファ最終行 */

int edsend()
{
    int s,i,x,y,flg=FALSE;
    static WINDOW wd=( 2,3,30,5,T_CYAN,T_YELLOW,T_WHITE,1,"手紙送信モード"
        ,2,3,1,0,1,0,0 );
    static char data[][DATA_WD]={ " 手紙を送信します。"
        , " はい "
        , " いいえ " };

    x=wherex();
    y=wherey();
    s=twindow(1, WD_OPEN,&wd,data );
    twindow( WD_CLOSE );
    window( 1,1,80,24 );
    gotoxy( x,y );
    if ( s==1 )
    {
        flg=YES;
        allclose();
        textreverse(NOREVERSE);
        if (AreaFlag==0){
            for(i=0;i<BUFLN;i++){
                if (line_trs(i)==FALSE) break;
            }
        }
        else if (AreaFlag==3){
            for(i=areastrline;i<=areasendline;i++){
                if (line_trs(i)==FALSE) break;
            }
        }
        if (siornum()==0) delay(100);
        textreverse(NOREVERSE);
    }
    window( 1,1,80,24 );
    gotoxy(x,y);
    return flg;
}

/*
-----
                エディタ内1行送信
-----
*/

int line_trs(int line)
{
    int i,p,d,s;
    unsigned int kc,kc1,kc2;

```

```

char    ch;

p=linepos[line];
for(i=0;i<linenum[line];i++){
    ch=editbuf[p+i];
    if(ch!=CR && ch<' '){ break;
    else if(iskanji(0x00ff&ch)!=0){
        kc1=0xff00&((0x00ff&ch)<<8);
        i++;
        ch=editbuf[p+i];
        kc2=0x00ff&ch;
        kc=kc1|kc2;
    }
    else    kc=0x00ff&ch;
    ksiooutput(kc);
    if(siornum()==0)    delay(80);
    while(siornum()!=0) fun_sioin();
    if((char)key_in()==ESC) return FALSE;
}
if(i==0)    return FALSE;
else{
    if(kc!=(0x00ff&CR)) ksiooutput(0x00ff&CR);
    return YES;
}
}

/*
-----
                        切          断
-----
*/
int connect()
{
    static WINDOW wd=( 2,3,40,5,T_RED,T_MAGENTA,T_WHITE,1,"回線切断モード",2,3,1,0,1,0,
    0 );
    static char data[][DATA_WD]=
    { " 回線を切断します。よろしいですか?",
      "      はい",
      "      いいえ" };

    int s;
    if ( ModemFlag == 0 ) errordisp("すでに切断されています。");
    else {
        s=twindow(1, WD_OPEN,&wd,data );
        twindow( WD_CLOSE );
        if ( s==1 ) {
            if ( cp.ltype==0 )
            {
                siostop();
                mesdisp(" 電話回線が切断されました。");
                ConnectFlag=0;
                ModemFlag=0;
                return YES;
            }
            else {
                EtherFlag=FALSE;
                if ( telcut()==YES ){
                    ConnectFlag=0;
                    ModemFlag=0;
                    return YES;
                }
            }
        }
    }
    return FALSE;
}

/*
-----
                        フラグ設定
-----
*/
void    setflag()
{
    static WINDOW wd=

```

```

    ( 30,8,26,7,T_CYAN,T_YELLOW,T_WHITE,0,"機能フラグ設定".2,5,0,0,1,1,0 );
static char menu[5][DATA_WD]=
    (" ハードコピー"," タイマー表示",
     " 制御ポート表示"," PC9801/UT100"," 機能設定終了");
int FLAG[4],s,i,wflag;

FLAG[0]=PrintFlag;
FLAG[1]=TimerFlag;
FLAG[2]=CodeFlag;
FLAG[3]=Key98Flag;
wflag=WD_OPEN;
while( 1 )
{
    for(i=0;i<4;++i) menu[i][0]=( FLAG[i]==0 ? ' ': '*' );
    if ( (s=twindow(1, wflag,&wd,menu ))== -1 ) break;
    if ( s==4 ) break;
    wflag=WD_REMAIN;
    FLAG[s]=( FLAG[s]==0 ? -1 : 0 );
    PrintFlag=FLAG[0];
    TimerFlag=FLAG[1];
    CodeFlag=FLAG[2];
    Key98Flag=FLAG[3];
    underlinedisp();
    textreverse( NOREVERSE );
}
twindow( WD_CLOSE );
}

/*
-----
                     ブレークキー処理
-----
*/
void breakcry()
{
    static WINDOW wd=( 2,3,30,3,T_CYAN,T_WHITE,T_WHITE,0,"",1,1,0,2,0,0,0 );
    static char data[1][DATA_WD]=(" 現在、ブレイク信号送出中");
    twindow(1, WD_OPEN,&wd,data );
    siobreak();
    twindow( WD_CLOSE );
}

/*
-----
                     行編集コマンド実行
-----
*/
int lined()
{
    static WINDOW wd=( 2,6,44,12,T_CYAN,T_YELLOW,T_WHITE,0," 1行送信".0,10,0,0,1,1,-1 );
    static char chat[10][DATA_WD],ch,NAME[80];
    int s,i;
    FILE *fp;

    sprintf( NAME,"%s%stobihino.lin",ENV );
    if ( (fp=fopen(NAME,"rb")) != NULL )
    {
        fread( chat,10*DATA_WD,1,fp );
    }
    fclose( fp );
    s=twindow(1, WD_OPEN,&wd,chat );
    twindow( WD_CLOSE );
    fp=fopen(NAME,"wb");
    fwrite( chat,10*DATA_WD,1,fp );
    fclose( fp );
    if ( s != -1 ) /* 1行送信 */
    {
        allclose();
        for(i=0;i<77;i++){
            ch=chat[s][i];
            if ( ch==0 ) break;
            ksiooutput(ch);
        }
    }
}

```



```

        ksiooutput(CR);
        return YES;
    }
    return FALSE;
}
/*
-----
予約コマンド実行
-----
*/
int yoyakucon()
{
    static WINDOW wd0=( 3,12,34,10,T_YELLOW,T_CYAN,T_WHITE,0,"予約コマンドファイル名",
                        ,0,0,0,0,1,1,-1 );
    static WINDOW wd=( 34,3,44,20,T_CYAN,T_YELLOW,T_WHITE,0,"予約コマンド送信",
                        ,0,10,0,0,1,1,-1 );
    static char yoyaku[20][DATA_WD],ch,NAME[80],NAME2[80],comfile[8][DATA_WD];
    int s,s2,i,wflg=WD_OPEN;
    FILE *fp;

    setmem( comfile,8*DATA_WD,0 );
    sprintf(NAME2,"%s¥¥lobi¥hino.ycm",ENU);
    if ( (fp=fopen( NAME2,"rb" )) != NULL ) /* 登録ファイル名に読み込み */
    {
        fread( comfile,8*DATA_WD,1,fp );
    }
    fclose( fp );
    while( 1 )
    {
        if ( (s2=twindow( 1,wflg,&wd0,comfile )) == -1 ) break;
        wflg=WD_REMAIN;
        setmem( yoyaku,20*DATA_WD,0 );
        sprintf(NAME,"%s¥¥¥s",ENU,comfile[s2] );
        if ( (fp=fopen(NAME,"rb")) != NULL )
        {
            fread( yoyaku,10*DATA_WD,1,fp );
        }
        fclose( fp );
        s=twindow(1, WD_OPEN,&wd,yoyaku );
        twindow( WD_CLOSE );
        if ( strlen( comfile[s2] )>0 )
        {
            fp=fopen(NAME,"wb");
            fwrite( yoyaku,10*DATA_WD,1,fp );
            fclose( fp );
        }
        if ( (fp=fopen( NAME2,"wb" )) != NULL ) /* 登録ファイル名のセーブ */
        {
            fwrite( comfile,8*DATA_WD,1,fp );
        }
        fclose( fp );
        if ( s != -1 ) /* 予約コマンド送信 */
        {
            allclose();
            for(i=0;i<77;i++){
                ch=yoyaku[s][i];
                if ( ch==0 || ch=='(' ) break;
                ksiooutput(ch);
            }
            ksiooutput(CR);
            return YES;
        }
    }
    twindow( WD_CLOSE );
    return FALSE;
}
/*
-----
最下行への表示
-----
*/
void underlinedisp()
{

```

```

underbardisp(LogFlag,PrintFlag,TimerFlag,CodeFlag,Key98Flag);
linebufdisp();
}

/*
-----
          ファイル転送コマンド実行
-----
*/

extern int CDsk;
extern char CDir[];
static char f1buff(8000);
int ftrans()
{
    static WINDOW wd={ 10,12,40,9,T_YELLOW,T_CYAN,T_WHITE,0,"ファイル転送",
                      2,7,0,0,1,0,0 };
    static WINDOW wd2={ 2,3,30,5,T_RED,T_CYAN,T_WHITE,1,"".2,3,1,0,1,0,0 };
    static char data2[3][DATA_WD]={"ファイル転送を実行します。",
                                     "はい",
                                     "いいえ"};

    int s,wflg=WD_OPEN,flg,x,y;

    x=wherex();
    y=wherey();
    while( 1 )
    {
        textcursor( NODISP_CURSOR );
        flg=1;
        while( flg )
        {
            if ( ( s=twindow(1, wflg,&wd,xfmenu) )== -1 ) s=7; /* 注意 */
            wflg=WD_REMAIN;
            switch( s )
            {
                case 0: getprotoc(); break;
                case 1: getdirect(); break;
                case 2: getlinecr(); break;
                case 3: getxfwait(); break;
                case 4: getlfpros(); break;
                case 5: getxfile(); break;
                case 6: flg=0;break; /* 実行 */
                default: twindow( WD_CLOSE ); /* 終了 */
                        textcursor( NODISP_CURSOR );
                        setdisk(CDsk);
                        chdir(CDir);
                        gotoxy( x,y );
                        return FALSE;
            }
        }
        s=twindow(1, WD_OPEN,&wd2,data2 );
        twindow( WD_CLOSE );
        if( s != 1 ) continue; /* 転送拒否 */

        allclose();
        window( 1,1,80,25 );
        gettext( 1,1,80,25,f1buff );
        mesdisp("転送を開始します。");
        clrscr();
        if(Protoc==0) /* 無手順 */
        {
            if(Direct==0) tty_receive();
            else tty_transm();
        }
        else if(Protoc==1) /* XMODEM */
        {
            if(Direct==0) xmod_receive();
            else xmod_transm();
        }
        else if(Protoc==2) /* YMODEM */
        {
            if(Direct==0) ymod_receive();
            else ymod_transm();
        }
    }
}

```

```

else if(Protoc==3)      /* K E R M I T */
{
    if(Direct==0)    kern_recive();
    else            kern_transm();
}
textcursor( NODISP_CURSOR );
mesdisp("転送が終了しました。");
window( 1,1,80,25 );
puttext( 1,1,80,25,ftbuff );
window( 1,1,80,24 );
gotoxy( x,y );
setdisk(CDisk);
chdir(CDir);
return YES;
}
}

/*
-----
                転送プロトコルの設定
-----
*/
void    getprotoc()
{
    static char pr[4][DATA_WD]=(" 無手順","XMODEM","YMODEM",
                                ,"KERMIT");
    static WINDOW wd=( 20,3,30,6,T_CYAN,T_YELLOW,T_WHITE,0,"プロトコル",
                        ,2,4,0,0,1,1,0 );
    int s;
    s=twindow(1, WD_OPEN,&wd,pr );
    if ( s != -1 ) Protoc=s;
    twindow( WD_CLOSE );
    sprintf((xfmenu[0]+16),"%s ",pr[Protoc]);
    return;
}
/*
-----
                転送方向の設定
-----
*/
void    getdirect()
{
    static char di[2][DATA_WD]=(" 受信(ｸﾞﾗﾝ)", "送信(ﾌﾞｯﾌﾟ)");
    static WINDOW wd=( 20,3,30,4,T_CYAN,T_YELLOW,T_WHITE,0,"転送方向",
                        ,2,2,0,0,1,0,0 );
    int s;
    s=twindow(1, WD_OPEN,&wd,di );
    if ( s != -1 ) Direct=s;
    twindow( WD_CLOSE );
    sprintf((xfmenu[1]+10),"%s ",di[Direct]);
    return;
}
/*
-----
                無手順転送LF処理
-----
*/
void    getlfpros()
{
    static char lf[2][DATA_WD]=(" 無視","付き");
    static WINDOW wd=( 20,3,30,4,T_CYAN,T_YELLOW,T_WHITE,0,"ラインフィード",
                        ,2,2,0,0,1,0,0 );
    int s;
    s=twindow(1, WD_OPEN,&wd,lf );
    if ( s != -1 ) LineFeed=s;
    sprintf((xfmenu[4]+14+6),"%s ",lf[LineFeed]);
    twindow( WD_CLOSE );
    return;
}
/*
-----
                強制改行字数設定
-----
*/

```

```

void    getlinecr()
{
    static WINDOW wd=( 20,3,30,3,T_CYAN,T_YELLOW,T_WHITE,0,"改行字数/行"
                      ,0,1,0,0,1,0,0 );
    static char data[1](DATA_WD);

    sprintf(data[0],"%d",LineNo);
    if ( twindow(1, WD_OPEN,&wd,data )!=-1 )
    {
        LineNo=atoi(data[0]);
        if ( LineNo>999 ) LineNo=999;
    }
    twindow( WD_CLOSE );
    sprintf(xfmenu[2]+14,"%3d ] ",LineNo);
    return;
}

/*
-----
                        送信後待ち時間
-----
*/
void    getxfwait()
{
    static WINDOW wd=( 20,3,20,3,T_CYAN,T_YELLOW,T_WHITE,0,"待ち時間"
                      ,0,1,0,0,1,0,0 );
    static char data[1](DATA_WD);
    sprintf(data[0],"%d",XfWait);
    if ( twindow(1, WD_OPEN,&wd,data )!=-1 )
    {
        XfWait=atoi(data[0]);
        if ( XfWait>99 ) XfWait=99;
    }
    twindow( WD_CLOSE );
    sprintf((xfmenu[3]+10),"%2d(秒) ] ",XfWait);
}

/*
-----
                        マルチファイル名の取得
-----
*/
void getMultiFile( void )
{
    static WINDOW wd=( 2,10,44,12,T_YELLOW,T_YELLOW,T_WHITE,0,"マルチファイル名入力"
                      ,0,10,0,0,1,1,-1 );

    int wflg=WD_OPEN;

    while( 1 )
    {
        if ( twindow( 1,wflg,&wd,MultiFile )== -1 ) break;
        wflg=WD_REMAIN;
        if ( *MultiFile[wd.1]!='\0' ) getfilnam(RET_OFF,&MultiFile[wd.1][0]);
    }
    twindow( WD_CLOSE );
}

/*
-----
                        転送ファイル名の取得
-----
*/
void    getxfile()
{
    if(Protoc==0 :: Protoc==1)
    {
        getfilnam(RET_ON,&MultiFile[0][0]);
        sprintf( xfmenu[5]+15,"%-s",MultiFile[0] );
    }
    else getMultiFile();
}

/*
-----
                        接続画面のカーソル位置の取得
-----

```

```

/*
void readxyvert()
{
    connect_x=wherex();
    connect_y=wherey();
}

/*
-----
カーソルのアトリビュート
-----
/*
void setattrib()
{
    int i,p;

    normvideo();
    for(i=0;i<AtribNum;i++){
        p=CrtAtrib[i];
        switch(p)
        {
            case 0: normvideo(); break;
            case 2: textvertical(VERTICAL); break;
            case 4: textlunder(UNDERLINE); break;
            case 5: textblink(BLINK); break;
            case 7: textreverse(REVERSE); break;
            case 8: textattr(SECRET); break;
            case 16: textattr(SECRET); break;
            case 17: textcolor(T_RED); break;
            case 18: textcolor(T_BLUE); break;
            case 19: textcolor(T_MAGENTA); break;
            case 20: textcolor(T_GREEN); break;
            case 21: textcolor(T_YELLOW); break;
            case 22: textcolor(T_CYAN); break;
            case 23: textcolor(T_WHITE); break;
            case 30: textcolor(T_BLACK); break;
            case 31: textcolor(T_RED); break;
            case 32: textcolor(T_GREEN); break;
            case 33: textcolor(T_YELLOW); break;
            case 34: textcolor(T_BLUE); break;
            case 35: textcolor(T_MAGENTA); break;
            case 36: textcolor(T_CYAN); break;
            case 37: textcolor(T_WHITE); break;
            case 40:
                textreverse(REVERSE);
                textcolor(T_BLACK);
                break;
            case 41:
                textreverse(REVERSE);
                textcolor(T_RED);
                break;
            case 42:
                textreverse(REVERSE);
                textcolor(T_GREEN);
                break;
            case 43:
                textreverse(REVERSE);
                textcolor(T_YELLOW);
                break;
            case 44:
                textreverse(REVERSE);
                textcolor(T_BLUE);
                break;
            case 45:
                textreverse(REVERSE);
                textcolor(T_MAGENTA);
                break;
            case 46:
                textreverse(REVERSE);
                textcolor(T_CYAN);
                break;
            case 47:
                textreverse(REVERSE);
                textcolor(T_WHITE);

```

```

        break;
    default:    break;
    }
}
if (DispCursorFlag==0)    textcursor(DISP_CURSOR);
else    textcursor(NODISP_CURSOR);
}

/*
-----
      アトリビュートバッファのクリア
-----
*/
void    clearAtrib()
{
    setmem(CrtAtrib,10,0);
    AtribNum=0;
}

/*
-----
      ファイル名バッファのクリア
-----
*/
void    clearMultiFile()
{
    setmem(MultiFile,(DATA_WD*10),0);
}

/*
-----
      接続用初期化
-----
*/
int initialize()
{
    if (sioinit()!=0){ /* check channel No. */
        errordisp("拡張ボードが装着されていません。");
        return FALSE;
    }
    sioconnect();
    if (ModemFlag!=0 && siosts()==0 && cp.ltype==0){ /* check CTS,RTS */
        errordisp("モデムの電源オフかモデムのトラブルです。");
        return FALSE;
    }
    KeyNoFlag=0;
    vt.KeyPad=0;
    vt.CursorKey=0;
    vt.Origin=0;
    vt.AutoLap=0;
    if (cp.code==0;:cp.code==4;:cp.code==5){
        vt.CODE=NULL;
        vt.SISO=0;
    }
    else if (cp.code==1){
        vt.CODE=NEWJIS;
        vt.SISO=FALSE;
    }
    else if (cp.code==2){
        vt.CODE=OLDJIS;
        vt.SISO=FALSE;
    }
    else if (cp.code==3){
        vt.CODE=NECJIS;
        vt.SISO=FALSE;
    }
    hist_yp=0;
    hist_xp=0;
    hist_line=0;
    hist_top=0;
    hist_end=0;
    hist_flag=0;
    clearCharBuff();
    clearMultiFile(); /* 転送ファイル領域のクリア */
}

```

```

clearAtrib();          /* CRTアトリビュート領域クリア */
return YES;
}

/*
.....
漢字入出力関係
.....
*/

int isiso_flag=FALSE;   /* シフトイン・フラグ */
int osiso_flag=FALSE;   /* 漢字JISフラグ */
int ikanji_flag=FALSE;  /* 漢字JISフラグ */
int okanji_flag=FALSE;  /* Xon/off フラグ */
int xonoff_flag=FALSE;

/*
-----
漢字を含めたデータ入力
-----
*/
int ksioinput()
{
    char    c;
    int x,y;
    unsigned int    JISKANJI; /* 漢字JISコード */
    unsigned int    SHIFTJIS; /* シフトJISコード */

    c=sioinput();
    /*..... コード・ダンプ処理 .....*/
    if((CodeFlag!=0 && c<' ')){
        textcolor(co.control);
        textreverse(co.R_control);
        cprintf("^%c", (c+'0'));
        funcoutput('^',(c+'0'));
        setatrib();
        return 0;
    }
    if(c==0)    return 0; /* NULL code */
    /*..... XON/XOFF処理 .....*/
    if(c==D3 && cp.xonoff==0){ /* あり */
        xonoff_flag=YES;
        return 0;
    }
    if(c==D1 && cp.xonoff==0){ /* なし */
        xonoff_flag=FALSE;
        return 0;
    }
    /*..... シフトイン・アウト・コード処理.....*/
    if(c==0x0e)
    {
        if((cp.code>0 && cp.code<4) :: cp.code==6)
        {
            isiso_flag=YES; /* S0 */
            return 0;
        }
    }
    if(c==0x0f)
    {
        if((cp.code>0 && cp.code<4) :: cp.code==6)
        {
            isiso_flag=FALSE; /* S1 */
            return 0;
        }
    }
    /*..... エスケープ・コード処理 .....*/
    if(c==ESC){
        clearEscBuff();
        EscBuff[0]=1;
        return 0;
    }
    if(c>=' ' && EscBuff[0]!=0){

```

```

EscBuff[EscBuff[0]]=c;
EscBuff[0]++;
if (EscBuff[0]>=20){
    clearEscBuff();
}
else if (EscBuff[0]==2 && check_vt1001(c)==YES){
    clearEscBuff();
}
else if (EscBuff[0]==3 && check_vt1002(c)==YES){
    clearEscBuff();
}
else if (EscBuff[1]=='['
    && (c=='0':::(c)=='A'&&c<='Z'):::(c)=='a'&&c<='z')){
    check_vt1003(c);
    clearEscBuff();
}
return 0;
}
/*..... キャラクター・コード処理 .....*/
if (c==' '){
    switch(cp.code)
    {
        case 0: /* シフト J I S */
            if (iskanji(c)!=0)
            {
                /* 1 バイト目 */
                SHIFTJIS=0xff00&((0x00ff&c)<<8);
                c=sioinput(); /* 2 バイト目 */
                SHIFTJIS=SHIFTJIS|(0x00ff&c);
            }
            else
                SHIFTJIS=0x00ff&c;
            return SHIFTJIS;
        case 1: /* 新 J I S */
            SHIFTJIS=checkjis(c);
            return SHIFTJIS;
        case 2: /* 旧 J I S */
            SHIFTJIS=checkjis(c);
            return SHIFTJIS;
        case 3: /* NEC 漢字 */
            SHIFTJIS=checkjis(c);
            return SHIFTJIS;
        case 4: /* DEC 漢字 */
            if ((0x80 & c)!=0){
                /* 1 バイト目 */
                JISKANJI=0x7f00&((0x00ff&c)<<8);
                c=sioinput(); /* 2 バイト目 */
                JISKANJI=JISKANJI|(0x007f&c);
                SHIFTJIS=jistojs(JISKANJI);
                return SHIFTJIS;
            }
            else
                return 0x00ff&c;
        case 5: /* J I S 8 */
            return 0x00ff&c;
        case 6: /* J I S 7 : ASCII */
            if (isiso_flag==YES)
            {
                c=(c&0x80);
                return 0x00ff&c;
            }
            else
                return (0x007f&c);
        default:
            return 0;
    }
}
else if (c<' ')
    return 0x00ff&c;
else
    return 0;
}

/*
-----
漢字 J I S の入力
-----
*/
int checkjis(char c)
{
    unsigned int JISKANJI;
    unsigned int SHIFTJIS;

```



```

int ch;

if(ikanji_flag==YES){
    JISKANJI=0xff00&((0x00ff&c)<<8);
    ch=sioinput(); /* 2 バイト目 */
    JISKANJI=JISKANJI|(0x00ff&c);
    SHIFTJIS=jisiojms(JISKANJI);
    return SHIFTJIS;
}
else{
    if(isiso_flag==YES) c=(c<>0x80);
    return 0x00ff&c;
}
}

/*
-----
漢字を含めたシリアルデータの出力
-----
*/
void ksiooutput(int data)
{
    int i,x,y;
    char c;
    unsigned int JISKANJI;
    char data1;
    char data2;

    while(xonoff_flag==YES){
        if(siorun()!=0){
            ksioinput();
            continue;
        }
        else{
            if((char)key_in()!=ESC) continue;
            else return;
        }
    }
    data1=(char)(0x00ff&(data>>8));
    data2=(char)(0x00ff&data);
    if(cp.method!=0){ /* 半二重処理 */
        textcolor(cp.half);
        if(data1!=0) putchar(data1);
        if(data2!=LF){
            putchar(data2);
            if(data2==CR) putchar(LF);
        }
        setatrib();
    }
    if(data2==CR){
        if(cp.sndcr==0) siooutput(CR);
        else{
            siooutput(CR);
            siooutput(LF);
        }
        return;
    }
    else if(data2==LF){
        if(cp.sndlf==0) siooutput(LF);
        return;
    }
    else if(data1==0 && data2!=0){
        switch(cp.code)
        {
            case 0: /* シフト J I S */
                siooutput(data2);
                return;
            case 1: /* NEC 漢字 */
                if(okanji_flag==YES){
                    okanji_flag=FALSE;
                    siooutput(ESC);
                    siooutput('H');
                }
                check_siso(data2);
        }
    }
}

```

```

        if(osiso_flag==YES) data2=(0x7f&data2);
        siooutput(data2);
        return;
    case 2: /* 新 J I S */
        if(okanji_flag==YES){
            okanji_flag=FALSE;
            siooutput(ESC);
            siooutput('(');
            siooutput('J');
        }
        check_siso(data2);
        if(osiso_flag==YES) data2=(0x7f&data2);
        siooutput(data2);
        return;
    case 3: /* 旧 J I S */
        if(okanji_flag==YES){
            okanji_flag=FALSE;
            siooutput(ESC);
            siooutput('(');
            siooutput('H');
        }
        check_siso(data2);
        if(osiso_flag==YES) data2=(0x7f&data2);
        siooutput(data2);
        return;
    case 4: /* D E C 漢字 */
        data2=(0x7f&data2);
        siooutput(data2);
        return;
    case 5: /* J I S 8 */
        siooutput(data2);
        return;
    case 6: /* J I S 7 */
        siooutput(0x7f&data2);
        return;
    default: break;
}
)
else if(data1!=0 && data2!=0){ /* 漢字 */
    if((cp.code)=1 && cp.code<=3){cp.code==6){
        if(osiso_flag==YES){
            osiso_flag=FALSE;
            siooutput(0x0f);
        }
    }
    switch(cp.code)
    {
    case 0: /* シフト J I S */
        siooutput(data1);
        siooutput(data2);
        return;
    case 1: /* 新 J I S */
        if(okanji_flag==FALSE){
            okanji_flag=YES;
            siooutput(ESC);
            siooutput('$');
            siooutput('B');
        }
        JISKANJ1=jmstojis(data);
        siooutput((char)((0xff00&JISKANJ1)>>8));
        siooutput((char)(0x00ff&JISKANJ1));
        return;
    case 2: /* 旧 J I S */
        if(okanji_flag==FALSE){
            okanji_flag=YES;
            siooutput(ESC);
            siooutput('$');
            siooutput('D');
        }
        JISKANJ1=jmstojis(data);
        siooutput((char)((0xff00&JISKANJ1)>>8));
        siooutput((char)(0x00ff&JISKANJ1));
        return;
    case 3: /* N E C 漢字 */

```

```

        if(okanji_flag==FALSE){
            okanji_flag=YES;
            siooutput(ESC);
            siooutput('K');
        }
        JISKANJl=jmstojis(data);
        siooutput((char)((0xff00&JISKANJl)>>8));
        siooutput((char)(0x00ff&JISKANJl));
        return;
    case 4: /* DEC 漢字 */
        JISKANJl=jmstojis(data);
        data1=(char)(0x007f&_rotr(JISKANJl,8));
        data2=(char)(0x007f&JISKANJl);
        siooutput(0x80 | data1);
        siooutput(0x80 | data2);
        break;
    default: break;
}
}
)

```

```

/*
-----
XON/XOFF 出力
-----
*/
void xon_out()
{
    if(cp.xonoff==0) siooutput(D3);
}

```

```

void xoff_out()
{
    if(cp.xonoff==0) siooutput(D1);
}

```

```

/*
-----
SI/SO のチェック
-----
*/
void check_siso(char data)
{
    if((cp.code>1 && cp.code<=3) :: cp.code==6){
        if(data>=0xa0 && data<=0xe0){ /* カタ */
            if(osiso_flag==0){
                osiso_flag=YES;
                siooutput(0x0e);
            }
            else{
                if(osiso_flag==YES){
                    osiso_flag=FALSE;
                    siooutput(0x0f);
                }
            }
        }
    }
}
)

```

```

/*
-----
SIO へのライン出力
-----
*/
void ksioprint(char *buff)
{
    char c;
    int i=0;

    while(1){
        c=*(buff+i);
        if(c==0) return;
        ksiooutput(c);
        i++;
    }
}

```

```

    )
}

/*
-----
          タイムアウト付き無手順データ受信
-----
*/
int toutksioinput(int t)
{
    int i;
    unsigned int    data;

    for(i=0;i<t;i++){
        data=ksioinput();
        if(data!=0) return data;
    }
    return 0;
}

/*
-----
          各種機能処理
-----
*/
void    funcoutput(char c1,char c2)
{
    int x,y;

    if(ConnectFlag==0) return;
    if(PrintFlag!=0)
    {
        if(c1!=0) printer(c1);
        printer(c2);
    }
    if(LogFlag!=0)
    {
        if(c1!=0) fputc(0x00ff&c1,Lp);
        fputc(0x00ff&c2,Lp);
        if(ferror(Lp)!=0)
        {
            x=wherex();
            y=wherey();
            errordisp("ディスク書き込みエラー");
            window(1,1,80,24);
            setatrib();
            gotoxy(x,y);
        }
    }
}

/*
-----
          エスケープ・データバッファのクリアー
-----
*/
void    clearEscBuff()
{
    setmem(EscBuff,20,0);
}

/*
-----
          最下位行への受信量の表示
-----
*/
void    linebufdisp()
{
    int x,y,i;
    long int    p,num;
    char    *b,ch;
    char    da[4];

    y=25;    x=17;

```

```

b=MK_FP(0xa000,BSPOS-1);
num=(0x0000ffff&hist_yp);
p=(num*100)/720;
sprintf(da,"%02d", (short)p);
for(i=0;i<3;i++){
    ch=*(da+i);
    if(ch==0) break;
    *(b+2+i)=ch;
}
}

/*
-----
            エスケープコード出力
-----
*/
void      escsioout(char *buff)
{
    int i;
    char  c;

    siooutput(ESC);
    for(i=0;i<9;i++){
        c=*(buff+i);
        if(c==0) break;
        ksiooutput(0x00ff&c);
    }
}

/*
.....
            V T 1 0 0 用 ハ ン ド ラ
            .....
*/

int XPOS;          /* 現在のコンソールのX位置 */
int YPOS;          /* 現在のコンソールのY位置 */

int SaveNum;
char  SaveAtrib[20];

extern int ikanji_flag; /* HANDLER.C */

/*
-----
            エスケープ・コードのチェック
-----
*/
int check_vt1001(char c)
{
    int x,y,i,com;

    switch(c)
    {
        case 'D':    putch(LF); return YES;
        case 'E':    putch(CR); putch(LF); return YES;
        case 'M':
            x=wherex();
            y=wherey();
            if(ScrollTp==2 && ScrollBm==24){
                if(y==1) insline();
                else     cursorup(1);
            }
            else{
                if(ScrollTp==y){
                    if(ScrollBm!=24){
                        gotoxy(1,ScrollBm);
                        delline();
                        gotoxy(x,y);
                    }
                    insline();
                }
            }
    }
}

```

```

        else    cursorup(1);
    }
    return YES;
case '7':
    XPOS=wherex();
    YPOS=wherey();
    for(i=0;i<10;i++)    SaveAtrib[i]=CrtAtrib[i];
    SaveNum=AtribNum;
    return YES;
case '8':
    gotoxy(XPOS,YPOS);
    for(i=0;i<10;i++)    CrtAtrib[i]=SaveAtrib[i];
    AtribNum=SaveNum;
    setatrib();
    return YES;
case '~':
    window(1,1,80,24);
    clrscr();
    return YES;
case 'K':
    if(cp.code==3)    /* NEC 漢字 */
    {
        ikanji_flag=YES;
        return YES;
    }
    else    return FALSE;
case 'H':
    if(cp.code==3)
    {
        ikanji_flag=FALSE;
        return YES;
    }
    else    return FALSE;
case '=':    vt.KeyPad=-1;    return YES;
case '>':    vt.KeyPad=0;    return YES;
default:    return FALSE;
}
}

```

/*

----- エスケープ・コードのチェック -----

*/

```

int check_vt1002(char c)
{
    switch(EscBuff[1])
    {
        case '$':
            if((cp.code==1 && c=='B')
                || (cp.code==2 && c=='@'))
            {
                ikanji_flag=YES;    return YES;
            }
            else    return FALSE;
        case 'J':
            if((cp.code==1 && c=='J')
                || (cp.code==2 && c=='H'))
            {
                ikanji_flag=FALSE;    return YES;
            }
            else    return FALSE;
        case 'H':    return YES;
        default:    return FALSE;
    }
}

```

/*

----- 'I' エスケープコードのチェック -----

*/

```

void    check_vt1003(char c)
{

```

```

char    crt[320];
int x,y,i;
char    ch;
int collow[2];
char    outbuf[9],linebuf[320];

switch(c)
{
    case    '@':
        x=wherex();
        y=wherey();
        getnum(collow,1);
        if(collow[0]==0)    collow[0]=1;
        movetext(x,y,(80-collow[0]),y,(x+collow[0]),y);
        gotoxy(x,y);
        for(i=0;i<collow[0];i++)    putchar(' ');
        gotoxy(x,y);
        return;
    case    'A':
        getnum(collow,1);
        if(collow[0]==0)    collow[0]=1;
        cursorup(collow[0]);
        return;
    case    'B':
        getnum(collow,1);
        if(collow[0]==0)    collow[0]=1;
        cursordown(collow[0]);
        return;
    case    'C':
        getnum(collow,1);
        if(collow[0]==0)    collow[0]=1;
        cursorright(collow[0]);
        return;
    case    'D':
        getnum(collow,1);
        if(collow[0]==0)    collow[0]=1;
        cursorleft(collow[0]);
        return;
    case    'H':
        getnum(collow,2);
        if(collow[0]==0)    collow[0]=1;
        if(collow[1]==0)    collow[1]=1;
        if(vt.Origin!=0)    y=collow[0]+ScrollTp-1;
        else    y=collow[0];
        x=collow[1];
        gotoxy(x,y);
        return;
    case    'J':
        getnum(collow,1);
        switch(collow[0])
        {
            case    0:
                x=wherex();
                y=wherey();
                window(1,y+1,80,24);
                clrscr();
                window(1,1,80,24);
                gotoxy(x,y);
                clrscr();
                return;
            case    1:
                x=wherex();
                y=wherey();
                window(1,1,80,y-1);
                clrscr();
                window(1,y,x,y);
                clrscr();
                window(1,1,80,24);
                gotoxy(x,y);
                return;
            case    2:
                window(1,1,80,24);
                clrscr();
                return;
        }
}

```

```

        default:    return;
    }
    case 'K':
        getnum(collow,1);
        switch(collow[0])
        {
            case 0:  clreol();    return;
            case 1:
                x=wherex();
                y=wherey();
                window(1,y,x,y);
                clrscr();
                window(1,1,80,24);
                gotoxy(x,y);
                return;
            case 2:
                x=wherex();
                y=wherey();
                window(1,y,80,y);
                clrscr();
                window(1,1,80,24);
                gotoxy(x,y);
                return;
        }
        default:    return;
    }
    case 'L':
        getnum(collow,1);
        if(collow[0]==0)    collow[0]=1;
        for(i=0;i<collow[0];i++)    insline();
        return;
    case 'M':
        getnum(collow,1);
        if(collow[0]==0)    collow[0]=1;
        for(i=0;i<collow[0];i++)    delline();
        return;
    case 'P':
        x=wherex();
        y=wherey();
        getnum(collow,1);
        if(collow[0]==0)    collow[0]=1;
        movetext((x+collow[0]),y,80,y,x,y);
        gotoxy((80-collow[0]+1),y);
        clreol();
        gotoxy(x,y);
        return;
    case 'f':
        getnum(collow,2);
        if(collow[0]==0)    collow[0]=1;
        if(collow[1]==1)    collow[1]=1;
        if(vt.Origin!=0)    y=collow[0]+ScrollTp-1;
        else    y=collow[0];
        x=collow[1];
        gotoxy(x,y);
        return;
    case 'g':    return;
    case 'h':    set_mode();    return;
    case 'l':    reset_mode();    return;
    case 'm':
        AtribNum=getnum(CrtAtrib,10);
        setatrib();
        return;
    case 'n':
        if(EscBuff[2]=='G'){
            x=wherex();
            y=wherey()-ScrollTp+1;
            setmen(outbuf,9,0);
            sprintf(outbuf,"%02d;%02dR",y,x);
            escioout(outbuf);
            return;
        }
        else    return;
    case 'r':
        getnum(collow,2);
        if(collow[0]==0)    collow[0]=1;

```



```

        if (collow[1] == 0)    collow[1] = 1;
        ScrollTp = collow[0];
        ScrollBm = collow[1];
        return;
    case 's':
        XPOS = wherex();
        YPOS = wherey();
        for (i = 0; i < 10; i++)    SaveAtrib[i] = CrtAtrib[i];
        SaveNum = AtribNum;
        return;
    case 'u':
        gotoxy(XPOS, YPOS);
        for (i = 0; i < 10; i++)    CrtAtrib[i] = SaveAtrib[i];
        AtribNum = SaveNum;
        setatrib();
        return;
    default:    return;
}

/*
-----
check '?' or '>' mode
-----
*/
void    set_mode()
{
    char    *mode[7] = {">1", "?1", "?5", "?6", "?7", "?25", "?47"};
    int x, y, i;

    for (i = 0; i < 7; i++) {
        if (strstr(&EscBuff[2], mode[i]) != NULL)    break;
    }
    switch(i)
    {
    case 0:
        x = wherex();
        y = wherey();
        window(1, 1, 80, 25);
        gotoxy(x, y);
        break;
    case 1:    vt.CursorKey = -1;                break;
    case 2:    textreverse(REVERSE);            break;
    case 3:    vt.Origin = -1;                    break;
    case 4:    vt.AutoLap = -1;                    break;
    case 5:
        textcursor(DISP_CURSOR);
        DispCursorFlag = 0;
        break;
    case 6:
        if (CrtSaveFlag == 0) {
            gettext(1, 1, 80, 24, CrtText);
            CrtSaveFlag = -1;
        }
        break;
    default:    break;
    }
}

void    reset_mode()
{
    char    *mode[7] = {">1", "?1", "?5", "?6", "?7", "?25", "?47"};
    int x, y, i;

    for (i = 0; i < 7; i++) {
        if (strstr(&EscBuff[2], mode[i]) != NULL)    break;
    }
    switch(i)
    {
    case 0:
        x = wherex();
        y = wherey();
        window(1, 1, 80, 25);
        gotoxy(x, y);

```

```

        break;
case 1:  vt.CursorKey=0;          break;
case 2:  textreverse(NOREVERSE); break;
case 3:  vt.Origin=0;            break;
case 4:  vt.AutoLap=0;           break;
case 5:
    textcursor(NODISP_CURSOR);
    DispCursorFlag=-1;
    break;
case 6:
    if (CrtSaveFlag!=0) {
        puttext(1,1,80,24,CrtText);
        CrtSaveFlag=0;
    }
    break;
default: break;
}
}

```

```

/*

```

カーソルアップ

```

*/
void cursorup(int line)
{
    int x,y;

    x=wherex();
    y=wherey();
    y-=line;
    if (y<1) y=1;
    gotoxy(x,y);
}

```

```

/*

```

カーソルダウン

```

*/
void cursordown(int line)
{
    int x,y;

    x=wherex();
    y=wherey();
    y+=line;
    if (y>24) y=24;
    gotoxy(x,y);
}

```

```

/*

```

カーソルライト

```

*/
void cursorright(int col)
{
    int x,y;

    x=wherex();
    y=wherey();
    x+=col;
    if (x>80) x=80;
    gotoxy(x,y);
}

```

```

/*

```

カーソルレフト

```

*/
void cursorleft(int col)
{

```

```

int x,y;

x=wherex();
y=wherey();
x-=col;
if(x<1)      x=1;
gotoxy(x,y);
)
}

/*
-----
      エスケープの数字データ
-----
*/
int getnum(int buff[],int num)
{
    char    curdat[3];
    char    ch;
    int i,q,j;

    for(i=0;i<num;i++) buff[i]=0;
    q=0;      i=0;      j=0;
    setmem(curdat,3,0);
    while(1){
        ch=EscBuff[2+i];
        if(ch>='0' && ch<='9'){
            curdat[j]=ch;
            i++;
            j++;
        }
        else if(ch==':'||ch=='0'||(ch>='A'&&ch<='Z')||
                (ch>='a'&&ch<='z')){
            buff[q]=atoi(curdat);
            setmem(curdat,3,0);
            q++;
            i++;
            j=0;
            if(q>num || ch=='0' ||
                (ch>='A'&&ch<='Z') ||
                (ch>='a'&&ch<='z')){
                break;
            }
        }
    }
    return q;
}

```

付録シ．ヒストリ機能・ソースファイル(hist.c)

```

/*
.....
.      パソコン通信ソフト「飛火野」
.
.      [ 履歴モード ]
.....
*/
#include      <stdio.h>
#include      <conio.h>
#include      <string.h>
#include      <jstring.h>
#include      <jctype.h>
#include      <mem.h>
#include      <dos.h>
#include      <process.h>
#include      <io.h>
#include      <stdlib.h>
#include      <alloc.h>
#include      "nfire.h"
#include      "fall.h"

int history();
int hist_func(int comd); /* ログヒストリ・各種処理 */
int hist_copy();        /* メニューによる各種処理 */
int hist_print();       /* ヒストリバッファのコピー */
int hist_file();        /* ヒストリーのプリント */
int hist_del();         /* ヒストリーのファイル書き込み */
int hist_serch();       /* ヒストリー行削除 */
int hist_topgo();       /* 検索 */
int hist_endgo();       /* 検索先頭 */
int hist_keyin();       /* 行先頭 */
void hist_up();         /* 行末尾 */
void hist_dn();         /* キー入力 */
void hist_rup();        /* UP */
void hist_rdn();        /* DOWN */
void hist_cr();         /* ROLL UP */
void hist_esc();        /* ROLL DOWN */
void hist_chardisp();   /* CR */
void histprint(int line,int low); /* ESC */
int setCharBuff();      /* ヒストリーバッファの表示 */
void clearCharBuff();   /* ヒストリのライン表示 */
char getCharBuff(int line,int col); /* 履歴バッファの取得 */
void putCharBuff(int line,int col,char c); /* Clear History character buffer */
void chgCharBuff();     /* 履歴バッファデータ取得 */
int chkallline(void);   /* 履歴バッファへ書き込み */
void freeCharBuff();    /* バッファの入れ替え */
void writeCharBuff(char c1,char c2); /* 全行処理の確認 */

/* ヒストリーバッファ位置 */

int Histy=0; /* ヒストリのカーソルライン */
int lin1,lin2;

/* ヒストリ内容書き込みファイル名 */

char hist_FileName[80];

/* 検索文字バッファ */

char srcbuff[60]; /* 検索文字のバッファ */

/* ログヒストリー用バッファ */

char *CharBuff; /* キャラクタ・バッファ・ポインタ */

/*
.....
.      履歴モードのメイン
.
.....
*/
int history()

```

```

{
static char histmenu[8][DATA_WD]={
    "便の複写","ファイル","プリント","文字検索",
    "履歴先頭","履歴末尾","範囲削除","メインメニュー"};
static WINDOW wd={ 20,3,30,10,T_CYAN,T_YELLOW,T_WHITE,0,"履歴モード"
    ,2,8,0,0,1,1,0 };
int s,wflg=WD_OPEN;
hist_flag=hist_top=hist_end=0;
setmem(srcbuff,60,0);
hist_line=hist_yp-(connect_y-1);
if(hist_line<0) hist_line=0;
Histy=connect_y-1;
window(1,1,80,24);
textcolor(co.history);
textcursor(NODISP_CURSOR);
hist_chardisp();
while( 1 )
{
    s=twindow(1,wflg,&wd,histmenu );
    wflg=WD_REMAIN;
    if ( hist_func( s )!= YES ) continue;
    switch(s){
        case 0: hist_copy(); break;
        case 1: hist_file(); break;
        case 2: hist_print(); break;
        case 3: if ( hist_serch()!=YES ) continue;
                break;
        case 4: hist_topgo(); break;
        case 5: hist_endgo(); break;
        case 6: hist_del(); break;
        case 7: allclose();
                return OPEN; /* メニューへ復帰 */
        default : break;
    }
    allclose();
    hist_keyin();
    wflg=WD_OPEN;
}
}
/*
-----
                        ヒストリーのキーイン
-----
*/
int hist_keyin()
{
int c;
mouseon();
MS_LB=MS_RB=0;
while( 1 )
{
    while( ( c=key_in() )==0 )
    {
        if ( MS_LB ){ /* 機能メニューに復帰 */
            mouseoff();
            return;
        }
        mouseput();
    }
    switch( c & 0x00ff )
    {
        case CR : hist_cr();break;
        case ESC : hist_esc();break;
        default : break;
    }
    switch( (c>>8) & 0x00ff )
    {
        case UP: hist_up(); break;
        case DOWN: hist_dn(); break;
        case RUP: hist_rup(); break;
        case RDOWN: hist_rdn(); break;
        case FUN10: MS_LB=1; break; /* 各機能メニューへ復帰 */
        default: break;
    }
}
}

```

```

    )
}
/*
-----
メニューによる各種処理
-----
*/
int hist_func(int comd)
{
    int x,y;

    if((comd>=0 && comd<=2) || comd==6){
        if(hist_flag==0){
            if(chkallline()==YES){
                lin1=0; lin2=hist_yp;
            }
            else return FALSE;
        }
        else{
            lin1=hist_top;
            lin2=hist_end;
        }
        if(comd==1){ /* ファイル */
            setmem(hist_FileName,80,0);
            getfilnam(RET_ON,hist_FileName);
            if(*hist_FileName!=0) return YES;
            else return FALSE;
        }
        else return YES;
    }
    else return YES;
}
/*
-----
バッファの先頭へ
-----
*/
int hist_topgo()
{
    allclose();
    hist_line=0;
    Histy=0;
    hist_chardisp();
    return;
}
/*
-----
バッファの末尾へ
-----
*/
int hist_endgo()
{
    allclose();
    hist_line=hist_yp-(connect_y-1);
    Histy=connect_y-1;
    hist_chardisp();
    return;
}
/*
-----
ヒストリーのカーソルアップ
-----
*/
#define HISTL hist_line+Histy

void hist_up()
{
    if(hist_line==0 && Histy==0){
        putch(8ELL);
        return;
    }
    if(hist_flag==0){
        textcolor(co.history);
    }
}

```

```

        textreverse(NOREVERSE);
    }
    else if(hist_flag==1){
        if(HISTL==hist_top){
            textcolor(co.kakuninn);
            textreverse(REVERSE);
        }
        else{
            textcolor(co.history);
            textreverse(NOREVERSE);
        }
    }
    else if(hist_flag==2){
        if(HISTL>=hist_top && HISTL<=hist_end){
            textcolor(co.kakuninn);
            textreverse(REVERSE);
        }
        else{
            textcolor(co.history);
            textreverse(NOREVERSE);
        }
    }
    histprint(hist_line,Histy);
    Histy--;
    if(Histy<0){
        Histy=0;
        hist_line--;
        if(hist_line<0) hist_line=0;
        gotoxy(1,1);
        insline();
    }
    textcolor(co.history);
    textreverse(REVERSE);
    histprint(hist_line,Histy);
}

/*
-----
      ヒストリーのカーソルダウン
-----
*/
void hist_dn()
{
    if((hist_line+Histy)>=hist_yp){
        putch(BELL);
        return;
    }
    if(hist_flag==0){
        textcolor(co.history);
        textreverse(NOREVERSE);
    }
    else if(hist_flag==1){
        if(HISTL==hist_top){
            textcolor(co.kakuninn);
            textreverse(REVERSE);
        }
        else{
            textcolor(co.history);
            textreverse(NOREVERSE);
        }
    }
    else if(hist_flag==2){
        if(HISTL>=hist_top && HISTL<=hist_end){
            textcolor(co.kakuninn);
            textreverse(REVERSE);
        }
        else{
            textcolor(co.history);
            textreverse(NOREVERSE);
        }
    }
    histprint(hist_line,Histy);
    Histy++;
    if(Histy>23){

```

```

        Histy=23;
        hist_line++;
        if((hist_line+24)>HISTLIN) hist_line=HISTLIN-24;
        gotoxy(1,1);
        delline();
        gotoxy(1,24);
    }
    textcolor(co.history);
    textreverse(REVERSE);
    histprint(hist_line,Histy);
}

/*
-----
      ヒストリーのROLL・UP
-----
*/
void hist_rup()
{
    hist_line+=(Histy+24);
    if(hist_line>hist_yp){
        hist_line=(hist_yp-23);
        putch(BELL);
    }
    hist_chardisp();
}

/*
-----
      ヒストリーのROLL・DOWN
-----
*/
void hist_rdn()
{
    hist_line-=(Histy+24);
    if(hist_line<0){
        hist_line=0;
        putch(BELL);
    }
    hist_chardisp();
}

/*
-----
      ヒストリーのCR
-----
*/
void hist_cr()
{
    if(hist_flag==0){
        hist_top=hist_line+Histy;
        hist_flag=1;
    }
    else if(hist_flag==1){
        hist_end=hist_line+Histy;
        hist_flag=2;
    }
    if(hist_flag==1 || hist_flag==2){
        textcolor(co.kakuninn);
        textreverse(REVERSE);
        histprint(hist_line,Histy);
    }
}

/*
-----
      ヒストリーのESC
-----
*/
void hist_esc()
{
    if(hist_flag!=0) hist_flag=0;
    hist_top=0;
    hist_end=0;
}

```



```

        hist_chardisp();
    }

    /*
    -----
    ヒストリーのバッファ表示
    -----
    */
    void hist_chardisp()
    {
        int i, p;

        window(1,1,80,25);
        textcolor(co.history);
        textcursor(NODISP_CURSOR);
        for(i=0; i<24; i++){
            p=hist_line+i;
            if(p>hist_yp) break;
            if(i==Histy){
                textreverse(REVERSE);
                textcolor(co.history);
            }
            else if(hist_flag!=0 && HISTL>=hist_top && HISTL<=hist_end){
                textreverse(REVERSE);
                textcolor(co.kakuninn);
            }
            else{
                textreverse(NOREVERSE);
                textcolor(co.history);
            }
            histprint(hist_line, i);
        }
        if(i<24){
            window(1,(i+1),80,24); clrscr();
        }
        window(1,1,80,24);
    }

    /*
    -----
    ヒストリのライン表示
    -----
    */
    void histprint(int line, int low)
    {
        if(low>=23) window(1,1,80,25);
        gotoxy(1, low+1);
        cprintf("%-80s", (CharBuff+(83-(line+low))));
        window(1,1,80,24);
    }

    /*
    -----
    ヒストリのエディットバッファへコピー
    -----
    */

    extern int endpos; /* ED.C */
    extern char editbuf[BUFNUM]; /* edit buffers */

    int hist_copy()
    {
        int q, i, j;
        char ch;

        edinit(); /* the initialization of new version */
        edbufnew();
        q=0;
        for(i=1; i<=lin2; i++){
            for(j=0; j<80; j++){
                ch=getCharBuff(i, j);
                if(ch<' ') break;
                else editbuf[q]=ch;
                q++;
            }
        }
    }

```

```

        if(q)=(BUFNUM-2))    break;
    )
    editbuf[q++]=CR;
    if(q)=(BUFNUM-1))    break;
)
endpos=q;
EdCopyFlag=FALSE;
EdItFlag=FALSE;
hist_flag=0;
hist_top=0;
hist_end=0;
hist_chardisp();
textreverse(NOREVERSE);
textcolor(co.history);
return;
)

/*
-----
                ヒストリのプリント
-----
*/
int hist_print()
{
    static WINDOW wd=( 2,3,34,3,T_CYAN,T_YELLOW,T_WHITE,0,"",1,1,0,2,1,0,0 );
    static char data[11][DATA_WD]=( "プリント出力中。中止は [ESC]" );

    int i,j,s,c;
    char    ch;

    twindow(1, WD_OPEN,&wd,data );
    for(i=lin1;i<=lin2;i++){
        for(j=0;j<80;j++){
            ch=getCharBuff(i,j);
            if(ch==0 || ch==CR) break;
            if( (key_in()==0x00ff&ESC) ){
                hist_flag=0;
                twindow( WD_CLOSE );
                return;
            }
            printer(ch);
        }
        printer(CR);
        printer(LF);
    }
    twindow( WD_CLOSE );
    hist_flag=0;
    hist_top=0;
    hist_end=0;
    hist_chardisp();
    return;
}

/*
-----
                ヒストリーメモリエリアの削除
-----
*/
int hist_del()
{
    int n1,n2,i;

    allclose();
    n1=lin2-lin1+1;
    n2=(hist_yp-lin2)-83;
    memmove((CharBuff+(83-lin1)),(CharBuff+(83-(lin2+1))),n2);
    hist_yp-=n1;
    if(hist_yp<0)    hist_yp=0;
    setmem((CharBuff+83-(hist_yp+1)),(n1*83),0);
    hist_top=0;
    hist_end=0;
    hist_flag=0;
    if((hist_line+23)>=lin1){

```

```

        Histy=(lin1-hist_line);
        hist_chardisp();
    }
    else if (lin1<hist_line){
        hist_line=(lin1-10);
        if (hist_line<0) hist_line=0;
        Histy=(lin1-hist_line);
        hist_chardisp();
    }
    linebufdisp();
    window(1,1,80,24);
    textcolor(co.history);
    textcursor(NODISP_CURSOR);
    return;
}

```

```

/*

```

----- ヒストリバッファをファイルに出力 -----

```

*/
int hist_file()
{
    char    ch;
    FILE    *p;
    int i,j,num;

    p=fopen(hist_FileName,"wb");
    if (p==NULL){
        errordisp("ファイルがオープンできません。");
        window(1,1,80,24);
        return;
    }
    for (i=lin1;i<=lin2;i++){
        for (j=0;j<80;j++){
            ch=getCharBuff(i,j);
            if (ch==0) break;
            else fputc(0x00ff&ch,p);
        }
        fputc(0x00ff&CR,p);
        fputc(0x00ff&LF,p);
    }
    fclose(p);
    hist_flag=0;
    hist_top=0;
    hist_end=0;
    hist_chardisp();
    return;
}

```

```

/*

```

----- 文字での検索 -----

```

*/
int hist_serch()
{
    static WINDOW wd={ 2,3,40,3,T_GREEN,T_YELLOW,T_WHITE,0,
        "検索文字列の入力",0,1,0,0,1,0,0 };
    static WINDOW wd2={ 2,22,32,3,T_BLUE,T_CYAN,T_WHITE,0,"",1,1,0,2,1,0,0 };
    static char data[1][DATA_WD];
    static char data2[1][DATA_WD]={ "再検索は(ス^ー)ス, 中止は(ESC)" };

    int s,i,y;
    char    c;

    s=twindow(1, WD_OPEN,&wd,data );
    twindow( WD_CLOSE );
    if ( s == -1 || data[0][0]==0) return FALSE; /* 検索文字の入力なし */
    strcpy( srcbuff,data[0] );
    allclose();
    while(1){
        for (i=(hist_line+Histy+1);i<=hist_yp;i++){
            if (jstrchr((CharBuff+(83-i)),srcbuff)!=NULL)

```

```

        break;
    }
    if (i > hist_yp) {
        errordisp(" 検索文字が見つかりません。");
        return YES;
    }
    if ((hist_line+20) < i) {
        hist_line = (i-10);
        Histy = 10;
    }
    else Histy = (i-hist_line);
    hist_chardisp();
    twindow(1, WD_OPEN, &wd2, data2); /* 検索を続けるか */
    while(1) {
        if ((c = (char) (0x00ff & key_in())) == 0) continue;
        if (c == ' ' || c == ESC) break;
    }
    twindow(WD_CLOSE);
}
return YES;
}
/*
-----
      □ ギングバッファの取得
-----
*/
int setCharBuff()
{
    if ((CharBuff = calloc(HISTLIN, 83)) == NULL) return FALSE;
    else return YES;
}
/*
-----
      □ ギングバッファエリアのクリア
-----
*/
void clearCharBuff()
{
    setmem(CharBuff, (HISTLIN*83), 0);
}
/*
-----
      □ ギングのデータ取得
-----
*/
char getCharBuff(int line, int col)
{
    char ch;

    ch = *(CharBuff + (83*line) + col);
    return ch;
}
/*
-----
      □ ギングのデータ書き込み
-----
*/
void putCharBuff(int line, int col, char c)
{
    *(CharBuff + (83*line) + col) = c;
}
/*
-----
      全行入れ替え
-----
*/
void chgCharBuff()
{
    int i, h;

```

```

h=(HISTLIN/2);
setmem(CharBuff,(h*83),0);
/* バッファの半分をクリア */
movmem((CharBuff+(83*h)),CharBuff,(h*83));
setmem((CharBuff+(83*h)),(h*83),0);
hist_yp=h;
)

/*
-----
全行処理の確認
-----
*/
int chkallline( void )
(
static WINDOW wd={ 2,3,30,5,T_CYAN,T_WHITE,T_WHITE,1,"確認モード"
,2,3,1,0,1,0,0 };
static char data[3][DATA_WD]={ "全行処理します。よろしい?",
" はい ",
" いいえ " };

int s;
s=twindow(1, WD_OPEN,&wd,data );
twindow( WD_CLOSE );
if ( s==1 ) return YES;
return FALSE;
)

/*
-----
バッファの解放
-----
*/
void freeCharBuff()
(
free(CharBuff);
)

/*
-----
バッファ書き込み処理
-----
*/
void writeCharBuff(char c1,char c2)
(
int i,k;

if(c1==0){
if(c2==' '){
if(hist_xp>79){
hist_yp++;
hist_xp=0;
if(hist_yp==HISTLIN) chgCharBuff();
}
putCharBuff(hist_yp,hist_xp,c2);
hist_xp++;
}
else if(c2==LF){
hist_yp++;
if(hist_yp==HISTLIN) chgCharBuff();
}
else if(c2==CR) hist_xp=0;
else if(c2==TAB){
k=8-(hist_xp-(hist_xp/8)*8);
for(i=0;i<k;i++){
putCharBuff(hist_yp,hist_xp,' ');
if(hist_xp==79){
hist_yp++;
hist_xp=0;
setmem((CharBuff+(83*hist_yp)),83,0);
}
else hist_xp++;
}
}
}
)
)

```

```
    else{
        if(hist_xp>78){
            hist_yp++;
            hist_xp=0;
            if(hist_yp==HISTLIN)    chgCharBuff();
        }
        putCharBuff(hist_yp,hist_xp,c1);
        hist_xp++;
        putCharBuff(hist_yp,hist_xp,c2);
        hist_xp++;
    }
}
```

付録M. ファイル転送処理用ハンドラ・ソースファイル (ftrans.c)

```

/*
.....
通信ソフト「飛火野」
.....
【ファイル転送】
.....
*/
#include <stdio.h>
#include <io.h>
#include <dos.h>
#include <dir.h>
#include <conio.h>
#include <process.h>
#include <string.h>
#include <stdio.h>
#include <ctype.h>
#include <alloc.h>
#include <sys%stat.h>
#include <jctype.h>
#include "nfire.h"
#include "fall.h"

#define ERR -1
#define ER1 -2
#define ER2 -3
#define ER3 -4
#define FILEERR ERR /* -1 */
#define TIMEOUT ER1 /* -2 */
#define FEND ER2 /* -3 */
#define PEND ER3 /* -4 */
#define TRUE 1 /* Boolean TRUE */
#define FALSE 0 /* Boolean FALSE */
#define BOOL int /* Boolean type */

FILE *fp; /* ファイル・ポインタ */
int fh; /* ファイルハンドル */

extern int Protoc; /* プロトコル */
extern int Direct; /* 転送方向 */
extern int LineFeed; /* 転送しF処理 */
extern int LineNo; /* 転送し行字数 */
extern int XfWait; /* 転送待時間 */

int rdxfile(char *filnam);
int wrxfile(char *filnam);
void getxfilnam(char *file, int line);
int gnxtfil(int *line);
int getxdir(char *pathnam);

char b7sioinput(int seconds);
int receive(int seconds);

void open_errmess();
void upend_mess();
void dnend_mess();
void xfil_errmess();
void dir_errmess();
void time_errmess();

void ftran_waku(char *direct, char *protoc);
void disp_XFILE(char *file);
void disp_FLENGTH(long int num);
void disp_PLENGTH(long int num);
void disp_PBLOCK(int num);
void disp_RETRY(int num);
void disp_STATUS(char *status);

int long XBYTE; /* 転送バイト数 */
int FDSK;
char XDRIIVE[3];
char DIR[50];

```

```

char    NAME[13];
char    EXT[5];
char    fldata[8][DATA_WD]={ "フ ァ イ ル 名 : ",
                              " 転 送 方 向 : ",
                              " 使 用 プ ロ ト コ ル : ",
                              " フ ァ イ ル サ イ ズ : ",
                              " 処 理 バ イ ト 数 : ",
                              " 処 理 パ ケ ッ ト 数 : ",
                              " リ ト ラ イ 回 数 : ",
                              " ス テ イ タ ス : " };

int i;
static WINDOW ftranwp={ 6,8,66,10,T_RED,T_CYAN,T_WHITE,0
                        ,"ファイル転送・ステイタス",1,0,0,2,0,0,0 };

/*
.....
*
*           無 手 順 ファ イ ル 転 送
*
*
.....
*/
void    tty_recive();
void    tty_transm();
void    ctrl_output(char ch);
int  xfkeychk();
#define EDFCMASS    " 同 名 の ファ イ ル が 存 在 し ま す 。 上 書 は ( Y )"
/*
.....
*
*           ファ イ ル 受 信
*
.....
*/
void    tty_recive()
{
    static WINDOW wd={ 2,22,76,3,T_RED,T_WHITE,T_WHITE,0,"",1,1,0,2,1,0,0 };
    static char data[1][DATA_WD]={ EDFCMASS };
    char    c1,c2;
    int Flag,x,y;
    unsigned int    c;

    x=wherex();
    y=wherey();
    if(rdxfile(&MultiFile[0][0])==YES){
        fclose(fp);
        twindow(1, WD_OPEN,&wd.data );
        while((c1=(char)key_in())!=0);
        twindow( WD_CLOSE );
        gotoxy(x,y);
        textcursor(DISP_CURSOR);
        if(c1=='Y' && c1!='y'){
            setatrib();
            setdisk(CDsk);
            chdir(CDir);
            return;
        }
    }
    if(wrxfile(&MultiFile[0][0])==ERR){
        open_errmess();
        window(1,1,80,24);
        gotoxy(x,y);
    }
    else{
        textreverse(NOREVERSE);
        textcursor(DISP_CURSOR);
        sioclear();
        Flag=0;
        while(1){
            if(xfkeychk()==FALSE)    break;
            if(siornum()!=0){
                Flag=0;
                c=ksioinput();
                c1=(char)(0x00ff&(c)>>8);
                c2=(char)(0x00ff&c);
                if(c1!=0){
                    textcolor(co.recv);

```



```

        ctrl_output(c1);
        fputc((0x00ff&c1),fp);
    }
    textcolor(co.recv);
    ctrl_output(c2);
    fputc((0x00ff&c2),fp);
    if (feof(fp)!=NULL) break;
}
else{
    Flag++;
    if (Flag>100){
        putch(BELL);
        ksiooutput(0x00ff&CR);
        break;
    }
    delay(100);
}
}
}
fclose(fp);
putch(BELL);
setatrib();
textcursor(DISP_CURSOR);
setdisk(CDsk);
chdir(CDir);
}

/*
.....
*          ファイル送信
*          .....
*/
void    tty_transm()
{
    char    c1,c2;
    unsigned int    c;
    int j,x,y;

    x=wherex();
    y=wherey();
    if (rdxfile(&MultiFile[0][0])!=ERR)
    {
        open_errmess();
        window(1,1,80,24);
        gotoxy(x,y);
        setatrib();
        textcursor(DISP_CURSOR);
        setdisk(CDsk);
        chdir(CDir);
        return;
    }
    textreverse(NOREVERSE);
    textcursor(DISP_CURSOR);
    j=0;
    while(1){
        while(siornum()!=0) ksioinput();
        c1=(char)fgetc(fp);
        if (feof(fp)!=0){
            sleep(10);
            while(siornum()!=0) ksioinput();
            break;
        }
        textcolor(co.send);
        ctrl_output(c1);
        j++;
        if (iskanji(0x00ff&c1)!=0){
            c=0xff00&((0x00ff&c1)<<8);
            c2=(char)fgetc(fp);
            ctrl_output(c2);
            c=c|(0x00ff&c2);
            textcolor(co.send);
            j++;
        }
        else    c=(0x00ff&c1);
    }
}

```

```

        if(j>LineNo){
            /* 強制改行 */
            ksiooutput(0x00ff&CR);
            if(LineFeed!=0) ksiooutput(0x00ff&LF);
            j=0;
        }
        else{
            if(c==(0x00ff&CR)){
                ksiooutput(0x00ff&CR); j=0;
            }
            else if(c==(0x00ff&LF)){
                if(LineFeed!=0) ksiooutput(0x00ff&LF); j=0;
            }
            else ksiooutput(c);
        }
        if(xfkeychk()==FALSE) break;
    }
    fclose(fp);
    ksiooutput(0x00ff&CR);
    setatrib();
    textcursor(DISP_CURSOR);
    setdisk(CDisk);
    chdir(CDir);
    return;
}

```

/*

----- 無手順ファイル転送のキーチェック -----

```

/*
int xfkeychk()
{
    char    c1,c2;
    int c;

    if((c1=key_in())!=0){
        if(c1==ESC) return FALSE;
        else{
            if(iskanji(c1)!=0){
                c=0xff00&((0x00ff&c1)<<8);
                c2=(char)key_in();
                c!=(0x00ff&c2);
            }
            else c=0x00ff&c1;
            ksiooutput(c);
            return 0;
        }
    }
    else return YES;
}

```

/*

----- 制御コードを含めた出力 -----

```

/*
void ctrl_output(char ch)
{
    if(ch==LF || ch==CR || ch>=' '){
        {
            putchar(ch);
        }
    }
    else{
        {
            textcolor(co.control);
            putchar('^');
            putchar(ch+0x40);
        }
    }
}

```

/*

```

.....
XMODEMファイル転送
.....

```

```

.....
*/
#define      SOH      0x01
#define      STX      0x02
#define      EOT      0x04
#define      ACK      0x06
#define      NAK      0x15
#define      CAN      0x18
#define      RETRYMAX  10

void      xmod_receive();
int  xrmodem();
int  xrecv_pak1(char *buff,int rec_no,int *num);
int  xrecv_pak2(char *buff,int rec_no,int *num);
void  xrcvs(char *buff,int num);

void      xmod_transm();
int  xsmodem();
int  xsend_pak1(char *buffer,int rec_no,int *num);
void  xsend_pak2(char *buffer,int rec_no,int *num);
int  xsend_last();
char  checkcrc();

unsigned int  calc_crc(char *buffer,int len);

int  BlockNO;
int  XMCRC_flag=0;      /* CRC タイプフラグ  none at CRC */

/*
.....
      ファイル受信
.....
*/
void      xmod_receive()
{
static WINDOW wd=( 2,22,76,3,T_RED,T_WHITE,T_WHITE,0,"",1,1,0,2,1,0,0 );
static char data[1][DATA_WD]=( EDFCMASS );
char      FILE[13],ch;
int  sts;

      if(rdxfile(&MultiFile[0][0])==YES)
      {
          fclose(fp);
          twindow(1, WD_OPEN,&wd,data );
          while((ch=(char)(0x00ff&key_in()))==0) ;
          twindow( WD_CLOSE );
          if(ch!='Y' && ch!='y')  return;
      }
      if(wrxfile(&MultiFile[0][0])==ERR)
      {
          open_errmess();
          siooutput(CAN); /* 処理ファイル無し */
          return;
      }
      ftran_waku("ダウンロード","X m o d e m");
      setmem(FILE,13,0);
      getxfilnam(FILE,0);
      disp_XFILE(FILE);
      disp_FLENGTH(0);
      disp_PLENGTH(0);
      disp_PBLOCK(0);
      disp_RETRY(0);
      BlockNO=0;
      sts=xrmodem();
      fclose(fp);
      twindow( WD_CLOSE );
      switch(sts)
      {
      case      TIMEOUT:      time_errmess(); return;
      case      PEND:      dnend_mess(); return;
      case      FILEERR:      xfil_errmess(); return;
      case      FEND:      break;
      default:      break;
      }

```

```

    }
    putch(BELL);
}

/*
.....
.      受信処理      .
.....
*/

int xrmodem()
{
    char    buffer[1100],ch;
    int  retry,i,s,sts,secnum,len;

    disp_STATUS("受信同期中");
    sioclear();
    sleep(10);
    for(i=0;i<RETRYMAX;i++)
    {
        disp_RETRY(i);
        if((ch=checkerc())!=0)
        {
            switch(ch)
            {
                case 'C':    XMCRC_flag=0xff;    break;
                case NAK:    XMCRC_flag=0;    break;
                case ESC:    return    PEND;
                default:    break;
            }
            break;
        }
    }
    if(i==RETRYMAX)    return    TIMEOUT;
    XBYTE=0;
    secnum=1;
    retry=0;
    disp_STATUS("データ受信中");
    while(1)
    {
        disp_RETRY(retry);
        if((char)key_in()==ESC) return    PEND;
        if(retry++>RETRYMAX){
            fclose(fp);
            return    TIMEOUT;
        }
        sts=xrecv_pak1(buffer,secnum,&len);
        if(sts==YES)
        {
            if(secnum==1){
                if(XMCRC_flag==0 && len==128){
                    disp_STATUS("受信(128/SUM)");
                }
                else if(XMCRC_flag!=0 && len==128){
                    disp_STATUS("受信(128/CRC)");
                }
                else if(XMCRC_flag!=0 && len==1024){
                    disp_STATUS("受信(1024/CRC)");
                }
            }
            if(write(fh,&buffer[2],len)==ERR)
            {
                fclose(fp);
                return    FILEERR;
            }
            retry=0;
            XBYTE+=len;
            disp_PLENGTH(XBYTE);
            disp_FLENGTH(XBYTE);
            BlockNO++;
            disp_PBLOCK(BlockNO);
            secnum++;
            siocoutput(ACK);
        }
    }
}

```

```

        else if(sts==FEND){
            siooutput(ACK);
            fclose(fp);
            disp_STATUS("受信終了");
            sleep(5);
            return FEND;
        }
        else{
            if(XMCRC_flag!=0) siooutput('C');
            else siooutput(ACK);
        }
    }
}

/*
-----
エラーチェック付きパケット受信
-----
*/
int xrecv_pak1(char *buff,int rec_no,int *num)
{
    int i,sts;
    char ch;

    for(i=0;i<5;i++){
        ch=sioinput();
        if(ch==STX || ch==SOH || ch==EOT) break;
    }
    if(i==5) return FALSE;
    if(ch==SOH){
        sts=xrecv_pak2(buff,rec_no,128);
        *num=128;
        return sts;
    }
    else if(ch==STX){
        sts=xrecv_pak2(buff,rec_no,1024);
        *num=1024;
        return sts;
    }
    else return FEND;
}

/*
-----
パケット受信
-----
*/
int xrecv_pak2(char *buff,int rec_no,int num)
{
    int i,j;
    unsigned int crc,mycrc;
    char sum;

    xrcvs(buff,num);
    if(0xFF!=(*(buff+*(buff+1)))) return FALSE;
    if(*(buff!=(char)rec_no) return FALSE;
    if(XMCRC_flag==0){
        sum=0;
        for(i=0;i<num;i++) sum+=*(buff+2+i); /* チェックサム */
        if(*(buff+num+2)!=sum) return FALSE;
        else return YES;
    }
    else{
        crc=calc_crc(buff+2,num); /* CRC チェック */
        mycrc=0x00ff&*(buff+num+2);
        mycrc=(0x1f00&(mycrc<<8))|(0x00ff&*(buff+num+3));
        if(crc!=mycrc) return FALSE;
        else return YES;
    }
}

/*
-----
1 パケット受信
-----
*/

```

```

-----
/*
void    xrcvs(char *buff,int num)
{
    int i,ch,n;

    if(XMCRC_flag==0)    n=num+3;
    else                  n=num+4;
    for(i=0;i<n;i++){
        ch=sioinput();
        *(buff+i)=ch;
    }
}

/*
-----
XMODEM/CRCのチェック
-----
/*
char    checkcrc()
{
    int i,j,k;

    for(j=0;j<4;j++){
        siooutput('C');
        for(k=0;k<3;k++){
            {
                if(siornum()!=0)    return 'C';
                sleep(1);
            }
            if((char)key_in()==ESC) return ESC;
        }
        siooutput(NAK);
        for(k=0;k<3;k++){
            if(siornum()!=0)    return NAK;
            if((char)key_in()==ESC) return ESC;
            sleep(1);
        }
        return NULL;
    }
}

/*
.....
・          ファイル送信          ・
.....
/*
void    xmod_transm()
{
    int sts;
    char    FILE[13];
    long int flen;

    if(rdxfile(&MultiFile[0][0])==ERR)
    {
        open_errmess();
        siooutput(CAN); /* 処理ファイル無し */
        return;
    }
    if((flen=filelength(fh))==ERR)
    {
        open_errmess();
        siooutput(CAN); /* 処理ファイルエラー */
        return;
    }
    XBYTE=0;
    ftran_waku("アップロード","Xmodem");
    setmem(FILE,13,0);
    getxfilnam(FILE,0);
    disp_XFILE(FILE);
    disp_FLENGTH(flen);
    disp_PLENGTH(0);
    disp_PBLOCK(0);
    disp_RETRY(0);
    BlockNO=0;
}

```

```

    sts=xsmodem();
    fclose(fp);
    twindow(WD_CLOSE);
    switch(sts)
    {
        case TIMEOUT:    time_errmess(); return;
        case PEND:       dnend_mess(); return;
        case FILEERR:    xfil_errmess(); return;
        case FEND:       break;
        default:         break;
    }
    putchar(BELL);
}

/*
.....
*          送信処理
.....
*/
int xsmodem()
{
    int i,sts,secnum,len;
    char    retry,buffer[1024],ch;

    disp_STATUS("送信同期中");
    sioclear();
    for(i=0;i<60;i++){
        disp_RETRY(i/10);
        ch=sioinput();
        if(ch=='C')
        {
            XMCRC_flag=0xff;    /* set CRC_flag */
            len=1024;
            break;
        }
        else if(ch==NAK)
        {
            XMCRC_flag=0;        /* set check sum */
            len=128;
            break;
        }
        else sioclear();
        if((char)key_in()==ESC) return PEND;
    }
    if(i==60) return TIMEOUT;
    secnum =1;
    retry=0;
    sts=YES;
    disp_STATUS("データ送信中");
    while(1){
        disp_RETRY(retry);
        if((char)key_in()==ESC) return PEND;
        if(retry++>RETRYMAX)
        {
            fclose(fp);
            return TIMEOUT;
        }
        setmem(buffer,len,0x1a);
        if(sts==YES && read(fh,buffer,len)<=0)
        {
            fclose(fp);
            sioclear();
            if(xsend_last()==YES){
                disp_STATUS("送信終了");
                sleep(5);
                return FEND;
            }
            else return TIMEOUT;
        }
        sioclear();
        sts=xsend_pak1(buffer,secnum,&len);
        if(sts==YES){
            if(secnum==1){
                if(XMCRC_flag==0 && len==128){

```



```

        for(i=0;i<=num;i++){
            siooutput(*(buffer+i)); /* Send 1-bytes */
            sum+=*(buffer+i); /* Add check-sum */
        }
        siooutput(sum); /* Send check-sum */
    }
    else
    {
        for(i=0;i<=num;i++){
            siooutput(*(buffer+i)); /* Send 1-bytes */
        }
        crc=calc_crc(buffer,num);
        siooutput((char)(0x00ff&(crc>>8)));
        siooutput((char)(0x00ff&crc));
    }
}

```

```
/*
```

```
-----
                        最終パケット送信
-----
```

```
/*
int xsend_last()
{
    int i;
    char ch;

    siooutput(EOT); /* Send EOT */
    if(ACK==sioinput()) return YES;
    return FALSE;
}

```

```
/*
.....
-
-      Y M O D E M ファイル転送
-
-
.....
*/

```

```

void ymod_receive();
int yrmodem();
int yrecv_pak1(char *buff,int rec_no);
int yrecv_pak2(char *buff,int rec_no,int len);
void yrcvs(char *buff,int len);
int ymod_rdfnam(char *buff,int rec_no);
void ymod_transm();
int ysmodem();
int ysend_pak1(char *buff,int rec_no);
void ysend_pak2(char *buff,int rec_no);
void ysend_pak3(char *buff,int rec_no);
int ysend_last();
int ymod_snfnam(char *buff,int *line);
int chkechar();
unsigned int calc_crc(char *buff,int len);

```

```
/*
.....
-
-      Y M O D E M ファイル受信
-
-
.....
*/

```

```

void ymod_receive()
{
    int sts;

    if(getxdir(&MultiFile[0][0])==ERR)
    {
        dir_errmess();
        return;
    }
    ftran_waku("ダウンロード","Y m o d e m");
    sts=yrmodem();
    fclose(fp);
}

```

```

twindow(WD_CLOSE);
switch(sts)
{
case TIMEOUT:    time_errmess(); return;
case PEND:       dnend_mess();   return;
case FILEERR:    xfil_errmess(); return;
case FEND:       break;
default:         break;
}
putch(BELL);
}

/*
.....
.          受信処理          .
.....
*/

int yrmodem()
{
    char    buffer[100];
    int i,j,sts,secnum;

    XBYTE=0;
    disp_STATUS("受信同期中");
    disp_FLENGTH(0);
    disp_PLENGTH(0);
    disp_PBLOCK(0);
    disp_RETRY(0);
    BlockNO=0;
    sleep(10);
    for(i=0;i<60;i++)        /* 60秒間待つ */
    {
        disp_RETRY(i/10);
        sioclear();
        siooutput('C');
        sleep(1);
        if(siorun()!=0)      break;
        if((char)key_in()==ESC) return PEND;
    }
    if(i>=60) return TIMEOUT;
    while(1)
    {
        if((char)key_in()==ESC) return PEND;
        secnum=0;
        XBYTE=0;
        disp_PLENGTH(XBYTE);
        disp_FLENGTH(XBYTE);
        BlockNO=0;
        disp_PBLOCK(BlockNO);
        sts=ynod_rdfnam(buffer,secnum);
        if(sts==FEND)
        {
            siooutput(ACK);
            disp_STATUS("受信終了");
            sleep(5);
            return FEND;
        }
        else if(sts==YES) siooutput('C');
        else return sts;
        secnum++;
        while(1)
        {
            sts=yrecv_pak1(buffer,secnum);
            if(sts==TIMEOUT) return TIMEOUT;
            if((char)sts==SOH)
            {
                if(write(fh,&buffer[2],128)==-1)
                {
                    fclose(fp);
                    return FILEERR;
                }
                XBYTE+=128;
                disp_PLENGTH(XBYTE);
            }
        }
    }
}

```

```

        BlockNO++;
        disp_PBLOCK(BlockNO);
        secnum++;
    }
    else if((char)sts==STX)
    {
        if(write(fh,&buffer[2],1024)==-1)
        {
            fclose(fp);
            return FILEERR;
        }
        XBYTE+=1024;
        disp_PLENGTH(XBYTE);
        BlockNO++;
        disp_PBLOCK(BlockNO);
        secnum++;
    }
    else if((char)sts==EOT)
    {
        siooutput('C');
        break;
    }
    else return TIMEOUT;
}
}
)

```

エラーチェック付きパケット受信

```

/*
int yrecv_pak1(char *buff,int rec_no)
{
    int retry,sts;
    char ch;

    for(retry=0;retry<RETRYMAX;retry++){
        disp_RETRY(retry);
        ch=sioinput();
        if(ch==SOH)
        {
            disp_STATUS("受信中(128)");
            sts=yrecv_pak2(buff,rec_no,128);
            if(sts==YES)
            {
                siooutput(ACK);
                return 0x00ff&SOH;
            }
            else siooutput(NAK);
        }
        else if(ch==STX)
        {
            disp_STATUS("受信中(1024)");
            sts=yrecv_pak2(buff,rec_no,1024);
            if(sts==YES)
            {
                siooutput(ACK);
                return 0x00ff&STX;
            }
            else siooutput(NAK);
        }
        else if(ch==EOT)
        {
            siooutput(ACK);
            fclose(fp);
            return 0x00ff&EOT;
        }
        else{
            sleep(1);
            sioclear();
            siooutput('C');
        }
    }
}

```

```

    return TIMEOUT;
}

/*
-----
                パケット受信
-----
*/
int yrecv_pak2(char *buff,int rec_no,int len)
{
    unsigned int    crc;
    unsigned int    mycrc;

    yrcvs(buff,len);
    if(0xFF!=(*(buff+++(buff+1)))) return FALSE; /* REC No. のチェック */
    else if(*(buff!=(char)rec_no) return FALSE;
    crc=crc_crc(buff+2,len);
    mycrc=0x00ff8(*(buff+len+2));
    mycrc=(0xff008(mycrc<<9))|(0x00ff8(*(buff+len+3)));
    if(crc!=mycrc) return FALSE;
    else return YES;
}

/*
-----
                1 パケット受信
-----
*/
void yrcvs(char *buff,int len)
{
    int i;
    char ch;

    for(i=0;i<(len+4);++i){
        ch=sioinput();
        *(buff+i)=ch;
    }
}

/*
-----
                ファイル名取得
-----
*/
int ynod_rdfnam(char *buff,int rec_no)
{
    char file[13];
    char number[6];
    char *p,ch;
    int retry,i;
    long int xlen;

    setmem(buff,128,0);
    p=(buff+2);
    for(retry=0;retry<RETRYMAX;retry++){
        disp_RETRY(retry);
        ch=sioinput();
        if(ch==SOH)
        {
            disp_STATUS("ファイル名受信");
            if(yrecv_pak2(buff,rec_no,128)==YES){
                siooutput(ACK);
                break;
            }
        }
        else siooutput('C');
    }
    if(retry==RETRYMAX) return TIMEOUT;
    if(*(buff+2)==0) return FEND; /* NULL in DATA Buffer */
    setmem(file,13,0);
    for(i=0;i<12;i++){
        if(*p==0) break;
        file[i]=*p;
    }
}

```

```

        p++;
    }
    p++;
    setmem(number,6,0);
    for(i=0;i<6;i++)
    {
        if(*p>='0' && *p<='9')
        {
            number[i]=*p;
            p++;
        }
        else break;
    }
    xlen=atoi(number);
    disp_XFILE(file);
    disp_FLENGTH(xlen);
    if(wrxfile(file)!=YES) return FILEERR;
    return YES;
}

/*
.....
*          Y M O D E M 送 信
*          .....
*/
void    ymod_transm()
{
    int sts;

    ftran_waku("アップロード","Y m o d e m");
    sts=ysmodem();
    fclose(fp);
    twindow(WD_CLOSE);
    switch(sts)
    {
        case TIMEOUT:    time_errmess(); return;
        case PEND:       upend_mess(); return;
        case FILEERR:    xfil_errmess(); return;
        case FEND:       break;
        default:         break;
    }
    putch(BELL);
}

/*
.....
*          送 信 処 理
*          .....
*/
int ysmodem()
{
    int i,sts,dsts,secnum;
    char    buffer(1100);
    char    ch;
    int line=0;

    sioclear();
    XBYTE=0;
    disp_STATUS("送信周期中");
    disp_PLENGTH(0);
    disp_FLENGTH(0);
    disp_PBLOCK(0);
    BlockNO=0;
    for(i=0;i<60;i++){
        disp_RETRY(i/10);
        ch=siinput();
        if(ch=='C') break;
        else sioclear();
        if((char)key_in()==ESC) return PEND;
    }
    if(i==60) return TIMEOUT;
    sioclear();
    while(1){
        if((char)key_in()==ESC) return PEND;

```

```

XBYTE=0;
disp_PLENGTH(0);
disp_FLENGTH(0);
BlockNO=0;
disp_PBLOCK(BlockNO);
sts=yndod_snfnan(buffer,&line);
if(sts==FEND)
{
    disp_STATUS("送信終了");
    sleep(5);
    return FEND;
}
else if(sts!=YES) return sts;
if((sts=chkcochar())!=YES) return sts;
secnum=1;
while(1)
{
    setmen(buffer,1024,0x1a);
    dsts=read(fh,buffer,1024);
    if(dsts==0)
    {
        fclose(fp);
        sioclear();
        if((sts=yndod_last())!=YES) return sts;
        if((sts=chkcochar())==YES) break;
        else return sts;
    }
    else if(dsts==-1)
    {
        fclose(fp);
        return FILEERR;
    }
    sioclear();
    XBYTE+=1024;
    disp_PLENGTH(XBYTE);
    BlockNO++;
    disp_PBLOCK(BlockNO);
    sts=yndod_pak1(buffer,secnum);
    if(sts==YES) secnum++;
    else return TIMEOUT;
}
line++;
}
}

/*
-----
エラー付きパケット送信
-----
*/
int ysend_pak1(char *buff,int rec_no)
{
    int retry;
    char ch;

    disp_STATUS("データ送信中");
    for(retry=0;retry<RETRYMAX;++retry){
        disp_RETRY(retry);
        ysend_pak2(buff,rec_no);
        ch=siocinput();
        if(ch==ACK) return YES;
    }
    return TIMEOUT;
}

/*
-----
パケット送信
-----
*/
void ysend_pak2(char *buff,int rec_no)
{
    unsigned int crc; /* check-sum */
    char mycrc;

```

```

int i;

siooutput(STX);          /* Send Start of Packets */
siooutput((char)rec_no);  /* Send recording number */
siooutput((char)0xFF-(char)rec_no); /* Send !recording number */
for(i=0;i<1024;i++)
{
    siooutput(*(buff+i));
}
crc=calc_crc(buff,1024); /* get crc */
mycrc=(char)((crc>>8)&0x00ff);
siooutput(mycrc);        /* Send check-sum */
mycrc=(char)(0x00ff&crc);
siooutput(mycrc);
}

/*
-----
                パケット送信
-----
*/
void    ysend_pak3(char *buff,int rec_no)
{
    int i;          /* internal counter */
    unsigned int crc; /* check-sum */
    char    mycrc;

    siooutput(SOH); /* Send Start of Packets */
    siooutput((char)rec_no); /* Send recording number */
    siooutput((char)0xFF-(char)rec_no); /* Send !recording number */
    for(i=0;i<128;i++)
    {
        siooutput(*(buff+i));
    }
    crc=calc_crc(buff,128); /* get crc */
    mycrc=(char)((crc>>8)&0x00ff);
    siooutput(mycrc);        /* Send check-sum */
    mycrc=(char)(0x00ff&crc);
    siooutput(mycrc);
}

/*
-----
                ファイル終了パケット送信
-----
*/
int ysend_last()
{
    int i,sts;

    disp_STATUS("ファイル終了");
    for(i=0;i<RETRYMAX;++i){
        disp_RETRY(i);
        siooutput(EOT); /* Send EOT */
        if(sioinput()==ACK) return YES;
    }
    return TIMEOUT;
}

/*
-----
                ファイル名送信
-----
*/
int ymod_snfnam(char *buff,int *line)
{
    char    file[53];
    long    flen,unixtime=0;
    unsigned int    crc;
    int slen,retry,sts;
    struct date    DATE;
    struct time    TIME;

    setmon(buff,140,0);

```

```

if(gnxtfil(line)==FEND)
{
    disp_STATUS("終了パケット");
    for(retry=0;retry<RETRYMAX;retry++)
    {
        disp_RETRY(retry);
        ysend_pak3(buff,0);
        if(sioinput()==ACK) return FEND;
    }
    return TIMEOUT;
}
if(rdxfile(&MultiFile[-line][0])==YES)
{
    getxfilnam(buff,-line);
    disp_XFILE(buff);
    flen=filelength(fh);
    disp_FLENGTH(flen);
    slen=strlen(buff);
    sprintf(&buff[slen+1],"%06ld ",flen);
    getdate(&DATE);
    gettime(&TIME);
    unixtodos(unixtime,&DATE,&TIME);
    sprintf(&buff[slen+1+7],"%ld 0",unixtime);
    disp_STATUS("ファイル名送信");
    for(retry=0;retry<RETRYMAX;retry++)
    {
        disp_RETRY(retry);
        ysend_pak3(buff,0);
        if(sioinput()==ACK) return YES;
    }
    return TIMEOUT;
}
else return FILEERR;
}

```

/*

----- チェック [C] キャラクタ -----

*/

```

int chkechar()
{
    int i;
    char ch;

    for(i=0;i<10;i++){
        ch=sioinput();
        if(ch=='C') return YES;
    }
    return TIMEOUT;
}

```

/*

----- CRC の計算 -----

*/

```

unsigned int calc_crc(char *buffer,int len)
{
    unsigned int crc;
    int i;

    crc=0;
    while(--len>=0)
    {
        crc=crc^(unsigned int)*buffer++ << 8;
        for(i=0;i<8;i++)
        {
            if(crc & 0x8000)
            {
                crc=(crc << 1)^0x1021;
            }
            else
            {
                crc << 1;
            }
        }
    }
}

```



```

        crc=(crc<<1);
    }
}
return 0xffff&crc;
}

.....
*
* K E R M I Tファイル転送用プロトコル サポートルーチン
*
* Created from KERMIT-Overviews/Instructions
* by Frank de Culz in Colombia university
*
* This KERMIT is The Public Domein Programs for
* All the people
*
.....
#define SP          0x20      /* Space */
#define CTRLD       0x04      /* Control-D for UNIX */
#define BRKCHR      CTRLD     /* is a break character */
#define MAXTRY      5         /* Maximum numbers of Retry */

#define MAXPAK      80        /* MAX of Packet length */
#define TIMEOUTSEC  5         /* Timeout in Seconds */
#define PADDING     0         /* Number of Padding character */
#define PADCHAR     0         /* Padding character */
#define MYEOL       CR        /* EOL characters */
#define QUOTE7      '#'       /* Control -Quoting character */
#define QUOTE8      '&'       /* 8-bit Quoting character */
#define CHKSUM      '1'       /* Checking Method is check-sum rules */
#define REPEAT      ' '       /* Repet character */
#define CAPAS       0         /* No flag */

void kerm_receive();
void kerm_transm();
char rpack(int *len,int *num,char *data);
int spack(char type,int num,int len,char *data);
void trs_menu3(char ch,char *mass);
char sinit();
char sfile();
char rinfo();
char sdata();
char seof();
char sbreak();
char rinit();
char rfile();
char rdata();
void bufemp(char *buffer,int len);
char tochar(char ch);
char unchar(char ch);
char ctl(char ch);

/*
.....
* グローバル変数
*
.....
*/

int size;      /* パケット・サイズ */
int n;         /* メッセージ番号 */
int numtry;    /* リトライ数 */
int oldtry;

char state;    /* K E R M I Tのカレントステータス */
int spsiz;    /* 送信用パケットサイズ */
int timint;   /* 外部からのタイムアウト秒数 */
int pad;      /* P A D文字数 */
char padchar; /* P A D文字 */
char eol;     /* 受信用 E O L 文字 */
char quote;   /* 制御用クォート文字 */
char quote8;  /* 8-bit クォート文字 */
char chkmd;   /* チェック方法 */
char repeat;  /* クォート文字の繰り返し */

```

```

char    capflag;      /* 拡張機能フラグ */

char    recpkt(MAXPAK); /* 受信用バケットバッファ */
char    packet(MAXPAK); /* 送信用バケットバッファ */
char    type;         /* パケット・タイプ */
char    k_filename[13]; /* ファイル名バッファ */

int LINE;             /* ファイル・バッファ・ライン */
int FSIZE;            /* ファイルの読み出しサイズ */
int FLAGA;

/*
.....
*
*          K E R M I Tファイル受信
*
.....
*/
void    kerm_receive()
{
    int sts;

    if (getxdir(&MultiFile[0][0]) == ERR)
    {
        dir_errmess();
        return;
    }
    ftran_waku("ダウンロード","K e r m i t");
    XBYTE=0;
    disp_FLENGTH(0);
    disp_PLENGTH(0);
    disp_PBLOCK(0);
    BlockNO=0;
    sts=recsw();
    fclose(fp);
    twindow(WD_CLOSE);
    switch(sts)
    {
        case TIMEOUT:    time_errmess(); return;
        case PEND:       dnend_mess();   return;
        case FILEERR:    xfil_errmess(); return;
        case FEND:       break;
        default:         break;
    }
    putch(BELL);
}

/*
.....
*
*          受信モード処理
*
.....
*/
int recsw()
{
    state='R';
    n=0;
    numtry=0;
    XBYTE=0;
    BlockNO=0;
    disp_FLENGTH(0);
    disp_PLENGTH(0);
    disp_PBLOCK(0);
    disp_RETRY(0);
    disp_STATUS("受信同期中");
    while(TRUE)
    {
        if ((char)key_in() == ESC) return PEND;
        disp_RETRY(numtry);
        switch(state)
        {
            case 'D':     disp_STATUS("データバケット");
                          state=rdata(); break;
            case 'F':     disp_STATUS("ファイル名バケット");
                          state=rfile(); break;
        }
    }
}

```

```

        case 'R':    disp_STATUS("初期化バケット");
                      state=rinit(); break;
        case 'A':    disp_STATUS("アトリビュート");
                      state=rinfo(); break;
        case 'C':    disp_STATUS("受信終了");
                      sleep(5);
                      return FEND;
        case 'U':    return PEND;
        case 'T':    return TIMEOUT;
        case 'E':    return FILEERR;
        default:     break;
    }
}

```

```

/*

```

```

-----
パラメータバケット受信
-----

```

```

/*
char    rinit()
{
    int num,len;

    if(numtry++>MAXTRY) return 'T';
    switch(rpack(&len,&num,packet))
    {
        case 'S':
            rpar(packet);
            spar(packet);
            spack('Y',n,10,packet);
            oldtry=numtry;
            numtry=0;
            n=(n+1)%64;
            return 'F';
        case 'A':
            spack('N',n,0,0);
            return state;
        default:     return 'U';
    }
}

```

```

/*

```

```

-----
ファイル名バケット受信
-----

```

```

/*
char    rfile()
{
    int num,len;

    if(numtry++>MAXTRY) return 'T';
    switch(rpack(&len,&num,packet))
    {
        case 'S':
            if(oldtry++>MAXTRY) return 'T';
            if(num==(n==0)?63:n-1)
            {
                spar(packet);
                spack('Y',num,10,packet);
                numtry=0;
                return state;
            }
            else
            {
                spack('N',num,0,0);
                numtry++;
                return state;
            }
        case 'Z':
            if(oldtry++>MAXTRY) return 'T';
            if(num==(n==0)?63:n-1)
            {
                spack('Y',num,0,0);
            }
    }
}

```

```

        numtry=0;
        return state;
    }
    else(
        spack('N',num,0,0);
        numtry++;
        return state;
    )
case 'F':
    if(num!=n){
        spack('N',n,0,0);
        numtry++;
        return state;
    }
    setmem(k_filnam,13,0);
    strncpy(k_filnam,packet,len);
    disp_XFILE(k_filnam);
    if(wrxfile(k_filnam)!=YES) return 'E';
    spack('Y',n,0,0);
    oldtry=numtry;
    numtry=0;
    n=(n+1)%64;
    return 'A';
case 'B':
    if(num!=n)
    {
        spack('N',n,0,0);
        numtry++;
        return state;
    }
    spack('Y',n,0,0);
    return 'C';
case FALSE:
    spack('N',n,0,0);
    return state;
default: return 'U';
}
)

```

/*

----- ファイル情報の取得 -----

/*

```

char rinfo()
{
    int num,len;
    long int flen;
    char data1[21];
    char data2[11];
    char *p,ch;

    if(numtry++>MAXTRY) return 'T';
    switch(rpack(&len,&num,packet))
    {
        case 'F':
            if(oldtry++>MAXTRY) return 'T';
            if(num==((n==0)?63:n-1)){
                spar(packet);
                spack('Y',n,10,packet);
                numtry=0;
                return state;
            }
            else
            {
                spack('N',num,0,0);
                numtry++;
                return state;
            }
        case 'A':
            if(num!=n){
                if(oldtry++>MAXTRY) return 'T';
                if(num==((n==0)?63:n-1))
                {

```

```

        spack('Y', num, 10, packet);
        numtry=0;
        return state;
    )
    spack('N', n, 0, 0);
    numtry++;
    return state;
}
if (FLAGA==0) {
    setmem(data1, 21, 0);
    strncpy(data1, packet, 20);
    if ((p=strstr(data1, "!#"))!=0)
    {
        setmem(data2, 11, 0);
        strncpy(data2, (p+2), 10);
        flen=atoi(data2);
        disp_FLENGTH(flen);
    }
    FLAGA=0xff;
}
spack('Y', n, 0, 0);
oldtry=numtry;
numtry=0;
n=(n+1)%64;
return 'A';
case 'D':
    if (num!=n) {
        if (oldtry++>MAXTRY) return 'T';
        if (num==((n==0)?63:n-1))
        {
            spack('Y', n, 0, 0);
            numtry=0;
            return state;
        }
        spack('N', n, 0, 0);
        numtry++;
        return state;
    }
    bufemp(packet, len); /* データ書き込み */
    XBYTE+=FSIZE;
    if (FLAGA==0) disp_FLENGTH(XBYTE);
    disp_PLENGTH(XBYTE);
    spack('Y', num, 0, 0);
    oldtry=numtry;
    numtry=0;
    n=(n+1)%64;
    return 'D';
case FALSE:
    spack('N', n, 0, 0);
    return state;
default: return 'U';
}
}

```

```

/*
-----
          パケットデータ受信
-----
*/
char    rdata()
{

```

```

    int num, len;

    if (numtry==0) {
        BlockNO++;
        disp_PBLOCK(BlockNO);
    }
    if (numtry++>MAXTRY) return 'T';
    switch (rpack(&len, &num, packet)) {
        case 'D':
            if (num!=n) {
                if (oldtry++>MAXTRY) return 'T';
                if (num==((n==0)?63:n-1))
                {

```

```

        spack('Y', num, 10, packet);
        numtry=0;
        return state;
    }
    spack('N', n, 0, 0);
    numtry++;
    return state;
}
bufemp(packet.len); /* データ書き込み */
XBYTE+=FSIZE;
if (FLAGA==0)    disp_FLENGTH(XBYTE);
disp_PLENGTH(XBYTE);
spack('Y', n, 0, 0);
oldtry=numtry;
numtry=0;
n=(n+1)%64;
return 'D';
case 'F':
    if (oldtry==MAXTRY) return 'T';
    if (num==(n==0)?63:n-1){
        spack('Y', num, 0, 0);
        numtry=0;
        return state;
    }
    else{
        spack('N', n, 0, 0);
        numtry++;
        return state;
    }
case 'Z':
    if (num!=n){
        spack('N', n, 0, 0);
        numtry++;
        return state;
    }
    spack('Y', n, 0, 0);
    fclose(fp);
    n=(n+1)%64;
    return 'F';
case FAIL:
    spack('N', n, 0, 0);
    return state;
default:    return 'U';
}
)
)

```

```

/*
.....
:
:          K E R M I Tファイル送信          :
:
:.....
*/

```

```

void    kerm_transm()
{
    int sts,i;

    ftran_waku("アップロード","K e r m i t");
    sts=sendsw();
    fclose(fp);
    twindow(WD_CLOSE);
    switch(sts)
    {
        case TIMEOUT:    time_errmess(); return;
        case PEND:        upond_mess();    return;
        case FILEERR:     xfil_errmess(); return;
        case FEND:        break;
        default:          break;
    }
    putch(BELL);
}

```

```

/*
-----

```

送信モード処理

```

/*
int sendsw()
{
    state='S';          /* Current status is INITIAL */
    n=0;
    numtry=0;
    LINE=0;
    XBYTE=0;
    eol=MYEOL;
    quote='\"';
    quote8='\\';
    disp_FLENGTH(0);
    disp_PLENGTH(0);
    disp_PBLOCK(0);
    disp_RETRY(0);
    disp_STATUS("送信同期中");
    sleep(2);
    sioclear();
    while(TRUE)
    {
        if((char)key_in()==ESC) return PEND;
        disp_RETRY(numtry);
        switch(state)
        {
            case 'D': disp_STATUS("データバケット");
                      state=sdata(); break;
            case 'F': disp_STATUS("ファイル名バケット");
                      state=sfile(); break;
            case 'Z': disp_STATUS("EOFバケット");
                      state=seof(); break;
            case 'S': disp_STATUS("初期化バケット");
                      state=sinit(); break;
            case 'B': disp_STATUS("EOTバケット");
                      state=sbreak(); break;
            case 'C': disp_STATUS("送信終了");
                      sleep(5);
                      return FEND;
            case 'A': return PEND;
            case 'I': return TIMEOUT;
            case 'E': return FILEERR;
            default: break;
        }
    }
}
*/

```

送信初期化

```

/*
char sinit()
{
    int num,len;
    long int flen;

    if(numtry++>MAXTRY) return 'I';
    spar(packet);          /* Making initial Packet */
    spack('S',n,10,packet); /* Send initial packet */
    switch(rpack(&len,&num,recpkt)) /* Receive Packet */
    {
        case 'N': return state;
        case 'Y':
            if(n!=num) return state;
            rpar(recpkt);
            numtry=0;
            n=(n+1)%64;
            if(rdxfile(&MultiFile[LINE][0])!=YES) return 'E';
            flen=filelength(fh);
            disp_FLENGTH(flen);
            setmem(k_filnam,13,0);
            getxfilnam(k_filnam,LINE);
            disp_XFILE(k_filnam);
    }
}
*/

```

```

        if(eol==0) eol=MYEOL;
        if(quote==0){
            quote='\"';
            quote8='\\';
        }
        return 'F';
    case FALE: return state;
    default: return 'A';
}

/*
-----
                        ファイル名送信
-----
*/
char    sfile()
{
    int num,len;

    if(numtry++>MAXTRY) return 'T';
    XBYTE=0;
    len=strlen(k_filnam);
    spack('F',n,len,k_filnam);
    switch(rpack(&len,&num,recpkt)){
        case 'N':
            if(n!=(num=(--num<0)?63:num)) return state;
        case 'Y':
            if(n!=num) return state;
            numtry=0;
            n=(n+1)%64;
            size=bufill(packet);
            XBYTE+=FSIZE;
            return 'D';
        case FALE: return state;
        default: return 'A';
    }
}

/*
-----
                        パケット送信
-----
*/
char    sdata()
{
    int num,len;

    if(numtry==0){
        BlockNO++;
        disp_PBLOCK(BlockNO);
    }
    if(numtry++>MAXTRY) return 'T';
    disp_PLENGTH(XBYTE);
    spack('D',n,size,packet);
    switch(rpack(&len,&num,recpkt)){
        case 'N':
            if(n!=(num=(--num<0)?63:num)) return state;
        case 'Y':
            if(n!=num) return state;
            numtry=0;
            n=(n+1)%64;
            if((size=bufill(packet))==EOF) return 'Z';
            else{
                XBYTE+=FSIZE;
                return 'D';
            }
        case FALE: return state;
        default: return 'A';
    }
}

/*
-----

```

ファイル終了パケット送信

```

/*
char    seof()
{
    int num,len;
    long int flen;

    if(numtry++>MAXTRY) return 'T';
    spack('Z'.n,0,packet);
    switch(rpack(&len,&num,recpkt))
    {
        case 'N':
            if(n!=(num=(--num<0)?63:num)) return state;
        case 'Y':
            if(n!=num) return state;
            numtry=0;
            n=(n+1)%64;
            fclose(fp);
            LINE++;
            if(gnxtfil(&LINE)==FEND) return 'B';
            if(rdxfile(&MultiFile[LINE][0])!=YES) return 'E';
            flen=filelength(fh);
            disp_FLENGTH(flen);
            setmem(k_filnam,13,0);
            getxfilnam(k_filnam,LINE);
            disp_XFILE(k_filnam);
            BlockNO=0;
            return 'F';
        case FALE: return state;
        default: return 'A';
    }
}
*/

```

ブレイクパケット送信

```

/*
char    sbreak()
{
    int num,len;

    if(numtry++>MAXTRY) return 'T';
    spack('B'.n,0,packet);
    switch(rpack(&len,&num,recpkt)){
        case 'N':
            if(n!=(num=(--num<0)?63:num)) return state;
        case 'Y':
            if(num!=n) return state;
            numtry=0;
            n=(n+1)%64;
            return 'C';
        case FALE: spack('N'.n,0,0); return state;
        default: return 'A';
    }
}
*/

```

1 パケット送信

```

/*
int spack(char type,int num,int len,char *data)
{
    int i;
    char    checksum.buffer[100],*bufp;

    setmem(buffer,100,0);
    bufp=buffer; /* set pointer */
    for(i=1;i<=pad;i++){
        siooutput(padchar);
    }
    *bufp++=SOH; /* Store SOH */
}

```

```

chksum=tochar((char)(len+3)); /* Check-sum culiculate */
*bufp++=tochar((char)(len+3)); /* Store data-length */
chksum+=tochar((char)num); /* Sum */
*bufp++=tochar((char)num); /* Store Sequential number */
chksum+=type; /* Sum */
*bufp++=type; /* Store Packet type */
for(i=0;i<len;i++){ /* Store Data */
    *bufp++=(data+i);
    chksum+=(data+i);
}
chksum=(chksum+(chksum&0xC0)/64)&63;
*bufp++=tochar(chksum);
*bufp=eol; /* Store EOL */
for(i=0;i<((bufp-buffer+1);++i)){
    siooutput(buffer[i]); /* Send Packet to foreign Stations */
}
)

/*
-----
          | パケット受信
-----
*/
char    rpack(int *len,int *num,char *data)
{
    int i,done,t=0;
    char    chksum,type,ch;

    while(SOH!=t){
        if(TIMEOUT==(t=receive(5))) return FALSE;
        t&=0x7f;
    }
    done=FALSE;
    while(!done){
        if(0==(ch=b7sioinput(5))) return FALSE;
        if(ch==SOH) continue;
        chksum=ch;
        *len=0x00ff&(uchar(ch)-3);
        if(0==(ch=b7sioinput(5))) return FALSE;
        if(ch==SOH) continue;
        chksum+=ch;
        *num=0x00ff&uchar(ch);
        if(0==(ch=b7sioinput(5))) return FALSE;
        if(ch==SOH) continue;
        chksum+=ch;
        type=ch;
        for(i=0;i<*len;i++){
            if(0==(ch=b7sioinput(5))) return FALSE;
            if(ch==SOH) continue;
            *(data+i)=ch;
            chksum+=ch;
        }
        data[*len]=0;
        if(0==(ch=b7sioinput(5))) return FALSE;
        if(ch==SOH) continue;
        done=TRUE;
    }
    chksum=(chksum+(chksum&0xC0)/64)&63;
    if(chksum!=uchar(ch)) return FALSE;
    return type;
}

/*
-----
          Read datas from File(fp) with quoting
-----
*/
int bufill(char *buffer)
{
    int i=0;
    char    ch;

    FSIZE=0;
    while(read(fh,&ch,1)>0)

```

```

(
    FSIZE++; /* 読み出しバイト数 */

    /* 8ビットクオートあり */
    if (quote8 != 'Y' && quote8) ' ' && (ch & 0x80) != 0)
    {
        ch &= 0x7f;
        if (ch == quote8)
        {
            buffer[i++] = quote;
            buffer[i++] = quote8;
        }
        else
        {
            buffer[i++] = quote8;
            if (ch < SP || ch == DEL || ch == quote)
            {
                buffer[i++] = quote;
                if (ch != quote) ch = ctl(ch);
            }
            buffer[i++] = ch;
        }
    }
    else
    {
        if (ch < SP || ch == DEL || ch == quote)
        {
            buffer[i++] = quote;
            if (ch != quote) ch = ctl(ch);
        }
        buffer[i++] = ch;
    }
    if (i >= spsiz-8) return i;
}
if (i == 0) return EOF;
else return i;
)

/*
-----
Write buffer to File(fp)
-----
*/
void bufemp(char *buffer, int len)
{
    int i, LEP;
    char ch;

    FSIZE = 0;
    for (i = 0; i < len; i++)
    {
        ch = buffer[i];
        if (cp.length != 0 && (0x7f & ch) == QUOTE8) /* 7bit data */
        {
            ch = buffer[++i] & 0x7f;
            if (ch == QUOTE7)
            {
                ch = buffer[++i] & 0x7f;
                if ((ch & 0x7f) != QUOTE7) ch = ctl(ch);
            }
            ch = ch & 0x80;
            write(fh, &ch, 1);
            FSIZE++;
        }
        else if (ch == QUOTE7)
        {
            ch = buffer[++i];
            if ((ch & 0x7f) != QUOTE7) ch = ctl(ch);
            write(fh, &ch, 1);
            FSIZE++;
        }
        else if (repeat > SP && ch == repeat) /* 反復フリビックス */
        {

```

```

        ch=buffer[++i];
        LEP=0x00ff&unchar(ch);
        ch=buffer[++i];
        if(cp.length!=0 && (0x7f&ch)==QUOTE8)
        {
            ch=buffer[++i]&0x7f;
            if(ch==QUOTE7)
            {
                ch=buffer[++i]&0x7f;
                if((ch&0177)!=QUOTE7)    ch=ctl(ch);
            }
            ch=ch:0x80;
        }
        else if(ch==QUOTE7)
        {
            ch=buffer[++i];
            if((ch&0177)!=QUOTE7)    ch=ctl(ch);
        }
        write(fh,&ch,LEP);
        FSIZE++;
    }
    else if(ch!=CR)
    {
        write(fh,&ch,1);
        FSIZE++;
    }
}

/*
-----
                Make Initial Packet
-----
*/
int spar(char *data)
{
    data[0]=tochar(MAXPAK);    /* MAX of Packet length */
    data[1]=tochar(TIMEOUTSEC); /* Timeout in Seconds */
    data[2]=tochar(PADDING);    /* Number of Padding character */
    data[3]=ctl(PADCHAR);    /* Padding character */
    data[4]=tochar(MYEOL);    /* EOL characters */
    data[5]=QUOTE7;    /* Control -Quoting character */
    if(cp.length==0)    data[6]='Y';    /* 8bit data */
    else    data[6]=QUOTE8;    /* 8-bit Quoting character */
    data[7]=CHKSUM;    /* Checking Method */
    data[8]=REPEAT;    /* Repet character */
    data[9]=tochar(CAPAS);    /* No flag */
}

/*
-----
                Set received initial-packet parameters
-----
*/
int rpar(char *data)
{
    spsiz = 0x00ff&unchar(data[0]); /* Packet size */
    if(spsiz==0)    spsiz=80;    /* default value */
    timint = 0x00ff&unchar(data[1]); /* Timeout in seconds */
    pad = 0x00ff&unchar(data[2]); /* Padding number */
    padchar= ctl( data[3]);    /* Padding character */
    eol = unchar(data[4]);    /* EOL character */
    quote = data[5];    /* control-quoting character */
    quote8 = data[6];    /* 8bit quote character */
    chkmd = unchar(data[7]);    /* check sum motod */
    /* repeat character */
    if(data[8]!=REPEAT) repeat=REPEAT;
    else    repeat=data[8];
    capflag= unchar(data[9]);    /* CAPS flag */
}

/*
-----
                Translate to GO Table
-----

```

```

-----
/*
char    unchar(char ch)
{
    return (ch-' ');
}

/*
-----
character from control code
-----
/*
char    tochar(char ch)
{
    return (ch+' ');
}

/*
-----
Translate to Control-G0
-----
/*
char    ctl(char ch)
{
    return (ch^64);
}

/*
.....
.
.      ファイル転送用ユーティティィー
.
.....
/*

/*
-----
ファイル読み込みオープン
-----
/*
int rdxfile(char *filnam)
{
    fp=fopen(filnam,"rb");
    fh=fopen(filnam,"rb");
    if(fp==NULL) return ERR;
    else return YES;
}

/*
-----
ファイル書き込みオープン
-----
/*
int wrxfile(char *filnam)
{
    fp=fopen(filnam,"wb");
    fh=fopen(filnam,"wb");
    if(fp==NULL) return ERR;
    else return YES;
}

/*
-----
次のファイルの読み込み
-----
/*
int gnxtfil(int *line)
{
    int i;

    for(i=*line;i<10;i++)
    {
        if(MultiFile[i][0]!=0) break;
    }
}

```

```

        if(i>=10)    return FEND;
        *line=i;
        return YES;
    }

/*
-----
                パス付きファイルからファイル名の抽出
-----
*/
void    getxfilnam(char *file,int line)
{
    setmem(XDRIVE,3,0);
    setmem(DIR,50,0);
    setmem(NAME,13,0);
    setmem(EXT,5,0);
    fnsplit(&MultiFile[line][0],XDRIVE,DIR,NAME,EXT);
    strcpy(file,NAME);
    strcat(file,EXT);
    FDSK=0x00ff&(XDRIVE[0]-'A');
}

/*
-----
                ディレクトリの設定
-----
*/
int getxdir(char *pathnam)
{
    setmem(XDRIVE,3,0);
    setmem(DIR,50,0);
    setmem(NAME,13,0);
    setmem(EXT,5,0);
    fnsplit(pathnam,XDRIVE,DIR,NAME,EXT);
    FDSK=0x00ff&(XDRIVE[0]-'A');
    seldisk(FDSK);
    chdir(DIR);
}

/*
-----
                data input from SIO
-----
*/
char    b7sioinput(int seconds)
{
    int ti;
    char    ch;

    ti=receive(seconds);
    if(ti==TIMEOUT)    return 0;
    ch=(char)ti;
    if(cp.length==0)    return ch;
    else                return 0x7f&ch;
}

/*
-----
                タイムアウト付き1字受信
-----
*/
int receive(int seconds)
{
    int i;
    char    ch;

    for(i=0;i<seconds;i++){
        ch=sioinput();
        if(ch!=0)    return 0x00ff&ch;
    }
    return TIMEOUT;
}

/*

```

```

-----
                メッセージ一覧
-----

```

```

/*
void    open_errmess()
{
    errordisp(" 処理ファイルがオープンできません。");
}

void    upend_mess()
{
    static WINDOW wd=( 2,22,76,3,T_RED,T_WHITE,T_WHITE,0,"",2,1,0,0,1,0,0 );
    static char data[1](DATA_WD)=( "アップロードを中止しました。" );

    twindow(1, WD_OPEN,&wd,data );
    twindow( WD_CLOSE );
}

void    dnend_mess()
{
    static WINDOW wd=( 2,22,76,3,T_RED,T_WHITE,T_WHITE,0,"",2,1,0,0,1,0,0 );
    static char data[1](DATA_WD)=( "ダウンロードを中止しました。" );

    twindow(1, WD_OPEN,&wd,data );
    twindow( WD_CLOSE );
}

void    xfil_errmess()
{
    errordisp(" ディスクのエラーです。");
}

void    dir_errmess()
{
    errordisp(" ディレクトリが見つかりません。");
}

void    time_errmess()
{
    errordisp(" 接続先から応答がありません。");
}
*/

```

```

-----
                枠作製
-----

```

```

/*
void    ftran_waku(char *direct,char *protoc)
{
    sprintf(ftdata[1]+17,"%-s",direct);
    sprintf(ftdata[2]+17,"%-s",protoc);
    twindow(1,WD_OPEN,&ftranwp,ftdata);
}
*/

```

```

-----
                処理ファイル名の表示
-----

```

```

/*
void    disp_XFILE(char *file)
{
    sprintf(ftdata[0]+17,"%-s¥n",file);
    twindow(1, WD_REMAIN,&ftranwp,ftdata );
}
*/

```

```

-----
                ファイルサイズ表示
-----

```

```

/*
void    disp_FLENGTH(long int num)
{
    sprintf(ftdata[3]+17,"%06ld¥n" "バイト",num);
}
*/

```

```

    twindow(1, WD_REMAIN, &ftranwp, fldata );
}

/*
-----
      処理サイズ表示
-----
*/
void    disp_PLENGTH(long int num)
{
    sprintf(fldata[4]+17, "%06ldh"イト", num);
    twindow(1, WD_REMAIN, &ftranwp, fldata );
}

/*
-----
      処理ブロック数の表示
-----
*/
void    disp_PBLOCK(int num)
{
    sprintf(fldata[5]+17, "%04d7`ブロック", num);
    twindow(1, WD_REMAIN, &ftranwp, fldata );
}

/*
-----
      リトライ回数表示
-----
*/
void    disp_RETRY(int num)
{
    sprintf(fldata[6]+17, "%02d @", num);
    twindow(1, WD_REMAIN, &ftranwp, fldata );
}

/*
-----
      ステータス表示
-----
*/
void    disp_STATUS(char *status)
{
    sprintf(fldata[7]+17, "%-14s", status);
    twindow(1, WD_REMAIN, &ftranwp, fldata );
}

```


付録N. 各種ハンドラ・ソースファイル(hand.c)

```

/*
.....
*
*               各種ハンドラ
*
.....
*/
#include      <stdio.h>
#include      <conio.h>
#include      <string.h>
#include      <jstring.h>
#include      <jctype.h>
#include      <mem.h>
#include      <dos.h>
#include      <bios98.h>
#include      <time.h>
#include      <graphics.h>
#include      <alloc.h>
#include      <stdlib.h>
#include      "nfire.h"
#include      "fall.h"

#define REC 1  /* recover reverse string */
#define REV 0  /* print reverse string */
#define OK 1
#define NO 0

static char SioBuff[4000]; /* S I O のバッファ */

static char buf[ICON_MAX_NO+2][1000]; /* アイコンパターン用バッファ */
static char ms_im[1000]; /* マウスパターン用バッファ */
static int col_tbl[]={ T_BLACK,T_BLUE,T_RED,T_MAGENTA,T_GREEN,T_CYAN,T_YELLOW,T_
HITE },
col_tbl[]={ BLACK,BLUE,RED,MAGENTA,GREEN,CYAN,YELLOW,WHITE,LIGHTRED,
LIGHTGREEN,LIGHTBLUE,LIGHTCYAN,LIGHTMAGENTA,BROWN },
IPX,IPY, /* メニュー表示位置 */
IPX2=MS_IX+30,IPY2=MS_IY+30; /* (旧表示位置) */
static struct ICON_POS ip[COM_NO]={ /* アイコンポジション設定 */
( 0,80,9,8 ),( 40,80,11,10 ),( 80,0,10,9 ),
( 0,40,0,1 ),( 40,40,1,2 ),( 80,40,2,3 ),
( 0,0,3,4 ),( 40,0,4,5 ),( 80,80,5,6 ), };
static unsigned char disp_buf[1000]; /* アイコン表示用のバッファ */
static struct FN_POS { /* ファイル名位置記憶用構造体 */
int x;
int y;
} Pos[100];

/*..... キー入力関係 .....*/
int key_in();
void key_clr();
int key_getc();
int key_gets();
int cmd_inp();

/*..... プリンター関係 .....*/
void printer(char data);

/*..... ファンクションキー関係 .....*/
extern struct FUN_CUR_KEY funkey[2]; /* ファンクションキーエリア */
void getfunkey(); /* Fキー内容取得 */
void putfunkey(); /* Fキー内容返却 */
void setfunkey(); /* Fキー内容設定 */

/*..... マウス関係 .....*/
int mouseinit(); /* マウスの初期化 */
void mousesense(int mx,int my); /* マウス感度の設定 */
void mouseon(); /* マウスカーソル表示 */
void mouseoff(); /* マウスカーソル消去 */
void mouseposset(int x,int y); /* マウスの位置セット */
int mouseinput(); /* 座標とキー入力 */
int mousediff();

```



```

        Sec--;
        return ( 0 );
    }
    if ( mode==WD_OPEN || mode==FW_OPEN ) /* ウィンドウのオープン */
    {
        Sec++; /* 次のセクションへ */
        UseNum++; /* 次のバッファ領域へ */
        if ( UseNum+MAXPAGE > MAX_PAGE ) return( 0 ); /* これ以上開けない */
        secst[Sec]=UseNum;
        if ( secon[Sec] < secst[Sec] || MAXPAGE==1 ) secon[Sec]=secst[Sec];
        tm = secon[Sec]-secst[Sec];
        for( i=0 ; i <= tm ; ++i ) /* カレントページへと自動的に開く */
        {
            getmem( wd_para,UseNum,wd[i].x-1,wd[i].y /* バッファ領域の獲得 */
                    ,wd[i].x+wd[i].wx-1+1,wd[i].y+wd[i].wy-1 );
            gettext( wd_para[UseNum].x1,wd_para[UseNum].y1, /* エリアの保存 */
                    wd_para[UseNum].x2,wd_para[UseNum].y2,wd_para[UseNum].adr );
            ul=( MAXPAGE > 1 ? wd[i].cmin : 0 );
            flame( mode,MAXPAGE,i,&wd[i] );
            if ( mode != FW_OPEN ) dspdata( ul,0,&wd[i],wddata );
            else fdspdata( ul,&wd[i],wddata );
            UseNum++;
        }
        UseNum--;
    }
    UsePage= secon[Sec]-secst[Sec]; /* 現在開いているページの設定 */
    if ( mode == WD_REMAIN ) {
        if ( Sec==1 ) return(-1);
        ul=( MAXPAGE > 1 ? wd[UsePage].cmin : 0 );
        if ( wd[UsePage].button != 2 )
            flame( -1,MAXPAGE,UsePage,&wd[UsePage] );
        dspdata( ul,0,&wd[UsePage],wddata );
    }
    if ( mode == FW_REMAIN ) {
        ul=0;
        if ( wd[UsePage].button != 2 )
            flame( -1,MAXPAGE,UsePage,&wd[UsePage] );
        fdspdata( ul,&wd[UsePage],wddata );
    }
    if ( wd[UsePage].inp_mode==1 ) return 0; /* 早々とリターン(表示のみの場合) */

    mouseposset( wd[UsePage].x+8-16,wd[UsePage].y+16-32 );
    mouseon();
    rewind( stdin );

    while( 1 )
    {
        if ( mode != FW_OPEN && mode != FW_REMAIN )
        {
            r1= ul+wd[UsePage].l;
            l2= wd[UsePage].l;
        }
        mouseput();
        if ( (c=( rc==0 ? wdkey_in(OK,&wd[UsePage],&wd[UsePage].l) : rc ))==0 )
        {
            if ( strlen( wddata[r1] )==0 && wd[UsePage].inp_mode==0
                && (wd[UsePage].inp_line==r1 || wd[UsePage].inp_line==1) ) c=CR;
            else if ( rc==0 && wd[UsePage].cursor_on==1 )
            {
                if ( mode==FW_OPEN || mode==FW_REMAIN )
                {
                    fprintf( REU,r1,&wd[UsePage],wddata );
                    } else print( REU,r1,&wd[UsePage],wddata );
            }
        }
    }
    rc=0;
    if ( mode == FW_OPEN || mode == FW_REMAIN )
    {
        if ( MS_LB )
        {
            for( i=ul;i<ul+wd[UsePage].inp_line+(wd[UsePage].wy-2);++i )
            {
                if ( (int)(MS_X/8+1)>Pos[i].x
                    && (int)(MS_X/8+1)<Pos[i].x+(int)((wd[UsePage].wx-4)/wd[UsePage].l

```

```

np_line)
    && (int)(MS_Y/16+1)==Pos[i].y ) break;
    }
    if ( r1==i ) c=CR;
    else if ( i<ul+wd[UsePage].inp_line+(wd[UsePage].wy-2) )
    {
        fprintf( REC,r1,&wd[UsePage].wddata );
        r1=i;
        fprintf( REU,r1,&wd[UsePage].wddata );
        delay( 150 );
        continue;
    }
    MS_LB=0;
}
}
else{
    if ( wd[UsePage].l-12 ){ /* 現在のカーソル位置の設定 */
        r1= ul+wd[UsePage].l;
        if ( r1>wd[UsePage].cmin && r1<wd[UsePage].cmax ){
            l3=wd[UsePage].l;wd[UsePage].l=12;
            print( REC,ul+wd[UsePage].l,&wd[UsePage].wddata );
            wd[UsePage].l=l3;
            delay( 120 );
            continue;
        }
        wd[UsePage].l=12;
    }
}
switch( c )
{
    case LEFT :if ( mode==FW_OPEN :: mode==FW_REMAIN )
    {
        fprintf( REC,r1,&wd[UsePage].wddata );
        if ( r1 > ul ) r1--;
        else if ( ul>0 ){
            ul=wd[UsePage].inp_line;
            r1--;
            fdspdata( ul,&wd[UsePage].wddata );
        }
        fprintf( REU,r1,&wd[UsePage].wddata );
        break;
    }
    if ( UsePage==0 :: MAXPAGE==1 ) break; /* クローズ処理 */
    puttext( wd_para[UseNum].x1,wd_para[UseNum].y1,
            wd_para[UseNum].x2,wd_para[UseNum].y2,
            wd_para[UseNum].adr );
    free( wd_para[UseNum].adr );
    UseNum--;
    UsePage--;
    secon[Sec]--;
    ul=( MAXPAGE > 1 ? wd[UsePage].cmin : 0 );
    break;
    case RIGHT:if ( mode==FW_OPEN :: mode==FW_REMAIN )
    {
        fprintf( REC,r1,&wd[UsePage].wddata );
        if ( r1 < ul+wd[UsePage].inp_line+(wd[UsePage].wy-2)-1 ) r1++;
        else if ( r1+1 < wd[UsePage].cmax-1 ){
            ul+=wd[UsePage].inp_line;
            r1++;
            fdspdata( ul,&wd[UsePage].wddata );
        }
        fprintf( REU,r1,&wd[UsePage].wddata );
        break;
    }
    if ( UsePage==MAXPAGE-1 :: MAXPAGE==1 ) break;
    print( REC,r1,&wd[UsePage].wddata );
    UsePage++;
    UseNum++;
    secon[Sec]++;
    getmem( wd_para,UseNum,wd[UsePage].x-1
            ,wd[UsePage].y
            ,wd[UsePage].x+wd[UsePage].wx-1+1
            ,wd[UsePage].y+wd[UsePage].wy-1 );
    gettex1( wd_para[UseNum].x1,wd_para[UseNum].y1,

```

```

        wd_para[UseNum].x2,wd_para[UseNum].y2,
        wd_para[UseNum].adr );
    flame( mode,MAXPAGE,UsePage,&wd[UsePage] );
    ul=( MAXPAGE > 1 ? wd[UsePage].cmin : 0 );
    dspdata( ul,0,&wd[UsePage],wddata );
    break;
case DOWN: if ( mode == FW_OPEN :: mode == FW_REMAIN )
    {
        fprintf( REC,r1,&wd[UsePage],wddata );
        if ( r1+wd[UsePage].inp_line <= ul+wd[UsePage].inp_line+(wd[UsePage]
        .wy-2)-1 ) r1+=wd[UsePage].inp_line;
        else {
            if ( r1+wd[UsePage].inp_line < wd[UsePage].cmax-1 ){
                ul+=wd[UsePage].inp_line;
                r1+=wd[UsePage].inp_line;
                fdspdata( ul,&wd[UsePage],wddata );
            }
            fprintf( REU,r1,&wd[UsePage],wddata );
            break;
        }
        if ( wd[UsePage].cmax <= 1 ) break;
        dtm=0;
        if ( wd[UsePage].l==wd[UsePage].wy-3
            && ul < wd[UsePage].cmax-(wd[UsePage].wy-2) )
        {
            /* スクロール処理 */
            ul++;
            print( REC,r1,&wd[UsePage],wddata );
            dspdata( ul,c,&wd[UsePage],wddata );
            dtm=20;
        }
        else if ( wd[UsePage].l < wd[UsePage].wy-3 )
        {
            /* スクロール無し */
            print( REC,r1,&wd[UsePage],wddata );
            wd[UsePage].l++;
            dtm=50;
        }
    }
    if ( dtm==0 && MAXPAGE==1 )
    {
        print( REC,r1,&wd[UsePage],wddata );
        wd[UsePage].l=wd[UsePage].cmin;
        ul=0;
    }
    if ( wd[UsePage].inp_line != -1 ) print( REU,ul+wd[UsePage].l,&wd[UsePa
    ge],wddata );
    delay( dtm );
    break;
case UP : if ( mode == FW_OPEN :: mode == FW_REMAIN )
    {
        fprintf( REC,r1,&wd[UsePage],wddata );
        if ( r1-wd[UsePage].inp_line >= ul ) r1-=wd[UsePage].inp_line;
        else {
            if ( ul>0 ){
                ul-=wd[UsePage].inp_line;
                r1-=wd[UsePage].inp_line;
                fdspdata( ul,&wd[UsePage],wddata );
            }
            fprintf( REU,r1,&wd[UsePage],wddata );
            break;
        }
        if ( wd[UsePage].cmax <= 1 ) break;
        dtm=0;
        if ( wd[UsePage].l==0 && ul > wd[UsePage].cmin )
        {
            /* スクロール処理 */
            ul--;
            print( REC,r1,&wd[UsePage],wddata );
            dspdata( ul,c,&wd[UsePage],wddata );
            dtm=20;
        }
        else if ( r1 > wd[UsePage].cmin )
        {
            /* スクロール無し */
            print( REC,r1,&wd[UsePage],wddata );
            wd[UsePage].l--;
            dtm=50;
        }
    }

```

```

if ( dtm==0 && MAXPAGE==1 )
{
    print( REC,r1,&wd[UsePage],wddata );
    wd[UsePage].l=wd[UsePage].wy-3;
    ul=wd[UsePage].cmax-(wd[UsePage].wy-2);
}
if ( wd[UsePage].inp_line != -1 ) print( REU,ul+wd[UsePage].l,&wd[UsePage].l,wddata );
delay( dtm );
break;
case CR : if ( wd[UsePage].inp_mode !=0 :: mode==FW_OPEN :: mode==FW_REMAIN
:: (r1 != wd[UsePage].inp_line && wd[UsePage].inp_line != -1) )
{
    mouseoff();
    flame( WD_CLOSEU,MAXPAGE,UsePage,&wd[UsePage] );
    return( r1 );
}
if (r1==wd[UsePage].inp_line :: wd[UsePage].inp_line== -1){
    mouseoff();
    l4=wd[UsePage].l;
    if ( wd[UsePage].inp_line== -1 ) /* どの行でも入力モード */
    {
        wd[UsePage].inp_line=r1;
        rc=input( r1,&wd[UsePage],wddata );
        wd[UsePage].inp_line= -1;
    } else rc=input( r1,&wd[UsePage],wddata );
    l5=wd[UsePage].l;
    wd[UsePage].l=l4;
    print( REC,ul+l4,&wd[UsePage],wddata );
    wd[UsePage].l=l5;
    if ( rc==CR ) {
        flame( WD_CLOSEU,MAXPAGE,UsePage,&wd[UsePage] );
        return( r1 ); /* 決定 */
    }
    mouseon();
}
break;
case ESC : delay( 100 );mouseoff(); /* ウィンドウ関数脱出 */
MS_LB=MS_RB=0;
flame( WD_CLOSEU,MAXPAGE,UsePage,&wd[UsePage] );
return( -1 );
case HOME: x=MS_X/8+1;y=MS_Y/16+1;
UpPage= -1;
for( i=0;i<UsePage;++i ) /* どのウィンドウにカーソルがあるか */
{
    if ( wd[i].x <= x && wd[i].x+wd[i].wx >= x
&& wd[i].y <= y && wd[i].y+wd[i].wy >= y ) UpPage=i;
}
if ( UpPage>=0 && UsePage != UpPage )
{
    for( i=UsePage ; i>=UpPage ; --i )
    {
        puttext( wd_para[UseNum].x1,wd_para[UseNum].y1,
wd_para[UseNum].x2,wd_para[UseNum].y2,
wd_para[UseNum].adr );
        UseNum--;
    }
    tmp=wd[UpPage];
    wd[UpPage]=wd[UsePage];
    wd[UsePage]=tmp;
    for( i=UpPage;i<=UsePage;++i )
    {
        UseNum++;
        wd_para[UseNum].x1=wd[i].x-1;
        wd_para[UseNum].y1=wd[i].y;
        wd_para[UseNum].x2=wd[i].x+wd[i].wx-1+1;
        wd_para[UseNum].y2=wd[i].y+wd[i].wy-1;
        gettext( wd_para[UseNum].x1,wd_para[UseNum].y1,
wd_para[UseNum].x2,wd_para[UseNum].y2,
wd_para[UseNum].adr );
        flame( mode,MAXPAGE,i,&wd[i] );
        ul=( MAXPAGE > 1 ? wd[i].cmin : 0 );
        dspdata( ul,0,&wd[i],wddata );
    }
}

```

```

        break;      /* 移動はしなくて良い */
    }
    if ( x+wd[UsePage].wx < 79 ) mx=x+2;
    else mx=79-wd[UsePage].wx;
    if ( y+wd[UsePage].wy < 25 ) my=y+1;
    else my=24-wd[UsePage].wy;
    if ( mx != wd[UsePage].x || my != wd[UsePage].y )
    {
        puttext( wd_para[UseNum].x1,wd_para[UseNum].y1,
                  wd_para[UseNum].x2,wd_para[UseNum].y2,
                  wd_para[UseNum].adr );
        wd[UsePage].x=mx;
        wd[UsePage].y=my;
        wd_para[UseNum].x1=wd[UsePage].x-1;
        wd_para[UseNum].y1=wd[UsePage].y;
        wd_para[UseNum].x2=wd[UsePage].x+wd[UsePage].wx-1+1;
        wd_para[UseNum].y2=wd[UsePage].y+wd[UsePage].wy-1;
        gettext( wd_para[UseNum].x1,wd_para[UseNum].y1,
                  wd_para[UseNum].x2,wd_para[UseNum].y2,
                  wd_para[UseNum].adr );
        flame( mode,MAXPAGE,UsePage,&wd[UsePage] );
        ul=( MAXPAGE > 1 ? wd[UsePage].cmin : 0 );
        if ( mode == FW_OPEN || mode == FW_REMAIN )
        {
            fdspdata( ul,&wd[UsePage],wddata );
        } else dspdata( ul,0,&wd[UsePage],wddata );
    }
    break;
default : break;
}

}

}

.....
*                  飛火野用ウィンドウ関数群
*                  .....
void allclose( void )
{
    while( twindow( WD_CLOSE ) != -1 );
}
void getmem( struct WD_PARA wd_para[],int i,int x1,int y1,int x2,int y2 )
{
    unsigned int len;
    wd_para[i].x1=x1;
    wd_para[i].y1=y1;
    wd_para[i].x2=x2;
    wd_para[i].y2=y2;
    len= ( x2 - x1 + 1 ) * ( y2 - y1 + 1 ) * 4;
    if ( ( wd_para[i].adr=malloc( sizeof(char) * len ) ) == NULL ) {
        clrscr();
        printf( "malloc によるメモリーの獲得に失敗しました。%n" );
        exit(1);
    }
    setmem( wd_para[i].adr,len,0 );
}

int wdkey_in( int mode,WINDOW *wd,int *l )
{
    int x=MS_X/8+1,y=MS_Y/16+1,c=0,k,c2,flg=0;

    k=key_in();
    c2=(k>>8)&0x00ff; /* let c=UP,DOWN,LEFT,RIGHT */
    if ( mode==NO )
    {
        if ( c2==UP ) c=0x000b;
        if ( c2==DOWN ) c=0x000a;
    }
    if ( c==0 && k&0x00ff ) c=k&0x00ff;
    if ( c==0 && c2 ) c=c2;
    if ( c==0 )
    {
        if ( MS_RB || MS_LB && wd->cursor_on==0 ) c=ESC;
        else

```

```

if ( x>wd->x && x<wd->x+wd->wx && y >=wd->y && y<=wd->y+wd->wy-1 )
( /* ウィンドウ領域の中 */
  if ( MS_LB ) /* 左ボタンを押した */
  {
    if ( (x==wd->x || x==wd->x+1) && y==wd->y) && wd->button!=2 ) c=ESC;
    if ( wd->button == 1 )
    {
      if ( x==wd->x+wd->wx-2 || x==wd->x+wd->wx-1 ) /* UP,DOWN area */
      {
        if ( y==wd->y+wd->wy-2 ) {
          c=DOWN;flg=1;
        }
        if ( y==wd->y+1 ) {
          c=UP;flg=1;
        }
      }
      if ( y==wd->y+wd->wy-1 ) /* LEFT,RIGHT area */
      {
        if ( x==wd->x+2 || x==wd->x+3 ) {
          c=LEFT;flg=0;
        }
        if ( x==wd->x+wd->wx-4 || x==wd->x+wd->wx-3 )
        {
          c=RIGHT;flg=0;
        }
      }
    }
    if ( y<wd->y+wd->wy-1 && y>wd->y && x>wd->x+1 && x<wd->x+wd->wx-2 )
    {
      if ( y-wd->y-1==wd->y )
      {
        c=CR; /* 決定 */
        delay( 100 );
      }
      else if ( c==0 ) c=ESC;
      if ( wd->cursor_on==1 ) /* 現在のカーソル位置セット */
      {
        *l = y-wd->y-1;
      }
    }
  }
} else if ( MS_LB ) { flg=1;c=HOME; } /* ウィンドウの移動 */
}
if ( MS_LB && flg !=1 ){
  while( MS_LB ) mouseput();
  MS_LB=1;
}

return c;
}

char input( int rl,WINDOW *wd,char wddata[][DATA_WD] )
{
  static char *ptr,oldstr[DATA_WD],str[DATA_WD];
  int c,al,ma,i,i2,numtmp,tmp,chflg=0,x,y,l2;

  tmp=(wd->lineno_on==0 ? NUM_MERGIN : 0 );
  numtmp=( wd->lineno_on==1 ? NUM_MERGIN : 0 );
  strcpy( str,wddata[rl] );
  strcpy( oldstr,str );
  rewind( stdin );
  mouseon();
  while( 1 )
  {
    textcursor( NODISP_CURSOR );
    i2=strlen( str );
    for(i=i2;i<(wd->wx-7+tmp-1);++i) str[i]=' ';
    gotoxy( wd->x+MERGIN-NUM_MERGIN+numtmp,wd->y+1+wd->inp_line );cprintf(str);
    for(i=i2;i<(wd->wx-7+tmp-1);++i) str[i]='\0';
    gotoxy( wd->x+MERGIN-NUM_MERGIN+numtmp+strlen(str),wd->y+1+wd->inp_line );

    if ( strlen(str)>(wd->wx-7+tmp-1)-1 ) textcursor( NODISP_CURSOR );
    else textcursor( DISP_CURSOR );

    l2=wd->l;
    MS_LB=MS_RB=0;
  }
}

```



```

do{
    mouseput();
    c=wdkey_in(NO,wd,&wd->l);
} while( c==0 );
if ( c==0x000a ;; c==0x000b ;; c==ESC ;; c==HOME ) {
    if ( c==ESC )
    {
        delay( 100 );
        if ( wd->l-12 ) c=0;
        else if ( MS_RB ){ setmem(str,DATA_WD,0); /* 内容のクリア */
            continue;
        }
    }
    jstrncpy(wddata[r1],str,jstrlen(str));
    break;
}

switch( c )
{
    case BS:al=jstrlen(str);ma=strlen(str);
        if ( ma>0 ){
            ptr=jstradv(str,al-1);
            *ptr='%0';
            if( ma-strlen(str) == 2 ) *(ptr+1)='%0';
        }
        break;
    case CR:jstrncpy( wddata[r1],str,jstrlen(str) );
        goto out;
    default: if ( strlen(str)<(wd->wx-7+tmp-1) )
        {
            al=jstrlen(str);
            ptr=jstradv(str,al);
            *ptr=c;
            if ( al == jstrlen(str) ){
                if ( strlen(str)==(wd->wx-7+tmp-1) ) {
                    putchar( 7 );
                    *ptr='%0';
                    rewind( stdin );
                }
                else ( c=getch();
                    *(ptr+1)=c;
                )
            }
            ) else rewind( stdin );
            chflg=1;
            break;
        }
}

out:
textcursor( NODISP_CURSOR );
mouseoff();
if ( c==CR && chflg ) c=0;
if ( c==0x000a ) c=DOWN;
if ( c==0x000b ) c=UP;
MS_LB=MS_RB=0;
return c;
}

void flame( int mode,int mxpg,int uspg,WINDOW *wd )
{
    int i,i2,x;

    if ( mode==WD_CLOSEU ;; mode== -1 )
    {
        if ( mxpg==1 && wd->button!=2 ){
            textreverse( REVERSE );
            textcolor( wd->col1 );
            gotoxy( wd->x,wd->y );
            if ( mode==-1 ){
                textcolor( T_BLUE );
                cprintf(" ■ ");
            } else cprintf(" ");
            textreverse( NOREVERSE );
            textcolor( T_WHITE );
        }
        return;
    }
}

```

```

)
window( wd->x,wd->y,wd->x+wd->wx-1,wd->y+wd->wy-1 );
clrscr();
window( 1,1,80,25 );
textcolor( wd->coll );
if ( wd->button == 2 )
{
    textreverse( NOREVERSE );
    gotoxy( wd->x,wd->y );
    cprintf(" P");
} else {
    textreverse( REVERSE );
    textcolor( T_BLUE );
    gotoxy( wd->x,wd->y );
    cprintf(" ■");
    textcolor( wd->coll );
    if ( wd->button==0 ) textreverse( NOREVERSE );
}
gotoxy( wd->x+wd->wx-1-1 , wd->y );
if ( wd->button == 1 ) cprintf(" "); else cprintf("┐");
gotoxy( wd->x , wd->y+wd->wy-1 );
if ( wd->button == 1 ) cprintf(" "); else cprintf("┌");
gotoxy( wd->x+wd->wx-1-1 , wd->y+wd->wy-1 );
if ( wd->button == 1 ) cprintf(" "); else cprintf("└");
for( i=wd->x+2;i<wd->x+wd->wx-1-1;i+=2 )
{
    if ( wd->button !=1 ){
        gotoxy( i,wd->y );cprintf("—");
    }
    gotoxy( i,wd->y+wd->wy-1 );cprintf("—");
}
for( i=wd->y+1;i<wd->y+wd->wy-1;++i )
{
    gotoxy( wd->x,i );
    if ( wd->button == 1 ) cprintf(" "); else cprintf(" |");
    gotoxy( wd->x+wd->wx-1-1,i );cprintf(" |");
}
if ( wd->button==1 ){
    gotoxy( wd->x+wd->wx-2,wd->y+1 );cprintf("▲");
    gotoxy( wd->x+wd->wx-2,wd->y+wd->wy-1-1 );cprintf("▼");
    if ( mxpg>1 && uspg>0 :: mode==FW_OPEN :: mode==FW_REMAIN ){
        textcolor( T_RED );
        gotoxy( wd->x+2 , wd->y+wd->wy-1 );
        cprintf("⌞");
        textcolor( wd->coll );
    }
    if ( mxpg>1 && uspg<mxpg-1 :: mode==FW_OPEN :: mode==FW_REMAIN ){
        textcolor( T_RED );
        gotoxy( wd->x+wd->wx-4,wd->y+wd->wy-1 );
        cprintf("⌟");
        textcolor( wd->coll );
    }
    textcolor( wd->col2 );
    for(i=wd->x+2;i<wd->x+wd->wx-2;++i){
        gotoxy( i,wd->y );cprintf(" ");
    }
}
textcolor( wd->col2 );
x=wd->x+(wd->wx)/2-strlen(wd->title)/2;
gotoxy( x,wd->y );
textreverse( REVERSE );
cprintf(wd->title);
textcolor( T_WHITE );
textreverse( NOREVERSE );
return;
}

void dspdata( int ul,int c,WINDOW *wd,char wddata[][DATA_WD] )
{
    int i,i2;
    window( wd->x+2,wd->y+1,
            wd->x+wd->wx-3,wd->y+wd->wy-2 );
    if ( c==UP ) { gotoxy( 1,1 );insline(); }
    if ( c==DOWN ) { gotoxy( 1,1 );delline(); }
}

```

```

if ( c==1 ) clrscr();
window( 1,1,80,25 );
for(i=ul;i<ul+wd->wy-2;++i)
{
    if ( c==UP && i-ul >0 || c==DOWN && i-ul <10 ) continue;
    gotoxy( wd->x+2,wd->y+1+i-ul );
    textcolor( wd->col1 );
    if ( wd->lineno_on==1 ) cprintf("%3d:",i+1 );
    textcolor( wd->col3 );
    cprintf("%-s",wddata[ i ] );
    textcolor( T_WHITE );
}

void print( int mode,int pl,WINDOW *wd,char wddata[][DATA_WD] )
{
    int i,len,tmp;
    tmp=(wd->lineno_on==0 ? NUM_MERGIN : 0 );
    len=strlen( wddata[pl] );
    for ( i=len;i<wd->wx-7+tmp-1;++i) wddata[pl][i]=' ';
    textcolor( wd->col1 );
    if ( mode==REU ) textreverse( REVERSE );
    gotoxy( wd->x+2,wd->y+wd->l+1 );
    if ( wd->lineno_on==1 ) cprintf("%3d:",pl+1);
    textcolor( wd->col3 );
    cprintf("%-s",wddata[pl] );
    textreverse( NOREVERSE );
    textcolor( T_WHITE );
    for ( i=len;i<wd->wx-7+tmp-1;++i) wddata[pl][i]='%0';
}

void fdspdata( int ul,WINDOW *wd,char wddata[][DATA_WD] )
{
    int i,i2,dx,dy;
    char prbuf[80];
    window( wd->x+2,wd->y+1,
            wd->x+wd->wx-3,wd->y+wd->wy-2 );
    clrscr();
    window( 1,1,80,25 );
    dx=dy=0;
    for( i=ul;i<ul+wd->inp_line-(wd->wy-2);++i )
    {
        Pos[i].x=wd->x+2+dx*(wd->wx-4)/wd->inp_line;
        Pos[i].y=wd->y+1+dy;
        gotoxy( Pos[i].x,Pos[i].y );
        setmem( prbuf,80,0 );
        for( i2=0 ; i2<(wd->wx-4)/wd->inp_line-2 ; i2++ )
            prbuf[i2]=( wddata[i][i2] == '%0' ? ' ' : wddata[i][i2] );
        textcolor( wd->col3 );
        cprintf("%-s",prbuf);
        dx++;
        if ( dx==wd->inp_line ) { dx=0;dy++; }
    }
    textcolor( T_WHITE );
}

void fprint( int mode,int pl,WINDOW *wd,char wddata[][DATA_WD] )
{
    int i;
    char prbuf[80];
    setmem( prbuf,80,0 );
    for( i=0 ; i<(wd->wx-4)/wd->inp_line-2 ; i++ )
        prbuf[i]=( wddata[pl][i] == '%0' ? ' ' : wddata[pl][i] );
    if ( mode==REU ) textreverse( REVERSE );
    gotoxy( Pos[pl].x,Pos[pl].y );
    textcolor( wd->col3 );
    cprintf("%-s",prbuf );
    textreverse( NOREVERSE );
    textcolor( T_WHITE );
}

.....
・      アイコンメニューによる機能選択      ・
.....
int cmd_inp()
{

```

```

int c,x,y,i,i2,flg=1,funcflg=0;
char c1,c2;

MS_LB=MS_RB=0;
x=wherex();      /* 初期状態の確保 */
y=wherey();
textcursor( NODISP_CURSOR );
dspconnectname();

while( flg )
{
    IPX= IPX2;
    IPY= IPY2;
    icon_disp();
    mouseon();
    funcflg=0;
    do {
        c=key_in();
        if ( (c>>8)&0x00ff ){
            mouseoff();
            switch( (c>>8)&0x00ff )
            {
                case UP:MS_Y-=20;break;
                case DOWN:MS_Y+=20;break;
                case LEFT:MS_X-=20;break;
                case RIGHT:MS_X+=20;break;
                case FUN1:k=3;funcflg=1;break;
                case FUN2:k=4;funcflg=1;break;
                case FUN3:k=5;funcflg=1;break;
                case FUN4:k=6;funcflg=1;break;
                case FUN5:k=7;funcflg=1;break;
                case FUN6:k=2;funcflg=1;break;
                case FUN7:k=8;funcflg=1;break;
                default:break;
            }
            mouseposset( MS_X,MS_Y );
            mouseon();
        }
        mouseput();
        if ( (char)(c&0x00ff)==CR ) MS_LB=1;
        if ( (char)(c&0x00ff)==ESC ) MS_RB=1;
    } while( MS_LB==0 && MS_RB==0 && funcflg==0 );
    mouseoff();
    if ( MS_LB || funcflg )
    {
        for(i=0;i<CON_N0;++i)
        {
            if ( [PX+ip[i].x <= MS_X && IPY+ip[i].y <= MS_Y
                && [PX+ip[i].x+39 >= MS_X && IPY+ip[i].y+39 >= MS_Y
                && MS_LB || funcflg ]
            {
                /* メニューの選択をした */
                if ( funcflg ) i=k;
                for( i2=0;i2<2;++i2 )
                {
                    putimage( (int)(IPX/8)*8+ip[i].x,(int)(IPY/16)*16+ip[i].y,
(void far *)buf[ip[i].num],COPY_PUT );
                    delay(150);
                    putimage( (int)(IPX/8)*8+ip[i1].x,(int)(IPY/16)*16+ip[i1].y,
(void far *)buf[8],COPY_PUT );
                    delay(100);
                }
                k=ip[i].com;
                icon_close();
                flg=0;
                break;
            }
        }
    }
    if ( MS_LB && flg || MS_RB ){ /* 呼出し取消し */
        if ( MS_X<ICON_XMAX ) IPX2= MS_X+30; else IPX2=ICON_XMAX+30;
        if ( MS_Y<ICON_YMAX ) IPY2= MS_Y+30; else IPY2=ICON_YMAX+30;
        icon_close();
    }
}

```

```

underlinedisp();
window( 1,1,80,24 );
gotoxy( x,y );
MS_LB=MS_RB=0;
return ( k );
}

/.....
・      アイコンデータの読み込み      ・
...../
void icset( void )
{
    int i,i2,i3,c,imc=0,flg=0;
    FILE *fp,*mouse,-icon;
    static char path_use[10],path2[20],path[80],ICN[80],MOU[80];

    cleardevice();
    sprintf(ICN,"%s%stobihino.icn",ENU);
    sprintf(MOU,"%s%stobihino.mou",ENU);
    if ( (icon=fopen(ICN,"rb")) == 0 ) /* 登録済み? */
    {
        fclose( icon );
        textcolor( T_YELLOW );
        gotoxy( 1,1 );cprintf("準備中です。");
        setactivepage(1);
        cleardevice();
        for ( i3=0 ; i3<30 ; ++i3 ){
            sprintf(path,"%s%$",ENU);
            itoa( i3,path_use,10 );
            strcpy( path2,path );
            strcat( path2,path_use );
            if ( (fp=fopen(path2,"r"))!=NULL ){
                for(i=0;i<40;i++){
                    for(i2=0;i2<40;i2++){
                        do {
                            c=fgetc( fp );
                        } while( c!='\n' );
                        putpixel( i2,i,col_tbl(c-'0') );
                    }
                }
                fclose(fp);
                if ( i3==MS_PAT_NO ){
                    getimage( 0,0,19,19,(void far *)ms_im );
                    flg=1;
                }
                else (
                    if ( imc<=ICON_MAX_NO ){
                        getimage( 0,0,39,39,(void far *)buf[imc] );
                        imc++;
                    }
                )
            }
            if ( imc-1 == ICON_MAX_NO && flg==1 ) break;
        }
    }
    icon=fopen(ICN,"wb"); /* アイコンデータの書き込み */
    fwrite( buf,1000,ICON_MAX_NO+1,icon );
    fclose( icon );
    mouse=fopen(MOU,"wb");
    fwrite( ms_im,1000,1,mouse );
    fclose(mouse);
} else { /* アイコンデータの読み込み */
    fread( buf,1000,ICON_MAX_NO+1,icon );
    fclose( icon );
    setmem( ms_im,1000,0 );
    mouse=fopen( MOU,"rb" );
    fread( ms_im,1000,1,mouse );
    fclose( mouse );
}
setvisualpage(0);
setactivepage(0);
return;
}

/.....
・      アイコン表示関係関数      ・
...../

```

```

...../
void icon_disp( void )
{
    int i,tmp;
    gettext( (int)((PX/8)+1 -1,      /* 左と右に1つつマージンを入れる */
              (int)((PY/16)+1,
              (int)((PX/8)+1 +14 +1,
              (int)((PY/16)+1 +7,
              disp_buf );
    window( (int)((PX/8)+1,(int)((PY/16)+1,(int)((PX/8)+1+14,(int)((PY/16)+1+7 );
    clrscr();
    window( 1,1,80,25 );
    IPX= (int)((PX/8)+8;
    IPY= (int)((PY/16)+16;
    for( i=0;i<COM_NO;++i ) {
        tmp=ip[i].num;
        putimage( [PX+ip[i].x,[PY+ip[i].y,(void far *)buf[tmp],COPY_PUT );
    }
}

void icon_close( void )
{
    int i;
    IPX= (int)((PX/8)+8;
    IPY= (int)((PY/16)+16;
    for( i=0;i<COM_NO;++i )
    {
        putimage( [PX+ip[i].x,[PY+ip[i].y,(void far *)buf[16],COPY_PUT );
    }
    puttext( (int)((PX/8)+1 -1,
              (int)((PY/16)+1,
              (int)((PX/8)+1 +14+1,
              (int)((PY/16)+1 +7,disp_buf );
}

```

```

/*
.....
MSDOS関係ハンドラ
.....
*/

```

キーからのデータ入力
(ファンクションキーも含む)

```

/*
int key_in()
{
    unsigned int    c;
    char            ch;

    ch=(char)(0x00ff&bdos(0x06,0xff,0));
    if(ch==0)        c=0;
    else
    if(ch==0xff){
        ch=(char)(0x00ff&bdos(0x07,0x00,0));
        switch(ch)
        {
            case 'H':    c=0x6200;    break; /* f.1 */
            case 'I':    c=0x6300;    break; /* f.2 */
            case 'J':    c=0x6400;    break; /* f.3 */
            case 'K':    c=0x6500;    break; /* f.4 */
            case 'L':    c=0x6600;    break; /* f.5 */
            case 'M':    c=0x6700;    break; /* f.6 */
            case 'N':    c=0x6800;    break; /* f.7 */
            case 'A':    c=0x6a00;    break; /* f.9 */
            case 'B':    c=0x6b00;    break; /* f.10 */
            case 'C':    c=0x3c00;    break; /* ROLL UP */
            case 'D':    c=0x3700;    break; /* ROLL DOWN */
            case 'E':    c=0x3800;    break; /* INS */
            case 'F':    c=0x3e00;    break; /* HOME */
            case 'G':    c=0x3f00;    break; /* HELP */
            default:      c=(0x00ff&ch); break;
        }
    }
    else if(ch==0xc)    c=0x3c00;
}

```

```

        else if(ch==0x1d)    c=0x3b00;
        else if(ch==0x1e)    c=0x3a00;
        else if(ch==0x1f)    c=0x3d00;
        else if(ch==0x7f)    c=0x3900;
        else c=(0x00ff&ch);
        return c;
    }
    /*
    -----
    キーバッファのクリア
    -----
    */
    void    key_clr()
    {
        union    REGS    regs;

        regs.h.ah=0x00;
        int86(0x10,&regs,&regs);
    }
    /*
    -----
    キーからの直接入力
    -----
    */
    int key_getc()
    {
        union    REGS    regs;

        regs.h.ah=0x05;
        int86(0x10,&regs,&regs);
        if(regs.h.bh==0)    return 0;
        if((regs.h.ah==0x0a && regs.h.ah<=0x0b)::
            (regs.h.ah==0x36 && regs.h.ah<=0x3f))
            return (0xff00&regs.x.ax);
        else    return (0x00ff&regs.x.ax);
    }
    /*
    -----
    キーからの直接ステータス
    -----
    */
    int key_gets()
    {
        union    REGS    regs;

        regs.h.ah=0x01;
        int86(0x10,&regs,&regs);
        if(regs.h.bh==0)    return 0;
        else    return 0xff;
    }
    /*
    .....
    プリンタへの出力
    .....
    */
    void    printer(char data)
    {
        int x;

        x=data;
        bdos(0x05,x,0);
    }
    /*
    .....
    *          ファンクションキーやカーソルキー関係          *
    .....
    */
    /*
    -----
    現在のファンクションキーの取得
    -----
    */
    void    getfunckey()
    {

```

```

union REGS regs;
struct SREGS sregs;

setmem(&funkey[0], 386, 0);
regs.h.cl=0x0c;
regs.x.ax=0;
regs.x.dx=FP_OFF(&funkey[0]);
sregs.ds=FP_SEG(&funkey[0]);
int86x(0xdc, &regs, &regs, &sregs);
)
/*
-----
                ファンクションキーの返却
-----
*/
void putfunkey()
{
    union REGS regs;
    struct SREGS sregs;

    regs.h.cl=0x0d;
    regs.x.ax=0;
    regs.x.dx=FP_OFF(&funkey[0]);
    sregs.ds=FP_SEG(&funkey[0]);
    int86x(0xdc, &regs, &regs, &sregs);
}
/*
-----
                ファンクションキーの設定
-----
*/
#define FUNCC 0xff

void setfunkey()
{
    union REGS regs;
    struct SREGS sregs;

    setmem(&funkey[1], 386, 0);
    sprintf(&funkey[1].func[0][0], "%cH", FUNCC); /* f-1 */
    sprintf(&funkey[1].func[1][0], "%cI", FUNCC); /* f-2 */
    sprintf(&funkey[1].func[2][0], "%cJ", FUNCC); /* f-3 */
    sprintf(&funkey[1].func[3][0], "%cK", FUNCC); /* f-4 */
    sprintf(&funkey[1].func[4][0], "%cL", FUNCC); /* f-5 */
    sprintf(&funkey[1].func[5][0], "%cM", FUNCC); /* f-6 */
    sprintf(&funkey[1].func[6][0], "%cN", FUNCC); /* f-7 */
    sprintf(&funkey[1].func[8][0], "%cA", FUNCC); /* f-9 */
    sprintf(&funkey[1].func[9][0], "%cB", FUNCC); /* f-10 */
    sprintf(&funkey[1].curs[0][0], "%cC", FUNCC); /* roll up */
    sprintf(&funkey[1].curs[1][0], "%cD", FUNCC); /* roll down */
    sprintf(&funkey[1].curs[2][0], "%cE", FUNCC); /* INS */
    sprintf(&funkey[1].curs[3][0], "%c", 0x7f); /* DEL */
    sprintf(&funkey[1].curs[4][0], "%c", 0x1e); /* UP */
    sprintf(&funkey[1].curs[5][0], "%c", 0x1d); /* LEFT */
    sprintf(&funkey[1].curs[6][0], "%c", 0x1c); /* RIGHT */
    sprintf(&funkey[1].curs[7][0], "%c", 0x1f); /* DOWN */
    sprintf(&funkey[1].curs[8][0], "%cF", FUNCC); /* HOME CLR */
    sprintf(&funkey[1].curs[9][0], "%cG", FUNCC); /* HELP */
    regs.h.cl=0x0d;
    regs.x.ax=0;
    regs.x.dx=FP_OFF(&funkey[1]);
    sregs.ds=FP_SEG(&funkey[1]);
    int86x(0xdc, &regs, &regs, &sregs);
}
/*
-----
                マウスハンドラー
-----
*/
/*
-----
                マウスの初期化
-----
*/

```



```

int mouseinit( void )
{
    union   REGS    regs;
    struct  SREGS    segregs;

    regs.x.ax=0;
    int86(0x33,&regs,&regs);
    if(regs.x.ax==0) return -1;
    mouseclr();
    mousosense(2,2);
    regs.x.ax=0x09;
    regs.x.bx=0;
    regs.x.cx=0x01;
    int86(0x33,&regs,&regs);
}
/*
-----
マウスの感度設定
-----
*/
void      mousosense(int mx,int my)  /* マウス感度の設定 */
{
    union   REGS    regs;

    regs.x.ax=15;
    regs.x.cx=mx;
    regs.x.dx=my;
    int86(0x33,&regs,&regs);
}
/*
-----
マウスの状態の把握
-----
*/
int mousediff( void ) /* マウスの状態に変化があったか？ あった！なかった0 */
/* 最新の状態を同時にセットします。 */
{
    MS_X2=MS_X;
    MS_Y2=MS_Y;
    MS_LB2=MS_LB;
    MS_RB2=MS_RB;
    mouseinput();
    if ( MS_X == MS_X2 && MS_Y==MS_Y2
        && MS_LB == MS_LB2 && MS_RB == MS_RB2 ) return ( 0 );
    return ( 1 );
}
/*
-----
マウスカーソルの表示
-----
*/
void mouseput( void ) /* マウスカーソルの表示
/* mousediff() を使用していますので
/* MS_X,MS_X2,MS_Y,MS_Y2,MS_LB,MS_LB2
/* MS_RB,MS_RB2 の内容は壊れます。
{
    if ( mousediff()==0 ) return;
    putimage( MS_X2,MS_Y2,(void far *)ms_im,XOR_PUT );
    putimage( MS_X,MS_Y,(void far *)ms_im,XOR_PUT );
    return;
}
/*
-----
マウスカーソルの点灯
-----
*/
void      mouseon()
{
    putimage( MS_X,MS_Y,(void far *)ms_im,XOR_PUT );
}
/*
-----
マウスカーソルの消去
-----
*/

```

```

/*
void    mouseoff()
{
    putimage( MS_X,MS_Y,(void far *)ms_im,XOR_PUT );
}
/*
-----
マウスカーソルのクリア
-----
/*
void    mouseclr()
{
    union    REGS    regs;

    regs.x.ax=2;
    int86(0x33,&regs,&regs);
}
/*
-----
マウスカーソルの位置設定
-----
/*
void    mouseposset(int x,int y)
{
    union    REGS    regs;

    regs.x.ax=4;
    regs.x.cx=x;
    MS_X=x;
    regs.x.dx=y;
    MS_Y=y;
    int86(0x33,&regs,&regs);
}
/*
-----
マウスカーソルの領域設定
-----
/*
void    mouseareaset( void )
{
    union    REGS    regs;

    regs.x.ax=0x10;          /* DIRECTION: HORIZON */
    regs.x.cx=0;
    regs.x.dx=640-20;
    int86(0x33,&regs,&regs);
    regs.x.ax=0x11;          /* DIRECTION: VERTICAL */
    regs.x.cx=16;
    regs.x.dx=400-36;
    int86(0x33,&regs,&regs);
}
/*
-----
マウスの各種データの取得
-----
/*
int mouseinput( void )
{
    union    REGS    regs;

    regs.x.ax=3;
    int86(0x33,&regs,&regs);
    MS_LB = regs.x.ax; /* 左ボタン */
    MS_RB = regs.x.bx; /* 右ボタン */
    MS_X = regs.x.cx; /* X座標 */
    MS_Y = regs.x.dx; /* Y座標 */
    return YES;
}
/*
.....
S I O の ハ ン ド ラ
.....
/*
/*

```

S I O の初期化

```

/*
int sioinit()
{
    int ch,data,sts;
    COM_INIT cint;

    setmem(SioBuff,4000,0);
    clearEscBuff();

    if ( cp.ltype==1 ) return 0;
    ch=cp.sio;          /* get sio channel No. */
    if(ch==0)
    {
        outportb(0x32,0x87);
        outportb(0x32,0x40);    /* ch-0 reset */
    }
    else if(ch==1)
    {
        outportb(0xb3,0x87);
        outportb(0xb3,0x40);    /* ch-1 reset */
    }
    else if(ch==2)
    {
        outportb(0xbb,0x87);
        outportb(0xbb,0x40);    /* ch-2 reset */
    }
    if(cp.xonoff==0)    cint.cmd=1;    /* Xon/off */
    else                cint.cmd=0;
    cint.transfer=0x07-cp.boudrate;    /* boudrate */
    switch(cp.parity)    /* parity */
    {
        case 0:
            data=0;
            break;
        case 1:
            data=0x30;
            break;
        case 2:
            data=0x10;
            break;
        default:
            break;
    }
    switch(cp.stopbit)    /* stop bit */
    {
        case 0: data:=0x40; break;
        case 1: data:=0x80; break;
        case 2: data:=0xc0; break;
        default:    break;
    }
    if(cp.length==0)    data:=0x0e;
    else                data:=0x0a;
    cint.mode_inf=data;
    cint.cmd_inf=0x17;
    cint.recv_buf=SioBuff;
    cint.recv_siz=4000;
    cint.time_snd=2;    /* 1秒待つ unit 500ms */
    cint.time_rcv=2;    /* 1秒待つ unit 500ms */
    switch(ch)
    {
        case 0: bios98com_init(&cint);    return 0;
        case 1: sts=bios98com_init_ch2(&cint);    return sts;
        case 2: sts=bios98com_init_ch3(&cint);    return sts;
        default:    return -1;
    }
}
/*

```

S I O のデータ入力

```

/*

```

```

char    sioinput()
{
    int data;
    COM_INFO cinf;

    if ( cp.ltype==0 )
    {
        cinf.cmd=4;
        switch(cp.sio)
        {
            case 0:
                bios98com(&cinf);
                data=cinf.data;
                break;
            case 1:
                bios98com_ch2(&cinf);
                data=cinf.data;
                break;
            case 2:
                bios98com_ch3(&cinf);
                data=cinf.data;
                break;
            default:
                data=0;
                break;
        }
    } else data=bufman(RCU,0);

    return (char)data;
}
/*

```

S I O の デ ー タ 出 力

```

/*
void    siooutput(char data)
{
    COM_INFO cinf;

    if ( cp.ltype==0 )
    {
        cinf.cmd=3;
        cinf.data=data;
        switch(cp.sio)
        {
            case 0:
                bios98com(&cinf);
                break;
            case 1:
                bios98com_ch2(&cinf);
                break;
            case 2:
                bios98com_ch3(&cinf);
                break;
            default:
                break;
        }
    } else bufman(SND,data);
}
/*

```

S I O の デ ー タ 数 取 り 込 み

```

/*
int    siornum()
{
    int data;
    COM_INFO cinf;

    if ( cp.ltype==0 )
    {
        cinf.cmd=2;
        switch(cp.sio)
        {

```

```

        case 0:
            bios98com(&cinf);
            data=cinf.data_size;
            break;
        case 1:
            bios98com_ch2(&cinf);
            data=cinf.data_size;
            break;
        case 2:
            bios98com_ch3(&cinf);
            data=cinf.data_size;
            break;
        default:
            break;
    }
} else data=bufman( SI2,0 );
return data;
}
/*

```

S I O のステータス読み込み

```

/*
int siosts()
{
    char    sts;

    if ( cp.ltype == 1 ) return (bufman( SI2,0 )!=0 ? 0x80 : 0 );
    switch(cp.sio)
    {
        case 0:
            sts=inportb(0x32);
            break;
        case 1:
            sts=inportb(0xb3);
            break;
        case 2:
            sts=inportb(0xbb);
            break;
        default:
            break;
    }
    return 0x00ff&sts;
}
/*

```

S I O のブレイク信号送出

```

/*
void siobreak()
{
    int ch;

    if ( cp.ltype == 1 ) return;
    if(cp.sio==0)
    {
        outportb(0x32,0x3f);
        sleep(5);
        outportb(0x32,0x37);
    }
    else if(cp.sio==1)
    {
        outportb(0xb3,0x3f);
        sleep(5);
        outportb(0xb3,0x37);
    }
    else if(cp.sio==2)
    {
        outportb(0xbb,0x3f);
        sleep(5);
        outportb(0xbb,0x37);
    }
}
/*

```

S I O の 接 続

```

/*
void    sioconnect()
{
    if ( cp.ltype==1 ) return;
    switch(cp.sio)
    {
        case 0: outportb(0x32,0x37);    break;
        case 1: outportb(0xb3,0x37);    break;
        case 2: outportb(0xbb,0x37);    break;
        default:    break;
    }
}
*/

```

S I O の 切 断

```

/*
void    siostop()
{
    if ( cp.ltype==0 )
    {
        switch(cp.sio)
        {
            case 0: outportb(0x32,0x15);    break;
            case 1: outportb(0xb3,0x15);    break;
            case 2: outportb(0xbb,0x15);    break;
            default:    break;
        }
    } else {
        telcut();
    }
}
*/

```

S I O バ ッ フ ェ ー ク リ ア ー

```

/*
void    sioclear()
{
    int i;
    COM_INFO cinf;

    if ( cp.ltype==1 ) return;
    switch(cp.sio)
    {
        case 0:
            cinf.cmd=2;
            bios98com(&cinf);
            for(i=0;i<(cinf.data_siz);i++){
                cinf.cmd=4;
                bios98com(&cinf);
            }
            break;
        case 1:
            cinf.cmd=2;
            bios98com_ch2(&cinf);
            for(i=0;i<(cinf.data_siz);i++){
                cinf.cmd=4;
                bios98com_ch2(&cinf);
            }
            break;
        case 2:
            cinf.cmd=2;
            bios98com_ch3(&cinf);
            for(i=0;i<(cinf.data_siz);i++){
                cinf.cmd=4;
                bios98com_ch3(&cinf);
            }
            break;
        default:    break;
    }
}

```

```

)
/*
.....
      モデムの処理
.....
*/
#define MODMAS1      "電話番号が設定されていません。"
#define MODMAS2      "【モデムの電源断】【モデム無し】【コマンド形式が異なる】の何  
れかです。"
#define MODMAS3      "モデムのコマンドエラー"
#define MODMAS4      "話中です。再ダイヤルしますのでしばらくお待ちください。中止は  
(ESC)"
#define MODMAS5      "話中のため接続できません。再ダイヤルする場合はIRIを押して下さい。"
#define MODMAS6      "ダイヤルを中止しました。"
#define MODMAS7      "相手キャリアが検出できません。"
#define MODMAS8      "ダイヤルトーンが検出できません。"
#define MODMAS9      "話中のためダイヤルを中止しました。"
#define MODMAS10     "インディケーションエラー"
#define MODMAS11     "モデムの非MNP初期化ファイルが有りません。"
#define MODMAS12     "モデムのMNP初期化ファイルが有りません。"
#define MODMAS13     "モデム用初期化ファイルの書式が間違っています。"

int ModemType; /* モデムの形式 */
int PhoneType; /* 電話形式 */

char    MODATA[20][40];

int automodem()
{
    WINDOW wd={ 2,20,76,3,T_CYAN,T_WHITE,T_WHITE,0,"",1,1,0,2,0,0,0 };
    static char mes[1][80];
    char    combuf[80];
    char    ch;
    int i,sts,count,x,y;
    int tc,t;

    x=wherex();
    y=wherey();
    sts=0;
    dspconnectname();
    openmessage();
    textcursor( NODISP_CURSOR );
    sprintf(mes[0],"電話番号：%sに接続中。中止は(ESC)",cp.telno);
    twindow(1, WD_OPEN, &wd,mes );
    if(cp.telno[0]!=' '){
        errordisp(MODMAS1);
        twindow( WD_CLOSE );
        return FALSE;
    }
    if((sts=modeminit())!=0){
        siostop(cp.sio);
        twindow( WD_CLOSE );
        switch(sts)
        {
            case FALSE: errordisp(MODMAS6); break;
            case 1: errordisp(MODMAS2); break;
            case 2: errordisp(MODMAS3); break;
            case 3: errordisp(MODMAS11); break;
            case 4: errordisp(MODMAS12); break;
            case 5: errordisp(MODMAS13); break;
        }
        window(1,1,80,24);
        gotoxy(x,y);
        return FALSE;
    }
    tc=cp.ringno;
    t=0;
    while(tc<100){
        sioclear();
        modemsnd();
        for(i=0;i<120;i++){ /* 2分待 */
            if(siornum()!=0){
                ch=sioinput();

```

```

        if(ch > ' '){
            sts=modemchk(ch);
            break;
        }
    }
    if(key_in()==0x00ff&ESC){          /* 中断 */
        sts=FALSE; break;
    }
    setmem(mes[0],80,0);
    sprintf(mes[0]," 第 %2d 回目のダイヤル中 [ 経過時間%4d 秒] ",t+1,i);
    twindow(1, WD_REMAIN,&wd,mes );
    sleep(1);
}
if(sts!=0){
    for(i=0;i<10;i++){
        if(siorum()!=0)    sioinput();
        delay(100);
    }
    if(sts==4 || sts==5){
        t++;
        if(t<tc){
            sprintf(mes[0],"%s",MODMAS4);
            twindow(1, WD_REMAIN,&wd,mes );
            if(modemwait()==FALSE) break;
            else continue;
        }
        else{
            sprintf(mes[0],"%s",MODMAS5);
            twindow(1, WD_REMAIN,&wd,mes );
            ch=getch();
            if(ch=='R' || ch=='r'){
                t=0;
                continue;
            }
            else break;
        }
    }
    else break;
}
else break;
}
else break;
}
twindow( WD_CLOSE );
if(sts!=0){
    switch(sts)
    {
        case 1:    errordisp(MODMAS7); break;
        case 2:    errordisp(MODMAS8); break;
        case 3:    errordisp(MODMAS8); break;
        case 4:    errordisp(MODMAS9); break;
        case 5:    errordisp(MODMAS9); break;
        case 6:    errordisp(MODMAS10); break;
        case FALSE: errordisp(MODMAS6); break;
        default:    break;
    }
    siostop();
    window(1,1,80,24);
    gotoxy(x,y);
    return FALSE;
}
setatrib();
gotoxy(x,y);
return YES;
}
/*
-----
          モデムの初期化
-----
*/
int modeminit()
{
    int sts,i,j,k;
    char  ch,NAME[80];
    FILE  *p;

```



```

/* 初期化ファイルの読みだし */
if(cp.mnp==0){
    sprintf( NAME,"%s%stobihino.mod",ENU );
    if((p=fopen(NAME,"rb"))==NULL) return 3;
}
else{
    sprintf( NAME,"%s%stobihino.mnp",ENU );
    if((p=fopen(NAME,"rb"))==NULL) return 4;
}
/* 読みだし領域のクリア */
setmem(&MODATA[0][0],800,0);
i=0;
j=0;
while(1){
    ch=(char)getc(p);
    if(!feof(p)){
        fclose(p); break;
    }
    if(ch==CR){
        MODATA[i][j]=CR;
        i++; j=0;
        if(i==20){
            fclose(p); return 5;
        }
    }
    else if(ch>' ' && ch!=0x27){
        MODATA[i][j]=ch;
        j++;
        if(j==40){
            fclose(p); return 5;
        }
    }
    else if(ch==0x27){
        for(k=0;k<40;k++){
            if((char)getc(p)==CR) break;
            if(!feof(p)) return 5;
        }
    }
}
if(strstr(&MODATA[0][0],"AT")!=0) ModemType=0; /* AT */
else ModemType=-1; /* CCITT */
for(i=0;i<20;i++){
    if(MODATA[i][0]==0) return YES;
    if((sts=modemcom(&MODATA[i][0]))!=0){
        fclose(p);
        return sts;
    }
}
return YES;
}
/*
-----
          モデムへコマンド出力
-----
*/
int modemcom(char *buff)
{
    char SIOBUF[31];
    char ch;
    int i,t;

    setmem(SIOBUF,31,0);
    sioclear();
    for(i=0;i<39;i++){
        ch=*(buff+i);
        if(ch==0) break;
        siocoutput(ch);
    }
    sleep(2);
    i=0;
    while(1){
        if((char)key_in()==ESC) return FALSE;
        ch=siocinput();
        if(ch==0) break;
    }
}

```

```

        SIOBUF[i++]=ch;
    }
    if(SIOBUF[0]==0)    return 1;  /* モデムから返答なし */
    if(ModemType==0){
        if(strstr(SIOBUF,"OK")!=0)    return YES;
    }
    else{
        if(strstr(SIOBUF,"UAL")!=0)    return YES;
    }
    return 2;  /* コマンドエラー */
}
/*
-----
                        ダイヤリング処理
-----
*/
void    modemand()
{
    char    ch;
    int i;

    sioclear();
    /* 電話番号の送出 */
    if(ModemType==0){  /* AT */
        if(PhoneType==0)    modemdial("ATDP");
        else                modemdial("ATDT");
    }
    else{  /* CCITT */
        if(PhoneType==0)    modemdial("CRNP");
        else                modemdial("CRNT");
    }
}
/*
-----
                        モデムのステータスチェック
-----
*/
int modemchk(char c)
{
    char    *tel_data[2][5]={
        {"NO C","NO D","NO T","REDI","BUSY"},
        {"CFIR","CFIA","CFIA","CFIE","CFIE"}
    };
    char    buf[20];
    char    ch;
    int i,j,d;

    setmem(buf,20,0);
    buf[0]=0;
    for(i=0;i<18;i++){
        while(siornum()==0);
        ch=sioinput();
        if(ch==CR || ch==LF)    break;
        buf[i+1]=ch;
    }
    if(ModemType==0){
        for(i=0;i<5;i++){
            if(strstr(buf,tel_data[0][i])!=NULL)    return i;
        }
        return YES;
    }
    else{
        for(i=0;i<5;i++){
            if(strstr(buf,tel_data[1][i])!=NULL)    return i;
        }
        return YES;
    }
}
/*
-----
                        ダイヤルコマンド送出ルーチン
-----
*/
void    modemdial(char *buff)

```

```

(
    char    ch;
    int i=0;

    while(1){
        ch=*(buff+i);
        if(ch==0) break;
        siooutput(ch);
        i++;
        delay(100);
    }
    for(i=0;i<32;i++){
        ch=cp.telno[i];
        if(ch==' ') break;
        siooutput(ch);
        delay(100);
    }
    siooutput(CR);
}
/*
-----
                話中による待ち時間
-----
*/
int modemwait()
{
    char    num[3];
    char    NUM[5];
    int i,count,wait;
    int x,y;

    x=wherex();
    y=wherey();
    count=0x00ff&cp.ringsec;
    wait=count;
    for(i=0;i<count;i++){
        gotoxy(x,y);
        cprintf(" [ 後%s 秒 待 ] ",wait);
        if(key_in()==0x00ff&ESC) return FALSE;
        sleep(1);
        wait--;
    }
    return YES;
}
/*
.....
*                  時刻関係
*                  .....
*/
/*
-----
                時刻読み出し
-----
*/
void timerread(int *Time)
{
    union REGS    regs;

    regs.h.ah=0x2c;
    int86(0x21,&regs,&regs);
    *Time=regs.h.dh;
    *(Time+1)=regs.h.cl;
    *(Time+2)=regs.h.ch;
}
/*
-----
                秒時刻読み出し
-----
*/
long int secread()
{
    time_t    *timer;
    long int t;

```

```

        timer=&t;
        time(timer);
        return t;
    }
    /*
    -----
    時刻表示
    -----
    */
    void    timerdisp(int *time,long int sec)
    {
        long int    ti,to;
        int in[3],ot[3],st[3],ss;

        in[0]= *time;
        in[1]= *(time+1);
        in[2]= *(time+2);
        ti=sec;
        timerread(ot);
        to=secread();
        ss=(short int)(to-ti);
        st[2]=ss/3600;
        st[1]=(ss-(st[2]*3600))/60;
        st[0]=ss-((st[2]*3600)+(st[1]*60));
        textcursor(NODISP_CURSOR);
        textreverse(NOREVERSE);
        textcolor(co.box);
        gotoxy(31,1);
        cprintf("r-----");
        gotoxy(31,2);
        cprintf("|");
        textcolor(co.ch);
        cprintf(" 接続時刻");
        textcolor(co.itime);
        cprintf("[%02d:%02d]",in[2],in[1]);
        textcolor(co.ch);
        cprintf(" 現在時刻");
        textcolor(co.ntime);
        cprintf("[%02d:%02d]",ot[2],ot[1]);
        textcolor(co.ch);
        cprintf(" 経過時間");
        textcolor(co.ltime);
        cprintf("[%02d:%02d:%02d]",st[2],st[1],st[0]);
        textcolor(co.box);
        cprintf("|");
        gotoxy(31,3);
        cprintf("l-----");
    }
    return;
}

```

付録〇．LAN端末接続用ハンドラ・ソースファイル(ether.c)

```

.....
/*
/*                      LAN 端末接続用ハンドラ
/*
/*
/*
.....

#include <dos.h>
#include <mem.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <conio.h>
#include "inet.h"
#include "nfire.h"
#include "fall.h"

#define ANYPORT 0

int InetUect;
int EtherFlag=FALSE,EtherFin=YES,LagFlag=FALSE,
    rcvp=0,rcvpmax=0,rcvlen=0,rcvflg=0;

static TCB tcb,tcbr;
char sbuf[SBUF_SIZE],rbuf[RBUF_SIZE];
char ldbuf[100];

int eid,cid;
int slen,rlen;
int result,ibpk;
EPAddress dest_addr;
InetTable far *inet;

InetTable far *find_inet();          /* InetBIOS 記述モジュール探索用関数 */
int httons(int data);                /* ネットワークオーダへの型変換関数 */
int ib_request(TCB *tcbp);           /* InetBIOS ルーチンの呼出し */
int ib_open(int service,int myport,int *eid); /* プログラム間通信の開始 */
int ib_close(int eid);                /* プログラム間通信の終了 */
int ib_call(int eid,EPAddress *raddr,int *cid); /* コネクションの開設(active) */
int ib_hang(int eid,int cid);         /* コネクションの切断 */
int ib_send(int eid, int cid, char *buf, int len, int *actlen);
int ib_recv(int eid, int cid, char *buf, int len, int *actlen);
int ib_shutdown(int eid,int cid);     /* コネクションの切断 */
int ib_capacity(int eid,int cid,int *len); /* 送信可能なデータサイズの確認 */
int ib_peek(int eid,int cid,int *len); /* 受信データサイズの確認 */
void sayReason(int result);           /* エラーメッセージの表示 */
int inittelnets();                   /* コネクション確立用関数 */
int revbuff(void);                   /* フレーム到着監視用関数 */
int recv(char *rbuf,int *rlen);      /* 受信フレームの読み込み */
int send(char *sbuf,int *slen);      /* TCP Telnet サーバへのデータの送信 */
int fintelnets();                    /* コネクションの切断(シャットダウン)from client */

int telcon() /* ホスト・コンピュータとの接続 */
{
    WINDOW wd=( 2,22,76,3,T_YELLOW,T_CYAN,T_WHITE,0,"LAN 端末接続",
        ,1,1,0,0,0,0 );
    static char data[80]="接続中です。少々お待ちください。";
    int i,ldlen,tmp;

    if ( inittelnets()==FALSE ) return FALSE;
    twindow( 1,WD_OPEN,&wd,data );
    delay( 700 );
    while( 1 ) /* ネゴシエーション・ルーチン */
    {
        setmem( ldbuf,100,0 );
        delay( 400 );
        if ( revbuff()==0 ) break;
        recv( ldbuf,&ldlen );
        for ( i=0;i<ldlen;++i){
            if ( ldbuf[i]==0xff )
            {
                if ( ldbuf[i+2]==0x18 )

```

```

        {
            tmp=3;
            send( &ldbuf[i],&tmp );
        }
        i+=2;
        continue;
    } else {
        rbuf[rcvpmax++]=ldbuf[i];
        rcvflg=1;
    }
}
}
twindow( WD_CLOSE );
return YES;
}

int bufman(int mode,char data) /* バッファ・マネージャ */
{
    int slen,i,rv=0;

    if ( LagFlag!=YES ){
        if( revbuff()!=0 && rcvpmax+ibpk<RBUFSIZE )
        {
            recv(&rbuf[rcvpmax],&rcvlen);
            rcvpmax += (rcvlen);
            rcvflg=1;
        }
    }
    switch( mode )
    {
        case RCU: if ( rcvflg )
            {
                rv=(int)rbuf[rcvp++];
                if ( rcvp==rcvpmax ) { rcvp=0;rcvpmax=0;rcvflg=0;rcvlen=0; }
            }
            break;
        case SIZ: rv=rcvpmax-rcvp;break;
        case SND: slen=1;
            if ( data==0x0d ) {
                slen=2;
                sbuf[0]=0x0d;
                sbuf[1]=0x0a;
            } else sbuf[0]=data;
            send(sbuf,&slen);
        default : break;
    }
    return rv;
}

int telcut( void ) /* 通信終了処理 */
{
    int rslt=YES;
    if ( EtherFin==YES ) return YES;
    if ( fintelnet()!=YES ) {
        mouseoff();
        mesdisp(" 強制的に通信を終了しました");
        mouseon();
        rslt=FALSE;
    }
    EtherFin=YES;
    EtherFlag=FALSE;
    LagFlag=FALSE;
    return rslt;
}

/* ..... */
/*                                     低レベルサブルーチン                                     */
/* ..... */
/* ..... */

int htons(int data) /* ネットワークオーダへの型変換関数 */
{
    return ((data>>8)&0xff);((data<<8)&0xff00);
}

```

```

int ib_request(TCB *tcb) /* InetBIOS ルーチンの呼出し */
{
    union REGS in,out;
    struct SREGS sreg;
    TCB far *tcb_ptr=(TCB far *)tcb;

    sreg.es=FP_SEG(tcb_ptr);
    in.x.bx=FP_OFF(tcb_ptr);
    int86x(InetVect+1, &in, &out, &sreg);
    return out.x.ax;
}

int ib_open(int service,int myport,int *eid) /* プログラム間通信の開始 */
{
    memset(&tcb,'%0',sizeof(TCB));
    tcb.Command=INETB_OPEN;
    tcb.Service=service;
    tcb.LocalAddr.InetAddr.ina_port=htons(myport);

    ib_request(&tcb);

    *eid=tcb.EndPoint;
    return tcb.Result;
}

int ib_close(int eid) /* プログラム間通信の終了 */
{
    memset(&tcb,'%0',sizeof(TCB));
    tcb.Command=INETB_CLOSE;
    tcb.EndPoint=eid;

    ib_request(&tcb);

    return tcb.Result;
}

int ib_call(int eid,EPAAddress *raddr,int *cid) /* コネクションの開設(active) */
{
    memset(&tcb,'%0',sizeof(TCB));
    tcb.Command=INETB_CALL;
    tcb.EndPoint=eid;
    memcpy(&tcb.RemAddr, raddr, sizeof(EPAAddress));

    ib_request(&tcb);

    *cid=tcb.Connection;
    return tcb.Result;
}

int ib_hang(int eid,int cid) /* コネクションの切断 */
{
    memset(&tcb,'%0',sizeof(TCB));
    tcb.Command=INETB_HANGUP;
    tcb.EndPoint=eid;
    tcb.Connection=cid;

    ib_request(&tcb);

    return tcb.Result;
}

/.....
・ ストリーム・サービスでの送受信関数 ・
/.....
int ib_send(int oid, int cid, char *buf, int len, int *action)
{
    memset(&tcb,'%0',sizeof(TCB));
    tcb.Command=INETB_SEND;
    tcb.EndPoint=oid;
    tcb.Connection=cid;
    tcb.Prmlen=len;
    tcb.Prmbuf=(char far *)buf;

    ib_request(&tcb);
}

```

```

        *actlen=tcb.Prmlen;
        return tcb.Result;
    }

int ib_recv(int eid, int cid, char *buf, int len, int *actlen)
{
    memset(&tcb, '0', sizeof(TCB));
    tcb.Command=INETB_RECV;
    tcb.EndPoint=eid;
    tcb.Connection=cid;
    tcb.Prmlen=len;
    tcb.Prmbuf=(char far *)buf;

    ib_request(&tcb);

    *actlen=tcb.Prmlen;
    return tcb.Result;
}

int ib_shutdown(int eid, int cid)
{
    memset(&tcb, '0', sizeof(TCB));
    tcb.Command=INETB_SEND;
    tcb.EndPoint=eid;
    tcb.Connection=cid;
    tcb.Flags=F_SHUTDOWN; /* shutdown flag */
    tcb.Prmlen=0;

    ib_request(&tcb);

    return tcb.Result;
}

int ib_capacity(int eid, int cid, int *len) /* 送信可能なデータサイズの確認 */
{
    memset(&tcb, '0', sizeof(TCB));
    tcb.Command=INETB_CAPACITY;
    tcb.EndPoint=eid;
    tcb.Connection=cid;

    ib_request(&tcb);

    *len=tcb.Prmlen;
    return tcb.Result;
}

int ib_peek(int eid, int cid, int *len) /* 受信データサイズの確認 */
{
    memset(&tcb, '0', sizeof(TCB));
    tcb.Command=INETB_PEEK;
    tcb.EndPoint=eid;
    tcb.Connection=cid;

    ib_request(&tcb);

    *len=tcb.Prmlen;
    return tcb.Result;
}

int getInetVectNumber(char *env)
{
    int i, j=0;

    for(i=0; env[i]!='\0'; i++) {
        if(env[i]>='0' && env[i]<='9')
            j=j*10+(env[i]-'0');
        else if(env[i]>='A' && env[i]<='F')
            j=j*16+(env[i]-'A'+10);
        else if(env[i]>='a' && env[i]<='f')
            j=j*16+(env[i]-'a'+10);
        else
            return NULL;
    }
    return j;
}

```



```

)

InetTable far *find_inet() /* InetBIOS のモジュール記述テーブルを探す */
(
    InetTable far *ptr;
    char *env,a[5];
    int i;

    a[4]='\0';
    if( (env=getenv("INETVECT"))!=NULL ) {
        InetVect=getInetVectNumber(env);
    }
    else
        InetVect=0x7e; /* default INT vector */
    for(ptr=(InetTable far *)getvect(InetVect);ptr;ptr=ptr->Common.NextTable) {
        for(i=0;i<4;i++) a[i]=ptr->Common.ModuleMagic[i];
        if(strcmp(a,"ASC")!=0)
            return NULL; /* InetBIOS なし */
        for(i=0;i<4;i++) a[i]=ptr->Common.ModuleType[i];
        if(strcmp(a,"NET")==0)
            return(ptr); /* InetBIOS 発見 */
    }
    return NULL; /* InetBIOS なし */
)

void sayReason(int result)
(
    mouseoff();
    EtherFlag=FALSE;
    switch(result) {
        case Elo:
            mesdisp(" | / O エラーが発生しました");
            break;
        case ENoMem:
            mesdisp(" メモリが足りません");
            break;
        case ENoDev:
            mesdisp(" アダプタがありません");
            break;
        case EINVAL:
            mesdisp(" コマンドまたはパラメータが不正です");
            break;
        case EMFile:
            mesdisp(" エンドポイントまたはコネクションの数が超過しました");
            break;
        case EMsgSize:
            mesdisp(" メッセージが長すぎます");
            break;
        case EOpNotSupp:
            mesdisp(" 指定された機能がサポートされていません");
            break;
        case EAddrInUse:
            mesdisp(" 指定されたアドレスは既に使用されています");
            break;
        case ENetDown:
            mesdisp(" ネットワークが停止しています");
            break;
        case EUnreachabl:
            mesdisp(" 指定されたエンドポイントには、送れません");
            break;
        case EConnAborted:
            mesdisp(" コネクションをこちらから強制的に切断しました");
            break;
        case EConnReset:
            mesdisp(" コネクションは相手から強制的に切断されました");
            break;
        case EShutdown: /* 相手は全てのデータを送り終えた */
            LagFlag=( rcvflag!=0 ? YES : FALSE );
            break;
        case ETimeOut:
            mesdisp(" タイムアウトしました");
            break;
        case EConnRefused:
            mesdisp(" コネクト要求が相手に拒否されました");
    }
}

```

```

        break;
    default:
        mesdisp("エラーコードそのものも不正です");
    }
}
mouseon();
}

/*.....*/
/*
/*                      中間レベルサブルーチン
/*
/*.....*/

int inittelnets() /* コネクション確立用関数 */
{
    static char adr[4][5];
    int i,i2,i3;
    /* InetBIOS のモジュール記述テーブルを探す */
    if( (inet=find_inet()) == NULL ) {
        mesdisp("fatal error: InetBIOS not loaded");
        EtherFlag=FALSE;
        EtherFin=YES;
        return FALSE;
    }
    /* 接続先のネットアドレス等のパラメータの設定 */
    memset(&dest_addr,'0',sizeof(EPAddress));
    dest_addr.Length=8;
    dest_addr.InetAddr.ina_port=htons(23); /* TCP telnet port */

    setmem( adr,20,0 );
    i2=0;
    for( i=0;i<strlen(cp.telno);++i )
    {
        i3=0;
        while( cp.telno[i] != '.' )
        {
            adr[i2][i3++]=cp.telno[i++];
        }
        i2++;
    }
    /* サーバのネットアドレス */
    dest_addr.InetAddr.ina_iaddr.ia_c[0]=(char)atoi(adr[0]);
    dest_addr.InetAddr.ina_iaddr.ia_c[1]=(char)atoi(adr[1]);
    dest_addr.InetAddr.ina_iaddr.ia_c[2]=(char)atoi(adr[2]);
    dest_addr.InetAddr.ina_iaddr.ia_c[3]=(char)atoi(adr[3]);

    /* プログラム間通信のエンドポイント識別子の取得 */
    if( ( result=ib_open(SRU_STRM, ANYPORT, &eid) )!=0 ) {
        mesdisp("イーサネットのオープンができません");
        mouseon();
        sayReason(result);
        mouseoff();
        EtherFlag=FALSE;
        EtherFin=YES;
        return FALSE;
    }
    /* TCP Telnet サーバに接続 */
    if( ( result=ib_call(eid,&dest_addr,&cid) )!=0 ) {
        mesdisp("コネクションを開設できません");
        mouseon();
        sayReason(result);
        mouseoff();
        return FALSE;
    }
    /* コネクションを開設した */
    return YES;
}

int revbuff(void) /* フレーム到着監視用関数 */
{
    int len;

    if( ( result=ib_peek(eid,cid,&len) )!=0 ) {
        /* データサイズが確認できない */
    }
}

```

```

        sayReason(result);
        return 0;
    }
    ibpk=len;
    return len;
}

int recv(char *rbuf,int *rlen) /* 受信フレームの読み込み */
{
    int len=*rlen,i;

    if( ( result=ib_recv(eid.cid,rbuf,RBUFSIZE-1,&len) )!=0 ) {
        /* データを受信できない */
        sayReason(result);
        return FALSE;
    }
    *rlen=len;
    return 1;
}

int send(char *sbuf,int *slen) /* TCP Telnet サーバへのデータの送信 */
{
    if( ( result=ib_send(eid.cid,sbuf,*slen,slen) )!=0 ) {
        /* データを送信できない */
        sayReason(result);
        return FALSE;
    }
    return 1;
}

int fintelnet() /* コネクションの切断(シャットダウン) from client */
{
    if( ( result=ib_shutdown(eid.cid) )!=0 ) {
        /* コネクションが終れない */
        sayReason(result);
        return FALSE;
    }
    if( ( result=ib_hang(eid.cid) )!=0 ) {
        /* コネクションが終れない */
        sayReason(result);
        return FALSE;
    }
    ib_close(eid);
    return YES;
}

```