パソコン内蔵通信プロセサシステム

木下雄弘
（㈱東芝　青梅工場）

近年，パーソナルコンピュータ用のマルチタスクオペレーティングシステムが開発されてきており，パーソナルコンピュータにおける通信制御ソフトウェアの効率性がより厳しく問われ始めている。即ち，パーソナルコンピュータにおいてさえもフロントエンドネットワークプロセサの様な通信制御装置が必要になりつつある。小文では，通信プロトコルやチャネルI／Oをサポートするパーソナルコンピュータ用内蔵通信制御装置の試作例及びマルチタスクオペレーティングシステムにより稼動するパーソナルコンピュータにおける有用性について紹介するものである。

Communication Adaptor for the Personal Computer　(In Japanese)

Katsuhiro KINOSHITA
Ome works, Toshiba corporation, 2-9 Suehiro-cho, Ome-shi, Tokyo, Japan

Recently the multi task operating system has developed for the personal computer. So the efficiency of the Communication software on the personal computer is to be very important. And the communication adaptor like a front-end network processor will be needed even in case of the communication on the personal computer.

This paper describes a specimen of the communication adaptor for the personal computer, which supports the process of a communication protocol & the channel I/O.

As a result, such an adaptor proved to be very useful for the personal computer on the multi task operating system.

1. 序

従来，MS-DOS*等をオペレーティングシステムにもつパーソナルコンピュータ及び低価格のワークステーションにおける通信は通信制御に伴う処理を全てメインCPUにて行う方式が主流であった。しかしながら，パーソナルコンピュータ及び低価格のワークステーション用の周辺機器装置の充実並びにマルチタスクオペレーティングシステムの出現等により，通信制御における割り込み処理，データ送受信バッファ管理及びプロトコル処理に伴うオーバヘッドは，パーソナルコンピュータ及び低価格のワークステーション（以下，メインプロセッサと呼ぶ。）においても，通信システムを構築するに当たり無視できないものとなってきている。そこで，この問題解決手段として，いわゆるアドオンCPUボードとも言うべき通信アダプタをV.24，X.21及びLANインタフェース等に対応して検討した。ここでは，V.24及びX.21用に試作した通信アダプタ（仮称マシンコード：CAN（Communication Adapter for Networks））についての報告をするものである。

尚，メインプロセッサにおいてはオペレーティングシステムとして，C-DOS**を採用した。

*：MS-DOSは，マイクロソフト社の登録商標である。

**：C-DOSは，デイジタルリサーチ社の登録商標である。

2. システムアプローチ

本システムを構築するに当たり，特に留意した点は以下の項目である。

①オペレーティングシステムの負荷分散等

②メインプロセッサとCANの結合方式

③CANのサポートする通信プロトコルをダイナミックに選択可能とする。

④上記①～③についてメインプロセッサのアーキテクチャの標準性を崩すことなくコストパフォーマンスを追求する。

以下，検討項目につき詳細に述べる。

3. ソフトウェア構造について

3.1 オペレーティングシステム

3.1.1 負荷分散

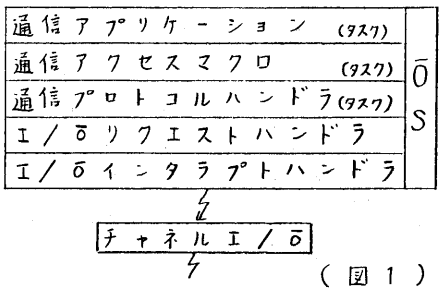
まず，メインプロセッサにおける通信制御に伴う処理は以下の様に大別される。

①通信アプリケーションからの要求受け付け処理（通信アクセスマクロ）

②通信プロトコルハンドリング処理

③チャネルI/Oに対する要求処理及び終了処理（I/Oリクエストハンドラ）

④チャネルI/Oからの割込み処理（I/Oインタラプトハンドラ）



（図1）

これらの処理の中で，最も優先的にCPUを必要とするI/OドライバであるI/Oインタラプトハンドラ及びI/Oリクエストハンドラがネックと

なった場合，オーバラン等のフェータルな現象が発生することは周知の事実である。　そこで，I／Oドライバについてはdmaの採用によるCPUの負荷軽減が考えられるが，通信プロトコルハンドラの動作がオペレーティングシステムのオーバヘッドや他のタスクの動作により遅れプロトコル上のリスポンスが遅れた場合，回線効率低下は容易に予想できる。　簡単な例として，マルチドロップ下におけるポーリング方式を考えると，ポーリングに対するリスポンスタイムの遅れがそのシステム全体に及ぼす影響やポーリングに伴い発生するディスパッチによる負荷等の問題が考えられる。　そこで，通信プロトコルハンドラをタスクとせずにドライバに組み入れることても考えうるが，プロトコルが重い場合再び問題となろう。　従って，効率性，再利用性及び保守性等の面から通信プロトコルハンドラ，I／Oリクエストハンドラ及びI／Oインタラプトハンドラを CANへ移し，通信プロトコルハンドラと通信アクセスマクロをdmaによりインタフェースを取ることとした。　これによりメインプロセッサ側のオペレーティングシステムの負荷を軽減し，更にはCANによる効率向上によりトータルスループットの向上を期待することとした。

3．1．2　CANにおいて

CANにおけるオペレーティングシステム（以降，I－15と呼ぶ。）は，ファームウェアの実行をタスク単位に制御し，タスク間の交信，バッファ用のメモリ管理及びタイマ管理等のサービスを提供するモジュールの集合とする。

3．1．2．1　I－15設計方針

I－15は，インテル社のIAPX186（以降，80186と呼ぶ。）を搭載した全ての通信用ハードウェア上（但し，マルチプロセッサ構成は対象外とする。）で動作することを目的とするが，80186のCPU本体と内蔵タイマユニットのみを使用し，その他のハードウェア（例えば，SIO（直列インタフェース）等の周辺素子）に依存する機能についてはファームウェアに任せるものとする。　即ち80186のCPU本体と内蔵タイマユニット以外の割り込み制御等に関する処理については，ファームウェアであるタスク及び固有のI／Oドライバで行うことを前提としROM化した。

3．1．2．2　サービス機能概略

I－15下で動作する各タスク（含割り込み処理モジュール等。）からI－15のサービス機能をアクセスする為には，入力パラメタをレジスタに代入し，各サービス機能ごとに対応する内部割り込み命令を実行する。　また出力パラメタは同様にレジスタ及びフラグに代入される。　主なサービス機能の概略は以下の通りである。

サービス	機能概略
タスク間交信	セマフォ方式によるタスク間のメッセージ交換（同期）
バッファ管理	タスクにて使用するメモリバッファの管理
タイマ管理	タスクにて制御を監視する為のタイマを提供。
キュー管理	バッファキュー管理
付加機能	システム停止及びシステム時刻の通知

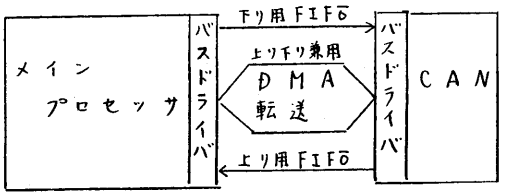
3．1．2．3　再入性

通信ファームウェアにおいては割り込みが多発するのが一般的ゆえ，割り込みに対する対応速度が要求される。　従ってI－15においては殆どのサービス処理モジュール（含タイマ割り込み処理モジュール）が再入可能とする必要がある。　しかし，排他制御を必要とするタスクの為に再入禁止とすることが可能である必要もある為，タスクがI－15のサービス機能をアクセスする際にサービス処理の再入の可否を選択可能とした。

3．2　メインプロセッサとの結合

3．2．1　方針

通常，メインプロセッサと通信プロセッサの間でインタフェースをとる場合，メインプロセッサ側に通信プロセッサとの共有メモリを有したバス接続が考えられる。　しかしながら，汎用オペレーティングシステムを搭載したメインプロセッサに，この様な共有メモリを持たせることはメインプロセッサを特化させることであり，標準性を失う。　我々は，メインプロセッサの標準性を保つため，共有メモリをメインプロセッサに持たせずメインプロセッサとCANの間はDMAによるバス接続を考えた。　しかし，メインプロセッサ側においては通信用DMACが1つのみしか標準確保できないため半二重のDMA転送となった。　そこでDMA転送におけるコンテンションを回避するためFIFOメモリを上り，下り専用を各々1本ずつ計2本導入し，メインプロセッサとCANにおけるDMA転送の同期をとることとした。　尚，CAN側のDMACは80186内蔵のものを使用することとした。



（図2）

3．2．2　ソフトウェアインタフェース概略

メインプロセッサとCANのバス接続におけるソフトウェアインタフェースとして，以下の様なFIFOおよびDMAによる転送フレーム形式を決定した。

（図3）

CI/RI	LI	パラメタ部	情報部
(1バイト)	(2バイト)	(可変長)	(可変長)

①CI／RI

コマンド／リスポンス識別子であり，メインプロセッサからCANへの方向の場合をコマンド，またその逆方向の場合をリスポンスとする。

コマンド／レスポンスには，CANのローダ起動，ファームウェアのダウンラインローディング，ファームウェア起動，コネクション確立／切断，データの送受信，アップラインローディングおよび異常通知等に関するものがある。

②LI

パラメタ部と情報部の合計バイト長である。

③パラメタ部

内容，長さはCI／RIによる。このフィールドまでをFIFOにより転送する。

④情報部

内容，長さはCI／RIによる。このフィールドはDMAにより転送する。

3．3　プロトコルローディング

CANにおいてサポートする通信プロトコルは，ダウンラインローディング要求コマンドをCAN側のローダ（I－15とともにCANのROMに内蔵。）に対し発効することによりCAN側のRAM上にメインプロセッサよりローディングされる。

又，CANにおけるバスドライバ及びチャネルI／Oドライバも同様である。即ち，メインプロセッサ側のアプリケーションが通信アクセスマクロを通じCANに対しプロトコル及びドライバをダウンラインローディングできる様にすることによりメインプロセッサ側のアプリケーションに多様性を与える。

3．4　通信アクセスマクロ

メインプロセッサにおける通信アプリケーションとバスドライバのインタフェースとなる通信アクセスマクロは，通信アプリケーションに対し，プロトコルを意識させず，通信をファイルI／Oと見做させる様にするために，①ファームウェアのダウンラインローディングと起動及び物理回線のコネクション，②論理回線のコネクション，③データ送信，④データ受信，⑤論理回線のディスコネクション並びに⑥物理回線のディスコネクションの機能エントリーをもつ。

3．5　まとめ

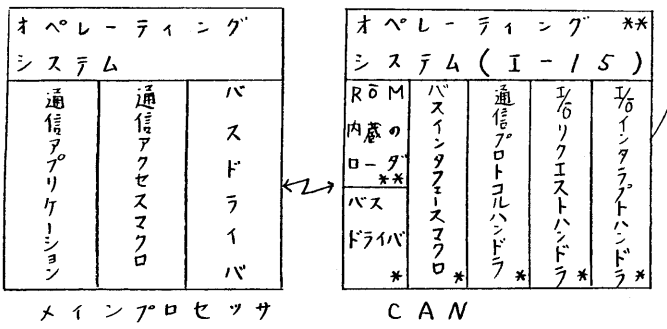
以上の検討から，メインプロセッサとCANにおけるソフトウェア構造の概略は（図4）の様になる。

4．ハードウェア構造について

4．1　方針

CANにおいては，メインプロセッサにおけるソフトウェアの移植性とI－15の構築環境を考慮し，CPUとしてインテル社のIAPX186（80186）を採用する。また，ローダ及びI－15専用ROM（64KB），ファームウェア用RAM（512KB）及び通信用のSIOを実装させ以下の仕様を満たすこととする。

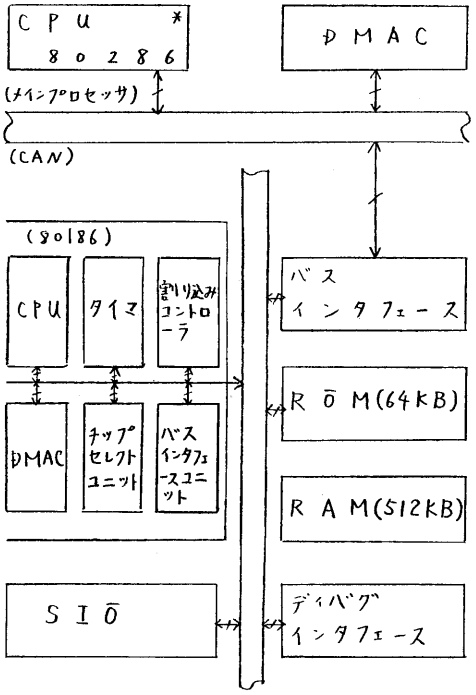
①適用機種；弊社のパソコン及びワークステーション
②実装形態；メインプロセッサに内蔵させる。
③通信チャネル数；試作では1チャネル
④適用回線；特定／公衆／DDX
⑤同期方式；外部同期／内部同期
⑥通信速度；300～48Kbps
⑦インタフェース；V．24及びX．21
⑧符号化方式；NRZ及びNRZI



（図4）

4．2　実装概略

以上のソフトウェア及びハードウェアの検討により，CANにはCPUとして80186，通信チャネルSIO，ROM（64KB），RAM（512KB）を搭載し，メインプロセッサとはDMAによるバス接続とした。又，DMA転送の同期手段としてFIFOメモリを採用することになった。以下，概略図として（図5）を示す。



（図5）

*：80286はインテル社の登録商標。

5．　CANの性能考察

現在，メインプロセッサ側の通信アクセスマクロ，バスドライバ並びにCAN側のバスドライバ，バスインタフェースマクロ，通信プロトコルハンドラ（X.25 '80），I/Oリクエストハンドラ及びI/Oインタラプトハンドラはテスト中であり，メインプロセッサを含めた本格的な性能テストは今回見送らざるを得ないが，以下CANの性能を机上計算にて考察する。（以降（図4）参照。）

いま，256バイトのデータを9600bpsで送受信する場合を想定すると，一般に1バイト当り8.33×10^{2}μsec以内の処理が必要であり，又256バイトのデータの転送時間は2.13×10^{5}μsec必要とする。ここで，CANにおけるI/Oインタラプトハンドラの処理時間（含I-15のオーバヘッド）をLI，又CANにおけるバスドライバの処理時間（含I-15のオーバヘッド）をBIとすると，送受信ともにLI＋BI＜8.33×10^{2}μsecかつ$\overline{O}$（LI＋BI）≦10μsecである。

更に，CANのI/Oリクエストハンドラの処理時間，CANのバスインタフェースマクロの処理時間及びバスドライバの処理時間の和をS（含I-15のオーバヘッド）とすると，送受信ともに，S＜2.13×10^{5}μsecかつ$\overline{O}$（S）≦10^{3}μsecである。従って，メインプロセッサ側の処理速度を考察から外した場合，CANがボトルネックとなることはないと判断する。

次に，仮にメインプロセッサ側の処理速度が遅い場合，CANにおいてウィンドウ制御に伴う処理を行わねばならない。ここで，RR，Pを受信もしくは，RNR．FまたはRR．Fを送信するためには，一般に4.99×10^{3}μsec必要とし，通信プロトコルハンドラの処理時間及びI/Oリクエストハンドラ（含ターミネーショ

ン処理）の処理時間の和をP（含エ－15のオーバヘッド）とすれば，送受信ともにLI＋BI＋P＜4．99×10³μsecかつ0（LI＋BI＋P）≦10²μsecである。　従って，メインプロセッサ側の処理速度低下に伴い発生するCANにおけるウィンドウ制御に伴う処理は保証されると判断する。

以上の考察から，メインプロセッサの処理速度に関係なく，CANがボトルネックとなることはないと判断する。

6．今後の課題

現段階においては，性能テスト未着手ゆえ，CANにおいてさえも推論の域を出ないが，今後メインプロセッサを含め，48Kbpsという条件でX．25（'80）のプロトコルを使用しCANの有用性を検証していき，更には近い将来において，複数回線（当面9600bps×2），異種プロトコル同時サポート（回線ごとに相異るプロトコルを動作させる。）について検討していくことを今後の課題とする。

謝辞

本検討に当たり，有意義なご指導並びに，ご助言を頂いた関係各位に深謝します。

参考文献

1)　和賀井・小森・田中・元岡：
〝網向きプロセス間通信制御プロセッサの構成〟
情報分散処理システム研究会
8－3，1981

2)　和賀井・田中・元岡：
〝網向きオペレーティング・システムについての一考察〟
信学研資EC77－43

3)　津田・筒井・前田・横畑：
〝マイクロコンピュータ用モジュール構造リアルタイムOSの一構成法〟
信学論（D）J69－D，2，
PP159－169，1986

4)　中島・斉藤・猪瀬：
〝ネットワーク・オペレーティング・システムにおけるプロセス管理方式〟
信学論（D）J69－D，3，
PP355－364，1986