ＩＴＲＯＮ／ＭＭＵ２８６

工藤　健治、　下原　明、　安田　秀徳　（　富士通株式会社　）
本田　昭人　　（　富士通マイコンシステムズ株式会社　）

１．はじめに

制御用リアルタイムOSの標準となりつつあるITRON[1][2][3][4][5]を、先進の16ビットマイクロプロセッサMBL80286（インテル社のiAPX286互換品：以下単に80286と記す）で実現することを試みた。

ITRONは、本来マイクロプロセッサのアーキテクチャに依存しないよう設計されてはいるが、80286が特長としているハードウェアによるメモリ管理機能（アドレス変換機能やメモリ保護機能等からなる：以下これをMMU機能と記す）を十分に活かすためには、従来の仕様に機能拡張をする必要がある。

そこで我々は、このMMU機能を活かした高度なメモリ管理機能を実現するための拡張機能と、その実現方法について考察したのでここに報告する。

なお、この拡張されたメモリ管理機能を、従来のITRONにあるメモリプール管理機能と区別するため特にMMUサポート機能という。本OSをMMUサポート機能を持つ80286用ITRONという意味で、ITRON/MMU286と呼ぶが、以下ではこれを短縮して単にITRON/286と記す。

このITRON/286は、従来のITRONとくらべて次のような特徴を持つ。

１）MMU機能の利用によるシステムの信頼性向上。
２）ハードウェアのタスク管理機能を利用したタスクスイッチの高速化。

２．ＩＴＲＯＮ／２８６の仕様

ITRON/286の外部仕様は、基本的に次のようになる。

（従来のITRON/86の仕様）　＋　（MMUサポート機能）

MMUサポート機能の基本設計方針は次のようにした。

１）制御用リアルタイムOSとしてシステムに組み込みで使用されるという性質をふまえ、仕様が重くなりすぎないようにする。
２）従来のITRON/86の仕様はできるだけ変更しないようにし、新しいシステムコールの追加で対応する。
３）柔軟なシステムを構築できるように、メモリ管理の拡張機能はプリミティブなレベルでサポートする。

以下に具体的な拡張機能とその実現方法をあげる。

１）拡張機能は、MMU機能を利用した主にメモリ保護に関連する次のものである。
仮想メモリの機能はITRONでは取り入れず、BTRON [6][7] に譲る。
- ○ 論理アドレスと物理アドレスの分離によるプログラムのリロケーション
- ○ タスクのメモリ空間の分離
- ○ 共有空間の管理
- ○ メモリのアクセス保護
- ○ 特権レベルによる保護

（注）　ITRONおよびBTRONは、（社）日本電子工業振興協会により市場調査されまとめられた基本システムおよびオペレーティングシステムの要求仕様をもとに、東京大学の坂村博士を中心とするグループにより機能設計されたものである。

2）上記機能を実現するために、メモリスペース（SPC）という新しいオブジェクトの概念を導入する。

○ メモリスペースは、スペースIDにより管理される。

○ メモリスペースには、ユーザによってメモリ空間の割り付けが行われる。

○ タスクは、ユーザが定義したメモリスペースとスペースIDにより対応づけられ、そのメモリ空間上で動作する。

3）タスクのメモリ空間を動的に管理するようなことは行わない。そのために従来より性能が低下することはない。

2．1　拡張されるシステムコール

MMUサポート機能に関するシステムコールを表１に示す。
これらは基本的にプリミティブな機能レベルでサポートしているが、その主なものを以下に説明する。

1）cre_spc
新しい論理空間（メモリスペース）を生成する。OSの内部処理としては、論理空間と物理空間を対応づけるメモリテーブルの領域を確保し、その管理データを割り付ける。

2）del_spc
論理空間を削除する。cre_spcで確保、設定された領域を解放する。

3）cre_map
物理空間から論理空間への割り付けを行う。論理空間への割り付けは、cre_spcで確保されたテーブルにデータを設定することによって行われる。

4）trf_map
ある論理空間から別の論理空間へ再割り付けを行う。ある論理空間に定義されている物理空間を別の論理空間に再定義して、もとの論理空間からは解放する。

5）sha_map
論理空間の共有割り付けを行う。ある論理空間に定義されている物理空間を別の論理空間にも定義し、しかももとの論理空間からは解放しない。

cre_map、trf_map および sha_mapでは、全タスクの共有空間を定義することができる。また他のメモリスペースと物理メモリの共有を行ったり、コードセグメントを書き換えるなどの柔軟な処理が可能である。

表１　ＭＭＵサポート機能システムコール

システムコール名	機　　能　　概　　要
cre_spc	新しい論理空間を生成する
del_spc	論理空間を削除する
exd_spc	自タスクを終了／削除し、論理空間も削除する
spc_adr	論理空間管理ブロックのアクセスアドレスを得る
cre_map	物理空間から論理空間への割り付けを行う
trf_map	ある論理空間から別の論理空間への再割り付けを行う
sha_map	論理空間の共有割り付けを行う
del_map	論理空間への割り付けを解除する
map_adr	マッピング管理ブロックのアクセスアドレスを得る
set_prt	メモリ保護属性を設定する
get_dat	システムデータの読み出しを行う

表２　ＭＭＵ複合システムコール

システムコール名	機　　能　　概　　要
mcre_tsk	独立した論理空間を持つタスクを生成する
mdel_tsk	独立した論理空間を持つタスクを削除する
mexd_tsk	独立した論理空間を持つ自タスクを終了、削除する
mget_blk	指定したアドレスにメモリを獲得する
mrel_blk	指定した論理空間のメモリブロックを解放する
msnd_msg	異なる論理空間へメッセージを送信する
mrcv_msg	異なる論理空間からメッセージを受信する

3．ＩＴＲＯＮ／２８６の設計

ITRON/286は、80286の機能を最大限に活かすように設計されている。ここでは、ITRON/286の設計上のポイントについて80286のアーキテクチャ[8][9][10][11][12]を見ながら、その概要を述べる。

3．1　ＭＢＬ８０２８６

80286は、8086との互換性を保ちつつ機能の強化拡張が行われている。80286には、8086と完全な互換性を保って動作するリアルモードと、先進の技術を取り込み拡張された機能をフルに活かして動作するプロテクトモードの２つの動作モードがある。80286は、リアルモードで動作する場合でも8086にくらべると性能強化されており、高速版8086として動作する。このため、従来のITRON/86をそのまま80286で動作させても、高性能ITRONとなる。しかし、これでは80286の機能を十分に活かしたものとはならないため、ITRON/286はプロテクトモードで動作するように設計されている。

80286の特徴として、次のようなものがあげられる。

1）高性能マイクロプロセッサ
　パイプライン・バス方式の採用により8086に比べて大幅なスピードアップを実現。
2）8086のソフトウェア資産継承
　プロテクトモード動作時でも基本命令セットは8086と同じであり、8086のソフトウェアを実行可能。
3）広大なアドレス空間
　物理アドレス空間16Mバイト、論理アドレス空間1Gバイトを持つ。
4）メモリ管理機能の実現
　メモリマネジメントユニット（MMU）の機能を内蔵することにより、仮想アドレス、メモリの書き込み保護等の機能を実現。タスクのメモリ空間分離や、仮想メモリも実現可能。
5）特権保護の実現
　４つの特権レベルによる保護機構を持つ。OS等の信頼度の高いプログラムが、タスク等のより信頼度の低いプログラムによって破壊されることを防ぐような機能が実現可能。
6）ハードウェアによるタスク管理
　タスクの情報を保持するタスクステートセグメント（TSS）を認識し、ハードウェアによるタスク管理を実現。高速なタスクスイッチが実現可能。
7）拡張プロセッサのサポート
　数値演算プロセッサ80287を利用可能。80287は80286と連動して高速な数値演算を実現可能。

80286をプロテクトモードで動作させると、上記の機能がすべて利用可能であり、この時80286はマルチタスクシステムを構築するのに最適なように設計されている。

図１に80286のレジスタセットを示す。80286で拡張されたレジスタについては、次節以降で説明を行う。

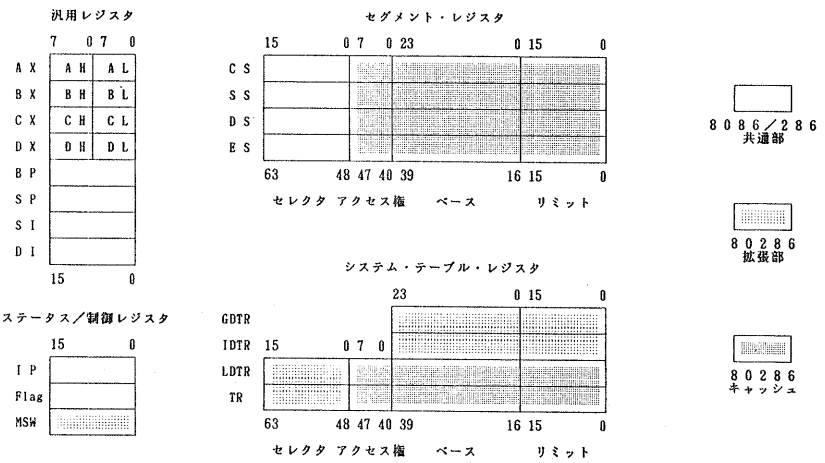

図１　８０２８６のレジスタセット

3．2　MMUサポート機能

ITRON/286は、80286のMMU機能を取り入れている。ここでは、このMMU機能のアーキテクチャと、ITRON/286ではそれをどのように利用してMMUサポート機能を実現しているかを示す。

80286は、セグメント方式のMMU機能を内蔵することにより、メモリ管理機能を実現している。80286ではいくつかのレジスタとメモリテーブルによってメモリが管理されている。

3．2．1　MMU用メモリテーブル

1）GDT（グローバルディスクリプタテーブル）
　システムに1つ存在する。全タスクに共通なメモリ空間を定義する。

2）LDT（ローカルディスクリプタテーブル）
　複数存在し、タスクに固有のメモリ空間を定義する。

これらのメモリテーブルには、ディスクリプタと呼ばれる8バイトのデータが定義されている。ディスクリプタにはアドレッシングに関する情報が保持されており、図2に示すような種類がある。図3にセグメントディスクリプタの形式を示す。80286では、このメモリテーブルを利用して論理アドレスを物理アドレスに変換する。

一般セグメントディスクリプタは、タスクが実際に利用するメモリ空間を定義する。ディスクリプタにはベースアドレス、すなわちセグメントの先頭アドレスと、その大きさおよび属性が定義されている。属性としては、実行可能、読み出し可能、書き込み可能などのほかに、そのセグメントの特権レベルがある。

特殊セグメントディスクリプタは、80286が管理するメモリテーブルを定義する。ディスクリプタの形式は基本的に一般セグメントディスクリプタと同じである。LDTディスクリプタとは、LDTを定義するものである。

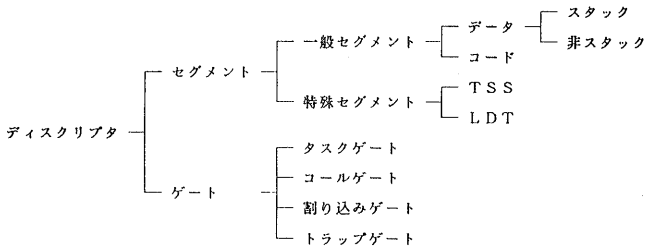

図2　ディスクリプタの分類

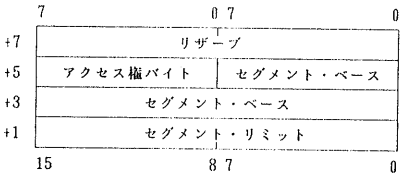

図3　セグメント・ディスクリプタ

3．2．2　MMU用レジスタ

メモリ管理に関係するレジスタには、GDTR、LDTR、および4つのセグメントレジスタ（CS,DS,ES,SS）がある。GDTRはGDTの存在する物理アドレスと大きさを定義する。LDTRは、GDT中に定義されているLDTディスクリプタの1つを示し、これにより間接的にLDTを定義する。LDTはタスク毎に定義されており、LDTRはタスクがスイッチされると自動的に変更されて別のLDTを指し、タスクに固有な論理アドレス空間を作り出す。

セグメントレジスタは、コード、データ、拡張データ、およびスタックのそれぞれが、現在どのセグメントを使用しているかを示す。各セグメントレジスタには、GDTまたは現在使用中のLDT上に存在するディスクリプタの1つを選択する、セレクタと呼ばれる値が保持されている。

3．2．3　MMU機構

MMU機構の構成を図４に示す。GDTRは、GDTを定義している。GDTには、いくつかのメモリセグメント、およびいくつかのLDTが定義されている。現在どのLDTが使用されているかは、LDTRの値によって選択される。GDTやLDTの中で定義されるメモリセグメントは、物理メモリを示す。

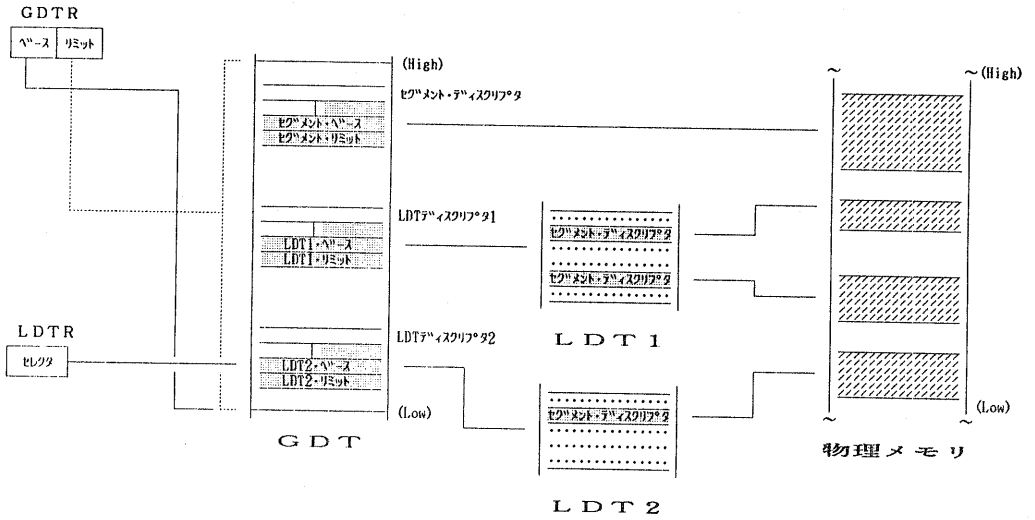

図４　ＭＭＵ機構

プログラム中でのアドレスは、セレクタ＋オフセットの値による論理アドレスによって指定される（図５）。セレクタには、GDTまたはLDTのどちらを選択するかの情報と、そのディスクリプタテーブル中のどのディスクリプタを選択するかの情報がある。この情報により選択されたディスクリプタによって、物理メモリ上のベースアドレスが決まる。これにオフセットの値を加えて指定されたアドレスで物理上の位置が決定される。

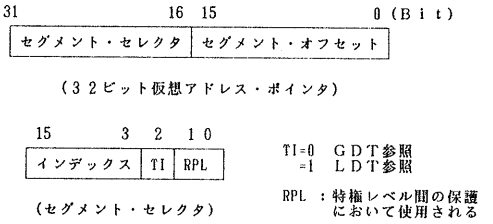

図５　論理アドレス

80286のMMU機能では、データセグメント以外への書き込みを禁止しているので、GDTやLDTなどのメモリテーブルにデータ設定をする場合には、これらの領域がデータセグメントとして参照できる必要がある。このためにセグメントの二重定義が必要となり、二重定義されたセグメントディスクリプタを特に〝別名ディスクリプタ〟という。

3．2．4　ＩＴＲＯＮ／２８６のメモリ管理

ITRON/286のMMUサポート機能は、主にGDT、LDTの操作を行うことにより実現される。

1）cre_spc、del_spc

これらのシステムコールは、LDTの領域確保・解放を行う。

cre_spcは次の処理を行う。

○ メモリスペースオブジェクト管理用の領域を確保し初期設定を行う。

○ LDT用のメモリ領域をフリーメモリより確保し、セレクタの空き領域管理等のための初期設定を行う。

○ GDTにLDTディスクリプタと、それを操作するためのLDT別名ディスクリプタを定義する。

del_spcは、cre_spcで確保、設定された領域を解放する。

2）cre_map、trf_map、sha_map、del_map

これらのシステムコールは、LDT中のディスクリプタの操作を行う。内部の処理は、主にディスクリプタテーブル空き領域の管理と、ディスクリプタの設定・解除である。

cre_mapでは、 指定されたメモリスペースに対応するLDTより、ディスクリプタの空き領域を獲得し、獲得したディスクリプタにパラメータで指定された物理メモリの位置、大きさの情報とその属性を設定する。

trf_mapでは、ソーススペースに対応するLDTの中の指定されたディスクリプタのデータをもとに、デスティネーションスペースに対して、cre_mapと同様な操作を行った後、ソーススペースのディスクリプタを削除する。

sha_mapの内部の処理はtrf_mapとほぼ同じで、もとの論理空間からの解放を行わない点が異なる。

del_mapは指定されたセレクタのディスクリプタの設定を解除する。

3．2．5　ＩＴＲＯＮ／２８６の動作

MMUサポート機能を用いてITRON/286がどのような動作をするかを示す。ITRON/286の動作時のメモリイメージを図６に示す。

ITRON/286では、タスクはスペースIDで対応づけられたメモリスペース上で動作する。異なったメモリスペースで動作するタスク間では、そのメモリ空間が分離されており、あるタスクの誤動作により別のタスクのデータが破壊されることを防いでいる。

タスクBとタスクCの例は、異なるメモリスペースに同一の物理メモリをcre_mapにより定義することによって、共有メモリとして扱うことが可能であることを示す。

より密接な関係を持って動作するタスクは、タスクDとタスクEのように、同一のメモリスペース上で動作させることも可能である。

ITRON/286はグローバルな共有メモリの扱いが可能である。cre_map等で論理空間を生成する時、スペースIDを(-1)に設定した場合はGDTにセグメントが定義され、この時決められるセレクタ値を用いればすべてのタスクからそのメモリへのアクセスが可能である。また、共有メモリ属性を付けておくと、メモリプールから獲得されるメモリはGDTに登録され、これにより動的に共有メモリを扱うことができる。

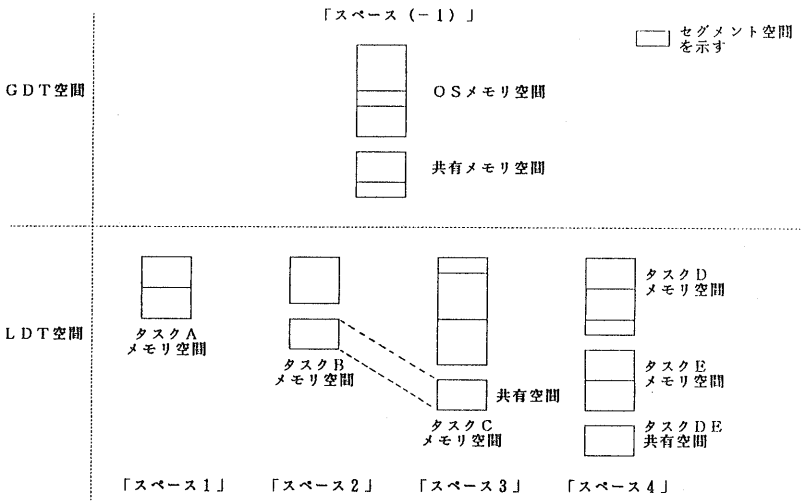

図６　ＩＴＲＯＮ／２８６のメモリイメージの例

3．3　特権レベル保護

ITRON/286では、80286の特権レベル保護機構を利用し、システムの信頼性を向上させている。

80286の特権レベル保護は、一般セグメントディスクリプタに設定されている特権レベルをもとに行われる。特権レベル保護の機構は複雑であるが、基本的にはある時点のCSセレクタ値が指すコードセグメントディスクリプタの特権レベルをその時のタスクの特権レベルとし、この特権レベルとアクセスしようとするデータセグメント等の特権レベルを比較することによりアクセス可能かどうかを判定し、不当なアクセスを防止しメモリ破壊等を防ぐことができる。

OSのデータ等システムで共有されるデータはGDTに設定されているため、どのタスクからもアクセス可能であるが、その特権レベルを上げることによりこれを保護できる。

ITRON/286における各種モジュールは次のような特権レベルに配置される。

特権レベル0　・・・・・　OSコード、OSデータ、OSスタック
特権レベル1　・・・・・　割り込みハンドラ、例外処理ハンドラ、等
特権レベル2　・・・・・　未使用
特権レベル3　・・・・・　ユーザタスクコード、データ、スタック

3．4　タスク管理機能

80286では、これまでソフトウェアで行っていたタスク管理機能の一部をハードウェア化している。ITRON/286はこれを利用して高速なタスクスイッチを実現している。

3．4．1　ＴＳＳとＴＲ

80286は、タスクの管理のために、タスクステートセグメント（TSS）を規定し、これを利用してタスク管理を行う（図７）。TSSには、実行を中断しているタスクのレジスタ等の状態を保持する領域や、特権レベルによって切り換わるタスクのスタックの情報などがある。TSSは、メモリ上に置かれ、GDT中にTSSディスクリプタとして定義されている。

現在実行中のタスクは、タスクレジスタ（TR）中に設定されているそのタスクのTSSを指すセレクタ値によって示され、タスクスイッチ時にはこれが自動的に切り換えられる。メモリ空間の切り換えは、TSS中のLDTセレクタの値が自動的にLDTRにロードされることにより行われる。

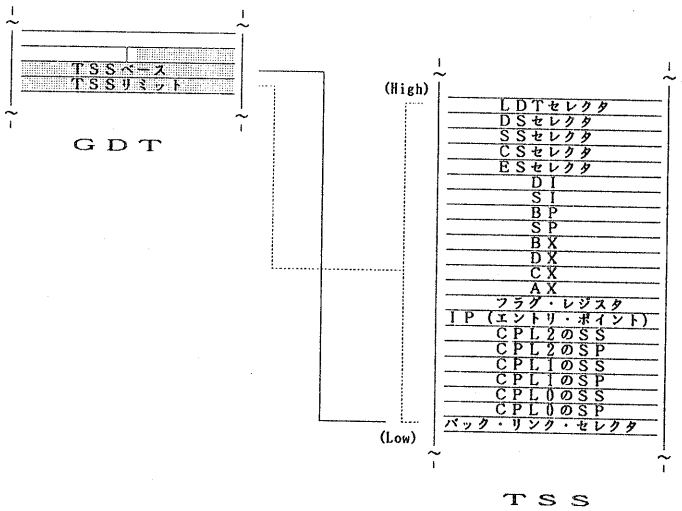

図７　タスクステートセグメント（ＴＳＳ）

80286の行うタスク管理の基本機能は、このTSSの情報をもとにタスクスイッチを１命令で自動的に行うことである。すなわち、飛び先番地にTSSのセレクタを指定したJMP命令を実行すると、TRレジスタで示されるそのとき実行中のタスクの情報がTSSにセーブされ、目的タスクの情報が指定されたTSSよりロードされ、レジスタやタスクのメモリ空間が切り換わる。

３．４．２　タスクスケジューリング

ITRON/286では、80286のタスク管理機能を利用し、高速なタスクスイッチを実現している。これにより、高度なメモリ管理機能を取り入れているにもかかわらず、従来と同等以上の機能を実現できる。

ITRON/286のスケジューリング処理は、次のような処理ステップにより実行される。

１）TCB（タスクコントロールブロック）を含む各種オブジェクトの操作
２）オブジェクトの変更にともなうタスクのスケジューリング
３）スケジューリングの結果によるタスクのディスパッチ処理

ITRON/286では、３）の部分をハード化するとともに、２）のスケジューリングのアルゴリズムを最適化することにより、性能向上を図っている。

４．おわりに

リアルタイムOS等のマルチタスクシステムにおいて、あるタスクからOSや他のタスクが破壊されることを防ぐのは、システムの信頼性の面からきわめて重要である。タスク間のメモリ破壊は最も発見されにくいバグの１つであるが、メモリ保護によってこの発生を防ぐことにより、開発効率は飛躍的に向上する。このことは特に大規模システムで重要である。したがって、特にメモリ管理機能の強化されたITRON/286は今後より広まるであろう大規模なリアルタイムシステムの開発に最適であるといえる。

また、ITRON/286では、高度なメモリ管理機能に柔軟に対処するため、システムコールはプリミティブな機能にとどめている。しかし、そのため操作が煩雑になるという面があるので、現在この点を考慮して、いくつかのシステムコールの機能を複合した、MMU複合システムコールの導入を予定している。参考までにMMU複合システムコールを表２に掲げておく。

最後に、ITRON/286の仕様化に当って、終始ご指導を賜っている東京大学の坂村健先生に謝意を表します。

参考文献

[1] 坂村　健：ITRONアーキテクチャ，情報処理学会第29回全国大会論文集（1984）
[2] 梅田　望夫，坂村　健：ITRON/68K，情報処理学会第29回全国大会論文集（1984）
[3] 松為　彰，坂村　健：ITRON/86，情報処理学会第29回全国大会論文集（1984）
[4] 長谷川　薫，高橋　久，門田　浩：V20/30用リアルタイムOSのインプリメンテーション，情報処理学会第29回全国大会論文集（1984）
[5] 長谷川　薫，高橋　久，門田　浩：V20/30とITRON，情報処理学会計算機アーキテクチャ研究会資料（1985）
[6] 坂村　健：BTRONスーパー・パーソナル・コンピュータ，情報処理学会計算機アーキテクチャ研究会資料（1985）
[7] 斎野　亨：「BTRON」登場，日経コンピュータ，1985-4-29 号
[8] MBL80286 ハードウェアマニュアル，富士通株式会社
[9] MBL80286 プログラミングマニュアル，富士通株式会社
[10] MBL80286 オペレーティングシステムライターズガイド，富士通株式会社
[11] 瀬谷　健二：80286アーキテクチャ詳細，インターフェイス（Nov.1985）
[12] 平野　浩介：80286ソフトウェア入門，インターフェイス（Nov.1985）