**Introductory of LSI Device 12/5/1984 – 12/7/1984**

## 開催要領

■セミナーコード
B－2501

■開催日
昭和59年12月
5日(水)～7日(金)

■受講料
1名につき50,000円
(昼食・テキスト代含む)

■申込・問い合せ先
日本工業技術センター
〒102 東京都千代田区飯田橋
3－11－13 豊国ビル
TEL 03(262)1962
FAX 03(262)1908
(自動受信)

申込要領は裏面記載

## セミナーポイント

Ⅰ. LSIデバイス基礎
沖電気工業
相良岩男氏

Ⅱ. LSIメモリデバイス
東芝
佐々木逸夫氏

Ⅲ. マイクロプロセッサ並びにマイクロコンピュータ
日本電気
小口哲司氏

Ⅳ. ゲートアレイ技術
富士通
本間明氏

Ⅴ. LSIの信頼性と故障対策
松下電子工業
渡辺紘氏

Ⅵ. A/D, D/A変換技術
シンキー
長橋芳行氏

技術系新入社員，新規配属者，周辺技術者等初学者を対象とした

## エレクトロニクス技術3日間集中基礎講座

# LSIデバイス入門

沖電気工業㈱ 相良岩男 監修

エレクトロニクス技術の源であるIC，LSIは他のデバイスに比べて非常に高機能なデバイスであること，常に非常なスピードで進歩していることなどから，電子機器システムの性能の大部分は，使うIC，LSIによって決定される。そこで本講座では，ユーザのためのLSIデバイス講座として，特に主要なLSI，マイクロプロセッサ，ICメモリ，コントロール用LSI，インタフェース用LSIなどについてその本質，特長，使用上の留意点，応用回路設計のポイントについて解説する。

## セミナーポイント

各種LSI（特にマイクロプロセッサ廻り）の物性，基本回路と応用法について素人にも解るように解説。

### 受講対象

IC，LSI，マイクロプロセッサのユーザー技術者及び半導体関連ディーラーの営業技術者

No. B-2501

LSIデバイス入門集中基礎講座

## 12月5日(水)

(10：00～13：00)
**Ⅰ. LSIデバイス基礎**
- LSIを理解するための物性
- バイポーラの構造と特性並びに基本回路
- MOSの構造と特性並びに基本回路
- LSIプロセス(N-MOSとC-MOS)と設計
- LSI実装技術

相良 岩男氏
沖電気工業㈱
電子デバイス事業部
総合技術部
副部長

(14：00～17：00)
**Ⅱ. LSIメモリデバイス**
- D-RAMの概要と特性
- S-RAMの概要と特性
- EP-ROMの概要と特性
- EE-ROMの概要と特性
- MASK-ROMの概要と特性
- メモリの技術動向

佐々木逸夫氏
㈱東芝
第2集積回路事業部
技術部
設計第2課

## 12月6日(木)

(10：00～13：00)
**Ⅲ. マイクロプロセッサ並びにマイクロコンピュータ**
- プロセッサ・コンピュータの基礎
- 4ビットコンピュータの特性
- 8ビットプロセッサとコンピュータの特性
- 16ビットプロセッサの特性
- 32ビットプロセッサの特性
- 周辺用コントローラ(CRT，LCD，FDD)

小口 哲司氏
日本電気㈱
超LSI開発本部
システム部
主任

(14：00～17：00)
**Ⅳ. ゲートアレイ技術**
- ゲートアレイの種類と特長
- ゲートアレイの製造工程
- ゲートアレイの電気的特性
- スタンダードセルなど次世代のセミカスタムLSI

本間 明氏
富士通㈱
半導体営業推進部
部長付

## 12月7日(金)

(10：00～13：00)
**Ⅴ. LSIの信頼性と故障対策**
- LSIの信頼性と試験法
- 各デバイスの信頼性(メモリ，マイコン etc)
- LSIの故障メカニズムと対策
- LSI使用上の留意点

渡辺 紘氏
松下電子工業㈱
IC事業部
品質技術部
課長

(14：00～17：00)
**Ⅵ. A/D，D/A変換技術**
- A/D，D/A変換器の分類
- A/D変換の周辺技術
- A/D，D/A変換回路の選定と設計の要点
- A/D，D/A変換器
- 応用回路の設計

長橋 芳行氏
㈱シンキー
取締役 技術部長

## 申込要領

■申込方法
最終頁の申込書のコピーに所要事項をご記入の上，郵送下さい。また電話での申込みも受付けますが，その場合も後日申込書を郵送下さい。ファクシミリ(自動受信)の申込みも受付けます。

■支払方法
受講料のお支払は，開催日前日までに，銀行振込又は現金書留でお願い致します。受講料はご返金は致しません。ご都合の悪い場合は代理の方がご出席願います。

取引銀行
住友●神田駅前(当) 256993
三菱●神田(㈱)4048297
第一勧業●飯田橋(㈱)1237359

■セミナー会場
日本工業技術センター
研修室 (地図は最終頁)
東京都千代田区飯田橋3-11-13
豊国ビル5F ☎03(262)1962
交通 ●国電/総武線「飯田橋」下車徒歩5分
●地下鉄/東西線・有楽町線「飯田橋」下車徒歩5分

会場が変更になる場合は受講証送付時にお知らせします。

**Microprocessor & Microcomputer (Tetsuji Oguchi) : from 10:00 – 13:00 on 12/6/1984 (Thu)**

エレクトロニクス技術

集中基礎講座

# LSIデバイス入門

監　修

相良　岩男

日本工業技術センター

## 執筆者・執筆分担一覧（アイウ順・敬称略）

| | |
|---|---|
| 小口哲司 | 日本電気㈱〔第Ⅲ章〕 |
| 相良岩男 | 沖電気工業㈱〔第Ⅰ章〕 |
| 佐々木逸夫 | ㈱東芝〔第Ⅱ章〕 |
| 長橋芳行 | ㈱シンキー〔第Ⅵ章〕 |
| 本間明 | 富士通㈱〔第Ⅳ章〕 |
| 渡辺紘 | 松下電子工業㈱〔第Ⅴ章〕 |

# 第Ⅲ章　マイクロプロセッサ並びに<br>マイクロコンピュータ技術

中央処理装置（CPU）や演算論理装置（ALU）のみを含みROM（読み出し専用メモリ）やRAM（任意読み書き可能メモリ）などで構成される主記憶部分や周辺装置制御回路などを外付けすることによってディジタル制御装置を構成することができるマイクロプロセッサや，上記した機能をすべて1個のチップに集積したマイクロコンピュータは，集積回路の回路／プロセス／量産技術の飛躍的な進歩とともに高度な機能を持ったチップが安価に供給されるようになってきた。既に，4ビットのマイクロコンピュータはVTRを筆頭とする家庭電気製品に組み込まれており，大量に使用されている。

マイクロプロセッサは外付けメモリ素子などと組み合わせ，2個以上のチップにより小型のコンピュータを構成するところから「マルチチップ•マイコン（マイクロコンピュータ）」と呼称することがある。また，マイクロコンピュータは，1個で小型コンピュータとしての機能を提供するところから，「シングルチップ・マイコン」と呼ぶことがある。以下，これらを総称して，単に，「マイコン」と表現することにする。

## 1. 集積回路技術の進歩

電子技術全般の進歩は集積回路技術の進展にその歩調をあわせているといっても過言ではない。集積回路事業は「金食い虫」であると一般に呼ばれている。わずか数ミリ角のチップを作るのに，膨大な額の投資を行なわなくてはならないからである。製品開発時には，回路のシュミレーションやフォトマスク作成などに大型コンピュータを駆使したCAD（COMPUTER AIDED DESIGN）手法がとられる。マスク露光，拡散，組み立てなどの製造装置，ウエハー処理や組み立て終了後の集積回路を検査するLSI（LARGE SCALE INTEGRATION）テスターなどの検査装置などは，集積回路の規模が大きくなり，そのマスクパターンの精細度が高くなるに従って，より複雑な処理能力を持ったものが必要となる。装置の更新は日常茶飯事であり償却期間は益々短くなってきている。

このような背景から，設計した集積回路が少量しか販売できなければ，開発費を回収することなど到底できない。集積回路技術が現実の装置に応用され進展するためには，集積回路を大量に使用できる環境が必要であった。1965年以降，この環境を提供してくれたのが電卓（電子式卓上計算機）である。当時の電卓は，TTL（TRANSISTOR TRANSISTOR LOGIC）やMOS（METAL OXIDE SEMICONDUCTOR）構造のSSI（SMALL SCALE INTEGRATION）などで組み立てられており，販売価格は数10万円もしていた。ところが，1台の電卓が数個のLSIで構成され，さらには1個のLSIだけで構成されるに至り，1970年代中頃には，電卓販売価格は1万円を割るまでに至った。

Translation of chapter 1.

## 1. Advances in integrated circuits

It is no exaggeration to say that the progress of entire electronic technology is keeping in step along with the progress of integrated circuit technology.

Integrated circuit technology is commonly called a “money pit”.

This is because a huge amount of money must be invested to make such a small mili-meter square chips. During LSI (Large Scale Integration) product development, a CAD (Computer Aided Design) approach that takes full advantage of mainframe is indispensable for logic simulation and photo mask creation.

Manufacturing equipment such as “photo mask exposure”, ”wafer diffusion”, “die encapsulation” and inspection equipment such as LSI testers that inspect integrated circuits after wafer processing and encapsulation must have more complex processing capability according to the rapid enhancement of both the scale of integrated circuits and the precision of photo mask patterns.

The higher the density, the more complex processing power is required. Updates of such equipment are a matter of common practice and the amortization period is becoming shorter and shorter.

Due to such background, if a small number of integrated circuits designed can only be sold, the development costs cannot be recovered.

In order for integrated circuit technology to be applied to actual devices and progressed, an environment in which integrated circuits could be used in large quantities was necessary. Since 1965, the desktop calculator has provided the environment.

At that time, desktop calculators were assembled with TTL (Transistor Transistor Logic) or SSI (Small Scale Integration) with MOS (Metal Oxide Semiconductor) structure and the sales price was several hundred thousand yen. <multiple $667s (in exchange rate of 150 yen/$)>
However, one desktop calculator consisted of several LSIs and finally consisted of a single chip LSI by the mid-1970s. The sales price of the desktop calculator had fallen below 10,000 yen. <below $33 (in exchange rate of 300 yen/$ in 1975)>

Once the development is completed and the cost spent for the development is recovered, the production cost will be amazingly less if the product can be produced and sold in high volume.

The sales price of the integrated circuit seems to be determined taking into consideration of complex factors such as the yield of each product that depends on the die size and manufacturing process, the package material in use, the richness of integrated functions, the amount of supply to the market, and the existence of competitors.

As equipment becomes less expensive due to the supply of inexpensive integrated circuits, it is natural that the next goal is to provide integrated circuits with higher functions at comparable prices.

By repeating such a favorable cycle, the electronics industry has evolved along with the progress of integrated circuits.

いったん開発が終了し，その開発に要した費用が回収されてしまえば，大量に生産し販売できる製品であれば，その生産コストは非常に安価なものとなる。集積回路製品の価格は，チップ・サイズや製造プロセスに依存する製品個々の歩留まり，使用しているパッケージ材料，集積している機能の豊富さ，市場への供給量，コンペティタの有無など複雑な要素を勘案して決定されているようである。安価な集積回路の供給により装置が低廉化されると，より機能の高い製品を同程度の価格で提供することが次の目標となってくるのは当然のことである。このような好ましい循環を繰り返すことにより，集積回路の進歩とともに，電子産業は発展してきた。

## 2. マイコンの進歩

1972年にインテル社から発表された４ビット・マイコン4004が世界初の製品であると言われている。この製品の開発の端緒となったものは電卓用のLSIの開発であったが，その汎用性を高めようとした結果，本製品が生まれたと言われている。４ビットと称しているとおり，4004が持つアドレスおよびデータ・バス幅は，いずれも４ビットであり，TTLと同じ16ピンDIL（DUAL IN LINE）パッケージに内蔵されていた。このため，外部メモリからの命令の取り込みやデータの格納におけるスピードは極端に遅かった。電卓用LSIは当時すでに，ワン・チップ化されており，電卓用あるいは同種の専用化された用途に4004のような製品が使用される余地は全くなかった。この結果，マイコンは汎用性のあるディジタル処理における製品応用分野で伸長していくことになる。

インテル社が発表した最初の８ビット・マイコン8008の開発後，その改良品であり後に８ビット・マイコンの標準品ともいわれる8080が1974年に発表された。しかしながら，8080の命令はユーザを満足させるほど魅力的なものではなかった。さらに，周辺回路に使用されるTTLに供給する＋５Ｖ電源の他に＋12Ｖおよび－５Ｖの計三種の電源を必要とした。これは後述するようにLSI内の論理ゲートに使用されている負荷MOSとしてエンハンスメント型のトランジスタが使用されていたからである。当時，MOSトランジスタのスレッショルド電圧（VTH）制御にイオン・インプランテーション技術が導入されつつある端境期であり，大量に量産すべき8080への適用が間に合わなかった。

その後，インテル社は8080に外付けせねばならなかったクロック・ジェネレータ8224やシステム・コントローラ8228を不要とし，供給者／需要者の両サイドにおいて量産性を高めた製品として8085を開発，また，ザイログ社は8080の命令を拡張し，ダイナミックRAMに対するリフレッシュを有効に行なう機能を内蔵した製品Ｚ80を発表，ようやくユーザが納得できるような８ビット・マイコンが誕生した。

プリンタ制御やキーボード制御などのように小規模なメモリだけで制御可能な分野に対するマイコンの応用を容易にするため，ROMやRAMを内蔵したシングルチップ・マイコンが開発されるようになった。インテルの８ビット・マイコン8048はその代表な製品である。４ビットの領域では，十進演算機能を重視した応用範囲のゆえにシステムそのものが一般に小さいものが多く，マルチチップ構成とすることは殆んどない。そのため，シングルチップ・マイコンの独壇場である。

シングルチップ・マイコンはマルチチップ・マイコンと較べて使用者が限定されてしまい，その品種も多く，さらに，品種間での命令の互換性が殆んどないために，一般に良く知られていないようであるが，4ビット・シングルチップ・マイコンはマイコンのなかでも最も需要が多く，生産量が多い。マイコン生産量の60パーセント程度を占める。電卓用LSIで培った信頼性が高くコストの安い製品を開発する技術の裏付けによって，特に，日本の集積回路メーカが得意とする分野である。各社各様の製品が発表されている。

16ビット・マイコンとしては，1978年にインテル社から発表された8086が代表的な製品であるといえる。現在，パーソナル・コンピュータの分野では，殆んど総ての製品に使用されている。64Kバイトのバウンダリにメモリ使用が分割されてしまうことや，命令そのものが8080との互換性を重んじるあまりに，それほど強力でなかったり，命令の整理が充分になされておらずソフトを書く際の手間が多いなどの理由から，モトローラ社が後になって発表した68000に移行するユーザもある。ただ，マイコン製品はソフトウエアの開発環境が満足すべきものであるか，利用環境が豊富であるか否かに依存する面もあり，その製品が持つ機能／性能などハードウエアのみによってユーザに受け入れられるわけではないところが興味深い。

## 3. マイコンの設計手順

マイコン用集積回路の開発は記憶素子の開発などとは異なり，多数のランダム・ロジックを内蔵していることや，最終的にその製品をシステムに組み込んだときに機能を十二分に発揮できるかどうか，あるいは，ユーザに受け入れられるかを前もって細かく評価せねばならぬなど，設計の複雑さは非常に高いといえる。一般に，次のような手順を踏む。

(1) システム設計……集積回路が内蔵する機能仕様をシステム的な見地から検討し，決定する。ソフトウエア開発環境の増強が必要であれば，ソフトウエア開発ツール，例えば，アセンブラなどの開発を促進する。

(2) 論理設計……機能仕様を実現する論理回路をゲートレベル若しくは機能ブラックボックス・レベルで設計する。

(3) 回路設計……マスクレイアウト設計における容易性を考慮しつつ，トランジスタ・レベルでの論理回路を設計する。論理設計とかなりな部分で内容が重複する。

(4) コンピュータ・シュミレーション……コンピュータに論理回路を記憶させ（接続情報），ある入力パターンを与えたときの出力パターンや内部回路の論理変化をモニタすることにより回路動作の良否を判断する。LSIを検査する際に使用するテストパターンを作成することにも使用する。

(5) TTLシュミレーション……モックアップとかブレッドボードと呼ぶ。集積回路の内部回路をTTLやCMOS SSIにより実現し，回路動作を把握する。コンピュータ・シュミレーションと比較して，回路動作確認のTATが少なくて済むが，モックアップ自身を正常動作させるに至るまでが大変であるという面も併せ持つ。最近では回路規模が大きくなってきたためゲートレベルで総ての回路をシュミレートすることは，ますます，難し

くなってきている。

(6) マスクレイアウト設計……回路図を基にしてトランジスタを配置／配線する。通常，実際のチップサイズの400倍とか1000倍の大きさの図面に，各マスク工程のパターン図を作成する。2.5ミクロンあるいは1ミクロン単位の細かさで設計することになる。最終的にチップの大きさや製造の電気的特性を決定する重要な設計工程である。

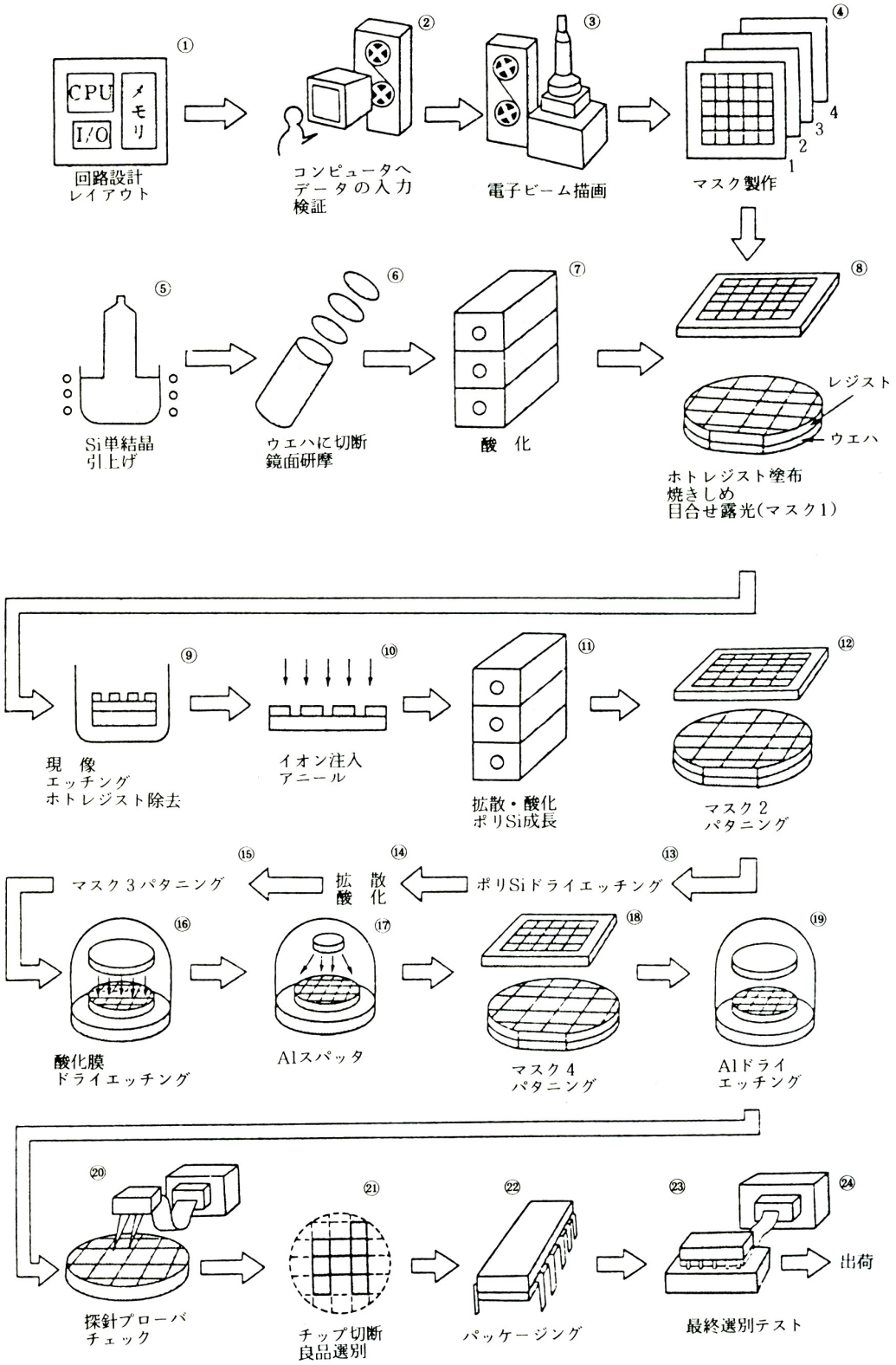



**図3.1　集積回路の製造工程**

(7) 試作……使用ウエハー処理工程や組み立て工程などの量産時における問題点をたたきだす。

(8) 検査……LSIテスタで製品の検査ができるようにテストパターンやテストプログラムを作成する。製品とLSIテスタとのインタフェースをとるため，テストボードやプローブカードを作成する。

(9) 評価……LSIテスタによって製品の電気的特性評価を行ない，その結果を電気的スペック・シートに反映する。実際のシステム上で動作させ，種々の動作モード上でも正常に動作することを確認する。

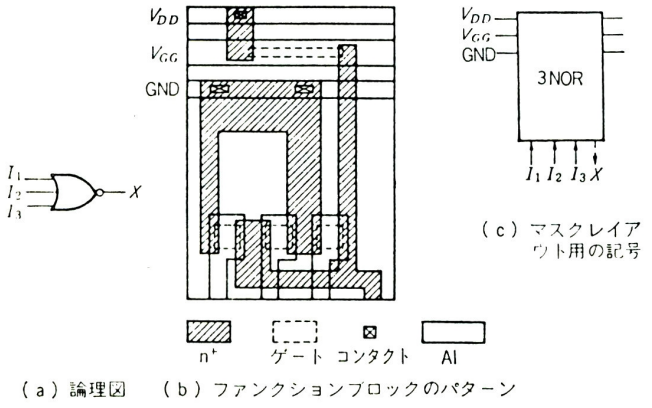



図 3.2　ビルディング・ブロック

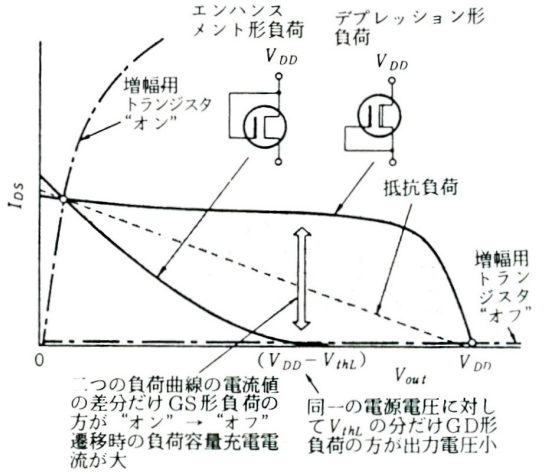
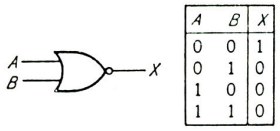
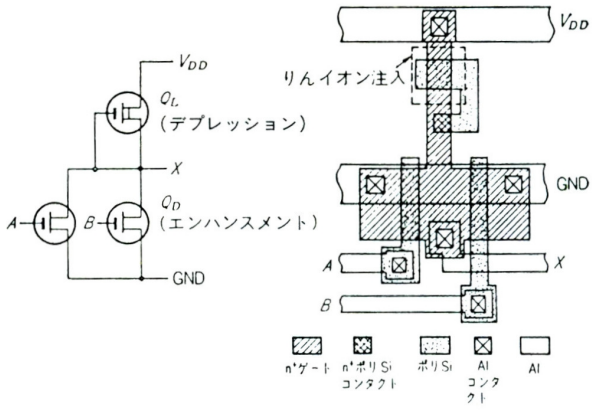



図 3.3　エンハンスメント負荷とデプレッション負荷

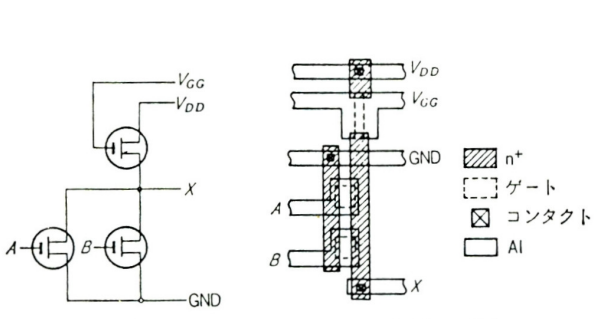



| A | B | X |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

（a）論理記号および真理値表




（c）nチャネルSiシリコンゲートE/D MOSを用いた場合




（b）nチャネルAlゲートE/E MOSを用いた場合

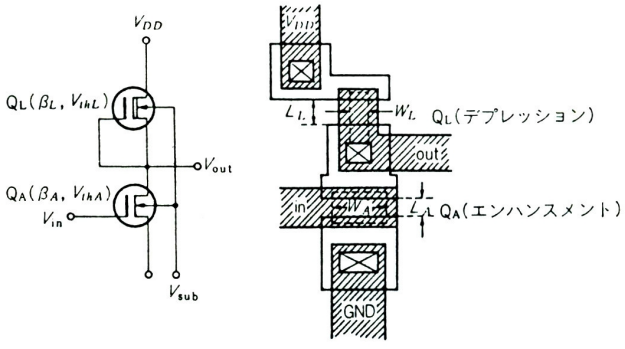



図3.4　各種MOSトランジスタ

このように，１個の製品を開発するだけでも相当な手間と時間を必要とする。さらに各工程間において，緊密な設計結果のフィードバックがなされなければ良い製品は生まれてこない。最近では，ビルディング・ブロック方式によりマスクレイアウト設計工程を簡略化し，上記(1)から(5)の部分とそれ以降の部分とをシステム設計者側と集積回路設計者側とに分業化するとともに，標準化によって(6)以降の工程の正規化を行なったゲートアレイが特殊用途向けとして使用されるようになってきている。

## 4. マイコン用LSIのデバイス設計

半導体集積回路に使用されるデバイスは，正孔と電子の２種のキャリアによって電流制御を行なうトランジスタにより構成するTTLに代表されるようなバイポーラ型と，多数キャリア（Ｐチャンネル MOSならば正孔，Ｎチャンネル MOSならば電子）のみにより電流制御を行なうMOSに代表されるモノポーラ型に大別できる。製造工程が比較的簡単で集積度を高くすることができ，低消費電力／低価格などの要因からマイコン用LSIとしてMOS型トランジスタを集積した集積回路が一般に使用されている。

MOS型トランジスタはドレイン，ゲート，ソースの３極によって構成され，そのgmはW／L（W：トランジスタのゲート幅，L：ゲート長）に比例する。このことは即ち，Lを小さくすれば，より小さなトランジスタによって同じgmを得ることができることを意味している。さらに，各ゲート間には電流負荷は存在せず，容量負荷しか存在しない。従って，Lが小さくなればその容量が極端に減少するため，高速化が期待できる。しかしながら，Lを小さくするにはマスク関係工程（マスク作成，目合わせ，露光，フォトレジスト，エッチングなど）の精度向上，トランジスタ性能のバラツキを抑える方法（VTH制御，ソース・ドレイン間の突き抜け防止やトランジスタ領域とそれ以外の領域との間の分離など）などが確立されね

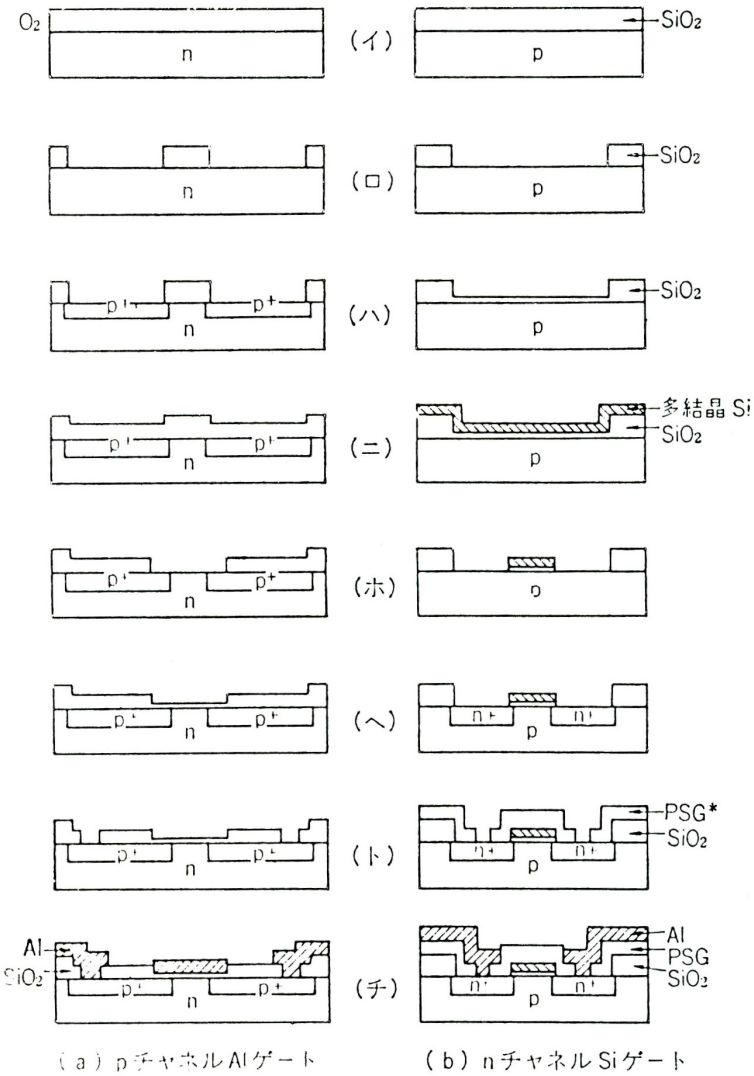



**図 3.5　マスク工程**

ばならず，一朝一夕には解決できる問題ではなかった。

1970年当時は10ミクロンのゲート長のトランジスタが使用されていた。VTHの制御はウエハーの不純物濃度やゲートの酸化膜厚（Tox）だけによって制御されていたため約３Ｖと高く，負荷MOSにエンハンスメント型トランジスタを使用し，一電源タイプのものでは－24Ｖ（Ｐチャンネル型）の電源を必要としていたため消費電力が非常に大きかった。

MOS型トランジスタを広汎に量産に供されている要素パラメータ別に分類すると次のようになる。これらを相互に組み合わせることにより，種々なMOS型トランジスタが供給されている。

(1) チャンネル；Ｐチャンネル，Ｎチャンネル，相補型（CMOS)。

(2) ゲート；メタルゲート（アルミ），シリコンゲート。

(3) トランジスタ動作モード；エンハンスメント型，ディプレッション型。

電卓用LSIはＰチャンネル・メタルゲートＥ／Ｅ（ゲートMOS，負荷MOSともにエンハンスメント型MOS使用）で製造されていたが，次第に，Ｅ／Ｄ型に発展し，さらに，表示素子として液晶が使用されるようになると低消費電力化のためCMOS（COMPLEMENTARY MOS）構造を採用するようになる。イオン•インプランテーション技術はMOSトランジスタのVTH制御に革命を起こした。この技術の普及により，精度良くVTHのシフトができるようになり，ディプレッション型トランジスタのITDやエンハンスメント型トンジスタのVTE，CMOSにおけるＰチャンネルおよびＮチャンネルのトランジスタのVTP，VTNを自由に調整し用途に合った集積回路を供給できるようになってきた。この結果，現在では，マイコン用LSIや記憶素子に供給する電源はTTLに供給する電源と同じ＋５Ｖだけで良いようになっている。

| | | |
|---|---|---|
| 素子寸法 | $t_{ox}$，$L$，$W$ | $1/\kappa$ |
| ドーピング濃度 | $N_A$ | $\kappa$ |
| 電　圧 | $V$ | $1/\kappa$ |
| 電　流 | $I$ | $1/\kappa$ |
| 容　量 | $\varepsilon A/t$ | $1/\kappa$ |
| 伝搬遅れ／回路 | $CV/I$ | $1/\kappa$ |
| 消費電力／回路 | $VI$ | $1/\kappa^2$ |
| 論理ゲート面積 | $A$ | $1/\kappa^2$ |
| 集積度 | $1/A$ | $\kappa^2$ |
| 消費電力密度 | $V \cdot I/A$ | 1 |
| 遅れ，電力積 | | $1/\kappa^3$ |
| 配線抵抗 | $R=\rho L/WT$ | $\kappa$ |
| 相対電位降下 | $IR/V$ | $\kappa$ |
| 配線による$CR$遅れ | $RC$ | 1 |
| 配線の電流密度 | | $\kappa$ |
| コンタクト抵抗 | $R_C$ | $\kappa^2$ |
| 接触電圧降下 | $V_C$ | $\kappa$ |
| 相当コンタクト電位降下 | $V_C/V$ | $\kappa^2$ |
| 配線の$RC$時定数／素子遅れ | | $\kappa$ |

**図3.6　スケーリング則**

マイコン用LSIには当初から一部の製品を除いてＮチャンネル・シリコンゲートＥ／Ｅ若しくはＥ／Ｄが使用されてきた。シリコンゲートとすることにより，トランジスタをマスク目合わせ上，セルフアライメント式に作成できるため，特性のバラツキが軽減できるとともに，ゲート容量が減少するため高速化が画れることによる。さらに，拡散層／アルミ層に加えてポリシリ層も配線領域に使用できることから，例えば，電源アルミ配線層の下にもトランジスタを形成するこ

とができるなど，集積密度を大幅に高めることができることも大きな利点である。現在，チャンネル長が2ミクロンを切る製品が量産されるようになってきており，集積度が非常に高くなってきている。1個の集積回路の消費電力が1ワットを超えると熱抵抗の小さいセラミック・パッケージにチップを格納せねばならなくなる。セラミックを使用したとしても2－3ワット程度が限度である。さらに，MOS型トランジスタは高温になるにつれて，その特性，特にスイッチングが急速に劣化する。

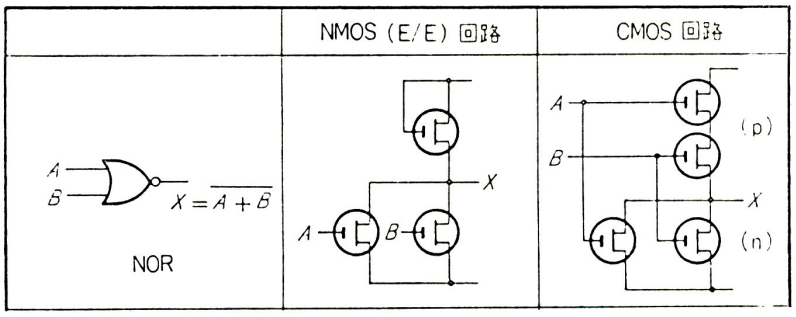



図 3.7　NMOSとCMOS

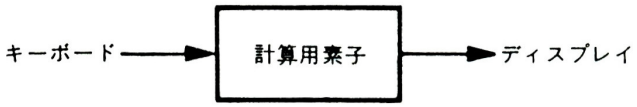



図 3.8　電卓の構成

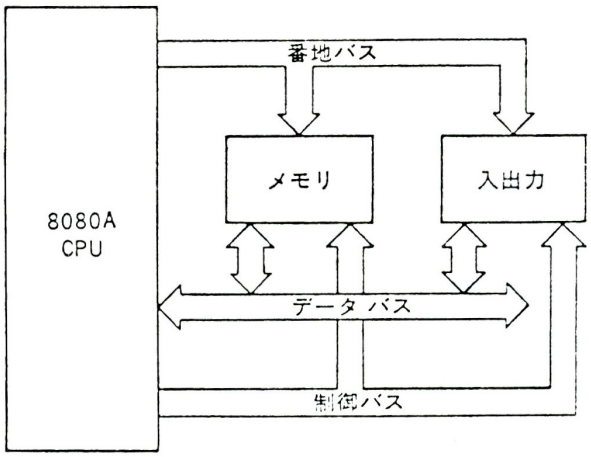



図 3.9　情報処理装置の構成

CMOS構造トランジスタを使用して論理回路を構成すれば，電源ラインからグランドへ貫通する電流パスは，ほんの僅かな時間のみとなるとともに，信号レベル変化時におこる容量に対する充放電による電力消費が主となるために，消費電力が大きく軽減できる。今後，開発されるマイコン用LSIの殆んど総てがCMOSにより製造されていくことになるであろう。

## 5. マイコンのシステム構成

情報処理装置は演算制御装置を中心として，入力装置，記憶装置により構成される。身近にある電卓を例にとれば，入力装置がキーボード，出力装置が液晶や蛍光表示管などの表示装置，演算制御／記憶部はLSI自身に内蔵されてしまっている。さらにキー走査用の信号やキーからの信号の入力，蛍光表示管走査信号あるいはドライバまでオンチップ化されているのでユーザは単に，キーボードと蛍光表示管だけを電卓用LSIに接続するだけでよい。

マイコンを使用したシステムにおいて，上述した各装置は次のような分類ができるであろう。

(1) 入力装置……キーボード，データ・タブレット，マウス，ライトペン，紙テープ・リーダ，通信回線など。

(2) 出力装置……CRT，液晶，プラズマ・ディスプレイなどの表示機器，プリンタ，紙テープ・パンチャ，通信回線など。

(3) 記憶装置……ROM，RAMなどの主記憶，フロッピーディスク，ハードディスク，MTなどの補助記憶。

(4) 演算制御装置……マイクロプロセッサ，浮動小数点演算器，DMA制御，割込制御など。

本稿では，上記各装置制御に関するマイコン用LSIの実際について，

(1) マイクロプロセッサ

(2) マイクロコンピュータ

(3) 周辺コントローラ

について，具体例をもとにしつつ，順次，説明を加えていくことにする。

これらのLSIは周辺回路を構成するTTLとの入出力インタフェースを直接行なえるように，その電気的スペックが決定されている。

## 6. マイクロプロセッサ用LSI

外部記憶装置や周辺機器との間でデータのやりとりを行なうために，マイクロプロセッサは次の3種のバス信号を入出力することができるようになっている。

(1) アドレスバス

(2) データバス

(3) 制御バス

通常，データバス幅によってプロセッサのビット数を定義しているが，プロセッサ内部のデータバス幅，あるいは，アキュムレータのビット数による場合もある。例えば，インテル社の16ビットマイコン8086の外部バス幅を8ビットに狭めた製品

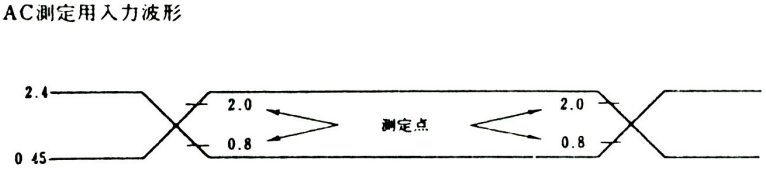



図 3.10　TTLコンパチビリティー

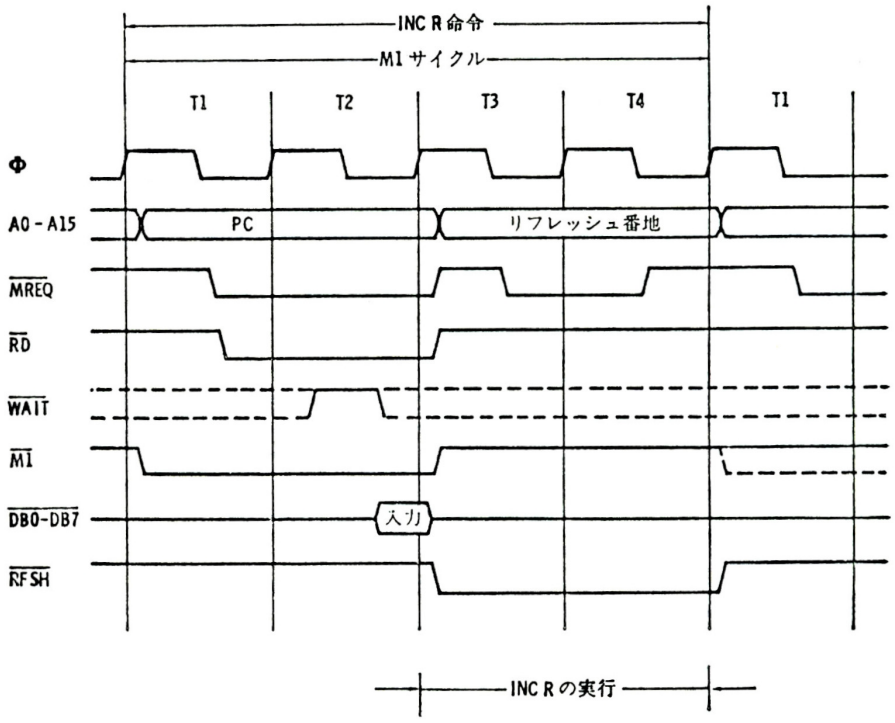



図 3.11　Z80における命令フェッチ

8088は一般に8ビット・マイコンと称しており，モトローラ社の16ビット・マイコン68000は32ビット・マイコンと呼ばれることがある。

マイコン・アーキテクチュアの流れは大きく分けて，インテル系とモトローラ系とがある。インテル系の場合，3ないし4クロックで記憶素子や周辺制御素子との間でのデータのアクセスを終了するが，モトローラ系の場合は1クロックで終了する。即ち，クロックの高レベル時にアドレスを出力し，低レベル時にデータを取り込む。このタイミングを判別させるために，CPUに供給するクロックが周辺制御素子へも供給されることになる。モトローラ系のCPUはクロック周波数が低いにも拘らず処理速度が速いような錯覚を与えるのは，この違いによる。

マイコンの命令サイクルは，次の4種の動作で構成される。

(1) 命令の取り込み（フェッチ）

(2) 命令の実行（エ

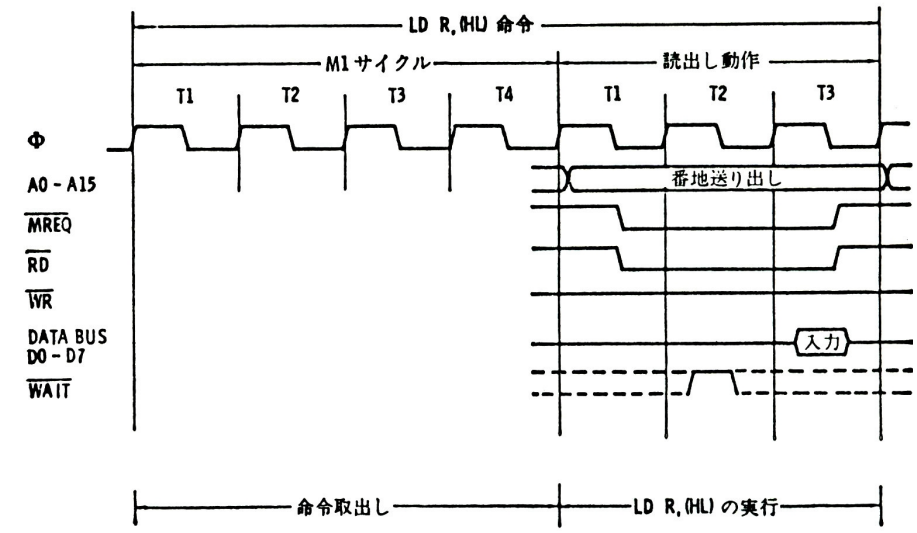



図 3.12　Z80におけるデータ・リード

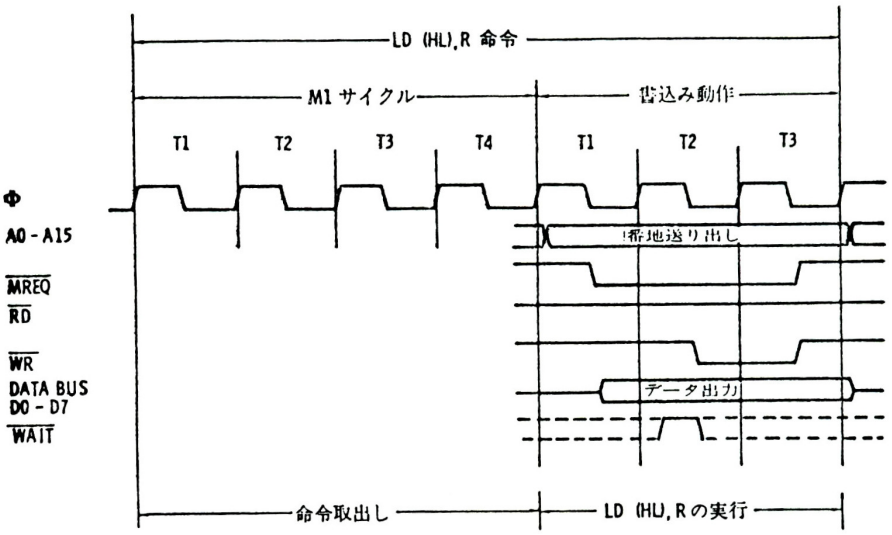



図 3.13　Z80における結果のストア

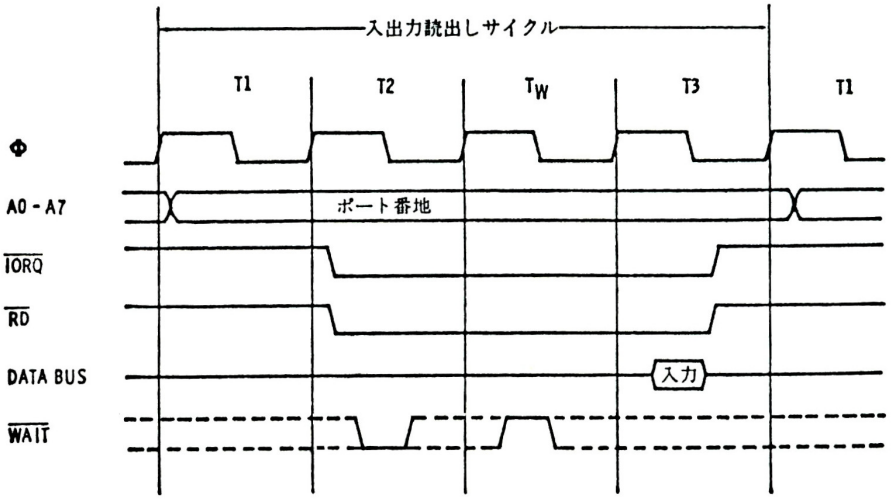



図 3.14　Z80Uにおける I/O命令とウエイト

クゼキュート）

(3) データの取り込み（リード）

(4) 処理結果の格納（ストア）

まず，ROMなどに格納されている命令コードを読み取り，命令を解釈し実行する。マイコン内部のレジスタにその処理結果が格納されるのであれば，(1)，(2)だけで命令サイクルを終了する。データをRAMなどの外部記憶から読み出して使用したり，結果をRAMにもどす場合には，(3)および(4)のサイクル（リード／ライト動作）が追加実行される。

Z80では，フェッチに4クロックを割り当てているが実際に命令コード読み出し動作を実行しているのは，この命令サイクルの初めの2クロックだけである。次の2クロックは，主記憶としてダイナミックRAMを使用した場合に有効性を発揮するメモリ・リフレッシュ・サイクルが挿入される。エクゼキュートに要する時間は命令によって異なる。結果のストアは3クロックである。命令フェッチ2クロッ

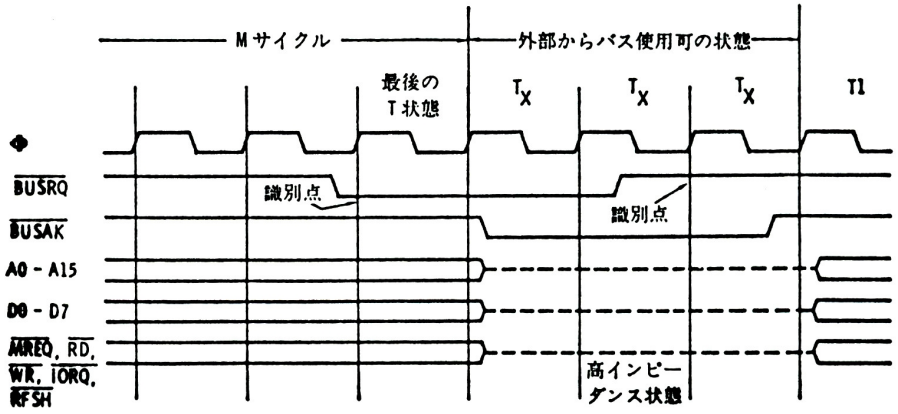



図 3.15　Z80におけるホールド要求信号

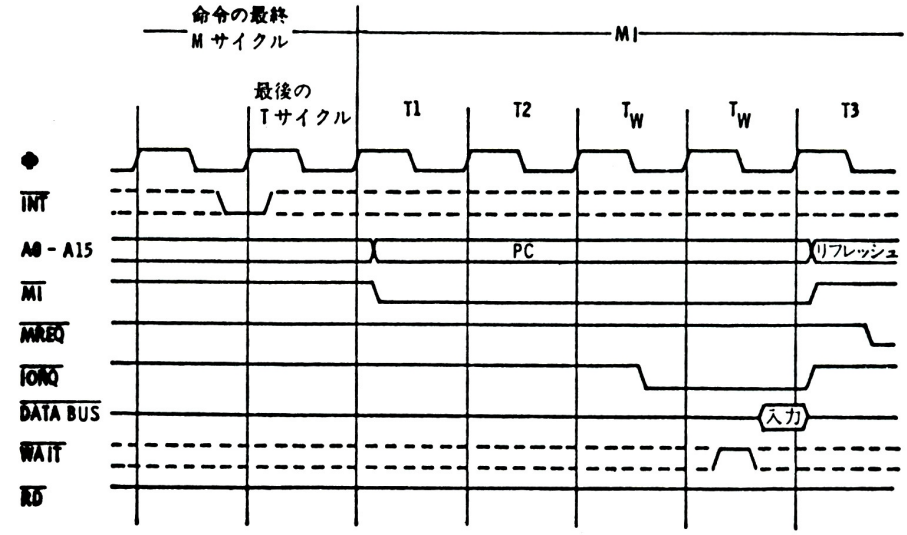



図 3.16　Z80における割込要求信号

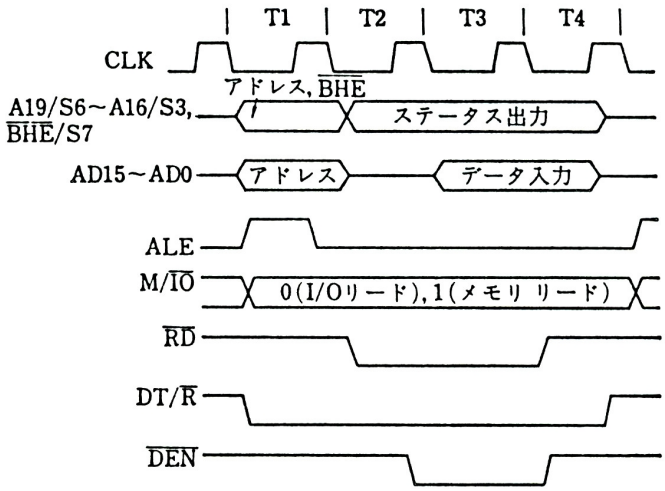



図 3.17　8086におけるデータリード

ク，ストア３クロックと違いがあるところがメモリ・インタフェースにおけるＺ80の弱点である。このように，外部メモリに対する読み出し／書き込みのサイクルが異なるため，Ｚ80に供給するクロック周波数が高くなると，一般に使用されるダイナミックRAMのサイクルタイムを満足しえなくなり，ウエイト・サイクルを挿入してメモリサイクルを引き延ばしてやる必要がでてくる。

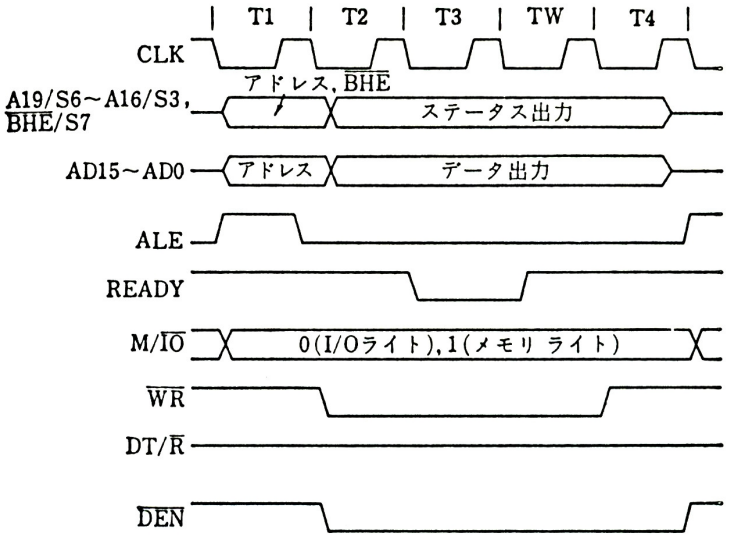



図 3.18　8086におけるライト

マイコンは一般にウエイト動作機能を持っている。メモリ・アクセス・サイクルの途中（正確には，最終ステートの開始時）で，ウエイト要求信号（WAITとかREADY）を受け取り，マイコンはそのときのアドレス／データ／制御バスの状態を保持する。マイコンが独自に規定しているメモリ・サイクル内でデータの転送が実行できないような場合に，ウエイト要求信号が発生するように装置設計をする。Ｚ80では，上記したようなウエイト・サイクルの挿入を容易とするために，特に，命令フェッチ時のタイミングであることを示す信号Ｍ１を出力している。一般に，リードやライトの信号が出力されている期間でウエイト状態に入り，ウエイトが解除されるまでリードやライト信号は能動状態（通常，低レベル）を保っている。リード時には，リード信号の低レベルの最終データをマイコンが読み込むことになるので，常に，この種のウエイトは有効であるが，ライト時のウエイト動作に多少，問題を残している。スタティックRAMの場合にはライト信号の立ち上がりエッジで書き込みを実行するので問題はないが，ダイナミックRAMの場合には，立ち下がりエッジでデータを書き込むスペックになっている。このため，いくらウエイトを挿入しても，立ち下がりエッジを遅らせることはできないので，ライト信号を外部回路によって遅延させるなどのなんらかの処置が必要となる。

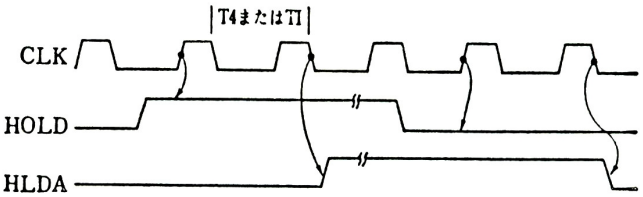



図 3.19　8086におけるホールド

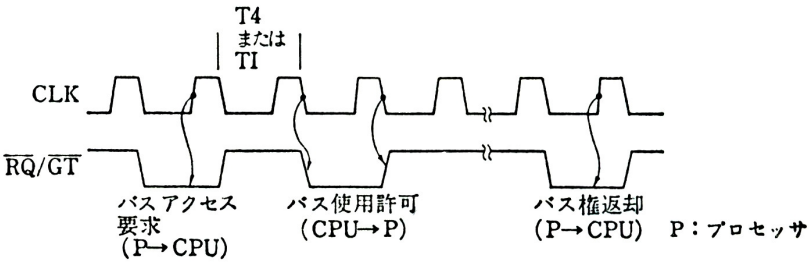
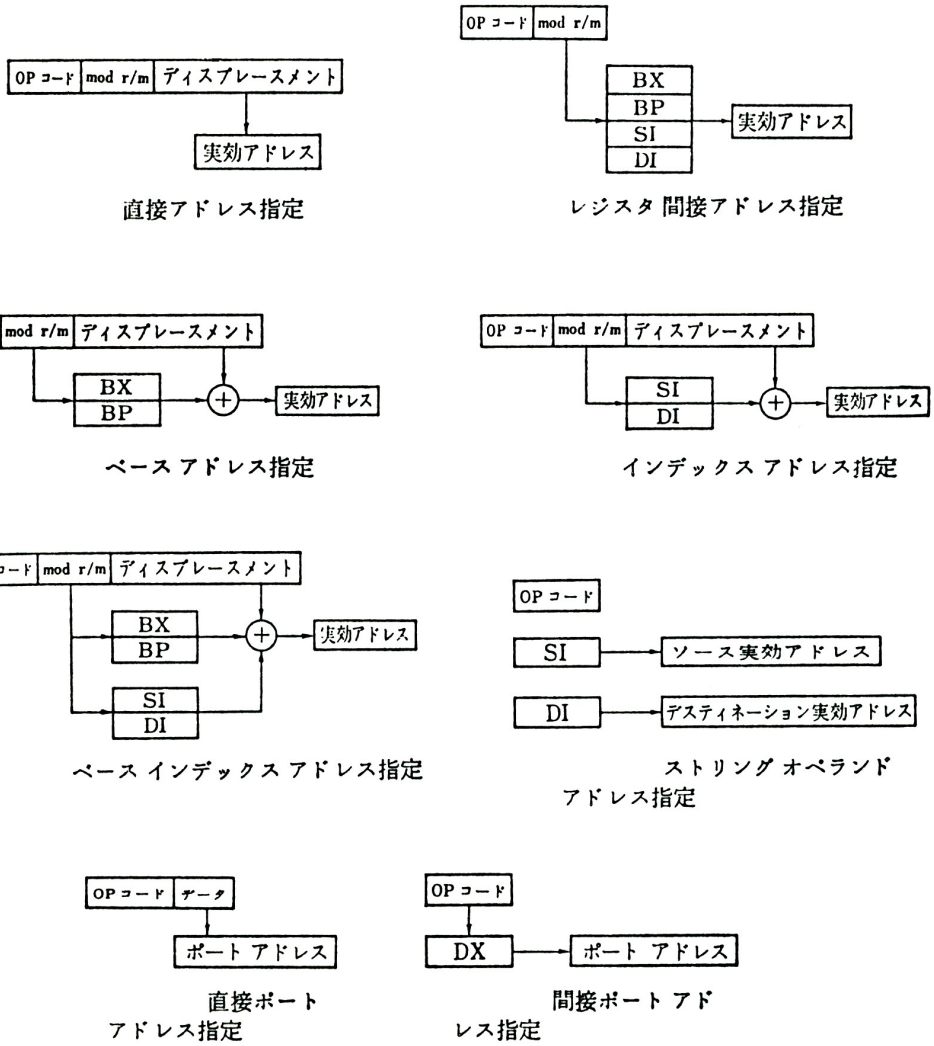



図 3.20　8086のリクエスト／グラント

さらに，ダイレクト・メモリ・アクセス（DMA）を行なったり，他のマイコンに制御を渡すなどの目的で，アドレス／データ／制御のバスをフローティング状態に移行する動作を行なうことができる。命令実行サイクルの最終マシン・サイクルの最終ステート開始時に，ホールド要求信号（HOLDとかBUS REQUEST）を受け取ると，ホールド承認信号（HOLD ACKNOWL-




| R/M \ MOD | 00 | 01 | 10 |
|---|---|---|---|
| 000 | (BX)+(SI) | (BX)+(SI)+disp 8 | (BX)+(SI)+disp 16 |
| 001 | (BX)+(DI) | (BX)+(DI)+disp 8 | (BX)+(DI)+disp 16 |
| 010 | (BP)+(SI) | (BP)+(SI)+disp 8 | (BP)+(SI)+disp 16 |
| 011 | (BP)+(DI) | (BP)+(DI)+disp 8 | (BP)+(DI)+disp 16 |
| 100 | (SI) | (SI)+disp 8 | (SI)+disp 16 |
| 101 | (DI) | (DI)+disp 8 | (DI)+disp 16 |
| 110 | ダイレクト アドレス | (BP)+disp 8 | (BP)+disp 16 |
| 111 | (BX) | (BX)+disp 8 | (BX)+disp 16 |

図3.21　8086のアドレッシング

EDGE）を出力するとともに，自分自身のバス出力信号をフローティング状態とし，他の装置から出力されるバス信号との衝突（BUS CONTENSION）を避けるようにした後，処理などを他の装置に引き渡す。通常，後述するDMAコントローラを使用して周辺制御回路を削減している。8086のマクシマム・モードでは，I／Oプロセッサ8089の浮動小数点演算プロセッサ8087などのコ・プロセッサ間とのインタフェースをより効率的なものとするため，REQUEST／GRANT端子を相互に結線することにより，処理の受渡しを実行している。

また，割込要求信号を受けつけることもできる。割込要求信号には，プログラムによってマス

| ニモニック | 機　　能 |
|---|---|
| AAA | ASCII Adjust for Addition |
| AAD | ASCII Adjust for Division |
| AAM | ASCII Adjust for Multiplication |
| AAS | ASCII Adjust for Subtraction |
| ADC | Add with Carry |
| ADD | Add |
| AND | And |
| CALL | Call |
| CBW | Convert Byte to Word |
| CLC | Clear Carry |
| CLD | Clear Direction |
| CLI | Clear Interrupt |
| CMC | Complement Carry |
| CMP | Compare |
| CMPB | Compare Byte (of string) |
| CMPW | Compare Word (of string) |
| CWD | Convert Word to Double Word |
| DAA | Decimal Adjust for Addition |
| DAS | Decimal Adjust for Subtraction |
| DEC | Decrement |
| DIV | Divide |
| ESC | Escape |
| HLT | Halt |
| IDIV | Integer Divide |
| IMUL | Integer Multiply |
| IN | Input |
| INC | Increment |
| INT | Interrupt |
| INTO | Interrupt on Overflow |
| IRET | Interrupt Return |
| JA | Jump on Above |
| JAE | Jump on Above or Equal |
| JB | Jump on Below |
| JBE | Jump on Below or Equal |
| JCXZ | Jump on CX Zero |
| JE | Jump on Equal |
| JG | Jump on Greater |
| JGE | Jump on Greater or Equal |
| JL | Jump on Less |
| JLE | Jump on Less or Equal |
| JMP | Jump |
| JNA | Jump on Not Above |
| JNAE | Jump on Not Above or Equal |
| JNB | Jump on Not Below |
| JNBE | Jump on Not Below or Equal |
| JNE | Jump on Not Equal |
| JNG | Jump on Not Greater |
| JNGE | Jump on Not Greater or Equal |
| JNL | Jump on Not Less |
| JNLE | Jump on Not Less or Equal |
| JNO | Jump on Not Overflow |
| JNP | Jump on Not Parity |
| JNS | Jump on Not Sign |
| JNZ | Jump on Not Zero |
| JO | Jump on Overflow |
| JP | Jump on Parity |
| JPE | Jump on Parity Even |
| JPO | Jump on Parity Odd |
| JS | Jump on Sign |
| JZ | Jump on Zero |
| LAHF | Load AH with Flags |
| LDS | Load Pointer into DS |
| LEA | Load Effective Address |
| LES | Load Pointer into ES |
| LOCK | Lock Bus |
| LODS | Load String |
| LOOP | Loop |
| LOOPE | Loop While Equal |
| LOOPNE | Loop While Not Equal |
| LOOPNZ | Loop While Not Zero |
| LOOPZ | Loop While Zero |
| MOV | Move |
| MOVS | Move string |
| MUL | Multiply |
| NEG | Negate |
| NOP | No operation |
| NOT | Not |
| OR | Or |
| OUT | Output |
| POP | Pop |
| POPF | Pop Flags |
| PUSH | Push |
| PUSHF | Push Flags |
| RCL | Rotate through Carry Left |
| RCR | Rotate through Carry Right |
| REP | Repeat |
| RET | Return |
| ROL | Rotate Left |
| ROR | Rotate Right |
| SAHF | Store AH into Flags |
| SAL | Shift Arithmetic Left |
| SAR | Shift Arithmetic Right |
| SBB | Subtract with Borrow |
| SCAS | Scan String |
| SHL | Shift Left |
| SHR | Shift Right |
| STC | Set Carry |
| STD | Set Direction |
| STI | Set Interrupt |
| STOS | Store String |
| SUB | Subtract |
| TEST | Test |
| WAIT | Wait |
| XCHG | Exchange |
| XLAT | Translate |
| XOR | Exclusive Or |

図3.22　8086の命令一覧表

クできるものとできないもの（ノンマスカブル）の２種ある。要求信号は，上述したホールド要求と同様に，命令実行サイクルの最終ステートの開始時に，その発生の有無を検出される。ノンマスカブル割込要求の場合には，次に，マイコンは割込ベクタなどを取り込む命令サイクル（INTERRUPT ACKNOWLEDGE）に入る。この割込制御についても，後述するような割込コントローラが一般に使用される。このような外部機器からの割込の他に，ソフトウエア自身によって発生できる割込（Ｚ80におけるRST；リスタート命令，8086におけるINT命令）や内部要因に起因（例えば，8086におけるDIVISION BY ZERO）する割込もある。これらの割込はマスクの状態に依存しない。ソフトウエア割込は，命令バイト長が少なくて済み，プログラム・カウンタがどのメモリ領域にあっても使用可能な簡易的サブルーチン・コール命令として使用される場合が多い。

一般に，種々のアドレッシング方法を持っているが，次の３種に要約できる。

(1) 直接アドレッシング

(2) 間接アドレッシング

(3) 相対アドレッシング

アドレスの対象となるものは，命令実行番地を格納するプログラム・カウンタの内容，即ち，命令実行番地の選択に関するものと，処理すべきデータが格納されている番地の選択に関するものの２通りがある。直接アドレッシングでは，命令コードのなかに，ジャンプ先の絶対アドレスや即値データ（IMMEDIATE DATA）が付随している。間接や相対アドレッシングには，色々なバリエーションがある。即値データが指す番地，レジスタが指す番地，さらに，ディスプレーメント演算が含まれる場合もある。即値データの場合のみを取り上げると，間接アドレッシングでは，即値データが示す番地や指定されたレジスタに格納されているデータが実行データとして採用される。相対アドレッシングでは，即値データやレジスタ内容はアドレス変位を示すデータとして使用され，現在アドレスとの間で演算され，結果が採用される。相対や間接アドレッシングのみによってプログラムが書ければ，そのプログラムをどのアドレスに再配置しても実行できる（リロケータブル）ことになる。

実行できる命令は次のように分類できる。

(1) 算術演算命令

(2) 論理演算命令

(3) 転送／シフト／回転命令

(4) 分岐／コール／ジャンプ命令

(5) その他のプロセッサ制御命令

１個の演算器は２種のデータを受けつけることができる。即ち，Ｓ１＋Ｓ２⟶Ｄ（ここで，ＳはSOURCE，ＤはDESTINATION)の如くな演算が可能である。ところが通常のマイコンの内部バスは次のような理由から，１バス構成を採用している。データの並列読み出し／書き込みができるように内蔵レジスタを高速アクセス可能なレジスタ・ファイル構成とし，２バス構成や３バス構成とすれば，データ処理時間を減少することができるが，配線領域が増大しチップ面積が大きくなったり，消費電力が増大する。さらに，並列アクセス可能なレジスタの種類が限定さ

れるなどの命令アーキテクチュア上に問題がでてくる。このため，通常のマイコンは１バス構成とし，内蔵レジスタはRAM構造としている。その結果，一般に，上記Ｓ１とＤとは同一のレジスタなどに格納されているデータが参照されることとなる。データとしてはマイコンが内蔵しているレジスタやメモリ，Ｉ／Ｏ（INPUT／OUTPUT DEVICE）の内容が参照される。ＤとＳ２とは次のような組み合わせで使用できる。ブロック転送命令やストリング命令などの特殊命令では，メモリ－メモリ間の転送も選択できる。

(1) レジスタ（ Ｄ）
　　レジスタ（Ｓ２）
(2) レジスタ（ Ｄ）
　　メモリ　（Ｓ２）
(3) メモリ　（ Ｄ）
　　レジスタ（Ｓ２）

マイコンの機能を補助する種々なコントローラなどをメモリとしてではなく，Ｉ／Ｏとしてアドレスをマッピングすることもできるのが一般的である。Ｉ／Ｏ

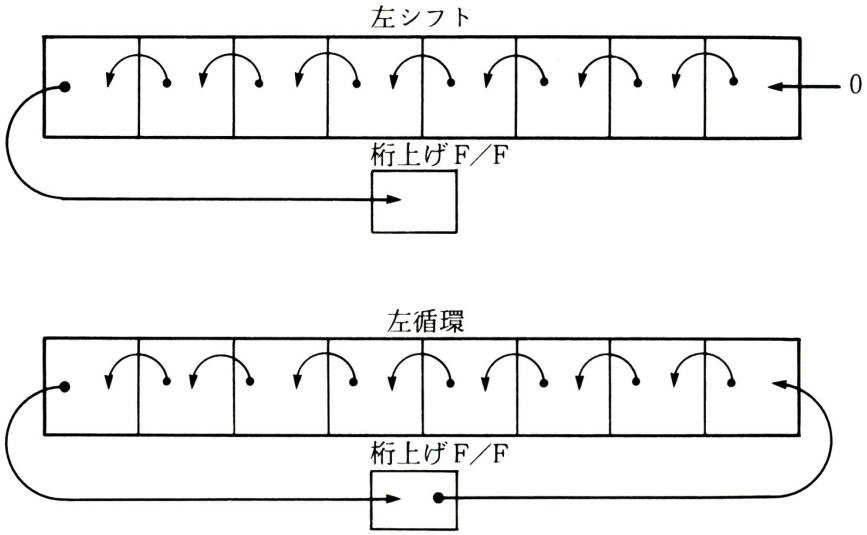
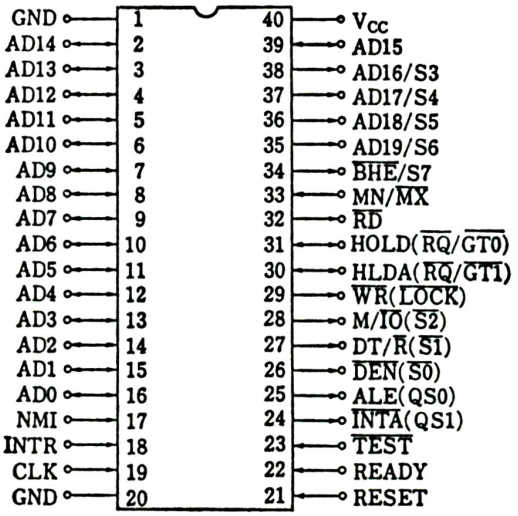



図 3.23　シフトと回転



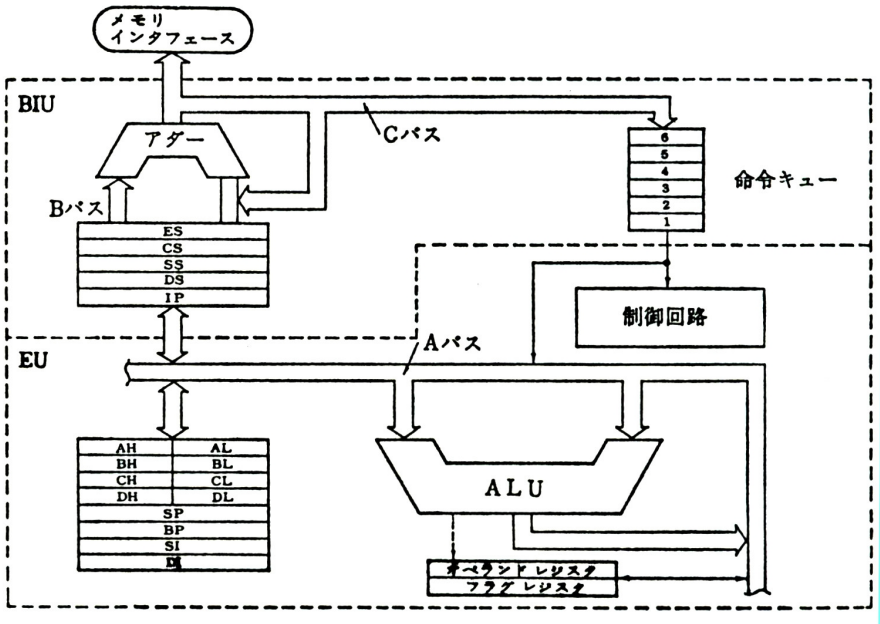




図 3.24　8086端子接続図およびブロック図

としてマップしたとき，データのアクセスは入出力命令（INとかOUT）により行ない，最大256種程度のポート・アドレスによって弁別する。

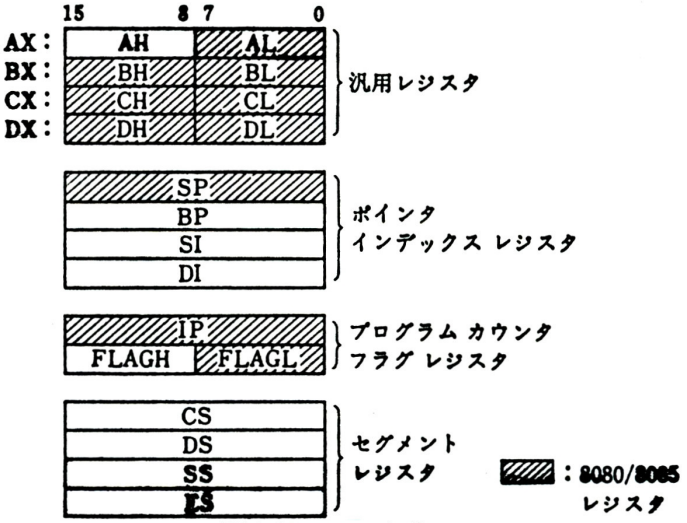



図3.25　8086内蔵レジスタ

サブルーチンをコールする際には，サブルーチンでの処理終了後，もとの処理の流れに復帰できるように，あらかじめ，戻り番地を記憶しておく必要がある。マルチチップ・マイコンでは，ある程度大きなメモリ実装領域を通常持っているので，戻り番地格納領域（スタック領域）をメモリ内に確保しておき，コール命令時には，そのときのスタック・ポインタが指しているアドレスに戻り番地を格納し，サブルーチンの最後に位置させるリターン命令によって，その戻り番地をプログラム・カウンタに引き戻す。データを一時的に退避（セーブ）するために，この領域を使用することもできる。PUSH／POPなどの命令により実行する。このスタック領域のデータの出し入れは，最

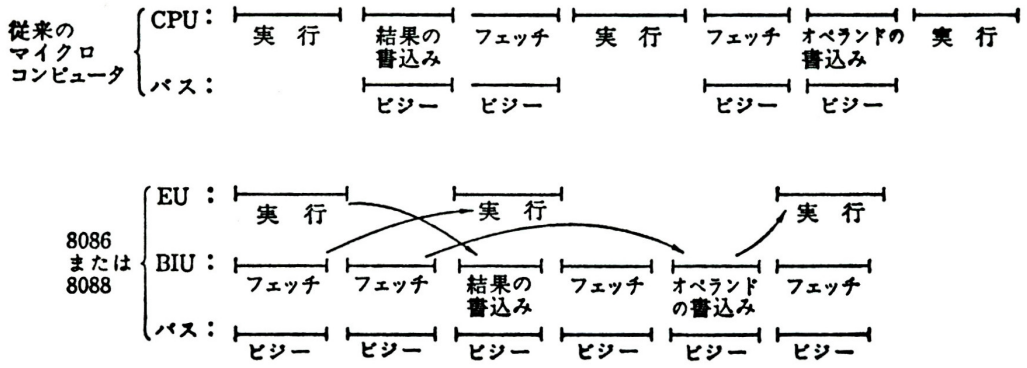



図3.26　8086パイプライン処理

| マシンサイクル | $M_1$ | | | | | | $M_2$ | | | $M_3$ | | | $M_4$ | | | $M_5$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| クロックサイクル | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ | $T_1$ | $T_2$ | $T_3$ | $T_1$ | $T_2$ | $T_3$ | $T_1$ | $T_2$ | $T_3$ | $T_1$ | $T_2$ | $T_3$ |
| マシンサイクルのタイプ | 命令コードフェッチ | | | | | | メモリリード | | | メモリリード | | | メモリライト | | | メモリライト | | |
| アドレスバス | PCの内容で命令の第1バイト目を指す | | | | | | PC＋1で命令の第2バイト目を指す | | | PC＋2で命令の第3バイト目を指す | | | SP－1のアドレスでスタックメモリを指す | | | SP－2のアドレスでスタックメモリを指す | | |
| データバス | CALL命令のコード | | | | | | ダイレクトアドレスの下位バイト | | | ダイレクトアドレスの上位バイト | | | 退避するPC＋3の上位バイト | | | 退避するPC＋3の下位バイト | | |

PC：プログラムカウンタ　　SP：スタックポインタ

図3.27　8085マシンサイクル例

後に入力したデータが最初に出力されてくるLIFO（LAST-IN FIRST-OUT）形式となっている。最初に入力したデータを最初に出力する形式を持つスタックをFIFO（FIRST-IN FIRST-OUT）と呼んでいる。後述する8086に搭載されている命令キュー・バッファはこのタイプのスタック・レジスタである。

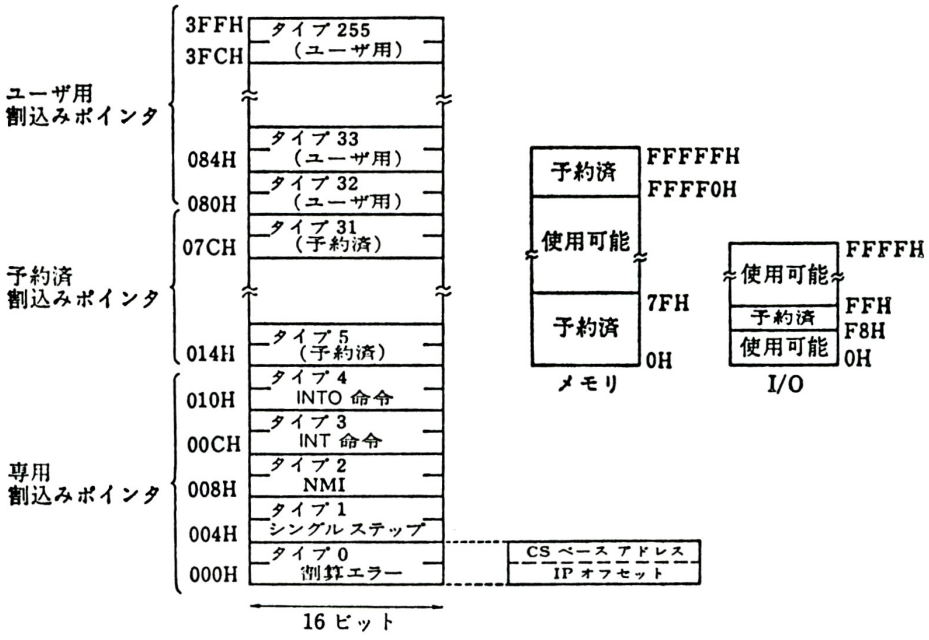



図 3.28　8086メモリ予約

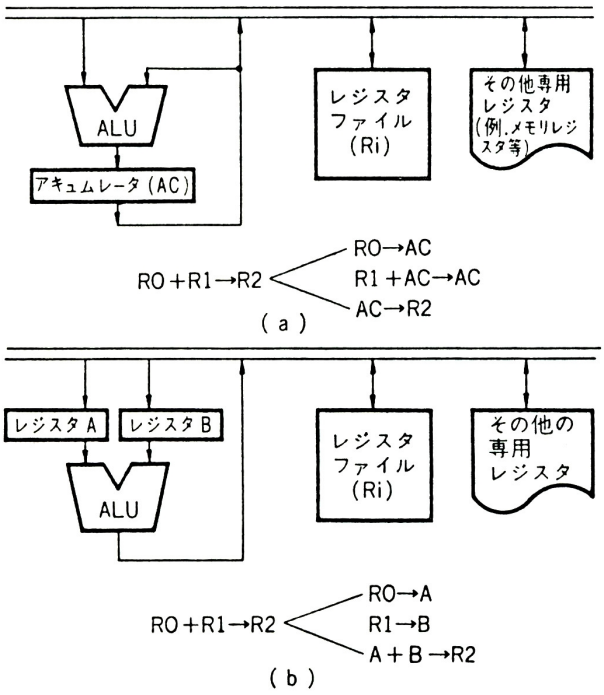



図 3.29　1バス構成例

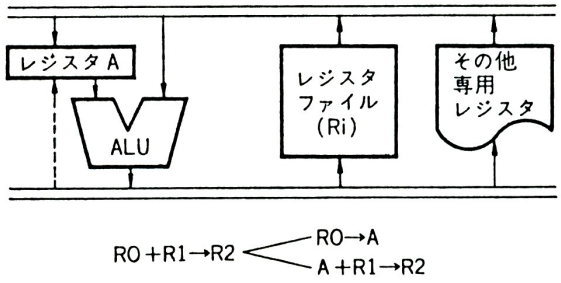



図 3.30　2バス構成例

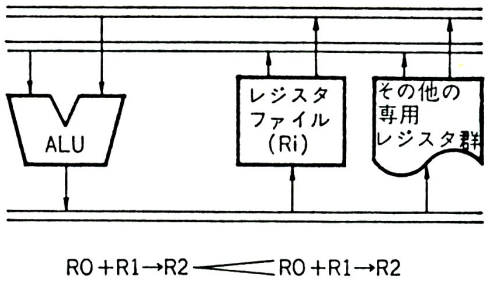



図 3.31　3バス構成例

演算命令などを実行した結果を基にして条件判定が容易に行なえるように，アキュムレータの状態などを記憶するフラグ・レジスタを持っている。頻繁に条件

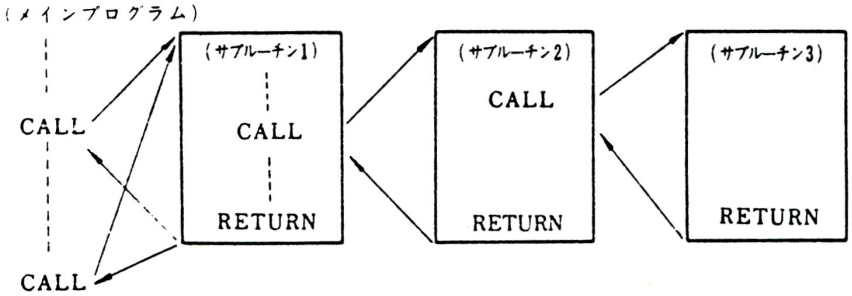



図 3.32　サブルーチン・コール

判断に使用されるフラグは演算結果がゼロであったことを示すゼロフラグとキャリーやボローが発生したことを示すキャリーフラグである。実際に制御プログラムを書く場合には，どのような命令を実行させた場合には，どのフラグの内容が変化するのかを，理解しておくとジャンプ命令を極力減らした効果の良いプログラムを設計できるようになる。

プログラムの流れの状態をレジスタに記憶させておいて，プログラムの分岐を制御させるようなとき，次のような2種の方法が使用される。

(1) コード化したフラグ

(2) ビット毎に独立なフラグ

(1)は1個の判断を行なうのに何種類もの状態が存在する場合に有効であり，(2)は1か0かの状態しかない単純な場合に使用される。ビットフラグの設定や判定を容易に行なえるように，ビット・マニピュレーションといえるほどのものではないが，ビット制御命令を持っている。

プログラムの分岐は，これらのフラグの状態を判断して分岐先アドレスへジャンプする場合と，次の命令をスキップする形を取る場合との2種ある。後者は後述するシングルチップ・マイコンで使用されることがある。

以下，インテル社の16ビット・マイコン8086を例にとって説明する。

チップ内をBIU（BUS INTERFACE UNIT）とEU（EXECUTION UNIT）とに大きく分けて設計を別々に行なったとされている。6バイトの命令キューを持っており，メモリのアクセスに空き時間が生じたことを判断して，命令キューの中に命令を順次スタックしていくことにより，パイプライン処理を可能にしている。このため，バスの使用効率を非常に高くすることができる。しかしながら，1系統の命令キューしか持たぬため，命令の分岐が生じた場合にはこの命令キューは何の役にも立たなくなる。

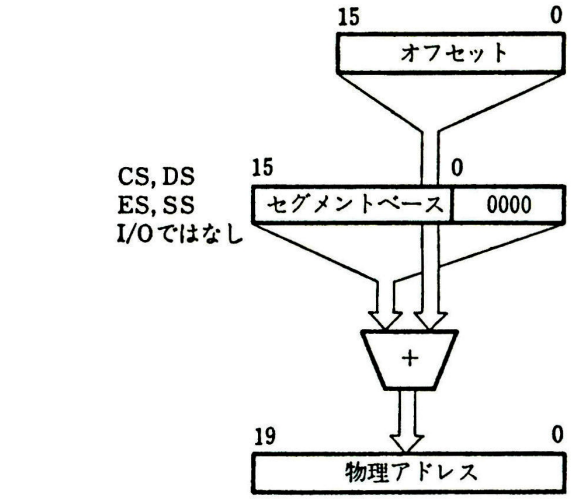
8ビット・マイクロプロセッサとのソース・レベルでの命令互換性を保つため，レジスタ構成は類似なものとしている。アドレスを拡張するため，4種のセグメント・レジスタが用意されている。実際にメモリに供給される物理アドレスは16ビットのオフセット・アドレスとセグメント・レジスタとの演算結果として与えられる。この結果，64Kバイトのアドレス空間の中だけであれば，1Mバイト中の任意のアドレスを指すことができるが，64Kバイトを超える場合には，セグメント・レジスタの内容を書き換えねばならない。

64Kバイトを超えるような大きなプログラムやデータを取り扱う場合に，この64Kバイトの境界制約はプログラマーにとって大きな負担となる。また，使用する命令によって，参照されるセグメント・レジスタの種類が暗黙的に変更されることもプログラマーを悩ます。任意にその種類を決定したいのであれば，セグメント・オーバーライド・プリフィックスを使用する逃げ道は作ってあるが，そのたびに設定しなおすのでは，プログラムや実行時間が長くなってしまう。

8086は種々なアドレッシングが可能であるが，その実効アドレス（EA）の形成にかなりな時間が費やされることに注意しなければならない。この時間を削減することができれば，処理能力が向上することは間違いない。

バイトのアドレスをバスに出力する。メモリ・アドレス信号A0および制御信号BHE／（BUS HIGH ENABLE）をメモリ・バンク選択に使用することにより，ローバイト，ハイバイト，同

一ワードアドレス内のワード，異なったワードアドレスにまたがったワードのすべてのデータ形式のアクセスが可能である。但し，ワード間にまたがったデータの場合には，当然のことながら，メモリ・アクセスを2回必要とする。ワードのデータを大量に扱うような場合は，特に，データの格納開始アドレスを偶数番地に意識的に設定しておかないと処理時間が遅くなる。モトローラの68000の場合には，このようなデータはアセンブル・エラーとして，はじきだすようにしている。



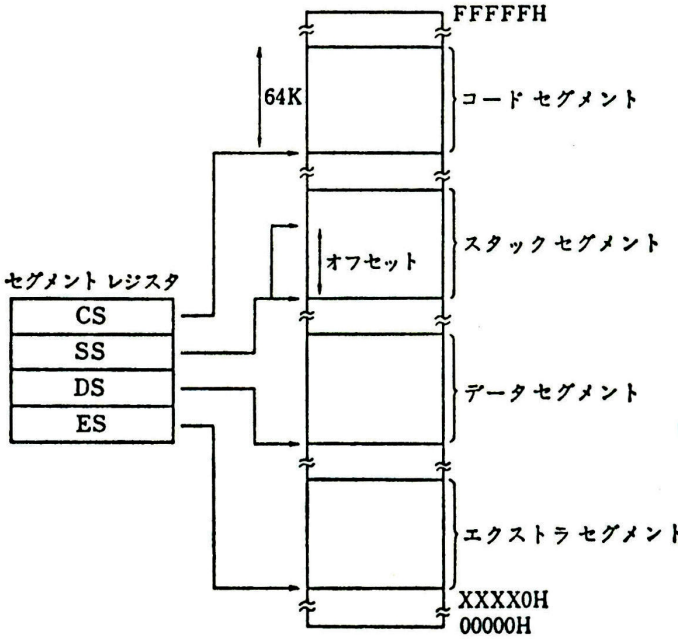



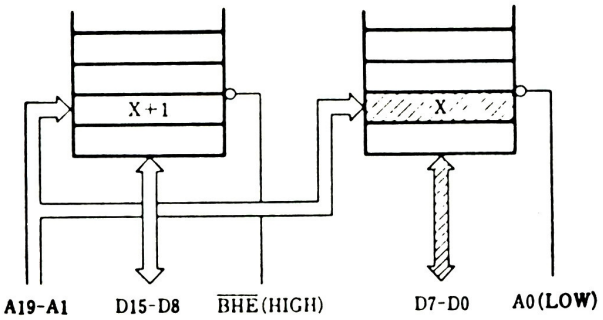

図 3.33　8086のセグメント

8086は，DMA転送などを司どるI／Oプロセッサ8089や浮動小数点演算などを高速に実行するプロセッサ8087との組み合わせでシステムを構成することができる。8086のミニマム・モードにおいて，ホールド要求／承認信号用として使用されていた端子をマクシマム・モードでは，8086とコ・プロセッサ間での処理の受渡し制御用として使用する。コ・プロセッサは8086と同じタイミングで命令を解読しており，自身が実行せねばならない命令に出会うと8086に対してバス要求を出す。次に，8086がグラント信号をコ・プロセッサに返した後は，コ・プロセッサがバスの制御権を握り，処理が終了するとバス・リリース信号を8086に送る。1個のチップ内にコ・プロセッサとしての機能を総て集積してしまうことができるならば，このようなチップ構成とする必要性はない。ワン・チップ化ができないときのチップ分割の1つの手法であると言ったほうがよい。

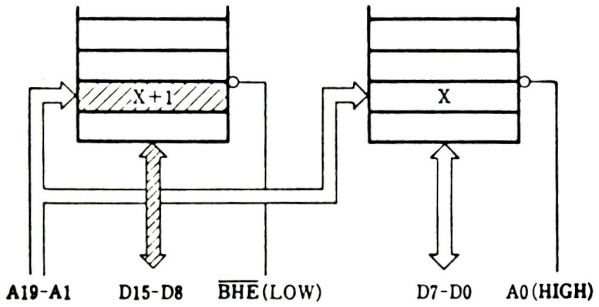
| メモリアドレス指定 | EA (クロック数) |
|---|---|
| 直接16ビットオフセットアドレス | 6 |
| ベースまたはインデックスレジスタによる間接 | 5 |
| ディスプレースメントを伴ったベースまたはインデックスレジスタによる間接 | 9 |
| インデックスレジスタとベースレジスタとの和による間接 | 7または8 |
| ディスプレースメントを伴ったベースレジスタとインデックスレジスタとの和による間接 | 11または12 |

図3.34　8086のアドレス演算時間

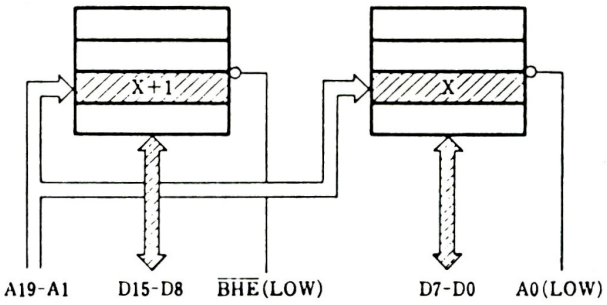
| BHE | A0 | 転 送 バ イ ト |
|---|---|---|
| 0 | 0 | 両バンクともに |
| 0 | 1 | 奇数バンクと D15～D8 との間 |
| 1 | 0 | 偶数バンクと D7～D0 との間 |
| 1 | 1 | 転送なし |



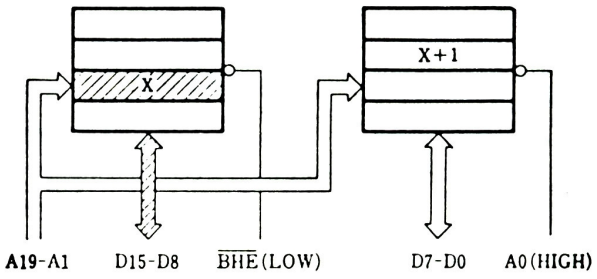

偶数アドレスバイトの転送



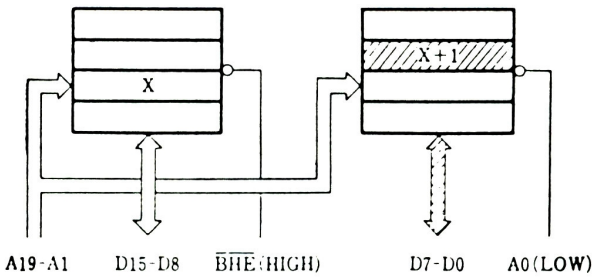

奇数アドレスバイトの転送




偶数アドレス語の転送



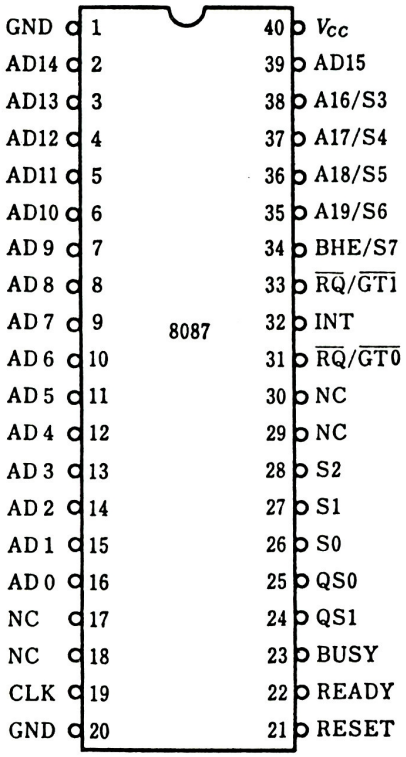

(a) 下位バイトの転送




(b) 上位バイトの転送

奇数アドレス語の転送

図3.35　8086のメモリ・アクセス



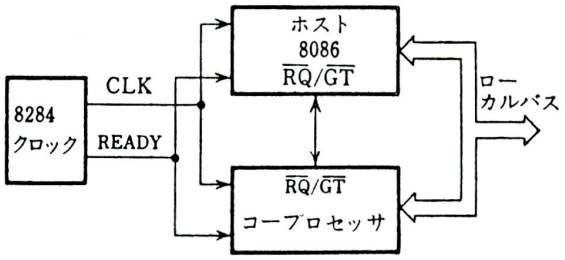




図3.36　8087端子接続図および接続法

8087はエスケープ命令（ESC）を常時，監視している。種々のデータ型による種々な演算を高速に実行するハードウエアを内蔵している。

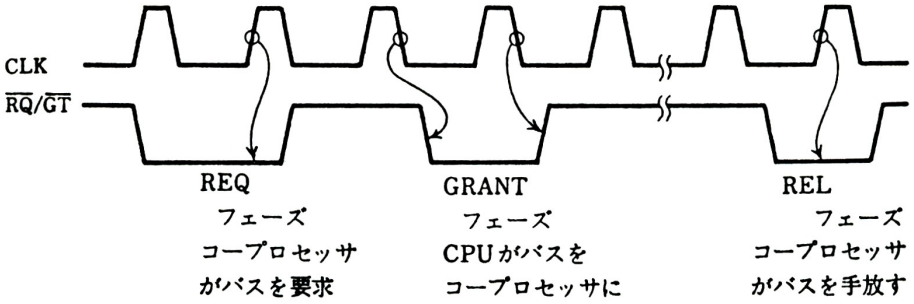



図3.37　リクエスト／グラント

データ形

| データ形 | ビット数 | 有効桁数（10 進） | 表現可能範囲（10 進） |
|---|---|---|---|
| 語整数 | 16 | 4 | $-32\,768 \leq X \leq +32\,767$ |
| 短整数 | 32 | 9 | $-2\times10^{9} \leq X \leq +2\times10^{9}$ |
| 長整数 | 64 | 18 | $-9\times10^{18} \leq X \leq +9\times10^{18}$ |
| 10 進数 | 80 | 18 | $-99\cdots99 \leq X \leq +99\cdots99$（18 桁） |
| 単精度実数 | 32 | 6-7 | $8.43\times10^{-37} \leq \lvert X\rvert \leq 3.37\times10^{38}$ |
| 倍精度実数 | 64 | 15-16 | $4.19\times10^{-307} \leq \lvert X\rvert \leq 1.67\times10^{308}$ |
| 内部実数 | 80 | 19 | $3.4\times10^{-4932} \leq \lvert X\rvert \leq 1.2\times10^{4932}$ |

| 1101 1××× | mod ××× r/m | disp-low | disp-high |
|---|---|---|---|

ESC 命令（×はドントケアを表す）

| 1101 1000 | mod 001 r/m | disp-low | disp-high |
|---|---|---|---|

ESC命令を使った FMUL 命令

| 種類 | 命令 |
|---|---|
| データ転送 | 置数，格納，交換 |
| 算術演算 | 加算，減算，乗算，除算，逆減算，逆除算，平方根，スケール，剰余，整数化，サイン変更，絶対値，実数分解 |
| 比較演算 | 比較，エグザミン，テスト |
| 関数演算 | 正接，逆正接，$2^{X}-1$，$Y\cdot\log_2(X+1)$，$Y\cdot\log_2 X$ |
| 定数 | 0，1，$\pi$，$\log_{10}2$，$\log_e2$，$\log_2 10$，$\log_2 e$ |
| プロセッサ制御 | コントロール語転送，ステイタス語転送，割込み禁止/解除，初期化，例外フラグクリヤ |

図3.38　8087データ型／命令

## 7. マイクロコンピュータ用LSI

8ビット・シングルチップ・マイコンのアーキテクチュアの原型は8080にあるが，4ビットの場合は電卓にある。いずれも，集積されているROMやRAMの容量に限度があることから，できる限り少ないプログラム・ステップで多くの処理ができるようにし，また，周辺機器の制御機能をできる限りオンチップ化する目的を持つため，マルチチップ・マイコンとは異なった，次のような特徴を持っている。

(1) 命令語長や命令実行時間の短縮

(2) クロック発振器，タイマー，イベント・カウンタ，シリアルI／O，アナログ／ディジタル（A／D）変換器，入出力パラレル・ポート，蛍光表示管（VFD；VACUUM FLUORESCENT DISPLAY／発光ダイオード（LED；LIGHT EMTTING DIODE）／液晶（LCD；LIQUID CRYSTAL DISPLAY）などの表示コントローラ／ドライバなどのオンチップ化

(3) 外部メモリ拡張機能

内蔵しているROMやRAMの容量が小さいことから直接アドレッシングを用いても，8048の場合，最大2バイトで命令を表現できる。ジャンプ命令や即値データを必要とする命令を除くと，他の命令は1バイトで構成している。発振周波数をチップ内で3分周した2相クロックによって内部論理回路を動作させている。内部クロック5クロックで1命令サイクルを形成しており，最大2サイクルで1個の命令の実行を終了する。12MHzのクロックを供給していても内部の論理回路は，実際には，その1／3の周波数で動作して

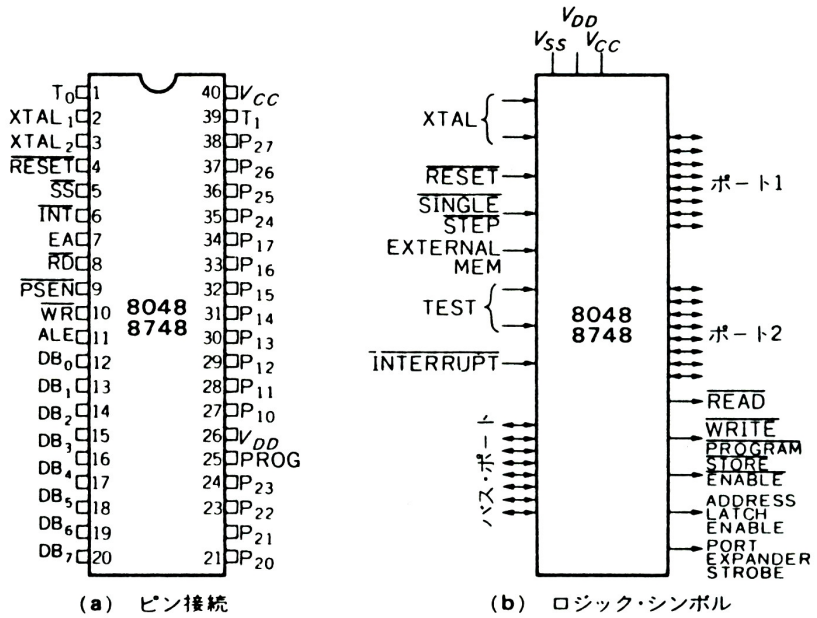



図 3.39 8048端子接続図

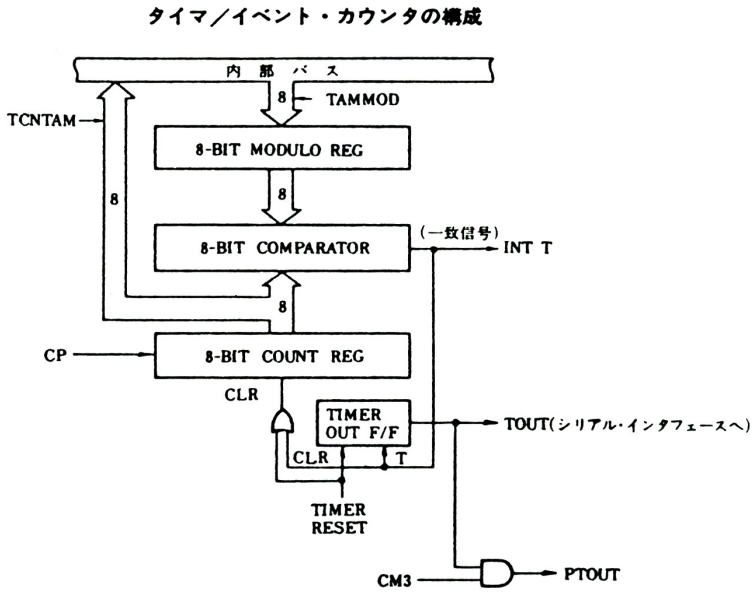



図 3.40 タイマー／イベント・カウンタ構成図

いる。8086の場合には，供給したクロックと同じ周波数の2相クロックを内部で発生させ論理回路を動作させているので，8 MHz規格品の内部論理回路は8 MHzで動作している。

フラグ・レジスタの内容の状態を基にジャンプするかどうかを決めるコンディショナル・ジャンプは，命令語のなかに，もともとジャンプ先アドレスを記述する部分を含んでいるため，命令を構成するバイト数が多い。また条件判断の種類が多くなってくると，他の命令のバリエーションを増加しにくくなってくる。ジャンプ命令やサブルーチン・コール命令は，通常のプログラムでの使用頻度が非常に高いことは重要なことである。このため2バイト命令ではなく1バイト命令でジャンプやコールが実行できれば，効率の良いプログラムが書けることになる。このため，種々な試みがなされている。

ジャンプ命令は無条件ジャンプのみにして

| | Mnemonic | Description | Bytes | Cycle |
|---|---|---|---|---|
| Accumulator | ADD A,R | Add register to A | 1 | 1 |
| | ADD A,@R | Add data memory to A | 1 | 1 |
| | ADD A,#data | Add immediate to A | 2 | 2 |
| | ADDC A,R | Add register with carry | 1 | 1 |
| | ADDC A,@R | Add data memory with carry | 1 | 1 |
| | ADDC A,#data | Add immediate with carry | 2 | 2 |
| | ANL A,R | And register to A | 1 | 1 |
| | ANL A,@R | And data memory to A | 1 | 1 |
| | ANL A,#data | And immediate to A | 2 | 2 |
| | ORL A,R | Or register to A | 1 | 1 |
| | ORL A,@R | Or data memory to A | 1 | 1 |
| | ORL A,#data | Or immediate to A | 2 | 2 |
| | XRL A,R | Exclusive Or register to A | 1 | 1 |
| | XRL A,@R | Exclusive or data memory to A | 1 | 1 |
| | XRL A,#data | Exclusive or immediate to A | 2 | 2 |
| | INC A | Increment A | 1 | 1 |
| | DEC A | Decrement A | 1 | 1 |
| | CLR A | Clear A | 1 | 1 |
| | CPL A | Complement A | 1 | 1 |
| | DA A | Decimal Adjust A | 1 | 1 |
| | SWAP A | Swap nibbles of A | 1 | 1 |
| | RL A | Rotate A left | 1 | 1 |
| | RLC A | Rotate A left through carry | 1 | 1 |
| | RR A | Rotate A right | 1 | 1 |
| | RRC A | Rotate A right through carry | 1 | 1 |
| Input／Output | IN A,P | Input port to A | 1 | 2 |
| | OUTL P,A | Output A to port | 1 | 2 |
| | ANL P,#data | And immediate to port | 2 | 2 |
| | ORL P,#data | Or immediate to BUS | 2 | 2 |
| | INS A,BUS | Input BUS to A | 1 | 2 |
| | OUTL BUS,A | Output A to BUS | 1 | 2 |
| | ANL BUS,#data | And immediate to BUS | 2 | 2 |
| | ORL BUS,#data | Or immediate to BUS | 2 | 2 |
| | MOVD A,P | Input Expander port to A | 1 | 2 |
| | MOVD P,A | Output A to Expander port | 1 | 2 |
| | ANLD P,A | And A to Expander port | 1 | 2 |
| | ORLD P,A | Or A to Expander port | 1 | 2 |
| Registers | INC R | Increment register | 1 | 1 |
| | INC @R | Increment data memory | 1 | 1 |
| | DEC R | Decrement register | 1 | 1 |
| Branch | JMP addr | Jump unconditional | 2 | 2 |
| | JMPP @A | Jump indirect | 1 | 2 |
| | DJNZ R,addr | Decrement register and skip | 2 | 2 |
| | JC addr | Jump on Carry=1 | 2 | 2 |
| | JNC addr | Jump on Carry=0 | 2 | 2 |
| | J Z addr | Jump on A Zero | 2 | 2 |
| | JNZ addr | Jump on A not Zero | 2 | 2 |
| | JT0 addr | Jump on T0=1 | 2 | 2 |
| | JNT0 addr | Jump on T0=0 | 2 | 2 |
| | JT1 addr | Jump on T1=1 | 2 | 2 |
| | JNT1 addr | Jump on T1=0 | 2 | 2 |
| | JF0 addr | Jump on F0=1 | 2 | 2 |
| | JF1 addr | Jump on F1=1 | 2 | 2 |
| | JTF addr | Jump on timer flag | 2 | 2 |
| | JNI addr | Jump on INT=0 | 2 | 2 |
| | JBb addr | Jump on Accumulator Bit | 2 | 2 |

**図 3.41　8048の命令の一部**

しまい，命令語の組み合わせ数を節約し，条件判断は命令実行時毎に行なって，条件がとれた場合には次の命令をノー・オペレーションとして取り扱い，スキップする方法を採るマイコンもある。さらに，この方法を発展させたものとして縦積み命令と称する命令を持つものもある。アキュムレータへの即値データのロード命令など特殊な命令に限定し，その命令が連続してプログラムされいる場合には，一番初めの命令だけを正常に実行した後は,連続する同種命令はスキップし実行しない。異なった値を同一のレジスタにロードしたい場合に，不要なジャンプ命令を取り除くことができ有効である。さらに，1バイトでプログラムできるコール命令が実行できるようになっている品種もある。前もって予約されているレジスタ領域にコール先のアドレスを格納しておき，コール命令実行時には，そのアドレス・テーブルを参照するようにしている。命令実行時間は多少増加するが，プログラム語数はかなり削減できる。

```
スタート→ MVI   A，0      ；  A←0
          MVI   A，1      ；  NOP(7 CLOCKS), L1＝1
          MVI   A，2      ；  NOP(7 CLOCKS), L1＝1
          MVI   L，0AH    ；  L←0AH
          MVI   L，0BH    ；  NOP(7 CLOCKS), L0＝1
割込み  → MVI   L，0CH    ；  NOP(7 CLOCKS), L0＝1
          LXI   H，00H    ；  NOP(10CLOCKS), L0＝1
```

**図 3.42　縦積命令**

命令実行時間を速めるために，サブルーチン・コール時のアドレスやデータのスタックのために専用のレジスタ（スタック・レジスタや裏返しレジスタ；ALTERNATE REGISTOR）を内蔵したり，内蔵レジスタのアドレスは専用のデータ・ポインタによって行なわせたりする。

タイマーやイベント・カウンタなどは当然のことながら，カウント・パルス源の選択ができる。また，カウント周波数が高い場合にはカウンタの前段にプリ・スケーラを挿入してカウント動作に入ることもできる。また，モデューロ・レジスタに値を設定し，モデューロ計数を行なうこともできる。シリアルI／Oは他の制御機器との間でのデータのやりとりをする目的で使用される。LCDコントローラなどは，このシリアルI／Oを介してシングルチップ・マイコンに接続される。

英数字を表示するには，セグメントに分割された表示素子を一般に使用する。点灯するセグメントの組み合わせによって個々の英数字を表現する。英数字の桁毎にデコータやドライバを配置するスタティック点灯の方法は殆ど用いられない。点灯する桁を動的にスキャンしていくダイナミック点灯が一般的である。蛍光表示管も発光ダイオードも点灯方式は同じである。スキャンをしていく桁信号とその桁の表示セグメントに合致するセグメント信号とをタイミングよく表示素子に供給してやる。但し，桁間での表示の〔にじみ〕をなくすため，桁信号に表示消去信号を挿入する必要がある。液晶は寿命などの関係からダイナミック駆動が必須である。液晶は同じ状態に留ることがないように，電位を均等に変化させ，かき混ぜる必要がある。このため，駆動法が若干異なる。コモン信号とセグメント信号は互いに逆相であって，常に信号レベルは変化している。電圧3分割法では，非点灯時には1／3の電圧変化，点灯時には1の電圧変化が液晶に与えられるようにする。英数字の表示だけであれば，コモン信号4本で表現可能である。従って，シングルチップ・マイコンが持っている端子数だけで12桁程度の英数字であれば直接ドライブができ

る。セグメントではなくドットに分割した液晶パネルをグラフィックス表示用として使用することもできるようになってきた。シングルチップ・マイコンのみによって制御することは，端子数の関係からできない。行／列両方向に多数のドライバをふんだんに使用することになる。1／100デューティ，電圧6分割法を用いて，640×200ドット程度の白黒表示が可能になってきている。

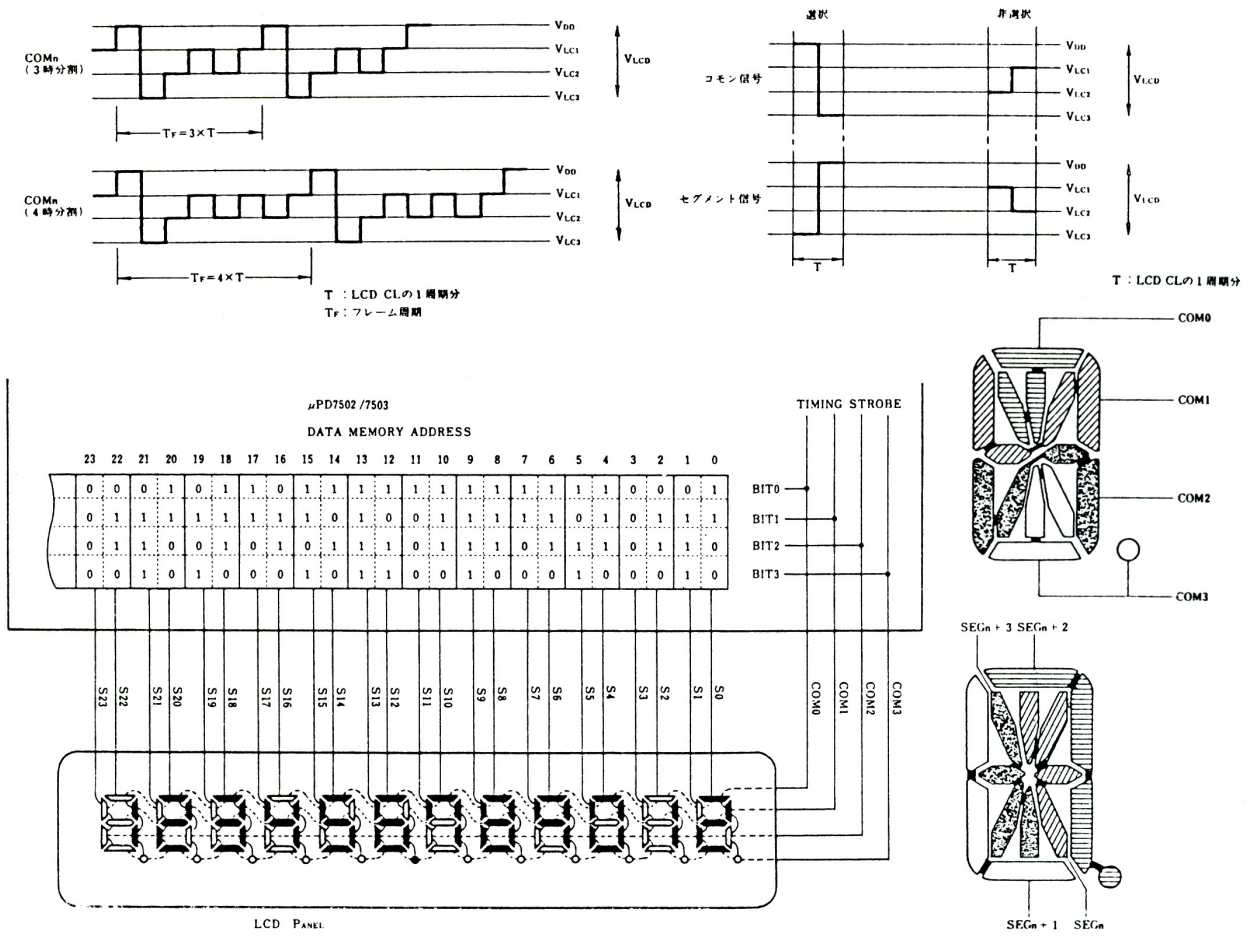



**図3.43　液晶ドライブ例**

ユーザが開発したプログラムは最終的にチップ上のROMに焼きつけられることになるが，新しくフォトマスクを作成するため，開発費用がユーザ負担となる。使用数量が少ないと，製品価格を上昇させる主因となる。そこで，消去可能ROM（EPROM；ERASABLE PROGRAMMABLE READ ONLY MEMORY）をオンチップ化したり，一般に市販されているEPROMをシングルチップ・マイコンの表面に取りつけた端子に直接接続したりできるようになった製品も開発されている。

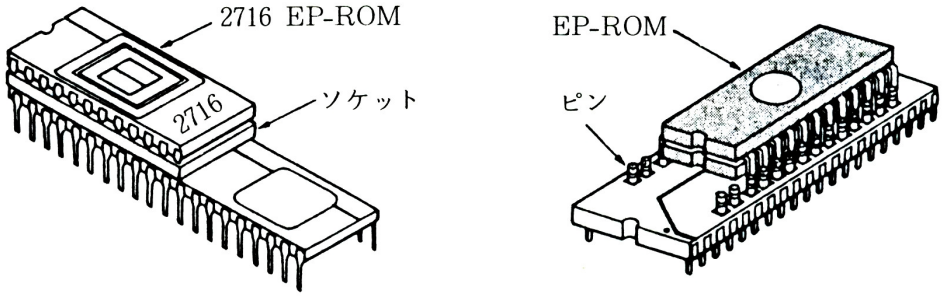



**図3.44　EPROMソケット付きマイコン**

プログラムをROM化するということは，

チップ上にあらかじめレイアウトされているROMのアドレス・デコーダ部を除いたメモリ部分のトランジスタ位置を変更することに相当する。LSIメーカとしては，できるだけ少ないフォトマスク変更で済ませたいし，マスク変更が必要となる工程は出来るだけ最後に近い工程とし，製品受注から出荷までのＴＡＴ（TURN AROUND TIME）を早めたいなどの理由から色々な方法が考えられている。マスク工程の早い順番に並べると次のようになる。

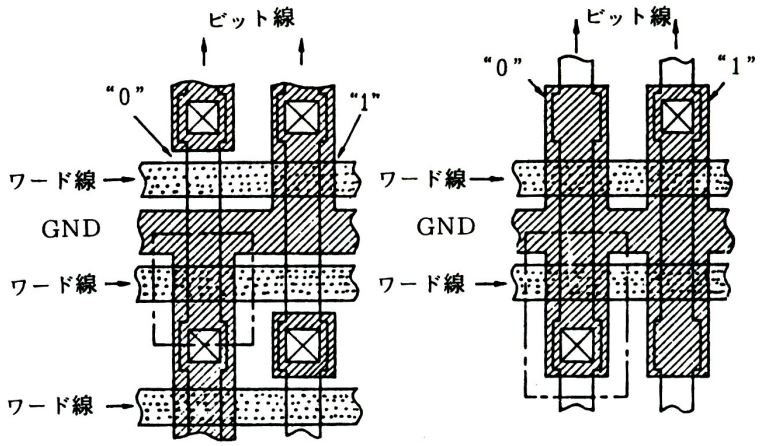



図 3.45　拡散層切り換えとコンタクト切り換え

(1)　拡散層切り換え
(2)　イオン注入切り換え
(3)　コンタクト切り換え
(4)　出力アルミ切り換え

マスク・レイアウト上，メモリ部分の面積を小さくできるのは，(1)と(2)である。どの方法を採るかはLSIメーカのポリシーに依存する。

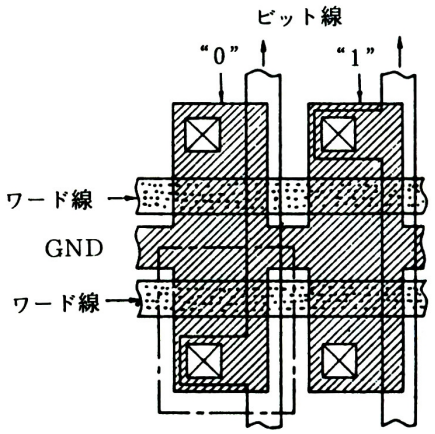



図 3.46　アルミ切り換え

## 8.　周辺コントローラLSI

CPUの機能を補助し，周辺装置を制御する目的で種々の周辺コントローラLSIが開発されてきた。CPUと，これらのLSIとの間のデータのやりとりや，動作設定指示などの伝達方法には次のような方法がある。

(1)　コマンド／パラメータ方式
　　(A)　FIFOバッファなし
　　(B)　FIFOバッファあり
(2)　トークン方式
(3)　コ・プロセッサ方式
(4)　直接アドレス方式

(1)－(A)はインテル社が開発した周辺コントローラLSIで採用されている一般的なインタフェース法である。コマンドによって動作指示をし，続いて送出する複数のパラメータによって，動作を決定する。コマンドとパラメータとを弁別するために，アドレス端子を必要とするためDMA転送によってコマンド／パラメータを設定できなくなるのが難点である。(1)－(B)は日本電気のグ

ラフィックス・ディスプレイ・コントローラμPD7220で初めて採用された。CPU側とコントローラ側での処理速度の調整をするために有効である。(2)は(1)と異なり，アドレス端子を必要としない。(1)のコマンドに相当するトークンが，パラメータに相当するデータをサンドイッチにした形で送出される。コントローラをリセットした後，初めて送出される情報がトークンであるとの判別をしている。このため，マルチ・タスクにおいて，トークン／データの位置関係が崩されると正常動作に復帰することができなくなることがある。トークンを付随させねばデータの転送ができないので，転送語数が増加する欠点を持つ。(3)はインテル社のコ・プロセッサで使用されている方法である。前述したように，特殊なインタフェース法を採るので，インタフェースできるCPUは限定されてしまう。(4)はコントローラが内蔵しているレジスタをCPU上にマップし直接アクセスする方法である。(2)におけるトークンはデータと一緒にアドレスとして与えることができるので，転送語数は少なくて済む。

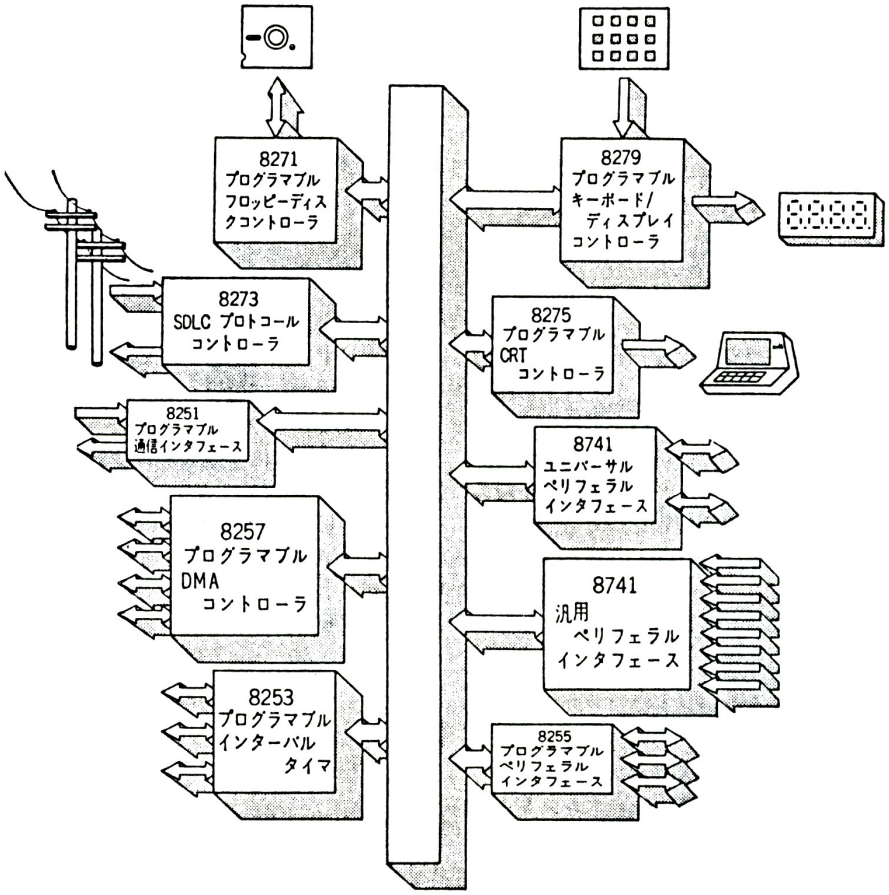



**図 3.47　周辺コントローラLSIの例**

さらに最近の周辺コントローラLSIは自分自身でアドレスを発生し，必要なデータを主記憶などから読み取り，処理を実行していく高度にインテリジェント化されたものもある。

主要な周辺装置を，その処理速度が遅くてもよい順番（割込やDMA転送の優先順位が低い順番としてもよい）に並べると，次のようになる。

(1)　キーボード
(2)　プリンタ
(3)　フロッピーディスクなどの補助記憶
(4)　CRT／液晶などの表示装置

キーボードやマウスなど人間が直接触れて，データを入力する周辺装置は，CPUの側からみると，非常に遅い入力装置であるといえる。さらに，これらの機器から指示された個別処理に多少時間がかかったとしても，オペレータは気付かない。

キーのオン／オフ時，ある期間，バウンシングとかチャタリングと呼ぶ不安定状態が必ず発生するので，この期間はキーのセンスを一時中断する。キーを同時に二重押しした場合には，そのキーを無視したりロール・オーバ処理する。キー入力バッファを設けたキー入力速度の調節をすること，繰り返しキー入力の制御，さらに，キー走査信号の発生などがキーボード制御の主要な機能である。これらは，すべて，シングルチップ・マイコンで実行されており，CPUとのインタフェースは割込による並列インタフェースやシリアル・インタフェースが用いられる。

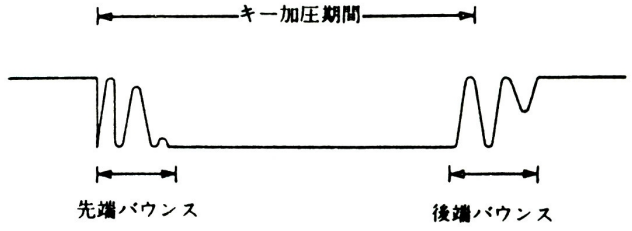



図 3.48　キー・バウンシング

プリンタの制御もキーボードの場合と同じく，シングルチップ・マイコンが使用されている。ライン・プリンタに代表されるパラレル・プリンタと，ドット・マトリックス・プリンタに代表されるシリアル・プリンタとに大別できる。ＣＰＵとのインタフェースは，通常，セントロニクス・並列インタフェースが使用される。ドット・プリンタでは，プリンタ・ヘッドの動きに合わせて，漢字などの文字コードをドットに展開したり，シリアル・イメージ・データの並べ替えをすることが主な制御である。

| | プリンタ | | プリンタ |
|---|---|---|---|
| 1 | DATA STROBE | 19 | ツイスト・ペア・グラウンド 1 |
| 2 | $DATA_1$ | 20 | ツイスト・ペア・グラウンド 2 |
| 3 | $DATA_2$ | 21 | ツイスト・ペア・グラウンド 3 |
| 4 | $DATA_3$ | 22 | ツイスト・ペア・グラウンド 4 |
| 5 | $DATA_4$ | 23 | ツイスト・ペア・グラウンド 5 |
| 6 | $DATA_5$ | 24 | ツイスト・ペア・グラウンド 6 |
| 7 | $DATA_6$ | 25 | ツイスト・ペア・グラウンド 7 |
| 8 | $DATA_7$ | 26 | ツイスト・ペア・グラウンド 8 |
| 9 | $DATA_8$ | 27 | ツイスト・ペア・グラウンド 9 |
| 10 | ACKNOWLEDGE | 28 | ツイスト・ペア・グラウンド10 |
| 11 | BUSY | 29 | ツイスト・ペア・グラウンド11 |
| 12 | PE（用紙切れ） | 30 | ツイスト・ペア・グラウンド31 |
| 13 | SELECT | 31 | INIT |
| 14 | AUTO FEED | 32 | FAULT |
| 15 | NC | 33 | 0 V |
| 16 | 0 V | 34 | 未定義 |
| 17 | FG（筐体グラウンド） | 35 | |
| 18 | ＋5 V | 36 | |

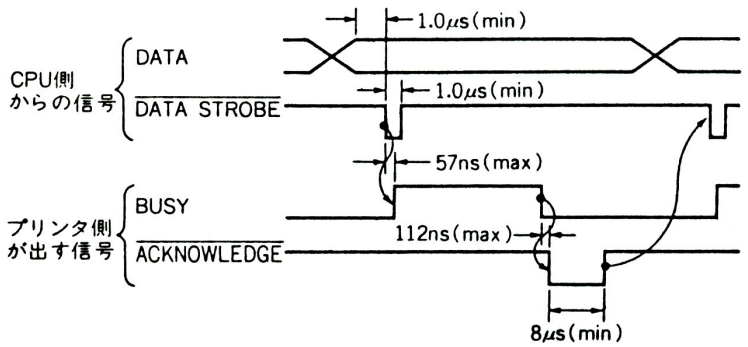
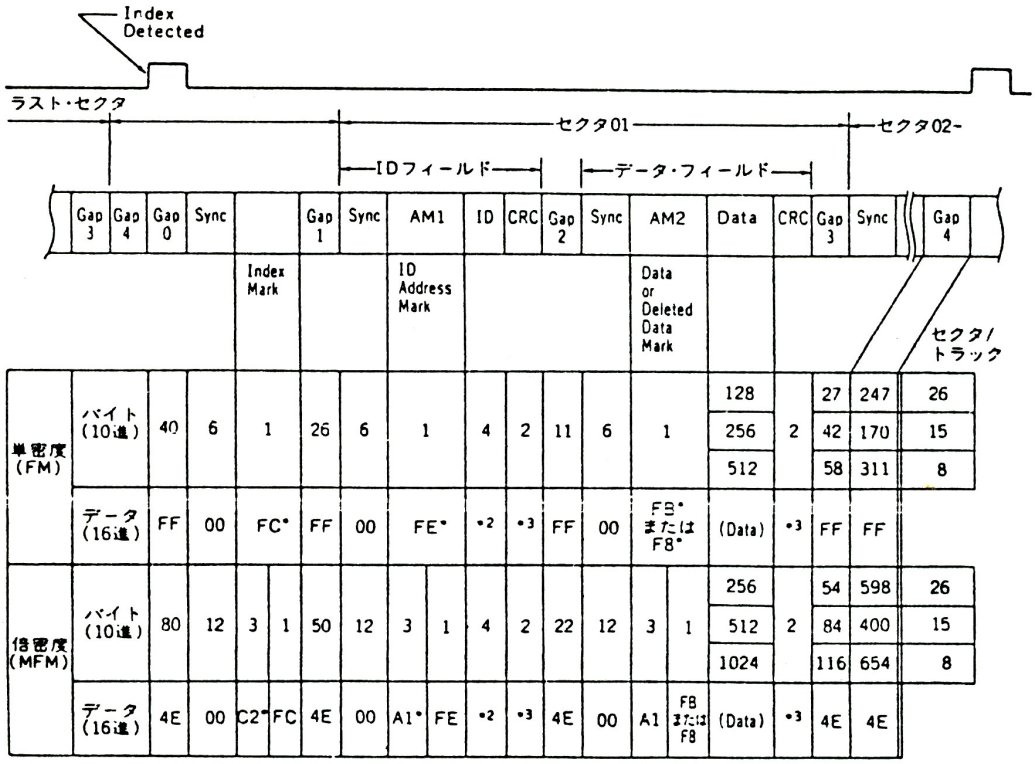



図 3.49　セントロニクス・インタフェース

フロッピーディスク制御になると，汎用型シングルチップ・マイコンではなく，専用の

| ディスク | セクタ数 | トラック00 | | トラック01～76 | |
|---|---|---|---|---|---|
| | | サイド0 | サイド1 | サイド0 | サイド1 |
| Diskette 1 | 26 | FM, 26 セクタ | | FM, 26 セクタ | |
| | 15 | 〃 | | FM, 15 セクタ | |
| | 8 | 〃 | | FM, 8 セクタ | |
| Diskette 2 | 26 | 〃 | FM, 26 セクタ | FM, 26 セクタ | FM, 26 セクタ |
| | 15 | 〃 | 〃 | FM, 15 セクタ | FM, 15 セクタ |
| Diskette 2D | 26 | 〃 | MFM, 26 セクタ | MFM, 26 セクタ | MFM, 26 セクタ |
| | 15 | 〃 | 〃 | MFM, 15 セクタ | MFM, 15 セクタ |
| | 8 | 〃 | 〃 | MFM, 8 セクタ | MFM, 8 セクタ |

注) トラック0, サイド0はすべて単密度26セクタに統一されている. FM：単密度　MFM：倍密度




* このデータ・バイトはミッシング・クロックを含む
*2 IDフィールド
*3 CRCチェック・バイト　公式 $G(X) = 1 + X^5 + X^{12} + X^{16}$

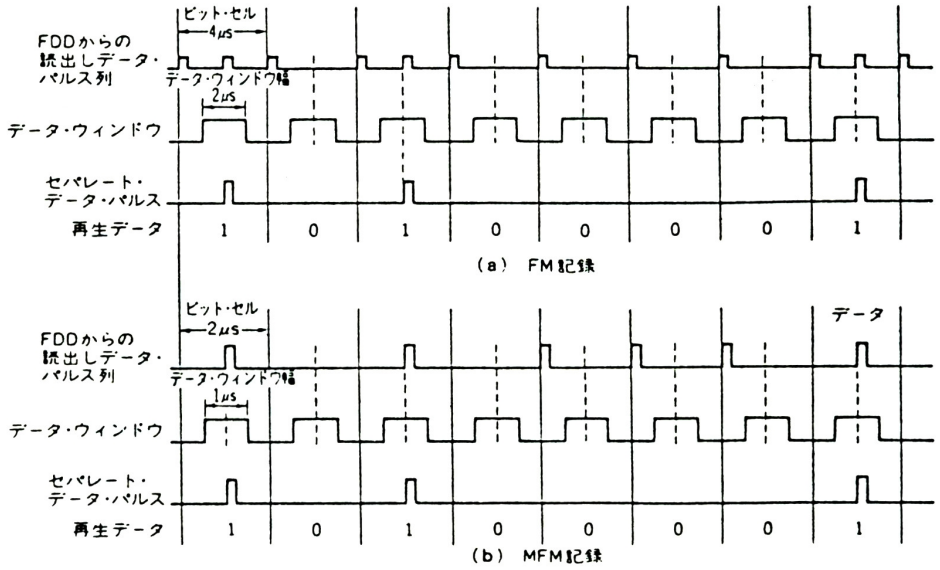



図3.50　IBMフォーマット, FM／MFM

周辺コントローラLSIが使用されるようになる。より速い処理が必要となるからである。フロッピーディスクはIBM社が決めたフォーマットに従って，データが記録されているものが多い。片面単密度／両面単密度／両面倍密度／両面倍倍密度のように，記録容量は増大してきた。ディスクの大きさも，8／5.25／3.5／3インチと種類が増えた。シリンダ／トラック／セクタに分け，各記録位置を定義する。ハード・セクタ方式とソフト・セクタ方式とがあるが，フロッピーディスクや最近のハードディスクでは，インデックス・ホールの位置確認だけでセクタ位置を決めていくソフト・セクタが主である。このため，モータの回転ムラやインデックス・ホールの検出誤差などの影響をなくすため，無効データを逐次，有効データの間に挿入していくディスク・フォーマットとなっている。アナログ信号系の処理やＶＦＯによるデータ・ウィンドウの作成など外付け回路で行なう処理は多い。フォーマティング部分を担当するデータ・フォーマッターという呼び方をする場合もある。

ＣＲＴの表示制御をするコントローラを主記憶と表示メモリとのシステム構成で分類すると次のようになる。

(1) ビデオRAM型

(2) DMA転送行バッファ型

(3) 独立メモリ型

さらに，ＣＲＴに供給する同期信号の発生機能や表示制御機能だけではなく，描画機能を内蔵しているコントローラもある。通常の文字表示制御に加え，表示画面上の1ドットが表示メモリの1ビットに対応しているグラフィックス制御も可能である。直線／円弧／塗

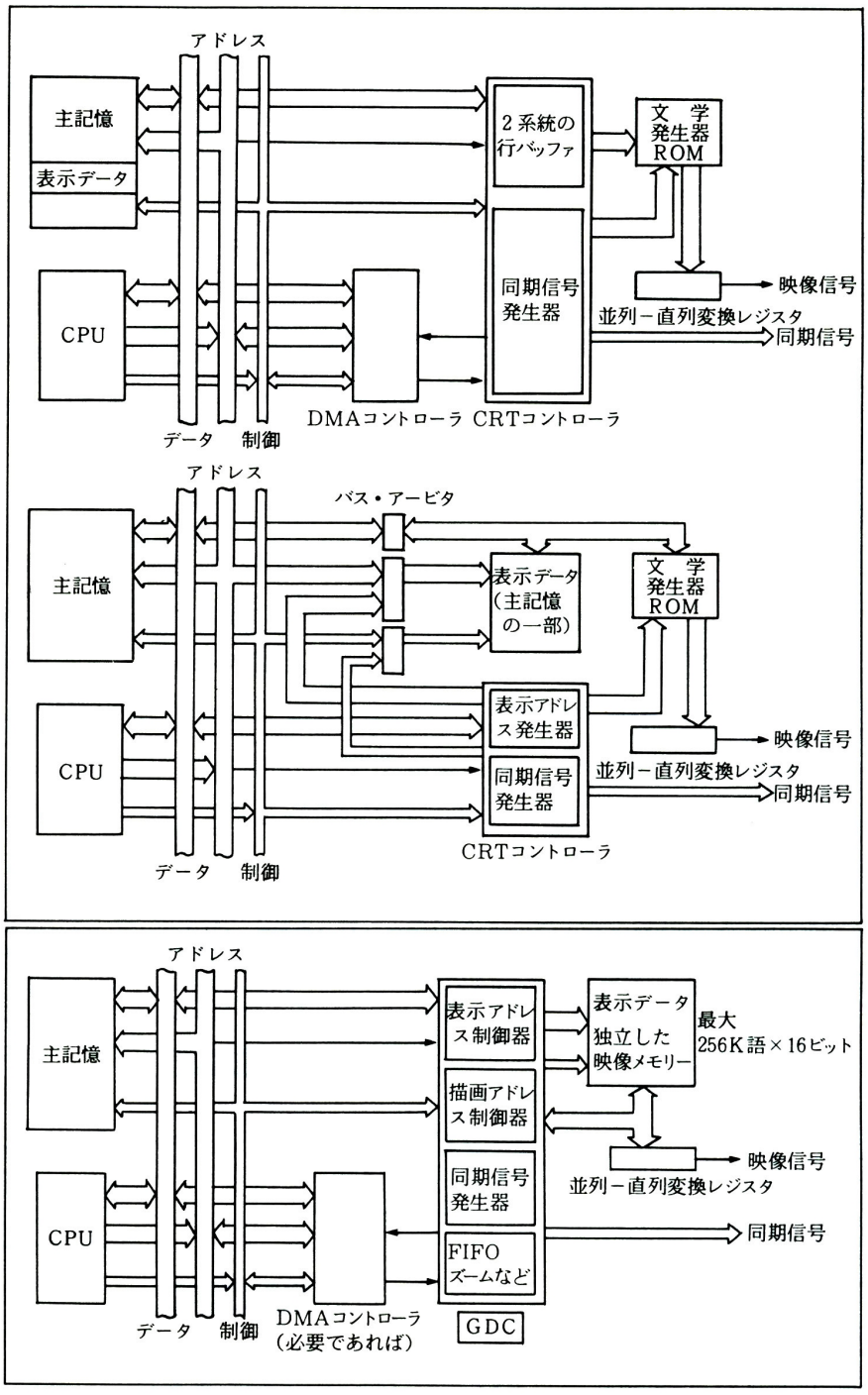



**図 3.51　主記憶と表示メモリのシステム構成**

りつぶしなどを高速に実行できる。CRTのかわりに，液晶パネルを表示素子として使用してもよい。液晶ドライバが，液晶表示に必要となる電位差信号を発生してくれる。ただ，640×200ドットの液晶パネルをドライブするには，1個数百円もする32本の駆動線を持つ行ドライバを4個，40本の駆動線を持つ列ドライバを32個も使用せねばならないことが普及を妨げている。

DMAコントローラ8237は4チャンネルのDMA要求源からの信号を受付け，DMA転送を実行できる。優先順位をつけたり，平等にサービスしたりすることもできる。DMA要求信号（DMA REQUEST）を受け取ると，CPUに対しホールド要求（HOLD REQUEST）を出す。CPUはホールド承認信号（HOLD ACKNOWLEDGE）を返し，バスの制御権をコントローラ側に移す。DMAコントローラは転送をするメモリのアドレスをアドレス・バスに出力すると同時に，DMA要求源に対し，チップ・セレクトに相当するDMA承認信号（DMA ACKNOWLEDGE）を返す。また，DMAの転送方向指定に従って，RD／WR信号を出力する。DMA転送方向として次の3種が選択できる。

(1) メモリ→I／O

(2) I／O→メモリ

(3) メモリ→メモリ

メモリ→メモリの転送の場合には，1語の転送につき2回のメモリ・アクセスを必要とするため，転送速度は(1)，(2)の場合と比較して遅くなる。アドレス線が16ビットを超えるシステムの場合であっても，バンク・アドレス・レジスタを各チャンネルに外付けして，バンク内64K語の連続的なDMA転送を可能とすることができる。

割込コントローラ8259Aは8チャンネル分の割込要求を制御できる。さらに，カスケード接続をすることにより，要求源を増加させることもできる。各チャンネルの受付け優先順位の回転やマスクが可能である。各チャンネル毎に異なる割込ベクタを発生できる。8080／8085モードと8086モードの2種のモード選択によって，ベクタ発生方法が異なる。8080は割込コントローラからの割込要求を受け取ると，割込承認信号（INTERRUPT ACKNOWLEDGE）を3回，8086は2回返す。この際，割込コントローラは8080モード設定時，3バイトのサブルーチン・コール命令を，8086モード設定時には2バイト目にソフトウエア割込の場合と同じ割込ベクタをバス上に出力する。このように，割込コントローラの動作は，使用するＣＰＵと密接な関わりを持つ。このため，総てのＣＰＵにインタフェースできるわけではない。

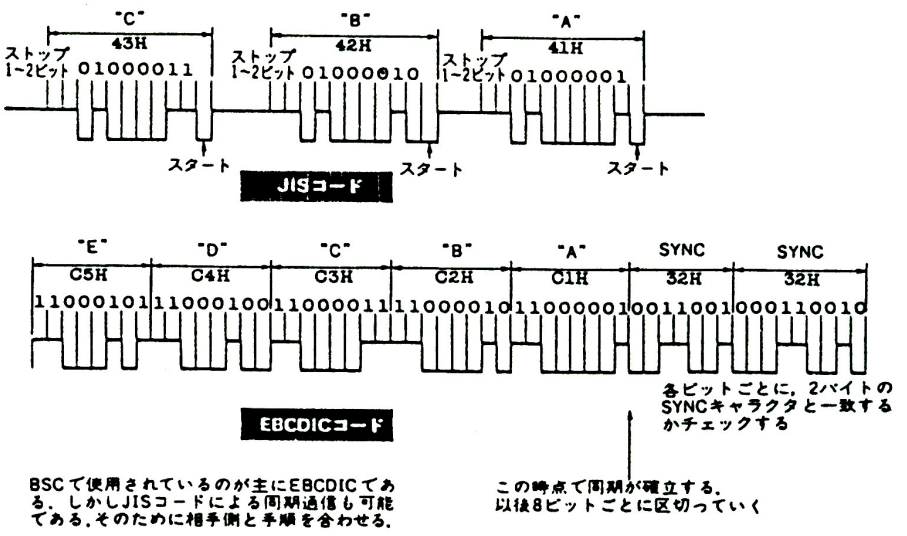



図 3.52　非同期／同期通信

USART（UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER AND TRANSMITTER）とも呼ばれるシリアル・コントローラ8251は，RS-232Cシリアル・インタフェース用として，よく使用される。シリアル信号はバイト単位に区切られて転送され，その先頭ビットはLSBである。非同期転送では，スタート・ビットとストップ・ビットにはさまれたデータを転送する。1バイト毎に分離されたデータを送ることになる。同期転送では，SYNCバイトにはさまれ連続したバイト・データを送るので，まとまったデータを短時間で送る場合に適している。

パラレル・インタフェース用LSI8255は8ビットのポートを3組持つ。各々のポートは入出力を任意に設定できる。シングルチップ・マイコンのポート制御機能の原型となったLSIである。

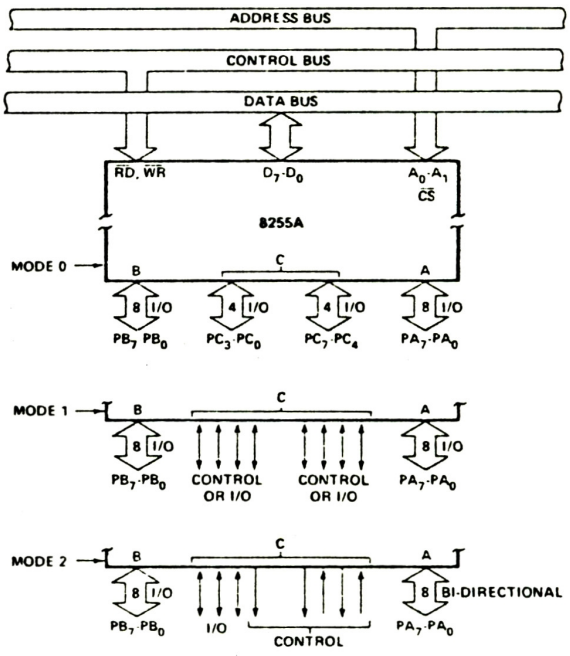



**図 3.53　8255動作モード**

## 9. 今後の動向

単位面積当たりの集積度が大きくなり，多くのゲートを1個のチップに集積できるようになるにつれて消費電力が益々増加し，また，NチャンネルMOSに比べチップ面積が約2倍になるCMOS構成にしても採算が取れる程度の歩留まりが得られるようになってきた。このため，今後開発される集積回路製品は殆どがCMOS構成となることは間違いない。マルチチップ・マイコンや周辺コントローラは，より複雑で高度な機能を集積することになり，シングルチップ・マイコンは周辺制御機能のオンチップ化が推進されていくことになる。

**〔参考文献〕**

(1) 「マイクロプロセッサ」，R.ZAKS著，禿節史訳，マイテック
(2) 「ザ8086ブック」，RASSEL RECTER／GEORGE ALEXY著，吉川敏則訳，産報
(3) 「マイクロコンピュータとその応用」，電子通信学会編，コロナ社
(4) 「ディジタル回路」，斎藤忠夫著，電子通信学会編，コロナ社
(5) 「16ビットマイクロプロセッサ」，森亮一監修，電子通信学会編，丸善
(6) 「16ビットマイクロプロセッサ」，御牧義著，昭晃堂
(7) 「LSI技術」，電子通信学会編，コロナ社
(8) 「Z80マイクロコンピュータ」，寺田浩詔著，丸善
(9) 「INTEL COMPONENT DATA CATALOG」，INTEL

⑽ 「マイクロコンピュータ教科書」，森亮一監訳，丸善
⑾ 「マイクロコンピュータの開発技法」，石田芳著，産報
⑿ 「超LSI入門」，太田邦一著，オーム社
⒀ 「新版データ通信」，平山博著，オーム社
⒁ 「LSI応用」，電子通信学会編，コロナ社
⒂ 「シングル・チップ」，日本電気株式会社