お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
3. 当社製品を改造、改変、複製等しないでください。
4. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害などを生じさせないようお客様の責任において冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、詳細につきましては製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

注 1. 本資料において使用されている「当社」とは、ルネサスエレクトロニクス株式会社およびルネサスエレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

RENESAS

保守／廃止

IE-78C11 スタンドアロン取扱説明書

MD シリーズ用

NEC 日本電気株式会社

EEP-615A
July 1987P

保守／廃止

IE-78C11 スタンドアロン取扱説明書

MD シリーズ用

NEC 日本電気株式会社

保守／廃止

目　次

保守／廃止

保守／廃止

保守／廃止

第１章　概　説

１．１　序　文

ＩＥ－７８Ｃ１１（Ｉｎ－ｃｉｒｃｕｉｔ Ｅｍｕｌａｔｏｒ ｆｏｒ μＰＤ７８Ｃ１１）はμＰＤ７８Ｃ１０，７８Ｃ１１，７８Ｃ１４を用いたシステムのハードウェアおよびソフトウェアの開発を効果的に行なうための開発サポート・システムです．

このＩＥ－７８Ｃ１１システムはドライバ，エミュレーション・プローブおよびＩＥコントロール・ボードにより構成されており，μＰＤ７８Ｃ１０，７８Ｃ１１，７８Ｃ１４の機能をほぼ完全にエミュレーションしています．このシステムは，以下に述べるように各種のエミュレーション機能，ブレーク機能，トレース機能，メモリ・マッピング機能等を備えているだけでなく，ＮＥＣマイコン開発用ホスト・システム（ＭＤシリーズ）とのコミュニケーション機能など将来の拡張に十分対応した設計となっています．

対象ＣＰＵはμＰＤ７８Ｃ１０，μＰＤ７８Ｃ１１，μＰＤ７８Ｃ１４であり，これらのディバグが可能です．

保守／廃止

１．２　ＩＥ－７８Ｃ１１基本仕様

ＩＥ－７８Ｃ１１仕様（スタンドアロン・ベース）

対象ＣＰＵ		μPD78C10，μPD78C11，μPD78C14
動作ＣＰＵクロック		最大15MHz
メモリマップ	ユーザ・メモリ	0 ～FFFFH(64Kバイト) のアドレス空間を256バイト単位でマッピング 可能
	エミュレーション・メモリ(標準64Kバイト装備)	物理アドレス空間(64Kバイト)を256バイト単位で マッピング可能 (リアルタイム 実行)
ブレーク	ブレーク・レジスタ	1.ロジカル・ブレーク・レジスタ BR0 ～ BR3 (4種類) BR0 ～ BR3の中でさらにフィジカル・ブレーク・レジスタ の設定が可能 2.フィジカル・ブレーク・レジスタ ① BRA アドレス,コンディション,データの設定が可能 ② BRD (外部センス・クリップのデータ でブレーク) ③ BRE (オペコード・フェッチ回数) ④ BRT (ブレーク・タイマ) 3.ブレーク・モード・レジスタ BRM (各ブレーク・レジスタ をハード・ウェア にセット します)
	設定条件	1.アドレス ポイント数(複数箇所),マスク 可, 範囲指定可 (64Kバイト内) 2.コンディション オペコード・フェッチ(M1リード),メモリ・リード,メモリ・ライト,メモリ・リクエスト,条件なし 3.データ ポイント数(複数箇所),マスク 可 4.オペコード・フェッチ回数(3～FFFFH) 5.ループ 回数(1～FFH) 6.タイマ (1～FFFFH ms) 7.外部センス・データ
	ブレーク要因	1.アドレス　2.データ 3.コンディション (オペコード・フェッチ(M1リード),メモリ・リード,メモリ・ライト, メモリ・リクエスト,条件なし) 4.外部センス・データ　5.インストラクション・カウント 6.タイム・カウント 7.MAP 違反 (マッピング 領域以外をアクセスしてブレーク) 8.強制ブレーク (キー入力によるブレーク)
トレース	トレース 容量	1023フレーム
	トレース 表示	インストラクション,マシン・サイクルの２種類選択可能
	トレースオン／オフ条件	下記のいずれかを選択 1.NON (トレース しません) 2.ALL (エミュレーションCPUの実行をすべてトレース) 3.TRX(TRX で指定された条件に従います)
	トレース内容	アドレス(16 本)/ データ(8本)/M1/RD/WR ポートA(8本)/ポートB,または外部センス・データ(8本)
外部 トレース・プローブ		8 本
シリアル・チャネル		2チャネル (RS-232-C) 110～9600ボー 1.TTY1 (コンソール 用) 25ピン・コネクタ 2.TTY2 (サブI/O 用) 16ピン・コネクタ
表　示		エミュレーションCPU(μPD78C10)の動作状態をLEDにより表示します。
その他		オンライン・アセンブル 可能

保守／廃止

１．２．１　エミュレータ部

（1）エミュレーション・プローブ	６４ピン・ＩＣソケット挿入形式を使用
（2）ＣＰＵ	μＰＤ７８Ｃ１０（１５ＭＨｚ）
（3）信号ピン・インタフェース	ＨＳＣＭＯＳによる１バッファ付（ＰＦ，ＰＤ）
（4）コントロール出力信号	エミュレーションＣＰＵ実行中はリアルタイムでコントロール
（5）$\overline{\text{RESET}}$，$\overline{\text{NMI}}$信号入力	完全エミュレート
（6）ＣＰＵ動作クロック	４ＭＨｚ～１５ＭＨｚ（外部クロック指定時） １２ＭＨｚ（内部クロック指定時）
（7）ＭＯＤＥ０，ＭＯＤＥ１	エミュレートしています。

１．２．２　ブレーク部

ブレーク機能はすべてハードウェアにより実現されています．

（1）ブレーク状態	内部モニタ・プログラム実行
（2）ブレーク機能	６種類

①　３組の独立したアドレス，データ，コントロール信号[注(1)]の組合せによるブレーク（ループ回数設定可能）

②　指定メモリ領域以外のアクセスによるブレーク

③　８本の外部センス信号の条件設定によるブレーク

④　内部タイマによるブレーク

⑤　コンソールからの強制ブレーク

⑥　マルチ・ディバッガ時の他のディバッガからのブレーク・トリガ信号によるブレーク

注１　ＯＰ－ＣＯＤＥ　ＦＥＴＣＨ（Ｍ１リード），ＭＥＭＯＲＹ　ＲＥＡＤ，ＭＥＭＯＲＹ　ＷＲＩＴＥ

保守／廃止

１．２．３　トレース部

（１）三つのトレース・モード
- ①　ノン・トレース
- ②　インストラクション毎のトレース
- ③　マシン・サイクル毎のトレース

（２）二つのトレース機能
- ①　常時トレース
- ②　指定された条件（アドレス，データ，コンディション，ポートＡ，ポートＢ）範囲のトレース

（３）トレース項目
- ①　アドレス・バス，データ・バス
- ②　$\overline{M1}$，$\overline{RD}$，$\overline{WR}$
- ③　ポートＡ，ポートＢまたは８本の外部センス信号

１．２．４　マッピング部

（１）マッピング状態
- ①　ノン・マッピング（初期状態では，プログラム・メモリはノン・マッピング
- ②　外部マッピング（ＵＳＥＲ）
- ③　内部マッピング（エミュレーション・メモリ）

（２）マッピング対象　；　６４Ｋバイト全域

（３）マッピング単位　；　メモリ２５６バイト単位

１．２．５　表　示　部

エミュレーションＣＰＵ（μＰＤ７８Ｃ１０）の動作状態をドライバ上面パネル上のＬＥＤにより表示します．

（１）スタンバイ・モード表示

①ＨＡＬＴ　：　ＨＬＴ命令を実行し，ＨＡＬＴモードに入ると点

保守／廃止

灯します.

②SOFT STOP ： STOP命令を実行し，ソフトウェアSTOPモードに入ると点灯します.

③HARD STOP ： $\overline{\text{STOP}}$端子にLOWレベルが入力され，ハードウェアSTOPモードに入ると点灯します.

（2）ラッチ・アップ表示

①LATCH UP ： ドライバ内部のCMOS ICがラッチ・アップを起こすと点灯します.

（点灯した場合は，1．5．5項を参照してください.）

1．2．6 その他の機能

（1）マルチ・ディバッガ機能

ホスト・システム（MD－086シリーズ）のスロットに最大2台まで同時に挿入（他IEシステムおよび同一IE－78C11と併用も可）して単一ユーザおよびマルチ・ユーザで利用可能.

（MD－086FDを使用している時は拡張シャーシMD－086EXを使用してください．MD－086FD－10では拡張シャーシは必要ありません.）

（2）ブレーク・トリガ入出力機能

ブレーク・トリガ信号をドライバ上面パネルの入出力端子より入出力できます.

①BREAK TRIGGER IN ： 他ディバッガからのブレーク・トリガ出力を入力することにより，他ディバッガと同期してブレークします.

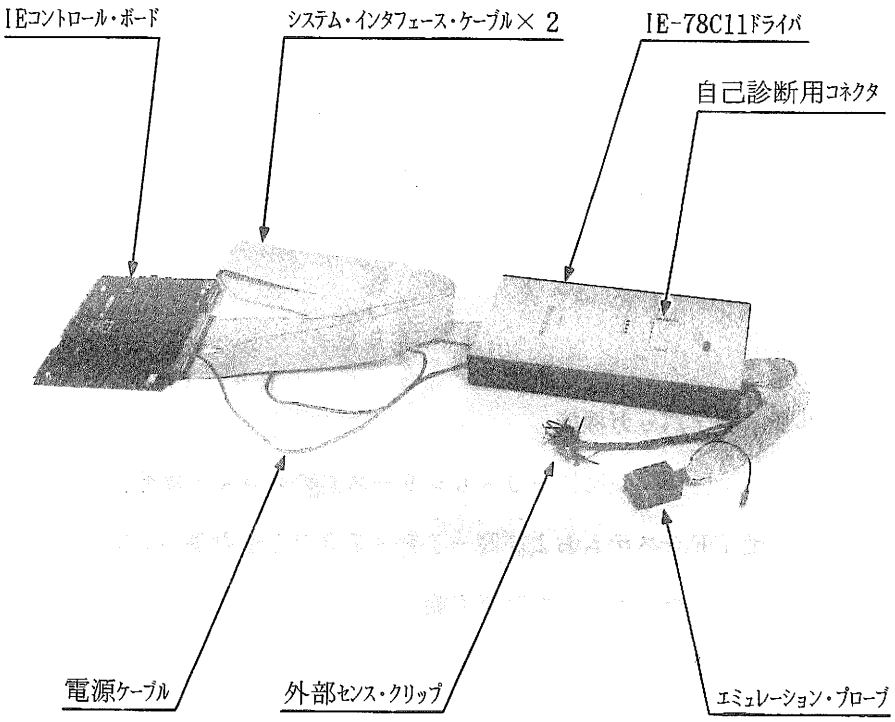
②BREAK TRIGGER OUT ： 他ディバッガへのブレーク・トリガ出力です．他ディバッガのブレークをコントロールします.

保守／廃止

１．３　ＩＥ－７８Ｃ１１ハードウェア仕様

１．３．１　外　観

写真１　ＩＥ－７８Ｃ１１　外観図



保守／廃止

１．３．２　基本仕様

（1）外形寸法

ＩＥコントロール・ボード	縦	２５０ｍｍ
	横	３０５ｍｍ
	高さ	約２０ｍｍ
ＩＥ－７８Ｃ１１ドライバ	縦	４００ｍｍ
	横	２３０ｍｍ
	高さ	４８ｍｍ

（2）重　量

ＩＥコントロール・ボード	５５０ｇ
ＩＥ－７８Ｃ１１ドライバ	２．６ｋｇ

（3）電　流	６．５Ａ以上（＋５Ｖ）
	５００ｍＡ以上（±１２Ｖ）
（4）使用温度範囲	１０～４０℃
（5）保存温度範囲	－４０～＋５５℃
（6）周囲湿度範囲	１０～９０％（ただし結露しないこと）

１．３．３　ＩＥ－７８Ｃ１１ドライバ・ブロック図

図１－１　ＩＥ－７８Ｃ１１　ドライバ・ブロック図

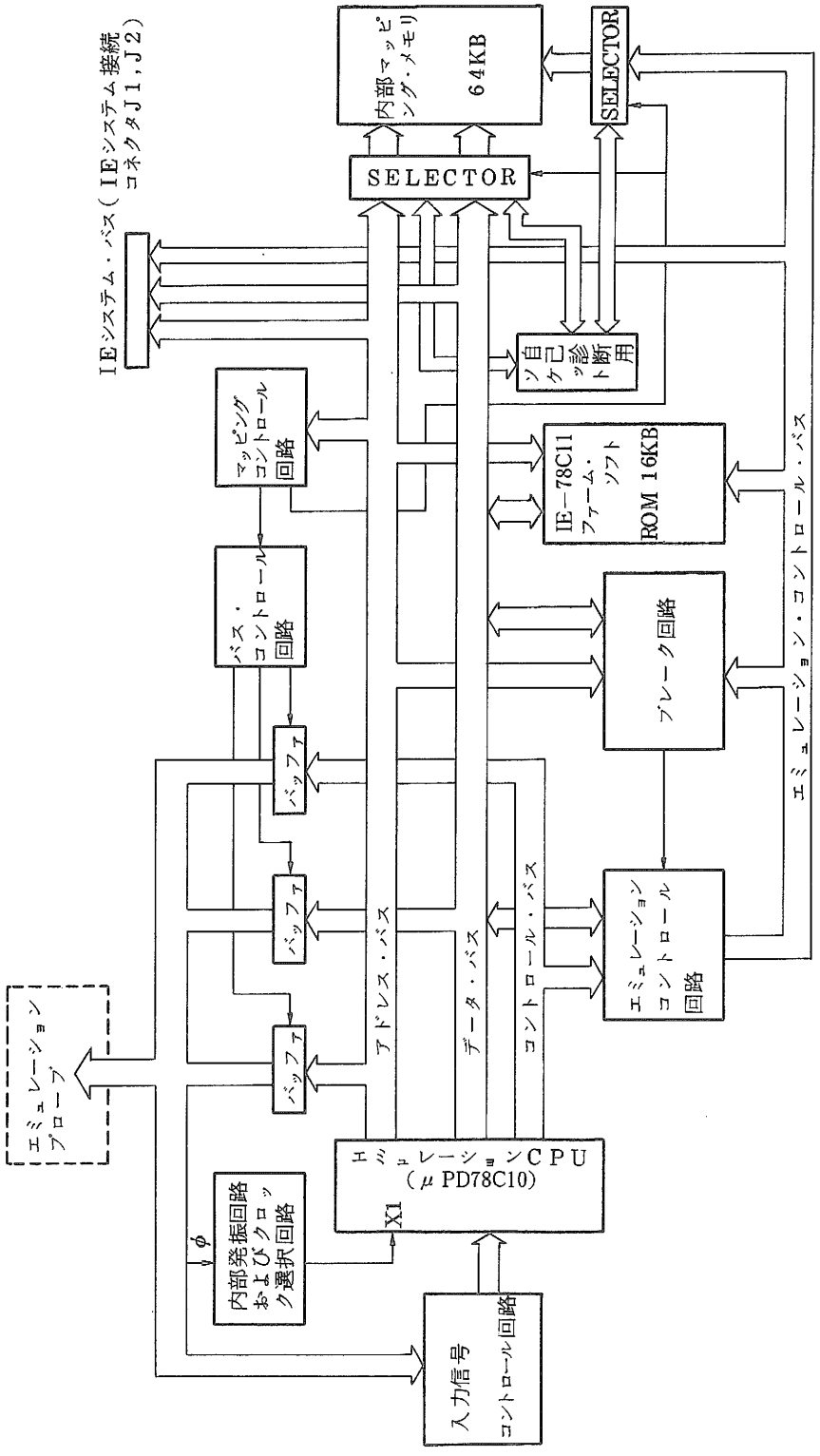
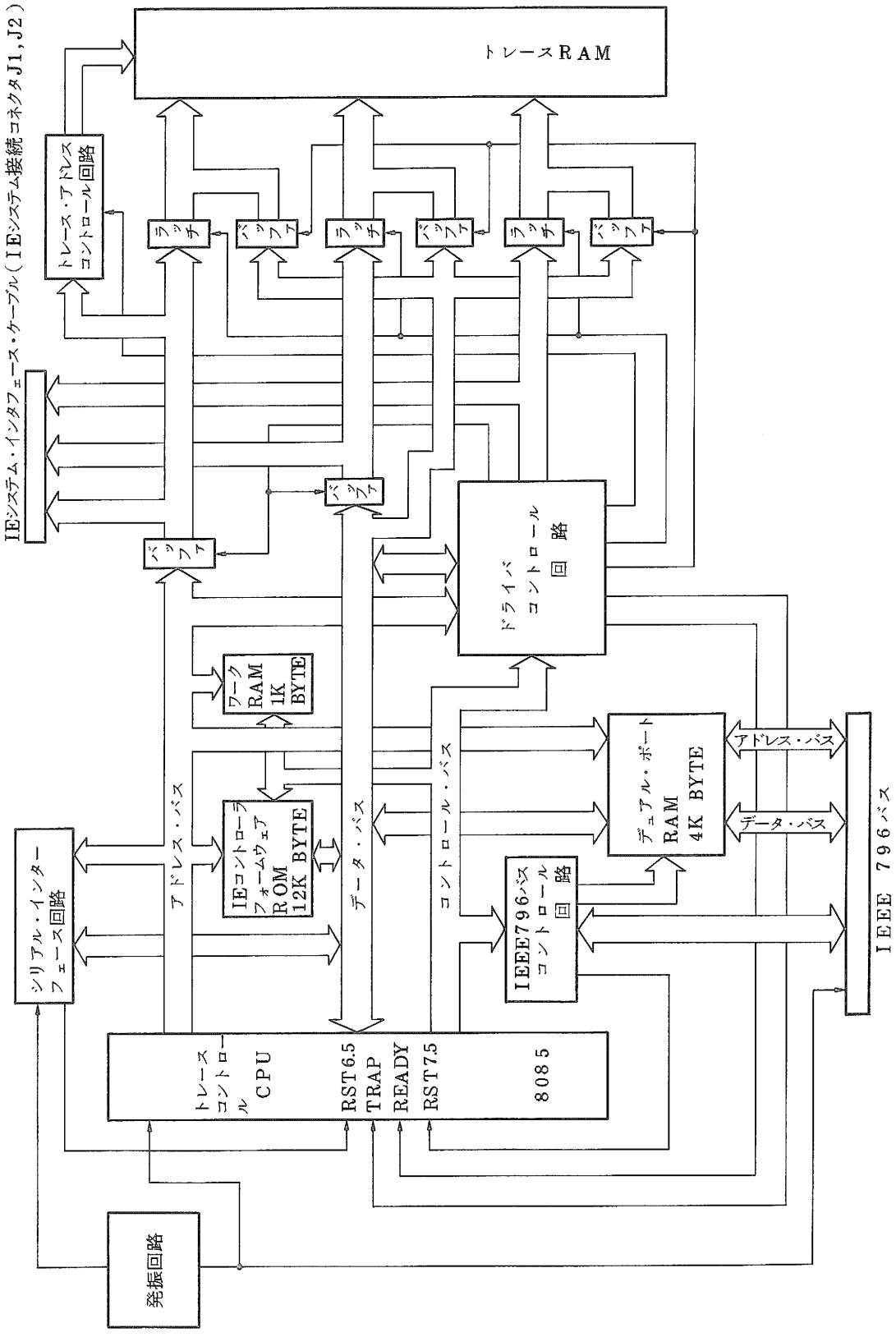


図１－２　ＩＥコントロール・ボード・ブロック図

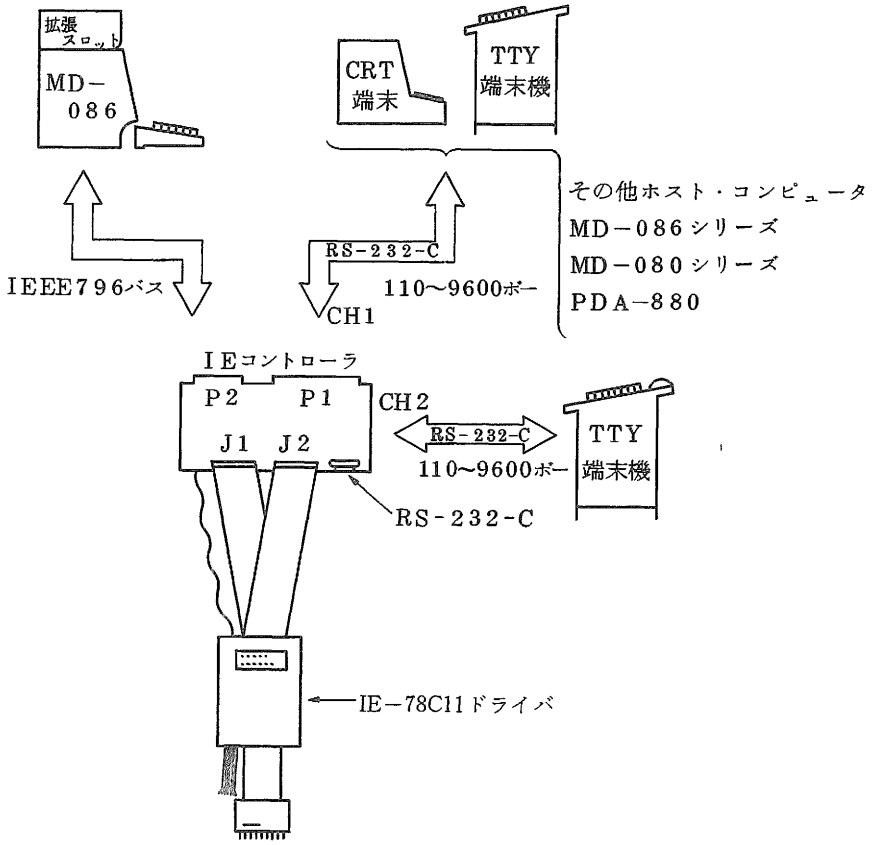


保守／廃止

１．３．４　ＩＥ－７８Ｃ１１システム構成図

ＩＥ－７８Ｃ１１は以下の形でシステムを構成して使用します．

図１－３



１．４　ＩＥ－７８Ｃ１１ソフトウェア仕様

１．４．１　概　要

ＩＥ－７８Ｃ１１標準のモニタ・ソフトは，４個のＥＰＲＯＭ（μＰＤ２７６４　１個，μＰＤ２７１６　２個，μＰＤ２７１２８　１個）の形で供給されます．そのうちμＰＤ２７１２８はＩＥ－７８Ｃ１１ドライバ・ボードに，μＰＤ２７６４とμＰＤ２７１６はＩＥコントロール・ボードに実装されております．

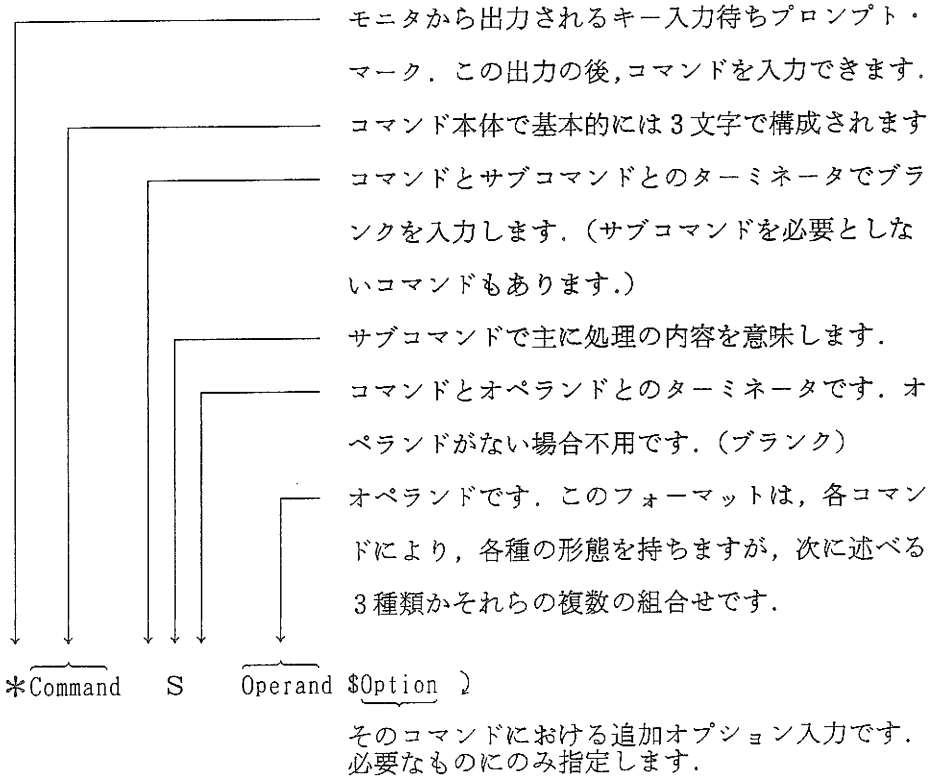
ＩＥ－７８Ｃ１１のモニタは，システムのセッティングが完了すればパワー・オンで実行可能です．

保守／廃止

1．4．2　コマンド形式

（1）コマンドは，1ライン入力形式になっており，以下の形式で入力します．

コマンド入力終了のターミネータは「改行復帰」入力です．



オペランドの種類

① 予約記号　　具体的な文字列でそれによりある処理を指定します．

② 数　　値　　具体的な数値を表現するもので式入力等があります．

③ 数 値 群　　具体的な数値の一群を表現するもので一般にその数値すべてに対して処理を行なうものをいいます．

保守／廃止

（2）コマンドの入力表現は，次のような表記形式で説明します．

大文字	キー入力する文字列または文字を意味します．
小文字	その位置に小文字の意味にしたがって入力することを意味します．（オペランド部参照）
⤶	改行復帰入力です．
␣	ブランク入力です．
{文字列 ： ：}	{ } の中に記述されている文字列のどれかを選ぶことを意味します．
[文字列 ： ：]	〔 〕の中に記述されている文字列の入力をしなくてもよいことを意味します．入力しなかった場合の処理は各コマンドで説明します．

注　一部この記号による表現を行なわず空白だけの表現も使用しています．

保守／廃止

（３）特殊キー入力

ＣＴＲＬ－Ｈ	１文字消去（カーソルは左へ一つ移動）
ＣＴＲＬ－Ｕ	１行削除（画面の削除はしません.）
ＣＴＲＬ－Ｘ	１行削除（画面からの削除）
ＣＴＲＬ－Ｅ	改行復帰（ただし，コマンドの実行はしません.）注(1)
ＣＴＲＬ－Ｒ	入力された１行の再表示
ＤＥＬ	１文字消去（カーソルは左へ一つ移動）
ＣＴＲＬ－Ｍ	改行復帰
ＣＴＲＬ－Ｊ	改行復帰
ＣＴＲＬ－Ｃ	コマンド実行の停止（ただし，エミュレーションＣＰＵが動いている時はブレークさせずに‘＊’に戻ります.）
ＥＳＣ	コマンド実行の停止（ただし，エミュレーションＣＰＵが動いている時は強制ブレークして‘＊’に戻ります.）
ＣＴＲＬ－Ｓ	出力の一時停止
ＣＴＲＬ－Ｑ	出力の再開
ＣＴＲＬ－Ｐ	ＣＨ１，ＣＨ２への同時出力 注(2)
ＣＴＲＬ－Ｄ	画面表示の停止（入力された時から画面表示をしませんが，実行は行なっています．再度入力すると元に戻ります.）

注１　コマンドの途中でＣＴＲＬ－Ｅを入力すると画面上は改行復帰します．しかしコマンド入力はそのまま続けることができます．かなり長いコマンドを連続して入力する時適当な所で改行をして見やすくするために利用します．

注２　ハード・コピーをとりたい時などＣＴＲＬ－ＰをＣＨ１から入力するとそれ以後のＣＨ１への出力はすべてＣＨ２へも出力されます．ＣＨ２への出力を解除するにはもう一度ＣＴＲＬ－Ｐを入力してください．

（４）キー入力文字数

入力文字数（ブランクも含む）は，最大１２４文字です．

１．４．３　コマンド一覧

①　メイン・コマンド部

ＭＡＰ　；　マッピング・コマンド

ＭＥＭ　；　メモリ・コマンド

ＲＥＧ　；　レジスタ・コマンド

ＭＤＲ　；　モード・レジスタ・コマンド

ＳＰＲ　；　特殊レジスタ・コマンド

ＬＯＤ　；　ロード・コマンド

ＳＡＶ　；　セーブ・コマンド

ＲＵＮ　；　エミュレーション・コマンド

ＢＲ？　；　ブレーク・コマンド
（？；Ａ，Ｄ，Ｅ，Ｔ，０，１，２，３，Ｍ）

ＴＲ？　；　トレース・コマンド（？；Ｃ，Ｄ，Ｍ，Ｐ，Ｘ，Ｓ）

ＣＬＫ　；　クロック・コマンド

ＲＥＳ　；　リセット・コマンド

ＭＡＴ　；　演算コマンド

ＳＵＦ　；　数値表現指定コマンド

ＡＳＭ　；　オンライン・アセンブル・コマンド

ＤＡＳ　；　逆アセンブル・コマンド

ＭＯＶ　；　メモリ転送コマンド

ＤＩＧ　；　自己診断コマンド

ＰＧＭ　；　ＰＲＯＭプログラマ制御コマンド

②　サブ・コマンド部

Ｂ　；　Ｂｒｅａｋ

Ｃ　；　Ｃｈａｎｇｅ

保守／廃止

D ; Display, Dump

E ; Examination

F ; Fill

G ; Get

H ; Hard

I ; IE, Instruction

K ; Kill

M ; Move, Mode, Machine cycle

N ; Normal

R ; internal Rom

S ; Step

T ; Trace

U ; User

V ; Verify

W ; internal r／W memory

X ; eXchange

③ オペランド部

addr	;	式入力可能な数値（2バイト）
addrx	;	'X' 入力可能な数値（2バイト）
partition	;	'X' 入力可能な数値領域
string	;	',' で区切られた、1バイト16進データ群
register－name	;	レジスタ名
mode－register－name	;	モード・レジスタ名
special－register－name	;	特殊レジスタ名
input－device－name	;	入力装置名
output－device－name	;	出力装置名

保守／廃止

ｖａｌｕｅ　；　数値（１バイト）

ｖａｌｕｅｓ　；　‘Ｘ’入力可能な数値（１バイト）

ｃｏｎｄ　；　ＣＯＮＤＩＴＩＯＮ
複数設定可能

ｓｔｅｐ　；　ステップ数（１バイト）

ｔｉｍｅ　；　時間単位（２バイト）（Ｍ＝ｍｓｅｃ）

ｄａｔａ　；　１バイト・データ

ｌｉｎｅ－Ｎｏ．　；　トレース・ポインタの移動ライン数

ｌｏｏｐ　；　ループ回数（１バイト）

ｃｏｕｎｔ　；　カウント値（２バイト）

④　オプション部

＄以後の入力

○メモリ・コマンド（Ｅ；テスト）Ａ　；　簡易テスト

○ロード・コマンド　ｂｉａｓ　；　アドレス・バイアス値

○エミュレーション・コマンド（Ｔ；トレース）

Ｄ　；　表示ディセーブル（ニーモニックを出力しない）

Ｅ　；　表示イネーブル（ニーモニックを出力，ディフォールト値）

Ｒ　；　レジスタ表示有り

1．4．4　コマンド入力形式

（1）マッピング・コマンド

*MAP [{R / W / U / K} [partition]] ↵

R;エミュレーション・メモリをROMとしてマッピング
W;エミュレーション・メモリをRAMとしてマッピング
U;User側にマッピング
K;マッピング の解除

（2）メモリ・コマンド

*MEM C〔 addr 〕↵　　メモリ の変更 (Change)

*MEM D〔 partition〕↵　　メモリ の表示 (Dump)

*MEM E partition〔$A〕↵　　メモリ のテスト (Examination)
$A ； 簡易テスト

*MEM {F / G} partition string ↵　　F ； メモリ の初期化(Fill)
G ； メモリ のサーチ (Get)

*MEM {M / V / X} partition addr ↵　　M ； メモリ の転送 (Move)
V ； メモリ の比較(Verify)
X ； メモリ の交換 (eXchange)

（3）レジスタ・コマンド

*REG [{C / D} 〔 register-name 〕] ↵　　C ； レジスタの変更
D ； レジスタの表示

（4）モード・レジスタ・コマンド

*MDR [{C 〔 mode-register-name1〕 / D 〔 mode-register-name 〕}] ↵　　C;モード・レジスタの変更
D;モード・レジスタの表示

（5）特殊レジスタ・コマンド

*SPR [{C 〔 special-register-name1 〕 / D 〔 special-register-name2 〕}] ↵　　C;変更
D;表示

（6）ロード・コマンド

*LOD [{␣ / =} input-device-name] 〔 $bias〕 ↵

（7）セーブ・コマンド

*SAV 〔 output-device-name 〕 〔 { ␣ / = } partition 〕 ↵

（8）エミュレーション・コマンド

*RUN 〔 { N / B } 〔 addr 〕 〕 ↵　　N;ブレークなしリアルタイム実行 / B;ブレーク付きリアルタイム実行

*RUN S 〔 addr〕〔,step-No. 〕 ↵　　S;リアルタイム・ステップ実行

*RUN T 〔 addr〕〔, { step-No. / ※ } 〕〔$ { D / E } 〕〔$R〕 ↵

※=register-name { = / > / < / =< / >= / <> / >< / => / <= } value

T;1ステップ 実行（表示付き）
D;ニーモニックを出力しません
E;ニーモニックを出力します
R;レジスタ表示

（9）ブレーク・コマンド

*BRA 〔 A=addrx〕〔 V=values 〕〔 C=cond〕〔 L=loop〕 ↵　　アドレス,データ,コンディション,ループ の設定

*BRD 〔 data〕 ↵　　外部センス・データ の設定

*BRE 〔 count 〕 ↵　　オペコード・フェッチ回数の設定

*BRT 〔 time〕 ↵　　ブレーク・タイマの設定

*BR { 0 / 3 } 〔 BRA,BRD-----〕 ↵　　ロジカル・ブレーク・レジスタに フィジカル・ブレーク・レジスタを設定

*BRM 〔 BR0,BR1-----〕 ↵　　ブレーク・レジスタ の設定

(10) トレース・コマンド

*TRX 〔 A=addrx〕〔 V=values 〕〔 C=cond〕〔 PA=values 〕〔 PB=values 〕 ↵　　アドレス,データ,コンディション,ポートの設定

*TRC 〔 { M / I } 〕 ↵　　表示のトレース・サイクルの指定 (M;マシン・サイクル I;インストラクション)

*TRD 〔 { ALL / line-No. / -line-No. } 〕 ↵　　トレース・データの表示 (ALL; すべてのダンプ)

*TRP [{0 / N / line-No. / -line-No.}] ↵　　トレース・ポインタ の変更
(O;OLD, N;NEW)

*TRM [{NON / ALL / TRX}] ↵　　トレース・モードの指定
NON;トレースしません
ALL;すべてトレース
TRX;TRXコマンドの条件に従いブレークし，
　　指定したアドレス範囲をトレースします

*TRS [{PB / EX}] ↵　　トレースの選択
(PB;ポートB,EX;外部ポート)

(11) クロック・コマンド

*CLK [{I / U}] ↵　　CPUクロックの指定
I;内部クロック
U;外部クロック

(12) リセット・コマンド

*RES〔 H 〕↵　　H;IE-78C11全体のリセット

(13) 演算コマンド

*MAT　expression ↵

(14) 数値基数指定コマンド

*SUF [{H / T / Q / Y}] ↵　　H;16進
T;10進
Q; 8進
Y; 2進

(15) アセンブル・コマンド

*ASM〔 addr〕↵

(16) 逆アセンブル・コマンド

*DAS〔 partition 〕↵

(17) メモリ転送コマンド

*MOV {U / I} partition addr ↵　　U;エミュレーション・メモリからユーザ・システム上のメモリへのデータ転送
I;ユーザ・システム上のメモリからエミュレーション・メモリへのデータ転送

(18) 自己診断コマンド

*DIG↵

(19) ＰＲＯＭプログラマ制御コマンド

*PGM↵

保守／廃止

１．５　ＩＥ－７８Ｃ１１使用方法

１．５．１　概　要

ＩＥ－７８Ｃ１１システムでは，ＩＥコントロール・ボードが外部システムとのインタフェースをすべて受けもちます．外部インタフェースはMD－０８６ホスト・システムとのインタフェース用としてＩＥＥＥ－７９６バス，標準外部インタフェースとしてＲＳ－２３２－Ｃ（ＣＨ１）を持っています．

またそれ以外にサブチャネルとしてもう一つのＲＳ－２３２－Ｃ（ＣＨ２）を持っています．

ＩＥ－７８Ｃ１１はこのうちホスト・システムの拡張スロットのバスに挿入して使用する方法とＣＨ１に端末を接続して使用するスタンドアロンとしての使用方法の二つの使用方法が可能です．またそのときのモニタは共通に利用でき，二つのモードは後述するＤＩＰ　ＳＷで選択します．

この取扱説明書はこのうちのスタンドアロンでの使用方法について詳しく述べ，ホスト・システムに接続して使用する方法については別に添付していますシステム・ソフトウェアの取扱説明書を参考にしてください．

１．５．２　スタンドアロンとしての使用方法

ＩＥコントロール・ボードのジャンパ，ＤＩＰ　ＳＷ等のシリアル・インタフェース初期設定は，ボーレート４８００ボー，ハンドシェイクはハードウェア・コントロール，モードはスタンドアロンでの使用モードとなっています．このシリアル・インタフェースの詳細に関しては第４章を参照してください．

スタンドアロン使用の場合の電源付ケースとしては，ＳＢ－９００７があります．

１．５．３　MD－０８０，０８６シリーズと接続して使用する方法

MD－０８０，０８６システムとＩＥ－７８Ｃ１１システムとを接続しＩＥシステム・ソフトを使用すると，非常に強力なμＰＤ７８Ｃ１０，７８Ｃ１１，７８Ｃ１４のディバッガとして利用できます．さらに，MD－０８０，０８６システム上で作成したμＰＤ７８Ｃ１０，７８Ｃ１１，７８Ｃ１４用オブジェクトを直接ＩＥ－７８Ｃ

保守／廃止

１１に転送したり，逆にＩＥ－７８Ｃ１１上のプログラムをオブジェクト・ファイルに登録可能となります．

さらにＩＥシステム・ソフトの特徴を以下に列記します．

① シンボリック・ディバグが可能

② オブジェクト・ファイルをディバッガで直接使用可能

③ マルチタスク機能を利用して複数のディバッガ・システムを一つのコンソールで操作可能（ＭＤ－０８６シリーズと接続時のみ）

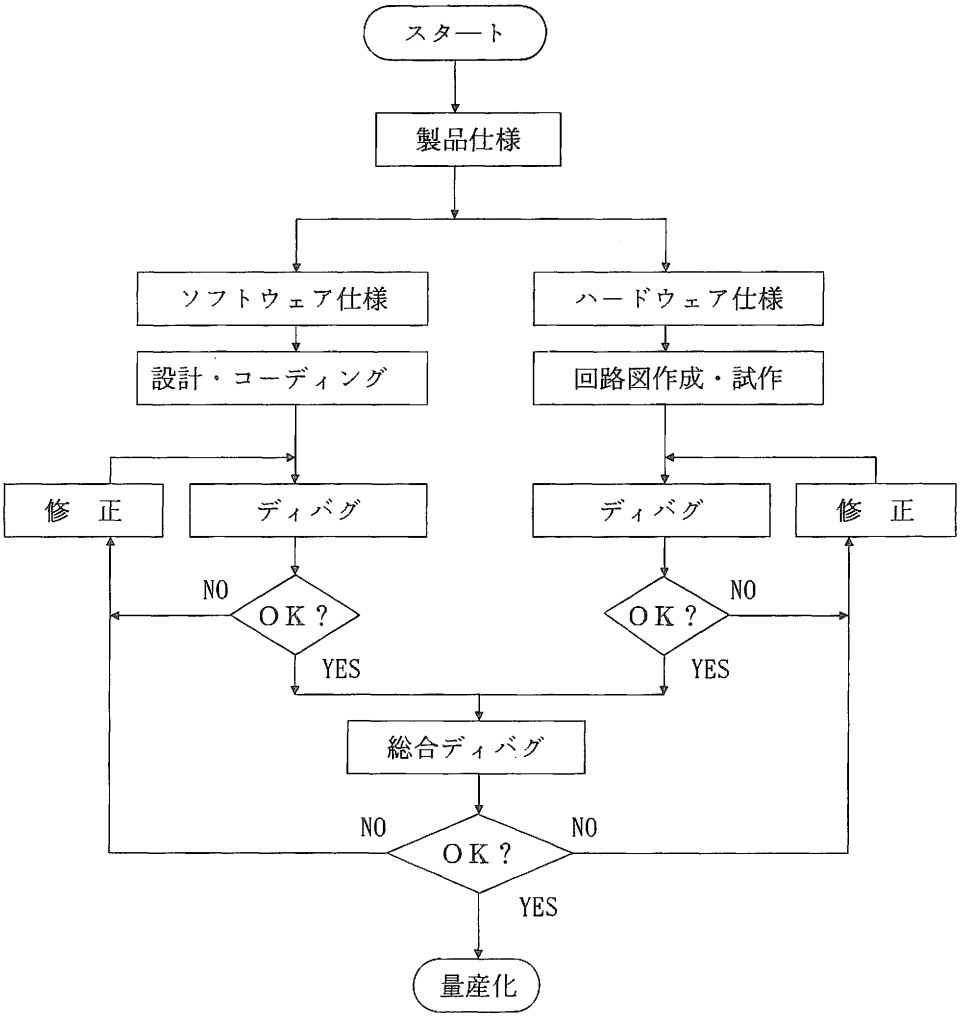
④ ソース・プログラムの作成からアセンブル，ディバグまでの一連の作業が一貫してできます．

保守／廃止

１．５．４　開発手順

以下に一般的システム開発の概略フローを示します．

図１－５　開発フロー



保守／廃止

注意　ＩＥ－７８Ｃ１１は，ハードウェアに依存しないソフトウェアだけのディバグも可能です．このときはターゲット・ハードウェアが存在しないので，使用法としてはエミュレーション・プローブの先をドライバ上にある自己診断用コネクタに挿入してください．ＭＯＤＥ０，ＭＯＤＥ１は１，１になります．

その時の設定は，以下のようにしてください．

① メモリは，使用部分のみ内部マッピングで，他はノン・マッピングとする．

② クロックは１２ＭＨｚで固定になります．

１．５．５　ラッチ・アップについて

ＣＭＯＳ　ＩＣは，入出力ピンに電源電圧より高い電圧が加わると，容易にラッチ・アップ現象を起こし，内部ＦＥＴがショート状態となり大電流が流れます．そのまま放置しておくとやがて致命的な破壊へと至ります．

このためＩＥ－７８Ｃ１１では，ラッチ・アップに対する警告回路が搭載されています．

（１）ラッチ・アップ警告回路の働き

ラッチ・アップが起きると，電源電流が増加するためそれを検出し，いち早くＣＭＯＳ　ＩＣの電源をしゃ断します．

また，同時にドライバ上面パネル上のＬＥＤで警告をし，外部に対してコントロール信号を出力します．

（２）ラッチ・アップ警告の解除

ラッチ・アップを引き起こした要因を取り除いた後，ドライバ・パネル側面のラッチ・アップ・リセット・スイッチ（ＬＡＴ　ＲＳＴ）を押してください．ドライバ内のＣＭＯＳ　ＩＣに再び電源が供給され，ＬＥＤが消灯します．このとき，ドライバ内部のハードウェアは不定状態にあるので，ハードウェア・リセットをかけてからディバグを再開してください．

ラッチ・アップ・リセット・スイッチを押しても警告が解除されない場合は，ドライバ内部の回路が破壊されている可能性があります．

保守／廃止

（３）ターゲット・システムの電源制御について

ラッチ・アップが起こった場合，ドライバ内部のＣＭＯＳ　ＩＣの電源を切るだけでなく，ターゲット・システム電源も切る必要があります．

このため，ＩＥ－７８Ｃ１１では警告回路の動作と同期して，外部の電源をコントロールするコントロール出力を備えています．

ドライバ上面パネル上のＰＯＷＥＲ　ＳＵＰＰＬＹ　ＣＯＮＴＯＲＯＬ端子（ＴＴＬレベル出力，ＬＯＷアクティヴ）にターゲット・システムの電源のコントロール端子を接続してください．

第２章　設　置

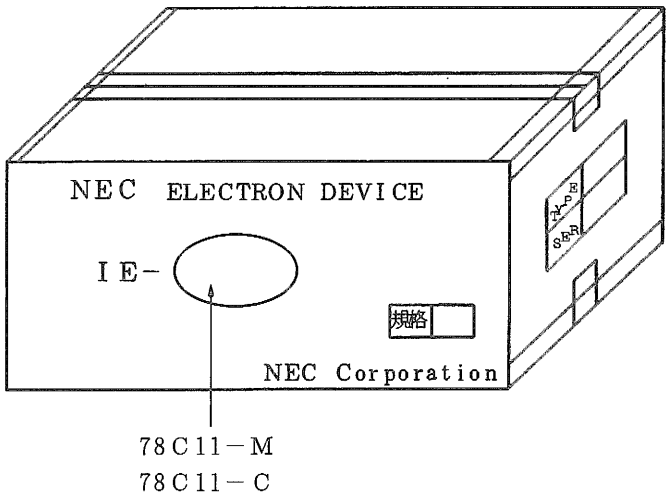
２．１　概　説

この章では，製品の開包からシステムを起動させ簡単なオペレーションを行なうまでの手順を説明します．

２．２　開　包

図２－１は，ＩＥ－７８Ｃ１１の梱包の図です．梱包箱を開けますと図２－２のようにドライバ・ユニット，付属品，コントロール・ボード，フロッピィ・ディスク，取扱説明書が入っています．

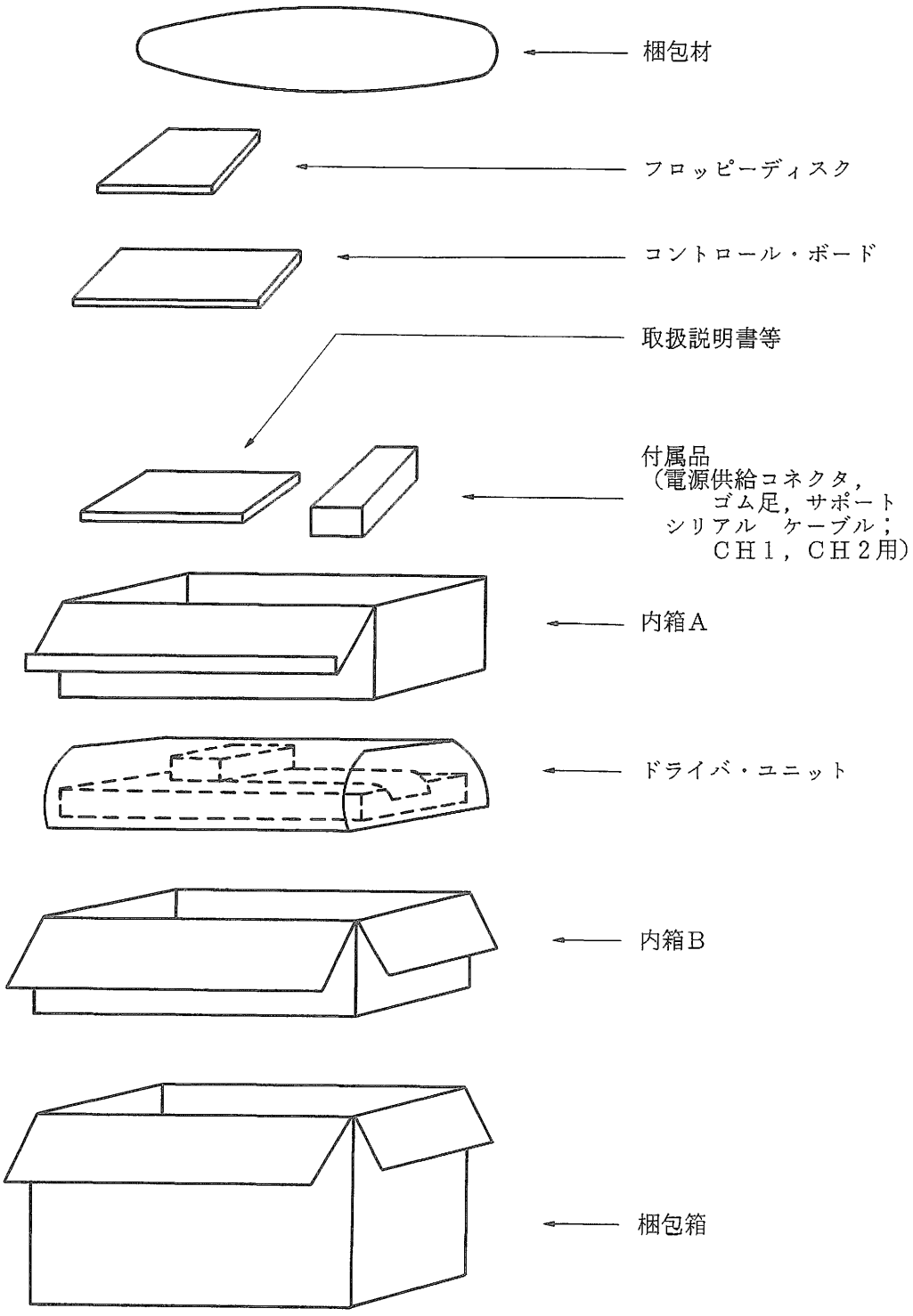
図２－１



注：梱包仕様は予告なく変更する場合があります．

保守／廃止

図２－２


梱包材
フロッピーディスク
コントロール・ボード
取扱説明書等
付属品
（電源供給コネクタ，
ゴム足，サポート
シリアル　ケーブル；
ＣＨ１，ＣＨ２用）
内箱Ａ
ドライバ・ユニット
内箱Ｂ
梱包箱

保守／廃止

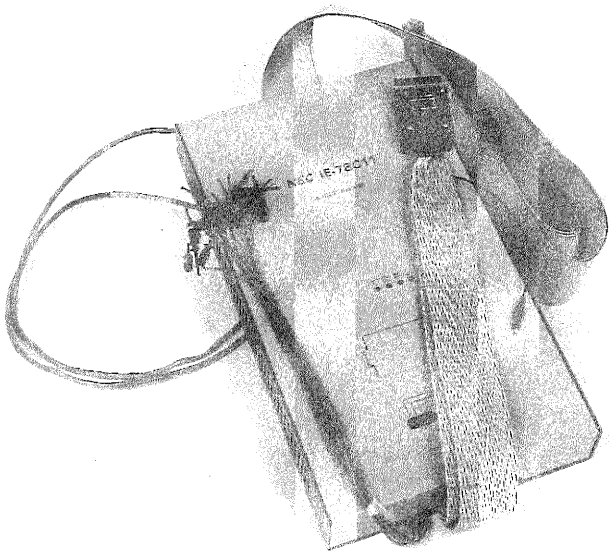
ＩＥ－７８Ｃ１１の梱包を解いた時に添付品リストを参考に部品をチェックし，すべてそろっているかを確認してください．部品は，小物が多く特にケーブル類は，細く断線しやすいので取扱いには十分注意してください．

部品のチェックが終わったならば，小物をなくさないように，保管には十分注意してください．

（1）ＩＥ－７８Ｃ１１ドライバ

○ドライバ・ユニット	1台	
○電源ケーブルＩ	1本	ドライバ・ユニット に付属されています。
○外部センス・クリップ	1本	
○システム・インタフェース・ケーブル	2本	
○エミュレーション・プローブ	1本	

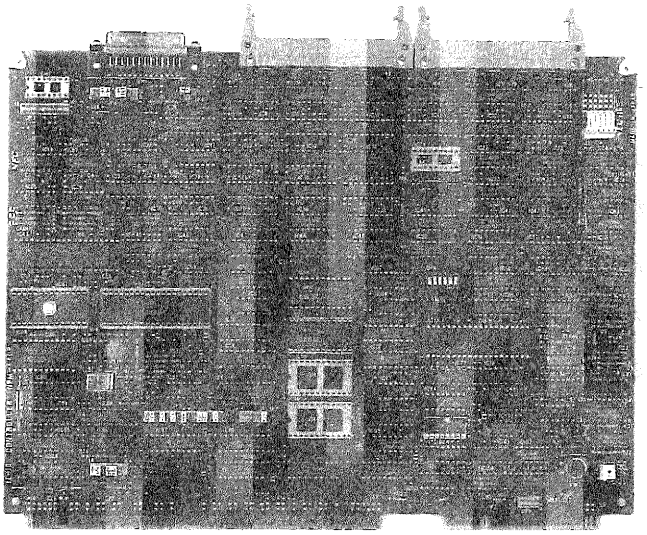
写真２



保守／廃止

（２）ＩＥ　コントロール・ボード　　１枚

写真３

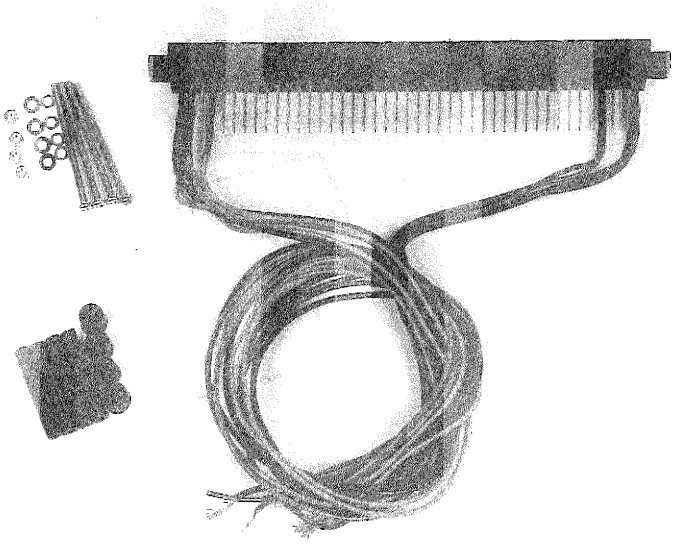


（３）付属品Ｉ

○電源ケーブルⅡ　　１本

○スペーサ（コントロール・ボードの足）　　４本

写真４



保守／廃止

（４）付属品Ⅱ

○シリアル・ケーブルⅠ（ＣＨ１用）　１本

○シリアル・ケーブルⅡ（ＣＨ２用）　１本

写真５

（５）その他

○フロッピィ・ディスク（８インチ）　１枚

○フロッピィ・ディスク保護用ファイル　１枚

○ＩＥ－７８Ｃ１１スタンドアロン取扱説明書　１部

○ＩＥ－７８Ｃ１１システム・ソフト取扱説明書　１部

○保証書　１部

○プログラム・プロダクト使用権設定契約書　１部

保守／廃止

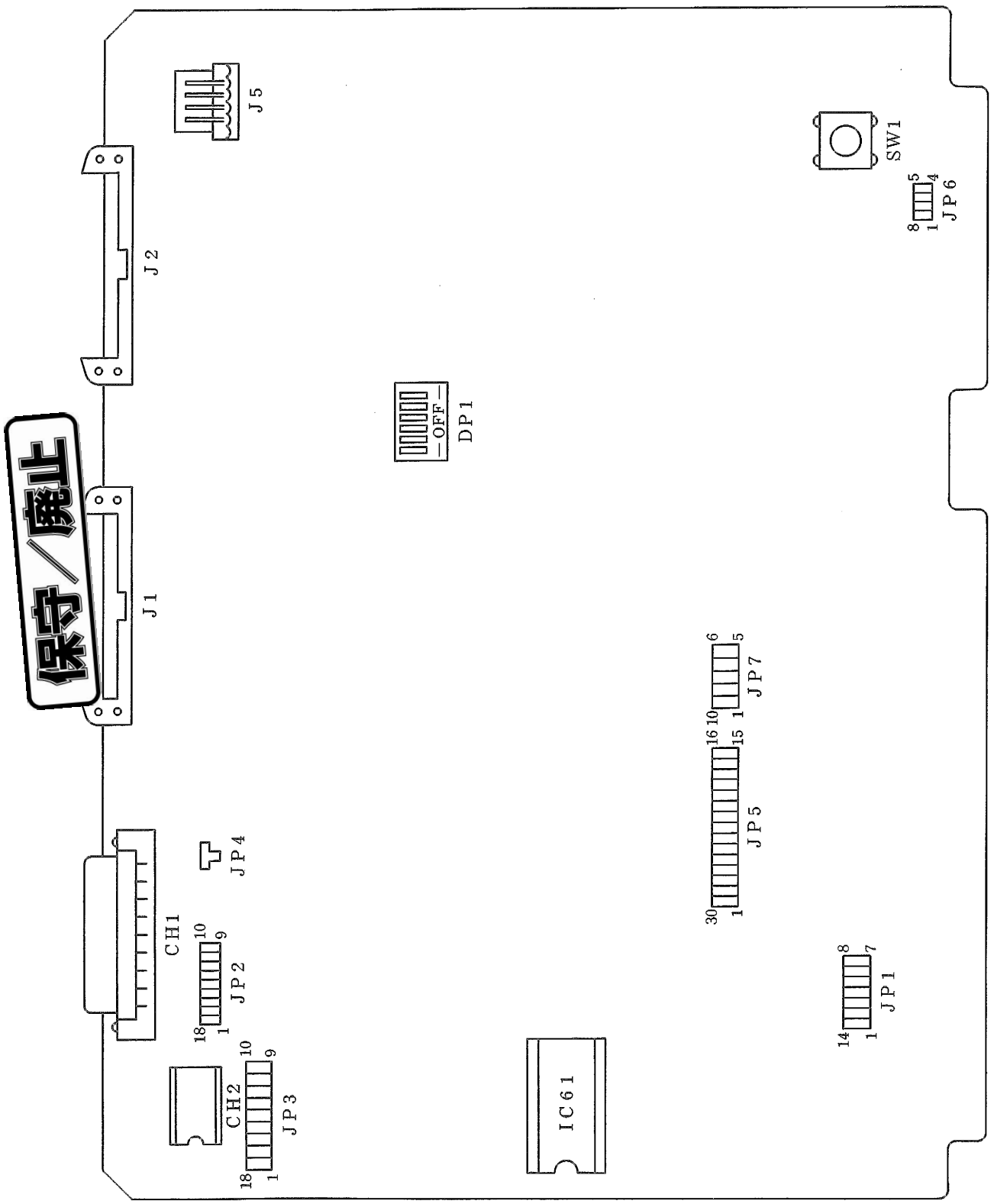
２．３　設置方法

下記の手順で設定してください．

２．３．１　ＩＥコントロール・ボードの設定

ＩＥコントロール・ボードのジャンパ，スイッチの設定，各ジャンパ，スイッチの位置は，図２－３の外観図がありますので参考にしてください．

図２－３

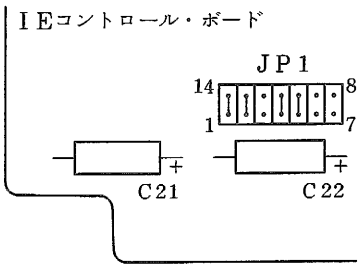

保守／禁止
J1
J2
J5
SW1
JP6
8
1
5
4
DP1
OFF
JP7
6
5
16
10
15
1
JP5
30
1
JP4
CH1
JP2
10
9
18
1
CH2
JP3
10
9
18
1
IC61
JP1
8
7
14
1

保守／廃止

（１）ＪＰ１～ＪＰ７までのジャンパが以下に示す図のとおりか，確認します．

① ＪＰ１

図２－４　スタンドアロン時のジャンパ接続図



位置	スタンドアロン時	MD-086シリーズに接続時
1-14	○	
2-13	○	
3-12		
4-11	○	○
5-10	○	
6-9		
7-8		

注(1)

○印は、ショートを意味します．

スタンドアロン時

１－１４，２－１３，４－１１，５－１０；ショート

他；オープン

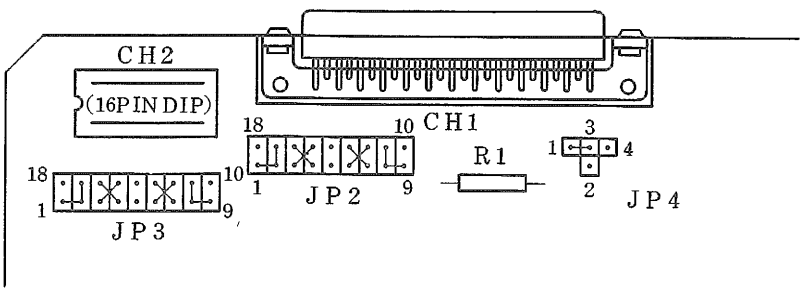
ホスト・マシン（ＭＤ－０８６シリーズ）接続時

４－１１；ショート　　他；オープン

注１）ＩＥＥＥ－７９６バス上のＰ１コネクタ１４ピンのターミネータ抵抗がある場合には，ＪＰ１の５－１０はオープンにしてください．

② ＪＰ２，ＪＰ３，ＪＰ４

図２－５



○ＪＰ２，ＪＰ３は同じ設定

１－２－１７，３－１５，４－１６，６－１２，７－１３，

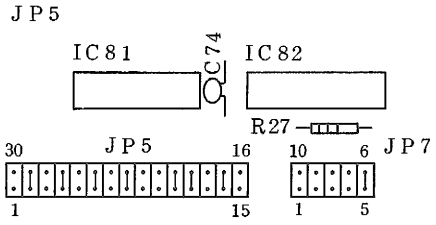
８－９－１１；ショート

５，１０，１４，１８；オープン

○ＪＰ４　　１－３のみショート

③　ＪＰ５

図２－６



位置	ショート
1-30	
2-29	○
3-28	
4-27	○
5-26	
6-25	○
7-24	
8-23	○
9-22	

位　置	ショート
10-21	
11-20	○
12-19	○
13-18	
14-17	○
15-16	

○印はショートを意味する．

２－２９，４－２７，６－２５，８－２３，１１－２０，１２－１９，

１４－１７；ショート

他；オープン

④　ＪＰ６

すべてオープン

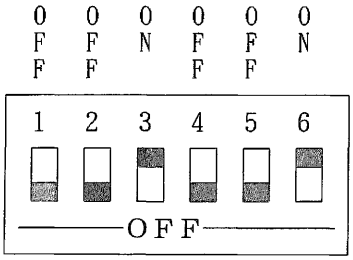
⑤　ＪＰ７

５－６；ショート

他；オープン

（2）ＤＰ１（ＤＩＰ　ＳＷ）を確認します．

図２－７



注）ＳＷ－Ｎｏ．１～３は，ボーレート
ＳＷ－Ｎｏ．４～６は，シリアル・プロトコル

初期設定

＊ボーレート　　４８００ボー

＊シリアル・プロトコル　　ハンドシェークはハードで行なっています．

ボーレートの指定については４．６，シリアル・プロトコルについては４．４を参照してください．

保守／廃止

（３）ジャンパ機能

＊印については，ジャンパの変更をしないでください。

ジャンパ No.	位　置	機　能
ＪＰ１	１－１４	$\overline{\text{ＢＣＬＫ}}$信号の設定 ＩＥ内部ではＩＥＥＥ　７９６バスの$\overline{\text{ＢＣＬＫ}}$信号を使用しています．ＩＥＥＥ－７９６バス上に他のボードから$\overline{\text{ＢＣＬＫ}}$信号が出力されている場合はオープンにし，出力されていない場合はショートにし，ＩＥコントロール・ボードからＩＥＥＥ－７９６バスへ$\overline{\text{ＢＣＬＫ}}$信号を出力します．
	２－１３	$\overline{\text{ＣＣＬＫ}}$信号の設定 １－１４端子の設定と同じ機能です．ショートにした場合は$\overline{\text{ＣＣＬＫ}}$信号をＩＥＥＥ－７９６バスへ出力します．
	＊ ３－１２	ＩＥのリセット信号の設定（$\overline{\text{ＩＮＩＴ}}$信号） ３－１２はオープン，４－１１はショートにしてください．
	＊ ４－１１	
	５－１０	ＩＥのリセット信号の設定 ＩＥＥＥ　７９６バス上の$\overline{\text{ＩＮＩＴ}}$信号にターミネート抵抗がある時はオープンにし，ない時はショートにしてください．
	７－８	$\overline{\text{ＢＰＲＯ}}$信号の設定 ショートにした場合はＩＥＥＥ－７９６バスへ$\overline{\text{ＢＰＲＯ}}$信号を出力し，オープンにした場合は出力しません．
ＪＰ２		ＣＨ．１のシリアル・インタフェース設定
ＪＰ３		ＣＨ．２のシリアル・インタフェース設定
ＪＰ４＊		ＩＥコントロール・ボード内のクロック・セレクト １－３のみショート
ＪＰ５		ＩＥ番号の設定
ＪＰ６＊		ＩＥコントロール・ボード内のハードウェア・インタフェースすべてオープンにしてください．
ＪＰ７＊		ＩＥとＭＤ－０８６とのインターフェース用です． ５－６をショートにしてください．

保守／廃止

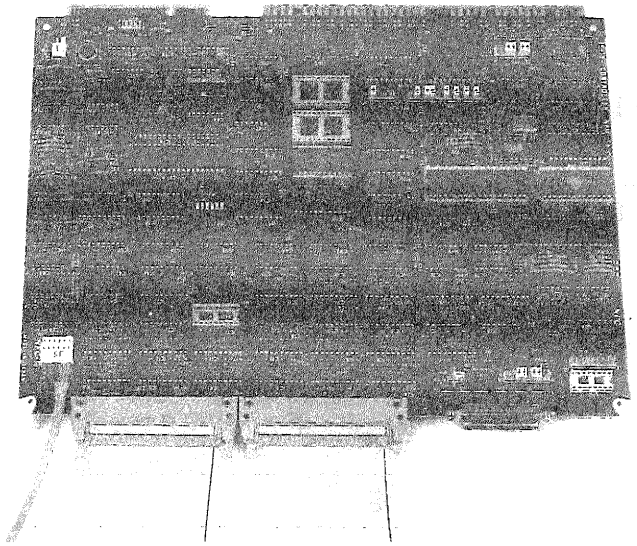
２．３．２　ＩＥコントロール・ボードとＩＥ－７８Ｃ１１ドライバとの接続

（1）ＩＥ－７８Ｃ１１ドライバから出ているＪ１，Ｊ２の二つの５０ピン・フラット・ケーブルの先のコネクタを，対応する‘Ｊ１’，‘Ｊ２’と名前のついたソケットに装着します．（写真６）

（2）同様にＩＥ－７８Ｃ１１ドライバより出ている電源ケーブルのコネクタを電源ソケットで装着します．（写真６）

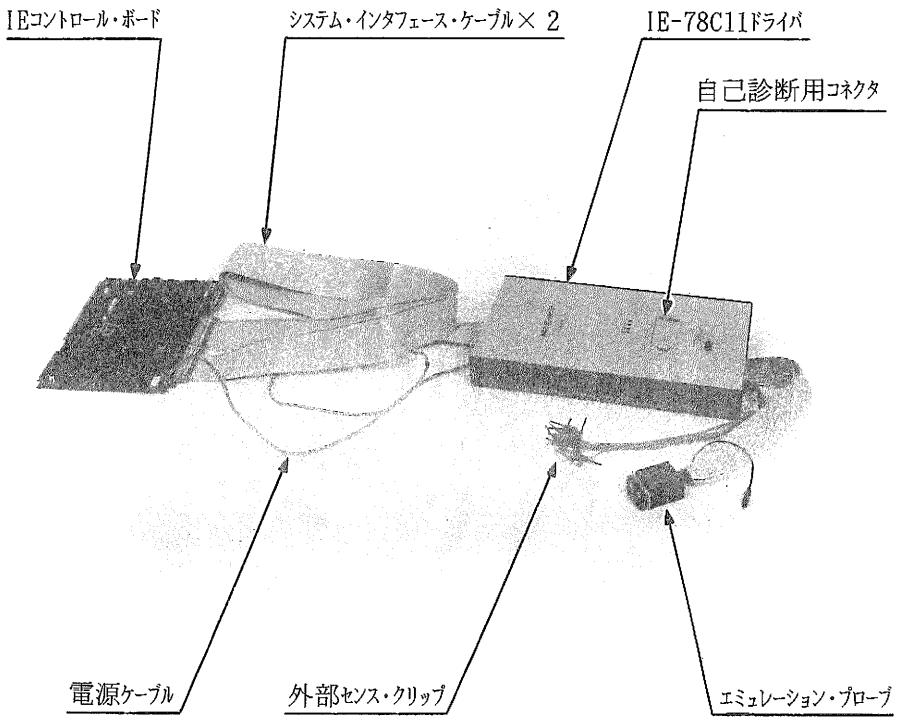
写真６

ＩＥ－７８Ｃ１１コントロール・ボードの外観



保守／廃止

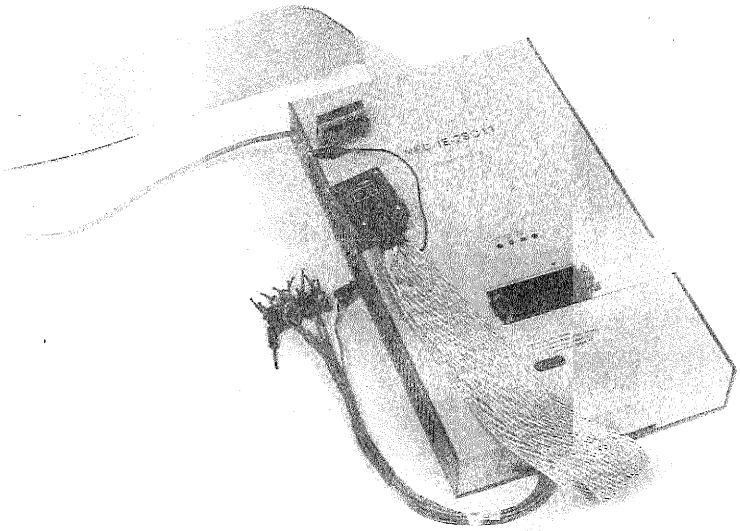
写真７　ＩＥ－７８Ｃ１１の外観



保守／廃止

２．３．３　ＩＥ－７８Ｃ１１ドライバのセッティング

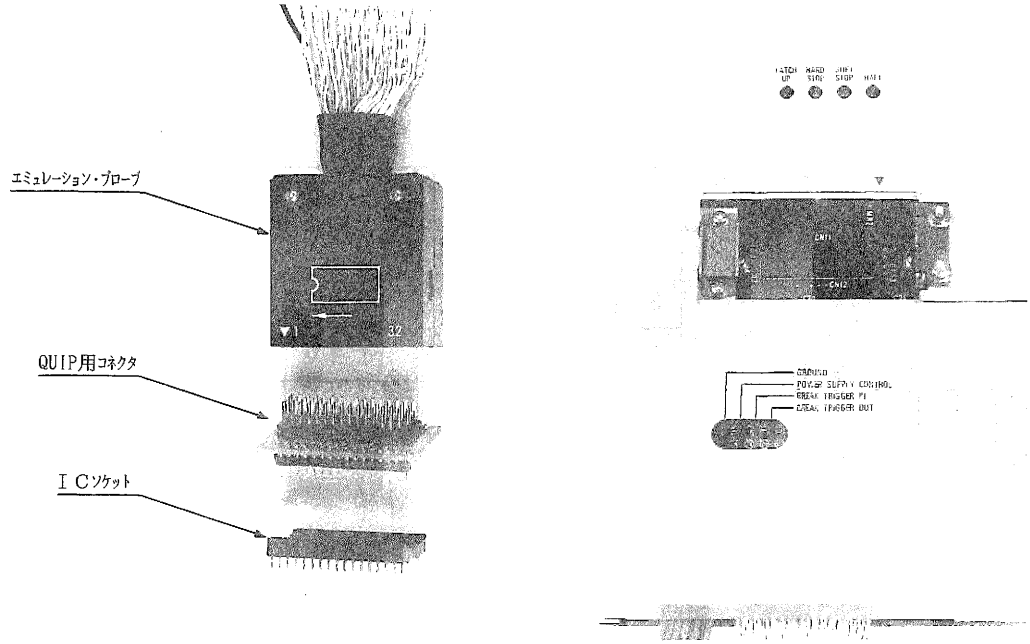
写真８　ＩＥ－７８Ｃ１１ドライバの外観図



保守／廃止

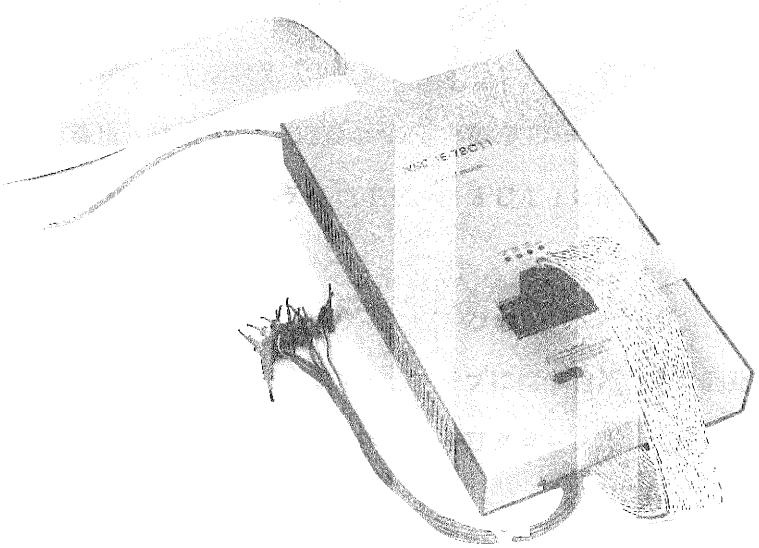
（1）エミュレーション・プローブのクロック切替スイッチはＣＬＫにします.

写真９　エミュレーション・プローブおよび自己診断用コネクタ



（2）次にＱＵＩＰ用コネクタを取りはずしたエミュレーション・プローブをＩＥ－７８Ｃ１１ドライバ上の自己診断用コネクタに挿入します.

写真１０

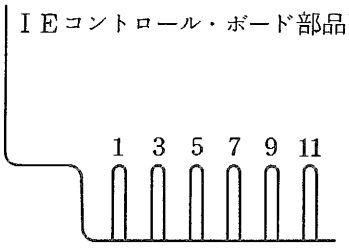


保守／廃止

２．３．４　ＩＥコントロール・ボードの電源端子の接続

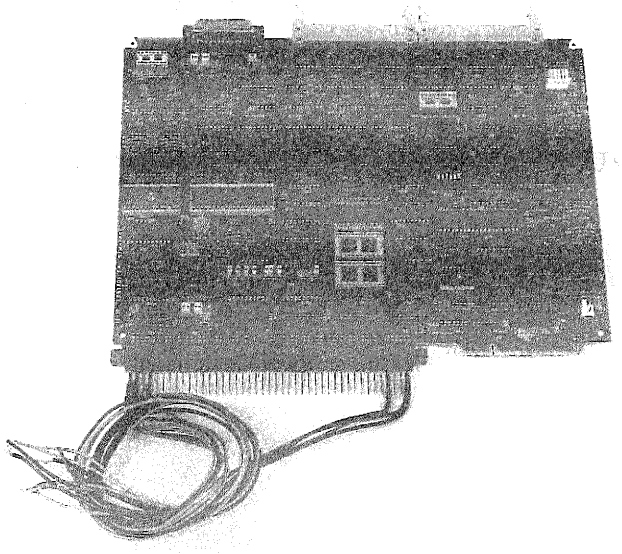
ＩＥＥＥ－７９６バスのＰ１ソケットより＋５Ｖ，＋１２Ｖ，－１２Ｖを供給してください．＋５Ｖは電流容量６．５Ａ以上，±１２Ｖは電流容量５００ｍＡ以上の電源を利用してください．（図２－８）

図２－８



ＰＩＮ	電源	ケーブル・カラー
１，２，８５，８６	ＧＮＤ	黒
３，４，５，６	＋５Ｖ	赤
７，８	＋１２Ｖ	橙
７９，８０	－１２Ｖ	青

写真１１



保守／廃止

エッジ・コネクタＰ１は，下記の製品を推奨します．

株式会社ケル社製　　１２５８－０８６－０３２

また，弊社では，５スロットのカード・ケージとしましてＳＢ－９０１７を，また±５Ｖ，±１２Ｖの電源を有するカード・ケージとしまして，ＳＢ－９００７を用意しております．

２．３．５　ＩＥコントロール・ボードとコンソールとの接続

ＩＥコントロール・ボードのＣＨ１（ＲＳ－２３２－Ｃ端子）とコンソールのＲＳ－２３２－Ｃ端子をシリアル・ケーブルで接続してください．なおその時に使用される端末もしくはＣＲＴの種類によりケーブルの作成およびＪ２の再設定が必要になりますので第４章を参考にしてください．（一般によく使用されている端末３品種については具体例を掲載しております．）

２．４　ターゲット回路との接続

２．４．１　接続および電源投入順序

ターゲット回路とＩＥ－７８Ｃ１１との接続は，以下に述べる手順で行なってください．

接続手順

① ＩＥ－７８Ｃ１１とターゲット・システムの電源をＯＦＦにする．

② エミュレーション・プローブをターゲット・システムの６４ピンＩＣソケットに挿入する．

③ ＩＥ－７８Ｃ１１の電源を投入する．

④ ターゲット・システムの電源を投入する．

⑤ 電源投入後必ずＲＥＳＨコマンドを実行してください．

電源の遮断はターゲット・システム，ＩＥ－７８Ｃ１１の順に行なってください．

２．４．２　エミュレーション・プローブのセッティング

ＩＥ－７８Ｃ１１のエミュレーション・プローブには水晶発振用の回路が内蔵されており，ターゲット・システムの発振回路をＣＰＵクロックとして使用する場合には

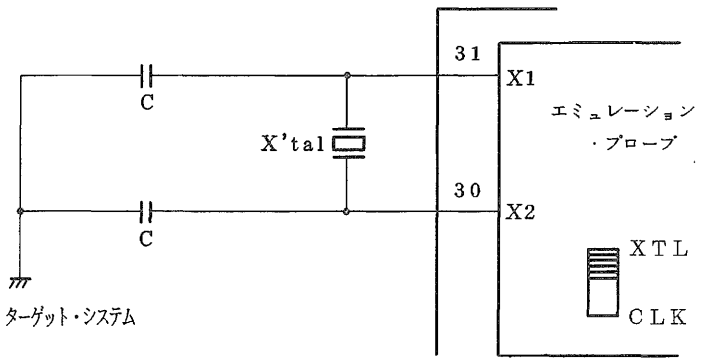
保守／廃止

発振回路に合わせてエミュレーション・プローブの切り換えスイッチを設定しなければなりません．（外部クロックを使用する時にはクロック・コマンド（３．１２参照）で外部クロックを指定（ＣＬＫ　Ｕ）してください．）

ここで，ターゲット・システムの発振回路を使用せずにＩＥ－７８Ｃ１１の内部クロック（１２ＭＨｚ）を使用する場合には，切り換えスイッチの設定を必要としておりません．（内部クロックを使用する時にはクロック・コマンドで内部クロックを指定してください．）

（１）ターゲット・システム上の発振回路が水晶発振子（Ｘ’ｔａｌ）またはセラミック発振子で構成されている場合はスイッチを‘ＸＴＬ’側にしてください．

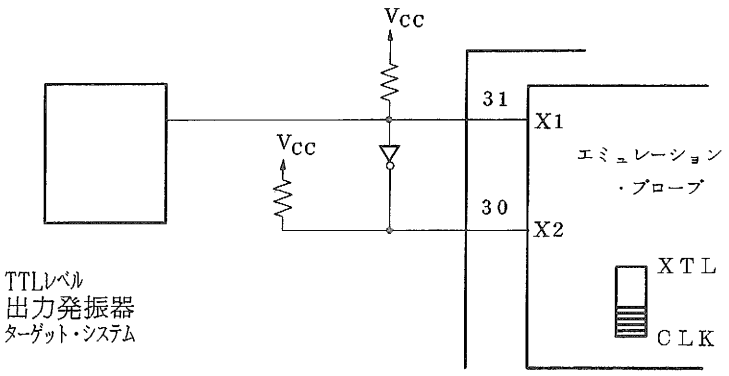
例１）



エミュレーション・プローブの切り換えスイッチを‘XTL’側にする．

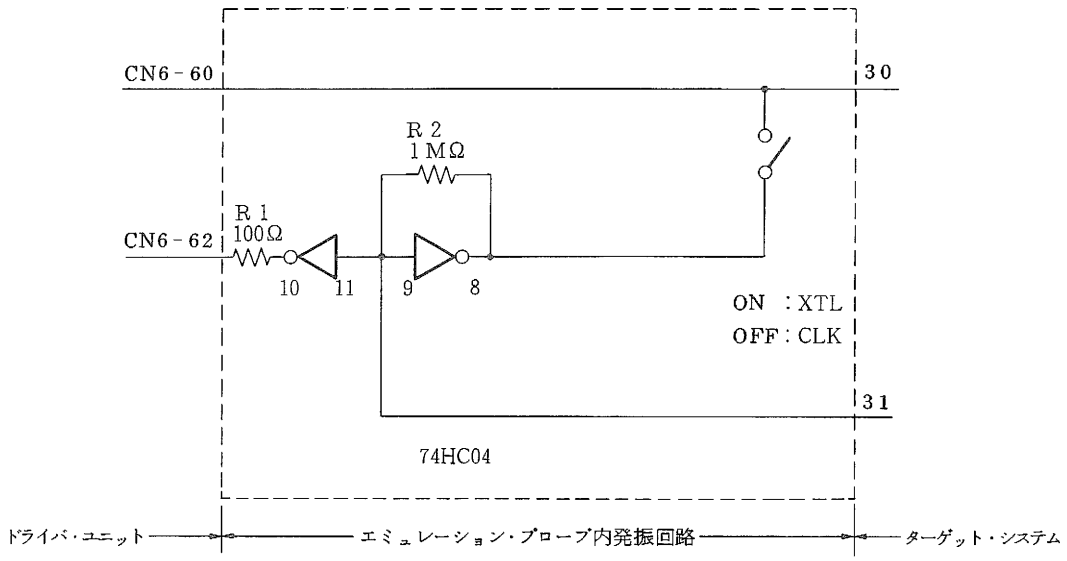
（2）ターゲット・システム上の発振回路がＴＴＬレベル出力の発振器で構成されている場合はスイッチを‘ＣＬＫ’側にしてください．

例2）



エミュレーション・プローブの切り換えスイッチを‘INT’側にする．

（3）エミュレーション・プローブ内部回路



保守／廃止

第３章　モニタ・コマンド詳細

３．１　初期設定

ＩＥ－７８Ｃ１１モニタのスタート時においては，数値入力基数は１６進数に，また，メモリ・マッピングはクリアされています．

エミュレーションＣＰＵのクロックは，内部クロック（１２ＭＨｚ）になっています．したがってスタート時には，メモリをアクセスするコマンドＭＥＭ（メモリ），ＲＵＮ（エミュレーション），ＡＳＭ（オンライン・アセンブル），ＤＡＳ（逆アセンブル），ＬＯＤ（ロード），ＳＡＶ（セーブ），ＭＯＶ（メモリ転送）の実行はできません．

基数には１６進数，１０進数，８進数，２進数があり，ＳＵＦ（サフィックス）コマンドにより２～１６進数に指定でき，設定後は基数にそった入力判断を行ないます．（ただし，コマンドによっては，数値入力が１６進入力と指定されていますので注意してください．）

モニタ・コマンドを実行する前にはマッピングの指定を必ず行なってください．

エミュレーション中にＣＴＲＬ－Ｃで‘＊’に抜けた時には，ドライバを起動する（ＭＥＭ，ＲＵＮ，ＬＯＤ，ＳＡＶ，ＡＳＭ，ＤＡＳ，ＭＯＶ，ＴＲ？，ＲＥＧ，ＣＬＫ，ＭＡＰ，ＢＲＡ～Ｃ，ＢＲＭ，ＰＧＭ）コマンドは実行できません．（実行するためにはＥＳＣキーかＲＥＳコマンドを実行します．）

また，ターゲット・システムを接続しないでソフト・ディバグする時には，エミュレーション・プローブを自己診断用ソケットに差してディバグしてください．

３．１．１　数値入力の説明

数値入力法は，いくつかの数字列と場合により最後に基数を付けて入力することにより実現します．

（ここで基数とは，その入力された文字列が１６進か，１０進か，８進か，２進かを指定するもので，１６進がＨ，１０進がＴ，８進がＱ，２進がＹに対応します．数値入力時に基数を入力しませんとＳＵＦコマンドによって指定された基数がとられます．）

次に基数別による記述例を示しますので参考にしてください．

例）

```
０ＦＦＦＦＨ═══ＦＦＦＦ  ┐
０１０１Ｈ═══１０１Ｈ    ┘ サフィックス１６進の場合

０６３２Ｑ═══６３２      ┐
０７１５２Ｑ═══７１５２Ｑ ┘ サフィックス　８進の場合

０１９５Ｔ═══１９５      ┐
０９９９９Ｔ═══９９９９Ｔ ┘ サフィックス１０進の場合

０１１０１Ｙ═══１１０１  ┐
００１１００Ｙ═══１１００Ｙ┘ サフィックス　２進の場合
```

エラー入力

```
Ｑ ┐
Ｈ │ データがなく，直接基数指定が
Ｔ │ きてしまった時
Ｙ ┘
```

また，有効桁数以上の入力は，有効桁数より上位桁の入力は無視されます．

（ただし，無視される上位桁の中に（０～９，Ａ～Ｆ）以外の文字があるとエラーになります．）

例）

```
１２Ａ４Ｆ６Ｈ═══Ａ４Ｆ６Ｈ        16bit 入力の場合
```

保守／廃止

１Ｆ５９ＢＣＨ═══ＢＣＨ　　8bit 入力の場合

２５２Ｔ═══２５２Ｔ　　8bit 入力の場合

１１０１１０１０Ｙ═══１１０１１０１０Ｙ　　8bit 入力の場合

１Ｇ６６Ｑ────→エラー　　8/16bit 入力時共にｴﾗｰ

また，サフィックス指定と異なった入力をした場合は，エラーとなります．

例）

サフィックスー８進

２７１８

Ｅ１９５Ｑ　　～部が８進数より大きい

サフィックスー１０進

Ｆ９１０

ＡＢ７ＣＴ　　～部が１０進数より大きい

サフィックスー２進

１２３４

０１２１Ｙ　　～部が２進数より大きい

有効桁数と最大値を次に示します．

入力 / ｻﾌｨｯｸｽ	8 bit	10 bit	16 bit
Ｈ	2桁 / ＦＦ	3桁 / ３ＦＦ	4桁 / ＦＦＦＦ
Ｔ	3桁 / ２５５	4桁 / １０２３	5桁 / ６５５３５
Ｑ	3桁 / ３７７	4桁 / １７７７	6桁 / １７７７７７
Ｙ	8桁 / 11111111	１０桁 / 1111111111	１６桁 / 1111111111111111

保守／廃止

３．１．２　オペランド入力形式

（１）ａｄｄｒ

アドレス値を数値形式で入力させる部分で数値は１６ｂｉｔ長です．

（２）ａｄｄｒｘ

アドレス値による数値範囲の指定で，通常のスタート／エンド・アドレスの記述を‘Ｘ’表現により置換えることができます．

例）
ＭＡＰ　Ｗ　０，ＦＦ↵　　ＭＡＰ　Ｗ　ＸＸ↵

この場合の‘ＸＸ’表現は，１６進数で０～Ｆまでが‘Ｘ’にあたります．サフィックスの変更により８進数の場合，０～７までが‘Ｘ’にあたり，２進数の場合は，０～１までが‘Ｘ’にあたります．

１０進数での‘Ｘ’入力は使用できませんので注意してください．

‘Ｘ’入力は，下位桁から連続して入力されていることが必要です．‘Ｘ’表現の間や後にデータが入り込んだ場合や上位桁のみの‘Ｘ’表現は使用できません．

例）
ＸＸ１ＸＹ（２進）――――――→エラー

ＸＸＸＱ（８進）――――――→エラー

ＸＸ１２３４Ｈ（１６進）――→エラー

ＸＸ１２３４Ｈの場合，有効桁の中に‘Ｘ’がなくてもエラーとなります．

ａｄｄｒｘの定義としては，‘Ｘ’入力だけを許し，他の入力形式は許しません．以降にａｄｄｒｘが出てきた場合は上の条件で入力を行なってください．

（３）ｐａｒｔｉｔｉｏｎ

ｐａｒｔｉｔｉｏｎの定義は，‘ａｄｄｒ，ａｄｄｒ’（スタート・アドレス，エンド・アドレス）またはａｄｄｒｘ（２）参照による数値範囲を示し，単独のａｄｄｒ表現は許しません．以降にｐａｒｔｉｔｉｏｎが出てきた場合は上の条件で行なってください．

保守／廃止

（4）ｐａｒｔｉｔｉｏｎｓ

ｐａｒｔｉｔｉｏｎｓの定義は，

〔ｐａｒｔｉｔｉｏｎ␣ｐａｒｔｉｔｉｏｎ␣……〕のように

ｐａｒｔｉｔｉｏｎをスペースで区切り複数個入力することを意味し、単独のａｄｄｒ表現は許しません．以降にｐａｒｔｉｔｉｏｎｓが出てきた場合は上の条件で行なってください．

（5）ａｄｄｒｓ

ａｄｄｒｓは，ａｄｄｒ，ｐａｒｔｉｔｉｏｎ，ｐａｒｔｉｔｉｏｎｓのすべての表現形式が可能です．

（6）ｓｔｒｉｎｇ

データ列を表わします．数値と数値を‘，’で区切って最大１０個まで入力可能です．ｓｔｒｉｎｇは１６進入力しか受付けません．（数値は８ｂｉｔ長）

（7）ｒｅｇｉｓｔｅｒ－ｎａｍｅ

ＩＥ－７８Ｃ１１の有するレジスタ・ネームです．（３．４参照）

レジスタは次に示すものがあります．

ＰＳＷ，Ｖ，Ａ，Ｂ，Ｃ，Ｄ，Ｅ，Ｈ，Ｌ，ＥＡ，ＳＰ，ＰＣ，Ｖ’，Ａ’，Ｂ’，Ｃ’，Ｄ’，Ｅ’，Ｈ’，Ｌ’，ＥＡ’

（8）ｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ１

ＩＥ－７８Ｃ１１の有するモード・レジスタ・ネームです．（３．５参照）

レジスタは次に示すものがあります．

ＭＡ，ＭＢ，ＭＣＣ，ＭＣ，ＭＭ，ＭＦ，ＴＭＭ，ＥＴＭＭ，ＥＯＭ，ＳＭＬ，ＳＭＨ，ＡＮＭ，ＺＣＭ

注　このレジスタに関しては，セットは可能ですが，すべてがＲＥＡＤ可能なわけではありません．

（9）ｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ２

（8）で示したモード・レジスタ・ネームの中でＲＥＡＤ可能なレジスタ・ネームです．（３．５参照）

レジスタは次に示すものがあります．

ＴＭＭ，ＥＯＭ，ＳＭＨ，ＡＮＭ

(10) ｓｐｅｃｉａｌ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ１

ＩＥ－７８Ｃ１１の有するＷＲＩＴＥ可能な特殊レジスタ・ネームです．（３．６参照）

レジスタは次に示すものがあります．

ＰＡ，ＰＢ，ＰＣ，ＰＤ，ＰＦ，ＭＫＨ，ＭＫＬ，ＴＸＢ，ＴＭ０，ＴＭ１，ＥＴＭ０，ＥＴＭ１

注　このレジスタに関しては，セットは可能ですが，すべてがＲＥＡＤ可能なわけではありません．

(11) ｓｐｅｃｉａｌ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ２

(10) で示した特殊レジスタ・ネームの中でＲＥＡＤ可能なレジスタ・ネームです．（３．６参照）

ＰＡ，ＰＢ，ＰＣ，ＰＤ，ＰＦ，ＭＫＨ，ＭＫＬ，ＲＸＢ，ＣＲ０，ＣＲ１，ＣＲ２，ＣＲ３，ＥＣＮＴ，ＥＣＰＴ

(12) ｉｎｐｕｔ／ｏｕｔｐｕｔ－ｄｅｖｉｃｅ－ｎａｍｅ

ＴＴＹ１　；　シリアル・インタフェースのＣＨ１に対応

ＴＴＹ２　；　シリアル・インタフェースのＣＨ２に対応

(13) ｓｔｅｐ－Ｎｏ．

ｓｔｅｐ－Ｎｏ．と表現されている部分は，３．１．１の数値形式で入力させる部分で数値は１６ｂｉｔ長です．

保守／廃止

(14) ｖａｌｕｅ

ｖａｌｕｅと表現されている部分は，３．１．１の数値形式で入力させる部分で数値は１６ｂｉｔ（または８ｂｉｔ）長です．

(15) ｅｘｐｒｅｓｓｉｏｎ

式は次の要素より構成されます．

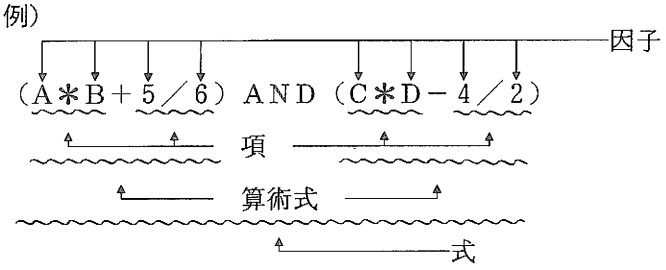
＜因子＞　；　ｖａｌｕｅ（それ自身で値を持ちます．）

＜項＞　　；　＜因子＞を乗除算したものです．

＜算術式＞；　＜項＞を加減算したものです．

＜式＞　　：　二つの＜算術式＞の比較演算

例）



使用可能な演算子として，＋，－，＊，／，ＡＮＤ，ＯＲ，ＸＯＲ，ＮＯＴがあります．（詳細は３．１４参照）

（注）カッコを使用した演算が可能です．ただし，‘（’　‘）’の組合わせ（ネスティング）回数３２回までです。

(16) ｌｉｎｅ－Ｎｏ．

ｌｉｎｅ－Ｎｏ．と表現されている部分は，３．１．１の数値形式で入力させる部分で数値は０～３ＦＦＨです．

(17) ｖａｌｕｅｓ

数値表現に関しては，(14) と同様ですが数字列にマスク条件が使用可能です．

例）

Ｖ＝１Ｘ３Ｑ　　{１０３, １１３, １２３, １３３, １４３, １５３, １６３, １７３}

通常のｖａｌｕｅも使用可能です．

（注）１０進数の使用不可

(18) ｃｏｎｄ

ブレーク，トレースのコンディションを指定します．（ＯＰ，ＲＤ，ＷＲ，ＭＲ，ＮＣ）

(19) ｌｏｏｐ

アドレス部，データ部，コンディション部によって設定されたブレーク条件を何回満足したらブレークするかの回数を設定します．数値は８ｂｉｔ長です．

３．１．３　コマンド入力時のターミネータ

ＩＥ－７８Ｃ１１のコマンド入力時のターミネータとして使用する‘␣’（ブランク）入力は，１個以上ならいくつでもかまいません．

例）ブランク入力例とエラー

```
*MEM␣D 0,FF ↵
     ↑―――――――ブランクが2 個                    ; OK
*␣MEM D ↵
  ↑―――――――プロンプト・マーク の次にくるブランク    ; OK
*M␣EM C ↵
   ↑―――――――コマンドを分割するブランク            ; ERROR
*MEM D 0,␣FF ↵
          ↑―――――カンマ の次にくるブランク         ; ERROR
```

保守／廃止

３．２　ＭＡＰ（マッピング）コマンド

表現形式：

＊ＭＡＰ [␣{Ｒ／Ｗ／Ｕ／Ｋ} [␣ｐａｒｔｉｔｉｏｎ]] ⤶

機　能：　メモリのマッピング状態の指定，表示，解除を行ないます。

メイン・コマンド	：	ＭＡＰ	マッピング・コマンド
サブ・コマンド	：	Ｒ	エミュレーションメモリを ＲＯＭとしてマッピング
	：	Ｗ	エミュレーションメモリを ＲＡＭとしてマッピング
	：	Ｕ	ＵＳＥＲ側にマッピング
	：	Ｋ	マッピングの解除
オペランド	：	ｐａｒｔｉｔｉｏｎ	‘Ｘ’入力可能な数値入力

上記の表現形式でｐａｒｔｉｔｉｏｎの入力には次の意味があります。

ｐａｒｔｉｔｉｏｎ入力なし　；　マッピング情報の表示
（Ｋの時はマッピングをすべて解除）

ｐａｒｔｉｔｉｏｎ入力あり　；　マッピング情報のセット
（Ｋの時はｐａｒｔｉｔｉｏｎ内を解除）

３．２．１　マッピング・コマンドの概要

マッピングはディバッガをＰＯＷＥＲ　ＯＮ時，ＲＥＳＥＴ　ＳＷを押した時，またはコマンド‘ＲＥＳ　Ｈ⤶’を実行した時にクリア状態，つまりメモリが接続されていない状態になっています．モニタのスタート後は，マッピングを指定しなければメモリをアクセスするコマンドは使用できません．メモリは２５６バイト単位でマッピングできます．

ＩＥ－７８Ｃ１１は最大６４Ｋバイトのメモリをアクセスできます．

このメモリ空間は，２５６バイトごとにユーザのターゲット・システムのメモリやエミュレーション・メモリにマッピングすることができます．

このメモリには，ユーザ・ターゲット・システムのプログラムをロードすることができます．

マッピングの種類は次の３種類があります．

① エミュレーション・メモリにマッピング，またライト・プロテクトをかけることができる．

② ユーザ・ターゲット・システムにマッピング

③ マッピングしない．

この３種類のマッピングの設定には，コマンドによるもの以外に起動時のＣＰＵのモードおよびＲＡＥの設定による自動設定があります．

ＲＡＥの設定はＥＮＡＢＬＥにセットすると，０ＦＦ００Ｈ～０ＦＦＦＦＨが自動的に内部ＲＡＭにマッピングされマッピング・コマンドでは変更できなくなります．ＤＩＳＡＢＬＥにセットすると，そのエリアはマッピング・コマンドによる設定の対象になります．

また起動時のＣＰＵのモード設定による内部ＲＯＭマッピングも同様に自動的に行なわれ，この部分も変更できなくなります．初期設定の詳細はＩＥ－７８Ｃ１１システム・ソフト取扱説明書に掲載されていますので参照してください．

３．２．２　マッピング・コマンド

マッピング・コマンドには次の三つのタイプがあります．

① 指定領域のマッピングの設定を行ないます．

② 現在のマッピング情報の表示を行ないます．

③ 指定領域のマッピングの解除を行ないます．

マッピングは，２５６バイト単位で行ないます．

マッピング指定の可能な範囲は（１）～（４）に示した条件により異なり，範囲を越えた指定は“ＭＡＰＰＩＮＧ　ＥＲＲＯＲ”となり，入力されたコマンドは無効となります．

保守／廃止

（1）ターゲットＣＰＵがμＰＤ７８Ｃ１０でＲＡＥがＥＮＡＢＬＥの時

エミュレーションＲＡＭエリア	（Ｗ）	０００００－ＦＦＦＦＨ
エミュレーションＲＯＭエリア	（Ｒ）	０００００－ＦＥＦＦＨ
ユーザ・システム・エリア	（Ｕ）	０００００－ＦＥＦＦＨ
解除	（Ｋ）	０００００－ＦＥＦＦＨ

（2）ターゲットＣＰＵがμＰＤ７８Ｃ１０でＲＡＥがＤＩＳＡＢＬＥの時

エミュレーションＲＡＭエリア	（Ｗ）	０００００－ＦＦＦＦＨ
エミュレーションＲＯＭエリア	（Ｒ）	０００００－ＦＦＦＦＨ
ユーザ・システム・エリア	（Ｕ）	０００００－ＦＦＦＦＨ
解除	（Ｋ）	０００００－ＦＦＦＦＨ

（3）ターゲットＣＰＵがμＰＤ７８Ｃ１１でＲＡＥがＥＮＡＢＬＥの時

エミュレーションＲＡＭエリア	（Ｗ）	１０００Ｈ－ＦＦＦＦＨ
エミュレーションＲＯＭエリア	（Ｒ）	００００－ＦＥＦＦＨ
ユーザ・システム・エリア	（Ｕ）	１０００Ｈ－ＦＥＦＦＨ
解除	（Ｋ）	１０００Ｈ－ＦＥＦＦＨ

（4）ターゲットＣＰＵがμＰＤ７８Ｃ１１でＲＡＥがＤＩＳＡＢＬＥの時

エミュレーションＲＡＭエリア	（Ｗ）	１０００Ｈ－ＦＦＦＦＨ
エミュレーションＲＯＭエリア	（Ｒ）	００００－ＦＦＦＦＨ
ユーザ・システム・エリア	（Ｕ）	１０００Ｈ－ＦＦＦＦＨ
解除	（Ｋ）	１０００Ｈ－ＦＦＦＦＨ

（5）ターゲットＣＰＵがμＰＤ７８Ｃ１４でＲＡＥがＥＮＡＢＬＥの時

エミュレーションＲＡＭエリア	（Ｗ）	４０００Ｈ－ＦＦＦＦＨ
エミュレーションＲＯＭエリア	（Ｒ）	００００Ｈ－ＦＥＦＦＨ
ユーザ・システム・エリア	（Ｕ）	４０００Ｈ－ＦＥＦＦＨ
解除	（Ｋ）	４０００Ｈ－ＦＥＦＦＨ

（6）ターゲットＣＰＵがμＰＤ７８Ｃ１４でＲＡＥがＤＩＳＡＢＬＥの時

エミュレーションＲＡＭエリア	（Ｗ）	４０００Ｈ－ＦＦＦＦＨ
エミュレーションＲＯＭエリア	（Ｒ）	００００Ｈ－ＦＦＦＦＨ
ユーザ・システム・エリア	（Ｕ）	４０００Ｈ－ＦＦＦＦＨ
解除	（Ｋ）	４０００Ｈ－ＦＦＦＦＨ

保守／廃止

３．２．３　マッピング指定

MAP␣ {W / R / U / K} ␣ｐａｒｔｉｔｉｏｎ↵

入力例）

```
＊MAP　W　１０００Ｈ，１３ＦＦＨ↵　　①
＊MAP　R　８００Ｈ，ＦＦＦＨ↵　　②
＊MAP　U　０Ｈ，７ＦＦＨ↵　　③
＊MAP　K　１０００Ｈ，１３ＦＦＨ↵　　④
```

①プログラム・メモリの１０００Ｈ～１３ＦＦＨ番地のアドレス空間をエミュレーションＲＡＭに指定します．

②プログラム・メモリの８００ＨからＦＦＦＨ番地のアドレス空間をエミュレーションＲＯＭに指定します．エミュレーションＲＯＭに指定されたエリアは実行中の書込みは禁止されます．

③プログラム・メモリの０Ｈ～７ＦＦＨ番地のアドレス空間をユーザ・システム側に指定します．

④プログラム・メモリの１０００Ｈ～１３ＦＦＨ番地のアドレス空間のマッピング指定を解除します．

例）

```
*MAP W 1000H,13FFH ↵          ←―― 1000H番地から13FFH番地のエリアをエミュレーションRAMに指定

*MAP ↵
 $ IE-INTROM MAPPING $
 $ IE-INTRAM MAPPING $
 1000-13FF FF00-FFFF
 $ USER $
 $ NON MAPPING $
 0000-0FFF 1400-FEFF

*MAP R 800H,FFFH ↵            ←―― 800H番地からFFFH番地のエリアをエミュレーションROMに指定
```

```
*MAP ↵

 $ IE-INTROM MAPPING $
 0800-0FFF
 $ IE-INTRAM MAPPING $
 1000-13FF FF00-FFFF
 $ USER $
 $ NON MAPPING $
 0000-07FF 1400-FEFF

*MAP U 0,7FFH ↵          ←―― ユーザ・メモリの0H番地から7FFH番地の2Kを指定

*MAP ↵

 $ IE-INTROM MAPPING $
 0800-0FFF
 $ IE-INTRAM MAPPING $
 1000-13FF FF00-FFFF
 $ USER $
 0000-07FF
 $ NON MAPPING $
 1400-FEFF

*MAP K 1000H,13FFH ↵     ←―― 1000H番地から13FFH番地のエリアの マッピング指定を解除

*MAP ↵

 $ IE-INTROM MAPPING $
 0800-0FFF
 $ IE-INTRAM MAPPING $
 FF00-FFFF
 $ USER $
 0000-07FF
 $ NON MAPPING $
 1000-FEFF

*MAP W XXXX ↵            ←―― 0H番地からFFFFH番地のエリアをエミュレーションRAM に指定
      ～～～ 'X' 入力
*MAP

 $ IE-INTROM MAPPING $
 $ IE-INTRAM MAPPING $
 0000-FFFF
 $ USER $
 $ NON MAPPING $

*
```

保守／廃止

３．２．４　表　示

MAP [␣ {W / R / U}] ⏎

入力例）

```
*MAP  W⏎          ①
*MAP  R⏎          ②
*MAP  U⏎          ③
*MAP⏎             ④
```

①エミュレーションＲＡＭにマッピングされたエリアを表示します．

②エミュレーションＲＯＭにマッピングされたエリアを表示します．

③ユーザ・システム側にマッピングされたエリアを表示します．

④マッピングされたエリアをすべて表示します．

例）

```
*MAP W⏎          ←――――――――――――――エミュレーションRAMの マッピング状態の表示

 $ IE-INTRAM MAPPING $
 FF00-FFFF

*MAP R⏎          ←――――――――――――――エミュレーションROMの マッピング状態の表示

 $ IE-INTROM MAPPING $
 0800-0FFF

*MAP U⏎          ←―――――――――――――― ユーザ・メモリの マッピング状態の表示

 $ USER $
 0000-07FF

*MAP⏎            ←――――――――――――――すべての マッピング状態の表示

 $ IE-INTROM MAPPING $
 0800-0FFF
 $ IE-INTRAM MAPPING $
 FF00-FFFF
 $ USER $
 0000-07FF
 $ NON MAPPING $
 1000-FEFF
*
```

３．２．５　解　除

ＭＡＰ␣Ｋ↵

ＩＥ－７８Ｃ１１のスターティング時に設定されたマッピング状態にします.

（1）ＣＰＵがμＰＤ７８Ｃ１０でＲＡＥをＥＮＡＢＬＥに指定した場合

```
*MAP K↵

*MAP↵

 $ IE-INTROM MAPPING $
 $ IE-INTRAM MAPPING $
 FF00-FFFF
 $ USER $
 $ NON MAPPING $
 0000-FEFF
```

（2）ＣＰＵがμＰＤ７８Ｃ１０でＲＡＥをＤＩＳＡＢＬＥに指定した場合

```
*MAP K↵

*MAP↵

 $ IE-INTROM MAPPING $
 $ IE-INTRAM MAPPING $
 $ USER $
 $ NON MAPPING $
 0000-FFFF
```

（3）ＣＰＵがμＰＤ７８Ｃ１１でＲＡＥをＥＮＡＢＬＥに指定した場合

```
*MAP K↵

*MAP↵

 $ IE-INTROM MAPPING $
 0000-0FFF
 $ IE-INTRAM MAPPING $
 FF00-FFFF
 $ USER $
 $ NON MAPPING $
 1000-FEFF
```

（４）ＣＰＵがμＰＤ７８Ｃ１１でＲＡＥをＤＩＳＡＢＬＥに指定した場合

```
*MAP K)

*MAP )

 $ IE-INTROM MAPPING $
 0000-0FFF
 $ IE-INTRAM MAPPING $
 $ USER $
 $ NON MAPPING $
 1000-FFFF
```

（５）ＣＰＵがμＰＤ７８Ｃ１４でＲＡＥをＥＮＡＢＬＥに指定した場合

```
*MAP K)

*MAP )

 $ IE-INTROM MAPPING $
 0000-3FFF
 $ IE-INTRAM MAPPING $
 FF00-FFFF
 $ USER $
 $ NON MAPPING $
 4000-FEFF
```

（６）ＣＰＵがμＰＤ７８Ｃ１４でＲＡＥがＤＩＳＡＢＬＥに指定した場合

```
*MAP K)

*MAP )

 $ IE-INTROM MAPPING $
 0000-3FFF
 $ IE-INTRAM MAPPING $
 $ USER $
 $ NON MAPPING $
 4000-FFFF
```

保守／廃止

３．３　ＭＥＭ（メモリ）コマンド

表現形式：

＊ＭＥＭ␣Ｃ〔␣ａｄｄｒ〕↵

＊ＭＥＭ␣Ｄ〔␣ｐａｒｔｉｔｉｏｎ〕↵

＊ＭＥＭ␣Ｅ␣ｐａｒｔｉｔｉｏｎ〔＄Ａ〕↵

＊ＭＥＭ␣{Ｆ／Ｇ}␣ｐａｒｔｉｔｉｏｎ〔␣ｓｔｒｉｎｇ〕↵

＊ＭＥＭ␣{Ｍ／Ｖ／Ｘ}␣ｐａｒｔｉｔｉｏｎ␣ａｄｄｒ↵

機　能：　メモリ内容の変更，表示，テスト，初期化，サーチ，転送，比較，交換を行ないます．

メイン・コマンド	：	ＭＥＭ	メモリ・コマンド
サブ・コマンド	：	Ｃ	メモリ・チェンジ（Ｃｈａｎｇｅ）
（省略不可）	：	Ｄ	メモリ・ダンプ　（Ｄｕｍｐ）
	：	Ｅ	メモリ・テスト　（Ｅｘａｍｉｎａｔｉｏｎ）
	：	Ｆ	メモリ初期化　（Ｆｉｌｌ）
	：	Ｇ	メモリ・サーチ　（Ｇｅｔ）
	：	Ｍ	メモリ・ムーブ　（Ｍｏｖｅ）
	：	Ｖ	メモリ比較　（Ｖｅｒｉｆｙ）
	：	Ｘ	メモリ交換　（ｅＸｃｈａｎｇｅ）
オペランド	：	ａｄｄｒ	式入力可能な数値（２バイト）
	：	ｐａｒｔｉｔｉｏｎ	‘Ｘ’入力可能　領域入力
	：	ｓｔｒｉｎｇ	１バイトの16進データの連続入力（データは１０個までで，ターミネータは‘，’です．）

オプション	： Ａ	サブ・コマンド‘Ｅ’の時の簡易テスト・スイッチ

３．３．１　メモリ・コマンド

メモリ・コマンドを実行する時は，マッピングがエミュレーション・メモリ，もしくはユーザ・システムにセットされている事を確認してください．もしマッピングされていない時は，‘ＮＯＮ－ＭＡＰ　ＡＲＥＡ　ＡＣＣＥＳＳ’と出力し，‘＊’に戻りコマンドは実行されません．

保守／廃止

３．３．２　メモリの変更（Ｃｈａｎｇｅ）

ＭＥＭ␣Ｃ␣ａｄｄｒ↵

アドレス入力は，１６進入力時で０～ＦＦＦＦＨの下位４桁が有効です．コマンドが正しく入力されるとアドレスとそのメモリの内容を表示し‘－’を出力しデータの入力待ちとなります．（～線は，ＩＥ－７８Ｃ１１からの出力を示します．）

```
*MEM C 10 ↵                    ﾏｯﾋﾟﾝｸﾞ されていないと，左のようにｴﾗｰ となり
~MAPPING ERROR                 ‘*’に戻ります．
*
```

マッピングを指定し，もう一度コマンドを入力します．

```
*MAP W 0,FFH ↵
```

```
*MEM C 10T ↵                   10進入力
000A  00-34 00-88 00-55 00-  ↵
*
```

HEXﾃﾞｰﾀ　入力

メモリ内容を変更したら‘－’の後に‘↵’を入力すればコマンドは終了します．

```
*MEM C 10T ↵
000A  34-34 88-88 55-55 00-  ↵
*
```

ｽﾍﾟｰｽ入力

‘－’の後にスペース入力しますと，データの変更をせずにもう一度そのデータをエコーバックして，次のメモリ内容の表示をします．

```
*MEM C ↵
0000  00-12 00-FF 00-67 00-          ‘←’
0002  67-      <-------------------(ASCII 5FH)
0001  FF-88 67-  ↵
*
```

サブ・コマンドの‘Ｃ’の後のアドレスを省略すると０番地とみなされます．‘－’の後のＡＳＣＩＩ規格‘←’（ＡＳＣＩＩ　５ＦＨ）入力は一つ前のアドレスのメモリに戻ります．

```
*MEM C 120954100010H ↵
0010 FF-FF FF-FF FF-FF FF- ↵
*
```

アドレスの桁数はＨＥＸ入力時，下位４桁（１６進で０ＦＦＦＦＨ以内）が有効になります．

```
*MEM C 10001Y  ↵
0011  FF-FF FF-FX ←――――――――１６進以外の入力
INPUT DATA ERROR
*
*MEM C 5Q  ↵
0005  00-  ↵
*
```

アドレスの入力はサフィックス・コマンド，またはアドレスの最後につけた基数（Ｈ，Ｔ，Ｑ，Ｙ）で決められます．

メモリの内容は１６進コードで入力してください．１６進以外の入力があると‘ＩＮＰＵＴ　ＤＡＴＡ　ＥＲＲＯＲ’と出力して‘＊’に戻ります．

```
*MEM  C  13Q ↵
000B 88-A ←――――――――――2 文字目にｽﾍﾟｰｽ入力
INPUT DATA ERROR
*
*MEM C 5 ↵
0005 00-Z ←――――――――――１６進以外の入力
INPUT DATA ERROR
*
*MEM C10 ↵
      ↑
      └――――――――――ｽﾍﾟｰｽが入っていない
COMMAND FORMAT ERROR
*
```

３．３．３　メモリのダンプ（Ｄｕｍｐ）

MEM␣D␣ｐａｒｔｉｔｉｏｎ↵

ｐａｒｔｉｔｉｏｎで入力された、領域のメモリをダンプします。

```
*MEM D 0,FH ↵
0000   12 88 67 00 00 00 00 00  00 00 34 88 55 00 00 00
*
*MEM D XH ↵
0000   12 88 67 00 00 00 00 00  00 00 34 88 55 00 00 00
*
*MEM D XQ ↵
0000   12 88 67 00 00 00 00 00
*
*MEM D XY ↵
0000   12 88
*
```

上のように，ｐａｒｔｉｔｉｏｎは‘X’入力を用いたエリア入力ができます．

```
*MEM D 2F0H,31FH ↵
02F0   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF

0300   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
0310   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
*
```

ダンプ表示で，アドレスの下位の２桁がＦＦになると次の一行をあけて上のように表示します．

```
*MEM D FFH,FFH ↵
00FF   FF
*MEM D 3XH ↵
0030   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
*
*MEM D XXXH ↵
0000   12 28 67 00 00 00 00 00  00 00 34 88 55 00 00 00
0010   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
0020   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0030   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
0040   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0050   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF

*                                                  CTRL-Cまたは ESCキー入力
```

保守／廃止

プログラム・メモリのダンプ中にＣＴＲＬ－ＣまたはＥＳＣキーを入力すると，印字を中止して‘＊’に戻ります．

```
*MAP K↵
*MAP W 0,FFH↵
*MEM D XXXH ↵
MAPPING ERROR
*
```

マッピングされていないエリアを指定すると‘ＭＡＰＰＩＮＧ　ＥＲＲＯＲ’を出力して‘＊’に戻ります．

```
*MEM D 10,5 ↵
 COMMAND FORMAT ERROR
*
*MEM FFH,FFH↵
 COMMAND FORMAT ERROR
*
*MEM D0,FFH ↵
 COMMAND FORMAT ERROR
*
```

保守／廃止

３．３．４　メモリのテスト（Ｅｘａｍｉｎａｔｉｏｎ）

ＭＥＭ␣Ｅ␣ｐａｒｔｉｔｉｏｎ〔$Ａ〕

ｐａｒｔｉｔｉｏｎ入力された領域のメモリのテストを行ないます。メモリに異常がある時は，アドレスとメモリ内容，書込みデータが出力されます．なお，テスト後，ｐａｒｔｉｔｉｏｎで入力された領域のデータは保障しません．また，ｐａｒｔｉｔｉｏｎの後に‘$Ａ’を入力することによって簡易テスト・モードになります．

簡易テストでは，メモリの内容を破壊しません．６４Ｋのメモリ・テスト時間は，約７分２０秒，６４Ｋの簡易テストは約１分３０秒です．

保守／廃止

３．３．５　メモリの初期化（Ｆｉｌｌ）

ＭＥＭ␣Ｆ␣ｐａｒｔｉｔｉｏｎ␣ｓｔｒｉｎｇ↵

ｓｔｒｉｎｇで指定されるデータの個数は，‘，’で区切って１０個までです．

```
*MEM F 0,FFH 00 ↵

MAPPING ERROR
*
*MAP W 0,FFH ↵

*MEM F 0,FFH 00 ↵

*MEM D 0,63 ↵
0000   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0010   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0020   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0030   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0040   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0050   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0060   00 00 00 00
*
```

プログラム・メモリの，０～ＦＦＨを００Ｈでイニシャライズ，次に０～ＦＦＨを１，２，３，４，５，６，７，８，９，０のｓｔｒｉｎｇでイニシャライズします．

```
*MEM F 0,FFH 1,2,3,4,5,6,7,8,9,0↵

*MEM D XXH↵
0000   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
0010   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
0020   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
0030   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
0040   05 06 07 08 09 00 01 02  03 04 05 06 07 08 09 00
0050   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
0060   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
0070   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
0080   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
0090   05 06 07 08 09 00 01 02  03 04 05 06 07 08 09 00
00A0   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
00B0   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
00C0   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
00D0   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
00E0   05 06 07 08 09 00 01 02  03 04 05 06 07 08 09 00
00F0   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
*
```

３．３．６　メモリのサーチ（Ｇｅｔ）

MEM␣G␣ｐａｒｔｉｔｉｏｎ␣ｓｔｒｉｎｇ↵

ｐａｒｔｉｔｉｏｎエリア内にある，ｓｔｒｉｎｇと同じデータの並びの先頭のアドレスを表示します．

ｓｔｒｉｎｇで指定されるデータの個数は‘，’で区切って１０個までです．

```
*MEM F 0H,110H 00 ↵

*MEM F F0,110H 55,AA,33,88↵

*MEM D F0H,120H ↵
00F0   55 AA 33 88 55 AA 33 88  55 AA 33 88 55 AA 33 88

0100   55 AA 33 88 55 AA 33 88  55 AA 33 88 55 AA 33 88
0110   55 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0120   FF
*
```

次に，メモリの中からＡＡ，３３のデータ列を探します．

```
*MEM G 0,FFH AA,33↵
00F1
00F5
00F9
00FD


*MEM G 100H,13FH AA,33,88,55,AA,33↵
0101
0105
0109

*
```

３．３．７　メモリの転送（Ｍｏｖｅ）

ＭＥＭ␣Ｍ␣ｐａｒｔｉｔｉｏｎ␣ａｄｄｒ↵

ｐａｒｔｉｔｉｏｎで指定されたメモリをａｄｄｒを先頭とするメモリへ転送します．

まず転送するメモリの内容を示します．

```
*MEM D 0,FFH ↵
0000   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
0010   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
0020   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
0030   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
0040   05 06 07 08 09 00 01 02  03 04 05 06 07 08 09 00
0050   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
*                                                  CTRL-Cまたは ESCｷｰ入力
```

次に転送先のメモリの内容を示します．

```
*MEM D 3000H,30FFH ↵
3000   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
3010   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
3020   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
3030   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
3040   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
3050   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
3060   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
3070   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
3080   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
3090   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
*                                                  CTRL-Cまたは ESCｷｰ入力
```

メモリ・ムーブ・コマンドを実行すると，３０００Ｈ～３０ＦＦＨに０～ＦＦＨのメモリの内容がコピーされます．０～ＦＦＨのメモリはそのままです．

```
*MEM M 0,FFH 3000H ↵

*MEM D 3000H,30FFH ↵
3000   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
3010   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
3020   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
3030   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
3040   05 06 07 08 09 00 01 02  03 04 05 06 07 08 09 00
3050   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
3060   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
3070   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
3080   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
*                                              CTRL-C または ESCｷｰ入力
```

保守／廃止

３．３．８　メモリの比較（Ｖｅｒｉｆｙ）

ＭＥＭ␣Ｖ␣ｐａｒｔｉｔｉｏｎ␣ａｄｄｒ↵

ｐａｒｔｉｔｉｏｎで指定されたメモリとａｄｄｒを先頭とするメモリとの比較を行ない，すべて一致したら何も表示せずに‘＊’に戻ります．一致しなければアドレスとデータを表示します．

メモリを次のようにイニシャライズします．

```
*MEM F 0,FFH 1,2,3,4,5,6,7,8,9,0↵

*MEM D XXH↵
0000   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
0010   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
0020   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
0030   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
0040   05 06 07 08 09 00 01 02  03 04 05 06 07 08 09 00
0050   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
0060   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
0070   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
0080   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
0090   05 06 07 08 09 00 01 02  03 04 05 06 07 08 09 00
00A0   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
00B0   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
00C0   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
00D0   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
00E0   05 06 07 08 09 00 01 02  03 04 05 06 07 08 09 00
00F0   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
*
```

次に，０～ＦＨのメモリの内容と２０Ｈ～２ＦＨのメモリの比較をします．

```
*MEM V 0,FH 20H ↵
0000 01   0020 03
0001 02   0021 04
0002 03   0022 05
0003 04   0023 06
0004 05   0024 07
0005 06   0025 08
0006 07   0026 09
0007 08   0027 00
0008 09   0028 01
0009 00   0029 02
000A 01   002A 03
000B 02   002B 04
000C 03   002C 05
000D 04   002D 06
000E 05   002E 07
000F 06   002F 08
*
```

比較したメモリのデータが一致しない時は，上のように出力します．

一致すると以下のようになります．

```
*MEM V 0,1FH 64H ↵
*
```

３．３．９　メモリ内容の交換（ｅＸｃｈａｎｇｅ）

ＭＥＭ␣Ｘ␣ｐａｒｔｉｔｉｏｎ␣ａｄｄｒ↵

ｐａｒｔｉｔｉｏｎのメモリとａｄｄｒを先頭とするメモリの内容を交換します．

以下に０～３ＦＨのメモリと，２０００Ｈ～２０３ＦＨのメモリの内容の交換例を示します．

```
*MEM D 0,3FH ↵
0000   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
0010   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
0020   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
0030   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
*
*MEM D 2000H,203FH ↵
2000   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
2010   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
2020   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
2030   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
*
*MEM X 0,3FH 2000H ↵

*MEM D 0,3FH ↵
0000   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
0010   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0020   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
0030   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
*
*MEM D 2000H,20FFH ↵
2000   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
2010   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
2020   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
2030   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
2040   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
2050   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
2060   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
2070   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00

*                                    CTRL-Cまたは ESCｷｰ入力
```

保守／廃止

３．４　ＲＥＧ（レジスタ）コマンド

表現形式：

＊ＲＥＧ [␣ {Ｃ / Ｄ} [␣ｒｅｇｉｓｔｅｒ－ｎａｍｅ]]

機　能：　レジスタの内容の変更，表示を行ないます。

メイン・コマンド	：	ＲＥＧ	
サブ・コマンド	：	Ｃ	レジスタ・チェンジ
	：	Ｄ	レジスタ・ダンプ
オペランド	：	ｒｅｇｉｓｔｅｒ－ｎａｍｅ	レジスタ名
オプション	：	なし	

３．４．１　レジスタの概要

コマンドで表記するレジスタ名はあらかじめ登録されております．レジスタの数値入力は，ＳＵＦコマンドで指定された基数に従い入力を判定します．（ただし，表示は１６進数です．）

（１）レジスタ名（ｒｅｇｉｓｔｅｒ－ｎａｍｅ）

ＰＳＷ	；	フラグ・レジスタ	ＰＣ	；	プログラム・カウンタ
Ｖ	；	レジスタＶ	Ｖ’	；	レジスタＶ’（裏）
Ａ	；	アキュームレータ	Ａ’	；	レジスタＡ’（裏）
Ｂ	；	レジスタＢ	Ｂ’	；	レジスタＢ’（裏）
Ｃ	；	レジスタＣ	Ｃ’	；	レジスタＣ’（裏）
Ｄ	；	レジスタＤ	Ｄ’	；	レジスタＤ’（裏）
Ｅ	；	レジスタＥ	Ｅ’	；	レジスタＥ’（裏）
Ｈ	；	レジスタＨ	Ｈ’	；	レジスタＨ’（裏）

保守／廃止

Ｌ　；　レジスタＬ　　Ｌ’　；　レジスタＬ’（裏）

ＥＡ　；　拡張アキュームレータ　　ＥＡ’；　拡張アキュームレータ（裏）

ＳＰ　；　スタック・ポインタ

保守／廃止

3．4．2 変 更

REG␣C〔␣register－name〕↵

指定されたレジスタの内容を変更します．register－nameが省略された場合は，すべてのレジスタの内容を順に変更します．

入力例）

```
＊REG  C  EA' ↵          ①
＊REG  C ↵               ②
```

① レジスタEA'の内容を変更します．レジスタEA'の内容が表示された後，直ちに‘↵’を入力しますと内容を変更せずに終了します．

② すべてのレジスタの内容を順次変更します．レジスタの内容が表示された後，‘␣’を入力しますと，内容を変更せずに次のレジスタを表示します．また，‘↵’を入力しますと，表示中のレジスタの内容は変更せずに終了します．

レジスタの変更順序を次に示します．

PSW，V，A，B，C，D，E，H，L，EA，SP，PC，V'，A'，B'，C'，D'，E'，H'，L'，EA'

このコマンドは最後のEA'が変更された時点で自動的に終了します．

以下に変更の例を示します．

*REG C↵ ←――― register-nameを省略した場合すべてのﾚｼﾞｽﾀの変更可能

```
PSW 00=1 ↵
V 8E=2 ↵
A D6=3 ↵
B 61=4 ↵
C 11=5 ↵
D 05=6 ↵
E 58=7 ↵
H EC=8 ↵
L E9=9 ↵
EA 2AB8=10 ↵
SP 7000=11 ↵
PC 0000=12 ↵
V' 00=13 ↵
A' 00=14 ↵
B' 00=15 ↵
```

保守／廃止

```
 C' 00=16 ↲
 D' 00=17 ↲
 E' 00=18 ↲
 H' 00=19 ↲
 L' 00=20 ↲
 EA' 0000=21↲
```

結果は以下のようになります．

```
*REG↲

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 01  02 03 04 05 06 07 08 09 0010 0011 0012
     V' A' B' C' D' E' H' L' EA'
     13 14 15 16 17 18 19 20 0021

*
```

以下に，指定されたレジスタのみの変更を示します．

```
*REG C PSW↲
                              ← PSWレジスタの変更
 PSW 01=7D↲

*REG↲

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 7D  02 03 04 05 06 07 08 09 0010 0011 0012
     V' A' B' C' D' E' H' L' EA'
     13 14 15 16 17 18 19 20 0021

*

*REG C V↲
                              ← Vレジスタの変更
 V 02=AA↲

 REG↲

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 7D  AA 03 04 05 06 07 08 09 0010 0011 0012
     V' A' B' C' D' E' H' L' EA'
     13 14 15 16 17 18 19 20 0021

*REG C A↲

 A 03=03
      └──────スペース入力した場合は変更は行なわれない．
*REG↲
```

保守／廃止

```
PSW V  A  B  C  D  E  H  L  EA   SP   PC
7D  AA 03 04 05 06 07 08 09 0010 0011 0012
    V' A' B' C' D' E' H' L' EA'
    13 14 15 16 17 18 19 20 0021

*
```

下記の場合エラーを生じます．

```
*REG C PSW'  ↵
       ↑
       └──────────登録されていない register-nameを使用した場合
COMMAND FORMAT ERROR

*REG CA  ↵
     ↑
     └──────────── register-nameの前にスペースがない場合
COMMAND FORMAT ERROR

*REG C A ↵

A 03=FHG
     ↑
     └──────────１６進以外の値を入力した場合
INPUT DATA ERROR
                  ──────────この場合，Aの値はそのまま
*REG D ↵

PSW V  A  B  C  D  E  H  L  EA   SP   PC
7D  AA 03 04 05 06 07 08 09 0010 0011 0012
    V' A' B' C' D' E' H' L' EA'
    13 14 15 16 17 18 19 20 0021

*REG C ↵

PSW 7D=00 ↵
V AA=11 ↵
A 03=22 ↵
B 04=33 ↵
C 05= ↵  ←──────────  '↵' を入力した場合は終了します．

*REG D ↵

PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  11 22 33 05 06 07 08 09 0010 0011 0012
    V' A' B' C' D' E' H' L' EA'
    13 14 15 16 17 18 19 20 0021

*REG C ↵

PSW 00=1 ↵
V 11=2 ↵
A 22=3 ↵
B 33=4 ↵
C 05=GFY ←──────────１６進以外の値を入力した場合も終了します．

INPUT DATA ERROR
```

保守／廃止

```
*REG ↵

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 01  02 03 04 05 06 07 08 09 0010 0011 0012
     V' A' B' C' D' E' H' L' EA'
     13 14 15 16 17 18 19 20 0021
```

以下に１６進以外の値入力の例を示します.

```
*SUF ↵

 H
```

```
*REG C ↵   ←――――――入力値の後にｻﾌｨｯｸｽ指定がない場合は以下１６進を示す.

 PSW 01=1Q ↵   ←――――――――――――8 進数の値を示す.
 V 02=2Q ↵
 A 03=4Q ↵
 B 04=10Q  ↵
 C 05=20Q  ↵
 D 06=40Q  ↵
 E 07=100Q ↵
 H 08=200Q ↵
 L 09=300Q ↵
 EA 0010=177777Q ↵
 SP 0011=100000Q ↵
 PC 0012=4000Q ↵
 V' 13=11Q ↵
 A' 14=22Q ↵
 B' 15=33  ↵   ←――――――――――――指示がないため１６進とみなす.
 C' 16=44Q ↵
 D' 17=55Q ↵
 E' 18=66Q ↵
 H' 19=77Q ↵
 L' 20=111Q  ↵
 EA' 0021=22222Q ↵
```

```
*REG ↵

 PSW V  A  B  C  D  E  H  L  EA   SP   PC   ┐
 01  02 04 08 10 20 40 80 C0 FFFF 8000 0800  ┘  １６進に変換されます.
     V' A' B' C' D' E' H' L' EA'
     09 12 33 24 2D 36 3F 49 2492
           ↑
*          └――値はそのまま

*REG C ↵

 PSW 01=1234 ↵
 V 02=2345H  ↵
 A 04=3456789H ↵
 B 08=BFDFBFCDE58F43C0A17574 ↵
 C 10= ↵

*REG ↵
```

```
 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 34  45 89 74 10 20 40 80 C0 FFFF 8000 0800
     V' A' B' C' D' E' H' L' EA'
     09 12 33 24 2D 36 3F 49 2492

*
```

ＰＳＷ　Ｖ，Ａ，Ｂ，Ｃ　レジスタは２桁表示の為，入力値は末尾２桁のみ有効となります．

```
*REG C↵

 PSW 34=547111Q  ↵
 V 45=645012Q  ↵
 A FF=67253361Q↵
 B 18=5136547380↵

 INPUT DATA ERROR

*REG↵

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 49  0A F1 74 10 20 40 80 C0 FFFF 8000 0800
     V' A' B' C' D' E' H' L' EA'
     09 12 33 24 2D 36 3F 49 2492

*
```

ＰＳＷ，Ｖ，Ａ，Ｂレジスタは２桁表示の為，入力値が８進の場合は末尾３桁は０００Ｑ～３７７Ｑまで有効となります．上の例で，Ｂレジスタ入力はーーー７３８Ｑなのでエラーとなります．結果はＰＳＷ，Ｖ，Ａレジスタまでは変更されましたがＢレジスタは元のままです．

保守／廃止

３．４．３　表　示

ＲＥＧ〔␣Ｄ〔␣ｒｅｇｉｓｔｅｒ－ｎａｍｅ〕〕↵

指定されたレジスタの内容を表示します．ｒｅｇｉｓｔｅｒ－ｎａｍｅが省略された場合は，すべてのレジスタの内容を表示します．

入力例）

```
＊ＲＥＧ　Ｄ　Ａ↵　　①
＊ＲＥＧ　Ｄ↵　　②
＊ＲＥＧ↵　　③
```

①　レジスタＡの内容を表示します．

②，③　すべてのレジスタの内容を表示します．

例）

```
*REG↵

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 7D  AA 22 04 05 06 07 08 09 0010 0011 0012
     V' A' B' C' D' E' H' L' EA'
     13 14 15 16 17 18 19 20 0021

*REG D↵

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 7D  AA 22 04 05 06 07 08 09 0010 0011 0012
     V' A' B' C' D' E' H' L' EA'
     13 14 15 16 17 18 19 20 0021

*
```

指示されたレジスタ表示を下記に示します．

```
*REG D PSW↵

 PSW 7D

*REG D V↵

 V AA
```

```
*REG D A )

 A F1

*

*REG D TY  )
 COMMAND FORMAT ERROR

*REG DA  )
 COMMAND FORMAT ERROR

*
```

登録されていない register-nameを指定したためにｴﾗｰが生じた.

ｽﾍﾟｰｽがないために ｴﾗｰが生じた.

保守／廃止

３．５　モード・レジスタ・コマンド

表現形式：

＊ＭＤＲ [␣ {Ｃ [␣ｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ１]
　　{Ｄ [␣ｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ２]}]

機　能：　モード・レジスタの変更，表示を行ないます．

メイン・コマンド　：　ＭＤＲ

サブ・コマンド　：　Ｃ　　レジスタ・チェンジ

：　Ｄ　　レジスタ・ダンプ

オペランド　：　ｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ１

モード・レジスタ名１

：　ｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ２

モード・レジスタ名２

３．５．１　ＩＥ－７８Ｃ１１の有するモード・レジスタ

（１）ｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ１

ＭＡ　；　ＭＯＤＥ　Ａレジスタ

ＭＢ　；　ＭＯＤＥ　Ｂレジスタ

ＭＣＣ　；　ＭＯＤＥ　ＣＯＮＴＲＯＬ　Ｃレジスタ

ＭＣ　；　ＭＯＤＥ　Ｃレジスタ

ＭＭ　；　ＭＥＭＯＲＹ　ＭＡＰＰＩＮＧレジスタ

ＭＦ　；　ＭＯＤＥ　Ｆレジスタ

ＴＭＭ　；　タイマ・モード・レジスタ

ＥＴＭＭ　；　タイマ／イベント・カウンタ・モード・レジスタ

ＥＯＭ　；　タイマ／イベント・カウンタ・アウトプット・モード・レジスタ

ＳＭＬ　；　シリアル・モード・レジスタ（Ｌｏｗ）

SMH　；　シリアル・モード・レジスタ（High）

ANM　；　A／Dチャネル・モード・レジスタ

ZCM　；　ゼロクロス・モード・レジスタ

注）mode－register－name1に属するレジスタは書込み可能レジスタです.

（2）mode－register－name2

TMM　；　タイマ・モード・レジスタ

EOM　；　タイマ／イベント・カウンタ・アウトプット・モード・レジスタ

SMH　；　シリアル・モード・レジスタ（High）

ANM　；　A／Dチャネル・モード・レジスタ

注）mode－register－name2に属するレジスタはリード／ライトともに可能です.

保守／廃止

３．５．２　変　更

ＭＤＲ␣Ｃ〔␣ｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ１〕↵

指定されたモード・レジスタの内容を変更します．ｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ１が省略されて場合は，すべてのモード・レジスタの内容を順に変更します．

入力例）

```
＊ＭＤＲ　Ｃ　ＭＣ↵　　①
＊ＭＤＲ　Ｃ↵　　②
```

① ＭＣレジスタの内容を変更します．ＭＣレジスタの内容が表示された後，直ちに‘↵’を入力しますと，内容を変更せずに終了します．

② すべてのモード・レジスタの内容を順次変更します．モード・レジスタの内容が表示された後，‘␣’を入力しますと内容を変更せずに，次のモード・レジスタを表示します．また，‘↵’を入力しますと表示中のモード・レジスタの内容を変更せずに終了します．モード・レジスタの変更順序を次に示します．

ＭＡ，ＭＢ，ＭＣＣ，ＭＣ，ＭＭ，ＭＦ，ＴＭＭ，ＥＴＭＭ，ＥＯＭ，
ＳＭＬ，ＳＭＨ，ＡＮＭ，ＺＣＭ

このコマンドは，最後のＺＣＭが変更された時点で自動的に終了します．

保守／廃止

３．５．３　表　示

ＭＤＲ〔␣Ｄ〔␣ｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ２〕〕↵

指定されたモード・レジスタの内容を表示します．ｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ２が省略された場合は，すべてのｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ２の内容を表示します．

入力例）

```
＊ＭＤＲ　Ｄ　ＴＭＭ↵　　①
＊ＭＤＲ　Ｄ↵　　②
＊ＭＤＲ↵　　③
```

①　ＴＭＭレジスタの内容を表示します．

②，③　すべてのｍｏｄｅ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ２の内容を表示します．

以下に例を示します．

例）

```
*MDR ↵

 TMM EOM SMH ANM
 FF  00  00  00

*MDR D ↵

 TMM EOM SMH ANM
 FF  00  00  00

*MDR D TMM ↵

 TMM FF

*MDR D EOM ↵

 EOM 00

*
```

保守／廃止

以下にモード・レジスタの変更と表示の例を示します．

例)

```
*MDR C ↵

 MA --=12 ↵
 MB --=23 ↵
 MCC --=34 ↵
 MC --=45 ↵
 MM --=56 ↵
 MF --=67 ↵
 TMM FF=78 ↵
 ETMM --=89 ↵
 EOM 00=90 ↵
 SML --=AB ↵
 SMH 00=BC ↵
 ANM 00=CD ↵
 ZCM --=CD ↵

*MDR ↵

 TMM EOM SMH ANM
 78  90  BC  CD

*MDR C MA ↵

 MA --=11

*MDR D MA ↵
 WRITE ONLY REGISTER

*

*MDR C

 MA --=--
 MB --=--
 MCC --=--
 MC --=--
 MM --=--
 MF --=--
 TMM 78=78
 ETMM --=--
 EOM 90=90
 SML --=--
 SMH BC=BC
 ANM CD=CD
 ZCM --=--

*

*MDR C AMN ↵
 COMMAND FORMAT ERROR

*
```

mode-register-name1 は表示する事はできない．

スペース入力の場合，元の値がそのまま残ります．

登録以外のmode-register-nameを指定した場合はエラーとなります．


```
*MDR C EOM )

 EOM 90=AA

*MDR D EOM )

 EOM AA

*MDR C EOM )

 EOM AA=AA
        └──────────────スペース入力

*MDR C EOM )

 EOM AA= )      ←──────────── ‘)’を入力の場合，変更を終了します．

*MDR )

 TMM EOM SMH ANM
 78  AA  BC  CD

*

*SUF )   ←──────────サフィックスが10進の場合

 T

*

*MDR C )

 MA --=12H )
 MB --=23H )
 MCC --=34Q  )
 MC --=45Q )
 MM --=56T )
 MF --=67T )
 TMM 78=78 )
 ETMM --=89  )
 EOM AA=111010Y  )
 SML --=1111Y  )
 SMH BC=33 )
 ANM CD=65H  )
 ZCM --=43H  )

*MDR )

 TMM EOM SMH ANM
 4E  3A  21  65

*
```

他のサフィックスでも上記のように可能です．

保守／廃止

3．6　特殊レジスタ・コマンド

表現形式：

```
        ┌    ┌C〔␣special－register－name1〕┐┐
*SPR   │␣ ┤                                 ├│ ↵
        └    └D〔␣special－register－name2〕┘┘
```

機　能：　特殊レジスタの変更，表示を行ないます．

メイン・コマンド	：	SPR	
サブ・コマンド	：	C	特殊レジスタ・チェンジ
	：	D	特殊レジスタ・ダンプ
オペランド	：	special－register－name1	特殊レジスタ名1
	：	special－register－name2	特殊レジスタ名2
オプション	：	なし	

3．6．1　IE－78C11の有する特殊レジスタ

(1) special－register－name1

PA	；	PORT　A　レジスタ
PB	；	PORT　B　レジスタ
PC	；	PORT　C　レジスタ
PD	；	PORT　D　レジスタ
PF	；	PORT　F　レジスタ
MKH	；	MASK　High　レジスタ
MKL	；	MASK　Low　レジスタ
TXB	；	TX　バッファ・レジスタ
TM0	；	タイマ・レジスタ0

TM1　；　タイマ・レジスタ1

ETM0　；　タイマ／イベント・カウンタ・モード・レジスタ0

ETM1　；　タイマ／イベント・カウンタ・モード・レジスタ1

注）special－register－name1に属するレジスタは書込み可能レジスタです.

(2) special－register－name2

PA　；　PORT A レジスタ

PB　；　PORT B レジスタ

PC　；　PORT C レジスタ

PD　；　PORT D レジスタ

PF　；　PORT F レジスタ

MKH　；　MASK High レジスタ

MKL　；　MASK Low レジスタ

RXB　；　RX バッファ・レジスタ

CR0　；　A／D変換リザルト0

CR1　；　A／D変換リザルト1

CR2　；　A／D変換リザルト2

CR3　；　A／D変換リザルト3

ECNT　；　タイマ／イベント・カウンタ・アップ・カウンタ

ECPT　；　タイマ／イベント・カウンタ・キャプチュア

注）special－register－name2に属するレジスタはリード可能です.

保守／廃止

３．６．２　変　更

ＳＰＲ␣Ｃ〔␣ｓｐｅｃｉａｌ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ１〕⤶

指定された特殊レジスタの内容を変更します．ｓｐｅｃｉａｌ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ１が省略された場合は，すべての特殊レジスタの内容を変更します．

入力例）

```
＊ＳＰＲ　Ｃ　ＰＦ⤶　　①
＊ＳＰＲ　Ｃ⤶　　②
```

① ＰＦレジスタの内容を変更します．ｓｐｅｃｉａｌ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ１が省略された場合は，すべての特殊レジスタの内容を順に変更します．

② すべての特殊レジスタの内容を順次変更します．特殊レジスタの内容が表示された後，‘␣’を入力しますと，内容を変更せずに次の特殊レジスタを表示します．また，‘⤶’を入力しますと，表示中の特殊レジスタの内容を変更せずに終了します．特殊レジスタの変更順序を次に示します．

ＰＡ，ＰＢ，ＰＣ，ＰＤ，ＰＦ，ＭＫＨ，ＭＫＬ，ＴＸＢ，ＴＭ０，ＴＭ１，ＥＴＭ０，ＥＴＭ１

ＥＴＭ１を変更した時点でコマンドの実行が自動的に終了されます．

（例１）特殊レジスタの変更がモード・レジスタの内容に影響される場合

```
①  *SPR C PA ⤶
     PA 00=12 ⤶
②  *SPR⤶
     PA PB PC PD PF MKH MKL RXB CR0 CR1 CR2 CR3 ECNT ECPT
     00 00 00 CB 00 07  FE  FF  FF  FF  FF  FF  0000 FFFF
③  *MDR C MA ⤶
     MA --=0⤶
```

④ *SPR ⤵

```
 PA PB PC PD PF MKH MKL RXB CR0 CR1 CR2 CR3 ECNT ECPT
 12 00 00 CB 00 07  FE  FF  FF  FF  FF  FF  0000 FFFF
*
```

①でＰＡレジスタを変更し，②で表示させたところ，ＰＡレジスタの内容に変更がみられなかった．これはＭＡレジスタが入力モードに設定されていたためです．③でＭＡレジスタを出力モードに変更し，④で再度表示させます．ＰＡレジスタの内容が変更されています．

ＰＡ，ＰＢ，ＰＣ，ＰＤ，ＰＦの特殊レジスタは，ＭＡ，ＭＢ，ＭＣＣ，ＭＣ，ＭＭ，ＭＦの各モード・レジスタの内容により，そのままでは直接変更されない場合がありますので注意してください．

（例２）ＭＢレジスタを出力モードに設定後，ＰＢレジスタを変更した場合

```
*SPR ⤵

 PA PB PC PD PF MKH MKL RXB CR0 CR1 CR2 CR3 ECNT ECPT
 12 00 00 CB 00 07  FE  FF  FF  FF  FF  FF  0000 FFFF

*MDR C MB  ⤵

 MB --=0 ⤵

*SPR C PB  ⤵

 PB 00=34  ⤵

*SPR ⤵

 PA PB PC PD PF MKH MKL RXB CR0 CR1 CR2 CR3 ECNT ECPT
 12 34 00 CB 00 07  FE  FF  FF  FF  FF  FF  0000 FFFF

*
```

（例３）ｓｐｅｃｉａｌ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ１の変更コマンドを入力し，順次‘␣’を入力した場合

```
*SPR C ↵

 PA 00=00
 PB 00=00
 PC 00=00
 PD CB=CB
 PF 00=00
 MKH 07=07
 MKL FE=FE
 TXB --=--      ┐
 TM0 --=--      │
 TM1 --=--      ├  special-register-name2にない特殊ﾚｼﾞｽﾀは，読出せないため
 ETM0 --=--     │   ‘--’を表示します．
 ETM1 --=--     ┘

*
```

（例４）読出し専用の特殊レジスタを変更しようとした場合

```
*SPR C RXB ↵

 READ ONLY REGISTER

*
```

Ｓｐｅｃｉａｌ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ１にない特殊レジスタは，読出し専用のため変更できません．

３．６．３　表　示

ＳＰＲ〔␣Ｄ〔␣ｓｐｅｃｉａｌ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ２〕〕↵

指定された特殊レジスタの内容を表示します．ｓｐｅｃｉａｌ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ２が省略された場合は，

すべてのｓｐｅｃｉａｌ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ２の内容を表示します．

入力例）

```
＊ＳＰＲ　Ｄ　ＲＸＢ↵　　①
＊ＳＰＲ　Ｄ↵　　②
＊ＳＰＲ↵　　③
```

①　ＲＸＢレジスタの内容を表示します．

②，③　すべてのｓｐｅｃｉａｌ－ｒｅｇｉｓｔｅｒ－ｎａｍｅ２の内容を表示します．

以下に例を示します．

例）

```
*SPR ↵

 PA PB PC PD PF MKH MKL RXB CR0 CR1 CR2 CR3 ECNT ECPT
 00 00 00 DB 00 07  FE  FF  FF  FF  FF  FF  0000 FFFF

*

*SPR D ↵

 PA PB PC PD PF MKH MKL RXB CR0 CR1 CR2 CR3 ECNT ECPT
 00 00 00 DB 00 07  FE  FF  FF  FF  FF  FF  0000 FFFF

*

*SPR D PA ↵

 PA 00

*

*SPR D MKH ↵

 MKH 07

*
```

保守／廃止

```
*SPR D TXB ↵
 WRITE ONLY REGISTER
*
```

special-register-namelを指定した場合は次のように表示されます.

保守／廃止

３．７　ＬＯＤ（ロード）コマンド

表現形式：

＊ＬＯＤ [{␣ / =} ｉｎｐｕｔ－ｄｅｖｉｃｅ－ｎａｍｅ] 〔␣＆ｂｉａｓ〕

機　能：　指定されたディバイスからＨＥＸ形式のデータをマッピングされたプログラム・メモリにバイアスを付けて転送（ロード）します．

メイン・コマンド	：	ＬＯＤ
サブ・コマンド	：	なし
オペランド	：	ｉｎｐｕｔ－ｄｅｖｉｃｅ－ｎａｍｅ　入力装置名
オプション	：	ｂｉａｓ　　バイアス・アドレス

入力装置は，ＩＥコントロール・ボードにあるシリアル端子（ＲＳ－２３２－Ｃ）のＴＴＹ１，ＴＴＹ２のうちどちらか一つを指定するものです．

保守／廃止

３．７．１　ロード

ＬＯＤ␣ＴＴＹ？＄ｂｉａｓ↵

○ ‘ＴＴＹ？’　：ＴＴＹ１かＴＴＹ２です．

○ ‘ＴＴＹ？’の前のスペースは‘＝’でもかまいません．

○＄ｂｉａｓは，数値入力でサフィクスに従います．

　省略した時は，ｂｉａｓは０になります．

以下に例を示します．

```
*MEM F 0,FFH 00 ↵

*MEM D XXH↵
0000   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0010   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0020   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0030   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0040   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0050   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0060   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0070   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00

*LOD TTY1 ↵                                    ←CTRL-Cまたは ESCｷｰ入力

*
*MEM D XXH↵
0000   FF FF FF FF FF 7A 33 26  00 FF FF FF FF FF FF FF
0010   21 1E 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0020   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
0030   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0040   05 06 07 08 09 00 01 02  03 04 05 06 07 08 09 00
0050   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
0060   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
0070   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
0080   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
0090   05 06 07 08 09 00 01 02  03 04 05 06 07 08 09 00
00A0   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
00B0   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
00C0   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
00D0   09 00 01 02 03 04 05 06  07 08 09 00 01 02 03 04
00E0   05 06 07 08 09 00 01 02  03 04 05 06 07 08 09 00
00F0   55 AA 33 88 55 AA 33 88  55 AA 33 88 55 AA 33 88

*
```

```
*MEM F 0,FFH 0↵

*MEM D XXH ↵
0000   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0010   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0020   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0030   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0040   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0050   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0060   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0070   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0070   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0080   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0090   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00A0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00B0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00C0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00D0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00E0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00F0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
*

*MEM F XXX 0↵

*MEM D 800,8FF ↵
 0800  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
 0810  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
 0820  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
 0830  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
 0840  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
 0850  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
 0860  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00

*LOD=TTY1$800  ↵                                  CTRL-Cまたは ESCｷｰ入力

*MEM D 800,8FF ↵
 0800  11 22 33 44 55 66 77 88  11 22 33 44 55 66 77 88
 0810  11 22 33 44 55 66 77 88  11 22 33 44 55 66 77 88
 0820  11 22 33 44 55 66 77 88  11 22 33 44 55 66 77 88
 0830  11 22 33 44 55 66 77 88  11 22 33 44 55 66 77 88
 0840  11 22 33 44 55 66 77 88  11 22 33 44 55 66 77 88
 0850  11 22 33 44 55 66 77 88  11 22 33 44 55 66 77 88
 0860  11 22 33 44 55 66 77 88  11 22 33 44 55 66 77 88

*                                                 CTRL-Cまたは ESCｷｰ入力
```

保守／廃止

３．８　ＳＡＶ（セーブ）コマンド

表現形式：

＊ＳＡＶ〔␣ｏｕｔｐｕｔ－ｄｅｖｉｃｅ－ｎａｍｅ〕

〔｛␣／＝｝ｐａｒｔｉｔｉｏｎ〕

機　能：　指定されたディバイスに，ｐａｒｔｉｔｉｏｎで指定されるプログラム・メモリの内容をＨＥＸ形式のデータでセーブします．

メイン・コマンド	：	ＳＡＶ	
サブ・コマンド	：	なし	
オペランド	：	ｏｕｔｐｕｔ－ｄｅｖｉｃｅ－ｎａｍｅ	出力装置名
	：	ｐａｒｔｉｔｉｏｎ	セーブ領域
オプション	：	なし	

３．８．１　セーブ

ＳＡＶ␣ＴＴＹ?␣ｐａｒｔｉｔｉｏｎ↵

○ ‘ＴＴＹ?’はＴＴＹ１（ＣＨ１）またはＴＴＹ２（ＣＨ２）

○ ‘ＴＴＹ?’と‘ｐａｒｔｉｔｉｏｎ’の間のスペースは‘＝’でもかまいません．

○ ‘ｐａｒｔｉｔｉｏｎ’は省略可能です．省略した時は，マッピングされているエリアをすべてセーブします．

以下に例を示します．

```
*MEM D XXH↵
0000   FF FF FF FF FF 7A 33 26  00 FF FF FF FF FF FF FF
0010   21 1E 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0020   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
0030   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0040   05 06 07 08 09 00 01 02  03 04 05 06 07 08 09 00
0050   01 02 03 04 05 06 07 08  09 00 01 02 03 04 05 06
0060   07 08 09 00 01 02 03 04  05 06 07 08 09 00 01 02
0070   03 04 05 06 07 08 09 00  01 02 03 04 05 06 07 08
                                 ↖CTRL-Cまたは ESCｷｰ入力
```

```
*SAV TTY1=0,FFH↵
:10000000FFFFFFFFFF7A332600FFFFFFFFFFFFFF29
:10001000211E0000000000000000000000000000A1
:10002000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFE0
:1000300000000000000000000000000000000000C0
:1000400005060708090001020304050607080900 60
:1000500001020304050607080900010203040506 5E
:1000600007080900010203040506070809000102 48
:1000700003040506070809000102030405060708 32
:1000800009000102030405060708090001020304 30
:1000900005060708090001020304050607080900 10
:1000A0000102030405060708090001020304050 60E
:1000B0000708090001020304050607080900010 2F8
:1000C0000304050607080900010203040506070 8F2
:1000D0000900010203040506070809000102030 4E0
:1000E0000506070809000102030405060708090 0C0
:1000F00055AA338855AA338855AA338855AA338818
:00000001FF
```

CRT が接続されている時は，
HEX形式のｺｰﾄﾞが表示されます．

```
*SAV TTY1 XXH ↵
:10000000FFFFFFFFFF7A332600FFFFFFFFFFFFFF29
:10001000211E000000000000000000000000000A1
:10002000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFE0
:10003000000000000000000000000000000000C0
:10004000050607080900010203040506070809060
:10005000010203040506070809000102030405065E
:10006000070809000102030405060708090001024 8
:10007000030405060708090001020304050607083 2
:10008000090001020304050607080900010203043 0
:10009000050607080900010203040506070809001 0
:1000A000010203040506070809000102030405060E
:1000B000070809000102030405060708090001 02F8
:1000C000030405060708090001020304050607 08E2
:1000D000090001020304050607080900010203 04E0
:1000E000050607080900010203040506070809 00C0
:1000F00055AA338855AA338855AA338855AA33 8818
:00000001FF

*SAV TTY2=0,FFH ↵

*
```

ＴＴＹ２にデータを送って‘＊’に戻ります.

保守／廃止

3．9　RUN（エミュレーション）コマンド

表現形式：

＊RUN [␣ {N / B} 〔␣addr〕]

＊RUN␣S 〔␣addr〕〔, step－No.〕

＊RUN␣T 〔␣addr〕〔, {step－No. / ※}〕〔$ {D / E}〕〔$R〕

※＝register－name {＝ / ＞ / ＜ / ＝＜ / ＞＝ / ＜＞ / ＞＜ / ＝＞ / ＜＝} value

機　能：　リアルタイム・エミュレーションおよび1ステップ・エミュレーションを行ないます.

メイン・コマンド	：	RUN	
サブ・コマンド	：	N	ノーマル（Normal）
	：	B	ブレーク（Break）
	：	S	ステップ（Step）
	：	T	トレース（Trace）
オペランド	：	addr	式入力可能な数値（2バイト）
	：	step－No.	ステップ回数（1バイト）
	：	value	数　値
オプション	：	D	表示ディセーブル
	：	E	表示イネーブル（ディフォールト）
	：	R	レジスタ表示あり

保守／廃止

３．９．１　エミュレーション・コマンドの概要

エミュレーション・コマンドには以下の種類があります.

○ＲＵＮ　Ｎ

ブレークなし　リアルタイム実行

○ＲＵＮ　Ｂ

ブレーク付き　リアルタイム実行

○ＲＵＮ　Ｓ

ステップ数指定リアルタイム実行

○ＲＵＮ　Ｔ

１ステップ実行（表示付き）

3．9．2　RUN　N（ブレークなしリアルタイム実行）

RUN␣N〔␣addr〕↵

入力例）

```
＊RUN　N↵                    ①
＊RUN　N　0↵                 ②
```

① 現在のプログラム・カウンタで示されるアドレスから，リアルタイムでエミュレーションを開始します．

② 0番地からリアルタイムでエミュレーションを開始します．

以下に例を示します．

例）

```
*RUN N 0↵

 USER-SYSTEM VDD-ON

 JUST MOMENT
 EMULATION START AT 0000

 ESC BREAK     ←――――(ESCｷｰ入力による強制ﾌﾞﾚｰｸ) ※
 TERMINATED

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 00  08 59 18 18 26 25 00 00 0000 0603 1E0B
     V' A' B' C' D' E' H' L' EA
     00 FF 20 30 00 04 06 FD 014C

*
```

※ ただし，HLT命令，STOP命令を実行中には‘NON　BREAK’と表示されます．次のコマンドはRESコマンドでなければなりません．

プログラムの中断は上記のようにESCキーを入力することにより行なえます．CTRL／C入力では，コマンド待ちになりますがCPUは実行したままになっていますので注意してください．

この状態ではほとんどのコマンドは受けつけられません．


```
*RUN T 100,2$E ↵

 ADRS   OBJECT          MNEMONIC
 0100   340010          LXI     H,0100OH

 ADRS   OBJECT          MNEMONIC
 0103   6091            XRA     A,A

 ONE STEP EMULATION STANDBY ←――――スペース入力

 ADRS   OBJECT          MNEMONIC
 0105   4DD2            MOV     MA,A

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 40  1C 00 6E FE 18 90 10 00 0000 0700 0107
     V' A' B' C' D' E' H' L' EA'
     00 FF 00 10 07 00 90 F0 014C ←――――― ‘↵’ 入力


*RUN T 100,2$D ↵

 ONE STEP EMULATION STANDBY ←――――スペース入力

 ADRS   OBJECT          MNEMONIC
 0105   4DD2            MOV     MA,A

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 40  1C 00 6E FE 18 90 10 00 0000 0700 0107
     V' A' B' C' D' E' H' L' EA'
     00 FF 00 10 07 00 90 F0 014C ←――――― ‘↵’ 入力
```


```
*RUN T 100,2$E$R ⤶

 ADRS   OBJECT          MNEMONIC
 0100   340010          LXI     H,01000H

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 44  1C 00 6E FE 18 90 10 00 0000 0700 0103
     V' A' B' C' D' E' H' L' EA'
     00 FF 00 10 07 00 90 F0 014C

 ADRS   OBJECT          MNEMONIC
 0103   6091            XRA     A,A

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 40  1C 00 6E FE 18 90 10 00 0000 0700 0105
     V' A' B' C' D' E' H' L' EA'
     00 FF 00 10 07 00 90 F0 014C

 ONE STEP EMULATION STANDBY ←――――スペース入力

 ADRS   OBJECT          MNEMONIC
 0105   4DD2            MOV     MA,A

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 40  1C 00 6E FE 18 90 10 00 0000 0700 0107
     V' A' B' C' D' E' H' L' EA'
     00 FF 00 10 07 00 90 F0 014C ←――――― '⤶' 入力


*RUN T 100,2$D$R ⤶

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 44  1C 00 6E FE 18 90 10 00 0000 0700 0103
     V' A' B' C' D' E' H' L' EA'
     00 FF 00 10 07 00 90 F0 014C

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 40  1C 00 6E FE 18 90 10 00 0000 0700 0105
     V' A' B' C' D' E' H' L' EA'
     00 FF 00 10 07 00 90 F0 014C

 ONE STEP EMULATION STANDBY ←――――スペース入力

 ADRS   OBJECT          MNEMONIC
 0105   4DD2            MOV     MA,A

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 40  1C 00 6E FE 18 90 10 00 0000 0700 0107
     V' A' B' C' D' E' H' L' EA'
     00 FF 00 10 07 00 90 F0 014C ←――――― '⤶' 入力

*
```

保守／廃止

３．９．３　ＲＵＮ　Ｂ（ブレーク付きリアルタイム実行）

Ｒ Ｕ Ｎ␣Ｂ␣ａ ｄ ｄ ｒ ⏎

このコマンドを入力しますと，ＩＥ－７８Ｃ１１はユーザ・システムの電源をチェックした後‘ＥＭＵＬＡＴＩＯＮ　ＳＴＡＲＴ　ＡＴ　ａｄｄｒ’と出力して，入力したａｄｄｒからエミュレーションをスタートします．（ａｄｄｒを省略した時は現在のＰＣから，エミュレーションをスタートします．）

強制的にブレークさせる時は，ＥＳＣキーを入力します．ＣＴＲＬ－Ｃを入力すると，エミュレーションＣＰＵはエミュレーションを実行したまま‘＊’に戻ります．ＥＳＣキーを入力するとアドレス，命令コード，ニーモニック，レジスタの内容を表示し，入力待ちになります．ここでスペース入力をすると，１ステップ実行を行ない（１ステップ実行した時はトレース表示コマンドを実行してもトレース・データは表示されません．）‘⏎’を入力すると‘＊’に戻ります．

ＲＵＮ　ＮコマンドとＲＵＮ　Ｂコマンドとの違いは以下のようになります．ＲＵＮ　Ｎはブレーク条件に関係なくエミュレーションを実行するだけですが，ＲＵＮ　Ｂはあらかじめセットされたブレーク条件がエミュレーション中に一致するとブレークします．そして，その時のブレーク原因を表示します．

例）

```
*BRA A=XX ⏎
*BRO BRA ⏎
*BRM BRO,BRA ⏎
*RUN B 0 ⏎
 USER-SYSTEM VDD-ON

 JUST MOMENT
 EMULATION START AT 0000
```

```
NORMAL BREAK
TERMINATED

PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  08 59 18 18 26 25 00 00 0000 0603 0001
    V' A' B' C' D' E' H' L' EA'
    00 FF 20 30 00 04 06 FD 014C

ONE STEP EMULATION STANDBY ←

ADRS    OBJECT          MNEMONIC
0001    00              NOP

PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  08 59 18 18 26 25 00 00 0000 0603 0002
    V' A' B' C' D' E' H' L' EA'
    00 FF 20 30 00 04 06 FD 014C ←――― スペース入力による 1 ステップ動作

ADRS    OBJECT          MNEMONIC
0002    00              NOP

PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  08 59 18 18 26 25 00 00 0000 0603 0003
    V' A' B' C' D' E' H' L' EA'
    00 FF 20 30 00 04 06 FD 014C ←

ADRS    OBJECT          MNEMONIC
0003    00              NOP

PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  08 59 18 18 26 25 00 00 0000 0603 0004
    V' A' B' C' D' E' H' L' EA'
    00 FF 20 30 00 04 06 FD 014C )  ←――― 1ステップ 動作終了のために ')' を入力
*
```

直後にリターン・キーを入力すると，プロンプトに戻ります．スペース・キーを入力すると，1ステップ動作となります．ただし，今までトレースされたデータは消去されます．

保守／廃止

３．９．４　ＲＵＮ　Ｓ（リアルタイム・ステップ実行）

ＲＵＮ␣Ｓ〔␣ａｄｄｒ〕〔，ｓｔｅｐ－Ｎｏ.〕↵

コマンド入力待ち，‘＊’の後に‘ＲＵＮ　Ｓ　ａｄｄｒ，ｓｔｅｐ－Ｎｏ．↵’と入力してください．ＩＥ－７８Ｃ１１は‘ＥＭＵＬＡＴＩＯＮ　ＳＴＡＲＴ　ＡＴ　ａｄｄｒ’と出力して，入力したａｄｄｒからリアルタイムでｓｔｅｐ数だけ命令を実行します．（ａｄｄｒを省略した時は現在のＰＣから，ｓｔｅｐ数だけ実行を行ない，ｓｔｅｐ数を省略した時はａｄｄｒから１ステップ・エミュレーションに入ります．）　Ｓｔｅｐ－Ｎｏ．は最大２５５までです．

入力例）

```
＊ＲＵＮ　Ｓ　０，６０↵　　①
＊ＲＵＮ　Ｓ　，５０↵　　②
＊ＲＵＮ　Ｓ　１０００↵　　③
＊ＲＵＮ　Ｓ↵　　④
```

① ０Ｈ番地から６０Ｈステップをリアルタイムで実行し，以後１ステップ動作となります．

② 現在のプログラム・カウンタの値から５０Ｈステップをリアルタイムで実行し，以後１ステップ動作となります．

③ １０００番地よりステップ動作となります．

④ 現在のプログラム・カウンタの値からステップ動作となります．

例）

```
*RUN S 0,60 ↵

 USER-SYSTEM VDD-ON

 JUST MOMENT
 EMULATION START AT 0000

 STEP BREAK
 TERMINATED

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 00  F6 1F D2 6F 01 20 A6 39 4CFD 0000 0006
     V' A' B' C' D' E' H' L' EA'
     00 00 00 00 8E 27 8E 27 0000
```

```
 ONE STEP EMULATION STANDBY ←――――スペース入力 1ステップ動作

 ADRS      OBJECT                MNEMONIC
 0006      41                    INR     A

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 10  F6 20 D2 6F 01 20 A6 39 4CFD 0000 0007
     V' A' B' C' D' E' H' L' EA'
     00 00 00 00 8E 27 8E 27 0000 ←――――― ‘⤶’ 入力 1ステップ動作終了

*RUN S ,50⤶

 USER-SYSTEM VDD-ON

 JUST MOMENT
 EMULATION START AT 0007

 STEP BREAK
 TERMINATED

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 00  F6 3A D2 6F 01 3B A6 39 4CFD 0000 0006
     V' A' B' C' D' E' H' L' EA'
     00 00 00 00 8E 27 8E 27 0000

 ONE STEP EMULATION STANDBY ←――――スペース入力

 ADRS      OBJECT                MNEMONIC
 0006      41                    INR     A

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 00  F6 3B D2 6F 01 3B A6 39 4CFD 0000 0007
     V' A' B' C' D' E' H' L' EA'
     00 00 00 00 8E 27 8E 27 0000 ←――――― ‘⤶’ 入力

*RUN S 1000 ⤶

 ONE STEP EMULATION START

 ADRS      OBJECT                MNEMONIC
 1000      00                    NOP

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 00  F6 3B D2 6F 01 3B A6 39 4CFD 0000 1001
     V' A' B' C' D' E' H' L' EA'
     00 00 00 00 8E 27 8E 27 0000 ←――――――――スペース入力

 ADRS      OBJECT                MNEMONIC
 1001      00                    NOP

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 00  F6 3B D2 6F 01 3B A6 39 4CFD 0000 1002
     V' A' B' C' D' E' H' L' EA'
     00 00 00 00 8E 27 8E 27 0000 ←――――― ‘⤶’ 入力


*
```

```
*RUN S↵

 ONE STEP EMULATION START

 ADRS      OBJECT                MNEMONIC
 1002      00                    NOP

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 00  F6 3B D2 6F 01 3B A6 39 4CFD 0000 1003
     V' A' B' C' D' E' H' L' EA'
     00 00 00 00 8E 27 8E 27 0000   <-------- スペース入力による1ステップ動作

 ADRS      OBJECT                MNEMONIC
 1003      00                    NOP

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 00  F6 3B D2 6F 01 3B A6 39 4CFD 0000 1004
     V' A' B' C' D' E' H' L' EA'
     00 00 00 00 8E 27 8E 27 0000   <--------

 ADRS      OBJECT                MNEMONIC
 1004      00                    NOP

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 00  F6 3B D2 6F 01 3B A6 39 4CFD 0000 1005
     V' A' B' C' D' E' H' L' EA'
     00 00 00 00 8E 27 8E 27 0000   <-------- 1ステップ 動作終了のための'↵'入力

*
```

３．９．５　ＲＵＮ　Ｔ（表示付き１ステップ実行）

RUN␣T〔␣addr〕〔,　{step−No. / ※}〕〔$　{D / E}〕〔$R〕↵

※＝register−name {＝ / ＞ / ＜ / ＝＜ / ＞＝ / ＜＞ / ＞＜ / ＝＞ / ＜＝} value

入力例）

```
＊RUN　T　１００，１０$E$R↵　　①
＊RUN　T　，Ｌ＝０↵　　②
＊RUN　T↵　　③
```

① １００番地から１ステップ・エミュレーションをアドレス，命令コード，ニーモニック，レジスタの内容を表示させながら１０Ｈだけステップ実行を行ないます．以後１ステップ動作となります．

② 現在のプログラム・カウンタの値から，Ｌレジスタの内容が‘０’になるまで表示なしで実行し，以後１ステップ動作となります．

③ 現在のプログラム・カウンタの内容がスタート・アドレスに指定されて，１ステップ動作となります．

例）

```
*RUN T 0,4$R↵

ADRS      OBJECT              MNEMONIC
0000      240001              LXI     D,00100H

PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  F6 01 D2 6F 01 00 A6 39 4CFD 0000 0003
    V' A' B' C' D' E' H' L' EA'
    00 00 00 00 8E 27 8E 27 0000

ADRS      OBJECT              MNEMONIC
0003      6900                MVI     A,00000H
```


```
PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  F6 00 D2 6F 01 00 A6 39 4CFD 0000 0005
    V' A' B' C' D' E' H' L' EA'
    00 00 00 00 8E 27 8E 27 0000
ADRS     OBJECT               MNEMONIC
0005     3C                   STAX    D+

PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  F6 00 D2 6F 01 01 A6 39 4CFD 0000 0006
    V' A' B' C' D' E' H' L' EA'
    00 00 00 00 8E 27 8E 27 0000

ADRS     OBJECT               MNEMONIC
0006     41                   INR     A

PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  F6 01 D2 6F 01 01 A6 39 4CFD 0000 0007
    V' A' B' C' D' E' H' L' EA'
    00 00 00 00 8E 27 8E 27 0000

ONE STEP EMULATION STANDBY ←――――スペース入力 1ステップ動作

ADRS     OBJECT               MNEMONIC
0007     FD                   JR      00005H

PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  F6 01 D2 6F 01 01 A6 39 4CFD 0000 0005
    V' A' B' C' D' E' H' L' EA'
    00 00 00 00 8E 27 8E 27 0000     ←―― ‘⤶’ 入力 1ステップ動作終了

*RUN T 0,PC=6$R ⤶

ADRS     OBJECT               MNEMONIC
0000     240001               LXI     D,00100H

PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  F6 01 D2 6F 01 00 A6 39 4CFD 0000 0003
    V' A' B' C' D' E' H' L' EA'
    00 00 00 00 8E 27 8E 27 0000

ADRS     OBJECT               MNEMONIC
0003     6900                 MVI     A,00000H

PSW V  A  B  C  D  E  H  L  EA   SP   PC
08  F6 00 D2 6F 01 00 A6 39 4CFD 0000 0005
    V' A' B' C' D' E' H' L' EA'
    00 00 00 00 8E 27 8E 27 0000

ADRS     OBJECT               MNEMONIC
0005     3C                   STAX    D+

PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  F6 00 D2 6F 01 01 A6 39 4CFD 0000 0006
    V' A' B' C' D' E' H' L' EA'
    00 00 00 00 8E 27 8E 27 0000

ONE STEP EMULATION STANDBY ←――――スペース入力
```


```
 ADRS      OBJECT                MNEMONIC
 0006      41                    INR      A
 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 00  F6 10 D2 6F 01 01 A6 39 4CFD 0000 0007
     V' A' B' C' D' E' H' L' EA'
     00 00 00 00 8E 27 8E 27 0000   <------- ')' 入力

*REG C A)

 A 01=0  )

*RUN T 0,A>2 )

 ADRS      OBJECT                MNEMONIC
 0000      240001                LXI      D,00100H

 ADRS      OBJECT                MNEMONIC
 0003      6900                  MVI      A,00000H

 ADRS      OBJECT                MNEMONIC
 0005      3C                    STAX     D+

 ADRS      OBJECT                MNEMONIC
 0006      41                    INR      A

 ADRS      OBJECT                MNEMONIC
 0007      FD                    JR       00005H

 ADRS      OBJECT                MNEMONIC
 0005      3C                    STAX     D+

 ADRS      OBJECT                MNEMONIC
 0006      41                    INR      A

 ADRS      OBJECT                MNEMONIC
 0007      FD                    JR       00005H

 ADRS      OBJECT                MNEMONIC
 0005      3C                    STAX     D+

 ADRS      OBJECT                MNEMONIC
 0006      41                    INR      A

 ONE STEP EMULATION STANDBY <-------スペース入力

 ADRS      OBJECT                MNEMONIC
 0007      FD                    JR       00005H

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 00  F6 03 D2 6F 01 03 A6 39 4CFD 0000 0005
     V' A' B' C' D' E' H' L' EA'
     00 00 00 00 8E 27 8E 27 0000   <------- ')' 入力
```


```
*REG↵

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 00  F6 03 D2 6F 01 03 A6 39 4CFD 0000 0005
     V' A' B' C' D' E' H' L' EA'
     00 00 00 00 8E 27 8E 27 0000

*
```

３．９．６　１ステップ動作

スペース・キーを入力します．‘↵’キーを入力することにより，１ステップ動作を終了しプロンプトへ戻ります．

```
ONE STEP EMULATION START

ADRS      OBJECT                MNEMONIC
EB45      00                    NOP

PSW V  A  B  C  D  E  H  L  EA   SP   PC
00  08 59 18 18 26 25 00 00 0000 0603 EB46
    V' A' B' C' D' E' H' L' EA'
    00 FF 20 30 00 04 06 FD 014C   ←――― ‘↵’入力後,ｺﾏﾝﾄﾞ 入力待ちに
                                        戻ります.
*
```

保守／廃止

３．１０　ＢＲ？（ブレーク）コマンド

表現形式：

＊ＢＲＡ〔␣Ａ＝ａｄｄｒｓ〕〔␣Ｖ＝ｖａｌｕｅｓ〕
〔␣Ｃ＝ｃｏｎｄ〕〔␣Ｌ＝ｌｏｏｐ〕↵

＊ＢＲＤ〔␣ｄａｔａ〕↵

＊ＢＲＥ〔␣ｃｏｕｎｔ〕↵

＊ＢＲＴ〔␣ｔｉｍｅ〕↵

＊ＢＲ{0 | … | 3}〔␣ＢＲＡ，ＢＲＤ……〕↵

＊ＢＲＭ〔␣ＢＲ０，ＢＲ１……〕↵

機　能：　各種ブレーク・レジスタの内容を表示または設定します．

メイン・コマンド：	ＢＲ？	（？＝Ａ，Ｄ，Ｅ，Ｔ，Ｍ，０～３）
サブ・コマンド	：	なし	
オペランド	：	ａｄｄｒｓ	マスクありアドレス入力
	：	ｃｏｎｄ	ブレーク・コンディション（複数設定可能）
	：	ｖａｌｕｅｓ	マスクありブレーク・データ条件（１バイト）
	：	ｌｏｏｐ	ループ回数（１～２５５回）
	：	ａｄｄｒ	式処理可能なアドレス入力
	：	ｃｏｕｎｔ	式処理可能な２バイト入力
	：	ｄａｔａ	１バイト入力
	：	ｔｉｍｅ	時間単位（ｍｓｅｃ）のセット
	：	ＢＲ０～ＢＲ３	ロジカル・ブレーク・レジスタ
	：	ＢＲＡ～ＢＲＴ	フィジカル・ブレーク・レジスタ
	：	ＢＲＭ	ブレーク・モード・レジスタ
オプション	：	なし	

保守／廃止

３．１０．１　ブレーク・コマンドの概要

ブレーク・コマンドは以下の３種類に大別されます．

（１）フィジカル・ブレーク・レジスタ（Physical break register）

ハードウェア上に直接セットするデータを保存するレジスタです．

ＢＲＡ，ＢＲＤ，ＢＲＥ，ＢＲＴ

（２）ロジカル・ブレーク・レジスタ（Logical break register）

（１）で示したレジスタの組合せをデータとして保存するレジスタです．

ＢＲ０，ＢＲ１，ＢＲ２，ＢＲ３

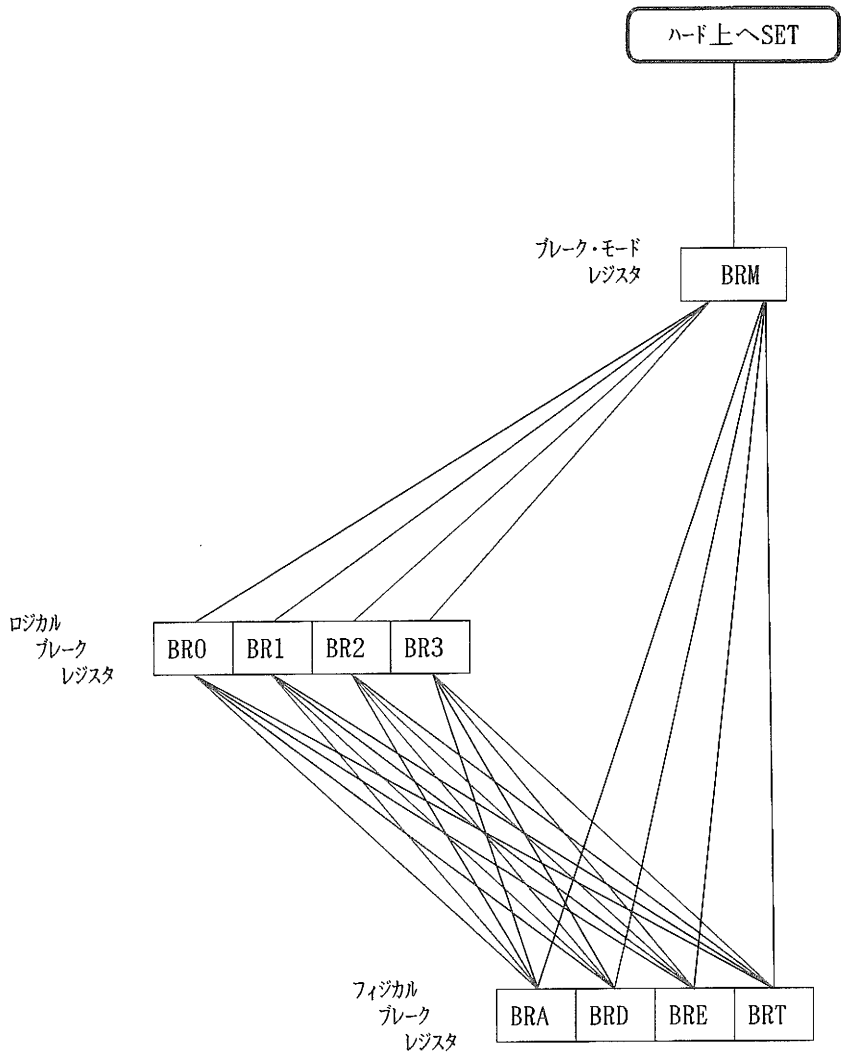
（３）ブレーク・モード・レジスタ（Break mode register）

ハードウェア上に（１），（２）で示されるブレーク・データをセットするコマンドです．このコマンドを実行するまでは，（１），（２）のデータの変更を行なっても，ハードウェア上にはセットできません．

ＢＲＭ

（注）以上のブレーク・レジスタはＰＯＷＥＲ　ＯＮ時には，何も設定されていない状態となっています．

ブレーク・レジスタ・コマンドの概略図



保守／廃止

３．１０．２　フィジカル・ブレーク・レジスタ

フィジカル・ブレーク・レジスタには以下の種類があります．

ＢＲＡ，ＢＲＤ，ＢＲＥ，ＢＲＴ

これは，ハードウェアにセットすべきブレーク・データを保存するためのレジスタで，このレジスタにデータをセットして‘ＲＵＮ　Ｂ’を実行してもブレークしません．

（１）ＢＲＡコマンド

ＢＲＡ〔␣Ａ＝ａｄｄｒｓ〕〔␣Ｖ＝ｖａｌｕｅｓ〕
〔␣Ｃ＝ｃｏｎｄ〕〔␣Ｌ＝ｌｏｏｐ〕

Ａ　；　ブレーク・アドレスの設定

Ｖ　；　ブレーク・データの設定

Ｃ　；　ブレーク・コンディションの設定

Ｌ　；　ループ回数の設定

ａｄｄｒｓ　；　アドレス部

ａｄｄｒまたはｐａｒｔｉｔｉｏｎをスペースで区切り最大５個まで設定できます．

ｖａｌｕｅｓ　；　データ部

８ビットまでのデータでＸ表現が可能です．（ただし，１０進のＸ表現は禁止）

ｃｏｎｄ　；　コンディション部

次に示すコンディション指定が可能です．省略された場合は条件なしとなります．

ＯＰ　；　オペコード・フェッチ（Ｍ１リード）

ＲＤ　；　メモリ・リード

ＷＲ　；　メモリ・ライト

ＭＲ　；　メモリ・リクエスト

ＮＣ　；　条件なし

ｌｏｏｐ　　；　ループ指定部

アドレス部，データ部，コンディション部によって設定された条件に対してのループ（通過回数）の設定が行なえます．ループ回数は１～２５５回です．

入力例）

```
＊ＢＲＡ　Ａ＝１０〉                                          ①
＊ＢＲＡ〉                                                    ②
＊ＢＲＡ　Ａ＝０，０ＦＨ　３０Ｈ，３ＦＨ　５０Ｈ　６０Ｈ〉      ③
＊ＢＲＡ　Ｖ＝５５Ｔ〉                                        ④
＊ＢＲＡ　Ｃ＝ＷＲ　Ｌ＝３〉                                  ⑤
＊ＢＲＡ　Ａ＝２０Ｈ　Ｖ＝０　Ｌ＝５Ｔ〉                       ⑥
＊ＢＲＡ　Ａ＝１ＸＸＨ　Ｖ＝１Ｘ０Ｘ１Ｙ　Ｃ＝ＲＤ　Ｌ＝２Ｑ〉  ⑦
```

① アドレス１０番地をアクセスしたらブレークします．

サフィックスの指定を省略すると現在のサフィックスとしてアドレス値を評価します．従って，現在のサフィックスが２進なら２Ｈ番地，８進なら８Ｈ番地，１０進ならＡＨ番地，１６進なら１０Ｈ番地となります．

② ＢＲＡに登録されているブレーク条件を表示します．

③ アドレスの０～０ＦＨ，３０～３ＦＨ，５０Ｈ，６０Ｈのいずれかをアクセスしたらブレークします．

④ データ５５Ｔをアクセスしたらブレークします．

⑤ データを３回書込むとブレークします．

⑥ アドレス２０Ｈ番地にデータ０Ｈを５回アクセスしたらブレークします．

⑦ アドレス１００～１ＦＦＨ番地の範囲でデータ１１Ｈ，１３Ｈ，１９Ｈ，１ＢＨのいずれかを２回リードしたらブレークします．

例）

```
*BRA 〉
 --    ←――――――未設定
*BRA A=10 〉      ――条件ｾｯﾄ
*BRA 〉
 A=10H   ←――――設定時のｻﾌｨｯｸｽを表示
*BRA A=1XXH V=1X0XY C=RD L=2Q 〉
*BRA 〉                    ←――――条件ｾｯﾄ
 A=1XXH V=1X0X1Y C=RD L=2Q 〉
*
```

保守／廃止

（2）ＢＲＤコマンド

ＢＲＤ␣ｄａｔａ⏎

ＢＲＤコマンドは，外部センス・クリップ８本のデータでブレークさせるためのオプション・ブレーク・コマンドです．入力可能なｄａｔａは１バイトです．

入力例）

```
＊ＢＲＤ　０１０１１０１０Ｙ⏎　①
＊ＢＲＤ⏎　②
```

① 外部センス・クリップからの入力データが０１０１１０１０Ｙの時をブレーク条件として設定します．

② コマンド入力待ち‘＊’で‘ＢＲＤ⏎’と入力すると，改行復帰してＢＲＤのセット・データを表示して‘＊’に戻ります．何もセットされていない時は‘－－’を表示します．

例）

```
*BRD⏎
 --
*BRD 5AH⏎
*BRD⏎
 5AH
*
```

（3）BREコマンド

BRE␣count↵

BREコマンドは，オペコード・フェッチ（M1リード）の回数（＝count）を指定できます．

入力例）

```
＊BRE  1000H↵          ①
＊BRE↵                 ②
```

① オペコード・フェッチを1000H回カウントした時をブレーク条件として設定します．countは2バイトで3～0FFFFHの範囲です．

② コマンド入力待ち‘＊’の後‘BRE↵’と入力すると改行復帰して現在セットされているデータを表示して‘＊’に戻ります．何もセットされていない時は‘−−’が表示され，改行復帰後‘＊’に戻ります．

例）

```
*BRE↵
 --
*BRE 1000 ↵
*BRE↵
 1000H
*
```

（4）ＢＲＴコマンド

ＢＲＴ␣ｔｉｍｅ⤶

ブレーク条件としてタイマを使用する時に，エミュレーション開始からブレークするまでの時間を設定します．設定可能な時間は，１～０ＦＦＦＦＨｍｓです．

入力例）

```
＊ＢＲＴ　１３００Ｈ⤶          ①
＊ＢＲＴ⤶                      ②
```

① ＢＲＴに時間１３００Ｈｍｓを設定します．ｔｉｍｅ（時間）は２バイト・データです．

② コマンド入力待ち‘＊’の後，‘ＢＲＴ⤶’と入力すると改行復帰して，現在セットされているデータを表示し‘＊’に戻ります．何もセットされていない時は‘－－’を表示して改行復帰後‘＊’に戻ります．

例）

```
*BRT⤶
 --
*BRT 1300 ⤶
*BRT⤶
 1300H
*
```

保守／廃止

３．１０．３　ロジカル・ブレーク・レジスタ

ＢＲ { 0 / 3 } 〔␣ＢＲＡ，ＢＲＤ……〕 ↵

（‘ＢＲＡ，ＢＲＤ……’の部分は，フィジカル・ブレーク・レジスタＢＲＡ，ＢＲＤ，ＢＲＥ，ＢＲＴから１個あるいは複数個選択できることを意味します.）

ロジカル・ブレーク・レジスタにはＢＲ０～ＢＲ３まであり，それぞれ同等の機能を持っています．その機能は，あらかじめブレーク時にセットしたいフィジカル・ブレーク・レジスタをロジカル・ブレーク・レジスタにセットしておけばＢＲＭコマンドでブレーク条件をセットする時にそのロジカル・ブレーク・レジスタを１個セットすればよいことになります．

入力例）

```
＊ＢＲ０　ＢＲＡ，ＢＲＤ，ＢＲＥ，ＢＲＴ↵　　①
＊ＢＲ０↵　　②
```

① コマンド入力待ち‘＊’の後，‘ＢＲ０　ＢＲＡ↵’と入力することにより，ロジカル・ブレーク・レジスタに指定されたフィジカル・ブレーク・レジスタがセットされます．フィジカル・ブレーク・レジスタを複数入力する時は，その間を‘，’で区切ってください．（ＢＲ１～ＢＲ３も同様）ブレーク・レジスタはいずれかの条件が満足したらブレークします．

② コマンド入力待ち‘＊’の後，‘ＢＲ０↵’と入力すると改行復帰して現在セットされているフィジカル・レジスタ名を表示して‘＊’に戻ります．何もセットされていない時は‘－－’を表示して，改行復帰後‘＊’に戻ります．

例）

```
*BR0↵
 --
*BR0 BRA,BRD,BRE,BRT↵
*BR0↵
 BRA,BRD,BRE,BRT
*
```

保守／廃止

３．１０．４　ブレーク・モード・レジスタ

ＢＲＭ〔␣ＢＲＡ，ＢＲＤ……〕↵

（‘ＢＲＡ，ＢＲＤ……’の部分は，ＢＲＡ，ＢＲＤ，ＢＲＥ，ＢＲＴ，ＢＲ０，ＢＲ１，ＢＲ２，ＢＲ３から１個あるいは複数個選択できることを意味します.）

ＢＲＭはブレーク付きエミュレーション（ＲＵＮ　Ｂ）の実行を行なう前にフィジカル・ブレーク・レジスタ，ロジカル・ブレーク・レジスタにセットしたブレーク条件をハードウェアにセットするためのコマンドです.

入力例）

＊ＢＲＭ　ＢＲＡ，ＢＲ０，ＢＲ１↵　｝
＊ＢＲＭ　ＢＲ０，ＢＲ１，ＢＲ２，ＢＲ３↵　｝　①

＊ＢＲＭ↵　②

① コマンド入力待ち‘＊’の後に‘ＢＲＭ　ＢＲＡ，ＢＲ０，ＢＲ１↵’のように入力してください.‘ＢＲＭ↵’の後に入力するのは，フィジカル・ブレーク・レジスタ名，ロジカル・ブレーク・レジスタ名を入力してください.複数の時は間に‘，’を入力して区切ってください.入力する順序にフィジカル・ブレーク・レジスタとロジカル・ブレーク・レジスタの別なく，どのレジスタ名からでも入力できます.

② コマンド入力待ち‘＊’の後に‘ＢＲＭ↵’と入力すると改行復帰して，現在セットされているロジカル・ブレーク・レジスタ名，フィジカル・ブレーク・レジスタ名を表示します.

例)

```
*BRM ↵
 --
*BRM BRA,BR0,BR1 ↵
*BRM ↵
 BRA,BR0,BR1
*BRM BRA,BRD,BR0,BR1,BR2,BR3,BRT ↵
*BRM ↵
 BRA,BRD,BR0,BR1,BR2,BR3,BRT
*BRM BR0,BR1,BR2,BR3 ↵
*BRM ↵
 BR0,BR1,BR2,BR3
*

*BRA A=35 V=69 C=NC L=1  ↵

*BRT 300 ↵

*BRM BRA,BRT ↵

*RUN B 0 ↵

 USER-SYSTEM VDD-ON

JUST MOMENT
EMULATION START AT 0000

NORMAL BREAK
 TERMINATED

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 48  08 AA 02 03 04 05 20 00 0000 3000 0037
     V' A' B' C' D' E' H' L' EA'
     80 10 20 30 40 50 60 70 014C

 ONE STEP EMULATION START  ↵

*
```

保守／廃止

```
*TRD -24⤶

ADDR   OBJECT            MNEMONIC                PA PB
002E   41                INR      A              00 00
002F   FD                JR       00035H         00 00
002D   3D                STAX     H+             00 00
002E   41                INR      A              00 00
002F   FD                JR       00035H         00 00
002D   3D                STAX     H+             00 00
002E   41                INR      A              00 00
002F   FD                JR       00035H         00 00
002D   3D                STAX     H+             00 00
002E   41                INR      A              00 00
002F   FD                JR       00035H         00 00
002D   3D                STAX     H+             00 00
002E   41                INR      A              00 00
002F   FD                JR       00035H         00 00
002D   3D                STAX     H+             00 00
002E   41                INR      A              00 00
002F   FD                JR       00035H         00 00
002D   3D                STAX     H+             00 00
002E   41                INR      A              00 00
002F   FD                JR       00035H         00 00
0030   747E20            EQI      H,00020H       00 00
0033   F9                JR       00031H         00 00
0034   C0                JR       00033H         00 00
0035   69AA              MVI      A,000AAH       00 00
NEWEST

*
```

保守／廃止

3．11　TR？（トレース）コマンド

表現形式：

＊TRC [␣{M / I}] ↵

＊TRD [␣{ALL / line－No. / －line－No.}] ↵

＊TRM [␣{NON / ALL / TRX}] ↵

＊TRP [␣{O / N / line－No. / －line－No.}] ↵

＊TRX 〔␣A＝addrx〕〔␣V＝value〕〔␣C＝cond〕
〔␣PA＝values〕〔␣PB＝values〕↵

＊TRS [␣{PB / EX}] ↵

機　能：　エミュレーションCPUの実行内容をトレースするために必要な種々の設定を行ないます．

メイン・コマンド	：	TR？　（？＝C，D，M，P，X，S）	
サブ・コマンド	：	なし	
オペランド	：	line－No.	変更するポインタ数（ライン数）
	：	O　（OLD）	最初にトレースした位置
	：	N　（NEW）	最後にトレースした位置
	：	ALL	トレース結果をすべて表示
	：	M	マシン・サイクル・トレースの表示
	：	I	インストラクション・トレースの表示
	：	PB	ポートB
	：	EX	外部センス・クリップ
オプション	：	なし	

保守／廃止

３．１１．１　トレース・コマンドの概要

トレース・コマンドには，ＴＲＣ，ＴＲＤ，ＴＲＭ，ＴＲＰ，ＴＲＸ，ＴＲＳの六つのコマンドがあります．

ＴＲＣ　：　表示のトレース・サイクル指定

ＴＲＤ　：　トレース・データの表示

ＴＲＭ　：　トレース・モードの指定

ＴＲＰ　：　トレース・ポインタの変更

ＴＲＸ　：　各種のトレース条件を設定

ＴＲＳ　：　トレースの選択（ＰＢとＥＸを選ぶ）

トレースＲＡＭは，マシン・サイクルで，深さ１０２３レベルでエミュレーション実行結果をトレースします．

POWER ON時には，トレース・サイクル（ＴＲＣ）はマシン・サイクル（Ｍ），トレース・モード（ＴＲＭ）はノン・トレース（ＮＯＮ）となっております．

トレース・データがないときにＴＲＤコマンドを実行すると‘ＮＯＮ　ＴＲＡＣＥ　ＤＡＴＡ’と表示されます．

保守／廃止

３．１１．２　ＴＲＣコマンド（トレース・サイクル）

ＴＲＣ [␣ {M / I}] ↵

ＴＲＣコマンドは，ＴＲＤ（トレース・ダンプ）時にマシン・サイクル表示にするか，インストラクション表示にするかの切換えコマンドです．

Ｍ　：　マシン・サイクル

Ｉ　：　インストラクション・サイクル

現在の状態を知りたいときには‘ＴＲＣ↵’と入力すれば改行復帰後，ＭまたはＩが表示されます．

例）

```
*TRD ALL ↵          ←――――――――マシン・サイクルでの表示

 OLDEST
 ADRS  DATA  CYC.   PA PB
 0100  34    OP     00 00
 0101  00    RD     00 00
 0102  10    RD     00 00
 0103  60    OP     00 00
 0104  91    RD     00 00
 0105  4D    OP     00 00
 0106  D2    RD     00 00
 0107  6B    OP     FF FF
 0108  FF    RD     FF FF
 0109  2D    OP     FF FF
 1000  11    RD     FF FF
 010A  4D    OP     FF FF
 010B  C0    RD     FF FF
 010C  53    OP     11 11
 010D  54    OP     11 11
 010E  09    RD     11 11
 010F  01    RD     11 11
 NEWEST

*TRC I ↵

*TRD ALL ↵          ←――――――――インストラクション・サイクルでの表示

 OLDEST
 ADRS   OBJECT          MNEMONIC                PA PB
 0100   340010          LXI     H,01000H        00 00
 0103   6091            XRA     A,A             00 00
 0105   4DD2            MOV     MA,A            00 00
 0107   6BFF            MVI     C,000FFH        FF FF
 0109   2D              LDAX    H+              FF FF
 010A   4DC0            MOV     PA,A            FF FF
 010C   53              DCR     C               11 11
 010D   540901          JMP     00109H          11 11
 NEWEST

*
```

保守／廃止

3．11．3　TRDコマンド（トレース・ダンプ）

```
        ┌   ┌ALL          ┐ ┐
TRD     │ ␣ │line−No.     │ │ ⤶
        └   └−line−No.    ┘ ┘
```

TRDコマンドは，リアルタイム実行（RUN　N／B／S）を行なった後に，その実行結果を表示するコマンドです．トレースする時の条件は，TRMコマンドの説明を参照してください．

コマンド入力待ち‘*’で，‘TRD　line−No．⤶’‘TRD　−line−No．⤶’と入力すると，改行復帰してヘッダを出力し，次の行から入力されたライン数だけトレース表示します．トレース・ポインタは，最後に表示したラインになります．

‘TRD⤶’と入力した時は，現在のポインタからトレースの終わりまで表示します．トレース・データすべてを表示する時は，‘TRD　ALL⤶’と入力してください．（ただし，TRMがNONとなっている時にリアルタイム実行した時と，1ステップ実行時には，データはトレースされないので‘TRD⤶’と入力すると‘NON　TRACE　DATA’と表示して‘*’に戻ります.）

トレース表示を中断させるには，CTRL−Cキー，ESCキー，のいずれかのキーを入力してください．中断したところが現在のトレース・ポインタとなります．

例）

```
*TRD ALL ↵

 OLDEST
 ADRS  DATA  CYC.  PA EX
 1011  12    RD    13 F6
 1012  13    RD    10 F7
 1013  14    RD    13 E6
 1014  21    RD    14 F5
 1019  12    RD    13 F2
 101A  13    RD    10 F3
 101B  14    RD    13 E0
 101C  21    RD    14 F1
 NEWEST

*TRD -3 ↵

 ADRS  DATA  CYC.  PA EX
 101A  13    RD    10 F3
 101B  14    RD    13 E0
 101C  21    RD    14 F1
 NEWEST

*TRD ↵

 ADRS  DATA  CYC.  PA EX
 101C  21    RD    14 F1
 NEWEST

*TRD 3 ↵

 ADRS  DATA  CYC.  PA EX
 101C  21    RD    14 F1
 NEWEST

*TRD -1 ↵

 ADRS  DATA  CYC.  PA EX
 101C  21    RD    14 F1
 NEWEST

*
```

保守／廃止

3．11．4　TRPコマンド（トレース・ポインタ）

TRP [␣ {O / N / line−No. / −line−No.}] ↵

TRPコマンドは，現在のトレース・ポインタを移動させるコマンドです．

リアルタイム実行（RUN　N，RUN　BまたはRUN　S）をした直後のポインタはトレース・データの最も新しいところを示しています．

コマンド入力待ち‘*’で以下の入力が変更できます．

1）‘TRP　O↵’

トレース・データの一番古い所（Old）へポインタを移動します．

2）‘TRP　N↵’

トレース・データの一番新しい所（New）へポインタを移動します．

3）‘TRP　line−No.↵’，‘TRP　−line−No.↵’

現在のポインタから数値で指定された数だけ移動します．

ただし，±1は現在のポインタです．

プラス方向（N方向）への移動の場合は，‘+’の入力はしません．

4）‘TRP↵’

トレースしたデータ数を表示します．

例）

```
*TRD ALL ↵

 OLDEST
 ADRS  DATA  CYC.   PA PB
 0100  34    OP     00 00
 0101  00    RD     00 00
 0102  10    RD     00 00
 0103  60    OP     00 00
 0104  91    RD     00 00
 0105  4D    OP     00 00
 0106  D2    RD     00 00
 0107  6B    OP     00 00
 0108  FF    RD     00 00
 0109  2D    OP     00 00
 1000  00    RD     00 00
 NEWEST

*TRP N ↵

*TRD ↵

 ADRS  DATA  CYC.   PA PB
 1000  00    RD     00 00
 NEWEST

*TRP 0 ↵

*TRD ↵

 OLDEST
 ADRS  DATA  CYC.   PA PB
 0100  34    OP     00 00
 0101  00    RD     00 00
 0102  10    RD     00 00
 0103  60    OP     00 00
 0104  91    RD     00 00
 0105  4D    OP     00 00
 0106  D2    RD     00 00
 0107  6B    OP     00 00
 0108  FF    RD     00 00
 0109  2D    OP     00 00
 1000  00    RD     00 00
 NEWEST

*TRP ↵

 000B

*TRP -5 ↵

*TRD ↵

 ADRS  DATA  CYC.   PA PB
 0106  D2    RD     00 00
 0107  6B    OP     00 00
 0108  FF    RD     00 00
 0109  2D    OP     00 00
 1000  00    RD     00 00
 NEWEST

*
```

3．11．5　TRMコマンド

TRM [␣ {NON / ALL / TRX}] ↵

TRMコマンドはトレース条件を指定するコマンドです．

トレース条件には以下の種類があります．

NON　：　トレースしません．

ALL　：　すべてトレースします．

TRX　：　TRXコマンドで指定したトレース条件に従ってトレースします．

入力例）

```
＊TRM　NON↵
＊TRM　ALL↵
＊TRM　TRX↵
＊TRM↵
```

コマンド入力待ち‘＊’で‘TRM　NON↵’と入力してください．そのモードにセットされ‘＊’に戻ります．（ALL，TRXも同様です．）

現在の状態を知りたい時は‘TRM↵’と入力すると改行復帰し，設定されたトレース条件を表示して‘＊’に戻ります．

保守／廃止

３．１１．６　ＴＲＸコマンド

ＴＲＸ〔␣Ａ＝ａｄｄｒｓ〕〔␣Ｖ＝ｖａｌｕｅｓ〕〔␣Ｃ＝ｃｏｎｄ〕
〔␣ＰＡ＝ｖａｌｕｅｓ〕〔␣ＰＢ＝ｖａｌｕｅｓ〕

Ａ　；　トレース・アドレスの設定

Ｖ　；　トレース・データの設定

Ｃ　；　トレース・コンディションの設定

ＰＡ　；　ポートＡのトレース設定

ＰＢ　；　ポートＢのトレース設定

ａｄｄｒｓ　：　アドレス部

ａｄｄｒまたはｐａｒｔｉｔｉｏｎをスペースで区切り最大５個まで設定できます．

ｖａｌｕｅｓ　：　データ部

８ビットまでのデータでＸ表現が可能です．（ただし，１０進のＸ表現は禁止）

ｃｏｎｄ　：　コンディション部

次に示すコンディション指定が可能です．省略された場合は条件なしとなります．

ＯＰ　：　オペコード・フェッチ（Ｍ１リード）

ＲＤ　：　メモリ・リード

ＷＲ　：　メモリ・ライト

ＮＣ　：　条件なし

入力例）

```
＊ＴＲＸ　Ａ＝１０↵                                          ①
＊ＴＲＸ↵                                                    ②
＊ＴＲＸ　Ａ＝０，０ＦＨ　３０Ｈ，３ＦＨ　５０Ｈ　６０Ｈ↵      ③
＊ＴＲＸ　Ｖ＝５５Ｔ↵                                        ④
＊ＴＲＸ　Ｃ＝ＷＲ↵                                          ⑤
＊ＴＲＸ　Ａ＝２０Ｈ　Ｖ＝０↵                                ⑥
＊ＴＲＸ　Ａ＝１ＸＸＨ　Ｖ＝１Ｘ０Ｘ１Ｙ　Ｃ＝ＲＤ↵          ⑦
＊ＴＲＸ　Ａ＝１０ＸＸＨ　ＰＡ＝１Ｘ０Ｘ１Ｙ↵                ⑧
```

① アドレスの１０番地をアクセスした時にトレースします．サフィックスの指定を省略すると現在のサフィックスとしてアドレス値を評価します．現在のサフィックスが２進なら２Ｈ番地，８進なら８Ｈ番地，１０進ならＡＨ番地，１６進なら１０Ｈ番地となります．

② ＴＲＸに登録されているトレース条件を表示します．

③ アドレスの０～０ＦＨ，３０～３ＦＨ，５０Ｈ，６０Ｈのいずれかをアクセスした時にトレースします．

④ データ５５Ｔをアクセスした時にトレースします．

⑤ データの書込みをトレースします．

⑥ アドレス２０Ｈ番地でデータ０Ｈをアクセスした時にトレースします．

⑦ アドレス１００～１ＦＦＨ番地の範囲でデータ１１Ｈ，１３Ｈ，１９Ｈ，１ＢＨのいずれかをリードした時にトレースします．

⑧ アドレス１０００Ｈ～１０ＦＦＨ番地の範囲で，ポートＡの値が，１０００１Ｙ，１００１１Ｙ，１１００１Ｙ，１１０１１Ｙの時にトレースします．

例)　以下にＴＲＸコマンドの使用例を示します.

```
*MEM D 10XX ↵

 1000  12 13 14 15 16 17 18 20  22 24 12 13 14 15 16 17
 1010  18 20 22 24 12 13 14 15  16 17 18 20 22 24 12 13
 1020  14 15 16 17 18 20 22 24  12 13 14 15 16 17 18 20
 1030  22 24 12 13 14 15 16 17  18 20 22 24 12 13 14 15
 1040  16 17 18 20 22 24 12 13  14 15 16 17 18 20 22 24
 1050  12 13 14 15 16 17 18 20  22 24 12 13 14 15 16 17
 1060  18 20 22 24 12 13 14 15  16 17 18 20 22 24 12 13
 1070  14 15 16 17 18 20 22 24  12 13 14 15 16 17 18 20
 1080  22 24 12 13 14 15 16 17  18 20 22 24 12 13 14 15
 1090  16 17 18 20 22 24 12 13  14 15 16 17 18 20 22 24
 10A0  12 13 14 15 16 17 18 20  22 24 12 13 14 15 16 17
 10B0  18 20 22 24 12 13 14 15  16 17 18 20 22 24 12 13
 10C0  14 15 16 17 18 20 22 24  12 13 14 15 16 17 18 20
 10D0  22 24 12 13 14 15 16 17  18 20 22 24 12 13 14 15
 10E0  16 17 18 20 22 24 12 13  14 15 16 17 18 20 22 24
 10F0  12 13 14 15 16 17 18 20  22 24 12 13 14 15 16 17

*DAS 100,120 ↵

 ADRS   OBJECT          MNEMONIC
 0100   340010          LXI     H,01000H
 0103   6091            XRA     A,A
 0105   ADD2            MOV     MA,A
 0107   6BFF            MVI     C,000FFH
 0109   2D              LDAX    H+
 010A   4DC0            MOV     PA,A
 010C   53              DCR     C
 010D   540901          JMP     00109H
 0110   340020          LXI     H,02000H
 0113   69FF            MVI     A,000FFH
 0115   4DD3            MOV     MB,A
 0117   6BFF            MVI     C,000FFH
 0119   4CC1            MOV     A,PB
 011B   3D              STAX    H+
 011C   53              DCR     C
 011D   541901          JMP     00119H
 0120   00              NOP

*BRA A=110 ↵

*BRM BRA ↵

*TRM TRX ↵

*TRX A=101XH C=RD PA=1XH ↵
```

←――――ポートＡの値が１０Ｈ～１ＦＨのときの１０１０Ｈ番地～１０１ＦＨ番地のデータリードをトレースするようにＴＲＸを設定します.


```
*RUN B 100 ↵

 USER-SYSTEM VDD-ON

 JUST MOMENT
 EMULATION START AT 0100

 NOPMAL BREAK
 TERMINATED

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 14  8E 17 16 FF 13 12 20 00 0000 0700 0113
     V' A' B' C' D' E' H' L' EA'
     00 FF FF FF 01 51 00 00 0151

 ONE STEP EMULATION STANDBY

*TRD ALL ↵          ←――――――トレース・データをすべて表示します.

 OLDEST
 ADRS  DATA  CYC.  PA PB
 1010  18    RD    17 17
 1011  20    RD    18 18
 1015  13    RD    12 12
 1016  14    RD    13 13
 1017  15    RD    14 14
 1018  16    RD    15 15
 1019  17    RD    16 16
 101A  18    RD    17 17
 101B  20    RD    18 18
 101F  13    RD    12 12
 NEWEST

*
```

保守／廃止

３．１１．７　ＴＲＳコマンド

ＴＲＳ〔␣｛ＰＢ／ＥＸ｝〕↵

ＰＢ　；　ポートＢ
ＥＸ　；　外部センス・データ

入力例）

```
＊ＴＲＳ　ＥＸ↵　　①
＊ＴＲＳ↵　　②
```

① 外部センス・クリップを接続した部分のトレースをします．

② 現在選択されているソース名を表示します．

例）

```
*DAS 100,10F ↵

 ADRS   OBJECT          MNEMONIC
 0100   340010          LXI     H,01000H
 0103   6091            XRA     A,A
 0105   4DD2            MOV     MA,A
 0107   6BFF            MVI     C,000FFH
 0109   2D              LDAX    H+
 010A   4DC0            MOV     PA,A
 010C   53              DCR     C
 010D   540901          JMP     00109H

*BRA A=110 ↵

*BRM BRA ↵

*TRX A=101XH C=RD PA=1XH ↵

*TRM TRX ↵

*TRS PB ↵          ←――――――ポートＢをトレースします.

*RUN B 100 ↵

 USER-SYSTEM VDD-ON

 JUST MOMENT
 EMULATION START AT 0100


 NORMAL BREAK
 TERMINATED

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 14  02 24 80 FF 80 80 20 00 0000 0700 0113
     V' A' B' C' D' E' H' L' EA'
     00 FF BF FF 00 3D 00 00 014C

 ONE STEP EMULATION STANDBY
```

```
*TRD ALL )

 OLDEST
 ADRS  DATA  CYC.   PA PB
 1011  12    RD     13 13
 1012  13    RD     10 10
 1013  14    RD     13 13
 1014  21    RD     14 14
 1019  12    RD     13 13
 101A  13    RD     10 10
 101B  14    RD     13 13
 101C  21    RD     14 14
 NEWEST

*TRS EX )          ←――――外部センス・クリップを接続した部分のトレースをします.

*RUN B 100 )

 USER-SYSTEM VDD-ON

 JUST MOMENT
 EMULATION START AT 0100

 NORMAL BREAK
 TERMINATED

 PSW V  A  B  C  D  E  H  L  EA   SP   PC
 14  02 00 80 FF 80 80 20 00 0000 0700 0113
     V' A' B' C' D' E' H' L' EA'
     00 FF BF FF 00 3D 00 00 014C

 ONE STEP EMULATION STANDBY

*TRD ALL )
 OLDEST

 ADRS  DATA  CYC.   PA EX
 1011  12    RD     13 F3
 1012  13    RD     10 F2
 1013  14    RD     13 F3
 1014  21    RD     14 F2
 1019  12    RD     13 F1
 101A  13    RD     10 F0
 101B  14    RD     13 F1
 101C  21    RD     14 E1
 NEWEST

*
```

保守／廃止

３．１２　ＣＬＫ（クロック）コマンド

表現形式：

＊ＣＬＫ〔␣{Ｉ／Ｕ}〕

機　能：　エミュレーションＣＰＵが使用するクロックの指定および表示を行ないます．

メイン・コマンド　：　ＣＬＫ

サブ・コマンド　：　Ｉ　　内部クロック　；ＩＥ－７８Ｃ１１内部のクロック（１２ＭＨｚ固定）を使用した場合

：　Ｕ　　外部クロック　；ターゲット・システムのクロックを使用する場合
ただし，４ＭＨｚから１５ＭＨｚの間で使用してください．

保守／廃止

３．１２．１　クロックの指定

ＣＬＫ␣{Ｕ／Ｉ}↵

エミュレーションＣＰＵの動作クロックを，ディバッガ内部のクロック（１２ＭＨｚ）またはターゲット・システムのクロック（４ＭＨｚ～１５ＭＨｚ）のどちらかに指定します．

入力例）

```
＊ＣＬＫ　Ｕ↵　　①
＊ＣＬＫ　Ｉ↵　　②
```

① エミュレーションＣＰＵの動作クロックをターゲット・システムのクロックに指定します．この時エミュレーション・プローブの設定を参考にエミュレーション・プローブのスイッチを設定してください．

② ＩＥ－７８Ｃ１１内部のクロック（１２ＭＨｚ）を指定．

注意）ＣＬＫコマンドでクロックを指定すると同時にエミュレーションＣＰＵのリセットが行なわれます．

３．１２．２　クロックの表示

ＣＬＫ↵

エミュレーションＣＰＵの動作クロックを表示します．

例）

```
*CLK ↵
 I
*CLK U ↵
*CLK ↵
 U
*CLK I ↵
*CLK ↵
 I
*
```

保守／廃止

３．１３　ＲＥＳ（リセット）コマンド

表現形式：

＊ＲＥＳ〔␣Ｈ〕↵

機　能：　エミュレーションＣＰＵのリセットまたはＩＥ－７８Ｃ１１ハードウェア全体のリセットを行ないます．

メイン・コマンド	：	ＲＥＳ
サブ・コマンド	：	Ｈ　　ＩＥ－７８Ｃ１１リセット
オペランド	：	なし
オプション	：	なし

保守／廃止

３．１３．１　エミュレーションＣＰＵのリセット

ＲＥＳ↵

エミュレーションＣＰＵのリセットが行なわれます．

例）

```
*RES↵
*
```

３．１３．２　ハード・リセット

ＲＥＳ　Ｈ↵

ＩＥ－７８Ｃ１１のリセットが行なわれます．

これは，コントロール・ボード上のリセット・スイッチを押した時と同等です．

例）

```
*RES↵     エミュレーションCPUのリセット
*


*RES H ↵
IE-78C11 MONITOR VX.X [XX XXX XX]
   COPYRIGHT (C) 1985 NEC CORPORATION

 USER-SYSTEM VDD-ON

 CPU 78C10
 EXTERNAL MEMORY SIZE 64K BYTE
 RAE (E:ENABLE,D:DISABLE)= E

*
```

保守／廃止

３．１４　ＭＡＴ（演算）コマンド

表現形式：

＊ＭＡＴ␣ｅｘｐｒｅｓｓｉｏｎ↵

機　能：　入力された式の演算をし，その結果を１６進数，１０進数，８進数，２進数の４通りで表示します．

メイン・コマンド　：　ＭＡＴ

ザブ・コマンド　：　なし

オペランド　：　ｅｘｐｒｅｓｓｉｏｎ　式処理可能な数値式

オプション　：　なし

保守／廃止

３．１４．１　演算コマンドの説明

ＭＡＴ␣ｅｘｐｒｅｓｓｉｏｎ⤸

コマンド入力した後に式の答が，１６進，１０進，８進，２進で表示されます．

（演算値は２バイト）表示が終わると‘＊’に戻ります．

以下，具体例を示します．

例）

```
*MAT ((A*3)-2)*111Y  ⤸
C4H,196T,304Q,11000100Y
*

*MAT 0AH+(18Q*10T)*11Y+100H  ⤸
          ↑―――――8 進数は0 ～7 まで
 INPUT DATA ERROR
*

*MAT 10+1Q ⤸
11H,17T,21Q,10001Y
*MAT  10Q*2  ⤸
10H,16T,20Q,10000Y
*MAT 10+(10Q*2)  ⤸
20H,32T,40Q,100000Y
*

*MAT 0AH+(12Q*10T)11Y+1000H  ⤸
                  ↑―――演算子がぬけている．
 INPUT DATA ERROR
*

*MAT 0AH+(12Q*10T)*11Y+1000H ⤸
1136H,4406T,10466Q,1000100110110Y
*

*MAT F00H+100H ⤸
1000H,4096T,10000Q,1000000000000Y
*

*MAT 5*5*5*5 ⤸
271H,625T,1161Q,1110001Y
*MAT 5*5*5*5*5 ⤸
C35H,3125T,6065Q,110000110101Y
*MAT 5*5*5*5*5*5 ⤸
3D09H,15625T,6411Q,111101000011001Y
*MAT 5*5*5*5*5*5*5 ⤸
312DH,12589T,30455Q,110001001011101Y
*MAT 1234H AND 0FFH  ⤸
34H,52T,64Q,110100Y
*
```

保守／廃止

ＭＡＴは，四則演算（　）処理，論理演算が可能です．

ＭＡＴコマンドの実行時の数値入力で数値の後に基数（Ｈ，Ｔ，Ｑ，Ｙ）のないものはＳＵＦコマンドの指定に従います．

演算子は

四則演算	＋	；	加算
	－	；	減算
	＊	；	乗算
	／	；	除算
論理演算	AND	；	論理積
	OR	；	論理和
	XOR	；	排他論理和
	NOT	；	否定

（　）処理の（　）の数は３２組まで可能です．

ただし，入力数値，演算結果が２バイトを越える時は下位２バイトのデータが有効となります．

演算子優先順位

1	＊，／
2	＋，－
3	NOT
4	AND
5	OR　，XOR

保守／廃止

3．15　SUF（数値基数指定）コマンド

表現形式：

＊SUF [␣ {H / T / Q / Y}] ↵

機　能：　数値入力の基数を指定または表示します．

メイン・コマンド	：	SUF	
サブ・コマンド	：	H	16進
	：	T	10進
	：	Q	8進
	：	Y	2進
オペランド	：	なし	
オプション	：	なし	

保守／廃止

３．１５．１　ＳＵＦ（数値基数）の表示と変更

ＳＵＦ⤶

現在のコマンド入力時の数値入力の基数を表示して‘＊’に戻ります．

例）

```
*SUF ⤶
 H                ；現在の数値入力は，１６進とみなします．
*
```

数値基数の変更を行なう場合は‘＊’の後に‘ＳＵＦ　Ａ⤶’（Ａ＝Ｈ，Ｔ，Ｑ，Ｙ）と入力してください．何も表示せずに‘＊’に戻ります．それ以後のコマンドの数値入力は，数値の後にＨ，Ｔ，Ｑ，Ｙがつかない限りＳＵＦで指定された基数で処理されます．

以下に例を示します．

```
*SUF ⤶
H
*MAT 10 ⤶
10H,16T,20Q,10000Y
*SUF T ⤶

*MAT 10 ⤶
AH,10T,12Q,1010Y
*SUF Q ⤶

*MAT 10 ⤶
8H,8T,10Q,1000Y
*SUF Y ⤶

*MAT 10 ⤶
2H,2T,2Q,10Y
*
*SUF A ⤶

 COMMAND FORMAT ERROR
*
```

保守／廃止

3．16　ASM（アセンブル）コマンド

表現形式:

＊ASM〔␣addr〕⤶

機　能：　指定されたアドレス以降のメモリの内容をアセンブル形式で変更します.

メイン・コマンド　：　ASM
サブ・コマンド　：　なし
オペランド　：　addr（スタート・アドレス）
オプション　：　なし

3．16．1　ASMコマンドの説明

ASM（アセンブル）コマンドは，マッピングされたプログラム・メモリ（RAM）のアセンブル形式による変更が可能なコマンドです.（ただし，マッピングされていないプログラム・メモリは‘NON－MAP　AREA　ACCESS’と出力され実行できません.）

操作手順を順番に説明します.

①コマンド入力待ち‘＊’の状態で‘ASM　addr⤶’と入力してください.
addrで指定されたアドレスからASMコマンドがスタートします.

②アドレスの1命令分を逆アセンブルし，‘＝’を出力して，入力待ちの状態となります.

③変更しない時は‘⤶’を入力して，次のアドレスに移り②へ戻ります.
変更する時は，変更したい命令のニーモニックを入力し，最後に‘⤶’を入力します．アセンブラ・コマンドを終了したい場合は‘END⤶’を入力することにより‘＊’に戻ります.

④入力されたニーモニックが正しい時には，そのオブジェクト・コードを出力し，入力待ちになります.
入力されたニーモニックに誤りがあった場合は，4バイトのNOPのオブジェクトとエラー・フラグが出力され，入力待ちになります.

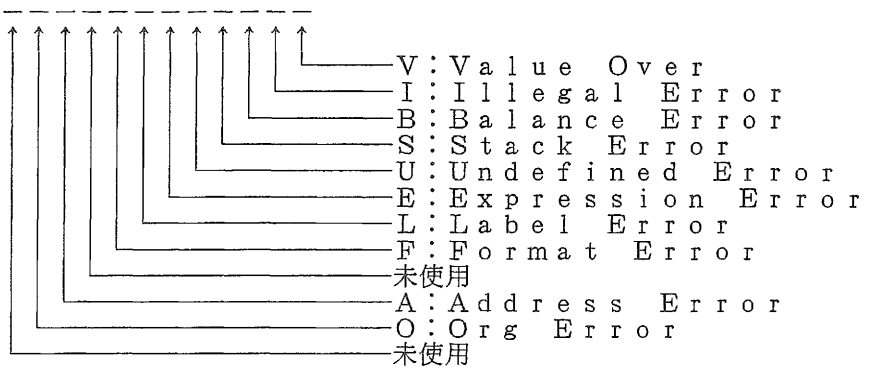
保守／廃止

⑤ ‘)’を入力するとオブジェクト・コードをメモリ上に書込み，アドレスを次の命令に移して②へ戻ります．

スペース・キー入力の時は，アセンブルされたオブジェクトをキャンセルし，アドレスそのままで②へ戻ります．

３．１６．２　エラー・フラグの説明

入力アセンブラがエラーの時は‘－’またはエラー・コードのイニシャルが１０個出力されます．



アセンブルしてエラーとなった時は，エラーのイニシャルが‘－’の代わりに出力されます．

Ｆｏｒｍａｔ　Ｅｒｒｏｒの時は下のようになります．

－－－－Ｆ－－－－－－－

なお，アセンブラの数値入力は，数値入力の後に基数（Ｈ，Ｔ，Ｑ，Ｙ）がない限りＳＵＦコマンドで指定された基数に従います．また，数値入力は式処理機能があり，ＭＡＴコマンドと同じ演算子が使用できます．

Value Over	：ｵﾍﾟﾗﾝﾄﾞ 部の数値入力がｵｰﾊﾞﾌﾛｰです．
Illegal Error	：使用できない文字が記述されています．
Balance Error	：括弧または引用符の記述が誤っています．
Stack Error	：式の記述が複雑すぎます．
Undefined Error	：ｱｾﾝﾌﾞﾗ 未定義入力があります．
Expression Error	：演算子の記述が誤っています．
Label Error	：解析が不可能なﾆｰﾓﾆｯｸ入力になっています．
Format Error	：入力形式が誤っています．
Address Error	：ｱﾄﾞﾚｽ入力がFFFFH 番地を越えています．
Org Error	：ｵﾙｸﾞ で指定されたｱﾄﾞﾚｽ値が，現在のｱﾄﾞﾚｽ値より低い場合に表示されます．

以下に例を示します.

```
*MEM D X↵

 0000  00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F

*ASM↵

 ADDR   OBJECT          MNEMONIC
 0000   00              NOP                         = ↵
 0001   0102            LDAW     00002H             = ↵
 0003   03              DCX      SP                 = ↵
 0004   040506          LX1      SP,00605H          = ↵
 0007   0708            ANI      A,00008H           =ANA V,A↵
 0007   60 08 ↵                                                    FORMAT
 0009   09              MOV      A,EAL              =ADD A,30 ↵      ERROR
 0009   00 00 00 00 ----F<------<------スペース入力
 0009   09              MOV      A,EAL              =ADD A,30H↵
 0009   00 00 00 00 ----F--------<------スペース入力
 000D   0D              MOV      A,E                =XRA L,A↵
 000D   60 17 ↵
 000F   0F              MOV      A,L                = ↵
 0010   FF              JR       00010H             = ↵
 0011   FF              JR       00011H             = ↵
 0012   FF              JR       00012H             =END↵
```

ＮＯＮ－ＭＡＰエリアをアクセスすると以下のようになります,

```
*MAP W 0,FF ↵

*MAP K 100,1FF↵

ASM FD↵

 ADDR   OBJECT          MNEMONIC
 00FD   0B              JR       00109H             =MOV A,B ↵
 00FD   0A
 00FE   4741            ONI      A,00041H           = ↵

 NON-MAP AREA ACCESS

*
```

保守／廃止

３．１７　ＤＡＳ（逆アセンブル）コマンド

表現形式：

＊ＤＡＳ〔␣ｐａｒｔｉｔｉｏｎ〕↵

機　能：　指定アドレス範囲のメモリの内容を逆アセンブル表示します．

メイン・コマンド　：　ＤＡＳ

サブ・コマンド　　：　なし

オペランド　　：　ｐａｒｔｉｔｉｏｎ（逆アセンブラ範囲）

オプション　　：　なし

３．１７．１　逆アセンブル・コマンドの説明

コマンド入力待ち‘＊’で‘ＤＡＳ　ｐａｒｔｉｔｉｏｎ↵’と入力してください．ｐａｒｔｉｔｉｏｎで指定された領域のメモリ内容を逆アセンブル表示します．

マッピングされていないところを逆アセンブラしようとすると‘ＮＯＮ－ＭＡＰ　ＡＲＥＡ　ＡＣＣＥＳＳ’と出力され，‘＊’に戻ります．

逆アセンブル表示を中断する時は，ＣＴＲＬ－Ｃ，またはＥＳＣキーを入力してください．

保守／廃止

３．１８　ＭＯＶ（メモリ転送）コマンド

表現形式：

＊ＭＯＶ␣{Ｕ／Ｉ}␣ｐａｒｔｉｔｉｏｎ␣ａｄｄｒ↵

機　能：　エミュレーション・メモリとユーザ・システム側メモリとの間で，ｐａｒｔｉｔｉｏｎで指定されたアドレス範囲のメモリ内容を，ａｄｄｒを先頭とするメモリ・エリアへ転送します．

メイン・コマンド　：　ＭＯＶ

サブ・コマンド　：　Ｕ　エミュレーション・メモリからユーザ・システム上のメモリへデータ転送をする．

：　Ｉ　ユーザ・システム上のメモリからエミュレーション・メモリへデータ転送をする．

オペランド　：　ｐａｒｔｉｔｉｏｎとａｄｄｒ

オプション　：　なし

３．１８．１　ＭＯＶコマンドの概要

ＭＯＶコマンドは，エミュレーション・メモリとユーザ・メモリとの間で，２５６バイト単位でデータ転送を行なうコマンドです．

転送先はマッピングされている必要がありますが，転送元はマッピングされている必要はありません．

保守／廃止

図３－１　メモリ内容の転送

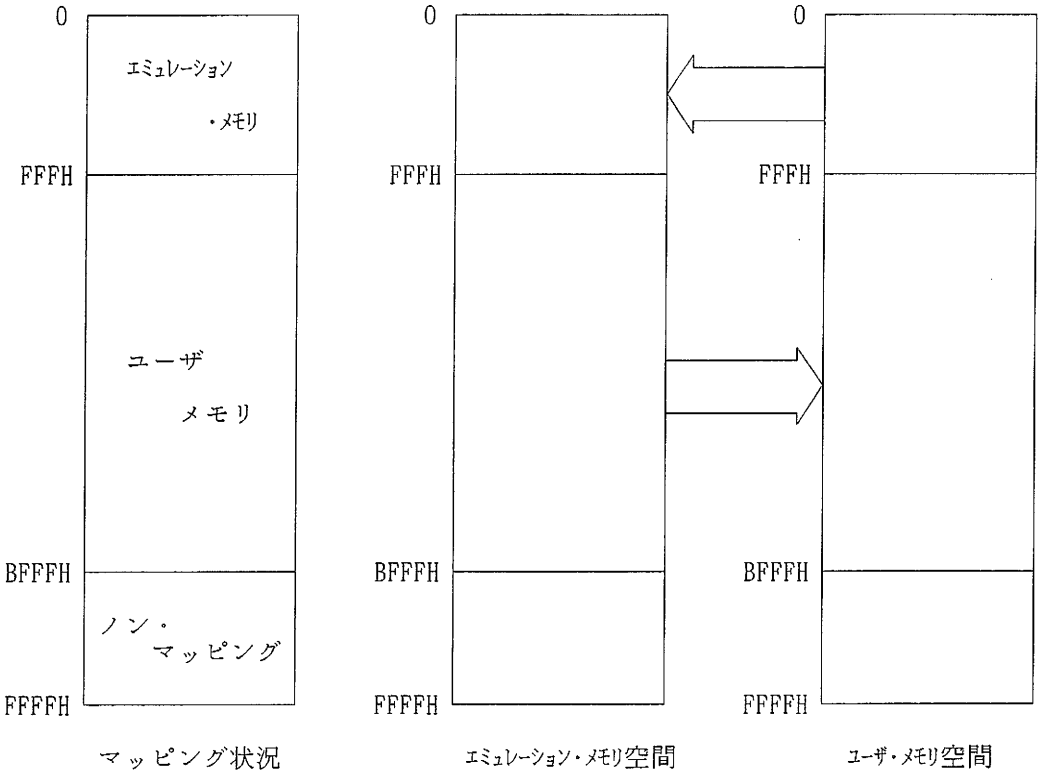


図３－１に示しましたように，マッピングされたメモリに同一アドレス空間のマッピングされていないメモリからデータ転送を行なうことができます．

ＣＴＲＬ－ＣおよびＥＳＣキーによるコマンド実行の停止は可能ですが，その場合メモリの内容は保障されません．

３．１８．２　エミュレーション・メモリからユーザ・システムへのメモリ転送

ＭＯＶ␣Ｕ␣ｐａｒｔｉｔｉｏｎ␣ａｄｄｒ↵

エミュレーション・メモリのｐａｒｔｉｔｉｏｎで指定されたアドレス範囲のメモリ内容を，ユーザ側のａｄｄｒで指定されたアドレスを先頭とする範囲へ転送します．

転送先のマッピングは，ユーザー・システム側に設定されてなければなりません．

以下に例を示します．

例）

```
*MAP W 0,1FFF ↵

*MEM D XXH↵
0000   01 02 03 04 05 06 07 08   09 00 01 02 03 04 05 06
0010   07 08 09 00 01 02 03 04   05 06 07 08 09 00 01 02
0020   03 04 05 06 07 08 09 00   01 02 03 04 05 06 07 08
0030   09 00 01 02 03 04 05 06   07 08 09 00 01 02 03 04
0040   05 06 07 08 09 00 01 02   03 04 05 06 07 08 09 00
0050   01 02 03 04 05 06 07 08   09 00 01 02 03 04 05 06
0060   07 08 09 00 01 02 03 04   05 06 07 08 09 00 01 02
0070   03 04 05 06 07 08 09 00   01 02 03 04 05 06 07 08
0080   09 00 01 02 03 04 05 06   07 08 09 00 01 02 03 04
0090   05 06 07 08 09 00 01 02   03 04 05 06 07 08 09 00
00A0   01 02 03 04 05 06 07 08   09 00 01 02 03 04 05 06
00B0   07 08 09 00 01 02 03 04   05 06 07 08 09 00 01 02
00C0   03 04 05 06 07 08 09 00   01 02 03 04 05 06 07 08
00D0   09 00 01 02 03 04 05 06   07 08 09 00 01 02 03 04
00E0   05 06 07 08 09 00 01 02   03 04 05 06 07 08 09 00
00F0   01 02 03 04 05 06 07 08   09 00 01 02 03 04 05 06
*MAP  U  0,0FFFH↵
*MEM  F  XXH↵
*MOV  U  0,FFH  0↵
```

```
*MEM D XXH ↵
0000   01 02 03 04 05 06 07 08   09 00 01 02 03 04 05 06
0010   07 08 09 00 01 02 03 04   05 06 07 08 09 00 01 02
0020   03 04 05 06 07 08 09 00   01 02 03 04 05 06 07 08
0030   09 00 01 02 03 04 05 06   07 08 09 00 01 02 03 04
0040   05 06 07 08 09 00 01 02   03 04 05 06 07 08 09 00
0050   01 02 03 04 05 06 07 08   09 00 01 02 03 04 05 06
0060   07 08 09 00 01 02 03 04   05 06 07 08 09 00 01 02
0070   03 04 05 06 07 08 09 00   01 02 03 04 05 06 07 08
0080   09 00 01 02 03 04 05 06   07 08 09 00 01 02 03 04
0090   05 06 07 08 09 00 01 02   03 04 05 06 07 08 09 00
00A0   01 02 03 04 05 06 07 08   09 00 01 02 03 04 05 06
00B0   07 08 09 00 01 02 03 04   05 06 07 08 09 00 01 02
00C0   03 04 05 06 07 08 09 00   01 02 03 04 05 06 07 08
00D0   09 00 01 02 03 04 05 06   07 08 09 00 01 02 03 04
00E0   05 06 07 08 09 00 01 02   03 04 05 06 07 08 09 00
00F0   01 02 03 04 05 06 07 08   09 00 01 02 03 04 05 06
*
```

３．１８．３　ユーザ・システムからエミュレーション・メモリへのメモリ転送

ＭＯＶ␣Ｉ␣ｐａｒｔｉｔｉｏｎ␣ａｄｄｒ↵

ユーザ・システム側のｐａｒｔｉｔｉｏｎで指定されたアドレス範囲のメモリ内容を，エミュレーション・メモリのａｄｄｒで指定されたアドレスを先頭とする範囲へ転送します．転送先のマッピングは，エミュレーション・メモリに設定されていなければなりません．

以下に例を示します．

例）

```
*MAP U 9XXXH ↵
*MAP W 5XXXH ↵
*MEM F 9000H,90FFH 9,8,7,6,5,4,3,2,1↵

*MEM D 9000H,901FH↵
9000   09 08 07 06 05 04 03 02   01 09 08 07 06 05 04 03
9010   02 01 09 08 07 06 05 04   03 02 01 09 08 07 06 05
*
*MEM D 50XXH↵
5000   00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00
5010   FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
5020   00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00
5030   FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
5040   00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00
5050   FF FF FF FF FF FF FF FF   FF FF FF FF FF FF FF FF
*                                          ↖CTRL-Cまたは ESCｷｰ入力
*MOV I 9000H,90FFH   5000H↵

*MEM D 500XH↵
5000   09 08 07 06 05 04 03 02   01 09 08 07 06 05 04 03
*
```

保守／廃止

３．１９　自己診断コマンド（ＤＩＧ）

表現形式：

＊ＤＩＧ〕

メイン・コマンド　：　ＤＩＧ

サブ・コマンド　　：　なし

オペランド　　　　：　なし

オプション　　　　：　なし

ＩＥ－７８Ｃ１１は自分自身で自己のハードウェアを検査する機能を持っています．このコマンドにより，ＩＥ－７８Ｃ１１の調子がおかしいと思われる時，その場で診断することができます．

自己診断機能は，以下の項目をチェックします．

1）エミュレーション・プローブ内発振回路

2）内部メモリ

3）ＭＯＤＥ０，ＭＯＤＥ１

4）ポートＤ，ポートＦ

5）ポートＡ，ポートＢ，ポートＣ

6）Ａ／Ｄ入力ポート（ＡＮ０～ＡＮ７）

7）シリアル入力

診断方法としては，ＩＥ－７８Ｃ１１の電源を切り，エミュレーション・プローブをドライバ・モジュール上部にある自己診断コネクタに差し込みます．このときプローブのスイッチは‘ＥＸＴ’側にします．そして，電源を入れると次のようにメッセージが出力されます．

```
IE-78C11 MONITOR VX.X[XX XXX XX]
   COPYRIGHT (C) 1985 NEC CORPORATION

USER-SYSTEM VDD-ON
```

```
CPU 78C10
EXTERNAL MEMORY SIZE 64K BYTE
RAE (E:ENABLE,D:DISABLE)=
```

次にＲＡＥをＤ（ＤＩＳＡＢＬＥ）にセットします．

プロンプト‘＊’が出力されたら自己診断コマンド‘ＤＩＧ↵’を入力し，自己診断を起動します．診断終了後は自動的にシステム・リセットされます．

以下に実行例を示します．

```
*DIG↵

EXTERNAL CLOCK TEST.OK
ALTERNATE RAM TEST..OK
USER MEMORY TEST....OK
HARDWARE TEST
  M0,M1.......OK
  PORT D,F....OK
  PORT A,B,C..OK
  ANALOG IN...OK
  SERIAL I/O..OK

DIAGNOSTIC COMPLETE
```

注）自己診断の結果がＮＧ（不良）となった場合は，購入先にご連絡ください．

保守／廃止

３．２０　ＰＧＭ（ＰＲＯＭプログラマ制御）コマンド

表現形式：

＊ＰＧＭ〕

機　　能：　ＣＨ２に接続されたＰＧ（ＰＲＯＭプログラマ）をＩＥ－７８Ｃ１１で使用しているコンソールより直接制御できるようにします．

メイン・コマンド　：　ＰＧＭ

サブ・コマンド　　：　なし

３．２０．１　ＰＧ－１０００，ＰＧ－２０００との接続

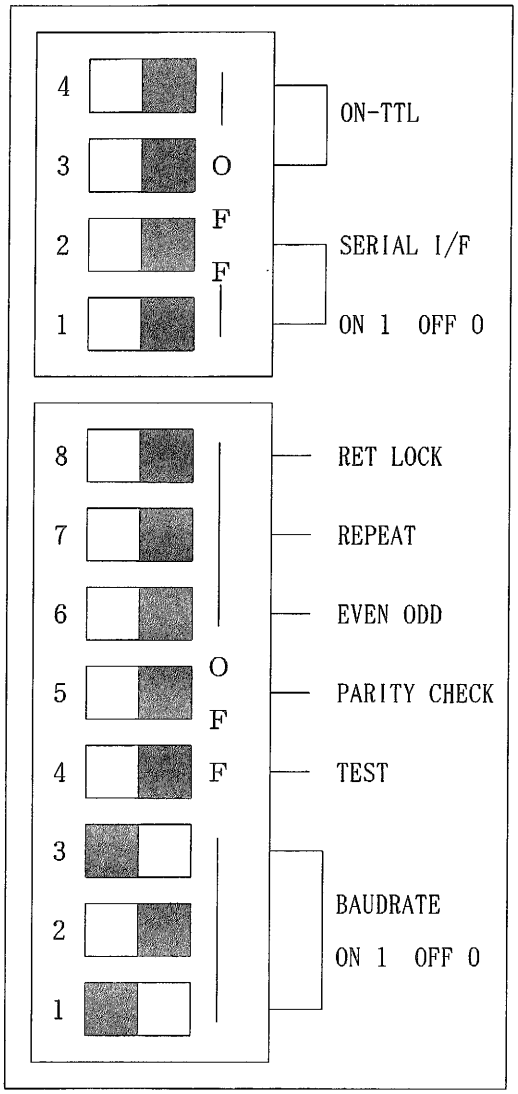
ＩＥ－７８Ｃ１１にＰＲＯＭプログラマ（ＰＧ－１０００，ＰＧ－２０００）を接続する際のハードウェア設定および接続法について説明します．

（1）ＩＥ－７８Ｃ１１，ＰＧ－１０００，ＰＧ－２０００およびターミナルのボーレートをかならず合わせてください．

ＩＥのボーレートは第４章を参照してください．ＰＧ側のボーレートは，図３－３（ＰＧ－１０００の場合），および図３－４（ＰＧ－２０００の場合）を参照してください．

（2）PG－1000を使用する場合は，図3－3のように底面部ディップ・スイッチを設定してください．ボーレートはIEのコントロール・ボードに合わせてください．

図3－3　PG－1000　底面部ディップ・スイッチの構成



ONを示します．

OFFを示します．

表3－2　PG－1000ボーレートの設定

1	2	3	BAUD RATE
0	0	0	110
0	0	1	300
0	1	0	600
0	1	1	1200
1	0	0	2400
1	0	1	4800
1	1	0	9600
1	1	1	－

（注意1）表3－2の‘1　2　3’は8連ディップ・スイッチの‘1　2　3’です．

（注意2）表3－2において1はON，0はOFFを示します．

保守／廃止

（3）ＰＧ－２０００を使用する場合は，図３－４のように底面部ディップ・スイッチを設定してください．ボーレートはＩＥのコントロール・ボードに合わせてください．

図３－４　ＰＧ－２０００　底面部ディップ・スイッチの機能

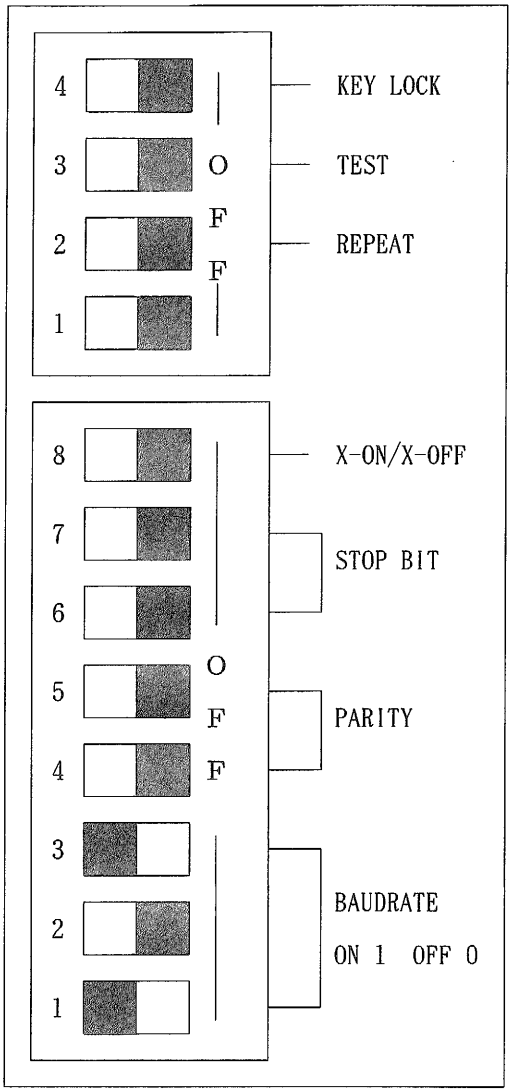


表３－３　ＰＧ－２０００ボーレートの設定

1	2	3	BAUD RATE
0	0	0	110
1	0	0	300
0	1	0	600
1	1	0	1200
0	0	1	2400
1	0	1	4800
0	1	1	9600

（注意１）表３－３の‘１　２　３’は８連ディップ・スイッチの‘１　２　３’です．

（注意２）表３－３において１はＯＮ，０はＯＦＦを示します．

保守／廃止

（4）ＩＥ－７８Ｃ１１の設定

(a) ＪＰ３を図３－６のジャンパ設定に従って変えてください．

(b) ＩＥコントロール・ボード上のディップ・スイッチ（ＤＰ１）のＮｏ．６はＯＮにしてください．

（注意）ＣＨ１側のリーダ処理はハードウェア・コントロール・モードを使用してください．

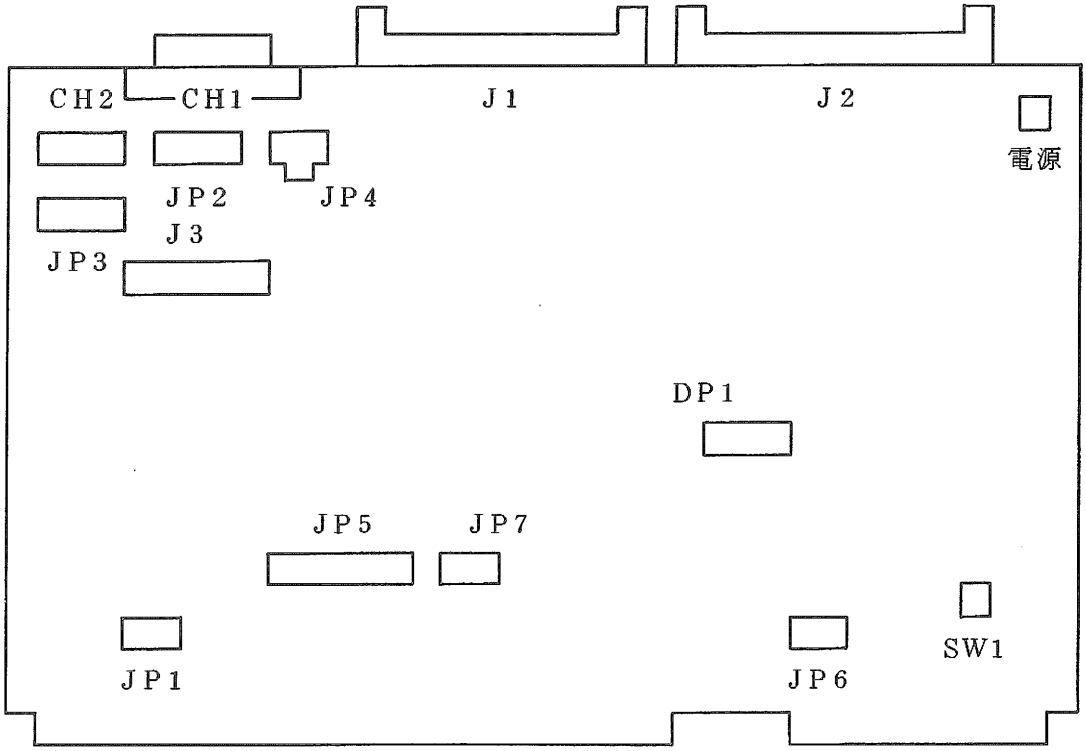
(c) ＰＧ－１０００およびＰＧ－２０００とＩＥ－７８Ｃ１１の接続にはＩＥ－７８Ｃ１１に付属しているシリアル・ケーブルⅡ（ＣＨ２用）を使用してください．

ＩＥ－７８Ｃ１１　ＣＨ２用　シリアル・ケーブルⅡの結線情報を図３－７に示します．

コミュニケーションがとれない場合は，図３－７に従ってチェックを行なってください．

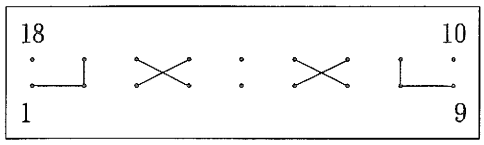
保守／廃止

図３－５　ＩＥ－７８Ｃ１１コントロール・ボード



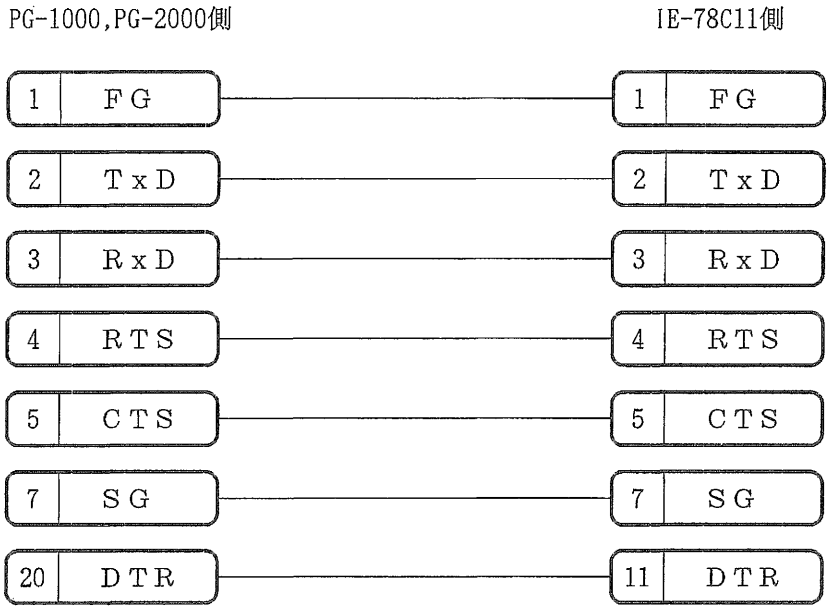
上図のＪＰ３を次のように設定します.

図３－６　ＪＰ３のジャンパ設定



次に接続時の結線情報を示します．

図３－７　結線情報



（５）ＰＧのシリアル・モード選択

（ａ）ＰＧ－１０００とＰＧ－２０００とは，取扱いが異なるので注意してください．

（ｂ）ＩＥ－７８Ｃ１１とＰＧ－１０００，ＰＧ－２０００のシリアル回線の通信開始方式

①ＰＧ－１０００の場合

RESET キー → SERIAL B キー

ＩＥ－７８Ｃ１１とＰＧ－１０００がシリアル回線でつながります．

②ＰＧ－２０００の場合

RESET キー → REM B キー → START キー

ＩＥ－７８Ｃ１１とＰＧ－２０００がシリアル回線でつながります．

保守／廃止

以下にＰＧ－１０００，ＰＧ－２０００のキー配置を示します．

図３－８　ＰＧ－１０００のキー配置

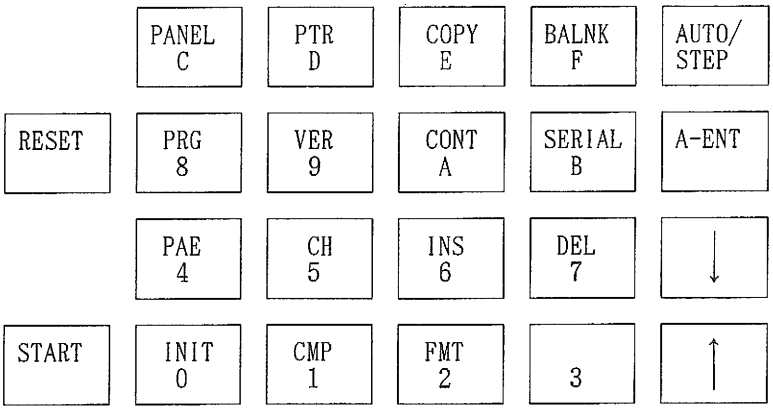
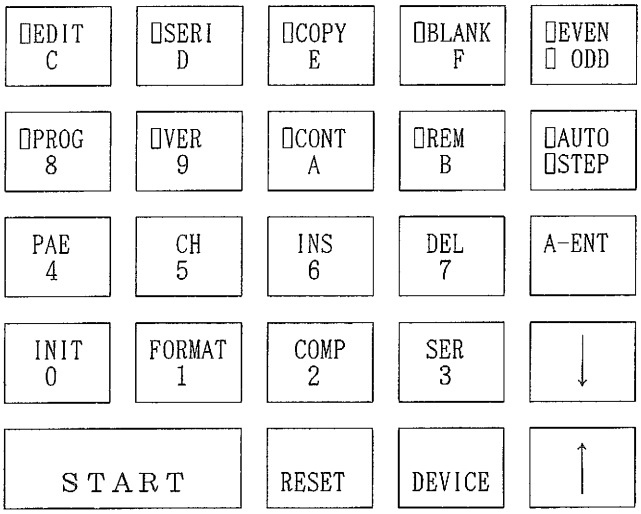


図３－９　ＰＧ－２０００のキー配置



保守／廃止

3．20．2　PG－1000およびPG－2000の概要

PG－1000およびPG－2000を使用して読み出し，書き込みができるROMの種類を示します．

PG－1000は，PG－1001モジュール，PG－1002モジュールおよびPG－1003モジュールのうち，どのモジュールを使用するかによってデータの読み出し，書き込みのできるROMがちがいます．(PG-1001,PG-1002,PG-1003 および PG-2000 取扱説明書　参照)

（1）PG-1000 を使用した場合

・PG-1001 モジュール使用の場合

μPD8741A，μPD8748，μPD8748H，

μPD8749H，μPD8755A

・PG-1002 モジュール使用の場合

μPB403，μPB405，μPB406，μPB409，

μPB417，μPB419，μPB423，μPB425，

μPB426，μPB428，μPB429

・PG-1003 モジュール使用の場合

μPD78P09R

（2）PG-2000 を使用した場合

μPD2716，μPD2732，μPD2732A

μPD2764，μPD27128，μPD27C64

μPD27256，μPD27C256

保守／廃止

３．２０．３　ＰＧＭコマンドの起動と終了

（１）ＰＧＭコマンドを入力しますと，ＰＧからのプロンプト‘＊’が出力されます．

```
＊ＰＧＭ↵
ＰＧＭ　ＭＯＤＥ

＊
```

プロンプトが出力されない場合は，３．２０．１　（５）（ｂ）の①（ＰＧ－１０００）または②（ＰＧ－２０００）の処理を再度行なってください．それでもプロンプトが出力されない場合は，再度設定の確認（１）～（４）を行なってください．

```
＊ＰＧＭ↵
ＰＧＭ　ＭＯＤＥ
■　←カーサが停止し，プロンプトが出力されない状態
```

（２）ＰＧＭコマンドを起動した時はトランジェント・モード（コンソールへエコーバックしないモード）になっています．ここで，‘Ｉ↵’と入力してください．インテリジェント・モード（コンソールへエコーバックするモード）へ移行します．

（３）ＰＧのコマンドを操作します．（実行例は３．２０．５を参照してください．）

（４）ＰＧＭコマンドを終了したい場合は，コンソールよりＣＴＲＬ－Ｚ（１ＡＨ）を入力することにより，通常のＩＥのコマンドの使用が可能となります．

次に表示例を示します．

```
＊　←CTRL-Z入力
ＰＧＭ　ＥＮＤ
＊
```

（注意）プロンプトはＩＥの場合も，ＰＧの場合も‘＊’なので注意してください．

コマンド入力ライン途中およびコマンド実行時にＣＴＲＬ－Ｚを入力することによってＰＧＭコマンドを終了し，ＩＥのコマンド待ちの状態に復帰するこ

保守／廃止

とができます.

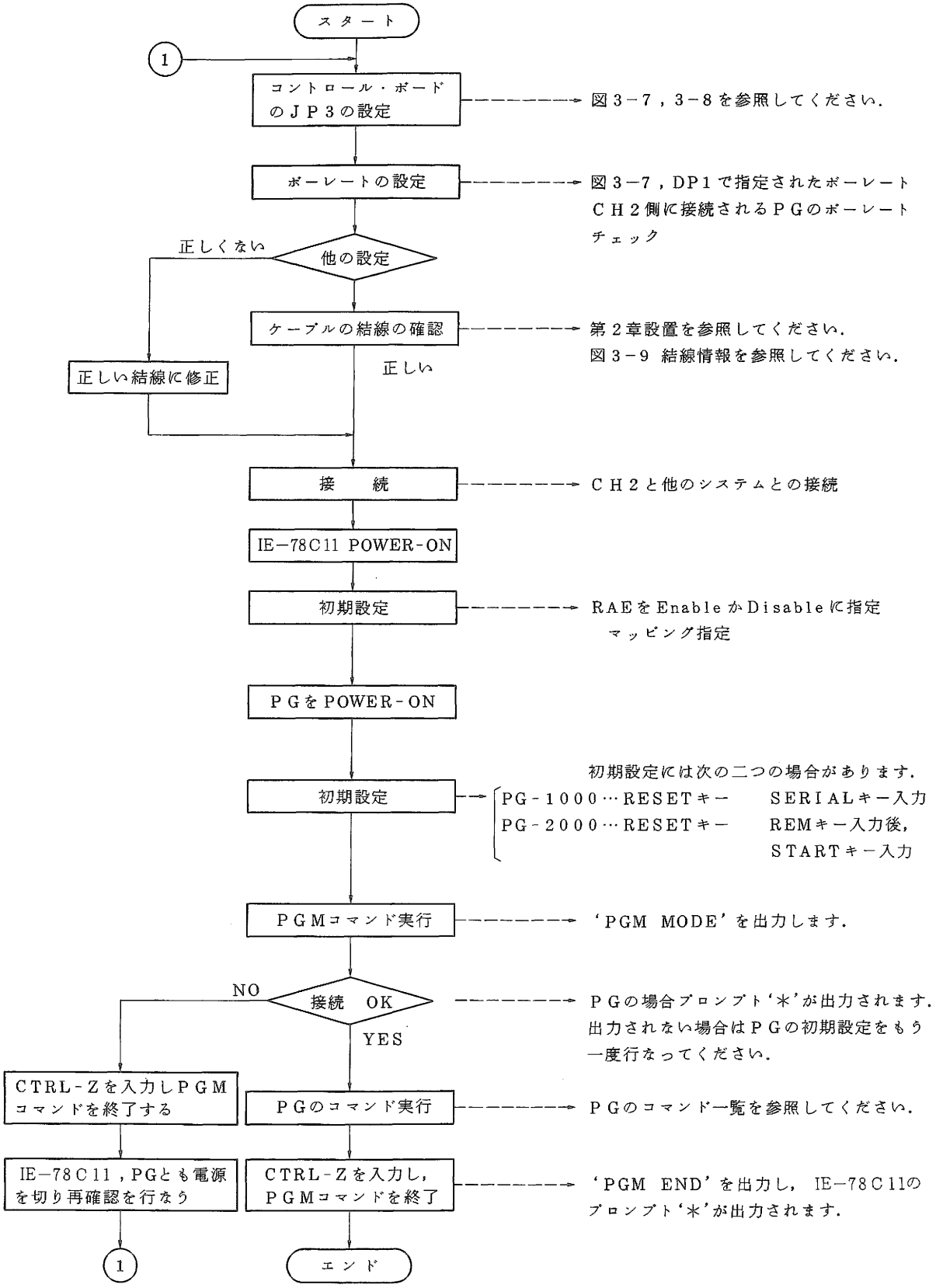
ただし，Ｌコマンドおよびｐコマンド（３．２０．４　（２）Ｌコマンド・Ｐコマンド詳細　参照）実行中はＣＴＲＬ－Ｚの使用ができません.

（注意）ＰＧを接続した状態でＩＥの通常のコマンド操作中にＣＴＲＬ－Ｐ（ＣＨ１とＣＨ２の同時出力機能）を入力しないでください.
ＣＴＲＬ－Ｐを入力した場合は，ＰＧ側にデータを出力してしまいＰＧが意図しない異常動作をおこす可能性があります.
ＰＧＭコマンドを実行するとＣＴＲＬ－Ｐモードは自動解除され，ＰＧＭコマンド終了後もＣＴＲＬ－Ｐは解除されたままになります.

次ページに，ハードウェアの設定から，ＰＧＭコマンドの実行終了までの一連の手順をフローチャートで示します.（図３－１０参照）

保守／廃止

図３－１０　ＰＧの接続，起動および終了手順



保守／廃止

３．２０．４　ＰＧＭコマンド詳細

ＰＧＭコマンドは，ＩＥ－７８Ｃ１１で使用している端末よりＰＧのコマンドをコントロールするためのものです．

また，ＩＥ－７８Ｃ１１内のマッピングされたメモリの内容を，ＰＧへ転送，または，ＰＧのメモリの内容を，ＩＥ－７８Ｃ１１のマッピングされたメモリへ転送することができます．

次に，ＰＧのコマンド一覧を示します．（表３－２参照）

Ｌコマンド，Ｐコマンド以外は，ＰＧのコンソール・モードのコマンドと同じですので，詳細はＰＧの取扱説明書を参考にしてください．

（１）ＰＧ－１０００，ＰＧ－２０００コマンド一覧

表３－２　ＰＧ－１０００，ＰＧ－２０００コマンド一覧

ｺﾏﾝﾄﾞ	形　式	コ　マ　ン　ド　内　容
Ａ	＊Ａｓ，ｅ，ｒ⤶	パラメータの設定をします．
Ｅ	＊Ｅｒ⤶	ＰＧのバッファのデータを変更します． 形式は以下の通りです． ＊Ｅｒ⤶ ｒ　XX- YY　XX- YY　XX-　⤶ 入力データ 現在設定されているデータ データ入力形式 ・データを入力する（１６進以外を入力するとその時点で‘？’表示） ・スペース・キー入力（データ変更なし）
Ｆ	＊Ｆｒ，ｒｅ，ｄ	ＰＧのバッファをｄで初期化をします．
Ｉ	＊Ｉ⤶	インテリジェント・モード（コンソールへエコーバックするモード）へ移行します．
Ｊ	＊Ｊ⤶	ＰＴＲよりの入力 パリティ・エラー時，‘？’を表示します．
Ｏ	＊Ｏｒ，ｒｅ⤶	ＰＧのバッファの内容を表示します． 表示形式 ｒ　00 00 00 00 00 00　…… 00
Ｒ	＊Ｒｒ，ｒｅ⤶	ＰＲＯＭの内容をＰＧのバッファへ転送します．
Ｓ	＊Ｓ⤶	ＰＲＯＭの選択
Ｔ	＊Ｔ⤶	トランジェント・モード（コンソールへエコーバックしないモード）へ移行します．

保守／廃止

ｺﾏﾝﾄﾞ	形　式	コ　マ　ン　ド　内　容
V	＊Ｖｓ,ｅ,ｒ〉	ＰＲＯＭの内容とＰＧバッファの内容との比較をします. データがちがう時，‘?’を表示します.
W	＊Ｗｓ,ｅ,ｒ〉	ＰＧのバッファの内容をＰＲＯＭに書き込みます. 書込みエラー時，‘?’を表示します.
Y	＊Ｙ〉	現在設定されているパラメータの表示および設定が可能です．ＰＲＯＭの開始番地，終了番地，ＰＧのバッファの開始番地の設定および表示が可能です.
Z	＊Ｚ〉	ＰＲＯＭにデータが書き込まれているかどうかをチェックします．データが書き込まれていない場合は何も表示せず，データが書き込まれている場合は‘?’を表示します.
L	＊Ｌbias〉 PARTITION=i,ie〉	ＩＥ－７８Ｃ１１のマッピングされているi番地から，ie番地までのデータをＰＧバッファのi+bias番地からie+bias 番地まで転送します.
P	＊Ｐｒ，re〉 bias=i〉	ＰＧのバッファのｒ番地からｒｅ番地までのデータをＩＥ－７８Ｃ１１のマッピングされているr+i 番地からre+i番地まで転送します.

注１）　ｓ：ＰＲＯＭ開始番地

ｅ：ＰＲＯＭ終了番地

ｒ：ＰＧバッファ開始番地

re：ＰＧバッファ終了番地

ｉ：ＩＥ－７８Ｃ１１の開始番地

ie：ＩＥ－７８Ｃ１１の終了番地

ｄ：データ

エラー条件

Ｓ ＞ｅ

ｒ ＞re

ｉ ＞ie

１６進以外の入力

注２）　ＳコマンドはＰＧ－２０００のみにあります.

（２）Ｌコマンド・Ｐコマンド詳細

（ａ）Ｌコマンド

```
＊ＬXXXX ⤶                          ；XXXX･･･ ＰＧのロード・バイアス
ＰＡＲＴＩＴＯＮ＝YYYY,ZZZZ ⤶       YYYY･･･ IE-78C11ﾒﾓﾘの転送開始番地
                                    ZZZZ･･･ IE-78C11ﾒﾓﾘの転送終了番地
＊                                  ただしXXXX,YYYY,ZZZZはHEX 数字で，
                                    下位４桁が有効です．
```

ＩＥ－７８Ｃ１１のマッピングされたメモリ（YYYYからZZZZ）の内容をＰＧのバッファの（XXXX＋YYYY）から（XXXX＋ZZZZ）までに転送します．

途中で実行を中止したいときは，ＥＳＣキーを入力してください．

ＥＳＣキーを入力すると，ＰＧのコマンド入力待ちになります．

ＰＡＲＴＩＴＩＯＮの省略入力はできません．

４Ｋｂｙｔｅのデータを ＩＥ－７８Ｃ１１のメモリからＰＧのバッファへ転送するのに約３０秒かかります．

（例）

① 通常のＬコマンドの使用例

```
*LO ⤶                              ；PGのメモリ・バイアスを０にセットIE
PARTITION=0,FF ⤶                     のメモリの０番地からＦＦ番地の内容
                                     をPGへロード
＊
```

② エラー入力時

```
*LO
PARTITION=0,FX ⤶                   ；PARTITION の入力エラー
INPUT DATA ERROR
*

*LO ⤶
PARTITION=0,FF ⤶                   ；指定されたIEのメモリがマッピングさ
NON-MAP AREA ACCESS                  れていないのでエラーになります
*

*LO ⤶
PARTITION=0,FF ⤶                   ；IEからPGへデータ転送中にデータがう
?                                    まく送れなかった場合
*                                    （パリティ・エラー等）

*L8000 ⤶                           ；ＰＧのメモリ・アドレスをオーバして
?                                    データを転送しようとした時
*                                    または，途中でメモリアドレスをオー
                                     バしたとき
```

エラー表示後はＰＧのコマンド入力待ち‘＊’になります．

（ｂ）Ｐコマンド

```
＊ＰXXXX,YYYY ↵          ; XXXX･･･ ＰＧのバッファの転送開始番地
ＢＩＡＳ＝ZZZZ ↵            YYYY･･･ ＰＧのバッファの転送終了番地
                            ZZZZ･･･ IE-78C11 のロード・バイアス
＊                          XXXX,YYYY,ZZZZはＨＥＸ数字で有効桁は
                            下位４桁です.
```

ＰＧのメモリ（XXXXからYYYY）の内容をＩＥ－７８Ｃ１１のマッピングされたメモリ（XXXX+ZZZZ ）から（YYYY+ZZZZ ）までに転送します.

転送が正常に終了した場合は‘＊’を表示します.

途中で実行を中止したいときは，ＥＳＣキーを入力してください.

ＥＳＣキーを入力しますと，ＰＧのコマンド入力待ちとなります.

ＢＩＡＳの省略入力はできません.

４Ｋｂｙｔｅのデータを ＩＥ－７８Ｃ１１のメモリからＰＧのバッファへ転送するのに約３０秒かかります.

（例）

① 通常のＰコマンドの使用例

```
＊Ｐ０，ＦＦ ↵          ; PGの０番地からFF番地の内容を
ＢＩＡＳ＝０ ↵              バイアス０でIE-78C11のメモリへ
                            転送します.
＊
```

② エラー表示

```
*P0,EF ↵                ; BIASの入力エラー
BIAS=X ↵
INPUT DATA ERROR
*

*P0,EF ↵                ; IE-78C11のメモリがマッピングさ
BIAS=100 ↵                  れていないのでエラーになります.
NON-MAP AREA ACCESS
*

*P0,1EF ↵               ; データ転送中にチェック・サム・
BIAS=25 ↵                   エラーが生じた場合
CHECK SUM ERROR
*

*P35,FF ↵               ; データ転送中に１６進数字以外の
BIAS=0 ↵                    キャラクタが転送された場合
BAD CHARACTER
*
```

```
*P0,1FF ↵                    ; IE-78C11のメモリ領域をオーバして
BIAS=FF20 ↵                    データを転送しようとしたとき
ADDRESS OVER
*

*P0,8000 ↵                   ; ＰＧのメモリ・アドレスより大きな
?                              アドレスを設定しようとした時
*
```

エラー表示後はＰＧのコマンド入力待ち‘＊’になります．

保守／廃止

３．２０．５　ＰＧＭコマンド実行例

ＰＧＭコマンドの実行例を順次説明します．

（１）ＩＥ－７８Ｃ１１の電源を投入すると，次のPOWER-ONメッセージが出力されます．

```
IE-78C11 MONITOR VX.X [XX XXX XX]
   COPYRIGHT (C) 1985 NEC CORPORATION

 USER-SYSTEM VDD-ON
 CPU 78C10
 EXTERNAL MEMORY SIZE 64K BYTE
 RAE (E:ENABLE,D:DISABLE)= D

 *
```

（２）コマンド受付け状態‘＊’が出力された後，マッピングの指定を行ないメモリをクリアします．

```
*MAP W XXXX ↵   ; 0 番地からFFFF番地までをIE-78C11の内部RAM へ
                  マッピングします．
*MEM F XXXX 0 ↵; 0 番地からFFFF番地までのIE-78C11のメモリを0
                  でクリアします．
```

（３）ＰＧＭコマンドを実行します．実行後，ＲコマンドによりＲＯＭの内容をＰＧのバッファへ転送します．

```
*PGM ↵          注）‘PGM MODE’以降の‘＊’はPGのプロンプトにな
PGM MODE            っています．

*                 ;I↵を入力（表３－２　Ｉコマンド参照）
*F0,1FFF,0 ↵      ;PGのメモリをクリアします．
*R ↵              ;ROM の内容をＹコマンドで設定したアドレス分だ
                    けPGのメモリへ転送します．
```

（４）読み込んだＲＯＭの内容をＩＥのメモリへ転送し，ＰＧＭコマンドを終了します．

```
*00,3F ↵          ;読み込んだＲＯＭの内容を表示します．
0000  69 00 6A 01 6B 02 6C 03 6D 04 6E 05 6F 06 68 07
0010  48 AF 48 AC 69 10 6A 11 6B 12 6C 13 6D 14 6E 15
0020  6F 16 68 17 54 24 00 00 00 00 00 00 00 00 00 00
0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
*P0,27 ↵          ;PGM のメモリの内容をIE-78C11のメモリへ転送しま
BIAS=0 ↵            す．

*                 ;CTRL-Zを入力することにより，PGM コマンドを終了
PGM END             します．
*                 ; ‘PGM END ’ 後の‘＊’はIE-78C11モニタのプロン
                    プトです．
```

（５）ＰＧＭコマンド終了後，データの転送されたメモリの内容を表示し確認します．

```
*MEM D 0,3F ↵        メモリの表示

  0000  69 00 6A 01 6B 02 6C 03 6D 04 6E 05 6F 06 68 07
  0010  48 AF 48 AC 69 10 6A 11 6B 12 6C 13 6D 14 6E 15
  0020  6F 16 68 17 54 24 00 00 00 00 00 00 00 00 00 00
  0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

*
```

（６）　アセンブル・コマンドにより，メモリの内容を変更します．

```
*ASM 14 ↵
ADRS    OBJECT      MNEMONIC
0014    6910        MVI       A,00010H      =MVI A,70 ↵
0014    6970 ↵
0016    6A11        MVI       B,00011H      =MVI B,71 ↵
0016    6A71 ↵
0018    6B12        MVI       C,00012H      =MVI C,72 ↵
0018    6B72 ↵
001A    6C13        MVI       D,00013H      =MVI D,73 ↵
001A    6C73 ↵
001C    6D14        MVI       E,00014H      =MVI E,74 ↵
001C    6D74 ↵
001E    6E15        MVI       H,00015H      =MVI H,75 ↵
001E    6E75 ↵
0020    6F16        MVI       L,00016H      =MVI L,76 ↵
0020    6F76 ↵
0022    6817        MVI       V,00017H      =END
```

（７）変更したメモリの内容をＰＧのメモリへ転送します．

```
*PGM ↵
PGM MODE

*LO ↵           ;IE-78C11のメモリ内容をＰＧのメモリへ転送します．
PARTITION=0,2F ↵
*
```

（８）未修正の箇所があったのでそれを変更後，ＰＲＯＭへ書き込んで終了します．

```
*00,3F ↵                          ;  (表示してみると23番地が未修正)
0000  69 00 6B 01 6A 02 6C 03   6D 04 6E 05 6F 06 68 07
0010  48 AF 48 AC 69 70 6A 71   6B 72 6C 73 6D 74 6E 75
0020  6F 76 68 17 54 24 00 00   00 00 00 00 00 00 00 00
0030  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00
*E22 ↵
0022  68-68 17-77 54- ↵          ; 23番地の内容を77H に変更します．
*W ↵                             ; PROMへコマンドで設定したアドレス
*V ↵                               分だけデータを書き込みおよびベリ
                                   ファイします．
*                                ; CTRL-Zを入力し，PGM コマンドを終
PGM END                            了します．
*                                ;'PGM END' 後の'* 'はIE-78C11のモ
                                   ニタのプロンプトなので注意してく
                                   ださい．
```

保守／廃止

コマンド一覧Ｉ

メイン・コマンド	サ　ブ	操　　作	機　　能
マッピング (MAP)	Def	MAP ↵	マッピング の表示
	W	MAP W partition ↵	partitionエリアをエミュレーションRAM としてマッピング 指定する.
	R	MAP R partition ↵	エミュレーションROM としてマッピング する.
	U	MAP U partition ↵	partitionエリアをユーザ・システムにマッピング指定する.
	K	MAP K partition ↵	partitionエリアのマッピング を解除する.
メモリ (MEM)	C	MEM C addr ↵	addr番地の内容を変更する.
	D	MEM D partition ↵	partitionエリアの内容を表示.
	E	MEM E partition ↵	partitionエリアのメモリ のテスト.
	F	MEM F partition string ↵	partitionエリアの内容をstringでイニシャライズ.
	G	MEM G partition string ↵	partitionエリアの内容とstringと一致した先頭アドレスの表示.
	M	MEM M partition addr ↵	partitionエリアの内容をaddrを先頭とする番地へ転送.
	V	MEM V partition addr ↵	partitionエリアの内容とaddrを先頭とするエリア と比較.
	X	MEM X partition addr ↵	partitionエリアの内容とaddrを先頭とするエリア の内容の交換.
レジスタ (REG)	C	REG C register-name ↵	指定したレジスタの内容の変更.
	D	REG D register-name ↵	指定したレジスタの内容の表示.
	Def	REG ↵	全レジスタの表示.
モード・レジスタ (MDR)	C	MDR C mode-register-name1	指定したモード・レジスタの内容の変更
	D	MDR D mode-register-name2	指定したモード・レジスタの内容の表示
	Def	MDR ↵	全モード・レジスタの表示
特殊レジスタ (SPR)	C	SPR C special-register-name1 ↵	指定した特殊レジスタの内容の変更
	D	SPR D special-register-name2 ↵	指定した特殊レジスタの内容の表示
	Def	SPR ↵	全特殊レジスタの表示

注）Ｄｅｆはｄｅｆａｕｌｔｅの略で，メイン・コマンドだけを入力した場合です.

保守／廃止

コマンド一覧Ⅱ

メイン・コマンド	サ　ブ	操　　作	機　　能
ロード (LOD)	－	LOD TTY?$addr ↵	TTY?よりバイアスaddrでプログラム・メモリ にロード する.
セーブ (SAV)	－	SAV　TTY? partition ↵	partitionエリアで示された内容をTTY?にセーブ する.
エミュレーション (RUN)	N	RUN N addr ↵	addr番地よりプログラム を実行.
	B	RUN B addr ↵	addr番地よりプログラム を実行し,ブレーク条件に従いブレーク.
	S	RUN S addr,step ↵	addr番地より,step 回数だけプログラム を実行.
	T	RUN T cond1 $as $R ↵ (as=D,E)	プログラム を実行し,ステップ 毎に表示. (cond1=addr,stepまたはaddr,register-name…value) (…は比較演算子
	Def	RUN ↵	現在の番地よりプログラム を実行.
フィジカル ブレーク (BR?)	A	BRA A=addrs V=values C=cond L=loop ↵	ブレーク条件の設定.アドレス,データ,コンディション, ループ 回数をセット します.
	D	BRD data ↵	オプション・ブレークのデータ をセット.
	E	BRE count ↵	外部ブレークまたはステップ.
	T	BRT count ↵	タイマ・データ のセット.
	Def	BRA ↵	データ の表示.
ロジカル ブレーク (BR?)	0　3	BRO BRA,… ↵	フィジカル・ブレーク・レジスタ の組合せデータ.
	Def	BR? ↵ (?=0～3)	データ の表示.
ブレーク モード (BRM)	—	BRM BRA,BRO,… ↵	フィジカル/ロジカル・ブレーク・データ のハード・セット.
	Def	BRM ↵	BRM にセット されたデータ の表示と ハード・セット.
トレース (TR?)	C	TRC a ↵ (a=I or M)	トレース・データ表示サイクルの指定.
	D	TRD b ↵ (b=トレース 数,ALL)	トレース・ダンプ.
	M	TRM c ↵ (c=NON,ALL)	トレースのモード・セット.
	P	TRP d ↵ (d=トレース 数,0,N)	トレース・ポインタ の移動.
	X	TRX A=addrs V=values C=cond PA=values PB=values ↵	トレース条件の設定.アドレス,データ,コンディション, ポート の設定
	S	TRS e ↵ (e=PB,EX)	ソース 選択
	Def	TR? ↵	データ の表示(TRPを除く)

保守／廃止

コマンド一覧Ⅲ

メイン・コマンド	サ　ブ	操　　作	機　　能
クロック (CLK)	I	CLK I ⤶	内部クロックへの指定
	U	CLK U ⤶	外部クロックへの指定.
	Def	CLK ⤶	現在使用しているクロックの表示.
リセット (RES)	H	RES H ⤶	IE-78C11 のリセット
	Def	RES ⤶	エミュレーションCPU のリセット
演算 (MAT)	—	MAT expression ⤶	演算コマンド.
サフィックス (SUF)	H	SUF H ⤶	入力数値をHEX.と見なす.
	T	SUF T ⤶	入力数値をDEC.と見なす.
	Q	SUF Q ⤶	入力数値をOCT.と見なす.
	Y	SUF Y ⤶	入力数値をBIN.と見なす.
	Def	SUF ⤶	現在のサフィックスの表示.
アセンブラ (ASM)	—	ASM addr ⤶	プログラム・メモリ のaddr番地から,オンライン・アセンブル を行なう
逆アセンブラ (DAS)	—	DAS partition ⤶	partitionエリアを逆アセンブル する.
メモリ 転送 (MOV)	I	MOV I partition addr ⤶	ユーザ・システムからエミュレーション・メモリへのメモリ転送を行なう.
	U	MOV U partition addr ⤶	エミュレーション・メモリからユーザ・システムへのメモリ転送を行なう.
自己診断 (DIG)	—	DIG ⤶	自己のハードウェアを診断する.
PROM プログラマ 制　御 (PGM)	—	PGM ⤶	ＰＲＯＭプログラマを制御できるようにする.

保守／廃止

メッセージ一覧

表　　示	意　　味
COMMAND FORMAT ERROR	ﾒｲﾝ･ｺﾏﾝﾄﾞ以後が間違っている.
MEMORY FAILURE	ﾌﾟﾛｸﾞﾗﾑ･ﾒﾓﾘ が異常.
CHECK SUM ERROR	ﾌﾟﾛｸﾞﾗﾑ･ﾛｰﾄﾞ 時に正しい ﾃﾞｰﾀが入力されなかったことを示します.
INPUT DATA ERROR	ﾃﾞｰﾀ の入力間違い.
NON-MAP AREA ACCESS	ﾏｯﾋﾟﾝｸﾞ されていないところをｱｸｾｽ.
MAPPING ERROR	ﾏｯﾋﾟﾝｸﾞ･ｱﾄﾞﾚｽ指定の誤り
SYSTEM DOWN	ｼｽﾃﾑ異常. EVACHIPが正しく動作していない.
COMMAND TOO LONG	ｺﾏﾝﾄﾞ入力文字数が多すぎます. (入力された文字数が 124個以上の時，出力されます.)
NON BREAK	強制ﾌﾞﾚｰｸを行なってもﾌﾞﾚｰｸしない時，出力されます. ﾌｧｰﾑｳｪｱ が正しく動作していないことを示します.
E-CPU RUN	ﾕｰｻﾞ･ﾌﾟﾛｸﾞﾗﾑ 実行中にCTRL-Cを入力して，ｺﾏﾝﾄﾞ待ちの状態にした時出力されます. この時,ｴﾐｭﾚｰｼｮﾝCPUは,ﾕｰｻﾞ･ﾌﾟﾛｸﾞﾗﾑを実行しているので，再度ｺﾏﾝﾄﾞ入力を行なうためには，強制ﾌﾞﾚｰｸするかﾘｾｯﾄ･ｺﾏﾝﾄﾞ を入力してください.
POINTER OVER SET	TRPｺﾏﾝﾄﾞでﾄﾚｰｽ･ﾃﾞｰﾀ数より多い数を入力した場合に表示されます.
OLDEST	TRPｺﾏﾝﾄﾞでﾄﾚｰｽ･ﾃﾞｰﾀの最も古い所へﾎﾟｲﾝﾀを移した時に表示されます.
NEWEST	TRPｺﾏﾝﾄﾞでﾄﾚｰｽ･ﾃﾞｰﾀの最も新しい所へﾎﾟｲﾝﾀを移した時に表示されます.
ADDRESS OVER	LODｺﾏﾝﾄﾞ，PGMｺﾏﾝﾄﾞでIEのﾒﾓﾘのOFFFH番地を越えてﾃﾞｰﾀ転送した時に表示されます.
NON TRACE DATA	ﾄﾚｰｽ･ﾃﾞｰﾀ がありません.

保守／廃止

第４章　シリアル・インタフェース

４．１　概　要

ＩＥ－７８Ｃ１１のシリアル・インタフェースは，ＲＳ－２３２－Ｃ規格に準拠しています．

しかし，機器ごとにＲＳ－２３２－Ｃのハンドシェイクおよびコード・プロートコルは少しずつ異なっているのが現状です．

そのためＩＥ－７８Ｃ１１は，コントローラ上のＤＩＰ　ＳＷＩＴＣＨとボード上のジャンパ（ＪＰ２およびＪＰ３）およびケーブル上配線の組合せで，一般に市販されている殆んどのターミナル，ＴＴＹ，ＰＴＰ，ＰＴＲ，ｅｔｃとインタフェースできるように考慮されています．

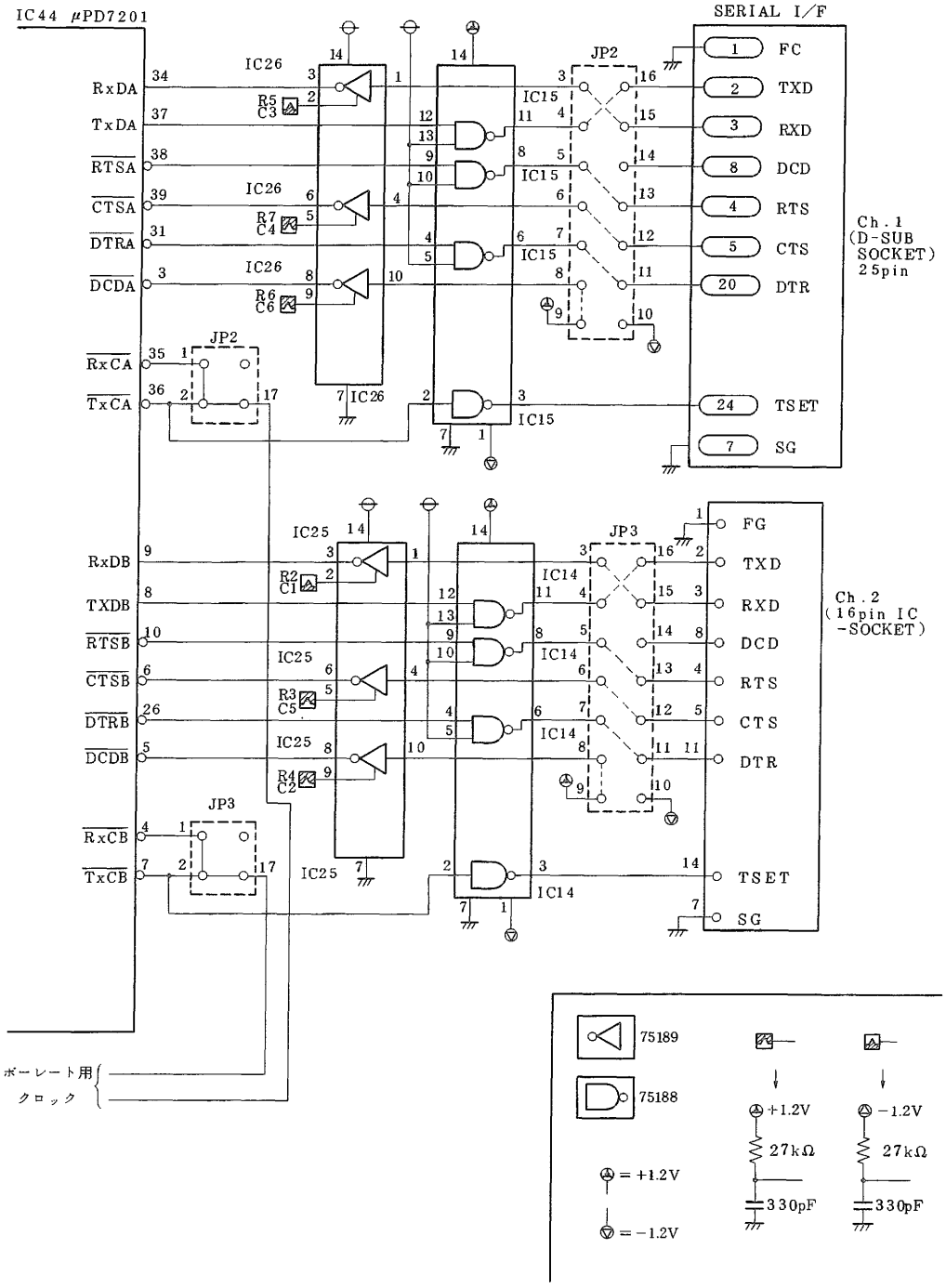
以下にその内容と，現在市販されているＲＳ－２３２－Ｃ規格のターミナルのうち三例について具体例を説明します．

保守／廃止

４．２　基本仕様

1）シリアル・コントローラ	μＰＤ７２０１マルチ・プロトコル シリアル・コントローラ
2）チャネル数	２．　ＣＨ１・・・・ターミナル用 ＣＨ２・・・・サブＩ／Ｏ用
3）ボーレート	１１０～９６００ボー （9600,4800,2400,1200,600,300,110 ）
4）通信方式	非同期
5）伝送フォーマット	スタート・ビット　１ビット キャラクタ長　８ビット パリティ・ビット　なし ストップ・ビット　２ビット
6）キャラクタ・コード	８ビット　ＡＳＣＩＩ （ただし最上位はソフトでマスク）
7）ハンドシェイク・プロトコル	コード・コントロールおよびＣＴＳと ＲＴＳによるハードウェア・コントロール

図４－１　シリアル回路



ＪＰ２，ＪＰ３の------は出荷時の設定です．

保守／廃止

４．３　ＲＳ－２３２－Ｃ端子説明

表４－１

ＰＩＮ　No. (Dサブコネクタ)	信　号　名	出力／入力 ※	内　　容	
2	ＴｘＤ(SD)	Ｏ	シリアル出力端子です．	この信号は，ＪＰ２のジャンパを変えることで機能を逆にできます．
3	ＲｘＤ(RD)	Ｉ	シリアル入力端子です．	
5	ＣＴＳ	Ｉ	ターミナル受信可出力のための入力端子です．ＨＩＧＨレベルでレディとみなします．	同　　上
２０	ＤＴＲ	Ｏ	ＩＥ－７８Ｃ１１が受信可であることを意味する出力端子です．ＨＩＧＨレベルでレディです．	
4	ＲＴＳ	Ｏ	ＩＥ－７８Ｃ１１からリーダに対する転送要求信号（モータＯＮ　ｅｔｃ.）です．ＨＩＧＨレベルで要求出力です． ただし，あとで述べるリーダ・コントロールをコードで行なう場合，常にＬＯＷレベルになります．	
8	ＤＣＤ	Ｉ	ターミナルが送受信可であることを確認する入力端子です．ＩＥ－７８Ｃ１１はこの入力がＨＩＧＨﾚﾍﾞﾙ でないかぎり動作しません．もし，ターミナルがこの信号を出力しておれば接続してもかまいませんがＩＥ－７８Ｃ１１内でＰＵＬＬ　ＵＰしておいてもかまいません． 出荷時はボード内でＨＩＧＨにＰＵＬＬ　ＵＰされチャネル端子へは出力されていません	
1	ＦＧ	―――	フレーム・グランド　　ボードの０Ｖ電源につながっています．	
7	ＳＧ	―――	シグナル・グランド　　同　　上	

※下欄で‘Ｏ’は出力，‘Ｉ’は入力を意味します．

保守／廃止

４．４　シリアル・プロートコルの設定

シリアル・インタフェースのハンドシェイクは，ＩＥコントロール・ボード上のDIP SWITCH DP１（図２－２，図２－６参照）の４～６番で指定できます．

表４－２

No.	ＯＮ　時	ＯＦＦ　時
4	ＣＨ１に対する指定です．ハンドシェイクをＣＴＳとＤＴＲによるハードウェア・コントロールとＸ－ＯＮ（１１Ｈ），Ｘ－ＯＦＦ（１３Ｈ）コードの転送によっても行ないます．もしＸ－ＯＮ，Ｘ－ＯＦＦコードの転送だけを使用する場合，ＣＴＳはＩＥ内部でＰＵＬＬＵＰ，ＤＴＲは接続しないでください．	ＣＨ１に対する指定です．ハンドシェイクをハードウェア・コントロールＣＴＳとＤＴＲのみで行ないます．
5	ＣＨ２に対する指定です．機能はＤＰ１No.４と同じです．	同　　左
6	ＣＨ１，またはＣＨ２のどちらかをリーダ，パンチャとして設定されている端末の転送要求をＲＴＳ信号のＯＮ／ＯＦＦで行ないます．	転送要求をＸ－ＯＮ，Ｘ－ＯＦＦコードで行ないます．

注）上記の設定で，リーダ，パンチャとしてアサインされているチャネルのハンドシェイクをＸ－ＯＮ，Ｘ－ＯＦＦで行なうよう指定した場合，ＩＥコントロール・ボード上のＤＰ１のＳＷ６は必ずＯＮとしリーダのスタート・コントロールはＲＴＳで行なうようにしてください．

保守／廃止

４．５　ハンドシェイク入出力規格

（１）出力規格

ＣＴＳ入力ＯＦＦ時　　；ＯＦＦ時点に出力中のコードのみ送り，次のコードは転送しません．

Ｘ－ＯＦＦ（１３Ｈ）の入力　　；Ｘ－ＯＦＦコード受付け時点から，出力中のコードを含めて最大４バイト出力します．

（２）入力規格

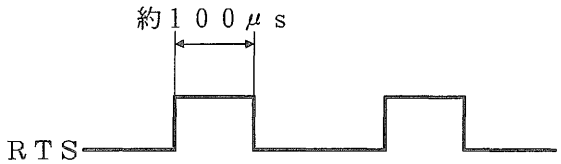
ＤＴＲ出力のＯＦＦ時　　；ＤＴＲ出力がＯＦＦになった後，最大４バイトまでは入力可能．

Ｘ－ＯＦＦ（１３Ｈ）出力時　　；Ｘ－ＯＦＦが出力されたのち，最大１０バイトまでは入力可能．

（３）リーダ規格

ハード信号によるハンドシェイク

ＲＴＳ信号のＯＮ／ＯＦＦによってリーダからデータを１バイト単位で読出す．



Ｘ－ＯＮ，Ｘ－ＯＦＦによるリーダ・コントロール

Ｘ－ＯＮ（１１Ｈ）の出力時

Ｘ－ＯＮ出力が出力された直後から，入力可能です．

Ｘ－ＯＦＦ（１３Ｈ）の出力時

Ｘ－ＯＦＦ出力後，最大４バイトまでは入力可能です．

保守／廃止

一般にコントロール・コードによるハンドシェイクは，タイミング時間の問題のため２～３バイトのから送りの恐れがあります．ＩＥ－７８Ｃ１１はコンソールからの入力に対しては，十分な余裕をもたせていますがＩＥ－７８Ｃ１１からの送出はどうしても最大４バイトは，から送りしてしまう可能性があります．そのためコントロール・コードを利用する場合，コンソールはバッファ機能がありその点を考慮されたものを利用してください．一般に工業用として設計されたＴＴＹ仕様およびＣＲＴはそれが可能なように設計されています．

４．６　ボーレートの指定

ボーレートはＩＥコントロール・ボード上のＤＩＰ　ＳＷＩＴＣＨ　ＤＰ１№１～№３で指定します．（図２－２，図２－６参照）

表４－３　ＤＰ１

ＳＷＩＴＣＨ　Ｎｏ．			内　　容
1	2	3	
ＯＮ	ＯＮ	ＯＮ	ＭＤ－０８６と接続して使用する場合の設定
ＯＮ	ＯＮ	ＯＦＦ	１１０ボー
ＯＮ	ＯＦＦ	ＯＮ	３００ボー
ＯＮ	ＯＦＦ	ＯＦＦ	６００ボー
ＯＦＦ	ＯＮ	ＯＮ	１２００ボー
ＯＦＦ	ＯＮ	ＯＦＦ	２４００ボー
ＯＦＦ	ＯＦＦ	ＯＮ	４８００ボー
ＯＦＦ	ＯＦＦ	ＯＦＦ	９６００ボー

注）　ボーレートは，ＣＨ１，ＣＨ２を別々にはセットできません．そのためＣＨ１，ＣＨ２に別の装置を接続した場合その２つの装置は同じボーレートである必要があります．その端末よりクロックが送られてくる場合，そのクロックがボーレートの１６倍であればＪＰ２，３（図２－４参照）の①－②－⑰の接続を①－②－⑱と変更すれば使用できます．

保守／廃止

４．７　シリアル・インタフェースの設定

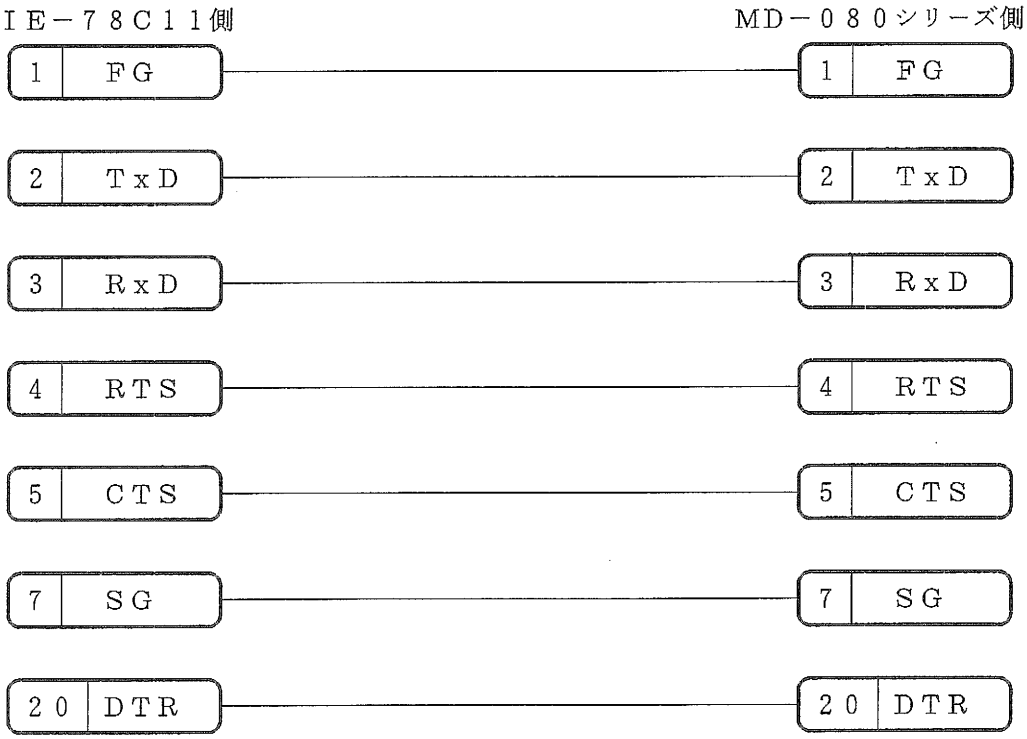
４．７．１　初期設定　（出荷時）

ＩＥ－７８Ｃ１１－Ｃベースでシリアル・ケーブルＩをＭＤ－０８０シリーズのシリアル・チャネルに接続し，ＭＤ－０８０シリーズからキー操作できます．

（ＩＥ－７８Ｃ１１システム・ソフト取扱説明書参照）

添付品のシリアル・ケーブルＩは，両端の同じ端子番号を接続した平行のケーブルになっています．ケーブルは，下記の端子が接続されていれば使用できます．

図４－２　ケーブル接続



保守／廃止

４．８　その他のシリアル・インタフェース

現在，市販されているＲＳ－２３２－Ｃ規格のターミナルのうち，次の３種類

①シチズン時計株式会社　　プロタイパー　ＭＯＤＥＬ７６５２

②アンリツ株式会社　　キャラクタ・ディスプレィ・ターミナル
ＤＤＹ８６，ＤＤＹ－８７０

③カシオ計算機株式会社　　タイピュータ　ＭＯＤＥＬ６５０

とのインタフェースについて説明します．

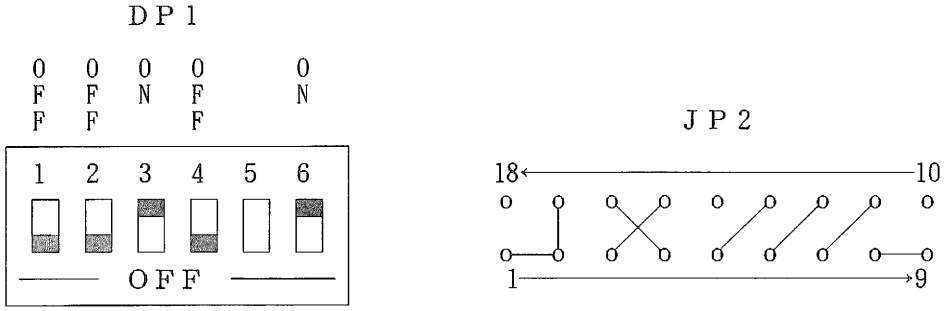
以下ＩＥ－７８Ｃ１１のＩＥコントロール・ボード上のＤＩＰ　ＳＷＩＴＣＨ（ＤＰ１）　ジャンパＪＰ２の設定とＲＳ－２３２－Ｃケーブルの接続方法について示します．

４．８．１　シチズン・プロタイパー・モデル７６５２タイプとのインタフェース

（１）すべてハードウェア・ハンドシェイクのとき

（ハンドシェイクをＣＴＳとＤＴＲで行ないパンチャ，リーダに対してはＲＴＳを使用する時の設定）

図４－３　ＩＥ－７８Ｃ１１設定

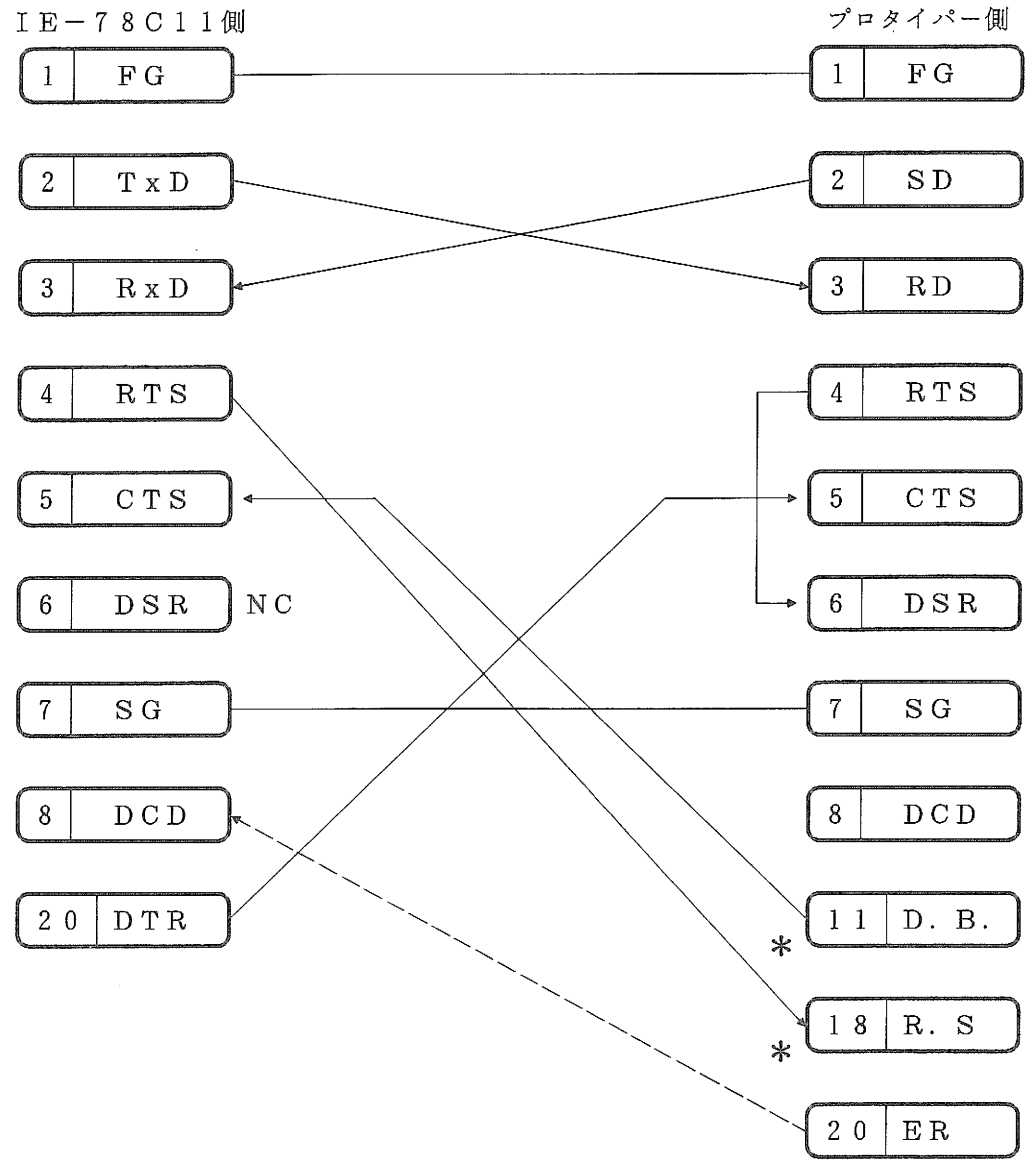


５番は不定（ＣＨ２に対する指定）

ボーレートは，４８００ボーとした場合を示します．

保守／廃止

図４－４　ケーブル接続

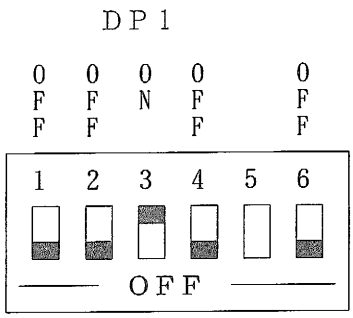


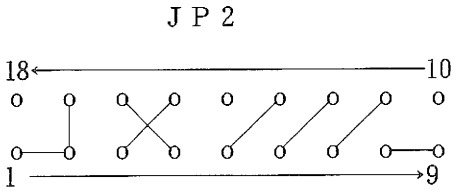
←－－－－－－を接続した場合
ＪＰ２の８－９を
８－１３と変更する．

＊はプロタイパー専用です．

（２）リーダをコードでコントロールするとき

図４－５　ＩＥ－７８Ｃ１１設定

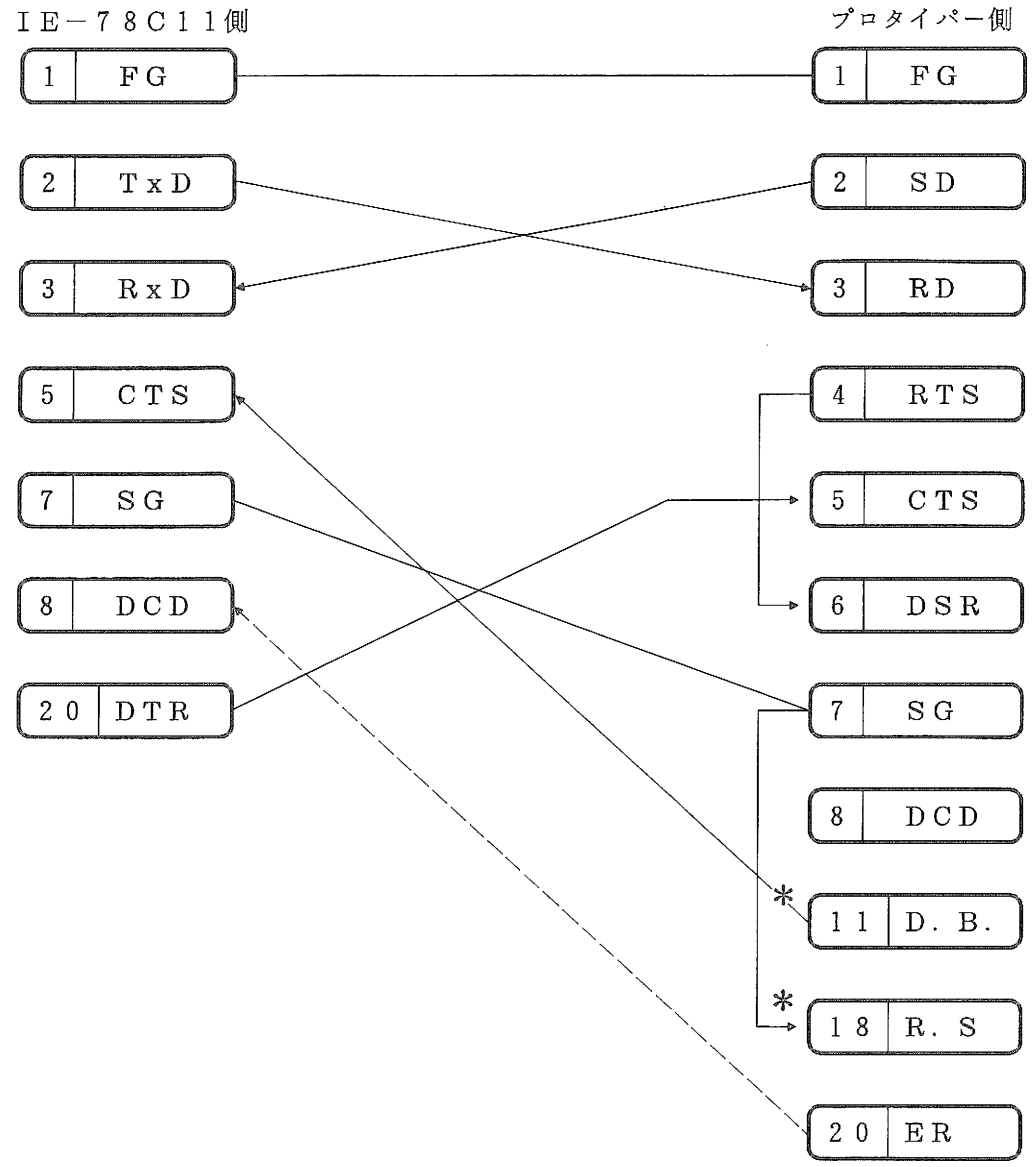




５番は不定（ＣＨ２に対する指定）

保守／廃止

図４－６　ケーブル接続



←－－－－－－を接続した場合
ＪＰ２の８－９を
８－１３と変更する．

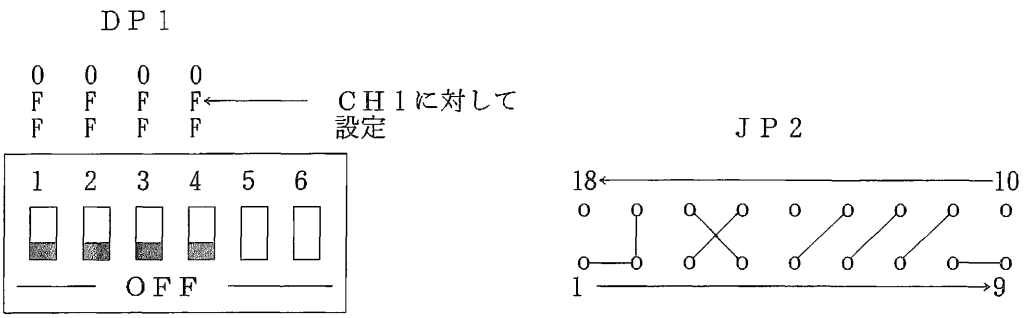
＊はプロタイパー専用です．

保守／廃止

４．８．２　アンリツ製ＤＤＹ８６シリーズとのインタフェース

ここでは，ボーレート９６００ボーでＣＨ１に接続した場合を示します．

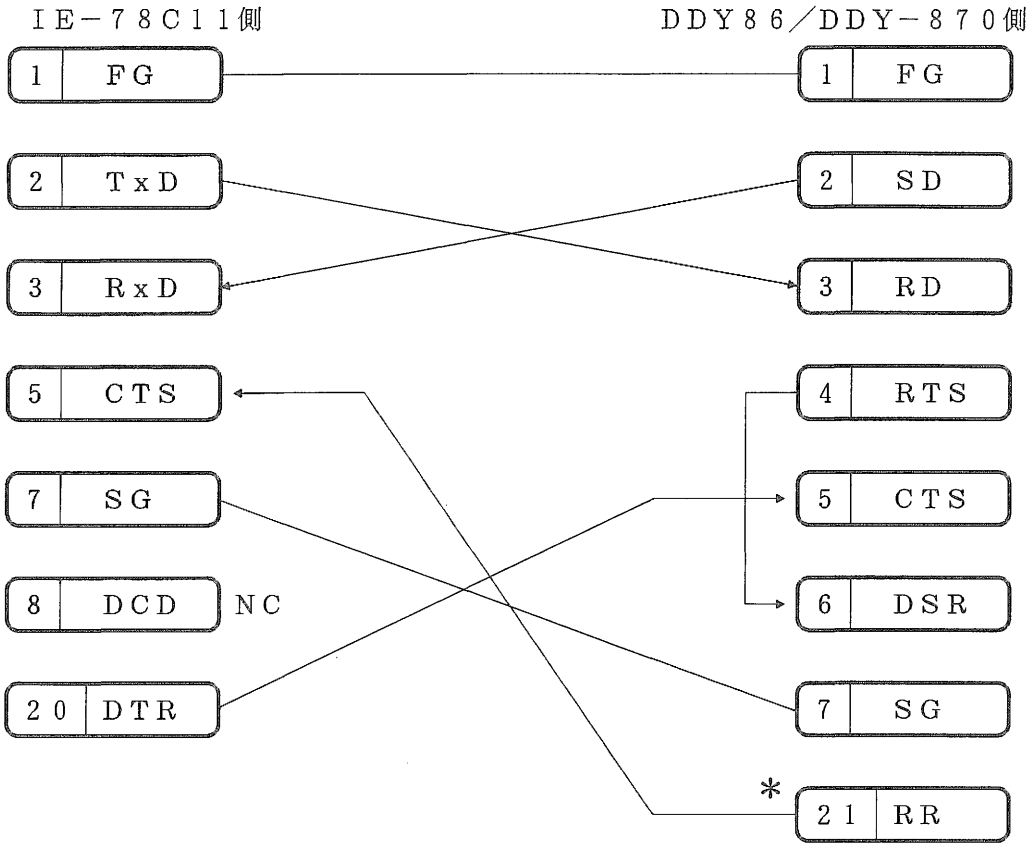
図４－７　ＩＥ－７８Ｃ１１設定



５，６番については場合に応じて設定してください．

保守／廃止

図４－８　ケーブル接続



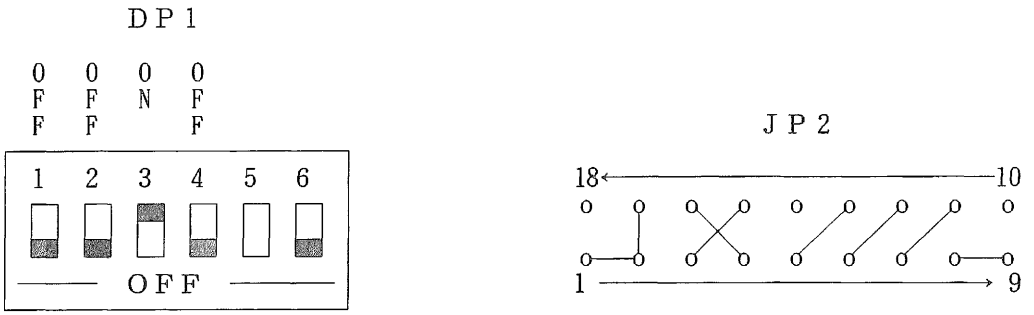
＊はＤＤＹ８６Ｄ専用です．

保守／廃止

４．８．３　カシオ・タイピュータ・モデル６５０とのインタフェース

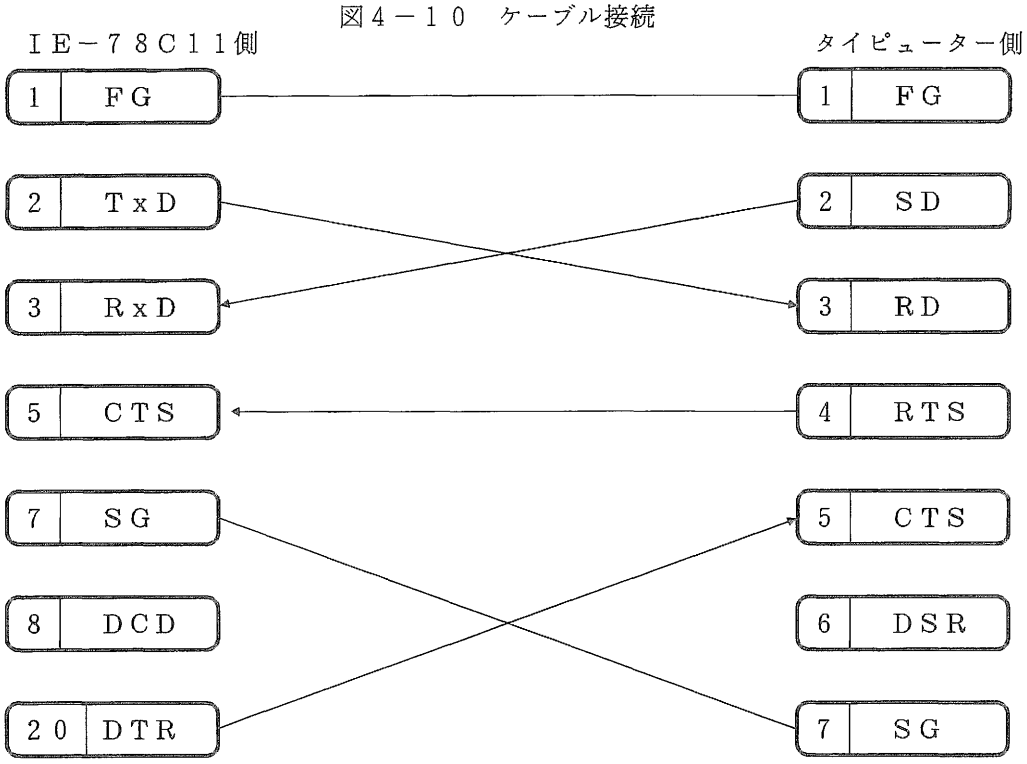
ボーレート４８００ボーでＣＨ１に接続した場合を示します．

図４－９　ＩＥ－７８Ｃ１１設定



５番はＣＨ２の設定

図４－１０　ケーブル接続



保守／廃止

第５章　注意事項

５．１　ＩＥ－７８Ｃ１１とμＰＤ７８Ｃ１１／７８Ｃ１０／７８Ｃ１４との違い

ＩＥ－７８Ｃ１１とμＰＤ７８Ｃ１０／７８Ｃ１１／７８Ｃ１４では，以下の点が異なります。

５．１．１　MMレジスタ，ＭＦレジスタの設定

ＩＥ－７８Ｃ１１では，メモリ・マッピング・レジスタ（ＭＭ）とモードＦレジスタ（ＭＦ）の設定を起動時に行いますので，この設定をプログラムの設定と同じにしてください。

メモリ・マッピングやポートＤ，Ｆの設定をダイナミックに変化させるようなプログラムのエミュレーションはできません。

５．１．２　ポートＤ，Ｆのエミュレーション

ＩＥでディバグする場合と本チップでプログラムする場合とで記述の方法が異なります。

本チップの場合

```
①MVI　PF，××H
　MVI　A，00H
②　MOV　MF，A
```

ＩＥ－７８Ｃ１１で上記プログラムを実行した場合，①を実行してもこの段階ではＥＶＡＣＨＩＰのポートＦは出力モードになっておりませんので，ＩＥ－７８Ｃ１１の出力ラッチには出力データが設定されません。したがって，②の命令を実行した場合にＩＥ－７８Ｃ１１のポートＦには不定なデータが出力されてしまいます。

ＩＥ－７８Ｃ１１でμＰＤ７８Ｃ１０／７８Ｃ１１／７８Ｃ１４の出力ポートのエミュレーションを行う場合には，以下のように設定をお願いします。

保守／廃止

```
MVI  PF, ××H
MVI  A, ××H
MOV  MF, A
①MVI  PF, ××H
```

①の命令を追加してください。

本チップでは必要ありませんが，入っていても問題ありません。

ただし，この場合にもポートの出力設定を実行してから，ポートの出力命令（ＭＶＩ ＰＦ，××Ｈ）を実行するまでは不定なデータが出力されてしまいますので注意をしてください。

なお，ポートＡ，Ｂ，ＣはＥＶＡＣＨＩＰの端子が直接ＩＥの外部接続端子に接続されていますのでこのような問題はありません。

図５－１　μＰＤ７８Ｃ１０／７８Ｃ１１／７８Ｃ１４　ポートＦの等価回路の一部

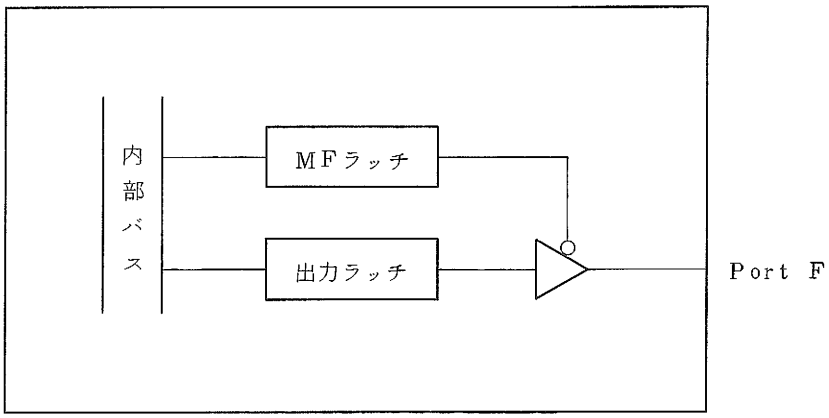
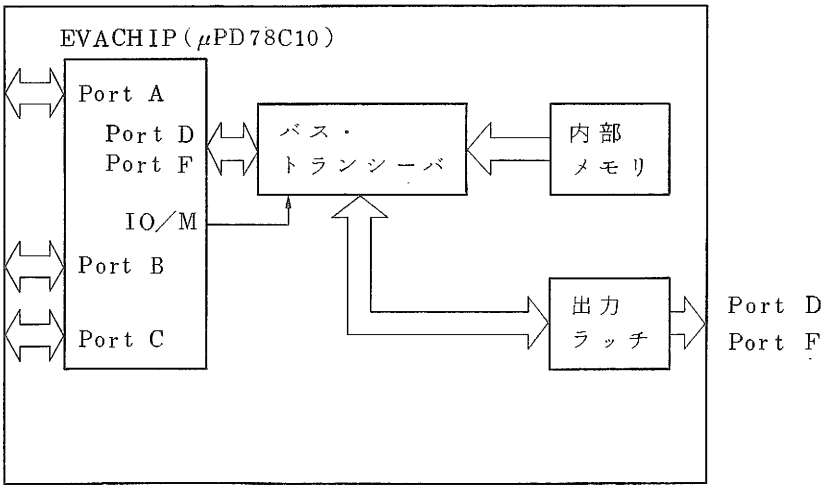


図５－２　ＩＥ－７８Ｃ１１　ポートＤ，Ｆのエミュレーション



保守／廃止

５．１．３　内部ＲＡＭ空間アクセス

ＩＥ－７８Ｃ１１では，内部ＲＡＭ空間をアクセスする際，ＭＭレジスタのＲＡＥビットのセット／リセットに関係なくアクセスすることができてしまいます。

本チップの場合は，ＲＡＥビットをセットしたときのみ内部ＲＡＭ空間をアクセスすることができます。

５．１．４　ハードウェアＳＴＯＰ機能

ハードウエアＳＴＯＰの機能のタイミングが本チップと異なります。

本チップの場合はＳＴＯＰ端子にｌｏｗレベルを入力してからｔｙｐｉｃａｌで３μｓｅｃ後にポート等Ｈｉ－ｚになりますが，ＩＥでは，ＳＴＯＰ端子にｌｏｗを入力するとただちにＨｉ－ｚとなってしまうため，ＥＶＡＣＨＩＰが動作している間おかしな動作をする可能性があります（ＩＥ上のメモリで動作しているときは問題ありませんが，ユーザ・メモリ上で動作しているとき問題となります)。

５．１．５　ポートＤ，Ｆのエミュレーション機能

μＰＤ７８Ｃ１０で使用する場合，ポートＤはアドレス／データ・バスに，ポートＦはアドレス・バスになります。

この場合，μＰＤ７８Ｃ１０はポートＤ，Ｆのエミュレーション機能をもっており，外部の簡単な回路により外部にポートＤ，Ｆを付加することができます。

しかしながら，本エミュレータＩＥ－７８Ｃ１１ではこの機能を持っておりません。したがいまして，本エミュレータは，このようなアプリケーションでは動作致しませんのでご注意ください。

保守／廃止

５．２　μＰＤ７８Ｃ１４サポート・モニタ機能

本製品ＩＥ－７８Ｃ１１は，従来のμＰＤ７８Ｃ１１／７８Ｃ１０に加えて，新たにμＰＤ７８Ｃ１４にも対応するモニタ・プログラムを搭載しております。この新モニタ・プログラムで追加された機能の取り扱い方法について以下に説明します。

５．２．１　ＩＥ－７８Ｃ１１起動時の初期設定

(1) ＭＯＤＥ０，ＭＯＤＥ１端子の設定

ＩＥ－７８Ｃ１１は電源投入時にユーザ・システムのＭＯＤＥ０，ＭＯＤＥ１端子の状態を読み込み，その値によりＣＰＵのモードを設定します。

したがって，ユーザ・システム上でＭＯＤＥ０，ＭＯＤＥ１端子を希望の設定にプルアップ，ダウンする必要があります。

ＭＯＤＥ１	ＭＯＤＥ０	モ　ー　ド
0	0	μＰＤ７８Ｃ１０ 外部４Ｋバイト
0	1	μＰＤ７８Ｃ１０ 外部１６Ｋバイト
1	0	μＰＤ７８Ｃ１１または μＰＤ７８Ｃ１４
1	1	μＰＤ７８Ｃ１０ 外部６４Ｋバイト

ユーザ・システムがない場合，ドライバ上の自己診断用コネクタに，エミュレーション・プローブを挿入することにより，ソフトウエア・ディバグが行えます。

しかし，この場合，ＭＯＤＥ１，ＭＯＤＥ０端子は，共に‘１’となり，μＰＤ７８Ｃ１０で外部６４Ｋバイト・モードとなります。また，ポートＡ，Ｂ，Ｃが内部で接続されているため，ポートＡ，Ｂ，Ｃを共に出力モードとして使用することができません。

このような制限を受けたくない場合，エミュレーション・プローブをどこにも接続しない状態で使用します。[注] この状態で電源投入を行うと，タイトル・メッセージを出力した後，‘ＵＳＥＲ－ＳＹＳＴＥＭ　ＶＤＤ－ＯＦＦ’のメッセージを出力し停止します。この時，ＥＳＣキー入力を行うと‘ＭＯＤＥ０＝’と出力しますので‘０’か‘１’を入力します。次に‘ＭＯＤＥ１＝’と出力しますので同様に‘０’か‘１’を入力してください。

これで任意のモードに設定されます。

例

```
IE-78C11 MONITOR V2.1 [19 MAR 86]
   COPYRIGHT (C) 1985 NEC CORPORATION

USER-SYSTEM VDD-OFF ←　ＥＳＣキー入力

SELECT MODE
MODE0=0
MODE1=0

CPU 78C10
EXTERNAL MEMORY SIZE 4K BYTE
RAE (E:ENABLE,D:DISABLE)=
```

注　ハードウエア・バージョンがＡのものは，使用できません。

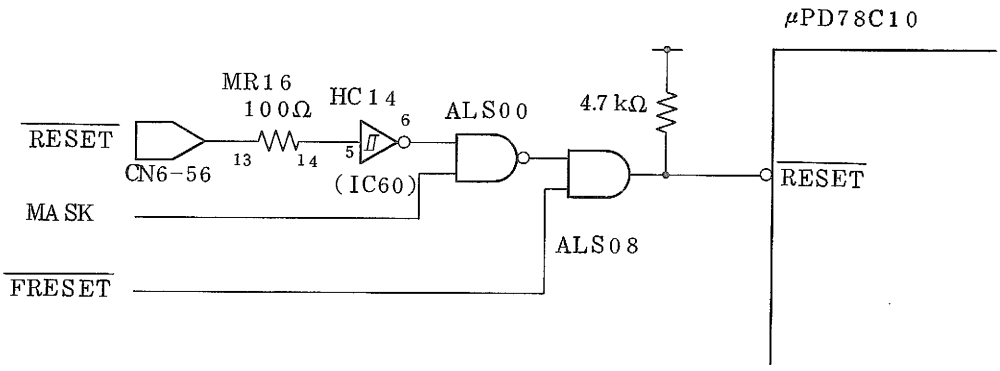
ハードウエア・バージョンは，コントロール／トレース・ボードのＣＨ１コネクタの横に捺印された１０桁のシリアル・ナンバの左から２番目のアルファベットで表されます。

Ｅ×××××××４
　└→ハードウエア・バージョン

ハードウエア・バージョンＡについて

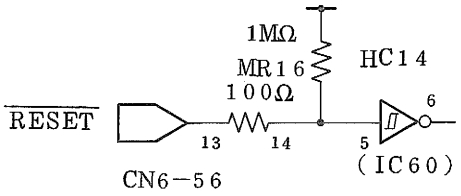
ＩＥのＲＥＳＥＴ回路が図５－３に示す回路構成であるため，１の例のとおり実行できません。図５－４または図５－５の処置を施すことにより実行可能となります。

図５－３　ハードウエア・バージョンＡ



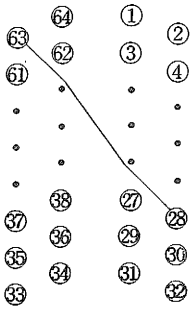
保守／廃止

図５－４　ハードウエア・バージョンＢ



MR16とHC14（IC60）との間を1MΩでプルアップする

図５－５　エミュレーション・プローブ



エミュレーション・プローブの63pinと28pinをジャンパ線で接続する．

(2) ＣＰＵおよび外部拡張メモリ容量の指定

モード端子をＭＯＤＥ０＝０，ＭＯＤＥ１＝１に設定すると初期設定時にモニタが，‘ＳＥＬＥＣＴ　ＣＰＵ（７８Ｃ１１：１／７８Ｃ１４：４）＝’とメッセージを出力しますので‘１’か‘４’を入力してＣＰＵの選択を行ってください。

‘１’を入力するとアドレス０００００－０ＦＦＦＨを‘４’を入力するとアドレス０００００－３ＦＦＦＨをエミュレーションＲＯＭとして割り当てます。

その後，モニタが外部拡張メモリ・サイズを尋ねてきますので容量の指定を行ってください。

例

```
IE-78C11 MONITOR V2.1 [19 MAR 86]
   COPYRIGHT (C) 1985 NEC CORPORATION

USER-SYSTEM VDD-ON

SELECT CPU(78C11:1/78C14:4)= 4  ←  ７８Ｃ１４を選択
EXTERNAL MEMORY SIZE (0:NON,1:256,2:4K,3:16K,4:48K)= 4  ←外部拡張
RAE (E:ENABLE,D:DISABLE)= E                              48Kﾊﾞｲﾄ

*
```