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。

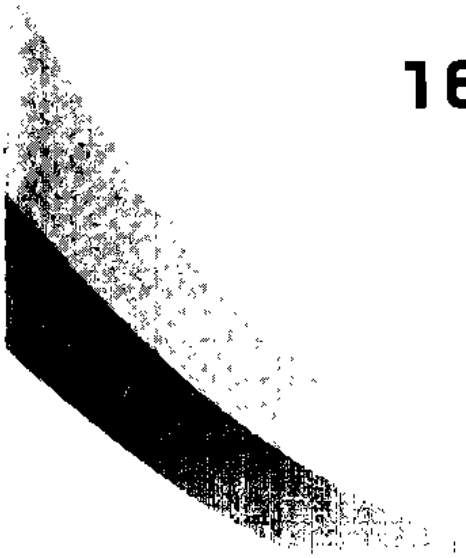
アプリケーション・ノート

RENESAS

保守／廃止

V53A™

16ビット・マイクロプロセッサ



アドレス拡張ソフトウエア編

μPD70236A

資料番号　U10188JJ2V0AN00（第2版）
（旧資料番号　IEA-763）
発行年月　July 1995 P

© NEC Corporation 1994

保守／廃止

保守／廃止

CMOSデバイスの一般的注意事項

①静電気対策（MOS全般）

注意　MOSデバイス取り扱いの際は静電気防止を心がけてください。

MOSデバイスは強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には，NECが出荷梱包に使用している導電性のトレーやマガジン・ケース，または導電性の緩衝材，金属ケースなどを利用し，組み立て工程にはアースを施してください。プラスチック板上に放置したり，端子を触ったりしないでください。

また，MOSデバイスを実装したボードについても同様の扱いをしてください。

②未使用入力の処理（CMOS特有）

注意　CMOSデバイスの入力レベルは固定してください。

バイポーラやNMOSのデバイスと異なり，CMOSデバイスの入力に何も接続しない状態で動作させると，ノイズなどに起因する中間レベル入力が生じ，内部で貫通電流が流れて誤動作を引き起こす恐れがあります。プルアップかプルダウンによって入力レベルを固定してください。また，未使用端子が出力となる可能性（タイミングは規定しません）を考慮すると，個別に抵抗を介してV_{DD}またはGNDに接続することが有効です。

資料中に「未使用端子の処理」について記載のある製品については，その内容を守ってください。

③初期化以前の状態（MOS全般）

注意　電源投入時，MOSデバイスの初期状態は不定です。

分子レベルのイオン注入量等で特性が決定するため，初期状態は製造工程の管理外です。電源投入時の端子の出力状態や入出力設定，レジスタ内容などは保証しておりません。ただし，リセット動作やモード設定で定義している項目については，これらの動作ののちに保証の対象となります。

リセット機能を持つデバイスの電源投入後は，まずリセット動作を実行してください。

V53Aは日本電気株式会社の商標です。

保守／廃止

○本資料の内容は，後日変更する場合があります。
○文書による当社の承諾なしに本資料の転載複製を禁じます。
○この製品を使用したことにより，第三者の工業所有権等にかかわる問題が発生した場合，当社製品の構造製法に直接かかわるもの以外につきましては，当社はその責を負いませんのでご了承ください。
○当社は品質，信頼性の向上に努めていますが，半導体製品はある確率で故障が発生します。当社半導体製品の故障により結果として，人身事故，火災事故，社会的な損害等を生じさせない冗長設計，延焼対策設計，誤動作防止設計等安全設計に十分ご注意願います。
○当社は，当社製品の品質水準を「標準水準」，「特別水準」およびお客様に品質保証プログラムを指定して頂く「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の上ご使用願います。

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

特別水準：輸送機器（自動車，列車，船舶等），交通用信号機器，防災／防犯装置，各種安全装置，生命維持を直接の目的としない医療機器

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，生命維持のための装置またはシステム等

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。当社製品を上記の「標準水準」の用途以外でご使用をお考えのお客様は，必ず事前に当社販売窓口までご相談頂きますようお願い致します。
○この製品は耐放射線設計をしておりません。

保守／廃止

本版で改訂された主な箇所

箇　所	内　容
p. 61, 63	3.4.4（1）　アセンブラ・ソース　コメント修正

巻末にアンケート・コーナを設けております。このドキュメントに対するご意見をお気軽にお寄せください。

保守／廃止

はじめに

対象者　このマニュアルは，μPD70236A（別名称V53A）の機能を理解し，それを用いたアプリケーション・システムの設計をするユーザを対象とします。

目　的　このマニュアルでは，実際にμPD70236Aを用いたプログラム例を取り上げ，ユーザに理解していただくことを目的としています。

構　成　このマニュアルは，大きく分けて次の内容で構成しています。

- アドレス空間拡張機能を使用したプログラム
- 拡張アドレス空間のコード領域
- 拡張アドレス空間のデータ領域
- 付録

読み方　このマニュアルの読者には，電気，論理回路，およびマイクロコンピュータに関する一般的知識を必要とします。
なお，このマニュアルでは，μPD70236Aという製品名を「V53A」の名称で，V53Aを用いたシステム例を「V53Aボード」の名称で統一して説明してあります。

- V53Aの機能をすでに理解しているユーザ
  →目次に従ってお読みください。

- 一通り，V53Aの機能および応用例を理解しようとするとき
  →まず，ユーザーズ・マニュアルをお読みください。
  次にこのアプリケーション・ノートをお読みください。

凡　例

データ表記の重み	：左が上位桁，右が下位桁
アクティブ・ロウの表記	：$\overline{\times\times\times}$（端子，信号名称の上に上線）
メモリ・マップのアドレス	：上部-上位，下部-下位
注	：本文中につけた注の説明
注意	：気をつけて読んでいただきたい内容
備考	：本文の補足説明
数の表記	：2進数…××××または××××B
	10進数…××××
	16進数…××××H

保守／廃止

まぎらわしい文字　　：0（ゼロ）↔O（オー）

1（イチ）↔ I（アイ）↔ l（エル）

レジスタのビットの表記　：ー…任意（設定値は意味を持ちません）

×…任意（設定値は意味を持ちますが，本ボードの説明には関係ありません）

2のべき数を示す接頭語（アドレス空間，メモリ容量）：

K（キロ）　：2^{10} = 1024

M（メガ）　：2^{20} = 1024^2

関連資料　・V53Aに関する資料

製品名		資料名		資料番号
μPD70236A（V53A）		データ・シート		U10120J
		ユーザーズ・マニュアル		U10108J
		レジスタ活用表		IEM-5604
		アプリケーション・ノート ハードウエア設計編		IEA-752
		アプリケーション・ノート アドレス拡張ソフトウエア編		このマニュアル
16ビットVシリーズ		ユーザーズ・マニュアル　命令編		IEU-804
リアルタイムOS	RX136	ユーザーズ・マニュアル	基礎編	EEU-707
			テクニカル編	EEU-719
			ニュークリアス・インストレーション編	EEU-775
インターツール	RA70116-I SP70116-I	ユーザーズ・マニュアル	言語編	EEU-861
			操作編	EEU-869
マルチタスク・ディバッガ	RX-DB116, 136	アプリケーション・ノート ターゲット依存部移植の手法		EEU-609

保守／廃止

目　　次

保守／廃止

図の目次

表の目次

第１章　アドレス空間拡張機能を使用したプログラム

１．１　アドレス空間拡張機能

V53Aはアドレス空間拡張機能を搭載しており，最大16 Mバイトまでアクセスできます。ここではアドレス空間拡張機能について説明します。

通常モードで得られる1 Mバイト（00000H-FFFFFH）のアドレス空間を基本アドレス空間，拡張モードで得られる16 Mバイト（000000H-FFFFFFH）のアドレス空間を拡張アドレス空間として説明します。

１．１．１　拡張方式

拡張方式としては，基本アドレス空間に拡張アドレス空間を実際に存在するかのようにマッピングするアドレス・リロケーション方式を採用しています。次の例は基本アドレス空間（80000H-8FFFFH）の64 Kバイトの空間に，拡張アドレス空間（200000H-20FFFFH）をリロケートした場合を示しています。

例

```
_TEST   Segment  at  8000H                          ;①
_TEST   Ends
            .
            .
        MOV  Word  Ptr  _TEST:[0000H],0AAAAH ;②
```

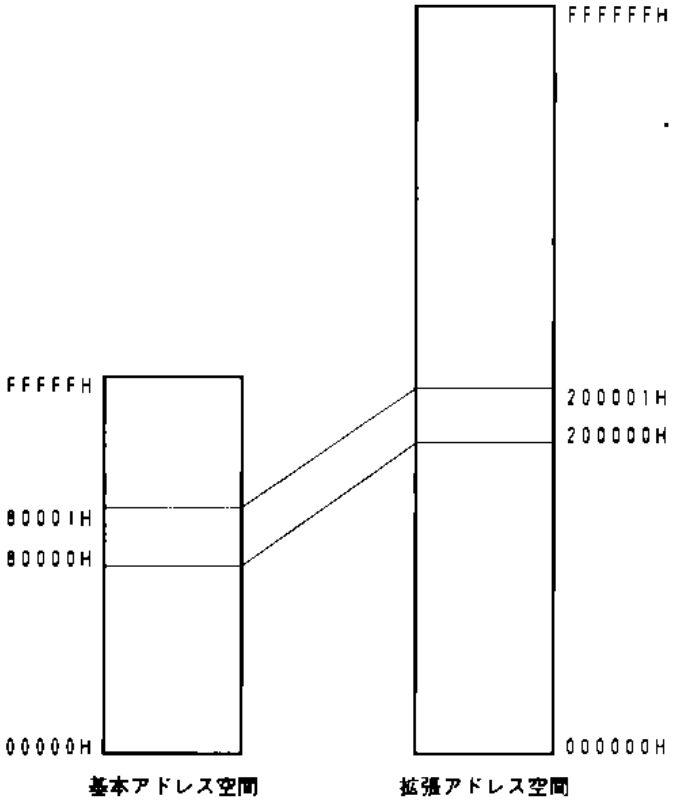
①のようにセグメント_TESTを設定した場合，通常モードと拡張モードでは②を実行すると次のようになります。

・通常モード：基本アドレス空間（80000H）にデータ（0AAAAH）が書き込まれます。

・拡張モード：拡張アドレス空間（200000H）にデータ（0AAAAH）が書き込まれます。

保守／廃止

図1－1　アドレス・リロケーション（データの書き込まれ方）



保守／廃止

１．１．２　アドレス・リロケーション

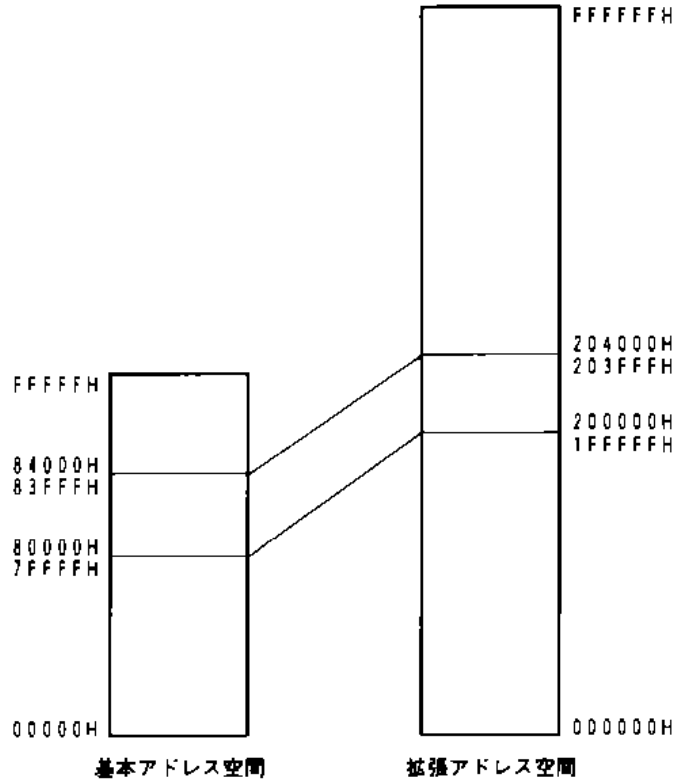
V53Aはページ・レジスタ（PGR1-PGR64）により、16 Kバイトごとにリロケートできます。

ページ・レジスタはI/O空間（FF00H-FF7EH）にマッピングされ、A14-A19により選択されます。

ページ・レジスタ（PGR1-PGR64）は下位10ビット（bit0-9）が有効で、リロケート・アドレスの上位10ビット（A14-A23）を設定します。

たとえば、基本アドレス空間（80000H-83FFFH）の16 Kバイト空間に、拡張アドレス空間（200000H-203FFFH）をリロケートする場合のページ・レジスタの設定例を次に示します。

図１－２　アドレス・リロケーション（80000H-83FFFH に200000H-203FFFHをリロケート）



（ａ）アセンブラ

```
                PGR33   EQU     0FF40H
;---------------------------------------
                MOV     AX , 0080H
                MOV     DX , PGR33
                OUT     DX , AX
```

保守／廃止

（b）C言語

```
#define         PGR33   0xFF40
/*****************************************/
void    pgr_init(void)|
        outport(PGR33 , 0x0080);

|
```

備考　outport（ ）について説明します。

void　　outport（int *portid*, int *value*）

*portid*で指定された出力ポートに*value*で指定したワード・データを書き込みます。

保守／廃止

１．２　プログラムの書き方

V53Aのアドレス空間拡張機能を使用するための手続きについて説明します。

次の（１）-（５）にアドレス空間拡張機能を使用したプログラムの書き方を示します。

（１）ページ・レジスタの初期化を行います。

（２）基本アドレス空間と拡張アドレス空間のリロケートする空間を確認して，必要なページ・レジスタの設定を行います。

（３）拡張アドレス・モード専用命令（BRKXA, RETXA）実行時に使用する割り込みベクタ・テーブルの設定を行います。

（４）拡張アドレス・モード専用命令BRKXAを実行し，拡張アドレス・モードを設定します（拡張アドレス空間のデータのリード／ライト，コードの実行ができます）。

（５）拡張アドレス・モード専用命令RETXAを実行し，拡張アドレス・モードを解除します。

上記の（１）-（５）について以下に示します。

１．２．１　ページ・レジスタの初期化

次に示す理由によりページ・レジスタPGR1-PGR64の初期化処理を行います。

- ページ・レジスタのリセット時の値が不定
- 拡張アドレス・モード時のリロケートを行っていない下位1 Mバイトのアドレス空間へのアクセスの保障

ページ・レジスタの初期化処理の例を次に示します。

（a）アセンブラ

```
                PGR1    EQU     0FF00H
                PGR64   EQU     0FF7EH
;---------------------------------------
                MOV     AX , 0000H
                MOV     DX , PGR1
next_pgr:       OUT     DX , AX
                INC     AX
                ADD     DX , 2
                CMP     DX , PGR64 + 2
                JB      next_pgr
```

保守／廃止

（b）C言語

```
#define         PGR1    0xFF00
#define         PGR64   0xFF7E
int     io_add , data
/*************************************/
void    pgr_init(void){
        data = 0x0000;
        for(io_add=PGR1 ; (unsined)io_add
                    != (PGR64 + 2) ; IO_ADD += 2){
                outport(io_add , data);
                data++;
        }
}
```

備考　outport（ ）について説明します。

void　outport（int *portid*, int *value*）

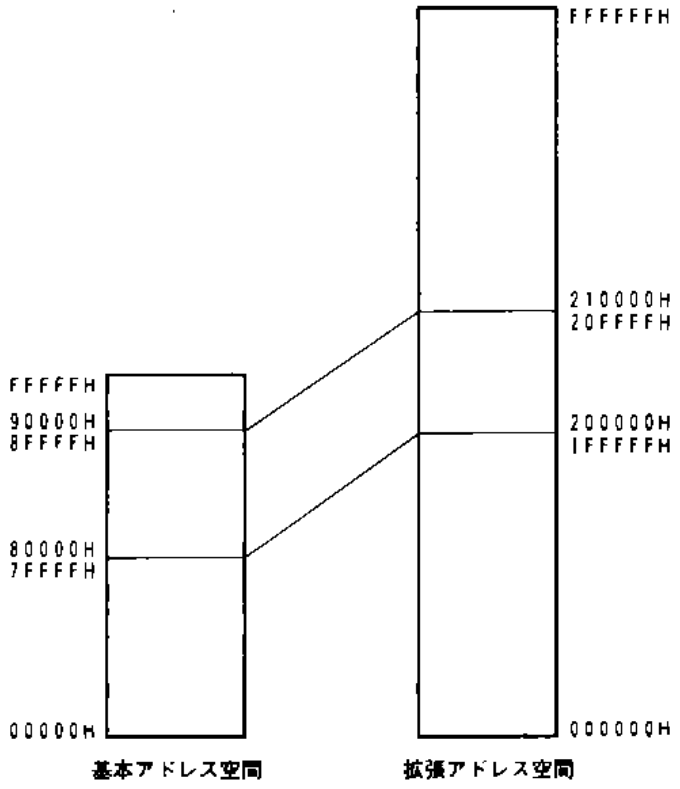
*portid*で指定された出力ポートに*value*で指定したワード・データを書き込みます。

保守／廃止

１．２．２　ページ・レジスタの設定

拡張アドレス空間（200000H-20FFFFH）を基本アドレス空間（80000H-8FFFFH）にリロケートする場合のページ・レジスタの設定例を次に示します。

図１－３　アドレス・リロケーション（80000H-8FFFFHに200000H-20FFFFHをリロケート）



（ａ）アセンブラ

```
                PGR33   EQU     0FF40H
                PGR36   EQU     0FF46H
;---------------------------------------
                MOV     AX , PGR33
                MOV     DX , 0080H
next_pgr:       OUT     DX , AX
                INC     AX
                ADD     DX , 2
                CMP     DX , PGR36 + 2
                JB      next_pgr
```

保守／廃止

（b）C言語

```
#define         PGR33   0xFF40
#define         PGR36   0xFF46
int     io_add , data
/*****************************************/
void    pgr_set(void){
        data = 0x0080;
        for(io_add=PGR33 ; (unsined)io_add
                    != (PGR36+2) ; IO_ADD += 2){
                outport(io_add , data);
                data++;
        }
}
```

1.2.3　拡張アドレス・モードのスイッチング

拡張アドレス・モードの設定／解除は、拡張アドレス・モードをサポートするための専用命令BRKXA/RETXAで行います。

```
BRKXA   imm8    ;(ｺｰﾄﾞ)0000 1111 1110 0000 imm8
RETXA   imm8    ;(ｺｰﾄﾞ)0000 1111 1111 0000 imm8
```

備考　imm8：00H-FFH範囲の定数

BRKXA/RETXAを実行するとimm8で指定したベクタ・テーブルのエントリに格納されているアドレスに制御を移し、拡張アドレス・モードの設定／解除を行います。

また、拡張アドレス・モード専用命令BRKXA/RETXAのサポートされていないアセンブラ、コンパイラを使用する場合、上記で示したコードを使用してBRKXA/RETXAを実行してください。

次に、拡張アドレス・モード専用命令BRKXA/RETXAのサポートされていないアセンブラ／コンパイラを使用し、拡張アドレス・モードでプログラム「FNC」を実行する場合のスイッチング例を示します（仮に、BRKXA/RETXA命令実行時に使用するベクタをそれぞれ254, 253とします）。

保守／廃止

（a）アセンブラ

```
;BRKXA,RETXAについてﾏｸﾛ･ﾌﾟﾛｸﾞﾗﾑを作成し、対応します。
;
BRKXA   Macro   Vect
        DB      0fh , 0e0h , Vect
        Endm
RETXA   Macro   Vect
        DB      0fh , 0f0h , Vect
        Endm
;ﾍﾞｸﾀ･ﾃｰﾌﾞﾙの設定
        MOV     AX , 0
        MOV     DS , AX
        MOV     SI , 254*4
        MOV     Word Ptr DS:[SI] , offset XA1
        MOV     Word Ptr DS:[SI + 2] , seg XA1
        MOV     SI , 253*4
        MOV     Word Ptr DS:[SI] , offset XA2
        MOV     Word Ptr DS:[SI + 2] , seg XA2
;ｽｲｯﾁﾝｸﾞ
        BRKXA   254
XA1:    CALL    FNC
        RETXA   253
XA2:    ;Continue Program
            .
            .
```

（b）C言語

```
/*BRKXA,RETXAについて、ｱｾﾝﾌﾞﾗの命令をCｿｰｽ･ﾌｧｲﾙに*/
/*直接書き込むことのできるｲﾝﾗｲﾝ･ｱｾﾝﾌﾞﾗにより対応*/
/*します。                                          */

/*ﾍﾞｸﾀ･ﾃｰﾌﾞﾙの設定*/
void    vector_init(void){
        poke(0 , 0xfe*4 , FP_OFF(xa))
        poke(0 , 0xfe*4+2 , FP_SEG(xa))
        poke(0 , 0xfd*4 , FP_OFF(retxa))
        poke(0 , 0xfd*4+2 , FP_SEG(retxa))
}
/*ｽｲｯﾁﾝｸﾞ*/
void    brkxa(void){
        asm     db 0fh , 0e0h , 0feh;
}
void    xa(void){
        FNC( );
        asm     db 0fh , 0f0h , 0fdh;
}
void    retxa{
}
```

保守／廃止

備考1．poke（ ）について説明します。

void　　poke（unsigned *segment*, unsigned *offset*, int *value*）

segment, *offset*で指定されたセグメント，オフセットで表されるメモリ空間に*value*で指定したワード・データを書き込みます。

2．FP_OFF, FP_SEGについて説明します。

#define FP_OFF（*FP*）

#define FP_SEG（*FP*）

FPで指定したfarポインタのオフセット，セグメントの設定，取得を行います。

3．関数内でレジスタ保存の操作を行うコンパイラの場合，スイッチングの部分を修正する必要があります。

例としてTURBO C ++（Ver1.01）の場合，（a）に示すCソースのアセンブル・リストを構築したものを（b）に，（b）を参考に修正したスイッチング部のプログラムを（c）に示します。

（a）Cソース・リスト

```
fnc(void){
    .
    .
}
```

（b）TURBO C++（Ver1.01）で構築したアセンブル・リスト

```
_fnc    proc    near
        push    bp
        mov
            .
            .
        pop     bp
        ret
_fnc    end P
```

保守／廃止

（c）修正したスイッチング・プログラム

```
Void    brkxa(Void){
        asm     pop     bp
        asm     db 0fh, 0e0h, 0fdh
}
Void    xa(Void){
        FNC( );
        asm     pop     bp
        asm     db 0fh, 0f0h, 0fdh
}
Void    retxa{
}
```

レジスタ保存の操作についてはコンパイラごとに異なるので、アセンブル・リストを構築するなどして十分な確認を行ってください。

4．使用するコンパイラによってはインライン・アセンブラでDB疑似命令が使用できない場合があります。

たとえば、Microsoft C（Ver6.0）はインライン・アセンブラでDB疑似命令が使用できないので、-emit疑似命令を使用します。-emitは現在のコード・セグメントの現在位置に単独のバイト・データ1つを定義します。-emitが定義できるのは、一度に1バイトだけです。

5．メイン・プログラムでvector_init（ ）を実行したあと、brkxa（ ）を呼び出すと、拡張モードでプログラム「FNC」を実行し、拡張モードを解除してからメイン・ルーチンに制御が戻ります。次の（1）-（4）にスイッチング部分の制御権の移動を示します。

（1）メイン・ルーチン

brkxa（ ）をコール

（2）brkxa（ ）ルーチン

BRKXA命令を実行（asm　　db 0fh, 0e0h, 0feh）

（3）xa（ ）ルーチン

（a）FNC（ ）を実行

（b）RETXA命令を実行（asm　　db 0fh, 0f0h, 0fdh）

（4）retxa（ ）ルーチン

RET命令を実行すると、メイン・ルーチンに制御が戻ります。

保守／廃止

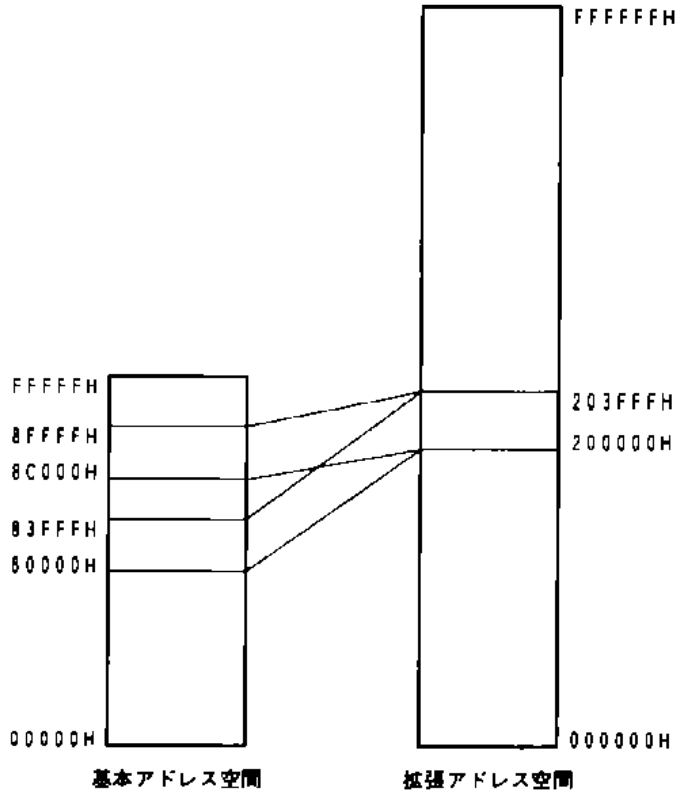
1.3　注意事項

V53Aのアドレス空間拡張機能を使用したソフトウエアを設計する際の注意事項と確認事項について説明します。

（1）複数のページ・レジスタに同じ値を設定した場合，拡張アドレス・モードでデータを書き込むとほかのアドレス空間にも自動的にデータが書き込まれます（同じデータがアドレス空間上に複数存在することになります）。

たとえば，PGR33とPGR36の両方に0080Hを設定した場合，拡張モードで80000Hにワード・データAAAAHを書き込むと，8C000Hにもワード・データAAAAHが自動的に書き込まれます。

図1－4　アドレス・リロケーション（PGR33とPGR36に0080Hを設定）



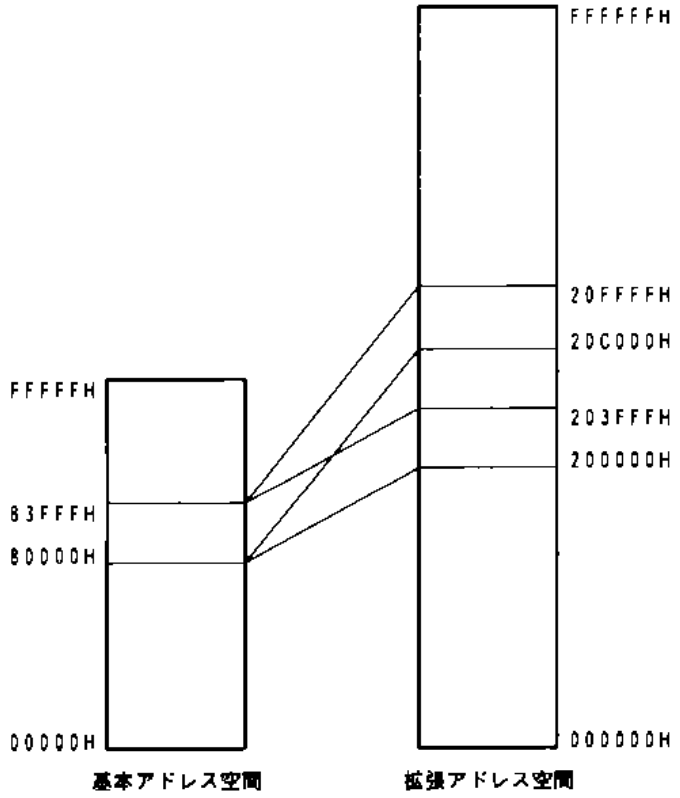
保守／廃止

（2）拡張アドレス・モード時はページ・レジスタをアクセスしないでください。アクセスした場合，そのあとの動作は保証できません。

ある特定の基本アドレス空間に複数ページの拡張アドレス空間をリロケートする場合，一度拡張アドレス・モードを解除してからページ・レジスタを再設定してください。

たとえば，80000H-83FFFHに200000H-203FFFH，20C000H-20FFFFHを連続してリロケートして，アクセスする場合は次のように制御してください。

図1－5　アドレス・リロケーション（PGR33に0080Hと0083Hを設定）



（a）PGR33に0080Hを設定します。

（b）BRKXA命令を実行します。

（c）80000H-83FFFH をアクセスします（200000H-203FFFHをアクセス）。

（d）RETXA命令を実行します。

（e）PGR33に0083Hを設定します。

（f）BRKXA命令を実行します。

（g）80000H-83FFFH をアクセスします（20C000H-20FFFFHをアクセス）。

（h）RETXA命令を実行します。

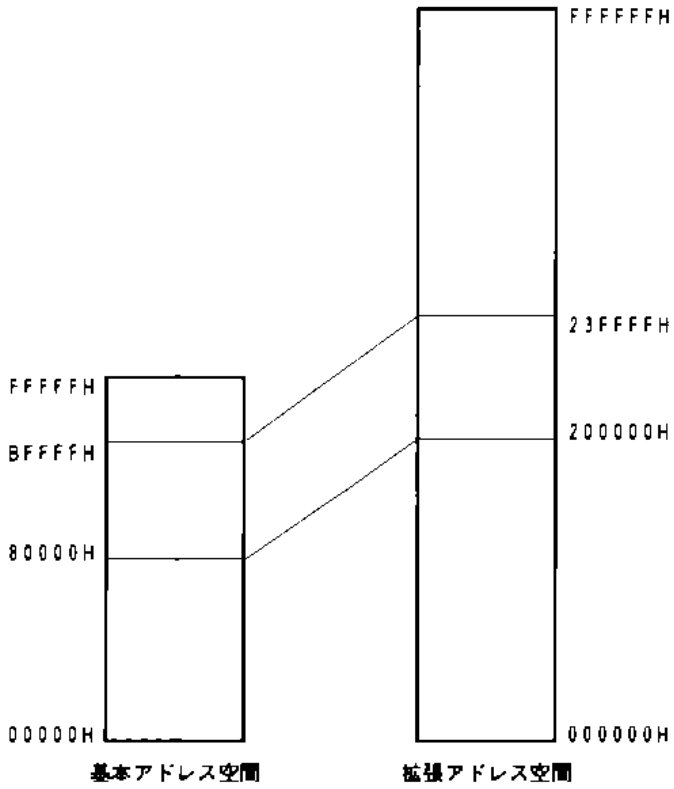
保守／廃止

（３）拡張モードで動作する割り込みルーチンを実行する場合，ページ・レジスタ設定時は割り込みを禁止してください。このとき，拡張アドレス空間のリロケート領域が複数存在する場合，ページ・レジスタの設定手順としては次の２通りあります。

また，設定手順（ａ），（ｂ）の比較を表１－１に示します。

（ａ）リロケートする拡張アドレス空間のページ・レジスタの設定を一度に行う。

図１－６　アドレス・リロケーション（ページ・レジスタの設定を一度に行った場合）



保守／廃止

（b）その時点でアクセスする必要のある拡張アドレス空間の領域を選択してリロケートを行う。

図1－7　アドレス・リロケーション（ページ・レジスタの設定を分割して行った場合）

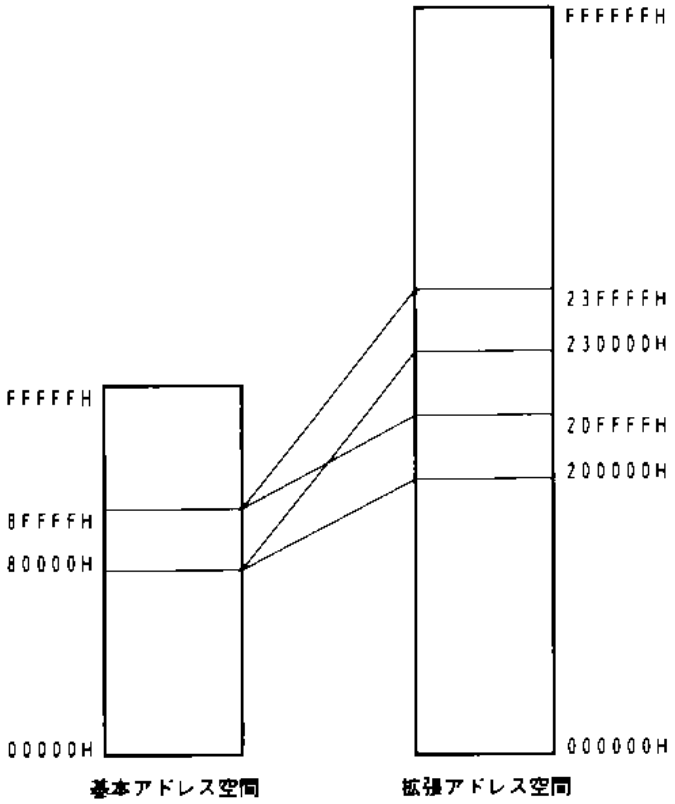


表1－1　設定手順（a）と（b）の比較

比較項目＼設定手順	（a）	（b）
割り込み禁止時間（合計）	短い	長い
フリー・エリア	大	小

備考1．手順（b）は，割り込み禁止時間（トータル）が長くかかりますが，1回に禁止する時間が短いので割り込み応答時間は早くなります。

2．手順（a）を行うと，特定の基本アドレス空間に複数の拡張アドレス空間のブロックをリロケートするので，手順（b）を行った場合よりもフリー・エリアを大きく取ることができます。

（4）拡張モードでメモリをアクセスする場合，20ビットから24ビットへのアドレス変換を行うのに1クロック必要です。したがって，通常モードでプログラム処理を行う場合に比べて処理時間が遅くなることがあります。

（5）ページ・レジスタ（PGR1-PGR64）へは，常に偶数アドレスに対するワード・アクセスを行ってください。

保守／廃止

〔メ　モ〕

保守／廃止

第２章　拡張アドレス空間のコード領域

拡張アドレス空間にプログラムをマッピングする場合について説明します。

２.１　機　　能

拡張アドレス空間をコード領域として利用することにより、大容量または多数のシステムを実行できます。

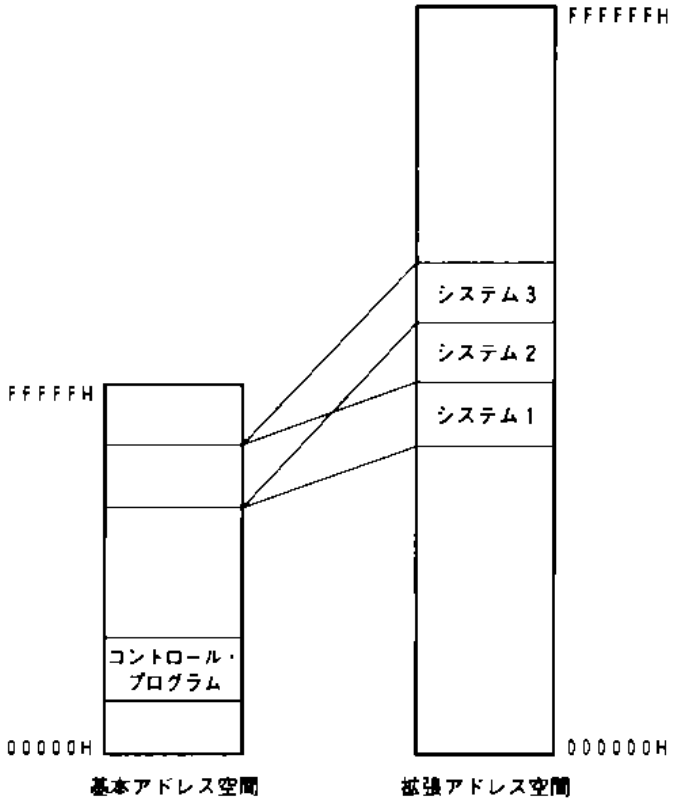
２.２　用 途 例

ワード・プロセッサなどのOA機器で使用するアプリケーション・ソフトのプログラムをマッピングします。

保守／廃止

２．３　マッピング例

図２－１　コード領域としての利用例



備考１．コントロール・プログラム：

ページ・レジスタの設定を行う通常モードで動作するプログラム

２．システム１-３：

下位１Mバイトのアドレス空間にリロケートし拡張アドレス・モードで動作するプログラム

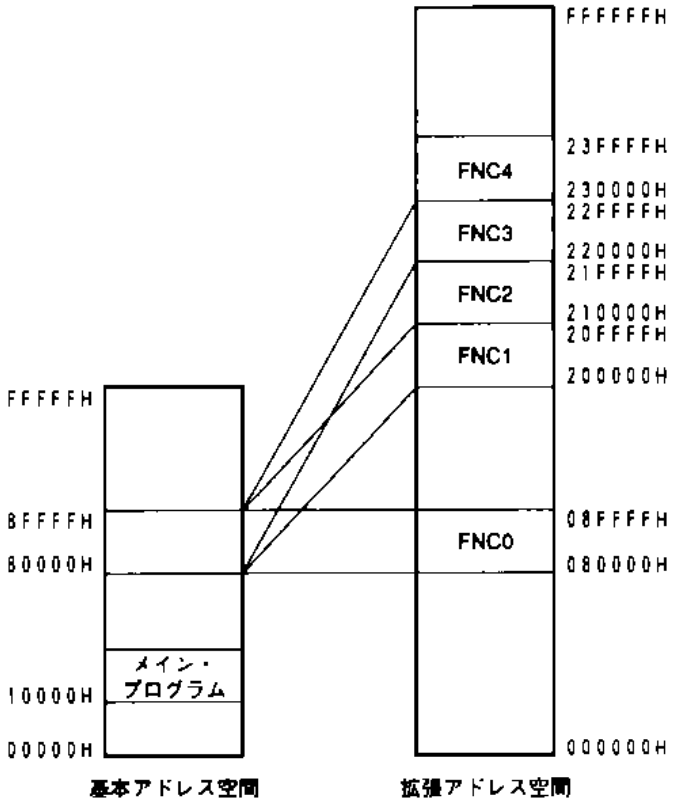
保守／廃止

2.4　プログラム例

拡張アドレス空間をコード領域として利用した場合のV53Aボード上で動作するプログラム例について説明します。

2.4.1　メモリ構成

図2－2　コード・マッピング例



保守／廃止

２．４．２　プログラム処理

各プログラムでは次のような動作を行います。

（１）メイン・プログラム

・通常モードで動作します。

・拡張アドレス・モード専用命令BRKXA, RETXAで使用するベクタ・テーブルの設定、CPUの初期化処理としてV53Aのシステム I/Oレジスタ、ページ・レジスタの初期設定を行います。

そのあと、変数CNTによりFNC0-FNC4の実行を制御します。

（２）FNC0-FNC4

・拡張アドレス・モードで動作します。

・各プログラムとも、文字列出力プログラムです。実行するとそれぞれ表２－１に示す文字列を内蔵SCUに出力します。

表２－１　文 字 列

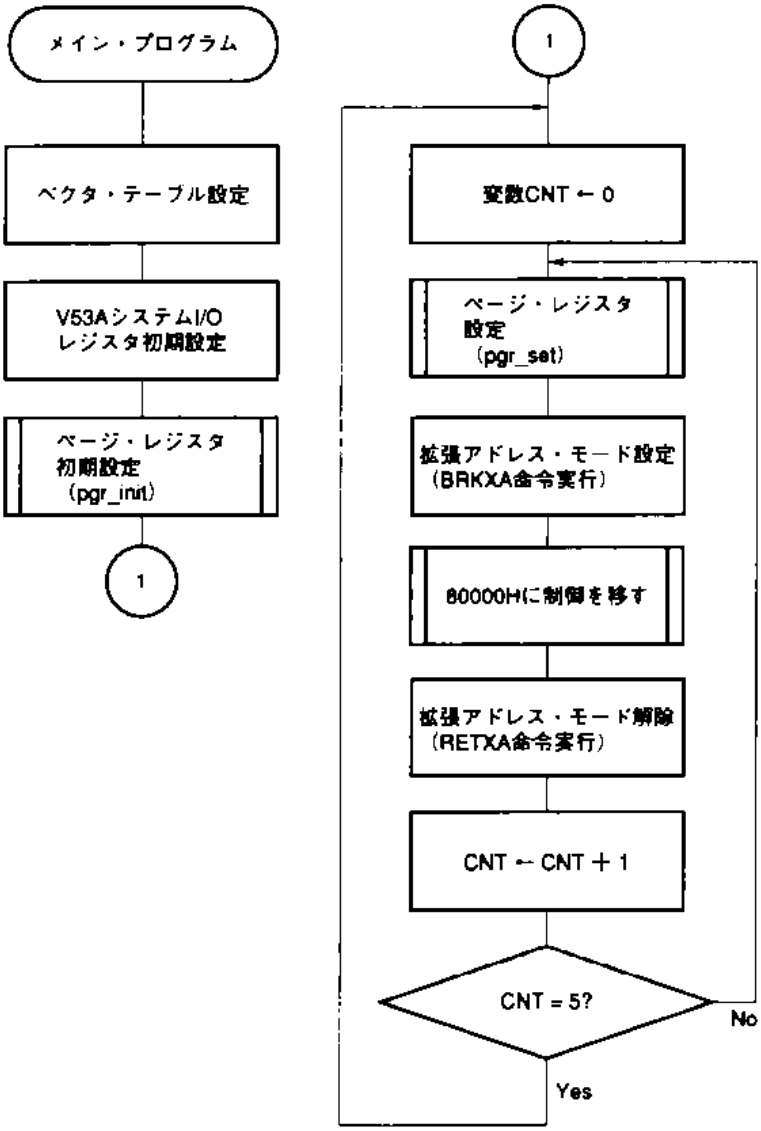
プログラム名	文 字 列
FNC0	* * * Access To Function 0 * * *
FNC1	* * * Access To Function 1 * * *
FNC2	* * * Access To Function 2 * * *
FNC3	* * * Access To Function 3 * * *
FNC4	* * * Access To Function 4 * * *

保守／廃止

2.4.3　フロー・チャート

（1）メイン・ルーチン

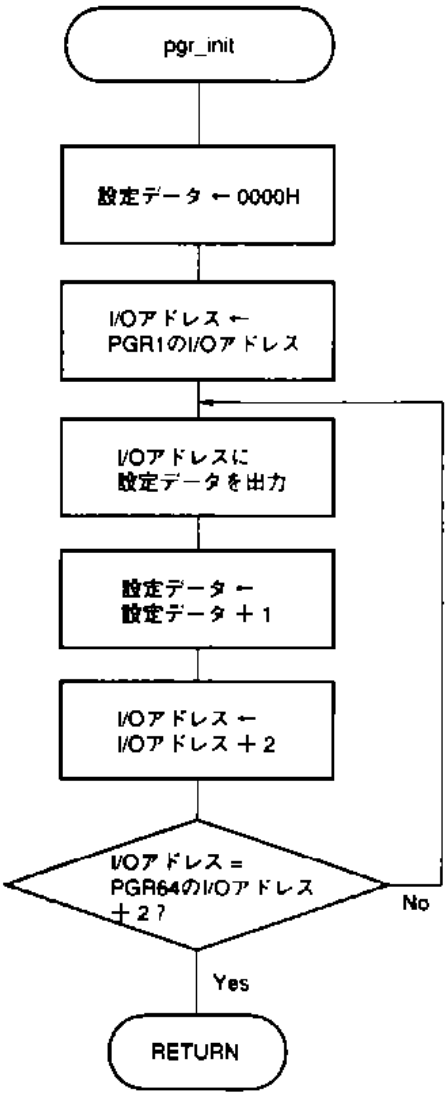
V53Aのイニシャライズ，FNC0-FNC4の実行制御を行います。



保守／廃止

（２）pgr_initルーチン

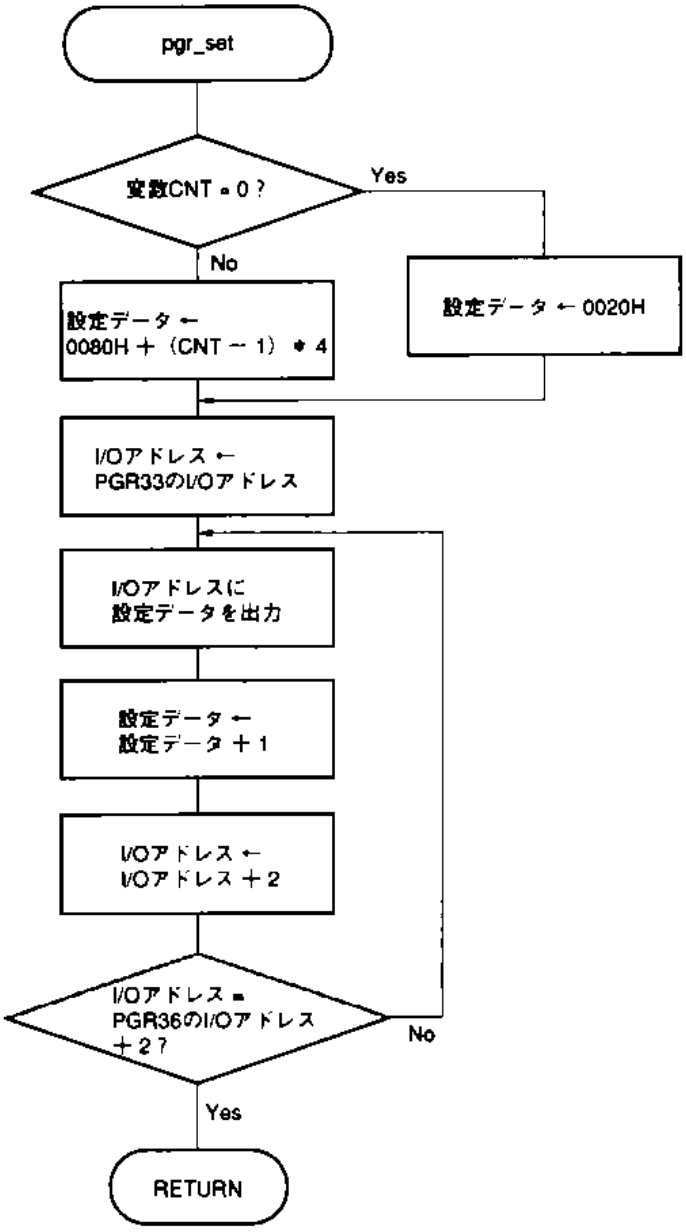
V53Aのページ・レジスタ初期設定を行います。



保守／廃止

（３）pgr_setルーチン

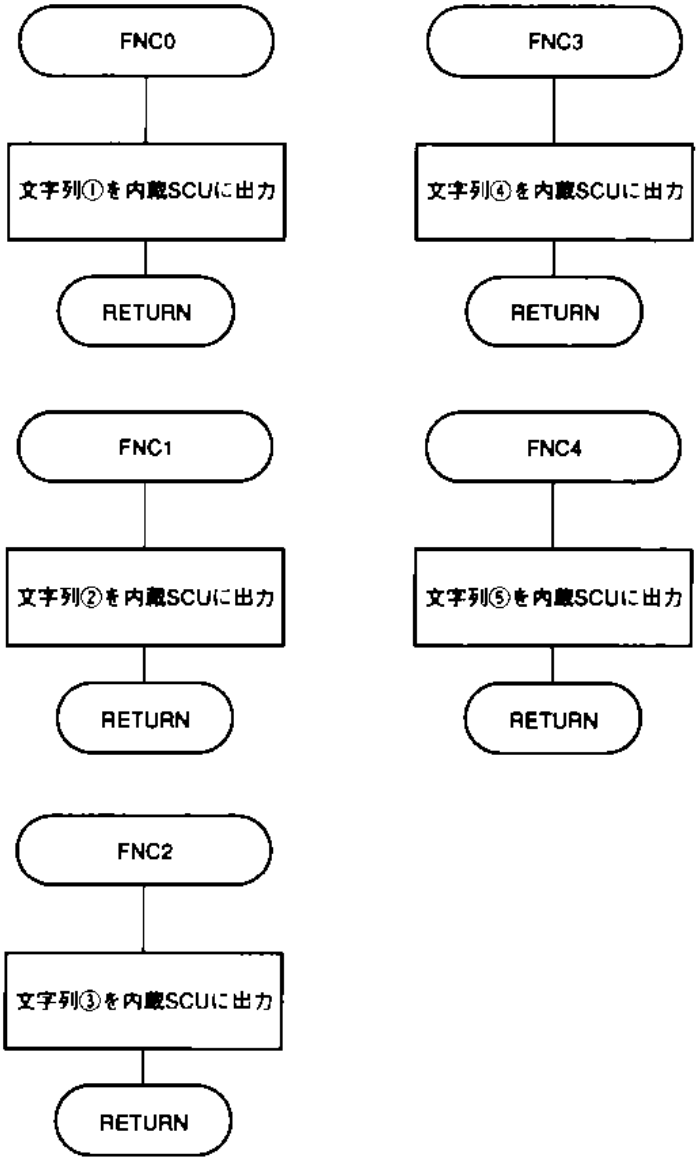
変数CNTの値により基本アドレス空間80000H-8FFFFHをリロケートするのに必要なPGR33-PGR36の設定を行います。



保守／廃止

（４）FNC0-FNC4ルーチン

ページ・レジスタPGR33-PGR36の設定により、FNC0-FNC4を実行します。



	文 字 列
①	＊＊＊ Access To Function 0 ＊＊＊
②	＊＊＊ Access To Function 1 ＊＊＊
③	＊＊＊ Access To Function 2 ＊＊＊
④	＊＊＊ Access To Function 3 ＊＊＊
⑤	＊＊＊ Access To Function 4 ＊＊＊

保守／廃止

2.4.4　ソース・プログラム

（1）アセンブラ・ソース

```
        DOSSEG
;+------------------------------------------------------------+
;|      セグメント定義          Version 1.0  <1990-09-09>      |
;+------------------------------------------------------------+
_TEXT           Segment byte public 'CODE'
_TEXT           Ends

_TEXTEND        Segment para public 'CODE'
Public          ECODE
ECODE           label   byte                ; コード･セグメントの最後
_TEXTEND        Ends

_DATA           Segment byte public 'DATA'
Public          SDATA
SDATA           label   byte                ; データ･セグメントの最初
_DATA           Ends

_DATAEND        Segment byte public 'STACK'
EDATA           label   byte                ; データ･セグメントの最後
_DATAEND        Ends

_STACK          Segment stack 'STACK'       ; スタック･セグメント
_STACK          Ends

CGROUP          GROUP   _TEXT,_TEXTEND
DGROUP          GROUP   _DATA,_DATAEND,_STACK
        Page
```

保守／廃止

```
        .XLIST
;-------------------------------------------------------------------------
SCU_CMD         EQU     0FE22H  ;内蔵ＳＣＵのコマンド・アドレス
SCU_MOD         EQU     0FE24H  ;内蔵ＳＣＵのモード・アドレス
SCU_STT         EQU     0FE22H  ;内蔵ＳＣＵのステータス・レジスタ・アドレス
SCU_RXD         EQU     0FE20H  ;内蔵ＳＣＵのデータ・アドレス
SCU_TXD         EQU     0FE20H  ;内蔵ＳＣＵのデータ・アドレス
SCU_IMK         EQU     0FE26H  ;内蔵ＳＣＵのマスク・レジスタ
;-------------------------------------------------------------------------
;１文字送信　マクロ
MON_STR Macro   addr
        Local   _str01 , _str02 , _str03
                MOV     SI,addr
_str01:
                LODS    Byte ptr DS:[SI]
                CMP     AL,0
                JE      _str03

                MOV     AH , AL
                MOV     DX , SCU_CMD
_str02:
                IN      AL , DX
                TEST    AL , 00000001B  ;送信データ・バッファ状態(TBRDY)の確認
                JZ      _str02          ;TBRDY=0 -> ﾙｰﾌﾟ

                MOV     DX , SCU_TXD
                MOV     AL , AH
                OUT     DX , AL         ;送信データを書き込み

                JMP     _str01
_str03: Endm
;-------------------------------------------------
        .LIST
```

保守／廃止

```
        NAME    PROGRAM0
        Page    60,132
        .186
;*******************************************************************************
        INCLUDE INIT_REG.ASI    ;
        INCLUDE T3.ASI          ;拡張モード用マクロ・プログラム
;*******************************************************************************
;
;       Program0
;

_TEXT           Segment
        Assume  CS:CGROUP , DS:CGROUP

                CLD                     ;DIRﾌﾗｸﾞをﾘｾｯﾄ(下位→上位)
                CLI                     ;外部割り込みを禁止

;*** ﾃﾞｰﾀ･ｾｸﾞﾒﾝﾄ･ﾚｼﾞｽﾀの設定 ***
                MOV     AX , CS
                MOV     DS , AX

                MOV     SI , OFFSET MSG_0
                MON_STR SI

                IRET

;*** ﾓﾆﾀ画面表示ﾃﾞｰﾀの設定 ***
MSG_0   DB      "*** Access to Function 0 ***" , 0DH , 0AH , 0

_TEXT           Ends
E:
        END
```

保守／廃止

```
        NAME    PROGRAM1
        Page    60,132
        .186
;*******************************************************************************
        INCLUDE INIT_REG.ASI    ;
        INCLUDE T3.ASI          ;拡張モード用マクロ・プログラム
;*******************************************************************************
;
;       Program1
;

_TEXT           Segment
        Assume  CS:CGROUP , DS:CGROUP

                CLD                     ;DIRﾌﾗｸﾞをﾘｾｯﾄ(下位→上位)
                CLI                     ;外部割り込みを禁止

;*** ﾃﾞｰﾀ･ｾｸﾞﾒﾝﾄ･ﾚｼﾞｽﾀの設定 ***
                MOV     AX , CS
                MOV     DS , AX

                MOV     SI , OFFSET MSG_1
                MON_STR SI

                IRET

;*** ﾓﾆﾀ画面表示ﾃﾞｰﾀの設定 ***
MSG_1   DB      "*** Access to Function 1 ***" , 0DH , 0AH , 0

_TEXT           Ends
E:
        END
```

保守／廃止

```
        NAME    PROGRAM2
        Page    60,132
        .186
;*******************************************************************************
        INCLUDE INIT_REG.ASI    ;
        INCLUDE T3.ASI          ;拡張モード用マクロ・プログラム
;*******************************************************************************
;
;       Program2
;

_TEXT           Segment
        Assume  CS:CGROUP , DS:CGROUP

                CLD                     ;DIRﾌﾗｸﾞをﾘｾｯﾄ(下位→上位)
                CLI                     ;外部割り込みを禁止

;*** ﾃﾞｰﾀ･ｾｸﾞﾒﾝﾄ･ﾚｼﾞｽﾀの設定 ***
                MOV     AX , CS
                MOV     DS , AX

                MOV     SI , OFFSET MSG_2
                MON_STR SI

                IRET

;*** ﾓﾆﾀ画面表示ﾃﾞｰﾀの設定 ***
MSG_2   DB      "*** Access to Function 2 ***" , 0DH , 0AH , 0

_TEXT           Ends
E:
        END
```

保守／廃止

```
        NAME    PROGRAM3
        Page    60,132
        .186
;*****************************************************************************
        INCLUDE INIT_REG.ASI    ;
        INCLUDE T3.ASI          ;拡張モード用マクロ・プログラム
;*****************************************************************************
;
;       Program3
;

_TEXT           Segment
        Assume  CS:CGROUP , DS:CGROUP

                CLD                     ;DIRﾌﾗｸﾞをﾘｾｯﾄ(下位→上位)
                CLI                     ;外部割り込みを禁止

;*** ﾃﾞｰﾀ･ｾｸﾞﾒﾝﾄ･ﾚｼﾞｽﾀの設定 ***
                MOV     AX , CS
                MOV     DS , AX

                MOV     SI , OFFSET MSG_3
                MON_STR SI

                IRET

;*** ﾓﾆﾀ画面表示ﾃﾞｰﾀの設定 ***
MSG_3   DB      "*** Access to Function 3 ***" , 0DH , 0AH , 0

_TEXT           Ends
E:
        END
```

保守／廃止

```
        NAME    PROGRAM4
        Page    60,132
        .186
;*******************************************************************************
        INCLUDE INIT_REG.ASI    ;
        INCLUDE T3.ASI          ;拡張モード用マクロ・プログラム
;*******************************************************************************
;
;       Program4
;

_TEXT           Segment
        Assume  CS:CGROUP , DS:CGROUP

                CLD                         ;DIRﾌﾗｸﾞをﾘｾｯﾄ(下位→上位)
                CLI                         ;外部割り込みを禁止

;*** ﾃﾞｰﾀ･ｾｸﾞﾒﾝﾄ･ﾚｼﾞｽﾀの設定 ***
                MOV     AX , CS
                MOV     DS , AX

                MOV     SI , OFFSET MSG_4
                MON_STR SI

                IRET

;*** ﾓﾆﾀ画面表示ﾃﾞｰﾀの設定 ***
MSG_4   DB      "*** Access to Function ***" , 0DH , 0AH , 0

_TEXT           Ends
E:
        END
```

保守／廃止

```
        .XLIST
;+------------------------------------------------------------------+
;|      拡張アドレス空間用セグメント定義                            |
;+------------------------------------------------------------------+
;------------------------------------------------------------------
_TEST           Segment at 8000h
_TEST           Ends
;------------------------------------------------------------------
        .LIST
```

保守／廃止

```
        .XLIST
;+-------------------------------------------------------------+
;|      モニタ用共通        Version 1.0  <1993-04-20>          |
;+-------------------------------------------------------------+
;システムＩ／Ｏレジスタ・アドレス
SCTL    EQU     0FFFEH  ;ｼｽﾃﾑ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
OPSEL   EQU     0FFFDH  ;内蔵ﾍﾟﾘﾌｪﾗﾙ選択ﾚｼﾞｽﾀ
OPHA    EQU     0FFFCH  ;内蔵ﾍﾟﾘﾌｪﾗﾙ･ﾘﾛｹｰｼｮﾝ･ﾚｼﾞｽﾀ  上位8ﾋﾞｯﾄ
DULA    EQU     0FFFBH  ;           〃                DMA  下位8ﾋﾞｯﾄ
IULA    EQU     0FFFAH  ;           〃                ICU  下位8ﾋﾞｯﾄ
TULA    EQU     0FFF9H  ;           〃                TCU  下位8ﾋﾞｯﾄ
SULA    EQU     0FFF8H  ;           〃                SCU  下位8ﾋﾞｯﾄ
WCY4    EQU     0FFF6H  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ4
WCY3    EQU     0FFF5H  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ3
WCY2    EQU     0FFF4H  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ2
WMB1    EQU     0FFF3H  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ﾒﾓﾘ領域設定ﾚｼﾞｽﾀ1
RFC     EQU     0FFF2H  ;ﾘﾌﾚｯｼｭ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
SBCR    EQU     0FFF1H  ;ｽﾀﾝﾊﾞｲ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
TCKS    EQU     0FFF0H  ;ﾀｲﾏ･ｸﾛｯｸ選択ﾚｼﾞｽﾀ
WAC     EQU     0FFEDH  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ﾒﾓﾘ･ｱﾄﾞﾚｽ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
WCY0    EQU     0FFECH  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ0
WCY1    EQU     0FFEBH  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ1
WMB0    EQU     0FFEAH  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ﾒﾓﾘ領域設定ﾚｼﾞｽﾀ0
BRC     EQU     0FFE9H  ;ﾎﾞｰ･ﾚｰﾄ･ｶｳﾝﾀ
BADR    EQU     0FFE1H  ;ﾊﾞﾝｸ･ｱﾄﾞﾚｽ･ﾚｼﾞｽﾀ
BSEL    EQU     0FFE0H  ;ﾊﾞﾝｸ選択ﾚｼﾞｽﾀ
PGR1    EQU     0FF00H  ;PGR1のI/Oｱﾄﾞﾚｽ
PGR33   EQU     0FF40H  ;PGR33のI/Oｱﾄﾞﾚｽ
PGR36   EQU     0FF46H  ;PGR36のI/Oｱﾄﾞﾚｽ
PGR64   EQU     0FF7EH  ;PGR64のI/Oｱﾄﾞﾚｽ
;-----------------------------------------------
;ＣＰＵ（Ｖ５３Ａ）のイニシャライズ　マクロ定義
;-----------------------------------------------
CPU_INIT Macro                                  ; CPUの初期設定
                MOV     DX , SCTL               ; ｼｽﾃﾑ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
                MOV     AL , 10H                ; SCUへの入力ｸﾛｯｸ ← ﾎﾞｰ･ﾚｰﾄ･ｼﾞｪﾈﾚｰﾀ
                OUT     DX , AL                 ; IOAG = 0
                MOV     DX , SBCR               ; ｽﾀﾝﾊﾞｲ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
                MOV     AL , 00H
                OUT     DX , AL

                MOV     DX , RFC                ;PSRAMのﾘﾌﾚｯｼｭ
                MOV     AL , 0D2H               ;DARAMとPSRAMの両方を使用するが,
                OUT     DX , AL                 ;PSRAMのほうがﾘﾌﾚｯｼｭ周期が短いの
                                                ;で,PSRAMのﾘﾌﾚｯｼｭ周期をもとに設定
                MOV     DX , OPHA               ;ｵﾝﾁｯﾌﾟ･ﾍﾟﾘﾌｪﾗﾙ･ﾊｲ･ｱﾄﾞﾚｽ･ﾚｼﾞｽﾀ
                MOV     AL , 0FEH               ;内蔵I/Oｱﾄﾞﾚｽ 0FExxH
                OUT     DX , AL
                MOV     DX , OPSEL              ;ｵﾝﾁｯﾌﾟ･ﾍﾟﾘﾌｪﾗﾙ･ｾﾚｸｼｮﾝ･ﾚｼﾞｽﾀ
                MOV     AL , 08H                ;Used SCU
                OUT     DX , AL
                MOV     DX , DULA               ;DMAUﾛｳ･ｱﾄﾞﾚｽ･ﾚｼﾞｽﾀ
                MOV     AL , 50H                ;I/Oｱﾄﾞﾚｽ 0FE50H-0FE5FH
                OUT     DX , AL
                MOV     DX , IULA               ;ICUﾛｳ･ｱﾄﾞﾚｽ･ﾚｼﾞｽﾀ
                MOV     AL , 40H                ;I/Oｱﾄﾞﾚｽ 0FE40H-0FE42H
```

保守／廃止

```
                OUT     DX , AL
                MOV     DX , TULA           ;TCUﾛｳ･ｱﾄﾞﾚｽ･ﾚｼﾞｽﾀ
                MOV     AL , 30H            ;I/Oｱﾄﾞﾚｽ 0FE30H-0FE36H
                OUT     DX , AL
                MOV     DX , SULA           ;SCUﾛｳ･ｱﾄﾞﾚｽ･ﾚｼﾞｽﾀ
                MOV     AL , 20H            ;I/Oｱﾄﾞﾚｽ 0FE20H-0FE26H
                OUT     DX , AL
                MOV     DX , TCKS           ;ﾀｲﾏ･ｸﾛｯｸ選択ﾚｼﾞｽﾀ
                MOV     AL , 00H
                OUT     DX , AL
                MOV     DX , WMB0           ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ﾒﾓﾘ領域設定ﾚｼﾞｽﾀ0
                MOV     AL , 55H
                OUT     DX , AL
                MOV     DX , WCY0           ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ0
                MOV     AL , 00H
                OUT     DX , AL
                MOV     DX , WCY1           ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ1
                MOV     AL , 13H
                OUT     DX , AL
                MOV     DX , WAC            ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ﾒﾓﾘ･ｱﾄﾞﾚｽ･ｺﾝﾄﾛｰﾙ･
                                            ;ﾚｼﾞｽﾀ
                MOV     AL , 00H
                OUT     DX , AL
                MOV     DX , WMB1           ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ﾒﾓﾘ領域設定ﾚｼﾞｽﾀ1
                MOV     AL , 75H
                OUT     DX , AL
                MOV     DX , WCY2           ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ2
                MOV     AL , 20H
                OUT     DX , AL
                MOV     DX , WCY3           ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ3
                MOV     AL , 72H            ;I/O 7wait
                OUT     DX , AL
                MOV     DX , WCY4           ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ4
                MOV     AL , 21H            ;DMA 2wait , Refresh 1wait
                OUT     DX , AL
                MOV     DX , BRC            ;ﾎﾞｰ･ﾚｰﾄ･ｶｳﾝﾀ
                MOV     AL , 130            ;20MHz  9600bps
;               MOV     AL , 43             ;13.3MHz 9600bps
                OUT     DX , AL
        Endm
;-----------------------------------------------
;    ページ・レジスタの初期化
;-----------------------------------------------
;
PGR_INIT        Macro
        Local   next_pgr
        MOV     DX , PGR1
        MOV     AX , 0000H
next_pgr:
        OUT     DX , AX
        ADD     DX , 2
        INC     AX
        CMP     DX , PGR64 + 2
        JB      short next_pgr
        Endm
```

保守／廃止

```
;-----------------------------------------------
;    拡張ｱﾄﾞﾚｽ･ﾓｰﾄﾞ切り替え用ﾍﾟｰｼﾞ･ﾚｼﾞｽﾀの設定
;-----------------------------------------------
;
PGR_SET         Macro   cnt
        Local   next_pgr_set , set_n , set_strad
        MOV     AX , cnt
        CMP     AX , 0
        JNE     set_n

        MOV     AX , 0020H
        JMP     set_strad

set_n:  DEC     AL
        MOV     DL , 4
        MUL     DL
        ADD     AX , 0080H

set_strad:
        MOV     DX , PGR33
next_pgr_set:
        OUT     DX , AX
        ADD     DX , 2
        INC     AX
        CMP     DX , PGR36 + 2
        JB      short next_pgr_set
        Endm
;-----------------------------------------------
;       拡張ｱﾄﾞﾚｽ･ﾓｰﾄﾞの設定
;-----------------------------------------------
BRKXA   Macro   vect
        DB      0FH , 0E0H , vect
        Endm
;-----------------------------------------------
;       拡張ｱﾄﾞﾚｽ･ﾓｰﾄﾞの解除
;-----------------------------------------------
RETXA   Macro   vect
        DB      0FH , 0F0H , vect ;RETXAｺﾏﾝﾄﾞ･ｺｰﾄﾞ
        Endm
;-----------------------------------------------
        .LIST
```

保守／廃止

```
        NAME    REGULAR1
        Page    60,132
        .186
;*************************************************************************
;*                                                                       *
;*      Extend address TEST PROGRAM - 1                                  *
;*                                      Version 1.00  <1994-02-01>       *
;*************************************************************************
        INCLUDE INIT_REG.ASI    ;通常ﾓｰﾄﾞ用ｾｸﾞﾒﾝﾄ定義
        INCLUDE INIT_EXT.ASI    ;拡張ﾓｰﾄﾞ用ｾｸﾞﾒﾝﾄ定義
        INCLUDE T6_REG.ASI      ;通常ﾓｰﾄﾞ用ﾏｸﾛ･ﾌﾟﾛｸﾞﾗﾑ
;*************************************************************************
;
;       MAIN PROGRAM
;

_TEXT           Segment
        Assume  CS:CGROUP , DS:CGROUP
RESET_START:
                CLD                     ;DIRﾌﾗｸﾞをﾘｾｯﾄ(下位→上位)
                CLI                     ;外部割り込みを禁止
;
;***  ﾍﾞｸﾀの設定  ***
;
                MOV     AX , 0
                MOV     DS , AX

                MOV     SI , 254 * 4    ;ﾍﾞｸﾀ254にXA1のｱﾄﾞﾚｽを設定(BRKXA命令で使用)
                MOV     Word Ptr DS:[SI]   , Offset XA1
                MOV     Word Ptr DS:[SI+2] , Seg  XA1

                MOV     SI , 253 * 4    ;ﾍﾞｸﾀ253にXA2のｱﾄﾞﾚｽを設定(RETXA命令で使用)
                MOV     Word Ptr DS:[SI]   , Offset XA2
                MOV     Word Ptr DS:[SI+2] , Seg  XA2

                MOV     SI , 252 * 4    ;ﾍﾞｸﾀ252に拡張ｱﾄﾞﾚｽ空間切り替え用領域の先頭
                                        ;ｱﾄﾞﾚｽを設定
                MOV     Word Ptr DS:[SI]   , 0
                MOV     Word Ptr DS:[SI+2] , _TEST
;
;*** ﾃﾞｰﾀ･ｾｸﾞﾒﾝﾄ･ﾚｼﾞｽﾀの設定 ***
;
                MOV     AX , CS
                MOV     DS , AX
;
;***  イニシャライズ  ***
;
                CPU_INIT            ;CPUのｲﾆｼｬﾗｲｽﾞ
                PGR_INIT            ;ﾍﾟｰｼﾞ･ﾚｼﾞｽﾀのｲﾆｼｬﾗｲｽﾞ
;
;***  文字表示プログラム  ***
;

mor:            MOV     Word Ptr CS:CNT, 0      ;変数CNTをｸﾘｱ
next_program:
                MOV     AX , Word Ptr CS:CNT
```

保守／廃止

```
                PGR_SET AX              ;PROGRAM0～4に使用するﾍﾟｰｼﾞ･ﾚｼﾞｽﾀを設定

                BRKXA   254             ;拡張ｱﾄﾞﾚｽ･ﾓｰﾄﾞの設定
XA1:            INT     252             ;PROGRAM0～4の実行(PGRの設定内容により選択)
                RETXA   253             ;拡張ｱﾄﾞﾚｽ･ﾓｰﾄﾞの解除

XA2:            INC     Word Ptr CS:CNT         ;変数CNTをｲﾝｸﾘﾒﾝﾄ
                CMP     Word Ptr CS:CNT, 5      ;変数CNTが5ならばnext-programへ
                                                ;ｼﾞｬﾝﾌﾟ
                JE      mor             ;5以外ならば次のﾃﾞｰﾀ･ﾃｰﾌﾞﾙをｱｸｾｽ

                MOV     AX , Word Ptr CS:CNT
                JMP     next_program
;
;***  変数ＣＮＴ  ***
;
CNT     DW      0       ;ﾃﾞｰﾀ･ﾃｰﾌﾞﾙをｱｸｾｽするのに使用する変数CNTを確保
_TEXT           Ends

        END     RESET_START
```

保守／廃止

（2）C言語ソース

```
/*-------------------------------------------------------------------------*/
/*      プロトタイプ宣言                                                   */
/*-------------------------------------------------------------------------*/
void    cpu_init ( void );
void    pgr_init ( void );
void    pgr_set ( int );
char    sendc ( char );
/*-------------------------------------------------------------------------*/
/*      ＣＰＵ　初期設定処理                                              */
/*-------------------------------------------------------------------------*/
void    cpu_init( void ) |
#if 1
        outportb( 0xFFFE , 0x10 ) ; /* SCTL  = 00010000B */
        outportb( 0xFFF1 , 0x00 ) ; /* SBCR  = 00000000B */
        outportb( 0xFFF2 , 0x92 ) ; /* RFC   = 10010010B */
        outportb( 0xFFFC , 0xFE ) ; /* OPHA  = 0FEH      */
        outportb( 0xFFFD , 0x08 ) ; /* OPSEL = 00001111B */
        outportb( 0xFFFB , 0x50 ) ; /* DULA  = 50H       */
        outportb( 0xFFFA , 0x40 ) ; /* IULA  = 40H       */
        outportb( 0xFFF9 , 0x30 ) ; /* TULA  = 30H       */
        outportb( 0xFFF8 , 0x20 ) ; /* SULA  = 20H       */
        outportb( 0xFFF0 , 0x00 ) ; /* TCKS  = 00000000B */
        outportb( 0xFFEA , 0x55 ) ; /* WMB0  = 01010101B */
        outportb( 0xFFEC , 0x00 ) ; /* WCY0  = 00000000B */
        outportb( 0xFFEB , 0x13 ) ; /* WCY1  = 00010010B */
        outportb( 0xFFED , 0x00 ) ; /* WAC   = 00000000B */
        outportb( 0xFFF3 , 0x75 ) ; /* WMB1  = 01110101B */
        outportb( 0xFFF4 , 0x20 ) ; /* WCY2  = 00100000B */
        outportb( 0xFFF5 , 0x72 ) ; /* WCY3  = 00100010B */
        outportb( 0xFFF6 , 0x21 ) ; /* WCY4  = 00100001B */
#endif
        outportb( 0xFFE9 , 130 )  ; /* BRC  = 130:20MHz -> 9600bps */
|
/*-------------------------------------------------------------------------*/
/*      ページ・レジスタ制御                                               */
/*-------------------------------------------------------------------------*/
/* page register I/O address */
#define PGR1_ADD        0xff00
#define PGR33_ADD       0xff40
#define PGR36_ADD       0xff46
#define PGR64_ADD       0xff7e
int     io_add , data;
```

保守／廃止

```
/*-------------------------------------------------------*/
/*      ページ・レジスタ初期設定処理                     */
/*-------------------------------------------------------*/
void    pgr_init( void ) {
                data = 0x0000;
                for( io_add = PGR1_ADD ; (unsigned)io_add != ( PGR64_ADD + 2 )
; io_add += 2 ){
                        outport( io_add , data );
                        data ++;
                }
        }
/*-------------------------------------------------------*/
/*      ページ・レジスタ設定                             */
/*-------------------------------------------------------*/
void    pgr_set( int cnt ) {

                if( cnt == 0 ){
                        data = 0x0020;
                }
                else{
                        data = 0x0080 + ( cnt - 1 ) * 4;
                }

                for( io_add = PGR33_ADD ; (unsigned)io_add != ( PGR36_ADD + 2 )
; io_add += 2 ){
                        outport( io_add , data );
                        data ++;
                }
        }
```

保守／廃止

```
#pragma inline
#include <dos.h>
#include <string.h>
/*************************************/
void    main ( void )   ;
void    cpu_init ( void ) ;
void    pgr_init ( void ) ;
void    pgr_set ( int ) ;
void    brkxa ( void );
void    xa ( void );
void    retxa ( void );
void    vector_init ( void );
char far ( *fnc )( );
int     cnt;
/*************************************/
void main( void ) {
        vector_init ();
        cpu_init() ;
        pgr_init() ;

        fnc = ( char far (*)() )0x80000000;

        cnt = 0;
        while( cnt != 5 ){
                pgr_set ( cnt );
                brkxa();
                cnt++;
        }
}
/*-------------------------------------------------------------------------*/
/*      拡張アドレス・モード・スイッチング・プログラム                     */
/*-------------------------------------------------------------------------*/
void    brkxa ( void ){
        |
        }
        asm     db      0fh , 0e0h , 0feh;  /*brkxa*/
}

void    xa ( void ){
        {
                (*fnc) ( );
        }
        asm     db      0fh , 0f0h , 0fdh;  /*retxa*/
}

void    retxa ( void ){
}
/*-------------------------------------------------------------------------*/
/*      ベクタの設定                                                       */
/*-------------------------------------------------------------------------*/
void    vector_init ( void ){
        poke( 0 , 0x0fe*4 , FP_OFF( xa ) ) ;
        poke( 0 , 0x0fe*4 + 2 , FP_SEG( xa ) ) ;
        poke( 0 , 0x0fd*4 , FP_OFF( retxa ) ) ;
        poke( 0 , 0x0fd*4 + 2 , FP_SEG( retxa ) ) ;
}
```

保守／廃止

```
/*****************************************/
/*      Function 0 program               */
/*      (80000h-8ffffh)                  */
/*****************************************/
#include <string.h>

/***  プロトタイプ宣言  ***/

void    main ( void );
char    sendc ( char );

/***  register address  ***/

#define SCU_STS 0xFE22          /* STATUS     REGISTER (READ)  */
#define SCU_TXD 0xFE20          /* 送信データ REGISTER (WRITE) */

/***  表示制御プログラム  ***/

void    main ( void ){

        char    *msg0;
        msg0 = "*** Access To Function 0 ***¥r¥n";

        while( *msg0 != '¥0' ){
                sendc( *msg0 );
                msg0++;
        }

}
/*** 1文字表示プログラム ***/
char    sendc( char c ) {
        while( (inportb( SCU_STS ) & 0x01) == 0 ) ;
        outportb( SCU_TXD , c ) ;
        return( c ) ;
}
```

保守／廃止

```
/*****************************************/
/*      Function 1 program               */
/*      (200000h-20ffffh)                */
/*****************************************/
#include <string.h>

/***  プロトタイプ宣言  ***/

void    main ( void );
char    sendc ( char );

/***  register address  ***/

#define SCU_STS 0xFE22          /* STATUS     REGISTER (READ)  */
#define SCU_TXD 0xFE20          /* 送信データ REGISTER (WRITE) */

/***  表示制御プログラム  ***/

void    main ( void ){

        char    *msg1;
        msg1 = "*** Access To Function 1 ***¥r¥n";

        while( *msg1 != '¥0' ){
                sendc( *msg1 );
                msg1++;
        }

}
/*** 1文字表示プログラム ***/
char    sendc( char c ) {
        while( (inportb( SCU_STS ) & 0x01) == 0 ) ;
        outportb( SCU TXD , c ) ;
        return( c ) ;
}
```

保守／廃止

```
/*****************************************/
/*      Function 2 program               */
/*      (210000h-21ffffh)                */
/*****************************************/
#include <string.h>

/***  プロトタイプ宣言  ***/

void    main ( void );
char    sendc ( char );

/***  register address  ***/

#define SCU_STS 0xFE22          /* STATUS      REGISTER (READ)  */
#define SCU_TXD 0xFE20          /* 送信データ  REGISTER (WRITE) */

/***  表示制御プログラム  ***/

void    main ( void )|

        char    *msg2;
        msg2 = "*** Access To Function 2 ***¥r¥n";

        while( *msg2 != '¥0' )|
                sendc( *msg2 );
                msg2++;
        }

}
/*** 1文字表示プログラム ***/
char    sendc( char c ) {
        while( (inportb( SCU_STS ) & 0x01) == 0 ) ;
        outportb( SCU_TXD , c ) ;
        return( c ) ;
}
```

保守／廃止

```
/*****************************************/
/*      Function 3 program               */
/*      (220000h-22ffffh)                */
/*****************************************/
#include <string.h>

/***  プロトタイプ宣言  ***/

void    main ( void );
char    sendc ( char );

/***  register address  ***/

#define SCU_STS 0xFE22          /* STATUS     REGISTER (READ)  */
#define SCU_TXD 0xFE20          /* 送信データ REGISTER (WRITE) */

/***  表示制御プログラム  ***/

void    main ( void )|

        char    *msg3;
        msg3 = "*** Access To Function 3 ***¥r¥n";

        while( *msg3 != '¥0' ){
                sendc( *msg3 );
                msg3++;
        }

}
/*** 1文字表示プログラム ***/
char    sendc( char c ) {
        while( (inportb( SCU_STS ) & 0x01) == 0 ) ;
        outportb( SCU_TXD , c ) ;
        return( c ) ;
}
```

保守／廃止

```
/*****************************************/
/*      Function 4 program               */
/*      (230000h-23ffffh)                */
/*****************************************/
#include <string.h>

/***  プロトタイプ宣言  ***/

void    main ( void );
char    sendc ( char );

/***  register address  ***/

#define SCU_STS 0xFE22          /* STATUS     REGISTER (READ)  */
#define SCU_TXD 0xFE20          /* 送信データ REGISTER (WRITE) */

/***  表示制御プログラム  ***/

void    main ( void ){

        char    *msg4;
        msg4 = "*** Access To Function 4 ***¥r¥n";

        while( *msg4 != '¥0' ){
                sendc( *msg4 );
                msg4++;
        }

}
/*** 1文字表示プログラム ***/
char    sendc( char c ) {
        while( (inportb( SCU_STS ) & 0x01) == 0 ) ;
        outportb( SCU_TXD , c ) ;
        return( c ) ;
}
```

〔メ　モ〕

保守／廃止

第３章　拡張アドレス空間のデータ領域

拡張アドレス空間にデータをマッピングする場合について説明します。

3.1　機　　能

拡張アドレス空間をデータ領域として利用することにより大容量のデータへのアクセスができます。

3.2　用 途 例

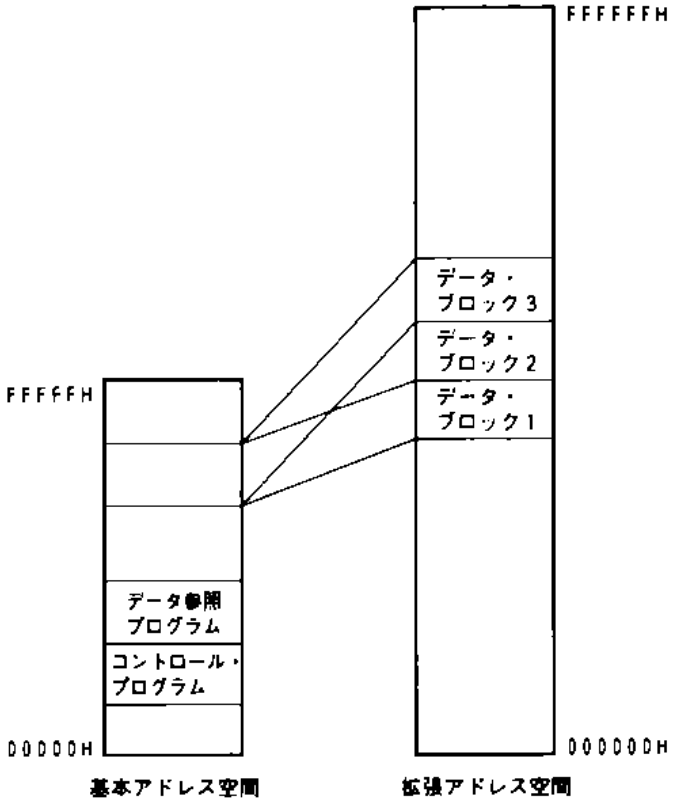
① ワード・プロセッサ、カー・ナビゲーション、ハンディ・ターミナルなどの機器で使用するアプリケーション・ソフトのデータをマッピングします。

② ワード・プロセッサ、カー・ナビゲーション、ハンディ・ターミナルなどディスプレイを搭載する機器のVRAM領域として利用します。

保守／廃止

3.3　マッピング例

図3－1　データ領域としての利用例



備考1．コントロール・プログラム：

ページ・レジスタの設定を行う通常モードで動作するプログラム

2．データ参照プログラム：

拡張アドレス・モードで動作し，拡張アドレス空間のデータをアクセスするプログラム

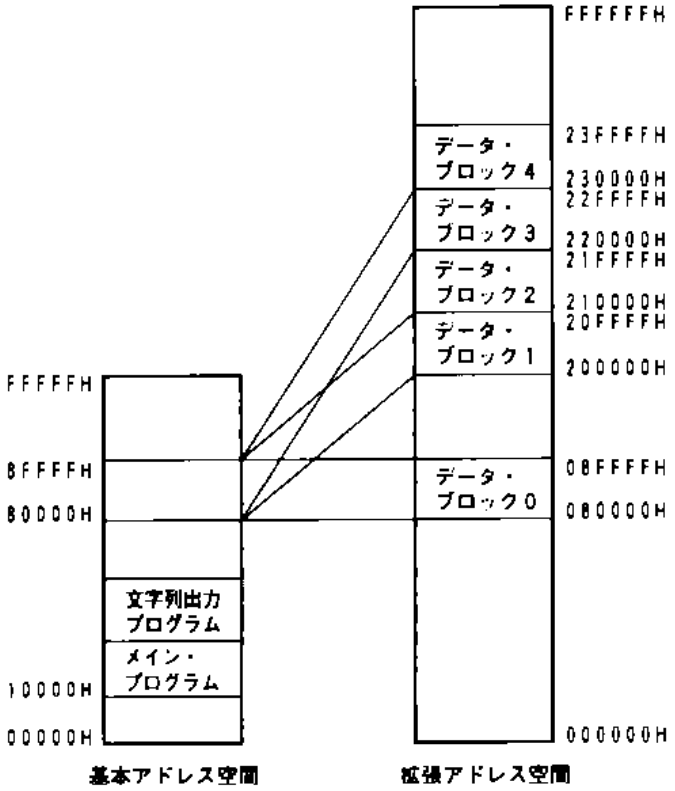
保守／廃止

3.4　プログラム例

拡張アドレス空間をコード領域として利用した場合のV53Aボード上で動作するプログラム例について説明します。

3.4.1　メモリ構成

図3－2　データ・マッピング例



保守／廃止

３．４．２　プログラム処理

各プログラムでは次のような動作を行います。

（１）メイン・プログラム

・通常モードで動作します。

・拡張アドレス・モード用命令BRKXA, RETXAで使用するベクタ・テーブルの設定、CPUの初期化処理としてV53Aのシステム I/Oレジスタ、ページ・レジスタの初期設定を行います。

そのあと、変数CNTによりデータ・ブロック０-４のリロケーションを制御し、拡張アドレス・モードにて文字列出力プログラム（msg_out）を呼び出します。

（２）文字列出力プログラム（msg_out）

・拡張アドレス・モードで動作します。

・変数CNTの値により表３－１に示す文字列をセレクトし、80000Hで表現されるアドレス空間にロードします。

そのあと、ロードした文字列を内蔵SCUに出力します。

表３－１　文 字 列

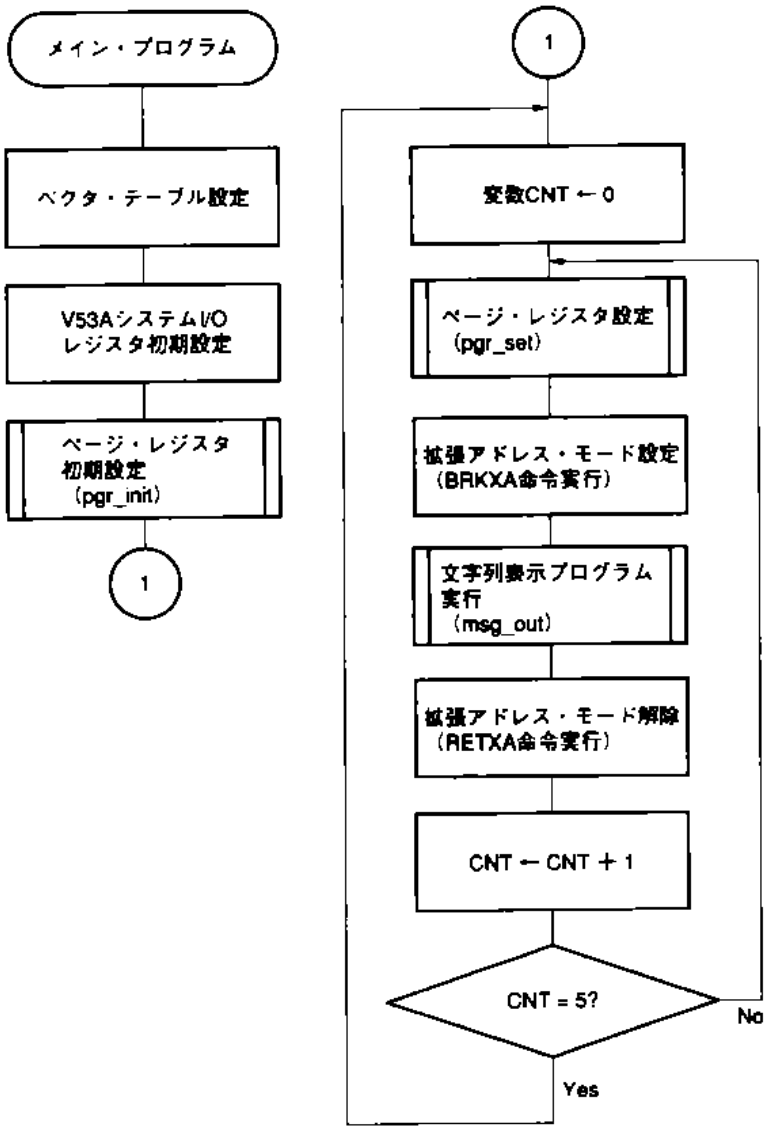
CNTの値	文 字 列
0	* * * Access To Data Block 0 * * *
1	* * * Access To Data Block 1 * * *
2	* * * Access To Data Block 2 * * *
3	* * * Access To Data Block 3 * * *
4	* * * Access To Data Block 4 * * *

保守／廃止

3.4.3　フロー・チャート

（1）メイン・ルーチン

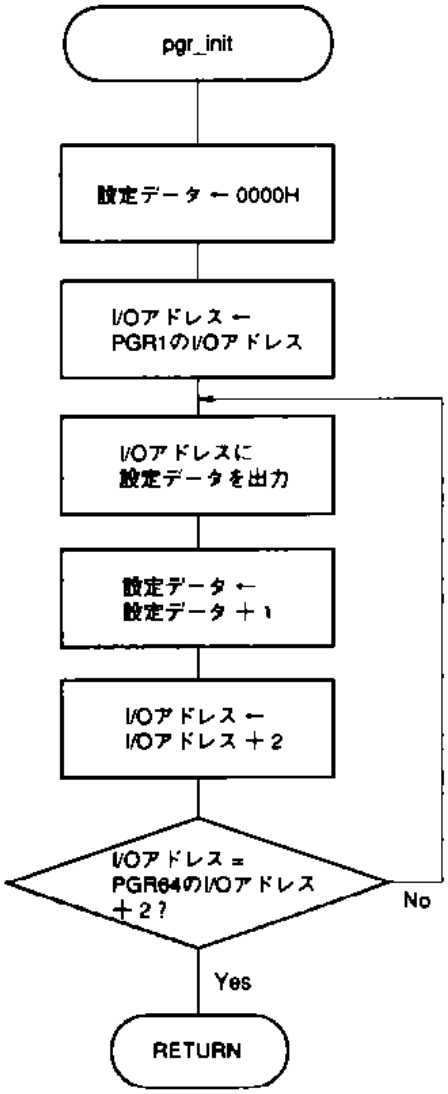
V53Aのイニシャライズおよび文字列表示プログラムmsg_outで参照するデータ・ブロックのセレクトを行います。



保守／廃止

（２）pgr_initルーチン

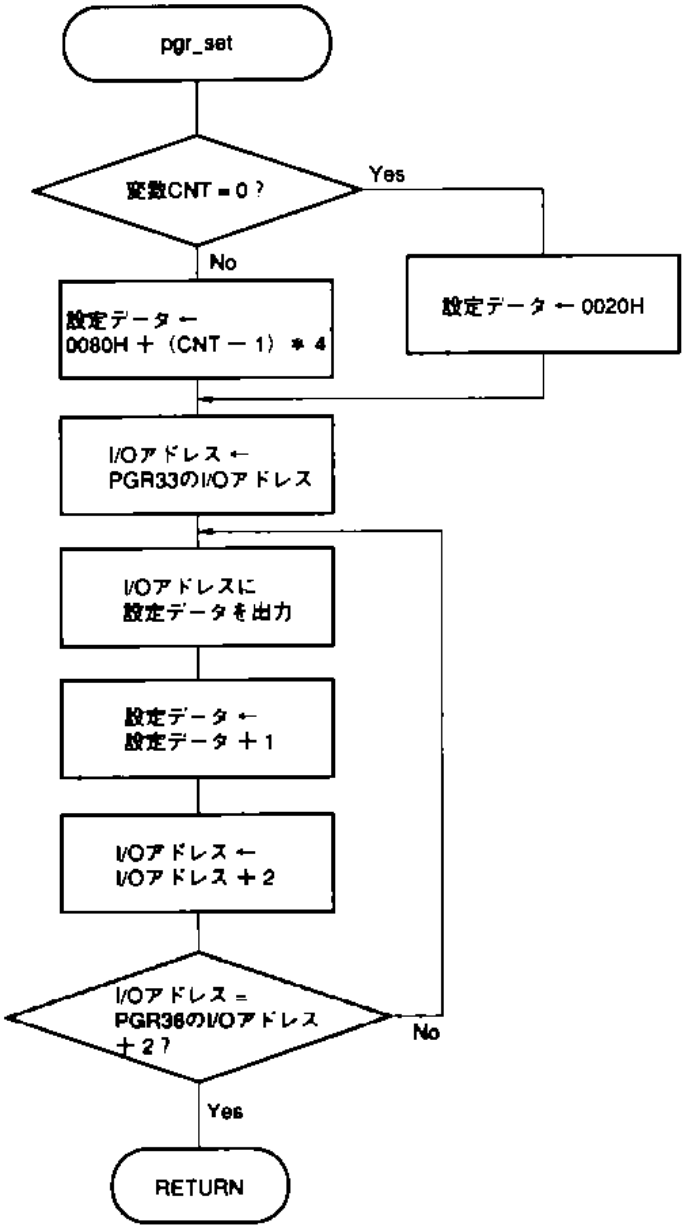
V53Aのページ・レジスタの初期設定を行います。



保守／廃止

（３）pgr_setルーチン

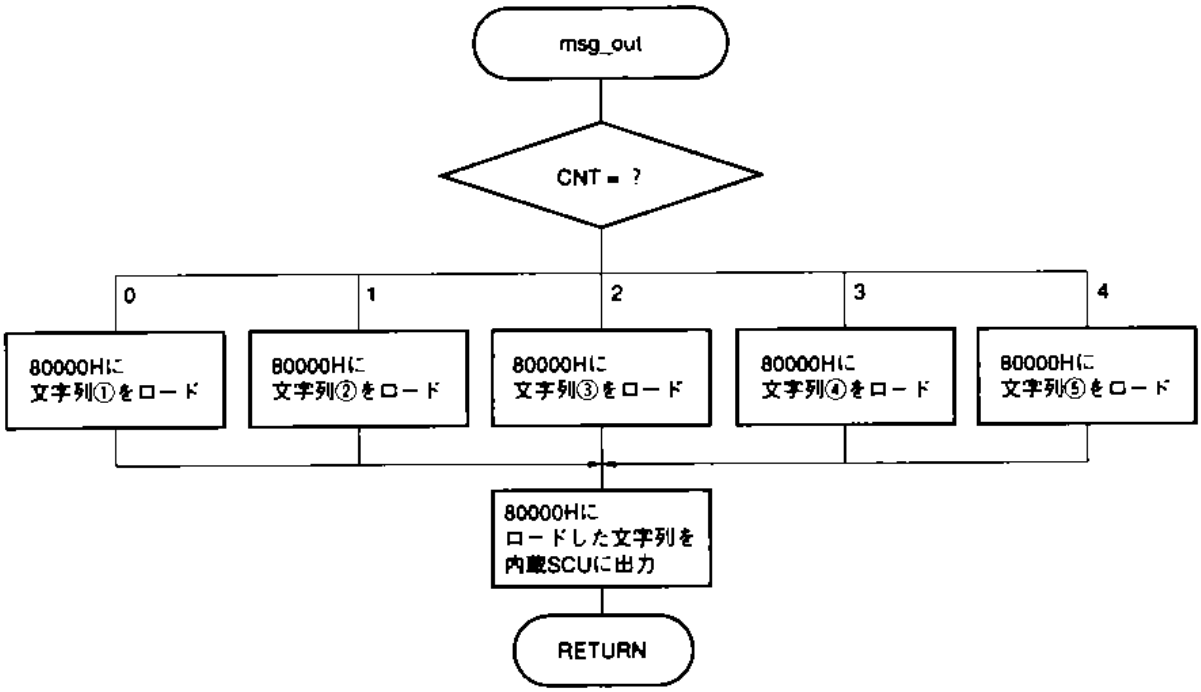
変数CNTの値２より基本アドレス空間80000H-8FFFFHをリロケートするのに必要なPGR33-PGR36の設定を行う。



保守／廃止

（４）msg_outルーチン

変数CNTの値により文字列①-⑤のメッセージをセレクトし、80000H-8FFFFHにロードします。そのあと，ロードしたメッセージを内蔵SCUに出力します。



	文字列
①	*** Access To Data Block 0 ***
②	*** Access To Data Block 1 ***
③	*** Access To Data Block 2 ***
④	*** Access To Data Block 3 ***
⑤	*** Access To Data Block 4 ***

保守／廃止

3.4.4　ソース・プログラム

（1）アセンブラ・ソース

```
        .XLIST
;+--------------------------------------------------------------+
;|      拡張アドレス空間用セグメント定義                        |
;+--------------------------------------------------------------+
;---------------------------------------------------------------
_TEST           Segment at 8000h
_TEST           Ends
;---------------------------------------------------------------
        .LIST
```

保守／廃止

```
        DOSSEG
;+------------------------------------------------------------------+
;|      セグメント定義          Version 1.0  <1990-09-09>          |
;+------------------------------------------------------------------+
_TEXT           Segment byte public 'CODE'
_TEXT           Ends

_TEXTEND        Segment para public 'CODE'
Public          ECODE
ECODE           label   byte                ; コード･セグメントの最後
_TEXTEND        Ends

_DATA           Segment byte public 'DATA'
Public          SDATA
SDATA           label   byte                ; データ･セグメントの最初
_DATA           Ends

_DATAEND        Segment byte public 'STACK'
EDATA           label   byte                ; データ･セグメントの最後
_DATAEND        Ends

_STACK          Segment stack 'STACK'       ; スタック･セグメント
_STACK          Ends

CGROUP          GROUP   _TEXT,_TEXTEND
DGROUP          GROUP   _DATA,_DATAEND,_STACK
        Page
```

保守／廃止

```
        .XLIST
;-------------------------------------------------------------------------------
SCU_CMD         EQU     0FE22H  ;内蔵ＳＣＵのコマンド・アドレス
SCU_MOD         EQU     0FE24H  ;内蔵ＳＣＵのモード・アドレス
SCU_STT         EQU     0FE22H  ;内蔵ＳＣＵのステータス・レジスタ・アドレス
SCU_RXD         EQU     0FE20H  ;内蔵ＳＣＵのデータ・アドレス
SCU_TXD         EQU     0FE20H  ;内蔵ＳＣＵのデータ・アドレス
SCU_IMK         EQU     0FE26H  ;内蔵ＳＣＵのマスク・レジスタ
;-------------------------------------------------------------------------------
;１文字送信　マクロ
MON_STR Macro   addr
        Local   _str01 , _str02 , _str03
                MOV     SI,addr
_str01:
                LODS    Byte ptr DS:[SI]
                CMP     AL,0
                JE      _str03

                MOV     AH , AL
                MOV     DX , SCU_CMD
_str02:
                IN      AL , DX
                TEST    AL , 00000001B  ;送信データ・バッファ状態(TBRDY)の確認
                JZ      _str02          ;TBRDY=0 -> ﾙｰﾌﾟ

                MOV     DX , SCU_TXD
                MOV     AL , AH
                OUT     DX , AL         ;送信データを書き込み

                JMP     _str01
_str03: Endm
;-----------------------------------------------
        .LIST
```

保守／廃止

```
        .XLIST
;+-------------------------------------------------------------+
;|      モニタ用共通          Version 1.0  <1993-04-20>          |
;+-------------------------------------------------------------+
;システムＩ／Ｏレジスタ・アドレス
SCTL    EQU     0FFFEH  ;ｼｽﾃﾑ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
OPSEL   EQU     0FFFDH  ;内蔵ﾍﾟﾘﾌｪﾗﾙ選択ﾚｼﾞｽﾀ
OPHA    EQU     0FFFCH  ;内蔵ﾍﾟﾘﾌｪﾗﾙ･ﾘﾛｹｰｼｮﾝ･ﾚｼﾞｽﾀ  上位8ﾋﾞｯﾄ
DULA    EQU     0FFFBH  ;          〃                DMA  下位8ﾋﾞｯﾄ
IULA    EQU     0FFFAH  ;          〃                ICU  下位8ﾋﾞｯﾄ
TULA    EQU     0FFF9H  ;          〃                TCU  下位8ﾋﾞｯﾄ
SULA    EQU     0FFF8H  ;          〃                SCU  下位8ﾋﾞｯﾄ
WCY4    EQU     0FFF6H  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ4
WCY3    EQU     0FFF5H  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ3
WCY2    EQU     0FFF4H  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ2
WMB1    EQU     0FFF3H  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ﾒﾓﾘ領域設定ﾚｼﾞｽﾀ1
RFC     EQU     0FFF2H  ;ﾘﾌﾚｯｼｭ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
SBCR    EQU     0FFF1H  ;ｽﾀﾝﾊﾞｲ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
TCKS    EQU     0FFF0H  ;ﾀｲﾏ･ｸﾛｯｸ選択ﾚｼﾞｽﾀ
WAC     EQU     0FFEDH  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ﾒﾓﾘ･ｱﾄﾞﾚｽ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
WCY0    EQU     0FFECH  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ0
WCY1    EQU     0FFEBH  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ1
WMB0    EQU     0FFEAH  ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ﾒﾓﾘ領域設定ﾚｼﾞｽﾀ0
BRC     EQU     0FFE9H  ;ﾎﾞｰ･ﾚｰﾄ･ｶｳﾝﾀ
BADR    EQU     0FFE1H  ;ﾊﾞﾝｸ･ｱﾄﾞﾚｽ･ﾚｼﾞｽﾀ
BSEL    EQU     0FFE0H  ;ﾊﾞﾝｸ選択ﾚｼﾞｽﾀ
PGR1    EQU     0FF00H  ;PGR1のI/Oｱﾄﾞﾚｽ
PGR33   EQU     0FF40H  ;PGR33のI/Oｱﾄﾞﾚｽ
PGR36   EQU     0FF46H  ;PGR36のI/Oｱﾄﾞﾚｽ
PGR64   EQU     0FF7EH  ;PGR64のI/Oｱﾄﾞﾚｽ
;-----------------------------------------------
;ＣＰＵ（Ｖ５３Ａ）のイニシャライズ　マクロ定義
;-----------------------------------------------
CPU_INIT Macro                          ; CPUの初期設定
                MOV     DX , SCTL       ; ｼｽﾃﾑ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
                MOV     AL , 10H        ; SCUへの入力ｸﾛｯｸ ← ﾎﾞｰ･ﾚｰﾄ･ｼﾞｪﾈﾚｰﾀ
                OUT     DX , AL         ; IOAG = 0
                MOV     DX , SBCR       ; ｽﾀﾝﾊﾞｲ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
                MOV     AL , 00H
                OUT     DX , AL

                MOV     DX , RFC        ;PSRAMのﾘﾌﾚｯｼｭ
                MOV     AL , 0D2H       ;DARAMとPSRAMの両方を使用するが、
                OUT     DX , AL         ;PSRAMのほうがﾘﾌﾚｯｼｭ周期が短いの
                                        ;で、PSRAMのﾘﾌﾚｯｼｭ周期をもとに設定
                MOV     DX , OPHA       ;ｵﾝﾁｯﾌﾟ･ﾍﾟﾘﾌｪﾗﾙ･ﾊｲ･ｱﾄﾞﾚｽ･ﾚｼﾞｽﾀ
                MOV     AL , 0FEH       ;内蔵I/Oｱﾄﾞﾚｽ 0FExxH
                OUT     DX , AL
                MOV     DX , OPSEL      ;ｵﾝﾁｯﾌﾟ･ﾍﾟﾘﾌｪﾗﾙ･ｾﾚｸｼｮﾝ･ﾚｼﾞｽﾀ
                MOV     AL , 08H        ;Used SCU
                OUT     DX , AL
                MOV     DX , DULA       ;DMAUﾛｳ･ｱﾄﾞﾚｽ･ﾚｼﾞｽﾀ
                MOV     AL , 50H        ;I/Oｱﾄﾞﾚｽ 0FE50H-0FE5FH
                OUT     DX , AL
                MOV     DX , IULA       ;ICUﾛｳ･ｱﾄﾞﾚｽ･ﾚｼﾞｽﾀ
                MOV     AL , 40H        ;I/Oｱﾄﾞﾚｽ 0FE40H-0FE42H
```

保守／廃止

```
                OUT     DX , AL
                MOV     DX , TULA       ;TCUﾛｳ･ｱﾄﾞﾚｽ･ﾚｼﾞｽﾀ
                MOV     AL , 30H        ;I/Oｱﾄﾞﾚｽ 0FE30H-0FE36H
                OUT     DX , AL
                MOV     DX , SULA       ;SCUﾛｳ･ｱﾄﾞﾚｽ･ﾚｼﾞｽﾀ
                MOV     AL , 20H        ;I/Oｱﾄﾞﾚｽ 0FE20H-0FE26H
                OUT     DX , AL
                MOV     DX , TCKS       ;ﾀｲﾏ･ｸﾛｯｸ選択ﾚｼﾞｽﾀ
                MOV     AL , 00H
                OUT     DX , AL
                MOV     DX , WMB0       ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ﾒﾓﾘ領域設定ﾚｼﾞｽﾀ0
                MOV     AL , 55H
                OUT     DX , AL
                MOV     DX , WCY0       ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ0
                MOV     AL , 00H
                OUT     DX , AL
                MOV     DX , WCY1       ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ1
                MOV     AL , 13H
                OUT     DX , AL
                MOV     DX , WAC        ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ﾒﾓﾘ･ｱﾄﾞﾚｽ･ｺﾝﾄﾛｰﾙ･ﾚｼﾞｽﾀ
                MOV     AL , 00H
                OUT     DX , AL
                MOV     DX , WMB1       ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ﾒﾓﾘ領域設定ﾚｼﾞｽﾀ1
                MOV     AL , 75H
                OUT     DX , AL
                MOV     DX , WCY2       ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ2
                MOV     AL , 20H
                OUT     DX , AL
                MOV     DX , WCY3       ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ3
                MOV     AL , 72H        ;I/O 7wait
                OUT     DX , AL
                MOV     DX , WCY4       ;ﾌﾟﾛｸﾞﾗﾏﾌﾞﾙ･ｳｴｲﾄ･ｻｲｸﾙ数設定ﾚｼﾞｽﾀ4
                MOV     AL , 21H        ;DMA 2wait , Refresh 1wait
                OUT     DX , AL
                MOV     DX , BRC        ;ﾎﾞｰ･ﾚｰﾄ･ｶｳﾝﾀ
                MOV     AL , 130        ;20MHz  9600bps
;               MOV     AL , 43         ;13.3MHz 9600bps
                OUT     DX , AL
        Endm
;-----------------------------------------------
;   ページ・レジスタの初期化
;-----------------------------------------------
;
PGR_INIT        Macro
        Local   next_pgr
        MOV     DX , PGR1
        MOV     AX , 0000H
next_pgr:
        OUT     DX , AX
        ADD     DX , 2
        INC     AX
        CMP     DX , PGR64 + 2
        JB      short next_pgr
        Endm
```

保守／廃止

```
;-----------------------------------------------
;   拡張ｱﾄﾞﾚｽ･ﾓｰﾄﾞ切り替え用ﾍﾟｰｼﾞ･ﾚｼﾞｽﾀの設定
;-----------------------------------------------
;
PGR_SET         Macro   cnt
        Local   next_pgr_set , set_n , set_strad
        MOV     AX , cnt
        CMP     AX , 0
        JNE     set_n

        MOV     AX , 0020H
        JMP     set_strad

set_n:  DEC     AL
        MOV     DL , 4
        MUL     DL
        ADD     AX , 0080H

set_strad:
        MOV     DX , PGR33
next_pgr_set:
        OUT     DX , AX
        ADD     DX , 2
        INC     AX
        CMP     DX , PGR36 + 2
        JB      short next_pgr_set
        Endm
;-----------------------------------------------
;       拡張ｱﾄﾞﾚｽ･ﾓｰﾄﾞの設定
;-----------------------------------------------
BRKXA   Macro   vect
        DB      0FH , 0E0H , vect
        Endm
;-----------------------------------------------
;       拡張ｱﾄﾞﾚｽ･ﾓｰﾄﾞの解除
;-----------------------------------------------
RETXA   Macro   vect
        DB      0FH , 0F0H , vect ;RETXAｺﾏﾝﾄﾞ･ｺｰﾄﾞ
        Endm
;-----------------------------------------------
        .LIST
```

保守／廃止

```
        NAME    T9
        Page    60,132
        .186
;*************************************************************************
;*                                                                       *
;*      Extend address TEST PROGRAM - 2                                  *
;*                                      Version 1.00  <1994-03-02>       *
;*************************************************************************
        INCLUDE INIT_REG.ASI    ;通常モード用セグメント定義
        INCLUDE INIT_EXT.ASI    ;拡張モード用セグメント定義
        INCLUDE T6_REG.ASI      ;通常モード用マクロ・プログラム
        INCLUDE T3.ASI          ;拡張モード用マクロ・プログラム
;*************************************************************************
;
;       MAIN PROGRAM
;
_TEXT           Segment
        Assume  CS:CGROUP , DS:_TEST , ES:_TEST
RESET_START:
                CLD                     ;DIRﾌﾗｸﾞをﾘｾｯﾄ(下位→上位)
                CLI                     ;外部割り込みを禁止
;
;***  ベクタの設定  ***
;
                MOV     AX , 0
                MOV     DS , AX

                MOV     SI , 254 * 4    ;ﾍﾞｸﾀ254にXA1のｱﾄﾞﾚｽを設定(BRKXA命令で使用)
                MOV     Word Ptr DS:[SI]   , Offset XA1
                MOV     Word Ptr DS:[SI+2] , Seg  XA1

                MOV     SI , 253 * 4    ;ﾍﾞｸﾀ253にXA2のｱﾄﾞﾚｽを設定(RETXA命令で使用)
                MOV     Word Ptr DS:[SI]   , Offset XA2
                MOV     Word Ptr DS:[SI+2] , Seg  XA2

                MOV     SI , 252 * 4    ;ﾍﾞｸﾀ252にMSG_OUTのｱﾄﾞﾚｽを設定
                MOV     Word Ptr DS:[SI]   , Offset MSG_OUT
                MOV     Word Ptr DS:[SI+2] , Seg  MSG_OUT
;
;*** データ・セグメント・レジスタの設定 ***
;
                MOV     AX , _TEST
                MOV     DS , AX
                MOV     ES , AX
;
;***  イニシャライズ  ***
;
                CPU_INIT                ;(REGULAR.ASI)CPUのｲﾆｼｬﾗｲｽﾞ
                PGR_INIT                ;(REGULAR.ASI)ﾍﾟｰｼﾞ･ﾚｼﾞｽﾀのｲﾆｼｬﾗｲｽﾞ
;
;***  文字表示プログラム  ***
;
mor:            MOV     Word Ptr CS:CNT, 0  ;変数CNTをｸﾘｱ
next_data_tbl:

                MOV     AX , Word Ptr CS:CNT
```

保守／廃止

```
                PGR_SET AX              ;tbl_0～4に使用するﾍﾟｰｼﾞ･ﾚｼﾞｽﾀを設定

                BRKXA   254             ;拡張ｱﾄﾞﾚｽ･ﾓｰﾄﾞの設定
XA1:            INT     252             ;MSG_OUTﾌﾟﾛｸﾞﾗﾑの実行
                RETXA   253             ;拡張ｱﾄﾞﾚｽ･ﾓｰﾄﾞの解除

XA2:            INC     Word Ptr CS:CNT         ;変数CNTをｲﾝｸﾘﾒﾝﾄ
                CMP     Word Ptr CS:CNT, 5      ;変数CNTが5ならばnext_data_tblへ
                                                ;ｼﾞｬﾝﾌﾟ
                JE      mor                     ;5以外ならば次のﾃﾞｰﾀ･ﾃｰﾌﾞﾙをｱｸｾｽ
                JMP     short next_data_tbl

;
;***  変数ＣＮＴ  ***
;
CNT     DW      ?       ;ﾃﾞｰﾀ･ﾃｰﾌﾞﾙをｱｸｾｽするのに使用する変数CNTを確保
;*****************************************************************************
```

保守／廃止

```
;*************************************************************************
;
;       INT 252 ROUTINE
;
MSG_OUT         PROC NEAR

                MOV     BX , Word Ptr CS:CNT
                SHL     BX , 1
                JMP     CS:WORD PTR SET_MSG[BX]
;
;*** ｼﾞｬﾝﾌﾟ･ﾃｰﾌﾞﾙ ***
;
SET_MSG DW      OFFSET  SET_MSG0
        DW      OFFSET  SET_MSG1
        DW      OFFSET  SET_MSG2
        DW      OFFSET  SET_MSG3
        DW      OFFSET  SET_MSG4

SET_MSG0:       MOV     SI , OFFSET MSG0
                JMP     data_set
SET_MSG1:       MOV     SI , OFFSET MSG1
                JMP     data_set
SET_MSG2:       MOV     SI , OFFSET MSG2
                JMP     data_set
SET_MSG3:       MOV     SI , OFFSET MSG3
                JMP     data_set
SET_MSG4:       MOV     SI , OFFSET MSG4

data_set:               MOV DI , 0              ;ﾃﾞｰﾀ･ﾃｰﾌﾞﾙの先頭ｱﾄﾞﾚｽのｵﾌｾｯﾄを
                                                ;設定する

next_data:
        LODS    Byte Ptr CS:[SI]        ;CS:[SI]で示されるﾒﾓﾘ領域よりﾊﾞｲﾄ単位で
                                        ;ﾃﾞｰﾀをALに格納し,SIをｲﾝｸﾘﾒﾝﾄする
        STOS    Byte Ptr ES:[DI]        ;ALの内容をES:[DI]で示されるﾒﾓﾘ領域に
                                        ;ﾊﾞｲﾄ単位で格納し,DIをｲﾝｸﾘﾒﾝﾄする
        CMP     AL , 0
        JNE     next_data

        MOV     SI , 0          ;ﾃﾞｰﾀ･ﾃｰﾌﾞﾙの先頭ｱﾄﾞﾚｽのｵﾌｾｯﾄを設定する
        MON_STR SI              ;(EXTEND.ASI)ﾓﾆﾀ画面に文字ﾃﾞｰﾀを出力する

                IRET
;
;*** 表示ﾒｯｾｰｼﾞ ***
;
MSG0:           DB      "*** Access to Data Block 0 ***" , 0DH , 0AH , 0
MSG1:           DB      "*** Access to Data Block 1 ***" , 0DH , 0AH , 0
MSG2:           DB      "*** Access to Data Block 2 ***" , 0DH , 0AH , 0
MSG3:           DB      "*** Access to Data Block 3 ***" , 0DH , 0AH , 0
MSG4:           DB      "*** Access to Data Block 4 ***" , 0DH , 0AH , 0

                ENDP
;*************************************************************************

_TEXT           Ends

        END     RESET_START
```

保守／廃止

（2）C言語ソース

```
/*-------------------------------------------------------------------------*/
/*      プロトタイプ宣言                                                   */
/*-------------------------------------------------------------------------*/
void    cpu_init ( void );
void    pgr_init ( void );
void    pgr_set ( int );
char    sendc ( char );
/*-------------------------------------------------------------------------*/
/*      ＣＰＵ　初期設定処理                                              */
/*-------------------------------------------------------------------------*/
void    cpu_init( void ) |
#if 1
        outportb( 0xFFFE , 0x10 ) ; /* SCTL  = 00010000B */
        outportb( 0xFFF1 , 0x00 ) ; /* SBCR  = 00000000B */
        outportb( 0xFFF2 , 0x00 ) ; /* RFC   = 01010010B (DRAM=D2H)(PSRAM=DEH) */
        outportb( 0xFFFC , 0xFE ) ; /* OPHA  = 0FEH      */
        outportb( 0xFFFD , 0x08 ) ; /* OPSEL = 00001111B */
        outportb( 0xFFFB , 0x50 ) ; /* DULA  = 50H       */
        outportb( 0xFFFA , 0x40 ) ; /* IULA  = 40H       */
        outportb( 0xFFF9 , 0x30 ) ; /* TULA  = 30H       */
        outportb( 0xFFF8 , 0x20 ) ; /* SULA  = 20H       */
        outportb( 0xFFF0 , 0x00 ) ; /* TCKS  = 00000000B */
        outportb( 0xFFEA , 0x55 ) ; /* WMB0  = 01010101B */
        outportb( 0xFFEC , 0x00 ) ; /* WCY0  = 00000000B */
        outportb( 0xFFEB , 0x13 ) ; /* WCY1  = 00010010B */
        outportb( 0xFFED , 0x00 ) ; /* WAC   = 00000000B */
        outportb( 0xFFF3 , 0x75 ) ; /* WMB1  = 01110101B */
        outportb( 0xFFF4 , 0x20 ) ; /* WCY2  = 00100000B */
        outportb( 0xFFF5 , 0x72 ) ; /* WCY3  = 00100010B */
        outportb( 0xFFF6 , 0x21 ) ; /* WCY4  = 00100001B */
#endif
        outportb( 0xFFE9 , 130 )   ; /* BRC  = 130:20MHz -> 9600bps */
|
/*-------------------------------------------------------------------------*/
/*      ページ・レジスタ制御                                               */
/*-------------------------------------------------------------------------*/
/* page register I/O address */
#define PGR1_ADD    0xff00
#define PGR33_ADD       0xff40
#define PGR36_ADD       0xff46
#define PGR64_ADD       0xff7e
int     io_add , data;
```

保守／廃止

```
/*-------------------------------------------------------*/
/*      ページ・レジスタ初期設定処理                     */
/*-------------------------------------------------------*/
void    pgr_init( void ) {
                data = 0x0000;
                for( io_add = PGR1_ADD ; (unsigned)io_add != ( PGR64_ADD + 2 )
; io_add += 2 ){
                        outport( io_add , data );
                        data ++;
                }
        }

/*-------------------------------------------------------*/
/*      ページ・レジスタ設定                             */
/*-------------------------------------------------------*/
void    pgr_set( int cnt ) {

                if( cnt == 0 ){
                        data = 0x0020;
                }
                else{
                        data = 0x0080 + ( cnt - 1 ) * 4;
                }

                for( io_add = PGR33_ADD ; (unsigned)io_add != ( PGR36_ADD + 2 )
; io_add += 2 ){
                        outport( io_add , data );
                        data ++;
                }
        }
/*-------------------------------------------------------*/
/*      ＳＣＵアドレス,コマンド                         */
/*-------------------------------------------------------*/
/* register address */
#define SCU_STS 0xFE22          /* STATUS     REGISTER (READ)  */
#define SCU_TXD 0xFE20          /* 送信データ REGISTER (WRITE) */
/*-------------------------------------------------------------*/
/*      １文字表示プログラム                                   */
/*-------------------------------------------------------------*/
char    sendc( char c ) {
        while( (inportb( SCU_STS ) & 0x01) == 0 ) ;
        outportb( SCU_TXD , c ) ;
        return( c ) ;
}
```

保守／廃止

```
#pragma inline
#include <dos.h>
#include <string.h>
/*************************************/
void    main ( void )   ;
void    vector_init ( void );
void    cpu_init ( void ) ;
void    pgr_set ( int ) ;
void    msg_out  ( void ) ;
void    sendc( char );
void    brkxa( void );
void    xa ( void );
void    retxa ( void );
int     cnt;
/*************************************/
void main( void ) {
        vector_init ();
        cpu_init() ;
        pgr_init() ;
        cnt = 0;
        while( cnt != 5 ){
                pgr_set ( cnt );
                brkxa();
                cnt++;
        }
}

void    msg_out( void ){
        char    far     *ex_mem;
        ex_mem = ( char far* )0x80000000;

        switch( cnt ){
                case 0:
                        strcpy( ex_mem , "*** Access to Data Block 0 ***¥r¥n"
);
                        break;
                case 1:
                        strcpy( ex_mem , "*** Access to Data Block 1 ***¥r¥n"
);
                        break;
                case 2:
                        strcpy( ex_mem , "*** Access to Data Block 2 ***¥r¥n"
);
                        break;
                case 3:
                        strcpy( ex_mem , "*** Access to Data Block 3 ***¥r¥n"
);
                        break;
                case 4:
                        strcpy( ex_mem , "*** Access to Data Block 4 ***¥r¥n"
);
                        break;
        }

        while( *ex_mem != '¥0' ){
                sendc( *ex_mem );
```

保守／廃止

```
                ex_mem++;
        }
}
/*------------------------------------------------------------------------*/
/*      拡張アドレス・モード・スイッチング・プログラム                    */
/*------------------------------------------------------------------------*/
void    brkxa ( void ){
        {
        }
        asm     db      0fh , 0e0h , 0feh;      /*brkxa*/
}

void    xa ( void ){
        {
                msg_out ();
        }
        asm     db      0fh , 0f0h , 0fdh;      /*retxa*/
}

void    retxa ( void ){
}
/*------------------------------------------------------------------------*/
/*      ベクタの設定                                                      */
/*------------------------------------------------------------------------*/
void    vector_init ( void ){
        poke( 0 , 0x0fe*4 , FP_OFF( xa ) ) ;
        poke( 0 , 0x0fe*4 + 2 , FP_SEG( xa ) ) ;
        poke( 0 , 0x0fd*4 , FP_OFF( retxa ) ) ;
        poke( 0 , 0x0fd*4 + 2 , FP_SEG( retxa ) ) ;
}
```

（メ　モ）

保守／廃止

付録A　動作確認用ハードウエア

このアプリケーション・ノートで示したプログラム例は、V53A　アプリケーション・ノート　ハードウエア設計編で示したV53Aボードをターゲットにしています。ここではV53Aボードの構成と機能について説明します。

A.1　仕　　様

V53Aボードは次の仕様に基づいて設計しています。

（1）主要デバイス

・CPU　　：μPD70236A（1つ）

【メモリ】

・PROM　　：μPD27C1001A（2つ）
・SRAM　　：μPD431008（4つ）
・DRAM　　：μPD42S18160（1つ）
・疑似SRAM　　：μPD428128（2つ）
・ページ機能付きPROM　：μPD27C4040（1つ）
・フラッシュ・メモリ　：μPD28F4000（1つ）

【周辺デバイス】

・シリアル・コントロール・ユニット　：μPD71051（1つ）
・LCDコントローラ　：μPD72030（1つ）
・フロッピィ・ディスク・コントローラ　：μPD72069（1つ）
・浮動小数点演算用コプロセッサ　：μPD72291（1つ）
・カレンダ時計　：μPD4991（1つ）
・パラレル・インタフェース・ユニット　：μPD71055（1つ）

（2）外部インタフェース

・RS-232-Cインタフェース（2チャネル）
・LCDモジュール
・PC-9800シリーズ用キーボード・インタフェース（1チャネル）
・バー・コード・リーダ・インタフェース（RS-232-Cインタフェース兼用）
・FDDインタフェース
・プリンタ・インタフェース

保守／廃止

（3）電源電圧

CPU，周辺デバイス：V_{DD} = ＋5 V

備考　CPUとメモリを3 Vで，外部I/Oを5 Vで動作させる場合の例はV53A　アプリケーション・ノート　ハードウエア設計編　第5章　レベル変換回路例を参照してください。

（4）回路仕様

・V53Aの最大動作周波数：20 MHz（μPD72291を使用しないとき）注1

・各デバイスに対するウエイト数注2

デバイス	ウエイト数
PROM	2ウエイト
SRAM	ノー・ウエイト
DRAM	1ウエイト
疑似SRAM	3ウエイト
ページ機能付きPROM	2ウエイト（ページ外アクセス） 1ウエイト（ページ内アクセス）
フラッシュ・メモリ	3ウエイト
シリアル・コントロール・ユニット	7ウエイト
LCDコントローラ	
フロッピィ・ディスク・コントローラ	
カレンダ時計	
パラレル・インタフェース・ユニット	

注1．μPD72291を使用する場合は，最大16 MHzです。

V53Aボードでは，μPD72291を使用するかどうかをスイッチで切り替えることができます（V53A　アプリケーション・ノート　ハードウエア設計編　第4章　コプロセッサ・インタフェース例参照）。

2．ウエイト・サイクルの挿入には，次の2通りの方法があります。

V53Aボードでは，ページ機能付きPROMは（1）の方法で，それ以外の各デバイスは（2）の方法で制御しています。

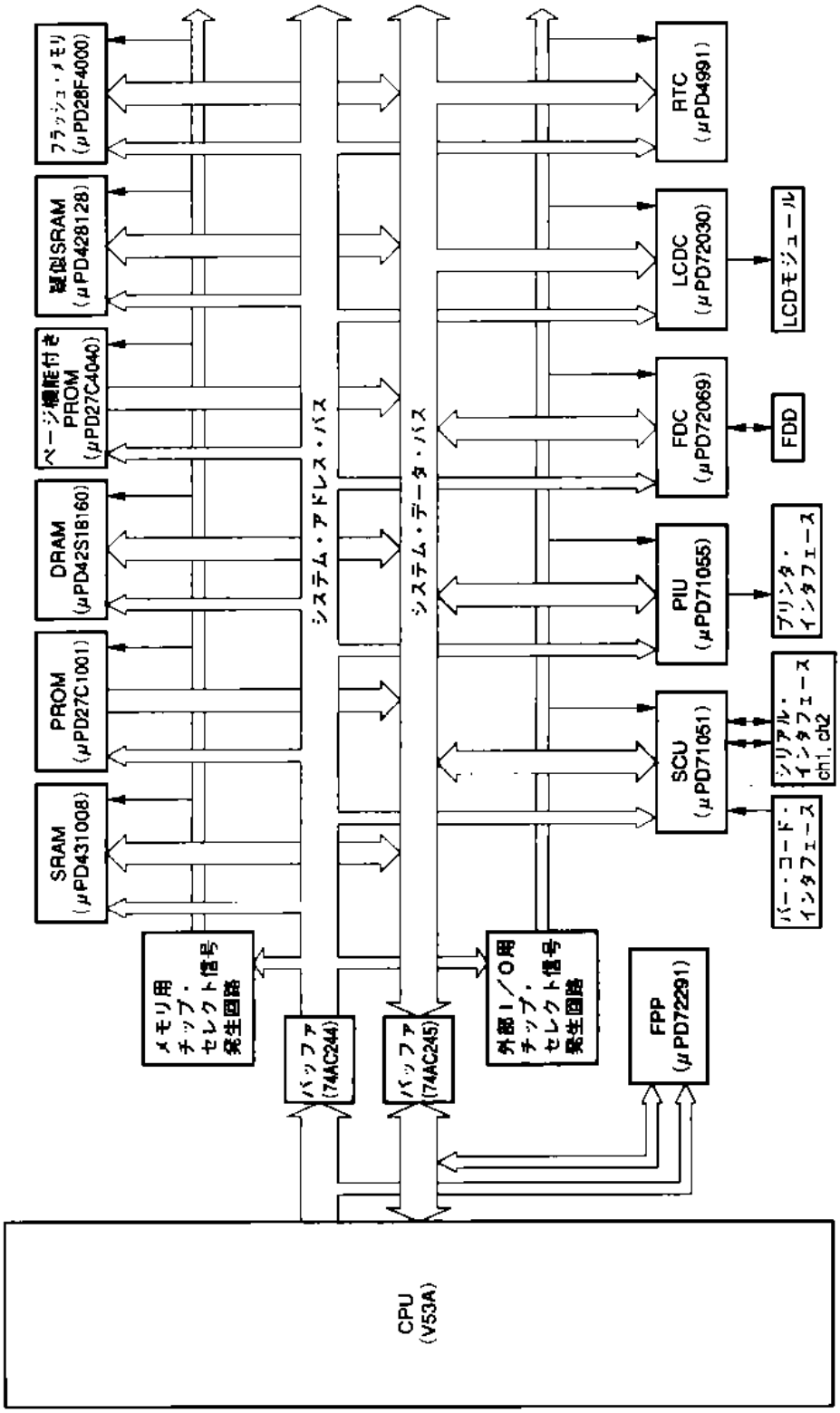
（1）ハードウエア的な方法…$\overline{\text{READY}}$端子を使用

（2）ソフトウエア的な方法…ウエイト・コントロール・ユニット（WCU）を使用

保守／廃止

A.2　構　　成

図A－1にV53Aボードのシステム・ブロック図を示します。



図A－1　システム・ブロック図

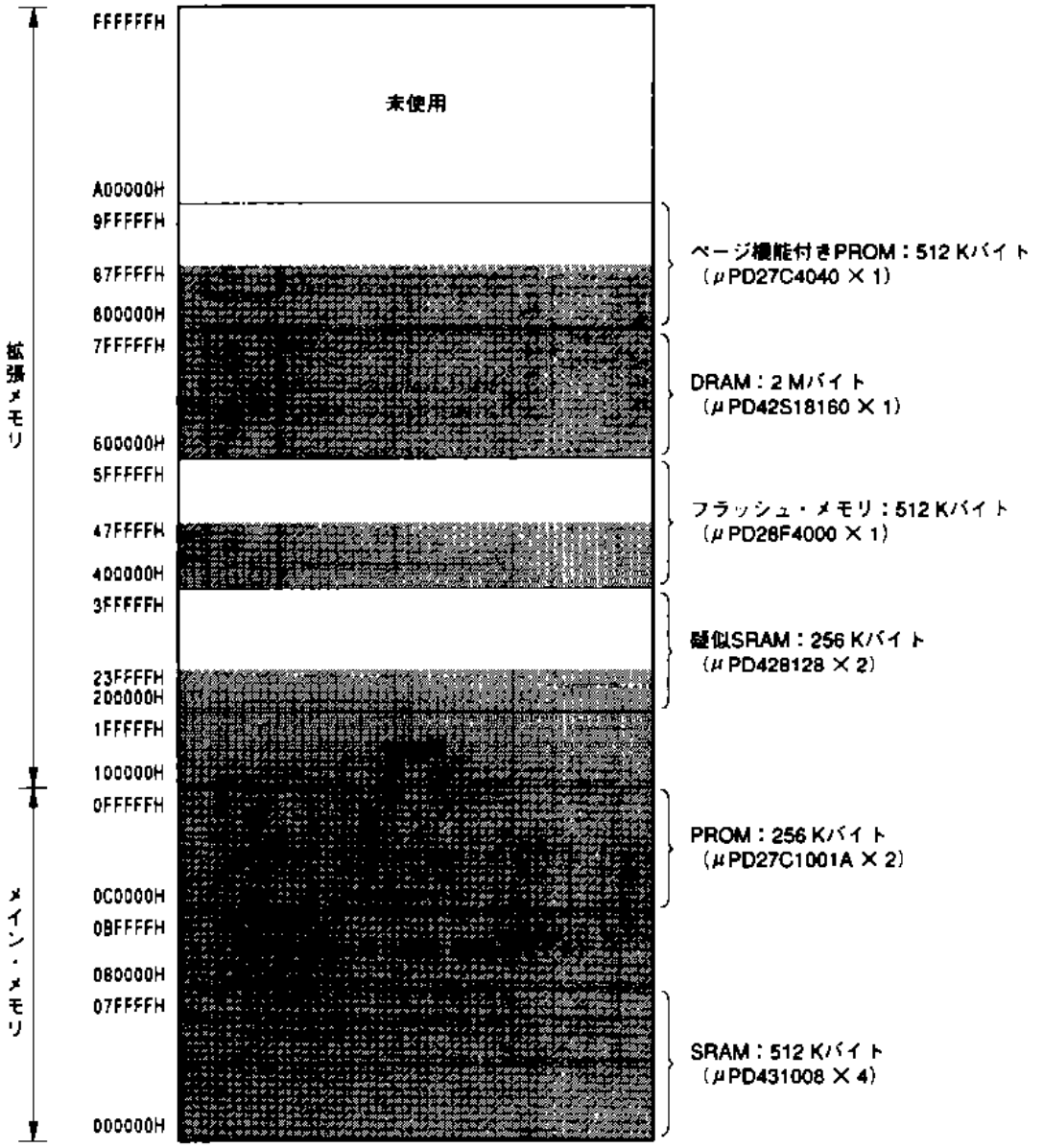
保守／廃止

A.3　メモリ・マップ

図A－2にV53Aボードのメモリ・マップを示します。

メイン・メモリ内の80000H-BFFFFHに切り替え用メモリ領域があります。ボード上にあるジャンパ（JP2, JP3）を切り替えることにより，拡張メモリ領域に割り当てられている各種メモリがアクセスできます（表A－1参照）。

図A－2　メモリ・マップ



備考　　のアドレスは実装領域（他はイメージ領域）

保守／廃止

表A－1　JP2, JP3の設定

JP2	JP3	アクセス・メモリ	アドレス
1	1	フラッシュ・メモリ	400000H-43FFFFH
1	0	疑似SRAM	200000H-23FFFFH
0	1	DRAM	680000H-6BFFFFH
0	0	ページ機能付きPROM	800000H-83FFFFH

備考　1：接続、0：オープン

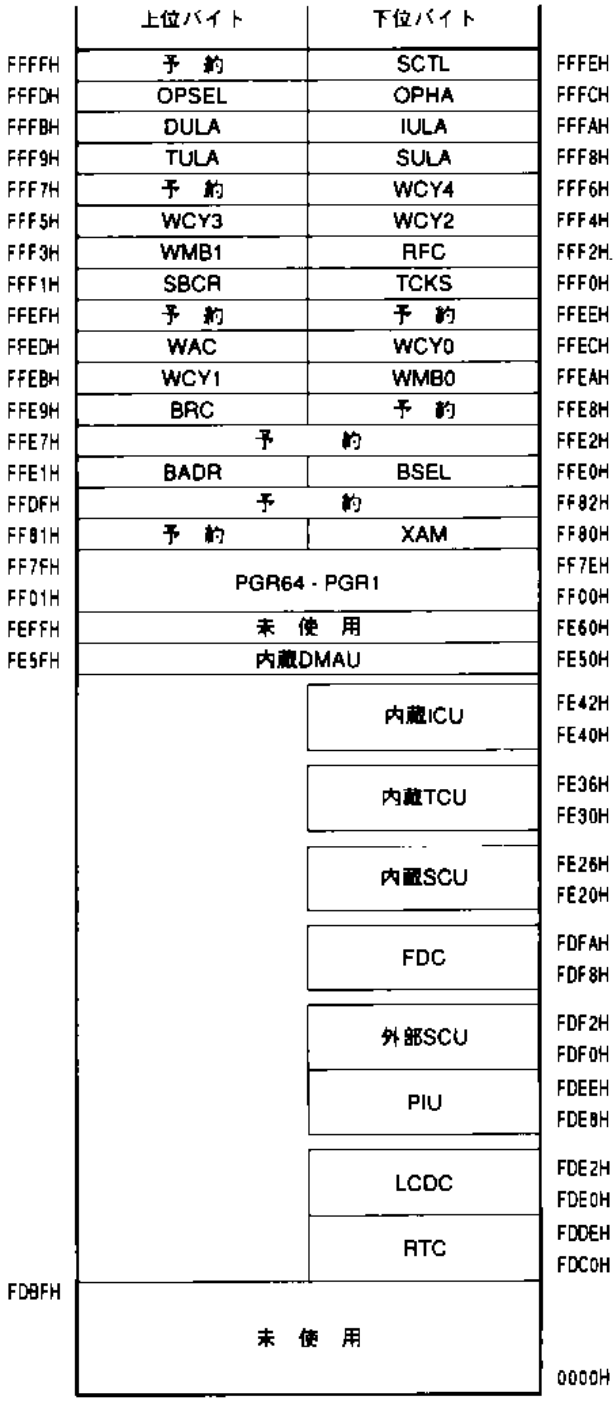
保守／廃止

A.4　I/Oマップ

V53Aは64 Kバイトまでの I/Oを，メモリとは独立した領域でアクセスすることができます。

図A－3にV53Aボードの I/Oマップを示します。

図A－3　I/Oマップ



保守／廃止

付録B　ソフトウエア開発環境

このアプリケーション・ノートで示したプログラム例は、Borland International inc. のTurbo C, Turbo Assemblerでコンパイル、アセンブルできるように記述しています。Turbo Assemblerと16ビットVシリーズの命令セットの対応を次に示します。

保守／廃止

表B－1　Turbo Assemblerとのニモニック対応表

Turbo Assembler	16ビット Vシリーズ	Turbo Assembler	16ビット Vシリーズ	Turbo Assembler	16ビット Vシリーズ	Turbo Assembler	16ビット Vシリーズ
AAA	ADJBA	JB	BC/BL	LOOP	DBNZ	SHR	SHR
AAD	CVTDB	JBE	BNH	LOOPE	DBNZE	SS:	SS:
AAM	CVTBD	JC	BC/BL	LOOPNE	DBNZNE	STC	SET1 CY
AAS	ADJBS	JCXZ	BCWZ	LOOPNZ	DBNZNE	STD	SET1 DIR
ADC	ADDC	JE	BE/BZ	LOOPZ	DBNZE	STI	EI
ADD	ADD	JG	BGT	MOV	MOV	STOS	STM/STMB/
AND	AND	JGE	BGE	MOVS	MOVBK		STMW
CALL	CALL	JL	BLT	MOVSB	MOVBKB	SUB	SUB
CBW	CVTBW	JLE	BLE	MOVSW	MOVBKW	TEST	TEST
CLC	CLR1 CY	JMP	BR	MUL	MULU	WAIT	POLL
CLD	CLR1 DIR	JNA	BNH	NEG	NEG	XCHG	XCH
CLI	DI	JNAE	BC/BL	NOP	NOP	XLAT	TRANS
CMC	NOT1 CY	JNB	BNC/BNL	NOT	NOT	XLATB	TRANSB
CMP	CMP	JNBE	BH	OR	OR	XOR	XOR
CMPS	CMPBK/	JNC	BNC/BNL	OUT	OUT	—	ADD4S
	CMPBKB/	JNE	BNE/BNZ	POP	POP	—	BRKEM
	CMPBKW	JNG	BLE	POPF	POP PSW	—	BRKXA
CS:	PS:	JNGE	BLT	PUSH	PUSH	—	CALLN
CWD	CVTWL	JNL	BGE	PUSHF	PUSH PSW	—	CHKIND
DAA	ADJ4A	JNLE	BGT	RCL	ROLC	—	CMP4S
DAS	ADJ4S	JNO	BNV	RCR	RORC	—	DISPOSE
DEC	DEC	JNP	BPO	REP	REP	—	EXT
DIV	DIVU	JNS	BP	REPE	REPE	—	FPO2
DS:	DS0:	JNZ	BNE/BNZ	REPNE	REPNE	—	INM
ES:	DS1:	JO	BV	REPNZ	REPNZ	—	INS
ESC	FPO1	JP	BPE	REPZ	REPZ	—	OUTM
HLT	HALT	JPE	BPE	RET	RET	—	PREPARE
IDIV	DIV	JPO	BPO	ROL	ROL	—	REPC
IMUL	MUL	JS	BN	ROR	ROR	—	REPNC
IN	IN	JZ	BE/BZ	SAHF	MOV PSW, AH	—	RETEM
INC	INC	LAHF	MOV AH, PSW	SAL	SHL	—	RETXA
INT	BRK	LDS	MOV DS0,	SAR	SHRA	—	ROL4
INT 3	BRK 3	LEA	LDEA	SBB	SUBC	—	ROR4
INTO	BRKV	LES	MOV DS1,	SCAS	CMPM/	—	SUB4S
IRET	RETI	LOCK	BUSLOCK		CMPMB/	—	TEST1
JA	BH	LODS	LDM/LDMB/		CMPMW		
JAE	BNC/BNL		LDMW	SHL	SHL		

備考　－：該当する命令なし

保守／廃止

アンケート記入のお願い

お手数ですが，このドキュメントに対するご意見をお寄せください。今後のドキュメント作成の参考にさせていただきます。

［ドキュメント名］V53A　アプリケーション・ノート　アドレス拡張ソフトウエア編
（U10188JJ2V0AN00（第２版））

［お名前など］（さしつかえのない範囲で）

御社名（学校名，その他）（　　）
ご住所（　　）
お電話番号（　　）
お仕事の内容（　　）
お名前（　　）

１．ご評価（各欄に○をご記入ください）

項　　目	大変良い	良　い	普　通	悪　い	大変悪い
全体の構成					
説明内容					
用語解説					
調べやすさ					
デザイン，字の大きさなど					
その他（　　）					
（　　）					

２．わかりやすい所（第　章，第　章，第　章，第　章，その他　　）
理由［　　］

３．わかりにくい所（第　章，第　章，第　章，第　章，その他　　）
理由［　　］

４．ご意見，ご要望

５．このドキュメントをお届けしたのは
NEC販売員，特約店販売員，NEC半導体ソリューション技術本部員，
その他（　　）

ご協力ありがとうございました。
下記あてにFAXで送信いただくか，最寄りの販売員にコピーをお渡しください。

NEC半導体インフォメーションセンター
FAX：（044）548-7900

キリトリ

保守／廃止

お問い合わせは，最寄りのNECへ

【営業関係お問い合わせ先】

半導体第一販売事業部 半導体第二販売事業部 半導体第三販売事業部	〒108-01 東京都港区芝五丁目7番1号（NEC本社ビル）	東 京	(03)3454-1111 （大代表）
中部支社 半導体販売部	〒460 名古屋市中区錦一丁目17番1号（NEC中部ビル）	名古屋	(052)222-2170
関西支社 半導体第一販売部 半導体第二販売部 半導体第三販売部	〒540 大阪市中央区城見一丁目4番24号（NEC関西ビル）	大 阪 大 阪 大 阪	(06) 945-3178 (06) 945-3200 (06) 945-3208

北海道支社	札 幌	(011)231-0161	小山支店	小 山	(0285)24-5011	富山支店	富 山 (0764)31-8461
東北支社	仙 台	(022)261-6511	長野支社	長 野	(0262)35-1444	三重支店	津 (0592)28-7341
盛岡支店	盛 岡	(0196)51-4344	松本支店	松 本	(0263)35-1866	京都支社	京 都 (075)344-7824
山形支店	山 形	(0236)23-5511	上越防支店	上越防	(0256)53-5350	神戸支社	神 戸 (078)333-3884
郡山支店	郡 山	(0249)23-5511	甲府支店	甲 府	(0552)24-4141	中国支社	広 島 (082)242-6504
いわき支店	いわき	(0246)21-5511	埼玉支社	大 宮	(048)641-1411	鳥取支店	鳥 取 (0857)27-5311
長岡支店	長 岡	(0258)36-2155	立川支支社	立 川	(0425)26-5981	岡山支店	岡 山 (086)225-4455
土浦支店	土 浦	(0298)23-6161	千葉支支社	千 葉	(043)238-8116	四国支社	高 松 (0878)36-1200
水戸支店	水 戸	(0292)26-1717	静岡支支社	静 岡	(054)255-2211	新居浜支店	新居浜 (0897)32-5001
神奈川支社	横 浜	(045)324-5511	沼津支支店	沼 津	(0559)63-4455	松山支店	松 山 (0899)45-4111
群馬支店	高 崎	(0273)26-1255	浜松支支店	浜 松	(053)452-2711	九州支社	福 岡 (092)271-7700
太田支店	太 田	(0276)46-4011	北陸支社	金 沢	(0762)23-1621	北九州支店	北九州 (093)541-2887
宇都宮支店	宇都宮	(0286)21-2281	福井支店	福 井	(0776)22-1865		

【本資料に関する技術お問い合わせ先】

半導体ソリューション技術本部 マイクロコンピュータ技術部	〒210 川崎市幸区堀川町三丁目464番地	川 崎	(044)548-8890	半導体 インフォメーションセンター FAX(044)548-7900 （FAXにてお願い致します）
半導体販売技術本部 東日本販売技術部	〒108-01 東京都港区芝五丁目7番1号（NEC本社ビル）	東 京	(03)3798-9819	
半導体販売技術本部 中部販売技術部	〒460 名古屋市中区錦一丁目17番1号（NEC中部ビル）	名古屋	(052)222-2125	
半導体販売技術本部 関西日本販売技術部	〒540 大阪市中央区城見一丁目4番24号（NEC関西ビル）	大 阪	(06) 945-3383	