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

# アプリケーション・ノート

保守／廃止

# V55PI™

## 16ビット・マイクロプロセッサ

## ファクシミリ編

## μPD70433

資料番号　U13256JJ2V0AN00（第2版）
（旧資料番号　IEA-727）
発行年月　January 1998　N　CP(K)

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。

保守／廃止

本資料に掲載の応用回路および回路定数は，例示的に示したものであり，量産設計を対象とするものではありません。

○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。
○本資料に記載された製品の使用もしくは本資料に記載の情報の使用に際して，当社は当社もしくは第三者の知的所有権その他の権利に対する保証または実施権の許諾を行うものではありません。上記使用に起因する第三者所有の権利にかかわる問題が発生した場合，当社はその責を負うものではあり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.3 | 表1－1　基本機能　修正 |
| p.12 | 図2－2　メモリ・インタフェースの概略図　修正 |
| p.14 | 表2－2　ポート割り付け表　修正 |
| p.19 | 表2－5　ポート2の動作（n＝0－5）　修正 |
| p.26 | 2.1.3 (2) (g) ポート6（P60-P63）　修正 |
| p.41 | 2.2.2 タイマ　修正 |
| p.57 | 2.2.4 A/Dコンバータ　追加 |
| p.101 | 付録A　コーディング・リスト　修正 |
| p.297 | 付録B (2) CPU周り　修正 |
| p.301 | 付録B (4) I/O CIRCUIT FOR VAR.CNS　修正 |
| p.308 | 付録B　品名一覧　修正 |
| p.313 | 付録C　コネクタの接続　追加 |

本文欄外の★印は，本版で改訂された主な箇所を示しています。

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。

# はじめに

**対象者**　このアプリケーション・ノートは，μPD70433（別名称V55PI）の機能を理解し，それを用いたファクシミリ・システムを設計するユーザのエンジニアを対象とします。

**目　的**　このアプリケーション・ノートは，次の構成に示すμPD70433の持つハードウエア，ソフトウエア機能を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**　このアプリケーション・ノートは大きく分けて次の内容で構成しています。

・ハードウエア編

・ソフトウエア編

**読み方**　このアプリーケーション・ノートの読者には，電気，論理回路，マイクロコンピュータの一般知識を必要とします。さらに理解を深めていただくために，**V55PI　ユーザーズ・マニュアル**をお読みください。なお，このアプリケーション・ノートは，μPD70433という製品名をV55PIの名称で説明してあります。

**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 |

**関連資料**　関連資料は暫定版の場合がありますが，この資料では「暫定」の表示をしておりません。あらかじめご了承ください。

- パンフレット（U10244J）
- データ・シート（U11775J）
- ユーザーズ・マニュアル　ハードウエア編（U10514J）
- 〃　命令編（U10231J）
- アプリケーション・ノート　ソフトウエア編（IEA-743）
- 〃　ファクシミリ編（このマニュアル）
- 〃　ハードウエア編（IEA-750）

保守／廃止

開発ツール関連資料一覧

| | 資料名 | | 資料番号 |
|---|---|---|---|
| ハードウエア | IE-70433-BX　ユーザーズ・マニュアル | | U10608J |
| ソフトウエア | RA70116-I, SP70116-I | | |
| | ユーザーズ・マニュアル | アセンブリ言語編 | EEU-861 |
| | | 操作編 | EEU-869 |

保守／廃止

# 目　次　要　約



★

保守／廃止

〔メ モ〕

保守／廃止

# 目　　次

保守／廃止

# 図　の　目　次

# 表　の　目　次

保守／廃止

# 第１編　ハードウエア

保守／廃止

〔メ モ〕

保守／廃止

# 第１章　システム概要

この章では，V55PIを使用し，作成したファクシミリ（以下FAX）の機能とシステムの構成について説明します。

## １．１　基本機能

このFAXの機能を表１－１に示します。

表１－１　基本機能 ★

| 機　　能 | 概　　要 |
|---|---|
| 伝送速度 | 9600/7200/4800/2400/（300）bps<br>（300 bpsは，HDLCによるハンドシェーク時のみ） |
| 符号化方式 | MH/MR |
| 通信方式 | ITU-T（旧CCITT）　T30　準拠 |
| 接続回線 | PSTN（一般公衆回線）<br>（音響カプラによる音響接続可能） |
| 最小伝送速度 | 10 ms/１ライン |
| 読み取り方式 | 密着イメージ・センサによる読み取り |
| 最大読み取り幅 | A4（216 mm） |
| 分解能 | 主走査線：8 dot/mm<br>副走査線：3.85 dot/mm（normal）<br>7.7 dot/mm（fine） |
| 印字方式 | サーマル・プリント・ヘッドによる感熱記録方式 |
| 最大印字幅 | A4（216 mm） |
| 記録紙長 | 18 m |
| 本体重量 | 2.8 kg |

## 1.2　ハードウエア構成

このFAXは次の4つの部分から構成されています。図1-1にブロック図を示します。

(1) **コントローラ部**

- ●μPD70433（別名称V55PI）
- ●メモリ（μPD43256，μPD428128，μPD27C1001）
- ●リアルタイム・クロック（μPD4991A）
- ●FPGA
- ●リセットIC
- ●RS-232-Cドライバ／レシーバ

(2) **画像読み取り部**

- ●密着センサ部
- ●画処理部

(3) **回線インタフェース部**

- ●モデム
- ●NCU

(4) **記録部**

- ●モータ・ドライバ
- ●パワー・リレー

図1-1　ブロック図

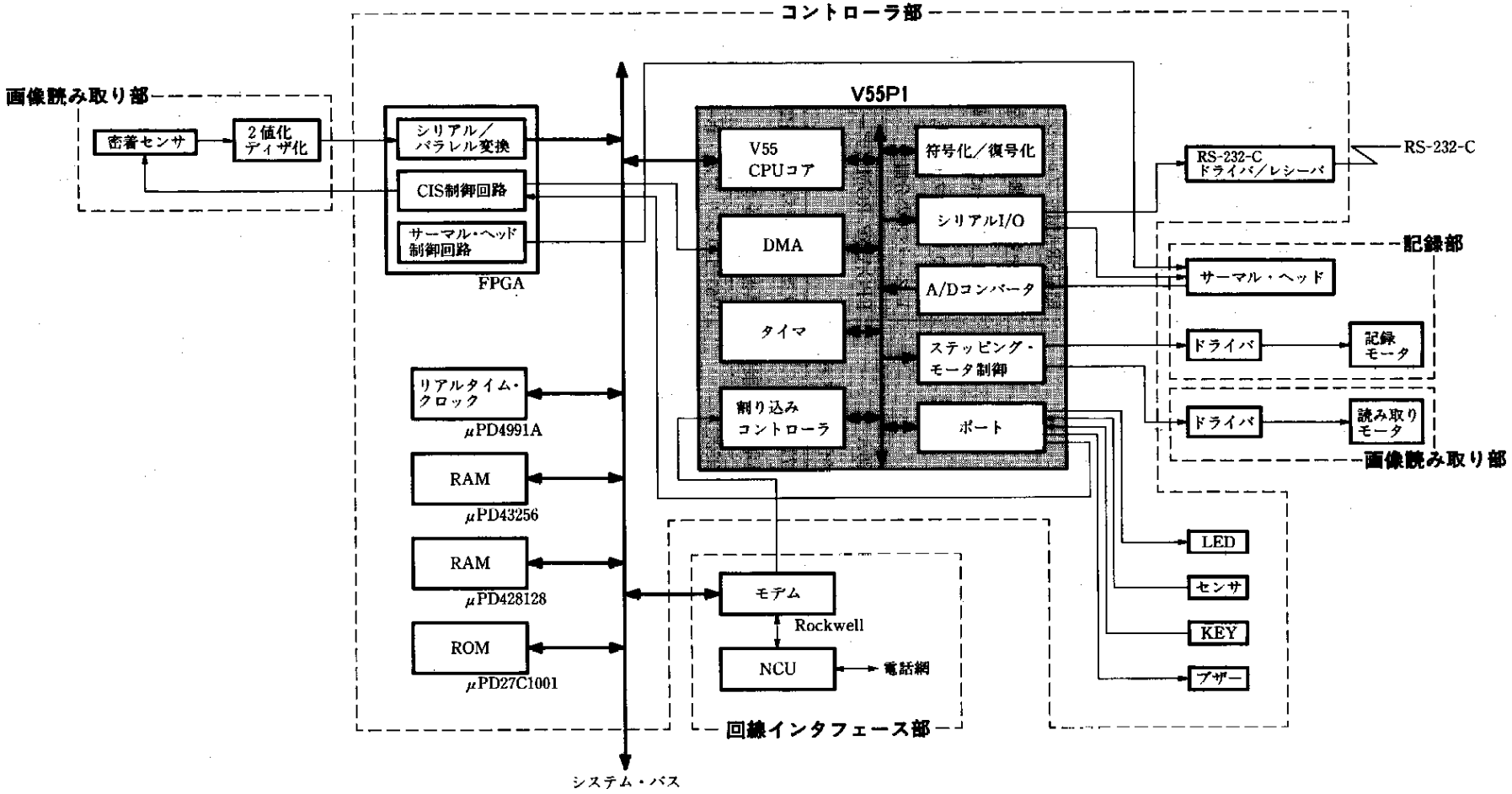
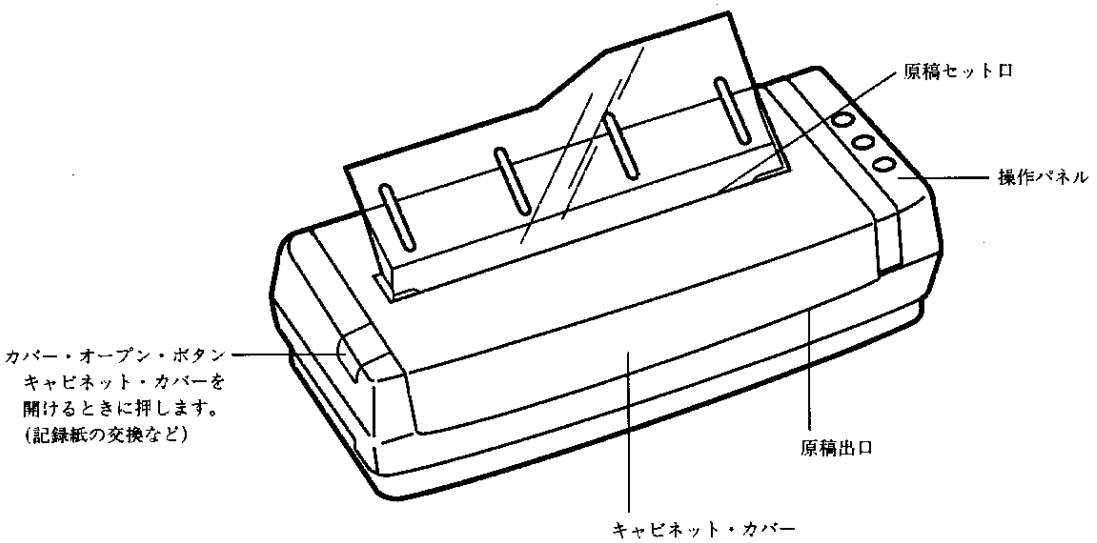


表1－2　ハードウエア構成

| 処理ブロック | 分　類 | 品　名 | 機　能 |
|---|---|---|---|
| コントローラ部 | CPU | μPD70433 | バス幅：8ビット，外部クロック：24 MHz |
| | メモリ | μPD43256 | ワーク用。また，電池によりバックアップし，ユーザ・ネーム，伝送速度，電話番号を保持。 |
| | | μPD428128 | ECM時の符号化データ・バッファ用。 |
| | | μPD27C1001 | プログラム用。 |
| | リアルタイム・クロック | μPD4991A | ヘッダ用の時計（電池によりバックアップ）。 |
| | FPGA | XC3030<br>(米，ザイリンクス社製) | V55PIの制御信号（$\overline{\text{ASTB}}$，$\overline{\text{RD}}$，$\overline{\text{WRL}}$など）からサーマル・ヘッド用ストローブ，センサ用タイミング信号，メモリ・セレクト信号，疑似SRAMリフレッシュ信号を生成。密着センサからの画像データをシリアル／パラレル変換し，DMAに対して転送要求を発生。メモリとのインタフェース，データ・バス／アドレス・バスの分離。 |
| | | XC1736A<br>(米，ザイリンクス社製) | FPGA用P-ROM（データ保持用）。 |
| | その他<br>外部デバイス | MB3771<br>（三菱電機㈱社製） | リセット・パルスを発生。<br>誤書き込みを防止。 |
| | | MAX232C<br>（米，マキシム社製） | V55PIのシリアル・ポートのTxD，RxDをRS-232-C用信号に変換。 |
| 画像読み取り部 | 密着センサ | — | FPGAで生成したタイミング信号に同期して，アナログ信号（画データ）を出力。<br>この信号を2値化してFPGAへ供給。 |
| | 画処理部 | — | センサからの出力信号をクロック・タイミングでサンプル・ホールド，クランプ，および増幅を行い，アナログ信号(画データ)を2値化。 |
| 回線インタフェース部 | モデム | P96EFX<br>(米，ロックウエル社製) | HDLC機能内蔵，9600 bps。<br>V55PIとバス直結インタフェースで，割り込み機能を併用して接続。 |
| | NCU | — | 回線制御。音響カプラ切り替え回路内蔵。 |
| 記録部 | モータ・ドライバ | TD62107F<br>(㈱東芝社製) | ステッピング・モータ駆動用（24 V/100 Ωのコイルをもつモータを駆動）。 |
| | パワー・リレー | | サーマル・ヘッドへの電源用リレー。<br>接点電流：5 A（サーマル・ヘッドの最大負荷が，4.5 Aのため） |

次に，このFAXのシステム構成を示します。

図1－2　システム構成



保守／廃止

〔メ モ〕

# 第２章　コントローラ部

この章では，コントローラ部について説明します。

## ２．１　CPU（V55PI）

このFAXは，CPUとしてV55PI（μPD70433）を搭載しています。

### ２．１．１　V55PI概要

V55PIは，おもにメイン制御，ステッピング・モータ制御，コーデック処理を行います。

このFAXで使用しているV55PIの内蔵周辺の機能を次に示します。

| | |
|---|---|
| DMAコントローラ | ：画データ取得用 |
| シリアル・ポート | ：サーマル・ヘッドへの画データ転送用 |
| タイマ | ：汎用タイマ |
| A/Dコンバータ | ：サーマル・ヘッドの温度検出用 |
| リアルタイム出力ポート | ：ステッピング・モータ制御用 |
| ウォッチドッグ・タイマ | ：暴走検出用 |
| 割り込みコントローラ | ：モデム割り込み |

保守／廃止

## 2.1.2　メモリ・インタフェース

SRAM, 疑似SRAM, EPROMとのインタフェース・メモリ・マップを図2-1に, メモリ・インタフェースの概略図を図2-2に示します。

### (1) SRAM

●SRAM L

ユーザ・ネーム, 伝送速度, 電話番号のバックアップ用として使用しています。そのため, 電源ON/OFF時の誤書き込み防止用に, $\overline{\text{RESET}}$信号を利用し, $\overline{\text{RESET}}$パルス間はアクセス禁止にしています。

●SRAM U

ワーク用として使用しています。

バックアップの必要がないため, アドレス・デコーダの信号を直接チップ・セレクト端子へ入力して使用しています。

### (2) 疑似SRAM

ECM時の符号化データ・バッファ用として使用しています。

プリチャージ・タイムを考慮する必要があるため, アドレス・デコード信号と$\overline{\text{ASTB}}$との論理和(OR)を取って, $\overline{\text{CE}}$入力としています。

リフレッシュは, $\overline{\text{CE}}$オンリー・リフレッシュを採用しています。

### (3) EPROM

プログラム用として使用しています。

$\overline{\text{CE}}$を常にグランドに落とし, アドレス・デコード信号と$\overline{\text{RD}}$信号の論理和(OR)を取って, $\overline{\text{OE}}$接続します。それにより, アクセス・タイムを短くしています。

各メモリ・ブロック分割, ウエイト挿入サイクルは, 以下のとおりです。レジスタ設定値は,「**2.1.4 初期設定**」を参照してください。

**表2-1　各メモリ・ブロックのウエイト数**

| メモリ・ブロック | メ　モ　リ | ウエイトの種類 | ウエイト数 |
|---|---|---|---|
| ブロック0 | SRAM | データ・ウエイト | 2 |
| ブロック1 | 疑似SRAM | | 1 |
| ブロック2 | メモリ・マップトI/O | | 1 |
| ブロック3 | EPROM | | 2 |

図2-1　メモリ・マップ

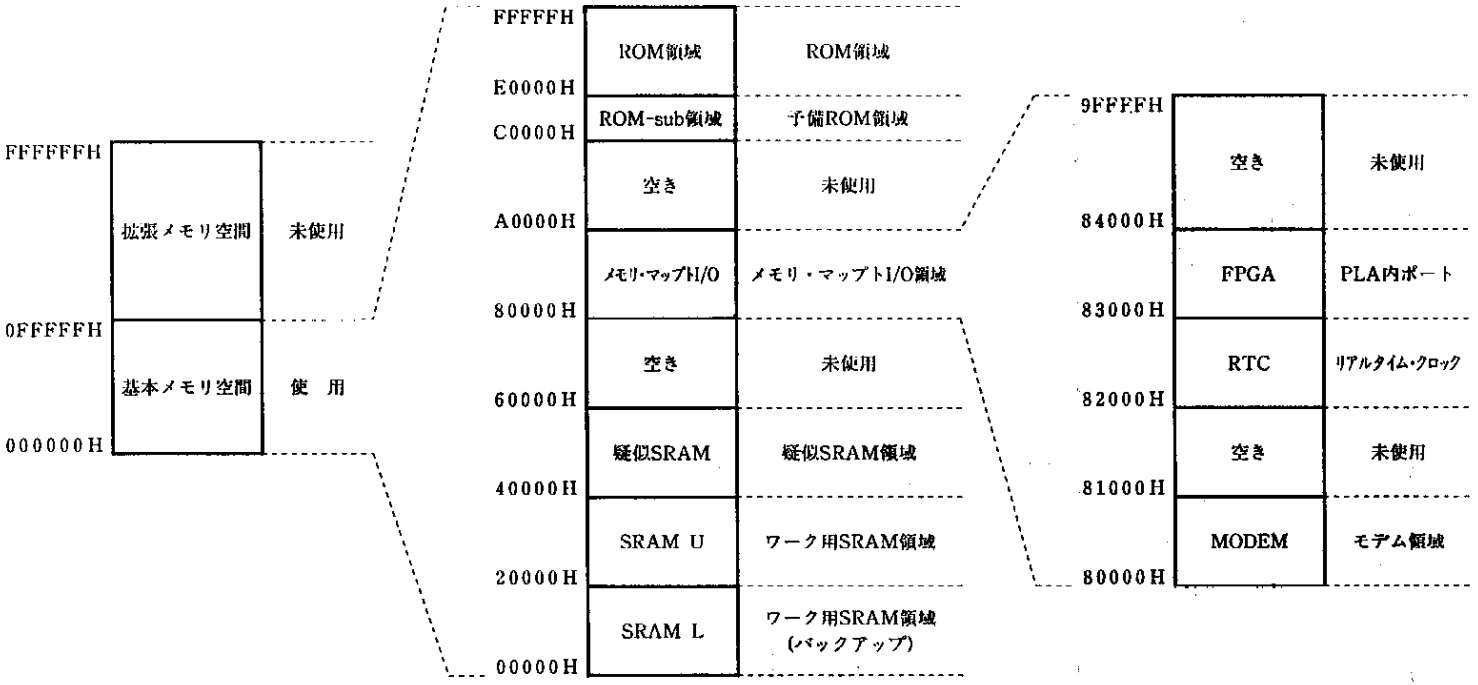
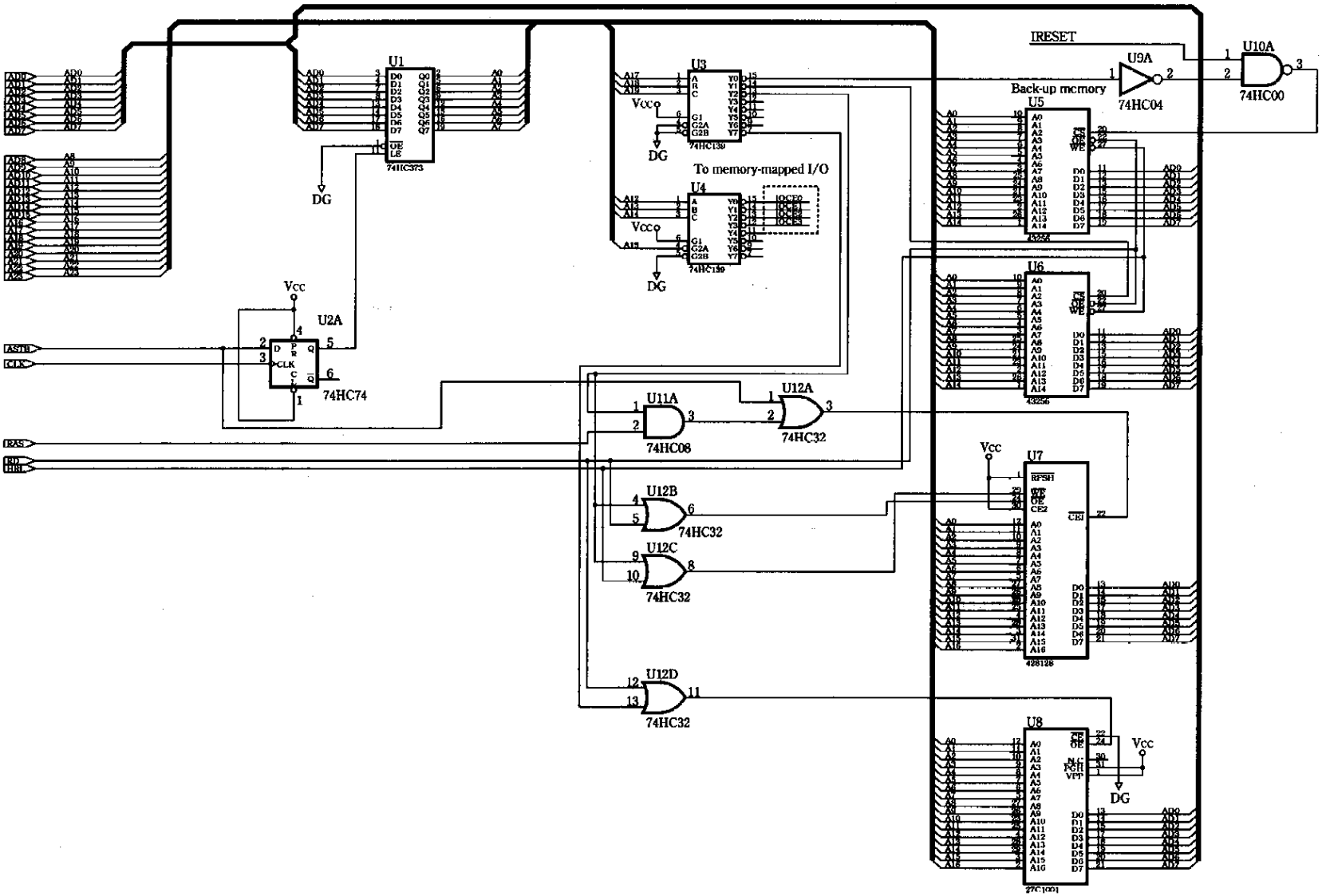


図2-2 メモリ・インタフェースの概略図★



## 2.1.3　ポート割り付けとレジスタ設定

### (1)　ポート割り付け

ポートにより，以下の信号を入力／出力する必要があります。

[入　　力]

・FINE，STOP，START　各キー入力
・音響カプラ検知（CPD）
・NCUからのフック（HOOK），リンガ（RING）入力信号
・原稿，感熱紙，カバー・オープン信号（GENCO，PAPER，COVER）
・サーマル・ヘッド・ストローブ・イネーブル信号（STBSENS＝ストローブ出力中を示します）
・ノンマスカブル割り込み要求入力信号（NMI）
・モデム割り込み入力信号（INTP1）
・UART受信データ入力信号（RxD1）
・UART送信許可信号（$\overline{\text{CTS1}}$）
・外部DMA要求信号（DMARQ0）

[出　　力]

・COVER open LED，PAPER empty LED，ERROR LED，FINE LED，START LED
・NMIイネーブル信号（NMIE）
・サーマル・ヘッド電源供給信号（THMP）
・サーマル・ヘッド・ラッチ信号（THLT）
・サーマル・ヘッド・ストローブON信号（STRS）
・密着センサ電源供給信号（CISP）
・中間調ON信号（DITH）
・モデム・リセット信号（MDMRES）
・モデム・ケーブル・イコライザ設定信号（CABL1，CABL2）
・読み取りモータ・イネーブル信号（RMOTE）
・印字モータ・イネーブル信号（PMOTE）
・モータ励磁信号：$\phi$0（MPH0）
　モータ励磁信号：$\phi$1（MPH1）
　モータ励磁信号：$\phi$2（MPH2）
　モータ励磁信号：$\phi$3（MPH3）
・PSTN／カプラ切り換え（PSTN/CPL）
・ライン・リレーON（LINE/PHN）

・ブザーON（BZE）
・DREQ-ON信号（DMAE）
・タイマ・ユニット信号（TO30）
・CSI送信データ信号（SO0）
・UART送信データ信号（TxD1）

［入 出 力］
・CSIシリアル・クロック信号（$\overline{SCK0}$）

上記の信号を次に示すP0X～P8X，PLAX（PLA内部ポート）に割り付けます。

表2-2　ポート割り付け表（1/2）

| ポート | 端子名称 | 入出力 | 機　　能 |
|---|---|---|---|
| P00 | — | | |
| P01 | NMIE | 出　力 | NMIイネーブル出力信号 |
| P02 | — | | |
| P03 | START LED | 出　力 | スタートLEDの点灯用出力信号 |
| P04 | FINE LED | 出　力 | ファインLEDの点灯用出力信号 |
| P05 | ERROR LED | 出　力 | エラーLEDの点灯用出力信号 |
| P06 | PAPER LED | 出　力 | ペーパLEDの点灯用出力信号 |
| P07 | COVER LED | 出　力 | カバー・オープンLEDの点灯用出力信号 |
| P10 | NMI | 入　力 | ノンマスカブル割り込み要求入力信号 |
| P11 | INTP0 | 入　力 | 未使用 |
| P12 | INTP1 | 入　力 | モデム割り込み入力信号 |
| P13 | — | | |
| P14 | START KEY | 入　力 | スタート・キー入力信号 |
| P15 | STOP KEY | | ストップ・キー入力信号 |
| P16 | FINE KEY | | ファイン・キー入力信号 |
| P20 | — | | |
| P21 | CPD | 入　力 | 音響カプラ検知入力信号 |
| ★ P22 | MDMRES | 出　力 | モデム・リセット出力信号 |
| P23 | CISP | 出　力 | 密着センサ電源供給出力信号 |
| ★ P24 | THMP | 出　力 | サーマル・ヘッド電源供給出力信号 |
| P25 | TO30 | 出　力 | タイマ・ユニット出力信号 |
| P30 | SO0 | 出　力 | CSI送信データ出力信号 |
| P31 | CABL2 | 出　力 | モデム・ケーブル・イコライザ設定出力信号 |
| P32 | $\overline{SCK0}$ | 入出力 | CSIシリアル・クロック入出力信号 |
| P33 | CABL1 | 出　力 | モデム・ケーブル・イコライザ設定出力信号 |

保守／廃止

表2-2 ポート割り付け表 (2/2)

| ポート | 端子名称 | 入出力 | 機能 |
|---|---|---|---|
| P34 | TxD1 | 出力 | UART送信データ出力信号 |
| P35 | RxD1 | 入力 | UART受信データ入力信号 |
| P36 | $\overline{CTS1}$ | 入力 | UART送信許可信号 |
| P40 | RING | 入力 | リンガ入力信号 |
| P41 | HOOK | 入力 | フック入力信号 |
| P42 | DITH | 出力 | 中間調ON出力信号 |
| P43 | — | | |
| P44 | PMOTE | 出力 | 印字モータ・イネーブル出力信号 |
| P45 | RMOTE | 出力 | 読み取りモータ・イネーブル出力信号 |
| P46 | THLT | 出力 | サーマル・ヘッド・ラッチ出力信号 |
| P47 | — | | |
| P50 | COVER | 入力 | カバー・オープン・センサ入力信号 |
| P51 | PAPER | | 感熱紙センサ入力信号 |
| P52 | GENCO | | 原稿センサ入力信号 |
| P60 | THTMP | 入力 | サーマル・ヘッド・ストローブ・イネーブル入力信号 ★ |
| P61 | — | | 未使用 |
| P62 | — | | |
| P63 | — | | |
| P70 | LINE/PHN | 出力 | ライン・リレーON出力信号 |
| P71 | — | | |
| P72 | PSTN/CPL | 出力 | PSTN／カプラ切り換え出力信号 |
| P73 | — | | |
| P74 | MPH3 | 出力 | モータ励磁信号（$\phi$3） |
| P75 | MPH2 | | モータ励磁信号（$\phi$2） |
| P76 | MPH1 | | モータ励磁信号（$\phi$1） |
| P77 | MPH0 | | モータ励磁信号（$\phi$0） |
| P80 | DMARQ0 | 入力 | 外部DMA要求信号（チャネル0） |
| P81 | — | | |

### (2)　レジスタ設定

各ポートには対応した制御レジスタが用意されています。

これらレジスタにポート各端子を汎用ポートとして使用するか，またはコントロール信号入出力として使用するかをビット単位で指定します。ポート・モードの場合は，入力／出力のどちらで使用するかもビット単位で指定します。

SFR（特殊機能レジスタ）の各レジスタ設定（ポート機能）を以下に示します。

#### (a)　ポート0（P00～P07）

8ビットの汎用入出力ポートです。ポート0モード・レジスタの指定により，ビット単位に入力／出力の切り替えができます。

●ポート0モード・レジスタ（PM0）

ポート0の入力／出力をビット単位で指定する8ビットのレジスタです。

全ビット出力に設定します。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM0 | PM0 7 | PM0 6 | PM0 5 | PM0 4 | PM0 3 | PM0 2 | PM0 1 | PM0 0 |
| 設定値 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| PM0n | 入出力指定 |
|---|---|
| 0 | 出　力 |
| 1 | 入　力 |

表2-3　ポート0の動作（n＝0-7）

| | ポート・モード | | 端子名称 |
|---|---|---|---|
| | PM0n＝1 | PM0n＝0 | |
| P00 | 入力ポート | 出力ポート | — |
| P01 | 入力ポート | 出力ポート | NMIE出力 |
| P02 | 入力ポート | 出力ポート | — |
| P03 | 入力ポート | 出力ポート | START LED出力 |
| P04 | 入力ポート | 出力ポート | FINE LED出力 |
| P05 | 入力ポート | 出力ポート | ERROR LED出力 |
| P06 | 入力ポート | 出力ポート | PAPER LED出力 |
| P07 | 入力ポート | 出力ポート | COVER LED出力 |

備考　　は，設定内容を示します。

(b)　ポート1（P10-P16）

7ビットの特殊入力ポートです。P10-P16は入力専用ポートとして機能するほか，外部割り込み入力，タイマ・ユニットへのキャプチャ・トリガ入力端子，またはA/Dコンバータ・トリガ入力端子として機能します。

P10はNMI，P11はINTP0，P12はINTP1として使用します。

表2-4　ポート1の動作（n＝0-6）

| | 機能 | | 端子名称 |
|---|---|---|---|
| P10 | NMI入力 | 兼　入力専用ポート | NMI入力 |
| P11 | INTP0入力 | 兼　入力専用ポート | INTP0入力 |
| P12 | INTP1入力 | 兼　入力専用ポート | INTP1入力 |
| P13 | INTP2入力 | 兼　入力専用ポート | — |
| P14 | INTP3入力／TI入力 | 兼　入力専用ポート | START KEY入力 |
| P15 | INTP4入力 | 兼　入力専用ポート | STOP KEY入力 |
| P16 | INTP5入力 | 兼　入力専用ポート | FINE KEY入力 |

備考　　は，設定内容を示します。

(c)　ポート 2 (P20-P25)

6 ビットの汎用入出力ポートです。ポート 2 モード・レジスタ (PM2) の設定により，ビット単位に入力／出力の指定ができます。

ポート 2 モード・コントロール・レジスタ (PMC2) の設定により，タイマ・ユニットの出力端子としても機能します。

●ポート 2 モード・コントロール・レジスタ (PMC2)

ポート 2 のポート／コントロール信号の指定を，ビット単位で行う 8 ビットのレジスタです。

P20～P24をポートとして使用します。P25はTO30の出力として使用します。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC2 | 0 | 0 | PMC25 | PMC24 | PMC23 | PMC22 | PMC21 | PMC20 |
| 設定値 | | | 1 | 0 | 0 | 0 | 0 | 0 |

| PMC2n | モード指定 |
|---|---|
| 0 | ポート・モード |
| 1 | コントロール・モード |

●ポート 2 モード・レジスタ (PM2)

ポート 2 の入力／出力をビット単位で指定する 8 ビットのレジスタです。

P20，P21は入力ポート，P22～P24は出力ポートとして使用します。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM2 | 1 | 1 | PM25 | PM24 | PM23 | PM22 | PM21 | PM20 |
| 設定値 | | | × | 0 | 0 | 0 | 1 | 1 |

| PM2n | 入出力指定 |
|---|---|
| 0 | 出　力 |
| 1 | 入　力 |

保守／廃止

表2-5　ポート2の動作（n＝0-5）

| | コントロール・モード | ポート・モード | | 端子名称 |
|---|---|---|---|---|
| | PMC2n=1 | PMC2n=0 | | |
| | | PM2n=1 | PM2n=0 | |
| P20 | PWM出力 | 入力ポート | 出力ポート | — |
| P21 | TO00出力 | 入力ポート | 出力ポート | CPD入力 |
| P22 | TO01出力 | 入力ポート | 出力ポート | MDMRES出力 ★ |
| P23 | TO20出力 | 入力ポート | 出力ポート | CISP出力 |
| P24 | TO21出力 | 入力ポート | 出力ポート | THMP出力 ★ |
| P25 | TO30出力 | 入力ポート | 出力ポート | TO30出力 |

備考1.　　は，設定内容を示します。

2.　×は0または1です。

(d) **ポート 3 (P30-P36)**

7 ビットの汎用入出力ポートです。ポート 3 モード・レジスタ (PM3) の設定により，ビット単位に入力／出力の指定ができます。

ポート 3 モード・コントロール・レジスタ (PMC3) の設定により，シリアル用入出力端子としても機能します。

P31，P33を出力ポートとして使用します。ほかは，コントロール・モードです。

●ポート 3 モード・コントロール・レジスタ (PMC3)

ポート 3 のポート／コントロール信号の指定を，ビット単位で行う 8 ビットのレジスタです。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC3 | 0 | PMC3 6 | PMC3 5 | PMC3 4 | PMC3 3 | PMC3 2 | PMC3 1 | PMC3 0 |
| 設定値 | | 1 | 1 | 1 | 0 | 1 | 0 | 1 |

| PMC3n | モード指定 |
|---|---|
| 0 | ポート・モード |
| 1 | コントロール・モード |

●ポート 3 モード・レジスタ (PM3)

ポート 3 の入力／出力をビット単位で指定する 8 ビットのレジスタです。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM3 | 1 | PM3 6 | PM3 5 | PM3 4 | PM3 3 | PM3 2 | PM3 1 | PM3 0 |
| 設定値 | | × | × | × | 0 | × | 0 | × |

| PM3n | 入出力指定 |
|---|---|
| 0 | 出　力 |
| 1 | 入　力 |

表2-6　ポート3の動作（n＝0-6）

| | コントロール・モード | ポート・モード | | 端子名称 |
|---|---|---|---|---|
| | PMC3n＝1 | PMC3n＝0 | | |
| | | PM3n＝1 | PM3n＝0 | |
| P30 | TxD0出力/SB0入出力/SO0出力 | 入力ポート | 出力ポート | SO0出力 |
| P31 | RxD0入力/SB1入出力/SI0入力 | 入力ポート | 出力ポート | CABL2出力 |
| P32 | TxC出力/SCK0入出力 | 入力ポート | 出力ポート | $\overline{\text{SCK0}}$入出力 |
| P33 | $\overline{\text{CTS0}}$入力 | 入力ポート | 出力ポート | CABL1出力 |
| P34 | TxD1出力/SO1出力 | 入力ポート | 出力ポート | TxD1出力 |
| P35 | RxD1入力/SI1入力 | 入力ポート | 出力ポート | RxD1入力 |
| P36 | $\overline{\text{CTS1}}$入力/SCK1入出力 | 入力ポート | 出力ポート | $\overline{\text{CST1}}$入力 |

備考1.　　は，設定内容を示します。

2.　×は0または1です。

(e)　ポート4（P40–P47）

8ビットの汎用入出力ポートです。ポート4モード・レジスタ（PM4）の設定により，ビット単位に入力／出力の指定ができます。

ポート4モード・コントロール・レジスタ（PMC4）の設定によりパラレル・インタフェース制御信号としても機能します。

●ポート4モード・コントロール・レジスタ（PMC4）

ポート4のポート／コントロール信号の指定を，バイト単位で行う8ビットのレジスタです。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC4 | $PMC4_7$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 設定値 | 0 | | | | | | | |

| $PMC4_7$ | モード指定 |
|---|---|
| 0 | ポート・モード |
| 1 | コントロール・モード |

●ポート4モード・レジスタ（PM4）

ポート4の入力／出力をビット単位で指定する8ビットのレジスタです。

P40～P42を入力ポート，P43～P47を出力ポートとして使用します。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM4 | PM4 7 | PM4 6 | PM4 5 | PM4 4 | PM4 3 | PM4 2 | PM4 1 | PM4 0 |
| 設定値 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

| PM4n | 入出力指定 |
|---|---|
| 0 | 出　力 |
| 1 | 入　力 |

保守／廃止

表2-7　ポート4の動作（n＝0-7）

| | コントロール・モード | ポート・モード | | 端子名称 |
|---|---|---|---|---|
| | $PMC4_7$＝1 | $PMC4_7$＝0 | | |
| | | PM4n＝1 | PM4n＝0 | |
| P40 | PD0入出力 | 入力ポート | 出力ポート | RING入力 |
| P41 | PD1入出力 | 入力ポート | 出力ポート | HOOK入力 |
| P42 | PD2入出力 | 入力ポート | 出力ポート | DITH出力 |
| P43 | PD3入出力 | 入力ポート | 出力ポート | — |
| P44 | PD4入出力 | 入力ポート | 出力ポート | PMOTE出力 |
| P45 | PD5入出力 | 入力ポート | 出力ポート | RMOTE出力 |
| P46 | PD6入出力 | 入力ポート | 出力ポート | THLT出力 |
| P47 | PD7入出力 | 入力ポート | 出力ポート | — |

**備考**　　は，設定内容を示します。

(f)　**ポート5（P50-P52）**

3ビットの汎用入出力ポートです。ポート5モード・レジスタ（PM5）の設定により，ビット単位に入力／出力の指定ができます。

ポート5モード・コントロール・レジスタ（PMC5）の設定によりパラレル・インタフェース制御信号としても機能します。

●ポート5モード・コントロール・レジスタ（PMC5）

ポート5のポート／コントロール信号の指定を，ビット単位で行う8ビットのレジスタです。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC5 | 0 | 0 | 0 | 0 | 0 | PMC52 | PMC51 | PMC50 |
| 設定値 | | | | | | 0 | 0 | 0 |

| PMC5n | モード指定 |
|---|---|
| 0 | ポート・モード |
| 1 | コントロール・モード |

●ポート5モード・レジスタ（PM5）

ポート5の入力／出力をビット単位で指定する8ビットのレジスタです。

P50～P52を入力ポートとして使用します。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM5 | 1 | 1 | 1 | 1 | 1 | PM52 | PM51 | PM50 |
| 設定値 | | | | | | 1 | 1 | 1 |

| PM5n | 入出力指定 |
|---|---|
| 0 | 出　力 |
| 1 | 入　力 |

表2－8　ポート5の動作（n＝0－2）

| | コントロール・モード | ポート・モード | | 端子名称 |
|---|---|---|---|---|
| | PMC5n＝1 | PMC5n＝0 | | |
| | | PM5n＝1 | PM5n＝0 | |
| P50 | $\overline{\text{DATASTB}}$入出力 | 入力ポート | 出力ポート | COVER入力 |
| P51 | $\overline{\text{ACK}}$入出力 | 入力ポート | 出力ポート | PAPER入力 |
| P52 | BUSY入出力 | 入力ポート | 出力ポート | GENCO入力 |

**備考**　　は，設定内容を示します。

★ (g) ポート6 (P60-P63)

4ビットの汎用入力ポートです。A/Dコンバータのアナログ入力としても機能します。

P60をアナログ入力として使用します。

表2-9　ポート6の動作

| | 機　　能 | 端子名称 |
|---|---|---|
| P60 | 入力専用ポート | THTMP入力 |
| P61 | 入力専用ポート | — |
| P62 | 入力専用ポート | — |
| P63 | 入力専用ポート | — |

備考 ▨ は，設定内容を示します。

(h)　**ポート 7 (P70-P77)**

8 ビットの汎用入出力ポートです。ポート 7 モード・レジスタ (PM7) の設定により，ビット単位に入力／出力の指定ができます。

ポート 7 モード・コントロール・レジスタ (PMC7) の設定により，リアルタイム出力ポートとしても機能します。

●ポート 7 モード・コントロール・レジスタ (PMC7)

ポート 7 のポート／コントロール信号の指定を，ビット単位で行う 8 ビットのレジスタです。

P70-P73をポートとして使用します。P74-P77はRTP出力として使用します。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC7 | PMC7 7 | PMC7 6 | PMC7 5 | PMC7 4 | PMC7 3 | PMC7 2 | PMC7 1 | PMC7 0 |
| 設定値 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |

| PMC7n | モード指定 |
|---|---|
| 0 | ポート・モード |
| 1 | コントロール・モード |

●ポート 7 モード・レジスタ (PM7)

ポート 7 の入力/出力をビット単位で指定する 8 ビットのレジスタです。

P73のみ入力ポート，ほかは出力ポートとして使用します。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM7 | PM7 7 | PM7 6 | PM7 5 | PM7 4 | PM7 3 | PM7 2 | PM7 1 | PM7 0 |
| 設定値 | × | × | × | × | 1 | 0 | 0 | 0 |

| PM7n | 入出力指定 |
|---|---|
| 0 | 出　力 |
| 1 | 入　力 |

**備考**　×は 0 または 1 です。

表2－10　ポート7の動作（n＝0－7）

| | コントロール・モード | ポート・モード | | 端子名称 |
|---|---|---|---|---|
| | PMC7n=1 | PMC7n=0 | | |
| | | PM7n=1 | PM7n=0 | |
| P70 | RTP0出力 | 入力ポート | 出力ポート | LINE/PHN出力 |
| P71 | RTP1出力 | 入力ポート | 出力ポート | — |
| P72 | RTP2出力 | 入力ポート | 出力ポート | PSTN/CPL出力 |
| P73 | RTP3出力 | 入力ポート | 出力ポート | — |
| P74 | RTP4出力 | 入力ポート | 出力ポート | MPH3出力 |
| P75 | RTP5出力 | 入力ポート | 出力ポート | MPH2出力 |
| P76 | RTP6出力 | 入力ポート | 出力ポート | MPH1出力 |
| P77 | RTP7出力 | 入力ポート | 出力ポート | MPH0出力 |

備考　　は，設定内容を示します。

(i)　ポート8（P80，P81）

2ビットの汎用入出力ポートです。ポート8モード・レジスタ（PM8）の設定により，ビット単位に入力／出力の指定ができます。

ポート8モード・コントロール・レジスタ（PMC8）の設定により，DMA制御端子としても機能します。

●ポート8モード・コントロール・レジスタ（PMC8）

ポート8のポート／コントロール信号の指定を，ビット単位で行う8ビットのレジスタです。

P80をDMA制御端子として使用します。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PMC8 | 0 | 0 | 0 | 0 | 0 | 0 | PMC81 | PMC80 |
| 設定値 | | | | | | | 0 | 1 |

| PMC8n | モード指定 |
|---|---|
| 0 | ポート・モード |
| 1 | コントロール・モード |

●ポート8モード・レジスタ（PM8）

ポート8の入力／出力をビット単位で指定する8ビットのレジスタです。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM8 | 1 | 1 | 1 | 1 | 1 | 1 | PM81 | PM80 |
| 設定値 | | | | | | | 1 | × |

| PM8n | 入出力指定 |
|---|---|
| 0 | 出　力 |
| 1 | 入　力 |

表2-11　ポート8の動作（n＝0，1）

| | コントロール・モード | ポート・モード | | |
|---|---|---|---|---|
| | PMC8n＝1 | PMC8n＝0 | | |
| | | PM8n＝1 | PM8n＝0 | |
| P80 | DMARQ0入力 | 入力ポート | 出力ポート | DMARQ0入力 |
| P81 | DMARQ1入力 | 入力ポート | 出力ポート | — |

備考1.　　は，設定内容を示します。

2.　×は0または1です。

## 2.1.4　初期設定

### (1)　初期設定の手順

V55PIのリセットは，**パワーオン・リセット**により実行されます。

V55PIのリセット後，次に示す手順でシステムの初期設定を行います。

**初期設定の手順**

① 開始

② STBCレジスタ（スタンバイ・コントロール・レジスタ）の設定

STBC←01H

③ PRCレジスタ（プロセッサ・コントロール・レジスタ）の設定

PRC←E4H

④ MBCレジスタ（メモリ・ブロック・コントロール・レジスタ）の設定

MBC←D6H

⑤ PWCレジスタ（プログラマブル・ウエイト・コントロール・レジスタ）の設定

PWC1←96H

PWC0←07H

⑥ RFMレジスタ（リフレッシュ・モード・レジスタ）の設定

RFM←30H

⑦ 各ポートの初期設定（「**2.1.3　ポート割り付けとレジスタ設定**」を参照）

⑧ 内部I/Oの初期設定（「**2.2　内部I/O**」を参照）

⑨ 割り込みマスク解除

⑩ 終了

保守／廃止

(2)　動作設定レジスタの初期設定

初期設定の手順の②～⑥に示した5つのレジスタは次のように設定します。

(a)　STBCレジスタ（スタンバイ・コントロール・レジスタ）

ストップ状態からの復帰判定用に使用します。SBFフラグは，専用命令によるセット(1)のみ可能です。

図2-3　STBCレジスタ

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | リセット時 |
|---|---|---|---|---|---|---|---|---|---|
| STBC | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SBF | (不定[注]) |
| 設定値 | | | | | | | | 1 | |

| SBF | 動　　作 |
|---|---|
| 0 | 電源投入時　$V_{DD}$=0 [V] |
| 1 | 命令による書き込み |

注　パワーオン・リセット時：00H

その他　　：変化せず

備考　▨ は，設定内容を示します。

保守／廃止

(b) PRCレジスタ（プロセッサ・コントロール・レジスタ）

CPUの動作クロック，ソフトウエア・インターバル・タイマの入力クロックの指定，CPUや内部システム制御に関連する項目を集中的に制御します。

図2-4　PRCレジスタ

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | リセット時 |
|---|---|---|---|---|---|---|---|---|---|
| PRC | 1 | 1 | ENCLK | 0 | TB1 | TB0 | PCK1 | PCK0 | (EEH) |
| 設定値 | | | 1 | | 0 | 1 | 0 | 0 | |

システム・クロックφの分周比指定

| PCK1 | PCK0 | 指　定 |
|---|---|---|
| 0 | 0 | φ＝発振周波数（$f_x$）×（1/2） |
| 0 | 1 | φ＝発振周波数（$f_x$）×（1/4） |
| 1 | 0 | φ＝発振周波数（$f_x$）×（1/8） |
| 1 | 1 | φ＝発振周波数（$f_x$）×（1/16） |

ソフトウエア・インターバル・タイマの入力クロックの指定

| TB1 | TB0 | 入力クロック周波数 | 最大インターバル時間 |
|---|---|---|---|
| 0 | 0 | $\phi/2^2$Hz | 21.0 ms |
| 0 | 1 | $\phi/2^5$Hz | 671 ms |
| 1 | × | 動作停止 | ∞ |

CLKOUT端子からのシステム・クロック出力制御指定

| ENCLK | 指　定 |
|---|---|
| 0 | 出力禁止（ロウ・レベルで停止） |
| 1 | 出力許可 |

（スタンバイ・モードにおけるCLKOUT端子出力レベル）

HALTモード：出力継続

STOPモード：出力停止（直前レベル保持）

備考1. ▨ は，設定内容を示します。

2. ×は0または1です。

保守／廃止

(c)　MBCレジスタ（メモリ・ブロック・コントロール・レジスタ）

基本メモリ空間を最大4ブロック（ブロック0-ブロック3）に，拡張メモリ空間を2ブロック（ブロック4，ブロック5）に分割指定します。

図2-5　MBCレジスタ

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | リセット時 |
|---|---|---|---|---|---|---|---|---|---|
| MBC | MB31 | MB30 | MB21 | MB20 | MB11 | MB10 | MB01 | MB00 | (FCH) |
| 設定値 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | |

ブロック0の上限アドレス

| MB01 | MB00 | 上限アドレス |
|---|---|---|
| 0 | 0 | 000000H (0K) |
| 0 | 1 | 01FFFFH (128K) |
| 1 | 0 | 03FFFFH (256K) |
| 1 | 1 | 07FFFFH (512K) |

ブロック1の上限アドレス

| MB11 | MB10 | 上限アドレス |
|---|---|---|
| 0 | 0 | 030000H (256K) |
| 0 | 1 | 07FFFFH (512K) |
| 1 | 0 | 0BFFFFH (768K) |
| 1 | 1 | 0FFFFFH (1M) |

ブロック2の上限アドレス

| MB21 | MB20 | 上限アドレス |
|---|---|---|
| 0 | 0 | 09FFFFH (640K) |
| 0 | 1 | 0BFFFFH (768K) |
| 1 | 0 | 0DFFFFH (896K) |
| 1 | 1 | 0FFFFFH (1M) |

ブロック4の上限アドレス

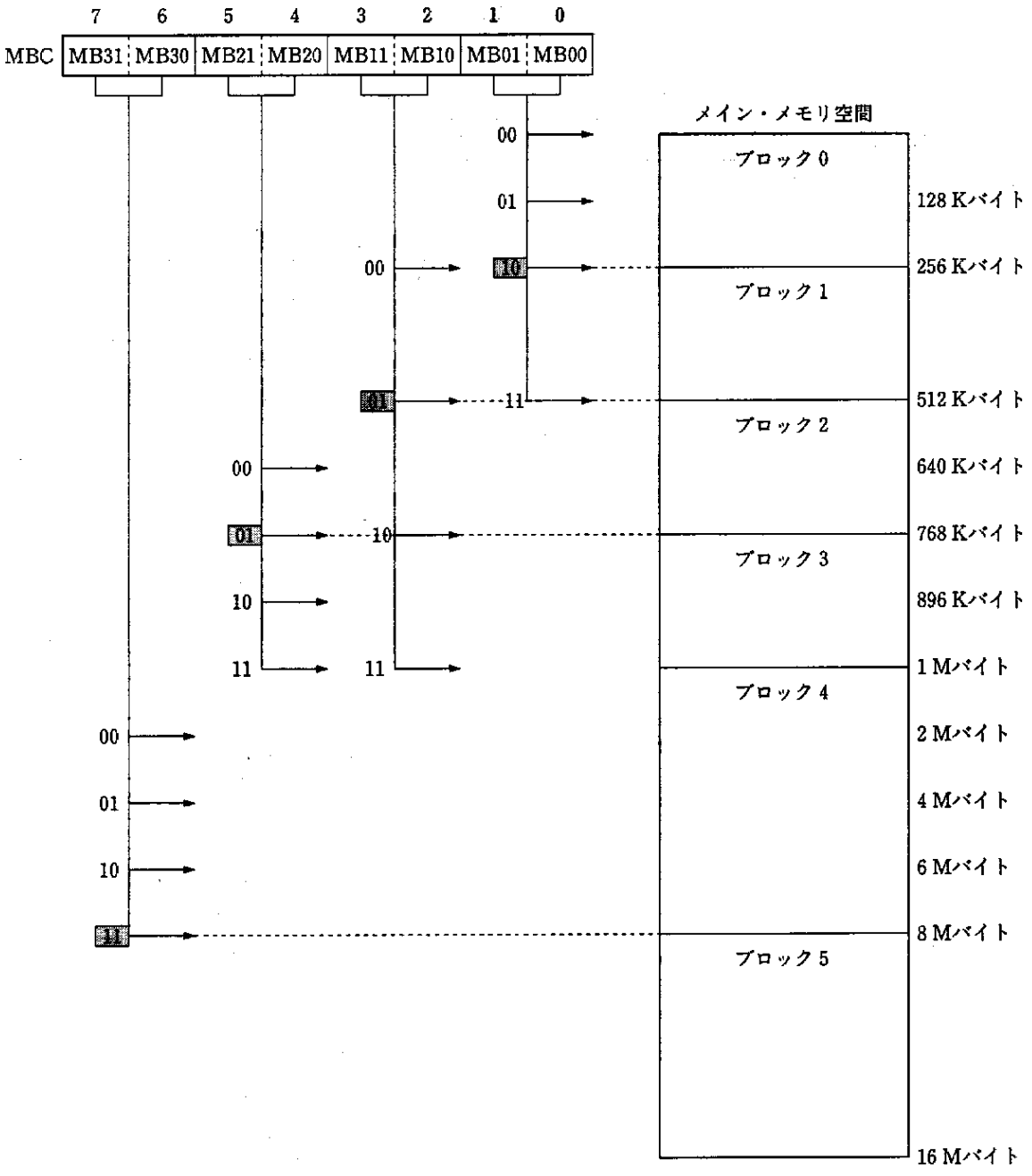
| MB31 | MB30 | 上限アドレス |
|---|---|---|
| 0 | 0 | 1FFFFFH (2M) |
| 0 | 1 | 3FFFFFH (4M) |
| 1 | 0 | 5FFFFFH (6M) |
| 1 | 1 | 7FFFFFH (8M) |

備考　▒▒ は，設定内容を示します。

保守／廃止

図2-6に，MBCレジスタの値をD6Hに設定した場合のメモリ・ブロック構成を示します。

図2-6　メモリの分割制御




備考　　は，設定内容を示します。

(d)　PWCレジスタ（プログラマブル・ウエイト・コントロール・レジスタ）

バス・サイクルに挿入するウエイト・ステート数を制御するレジスタです。

図2-7　PWCレジスタ

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | リセット時 |
|---|---|---|---|---|---|---|---|---|---|
| PWC1 | (BLOCK3) | | (BLOCK2) | | (BLOCK1) | | (BLOCK0) | | (EAH) |
| | DW31 | DW30 | DW21 | DW20 | DW11 | DW10 | DW01 | DW00 | |
| 設定値 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| PWC0 | BLOCK4 | BLOCK1 | (I/O空間) | | (BLOCK5) | | (BLOCK4) | | (AAH) |
| | AW1 | AW0 | IOW1 | IOW0 | DW51 | DW50 | DW41 | DW40 | |
| 設定値 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | |

データ・ウエイト（DW，IOW）

| DWn1/IOW1 | DWn0/IOW0 | ウエイト・ステート |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 2 |
| 1 | 1 | 3 |

アドレス・ウエイト（AW）

| AWn | | ウエイト・ステート |
|---|---|---|
| AW0 | 0 | 挿入しない（ブロック1） |
| | 1 | 挿入する（ブロック1） |
| AW1 | 0 | 挿入しない（ブロック4） |
| | 1 | 挿入する（ブロック4） |

備考　　は，設定内容を示します。

保守／廃止

(e) RFMレジスタ（リフレッシュ・モード・レジスタ）

リフレッシュ動作を制御する8ビットのレジスタです。

図2-8　RFMレジスタ

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | リセット時 |
|---|---|---|---|---|---|---|---|---|---|
| RFM | 0 | HLDRF | HLTRF | RFEN | 0 | RFT2 | RFT1 | RFT0 | (77H) |
| 設定値 | | 0 | 1 | 1 | | 0 | 0 | 0 | |

リフレッシュの周期の指定
（システム・クロック$\phi$＝12.5 MHz）

| RFT2 | RFT1 | RFT0 | リフレッシュの周期 |
|---|---|---|---|
| 0 | 0 | 0 | 64/$\phi$ (5.12 $\mu$s) |
| 0 | 0 | 1 | 128/$\phi$ (10.24 $\mu$s) |
| 0 | 1 | 0 | 192/$\phi$ (15.36 $\mu$s) |
| 0 | 1 | 1 | 256/$\phi$ (20.48 $\mu$s) |
| 1 | 0 | 0 | 8/$\phi$ (0.64 $\mu$s) |
| 1 | 0 | 1 | 16/$\phi$ (1.28 $\mu$s) |
| 1 | 1 | 0 | 24/$\phi$ (1.92 $\mu$s) |
| 1 | 1 | 1 | 32/$\phi$ (2.56 $\mu$s) |

リフレッシュ・サイクルの自動挿入の指定

| RFEN | リフレッシュ・サイクル自動挿入 |
|---|---|
| 0 | 自動挿入を禁止 |
| 1 | 自動挿入を許可 |

HALTモード中の
リフレッシュ・サイクルの自動挿入の指定

| HLTRF | リフレッシュ・サイクル自動挿入 |
|---|---|
| 0 | 自動挿入を禁止 |
| 1 | 自動挿入を許可[注1] |

ホールド・モード中の
リフレッシュ・サイクルの自動挿入の指定

| HLDRF | リフレッシュ・サイクル自動挿入 |
|---|---|
| 0 | 自動挿入を禁止 |
| 1 | 自動挿入を許可[注2] |

注1. RFENビット＝0の場合は，HLTRFビットの内容にかかわらず禁止となります。

2. 許可状態時には，リフレッシュ・タイミングごとに$\overline{\text{HLDAK}}$出力を強制的にハイ・レベルに立ち上げ，リフレッシュ・サイクルを自動挿入します。

備考　　は，設定内容を示します。

保守／廃止

## 2.2　内部I/O

### 2.2.1　DMAコントローラ

チャネル0をシングルトランスファ・モードで使用します。

(1)　DMAモード・レジスタ（DMAM0）

DMAの動作モード，DMA転送の禁止／許可などを指定する8ビットのレジスタです。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | リセット時 |
|---|---|---|---|---|---|---|---|---|---|
| DMAM0 (0FFF9CH) | MD2 | MD1 | MD0 | EDMAS | EDMA | TDMA | TS1 | TS0 | (E0H) |
| 設定値 | 1 | 0 | 0 | 0 | 0 | × | 0 | 1 | |

次ページ参照

DMA転送のソフトウエア起動ビット

| EDMA | TDMA | 転送スタート |
|---|---|---|
| 0 | × | DMA転送禁止状態 |
| 1 | 0 | 状態維持 |
| 1 | 1 | DMA転送スタート（ソフトウエア・トリガ） |

DMA転送許可／禁止制御ビット

| EDMA | DMA転送の許可／禁止 |
|---|---|
| 0 | DMA転送禁止状態 |
| 1 | DMA転送許可状態 |

DMA転送許可／禁止制御ビット（サブチャネル）

| EDMAS | DMA転送の許可／禁止 |
|---|---|
| 0 | DMA転送禁止状態 |
| 1 | DMA転送許可状態 |

動作モードの指定

| MD2 | MD1 | MD0 | 動作モード | |
|---|---|---|---|---|
| 0 | 0 | 0 | I/O→メモリ転送 | インテリジェントDMAモード-1(リング・バッファ方式) |
| 0 | 0 | 1 | I/O←メモリ転送 | インテリジェントDMAモード-2(カウンタ制御方式) |
| 0 | 1 | 0 | I/O↔メモリ転送 | ネクスト・アドレス指定モード |
| 0 | 1 | 1 | 設定禁止 | |
| 1 | 0 | 0 | I/O↔メモリ転送 | 2チャネル動作モード（終了時停止） |
| 1 | 0 | 1 | I/O↔メモリ転送 | 2チャネル動作モード（繰り返し） |
| 1 | 1 | 0 | メモリ↔メモリ転送 | メモリーメモリ転送モード(終了時停止) |
| 1 | 1 | 1 | メモリ↔メモリ転送 | メモリーメモリ転送モード(繰り返し) |

備考1.　　は，設定内容を示します。

2.　×は0または1です。

保守／廃止

TS1，TS0 のDMA起動要求，転送I/Oの指定

（チャネル0）

| 対応転送モード（転送対象） | TS1 | TS0 | メイン・チャネル | | サブチャネル | |
|---|---|---|---|---|---|---|
| | | | DMA起動トリガ | 転送I/O | DMA起動トリガ | 転送I/O |
| I/O－メモリ転送 | 0 | 0 | パラレル・インタフェース INTPAI | データ入出力バッファ (PAB) | シリアル送信完了 INTST0 | 送信バッファ (TxB0) |
| | | | 転送方向（I/O↔メモリ）注 | | 転送方向（メモリ→I/O） | |
| | 0 | 1 | DMARQ0 | 外部I/O ($\overline{DMAAK0}$) | DMARQ1 | 外部I/O ($\overline{DMAAK1}$) |
| | | | 転送方向（I/O→メモリ） | | 転送方向(メイン・チャネルと同方向) | |
| | 1 | 0 | 内蔵タイマ CM23一致信号 | 外部I/O ($\overline{DMAAK0}$) | DMARQ1 | 外部I/O ($\overline{DMAAK1}$) |
| | | | 転送方向（I/O↔メモリ） | | 転送方向(メイン・チャネルと逆方向) | |
| | 1 | 1 | シリアル送信完了 INTST0 | 送信バッファ (TxB0) | シリアル送信完了 INTST1 | 送信バッファ (TxB1) |
| | | | 転送方向（メモリ↔I/O） | | 転送方向（メモリ→I/O） | |
| メモリ－メモリ転送 | 0 | 0 | 設定禁止 | | | |
| | 0 | 1 | DMARQ0 | — | | |
| | 1 | 0 | CM23一致信号 | — | | |
| | 1 | 1 | 設定禁止 | | | |

注　メイン・チャネル側の転送方向をメモリ→I/O（DIR＝1）を選択した場合，サブチャネル側は使用できません。

**注意　メモリ－メモリ転送では，$\overline{DMAAKn}$および内蔵タイマ，シリアル制御回路に対するアクノリッジ信号は出力しません。**

**備考**　▒▒ は，設定内容を示します。

保守／廃止

### (2) DMAコントロール・レジスタ0（DMAC0）

DMAの転送モード，メモリのソース・アドレス，デスティネーション・アドレスの更新方法などを指定する8ビットのレジスタです。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | リセット時 |
|---|---|---|---|---|---|---|---|---|---|
| DMAC0 (0FFF9DH) | TMD1 | TMD0 | DIR | W | PD1 | PD0 | PS1 | PS0 | (00H) |
| 設定値 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | |

**アドレスの更新指定**

| PD1/PS1[注] | PD0/PS0[注] | アドレスの更新モード |
|---|---|---|
| 0 | 0 | 更新せず |
| 0 | 1 | インクリメント |
| 1 | 0 | デクリメント |
| 1 | 1 | 更新せず |

**転送処理方法の指定**

| W | 転送処理方法 |
|---|---|
| 0 | バイト転送 |
| 1 | ワード転送 |

**転送方向の指定**

| DIR | 転送方向 |
|---|---|
| 0 | I/O→メモリ転送 |
| 1 | メモリ→I/O転送 |

**転送モードの指定**

| TMD1 | TMD0 | 転送モード |
|---|---|---|
| 0 | 0 | シングルトランスファ・モード |
| 0 | 1 | ディマンド・リリース・モード |
| 1 | 0 | バースト・モード |
| 1 | 1 | シングルステップ・モード |

注　各動作モードのPD1，PD0，PS1，PS0ビットの意味は次に示すとおりです。
また，ネクスト・アドレス指定モードのとき，PS1＝0，PS0＝0を設定してください。

| 動作モード | PD1，PD0 | PS1，PS0 |
|---|---|---|
| インテリジェントDMAモード-1 | メモリ・アドレスの更新指定 | DPTCnの更新指定 |
| インテリジェントDMAモード-2 | メモリ・アドレスの更新指定 | DPTCnの更新指定 |
| ネクスト・アドレス指定モード | メモリ・アドレスの更新指定 | “00”を設定する |
| 2チャネル動作モード | メイン・チャネル側のアドレス更新指定 | サブチャネル側のアドレス更新指定 |
| メモリーメモリ転送モード | デスティネーション・アドレス更新指定 | ソース・アドレスの更新指定 |

備考　　は，設定内容を示します。

(3) ターミナル・カウント・モジュロ・レジスタ0（TCM0）

DMA転送回数の情報を$TC_n$にリロードする21ビットのモジュロ・レジスタです。

TCM0 ← FFH（回転回数−1）

転送バイト数：256バイト

(4) ターミナル・カウント0（TC0）

TC0 ← TCM0（TC0へ情報をリロード）

保守／廃止

## 2.2.2　タ イ マ　★

タイマ0は，RTP用ワンショット・タイマとして使用します。

タイマ1は，1 msタイマとして使用します。

タイマ2は，10 msタイマとして使用します。

タイマ3は，0.8 msタイマとして使用します。

### (1)　タイマ0の場合

#### (a)　タイマ・コントロール・レジスタ0（TMC0）

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TMC0 (0FFF30H) | CE | CLR | OVF | PRM | CE | 0 | OVF | 0 |
| 設定値 | × | 0 | 0 | 0 | × | 0 | 0 | 0 |

TM0

| PRM | TMnのカウント・クロックのソース選択 |
|---|---|

| タイマ | PRM | カウント・クロック |
|---|---|---|
| TM0 | 0 | $\phi$/8（固定） |

（$\phi$：システム・クロック）

| CE | タイマnの動作制御フラグ注 |
|---|---|

| CE | 動　作 |
|---|---|
| 0 | TMnはクリアされ，停止状態 |
| 1 | TMnはカウント動作を開始 |

注　CE=0のときに命令によるタイマ・カウンタへの書き込みを許可します。
CE=1のときには値は書き込まれません。

| OVF | タイマのオーバフロー・ステータス・フラグ |
|---|---|

| タイマ | OVF | 状　態 |
|---|---|---|
| TM0 | 0 | TM0からキャリー信号が発生していない |
| | 1 | TM0からキャリー信号が発生している |

#### (b)　タイマ・コンペア・レジスタ（CM00，CM01）

タイマ・レジスタ0（TM0）と常に比較を行い，一致検出すると同時に割り込み要求（INTCM00，INTCM01）が発生します。

CM00　　=1149H
(0FFF4CH)

CM01　　=1149/2H
(0FFF4EH)

**備考1.** ▒▒ は，設定内容を示します。

2.　×は0または1です。

### (2) タイマ1の場合

#### (a) タイマ・コントロール・レジスタ0（TMC0）

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TMC0 (0FFF30H) | CE | CLR | OVF | PRM | CE | 0 | OVF | 0 |
| 設定値 | × | 0 | 0 | 0 | × | 0 | 0 | 0 |

TM1

**CE　タイマnの動作制御フラグ注**

| CE | 動　作 |
|---|---|
| 0 | TMnはクリアされ，停止状態 |
| 1 | TMnはカウント動作を開始 |

注　CE＝0のときに命令によるタイマ・カウンタへの書き込みを許可します。
CE＝1のときには値は書き込まれません。

**CLR　タイマ1（TM1）の動作モードの指定**

| CLR | モ　ー　ド | 動　作 |
|---|---|---|
| 0 | インターバル・タイマ | CM10レジスタの一致信号でカウンタをクリア |
| 1 | フリーランニング・タイマ | CM10レジスタの一致信号にかかわらず，カウント動作を継続 |

**OVF　タイマのオーバフロー・ステータス・フラグ**

| タイマ | OVF | 状　態 |
|---|---|---|
| TM1 | 0 | TM1からキャリー信号が発生していない |
| | 1 | TM1からキャリー信号が発生している |

**PRM　TMnのカウント・クロックのソース選択**

| タイマ | PRM | カウント・クロック |
|---|---|---|
| TM1 | 0 | φ/8（固定） |
| | 1 | 外部クロック（TI） |

（φ：システム・クロック）

#### (b) タイマ・コンペア・レジスタ（CM10，CM11）

タイマ・レジスタ1（TM1）と常に比較を行い，一致検出すると同時に割り込み要求（INTCM10，INTCM11）が発生します。

CM10　　=05DCH
(0FFF50H)

CM11　　=05DC/2H
(0FFF52H)

**備考1.** ▒▒ は，設定内容を示します。

2. ×は0または1です。

(3)　タイマ2の場合

(a)　タイマ・コントロール・レジスタ1（TMC1）

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TMC1 (0FFF31H) | CE | CCLR | CSL | PRM | CE | 0 | 0 | PRM |
| 設定値 | × | 0 | 0 | 0 | × | 0 | 0 | 0 |

TM2

| CE | タイマnの動作制御フラグ注 |
|---|---|
| CE | 動　作 |
| 0 | TMnはクリアされ，停止状態 |
| 1 | TMnはカウント動作を開始 |

注　CE＝0のときに命令によるタイマ・カウンタへの書き込みを許可します。
CE＝1のときには値は書き込まれません。

| PRM | TMnのカウント・クロックのソース選択 | |
|---|---|---|
| タイマ | PRM | カウント・クロック |
| TM2 | 0 | $\phi$/8 |
| | 1 | $\phi$/32 |

（$\phi$：システム・クロック）

(b)　タイマ・コンペア・レジスタ（CM21）

タイマ・レジスタ2（TM2）と常に比較を行い，一致信号は，タイマ出力機能（TO20，TO21）に使用されます。特にCM21は一致により割り込み要求（INTCM21）が発生します。

CM21　　＝3A98H
(0FFF5AH)

備考1.　　は，設定内容を示します。

2.　×は0または1です。

保守／廃止

### (4)　タイマ3の場合

#### (a)　タイマ・コントロール・レジスタ1（TMC1）

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TMC1 (0FFF31H) | CE | CCLR | CSL | PRM | CE | 0 | 0 | PRM |
| 設定値 | × | 0 | 0 | 0 | × | 0 | 0 | 0 |

TM3

**CE　タイマnの動作制御フラグ注**

| CE | 動　作 |
|---|---|
| 0 | TMnはクリアされ，停止状態 |
| 1 | TMnはカウント動作を開始 |

注　CE＝0のときに命令によるタイマ・カウンタへの書き込みを許可します。
CE＝1のときには値は書き込まれません。

**CCLR，CSL　TO30端子出力動作の指定**

| CCLR | CSL | タイマ3のセット／リセット出力制御 |
|---|---|---|
| 0 | 0 | CCD20，CCD21出力の制御を受けずセット／リセット信号 |
| 0 | 1 | 設定禁止 |
| 1 | 0 | TO30端子のセット／リセット出力制御信号としてCCD20を選択 |
| 1 | 1 | TO30端子のセット／リセット出力制御信号としてCCD21を選択 |

**PRM　TMnのカウント・クロックのソース選択**

| タイマ | PRM | カウント・クロック |
|---|---|---|
| TM3 | 0 | $\phi$/8 |
| | 1 | $\phi$/32 |

（$\phi$：システム・クロック）

#### (b)　タイマ・コンペア・レジスタ（CM30，CM31）

タイマ・レジスタ3（TM3）と常に比較を行い，一致信号は，タイマ出力機能（TO30）に使用されます。また，CM31は一致により割り込み要求（INTCM31）が発生します。

CM30　　=1200H
(0FFF64H)

CM31　　=1050H
(0FFF66H)

**備考1.** ▒▒ は，設定内容を示します。

2. ×は0または1です。

### (5) タイマ出力制御レジスタ（TOC0，TOC1）

外部出力端子TO00，TO01，TO20，TO21，TO30の出力レベルを制御するレジスタです。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TOC0 (0FFF32H) | SE | RE | SE | RE | SE | RE | SE | RE |
| 設定値 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | TO30 | | TO20 | | TO01 | | TO00 | |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TOC1 (0FFF33H) | 0 | 0 | 0 | 0 | 0 | MOD | SE | RE |
| 設定値 | | | | | | 0 | 1 | 1 |
| | | | | | | | TO21 | |

**TO21出力信号の出力モード選択フラグ**

| MOD | モード | 動　作 |
|---|---|---|
| 0 | トグル出力モード | CM23レジスタの一致信号でTO21出力レベルを反転 |
| 1 | セット／リセット出力モード | CM22レジスタの一致信号でTO21の出力レベルをセット（1）<br>CM21レジスタの一致信号でTO21の出力レベルをリセット（0） |

**TO00，TO01，TO20，TO21，TO30端子制御フラグ**

| SE | RE | 出力状態 |
|---|---|---|
| 0 | 0 | 保持，停止 |
| 0 | 1 | 0出力，停止 |
| 1 | 0 | 1出力，停止 |
| 1 | 1 | スタート |

**備考** ▒▒ は，設定内容を示します。

保守／廃止

(6) リアルタイム出力ポート機能

ステッピング・モータ制御用として使用します。

(a) リアルタイム出力ポート・コントロール・レジスタ（RTPC）

リアルタイム出力ポートの動作モードを指定する8ビットのレジスタです。1/8ビットのメモリ・アクセスでリード／ライトが可能です。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RTPC (0FFF2CH) | TRG | BYTE | DLY | 0 | 0 | 0 | 0 | 0 |
| 設定値 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

データ出力時のディレイの指定

| DLY | モード設定 |
|---|---|
| 0 | ディレイなしモード |
| 1 | ディレイありモード |

転送タイミングの指定

| BYTE | | 転送タイミング |
|---|---|---|
| 0 | P7L | INTCM00のタイミング |
| | P7H | INTCM01のタイミング |
| 1 | P7L<br>P7H | INTCM01のタイミング |

転送タイミングのトリガ指定

| TRG | トリガ |
|---|---|
| 0 | タイマ0の割り込み要求 |
| 1 | このビットへの書き込み |

備考 ▒▒ は，設定内容を示します。

出力データがポート7バッファ(P7H，P7L)に書き込まれると，タイマ0からの割り込み要求（INTCM00）によって，ポート7バッファの内容はリアルタイム出力ポート（RTP）へ転送され，端子（P74-P77）へ出力されます。

リアルタイム出力ポートに出力するデータ転送は，マクロ・サービス処理のRTOPTRN(REAL TIME OUTPUT PORT TRANSFER)動作モードで行います。以下に，RTOPTRNの動作を説明します。

また，マクロ・サービス機能の詳細については，**V55PI ユーザーズ・マニュアル(暫定) ハードウエア編**を参照してください。

保守／廃止

［動　　作］

チャネル・ポインタ（CH.PTR）でマクロ・サービス・チャネルの先頭アドレスを指定します。

1回のマクロ・サービス転送要求に対して次の動作が行われます。

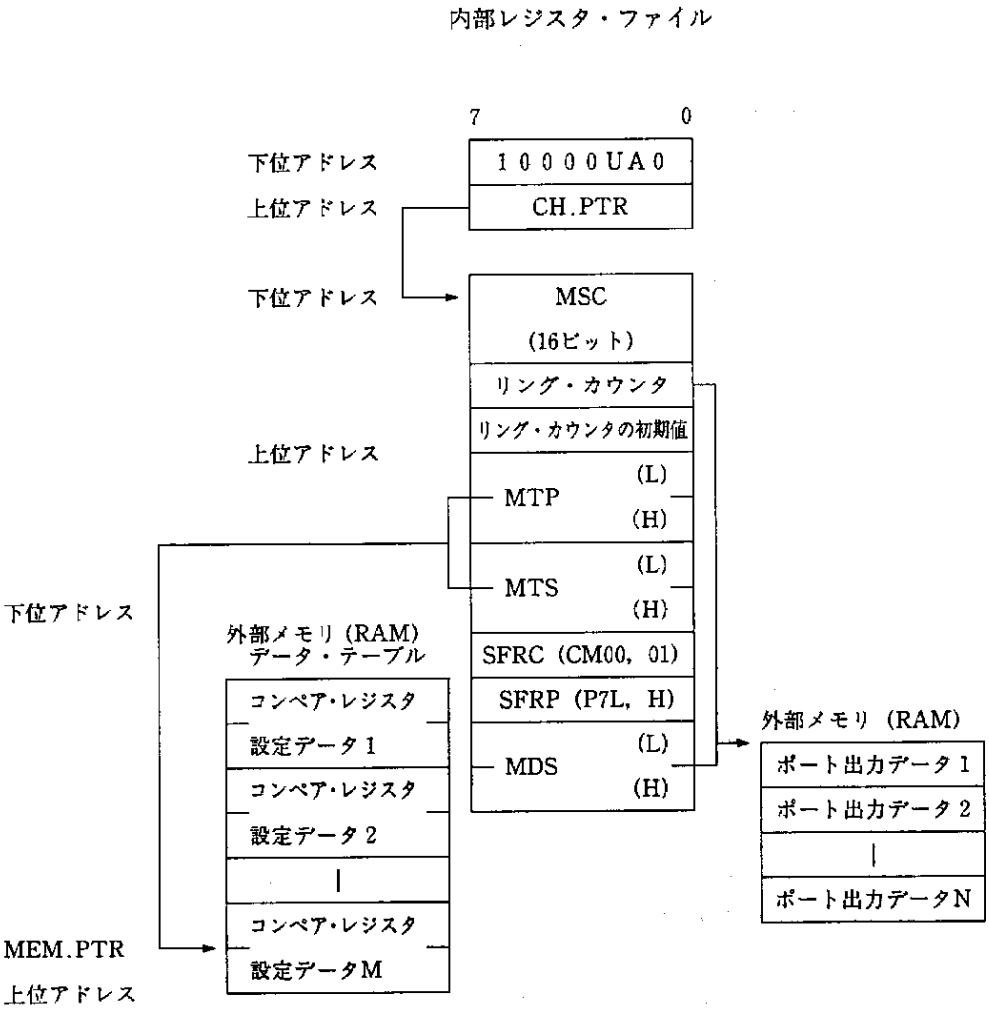
① SFRCで指定されるタイマ・コンペア・レジスタ（CM00，CM01）へMTS，MTPでアドレスされるデータ・テーブルからワード・データを転送します。

② リアルタイム出力ポートへ出力データ（バイト・データ）を転送します。MEM.PTR〔メモリ・セグメント（MDS）とメモリ・ポインタ（リング・カウンタの初期値－リング・カウンタ）〕でアドレスされる外部メモリからSFRPで指定されるリアルタイム出力ポートへバイト単位でデータを転送し，リング・カウンタの値をデクリメント（－1）します（リアルタイム出力ポート以外に，汎用ポートにデータ転送することも可能です）。

③ リング・カウンタ（RC）の値が0になったとき，RCの初期値をRCに自動的に再設定します。

④ マクロ・サービス・カウンタ（MSC）＝0になったとき，対応するMS/$\overline{\text{INT}}$を0にし，ベクタ割り込みまたはレジスタ・バンク切り替えを発生します。

［マクロ・サービス・コントロール・ワード］




**備考**　外部メモリは1Mバイトのメモリ空間です。

保守／廃止

## 2.2.3　シリアル・インタフェース

チャネル0は，サーマル・ヘッドへのデータ転送用として，クロック同期式モードで使用します。

チャネル1は，PCへのメッセージ転送用として，UARTモードで使用します。

### (1)　プロトコル選択レジスタ（ASP）

シリアル・インタフェース・ユニット各チャネルの動作プロトコルを指定するレジスタです。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | リセット時 |
|---|---|---|---|---|---|---|---|---|---|
| ASP (0FFF7FH) | 0 | 0 | 0 | 0 | 0 | 0 | P (ch1) | P (ch0) | (00H) |
| 設定値 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |

**動作プロトコルの指定**

| P | 動作プロトコル |
|---|---|
| 1 | 調歩同期式（UART） |
| 0 | クロック同期式（CSI） |

**○プロトコル選択と制御レジスタ**

| レジスタ・アドレス | 選択したプロトコル | |
|---|---|---|
| | 調歩同期式P=1 | クロック同期式P=0 |
| 0FFF73H | UARTM0（動作モード） | CSIM0（動作モード） |
| 0FFF7BH | UARTM1（動作モード） | CSIM1（動作モード） |
| 0FFF74H | UARTS0（ステータス） | SBIC0（制御） |
| 0FFF7CH | UARTS1（ステータス） | 未使用[注1] |
| 0FFF75H | TxB0（送信バッファ） | SIO0（シフト・レジスタ）[注2] |
| 0FFF7DH | TxB1（送信バッファ） | SIO1（シフト・レジスタ）[注2] |
| 0FFF76H | RxB0（受信バッファ） | |
| 0FFF7EH | RxB1（受信バッファ） | |
| 0FFF72H | PRS0（プリスケーラ） | |
| 0FFF7AH | PRS1（プリスケーラ） | |
| 0FFF70H | TxBRG0（送信BRG） | |
| 0FFF78H | TxBRG1（送信BRG） | |
| 0FFF71H | RxBRG0（受信BRG） | 未使用[注3] |
| 0FFF79H | RxBRG1（受信BRG） | |

注1．ch1の3線式シリアルI/Oモード（IOEモード）では，SBICレジスタは使用しません。

2．クロック同期式モードでは，送信シフト・レジスタ(SIO)に直接送信データを書き込みます。

3．クロック同期式モードでは，送受信ともデータの転送速度は同一です。送信側のボー・レート・ジェネレータ（BRG）のみを使用します。

**備考**　　は，設定内容を示します。

○プロトコル選択と制御端子

| 制御端子 | 選択したプロトコル | |
|---|---|---|
| | 調歩同期式P=1 | クロック同期式P=0 |
| 送信データ出力端子 | TxD0 | SO0/SB0 (ch0) |
| | TxD1 | SO1 (ch1) |
| 受信データ出力端子 | RxD0 | SI0/SB1 (ch0) |
| | RxD1 | SI1 (ch1) |
| クロック端子 | TxC | $\overline{SCK0}$ (ch0) |
| 送信制御端子 | CTS0 | 未使用 (ch0) |
| クロック／制御端子 | CTS1 | $\overline{SCK1}$ (ch1) |

備考　　は，設定内容を示します。

(2)　プリスケーラ・レジスタ0（PRS0）

ボー・レート・ジェネレータへの入力クロックを指定する8ビットのレジスタです。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | リセット時 |
|---|---|---|---|---|---|---|---|---|---|
| PRS0 (0FFF7FH) | 0 | 0 | TSK2 | TSK1 | TSK0 | RSK2 | RSK1 | RSK0 | (00H) |
| 設定値 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

受信シリアル・クロック発生回路への入力クロックの指定

| RSK2 | RSK1 | RSK0 | 入力クロック |
|---|---|---|---|
| 0 | 0 | 0 | 内部クロック　$\phi/2$ |
| 0 | 0 | 1 | 内部クロック　$\phi/2^2$ |
| 0 | 1 | 0 | 内部クロック　$\phi/2^3$ |
| 0 | 1 | 1 | 内部クロック　$\phi/2^4$ |
| 1 | 0 | 0 | 内部クロック　$\phi/2^5$ |
| 1 | 0 | 1 | 内部クロック　$\phi/2^6$ |
| 1 | 1 | 0 | 内部クロック　$\phi/2^7$ |
| 1 | 1 | 1 | 内部クロック　$\phi/2^8$ |

送信シリアル・クロック発生回路への入力クロックの指定

| TSK2 | TSK1 | TSK0 | 入力クロック |
|---|---|---|---|
| 0 | 0 | 0 | 内部クロック　$\phi/2$ |
| 0 | 0 | 1 | 内部クロック　$\phi/2^2$ |
| 0 | 1 | 0 | 内部クロック　$\phi/2^3$ |
| 0 | 1 | 1 | 内部クロック　$\phi/2^4$ |
| 1 | 0 | 0 | 内部クロック　$\phi/2^5$ |
| 1 | 0 | 1 | 内部クロック　$\phi/2^6$ |
| 1 | 1 | 0 | 内部クロック　$\phi/2^7$ |
| 1 | 1 | 1 | 内部クロック　$\phi/2^8$ |

備考　　は，設定内容を示します。

保守／廃止

### (3)　クロックト・シリアル・インタフェース・モード・レジスタ0（CSIM0）

クロックト・シリアル・インタフェースの基本動作モードを指定する8ビットのレジスタです。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | リセット時 |
|---|---|---|---|---|---|---|---|---|---|
| CSIM0 (0FFF73H) | CTxE | CRxE | WUP | MOD2 | MOD1 | MOD0 | 0 | CLS0 | (00H) |
| 設定値 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | |

**動作クロックの指定（ch0，ch1）**

| CLS0 | クロックの指定 | $\overline{SCK}$端子 |
|---|---|---|
| 0 | 外部クロック：スレーブ | 入　力 |
| 1 | 内部クロック：マスタ[注1] | 出　力 |

**各種動作モードの指定（ch1）**

| MOD0 | 動作モード | 転送方向 |
|---|---|---|
| 0 | IOEモード | MSBファースト |
| 1 | IOEモード | LSBファースト |

**各種動作モードの指定（ch0）**

| MOD1 | MOD0 | 動作モード | 転送方向 |
|---|---|---|---|
| 0 | 0 | IOEモード | MSBファースト |
| 0 | 1 | IOEモード | LSBファースト |
| 1 | 0 | SBIモード | MSBファースト |
| 1 | 1 | 設定禁止 | |

**使用端子の指定（ch0）**

| MOD2 | 使用端子と動作 |
|---|---|
| 0 | SO0/SB0端子で送受信動作 |
| 1 | SI0/SB1端子で送受信動作 |

**ウエイク・アップ動作の指定[注2]（ch0）**

| WUP | 動　　作 |
|---|---|
| 0 | 転送ごとに割り込み発生 |
| 1 | アドレス受信時割り込み発生 |

**受信許可／禁止の指定（ch0，ch1）**

| CRxE | 動　　作 |
|---|---|
| 0 | 受信動作を禁止 |
| 1 | 受信動作を許可 |

**送信許可／禁止の指定（ch0，ch1）**

| CTxE | 動　　作 |
|---|---|
| 0 | 送信動作を禁止 |
| 1 | 送信動作を許可 |

注1．BRG出力

2．SBIモードで有効

**備考**　　は，設定内容を示します。

保守／廃止

### (4) UARTモード・レジスタ1（UARTM1）

UARTの基本動作モードを指定する8ビットのレジスタです。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | リセット時 |
|---|---|---|---|---|---|---|---|---|---|
| UARTM1 (0FFF7BH) | TxE | RxE | PRTY1 | PRTY0 | CL/WUPT | SL | ASIE | WUP | (00H) |
| 設定値 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | |

**ウエイク・アップ動作許可指定**

| WUP | 指　定 |
|---|---|
| 0 | 使用しない |
| 1 | 使用する |

**AS割り込み機能[注]の動作許可指定**

| ASIE | 指　定 |
|---|---|
| 0 | 割り込み禁止 |
| 1 | 割り込み許可 |

**ストップ・ビット長の指定**

| SL | ストップ・ビット長 |
|---|---|
| 0 | 1 |
| 1 | 2 |

**キャラクタ長の指定**

| WUP | CL/WUPT | キャラクタ長 | ウェーク・アップ・ビット |
|---|---|---|---|
| 0 | 0 | 7ビット | 使用しない |
| 0 | 1 | 8ビット | 使用しない |
| 1 | 0 | 8ビット | 0送出 |
| 1 | 1 | 8ビット | 1送出 |

**パリティの不可指定**

| PRTY1 | PRTY0 | 動　作 |
|---|---|---|
| 0 | 0 | パリティなし |
| 0 | 1 | 0パリティ |
| 1 | 0 | 奇数パリティ |
| 1 | 1 | 偶数パリティ |

**受信許可の指定**

| RxE | 動　作 |
|---|---|
| 0 | 受信禁止状態 |
| 1 | 受信許可状態 |

**送信許可の指定**

| TxE | $\overline{CTS0}$ | 動　作 |
|---|---|---|
| 0 | × | 送信禁止状態 |
| 1 | 0 | 送信許可状態 |
| 1 | 1 | 送信禁止状態 |

注　AS（All Sent）：オール・セント状態検出割り込み

**注意　UARTが送信中のとき，ASIEフラグを除くUARTMレジスタのビット・データの変更はできません。**

**備考1.** ▨ は，設定内容を示します。

2. ×は0または1です。

保守／廃止

## 2.2.4　A/Dコンバータ ★

サーマル・ヘッドの温度検出用として使用します。

### (1)　A/Dコンバータ・モード・レジスタ（ADM）

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADM (0FFE20H) | CS | TRG | 0 | FR | 0 | AN1 | AN0 | MS |
| 設定値 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**変換動作モードの指定**

| MS | 変換動作モード |
|---|---|
| 0 | スキャン・モード |
| 1 | セレクト・モード |

**アナログ入力の指定**

| AN1 | AN0 | アナログ入力 | |
|---|---|---|---|
| | | スキャン・モード時 | セレクト・モード時 |
| 0 | 0 | ANI0入力をスキャン | ANI0入力をセレクト |
| 0 | 1 | ANI0, ANI1入力をスキャン | ANI1入力をセレクト |
| 1 | 0 | ANI0, ANI1, ANI2入力をスキャン | ANI2入力をセレクト |
| 1 | 1 | ANI0, ANI1, ANI2, ANI3入力をスキャン | ANI3入力をセレクト |

**変換動作時間の切り替え**

| FR | 変換動作時間 |
|---|---|
| 0 | 160ステート（$\phi$>8 MHzのとき） |
| 1 | 120ステート（$\phi \leq$8 MHzのとき） |

（$\phi$：システム・クロック）

**外部端子トリガの制御**

| TRG | トリガ制御 |
|---|---|
| 0 | 外部端子トリガ禁止 |
| 1 | 外部端子トリガ許可 |

**A/D変換動作制御**

| CS | 変換動作 |
|---|---|
| 0 | A/D変換動作停止 |
| 1 | A/D変換動作開始 |

**備考**　　は，設定内容を示します。

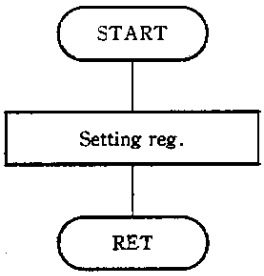
## 2.3　外部I/O

### 2.3.1　リアルタイム・クロック（μPD4991A）

リアルタイム・クロック（RTC）は，NEC製μPD4991Aを使用します。

メモリ・マップドI/Oであるため，RTC先頭アドレスからメモリ上に構造体を配置し，日時データをワーク・メモリ上に読み出します（この際，同時にHEXデータをASCIIコードへ置き換えます）。

また，設定日時も同様にして，RTCへ書き込みます。

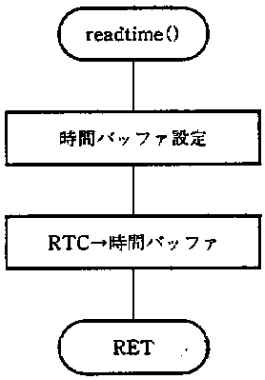



Setting RTC to run.

Setting mode reg.to basic mode.
Setting CNT1 reg.to run.

Initialization for RTC.

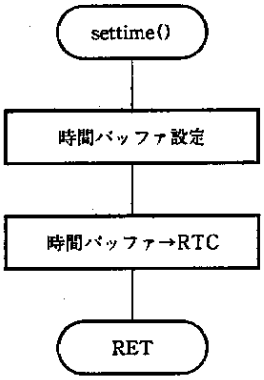
---




日時データ転送，および数字ASCIIコード変換

Getting data from RTC.

---

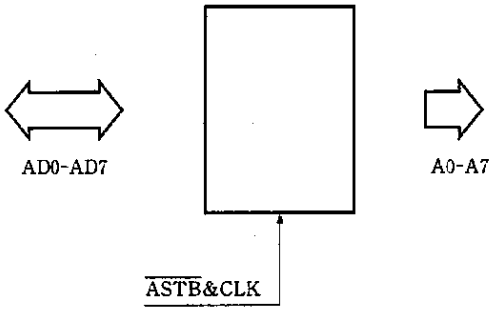



Setting time of RTC.

## 2.3.2　その他外部I/O

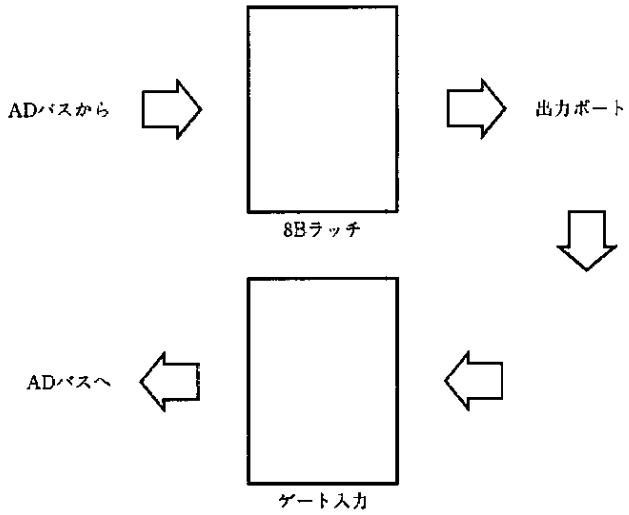
### (1)　アドレス・バス分離機能

V55PIは，アドレス・バス，データ・バスが共通であるため，$\overline{\text{ASTB}}$信号にてアドレス・バスをラッチし，周辺ICへ供給します。



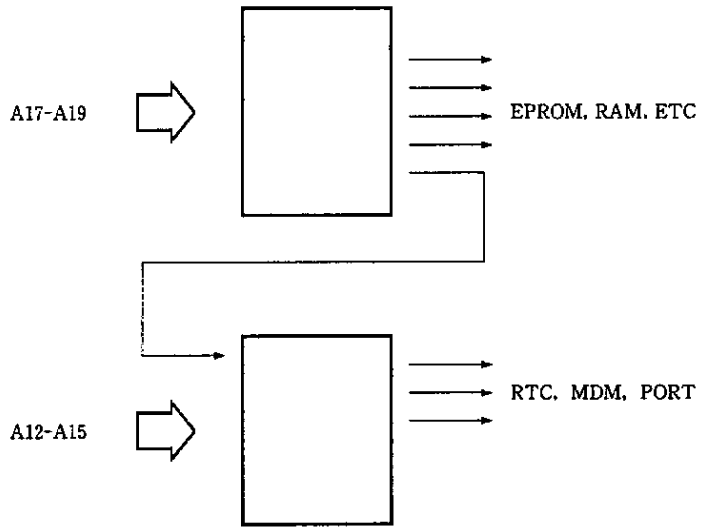

### (2)　出力ポート機能

8ビット－1チャネルの出力ポートです。FPGA内部で作り出すブザー用発振回路，DREQ発生回路，サーマル・ヘッド用ストローブ信号発生回路の制御に3ビット用い，ほかは汎用出力ポートとして，外部へ出力します。



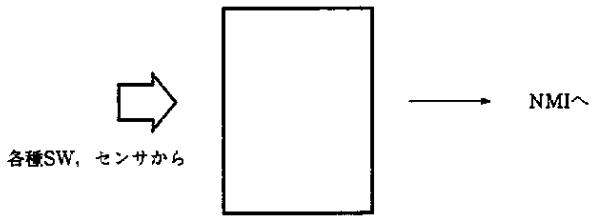
### (3) アドレス・デコード回路

アドレス・バスからSRAM，疑似SRAM，RTC，モデム，EPROMに対するCS信号を作成します(モデム，RTCは，メモリ・マップトI/Oとします)。




### (4) NMI信号発生回路

NMIにてSTOPモードから抜け出すため，各種スイッチ，センサからの信号の論理和 (OR) を取り，CPUへ信号を送ります。



# 第３章　画像読み取り部

V55PIデモセットでは，密着イメージ・センサを使用しており，この出力はアナログとなっています。
画像読み取り部の構成は図３-１に示すとおりです。
以下に，密着センサを使用した処理方法を説明します。

## ３.１　密着センサ用タイミング作成回路

密着センサには，FPGA内部で生成した500 kHzのクロック（CLK）と5 msごとのパルス（SIパルス）を入力します。

密着センサは，このクロックとSIパルスの入力により，5 msごとにFPGAにシリアル・データを出力します。このシリアル・データは，FPGA内部でパラレル・データに変換されます。変換されたデータはCPU（V55PI）側で画データ受け入れの準備ができた段階で内蔵DMAを起動して，ライン・メモリに取り込みます。密着センサより出力されたデータは，定期的（5 msごと）に出力されるため，このデータがメモリに取り込まれるか，メモリに取り込まれず廃棄されるかは，V55PIの内蔵DMAが転送許可状態になっているかどうかで決まります。

以下に１ライン分のデータ処理が終了した場合と，１ライン分のデータを処理中の場合の例を示します。

**○１ライン分のデータ処理が終了した場合**

V55PIの内蔵DMAは転送許可の状態になります。このとき，V55PIは内蔵DMAが転送許可の状態にあることをFPGAに通知します。FPGAは次のSIパルスを密着センサに発生し，その後８ビットのデータごとにV55PIに対してDMA要求信号（DREQ）を発生します。V55PIは，このDREQ信号により８ビットのデータをFPGAのバッファからライン・メモリに216回（216バイト＝1728ビット：１ライン分）DMA転送します。このように密着センサより出力されたデータはライン・メモリに取り込まれます。

**○１ライン分のデータを処理中の場合**

V55PIの内蔵DMAは転送不可の状態になります。このため，１ライン分のデータを処理中の場合には，密着センサから5 msごとにFPGAに出力されるデータはDMAが起動しないため，メモリに取り込まれずに廃棄されます。

上記のように，密着センサから出力されたデータはライン・メモリに取り込まれます。

保守／廃止

図３－１　画像読み取り部

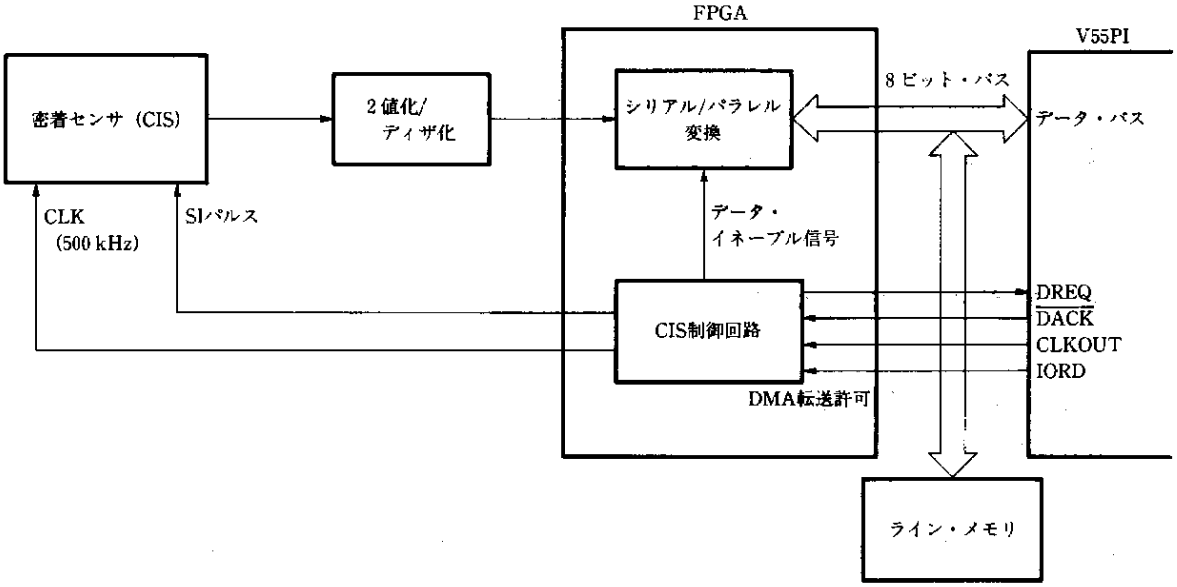
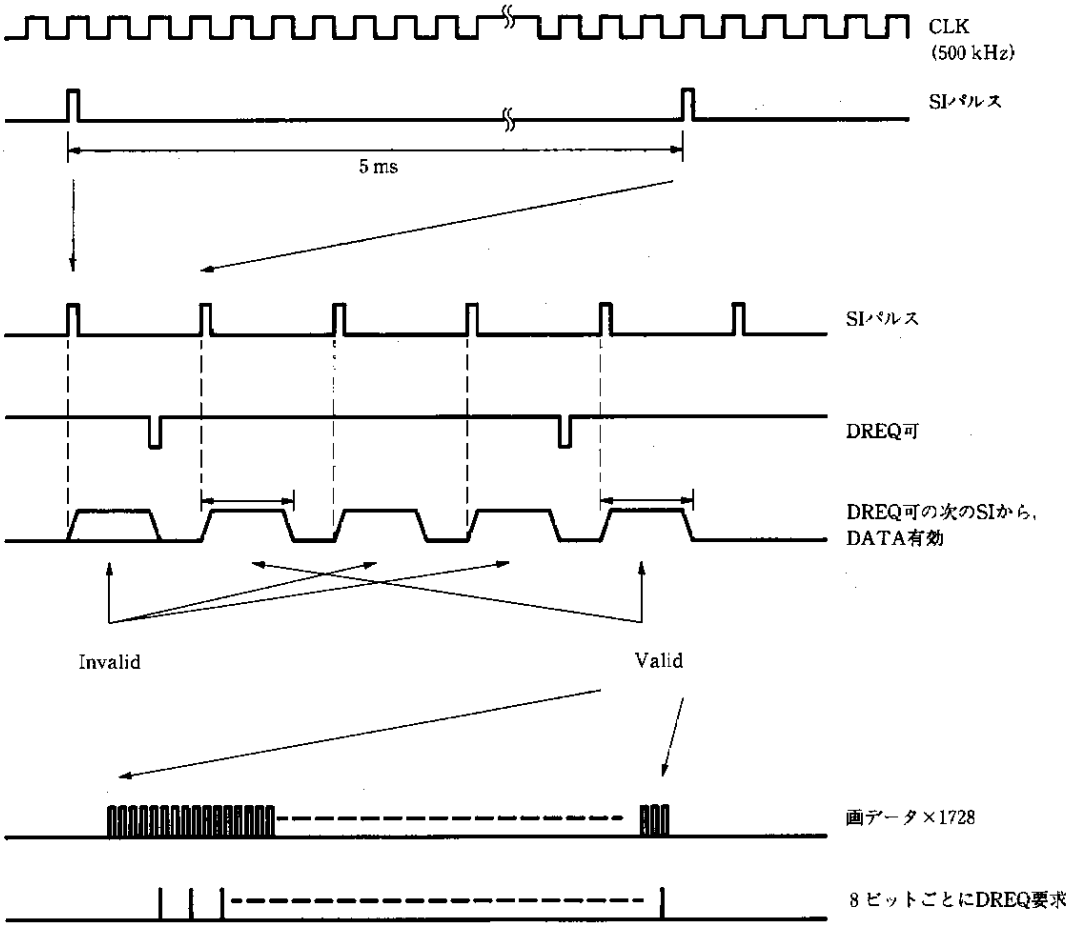



図３－２　密着センサ・データ読み込みタイミング



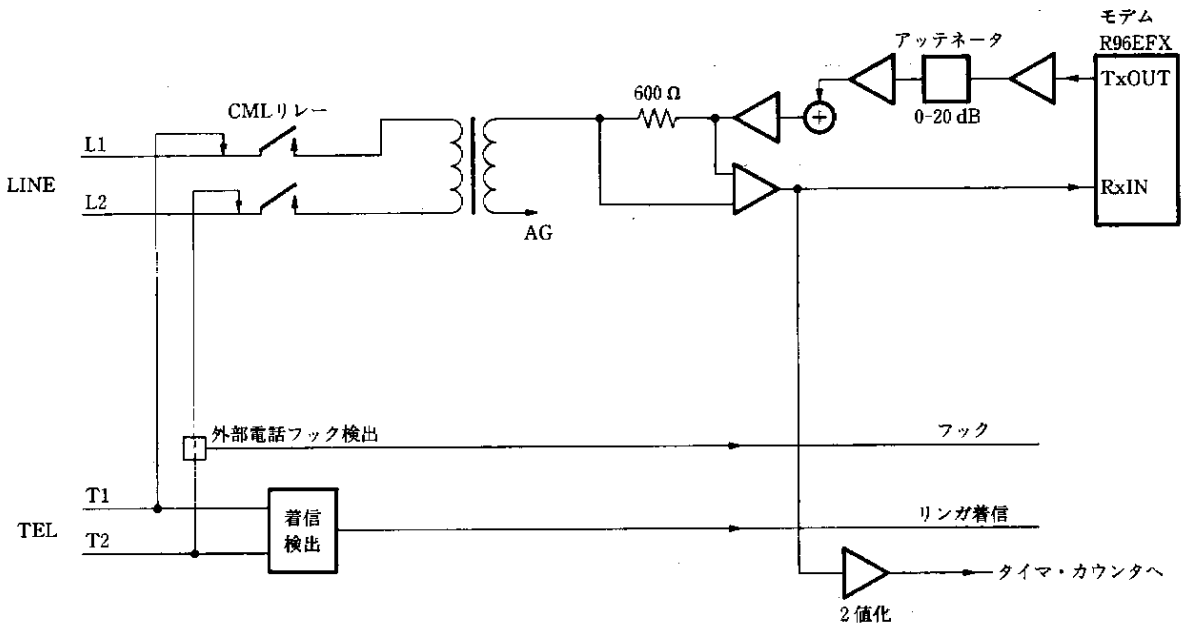
保守／廃止

# 第４章　回線インタフェース部

## ４.１　回 路 図 ★

次に回線インタフェース部の回路図を示します。

図４－１　回線インタフェース部の回路図




## ４.２　ネットワーク・コントロール・ユニット ★

ネットワーク・コントロール・ユニット（以下NCU）は，ファクシミリと一般公衆電話網との接続制御を行います。

次にNCUの機能を示します。

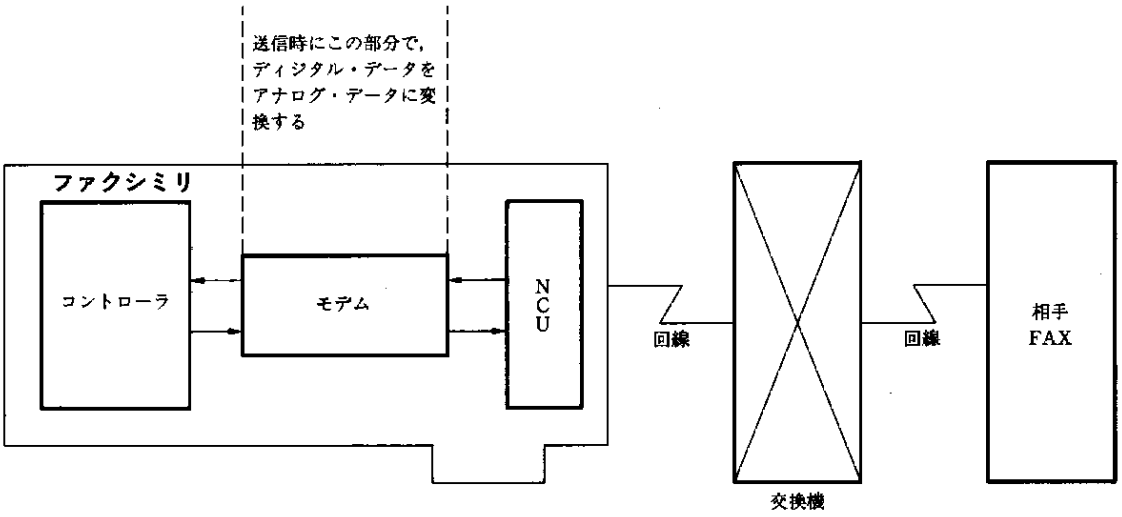
- 回線捕捉
- パルス／トーン信号のダイアリング
- 着信検出（呼び出し信号：16 Hz）
- 外部電話オフフック検出
- 各種トーナル信号発生
- モデム送信レベル調整

保守／廃止

## ★ 4.3 モデム

モデムとは，modulator-demodulatorの略で，A/D，D/A変換を行うものです。

図4－2　モデムの動作



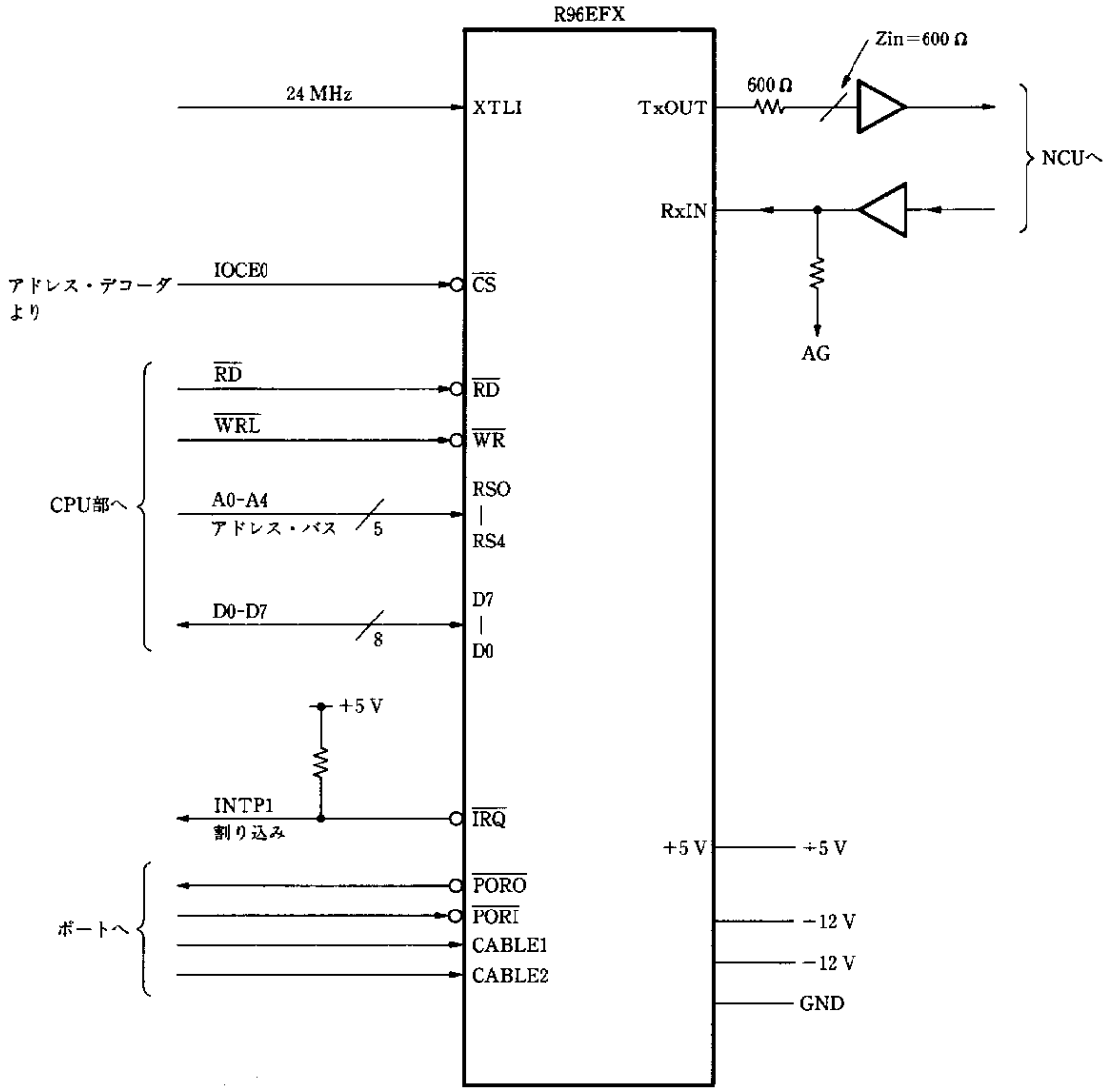

次に，モデムのハードウエア・インタフェースを示します（図4－3参照）。

図の中で，$\overline{CS}$はメモリ・マップトI/O用のアドレス・デコーダに接続され，モデムの$\overline{RD}$，$\overline{WR}$信号はCPUの$\overline{RD}$，$\overline{WRL}$に接続されています。

また，$\overline{PORO}$はI/Oポートの出力に接続されており，$\overline{PORI}$はI/Oポートの入力に接続されています。

モデムからの割り込み信号$\overline{IRQ}$は，CPUのINTP1に接続されており，CPUに対して外部割り込みを発生させます。

図４－３　モデムのハードウエア・インタフェース



保守／廃止

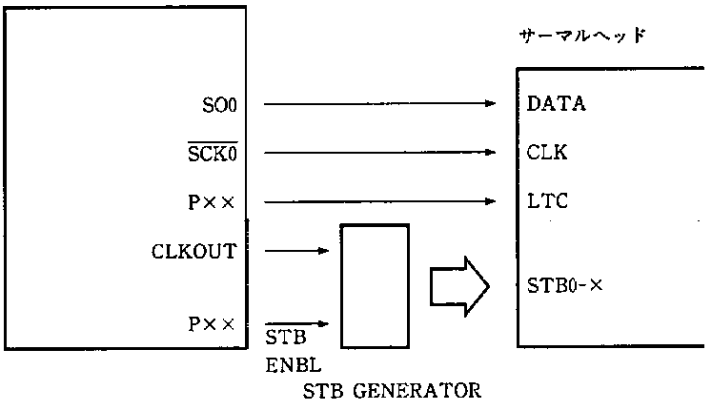
〔メ モ〕

# 第５章　記 録 部

V55PIデモセットでは，サーマル・プリント・ヘッドによる熱記録方式を採用しています。
以下に，そのインタフェースとタイミングを記述します。

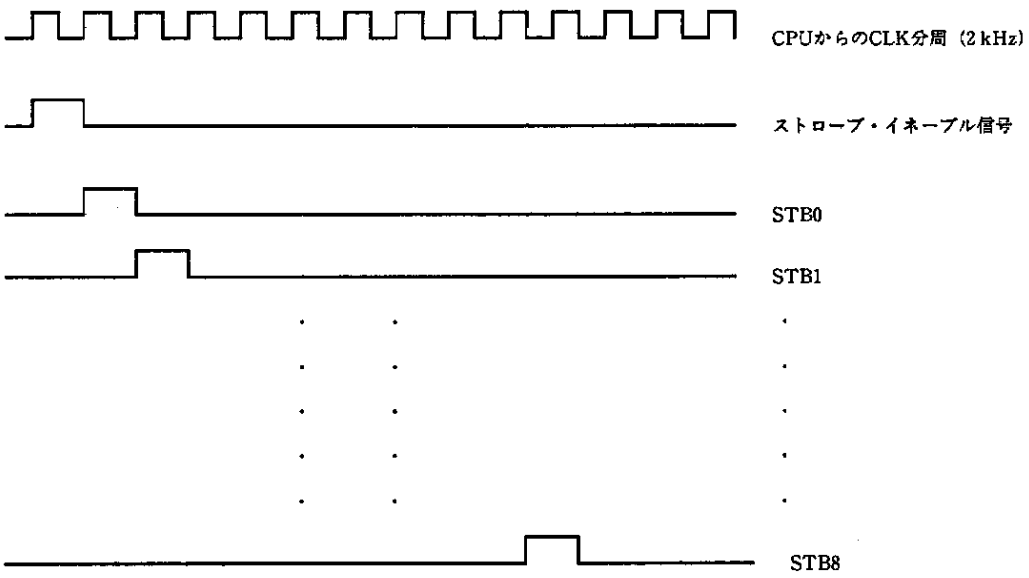
## ５.１　データ出力部（サーマル・ヘッド）

サーマル・ヘッドへのデータ転送は，シリアル・インタフェースをCSIモード設定し，これを利用します。
接続図は，以下のとおりです。



## 5.2 サーマル・ヘッド用ストローブ・パルス発生回路

9ストローブ・サーマル・ヘッドのストローブ信号発生回路をCPUのCLKOUT分周から作り出し，(2)のポート信号に従い出力します。



保守／廃止

# 第２編　ソフトウエア

〔メ モ〕

# 第１章　ソフトウエアの概要

この章では，FAXで動作するソフトウエアについて説明します。

## １．１　状態遷移

V55PIデモセット用ソフトウエアは，C言語，アセンブラ（共に，Microsoft社製）で記述されており，FAXとして，図１－１で示す状態遷移が保たれるように設計されています。

図１－１　状態遷移図

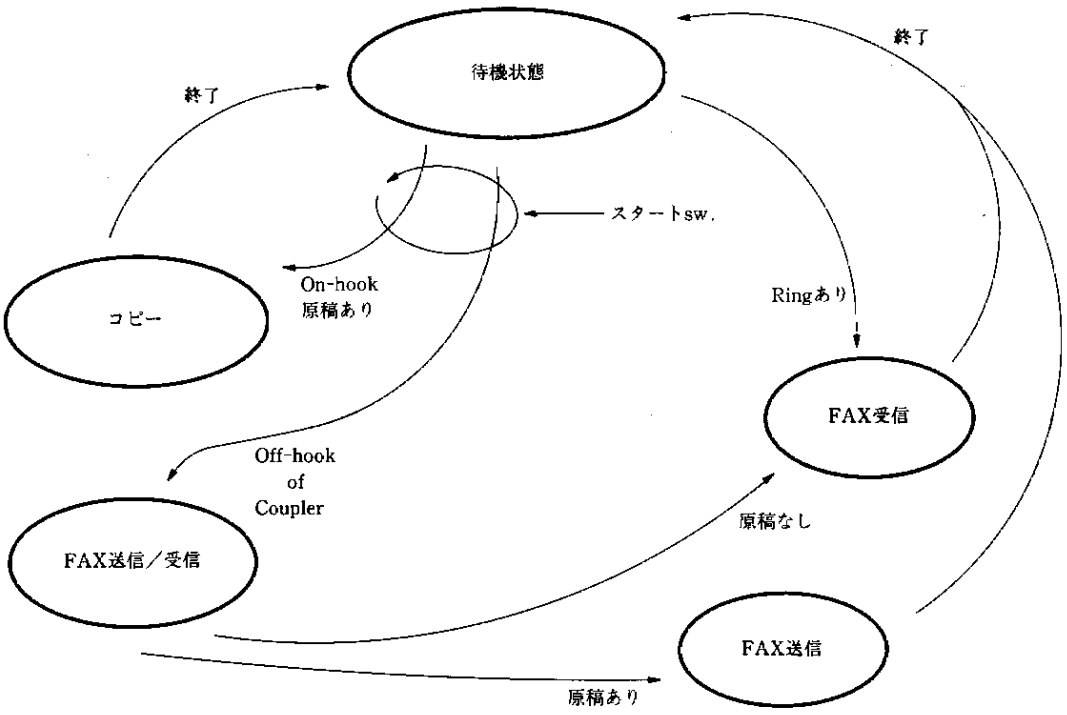


図1-1の状態遷移を詳しくしたものを，図1-2に示します。

図1-2　状態遷移フロー・チャート

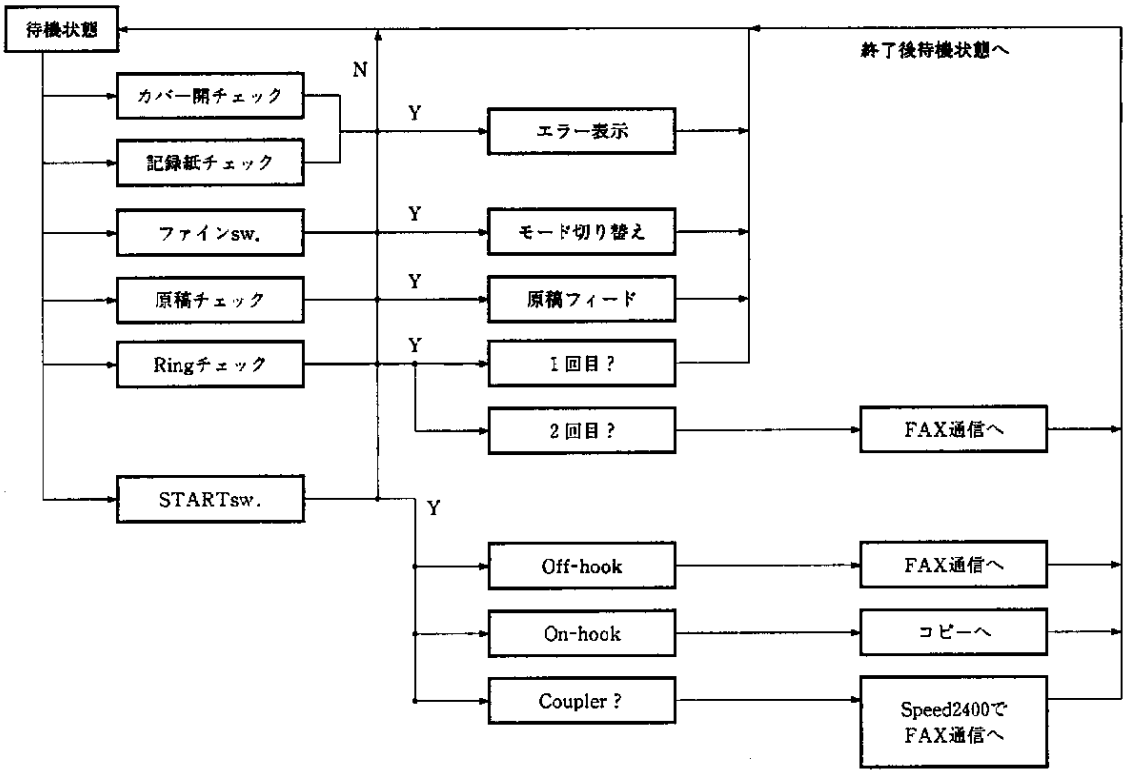



図1-2で示されるように，待機時は順次センサ，スイッチを監視し，Hook状態，原稿の有無により，コピー，FAX送信/受信と動作分けを行います。

## 1.2　ソフトウエア構成

セット用ソフトウエアは，ミディアム・モデルにて記述されており，このROM化は市販ディバッガ付属のミディアム・モデル用ROM化プログラムで行っています。

ソフトウエア中，全体を管理しているメイン・ルーチン，およびスピードを要求されないサブルーチンはC言語で，割り込みルーチン，高速処理ルーチンはアセンブラで記述されています。

次に，リンクされているすべてのファイルとその内容を表1-1に示します。

表1-1　関連ファイル一覧

| ファイル名 | 内　　容 | ページ |
|---|---|---|
| MAI.C | メイン・ルーチン。<br>各種スイッチ，センサの状態により，動作を決定。 | p.103～109 |
| INT.ASM | 割り込み時のバンク，ベクタ設定，および割り込み処理。 | p.110～123 |
| REG55.C | V55PI固有の内部特殊機能レジスタの初期設定，割り込み設定など。 | p.124～126 |
| MEM.C | メモリ割り付け，構造体確保など，メモリに関わる初期設定を行うルーチン。 | p.127 |
| BRW.C | 読み取り，印字に関するbios。<br>サーマル・ヘッドへのデータ転送／ヘッド制御，密着センサ制御など。 | p.128～134 |
| BMTR.C | 読み取り，印字両モータ・ドライブ制御用サブルーチン。 | p.135～141 |
| BBIO.C | LED，ブザー，タイマー制御，および各種センサ，スイッチ入力チェックなど。 | p.142～151 |
| CPY.C | コピー時の先頭／最終処理，およびコピー・ループ処理。 | p.152～154 |
| BRTC.C | RTCの設定，書き換えなどを行うルーチン。 | p.155～156 |
| SR.C | PhaseC中，画データを符号化しモデムへ送出，またはモデムからのデータを復号化しプリント・ヘッドへ出力するルーチン。 | p.157～175 |
| ECMSR.C | ECM通信のためのテスト・ルーチン。 | p.176～186 |
| TASK00.C | タスク管理，および状態チェックのためのルーチン。 | p.187 |
| TASK01.C | プリントのためのタスク制御。 | p.188～191 |
| TASK02.C | 画データ取り込み用タスク制御。 | p.192～193 |
| TASK03.C | ターミナル・メッセージ出力用タスク制御。 | p.194～197 |
| HEAD.C | ユーザ情報（ヘッダ，TEL番号，受信リング数）設定用ルーチン。 | p.198～203 |
| CRG.C | キャラクタ・パターン格納用ファイル。 | p.204～218 |
| ASR.ASM | 符号化／復号化データ，送信／受信割り込み処理ルーチン。 | p.219～221 |
| ACODE.ASM | 符号化処理初期化，制御ルーチン。 | p.222～232 |
| ADECODE.ASM | 復号化初期化，制御ルーチン。 | p.233～240 |
| SLEEP.ASM | 待機時，CPUをSTOP/HALTさせる処理ルーチン。 | p.241～242 |
| MHTBL_E.ASM | MH符号化用参照テーブル。 | p.243～246 |
| MHTBL_D.ASM | MH復号化用参照テーブル。 | p.247～252 |

**備考**　拡張子　ASM：アセンブラ

　　拡張子　C　　：C言語

表1-1のファイルを分類分けにすると，次のように大別されます。

| メイン・ルーチン | 各アプリケーション | Driver/Bios群 | |
|---|---|---|---|
| MAI.C | | | |
| 処理により ↓ | | | |
| | CPY.C | BRW.C | ASR.ASM |
| | HEAD.C | BMTR.C | ACODE.ASM |
| | | BBIO.C | ADECODE.ASM |
| | | BRTC.C | SLEEP.ASM |
| | | SR.C | |
| | | ECMSR.C | |
| | | TASK0×.C | |

| | |
|---|---|
| Initialize： | REG55.C |
| | MEM.C |
| DATA参照用： | CRG.C |
| | MHTBL_E.ASM |
| | MHTBL_D.ASM |

上記のファイルをリンクし，これを市販HEX変換ソフトを用いて，ROM用HEXファイルを作成します。

# 第２章　圧縮／伸長ソフトウエア

## ２.１　V55PIのFAX専用命令

V55PIでは，ソフトウエア・コーデック用の専用命令として，圧縮系命令と伸長系命令をサポートしています。

**[圧縮系命令]**

・データ送出命令（ALBIT）
・変化点テーブル作成命令（COLTRP）
・MH符号化命令（MHENC）
・MR符号化命令（MRENC）

**[伸長系命令]**

・EOL検出命令（SCHEOL）
・１ビット検出命令（GETBIT）
・MH復号化命令（MHDEC）
・MR復号化命令（MRDEC）
・画素データ作成命令（CNVTRP）

V55PIは，上記のFAX専用命令により，従来専用のLSI，ゲートアレイ，またはソフトウエアで行っていたコーデック処理を小規模で，簡単なプログラムにより高速に実現できます。

V55PIのFAX専用命令を実行する際，必要なメモリ領域を次に示します。

| | |
|---|---|
| レジスタ・ファイル | ：パラメータ設定用レジスタ・バンク |
| 外部RAM | ：符号化ライン変換テーブル，参照ライン変換テーブル |
| 外部RAM（ユーザ設定） | ：画データ・バッファ，送信バッファ，受信バッファ，プリント・バッファ |
| ROM | ：符号化変換テーブル，復号化変換テーブル |

**(1)　レジスタ・ファイル**

パラメータ設定用レジスタ・バンクです。圧縮処理，伸長処理のために各々１バンク必要です。送受信を同時に実行させない場合は，２つのバンクを確保しておく必要はありません。

**(2)　外部RAM**

・符号化ライン変換テーブル

符号化を行うために必要な変化点情報格納領域です。

（1ライン中に現れる最大の変化点数+2）ワードが必要です。

A4判，ノーマル・モードでは，1730ワード必要です。

・参照ライン変換テーブル

参照ラインの変化点情報格納領域です。

（1ライン中に現れる最大の変化点数+2）ワードが必要です。

A4判，ノーマル・モードでは，1730ワード必要です。

・画データ・バッファ

スキャナから読み込まれた画素データの格納領域です。

最低でも（1ラインの最大画素数÷8）バイトが必要です。

A4判，ノーマル・モードでは，216（1728÷8）バイト必要です。

・送信バッファ

モデムに符号化データを引き渡すためのバッファです。

1ワード以上あれば，命令の実行は可能です。1ラインの最大送信符号量にあわせて選択します（転送回数，実行時間の短縮のため）。

（1ラインの最大送信符号ビット数÷8）バイトが必要です。

A4判，ノーマル・モードでは，256～2Kバイト程度必要です。

・受信バッファ

モデムから受信された符号データの格納領域です。

最低でも（1ラインの最大受信符号ビット数÷8）バイトが必要です。

A4判，ノーマル・モードでは，256～2Kバイト程度必要です。

・プリント・バッファ

記録系に復号化された画素データを引き渡すためのバッファです。

（1ラインの最大画素数÷8）バイトが必要です。

A4判，ノーマル・モードでは，216（1728÷8）バイト必要です。

(3) ROM

・符号化変換テーブル

MH/MR符号化のための変換テーブルです。

512バイト必要です。

・復号化変換テーブル

MH/MR復号化のための変換テーブルです。

2304バイト必要です。

表2－1　必要なメモリの容量

| A4，ノーマル・モード | | | |
|---|---|---|---|
| メ　モ　リ | | 圧　縮　処　理 | 伸　長　処　理 |
| レジスタ・ファイル | | 1バンク | 1バンク |
| 外部ROM(テーブル) | | 512＋2304＝2816バイト | |
| 外部RAM | MH | 3676バイト＋送信バッファ<br>(3460バイト＋216バイト＋送信バッファ) | 3676バイト＋受信バッファ<br>(3460バイト＋216バイト＋受信バッファ) |
| | MR | 7136バイト＋送信バッファ<br>(3460バイト×2＋216バイト＋送信バッファ) | 7136バイト＋受信バッファ<br>(3460バイト×2＋216バイト＋受信バッファ) |

**備考**　送信バッファおよび受信バッファのサイズは任意（256～2 Kバイト程度）です。

保守／廃止

## 2.1.1　圧縮系命令

圧縮系命令について説明します。

### (1)　データ送出命令（ALBIT）

符号化処理においてEOL, FILLなど16ビット以下のデータを送信バッファへ出力するための命令です。

### (2)　変化点テーブル作成命令（COLTRP）

符号化処理において，画素データの変化点情報を変化点テーブルに生成する命令です。

### (3)　MH符号化命令（MHENC）

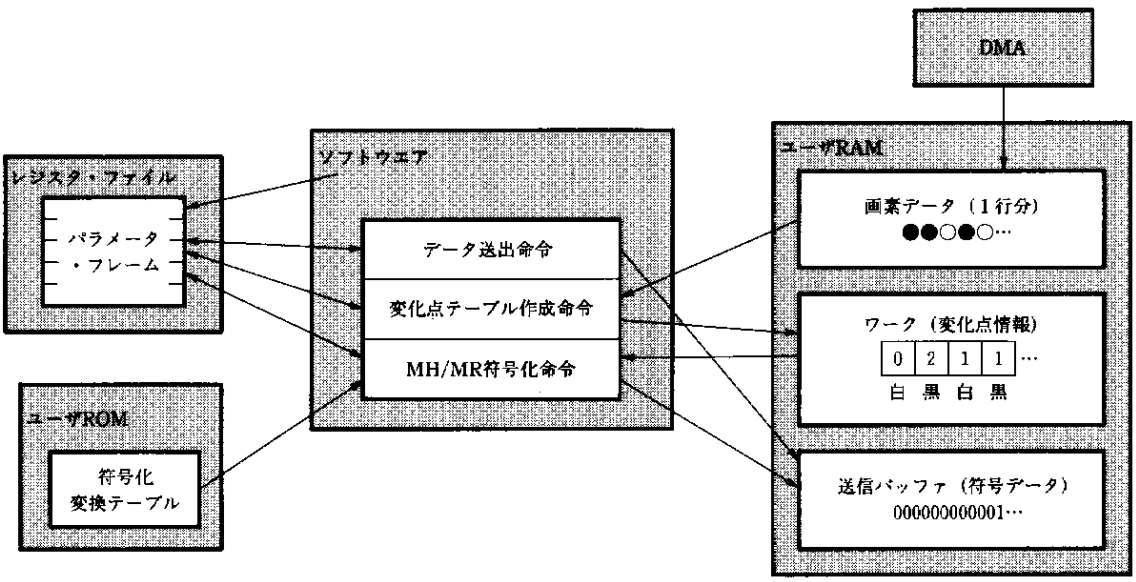
１ライン分の画素データの変化点情報を入力し，符号変換テーブルによりMH符号化を行います。その符号を送信バッファに送出する命令です。

### (4)　MR符号化命令（MRENC）

参照ラインの変化点情報と符号化ラインの変化点情報を入力し，符号変換テーブルにより１ライン分のMR符号化を行います。その符号を送信バッファに送出する命令です。

次に符号化専用命令とデータの関係について，図２－１に示します。

**図２－１　符号化専用命令とデータの関係**



## 2.1.2　伸長系命令

伸長系命令について説明します。

### (1)　EOL検出命令（SCHEOL）

復号化処理でEOLを検出するための命令です。

### (2)　1ビット検出命令（GETBIT）

復号化処理でタグなどの1ビットを検出するための命令です。

### (3)　MH復号化命令（MHDEC）

受信バッファの符号データを入力し，復号化変換テーブルによりMH復号化を行います。その生成された変化点情報を変化点テーブルに送出する命令です。

### (4)　MR復号化命令（MRDEC）

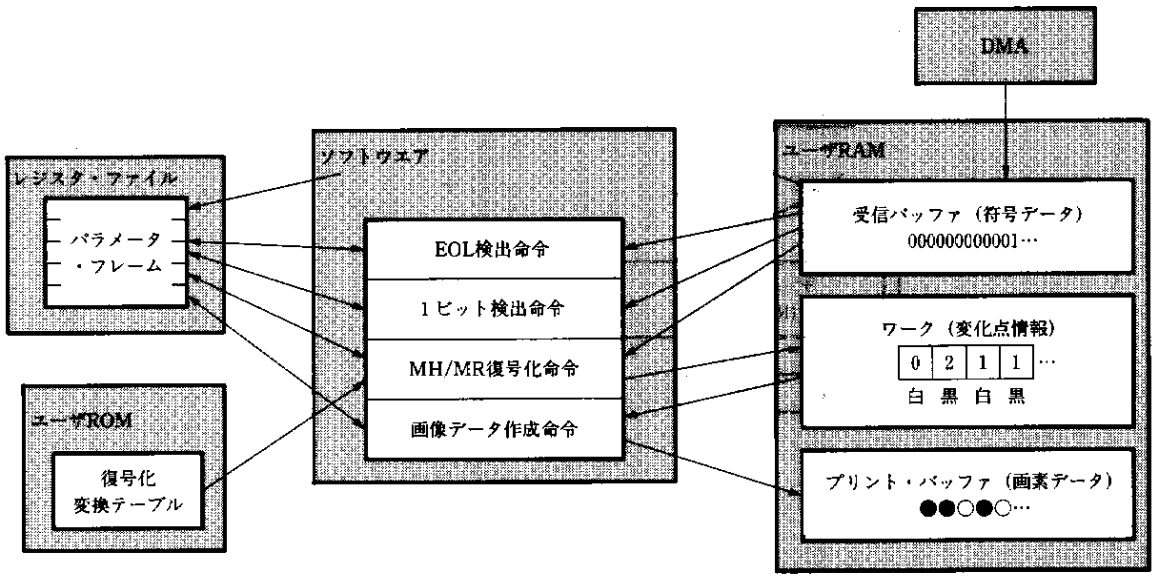
受信バッファの符号データと参照ラインの変化点情報を入力し，復号化変換テーブルによりMR復号化を行います。その生成された変化点情報を変化点テーブルに送出する命令です。

### (5)　画素データ作成命令（CNVTRP）

変化点テーブルの1ライン分の変化点情報を，画素データに変換し，プリント・バッファに送出する命令です。

次に復号化専用命令とデータの関係について，図2-2に示します。

図2-2　復号化専用命令とデータの関係

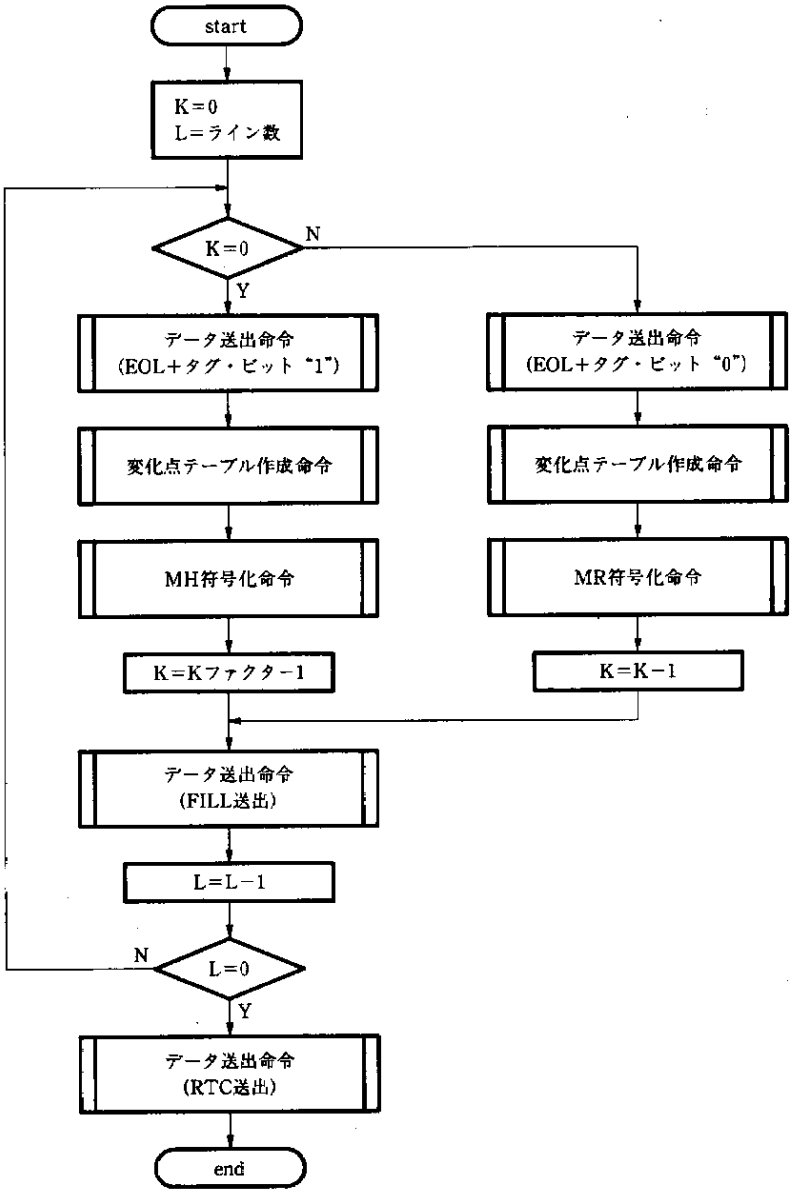


## 2.1.3　符 号 化

符号化を行う場合，まず，データ送出命令を用いて，送信バッファにEOL(MR符号化の場合はEOL+タグ・ビット）を送出します。次に1ライン分の画素データを変化点テーブル作成命令により，変化点情報に変換し，ワーク・エリアに格納します。この変化点情報はMH/MR符号化命令により符号化変換テーブルを用いて符号データに変換され，送信バッファに送出されます。

例として，MR符号化の処理フローを図2-3に示します。

図2-3　MR符号化の処理フロー



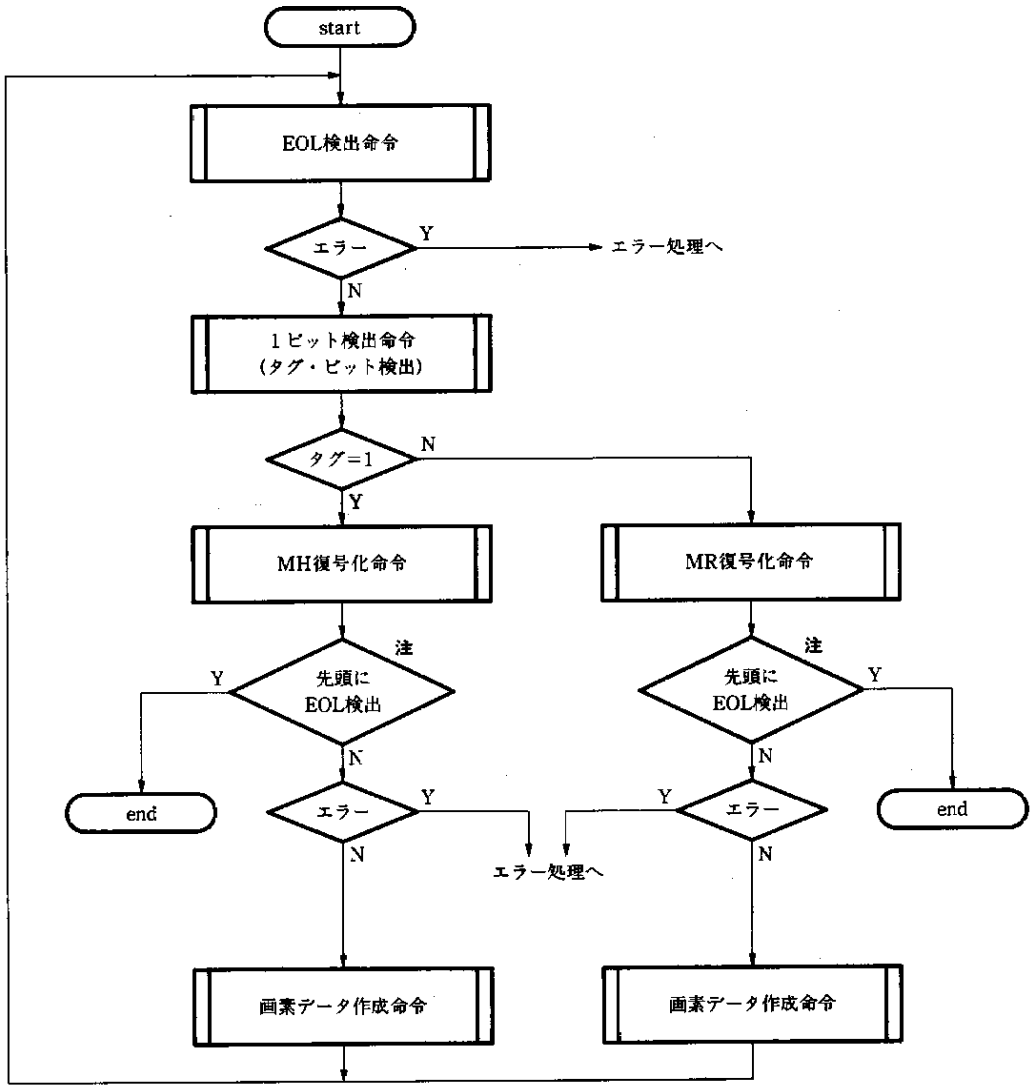
保守／廃止

## 2.1.4　復 号 化

復号化を行う場合，まず，EOL検出命令を用いて，符号データからEOL（MR符号化の場合は1ビット検出命令よりタグ・ビット）を検出します。次に符号データをMH/MR復号化命令により，1ライン分の変化点情報に変換しワーク・エリアに格納します。このとき，符号化データは復号化変換テーブルを用いて変化点情報に変換されます。この変化点情報を画素データ作成命令により画素データに変換し，プリント・バッファに送出します。

例として，MR復号化の処理フローを図2－4に示します。

図2－4　MR復号化の処理フロー




注　RTCは2個のEOLで検出

保守／廃止

## 2.1.5　変化点テーブル

MR符号化／復号化時には，最低でも2ライン分の画データの変化点情報を格納する変化点テーブルが必要です。

次に変化点テーブルを示します。

符号化時：符号化ライン変化点テーブル
　　参照ライン変化点テーブル
復号化時：復号化ライン変化点テーブル
　　参照ライン変化点テーブル

このプログラムでは，2ライン分の変化点テーブルとして，変化点テーブルAと変化点テーブルBを使用しています。この変化点テーブルAと変化点テーブルBは，図2-5のようにKパラメータの値(ノーマル・モードのときK=2，ファイン・モードのときK=4）によって，符号化（復号化）ライン用と参照ライン用とに切り替えて使用しています。

前ラインの符号化（復号化）時に，符号化（復号化）ライン用として使用した変化点テーブルを，次のラインの符号化(復号化)時に参照ライン用の変化点テーブルとして使用することにより，MR符号化（復号化）は，各ラインに対してのみFAX専用命令を実行することで実現できます。

また，実際のプログラミングでは，Kパラメータの値が偶数か奇数かによって，変化点テーブルを以下のように符号化（復号化）用と参照ライン用に区別して使用しています。

図2-5　MH符号化（復号化）（K＝4）の変化点テーブル

K＝偶数[注]
　変化点テーブルA：符号化（復号化）ライン変化点テーブル
　変化点テーブルB：参照ライン変化点テーブル
K＝奇数
　変化点テーブルA：参照ライン変化点テーブル
　変化点テーブルB：符号化（復号化）ライン変化点テーブル

MH（1ライン目）→MR（2ライン目）→MR（3ライン目）→MR（4ライン目）
K=0　　K=1　　K=2　　K=3

| | | | | |
|---|---|---|---|---|
| 変化点テーブルA | 符号化ライン→参照ライン | | 符号化ライン→参照ライン | |
| 変化点テーブルB | 参照ライン | 符号化ライン→参照ライン | | 符号化ライン |

注　MH符号化（復号化）時にも使用

符号化／復号化の画データ，変化点テーブル，符号化バッファのエリア，および各サイズは，表2－2のとおりです（画データは，実際のビット展開された画生データを示し，符号化バッファとは，符号化されたデータ格納用バッファを示します）。

**表2－2　符号化／復号化時のバッファ・エリア，およびサイズ**

| バッファ名 | 目　　的 | 先頭アドレス | サイズ |
|---|---|---|---|
| 画データ | 画データの取り込み，出力用 | 20000H | 2000H |
| 変化点A | 変化点テーブル格納1(MH/MR) | 24000H | 2000H |
| 変化点B | 変化点テーブル格納2 (MR) | 26000H | 2000H |
| 符号化 | 符号化データ格納 | 22000H | 2000H |

## 2.2　圧縮／伸長ソフトウエアの概要

この節以降では，V55PIデモセットで実際にインプリメントしたプログラムの説明をします。

フロー・チャート，および各レジスタ設定について説明します。

ソース・ファイルは，「SR.C」，「ACODE.ASM」，「ADECODE.ASM」，「ASR.ASM」に格納されています。ただし，この中の主要部分を抜粋し，説明します。

一連のソフトウエアの概略をフロー・チャートに示します（「A.10　SR.C」を参照してください）。

### (1)　1ページ符号化処理

図2-6　1ページ符号化処理のフロー・チャート（Send()）

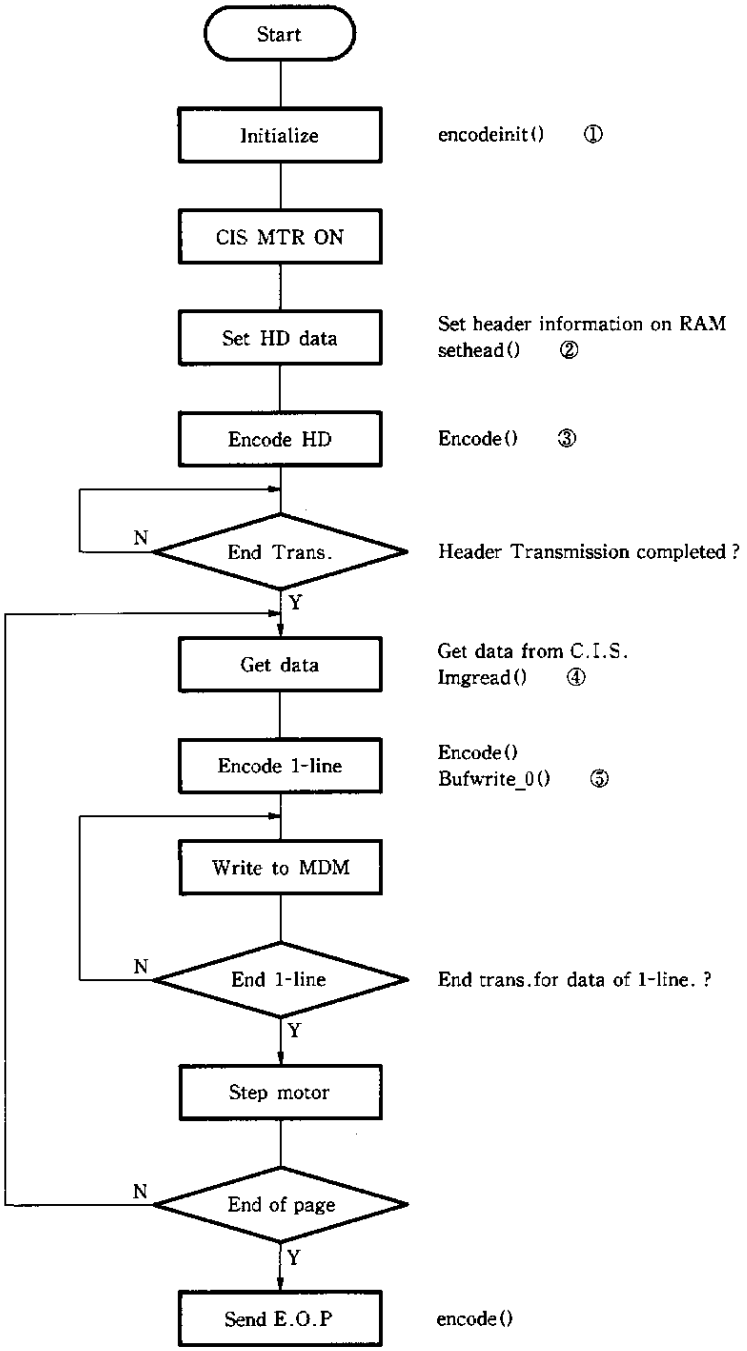


図2－7　encode()

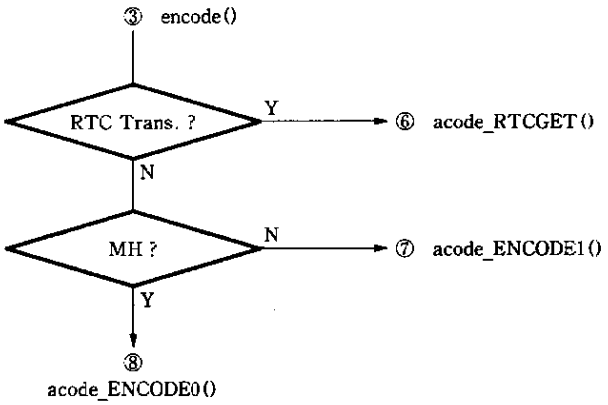
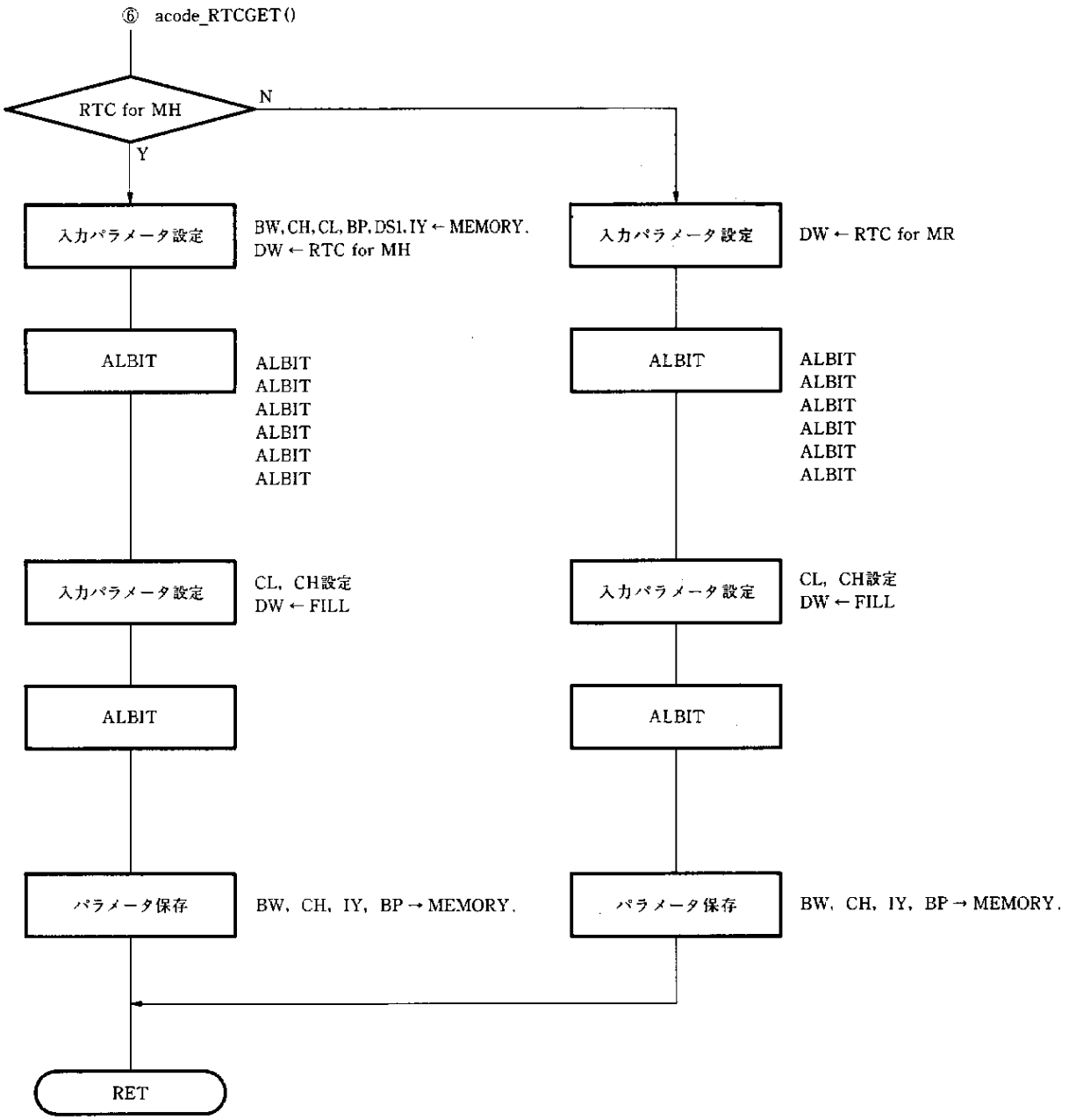


図2－8　acode_RTCGET()



以下，符号化につき「A.10　SR.C」，「A.19　ACODE.ASM」を参照してください。
図中の各ルーチン（①～⑧）について，説明します。

① encodeinit()

MH/MR，Fine/Normalによって，あらかじめ確保された必要変数の初期化を行います。
この時点では，まだFAX専用命令は使用しません。

② sethead()

ヘッダとして送出されるデータ（ユーザ・ネーム，ページ数，年月日など）を1ラインずつビット展開し，これをメモリにセットします。

③ encode()

ビット展開されたデータをコーディングする符号化の主ルーチンです（図2-7を参照）。ALBIT，COLTRP，MHENCと一連のFAX専用命令を使用します。

④ imgreadf()

DMAを起動し，指定された画データ・バッファへ取り込むルーチンです。
特に，複雑なことをする訳ではなく，DMAの起動とデータの受け渡しです。

⑤ bufwrite_0()

符号化データ列をモデム送信用バッファへ書き込むルーチンです。
符号化命令の出力パラメータ中の情報を元に，バッファへデータを書き込み，データ数をカウント・アップします。

⑥ acode_RTCGET()

MH/MRのRTC（リアルタイム・クロック）をFAX専用命令にて作り出し，符号化バッファへ書き込むルーチンです（図2-8を参照）。

⑦ acode_ENCODE1()

1ラインMR符号化を行うルーチンです。
ALBIT，COLTRP，MRENCと一連のFAX専用命令を用いて，MR符号化を行います。詳細は，「2.4.1　MR符号化」を参照してください。

⑧ acode_ENCODE0()

1ラインMH符号化を行うルーチンです。
上記のacode_ENCODE1()と同様に，FAX専用命令を用いてMH符号化を行います。詳細は，「2.3.1　MH符号化」を参照してください。

(2)　１ページ復号化処理

図２－９　１ページ復号化処理のフロー・チャート（Receive()）

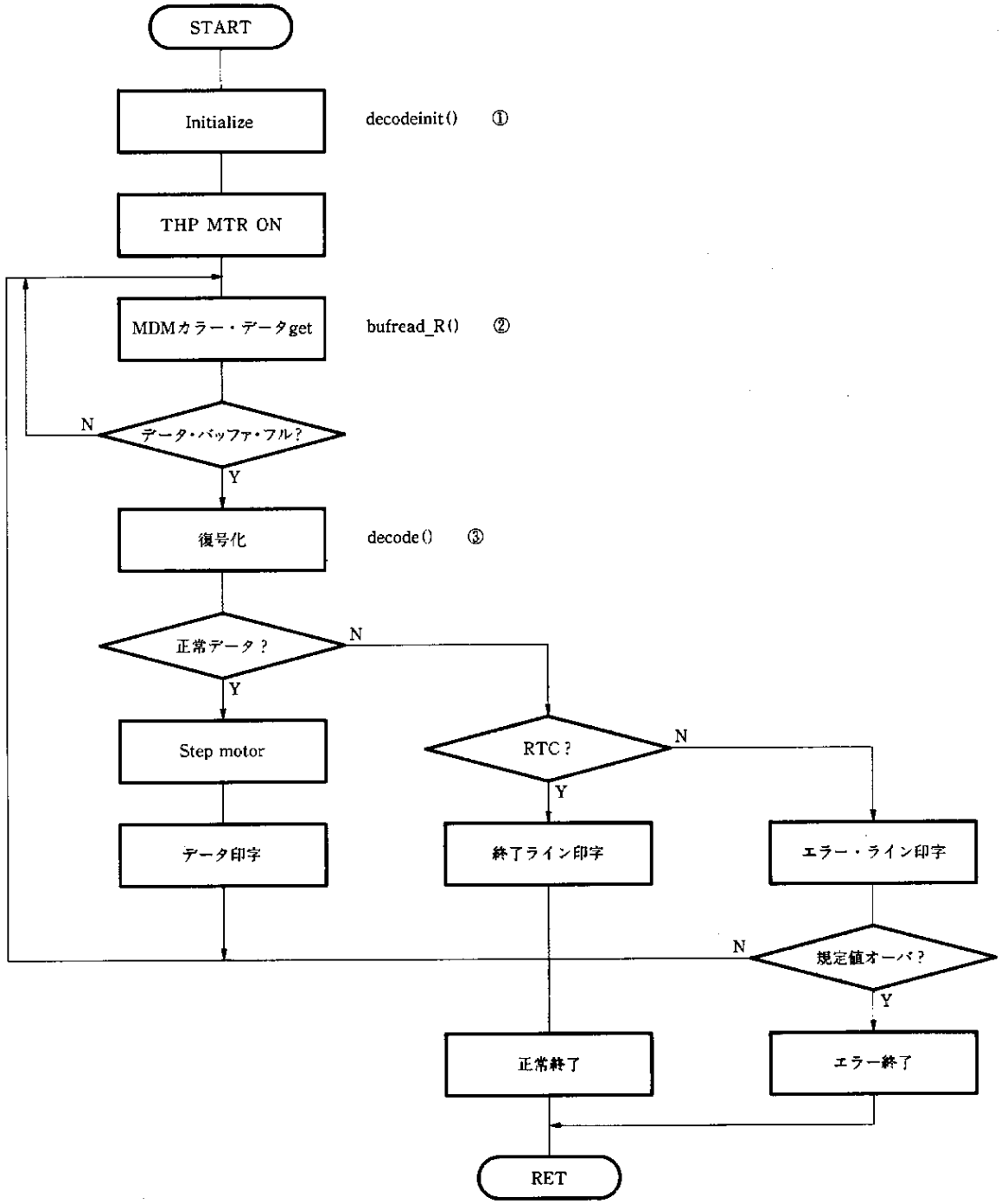
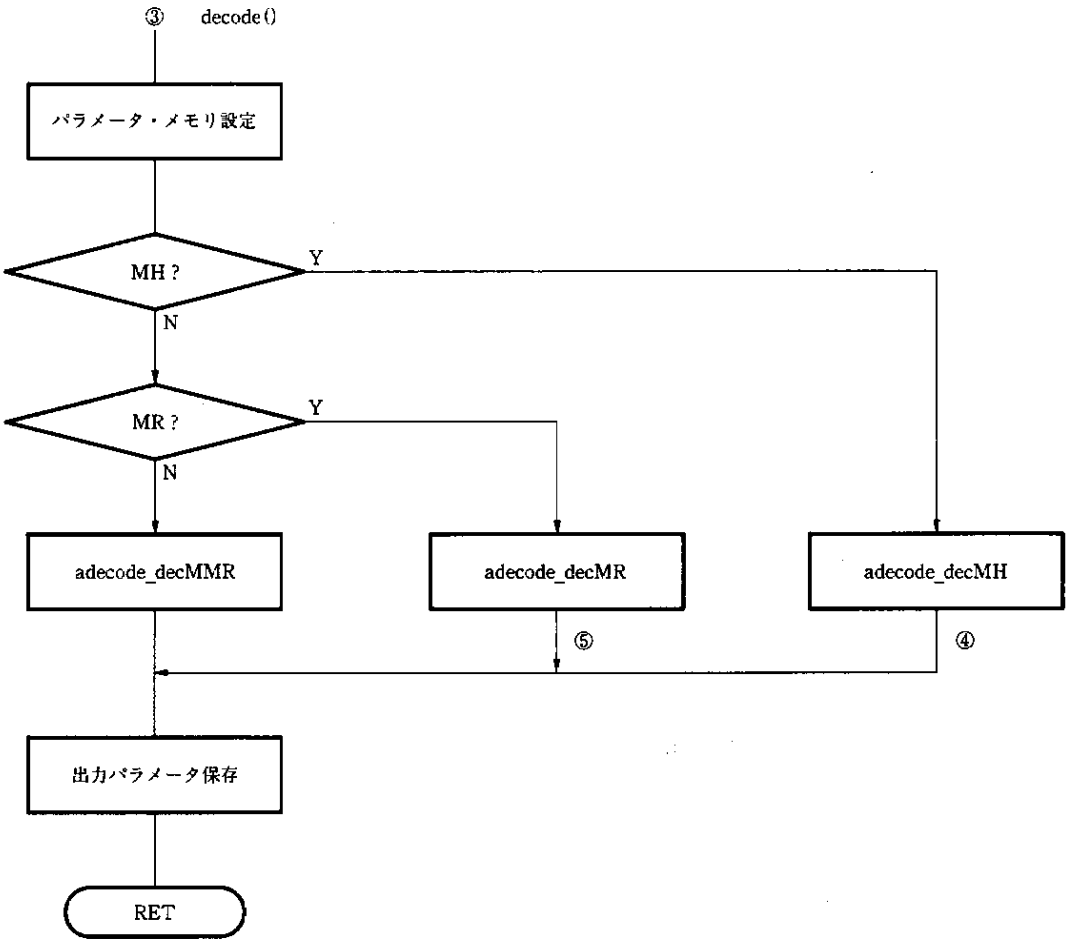


図2－10　decode()



以下，復号化につき「A.10　SR.C」,「A.20　ADECODE.ASM」を参照してください。
図中の各ルーチン（①~⑤）について説明します。

① decodeinit()

復号化の際，各メモリ変数，レジスタを初期化するルーチンです。

② bufread_R()

モデムからのデータ取り込みは，すべて割り込みにて処理します。

割り込み処理ルーチンにて,バッファへ取り込まれたデータをワーク・エリアへ取り込むルーチンです。

③ decode()

上記，復号化されたままのデータを実際の画データへ展開するルーチンです。次に示すFAX専用命令を使用します。

SCHEOL，MHDEC，CNVTRP，GETBIT，MRDEC

④ adecode_decMH

1ラインMH復号化を行うルーチンです。

詳細は，「2.3.2　MH復号化」を参照してください。

⑤ adecode_decMR

1ラインMR復号化を行うルーチンです。

詳細は，「2.4.2　MR復号化」を参照してください。

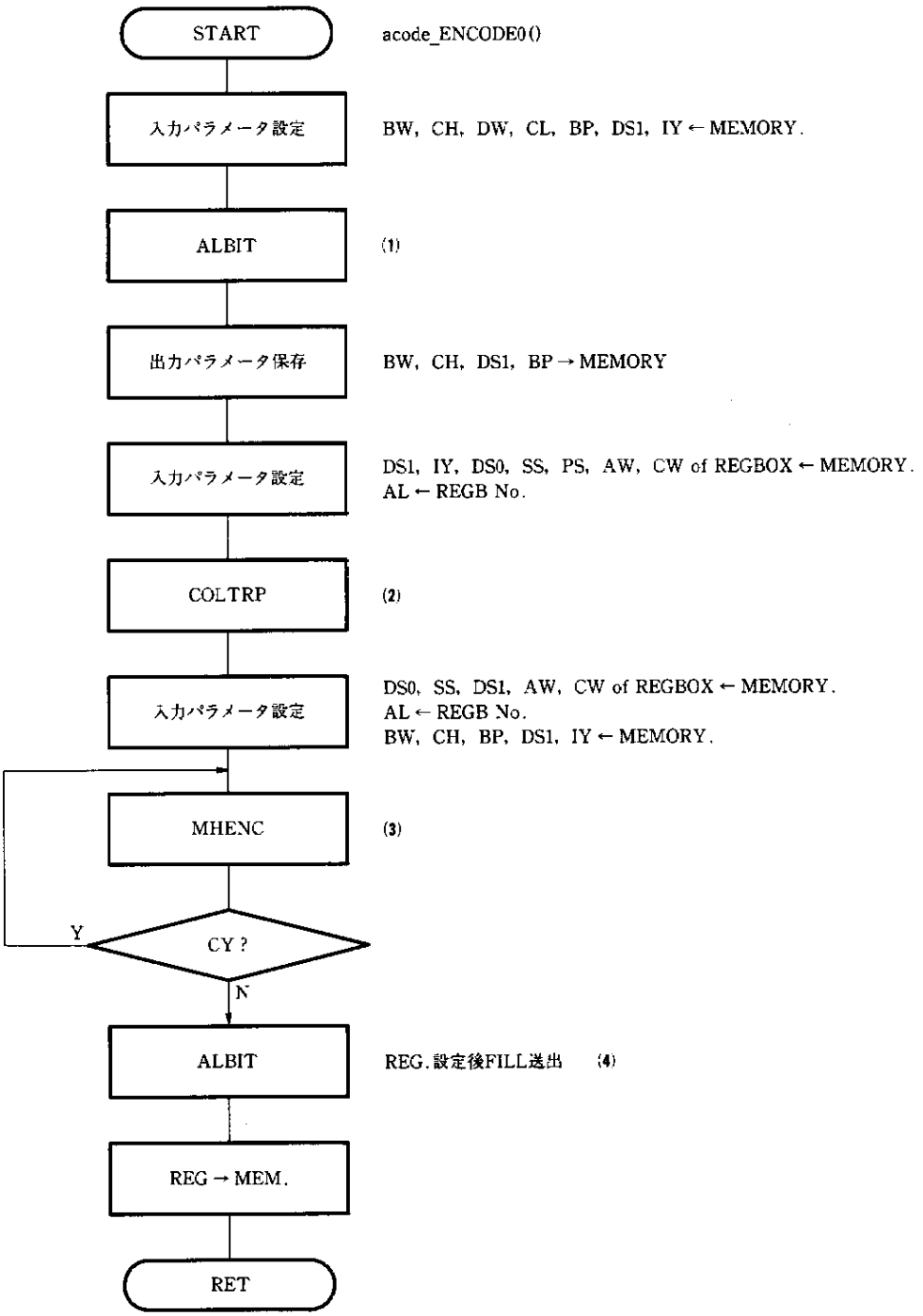
## 2.3　MH符号化／復号化

### 2.3.1　MH符号化

acode_ENCODE0()にて，1ラインの符号化を行います（「A.19　ACODE.ASM」を参照してください）。

次に，そのフロー・チャートを示します。

図2-11　acode_ENCODE0()



図中の(1)～(4)の処理を次に示します。

(1)　EOL符号を出力します。

(2)　画素データを読み込み，変化点情報から変化点テーブルを作成します。

(3)　1ラインの画素データの変化点情報を入力し，符号変換テーブルによりMH符号化を行います。CYフラグにより，処理が終了したかどうかを判断しています。

(4)　FILLビットを送出しています。

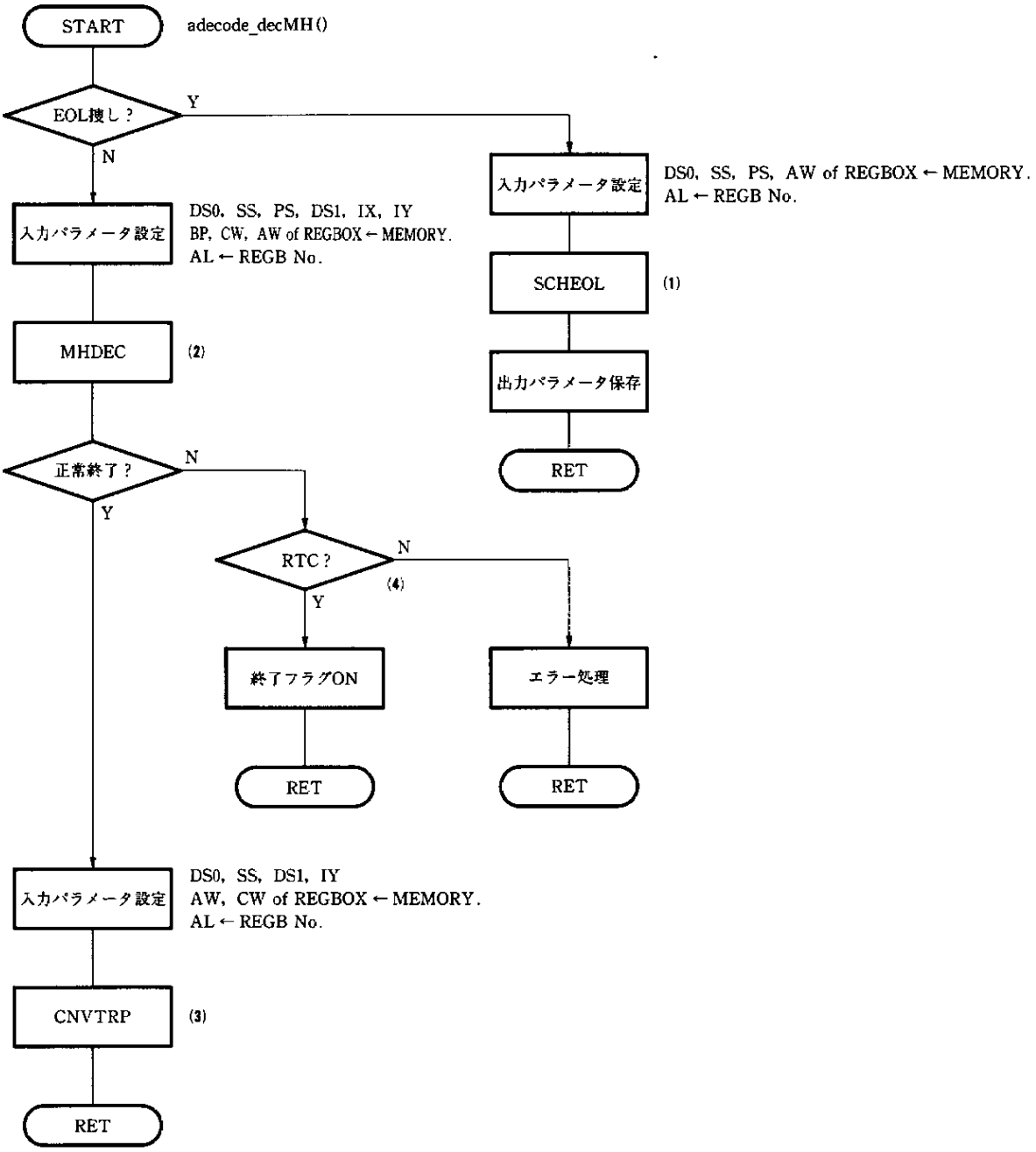
保守／廃止

## 2.3.2　MH復号化

adecode_decMH()にて，MH復号化を行います（「A.20　ADECODE.ASM」を参照してください）。

次にそのフロー・チャートを示します。

図2－12　adecode_decMH()



図中の(1)～(4)の処理を次に示します。

(1)　EOLを検出します。

(2)　符号データを入力し，復号化変換テーブルによりMH復号化を行います。出力されるのは，変化点情報です。

(3)　変化点情報から画素データを生成します。

(4)　１ページが終了であれば，終了します。

保守／廃止

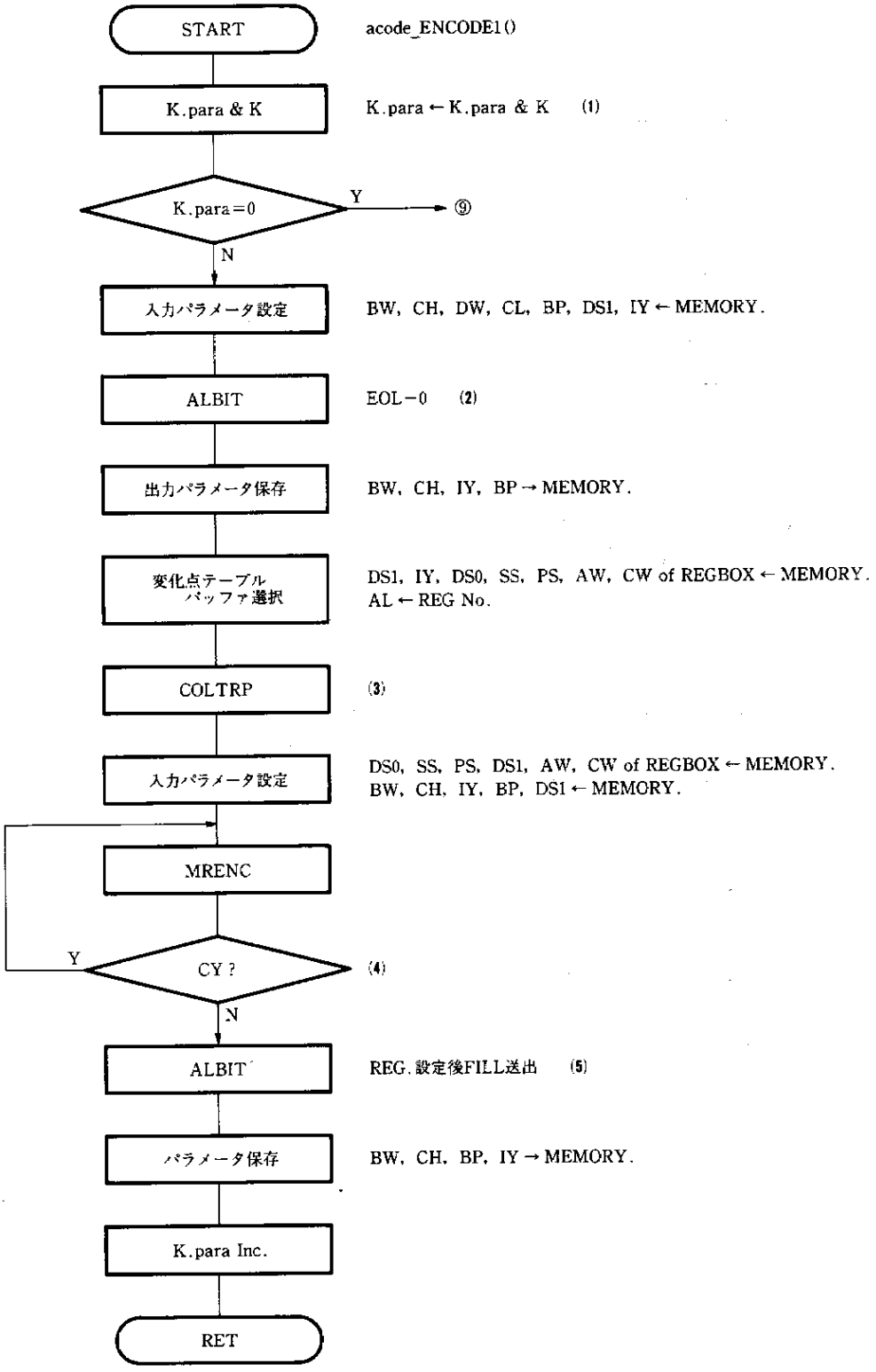
## 2.4　MR符号化／復号化

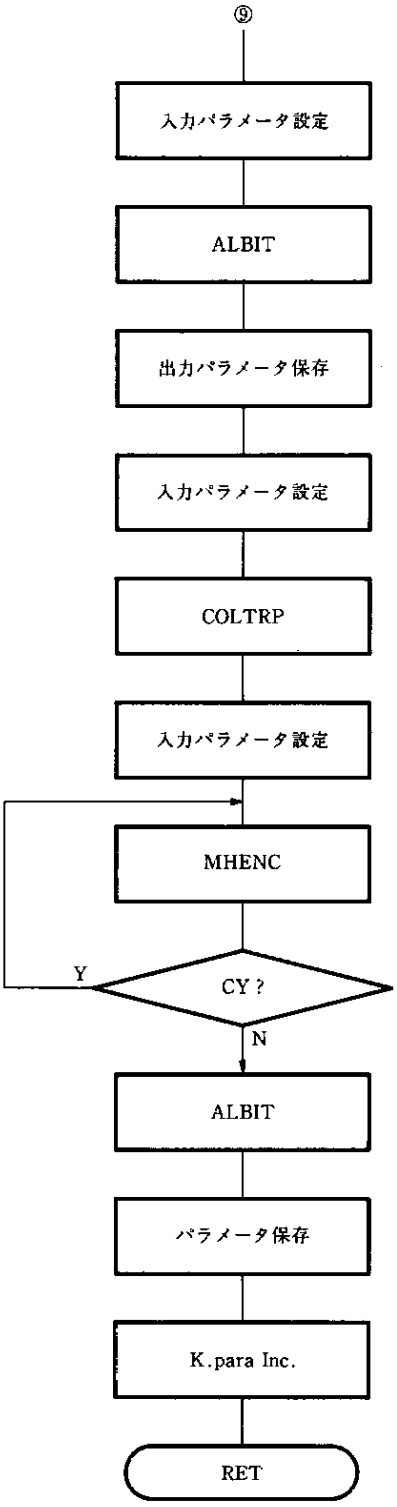
### 2.4.1　MR符号化

acode_ENCODE1()にて，1ラインのMR符号化を行います（「A.19　ACODE.ASM」を参照してください）。

次にそのフロー・チャートを示します。

図2－13　acode ENCODE1()






入力パラメータ設定：BW, CH, DW, CL, BP, DS1, IY ← MEMORY.

ALBIT：EOL+1

出力パラメータ保存：BW, CH, IY, BP → MEMORY

入力パラメータ設定：DS1, IY, DS0, SS, PS, AW, CW of REGBOX ← MEMORY.
AL ← REG No.

入力パラメータ設定：DS0, SS, PS, DS1, AW, CW of REGBOX ← MEMORY.
BW, CH, IY, BP, DS1 ← MEMORY.

ALBIT：REG.設定後FILL送出

パラメータ保存：BW, CH, BP, IY → MEMORY.

図中の(1)～(5)の処理を次に示します。

(1)　Kパラメータを設定し，もしK＝0であれば，MH符号化の処理へ移行します。⑨以降のフローはMH符号化処理になります。

(2)　EOLを送出します。

(3)　画素データを読み込み，変化点情報から変化点テーブルを作成します。

(4)　1ラインの画素データの変化点情報を入力し，符号変換テーブルによりMR符号化を行います。CYフラグにより処理が終了したかどうかを判断しています。

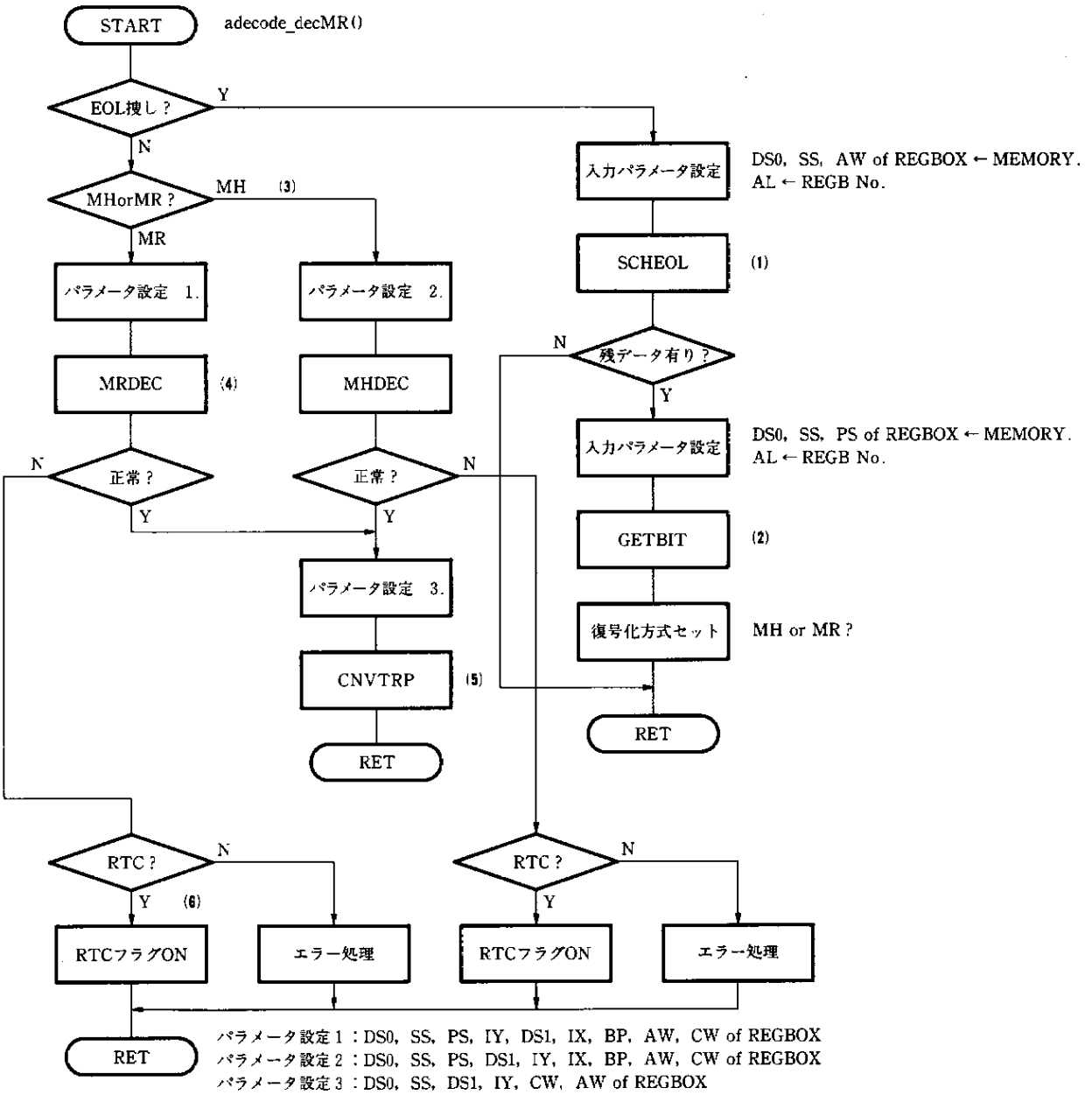
(5)　FILLビットを送出しています。

## 2.4.2　MR復号化

adecode_decMR()にて，MR復号化を行います（「A.20　ADECODE.ASM」を参照してください）。

次にそのフロー・チャートを示します。

図2－14　adecode_decMR()



図中の(1)～(6)の処理を次に示します。

(1)　EOLを検出します。

(2)　タグ・ビットを検出し，MR復号処理かMH復号処理かの方式設定をします。

(3)　MH復号処理方式であれば，MH復号処理を実行します。

(4)　MR復号処理方式であれば，符号データと参照ラインの変化点情報を入力し，復号化変換テーブルによりMR復号化を行います。出力されるのは，変化点情報です。

(5)　変化点情報から画素データを生成します。

(6)　１ページが終了であれば，終了します。

保守／廃止

〔メ モ〕

# 付録A　コーディング・リスト　★

ここでは，以下のプログラム・リストを表示します。

- MAI.C
- INT.ASM
- REG55.C
- MEM.C
- BRW.C
- BMTR.C
- BBIO.C
- CPY.C
- BRTC.C
- SR.C
- ECMSR.C
- TASK00.C
- TASK01.C
- TASK02.C
- TASK03.C
- HEAD.C
- CRG.C
- ASR.ASM
- ACODE.ASM
- ADECODE.ASM
- SLEEP.ASM
- MHTBL_E.ASM
- MHTBL_D.ASM
- ADECOMR.ASM
- V25.MAC
- V55PI.MAC
- V55PI_2.MAC
- V55PI_3.MAC
- MC329.MAC
- SEGMENT.MAC
- BASICDEF.H

- CONSTBIN.H
- HD329.H
- INDAT55.H

## A.1　MAI.C

MAI.C

```
/*********************************************************************/
/*      FILE NAME       'ＭＡＩ．Ｃ'                                 **/
/*      メインプログラム                                          **/
/*      1999/01/01                                                   **/
/*********************************************************************/

/*      ２進数ファイル  **/
#include        "constbin.h"

/*      ＃３２８固有ファイル    **/
#include        "hd329.h"

/*      Ｖ５５ＰＩ内部データ領域・マップ **/
#include        "indat55.h"

uchar   fdate[16];
uchar   RNGM;
uchar   RNGCM;
uchar   TELNO[20];      /* 実際はバックアップする。 **/
uchar   SPEED;  /* 電送スピード**/
uchar   LEV;
struct  {       uchar   poll_tx_sw:1;   /* ポーリング送信（被呼）**/
                uchar   poll_rx_sw:1;   /* ポーリング受信（発呼）**/
                uchar   ecm_sw  :1;     /* ＥＣＭ **/
                uchar   mmr_sw  :1;     /* ＭＭＲ **/
                uchar   sfin_sw :1;     /* Ｓファイン **/
        }memory_switch;

uchar   FMODE;

/*      ＰＬＡ入出力アドレス    **/
extern  uchar far* PLA;

/*      extern  uchar   STP;    **/

uchar           rrk_genco;

/* ************************************************** *
 *      メインループ　キー入力，自動受信待ち          *
 * ************************************************** **/
main()
{
uint    cc;
ulong   t_10s;
uchar   i;

        startup();
        ring_clear();
        if(!irepon(GENCS)) endfeed();
        timeclr();


        while(1){
                timeset(&t_10s,10000);

                while(1){
                        timeclr_A();
                        rollchk();
                        if( !task00_exist('?') ){
                                genkochk();
                        }
                        coverchk();
                        starlwink();
                        thtempchk();

                        if(!irepon(GENCS)){
                                rrk_genco = ON;
                        }else{
                                rrk_genco = OFF;
```

1

MAI.C

```
                        }

                        if( !task00_exist('?') ){

                                if( (cc = key_search()) != 0 ){
                                        timeclr();
                                        fahren(cc);
                                        break;
                                }

                                if (irepon(RINGS)) {
                                        timeset(&t_10s,20000);
                                }
                                if(ring_check_10m(RNGM) || cu_check() || cu2_check()){
                                        timeclr();
                                        fahren(0x10);
                                        break;
                                }
                        }else{
                                if(keyin(ON)){
                                        pepo(3);
                                }
                        }

                        if (timechk(&t_10s)) {
                                ring_clear();
                                if( !task00_exist('?') ){
                                        if(!neruna()){
                                                clrthermal_0();
                                                motors_off();
                                                SLEEP();
                                                break;
                                        }else{
                                                timeset(&t_10s,1000);
                                        }
                                }
                        }
                }
        }
}

key_search()
{
uint    retval;
        switch(keyin(ON)) {
        case STARS:
            if(errcheck()){
                retval =0x06;
            }else{
                retval =0x01;
            }
        break;

        case STOPS:
                retval =0x04;
        break;

        case FINES:
                retval =0x02;
        break;

        case ( STOPS | STARS ):
        retval =0;
        break;

        case ( STARS | FINES ):
                retval =0x05;
        break;
```

2

MAI.C

```
        case ( STOPS | FINES ):
                retval =0x05;
        break;

        case ( STOPS | FINES | STARS ):
                retval =0x05;
        break;

        default:
                retval =0;
        break;
        }
return(retval);
}

fahren(ushort ctrl)
{
        switch(ctrl){
        case 0x00:
                        /*NOP**/
        break;
        case 0x01:
                        start(1);
                        ring_clear();
        break;
        case 0x02:
                        setfine(ON);
        break;
        case 0x04:
                        stop();
        break;
        case 0x05:
                        stop_2();
        break;
        case 0x06:
                        pepo(1);
        break;
        case 0x08:
                        g3_test();
        break;
        case 0x10:
                        start(0);
                        ring_clear();
        break;
        default:
                        pepo(1); pepo(1); pepo(1); pepo(1);
        break;
        }
}

/*************************************************************
 *      電源投入直後の処理                                   *
 *************************************************************/
extern uchar    RFIF[];
startup()
{

    _disable(); /*DI**/


        task00();
        memini();
        regini();
        pla_ini();

        led(POWERL,ON);
        rtcini();

        task03_on();
```

3

保守／廃止

MAI.C

```
    _enable();  /*EI**/
        mdm_init();

        memory_switch.ecm_sw = 0;

        knight_rider();
        RNGM=RNGCM=2;
        set_TELNO();
        SPEED=0x96;

        beep(666/3);
        monitor(OFF);

}
/*************************************************************
 *      スタートＳＷ処理ルーチン                             *
 *************************************************************/
start(ushort ctrl)
{
    if(ctrl != 0){
        if( irepon(COUPS)
         || irepon(HOOKS) ){
                fax(OFF);
                FMODE=0;
                beep(1000);
        }
        else{
                startcopy();
                pepo(1);
        }
    }else{
                fax(ON);
                FMODE=0;
                beep(1000);
    }
}
start_2()
{
                startcopy_2();
                pepo(1);
}

stop_2()
{
                stop_print_2();
                pepo(1);
}

/*************************************************************
 *      ストップＳＷ処理ルーチン                             *
 *************************************************************/
stop()
{
        if(errcheck()) led(ERROL,OFF);

        if(!irepon(GENCS)) endfeed();
}
```

4

保守／廃止

MAI.C

```
/*********************************************************
 *      コピー処理ルーチン                               *
 *********************************************************/
startcopy()
{
        if(!irepon(PAPES) || irepon(GENCS)) {
                pepo(1);
                return(-1);
        }

        winker(STARL,OFF,OFF);
        led(STARL|FINEL,ON);
        FMODE=0x01;

        copy();

        winker(FINEL,OFF);

        led(STARL|FINEL,OFF);
        FMODE=0x00;
}

startcopy_2()
{
        if(!irepon(PAPES) || irepon(GENCS)) {
                pepo(1);
                return(-1);
        }

        winker(STARL,OFF,OFF);
        led(STARL|FINEL,ON);
        FMODE=0x01;

        sramtest();

        winker(FINEL,OFF);

        led(STARL|FINEL,OFF);
        FMODE=0x00;
}

/*********************************************************
 *      ファックス処理ルーチン                           *
 *********************************************************/
fax(sw)
uchar sw;
{
ushort  rest;

        if(sw == ON) t30(ON);
        else if(sw == OFF) t30(OFF);

        wait(100);

        while( task01_on_11() == 0 );

        if(rrk_genco == ON){
                task02_on_21();
        }

        while(irepon(HOOKS)) {
                pepo(1);
                if(irepon(STOPS)) break;
        }

}


/*********************************************************
```

5

保守／廃止

MAI.C

```
        ＰＬＡ初期出力
************************************************************/

pla_ini()
{
        PLA[0] = B10000100;
}

/************************************************************
 ************************************************************/
neruna()
{
        return(
                !irepon(GENCS) ||
                !irepon(PAPES) ||
                 irepon(COVES) ||
                 (FMODE==2) ||
                 irepon(HOOKS)
                );
}

/************************************************************
 *    ラム初期化                                            *
 ************************************************************/
extern  B_INFO *bufrtc;
ramclr()
{
ushort  i;

        for(i=0;i<16;i++) bufrtc->name[i] = 0x20;
                          bufrtc->ring[0]    =  2  ;
                          bufrtc->repo1[0] = 'N' ;
                          bufrtc->repo2[0] = 'N' ;
        for(i=0;i<20;i++) bufrtc->tel[i]  = 0x20;
                          bufrtc->speed[0]   = 0x96;
        for(i=0;i<2;i++)  bufrtc->year[i] = 0x30;
        for(i=0;i<2;i++)  bufrtc->month[i]= 0x30;
        for(i=0;i<2;i++)  bufrtc->date[i] = 0x30;
        for(i=0;i<2;i++)  bufrtc->hour[i] = 0x30;
        for(i=0;i<2;i++)  bufrtc->min[i]  = 0x30;
        for(i=0;i<35;i++) bufrtc->KANRI[i].kanjc = 0x00;

        settime();

        return(0);
}

/************************************************************
 **      ＬＥＤチェック                                     **
 ************************************************************/
knight_rider()
{
uchar           i,ii;
static  uchar   d[]={ COVEL,PAPEL,ERROL,FINEL,STARL,0 };

        for(ii=1;ii<=3;ii++){
            for(i=0;d[i]!=0;i++) {
                led(d[i],ON);
                wait(50);
                led(d[i],OFF);
                wait(1);
            }
            for(i=sizeof(d)-2;(i>= 0) && (i<=sizeof(d)-2);i--) {
                led(d[i],ON);
                wait(50);
                led(d[i],OFF);
                wait(1);
            }
        }
```

6

MAI.C

```
        led((COVEL|PAPEL|ERROL|FINEL|STARL),OFF);       wait_halt(333);
        led((COVEL|PAPEL|ERROL|FINEL|STARL),ON);        wait_halt(666);
        led((COVEL|PAPEL|ERROL|FINEL|STARL),OFF);
}
/*************************************************************
 **     電話番号の初期設定                                    **
 *************************************************************/
set_TELNO()
{
ushort  i;
        for(i=0;i<20;i++){
                TELNO[i]=' ';
        }
}


g3_test()
{
short ret,i;
uchar   param[8];

        LEV=mdm_init();

        LEV=readlev();
        MDMINI_FAR();

        led(ERROL,OFF);
        winker(STARL, ON,  1);

        while(irepon(STOPS));
        wait(666/2);

        coupler(OFF); relay(ON);


        INIT_G3T();
        MDMG3T_FAR();
        while(!irepon(STOPS));

        relay(OFF);

        winker(STARL|FINEL,OFF,OFF);
        buzzer(OFF);
        setfine(OFF);
        monitor(OFF);

        return(0);
}
/*************************************************************************************/
```

7

保守／廃止

## A.2　INT.ASM

INT.ASM

```
;*************************************************************************/
;*                                                                       **/
;*      ＃３２８　割り込み設定及び処理ルーチン                           **/
;*            Ｖ５５ＰＩレジスタファイルの初期設定                       **/
;*            Ｖ５５ＰＩ割り込みベクタの初期設定                         **/
;*            割り込みサービスルーチン                          **/
;*                                  １９９１年０７月０２日               **/
;*                                  ver. 1.00                            **/
;*                                  rev. 2.00       ９３　１月１／２**/
;*************************************************************************/
;
;
   BNK_INT     equ     1       ;割り込み応答方式
;
        INCLUDE SEGMENT.MAC             ;セグメント定義マクロ
        INCLUDE V25.MAC                 ;Ｖ２５特殊命令マクロその他定義
        INCLUDE V55PI.MAC               ;Ｖ５５マクロ
        INCLUDE V55PI_2.MAC             ;Ｖ５５マクロ
        INCLUDE V55PI_3.MAC             ;Ｖ５５マクロ
;
        INCLUDE MC329.MAC               ;＃３２９用ヘッダファイル
;
;-------------------------------------------------------------------------
        PUBLIC  _VSET                   ;割り込みベクタセットアップ
        public  _BNKSET
;
        public  WAKEUP
        public  ERTRAP
        public  ERTRAP2
        public  INT1MS
        public  INT10MS
        public  INT_48
;
        PUBLIC  INTWDT
        public  INT1MR
        public  INTPRN
;
        public  INTMDM
;
        public  _VSET_NMI
;
        EXTRN   TCFM_INT:far
        EXTRN   STDM_INT:far
extrn   _task01:FAR
extrn   _task02:FAR
extrn   _task03:FAR
;
;-------------------------------------------------------
        @DEFSEG
;-------------------------------------------------------

;-------------------------------------------------------
        @BSEG
;-------------------------------------------------------
;
comm near _TIME1:       Dword:1 ;１ｍｓｅｃアップカウンタ，Ｍａｘ　４５ｄａｙｓ
comm near _TIME2:       Word:1  ;１ｍｓｅｃダウンカウンタ，Ｍａｘ　６５ｓｅｃ
COMM NEAR _TT128:       BYTE:1  ;フィード機能用のタイマ

COMM NEAR _TM128:       BYTE:1  ;１２８ｍｓカウンタ
COMM NEAR _TMTSK:       BYTE:1  ;ＴＳＫカウンタ
COMM NEAR _TBIC_ENA:    BYTE:1  ;ＴＢＩＣ制御
COMM NEAR _10MS_ENA:    BYTE:1  ;１０ＭＳ制御

comm near _T1:  byte:1  ;Ｔ１タイマ
comm near _T2:  byte:1  ;Ｔ２タイマ
comm near _T3:  byte:1  ;Ｔ３タイマ
comm near _T4:  byte:1  ;Ｔ４タイマ
comm near _T5:  byte:1  ;Ｔ５タイマ
```

1

保守／廃止

INT.ASM

```
comm near _KT:  Word:1  ;通信用タイムカウンタ
;
comm near _MSTATE:      byte:1  ;モーターはステップする？
comm near _MTR_S:       byte:1  ;モーターの状態
comm near _MTIME:       byte:1  ;モーターの待ち時間
comm near _MTIME_C:     byte:1  ;モーターの待ち時間定数
comm near _WINKF:       byte:1  ;ファインＬＥＤウインカーフラグ
comm near _WINK:        byte:1  ;スタート　〃
comm near _BZSW:        byte:1  ;ブザーをならす？
comm near _STP:         byte:1  ;ストップチェックフラグ
comm near _DANGER:      byte:1  ;cover,papes
comm near _DANG_X:      byte:1  ;
comm near T128CNT2:     byte:1  ;128mS *2
comm near T128CNT3:     byte:1  ;
comm near T128CNT4:     byte:1  ;128mS *4 カウンタ
comm near T128CNT5:     byte:1  ;
comm near T128CNT6:     byte:1  ;128mS *6
comm near T128CNT:      byte:1  ;1000msカウンタ
comm near _WINKP:       byte:1  ;
comm near _PW200MS:     byte:1  ;
comm near _W200MS:      byte:1  ;TIME OF INTERVAL
comm near _PW200MS:     byte:1  ;TIME OF INTERVAL
comm near _MKEIKA:      Word:1  ;
;
comm near _V25R_PMT:    byte:1  ;
;
comm near STARL_MM:     byte:1  ;ＳＴＡＲＬメモ
;
COMM NEAR _SW_MEMO:     byte:1  ;スイッチメモ
;
extrn   _v55reg:dword
extrn   _PLA:dword
EXTRN   _nama_data:DWORD
;
COMM NEAR _endf_DMA0:   byte:1  ;ＤＭＡ０
;
COMM NEAR _SVSP:        WORD:1
COMM NEAR _SVBP:        WORD:1
COMM NEAR TMP_S:        BYTE:1
COMM NEAR _TMPBUF:      BYTE:16
;
;-------------------------------------------------------
        @ENDBSEG
;-------------------------------------------------------

;-------------------------------------------------------
        @CSEG
;-------------------------------------------------------
;************************************************************************
;************************************************************************
        public  WAKEUP
WAKEUP  PROC    FAR
;
        IRET
;
WAKEUP  ENDP
;
;
;
;*********************************************************
;*      割り込みベクタセットアップルーチン               *
_VSET   PROC    FAR
;
        @ENTASM
        @PUSH_I
;
        PUSH    DS
        PUSH    ES
;
```

2

INT.ASM

```
        MOV     AX,SEG VTBL             ;VECTOR 00000H-00100H
        MOV     DS,AX
        MOV     AX,0
        MOV     ES,AX
        CLD                             ;DIRECTION SI+=2,DI+=2

        MOV     SI,OFFSET VTBL+0*4      ; VECTOR-0
        MOV     DI,0*4                  ; DIV-ERR
        MOV     CX,2
        REP     MOVSW

        MOV     SI,OFFSET VTBL+1*4      ; VECTOR-1
        MOV     DI,1*4                  ; Single Step
        MOV     CX,2
        REP     MOVSW

        MOV     SI,OFFSET VTBL+4*4      ; VECTOR-4･･･48
        MOV     DI,4*4                  ;
        MOV     CX,(OFFSET VTBLE-OFFSET VTBL)/2-8       ; for partner
        REP     MOVSW

    _VSET_NMI   label   near
        push    es
        mov     ax,0FFFFh
        mov     es,ax
        mov     ax,es:word ptr [1]
        mov     bx,es:word ptr [3]
        pop     es
        cmp     bx,0E000h
    jnz _VSET_NMI_END
        cmp     ax, 0000h
    jnz _VSET_NMI_END

        MOV     SI,OFFSET VTBL+2*4      ; VECTOR-2
        MOV     DI,2*4                  ; NMI
        MOV     CX,2
        REP     MOVSW

        MOV     SI,OFFSET VTBL+3*4      ; VECTOR-3
        MOV     DI,3*4                  ; int 3 (brk 3)
        MOV     CX,2
        REP     MOVSW
    _VSET_NMI_END:

        mov     DI,49*4                 ; VECTOR-49･･･100h
        mov     cx,100h-49
    @@:                                 ;ベクタ４９から２５５までの初期化
        mov     word ptr es:[DI],offset ERTRAP
        mov     word ptr es:[DI+2],seg ERTRAP
        add     DI,4
        loop    @B
;
        POP     ES
        POP     DS
;
        mov     byte ptr _TBIC_ENA,1
;
        @POP_I
        @RETASM
;
;
VTBL    DW      OFFSET ERTRAP,SEG ERTRAP        ;No.0  dev_err
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.1  sgl_step
        DW      OFFSET WAKEUP,SEG WAKEUP        ;No.2  NMI,     WAKE-UP CALLING
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.3  brk-3
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.4  brkv
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.5  chkind
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.6  intIO
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.7  fpo
```

3

INT.ASM

```
        DW      OFFSET INTWDT,SEG INTWDT        ;No.8  INTWDT
        DW      000Dh,0000h                     ;No.9  INTP0
        DW      000Ch,0000h     ;INTmdm by BANK ;No.10 INTP1
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.11 INTP2
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.12 INTP3
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.13 INTP4
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.14 INTP5
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.15 system_reserved
        DW      OFFSET RTPTIM,SEG RTPTIM        ;No.16 INTCM00 on Vector
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.17 INTCM01
        DW      000Dh,0000h     ;INT1MS by BANK ;No.18 INTCM10
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.19 INTCM11
        DW      000Eh,0000h     ;INT10MS by BANK        ;No.20 INTCM21
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.21 INTCM31
        DW      OFFSET INTIMR,SEG INTIMR        ;INTIMR by VECTOR;No.22 INTD0
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.23 INTD0S
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.24 INTD1
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.25 INTD1S
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.26 INTSER0
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.27 INTSER1
        DW      OFFSET INTPRN,SEG INTPRN        ;No.28 INTCSI0/INTSR0
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.29 INTCSI1/INTSR1
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.30 INTST0
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.31 INTST1
        DW      OFFSET INT500M,SEG INT500M      ;No.32 INTSIT
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.33 system_reserved
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.34 system_reserved
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.35 system_reserved
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.36 INTPAI
        DW      OFFSET ERTRAP2,SEG ERTRAP2      ;No.37 INTAD
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.38 system_reserved
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.39 system_reserved
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.40 system_reserved
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.41 system_reserved
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.42 system_reserved
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.43 system_reserved
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.44 system_reserved
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.45 system_reserved
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.46 system_reserved
        DW      OFFSET ERTRAP,SEG ERTRAP        ;No.47 system_reserved
;
        DW      OFFSET INT_48,SEG INT_48        ;No.48 SOFT INT #48
VTBLE:
;
_VSET   ENDP

;*****************************************************************************
;*****************************************************************************
_BNKSET PROC    FAR
;
        @ENTASM
;
;
        $1      equ     OFFSET INTMDM
        $2      equu    SEG INTMDM
        MOV_IRAM_IMMW   REGB_12+REGB_vect_PC,$1
        MOV_IRAM_IMMW   REGB_12+REGB_PS,$2
        MOV_IRAM_IMMW   REGB_12+REGB_DS,DGROUP
;
        $1      equ     OFFSET INT1MS
        $3      equ     ASM_TEXT
        MOV_IRAM_IMMW   REGB_13+REGB_vect_PC,$1
        MOV_IRAM_IMMW   REGB_13+REGB_PS,$3
        MOV_IRAM_IMMW   REGB_13+REGB_DS,DGROUP
;
        $1      equ     OFFSET INT10MS
        $3      equ     ASM_TEXT
        MOV_IRAM_IMMW   REGB_14+REGB_vect_PC,$1
        MOV_IRAM_IMMW   REGB_14+REGB_PS,$3
```

4

保守／廃止

INT.ASM

```
        MOV_IRAM_IMMW   REGB_14+REGB_DS,DGROUP
;
        $1      equ     OFFSET TXimgrd
        $3      equ     ASM_TEXT
        MOV_IRAM_IMMW   REGB_07+REGB_PC_esc,$1
        MOV_IRAM_IMMW   REGB_07+REGB_PS,$3
        MOV_IRAM_IMMW   REGB_07+REGB_DS,DGROUP
;
        @RETASM
;
_BNKSET ENDP


;////////////////////////////////////////////////////////
;////////////////////////////////////////////////////////
;
;********************************************************
;********************************************************
ERTRAP  PROC    FAR
;
        IRET
;
ERTRAP  ENDP

;       for debug

ERTRAP2 PROC    FAR
;
        nop
        @FINT
        IRET
;
ERTRAP2 ENDP
;
;
;********************************************************
;*      タイマユニット1                                 *
;********************************************************
INT1MS  PROC    FAR
;
        movspa
;
        les     bx,DWORD PTR _PLA
        or      BYTE PTR es:[bx],20h
;
        MOV     AX,SEG DGROUP
        MOV     DS,AX
;
        CALL    MS1
;
        inc     byte ptr _TM128
        test    byte ptr _TM128,01111111b
        jnz     @F
        call    TMR128
    @@:
;
        and     byte ptr _TMTSK,00001111b
        dec     byte ptr _TMTSK
        jnz     @F
        mov     byte ptr _TMTSK,10
    @@:
;
        les     bx,DWORD PTR _PLA
        and     BYTE PTR es:[bx],not 20h
;
        @FINT
        retrbi
;
        nop
```

5

保守／廃止

INT.ASM

```
INT1MS  ENDP
;
;
;*********************************************************
;*********************************************************
        public  INT10MS
INT10MS PROC    FAR
;
        movspa
        MOV     AX,SEG DGROUP
        MOV     DS,AX
;
        RSTWDT
;
        les     bx,DWORD PTR _PLA
        or      BYTE PTR es:[bx],10h
;
;



        sti
        call    FAR PTR _task01
        call    FAR PTR _task02
        call    FAR PTR _task03
        cli
;
INT10MS_pass:
;
        les     bx,DWORD PTR _PLA
        and     BYTE PTR es:[bx],not 10h
;
        @FINT
        retrbi
;
INT10MS ENDP
;
;
;【ＩＮＴ　４８】
INT_48  PROC    FAR
        @PUSHR
        MOV     AX,SEG DGROUP
        MOV     DS,AX
;
        les     bx,DWORD PTR _PLA
        or      BYTE PTR es:[bx],10h
        and     BYTE PTR es:[bx],not 10h
;
        sti
        call    FAR PTR _task01
        call    FAR PTR _task02
        call    FAR PTR _task03
        cli
;
        @POPR
        IRET
        nop
INT_48  ENDP
;*********************************************************
;*********************************************************
        public  TMR128
TMR128  PROC    near
;
        INC     BYTE PTR T128CNT
;
        CALL    MS128
;
        test    BYTE PTR T128CNT,00000001b
        JNZ     @F
        CALL    MS256
```

6

INT.ASM

```
        jmp     TMR128_2
    @@:
;
        test    BYTE PTR T128CNT,00000010b
        JNZ     @F
        CALL    MS512
        jmp     TMR128_2
    @@:
;
        test    BYTE PTR T128CNT,00000100b
        JNZ     @F
        CALL    SEC1
        jmp     TMR128_2
    @@:
;
        test    BYTE PTR T128CNT,00001000b
        JNZ     @F
        CALL    SEC2
        jmp     TMR128_2
    @@:
;
TMR128_2:
        ret
;
TMR128  ENDP

;---------------------------------------
;---------------------------------------
;
        public  MS1
MS1     PROC    near
;
        ;タイマ機能ドライブ
        RSTWDT
        CMP     WORD PTR _TIME2,0
        JE      @F
        DEC     WORD PTR _TIME2
    @@:

        add     WORD PTR _TIME1,1
        adc     WORD PTR _TIME1+2,0

        add     BYTE PTR _MTIME,1
        sbb     BYTE PTR _MTIME,0
;
;
    @@:
;
        RET
        nop
;
MS1     ENDP
;
;
;
;
;
;
;
;
;---------------------------------------------------
;---------------------------------------------------
        public  MS128
MS128   PROC    near
;
        PUSH    AX
        PUSH    BX
;
        les     bx,DWORD PTR _v55reg
```

7

保守／廃止

INT.ASM

```
        MOV     al,ES:[BX].P1           ;P1
        and     al,78h  ;01111000b
        MOV     ah,al
        MOV     al,ES:[BX].P5           ;P5
        and     al,03h  ;00000111b
        or      ah,al
        mov     al,ah
                                        ;bit7   ----
                                        ;bit6   FINE key
                                        ;bit5   STOP key
                                        ;bit4   START key
                                        ;bit3   COPY key
                                        ;bit2   GENCO sens
                                        ;bit1   PAPER sens
                                        ;bit0   COVER sens
;
        mov     byte ptr _SW_MEMO,ah
;
;
        MOV     AL,AH
        AND     AL,STOPS
        JNZ     @F                      ;pass if it is 1.
        MOV     _STP,ON
        MOV     BYTE PTR _W200MS,0
    @@:
;
        MOV     AL,AH
        AND     AL,COVES
        JNZ     @F
        MOV     _DANGER,ON
        or      _DANG_X,00000001b
        JMP     MS128_5
    @@:
;
        MOV     AL,AH
        AND     AL,PAPES
        JNZ     @F
        MOV     _DANGER,ON
        or      _DANG_X,00000100b
        JMP     MS128_5
    @@:
;
;
;
        MOV     _DANGER,OFF
        MOV     _DANG_X,0
;
;
MS128_5:
        les     bx,DWORD PTR _v55reg
        MOV     al,ES:[BX].P0           ;P0
        mov     bx,ax
;
        CMP     BYTE PTR _WINK,0
        JZ      @F
        CMP     BYTE PTR _W200MS,0
        JNZ     @F
        xor     bl,STARL
    @@:
;
        CMP     BYTE PTR _WINKF,0
        JZ      @F
        xor     bl,FINEL
    @@:
;
        CMP     BYTE PTR _WINKP,0
        JZ      @F
        xor     bl,POWERL
    @@:
```

8

INT.ASM

```
;
        mov     ax,bx
        les     bx,DWORD PTR _v55reg
        MOV     ES:[BX].P0,al           ;P0
;
;
;
        CMP     BYTE PTR _TT128,0
        JZ      @F
        DEC     BYTE PTR _TT128
    @@:
;
        POP     BX
        POP     AX
        RET
;
MS128   ENDP
;
;
;-------------------------------------
;-------------------------------------
        public  MS256
MS256   PROC    near
;
        PUSH    AX
        PUSH    BX
;
        les     bx,DWORD PTR _v55reg
        MOV     al,ES:[BX].P0           ;P0
        mov     bx,ax
;
        CMP     BYTE PTR _WINK,0
        JZ      @F
        CMP     BYTE PTR _W200MS,1
        JNZ     @F
        xor     bl,STARL
    @@:
;
        CMP     BYTE PTR _WINKF,0
        JZ      @F
        CMP     BYTE PTR _FW200MS,1
        JZ      @F
        xor     bl,FINEL
    @@:
;
        CMP     BYTE PTR _WINKP,0
        JZ      @F
        CMP     BYTE PTR _PW200MS,1
        JZ      @F
        xor     bl,POWERL
    @@:
;
        CMP     BYTE PTR _WINK,0
        JZ      @F
        CMP     BYTE PTR _W200MS,2
        JC      @F
        or      bl,STARL
    @@:
;
        mov     ax,bx
        les     bx,DWORD PTR _v55reg
        MOV     ES:[BX].P0,al           ;P0
;
;
        POP     BX
        POP     AX
        RET
;
MS256   ENDP
```

9

INT.ASM

```
;
;
;-------------------------------------
;-------------------------------------
        public  MS512
MS512   PROC    near
;
        PUSH    AX
        PUSH    BX
;
        les     bx,DWORD PTR _v55reg
        MOV     al,ES:[BX].P0           ;P0
        mov     bx,ax
;
;
        CMP     BYTE PTR _WINKF,0
        JZ      @F
        CMP     BYTE PTR _PW200MS,2
        JNZ     @F
        xor     bl,FINEL
    @@:
;
        CMP     BYTE PTR _WINKP,0
        JZ      @F
        CMP     BYTE PTR _PW200MS,2
        JNZ     @F
        xor     bl,POWERL
    @@:
;
        mov     ax,bx
        les     bx,DWORD PTR _v55reg
        MOV     ES:[BX].P0,al           ;P0
;
        CMP     BYTE PTR _BZSW,0
        JZ      @F
        les     bx,DWORD PTR _PLA
        xor     BYTE PTR es:[bx],BUZZ
    @@:
;
        POP     BX
        POP     AX
        RET
;
MS512   ENDP
;
;
;---------------------------------------
;---------------------------------------
        public  SEC1
SEC1    PROC    near
;
        PUSH    AX
        PUSH    BX
;
        les     bx,DWORD PTR _v55reg
        MOV     al,ES:[BX].P0           ;P0
        mov     bx,ax
;
        CMP     BYTE PTR _WINK,0
        JZ      @F
        CMP     BYTE PTR _W200MS,2
        JNZ     @F
        xor     bl,STARL
    @@:
;
        mov     ax,bx
        les     bx,DWORD PTR _v55reg
        MOV     ES:[BX].P0,al           ;P0
;
```

保守／廃止

INT.ASM

```
;
;
        CMP     BYTE PTR _T1,0
        JZ      @F
        DEC     BYTE PTR _T1
    @@:
;
        CMP     BYTE PTR _T2,0
        JZ      @F
        DEC     BYTE PTR _T2
    @@:
;
        CMP     BYTE PTR _T3,0
        JZ      @F
        DEC     BYTE PTR _T3
    @@:
;
        CMP     BYTE PTR _T4,0
        JZ      @F
        DEC     BYTE PTR _T4
    @@:
;
        CMP     BYTE PTR _T5,0
        JZ      @F
        DEC     BYTE PTR _T5
    @@:
;
        MOV     AX,_KT
        INC     AX
        MOV     _KT,AX
;
        POP     BX
        POP     AX
        RET
;
SEC1    ENDP
;
;
;----------------------------------------
;----------------------------------------
        public  SEC2
SEC2    PROC    near
;
        PUSH    AX
        PUSH    BX
;
        les     bx,DWORD PTR _v55reg
        MOV     al,ES:[BX].P0           ;P0
        mov     bx,ax
;
        CMP     BYTE PTR _WINK,0
        JZ      @F
        CMP     BYTE PTR _W200MS,3
        JNZ     @F
        and     bl,not STARL
    @@:
;
        mov     ax,bx
        les     bx,DWORD PTR _v55reg
        MOV     ES:[BX].P0,al           ;P0
;
        POP     BX
        POP     AX
        RET
;
SEC2    ENDP
;
```

11

保守／廃止

INT.ASM

```
;*******************************************************************************
;*******************************************************************************
INTIMR  PROC    FAR
;
        push    bx
        push    es

        les     bx,DWORD PTR _PLA
        and     BYTE PTR es:[bx],not 01h
;
        les     bx,DWORD PTR _v55reg
        mov     BYTE PTR [BX].DMAM0,081H
;
        or      _endf_DMA0,0ffh
;
        pop     es
        pop     bx
;
        @FINT
        IRET
INTIMR  ENDP
;
;*******************************************************************************
;*******************************************************************************
INTMDM  PROC    FAR
        movspa
;
        mov     ax,8000h
        mov     es,ax
        mov     al,es:byte ptr [0010h]
;
        @FINT                   ;内部割り込みコントローラ復帰
        retrbi
INTMDM  ENDP
;
;*******************************************************************************
;*******************************************************************************
INTWDT  PROC    FAR
        nop
        @FINT
        IRET
INTWDT  ENDP
;
;*******************************************************************************
;*******************************************************************************
INTPRN  PROC    FAR
        nop
        @FINT
        IRET
INTPRN  ENDP
;
;************************************************
;
        PUBLIC  _MTR12TBL
;
RTPTIM  PROC    FAR

        @PUSHR
;
        LES     BX,DWORD PTR _v55reg
        MOV     AL,01000011B
        MOV     ES:[BX].IC16,AL
        MOV     AL,ES:[BX].TMC0
        AND     AL,11110010B
        MOV     ES:[BX].TMC0,AL
        MOV     BYTE PTR _MTIME,0
;
        @POPR
;
```

12

INT.ASM

```
        @FINT
        IRET
;;;
_MTR12TBL:      DB      066h,022h
                DB      0AAh,088h
                DB      099h,011h
                DB      055h,044h
;;;
RTPTIM  ENDP
;
;***********************************************
;***********************************************
;
        PUBLIC  INT500M
INT500M PROC    FAR
;
;
        @PUSHR
;
        LES     BX,DWORD PTR _v55reg
        XOR     AH,AH
        MOV     AL,ES:[BX].ADCR0
        XOR     BH,BH
        MOV     BL,TMP_S
        MOV     DS:BYTE PTR _TMPBUF[BX],AL
        INC     BL
        AND     BL,00001111B
        MOV     BYTE PTR TMP_S,BL
;
        @POPR
        @FINT
        IRET
;
INT500M ENDP
;
;
;-------------------------------------------------------
;-------------------------------------------------------
public  _CHGTASK
_CHGTASK        PROC    FAR
        PUSH    AX
;
        XOR     AX,AX
        MOV     AL,07H
        MOVSPB_AX
        TSKSW_AX
;
        POP     AX
        RET
_CHGTASK        ENDP
;
;-------------------------------------------------------
;-------------------------------------------------------
        PUBLIC  TXimgrd
;
TXimgrd PROC    FAR
;
        XOR     AX,AX
        MOV     AL,0FH
        MOVSPB_AX
        TSKSW_AX
        jmp     TXimgrd
;
TXimgrd ENDP
;
;
;-------------------------------------------------------
        @ENDCSEG
;-------------------------------------------------------
```

13

INT.ASM

```
        END
```

14

## A.3　REG55.C

REG55.C

```
/*************************************************************************/
/*                                                                       **/
/*      Ｖ５５ＰＩ特殊機能レジスタ領域の初期設定                        **/
/*                                                                       **/
/*************************************************************************/

    #define     BNK_INT 1       /*割り込み応答方式**/

/*      Ｃ言語基本設定  **/
#include        "BASICDEF.H"

/*      Ｖ５５ＰＩの構造体ファイル      **/
#include        "INDAT55.H"

/*      Ｖ５５ＰＩの構造体とベースアドレスを設定**/
struct  v55pi_regs far  *v55reg =(struct v55pi_regs far *)base;
                                /* base = 0xff000e00 **/

/*これを呼ぶときはＤＩ状態であること**/
regini()        /*次のポート入出力設定**/
{
uchar   time_mem;
ushort  tada;

        v55reg->STBC    = B00000001;

        v55reg->PRC     = B11100101;
        v55reg->PRC     = B11100100;

        v55reg->MBC     = B11010110;
        v55reg->PWC1    = B10010110;
        v55reg->PWC0    = B00000111;
        v55reg->RFM     = B00110000;
        v55reg->IC09    = B01000011;
        v55reg->IC10    = B01000011;
        v55reg->IC11    = B01000011;
        v55reg->IC12    = B01000011;
        v55reg->IC13    = B01000011;
        v55reg->IC14    = B01000011;
        v55reg->IC16    = B01000011;
        v55reg->IC17    = B01000011;
        v55reg->IC18    = B01000011;
        v55reg->IC19    = B01000011;
        v55reg->IC20    = B01000011;
        v55reg->IC21    = B01000011;
        v55reg->IC22    = B01000011;
        v55reg->IC23    = B01000011;
        v55reg->IC24    = B01000011;
        v55reg->IC25    = B01000011;
        v55reg->IC26    = B01000011;
        v55reg->IC27    = B01000011;
        v55reg->IC28    = B01000011;
        v55reg->IC29    = B01000011;
        v55reg->IC30    = B01000011;
        v55reg->IC31    = B01000011;
        v55reg->IC32    = B01000011;
        v55reg->IC36    = B01000011;
        v55reg->IC37    = B01000011;

        v55reg->STMC    = 0xb71b;

        for( tada = 0xffff; tada != 0; tada-- );


        v55reg->PRDC    = B00000000;
        v55reg->P0      = B00000000;
        v55reg->PM0     = B00000000;
        v55reg->PMC2    = B00100000;
        v55reg->P2      = B11100111;
```

1

保守／廃止

REG55.C

```
        v55reg->PM2     = B11100011;
        v55reg->PMC3    = B01110101;
        v55reg->P3      = B11110101;
        v55reg->PM3     = B11110101;
        v55reg->PMC4    = B00000000;
        v55reg->P4      = B00110111;
        v55reg->PM4     = B00000111;
        v55reg->PMC5    = B00000000;
        v55reg->PM5     = B11111111;
        v55reg->PMC7    = B11110000;
        v55reg->P7      = B00001000;
        v55reg->PM7     = B11111000;

        v55reg->PMC8    = B00000001;
        v55reg->PM8     = B11111111;
        v55reg->P7H     = B00000000;
        v55reg->RTPC    = B01000000;
        v55reg->RTPD    = 0;
        v55reg->ADM     = B10000001;




        v55reg->ASP     = B00000010;
        v55reg->TxBRG0  = 0;
        v55reg->PRS0    = 0;
        v55reg->UARTM0_CSIM0    = B10000101;
        v55reg->TxBRG1  = 155;
        v55reg->RxBRG1  = 155;
        v55reg->PRS1    = B00010010;
        v55reg->UARTM1_CSIM1    = B11001100;

        v55reg->TOC0    = 0xff;
        v55reg->TOC1    = 0x03;
        v55reg->TMC0    = B00000000;
        v55reg->TMC1    = B00000000;

        time_mem = B00001111;


    if( time_mem & B00001000 ){
        v55reg->CM31    = 1200;
        v55reg->CM30    = v55reg->CM31-(v55reg->CM31/8);
        v55reg->TMC1    |= B10000000;
    }
    if( time_mem & B00000100 ){
        v55reg->CM21    = 0x3A98;
        v55reg->TMC1    |= B00001000;
    }

    if( time_mem & B00000010 ){
        v55reg->CM10    = 0x5DC;
        v55reg->CM11    = v55reg->CM10 >> 1;
        v55reg->TMC0    |= B10000000;
    }

    if( time_mem & B00000001 ){
        v55reg->CM00    = 0x1149;
        v55reg->CM01    = v55reg->CM00 >> 1;
    }

        v55reg->DMAM0   = B10000001;
        v55reg->DMAC0   = B00000101;
        v55reg->+TCM0   = 0x100L-1;
        v55reg->TC0     = v55reg->TCM0;

        v55reg->IMC     = B10000000;
        v55reg->INTM0   = B00000101;
```

2

保守／廃止

REG55.C

```
        v55reg->INTM1   = B00000000;
#define msk_int_EX0     B01000000;
#define msk_int_MDM     B00000000;
#define msk_int_1MS     B00000000;
#define msk_int_10M     B01000000;
#define msk_int_10M2    B00000000;
#define msk_int_DMA     B01000000;

        v55reg->IC09    = B00010001 | msk_int_EX0;
        v55reg->IC10    = B00010000 | msk_int_MDM;
        v55reg->IC11    = B01000011;
        v55reg->IC12    = B01000011;
        v55reg->IC13    = B01000011;
        v55reg->IC14    = B01000011;
        v55reg->IC16    = B01000011;
        v55reg->IC17    = B01000011;
        v55reg->IC18    = B00010001 | msk_int_1MS;
        v55reg->IC19    = B01000011;
        v55reg->IC20    = B00010011 | msk_int_10M2;
        v55reg->IC21    = B01000011;
        v55reg->IC22    = B00000001 | msk_int_DMA;
        v55reg->IC23    = B01000011;
        v55reg->IC24    = B01000011;
        v55reg->IC25    = B01000011;
        v55reg->IC26    = B01000011;
        v55reg->IC27    = B01000011;
        v55reg->IC28    = B01000011;
        v55reg->IC29    = B01000011;
        v55reg->IC30    = B01000011;
        v55reg->IC31    = B01000011;
        v55reg->IC32    = B00000011;

}
/*****************************************************************************/
```

3

## A.4　MEM.C

MEM.C

```
/*****************************************************************************/
/*                                                                           **/
/*      メモリの枠及び内容の初期設定                                         **/
/*                                                                           **/
/*****************************************************************************/

/*      C言語基本設定  **/
#include        "BASICDEF.H"

/*      #３２８ファイル+**/
#include        "hd329.h"

        struct  nama_data_s far *nama_data
                = (struct nama_data_s far *)SRAM_20;
        struct  comp_data_s far *comp_data
                = (struct comp_data_s far *)SRAM_21;
        struct  trnp_tabl_s far *trnp_tabl_A
                = (struct trnp_tabl_s far *)SRAM_22;
        struct  trnp_tabl_s far *trnp_tabl_B
                = (struct trnp_tabl_s far *)SRAM_23;
        struct  comp_ecm_data_s far *s_data1
                = (struct comp_ecm_data_s far *)PSRAM1;
        struct  comp_ecm_data_s far *s_data2
                = (struct comp_ecm_data_s far *)PSRAM2;

        uchar far* Modem =(uchar far*)MMIO_Modem;
        uchar far* IOCE1 =(uchar far*)MMIO_CE1;
        uchar far* RTC   =(uchar far*)MMIO_RTC;
        uchar far* PLA   =(uchar far*)MMIO_PLA;

/*********************************************************************/
/**      メモリのイニシャライズ                                     **/
/*********************************************************************/
memini()
{
        VSET();
        BNKSET();

        PLA[0] &= B11111110;


}
```

1

## A.5　BRW.C

BRW.C

```
/*********************************************************
 *      ＃３２８サーマル／ｃｌｓドライブ用              *
 *********************************************************/

/*      Ｃ言語基本設定  **/
#include        "BASICDEF.H"
/*      ＃３２９ファイル**/
#include        "HD329.H"
/*      Ｖ５５ＰＩ内部データ領域・マップ **/
#include        "INDAT55.H"
/*      Ｖ５５ＰＩの構造体とベースアドレスを設定**/
extern  struct  v55pi_regs far  *v55reg;
/*      ＰＬＡ入出力アドレス    **/
extern  uchar far* PLA;
/*      外部ＤＭＡ入出力アドレス**/
        uchar far* EX_DMA = (uchar far*)0x80001000;

extern  struct  nama_data_s far *nama_data;     /**画像生データ**/
        uchar   DEST[ LENGTH+5 ];

extern  unsigned char   endf_DMA0;


uchar far* imgread_c(linen,size)
uchar   linen;
ushort  size;
{
uchar far*      ptr;

        img_input(linen % 32,size % 0x101);
        ptr = (uchar far*)&(nama_data->arry[(linen % 32)][0]);
        return(ptr);
}

uchar far* imgread_1(linen,size)
uchar   linen;
ushort  size;
{
uchar far*      ptr;

        img_input1(linen % 32,size % 0x101);
        ptr = (uchar far*)&(nama_data->arry[(linen % 32)][0]);
        return(ptr);
}




img_input(uchar linen,ushort size)
{
register        reg1,reg2;
ulong   end_addr;


        v55reg->IC22    = B01000001;

        end_addr = (ulong)&nama_data->arry[linen % 32][(size-1) % 0x100];
        PLA[0] &= B11111110;
        v55reg->DMAM0   = B10000001;
        v55reg->DMAC0   = B00000100;
        v55reg->MAR0    = ( 0x000ffff0 & (end_addr>>12) )
                                + ( 0x0000ffff &  end_addr);
        v55reg->TC0     = (size-1) % 0x100;

        v55reg->TCM0    = 0x00006666;

        v55reg->IC22    &= B01111111;
        v55reg->DMAS    = B00000001;
```

1

保守／廃止

BRW.C

```
        v55reg->IC22    &= B10111111;

        endf_DMA0 = 0;
        v55reg->DMAM0   = B10001001;
        PLA[0] |= B00000001;

        while( endf_DMA0 == 0 );


}


img_input1(uchar linen,ushort size)
{
register        reg1,reg2;
ulong   end_addr;

        v55reg->IC22    = B01000001;
        end_addr = (ulong)&nama_data->arry[linen % 32][(size-1) % 0x100];
        PLA[0] &= B11111110;
        v55reg->DMAM0   = B10000001;
        v55reg->DMAC0   = B00000100;
        v55reg->MAR0    = ( 0x000ffff0 & (end_addr>>12) )
                                + ( 0x0000ffff &  end_addr);
        v55reg->TC0     = (size-1) % 0x100;
        v55reg->TCM0    = 0x00006666;
        v55reg->IC22    &= B01111111;
        v55reg->DMAS    = B00000001;
        v55reg->IC22    &= B10111111;
        endf_DMA0 = 0;
        v55reg->DMAM0   = B10001001;
        PLA[0] |= B00000001;

}

/*********************************************************
 *      密着ＬＥＤ電源コントロール                       *
 *      in...   uchar   ON/OFF                  *
 *      out..   無し                                     *
 *********************************************************/
cisp(sw)
uchar   sw;
{
        if( sw == ON){
                v55reg->P2 |= CIS_P;
        }else{
                v55reg->P2 &= (uchar)~CIS_P;
        }
}

/*********************************************************
 *      サーマル電源コントロール                 *
 *      in...   uchar   ON/OFF                           *
 *      out..   無し                                     *
 *********************************************************/
thermalp(sw)
uchar   sw;
{
        if(sw == ON){
                v55reg->P2 |= THM_P;
        }else{
                v55reg->P2 &= (uchar)~THM_P;
        }
}

/*****************************************************************************
*****************************************************************************/

w_ther()
```

2

BRW.C

```
{
uchar   i;

        for(i=0;i<214;i++){
                for(;i<2;i++)v55reg->TxB0_SIO0 = 0x00;
                v55reg->TxB0_SIO0 = nama_data->arry[0][i];
        }
        for(;i<216;i++)v55reg->TxB0_SIO0 = 0x00;

        ltc_str();
}



ltc()
{
                v55reg->P4 |= TH_LTC;
                v55reg->P4 &= (uchar)~TH_LTC;
                        PLA[0] |= TH_LTC;
                        PLA[0] &= (uchar)~TH_LTC;
}

ltc_str()
{
while( !((uchar)TH_END & v55reg->P8));

        if(1){
                v55reg->P4 |= TH_LTC;
                v55reg->P4 &= (uchar)~TH_LTC;
                        PLA[0] |= TH_LTC;
                        PLA[0] &= (uchar)~TH_LTC;
                PLA[0] |= TH_STB;
                PLA[0] &= (uchar)~TH_STB;
return(ON);
        }else{
return(OFF);
        }
}

ltc_str_2()
{
if( !((uchar)TH_END & v55reg->P8)){
return(OFF);
}
        if(1){
                v55reg->P4 |= TH_LTC;
                v55reg->P4 &= (uchar)~TH_LTC;
                        PLA[0] |= TH_LTC;
                        PLA[0] &= (uchar)~TH_LTC;
                PLA[0] |= TH_STB;
                PLA[0] &= (uchar)~TH_STB;
return(ON);
        }else{
return(OFF);
        }
}

str()
{

        if(1){
                PLA[0] |= TH_STB;
                PLA[0] &= (uchar)~TH_STB;
                }
}

str_2()
{
        if( !((uchar)TH_END & v55reg->P8)){
```

3

BRW.C

```
                return(OFF);
        }
        if(1){
                PLA[0] |= TH_STB;
                PLA[0] &= (uchar)~TH_STB;
                return(ON);
        }else{
                return(OFF);
        }
}

str_end()
{
        if( !((uchar)TH_END & v55reg->P8))      return(0);
        else                    return(1);
}




allptn()
{
}

chprt()
{
}


chloop(i)
uchar   i;
{
}


/*************************************************************************
*************************************************************************/

chth()
{
uchar   i;

        for(i=0;i<(LENGTH-1);i++)
                {
                        v55reg->TxB0_SIO0 = DEST[i];


                }
}
/*******************************************
        サーマルヘッドのクリアー
*******************************************/
clrthermal_0()
{
static  uchar   i;
        for(i=0;i<(LENGTH-1);i++){
                        v55reg->TxB0_SIO0 = 0x00;
        }
        ltc();
}

clrthermal()
{
static  uchar   i;
        for(i=0;i<(LENGTH-1);i++){
                        v55reg->TxB0_SIO0 = 0x00;
        }
        ltc_str();
```

4

保守／廃止

BRW.C

```
}

/*********************************************
        特殊なパターン
*********************************************/
void near line_nop()
{}
line_dotted(uchar ctrl)
{
register        i;
        for(i=0; i<(LENGTH/2); i++){
                v55reg->TxB0_SIO0 = B10101010;
        }
        for(    ; i< LENGTH    ; i++){
                v55reg->TxB0_SIO0 =         B01010101;
        }
        if(ctrl!=0)ltc_str();
}
line_dash(uchar ctrl)
{
register        i;
        for(i=0; i<LENGTH; i++){
                v55reg->TxB0_SIO0 = B10011001;
        }
        if(ctrl!=0)ltc_str();
}
line_dash2(uchar ctrl)
{
register        i;
        for(i=0; i<(LENGTH/2); i++){
                v55reg->TxB0_SIO0 = B11110000;
                        line_nop();
                v55reg->TxB0_SIO0 =         B00001111;
        }
        if(ctrl!=0)ltc_str();
}
line_dash4(uchar ctrl)
{
register        i;
        for(i=0; i<(LENGTH/4); i++){
                v55reg->TxB0_SIO0 = B11111111;
                        line_nop();
                v55reg->TxB0_SIO0 =         B00000000;
                        line_nop();
                v55reg->TxB0_SIO0 =                 B00000000;
                        line_nop();
                v55reg->TxB0_SIO0 =                         B11111111;
        }
        if(ctrl!=0)ltc_str();
}

/*********************************************************
 *      ページ終了のラインを引くルーチン        *
 *              in...  なし                              *
 *              out..  なし                              *
 *********************************************************/
#define CONST_line      47
ushort far      line_count_end_C = CONST_line +13;
ushort          line_count_end   = CONST_line +13;
line()
{
uchar   i;
        thermalp(ON);
                for(i=0; i<(LENGTH/2); i++){
                        v55reg->TxB0_SIO0 = B11100110;
                        line_nop();
                        v55reg->TxB0_SIO0 =         B01100111;
                }
        ltc_str();
```

BRW.C

```
        while( str_end() != 1 );

        thermalp(OFF);
        clrthermal();

        line_count_0();
}
line_count_0()
{
        line_count_end = 0;
}
line_count_plus( ushort length )
{
        line_count_end += length;
}
ushort line_count_judg()
{
    if( line_count_end >= (line_count_end_C) ){
        return(0);
    }else{
        return( (line_count_end_C) - line_count_end );
    }
}
ushort line_count_end_get()
{
        line_count_end = line_count_end_C;
return( line_count_end-5 );
}

extern  uchar   step_task[8][8];
task01_4_on()
{
        step_task[1][4]=1;
}
task01_4()
{
static  uchar   i;
    switch( step_task[1][4] ){
    case 1:
        thermalp(ON);
                for(i=0; i<(LENGTH/2); i++){
                        v55reg->TxB0_SIO0 = B01100111;
                        line_nop();
                        v55reg->TxB0_SIO0 =         B11100110;
                }
        ltc_str();
        while( str_end() != 1 );
    step_task[1][4] = 2;
    break;

    case 2:
        if( str_end() == 1 ){
        thermalp(OFF);
        clrthermal();
    step_task[1][4] = 0;
        }
    break;
    }
}

white()
{
}



print(data0,data1)
char    *data0,*data1;
{
```

6

保守／廃止

BRW.C

```
}


/*****************************************************************************/
uchar   endf_DMA0;
uchar   far *nama_de;
uchar   far *nama_deta;
uchar   bombo[256];
ushort  bomb2[256];
extern  struct  nama_data_s far *nama_data;
bon2()
{
register        reg1,reg2;
ulong   end_addr;
ushort  stk_MK0,stk_MK1;

        v55reg->CM10    = 0x96;

                endf_DMA0 = 0;
                end_addr = (ulong)&nama_data->arry[0x00][216-1];
                PLA[0] &= B11111110;
                v55reg->DMAM0   = B10000001;
                v55reg->DMAC0   = B00000100;
                v55reg->MAR0    = ( 0x000ffff0 & (end_addr>>12) )
                                + ( 0x0000ffff &  end_addr);
                v55reg->MAR0    |= 0xfff00000;
                v55reg->TC0     = 216L-1;
                v55reg->TCM0    = 0x001fffff;

        _disable();
                v55reg->IC22 &= B01111111;
                v55reg->DMAS    = B00000001;
                v55reg->DMAM0   = B10001001;
                PLA[0] |= B00000001;

        while((reg1 = v55reg->TC0) != (reg2 = v55reg->TCM0));

                PLA[0] &= B11111110;
                v55reg->DMAM0   = B10000001;
        _enable();

        wait(20);
}
/*****************************************************************/
bon4()
{
ulong   bgn_addr;

        bgn_addr = (ulong)&nama_data->arry[0x00][0x00];
        PLA[0]    &= B11111110;
        EX_DMA[0] &= B11101111;
        *(ushort far*)(&EX_DMA[2])
         =(ushort)(
                (       ( 0x000ffff0 & (bgn_addr>>12) )
                        + ( 0x0000ffff & bgn_addr)      )>>8
          );

        _disable();
        EX_DMA[0] |= B00010000;
        PLA[0]    |= B00000001;
        while( (EX_DMA[1] & B00000001) == 0 );
        PLA[0]    &= B11111110;
        EX_DMA[0] &= B11101111;
        _enable();

        wait(20);
}
```

## A.6　BMTR.C

BMTR.C

```
/*****************************************************************
*       ＃３２９                                                 *
*       モーターの基本ＢＩＯＳ          " ｂｍｔｒ．ｃ" *
*               各種駆動ルーチン                                 *
*               ＭＯＴＯＲ                                       *
*****************************************************************/

/*      Ｃ言語基本設定  **/
#include        "BASICDEF.H"
/*      ＃３２８ファイル+**/
#include        "HD329.h"
/*      Ｖ５５ＰＩの構造体ファイル      **/
#include        "INDAT55.H"

/*      Ｖ５５ＰＩの構造体とベースアドレスを設定**/
extern  struct  v55pi_regs far  *v55reg;


extern  uchar   MTR_S;          /*読取りモーターステータス**/
extern  uchar   MSTATE;
extern  uchar   MTIME;
extern  uchar   MTIME_C;
extern  ushort  MKEIKA;
extern  ushort  MKEIKA12;
extern  uchar   far MTR12TBL[];

#define M_STEP  2

/*****************************************
 *      ＭＯＴＯＲ ()                    *
 *      リアルタイム出力ポートを使用     *
 *      ベースソフトで１回目を設定       *
 *      リアルタイムポートが２発目を     *
 *      自動的に出力。                   *
 *      in..... Nothing                  *
 *      out.... Nothing                  *
 *****************************************/
motor()
{
uchar   tmp;

        while(MTIME < 5);
        _disable();
        v55reg->RTP =
                ((v55reg->RTP) & B00001111) | ((MTR12TBL[MTR_S]) & B11110000);
        MTR_S++;
        MTR_S=MTR_S & B00000111;
        v55reg->P7H = (MTR12TBL[MTR_S]);
        MTR_S++;
        MTR_S=MTR_S & B00000111;
        MTIME=0;
        v55reg->IC16    = B00000011;

        v55reg->TMC0    |= B00001000;
        _enable();

}


/*************************************************
 *      chgmvol(sw)                              *
 *      in....uchar     ON/OFF          *
 *      out...nothing                            *
 *************************************************/

chgmvol(sw)
uchar   sw;
{
        if(sw==ON){
```

保守／廃止

BMTR.C

```
        }
        else{
        }
}

/************************************************
 *      モーターの逆転ステップルーチン          *
 *              REVMOTOR        *
 *      in...   なし                            *
 *      out..   なし                            *
 ************************************************/
revmotor()
{


        switch(MTR_S){

                case 3: v55reg->P7H = 0x55;
                        MTR_S = 2;
                break;

                case 2: v55reg->P7H = 0x99;
                        MTR_S = 1;
                break;

                case 1: v55reg->P7H = 0xaa;
                        MTR_S = 0;
                break;

                case 0: v55reg->P7H = 0x66;
                        MTR_S = 3;
                break;

                default:  MTR_S = 0;
                break;
                }

}
/************************************************************
 *      モーター電源コントロール                *
 *      in...   uchar   MOT_IR/MOT_PR,ON/OFF    *
 *      out..   無し                                    *
 ************************************************************/
motorp(gme,sw)
uchar   gme,sw;
{
        switch(gme){
           case MOT_IR:
                if(sw==ON)      v55reg->P4 &= (uchar)~MOT_IR;
                else            v55reg->P4 |=        MOT_IR;
           break;

           case MOT_PR:
                if(sw==ON)      v55reg->P4 &= (uchar)~MOT_PR;
                else            v55reg->P4 |=        MOT_PR;
           break;

           case MOT_PR|MOT_IR:
           default:
                if(sw==ON)      v55reg->P4 &= (uchar)(~MOT_IR & ~MOT_PR);
                else            v55reg->P4 |=        ( MOT_IR |  MOT_PR);
           break;
        }
}
/************************************************************
 *      モーター電源コントロール，シャットアウト*
 ************************************************************/
motors_off()
{
```

2

保守／廃止

BMTR.C

```
        motorp(MOT_PR|MOT_IR,OFF);
}
/*********************************************************
 *      原稿フィード                                     *
 *      in...   uchar                                    *
 *      out..   なし                                     *
 *********************************************************/
feed()
{
        motorp(MOT_IR,ON);

        feedvarlen(MOT_IR,14);

        motorp(MOT_IR,OFF);
}
feed_2()
{
        motorp(MOT_IR,ON);

        feedvarlen(MOT_IR,2);

        motorp(MOT_IR,OFF);
}

/*********************************************************
 *      原稿排出フィード                                 *
 *      in...   uchar                                    *
 *      out..   なし                                     *
 *********************************************************/
endfeed()
{
ushort  i,j;

        motorp(MOT_IR,ON);

        if (!irepon(GENCS)) {
                genko_ari(0);
                while(genko_ari('?')) {
                        motor();
                        wait_mtr();
                }
        }else{
                for(i=0;i<200;i++){
                        motor();
                        wait_mtr();
                }
        }

        pepo(1);
        motorp(MOT_IR,OFF);
}

extern  uchar   step_task[8][8];
task02_4_on()
{
    step_task[2][4]=1;
}
task02_4()
{
static  ushort  i,j;

    switch( step_task[2][4] ){
    case 1:
                motorp(MOT_IR,ON);
    step_task[2][4]=2;
    break;

    case 2:
    case 3:
```

3

保守／廃止

BMTR.C

```
        if (!irepon(GENCS)) {
                genko_ari(0);
                motor();
        }else{
    step_task[2][4]=4;
        }
    break;

    case 4:
        i=0;
    step_task[2][4]=5;
    case 5:
        if(i<300) {
                i++;
                motor();
        }else{
                pepo(1);
                motorp(MOT_IR,OFF);
    step_task[2][4]=0;
        }
    break;
    }
}

/*********************************************************
 *      初期のの記録紙フィード                           *
 *      in...   uchar                                    *
 *      out..   なし                                     *
 *********************************************************/
feed_init()
{
        feedvarlen(MOT_PR,8);
}
/*********************************************************
 *      ８ｍｍ位の記録紙フィード                 *
 *      in...   uchar                                    *
 *      out..   なし                                     *
 *********************************************************/
feed8()
{
ushort  i,j;

        motorp(MOT_PR,ON);

        for(i=0;i<60;i++) {
                motor();
                wait_mtr();
                if(!irepon(PAPES))break;
        }

        motorp(MOT_PR,OFF);
}

/*********************************************************
 *      ２４ｍｍ位の記録紙フィード                       *
 *      in...   uchar                            *
 *      out..   なし                                     *
 *********************************************************/
feed24()
{
ushort  i,j;

        motorp(MOT_PR,ON);

        for(i=0;i<195;i++) {
                motor();
                wait_mtr();
                if(!irepon(PAPES))break;
        }
```

4

保守／廃止

BMTR.C

```
        motorp(MOT_PR,OFF);
}



/*********************************************************
 *      原稿・記録紙フィード                             *
 *      in...   uchar   which;どちらのモーター  *
 *              ushort length;長さ（mm）        *
 *      out..   なし                                     *
 *              長さは可変長、モーター選択可             *
 *********************************************************/
extern  uchar   TT128;
feedvar}en(which,length)
uchar   which;
ushort  length;
{
ushort          i;
static  ushort  flag_slow;

        motorp(which,ON);

        if( TT128 == 0 ){
                flag_slow = 48;
        }

        for(i=0;i<length*4;i++) {
                motor();
                wait_mtr();
                if(flag_slow != 0){
                        flag_slow--;
                        wait(8);
                }
                if(!irepon(PAPES))break;
        }

        motorp(which,OFF);

        line_count_plus( length );

        TT128 = 4;
}

extern  uchar   step_task[8][8];
task01_3_on()
{
        step_task[1][3]=1;
}
task01_3(which,length)
uchar   which;
ushort  length;
{
static  ushort  i;

    switch( step_task[1][3] ){
    case 1:
        motorp(which,ON);
        i = 0;
    step_task[1][3] = 2;
    break;

    case 2:
        if( i<length*4 ){
            if( MSTATE == 0 ){
                i++;
                motor();
            }
        }else{
```

保守／廃止

BMTR.C

```
    step_task[1][3] = 3;
        }
                if(!irepon(PAPES)){
    step_task[1][3] = 3;
                }
    break;

    case 3:
        motorp(which,OFF);

        line_count_plus( length );
    step_task[1][3] = 0;
    break;
    }
}


/*********************************************************
 *      記録紙の前方フィード                             *
 *              ロール紙切れはみない                     *
 *      in...   uchar                           *
 *      out..   なし                                     *
 *********************************************************/
feedfd()
{
ushort  i,j;

        motorp(MOT_PR,ON);

        for(i=0;i<180;i++) {
                motor();
                wait_mtr();
        }

        motorp(MOT_PR,OFF);

}

/*********************************************************
 *      記録紙の逆転フィード                             *
 *              ロール紙切れはみない                     *
 *      in...   uchar                           *
 *      out..   なし                                     *
 *********************************************************/
feedbk()
{
ushort  i,j;

        motorp(MOT_PR,ON);

        for(i=0;i<170;i++) {
                revmotor();
                wait_mtr();
        }

        motorp(MOT_PR,OFF);

}


/*********************************************************
 *      原稿・記録紙フィード（逆転）                     *
 *      in...   uchar   which;どちらのモーター  *
 *              ushort length;長さ（mm）        *
 *      out..   なし                                     *
 *              長さは可変長、モーター選択可             *
 *********************************************************/
feedrevlen(which,length)
uchar   which;
```

保守／廃止

BMTR.C

```
ushort  length;
{
ushort  i;


        motorp(which,ON);

        for(i=0;i<length*8;i++) {
                revmotor();
                wait(6);
        }

        motorp(which,OFF);
}
/*************************************************************
 *      W a i t   t i m e r                                  *
 *      in...   ushort count;c o u n t          *
 *      out..   なし                                         *
 *************************************************************/
#pragma loop_opt( off )
wait_mtr()
{
        while( MSTATE != 0);
}
#pragma loop_opt( on )
```

保守／廃止

## A.7　BBIO.C

BBIO.C

```
/*****************************************************************************
        スイッチ・ＬＥＤ・センサー・ＮＣＵＢＩＯＳプログラム
        file name "BBIO.C"
*****************************************************************************/

/*      Ｃ言語基本設定  **/
#include        "BASICDEF.H"
/*      ＃３２９ファイル+**/
#include        "hd329.h"
/*      Ｖ５５ＰＩの構造体ファイル      **/
#include        "INDAT55.H"

/*      Ｖ５５ＰＩの構造体とベースアドレスを設定**/
extern  struct  v55pi_regs far  *v55reg;
/*      ＰＬＡ入出力アドレス    **/
extern  uchar far* PLA;


extern  uchar   FMODE;
extern uchar    W200MS;
extern uchar    FW200MS;
extern uchar    WINKP;
extern uchar    PW200MS;
extern uchar    TMPBUF[];

/*****************************************************************************
        スイッチ・センサー入力ＢＩＯＳ

        STARS,STOPS,FINES,TALKS,ATMTS,COVES
        PAPES,GENCS,HOOKS,FNETS,RINGS,COUPS

*****************************************************************************/
/*************************************************
 *      スイッチ・センサー入力ＢＩＯＳ          *
 *      in...   uchar スイッチ                  *
 *      out..   ON/OFF                          *
 *      for FAX55 ES                            *
 *              STARS   押してる／押してない    *
 *              STOPS   押してる／押してない    *
 *              FINES   押してる／押してない    *
 *              HOOKS   ON:  フックを上げてる   *
 *              RINGS   OFF: 鳴ってる           *
 *              COUPS   ON:  ある               *
 *              GENCS   ない／ある              *
 *              PAPES   ある／ない              *
 *              COVES   開いてる／閉まってる    *
 *              HEADS   OFF: 熱い               *
 *************************************************/
irepon(sensor)
ushort sensor;
{
        if( sensor == PAPES ){
                if( ( (uchar)sensor & v55reg->P5 ) == 0 ){
                                return(OFF);
                }else{
                                return(ON);
                }
        }else
        if( (sensor & 0xff00) == 0x100 ){
                if( ( (uchar)sensor & v55reg->P1 ) == 0 ){
                                return(ON);
                }else{
                                return(OFF);
                }
        }else if( (sensor & 0xff00) == 0x200 ){
                if( ( (uchar)sensor & v55reg->P2 ) == 0 ){
                                return(ON);
                }else{
                                return(OFF);
```

1

保守／廃止

BBIO.C

```
                }
        }else if( (sensor & 0xff00) == 0x400 ){
                if( ( (uchar)sensor & v55reg->P4 ) == 0 ){
                                return(ON);
                }else{
                                return(OFF);
                }
        }else if( (sensor & 0xff00) == 0x500 ){
                if( ( (uchar)sensor & v55reg->P5 ) == 0 ){
                                return(ON);
                }else{
                                return(OFF);
                }
        }else if( (sensor & 0xff00) == 0x700 ){
                if( ( (uchar)sensor & v55reg->P7 ) == 0 ){
                                return(ON);
                }else{
                                return(OFF);
                }
        }else if( (sensor & 0xff00) == 0x800 ){
                if( ( (uchar)sensor & v55reg->P8 ) == 0 ){
                                return(ON);
                }else{
                                return(OFF);
                }
        }else{
                                return(OFF);
        }
}

/************************************************
 *      スイッチ入力ＢＩＯＳ                    *
 *      in...   char    bel クリック音ON/OFF     *
 *      out..   uchar sw;                        *
 ************************************************/
keyin(bel)
uchar   bel;
{
uint            sw;
static  uint    swmm;
uint            ret_val;

        sw = ret_val = 0;

        if(irepon(STARS)) sw|=STARS;
        if(irepon(FINES)) sw|=FINES;
        if(irepon(STOPS)) sw|=STOPS;

        if(( sw & ~swmm ) & ( STARS | FINES | STOPS ))
        {
                if(bel==ON) pepo(1);
                ret_val = sw & ( STARS | FINES | STOPS );
        }

        swmm = sw;

        return(ret_val);
}



/*****************************************************************************
        ＬＥＤ　点灯・消灯・点滅　各種ＢＩＯＳ
*****************************************************************************/
/************************************************
 *      ＬＥＤ点灯・消灯ＢＩＯＳ        *
 *      in...   char ledno, sw (ON/OFF)         *
 *      out..   なし                            *
 ************************************************/
```

2

保守／廃止

BBIO.C

```
led(ledno,sw)
uchar    ledno,sw;
{
        if(sw == ON) {
                v55reg->P0 |= ledno;
        }
         else {
                v55reg->P0 &= (uchar)~ledno;
        }
}


/************************************************
 *      ＬＥＤ反転ＢＩＯＳ                     *
 *      in...   char ledno                      *
 *      out..   なし                            *
 ************************************************/
ledrev(ledno)
uchar  ledno;
{
                v55reg->P0 ^= ledno;
}

/************************************************
 *      スタート、ファインＬＥＤ点滅ＢＩＯＳ   *
 *      in...char ledno, sw (ON/OFF),time(131mS)*
 *      out..なし                               *
 ************************************************/
winker(ledno,sw,time)
uchar   ledno,sw,time;
{

        if(ledno & STARL) {
                if( (sw == ON)
                 &&( (time >= 0)
                   &&(time <  4) ) ){
                        W200MS=time;
                        WINK = ON;
                        led(STARL,ON);
                }
                else{
                        W200MS=OFF;
                        WINK = OFF;
                        led(STARL,OFF);
                        }
        }

        if(ledno & POWERL) {
                if(sw == ON) {
                        PW200MS=time;
                        WINKP = ON;
                        led(POWERL,ON);
                }
                else        {
                        PW200MS=OFF;
                        WINKP = OFF;
                        led(POWERL,OFF);
                        }
        }

        if(ledno & FINEL) {
                if(sw == ON) {
                        PW200MS=time;
                        WINKP = ON;
                        led(FINEL,ON);
                }
                else        {
                        PW200MS=OFF;
                        WINKP=OFF;
```

3

保守／廃止

BBIO.C

```
                        led(FINEL,OFF);
                }
        }
}

/*****************************************************************************
        通信関係各種切換　ＢＩＯＳ
*****************************************************************************/
/*************************************************
 *              モデム切換　ＢＩＯＳ             *
 *      in...   uchar ON/OFF                     *
 *      out..   なし                             *
 *************************************************/
modemoki(sw)
uchar   sw;
{
        if(sw==ON){
        }else{
        }
}

/*************************************************
 *      モデムリセット　ＢＩＯＳ                *
 *      in...   なし                             *
 *      out..   なし                             *
 *************************************************/
modemrst()
{
                v55reg->P2 &= (uchar)~MDRST;
                v55reg->P2 |= MDRST;
}

/*************************************************
 *      エラーＬＥＤ点灯チェックＢＩＯＳ        *
 *      in...   なし                             *
 *      out..   0<>ＬＥＤ点灯                   *
 *************************************************/
errcheck()
{
        if(v55reg->P0 & ERROL)  return(-1);
        else                    return(0);
}

/*************************************************
 *      ロール紙のチェック                       *
 *      in...   なし                             *
 *      out..   0<>ロール紙なし                  *
 *************************************************/
rollchk()
{
static  char    bel;

        if(irepon(PAPES)) {
                led(PAPEL,OFF);
                bel=OFF;
        }
        else {
                if(bel==OFF){
                        led(PAPEL,ON);
                        beep(3000);
                        bel=ON;
                }
                return(-1);
        }

        return(0);
}

/*************************************************
```

4

保守／廃止

BB10.C

```
 *      原稿のチェック                          *
 *      in...   なし                            *
 *      out..   なし                            *
 ************************************************/
genkochk()
{
static  uchar   kami1,kami2;
static  ulong   Time;
static  uchar   sumi;

        kami1 = (uchar)irepon(GENCS);

        if( !kami1 && kami2 ){
            timeset(&Time, 1000);
            sumi = 0;
        }else if( kami1 ){
            sumi = 1;
        }

        if( timechk(&Time)
         && sumi == 0 ){
                feed();
                sumi = 1;
        }

        kami2 = kami1;
}
genkochk_2()
{
static  uchar   kami1,kami2;
static  ulong   Time;
static  uchar   sumi;

        kami1 = (uchar)irepon(GENCS);

        if( !kami1 && kami2 ){
            timeset(&Time, 1000);
            sumi = 0;
        }else if( kami1 ){
            sumi = 0xff;
        }

        if( timechk(&Time)
         && sumi < 7 ){
                feed_2();
                sumi++;
        }

        kami2 = kami1;

return( (sumi == 0x00) || ( sumi >= 7 ) );
}

genko_ari(uchar ctrl)
{
static  ushort  Amari, Yon;
static  uchar   trig;
#define C_Yon   100
    if( ctrl == 0 ){
                Yon = C_Yon;
        Amari = Yon;
        trig  = 0;
    }else if( ctrl == 1 ){
                Yon = C_Yon;
        if (FMODE == 0x00)
                Yon /= 2;
        Amari = Yon;
        trig  = 0;
    }else{
```

5

保守／廃止

BBIO.C

```
        if( (!irepon(GENCS)) && (trig == 0) )
                return (1);

        trig  = 1;
        if (--Amari == 0)
                return (0);
        else
                return (1);
    }
}

/*********************************************************
 *      カバーオープンのチェックに                       *
 *********************************************************/
coverchk()
{
static uchar    cover;
ushort  i;

        if(irepon(COVES)) {
                if(cover==OFF){
                        led(COVEL,ON);
                        beep(3000);
                        cover=ON;
                        return(-1);
                }
                else    return(-1);
        }
        else{
                led(COVEL,OFF);
                cover=OFF;
                return(0);
        }

}

/*****************************************************************************
        ＢＵＺＺＥＲ　発音各種ＢＩＯＳ
*****************************************************************************/

/************************************************
 *      ブザーオン／オフＢＩＯＳ        *
 *      in...    ushort time;           *
 *      out..    なし                   *
 ************************************************/
void buzzer(sw)
ushort  sw;
{
        if(sw==ON){
                PLA[0] &= (uchar)~BUZZ;
                BZSW=ON;
        }else{
                PLA[0] |= BUZZ;
                BZSW=OFF;
        }
}


/************************************************
 *      指定時間だけブザーを鳴らすＢＩＯＳ      *
 *      in...    ushort time;           *
 *      out..    なし                   *
 ************************************************/
beep(time)
ushort  time;
{
                PLA[0] &= (uchar)~BUZZ;
                wait(time);
                PLA[0] |= BUZZ;
```

6

BBIO.C

```
}

/************************************************
 *      指定回数だけブザーを鳴らすＢＩＯＳ      *
 *      in...   uchar   count;          *
 *      out..   なし                            *
 ************************************************/
pepo(count)
uchar   count;
{
        for(;count!=0;count--) {
                beep(100);
                wait(100);
        }
}

/************************************************
 *      指定された時間だけ待つＢＩＯＳ          *
 *              (time) msec                     *
 *      in...   ushort time;            *
 *      out..   なし                            *
 ************************************************/
wait(time)
ushort time;
{
ushort  bubu[4][2];
        timer(time);
        while(TIME2!=0);
}
wait_halt(time)
ushort time;
{
        timer(time);
        while(TIME2!=0){
                cpu_halt();
        }
}

/************************************************
 *      １ｍｓｅｃ割り込みタイマにデータセット  *
 *              (time) msec                     *
 *      in...   ushort time;            *
 *      out..   なし                            *
 ************************************************/
timer(time)
ushort  time;
{
        TIME2=time;
}

/************************************************
 *      １ｍｓｅｃ割り込みタイマ0初期化        *
 *      in...   なし                            *
 *      out..   なし                            *
 ************************************************/
timeclr()
{
    _disable();
        TIME1 =0;
    _enable();
}
timeclr_A()
{
    _disable();
        if(TIME1 >= (-1L*60*60*1000) )  TIME1 =0;
    _enable();
}

/************************************************
```

保守／廃止

BBIO.C

```
 *      指定時間の経過チェック                    *
 *      in...   ulong   *time                     *
 *      out..   1..timeup 0..no                   *
 ************************************************/
timechk(time)
ulong   *time;
{
    _disable();
        if(TIME1 >= *time){
    _enable();
        return(1);
        }else{
    _enable();
        return(0);
        }
}

/************************************************************
 *      タイマのセット                                      *
 *      in...   ulong   *tbuf(32bit)４９日タイマ*
 *              ushort  stime(16bit)０～６５秒              *
 *      out..   なし                                        *
 ************************************************************/
timeset(tbuf,stime)
ulong   *tbuf;
ushort  stime;
{
    _disable();
        *tbuf=(TIME1+(ulong)stime);
    _enable();
}

/************************************************
 *    ファインモード設定＆ＬＥＤ反転ＢＩＯＳ    *
 *      in...   なし                            *
 *      out..   なし                            *
 ************************************************/
setfine(sw)
uchar   sw;
{
        if(sw == ON) {
                FMODE++;

                if(FMODE == 1) {
                        winker(FINEL,OFF,OFF);
                        led(FINEL,ON);
                        dithmod(OFF);
                }


                else {
                        FMODE=0;
                        winker(FINEL,OFF,OFF);
                        led(FINEL,OFF);
                        dithmod(OFF);
                        }
        }

        if((sw==OFF)||(FMODE==0)) {
                FMODE=0;
                winker(FINEL,OFF,OFF);
                led(FINEL,OFF);
                dithmod(OFF);
        }

}

/************************************************
 ************************************************/
```

8

保守／廃止

BBIO.C

```
p_hold()
{
        v55reg->P0 = (~PHOLD);

}


/************************************************
 *      マシンを擬似中間調にする          *
 *      in.....無し                                 *
 *      out....無し                                 *
 ************************************************/
 dithmod(sw)
 uchar  sw;
 {

        if(sw){
                v55reg->P4 |= DITH;
        }else{
                v55reg->P4 &= (uchar)~DITH;
        }
 }



/****************************************
 *      STARLWINK                       *
 *      in....nothing                   *
 *      out...nothing                   *
 ****************************************/

starlwink()
{
static  uchar   winkflag;

        if(!irepon(GENCS)){
            if(winkflag==OFF){
                winker(STARL, ON,  0);
                winkflag=ON;
            }
        }
        else    {
                winkflag=OFF;
                winker(STARL,OFF,OFF);
        }
}


/****************************************
 *      thtempchk()                     *
 *      in....無し                      *
 *      out...無し                      *
 ****************************************/
thtempchk()
{
ulong   t;

        if( v55reg->ADCR0 <= 0x33 ){
                beep(3000);
                led(ERROL,ON);
                while( v55reg->ADCR0 <= 0x33 ){
                        timeset(&t,10000);
                        while(!timechk(&t)){
                                led(ERROL,ON);
                                wait(200);
                                led(ERROL,OFF);
                                wait(800);
                        }
```

保守／廃止

BB10.C

```
                }
                timeset(&t,20000);
                while(!timechk(&t)){
                        led(ERROL,ON);
                        wait(200);
                        led(ERROL,OFF);
                        wait(800);
                }
                led(ERROL,OFF);
        }

}


/************************************************
 *      ｓｔｗｉｄｔｈ（）                      *
 *      サーマルヘッドの温度最適幅のセット      *
 *      但し、低温時あまりストローブ幅を長く    *
 *      しすぎると、受信で１ライン２０ｍＳを    *
 *      超えてしまうので注意                    *
 *      in...温度（ｔｅｍｐｃｈｋ（））        *
 *      out..無し                               *
 ************************************************/

stwidth()
{
unsigned char   i,tmp_ave;
unsigned short  tmp;

        tmp=0;

        for(i=0;i<16;i++){
                tmp=tmp+TMPBUF[i];
        }
        tmp_ave=tmp/16;

        v55reg->TMC1 &= B01111001;

        if((tmp_ave*0x0c) > 1200)
                v55reg->CM31=1200;
        else
                v55reg->CM31 = tmp_ave*0x0a;

        v55reg->CM30 = v55reg->CM31 - 1;

        v55reg->TMC1 |= B10000000;


}
```

10

## A.8　CPY.C

CPY.C

```
/*****************************************************************************

*****************************************************************************/

#include        "hd329.h"

extern    struct        databuf far     *p_data;
extern unsigned char    DANGER;
extern  unsigned int    Amari, Yon;


/*********************************************************
 *      コピー処理ルーチン                               *
 *********************************************************/
copy()
{

        copystart();
        m_copy();
        copyend();
}

/*********************************************************
 *      コピー時はこのルーチンでループを描く             *
 *      カバー／記録紙／原稿／ストップを見ながら  *
 *      処理を行う。                                     *
 *              in...なし                                *
 *              out..なし                                *
 *********************************************************/
uchar far*      imgread_c(uchar, ushort);
void            imgwrite(uchar, ushort);
uchar far*      img_ptr(uchar);

m_copy()
{
ushort  size_line;
uchar   linen_imr,linen_prn,linen_cnt;
uchar   i;
uchar far*      ipf;
uint    head_lnn;

        genko_ari(0);
        linen_imr=linen_prn=linen_cnt=0;
        size_line = 216;
        stwidth();

        set_HC(0);
        set_Head_Fn(1);
        head_lnn=sethead(img_ptr(linen_imr), 0);
        linen_imr = head_lnn;   linen_imr %= 32;
        linen_cnt = head_lnn;
        ipf=img_ptr(linen_imr);
        for(i=0; i<size_line; i++){
                *(ipf+i)=0;
        }
        linen_imr++;    linen_imr %= 32;
        linen_cnt++;

        while(linen_cnt!=0){
                imgwrite(linen_prn,size_line);
                linen_prn++;
                stwidth();
                ltc_str();
                motor();
                linen_cnt--;
        }


        while(chksens()){
```

1

保守／廃止

CPY.C

```
                imgread_c(linen_imr,size_line);
                linen_imr++;
                imgwrite(linen_prn,size_line);
                linen_prn++;
                stwidth();
                ltc_str();
                motor();
        }
}

/*****************************************
 *      コピーを中断するのは以下の       *
 *      時だけ。                         *
 *      in....なし                       *
 *      out...unsigned char              *
 *****************************************/


chksens()
{
        if(     !irepon(STOPS)
                && genko_ari('?')
                && irepon(PAPES)
                && (!irepon(COVES))
                && !DANGER      ){
return(1);
        }else{
return(0);
        }
}


/*************************************************
 *      コピーする時のいろんな処理               *
 *      のルーチン。                             *
 *************************************************/


copystart()
{
        readtime();
        setfdate(fdate);

        clrthermal();
        motorp(MOT_PR,ON);
        thermalp(ON);

        line();
        feedvarlen(MOT_PR,5);

        feedvarlen(MOT_IR,5);
        thermalp(ON);
        cisp(ON);
        motorp(MOT_PR | MOT_IR,ON);

        wait(100);

        imgread(0,216);
        wait(100);
}

/*************************************************
 *      コピーエンド時のいろんな処理の           *
 *      ルーチン                                 *
 *************************************************/


copyend()
{
```

2

保守／廃止

CPY.C

```
        motorp(MOT_IR,OFF);
        cisp(OFF);
        motorp(MOT_PR,OFF);
        thermalp(OFF);
        if(rollchk()){
                endfeed();
                return(-1);
                }
        if(coverchk())return(-1);

        feedvarlen(MOT_PR,5);
        line();
        feedvarlen(MOT_PR,5);
        line_count_0();
        feedvarlen(MOT_PR,line_count_end_get());

        endfeed();
        setfine(OFF);
}
```

3

## A.9　BRTC.C

BRTC.C

```
/*****************************************************************************
        ＃３２９
*****************************************************************************/

/*      Ｃ言語基本設定  **/
#include        "BASICDEF.H"

/*      ＃３２８ファイル+**/
#include        "hd329.h"

/*      ＲＴＣ内のレジスタ      **/
#define ONESEC          0x00            /*  １秒桁**/
#define TENSEC          0x01            /*１０秒桁**/
#define ONEMIN          0x02            /*  １分桁**/
#define TENMIN          0x03            /*１０分桁**/
#define ONEHOUR         0x04            /*  １時桁**/
#define TENHOUR         0x05            /*１０時桁**/
#define WEEK            0x06            /*  曜日桁**/
#define ONEDATE         0x07            /*  １日桁**/
#define TENDATE         0x08            /*１０日桁**/
#define ONEMONTH        0x09            /*  １月桁**/
#define TENMONTH        0x0a            /*１０月桁**/
#define ONEYEAR         0x0b            /*  １年桁**/
#define TENYEAR         0x0c            /*１０年桁**/
#define RTCCNT1         0x0d            /*コントロールレジスター１**/
#define RTCCNT2         0x0e            /*コントロールレジスター２**/
#define RTCMODE         0x0f            /*モードレジスター**/

        uchar far* RTC;         /**ＲＴＣ入出力アドレス（ＭＭＩＯ）**/

        B_INFO *bufrtc =(B_INFO *)B_ADR;        /*ＲＴＣ情報バッファ
                                                ，通信管理情報**/

/*************************************************************
                ＲＴＣイニシャライズ
*************************************************************/
rtcini()
{
uchar   boo;
        RTC[RTCMODE]    =0x00;
        RTC[RTCCNT1]    =0x00;
        boo = bufrtc -> KANRI[34].result;
}

/*************************************************
 *      時間を引っ張ってくるルーチン             *
 *              in.....なし                      *
 *              out....なし                      *
 *************************************************/
readtime()
{
        RTC[RTCMODE]    =0x73;

        bufrtc -> year[0]       = ((0x0f & RTC[TENYEAR] ) | 0x30);
        bufrtc -> year[1]       = ((0x0f & RTC[ONEYEAR] ) | 0x30);
        bufrtc -> month[0]      = ((0x0f & RTC[TENMONTH]) | 0x30);
        bufrtc -> month[1]      = ((0x0f & RTC[ONEMONTH]) | 0x30);
        bufrtc -> date[0]       = ((0x0f & RTC[TENDATE] ) | 0x30);
        bufrtc -> date[1]       = ((0x0f & RTC[ONEDATE] ) | 0x30);
        bufrtc -> hour[0]       = ((0x0f & RTC[TENHOUR] ) | 0x30);
        bufrtc -> hour[1]       = ((0x0f & RTC[ONEHOUR] ) | 0x30);
        bufrtc -> min[0]        = ((0x0f & RTC[TENMIN]  ) | 0x30);
        bufrtc -> min[1]        = ((0x0f & RTC[ONEMIN]  ) | 0x30);

}

/*********************************************************
 *      フォーマット付き日付けデータ                     *
 *      in...   fdate                                    *
```

1

保守／廃止

BRTC.C

```
 *      out..   無し                                          *
 ************************************************************/

setfdate(fdate)
uchar   *fdate;
{
uchar   i;

        *fdate=0x27;
        fdate++;

        for(i=0;i<2;i++,fdate++) *fdate=(bufrtc -> year[i]);

        *fdate=' ';
        fdate++;

        for(i=0;i<2;i++,fdate++) *fdate=(bufrtc -> month[i]);

        *fdate='/';
        fdate++;

        for(i=0;i<2;i++,fdate++) *fdate=(bufrtc -> date[i]);

        *fdate=' ';
        fdate++;

        for(i=0;i<2;i++,fdate++) *fdate=(bufrtc -> hour[i]);

        *fdate=':';
        fdate++;

        for(i=0;i<2;i++,fdate++) *fdate=(bufrtc -> min[i]);

        *fdate=0;

}


/*****************************************************************************
        ＲＴＣへ時間をセットする関数
*****************************************************************************/
settime()
{
        RTC[RTCMODE]    =0x07;


        RTC[ONEYEAR]    =(bufrtc ->year[1] & 0xf);
        RTC[TENYEAR]    =(bufrtc ->year[0] & 0xf);
        RTC[ONEMONTH]   =(bufrtc ->month[1] & 0xf);
        RTC[TENMONTH]   =(bufrtc ->month[0] & 0xf);
        RTC[ONEDATE]    =(bufrtc ->date[1] & 0xf);
        RTC[TENDATE]    =(bufrtc ->date[0] & 0xf);
        RTC[ONEHOUR]    =(bufrtc ->hour[1] & 0xf);
        RTC[TENHOUR]    =(bufrtc ->hour[0] & 0xf);
        RTC[ONEMIN]     =(bufrtc ->min[1] & 0xf);
        RTC[TENMIN]     =(bufrtc ->min[0] & 0xf);

        RTC[RTCMODE]    =0x73;
}
```

2

保守／廃止

## A.10　SR.C

SR.C

```
/* ************************************************** **/
/*      Ｔ３０送受信プログラム                          **/
/* ************************************************** **/

/*      Ｃ言語基本設定  **/
#include        "BASICDEF.H"
/*      ＃３２９ファイル**/
#include        "HD329.H"
/*      Ｖ５５ＰＩ内部データ領域・マップ **/
#include        "INDAT55.H"
/*      Ｖ５５ＰＩの構造体とベースアドレスを設定**/
extern  struct  v55pi_regs far  *v55reg;

/*      Ｔ３０の頭ファイル      **/
#include        "t30.h"

timeset(ulong*,ushort);
timechk(ulong*);

extern  uchar   FMODE;          /* ファイン／ノーマルフラグ **/

#define BUFSIZE         0x2000
#define BUFSIZE_R       0x8000

extern  uchar   *SR_BTOP;       /* バッファトップポインタ **/
extern  uchar   *SR_BBOT;       /* バッファボトムポインタ **/
extern  ushort  SR_COUNT;       /* バッファカウンタ **/
        ushort          SR_COUNT_add;
struct{
        uchar   SR_BUF[BUFSIZE];/* バッファ本体 **/
        uchar   SR_ENDF;        /* 終了フラグ **/
}SR_info;
extern  uchar * SR_BTOP_G;      /* 疑似バッファトップポインタ **/
extern  uchar * SR_BBOT_G;      /* 疑似バッファボトムポインタ **/
extern  uchar   DANGER;         /* このフラグが立ったら危険 **/
extern  uchar   STP;            /* STP FLAG **/
extern  unsigned char   endf_DMA0;      /*ＤＭＡエンドフラグ    **/
ulong   Gobyo;                  /* 5 sec **/
uchar   Gostop;

extern  struct  nama_data_s far *nama_data;     /**画像生データ**/
extern  struct  comp_data_s far *comp_data;     /**画像圧縮データ**/
extern  struct  trnp_tabl_s far *trnp_tabl_A;   /**変化点テーブルＡ**/
extern  struct  trnp_tabl_s far *trnp_tabl_B;   /**変化点テーブルＢ**/


/*      関数定義**/
void    bufinit(void);
uint    encodeinit(uchar, ushort, uchar);
uint    encodeinit_r(uchar, ushort);
uchar far*      imgread(uchar, ushort);
uint    encode(uchar far*,uchar,uchar);
int     bufwrite  (char *, ushort, uchar );
int     bufwrite_0(char far*, ushort );
int     bufwrite_1(char far*, ushort );
int     bufwrite_2(char far*, ushort );
int     bufwrite_3(char *, ushort );
int     decodeinit(uchar);
int     decodeinit_r(uchar);
int     decodeinit_ec(uchar);
int     decodeinit_ec_r(uchar);
int     decode(uchar far*, uchar, uint);
void    imgwrite(uchar, ushort);
uchar far*      img_ptr(uchar);

/*********************************************************
 *      送信メインルーチン                               *
 *      in...   uchar 符号方式(MH=0,MR=1,MMR=2) *
 *      in...   ushort  1ライン符号化バイト数  *
```

1

保守／廃止

SR.C

```
 *      out..   なし                                          *
 ************************************************************/
uchar           linen = 0;
ushort          size_line = 216;
uchar           SR_FILL;


Send(code,clen,fif_fine)
uchar   code;
ushort  clen;
uchar   fif_fine;
{
uint            len,a_len;
uchar           i,linen_imr,linen_enc,linen_cnt,ii,jj,kk,pagef;
static  ulong   t_minscan;
uchar far*      ipf;
uint            head_lnn;

        encodeinit(code,clen,fif_fine);


        motorp(MOT_IR,ON);
        cisp(ON);
        wait(200);
        genko_ari(1);

        pagef=getpage();
        linen_imr =0;   linen_enc =0;   linen_cnt =0;

        ipf=img_ptr(linen_imr);

        for(i=0; i<size_line; i++){
                *(ipf+i)=B00000000;
        }

        *(ipf+  0)=B10110111;
        *(ipf+(size_line-1))=B11101101;
        linen_imr++;    linen_imr %= 32;
        linen_cnt++;

        set_HC(1);
        set_Head_Fn(FMODE);
        head_lnn=sethead(img_ptr(linen_imr),pagef);
        linen_imr += head_lnn;  linen_imr %= 32;
        linen_cnt + head_lnn;


        while(linen_cnt!=0){
                len=encode(img_ptr(linen_enc),code,OFF);
                bufwrite_0((char far*)comp_data  ,len  );

                if((clen/8-len)>0)fillwrite(clen/8-len+1);

                linen_enc++;    linen_enc %= 32;
                linen_cnt--;
        }



        imgread_1(linen_imr,size_line);

        linen_imr++;    linen_imr %= 32;
        linen_cnt++;
        motor();


        while (1) {

                if(linen_cnt<32){
```

2

SR.C

```
                while(endf_DMA0==0);
                imgread_1(linen_imr,size_line);

                linen_imr++;    linen_imr %= 32;
                linen_cnt++;
                motor();
        }

        if(linen_cnt!=0){
                len=encode(img_ptr(linen_enc),code,OFF);
                bufwrite_0((char far*)comp_data  ,len  );

                if((clen/8-len)>0)fillwrite(clen/8-len+1);

                linen_enc++;    linen_enc %= 32;
                linen_cnt--;
        }

        if(STP){
                pepo(1);
                winker(STARL,ON,OFF);
                break;
        }

        if(FMODE==0x00){
                motor();
        }

        if(!genko_ari('?')) break;
}

while(endf_DMA0==0);
imgread_1(linen_imr,size_line);
linen_imr++;    linen_imr %= 32;
linen_cnt++;
motor();
if(FMODE==0x00){
        motor();
}
while(endf_DMA0==0);
imgread_1(linen_imr,size_line);
linen_imr++;    linen_imr %= 32;
linen_cnt++;
motor();
if(FMODE==0x00){
        motor();
}

if(STP==OFF && DANGER==OFF){
        buzzer(ON);
        winker(STARL, ON,  0);
}

cisp(OFF);

timeset(&Gobyo, 3000);
Gostop = OFF;
while((SR_COUNT!=0) || !timechk(&Gobyo) ){

        if( STP == OFF ){
                genkochk_2();
                if(keyin(ON) == FINES) setfine(ON);
        }else{
                break;
        }
}


len=encode(img_ptr(linen_enc),code,OFF);
```

3

保守／廃止

SR.C

```
        bufwrite_0((char far*)comp_data  ,len  );

        timeset(&Gobyo, 3000);
        while((SR_COUNT!=0) || !timechk(&Gobyo) ){

                if( STP == OFF ){
                        genkochk_2();
                        if(keyin(ON) == FINES) setfine(ON);
                }else{
                        break;
                }
        }

        wait_FILL();
        len=encode(img_ptr(linen_enc),code, ON);
        bufwrite_0((char far*)comp_data  ,len  );

        while(SR_COUNT!=0);
        SR_info.SR_ENDF = ON;

        if( STP == OFF ){
                while( !genkochk_2() );
        }

        motorp(MOT_IR,OFF);
        return (0);
}



wait_FILL()
{
        SR_FILL = 0;
        while(SR_FILL == 0);
}


/*********************************************************
 *      画像データを指定バッファに１ライン読み込む      *
 *      in...   char *バッファポインタ                   *
 *      in...   ushort  *バッファサイズ          *
 *      out..   なし                                     *
 *********************************************************/
uchar far* imgread(linen,size)
uchar   linen;
ushort  size;
{
uchar far*      ptr;
    if(is_mdmIE2()){
      mdmIE2_sw(0);
      while(is_mdmIA2());
        img_input(linen % 32,size % 0x101);
      set_mdmDBUFF( get_mdmDBUFF() );
      mdmIE2_sw(1);
    }else{
        img_input(linen % 32,size % 0x101);
    }
        ptr = (uchar far*)&(nama_data->arry[(linen % 32)][0]);
return(ptr);
}

/*********************************************************
 *      ＣＥＰ符号化初期化                              *
 *      in...   uchar 符号方式(MH=0,MR=1,MMR=2) *
 *      in...   ushort  最小符号化バイト数       *
 *      out..   なし                                     *
 *********************************************************/
```

4

SR.C

```
struct {
        uchar   K;              /*Kカウンタ**/
        uchar   L;              /*Lカウンタ**/
        ushort  OD_DT;          /*半端符号データセーブ領域**/
        uchar   OD_BT;          /*半端符号データビット数セーブ領域**/
        ushort  TRBP;           /*送信バッファオフセットポインタ**/
        ushort  TRBSZP;         /*送信バッファ残りサイズセーブ領域**/
        ushort  ATRBP;          /*符号データの格納領域のオフセットポインタ**/
        ushort  ADBP;           /*入力情報の格納領域のオフセットポインタ**/
        ushort  ATR_LST;        /*1ライン処理開始時の出力情報のオフセットアドレスの記憶領域
**/
        ushort  TR_LST;         /*1ライン処理開始時の送信バッファのオフセットアドレスの記憶
領域**/
        ushort  R_cnt;          /*ＲＴＣコード送信用カウンタ**/
        ushort  RTCDP[5];       /*ＲＴＣコード格納領域**/
}comp_work;
/*圧縮処理用パラメータ**/
struct{
        uchar far*      gptr;   /* 画像データポインタ **/
        ushort          gcnt;   /* 画像データバイト数 **/
        ushort          clen;   /* 符号データバイト数，何の？ **/
        uchar           Kval_m1; /* K値－1 **/
        uchar           CODE;   /* 符号化方式 **/
        ushort          clen1L; /* 1走査線符号データバイト数 **/
}comp_param;

uint encodeinit(code,clen,fif_fine)
uchar   code;
ushort  clen;
uchar   fif_fine;
{
        comp_param.gcnt = 216;

    if( fif_fine == 1 ){
        comp_param.Kval_m1 = 4-1;
    }else{
        comp_param.Kval_m1 = 2-1;
    }

    if( code == 0 ){
        comp_param.CODE = 0;
    }else{
        comp_param.CODE = 1;
    }

        acode_INIT();
}
uint encodeinit_r(code,clen)
uchar   code;
ushort  clen;
{
struct {        char    func;
                char    ret;
        uchar   mode;
        uchar   fine;
        uchar   gaso;
        uchar   yobi;
        ushort  mbitl;
} cmdp;
        return((int )cmdp.ret);
}

/*********************************************************
 *      画像データ符号化                        *
 *      in...   char *画像バッファポインタ              *
 *      in...   uchar *符号化方式               *
 *      in...   uchar *符号化終了フラグ         *
 *      out..   int 符号化データ数                      *
 *********************************************************/
```

保守／廃止

SR.C

```
uint encode(linebuf,code,endf)
uchar far*      linebuf;
uchar           code;
uchar           endf;
{
        comp_param.gptr = linebuf;

    if(endf!=ON) {
        switch(code) {
        case 0 :
                acode_ENCODE0();
        break;
        case 1 :
            if(1){
              wait_FILL();
              mdmIE2_sw(0);
              while(is_mdmIA2());
                _disable();
                acode_ENCODE1();
                _enable()+
              set_mdmDBUFF( get_mdmDBUFF() );
              mdmIE2_sw(1);
            }else{
                acode_ENCODE1();
            }
        break;
        case 2 :
                acode_ENCODE2();
        break;
        }
return(comp_param.clen1L);
    }else{
        acode_RTCGET();
return(comp_param.clen1L);
    }
}


ushort far*     guiji;
/*************************************************************
 *      送受信バッファを初期化する                           *
 *      in...   なし                                         *
 *      out..   なし                                         *
 *************************************************************/
void bufinit()
{
uint    i;
        for(i=0; i<BUFSIZE; i++) SR_info.SR_BUF[i]=0;
        SR_BTOP=SR_info.SR_BUF;
        SR_BBOT=SR_info.SR_BUF;
        SR_BTOP_G=0;
        SR_BBOT_G=0;
        SR_COUNT=0x00;
        SR_COUNT_add=0x00;
        SR_info.SR_ENDF=OFF;
        guiji = (ushort far*)0x40000000;
}

/*************************************************************
 *      符号化データを送信バッファに書き込む                 *
 *      in...   char *符号バッファポインタ                   *
 *      in...   ushort 符号化データ数                        *
 *      in...   uchar  書き込み終了フラグ                    *
 *      out..   なし                                         *
 *************************************************************/
bufwrite(fptr,fcnt,eflag)
char    *fptr;
ushort  fcnt;
uchar   eflag;
```

6

SR.C

```
{
register char   *ptr;
register uint    cnt;

        for(ptr=fptr,cnt=fcnt;cnt!=0;cnt--) {
                if(SR_BTOP==&SR_info.SR_ENDF) SR_BTOP=SR_info.SR_BUF;
                while(SR_COUNT==sizeof(SR_info.SR_BUF)){
                }
                *SR_BTOP++ = *ptr++;
                SR_COUNT++;
        }

        if(eflag==ON) SR_info.SR_ENDF = ON;
        else SR_info.SR_ENDF = OFF;

        return (0);
}

bufwrite_0(fptr,fcnt)
char far*       fptr;
ushort          fcnt;
{
register char far*      ptr;
register uint           cnt;

        SR_COUNT_add = 0;
        for(ptr=fptr,cnt=fcnt;cnt!=0;cnt--) {
                if( SR_BTOP == &SR_info.SR_ENDF ) SR_BTOP=SR_info.SR_BUF;
                while( (SR_COUNT+SR_COUNT_add) == sizeof(SR_info.SR_BUF) );
                *SR_BTOP++ = *ptr++;
                SR_COUNT_add++;
        }
        SR_COUNT += SR_COUNT_add;

        return (0);
}

fillwrite(cont)
unsigned short cont;
{
register unsigned short cnt;

        for(cnt=cont;cnt!=0;cnt--) {
                if( SR_BTOP == &SR_info.SR_ENDF ) SR_BTOP=SR_info.SR_BUF;
                while( SR_COUNT == sizeof(SR_info.SR_BUF) );
                _disable();
                *SR_BTOP++ = 0;
                SR_COUNT++;
                _enable();
        }

 }



bufwrite_1(fptr,fcnt)
char far*       fptr;
ushort          fcnt;
{
register char far*      ptr;
register uint           cnt;

        for(ptr=fptr,cnt=fcnt;cnt!=0;cnt--) {
                if( SR_BTOP == &SR_info.SR_ENDF ) SR_BTOP=SR_info.SR_BUF;
                while( (SR_COUNT+SR_COUNT_add) == sizeof(SR_info.SR_BUF) );
                *SR_BTOP++ = *ptr++;
                SR_COUNT_add++;
        }
```

7

SR.C

```
        SR_COUNT += SR_COUNT_add;

        return (0);
}
bufwrite_2(fptr,fcnt)
char far*       fptr;
ushort          fcnt;
{
register char far*      ptr;
register uint           cnt;

        SR_COUNT_add = 0;
        for(ptr=fptr,cnt=fcnt;cnt!=0;cnt--) {
                if( SR_BTOP == &SR_info.SR_ENDF ) SR_BTOP=SR_info.SR_BUF;
                while( (SR_COUNT+SR_COUNT_add) == sizeof(SR_info.SR_BUF) );
                *SR_BTOP++ = *ptr++;
                SR_COUNT_add++;
        }

        return (0);
}

/*********************************************************
 *      受信メインルーチン                               *
 *      in...   uchar 符号方式(MH=0,MR=1,MMR=2) *
 *      out..   なし                                     *
 *********************************************************/
ushort  BUFZ = 0x20;
uchar   LE_P_C[]={11,22};
uchar   LE_T_C[]={108,216};
Receive(code)
uchar   code;
{
char    ret;
uchar   line_err_P,line_err_T;
uchar   linen_prn,linen_dec,linen_cnt;
uchar   prn_cycle;
uint    kekka;
uchar   recv_end;
uchar   stat;
ulong   t;
uint    clen;
uint    clen_rest;
uchar   i;
uchar far*      img_p;

#define T_abort 5000+2500

        motorp(MOT_PR,ON);
        thermalp(ON);
        stwidth();

        decodeinit(code);
        clen_rest=0;
        clen=0;
        line_err_P=0;
        line_err_T=0;
        ret = 0;

        linen_prn =0;   linen_dec =0;   linen_cnt =0;
        prn_cycle =0;
        recv_end =0;
        timeset(&t,T_abort+3000);

        img_p = (uchar far*)(img_ptr(linen_prn));
        for(i=0; i<216; i++){
                *(img_p+i)=0x00;
        }
        imgwrite(linen_prn,size_line);
```

保守／廃止

SR.C

```
        while(1) {

          if(recv_end == 0){
            if(linen_cnt < 32){ /*32 is line buffer number**/

                if(clen_rest==0){
                    if( SR_COUNT >= BUFZ ){
                        clen =  bufread_R((char far*)comp_data,BUFZ);
                        if(clen!=0){
                                clen_rest=1;
                        }
                    }
                }else{
                  kekka=decode(img_ptr(linen_dec),code,clen);
                  stat=kekka;
                                clen_rest=kekka>>8;
                  if      (stat == 0x00){
                  }else if(stat == 0x10){
                        linen_cnt++;
                        linen_dec++;
                        line_err_P=0;
                        timeset(&t,T_abort);
                  }else if(stat == 0x01){
                                recv_end =1;
                  }else{
                        linebufR_dotted(linen_dec);
                        linen_cnt++;
                        linen_dec++;
                        line_err_P++;
                        line_err_T++;
                        if((line_err_P>LE_P_C[FMODE])
                        || (line_err_T>LE_T_C[FMODE])){
                                ret = -1;
                                recv_end =1;
                        }
                  }
                }
            }
          }

            if((linen_cnt != 0)
            && (str_end() != 0)){

              if(FMODE==0x00) {
                if(!(prn_cycle++ & B00000001)){
                        ltc();
                        motor();
                        stwidth();
                        str();
                }else{
                        motor();
                        stwidth();
                        str();
                    imgwrite(linen_prn,size_line);
                  linen_prn++;
                  linen_cnt--;
                  }
              }else{
                        ltc();
                        motor();
                        stwidth();
                        str();
                    imgwrite(linen_prn,size_line);
                  linen_prn++;
                  linen_cnt--;
              }
            }else if( recv_end != 0 ){

              if(FMODE==0x00) {
```

保守／廃止

SR.C

```
                        ltc();
                        motor();
                        stwidth();
                        str();
                        while( str_end() == 0 );
                        motor();
                        stwidth();
                        str();
                        while( str_end() == 0 );
                }else{
                        ltc();
                        motor();
                        stwidth();
                        str();
                        while( str_end() == 0 );
                }
        break;
            }

                if( timechk(&t) ){
                        ret = -2;
        break;
                }

                if( DANGER ){
                        ret = -3;
        break;
                }

        }


        thermalp(OFF);
        motorp(MOT_PR,OFF);

        if (ret == 0) {
                return (0);
        } else {
                return (-1);
        }
}

/*********************************************************
 *      ＣＥＰ復号化初期化                               *
 *      in...   uchar 符号方式(MH=0,MR=1,MMR=2) *
 *      out..   なし                                     *
 *********************************************************/

struct {
        uchar   K;
        uchar   dum0;
        ushort  OD_DT;
        uchar   OD_BT;
        uchar   dum1;
        ushort  CDBP;
        ushort  RVBP;
        ushort  RVBSZP;
        ushort  APBP;
        ushort  EOLFLG;
        ushort  sequ;
        ushort  TRNP_SEG;
        ushort  TRNP_OFS;
}expn_work;
struct{
        uchar far*      gptr;
        ushort          gcnt;
        uchar far*      cptr;
        ushort          clen;
        ushort          clen_rest;
```

10

SR.C

```
        uchar           Kval_m1;
        uchar           CODE;
        ushort          clen1L;
        ushort          RTCFLG;
        uchar           stat;
}expn_param;

        uchar   DecInited;

decodeinit(code)
uchar   code;
{
        expn_param.CODE = code;
      if(FMODE==0x00){
        expn_param.Kval_m1 =1;
      }else{
        expn_param.Kval_m1 =3;
      }
        expn_param.gcnt=216;
        expn_param.RTCFLG=0;

        expn_param.cptr = (uchar far*)comp_data;
        expn_param.clen = BUFZ;

        adecode_DECIN1();
}
decodeinit_r(code)
uchar   code;
{
struct {         char   func;
                 char   ret;
        uchar   mode;
        uchar   fine;
        uchar   gaso;
        uchar   yobi;
        ushort  mbit1;
} cmdp;
        return((int )cmdp.ret);
}

decodeinit_ec(code)
uchar   code;
{
struct{  char   func;
         char   ret;
         uchar  mode;
         uchar  fine;
         uchar  gaso;
         uchar  yobi;
         ushort mbit1;
} cmdp;

        cmdp.func = 0x00;
        cmdp.ret  = 0x00;
        cmdp.mode = code;
        cmdp.fine = 0x00;
        cmdp.gaso = 0x00;
        cmdp.mbit1= 0;

        ADECODE(&cmdp);

        DecInited = 1;
        return((int )cmdp.ret);
}

decodeinit_ec_r(code)
uchar   code;
{
struct{  char   func;
```

11

保守／廃止

SR.C

```
        char   ret;
        uchar  mode;
        uchar  fine;
        uchar  gaso;
        uchar  yobi;
        ushort mbitl;
} cmdp;

        return((int )cmdp.ret);
}

/*************************************************************
 *      １受信バッファ復号化                                 *
 *      in...   char *画像バッファポインタ                   *
 *      in...   uchar *符号化方式                            *
 *      out..   int  画像データ数                            *
 *               0 = 復号終了（ＲＴＣ検出）                 *
 *              -1 = ラインエラー                            *
 *              -2 = 符号データが何もない                    *
 *************************************************************/
decode(linebuf,code,clen)
uchar far*      linebuf;
uchar           code;
uint            clen;
{
        expn_param.gptr = linebuf;
        expn_param.cptr = (uchar far*)comp_data;
        expn_param.clen = clen;
        expn_param.stat = 0x00;

        switch(code) {
        case 0 :
                adecode_DECMH();
        break;
        case 1 :
                adecode_DECMR();
        break;
        case 2 :
                adecode_DECMMR();
        break;
        }

        if      ( expn_param.stat == 0x00 ){
                return((expn_param.clen_rest<<8) | 0x00);
        }else if( expn_param.stat == 0x10 ){
                return((expn_param.clen_rest<<8) | 0x10);
        }else if( expn_param.RTCFLG != 0 ){
                return((expn_param.clen_rest<<8) | 0x01);
        }else{
                return((expn_param.clen_rest<<8) | 0xff);
        }
}

/*************************************************************
 *      画像データをサーマルに転送                           *
 *      in...   char *バッファポインタ                       *
 *      in...   ushort  バッファサイズ                       *
 *      out..   なし                                         *
 *************************************************************/
void imgwrite(linen,length)
uchar   linen;
ushort  length;
{
uchar far*      ptr;
        ptr = (uchar far*)&(nama_data->arry[(linen % 32)][0]);
        img_output(ptr,length);
}

img_output(buf,len)
```

12

SR.C

```
uchar far*      buf;
ushort          len;
{
uchar far*      ptr;
register ushort cnt;

        for(ptr=buf,cnt=len;cnt!=0;cnt--) v55reg->TxB0_SIO0 = *ptr++;
}


void imgwrite_2(buf,len)
uchar far*      buf;
ushort          len;
{
static  uchar far*      ptr;
static  ushort          cnt;

        for(ptr=buf,cnt=len;cnt!=0;cnt--) v55reg->TxB0_SIO0 = *ptr++;
}

uchar far* img_ptr(linen)
uchar   linen;
{
uchar far*      ptr;


        ptr = (uchar far*)&(nama_data->arry[(linen % 32)][0]);
        return(ptr);
}

/*********************************
 *  エラーを示すパターン.        *
 *********************************/
linebufR_dotted(uchar linen)
{
register        i;
uchar far*      ptr;
        ptr=img_ptr(linen);
        for(i=0; i< 216; i++){
                *(ptr+i) = B10101010;
        }
}
/*************************************************************
 *      符号化データを受信バッファから読み出す               *
 *      in...   char *符号バッファポインタ                   *
 *      in...   ushort  符号バッファサイズ                   *
 *      out..   符号データ数,-1=データなし                   *
 *************************************************************/
bufread(fptr,fcnt)
char    *fptr;
ushort  fcnt;
{
register char   *ptr;
register uint    cnt;

        for(ptr=fptr,cnt=fcnt;cnt!=0;cnt--) {
                if(SR_BBOT==&SR_info.SR_ENDP) SR_BBOT=SR_info.SR_BUF;
                if(SR_COUNT==0x00) break;
                *ptr++ = *SR_BBOT++;
                SR_COUNT--;
        }

        if(cnt==fcnt) return(-1);

        return(fcnt-cnt);
}


bufread_R(fptr,fcnt)
```

保守／廃止

SR.C

```
char far*       fptr;
ushort          fcnt;
{
register uint   cnt;
ushort          ffcnt;
ushort          SR_COUNT_mm;

    SR_COUNT_mm = SR_COUNT;

    if( SR_COUNT_mm >= (fcnt) ){
        ffcnt = fcnt;
    }else{
        ffcnt = (SR_COUNT_mm>>1}<<1;
    }
    SR_BBOT_G += ffcnt;
    SR_COUNT -= ffcnt;
    return(ffcnt);

}

/*****************************************
 *      ダミーの白ラインを送信する       *
 *****************************************/

int     dummy_white(code)
uchar   code;
{
        static  uchar   fugo_buf[0x800];
        static  uchar   gazo_buf[216];
        uint            len;
        ushort  i;

        for (i = 0; i < 216; i++)
                gazo_buf[i] = 0;

        for (i = 0; i < 16; i++) {
                len = encode(gazo_buf, code, OFF);
                if (bufwrite(fugo_buf, len, OFF) != 0)
                        return (-1);
        }
        return (0);
}

/*************************************************
 *      ＲＴＣの後，ダミーデータを送信する       *
 *************************************************/

int     dummy_send()
{
        static  uchar   dmy_dat[] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};

        return (bufwrite(dmy_dat, sizeof(dmy_dat), ON));
}

extern  ushort  FILD_SIZE;
extern  uchar   RCP_DETC;
extern  uchar   EOF_DETC;
extern  uchar   FILD_NUM;
extern  uchar   PRAM_NUM;
extern  uchar   PRAM_STAT[0x20];
extern  uchar   PRAM_STAT_TMP[0x20];
extern struct   comp_ecm_data_s far *s_data1;
extern struct   comp_ecm_data_s far *s_data2;
extern  uchar           FMODE_ecm[2],CODE_ecm[2];
extern  struct  indx_ecmm_st    indx_ecmm;
extern  uchar   step_task[8][8];

extern  uchar   BLOCK_std;
extern  uchar   ecm_buf_ful;
```

14

保守／廃止

SR.C

```
extern  uchar   kamikomi_time;

/*********************************************************
 *      送信メインルーチン                                *
 *      in...   uchar 符号方式(MH=0,MR=1,MMR=2) *
 *      in...   ushort  1ライン符号化バイト数  *
 *      out..   なし                                      *
 *********************************************************/
task02_1_on()
{
    step_task[2][1]=1;
}
task02_1(uchar ctrl)
{
static  uchar   code;
static  uchar   fmode;
static  ushort  clen = 0;
static  uint    len;
static  uchar   i,j[27];
static  uchar   pagef;

    if(ctrl==0){
    }else{
    }

    switch( step_task[2][1] ){
    case 1:
        fmode = FMODE_ecm[(uchar)indx_ecmm.output];
        code  = CODE_ecm[(uchar)indx_ecmm.output];

        pagef=getpage();

        motorp(MOT_IR,ON);
        cisp(ON);
    if( BLOCK_std == 1 ){
        wait(200);
    }else{
        wait(1000);
    }

                indx_ecmm.input=0;

    if( BLOCK_std == 1 ){
        genko_ari(1);
        BLOCK_std = 1;
        imgread(linen,size_line);
    }else{
    }

    step_task[2][1]=2;
    break;

    case 2:
                task02_2_on();
    step_task[2][1]=3;
    case 3:
                task02_2();
        if(step_task[2][2] == 0){
      if( ecm_buf_ful == ON ){
    step_task[2][1]=14;
      }else{
    step_task[2][1]=4;
      }
        }
    break;

    case 4:
        cisp(OFF);
```

15

保守／廃止

SR.C

```
        if(STP==OFF && DANGER==OFF){
                buzzer(ON);
                winker(STARL, ON,  0);
    step_task[2][1]=5;
        }else{
    step_task[2][1]=0;
        }
    break;

    case 14:
        cisp(OFF);
    step_task[2][1]=0;
    break;

    case 5:
        kamikomi_time = 1;
        timeset(&Gobyo, 6000);
        Gostop = OFF;
        motorp(MOT_IR,ON);
    step_task[2][1]=6;
    case 6:
        if(!timechk(&Gobyo) ){

                if( STP == OFF ){
                        genkochk_2();
                        if(keyin(ON) == PINES) setfine(ON);
                }else{
    step_task[2][1]=7;
                }
        }else{
    step_task[2][1]=7;
        }
    break;
    case 7:
        if( STP == OFF ){
                while( !genkochk_2() );
        }

        winker(STARL, ON,  3);
        buzzer(OFF);
        motorp(MOT_IR,OFF);
        kamikomi_time = 0;
    step_task[2][1]=0;
    break;
    }
return (0);
}

/*********************************************************
 *      受信メインルーチンＥＣＭ                         *
 *      in...   無し                                     *
 *      out..   なし                                     *
 *      各種パラメータ設定済み。モデム受信体制設定済み。 *
 *      割り込み処理ルーチン設定済み。                   *
 *********************************************************/


Receive_ecm()
{
ulong   t1,t2;
char    ret;

        RCP_DETC = 0;

        timeset(&t1,18000);
        timeset(&t2,36000);
        while(1) {
                if( RCP_DETC != 0 )
                        break;
```

16

SR.C

```
                if( !mdm_cdet() || mdm_Qchk() ){
                    if( timechk(&t1) ){
                        break;
                    }
                }else{
                        timeset(&t1,18000);
                }
                if( mdm_300det() ){
                        break;
                }
                if( EOF_DETC == 0 ){
                    if( timechk(&t2) ){
                        break;
                    }
                }else{
                        timeset(&t2,36000);
                        EOF_DETC = 0;
                }
        }

        if (ret == 0) {
                return (0);
        } else {
                return (-1);
        }
}



receive_frame_count()
{
uchar   i,j;
ushort  retval;
        retval = 0;
        for(i=0; i<0x20; i++){
                switch( FRAM_STAT_TMP[i] ){
                case 0x00:
                                        retval += 8;
                break;
                case 0xff:
                break;
                default:
                        for(j=0; j<8; j++){
                                if(!( (1<<j) & FRAM_STAT_TMP[i] )){
                                        retval++;
                                }
                        }
                break;
                }
        }
        retval--;
return(retval);
}

receive_ecm_print(code)
uchar   code;
{
uchar   i;
char            ret;
uchar   line_error;

        ecm_write(code);

        motorp(MOT_PR,OFF);

        thermalp(OFF);

        if (ret == 0) {
```

17

SR.C

```
                return (0);
        } else {
                return (-1);
        }
}

task01_1_on()
{
        step_task[1][1]=1;
}

task01_1()
{
    switch( step_task[1][1] ){
    case 1:
        motorp(MOT_PR,ON);
        thermalp(ON);
        task01_2_on();
    step_task[1][1] = 2;
    case 2:
        task01_2();
        if( step_task[1][2] == 0 ){
    step_task[1][1] = 3;
        }
    break;

    case 3:
        motorp(MOT_PR,OFF);
        thermalp(OFF);
    step_task[1][1] = 0;
    break;
    }
}

far AMRCVI_ECM(uchar data)
{
    if( indx_ecmm.input == 0 ){
        s_data1 -> arry[( (ushort)FILD_SIZE*(uchar)FRAM_NUM )
                            + (uchar)FILD_NUM          ] = data;
    }else{
        s_data2 -> arry[( (ushort)FILD_SIZE*(uchar)FRAM_NUM )
                            + (uchar)FILD_NUM          ] = data;
    }
}

far AMSNDI_ECM()
{
uchar data;
    if( indx_ecmm.output == 0 ){
        data
        = s_data1 -> arry[( (ushort)FILD_SIZE*(uchar)FRAM_NUM )
                              + (uchar)FILD_NUM               ];
    }else{
        data
        = s_data2 -> arry[( (ushort)FILD_SIZE*(uchar)FRAM_NUM )
                              + (uchar)FILD_NUM               ];
    }
return(data);
}

ushort  index_G3T;
far AMRCVI_G3T(ushort data)
{
        if(( (uchar)(data>>8) == 0 )
         &&( index_G3T != 0 )){
                do{
        s_data1 -> arry[ index_G3T++ ] = 0;
                s_data1 -> arry[ index_G3T+0xf ] = 0;
                }while( (index_G3T % 0x10) != 0 );
```

18

SR.C

```
        }
        s_data1 -> arry[ index_G3T++ ] = (uchar)(data>>0);
                s_data1 -> arry[ index_G3T+0xf ] = 0;
        s_data1 -> arry[ index_G3T++ ] = (uchar)(data>>8);
                s_data1 -> arry[ index_G3T+0xf ] = 0;
}

far INIT_G3T()
{
        index_G3T = 0;
}
```

## A.11　ECMSR.C

ECMSR.C

```
/************************************************
 ************************************************/

/*      2進数ファイル  +**/
#include        "constbin.h"
/*      #328ファイル+**/
#include        "hd329.h"
/*      T30の頭ファイル       **/
#include        "t30.h"

/*Functions**/
uchar far*      img_ptr(uchar);

/*Constants**/
#define ECMBUFSIZE      0x00010000

/*Values  &  Pointers**/
extern struct   nama_data_s far *nama_data;     /**画像生データ**/
extern struct   comp_ecm_data_s far *s_data1;   /*最も外で定義する**/
extern struct   comp_ecm_data_s far *s_data2;   /*最も外で定義する**/
extern  uchar   FMODE;          /* ファイン／ノーマルフラグ **/
extern  uchar           FMODE_ecm[2],CODE_ecm[2];
extern  uchar           TERM_ecm[2];
extern  struct  indx_ecmm_st    indx_ecmm;

extern  uchar   DANGER;         /* このフラグが立ったら危険 **/
extern  uchar   STP;            /* STP FLAG **/
ushort  Gobyo;                  /* 5 sec **/
uchar   Gostop;
static  uchar   end;
        uchar   ecm_buf_ful;
static  uchar   nomore;
static  char    storeflag;
        char    fugobuf[1024];
        ushort  size_fugobuf = 1024;
char            store[1024];
ushort          j_store;

ulong   ecmsr_count;    /* バッファカウンタ **/
ulong   ecmsr_total;
uchar   rtc_det_ecm;    /* RTC検出フラグ+**/
extern  uchar   PAGE;   /* ページ数カウント **/
extern  uchar   BLOCK;  /* ブロック数カウント **/

uchar           BLOCK_std;      /* ブロック数 **/
uchar           ERR_flag;       /* エラーフラグ **/
uchar           ERR_trap;       /* エラートラップ **/

sramtest()
{

        ecm_copystart();
        ecm_copyread();
        ecm_copyend();
}

/************************************************
 *      コピーする時のいろんな処理              *
 *      のルーチン。                            *
 ************************************************/
ecm_copystart()
{
        readtime();
        setfdate(fdate);

        cisp(ON);
        motorp(MOT_IR,ON);

        wait(100);
```

1

保守／廃止

ECMSR.C

```
        img_input();
        wait(100);
}
ecm_copystart2()
{
        clrthermal();
        motorp(MOT_PR,ON);
        thermalp(ON);

        line();
        feedvarlen(MOT_PR,5);

        feedvarlen(MOT_IR,5);
        thermalp(ON);
        motorp(MOT_PR,ON);
}

/************************************************
 *      コピーエンド時のいろんな処理の          *
 *      ルーチン                                *
 ************************************************/
ecm_copyend()
{
        motorp(MOT_IR,OFF);
        cisp(OFF);
}
ecm_copyend2()
{
        motorp(MOT_PR,OFF);
        thermalp(OFF);
        if(rollchk()){
                endfeed();
                return(-1);
                }
        if(coverchk())return(-1);

        feedvarlen(MOT_PR,5);
        line();
        feedvarlen(MOT_PR,5);
        line_count_0();
        feedvarlen(MOT_PR,line_count_end_get());

        endfeed();
        setfine(OFF);
}

/************************************************
 *      ＥＣＭ＿ＣＯＰＹ（）                   *
 *      擬似Ｓ－ＲＡＭへ一旦符号化データを      *
 *      格納してからコピーする                  *
 *      in....無し                              *
 *      out...無し                              *
 ************************************************/

ecm_copyread()
{
uchar   code_cpy;

        end=nomore=OFF;
        genko_ari(1);
        storeflag=OFF;
        stwidth();
        indx_ecmm.input  = 0;
        indx_ecmm.output = 0;
        FMODE = 1;
        code_cpy = 1;

        BLOCK_std = 1;
        ecm_read(code_cpy);
```

2

保守／廃止

ECMSR.C

```
        indx_ecmm.input++;
        BLOCK_std++;
        if(end==OFF){
                ecm_read(code_cpy);
                indx_ecmm.input++;
        }
}

ecm_copywrite()
{
uchar   code_cpy;

        end=nomore=OFF;
        genko_ari(1);
        storeflag=OFF;
        stwidth();
        indx_ecmm.input  = 0;
        indx_ecmm.output = 0;
        FMODE = 1;
        code_cpy = 1;

        BLOCK_std = 1;
        ecm_write(code_cpy);
        indx_ecmm.output++;
        BLOCK_std++;
        if(nomore==OFF){
                ecm_write(code_cpy);
                indx_ecmm.output++;
        }
}




/************************************************
 *      ＥＣＭＲＥＡＤ（）                     *
 *      符号化されたデータを擬似Ｓ－ＲＡＭへ    *
 *      読み込む                                *
 *      in.....uchar    code                    *
 *                符号化方式                    *
 *      out....無し                             *
 ************************************************/
extern  ushort  size_line;
ecm_read(code)
uchar   code;
{
static  ushort  len;
static  ushort  results;
ushort          i;
static  ushort  j;
uchar   linen_imr,linen_enc,linen_cnt;

        motorp(MOT_PR,OFF);
        motorp(MOT_IR,ON);
        ecmbufinit();
        if( BLOCK_std == 1 ){
                encodeinit(code,0);
                storeflag=OFF;
                linen_imr =0;   linen_enc =0;   linen_cnt =0;
        }else{
                encodeinit_r(code,0);
        }
        results=0x00000000;
        end=OFF;


        while(1) {
                if(!storeflag){
```

保守／廃止

ECMSR.C

```
                        imgread(linen_imr,size_line);
                        if(end) len=encode(img_ptr(linen_enc),fugobuf,code,ON);
                        else    len=encode(img_ptr(linen_enc),fugobuf,code,OFF);
                        results = ecmbufwrite_2(fugobuf,len,OFF);
                }else{
                        results = ecmbufwrite_2(store,j,OFF);
                        storeflag=OFF;
                }

                if(results){
                        ecm_buf_ful=ON;
                        for(j=0,i=0;i<results;i++,j++){
                                store[j]=fugobuf[len-results+i];
                        }
                        storeflag=ON;
                        motorp(MOT_IR,OFF);
                        return(-1);
                }else{
                        ecm_buf_ful=OFF;
                }

                motor();
                if(end==ON){
                        motorp(MOT_IR,OFF);
                        return(0);
                }
                if(!chksens()){
                        end=ON;
                }
        }


        motorp(MOT_IR,OFF);
        return(0);
}

extern  uchar   step_task[8][8];
task02_2_on()
{
    step_task[2][2]=1;
}
task02_2()
{
static  uchar   code;
static  uchar   fmode;
static  ushort  len;
static  ushort  results;
ushort          i;
uchar   linen_imr,linen_enc,linen_cnt;

    switch( step_task[2][2] ){
    case 1:
        code  = CODE_ecm[(uchar)indx_ecmm.input];
        fmode = FMODE_ecm[(uchar)indx_ecmm.input];
        motorp(MOT_IR,ON);

        ecmbufinit();
    if( BLOCK_std == 1 ){
        encodeinit(code,0);
        BLOCK_std = 2;
                storeflag=OFF;
        linen_imr =0;   linen_enc =0;   linen_cnt =0;
        end=OFF;
    }else{
        encodeinit_r(code,0);
    }
    step_task[2][2]=2;
    break;
```

保守／廃止

ECMSR.C

```
case 2:
    if(end==OFF){
            if(!storeflag){
                    imgread(linen_imr,size_line);
                    len=encode(img_ptr(linen_enc),fugobuf,code,OFF);
                    results = ecmbufwrite_2(fugobuf,len,OFF);
            }else{
                    results = ecmbufwrite_2(store,j_store,OFF);
                    storeflag=OFF;
            }
          if(results){
                    ecm_buf_ful=ON;
                    for(j_store=0,i=0;i<results;i++,j_store++){
                            store[j_store]=fugobuf[len-results+i];
                    }
                    storeflag=ON;
step_task[2][2]=11;
break;
          }else{
                    ecm_buf_ful=OFF;
            motor();
            if(fmode==0x00){
step_task[2][2]=3;
            }
            if(!chksens()){
                    end=ON;
            }
          }
    }else{
            if(!storeflag){
                    imgread(linen_imr,size_line);
                    len=encode(img_ptr(linen_enc),fugobuf,code,ON);
                    results = ecmbufwrite_2(fugobuf,len,OFF);
            }else{
                    results = ecmbufwrite_2(store,j_store,OFF);
                    storeflag=OFF;
            }
          if(results){
                    ecm_buf_ful=ON;
                    for(j_store=0,i=0;i<results;i++,j_store++){
                            store[j_store]=fugobuf[len-results+i];
                    }
                    storeflag=ON;
                    TERM_ecm[(uchar)indx_ecmm.input] = OFF;
step_task[2][2]=11;
break;
          }else{
                    ecm_buf_ful=OFF;
            motor();
                if( storeflag == OFF ){
                    TERM_ecm[(uchar)indx_ecmm.input] = ON;
                }else{
                    TERM_ecm[(uchar)indx_ecmm.input] = OFF;
                }
                    motorp(MOT_IR,OFF);
step_task[2][2]=99;
          }
    }
break;

case 3:
            motor();
step_task[2][2]=2;
break;

case 10:
            motor();
step_task[2][2]=11;
break;
```

5

保守／廃止

ECMSR.C

```
    case 11:
                        motorp(MOT_IR,OFF);
    step_task[2][2]=0;
    break;

    case 99:
                ecmsr_total = ecmsr_count;
        motorp(MOT_IR,OFF);
    step_task[2][2]=0;
    break;
    }
}

ecmbufinit()
{
        ecmsr_count = 0x00000000;
        ecmsr_total = 0xffffffff;
}


/************************************************
 *      符号化データを擬似Ｓ－ＲＡＭ            *
 *      へ書き込む。                            *
 *      データが満タンになったら、              *
 *      入り切らなかった数を返す        *
 ************************************************/

ecmbufwrite(fptr,fcnt,eflag)
char    *fptr;
ushort  fcnt;
uchar   eflag;
{
register char   *ptr;
register ushort cnt;

        for(ptr=fptr,cnt=fcnt;cnt!=0;ecmsr_count++,cnt--) {
                if(ecmsr_count==ECMBUFSIZE){
                        return(cnt);
                }else{
                        s_data1 -> arry[(ushort)ecmsr_count]=*ptr++;
                }
        }
        return(0);
}

ecmbufwrite_2(fptr,fcnt,eflag)
char    *fptr;
ushort  fcnt;
uchar   eflag;
{
register char   *ptr;
register ushort cnt;
    if( indx_ecmm.input == 0 ){
        for(ptr=fptr,cnt=fcnt;cnt!=0;cnt--) {
                if(ecmsr_count==ECMBUFSIZE){
                        return(cnt);
                }else{
                        s_data1 -> arry[(ushort)ecmsr_count]=*ptr++;
                        ecmsr_count++;
                }
        }
    }else{
        for(ptr=fptr,cnt=fcnt;cnt!=0;cnt--) {
                if(ecmsr_count==ECMBUFSIZE){
                        return(cnt);
                }else{
                        s_data2 -> arry[(ushort)ecmsr_count]=*ptr++;
                        ecmsr_count++;
```

6

保守／廃止

ECMSR.C

```
                }
        }
    }
        return(0);
}


ecm_write(code)
uchar   code;
{
uchar   i;
char            ret;
uchar           wait;
uchar   linen_imr,linen_enc,linen_cnt;

        ecmbufinit();
        linen_imr =0;   linen_enc =0;   linen_cnt =0;
    if( BLOCK_std == 1 ){
        decodeinit(code);
    }else{
        decodeinit_r(code);
    }
        motorp(MOT_PR,ON);
        thermalp(ON);
        wait = 1;
        ERR_flag = 0;
        ERR_trap = 0;

        while(1){

                imgwrite(linen_enc,size_line);

                    if(wait == 0){
                        ltc_str();
                        motor();
                    }

                while(1){
                                ERR_flag = 0;
                        ret=ecmdecode(img_ptr(linen_enc),code);
                        if(ret==0){
                                 nomore=ON;
                                 motorp(MOT_PR,OFF);
                                 thermalp(OFF);
                                 return(0);
                        }else if(ret==-2){
                                return(-2);
                        }else if(ret!=-1){
                                break;
                        }else{
                                ERR_flag = 1;
                                ERR_trap |= 1;
                                linebuf_dotted(0);
                                break;
                        }
                }

                if(FMODE==0x00) {
                    if(wait == 0){
                        str();
                        motor();
                    }
                }
                wait = 0;
        }
}

linebuf_dotted(uchar ctrl)
{
```

ECMSR.C

```
register        i;
uchar far*      ip;
        ip=(uchar far*)img_ptr(0);
        for(i=0; i< size_line; i++){
                *(ip+i) = B10101010;
        }
        if(ctrl!=0)ltc_str();
}
linebuf2_dotted(uchar ctrl
               ,uchar far* ptr)
{
register        i;
        for(i=0; i< 216; i++){
                *(ptr+i) = B10101010;
        }
}


task01_2_on()
{
        step_task[1][2]=1;
}
task01_2()
{
static  char    ret;
static  char    code;
static  uchar   sub_step;
static  struct{ unsigned        indx0:3;
                unsigned        indx1:3;
                unsigned        flag :8;
        }linbf;

    switch( step_task[1][2] ){
    case 1:
                                rtc_det_ecm = 0;
        ecmbufinit();
        code = CODE_ecm[(uchar)indx_ecmm.output];
      if( BLOCK == 1 ){
        decodeinit_ec(code);
      }else{
        decodeinit_ec_r(code);
      }
    sub_step = 1;
      if( BLOCK == 1 ){
        linbf.indx0 = 0;
        linbf.indx1 = 0;
      }
        linbf.flag  = 0;
    step_task[1][2] = 2;
    break;

    case 2:
                if( ( linbf.flag & ( 1 << linbf.indx0 ) ) == 0 ){
                        ret=ecmdecode(img_ptr(linbf.indx0),code);
                        if(ret==0){
                                rtc_det_ecm = 1;
                                nomore=ON;
    step_task[1][2] = 0xff;
                        }else if(ret==-2){
    step_task[1][2] = 0xff;
                        }else if(ret!=-1){
    step_task[1][2] = 3;
                        }else{
                                linebuf2_dotted(0,img_ptr(linbf.indx0));
    step_task[1][2] = 3;
                        }
                }
    break;
```

保守／廃止

ECMSR.C

```
    case 3:
                linbf.flag |= ( 1 << linbf.indx0 );
                linbf.indx0++;
    step_task[1][2] = 2;
    break;
    }

    if( linbf.flag != 0 )
    switch( sub_step ){
    case 1:
                imgwrite_2(img_ptr(linbf.indx1),size_line);
    sub_step = 2;

    case 2:
        if( PMODE_ecm[(uchar)indx_ecmm.output] == 0x00 ) {
                        if( str_end() != 0 ){
                            ltc();
                            motor();
                            str();
    sub_step = 3;
                        }
        }else{
                        if( str_end() != 0 ){
                            ltc();
                            motor();
                            str();
                linbf.flag &= ~( 1 << linbf.indx1 );
                linbf.indx1++;
    sub_step = 1;
                        }
        }
    break;

    case 3:
                        if( str_end() != 0 ){
                            motor();
                            str();
                linbf.flag &= ~( 1 << linbf.indx1 );
                linbf.indx1++;
    sub_step = 1;
                        }
    break;
    }

  if(( step_task[1][2] == 0xff ) && ( linbf.flag == 0 )){
    step_task[1][2] = 0;
  }
}



extern  uchar   DecInited;

ecmdecode(linebuf,code)
char    *linebuf;
uchar   code;
{
static  ushort  t;
static  int     flen;
static struct { char    func;
                char    ret;
                char    *fptr;
                short   fcnt;
                char    *gptr;
                short   gcnt;
} cmdp;

        if (DecInited != 0) {
                DecInited = 0;
```

保守／廃止

ECMSR.C

```
                cmdp.gcnt = 0;
                cmdp.fcnt = 0;
        }

        switch(code) {
                case 0 : cmdp.func = 0x01; break;
                case 1 : cmdp.func = 0x03; break;
                case 2 : cmdp.func = 0x05; break;
        }

        if(cmdp.gcnt==0x00) {
                cmdp.gptr = linebuf;
                cmdp.gcnt = 0x00;
        }

        while(1) {


                if(cmdp.fcnt==0) {
                        while(1) {
                                flen=ecmbufread_2(fugobuf,size_fugobuf);
                                if(flen==-2){
                                        return(-2);
                                }
                                else if(flen!=-1){
                                        break;
                                }
                                else{
                                }
                        }
                        cmdp.fptr = fugobuf;
                        cmdp.fcnt = flen;
                }

                cmdp.ret  = 0x00;
                ADECODE(&cmdp);

                if(cmdp.ret!=0x20) break;

        }

        if(cmdp.ret==0x10) {
                cmdp.gcnt = 0x00;
                return(-1);
        }

        if(cmdp.ret==0x30) {
                 cmdp.fcnt=cmdp.gcnt=0x00;
                 return(0);
        }

        cmdp.gcnt = 0x00;

        return(216);
}


ecmbufread(fptr,fcnt)
char    *fptr;
ushort  fcnt;
{
register char   *ptr;
register uint    cnt;

        for(ptr=fptr,cnt=fcnt;cnt!=0;cnt--,ecmsr_count++) {
                if(ecmsr_count==ECMBUFSIZE) {
                        return(-2);
                }
                *ptr++ = s_data1 -> arry[(ushort)ecmsr_count];
```

10

保守／廃止

ECMSR.C

```
        }

        if(cnt==fcnt) return(-1);

        return(fcnt-cnt);
}

ecmbufread_2(fptr,fcnt)
uchar   *fptr;
ushort   fcnt;
{
register uchar  *ptr;
register ushort  cnt;

        if(ecmsr_count>=ECMBUFSIZE) return(-2);

    if( indx_ecmm.output == 0 ){
        for(ptr=fptr,cnt=fcnt;cnt!=0;cnt--,ecmsr_count++) {
                if(ecmsr_count==ECMBUFSIZE)     break;
                *ptr++ = s_data1 -> arry[(ushort)ecmsr_count];
        }
    }else{
        for(ptr=fptr,cnt=fcnt;cnt!=0;cnt--,ecmsr_count++) {
                if(ecmsr_count==ECMBUFSIZE)     break;
                *ptr++ = s_data2 -> arry[(ushort)ecmsr_count];
        }
    }

        if(cnt==fcnt) return(-1);

        return(fcnt-cnt);
}

uchar   int_vect[0x200];
ushort  int_vect_id;
tess_int_trace(ushort vect)
{
    if(int_vect_id!=0x1ff){
        int_vect[(int_vect_id++) & 0x1ff] = (uchar)vect;
    }
}
```

11

## A.12　TASK00.C

TASK00.C

```
/* ******************************************************** */
/*      "ＴＡＳＫ００"                                      */
/* ******************************************************** */
/* ******************************************************** */

/*      ２進数ファイル  +*/
#include        "constbin.h"
/*      ＃３２８ファイル+*/
#include        "hd329.h"
/*      Ｔ３０のファイル+*/
#include        "t30.h"

        uchar   step_task[8][8];
        uchar   hold_task[8][2];

task00()
{
uchar   i,j,k;
        for(i=0; i<8; i++){
                for(j=0; j<8; j++){
                        step_task[i][j] = 0;
                }
                for(k=0; k<2; k++){
                        hold_task[i][k] = 0;
                }
        }
}

task00_exist(uchar ctrl)
{
uchar   i;
        for(i=0; i<8; i++){
                if( step_task[i][0] != 0 )
                        return(1);
        }
                        return(0);
}
```

1

## A.13　TASK01.C

TASK01.C

```
/* ******************************************************** */
/*      “ＴＡＳＫ０１”・・・タスクＮｏ．１
       プリントタスク                                      */
/* ******************************************************** */
/* ******************************************************** */

/*      ２進数ファイル  +*/
#include        "constbin.h"
/*      ＃３２８ファイル+*/
#include        "hd329.h"
/*      Ｔ３０ファイル  +*/
#include        "t30.h"

extern  uchar   step_task[8][8];
extern  uchar   hold_task[8][2];

extern  uchar   PAGE;   /* ページ数カウント */
extern  uchar   BLOCK;  /* ブロック数カウント */
extern  uchar   RSF;            /* 受信 or 送信フラグ */
extern  uchar   rtc_det_ecm;            /* ＲＴＣ検出フラグ+*/

/*********************************/
/*タスクＮｏ１：プリンタ. +*/
/*********************************/
task01_on()
{
    if( hold_task[1][0] != 0 )
return(0);

    _disable();
    if( step_task[1][0] == 0 ){
        chng_indx_ecmm();
        step_task[1][0] = 1;
    }else{
        hold_task[1][1] = hold_task[1][0];
        hold_task[1][0] = 1;
    }
    _enable();
return(1);
}

task01_on_11()
{
    if( hold_task[1][1] != 0 )
return(0);

    _disable();
  if( hold_task[1][0] != 0 ){
        hold_task[1][1] = 11;
  }else{
    if( step_task[1][0] == 0 ){
        step_task[1][0] = 11;
    }else{
        hold_task[1][1] = hold_task[1][0];
        hold_task[1][0] = 11;
    }
  }
    _enable();
return(1);
}

task01_on_21()
{
    if( hold_task[1][0] != 0 )
return(0);

    _disable();
    if( step_task[1][0] == 0 ){
        chng_indx_ecmm();
```

1

保守／廃止

TASK01.C

```
        step_task[1][0] = 21;
    }else{
        hold_task[1][1] = hold_task[1][0];
        hold_task[1][0] = 21;
    }
    _enable();
return(1);
}

task01_on_31()
{
    if( hold_task[1][0] != 0 )
return(0);

    _disable();
    if( step_task[1][0] == 0 ){
        step_task[1][0] = 31;
    }else{
        hold_task[1][1] = hold_task[1][0];
        hold_task[1][0] = 31;
    }
    _enable();
return(1);
}

task01_on_test()
{
        step_task[1][0] = 0xf0;
}

task01_off()
{
        step_task[1][0] = 0;
        step_task[1][1] = 0;
        step_task[1][2] = 0;
        step_task[1][3] = 0;
}

task01()
{
static  ushort  gegege;
static  ushort  rest;

    switch( step_task[1][0] ){
    case 1:
                        task01_1_on();
    step_task[1][0] = 2;
    case 2:
                task01_1();
      if( step_task[1][1] == 0 ){
    step_task[1][0] = 3;
        if( rtc_det_ecm == 0 ){
    step_task[1][0] = 0xff;
        }
      }
    break;

    case 3:
        task01_3_on();
    step_task[1][0] = 4;
    case 4:
        task01_3(MOT_PR,5);
        if( step_task[1][3] == 0 ){
    step_task[1][0] = 5;
        }
    break;

    case 5:
        task01_3_on();
```

2

保守／廃止

TASK01.C

```
    step_task[1][0] = 6;
    case 6:
        task01_3(MOT_PR,5);
        if( step_task[1][3] == 0 ){
                                    line_count_0();
    step_task[1][0] = 0xff;
        }
    break;

    case 11:
        if( ( rest = line_count_judg() ) ){
    step_task[1][0] = 12;
        }else{
    step_task[1][0] = 0xff;
        }
    break;

    case 12:
        task01_4_on();
    step_task[1][0] = 13;
    case 13:
        task01_4();
        if( step_task[1][4] == 0 ){
        line_count_0();
    step_task[1][0] = 14;
        }
    break;

    case 14:
        task01_3_on();
    step_task[1][0] = 15;
    case 15:
        task01_3(MOT_PR,rest);
        if( step_task[1][3] == 0 ){
    step_task[1][0] = 0xff;
        }
    break;

    case 21:
        task01_4_on();
    step_task[1][0] = 22;
    case 22:
        task01_4();
        if( step_task[1][4] == 0 ){
    step_task[1][0] = 23;
        }
    break;

    case 23:
        task01_3_on();
    step_task[1][0] = 24;
    case 24:
        task01_3(MOT_PR,5);
        if( step_task[1][3] == 0 ){
    step_task[1][0] = 1;
        }
    break;

    case 31:
        task01_4_on();
    step_task[1][0] = 32;
    case 32:
        task01_4();
        if( step_task[1][4] == 0 ){
    step_task[1][0] = 33;
        }
    break;

    case 33:
```

3

保守／廃止

TASK01.C

```
        task01_3_on();
    step_task[1][0] = 34;
    case 34:
        task01_3(MOT_PR,5);
        if( step_task[1][3] == 0 ){
    step_task[1][0] = 0;
        }
    break;

    case 0xf0:
        gegege = 0x0100;
        while( gegege-- != 0 );
    break;

    case 0xff:
    if( hold_task[1][0] != 0 ){
      if(( hold_task[1][0] == 1 )
       ||( hold_task[1][0] == 21 )){
        chng_indx_ecmm();
      }
    step_task[1][0] = hold_task[1][0];
        hold_task[1][0] = hold_task[1][1];
        hold_task[1][1] = 0;
    }else{
    step_task[1][0] = 0;
    }
        ende_indx_ecmm();
    break;

    default:
    break;
    }
}
```

4

## A.14　TASK02.C

TASK02.C

```
/* ******************************************************** **/
/*      "ＴＡＳＫ０２" ・・・タスクＮｏ．２                  */
/* ******************************************************** **/
/* ******************************************************** **/

/*      ２進数ファイル  **/
#include        "constbin.h"
/*      ＃３２８ファイル**/
#include        "hd329.h"
/*      Ｔ３０のファイル**/
#include        "t30.h"

extern  uchar   step_task[8][8];        /*タクストリガ８＊８個分 **/
extern  uchar   hold_task[8][2];        /*タクストリガ８保留分 **/

extern  uchar   PAGE;   /* ページ数カウント **/
extern  uchar   BLOCK;  /* ブロック数カウント **/
extern  uchar   RSF;            /* 受信 or 送信フラグ **/
extern  uchar   rtc_det_ecm;            /* ＲＴＣ検出フラグ **/

/***********************************/
/*タスクＮｏ１：プリンタ．  **/
/***********************************/
task02_on()
{
    if( hold_task[2][0] != 0 )
return(0);

    _disable();
    if( step_task[2][0] == 0 ){
        step_task[2][0] = 1;
    }else{
        hold_task[2][1] = hold_task[2][0];
        hold_task[2][0] = 1;
    }
    _enable();
return(1);
}

task02_on_11()
{
    if( hold_task[2][1] != 0 )
return(0);

    _disable();
  if( hold_task[2][0] != 0 ){
        hold_task[2][1] = 11;
  }else{
    if( step_task[2][0] == 0 ){
        step_task[2][0] = 11;
    }else{
        hold_task[2][1] = hold_task[2][0];
        hold_task[2][0] = 11;
    }
  }
    _enable();
return(1);
}

task02_on_21()
{
    if( hold_task[2][1] != 0 )
return(0);

    _disable();
  if( hold_task[2][0] != 0 ){
        hold_task[2][1] = 21;
  }else{
    if( step_task[2][0] == 0 ){
```

1

TASK02.C

```
        step_task[2][0] = 21;
    }else{
        hold_task[2][1] = hold_task[2][0];
        hold_task[2][0] = 21;
    }
  }
    _enable();
return(1);
}
task02_off()
{
        step_task[2][0] = 0;
        step_task[2][1] = 0;
        step_task[2][2] = 0;
        step_task[2][3] = 0;
}
task02()
{
static  ushort  gegege;
static  ushort  rest;

    switch( step_task[2][0] ){
    case 1:
                task02_1_on();
    step_task[2][0] = 2;
    case 2:
                task02_1(0);
      if( step_task[2][1] == 0 ){
    step_task[2][0] = 0xff;
      }
    break;

    case 11:
    break;

    case 21:
                task02_4_on();
    step_task[2][0] = 22;
    case 22:
                task02_4();
      if( step_task[2][4] == 0 ){
    step_task[2][0] = 0xff;
      }
    break;

    case 0xf0:
        gegege = 0x0100;
        while( gegege-- != 0 );
    break;

    case 0xff:
    if( hold_task[2][0] != 0 ){
      if(( hold_task[2][0] == 1 )
       ||( hold_task[2][0] == 11 )
       ||( hold_task[2][0] == 21 )){
      }
    step_task[2][0] = hold_task[2][0];
        hold_task[2][0] = hold_task[2][1];
        hold_task[2][1] = 0;
    }else{
    step_task[2][0] = 0;
    }
    break;

    default:
    break;
    }
}
```

2

## A.15　TASK03.C

TASK03.C

```
/* ************************************************** **/
/*      “ＴＡＳＫ０３”・・・タスクＮｏ．３
       デモタスク                          **/
/* ************************************************** **/
/* 修正                                          **/
/* 年月日と内容                                        **/
/*  199X/XX/XX
       ＸＸＸＸＸＸＸＸ
       **/
/* ************************************************** **/

/*      ２進数ファイル  **/
#include       "constbin.h"
/*      #３２８ファイル**/
#include       "hd329.h"
/*      Ｖ５５ＰＩ内部データ領域・マップ **/
#include       "indat55.h"

extern  uchar   step_task[8][8];        /*タクストリガ８＊８個分 **/
extern  uchar   hold_task[8][2];        /*タクストリガ８保留分 **/

/*********************************/
/*タスクＮｏ３：デモ．  **/
/*********************************/
task03_on()
{
        ss_test();
}
task03()
{
static  uchar   devide;
    if( (devide++ % 2) == 0 ){
        task_ss_ope();
    }
}

extern  struct  v55pi_regs far  *v55reg;
#define C_R     0x00
#define E_O_P   0x0a
#define MSGLNGTH        0x48
static  uchar   *S_DATAP;

/************************************************
 *      シリアルポートからメッセージを出力      *
 *      ＰＣ９８０１側は、ターミナルモード      *
 ************************************************/


unsigned char   *page_98[4][24];
unsigned char   chg_s_to_r;

ss_msginit()
{
page_98[0][0]="3           μＰＤ７０４３３";
page_98[0][1]="3            （Ｖ５５ＰＩ）";
page_98[0][2]="3  １６ビット・マイクロコンピュータ  ";
page_98[0][3]="2  ";
page_98[0][4]="2    μＰＤ７０４３３（別名称Ｖ５５ＰＩ）は、１６ビットＣＰＵ、ＲＡＭ、シリア
";
page_98[0][5]="2  ル・インターフェース、パラレル・インターフェース、Ａ／Ｄコンバータ、タイマ
";
page_98[0][6]="2  ＤＭＡコントローラ、割り込みコントローラなどを１チップに集積したマイクロコ
";
page_98[0][7]="2  ンピュータです。";
page_98[0][8]="2    μＰＤ７０４３３はＶ５５ファミリの１つで、シングルチップ・マイクロコンピ
";
page_98[0][9]="2  ュータμＰＤ７０３２０、７０３３０（別称Ｖ２５、Ｖ３５）とソ¥フトウェア・
コ";
page_98[0][10]="2  ンパチブルです。Ｖ５５ファミリはＶ２５の後継機種として、より高機能¥、高性
```

1

保守／廃止

TASK03.C

```
能¥";
page_98[0][11]="2  化を図り、応用分野ごとに製品展開を行っています。";
page_98[0][12]="2    特にμＰＤ７０４３３では、プリンタ、ファクシミリ応用分野を対象としてい
ます";
page_98[0][13]="2  ";
page_98[0][14]="2  ";
page_98[0][15]="2  ";
page_98[0][16]="2  ";
page_98[0][17]="2  ";
page_98[0][18]="2  ";
page_98[0][19]="2  ";
page_98[0][20]="2  ";
page_98[0][21]="2  ";
page_98[0][22]="2  キャリジリターンを押して下さい !!!";
page_98[0][23]="0¥n";


page_98[1][0]="3  特徴";
page_98[1][1]="2  ○内部１６ビット・アーキテクチャ、外部１６／８ビット・データ・バス幅";
page_98[1][2]="2    切り替え可能¥";
page_98[1][3]="2  ○μＰＤ７０３２０、７０３３０とソ¥フトウエア・コンパチブル（追加命令有り
）";
page_98[1][4]="2  ○最小インストラクション・サイクル：１６０ｎＳ／１２．５ＭＨｚ";
page_98[1][5]="2    （外部２５ＭＨｚ時）";
page_98[1][6]="2  ○メモリ空間：１６Ｍバイト――┬―基本メモリ空間１Ｍバイト";
page_98[1][7]="2                              └―拡張メモリ空間１６Ｍバイト";
page_98[1][8]="2              レジスタ・ファイル空間　５１２バイト／１６レジスタ・バンク";
page_98[1][9]="2  ○Ｉ／Ｏ空間：６４Ｋバイト";
page_98[1][10]="2  ○メモリ空間を可変サイズ（最大６ブロック）で分割制御";
page_98[1][11]="6    ◆プログラマブル・ウエイト機能¥";
page_98[1][12]="2  ○Ｉ／Ｏライン（入力ポート：１１ビット、入出力ポート４２ビット）";
page_98[1][13]="2  ○ＤＭＡコントローラ（ＤＭＡＣ）：最大４チャネル構¥成可能¥";
page_98[1][14]="6    ◆４種類のＤＭＡ転送モード（外部、シリアルＩ／Ｆ、タイマ、パラレルＩ／
Ｆ";
page_98[1][15]="6      から選択可能¥）";
page_98[1][16]="6    ◆インテリジェントＤＭＡモード－１、－２";
page_98[1][17]="2  ○シリアル・インターフェース：２チャネル";
page_98[1][18]="6    ◆調歩同期モード（ＵＡＲＴ）またはクロック同期モード（ＣＳＩ）のいづれ
かを";
page_98[1][19]="6      選択可能¥";
page_98[1][20]="2 ";
page_98[1][21]="2 ";
page_98[1][22]="2   キャリジリターンを押して下さい。 !!! ";
page_98[1][23]="0¥n";


page_98[2][0]="2  ○パラレル・インターフェース：８ビット";
page_98[2][1]="6    ◆セントロニクス・データ入力および汎用データ入出力";
page_98[2][2]="2  ○Ａ／Ｄコンバータ（８ビット）：４チャネル";
page_98[2][3]="2  ○リアルタイム出力ポート";
page_98[2][4]="2  ○ＰＷＭ出力機能¥";
page_98[2][5]="2  ○割り込みコントローラ";
page_98[2][6]="6    ◆プログラマブル・プライオリティ（４レベル）";
page_98[2][7]="6    ◆３種類の割り込み応答方式";
page_98[2][8]="6    ◆ベクタ割り込み機能¥、レジスタ・バンク切り替え機能¥、マクロ・サービス機
能¥";
page_98[2][9]="2  ○１６ビットタイマ：４チャネル";
page_98[2][10]="2  ○ウォッチドック・タイマ機能¥";
page_98[2][11]="2  ○ソ¥フトウェア・インターバル・タイマ";
page_98[2][12]="2  ○ＤＲＡＭ、疑似Ｓ－ＲＡＭリフレッシュ機能¥";
page_98[2][13]="2  ○スタンバイ機能¥（ＳＴＯＰモード、ＨＯＬＴモード）";
page_98[2][14]="2  ○クロック発生回路内臓";
page_98[2][15]="2 ";
page_98[2][16]="3  用途";
page_98[2][17]="2  ○シリアル通信やパラレル通信を用いたデータ処理システムの制御用";
page_98[2][18]="2    （データ処理端末、プリンタ、Ｇ４ファックスなど）";
page_98[2][19]="2 ";
page_98[2][20]="2 ";
page_98[2][21]="2 ";
```

2

保守／廃止

TASK03.C

```
page_98[2][22]="2　キャリジリターンを押して下さい。";
page_98[2][23]="0¥n";


}

/************************************************
 ************************************************/

ss_test()
{
static  unsigned char   is_inited;

        v55reg->TxBRG1  = 77;
        v55reg->RxBRG1  = 77;
        v55reg->PRS1    = B00101101;    /*1200**/

        ss_msginit();
        chg_s_to_r=OFF;
        setfirstdata(page_98[0][0]);
        serial(ON);

}


setfirstdata(pagep)
unsigned char   *pagep;
{

        setmsgptr(pagep);
        v55reg->TxB1_SIO1=*S_DATAP;
        S_DATAP++;
}

serial(sw)
unsigned char   sw;
{

        if(sw==ON)v55reg->UARTM1_CSIM1 = 0x88;

        else    v55reg->UARTM1_CSIM1=0x08;

}



/****************************************
 ****************************************/

task_ss_ope()
{
static  unsigned char   tbuf,d_get;
unsigned short  j;
static  unsigned char   count;
static  unsigned char   i;
unsigned char   dammy;


        if(chg_s_to_r==ON){
                i=chgsp_to_rcv();
                chg_s_to_r=OFF;
                return(0);
        }

        if((v55reg->UARTM1_CSIM1 & 0x80) !=0){

                if(v55reg->U1RTS1 & 0x20 ==0)return(0);

                dammy=d_get=OFF;
```

TASK03.C

```
                switch(*S_DATAP){

                        case    C_R:    v55reg->TxB1_SIO1=0x0d;
                                        count++;
                                        S_DATAP++;
                                        break;

                        case    E_O_P:  v55reg->TxB1_SIO1=E_O_P;
                                        count++;
                                        chg_s_to_r=ON;
                                        S_DATAP+=2;
                                        break;

                        default:        v55reg->TxB1_SIO1=*S_DATAP;
                                        S_DATAP++;
                                        break;
                }
        }

        else    {

                if(((v55reg->U1RTS1 & 0x10) == 0) && (d_get==OFF))return(0);

                tbuf=v55reg->RxB1;
                if((tbuf==0x0d) && (d_get==OFF)){
                        d_get=ON;
                }
                else    {
                                tbuf=d_get=OFF;
                                v55reg->TxB1_SIO1=*S_DATAP;
                                S_DATAP++;
                                v55reg->UARTM1_CSIM1 = 0x88;
                }
        }

        if(count>=MSGLNGTH){
                setmsgptr(page_98[0][0]);
                count=0;
        }

}


setmsgptr(msgptr)
unsigned char   *msgptr;
{
        S_DATAP=msgptr;
}

/************************************************
 *      １ページ文メッセージを出し終わると      *
 *      シリアルポートを受信専用に変え          *
 *      何かデータが入って来るのを待ちます      *
 *      in...nothing                            *
 *      out..nothing                            *
 ************************************************/

chgsp_to_rcv()
{
unsigned short  j;

        j=v55reg->RxB1;
        v55reg->UARTM1_CSIM1=0x48;
        return(0);
}
```

4

## A.16　HEAD.C

HEAD.C

```
/*****************************************************************************
        ＃３２９
        ヘッダ作成プログラムズ
*****************************************************************************/

#include        "HD329.H"

/* ＲＴＣ年月日時間構造体 */
        struct  RTC_DATA far    *r_data;

/* ヘッダーキャラ格納配列（１ライン１０８文字並べられる。） */
uchar   head_chara_line[108];

/* 関数プロトタイプ宣言 */
        sethead(uchar far*,uchar);
uchar   *setfdata(uchar *chara_line,uchar page);
uchar   *set_page(uchar *page_p,uchar page);
uchar   *set_y_m_d_t(uchar *ymdt_p);
uchar   *set_from(uchar *from_p);
uchar   *set_name(uchar *name_p);
uchar far* chara_to_bits(uchar far*p_top,uchar *chara_line_top);
uchar*  set_config(uchar*);
uchar*  set_blank(uchar*, uchar);

extern  uchar far       ptn1[256][2*9];
extern  uchar far       MIRRORTB[256];

static  uchar   sw_HC;
static  uchar   sw_Head_Fn;


set_HC(uint c)
{
        if(c!=0){
                sw_HC=1;
        }else{
                sw_HC=0;
        }
}
set_Head_Fn(uint c)
{
        if(c!=0){
                sw_Head_Fn=1;
        }else{
                sw_Head_Fn=0;
        }
}

/*********************************************************
 *      ヘッダー作成ルーチン                             *
 *      ユーザーネームは                        *
 *      "Ｖ５５ＰＩ　ＤＥＭＯ　ＳＹＳＴＥＭ"          *
 *      固定。年・月・日・時間だけが可変です。           *
 *      in.....uchar   *p                                *
 *       （ビットデータ格納先頭ポインタ。縦９ビット      *
 *        横１６ビットのキャラを［１１］［２１６］へ    *
 *        展開する。）                                   *
 *      uchar   page;                                    *
 *       （表示ページナンバー）                          *
 *                                                       *
 *      out.....int                                      *
 *       （生成した行数を返値。意味が有ります）　ＮＢ    *
 *********************************************************/

sethead(p,page)
uchar far*      p;
uchar           page;
{
uint    line;
```

1

HEAD.C

```
        r_data = (struct RTC_DATA far *)MMIO_RTC;

        chara_to_bits(p,setfdata(head_chara_line,page));

        line = (9+1)<<sw_Head_Fn;
        line = ( line<=80 ? line : 80 );
        return( (9+1)<<sw_Head_Fn );
}

/*************************************************************
 *      ＦＯＲＭＡＴ付きデータ                               *
 *      名前　時間　日付　年　ページ                         *
 *      の順に並べます。                          *
 *      in....uchar     *chara_line;              *
 *           uchar      page;                     *
 *           キャラクタを並べる先頭ポインタとページ数  *
 *      out...キャラクタの先頭ポインタ                       *
 *************************************************************/



uchar   *setfdata(chara_line,page)
uchar   *chara_line;
uchar   page;
{
uchar   i;
uchar*  line_p;
uchar*  j;

        set_blank(chara_line, 108);

        line_p=chara_line;
        line_p=set_blank(line_p, 10);
        line_p=set_from(line_p);
                line_p=set_blank(line_p, 2);
        line_p=set_name(line_p);
                line_p=set_blank(line_p, 6);
                line_p=set_config(line_p);

        line_p=chara_line+78;
        line_p=set_y_m_d_t(line_p);
        line_p=set_page(line_p,page);

        return(chara_line);
}


/*************************************************************
 *      ページ数のセット。ＨＥＸをＤＥＣへ変換              *
 *      した後、ポインターを進めて返値します。               *
 *************************************************************/

uchar   *set_page(page_p,page)
uchar   *page_p;
uchar   page;
{
uchar   i,d_page[2];

        for(i=0;i<10;i++,page_p++)*page_p=' ';
        *page_p='P';
        page_p++;
        hex_to_dec(d_page,page);
        *page_p=d_page[0];
        page_p++;
        *page_p=d_page[1];
        page_p++;
        return(page_p);
}
```

2

保守／廃止

HEAD.C

```
hex_to_dec(pp1,page)
uchar   *pp1;
uchar   page;
{
uchar   i,j;

        if((page & 0x0f) < 0x0a){
                *(pp1+1)=(0x0f & page)+0x30;
                *pp1=(0xf0 & page)+0x30;
        }

        else    {
                *(pp1+1)=(0x0f & page)-0x01+0x30;
                *pp1=(0xf0 & page)+0x01+0x30;
        }
}


/*************************************************
 *      与えられたポインターの所から、           *
 *      時：分　月／日　年                       *
 *      を並べて、ポインターを進めた後返値します*
 *************************************************/

uchar   *set_y_m_d_t(ymdt_p)
uchar   *ymdt_p;
{
uchar   i;


        *ymdt_p=((0x0f & r_data->tenhour) | 0x30);
        ymdt_p++;
        *ymdt_p=((0x0f & r_data->onehour) | 0x30);
        ymdt_p++;
        *ymdt_p=':';
        ymdt_p++;
        *ymdt_p=((0x0f & r_data->tenmin) | 0x30);
        ymdt_p++;
        *ymdt_p=((0x0f & r_data->onemin) | 0x30);
        ymdt_p++;
        *ymdt_p=' ';
        ymdt_p++;
        *ymdt_p=((0x0f & r_data->tenmonth) | 0x30);
        ymdt_p++;
        *ymdt_p=((0x0f & r_data->onemonth) | 0x30);
        ymdt_p++;
        *ymdt_p='/';
        ymdt_p++;
        *ymdt_p=((0x0f & r_data->tendate) | 0x30);
        ymdt_p++;
        *ymdt_p=((0x0f & r_data->onedate) | 0x30);
        ymdt_p++;
        *ymdt_p=' ';
        ymdt_p++;

        *ymdt_p=((0x0f & r_data->tenyear) | 0x30);
        ymdt_p++;
        *ymdt_p=((0x0f & r_data->oneyear) | 0x30);
        ymdt_p++;

        return(ymdt_p);
}

/*************************************************
 *      ＦＲＯＭ／ＣＯＰＹの文字を並べます      *
 *      ポインターを進めて返値します。           *
 *************************************************/
```

HEAD.C

```
uchar*  set_from(from_p)
uchar*  from_p;
{
uchar   i;

      if(sw_HC!=0){
        *from_p='[';    from_p++;
        *from_p='F';    from_p++;
        *from_p='A';    from_p++;
        *from_p='C';    from_p++;
        *from_p='S';    from_p++;
        *from_p='I';    from_p++;
        *from_p='M';    from_p++;
        *from_p='I';    from_p++;
        *from_p='L';    from_p++;
        *from_p='E';    from_p++;
        *from_p=']';    from_p++;
      }else{
        *from_p='<';    from_p++;
        *from_p='<';    from_p++;
        *from_p='C';    from_p++;
        *from_p='O';    from_p++;
        *from_p='P';    from_p++;
        *from_p='Y';    from_p++;
        *from_p='>';    from_p++;
        *from_p='>';    from_p++;
      }

return(from_p);
}



/************************************************
 *      与えられたポインターの場所から          *
 *      名前を並べて格納します。        *
 *      ポインターを進めて返値します            *
 ************************************************/

uchar   v55_name[]="V55PI DEMO SYSTEM";


uchar   *set_name(name_p)
uchar   *name_p;
{
uchar   *name_p_tp;

        name_p_tp=name_p;
        name_p=name_p+17;
        d_copy(name_p_tp,v55_name,17);

        return(name_p);
}



d_copy(p1,p2,cnt)
uchar   *p1;
uchar   *p2;
uchar   cnt;
{
        while(cnt != 0){
                *p1=*p2;
                p1++;
                p2++;
                cnt--;
        }
}
```

4

保守／廃止

HEAD.C

```
/************************************************
 *      与えられたポインターの場所から          *
 *      モードを並べて格納します。              *
 *      ポインターを進めて返値します            *
 ************************************************/

uchar*  set_config(from_p)
uchar*  from_p;
{
uchar*  from_p_tp;

        from_p_tp = from_p;
        from_p   += 10;
    if(sw_Head_Fn==1){
        *from_p_tp='B'; from_p_tp++;
        *from_p_tp='Y'; from_p_tp++;
        *from_p_tp=' '; from_p_tp++;
        *from_p_tp='F'; from_p_tp++;
        *from_p_tp='I'; from_p_tp++;
        *from_p_tp='N'; from_p_tp++;
        *from_p_tp='E'; from_p_tp++;
    }else if(sw_Head_Fn==2){
        *from_p_tp='B'; from_p_tp++;
        *from_p_tp='Y'; from_p_tp++;
        *from_p_tp=' '; from_p_tp++;
        *from_p_tp='S'; from_p_tp++;
        *from_p_tp='-'; from_p_tp++;
        *from_p_tp='F'; from_p_tp++;
        *from_p_tp='I'; from_p_tp++;
        *from_p_tp='N'; from_p_tp++;
        *from_p_tp='E'; from_p_tp++;
    }else{
        /*NOP**/
    }
return(from_p);
}



/************************************************
 *      与えられたポインターの場所から          *
 *      ブランクを並べて格納します。            *
 *      ポインターを進めて返値します            *
 ************************************************/

uchar   *set_blank(uchar* blank_p, uchar kazu)
{
uchar   i;
uchar   limit;

        limit = ( kazu <= 108 ? kazu : 108 );
        for(i=0; i<limit; i++,blank_p++)        *blank_p=' ';

        return(blank_p);
}

/************************************************************
 *      キャラクター列１０８個（chara_line_topから）       *
 *      のデータをビット展開し、p_top  から始まる           *
 *      バッファ列へ格納する                                *
 ************************************************************/

uchar far* chara_to_bits(p_top,chara_line_top)
uchar far*      p_top;
uchar*          chara_line_top;
{
uchar   i,j,k;
```

5

HEAD.C

```
uchar   CHCODE[108];
uchar far*      pp_top;
uchar far*      tp_top;
uchar   density;

        density = 1<<sw_Head_Fn;        density = ( density<=4 ? density : 4 );
        pp_top=p_top;
        d_copy(CHCODE,chara_line_top,108);
        for(j=0;j<18;j+=2) {
            for(k=1; k<=density; k++){
                tp_top =p_top;
                for(i=0;i<216;i+=2,tp_top+=2) {
                        *(tp_top+0)=MIRRORTB[ptn1[CHCODE[i/2]][j]];
                        *(tp_top+1)=MIRRORTB[ptn1[CHCODE[i/2]][j+1]];
                }
                p_top +=0x100;
            }
        }
            for(k=1; k<=density; k++){
                tp_top =p_top;
                for(i=0;i<216;i+=2,tp_top+=2) {
                        *(ushort far*)tp_top=0x0000;
                }
                p_top +=0x100;
            }
        return(pp_top);
}
```

6

## A.17　CRG.C

CRG.C

```
/* ******************************************************** */
/* ******************************************************** */

#include        "BASICDEF.H"

#define WHITE   {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}
#define LENGTH  216

uchar   CHCODE[(LENGTH/2)+1];

uchar far ptn1[256][2*9] = {


WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,

        /* 0x10-        */

WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,

        /* 0x20-        */

WHITE,

{    /* ! PATTERN */
    0x03,0x00,0x03,0x00,0x03,0x00,0x03,0x00,
    0x03,0x00,0x00,0x00,0x03,0x00,0x03,0x00,
    0x00,0x00
},

{    /* " PATTERN */
    0x19,0x80,0x19,0x80,0x19,0x80,0x33,0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00
},

{    /* # PATTERN */
    0x03,0x30,0x03,0x30,0x1F,0xFC,0x06,0x60,
    0x06,0x60,0x3F,0xF8,0x0C,0xC0,0x0C,0xC0,
    0x0C,0xC0
```

1

保守／廃止

CRG.C

```
},


{    /* $ PATTERN */
    0x01,0x80,0x0F,0xF0,0x19,0x98,0x19,0x80,
    0x0F,0xF0,0x01,0x98,0x19,0x98,0x0F,0xF0,
    0x01,0x80
},

{    /* % PATTERN */
    0x1C,0x18,0x36,0x30,0x36,0x60,0x1C,0xC0,
    0x01,0x80,0x03,0x38,0x06,0x6C,0x0C,0x6C,
    0x18,0x38
},

{    /* & PATTERN */
    0x0F,0x80,0x18,0xC0,0x18,0x00,0x0E,0x00,
    0x18,0x78,0x30,0xCC,0x30,0x78,0x18,0xE0,
    0x0F,0xBC
},

{    /* ' PATTERN */
    0x0E,0x00,0x0E,0x00,0x06,0x00,0x0C,0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00
},

{    /* ( PATTERN */
    0x00,0x60,0x00,0xC0,0x01,0x80,0x01,0x80,
    0x01,0x80,0x01,0x80,0x00,0xC0,0x00,0x60,
    0x00,0x00
},

{    /* ) PATTERN */
    0x06,0x00,0x03,0x00,0x01,0x80,0x01,0x80,
    0x01,0x80,0x01,0x80,0x03,0x00,0x06,0x00,
    0x00,0x00
},

{    /* * PATTERN */
    0x00,0x00,0x06,0x30,0x03,0x60,0x01,0xC0,
    0x1F,0xFC,0x01,0xC0,0x03,0x60,0x06,0x30,
    0x00,0x00
},

{    /* + PATTERN */
    0x00,0x00,0x01,0x80,0x01,0x80,0x01,0x80,
    0x1F,0xF8,0x01,0x80,0x01,0x80,0x01,0x80,
    0x00,0x00
},

{    /* , PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00,0x1C,0x00,0x1C,0x00,0x0C,0x00,
    0x18,0x00
},

{    /* - PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x0F,0xE0,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00
},

{    /* . PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00,0x18,0x00,0x3C,0x00,0x18,0x00,
    0x00,0x00
},
```

2

CRG.C

```
{    /* / PATTERN */
    0x00,0x00,0x00,0x30,0x00,0x60,0x00,0xC0,
    0x01,0x80,0x03,0x00,0x06,0x00,0x0C,0x00,
    0x00,0x00
},

        /* 0x30-        */


{    /* 0 PATTERN */
    0x1F,0xF0,0x30,0x38,0x60,0x6C,0x60,0xCC,
    0x61,0x8C,0x63,0x0C,0x66,0x0C,0x3C,0x18,
    0x1F,0xF0
},

{    /* 1 PATTERN */
    0x07,0x00,0x0F,0x00,0x1B,0x00,0x03,0x00,
    0x03,0x00,0x03,0x00,0x03,0x00,0x03,0x00,
    0x1F,0xE0
},

{    /* 2 PATTERN */
    0x3F,0xF0,0x60,0x18,0x60,0x18,0x00,0x18,
    0x00,0xF0,0x07,0x80,0x1C,0x00,0x30,0x00,
    0x7F,0xF8
},

{    /* 3 PATTERN */
    0x3F,0xF8,0x60,0x0C,0x00,0x0C,0x00,0x0C,
    0x0F,0xF8,0x00,0x0C,0x00,0x0C,0x60,0x0C,
    0x3F,0xF8
},

{    /* 4 PATTERN */
    0x01,0xC0,0x03,0xC0,0x06,0xC0,0x0C,0xC0,
    0x18,0xC0,0x30,0xC0,0x7F,0xFC,0x00,0xC0,
    0x00,0xC0
},

{    /* 5 PATTERN */
    0x7F,0xF8,0x60,0x00,0x60,0x00,0x7F,0xF0,
    0x00,0x18,0x00,0x0C,0x00,0x0C,0x60,0x18,
    0x3F,0xF0
},

{    /* 6 PATTERN */
    0x3F,0xF8,0x60,0x00,0x60,0x00,0x60,0x00,
    0x7F,0xF8,0x60,0x0C,0x60,0x0C,0x60,0x0C,
    0x3F,0xF8
},

{    /* 7 PATTERN */
    0x7F,0xFC,0x00,0x0C,0x00,0x18,0x00,0x30,
    0x00,0x60,0x00,0xC0,0x01,0x80,0x03,0x00,
    0x06,0x00
},

{    /* 8 PATTERN */
    0x1F,0xF0,0x60,0x0C,0x60,0x0C,0x60,0x0C,
    0x3F,0xF8,0x60,0x0C,0x60,0x0C,0x60,0x0C,
    0x3F,0xF8
},

{    /* 9 PATTERN */
    0x3F,0xF8,0x60,0x0C,0x60,0x0C,0x60,0x0C,
    0x3F,0xFC,0x00,0x0C,0x00,0x0C,0x00,0x0C,
    0x3F,0xF8
},
```

3

CRG.C

```
{    /* : PATTERN */
    0x00,0x00,0x03,0x80,0x03,0x80,0x00,0x00,
    0x00,0x00,0x03,0x80,0x03,0x80,0x00,0x00,
    0x00,0x00
},

{    /* ; PATTERN */
    0x03,0x80,0x03,0x80,0x00,0x00,0x00,0x00,
    0x00,0x00,0x03,0x80,0x03,0x80,0x01,0x80,
    0x03,0x00
},

{    /* < PATTERN */
    0x00,0x00,0x00,0x60,0x01,0x80,0x06,0x00,
    0x18,0x00,0x06,0x00,0x01,0x80,0x00,0x60,
    0x00,0x00
},

{    /* = PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x3F,0xF8,
    0x00,0x00,0x00,0x00,0x3F,0xF8,0x00,0x00,
    0x00,0x00
},

{    /* > PATTERN */
    0x00,0x00,0x18,0x00,0x06,0x00,0x01,0x80,
    0x00,0x60,0x01,0x80,0x06,0x00,0x18,0x00,
    0x00,0x00
},

{    /* ? PATTERN */
    0x1F,0xF0,0x30,0x18,0x60,0x0C,0x60,0x0C,
    0x00,0x18,0x03,0xF0,0x03,0x00,0x00,0x00,
    0x03,0x00
},

        /* 0x40-        */


{    /* @ PATTERN */
    0x07,0xE0,0x0C,0x30,0x18,0x18,0x31,0xCC,
    0x33,0x6C,0x33,0x6C,0x1B,0xF8,0x0C,0x38,
    0x07,0xEC
},



{    /* A PATTERN */
    0x03,0x00,0x07,0x80,0x0C,0xC0,0x18,0x60,
    0x30,0x30,0x7F,0xF8,0xC0,0x0C,0xC0,0x0C,
    0xC0,0x0C
},

{    /* B PATTERN */
    0x7F,0xF0,0x60,0x18,0x60,0x0C,0x60,0x18,
    0x7F,0xF0,0x60,0x18,0x60,0x0C,0x60,0x18,
    0x7F,0xF0
},

{    /* C PATTERN */
    0x1F,0xE0,0x30,0x18,0x60,0x0C,0x60,0x00,
    0x60,0x00,0x60,0x00,0x60,0x0C,0x30,0x18,
    0x1F,0xF0
},

{    /* D PATTERN */
    0x7F,0xC0,0x60,0x70,0x60,0x18,0x60,0x0C,
    0x60,0x0C,0x60,0x0C,0x60,0x18,0x60,0x70,
    0x7F,0xC0
```

4

CRG.C

```
},

{    /* E PATTERN */
    0x7F,0xFC,0x60,0x00,0x60,0x00,0x60,0x00,
    0x7F,0xE0,0x60,0x00,0x60,0x00,0x60,0x00,
    0x7F,0xFC
},

{    /* F PATTERN */
    0x7F,0xFC,0x60,0x00,0x60,0x00,0x60,0x00,
    0x7F,0xE0,0x60,0x00,0x60,0x00,0x60,0x00,
    0x60,0x00
},

{    /* G PATTERN */
    0x1F,0xE0,0x30,0x30,0x60,0x00,0x60,0x00,
    0x60,0xFC,0x60,0x30,0x30,0x30,0x1F,0xF0,
    0x00,0x30
},

{    /* H PATTERN */
    0x60,0x0C,0x60,0x0C,0x60,0x0C,0x60,0x0C,
    0x7F,0xFC,0x60,0x0C,0x60,0x0C,0x60,0x0C,
    0x60,0x0C
},

{    /* I PATTERN */
    0x07,0x80,0x03,0x00,0x03,0x00,0x03,0x00,
    0x03,0x00,0x03,0x00,0x03,0x00,0x03,0x00,
    0x07,0x80
},

{    /* J PATTERN */
    0x00,0xFC,0x00,0x30,0x00,0x30,0x00,0x30,
    0x60,0x30,0x60,0x30,0x60,0x30,0x30,0x60,
    0x1F,0xC0
},

{    /* K PATTERN */
        B01100000,B00111100,
        B01100000,B00110000,
        B01100000,B11100000,
        B01100011,B10000000,
        B01101111,B00000000,
        B01111101,B10000000,
        B01100000,B11100000,
        B01100000,B00110000,
        B01100000,B00111100
},

{    /* L PATTERN */
    0x60,0x00,0x60,0x00,0x60,0x00,0x60,0x00,
    0x60,0x00,0x60,0x00,0x60,0x00,0x60,0x00,
    0x7F,0xFC
},

{    /* M PATTERN */
    0x60,0x0C,0x78,0x3C,0x6C,0x6C,0x66,0xCC,
    0x63,0x8C,0x60,0x0C,0x60,0x0C,0x60,0x0C,
    0x60,0x0C
},

{    /* N PATTERN */
    0x60,0x0C,0x78,0x0C,0x6C,0x0C,0x66,0x0C,
    0x63,0x8C,0x60,0xCC,0x60,0x6C,0x60,0x3C,
    0x60,0x0C
},

{    /* O PATTERN */
```

5

CRG.C

```
    0x0F,0xE0,0x30,0x18,0x60,0x0C,0x60,0x0C,
    0x60,0x0C,0x60,0x0C,0x60,0x0C,0x30,0x18,
    0x0F,0xE0
},

/*      0x50    */

{   /* P PATTERN */
    0x7F,0xF0,0x60,0x18,0x60,0x0C,0x60,0x0C,
    0x60,0x18,0x7F,0xF0,0x60,0x00,0x60,0x00,
    0x60,0x00
},

{   /* Q PATTERN */
    0x1F,0xF0,0x30,0x18,0x60,0x0C,0x60,0x0C,
    0x60,0x0C,0x67,0x8C,0x30,0xD8,0x1F,0xF0,
    0x00,0x3C
},

{   /* R PATTERN */
    0x7F,0xF0,0x60,0x18,0x60,0x0C,0x60,0x0C,
    0x60,0x18,0x7F,0xF0,0x60,0x18,0x60,0x0C,
    0x60,0x0C
},

{   /* S PATTERN */
    0x1F,0xF0,0x30,0x1C,0x60,0x00,0x38,0x00,
    0x0F,0xE0,0x00,0x38,0x00,0x0C,0x70,0x18,
    0x1F,0xF0
},

{   /* T PATTERN */
    0xFF,0xFC,0x03,0x00,0x03,0x00,0x03,0x00,
    0x03,0x00,0x03,0x00,0x03,0x00,0x03,0x00,
    0x03,0x00
},

{   /* U PATTERN */
    0x60,0x0C,0x60,0x0C,0x60,0x0C,0x60,0x0C,
    0x60,0x0C,0x60,0x0C,0x60,0x0C,0x30,0x18,
    0x1F,0xF0
},

{   /* V PATTERN */
    0x60,0x0C,0x60,0x0C,0x60,0x0C,0x60,0x0C,
    0x30,0x18,0x18,0x30,0x0C,0x60,0x06,0xC0,
    0x03,0x80
},

{   /* W PATTERN */
    0x60,0x0C,0x60,0x0C,0x60,0x0C,0x60,0x0C,
    0x63,0x8C,0x66,0xCC,0x6C,0x6C,0x78,0x3C,
    0x60,0x0C
},

{   /* X PATTERN */
    0x60,0x18,0x30,0x30,0x18,0x60,0x0C,0xC0,
    0x03,0x00,0x0C,0xC0,0x18,0x60,0x30,0x30,
    0x60,0x18
},

{   /* Y PATTERN */
    0x60,0x18,0x30,0x30,0x18,0x60,0x0C,0xC0,
    0x07,0x80,0x03,0x00,0x03,0x00,0x03,0x00,
    0x03,0x00
},

{   /* Z PATTERN */
    0x7F,0xFC,0x00,0x18,0x00,0x70,0x00,0xC0,
```

CRG.C

```
    0x03,0x80,0x06,0x00,0x1C,0x00,0x30,0x00,
    0x7F,0xFC
},

{    /* [ PATTERN */
        B00000111,B11100000,
        B00000110,B00000000,
        B00000110,B00000000,
        B00000110,B00000000,
        B00000110,B00000000,
        B00000110,B00000000,
        B00000110,B00000000,
        B00000110,B00000000,
        B00000111,B11100000
},

{    /* ¥ PATTERN */
    0x70,0x38,0x18,0x60,0x0F,0xC0,0x03,0x00,
    0x3F,0xF0,0x03,0x00,0x3F,0xF0,0x03,0x00,
    0x03,0x00
},

{    /* ] PATTERN */
        B00000111,B11100000,
        B00000000,B01100000,
        B00000000,B01100000,
        B00000000,B01100000,
        B00000000,B01100000,
        B00000000,B01100000,
        B00000000,B01100000,
        B00000000,B01100000,
        B00000111,B11100000
},

{    /* ^ PATTERN */
    0x00,0x00,0x00,0x00,0x03,0x00,0x0C,0xC0,
    0x30,0x30,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00
},

{    /* _ PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x1F,0xF8
},

        /* 0x60-        */


WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,

        /* 0x70-        */
```

CRG.C

```
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,

        /* 0x80-        */

WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,

        /* 0x90-        */

WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,


        /* 0xa0-        */

WHITE,


{    /* ｡ PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00,0x1C,0x00,0x36,0x00,0x36,0x00,
    0x1C,0x00
},
```

8

CRG.C

```
{    /* 「 PATTERN */
    0x00,0x7C,0x00,0x60,0x00,0x60,0x00,0x60,
    0x00,0x60,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00
},


{    /* 」 PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x18,0x00,0x18,0x00,0x18,0x00,0x18,
    0x00,0xF8
},


{    /* 、 PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x0C,0x00,
    0x06,0x00
},

{    /* ・ PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x07,0x00,
    0x07,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00
},

{    /* ｦ PATTERN */
    0x7F,0xFC,0x00,0x0C,0x00,0x0C,0x3F,0xFC,
    0x00,0x0C,0x00,0x18,0x00,0x18,0x00,0x70,
    0x0F,0xC0
},

{    /* ｧ PATTERN */
    0x00,0x00,0x00,0x00,0x3F,0xE0,0x00,0x60,
    0x06,0xC0,0x07,0x80,0x0C,0x00,0x18,0x00,
    0x00,0x00
},

{    /* ｨ PATTERN */
    0x00,0x00,0x00,0x00,0x00,0xC0,0x01,0x80,
    0x07,0x80,0x0D,0x80,0x19,0x80,0x01,0x80,
    0x00,0x00
},

{    /* ｩ PATTERN */
    0x00,0x00,0x00,0x00,0x03,0x00,0x1F,0xF0,
    0x18,0x30,0x00,0x30,0x00,0x60,0x03,0xC0,
    0x00,0x00
},

{    /* ｪ PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0xF0,
    0x01,0x80,0x01,0x80,0x01,0x80,0x0F,0xF0,
    0x00,0x00
},

{    /* ｫ PATTERN */
    0x00,0x00,0x00,0x00,0x01,0x80,0x1F,0xF0,
    0x03,0x80,0x07,0x80,0x1D,0x80,0x01,0x80,
    0x00,0x00
},

{    /* ｬ PATTERN */
    0x00,0x00,0x00,0x00,0x06,0x00,0x1F,0xF8,
    0x06,0x30,0x03,0x60,0x03,0x00,0x01,0x80,
    0x00,0x00
},
```

9

保守／廃止

CRG.C

```
{    /* ｧ PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x07,0xF0,
    0x00,0x30,0x00,0x30,0x00,0x30,0x1F,0xFC,
    0x00,0x00
},

{    /* ｨ PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0xF0,
    0x00,0x30,0x07,0xF0,0x00,0x30,0x0F,0xF0,
    0x00,0x00
},

{    /* ｩ PATTERN */
    0x00,0x00,0x00,0x00,0x36,0xC0,0x36,0xC0,
    0x00,0xC0,0x00,0xC0,0x01,0x80,0x07,0x00,
    0x00,0x00
},

        /* 0xb0-        */


{    /* ｰ PATTERN */
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x3F,0xF8,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00
},

{    /* ｱ PATTERN */
    0x7F,0xFC,0x00,0x0C,0x00,0x0C,0x03,0x38,
    0x03,0xE0,0x03,0x00,0x03,0x00,0x06,0x00,
    0x0C,0x00
},

{    /* ｲ PATTERN */
    0x00,0x1C,0x00,0x70,0x00,0xC0,0x01,0x80,
    0x07,0x80,0x0D,0x80,0x79,0x80,0x01,0x80,
    0x01,0x80
},

{    /* ｳ PATTERN */
    0x03,0x00,0x03,0x00,0x7F,0xFC,0x60,0x0C,
    0x60,0x0C,0x00,0x18,0x00,0x30,0x00,0x60,
    0x0F,0xC0
},

{    /* ｴ PATTERN */
    0x00,0x00,0x1F,0xF8,0x01,0x80,0x01,0x80,
    0x01,0x80,0x01,0x80,0x01,0x80,0x01,0x80,
    0x3F,0xFC
},

{    /* ｵ PATTERN */
    0x01,0x80,0x01,0x80,0x7F,0xFC,0x01,0x80,
    0x07,0x80,0x0D,0x80,0x19,0x80,0x31,0x80,
    0x61,0x80
},

{    /* ｶ PATTERN */
    0x06,0x00,0x06,0x00,0x7F,0xFC,0x06,0x0C,
    0x06,0x0C,0x0C,0x0C,0x18,0x18,0x30,0x30,
    0x61,0xE0
},

{    /* ｷ PATTERN */
    0x03,0x00,0x03,0x00,0x3F,0xF8,0x03,0x00,
    0x7F,0xFC,0x01,0x80,0x01,0x80,0x00,0xC0,
    0x00,0xC0
```

CRG.C

```
},

{    /* ｸ PATTERN */
    0x0C,0x00,0x0F,0xF8,0x18,0x18,0x30,0x18,
    0x00,0x30,0x00,0x30,0x00,0x60,0x00,0xC0,
    0x07,0x80
},

{    /* ｹ PATTERN */
    0x18,0x00,0x18,0x00,0x1F,0xFC,0x30,0xC0,
    0xE0,0xC0,0x01,0x80,0x01,0x80,0x03,0x00,
    0x0E,0x00
},

{    /* ｺ PATTERN */
    0x00,0x00,0x7F,0xF8,0x00,0x18,0x00,0x18,
    0x00,0x18,0x00,0x18,0x00,0x18,0x00,0x18,
    0x7F,0xF8
},

{    /* ｻ PATTERN */
    0x0C,0x30,0x0C,0x30,0x3F,0xFC,0x0C,0x30,
    0x0C,0x30,0x00,0x30,0x00,0x60,0x00,0xC0,
    0x03,0x80
},

{    /* ｼ PATTERN */
    0x30,0x00,0x18,0x0C,0x00,0x0C,0x60,0x18,
    0x30,0x18,0x00,0x30,0x00,0x30,0x00,0xE0,
    0x7F,0x80
},

{    /* ｽ PATTERN */
    0x00,0x00,0x7F,0xFC,0x00,0x18,0x00,0x30,
    0x00,0xE0,0x03,0x80,0x06,0xE0,0x1C,0x30,
    0x70,0x1C
},

{    /* ｾ PATTERN */
    0x0C,0x00,0x0C,0x00,0xFF,0xFC,0x0C,0x18,
    0x0C,0x30,0x0C,0x60,0x0C,0x00,0x0C,0x00,
    0x0F,0xFC
},

{    /* ｿ PATTERN */
    0x60,0x0C,0x30,0x0C,0x18,0x18,0x08,0x18,
    0x00,0x30,0x00,0x60,0x00,0xC0,0x01,0x80,
    0x1F,0x00
},

        /*      0xc0    */

{    /* ﾀ PATTERN */
    0x18,0x00,0x1F,0xFC,0x18,0x0C,0x18,0x0C,
    0x33,0x18,0x60,0xD8,0x00,0x30,0x00,0x60,
    0x0F,0xC0
},

{    /* ﾁ PATTERN */
    0x00,0xE0,0x0F,0x80,0x01,0x80,0x01,0x80,
    0x7F,0xFC,0x01,0x80,0x01,0x80,0x03,0x00,
    0x1E,0x00
},

{    /* ﾂ PATTERN */
    0x66,0x0C,0x66,0x0C,0x33,0x18,0x00,0x18,
    0x00,0x30,0x00,0x60,0x0C,0xC0,0x01,0x80,
    0x0F,0x00
},
```

保守／廃止

CRG.C

```
{    /* ﾃ PATTERN */
    0x3F,0xF0,0x00,0x00,0x00,0x00,0xFF,0xFC,
    0x03,0x00,0x03,0x00,0x03,0x00,0x06,0x00,
    0x1C,0x00
},

{    /* ﾄ PATTERN */
    0x06,0x00,0x06,0x00,0x06,0x00,0x07,0x80,
    0x06,0xE0,0x06,0x30,0x06,0x00,0x06,0x00,
    0x06,0x00
},

{    /* ﾅ PATTERN */
    0x03,0x00,0x03,0x00,0xFF,0xFC,0x03,0x00,
    0x03,0x00,0x03,0x00,0x06,0x00,0x0C,0x00,
    0x38,0x00
},

{    /* ﾆ PATTERN */
    0x00,0x00,0x00,0x00,0x0F,0xE0,0x00,0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x7F,0xFC
},

{    /* ﾇ PATTERN */
    0x00,0x00,0x3F,0xFC,0x00,0x18,0x00,0x30,
    0x0C,0x60,0x06,0x80,0x03,0x80,0x0E,0xE0,
    0x78,0x38
},

{    /* ﾈ PATTERN */
    0x03,0x00,0x03,0x00,0x7F,0xF8,0x00,0x30,
    0x00,0xE0,0x03,0xC0,0x1F,0x70,0x73,0x1C,
    0x03,0x00
},

{    /* ﾉ PATTERN */
    0x00,0x0C,0x00,0x0C,0x00,0x18,0x00,0x30,
    0x00,0x60,0x01,0xC0,0x03,0x00,0x06,0x00,
    0x3C,0x00
},

{    /* ﾊ PATTERN */
    0x18,0xC0,0x18,0xC0,0x18,0xC0,0x30,0x60,
    0x30,0x20,0x60,0x30,0x60,0x30,0xC0,0x18,
    0xC0,0x0C
},

{    /* ﾋ PATTERN */
    0x60,0x00,0x60,0x00,0x7F,0xF8,0x60,0x00,
    0x60,0x00,0x60,0x00,0x60,0x00,0x60,0x00,
    0x7F,0xFC
},

{    /* ﾌ PATTERN */
    0x00,0x00,0x7F,0xFC,0x00,0x0C,0x00,0x0C,
    0x00,0x18,0x00,0x30,0x00,0x60,0x00,0xC0,
    0x0F,0x80
},

{    /* ﾍ PATTERN */
    0x00,0x00,0x06,0x00,0x0F,0x00,0x19,0xC0,
    0x30,0x60,0x60,0x30,0x00,0x18,0x00,0x0C,
    0x00,0x00
},

{    /* ﾎ PATTERN */
    0x03,0x00,0x03,0x00,0x7F,0xF8,0x03,0x00,
```

CRG.C

```
    0x03,0x00,0x33,0x30,0x63,0x18,0x63,0x18,
    0x63,0x18
},

{    /* ﾏ PATTERN */
    0x00,0x00,0x7F,0xF8,0x00,0x30,0x00,0x60,
    0x0C,0xC0,0x07,0x80,0x03,0x00,0x01,0x80,
    0x00,0xC0
},
        /* 0xd0-        */

{    /* ﾐ PATTERN */
    0x7C,0x00,0x07,0x80,0x00,0xF0,0x1C,0x00,
    0x07,0x00,0x00,0x00,0x7C,0x00,0x07,0x80,
    0x00,0xF0
},

{    /* ﾑ PATTERN */
    0x01,0x80,0x03,0x00,0x06,0x00,0x0C,0x00,
    0x18,0x60,0x30,0x30,0x30,0xF8,0x63,0x8C,
    0x7E,0x0C
},

{    /* ﾒ PATTERN */
    0x00,0x30,0x00,0x30,0x18,0x60,0x0E,0xC0,
    0x03,0x80,0x03,0xC0,0x06,0x60,0x0C,0x30,
    0x38,0x00
},

{    /* ﾓ PATTERN */
    0x3F,0xF8,0x03,0x00,0x03,0x00,0x7F,0xFC,
    0x03,0x00,0x03,0x00,0x03,0x00,0x03,0x00,
    0x01,0xFC
},

{    /* ﾔ PATTERN */
    0x18,0x00,0x18,0x00,0x7F,0xFC,0x18,0x18,
    0x0C,0x30,0x0C,0x60,0x0C,0x00,0x06,0x00,
    0x06,0x00
},

{    /* ﾕ PATTERN */
    0x00,0x00,0x1F,0xE0,0x00,0x60,0x00,0x60,
    0x00,0x60,0x00,0x60,0x00,0x60,0x00,0x60,
    0x7F,0xFC
},

{    /* ﾖ PATTERN */
    0x00,0x00,0x7F,0xFC,0x00,0x0C,0x00,0x0C,
    0x3F,0xFC,0x00,0x0C,0x00,0x0C,0x00,0x0C,
    0x7F,0xFC
},

{    /* ﾗ PATTERN */
    0x1F,0xF0,0x00,0x00,0x7F,0xFC,0x00,0x0C,
    0x00,0x18,0x00,0x18,0x00,0x30,0x00,0xE0,
    0x0F,0x80
},

{    /* ﾘ PATTERN */
    0x30,0x18,0x30,0x18,0x30,0x18,0x30,0x18,
    0x30,0x18,0x00,0x30,0x00,0x30,0x00,0x60,
    0x03,0xC0
},

{    /* ﾙ PATTERN */
    0x18,0xC0,0x18,0xC0,0x18,0xC0,0x18,0xC0,
    0x18,0xC0,0x18,0xC0,0x30,0xCC,0x60,0xD8,
    0xC0,0xE0
```

CRG.C

```
},

{    /* ﾚ PATTERN */
    0x60,0x00,0x60,0x00,0x60,0x00,0x60,0x18,
    0x60,0x30,0x60,0x60,0x61,0xC0,0x67,0x00,
    0x7C,0x00
},

{    /* ﾛ PATTERN */
    0x00,0x00,0x7F,0xF8,0x60,0x18,0x60,0x18,
    0x60,0x18,0x60,0x18,0x60,0x18,0x60,0x18,
    0x7F,0xF8
},

{    /* ﾜ PATTERN */
    0x7F,0xFC,0x60,0x0C,0x60,0x0C,0x60,0x0C,
    0x00,0x18,0x00,0x18,0x00,0x30,0x00,0xE0,
    0x07,0x80
},

{    /* ﾝ PATTERN */
    0x78,0x00,0x1C,0x0C,0x00,0x0C,0x00,0x18,
    0x00,0x30,0x00,0x60,0x01,0xC0,0x07,0x00,
    0x7C,0x00
},

{    /* ﾞ PATTERN */
    0xD8,0x00,0x6C,0x00,0x36,0x00,0x00,0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00
},

{    /* ﾟ PATTERN */
    0x1C,0x00,0x36,0x00,0x36,0x00,0x1C,0x00,
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00
},

        /* 0xe0          */

WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,



        /* 0xf0-         */


WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
WHITE,
```

14

保守／廃止

CRG.C

```
WHITE,

{    /* はさみA PATTERN */
    0x03,0xFC,0x00,0xFE,0x00,0x7F,0x00,0x1F,
    0x00,0x03,0x00,0x1F,0x00,0x7F,0x00,0xFE,
    0x03,0xFC
},


{    /* はさみB PATTERN */
    0x03,0xF8,0x06,0x0C,0x3E,0x0C,0xFF,0xF8,
    0xFC,0x00,0xFF,0xF8,0x3E,0x0C,0x06,0x0C,
    0x03,0xF8
},

{    /* にぎりばさみ　A PATTERN */
    0x1F,0xFF,0x07,0x80,0x00,0xE0,0x00,0x38,
    0x00,0x3E,0x00,0xE3,0x07,0x80,0x1F,0xFF,
    0x00,0x00
},


{    /* にぎりばさみ　B PATTERN */
    0xFF,0xF0,0x00,0x38,0x7F,0x9C,0xC0,0xCC,
    0xC0,0xCC,0xFF,0x9C,0x00,0x38,0xFF,0xF0,
    0x00,0x00
},


{    /* 真ん中のてん */
    0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x03,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00
},


WHITE,
WHITE,
WHITE,

};
```

保守／廃止

## A.18　ASR.ASM

ASR.ASM

```
        PAGE    64,132
        TITLE   <ASR> **** ASMBLER SEND RECEIVE BIOS ****
        NAME    ASR
;*************************************************************
;*      #３２８　アセンブラ　送受信ＢＩＯＳ                 *
;*              For MSC ver 5.1 Small Model                  *
;*************************************************************
;
        ;.MODEL medium
;
        PUBLIC  _AMSNDI         ;割り込み送信時データ読み出し
        PUBLIC  _AMRCVI         ;割り込み受信時データ書き込み
;
        INCLUDE V25.MAC
        INCLUDE SEGMENT.MAC
;
        BUFSIZE         EQU     2000h   ;送受信バッファサイズ，ＳＲ．ｃ参照
        BUFSIZE_R       EQU     8000h   ;受信バッファサイズ，ＳＲ．ｃ参照

;=============================================================
        @DEFSEG
;=============================================================

;=============================================================
        @BSEG
;=============================================================


comm near _SR_BTOP:     word    ;バッファトップポインタ
comm near _SR_BBOT:     word    ;バッファボトムポインタ
comm near _SR_COUNT:    word    ;バッファカウンタ
;
comm near _SR_BTOP_G:   word    ;疑似バッファトップポインタ
comm near _SR_BBOT_G:   word    ;疑似バッファボトムポインタ
comm near _guiji:       dword   ;疑似バッファポインタ
;
SR_info_st      struc
        SR_BUF  db      BUFSIZE dup(?)  ;バッファ本体
        SR_ENDF db      ?               ;終了フラグ
SR_info_st      ends
EXTRN   _SR_info:word
;
comm near _SR_DEBUG:    byte    ;
comm near _SR_FILL:     byte    ;
comm near _SR_overflow: byte

;=============================================================
        @ENDBSEG
;=============================================================

;=============================================================
        @CSEG
;=============================================================

;*************************************************************
;*      送信バッファからデータを１バイト取り出す*
;*      AH...   0=データあり、0<>データ終り                  *
;*      AL...   データ本体                                   *
;*************************************************************
_AMSNDI PROC    far
;
        PUSH    SI
;
        mov     _SR_FILL,0
;
        CMP     _SR_COUNT,0             ;送信データあり？
        JNE     AMS10
;
        CMP     _SR_info.SR_ENDF,0              ;最終データ？
```

保守／廃止

ASR.ASM

```
        JNE     AMS00

        ;MOV    word ptr _ﾒｸﾙﾃﾞｰﾀｶﾞﾅｲ,1 ;送るべきデータが無い！！
        MOV     AH,2
        MOV     AL,0                    ;だみーデータ＝‘0’
        mov     _SR_FILL,1
        JMP     SHORT AMSEND

AMS00:  XOR     AX,AX                   ;最終データ
        MOV     AH,01H                  ;だみーデータ＝‘0’
        mov     _SR_FILL,1
        JMP     SHORT AMSEND
;
AMS10:  MOV     SI,_SR_BBOT             ;ポインタ読みだし
        CMP     SI,OFFSET DGROUP:_SR_info.SR_ENDP       ;ポインタチェック
        JC      AMS11
        MOV     SI,OFFSET DGROUP:_SR_info.SR_BUF        ;ポインタ初期化
        MOV     _SR_BBOT,SI

AMS11:  MOV     AL,[SI]                 ;送信データ読み出し
        XOR     AH,AH
        INC     _SR_BBOT                ;ポインタ加算
        DEC     _SR_COUNT               ;カウンタ減算
;
AMSEND: POP     SI
        RET
;
_AMSNDI ENDP

;*********************************************************
;*      受信バッファへデータを１バイト書き出す           *
;*        疑似ＳＲＡＭを使う版                          *
;*      AH...   なし                                     *
;*      AL...   データ本体                               *
;*********************************************************
_AMRCVI PROC    far
;
        PUSH    SI
        push    es
        push    bx
;
        CMP     _SR_COUNT,BUFSIZE_R     ;バッファフル？
        JC      AMR10

        mov     _SR_overflow,1
        JMP     SHORT AMREND
;
AMR10:
        mov     bx,word ptr _guiji+2    ;疑似バッファのセグメント
        mov     es,bx
        MOV     SI,_SR_BTOP_G           ;ポインタ読みだし

AMR11:  MOV     es:[SI],AL              ;受信データ書き込み
        INC     _SR_BTOP_G              ;ポインタ加算
        INC     _SR_COUNT               ;カウンタ加算
;
AMREND:
        pop     bx
        pop     es
        POP     SI
        RET
;
_AMRCVI ENDP



;*****************************************************************
        @ENDCSEG
;*****************************************************************
```

2

ASR.ASM

```
;
        END
```

## A.19　ACODE.ASM

ACODE.ASM

```
        PAGE    64,132
        TITLE   <ACODE> **** ASSEMBLER ENCODE Sub-PGM ****
        NAME    ACODE
;*****************************************************************
;*      FILE NAME       :ACODE.ASM              (ver.3.00)      *
;*                                                              *
;*      FUNCTION        :CODEC PGM                              *
;*                                                              *
;*****************************************************************
;
        ;.MODEL medium
;
        EXTRN   MHTBL_E:far             ;符号化変換テーブル
;
        INCLUDE V25.MAC
        INCLUDE SEGMENT.MAC
        INCLUDE V55PI.MAC
        INCLUDE V55PI_2.MAC
        INCLUDE V55PI_3.MAC
;
;       符号化方式
        MH      EQU     0               ;1次元符号化
        MR      EQU     1               ;2次元符号化(K=1)
        MMR     EQU     2               ;2次元符号化(K=無限大)

;       副走査線密度
        NORMAL  EQU     0               ;ノーマルモード
        FINE    EQU     1               ;ファインモード

;===============================================================
        @DEFSEG
;===============================================================

;===============================================================
        @DSEG
;===============================================================

comm near CODEC :byte   ;現在の符号化モード

comm near GANG  :byte   ;

;/*圧縮処理用ワークエリア，C言語と共有**/
comp_work_st    struc
        K       DB      ?       ;Kカウンタ
        L       DB      ?       ;Lカウンタ
        OD_DT   DW      ?       ;半端符号データセーブ領域
        OD_BT   DB      ?       ;半端符号データビット数セーブ領域
                db      ?       ;for word alignment
        TRBP    DW      ?       ;送信バッファオフセットポインタ
        TRBSZP  DW      ?       ;送信バッファ残りサイズセーブ領域
        ATRBP   DW      ?       ;符号データの格納領域のオフセットポインタ
        ADBP    DW      ?       ;入力情報の格納領域のオフセットポインタ
        ATR_LST DW      ?       ;1ライン処理開始時の出力情報のオフセットアドレスの記憶領域
        TR_LST  DW      ?       ;1ライン処理開始時の送信バッファのオフセットアドレスの記憶
領域
        R_cnt   DW      ?       ;RTCコード送信用カウンタ
        RTCDP   DW      5 dup(?);RTCコード格納領域
comp_work_st    ends
EXTRN   _comp_work:word
;/*圧縮処理用パラメータ**/
comp_param_st   struc
        gptr    dd ?    ;/* 画像データポインタ **/
        gcnt    dw ?    ;/* 画像データバイト数 **/
        clen    dw ?    ;/* 符号データバイト数，何の？ **/
        Kval_m1 db ?    ;/* K値－1 **/
        CODEm   db ?    ;/* 符号化方式 **/
        clen1L  dw ?    ;/* 1走査線符号データバイト数 **/
comp_param_st   ends
EXTRN   _comp_param:word
```

1

ACODE.ASM

```
EXTRN   _nama_data:DWORD        ;/**画像生データ**/
EXTRN   _comp_data:DWORD        ;/**画像圧縮データ**/
EXTRN   _trnp_tabl_A:DWORD      ;/**変化点テーブルＡのポインタ**/
EXTRN   _trnp_tabl_B:DWORD      ;/**変化点テーブルＢのポインタ**/
;NOTE: _trnp_tabl_A & _trnp_tabl_B have the same segment value.

$_TRBSZP        equ     2000h   ;送信バッファサイズ

;=================================================================
        @ENDDSEG
;=================================================================

;=================================================================
        @CSEG
;=================================================================

;*************************************************
;*      Ｃｏｄｅｃ初期設定                       *
;*************************************************
        public  _acode_INIT
_acode_INIT     PROC    far
;
        les     di,DWORD PTR _nama_data ; es <- input data area segment
        ;mov     dw,es:word ptr [di]
        mov     _comp_work.L,dl                 ; L <- line
        mov     _comp_work.K,0                  ; K=0
;====================================;
;       initialize
;====================================;
        mov     _comp_work.OD_DT,0
        mov     _comp_work.OD_BT,0
        mov     _comp_work.TRBSZP,$_TRBSZP
;
        cmp     _comp_param.CODEm,0
        jnz     @F
        mov     CODEC,MH
    jmp NODE
    @@:
        mov     CODEC,MR
    NODE:
;
        mov     GANG,0
;
        ret
;
_acode_INIT     ENDP

;*************************************************
;*      ＭＨ等倍符号化                           *
;*************************************************
        public  _acode_ENCODE0
        public  _acode_ENCODE0_0,_acode_ENCODE0_1,_acode_ENCODE0_2
_acode_ENCODE0  PROC    far
        @PUSHR
;
;====================================;
;       Entry                        ;
;====================================;
        mov     _comp_work.OD_DT,0
        mov     _comp_work.OD_BT,0
        mov     _comp_work.TRBSZP,$_TRBSZP
;
;====================================;
;       Make EOL                     ;
;====================================;
_acode_ENCODE0_0        label   near
        mov     bx,_comp_work.OD_DT     ; set parameter
        mov     ch,_comp_work.OD_BT
```

2

保守／廃止

ACODE.ASM

```
        mov     dx,0800h                ; EOL = 0800h
        mov     cl,12
        mov     bp,_comp_work.TRBSZP
        les     ax,_comp_data
        mov     di,ax                   ; di <- trans buffer offset

        V55_ALBIT       ;【ＡＬＢＩＴ】

        mov     _comp_work.OD_DT,bx     ; set parameter
        mov     _comp_work.OD_BT,ch
        mov     _comp_work.TRBP,di      ; di <- trans buffer offset
        mov     _comp_work.TRBSZP,bp

;=====================================;
;       Make Turning point table (1-line);
;=====================================;
_acode_ENCODE0_1        label   near
        les     di,_trnp_tabl_A                 ; turning point table
        mov     ax,es
        MOV_IRAM_AX     REGB_09+REGB_ES
        mov     ax,di
        MOV_IRAM_AX     REGB_09+REGB_DI

        les     di,_comp_param.gptr
        mov     ax,es
        MOV_IRAM_AX     REGB_09+REGB_DS
        mov     ax,di
        MOV_IRAM_AX     REGB_09+REGB_SS
        ;MOV_IRAM_IMMW  REGB_09+REGB_PS,216
        mov     ax,_comp_param.gcnt
        MOV_IRAM_AX     REGB_09+REGB_PS

        MOV_IRAM_IMMW   REGB_09+REGB_AX,0       ; 0 clear
        MOV_IRAM_IMMW   REGB_09+REGB_CX,0

        mov     al,09h                          ; bank number is 9
        V55_COLTRP      ;【ＣＯＬＴＲＰ】

;=====================================;
;       MH encode (1 line)              ;
;=====================================;
_acode_ENCODE0_2        label   near
        les     di,_trnp_tabl_A                 ; turning point table addr
        mov     ax,es
        MOV_IRAM_AX     REGB_08+REGB_DS
        mov     ax,di
        MOV_IRAM_AX     REGB_08+REGB_SS

        $1      equ     seg MHTBL_E             ; encode table
        MOV_IRAM_IMMW   REGB_08+REGB_ES,$1

        mov     bx,_comp_work.OD_DT     ; set parameter
        mov     ch,_comp_work.OD_BT
        les     ax,_comp_data           ; es <- trans buffer segment
        mov     di,_comp_work.TRBP      ; di <- trans buffer offset
        mov     bp,_comp_work.TRBSZP

        MOV_IRAM_IMMW   REGB_08+REGB_AX,0       ; 0 clear
        MOV_IRAM_IMMW   REGB_08+REGB_CX,0

    @@:
        mov     al,08h                          ; bank number is 8
        V55_MHENC       ;【ＭＨＥＮＣ】
        jc      @B

    ;=================================;
    ;   16bits alignment              ;
    ;=================================;
        cmp     ch,0
```

3

保守／廃止

ACODE.ASM

```
        jz      @F
        mov     dx,0000h                ; fill bits
        mov     cl,10h
        and     ch,0Fh
        sub     cl,ch
        V55_ALBIT       ;【ＡＬＢＩＴ】
    @@:
    ;===================================;
        mov     _comp_work.OD_DT,bx     ; set parameter
        mov     _comp_work.OD_BT,ch
        mov     _comp_work.TRBP,di      ; di <- trans buffer offset
        mov     _comp_work.TRBSZP,bp

        mov     ax,$_TRBSZP
        sub     ax,_comp_work.TRBSZP
        mov     _comp_param.clen1L,ax

        @POPR
        ret
;
_acode_ENCODE0  ENDP

;************************************************
;*      ＭＲ等倍符号化                          *
;************************************************
        public  _acode_ENCODE1
        public  _acode_ENCODE1_00,_acode_ENCODE1_10
        public   acode_ENCODE1_TURN_OVER
_acode_ENCODE1  PROC    far
        @PUSHR
;
;====================================;
;       Entry                        ;
;====================================;
        mov     _comp_work.OD_DT,0
        mov     _comp_work.OD_BT,0
        mov     _comp_work.TRBSZP,$_TRBSZP
;
    ; K parameter judg
        mov     ah,_comp_param.Kval_m1
        and     _comp_work.K,ah
        mov     al,_comp_work.K
        and     al,al
        jz      _acode_ENCODE1_00       ;１次元符号化
        jmp     _acode_ENCODE1_10       ;２次元符号化
;
;====================================;
;       Make EOL1D=EOL+1             ;
;====================================;
_acode_ENCODE1_00       label   near
        mov     bx,_comp_work.OD_DT     ; set parameter
        mov     ch,_comp_work.OD_BT
        mov     dx,1800h                ; EOL = 1800h
        mov     cl,13
        mov     bp,_comp_work.TRBSZP
        les     ax,_comp_data
        mov     di,ax                   ; di <- trans buffer offset

        V55_ALBIT       ;【ＡＬＢＩＴ】

        mov     _comp_work.OD_DT,bx     ; set parameter
        mov     _comp_work.OD_BT,ch
        mov     _comp_work.TRBP,di      ; di <- trans buffer offset
        mov     _comp_work.TRBSZP,bp

;====================================;
;       Make Turning point table (1-line);
;====================================;
_acode_ENCODE1_01       label   near
```

保守／廃止

ACODE.ASM

```
        les     di,_trnp_tabl_A                 ; turning point table
        mov     ax,es
        MOV_IRAM_AX     REGB_09+REGB_ES
        mov     ax,di
        MOV_IRAM_AX     REGB_09+REGB_DI

        les     di,_comp_param.gptr             ; img data buffer joho
        mov     ax,es
        MOV_IRAM_AX     REGB_09+REGB_DS
        mov     ax,di
        MOV_IRAM_AX     REGB_09+REGB_SS
        ;MOV_IRAM_IMMW  REGB_09+REGB_PS,216
        mov     ax,_comp_param.gcnt
        MOV_IRAM_AX     REGB_09+REGB_PS

        MOV_IRAM_IMMW   REGB_09+REGB_AX,0       ; 0 clear
        MOV_IRAM_IMMW   REGB_09+REGB_CX,0

        mov     al,09h                          ; bank number is 9
        V55_COLTRP      ;【ＣＯＬＴＲＰ】

;=====================================;
;       MH encode (i line)            ;
;=====================================;
_acode_ENCODE1_02       label   near
        les     di,_trnp_tabl_A                 ; turning point table addr
        mov     ax,es
        MOV_IRAM_AX     REGB_08+REGB_DS
        mov     ax,di
        MOV_IRAM_AX     REGB_08+REGB_SS

        $1      equ     seg MHTBL_E             ; encode table
        MOV_IRAM_IMMW   REGB_08+REGB_ES,$1

        mov     bx,_comp_work.OD_DT     ; set parameter
        mov     ch,_comp_work.OD_BT
        les     ax,_comp_data           ; es <- trans buffer segment
        mov     di,_comp_work.TRBP      ; di <- trans buffer offset
        mov     bp,_comp_work.TRBSZP

        MOV_IRAM_IMMW   REGB_08+REGB_AX,0       ; 0 clear
        MOV_IRAM_IMMW   REGB_08+REGB_CX,0

    @@:
        mov     al,08h                          ; bank number is 8
        V55_MHENC       ;【ＭＨＥＮＣ】
        jc      @B

    ;=================================;
    ;   16bits alignment              ;
    ;=================================;
        cmp     ch,0
        jz      @F
        mov     dx,0000h                ; fill bits
        mov     cl,10h
        and     ch,0Fh
        sub     cl,ch
        V55_ALBIT       ;【ＡＬＢＩＴ】
    @@:
    ;=================================;

        mov     _comp_work.OD_DT,bx     ; set parameter
        mov     _comp_work.OD_BT,ch
        mov     _comp_work.TRBP,di      ; di <- trans buffer offset
        mov     _comp_work.TRBSZP,bp
;
        JMP     _acode_ENCODE1_99
;;;
;=====================================;
```

5

保守／廃止

ACODE.ASM

```
;       Make EOL2D=EOL+0                ;
;=======================================;
_acode_ENCODE1_IO       label   near
        mov     bx,_comp_work.OD_DT     ; set parameter
        mov     ch,_comp_work.OD_BT
        mov     dx,0800h                ; EOL = 0800h
        mov     cl,13
        mov     bp,_comp_work.TRBSZP
        les     ax,_comp_data
        mov     di,ax                   ; di <- trans buffer offset

        V55_ALBIT       ;【ＡＬＢＩＴ】

        mov     _comp_work.OD_DT,bx     ; set parameter
        mov     _comp_work.OD_BT,ch
        mov     _comp_work.TRBP,di      ; di <- trans buffer offset
        mov     _comp_work.TRBSZP,bp

;=======================================;
;       Make Turning point table (1-line);
;=======================================;
_acode_ENCODE1_11       label   near
    ; K parameter judg
        mov     al,_comp_work.K
        test    al,00000001b
    jnz @F
        ;A-table is the coding line, B-table is the reference line
        les     di,_trnp_tabl_A                 ; turning point A table addr
        mov     ax,es
        MOV_IRAM_AX     REGB_09+REGB_ES
        mov     ax,di
        MOV_IRAM_AX     REGB_09+REGB_DI
    jmp NODE_1
    @@:
        ;B-table is the coding line, A-table is the reference line
        les     di,_trnp_tabl_B                 ; turning point B table addr
        mov     ax,es
        MOV_IRAM_AX     REGB_09+REGB_ES
        mov     ax,di
        MOV_IRAM_AX     REGB_09+REGB_DI
    NODE_1:

        les     di,_comp_param.gptr             ; img data buffer joho
        mov     ax,es
        MOV_IRAM_AX     REGB_09+REGB_DS
        mov     ax,di
        MOV_IRAM_AX     REGB_09+REGB_SS
        ;MOV_IRAM_IMMW  REGB_09+REGB_PS,216
        mov     ax,_comp_param.gcnt
        MOV_IRAM_AX     REGB_09+REGB_PS

        MOV_IRAM_IMMW   REGB_09+REGB_AX,0       ; 0 clear
        MOV_IRAM_IMMW   REGB_09+REGB_CX,0

        mov     al,09h                          ; bank number is 9
        V55_COLTRP      ;【ＣＯＬＴＲＰ】

;=======================================;
;       MR encode (1 line)              ;
;=======================================;
_acode_ENCODE1_12       label   near
    ; K parameter judg
        mov     al,_comp_work.K
        test    al,00000001b
    jnz @F
        ;A-table is the coding line, B-table is the reference line
        les     di,_trnp_tabl_A                 ; turning point A table addr
        mov     ax,es
        MOV_IRAM_AX     REGB_08+REGB_DS         ; seg
```

ACODE.ASM

```
        mov     ax,di
        MOV_IRAM_AX     REGB_08+REGB_SS         ; offset, coding line
        les     di,_trnp_tabl_B                 ; turning point B table addr
        mov     ax,di
        MOV_IRAM_AX     REGB_08+REGB_PS         ; offset, refference line
    jmp NODE_2
    @@:
        ;B-table is the coding line, A-table is the reference line
        les     di,_trnp_tabl_B                 ; turning point B table addr
        mov     ax,es
        MOV_IRAM_AX     REGB_08+REGB_DS         ; seg
        mov     ax,di
        MOV_IRAM_AX     REGB_08+REGB_SS         ; offset, coding line
        les     di,_trnp_tabl_A                 ; turning point A table addr
        mov     ax,di
        MOV_IRAM_AX     REGB_08+REGB_PS         ; offset, refference line
    NODE_2:

        $1      equ     seg MHTBL_E             ; encode table
        MOV_IRAM_IMMW   REGB_08+REGB_ES,$1

        mov     bx,_comp_work.OD_DT     ; set parameter
        mov     ch,_comp_work.OD_BT
        les     ax,_comp_data           ; es <- trans buffer segment
        mov     di,_comp_work.TRBP      ; di <- trans buffer offset
        mov     bp,_comp_work.TRBSZP

        MOV_IRAM_IMMW   REGB_08+REGB_AX,0       ; 0 clear
        MOV_IRAM_IMMW   REGB_08+REGB_CX,0

        pushf
        pop     ax
        test    ax,0200h        ;test IE
        jz      acode_ENCODE1_12_MRENC_2
  acode_ENCODE1_12_MRENC_1:     ;IE=1
    @@:
        mov     al,08h                          ; bank number is 8
          cli
        V55_MRENC       ;【M R E N C】
          sti
        jc      @B
   jmp    acode_ENCODE1_12_MRENC_3
  acode_ENCODE1_12_MRENC_2:     ;IE=0
    @@:
        mov     al,08h                          ; bank number is 8
        V55_MRENC       ;【M R E N C】
        jc      @B
  acode_ENCODE1_12_MRENC_3:

    ;=================================;
    ;   16bits alignment              ;
    ;=================================;
        cmp     ch,0
        jz      @F
        mov     dx,0000h                ; fill bits
        mov     cl,10h
        and     ch,0Fh
        sub     cl,ch
        V55_ALBIT       ;【A L B I T】
    @@:
    ;=================================;

        mov     _comp_work.OD_DT,bx     ; set parameter
        mov     _comp_work.OD_BT,ch
        mov     _comp_work.TRBP,di      ; di <- trans buffer offset
        mov     _comp_work.TRBSZP,bp
;
_acode_ENCODE1_99       label   near
;
```

7

ACODE.ASM

```
    ; K parameter incliment
        inc     _comp_work.K
        mov     ah,_comp_param.Kval_m1
        and     _comp_work.K,ah

        mov     ax,_comp_work.TRBSZP
        sub     ax,$_TRBSZP
        cmp     ax,976
        jnc     @F
acode_ENCODE1_TURN_OVER         label   near
        mov     _comp_work.K,0
        jmp     _acode_ENCODE1_00
   @@:

        mov     ax,$_TRBSZP
        sub     ax,_comp_work.TRBSZP
        mov     _comp_param.clen1L,ax

        @POPR
        ret
;
_acode_ENCODE1  ENDP

;************************************************
;*      ＭＭＲ等倍符号化                        *
;************************************************
        public  _acode_ENCODE2
_acode_ENCODE2  PROC    far
        ret
_acode_ENCODE2  ENDP

;************************************************
;************************************************
        public  _acode_RTCGET
_acode_RTCGET   PROC    far
;
        CMP     CODEC,MH
        JNZ     RTCMR

        @PUSHR
;=====================================;
;       Entry                         ;
;=====================================;
        mov     _comp_work.OD_DT,0
        mov     _comp_work.OD_BT,0              ;
        mov     _comp_work.TRBSZP,$_TRBSZP
;
;=====================================;
;       Make MHRTC=EOL*6              ;
;=====================================;
        mov     bx,_comp_work.OD_DT     ; set parameter
        mov     ch,_comp_work.OD_BT
        mov     dx,0800h                ; EOL = 0800h
        mov     cl,12
        mov     bp,_comp_work.TRBSZP
        les     ax,_comp_data
        mov     di,ax                   ; di <- trans buffer offset

        V55_ALBIT       ;【ＡＬＢＩＴ】
        V55_ALBIT       ;【ＡＬＢＩＴ】
        V55_ALBIT       ;【ＡＬＢＩＴ】
        V55_ALBIT       ;【ＡＬＢＩＴ】
        V55_ALBIT       ;【ＡＬＢＩＴ】
        V55_ALBIT       ;【ＡＬＢＩＴ】

    ;=================================;
    ;   16bits alignment              ;
    ;=================================;
        cmp     ch,0
```

8

ACODE.ASM

```
        jz      @F
        mov     dx,0000h                ; fill bits
        mov     cl,10h
        and     ch,0Fh
        sub     cl,ch
        V55_ALBIT       ;【A L B I T】
    @@:
    ;===================================;

        mov     _comp_work.OD_DT,bx     ; set parameter
        mov     _comp_work.OD_BT,ch
        mov     _comp_work.TRBP,di      ; di <- trans buffer offset
        mov     _comp_work.TRBSZP,bp

        mov     ax,$_TRBSZP
        sub     ax,_comp_work.TRBSZP
        mov     _comp_param.clen1L,ax   ;1走査線符号データバイト数

        @POPR

        Jmp     CODRTCGOOD
;
;
RTCMR:  CMP     CODEC,MR
        JNZ     RTCMMR
;
        @PUSHR
;====================================;
;       Entry                        ;
;====================================;
        mov     _comp_work.OD_DT,0
        mov     _comp_work.OD_BT,0
        mov     _comp_work.TRBSZP,$_TRBSZP
;
;====================================;
;       Make MRRTC=(EOL+1)*6         ;
;====================================;
        mov     bx,_comp_work.OD_DT     ; set parameter
        mov     ch,_comp_work.OD_BT
        mov     dx,1800h                ; EOL = 1800h
        mov     cl,13
        mov     bp,_comp_work.TRBSZP
        les     ax,_comp_data
        mov     di,ax                   ; di <- trans buffer offset

        V55_ALBIT       ;【A L B I T】
        V55_ALBIT       ;【A L B I T】
        V55_ALBIT       ;【A L B I T】
        V55_ALBIT       ;【A L B I T】
        V55_ALBIT       ;【A L B I T】
        V55_ALBIT       ;【A L B I T】

    ;===================================;
    ;   16bits alignment                ;
    ;===================================;
        cmp     ch,0
        jz      @F
        mov     dx,0000h                ; fill bits
        mov     cl,10h
        and     ch,0Fh
        sub     cl,ch
        V55_ALBIT       ;【A L B I T】
    @@:
    ;===================================;

        mov     _comp_work.OD_DT,bx     ; set parameter
        mov     _comp_work.OD_BT,ch
        mov     _comp_work.TRBP,di      ; di <- trans buffer offset
        mov     _comp_work.TRBSZP,bp
```

9

保守／廃止

ACODE.ASM

```
        mov     ax,$_TRBSZP
        sub     ax,_comp_work.TRBSZP
        mov     _comp_param.clen1L,ax

        @POPR
;
        Jmp     CODRTCGOOD
;;;
RTCMMR: CMP     CODEC,MMR
        JNZ     CODRTCERR
;
        @PUSHR
;=====================================;
;       Entry                         ;
;=====================================;
        mov     _comp_work.OD_DT,0
        mov     _comp_work.OD_BT,0
        mov     _comp_work.TRBSZP,$_TRBSZP
;
;=====================================;
;       Make MRRTC=(EOL+1)*2          ;
;=====================================;
        mov     bx,_comp_work.OD_DT     ; set parameter
        mov     ch,_comp_work.OD_BT
        mov     dx,1800h                ; EOL = 1800h
        mov     cl,13
        mov     bp,_comp_work.TRBSZP
        les     ax,_comp_data
        mov     di,ax                   ; di <- trans buffer offset

        V55_ALBIT       ;【A L B I T】
        V55_ALBIT       ;【A L B I T】

    ;=================================;
    ;   16bits alignment              ;
    ;=================================;
        cmp     ch,0
        jz      @F
        mov     dx,0000h                ; fill bits
        mov     cl,10h
        and     ch,0Fh
        sub     cl,ch
        V55_ALBIT       ;【A L B I T】
    @@:
    ;=================================;

        mov     _comp_work.OD_DT,bx     ; set parameter
        mov     _comp_work.OD_BT,ch
        mov     _comp_work.TRBP,di      ; di <- trans buffer offset
        mov     _comp_work.TRBSZP,bp

        mov     ax,$_TRBSZP
        sub     ax,_comp_work.TRBSZP
        mov     _comp_param.clen1L,ax

        @POPR
;
        Jmp     CODRTCGOOD
;
CODRTCGOOD:
CODRTCERR:

        ret
;
_acode_RTCGET   ENDP

;*************************************************
;*      その他サブルーチン                       *
```

10

ACODE.ASM

```
;************************************************
;今は昔
        PUBLIC  _ACODE
_ACODE  PROC    far
        ret
_ACODE  ENDP

;================================================================
        @ENDCSEG
;================================================================
;
        END
```

11

保守／廃止

## A.20　ADECODE.ASM

ADECODE.ASM

```
        PAGE    64,132
        TITLE   <ADECODE> **** ASSEMBLER DECODE Sub-PGM ****
        NAME    ADECODE
;*******************************************************************
;*      FILE NAME       :ADECODE.ASM            (ver.3.00)      *
;*                                                              *
;*      FUNCTION        :DECODE BIOS                            *
;*                                                              *
;*******************************************************************
;
        EXTRN   MHTBL_D:far             ;復号化変換テーブル
;
        INCLUDE V25.MAC
        INCLUDE SEGMENT.MAC
        INCLUDE V55PI.MAC
        INCLUDE V55PI_2.MAC
        INCLUDE V55PI_3.MAC
;
;       符号化方式
        MH      EQU     0               ;1次元符号化
        MR      EQU     1               ;2次元符号化(K=1)
        MMR     EQU     2               ;2次元符号化(K=無限大)

;       副走査線密度
        FINE    EQU     0               ;ファインモード
        NORMAL  EQU     1               ;ノーマルモード

        EXTRN   _MIRRORTB:BYTE

;================================================================
        @DEFSEG
;================================================================

;================================================================
        @DSEG
;================================================================

COMM NEAR BANK_cont:    word:16 ;
COMM NEAR BANK_stak:    word:16 ;

;/*伸張処理用ワークエリア, A語と共有**/
decomp_work_st  struc
        K       db ?    ;/*Kカウンタ**/
                db ?    ;/*for alignment**/
        OD_DT   dw ?    ;/*半端符号データセーブ領域**/
        OD_BT   db ?    ;/*半端符号データビット数セーブ領域**/
                db ?    ;/*for alignment**/
        CDBP    dw ?    ;/*入力情報の格納領域のオフセットポインタ**/
        RVBP    dw ?    ;/*受信バッファオフセットポインタ**/
        RVBSZP  dw ?    ;/*受信バッファサイズ**/
        APBP    dw ?    ;/*全プリントデータのオフセットポインタ**/
        EOLFLG  dw ?    ;/*ＥＯＬ検出フラグ**/
        sequ    dw ?    ;/*シーケンス**/
        TRNP_SEG dw ?   ;/*当該変化点テーブルセグ**/
        TRNP_OFS dw ?   ;/*当該変化点テーブルオフ**/
decomp_work_st  ends
EXTRN   _expn_work:word
;/*伸張処理用パラメータ**/
decomp_param_st struc
        gptr    dd ?    ;/* 画像データポインタ **/
        gcnt    dw ?    ;/* 画像データバイト数 **/
        cptr    dd ?    ;/* 符号データポインタ **/
        clen    dw ?    ;/* 符号データバイト数 **/
        clen_rest dw ?  ;/* 符号データ残りバイト数 **/
        Kval_m1 db ?    ;/* K値－1 **/
        CODEm   db ?    ;/* 符号化方式 **/
        clen1L  dw ?    ;/* 1走査線符号データバイト数 **/
        RTCFLG  dw ?    ;/*ＲＴＣ検出フラグ**/
        stat    db ?    ;/* 終了状態ステータス＋ **/
```

ADECODE.ASM

```
decomp_param_st ends
EXTRN   _expn_param:word

EXTRN   _nama_data:DWORD        ;/**画像生データ**/
EXTRN   _comp_data:DWORD        ;/**画像圧縮データ**/
EXTRN   _trnp_tabl_A:DWORD      ;/**変化点テーブルＡのポインタ**/
EXTRN   _trnp_tabl_B:DWORD      ;/**変化点テーブルＢのポインタ**/
;NOTE: _trnp_tabl_A & _trnp_tabl_B have the same segment value.

COMM NEAR AX_mm:        word    ;ＡＸ
COMM NEAR BX_mm:        word    ;ＢＸ
COMM NEAR CX_mm:        word    ;ＣＸ
COMM NEAR DX_mm:        word    ;ＤＸ

;================================================================
        @ENDDSEG
;================================================================

;================================================================
        @CSEG
;================================================================

;*************************************************
;*      機能０　ＣＥＰ初期設定                  *
;*************************************************
        public  _adecode_DECINI
_adecode_DECINI PROC    far
;
;=====================================;
;       Initialize                    ;
;=====================================;
        mov     _expn_work.sequ,0       ;シーケンスを初期化

        mov     ax,0
        MOV_IRAM_AX     REGB_10+REGB_DS2
        MOV_IRAM_AX     REGB_10+REGB_DS3
        MOV_IRAM_AX     REGB_10+REGB_PSW_esc
        MOV_IRAM_AX     REGB_10+REGB_PC_esc
        MOV_IRAM_AX     REGB_10+REGB_DS
        MOV_IRAM_AX     REGB_10+REGB_SS
        MOV_IRAM_AX     REGB_10+REGB_PS
        MOV_IRAM_AX     REGB_10+REGB_ES
        MOV_IRAM_AX     REGB_10+REGB_DI
        MOV_IRAM_AX     REGB_10+REGB_SI
        MOV_IRAM_AX     REGB_10+REGB_BP
        MOV_IRAM_AX     REGB_10+REGB_SP
        MOV_IRAM_AX     REGB_10+REGB_BX
        MOV_IRAM_AX     REGB_10+REGB_DX
        MOV_IRAM_AX     REGB_10+REGB_CX
        MOV_IRAM_AX     REGB_10+REGB_AX

        mov     _expn_work.OD_DT,0
        mov     _expn_work.OD_BT,0

        mov     _expn_work.K,0FFh

        mov     _expn_param.clen_rest,0

        mov     dx,_expn_work.OD_DT
        MOV_IRAM_DX     REGB_10+REGB_PSW_esc
        mov     dl,_expn_work.OD_BT
        mov     dh,0
        MOV_IRAM_DX     REGB_10+REGB_PC_esc


        MOV_IRAM_IMMW   REGB_10+REGB_DS,4000h
        MOV_IRAM_IMMW   REGB_10+REGB_SS,0000h
        MOV_IRAM_IMMW   REGB_10+REGB_PS,0000h
        MOV_AX_IRAM     REGB_10+REGB_PS
```

2

保守／廃止

ADECODE.ASM

```
        mov     _expn_param.clen,ax

        call    Bank_save

        ret
;
_adecode_DECINI ENDP
;
;************************************************
;*      機能1   MH等倍復号化                    *
;************************************************
        public  _adecode_DECMH
        public  _adecode_DECMH_00,                      _adecode_DECMH_02
        public  _adecode_DECMH_03
        public  _adecode_DECMH_F0
        public   adecode_DECMH_03_A, adecode_DECMH_03_B, adecode_DECMH_03_C
        public   adecode_DECMH_03_D, adecode_DECMH_03_E, adecode_DECMH_03_F
_adecode_DECMH  PROC    far
        @PUSHR
;=====================================;
;       Entry                         ;
;=====================================;
;
;=====================================;
;             Search EOL              ;
;=====================================;
_adecode_DECMH_00:
        cmp     _expn_work.sequ,02h     ;? sequ-2
        jc      @F
        jmp     _adecode_DECMH_02
    @@:
;☆
        cmp     _expn_param.clen_rest,00
        jnz     @F
        mov     bx,_expn_param.clen
        MOV_AX_IRAM     REGB_10+REGB_PS
        add     ax,bx
        MOV_IRAM_AX     REGB_10+REGB_PS
    @@:

        cmp     _expn_work.sequ,00h     ;? sequ-0
        jnz     @F
        MOV_IRAM_IMMW   REGB_10+REGB_AX,0
        mov     _expn_work.sequ,01h     ;Next seq
    @@:

        call    BANK_probe
        call    Bank_save

        mov     ah,00h
        mov     al,0ah                          ; bank number is 10
        V55_SCHEOL      ;【ＳＣＨＥＯＬ】
        nop

        call    BANK_probe

        call    get_NZ  ;bx <- NZflag
        call    get_CY  ;cx <- CYflag
        mov     _expn_param.clen_rest,bx

        jc      @F
        MOV_AX_IRAM     REGB_10+REGB_PS
        mov     _expn_param.clen,ax
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,03h     ;OK, Next seq, pass 02h seq
        jmp     _adecode_DECMH_END
    @@:
        ;CY=1
        cmp     ah,03h
```

3

保守／廃止

ADECODE.ASM

```
        jz      adecode_DECMH_00_B
        cmp     ah,00h
        jz      adecode_DECMH_00_A
        jmp     adecode_DECMH_00_C
adecode_DECMH_00_A:
        call    Bank_load
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,01h     ;NG, Again
        jmp     _adecode_DECMH_END
adecode_DECMH_00_B:
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,03h     ;NG but OK, Next seq, pass 02h seq
        jmp     _adecode_DECMH_END
adecode_DECMH_00_C:
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,01h     ;OK
        jmp     _adecode_DECMH_END

;=====================================;
;           Get TAG (1bit)            ;
;=====================================;
_adecode_DECMH_02:
        cmp     _expn_work.sequ,03h     ;? sequ-3
        jnc     _adecode_DECMH_03
;☆
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,03h     ;Next seq
        jmp     _adecode_DECMH_END

;=====================================;
;         MH decode (1 line)          ;
;=====================================;
_adecode_DECMH_03:
        cmp     _expn_work.sequ,05h     ;? sequ-5
        jc      @F
        jmp     _adecode_DECMH_F0
    @@:
;☆
        cmp     _expn_param.clen_rest,00
        jnz     @F
        mov     bx,_expn_param.clen
        MOV_AX_IRAM     REGB_10+REGB_PS
        add     ax,bx
        MOV_IRAM_AX     REGB_10+REGB_PS
    @@:

        cmp     _expn_work.sequ,03h     ;? sequ-3
        jnz     @F
        les     di,_trnp_tabl_A         ; turning point table addr
        mov     ax,es
        MOV_IRAM_AX     REGB_10+REGB_ES ; turning point table segment
        mov     _expn_work.TRNP_SEG,ax
        mov     ax,di
        MOV_IRAM_AX     REGB_10+REGB_DI ; turning point table offset
        mov     _expn_work.TRNP_OFS,ax
        MOV_IRAM_IMMW   REGB_10+REGB_SI,1728    ;1728bit
        $1      equ     seg MHTBL_D             ; decode table
        MOV_IRAM_IMMW   REGB_10+REGB_BP,$1
        MOV_IRAM_IMMW   REGB_10+REGB_CX,0       ; 0ークリア
        MOV_IRAM_IMMW   REGB_10+REGB_AX,0
        mov     _expn_work.sequ,04h     ;Next seq
    @@:

        call    BANK_probe
        call    Bank_save

        mov     ah,00h
        mov     al,0ah                          ; bank number is 10
        V55_MHDEC       ;【MHDEC】
```

4

ADECODE.ASM

```
        nop

        call    BANK_probe

        call    get_NZ  ;bx <- NZflag
        call    get_CY  ;cx <- CYflag
        mov     _expn_param.clen_rest,bx

        jc      @F
        ;CY=0
        mov     _expn_work.sequ,00h     ;Next seq
        jmp     _adecode_DECMH_F0
    @@:
        ;CY=1
        cmp     ah,05h
        jz      adecode_DECMH_03_F
        cmp     ah,04h
        jz      adecode_DECMH_03_E
        cmp     ah,03h
        jz      adecode_DECMH_03_D
        cmp     ah,02h
        jz      adecode_DECMH_03_C
        cmp     ah,01h
        jz      adecode_DECMH_03_B
        cmp     ah,00h
        jz      adecode_DECMH_03_A
        jmp     adecode_DECMH_03_F
adecode_DECMH_03_A:
        call    Bank_load
         MOV_AX_IRAM    REGB_10+REGB_PS
         cmp    ax,2000h                ;xx-2000h
         jnc    adecode_DECMH_03_F
        mov     _expn_param.stat,0
        jmp     _adecode_DECMH_END
adecode_DECMH_03_B:
        mov     _expn_param.stat,20h
        mov     _expn_param.RTCPLG,1
        jmp     _adecode_DECMH_END
adecode_DECMH_03_C:
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,00h     ;Next seq
        jmp     _adecode_DECMH_END
adecode_DECMH_03_D:
        jmp     adecode_DECMH_03_F
adecode_DECMH_03_E:
        jmp     adecode_DECMH_03_F
adecode_DECMH_03_F:
        MOV_AX_IRAM     REGB_10+REGB_PS
        mov     _expn_param.clen,ax
        mov     _expn_param.stat,03h
        mov     _expn_work.sequ,00h     ;Again
        jmp     _adecode_DECMH_END

;=====================================;
; Convert turning point into image data ;
;=====================================;
_adecode_DECMH_F0:
        mov     ax,_expn_work.TRNP_SEG
        MOV_IRAM_AX     REGB_11+REGB_DS
        mov     ax,_expn_work.TRNP_OFS
        MOV_IRAM_AX     REGB_11+REGB_SS
        les     di,_expn_param.gptr
        mov     ax,es
        MOV_IRAM_AX     REGB_11+REGB_ES
        mov     ax,di
        MOV_IRAM_AX     REGB_11+REGB_DI
        MOV_IRAM_IMMW   REGB_11+REGB_CX,0
        MOV_IRAM_IMMW   REGB_11+REGB_AX,0
```

5

保守／廃止

ADECODE.ASM

```
        mov     al,0bh                          ; bank number is 11
        mov     ah,00h
        V55_CNVTRP      ;【ＣＮＶＴＲＰ】
        nop

        mov     _expn_param.stat,10h
        mov     _expn_work.sequ,00h     ;Next seq
        jmp     _adecode_DECMH_END
_adecode_DECMH_END:
;=====================================;
;       Exit                          ;
;=====================================;
        @POPR
        ret
;
_adecode_DECMH  ENDP
;
;************************************************
;*      機能３　ＭＲ等倍復号化                 *
;************************************************
        public _adecode_DECMR
_adecode_DECMR  PROC    far
;
        include adecoMR.asm
;
_adecode_DECMR  ENDP

;************************************************
;*      機能５　ＭＭＲ等倍復号化               *
;************************************************
        public _adecode_DECMMR
_adecode_DECMMR PROC    far
;
        ret
;
_adecode_DECMMR ENDP
;
;************************************************
;************************************************
get_NZ  proc    near
        jnz     @F
        mov     bx,0
        ret
    @@:
        mov     bx,1
        ret
get_NZ  endp
;
get_CY  proc    near
        jc      @F
        mov     cx,0
        ret
    @@:
        mov     cx,1
        ret
get_CY  endp
;
;デバグ用のバンク１０モニタ
        public  BANK_probe
BANK_probe      proc    near
        ret
BANK_probe      endp
;
        public  BANK_save
BANK_save       proc    near
        push    ax
        MOV_AX_IRAM     REGB_10+00h
        mov     BANK_stak+00,ax
        MOV_AX_IRAM     REGB_10+02h
```

ADECODE.ASM

```
        mov     BANK_stak+02h,ax
        MOV_AX_IRAM     REGB_10+04h
        mov     BANK_stak+04h,ax
        MOV_AX_IRAM     REGB_10+06h
        mov     BANK_stak+06h,ax
        MOV_AX_IRAM     REGB_10+08h
        mov     BANK_stak+08h,ax
        MOV_AX_IRAM     REGB_10+0Ah
        mov     BANK_stak+0Ah,ax
        MOV_AX_IRAM     REGB_10+0Ch
        mov     BANK_stak+0Ch,ax
        MOV_AX_IRAM     REGB_10+0Eh
        mov     BANK_stak+0Eh,ax
        MOV_AX_IRAM     REGB_10+10h
        mov     BANK_stak+10h,ax
        MOV_AX_IRAM     REGB_10+12h
        mov     BANK_stak+12h,ax
        MOV_AX_IRAM     REGB_10+14h
        mov     BANK_stak+14h,ax
        MOV_AX_IRAM     REGB_10+16h
        mov     BANK_stak+16h,ax
        MOV_AX_IRAM     REGB_10+18h
        mov     BANK_stak+18h,ax
        MOV_AX_IRAM     REGB_10+1Ah
        mov     BANK_stak+1Ah,ax
        MOV_AX_IRAM     REGB_10+1Ch
        mov     BANK_stak+1Ch,ax
        MOV_AX_IRAM     REGB_10+1Eh
        mov     BANK_stak+1Eh,ax
        pop     ax
        ret
BANK_save       endp
;
        public  BANK_load
BANK_load       proc    near
        push    ax
        mov     ax,BANK_stak+00h
        MOV_IRAM_AX     REGB_10+00h
        mov     ax,BANK_stak+02h
        MOV_IRAM_AX     REGB_10+02h
        mov     ax,BANK_stak+04h
        MOV_IRAM_AX     REGB_10+04h
        mov     ax,BANK_stak+06h
        MOV_IRAM_AX     REGB_10+06h
        mov     ax,BANK_stak+08h
        MOV_IRAM_AX     REGB_10+08h
        mov     ax,BANK_stak+0Ah
        MOV_IRAM_AX     REGB_10+0Ah
        mov     ax,BANK_stak+0Ch
        MOV_IRAM_AX     REGB_10+0Ch
        mov     ax,BANK_stak+0Eh
        MOV_IRAM_AX     REGB_10+0Eh
        mov     ax,BANK_stak+10h
        MOV_IRAM_AX     REGB_10+10h
        mov     ax,BANK_stak+12h
        MOV_IRAM_AX     REGB_10+12h
        mov     ax,BANK_stak+14h
        MOV_IRAM_AX     REGB_10+14h
        mov     ax,BANK_stak+16h
        MOV_IRAM_AX     REGB_10+16h
        mov     ax,BANK_stak+18h
        MOV_IRAM_AX     REGB_10+18h
        mov     ax,BANK_stak+1Ah
        MOV_IRAM_AX     REGB_10+1Ah
        mov     ax,BANK_stak+1Ch
        MOV_IRAM_AX     REGB_10+1Ch
        mov     ax,BANK_stak+1Eh
        MOV_IRAM_AX     REGB_10+1Eh
        pop     ax
```

7

ADECODE.ASM

```
        ret
BANK_load       endp
;
;*************************************************
;*************************************************
        PUBLIC  _ADECODE
_ADECODE        PROC    far
        ret
_ADECODE        ENDP

;=============================================================
        @ENDCSEG
;=============================================================
;
        END
```

8

## A.21　SLEEP.ASM

SLEEP.ASM

```
;
;*********************************************
;       in....無し                           *
;       out...無し                           *
;*********************************************
;
        ;.MODEL medium
;
        INCLUDE SEGMENT.MAC
        INCLUDE V25.MAC
;
        INCLUDE MC329.MAC
;
        PUBLIC  _SLEEP
        PUBLIC  _cpu_halt
;
extrn   _v55reg:dword
extrn   _PLA:dword
;-------------------------------------------------------
        @DEFSEG
;-------------------------------------------------------

;-------------------------------------------------------
        @CSEG
;-------------------------------------------------------
;
_SLEEP  PROC    far
        @ENTASM
;
        CLI
        PUSH    ES
        PUSH    BX
        PUSH    AX

        les     bx,DWORD PTR _PLA
        MOV     BX,0                    ;
        MOV     AL,ES:[BX]
        and     AL,01111111b            ;
        MOV     ES:[BX],AL              ;

        les     bx,DWORD PTR _v55reg
        MOV     AL,ES:[BX].P0
        OR      AL,01h
        MOV     ES:[BX].P0,AL
        RSTWDT

        @STOP

        AND     AL,0FEh
        MOV     ES:[BX].P0,AL

        les     bx,DWORD PTR _PLA
        MOV     BX,0
        MOV     AL,ES:[BX]
        or      AL,10000000b
        MOV     ES:[BX],AL

        POP     AX
        POP     BX
        POP     ES
        STI
;
        @RETASM
;
_SLEEP  ENDP
;
_cpu_halt       PROC    far
;
        NOP
```

1

SLEEP.ASM

```
        hlt
;
        ret
;
_cpu_halt       ENDP
;
;/*************************************************
; *      メモリ間の複写                           *
; *      in...   unsigned char *srci,*dsti        *
; *        ...   unsigned short cnti              *
; *      out..   なし                             *
; *************************************************/
; Line 1
; Line 8
        PUBLIC  _mcopy
_mcopy  PROC FAR
        push    bp
        mov     bp,sp
        push    di
        push    si
;       src = 6
;       dst = 8
;       cnt = 10
        mov     dx,WORD PTR [bp+10]     ;cnt
; Line 11
        or      dx,dx
        je      $EX161
        mov     si,WORD PTR [bp+6]      ;src
        mov     di,WORD PTR [bp+8]      ;dst
$F162:
        lodsb
        inc     di
        mov     BYTE PTR [di-1],al
        dec     dx
        jne     $F162
; Line 12
$EX161:
        pop     si
        pop     di
        mov     sp,bp
        pop     bp
        ret

_mcopy  ENDP
;-------------------------------------------------------
        @ENDCSEG
;-------------------------------------------------------
        END
```

2

## A.22　MHTBL_E.ASM

MHTBL_E.ASM

```
;*********************************************
;*      MH encode.(compression) table        *
;*         data -> compression code          *
;*===========================================*
;*             22-25561 : R.Yabui            *
;*                        1990.01.05         *
;*                        1990.03.23         *
;*                        1990.06.23         *
;*********************************************
;
        INCLUDE SEGMENT.MAC
;
;       address= +XXXXh
;
;       -1+1+----6bit---+1-      -----8bit(H)----+----8bit(L)-----
;       +-+-+-+-+-+-+-+-+-+      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
;       |a|b|     c     |0| ===> |  code length  |               |
;       +-+-+-+-+-+-+-+-+-+      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
;
;       * Run Length >= 64
;        Run_Length = 64T +T
;        a: 0 = white
;           1 = black
;        b: 0 = terminate --- c = T
;           1 = makeup    --- c = M
;
;       * Run Length < 64
;        Only change the terminate code
;
if(0)
code    segment at      0FC00h
        ORG     2000h
else
;-------------------------------------------------------
        ;@CSEG
        @SEGMHTBL_E
;-------------------------------------------------------
        public  MHTBL_E
MHTBL_E proc    far
        ORG     0000h
endif
;++++++++++ White Encode Table ++++++++++
;-------------- Terminate ---------------
        db      0ACh, 008h              ; address= +0000h
        db      038h, 006h
        db      00Eh, 004h
        db      001h, 004h
        db      00Dh, 004h
        db      003h, 004h
        db      007h, 004h
        db      00Fh, 004h
        db      019h, 005h              ; address= +0010h
        db      005h, 005h
        db      01Ch, 005h
        db      002h, 005h
        db      004h, 006h
        db      030h, 006h
        db      00Bh, 006h
        db      02Bh, 006h
        db      015h, 006h              ; address= +0020h
        db      035h, 006h
        db      072h, 007h
        db      018h, 007h
        db      008h, 007h
        db      074h, 007h
        db      060h, 007h
        db      010h, 007h
        db      00Ah, 007h              ; address= +0030h
        db      06Ah, 007h
```

1

保守／廃止

MHTBL_E.ASM

```
        db      064h, 007h
        db      012h, 007h
        db      00Ch, 007h
        db      040h, 008h
        db      0C0h, 008h
        db      058h, 008h
        db      0D8h, 008h              ; address= +0040h
        db      048h, 008h
        db      0C8h, 008h
        db      028h, 008h
        db      0A8h, 008h
        db      068h, 008h
        db      0E8h, 008h
        db      014h, 008h
        db      094h, 008h              ; address= +0050h
        db      054h, 008h
        db      0D4h, 008h
        db      034h, 008h
        db      0B4h, 008h
        db      020h, 008h
        db      0A0h, 008h
        db      050h, 008h
        db      0D0h, 008h              ; address= +0060h
        db      04Ah, 008h
        db      0CAh, 008h
        db      02Ah, 008h
        db      0AAh, 008h
        db      024h, 008h
        db      0A4h, 008h
        db      01Ah, 008h
        db      09Ah, 008h              ; address= +0070h
        db      05Ah, 008h
        db      0DAh, 008h
        db      052h, 008h
        db      0D2h, 008h
        db      04Ch, 008h
        db      0CCh, 008h
        db      02Ch, 008h
;-------------- Terminate ---------------
        db      000h, 000h              ; address= +8000h / no data
        db      01Bh, 005h
        db      009h, 005h
        db      03Ah, 006h
        db      076h, 007h
        db      06Ch, 008h
        db      0ECh, 008h
        db      026h, 008h
        db      0A6h, 008h              ; address= +0090h
        db      016h, 008h
        db      0E6h, 008h
        db      033h, 009h
        db      0B3h, 009h
        db      04Bh, 009h
        db      0CBh, 009h
        db      02Bh, 009h
        db      0ABh, 009h              ; address= +00A0h
        db      06Bh, 009h
        db      0EBh, 009h
        db      01Bh, 009h
        db      09Bh, 009h
        db      05Bh, 009h
        db      0DBh, 009h
        db      019h, 009h
        db      099h, 009h              ; address= +00B0h
        db      059h, 009h
        db      006h, 006h
        db      0D9h, 009h
        db      010h, 00Bh
        db      030h, 00Bh
```

2

METBL_E.ASM

```
        db      0B0h, 00Bh
        db      048h, 00Ch
        db      0C8h, 00Ch              ; address= +00C0h
        db      028h, 00Ch
        db      0A8h, 00Ch
        db      068h, 00Ch
        db      0E8h, 00Ch
        db      038h, 00Ch
        db      0B8h, 00Ch
        db      078h, 00Ch
        db      0F8h, 00Ch              ; address= +00D0h

;++++++++++ Black Encode Table ++++++++++
if(0)
        ORG     2100h
else
        ORG     0100h
endif
;-------------- Terminate ---------------
        db      0ECh, 00Ah              ; address= +0100h
        db      002h, 003h
        db      003h, 002h
        db      001h, 002h
        db      006h, 003h
        db      00Ch, 004h
        db      004h, 004h
        db      018h, 005h
        db      028h, 006h              ; address= +0110h
        db      008h, 006h
        db      010h, 007h
        db      050h, 007h
        db      070h, 007h
        db      020h, 008h
        db      0E0h, 008h
        db      018h, 009h
        db      0E8h, 00Ah              ; address= +0120h
        db      018h, 00Ah
        db      010h, 00Ah
        db      0E6h, 00Bh
        db      016h, 00Bh
        db      036h, 00Bh
        db      0ECh, 00Bh
        db      014h, 00Bh
        db      0E8h, 00Bh              ; address= +0130h
        db      018h, 00Bh
        db      053h, 00Ch
        db      0D3h, 00Ch
        db      033h, 00Ch
        db      0B3h, 00Ch
        db      016h, 00Ch
        db      096h, 00Ch
        db      056h, 00Ch              ; address= +0140h
        db      0D6h, 00Ch
        db      04Bh, 00Ch
        db      0CBh, 00Ch
        db      02Bh, 00Ch
        db      0ABh, 00Ch
        db      06Bh, 00Ch
        db      0EBh, 00Ch
        db      036h, 00Ch              ; address= +0150h
        db      0B6h, 00Ch
        db      05Bh, 00Ch
        db      0DBh, 00Ch
        db      02Ah, 00Ch
        db      0AAh, 00Ch
        db      06Ah, 00Ch
        db      0EAh, 00Ch
        db      026h, 00Ch              ; address= +0160h
        db      0A6h, 00Ch
```

3

MHTBL_E.ASM

```
        db      04Ah, 00Ch
        db      0CAh, 00Ch
        db      024h, 00Ch
        db      0ECh, 00Ch
        db      01Ch, 00Ch
        db      0E4h, 00Ch
        db      014h, 00Ch              ; address= +0170h
        db      01Ah, 00Ch
        db      09Ah, 00Ch
        db      0D4h, 00Ch
        db      034h, 00Ch
        db      05Ah, 00Ch
        db      056h, 00Ch
        db      0E6h, 00Ch

;-------------- Makeup ------------------
        db      000h, 000h              ; address= +0180h / no data
        db      0F0h, 00Ah
        db      013h, 00Ch
        db      093h, 00Ch
        db      0DAh, 00Ch
        db      0CCh, 00Ch
        db      02Ch, 00Ch
        db      0ACh, 00Ch
        db      036h, 00Dh              ; address= +0190h
        db      0B6h, 00Dh
        db      052h, 00Dh
        db      0D2h, 00Dh
        db      032h, 00Dh
        db      0B2h, 00Dh
        db      04Eh, 00Dh
        db      0CEh, 00Dh
        db      02Eh, 00Dh              ; address= +01A0h
        db      0AEh, 00Dh
        db      06Eh, 00Dh
        db      0EEh, 00Dh
        db      04Ah, 00Dh
        db      0CAh, 00Dh
        db      02Ah, 00Dh
        db      0AAh, 00Dh
        db      05Ah, 00Dh              ; address= +01B0h
        db      0DAh, 00Dh
        db      026h, 00Dh
        db      0A6h, 00Dh
        db      010h, 00Bh
        db      030h, 00Bh
        db      0B0h, 00Bh
        db      048h, 00Ch
        db      0C8h, 00Ch              ; address= +01C0h
        db      028h, 00Ch
        db      0A8h, 00Ch
        db      068h, 00Ch
        db      0E8h, 00Ch
        db      038h, 00Ch
        db      0B8h, 00Ch
        db      078h, 00Ch
        db      0F8h, 00Ch              ; address= +01D0h

if(0)
code    ends
else
MHTBL_E endp
;-------------------------------------------------------
        ;@ENDCSEG
        @ENDMHTBL_E
;-------------------------------------------------------
endif
        end
```

4

## A.23　MHTBL_D.ASM

MHTBL_D.ASM

```
;*********************************************
;*      MH decode (expansion) table          *
;*         compression code -> data          *
;*=========================================*
;*             22-25561 : R.Yabui          *
;*                      1990.01.06         *
;*********************************************
;
        INCLUDE SEGMENT.MAC
;
;       * EOL
;          Number of '0' = 11
;
;       * EOL + '1'
;        -----4bit---+-1-+-------------7bit---------------
;        +---+---+---+---+---+---+---+---+---+---+---+---+
;        |num of'0'+1|col|           seach data           | ---+
;        +---+---+---+---+---+---+---+---+---+---+---+---+    |
;            num of '0'+1: number of '0'+1                    |
;            col: color                                       |
;            seach data: 7 bits                               |
;            ex) 0001 1011xxxx                                |
;                num of '0' = 3                               |
;                seach data = 1011xxx                         |
;                                                             |
;        --1-+-1-+---------6bit-----------                    |
;        +---+---+---+---+---+---+---+---+                    |
;        |col|t/m|       run length      | <------------------+
;        +---+---+---+---+---+---+---+---+ ------------+
;            col: color                                  |
;            t/m: 0 = terminate code                     |
;                 1 = makeup code                        |
;            run length: 6 bits                          |
;                                                        |
;        ------4bit------+-----4bit-------               |
;        +---+---+---+---+---+---+---+---+               |
;        | 0   0   0   0 |sch data length| <-------------+
;        +---+---+---+---+---+---+---+---+
;         data length
;            terminate:  number of '0'+1 + seach data length
;            makeup    : (number of '0'+1 + seach data length)*64
;
;+++++++++++++ Decode Table ++++++++++++++
if(0)
dedt    segment at      0000h                   ;       offset
else
;-------------------------------------------------------
        ;@CSEG
        @SEGMHTBL_D
;-------------------------------------------------------
        public  MHTBL_D
MHTBL_D proc    far
        ORG     0000h
endif
;------ White Terminal Data Table -------
        db      05h,02h,02h,03h,03h,03h,03h,03h   ; W.T=000h-007h
        db      04h,04h,02h,03h,03h,01h,05h,05h   ; W.T=008h-00Fh
        db      05h,05h,05h,03h,03h,04h,01h,02h   ; W.T=010h-017h
        db      05h,05h,04h,05h,04h,01h,01h,04h   ; W.T=018h-01Fh
        db      04h,04h,04h,04h,04h,04h,04h,05h   ; W.T=020h-
        db      05h,05h,05h,05h,05h,02h,02h,03h
        db      03h,06h,06h,06h,06h,05h,05h,06h   ; W.T=030h-
        db      06h,06h,06h,06h,06h,05h,05h,05h
;------- White Makeup Data Table --------
        db      00h,04h,04h,04h,05h,05h,05h,06h   ; W.M=040h-
        db      06h,06h,06h,07h,07h,07h,07h,07h
        db      07h,07h,07h,07h,07h,07h,07h,07h   ; W.M=050h-057h
        db      07h,07h,04h,07h                   ; W.M=058h-05Bh
;---- White/Black Addition Data Table ---
```

1

MHTBL_D.ASM

```
        db      03h,03h,03h,04h                 ; W/B.M=05Ch-05Fh
        db      04h,04h,04h,04h,04h,04h,04h,04h ; W/B.M=060h-067h
        db      04h,00h,00h,00h,00h,00h,00h,00h ; W/B.M=068h dummy=069h-
        db      00h,00h,00h,00h,00h,00h,00h,00h ; dummy
        db      00h,00h,00h,00h,00h,00h,00h,00h
;------ Black Terminate Data Table ------
        db      05h,01h,01h,01h,01h,01h,01h,01h ; B.T=080h-087h
        db      02h,02h,02h,02h,02h,02h,02h,04h ; B.T=088h-08Fh
        db      04h,04h,03h,06h,06h,06h,05h,05h ; B.T=090h-
        db      04h,04h,07h,07h,07h,07h,06h,06h
        db      06h,06h,07h,07h,07h,07h,07h,07h ; B.T=0A0h-
        db      06h,06h,07h,07h,06h,06h,06h,06h
        db      06h,06h,06h,06h,05h,05h,05h,05h ; B.T=0B0h-
        db      05h,06h,06h,05h,05h,06h,06h,06h
;------- Black Makeup Data Table --------
        db      00h,03h,07h,07h,06h,05h,05h,05h ; B.M=0C0h-
        db      06h,06h,06h,06h,06h,06h,06h,06h
        db      06h,06h,06h,06h,06h,06h,06h,06h ; B.M=0D0h-0D7h
        db      06h,06h,06h,06h                 ; B.M=0D8h-0DBh

        ORG     0100h
;+++++++++++++ Decode Table ++++++++++++
        db      003h,005h,009h,006h,042h,00Eh,004h,007h   ; address=0100h-
        db      003h,005h,010h,006h,008h,041h,004h,007h
        db      003h,005h,009h,006h,042h,00Fh,004h,007h   ; address=0110h-
        db      003h,005h,011h,006h,008h,041h,004h,007h
        db      003h,005h,009h,006h,042h,00Eh,004h,007h   ; address=0120h-
        db      003h,005h,010h,006h,008h,041h,004h,007h
        db      003h,005h,009h,006h,042h,00Fh,004h,007h   ; address=0130h-
        db      003h,005h,011h,006h,008h,041h,004h,007h
        db      003h,005h,009h,006h,042h,00Eh,004h,007h   ; address=0140h-
        db      003h,005h,010h,006h,008h,041h,004h,007h
        db      003h,005h,009h,006h,042h,00Fh,004h,007h   ; address=0150h-
        db      003h,005h,011h,006h,008h,041h,004h,007h
        db      003h,005h,009h,006h,042h,00Eh,004h,007h   ; address=0160h-
        db      003h,005h,010h,006h,008h,041h,004h,007h
        db      003h,005h,009h,006h,042h,00Fh,004h,007h   ; address=0170h-
        db      003h,005h,011h,006h,008h,041h,004h,007h
;
        db      083h,082h,083h,082h,083h,082h,083h,082h   ; address=0180h-
        db      083h,082h,083h,082h,083h,082h,083h,082h
        db      083h,082h,083h,082h,083h,082h,083h,082h   ; address=0190h-
        db      083h,082h,083h,082h,083h,082h,083h,082h
        db      083h,082h,083h,082h,083h,082h,083h,082h   ; address=01A0h-
        db      083h,082h,083h,082h,083h,082h,083h,082h
        db      083h,082h,083h,082h,083h,082h,083h,082h   ; address=01B0h-
        db      083h,082h,083h,082h,083h,082h,083h,082h
        db      083h,082h,083h,082h,083h,082h,083h,082h   ; address=01C0h-
        db      083h,082h,083h,082h,083h,082h,083h,082h
        db      083h,082h,083h,082h,083h,082h,083h,082h   ; address=01D0h-
        db      083h,082h,083h,082h,083h,082h,083h,082h
        db      083h,082h,083h,082h,083h,082h,083h,082h   ; address=01E0h-
        db      083h,082h,083h,082h,083h,082h,083h,082h
        db      083h,082h,083h,082h,083h,082h,083h,082h   ; address=01F0h-
        db      083h,082h,083h,082h,083h,082h,083h,082h

        ORG     0200h
;+++++++++++++ Decode Table ++++++++++++
        db      00Bh,05Ah,018h,002h,01Bh,049h,037h,002h   ; address=0200h-
        db      00Bh,047h,033h,002h,057h,053h,049h,002h
        db      00Bh,05Ah,031h,002h,03Bh,04Fh,039h,002h   ; address=0210h-
        db      00Bh,04Bh,019h,002h,012h,044h,043h,002h
        db      00Bh,05Ah,018h,002h,01Bh,04Dh,038h,002h   ; address=0220h-
        db      00Bh,048h,034h,002h,059h,055h,049h,002h
        db      00Bh,05Ah,032h,002h,03Ch,051h,03Ah,002h   ; address=0230h-
        db      00Bh,04Ah,019h,002h,012h,044h,048h,002h
        db      00Bh,05Ah,018h,002h,01Bh,049h,037h,002h   ; address=0240h-
        db      00Bh,047h,033h,002h,058h,054h,049h,002h
        db      00Bh,05Ah,031h,002h,03Bh,050h,039h,002h   ; address=0250h-
```

2

保守／廃止

MHTBL_D.ASM

```
        db      00Bh,04Ch,019h,002h,012h,044h,043h,002h
        db      00Bh,05Ah,018h,002h,01Bh,04Eh,038h,002h    ; address=0260h-
        db      00Bh,048h,034h,002h,05Bh,056h,043h,002h
        db      00Bh,05Ah,032h,002h,03Ch,052h,03Ah,002h    ; address=0270h-
        db      00Bh,04Ah,019h,002h,012h,044h,043h,002h
;
        db      081h,084h,081h,084h,081h,084h,081h,084h    ; address=0280h-
        db      081h,084h,081h,084h,081h,084h,081h,084h
        db      081h,084h,081h,084h,081h,084h,081h,084h    ; address=0290h-
        db      081h,084h,081h,084h,081h,084h,081h,084h
        db      081h,084h,081h,084h,081h,084h,081h,084h    ; address=02A0h-
        db      081h,084h,081h,084h,081h,084h,081h,084h
        db      081h,084h,081h,084h,081h,084h,081h,084h    ; address=02B0h-
        db      081h,084h,081h,084h,081h,084h,081h,084h
        db      081h,084h,081h,084h,081h,084h,081h,084h    ; address=02C0h-
        db      081h,084h,081h,084h,081h,084h,081h,084h
        db      081h,084h,081h,084h,081h,084h,081h,084h    ; address=02D0h-
        db      081h,084h,081h,084h,081h,084h,081h,084h
        db      081h,084h,081h,084h,081h,084h,081h,084h    ; address=02E0h-
        db      081h,084h,081h,084h,081h,084h,081h,084h
        db      081h,084h,081h,084h,081h,084h,081h,084h    ; address=02F0h-
        db      081h,084h,081h,084h,081h,084h,081h,084h

        ORG     0300h
;+++++++++++++ Decode Table +++++++++++++
        db      00Ch,01Ch,027h,00Ah,035h,03Fh,02Bh,00Ah    ; address=0300h-
        db      00Ch,03Dh,029h,00Ah,01Ah,045h,015h,00Ah
        db      00Ch,01Ch,028h,00Ah,036h,000h,02Ch,00Ah    ; address=0310h-
        db      00Ch,03Eh,02Ah,00Ah,01Ah,046h,015h,00Ah
        db      00Ch,01Ch,027h,00Ah,035h,03Fh,02Bh,00Ah    ; address=0320h-
        db      00Ch,03Dh,029h,00Ah,01Ah,045h,015h,00Ah
        db      00Ch,01Ch,028h,00Ah,036h,000h,02Ch,00Ah    ; address=0330h-
        db      00Ch,03Eh,02Ah,00Ah,01Ah,046h,015h,00Ah
        db      00Ch,01Ch,027h,00Ah,035h,03Fh,02Bh,00Ah    ; address=0340h-
        db      00Ch,03Dh,029h,00Ah,01Ah,045h,015h,00Ah
        db      00Ch,01Ch,028h,00Ah,036h,000h,02Ch,00Ah    ; address=0350h-
        db      00Ch,03Eh,02Ah,00Ah,01Ah,046h,015h,00Ah
        db      00Ch,01Ch,027h,00Ah,035h,03Fh,02Bh,00Ah    ; address=0360h-
        db      00Ch,03Dh,029h,00Ah,01Ah,045h,015h,00Ah
        db      00Ch,01Ch,028h,00Ah,036h,000h,02Ch,00Ah    ; address=0370h-
        db      00Ch,03Eh,02Ah,00Ah,01Ah,046h,015h,00Ah
;
        db      086h,085h,086h,085h,086h,085h,086h,085h    ; address=0380h-
        db      086h,085h,086h,085h,086h,085h,086h,085h
        db      086h,085h,086h,085h,086h,085h,086h,085h    ; address=0390h-
        db      086h,085h,086h,085h,086h,085h,086h,085h
        db      086h,085h,086h,085h,086h,085h,086h,085h    ; address=03A0h-
        db      086h,085h,086h,085h,086h,085h,086h,085h
        db      086h,085h,086h,085h,086h,085h,086h,085h    ; address=03B0h-
        db      086h,085h,086h,085h,086h,085h,086h,085h
        db      086h,085h,086h,085h,086h,085h,086h,085h    ; address=03C0h-
        db      086h,085h,086h,085h,086h,085h,086h,085h
        db      086h,085h,086h,085h,086h,085h,086h,085h    ; address=03D0h-
        db      086h,085h,086h,085h,086h,085h,086h,085h
        db      086h,085h,086h,085h,086h,085h,086h,085h    ; address=03E0h-
        db      086h,085h,086h,085h,086h,085h,086h,085h
        db      086h,085h,086h,085h,086h,085h,086h,085h    ; address=03F0h-
        db      086h,085h,086h,085h,086h,085h,086h,085h

        ORG     0400h
;+++++++++++++ Decode Table +++++++++++++
        db      014h,013h,023h,001h,021h,01Fh,025h,001h    ; address=0400h-
        db      014h,013h,024h,001h,022h,020h,026h,001h
        db      014h,013h,023h,001h,021h,01Fh,025h,001h    ; address=0410h-
        db      014h,013h,024h,001h,022h,020h,026h,001h
        db      014h,013h,023h,001h,021h,01Fh,025h,001h    ; address=0420h-
        db      014h,013h,024h,001h,022h,020h,026h,001h
        db      014h,013h,023h,001h,021h,01Fh,025h,001h    ; address=0430h-
        db      014h,013h,024h,001h,022h,020h,026h,001h
```

MHTBL_D.ASM

```
        db      014h,013h,023h,001h,021h,01Fh,025h,001h   ; address=0440h-
        db      014h,013h,024h,001h,022h,020h,026h,001h
        db      014h,013h,023h,001h,021h,01Fh,025h,001h   ; address=0450h-
        db      014h,013h,024h,001h,022h,020h,026h,001h
        db      014h,013h,023h,001h,021h,01Fh,025h,001h   ; address=0460h-
        db      014h,013h,024h,001h,022h,020h,026h,001h
        db      014h,013h,023h,001h,021h,01Fh,025h,001h   ; address=0470h-
        db      014h,013h,024h,001h,022h,020h,026h,001h
;
        db      089h,087h,088h,087h,089h,087h,088h,087h   ; address=0480h-
        db      089h,087h,088h,087h,089h,087h,088h,087h
        db      089h,087h,088h,087h,089h,087h,088h,087h   ; address=0490h-
        db      089h,087h,088h,087h,089h,087h,088h,087h
        db      089h,087h,088h,087h,089h,087h,088h,087h   ; address=04A0h-
        db      089h,087h,088h,087h,089h,087h,088h,087h
        db      089h,087h,088h,087h,089h,087h,088h,087h   ; address=04B0h-
        db      089h,087h,088h,087h,089h,087h,088h,087h
        db      089h,087h,088h,087h,089h,087h,088h,087h   ; address=04C0h-
        db      089h,087h,088h,087h,089h,087h,088h,087h
        db      089h,087h,088h,087h,089h,087h,088h,087h   ; address=04D0h-
        db      089h,087h,088h,087h,089h,087h,088h,087h
        db      089h,087h,088h,087h,089h,087h,088h,087h   ; address=04E0h-
        db      089h,087h,088h,087h,089h,087h,088h,087h
        db      089h,087h,088h,087h,089h,087h,088h,087h   ; address=04F0h-
        db      089h,087h,088h,087h,089h,087h,088h,087h

        ORG     0500h
;+++++++++++++ Decode Table +++++++++++++
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh   ; address=0500h-
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh   ; address=0510h-
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh   ; address=0520h-
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh   ; address=0530h-
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh   ; address=0540h-
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh   ; address=0550h-
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh   ; address=0560h-
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh   ; address=0570h-
        db      017h,00Dh,02Fh,00Dh,017h,00Dh,030h,00Dh
;
        db      08Ah,08Fh,08Bh,08Ch,08Ah,094h,08Bh,08Ch   ; address=0580h-
        db      08Ah,0C2h,08Bh,08Ch,08Ah,095h,08Bh,08Ch
        db      08Ah,08Fh,08Bh,08Ch,08Ah,0A4h,08Bh,08Ch   ; address=0590h-
        db      08Ah,09Ch,08Bh,08Ch,08Ah,080h,08Bh,08Ch
        db      08Ah,08Fh,08Bh,08Ch,08Ah,0A2h,08Bh,08Ch   ; address=05A0h-
        db      08Ah,09Ah,08Bh,08Ch,08Ah,0AAh,08Bh,08Ch
        db      08Ah,08Fh,08Bh,08Ch,08Ah,0A6h,08Bh,08Ch   ; address=05B0h-
        db      08Ah,093h,08Bh,08Ch,08Ah,080h,08Bh,08Ch
        db      08Ah,08Fh,08Bh,08Ch,08Ah,094h,08Bh,08Ch   ; address=05C0h-
        db      08Ah,0C3h,08Bh,08Ch,08Ah,095h,08Bh,08Ch
        db      08Ah,08Fh,08Bh,08Ch,08Ah,0A5h,08Bh,08Ch   ; address=05D0h-
        db      08Ah,09Dh,08Bh,08Ch,08Ah,080h,08Bh,08Ch
        db      08Ah,08Fh,08Bh,08Ch,08Ah,0A3h,08Bh,08Ch   ; address=05E0h-
        db      08Ah,09Bh,08Bh,08Ch,08Ah,0ABh,08Bh,08Ch
        db      08Ah,08Fh,08Bh,08Ch,08Ah,0A7h,08Bh,08Ch   ; address=05F0h-
        db      08Ah,093h,08Bh,08Ch,08Ah,080h,08Bh,08Ch

        ORG     0600h
;+++++++++++++ Decode Table +++++++++++++
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h   ; address=0600h-
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h   ; address=0610h-
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h   ; address=0620h-
```

MHTBL_D.ASM

```
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h   ; address=062Eh-
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h   ; address=0640h-
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h   ; address=0650h-
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h   ; address=0660h-
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h   ; address=0670h-
        db      02Dh,016h,02Eh,016h,02Dh,016h,02Eh,016h
;
        db      08Dh,091h,097h,08Eh,08Dh,09Eh,0B9h,08Eh   ; address=0680h-
        db      08Dh,0B0h,0ACh,08Eh,08Dh,0A8h,090h,08Eh
        db      08Dh,091h,0B2h,08Eh,08Dh,0A0h,0BDh,08Eh   ; address=0690h-
        db      08Dh,0BEh,0AEh,08Eh,08Dh,096h,090h,08Eh
        db      08Dh,091h,097h,08Eh,08Dh,09Fh,0BAh,08Eh   ; address=06A0h-
        db      08Dh,0B1h,0ADh,08Eh,08Dh,0A9h,090h,08Eh
        db      08Dh,091h,0B3h,08Eh,08Dh,0A1h,0C4h,08Eh   ; address=06B0h-
        db      08Dh,0BFh,0AFh,08Eh,08Dh,096h,090h,08Eh
        db      08Dh,091h,097h,08Eh,08Dh,09Eh,0B9h,08Eh   ; address=06C0h-
        db      08Dh,0B0h,0ACh,08Eh,08Dh,0A8h,090h,08Eh
        db      08Dh,091h,0B2h,08Eh,08Dh,0A0h,0BDh,08Eh   ; address=06D0h-
        db      08Dh,0BEh,0AEh,08Eh,08Dh,096h,090h,08Eh
        db      08Dh,091h,097h,08Eh,08Dh,09Fh,0BAh,08Eh   ; address=06E0h-
        db      08Dh,0B1h,0ADh,08Eh,08Dh,0A9h,090h,08Eh
        db      08Dh,091h,0B3h,08Eh,08Dh,0A1h,0C4h,08Eh   ; address=06F0h-
        db      08Dh,0BFh,0AFh,08Eh,08Dh,096h,090h,08Eh

        ORG     0700h
;+++++++++++++ Decode Table +++++++++++++
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh   ; address=0700h-
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh   ; address=0710h-
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh   ; address=0720h-
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh   ; address=072Eh-
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh   ; address=0740h-
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh   ; address=0750h-
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh   ; address=0760h-
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh   ; address=0770h-
        db      01Dh,01Eh,01Dh,01Eh,01Dh,01Eh,01Dh,01Eh
;
        db      092h,099h,0B8h,0B6h,0B4h,0C6h,0BCh,0C1h   ; address=0780h-
        db      092h,0DAh,0D6h,0D0h,0CCh,0C8h,098h,0C1h
        db      092h,099h,0D4h,0CEh,0CAh,0C7h,0D8h,0C1h   ; address=0790h-
        db      092h,0C5h,0BBh,0D2h,0B7h,0B5h,098h,0C1h
        db      092h,099h,0B8h,0B6h,0B4h,0C6h,0BCh,0C1h   ; address=07A0h-
        db      092h,0DBh,0D7h,0D1h,0CDh,0C9h,098h,0C1h
        db      092h,099h,0D5h,0CFh,0CBh,0C7h,0D9h,0C1h   ; address=07B0h-
        db      092h,0C5h,0BBh,0D3h,0B7h,0B5h,098h,0C1h
        db      092h,099h,0B8h,0B6h,0B4h,0C6h,0BCh,0C1h   ; address=07C0h-
        db      092h,0DAh,0D6h,0D0h,0CCh,0C8h,098h,0C1h
        db      092h,099h,0D4h,0CEh,0CAh,0C7h,0D8h,0C1h   ; address=07D0h-
        db      092h,0C5h,0BBh,0D2h,0B7h,0B5h,098h,0C1h
        db      092h,099h,0B8h,0B6h,0B4h,0C6h,0BCh,0C1h   ; address=07E0h-
        db      092h,0DBh,0D7h,0D1h,0CDh,0C9h,098h,0C1h
        db      092h,099h,0D5h,0CFh,0CBh,0C7h,0D9h,0C1h   ; address=07F0h-
        db      092h,0C5h,0BBh,0D3h,0B7h,0B5h,098h,0C1h

        ORG     0800h
;+++++++++++++ Decode Table +++++++++++++
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=0800h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
```

5

MHTBL_D.ASM

```
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=0810h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=0820h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=0830h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=0840h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=0850h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=0860h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=0870h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
;
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=0880h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=0890h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=08A0h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=08B0h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=08C0h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=08D0h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=08E0h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
        db      05Ch,05Dh,061h,065h,05Fh,05Eh,063h,067h   ; address=08F0h-
        db      05Ch,05Dh,062h,066h,060h,05Eh,064h,068h
if(0)
dedt    ends
else
MHTBL_D endp
;-------------------------------------------------------
        ;@ENDCSEG
        @ENDMHTBL_D
;-------------------------------------------------------
endif
        end
```

6

## A.24　ADECOMR.ASM

ADECOMR.ASM

```
        public  _adecode_DECMR
        public  _adecode_DECMR_00,                      _adecode_DECMR_02
        public  _adecode_DECMR_03,_adecode_DECMR_13
        public  _adecode_DECMR_F0
        public   adecode_DECMR_03_A, adecode_DECMR_03_B, adecode_DECMR_03_C
        public   adecode_DECMR_03_D, adecode_DECMR_03_E, adecode_DECMR_03_F
        public   adecode_DECMR_13_A, adecode_DECMR_13_B, adecode_DECMR_13_C
        public   adecode_DECMR_13_D, adecode_DECMR_13_E, adecode_DECMR_13_F
;
        @PUSHR
;=====================================;
;       Entry                         ;
;=====================================;
;
;=====================================;
;             Search EOL              ;
;=====================================;
_adecode_DECMR_00:      ;ＥＯＬ探し
        cmp     _expn_work.sequ,02h     ;? sequ-2
        jc      @F
        jmp     _adecode_DECMR_02
    @@:
;☆
        cmp     _expn_param.clen_rest,00
        jnz     @F
;       les     ax,_expn_param.cptr     ;符号データセグ&オフセ
;       mov     bx,es
;       MOV_IRAM_BX     REGB_10+REGB_DS ;セグ
;       MOV_IRAM_AX     REGB_10+REGB_SS ;オフセ
;       mov     ax,_expn_param.clen     ;コード長
;       MOV_IRAM_AX     REGB_10+REGB_PS
        mov     bx,_expn_param.clen     ;コード長，継ぎ足し方式
        MOV_AX_IRAM     REGB_10+REGB_PS
        add     ax,bx
        MOV_IRAM_AX     REGB_10+REGB_PS
    @@:

        cmp     _expn_work.sequ,00h     ;? sequ-0
        jnz     @F
        MOV_IRAM_IMMW   REGB_10+REGB_AX,0       ; ０－クリア
        mov     _expn_work.sequ,01h     ;Next seq
    @@:

        call    BANK_probe
        call    Bank_save

        mov     al,0ah                          ; bank number is 10
        mov     ah,00h
        V55_SCHEOL      ;【ＳＣＨＥＯＬ】
        nop

        call    BANK_probe

        call    get_NZ  ;bx <- NZflag
        call    get_CY  ;cx <- CYflag
        mov     _expn_param.clen_rest,bx        ;clen_rest <- 残り有無

        jc      @F
        ;CY=0
        MOV_AX_IRAM     REGB_10+REGB_PS
        mov     _expn_param.clen,ax
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,02h     ;OK, Next seq
        cmp     _expn_param.clen_rest,0 ;NZ-flag
        jnz     _adecode_DECMR_02       ;in direct
        jmp     _adecode_DECMR_END
    @@:
        ;CY=1
        cmp     ah,03h                  ;任意のデータ＋ＥＯＬ
```

1

ADECOMR.ASM

```
        jz      adecode_DECMR_00_B
        cmp     ah,00h                  ;中断；全て００ｈ
        jz      adecode_DECMR_00_A
        jmp     adecode_DECMR_00_C
adecode_DECMR_00_A:
        call    Bank_load
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,01h     ;NG, Again
        jmp     _adecode_DECMR_END
adecode_DECMR_00_B:
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,02h     ;NG but OK, Next seq
        cmp     _expn_param.clen_rest,0 ;NZ-flag
        jnz     _adecode_DECMR_02       ;in direct
        jmp     _adecode_DECMR_END
adecode_DECMR_00_C:
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,01h     ;OK
        jmp     _adecode_DECMR_END

;=====================================;
;           Get TAG (1bit)            ;
;=====================================;
_adecode_DECMR_02:      ;付加ビットの検出
        cmp     _expn_work.sequ,03h     ;? sequ-3
        jnc     _adecode_DECMR_03
;☆
        cmp     _expn_param.clen_rest,00
        jnz     @F
;       les     ax,_expn_param.cptr     ;符号データセグ&オフセ
;       mov     bx,es
;       MOV_IRAM_BX     REGB_10+REGB_DS ;セグ
;       MOV_IRAM_AX     REGB_10+REGB_SS ;オフセ
;       mov     ax,_expn_param.clen     ;コード長
;       MOV_IRAM_AX     REGB_10+REGB_PS
        mov     bx,_expn_param.clen     ;コード長，継ぎ足し方式
        MOV_AX_IRAM     REGB_10+REGB_PS
        add     ax,bx
        MOV_IRAM_AX     REGB_10+REGB_PS
    @@:

        call    BANK_probe

        mov     al,0ah                          ; bank number is 10
        mov     ah,00h
        V55_GETBIT      ;【ＧＥＴＢＩＴ】
        nop

        call    BANK_probe

        call    get_NZ  ;bx <- NZflag
        call    get_CY  ;cx <- CYflag
        mov     _expn_param.clen_rest,bx        ;clen_rest <- 残り有無

        jc      @F
        ;CY=0   ;２次元ライン
        add     _expn_work.K,1
        sbb     _expn_work.K,0
        cmp     _expn_work.K,0FFh
        jz      adecode_DECMR_K_err
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,13h
        jmp     _adecode_DECMR_END
    @@:
        ;CY=1   ;１次元ライン
        mov     _expn_work.K,0
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,03h     ;Next seq
        jmp     _adecode_DECMR_END
```

2

保守／廃止

ADECOMR.ASM

```
 adecode_DECMR_K_err:
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,00h     ;Next seq
        jmp     _adecode_DECMR_END

;=====================================;
;         MH decode (1 line)          ;
;=====================================;
_adecode_DECMR_03:      ;データ復号化
        cmp     _expn_work.sequ,05h     ;? sequ-5
        jc      @F
        jmp     _adecode_DECMR_13
    @@:
;☆
        cmp     _expn_param.clen_rest,00
        jnz     @F
        mov     bx,_expn_param.clen     ;コード長，継ぎ足し方式
        MOV_AX_IRAM     REGB_10+REGB_PS
        add     ax,bx
        MOV_IRAM_AX     REGB_10+REGB_PS
    @@:

        cmp     _expn_work.sequ,03h     ;? sequ-3
        jnz     @F
        les     di,_trnp_tabl_A         ; turning point table
        mov     ax,es
        MOV_IRAM_AX     REGB_10+REGB_ES ; turning point table segment
        mov     _expn_work.TRNP_SEG,ax  ;/*当該変化点テーブルセグ*/
        mov     ax,di
        MOV_IRAM_AX     REGB_10+REGB_DI turning point table offset
        mov     _expn_work.TRNP_OFS,ax  ;/*当該変化点テーブルオフ*/
        MOV_IRAM_IMMW   REGB_10+REGB_SI,1728    ;１７２８ｂｉｔ
        $1      equ     seg MHTBL_D             ; decode table
        MOV_IRAM_IMMW   REGB_10+REGB_BP,$1
        MOV_IRAM_IMMW   REGB_10+REGB_CX,0       ; ０－クリア
        MOV_IRAM_IMMW   REGB_10+REGB_AX,0
        mov     _expn_work.sequ,04h     ;Next seq
    @@:

        call    BANK_probe
        call    Bank_save

        mov     al,0ah                          ; bank number is 10
        mov     ah,00h
        V55_MHDEC       ;【ＭＨＤＥＣ】
        nop

        call    BANK_probe

        call    get_NZ  ;bx <- NZflag
        call    get_CY  ;cx <- CYflag
        mov     _expn_param.clen_rest,bx        ;clen_rest <- 残り有無

        jc      @F
        ;CY=0
        mov     _expn_work.sequ,00h     ;Next seq
        jmp     _adecode_DECMR_F0       ;ｇｏｔｏ　"ＰＲＩＮＴ"
    @@:
        ;CY=1
        cmp     ah,05h
        jz      adecode_DECMR_03_F
        cmp     ah,04h
        jz      adecode_DECMR_03_E
        cmp     ah,03h
        jz      adecode_DECMR_03_D
        cmp     ah,02h
        jz      adecode_DECMR_03_C
        cmp     ah,01h
        jz      adecode_DECMR_03_B
```

$

ADECOMR.ASM

```
        cmp     ah,00h
        jz      adecode_DECMR_03_A
        jmp     adecode_DECMR_03_F
adecode_DECMR_03_A:                     ;中断
        call    Bank_load
         MOV_AX_IRAM    REGB_10+REGB_PS
         cmp    ax,2000h                ;xx-2000h
         jnc    adecode_DECMR_03_F
        mov     _expn_param.stat,0      ;データの補充が必要
        jmp     _adecode_DECMR_END
adecode_DECMR_03_B:                     ;ＥＯＬ
        mov     _expn_param.stat,20h    ;ＲＴＣ
        mov     _expn_param.RTCFLG,1    ;ＲＴＣ検出
        jmp     _adecode_DECMR_END
adecode_DECMR_03_C:                     ;ＦＩＬＬ＋ＥＯＬ
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,00h     ;Next seq
        jmp     _adecode_DECMR_END
adecode_DECMR_03_D:                     ;満たないデータ＋ＥＯＬ
        jmp     adecode_DECMR_03_F
adecode_DECMR_03_E:                     ;1走査線の画素数超過
        jmp     adecode_DECMR_03_F
adecode_DECMR_03_F:
        MOV_AX_IRAM     REGB_10+REGB_PS
        mov     _expn_param.clen,ax
        mov     _expn_param.stat,03h    ;エラー
        mov     _expn_work.sequ,00h     ;Again
        jmp     _adecode_DECMR_END

;=======================================;
;         MR decode (1 line)            ;
;=======================================;
_adecode_DECMR_13:      ;データ復号化
        cmp     _expn_work.sequ,15h     ;? sequ-15h
        jc      @F
        jmp     _adecode_DECMR_F0
    @@:
;☆
        cmp     _expn_param.clen_rest,00
        jnz     @F
        mov     bx,_expn_param.clen
        MOV_AX_IRAM     REGB_10+REGB_PS
        add     ax,bx
        MOV_IRAM_AX     REGB_10+REGB_PS
    @@:

        cmp     _expn_work.sequ,13h     ;? sequ-13
        jnz     @F
        MOV_DX_IRAM     REGB_10+REGB_DI ; memo turning point table
        les     di,_trnp_tabl_A         ; turning point table
        mov     ax,es
        MOV_IRAM_AX     REGB_10+REGB_ES ; turning point table segment
        mov     _expn_work.TRNP_SEG,ax  ;/*当該変化点テーブルセグ**/
        and     dx,6000h        ;ｷｲ
        MOV_IRAM_DX     REGB_10+REGB_SI ; turning point table, reference line
        xor     dx,2000h        ;ﾃﾞ
        MOV_IRAM_DX     REGB_10+REGB_DI ; turning point table, coding line
        mov     _expn_work.TRNP_OFS,dx  ;/*当該変化点テーブルオフ**/
        $1      equ     seg MHTBL_D             ; decode table
        MOV_IRAM_IMMW   REGB_10+REGB_BP,$1
        MOV_IRAM_IMMW   REGB_10+REGB_CX,0       ; 0ークリア
        MOV_IRAM_IMMW   REGB_10+REGB_AX,0
        mov     _expn_work.sequ,14h     ;Next seq
    @@:

        call    BANK_probe
        call    Bank_save

        mov     al,0ah                          ; bank number is 10
```

保守／廃止

ADECOMR.ASM

```
        mov     ah,00h
        V55_MRDEC       ;【ＭＲＤＥＣ】
        nop

        call    BANK_probe

        call    get_NZ  ;bx <- NZflag
        call    get_CY  ;cx <- CYflag
        mov     _expn_param.clen_rest,bx        ;clen_rest <- 残り有無

        jc      @F
        ;CY=0
        mov     _expn_work.sequ,00h
        jmp     _adecode_DECMR_F0
    @@:
        ;CY=1
        cmp     ah,05h
        jz      adecode_DECMR_13_F
        cmp     ah,04h
        jz      adecode_DECMR_13_E
        cmp     ah,03h
        jz      adecode_DECMR_13_D
        cmp     ah,02h
        jz      adecode_DECMR_13_C
        cmp     ah,01h
        jz      adecode_DECMR_13_B
        cmp     ah,00h
        jz      adecode_DECMR_13_A
        jmp     adecode_DECMR_13_F
adecode_DECMR_13_A:                     ;中断
        call    Bank_load
         MOV_AX_IRAM    REGB_10+REGB_PS ;コード長が２０００ｈならばエラー
         cmp    ax,2000h                ;xx-2000h
         jnc    adecode_DECMR_13_F
        mov     _expn_param.stat,0      ;データの補充が必要
        jmp     _adecode_DECMR_END
adecode_DECMR_13_B:                     ;ＥＯＬ
        mov     _expn_param.stat,20h    ;ＲＴＣ
        mov     _expn_param.RTCFLG,1    ;ＲＴＣ検出
        jmp     _adecode_DECMR_END
adecode_DECMR_13_C:                     ;ＦＩＬＬ＋ＥＯＬ
        mov     _expn_param.stat,0
        mov     _expn_work.sequ,00h     ;Next seq
        jmp     _adecode_DECMR_END
adecode_DECMR_13_D:                     ;満たないデータ＋ＥＯＬ
        jmp     adecode_DECMR_13_F
adecode_DECMR_13_E:                     ;１走査線の画素数超過
        jmp     adecode_DECMR_13_F
adecode_DECMR_13_F:                     ;異常なコード
        MOV_AX_IRAM     REGB_10+REGB_PS
        mov     _expn_param.clen,ax
        mov     _expn_param.stat,03h    ;エラー
        mov     _expn_work.sequ,00h     ;Again
        jmp     _adecode_DECMR_END

;=======================================;
; Convert turning point into image data ;
;=======================================;
_adecode_DECMR_F0:      ;画像生データ再生
        mov     ax,_expn_work.TRNP_SEG  ;/*当該変化点テーブルセグ**/
        MOV_IRAM_AX     REGB_11+REGB_DS
        mov     ax,_expn_work.TRNP_OFS  ;/*当該変化点テーブルオフ**/
        MOV_IRAM_AX     REGB_11+REGB_SS
        les     di,_expn_param.gptr             ; ラインバッファ
        mov     ax,es
        MOV_IRAM_AX     REGB_11+REGB_ES
        mov     ax,di
        MOV_IRAM_AX     REGB_11+REGB_DI
        MOV_IRAM_IMMW   REGB_11+REGB_CX,0       ; ０－クリア
```

5

ADECOMR.ASM

```
        MOV_IRAM_IMMW   REGB_11+REGB_AX,0

        mov     al,0bh                          ; bank number is 11
        mov     ah,00h
        V55_CNVTRP      ;【C N V T R P】
        nop

        mov     _expn_param.stat,10h
        mov     _expn_work.sequ,00h     ;Next seq
        jmp     _adecode_DECMR_END
_adecode_DECMR_END:
;=======================================;
;       Exit                            ;
;=======================================;
        @POPR
        ret
```

## A.25　V25.MAC

V25.MAC

```
        .XLIST
;*************************************************************
;*      ＃３２８　アセンブラ　インクルードファイル          *
;*              For MSC ver 5.1 Medium Model                 *
;*              FILE NAME 'V25.MAC'                          *
;*************************************************************
;
        .186    ;１８０１８６の命令には対応しておく
;
;*****************************************************************************
;       ８０１６８及びＶ２５のマクロ等
;*****************************************************************************
@PUSHR  MACRO
        pusha
        push    ds              ;DSを退避
        push    es              ;ESを退避
        ENDM

@POPR   MACRO
        pop     es              ;ESを復帰
        pop     ds              ;DSを復帰
        popa
        ENDM

;/////////////////////////////////////////
;       プッシュ＆ポップＳＩ，ＤＩ定義
;/////////////////////////////////////////
@PUSH_I MACRO
;
        PUSH    SI      ;レジスタ変数退避
        PUSH    DI      ;
;
                ENDM

@POP_I  MACRO
;
        POP     DI      ;レジスタ変数復帰
        POP     SI
;
                ENDM

;/////////////////////////////////////////

;/////////////////////////////////////////
;       プッシュ＆ポップＥＳ－ＢＸ定義
;/////////////////////////////////////////
push_esbx       MACRO
;
        PUSH    BX      ;レジスタ変数退避
        PUSH    ES      ;
;
                ENDM

pop_esbx        MACRO
;
        POP     ES      ;レジスタ変数復帰
        POP     BX
;
                ENDM

;/////////////////////////////////////////

;Ｖ２５　ｏｎｌｙ
@STOP   MACRO
        DB      0FH,9EH         ;V25 STOP MODE
        ENDM

@FINT   MACRO
        DB      0FH,92H         ;FINISHED INTERRUPT
```

1

保守／廃止

V25.MAC

```
        ENDM
;
retrbi  MACRO                   ;オペランド有り
        db      0fh             ;指定された特殊機能レジのビットが1のとき,
        db      91h
        ENDM
;
fint    MACRO                   ;オペランド無し
        db      0fh
        db      92h
        ENDM
;
stop    MACRO                   ;オペランド無し
        db      0fh
        db      9eh
        ENDM
;
brkcs_ax        MACRO           ;オペランド有り
        db      0fh
        db      2dh
        db      0c0h            ; 11000 ax
        ENDM
;
brkcs_cx        MACRO           ;オペランド有り
        db      0fh
        db      2dh
        db      0c1h            ; 11000 cx
        ENDM
;
brkcs_dx        MACRO           ;オペランド有り
        db      0fh
        db      2dh
        db      0c2h            ; 11000 dx
        ENDM
;
brkcs_bx        MACRO           ;オペランド有り
        db      0fh
        db      2dh
        db      0c3h            ; 11000 bx
        ENDM
;
;
page
;
brkcs_sp        MACRO           ;オペランド有り
        db      0fh
        db      2dh
        db      0c4h            ; 11000 sp
        ENDM
;
brkcs_bp        MACRO           ;オペランド有り
        db      0fh
        db      2dh
        db      0c5h            ; 11000 bp
        ENDM
;
brkcs_ix        MACRO           ;オペランド有り
        db      0fh
        db      2dh
        db      0c6h            ; 11000 ix
        ENDM
;
brkcs_iy        MACRO           ;オペランド有り
        db      0fh
        db      2dh
        db      0c7h            ; 11000 iy
        ENDM
;
tsksw_ax        MACRO           ;オペランド有り
```

2

V25.MAC

```
        db      0fh
        db      94h
        db      0f8h            ; 11111 ax
        ENDM
;
tsksw_cx        MACRO           ;オペランド有り
        db      0fh
        db      94h
        db      0f9h            ; 11111 cx
        ENDM
;
tsksw_dx        MACRO           ;オペランド有り
        db      0fh
        db      94h
        db      0fah            ; 11111 dx
        ENDM
;
tsksw_bx        MACRO           ;オペランド有り
        db      0fh
        db      94h
        db      0fbh            ; 11111 bx
        ENDM
;
;
page
;
tsksw_sp        MACRO           ;オペランド有り
        db      0fh
        db      94h
        db      0fch            ; 11111 sp
        ENDM
;
tsksw_bp        MACRO           ;オペランド有り
        db      0fh
        db      94h
        db      0fdh            ; 11111 bp
        ENDM
;
tsksw_ix        MACRO           ;オペランド有り
        db      0fh
        db      94h
        db      0feh            ; 11111 ix
        ENDM
;
tsksw_iy        MACRO           ;オペランド有り
        db      0fh
        db      94h
        db      0ffh            ; 11111 iy
        ENDM
;
movspa  MACRO                   ;オペランド無し
        db      0fh
        db      25h
        ENDM
;
movspb_ax       MACRO           ;オペランド有り
        db      0fh
        db      95h
        db      0f8h
        ENDM
;
movspb_cx       MACRO           ;オペランド有り
        db      0fh
        db      95h
        db      0f9h
        ENDM
;
movspb_dx       MACRO           ;オペランド有り
        db      0fh
```

3

V25.MAC

```
        db      95h
        db      0fah
        ENDM
;
movspb_bx       MACRO           ;オペランド有り
        db      0fh
        db      95h
        db      0fbh
        ENDM
;
;
page
;
movspb_sp       MACRO           ;オペランド有り
        db      0fh
        db      95h
        db      0fch
        ENDM
;
movspb_bp       MACRO           ;オペランド有り
        db      0fh
        db      95h
        db      0fdh
        ENDM
;
movspb_ix       MACRO           ;オペランド有り
        db      0fh
        db      95h
        db      0feh
        ENDM
;
movspb_iy       MACRO           ;オペランド有り
        db      0fh
        db      95h
        db      0ffh
        ENDM
;
;
        .LIST
        .LALL
```

4

## A.26　V55PI.MAC

V55PI.MAC

```
        .XLIST
;*************************************************************
;*************************************************************
;
        .186    ;１８０１８６の命令には対応しておく
;
;       Ｖ５５ＰＩのマクロ
;
;変化点テーブル作成命令
V55_COLTRP      MACRO
        DB      0Fh,9Bh
        ENDM

;データ送出命令
V55_ALBIT       MACRO
        DB      0Fh,9Ah
        ENDM

;ＭＨ符号化命令
V55_MHENC       MACRO
        DB      0Fh,93h
        ENDM

;ＭＲ符号化命令
V55_MRENC       MACRO
        DB      0Fh,97h
        ENDM

;ＥＯＬ検出命令
V55_SCHEOL      MACRO
        DB      0Fh,78h
        ENDM

;１ビット（タグ）検出命令
V55_GETBIT      MACRO
        DB      0Fh,79h
        ENDM

;ＭＨ復号化命令
V55_MHDEC       MACRO
        DB      0Fh,7Ch
        ENDM

;ＭＲ復号化命令
V55_MRDEC       MACRO
        DB      0Fh,7Dh
        ENDM

;画素データ作成命令
V55_CNVTRP      MACRO
        DB      0Fh,7Ah
        ENDM
```

1

保守／廃止

## A.27　V55PI_2.MAC

V55PI_2.MAC

```
;/************************************************************************/
;/*                                                                      */
;/*            Ｖ５５ＰＩレジスタファイル空間アクセス用マクロの定義     */
;/*                                                ver. 1.00             */
;/************************************************************************/

;アクセス方向はレジスタあるいは直接からレジスタファイルへ

;       MOV     IRAM:[ADDR],immw の代わり
MOV_IRAM_IMMW   MACRO   ADDR,IMMW
        DB      0F1h            ;(IRAM prefix)
        DB      0C7h            ;MOV
        DB       06h            ;
        DW      ADDR and 1FFh   ;IRAM:[ADDR],
        DW      IMMW            ;immw
        ENDM

;       MOV     IRAM:[ADDR],AX の代わり
MOV_IRAM_AX     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB      0A3h            ;MOV
        DW      ADDR and 1FFh   ;IRAM:[ADDR],AX
        ENDM

;       MOV     IRAM:[ADDR],BX の代わり
MOV_IRAM_BX     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       89h            ;MOV
        DB       1Eh
        DW      ADDR and 1FFh   ;IRAM:[ADDR],BX
        ENDM

;       MOV     IRAM:[ADDR],CX の代わり
MOV_IRAM_CX     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       89h            ;MOV
        DB       0Eh
        DW      ADDR and 1FFh   ;IRAM:[ADDR],CX
        ENDM

;       MOV     IRAM:[ADDR],DX の代わり
MOV_IRAM_DX     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       89h            ;MOV
        DB       16h
        DW      ADDR and 1FFh   ;IRAM:[ADDR],DX
        ENDM

if(1)
MOV_IRAM_IMMB   MACRO   ADDR,IMMB
        DB      0F1h            ;(IRAM prefix)
        DB      0C6h            ;MOV
        DB       06h            ;
        DW      ADDR and 1FFh   ;IRAM:[ADDR],
        DB      IMMB            ;immw
        ENDM

;       MOV     IRAM:[ADDR],AL の代わり
MOV_IRAM_AL     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB      0A2h
        DW      ADDR and 1FFh   ;IRAM:[ADDR],AL
        ENDM

;       MOV     IRAM:[ADDR],AH の代わり
MOV_IRAM_AH     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       88h            ;MOV
        DB       26h
```

1

V55PI_2.MAC

```
        DW      ADDR and 1FFh   ;IRAM:[ADDR],AH
        ENDM

;       MOV     IRAM:[ADDR],BL の代わり
MOV_IRAM_BL     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       88h            ;MOV
        DB       1Eh
        DW      ADDR and 1FFh   ;IRAM:[ADDR],BL
        ENDM

;       MOV     IRAM:[ADDR],BH の代わり
MOV_IRAM_BH     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       88h            ;MOV
        DB       3Eh
        DW      ADDR and 1FFh   ;IRAM:[ADDR],BH
        ENDM

;       MOV     IRAM:[ADDR],CL の代わり
MOV_IRAM_CL     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       88h            ;MOV
        DB       0Eh
        DW      ADDR and 1FFh   ;IRAM:[ADDR],CL
        ENDM

;       MOV     IRAM:[ADDR],CH の代わり
MOV_IRAM_CH     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       88h            ;MOV
        DB       2Eh
        DW      ADDR and 1FFh   ;IRAM:[ADDR],CH
        ENDM

;       MOV     IRAM:[ADDR],DL の代わり
MOV_IRAM_DL     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       88h            ;MOV
        DB       16h
        DW      ADDR and 1FFh   ;IRAM:[ADDR],DL
        ENDM

;       MOV     IRAM:[ADDR],DH の代わり
MOV_IRAM_DH     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       88h            ;MOV
        DB       36h
        DW      ADDR and 1FFh   ;IRAM:[ADDR],DH
        ENDM
endif

;アクセス方向はレジスタファイルからレジスタ（１６ビット）へ

;       MOV     AX,IRAM:[ADDR] の代わり
MOV_AX_IRAM     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB      0A1h            ;MOV
        DW      ADDR and 1FFh   ;AX,IRAM:[ADDR]
        ENDM

;       MOV     BX,IRAM:[ADDR] の代わり
MOV_BX_IRAM     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       8Bh            ;MOV
        DB       1Eh
        DW      ADDR and 1FFh   ;BX,IRAM:[ADDR]
        ENDM
```

2

V55P1_2.MAC

```
;       MOV     CX,IRAM:[ADDR] の代わり
MOV_CX_IRAM     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       8Bh            ;MOV
        DB       0Eh
        DW      ADDR and 1FFh   ;CX,IRAM:[ADDR]
        ENDM

;       MOV     DX,IRAM:[ADDR] の代わり
MOV_DX_IRAM     MACRO   ADDR
        DB      0F1h            ;(IRAM prefix)
        DB       8Bh            ;MOV
        DB       16h
        DW      ADDR and 1FFh   ;DX,IRAM:[ADDR]
        ENDM
;
RSTWDT  MACRO
        DB      0FH
        DB      96H
        DB      082H
        DB      07DH
        ENDM
;
TSKSW_AX MACRO
        DB      0FH
        DB      94H
        DB      0F8H
        ENDM
;
MOVSPB_AX MACRO
        DB      0FH
        DB      95H
        DB      0F8H
        ENDM
```

3

保守／廃止

## A.28　V55PI_3.MAC

V55PI_3.MAC

```
;/*************************************************************************/
;/*                                                                       */
;/*            Ｖ５５ＰＩレジスタファイル空間ポインタの定義               */
;/*                                  １９９１年６月６日                   */
;/*                                        ver. 1.00                      */
;/*************************************************************************/

;/*レジスタバンクのポインタの定義**/
;使用法
;       MOV_IRAM_IMMW   REGB_00+REGB_AX,0
;                               ;レジスタバンク０のＡＸに００００hを入れます.
;       MOV_IRAM_CX     REGB_05+REGB_CX,CX
;                               ;レジスタバンク５のＣＸに現ＣＸを入れます.

;各レジスタバンクの開始アドレス
REGB_00         equ     000h
REGB_01         equ     020h
REGB_02         equ     040h
REGB_03         equ     060h
REGB_04         equ     080h
REGB_05         equ     0A0h
REGB_06         equ     0C0h
REGB_07         equ     0E0h
REGB_08         equ     100h
REGB_09         equ     120h
REGB_10         equ     140h
REGB_11         equ     160h
REGB_12         equ     180h
REGB_13         equ     1A0h
REGB_14         equ     1C0h
REGB_15         equ     1E0h

;各レジスタバンクの開始アドレスからのオフセット
REGB_DS2        equ     00h     ;
REGB_vect_PC    equ     02h     ;
 REGB_DS3       equ     02h     ;vect_PC & DS3 have the same relative pointers.
REGB_PSW_esc    equ     04h     ;
REGB_PC_esc     equ     06h     ;
REGB_DS         equ     08h     ;
REGB_SS         equ     0Ah     ;
REGB_PS         equ     0Ch     ;
REGB_ES         equ     0Eh     ;
REGB_DI         equ     10h     ;
REGB_SI         equ     12h     ;
REGB_BP         equ     14h     ;
REGB_SP         equ     16h     ;
REGB_BX         equ     18h     ;
REGB_DX         equ     1Ah     ;
REGB_CX         equ     1Ch     ;
REGB_AX         equ     1Eh     ;



;/*Ｖ５５マクロサービスチャネルのポインタの定義**/
;使用法
;       MOV_IRAM_IMMW   MS_INTP0,MS_EVTCNT_INC
;       MOV_IRAM_IMMW   MS_INTP0+1,5h
;                               ;ＩＮＴＰ０端子入力をイベントカウントで，５分周に設定します
;
;       MOV_IRAM_IMMW   MS_INTCM01,MS_RTOPTRVN_U0A0
;       MOV_IRAM_IMMW   MS_INTCM01+1,MS_MSC
;       MOV_IRAM_IMMW   MS_MSC,X0
;       MOV_IRAM_IMMW   MS_MSC,X1
;                               ;タイマ０１のタイミングで，リアルタイム出力を
;                               ;マクロサービスでします.
;

 MS_work        equ     000h;   /* マクロ・サービス・ワーク・エリア **/
MS_INTPA1       equ     008h;   /* マクロ・サービス・コントロール・ワード・エリア **/
```

1

V55PI_3.MAC

```
MS_INTAD        equ     00Ah;
 MS_reserve_0C  equ     00Ch;
MS_INTP0        equ     012h;
MS_INTP1        equ     014h;
MS_INTP2        equ     016h;
MS_INTP3        equ     018h;
MS_INTP4        equ     01Ah;
MS_INTP5        equ     01Ch;
 MS_reserve_1E  equ     01Eh;
MS_INTCM00      equ     020h;
MS_INTCM01      equ     022h;   ＣＭ０１の一致検出
MS_INTCM10      equ     024h;
MS_INTCM11      equ     026h;
MS_INTCM21      equ     028h;
MS_INTCM31      equ     02Ah;
MS_INTD0        equ     02Ch;
MS_INTD0S       equ     02Eh;
MS_INTD1        equ     030h;
MS_INTD1S       equ     032h;
MS_INTSER0      equ     034h;
MS_INTSER1      equ     036h;
MS_INTSR0       equ     038h;
MS_INTSR1       equ     03Ah;
MS_INTST0       equ     03Ch;
MS_INTST1       equ     03Eh;
 MS_MSC         equ     040h;   /* マクロ・サービス・チャネル・エリア **/

MS_EVTCNT_INC   equ     00h;    イベントカウント
MS_EVTCNT_DEC   equ     02h;
MS_DTACMP       equ     08h;    データ比較，ただしバイト
MS_DTADIF_W     equ     12h;    差分の計算
MS_DTADIF_B     equ     10h;
MS_BLKTRS_M0D0W equ     40h;    ＳＦＲとのブロック転送：ＭＥＭ←ＳＦＲ，基本メモリ空間，ワー
ド指定
MS_BLKTRS_M0D0B equ     42h;                            ＭＥＭ←ＳＦＲ，基本メモリ空間，バイ
ト指定
MS_BLKTRS_M0D1W equ     48h;                            ＳＦＲ←ＭＥＭ，基本メモリ空間，ワー
ド指定
MS_BLKTRS_M0D1B equ     4Ah;                            ＳＦＲ←ＭＥＭ，基本メモリ空間，バイ
ト指定
MS_BLKTRS_M1D0W equ     50h;                            ＭＥＭ←ＳＦＲ，拡張メモリ空間，ワー
ド指定
MS_BLKTRS_M1D0B equ     52h;                            ＭＥＭ←ＳＦＲ，拡張メモリ空間，バイ
ト指定
MS_BLKTRS_M1D1W equ     58h;                            ＳＦＲ←ＭＥＭ，拡張メモリ空間，ワー
ド指定
MS_BLKTRS_M1D1B equ     5Ah;                            ＳＦＲ←ＭＥＭ，拡張メモリ空間，バイ
ト指定
MS_BLKTRSC_M0D0W        equ     60h;    ＳＦＲとのブロック転送，ただし一致判定付き
MS_BLKTRSC_M0D0B        equ     62h;
MS_BLKTRSC_M0D1W        equ     68h;
MS_BLKTRSC_M0D1B        equ     6Ah;
MS_BLKTRSC_M1D0W        equ     70h;
MS_BLKTRSC_M1D0B        equ     72h;
MS_BLKTRSC_M1D1W        equ     78h;
MS_BLKTRSC_M1D1B        equ     7Ah;
MS_RTOPTRVN_U0A0        equ     80h;    リアルタイム出力ポート
MS_RTOPTRVN_U0A1        equ     82h;
MS_RTOPTRVN_U1A0        equ     84h;
MS_RTOPTRVN_U1A1        equ     86h;
```

2

## A.29　MC329.MAC

MC329.MAC

```
;/*************************************************************************/
;/*************************************************************************/
;
ON      EQU     0FFh
OFF     EQU      00h

;/*ＬＥＤ' ｓ**/
COVEL   EQU      80h            ;/*カバー　ＬＥＤ**/
PAPEL   EQU      40h            ;/*ペーパーＬＥＤ**/
ERROL   EQU      20h            ;/*エラー　ＬＥＤ**/
FINEL   EQU      10h            ;/*ファインＬＥＤ**/
STARL   EQU      08h            ;/*スタートＬＥＤ**/
POWERL  EQU      04h            ;/*ＰＯＷＥＲ　ＬＥＤ**/

;ブザ
BUZZ    EQU     04h                     ;ブザービット，ｏｎ　ＰＬＡ

;入力
FINES   EQU     40h                     ;ファインＳＷビット
STOPS   EQU     20h                     ;ストップＳＷビット
STARS   EQU     10h                     ;スタートＳＷビット
COPYS   EQU     08h                     ;コピーＳＷビット
GENCS   EQU     04h                     ;原稿 sw
PAPES   EQU     02h                     ;記録紙 sw
COVES   EQU     01h                     ;蓋 sw


V55REG_ST       STRUC
        ADCR0   DB      ?       ;/* Ａ／Ｄコンバージョン・リザルト・レジ０～３ **/
                reserve_E01     DB      1 DUP (?)
        ADCR1   DB      ?
                reserve_E03     DB      1 DUP (?)
        ADCR2   DB      ?
                reserve_E05     DB      1 DUP (?)
        ADCR3   DB      ?
                reserve_E07     DB      9 DUP (?)
;
        PAB     DB      ?       ;/* パラレル・Ｉ／Ｆ・バッファ **/
                reserve_E11     DB      7 DUP (?)
        PAC0    DB      ?       ;/* パラレル・Ｉ／Ｆ・コントロール・レジ０～１ **/
        PAC1    DB      ?       ;
        PAS     DB      ?       ;/* パラレル・Ｉ／Ｆ・ステータス・レジ **/
                reserve_E1B     DB      1 DUP (?)
        PAI1    DB      ?       ;/* パラレル・Ｉ／Ｆ・アクノリッジ・インタバル・レジ１～２ *
*/
        PAI2    DB      ?       ;
                reserve_E1E     DB      2 DUP (?)

        ADM     DB      ?       ;/* Ａ／Ｄコンバータ・モード・レジ **/
                reserve_E21     DB      9Fh DUP (?)

        MK0     DW      ?       ;/* 割り込みマスク・フラグ・レジ０～１ **/
        MK1     DW      ?       ;
        ISPR    DB      ?       ;/* インサービス・プライオリティ・レジ **/
        IMC     DB      ?       ;/* 割り込みモード・コントロール・レジ **/
                reserve_EC6     DB      3 DUP (?)

        IC09    DB      ?       ;/* 割り込み要求制御レジ０９～３７ **/
        IC10    DB      ?       ;
        IC11    DB      ?       ;
        IC12    DB      ?       ;
        IC13    DB      ?       ;
        IC14    DB      ?       ;
                reserve_ECF     DB      1 DUP (?)
        IC16    DB      ?       ;
        IC17    DB      ?       ;
        IC18    DB      ?       ;
        IC19    DB      ?       ;
        IC20    DB      ?       ;
```

1

保守／廃止

MC329.MAC

```
IC21    DB      ?       ;
IC22    DB      ?       ;
IC23    DB      ?       ;
IC24    DB      ?       ;
IC25    DB      ?       ;
IC26    DB      ?       ;
IC27    DB      ?       ;
IC28    DB      ?       ;
IC29    DB      ?       ;
IC30    DB      ?       ;
IC31    DB      ?       ;
IC32    DB      ?       ;
        reserve_EE1     DB      3 DUP (?)
IC36    DB      ?       ;
IC37    DB      ?       ;
        reserve_EE6     DB      1Ah DUP (?)

P0      DB      ?       ;/* ポート0～8 **/
P1      DB      ?       ;
P2      DB      ?       ;
P3      DB      ?       ;
P4      DB      ?       ;
P5      DB      ?       ;
P6      DB      ?       ;
P7      DB      ?       ;
P8      DB      ?       ;
        reserve_F09     DB      3h DUP (?);
PRDC    DB      ?       ;/* ポート・リード・コントロール・レジ **/
        reserve_F0D     DB      1h DUP (?);
RTP     DB      ?       ;/* リアルタイム出力ポート **/
        reserve_F0F     DB      1h DUP (?);
PM0     DB      ?       ;/* ポート0～8モード・レジ **/
        reserve_F11     DB      1h DUP (?);
PM2     DB      ?       ;
PM3     DB      ?       ;
PM4     DB      ?       ;
PM5     DB      ?       ;
        reserve_F16     DB      1h DUP (?);
PM7     DB      ?       ;
PM8     DB      ?       ;
        reserve_F19     DB      9h DUP (?);
PMC2    DB      ?       ;/* ポート2～8モード・コントロール・レジ **/
PMC3    DB      ?       ;
PMC4    DB      ?       ;
PMC5    DB      ?       ;
        reserve_F26     DB      1h DUP (?);
PMC7    DB      ?       ;
PMC8    DB      ?       ;
        reserve_F29     DB      3h DUP (?);
RTPC    DB      ?       ;/* リアルタイム出力ポート・コントロール・レジ **/
RTPD    DB      ?       ;/* リアルタイム出力ポート・ディレイ指定レジ **/
P7L     DB      ?       ;/* ポート7バッファ，下位 **/
P7H     DB      ?       ;/* ポート7バッファ，上位 **/

TMC0    DB      ?       ;/* タイマ・コントロール・レジ0～1 **/
TMC1    DB      ?       ;
TOC0    DB      ?       ;/* タイマ出力制御レジ0～1 **/
TOC1    DB      ?       ;
INTM0   DB      ?       ;/* 外部割り込みモード・レジ0～1 **/
INTM1   DB      ?       ;
        reserve_F36     DB      0Ah DUP (?);

TM0     DW      ?       ;/* タイマ・レジ0～3 **/
TM1     DW      ?       ;
TM2     DW      ?       ;
TM3     DW      ?       ;
CT00    DW      ?       ;/* タイマ・キャプチャ・レジ00～01 **/
CT01    DW      ?       ;
CM00    DW      ?       ;/* タイマ・コンペア・レジ00～01 **/
```

2

保守／廃止

MC329.MAC

```
        CM01    DW      ?       ;
        CT10    DW      ?       ;/* タイマ・キャプチャ・レジ１０ **/
        CM10    DW      ?       ;/* タイマ・コンペア・レジ１０～１１ **/
        CM11    DW      ?       ;
                reserve_F56     DB      2h DUP (?);
        CM20    DW      ?       ;/* タイマ・コンペア・レジ２０～２３ **/
        CM21    DW      ?       ;
        CM22    DW      ?       ;
        CM23    DW      ?       ;
        WDM     DW      ?       ;/* ヲッチドッグ・タイマ・モード・レジ **/
                reserve_F62     DB      2h DUP (?);
        CM30    DW      ?       ;/* タイマ・コンペア・レジ３０～３１ **/
        CM31    DW      ?       ;
                reserve_F68     DB      4h DUP (?);

        PWM     DB      ?       ;/* ＰＷＭレジ **/
        PWMC    DB      ?       ;/* ＰＷＭコントロール・レジ **/
                reserve_F6E     DB      2h DUP (?);

        TxBRG0  DB      ?       ;/* 送信ビットレート・ジェネレータ・レジ０ **/
        RxBRG0  DB      ?       ;/* 受信ビットレート・ジェネレータ・レジ０ **/
        PRS0    DB      ?       ;/* プリスケーラ・レジ０ **/
        UARTM0_CSIM0    DB      ?       ;/* ＵＡＲＴモード・レジ０／クロックト・シリアル・Ｉ
／Ｆ・モード・レジ０ **/
        UARTS0_SBIC0    DB      ?       ;/* ＵＡＲＴステータス・レジ０／ＳＢＩコントロール・
レジ０ **/
        TxB0_SIO0       DB      ?       ;/* 送信バッファ０ **/
        RxB0    DB      ?       ;/* 受信バッファ０ **/
                reserve_F77     DB      1h DUP (?);
        TxBRG1  DB      ?       ;/* 送信ビットレート・ジェネレータ・レジ１ **/
        RxBRG1  DB      ?       ;/* 受信ビットレート・ジェネレータ・レジ１ **/
        PRS1    DB      ?       ;/* プリスケーラ・レジ１ **/
        UARTM1_CSIM1    DB      ?       ;/* ＵＡＲＴモード・レジ１／クロックト・シリアル・Ｉ
／Ｆ・モード・レジ１ **/
        U1RTS1  DB      ?       ;/* ＵＡＲＴステータス・レジ１ **/
        TxB1_SIO1       DB      ?       ;/* 送信バッファ１ **/
        RxB1    DB      ?       ;/* 受信バッファ１ **/
        ASP     DB      ?       ;/* プロトコル選択レジ **/

        TC0     DD      ?       ;/* ターミナル・カウンタ０ **/
        TCM0    DD      ?       ;/* ターミナル・カウンタ・モジュロ・レジ０ **/
        UDC0    DD      ?       ;/* ＤＭＡアップ・ダウン・カウンタ０ **/
        DCM0    DD      ?       ;/* ＤＭＡコンペア・レジ０ **/
        MAR0    DD      ?       ;/* ＤＭＡメモリアドレス・レジ０ **/
        DPTC0   DD      ?       ;/* ＤＭＡリードライト・ポインタ０ **/
                reserve_F98     DB      4h DUP (?);
        DMAM0   DB      ?       ;/* ＤＭＡモード・レジ０ **/
        DMAC0   DB      ?       ;/* ＤＭＡコントロール・レジ０ **/
        DMAS    DB      ?       ;/* ＤＭＡステータス・レジ０ **/
                reserve_F9F     DB      1h DUP (?);
        TC1     DD      ?       ;/* ターミナル・カウンタ１ **/
        TCM1    DD      ?       ;/* ターミナル・カウンタ・モジュロ・レジ１ **/
        UDC1    DD      ?       ;/* ＤＭＡアップ・ダウン・カウンタ１ **/
        DCM1    DD      ?       ;/* ＤＭＡコンペア・レジ１ **/
        MAR1    DD      ?       ;/* ＤＭＡメモリアドレス・レジ１ **/
        DPTC1   DD      ?       ;/* ＤＭＡリードライト・ポインタ１ **/
                reserve_FB8     DB      4h DUP (?);
        DMAM1   DB      ?       ;/* ＤＭＡモード・レジ１ **/
        DMAC1   DB      ?       ;/* ＤＭＡコントロール・レジ１ **/
                reserve_FBE     DB      22h DUP (?);

        STC_st  DW      ?       ;/* ソフトウェア・タイマ・カウンタ **/
        STMC    DW      ?       ;/* ソフトウェア・タイマ・カウンタ・コンペア・レジ **/
                reserve_FE4     DB      4h DUP (?);

        PWC0    DB      ?       ;/* プログラマブル・ウェイト・コントロール・レジ０～１ **/
        PWC1    DB      ?       ;

        MBC     DB      ?       ;/* メモリ・ブロック・コントロール・レジ **/
```

保守／廃止

MCS29.MAC

```
                reserve_FEB     DB      1h DUP (?);
        RFM     DB      ?       ;/* リフレッシュ・モード・レジ **/
                reserve_FED     DB      1h DUP (?);
        STBC    DB      ?       ;/* スタンバイ・コントロール・レジ **/
        PRC     DB      ?       ;/* プロセッサ・コントロール・レジ **/
                reserve_FF0     DB      10h DUP (?);
;
V55REG_ST       ENDS
```

4

## A.30　SEGMENT.MAC

SEGMENT.MAC

```
        .XLIST
;*************************************************************
;*      ＃３２８　アセンブラ　インクルードファイル          *
;*              For MSC ver 5.1 Medium Model                 *
;*              FILE NAME 'SEGMENT.MAC'                      *
;*************************************************************

;/////////////////////////////////////////
;       セグメント配列定義
;/////////////////////////////////////////
@DEFSEG         MACRO
;
ASM_TEXT        SEGMENT  PAGE PUBLIC 'CODE'
ASM_TEXT        ENDS
;
_DATA   SEGMENT  para PUBLIC 'DATA'
_DATA   ENDS
CONST   SEGMENT  WORD PUBLIC 'CONST'
CONST   ENDS
_BSS    SEGMENT  para PUBLIC 'BSS'
_BSS    ENDS
;
;FAR_DATA segment para public 'FAR_DATA'

DGROUP  GROUP   CONST, _BSS, _DATA
        ASSUME  CS: ASM_TEXT, DS: DGROUP, SS: DGROUP
;
                ENDM

;/////////////////////////////////////////
;       コードセグメント開始と終了定義
;/////////////////////////////////////////
@CSEG           MACRO
;
ASM_TEXT        SEGMENT  PAGE PUBLIC 'CODE'
        ASSUME  CS: ASM_TEXT
;
                ENDM

@ENDCSEG        MACRO
;
ASM_TEXT        ENDS
;
                ENDM

;/////////////////////////////////////////
;       データセグメント開始，終了定義
;       （初期化されたデータ領域）
;/////////////////////////////////////////
@DSEG           MACRO
;
_DATA       SEGMENT
;       ASSUME  DS: DGROUP
;
                ENDM

@ENDDSEG        MACRO
;
_DATA   ENDS
;
                ENDM

;/////////////////////////////////////////
;       ＢＳＳセグメント開始，終了定義
;       （初期化されないデータ領域）
;/////////////////////////////////////////
@BSEG           MACRO
;
_BSS        SEGMENT
```

1

SEGMENT.MAC

```
;       ASSUME  DS: DGROUP
;
                ENDM

@ENDBSEG        MACRO
;
_BSS    ENDS
;
                ENDM

;////////////////////////////////////////////
;       ＭＨＴＢＬ＿Ｅセグメント開始と終了定義
;////////////////////////////////////////////
@SEGMHTBL_E     MACRO
;
;MHTBL_E_TEXT   SEGMENT  PARA PUBLIC 'CODE'      ;パラグラフ境界指定
;       ASSUME  CS: MHTBL_E_TEXT
MHTBL_E_TEXT    SEGMENT  PAGE PUBLIC 'CODE'      ;ページ境界指定
        ASSUME  CS: MHTBL_E_TEXT
;
                ENDM

@ENDMHTBL_E     MACRO
;
MHTBL_E_TEXT    ENDS
;
                ENDM

;////////////////////////////////////////////
;       ＭＨＴＢＬ＿Ｄセグメント開始と終了定義
;////////////////////////////////////////////
@SEGMHTBL_D     MACRO
;
;MHTBL_D_TEXT   SEGMENT  PARA PUBLIC 'CODE'      ;パラグラフ境界指定
;       ASSUME  CS: MHTBL_D_TEXT
MHTBL_D_TEXT    SEGMENT  PAGE PUBLIC 'CODE'      ;ページ境界指定
        ASSUME  CS: MHTBL_D_TEXT
;
                ENDM

@ENDMHTBL_D     MACRO
;
MHTBL_D_TEXT    ENDS
;
                ENDM

;////////////////////////////////////////////////////////////////////////

;////////////////////////////////////
;       アセンブラ開始，終了定義
;////////////////////////////////////
@ENTASM         MACRO
;
        PUSH    BP
        MOV     BP,SP
;
                ENDM

@RETASM         MACRO
;
        MOV     SP,BP
        POP     BP
        RET
;
                ENDM

        .LIST
        .ALL
```

2

## A.31　BASICDEF.H

BASICDEF.H

```
/*****************************************************************************/
/*****************************************************************************/

/*      変数タイプ，函数タイプ  **/
typedef unsigned char  byte ;
typedef unsigned short word ;
typedef unsigned long  dwrd ;

typedef unsigned char  uchar ;
typedef unsigned short ushor ;
typedef unsigned short ushort;
typedef unsigned long  ulong ;
typedef unsigned int   uint ;

typedef         char  ibyte ;
typedef         short iword ;
typedef         long  idwrd ;

void    _enable();
void    _disable();
int     inp(uint);
int     outp(uint,int);
#pragma intrinsic( _enable, _disable )
#pragma intrinsic( inp, outp )

/*      ２進数字，００ｈ～ＦＦｈ**/
#define B00000000       0x00
#define B00000001       0x01
#define B00000010       0x02
#define B00000011       0x03
#define B00000100       0x04
#define B00000101       0x05
#define B00000110       0x06
#define B00000111       0x07
#define B00001000       0x08
#define B00001001       0x09
#define B00001010       0x0a
#define B00001011       0x0b
#define B00001100       0x0c
#define B00001101       0x0d
#define B00001110       0x0e
#define B00001111       0x0f

#define B00010000       0x10
#define B00010001       0x11
#define B00010010       0x12
#define B00010011       0x13
#define B00010100       0x14
#define B00010101       0x15
#define B00010110       0x16
#define B00010111       0x17
#define B00011000       0x18
#define B00011001       0x19
#define B00011010       0x1a
#define B00011011       0x1b
#define B00011100       0x1c
#define B00011101       0x1d
#define B00011110       0x1e
#define B00011111       0x1f

#define B00100000       0x20
#define B00100001       0x21
#define B00100010       0x22
#define B00100011       0x23
#define B00100100       0x24
#define B00100101       0x25
#define B00100110       0x26
#define B00100111       0x27
#define B00101000       0x28
```

1

BASICDEF.H

```
#define B00101001       0x29
#define B00101010       0x2a
#define B00101011       0x2b
#define B00101100       0x2c
#define B00101101       0x2d
#define B00101110       0x2e
#define B00101111       0x2f

#define B00110000       0x30
#define B00110001       0x31
#define B00110010       0x32
#define B00110011       0x33
#define B00110100       0x34
#define B00110101       0x35
#define B00110110       0x36
#define B00110111       0x37
#define B00111000       0x38
#define B00111001       0x39
#define B00111010       0x3a
#define B00111011       0x3b
#define B00111100       0x3c
#define B00111101       0x3d
#define B00111110       0x3e
#define B00111111       0x3f

#define B01000000       0x40
#define B01000001       0x41
#define B01000010       0x42
#define B01000011       0x43
#define B01000100       0x44
#define B01000101       0x45
#define B01000110       0x46
#define B01000111       0x47
#define B01001000       0x48
#define B01001001       0x49
#define B01001010       0x4a
#define B01001011       0x4b
#define B01001100       0x4c
#define B01001101       0x4d
#define B01001110       0x4e
#define B01001111       0x4f

#define B01010000       0x50
#define B01010001       0x51
#define B01010010       0x52
#define B01010011       0x53
#define B01010100       0x54
#define B01010101       0x55
#define B01010110       0x56
#define B01010111       0x57
#define B01011000       0x58
#define B01011001       0x59
#define B01011010       0x5a
#define B01011011       0x5b
#define B01011100       0x5c
#define B01011101       0x5d
#define B01011110       0x5e
#define B01011111       0x5f

#define B01100000       0x60
#define B01100001       0x61
#define B01100010       0x62
#define B01100011       0x63
#define B01100100       0x64
#define B01100101       0x65
#define B01100110       0x66
#define B01100111       0x67
#define B01101000       0x68
#define B01101001       0x69
```

2

保守／廃止

BASICDEF.H

```
#define B01101010       0x6a
#define B01101011       0x6b
#define B01101100       0x6c
#define B01101101       0x6d
#define B01101110       0x6e
#define B01101111       0x6f

#define B01110000       0x70
#define B01110001       0x71
#define B01110010       0x72
#define B01110011       0x73
#define B01110100       0x74
#define B01110101       0x75
#define B01110110       0x76
#define B01110111       0x77
#define B01111000       0x78
#define B01111001       0x79
#define B01111010       0x7a
#define B01111011       0x7b
#define B01111100       0x7c
#define B01111101       0x7d
#define B01111110       0x7e
#define B01111111       0x7f

#define B10000000       0x80
#define B10000001       0x81
#define B10000010       0x82
#define B10000011       0x83
#define B10000100       0x84
#define B10000101       0x85
#define B10000110       0x86
#define B10000111       0x87
#define B10001000       0x88
#define B10001001       0x89
#define B10001010       0x8a
#define B10001011       0x8b
#define B10001100       0x8c
#define B10001101       0x8d
#define B10001110       0x8e
#define B10001111       0x8f

#define B10010000       0x90
#define B10010001       0x91
#define B10010010       0x92
#define B10010011       0x93
#define B10010100       0x94
#define B10010101       0x95
#define B10010110       0x96
#define B10010111       0x97
#define B10011000       0x98
#define B10011001       0x99
#define B10011010       0x9a
#define B10011011       0x9b
#define B10011100       0x9c
#define B10011101       0x9d
#define B10011110       0x9e
#define B10011111       0x9f

#define B10100000       0xa0
#define B10100001       0xa1
#define B10100010       0xa2
#define B10100011       0xa3
#define B10100100       0xa4
#define B10100101       0xa5
#define B10100110       0xa6
#define B10100111       0xa7
#define B10101000       0xa8
#define B10101001       0xa9
#define B10101010       0xaa
```

3

保守／廃止

BASICDEF.H

```
#define B10101011       0xab
#define B10101100       0xac
#define B10101101       0xad
#define B10101110       0xae
#define B10101111       0xaf

#define B10110000       0xb0
#define B10110001       0xb1
#define B10110010       0xb2
#define B10110011       0xb3
#define B10110100       0xb4
#define B10110101       0xb5
#define B10110110       0xb6
#define B10110111       0xb7
#define B10111000       0xb8
#define B10111001       0xb9
#define B10111010       0xba
#define B10111011       0xbb
#define B10111100       0xbc
#define B10111101       0xbd
#define B10111110       0xbe
#define B10111111       0xbf

#define B11000000       0xc0
#define B11000001       0xc1
#define B11000010       0xc2
#define B11000011       0xc3
#define B11000100       0xc4
#define B11000101       0xc5
#define B11000110       0xc6
#define B11000111       0xc7
#define B11001000       0xc8
#define B11001001       0xc9
#define B11001010       0xca
#define B11001011       0xcb
#define B11001100       0xcc
#define B11001101       0xcd
#define B11001110       0xce
#define B11001111       0xcf

#define B11010000       0xd0
#define B11010001       0xd1
#define B11010010       0xd2
#define B11010011       0xd3
#define B11010100       0xd4
#define B11010101       0xd5
#define B11010110       0xd6
#define B11010111       0xd7
#define B11011000       0xd8
#define B11011001       0xd9
#define B11011010       0xda
#define B11011011       0xdb
#define B11011100       0xdc
#define B11011101       0xdd
#define B11011110       0xde
#define B11011111       0xdf

#define B11100000       0xe0
#define B11100001       0xe1
#define B11100010       0xe2
#define B11100011       0xe3
#define B11100100       0xe4
#define B11100101       0xe5
#define B11100110       0xe6
#define B11100111       0xe7
#define B11101000       0xe8
#define B11101001       0xe9
#define B11101010       0xea
#define B11101011       0xeb
```

4

BASICDEF.H

```
#define B11101100       0xec
#define B11101101       0xed
#define B11101110       0xee
#define B11101111       0xef

#define B11110000       0xf0
#define B11110001       0xf1
#define B11110010       0xf2
#define B11110011       0xf3
#define B11110100       0xf4
#define B11110101       0xf5
#define B11110110       0xf6
#define B11110111       0xf7
#define B11111000       0xf8
#define B11111001       0xf9
#define B11111010       0xfa
#define B11111011       0xfb
#define B11111100       0xfc
#define B11111101       0xfd
#define B11111110       0xfe
#define B11111111       0xff
```

5

## A.32　CONSTBIN.H

CONSTBIN.H

```
/*************************************************************************/
/*                                                                       */
/*      Ｃコンパイラ用２進数定義（ｕｃｈａｒ）のヘッダファイル          */
/*                                                                       */
/*************************************************************************/

/*      ２進数字  */
#define B00000000       0x00
#define B00000001       0x01
#define B00000010       0x02
#define B00000011       0x03
#define B00000100       0x04
#define B00000101       0x05
#define B00000110       0x06
#define B00000111       0x07
#define B00001000       0x08
#define B00001001       0x09
#define B00001010       0x0a
#define B00001011       0x0b
#define B00001100       0x0c
#define B00001101       0x0d
#define B00001110       0x0e
#define B00001111       0x0f

#define B00010000       0x10
#define B00010001       0x11
#define B00010010       0x12
#define B00010011       0x13
#define B00010100       0x14
#define B00010101       0x15
#define B00010110       0x16
#define B00010111       0x17
#define B00011000       0x18
#define B00011001       0x19
#define B00011010       0x1a
#define B00011011       0x1b
#define B00011100       0x1c
#define B00011101       0x1d
#define B00011110       0x1e
#define B00011111       0x1f

#define B00100000       0x20
#define B00100001       0x21
#define B00100010       0x22
#define B00100011       0x23
#define B00100100       0x24
#define B00100101       0x25
#define B00100110       0x26
#define B00100111       0x27
#define B00101000       0x28
#define B00101001       0x29
#define B00101010       0x2a
#define B00101011       0x2b
#define B00101100       0x2c
#define B00101101       0x2d
#define B00101110       0x2e
#define B00101111       0x2f

#define B00110000       0x30
#define B00110001       0x31
#define B00110010       0x32
#define B00110011       0x33
#define B00110100       0x34
#define B00110101       0x35
#define B00110110       0x36
#define B00110111       0x37
#define B00111000       0x38
#define B00111001       0x39
#define B00111010       0x3a
```

1

CONSTBIN.H

```
#define B00111011       0x3b
#define B00111100       0x3c
#define B00111101       0x3d
#define B00111110       0x3e
#define B00111111       0x3f

#define B01000000       0x40
#define B01000001       0x41
#define B01000010       0x42
#define B01000011       0x43
#define B01000100       0x44
#define B01000101       0x45
#define B01000110       0x46
#define B01000111       0x47
#define B01001000       0x48
#define B01001001       0x49
#define B01001010       0x4a
#define B01001011       0x4b
#define B01001100       0x4c
#define B01001101       0x4d
#define B01001110       0x4e
#define B01001111       0x4f

#define B01010000       0x50
#define B01010001       0x51
#define B01010010       0x52
#define B01010011       0x53
#define B01010100       0x54
#define B01010101       0x55
#define B01010110       0x56
#define B01010111       0x57
#define B01011000       0x58
#define B01011001       0x59
#define B01011010       0x5a
#define B01011011       0x5b
#define B01011100       0x5c
#define B01011101       0x5d
#define B01011110       0x5e
#define B01011111       0x5f

#define B01100000       0x60
#define B01100001       0x61
#define B01100010       0x62
#define B01100011       0x63
#define B01100100       0x64
#define B01100101       0x65
#define B01100110       0x66
#define B01100111       0x67
#define B01101000       0x68
#define B01101001       0x69
#define B01101010       0x6a
#define B01101011       0x6b
#define B01101100       0x6c
#define B01101101       0x6d
#define B01101110       0x6e
#define B01101111       0x6f

#define B01110000       0x70
#define B01110001       0x71
#define B01110010       0x72
#define B01110011       0x73
#define B01110100       0x74
#define B01110101       0x75
#define B01110110       0x76
#define B01110111       0x77
#define B01111000       0x78
#define B01111001       0x79
#define B01111010       0x7a
#define B01111011       0x7b
```

2

保守／廃止

CONSTBIN.H

```
#define B01111100       0x7c
#define B01111101       0x7d
#define B01111110       0x7e
#define B01111111       0x7f

#define B10000000       0x80
#define B10000001       0x81
#define B10000010       0x82
#define B10000011       0x83
#define B10000100       0x84
#define B10000101       0x85
#define B10000110       0x86
#define B10000111       0x87
#define B10001000       0x88
#define B10001001       0x89
#define B10001010       0x8a
#define B10001011       0x8b
#define B10001100       0x8c
#define B10001101       0x8d
#define B10001110       0x8e
#define B10001111       0x8f

#define B10010000       0x90
#define B10010001       0x91
#define B10010010       0x92
#define B10010011       0x93
#define B10010100       0x94
#define B10010101       0x95
#define B10010110       0x96
#define B10010111       0x97
#define B10011000       0x98
#define B10011001       0x99
#define B10011010       0x9a
#define B10011011       0x9b
#define B10011100       0x9c
#define B10011101       0x9d
#define B10011110       0x9e
#define B10011111       0x9f

#define B10100000       0xa0
#define B10100001       0xa1
#define B10100010       0xa2
#define B10100011       0xa3
#define B10100100       0xa4
#define B10100101       0xa5
#define B10100110       0xa6
#define B10100111       0xa7
#define B10101000       0xa8
#define B10101001       0xa9
#define B10101010       0xaa
#define B10101011       0xab
#define B10101100       0xac
#define B10101101       0xad
#define B10101110       0xae
#define B10101111       0xaf

#define B10110000       0xb0
#define B10110001       0xb1
#define B10110010       0xb2
#define B10110011       0xb3
#define B10110100       0xb4
#define B10110101       0xb5
#define B10110110       0xb6
#define B10110111       0xb7
#define B10111000       0xb8
#define B10111001       0xb9
#define B10111010       0xba
#define B10111011       0xbb
#define B10111100       0xbc
```

3

CONSTBIN.H

```
#define B10111101       0xbd
#define B10111110       0xbe
#define B10111111       0xbf

#define B11000000       0xc0
#define B11000001       0xc1
#define B11000010       0xc2
#define B11000011       0xc3
#define B11000100       0xc4
#define B11000101       0xc5
#define B11000110       0xc6
#define B11000111       0xc7
#define B11001000       0xc8
#define B11001001       0xc9
#define B11001010       0xca
#define B11001011       0xcb
#define B11001100       0xcc
#define B11001101       0xcd
#define B11001110       0xce
#define B11001111       0xcf

#define B11010000       0xd0
#define B11010001       0xd1
#define B11010010       0xd2
#define B11010011       0xd3
#define B11010100       0xd4
#define B11010101       0xd5
#define B11010110       0xd6
#define B11010111       0xd7
#define B11011000       0xd8
#define B11011001       0xd9
#define B11011010       0xda
#define B11011011       0xdb
#define B11011100       0xdc
#define B11011101       0xdd
#define B11011110       0xde
#define B11011111       0xdf

#define B11100000       0xe0
#define B11100001       0xe1
#define B11100010       0xe2
#define B11100011       0xe3
#define B11100100       0xe4
#define B11100101       0xe5
#define B11100110       0xe6
#define B11100111       0xe7
#define B11101000       0xe8
#define B11101001       0xe9
#define B11101010       0xea
#define B11101011       0xeb
#define B11101100       0xec
#define B11101101       0xed
#define B11101110       0xee
#define B11101111       0xef

#define B11110000       0xf0
#define B11110001       0xf1
#define B11110010       0xf2
#define B11110011       0xf3
#define B11110100       0xf4
#define B11110101       0xf5
#define B11110110       0xf6
#define B11110111       0xf7
#define B11111000       0xf8
#define B11111001       0xf9
#define B11111010       0xfa
#define B11111011       0xfb
#define B11111100       0xfc
#define B11111101       0xfd
```

CONSTBIN.H

```
#define B11111110       0xfe
#define B11111111       0xff
```

## A.33　HD329.H

HD329.H

```
/*************************************************/
/*************************************************/

/*基本設定**/
typedef unsigned char  uchar ;
typedef unsigned short ushor ;
typedef unsigned short ushort;
typedef unsigned long  ulong ;
typedef unsigned int   uint ;

void    _enable();
void    _disable();
int     inp(uint);
int     outp(uint,int);
#pragma intrinsic( _enable, _disable )
#pragma intrinsic( inp, outp )

#define ON      0xff
#define OFF     0x00
#define DATA_H  216
#define LENGTH  216
#define TIMEUP          0x00


/*キー入力**/
#define FINES           0x140
#define STOPS           0x120
#define STARS           0x110
#define COPYS           0x108

/*センサー**/
#define GENCS           0x504
#define PAPES           0x502
#define COVES           0x501
#define HOOKS           0x402
#define RINGS           0x401
#define COUPS           0x202

/*モデム等**/
#define NCUCP           0x04
#define LINER           0x01
#define MDRST           0x04

/*ＬＥＤ’ｓ**/
#define COVEL           0x80
#define PAPEL           0x40
#define ERROL           0x20
#define FINEL           0x10
#define STARL           0x08
#define POWERL          0x04


/* ＭＯＴＯＲ **/
#define MOT_PR  0x10
#define MOT_IR  0x20


#define M_13    0x04
#define M_24    0x02

/*ｂｕｚｚ**/
#define BUZZ    0x04

/*ＩＭＲの制御**/
#define DMAE    0x01
#define DITH    0x08
#define CIS_P   0x08

/*プリントヘッド制御**/
```

1

保守／廃止

HD329.H

```
#define TH_LTC  0x40
#define TH_STB  0x02
#define TH_END  0x802
#define THM_P   0x10

/*ＮＭＩ・パワーホールド等*/
#define IN_NMI  0x01
#define PHOLD   0x04


/*ｐｐｉ　ｒｅｇｓ*/
#define PPIPA           0x60
#define PPIPB           0x61
#define PPIPC           0x62
#define PPICNT          0x63


#define PORTSTL         0x40
#define PORTSTH         0x41
#define PORTQ           0x42

#define MDCHG   0x20


/*********************************************
 *      ＲＡＭ領域                           *
 *********************************************/

extern  uchar   fdate[16];
extern  uchar   RNGM;
extern  uchar   RNGCM;
extern  uchar   TELNO[20];
extern  uchar   SPEED;
extern  uchar   LEV;

extern  ulong   TIME1;
extern  ushort  TIME2;
extern  uchar   BZSW;
extern  uchar   WINKP;
extern  uchar   WINK;


/*********************************************************************/
/*********************************************************************/

#define B_ADR   0x7000

#define SRAM_20 0x20000000
#define SRAM_21 0x20002000
#define SRAM_22 0x20004000
#define SRAM_23 0x20006000
#define PSRAM1  0x40000000
#define PSRAM2  0x50000000

struct  nama_data_s{
                uchar   arry[0x20][0x100];      /**xx[32][256]***/
        };
struct  comp_data_s{
                uchar   arry[0x2000];
        };
struct  trnp_tabl_s{
                uchar   arry[0x2000];
        };
struct  comp_ecm_data_s{
                uchar   arry[0x10000];
        };

#define MMIO_Modem      0x80000000;
#define MMIO_CE1        0x80001000;
```

2

保守／廃止

HD329.H

```
#define MMIO_RTC        0x80002000;
#define MMIO_PLA        0x80003000;

struct  RTC_DATA{       /* RAM */
        uchar   onesec;
        uchar   tensec;
        uchar   onemin;
        uchar   tenmin;
        uchar   onehour;
        uchar   tenhour;
        uchar   week;
        uchar   onedate;
        uchar   tendate;
        uchar   onemonth;
        uchar   tenmonth;
        uchar   oneyear;
        uchar   tenyear;
        uchar   rtccnt1;
        uchar   rtccnt2;
        uchar   rtcmode;
        };


/*************************************************
 *************************************************/

struct  time_data{
                uchar   year[2];
                uchar   month[2];
                uchar   date[2];
                uchar   hour[2];
                uchar   min[2];
        };

typedef struct {
                uchar   syojo[6];
                uchar   name[16];
                uchar   ring[1];
                uchar   repo1[1];
                uchar   repo2[1];
                uchar   tel[20];
                uchar   speed[1];
                uchar   year[2];
                uchar   month[2];
                uchar   date[2];
                uchar   hour[2];
                uchar   min[2];

                struct {
                        uchar   kanjo;
                        uchar   hizuke[16];
                        uchar   usetime[2];
                        uchar   youid[21];
                        uchar   soujusin;
                        uchar   count;
                        uchar   result;
                        } KANRI[35];
                } B_INFO;

/*****************************************************************/
/*【付録】ポート入出力一覧表

name    bit7    bit6    bit5    bit4    bit3    bit2    bit1    bit0

P0X     COVER   PAPER   ERROR   PINEL   STARTL  (POWER) ----    NMIE
        open    empty

P1X     ----    FINE    STOP    START   (COPY   ----    ----    ----
                key     key     key      key)
```

HD329.H

```
P2X  ----   ----   ----   THMP   CISP   /MDMRES CPLD   ----
P3X  ----   ----   ----   ----   CABL2  ----    CABL1  ----
P4X  ----   THLT   /RMOTE  /PMOTE DIZ    ----    HOOK   RING
P5X  ----   ----   ----   ----   ----   GENCO   PAPER  COVER
                                       sensor  sensor sensor
P7X  MPH0   MPH1   MPH2   MPH3   ----   PSTN    ----   LINE
                                       /CPL            /PHN
P8X  ----   ----   ----   ----   ----   ----    STBSENS ----
PLA  /RFSH  ----   ----   ----   ----   /BZE    STRS   DMAE
**/
/*****************************************************************/
```

## A.34　INDAT55.H

INDAT55.H

```
/*************************************************************************/
/*                                                                       */
/*              Ｖ５５ＰＩ特殊機能レジスタ領域のマップ                   */
/*                                                                       */
/*************************************************************************/

/*特殊機能レジスタ領域ベースアドレス**/
#define base            0xff000e00

/*特殊機能レジスタ構造体タグ定義**/
struct  v55pi_regs{
                uchar   ADCR0;          /* Ａ／Ｄコンバージョン・リザルト・レジ０～３ **/
                        uchar reserve_E01[1];
                uchar   ADCR1;
                        uchar reserve_E03[1];
                uchar   ADCR2;
                        uchar reserve_E05[1];
                uchar   ADCR3;
                        uchar reserve_E07[9];

                uchar   PAB;            /* パラレル・Ｉ／Ｆ・バッファ **/
                        uchar reserve_E11[7];
                uchar   PAC0;           /* パラレル・Ｉ／Ｆ・コントロール・レジ０～１ **/
                uchar   PAC1;
                uchar   PAS;            /* パラレル・Ｉ／Ｆ・ステータス・レジ **/
                        uchar reserve_E1B[1];
                uchar   PAI1;           /* パラレル・Ｉ／Ｆ・アクノリッジ・インタバル・レジ
１～２ **/
                uchar   PAI2;
                        uchar reserve_E1E[2];

                uchar   ADM;            /* Ａ／Ｄコンバータ・モード・レジ **/
                        uchar reserve_E21[0x9f];

                ushort  MK0;            /* 割り込みマスク・フラグ・レジ０～１ **/
                ushort  MK1;
                uchar   ISPR;           /* インサービス・プライオリティ・レジ **/
                uchar   IMC;            /* 割り込みモード・コントロール・レジ **/
                        uchar reserve_EC6[3];

                uchar   IC09;           /* 割り込み要求制御レジ０９～３７ **/
                uchar   IC10;
                uchar   IC11;
                uchar   IC12;
                uchar   IC13;
                uchar   IC14;
                        uchar reserve_ECF[1];
                uchar   IC16;
                uchar   IC17;
                uchar   IC18;
                uchar   IC19;
                uchar   IC20;
                uchar   IC21;
                uchar   IC22;
                uchar   IC23;
                uchar   IC24;
                uchar   IC25;
                uchar   IC26;
                uchar   IC27;
                uchar   IC28;
                uchar   IC29;
                uchar   IC30;
                uchar   IC31;
                uchar   IC32;
                        uchar reserve_EE1[3];
                uchar   IC36;
                uchar   IC37;
                        uchar reserve_EE6[0x1a];

                uchar   P0;             /* ポート０～８ **/
```

1

保守／廃止

INDAT55.H

```
uchar   P1;
uchar   P2;
uchar   P3;
uchar   P4;
uchar   P5;
uchar   P6;
uchar   P7;
uchar   P8;
        uchar reserve_F09[3];
uchar   PRDC;           /* ポート・リード・コントロール・レジ **/
        uchar reserve_F0D[1];
uchar   RTP;            /* リアルタイム出力ポート **/
        uchar reserve_F0F[1];
uchar   PM0;            /* ポート０～８モード・レジ **/
        uchar reserve_F11[1];
uchar   PM2;
uchar   PM3;
uchar   PM4;
uchar   PM5;
        uchar reserve_F16[1];
uchar   PM7;
uchar   PM8;
        uchar reserve_F19[9];
uchar   PMC2;           /* ポート２～８モード・コントロール・レジ **/
uchar   PMC3;
uchar   PMC4;
uchar   PMC5;
        uchar reserve_F26[1];
uchar   PMC7;
uchar   PMC8;
        uchar reserve_F29[3];
uchar   RTPC;           /* リアルタイム出力ポート・コントロール・レジ **/
uchar   RTPD;           /* リアルタイム出力ポート・ディレイ指定レジ **/
uchar   P7L;            /* ポート７バッファ，下位 **/
uchar   P7H;            /* ポート７バッファ，上位 **/

uchar   TMC0;           /* タイマ・コントロール・レジ０～１ **/
uchar   TMC1;
uchar   TOC0;           /* タイマ出力制御レジ０～１ **/
uchar   TOC1;
uchar   INTM0;          /* 外部割り込みモード・レジ０～１ **/
uchar   INTM1;
        uchar reserve_F36[0x0a];

ushort  TM0;            /* タイマ・レジ０～３ **/
ushort  TM1;
ushort  TM2;
ushort  TM3;
ushort  CT00;           /* タイマ・キャプチャ・レジ００～０１ **/
ushort  CT01;
ushort  CM00;           /* タイマ・コンペア・レジ００～０１ **/
ushort  CM01;
ushort  CT10;           /* タイマ・キャプチャ・レジ１０ **/
ushort  CM10;           /* タイマ・コンペア・レジ１０～１１ **/
ushort  CM11;
        uchar reserve_F56[2];
ushort  CM20;           /* タイマ・コンペア・レジ２０～２３ **/
ushort  CM21;
ushort  CM22;
ushort  CM23;
ushort  WDM;            /* ヲッチドッグ・タイマ・モード・レジ **/
        uchar reserve_F62[2];
ushort  CM30;           /* タイマ・コンペア・レジ３０～３１ **/
ushort  CM31;
        uchar reserve_F68[4];

uchar   PWM;            /* ＰＷＭレジ **/
uchar   PWMC;           /* ＰＷＭコントロール・レジ **/
        uchar reserve_F6E[2];
```

2

保守／廃止

INDAT55.H

```
                uchar   TxBRG0;         /* 送信ビットレート・ジェネレータ・レジ0 **/
                uchar   RxBRG0;         /* 受信ビットレート・ジェネレータ・レジ0 **/
                uchar   PRS0;           /* プリスケーラ・レジ0 **/
                uchar   UARTM0_CSIM0;   /* ＵＡＲＴモード・レジ0／クロックト・シリアル・Ｉ
／Ｆ・モード・レジ0 **/
                uchar   UARTS0_SBIC0;   /* ＵＡＲＴステータス・レジ0／ＳＢＩコントロール・
レジ0 **/
                uchar   TxB0_SIO0;      /* 送信バッファ0 **/
                uchar   RxB0;           /* 受信バッファ0 **/
                        uchar reserve_F77[1];
                uchar   TxBRG1;         /* 送信ビットレート・ジェネレータ・レジ1 **/
                uchar   RxBRG1;         /* 受信ビットレート・ジェネレータ・レジ1 **/
                uchar   PRS1;           /* プリスケーラ・レジ1 **/
                uchar   UARTM1_CSIM1;   /* ＵＡＲＴモード・レジ1／クロックト・シリアル・Ｉ
／Ｆ・モード・レジ1 **/
                uchar   U1RTS1;         /* ＵＡＲＴステータス・レジ1 **/
                uchar   TxB1_SIO1;      /* 送信バッファ1 **/
                uchar   RxB1;           /* 受信バッファ1 **/
                uchar   ASP;            /* プロトコル選択レジ **/

                ulong   TC0;            /* ターミナル・カウンタ0 **/
                ulong   TCM0;           /* ターミナル・カウンタ・モジュロ・レジ0 **/
                ulong   UDC0;           /* ＤＭＡアップ・ダウン・カウンタ0 **/
                ulong   DCM0;           /* ＤＭＡコンペア・レジ0 **/
                ulong   MAR0;           /* ＤＭＡメモリアドレス・レジ0 **/
                ulong   DPTC0;          /* ＤＭＡリードライト・ポインタ0 **/
                        uchar reserve_F98[4];
                uchar   DMAM0;          /* ＤＭＡモード・レジ0 **/
                uchar   DMAC0;          /* ＤＭＡコントロール・レジ0 **/
                uchar   DMAS;           /* ＤＭＡステータス・レジ0 **/
                        uchar reserve_F9F[1];
                ulong   TC1;            /* ターミナル・カウンタ1 **/
                ulong   TCM1;           /* ターミナル・カウンタ・モジュロ・レジ1 **/
                ulong   UDC1;           /* ＤＭＡアップ・ダウン・カウンタ1 **/
                ulong   DCM1;           /* ＤＭＡコンペア・レジ1 **/
                ulong   MAR1;           /* ＤＭＡメモリアドレス・レジ1 **/
                ulong   DPTC1;          /* ＤＭＡリードライト・ポインタ1 **/
                        uchar reserve_FB8[4];
                uchar   DMAM1;          /* ＤＭＡモード・レジ1 **/
                uchar   DMAC1;          /* ＤＭＡコントロール・レジ1 **/
                        uchar reserve_FBE[0x22];

                ushort  STC;            /* ソフトウェア・タイマ・カウンタ **/
                ushort  STMC;           /* ソフトウェア・タイマ・カウンタ・コンペア・レジ **
/
                        uchar reserve_FE4[4];

                uchar   PWC0;           /* プログラマブル・ウェイト・コントロール・レジ0～
1 **/
                uchar   PWC1;

                uchar   MBC;            /* メモリ・ブロック・コントロール・レジ **/
                        uchar reserve_FEB[1];
                uchar   RFM;            /* リフレッシュ・モード・レジ **/
                        uchar reserve_FED[1];
                uchar   STBC;           /* スタンバイ・コントロール・レジ **/
                uchar   PRC;            /* プロセッサ・コントロール・レジ **/
                        uchar reserve_FF0[0X10];
        };
```

3

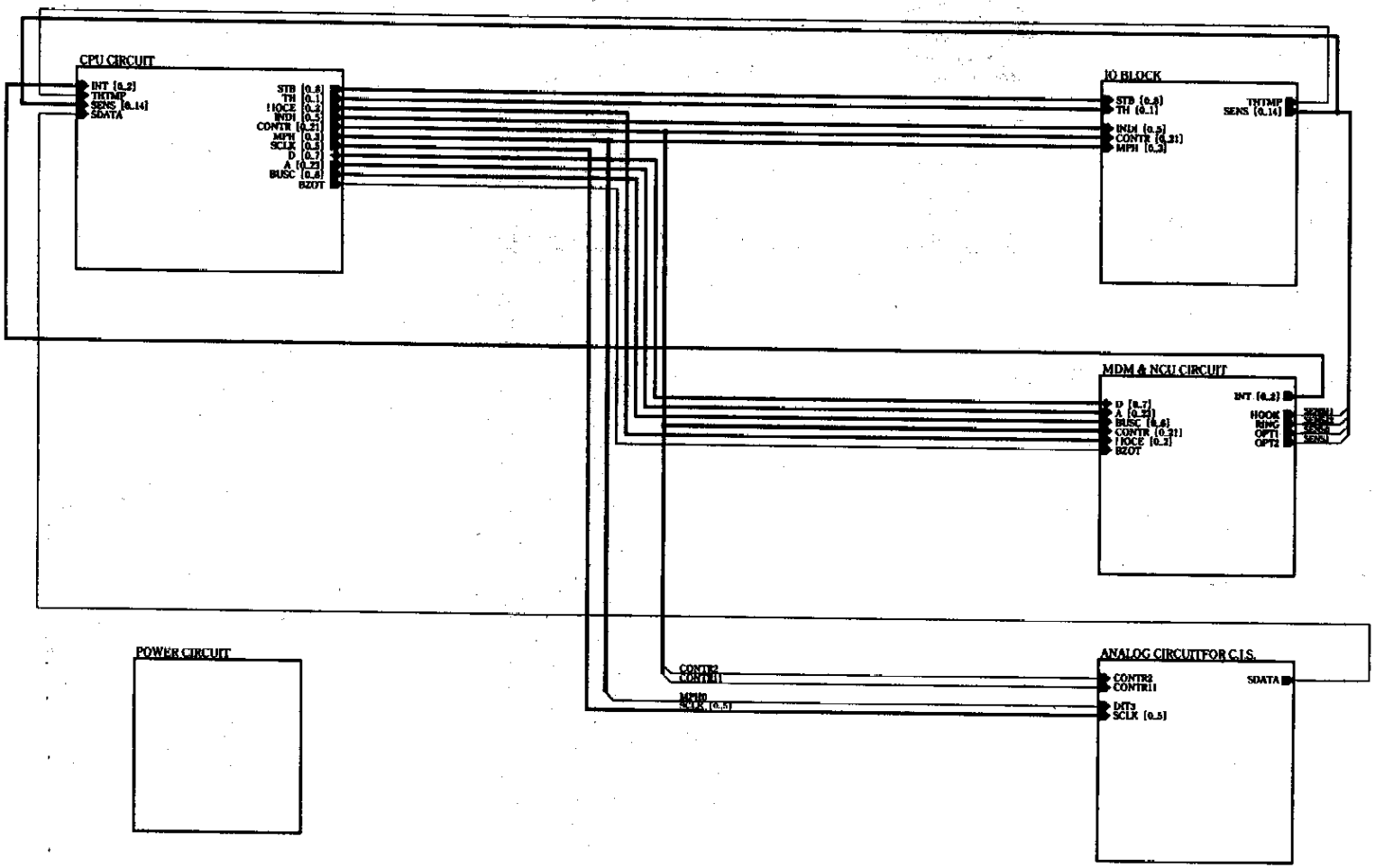
〔メ モ〕

# 付録B　回 路 図

〔メ モ〕

保守／廃止

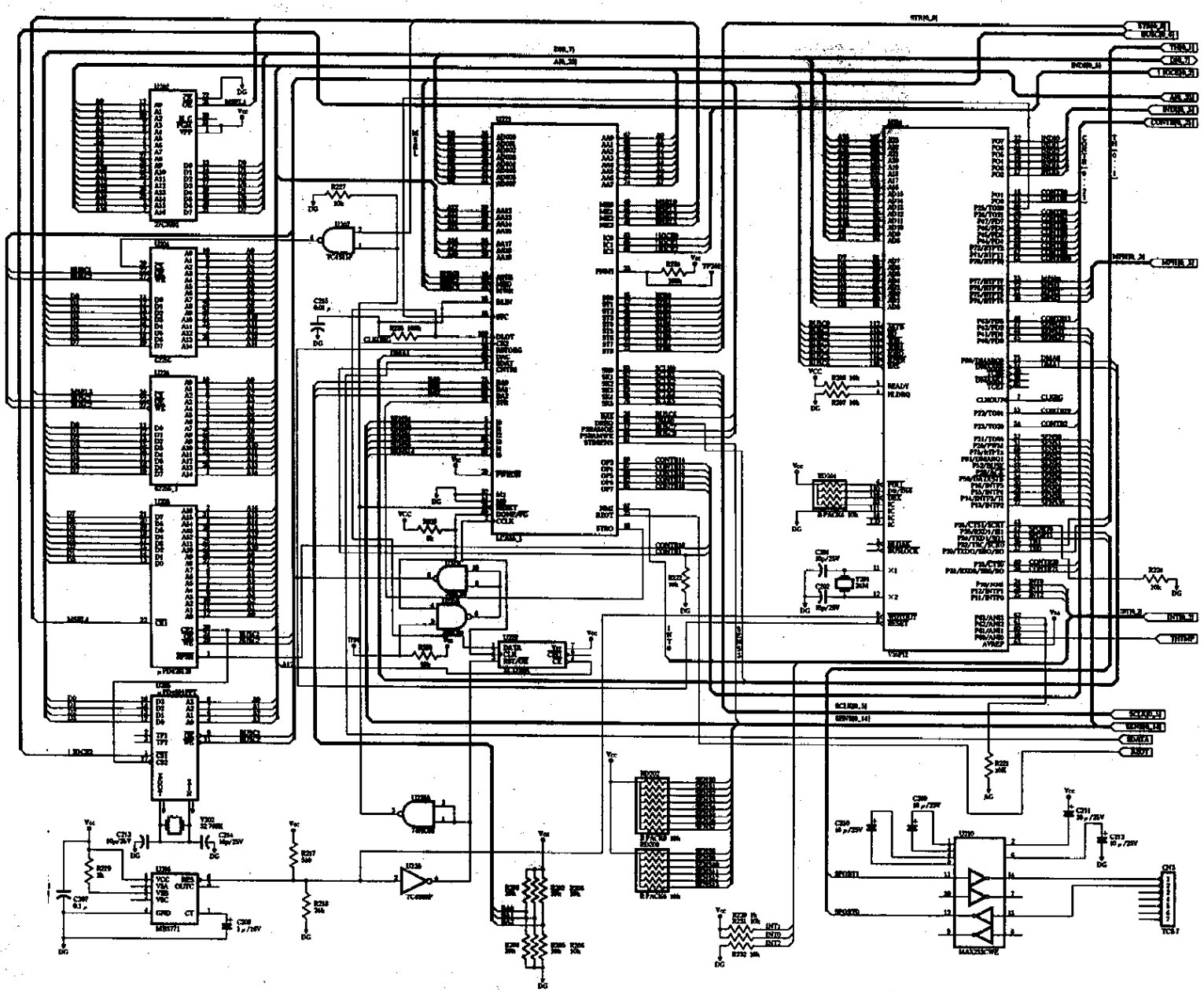
(1) 全体回路図


CPU CIRCUIT
INT [0..2]
TNTMP
SENS [0..14]
SDATA
STB [0..1]
TH [0..1]
!IOCE [0..2]
INDI [0..5]
CONTR [0..21]
MPH [0..3]
SCLK [0..5]
D [0..7]
A [0..23]
BUSC [0..6]
BZOT
IO BLOCK
STB [0..1]
TH [0..1]
INDI [0..5]
CONTR [0..21]
MPH [0..3]
TNTMP
SENS [0..14]
MDM & NCU CIRCUIT
D [0..7]
A [0..23]
BUSC [0..6]
CONTR [0..21]
!IOCE [0..2]
BZOT
INT [0..2]
HOOK
RING
OPT1
OPT2
POWER CIRCUIT
CONTR2
CONTR11
MPH0
SCLK [0..5]
ANALOG CIRCUITFOR C.I.S.
CONTR2
CONTR11
INT3
SCLK [0..5]
SDATA

# 空白ページ

保守／廃止

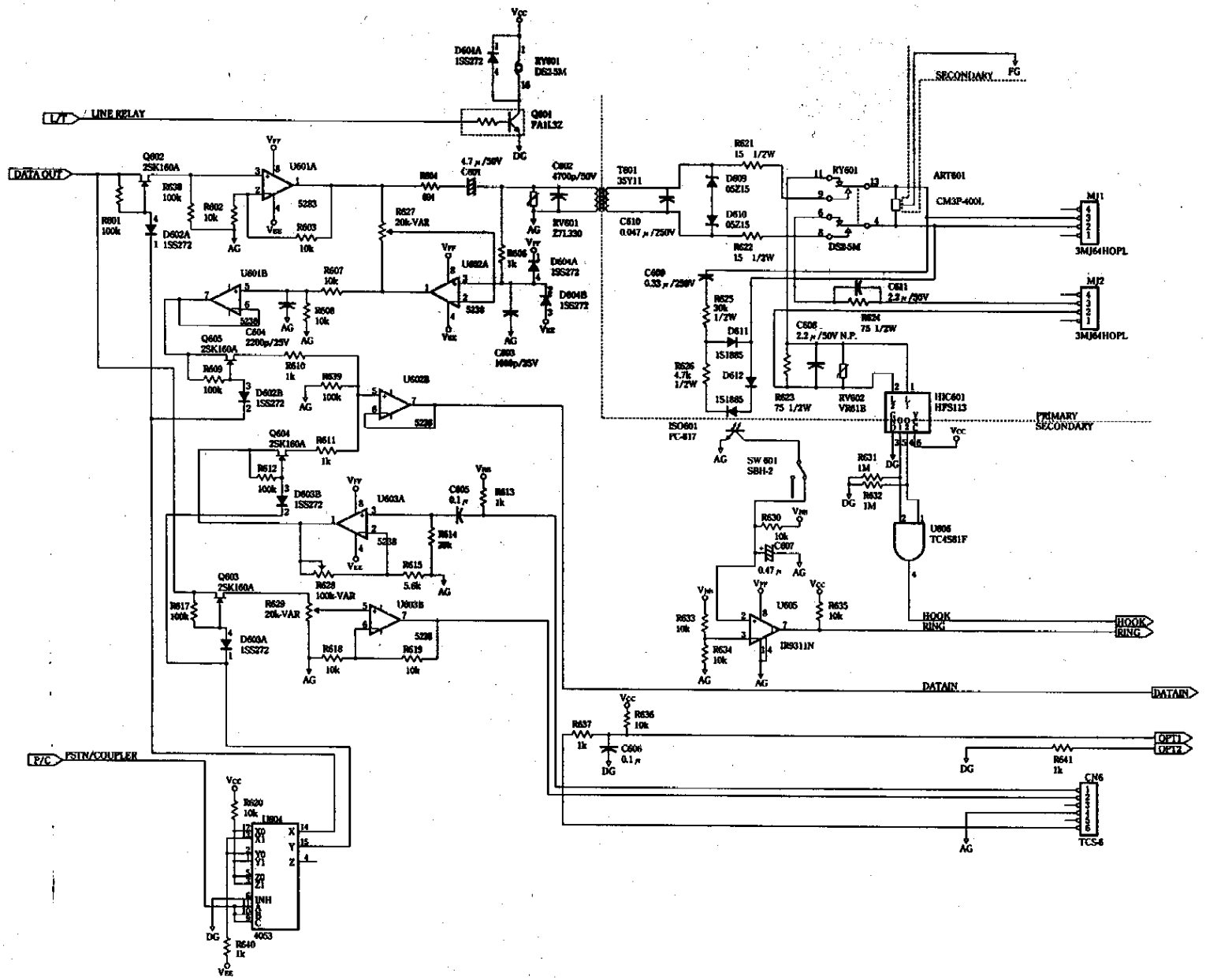
(2) CPU周り



保守／廃止

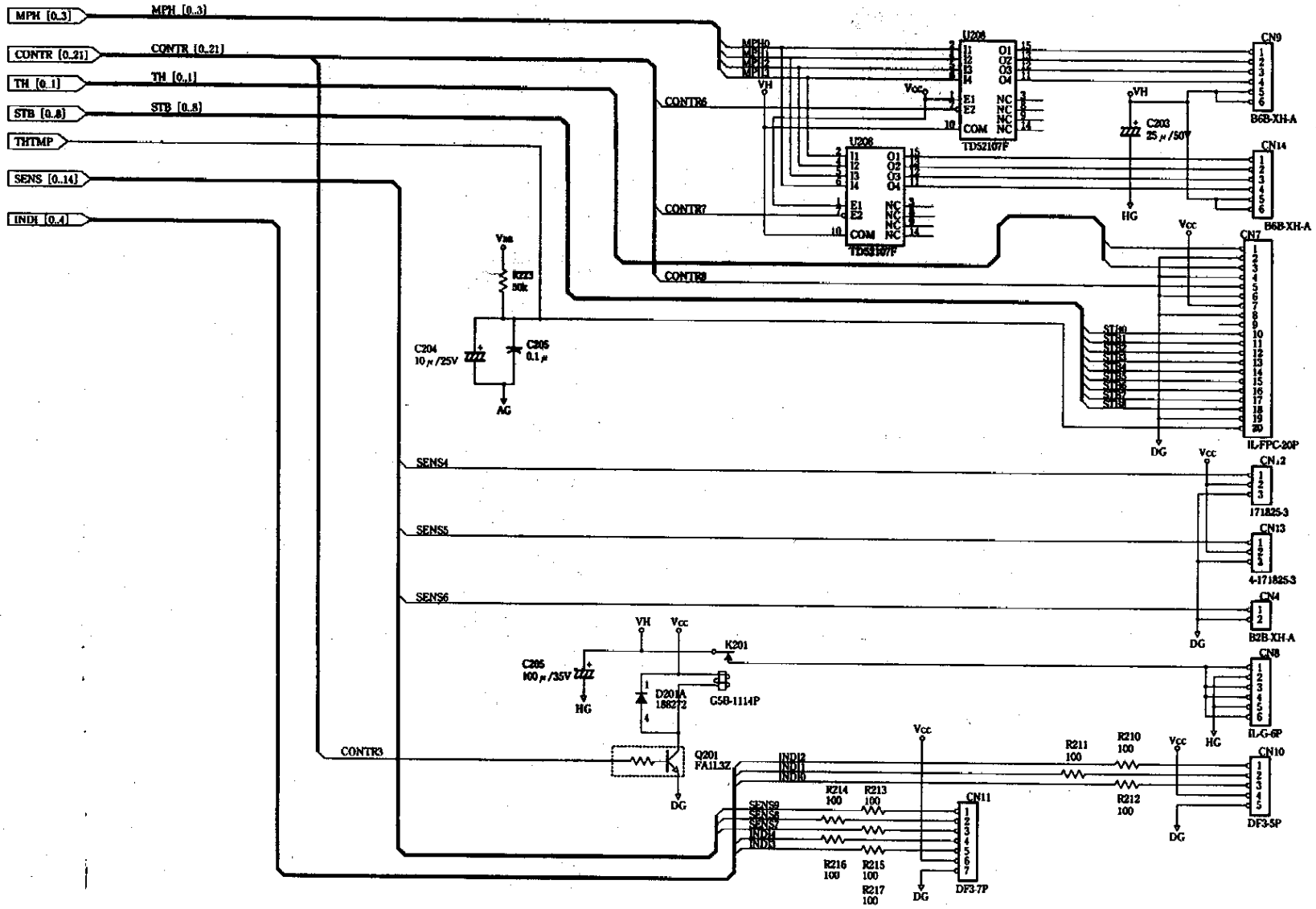
# 空白ページ

(3) MDM DATA SELECTOR & NCU



# 空白ページ

(4) I/O CIRCUIT FOR VAR.CNS

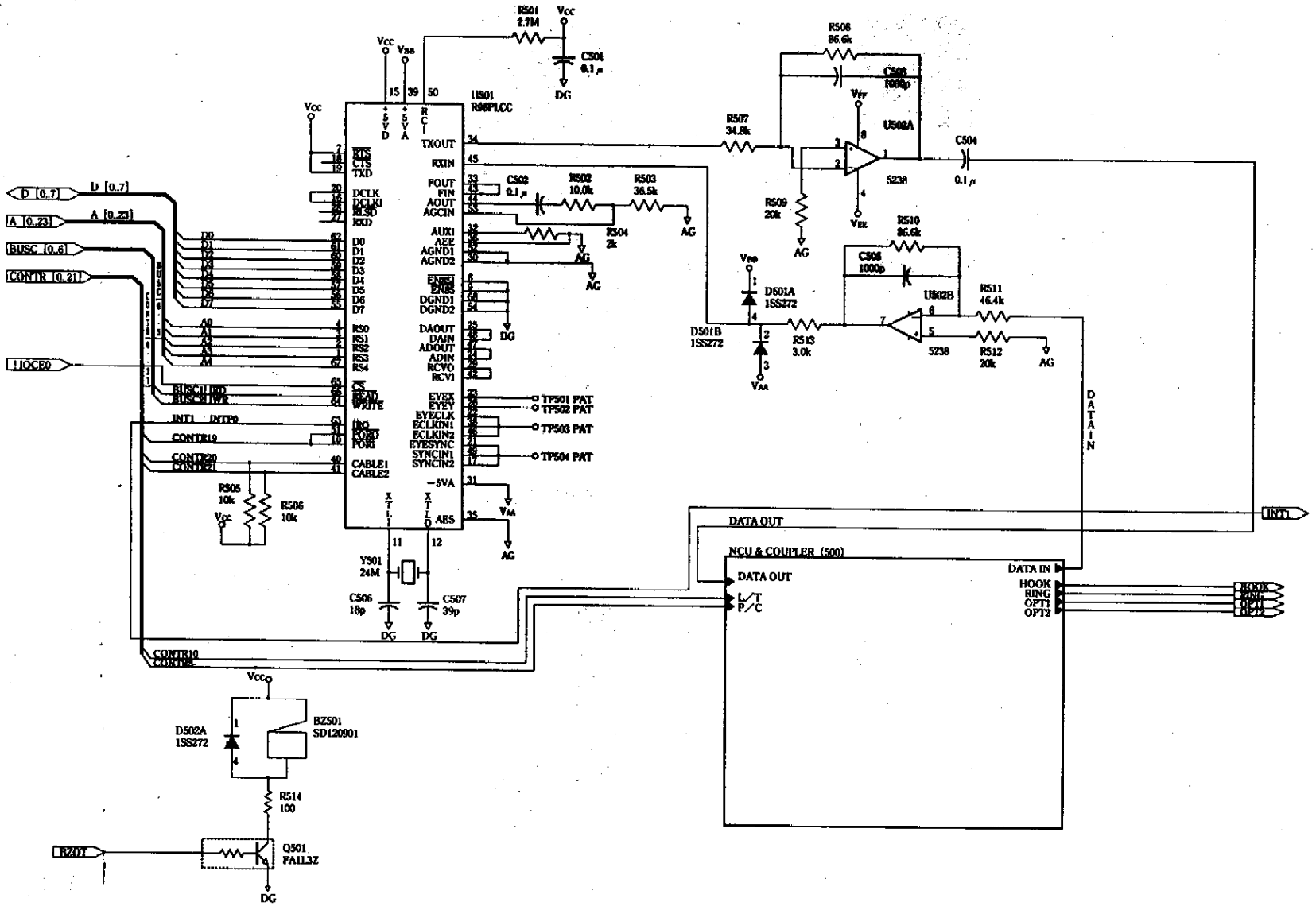


保守／廃止

# 空白ページ

保守／廃止

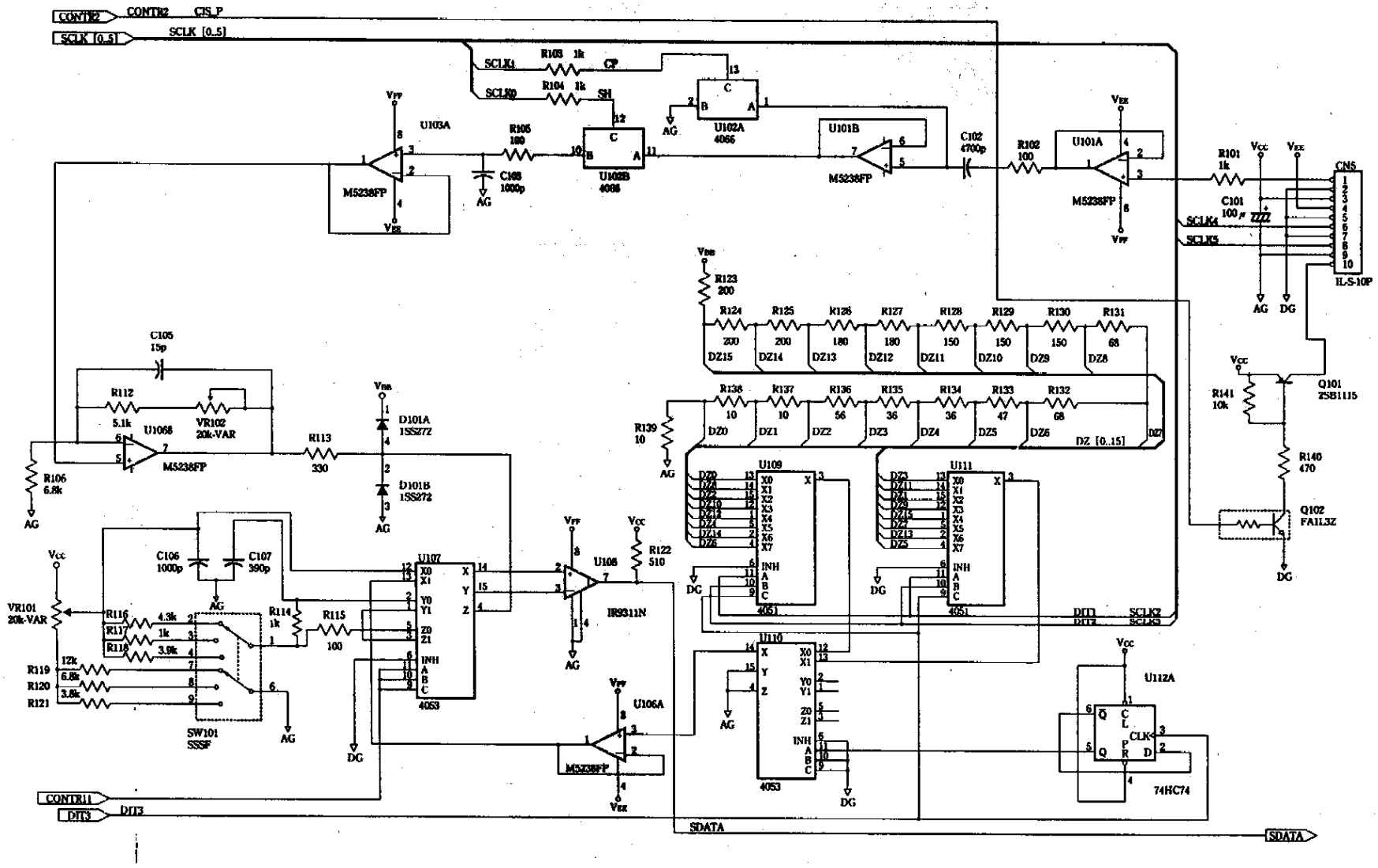
(5) MDM & ANALOG



保守／廃止

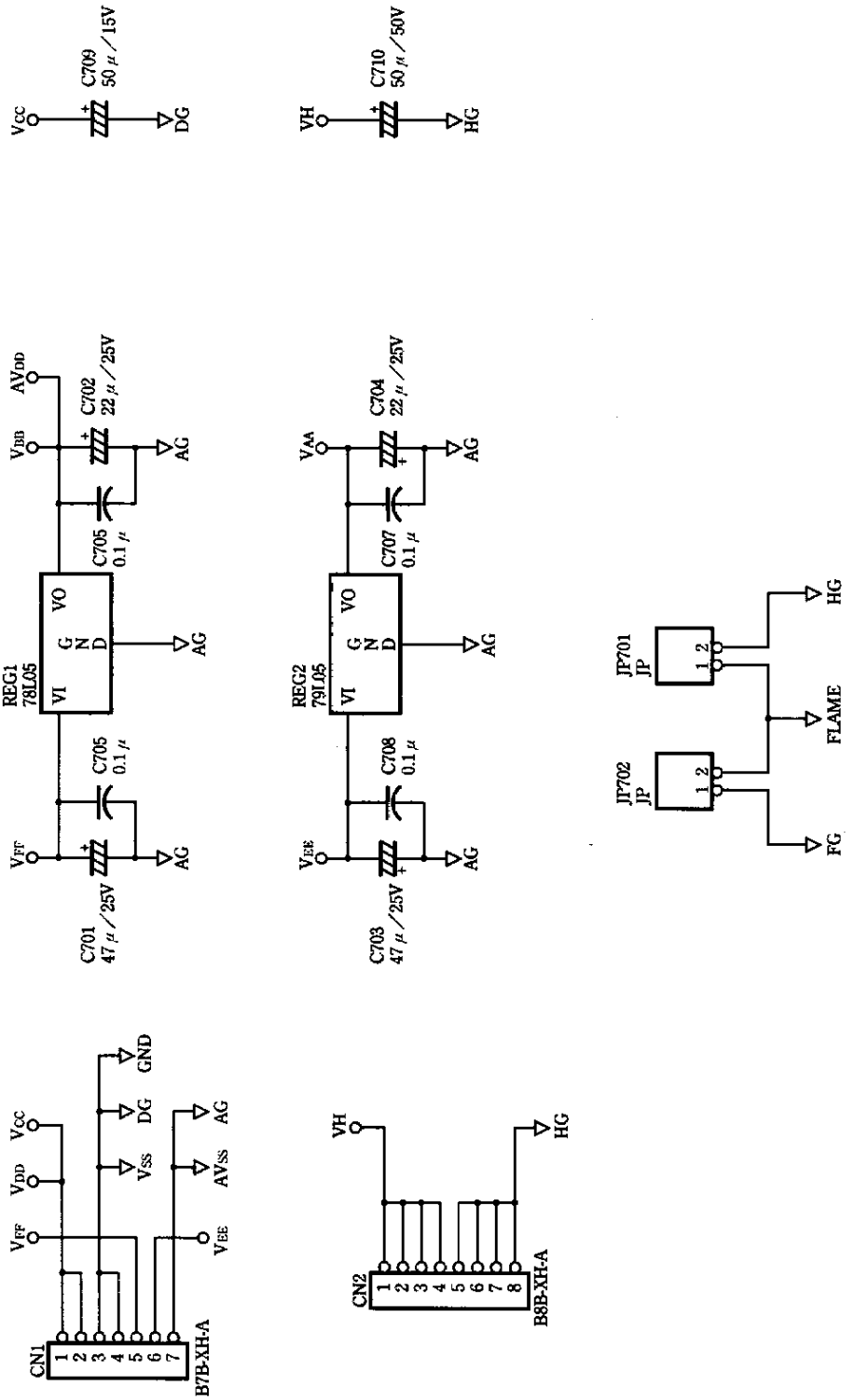
# 空白ページ

(6) ANALOG CIRCUIT FOR MODEM



# 空白ページ

(7) POWER SOURCE



## 品名一覧 (1/4)

| Item | Quantity | Reference | Part | 品名 | メーカー名 |
|---|---|---|---|---|---|
| 1 | 1 | ART601 | CM3P-400L | アレスタ | 石塚電子 |
| 2 | 1 | BZ501 | SD120901 | サウンデューサ | TDK |
| 3 | 1 | C101 | 100u | 電解コンデンサ | エルナ |
| 4 | 1 | C102 | 4700p | セラミック・コンデンサ | KCK |
| 5 | 4 | C103,C106,C503,C505 | 1000p | セラミック・コンデンサ | KCK |
| 6 | 5 | C204,C209,C210,C211,C212 | 10u/25V | 電解コンデンサ | エルナ |
| 7 | 1 | C105 | 15p | セラミック・コンデンサ | KCK |
| 8 | 1 | C107 | 390p | セラミック・コンデンサ | KCK |
| 9 | 11 | C205,C207,C501,C502,C504,C605, C606,C705,C706,C707,C708 | 0.1u | セラミック・コンデンサ | KCK |
| 10 | 1 | C208 | 1u/16V | タンタル・コンデンサ | エルナ |
| 11 | 1 | C506 | 18p | セラミック・コンデンサ | KCK |
| 12 | 1 | C507 | 39p | セラミック・コンデンサ | KCK |
| 13 | 1 | C601 | 4.7u/50V | 電解コンデンサ | エルナ |
| 14 | 1 | C602 | 4700p/50V | フィルム・コンデンサ | KCK |
| 15 | 1 | C603 | 1000p/25V | セラミック・コンデンサ | KCK |
| 16 | 1 | C604 | 2200p/25V | セラミック・コンデンサ | KCK |
| 17 | 4 | C201,C202,C213,C214 | 10p/25V | セラミック・コンデンサ | KCK |
| 18 | 1 | C607 | 0.47u | 電解コンデンサ | エルナ |
| 19 | 1 | C608 | 2.2u/50V N.P. | 電解コンデンサ | エルナ |
| 20 | 1 | C609 | 0.33u/250V | フィルム・コンデンサ | エルナ |
| 21 | 1 | C610 | 0.047u/250V | フィルム・コンデンサ | エルナ |
| 22 | 1 | C611 | 2.2u/50V | 電解コンデンサ | エルナ |
| 23 | 2 | C701,C703 | 47u/25V | 電解コンデンサ | エルナ |
| 24 | 2 | C702,C704 | 22u/25V | 電解コンデンサ | エルナ |
| 25 | 1 | CN1 | B7B-XH-A | コネクタ | 日圧 |
| 26 | 1 | CN2 | B8B-XH-A | コネクタ | 日圧 |
| 27 | 1 | CN4 | B2B-XH-A | コネクタ | 日圧 |
| 28 | 1 | CN5 | IL-S-10P | コネクタ | 航空電子 |
| 29 | 1 | CN6 | TCS-6 | コネクタ | ホシデン |
| 30 | 1 | CN7 | IL-FPC-20P | コネクタ | 航空電子 |
| 31 | 1 | CN8 | IL-G-6P | コネクタ | 航空電子 |
| 32 | 2 | CN9,CN14 | B6B-XH-A | コネクタ | 日圧 |
| 33 | 1 | CN10 | DF3-5P | コネクタ | ヒロセ |
| 34 | 1 | CN11 | DF3-7P | コネクタ | ヒロセ |
| 35 | 1 | CN12 | 171825-3 | コネクタ | 日本アンプ |
| 36 | 1 | CN13 | 4-171825-3 | コネクタ | 日本アンプ |

保守／廃止

## 品名一覧 (2/4)

| Item | Quantity | Reference | Part | 品名 | メーカー名 |
|---|---|---|---|---|---|
| 37 | 8 | D101,D201,D501,D502,D601,D602, D603,D604 | 1SS272 | ダイオード | 東芝 |
| 38 | 2 | D611,D612 | 1S1885 | ダイオード | 東芝 |
| 39 | 2 | D609,D610 | 05Z15 | ダイオード | NEC |
| 40 | 1 | HIC601 | HFS113 | カレント・ディテクタ | 村田製作所 |
| 41 | 1 | IS0601 | PC-817 | オプト・アイソレータ | シャープ |
| 42 | 2 | MJ1,MJ2 | 3MJ64HOPL | モジュラ・ジャック | 沖電線 |
| 43 | 1 | Q101 | 2SB1115 | トランジスタ | NEC |
| 44 | 4 | Q102,Q201,Q501,Q601 | FA1L3Z | 複合トランジスタ | NEC |
| 45 | 30 | R141,R201,R202,R203,R204,R205, R206,R207,R208,R221,R222,R223, R224,R227,R231,R232,R505,R506, R602,R603,R607,R608,R618,R619, R620,R630,R633,R634,R635,R636 | 10K | チップ抵抗 1/8W | 北陸抵抗器 |
| 46 | 14 | R101,R103,R104,R114,R117,R219, R230,R606,R610,R611,R613,R637, R640,R641 | 1K | チップ抵抗 1/8W | 北陸抵抗器 |
| 47 | 12 | R102,R105,R115,R210,R211,R212, R213,R214,R215,R216,R217,R514 | 100 | チップ抵抗 1/8W | 北陸抵抗器 |
| 48 | 2 | R106,R120 | 6.8K | チップ抵抗 1/8W | 北陸抵抗器 |
| 49 | 3 | R509,R512,R614 | 20K | チップ抵抗 1/8W | 北陸抵抗器 |
| 50 | 2 | R631,R632 | 1M | チップ抵抗 1/8W | 北陸抵抗器 |
| 51 | 1 | R112 | 5.1K | チップ抵抗 1/8W | 北陸抵抗器 |
| 52 | 1 | R113 | 330 | チップ抵抗 1/8W | 北陸抵抗器 |
| 53 | 1 | R116 | 4.3K | チップ抵抗 1/8W | 北陸抵抗器 |
| 54 | 1 | R118 | 3.9K | チップ抵抗 1/8W | 北陸抵抗器 |
| 55 | 1 | R119 | 12K | チップ抵抗 1/8W | 北陸抵抗器 |
| 56 | 1 | R121 | 3.8K | チップ抵抗 1/8W | 北陸抵抗器 |
| 57 | 2 | R122,R217 | 510 | チップ抵抗 1/8W | 北陸抵抗器 |
| 58 | 3 | R123,R124,R125 | 200 | チップ抵抗 1/8W | 北陸抵抗器 |
| 59 | 2 | R126,R127 | 180 | チップ抵抗 1/8W | 北陸抵抗器 |
| 60 | 3 | R128,R129,R130 | 150 | チップ抵抗 1/8W | 北陸抵抗器 |
| 61 | 2 | R131,R132 | 68 | チップ抵抗 1/8W | 北陸抵抗器 |
| 62 | 1 | R133 | 47 | チップ抵抗 1/8W | 北陸抵抗器 |
| 63 | 2 | R134,R135 | 36 | チップ抵抗 1/8W | 北陸抵抗器 |
| 64 | 1 | R136 | 56 | チップ抵抗 1/8W | 北陸抵抗器 |
| 65 | 3 | R137,R138,R139 | 10 | チップ抵抗 1/8W | 北陸抵抗器 |
| 66 | 1 | R140 | 470 | チップ抵抗 1/8W | 北陸抵抗器 |
| 67 | 1 | R218 | 24K | チップ抵抗 1/8W | 北陸抵抗器 |
| 68 | 1 | R501 | 2.7M | チップ抵抗 1/8W | 北陸抵抗器 |

## 品名一覧 (3/4)

| Item | Quantity | Reference | Part | 品名 | メーカー名 |
|---|---|---|---|---|---|
| 69 | 1 | R502 | 10.0k | チップ抵抗 1/8W | 北陸抵抗器 |
| 70 | 1 | R503 | 36.5k | チップ抵抗 1/8W | 北陸抵抗器 |
| 71 | 1 | R504 | 2k | チップ抵抗 1/8W | 北陸抵抗器 |
| 72 | 1 | R507 | 34.8k | チップ抵抗 1/8W | 北陸抵抗器 |
| 73 | 2 | R508,R510 | 86.6k | チップ抵抗 1/8W | 北陸抵抗器 |
| 74 | 1 | R511 | 46.4k | チップ抵抗 1/8W | 北陸抵抗器 |
| 75 | 1 | R513 | 3.0k | チップ抵抗 1/8W | 北陸抵抗器 |
| 76 | 8 | R234,R235,R601,R609,R612,R617, R638,R639 | 100k | チップ抵抗 1/8W | 北陸抵抗器 |
| 77 | 1 | R604 | 604 | チップ抵抗 1/8W | 北陸抵抗器 |
| 78 | 1 | R615 | 5.6k | チップ抵抗 1/8W | 北陸抵抗器 |
| 79 | 2 | R621,R622 | 15 | チップ抵抗 1/8W | 北陸抵抗器 |
| 80 | 2 | R623,R624 | 75 | チップ抵抗 1/8W | 北陸抵抗器 |
| 81 | 1 | R625 | 30k | チップ抵抗 1/8W | 北陸抵抗器 |
| 82 | 1 | R626 | 4.7k | チップ抵抗 1/8W | 北陸抵抗器 |
| 83 | 1 | REG1 | 78L05 | レギュレータ | NEC |
| 84 | 1 | REG2 | 79L05 | レギュレータ | NEC |
| 85 | 1 | RV601 | Z7L330 | バリスタ | 新電元工業 |
| 86 | 1 | RV602 | VR61B | バリスタ | 新電元工業 |
| 87 | 1 | RY601 | DS2-5M | リレー | 松下 |
| 88 | 1 | SW101 | SSSF | スイッチ | アルプス |
| 89 | 1 | T601 | 35Y11 | ライン・トランス | 田村製作所 |
| 90 | 4 | TP501,TP502,TP503,TP504 | PAT | テスト・パット | |
| 91 | 1 | U112 | 74HC74 | IC | NEC |
| 92 | 3 | U101,U103,U106 | M5238FP | IC | 三菱電機 |
| 93 | 1 | U102 | 4066 | IC | NEC |
| 94 | 2 | U108,U605 | IR9311N | IC | シャープ |
| 95 | 3 | U107,U110,U604 | 4053 | IC | NEC |
| 96 | 2 | U109,U111 | 4051 | IC | NEC |
| ★ 97 | 1 | U204 | 43256 | S-RAM | NEC |
| ★ 98 | 1 | U224 | 43256_1 | S-RAM | NEC |
| 99 | 1 | U225 | 74HC00 | IC | NEC |
| 100 | 1 | U205 | μPD4991FPT | RTC | NEC |
| 101 | 1 | U207 | TC4S11F | 1ゲートIC | 東芝 |
| 102 | 2 | U208,U209 | TD62107F | トランジスタ・アレイ | 東芝 |
| 103 | 1 | U206 | MB3771 | IC | 富士通 |
| 104 | 1 | U501 | R96PLCC | モデム | ロックウェル |

## 品名一覧 (4/4)

| Item | Quantity | Reference | Part | 品名 | メーカー名 |
|---|---|---|---|---|---|
| 105 | 4 | U502,U601,U602,U603 | 5238 | IC | 三菱電機 |
| 106 | 1 | U606 | TC4S81F | 1ゲートIC | 東芝 |
| 107 | 2 | Y201,Y501 | 24M | クリスタル | TEW |
| 108 | 1 | Y202 | 32.768K | クリスタル | TEW |
| 109 | 4 | Q602,Q603,Q604,Q605 | 2SK160A | トランジスタ | |
| 110 | 1 | U203 | μPD428128 | 疑似SRAM | NEC |
| 111 | 1 | U202 | 27C1001 | EPROM | NEC |
| 112 | 4 | VR101,VR102,R627,R629 | 20k-VAR | 可変抵抗 | |
| 113 | 1 | R628 | 100k-VAR | 可変抵抗 | |
| 114 | 1 | C206 | 100u/35V | 電解コンデンサ | エルナ |
| 115 | 1 | C709 | 50u/16V | 電解コンデンサ | エルナ |
| 116 | 1 | CN3 | TCS-7 | ミニチュア・ジャック | 星電器 |
| 117 | 2 | JP701,JP702 | JP | スルー・ホール2.54 | |
| 118 | 1 | K201 | G6B-1114P | パワー・リレー | オムロン |
| 119 | 1 | RD201 | R-PACK4 10k | ラダー抵抗 10k | 北陸抵抗器 |
| 120 | 1 | RD202 | R-PACK8 10k | ラダー抵抗 10k | 北陸抵抗器 |
| 121 | 1 | RD203 | R-PACK6 10k | ラダー抵抗 10k | 北陸抵抗器 |
| 122 | 1 | R223 | 50k | チップ抵抗 1/8W | 北陸抵抗器 |
| 123 | 1 | R228 | 5k | チップ抵抗 1/8W | 北陸抵抗器 |
| 124 | 1 | C203 | 25u/50V | 電解コンデンサ | エルナ |
| 125 | 1 | C710 | 50u/50V | 電解コンデンサ | エルナ |
| 126 | 1 | U210 | MAX232CWE | RS232D&R | MAXIM |
| 127 | 1 | U222 | XC1736A | シリアルPROM | XILINX |
| 128 | 1 | U223 | TC4S69F | 1ゲートIC | 東芝 |
| 129 | 1 | SW 601 | SBH-2 | スライド・スイッチ | 三星 |
| 130 | 1 | U201 | V55PI2 | MPU | NEC |
| 131 | 1 | U221 | LCA55_1 | ゲートアレイ | XILINX |

〔メ モ〕

# 付録Ｃ　コネクタの接続　★

保守／廃止

# コネクタの接続

各コネクタの信号内容を以下に示します。

## ●CN1，2

各電源供給を受けるコネクタです。

CN1

| | |
|---|---|
| No.1 | +5 V |
| 2 | +5 V |
| 3 | +5ret |
| 4 | +5ret |
| 5 | +12 V |
| 6 | −12 V |
| 7 | ±12ret |

CN2

| | |
|---|---|
| No.1 | +24 V |
| 2 | +24V |
| 3 | +24 V |
| 4 | +24 V |
| 5 | +24ret |
| 6 | +24ret |
| 7 | +24ret |
| 8 | +24ret |

## ●CN3

RS-232-Cとのインタフェース・コネクタです。

| | |
|---|---|
| No.1 | RS-232-C用TxD信号 |
| 2 | RS-232-C用RxD信号 |
| 3 | GND |
| 4 | —— |
| 5 | —— |
| 6 | —— |
| 7 | —— |

## ●CN4

カバーに取り付けられたリミット・スイッチに接続されており，カバーが開くとショートします。

| | |
|---|---|
| No.1 | カバー・オープン・センサ信号 |
| 2 | GND |

## ●CN5

密着センサとのインタフェース・コネクタです。

| | |
|---|---|
| No.1 | 密着センサ・データ信号 |
| 2 | GND |
| 3 | +5 V |
| 4 | −12 V |
| 5 | GND |
| 6 | SIパルス信号 |
| 7 | GND |
| 8 | データCLK信号 |
| 9 | GND |
| 10 | 密着センサ電源供給信号 |

## ●CN6

音響カプラとのインタフェース・コネクタです。

| | |
|---|---|
| No.1 | 音響カプラ用RxD信号 |
| 2 | 音響カプラ用TxD信号 |
| 3 | —— |
| 4 | ±12ret |
| 5 | —— |
| 6 | 音響カプラ検知信号 |

●CN7

サーマル・プリント・ヘッドのコントロール信号用コネクタです。

| | |
|---|---|
| No.1 | サーマル・ヘッド・データ信号 |
| 2 | GND |
| 3 | データCLK信号 |
| 4 | GND |
| 5 | サーマル・ヘッド・ラッチ信号 |
| 6 | GND |
| 7 | +5 V |
| 8 | GND |
| 9 | ―― |
| 10 | ストローブ信号 0 |
| 11 | ストローブ信号 1 |
| 12 | ストローブ信号 2 |
| 13 | ストローブ信号 3 |
| 14 | ストローブ信号 4 |
| 15 | ストローブ信号 5 |
| 16 | ストローブ信号 6 |
| 17 | ストローブ信号 7 |
| 18 | ストローブ信号 8 |
| 19 | GND |
| 20 | ストローブ・イネーブル信号 |

●CN8

サーマル・プリント・ヘッドへの印字用電源を供給するコネクタです。

| | |
|---|---|
| No.1 | +24 V |
| 2 | +24ret |
| 3 | +24 V |
| 4 | +24 V |
| 5 | +24ret |
| 6 | +24 V |

●CN9，14

ステッピング・モータの駆動用コネクタです。

| | |
|---|---|
| No.1 | モータ励磁信号（$\phi$0） |
| 2 | モータ励磁信号（$\phi$1） |
| 3 | モータ励磁信号（$\phi$2） |
| 4 | モータ励磁信号（$\phi$3） |
| 5 | +24 V |
| 6 | +24 V |

●CN10

インジケータ・パネル上のLEDランプの駆動用コネクタです。

| | |
|---|---|
| No.1 | エラーLED点灯用信号 |
| 2 | ペーパLED点灯用信号 |
| 3 | カバー・オープンLED点灯用信号 |
| 4 | +5 V |
| 5 | GND |

●CN11

スイッチ基板上の各スイッチ入力と，「スタート」，「ファイン」LEDランプの駆動用コネクタです。

| | |
|---|---|
| No.1 | スタート・キー信号 |
| 2 | ストップ・キー信号 |
| 3 | ファイン・キー信号 |
| 4 | スタートLED点灯用信号 |
| 5 | ファインLED点灯用信号 |
| 6 | +5 V |
| 7 | GND |

●CN12

原稿センサの有無を読み込むためのコネクタです。

| | |
|---|---|
| No.1 | 原稿センサ信号 |
| 2 | +5 V |
| 3 | GND |

●CN13

感熱記録紙の有無を読み込むためのコネクタです。

| | |
|---|---|
| No.1 | 感熱紙センサ信号 |
| 2 | +5 V |
| 3 | GND |

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<br>半導体第二販売事業部<br>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 半導体第一販売部<br>半導体第二販売部 | 〒460 名古屋市中区錦一丁目17番1号（NEC中部ビル） | 名古屋<br>名古屋 | (052)222-2170<br>(052)222-2190 |
| 関西支社 半導体第一販売部<br>半導体第二販売部<br>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<br>大阪<br>大阪 | (06) 945-3178<br>(06) 945-3200<br>(06) 945-3208 |

| | | | | | |
|---|---|---|---|---|---|
| 北海道支社 | 札幌 (011)251-5599 | 太田支店 | 太田 (0276)46-4011 | 福井支店 | 福井 (0776)22-1866 |
| 東北支社 | 仙台 (022)267-8740 | 宇都宮支店 | 宇都宮 (028)621-2281 | 富山支店 | 富山 (0764)31-8461 |
| 岩手支店 | 盛岡 (019)651-4344 | 小山支店 | 小山 (0285)24-5011 | 三重支店 | 津 (0592)25-7341 |
| 郡山支店 | 郡山 (0249)23-5511 | 長野支社 | 松本 (0263)35-1662 | 京都支社 | 京都 (075)344-7824 |
| いわき支店 | いわき (0246)21-5511 | 甲府支店 | 甲府 (0552)24-4141 | 神戸支社 | 神戸 (078)333-3854 |
| 長岡支店 | 長岡 (0258)36-2155 | 埼玉支社 | 大宮 (048)649-1415 | 中国支社 | 広島 (082)242-5504 |
| 土浦支店 | 土浦 (0298)23-6161 | 立川支社 | 立川 (0425)26-5981 | 鳥取支店 | 鳥取 (0857)27-5311 |
| 水戸支店 | 水戸 (029)226-1717 | 千葉支社 | 千葉 (043)238-8116 | 岡山支店 | 岡山 (086)225-4455 |
| 神奈川支社 | 横浜 (045)682-4524 | 静岡支社 | 静岡 (054)254-4794 | 松山支店 | 松山 (089)945-4149 |
| 群馬支店 | 高崎 (0273)26-1255 | 北陸支社 | 金沢 (076)232-7303 | 九州支社 | 福岡 (092)261-2806 |

**【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<br>マイクロコンピュータ技術部 | 〒210 川崎市幸区堀越三丁目484番地 | 川崎 | (044)548-7924 |
| 半導体販売技術本部<br>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)3798-9619 |
| 半導体販売技術本部<br>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<br>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 |

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)

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］ V55PI　アプリケーション・ノート　ファクシミリ編

（U13256JJ2V0AN00（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