お客様各位

カタログ等資料中の旧社名の扱いについて

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

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

2010年4月1日
ルネサスエレクトロニクス株式会社

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

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

ご注意書き

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

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

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

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

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

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

アプリケーション・ノート

RENESAS

78K/Ⅱシリーズ

8ビット・シングルチップ・マイクロコンピュータ

応用編

μPD78214シリーズ
μPD78218Aシリーズ
μPD78224シリーズ
μPD78234シリーズ
μPD78244シリーズ

資料番号　IEA−700C（第4版）
発行年月　April 1997　N

アプリケーション・ノート

NEC

78K/Ⅱシリーズ

8ビット・シングルチップ・マイクロコンピュータ

応用編

μPD78214シリーズ
μPD78218Aシリーズ
μPD78224シリーズ
μPD78234シリーズ
μPD78244シリーズ

© NEC Corporation 1991

本製品のうち，外国為替および外国貿易管理法の規定により戦略物資等（または役務）に該当するものについては，日本国外に輸出する際に，同法に基づき日本国政府の輸出許可が必要です。

○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。

M7 94.11

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。

本版で改訂された主な箇所

箇　所	内　容
全　般	μPD78P244に関する記述削除
p. 314, 317	プログラム・リスト変更

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，78K/IIシリーズ製品の機能を理解し，78K/IIシリーズ製品を用いたアプリケーション・プログラムを設計するユーザのエンジニアを対象とします。

78K/IIシリーズ製品

- μPD78214シリーズ　：μPD78212，78213，78214，78P214，78212(A)，78213(A)
  μPD78214(A)，78P214(A)
- μPD78218Aシリーズ：μPD78217A，78218A，78P218A，78218A(A)
- μPD78224シリーズ　：μPD78220，78224，78P224
- μPD78234シリーズ　：μPD78233，78234，78237，78238，78P238
  μPD78234(A)，78238(A)
- μPD78244シリーズ　：μPD78243，78244

目　的　このアプリケーション・ノートは，78K/IIシリーズ製品の応用プログラムについてユーザに理解していただくことを目的とします。

なお，掲載のプログラムおよびハードウエア構成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。

構　成　このアプリケーション・ノートでは，おもに次の内容について説明しています。

- タイマ／カウンタのプログラム例
- マクロ・サービスの応用例
- ウォッチドッグ・タイマ
- 表示／キー入力
- SBIの応用

なお，次の２つのアプリケーション・ノートも別に用意しています。

- 基礎編（IEA-607）
- 浮動小数点演算プログラム編（IEA-686）

読み方　このアプリケーション・ノートでは，特に断りがない場合は，78K/IIシリーズ全製品を対象としています。

違いがある場合は，インデクスで区別ができるようになっています。

●インデクスの見方

 ：そのまま使用できます。

 ：簡単な変更で使用できます。

な　　し：このままでは使用できません。

なお，詳細は本文を参照してください。

●一通り使い方の例を見たい場合

→目次に従って読んでください。

●命令の使い方の例を見たい場合

→目次に従って読んでください。

●内蔵ハードウエアの使い方の例を見たい場合

→一通り見たい場合は順番に，特定のものは**目次**または**付録D**を利用してください。

●特定の製品の応用例を見たい場合

→**付録D**を利用してください。

●応用方法が分かっていて例を見たい場合

→**目次**と**付録D**を併用してください。

●構造化アセンブラでのプログラム例を見たい場合

→**付録D**を利用してください。

品質水準　μPD78212(A)，78213(A)，78214(A)，78P214(A)はμPD78212，78213，78214，78P214の，μPD78218A(A)はμPD78218Aの，またμPD78234(A)，78238(A)はμPD78234，78238の『特別』品質水準品です。

このアプリケーション・ノート中の使用例は，一般電子機器用の『標準』品質水準品用に作成してあります。『特別』品質水準品を要求する用途にこのアプリケーション・ノート中の使用例を使用する場合は，実際に使用する各部品および回路について，その品質水準についてご検討のうえご使用ください。

品質水準とその応用分野の詳細については当社発行の資料「NEC 半導体デバイスの品質水準」(IEI-620)をご覧ください。

応用分野　○標準品　　○特別品

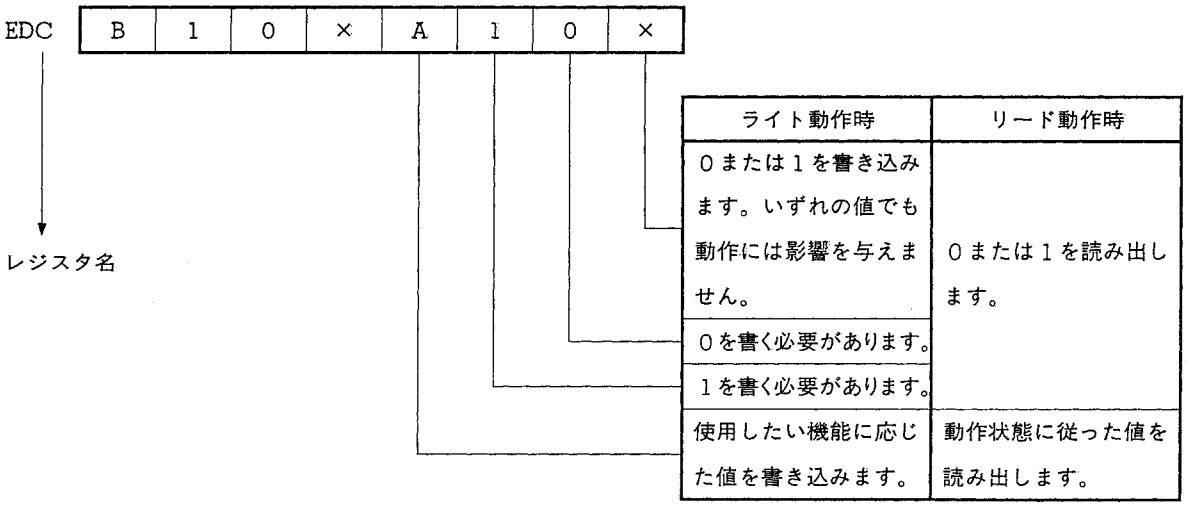
- ●プリンタ
- ●カメラ
- ●タイプライタ
- ●PPC
- ●電子楽器
- ●エアコンなど

- ●自動車電装
- ●燃料制御など

凡　例

データ表記の重み	：左側が上位桁，右側が下位桁
アクティブ・ロウの表記	：$\overline{\times\times\times}$（端子，信号名称の上に線）
注	：本文中につけた注の説明
注意	：特に気をつけていただきたい内容
備考	：本文の補足説明
数の表記	：２進数…××××Bまたは××××
	：10進数…××××
	：16進数…××××H

レジスタ表記



本文中のレジスタ表記に『設定禁止』と書いてあるコードの組み合わせは，絶対に書き込まないでください。

まぎらわしい文字　：0（ゼロ），O（オー）

：1（イチ），l（エル），I（アイ）

関連資料

○78K/IIシリーズ共通資料

資料名		資料番号
ユーザーズ・マニュアル　命令編		IEU-754
SBI　ユーザーズ・マニュアル		IEM-5040
アプリケーション・ノート	基礎編	IEA-607
	応用編	このアプリケーション・ノート
	浮動小数点演算プログラム編	IEA-686
セレクション・ガイド		IF-304
開発ツール　セレクション・ガイド		EF-231
インストラクション活用表		IEM-5101
インストラクション・セット		IEM-5102

○個別資料

●μPD78214シリーズ

品名 / 資料名	μPD78212	μPD78213	μPD78214	μPD78P214
パンフレット	IB-5036			
データ・シート	IC-8149	IC-7649		IC-7732
ユーザーズ・マニュアル　ハードウエア編	IEM-5119			
モード・レジスタ活用表	IEM-5100			

品名 / 資料名	μPD78212(A)	μPD78213(A)	μPD78214(A)	μPD78P214(A)
データ・シート	IC-8147	IC-8234		IC-8589
ユーザーズ・マニュアル　ハードウエア編	IEM-5119			
モード・レジスタ活用表	IEM-5100			

● μPD78218Aシリーズ

資料名＼品名	μPD78217A	μPD78218A	μPD78P218A	μPD78218A(A)
パンフレット	IF-288			
データ・シート	IC-8132	IC-8131	IC-8133	IC-8685
ユーザーズ・マニュアル　ハードウエア編	IEU-755			
特殊機能レジスタ活用表	IEM-5532			

● μPD78224シリーズ

資料名＼品名	μPD78220	μPD78224	μPD78P224
パンフレット	IB-5011		
データ・シート	IC-5457		IC-7757
ユーザーズ・マニュアル　ハードウエア編	IEU-5019		
特殊機能レジスタ活用表	IEM-999		

● μPD78234シリーズ

資料名＼品名	μPD78233	μPD78234	μPD78237	μPD78238	μPD78P238
パンフレット	IF-207				
データ・シート	IC-7902		IC-8348	IC-8028	IC-8030
ユーザーズ・マニュアル　ハードウエア編	IEU-718				
特殊機能レジスタ活用表	IEM-5515				

資料名＼品名	μPD78234(A)	μPD78238(A)
パンフレット	—	
データ・シート	IC-8146	IC-8727
ユーザーズ・マニュアル　ハードウエア編	IEU-718	
特殊機能レジスタ活用表	—	

●μPD78244シリーズ

品名 / 資料名	μPD78243	μPD78244
パンフレット	IF-214	
データ・シート	IC-8355	IC-8070
ユーザーズ・マニュアル　ハードウエア編	IEU-747	
特殊機能レジスタ活用表	IEM-5528	

概　説 1

タイマ／カウンタのプログラム応用例 2

マクロ・サービスの応用例 3

ウォッチドッグ・タイマ 4

リアルタイム出力ポートによるステッピング・モータの制御 5

リアルタイム出力ポートによるインバータの制御 6

表示／キー入力 7

アナログ・キー入力 8

内蔵ROMチェック 9

SBIの応用 10

ホームバス・システムへの応用 11

付　録 付

μPD78214シリーズ応用例 214

μPD78218Aシリーズ応用例 218A

μPD78224シリーズ応用例 224

μPD78234シリーズ応用例 234

μPD78244シリーズ応用例 244

目　　次

第1章　概　　説 … 1

1.1　78K/IIシリーズ製品の特徴 … 1
1.2　アプリケーション・プログラムの利用方法 … 3

第2章　タイマ／カウンタのプログラム応用例 … 5

2.1　マクロ・サービスによるフリー・ランニング・インターバル・タイマ … 6
2.2　高分解能PWM出力 … 34
2.3　高分解能キャプチャ機能 … 55
2.3.1　動作概要 … 55
2.3.2　プログラム説明 … 57
2.4　積分型A/Dコンバータ … 72
2.5　DTMF信号の生成 … 89
2.5.1　動作概要 … 90
2.5.2　ベクタ割り込みを使用したプログラム例 … 92
2.5.3　マクロ・サービスを使用したプログラム例 … 114

第3章　マクロ・サービスの応用例 … 139

3.1　プログラマブル・イベント・カウンタ … 140
3.2　パルス周期の測定 … 150
3.2.1　16ビット・タイマ／カウンタを使用したプログラム例 … 150
3.2.2　8ビット・タイマ／カウンタ1を使用したプログラム例 … 165
3.3　アナログ波形出力 … 180
3.4　階段波形出力によるA/D変換 … 195
3.4.1　ベクタ割り込みを使用したプログラム例 … 196
3.4.2　マクロ・サービスを使用したプログラム例 … 215
3.5　リアルタイム出力ポートによるPWM出力 … 237

第4章　ウォッチドッグ・タイマ … 253

4.1　外付けタイマによるウォッチドッグ・タイマ … 253
4.2　内部タイマによるウォッチドッグ・タイマ … 260

第5章　リアルタイム出力ポートによるステッピング・モータの制御 … 271

5.1　高分解能タイミングによる制御 … 271
5.2　2軸同時制御 … 285

第6章　リアルタイム出力ポートによるインバータの制御 … 321

6.1　概　　要 … 321
6.2　機能概要 … 322
6.2.1　実現手段 … 322
6.2.2　PWM信号の生成方法 … 323
6.2.3　インバータの出力周波数 … 324
6.2.4　マクロ・サービス・タイプCの利用方法 … 325
6.2.5　周波数の移行方法 … 328
6.3　ハードウエア構成 … 330
6.3.1　システム構成 … 330
6.3.2　CPUブロック … 331
6.3.3　インバータ・ブロック … 332
6.4　制御方法（データ編） … 333
6.4.1　概　　要 … 333
6.4.2　PWMデータの構成 … 334
6.5　制御方法（プログラム編） … 338
6.5.1　状態遷移図，状態遷移表 … 338
6.5.2　サブルーチン構成 … 340
6.5.3　ワークRAM，フラグ … 341
6.5.4　コントロール・エリア構成 … 343
6.5.5　プログラム説明 … 344
6.6　出力波形例 … 372
6.6.1　ターゲット回路 … 372
6.6.2　リアルタイム出力ポート波形 … 373
6.6.3　各相のリアルタイム出力ポート波形 … 375
6.6.4　デッド・タイム波形 … 376
6.6.5　無負荷時の相間波形 … 376
6.6.6　負荷接続時の相間波形 … 377
6.7　プログラム・リスト … 378
6.7.1　インバータ制御プログラム … 378
6.7.2　PWMデータ … 392
6.8　PWMデータ作成時の留意点 … 398
6.8.1　スイッチング・タイミング … 398
6.8.2　デッド・タイムの加味 … 400

第7章　表示／キー入力 … 403

7.1　ベクタ割り込みによるプログラム例 … 409
7.2　マクロ・サービスによるプログラム例 … 437

第8章　アナログ・キー入力 … 475

8.1　A/Dコンバータによるアナログ・キー入力 … 475
8.2　アナログ・キー入力 … 484

第９章　内蔵ROMチェック … 491

9.1　チェック・サム … 492
9.2　外部出力 … 496
9.3　内部比較 … 502

第10章　SBIの応用 … 509

10.1　機能概要 … 509
10.1.1　接 続 図 … 509
10.1.2　アドレス，コマンドの定義 … 510
10.1.3　構成概要 … 511
10.1.4　状態遷移図 … 511
10.2　応用プログラムの環境 … 514
10.2.1　ワークRAM … 514
10.2.2　使用フラグ … 515
10.2.3　シンボル定義 … 516
10.2.4　インターバル・タイマ割り込み … 519
10.3　基本通信手段 … 520
10.3.1　１バイト送受信 … 520
10.3.2　シリアル転送完了割り込み … 525
10.3.3　タイマ割り込み … 526
10.3.4　１バイト送受信の状態遷移図 … 527
10.4　動作説明 … 528
10.4.1　１バイトの送受信処理 … 534
10.4.2　INTCSI割り込み処理 … 539
10.4.3　$\overline{\text{ACK}}$チェック処理 … 552
10.4.4　マクロ・サービス・タイム・アウト処理 … 559
10.4.5　バックグランド処理 … 563
10.4.6　イニシャライズ処理 … 575
10.4.7　WRITEコマンド処理 … 586
10.4.8　READコマンド処理 … 589
10.4.9　LWRITEコマンド処理 … 592
10.4.10　LREADコマンド処理 … 599
10.4.11　DETACHコマンド処理 … 605
10.4.12　CHGMSTコマンド処理 … 609
10.4.13　MACWコマンド処理 … 615
10.4.14　MACRコマンド処理 … 624
10.4.15　DSPON/DSPOFFコマンド処理 … 634
10.5　SBI使用上の注意事項 … 636

第11章　ホームバス・システムへの応用 … 637

11.1　概　　要 … 637
11.2　通信プロトコル … 638
11.2.1　制御信号と制御方式 … 639
11.2.2　休止時間，休止期間 … 641

11.2.3　パケット優先　…　641
11.2.4　競合でのビット照合と衝突検出　…　643
11.2.5　同期回復　…　644
11.2.6　短電文割り込み　…　645
11.2.7　誤り検出　…　646
11.2.8　同　　報　…　647
11.2.9　制御コード　…　647
11.2.10　電文長コード　…　648
11.2.11　チェック・コード　…　648
11.2.12　ダ　ミ　ー　…　648
11.2.13　アドレス・コードの割り当て　…　649
11.2.14　ヘッダ・コード　…　650
11.2.15　システム共通コマンド　…　650
11.3　ハードウエア構成　…　653
11.3.1　システム構成　…　653
11.3.2　送受信回路　…　654
11.3.3　内部RAMの割り付け　…　655
11.4　動作概要　…　656
11.4.1　ホームバス・システムとしての動作　…　656
11.4.2　送受信の実現方法　…　656
11.4.3　送信波形データの形式　…　658
11.4.4　データ・バッファの構成　…　659
11.4.5　送信要求，短電文割り込み要求の発行　…　660
11.4.6　休止時間，休止期間のカウント方法　…　660
11.4.7　エラー検出　…　661
11.4.8　受信タイム・アウト　…　661
11.4.9　コマンド処理　…　662
11.5　制御プログラム　…　665
11.5.1　処理概要と状態遷移　…　665
11.5.2　モジュール構成　…　668
11.5.3　ワークRAM，ワーク・フラグ　…　669
11.5.4　プログラム説明　…　683
11.5.5　プログラム・リスト　…　756

付録A　78K/IIシリーズのプログラムの留意事項　…　809

A.1　ベクタ割り込み処理の留意事項　…　809
A.2　78K/IIシリーズの外部拡張データ・メモリのアクセス方法の留意事項　…　811
A.3　ポート0とリアルタイム出力ポートのアクセスの注意点　…　812

付録B　78K/IIシリーズ製品一覧　…　813

付録C　RA78K/II　…　821

C.1　RA78K/II添付ファイルについて　…　821
C.1.1　INTMS.DEF　…　822

付録D　プログラム，デバイス，内蔵周辺ハードウエアの対応表 … 823

図　の　目　次（1/6）

図番号	タイトル，ページ
1-1	周辺機能の関連図 … 2
1-2	クロック発生回路のブロック図 … 3
2-1	マクロ・サービスによるTO2，TO3端子からのタイマ出力タイミング … 6
2-2	マクロ・サービスによる８ビット・コンペア・レジスタ（CR20，CR21）へのデータ転送 … 7
2-3	マクロ・サービス用データ・テーブルのメモリ・マップ … 9
2-4	10ビット分解能PWM出力のタイミング … 34
2-5	マクロ・サービスによるCR20へのデータ転送 … 35
2-6	長時間パルスの周期測定のタイミング・チャート … 55
2-7	タイマ起動直後のタイミング・チャート … 57
2-8	キャプチャ動作とINTC20マクロ・サービスのタイミング … 58
2-9	積分型A/Dコンバータのブロック図 … 72
2-10	積分型A/Dコンバータのプログラムのタイミング・チャート … 74
2-11	オペアンプ２の出力波形 … 75
2-12	キー配置と周波数 … 89
2-13	信号送出時間とミニマム・ポーズ（ハイ・アクティブの場合） … 89
2-14	TO2，TO3タイマ出力の合成波形例 … 90
2-15	キーとキー・データの対応 … 106
3-1	INTP0端子入力とマクロ・サービス完了割り込みのタイミング・チャート … 140
3-2	プログラマブル・イベント・カウンタのマクロ・サービスのメモリ・マップと動作ブロック図 … 141
3-3	INTP3端子入力によるキャプチャ動作とベクタ割り込み処理のタイミング … 151
3-4	TM0によるパルス周期測定のマクロ・サービスのメモリ・マップと動作ブロック図 … 152
3-5	INTP3入力と割り込み処理のタイミング例 … 153
3-6	INTP4端子入力によるTM1値の転送タイミング … 165
3-7	TM1によるパルス周期測定のマクロ・サービスのメモリ・マップと動作ブロック図 … 166
3-8	アナログ波形出力のハードウエア構成図 … 180
3-9	ポートへの設定値とD/Aコンバータ出力電圧の関係 … 180
3-10	出力サイン波とデータの関係 … 181
3-11	ポートへのデータ転送タイミング・チャート … 181
3-12	アナログ波形出力のマクロ・サービスのメモリ・マップと動作ブロック図 … 182
3-13	階段波形出力によるA/D変換のハードウエア構成図 … 195

図　の　目　次（2/6）

図番号	タイトル，ページ
3-14	ベクタ割り込みによるA/D変換のD/Aコンバータ出力とコンパレータ出力のタイミング … 196
3-15	ベクタ割り込みによるA/D変換のポートへのデータ転送タイミング・チャート … 197
3-16	ベクタ割り込みによるA/D変換のマクロ・サービスのメモリ・マップと動作ブロック図 … 198
3-17	マクロ・サービスによるA/D変換のD/Aコンバータ出力とコンパレータ出力のタイミング … 215
3-18	マクロ・サービスによるA/D変換のポートへのデータ転送タイミング・チャート … 216
3-19	マクロ・サービスによるA/D変換のINTP0マクロ・サービスのメモリ・マップと動作ブロック図 … 217
3-20	リアルタイム出力ポートによるPWM出力のタイミング・チャート … 237
3-21	出力タイミング・データと出力データのメモリ・マップ … 238
3-22	PWM波形の切り替わりタイミング … 238
4-1	ウォッチドッグ回路の接続ブロック図 … 253
4-2	μPD74HC123Aとの接続のタイミング・チャート … 254
4-3	内部タイマによるウォッチドッグ・タイマのブロック図 … 260
4-4	内部タイマによるウォッチドッグ・タイマのタイミング・チャート … 260
5-1	９ビット分解能のタイミング制御のタイミング・チャート … 272
5-2	９ビット分解能のタイミング制御のメモリ・マップ … 273
5-3	ステッピング・モータ制御のハードウエア・ブロック図 … 285
5-4	ステッピング・モータの２軸同時制御のプロフィール … 286
5-5	フラグ説明 … 287
5-6	加速プロフィール … 288
5-7	マクロ・サービス・タイプCのタイミング・チャート … 290
5-8	初期起動時の動作 … 291
5-9	停止状態からのスタート時のステップ・レート … 292
5-10	モータ０正回転加速時のマクロ・サービスの使用状態 … 293
5-11	モータ０正回転定速度時のマクロ・サービスの使用状態 … 294
5-12	逆転時のマクロ・サービス・ポインタとリング・カウンタの変更 … 295
6-1	インバータ・ブロックの概略図 … 322
6-2	三角波による正弦波疑似PWM波形の生成 … 323

図　の　目　次（3/6）

図番号	タイトル，ページ
6-3	インバータ制御ブロック図 … 326
6-4	デッド・タイムの考慮 … 327
6-5	出力データ領域の使用方法 … 327
6-6	出力タイミング・データの切り替え … 328
6-7	変更要求の受け付け … 329
6-8	78K/IIインバータ制御システム構成図 … 330
6-9	メモリ・マップ … 331
6-10	インバータ回路構成図 … 332
6-11	208 Hz時の変調波形 … 333
6-12	PWMデータ例 … 336
6-13	データ配置テーブル例 … 337
6-14	状態遷移図 … 338
6-15	サブルーチン関連図 … 340
6-16	208 Hzの場合のコントロール・エリア … 343
6-17	CRC0レジスタの設定 … 348
6-18	割り込み関連のモード・レジスタの設定 … 349
6-19	208 Hz → 174 Hzの場合のデータ展開 … 363
6-20	174 Hz → 208 Hzの場合のデータ展開 … 363
6-21	コントロール・データの更新タイミング … 364
6-22	周波数移行時のデータの切り替え（低周波側へ移行中） … 367
6-23	ターゲット回路図 … 372
6-24	リアルタイム出力ポート波形 … 373, 374
6-25	各相のリアルタイム出力ポート波形 … 375
6-26	デッド・タイム波形 … 376
6-27	無負荷時の相間波形 … 376
6-28	負荷波形 … 377
6-29	ゼロクロス後のスイッチング … 399
6-30	ゼロクロス前のスイッチング … 399
6-31	マクロ・サービスのタイミング … 400
7-1	キー・コードとキーの対応 … 403
7-2	8セグメントLED 4桁と16（4×4）キー・マトリクス … 404
7-3	表示のタイミング … 405
7-4	キー入力の様子（ロウ・アクティブの場合） … 407

図　の　目　次（4/6）

図番号	タイトル，ページ
7-5	キー入力信号のチャタリングの様子（ロウ・アクティブの場合） … 407
7-6	ソフトウエア上の有効キー（ロウ・アクティブの場合） … 408
7-7	表示とキー入力のタイミング … 409
7-8	INTP0発生回路とタイミング … 437
7-9	表示とキー入力のタイミング（マクロ・サービス） … 438
7-10	マクロ・サービスによるセグメント／スキャン信号出力（INTC10） … 442
7-11	マクロ・サービスによるディジット信号出力（INTC11） … 443
7-12	マクロ・サービスによるキー・リターン信号入力（INTP0） … 444
8-1	アナログ・キーの回路例 … 475
8-2	キーと入力電圧の関係 … 476
8-3	４つのキーに対する電圧 … 484
8-4	キー入力で用いるワーク・エリア … 484
8-5	アナログ・キーの回路例 … 485
9-1	チェック・サム・プログラムのメモリ・マップ … 492
9-2	内蔵ROMデータの外部出力動作ブロック図 … 496
9-3	ポート０への出力タイミング … 496
9-4	内蔵ROMデータの内部比較の動作ブロック図 … 502
10-1	SBI接続図 … 509
10-2	マスタの状態遷移図 … 512
10-3	スレーブの状態遷移図 … 513
10-4	バス・チェック … 521
10-5	アドレス送信 … 521
10-6	コマンド送信 … 521
10-7	データ送信 … 522
10-8	送信終了待ち … 522
10-9	データ受信 … 523
10-10	受信終了待ち … 523
10-11	１バイト送受信の状態遷移図 … 527
10-12	モジュール関連図 … 530，531
10-13	マクロ・サービス・タイム・アウト処理 … 559
10-14	LWRITEコマンドによる転送データ … 592

図　の　目　次（5/6）

図番号	タイトル，ページ
10-15	ウエイク・アップ設定（WUP←1）タイミングの注意事項 … 606
10-16	MACWコマンドのメモリ構成（マスタ） … 616
10-17	MACWコマンドのメモリ構成（スレーブ） … 617
10-18	MACRコマンドのメモリ構成（マスタ） … 625
10-19	MACRコマンドのメモリ構成（スレーブ） … 626
11-1	制御信号の基本構成 … 639
11-2	制御信号の基本構成 … 639
11-3	制御信号のフォーマット … 640
11-4	休止時間と休止期間のタイミング … 641
11-5	優先コードのビット割り付け … 641
11-6	ビット・サンプル点 … 643
11-7	同期回復のタイミング … 644
11-8	ブレーク信号 … 645
11-9	同報時の相手アドレスのビット割り付け … 647
11-10	制御コードのビット割り付け … 647
11-11	ヘッダ・コードのビット割り付け … 650
11-12	システム構成図 … 653
11-13	送受信回路 … 654
11-14	内部RAMの割り付け … 655
11-15	本プログラム例のホームバス・システム … 656
11-16	送受信のタイミング・チャート … 657
11-17	競合送信のタイミング・チャート … 658
11-18	1ビットの構成 … 658
11-19	送信データと送信波形データの対応 … 659
11-20	送受信バッファの名称と構成 … 659
11-21	受信タイム・アウトの判定タイミング … 662
11-22	異なるアドレス・グループのシーケンス … 664
11-23	同アドレス・グループのシーケンス … 664
11-24	IFUが先に登録要求を出す場合のシーケンス … 664
11-25	状態遷移図 … 666
11-26	概要フロー・チャート … 667
11-27	モジュール，サブルーチン構成図 … 668
11-28	フロー・チャート例 … 683

図　の　目　次（6/6）

図番号	タイトル，ページ
11-29	送信開始処理の流れ　…　686
11-30	タイマ1起動から波形出力までのタイミング　…　687
11-31	エラー・フラグ・エリアのフォーマット（受信）　…　718
11-32	エラー・フラグ・エリアのフォーマット（送信）　…　719
A-1	プログラム・ステータス・ワード　…　809
A-2	ベクタ割り込みとPSWの関係　…　810

表 の 目 次 (1/5)

表番号	タイトル，ページ
2-1	高分解能PWM出力のプログラムで用いるワーク・エリア … 36
2-2	長時間パルスの周期測定で使用するRAM一覧 … 59
2-3	長時間パルスの周期測定の出力パラメータ … 59
2-4	長時間パルスの周期測定の使用レジスタ … 60
2-5	積分型A/Dコンバータのプログラムで使用する内部ハードウエア一覧 … 73
2-6	積分型A/Dコンバータのプログラム例の使用RAM一覧 … 79
2-7	積分型A/Dコンバータのプログラム例の出力パラメータ … 79
2-8	積分型A/Dコンバータのプログラム例の使用レジスタ … 79
2-9	規定周波数とCR20，CR21への設定値，出力回数の対応表 … 91
2-10	DTMF信号の生成（ベクタ割り込み）で使用するRAM一覧 … 93
2-11	DTMF信号の生成（ベクタ割り込み）の入力パラメータ … 93
2-12	DTMF信号の生成（ベクタ割り込み）の出力パラメータ … 93
2-13	DTMF信号の生成（ベクタ割り込み）の使用レジスタ … 94
2-14	DTMF信号の生成（マクロ・サービス）で使用するRAM一覧 … 115
2-15	DTMF信号の生成（マクロ・サービス）の入力パラメータ … 115
2-16	DTMF信号の生成（マクロ・サービス）の出力パラメータ … 116
2-17	DTMF信号の生成（マクロ・サービス）の使用レジスタ … 116
3-1	プログラマブル・イベント・カウンタの使用RAM一覧 … 142
3-2	プログラマブル・イベント・カウンタの入力パラメータ … 142
3-3	プログラマブル・イベント・カウンタの使用レジスター覧 … 143
3-4	TM0によるパルス周期測定の使用RAM一覧 … 155
3-5	TM0によるパルス周期測定の出力パラメータ … 155
3-6	TM0によるパルス周期測定の使用レジスタ … 155
3-7	TM1によるパルス周期測定の使用RAM一覧 … 168
3-8	TM1によるパルス周期測定の出力パラメータ … 168
3-9	TM1によるパルス周期測定の使用レジスタ … 169
3-10	アナログ波形出力のモジュール構成一覧 … 183
3-11	アナログ波形出力の使用RAM一覧 … 184
3-12	アナログ波形出力の入力パラメータ … 184
3-13	アナログ波形出力の出力パラメータ … 184
3-14	ベクタ割り込みによるA/D変換のモジュール構成一覧 … 199
3-15	ベクタ割り込みによるA/D変換の使用RAM一覧 … 200
3-16	アナログ波形出力の入力パラメータ … 200

表 の 目 次 (2/5)

表番号	タイトル，ページ
3-17	ベクタ割り込みによるA/D変換の出力パラメータ … 201
3-18	マクロ・サービスによるA/D変換のモジュール構成一覧 … 218
3-19	マクロ・サービスによるA/D変換の使用RAM一覧 … 219
3-20	マクロ・サービスによるA/D変換の入力パラメータ … 220
3-21	マクロ・サービスによるA/D変換の出力パラメータ … 220
3-22	マクロ・サービスによるA/D変換の使用レジスター覧 … 220
3-23	リアルタイム出力ポートによるPWM出力で使用するRAM一覧 … 240
3-24	リアルタイム出力ポートによるPWM出力の入力パラメータ … 240
3-25	リアルタイム出力ポートによるPWM出力の使用レジスタ … 241
4-1	μPD74HC123Aの入力，出力の真理値表 … 253
5-1	リアルタイム出力ポートによるステッピング・モータの制御の使用RAM一覧 … 274
5-2	モータ別のパラメータ … 286
5-3	外部割り込み要求に対する処理 … 286
5-4	モータ０のステップ・レートのパラメータ … 288
5-5	モータ１のステップ・レートのパラメータ … 289
5-6	モータ０のタイミング・データ … 289
5-7	モータ１のタイミング・データ … 289
6-1	インバータ出力周波数 … 324
6-2	マクロ・サービス・タイプCの機能分担 … 325
6-3	ポート機能 … 332
6-4	PWMデータの仕様 … 334
6-5	状態遷移表 … 339
6-6	サブルーチン一覧 … 340
6-7	ワークRAM … 341
6-8	フ ラ グ … 341
6-9	ワークRAM，フラグ動作 … 342
6-10	外部呼び出しサブルーチン一覧 … 344
6-11	ワークRAM … 345
6-12	FATとインバータ出力周波数 … 345
6-13	フ ラ グ … 346
6-14	ワークRAM … 347

表 の 目 次 (3/5)

表番号	タイトル，ページ
6-15	フ ラ グ … 347
6-16	使用するタイマ割り込み … 347
6-17	INTC01のために設定するSFR … 347
6-18	INTC10，INTC11のイニシャライズ一覧 … 349
6-19	コンペア・レジスタの初期値 … 355
6-20	処理要求優先順位 … 360
6-21	周波数移行時の変動周波数と加減回数 … 362
6-22	増減動作 … 363
6-23	バッファ領域 … 366
7-1	表示およびキー入力の仕様 … 403
7-2	モジュール一覧 … 411
7-3	表示／キー入力共通ワーク・エリア … 413
7-4	表示関連ワーク・エリア … 414
7-5	キー入力関連ワーク・エリア … 414
7-6	入力パラメータ … 415
7-7	出力パラメータ … 415
7-8	使用レジスタ … 416
7-9	マクロ・サービスと各動作の対応 … 437
7-10	モジュール一覧 … 445
7-11	表示／キー入力関連ワーク・エリア１ … 447
7-12	表示／キー入力関連ワーク・エリア２ … 448
7-13	入力パラメータ … 448
7-14	出力パラメータ … 449
7-15	使用レジスタ … 449
8-1	A/Dコンバータによるアナログ・キー入力で使用するRAM一覧 … 477
8-2	A/Dコンバータによるアナログ・キー入力の出力パラメータ … 478
8-3	A/Dコンバータによるアナログ・キー入力の使用レジスタ … 478
8-4	各キーに対するキー・コード … 484
9-1	内蔵ROMサイズ一覧 … 491
9-2	内部比較プログラムで使用するワーク・エリア … 503

表 の 目 次 (4/5)

表番号	タイトル，ページ
10-1	コマンド … 510
10-2	ワークRAM … 514
10-3	使用フラグ … 515
10-4	マスタの送受信用フラグ … 520
10-5	スレーブの送受信用フラグ … 524
10-6	各モジュールの処理概要 … 528, 529
10-7	ワークの各モジュールでの使用状況 … 532, 533
10-8	1バイト送受信サブルーチン … 534
10-9	レーベルとプログラム説明開始番号 … 534
10-10	マスタ・フラグとINTCSI処理の関係 … 539
10-11	マスタ・フラグと$\overline{\text{ACK}}$チェック処理の関係 … 552
10-12	マスタ交換の可否 … 609
10-13	MACWコマンドのチャネル・ポインタとマクロ・サービス・カウンタ（マスタ） … 615
10-14	MACWコマンドのチャネル・ポインタとマクロ・サービス・カウンタ（スレーブ） … 616
10-15	MACRコマンドのチャネル・ポインタとマクロ・サービス・カウンタ（マスタ） … 624
10-16	MACRコマンドのチャネル・ポインタとマクロ・サービス・カウンタ（スレーブ） … 625
10-17	スタティックLEDの表示状態 … 634
11-1	ホームバス・システムのプロトコル項目 … 638
11-2	パケット・プライオリティ … 642
11-3	ACK/NAKとエラー … 646
11-4	短電文と長電文の区別 … 648
11-5	アドレス・コードの割り当て … 649
11-6	システム共通コマンド一覧 … 652
11-7	HBC用のコマンドとデータ・フォーマット … 663
11-8	IFU用のコマンドとデータ・フォーマット … 663
11-9	ワークRAM一覧 … 669
11-10	ワーク・フラグ一覧 … 670
11-11	ワークRAM説明 … 671
11-12	ワーク・フラグ説明 … 675
11-13	使用するタイマ割り込み … 689
11-14	使用するエッジ割り込み … 690
11-15	アドレスとアドレス・グループの対応 … 698
11-16	HBC側のシステム立ち上げモード番号とコマンド送受信シーケンスの対応 … 698

表　の　目　次（5/5）

表番号	タイトル，ページ
11－17	IFU側のシステム立ち上げモード番号とコマンド送受信シーケンスの対応　…　700
11－18	受信エラーとERROR値，参照テーブルの対応　…　719
11－19	送信エラーとERROR値，参照テーブルの対応　…　719

第1章　概　　説

1.1　78K/IIシリーズ製品の特徴

78K/IIシリーズは，78Kシリーズに属する8ビット・シングルチップ・マイクロコンピュータです。高性能8ビットCPU，ROM，RAMを搭載し，さらに各種の周辺ハードウエアをワンチップに集積しています。

78K/IIシリーズは，16ビット演算命令，乗除算命令，ビット操作命令などの豊富な命令セットを備えています。また，大容量データに対応するために，データ用のアドレス空間を1Mバイトに拡張できます。

OA機器などへの応用に適した周辺ハードウエアとして，タイマ／カウンタ，シリアル・インタフェース，アナログ入力およびハードウエア・データ転送処理を備えた高機能割り込みコントローラを内蔵しています。また，ステッピング・モータの制御などに有効なリアルタイム出力ポートも内蔵しています。

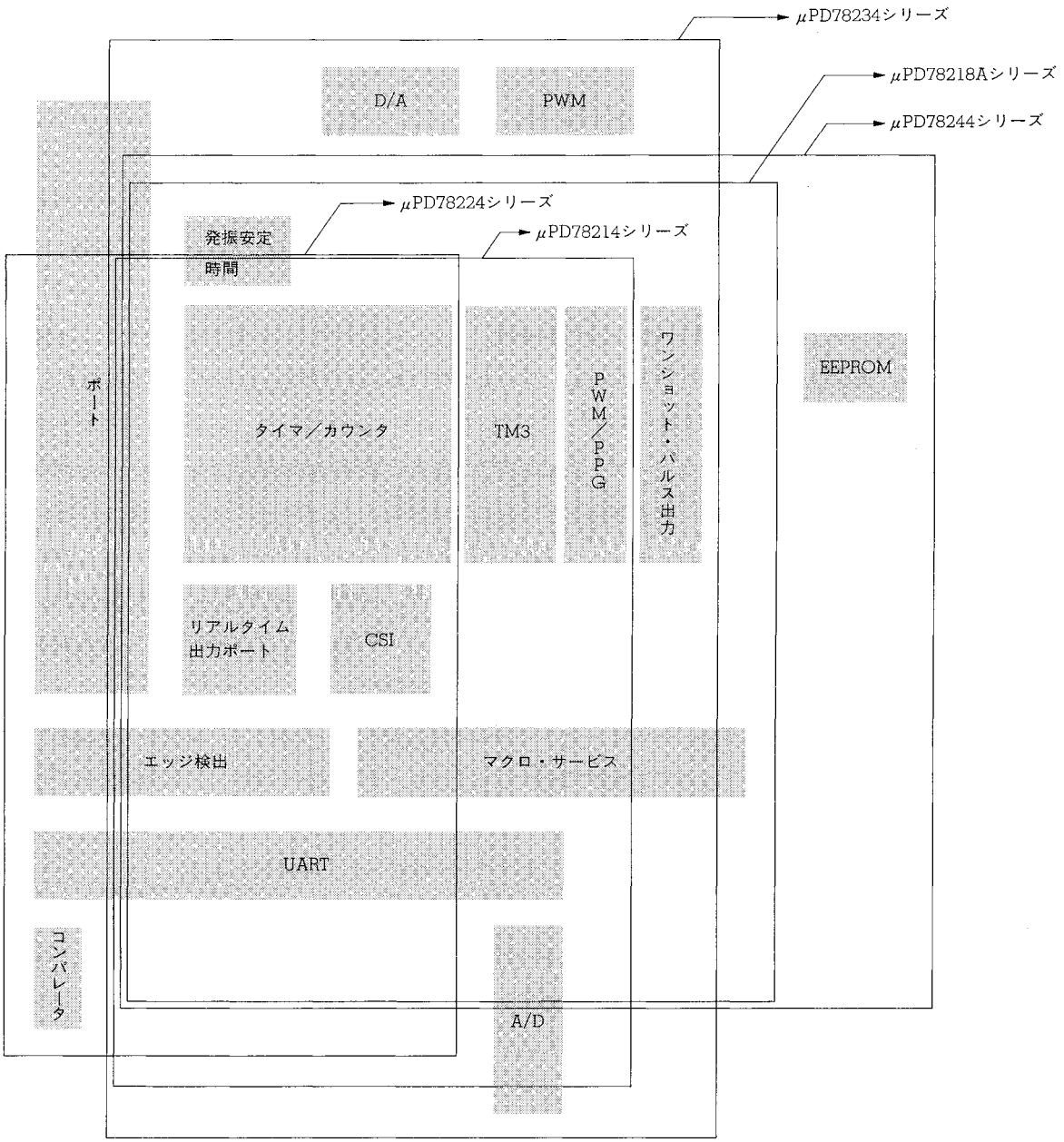
78K/IIシリーズとして，5つのシリーズ(μPD78214シリーズ，μPD78218Aシリーズ，μPD78224シリーズ，μPD78234シリーズ，μPD78244シリーズ)を用意しており，用途に応じて最適なシリーズを選択することができます。

各シリーズは，周辺ハードウエアが異なるだけでCPUは同じです。したがって，命令セットはすべて共通です。さらに，各周辺ハードウエアも基本機能は同一であり，各周辺ハードウエアを制御するプログラムもそのほとんどを共通に使用することができます（図1－1参照)。

また，シリーズ内の各製品については，内蔵するメモリのサイズだけが異なります。

なお，μPD78224シリーズでは，クロック同期式シリアル・インタフェースとタイマ／カウンタ関係のビット名称および機能について他のシリーズと一部違いがあります。このアプリケーション・ノートでは，他のシリーズのビット名称および機能で表現してあります。なお，付録Dでそのまま使用可となっているアプリケーション例の場合は，プログラムを変更せずにそのまま使用できます。

図 1－1　周辺機能の関連図



1.2 アプリケーション・プログラムの利用方法

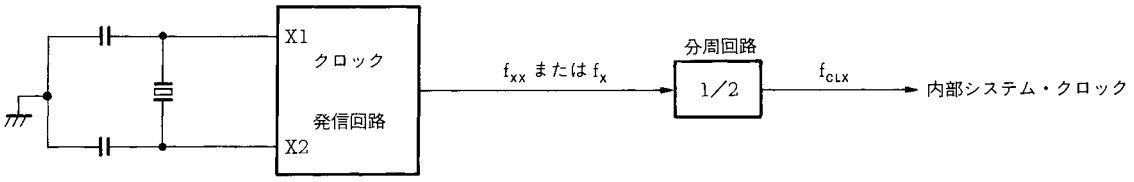
具体的な利用方法は，各アプリケーション・プログラムの解説にしたがってください。

このアプリケーション・ノートに記載されているアプリケーション・プログラムは，日本電気株式会社製のRA78K/II　アセンブラ・パッケージのバージョン　4.00以上でアセンブルできるように記述してあります。

アプリケーション・プログラムによっては，他のアプリケーション・プログラムを呼び出す構成となっているものがありますので，その場合は解説にしたがって，リンカによりリンクしてください。また，ワーク・エリアを必要とする場合がありますので，解説にしたがって領域の確保を行い，パブリック宣言を行ってください。

内蔵ハードウエアは，内部システム・クロックに動作が依存します。78K/IIシリーズ製品では，外部から供給されるクロック（f_{XX}，f_X）を2分周して，内部システム・クロック（f_{CLK}）としています。

図1-2　クロック発生回路のブロック図



備考　f_{XX} ：クリスタル／セラミック発振周波数

f_X ：外部クロック周波数

f_{CLK}：内部システム・クロック周波数（$=\frac{1}{2}f_{XX}$または$\frac{1}{2}f_X$）

以後，このアプリケーション・ノートでは内部システム・クロックをf_{CLK}と表現します。

第2章　タイマ／カウンタのプログラム応用例

タイマ／カウンタのプログラム応用例として，次の項目のプログラム例を示します。

1. マクロ・サービスによるフリー・ランニング・タイマ
2. 高分解能PWM出力
3. 高分解能キャプチャ機能
4. 積分型A/Dコンバータ
5. DTMF信号の生成

なお，これらの応用例は，一部を除いて，タイマ／カウンタだけではなく，マクロ・サービスの応用例ともなっています。

214
218A
234
244

2.1　マクロ・サービスによるフリー・ランニング・インターバル・タイマ

8ビット・タイマ／カウンタ2を用いたフリー・ランニング・インターバル・タイマ動作をマクロ・サービスにより行います。マクロ・サービスを2本同時に使用することによって，2つのコンペア・レジスタへのデータの転送を，割り込み要求発生ごとにベクタ割り込み処理で行う必要がなくなります。マクロ・サービスのタイプAとタイプBの2種を使用したプログラム例を示します。本文中の説明では，特に違いのある場合は(a) **マクロ・サービス・タイプA**，(b) **マクロ・サービス・タイプB**のように項目ごとに特有の説明になっています。

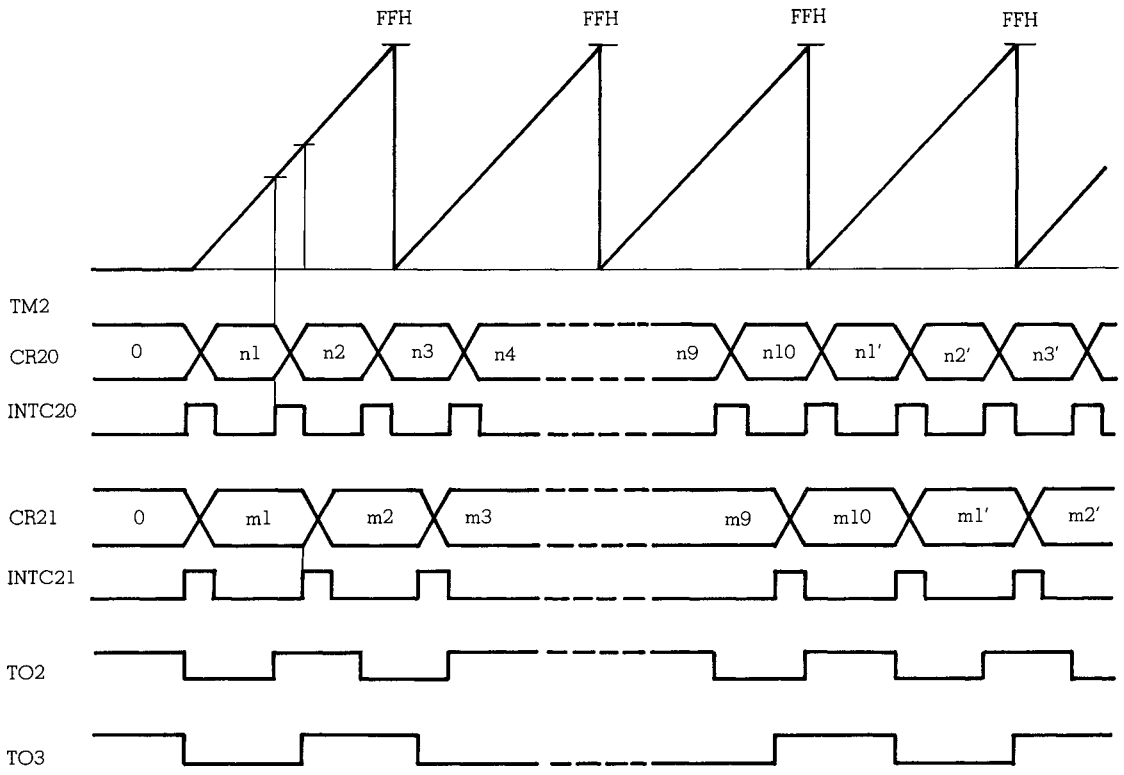
なお，この応用例は，μPD78224シリーズでは使用できません。

(1) 動作概要

INTC20，INTC21割り込み要求を使用して，マクロ・サービスによりTO2，TO3端子から2種類のタイマ出力を行うプログラム例を示します。TO2端子へのタイマ出力処理をマクロ・サービス・タイプAで，TO3端子へのタイマ出力処理をマクロ・サービス・タイプBで行います。

マクロ・サービスによるTO2，TO3端子からのタイマ出力のタイミングを図2－1に示します。

図2－1　マクロ・サービスによるTO2，TO3端子からのタイマ出力タイミング

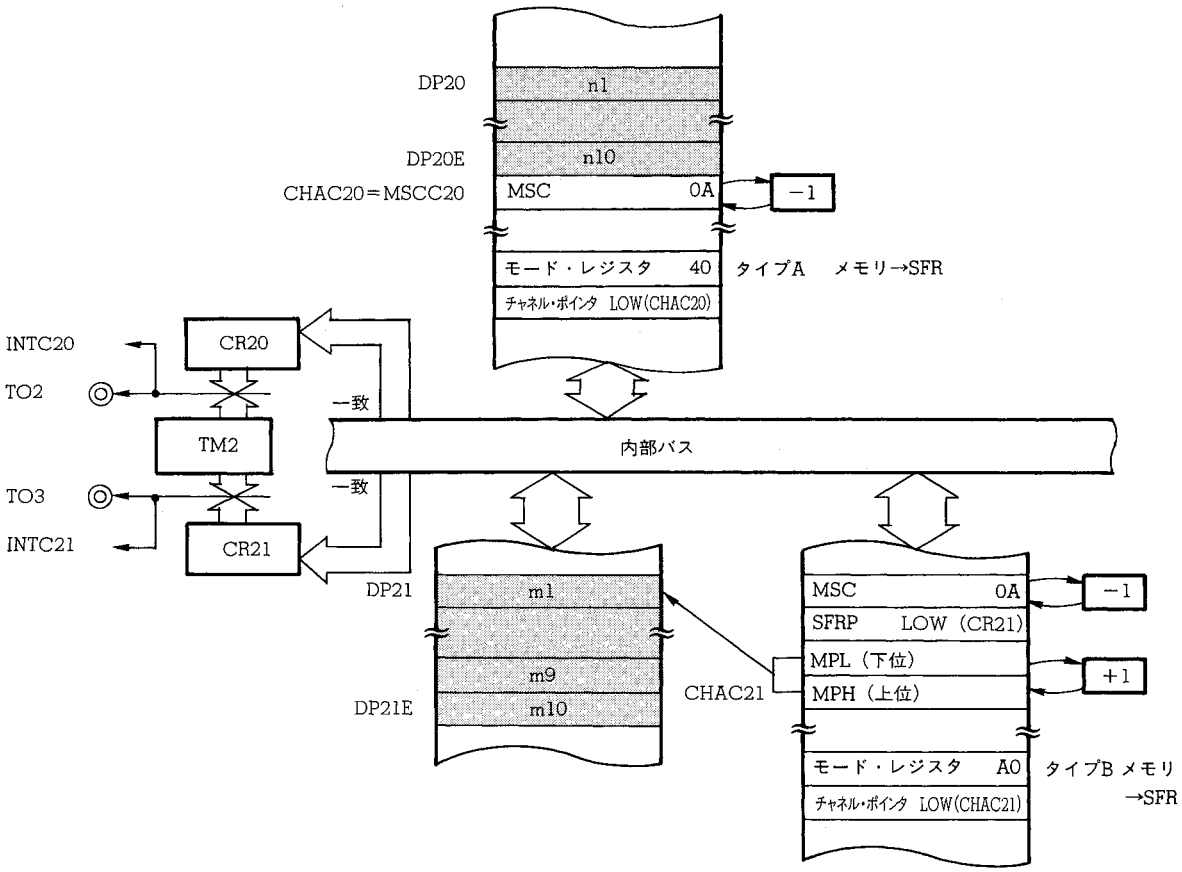


タイマをフリー・ランニングさせて一定周期のインターバルを作るには，タイマ割り込み要求発生時に，現在のコンペア・レジスタの値にインターバル周期分の値を加算します。これをマクロ・サービスで行うには，タイマ起動前にコンペア・レジスタへ転送する値をあらかじめ計算して，データ・テーブルとしてマクロ・サービス・バッファに用意しておきます。タイマ起動後，指定した回数だけマクロ・サービスが行われ，さらにマクロ・サービスが行われた回数分だけタイマ出力が反転します。

指定した回数のマクロ・サービスが行われた時点で，マクロ・サービス完了割り込みが発生します。この完了割り込み処理中で，次回のマクロ・サービスを行うためのデータ・テーブルを更新します。

次に，この応用例におけるマクロ・サービスのメモリ・マップ，および動作ブロック図を示します。

図2-2　マクロ・サービスによる8ビット・コンペア・レジスタ（CR20，CR21）へのデータ転送



2
214
218A
234
244

(2) プログラム説明（概要）

このプログラム例では，INTC20，INTC21マクロ・サービスとも処理概要は同様です。処理内容を大別すると，次のようになります。

(i) マクロ・サービス・イニシャライズ処理
(ii) マクロ・サービス完了割り込み（ベクタ割り込み）処理
(iii) マクロ・サービス用データ・テーブル作成処理

処理内容の詳細を次に説明します。

(i) マクロ・サービス・イニシャライズ処理

このプログラム例では，マクロ・サービス・イニシャライズ処理のうち，マクロ・サービス・カウンタ，割り込みサービス・モードの設定，および初期のテーブル・データの作成を，イニシャライズ処理中でINTC20，INTC21ベクタ割り込み要求を発生させて，マクロ・サービス完了割り込み処理を１回ずつ行うことによって行います。

したがって，このイニシャライズ処理を行う前にEI命令を実行して，割り込み許可状態にしておく必要があります。

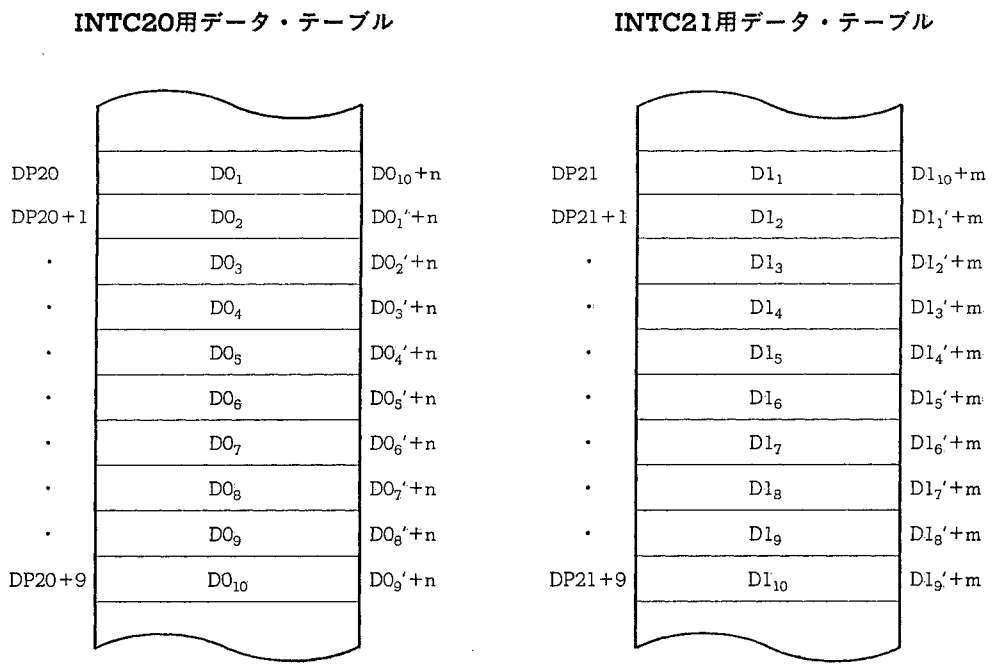
(ii) マクロ・サービス完了割り込み（ベクタ割り込み）処理

マクロ・サービス完了時の処理で，マクロ・サービスの再設定，データ・テーブルの作成を行います。このプログラム例では，この処理をマクロ・サービスのイニシャライズ処理の一部としても用いています。

(iii) マクロ・サービス用データ・テーブル作成処理

一連のマクロ・サービスで，コンペア・レジスタへのデータ転送を行う回数（マクロ・サービス・カウンタの値）分のデータ・テーブルをあらかじめ用意しておきます。このプログラム例では，INTC20，INTC21とも10回のマクロ・サービス終了時に完了割り込みが発生しますので，INTC20，INTC21ともテーブル・データは10個必要です。図２－３にそれぞれのデータ・テーブルのメモリ・マップを示します。

図2-3　マクロ・サービス用データ・テーブルのメモリ・マップ



n　：INTC20インターバル周期
DOn：前回作成した値
DOn'：新規に作成した値

m　：INTC21インターバル周期
DIn：前回作成した値
DIn'：新規に作成した値

(3) プログラム説明（詳細）

(a) マクロ・サービス・タイプA（INTC20）

(i) マクロ・サービス・イニシャライズ処理［レーベル名称：MINI20］

① INTC20マクロ・サービス・モード・レジスタの設定を行います。
　タイプA，メモリ→SFRに設定します。

② INTC20マクロ・サービス・チャネル・ポインタの設定を行います。

③ データ・テーブルの最後のメモリをイニシャライズ（0クリア）します。データ・テーブルのメモリ・マップについては，図2-3を参照してください。

④ INTC20割り込み要求のマスクを解除します。

⑤ INTC20割り込み要求フラグをセット(1)します。
　→INTC20ベクタ割り込み処理が行われます。

2
214
218A
234
244

注意　この応用例のように，割り込み要求フラグをセット後，続けて割り込み関係レジスタ（この応用例ではISMOレジスタ）を操作する命令を記述する場合，割り込み要求フラグのセットとレジスタ操作の間に，NOP命令２個分相当のウエイトを作り，割り込み関係レジスタを連続的に操作する前に，確実に割り込みを発生させてください。

⑥　INTC20割り込みサービス・モードがマクロ・サービスに設定されたことをチェックします。

⑦　INTC20割り込み要求フラグをセット(1)します。

→INTC20マクロ・サービス処理が１回行われます。データ・テーブルの先頭データがコンペア・レジスタ（CR20）へ転送され，最初のタイマ出力のパルス幅が決定します。

(ii) マクロ・サービス完了割り込み（ベクタ割り込み）処理［レーベル名称：INTC20］

①　レジスタ・バンク１に切り替えます。

②　INTC20マクロ・サービス・カウンタを設定します。

③　INTC20割り込みサービス・モードをマクロ・サービスに設定します。

④　INTC20マクロ・サービス用データ・テーブルを作成します。

(iii) マクロ・サービス用データ・テーブル作成処理［レーベル名称：TABLO］

①　INTC20用データ・テーブルの先頭アドレスをHLレジスタにセットします。HLレジスタをポインタとして使用します。

②　テーブル・データの個数（10個＝マクロ・サービス・カウンタ値）をBレジスタにセットします。Bレジスタをカウンタとして使用します。

③　INTC20用データ・テーブルの最後のメモリの値をAレジスタにセットします。

④　Aレジスタの値にインターバル周期分の値を加算します。加算値のオーバフローは無視します。

⑤　加算値を現在のポインタの示すメモリに格納し，その後ポインタ（HLレジスタ）をインクリメントします。

⑥　カウンタ（Bレジスタ）をデクリメントし，カウンタが０になるまで④，⑤の処理を繰り返します。

(b) マクロ・サービス・タイプB (INTC21)

(i) マクロ・サービス・イニシャライズ処理 [レーベル名称：MINI21]

① INTC21マクロ・サービス・モード・レジスタの設定を行います。
タイプB，メモリ→SFRに設定します。

② INTC21マクロ・サービス・チャネル・ポインタの設定を行います。

③ INTC21マクロ・サービスSFRポインタの設定を行います。

④ データ・テーブルの最後のメモリをイニシャライズ（0クリア）します。データ・テーブルのメモリ・マップについては，図2-3を参照してください。

⑤ INTC21割り込み要求のマスクを解除します。

⑥ INTC21割り込み要求フラグをセット(1)します。
→INTC21ベクタ割り込み処理が行われます。

注意　この例のように，割り込み要求フラグをセット後，続けて割り込み関係レジスタ（この応用例ではISMOレジスタ）を操作する命令を記述する場合，割り込み要求フラグのセットとレジスタ操作の間に，NOP命令2個分相当のウエイトを作り，割り込み関係レジスタを連続的に操作する前に，確実に割り込みを発生させてください。

⑦ INTC21割り込みサービス・モードがマクロ・サービスに設定されたことをチェックします。

⑧ INTC21割り込み要求フラグをセット(1)します。
→INTC21マクロ・サービス処理が1回行われます。データ・テーブルの先頭データがコンペア・レジスタ（CR21）へ転送され，最初のタイマ出力のパルス幅が決定します。

(ii) マクロ・サービス完了割り込み（ベクタ割り込み）処理 [レーベル名称：INTC21]

① レジスタ・バンク1に切り替えます。

② INTC21マクロ・サービス・カウンタを設定します。

③ INTC21割り込みサービス・モードをマクロ・サービスに設定します。

④ INTC21マクロ・サービス・ポインタを再設定します。

⑤ INTC21マクロ・サービス用データ・テーブルを作成します。

2
214
218A
234
244

(iii) マクロ・サービス用データ・テーブル作成処理［レーベル名称：TABL1］

① INTC21用データ・テーブルの先頭アドレスをHLレジスタにセットします。HLレジスタをポインタとして使用します。

② テーブル・データの個数（10個＝マクロ・サービス・カウンタ値）をBレジスタにセットします。Bレジスタをカウンタとして使用します。

③ INTC21用データ・テーブルの最後のメモリの値をAレジスタにセットします。

④ Aレジスタの値にインターバル周期分の値を加算します。加算値のオーバフローは無視します。

⑤ 加算値を現在のポインタの示すメモリに格納し，その後ポインタ（HLレジスタ）をインクリメントします。

⑥ カウンタ（Bレジスタ）をデクリメントし，カウンタが0になるまで④，⑤の処理を繰り返します。

(4) モード・レジスタ設定例

(a) マクロ・サービス・タイプA (INTC20)

割り込みマスク・レジスタH

	7	6	5	4	3	2	1	0	
MK0H	CSIMK	STMK	SRMK	SERMK	CMK20	PMK5	PMK4	CMK21	(リセットでFFH)
設定例	×	×	×	×	0	×	×	×	×：操作しません

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

割り込みサービス・モード・レジスタH

	7	6	5	4	3	2	1	0	
ISM0H	CSIISM	STISM	SRISM	0	CISM20	PISM5	PISM4	CSIM21	(リセットで00H)
設定例	×	×	×	×	1	×	×	×	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

2
214
218A
234
244

マクロ・サービス・モード・レジスタ

7	6	5	4	3	2	1	0
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例 0	1	0	0	0	0	0	0

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし データ転送のみ	POL
1	0	0	1				POH
1	0	1	0			リング制御なし 自動加算つき	POL
1	0	1	1				POH
1	1	0	0			リング制御つき データ転送のみ	POL
1	1	0	1				POH
1	1	1	0			リング制御つき 自動加算つき	POL
1	1	1	1				POH

(b) マクロ・サービス・タイプB (INTC21)

割り込みマスク・レジスタH

	7	6	5	4	3	2	1	0	
MK0H	CSIMK	STMK	SRMK	SERMK	CMK20	PMK5	PMK4	CMK21	(リセットでFFH)
設定例	×	×	×	×	×	×	×	0	×：操作しません

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

割り込みサービス・モード・レジスタH

	7	6	5	4	3	2	1	0	
ISM0H	CSIISM	STISM	SRISM	0	CISM20	PISM5	PISM4	CSIM21	(リセットで00H)
設定例	×	×	×	×	×	×	×	1	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

2
214
218A
234
244

マクロ・サービス・モード・レジスタ

7	6	5	4	3	2	1	0
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例 1	0	1	0	0	0	0	0

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	
MOD3 MOD2 MOD1 MOD0	タイプA	タイプB	タイプC	
			MPTL保持	MPTLインクリメント

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC		
0	0	0	0	メモリからSFRへデータ転送				
0	0	0	1	SFRからメモリへデータ転送				
1	0	0	0			リング制御なし	データ転送のみ	POL
1	0	0	1					POH
1	0	1	0				自動加算つき	POL
1	0	1	1					POH
1	1	0	0			リング制御つき	データ転送のみ	POL
1	1	0	1					POH
1	1	1	0				自動加算つき	POL
1	1	1	1					POH

（c）タイマ出力（共通）

タイマ出力コントロール・レジスタ

	7	6	5	4	3	2	1	0	
TOC	ENT03	ALV3	ENT02	ALV2	ENT01	ALV1	ENT00	ALV0	（リセットで00H）
設定例	1	0	1	0	0	0	0	0	

ALV0	TO0端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT00	TO0端子の動作指定
0	$\overline{\text{ALV0}}$を出力
1	パルス出力許可

ALV1	TO1端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT01	TO1端子の動作指定
0	$\overline{\text{ALV1}}$を出力
1	パルス出力許可

ALV2	TO2端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

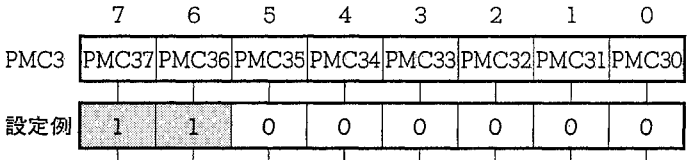
ENT02	TO2端子の動作指定
0	$\overline{\text{ALV2}}$を出力
1	パルス出力許可

ALV3	TO3端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT03	TO3端子の動作指定
0	$\overline{\text{ALV3}}$を出力
1	パルス出力許可

2
214
218A
234
244

ポート3モード・コントロール・レジスタ



PMC30	P30端子のコントロール・モード指定
0	入出力ポート・モード
1	RxD入力モード

PMC31	P31端子のコントロール・モード指定
0	入出力ポート・モード
1	TxD出力モード

PMC32	P32端子のコントロール・モード指定
0	入出力ポート・モード
1	$\overline{SCK}$入出力モード

PMC33	P33端子のコントロール・モード指定
0	入出力ポート・モード
1	SO出力／SB0入出力モード

PMC3n	PM3n端子のコントロール・モード指定（n=4-7）
0	入出力ポート・モード
1	TOm出力モード（m=0-3）

キャプチャ／コンペア・コントロール・レジスタ2

	7	6	5	4	3	2	1	0	
CRC2	MOD1	MOD0	CLR22	1[注]	CLR21	0	0	0	（リセットで10H）
設定例	0	0	0	1	0	0	0	0	

CLR21	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（CR21とTM2が一致したらクリア）

CLR22	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（キャプチャしたらクリア）

MOD1	MOD0	タイマ出力モード指定		
		TO2	TO3	その他条件
0	0	トグル出力	トグル出力	—
0	1	PWM出力	トグル出力	CLR21ビット=0
1	0	PWM出力	PWM出力	CLR22ビット=0
1	1	PPG出力	トグル出力	CLR21ビット=1 CLR22ビット=0

注　必ず“1”に設定してください。

2
214
218A
234
244

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	0	0	0	0	0	0	0	

8ビット・タイマ／カウンタ1

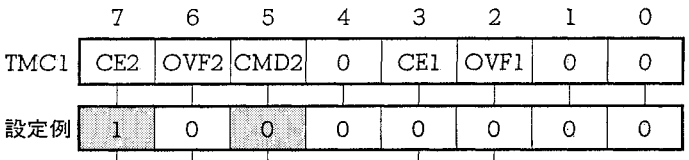
PRS12	PRS11	PRS10	タイマ／カウンタ1のカウント・クロック周波数の指定
0	0	0	$f_{CLK}/16$[注]
0	0	1	
0	1	0	
0	1	1	$f_{CLK}/32$
1	0	0	$f_{CLK}/64$
1	0	1	$f_{CLK}/128$
1	1	0	$f_{CLK}/256$
1	1	1	$f_{CLK}/512$

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	タイマ／カウンタ3のカウント・クロック周波数の指定
0	0	0	0	$f_{CLK}/16$
0	0	0	1	
0	0	1	0	
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	1	1	1	外部クロック（CI）

注　f_{CLK}：内部システム・クロック周波数（$f_{XX}/2$）

タイマ・コントロール・レジスタ1



	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	1	0	0	0	0	0	0	0	

8ビット・タイマ／カウンタ1

OVF1	タイマ／カウンタ1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE1	タイマ／カウンタ1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

CMD2	タイマ／カウンタ2の動作モード指定
0	通常モード
1	ワンショット・モード

OVF2	タイマ／カウンタ2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE2	タイマ／カウンタ2のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

2
214
218A
234
244

(5) 入力パラメータ

(a) マクロ・サービス・タイプA (INTC20)

(i) マクロ・サービス

MSCC20：マクロ・サービス・カウンタの配置アドレスです。

MSC20D：マクロ・サービス・カウンタへの設定値です。この設定回数分の転送が行われたのちにマクロ・サービス完了割り込みが発生します。

DP20　：データ・テーブルの先頭アドレスです。マクロ・サービスによる転送は，このポインタから始めます。

DP20E　：データ・テーブルの最後のアドレスです。

(ii) ベクタ割り込み

INTVL8　：TO2端子からのタイマ出力の周期を決定する値です。データ・テーブル作成の際に，直接のテーブル・データにこの値を加算していきます。

(b) マクロ・サービス・タイプB (INTC21)

(i) マクロ・サービス

MPC21P：マクロ・サービス・ポインタ（下位）の配置アドレスです。このプログラムでは，マクロ・サービス・ポインタの格納用メモリとしてMSP21とMSP21＋1の1ワードのメモリを一度に確保しています。

SFRC21　：SFRポインタの配置アドレスです。MSP21－1のアドレスに設定します。

MSCC21：マクロ・サービス・カウンタの配置アドレスです。

MSC21D：マクロ・サービス・カウンタへの設定値です。この設定回数分の転送が行われたのちにマクロ・サービス完了割り込みが発生します。

DP21　：データ・テーブルの先頭アドレスです。マクロ・サービスによる転送は，このポインタから始めます。

DP21E　：データ・テーブルの最後のアドレスです。

備考　MPC21P，SFRC21，MSCC21は，ベクタ・テーブルとマクロ・サービスの領域の定義ファイル(INTMS．DEF)中のマクロを参照することで定義しています。

(ii) ベクタ割り込み

INTVL9　：TO3端子からのタイマ出力の周期を決定する値です。データ・テーブル作成の際に，直前のテーブル・データにこの値を加算していきます。

(6) 使用レジスタ

(a) マクロ・サービス・タイプA (INTC20)

A, B, HL (レジスタ・バンク1)

(b) マクロ・サービス・タイプB (INTC21)

A, B, HL (レジスタ・バンク1)

(7) プログラム使用例

TO2端子から周波数3125 Hz, TO3端子から周波数2083 Hzのタイマ出力を行う例を示します。この場合，INTC20を160 μsごとに，INTC21を240 μsごとにそれぞれ発生させます。f_{CLK}=6 MHzとし，カウント・クロックを$f_{CLK}/16$に設定すると，入力パラメータINTVL8, INTVL9の値は，次のようになります。

$$INTVL8 = \frac{160\times10^{-6}}{16/(6\times10^{6})} = 60$$

$$INTVL9 = \frac{240\times10^{-6}}{16/(6\times10^{6})} = 90$$

プログラム例として次のように用います。

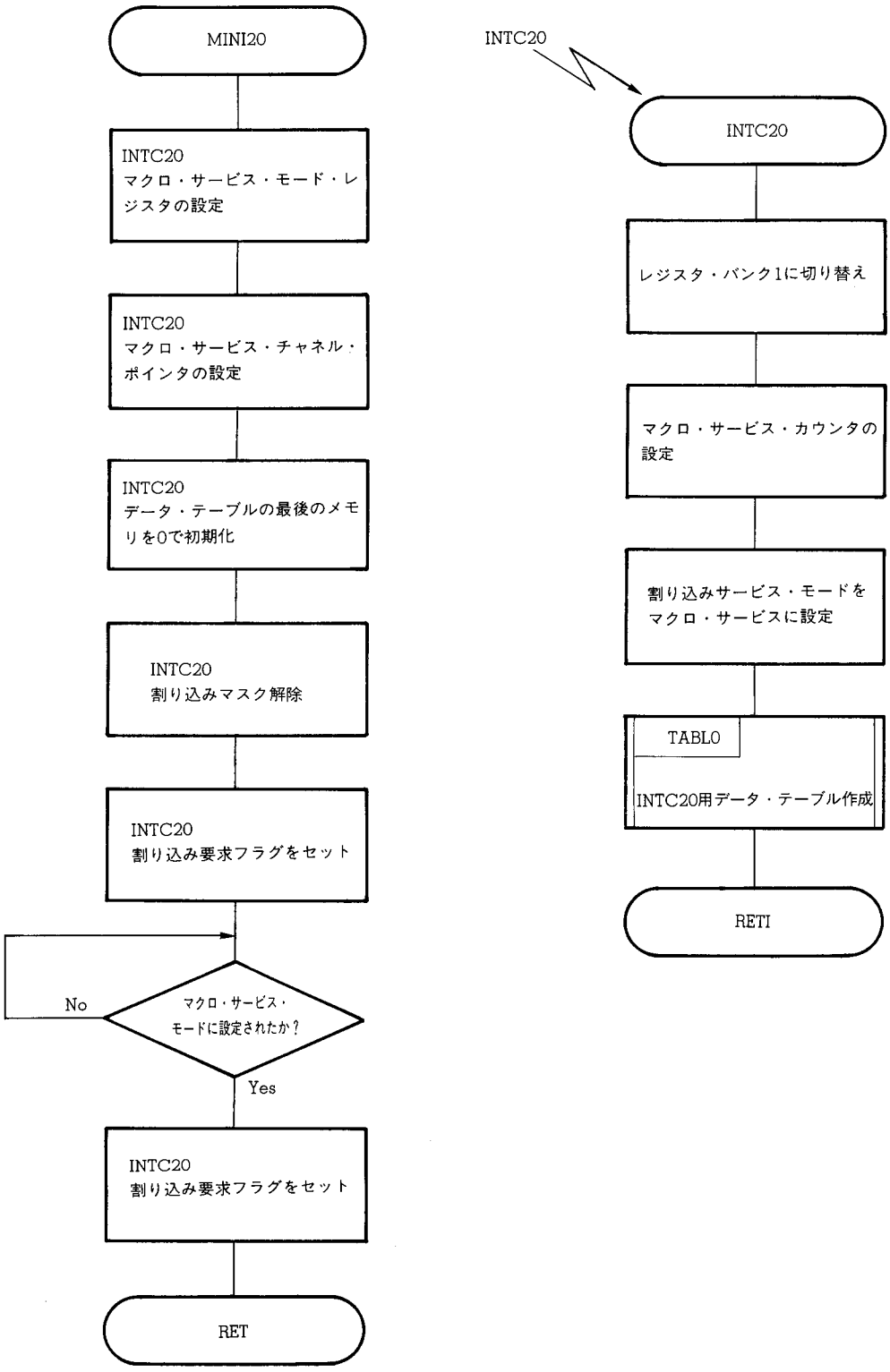
```
        PUBLIC  INTVL8, INTVL9
        PUBLIC  DP20, MSC20D, DP20E, MSCC20
        PUBLIC  DP21, MSC21D, DP21E

        EXTRN   MINI20, MINI21

INTVL8  EQU     60              ; M_free running timer INTC20 compare data
INTVL9  EQU     90              ; M_free running timer INTC21 compare data

;       #####   define data of INTC20 macro service (type A)   #####

MSC20D  EQU     10              ; macro service counter data

FR20_D  DSEG    SADDR
DP20:   DS      MSC20D          ; macro service buffer
MSCC20: DS      1               ; macro service counter
DP20E   EQU     MSCC20-1        ; macro service end of buffer

;       #####   define data of INTC21 macro service (type B)   #####

MSC21D  EQU     10              ; macro service counter data

FR21_D  DSEG    SADDR
DP21:   DS      MSC21D-1        ; macro service buffer
DP21E:  DS      1               ; macro service buffer (end of buffer)
;
;*************************************************************
;*      MAIN ROUTNE                                          *
;*              macro service initialize                     *
;               timer output initialize                      *
;*************************************************************
;
        CSEG
FRUN_M:
            :
        EI
        CALL    !MINI20         ; initialize of INTC20 macro service
        CALL    !MINI21         ; initialize of INTC21 macro service
        CALL    !TOUT           ; timer output
            :
;*************************************************************
;*              timer output initialize                      *
;*                      from TO2, TO3                        *
;*************************************************************
;
TOUT:
        MOV     TOC, #10100000B  ; timer output, active level high
        MOV     PMC3, #11000000B ; P3 control port
        MOV     CRC2, #00010000B ; set timer free running mode
        MOV     PRM1, #00000000B ; prescaler fclk/16
        MOV     TMC1, #10000000B ; timer count start
        RET
            :
```

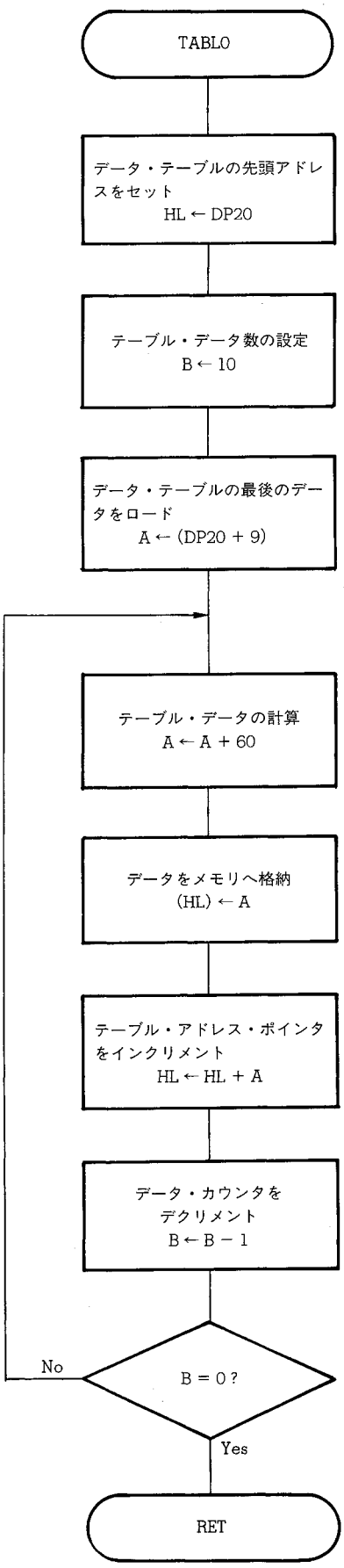
注意　ユーザ・プログラム中でINTC20，INTC21以外の割り込みを使用している場合は，EI命令実行のタイミングに注意する必要があります。

〔メ　モ〕

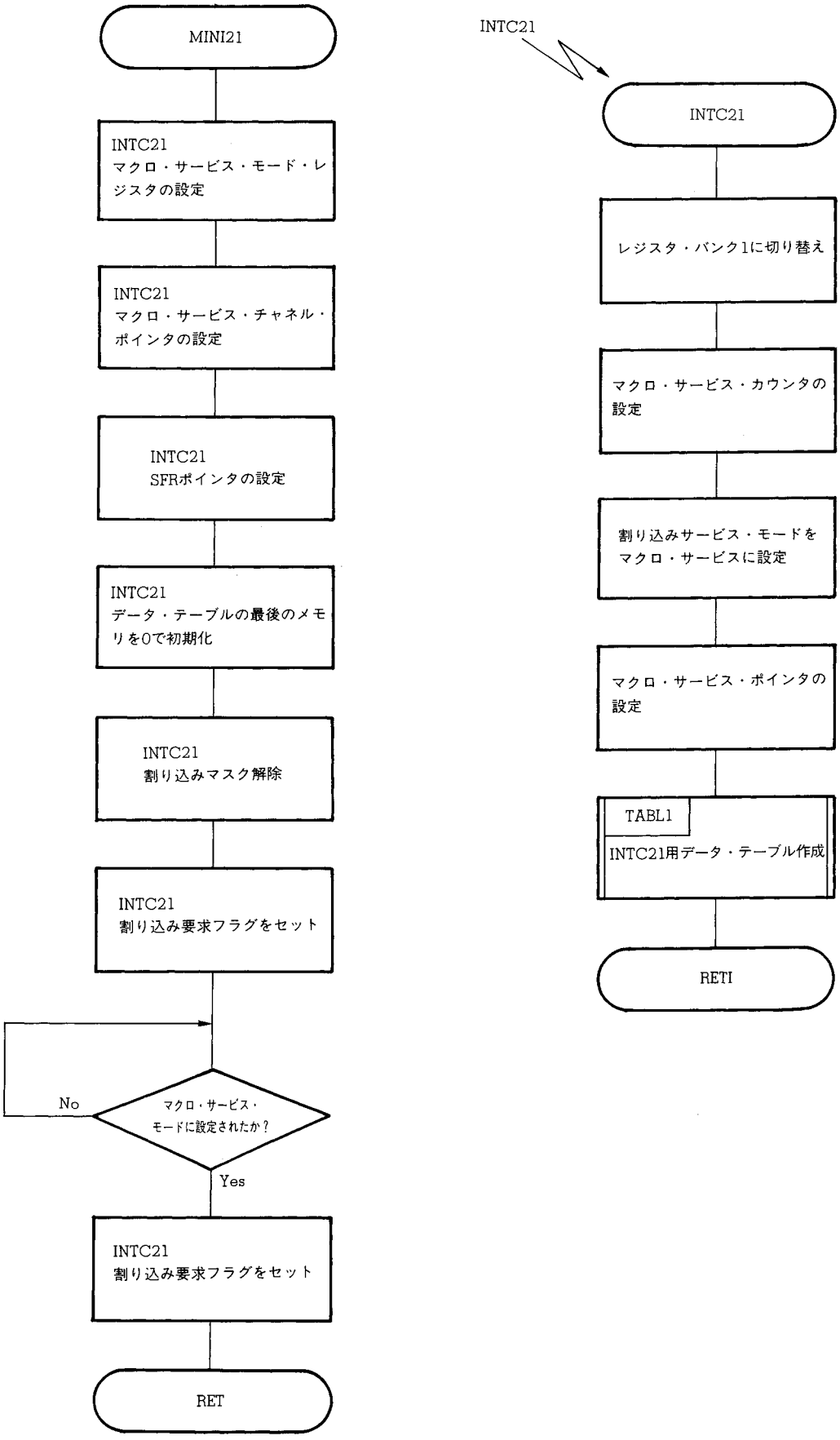
(8) フロー・チャート

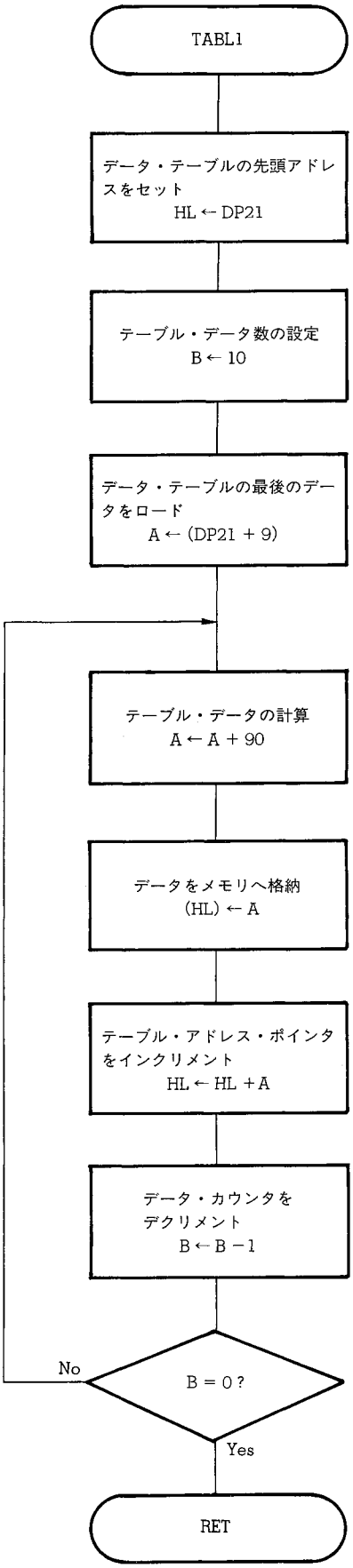
(a) マクロ・サービス・タイプA (INTC20)




TABL0
データ・テーブルの先頭アドレスをセット
HL ← DP20
テーブル・データ数の設定
B ← 10
データ・テーブルの最後のデータをロード
A ← (DP20 + 9)
テーブル・データの計算
A ← A + 60
データをメモリへ格納
(HL) ← A
テーブル・アドレス・ポインタをインクリメント
HL ← HL + A
データ・カウンタをデクリメント
B ← B − 1
B = 0 ?
No
Yes
RET

(b) マクロ・サービス・タイプB (INTC21)




TABL1
データ・テーブルの先頭アドレスをセット
HL ← DP21
テーブル・データ数の設定
B ← 10
データ・テーブルの最後のデータをロード
A ← (DP21 + 9)
テーブル・データの計算
A ← A + 90
データをメモリへ格納
(HL) ← A
テーブル・アドレス・ポインタをインクリメント
HL ← HL + A
データ・カウンタをデクリメント
B ← B − 1
No
B = 0 ?
Yes
RET

(9) プログラム・リスト

(a) マクロ・サービス・タイプA (INTC20)

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS. DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    MINI0
;
;*************************************************************
;* 8bit-Timer / Counter-2                                    *
;*    free running interval timer by macro service TYPE A    *
;*************************************************************
;
        PUBLIC  MINI20
        EXTRN   INTVL8
        EXTRN   DP20,MSC20D,DP20E,MSCC20

        CMCW20                  ; INTC20 macro service control word
        CVENT20                 ; INTC20 vector

;*************************************************************
;*      INTC20 macro service initialize (TYPE A)             *
;*************************************************************
;
        CSEG
MINI20:
        MOV     MSMC20,#01000000B       ; set INTC20 macro service mode
        MOV     CHPC20,#LOW(MSCC20)  ; set INTC20 macro service channel pointer
        MOV     DP20E,#0                ; initialize of INTC20 data pointer
        CLR1    CMK20                   ; open INTC20 mask
        SET1    CIF20                   ; set INTC20 request flag
        NOP
        NOP
ISMLP0:
        BF      CISM20,$ISMLP0
        SET1    CIF20                   ; set INTC20 request flag
        RET
;
;*************************************************************
;*      complete of INTC20 macro sevice                      *
;*                  macro service reset                      *
;*                  make data table                          *
;*************************************************************
;
INTC20:
        SEL     RB1                     ; change register bank
        MOV     MSCC20,#LOW(MSC20D)     ; set macro service counter data
        SET1    CISM20                  ; set macro service mode
        CALL    !TABL0
        RETI
```

```
;
;*************************************************************
;*      INTC20 macro service data table                      *
;*************************************************************
;
TABL0:
        MOVW    HL,#DP20        ; set data pointer
        MOV     B,#LOW(MSC20D)  ; set data counter
        MOV     A,DP20E         ; get table end data
ADLP0:
        ADD     A,#LOW(INTVL8)  ; add data
        MOV     [HL+],A         ; data pointer +1
        DBNZ    B,$ADLP0        ; data counter -1
        RET
;
        END
```


214
218A
234
244

(b) マクロ・サービス・タイプB (INTC21)

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    MINI1
;
;*************************************************************
;* 8bit-Timer / Counter-2                                    *
;*    free running interval timer by macro service TYPE B    *
;*************************************************************
;
        PUBLIC  MINI21
        EXTRN   INTVL9
        EXTRN   DP21,MSC21D,DP21E
;
        DSEG_B  C21,'SADDR'
        CMCW21

        CVENT21                 ; INTC21 vector
;
;*************************************************************
;*      INTC21 macro service initialize                      *
;*************************************************************
;
        CSEG
MINI21:
        MOV     MSMC21,#10100000B       ; set INTC21 macro service mode
        MOV     CHPC21,#LOW(CHAC21)  ; set INTC21 macro service channel pointer
        MOV     SFRC21,#LOW(CR21)       ; set INTC21 sfr pointer(CR21)
        MOV     DP21E,#0                ; initialize of INTC21 data pointer
        CLR1    CMK21                   ; open INTC21 mask
        SET1    CIF21                   ; set INTC21 request flag
        NOP
        NOP
ISMLP1:
        BF      CISM21,$ISMLP1
        SET1    CIF21                   ; set INTC21 request flag
        RET
;
;*************************************************************
;*      complete of INTC21 macro service                     *
;*                      macro service reset                  *
;*                      make data table                      *
;*************************************************************
;
INTC21:
        SEL     RB1                     ; change register bank
        MOV     MSCC21,#LOW(MSC21D)     ; set macro service counter data
        SET1    CISM21                  ; set macro service mode
        MOVW    MPC21P,#DP21            ; set macro service pointer
        CALL    !TABL1
        RETI
```

```
;
;*************************************************************
;*      INTC21 macro service data table                      *
;*************************************************************
;
TABL1:
        MOVW    HL,#DP21        ; set data pointer
        MOV     B,#LOW(MSC21D)  ; set data counter
        MOV     A,DP21E         ; get table end data
ADLP1:
        ADD     A,#LOW(INTVL9)  ; add data
        MOV     [HL+],A         ; data pointer +1

        DBNZ    B,$ADLP1        ; data counter -1
        RET
;
        END
```


214
218A
234
244

2.2　高分解能PWM出力

8ビット・タイマ／カウンタ2とマクロ・サービスを使用し，PWM出力の分解能を上げた例を示します。

8ビット・タイマ／カウンタ2の分解能は8ビットですので，PWM出力を行った場合，タイマ2において出力可能な波形のデューティは，1/256が最小となります。この応用例は，数回分のPWM波形を平均することにより，分解能を上げるというものです。

この応用例は，μPD78224シリーズでは使用できません。

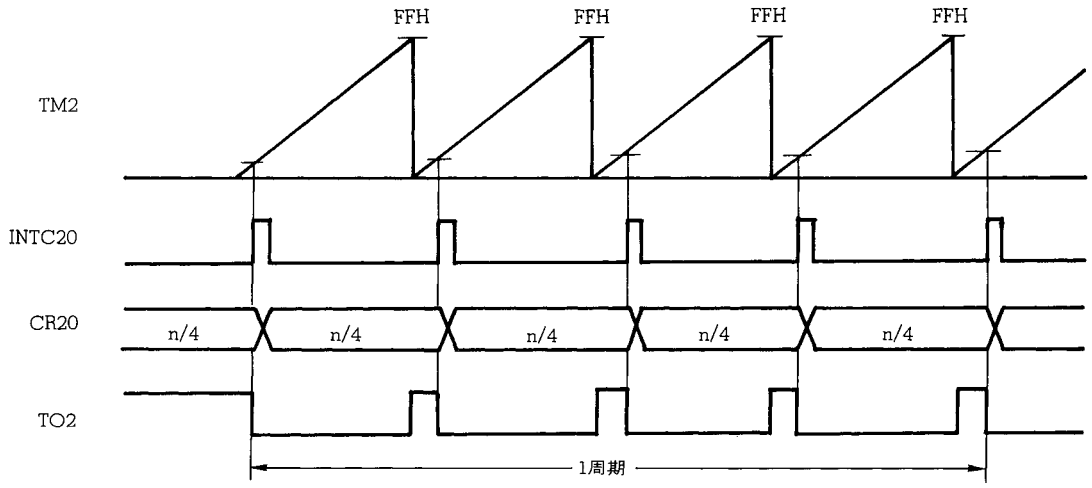
(1) 動作概要

INTC20割り込み要求を使用して，TO2端子から10ビット分解能を持ったPWM出力を行うプログラム例を示します。マクロ・サービスは，タイプAを使用します。

10ビット分解能のPWM出力は，タイマ2の4回分のPWM出力を1周期とする（4回分のPWM出力を平均する）ことによって実現できます。この場合のデューティの最小変化幅は，$1/(2^8 \times 4) = 1/1024$となり，出力可能なデューティは，4/1024から1020/1024になります。図2-4に10ビット分解能PWM出力のタイミング・チャートを示します。

図2-4　10ビット分解能PWM出力のタイミング

〈デューティ n/1024の場合〉



$$1\text{周期} = \frac{2^{10}}{f_{CLK}/X}$$

X＝16，32，64，128，256，512

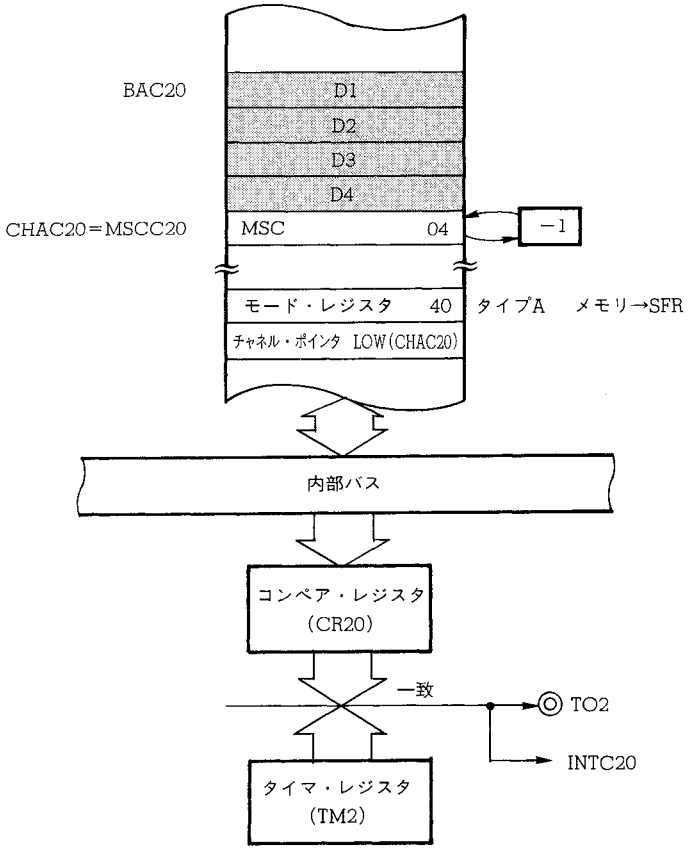
8ビット・タイマ／カウンタ2のカウント・クロックは，$f_{CLK}/16$を選択します。
この場合の1周期の幅は，次のようになります。

$$\frac{1024}{f_{CLK}/16} = 2.7\ \text{ms}$$

コンペア・レジスタ（CR20）への値の設定は，マクロ・サービスで行いますので，マクロ・サービス・バッファに，データ・テーブルとして4回分の転送データをあらかじめ用意しておきます。このデータ・テーブルへの設定値は，タイマのフル・カウント4回分に対するデューティ比から求められる値の1/4です。ただし，コンペア・レジスタ（CR20）へは，整数値しか設定できませんので，小数点以下の値は，各テーブル・データへ振り分ける必要があります。

図2-5にマクロ・サービスによる8ビット・コンペア・レジスタ（CR20）へのデータ転送のメモリ・マップ，および動作ブロック図を示します。

図2-5　マクロ・サービスによるCR20へのデータ転送



2
214
218A
234
244

また出力波形のデューティを変更する場合は，RAM中のワーク・エリアに次のデューティを決定する値をセットし，デューティ変更フラグをセットします。現在処理中のマクロ・サービスが終了した時点（マクロ・サービス完了割り込み処理）で，次のマクロ・サービスのためのデータ・テーブルを作成します。

使用するワーク・エリアを表2-1に示します。

表2-1　高分解能PWM出力のプログラムで用いるワーク・エリア

ワーク・エリア名称	用　　途	バイト数
DUTY5	デューティを決定する値を格納する	2バイト
C_FLG	デューティ変更要求の有無を判断する	1ビット

なおDUTY5の値は10ビット・データとして扱います。

(2) プログラム説明…(8) プログラム・リスト参照

(a) メイン処理［レーベル名称：PWM_M］

メインでは，マクロ・サービスのイニシャライズ処理，データ・テーブル作成，PWM出力処理ルーチンを呼び出します。

① マクロ・サービスのイニシャライズを行います。

② データ・テーブルを作成します。

③ PWM出力のイニシャライズを行います。

(b) マクロ・サービス・イニシャライズ［レーベル名称：MINIT］

① INTC20マクロ・サービス・モード・レジスタの設定を行います。
タイプA，メモリ→SFRに設定します。

② INTC20マクロ・サービス・チャネル・ポインタの設定を行います。

③ INTC20マクロ・サービス・カウンタの設定を行います。

④ INTC20割り込みサービス・モードをマクロ・サービスに設定します。

(c) データ・テーブル作成［レーベル名称：TABLE］

テーブルへの設定データ（コンペア・レジスタ（CR20）への設定値）の計算を行い，計算値を4バイトのメモリへ格納します。

① n/4（n：デューティ値）の商（q）と余り（r）を求めます。商（q）が，4バイトのデータ・テーブルへ設定する値となります。ただしn/4の結果，余りが生じた場合は，その余りをデータ・テーブルへ均等に振り分けています。

例1．n＝22の場合

22 ÷ 4 ＝ 5 … 2

2 ― テーブルへ1ずつ振り分ける

5 ― テーブルへの設定値

この場合，次のような振り分け方が適当です。

(A)

テーブル

5+1	5	5+1	5

あるいは，

(B)

テーブル

5	5+1	5	5+1

このプログラムでは，(A)の方法で行います。

プログラムでの処理方法として，次のようなサブ・テーブルを用意して，余りに応じて4バイトのデータ・テーブルのどのメモリに＋1の値を設定すればよいか検出します。

```
SHTBL:
        DB      00000000B
        DB      10000000B
        DB      10100000B
        DB      11100000B
```

備考　サブテーブルへ8ビット分のデータを設定すれば，11ビット分解能PWM出力（8回のPWM出力を平均する）のテーブル・データを計算することができます。

2
214
218A
234
244

② サブ・テーブルを参照して，余り（r）をデータ・テーブルに振り分けます。余り（r）をAレジスタにセットして，'MOV A, SHTBL[A]'という命令を実行すると，余りに応じたサブ・テーブルを参照することができます。**例1**の場合では，参照するサブ・テーブル値は'10100000B'となります。この値を左シフトし，CYと商（q）を加算する処理を4回繰り返すと，4バイトのデータ・テーブルへ余りを均等に振り分けることができます（**例2**参照）。

例2. 余り（r）＝2の場合

MOV A, SHTBL [A] ; A←10100000B

	CY	
初期状態	0	10100000
1回シフト	1	01000000
2回シフト	0	10000000
3回シフト	1	00000000
4回シフト	0	00000000

データ・テーブル	q+1	q+0	q+1	q+0

(d) 商（q），余り（r）の計算［レーベル名称：DATA］

① デューティを決定する値n（DUTY5の値）をAXレジスタにセットします。

② AXレジスタの値に対して，16ビット右シフト命令（SHRW）を2回実行することによって，n/4の計算を行います。

(e) PWM出力イニシャライズ［レーベル名称：PWM20］

① TO2タイマ出力のアクティブ・レベルをハイ・レベルに設定し，タイマ出力を許可します。

注意　PWM/PPG出力では，タイマ・コントロール・レジスタ（TOC）のALV2ビット＝0に設定した場合，アクティブ・レベルはハイ・レベルとなります。

② P36をTO2出力端子として用いるために，コントロール・ポートに指定します。

③ 8ビット・タイマ・レジスタ（TM2）と8ビット・コンペア・レジスタ（CR20）の一致によるタイマのクリアを禁止し，TO2端子をPWM出力モードに設定します。

④ 8ビット・タイマ／カウンタ2のカウント・クロックをf_{CLK}/16に指定します。

⑤ 8ビット・コンペア・レジスタ（CR20）へ最初のPWM出力のパルス幅を設定します。マクロ・サービスによりCR20へデータを転送するので，最初の転送データは，データ・テーブルの先頭データとなります。したがって，CR20の初期値には，データ・テーブルの先頭アドレス＋3のメモリ値（データ・テーブルの最後の値）を設定しておきます。

⑥ INTC20割り込み要求のマスクを解除します。

⑦ 8ビット・タイマ／カウンタ2のカウント動作を許可します。

⑧ 割り込み許可状態にします。

(f) INTC20マクロ・サービス完了割り込み処理［レーベル名称：INTC20］

① レジスタ・バンク1に切り替えます。

② デューティ変更要求の有無の判断を行います。デューティ変更要求がある（C_FLG＝1）場合は，C_FLGをクリア(0)したのち，次のデューティを決定するテーブルを作成します。デューティ変更要求がない場合は，これらの処理は何も行いません。

③ INTC20マクロ・サービス・カウンタを再設定します。

④ INTC20割り込みサービス・モードをマクロ・サービスに設定します。

2
214
218A
234
244

(3) モード・レジスタ設定例

マクロ・サービス・モード・レジスタ

7	6	5	4	3	2	1	0
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例 0	1	0	0	0	0	0	0

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

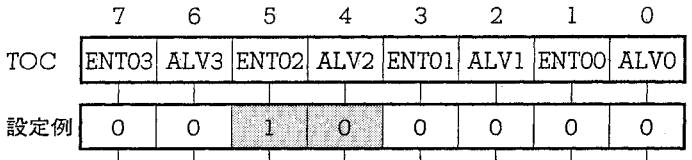
MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし	データ転送のみ POL
1	0	0	1				POH
1	0	1	0				自動加算つき POL
1	0	1	1				POH
1	1	0	0			リング制御つき	データ転送のみ POL
1	1	0	1				POH
1	1	1	0				自動加算つき POL
1	1	1	1				POH

割り込みサービス・モード・レジスタH

7	6	5	4	3	2	1	0	
ISM0H CSIISM	STISM	SRISM	0	CISM20	PISM5	PISM4	CSIM21	（リセットで00H）
設定例 ×	×	×	×	1	×	×	×	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

タイマ出力コントロール・レジスタ



	7	6	5	4	3	2	1	0	
TOC	ENT03	ALV3	ENT02	ALV2	ENT01	ALV1	ENT00	ALV0	（リセットで00H）
設定例	0	0	1	0	0	0	0	0	

ALV0	TO0端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT00	TO0端子の動作指定
0	$\overline{\text{ALV0}}$を出力
1	パルス出力許可

ALV1	TO1端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT01	TO1端子の動作指定
0	$\overline{\text{ALV1}}$を出力
1	パルス出力許可

ALV2	TO2端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

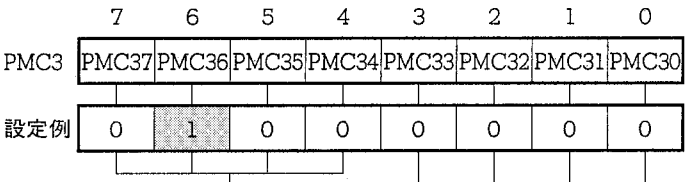
ENT02	TO2端子の動作指定
0	$\overline{\text{ALV2}}$を出力
1	パルス出力許可

ALV3	TO3端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT03	TO3端子の動作指定
0	$\overline{\text{ALV3}}$を出力
1	パルス出力許可

2
214
218A
234
244

ポート3モード・コントロール・レジスタ



PMC30	P30端子のコントロール・モード指定
0	入出力ポート・モード
1	RxD入力モード

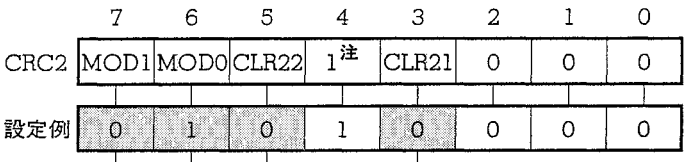
PMC31	P31端子のコントロール・モード指定
0	入出力ポート・モード
1	TxD出力モード

PMC32	P32端子のコントロール・モード指定
0	入出力ポート・モード
1	$\overline{\text{SCK}}$入出力モード

PMC33	P33端子のコントロール・モード指定
0	入出力ポート・モード
1	SO出力／SB0入出力モード

PMC3n	PM3n端子のコントロール・モード指定(n=4-7)
0	入出力ポート・モード
1	TOm出力モード（m=0-3）

キャプチャ／コンペア・コントロール・レジスタ2



（リセットで10H）

CLR21	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（CR21とTM2が一致したらクリア）

CLR22	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（キャプチャしたらクリア）

MOD1	MOD0	タイマ出力モード指定		
		TO2	TO3	その他条件
0	0	トグル出力	トグル出力	—
0	1	PWM出力	トグル出力	CLR21ビット=0
1	0	PWM出力	PWM出力	CLR22ビット=0
1	1	PPG出力	トグル出力	CLR21ビット=1 CLR22ビット=0

注　必ず“1”に設定してください。

2
214
218A
234
244

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	0	0	0	0	0	0	0	

8ビット・タイマ／カウンタ1

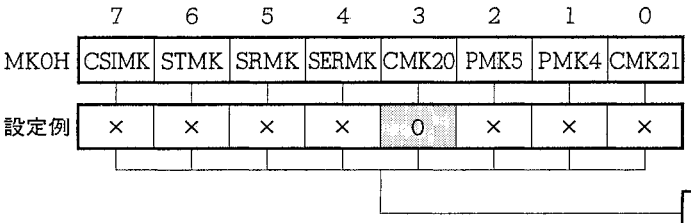
PRS12	PRS11	PRS10	タイマ／カウンタ1のカウント・クロック周波数の指定
0	0	0	$f_{CLK}/16$[注]
0	0	1	
0	1	0	
0	1	1	$f_{CLK}/32$
1	0	0	$f_{CLK}/64$
1	0	1	$f_{CLK}/128$
1	1	0	$f_{CLK}/256$
1	1	1	$f_{CLK}/512$

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	タイマ／カウンタ3のカウント・クロック周波数の指定
0	0	0	0	$f_{CLK}/16$
0	0	0	1	
0	0	1	0	
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	1	1	1	外部クロック（CI）

注　f_{CLK}：内部システム・クロック周波数（$f_{XX}/2$）

割り込みマスク・レジスタH



MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

タイマ・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	1	0	0	0	0	0	0	0	

8ビット・タイマ／カウンタ1

OVF1	タイマ／カウンタ1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE1	タイマ／カウンタ1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

CMD2	タイマ／カウンタ2の動作モード指定
0	通常モード
1	ワンショット・モード

OVF2	タイマ／カウンタ2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE2	タイマ／カウンタ2のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

(4) 入力パラメータ

データ・テーブル作成

DUTY5：デューティを決定する値（10ビット）を設定します。

C_FLG：デューティ変更要求の有無を判断するフラグです。デューティ変更要求があったときにセット(1)します。

(5) 使用レジスタ

(a) マクロ・サービス

なし

(b) データ・テーブル作成（1回目）

A, X, B, D, E, HL

(c) マクロ・サービス完了割り込み

A, X, B, D, E, HL（レジスタ・バンク1）

(6) プログラム使用例

TO2端子から，デューティ265/1024のPWM出力を行う例を示します。この場合，8ビット・コンペア・レジスタ（CR20）への設定値は265/4です。実際には，

$265 \div 4 = 66 \cdots 1$

の結果より，次のようなデータ・テーブルが作成されます。

67	66	66	66

このテーブル・データが，マクロ・サービスで連続的に転送されます。

214

218A

234

244

8ビット・タイマ／カウンタ2のカウント・クロックは，$f_{CLK}/16$を選択するので，1周期（4回分のタイマのフル・カウント周期）中のアクティブ・レベルのパルス幅は，次のようになります。

$$\frac{1024}{f_{CLK}/16}\times\frac{265}{1024}=707\,\mu s$$

プログラム例として，次のように用います。

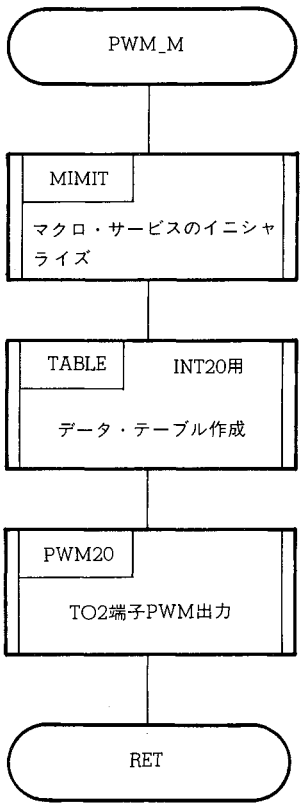
なおデューティ変更要求処理において，次のPWM出力のデューティを決定する値は，何らかの方法でAXレジスタにセットされているものとしています。

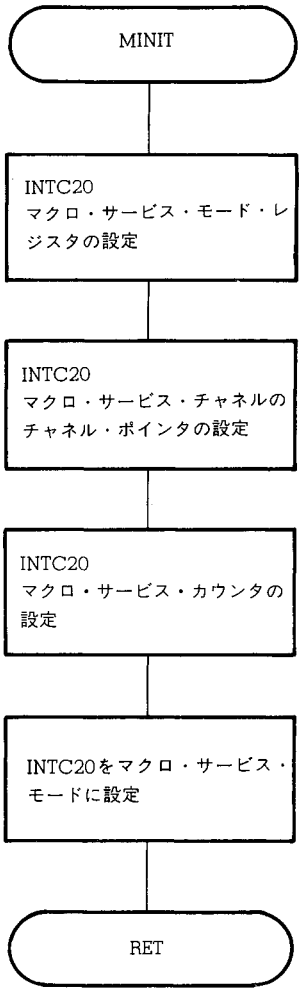
またデューティ変更要求フラグ（C_FLG）のセットは，DUTY5の値を設定したあとに行ってください。C_FLGは，イニシャライズ処理でクリア（0）しておく必要があります。

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードします。

```
$       IC(INTMS.DEF)
                :
                :
        PUBLIC  C_FLG,DUTY5
        EXTRN   PWM_M                           ; PACKAGE
;
DUTY5D  EQU     265             ; data of duty

        BSEG
C_FLG   DBIT

PWM_M_D DSEG    SADDR
DUTY5:  DS      2
                :
                :
        CSEG
P_MAIN:
                :
                :
        MOVW    AX,#DUTY5D
        MOVW    DUTY5,AX        ; set duty to RAM
        CALL    !PWM_M
                :
                :
C_DTY:
                :
                :
        MOVW    DUTY5,AX        ; set duty
        SET1    C_FLG           ; set flag of change duty
                :
                :
                :
                :
```

(7) フロー・チャート





2
214
218A
234
244

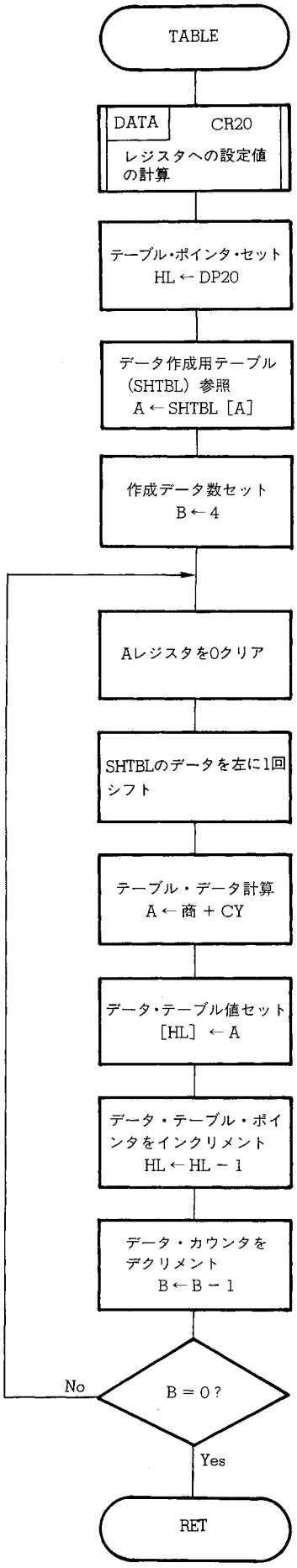
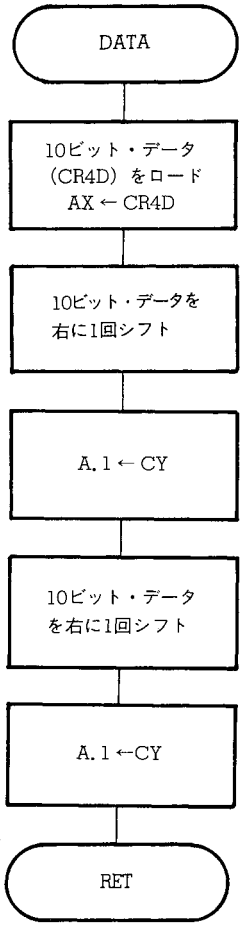
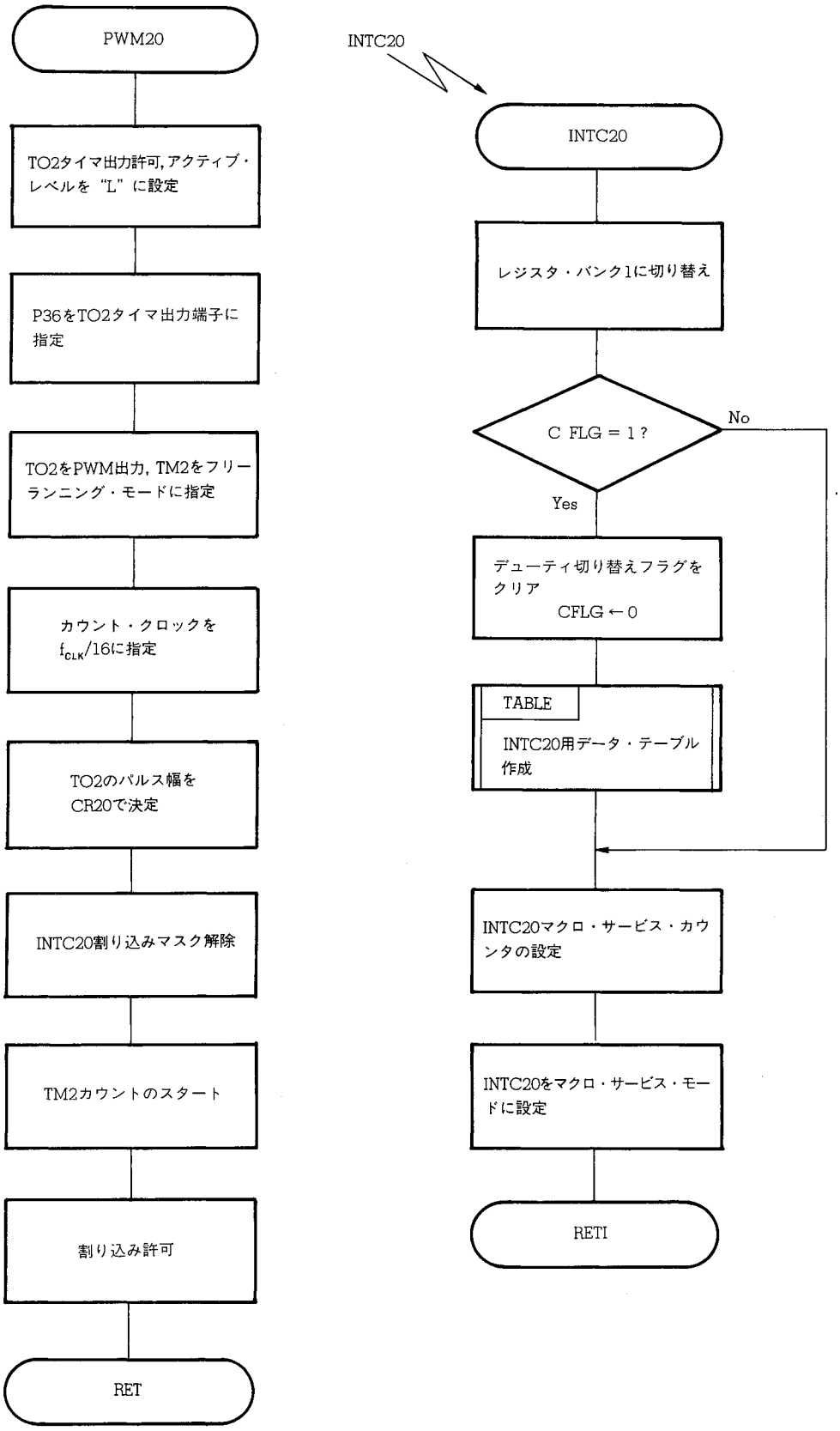


TABLE
DATA
CR20
レジスタへの設定値の計算
テーブル・ポインタ・セット
HL ← DP20
データ作成用テーブル（SHTBL）参照
A ← SHTBL [A]
作成データ数セット
B ← 4
Aレジスタを0クリア
SHTBLのデータを左に1回シフト
テーブル・データ計算
A ← 商 + CY
データ・テーブル値セット
[HL] ← A
データ・テーブル・ポインタをインクリメント
HL ← HL − 1
データ・カウンタをデクリメント
B ← B − 1
B = 0 ?
No
Yes
RET


DATA
10ビット・データ（CR4D）をロード
AX ← CR4D
10ビット・データを右に1回シフト
A.1 ← CY
10ビット・データを右に1回シフト
A.1 ←CY
RET



2
214
218A
234
244

(8) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    PWMM
;
;*************************************************************
;*       8bit-Timer / Counter-2                              *
;*                      10bit PWM output                     *
;*************************************************************
;
        PUBLIC  PWM_M
        EXTRN   DUTY5
        EXTBIT  C_FLG

MSCD    EQU     4               ; INTC20 macro service counter data

        DSEG_A  C20,MSCD,'SADDR'
        CMCW20

;
        CVENT20                 ; INTC20 vector
;
;*************************************************************
;*      MAIN ROUTIN                                          *
;*              macro sevice initialize                      *
;*              make data table                              *
;*************************************************************
;
        CSEG
PWM_M:
        CALL    !MINIT          ; initialize of macro service
        CALL    !TABLE          ; make macro service data table
        CALL    !PWM20          ; PWM output
        RET
;
;*************************************************************
;*      INTC20 maacro service initialize                     *
;*************************************************************
;
MINIT:
        MOV     MSMC20,#01000000B       ; set macro service mode
        MOV     CHPC20,#LOW(CHAC20)     ; set macro service channel pointer
        MOV     MSCC20,#MSCD            ; set macro service counter data
        SET1    CISM20                  ; set INTC20 to macro service mode
        RET
```

```
;
;*************************************************************
;*      macro service data table                             *
;*              A : remainder                                *
;*              X : quotient                                 *
;*************************************************************
;
TABLE:
        CALL    !DATA           ; make data
        MOVW    HL,#BAC20       ; set table data pointer
        MOV     E,#0
        MOV     A,SHTBL[A]
        MOV     D,A
        MOV     B,#MSCD
LOOP:
        MOV     A,E             ; A register 0 clear
        ADD     D,D             ; shift to left once
        ADDC    A,X             ; count table data
        MOV     [HL+],A         ; data pointer +1
        DBNZ    B,$LOOP
        RET
SHTBL:
        DB      00000000B       ; shift data
        DB      10000000B
        DB      10100000B
        DB      11100000B
;
;*************************************************************
;*              make table data                              *
;*************************************************************
;
DATA:
        MOVW    AX,DUTY5        ; 10bit data set
        SHRW    AX,1            ; 2/10bit data
        MOV1    A.1,CY          ; set CY
        SHRW    AX,1            ; 2/10bit data
        MOV1    A.1,CY          ; set CY
        RET
;
;*************************************************************
;*              PWM output initialize                        *
;*                      from TO2                             *
;*************************************************************
;
PWM20:
        MOV     TOC,#00100000B  ; timer output,active level high
        MOV     PMC3,#01000000B ; P3 control port
        MOV     CRC2,#01010000B ; set timer free running mode
        MOV     PRM1,#00000000B ; set prescaler fclk/16
        MOV     CR20,BAC20+MSCD-1  ; set table end data for TO2 pulse width
        CLR1    CMK20           ; open INTC20 mask
        MOV     TMC1,#10000000B ; start timer
        EI
        RET
```


214

218A

234

244

```
;
;*************************************************************
;*      complete of INTC20 macro service                     *
;*************************************************************
;
INTC20:
        SEL     RB1             ; change register bank
        BF      C_FLG,$M_COMP   ; change duty?
        CLR1    C_FLG           ; change flag of duty clear
        CALL    !TABLE          ; make data table
M_COMP:
        MOV     MSCC20,#MSCD    ; set INTC20 macro serbice counter
        SET1    CISM20          ; set INTC20 to macro service mode
        RETI
;
        END
```

2.3　高分解能キャプチャ機能

８ビット・タイマ／カウンタのキャプチャ機能とマクロ・サービスを使用して，16ビット分解能のキャプチャ機能を実現するプログラム例を示します。

このプログラム例では，短時間パルスから長時間パルスまで（数10 μs～数s）幅広く応用できます。

この応用例は，μPD78224シリーズの場合は使用できません。また，他のシリーズでは，タイマ／カウンタ２だけではなく，タイマ／カウンタ１でも使用できます。

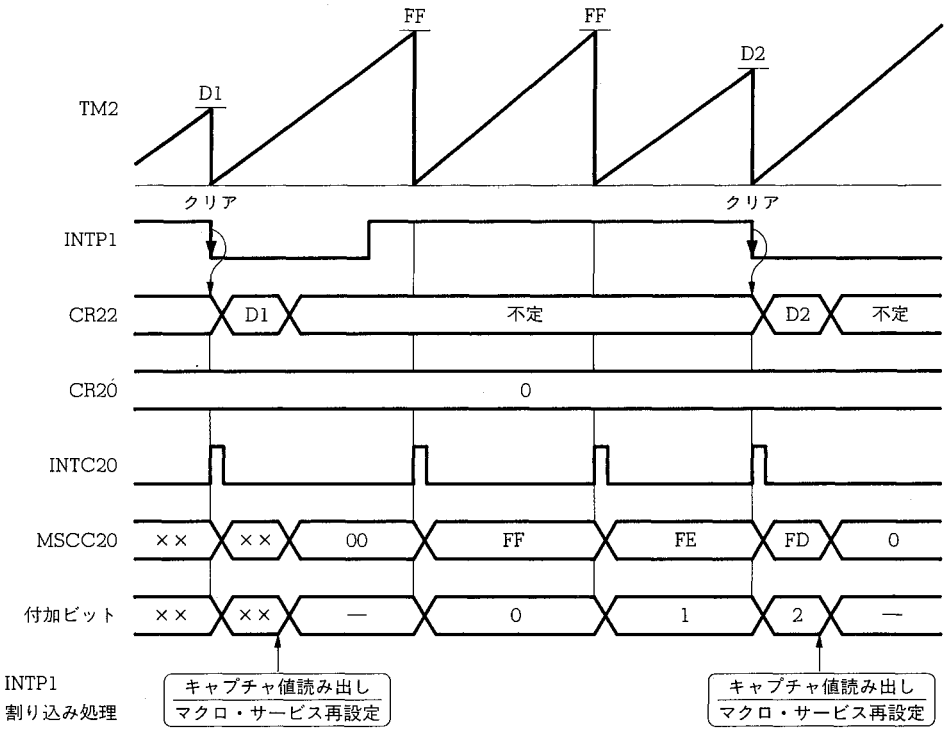
2.3.1　動作概要

INTP1端子に入力されるパルスの周期を測定します。

８ビット・タイマ／カウンタ２のオーバフロー回数をマクロ・サービスによってカウントし，キャプチャ値の上位８ビットとします。また，キャプチャ・レジスタ（CR22）に取り込まれた値は，そのままキャプチャ値の下位８ビットとします。

図２-６にタイミング・チャートを示します。

図２-６　長時間パルスの周期測定のタイミング・チャート



コンペア・レジスタ(CR20)に０を設定しておくと，タイマがオーバフローしたときに一致信号(INTC20)が発生します。INTC20をマクロ・サービスで処理し，マクロ・サービス・カウンタに０を初期設定しておくと，INTP1エッジ検出時のマクロ・サービス・カウンタ（MSCC20）の値から上位８ビットのキャ

プチャ値を得ることができます（MSCC20を反転）。

また，このプログラム例では，INTC20マクロ・サービス完了割り込みが発生した（マクロ・サービスが256回行われた）ら，オーバフロー（測定不可能）とし，オーバフロー・フラグをセットしています。

マクロ・サービスは，タイプB，メモリ→SFR（CR21）を使用しますが，タイマのオーバフロー回数をカウントするだけが目的ですので，転送動作は意味を持ちません。したがって，転送先SFRは，使用システムにあわせ，書き込んでも問題のないSFR（システムで使用しない書き込み可能なSFR）を選択してください。また転送データは，どんなデータでもかまいません。この応用例では，マクロ・サービス・ポインタを0H番地に設定しています。

このプログラム例では，TM2のカウント・クロックを$f_{CLK}/64$（f_{CLK}：6 MHz）としており，この場合の測定可能なパルス周期は，次の範囲となります。

最小：（INTP1割り込み受け付け時間）＋（INTP1割り込み処理時間）＋
（1回のマクロ・サービス実行時間）
最大：$64/f_{CLK} \times 256 \times 256 \fallingdotseq 699$ ms

備考　TM2のカウント・クロックを$f_{CLK}/512$に変更すると，最大，次に示すパルスの入力が可能となります。

$512/f_{CLK} \times 256 \times 256 \fallingdotseq 5.6$s

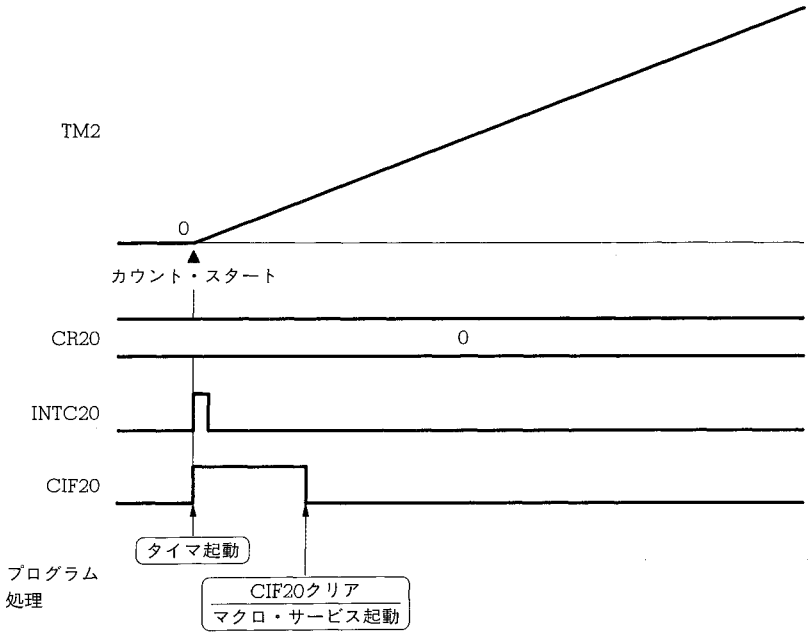
2.3.2　プログラム説明

(1) 処理概要（「(8) SPDチャート」参照）

(a) イニシャライズ処理［レーベル名称：CAP_INI］

8ビット・タイマ／カウンタ2，ワーク・フラグ，INTC20マクロ・サービスのイニシャライズを行います。ただしCR20＝0のため，タイマ起動直後すぐにINTC20が発生します。そのため，マクロ・サービスの起動は，タイマを起動後INTC20割り込み要求フラグ(CIF20)をチェックし，CIF20＝1になったら，CIF20をクリアしたあとに行います（図2-7参照）。

図2-7　タイマ起動直後のタイミング・チャート



2
214
218A
234
244

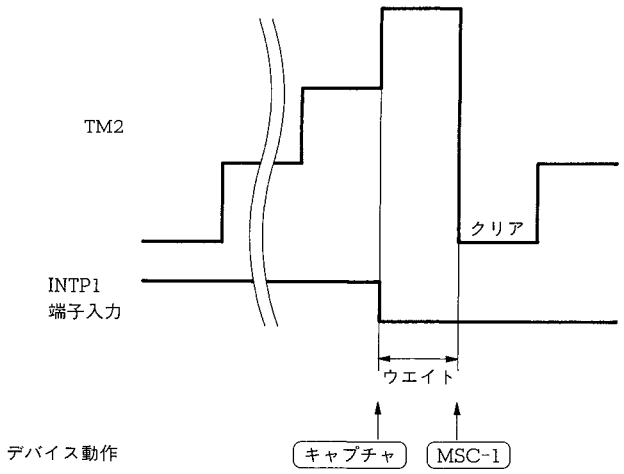
(b) INTP1ベクタ割り込み処理［レーベル名称：INTP1］

(i) キャプチャ値の読み出し

キャプチャ後，タイマがクリアされるまでには，最大でタイマ／カウンタの1カウントの時間がかかります。したがって，マクロ・サービス・カウンタ（キャプチャ値の上位8ビット）を読み出す前に，タイマ／カウンタの1カウント分のウエイトを入れる必要があります（**図2－8**参照）。

上位，下位のキャプチャ値を読み出したあと，キャプチャ終了フラグ(CAPENDF)をセット(1)します。

図2－8　キャプチャ動作とINTC20マクロ・サービスのタイミング



(ii) マクロ・サービスの再設定

INTP1割り込み処理中も，INTC20のサービス・モードはマクロ・サービスのままとなっています。したがって，マクロ・サービスの再設定は，マクロ・サービス・カウンタとマクロ・サービス・ポインタの設定のみ行います。

(c) INTC20マクロ・サービス完了割り込み処理［レーベル名称：INTC20］

INTC20マクロ・サービス完了割り込みが発生したら，オーバフローとし，この時点でのキャプチャ値は無効となります。プログラム処理では，オーバフロー・フラグ（OVFLWF）をセット(1)します。

(2) 使用するRAM

このプログラム例では，キャプチャ値(16ビット)の取り込み用に2バイトと2ビット，INTC20マクロ・サービス・チャネルとして，4バイトのRAMを使用します。表2-2に使用RAM一覧を示します。

表2-2　長時間パルスの周期測定で使用するRAM一覧

RAM名称	配置領域	用　　途	バイト数	初期値
CAPDATP	SADDR注	キャプチャ値（16ビット）を格納	2	0000H
CAPENDF		キャプチャ終了の判断	1ビット	0
OVFLWF		オーバフローの有無の判断	1ビット	0
MPC20P		INTC20マクロ・サービス・ポインタ	2	C20BUF
SFRC20		〃　　SFRポインタ	1	LOW CR21
MSCC20		〃　　カウンタ	1	0

注　SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

(3) 入出力パラメータ

(a) 入力パラメータ

なし

(b) 出力パラメータ

表2-3　長時間パルスの周期測定の出力パラメータ

パラメータ	固定／可変	内　　容
CAPDATP	可変値	キャプチャ値（16ビット）
CAPENDF	可変値	キャプチャ終了フラグ （このプログラムから復帰時は1）
OVFLWF	可変値	オーバフロー・フラグ

2
214
218A
234
244

(4) 使用レジスタ

表2－4　長時間パルスの周期測定の使用レジスタ

ルーチン	レーベル	レジスタ	バンク
イニシャライズ	CAP_INI	なし	—
INTP1ベクタ割り込み	INTP1	A	1
INTC20マクロサービス完了割り込み	INTC20	なし	—

(5) 使用スタック

このプログラム例のサブルーチン，および割り込み処理で使用するスタック・サイズは，3バイトです（ネスティング：1レベル）。

(6) モード・レジスタ設定例

キャプチャ／コンペア・コントロール・レジスタ2

	7	6	5	4	3	2	1	0
CRC2	MOD1	MOD0	CLR22	1注	CLR21	0	0	0
設定例	0	0	1	1	0	0	0	0

（リセットで10H）

CLR21	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（CR21とTM2が一致したらクリア）

CLR22	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（キャプチャしたらクリア）

MOD1	MOD0	タイマ出力モード指定		
		TO2	TO3	その他条件
0	0	トグル出力	トグル出力	—
0	1	PWM出力	トグル出力	CLR21ビット=0
1	0	PWM出力	PWM出力	CLR22ビット=0
1	1	PPG出力	トグル出力	CLR21ビット=1 CLR22ビット=0

注　必ず“1”に設定してください。

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	1	0	0	0	0	0	0	

8ビット・タイマ／カウンタ1

PRS12	PRS11	PRS10	タイマ／カウンタ1のカウント・クロック周波数の指定
0	0	0	$f_{CLK}/16$[注]
0	0	1	
0	1	0	
0	1	1	$f_{CLK}/32$
1	0	0	$f_{CLK}/64$
1	0	1	$f_{CLK}/128$
1	1	0	$f_{CLK}/256$
1	1	1	$f_{CLK}/512$

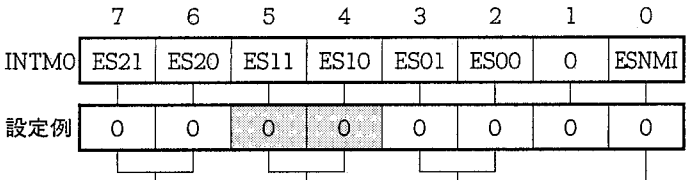
8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	タイマ／カウンタ3のカウント・クロック周波数の指定
0	0	0	0	$f_{CLK}/16$
0	0	0	1	
0	0	1	0	
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	1	1	1	外部クロック（CI）

注　f_{CLK}：内部システム・クロック周波数（$f_{XX}/2$）

2
214
218A
234
244

外部割り込みモード・レジスタ0



	7	6	5	4	3	2	1	0	
INTM0	ES21	ES20	ES11	ES10	ES01	ES00	0	ESNMI	（リセットで00H）
設定例	0	0	0	0	0	0	0	0	

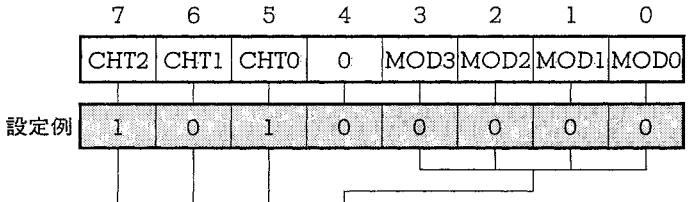
ESNMI	NMI端子入力，検出エッジ指定
0	立ち下がりエッジ
1	立ち上がりエッジ

ES01	ES00	INTP0端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES11	ES10	INTP1端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES21	ES20	INTP2端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

マクロ・サービス・モード・レジスタ



CHT0	0	1	0		1	
CHT1	1	0	1			
CHT2	0	1	1			

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし	データ転送のみ POL
1	0	0	1				POH
1	0	1	0				自動加算つき POL
1	0	1	1				POH
1	1	0	0			リング制御つき	データ転送のみ POL
1	1	0	1				POH
1	1	1	0				自動加算つき POL
1	1	1	1				POH

2
214
218A
234
244

タイマ・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	1	×	×	×	×	×	×	×	×：操作しません

8ビット・タイマ／カウンタ1

OVF1	TM1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE1	TM1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

CMD2	TM2のカウント動作モード指定
0	通常モード
1	ワンショット・モード

OVF2	TM2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE2	TM2のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

割り込み要求フラグ・レジスタL

	7	6	5	4	3	2	1	0	
IF0L	CIF11	CIF10	CIF01	CIF00	PIF3	PIF2	PIF1	PIE0	（リセットで00H）
設定例	×	×	×	×	×	×	0	×	×：操作しません

IF	割り込み要求フラグ
0	割り込み要求が発生していない
1	割り込み要求発生

割り込み要求フラグ・レジスタH

	7	6	5	4	3	2	1	0	
IF0H	CSIIF	STIF	SRIF	SERIF	CIF20	PIF5	PIF4	CIF21	（リセットで00H）
設定例	×	×	×	×	0	×	×	×	×：操作しません

IF	割り込み要求フラグ
0	割り込み要求が発生していない
1	割り込み要求発生

割り込みサービス・モード・レジスタH

	7	6	5	4	3	2	1	0	
ISM0H	CSIISM	STIISM	SRISM	0	CISM20	PISM5	PISM4	CISM21	（リセットで00H）
設定例	×	×	×	×	1	×	×	×	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

2
214
218A
234
244

割り込みマスク・レジスタL

	7	6	5	4	3	2	1	0	
MK0L	CMK11	CMK10	CMK01	CMK00	PMK3	PMK2	PMK1	PMK0	（リセットでFFH）
設定例	×	×	×	×	×	×	0	×	×：操作しません

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

割り込みマスク・レジスタH

	7	6	5	4	3	2	1	0	
MK0H	CSIMK	STMK	SRMK	SERMK	CMK20	PMK5	PMK4	CMK21	（リセットでFFH）
設定例	×	×	×	×	0	×	×	×	×：操作しません

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

(7) プログラム使用例

このプログラムの使用例を示します。

```
EXTRN   CAP_INI                 ; package
EXTRN   CAPDATP                 ; work area
EXTBIT  CAPENDF,OVFLWF          ; work flag
        :
        :
CSEG
        :
        :
CALL    !CAP_INI                ; initialize of capture function
EI                              ; enable interrupt
        :
        :
    if_bit(CAPENDF)             ; IF end capture ?
        if_bit(!OVFLWF)         ; THEN not over flow timer ?
            CLR1    CAPENDF     ;    clear capture end flag
            AX=CAPDATP          ;    THEN read capture data
        :
        :
        else                    ;    ELSE (case of over flow timer)
        :
        :
        endif
    endif
        :
        :
        :
        :
```


214

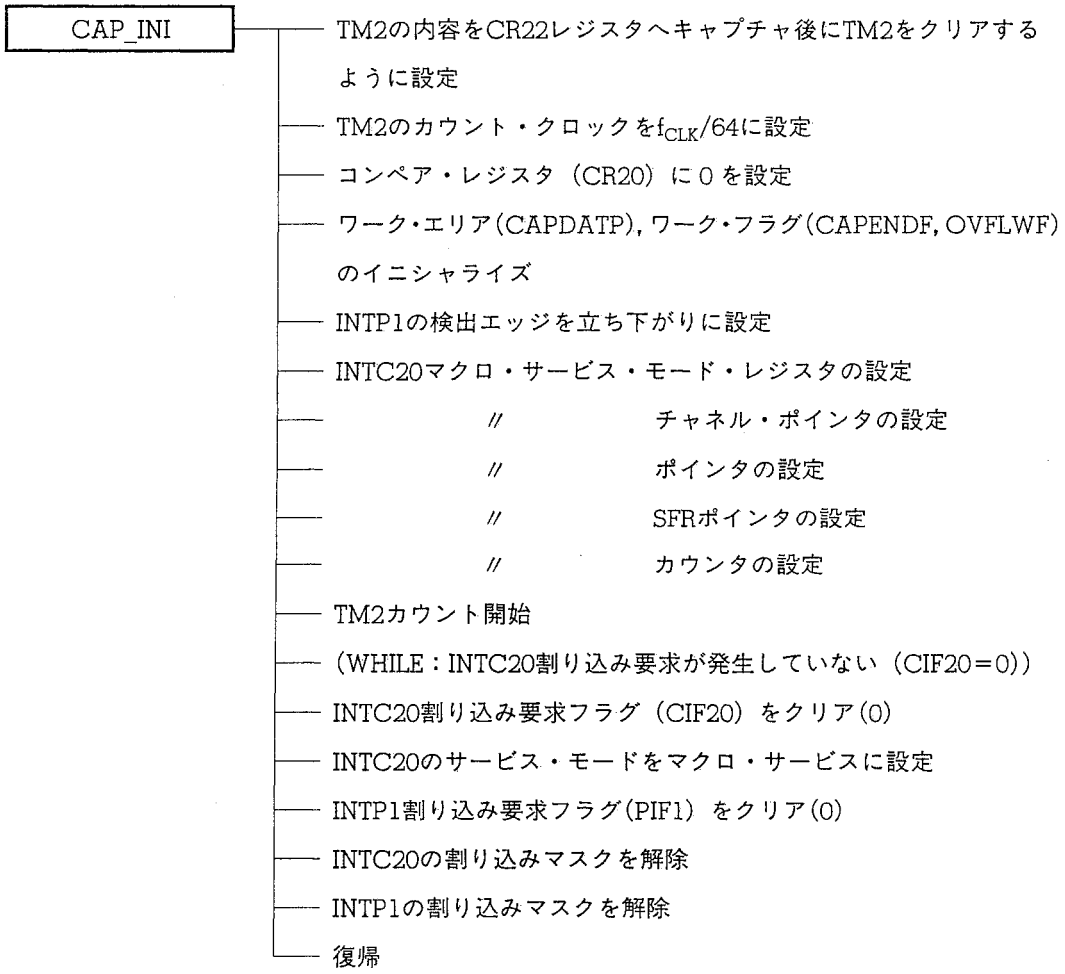
218A

234

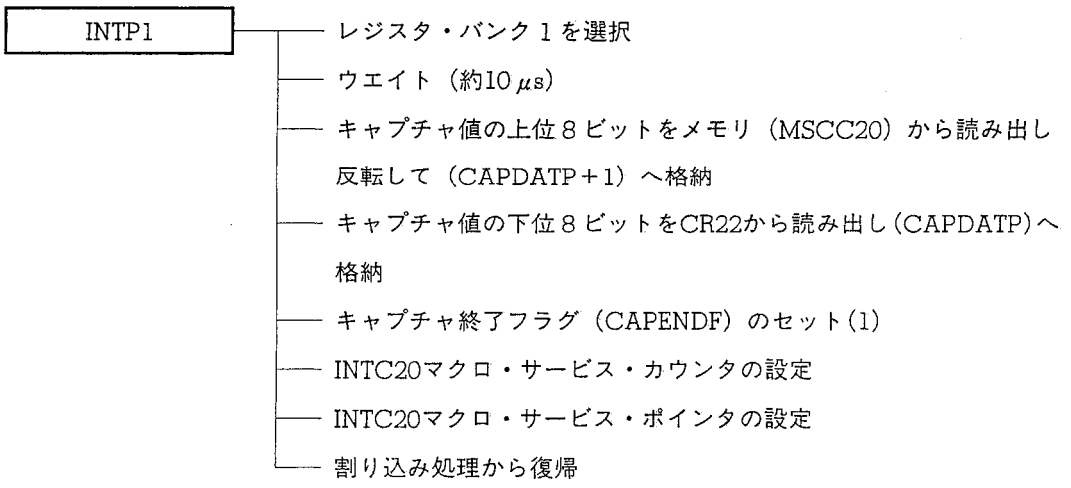
244

(8) SPDチャート

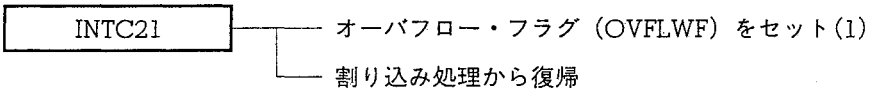
●イニシャライズ処理



●INTP1ベクタ割り込み処理（キャプチャ割り込み）



●INTC20マクロ・サービス完了割り込み処理



214
218A
234
244

(9) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードしています。

また，プログラムは構造化アセンブリ言語で記述されています。

```
$       IC(INTMS.DEF)
        NAME    CAPT
;*************************************************************
;*      HIGH RESOLUTE CAPTURE FUNCTION BY 8-BIT TIMER      *
;*************************************************************

        PUBLIC  CAP_INI                 ; package
        PUBLIC  CAPDATP                 ; work area
        PUBLIC  CAPENDF,OVFLWF          ; work flag

;       *** define work area ***

        BSEG
CAPENDF DBIT                            ; capture end flag
OVFLWF  DBIT                            ; over flow flag

CAPWKS  DSEG    SADDR
CAPDATP:DS      2                       ; capture data store area

;       *** define macro service area ***

C20BUF  EQU     0                       ; macro service pointer (dummy)

        CMCW20                          ; macro service control word
        DSEG_B  C20,'SADDR'             ; macro service channel

;       *** define vector table ***

        PVENT1                          ; INTP0 vector table
        CVENT20                         ; INTC20 vector table

;*****************************************
;*      initialize routine               *
;*****************************************

CAPTCS  CSEG
CAP_INI:
        CRC2=#00110000B                 ; enable clear TM2 by CR22 capture
        PRM1=#01000000B                 ; set prescaler fclk/64
        CR20=#0                         ; initialize CR20
        CAPDATP=#0000H                  ; initialize capture data store area
        CLR1    CAPENDF                 ; set case of not end capture
        CLR1    OVFLWF                  ; set case of no timer over flow
        INTM0=#00000000B                ; set INTP1 falling edge
```

```
        MSMC20=#10100000B               ; TYPE B, memory -> SFR
        CHPC20=#(LOW CHAC20)            ; channel pointer
        MPC20P=#C20BUF                  ; macro service pointer
        SFRC20=#(LOW CR21)              ; SFR pointer (dummy)
        MSCC20=#0                       ; macro service counter (256)

        TMC1|=#10000000B                ; start TM2

        while_bit(!CIF20)               ; wait while not occure INTC20
        endw

        CLR1    CIF20                   ; clear INTC20 request flag
        SET1    CISM20                  ; set INTC20 macro service mode
        CLR1    PIF1                    ; clear INTP1 request flag
        CLR1    CMK20                   ; open mask INTC20
        CLR1    PMK1                    ;           INTP1
        RET

;****************************************
;*      INTP1 interrupt routine         *
;*         read capture data            *
;****************************************

INTP1:
        SEL     RB1                     ; select register bank 1
        A=#5
        while(A!=#0)                    ; capture data read wait
                A--
        endw
        A=MSCC20                        ; read higher 8-bit capture data
        A^=#0FFH                        ; reverse higher 8-bit capture data
        (CAPDATP+1)=A                   ; store higher 8-bit capture data
        (CAPDATP+0)=CR22                ; read lower 8-bit capture data
        SET1    CAPENDF                 ; set case of end capture
        MSCC20=#0                       ; set macro service counter
        MPC20P=#C20BUF                  ; set macro service pointer
        RETI

;****************************************
;*      INTC20 end of interrupt routine *
;*         case of timer over flow      *
;****************************************

INTC20:
        SET1    OVFLWF                  ; set over flow flag
        RETI

        END
```

2.4　積分型A/Dコンバータ

ここでは，デュアル・スロープ式積分型A/Dコンバータの応用例を取り上げます。

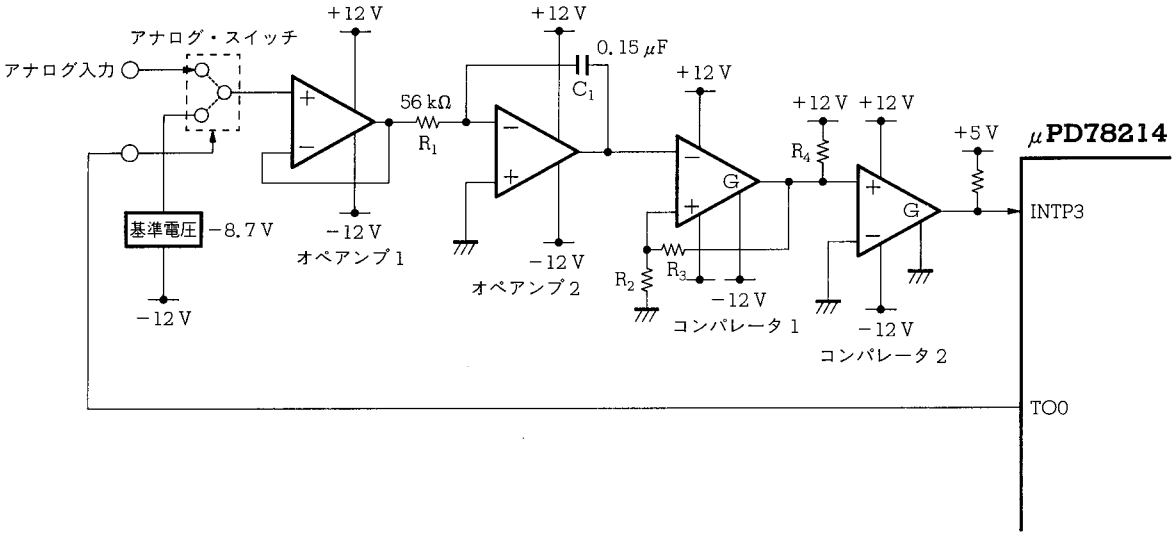
デュアル・スロープ式積分型A/Dコンバータは，一定時間，積分回路で入力電圧を積分（充電）し，この電圧を逆積分（放電）する時間から入力電圧を求める方法です。比較的，変換時間に余裕があり（数ms～数100 ms），高分解能を要求する用途に適しています。

なお，この応用例に掲載しているプログラム例のソース・プログラムは，78K/IIシリーズ　構造化アセンブラ（ST78K2）で記述されています。

(1) 動作概要

外部に図2-9に示すような回路を接続し，16ビット・タイマ／カウンタのタイマ出力TO0とINTP3エッジ割り込みを使用して，12ビット分解能の積分型A/Dコンバータを実現します。

図2-9　積分型A/Dコンバータのブロック図



(a) ハードウエア動作

アナログ入力はアナログ・スイッチに入力されます。また，放電時の基準電圧は，基準電圧発生回路から−8.7 Vがアナログ・スイッチに入力されます。アナログ入力と基準電圧のスイッチの切り替えは，TO0端子からのタイマ出力で行います（TO0＝'H'…基準電圧側，TO0＝'L'…アナログ入力側）。

オペアンプ1に入力された電圧は，ボルテージ・フォロア回路によって，アナログ入力や，基準電圧発生回路の入力インピーダンスの差が，積分時定数に影響しないようにインピーダンス変換され，オペアンプ2の積分回路に入力されます。積分の時定数は，抵抗56 kΩとコンデンサ0.15 μFの積で決定します。

積分された電圧は，コンパレータ1で0 Vと比較されます。ここでは，積分波形が極めて緩やかな場合に，ノイズによってコンパレータの出力にチャタリングが発生するのを防ぐために，R_2，R_3によってヒステリシスを設けています。ここでのスレッシュホールド電圧は，約±0.1 Vとしています。

コンパレータ1は，出力がオープン・コレクタであるため，プルアップ抵抗R_4を出力に付けています。ここでの出力電圧は，おおよそ±10 Vであるため，コンパレータ2で0 V，5 Vの電圧に変換しています。

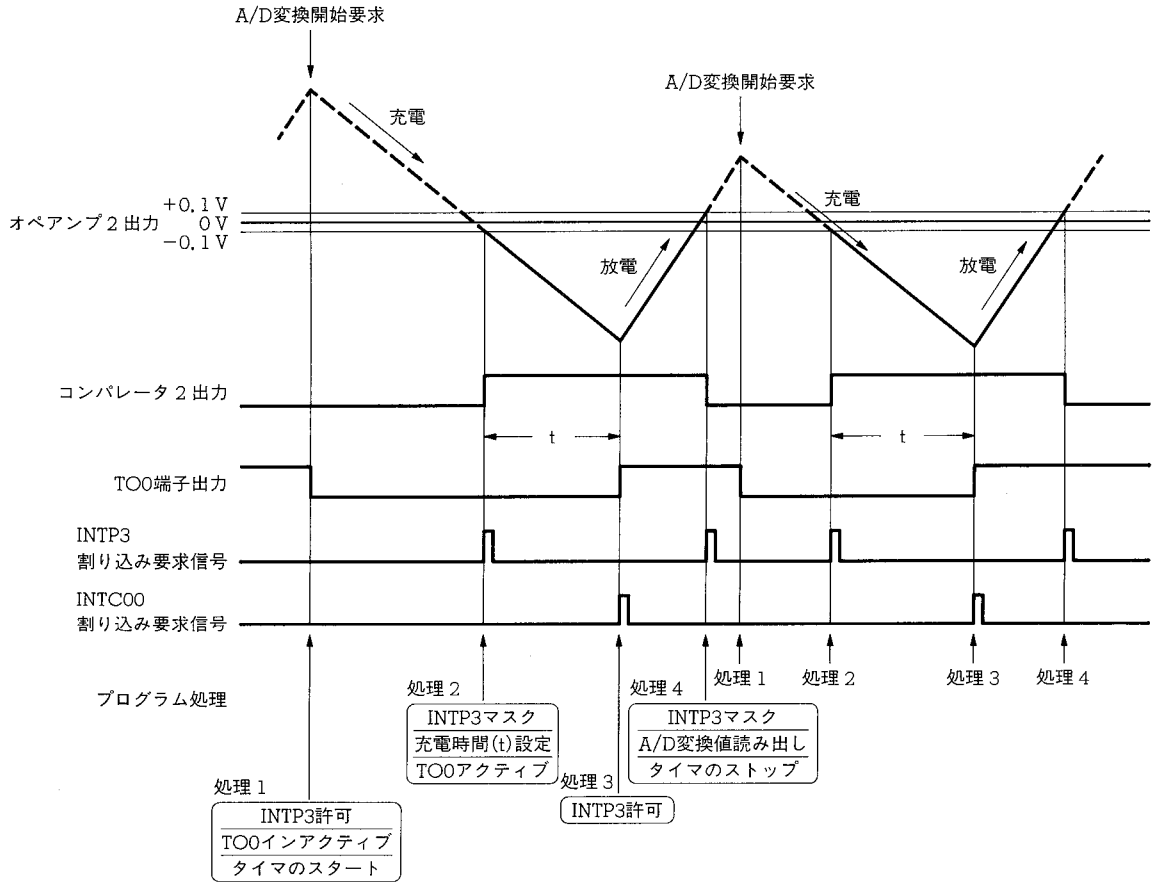
(b) ソフトウエアでの処理

この応用例で使用する内部ハードウエア機能を表2-5に，また，得られる波形のタイミング・チャートを図2-10に示します。

表2-5　積分型A/Dコンバータのプログラムで使用する内部ハードウエアー覧

内部ハードウエア	機　能	用　途
16ビット・タイマ／カウンタ	コンペア（CR00）	リセット後，オペアンプ出力を初期状態にするためのウエイト時間を設定する。
		アナログ・スイッチ切り替えのためのTO0反転を制御する。
	キャプチャ（CR02）	タイマのキャプチャ値により，A/D変換値を決定する。
エッジ割り込み	INTP3 エッジ割り込み （両エッジ）	TM0カウント値をキャプチャするためのトリガ。充電時は約−0.1 V，放電時は約+0.1 Vでコンパレータ出力が反転し，割り込みが発生する。

図2－10　積分型A/Dコンバータのプログラムのタイミング・チャート



A/D変換開始要求で，TO0←'L'とし，タイマ（TM0）のカウントをスタートします（処理1）。TO0←'L'で，アナログ・スイッチはアナログ入力側に切り替わり，充電を開始します。オペアンプ2の出力が約−0.1 Vを通過すると，コンパレータ2の出力が'H'になり，ここでINTP3エッジ割り込みが発生します。この割り込み処理で，充電時間（t）を決定する値をコンペア・レジスタ（CR00）に設定し，タイマ出力を許可します（処理2）。

tの時間だけ充電されると，TO0←'H'となり，アナログ・スイッチは基準電圧側に切り替わり，放電が開始されます。入力電圧に応じた速度で放電され，約+0.1 Vを通過すると，再びINTP3エッジ割り込みが発生します。

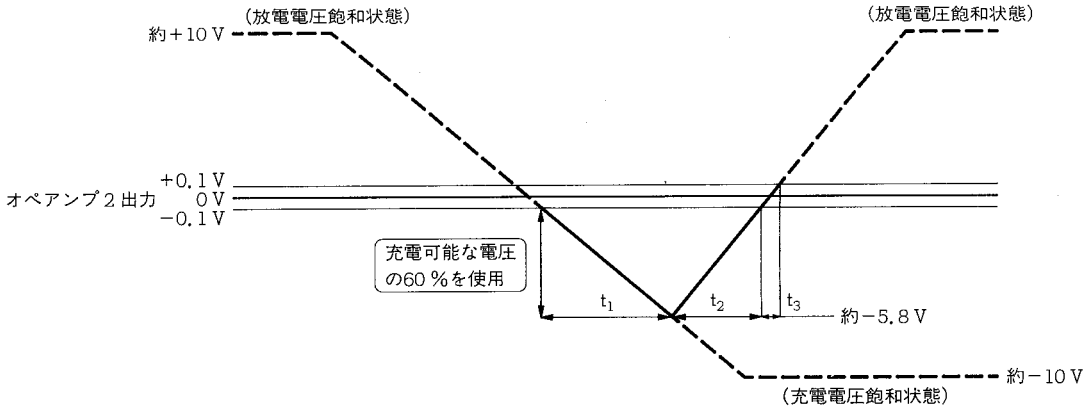
この割り込み処理で，キャプチャ・レジスタ（CR02）の値を読み出すと，A/D変換値を求めることができます(処理4)。ただし，充電時間に対し，ヒステリシス回路による不感帯の電圧幅の約0.2 Vを放電する時間だけ遅れてINTP3が発生しますので，得られた値から遅れ分の値を差し引く必要があります。

なお，この応用例におけるA/D変換時間は，A/D変換開始要求（処理1）から，A/D変換値読み出し（処理4）までの時間となります。

(c) 充電時間と放電時間

図2-9の回路によって出力されるオペアンプ2の波形は，図2-11のようになります。

図2-11　オペアンプ2の出力波形



入力電圧範囲を0 V-5 Vとし，有効な分解能を12ビットとすると，アナログ入力が5 Vの場合のA/D変換値は，$2^{12}=4096$となりますので，図2-11のt_2の時間が，16ビット・タイマ／カウンタの4096カウント分となります。16ビット・タイマ／カウンタのカウント・クロックは，$f_{CLK}/8$ ($f_{CLK}=6$ MHz)ですので，アナログ入力が5 Vの場合の放電時間(t_2)は，次のようになります。

$$t_2 = \frac{8}{f_{CLK}} \times 4096$$
$$\fallingdotseq 5.46\ \text{ms}$$

さらに，アナログ入力（最大）と基準電圧の比は，5 V：8.7 Vですので，充電時間（t_1）は，次のようになります。

$$t_1 = 5.46\ \text{ms} \times 8.7/5$$
$$= 9.5\ \text{ms}$$

この時間でTO0が反転するように，コンペア・レジスタ（CR00）へ値を設定します。

また，この回路では，充電時は約−10 V，放電時は約+10 Vで，電圧が飽和します。この応用例では，飽和するのを防ぐため，9.5 msの充電時間で，充電時の飽和電圧（約−10 V）の約60 %（約−6 V）が充電されるようにしています。

アナログ・スイッチに5 Vの電圧を入力時に，9.5 ms（t_1）の時間で，約−6 Vまで充電する場合の時定数は，次のようになります。

$$V_0 = \frac{V_i t}{C_1 R_1} \quad より，$$

$$|-6\,V| = \frac{5\,V \times 9.5\,ms}{C_1 R_1}$$

$$C_1 R_1 \fallingdotseq 0.0079$$

備考　V_0：オペアンプ2の出力電圧

V_i：オペアンプ2への入力電圧

この値に近くなるように，R_1とC_1（**図2－9**参照）を決定します。

この回路例では，R_1=56 kΩ，C_1=0.15 μFとしており，5 V入力時の充電電圧は，次のようになります。

$$V_0 = -\frac{5 \times 9.5 \times 10^{-3}}{56 \times 10^3 \times 0.15 \times 10^{-6}} + 0.1\,V^{注} \fallingdotseq -5.8\,V$$

注　ヒステリシス回路による一側の不感帯の電圧幅

また，補正用のt_3の時間は，次のようになります。

$$V_0 = \frac{V_i t}{C_1 R_1} \quad より，$$

$$0.2 = \frac{8.7 t_3}{56 \times 10^3 \times 0.15 \times 10^{-6}}$$

$$t_3 \fallingdotseq 1.93\,\mu s$$

ただし，この時間は，コンパレータのヒステリシスおよびC_1R_1の特性などによる誤差によって変動しますので，システムに合わせて確定する必要があります。

(2) プログラム説明

(a) 処理概要（「(8) SPDチャート」参照）

(i) 積分型A/D変換設定処理［レーベル名称：INTG_AD］

① タイマ出力（TO0），INTP3エッジ割り込み，ワーク・エリア（ADRSLTP），ワーク・フラグ（ADENDF，CHRGF）のイニシャライズを行います。タイマは，フリー・ランニングとします。INTP3は，充電時，放電時の両方の場合に発生しますので，両エッジ有効に指定します。

② オペアンプ2の初期出力電圧が，＋側に十分放電されるまでウエイトします。このウエイトは，オペアンプ2の出力が，－側にあるときに，最初のA/D変換が開始され，充電電圧が不正となるのを避けるためです。

(ii) A/D変換開始処理［レーベル名称：ST_ADCNV］

この処理は，ユーザ・プログラムで，A/D変換中でないことをチェックしてから呼び出してください。

① INTP3エッジ割り込みを許可します。このあとINTP3は，充電中に発生することになります。

② TO0にロウ・レベル（アクティブ・ハイの場合のインアクティブ・レベル）を出力します。

③ タイマ（TM0）をスタートします。

(iii) タイマ（TM0）のキャプチャ割り込み処理［レーベル名称：INTP3］

この処理には，充電時と放電時の両方の場合に分岐することになりますので，どちらの場合の処理なのかを判断するために，ワーク・フラグ（CHRGF）を使用しています。

① 自分自身の割り込み（INTP3）をマスクします。これは，充電中，所定の充電時間（9.5 ms）が経過する前に，オペアンプ出力のチャタリングにより，INTP3が発生した場合に，誤って割り込みが受け付けられることのないようにするためです。

② 充電時に発生したINTP3（以降充電モード（CHRGF＝0）と表現します）なのか，放電時に発生したINTP3（以降放電モード（CHRGF＝1）と表現します）なのかをチェックします。

〈充電モードの場合〉

③ 充電時間（次にTO0が反転するまでの時間（9.5 ms））を計るために，コンペア・レジスタ（CR00）へ値を設定します。

④　次に発生するINTP3を，放電モードに設定します（CHRGF←1）。

⑤　INTC00割り込みを許可します。

⑥　タイマ出力（TO0）を許可します。このあと，タイマ・レジスタ（TM0）値とコンペア・レジスタ（CR00）値が一致すると，アクティブ・レベル（ハイ・レベル）が出力されます。

〈放電モードの場合〉

⑦　タイマ（TM0）を停止します。次にA/D変換開始要求があるまで，タイマは停止したままとなります。

⑧　A/D変換値（暫定）を読み出します。A/D変換値（暫定）は，この時点での（キャプチャ・レジスタ（CR02）値）－（コンペア・レジスタ（CR00）値）で求めることができます。

⑨　A/D変換値（暫定）から，ヒステリシス回路による遅れ時間（**(1) 動作概要の(c) 充電時間と放電時間の備考**参照）に相当する値を差し引き，A/D変換値を確定します。

⑩　A/D変換値をメモリ（ADRSLTP）へ格納し，A/D変換終了フラグ（ADENDF）をセット（1）します。

⑪　次にA/D変換開始要求があった場合に発生するINTP3を，充電モードに設定します（CHRGF←0）。

(iv) INTC00タイマ割り込み処理［レーベル名称：INTC00］

INTC00割り込み処理を受け付けなくてもTO0は反転し，アナログ・スイッチは切り替わりますが，この応用例では，充電時のオペアンプ出力のチャタリングによる誤動作を防ぐために，充電モードのINTP3割り込み処理内で，INTP3をマスクしていますので，この処理でINTP3を再び許可する必要があります。

①　INTC00割り込みをマスクをします。

②　INTP3エッジ割り込みを許可します。このあと発生するINTP3は，放電時に発生する割り込みとなります。

(b) 使用するRAM

このプログラム例では，２バイトのワーク・エリアと，２ビットのワーク・フラグを使用します。

表２-６に使用RAM一覧を示します。

表2-6　積分型A/Dコンバータのプログラム例の使用RAM一覧

RAM名称	配置領域	用　　途	バイト数	初期値
ADRSLTP	SADDR注	A/D変換値（12ビット）を格納	2	0000H
ADENDF		A/D変換終了を判断	1ビット	0
CHRGF		充電／放電モードを判断	1ビット	0

注　SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

(3) 入出力パラメータ

(a) 入力パラメータ

なし

(b) 出力パラメータ

表2-7　積分型A/Dコンバータのプログラム例の出力パラメータ

パラメータ	固定／可変	内　　容
ADRSLTP	可変値	A/D変換値
AVENDF	可変値	A/D変換終了フラグ （このプログラムから復帰時は1）
CHRGF	可変値	充電／放電モード判断フラグ （このプログラム内で0および1に変化）

(4) 使用レジスタ

表2-8　積分型A/Dコンバータのプログラム例の使用レジスタ

ルーチン	レーベル	レジスタ	バンク
積分型A/D変換設定処理	INTG_AD	なし	—
A/D変換開始処理	ST_ADCNV	なし	—
タイマのキャプチャ割り込み処理	INTP3	AX	1
INTC00タイマ割り込み処理	INTC00	なし	—

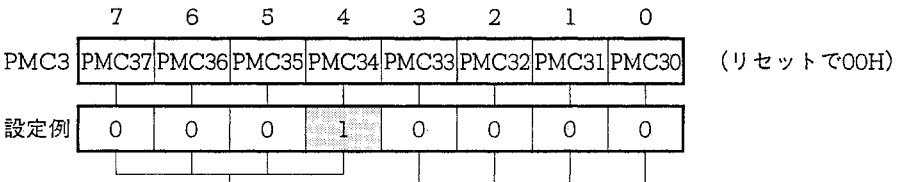
(5) 使用スタック

このプログラム例のサブルーチン，および割り込み処理で使用するスタック・サイズは，2バイト（ネスティング・レベル：1レベル）です。

ただし，A/D変換開始要求を割り込み処理で行った場合は，5バイト（ネスティング・レベル：2レベル）となります。

(6) モード・レジスタ設定例

ポート3モード・コントロール・レジスタ



PMC30	P30端子のコントロール・モード指定
0	入出力ポート・モード
1	RxD入力モード

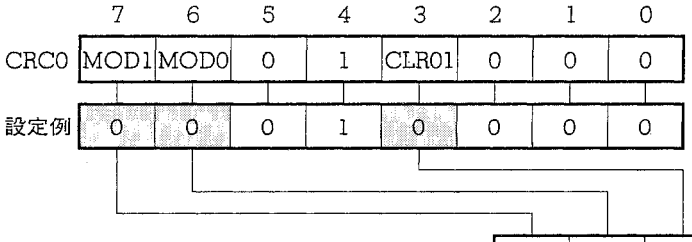
PMC31	P31端子のコントロール・モード指定
0	入出力ポート・モード
1	TxD出力モード

PMC32	P32端子のコントロール・モード指定
0	入出力ポート・モード
1	$\overline{\text{SCK}}$入出力モード

PMC33	P33端子のコントロール・モード指定
0	入出力ポート・モード
1	SO出力/SB0入出力モード

PMC3n	PM3n端子のコントロール・モード指定（n＝4－7）
0	入出力ポート・モード
1	TOm出力モード（m＝0－3）

キャプチャ／コンペア・コントロール・レジスタ0



MOD1	MOD0	CLR01	タイマ出力モード指定 TO0	タイマ出力モード指定 TO1	TM0=CR01時のTM0のクリア動作
0	0	0	トグル出力	トグル出力	禁止
0	0	1	トグル出力	トグル出力	許可
0	1	0	PWM出力	トグル出力	禁止
0	1	1	設定禁止		
1	0	0	PWM出力	PWM出力	禁止
1	0	1	設定禁止		
1	1	0	設定禁止		
1	1	1	PPG出力	トグル出力	許可

214
218A
224
234
244

(7) プログラム使用例

割り込み要求によって，A/D変換開始要求を行う例を示します。ただし，割り込みのイニシャライズ（マスク解除など）は含んでいませんので，処理を追加する必要があります。

また，A/D変換開始要求処理では，タイマ（TM0）が停止していることをチェックして，要求の受け付けの可否を判断しています。

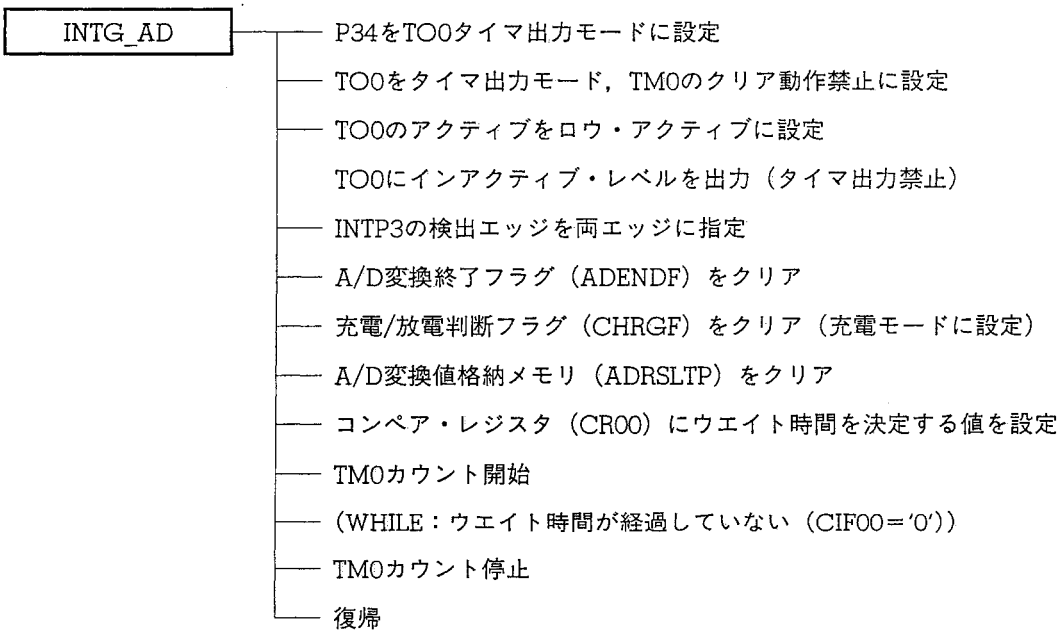
```
        EXTRN   INTG_AD,ST_ADCNV        ; package
        EXTRN   ADRSLTP                 ; work area
        EXTBIT  ADENDF                  ; work flag

MM0     EQU     MM.0                    ; P4 mode setting flag

;       --- define stack area ---

STACKS  DSEG                            ; stack segment
        DS      15
STACK:

;       --- define vector table ---

        RSTVENT                         ; RESET vector table
        PVENT5                          ; INTP5 vector table

;---------------------------------------
;|      main routine                   |
;---------------------------------------

INTGABS CSEG
RST:
INTGAD_B:
        MM=#10000000B                   ; single chip mode
        PM6=#11110000B                  ; memory bank 0
        SP=#STACK                       ; stack pointer

        P4=#0FFH                        ; P4 initial data
        P5=#0FFH                        ; P5 initial data
        SET1    MM0                     ; set P4 output port
        PM5=#00000000B                  ; set P5 output port

        CLR1    PMK5                    ; open INTP5 mask
        CALL    !INTG_AD                ; initialize of A/D conversion
        EI                              ; enable interrupt

        while(forever)                  ; LOOP forever
                if_bit(ADENDF)          ; IF A/D conversion end
                        CLR1    ADENDF  ; THEN clear A/D conversion end flag
                        AX=ADRSLTP      ;      read A/D conversion data
                        A^=#0FFH        ;      reverse higher 8 bit data
                        P4=A            ;      output higher 8 bit data to P4
                        A=#0FFH
                        A^=X            ;      reverse lower 8 bit data
                        P5=A            ;      output lower 8 bit data to P5
                endif
        endw
```

```
;---------------------------------------
;|      request of                     |
;|        starting A/D conversion      |
;|              by INTP5               |
;---------------------------------------

INTP5:
        SEL     RB1                     ; set register bank 1
        A=TMC0
        if_bit(!A.3)                    ; IF not A/D converting ?(timer stop ?)
                CALL    !ST_ADCNV       ; THEN start A/D conversion
        endif
        RETI

        END
                :
                :
        EXTRN   INTG_AD,ST_ADCNV        ; package
        EXTRN   ADRSLTP                 ; work area
        EXTBIT  ADENDF                  ; work flag
                :
                :
        CSEG
MAIN:
                :
                :
        CALL    !INTG_AD                ; initialize of A/D conversion
        EI                              ; enable interrupt
                :
                :
        while(forever)                  ; LOOP forever
                if_bit(ADENDF)          ; IF A/D conversion end
                :
                :
                        AX=ADRSLTP      ;       read A/D conversion data
                :
                :
                endif
        endw
                :
                :
INT:
        A=TMC0
        if_bit(!A.3)                    ; IF not A/D converting ?(timer stop ?)
                CALL    !ST_ADCNV       ; THEN start A/D conversion
        endif
        RETI
                :
                :
```

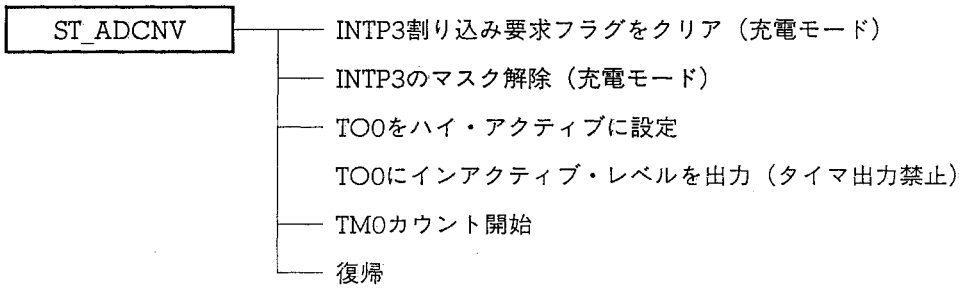
2
214
218A
224
234
244

(8) SPDチャート

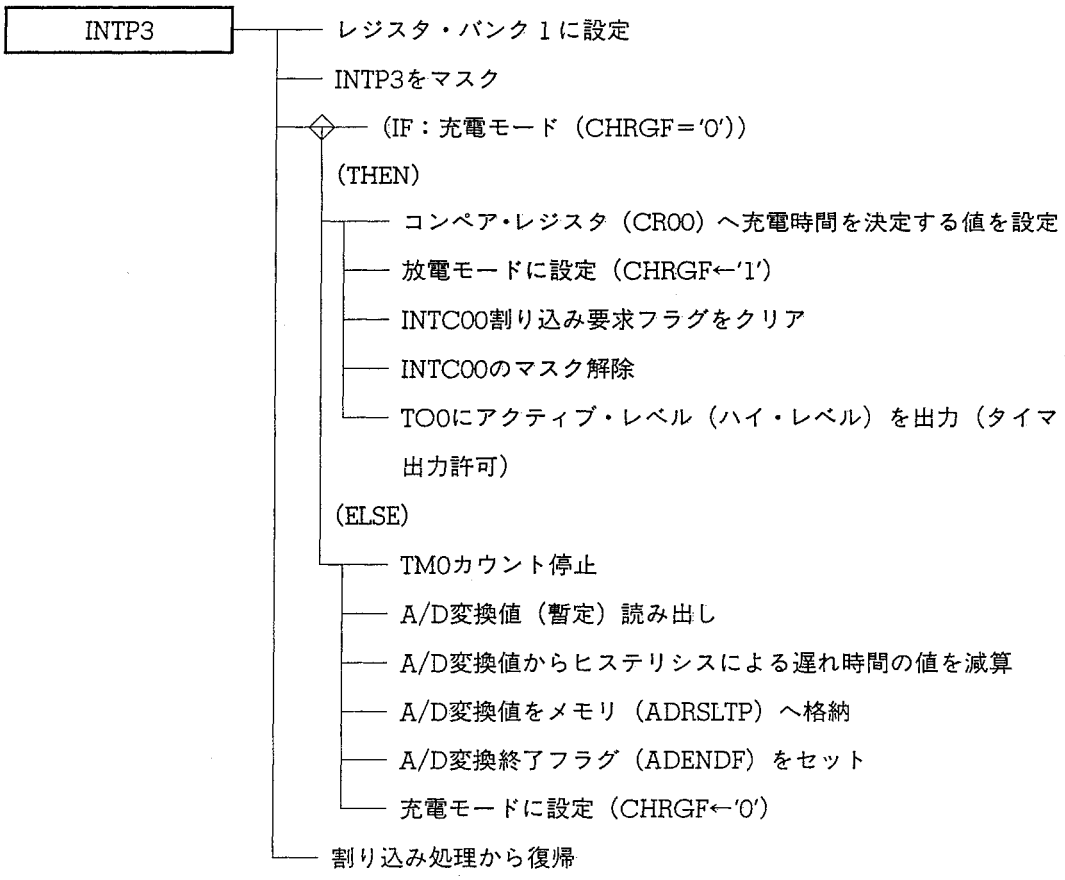
●積分型A/D変換設定処理



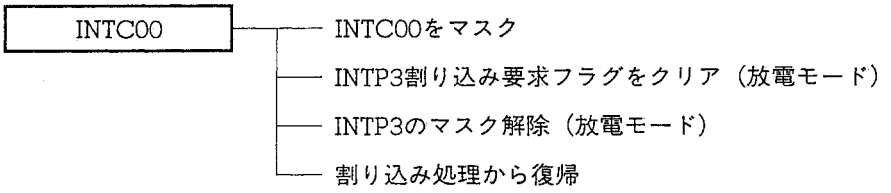
●A/D変換開始処理



●タイマ（TM0）のキャプチャ割り込み処理



●INTC00タイマ割り込み処理



(9) プログラム・リスト

このプログラム例は，構造化アセンブラ用に書かれています。また，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    INTGAD
;*************************************************************
;*      INTEGRATED-TYPE A/D CONVERSION                       *
;*************************************************************

        PUBLIC  INTG_AD,ST_ADCNV; package
        PUBLIC  ADRSLTP         ; work area
        PUBLIC  ADENDF          ; work flag

WAITTIM EQU     18750           ; initial waiting time(25ms)
CHRGTIM EQU     7125            ; charging time(9.5ms)
DLYTIM  EQU     145             ; delay time by hysteresis(193us)

;       *** define work area ***

ADBITS  BSEG
ADENDF  DBIT                    ; A/D conversion end flag
CHRGF   DBIT                    ; charging mode flag

ADWORKS DSEG    SADDR
ADRSLTP:DS      2               ; A/D conversion data store area

;       *** define vector table ***

        PVENT3                  ; INTP3 vector table
        CVENT00                 ; INTC00 vector table

;*****************************************
;*      initialize of                    *
;*      integrated-type A/D conversion   *
;*****************************************

INTGADS CSEG
INTG_AD:
        PMC3=#00010000B         ; set T00 output mode
        CRC0=#00010000B         ; set T00 timer out, disable clear TM0
        TOC=#00000000B          ; set T00 low active, output inactive level
        INTM1=#00000011B        ; set INTP3 edge falling & rising
        CLR1    ADENDF          ; clear A/D conversion end flag
        CLR1    CHRGF           ; clear charging mode flag
        ADRSLTP=#0000H          ; clear A/D conversion data store area
```

```
        CR00=#WAITTIM           ; set initial waiting time
        TMC0=#00001000B         ; start TM0
        while_bit(!CIF00)       ; waiting until output voltage is +side
        endw
        TMC0=#00000000B         ; stop TM0

        RET

;****************************************
;*      start routine of                *
;*              A/D conversion          *
;****************************************

ST_ADCNV:
        CLR1    PIF3            ; clear INTP3 request flag
        CLR1    PMK3            ; open INTP3 mask
        TOC=#00000001B          ; output inactive level(LOW) to TO0
        TMC0=#00001000B         ; start TM0
        RET

;****************************************
;*      capture interrupt by TM0        *
;*              INTP3                   *
;****************************************

INTP3:
        SEL     RB1             ; set register bank 1
        SET1    PMK3            ; mask INTP3
        if_bit(!CHRGF)          ; IF charging mode
                AX=CR02         ; THEN
                AX+=#CHRGTIM    ; set charge time data to CR00
                CR00=AX
                SET1    CHRGF   ; set discharging mode
                CLR1    CIF00   ; clear INTC00 request flag
                CLR1    CMK00   ; open INTC00 mask
                TOC=#00000011B  ; output active level(HIGH) to TO0
        else                    ; ELSE : discharging mode
                TMC0=#00000000B ; stop TM0
                AX=CR02         ; read A/D conversion data
                AX-=CR00
                AX-=#DLYTIM     ; subtract delay time data
                ADRSLTP=AX      ; store A/D conversion data
                SET1    ADENDF  ; set A/D conversion end flag
                CLR1    CHRGF   ; set charging mode
        endif
        RETI
```

```
;****************************************
;*      INTC00 interrupt by match       *
;*             between TM0 & CR00       *
;****************************************

INTC00:
        SET1    CMK00           ; mask INTC00
        CLR1    PIF3            ; clear INTP3 request flag
        CLR1    PMK3            ; open INTP3 mask
        RETI

        END
```

2.5　DTMF信号の生成

8ビット・タイマ／カウンタ2のタイマ出力機能を用いて，DTMF（Dual Tone Multiple Frequency）信号を生成します。

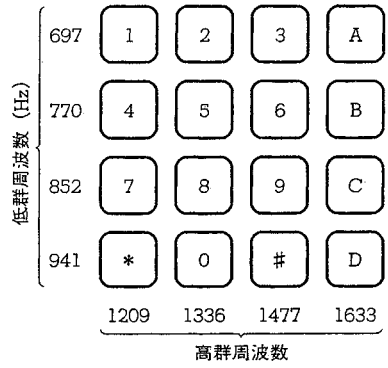
DTMF信号は，高（4種），低（4種）それぞれ1種類ずつの周波数信号を組み合わせることによって生成する16種類の信号です。押しボタン・ダイヤル式の電話機に適用され，16個のキーによって，周波数信号を選択します。

DTMF信号に関して，技術基準で次のように定められています（この応用例に関係のある項目のみ抜粋）。

●周波数

電話機のキー配置と，それぞれのキーに対応する周波数は，図2-12のように定められています。また各周波数の誤差は，±1.5％以内であることが定められています。

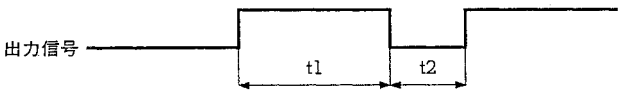
図2-12　キー配置と周波数



●信号送出時間とミニマム・ポーズ

信号送出時間は，周波数信号を出力し続ける時間であり，ミニマム・ポーズは，次の周波数信号を出力するまでの最小の休止時間です。これらの時間は，図2-13のように定められています。

図2-13　信号送出時間とミニマム・ポーズ（ハイ・アクティブの場合）



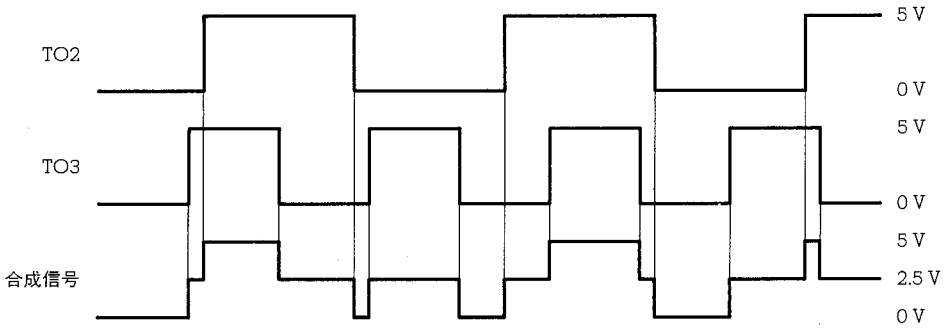
備考　t1：信号送出時間　　…50 ms以上
t2：ミニマム・ポーズ…30 ms以上
t1＋t2　　…120 ms以上

2
214
218A
234
244

2.5.1　動作概要

8ビット・タイマ／カウンタ2のタイマ出力端子TO2より低群（キー配列の縦），TO3より高群（キー配列の横）の周波数信号を出力します。TO2，TO3のタイマ出力を同時に行うと，図2-14のような合成波形（例）が得られます。

図2-14　TO2，TO3タイマ出力の合成波形例



また，このプログラム例では，信号送出時間：60 ms，ポーズ：60 msとします。

(1) 周波数信号の出力

各キーに対応する周波数（低群，高群）から算出した周波数データを，それぞれコンペア・レジスタ（CR20，CR21）に設定します。タイマはフリー・ランニングとし，INTC20，INTC21割り込みごとにCR20，CR21へ，各周波数データを加算していきます。1回のキー入力で，この動作を信号送出時間（60 ms）が経過するまで継続します。

信号送出時間は，高群周波数信号の出力回数（INTC21の発生回数）によってカウントします。低群周波数出力は，高群周波数出力のカウントが終了した時点で停止させます。

カウント方法は次の二通りとし，プログラム例はそれぞれについて示しています。

- 周波数データの設定をINTC20，INTC21ベクタ割り込みで行い，INTC21の発生回数をプログラムでカウントします。
- 周波数データの設定をINTC20，INTC21マクロ・サービスで行い，INTC21マクロ・サービス・カウンタにカウント数を設定します。マクロ・サービスで転送する周波数データは，ROMにテーブルとして持っておきますので，マクロ・サービス・タイプBを使用します。

表2-9に規定周波数とコンペア・レジスタ（CR20，CR21）への設定値，出力回数の対応表を示します。

表2－9　規定周波数とCR20，CR21への設定値，出力回数の対応表

群別	規定周波数	パルス幅	CR2×	カウント	クロック	出力周波数	誤差率
低群	697 Hz	717 μs	135	83	$f_{CLK}/32$	694.4 Hz	−0.4 %
	770 Hz	650 μs	122	92		768.4 Hz	−0.2 %
	852 Hz	587 μs	110	102		852.3 Hz	+0.03 %
	941 Hz	531 μs	100	113		937.5 Hz	−0.4 %
高群	1209 Hz	414 μs	78	144		1201.9 Hz	+0.6 %
	1336 Hz	374 μs	70	161		1339.3 Hz	+0.2 %
	1477 Hz	339 μs	63	179		1488.1 Hz	+0.8 %
	1633 Hz	306 μs	57	197		1644.7 Hz	+0.7 %

備考　CR2×　：低群…CR20，高群…CR21

カウント：60 ms間の出力回数（60 ms/実際の出力パルス幅）

クロック：TM2のカウント・クロック

誤差率　：許容誤差±1.5 %

(2) ポーズ時間のカウント

周波数信号の出力終了後，同じTM2を使用して，ポーズ時間（60 ms）をカウントします。TM2のカウント・クロックは，$f_{CLK}/512$を選択し，割り込みソースはINTC21とします。

INTC21のインターバル時間を10 msとし，これをマクロ・サービス・タイプB（メモリ→SFR）により6回カウントすることによって，60 msの時間を作ります。マクロ・サービスは，カウントのみの目的で使用しますので転送先SFRはダミーとなります（このプログラム例ではCR20）。

TM2とCR21の一致でタイマをクリアするモードに設定し，コンペア・レジスタ（CR21）への設定値は次のようになります。

$$\frac{10\times10^{-3}}{512/(6\times10^{6})}-1\fallingdotseq 116$$

2.5.2　ベクタ割り込みを使用したプログラム例

(1) プログラム説明（「(7) SPDチャート」参照）

(a) 処理概要

(i) イニシャライズ処理［レーベル名称：DTMF_INI］

タイマ出力の設定と割り込みマスクの解除を行います。

(ii) DTMF信号出力開始処理［レーベル名称：ST_DTMF］

① キー・データ（0H-FH：ユーザ・プログラムで生成）に応じて，周波数データ・テーブルより参照した周波数データを，コンペア・レジスタ（CR20，CR21）に初期設定します。

② TM2のイニシャライズ後，タイマをスタートします。

(iii) INTC20割り込み処理［レーベル名称：INTC20］

現在のCR20値に低群周波数データを加算します。

(iv) INTC21割り込み処理［レーベル名称：INTC21］

この処理に入るのは，次の二通りの場合があり，周波数信号出力フラグ(FRQOUTF)によって，2つの状態を判断します。

- 周波数信号の出力中（FRQOUTF＝1）
- ポーズ時間のカウント終了（FRQOUTF＝0）

① FQOUTF＝1 の場合

信号送出時間が経過していなければ(OUTCNT≠0)，現在のCR21値に高群周波数データを加算します。

信号送出時間が経過していれば(OUTCNT＝0)，ポーズ時間のカウントを開始します。タイマ出力を禁止し，TM2のモードをポーズ時間のカウント用に再設定します。また，INTC21マクロ・サービスの初期設定を行います。

② FRQOUTF＝0 の場合

タイマを停止させ，1回のDTMF信号出力を終了します。次の周波数信号の出力が可能となります。

(b) 使用するRAM

このプログラム例では，周波数信号出力用に，4バイトと1ビット，INTC21マクロ・サービス・チャネルとして4バイトのRAMを使用します。

表2－10に使用RAM一覧を示します。

表2－10　DTMF信号の生成（ベクタ割り込み）で使用するRAM一覧

RAM名称	配置領域	用　　途	バイト数	初期値
OUTCNT	SADDR注	信号送出時間のカウント	1	不定
FRQDAT0		低群周波数データの一時保存	1	不定
FRQDAT1		高群周波数データの一時保存	1	不定
PRM1D		TM2のプリスケーラ・データの一時保存	1	不定
FRQOUTF		周波数信号出力中であるかの判断	1ビット	1
SFRC21		INTC21マクロ・サービスSFRポインタ	1	LOW CR20
MRC21P		〃　　ポインタ	2	C21BUF
MSCC21		〃　　カウンタ	1	PCOUNT

注　SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

(2) 入出力パラメータ

(a) 入力パラメータ

表2－11　DTMF信号の生成（ベクタ割り込み）の入力パラメータ

パラメータ	固定／可変	内　　容
KEYDATA	可変値	有効キー・データ

(b) 出力パラメータ

表2－12　DTMF信号の生成（ベクタ割り込み）の出力パラメータ

パラメータ	固定／可変	内　　容
PRM1D	可変値	TM2のプリスケーラ・データ ユーザ・プログラムでTM1を使用する場合に参照。

2
214
218A
234
244

(3) 使用レジスタ

表2－13　DTMF信号の生成（ベクタ割り込み）の使用レジスタ

ルーチン	レーベル	レジスタ	バンク
イニシャライズ処理	DTMF_INI	なし	—
DTMF信号出力開始処理	ST_DTMF	A, B	0
INTC20割り込み処理	INTC20	なし	—
INTC21割り込み処理	INTC21	A	1

(4) 使用スタック

このプログラム例のサブルーチン，および割り込み処理で使用するスタック・サイズは，3バイトです（ネスティング：1レベル）。

(5) モード・レジスタ設定例

(a) 周波数信号出力

ポート3モード・コントロール・レジスタ

	7	6	5	4	3	2	1	0	
PMC3	PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30	(リセットでFFH)
設定例	1	1	×	×	×	×	×	×	×：操作しません

PMC30	P30端子のコントロール・モード指定
0	入出力ポート・モード
1	RxD入力モード

PMC31	P31端子のコントロール・モード指定
0	入出力ポート・モード
1	TxD出力モード

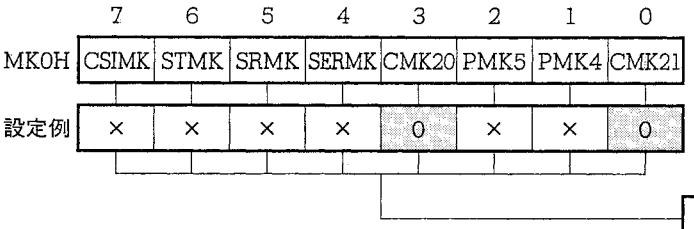
PMC32	P32端子のコントロール・モード指定
0	入出力ポート・モード
1	$\overline{\text{SCK}}$入出力モード

PMC33	P33端子のコントロール・モード指定
0	入出力ポート・モード
1	SO出力／SB0入出力モード

PMC3n	PM3n端子のコントロール・モード指定 (n=4-7)
0	入出力ポート・モード
1	TOm出力モード (m=3)

2
214
218A
234
244

割り込みマスク・レジスタH



MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	0	1	1	×	×	×	×	×：操作しません

8ビット・タイマ／カウンタ1

PRS12	PRS11	PRS10	タイマ／カウンタ1のカウント・クロック周波数の指定
0	0	0	$f_{CLK}/16$[注]
0	0	1	
0	1	0	
0	1	1	$f_{CLK}/32$
1	0	0	$f_{CLK}/64$
1	0	1	$f_{CLK}/128$
1	1	0	$f_{CLK}/256$
1	1	1	$f_{CLK}/512$

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	タイマ／カウンタ3のカウント・クロック周波数の指定
0	0	0	0	$f_{CLK}/16$
0	0	0	1	
0	0	1	0	
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	1	1	1	外部クロック（CI）

注　f_{CLK}：内部システム・クロック周波数（$f_{XX}/2$）

2
214
218A
234
244

キャプチャ／コンペア・コントロール・レジスタ2

	7	6	5	4	3	2	1	0
CRC2	MOD1	MOD0	CLR22	1[注]	CLR21	0	0	0
設定例	0	0	0	1	0	0	0	0

（リセットで00H）

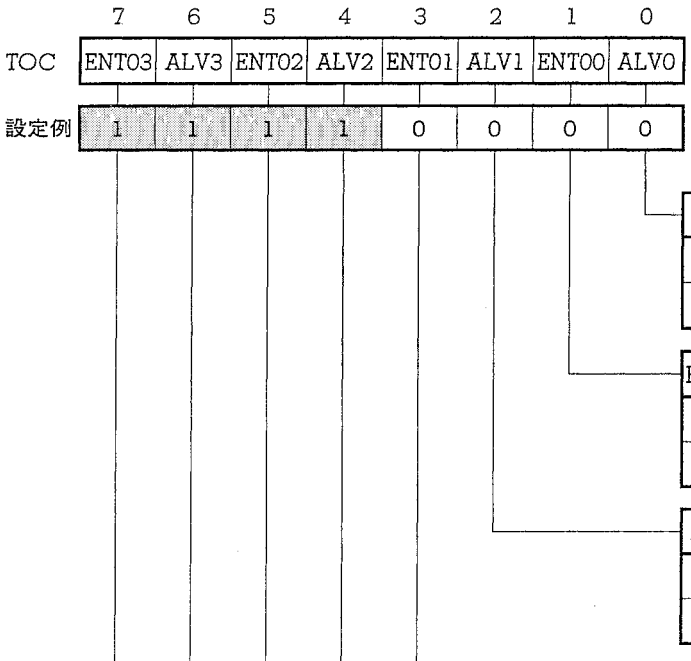
CLR21	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（CR21とTM2が一致したらクリア）

CLR22	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（キャプチャしたらクリア）

MOD1	MOD0	タイマ出力モード指定		
		TO2	TO3	その他条件
0	0	トグル出力	トグル出力	—
0	1	PMW出力	トグル出力	CLR21ビット=0
1	0	PMW出力	PMW出力	CLR22ビット=0
1	1	PPG出力	トグル出力	CLR21ビット=1 CLR22ビット=0

注　必ず“1”に設定してください。

タイマ出力コントロール・レジスタ



ALV0	TO0端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT00	TO0端子の動作指定
0	$\overline{\text{ALV0}}$を出力
1	パルス出力許可

ALV1	TO1端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT01	TO1端子の動作指定
0	$\overline{\text{ALV1}}$を出力
1	パルス出力許可

ALV2	TO2端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT02	TO2端子の動作指定
0	$\overline{\text{ALV2}}$を出力
1	パルス出力許可

ALV3	TO3端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT03	TO3端子の動作指定
0	$\overline{\text{ALV3}}$を出力
1	パルス出力許可

2
214
218A
234
244

タイマ・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	1	×	×	×	×	×	×	×	×：操作しません

8ビット・タイマ／カウンタ1

OVF1	TM1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE1	TM1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

CMD2	TM2のカウント動作モード指定
0	通常モード
1	ワンショット・モード

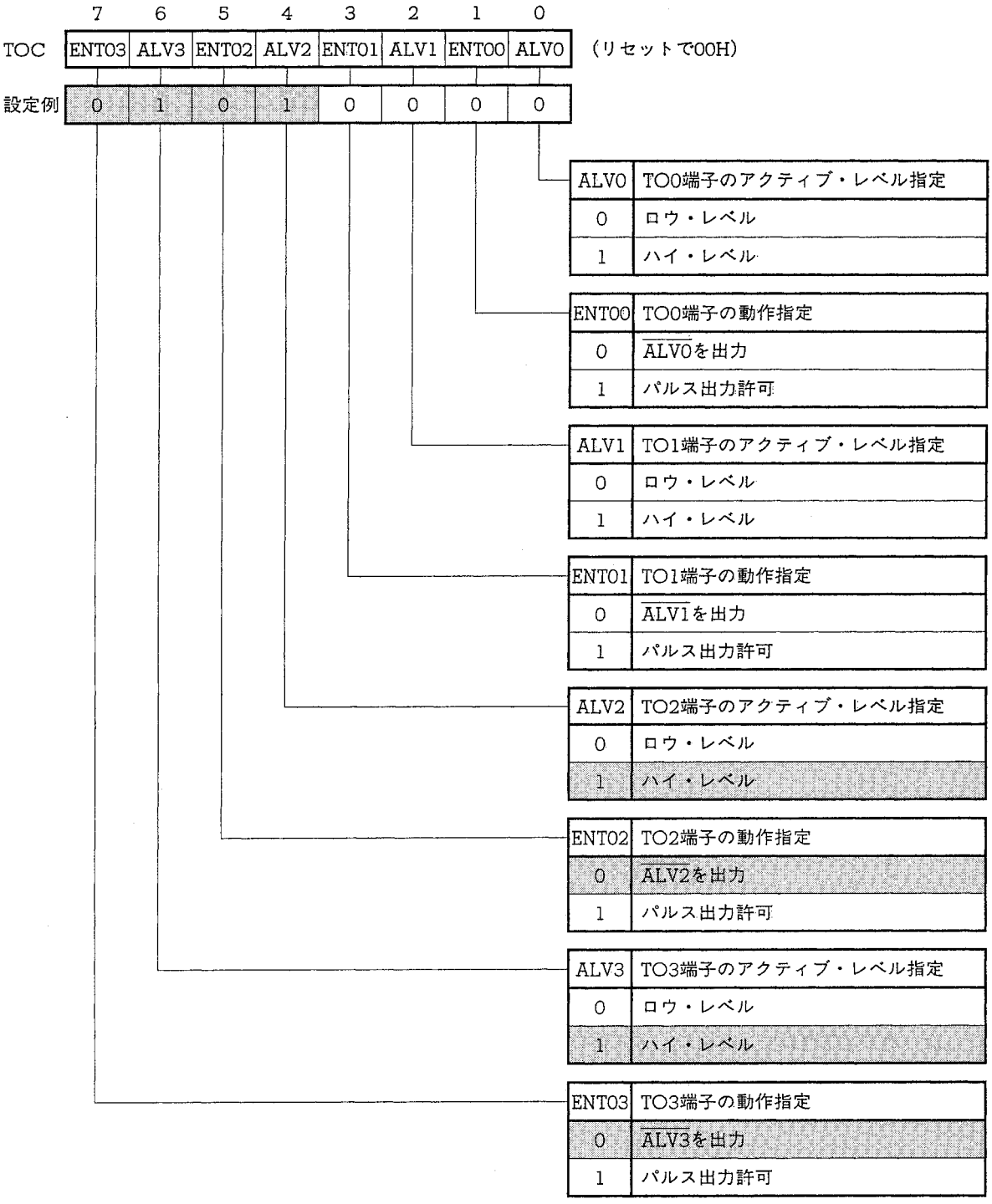
OVF2	TM2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE2	TM2のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

(b) ポーズのカウント

タイマ出力コントロール・レジスタ



	7	6	5	4	3	2	1	0	
TOC	ENT03	ALV3	ENT02	ALV2	ENT01	ALV1	ENT00	ALV0	(リセットで00H)
設定例	0	1	0	1	0	0	0	0	

ALV0	TO0端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT00	TO0端子の動作指定
0	$\overline{\text{ALV0}}$を出力
1	パルス出力許可

ALV1	TO1端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT01	TO1端子の動作指定
0	$\overline{\text{ALV1}}$を出力
1	パルス出力許可

ALV2	TO2端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT02	TO2端子の動作指定
0	$\overline{\text{ALV2}}$を出力
1	パルス出力許可

ALV3	TO3端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT03	TO3端子の動作指定
0	$\overline{\text{ALV3}}$を出力
1	パルス出力許可

2
214
218A
234
244

タイマ・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	1注	×	×	×	×	×	×	×	×：操作しません

8ビット・タイマ／カウンタ1

OVF1	TM1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFHから00Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE1	TM1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

CMD2	TM2のカウント動作モード指定
0	通常モード
1	ワンショット・モード

OVF2	TM2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFHから00Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE2	TM2のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

注　ポーズ時間カウント後に，ソフトウエアでリセット（0）します。

キャプチャ／コンペア・コントロール・レジスタ2

	7	6	5	4	3	2	1	0	
CRC2	MOD1	MOD0	CLR22	1注	CLR21	0	0	0	（リセットで10H）
設定例	0	0	0	1	1	0	0	0	

CLR21	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（CR21とTM2が一致したらクリア）

CLR22	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（キャプチャしたらクリア）

MOD1	MOD0	タイマ出力モード指定		
		TO2	TO3	その他条件
0	0	トグル出力	トグル出力	—
0	1	PWM出力	トグル出力	CLR21ビット=0
1	0	PWM出力	PWM出力	CLR22ビット=0
1	1	PPG出力	トグル出力	CLR21ビット=1 CLR22ビット=0

注　必ず“1”に設定してください。

2
214
218A
234
244

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	1	1	1	×	×	×	×	×：操作しません

8ビット・タイマ／カウンタ1

PRS12	PRS11	PRS10	タイマ／カウンタ1のカウント・クロック周波数の指定
0	0	0	$f_{CLK}/16$注
0	0	1	
0	1	0	
0	1	1	$f_{CLK}/32$
1	0	0	$f_{CLK}/64$
1	0	1	$f_{CLK}/128$
1	1	0	$f_{CLK}/256$
1	1	1	$f_{CLK}/512$

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	タイマ／カウンタ3のカウント・クロック周波数の指定
0	0	0	0	$f_{CLK}/16$
0	0	0	1	
0	0	1	0	
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	1	1	1	外部クロック（CI）

注　f_{CLK}：内部システム・クロック周波数（$f_{XX}/2$）

マクロ・サービス・モード・レジスタ

	7	6	5	4	3	2	1	0
	CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例	1	0	1	0	0	0	0	0

CHT0	0	1	0	1			
CHT1	1	0	1				
CHT2	0	1	1				

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし	データ転送のみ　POL
1	0	0	1				POH
1	0	1	0				自動加算つき　POL
1	0	1	1				POH
1	1	0	0			リング制御つき	データ転送のみ　POL
1	1	0	1				POH
1	1	1	0				自動加算つき　POL
1	1	1	1				POH

割り込みサービス・モード・レジスタH

	7	6	5	4	3	2	1	0	
ISM0H	CSIISM	STISM	SRISM	0	CISM20	PISM5	PISM4	CISM21	（リセットで00H）
設定例	×	×	×	×	×	×	×	1	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

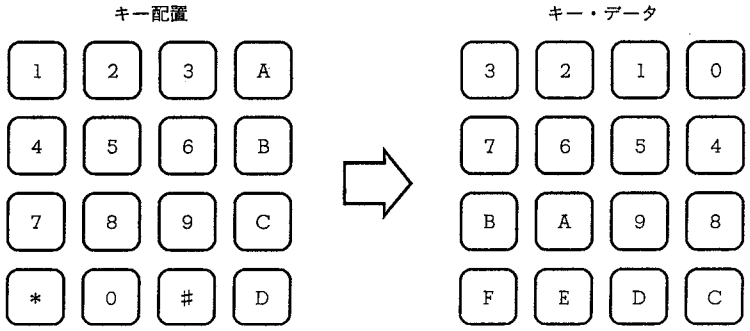
2
214
218A
234
244

(6) プログラム使用例

このプログラム例の使用方法を示します。

周波数信号の出力開始は，TM2のカウントが停止している（信号出力中でない）ときに行ってください。また，キー入力処理はユーザ・プログラムで行い，16個のキーに対して，0H-FHのキー・データに変換してください。キーとキー・データの対応を図2-15に示します。

図2-15　キーとキー・データの対応



有効なキー・データは，ショート・ダイレクト・アドレシング領域のRAM（KEYDATA）に格納し，有効キー・フラグ（KEYOK）をセット(1)してください（**第7章　表示／キー入力**のキー入力ルーチンが使用可能です）。

また，キー・スキャンなどで，タイマ1（TM1）を使用する場合は，プリスケーラ設定の際，TM2の設定を壊さないようRAM（PRM1D）の値と論理演算してから，PRM1レジスタへ値を設定してください。

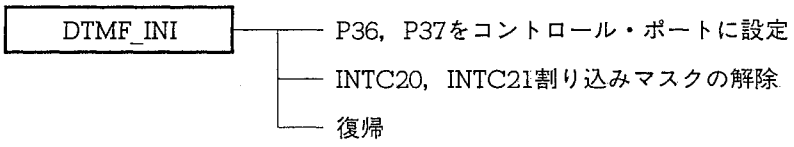
次にプログラム例を示します。

```
EXTRN   DTMF_INI,ST_DTMF        ; package
EXTRN   PRM1D                   ; work area
EXTBIT  KEYOK                   ; work flag
        :
        :
CSEG
        :
        :
CALL    !DTMF_INI               ; initialize of generation DTMF signal
        :
        :
EI
        :
        :
    A=TMC1
    if_bit(!A.7)                ; end counting pause (TM2 is stopped)?
        if_bit(KEYOK)           ; THEN intput key ?
            CLR1    KEYOK
            CALL    !ST_DTMF    ;       start generation of DTMF signal
        :
        :
        endif
    endif
        :
        :
```

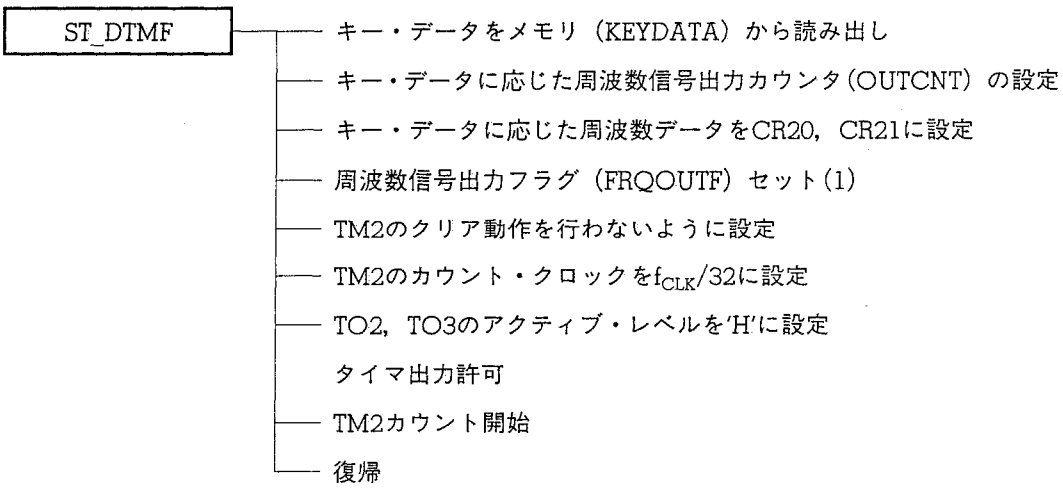

214
218A
234
244

(7) SPDチャート

●イニシャライズ処理

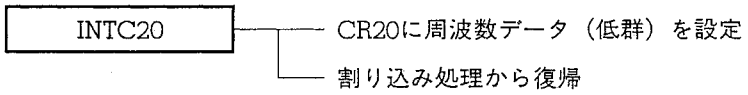


●DTMF信号出力開始処理



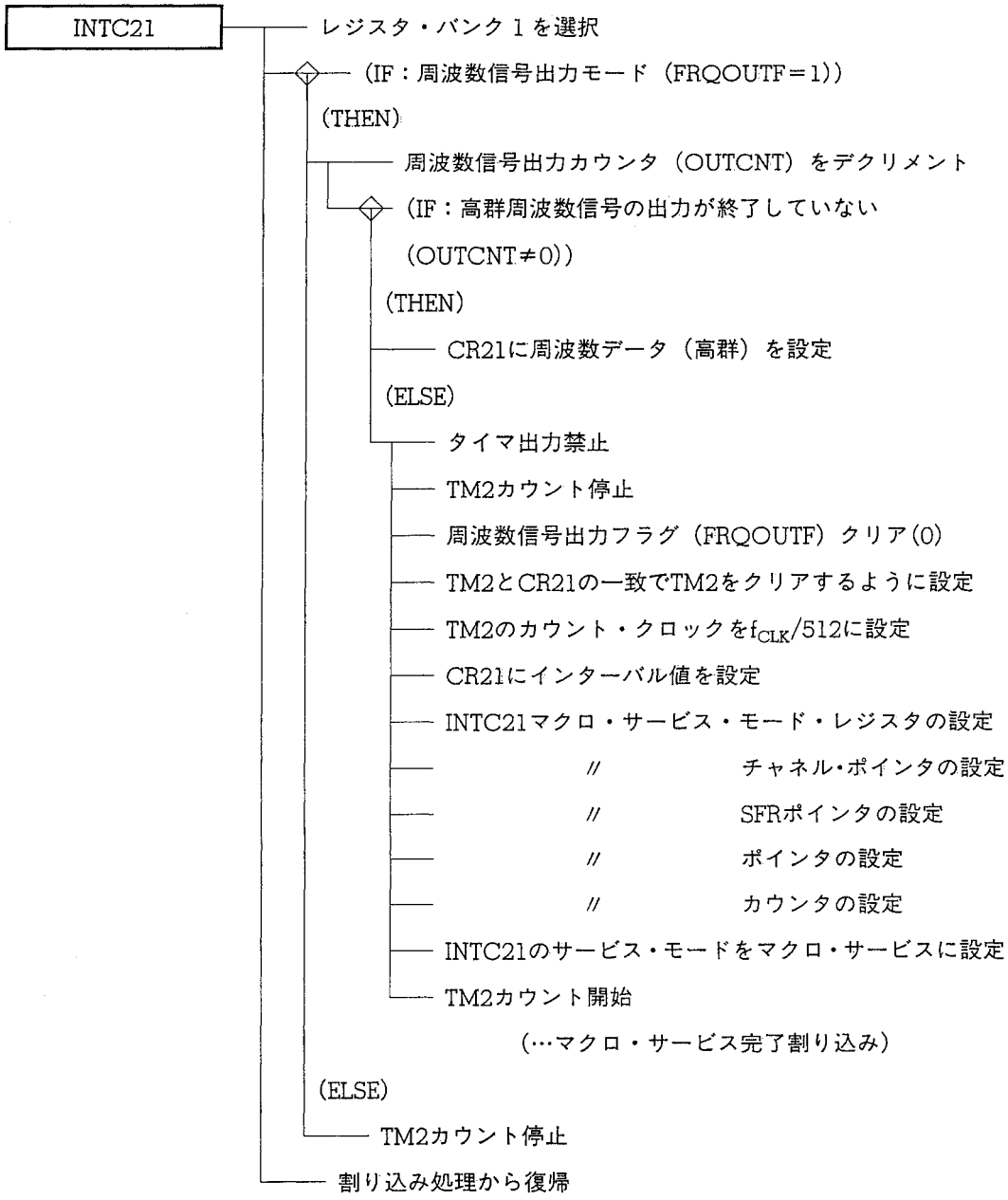
●INTC20割り込み処理

低群周波数信号出力



●**INTC21割り込み処理**

高群周波数信号出力，ポーズのカウント



2
214
218A
234
244

(8) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービス領域の定義用ファイル（INTMS.DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    DTMFV
;*************************************************************
;*      GENERATION OF DTMF SIGNAL                            *
;*              BY VECTOR INTERRUPT                          *
;*************************************************************

        PUBLIC  DTMF_INI,ST_DTMF        ; package
        PUBLIC  PRM1D                   ; work area
        EXTRN   KEYDATA                 ; work area

;       *** define data ***

PCOUNT  EQU     6                       ; pause count
PINTVL  EQU     116                     ; pause interval

;       *** define work area ***

        BSEG
FRQOUTF DBIT                            ; frequency signal output mode flag

FWORKS  DSEG    SADDR
OUTCNT: DS      1                       ; output count area
FRQDAT0:DS      1                       ; lower frequency(INTC20) store area
FRQDAT1:DS      1                       ; higher frequency(INTC21) store area
PRM1D:  DS      1                       ; PRM1 register data store area

;       *** define macro service area ***

C21BUF  EQU     0                       ; dummy data area

        CMCW21                          ; INTC21 macro service contol word
        DSEG_B  C21,'SADDR'             ;        macro service channel

;       *** define vector table **

        CVENT20                         ; INTC20 vector table
        CVENT21                         ; INTC21 vector table
```

```
;****************************************
;*      initialize routine              *
;****************************************

DTMFICS CSEG
DTMF_INI:
        PMC3|=#11000000B                ; set P36,P37 control port
        CLR1    CMK20                   ; open mask INTC20
        CLR1    CMK21                   ;           INTC21
        RET

;****************************************
;*      start DTMF signal output        *
;****************************************

ST_DTMF:
        B=KEYDATA (A)                   ; read key data
        OUTCNT=CNTTBL[B] (A)            ; set output count
        B+=B                            ; count * 2
        FRQDAT0=FRQTBL[B] (A)           ; store frequency data (low)
        CR20=FRQDAT0                    ; set frequency data to CR20
        B++                             ; increment table pointer
        FRQDAT1=FRQTBL[B] (A)           ; store frequency data (high)
        CR21=FRQDAT1                    ; set frequency data to CR21
        SET1    FRQOUTF                 ; set frequency signal output mode
        A=PRM1D
        A&=#00001111B
        A|=#00110000B
        PRM1D=A                         ; save prescaler data
        PRM1=A                          ; set prescaler fclk/32
        CRC2=#00010000B                 ; disable clear TM2
        TOC=#11110000B                  ; enable timer output
        TMC1|=#10000000B                ; start TM2
        RET

;****************************************
;*      INTC20 interrupt routine        *
;*      for lower frequency signal      *
;****************************************

INTC20:
        CR20+=FRQDAT0                   ; set frequency data to CR20
        RETI
```


214

218A

234

244

```
;*************************************
;*      INTC21 interrupt routine      *
;*      for higher frequency signal   *
;*            & counting pause        *
;*************************************

INTC21:
        SEL     RB1                     ; select register bank 1
        if_bit(FRQOUTF)                 ; frequency signal output mode ?
            OUTCNT--                    ; THEN decrement output counter
            if(OUTCNT!=#0) (A)          ;     not end frequency signal output?
                CR21+=FRQDAT1           ;     THEN set frequency data to CR21
            else                        ;     ELSE
                TOC=#01010000B          ;     prohibit timer output
                TMC1&=#01111111B        ;     stop TM2
                CLR1    FRQOUTF         ;     set pause count mode
                CRC2=#00011000B         ;     enable clear TM2 by CR21
                A=PRM1D
                A&=#00001111B
                A|=#01110000B
                PRM1D=A                 ;     save prescaler data
                PRM1=A                  ;     set prescaler fclk/512
                CR21=#PINTVL            ;     set interval data for pause count

                MSMC21=#10100000B       ;     TYPE B, memory -> SFR
                CHPC21=#(LOW CHAC21)    ;     channel pointer
                SFRC21=#(LOW CR20)      ;     SFR pointer
                MPC21P=#C21BUF          ;     macro service pointer
                MSCC21=#PCOUNT          ;     macro service counter
                SET1    CISM21          ;     set INTC21 macro service mode
                TMC1|=#10000000B        ;     start TM2
            endif
        else                            ; ELSE
            TMC1&=#01111111B            ;     stop TM2
        endif
        RETI
```

```
;****************************************
  ;*    frequency data table            *
;****************************************

FRQTBL:
;                LOW  HIGH               ; KEYDATA       KEYNAME
        DB       135, 57                 ;   0             A
        DB       135, 63                 ;   1             3
        DB       135, 70                 ;   2             2
        DB       135, 78                 ;   3             1
        DB       122, 57                 ;   4             B
        DB       122, 63                 ;   5             6
        DB       122, 70                 ;   6             5
        DB       122, 78                 ;   7             4
        DB       110, 57                 ;   8             C
        DB       110, 63                 ;   9             9
        DB       110, 70                 ;   A             8
        DB       110, 78                 ;   B             7
        DB       100, 57                 ;   C             D
        DB       100, 63                 ;   D             #
        DB       100, 70                 ;   E             0
        DB       100, 78                 ;   F             *
CNTTBL:
        DB       197,179,161,144         ; 0,1,2,3
        DB       197,179,161,144         ; 4,5,6,7
        DB       197,179,161,144         ; 8,9,A,B
        DB       197,179,161,144         ; C,D,E,F

;       *** reference of frequency & data ***
;
;       FREQUENCY(Hz)  |  DATA(CR2X)
;      ----------------|-------------
;            697       |    135
;            770       |    122
;            852       |    110
;            941       |    100
;           1209       |     78
;           1336       |     70
;           1477       |     63
;           1633       |     57

        END
```

2
214
218A
234
244

2.5.3　マクロ・サービスを使用したプログラム例

(1) プログラム説明（「(7) SPDチャート」参照）

(a) 処理概要

(i) イニシャライズ処理［レーベル名称：DTMF_INI］

タイマ出力の設定と割り込みマスクの解除を行います。

(ii) DTMF信号出力開始処理［レーベル名称：ST_DTMF］

① キー・データ（0H-FH：ユーザ・プログラムで生成）に応じて，周波数データ・テーブルの先頭アドレスを，INTC20，INTC21マクロ・サービス・ポインタに設定します。

② INTC20，INTC21マクロ・サービスのイニシャライズを行います。

③ TM2のイニシャライズ後，タイマをスタートします。

(iii) INTC20マクロ・サービス完了割り込み処理［レーベル名称：INTC20］

マクロ・サービス・カウンタ（MSCC20）に0を設定しているため，通常は，この割り込みは発生しません。何らかの原因で，MSCC20＝0になったときの暴走回避処理として，RETI命令のみ記述しています。

(iv) INTC21マクロ・サービス完了割り込み処理［レーベル名称：INTC21］

この処理に入るのは，次の二通りの場合があり，周波数信号出力フラグ(FRQOUTF)によって，2つの状態を判断します。

- 周波数信号の出力終了（FRQOUTF＝1）
- ポーズ時間のカウント終了（FRQOUTF＝0）

① FQOUTF＝1 の場合

INTC20マクロ・サービスを停止させ，ポーズ時間のカウントのために，TM2の設定を変更し，ポーズ時間のカウント用にINTC21マクロ・サービスの設定を行います。

② FRQOUTF＝0 の場合

タイマを停止させ，1回のDTMF信号出力を終了します。次の周波数信号の出力が可能となります。

(b) 使用するRAM

このプログラム例では，周波数信号出力用に，1バイトと1ビット，INTC20，INTC21マクロ・サービス・チャネルとして8バイト（4バイト×2）のRAMを使用します。

表2-14に使用RAM一覧を示します。

表2-14　DTMF信号の生成（マクロ・サービス）で使用するRAM一覧

RAM名称	配置領域	用　　途	バイト数	初期値
PRM1D	SADDR注	TM2のプリスケーラ・データの一時保存	1	不定
FRQOUTF		周波数信号出力中であるかの判断	1ビット	1
SFRC20		INTC20マクロ・サービスSFRポインタ	1	LOW CR20
MPC20P		〃　　ポインタ	2	FRQPNT[m]
MSC20P		〃　　カウンタ	1	0
SFRC21		INTC21マクロ・サービスSFRポインタ	1	LOW CR21 LOW CR20
MPC21P		〃　　ポインタ	2	FRQPNT[m+2] C21BUF
MSCC21		〃　　カウンタ	1	CNTTBL[n] PCOUNT

注　SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

n：キー・データ，m：キー・データ×4

上段：周波数信号出力中のINTC21マクロ・サービス

下段：ポーズ・カウント用のINTC21マクロ・サービス

(2) 入出力パラメータ

(a) 入力パラメータ

表2-15　DTMF信号の生成（マクロ・サービス）の入力パラメータ

パラメータ	固定／可変	内　　容
KEYDATA	可変値	有効キー・データ

2
214
218A
234
244

(b) 出力パラメータ

表2－16　DTMF信号の生成（マクロ・サービス）の出力パラメータ

パラメータ	固定／可変	内　　容
PRM1D	可変値	TM2のプリスケーラ・データ ユーザ・プログラムでTM1を使用する場合に参照。

(3) 使用レジスタ

表2－17　DTMF信号の生成（マクロ・サービス）の使用レジスタ

ルーチン	レーベル	レジスタ	バンク
イニシャライズ	DTMF_INI	なし	—
DTMF信号出力開始	ST_DTMF	A, X, B	0
INTC20ベクタ割り込み	INTC20	なし	—
INTC21ベクタ割り込み	INTC21	A	1

(4) 使用スタック

このプログラム例のサブルーチン，および割り込み処理で使用するスタック・サイズは，3バイトです（ネスティング：1レベル）。

(5) モード・レジスタ設定例

(a) 周波数信号出力

ポート3モード・コントロール・レジスタ

	7	6	5	4	3	2	1	0	
PMC3	PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30	(リセットで00H)
設定例	1	1	×	×	×	×	×	×	×：操作しません

PMC30	P30端子のコントロール・モード指定
0	入出力ポート・モード
1	RxD入力モード

PMC31	P31端子のコントロール・モード指定
0	入出力ポート・モード
1	TxD出力モード

PMC32	P32端子のコントロール・モード指定
0	入出力ポート・モード
1	$\overline{\text{SCK}}$入出力モード

PMC33	P33端子のコントロール・モード指定
0	入出力ポート・モード
1	SO出力／SB0入出力モード

PMC3n	PM3n端子のコントロール・モード指定 (n=4-7)
0	入出力ポート・モード
1	TOm出力モード (m=3)

割り込みマスク・レジスタH

	7	6	5	4	3	2	1	0	
MK0H	CSIMK	STMK	SRMK	SERMK	CMK20	PMK5	PMK4	CMK21	(リセットでFFH)
設定例	×	×	×	×	0	×	×	0	×：操作しません

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

2
214
218A
234
244

キャプチャ／コンペア・コントロール・レジスタ2

	7	6	5	4	3	2	1	0	
CRC2	MOD1	MOD0	CLR22	1[注]	CLR21	0	0	0	（リセットで10H）
設定例	0	0	0	1	0	0	0	0	

CLR21	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（CR21とTM2が一致したらクリア）

CLR22	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（キャプチャしたらクリア）

MOD1	MOD0	タイマ出力モード指定		
		TO2	TO3	その他条件
0	0	トグル出力	トグル出力	—
0	1	PWM出力	トグル出力	CLR21ビット=0
1	0	PWM出力	PWM出力	CLR22ビット=0
1	1	PPG出力	トグル出力	CLR21ビット=1 CLR22ビット=0

注　必ず“1”に設定してください。

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	0	1	1	×	×	×	×	×：操作しません

8ビット・タイマ／カウンタ1

PRS12	PRS11	PRS10	タイマ／カウンタ1のカウント・クロック周波数の指定
0	0	0	$f_{CLK}/16$[注]
0	0	1	
0	1	0	
0	1	1	$f_{CLK}/32$
1	0	0	$f_{CLK}/64$
1	0	1	$f_{CLK}/128$
1	1	0	$f_{CLK}/256$
1	1	1	$f_{CLK}/512$

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	タイマ／カウンタ3のカウント・クロック周波数の指定
0	0	0	0	$f_{CLK}/16$
0	0	0	1	
0	0	1	0	
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	1	1	1	外部クロック（CI）

注　f_{CLK}：内部システム・クロック周波数（$f_{XX}/2$）

タイマ出力コントロール・レジスタ

	7	6	5	4	3	2	1	0	
TOC	ENT03	ALV3	ENT02	ALV2	ENT01	ALV1	ENT00	ALV0	（リセットで00H）
設定例	1	1	1	1	0	0	0	0	

ALV0	TO0端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT00	TO0端子の動作指定
0	$\overline{\text{ALV0}}$を出力
1	パルス出力許可

ALV1	TO1端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT01	TO1端子の動作指定
0	$\overline{\text{ALV1}}$を出力
1	パルス出力許可

ALV2	TO2端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT02	TO2端子の動作指定
0	$\overline{\text{ALV2}}$を出力
1	パルス出力許可

ALV3	TO3端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT03	TO3端子の動作指定
0	$\overline{\text{ALV3}}$を出力
1	パルス出力許可

タイマ・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	1	×	×	×	×	×	×	×	×：操作しません

8ビット・タイマ／カウンタ1

OVF1	TM1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE1	TM1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

CMD2	TM2のカウント動作モード指定
0	通常モード
1	ワンショット・モード

OVF2	TM2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE2	TM2のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

2
214
218A
234
244

マクロ・サービス・モード・レジスタ

7	6	5	4	3	2	1	0
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例 1	0	1	0	0	0	0	0

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0		リング制御なし	データ転送のみ	POL
1	0	0	1				POH
1	0	1	0			自動加算つき	POL
1	0	1	1				POH
1	1	0	0		リング制御つき	データ転送のみ	POL
1	1	0	1				POH
1	1	1	0			自動加算つき	POL
1	1	1	1				POH

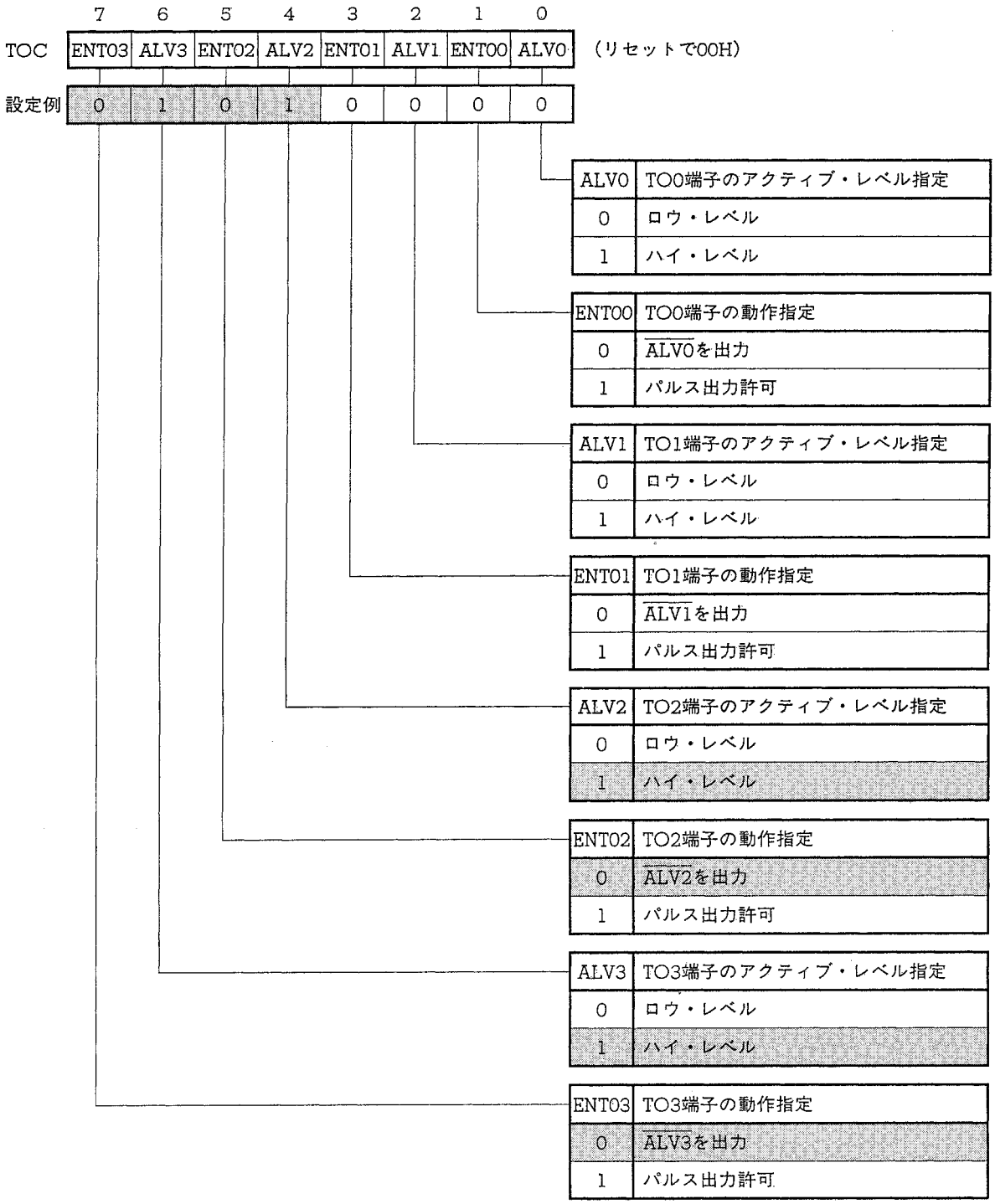
割り込みサービス・モード・レジスタH

7	6	5	4	3	2	1	0	
ISM0H CSIISM	STISM	SRISM	0	CISM20	PISM5	PISM4	CISM21	（リセットで00H）
設定例 ×	×	×	×	1	×	×	1	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

(b) ポーズのカウント

タイマ出力コントロール・レジスタ



	7	6	5	4	3	2	1	0	
TOC	ENT03	ALV3	ENT02	ALV2	ENT01	ALV1	ENT00	ALV0	(リセットで00H)
設定例	0	1	0	1	0	0	0	0	

ALV0	TO0端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT00	TO0端子の動作指定
0	$\overline{\text{ALV0}}$を出力
1	パルス出力許可

ALV1	TO1端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT01	TO1端子の動作指定
0	$\overline{\text{ALV1}}$を出力
1	パルス出力許可

ALV2	TO2端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT02	TO2端子の動作指定
0	$\overline{\text{ALV2}}$を出力
1	パルス出力許可

ALV3	TO3端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT03	TO3端子の動作指定
0	$\overline{\text{ALV3}}$を出力
1	パルス出力許可

2
214
218A
234
244

タイマ・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	1[注]	×	×	×	×	×	×	×	×：操作しません

8ビット・タイマ／カウンタ1

OVF1	TM1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFHから00Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE1	TM1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

CMD2	TM2のカウント動作モード指定
0	通常モード
1	ワンショット・モード

OVF2	TM2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFHから00Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE2	TM2のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

注　ポーズ時間カウント後に，ソフトウエアでリセット（0）します。

割り込みサービス・モード・レジスタH

	7	6	5	4	3	2	1	0	
ISM0H	CSIISM	STISM	SRISM	0	CISM20	PISM5	PISM4	CISM21	（リセットで00H）
設定例	×	×	×	×	0	×	×	1	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

キャプチャ／コンペア・コントロール・レジスタ2

	7	6	5	4	3	2	1	0	
CRC2	MOD1	MOD0	CLR22	1[注]	CLR21	0	0	0	（リセットで10H）
設定例	0	0	0	1	1	0	0	0	

CLR21	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（CR21とTM2が一致したらクリア）

CLR22	タイマ／カウンタ2のクリア動作
0	禁止
1	許可（キャプチャしたらクリア）

MOD1	MOD0	タイマ出力モード指定		
		TO2	TO3	その他条件
0	0	トグル出力	トグル出力	—
0	1	PWM出力	トグル出力	CLR21ビット=0
1	0	PWM出力	PWM出力	CLR22ビット=0
1	1	PPG出力	トグル出力	CLR21ビット=1 CLR22ビット=0

注　必ず“1”に設定してください。

2
214
218A
234
244

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	1	1	1	×	×	×	×	×：操作しません

8ビット・タイマ／カウンタ1

PRS12	PRS11	PRS10	タイマ／カウンタ1のカウント・クロック周波数の指定
0	0	0	$f_{CLK}/16$注
0	0	1	
0	1	0	
0	1	1	$f_{CLK}/32$
1	0	0	$f_{CLK}/64$
1	0	1	$f_{CLK}/128$
1	1	0	$f_{CLK}/256$
1	1	1	$f_{CLK}/512$

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	タイマ／カウンタ3のカウント・クロック周波数の指定
0	0	0	0	$f_{CLK}/16$
0	0	0	1	
0	0	1	0	
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	1	1	1	外部クロック（CI）

注　f_{CLK}：内部システム・クロック周波数（$f_{XX}/2$）

マクロ・サービス・モード・レジスタ

7	6	5	4	3	2	1	0
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例 1	0	1	0	0	0	0	0

CHT0	0	1	0	1			
CHT1	1	0	1				
CHT2	0	1	1				

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC		
						MPTL保持	MPTLインクリメント	
0	0	0	0	メモリからSFRへデータ転送				
0	0	0	1	SFRからメモリへデータ転送				
1	0	0	0			リング制御なし	データ転送のみ	POL
1	0	0	1					POH
1	0	1	0				自動加算つき	POL
1	0	1	1					POH
1	1	0	0			リング制御つき	データ転送のみ	POL
1	1	0	1					POH
1	1	1	0				自動加算つき	POL
1	1	1	1					POH

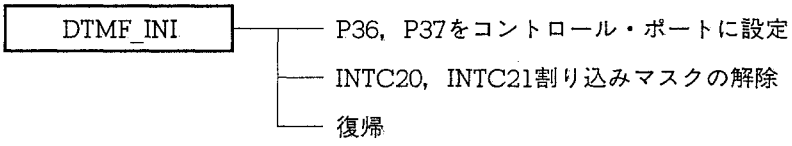
2
214
218A
234
244

(6) プログラム使用例

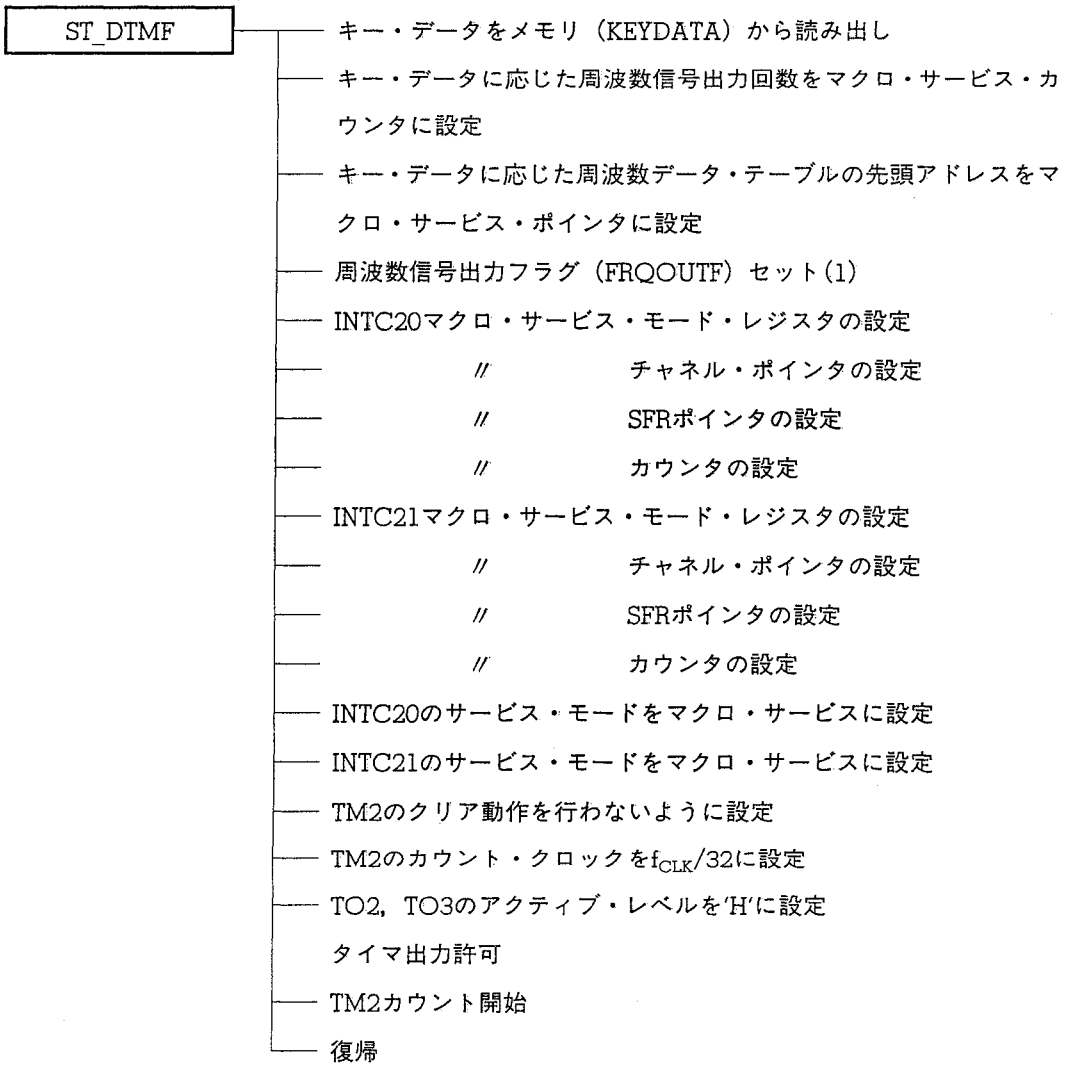
2.5.2　ベクタ割り込みを使用したプログラム例と同様です。

(7) SPDチャート

●イニシャライズ処理



●DTMF信号出力開始処理



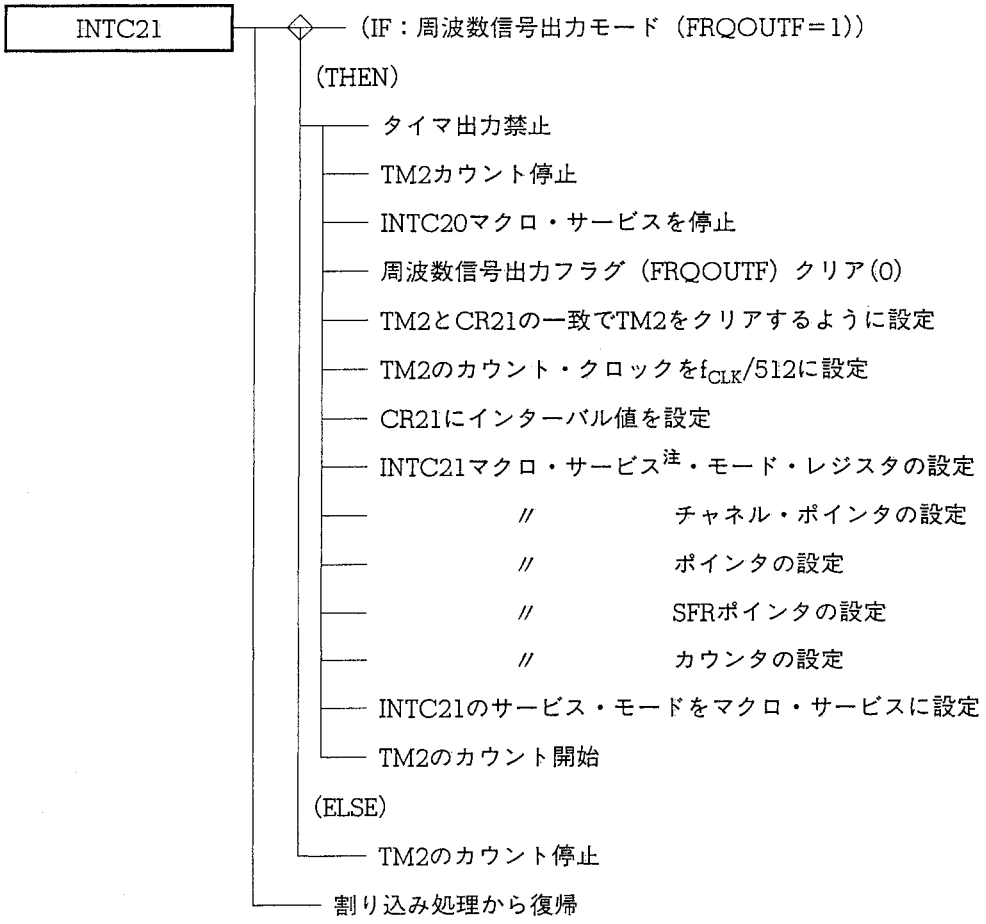
●INTC20マクロ・サービス完了割り込み処理

低群周波数出力



●INTC21マクロ・サービス完了割り込み処理

高群周波数出力，ポーズのカウント



注　ポーズ・カウント用マクロ・サービス

2
214
218A
234
244

(8) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービス領域の定義用ファイル（INTMS.DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    DTMFM
;*************************************************************
;*      GENERATION OF DTMF SIGNAL                            *
;*              BY MACRO SERVICE                             *
;*************************************************************

        PUBLIC  DTMF_INI,ST_DTMF        ; package
        PUBLIC  PRM1D                   ; work area
        EXTRN   KEYDATA                 ; work area

;       *** define data ***

PCOUNT  EQU     6                       ; pause count
PINTVL  EQU     116                     ; pause interval

;       *** define work area ***

        BSEG
FRQOUTF DBIT                            ; frequency signal output mode flag

WORKS   DSEG    SADDR
PRM1D:  DS      1                       ; PRM1 register data store area

;       *** define macro service area ***

C21BUF  EQU     0                       ; dummy data area

        CMCW20                          ; INTC20 macro service control word
        DSEG_B  C20,'SADDR'             ;        macro service channel

        CMCW21                          ; INTC21 macro service control word
        DSEG_B  C21,'SADDR'             ;        macro service channel

;       *** define vector table **

        CVENT20                         ; INTC20 vector table
        CVENT21                         ; INTC21 vector table
```

```
;****************************************
;*      initialize of                   *
;*              generation DTMF         *
;****************************************

DTMFICS CSEG
DTMF_INI:
        PMC3|=#11000000B                ; set P36,P37 control port
        CLR1    CMK20                   ; open mask INTC20
        CLR1    CMK21                   ;           INTC21
        RET

;****************************************
;*      start output DTMF signal        *
;****************************************

ST_DTMF:
        B=KEYDATA (A)                   ; read key data
        MSCC21=CNTTBL[B] (A)            ; MSC <- output count
        B<<=2                           ; count * 4
        X=FRQPNT[B] (A)                 ; refer first address of
        A=(FRQPNT+1)[B]                 ;                 lower frequency data
        MPC20P=AX                       ; set INTC20 macro service pointer
        X=(FRQPNT+2)[B] (A)             ; refer first address of
        A=(FRQPNT+3)[B]                 ;                 higher frequency data
        MPC21P=AX                       ; set INTC21 macro service pointer

        SET1    FRQOUTF                 ; set frequency signal output mode
                                        ; INTC20 INITIALIZE
        MSMC20=#10100000B               ; TYPE B, memory -> SFR
        CHPC20=#(LOW CHAC20)            ; channel pointer
        SFRC20=#(LOW CR20)              ; SFR pointer
        MSCC20=#0                       ; macro service counter
                                        ; INTC21 INITIALIZE
        MSMC21=#10100000B               ; TYPE B, memory -> SFR
        CHPC21=#(LOW CHAC21)            ; channel pointer
        SFRC21=#(LOW CR21)              ; SFR pointer

        SET1    CISM20                  ; set INTC20 macro service mode
        SET1    CISM21                  ; set INTC21 macro service mode

        CR20=#0                         ; first data of compare register
        CR21=#0
        CRC2=#00010000B                 ; disable clear TM2
        A=PRM1D
        A&=#00001111B
        A|=#00110000B
        PRM1D=A                         ; save prescaler data
        PRM1=A                          ; set prescaler fclk/32
        TOC=#11110000B                  ; enable timer output
        TMC1|=#10000000B                ; start TM2
        RET
```

2
214
218A
234
244

```
;*****************************************
;*      end of INTC20 macro service      *
;*      for lower frequency signal       *
;*****************************************

INTC20:
        RETI

;*****************************************
;*      end of INTC21 macro service      *
;*      for higher frequency signal      *
;*              & counting pause         *
;*****************************************

INTC21:
        SEL     RB1                     ; select register bank 1
        if_bit(FRQOUTF)                 ; frequency signal output mode ?
                TOC=#01010000B          ;     prohibit timer output
                TMC1&=#01111111B        ;     stop TM2
                CLR1    CISM20          ;     set INTC20 to vector mode
                CLR1    FRQOUTF         ;     set pause count mode
                CRC2=#00011000B         ;     enable clear TM2 by CR21
                A=PRM1D
                A&=#00001111B
                A|=#01110000B
                PRM1D=A                 ;     save prescaler data
                PRM1=A                  ;     set prescaler fclk/512
                CR21=#PINTVL            ;     set interval data for pause count

                MSMC21=#10100000B       ;     TYPE B, memory -> SFR
                CHPC21=#(LOW CHAC21)    ;     channel pointer
                SFRC21=#(LOW CR20)      ;     SFR pointer
                MPC21P=#C21BUF          ;     macro service pointer
                MSCC21=#PCOUNT          ;     macro service counter
                SET1    CISM21          ;     set INTC21 macro service mode
                TMC1|=#10000000B        ;     start TM2
        else                            ; ELSE
                TMC1&=#01111111B        ;     stop TM2
        endif
        RETI
```

```
;****************************************
;*      frequency data pointer          *
;****************************************

FRQPNT:
;               LOW  HIGH               ; KEYDATA       KEYNAME
        DW      F697,F1633              ;   0             A
        DW      F697,F1477              ;   1             3
        DW      F697,F1336              ;   2             2
        DW      F697,F1209              ;   3             1
        DW      F770,F1633              ;   4             B
        DW      F770,F1477              ;   5             6
        DW      F770,F1336              ;   6             5
        DW      F770,F1209              ;   7             4
        DW      F852,F1633              ;   8             C
        DW      F852,F1477              ;   9             9
        DW      F852,F1336              ;   A             8
        DW      F852,F1209              ;   B             7
        DW      F941,F1633              ;   C             D
        DW      F941,F1477              ;   D             #
        DW      F941,F1336              ;   E             0
        DW      F941,F1209              ;   F             *

CNTTBL:                                 ; KEYDATA
        DB      198,180,162,145         ; 0,1,2,3
        DB      198,180,162,145         ; 4,5,6,7
        DB      198,180,162,145         ; 8,9,A,B
        DB      198,180,162,145         ; C,D,E,F
```

2
214
218A
234
244

```
;*****************************************
;*      frequency data table             *
;*              for macro service        *
;*****************************************

;       --- 697Hz ---
F697:
        DB      135, 14,149, 28,163, 42,177, 56,191, 70
        DB      205, 84,219, 98,233,112,247,126,  5,140
        DB       19,154, 33,168, 47,182, 61,196, 75,210
        DB       89,224,103,238,117,252,131, 10,145, 24
        DB      159, 38,173, 52,187, 66,201, 80,215, 94
        DB      229,108,243,122,  1,136, 15,150, 29,164
        DB       43,178, 57,192, 71,206, 85,220, 99,234
        DB      113,248,127,  6,141, 20,155, 34,169, 48
        DB      183, 62,197, 76

;       --- 770Hz ---
F770:
        DB      122,244,110,232, 98,220, 86,208, 74,196
        DB       62,184, 50,172, 38,160, 26,148, 14,136
        DB        2,124,246,112,234,100,222, 88,210, 76
        DB      198, 64,186, 52,174, 40,162, 28,150, 16
        DB      138,  4,126,248,114,236,102,224, 90,212
        DB       78,200, 66,188, 54,176, 42,164, 30,152
        DB       18,140,  6,128,250,116,238,104,226, 92
        DB      214, 80,202, 68,190, 56,178, 44,166, 32
        DB      154, 20,142,  8,130,252,118,240,106,228
        DB       94,216

;       --- 852Hz ---
F852:
        DB      110,220, 74,184, 38,148,  2,112,222, 76
        DB      186, 40,150,  4,114,224, 78,188, 42,152
        DB        6,116,226, 80,190, 44,154,  8,118,228
        DB       82,192, 46,156, 10,120,230, 84,194, 48
        DB      158, 12,122,232, 86,196, 50,160, 14,124
        DB      234, 88,198, 52,162, 16,126,236, 90,200
        DB       54,164, 18,128,238, 92,202, 56,166, 20
        DB      130,240, 94,204, 58,168, 22,132,242, 96
        DB      206, 60,170, 24,134,244, 98,208, 62,172
        DB       26,136,246,100,210, 64,174, 28,138,248
        DB      102,212
```

```
;       --- 941Hz ---
F941:
        DB      100,200,  44,144,244,  88,188,  32,132,232
        DB       76,176,  20,120,220,  64,164,   8,108,208
        DB       52,152,252,  96,196,  40,140,240,  84,184
        DB       28,128,228,  72,172,  16,116,216,  60,160
        DB        4,104,204,  48,148,248,  92,192,  36,136
        DB      236,  80,180,  24,124,224,  68,168,  12,112
        DB      212,  56,156,   0,100,200,  44,144,244,  88
        DB      188,  32,132,232,  76,176,  20,120,220,  64
        DB      164,   8,108,208,  52,152,252,  96,196,  40
        DB      140,240,  84,184,  28,128,228,  72,172,  16
        DB      116,216,  60,160,   4,104,204,  48,148,248
        DB       92,192,  36

;       --- 1209Hz ---
F1209:
        DB       78,156,234,  56,134,212,  34,112,190,  12
        DB       90,168,246,  68,146,224,  46,124,202,  24
        DB      102,180,   2,  80,158,236,  58,136,214,  36
        DB      114,192,  14,  92,170,248,  70,148,226,  48
        DB      126,204,  26,104,182,   4,  82,160,238,  60
        DB      138,216,  38,116,194,  16,  94,172,250,  72
        DB      150,228,  50,128,206,  28,106,184,   6,  84
        DB      162,240,  62,140,218,  40,118,196,  18,  96
        DB      174,252,  74,152,230,  52,130,208,  30,108
        DB      186,   8,  86,164,242,  64,142,220,  42,120
        DB      198,  20,  98,176,254,  76,154,232,  54,132
        DB      210,  32,110,188,  10,  88,166,244,  66,144
        DB      222,  44,122,200,  22,100,178,   0,  78,156
        DB      234,  56,134,212,  34,112,190,  12,  90,168
        DB      246,  68,146,224
```

```
;       --- 1336Hz ---
F1336:
        DB       70,140,210,  24,  94,164,234,  48,118,188
        DB        2,  72,142,212,  26,  96,166,236,  50,120
        DB      190,   4,  74,144,214,  28,  98,168,238,  52
        DB      122,192,   6,  76,146,216,  30,100,170,240
        DB       54,124,194,   8,  78,148,218,  32,102,172
        DB      242,  56,126,196,  10,  80,150,220,  34,104
        DB      174,244,  58,128,198,  12,  82,152,222,  36
        DB      106,176,246,  60,130,200,  14,  84,154,224
        DB       38,108,178,248,  62,132,202,  16,  86,156
        DB      226,  40,110,180,250,  64,134,204,  18,  88
        DB      158,228,  42,112,182,252,  66,136,206,  20
        DB       90,160,230,  44,114,184,254,  68,138,208
        DB       22,  92,162,232,  46,116,186,   0,  70,140
        DB      210,  24,  94,164,234,  48,118,188,   2,  72
        DB      142,212,  26,  96,166,236,  50,120,190,   4
        DB       74,144,214,  28,  98,168,238,  52,122,192
        DB        6

;       --- 1477Hz ---
F1477:
        DB       63,126,189,252,  59,122,185,248,  55,118
        DB      181,244,  51,114,177,240,  47,110,173,236
        DB       43,106,169,232,  39,102,165,228,  35,  98
        DB      161,224,  31,  94,157,220,  27,  90,153,216
        DB       23,  86,149,212,  19,  82,145,208,  15,  78
        DB      141,204,  11,  74,137,200,   7,  70,133,196
        DB        3,  66,129,192,255,  62,125,188,251,  58
        DB      121,184,247,  54,117,180,243,  50,113,176
        DB      239,  46,109,172,235,  42,105,168,231,  38
        DB      101,164,227,  34,  97,160,223,  30,  93,156
        DB      219,  26,  89,152,215,  22,  85,148,211,  18
        DB       81,144,207,  14,  77,140,203,  10,  73,136
        DB      199,   6,  69,132,195,   2,  65,128,191,254
        DB       61,124,187,250,  57,120,183,246,  53,116
        DB      179,242,  49,112,175,238,  45,108,171,234
        DB       41,104,167,230,  37,100,163,226,  33,  96
        DB      159,222,  29,  92,155,218,  25,  88,151,214
        DB       21,  84,147,210,  17,  80,143,206,  13
```

```
;       --- 1633Hz ---
F1633:
        DB       57,114,171,228,  29,  86,143,200,   1,  58
        DB      115,172,229,  30,  87,144,201,   2,  59,116
        DB      173,230,  31,  88,145,202,   3,  60,117,174
        DB      231,  32,  89,146,203,   4,  61,118,175,232
        DB       33,  90,147,204,   5,  62,119,176,233,  34
        DB       91,148,205,   6,  63,120,177,234,  35,  92
        DB      149,206,   7,  64,121,178,235,  36,  93,150
        DB      207,   8,  65,122,179,236,  37,  94,151,208
        DB        9,  66,123,180,237,  38,  95,152,209,  10
        DB       67,124,181,238,  39,  96,153,210,  11,  68
        DB      125,182,239,  40,  97,154,211,  12,  69,126
        DB      183,240,  41,  98,155,212,  13,  70,127,184
        DB      241,  42,  99,156,213,  14,  71,128,185,242
        DB       43,100,157,214,  15,  72,129,186,243,  44
        DB      101,158,215,  16,  73,130,187,244,  45,102
        DB      159,216,  17,  74,131,188,245,  46,103,160
        DB      217,  18,  75,132,189,246,  47,104,161,218
        DB       19,  76,133,190,247,  48,105,162,219,  20
        DB       77,134,191,248,  49,106,163,220,  21,  78
        DB      135,192,249,  50,107,164,221

        END
```


214
218A
234
244

第3章　マクロ・サービスの応用例

マクロ・サービスの応用例として，次の項目のプログラム例を示します。

1. プログラマブル・イベント・カウンタ
2. パルス周期の測定
3. アナログ波形出力
4. 階段波形出力によるA/D変換

なおこの応用例は，すべて78K/IIシリーズ　構造化アセンブラ（ST78K/II）にのっとって，ソース・プログラムを記述しています。

3.1　プログラマブル・イベント・カウンタ

外部で発生したイベントを，マクロ・サービスによりカウントする例を示します。

指定回数のカウント終了時にしかベクタ割り込みが発生しませんので，イベントのカウントのみを行いたいときに有効です。

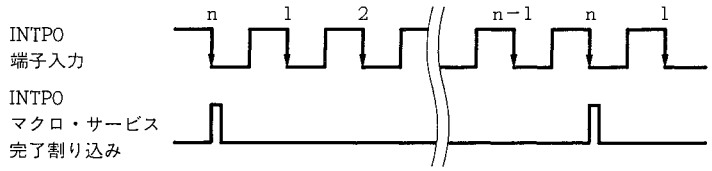
この応用例は，μPD78224シリーズの場合，INTP0端子をINTP4端子に変更することで使用することができます。また，μPD78224シリーズを除く他のシリーズでは，INTP0端子だけでなく，他の外部割り込み入力端子でも使用できます。

(1) 動作概要

INTP0端子に入力されるパルスの有効エッジを，マクロ・サービスによってn回カウントします。

n回カウント終了で発生するマクロ・サービス完了割り込み処理で，マクロ・サービスを再起動し，カウント動作を連続的に行います。図3-1に，INTP0端子入力とマクロ・サービス完了割り込みのタイミング・チャートを示します。

図3-1　INTP0端子入力とマクロ・サービス完了割り込みのタイミング・チャート

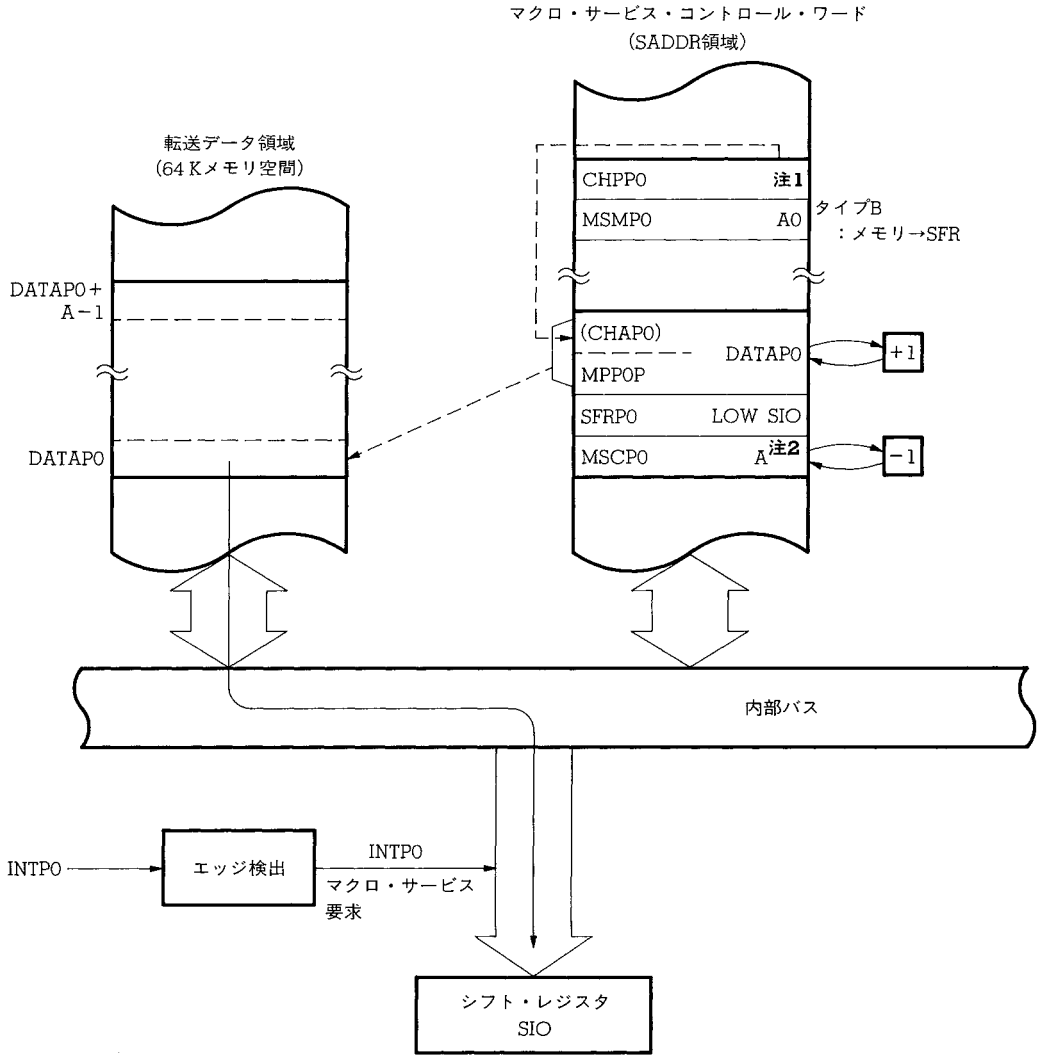


備考　INTP0立ち下がりエッジ有効の場合

注意　INTP0端子入力の周期は，片エッジ有効の場合，1回のマクロ・サービス処理時間（タイプB）以上となるようにしてください。

マクロ・サービスは，タイプB，メモリ→SFR（SIO）を使用しますが，この応用例では，パルスをカウントするだけが目的ですので，転送動作は意味を持ちません（マクロ・サービス・カウンタのみが意味を持ちます）。したがって，転送先SFRは，使用システムに合わせ，書き込んでも問題のないSFR（システムで使用しない書き込み可能なSFR）を選択してください。また，転送データは，どんなデータでもかまいません。この応用例では，マクロ・サービス・ポインタを0H番地に設定しています。図3-2に，マクロ・サービスのメモリ・マップ，および動作ブロック図を示します。

図3-2　プログラマブル・イベント・カウンタのマクロ・サービスのメモリ・マップと動作ブロック図



注1.　CHAP0（=MPP0P+1）の下位バイト

2.　Aレジスタにデータを設定

(2) プログラム説明

(a) 処理概要

(i) メイン処理［レーベル名称：EVNTCNT］

マクロ・サービスのイニシャライズを行います。詳細は，図3-2　マクロ・サービスのメモリ・マップ，(8) SPDチャートを参照してください。

(ii) マクロ・サービス再設定処理［レーベル名称：MACEND］

マクロ・サービスの再設定を行います。

(b) 使用するRAM

このプログラム例では，マクロ・サービス・チャネルとして，4バイトのRAMを使用します。表3-1に使用RAM一覧を示します。

表3-1　プログラマブル・イベント・カウンタの使用RAM一覧

RAM名称	配置領域	用　　途	バイト数	初期値
MPP0P	SADDR[注1]	INTP0マクロ・サービス・ポインタ	2	DATAP0
SFRP0		INTP0マクロ・サービスSFRポインタ	1	LOW SIO
MSCP0		INTP0マクロ・サービス・カウンタ	1	A[注2]

注1. SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

2. Aレジスタ値を設定します。

(3) 入出力パラメータ

(a) 入力パラメータ

表3-2　プログラマブル・イベント・カウンタの入力パラメータ

パラメータ	固定／可変	内　　容
Aレジスタ（バンク0）	可変値	イベント・カウント数

(b) 出力パラメータ

なし

(4) 使用レジスタ

表3-3　プログラマブル・イベント・カウンタの使用レジスター覧

ルーチン	レーベル	レジスタ	バンク
メイン	EVNTCNT	A	0
マクロ・サービス再設定	MACEND	A	0

(5) 使用スタック

このプログラム例の，サブルーチンおよび割り込み処理で使用するスタック・サイズは，5バイトです（ネスティング：2レベル）。

(6) モード・レジスタ設定例

(a) プログラム使用例でのモード・レジスタ

外部割り込みモード・レジスタ0

	7	6	5	4	3	2	1	0	
INTM0	ES21	ES20	ES11	ES10	ES01	ES00	0	ESNMI	(リセットで00H)
設定例	0	0	0	0	0	0	0	0	

ESNMI	NMI端子入力，検出エッジ指定
0	立ち下がりエッジ
1	立ち上がりエッジ

ES01	ES00	INTP0端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES11	ES10	INTP1端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES21	ES20	INTP2端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

(b) プログラム・リストでのモード・レジスタ

マクロ・サービス・モード・レジスタ（INTP0）

7	6	5	4	3	2	1	0
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例 1	0	1	0	0	0	0	0

CHT0	0	1	0	1			
CHT1	1	0	1				
CHT2	0	1	1				
MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし	データ転送のみ　POL
1	0	0	1				POH
1	0	1	0				自動加算つき　POL
1	0	1	1				POH
1	1	0	0			リング制御つき	データ転送のみ　POL
1	1	0	1				POH
1	1	1	0				自動加算つき　POL
1	1	1	1				POH

3
214
218A
224
234
244

割り込みサービス・モード・レジスタL

	7	6	5	4	3	2	1	0	
ISM0L	CISM11	CISM10	CISM01	CISM00	PISM3	PISM2	PISM1	PISM0	（リセットで00H）
設定例	×	×	×	×	×	×	×	1	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

割り込みマスク・レジスタL

	7	6	5	4	3	2	1	0	
MK0L	CMK11	CMK10	CMK01	CMK00	PMK3	PMK2	PMK1	PMK0	（リセットでFFH）
設定例	×	×	×	×	×	×	×	0	×：操作しません

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

(7) プログラム使用例

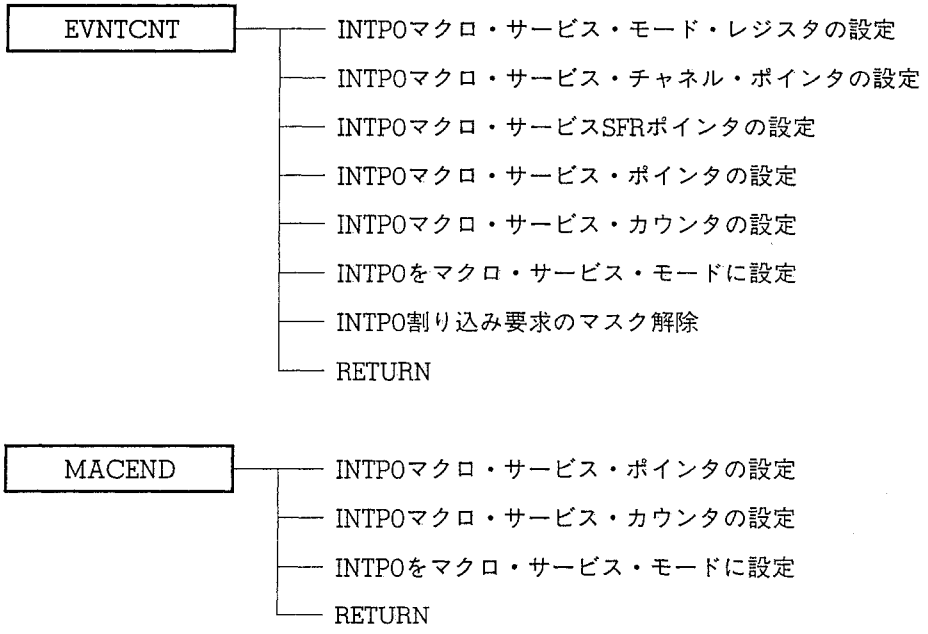
このプログラムでは，イベント・カウント数(n)を0-255（0：n＝256）の値に，任意に変更できます。また，イベント・カウント終了で何か処理を行う場合は，マクロ・サービス完了割り込み処理内の，マクロ・サービスの再設定処理（MACEND）以前に行ってください。

次にn＝5とした場合の使用例を示します。

```
$       IC(INTMS.DEF)
                :
                :
        EXTRN   EVNTCNT,MACEND          ; package

CNTNUM  EQU     5                       ; event count number
                :
                :
        PVENT0                          ; INTP0 vector table
                :
                :
        CSEG
                :
                :
        MOV     INTM0,#00000000B        ; set falling edge of INTP0
        MOV     A,#CNTNUM               ; set event count number
        CALL    !EVNTCNT                ; call EVNTCNT package
        EI
                :
                :
;       ***** end of INTP0 macro service interrupt routine *****

INTP0:
                :
                :
        MOV     A,#CNTNUM               ; set event count number
        CALL    !MACEND                 ; end of macro service
        RETI
                :
                :
```

3
214
218A
224
234
244

(8) SPDチャート


EVNTCNT
INTP0マクロ・サービス・モード・レジスタの設定
INTP0マクロ・サービス・チャネル・ポインタの設定
INTP0マクロ・サービスSFRポインタの設定
INTP0マクロ・サービス・ポインタの設定
INTP0マクロ・サービス・カウンタの設定
INTP0をマクロ・サービス・モードに設定
INTP0割り込み要求のマスク解除
RETURN
MACEND
INTP0マクロ・サービス・ポインタの設定
INTP0マクロ・サービス・カウンタの設定
INTP0をマクロ・サービス・モードに設定
RETURN

(9) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，マクロ・サービスの領域定義用ファイル（INTMS.DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    EVNT
;*************************************************************
;*      APPLICATION OF MACRO SERVICE                         *
;*      < PROGRAMABLE EVENT COUNTER >                        *
;*              INPUT : A = event count                      *
;*************************************************************

        PUBLIC  EVNTCNT,MACEND          ; package

;       *** define INTP0 macro service area ***

        DSEG_B  P0,'SADDR'              ; INTP0 macro service channel
        PMCW0                           ; INTP0 macro service control word

;       *** initialize ***

EVNTS   CSEG
EVNTCNT:                                ; -- INTP0 macro service initialize --
        MSMP0=#10100000B                ; TYPE B, memory --> SFR
        CHPP0=#(LOW CHAP0)              ; channel pointer
        SFRP0=#(LOW SIO)                ; SFR pointer(dummy)
        MPP0P=#DATAP0                   ; macro service pointer(dummy)
        MSCP0=A                         ; macro service counter
        SET1    PISM0                   ; set INTP0 macro service mode
        CLR1    PMK0                    ; open mask INTP0
        RET

;       *** reset macro service routine ***

MACEND:
        MPP0P=#DATAP0                   ; macro service pointer
        MSCP0=A                         ; macro service counter
        SET1    PISM0                   ; set INTP0 macro service mode
        RET

;       *** macro service buffer ***

DATAP0  EQU     0                       ; dummy data area top address

        END
```

3.2　パルス周期の測定

タイマ／カウンタを使用して，外部入力パルスの周期の平均を測定するプログラム例を示します。

3.2.1　16ビット・タイマ／カウンタを使用したプログラム例

16ビット・タイマ／カウンタによるパルス周期測定は，次の式で求められます。

{(n回目のINTP3端子入力の有効エッジ検出によりキャプチャ・レジスタ（CR02）に取り込んだTM0値）－（n－1回目のINTP3端子入力の有効エッジ検出によりキャプチャ・レジスタに取り込んだTM0値)}×カウント・クロック（$8/f_{CLK}$）

数回の入力パルス周期の平均を求める際は，マクロ・サービスによって，数回のキャプチャ値をメモリへ転送する方法が有効ですが，CR02は16ビット・レジスタのため，マクロ・サービスは使用できません。そこで，別の方法でマクロ・サービスを応用し，TM0によるパルス周期測定を実現した例を示します。

この応用例をμPD78224シリーズで使用する場合，INTP3端子とINTP4端子を接続し，INTP3割り込みおよびマクロ・サービスをINTP4割り込みおよびマクロ・サービスに変更します。

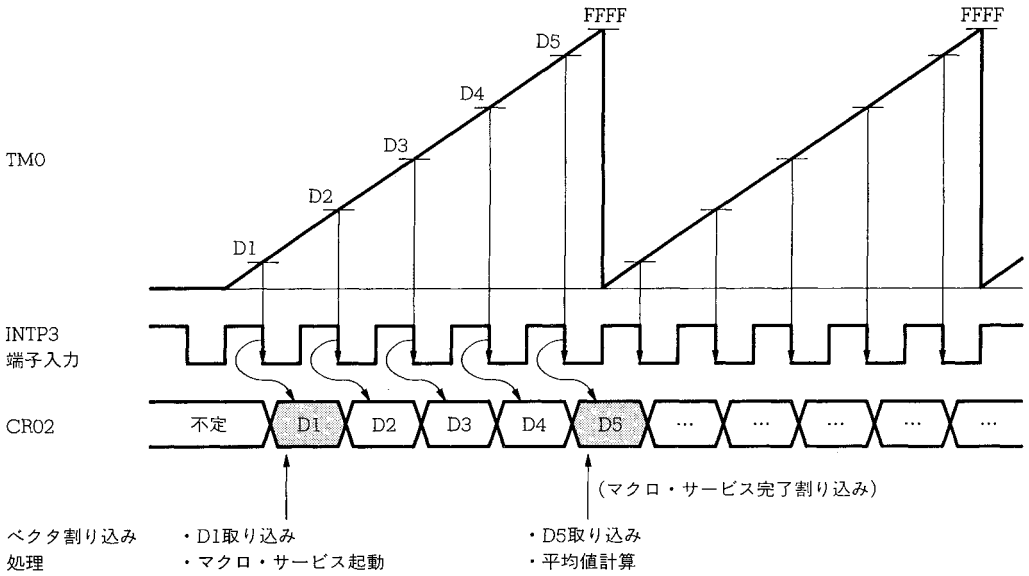
(1) 動作概要

INTP3端子に一定周期で入力されるパルス４発の平均パルス周期を測定します。図３-３に示すように，次の式で５回のキャプチャ値間の平均が求められます。

{(５回目のキャプチャ値)－(１回目のキャプチャ値)}/4

５回のキャプチャ動作をカウントするのに，マクロ・サービスを使用します。平均値の計算は，マクロ・サービス完了割り込みで行います。

図3-3　INTP3端子入力によるキャプチャ動作とベクタ割り込み処理のタイミング



備考　INTP3立ち下がりエッジ有効の場合

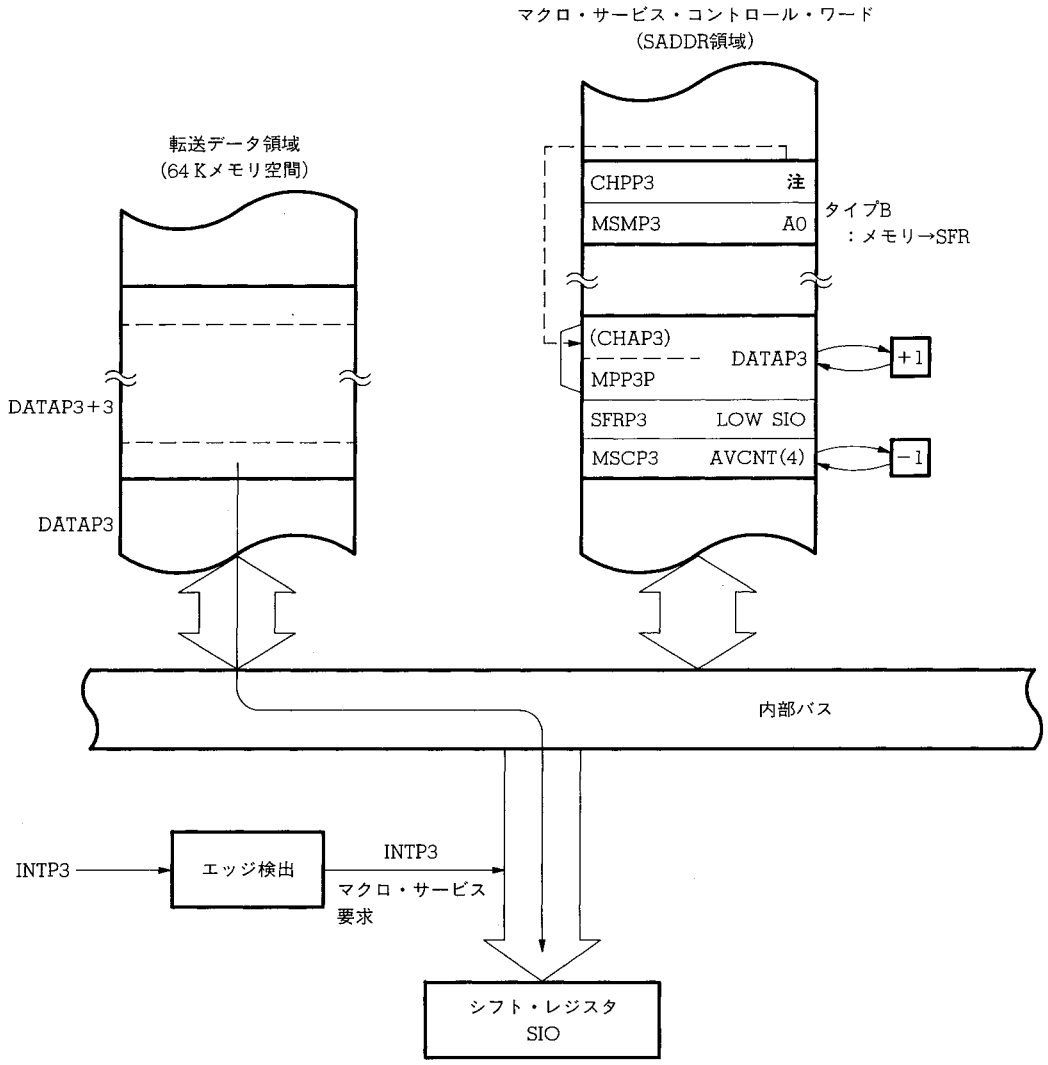
キャプチャ動作は，INTP3の有効エッジ検出ごとにハードウエアで行われますが，実際に必要となるのは，1回目と5回目のキャプチャ値のみです。この応用例では，1回目と5回目のキャプチャ値の取り込みをINTP3ベクタ割り込み処理で行います（図3-3参照）。

214
218A
224
234
244

マクロ・サービスは，タイプB，メモリ→SFR（SIO）を使用しますが，**3.1**と同様に，キャプチャ回数をカウントするだけが目的ですので，転送動作は意味を持ちません。

図3-4にマクロ・サービスのメモリ・マップと動作ブロック図を示します。

図3-4　TMOによるパルス周期測定のマクロ・サービスのメモリ・マップと動作ブロック図



注　CHAP3（＝MPP3P＋1）の下位バイト

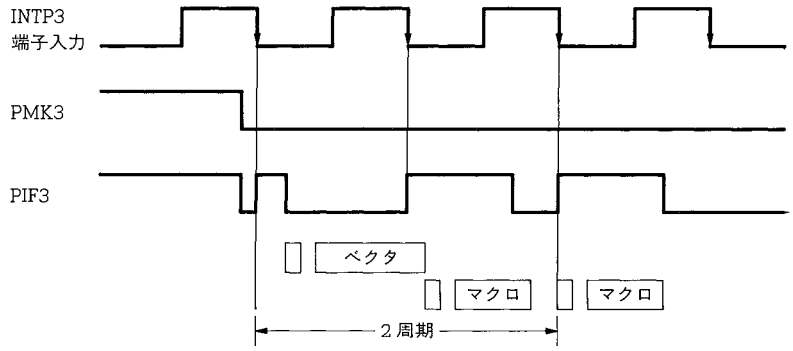
この応用例では，なんらかの要求があった場合にのみ，パルス周期測定処理を行います。したがって，一度処理が終了したら，INTP3割り込み要求をマスク状態にしておきます。

なお，この応用例では，タイマ／カウンタのオーバフローは監視していませんので，測定可能なパルス周期は次の範囲となります。

- ●最小：（ベクタ割り込み受け付け時間＋1回目のベクタ割り込み処理時間＋マクロ・サービス受け付け時間＋1回のマクロ・サービス処理時間）／2[注]
- ●最大：タイマのフル・カウント1周期／4

注　1回目のベクタ割り込み処理と次回のマクロ・サービスのタイミングが，図3-5に示す範囲内となるように，INTP3のパルスを入力してください。

図3-5　INTP3入力と割り込み処理のタイミング例



□：割り込み受け付け時間

(2) プログラム説明

(a) 処理概要

(i) メイン処理［レーベル名称：PULSEO］

INTP3マクロ・サービスのイニシャライズ後，マクロ・サービス判断フラグ（MACF）をクリア（0：初期状態）します。マクロ・サービスのイニシャライズの詳細については，**図3-4のメモリ・マップ，(8) SPDチャート**を参照してください。

(ii) パルス周期測定要求処理［レーベル名称：RQ_PULSE］

パルス周期測定要求以前にセット(1)されたINTP3の割り込み要求フラグをクリア(0)し，パルス周期測定を開始するために，INTP3割り込み要求のマスクを解除します。

(iii) INTP3ベクタ割り込み処理［レーベル名称：INTP3］

この処理に入るには，次の二とおりの場合があります。

- パルス周期測定開始後，1回目のベクタ割り込み
  （マクロ・サービス判断フラグ（MACF）＝0）
- マクロ・サービス完了割り込み
  （マクロ・サービス判断フラグ（MACF）＝1）

それぞれについて説明します。

●マクロ・サービス判断フラグ（MACF）＝0の場合

この処理に入ることで，1回目のキャプチャが行われたことになります。

1回目のキャプチャ値（CR02）をメモリ（FSTCAPT）に格納し，マクロ・サービス判断フラグ（MACF）をセット(1)します。その後，INTP3マクロ・サービス・ポインタとマクロ・サービス・カウンタを設定し，INTP3マクロ・サービスを起動します。ここで起動したマクロ・サービスにより，2回～5回のキャプチャをカウントすることになります。

●マクロ・サービス判断フラグ（MACF）＝1の場合

この処理に入ることで，5回のキャプチャが終了したことになります。

マクロ・サービス判断フラグ（MACF）をクリア（0：初期状態）したあと平均キャプチャ値の計算を行います。5回目にキャプチャされた値（CR02）を読み出し，この値と1回目にキャプチャされた値（FSTCAPT）の差を入力パルス数（平均回数（AVCNT）＝4）で割ることによって，4発のパルスの平均キャプチャ値を求めます。求められたキャプチャ値は，メモリ（AVCAPT）に格納します。

その後，平均キャプチャ値計算終了フラグ（AVENDF）をセット(1)し，INTP3割り込み要求をマスクします。したがって，マクロ・サービスの再設定がされませんので，次にパルス周期測定要求があるまで，キャプチャ回数のカウントは行いません。

(b) 使用するRAM

このプログラム例では，平均キャプチャ値計算用に4バイトと2ビット，マクロ・サービス・チャネルとして4バイトのRAMを使用します。表3-4に使用RAM一覧を示します。

表3－4　TM0によるパルス周期測定の使用RAM一覧

RAM名称	配置領域	内　　容	バイト数	初期値
FSTCAPT	SADDR[注]	1回目のキャプチャ値を格納	2	不定
AVCAPT		平均キャプチャ値を格納	2	
MACF		1回目のキャプチャ終了かマクロ・サービス完了割り込みかの判断	1ビット	0
AVENDF		平均キャプチャ値計算終了の判断	1ビット	0
MPP3P		INTP3マクロ・サービス・ポインタ	2	DATAP3
SFRP3		INTP3マクロ・サービスSFRポインタ	1	LOW SIO
MSCP3		INTP3マクロ・サービス・カウンタ	1	AVCNT

注　SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

(3) 入出力パラメータ

(a) 入力パラメータ

なし

(b) 出力パラメータ

表3－5　TM0によるパルス周期測定の出力パラメータ

パラメータ	固定／可変	内　　容
AVCAPT	可変値	平均キャプチャ値
AVENDF	可変値	平均キャプチャ値計算終了フラグ （本プログラムから復帰時は1）

(4) 使用レジスタ

表3－6　TM0によるパルス周期測定の使用レジスタ

ルーチン	レーベル	レジスタ	バンク
メイン	PULSE0	なし	－
パルス周期測定要求	RQ_PULSE	なし	－
INTP3ベクタ割り込み	INTP3	AX，D	1

(5) 使用スタック

パルス周期測定要求を割り込み要求で行った場合，本プログラム例のサブルーチンおよび割り込み処理で使用するスタック・サイズは，5バイトです（ネスティング：2レベル）。

(6) モード・レジスタ設定例

(a) プログラム使用例でのモード・レジスタ

キャプチャ／コンペア・コントロール・レジスタ0

	7	6	5	4	3	2	1	0	
CRC0	MOD1	MOD0	0	1	CLR01	0	0	0	(リセットで10H)
設定例	0	0	0	1	0	0	0	0	

MOD1	MOD0	CLR01	タイマ出力モード指定		TM0=CR01時のTM0のクリア動作
			TO0	TO1	
0	0	0	トグル出力	トグル出力	禁止
0	0	1	トグル出力	トグル出力	許可
0	1	0	PWM出力	トグル出力	禁止
0	1	1	設定禁止		
1	0	0	PWM出力	PWM出力	禁止
1	0	1	設定禁止		
1	1	0	設定禁止		
1	1	1	PPG出力	トグル出力	許可

タイマ・コントロール・レジスタ0

	7	6	5	4	3	2	1	0	
TMC0	CE3	0	0	0	CE0	OVF0	0	0	（リセットで00H）
設定例	0	0	0	0	1	0	0	0	

16ビット・タイマ／カウンタ

OVF0	TM0のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFFFHから0000Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE0	TM0のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ・ユニット3

CE3	TM3のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

214
218A
224
234
244

外部割り込みモード・レジスタ1

	7	6	5	4	3	2	1	0	
INTM1	0	0	ES51	ES50	ES41	ES40	ES31	ES30	（リセットで00H）
設定例	0	0	0	0	0	0	0	0	

ES31	ES30	INTP3端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES41	ES40	INTP4端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	INTC30を選択
1	1	設定禁止

ES51	ES50	INTP5端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	

(b) プログラム・リストでのモード・レジスタ

マクロ・サービス・モード・レジスタ（INTP3）

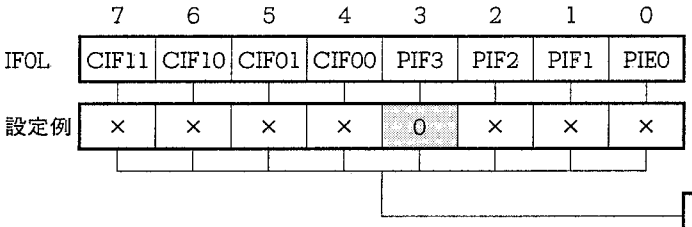
7	6	5	4	3	2	1	0
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例 1	0	1	0	0	0	0	0

CHT0	0	1	0	1			
CHT1	1	0	1				
CHT2	0	1	1				

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC		
						MPTL保持	MPTLインクリメント	
0	0	0	0	メモリからSFRへデータ転送				
0	0	0	1	SFRからメモリへデータ転送				
1	0	0	0			リング制御なし	データ転送のみ	POL
1	0	0	1					POH
1	0	1	0				自動加算つき	POL
1	0	1	1					POH
1	1	0	0			リング制御つき	データ転送のみ	POL
1	1	0	1					POH
1	1	1	0				自動加算つき	POL
1	1	1	1					POH

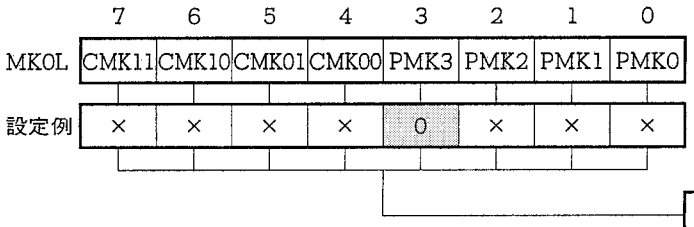
3
214
218A
224
234
244

割り込み要求フラグ・レジスタL



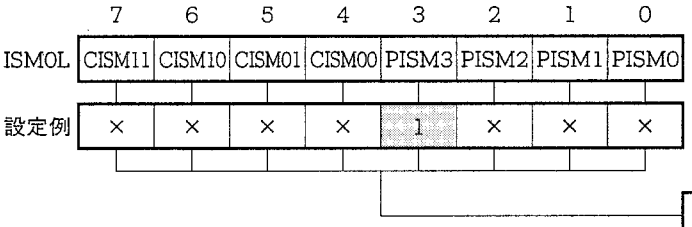
IF	割り込み要求フラグ
0	割り込み要求が発生していない
1	割り込み要求発生

割り込みマスク・レジスタL



MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

割り込みサービス・モード・レジスタL



ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

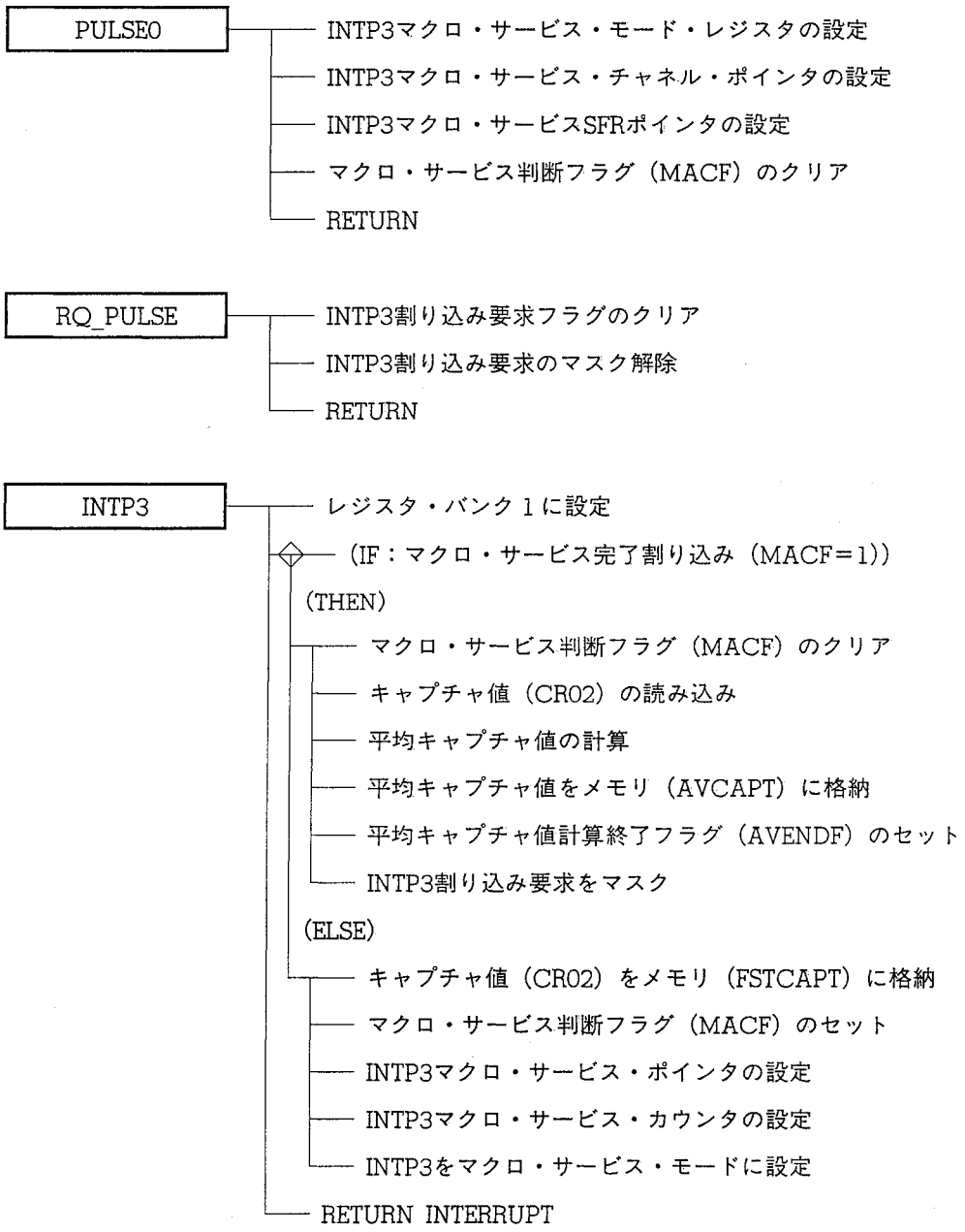
(7) プログラム使用例

割り込み要求によって，パルス周期測定要求を行う例を示します。ただし，割り込みのイニシャライズ（マスク解除など）は含んでいませんので，処理を追加する必要があります。

なお，平均回数を変更する場合は，AVCNTの値を変更してください（(i) **プログラム・リスト**参照）。ただし，AVCNTを大きな値にするほど入力可能なパルスの最大周期が小さくなりますので，注意が必要です。

```
        EXTRN   PULSE0,RQ_PULSE         ; package
        EXTRN   AVCAPT                  ; work area
        EXTBIT  AVENDF                  ; work flag
                :
                :
        CSEG
                :
                :
        MOV     CRC0,#00010000B         ; disable clear TM0 by CR01
        MOV     TMC0,#00001000B         ; count start TM0
        MOV     INTM1,#00000000B        ; set falling edge of INTP5,INTP3
        CLR1    AVENDF                  ; clear average end flag
        CALL    !PULSE0                 ; call PULSE0 package
        EI
PLS0LP:
        BF      AVENDF,$PLS0LP          ; average end ?
                :
                :
        MOVW    AX,AVCAPT               ; read average capture data
                :
                :
;       *** INTP5 interrupt routine ***

INT:
        CALL    !RQ_PULSE               ; mesurement of pulse request rountine
        RETI
                :
                :
```

(8) SPDチャート


PULSE0
INTP3マクロ・サービス・モード・レジスタの設定
INTP3マクロ・サービス・チャネル・ポインタの設定
INTP3マクロ・サービスSFRポインタの設定
マクロ・サービス判断フラグ（MACF）のクリア
RETURN
RQ_PULSE
INTP3割り込み要求フラグのクリア
INTP3割り込み要求のマスク解除
RETURN
INTP3
レジスタ・バンク1に設定
（IF：マクロ・サービス完了割り込み（MACF＝1））
(THEN)
マクロ・サービス判断フラグ（MACF）のクリア
キャプチャ値（CR02）の読み込み
平均キャプチャ値の計算
平均キャプチャ値をメモリ（AVCAPT）に格納
平均キャプチャ値計算終了フラグ（AVENDF）のセット
INTP3割り込み要求をマスク
(ELSE)
キャプチャ値（CR02）をメモリ（FSTCAPT）に格納
マクロ・サービス判断フラグ（MACF）のセット
INTP3マクロ・サービス・ポインタの設定
INTP3マクロ・サービス・カウンタの設定
INTP3をマクロ・サービス・モードに設定
RETURN INTERRUPT

(9) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    PULS0
;*************************************************************
;*      APPLICATION OF MACRO SERVICE                         *
;*      < MEASUREMENT OF PULSE CYCLE >                       *
;*              by 16bit timer/counter                       *
;*************************************************************

        PUBLIC  PULSE0,RQ_PULSE         ; package
        PUBLIC  AVCAPT                  ; work area
        PUBLIC  AVENDF                  ; work flag

AVCNT   EQU     4                       ; average count

;       *** define work area ***

        BSEG
MACF    DBIT                            ; judgment flag of macro service
AVENDF  DBIT                            ; average end flag

WORKS   DSEG    SADDR
FSTCAPT:DS      2                       ; first capture data store memory
AVCAPT: DS      2                       ; average capture data store memory

;       *** define INTP3 macro service area ***

        DSEG_B  P3,'SADDR'              ; INTP3 macro service channel
        PMCW3                           ; INTP3 macro service control word

        PVENT3                          ; INTP3 vector table

;       *** initialize ***

PULS0S  CSEG
PULSE0:                                 ; -- INTP3 macro service initialize --
        MSMP3=#10100000B                ; TYPE B, memory --> SFR
        CHPP3=#(LOW CHAP3)              ; channel pointer
        SFRP3=#(LOW SIO)                ; SFR pointer(dummy)
        CLR1    MACF                    ; clear judgment flag of macro service
        RET
```

```
;       *** measurement of pulse request routine ***

RQ_PULSE:
        CLR1    PIF3                    ; clear INTP3 request flag
        CLR1    PMK3                    ; open INTP3 mask
        RET

;       *** INTP3 interrupt routine ***

INTP3:
        SEL     RB1                     ; set register bank1
        if_bit(MACF)                    ; MACF=1 ?
                                        ; THEN
                CLR1    MACF            ; clear judgment flag of macro service
                AX=CR02                 ; read end capture data
                AX-=FSTCAPT             ; end capture data - first capture data
                D=#AVCNT
                AX/=D                   ; AX/average count
                AVCAPT=AX               ; store average capture data
                SET1    AVENDF          ; set average end flag
                SET1    PMK3            ; mask INTP3
        else
                                        ; ELSE
                FSTCAPT=CR02 (AX)       ; store first capture data(CR02)
                SET1    MACF            ; set judgment flag of macro service
                MPP3P=#DATAP3           ; macro service pointer
                MSCP3=#AVCNT            ; macro service counter
                SET1    PISM3           ; set INTP3 macro service mode
        endif
        RETI

;       *** macro service buffer ***

DATAP3  EQU     0                       ; dummy data area top address

        END
```

3.2.2　8ビット・タイマ／カウンタ1を使用したプログラム例

8ビット・タイマ／カウンタ1によってパルス周期を測定する場合，キャプチャ／コンペア・レジスタ（CR11）をキャプチャ・レジスタとして使用します。この際，INTC11割り込み要求をインターバル・タイマ割り込みとして使用することはできません。

本プログラム例では，CR11レジスタをキャプチャ・レジスタとして使用せずに，パルス周期測定を実現しています。

(1) 動作概要

INTP4端子に一定周期で入力されるパルス9発の平均パルス周期を測定します。

INTP4端子入力の有効エッジ検出で，タイマ・レジスタ（TM1）値をマクロ・サービスによってメモリへ転送します（ここでは，転送されたTM1値をキャプチャ値と呼びます）。

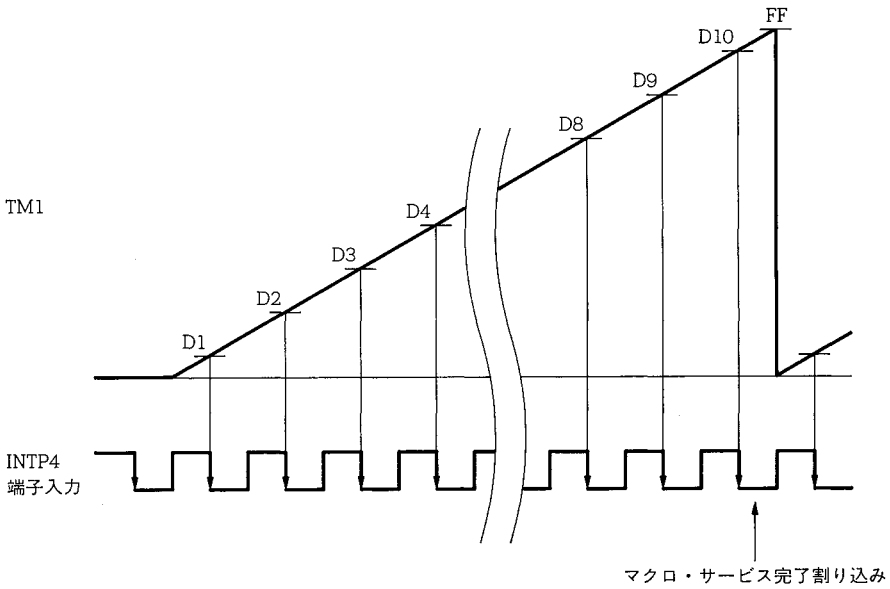
図3-6に示すように，10回のキャプチャ値間の平均は次の式で求められます。

(D10-D1)/9(9＝入力パルス数)

この処理をマクロ・サービス完了割り込み処理で行います。

この応用例では，マクロ・サービス完了割り込み処理でマクロ・サービスを再起動していますので，パルス周期測定処理は連続的に行われます。

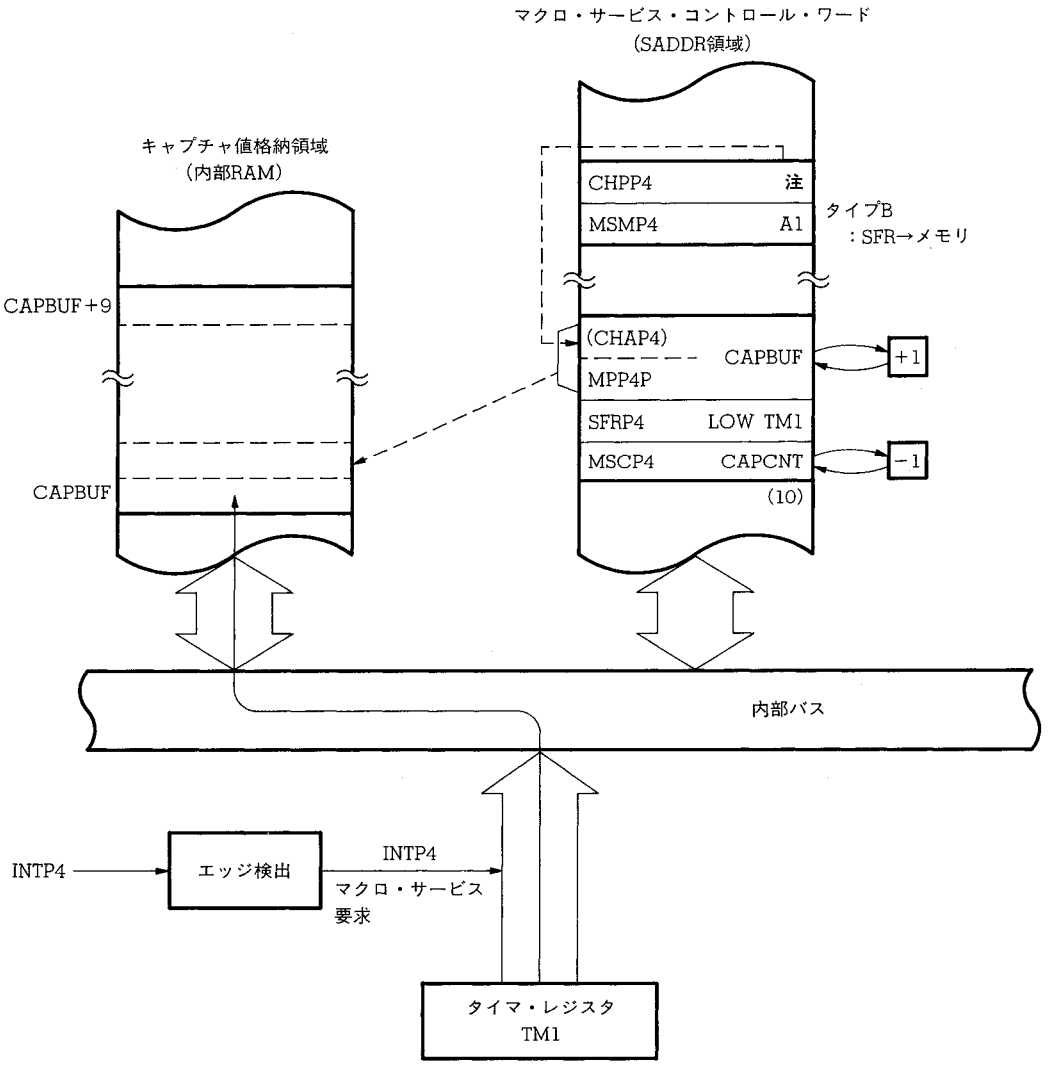
図3-6　INTP4端子入力によるTM1値の転送タイミング



備考　INTP4立ち下がりエッジ有効の場合

マクロ・サービスは，タイプB，SFR（TM1）→メモリを使用します。図3-7にマクロ・サービスのメモリ・マップと動作ブロック図を示します。

図3-7　TM1によるパルス周期測定のマクロ・サービスのメモリ・マップと動作ブロック図



注　CHAP4（＝MPP4P+1）の下位バイト

なお，この応用例では，タイマ／カウンタのオーバフローは監視していませんので，測定可能なパルス周期は次の範囲となります。

- ●最小：（1回のマクロ・サービス処理時間）+（次回のマクロ・サービスが保留される時間[注]）
- ●最大：タイマ／カウンタのフル・カウント周期／9

注　たとえば次のようなプログラムの場合，CLR1命令実行後，次の割り込みの受け付けが可能な状態になっても，＊の命令実行中は割り込みが保留となりますので，その分の時間を加味しなければなりません。

```
MAIN：
                :
                :
            EI
LOOP：
          *BF     FLAG,  $LOOP        ；PLAGは割り込み関連のフラグ
                :
                :
INT：
                :
                :
            CLR1  PIF4                ；割り込み要求フラグのクリア
          *SET1   PISM4               ；マクロ・サービス再起動
          *RETI
```

(2) プログラム説明

(a) 処理概要

(i) メイン処理［レーベル名称：PULSE1］

INTP4マクロ・サービスのイニシャライズを行います。そして，それまで保留されていたINTP4割り込み要求フラグをクリア(0)したあと，INTP4マクロ・サービスを起動します。マクロ・サービスのイニシャライズの詳細については，**図3-7　メモリ・マップ，(8) SPDチャート**を参照してください。

(ii) INTP4マクロ・サービス完了割り込み処理［レーベル名称：INTP4］

平均キャプチャ値の計算処理を行います。10回目のキャプチャ値（CAPBUF+9）と，1回目のキャプチャ値（CAPBUF）の差を，入力パルス数（平均回数（CAPCNT−1）=9）で割ることによって，9発のパルスの平均キャプチャ値を求めます。求められた平均キャプチャ値は，メモリ（AVCAPT）に格納します。

その後，平均キャプチャ値計算終了フラグ（AVENDF）をセット(1)し，INTP4マクロ・サービスの再設定を行います。再設定処理では，平均キャプチャ値計算中にセット(1)されたINTP4割り込み要求フラグをクリア(1)し，マクロ・サービスを再起動します。

3
214
218A
224
234
244

(b) 使用RAM

この応用例では，平均キャプチャ値計算用に，1バイトと1ビット，マクロ・サービス・チャネルとして4バイト，キャプチャ値の転送メモリ（マクロ・サービス・バッファ）として10バイトのRAMを使用します。表3-7に使用RAM一覧を示します。

表3-7　TM1によるパルス周期測定の使用RAM一覧

RAM名称	配置領域	用　　途	バイト数	初期値
AVCAPT	SADDR[注]	平均キャプチャ値を格納	1	不定
AVENDF		平均キャプチャ値計算終了の判断	1ビット	0
MPP4P		INTP4マクロ・サービス・ポインタ	2	CAPBUF
SFRP4		INTP4マクロ・サービスSFRポインタ	1	LOW　TM1
MSCP4		INTP4マクロ・サービス・カウンタ	1	CAPCNT
CAPBUF	RAM[注]	キャプチャ値を転送する（マクロ・サービス・バッファ）	10	不定

注　SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）
　　RAM　：メモリ・バンク0内の任意のRAM領域

(3) 入出力パラメータ

(a) 入力パラメータ

なし

(b) 出力パラメータ

表3-8　TM1によるパルス周期測定の出力パラメータ

パラメータ	固定／可変	内　　容
AVCAPT	可変値	平均キャプチャ値
AVENDF	可変値	平均キャプチャ値計算終了フラグ（本プログラムから復帰時は1）

(4) 使用レジスタ

表3－9　TM1によるパルス周期測定の使用レジスタ

ルーチン	レーベル	レジスタ	バンク
メイン	PULSE1	なし	－
INTP4マクロ・サービス 完了割り込み	INTP4	A, X, D, HL	1

(5) 使用スタック

この応用例のサブルーチンおよび割り込み処理で使用するスタック・サイズは，3バイトです（ネスティング：1レベル）。

(6) モード・レジスタ設定例

(a) プログラム使用例でのモード・レジスタ

キャプチャ／コンペア・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
CRC1	0	0	0	0	CLR11	CM	CLR10	0	(リセットで00H)
設定例	0	0	0	0	0	0	0	0	

CLR10	TM1とCR10レジスタの内容一致でのクリア動作
0	禁止
1	許可

CLR11	CM	CR11レジスタの動作指定	TM1のクリア動作
0	0	コンペア・レジスタ	禁止
1	0		許可(TM1とCR11レジスタの内容一致のとき)
0	1	キャプチャ・レジスタ	禁止
1	1		許可（TM1の内容をCR11レジスタにキャプチャしたとき）

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	0	0	0	0	0	0	0	

8ビット・タイマ／カウンタ1

PRS12	PRS11	PRS10	タイマ／カウンタ1のカウント・クロック周波数の指定
0	0	0	$f_{CLK}/16$[注]
0	0	1	
0	1	0	
0	1	1	$f_{CLK}/32$
1	0	0	$f_{CLK}/64$
1	0	1	$f_{CLK}/128$
1	1	0	$f_{CLK}/256$
1	1	1	$f_{CLK}/512$

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	タイマ／カウンタ3のカウント・クロック周波数の指定
0	0	0	0	$f_{CLK}/16$
0	0	0	1	
0	0	1	0	
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	1	1	1	外部クロック（CI）

注　f_{CLK}：内部システム・クロック周波数（$f_{XX}/2$）

3
214
218A
224
234
244

タイマ・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	0	0	0	0	1	0	0	0	

8ビット・タイマ／カウンタ1

OVF1	TM1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFHから00Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE1	TM1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

CMD2	TM2のカウント動作モード指定
0	通常モード
1	ワンショット・モード

OVF2	TM2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFHから00Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE2	TM2のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

外部割り込みモード・レジスタ 1

	7	6	5	4	3	2	1	0	
INTM1	0	0	ES51	ES50	ES41	ES40	ES31	ES30	（リセットで00H）
設定例	0	0	0	0	0	0	0	0	

ES31	ES30	INTP3端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES41	ES40	INTP4端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	INTC30を選択
1	1	設定禁止

ES51	ES50	INTP5端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	

3
214
218A
224
234
244

(b) プログラム・リストでのモード・レジスタ

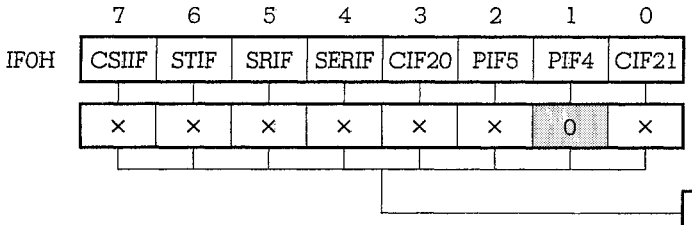
マクロ・サービス・モード・レジスタ（INTP4）

7	6	5	4	3	2	1	0	
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0	（リセットで00H）
設定例 1	0	1	0	0	0	0	1	

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

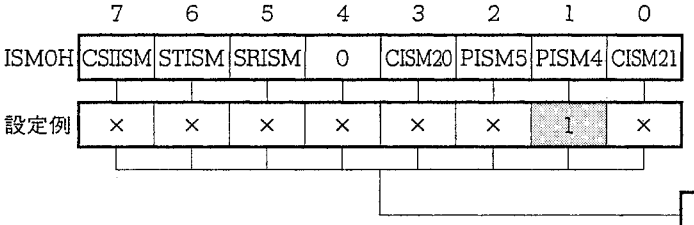
MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし データ転送のみ	POL
1	0	0	1				POH
1	0	1	0			リング制御なし 自動加算つき	POL
1	0	1	1				POH
1	1	0	0			リング制御つき データ転送のみ	POL
1	1	0	1				POH
1	1	1	0			リング制御つき 自動加算つき	POL
1	1	1	1				POH

割り込み要求フラグ・レジスタH



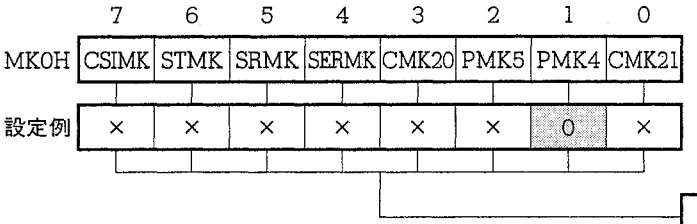
IF	割り込み要求フラグ
0	割り込み要求が発生していない
1	割り込み要求発生

割り込みサービス・モード・レジスタH



ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

割り込みマスク・レジスタH



MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

3
214
218A
224
234
244

(7) プログラム使用例

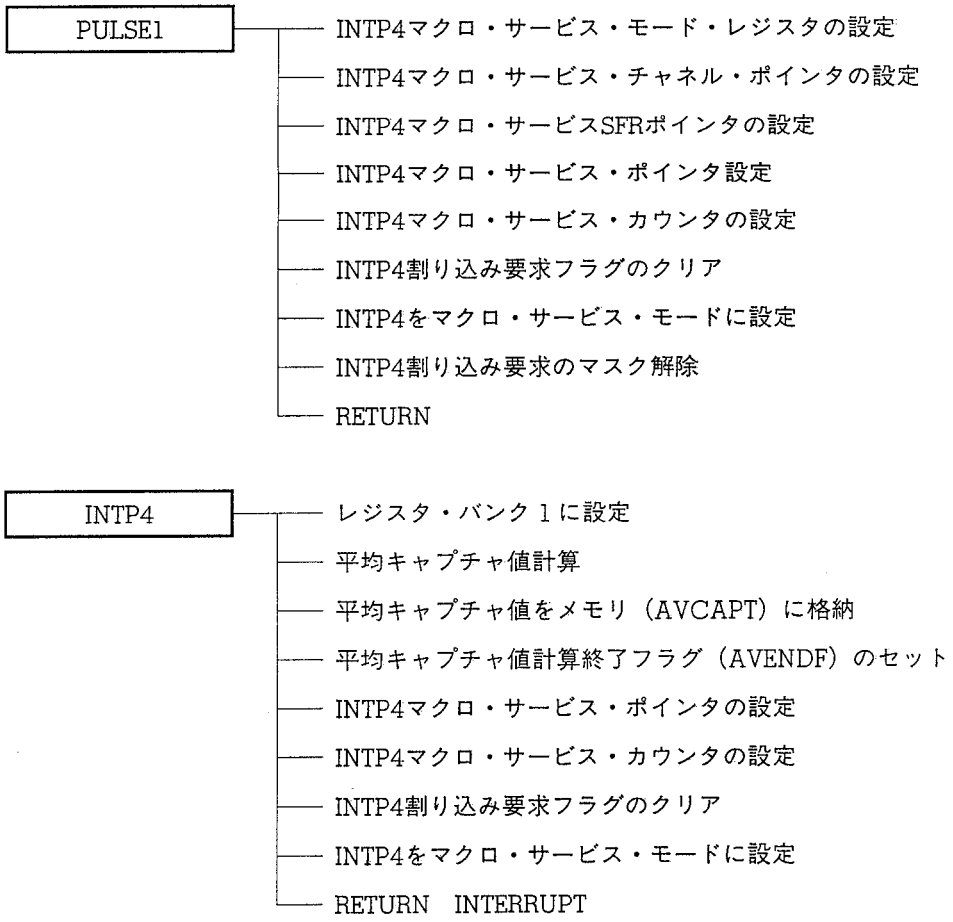
このプログラムの使用例を示します。

TM1のカウント・クロックは，$f_{CLK}/16$を使用します。

なお，キャプチャ回数を変更する場合は，CAPCNTの値を変更してください（**(9) プログラム・リスト**参照）。ただし，CAPCNTを大きな値にするほど入力可能なパルスの最大周期が小さくなりますので，注意が必要です。

```
        EXTRN   PULSE1                  ; package
        EXTRN   AVCAPT                  ; work area
        EXTBIT  AVENDF                  ; work flag
                :
                :
        CSEG
                :
                :
        MOV     CRC1,#00000000B         ; disable clear TM1
        MOV     PRM1,#00000000B         ; TM1 prescaler = fCLK/16
        MOV     TMC1,#00001000B         ; count start TM1
        MOV     INTM1,#00000000B        ; set falling edge of INTP4
        CLR1    AVENDF                  ; clear average end flag
        CALL    !PULSE1                 ; call PULSE1 package
        EI
PULS1LP:
        BF      AVENDF,$PULS1LP         ; average end ?
                :
                :
        MOV     A,AVCAPT                ; read average capture data
                :
                :
```

(8) SPDチャート


PULSE1
INTP4マクロ・サービス・モード・レジスタの設定
INTP4マクロ・サービス・チャネル・ポインタの設定
INTP4マクロ・サービスSFRポインタの設定
INTP4マクロ・サービス・ポインタ設定
INTP4マクロ・サービス・カウンタの設定
INTP4割り込み要求フラグのクリア
INTP4をマクロ・サービス・モードに設定
INTP4割り込み要求のマスク解除
RETURN
INTP4
レジスタ・バンク1に設定
平均キャプチャ値計算
平均キャプチャ値をメモリ（AVCAPT）に格納
平均キャプチャ値計算終了フラグ（AVENDF）のセット
INTP4マクロ・サービス・ポインタの設定
INTP4マクロ・サービス・カウンタの設定
INTP4割り込み要求フラグのクリア
INTP4をマクロ・サービス・モードに設定
RETURN　INTERRUPT

(9) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    PULS1
;*************************************************************
;*      APPLICATION OF MACRO SERVICE                         *
;*      < MEASUREMENT OF PULSE CYCLE >                       *
;*              by 8bit timer/counter 1                      *
;*************************************************************

        PUBLIC  PULSE1                  ; package
        PUBLIC  AVCAPT                  ; work area
        PUBLIC  AVENDF                  ; work flag

CAPCNT  EQU     10                      ; capture count

;       *** define work area ***

        BSEG
AVENDF  DBIT                            ; average end flag

WORKS   DSEG    SADDR
AVCAPT: DS      1                       ; average capture data store memory

MSBUFS  DSEG
CAPBUF: DS      CAPCNT                  ; macro service buffer

;       *** define INTP4 macro service area ***

        DSEG_B  P4,'SADDR'              ; INTP4 macro service channel
        PMCW4                           ; INTP4 macro service control word

        PVENT4                          ; INTP4 vector table

;       *** initialize ***

PULS1S  CSEG
PULSE1:                                 ; -- INTP4 macro service initialize --
        MSMP4=#10100001B                ; TYPE B, SFR --> memory
        CHPP4=#(LOW CHAP4)              ; channel pointer
        SFRP4=#(LOW TM1)                ; SFR pointer
        MPP4P=#CAPBUF                   ; macro service pointer
        MSCP4=#CAPCNT                   ; macro service counter
        CLR1    PIF4                    ; clear INTP4 request flag
        SET1    PISM4                   ; set INTP4 macro service mode
        CLR1    PMK4                    ; open INTP4 mask
        RET
```

```
;       *** end of INTP4 macro service routine ***

INTP4:

;       ---- average rountine ----

        SEL     RB1                     ; set register bank1
        A=!CAPBUF+(CAPCNT-1)            ; read end capture data
        HL=#CAPBUF
        A-=[HL]                         ; end capture data - first capture data
        X=#00H
        A<->X
        D=#CAPCNT-1
        AX/=D                           ; AX/(capture count-1)
        A<->X
        AVCAPT=A                        ; store average capture data
        SET1    AVENDF                  ; set average end flag

;       ---- reset macro service rountine ----

        MPP4P=#CAPBUF                   ; macro service pointer
        MSCP4=#CAPCNT                   ; macro service counter
        CLR1    PIF4                    ; clear INTP4 request flag
        SET1    PISM4                   ; set INTP4 macro service mode
        RETI

        END
```

3.3　アナログ波形出力

ポート（出力モード）にD/Aコンバータを接続します。このポートへ，あらかじめテーブルに用意してあるデータを転送するマクロ・サービスを実行することによって，アナログ波形を出力するプログラム例を示します。

テーブルに用意するデータによって，サイン波，三角波，階段波，のこぎり波など，自由自在に波形を作ることができます。

なお，この応用例では，６ビット分解能の電圧出力型D/Aコンバータを使用します。

この応用例をμPD78224シリーズで使用する場合，８ビット・タイマ／カウンタ３を使用している部分を８ビット・タイマ／カウンタ１を使用するように変更します。

図３－８　アナログ波形出力のハードウエア構成図

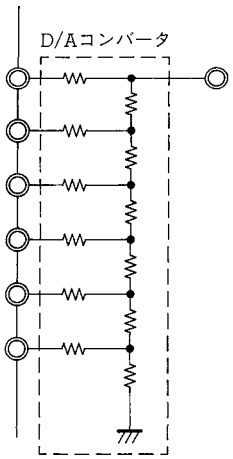
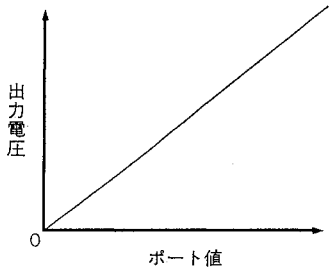


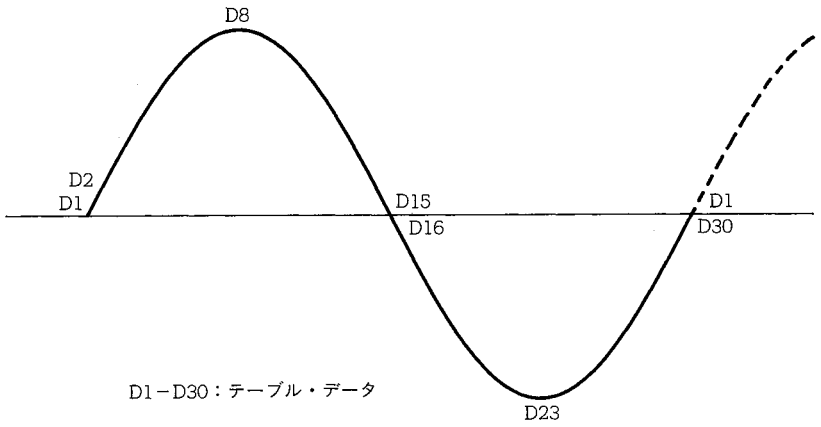
図３－９　ポートへの設定値とD/Aコンバータ出力電圧の関係



(1) 動作概要

テーブルに用意されている30バイトのデータを，マクロ・サービスで順次ポートへ転送し，外付けのD/Aコンバータよりサイン波を出力します。図3-10に出力サイン波とデータの関係を示します。

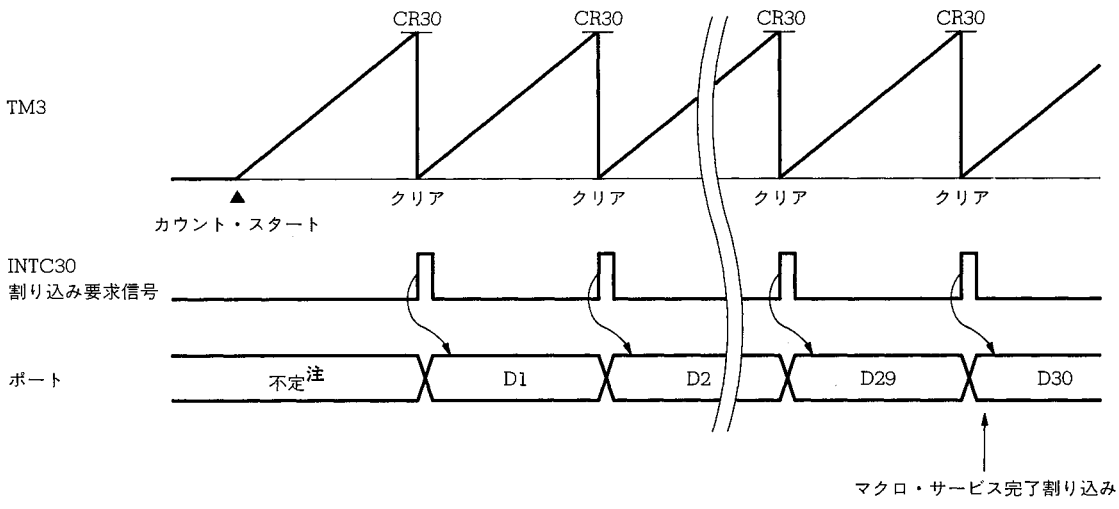
図3-10　出力サイン波とデータの関係



サイン波の周期は，データをポートへ転送するタイミングで決定します。本プログラム例では，8ビット・タイマ／カウンタ3のインターバル・タイマ割り込みINTC30により転送タイミングを作っています。

図3-11にポートへのデータ転送タイミング・チャートを示します。

図3-11　ポートへのデータ転送タイミング・チャート



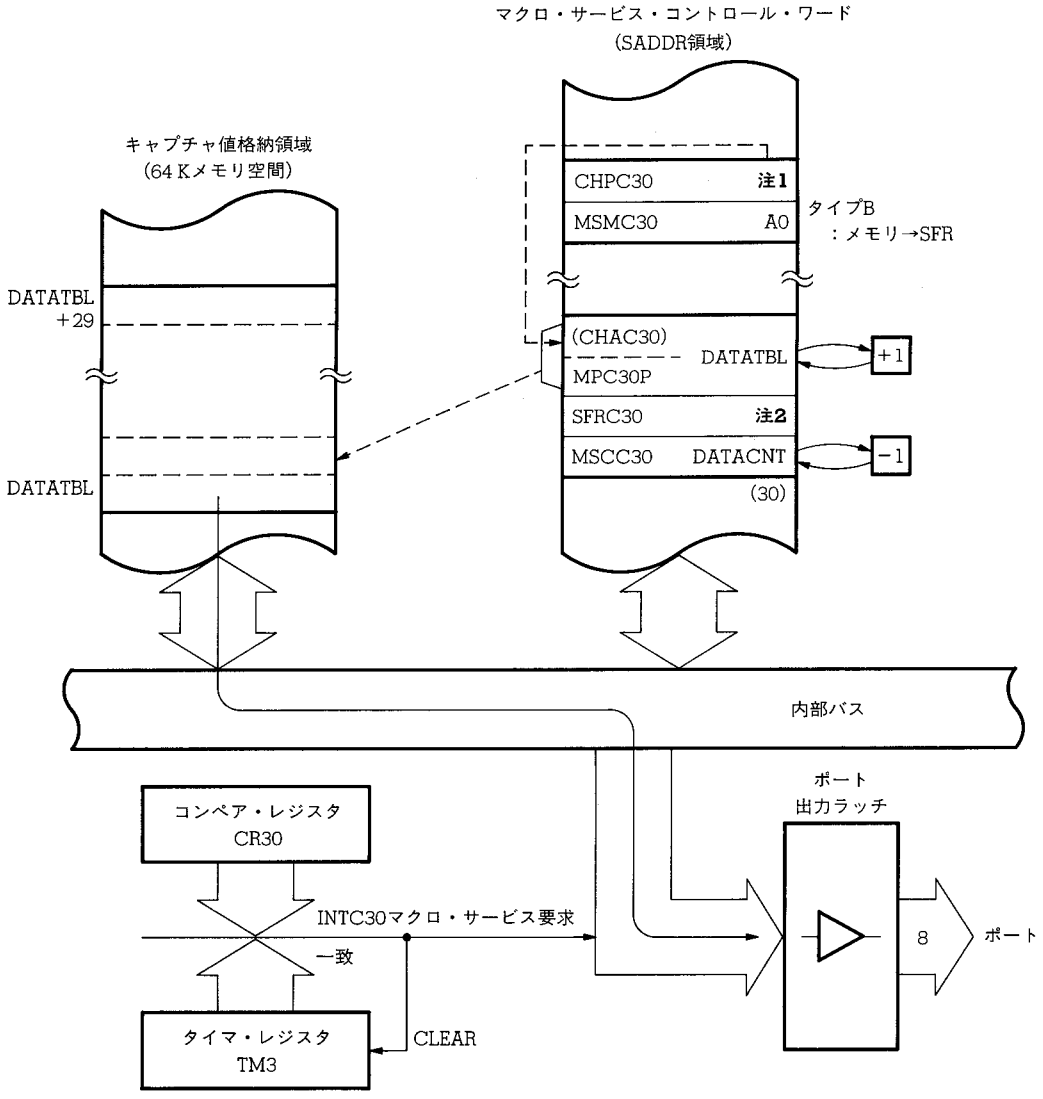
注　初期出力電圧の値を設定

マクロ・サービス完了割り込みでは，INTC30マクロ・サービスを再起動していますので，サイン波が連続的に出力されます。

マクロ・サービスは，タイプB，メモリ→SFR（ポート）を使用します。

図3-12にマクロ・サービスのメモリ・マップと動作ブロック図を示します。

図3-12　アナログ波形出力のマクロ・サービスのメモリ・マップと動作ブロック図



注1. CHAC30（＝MPC30P＋1）の下位バイト

2. OUTPORT（＝データ出力ポートのアドレス）の下位バイト

(2) プログラム説明

(a) モジュール構成

この応用例は，表3－10に示すモジュールで構成されています。

表3－10　アナログ波形出力のモジュール構成一覧

モジュール名	レーベル	処　　理
ANPLS	ANOUT	アナログ波形出力（INTC30マクロ・サービス）処理
SIN_DATA	DATATBL	データ・テーブル定義（処理はしない）

(b) 処理概要

(i) アナログ波形出力処理［モジュール名称：ANPLS］

●メイン処理［レーベル名称：ANOUT］

INTC30マクロ・サービスのイニシャライズを行います。詳細は，**図3－12　メモリ・マップ，(8) SPDチャート**を参照してください。

●INTC30マクロ・サービス完了割り込み処理［レーベル名称：INTC30］

マクロ・サービスの再設定を行います。

(ii) データ・テーブル定義［モジュール名称：SIN_DATA］

サイン波を出力するための30バイトのデータと，データ個数を定義します。

テーブル・データを変更する場合は，このモジュール内のデータを変更してください。

(c) 使用するRAM

この応用例では，INTC30マクロ・サービス・チャネルとして，4バイトのRAMを使用します。表3－11に使用RAM一覧を示します。

表3－11　アナログ波形出力の使用RAM一覧

RAM名称	配置領域	用　　途	バイト数	初期値
MPC30P	SADDR[注]	INTC30マクロ・サービス・ポインタ	2	DATATBL
SFRC30		INTC30マクロ・サービスSFRポインタ	1	LOW OUTPORT
MSCC30		INTC30マクロ・サービス・カウンタ	1	DATACNT

注　SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

(3) 入出力パラメータ

(a) 入力パラメータ

表3－12　アナログ波形出力の入力パラメータ

モジュール	パラメータ	固定／可変	内　　容
ANPLS	DATATBL	固定値	データ・テーブルの先頭アドレス
	DATACNT		テーブル・データ数
	OUTPORT		データ出力ポートのアドレス
SIN_DATA	なし	—	————————

(b) 出力パラメータ

表3－13　アナログ波形出力の出力パラメータ

モジュール	パラメータ	固定／可変	内　　容
ANPLS	なし	—	————————
SIN_DATA	DATATBL	固定値	データ・テーブルの先頭アドレス
	DATACNT		テーブル・データ数

(4) 使用レジスタ

なし

(5) 使用スタック

この応用例の，サブルーチンおよび割り込み処理で使用するスタック・サイズは，3バイトです（ネスティング：1レベル）。

(6) モード・レジスタ設定例

(a) プログラム使用例でのモード・レジスタ

メモリ拡張モード・レジスタ

	7	6	5	4	3	2	1	0	
MM	IFCH	MM6	PW21	PW20	0	MM2	MM1	MM0	(リセットで20H)
	×	×	×	×	0	0	0	1	×：操作しません

MM2	MM1	MM0	モード	P50-P57	P40-P47	P65	P64
0	0	0	シングルチップ・モード	ポート・モード	入力ポート	ポート・モード	
0	0	1			出力ポート		
1	1	1	外部メモリ拡張モード	A8-A15	AD0-AD7	$\overline{WR}$	$\overline{RD}$

PW21	PW20	ウエイト数（指定範囲00000H-0FFFFH）
0	0	0ウエイト
0	1	1ウエイト
1	0	2ウエイト
1	1	$\overline{WAIT}$端子入力のロウ・レベル期間に相当するウエイト数

MM6	1Mバイト拡張モード指定
0	P60-P63は汎用出力ポート
1	P60-P63の出力ラッチ（P60-P63）は外部メモリ拡張モード時の上位アドレス(A16-A19)を記憶し，P60-P63はA16-A19の出力端子として機能する。

IFCH	内部フェッチ・サイクル制御
0	外部ROMフェッチ・サイクルと同様な命令実行サイクルとなる。
1	高速に内部ROMフェッチ動作を行う。 （実行サイクルは外部ROMフェッチのときより速くなる）

プリスケーラ・モード・レジスタ0

	7	6	5	4	3	2	1	0	
PRM0	PRS3	PRS2	PRS1	PRS0	0	0	0	0	（リセットで00H）
設定例	0	0	1	0	0	0	0	0	

8ビット・タイマ／カウンタ3　　f_{CLK}=6 MHz

PRS23	PRS22	PRS21	PRS20	カウント・クロック[Hz]の指定
0	0	0	0	$f_{CLK}/8$
0	0	0	1	
0	0	1	0	$f_{CLK}/16$
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	×	×	×	設定禁止

備考　f_{CLK}：内部システム・クロック

外部割り込みモード・レジスタ 1

	7	6	5	4	3	2	1	0	
INTM1	0	0	ES51	ES50	ES41	ES40	ES31	ES30	（リセットで00H）
設定例	0	0	0	0	1	0	0	0	×：操作しません

ES31	ES30	INTP3端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES41	ES40	INTP4端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	INTC30を選択
1	1	設定禁止

ES51	ES50	INTP5端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	

タイマ・コントロール・レジスタ0

	7	6	5	4	3	2	1	0	
TMC0	CE3	0	0	0	CE0	CVF0	0	0	（リセットで00H）
設定例	1	0	0	0	0	0	0	0	

16ビット・タイマ／カウンタ

OVF0	TM0のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFFFHから0000Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE0	TM0のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ・ユニット3

CE3	TM3のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

(b) プログラム・リストでのモード・レジスタ

マクロ・サービス・モード・レジスタ（INTC30）

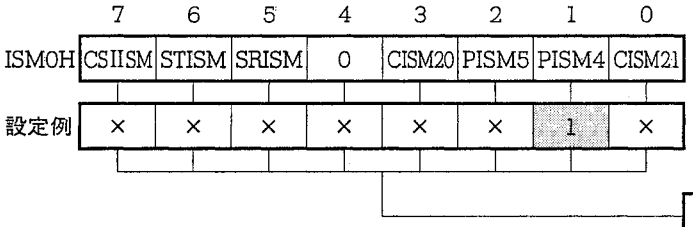
7	6	5	4	3	2	1	0
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例 1	0	1	0	0	0	0	0

CHT0	0	1	0	1			
CHT1	1	0	1				
CHT2	0	1	1				

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC		
						MPTL保持	MPTLインクリメント	
0	0	0	0	メモリからSFRへデータ転送				
0	0	0	1	SFRからメモリへデータ転送				
1	0	0	0			リング制御なし	データ転送のみ	POL
1	0	0	1					POH
1	0	1	0				自動加算つき	POL
1	0	1	1					POH
1	1	0	0			リング制御つき	データ転送のみ	POL
1	1	0	1					POH
1	1	1	0				自動加算つき	POL
1	1	1	1					POH

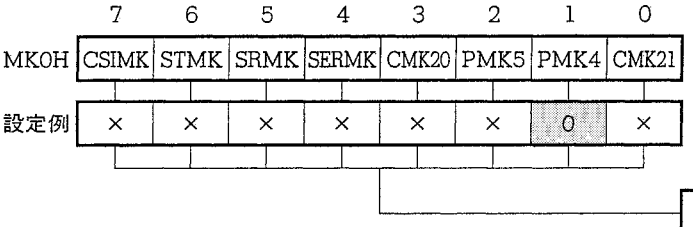
3
214
218A
224
234
244

割り込みサービス・モード・レジスタH



ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

割り込みマスク・レジスタH



MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

(7) プログラム使用例

ポート4へテーブル・データを順次転送し，周期2 ms（周波数500 Hz）のサイン波を出力する例を示します（D/Aコンバータを P40-P45に接続）。

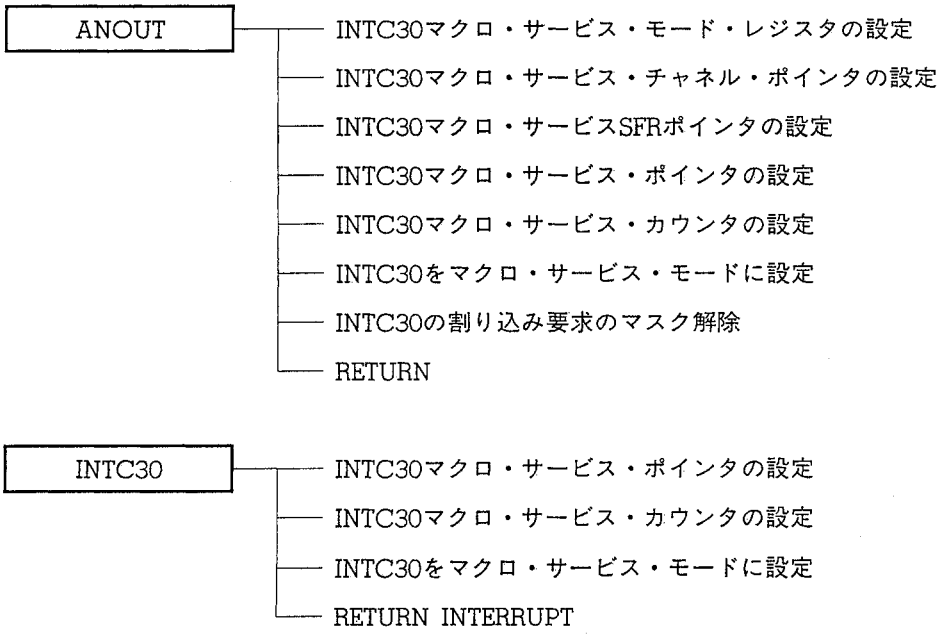
TM3のカウント・クロックは，$f_{CLK}/16$を使用します。この場合，INTC30のインターバル周期は，2 ms/30となり，コンペア・レジスタ（CR30）への設定値は，次のようになります。

$$(2\ \text{ms}/30 \times f_{CLK}/16) - 1 = 24 \quad (f_{CLK} : 6\ \text{MHz})$$

なお，プログラム拡張モードのシステムでこのプログラム例を使用する場合は，データ出力ポート（OUTPORT）として，ポート4，ポート5以外の出力ポートを使用してください。

```
        PUBLIC  OUTPORT                 ; address
        EXTRN   ANOUT                   ; package
        EXTRN   DATATBL                 ; address

OUTPORT EQU     0FF04H                  ; data output port (P4)
INTVL   EQU     24                      ; TM3 interval data (66.6us)
MM0     EQU     MM.0                    ; P4 mode bit
                :
                :
        CSEG
                :
                :
        MOV     A,!DATATBL
        MOV     OUTPORT,A               ; set first output data
        SET1    MM0                     ; P4 output mode
        MOV     PRM0,#00100000B         ; TM3 prescaler = fCLK/16
        MOV     CR30,#INTVL             ; set interval data
        MOV     INTM1,#00001000B        ; select INTC30
        CALL    !ANOUT                  ; call ANOUT package
        MOV     TMC0,#10000000B         ; count start TM3
        EI
                :
                :
```

(8) SPDチャート


ANOUT
INTC30マクロ・サービス・モード・レジスタの設定
INTC30マクロ・サービス・チャネル・ポインタの設定
INTC30マクロ・サービスSFRポインタの設定
INTC30マクロ・サービス・ポインタの設定
INTC30マクロ・サービス・カウンタの設定
INTC30をマクロ・サービス・モードに設定
INTC30の割り込み要求のマスク解除
RETURN
INTC30
INTC30マクロ・サービス・ポインタの設定
INTC30マクロ・サービス・カウンタの設定
INTC30をマクロ・サービス・モードに設定
RETURN INTERRUPT

(9) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードしています。

(a) アナログ波形出力処理

```
$       IC(INTMS.DEF)
        NAME    ANPLS
;*************************************************************
;*      APPLICATION OF MACRO SERVICE                         *
;*      < OUTPUT ANALOG PULSE >                              *
;*************************************************************

        PUBLIC  ANOUT                   ; package
        EXTRN   DATATBL                 ; package
        EXTRN   DATACNT                 ; data
        EXTRN   OUTPORT                 ; address

CMK30   EQU     MK0H.1                  ; INTC30 mask flag
CISM30  EQU     ISM0H.1                 ; INTC30 service mode flag

;       *** define INTC30 macro service area ***

        DSEG_B  C30,'SADDR'             ; INTC30 macro service channel
        CMCW30                          ; INTC30 macro service control word

        CVENT30                         ; INTC30 vector table

;       *** initialize ***

ANPLSS  CSEG
ANOUT:                                  ; -- INTC30 macro service initialize --
        MSMC30=#10100000B               ; TYPE B, memory --> SFR
        CHPC30=#(LOW CHAC30)            ; channel pointer
        SFRC30=#(LOW OUTPORT)           ; SFR pointer
        MPC30P=#DATATBL                 ; macro service pointer
        MSCC30=#DATACNT                 ; macro service counter
        SET1    CISM30                  ; set INTC30 macro service mode
        CLR1    CMK30                   ; open INTC30 mask
        RET

;       *** end of INTC30 macro service routine ***

INTC30:
        MPC30P=#DATATBL                 ; macro service pointer
        MSCC30=#DATACNT                 ; macro service counter
        SET1    CISM30                  ; set INTC30 macro service mode
        RETI

        END
```

(b) データ・テーブル定義

```
        NAME    SIN_DATA
;*************************************************************
;*      APPLICATION OF MACRO SERVICE                         *
;*      < OUTPUT ANALOG PULSE (data table) >                 *
;*************************************************************

        PUBLIC  DATATBL                 ; package
        PUBLIC  DATACNT                 ; data

DATACNT EQU     30                      ; data count

DATAS   CSEG
DATATBL:
        DB      1FH,26H,2CH,32H,36H,3AH,3DH,3EH,3EH,3DH
        DB      3AH,36H,32H,2CH,26H,1FH,18H,12H,0CH,08H
        DB      04H,01H,00H,00H,01H,04H,08H,0CH,12H,18H

        END
```

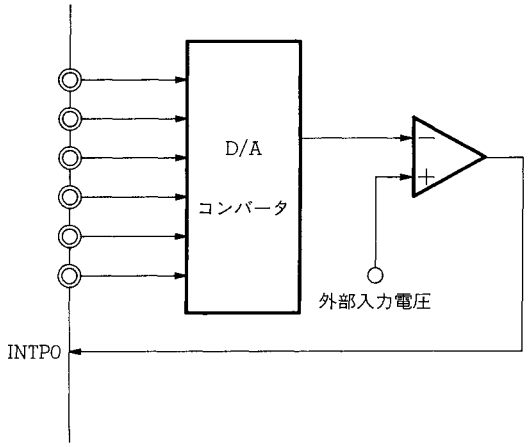
3.4　階段波形出力によるA/D変換

図3-13に示すように，**3.3　アナログ波形出力**で使用したハードウエア構成に，さらにコンパレータを追加し，コンパレータ出力を78K/IIのINTP0エッジ入力端子に接続します。コンパレータ出力は，通常ハイ・レベルで，D/Aコンバータからの入力電圧（－側）の方が，外部入力電圧（＋側）より高くなったときに，ロウ・レベルになるものとします。

一方プログラムでは昇順のデータを，マクロ・サービスでD/Aコンバータへ転送します。これによって，D/Aコンバータから階段波形が出力されます。

以上のハードウエアとマクロ・サービスを使用してA/D変換を実現するプログラム例を示します。

図3-13　階段波形出力によるA/D変換のハードウエア構成図



3
214
218A
224
234
244

3.4.1　ベクタ割り込みを使用したプログラム例

A/D変換値の取り込みを，ベクタ割り込み処理で行う例です。

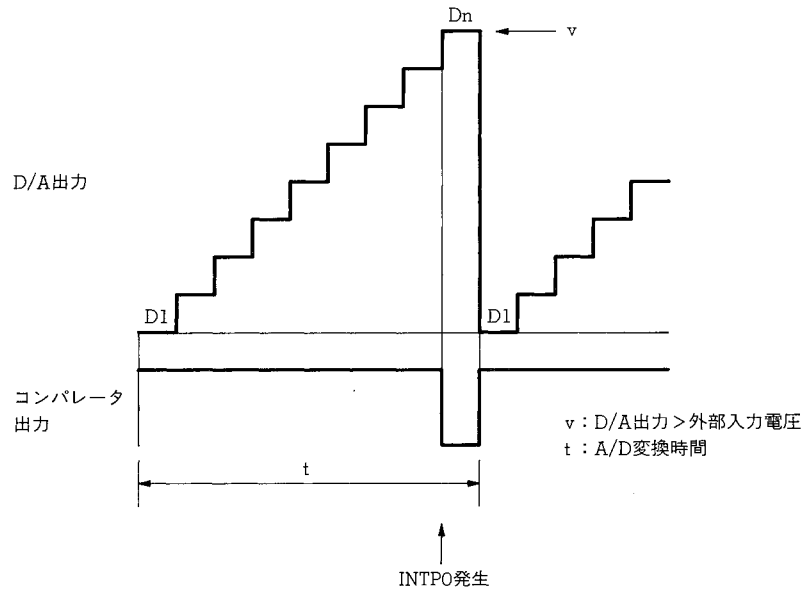
本応用例をμPD78224シリーズで使用する場合，8ビット・タイマ／カウンタ3を使用している部分を8ビット・タイマ／カウンタ1を使用するように変更します。

(1) 動作概要

データ・テーブルに，0～63（6ビット長）のデータを昇順に用意しておき，このデータをマクロ・サービスによって，D/Aコンバータが接続されているポートへ順次転送します。D/Aコンバータからコンパレータへ入力される電圧が外部入力電圧より高くなると，コンパレータ出力が変化し（'L'になり），INTP0エッジ割り込みが発生します（**図3－14**参照）。このINTP0ベクタ割り込み処理で，マクロ・サービスを停止し，データ出力ポートの値を読み出すと，そのときのコンパレータへの外部入力電圧のA/D変換値が得られます。

この際，1回のA/D変換時間（t）は，ポートへのデータ転送タイミングと，コンパレータ出力変化までのマクロ・サービス実行回数（n）で決定します。したがって，外部入力電圧が低いほどマクロ・サービス実行回数は少なく，A/D変換時間も短くなります。

図3－14　ベクタ割り込みによるA/D変換のD/Aコンバータ出力とコンパレータ出力のタイミング



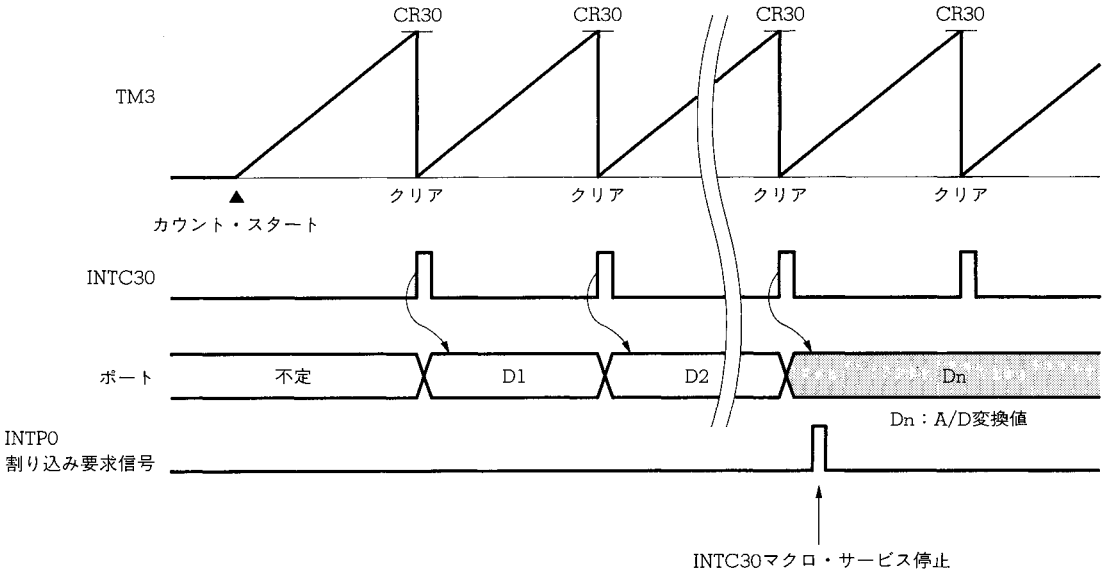
注意　この応用例で使用するハードウエア構成では，サンプル＆ホールド回路を持っていません。したがって，変換終了までにコンパレータ入力電圧が周波数特性やノイズなどにより1LSB以上変化した場合，A/D変換精度が落ちます。

この応用例では，データ転送タイミングを８ビット・タイマ／カウンタ３のインターバル・タイマ割り込みINTC30で作っています。このインターバル時間は，次式で求められる以上の時間が必要です。

（コンパレータの比較処理時間）＋（INTP0割り込み受け付け時間）＋（ポート値の読み出し終了までのINTP0割り込み処理時間）

図３－15にポートへのデータ転送タイミング・チャートを示します。

図３－15　ベクタ割り込みによるA/D変換のポートへのデータ転送タイミング・チャート

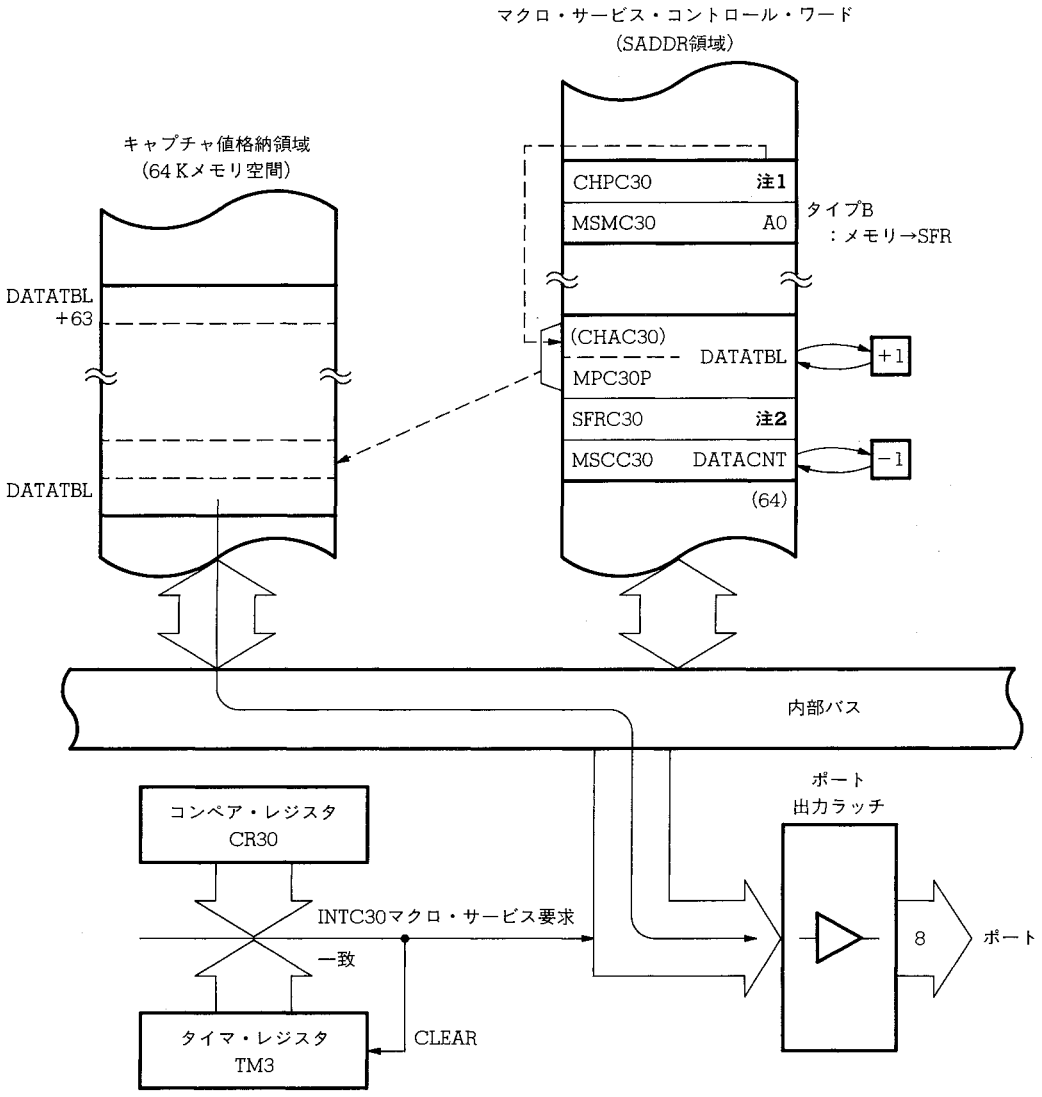


INTP0割り込み処理では，ポートへのデータ転送処理（INTC30マクロ・サービス処理）を停止し，その時点でのポート値（A/D変換値：Dn）を読み出します。

A/D変換値(Dn)を読み出したあと，次に受け付けられたINTC30割り込み処理でINTC30マクロ・サービスを再起動し，A/D変換を連続的に行います。

マクロ・サービスは，タイプB，メモリ→SFR（ポート）を使用します。図３－16にマクロ・サービスのメモリ・マップと動作ブロック図を示します。

図3－16　ベクタ割り込みによるA/D変換のマクロ・サービスのメモリ・マップと動作ブロック図



注1. CHAC30（＝MPC30P＋1）の下位バイト

2. OUTPORT（＝データ出力ポートのアドレス）の下位バイト

(2) プログラム説明

(a) モジュール構成

この応用例は，表3-14に示すモジュールで構成されています。

表3-14　ベクタ割り込みによるA/D変換のモジュール構成一覧

モジュール名	レーベル	処　　理
VADCNV	V_ADCNV	A/D変換（INTP0ベクタ割り込み）処理
ANPLS[注]	ANOUT	アナログ波形出力（INTC30マクロ・サービス）処理
AD_DATA	DATATBL	データ・テーブル定義（処理はしない）

注　3.3　アナログ波形出力のモジュールを再利用しています。

(b) 処理概要

(i) A/D変換処理［モジュール名称：VADCNV］

●メイン処理［レーベル名称：V_ADCNV］

INTP0割り込み要求のマスクを解除し，アナログ波形（階段波形）出力処理ルーチン（ANOUT）を呼び出します。これによって，INTC30マクロ・サービスを起動します。

●INTP0ベクタ割り込み処理［レーベル名称：INTP0］

このルーチンに入ると，A/D変換処理が終了したことになります。

INTC30マクロ・サービス（アナログ波形出力）を停止したあと，データ出力ポート（OUTPORT）の値を読み出し，メモリ（ADRSLT）へ格納します。

その後，A/D変換終了フラグ（ADENDF）をセット(1)します。

(ii) アナログ波形出力処理［モジュール名称：ANPLS］

●メイン処理［レーベル名称：ANOUT］

INTC30マクロ・サービスのイニシャライズを行います。詳細は，**図3-16　メモリ・マップ，(8) SPDチャート**を参照してください。

●INTC30マクロ・サービス完了割り込み処理［レーベル名称：INTC30］

マクロ・サービスの再設定を行います。

214
218A
224
234
244

(iii) データ・テーブル定義［モジュール名称：AD_DATA］

マクロ疑似命令を使用し，定義するデータの個数回だけ定義データをインクリメントしていくようにソース・プログラムを記述しています。

データの個数を変更する(ポート数を増やし，A/D変換の分解能を上げる)場合は，DATACNT値をポート数に応じて変更してください。

(c) 使用するRAM

本プログラム例では，A/D変換用に1バイトと1ビット，INTC30マクロ・サービス・チャネルとして4バイトのRAMを使用します。表3-15に使用RAM一覧を示します。

表3-15　ベクタ割り込みによるA/D変換の使用RAM一覧

RAM名称	配置領域	用　　途	バイト数	初期値
ADRSLT	SADDR[注]	A/D変換値を格納	1	不定
ADENDF		A/D変換終了の判断	1ビット	0
MPC30P		INTC30マクロ・サービス・ポインタ	2	DATATBL
SFRC30		INTC30マクロ・サービスSFRポインタ	1	LOW OUTPORT
MSCC30		INTC30マクロ・サービス・カウンタ	1	DATACNT

注　SADDR：ショート・ダイレクト・アドレシングの適用範囲(0FE20H-0FEFFH)

(3) 入出力パラメータ

(a) 入力パラメータ

表3-16　アナログ波形出力の入力パラメータ

モジュール	パラメータ	固定／可変	内　　容
VADCNV	OUTPORT	固定値	データ出力ポートのアドレス
ANPLS	DATATBL	固定値	データ・テーブルの先頭アドレス
	DATACNT		テーブル・データ数
	OUTPORT		データ出力ポートのアドレス
AD_DATA	なし	—	――――――――

(b) 出力パラメータ

表3-17　ベクタ割り込みによるA/D変換の出力パラメータ

モジュール	パラメータ	固定／可変	内　　容
VADCNV	ADRSLT	可変値	A/D変換値
	ADENDF		A/D変換終了フラグ （本プログラムから復帰時は1）
ANPLS	なし	—	――――――――
AD_DATA	DATATBL	固定値	データ・テーブルの先頭アドレス
	DATACNT		テーブル・データ数

(4) 使用レジスタ

なし

(5) 使用スタック

この応用例の，サブルーチンおよび割り込み処理で使用するスタック・サイズは，4バイトです（ネスティング：2レベル）。

(6) モード・レジスタ設定例

(a) プログラム使用例でのモード・レジスタ

メモリ拡張モード・レジスタ

	7	6	5	4	3	2	1	0	
MM	IFCH	MM6	PW21	PW20	0	MM2	MM1	MM0	(リセットで20H)
	×	×	×	×	0	0	0	1	×：操作しません

MM2	MM1	MM0	モード	P50-P57	P40-P47	P65	P64
0	0	0	シングルチップ・モード	ポート・モード	入力ポート	ポート・モード	
0	0	1			出力ポート		
1	1	1	外部メモリ拡張モード	A8-A15	AD0-AD7	$\overline{WR}$	$\overline{RD}$

PW21	PW20	ウエイト数（指定範囲00000H-0FFFFH）
0	0	0ウエイト
0	1	1ウエイト
1	0	2ウエイト
1	1	$\overline{WAIT}$端子入力のロウ・レベル期間に相当するウエイト数

MM6	1Mバイト拡張モード指定
0	P60-P63は汎用出力ポート
1	P60-P63の出力ラッチ（P60-P63）は外部メモリ拡張モード時の上位アドレス(A16-A19)を記憶し，P60-P63はA16-A19の出力端子として機能する。

IFCH	内部フェッチ・サイクル制御
0	外部ROMフェッチ・サイクルと同様な命令実行サイクルとなる。
1	高速に内部ROMフェッチ動作を行う。 （実行サイクルは外部ROMフェッチのときより速くなる）

プリスケーラ・モード・レジスタ0

	7	6	5	4	3	2	1	0	
PRM0	PRS3	PRS2	PRS1	PRS0	0	0	0	0	（リセットで00H）
設定例	0	0	1	0	0	0	0	0	

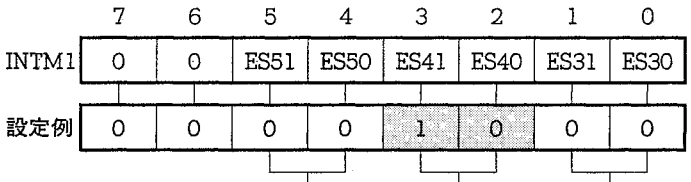
8ビット・タイマ／カウンタ3　　f_{CLK}=6 MHz

PRS23	PRS22	PRS21	PRS20	カウント・クロック[Hz]の指定
0	0	0	0	$f_{CLK}/8$
0	0	0	1	
0	0	1	0	$f_{CLK}/16$
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	×	×	×	設定禁止

備考　f_{CLK}：内部システム・クロック

3
214
218A
224
234
244

外部割り込みモード・レジスタ1

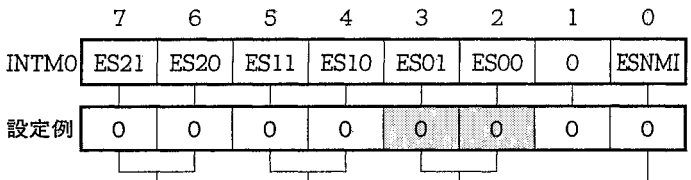


ES31	ES30	INTP3端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES41	ES40	INTP4端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	INTC30を選択
1	1	設定禁止

ES51	ES50	INTP5端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	

外部割り込みモード・レジスタ0



ESNMI	NMI端子入力，検出エッジ指定
0	立ち下がりエッジ
1	立ち上がりエッジ

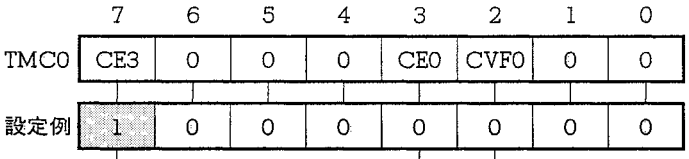
ES01	ES00	INTP0端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES11	ES10	INTP1端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES21	ES20	INTP2端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

3
214
218A
224
234
244

タイマ・コントロール・レジスタ0



	7	6	5	4	3	2	1	0	
TMC0	CE3	0	0	0	CE0	CVF0	0	0	（リセットで00H）
設定例	1	0	0	0	0	0	0	0	

16ビット・タイマ／カウンタ

OVF0	TM0のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFFFHから0000Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE0	TM0のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ・ユニット3

CE3	TM3のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

(b) プログラム・リストでのモード・レジスタ

割り込みマスク・レジスタL

	7	6	5	4	3	2	1	0	
MK0L	CMK11	CMK10	CMK01	CMK00	PMK3	PMK2	PMK1	PMK0	(リセットでFFH)
設定例	×	×	×	×	×	×	×	0	×：操作しません

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

割り込みサービス・モード・レジスタH

	7	6	5	4	3	2	1	0	
ISM0H	CSIISM	STISM	SRISM	0	CISM20	PISM5	PISM4	CISM21	(リセットで00H)
設定例	×	×	×	×	×	×	0	×	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

3
214
218A
224
234
244

マクロ・サービス・モード・レジスタ（INTC30）

	7	6	5	4	3	2	1	0
	CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例	1	0	1	0	0	0	0	0

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし	データ転送のみ POL
1	0	0	1				POH
1	0	1	0				自動加算つき POL
1	0	1	1				POH
1	1	0	0			リング制御つき	データ転送のみ POL
1	1	0	1				POH
1	1	1	0				自動加算つき POL
1	1	1	1				POH

割り込みサービス・モード・レジスタH

	7	6	5	4	3	2	1	0	
ISM0H	CSIISM	STISM	SRISM	0	CISM20	PISM5	PISM4	CISM21	（リセットで00H）
設定例	×	×	×	×	×	×	1	×	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

割り込みマスク・レジスタH

	7	6	5	4	3	2	1	0	
MK0H	CSIMK	STMK	SRMK	SERMK	CMK20	PMK5	PMK4	CMK21	（リセットでFFH）
設定例	×	×	×	×	×	×	0	×	×：操作しません

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

3
214
218A
224
234
244

(7) プログラム使用例

32 μsのタイミングで，ポート4へテーブル・データを順次転送する場合のプログラム例を示します。

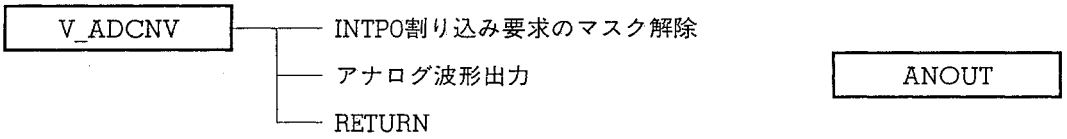
TM3のカウント・クロックは，$f_{CLK}/16$を使用します。この場合，コンペア・レジスタ（CR30）への設定値は次のようになります。

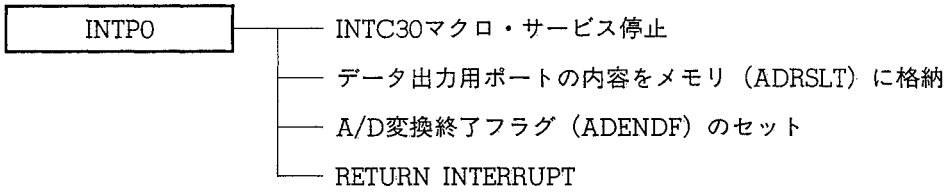
$$(32\,\mu s \times f_{CLK}/16) - 1 = 11 \quad (f_{CLK} : 6\,MHz)$$

```
        PUBLIC  OUTPORT                 ; address
        EXTRN   V_ADCNV                 ; package
        EXTRN   ADRSLT                  ; work area
        EXTRN   DATATBL                 ; address
        EXTBIT  ADENDF                  ; work flag

OUTPORT EQU     0FF04H                  ; data output port (P4)
INTVL   EQU     11                      ; TM3 interval data (32us)
MM0     EQU     MM.0                    ; P4 mode bit
                :
                :
        CSEG
                :
                :
        MOV     A,!DATATBL
        MOV     OUTPORT,A               ; set first output data
        SET1    MM0                     ; P4 output mode
        MOV     PRM0,#00100000B         ; TM3 prescaler = fCLK/16
        MOV     CR30,#INTVL             ; set interval data
        MOV     INTM1,#00001000B        ; select INTC30
        MOV     INTM0,#00000000B        ; set falling edge of INTP0
        CLR1    ADENDF                  ; clear A/D conversion end flag
        CALL    !V_ADCNV                ; call V_ADCNV package
        MOV     TMC0,#10000000B         ; count start TM3
        EI                              ; enable interrupt
VADLP:
        BF      ADENDF,$VADLP           ; A/D conversion end ?
                :
                :
        MOV     A,ADRSLT                ; read A/D conversion result
                :
                :
```

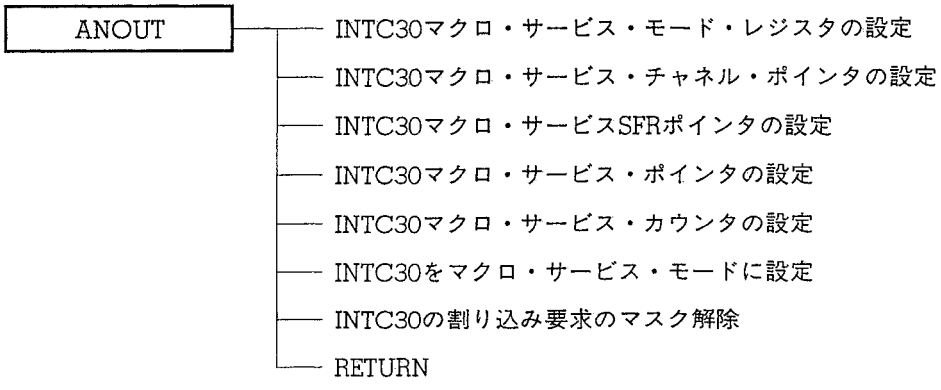
(8) SPDチャート

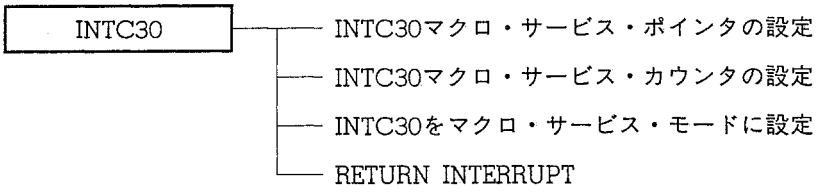
(a) A/D変換処理［モジュール名称：VADCNV］





(b) アナログ波形出力処理［モジュール名称：ANPLS］





(9) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードしています。

●A/D変換処理

```
$       IC(INTMS.DEF)
        NAME    VADCNV
;*************************************************************
;*      APPLICATION OF MACRO SERVICE                         *
;*      < A/D CONVERSION BY OUTPUT RAMP PULSE >              *
;*                      (INTP0 vector)                       *
;*************************************************************

        PUBLIC  V_ADCNV                 ; package
        PUBLIC  ADRSLT                  ; work area
        PUBLIC  ADENDF                  ; work flag
        EXTRN   ANOUT                   ; package
        EXTRN   OUTPORT                 ; address

CISM30  EQU     ISM0H.1                 ; INTC30 service mode flag

;       *** define work area ***

        BSEG
ADENDF  DBIT                            ; A/D conversion end flag

WORKS   DSEG    SADDR
ADRSLT: DS      1                       ; A/D conversion result store memory

        PVENT0                          ; INTP0 vector table

;       *** initialize ***

VADCNVS CSEG
V_ADCNV:
        CLR1    PMK0                    ; open INTP0 mask
        CALL    !ANOUT                  ; call ANOUT package
        RET

;       *** INTP0 interrupt routine ***

INTP0:
        CLR1    CISM30                  ; stop INTC30 macro service
        ADRSLT=OUTPORT                  ; store A/D conversion result
        SET1    ADENDF                  ; set A/D conversion end flag
        RETI
;
        END
```

●アナログ波形出力処理

```
$       IC(INTMS.DEF)
        NAME    ANPLS
;*************************************************************
;*      APPLICATION OF MACRO SERVICE                         *
;*      < OUTPUT ANALOG PULSE >                              *
;*************************************************************

        PUBLIC  ANOUT                   ; package
        EXTRN   DATATBL                 ; package
        EXTRN   DATACNT                 ; data
        EXTRN   OUTPORT                 ; address

CMK30   EQU     MK0H.1                  ; INTC30 mask flag
CISM30  EQU     ISM0H.1                 ; INTC30 service mode flag

;       *** define INTC30 macro service area ***

        DSEG_B  C30,'SADDR'             ; INTC30 macro service channel
        CMCW30                          ; INTC30 macro service control word

        CVENT30                         ; INTC30 vector table

;       *** initialize ***

ANPLSS  CSEG
ANOUT:                                  ; -- INTC30 macro service initialize --
        MSMC30=#10100000B               ; TYPE B, memory --> SFR
        CHPC30=#(LOW CHAC30)            ; channel pointer
        SFRC30=#(LOW OUTPORT)           ; SFR pointer
        MPC30P=#DATATBL                 ; macro service pointer
        MSCC30=#DATACNT                 ; macro service counter
        SET1    CISM30                  ; set INTC30 macro service mode
        CLR1    CMK30                   ; open INTC30 mask
        RET

;       *** end of INTC30 macro service routine ***

INTC30:
        MPC30P=#DATATBL                 ; macro service pointer
        MSCC30=#DATACNT                 ; macro service counter
        SET1    CISM30                  ; set INTC30 macro service mode
        RETI

        END
```

3
214
218A
224
234
244

●データ・テーブル定義

```
        NAME    AD_DATA
;*************************************************************
;*      APPLICATION OF MACRO SERVICE                         *
;*      < A/D CONVERSION BY OUTPUT RAMP PULSE >              *
;*                                      (data table)         *
;*************************************************************

        PUBLIC  DATATBL                 ; package
        PUBLIC  DATACNT                 ; data

DATACNT EQU     64                      ; data count

DATAS   CSEG
DATATBL:
DATA    SET     0
        REPT    DATACNT

        DB      DATA                    ; define data table

DATA    SET     DATA+1
        ENDM

        END
```

3.4.2　マクロ・サービスを使用したプログラム例

A/D変換値の取り込みを，マクロ・サービスで行う例です。

本応用例をμPD78224シリーズで使用する場合，8ビット・タイマ／カウンタ3を使用している部分を8ビット・タイマ／カウンタ1に変更し，INTP0端子をINTP4端子に変更します。

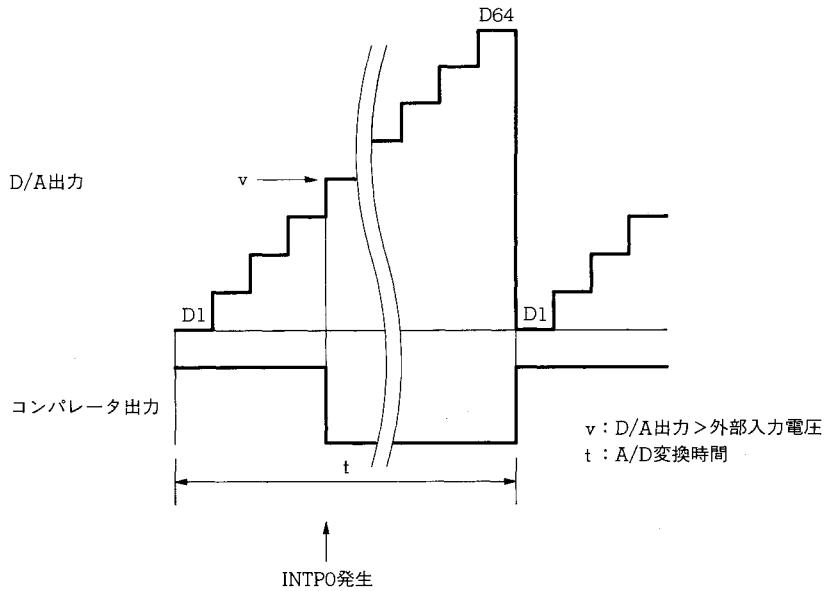
(1) 動作概要

3.4.1　ベクタ割り込みを使用したプログラム例と同様に，テーブルに用意してある0-63のデータをINTC30マクロ・サービスでポートへ転送し，コンパレータへの外部入力電圧より，D/Aコンバータからの入力電圧の方が高くなったときにINTP0が発生します。このときのポートの値を読み出すとA/D変換値が得られるわけですが，ここではさらに，INTP0マクロ・サービスで，ポートの値をメモリへ連続して10バイト転送します。マクロ・サービス完了割り込み処理では，10バイトのA/D変換値の平均を計算します。

3.4.1と異なり，INTP0発生でINTC30マクロ・サービスを停止しませんので，1回のA/D変換で64回のマクロ・サービスが実行されます。したがってA/D変換時間は，外部入力電圧にかかわらず（INTC30インターバル時間×64回）となります。

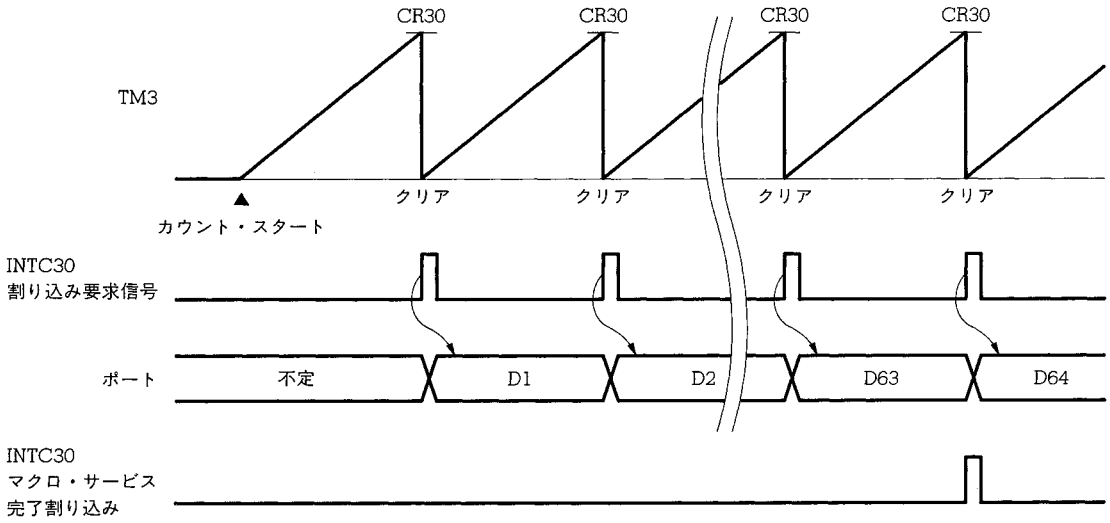
図3-17にD/Aコンバータ出力とコンパレータ出力のタイミング・チャート，図3-18にポートへのデータ転送タイミング・チャートを示します。

図3-17　マクロ・サービスによるA/D変換のD/Aコンバータ出力とコンパレータ出力のタイミング



3
214
218A
224
234
244

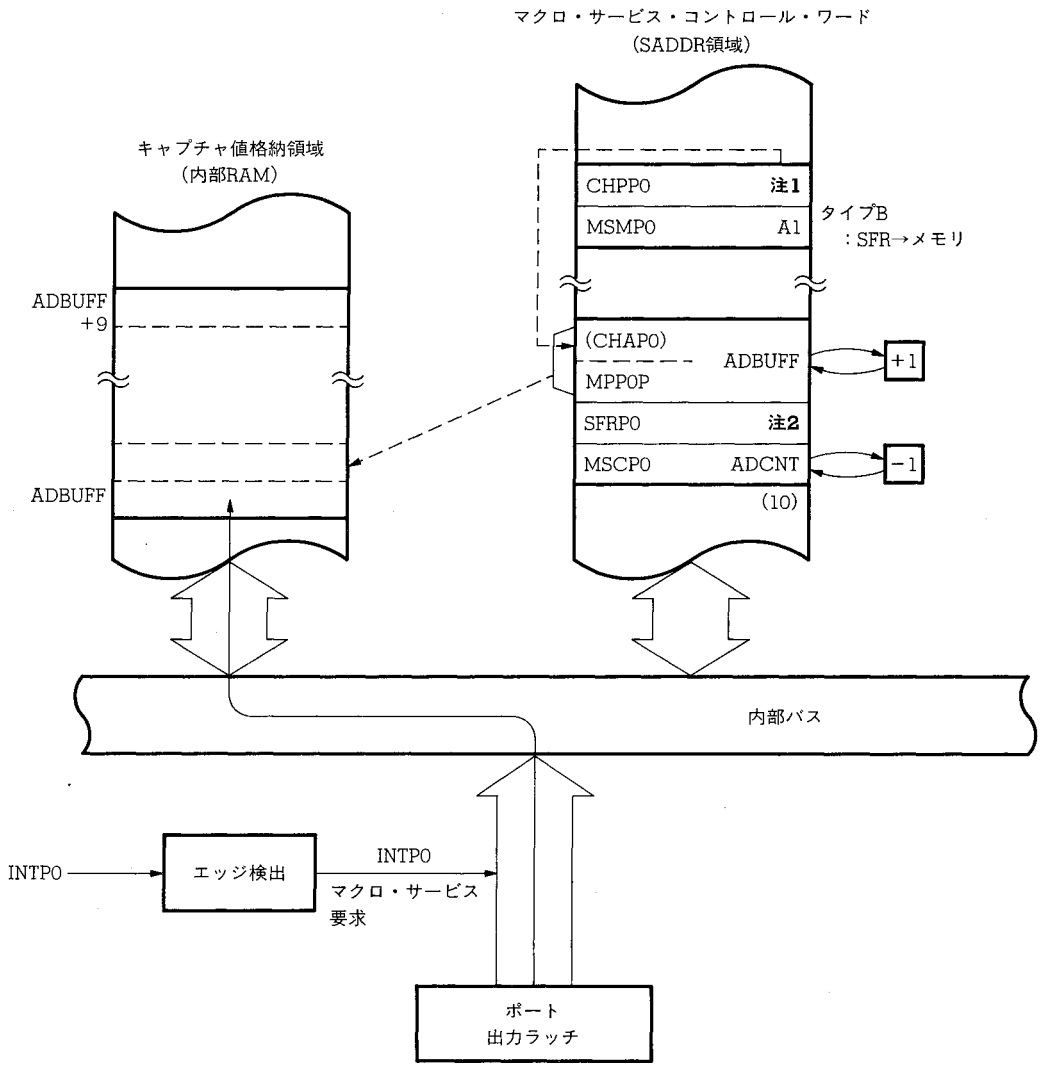
図3－18　マクロ・サービスによるA/D変換のポートへのデータ転送タイミング・チャート



INTP0マクロ・サービスは，タイプB，SFR（ポート）→メモリを使用します。図3-19にINTP0マクロ・サービスのメモリ・マップと動作ブロック図を示します。

なお，INTC30マクロ・サービスのメモリ・マップと動作ブロック図は，**3.4.1**と同様です。

図3-19　マクロ・サービスによるA/D変換のINTP0マクロ・サービスのメモリ・マップと動作ブロック図



注1. CHAP0（＝MPP0P＋1）の下位バイト

2. OUTPORT（＝データ出力ポートのアドレス）の下位バイト

3
214
218A
224
234
244

(2) プログラム説明

(a) モジュール構成

この応用例は，表3－18に示すモジュールで構成されています。

表3－18　マクロ・サービスによるA/D変換のモジュール構成一覧

モジュール名	レーベル	処　　理
MADCNV	M_ADCNV	A/D変換（INTP0マクロ・サービス）処理
ANPLS[注1]	ANOUT	アナログ波形出力（INTC30マクロ・サービス）処理
AD_DATA[注2]	DATATBL	データ・テーブル定義（処理はしない）

注1．**3.3　アナログ波形出力**のモジュールを再利用しています。

2．**3.4.1　ベクタ割り込みを使用したプログラム例**と同じモジュールです。

(b) 処理概要

(i) A/D変換処理［モジュール名称：MADCNV］

●メイン処理［レーベル名称：M_ADCNV］

INTP0マクロ・サービスのイニシャライズ後，アナログ波形（階段波形）出力処理ルーチン（ANOUT）を呼び出します。マクロ・サービスのイニシャライズの詳細については，**図3－19　メモリ・マップ，(8) SPDチャート**を参照してください。

●INTP0マクロ・サービス完了割り込み処理［レーベル名称：INTP0］

このルーチンに入ると，10回のA/D変換処理が終了したことになります。10バイトのA/D変換値格納メモリ(ADBUFF～ADBUFF＋9)の値の総和をデータ数(DATACNT＝10)で割ることによって，10バイトのA/D変換値の平均を求めます。求められた平均A/D変換値をメモリ（ADRSLT）へ格納し，A/D変換終了フラグ（ADENDF）をセット(1)します。その後，INTP0マクロ・サービスを再起動します。

(ii) アナログ波形出力処理［モジュール名称：ANPLS］

●メイン処理［レーベル名称：ANOUT］

INTC30マクロ・サービスのイニシャライズを行います。詳細は，**図3－16　メモリ・マップ，(8) SPDチャート**を参照してください。

●INTC30マクロ・サービス完了割り込み処理［レーベル名称：INTC30］

マクロ・サービスの再設定を行います。

(iii) データ・テーブル定義［モジュール名称：AD_DATA］

マクロ疑似命令を使用し，定義するデータの個数回だけ定義データをインクリメントして行くようにソース・プログラムを記述しています。

データの個数を変更する(ポート数を増やし，A/D変換の分解能を上げる)場合は，DATACNT値を，ポート数に応じて変更してください。

(c) 使用するRAM

この応用例では，A/D変換用に11バイトと1ビット，INTP0マクロ・サービス・チャネルとして4バイト，INTC30マクロ・サービス・チャネルとして，4バイトのRAMを使用します。表3-19に使用RAM一覧を示します。

表3-19　マクロ・サービスによるA/D変換の使用RAM一覧

RAM名称	配置領域	用　途	バイト数	初期値
ADRSLT	SADDR注	平均A/D変換値を格納	1	不定
ADENDF		A/D変換終了の判断	1ビット	0
ADBUFF	RAM注	A/D変換値の転送先 (INTP0マクロ・サービス・バッファ)	10	不定
MPP0P	SADDR	INTP0マクロ・サービス・ポインタ	2	ADBUFF
SFRP0		INTP0マクロ・サービスSFRポインタ	1	LOW OUTPORT
MSCP0		INTP0マクロ・サービス・カウンタ	1	ADCNT
MPC30P		INTC30マクロ・サービス・ポインタ	2	DATATBL
SFRC30		INTC30マクロ・サービスSFRポインタ	1	LOW OUTPORT
MSCC30		INTC30マクロ・サービス・カウンタ	1	DATACNT

注　SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

RAM　：メモリ・バンク0内の任意のRAM領域

(3) 入出力パラメータ

(a) 入力パラメータ

表3-20　マクロ・サービスによるA/D変換の入力パラメータ

モジュール	パラメータ	固定／可変	内　　容
MADCNV	OUTPORT	固定値	データ出力ポートのアドレス
ANPLS	DATATBL	固定値	データ・テーブルの先頭アドレス
	DATACNT		テーブル・データ数
	OUTPORT		データ出力ポートのアドレス
AD_DATA	なし	—	————

(b) 出力パラメータ

表3-21　マクロ・サービスによるA/D変換の出力パラメータ

モジュール	パラメータ	固定／可変	内　　容
MADCNV	ADRSLT	可変値	平均A/D変換値
	ADENDF		A/D変換終了フラグ （このプログラムから復帰時は1）
ANPLS	なし	—	————
AD_DATA	DATATBL	固定値	データ・テーブルの先頭アドレス
	DATACNT		テーブル・データ数

(4) 使用レジスタ

表3-22　マクロ・サービスによるA/D変換の使用レジスター覧

モジュール	ル　ー　チ　ン	レジスタ	バンク
MADCNV	メイン	なし	—
	INTP0マクロ・サービス完了割り込み	AX, B, C, HL	1
ANPLS	アナログ波形出力	なし	—
AD_DATA	データ・テーブル	なし	—

(5) 使用スタック

この応用例の，サブルーチンおよび割り込み処理で使用するスタック・サイズは，4バイトです（ネスティング：2レベル）。

(6) モード・レジスタ設定例

(a) プログラム使用例でのモード・レジスタ

メモリ拡張モード・レジスタ

	7	6	5	4	3	2	1	0	
MM	IFCH	MM6	PW21	PW20	0	MM2	MM1	MM0	(リセットで20H)
	×	×	×	×	0	0	0	1	×：操作しません

MM2	MM1	MM0	モード	P50-P57	P40-P47	P65	P64
0	0	0	シングルチップ・モード	ポート・モード	入力ポート	ポート・モード	
0	0	1			出力ポート		
1	1	1	外部メモリ拡張モード	A8-A15	AD0-AD7	$\overline{WR}$	$\overline{RD}$

PW21	PW20	ウエイト数（指定範囲00000H-0FFFFH）
0	0	0ウエイト
0	1	1ウエイト
1	0	2ウエイト
1	1	$\overline{WAIT}$端子入力のロウ・レベル期間に相当するウエイト数

MM6	1Mバイト拡張モード指定
0	P60-P63は汎用出力ポート
1	P60-P63の出力ラッチ（P60-P63）は外部メモリ拡張モード時の上位アドレス(A16-A19)を記憶し，P60-P63はA16-A19の出力端子として機能する。

IFCH	内部フェッチ・サイクル制御
0	外部ROMフェッチ・サイクルと同様な命令実行サイクルとなる。
1	高速に内部ROMフェッチ動作を行う。 （実行サイクルは外部ROMフェッチのときより速くなる）

プリスケーラ・モード・レジスタ0

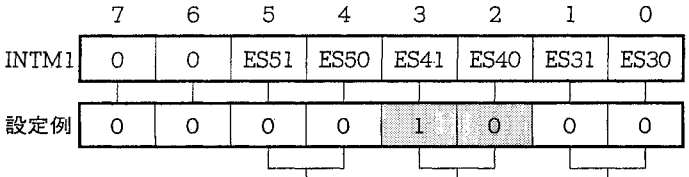
	7	6	5	4	3	2	1	0	
PRM0	PRS3	PRS2	PRS1	PRS0	0	0	0	0	（リセットで00H）
設定例	0	0	1	0	0	0	0	0	

8ビット・タイマ／カウンタ3　　f_{CLK}=6 MHz

PRS23	PRS22	PRS21	PRS20	カウント・クロック[Hz]の指定
0	0	0	0	$f_{CLK}/8$
0	0	0	1	
0	0	1	0	$f_{CLK}/16$
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	×	×	×	設定禁止

備考　f_{CLK}：内部システム・クロック

外部割り込みモード・レジスタ1



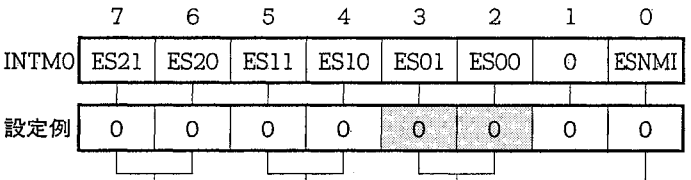
ES31	ES30	INTP3端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES41	ES40	INTP4端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	INTC30を選択
1	1	設定禁止

ES51	ES50	INTP5端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	

3
214
218A
224
234
244

外部割り込みモード・レジスタ0



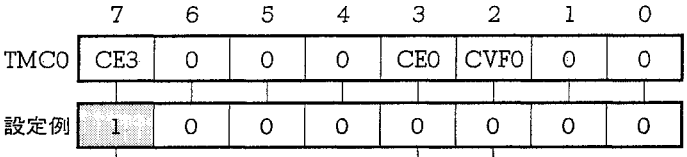
ESNMI	NMI端子入力，検出エッジ指定
0	立ち下がりエッジ
1	立ち上がりエッジ

ES01	ES00	INTP0端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES11	ES10	INTP1端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES21	ES20	INTP2端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

タイマ・コントロール・レジスタ0



16ビット・タイマ／カウンタ

OVF0	TM0のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFFFHから0000Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE0	TM0のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ・ユニット3

CE3	TM3のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

3
214
218A
224
234
244

(b) プログラム・リストでのモード・レジスタ

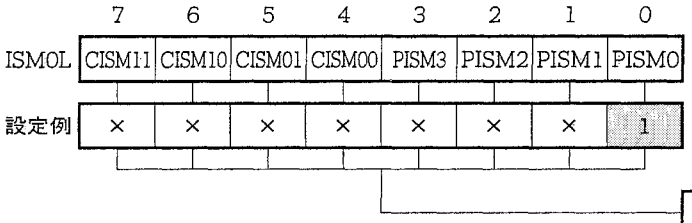
マクロ・サービス・モード・レジスタ（INTPO）

7	6	5	4	3	2	1	0	
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0	（リセットで00H）
設定例 1	0	1	0	0	0	0	1	

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

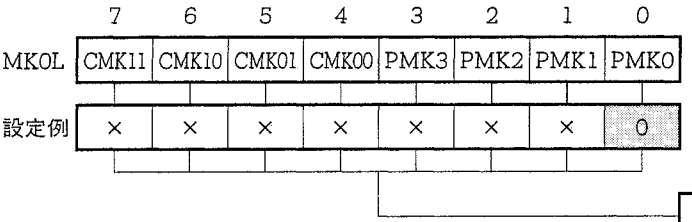
MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし	データ転送のみ POL
1	0	0	1				データ転送のみ POH
1	0	1	0				自動加算つき POL
1	0	1	1				自動加算つき POH
1	1	0	0			リング制御つき	データ転送のみ POL
1	1	0	1				データ転送のみ POH
1	1	1	0				自動加算つき POL
1	1	1	1				自動加算つき POH

割り込みサービス・モード・レジスタL



ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

割り込みマスク・レジスタL



MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

3
214
218A
224
234
244

マクロ・サービス・モード・レジスタ（INTC30）

7	6	5	4	3	2	1	0
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例 1	0	1	0	0	0	0	0

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし データ転送のみ	POL
1	0	0	1				POH
1	0	1	0			自動加算つき	POL
1	0	1	1				POH
1	1	0	0			リング制御つき データ転送のみ	POL
1	1	0	1				POH
1	1	1	0			自動加算つき	POL
1	1	1	1				POH

割り込みサービス・モード・レジスタH

	7	6	5	4	3	2	1	0	
ISM0H	CSIISM	STISM	SRISM	0	CISM20	PISM5	PISM4	CISM21	（リセットで00H）
設定例	×	×	×	×	×	×	1	×	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

割り込みマスク・レジスタH

	7	6	5	4	3	2	1	0	
MK0H	CSIMK	STMK	SRMK	SERMK	CMK20	PMK5	PMK4	CMK21	（リセットでFFH）
設定例	×	×	×	×	×	×	0	×	×：操作しません

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

3
214
218A
224
234
244

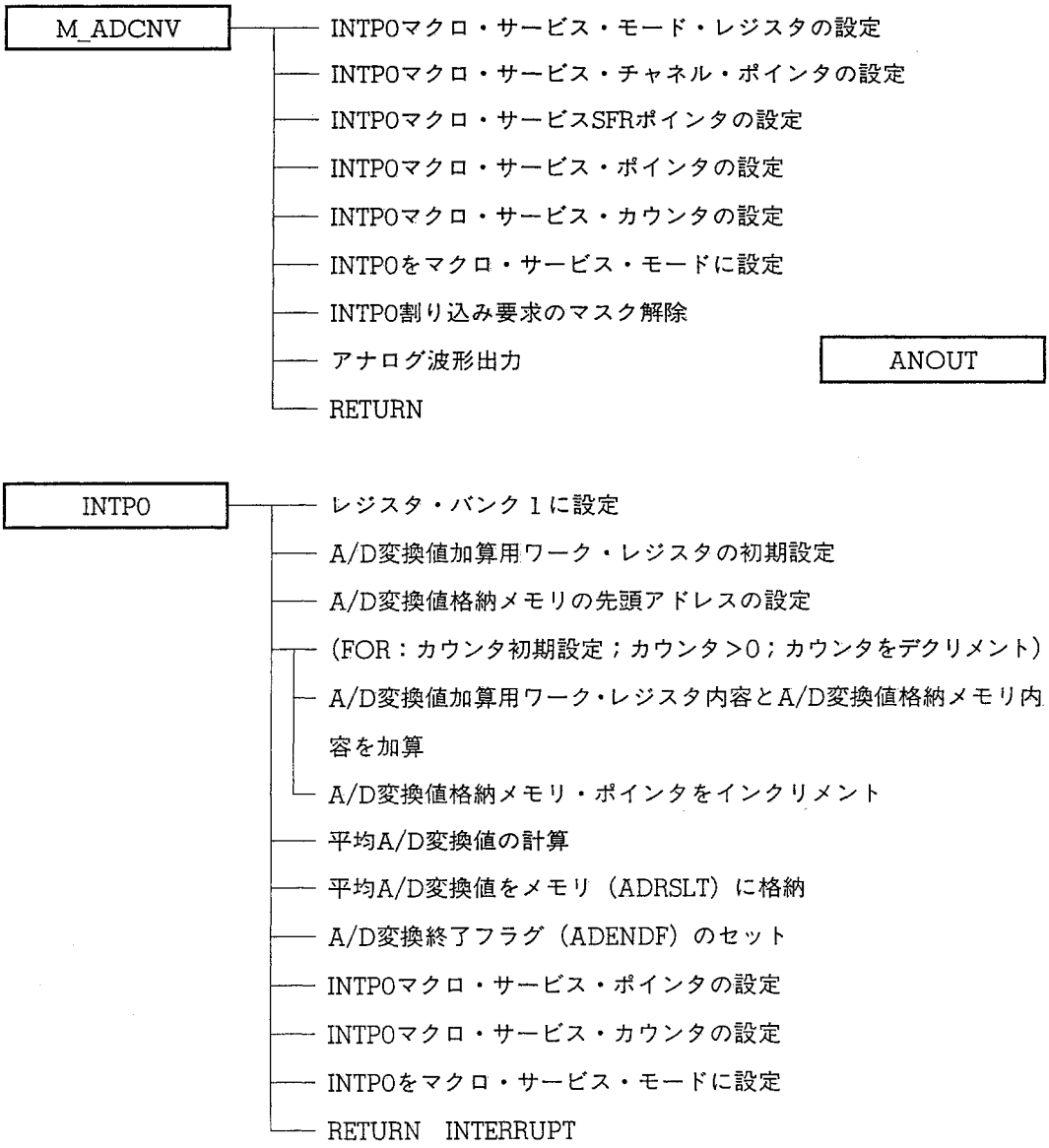
(7) プログラム使用例

3.4.1と同様に，32 μsのタイミングで，ポート4へテーブル・データを順次転送する場合のプログラム例を示します。

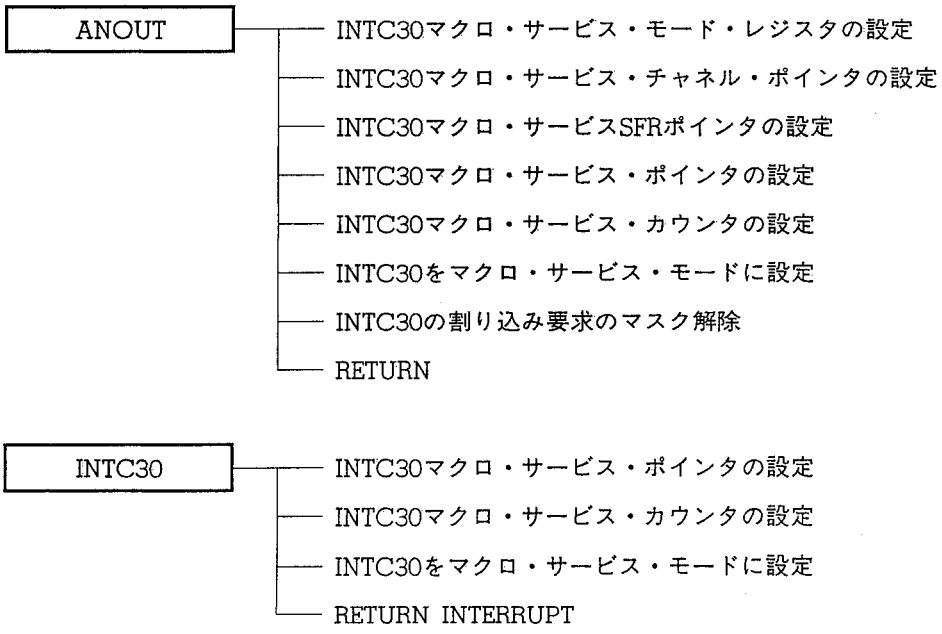
```
        PUBLIC  OUTPORT                 ; address
        EXTRN   V_ADCNV                 ; package
        EXTRN   ADRSLT                  ; work area
        EXTRN   DATATBL                 ; address
        EXTBIT  ADENDF                  ; work flag

OUTPORT EQU     0FF04H                  ; data output port (P4)
INTVL   EQU     11                      ; TM3 interval data (32us)
MM0     EQU     MM.0                    ; P4 mode bit
                :
                :
        CSEG
                :
                :
        MOV     A,!DATATBL
        MOV     OUTPORT,A               ; set first output data
        SET1    MM0                     ; P4 output mode
        MOV     PRM0,#00100000B         ; TM3 prescaler = fCLK/16
        MOV     CR30,#INTVL             ; set interval data
        MOV     INTM1,#00001000B        ; select INTC30
        MOV     INTM0,#00000000B        ; set falling edge of INTP0
        CLR1    ADENDF                  ; clear A/D conversion end flag
        CALL    !V_ADCNV                ; call V_ADCNV package
        MOV     TMC0,#10000000B         ; count start TM3
        EI                              ; enable interrupt
VADLP:
        BF      ADENDF,$VADLP           ; A/D conversion end ?
                :
                :
        MOV     A,ADRSLT                ; read A/D conversion result
                :
                :
```

(8) SPDチャート

(a) A/D変換処理［モジュール名称：MADCNV］



(b) アナログ波形出力処理［モジュール名称：ANPLS］



(9) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS. DEF）をインクルードしています。

●A/D変換処理

```
$       IC(INTMS.DEF)
        NAME    MADCNV
;*************************************************************
;*      APPLICATION OF MACRO SERVICE                         *
;*      < A/D CONVERSION BY OUTPUT RAMP PULSE >              *
;*                      (INTP0 macro service)                *
;*************************************************************

        PUBLIC  M_ADCNV                 ; package
        PUBLIC  ADRSLT                  ; work area
        PUBLIC  ADENDF                  ; work flag
        EXTRN   ANOUT                   ; package
        EXTRN   OUTPORT                 ; address

ADCNT   EQU     10                      ; data count

;       *** define work area ***

        BSEG
ADENDF  DBIT                            ; A/D conversion end flag

WORKS   DSEG    SADDR
ADRSLT: DS      1                       ; average A/D conversion result
                                        ; store area
MSBUFS  DSEG
ADBUFF: DS      ADCNT                   ; INTP0 macro service buffer

;       *** define INTP0 macro service area ***

        DSEG_B  P0,'SADDR'              ; INTP0 macro service channel
        PMCW0                           ; INTP0 macro service control word

        PVENT0                          ; INTP0 vector table
```

```
;       *** initialize ***

MADCNVS CSEG
M_ADCNV:                                ; -- INTP0 macro service initialize --
        MSMP0=#10100001B                ; TYPE B, SFR --> memory
        CHPP0=#(LOW CHAP0)              ; channel pointer
        SFRP0=#(LOW OUTPORT)            ; SFR pointer
        MPP0P=#ADBUFF                   ; macro service pointer
        MSCP0=#ADCNT                    ; macro service counter
        SET1    PISM0                   ; set INTP0 macro service mode
        CLR1    PMK0                    ; open INTP0 mask
        CALL    !ANOUT                  ; call ANOUT package
        RET

;       *** end of INTP0 macro service routine ***

INTP0:
;       ---- average routine ----

        SEL     RB1                     ; set register bank 1
        AX=#0
        HL=#ADBUFF                      ; set macro service buffer top address
        C=#0
        for(B=#ADCNT;B>C;B--)           ; B is average count
                A+=[HL+]                ; add lower 8bit
                ADDC    X,C             ; add higher 8bit
        next
        A<->X
        B=#ADCNT                        ; set average count
        AX/=B                           ; AX/average count
        A<->X
        ADRSLT=A                        ; store A/D conversion result

;       ---- reset macro service routine ----

        SET1    ADENDF                  ; set A/D conversion end flag
        MPP0P=#ADBUFF                   ; macro service pointer
        MSCP0=#ADCNT                    ; macro service counter
        SET1    PISM0                   ; set INTP0 macro service mode
        RETI

        END
```

●アナログ波形出力処理

```
$       IC(INTMS.DEF)
        NAME    ANPLS
;*************************************************************
;*      APPLICATION OF MACRO SERVICE                         *
;*      < OUTPUT ANALOG PULSE >                              *
;*************************************************************

        PUBLIC  ANOUT                   ; package
        EXTRN   DATATBL                 ; package
        EXTRN   DATACNT                 ; data
        EXTRN   OUTPORT                 ; address

CMK30   EQU     MK0H.1                  ; INTC30 mask flag
CISM30  EQU     ISM0H.1                 ; INTC30 service mode flag

;       *** define INTC30 macro service area ***

        DSEG_B  C30,'SADDR'             ; INTC30 macro service channel
        CMCW30                          ; INTC30 macro service control word

        CVENT30                         ; INTC30 vector table

;       *** initialize ***

ANPLSS  CSEG
ANOUT:                                  ; -- INTC30 macro service initialize --
        MSMC30=#10100000B               ; TYPE B, memory --> SFR
        CHPC30=#(LOW CHAC30)            ; channel pointer
        SFRC30=#(LOW OUTPORT)           ; SFR pointer
        MPC30P=#DATATBL                 ; macro service pointer
        MSCC30=#DATACNT                 ; macro service counter
        SET1    CISM30                  ; set INTC30 macro service mode
        CLR1    CMK30                   ; open INTC30 mask
        RET

;       *** end of INTC30 macro service routine ***

INTC30:
        MPC30P=#DATATBL                 ; macro service pointer
        MSCC30=#DATACNT                 ; macro service counter
        SET1    CISM30                  ; set INTC30 macro service mode
        RETI

        END
```

●データ・テーブル定義

```
        NAME    AD_DATA
;*************************************************************
;*      APPLICATION OF MACRO SERVICE                         *
;*      < A/D CONVERSION BY OUTPUT RAMP PULSE >              *
;*                                      (data table)         *
;*************************************************************

        PUBLIC  DATATBL                 ; package
        PUBLIC  DATACNT                 ; data

DATACNT EQU     64                      ; data count

DATAS   CSEG
DATATBL:
DATA    SET     0
        REPT    DATACNT

        DB      DATA                    ; define data table

DATA    SET     DATA+1
        ENDM

        END
```

3.5　リアルタイム出力ポートによるPWM出力

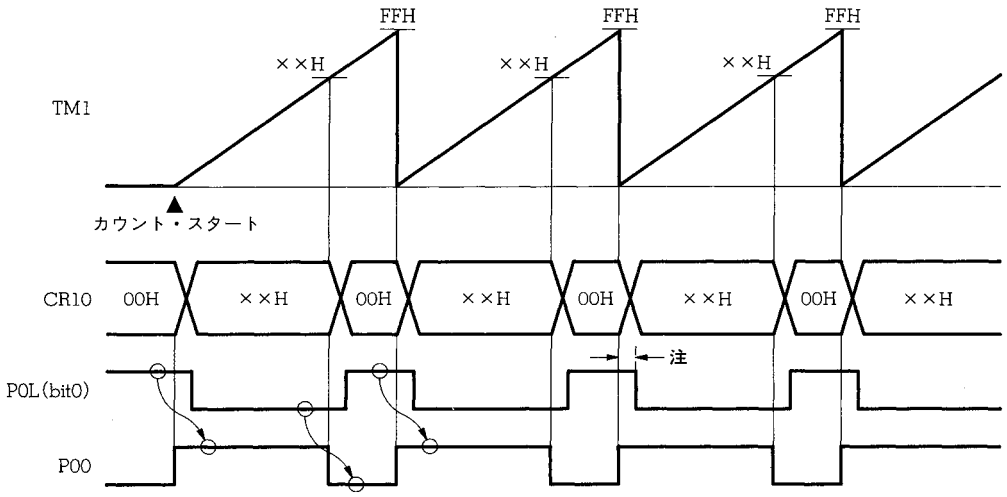
リアルタイム出力ポートとマクロ・サービス・タイプCを使用して，PWM波形のデューティを制御します。

(1) 動作概要

P00端子からPWM出力を行います。

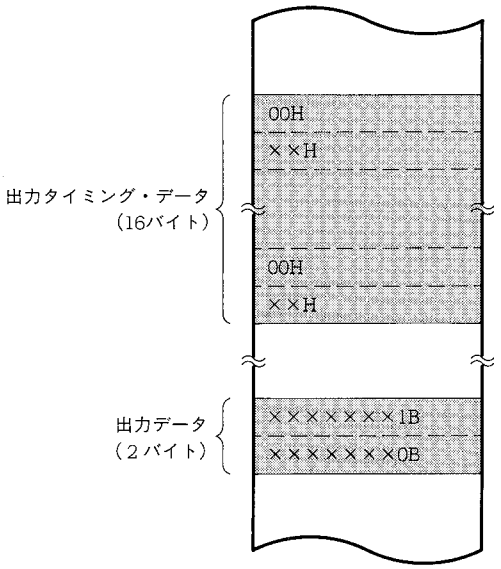
マクロ・サービス・タイプCのモードは，リング制御あり，自動加算なしとします。コンペア・レジスタ（CR10）へ転送するタイミング・データ（××H）でPWM波形のデューティを制御し，バッファ・レジスタ（P0L）へ転送するデータ（××××××××1B，××××××××0B）でPWM波形のレベル（H，L）を制御します（ハイ・アクティブ）。図3-20にタイミング・チャート，図3-21に出力タイミング・データと出力データのメモリ・マップを示します。

図3-20　リアルタイム出力ポートによるPWM出力のタイミング・チャート



注　マクロ・サービスによる遅れ時間

図3－21　出力タイミング・データと出力データのメモリ・マップ

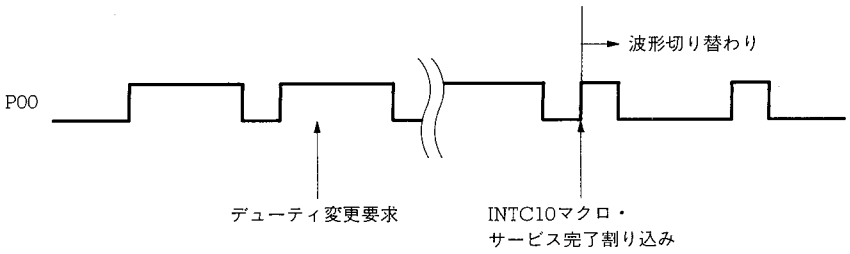


出力タイミング・データの××HでPWM波形のアクティブ幅を決定し，00Hで周期を決定します。周期は，タイマのフルカウント分の時間となります。

PWM周期	＝タイマのカウント・クロック周期 × 256
PWMアクティブ幅	＝タイマのカウント・クロック周期 × ××H

PWM波形のデューティを変更する場合は，RAMにデューティを決定する値を設定し，デューティ変更フラグをセット(1)します。実際に波形が切り替わるのは，指定回数のマクロ・サービスの終了後となります（**図3－22**参照）。

図3－22　PWM波形の切り替わりタイミング



また，初期出力レベルが'H'となるよう，タイマをスタートする前に，P0，P0Lへ××××××××1Bを設定してください。

この応用例における出力可能なPWM波形のパルス幅は，マクロ・サービスに要する時間を考慮し

て，次の範囲となります。

最小：（マクロ・サービス実行時間）＋（ベクタ割り込み受け付け時間）＋ （マクロ・サービス完了割り込み処理時間）＋（マクロ・サービス待ち時間）注
最大：$512/f_{CLK} \times 255 \fallingdotseq 21.8$ ms

注　マクロ・サービス待ち時間とは，命令実行中に割り込み要求が発生し，実際にマクロ・サービスが受け付けられるまでの時間です。

また，1カウント・クロック単位でデューティを制御する場合は，1カウント・クロックが上記の最小パルス幅以上のカウント・クロック（少なくとも$f_{CLK}/128$）を選択する必要があります。

(2) プログラム説明

(a) 処理概要（「(8) SPDチャート」参照）

(i) PWM出力イニシャライズ処理［レーベル名称：PWM_INI］

リアルタイム出力ポート，タイマ，マクロ・サービスのイニシャライズを行います。また，初期のタイミング・データ・テーブルを作成します。

(ii) INTC10マクロ・サービス完了割り込み処理［レーベル名称：INTC10］

デューティ変更フラグ（CHGDTYF）をチェックし，CHGDTYF＝1なら新しいタイミング・データ・テーブルを変更します。CHGDTYF＝0ならマクロ・サービスの再設定のみ行います。

(iii) タイミング・データ・テーブル作成処理［レーベル名称：TIMDATA］

デューティ値設定エリア（DTYDATA）の値と，00Hを，交互にタイミング・データ・バッファに書き込みます。

(b) 使用するRAM

このプログラム例で使用するRAM一覧を表3-23に示します。

3
214
218A
224
234
244

表3－23　リアルタイム出力ポートによるPWM出力で使用するRAM一覧

RAM名称	配置領域	用　　途	バイト数	初期値
TIMBUF	RAM注	タイミング・データを定義	TIMCNT	不定
MSCC10	SADDR注	INTC10マクロ・サービス・カウンタ	1	TIMCNT
RCC10		〃　　リング・カウンタ	1	DATCNT
MODC10		〃　　モジュロ・レジスタ	1	DATCNT
MPDC10P		INTC10データ用マクロ・サービス・ポインタ	2	DATBUF
MPTC10P		INTC10タイミング用マクロ・サービス・ポインタ	2	TIMBUF

注　RAM　：メモリ・バンク0内の任意のRAM領域

SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

(3) 入出力パラメータ

(a) 入力パラメータ

表3－24　リアルタイム出力ポートによるPWM出力の入力パラメータ

パラメータ	固定／可変	内　　容
DTYDATA	可変値	デューティを決定する値
CHGDTYF	可変値	デューティ変更フラグ

(b) 出力パラメータ

なし

(4) 使用レジスタ

表3-25　リアルタイム出力ポートによるPWM出力の使用レジスタ

ルーチン	レーベル	レジスタ	バンク
イニシャライズ	PWM_INI	なし	—
INTC10マクロ・サービス完了割り込み	INTC10	A, B, C, HL	1
タイミング・データ・テーブル作成	TIMDATA	A, B, C, HL	注

注　INTC10マクロ・サービス完了割り込み内でコールされる場合はバンク1，イニシャライズ処理でコールされる場合は，ユーザ・プログラムで使用しているバンクとなります。

(5) 使用スタック

このプログラム例のサブルーチン，および割り込み処理で使用するスタック・サイズは，5バイトです（ネスティング：2レベル）。

(6) モード・レジスタ設定例

リアルタイム出力ポート・コントロール・レジスタ

	7	6	5	4	3	2	1	0	
RTPC	BYTE	0注	0注	P0MH	EXTR	0注	0注	P0ML	（リセットで00H）
設定例	0	0	0	0	0	0	0	1	

P0ML	P0Lの動作指定
0	ポート・モード
1	リアルタイム出力ポート・モード

EXTR	INTP0によるバッファ・レジスタから出力ラッチへのデータ転送の許可
0	許可しない
1	許可する　BYTE=0：P0Lのみ転送される BYTE=1：P0L/Hが転送される

P0MH	P0Hの機能指定
0	ポート・モード
1	リアルタイム出力ポート・モード

BYTE	リアルタイム出力ポート・モードの動作モード
0	4ビット・セパレート・リアルタイム出力ポート
1	8ビット・リアルタイム出力ポート

注　ビット1，2，5，6には必ず“0”を書き込んでください。

キャプチャ／コンペア・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
CRC1	0	0	0	0	CLR11	CM	CLR10	0	（リセットで00H）
設定例	0	0	0	0	0	0	0	0	

CLR10	TM1とCR10レジスタの内容一致でのクリア動作
0	禁止
1	許可

CLR11	CM	CR11レジスタの動作指定	TM1のクリア動作
0	0	コンペア・レジスタ	禁止
1	0		許可（TM1とCR11レジスタの内容一致のとき）
0	1	キャプチャレジスタ	禁止
1	1		許可（TM1の内容をCR11レジスタにキャプチャしたとき）

3
214
218A
224
234
244

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	0	0	0	0	1	0	1	

8ビット・タイマ／カウンタ1

PRS12	PRS11	PRS10	タイマ／カウンタ1のカウント・クロック周波数の指定
0	0	0	$f_{CLK}/16$[注]
0	0	1	
0	1	0	
0	1	1	$f_{CLK}/32$
1	0	0	$f_{CLK}/64$
1	0	1	$f_{CLK}/128$
1	1	0	$f_{CLK}/256$
1	1	1	$f_{CLK}/512$

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	タイマ／カウンタ3のカウント・クロック周波数の指定
0	0	0	0	$f_{CLK}/16$
0	0	0	1	
0	0	1	0	
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	1	1	1	外部クロック(CI)

注　f_{CLK}：内部システム・クロック周波数($f_{XX}/2$)

マクロ・サービス・モード・レジスタ

	7	6	5	4	3	2	1	0
	CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例	1	1	1	0	1	1	0	0

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし	データ転送のみ　POL
1	0	0	1				POH
1	0	1	0				自動加算つき　POL
1	0	1	1				POH
1	1	0	0			リング制御つき	データ転送のみ　POL
1	1	0	1				POH
1	1	1	0				自動加算つき　POL
1	1	1	1				POH

割り込みサービス・モード・レジスタL

	7	6	5	4	3	2	1	0	
ISM0L	CISM11	CISM10	CISM01	CISM00	PISM3	PISM2	PISM1	PISM0	（リセットで00H）
設定例	×	1	×	×	×	×	×	×	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

3
214
218A
224
234
244

タイマ・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	×	×	×	×	1	×	×	×	×：操作しません

8ビット・タイマ／カウンタ1

OVF1	TM1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE1	TM1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

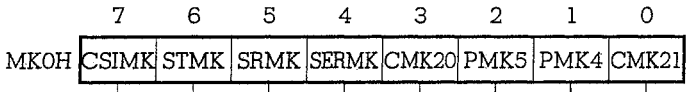
CMD2	TM2のカウント動作モード指定
0	通常モード
1	ワンショット・モード

OVF2	TM2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE2	TM2のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

割り込みマスク・レジスタH



	7	6	5	4	3	2	1	0	
MK0H	CSIMK	STMK	SRMK	SERMK	CMK20	PMK5	PMK4	CMK21	（リセットでFFH）
設定例	×	0	×	×	×	×	×	×	×：操作しません

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

(7) プログラム使用例

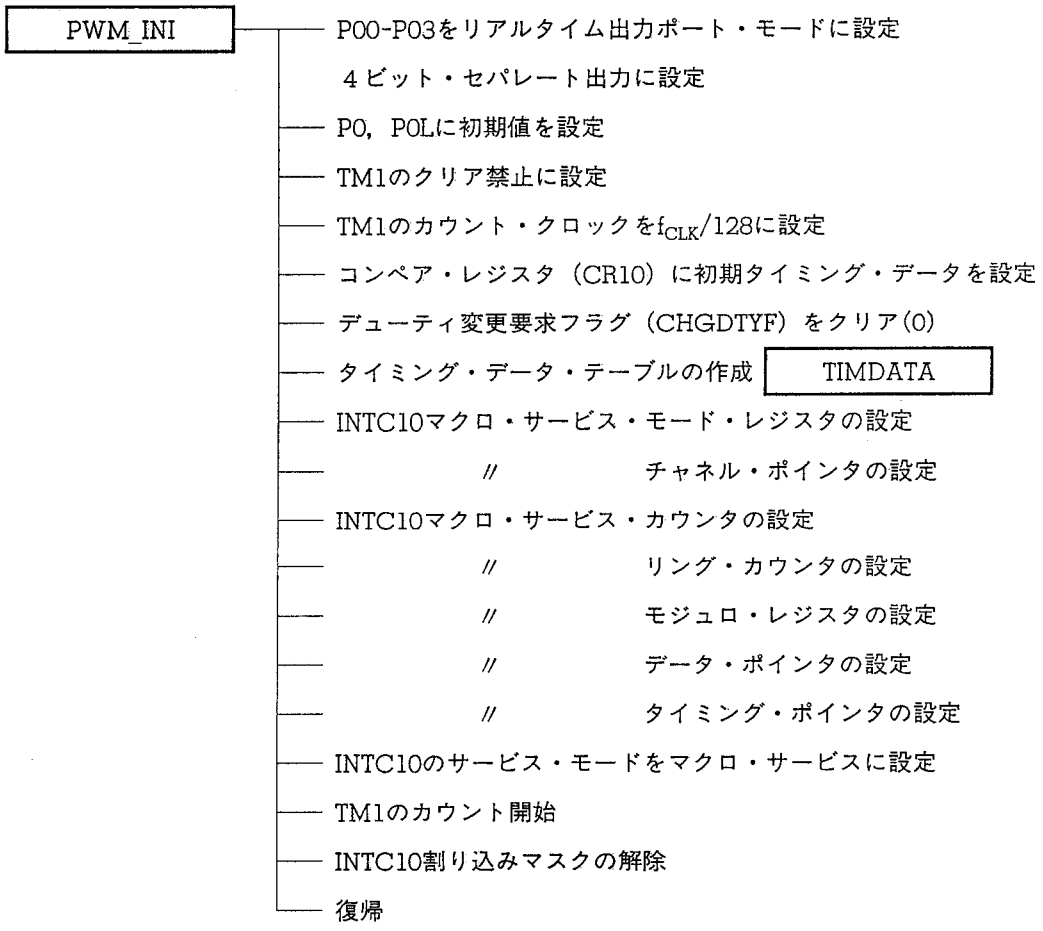
このプログラム例の使用例を示します。デューティを変更する場合は，デューティ変更要求処理内で，デューティ値設定エリア（DTYDATA）にデューティ値を設定してください。

```
        PUBLIC  CHGDTYF,DTYDATA         ; work area
        EXTRN   PWM_INI                 ; package

;       --- define work area ---

        BSEG
CHGDTYF DBIT                            ; request of change duty flag

PWMWKS  DSEG    SADDR
DTYDATA:DS      1                       ; duty data store area
                :
                :
        CSEG
                :
                :
        CALL    !PWM_INI                ; initialize of capture function
        EI                              ; enable interrupt
                :
                :
;       ***change duty***
                :
                :
        DTYDATA=A                       ; A <- duty data
        SET1    CHGDTYF
                :
                :
```

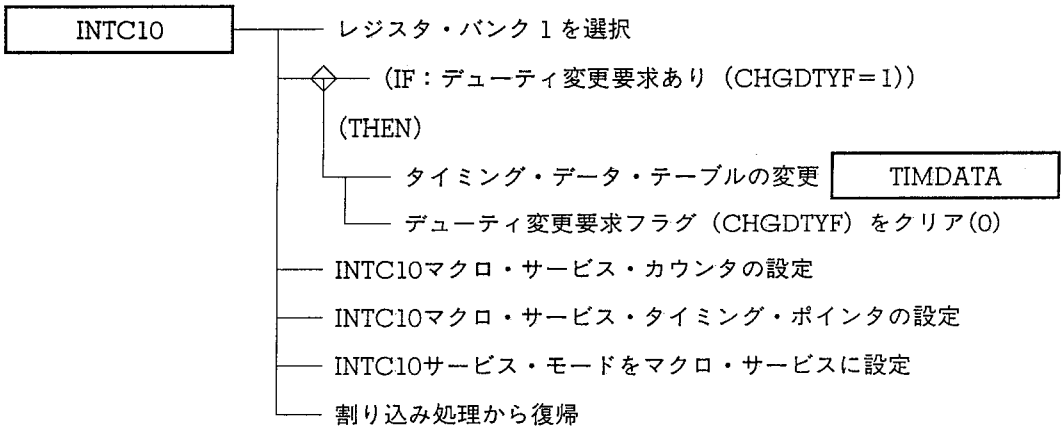
3
214
218A
224
234
244

(8) SPDチャート

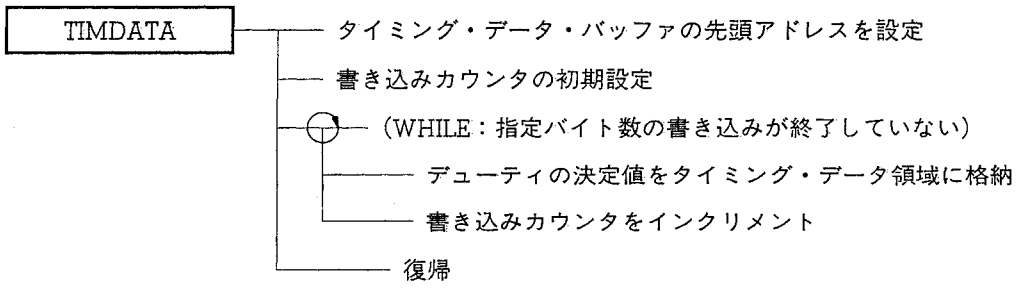
●イニシャライズ処理



●INTC10マクロ・サービス完了割り込み処理



●タイミング・データ・テーブル作成処理



(9) プログラム・リスト

このプログラム例ではソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    RLPWM
;*************************************************************
;*      PWM OUTPUT USED REALTIME OUTPUT PORT                 *
;*************************************************************

        PUBLIC  PWM_INI                 ; package
        EXTRN   DTYDATA                 ; work area
        EXTBIT  CHGDTYF                 ; work flag

;       *** define data ***

LOWDAT  EQU     00000000B               ; LOW level data
HIGHDAT EQU     00000001B               ; HIGH level data
DATCNT  EQU     2                       ; number of output data
TIMCNT  EQU     16                      ; number of timing data

;       *** define macro service area ***

MSBUF   DSEG
TIMBUF: DS      TIMCNT                  ; macro service buffer

        CMCW10                          ; macro service control word
        DSEG_CR C10,'SADDR'             ; macro service channel

;       *** define vector table ***

        CVENT10                         ; INTC10 vector table

;*****************************************
;*      initialize of PWM output        *
;*****************************************

PWMCS   CSEG
PWM_INI:
        RTPC=#00000001B                 ; set P00-P03 to realtime output mode
        P0=#LOWDAT                      ; P0 first output data
        P0L=#HIGHDAT
        CRC1=#00000000B                 ; disable clear TM1
        CR10=#0                         ; first timing data
        PRM1=#00000101B                 ; set prescaler fclk/128
        CLR1    CHGDTYF                 ; set case of no request of change duty
```

```
        CALL    !TIMDATA                ; make timing data
        MSMC10=#11101100B               ; TYPE C, ring control mode
        CHPC10=#(LOW CHAC10)            ; channel pointer
        MSCC10=#TIMCNT                  ; macro service counter
        RCC10=#DATCNT                   ; ring counter
        MODC10=#DATCNT                  ; modulo register
        MPDC10P=#DATBUF                 ; data pointer
        MPTC10P=#TIMBUF                 ; timing data pointer
        SET1    CISM10                  ; set INTC10 macro service mode

        TMC1|=#00001000B                ; start TM1
        CLR1    CMK10                   ; open mask INTC10
        RET

;****************************************
;*      INTC10 interrupt routine        *
;*        end of macro service          *
;****************************************

INTC10:
        SEL     RB1                     ; select register bank 1
        if_bit(CHGDTYF)                 ; request of change duty ?
                CALL    !TIMDATA        ; make timing data
                CLR1    CHGDTYF         ; set case of no request of change duty
        endif

        MSCC10=#TIMCNT                  ; macro service counter
        MPTC10P=#TIMBUF                 ; timing data pointer
        SET1    CISM10                  ; set INTC10 macro service mode
        RETI

;****************************************
;*      make timing data routine        *
;****************************************

TIMDATA:
        HL=#TIMBUF                      ; set timing buffer first address
        BC=#0000H
        while(B<#(TIMCNT/2)) (A)        ; while not end writing data
                [HL+]=DTYDATA (A)       ;     write duty data
                [HL+]=C (A)             ;     write cycle data
                B++                     ;     increment counter
        endw
        RET

;       *** define PWM level data ***

DATBUF:
        DB      LOWDAT                  ; low level data
        DB      HIGHDAT                 ; high level data

        END
```


214
218A
224
234
244

第4章　ウォッチドッグ・タイマ

次の二通りの方法で，ウォッチドッグ・タイマを実現します。

- 外部にタイマを接続する
- 内部タイマを使用する

4.1　外付けタイマによるウォッチドッグ・タイマ

78K/IIの出力ポート（P60）とNMI端子に，ウォッチドッグ回路（μPD74HC123A）を接続します（図4－1参照）。

図4－1　ウォッチドッグ回路の接続ブロック図

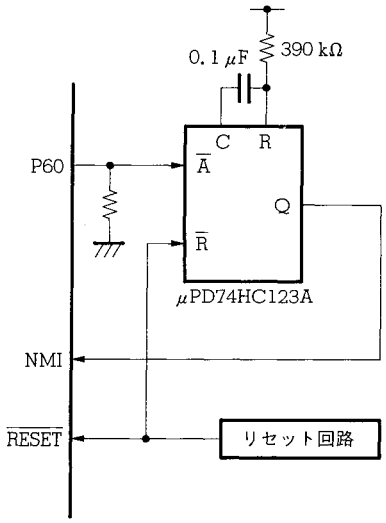


表4－1　μPD74HC123Aの入力，出力の真理値表

入　力		出　力
$\overline{\text{RESET}}$	$\overline{\text{A}}$	Q
H	↓	⊓
↑	L	⊓

μPD74HC123Aは，$\overline{\text{A}}$への入力パルス（トリガ）によって，Qからの出力パルスの幅を拡張する機能を持っています。$\overline{\text{A}}$へのパルス入力がなくなると，CRで決定される時間後（この回路例では，0.1 μF×390 kΩ＝39 ms）にQの出力が反転します。

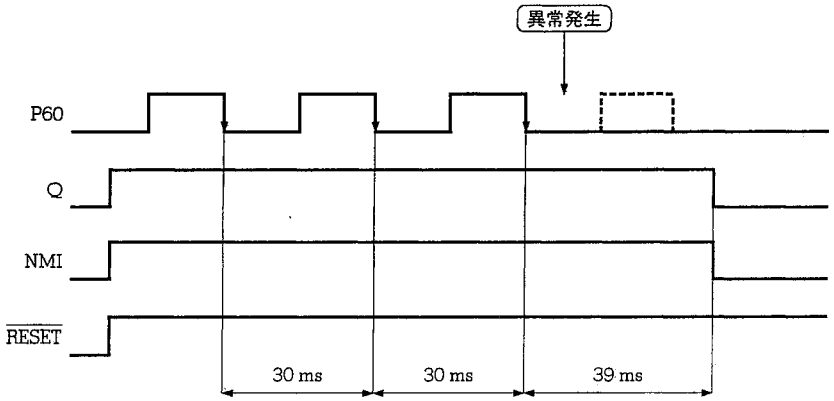
(1) 動作概要

P60から周期が約30 msのパルスをプログラムにより定期的に出力します。プログラムの暴走により，P60からのパルス出力が停止すると，前回のパルス出力から39 ms（図**4－1**　CRによる）後に，μPD74HC123AのQ出力が反転し，NMI端子に有効エッジ（立ち下がり）が入力されます。

また，パワーオン・リセット，およびハードウエア・リセット時は，$\overline{\text{RESET}}$解除（立ち上がりエッジ）により，NMI端子，Q端子ともハイ・レベルとなります。

図4－2にタイミング・チャートを示します。

図4－2　μPD74HC123Aとの接続のタイミング・チャート



注意　ポート（P60）出力の最小パルス幅は，37 ns以上必要です（μPD74HC123Aのスペックより）。

(2) プログラム説明（「(7) プログラム例」参照）

(a) 処理概要

(i) メイン処理［レーベル名称：RST］

メイン処理の一周期でP60をセット／クリアします。プログラムのリセット直後，イニシャライズやRAMクリアなどで30 ms以上かかる場合は，イニシャライズ処理やRAMクリア処理の途中に，P60のセット／クリア処理を入れる必要があります。

(ii) NMI処理［レーベル処理：NMI］

① NMI処理中にマクロ・サービスが発生することのないよう，すべての割り込みをマスクします。

② プログラム・リセット後のNMIと他のすべての割り込みの受け付けを許可し，プログラム先頭アドレスに分岐します。

(b) 使用するRAM

なし

(3) 入出力パラメータ

なし

(4) 使用レジスタ

なし

(5) 使用スタック

このプログラム例のサブルーチン，および割り込み処理で使用するスタック・サイズは，3バイトです（ネスティング：1レベル）。

(6) モード・レジスタ設定例

外部割り込みモード・レジスタ0

	7	6	5	4	3	2	1	0	
INTM0	ES21	ES20	ES11	ES10	ES01	ES00	0	ESNMI	(リセットで00H)
設定例	0	0	0	0	0	0	0	0	

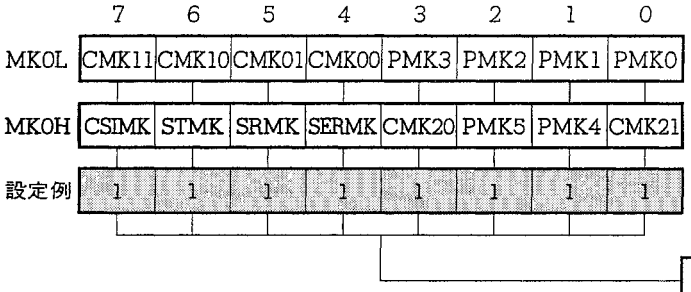
ESNMI	NMI端子入力，検出エッジ指定
0	立ち下がりエッジ
1	立ち上がりエッジ

ES01	ES00	INTP0端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES11	ES10	INTP1端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

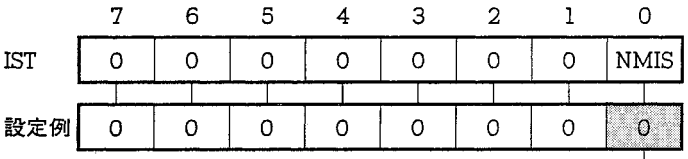
ES21	ES20	INTP2端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

割り込みマスク・レジスタL



MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

割り込みステータス・レジスタ



NMIS	ノンマスカブル割り込み要求による多重割り込みの処理	ノンマスカブル割り込み要求の受け付け状態
0	許可	復帰時，または受け付けていないとき
1	禁止	受け付けたとき，または処理中

4
214
218A
224
234
244

プログラム・ステータス・ワード

	7	6	5	4	3	2	1	0	
PSW	IE	Z	RBSI	AC	RBS0	0	ISP	CY	（リセットで02H）
設定例	×	×	×	×	×	0	1	×	×：操作しません

CY	キャリー・フラグ

ISP	割り込み優先順位ステータス・フラグ
0	低優先順位のベクタ割り込み受け付け禁止
1	優先順位にかかわらずベクタ割り込み受け付け許可

RBS1	RBS0	レジスタ・バンク選択フラグ
0	0	レジスタ・バンク0選択状態
0	1	レジスタ・バンク1選択状態
1	0	レジスタ・バンク2選択状態
1	1	レジスタ・バンク3選択状態

AC	補助キャリー・フラグ

Z	ゼロ・フラグ

IE	割り込み要求許可フラグ
0	ベクタ割り込み受け付け禁止（DI状態）
1	ベクタ割り込み受け付け許可（EI状態）

(7) プログラム例

メイン処理の一周期でポートをセット／クリアする場合の，メイン処理とNMI処理のプログラム例を示します。このプログラム例では，ソース・プログラムの先頭でベクタ・テーブル・アドレスとマクロ・サービス領域定義用ファイル（INTMS.DEF）をインクルードしています。

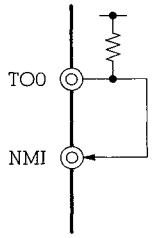
```
$       IC(INTMS.DEF)
;       NAME    WDT_EXT
;*************************************************************
;*      WATCH DOG TIMER BY EXTERNAL TIMER                    *
;*************************************************************

NMIS    EQU     IST.0                   ; NMI status flag
ISP     EQU     PSW.1                   ; interrupt service priority flag
PLSOUTP EQU     P6.0                    ; pulse output port

;       --- define vector table ---

        RSTVENT                         ; RESET vector table
        NMIVENT                         ; NMI vector table

;****************************************
;*      main routine                    *
;****************************************

WDTEBS  CSEG
RST:
        SP=#STACK                       ; stack pointer
        PM6=#11110000B                  ; memory bank 0
        MM=#10000000B                   ; single chip mode
                :
                :
        INTM0=#00000000B                ; set NMI falling edge
        EI                              ; enable interrupt
                :
                :
        while(forever)                  ; LOOP forever
                SET1    PLSOUTP
                CLR1    PLSOUTP
                :
                :
        endw
                :
                :
;****************************************
;*      NMI routine                     *
;****************************************

NMI:
        MK0=#0FFFFH                     ; mask all interrupt
        CLR1    NMIS                    ; enable NMI
        SET1    ISP                     ; enable all maskable interrupt
        BR      RST                     ; branch RESET address
                :
                :
```

4
214
218A
224
234
244

4.2　内部タイマによるウォッチドッグ・タイマ

16ビット・タイマ／カウンタのタイマ出力端子（TO0）とNMI端子を接続し，ウォッチドッグ・タイマを実現します（**図4－3**参照）。

図4－3　内部タイマによるウォッチドッグ・タイマのブロック図

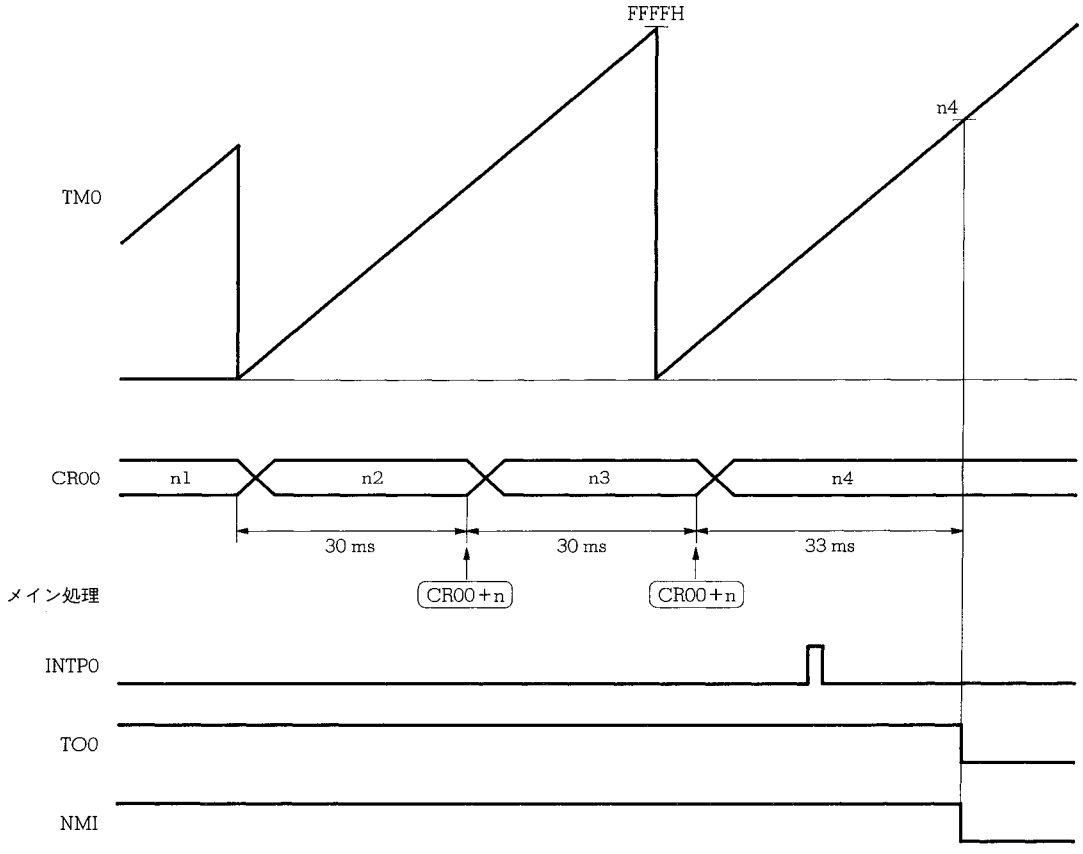


（1）動作概要

16ビット・タイマのコンペア・レジスタ（CR00）値を，30 msごとにプログラムで更新します。

プログラムの暴走により，CR00値の更新が行われなくなると，前回の更新から33 ms後に，TM0とCR00の一致信号が発生し，TO0出力レベルが反転します（ロウ・アクティブ）。このエッジ（立ち下がり）が，NMI端子に入力されます。図4－4にタイミング・チャートを示します。

図4－4　内部タイマによるウォッチドッグ・タイマのタイミング・チャート



(2) プログラム説明

(a) 処理概要（「(7) プログラム例」参照）

(i) メイン処理［レーベル名称：RST］

① タイマ／カウンタとタイマ出力のイニシャライズを行います。

② メイン処理の一周期で，16ビット・コンペア・レジスタ（CR00）値を1回更新します。プログラムのリセット直後，イニシャライズやRAMクリアなどで，30 ms以上かかる場合は，イニシャライズ処理やRAMクリア処理の途中に，CR00値の更新処理を入れる必要があります。

(ii) NMI処理［レーベル名称：NMI］

① NMI処理中にマクロ・サービスが発生することのないよう，すべての割り込みをマスクします。

② タイマのカウントとタイマ出力を停止します。

③ プログラム・リセット後のNMIと他のすべての割り込みの受け付けを許可し，プログラム先頭アドレスに分岐します。

(b) 使用するRAM

なし

(3) 入出力パラメータ

なし

(4) 使用レジスタ

なし

(5) 使用スタック

このプログラム例のサブルーチン，および割り込み処理で使用するスタック・サイズは，3バイトです（ネスティング：1レベル）。

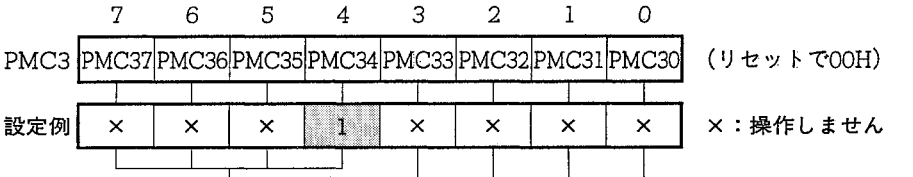
(6) モード・レジスタ設定例

キャプチャ／コンペア・コントロール・レジスタ0

	7	6	5	4	3	2	1	0	
CRC0	MOD1	MOD0	0	1	CLR01	0	0	0	(リセットで10H)
設定例	0	0	0	1	0	0	0	0	

MOD1	MOD0	CLR01	タイマ出力モード指定		TM0＝CR01時の
			TO0	TO1	TM0のクリア動作
0	0	0	トグル出力	トグル出力	禁止
0	0	1	トグル出力	トグル出力	許可
0	1	0	PWM出力	トグル出力	禁止
0	1	1	設定禁止		
1	0	0	PWM出力	PWM出力	禁止
1	0	1	設定禁止		
1	1	0	設定禁止		
1	1	1	PPG出力	トグル出力	許可

ポート3モード・コントロール・レジスタ



PMC30	P30端子のコントロール・モード指定
0	入出力ポート・モード
1	RxD入力モード

PMC31	P31端子のコントロール・モード指定
0	入出力ポート・モード
1	TxD出力モード

PMC32	P32端子のコントロール・モード指定
0	入出力ポート・モード
1	$\overline{\text{SCK}}$入出力モード

PMC33	P33端子のコントロール・モード指定
0	入出力ポート・モード
1	SO出力／SB0入出力モード

PMC3n	PM3n端子のコントロール・モード指定（n=4-7）
0	入出力ポート・モード
1	TOm出力モード（m=0-3）

214
218A
224
234
244

タイマ出力コントロール・レジスタ

	7	6	5	4	3	2	1	0	
TOC	ENT03	ALV3	ENT02	ALV2	ENT01	ALV1	ENT00	ALV0	（リセットで00H）
設定例	0	0	0	0	0	0	1	0	

ALV0	TO0端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT00	TO0端子の動作指定
0	$\overline{\text{ALV0}}$を出力
1	パルス出力許可

ALV1	TO1端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT01	TO1端子の動作指定
0	$\overline{\text{ALV1}}$を出力
1	パルス出力許可

ALV2	TO2端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT02	TO2端子の動作指定
0	$\overline{\text{ALV2}}$を出力
1	パルス出力許可

ALV3	TO3端子のアクティブ・レベル指定
0	ロウ・レベル
1	ハイ・レベル

ENT03	TO3端子の動作指定
0	$\overline{\text{ALV3}}$を出力
1	パルス出力許可

外部割り込みモード・レジスタ0

	7	6	5	4	3	2	1	0	
INTM0	ES21	ES20	ES11	ES10	ES01	ES00	0	ESNMI	(リセットで00H)
設定例	0	0	0	0	0	0	0	0	

ESNMI	NMI端子入力，検出エッジ指定
0	立ち下がりエッジ
1	立ち上がりエッジ

ES01	ES00	INTP0端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES11	ES10	INTP1端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES21	ES20	INTP2端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

4
214
218A
224
234
244

タイマ・コントロール・レジスタ0

	7	6	5	4	3	2	1	0	
TMC0	CE3	0	0	0	CE0	OVF0	0	0	（リセットで00H）
設定例	×	×	×	×	1	×	×	×	

16ビット・タイマ／カウンタ

OVF0	TM0のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFFFHから0000Hへのカウント・アップ)

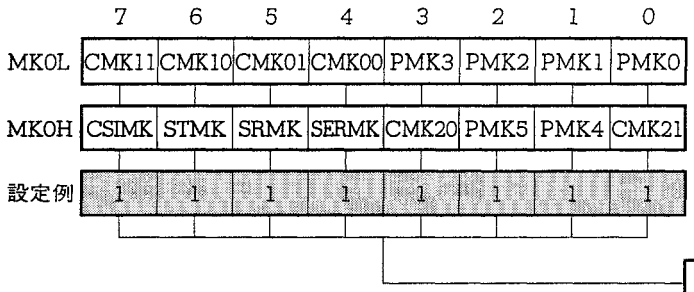
備考　このビットはソフトウエアでのみリセットされます。

CE0	TM0のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ・ユニット3

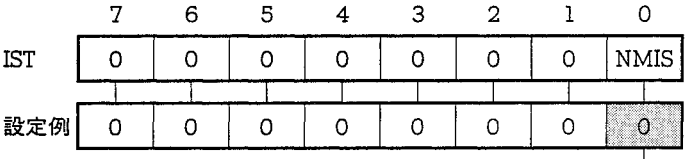
CE3	TM3のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

割り込みマスク・レジスタL



MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

割り込みステータス・レジスタ



NMIS	ノンマスカブル割り込み要求による多重割り込みの処理	ノンマスカブル割り込み要求の受け付け状態
0	許可	復帰時，または受け付けていないとき
1	禁止	受け付けたとき，または処理中

4
214
218A
224
234
244

プログラム・ステータス・ワード

	7	6	5	4	3	2	1	0	
PSW	IE	Z	RBSI	AC	RBS0	0	ISP	CY	（リセットで02H）
設定例	×	×	×	×	×	0	1	×	×：操作しません

CY	キャリー・フラグ

ISP	割り込み優先順位ステータス・フラグ
0	低優先順位のベクタ割り込み受け付け禁止
1	優先順位にかかわらずベクタ割り込み受け付け許可

RBS1	RBS0	レジスタ・バンク選択フラグ
0	0	レジスタ・バンク０選択状態
0	1	レジスタ・バンク１選択状態
1	0	レジスタ・バンク２選択状態
1	1	レジスタ・バンク３選択状態

AC	補助キャリー・フラグ

Z	ゼロ・フラグ

IE	割り込み要求許可フラグ
0	ベクタ割り込み受け付け禁止（DI状態）
1	ベクタ割り込み受け付け許可（EI状態）

(7) プログラム例

メイン処理の一周期でコンペア・レジスタ（CR00）値を更新する場合の，メイン処理とNMI処理のプログラム例を示します。このプログラム例では，ソース・プログラムの先頭でベクタ・テーブル・アドレスとマクロ・サービス領域定義用ファイル（INTMS.DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    WDT_INT
;*************************************************************
;*      WATCH DOG TIMER BY INTERNAL TIMER                    *
;*************************************************************

NMIS    EQU     IST.0                   ; NMI status flag
ISP     EQU     PSW.1                   ; interrupt service priority flag
PLSOUTP EQU     P6.0                    ; pulse outoput port
INTVL   EQU     24750                   ; TM0 interval data

;       --- define vector table ---

        RSTVENT                         ; RESET vector table
        NMIVENT                         ; NMI vector table

;****************************************
;*      main routine                    *
;****************************************

WDTIBS  CSEG
RST:
        SP=#STACK                       ; stack pointer
        PM6=#11110000B                  ; memory bank 0
        MM=#10000000B                   ; single chip mode
                :
                :
        CRC0=#00010000B                 ; disable clear TM0
        CR00=#INTVL                     ; set TM0 interval data
        PMC3|=#00010000B                ; set P34 control port
        TOC=#00000010B                  ; T00 active low, enable timer output
        INTM0=#00000000B                ; set NMI falling edge
        TMC0|=#00001000B                ; start TM0
                :
                :
        EI                              ; enable interrupt
        while(forever)                  ; LOOP forever
                :
                :
                AX=TM0                  ; read TM0 count data
                AX+=#INTVL              ; update next interval data
                CR00=AX
                :
                :
        endw
```

4
214
218A
224
234
244

```
;****************************************
;*      NMI routine                     *
;****************************************

NMI:
        MK0=#0FFFFH                     ; mask all interrupt
        TOC=#00000000B                  ; stop timer output
        TMC0=#00000000B                 ; stop timer
        CLR1    NMIS                    ; enable NMI
        SET1    ISP                     ; enable all maskable interrupt
        BR      RST                     ; branch RESET address
                :
                :
```

第５章　リアルタイム出力ポートによるステッピング・モータの制御

リアルタイム出力ポートとマクロ・サービスを使用した，ステッピング・モータの開ループ制御の応用プログラムを２種類示します。

- 高分解能タイミングによる制御
- ２軸同時制御

なお，基本的なステッピング・モータの制御については，**78K/IIシリーズ　アプリケーション・ノート　基礎編（IEA-607）**をご覧ください。

5.1　高分解能タイミングによる制御

ステッピング・モータの制御をマクロ・サービス・タイプCで行う場合，タイミング制御の分解能は，タイマ（TM1）の分解能に依存して８ビットとなります。

ここでは，タイミング制御の分解能を上げた例を示します。

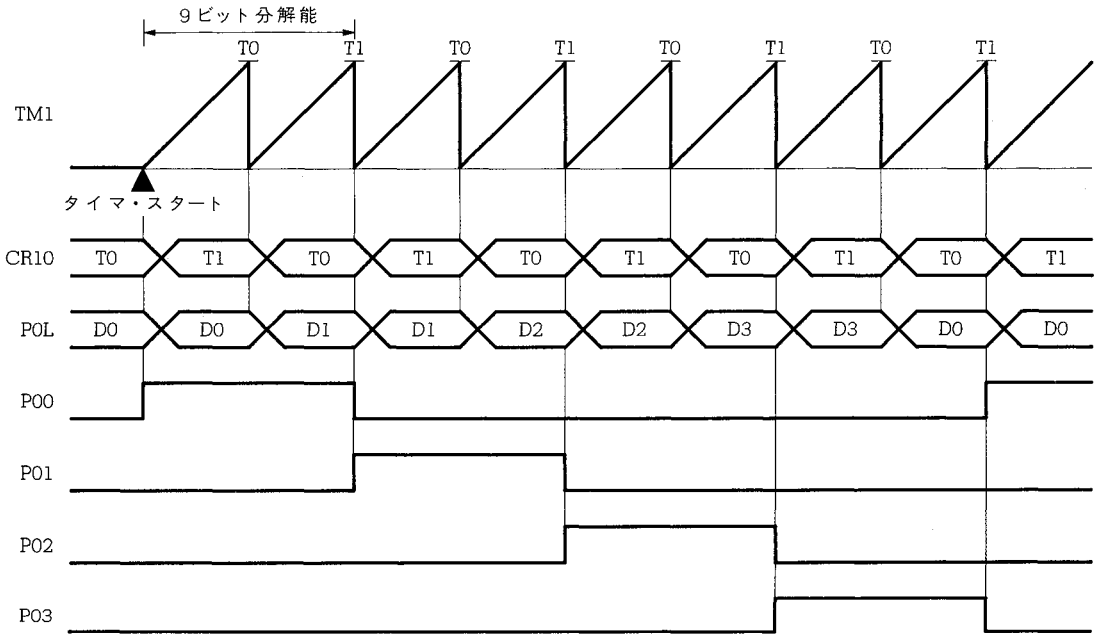
（1）動作概要

ステッピング・モータの回転速度200 ppsを，９ビットの分解能で制御します。タイマのカウント・クロックは，$f_{CLK}/256$を選択します。

８ビット・タイマ／カウンタ１の２回分のカウントを利用することによって，$2^9=512$（９ビット）の分解能が得られます。図５－１に９ビット分解能制御のタイミング・チャートを示します。

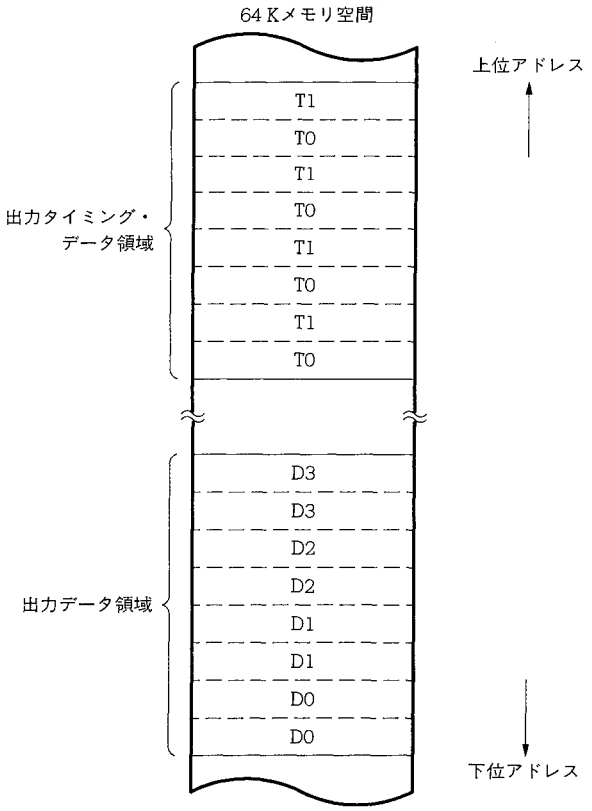
214
218A
224
234
244

図5－1　9ビット分解能のタイミング制御のタイミング・チャート



T0, T1のそれぞれのタイミングでD0のデータを出力すると，TM1が2回カウントする間，D0のデータを出力し続け，結果的に9ビットの分解能が得られます。このタイミングを実現するために，メモリに図5－2のような形でデータを持っておきます。

図5-2　9ビット分解能のタイミング制御のメモリ・マップ



タイミング・データ（T0，T1）は，次の式により求めます。

（1ステップ時間/カウント・クロック周期)/2-1

商はT0のデータとし，計算結果の余り（整数部のみ）を，商に加えてT1のデータとします。200 ppsの場合，1ステップ5 msのタイミングとなるので，次式によりT0，T1は57，78となります。

$(5\ \mathrm{ms}/(256/f_{CLK}))/2-1=57\cdots1$

なお，タイマのカウント・クロックは，8ビット分解能で制御した場合に対し，2倍の速さのカウント・クロックを選択することになります。

5
214
218A
224
234
244

(2) プログラム説明

(a) 処理概要（「(8) SPDチャート」参照）

(i) イニシャライズ処理［レーベル名称：MCR_INI］

リアルタイム出力ポート，タイマ・マクロ・サービスのイニシャライズを行います。

(ii) INTC10マクロ・サービス完了割り込み処理［レーベル名称：INTC10］

マクロ・サービスの再設定を行います。

(b) 使用するRAM

このプログラム例で使用するRAM一覧を示します。

表5-1　リアルタイム出力ポートによるステッピング・モータの制御の使用RAM一覧

RAM名称	配置領域	用　　途	バイト数	初期値
MSCC10	SADDR[注]	INTC10マクロ・サービス・カウンタ	1	DATCNT
MPDC10P		INTC10データ用マクロ・サービス・ポインタ	2	PTRNDAT
MSTC10P		INTC10タイミング用マクロ・サービス・ポインタ	2	TIMEDAT

注　SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

(3) 入出力パラメータ

なし

(4) 使用レジスタ

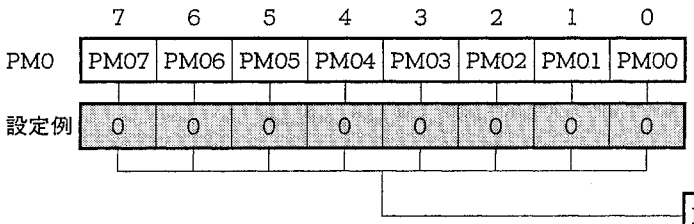
なし

(5) 使用スタック

このプログラム例のサブルーチン，および割り込み処理で使用するスタック・サイズは，3バイトです（ネスティング：1レベル）。

(6) モード・レジスタ設定例

ポート0モード・レジスタ



PM0n	P0n端子入出力モード指定（n＝0-7）
00H	出力モード（出力バッファ・オン）
FFH	ハイ・インピーダンス状態（出力バッファ・オフ）
上記以外	設定禁止

5
214
218A
224
234
244

リアルタイム出力ポート・コントロール・レジスタ

	7	6	5	4	3	2	1	0	
RTPC	BYTE	0注	0注	POMH	EXTR	0注	0注	POML	（リセットで00H）
設定例	0	0	0	0	0	0	0	1	

POML	POLの動作指定
0	ポート・モード
1	リアルタイム出力ポート・モード

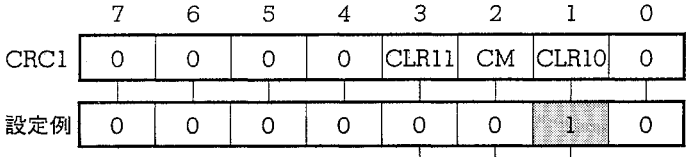
EXTR	INTPOによるバッファ・レジスタから出力ラッチへのデータ転送の許可
0	許可しない
1	許可する　BYTE=0：POLのみ転送される BYTE=1：POL/Hが転送される

POMH	POHの機能指定
0	ポート・モード
1	リアルタイム出力ポート・モード

BYTE	リアルタイム出力ポート・モードの動作モード
0	4ビット・セパレート・リアルタイム出力ポート
1	8ビット・リアルタイム出力ポート

注　ビット1，2，5，6には必ず“0”を書き込んでください。

キャプチャ／コンペア・コントロール・レジスタ1



CLR10	TM1とCR10レジスタの内容一致でのクリア動作
0	禁止
1	許可

CLR11	CM	CR11レジスタの動作指定	TM1のクリア動作
0	0	コンペア・レジスタ	禁止
1	0		許可（TM1とCR11レジスタの内容一致のとき）
0	1	キャプチャ・レジスタ	禁止
1	1		許可（TM1の内容をCR11レジスタにキャプチャしたとき）

214
218A
224
234
244

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	0	0	0	0	1	1	0	

8ビット・タイマ／カウンタ1

PRS12	PRS11	PRS10	タイマ／カウンタ1のカウント・クロック周波数の指定
0	0	0	$f_{CLK}/16$[注]
0	0	1	
0	1	0	
0	1	1	$f_{CLK}/32$
1	0	0	$f_{CLK}/64$
1	0	1	$f_{CLK}/128$
1	1	0	$f_{CLK}/256$
1	1	1	$f_{CLK}/512$

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	タイマ／カウンタ3のカウント・クロック周波数の指定
0	0	0	0	$f_{CLK}/16$
0	0	0	1	
0	0	1	0	
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	1	1	1	外部クロック(CI)

注　f_{CLK}：内部システム・クロック周波数($f_{XX}/2$)

マクロ・サービス・モード・レジスタ

7	6	5	4	3	2	1	0
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例 1	1	1	0	1	0	0	0

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			データ転送のみ	POL
1	0	0	1				POH
1	0	1	0			自動加算つき	POL
1	0	1	1				POH
1	1	0	0			データ転送のみ	POL
1	1	0	1				POH
1	1	1	0			自動加算つき	POL
1	1	1	1				POH

リング制御なし

リング制御つき

割り込みサービス・モード・レジスタL

7	6	5	4	3	2	1	0	
ISM0L	CISM11	CISM10	CISM01	CISM00	PISM3	PISM2	PISM1	PISM0	（リセットで00H）
設定例	×	1	×	×	×	×	×	×	×：操作しません

ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

5
214
218A
224
234
244

タイマ・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	×	×	×	×	1	×	×	×	×：操作しません

8ビット・タイマ／カウンタ1

OVF1	TM1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE1	TM1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

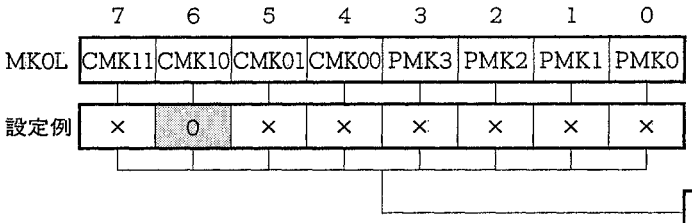
CMD2	TM2のカウント動作モード指定
0	通常モード
1	ワンショット・モード

OVF2	TM2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE2	TM2のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

割り込みマスク・レジスタL



MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

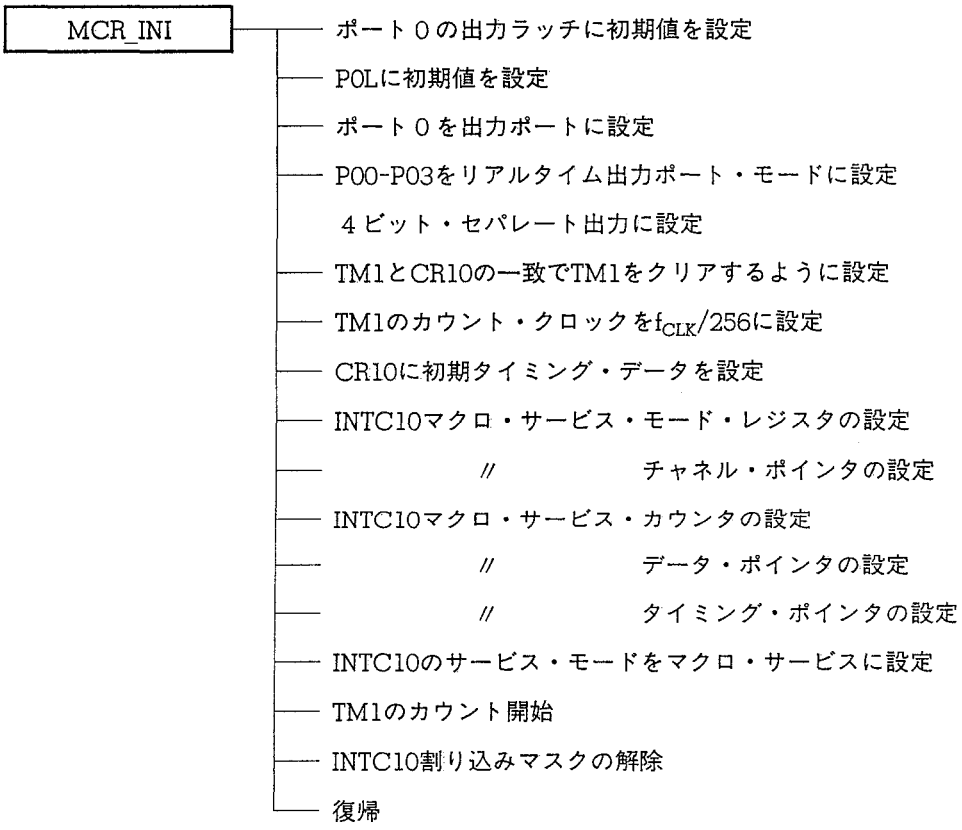
(7) プログラム使用例

このプログラムの使用例を示します。

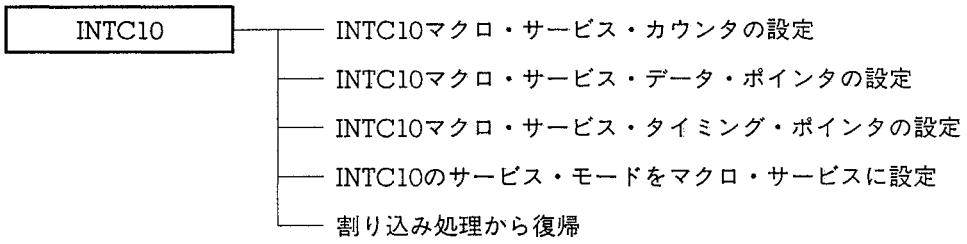
```
CSEG
        :
        :
CALL    !MCR_INI
EI
        :
        :
```

(8) SPDチャート

●マクロ・サービス・イニシャライズ処理



●INTC10マクロ・サービス完了割り込み処理



(9) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    STEP
;*************************************************************
;*      OPEN LOOP CONTROL OF STEPPING MOTOR                  *
;*************************************************************

        PUBLIC  MCR_INI                 ; package

;       *** define data ***

DATCNT  EQU     8

;       *** define macro service area ***

        CMCW10                          ; macro service control word
        DSEG_C  C10,'SADDR'             ; macro service channel

;       *** define vector table ***

        CVENT10                         ; INTC10 vector table

;****************************************
;*      initialize of macro service     *
;****************************************

STEPCS  CSEG
MCR_INI:
        P0=#00H                         ; set first pattern data
        P0L=#00H
        PM0=#00000000B                  ; set P0 output port
        RTPC=#00000001B                 ; set P00-P03 to realtime output mode
        CRC1=#00000010B                 ; enable clear TM1 by CR10
        PRM1=#00000110B                 ; set prescaler fclk/256
        CR10=#0

        MSMC10=#11101000B               ; TYPE C
        CHPC10=#(LOW CHAC10)            ; channel pointer
        MSCC10=#DATCNT                  ; macro service counter
        MPDC10P=#PTRNDAT                ; pattern data pointer
        MPTC10P=#TIMEDAT                ; timing data pointer
        SET1    CISM10                  ; set INTC10 macro service mode

        TMC1|=#00001000B                ; start TM1
        CLR1    CMK10                   ; open mask INTC10
        RET
```

5
214
218A
224
234
244

```
;****************************************
;*      INTC10 interrupt routine        *
;*        end of macro service          *
;****************************************

INTC10:
        MSCC10=#DATCNT                  ; macro service counter
        MPDC10P=#PTRNDAT                ; pattern data pointer
        MPTC10P=#TIMEDAT                ; timing data pointer
        SET1    CISM10                  ; set INTC10 macro service mode
        RETI

;       *** define pattern data of motor cycle ***

PTRNDAT:
        DB      00000001B
        DB      00000001B
        DB      00000010B
        DB      00000010B
        DB      00000100B
        DB      00000100B
        DB      00001000B
        DB      00001000B

;       *** define timing data of motor cycle ***

TIMEDAT:                                ; about 5ms
        DB      57
        DB      58
        DB      57
        DB      58
        DB      57
        DB      58
        DB      57
        DB      58

        END
```

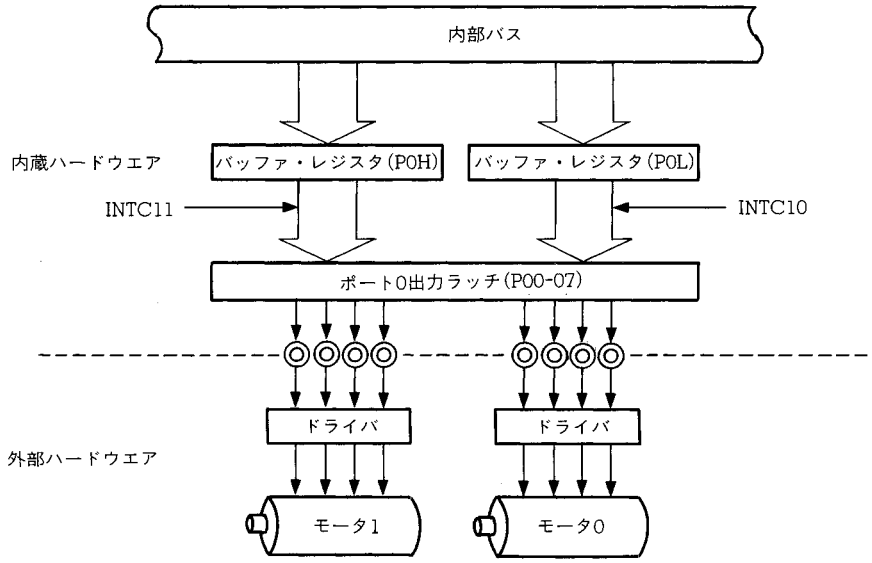
5.2　2軸同時制御

リアルタイム出力ポートをマクロ・サービス機能により制御し，ステッピング・モータの開ループ制御を行います。マクロ・サービスはタイプCのリング制御および自動加算モードを用いることにより2軸同時制御を行います。

ステッピング・モータは，入力パルスに応じてステップ動作します。したがって，DCモータや，ACモータのように，回転数検出のためのロータリ・エンコーダや位置検出を行うセンサを用いて閉ループ・システムを構成しなくても，容易に制御を行えます。

この応用例で用いるステッピング・モータ制御のハードウエア・ブロック図を図5-3に示します。

図5-3　ステッピング・モータ制御のハードウエア・ブロック図



(1) 動作概要

ここでは，制御例として，POL，POHにそれぞれ接続されている4相ステッピング・モータを2軸同時制御します。ただし，内部システム・クロック（f_{CLK}）は，5 MHzとします。モータの回転の制御は，NMI，INTP0，INTP4，INTP5の外部割り込み要求によって行います。動作プロフィールを図5-4に示します。

図5-4　ステッピング・モータの2軸同時制御のプロフィール

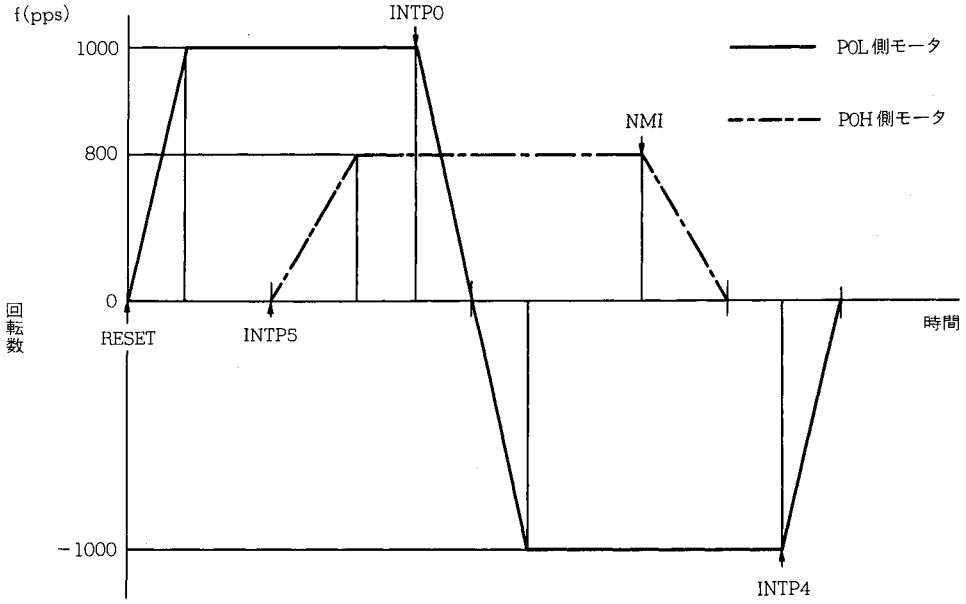


表5-2　モータ別のパラメータ

	励磁パターン	最大ステップ・レート	直線加速時間	加減速ステップ数
モータ0	1-2相励磁	1000 pps	20 ms	10
モータ1	1相励磁	800 pps	15 ms	6

表5-3　外部割り込み要求に対する処理

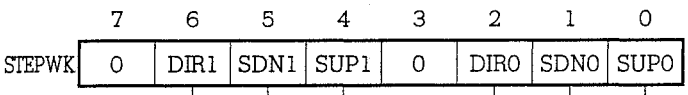
	RESET	INTP0	INTP4	INTP5	NMI
モータ0	正回転加速	逆転	停止	−	−
モータ1	−	−	−	正回転加速	停止

(2) プログラム説明…(7) プログラム・リスト参照

(a) フラグ管理

それぞれのモータの状態はフラグで管理されています。次にフラグを示します。

図5-5　フラグ説明



SDN0	SUP0	モータ0の状態
0	0	停止
0	1	加速
1	0	減速
1	1	停止

DIR0	モータ0の回転方向
0	正回転
1	逆回転

SDN1	SUP1	モータ1の状態
0	0	停止
0	1	加速
1	0	減速
1	1	停止

DIR1	モータ1の回転方向
0	正回転
1	逆回転

5
214
218A
224
234
244

(b) ステップ・レートの計算方法

定速度F_nに達するまでのステップ数Nは，

$$N = \frac{1}{2}(T_n - T_0) \times (F_n - F_0)$$

直線加速度βは，

$$\beta = \frac{F_n}{T_n}$$

また，T_m，T_{m+1}で囲まれる部分の面積Sが1ステップに等しいので，

$$S = \frac{1}{2}(F_{m+1} + F_m) \times (T_{m+1} - T_m)$$

$$= \frac{\beta}{2}(T_{m+1} + T_m) \times (T_{m+1} - T_m)$$

$$S = \frac{\beta}{2}(T_{m+1}^2 - T_m^2) = 1$$

$$\therefore T_{m+1} = \sqrt{\frac{2}{\beta} + T_m^2}$$

以上の式に，表5-2のパラメータを設定すると，各ステップ間の時間は表5-4，表5-5のようになります。

図5-6　加速プロフィール

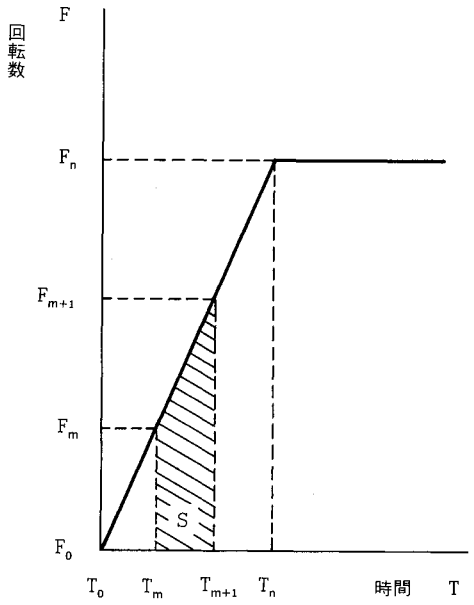


表5-4　モータ0のステップ・レートのパラメータ

ステップ間	時間（ms）
$T_1 - T_0$	6.32
$T_2 - T_0$	8.94
$T_3 - T_0$	10.95
$T_4 - T_0$	12.65
$T_5 - T_0$	14.14
$T_6 - T_0$	15.49
$T_7 - T_0$	16.79
$T_8 - T_0$	17.89
$T_9 - T_0$	18.97
$T_{10} - T_0$	20.00

ステップ間	時間（ms）
$T_1 - T_0$	6.32
$T_2 - T_1$	2.62
$T_3 - T_2$	2.01
$T_4 - T_3$	1.69
$T_5 - T_4$	1.49
$T_6 - T_5$	1.35
$T_7 - T_6$	1.24
$T_8 - T_7$	1.16
$T_9 - T_8$	1.09
$T_{10} - T_9$	1.03
定速度領域	1.00

表5-5　モータ1のステップ・レートのパラメータ

ステップ間	時間（ms）
T_1-T_0	6.12
T_2-T_0	8.66
T_3-T_0	10.61
T_4-T_0	12.25
T_5-T_0	13.69
T_6-T_0	15.00

ステップ間	時間（ms）
T_1-T_0	6.12
T_2-T_1	2.54
T_3-T_2	1.95
T_4-T_3	1.64
T_5-T_4	1.45
T_6-T_5	1.31
定速度領域	1.25

ここで内部システム・クロック：f_{CLK}=5 MHz，8ビット・タイマ／カウンタ1のカウント・クロックを$f_{CLK}/128$とした場合，各ステップ間に設定するタイミング・データの値は次式で求められます。

$$\text{各ステップ間のタイミング・データ}=\frac{T_n-T_{n-1}}{128/f_{CLK}}$$

この式より算出したタイミング・データを表5-6，表5-7に示します。

表5-6　モータ0のタイミング・データ

ステップ間	タイミング・データ（HEX）
T_1-T_0	F7
T_2-T_1	66
T_3-T_2	4F
T_4-T_3	42
T_5-T_4	3A
T_6-T_5	35
T_7-T_6	30
T_8-T_7	2D
T_9-T_8	2A
$T_{10}-T_9$	28
定速度領域	27

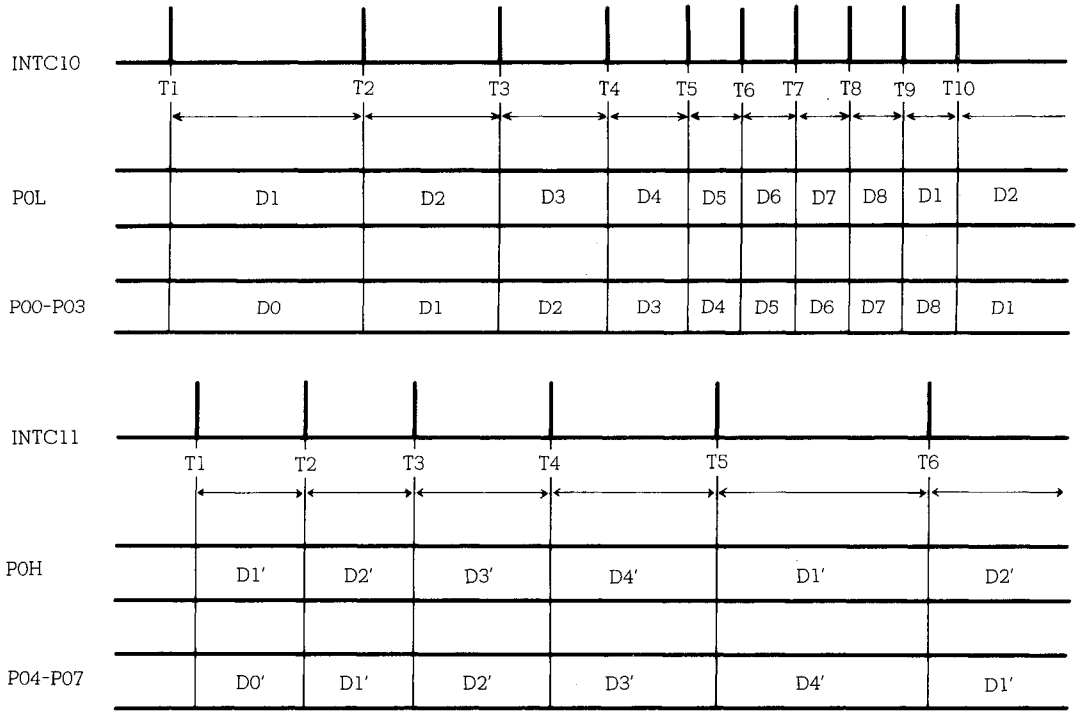
表5-7　モータ1のタイミング・データ

ステップ間	タイミング・データ（HEX）
T_1-T_0	EF
T_2-T_1	63
T_3-T_2	4C
T_4-T_3	40
T_5-T_4	38
T_6-T_5	33
定速度領域	31

(c) マクロ・サービス・タイプCの動作

リアルタイム出力ポートの機能とマクロ・サービス・タイプCを組み合わせることによりステッピング・モータなどの開ループ制御が可能です。8ビット・タイマ／カウンタ1からの2本の割り込み要求INTC10，INTC11とリアルタイム出力ポートの動作についてタイミング・チャートを図5-7に示します。

図5-7　マクロ・サービス・タイプCのタイミング・チャート



2系統のリアルタイム出力ポートを制御するためには，1つのタイマでコントロールするのでタイマのクリア動作ができません。

この場合には，タイマは0-FFHのフリーランニングで動作させ，マクロ・サービスはタイプCの中でも，自動加算モード，リング制御機能付きを用います。このモードの場合はマクロ・サービス・カウンタにはステッピング・モータに与える総パルス数を設定します。また，定速度領域ではタイマ用マクロ・サービス・ポインタを保持するモードを用います。

(d) 2軸同時制御の場合の転送データ

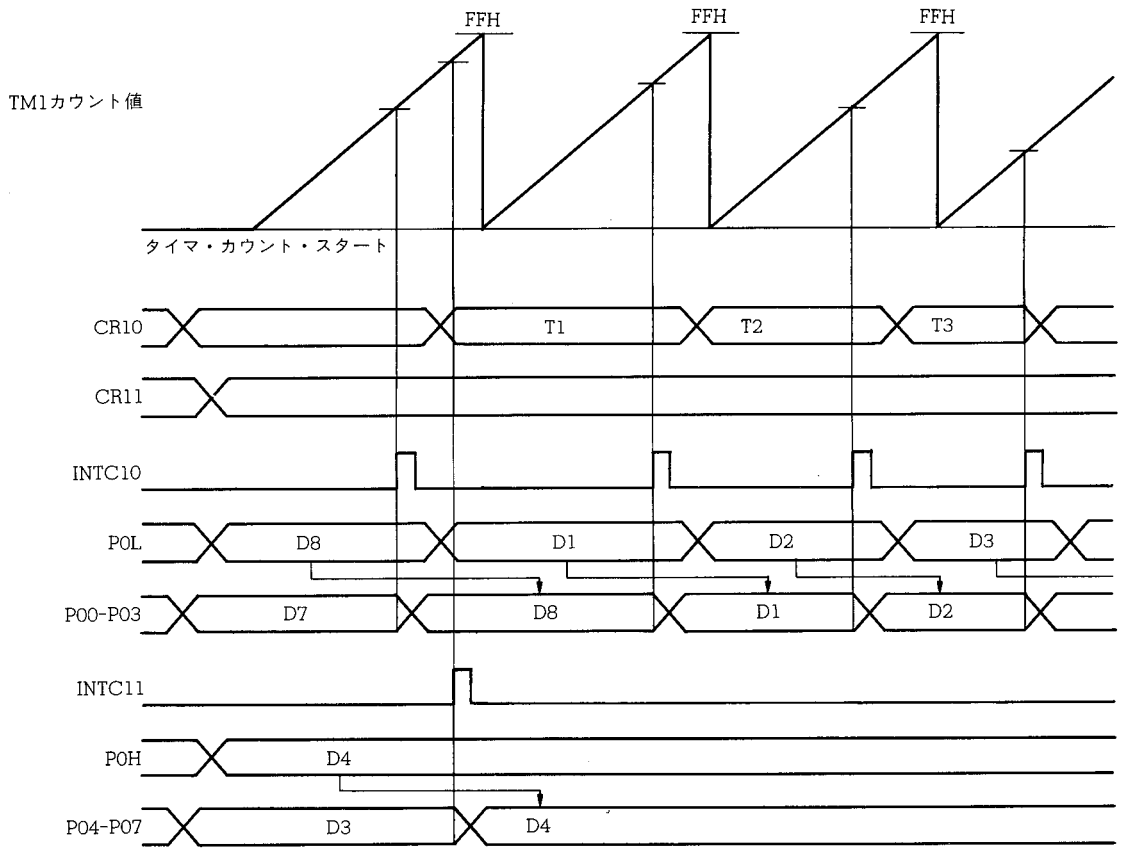
2系統のステッピング・モータを1つのタイマにより起動する場合，1系統のみの場合に対して，次の処理が必要となります。

(i) 初期起動時の動作

POLへのマクロ・サービス転送を行う場合，CR10の設定値とTM1の一致によるINTC10割り込み要求によって起動しますが，このとき，CR11とTM1の一致によってINTC11割り込み要求も発生します。

ここで，リアルタイム出力ポートのバッファ・レジスタ（POL，POH）から出力ラッチへの転送はマクロ・サービスの機能ではなく，リアルタイム出力ポートの機能としてハードウエア的に転送が行われます。したがって，ポート0の上位，下位ともにリアルタイム出力ポートの動作モードに設定すると，2系統とも動作してしまいます。

図5-8　初期起動時の動作

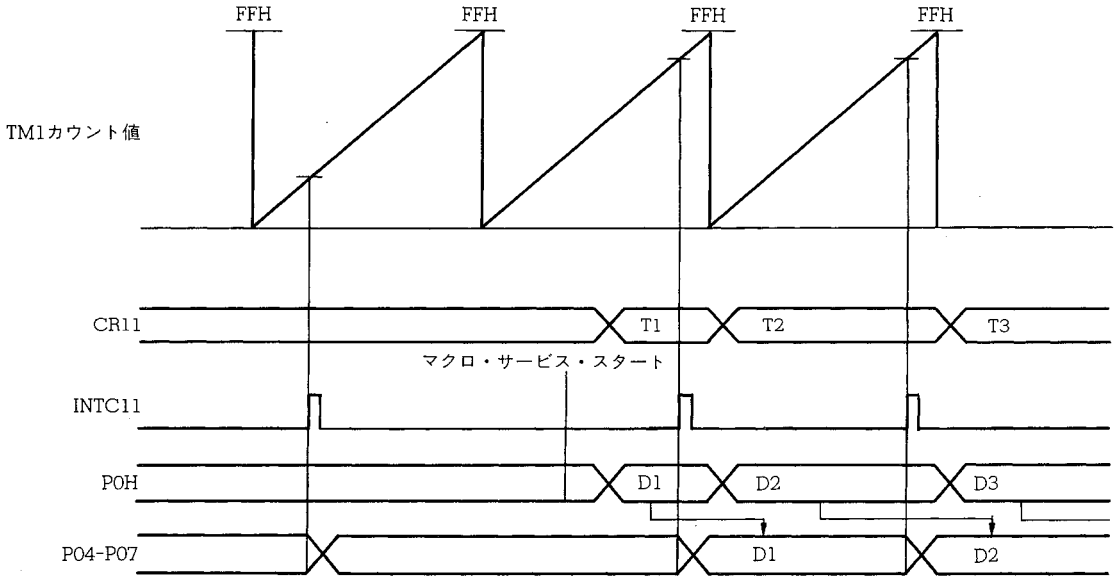


この状態を避けるため，リアルタイム出力ポートの動作設定を上位，下位別々に行うか，または，出力ラッチと同じ値をバッファ・レジスタに設定しておきます。

(ii) 停止状態からのスタート時のステップ・レート

タイマが動作している状態で，停止状態のモータをスタートする場合には，すでに割り込み要求が発生している場合が考えられます。この状態でマクロ・サービスを起動するとINTC10の受け付けと，タイマのカウント値は非同期となるため，初回のステップ・レートが未確定の状態となります。

図5-9　停止状態からのスタート時のステップ・レート



このように，最長のステップ・レートに相当する部分がなくなり，2ステップ目のステップ・レートでモータが起動されます。回避方法として次の2つの方法があります。

① 最長のタイミング・データを2つ用意する。
② 2ステップ目のステップ・レートで起動された場合でも，モータが脱調しないようにステップ・レートを設定する。

この応用例では①の方法を採用しています。

ほかに，マクロ・サービスをスタートする前に，INTC11割り込み要求フラグをクリア(0)する方法が考えられますが，CIF11フラグのクリア(0)とマクロ・サービスのスタートとの間に，CIF11がセット(1)されたり，あるいは，別の割り込み要求の発生による割り込み処理が行われた場合は，最長のステップ・レートのタイミングが生成できなくなります。

(e) マクロ・サービスの使用状態

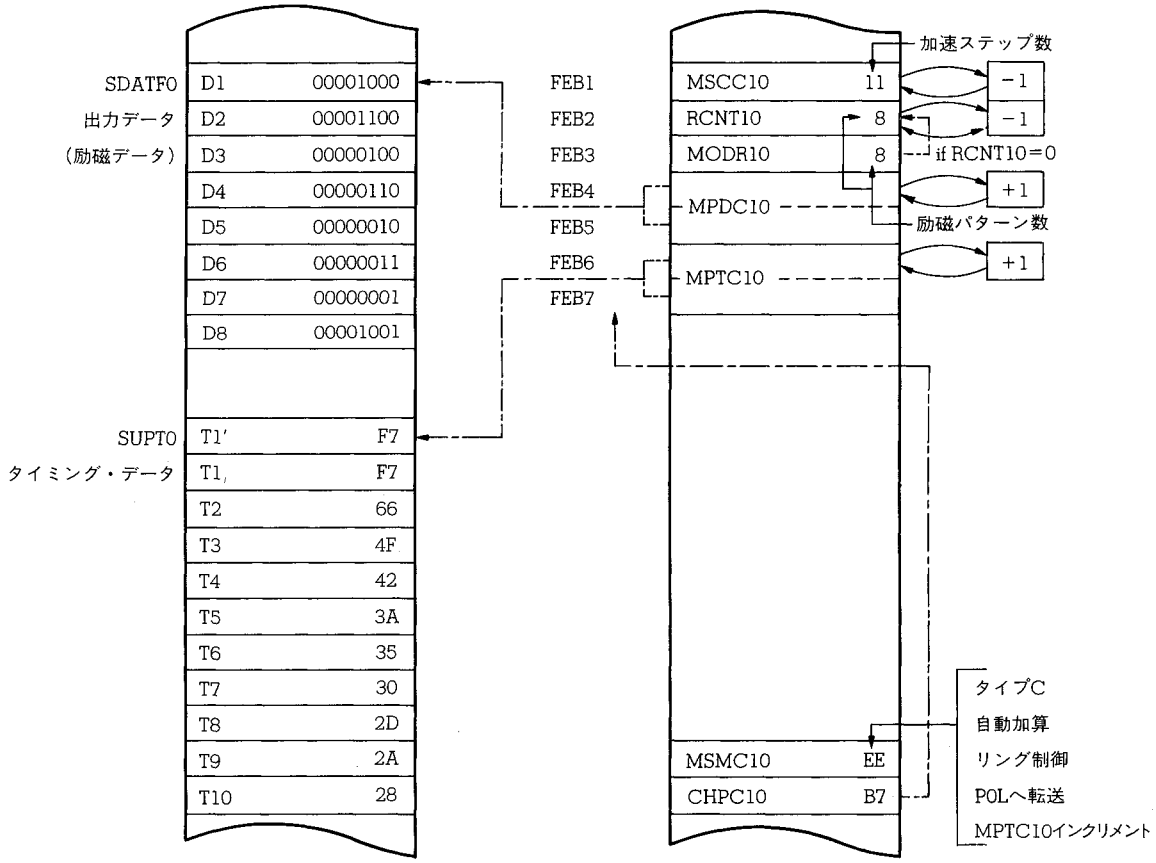
マクロ・サービスはタイプCのリング制御，自動加算モードを用いますが，タイマ用マクロ・サービス・ポインタをインクリメントするモードと保持するモードの両方を状態に応じて使用しています。

この制御は，マクロ・サービス・モード・レジスタのCHT0のオン／オフによって行います。加速，減速時にはマクロ・サービス・ポインタをインクリメントするモードを，定速度時にはマクロ・サービス・ポインタを保持するモードを用いています。

ここでは，例としてモータ0の正回転加速時と正回転定速度時について，マクロ・サービスの使用状態をメモリ・マップで示します。

(i) モータ0正回転加速時

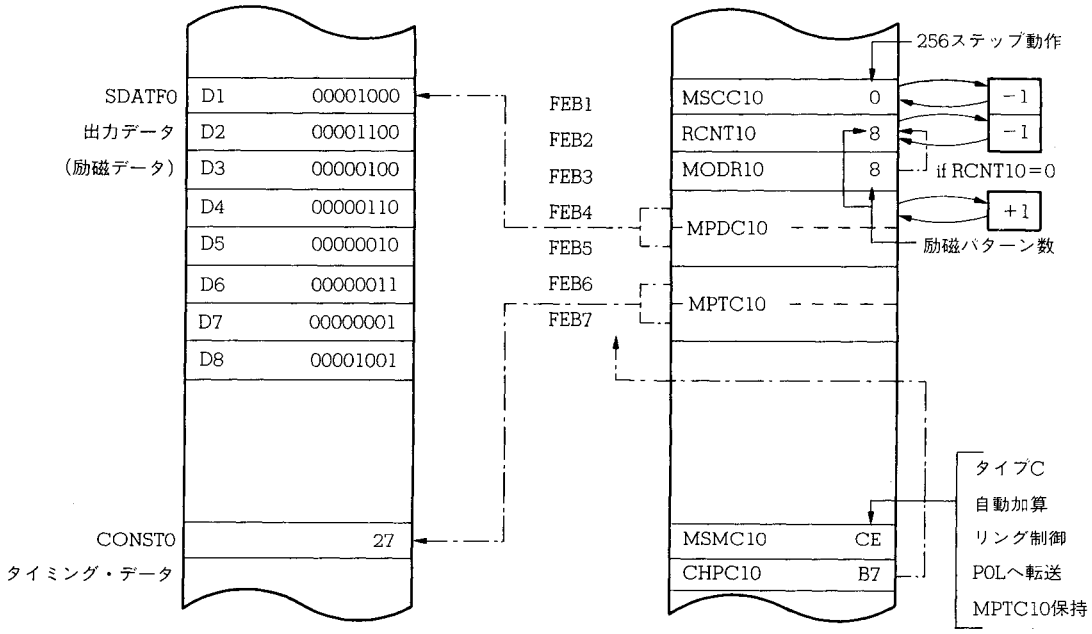
図5-10　モータ0正回転加速時のマクロ・サービスの使用状態



5
214
218A
224
234
244

(ii) モータ0正回転定速度時

図5-11　モータ0正回転定速度時のマクロ・サービスの使用状態



(f) 逆転制御

逆転の要求は，この応用例の場合，外部割り込み要求によって発生します。このとき，減速状態とし，減速が完了後，逆転の加速を行います。

逆転制御では2つの処理が必要となります。

(i) ステップ・レートの変更，マクロ・サービス・ポインタのインクリメント・モードへの移行。

(ii) 出力データ・ファイルの切り替え

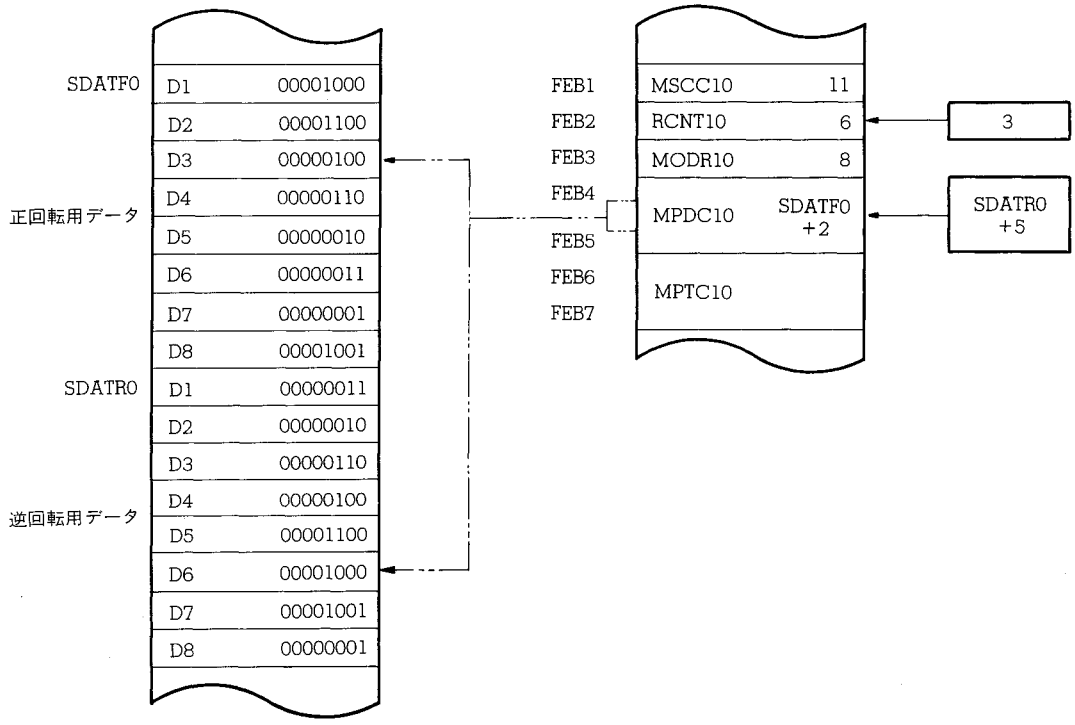
(i)の処理は，タイマ用マクロ・サービス・ポインタの変更とマクロ・サービス・モード・レジスタの設定値変更によって行います。

(ii)の処理は，減速完了時に行うもので，マクロ・サービス・ポインタとリング・カウンタの変更が必要です。この場合，減速完了時のバッファ・レジスタの値が断定できないため，減速完了時のリング・カウンタの値を用いて，マクロ・サービス・ポインタとリング・カウンタの再設定を行っています。この処理では，マクロ・サービス転送によってマクロ・サービス・チャネルの内容が変化するのを防止するため，割り込みをマスクしています。

図5－12のように励磁データを配置し，データ用マクロ・サービス・ポインタとリング・カウンタの値を次式によって算出します。

- データ用マクロ・サービス・ポインタ(MPDC10)＝逆転用データ・ファイル先頭アドレス(SDATR0)＋リング・カウンタ(RCNT10)－1
- リング・カウンタ (RCNT10)＝励磁データ数(8)＋1－リング・カウンタ(RCNT10)

図5－12　逆転時のマクロ・サービス・ポインタとリング・カウンタの変更



以上より，それぞれのモータのデータ・ファイルは次のような形で用意しています。

```
;*************************
;***   DATA PROFILE   ***
;*************************
;
;       *** motor 0 ***
;
PARAM0  CSEG    PAGE

MODDT0: DB      8                       ; number of step pattern
UDSTP0: DB      11                      ; number of speed up/down step
;
;       *** data profile ***
;
SDATF0: DB      00001000B               ; forward
        DB      00001100B
        DB      00000100B
        DB      00000110B
        DB      00000010B
        DB      00000011B
        DB      00000001B
        DB      00001001B
SDATR0:
        DB      00000011B               ; reverse
        DB      00000010B
        DB      00000110B
        DB      00000100B
        DB      00001100B
        DB      00001000B
        DB      00001001B
        DB      00000001B
;
;       *** timing data profile ***
;
SUPT0:  DB      0F7H                    ; speed up timing for dummy cycle
        DB      0F7H, 66H, 4FH, 42H, 3AH ; speed up timing
        DB      35H, 30H, 2DH, 2AH, 28H
CONST0: DB      27H                     ; constant timing
SDNT0:  DB      28H, 2AH, 2DH, 30H, 35H ; speed down timing
        DB      3AH, 42H, 4FH, 66H, 0F7H
        DB      0F7H                    ; speed down timing for dummy cycle
$       EJECT
;
;       *** motor 1 ***
;
PARAM1  CSEG    PAGE

MODDT1: DB      4                       ; number of step pattern
UDSTP1: DB      7                       ; number of speed up/down step
;
```

```
;       *** data profile ***
;
SDATF1:
        DB      10000000B               ; forward
        DB      01000000B
        DB      00100000B
        DB      00010000B
SDATR1:
        DB      01000000B               ; reverse
        DB      10000000B
        DB      00010000B
        DB      00100000B
;
;       *** timing data profile ***
;
SUPT1:  DB      0EFH                    ; speed up timing for dummy cycle
        DB      0EFH,63H,4CH,40H        ; speed up timing
        DB      38H,33H
CONST1: DB      31H                     ; constant timing
SDNT1:  DB      33H,38H,40H,4CH         ; speed down timing
        DB      63H,0EFH
        DB      0EFH                    ; speed down timing for dummy cycle
;
        END
```

(g) 外部割り込みおよびマクロ・サービス完了割り込みの処理

(i) RESET入力

システム・イニシャライズ処理ならびにモータ0のスタートを行います。

(ii) INTC10割り込み処理（マクロ・サービス完了割り込み）

モータ0に設定した一連の動作が終了した状態です。ここでは次の3つの状態が考えられます。

① 加速終了

次に定速度状態に設定します。

② 定速状態での256ステップ動作時

再び定速状態に設定します。

③ 減速終了

順方向に回転した場合は逆転，逆方向に回転した場合は停止とします。

5
214
218A
224
234
244

(iii) INTC11割り込み処理（マクロ・サービス完了割り込み）

モータ1に設定した一連の動作が終了した状態です。ここでは次の3つの状態が考えられます。

① 加速終了

次に定速度状態に設定します。

② 定速度状態での256ステップ動作時

再び定速度状態に設定します。

③ 減速終了

停止します。

(iv) INTP0入力

モータ0の減速を行います。減速が終了するとマクロ・サービス完了割り込みによって逆転動作に移行します。この処理ではINTC10割り込み要求をマスクしています。

(v) INTP4入力

モータ0の減速を行います。減速が終了するとマクロ・サービス完了割り込みによってモータ0は停止します。この処理ではINTC10割り込み要求をマスクしています。

(vi) INTP5入力

モータ1のスタートを行います。

(vii) NMI入力

モータ1の減速を行います。減速が終了するとマクロ・サービス完了割り込みによってモータ1は停止します。この処理ではINTC10割り込み要求をマスクしています。

(3) モード・レジスタ設定例

ポート0モード・レジスタ

	7	6	5	4	3	2	1	0	
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	（リセットでFFH）
設定例	0	0	0	0	0	0	0	0	

PM0n	P0n端子入出力モード指定（n=0-7）
00H	出力モード（出力バッファ・オフ）
FFH	ハイ・インピーダンス状態（出力バッファ・オン）
上記以外	設定禁止

リアルタイム出力ポート・コントロール・レジスタ

	7	6	5	4	3	2	1	0	
RTPC	BYTE	0[注]	0[注]	POMH	EXTR	0[注]	0[注]	POML	（リセットで00H）
設定例	0	0	0	0	0	0	0	1	

POML	POLの動作指定
0	ポート・モード
1	リアルタイム出力ポート・モード

EXTR	INTPOによるバッファ・レジスタから出力ラッチへのデータ転送の許可
0	許可しない
1	許可する　BYTE=0：POLのみ転送される BYTE=1：POL/Hが転送される

POMH	POHの機能指定
0	ポート・モード
1	リアルタイム出力ポート・モード

BYTE	リアルタイム出力ポート・モードの動作モード
0	4ビット・セパレート・リアルタイム出力ポート
1	8ビット・リアルタイム出力ポート

注　ビット1，2，5，6には必ず“0”を書き込んでください。

5
214
218A
224
234
244

タイマ・コントロール・レジスタ1

	7	6	5	4	3	2	1	0
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0
設定例	0	0	0	0	1	0	0	0

（リセットで00H）

8ビット・タイマ／カウンタ1

OVF1	TM1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFHから00Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE1	TM1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

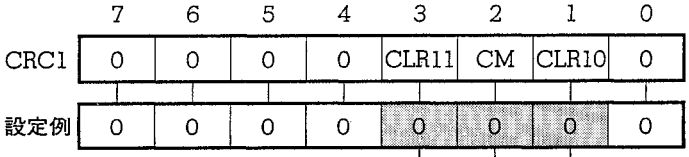
CMD2	TM2のカウント動作モード指定
0	通常モード
1	ワンショット・モード

OVF2	TM2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFHから00Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE2	TM2のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

キャプチャ／コンペア・コントロール・レジスタ1



（リセットで00H）

CLR10	TM1とCR10レジスタの内容一致でのクリア動作
0	禁止
1	許可

CLR11	CM	CR11レジスタの動作指定	TM1のクリア動作
0	0	コンペア・レジスタ	禁止
1	0		許可（TM1とCR11レジスタの内容一致のとき）
0	1	キャプチャ・レジスタ	禁止
1	1		許可（TM1の内容をCR11レジスタにキャプチャしたとき）

5
214
218A
224
234
244

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	0	0	0	0	1	0	1	

8ビット・タイマ／カウンタ1

PRS12	PRS11	PRS10	カウント・クロック［Hz］の指定
0	0	0	$f_{CLK}/16$注
0	0	1	
0	1	0	
0	1	1	$f_{CLK}/32$
1	0	0	$f_{CLK}/64$
1	0	1	$f_{CLK}/128$
1	1	0	$f_{CLK}/256$
1	1	1	$f_{CLK}/512$

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	カウント・クロック［Hz］の指定
0	0	0	0	$f_{CLK}/16$
0	0	0	1	
0	0	1	0	
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	1	1	1	外部クロック（CI）

注　f_{CLK}：内部システム・クロック

マクロ・サービス・モード・レジスタ

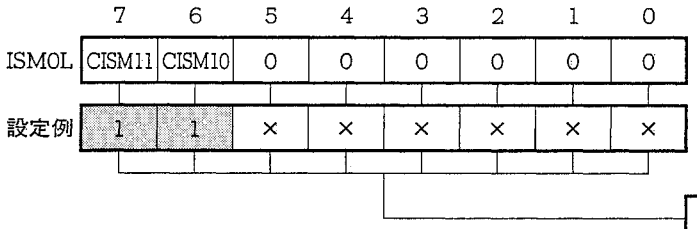
7	6	5	4	3	2	1	0	
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0	（リセットで00H）
設定例 1	1	—	0	1	1	1	0	

CHT0	0	1	0	1	
CHT1	1	0	1		
CHT2	0	1	1		

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし	データ転送のみ POL
1	0	0	1				POH
1	0	1	0				自動加算つき POL
1	0	1	1				POH
1	1	0	0			リング制御つき	データ転送のみ POL
1	1	0	1				POH
1	1	1	0				自動加算つき POL
1	1	1	1				POH

5
214
218A
224
234
244

割り込みサービス・モード・レジスタL



ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

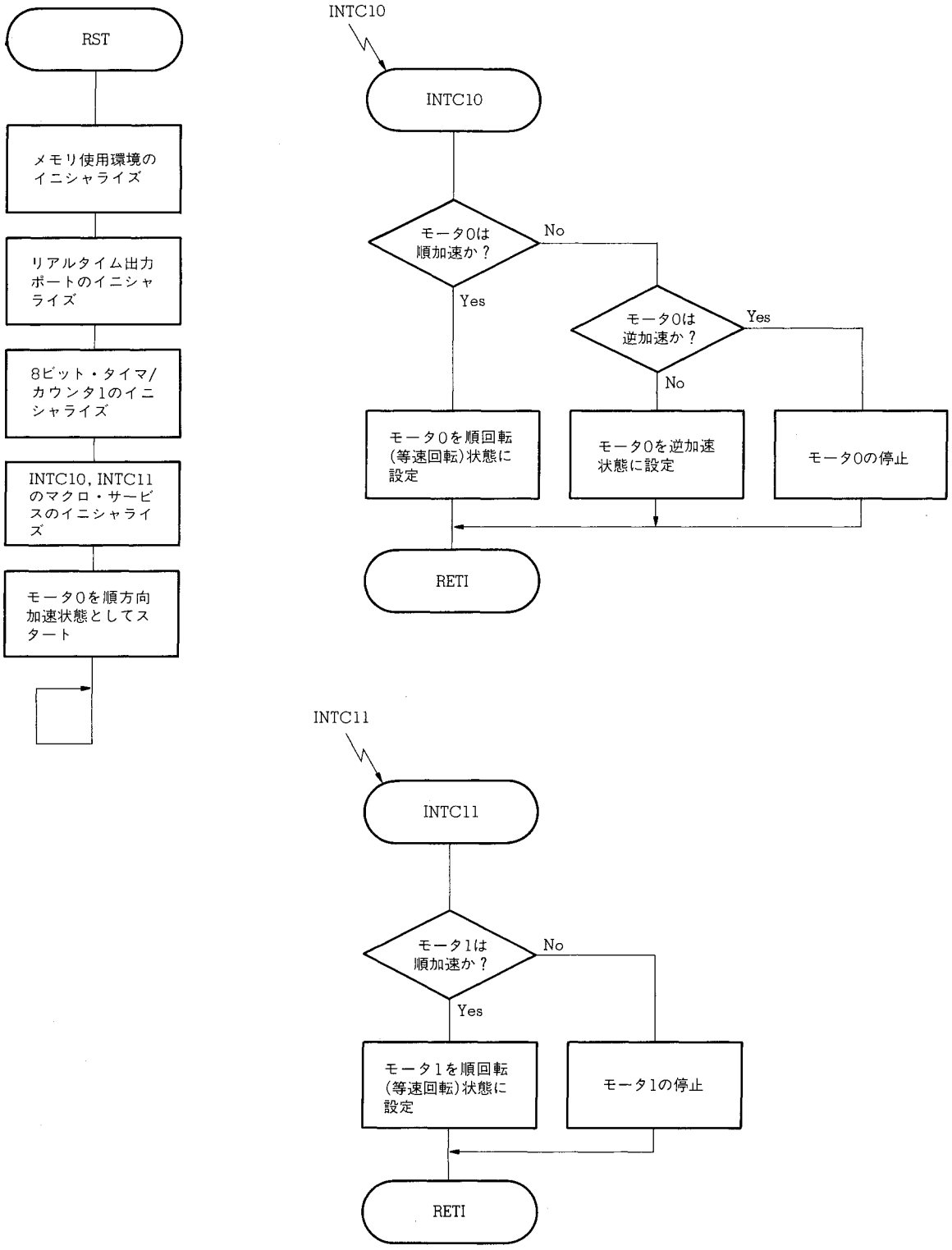
割り込みマスク・レジスタL

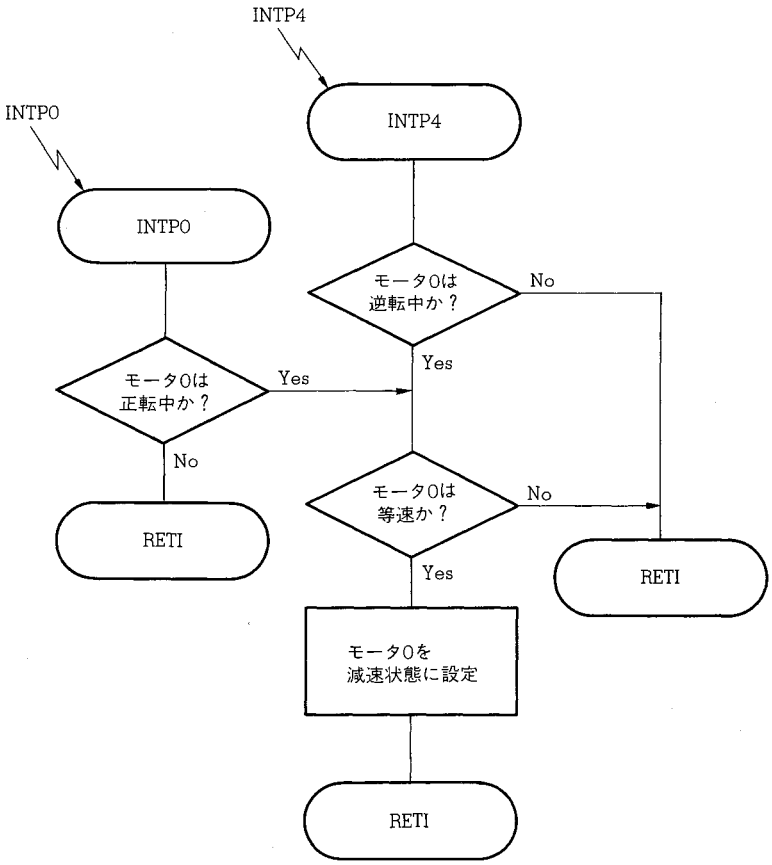


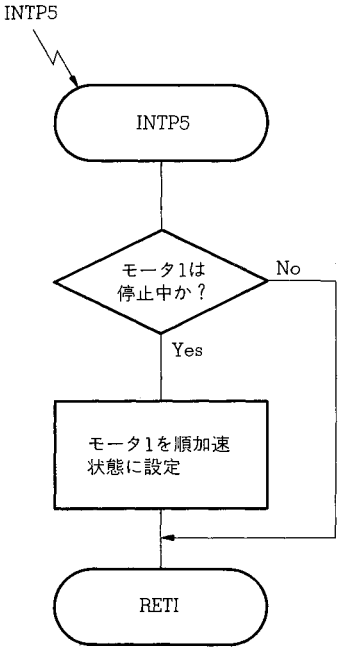
MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

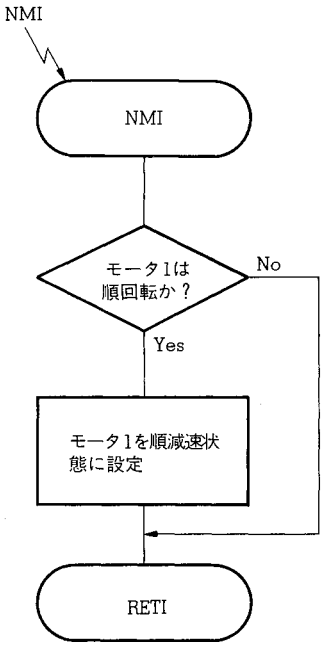
〔メ　モ〕

(4) 概要フロー・チャート

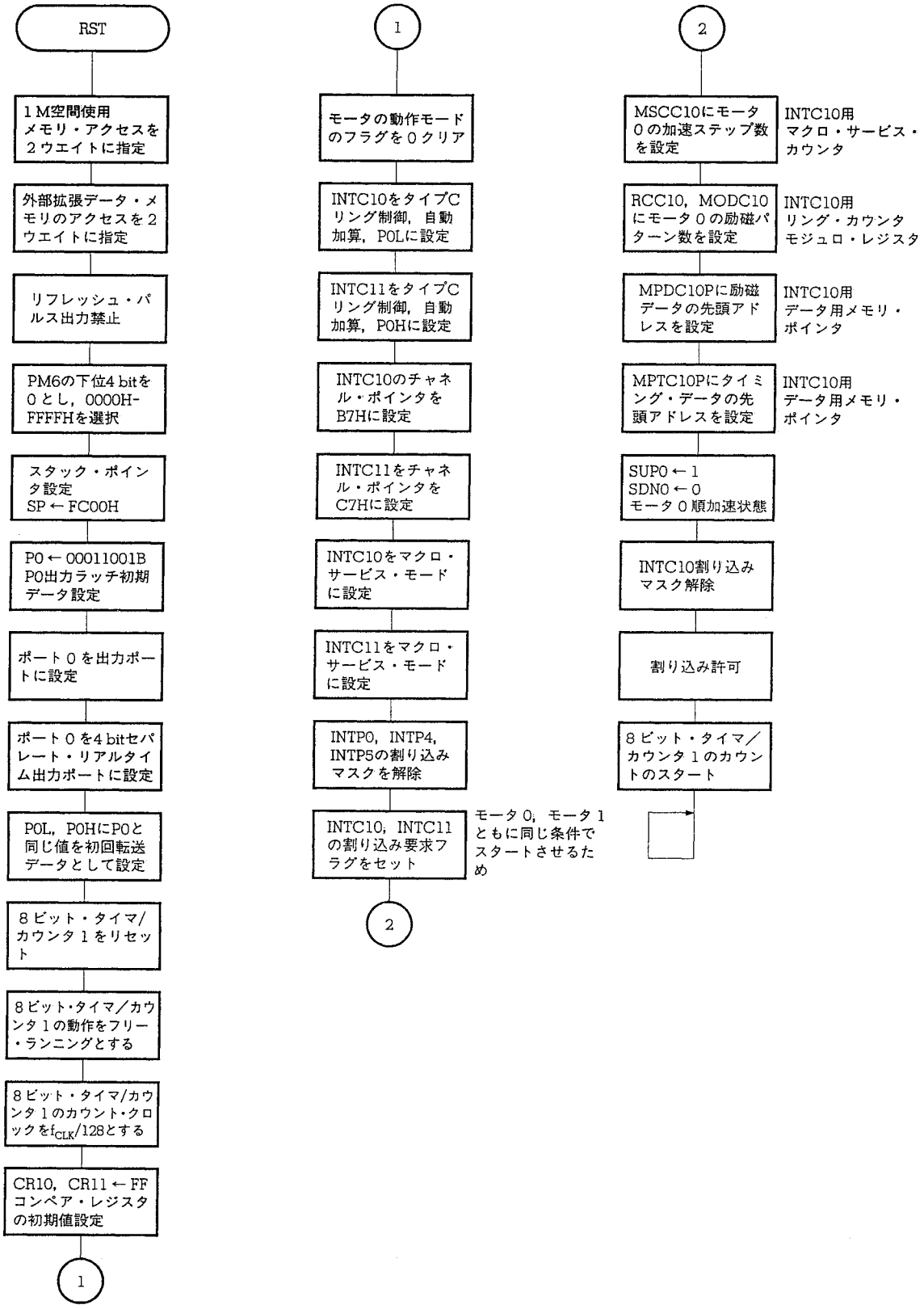

RST
メモリ使用環境のイニシャライズ
リアルタイム出力ポートのイニシャライズ
8ビット・タイマ/カウンタ1のイニシャライズ
INTC10, INTC11のマクロ・サービスのイニシャライズ
モータ0を順方向加速状態としてスタート
INTC10
INTC10
モータ0は順加速か？
No
Yes
モータ0は逆加速か？
Yes
No
モータ0を順回転(等速回転)状態に設定
モータ0を逆加速状態に設定
モータ0の停止
RETI
INTC11
INTC11
モータ1は順加速か？
No
Yes
モータ1を順回転(等速回転)状態に設定
モータ1の停止
RETI

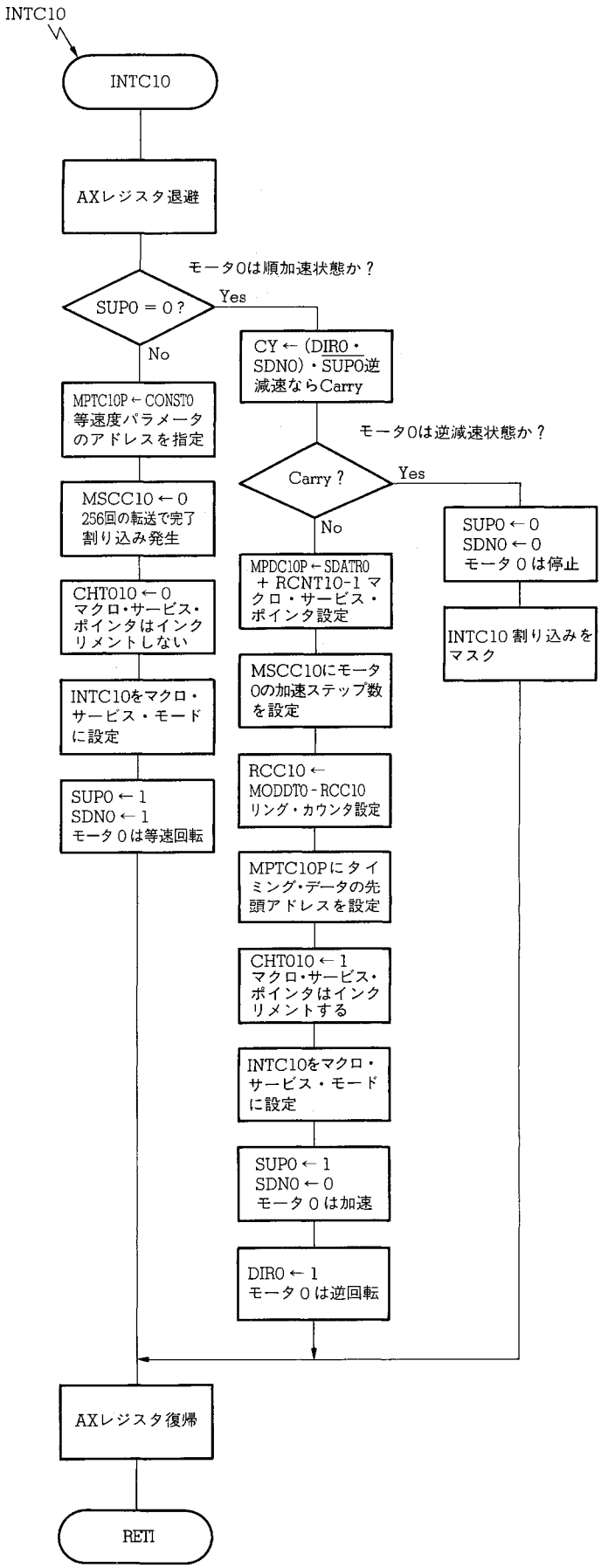

INTP4
INTP4
INTP0
INTP0
モータ0は
逆転中か？
No
Yes
モータ0は
正転中か？
Yes
No
モータ0は
等速か？
No
RETI
Yes
RETI
モータ0を
減速状態に設定
RETI

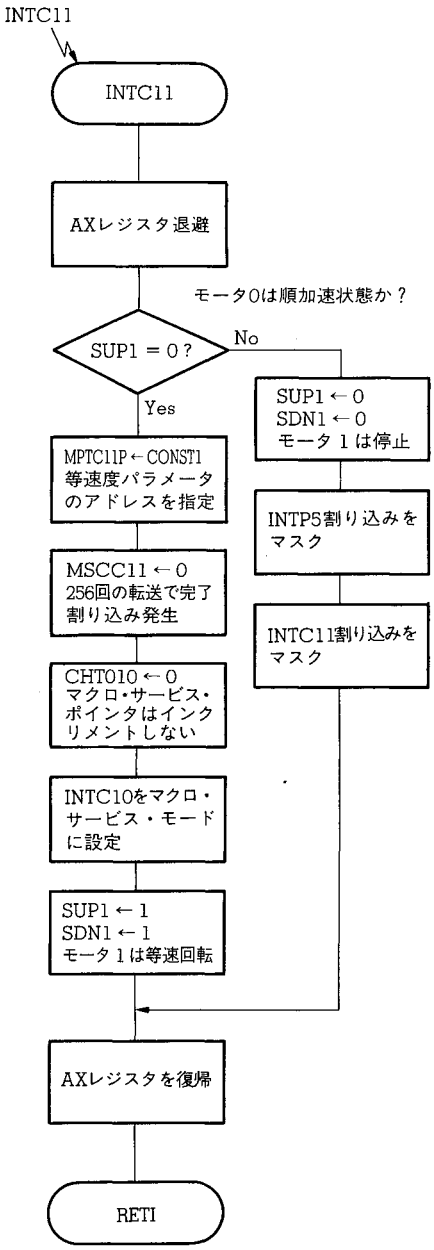

INTP5
INTP5
モータ1は
停止中か？
No
Yes
モータ1を順加速
状態に設定
RETI


NMI
NMI
モータ1は
順回転か？
No
Yes
モータ1を順減速状
態に設定
RETI

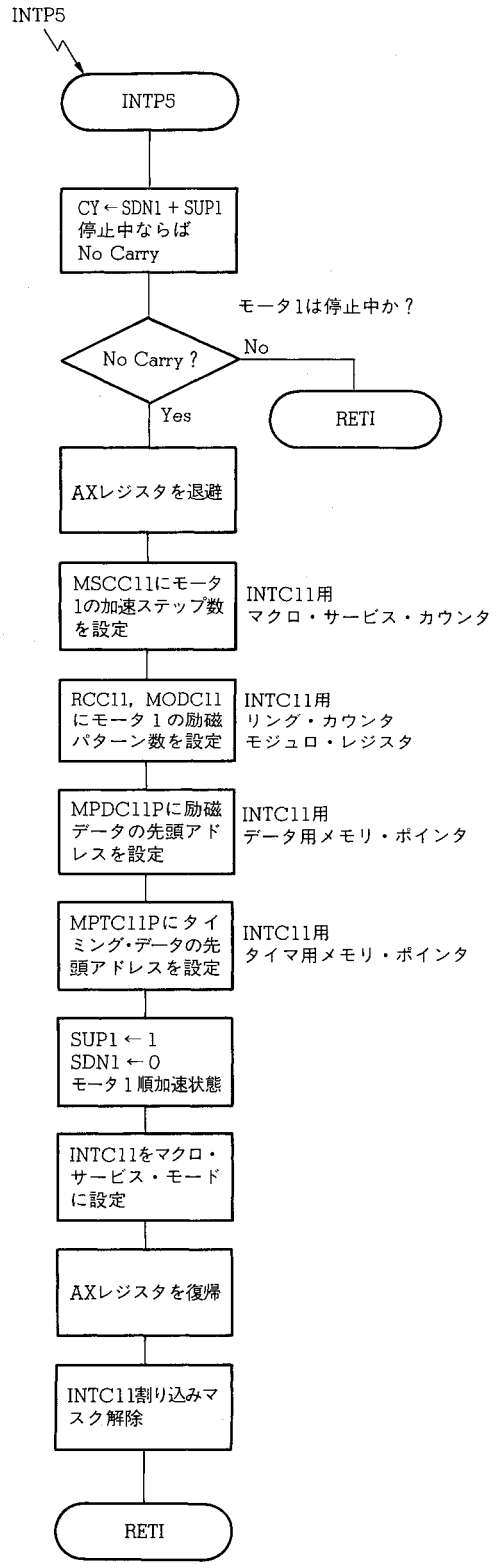
(5) 詳細フロー・チャート

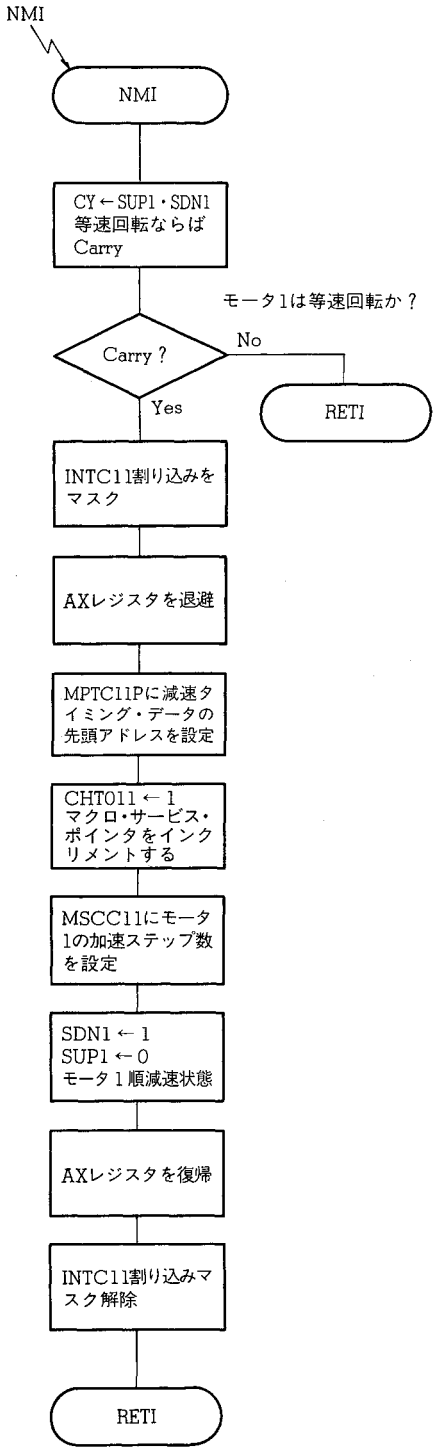

RST
1M空間使用メモリ・アクセスを2ウエイトに指定
外部拡張データ・メモリのアクセスを2ウエイトに指定
リフレッシュ・パルス出力禁止
PM6の下位4 bitを0とし，0000H-FFFFHを選択
スタック・ポインタ設定 SP←FC00H
P0←00011001B P0出力ラッチ初期データ設定
ポート0を出力ポートに設定
ポート0を4 bitセパレート・リアルタイム出力ポートに設定
P0L，P0HにP0と同じ値を初回転送データとして設定
8ビット・タイマ/カウンタ1をリセット
8ビット・タイマ/カウンタ1の動作をフリー・ランニングとする
8ビット・タイマ/カウンタ1のカウント・クロックをf_{CLK}/128とする
CR10，CR11←FFコンペア・レジスタの初期値設定
1
1
モータの動作モードのフラグを0クリア
INTC10をタイプCリング制御，自動加算，P0Lに設定
INTC11をタイプCリング制御，自動加算，P0Hに設定
INTC10のチャネル・ポインタをB7Hに設定
INTC11をチャネル・ポインタをC7Hに設定
INTC10をマクロ・サービス・モードに設定
INTC11をマクロ・サービス・モードに設定
INTP0，INTP4，INTP5の割り込みマスクを解除
INTC10，INTC11の割り込み要求フラグをセット
モータ0，モータ1ともに同じ条件でスタートさせるため
2
2
MSCC10にモータ0の加速ステップ数を設定
INTC10用マクロ・サービス・カウンタ
RCC10，MODC10にモータ0の励磁パターン数を設定
INTC10用リング・カウンタモジュロ・レジスタ
MPDC10Pに励磁データの先頭アドレスを設定
INTC10用データ用メモリ・ポインタ
MPTC10Pにタイミング・データの先頭アドレスを設定
INTC10用データ用メモリ・ポインタ
SUP0←1 SDN0←0 モータ0順加速状態
INTC10割り込みマスク解除
割り込み許可
8ビット・タイマ/カウンタ1のカウントのスタート

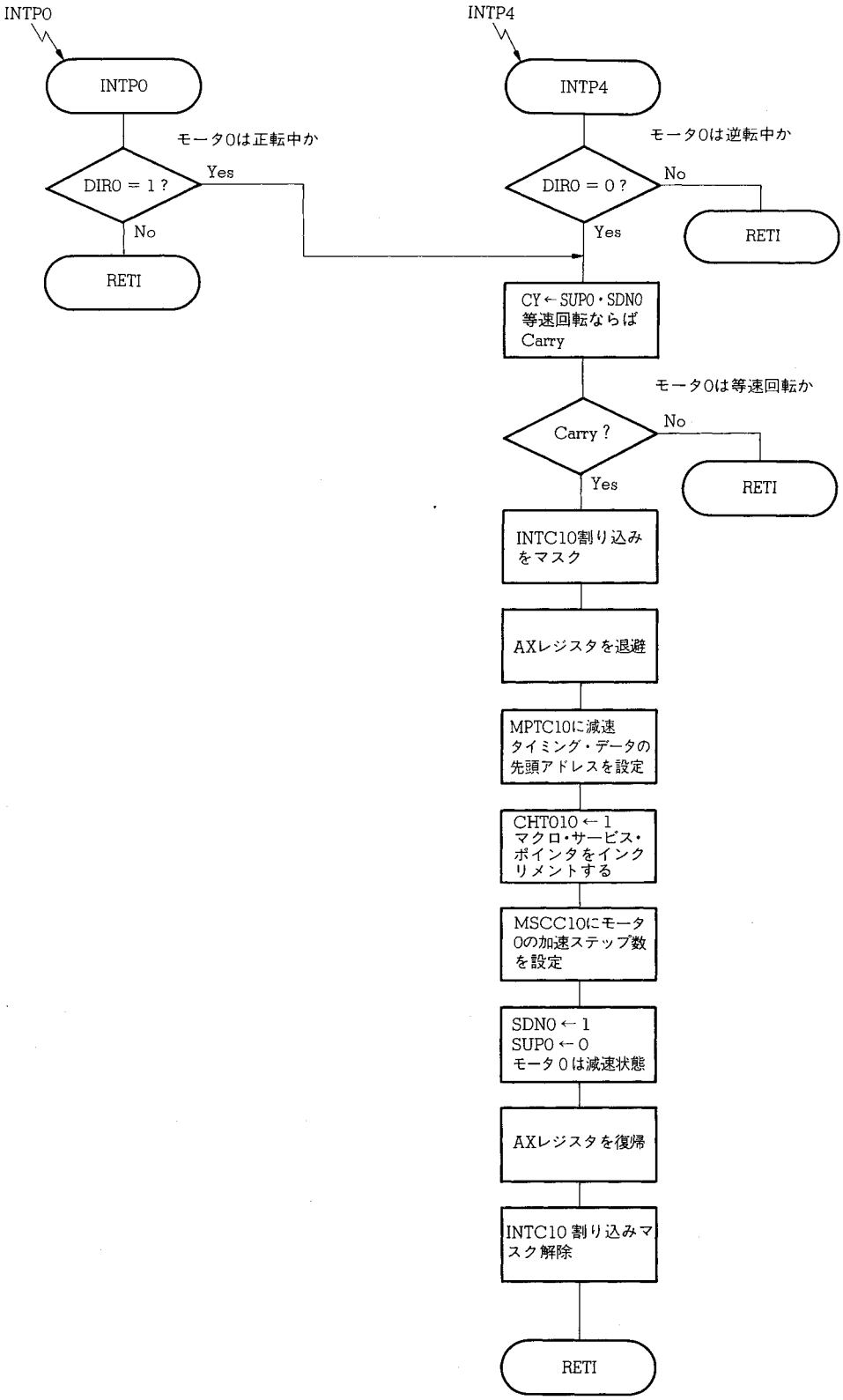




5
214
218A
224
234
244


INTP5
INTP5
CY←SDN1＋SUP1
停止中ならば
No Carry
モータ1は停止中か？
No Carry ?
No
Yes
RETI
AXレジスタを退避
MSCC11にモータ1の加速ステップ数を設定
INTC11用
マクロ・サービス・カウンタ
RCC11, MODC11にモータ1の励磁パターン数を設定
INTC11用
リング・カウンタ
モジュロ・レジスタ
MPDC11Pに励磁データの先頭アドレスを設定
INTC11用
データ用メモリ・ポインタ
MPTC11Pにタイミング・データの先頭アドレスを設定
INTC11用
タイマ用メモリ・ポインタ
SUP1←1
SDN1←0
モータ1順加速状態
INTC11をマクロ・サービス・モードに設定
AXレジスタを復帰
INTC11割り込みマスク解除
RETI


NMI
NMI
CY←SUP1・SDN1
等速回転ならば
Carry
モータ1は等速回転か？
Carry ?
No
Yes
RETI
INTC11割り込みをマスク
AXレジスタを退避
MPTC11Pに減速タイミング・データの先頭アドレスを設定
CHT011←1
マクロ・サービス・ポインタをインクリメントする
MSCC11にモータ1の加速ステップ数を設定
SDN1←1
SUP1←0
モータ1順減速状態
AXレジスタを復帰
INTC11割り込みマスク解除
RETI



5
214
218A
224
234
244

(6) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードします。

```
$       IC(INTMS.DEF)
        NAME    STEPR

;*************************************************************
;* real time output port application                         *
;*      control two stepping motors                          *
;*                                                           *
;*      INTC11 : P0H motor                                   *
;*      INTC10 : P0L motor                                   *
;*************************************************************
;
;       --- for system ---
;
M214    EQU     01000111B       ; uPD78214 MM data
PWDAT   EQU     10000000B       ; programable wait
RFDAT   EQU     10010000B       ; refresh mode
PM6DAT  EQU     0F0H            ; PM6 data
;
;       --- for work ---
;
STKSEG  DSEG    AT      0FE00H
        DS      20H
STACK:

STEP_B  DSEG    SADDR
STEPWK: DS      1               ; flags work area
;                                  7    6    5    4    3    2    1    0
SUP0    EQU     STEPWK.0        ; ---- DIR1 SDN1 SUP1 ---- DIR0 SDN0 SUP0
SDN0    EQU     STEPWK.1        ;       I
DIR0    EQU     STEPWK.2        ;       ------ 0:FORWARD  1:REVERSE
SUP1    EQU     STEPWK.4        ;
SDN1    EQU     STEPWK.5        ;       SDN SUP
DIR1    EQU     STEPWK.6        ;        0   0 --- STOP
;                                        0   1 --- SPEED UP
;                                        1   0 --- SPEED DOWN
;                                        1   1 --- CONSTANT
;
;       --- for macro service ---
;
        DSEG_CR C10,'SADDR'     ; INTC10 macro service channel (type C )
        CMCW10                  ;        macro service control word
        DSEG_CR C11,'SADDR'     ; INTC11 macro service channel (type C )
        CMCW11                  ;        macro service control word

CHT010  EQU     MSMC10.5        ; CHT10 bit for macro service counter
CHT011  EQU     MSMC11.5        ; CHT10 bit for macro service counter
```

```
;************************
;***   VECTOR TABLE   ***
;************************

        RSTVENT                 ; RESET  vector : motor 0 start
        NMIVENT                 ; NMI    vector : motor 1 stop
        PVENT0                  ; INTP0  vector : motor 0 reverse
        PVENT4                  ; INTP4  vector : motor 0 stop
        PVENT5                  ; INTP5  vector : motor 1 start
        CVENT10                 ; INTC10 vector
        CVENT11                 ; INTC11 vector

;*****************************
;***   SYSTEM INITIALIZE   ***
;*****************************

        CSEG
RST:
        MOV     MM,#M214        ; set memory mapping register
        MOV     PW,#PWDAT       ; set programable wait register
        MOV     RFM,#RFDAT      ; set refresh mode register
        MOV     PM6,#PM6DAT
        MOVW    SP,#STACK       ; set stack pointer

;*********************************
;***   HARD WARE INITIALIZE   ***
;*********************************

        MOV     P0,#00011001B   ; P0 output latch first data
        MOV     PM0,#0          ; P0 <- output port
        MOV     RTPC,#00010001B ; initialize real time output port
        MOV     P0L,#00001001B  ; real time output port first data
        MOV     P0H,#00010000B

        MOV     TMC1,#00000000B ; TM1 initialize
        MOV     CRC1,#00000000B ; clear disable TM1 by CR10 or CR11
        MOV     PRM1,#00000101B ; fclk/128 (TM1 count clock)
        MOV     CR10,#0FFH
        MOV     CR11,#0FFH

;************************************
;***   MACRO SERVICE INITIALIZE   ***
;************************************

        MOV     STEPWK,#0               ; mode clear
        MOV     MSMC10,#11101110B       ; INTC10 : type c , P0L , byte counter
        MOV     MSMC11,#11101111B       ; INTC11 : type c , P0H , byte counter
        MOV     CHPC10,#LOW(CHAC10)     ; set macro service channel pointer
        MOV     CHPC11,#LOW(CHAC11)
        SET1    CISM10                  ; set macro service mode
        SET1    CISM11
        MOV     MK0L,#11111110B         ; INTP0 mask open
        MOV     MK0H,#11111001B         ; INTP4,INTP5 mask open
        SET1    CIF10                   ; set interrupt flags
        SET1    CIF11
```

5
214
218A
224
234
244

```
;*************************
;***   START MOTOR 0   ***
;*************************

        MOV     A,!UDSTP0       ; set macro service counter
        MOV     MSCC10,A
        MOV     A,!MODDT0       ; set ring counter,modulo register
        MOV     RCC10,A
        MOV     MODC10,A
        MOVW    MPDC10P,#SDATF0 ; set memory pointer for data
        MOVW    MPTC10P,#SUPT0

        SET1    SUP0            ; speed up mode
        CLR1    SDN0

        CLR1    CMK10           ; INTC10 mask open
        EI                      ; enable interrupt
        MOV     TMC1,#00001000B ; timer start

;***********************
;***   DUMMY LOOP    ***
;***********************

        BR      $$

;*************************************************
;***   COMPLETE OF MACRO SERVICE by [INTC10]   ***
;*************************************************

INTC10:
        PUSH    AX              ; save register
        BF      SUP0,$IC0_J1
;
;       ---  motor 0 constant mode ---
;
;       DIR0 SDN0 SUP0
;        1/0  1/0   1
;
        MOVW    AX,#CONST0      ; set memory pointer for timing
        MOVW    MPTC10P,AX
        MOV     MSCC10,#0       ; set macro service counter

        CLR1    CHT010          ; constant mode
        SET1    CISM10          ; set macro service mode
        SET1    SUP0            ; constant mode
        SET1    SDN0
        BR      IC0_J3
;
IC0_J1:
        MOV1    CY,DIR0
        AND1    CY,SDN0
        AND1    CY,/SUP0
        BNC     $IC0_J2
;
```

```
;       --- motor 0 stop ---
;
;       DIR0 SDN0 SUP0
;         1    1    0
;
        CLR1    SUP0            ; stop mode
        CLR1    SDN0
        SET1    CMK10           ; INTC10 mask
        BR      IC0_J3
;
;       ---  motor 0 reverse ---
;
;       DIR0 SDN0 SUP0
;         0    1    0
;
IC0_J2:                         ; set reverse pointer
        MOV     A,RCC10         ; set memory poiter for data
        MOV     X,#0
        XCH     A,X
        ADDW    AX,#SDATR0
        DECW    AX
        MOVW    MPDC10P,AX

        MOV     A,!UDSTP0       ; set macro service counter
        MOV     MSCC10,A

        MOV     A,!MODDT0       ; set ring counter
        INC     A
        SUB     A,RCC10
        MOV     RCC10,A

        MOVW    MPTC10P,#SUPT0  ; set memory pointer for timing

        SET1    CHT010          ; no constant mode
        SET1    CISM10          ; set macro service mode
        SET1    SUP0            ; speed up mode
        CLR1    SDN0
        SET1    DIR0            ; reverse mode
IC0_J3:
        POP     AX              ; restore register
        RETI

;************************************************
;***   COMPLETE OF MACRO SERVICE by [INTC11]   ***
;************************************************

INTC11:
        PUSH    AX              ; save register
        BF      SUP1,$IC1_J1
;
```

5
214
218A
224
234
244

```
;       ---  motor 1 constant mode ---
;
;       DIR1 SDN1 SUP1
;         0   1/0   1
;
        MOVW    AX,#CONST1      ; set memory pointer for timing
        MOVW    MPTC11P,AX
        MOV     MSCC11,#0       ; set macro service counter

        CLR1    CHT011          ; constant mode
        SET1    CISM11          ; set macro service mode
        SET1    SUP1            ; constant mode
        SET1    SDN1
        BR      IC1_J2
;
;       --- motor 1 stop ---
;
;       DIR1 SDN1 SUP1
;         0    1    0
;
IC1_J1:
        CLR1    SUP1            ; stop mode
        CLR1    SDN1
        SET1    PMK5            ; INTP5 mask
        SET1    CMK11           ; INTC11 mask
IC1_J2:
        POP     AX              ; restore register
        RETI

;*********************************
;***   EXT. INTERRUPT [NMI]   ***
;*********************************

NMI:
        MOV1    CY,SUP1         ; check constant mode
        AND1    CY,SDN1
        BC      $NMI_J1
        RETI                    ; no constant mode
;
;       --- motor 1 speed down ---
;
NMI_J1:
        SET1    CMK11           ; INTC11 mask
        PUSH    AX              ; save register

        MOVW    MPTC11P,#SDNT1  ; set memory pointer for timing
        SET1    CHT011          ; no constant mode
        MOV     A,!UDSTP1       ; set macro service counter
        MOV     MSCC11,A

        SET1    SDN1            ; motor 1 speed down mode
        CLR1    SUP1
        POP     AX              ; restore register
        CLR1    CMK11           ; INTC11 mask open
        RETI
```

```
;****************************************
;***   EXT. INTERRUPT [INTP0, INTP4]   ***
;****************************************

INTP4:
        BT      DIR0,$IP4_J1
        RETI
INTP0:
        BF      DIR0,$IP4_J1
        RETI
IP4_J1:
        MOV1    CY,SUP0         ; check constant mode
        AND1    CY,SDN0
        BC      $IP4_J2
        RETI                    ; no constant mode
;
;       --- motor 0 speed down ---
;
IP4_J2:
        SET1    CMK10           ; INTC10 mask
        PUSH    AX              ; save register

        MOVW    MPTC10P,#SDNT0  ; set memory pointer for timing
        SET1    CHT010          ; no constant mode
        MOV     A,!UDSTP0       ; set macro service counter
        MOV     MSCC10,A

        SET1    SDN0            ; motor 0 speed down mode
        CLR1    SUP0
        POP     AX              ; restore register
        CLR1    CMK10           ; INTC10 mask open
        RETI
;************************************************
;***   EXT. INTERRUPT [INTP5] MOTOR 1 START   ***
;************************************************

INTP5:
        MOV1    CY,SDN1         ; check stop mode
        OR1     CY,SUP1
        BNC     $IP5_J1
        RETI                    ; no stop mode
IP5_J1:
        PUSH    AX              ; save register

        MOV     A,!UDSTP1       ; set macro service counter
        MOV     MSCC11,A
        MOV     A,!MODDT1       ; set ring counter,modulo register
        MOV     RCC11,A
        MOV     MODC11,A
        MOVW    MPDC11P,#SDATF1 ; set memory pointer for data
        MOVW    MPTC11P,#SUPT1
```

5
214
218A
224
234
244

```
        SET1    SUP1            ; motor 1 speed up mode
        CLR1    SDN1
        SET1    CISM11          ; set macro service mode
        POP     AX              ; restore register
        CLR1    CMK11           ; INTC11 mask open
        RETI

;************************
;***   DATA PROFILE   ***
;************************
;
;       *** motor 0 ***
;
PARAM0  CSEG    PAGE

MODDT0: DB      8                       ; number of step pattern
UDSTP0: DB      11                      ; number of speed up/down step
;
;       *** data profile ***
;
SDATF0: DB      00001000B               ; forward
        DB      00001100B
        DB      00000100B
        DB      00000110B
        DB      00000010B
        DB      00000011B
        DB      00000001B
        DB      00001001B
SDATR0:
        DB      00000011B               ; reverse
        DB      00000010B
        DB      00000110B
        DB      00000100B
        DB      00001100B
        DB      00001000B
        DB      00001001B
        DB      00000001B
;
;       *** timing data profile ***
;
SUPT0:  DB      0F7H                    ; speed up timing for dummy cycle
        DB      0F7H, 66H, 4FH, 42H, 3AH    ; speed up timing
        DB      35H, 30H, 2DH, 2AH, 28H
CONST0: DB      27H                     ; constant timing
SDNT0:  DB      28H, 2AH, 2DH, 30H, 35H     ; speed down timing
        DB      3AH, 42H, 4FH, 66H, 0F7H
        DB      0F7H                    ; speed down timing for dummy cycle
;
```

```
;       *** motor 1 ***
;
PARAM1  CSEG    PAGE

MODDT1: DB      4                       ; number of step pattern
UDSTP1: DB      7                       ; number of speed up/down step
;
;       *** data profile ***
;
SDATF1:
        DB      10000000B               ; forward
        DB      01000000B
        DB      00100000B
        DB      00010000B
SDATR1:
        DB      01000000B               ; reverse
        DB      10000000B
        DB      00010000B
        DB      00100000B
;
;       *** timing data profile ***
;
SUPT1:  DB      0EFH                    ; speed up timing for dummy cycle
        DB      0EFH, 63H, 4CH, 40H     ; speed up timing
        DB      38H, 33H
CONST1: DB      31H                     ; constant timing
SDNT1:  DB      33H, 38H, 40H, 4CH      ; speed down timing
        DB      63H, 0EFH
        DB      0EFH                    ; speed down timing for dummy cycle
;
        END
```


214
218A
224
234
244

第6章　リアルタイム出力ポートによるインバータの制御

6.1 概　　要

リアルタイム出力ポートのアプリケーション例として，PWM出力の機能を三相インバータ制御に応用した例を紹介します。

インバータの用途として，特に誘導電動機の制御があげられます。一般に，誘導電動機は同一出力の直流機と比較すると安価ですが，速度制御を行うには周波数，極数もしくは，すべりを変えなければならないため，直流機の制御の場合に比べて複雑になります。この誘導機に対してインバータを用いることにより，周波数可変による速度制御が可能となります。

この章の内容は，1KバイトのRAMと，32KバイトのROMを内蔵する，μPD78238，μPD78218Aをターゲット CPUとしています。

なお，本文中ではμPD78218Aを代表品種として説明しています。

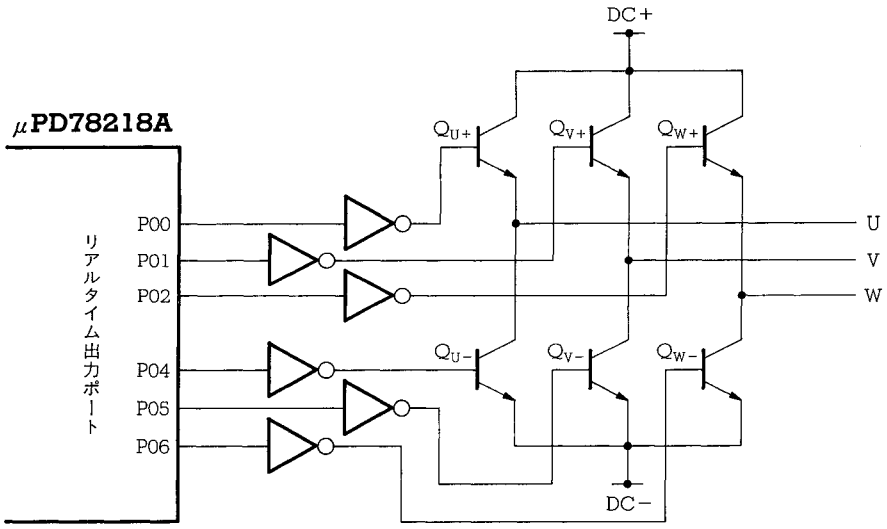
6.2　機能概要

6.2.1　実現手段

この応用例のインバータは三相インバータです。制御方法は正弦波PWM方式で，このPWM信号をリアルタイム出力ポート＋マクロ・サービス・タイプCで出力しています。

図6－1にインバータ・ブロックの概略図を示します。

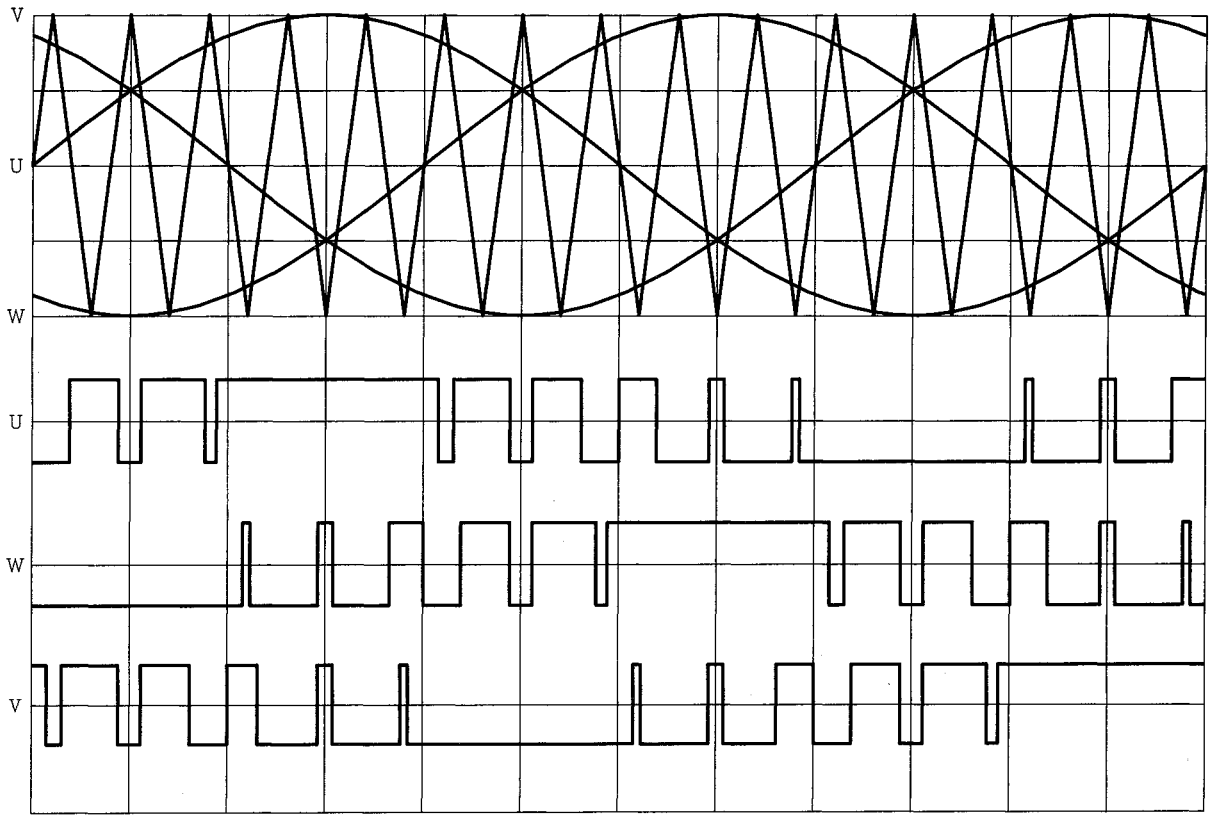
図6－1　インバータ・ブロックの概略図



6.2.2　PWM信号の生成方法

PWM信号の生成は，図6－2に示すように，正弦波を三角波（キャリア）で変調して行います。このときのキャリア周波数は，3.125 kHzを使用しています。この値は，8ビット・タイマ／カウンタ1の最小カウント・クロックの1/120となります。

図6－2　三角波による正弦波疑似PWM波形の生成



6.2.3　インバータの出力周波数

インバータ出力周波数は，「PWM信号の生成方法」で触れているキャリア周波数に依存します。この応用例では，インバータ出力周波数の周期が，キャリア周期の３の整数倍になるように設定しています。理由を次に示します。

（a）キャリアの倍数である

正弦波を１周期単位で扱わないと，データが膨大になること。

（b）キャリアの３の整数倍である

正弦波データは，各相で120°ずつの位相差を持っており，三相全体では，相が入れ替わるのみで，120°単位で波形が合同であり，1/3のデータ量で済むため。

この応用例では，約33 Hzから約208 Hzに７点の固定周波数点を設けています。表６－１に，この応用例での固定出力周波数を示します。

表６－１　インバータ出力周波数

出力周波数番号	1	2	3	4	5	6	7
出力周波数［Hz］	32.55	43.40	65.10	86.81	130.21	173.61	208.33
出力周期［ms］	30.72	23.04	15.36	11.52	7.68	5.76	4.80
キャリア（3.125 kHz）／出力周波数	96	72	48	36	24	18	15

なお，通常，低回転時の省電力効果を得るためには，V/F＝一定とし電圧の比率を下げますが，この応用例では，周波数のみ可変としています。

6.2.4　マクロ・サービス・タイプCの利用方法

この応用例のインバータ制御では，リアルタイム出力ポートをマクロ・サービス・タイプCの自動加算モードで制御しています。マクロ・サービス・タイプCで扱うデータは，タイミング・データと，出力データで，それぞれ，インバータのトランジスタをON/OFFする期間と，ON/OFFするトランジスタを決定します。

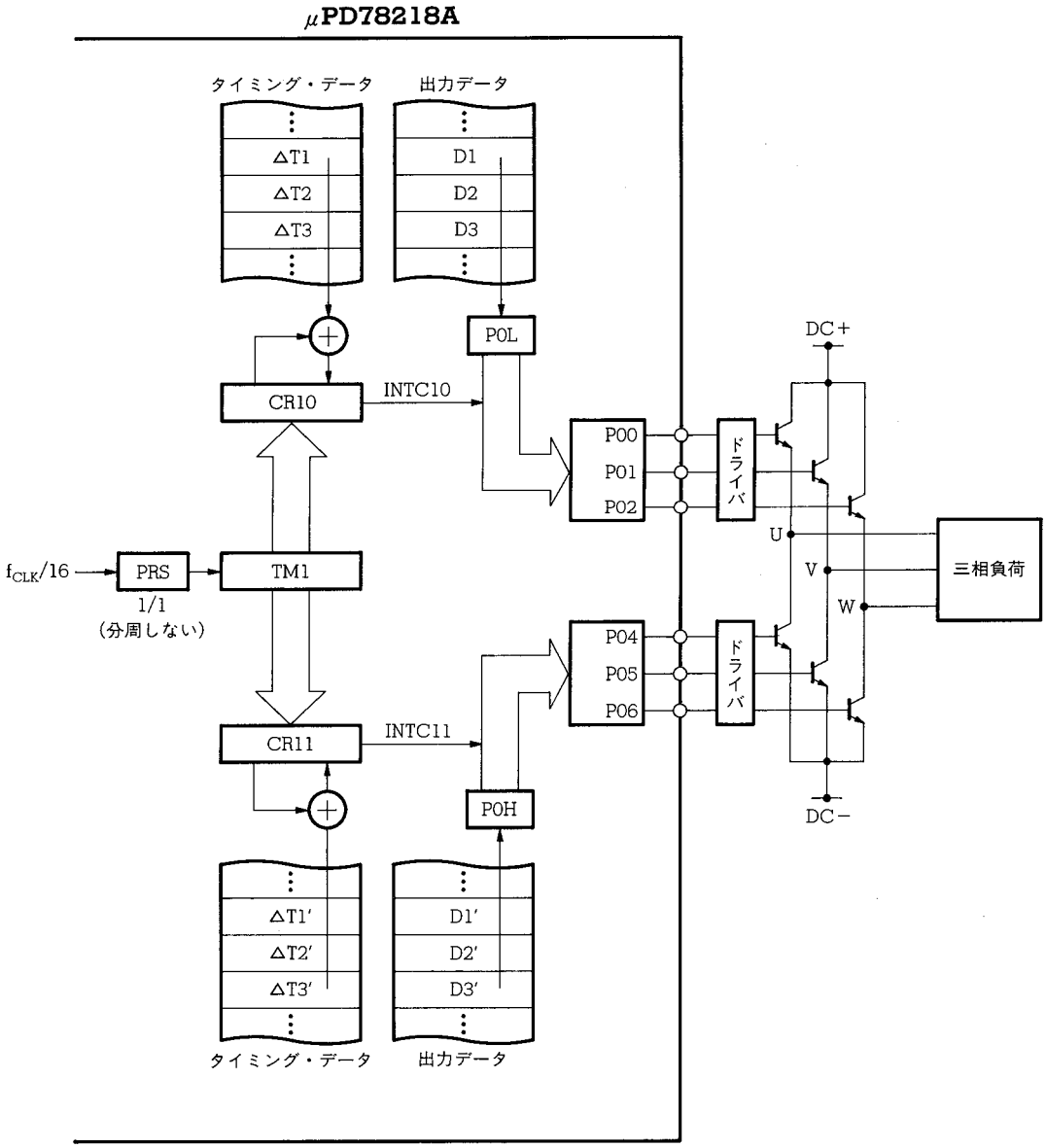
マクロ・サービス・タイプCの割り込みソースである８ビット・タイマ／カウンタ１の割り込み要求信号のINTC10, INTC11の両方を用い，前者をDC電源に対して＋側のトランジスタのON/OFF制御，後者をDC電源に対して－側のトランジスタのON/OFF制御に使用しています。したがって，２系統のマクロ・サービスでインバータ制御を行います。

表６－２　マクロ・サービス・タイプCの機能分担

トランジスタ	割り込み要求	リアルタイム出力ポート	出力ラッチ
Q_{U+}, Q_{V+}, Q_{W+}	INTC10	P0L	P00, P01, P02
Q_{U-}, Q_{V-}, Q_{W-}	INTC11	P0H	P04, P05, P06

図６－３にインバータ制御を行うためのブロック図を示します。

図６－３　インバータ制御ブロック図



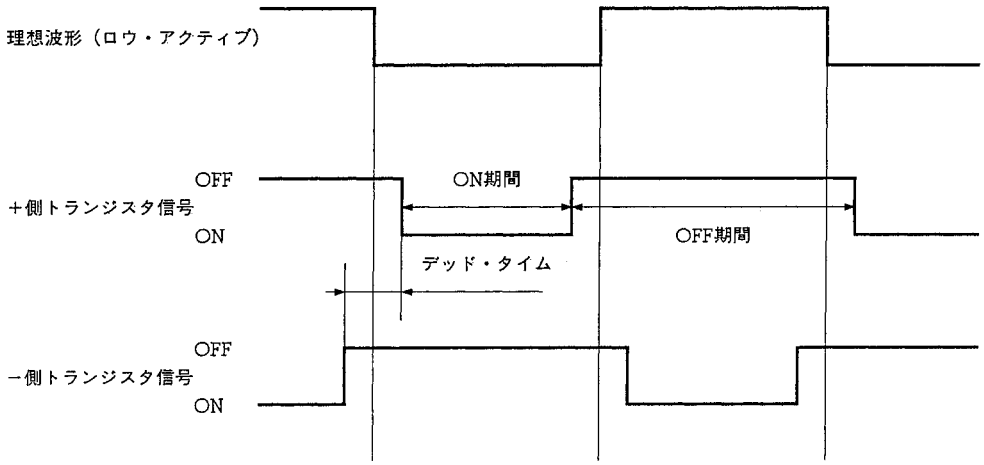
(1) 出力タイミング・データ領域

出力タイミング・データ領域はインバータをON/OFFするタイミング，すなわちPWM波形のパルス幅を決定します。

三相交流波形は，1/3周期の波形の繰り返しで表すことができるため，時間データである，出力タイミング・データは1/3周期分だけのデータで持っています。

ここで，－側のトランジスタの信号は，＋側のトランジスタの信号の反転となりますが，トランジスタはOFF時に蓄積時間があるため，ショート期間が発生します。このため，両方のトランジスタがOFFしている期間をスイッチング付近に設けています。これをデッド・タイムといいます。

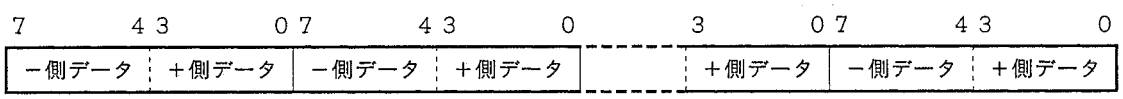
図6－4　デッド・タイムの考慮



(2) 出力データ領域

出力データは，それぞれ4ビットで構成し，1ビットは1つのトランジスタのON/OFFを決定します。出力データは1/3周期での再現性はないため，1周期分のデータを持っています。ここで，マクロ・サービスで転送する＋側のデータはPOLに，－側のデータはPOHにそれぞれ4ビットのデータを書き込むので，図6－5のように同じ領域の下位4ビットと上位4ビットにそれぞれのデータを格納しておきます。

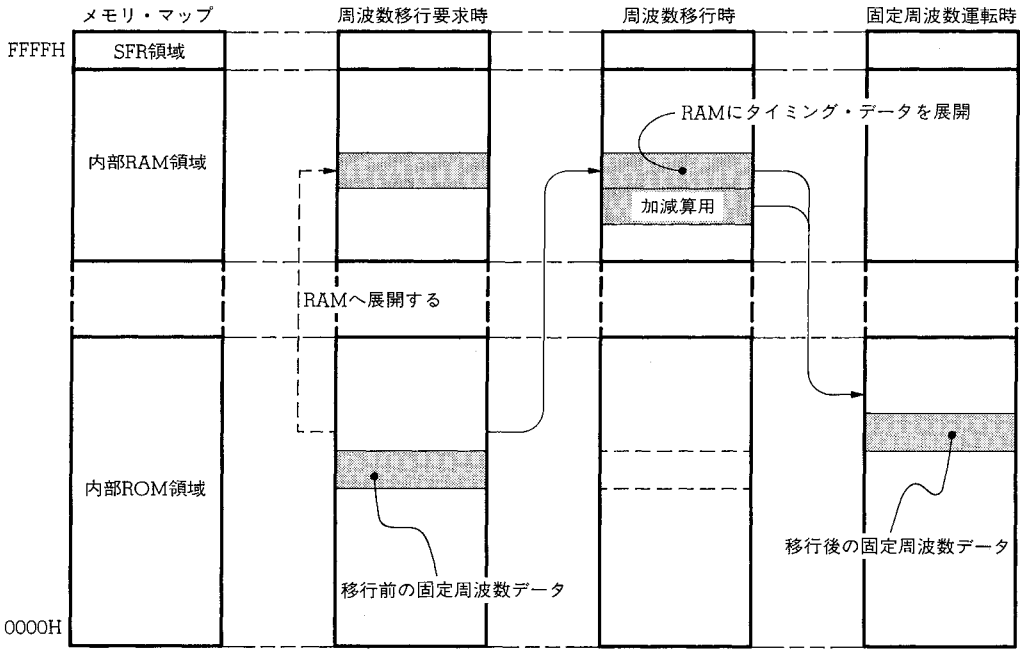
図6－5　出力データ領域の使用方法



6.2.5　周波数の移行方法

インバータ出力周波数をf→f'へ移行する場合，急激な出力周波数の変化にモータが追従できない可能性があるため，２秒ごとに，数Hzずつ周波数を変化させています。これを実現するために，２秒ごとに，タイミング・データの内容をインクリメント／デクリメントさせる手法を用いています。この際，固定周波数による運転時にROM上にあったタイミング・データを周波数移行状態ではRAM上に展開し，目標周波数に達した時点でROM内の固定周波数に切り替えています。バッファRAMは２組あり，加減算用と出力用のタイミング・データを持たせ，２秒ごとに加減算を行い，加減算完了後にマクロ・サービスで使用するバッファRAMを切り替えています。

図6－6　出力タイミング・データの切り替え



周波数移行時の目標周波数の変更要求があった場合は，目標周波数の変更を受け付け，現在の移行によって最寄りの固定周波数に達した時点で，再度目標周波数を見直し，移行状態を設定しています。したがって，連続移行状態である場合も，目標周波数の変更要求を受け付けます。

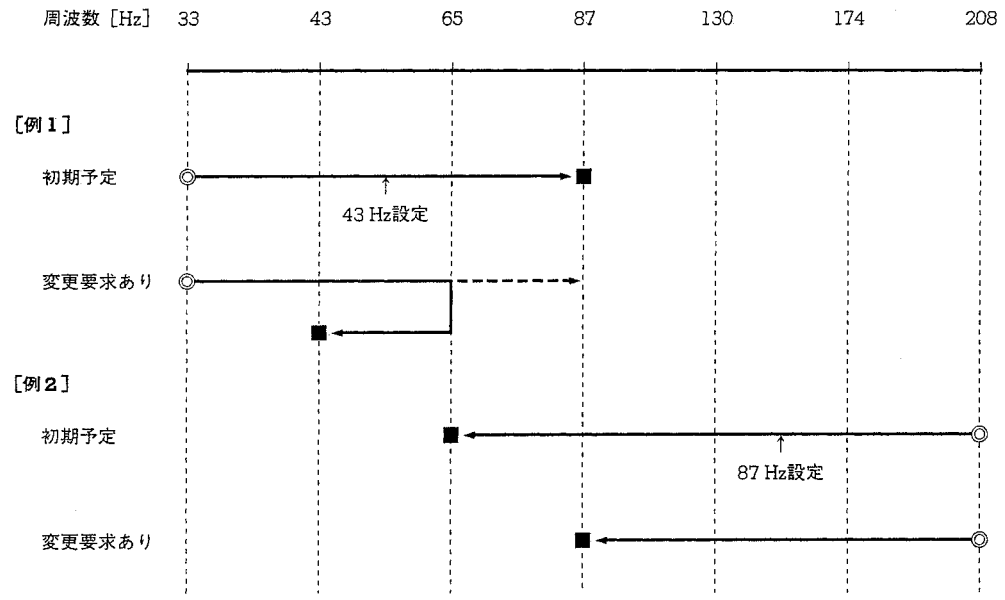
(1) 例 1

33 Hz→87 Hz移行時で，現在43 Hzから65 Hzに移行中に，43 Hzが目標周波数に指定された場合は，65 Hzに到達後43 Hzへ移行する。

(2) 例 2

208 Hz→65 Hz移行時で，現在174 Hzから130 Hzに移行中に，87 Hzが目標周波数に指定された場合は，87 Hz到達時点で移行を終了する。

図 6－7　変更要求の受け付け



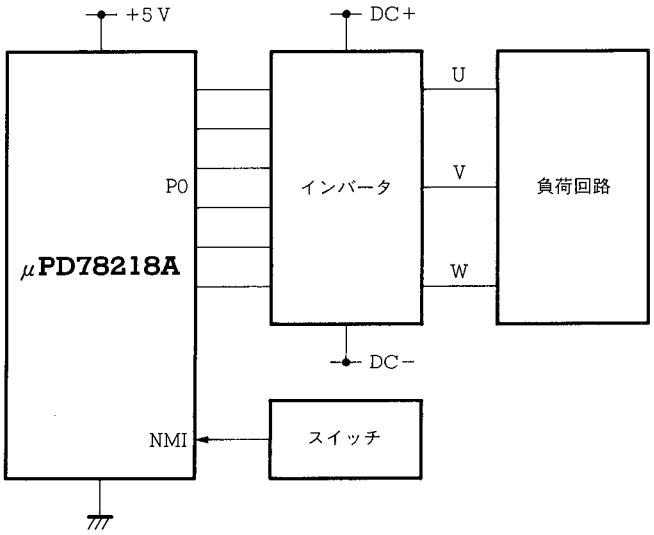
◎：初期周波数　　■：移行後最終周波数

6.3　ハードウエア構成

6.3.1　システム構成

この応用例の構成図を図6-8に示します。

図6-8　78K/IIインバータ制御システム構成図



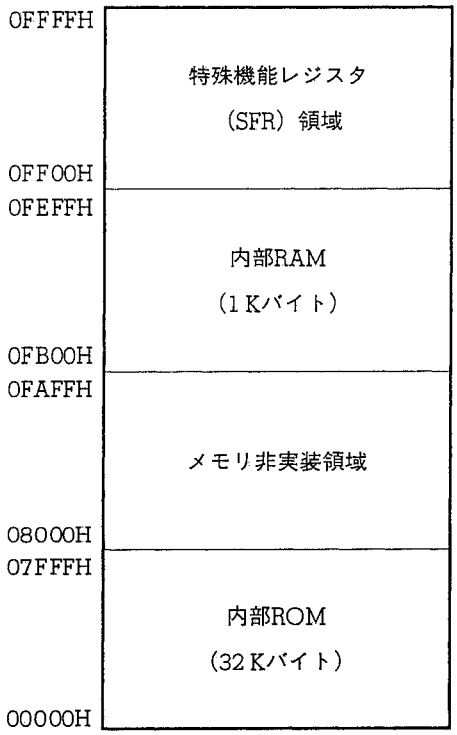
6.3.2　CPUブロック

ターゲットCPUはμPD78218Aとします。インバータ制御用として，ポート０をリアルタイム出力ポートとして機能させ，インバータのパワー・トランジスタのON/OFF制御を行います。メモリは，外部拡張を行わず，内部ROM，RAMのみを使用しています。

(1) メモリ・マップ

この応用例のメモリ・マップを図６－９に示します。

図６－９　メモリ・マップ



(2) ポート機能

この応用例のインバータに関するポート機能を表6-3に示します。

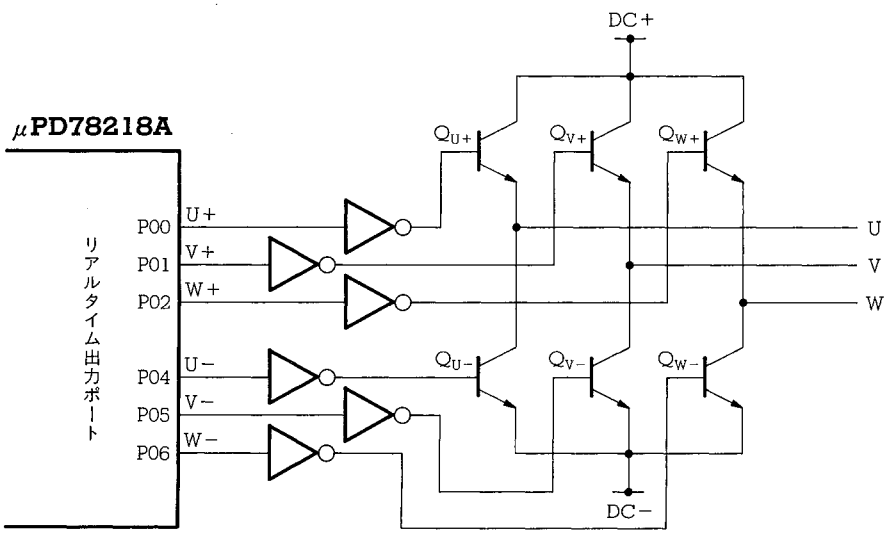
表6-3　ポート機能

ポート		機能名称	入出力	端　子　機　能	アクティブ・レベル
0	0	—	出　力	インバータ・トランジスタ スイッチング信号（+側）	Low
	1				
	2				
	4			インバータ・トランジスタ スイッチング信号（−側）	
	5				
	6				
2	0	NMI	—	緊急停止信号入力	立ち下がり

6.3.3　インバータ・ブロック

この応用例のインバータは三相トランジスタ・インバータで，図6-10に示すようなロウ・アクティブのインバータ・ブロックを想定しています。インバータ回路構成を図6-10に示します。

図6-10　インバータ回路構成図



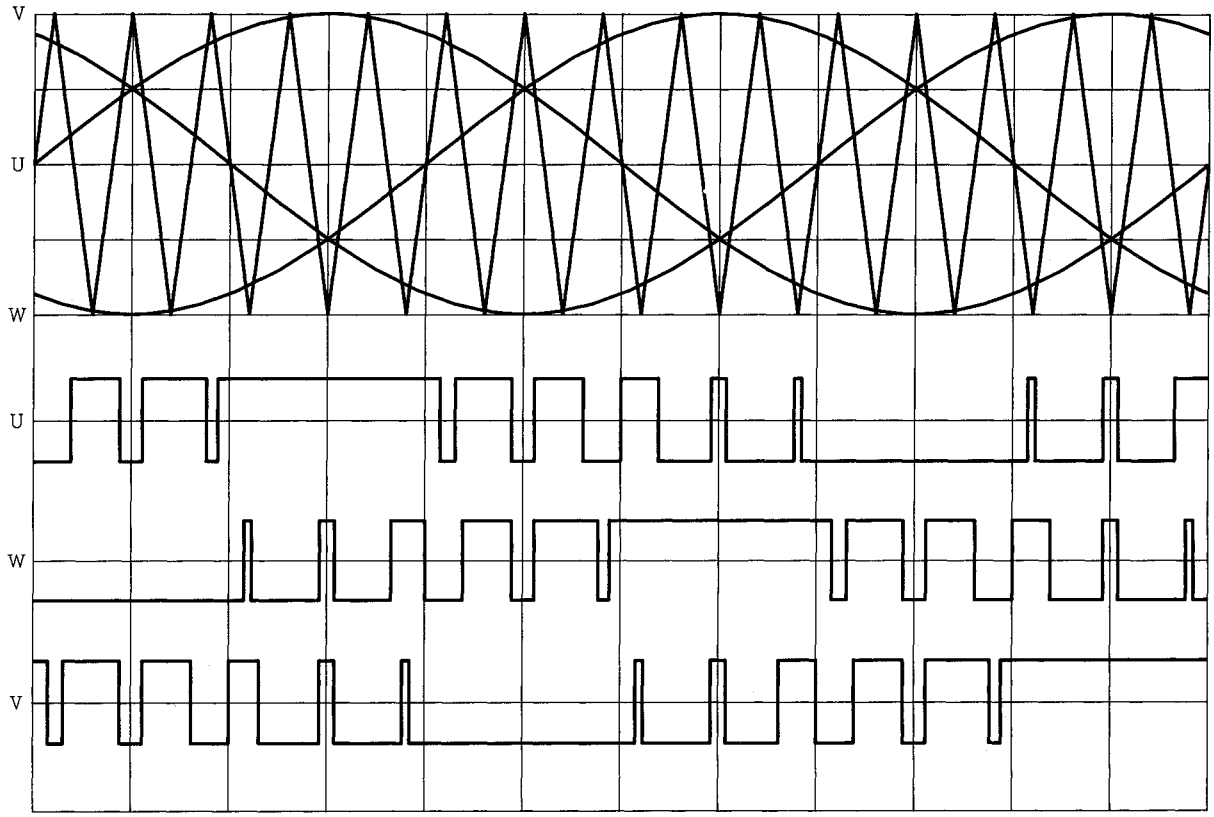
6.4　制御方法（データ編）

6.4.1　概　　要

PWM信号は，正弦波を三角波（キャリア）で変調して生成しています。ここで，U相＋側，V相＋側，W相＋側，U相－側，V相－側，W相－側の６種類のデータを合成し，1/3周期のタイミング・データ（パルス幅）と，１周期の出力データ（インバータ・トランジスタON/OFFデータ）を用意しています。

図６－11に208 Hz時の三相分の変調波形を示します。

図６－11　208 Hz時の変調波形



6.4.2　PWMデータの構成

キャリアで変調された単相分の変調データをタイマのカウント・クロック単位で算出します。

ここで，トランジスタのスイッチング時間と，マクロ・サービスの処理速度の関係から，短いパルス幅は無視し，また，他相とのスイッチング時間の差が小さい場合には，他方のスイッチング・タイミングに合わせ込みを行っています。

(1) PWMデータの仕様

この応用例で扱うPWMデータの仕様を表6-4に示します。

表6-4　PWMデータの仕様

最小パルス幅（単相あたり）	26.7 μs
最小スイッチング・タイミング（＋側全相および－側全相）	26.7 μs
デッド・タイム	13.3 μs
各相の位相差（時間軸－方向）	120°注

注　$V_U=\sin\theta$，$V_V=\sin\left(\theta+\frac{4}{3}\pi\right)$，$V_W=\sin\left(\theta+\frac{2}{3}\pi\right)$

なお，デッド・タイムについては，－側の波形を，＋側の波形に対して，デッド・タイム分遅らせて出力する形式をとっています。

(2) データ構成

同一相において，＋側トランジスタ用と，－側トランジスタ用の２つのデータ領域を用意しています。ここで，タイミング・データは1/3周期のデータであり，出力データは１周期分あるため，出力データは，0－2π/3，2π/3－4π/3，4π/3－2πの範囲に分けて用意しています。データ項目を以下に示します。

(a) ＋側トランジスタ用タイミング・データ
(b) －側トランジスタ用タイミング・データ
(c) 出力データ［＋側］　0－2π/3
(d) 出力データ［＋側］2π/3－4π/3
(e) 出力データ［＋側］4π/3－2π
(f) 出力データ［－側］　0－2π/3
(g) 出力データ［－側］2π/3－4π/3
(h) 出力データ［－側］4π/3－2π
(i) 1/3周期のデータ数
(j) 加減回数

上記のデータ項目のうち，(c)-(e)と(f)-(h)は，それぞれ下位４ビット，上位４ビットのみがPOL，POHに転送される有効データなので，領域を共通化しています。さらに，マクロ・サービス完了割り込みでのポインタ更新処理の簡略化をはかるため，ギャップを設けず連続的に配置しています。

また，**(i) 1/3周期のデータ数**は，マクロ・サービスの転送回数，すなわちマクロ・サービス・カウンタの設定値に利用しています。また，**(j) 加減回数**は，周波数を移行する際，何回の移行によって，隣接する周波数に達するかを決定するものです。周波数の移行については，**6.5　制御方法（プログラム編）**を参照してください。

(3) PWMデータ・ソース・ファイル形式

(a) PWMデータ

PWMデータは，各周波数ごとに，タイミング・データと出力データにセグメント名とラベルを付けており，すべての周波数のデータを1つのファイルとして持っています。

図6-12にPWMデータの一部分を示します。

図6-12　PWMデータ例

```
;         FREQUENCY   =208 [Hz]
;         DATA LENGTH=10

F208TM    CSEG
TMP208 :  DB          040H, 044H, 027H, 047H, 040H, 02BH, 058H, 013H, 05BH, 02BH
TMM208 :  DB          02BH, 05BH, 013H, 058H, 02BH, 040H, 047H, 027H, 044H, 040H

F208DT    CSEG
DT208 :   DB          0ACH, 0BFH, 08CH, 0BEH, 09CH, 0BEH, 098H, 0FEH, 09AH, 0DEH
          DB          09AH, 0DFH, 08AH, 0DBH, 0CAH, 0DBH, 0C8H, 0FBH, 0C9H, 0EBH
          DB          0C9H, 0EFH, 089H, 0EDH, 0A9H, 0EDH, 0A8H, 0FDH, 0ACH, 0BDH


;         FREQUENCY   =174 [Hz]
;         DATA LENGTH=12

F174TM    CSEG
                .
                .
                .
```

(b) データ配置テーブル

各周波数のPWMデータ配置アドレスを高い周波数から降順に，図6－13のように配置しています。制御プログラムでは，このテーブルから，マクロ・サービス・チャネルの設定を行います。これらの情報のまとまりをコントロール・エリアと呼んでいます。このファイルは，**(a) PWMデータ**を後方に結合し，PWMDATA.ASMのファイルとしています。

図6－13　データ配置テーブル例

```
        PUBLIC  CNT208
CNT208 :                ; 208 Hz
        DW      TMP208          ; timing data +
        DW      TMM208          ; timing data -
        DW      DT208           ; output data
        DB      18              ; data length (1/3 cycle)
        DB      6               ; add/sub counter

        PUBLIC  CNT174
CNT174 :                ; 174 Hz
        DW      TMP174          ; timing data +
        DW      TMM174          ; timing data -
        DW      DT174           ; output data
        DB      20              ; data length (1/3 cycle)
        DB      11              ; add/sub counter
        .
        .
        .
```

6.5　制御方法（プログラム編）

6.5.1　状態遷移図，状態遷移表

図6－14に状態遷移図，表6－5に状態遷移表を示します。ただし，状態遷移図には，NMIによる緊急停止処理は発生箇所を断定できないため，含まれていません。

図6－14　状態遷移図

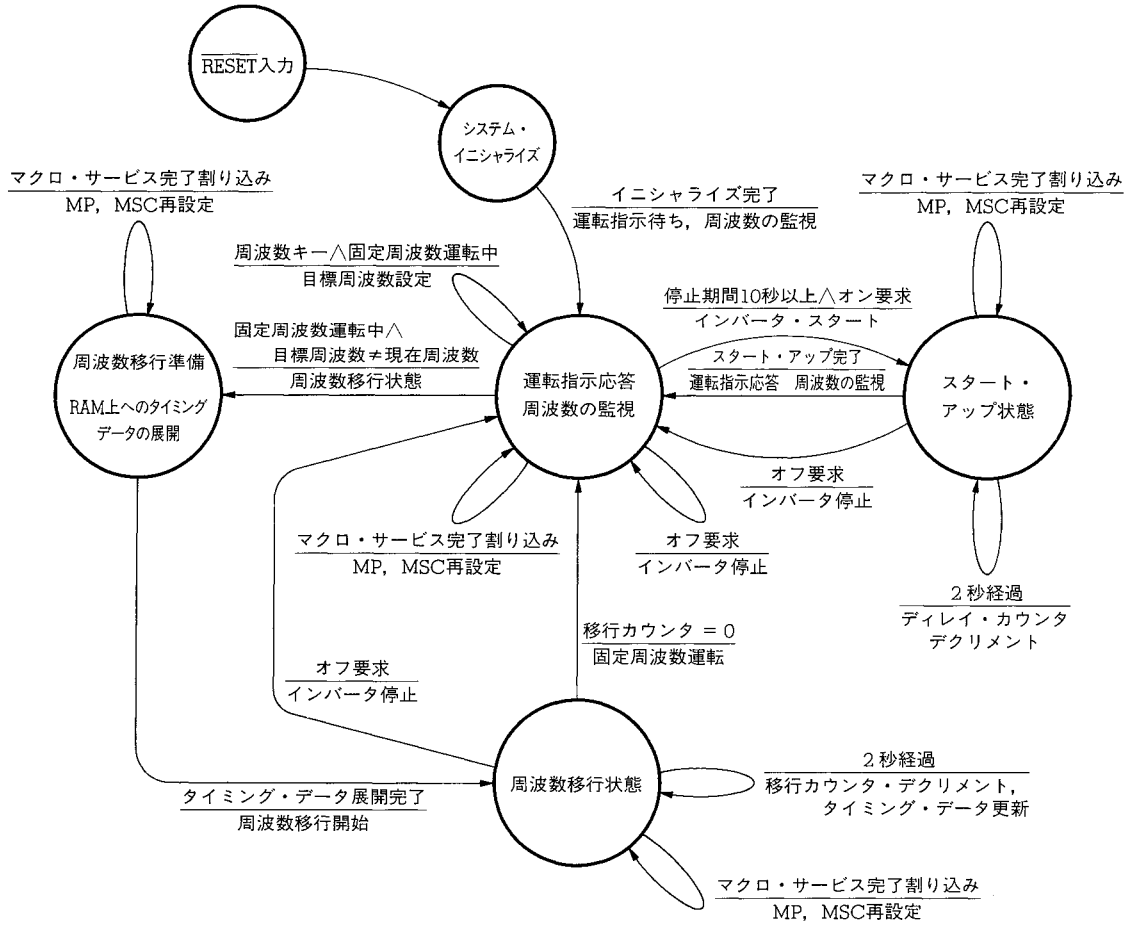


表6-5　状態遷移表

要求＼状態	① 完全停止	② スタート・アップ中	③ 固定周波数運転	④ 周波数移行状態	⑤ 停止完了待ち状態	⑥ イニシャライズ	⑦ オフロック	遷移条件 FMFLAG	STUPOK	STPOK	FSTAT	FAT	ONFLAG	OFFLAG	遷移によるフラグ，ワークの変化 FMFLAG	STUPOK	STPOK	FSTAT	FAT	ONFLAG	OFFLAG	DELAY
ON要求　*2	②	—	—	—	P	—	—	0	?	1	0	?	1	0	—	0	—	01H	—	0	—	4
OFF要求　*2	—	⑤	⑤	⑤	—	—	—	?	?	1	?	?	0	1	0	—	0	0	01H	—	0	5
周波数指示　*2	P	P	④	P	P	—	—	0	1	1	*1	*1	0	0	1	—	—	1-7	1-7	—	—	—
NMI発生	⑦	⑦	⑦	⑦	⑦	⑦	—	?	?	?	?	?	?	?	—	—	—	—	—	—	—	—
RESET	⑥	⑥	⑥	⑥	⑥	⑥	⑥	?	?	?	?	?	?	?	0	0	0	0	01H	0	0	4
処理完了状態		③		③	①	⑤																
状態 FMFLAG	0	0	0	1	0	0	?															
状態 STUPOK	1	0	1	1	1	1	?															
状態 STPOK	1	1	1	1	0	0	?															
状態 FSTAT	0	1	FAT	1-7	0	0	?															
状態 FAT	1	?	1-7	1-7	1	1	?															
状態 DELAY	0	4	0	0	5	4	?															

*1：FAT≠FSTAT
*2：バックグランド処理で設定します
○：移行先
—：受け付けず
？：不定
P：保留

備考　フラグおよびワークについては，**6.5.3　ワークRAM，フラグ**を参照してください。

6.5.2　サブルーチン構成

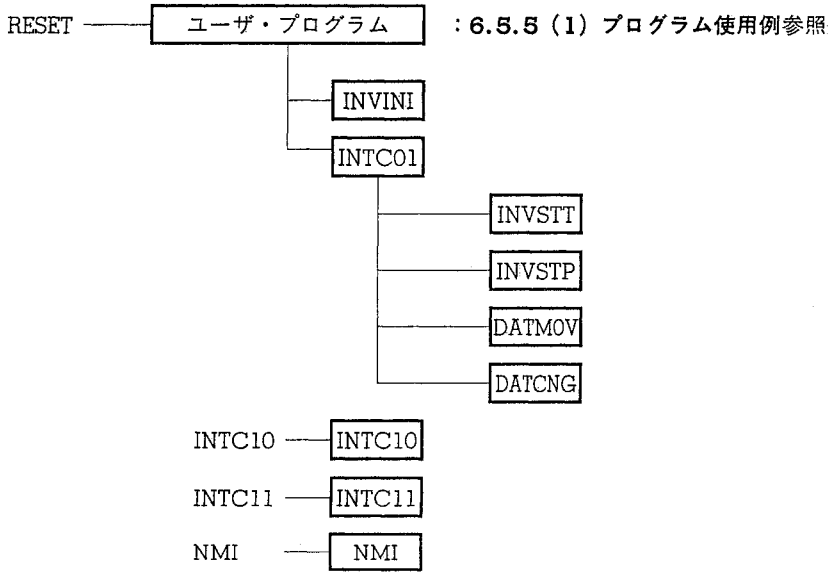
図6－14，図6－15の動作を実現するために，表6－6に示すサブルーチンを使用しています。

表6－6　サブルーチン一覧

SYMBOL	処　理　概　要
INVINI	インバータ動作のためのイニシャライズ処理
INVSTT	インバータ・スタート処理
INVSTP	インバータ停止処理
NMI	インバータ緊急停止処理
DATMOV	周波数移行状態への設定
INTC01	運転指示応答処理，ディレイ処理
DATCNG	周波数移行時のタイミング・データ更新処理
INTC10	＋側マクロ・サービス再起動処理
INTC11	－側マクロ・サービス再起動処理

これらのサブルーチンの関連図を図6－15に示します。

図6－15　サブルーチン関連図



6.5.3　ワークRAM，フラグ

この応用例で使用しているワークRAMとフラグを表6-7，表6-8に示します。

表6-7　ワークRAM

SYMBOL	バイト数	用　　途
BUFP0	128	周波数移行中の＋側タイミング・データ領域0
BUFM0	128	周波数移行中の－側タイミング・データ領域0
BUFP1	128	周波数移行中の＋側タイミング・データ領域1
BUFM1	128	周波数移行中の－側タイミング・データ領域1
TDP	2	タイミング・データ＋側先頭アドレス
TDM	2	タイミング・データ－側先頭アドレス
DT	2	出力データ先頭アドレス
MSCD	1	マクロ・サービス・カウンタ設定値
FSTAT	1	現在の周波数
FAT	1	目標周波数
FMCNT	1	周波数移行カウンタ
SECCNT	1	2秒用カウンタ
DSLP	1	＋側出力データ選択カウンタ
DSLM	1	－側出力データ選択カウンタ
DELAY	1	ディレイ・カウンタ（DELAY＋1）×2が待ち時間

表6-8　フラグ

SYMBOL	用　　途	0の意味	1の意味
FMFLAG	周波数移行状態	非周波数移行状態	周波数移行状態
ASFLAG	加減フラグ	加算モード	減算モード
STUPOK	スタート完了フラグ	非完了状態	完了状態
STPOK	停止完了フラグ	非完了状態	完了状態
ONFLAG	ONフラグ	ON要求なし	ON要求あり
OFFLAG	OFFフラグ	OFF要求なし	OFF要求あり
BNKSLT	バッファRAM選択状態	バッファRAM0使用中	バッファRAM1使用中

前記のワークRAMとフラグは，それぞれのサブルーチンで次のように操作されます。

表6-9　ワークRAM，フラグ動作

サブ・ルーチン / フラグ，ワークRAM	INVINI	INVSTT	INVSTP	NMI	DATMOV		INTC01	DATCNG		INTC10	INTC11
					(FAT) > (FSTAT)	(FAT) < (FSTAT)		(FMCNT) ≧ 0	(FMCNT) < 0		
FMFLAG	0	—	0	—	1	1	(C)	—	0	—	—
ASFLAG	—	—	—	—	1	0	—	C	—	—	—
STUPOK	0	—	—	—	—	—	(*)	—	—	—	—
STPOK	0	—	0	—	—	—	(*)	—	—	—	—
ONFLAG	0	0	0	—	—	—	C	—	—	—	—
OFFLAG	0	—	0	—	—	—	(*)	—	—	—	—
BNKSLT	—	—	—	—	0	0	—	*	—	—	—
TDP	—	33Hz	—	—	BUFP	BUFP	—	—	*	R	—
TDM	—	33Hz	—	—	BUFM	BUFM	—	—	*	—	R
DT	—	33Hz	—	—	*	*	—	—	*	—	—
MSCD	—	33Hz	—	—	*	*	—	—	*	—	—
PMD	—	33Hz	—	—	*	*	—	—	*	—	—
FSTAT	0	01H	0	—	+1	−1	(C)	—	—	—	—
FAT	01H	—	01H	—	C	C	(C)	—	—	—	—
FMCNT	—	—	—	—	*	*	—	−1	−1	—	—
SECCNT	40	—	—	—	—	—	−1	—	—	—	—
DSLP	—	0	—	—	—	—	—	—	—	*	—
DSLM	—	0	—	—	—	—	—	—	—	—	*
DELAY	4	4	5	—	—	—	(*)	—	—	—	—

備考　—　：操作せず

＊　：変化

C　：チェック

R　：読み出しのみ

（ ）：（ ）内の動作をすることがある

6.5.4　コントロール・エリア構成

マクロ・サービス・チャネルを管理するために，1つの周波数に対し8バイトのコントロール・エリアを設けています。

図6－16に，208 Hzの場合のコントロール・エリアを示します。

図6－16　208 Hzの場合のコントロール・エリア

```
CNT208：
        DW      TMP208      ；＋側タイミング・データの先頭アドレス
        DW      TMM208      ；－側タイミング・データの先頭アドレス
        DW      DT208       ；出力データの先頭アドレス
        DB      18          ；1/3周期のデータ数
        DB      6           ；周波数移行時の移行カウント数
```

このコントロール・エリアの「周波数移行時の移行カウント数」を除いた値をsaddr領域のTDP以降（TDP，TDM，DT，MSCD）に転送し，マクロ・サービスの再設定に使用しています。なお，TDP以降の配置は，ギャップを含まず，TDP，TDM，DT，MSCDの順序でなければなりません。

6.5.5　プログラム説明

(1) プログラム使用例

実際のインバータでは，外部から何らかの要求で，目標の周波数を出力することになります。この応用例では，インバータを制御する部分のみについて扱っていますので，これに対して運転指令を行う使用例を以降で説明します。なお，インバータ制御プログラムを使用するにあたっては，次のように，外部参照疑似命令を記述してください。

```
EXTRN     INVINI, INTC01          ; process
EXTRN     FAT                     ; value
EXTBIT    ONFLAG, OFFLAG          ; bit
```

(a) サブルーチン

インバータ制御プログラムを動作させるために表6-10に示す，3つの外部から呼び出すサブルーチンを用意しています。

表6-10　外部呼び出しサブルーチン一覧

SYMBOL	処　理　概　要
INVINI	インバータ動作のためのイニシャライズ処理
INTC01	運転指示応答処理，ディレイ処理

(i) INVINI処理

インバータ制御に必要なイニシャライズ処理を行います。最初に1回呼び出してください。次に呼び出し例を示します。

```
EXTRN     INVINI, INTC01 ; 呼び出し処理の外部参照疑似命令
          .
CALL      ! INVINI       ; インバータのイニシャライズ
          .
```

(ii) INTC01処理

動作のON/OFF，運転周波数のチェックを行っています。INVINIによってINTC01の割り込みインターバル時間は50 msに設定されるので，メイン・ループのフラグ・チェックによって呼び出してください。次に呼び出し例を示します。

```
        EXTRN     INVINI, INTC01 ; 呼び出し処理の外部参照疑似命令
                  .
                  .
MAIN :
                  .
                  .
;
;       ---CHECK    INTC01---
;
CHKC01 :
        BF        CIF01, $MAIN ; CIF01のフラグのチェック
        CLR1      CIF01        ; セットされていたらクリア
        CALL      !INTC01      ; INTC01の呼び出し
        BR        MAIN
```

(b) 媒介シンボル

運転指示は，シンボル（ワークRAM）に設定された値で行います。インバータ制御プログラムに運転指示を行うシンボルを表6-11-表6-13に示します。

```
EXTRN     INVINI, INTC01          ; process
EXTRN     FAT                     ; value
EXTBIT    ONFLAG, OFFLAG          ; bit
```

表6-11　ワークRAM

SYMBOL	バイト数	用　　途
FAT	1	目標周波数を設定

表6-12　FATとインバータ出力周波数

FAT	1	2	3	4	5	6	7
周波数[Hz]	32.55	43.40	65.10	86.81	130.21	173.61	208.33

表6-13　フ ラ グ

SYMBOL	用　　途
ONFLAG	ONする場合に1にセット注
OFFLAG	OFFする場合に1にセット注

注　ONFLAGとOFFLAGは同時に1に設定しないでください。

これらのシンボルに設定された値は，前述のINTC01処理内で参照し，処理されますので，INTC01を呼び出す前に，シンボル値を変更するのが適当です。次に一例を示します。

```
                .
                .
        MOV     FAT, A          ; 目標周波数を設定
                .
                .
        SET1    ONFLAG          ; インバータ・スタート指示
                .
                .
CHKC01:
        BF      CIF01, $MAIN    ; CIF01のフラグのチェック
        CLR1    CIF01           ; セットされていたらクリア
        CALL    !INTC01         ; INTC01の呼び出し
                .
                .
```

(2) イニシャライズ処理

サブルーチン名：INVINI

(a) 処理内容

インバータ関係のワーク，フラグおよびタイマ割り込みのイニシャライズを行っています。

(i) ワーク，フラグのイニシャライズ

インバータの初期状態として，表6-14，表6-15のように設定します。

表6－14　ワークRAM

SYMBOL	設定値	設　定　値　の　意　味
FSTAT	0	現在の周波数を停止状態として0に設定
FAT	1	初期起動周波数として33 Hzを指定
SECCNT	1	2秒用タイマのカウンタに1を設定（50 ms後に2秒タイマ処理を実行）
DELAY	4	RESET後ディレイとして10秒に設定

表6－15　フ ラ グ

SYMBOL	設定値	設　定　値　の　意　味
FMFLAG	0	非周波数移行状態
STUPOK	1	スタート・アップ完了状態
STPOK	0	停止非完了状態
ONFLAG	0	ON要求なし
OFFLAG	0	OFF要求なし

(ii) タイマ割り込みのイニシャライズ

この応用例では，表6－16に示すタイマ割り込みを使用しています。

表6－16　使用するタイマ割り込み

割り込みソース	用　途	周期 [ms]	優先順位	処理形態	マスク設定
INTC10	インバータ制御用（＋側）	—	高位	マクロ・サービス	動作時解除
INTC11	インバータ制御用（－側）	—	高位	マクロ・サービス	動作時解除
INTC01	移行，スタート，停止監視用	50	低位	フラグ・チェック	マスク状態

① INTC01のイニシャライズ

周波数移行，スタート・アップ，停止およびディレイの監視用として，50 ms周期のINTC01の割り込みを使用しています。表6－17に示すSFRの設定を行っています。

表6－17　INTC01のために設定するSFR

SFR	設定値	設　定　値　の　意　味
CRC0	00011000B	TM0とCR01の一致でTM0をクリア
CR01	37500-1	50 msのインターバルに設定

図6－17　CRC0レジスタの設定

キャプチャ／コンペア・コントロール・レジスタ0

	7	6	5	4	3	2	1	0	
CRC0	MOD1	MOD0	0	1	CLR01	0	0	0	（リセットで10H）
設定例	0	0	0	1	1	0	0	0	

MOD1	MOD0	CLR01	タイマ出力モード指定		TM0＝CR01時の
			TO0	TO1	TM0のクリア動作
0	0	0	トグル出力	トグル出力	禁止
0	0	1	トグル出力	トグル出力	許可
0	1	0	PWM出力	トグル出力	禁止
0	1	1	設定禁止		
1	0	0	PWM出力	PWM出力	禁止
1	0	1	設定禁止		
1	1	0	設定禁止		
1	1	1	PPG出力	トグル出力	許可

②　INTC10，INTC11のイニシャライズ

INTC10，INTC11は，マクロ・サービス・タイプCでインバータ制御を行うための割り込みです。ここでは，表6－18に示すSFRとマクロ・サービス・コントロール・ワードの設定を行っています。

表6－18　INTC10，INTC11のイニシャライズ一覧

SYMBOL	設定値	設　定　値　の　意　味
CRC1	00000000B	TM1のクリア動作を禁止
CR10	0	初期値として0を設定
CR11		
MSMC10	11101010B	タイプC，自動加算，POLへ転送
MSMC11	11101011B	タイプC，自動加算，POHへ転送
CHPC10	LOW CHAC10	マクロ・サービス・チャネルの最上位アドレスの下位8ビット
CHPC11	LOW CHAC11	
CPR10	0	INTC10，INTC11はインバータ制御に使用するため，タイミングの関係で高優先に設定
CPR11		
CISM10	1	割り込みサービス・モードをマクロ・サービスに設定
CISM11		
CMK10	0	INTC10，INTC11の割り込みマスク・フラグを解除
CMK11		

図6－18　割り込み関連のモード・レジスタの設定

キャプチャ／コンペア・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
CRC1	0	0	0	0	CLR11	CM	CLR10	0	（リセットで00H）
設定例	0	0	0	0	0	0	0	0	

CLR10	TM1とCR10レジスタの内容一致でのクリア動作
0	禁止
1	許可

CLR11	CM	CR11レジスタの動作指定	TM1のクリア動作
0	0	コンペア・レジスタ	禁止
1	0		許可（TM1とCR11レジスタの内容一致のとき）
0	1	キャプチャ・レジスタ	禁止
1	1		許可（TM1の内容をCR11レジスタにキャプチャしたとき）

マクロ・サービス・モード・レジスタ（INTC10）

7	6	5	4	3	2	1	0	
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0	（リセットで00H）
設定例 1	1	1	0	1	0	1	0	

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0		リング制御なし	データ転送のみ	POL
1	0	0	1				POH
1	0	1	0			自動加算つき	POL
1	0	1	1				POH
1	1	0	0		リング制御つき	データ転送のみ	POL
1	1	0	1				POH
1	1	1	0			自動加算つき	POL
1	1	1	1				POH

マクロ・サービス・モード・レジスタ（INTC11）

	7	6	5	4	3	2	1	0
	CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0
設定例	1	1	1	0	1	0	1	1

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし	データ転送のみ　POL
1	0	0	1				POH
1	0	1	0				自動加算つき　POL
1	0	1	1				POH
1	1	0	0			リング制御つき	データ転送のみ　POL
1	1	0	1				POH
1	1	1	0				自動加算つき　POL
1	1	1	1				POH

優先順位指定フラグ・レジスタL

	7	6	5	4	3	2	1	0	
PROL	CPR11	CPR10	CPR01	CPR00	PPR3	PPR2	PPR1	PPR0	（リセットでFFH）
設定例	0	0	1	1	1	1	1	1	

PR	優先順位指定フラグ
0	高優先順位レベル
1	低優先順位レベル

割り込みサービス・モード・レジスタL

	7	6	5	4	3	2	1	0	
ISMOL	CISM11	CISM10	CISM01	CISM00	PISM3	PISM2	PISM1	PISM0	（リセットで00H）
設定例	1	1	0	0	0	0	0	0	×：操作しません

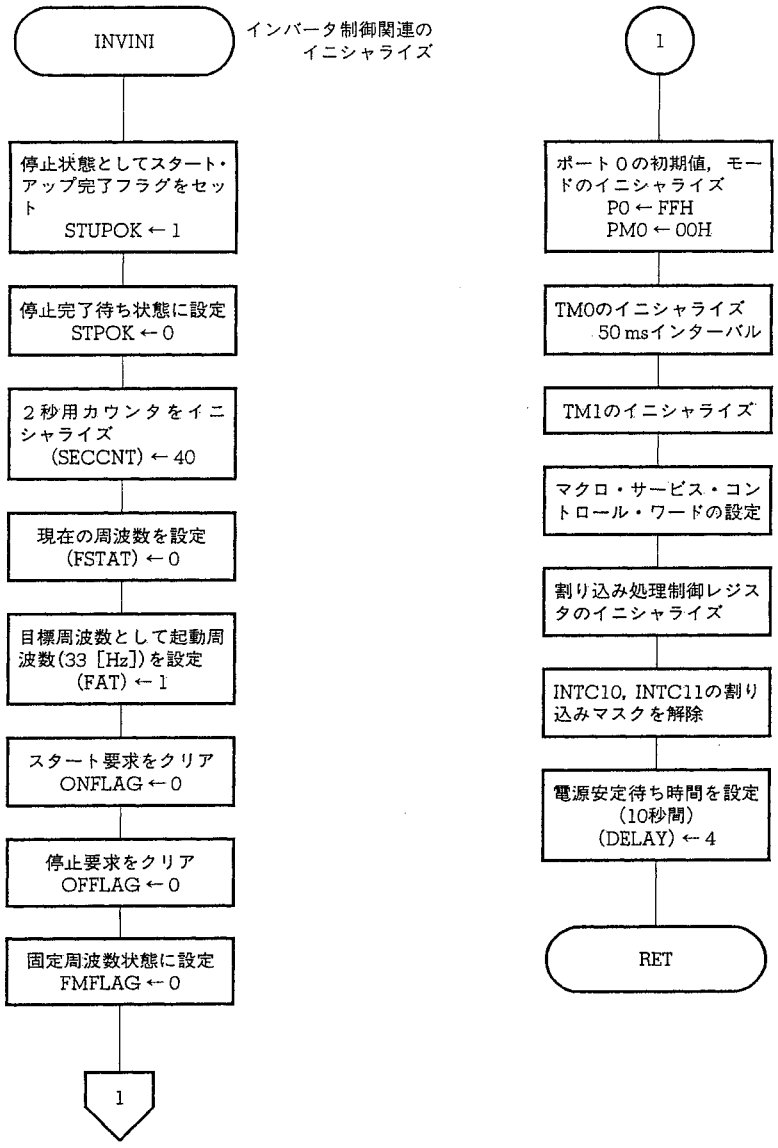
ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

割り込みマスク・レジスタL

	7	6	5	4	3	2	1	0	
MKOL	CMK11	CMK10	CMK01	CMK00	PMK3	PMK2	PMK1	PMK0	（リセットでFFH）
設定例	0	0	1	1	1	1	1	1	×：操作しません

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

(b) フロー・チャート


INVINI
インバータ制御関連の
イニシャライズ
停止状態としてスタート・
アップ完了フラグをセッ
ト
STUPOK ← 1
停止完了待ち状態に設定
STPOK ← 0
2秒用カウンタをイニ
シャライズ
(SECCNT) ← 40
現在の周波数を設定
(FSTAT) ← 0
目標周波数として起動周
波数(33 [Hz])を設定
(FAT) ← 1
スタート要求をクリア
ONFLAG ← 0
停止要求をクリア
OFFLAG ← 0
固定周波数状態に設定
FMFLAG ← 0
1
1
ポート0の初期値，モー
ドのイニシャライズ
P0 ← FFH
PM0 ← 00H
TM0のイニシャライズ
50 msインターバル
TM1のイニシャライズ
マクロ・サービス・コン
トロール・ワードの設定
割り込み処理制御レジス
タのイニシャライズ
INTC10, INTC11の割り
込みマスクを解除
電源安定待ち時間を設定
(10秒間)
(DELAY) ← 4
RET

(3) インバータ・スタート処理

サブルーチン名：INVSTT

(a) 処理内容

インバータの運転開始処理を行います。初期周波数は，33 Hzです。なお，このサブルーチンは，ベクタ割り込み禁止の状態で処理されます。ここでの主な処理を次に示します。

- ●起動周波数の設定
- ●マクロ・サービス用パラメータのRAMへの展開
- ●マクロ・サービス・チャネルの設定
- ●バッファ・レジスタへの初期値設定
- ●リアルタイム出力許可とタイマ・スタート

(i) 起動周波数の設定

起動周波数を最低周波数の33 Hzに設定しています。スタート後のウエイト時間として，10秒を設けるため，スタート・アップ完了フラグをクリアし，INVSTTの終端で，ディレイ時間を10秒に設定しています。

(ii) マクロ・サービス用パラメータのRAMへの展開

マクロ・サービスのパラメータは，各周波数ごとに，PWMデータの中のコントロール・エリアに配置されています。このうち，現在の周波数については，コントロール・エリアの内容を内部RAMに展開し，アクセス時間を節約しています。

(iii) マクロ・サービス・チャネルの設定

マクロ・サービス・ポインタとマクロ・サービス・カウンタの設定を行います。このとき，設定値は内部RAMに展開されたコントロール・エリアのものを転送します。

(iv) バッファ・レジスタへの初期値設定

バッファ・レジスタに設定する初期値は，出力データ領域の最後のデータとしています。アドレス計算を行い，POL，POHそれぞれに初期値を設定しています。

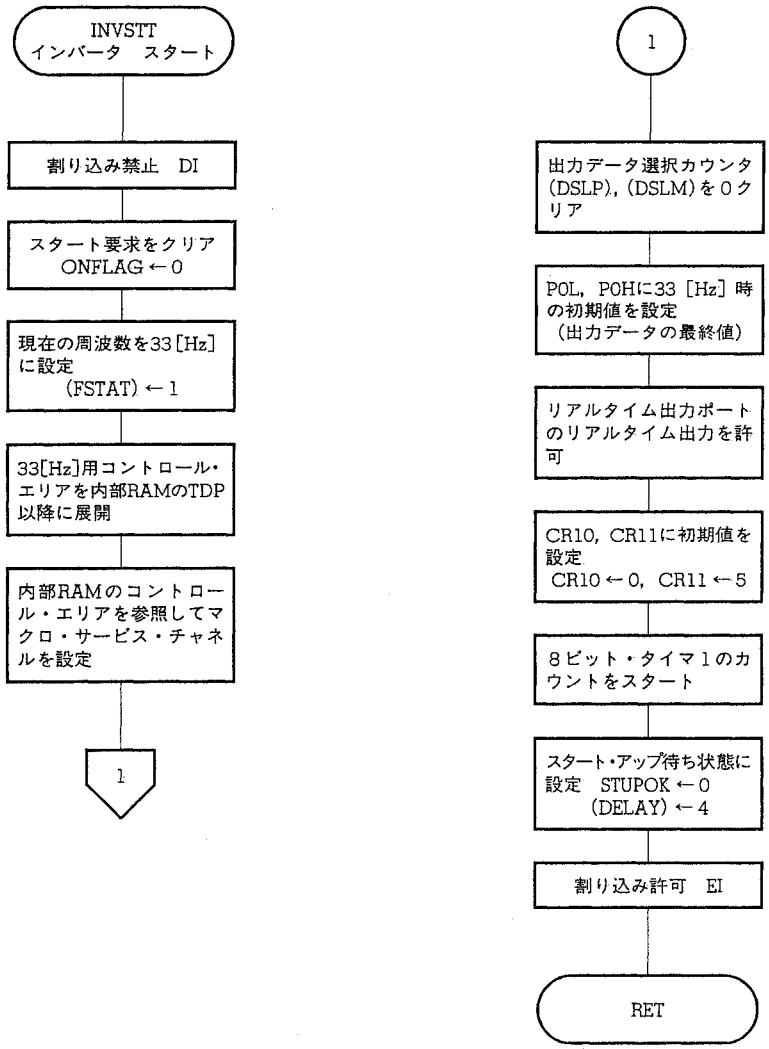
(v) リアルタイム出力許可とタイマ・スタート

P0L, P0Hをそれぞれ4ビット単位のリアルタイム出力ポートとして機能させます。ここで，タイミング・データは，最初の割り込み要求を，INTC10に対して，INTC11をデッド・タイム（13.3 μs）分遅く発生させる形式のため，それぞれのコンペア・レジスタに，表6-19のように初期値を設定しています。このあと，8ビット・タイマ／カウンタ1をスタートし，割り込みを許可しています。

表6-19　コンペア・レジスタの初期値

コンペア・レジスタ	初期値	初期割り込み発生相対時間
CR10	0	13.3 μs
CR11	5	

(b) フロー・チャート



(4) インバータ・ストップ処理

サブルーチン名：INVSTP

(a) 処理内容

インバータの停止処理を行います。ここでは，インバータ・トランジスタのオフを最優先に行います。ここでの主な処理を次に示します。

- ポート0のオフ
- INTC10，INTC11の発生停止
- 停止状態のステータス設定
- 停止時間12秒を設定

(i) ポート0のオフ

インバータ・トランジスタをすべてオフ状態にするために，リアルタイム出力の禁止と，ポート0をすべてインアクティブ状態に設定しています。

(ii) INTC10，INTC11の発生停止

インバータのスイッチングを行っていた，INTC10，INTC11の割り込み要求の発生を停止します。まず，TM1のカウント動作を停止させ，CIF10，CIF11の割り込み要求フラグをクリアしています。

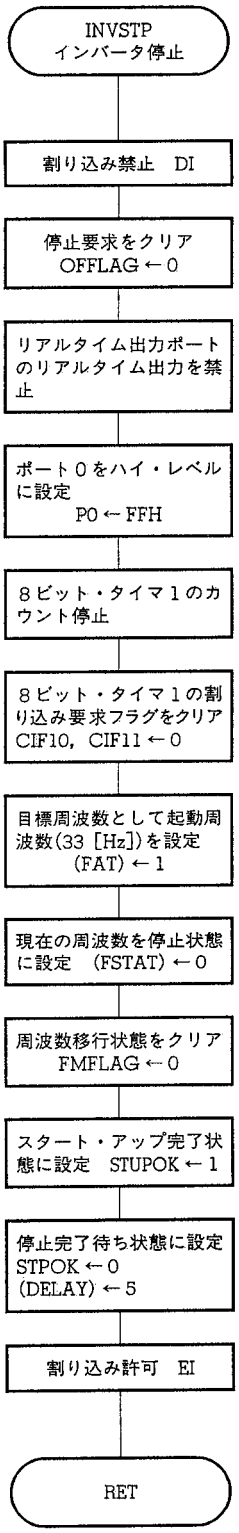
(iii) 停止状態のステータス設定

停止状態とするために，現在の周波数をオフ状態の“0”，目標周波数は，次の起動に備えて，最低周波数の“1”に設定しています。また，周波数移行状態もクリアしています。

(iv) 停止時間12秒を設定

インバータの負荷が実際に停止するまでの待ち時間として，12秒を設けています。そのため，停止完了フラグをクリアし，ディレイ時間を12秒に設定しています。なお，スタート・アップ中にオフされる可能性があるため，スタート・アップ完了フラグをセットしています。

(b) フロー・チャート


INVSTP
インバータ停止
割り込み禁止　DI
停止要求をクリア
OFFLAG ← 0
リアルタイム出力ポート
のリアルタイム出力を禁
止
ポート0をハイ・レベル
に設定
P0 ← FFH
8ビット・タイマ1のカ
ウント停止
8ビット・タイマ1の割
り込み要求フラグをクリア
CIF10, CIF11 ← 0
目標周波数として起動周
波数(33 [Hz])を設定
(FAT) ← 1
現在の周波数を停止状態
に設定　(FSTAT) ← 0
周波数移行状態をクリア
FMFLAG ← 0
スタート・アップ完了状
態に設定　STUPOK ← 1
停止完了待ち状態に設定
STPOK ← 0
(DELAY) ← 5
割り込み許可　EI
RET

(5) NMIによる緊急停止

サブルーチン名：NMI

(a) 処理内容

インバータの緊急停止を行います。ここでは，INVSTP同様インバータ・トランジスタのオフを最優先に行います。なお，NMI処理内では永久ループに入るため，リセット以外では解除できません。ここでの主な処理を次に示します。

- ポート0のオフ
- マスカブル割り込みの禁止

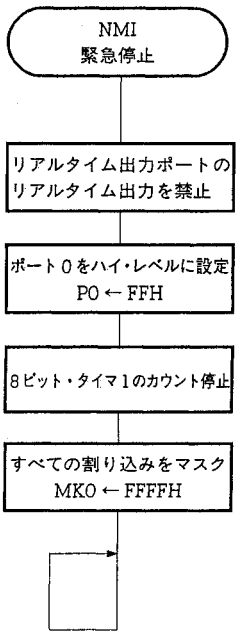
(i) ポート0のオフ

インバータ・トランジスタをすべてオフ状態にするために，リアルタイム出力の禁止と，ポート0をすべてインアクティブ状態に設定しています。

(ii) マスカブル割り込みの禁止

TM1のカウント動作を停止させ，すべてのマスカブル割り込みをマスク状態に設定します。このあと永久ループとなります。

(b) フロー・チャート



(6) INTC01タイマ割り込み処理

サブルーチン名：INTC01

(a) 処理内容

この処理は，INTC01割り込み要求フラグ(CIF01)のチェックで50 msごとに呼び出してください。ここでは，周波数の監視と，周波数移行速度を制御しています。この処理が状態遷移の中枢となっています。ここでの主な処理を次に示します。

- ●停止要求処理
- ● ２秒タイマ処理
- ●ディレイ処理
- ●スタート・アップ処理
- ●停止完了待ち処理
- ●周波数移行処理呼び出し
- ●周波数変更呼び出し処理
- ●スタート要求処理

(i) 停止要求処理

停止要求のサンプリングは，50 msごとに行い，停止要求があった場合は，インバータ停止処理［INVSTP］を呼び出します。ここで，すでに停止中の場合（FSTAT＝0）は，停止処理は行いません。

(ii) ２秒タイマ処理

50 msのインターバルのINTC01割り込み要求から，２秒用カウンタ（SECCNT）を使用して，２秒のタイマを作っています。

(iii) ディレイ処理

オン要求と，周波数指示には，次の状態に遷移するまでの時間を設定できます。

それぞれの時間は，DELAYに設定します。この値が０以外の場合は，２秒ごとにデクリメントされ，０の場合は，オン要求，周波数指示，周波数移行を受け付けることになります。ただし，複数の要求の競合を避けるために，表６－20に示すように，優先順位を付け，同時に複数の処理を行わないようにしています。

表6－20　処理要求優先順位

優先順位	処理要求	呼び出しサブルーチン
1（高位）	緊急停止要求（NMI）注1	NMI
2	停止要求注2	INVSTP
3	スタート・アップ完了待ち	—
4	停止完了待ち	—
5	タイミング・データ更新	DATCNG
6	周波数移行設定	DATMOV
7（低位）	インバータ・スタート	INVSTT

注1．NMIで受け付けられます。

2．50 msごとに受け付けられます。

(iv) スタート・アップ処理

オン要求によってインバータのスタートが行われますが，負荷がモータの場合は，すべりが最小になるまで，ある程度の時間を必要とします。この応用例では，INVSTTで10秒をDELAYに設定しています。ここでは，DELAYが0になった場合，スタート・アップ完了状態として，STUPOK←1としています。

(v) 停止完了待ち処理

オフ要求によってインバータの停止が行われますが，負荷がモータの場合は，実際に停止するまで，ある程度の時間を必要とします。この応用例では，INVSTPで12秒をDELAYに設定しています。ただし，停止要求は50 msごとに受け付け，また，停止完了状態から別の状態に移行するにはさらに2秒必要とするため，実際の停止完了待ち時間は，12秒～14秒となります。ここでは，DELAYが0になった場合，停止完了状態として，STPOK←1としています。

(vi) 周波数移行処理呼び出し

周波数移行状態に設定されている場合は，2秒ごとに，タイミング・データの更新を行います。したがって，FMFLAG＝1の場合は，DATCNG処理を呼び出します。

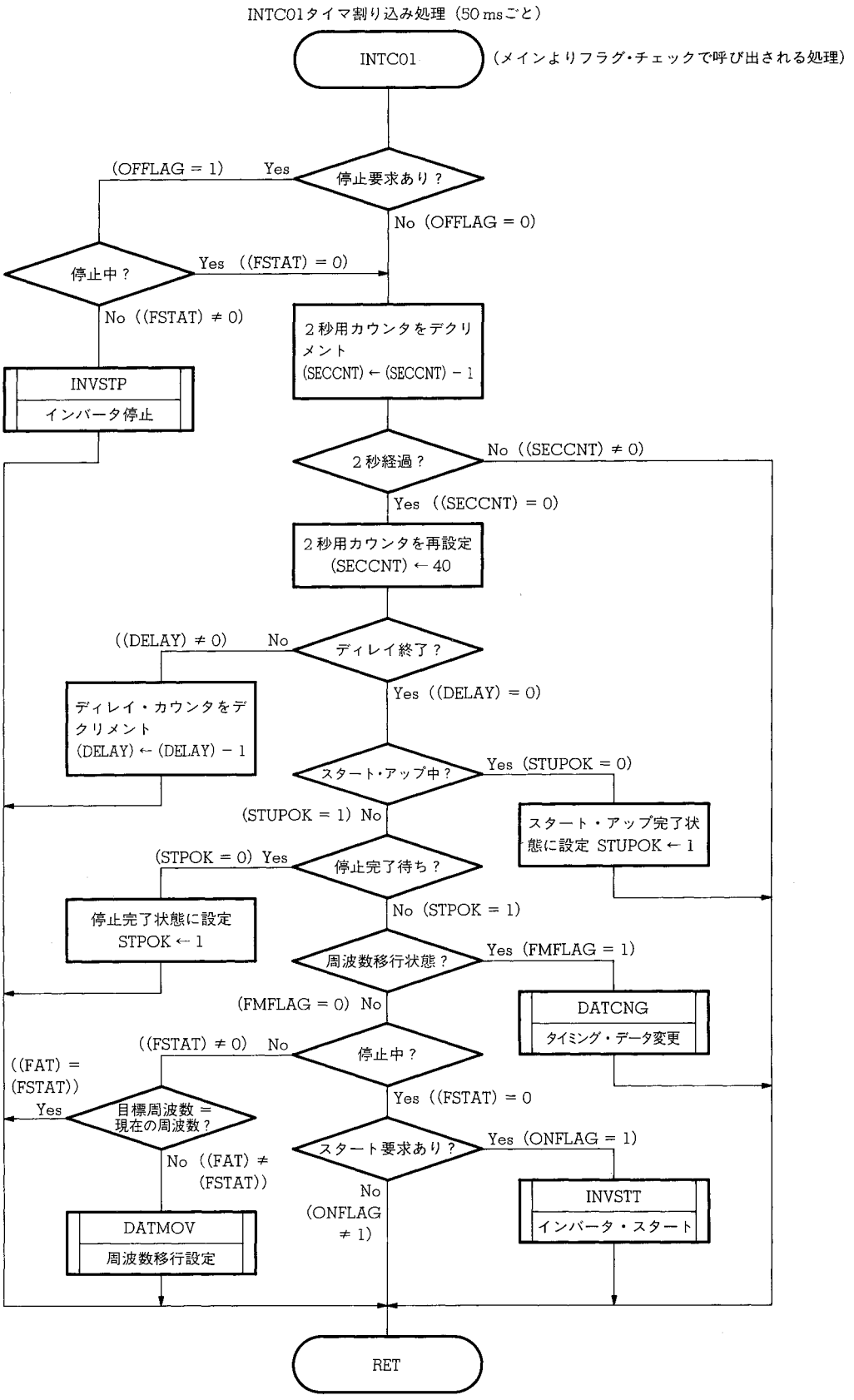
(vii) 周波数変更呼び出し処理

停止中以外で，目標周波数と，現在の周波数が異なる場合は，周波数移行状態に設定します。周波数移行状態への設定と，データ処理は，DATMOVで行われます。

(viii) スタート要求処理

スタート要求は，INTC01処理内で，最後にチェックされます。このときは，完全停止状態です。スタート要求がある場合は，インバータ・スタート処理INVSTTを呼び出します。

(b) フロー・チャート


INTC01タイマ割り込み処理（50 msごと）
INTC01
（メインよりフラグ・チェックで呼び出される処理）
停止要求あり？
(OFFLAG = 1)
Yes
No (OFFLAG = 0)
停止中？
Yes ((FSTAT) = 0)
No ((FSTAT) ≠ 0)
2秒用カウンタをデクリメント
(SECCNT) ← (SECCNT) − 1
INVSTP
インバータ停止
2秒経過？
No ((SECCNT) ≠ 0)
Yes ((SECCNT) = 0)
2秒用カウンタを再設定
(SECCNT) ← 40
ディレイ終了？
((DELAY) ≠ 0)
No
Yes ((DELAY) = 0)
ディレイ・カウンタをデクリメント
(DELAY) ← (DELAY) − 1
スタート・アップ中？
Yes (STUPOK = 0)
(STUPOK = 1) No
スタート・アップ完了状態に設定 STUPOK ← 1
停止完了待ち？
(STPOK = 0) Yes
No (STPOK = 1)
停止完了状態に設定
STPOK ← 1
周波数移行状態？
Yes (FMFLAG = 1)
(FMFLAG = 0) No
DATCNG
タイミング・データ変更
停止中？
((FSTAT) ≠ 0) No
Yes ((FSTAT) = 0
((FAT) = (FSTAT))
Yes
目標周波数 = 現在の周波数？
No ((FAT) ≠ (FSTAT))
スタート要求あり？
Yes (ONFLAG = 1)
No (ONFLAG ≠ 1)
DATMOV
周波数移行設定
INVSTT
インバータ・スタート
RET

(7) 周波数移行状態設定処理

サブルーチン名：DATMOV

(a) 処理内容

この処理は，INTC01処理から，停止中以外で，現在の周波数と目標周波数が異なる場合に呼び出されます。ここでは，現在の周波数と目標周波数を比較し，それによってタイミング・データにオフ・セットを加えてRAMに展開しています。このRAMに展開されたタイミング・データは後述のDATCNG処理で2秒ごとに更新されます。ここで，各周波数におけるパラメータを表6-21に示します。

表6-21　周波数移行時の変動周波数と加減回数（f_{CLK}[注1]＝6 MHz）

キャリア周波数［kHz］	キャリア周期［μs］（＝α）	f_{CLK}/16に対する周期の倍率
3.125	320	120

周波数　f［Hz］	32.55	43.40	65.10	86.81	130.21	173.61	208.33
周期　　T［ms］	30.72	23.04	15.36	11.52	7.68	5.76	4.80
T/α（＝β）	96	72	48	36	24	18	15
最大データ数 β×2（1/3周期）	192	144	96	72	48	36	30
$\frac{T/3}{\text{タイマ・カウント・クロック}}$（＝m）	3720	2880	1920	1440	960	720	600
実際のデータ数（＝n）	98	74	50	36	26	20	18
変動周波数［Hz］[注2]	—	1.087	1.652	2.117	3.433	4.692	6.068
加減回数＝$\frac{m'-m}{n}$	—	11.351 →11	19.200 →19	13.333 →13	18.462 →18	12.000 →11	6.666 → 6

注1. f_{CLK}：内部システム・クロック周波数

2. 変動周波数＝$\left|\frac{f_{CLK}/16}{3}\left(\frac{1}{m+n}-\frac{1}{m}\right)\right|$

ここでの主な処理を次に示します。

- ●タイミング・データ展開
- ●コントロール・データの更新

(i) タイミング・データ展開

周波数の移行は，タイミング・データの合計値を増減することで行っています。表6－21における加減回数が，隣接する周波数に移行する場合のタイミング・データの増減回数となります。この増減の動作は，周波数の移行方向によって表6－22のように異なります。

表6－22　増減動作

移行方向	増減動作	ASFLAG
低周波→高周波	デクリメント	1
高周波→低周波	インクリメント	0

ここで，低周波→高周波の移行の場合，元のデータをデクリメントした場合は，アンダフローの発生が考えられます。そこで，低周波→高周波の移行の場合は，高周波側のタイミング・データのすべてに加減回数を足したものを使用し，これをデクリメントします。なお，ここでバッファRAMに展開するものは，＋側，－側とも2つあるバッファRAMのうち，片方だけ（バッファRAM0：FC00H-FCFFH）に展開します。

図6－19に208 Hz→174 Hzの場合，図6－20に174 Hz→208 Hzの場合の展開について示します。

図6－19　208 Hz→174 Hzの場合のデータ展開

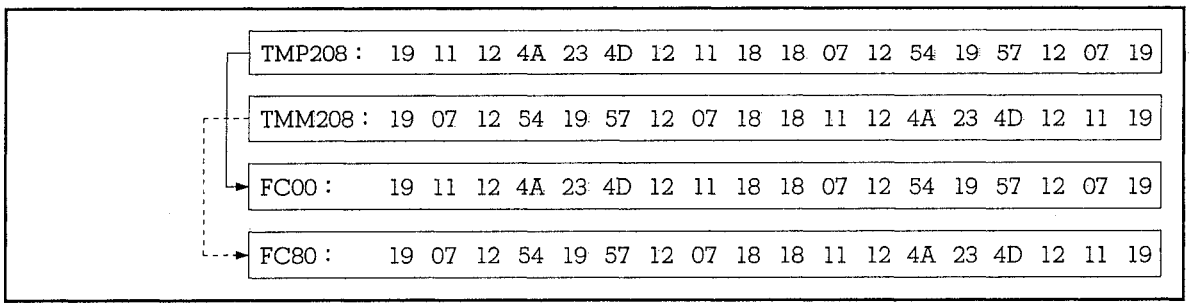
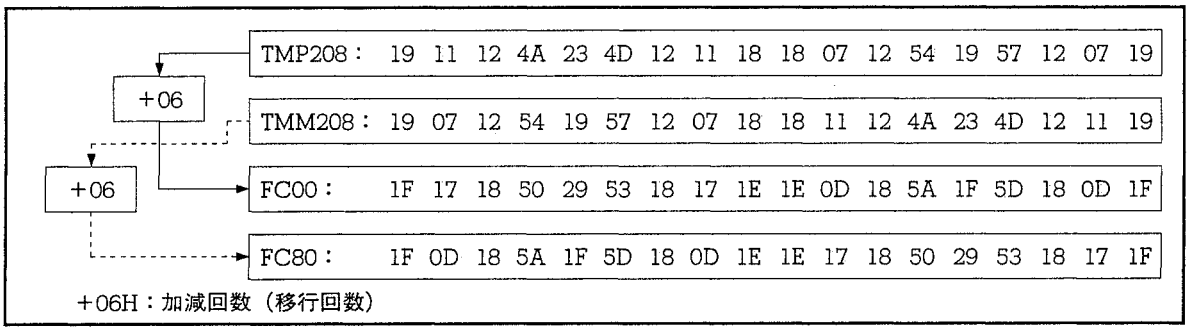


図6－20　174 Hz→208 Hzの場合のデータ展開

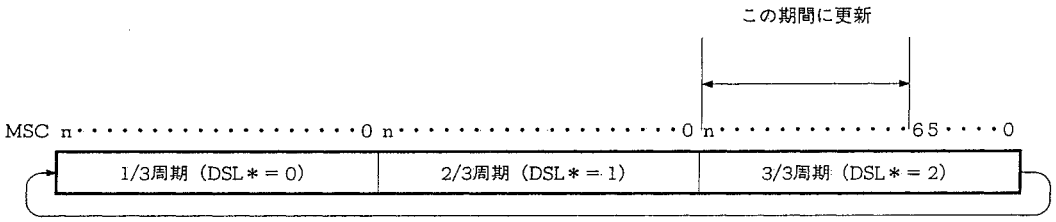


(ii) コントロール・データの更新

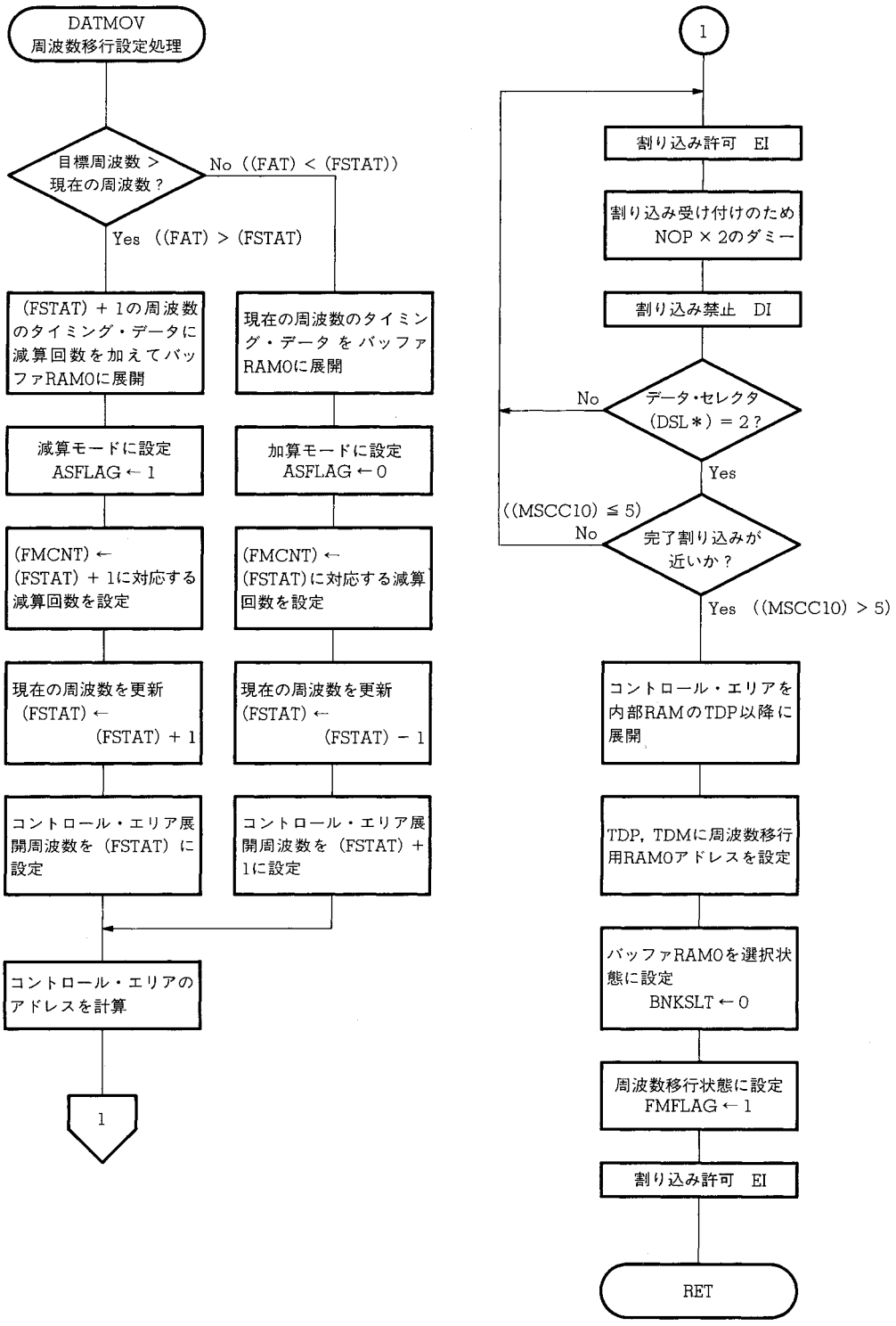
コントロール・データの更新は，出力データのパターンが変わるため，3/3周期で行う必要があります。また，更新中にマクロ・サービス完了割り込みが発生しないようにするため，マクロ・サービス完了割り込みが発生する付近を避けて更新を行っています(図6－21参照)。

更新中に，ベクタ割り込みが発生して，更新処理が中断されると，更新中にマクロ・サービス完了割り込みが発生する可能性があるため，ベクタ割り込みを禁止してこれらの処理を行っています。

図6－21　コントロール・データの更新タイミング



(b) フロー・チャート


DATMOV
周波数移行設定処理
目標周波数 >
現在の周波数？
No ((FAT) < (FSTAT))
Yes ((FAT) > (FSTAT)
(FSTAT) + 1の周波数のタイミング・データに減算回数を加えてバッファRAM0に展開
現在の周波数のタイミング・データをバッファRAM0に展開
減算モードに設定
ASFLAG ← 1
加算モードに設定
ASFLAG ← 0
(FMCNT) ←
(FSTAT) + 1に対応する減算回数を設定
(FMCNT) ←
(FSTAT)に対応する減算回数を設定
現在の周波数を更新
(FSTAT) ←
(FSTAT) + 1
現在の周波数を更新
(FSTAT) ←
(FSTAT) − 1
コントロール・エリア展開周波数を (FSTAT) に設定
コントロール・エリア展開周波数を (FSTAT) + 1に設定
コントロール・エリアのアドレスを計算
1
1
割り込み許可　EI
割り込み受け付けのため
NOP × 2のダミー
割り込み禁止　DI
No
データ・セレクタ
(DSL*) = 2？
Yes
((MSCC10) ≦ 5)
No
完了割り込みが
近いか？
Yes ((MSCC10) > 5)
コントロール・エリアを内部RAMのTDP以降に展開
TDP, TDMに周波数移行用RAM0アドレスを設定
バッファRAM0を選択状態に設定
BNKSLT ← 0
周波数移行状態に設定
FMFLAG ← 1
割り込み許可　EI
RET

(8) タイミング・データ更新処理

サブルーチン名：DATCNG

(a) 処理内容

周波数移行時のデータの更新処理と移行完了時の固定データへの切り替えを行っています。この処理の呼び出しは，周波数移行状態時に２秒ごとに行っています。

周波数移行時のタイミング・データは，バッファRAM（FC00H-FDFFH）に展開しています。バッファRAMは２組あり，マクロ・サービス出力用ではない方のバッファRAMのデータを，２秒ごとにインクリメント／デクリメントし，加減算完了後，マクロ・サービス出力用に設定します。なお，＋側／－側のバッファ領域を表６－23に示します。

表６－23　バッファ領域

SYMBOL	バイト数	アドレス	用　　途
BUFP0	128	FC00H-FC7FH	周波数移行中の＋側タイミング・データ領域０
BUFM0	128	FC80H-FCFFH	周波数移行中の－側タイミング・データ領域０
BUFP1	128	FD00H-FD7FH	周波数移行中の＋側タイミング・データ領域１
BUFM1	128	FD80H-FDFFH	周波数移行中の－側タイミング・データ領域１

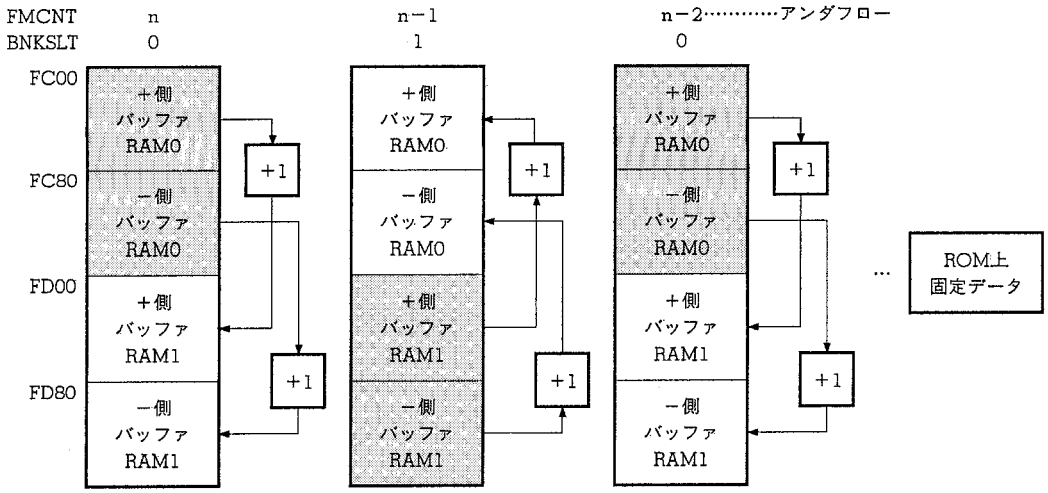
ここでの主な処理を次に示します。

- タイミング・データ増減処理
- 移行完了処理

(i) タイミング・データ増減処理

周波数移行状態設定処理（DATMOV）で設定された周波数移行カウンタ（FMCNT）をデクリメントし，アンダフローが発生しなければ，タイミング・データの更新を行います。更新対象は，現在マクロ・サービスで使用していないバッファRAMとなります。ここで，ASFLAG＝1の場合はデクリメント，ASFLAG＝0の場合はインクリメントします。このあと，バッファRAMの切り替え，コントロール・エリア内のタイミング用のマクロ・サービス・ポインタの更新を行いますが，マクロ・サービス完了割り込みが発生する付近と，3/3周期以外の状態を避けて行っています。

図6－22　周波数移行時のデータの切り替え（低周波側へ移行中）

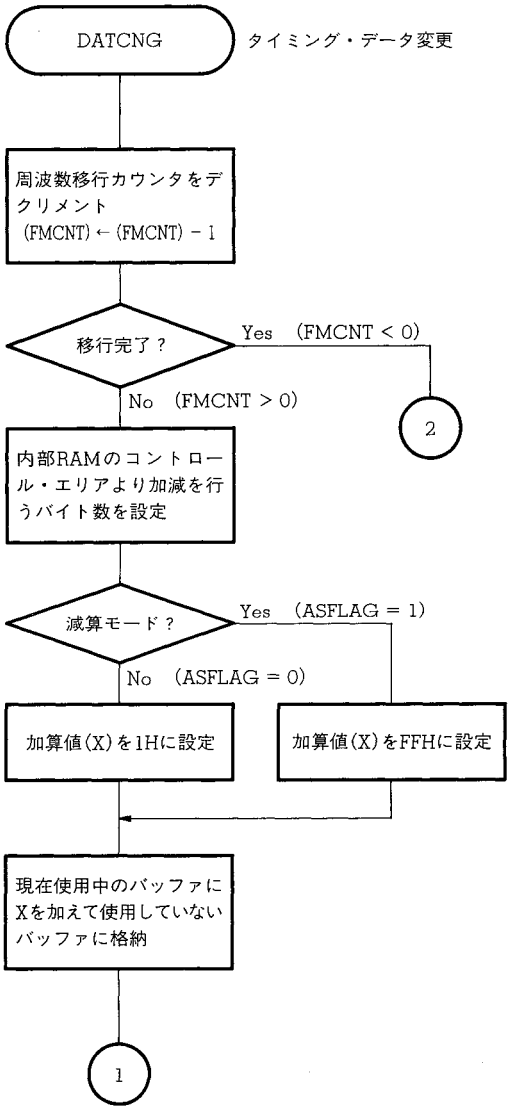


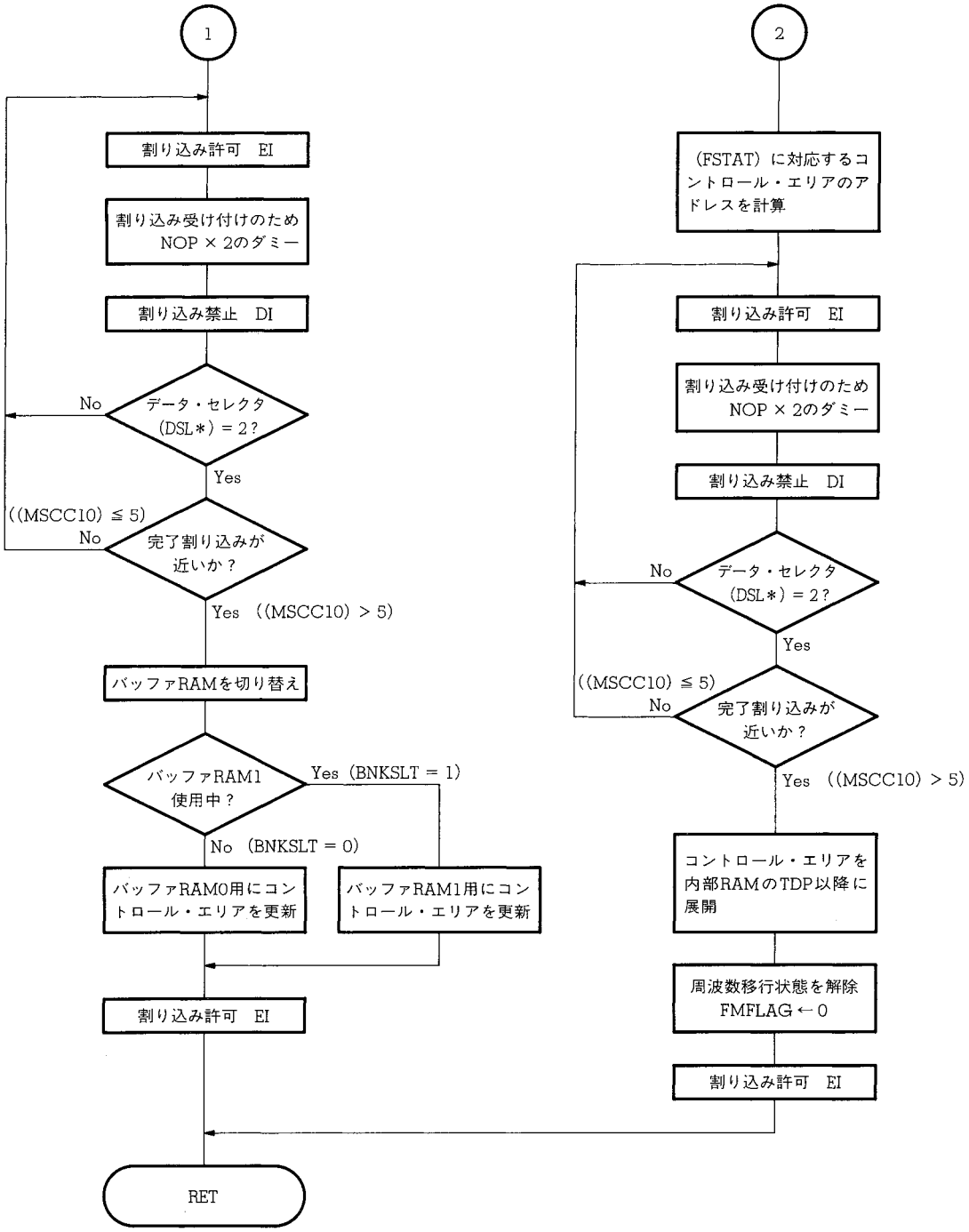
：マクロ・サービスで使用中のバッファ

(ii) 移行完了処理

周波数移行カウンタ（FMCNT）をデクリメントし，アンダフローが発生した場合は，移行完了状態となります。この場合は，タイミング・データをROM上のものに切り替えるため，コントロール・エリアの更新と周波数移行状態のクリア（FMFLAG←0）を行います。なお，マクロ・サービス完了割り込みが発生する付近と，3/3周期以外の状態を避けて行っています。

(b) フロー・チャート


DATCNG
タイミング・データ変更
周波数移行カウンタをデクリメント
(FMCNT) ← (FMCNT) − 1
移行完了？
Yes　(FMCNT < 0)
No　(FMCNT > 0)
2
内部RAMのコントロール・エリアより加減を行うバイト数を設定
減算モード？
Yes　(ASFLAG = 1)
No　(ASFLAG = 0)
加算値(X)を1Hに設定
加算値(X)をFFHに設定
現在使用中のバッファにXを加えて使用していないバッファに格納
1


1
割り込み許可　EI
割り込み受け付けのため
NOP × 2のダミー
割り込み禁止　DI
No
データ・セレクタ
(DSL*) = 2?
Yes
((MSCC10) ≦ 5)
No
完了割り込みが
近いか？
Yes　((MSCC10) > 5)
バッファRAMを切り替え
バッファRAM1
使用中？
Yes (BNKSLT = 1)
No　(BNKSLT = 0)
バッファRAM0用にコン
トロール・エリアを更新
バッファRAM1用にコン
トロール・エリアを更新
割り込み許可　EI
RET
2
(FSTAT) に対応するコ
ントロール・エリアのア
ドレスを計算
割り込み許可　EI
割り込み受け付けのため
NOP × 2のダミー
割り込み禁止　DI
No
データ・セレクタ
(DSL*) = 2?
Yes
((MSCC10) ≦ 5)
No
完了割り込みが
近いか？
Yes　((MSCC10) > 5)
コントロール・エリアを
内部RAMのTDP以降に
展開
周波数移行状態を解除
FMFLAG ← 0
割り込み許可　EI

(9) マクロ・サービス再起動処理

サブルーチン名：INTC10, INTC11

(a) 処理内容

INTC10, INTC11のマクロ・サービス完了割り込み処理です。ここでは，RAM上のコントロール・エリアからマクロ・サービス・チャネルに再設定する値を読み出して，書き込んでいます。両者の完了割り込みとも同一の処理を行っています。ここでの主な処理を次に示します。

- タイマ用マクロ・サービス・ポインタ更新
- データ用マクロ・サービス・ポインタ更新
- 割り込みサービス・モードの再設定
- データ選択カウンタ更新

(i) タイマ用マクロ・サービス・ポインタ更新

タイマ用のマクロ・サービス・ポインタへの設定値をコントロール・エリアより読み出して，書き込みます。

(ii) データ用マクロ・サービス・ポインタ更新

出力データは，1周期分用意されているため，データ選択カウンタ（DSLP, DSLM）が2の場合のみ，データ用のマクロ・サービス・ポインタへの設定値をコントロール・エリアより読み出して，書き込みます。

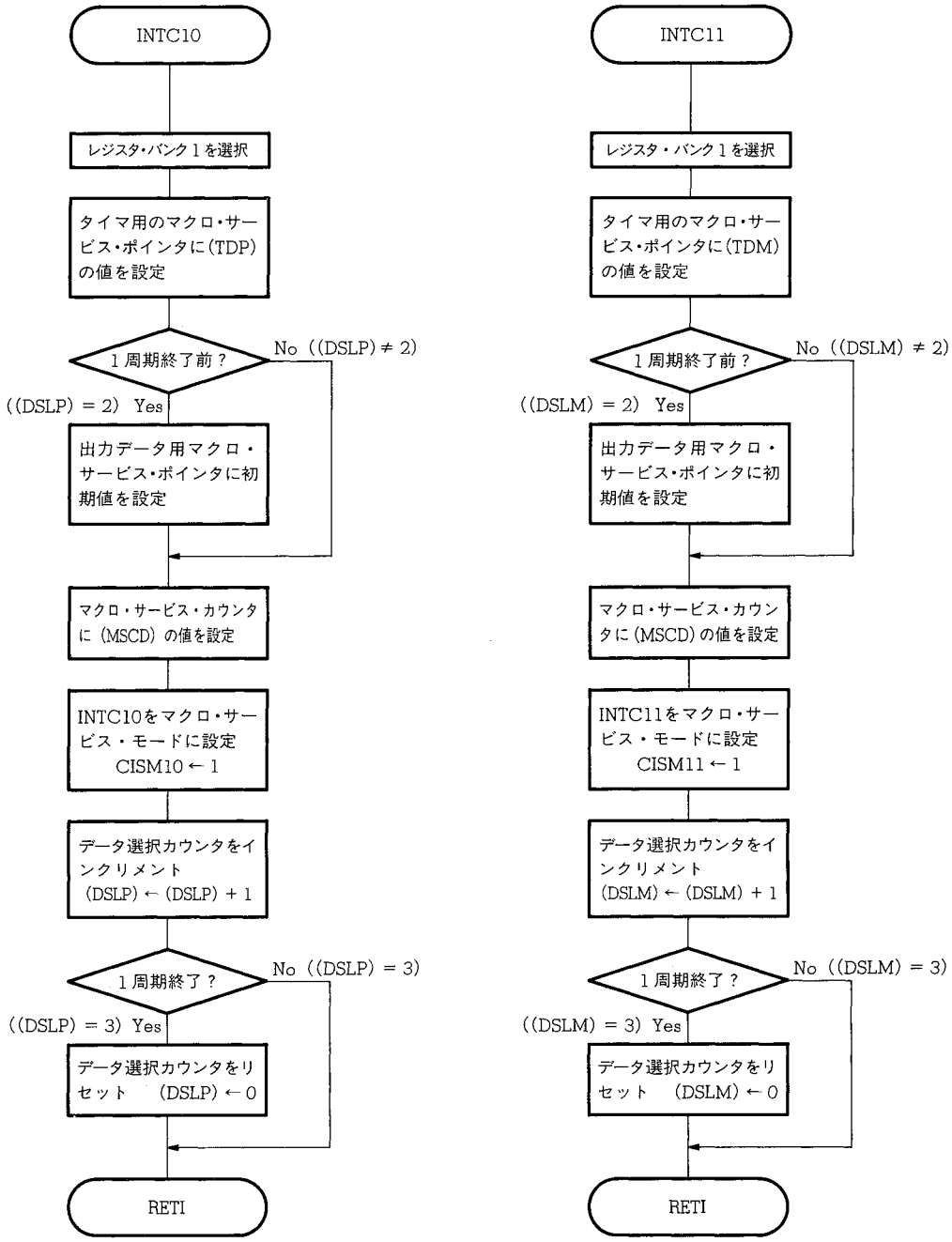
(iii) 割り込みサービス・モードの再設定

割り込みサービス・モードが，ベクタ割り込みモードとなっているため，マクロ・サービス・モードに再設定します。

(iv) データ選択カウンタ更新

データ選択カウンタ(DSLP, DSLM)をインクリメントし，3になった場合は，0に再設定します。

(b) フロー・チャート


INTC10
レジスタ・バンク1を選択
タイマ用のマクロ・サービス・ポインタに(TDP)の値を設定
1周期終了前？
No ((DSLP) ≠ 2)
((DSLP) = 2) Yes
出力データ用マクロ・サービス・ポインタに初期値を設定
マクロ・サービス・カウンタに(MSCD)の値を設定
INTC10をマクロ・サービス・モードに設定
CISM10 ← 1
データ選択カウンタをインクリメント
(DSLP) ← (DSLP) + 1
1周期終了？
No ((DSLP) = 3)
((DSLP) = 3) Yes
データ選択カウンタをリセット (DSLP) ← 0
RETI
INTC11
レジスタ・バンク1を選択
タイマ用のマクロ・サービス・ポインタに(TDM)の値を設定
1周期終了前？
No ((DSLM) ≠ 2)
((DSLM) = 2) Yes
出力データ用マクロ・サービス・ポインタに初期値を設定
マクロ・サービス・カウンタに(MSCD)の値を設定
INTC11をマクロ・サービス・モードに設定
CISM11 ← 1
データ選択カウンタをインクリメント
(DSLM) ← (DSLM) + 1
1周期終了？
No ((DSLM) = 3)
((DSLM) = 3) Yes
データ選択カウンタをリセット (DSLM) ← 0
RETI

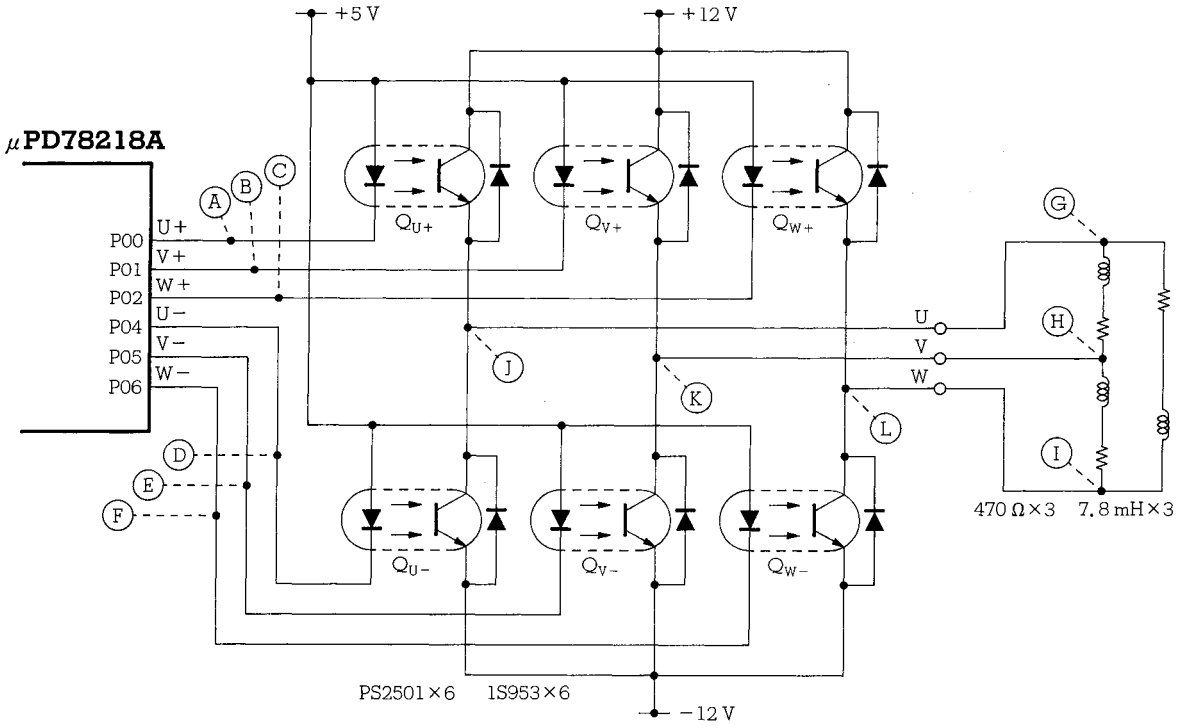
6.6　出力波形例

この応用例の出力波形例を示します。

6.6.1　ターゲット回路

ここで取り上げる出力波形例は，図6－23のターゲット回路で測定したものです。

図6－23　ターゲット回路図

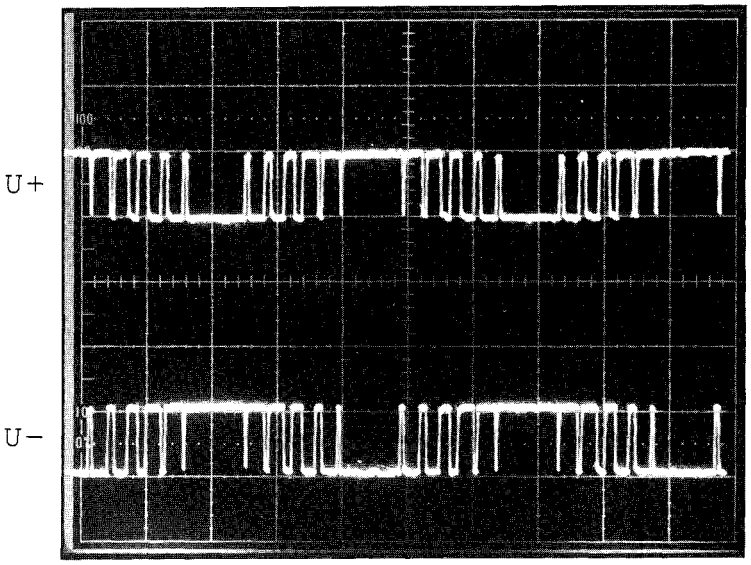


6.6.2　リアルタイム出力ポート波形

各周波数のリアルタイム出力ポート波形を示します。インバータ出力周波数は208.33 Hz，65.10 Hz，32.55 Hzで，測定点は，Ⓐ-GND，Ⓓ-GNDです。

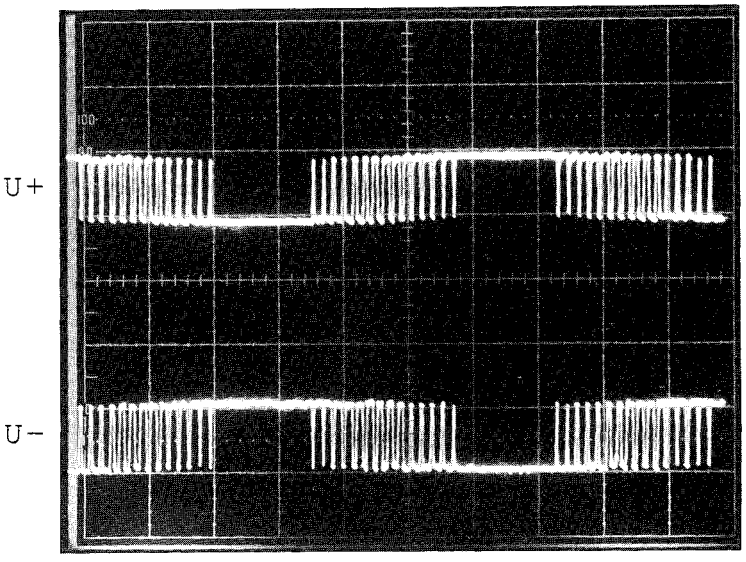
図6－24　リアルタイム出力ポート波形（1/2）

208.33 Hz



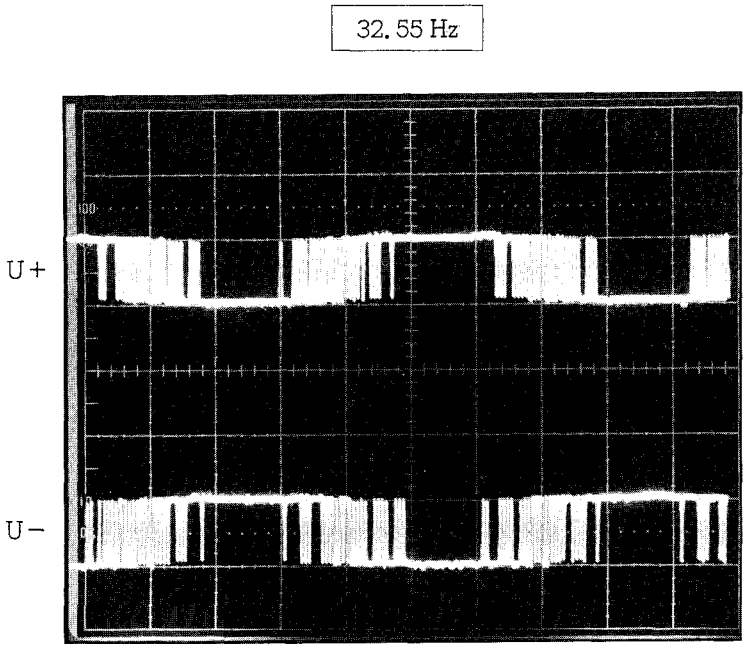
1 ms/div

65.10 Hz



2 ms/div

図6－24　リアルタイム出力ポート波形（2/2）

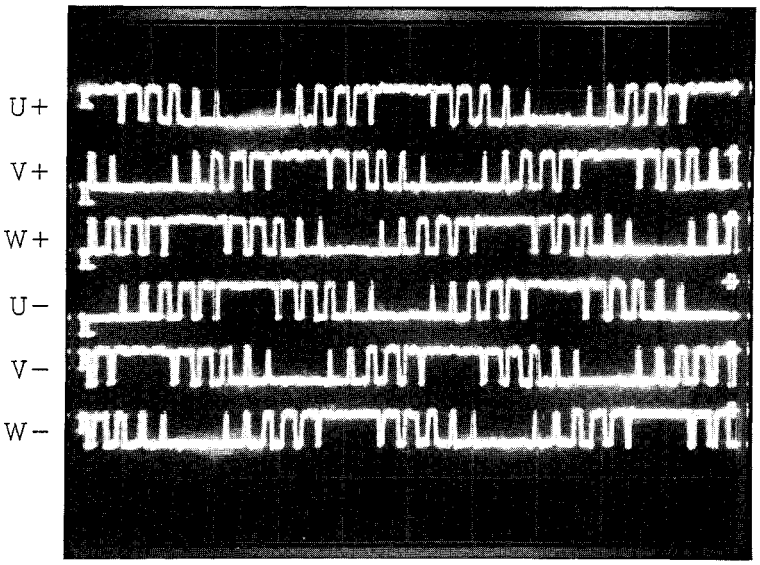


6.6.3　各相のリアルタイム出力ポート波形

この応用例では，各相の位相差は240°です。インバータ出力周波数は208.33 Hzと65.10 Hzです。測定点は，Ⓐ-GND，Ⓑ-GND，Ⓒ-GND，Ⓓ-GND，Ⓔ-GND，Ⓕ-GNDです。

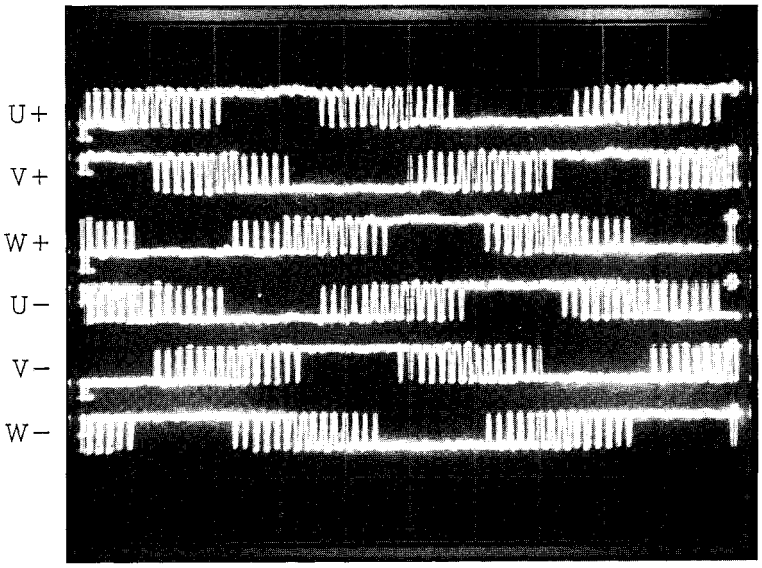
図6－25　各相のリアルタイム出力ポート波形

208.33 Hz



1 ms/div

65.10 Hz

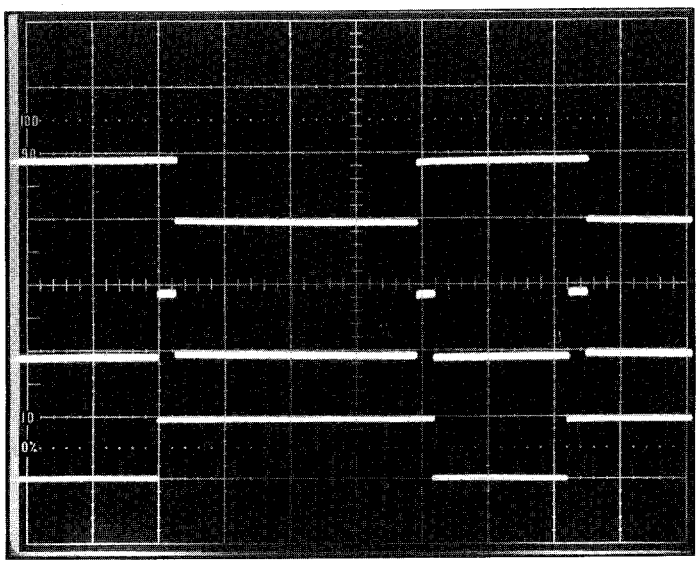


2 ms/div

6.6.4　デッド・タイム波形

この応用例では，10.67 μsのデッド・タイムを設けています。インバータ出力周波数が65.10 Hzのときの，ある点での拡大波形を示します。測定点はⒶ-GND，Ⓓ-GNDです。

図6－26　デッド・タイム波形

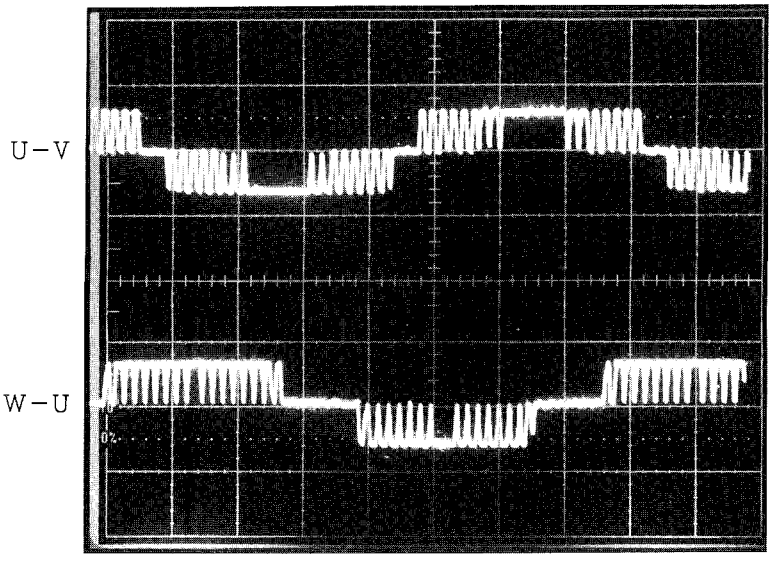


50 μs/div

6.6.5　無負荷時の相間波形

負荷を接続しない場合の，インバータ出力の相間電圧波形を示します。インバータ出力周波数は65.10 Hzで，測定点はⒼ-Ⓗ，Ⓘ-Ⓖです。

図6－27　無負荷時の相間波形



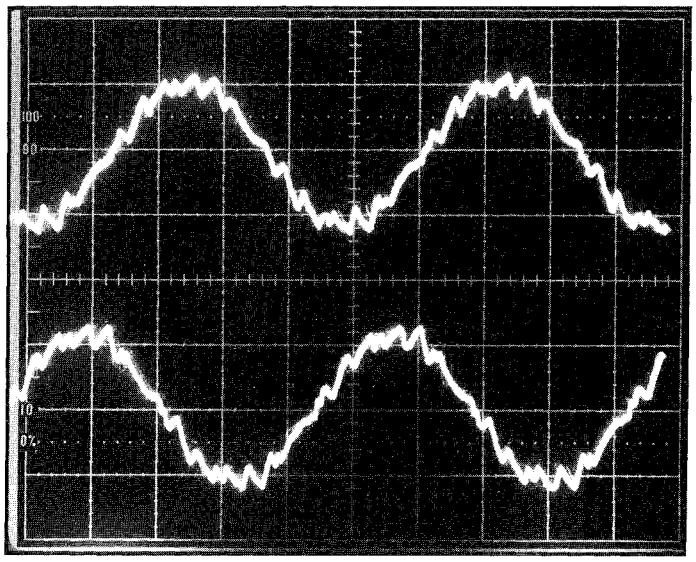
2 ms/div

6.6.6　負荷接続時の相間波形

この応用例では，疑似負荷として，7.8 mHのインダクタと470 Ωの抵抗をシリーズ接続し，これをΔ結線として三相負荷に使用しました。この負荷を接続した場合の，インバータ出力の相間電圧波形を示します。インバータ出力周波数は208.33 Hzと65.10 Hzです。測定点はⒼ-Ⓗ，Ⓘ-Ⓖです。

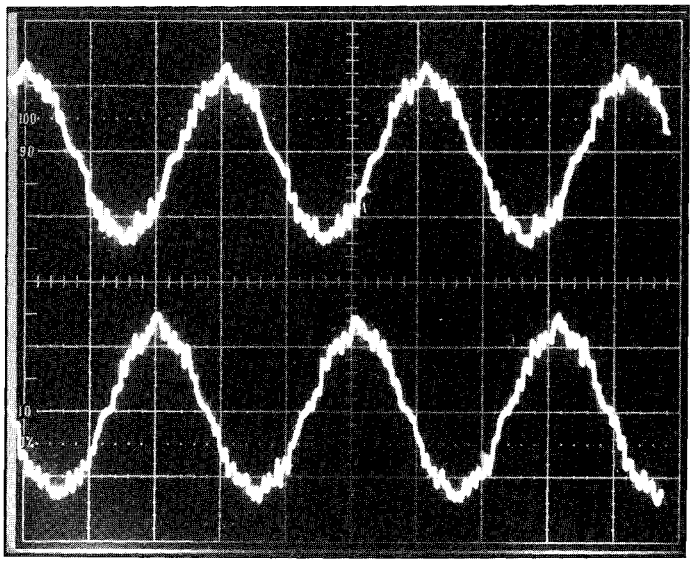
図6－28　負荷波形

208.33 Hz



1 ms/div

65.10 Hz



5 ms/div

6.7　プログラム・リスト

6.7.1　インバータ制御プログラム

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードします。

```
$       IC(INTMS.DEF)
;************************************************
;*      78K/II Seriese Inverter Control         *
;*              for uPD78218A Seriese           *
;*                  uPD78234  Seriese           *
;************************************************

        NAME    INVERTER

        PUBLIC  INVINI,INTC01,CHKDAT            ; process
        PUBLIC  FAT,FSTAT                       ; value
        PUBLIC  ONFLAG,OFFLAG                   ; bit
        PUBLIC  FMFLAG,ASFLAG,STUPOK,STPOK

        EXTRN   CNT33,CNT43,CNT65,CNT87         ; control data address
        EXTRN   CNT130,CNT174,CNT208
```

```
;************************************************
;*      DEFINE SYMBOL                           *
;************************************************
;
;       --- DATA DEFINE ---
;
SFTPRE  EQU     40                      ; for 2sec timer
ALLOFF  EQU     0FFH                    ; P0 inactive level
C01ITVL EQU     37500-1                 ; INTC01 interval value
CNTLEN  EQU     7                       ; control area transmit length
ADJTM   EQU     5+1                     ; adjust timing for MSC
DLYTM   EQU     5                       ; CR10 , CR11 delay time
;
;       --- DEFINE WORK FLAG ---
;
INV_B   BSEG
FMFLAG  DBIT                            ; frequency move flag
ASFLAG  DBIT                            ; add/sub flag for timing data
STUPOK  DBIT                            ; complete of start-up flag
STPOK   DBIT                            ; complete of stop flag
ONFLAG  DBIT                            ; on request flag
OFFLAG  DBIT                            ; off request flag
BNKSLT  DBIT                            ; select buffer bank flag

INV_DP  DSEG    AT      0FC00H
BUFP0:  DS      128                     ; moving frequency buffer [plus0]
BUFM0:  DS      128                     ; moving frequency buffer [minus0]
BUFP1:  DS      128                     ; moving frequency buffer [plus1]
BUFM1:  DS      128                     ; moving frequency buffer [minus1]


INV_DI  DSEG    SADDR
TDP:    DS      2               ; timing data <+> start address
TDM:    DS      2               ; timing data <-> start address
DT:     DS      2               ; output data start address
MSCD:   DS      1               ; macro service counter temporary

FSTAT:  DS      1               ; now frequency
FAT:    DS      1               ; target frequency
FMCNT:  DS      1               ; frequency move counter
SECCNT: DS      1               ; 2 second counter (INTC01 counter)
DSLP:   DS      1               ; macro service sub-counter (for INTC10)
DSLM:   DS      1               ; macro service sub-counter (for INTC11)
DELAY:  DS      1               ; delay (wait) counter
```

```
        DSEG_C  C10,'SADDR'     ; INTC10 macro service channel (type C )
        CMCW10                  ;        macro service control word
        DSEG_C  C11,'SADDR'     ; INTC11 macro service channel (type C )
        CMCW11                  ;        macro service control word

;************************************************
;*      DEFINE VECTOR TRABLE                    *
;************************************************

        NMIVENT                         ; NMI    vector
        CVENT10                         ; INTC10 vector
        CVENT11                         ; INTC11 vector
```

```
;************************************************
;*      INVERTRT INITIALIZE                     *
;*              input  : nothing (from MAIN)    *
;*              output : FMFLAG <- 0            *
;*                       STUPOK <- 0            *
;*                       STPOK <- 0             *
;*                       OMFLAG <- 0            *
;*                       OFFLAG <- 0            *
;*                       FSTAT <- 0             *
;*                       FAT <- 1               *
;*                       SECCNT <- 40           *
;*                       DELAY<- INIDLY (4)     *
;*              break reg. : AX,B,DE,HL         *
;************************************************

INV_C   CSEG

INIDLY  EQU     4                       ; (4+1)*2=10sec delay
RSTPRE  EQU     1                       ; first software pre-scaler data

INVINI:
        SET1    STUPOK                  ; set power on wait (10sec)
        CLR1    STPOK
        MOV     SECCNT,#RSTPRE

        MOV     FSTAT,#0                ; off status
        MOV     FAT,#1
        CLR1    ONFLAG
        CLR1    OFFLAG
        CLR1    FMFLAG

        MOV     P0,#ALLOFF              ; initialize port
        MOV     PM0,#00H
;
;       --- intialize timer / counter INTC01,INTC10,INTC11 ---
;
        MOV     CRC0,#00011000B         ; enable clear TM0 by CR01
        MOVW    CR01,#C01ITVL           ; INTC01 : 50msec
        MOV     CRC1,#00000000B         ; disable clear TM1
        MOV     CR10,#0
        MOV     CR11,#0

        MOV     MSMC10,#11101010B       ; initialize macro service
        MOV     CHPC10,#LOW CHAC10
        MOV     MSMC11,#11101011B
        MOV     CHPC11,#LOW CHAC11

        CLR1    CPR10                   ; initialize interrupt
        CLR1    CPR11
        SET1    CISM10
        SET1    CISM11
        CLR1    CMK10
        CLR1    CMK11

        MOV     DELAY,#INIDLY
        RET
```

```
;***********************************************
;*      INVERTER START                         *
;*              input  : nothing (from INTC01) *
;*              output : run 33[Hz]            *
;*                       STUPOK <- 0           *
;*                       DELAY <- STTDLY (4)   *
;*                       FSTAT <- 1            *
;*                       DSLP,DSLM <- 0        *
;*              break reg. : AX,B,DE,HL        *
;***********************************************

STTDLY  EQU     4                       ; (4+1)*2=10sec delay

INVSTT:
        DI                              ; interrupt disable
        CLR1    ONFLAG
        CLR1    STUPOK
        MOV     FSTAT,#1

        MOVW    HL,#CNT33               ; open 33Hz control data
        MOVW    DE,#TDP
        MOV     B,#CNTLEN
OPEN_33:
        MOV     A,[HL+]
        MOV     [DE+],A
        DBNZ    B,$OPEN_33

        MOVW    AX,TDP                  ; set macro sevice channel
        MOVW    MPTC10P,AX
        MOVW    AX,TDM
        MOVW    MPTC11P,AX
        MOVW    AX,DT
        MOVW    MPDC10P,AX
        MOVW    MPDC11P,AX
        MOV     MSCC10,MSCD
        MOV     MSCC11,MSCD

        MOV     DSLP,#0                 ; set data select counter
        MOV     DSLM,#0

        MOVW    HL,AX                   ; set first data
        MOV     A,MSCD
        MOV     X,#3
        MULU    X
        DECW    AX
        ADDW    AX,HL
        MOVW    HL,AX
        MOV     A,[HL]
        MOV     P0L,A
        MOV     P0H,A

        MOV     RTPC,#00010001B         ; real-time output port start
        MOV     CR10,#0
        MOV     CR11,#DLYTM
        OR      TMC1,#00001000B         ; start TM1

        MOV     DELAY,#STTDLY           ; wait 10sec
        EI
        RET
```

```
;************************************************
;*      INVERTER STOP                           *
;*              input  : nothing (from INTC01)  *
;*              output : inverter stop          *
;*                       FAT <- 1               *
;*                       FSTAT <- 0             *
;*                       FMFLAG <- 0            *
;*                       STPOK <-0              *
;*                       DELAY <- STPDLY (5)    *
;*              break reg. : nothing            *
;************************************************

STPDLY  EQU     5                       ; (5+1)*2=12sec delay

INVSTP:
        DI                              ; interrupt disable
        CLR1    OFFLAG

        MOV     RTPC,#00000000B
        MOV     P0,#ALLOFF
        AND     TMC1,#11100000B         ; TM1 stop
        AND     IF0L,#00111111B         ; clear CIF10,CIF11
        MOV     FAT,#1
        MOV     FSTAT,#0
        CLR1    FMFLAG
        CLR1    STPOK
        SET1    STUPOK

        MOV     DELAY,#STPDLY           ; 10-12sec wait
        EI
        RET


;************************************************
;*      EMEGENCY INVERTER STOP                  *
;*              input  : NMI input              *
;*              output : inverter stop          *
;*                       lock program           *
;*              break reg. : nothing            *
;************************************************

NMI:
        MOV     RTPC,#00000000B
        MOV     P0,#ALLOFF
        AND     TMC1,#11100000B         ; TM1 stop
        MOVW    MK0,#0FFFFH             ; mask all interrupt
        BR      $$
```


218A

```
;***********************************************
;*      SET FREQUENCY MOVE                     *
;*              input  : nothing (from INTC01) *
;*              output : make next frequency   *
;*                       FSTAT <- +/- (1-7)    *
;*                       FMFLAG <- 1           *
;*                       ASFLAG <- 1/0         *
;*                       TDP <- BUFP0          *
;*                       TDM <- BUFM0          *
;*                       BNKSLT <- 0           *
;*                       FMCNT,DT,MSCD,PMD <- ? *
;*              break reg. : AX,BC,DE,HL       *
;***********************************************

DATMOV:
        CMP     FSTAT,FAT
        BNC     $DN_FREQ
;
;       --- FAT > FSTAT ---
;
UP_FREQ:
        MOV     A,FSTAT
        INC     A
        CALL    !MK_ADCNT               ; make address & counter
        PUSH    DE
        PUSH    BC

        MOVW    DE,#BUFP0               ; OPEN +
DMV_L1:
        MOV     A,[HL+]
        ADD     A,C
        MOV     [DE+],A
        DBNZ    B,$DMV_L1

        POP     BC                      ; OPEN -
        POP     DE
        MOVW    HL,#BUFM0
DMV_L2:
        MOV     A,[DE+]
        ADD     A,C
        MOV     [HL+],A
        DBNZ    B,$DMV_L2

        SET1    ASFLAG
        MOV     A,C
        MOV     FMCNT,A
        INC     FSTAT

        MOV     A,FSTAT                 ; set open frequency
        DEC     A
        BR      OPEN_FRQ
```

```
;
;       --- FAT < FSTAT ---
;
DN_FREQ:
        MOV     A, FSTAT
        CALL    !MK_ADCNT               ; make address & counter
        PUSH    DE
        PUSH    BC

        MOVW    DE, #BUFP0              ; OPEN +
DMV_L3:
        MOV     A, [HL+]
        MOV     [DE+], A
        DBNZ    B, $DMV_L3

        POP     BC                      ; OPEN -
        POP     DE
        MOVW    HL, #BUFM0
DMV_L4:
        MOV     A, [DE+]
        MOV     [HL+], A
        DBNZ    B, $DMV_L4

        CLR1    ASFLAG
        MOV     A, C
        MOV     FMCNT, A
        DEC     FSTAT

        MOV     A, FSTAT                ; set open frequency
OPEN_FRQ:
        MOV     B, A                    ; top of control data address
        ADD     B, B
        MOV     A, CNT_TBL[B]
        MOV     L, A
        MOV     A, CNT_TBL+1[B]
        MOV     H, A
        MOVW    DE, #TDP
        MOV     B, #CNTLEN
TIMEWAT1:                               ; adjust timing
        EI
        NOP                             ; enable interrupt cycle
        NOP
        DI
        CMP     DSLP, #2
        BNE     $TIMEWAT1
        CMP     DSLM, #2
        BNE     $TIMEWAT1
        CMP     MSCC10, #ADJTM
        BC      $TIMEWAT1
OPEN_L1:                                ; data open
        MOV     A, [HL+]
        MOV     [DE+], A
        DBNZ    B, $OPEN_L1

        CLR1    BNKSLT                  ; use buffer bank 0
        MOVW    TDP, #BUFP0
        MOVW    TDM, #BUFM0
        SET1    FMFLAG
        EI
        RET
```

```
;
;       --- make address & counter sub-routine ---
;
;               input  : A  <- frequency
;               output : DE <- timing data - address
;                        HL <- timing data + address
;                        B  <- data length
;                        C  <- add/sub counter
;               break.reg : AX

MK_ADCNT:
        DEC     A                       ; top of control data address
        MOV     B,A
        ADD     B,B
        MOV     A,CNT_TBL[B]
        MOV     E,A
        MOV     A,CNT_TBL+1[B]
        MOV     D,A

        MOV     A,[DE+6]
        MOV     B,A                     ; B : data lenfth
        MOV     A,[DE+7]
        MOV     C,A                     ; C : add/sub counter
        MOVW    AX,[DE]
        MOVW    HL,AX                   ; HL : timing data + address
        INCW    DE
        INCW    DE
        MOVW    AX,[DE]
        MOVW    DE,AX                   ; DE : timing data - address
        RET
```

```
;***********************************************
;*      INTC01 TIMER INTERRUPT (FLAG CHECK)     *
;*              input  : nothing                *
;*                       (from MAIN INTC01 50ms)*
;*              output : system control         *
;*                       STUPOK,STPOK <- ?      *
;*                       OFFLAG <- ?            *
;*                       SECCNT <- SECCNT-1     *
;*              break reg. : AX,BC,DE,HL        *
;***********************************************

INTC01:
        BF      OFFLAG,$NO_OFFRQ        ; check OFF request
        CMP     FSTAT,#0                ; check OFF status
        BE      $NO_OFFRQ
        CALL    !INVSTP                 ; stop
        BR      EXITC01
NO_OFFRQ:
        DBNZ    SECCNT,$EXITC01         ; 2sec timer
        MOV     SECCNT,#SFTPRE
        CMP     DELAY,#0                ; check delay time
        BNE     $DELAY_NZ
        BF      STUPOK,$CPLT_STT        ; check start complete
        BF      STPOK,$CPLT_STP         ; check stop complete
        BT      FMFLAG,$CHNG_RQ         ; check frequency move status
        CMP     FSTAT,#0
        BNE     $F_CHECK
        BT      ONFLAG,$ON_RQ
        BR      EXITC01

DELAY_NZ:
        DEC     DELAY
        BR      EXITC01
CPLT_STT:
        SET1    STUPOK                  ; start complete
        BR      EXITC01
CPLT_STP:
        SET1    STPOK                   ; stop complete
        BR      EXITC01
CHNG_RQ:
        CALL    !DATCNG                 ; data change
        BR      EXITC01
F_CHECK:
        CMP     FAT,FSTAT               ; check target frequency
        BE      $EXITC01
        CALL    !DATMOV                 ; data move
        BR      EXITC01
ON_RQ:
        CALL    !INVSTT                 ; start
EXITC01:
        RET
```


218A

234

```
;*************************************************
;*      DATA OPEN, CHANGE                        *
;*              input  : nothing (from INTC01)   *
;*              output : change frequency data   *
;*                       FMCNT <- FMCNT-1        *
;*                       FMFLAG <- ?             *
;*                       BNKSLT <- ?             *
;*                       make new timing data    *
;*              break reg. : AX, BC, DE, HL      *
;*************************************************

DATCNG:
        SUB     FMCNT, #1               ; chek move end
        BC      $END_MOVE

        MOV     A, MSCD                 ; add/sub counter
        MOV     B, A
        MOV     C, A
        MOV     X, #0FFH                ; -
        BT      ASFLAG, $CALC_ADR
        MOV     X, #1H                  ; +
CALC_ADR:
        BT      BNKSLT, $MVP1TO0
        MOVW    HL, #BUFP0
        MOVW    DE, #BUFP1
        BR      CNGCALCP
MVP1TO0:
        MOVW    HL, #BUFP1
        MOVW    DE, #BUFP0
CNGCALCP:
        MOV     A, [HL+]
        ADD     A, X
        MOV     [DE+], A
        DBNZ    B, $CNGCALCP

        BT      BNKSLT, $MVM1TO0
        MOVW    HL, #BUFM0
        MOVW    DE, #BUFM1
        BR      CNGCALCM
MVM1TO0:
        MOVW    HL, #BUFM1
        MOVW    DE, #BUFM0
CNGCALCM:
        MOV     A, [HL+]
        ADD     A, X
        MOV     [DE+], A
        DBNZ    C, $CNGCALCM

TIMEWAT2:                               ; adjust timing
        EI
        NOP                             ; enable interrupt cycle
        NOP
        DI
        CMP     DSLP, #2
        BNE     $TIMEWAT2
        CMP     DSLM, #2
        BNE     $TIMEWAT2
        CMP     MSCC10, #ADJTM
        BC      $TIMEWAT2
```

```
        NOT1    BNKSLT                  ; change buffer bank
        BT      BNKSLT,$BNKSET
        MOVW    TDP,#BUFP0
        MOVW    TDM,#BUFM0
        BR      RETDATCG
BNKSET:
        MOVW    TDP,#BUFP1
        MOVW    TDM,#BUFM1
RETDATCG:
        EI
        RET

END_MOVE:                               ; end of moving
        MOV     A,FSTAT
        DEC     A
        MOV     B,A                     ; top of control data address
        ADD     B,B
        MOV     A,CNT_TBL[B]
        MOV     L,A
        MOV     A,CNT_TBL+1[B]
        MOV     H,A
        MOVW    DE,#TDP
        MOV     B,#CNTLEN
TIMEWAT3:                               ; adjust timing
        EI
        NOP                             ; enable interrupt cycle
        NOP
        DI
        CMP     DSLP,#2
        BNE     $TIMEWAT3
        CMP     DSLM,#2
        BNE     $TIMEWAT3
        CMP     MSCC10,#ADJTM
        BC      $TIMEWAT3
OPEN_L2:                                ; data open
        MOV     A,[HL+]
        MOV     [DE+],A
        DBNZ    B,$OPEN_L2
        CLR1    FMFLAG
        EI
        RET
```

```
;************************************************
;*      COMPLETE INTC10, INTC11 MACRO SERVICE    *
;*              input  : complete INTC10, INTC11 *
;*                                macro service  *
;*              output : DSL* <- DSL*+1          *
;*                       INTC10, INTC11          *
;*                        macro service channel *
;*              break reg. : AX (RB1)            *
;************************************************

INTC10:
        SEL     RB1
        MOVW    AX, TDP                 ; INTC10 MPT
        MOVW    MPTC10P, AX
        CMP     DSLP, #2                ; INTC10 MPD
        BNE     $C10_J1
        MOVW    AX, DT
        MOVW    MPDC10P, AX
C10_J1:
        MOV     MSCC10, MSCD            ; INTC10 MSC
        SET1    CISM10
        INC     DSLP
        CMP     DSLP, #3                ; check 1 cycle
        BNE     $C10_J2
        MOV     DSLP, #0
C10_J2:
        RETI
;
INTC11:
        SEL     RB1
        MOVW    AX, TDM                 ; INTC11 MPT
        MOVW    MPTC11P, AX
        CMP     DSLM, #2                ; INTC11 MPD
        BNE     $C11_J1
        MOVW    AX, DT
        MOVW    MPDC11P, AX
C11_J1:
        MOV     MSCC11, MSCD            ; INTC11 MSC
        SET1    CISM11
        INC     DSLM
        CMP     DSLM, #3                ; check 1 cycle
        BNE     $C11_J2
        MOV     DSLM, #0
C11_J2:
        RETI
```

```
;************************************************
;*      CHECK TIMING DATA                       *
;*              input  : nothing (from INVINI)  *
;*              output : ok  : CY <- 0          *
;*                       bad : CY <- 1          *
;*              break reg. : AX,BC,DE,HL        *
;************************************************

FRQ_NUM EQU     7                       ; numbers of frequency

CHKDAT:
        MOV     B,#FRQ_NUM
CKD_L1:
        MOVW    HL,#CNT_TBL             ; HL : top address of frequency table
        MOV     A,B
        DEC     A
        ADD     A,A
        ADD     L,A
        MOV     A,#0
        ADDC    H,A

        MOVW    AX,[HL]                 ; HL : control data top address
        MOVW    HL,AX

        MOVW    AX,[HL]                 ; timing data top address
        MOVW    DE,AX
        MOV     A,[HL+6]                ; data length
        MOV     C,A
        ADD     C,C
        MOV     A,#DLYTM-1
CKD_L2:
        CMP     A,[DE+]                 ; check data
        NOT1    CY
        BC      $CKD_J1
        DBNZ    C,$CKD_L2
        DBNZ    B,$CKD_L1
        CLR1    CY                      ; no error
CKD_J1:
        RET

;************************************************
;*      CONTROL DATA ADDRESS TABLE              *
;************************************************

CNT_TBL:
        DW      CNT33
        DW      CNT43
        DW      CNT65
        DW      CNT87
        DW      CNT130
        DW      CNT174
        DW      CNT208

        END
```

6.7.2 PWMデータ

```
;************************************************
;*      CONTROL DATA AREA                       *
;************************************************

CNT_C   CSEG

        PUBLIC  CNT208
CNT208:                 ; 208Hz
        DW      TMP208                  ; timing data +
        DW      TMM208                  ; timing data -
        DW      DT208                   ; output data
        DB      18                      ; data length (1/3 cycle)
        DB      6                       ; add/sub counter

        PUBLIC  CNT174
CNT174:                 ; 174Hz
        DW      TMP174                  ; timing data +
        DW      TMM174                  ; timing data -
        DW      DT174                   ; output data
        DB      20                      ; data length (1/3 cycle)
        DB      11                      ; add/sub counter

        PUBLIC  CNT130
CNT130:                 ; 130Hz
        DW      TMP130                  ; timing data +
        DW      TMM130                  ; timing data -
        DW      DT130                   ; output data
        DB      26                      ; data length (1/3 cycle)
        DB      18                      ; add/sub counter

        PUBLIC  CNT87
CNT87:                  ; 87Hz
        DW      TMP87                   ; timing data +
        DW      TMM87                   ; timing data -
        DW      DT87                    ; output data
        DB      36                      ; data length (1/3 cycle)
        DB      13                      ; add/sub counter

        PUBLIC  CNT65
CNT65:                  ; 65Hz
        DW      TMP65                   ; timing data +
        DW      TMM65                   ; timing data -
        DW      DT65                    ; output data
        DB      50                      ; data length (1/3 cycle)
        DB      19                      ; add/sub counter

        PUBLIC  CNT43
CNT43:                  ; 43Hz
        DW      TMP43                   ; timing data +
        DW      TMM43                   ; timing data -
        DW      DT43                    ; output data
        DB      74                      ; data length (1/3 cycle)
        DB      11                      ; add/sub counter
```

```
        PUBLIC  CNT33
CNT33:                  ; 33Hz
        DW      TMP33                   ; timing data +
        DW      TMM33                   ; timing data -
        DW      DT33                    ; output data
        DB      98                      ; data length (1/3 cycle)
        DB      0                       ; add/sub counter


;************************************************
;*      INVERTER OUTPUT DATA                    *
;************************************************

;       FREQUENCY    = 208[Hz]
;       DATA LENGTH = 18

F208TM  CSEG
TMP208: DB      019H,011H,012H,04AH,023H,04DH,012H,011H,018H,018H
        DB      007H,012H,054H,019H,057H,012H,007H,019H
TMM208: DB      019H,007H,012H,054H,019H,057H,012H,007H,018H,018H
        DB      011H,012H,04AH,023H,04DH,012H,011H,019H

F208DT  CSEG
DT208:  DB      08FH,0ADH,0BCH,08FH,0BCH,09EH,08FH,09EH,0BCH,0F8H
        DB      0BCH,09EH,0F8H,09EH,0DAH,0F8H,0DAH,09EH
        DB      08FH,09EH,0DAH,08FH,0DAH,0CBH,08FH,0CBH,0DAH,0F8H
        DB      0DAH,0CBH,0F8H,0CBH,0E9H,0F8H,0E9H,0CBH
        DB      08FH,0CBH,0E9H,08FH,0E9H,0ADH,08FH,0ADH,0E9H,0F8H
        DB      0E9H,0ADH,0F8H,0ADH,0BCH,0F8H,0BCH,0ADH



;       FREQUENCY    = 174[Hz]
;       DATA LENGTH = 20

F174TM  CSEG
TMP174: DB      019H,010H,014H,047H,025H,051H,00BH,017H,010H,03CH
        DB      046H,010H,00DH,00BH,058H,017H,058H,014H,006H,019H
TMM174: DB      019H,006H,014H,051H,01BH,05BH,00BH,00DH,010H,046H
        DB      03CH,010H,017H,00BH,04EH,021H,04EH,014H,010H,019H

F174DT  CSEG
DT174:  DB      08FH,0ADH,0BCH,08FH,0BCH,09EH,08FH,09EH,0BCH,09EH
        DB      0BCH,0F8H,0BCH,09EH,0F8H,09EH,0DAH,0F8H,0DAH,09EH
        DB      08FH,09EH,0DAH,08FH,0DAH,0CBH,08FH,0CBH,0DAH,0CBH
        DB      0DAH,0F8H,0DAH,0CBH,0F8H,0CBH,0E9H,0F8H,0E9H,0CBH
        DB      08FH,0CBH,0E9H,08FH,0E9H,0ADH,08FH,0ADH,0E9H,0ADH
        DB      0E9H,0F8H,0E9H,0ADH,0F8H,0ADH,0BCH,0F8H,0BCH,0ADH
```

```
;       FREQUENCY    = 130[Hz]
;       DATA LENGTH = 26

F130TM  CSEG
TMP130: DB      01AH,00FH,015H,043H,00CH,019H,059H,021H,050H,00FH
        DB      014H,012H,03BH,045H,012H,00AH,00FH,055H,01BH,05BH
        DB      018H,00CH,04DH,015H,005H,01AH
TMM130: DB      01AH,005H,015H,04DH,00CH,00FH,063H,017H,05AH,00FH
        DB      00AH,012H,045H,03BH,012H,014H,00FH,04BH,025H,051H
        DB      022H,00CH,043H,015H,00FH,01AH

F130DT  CSEG
DT130:  DB      08FH,0ADH,0BCH,0ADH,08FH,0BCH,08FH,0BCH,09EH,08FH
        DB      09EH,0BCH,09EH,0BCH,0F8H,0BCH,09EH,0F8H,09EH,0F8H
        DB      0DAH,09EH,0DAH,0F8H,0DAH,09EH
        DB      08FH,09EH,0DAH,09EH,08FH,0DAH,08FH,0DAH,0CBH,08FH
        DB      0CBH,0DAH,0CBH,0DAH,0F8H,0DAH,0CBH,0F8H,0CBH,0F8H
        DB      0E9H,0CBH,0E9H,0F8H,0E9H,0CBH
        DB      08FH,0CBH,0E9H,0CBH,08FH,0E9H,08FH,0E9H,0ADH,08FH
        DB      0ADH,0E9H,0ADH,0E9H,0F8H,0E9H,0ADH,0F8H,0ADH,0F8H
        DB      0BCH,0ADH,0BCH,0F8H,0BCH,0ADH


;       FREQUENCY    = 87[Hz]
;       DATA LENGTH = 36

F87TM   CSEG
TMP87:  DB      03FH,03FH,011H,015H,00EH,049H,025H,057H,022H,051H
        DB      00AH,019H,00CH,044H,013H,012H,015H,039H,043H,015H
        DB      008H,013H,04EH,00CH,00FH,00AH,058H,01AH,05DH,018H
        DB      05BH,00EH,00BH,011H,049H,035H
TMM87:  DB      035H,049H,011H,00BH,00EH,053H,01BH,061H,018H,05BH
        DB      00AH,00FH,00CH,04EH,013H,008H,015H,043H,039H,015H
        DB      012H,013H,044H,00CH,019H,00AH,04EH,024H,053H,022H
        DB      051H,00EH,015H,011H,03FH,03FH

F87DT   CSEG
DT87:   DB      0BCH,0ADH,08FH,0ADH,0BCH,08FH,0BCH,08FH,0BCH,09EH
        DB      08FH,09EH,0BCH,09EH,08FH,09EH,0BCH,09EH,0BCH,0F8H
        DB      0BCH,09EH,0BCH,0F8H,0BCH,09EH,0F8H,09EH,0F8H,09EH
        DB      0DAH,0F8H,0DAH,09EH,0DAH,09EH
        DB      0DAH,09EH,08FH,09EH,0DAH,08FH,0DAH,08FH,0DAH,0CBH
        DB      08FH,0CBH,0DAH,0CBH,08FH,0CBH,0DAH,0CBH,0DAH,0F8H
        DB      0DAH,0CBH,0DAH,0F8H,0DAH,0CBH,0F8H,0CBH,0F8H,0CBH
        DB      0E9H,0F8H,0E9H,0CBH,0E9H,0CBH
        DB      0E9H,0CBH,08FH,0CBH,0E9H,08FH,0E9H,08FH,0E9H,0ADH
        DB      08FH,0ADH,0E9H,0ADH,08FH,0ADH,0E9H,0ADH,0E9H,0F8H
        DB      0E9H,0ADH,0E9H,0F8H,0E9H,0ADH,0F8H,0ADH,0F8H,0ADH
        DB      0BCH,0F8H,0BCH,0ADH,0BCH,0ADH
```

```
;       FREQUENCY    = 65[Hz]
;       DATA LENGTH = 50

F65TM   CSEG
TMP65:  DB      040H,03CH,014H,013H,011H,044H,00DH,018H,00BH,04CH
        DB      024H,057H,022H,053H,022H,00AH,048H,00EH,015H,011H
        DB      040H,015H,010H,017H,038H,042H,017H,006H,015H,04AH
        DB      011H,00BH,00EH,052H,00AH,011H,061H,01AH,05DH,019H
        DB      05CH,00BH,00EH,00DH,04EH,011H,009H,014H,046H,036H
TMM65:  DB      036H,046H,014H,009H,011H,04EH,00DH,00EH,00BH,056H
        DB      01AH,061H,018H,05DH,018H,00AH,052H,00EH,00BH,011H
        DB      04AH,015H,006H,017H,042H,038H,017H,010H,015H,040H
        DB      011H,015H,00EH,048H,00AH,01BH,057H,024H,053H,023H
        DB      052H,00BH,018H,00DH,044H,011H,013H,014H,03CH,040H

F65DT   CSEG
DT65:   DB      0BCH,0ADH,08FH,0ADH,0BCH,0ADH,08FH,0ADH,0BCH,08FH
        DB      0BCH,08FH,0BCH,08FH,09EH,0BCH,09EH,08FH,09EH,0BCH
        DB      09EH,08FH,09EH,0BCH,09EH,0BCH,0F8H,0BCH,09EH,0BCH
        DB      0F8H,0BCH,09EH,0BCH,0F8H,09EH,0F8H,09EH,0F8H,09EH
        DB      0DAH,0F8H,0DAH,09EH,0DAH,0F8H,0DAH,09EH,0DAH,09EH
        DB      0DAH,09EH,08FH,09EH,0DAH,09EH,08FH,09EH,0DAH,08FH
        DB      0DAH,08FH,0DAH,08FH,0CBH,0DAH,0CBH,08FH,0CBH,0DAH
        DB      0CBH,08FH,0CBH,0DAH,0CBH,0DAH,0F8H,0DAH,0CBH,0DAH
        DB      0F8H,0DAH,0CBH,0DAH,0F8H,0CBH,0F8H,0CBH,0F8H,0CBH
        DB      0E9H,0F8H,0E9H,0CBH,0E9H,0F8H,0E9H,0CBH,0E9H,0CBH
        DB      0E9H,0CBH,08FH,0CBH,0E9H,0CBH,08FH,0CBH,0E9H,08FH
        DB      0E9H,08FH,0E9H,08FH,0ADH,0E9H,0ADH,08FH,0ADH,0E9H
        DB      0ADH,08FH,0ADH,0E9H,0ADH,0E9H,0F8H,0E9H,0ADH,0E9H
        DB      0F8H,0E9H,0ADH,0E9H,0F8H,0ADH,0F8H,0ADH,0F8H,0ADH
        DB      0BCH,0F8H,0BCH,0ADH,0BCH,0F8H,0BCH,0ADH,0BCH,0ADH
```

```
;       FREQUENCY    = 43[Hz]
;       DATA LENGTH = 74

F43TM   CSEG
TMP43:  DB      040H,03BH,016H,010H,015H,040H,011H,014H,010H,045H
        DB      00DH,018H,00CH,04AH,024H,056H,024H,056H,023H,053H
        DB      023H,053H,023H,00AH,048H,00EH,016H,00FH,043H,012H
        DB      012H,014H,03DH,016H,010H,017H,038H,042H,017H,006H
        DB      016H,047H,014H,008H,012H,04DH,00FH,00CH,00EH,052H
        DB      00AH,010H,060H,01AH,061H,019H,05EH,019H,05DH,019H
        DB      05CH,00CH,00EH,00DH,04FH,010H,00AH,011H,04AH,015H
        DB      006H,016H,045H,036H
TMM43:  DB      036H,045H,016H,006H,015H,04AH,011H,00AH,010H,04FH
        DB      00DH,00EH,00CH,054H,01AH,060H,01AH,060H,019H,05DH
        DB      019H,05DH,019H,00AH,052H,00EH,00CH,00FH,04DH,012H
        DB      008H,014H,047H,016H,006H,017H,042H,038H,017H,010H
        DB      016H,03DH,014H,012H,012H,043H,00FH,016H,00EH,048H
        DB      00AH,01AH,056H,024H,057H,023H,054H,023H,053H,023H
        DB      052H,00CH,018H,00DH,045H,010H,014H,011H,040H,015H
        DB      010H,016H,03BH,040H

F43DT   CSEG
DT43:   DB      0BCH,0ADH,08FH,0ADH,0BCH,0ADH,08FH,0ADH,0BCH,0ADH
        DB      08FH,0ADH,0BCH,08FH,0BCH,08FH,0BCH,08FH,0BCH,08FH
        DB      0BCH,08FH,09EH,0BCH,09EH,08FH,09EH,0BCH,09EH,08FH
        DB      09EH,0BCH,09EH,08FH,09EH,0BCH,09EH,0BCH,0F8H,0BCH
        DB      09EH,0BCH,0F8H,0BCH,09EH,0BCH,0F8H,0BCH,09EH,0BCH
        DB      0F8H,09EH,0F8H,09EH,0F8H,09EH,0F8H,09EH,0F8H,09EH
        DB      0DAH,0F8H,0DAH,09EH,0DAH,0F8H,0DAH,09EH,0DAH,0F8H
        DB      0DAH,09EH,0DAH,09EH
        DB      0DAH,09EH,08FH,09EH,0DAH,09EH,08FH,09EH,0DAH,09EH
        DB      08FH,09EH,0DAH,08FH,0DAH,08FH,0DAH,08FH,0DAH,08FH
        DB      0DAH,08FH,0CBH,0DAH,0CBH,08FH,0CBH,0DAH,0CBH,08FH
        DB      0CBH,0DAH,0CBH,08FH,0CBH,0DAH,0CBH,0DAH,0F8H,0DAH
        DB      0CBH,0DAH,0F8H,0DAH,0CBH,0DAH,0F8H,0DAH,0CBH,0DAH
        DB      0F8H,0CBH,0F8H,0CBH,0F8H,0CBH,0F8H,0CBH,0F8H,0CBH
        DB      0E9H,0F8H,0E9H,0CBH,0E9H,0F8H,0E9H,0CBH,0E9H,0F8H
        DB      0E9H,0CBH,0E9H,0CBH
        DB      0E9H,0CBH,08FH,0CBH,0E9H,0CBH,08FH,0CBH,0E9H,0CBH
        DB      08FH,0CBH,0E9H,08FH,0E9H,08FH,0E9H,08FH,0E9H,08FH
        DB      0E9H,08FH,0ADH,0E9H,0ADH,08FH,0ADH,0E9H,0ADH,08FH
        DB      0ADH,0E9H,0ADH,08FH,0ADH,0E9H,0ADH,0E9H,0F8H,0E9H
        DB      0ADH,0E9H,0F8H,0E9H,0ADH,0E9H,0F8H,0E9H,0ADH,0E9H
        DB      0F8H,0ADH,0F8H,0ADH,0F8H,0ADH,0F8H,0ADH,0F8H,0ADH
        DB      0BCH,0F8H,0BCH,0ADH,0BCH,0F8H,0BCH,0ADH,0BCH,0F8H
        DB      0BCH,0ADH,0BCH,0ADH
```

```
;       FREQUENCY    = 33[Hz]
;       DATA LENGTH = 98

F33TM   CSEG
TMP33:  DB      041H,039H,017H,010H,016H,03DH,014H,012H,013H,041H
        DB      011H,014H,010H,045H,00DH,018H,00CH,049H,00AH,01AH
        DB      056H,024H,056H,023H,056H,023H,054H,022H,054H,023H
        DB      053H,00AH,019H,00CH,047H,00EH,016H,00FH,043H,011H
        DB      014H,012H,03FH,014H,011H,016H,03BH,03FH,037H,041H
        DB      035H,045H,016H,007H,014H,049H,012H,00AH,011H,04DH
        DB      00FH,00CH,00EH,051H,00CH,00FH,00AH,055H,01AH,060H
        DB      01AH,060H,019H,05EH,019H,05EH,019H,05DH,019H,00AH
        DB      053H,00CH,00EH,00DH,04FH,010H,00AH,011H,04BH,013H
        DB      008H,014H,047H,016H,006H,017H,043H,037H
TMM33:  DB      037H,043H,017H,006H,016H,047H,014H,008H,013H,04BH
        DB      011H,00AH,010H,04FH,00DH,00EH,00CH,053H,00AH,010H
        DB      060H,01AH,060H,019H,060H,019H,05EH,018H,05EH,019H
        DB      05DH,00AH,00FH,00CH,051H,00EH,00CH,00FH,04DH,011H
        DB      00AH,012H,049H,014H,007H,016H,045H,035H,041H,037H
        DB      03FH,03BH,016H,011H,014H,03FH,012H,014H,011H,043H
        DB      00FH,016H,00EH,047H,00CH,019H,00AH,04BH,024H,056H
        DB      024H,056H,023H,054H,023H,054H,023H,053H,023H,00AH
        DB      049H,00CH,018H,00DH,045H,010H,014H,011H,041H,013H
        DB      012H,014H,03DH,016H,010H,017H,039H,041H

F33DT   CSEG
DT33:   DB      0BCH,0ADH,08FH,0ADH,0BCH,0ADH,08FH,0ADH,0BCH,0ADH
        DB      08FH,0ADH,0BCH,0ADH,08FH,0ADH,0BCH,0ADH,08FH,0BCH
        DB      08FH,0BCH,08FH,0BCH,08FH,0BCH,08FH,0BCH,08FH,0BCH
        DB      09EH,08FH,09EH,0BCH,09EH,08FH,09EH,0BCH,09EH,08FH
        DB      09EH,0BCH,09EH,08FH,09EH,0BCH,09EH,0BCH,09EH,0BCH
        DB      09EH,0BCH,0F8H,0BCH,09EH,0BCH,0F8H,0BCH,09EH,0BCH
        DB      0F8H,0BCH,09EH,0BCH,0F8H,0BCH,09EH,0F8H,09EH,0F8H
        DB      09EH,0F8H,09EH,0F8H,09EH,0F8H,09EH,0F8H,0DAH,09EH
        DB      0DAH,0F8H,0DAH,09EH,0DAH,0F8H,0DAH,09EH,0DAH,0F8H
        DB      0DAH,09EH,0DAH,0F8H,0DAH,09EH,0DAH,09EH
        DB      0DAH,09EH,08FH,09EH,0DAH,09EH,08FH,09EH,0DAH,09EH
        DB      08FH,09EH,0DAH,09EH,08FH,09EH,0DAH,09EH,08FH,0DAH
        DB      08FH,0DAH,08FH,0DAH,08FH,0DAH,08FH,0DAH,08FH,0DAH
        DB      0CBH,08FH,0CBH,0DAH,0CBH,08FH,0CBH,0DAH,0CBH,08FH
        DB      0CBH,0DAH,0CBH,08FH,0CBH,0DAH,0CBH,0DAH,0CBH,0DAH
        DB      0CBH,0DAH,0F8H,0DAH,0CBH,0DAH,0F8H,0DAH,0CBH,0DAH
        DB      0F8H,0DAH,0CBH,0DAH,0F8H,0DAH,0CBH,0F8H,0CBH,0F8H
        DB      0CBH,0F8H,0CBH,0F8H,0CBH,0F8H,0CBH,0F8H,0E9H,0CBH
        DB      0E9H,0F8H,0E9H,0CBH,0E9H,0F8H,0E9H,0CBH,0E9H,0F8H
        DB      0E9H,0CBH,0E9H,0F8H,0E9H,0CBH,0E9H,0CBH
        DB      0E9H,0CBH,08FH,0CBH,0E9H,0CBH,08FH,0CBH,0E9H,0CBH
        DB      08FH,0CBH,0E9H,0CBH,08FH,0CBH,0E9H,0CBH,08FH,0E9H
        DB      08FH,0E9H,08FH,0E9H,08FH,0E9H,08FH,0E9H,08FH,0E9H
        DB      0ADH,08FH,0ADH,0E9H,0ADH,08FH,0ADH,0E9H,0ADH,08FH
        DB      0ADH,0E9H,0ADH,08FH,0ADH,0E9H,0ADH,0E9H,0ADH,0E9H
        DB      0ADH,0E9H,0F8H,0E9H,0ADH,0E9H,0F8H,0E9H,0ADH,0E9H
        DB      0F8H,0E9H,0ADH,0E9H,0F8H,0E9H,0ADH,0F8H,0ADH,0F8H
        DB      0ADH,0F8H,0ADH,0F8H,0ADH,0F8H,0ADH,0F8H,0BCH,0ADH
        DB      0BCH,0F8H,0BCH,0ADH,0BCH,0F8H,0BCH,0ADH,0BCH,0F8H
        DB      0BCH,0ADH,0BCH,0F8H,0BCH,0ADH,0BCH,0ADH

        END
```

6.8　PWMデータ作成時の留意点

6.8.1　スイッチング・タイミング

マクロ・サービスでもっとも時間的に厳しい部分は，マクロ・サービス完了割り込み付近となります。すなわち，いずれかの相のゼロクロス付近となります。ここで，ゼロクロス点から最初にスイッチングするまでの時間，ならびに，ゼロクロス直前のスイッチングからゼロクロス点まで時間は，図6－29に示すように，基準正弦波より，120°位相が進んだ波形，および240°位相が進んだ波形と，キャリア波形との交点になります。

したがって，次のようになります。

$\sin\left(2\pi f_i t+\frac{2}{3}\pi f_i\right) = 4 f_c t$

ただし，f_i ＝ インバータ出力周波数

f_c ＝ キャリア周波数

$f_c \gg f_i$かつ$t<\frac{1}{4f_c}$のとき

$\sin\frac{2}{3}\pi \fallingdotseq 4f_c t$

キャリア周波数＝3.125 kHzとすると，近似的に

$$\sin\frac{2}{3}\pi\times\frac{1}{3.125\times10^{3}}\times\frac{1}{4} \fallingdotseq 0.866\times80\times10^{-6}$$

$$= 69.28\,\mu s$$

図6－29　ゼロクロス後のスイッチング

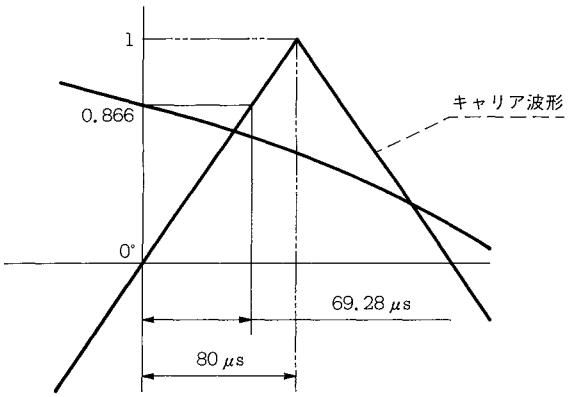
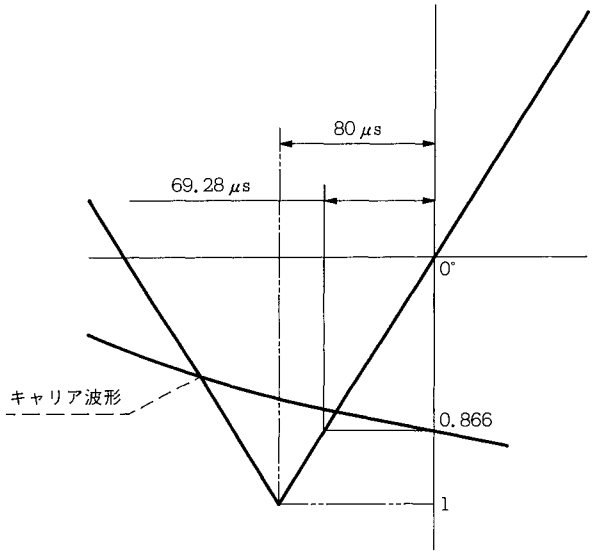


図6－30　ゼロクロス前のスイッチング



6.8.2　デッド・タイムの加味

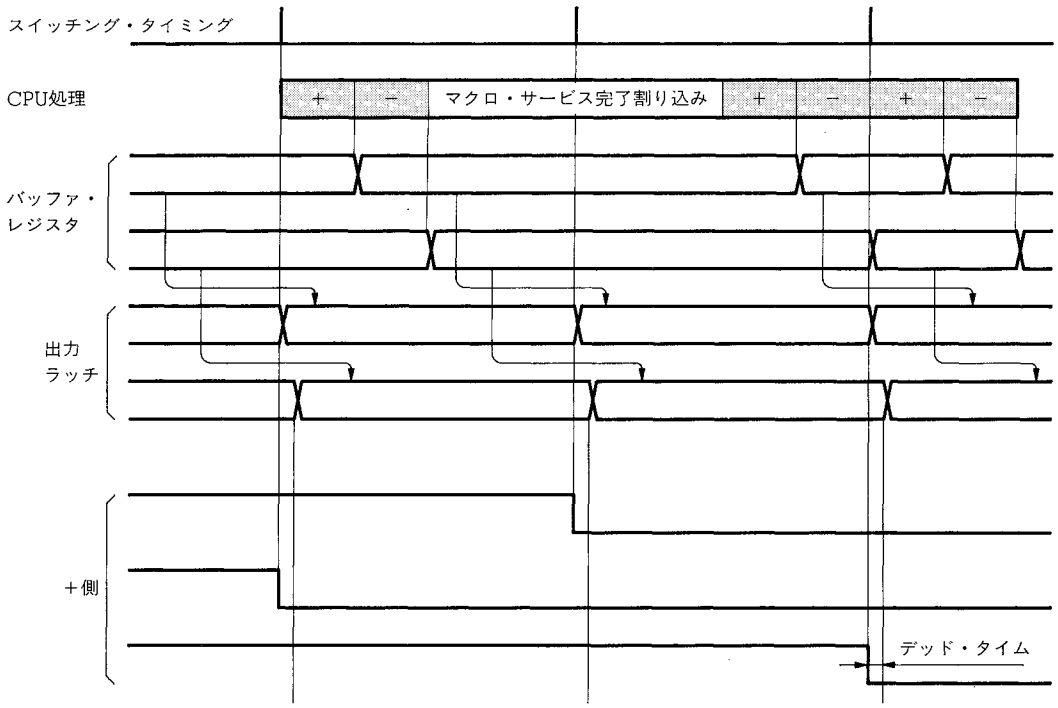
前述のスイッチング・タイミングにおいて，同一タイミングで，＋側トランジスタと－側トランジスタをスイッチングできないため，デッド・タイムが必要となり，ON側のタイミングは，デッド・タイム分遅れることになります。

したがって，前述のスイッチング・タイミングから，このデッド・タイムを差し引いた時間が，実質のスイッチング時間となります。この応用例では13.33 μsのデッド・タイムを使用していますので，次のようになります。

$$69.28\,\mu s - 13.33\,\mu s = 55.95\,\mu s$$

ここで，ゼロクロス付近のマクロ・サービスのタイミングを考えると図6－31のようになります。

図6－31　マクロ・サービスのタイミング



したがって，＋側／－側の完了割り込み，＋側／－側のマクロ・サービス２回分が，デッド・タイムを加味したスイッチング時間の２倍より小さくなければなりません。

ここで，完了割り込みの処理時間は11.50 μs，マクロ・サービス・タイプCの処理時間は8.17 μsなので，最小時間およびゼロクロス付近での余裕時間は次のようになります。

●最小時間

$$11.50\,\mu s \times 2 + 8.17\,\mu s \times 2 \times 2 = 55.67\,\mu s$$

●ゼロクロス付近での余裕時間

$$55.95\,\mu s \times 2 - 55.67\,\mu s = 56.23\,\mu s$$

したがって，少なくとも，これより長いベクタ割り込み処理では，ネスティングを許可するなどの対策が必要となります。

第7章　表示／キー入力

8セグメントLED 4桁の表示と，16 (4×4) キーのキー入力をベクタ割り込みとマクロ・サービスの二通りで処理した応用例を示します。

表7-1に表示およびキー入力の仕様，図7-2に回路図を示します。

表7-1　表示およびキー入力の仕様

	項　目	仕　様
表示	セグメント信号	P00-P07（ハイ・アクティブ）
	ディジット信号	P34-P37（ロウ・アクティブ）
キー入力	キー・スキャン信号	P00-P03（　〃　）
	キー・リターン信号	P24-P27（　〃　）
	入力形式	ロック・アウト方式，多重キー禁止
	チャタリング回避時間	20 ms（Min.）－25 ms（Max.）
	キー・コードとキーの対応	図7-1に示す。

図7-1　キー・コードとキーの対応

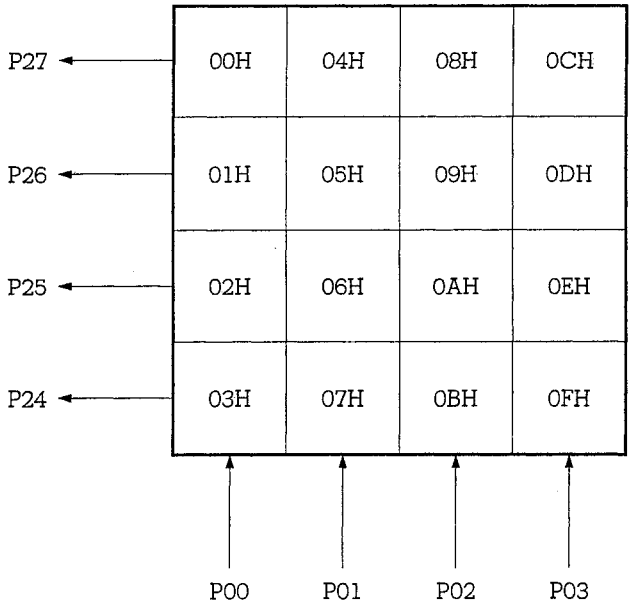
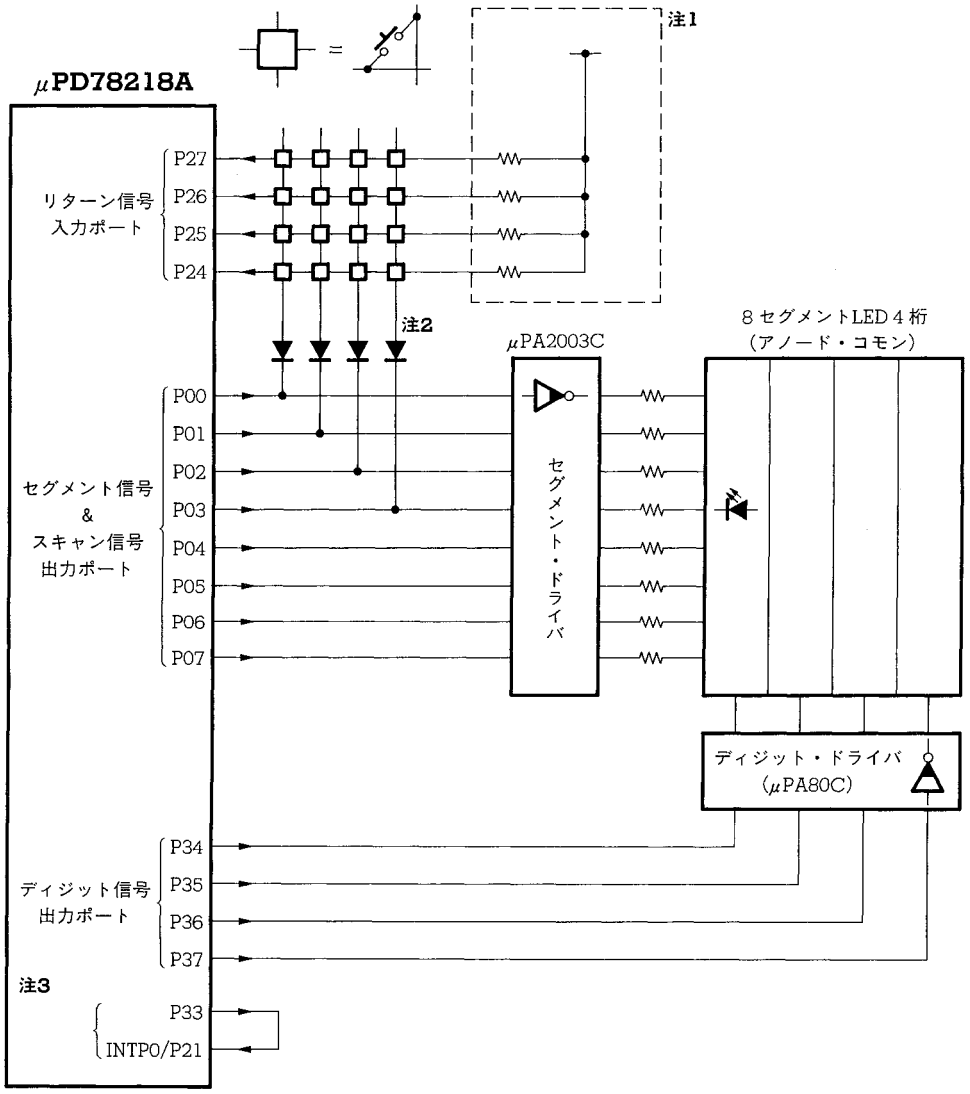


図7-2　8セグメントLED4桁と16（4×4）キー・マトリクス



注1. P22-P27は，ソフトウエアにより6ビット一括で内蔵プルアップ抵抗の接続指定が可能であり，この機能を使用すれば部品点数の削減が図れます（ただし，μPD78224シリーズではできません）。

2. 2つ以上同時にキーが押された場合のポート破壊防止のため，ダイオードを接続しています。

3. マクロ・サービスを使ったプログラムでのみ使用するポート

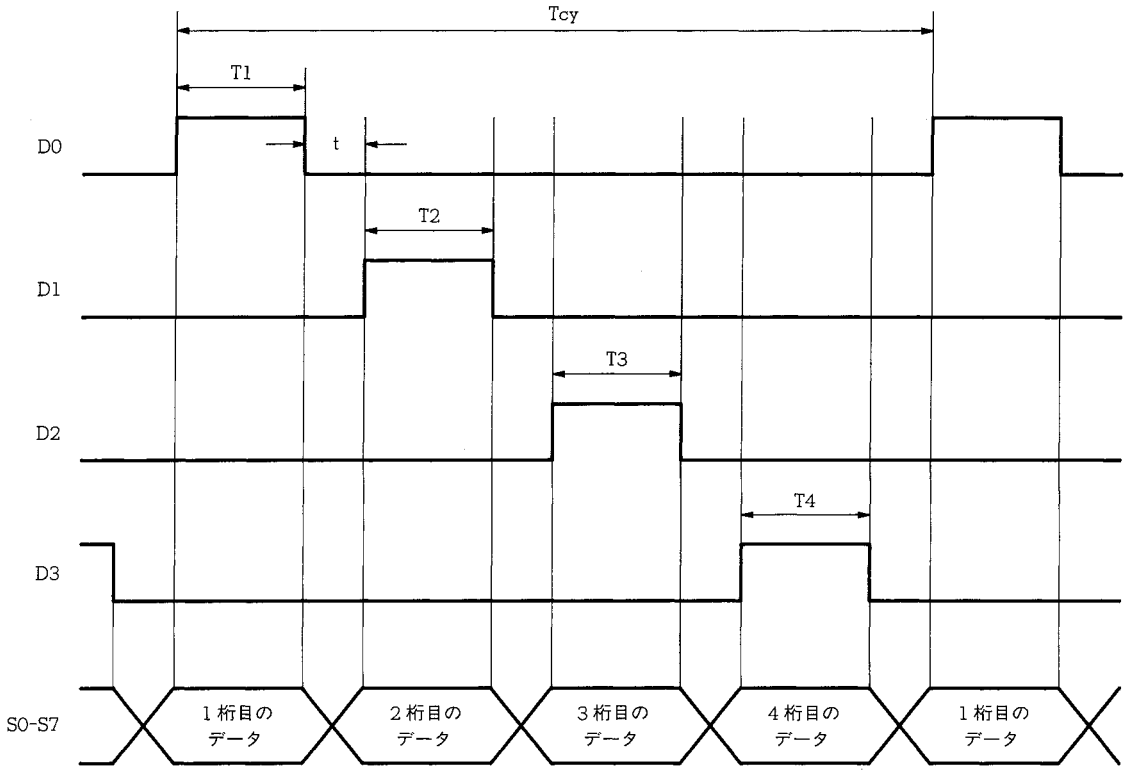
(1) 表示について

シングルチップ・マイクロコンピュータによって表示を制御する場合は，ポート数の制限によりダイナミック表示にするのが一般的です。

ダイナミック表示とは，各表示桁を一定周期で順番に点灯させる方法です。

図7-3に，ダイナミック表示の処理を行う場合のセグメント信号（S0-S7）と，ディジット信号（D0-D3）の出力タイミングを示します。ディジット信号D0から順に出力していき，D3の出力を終了すると再びD0からの出力を行います。セグメント信号は，ディジット信号がアクティブになっている桁の表示データを出力します。

図7-3　表示のタイミング



表示桁のスキャン・サイクル・タイム（Tcy）の設定値は，低周波数による人間の目に対するちらつき感を防ぐために，通常10 ms－5 ms（100 Hz－200 Hz）程度にします。室内蛍光灯下のちらつきを防ぐため，商用電源周波数の倍数（100 Hzまたは120 Hz）は避けてください。

このとき，Tcyを大きくしすぎると消灯状態が目立つようになり，表示がちらついてしまいます。

7
214
218A
224
234
244

また，ある桁から次の桁へ表示を切り替える際には，前の桁のセグメント出力が次の桁に表示されないように，表示の切れ目の時間（t）をとる必要があります。

この時間tは，セグメント信号，ディジット信号の出力を変更するのに要する時間(ドライバの遅延時間，ソフトウエア／マクロ・サービスの処理時間）によって決定します。

なお，tを大きくとりすぎると，表示の点灯時間が削減され表示が暗くなりますので，表示デューティは十分検討する必要があります。表示デューティは次の式で求められます。

$$Duty = \frac{\text{各表示桁の点灯時間}}{\text{全消灯時間＋全点灯時間}} = \frac{Tn}{(Tn+t)\times n} \quad (n = 1-4)$$

(2) キー入力について

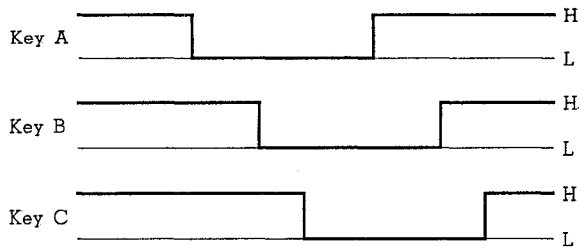
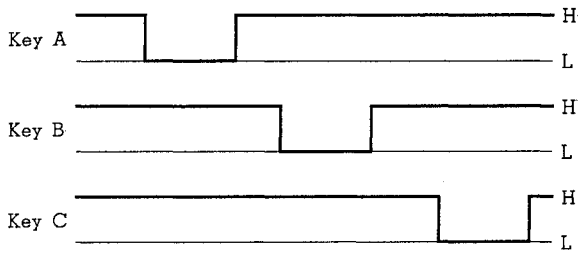
(a) キーの二重押しに対する処理

次に示す二通りの方法があります。

- ●ロック・アウト方式　　：すべてのキーが離された状態から，入力された１つのキーを有効なキーとして取り込みます。
- ●２キー・ロール・オーバ方式：１つのキーが押された状態で，もう一つのキーが押されると，あとから押されたキーも有効なキー入力として取り込みます（3つ以上のキーが同時に押されたときには，キー入力は無効となります)。

表7-4に，ロック・アウト方式と，２キー・ロール・オーバ方式のキー入力の様子を示します。

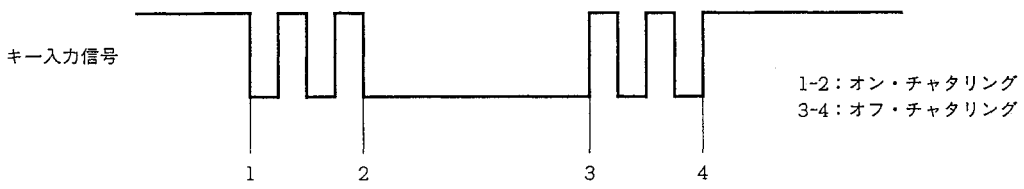
図7-4　キー入力の様子（ロウ・アクティブの場合）

キー入力タイミング	キー入力方法	
	ロック・アウト方式	2キー・ロール・オーバ方式
(1) 	Key Aのみ取り込む	Key A，Key Bを取り込む
(2) 	Key A，B，Cとも取り込む	Key A，B，Cとも取り込む

(b) キーのチャタリングについて

キーをオン，またはオフする場合，キー信号がハイ・レベル，またはロウ・レベルに安定するまでの間に，チャタリングと呼ばれる不安定な状態が存在します（オン・チャタリング，オフ・チャタリング）。誤動作を引き起こす恐れがあるため，ハードウエアでチャタリング防止回路を設けていない限り，ソフトウエアでキー信号のチャタリングを回避しなければなりません。

図7-5　キー入力信号のチャタリングの様子（ロウ・アクティブの場合）



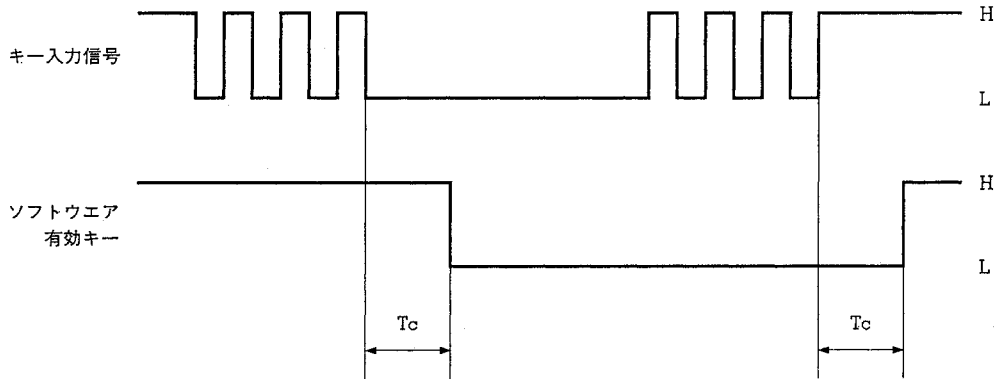
ソフトウエアで，チャタリングを制御する場合には，キー信号がある一定時間以上安定した場合に，そのキー信号を有効とするという方法をとります。

この有効とするまでの時間をチャタリング吸収時間と呼びます。

チャタリング吸収時間を考慮してプログラムを組んだ場合，ソフトウエア上の有効キー信号は図7-6のようになります。

7
214
218A
224
234
244

図7-6　ソフトウエア上の有効キー（ロウ・アクティブの場合）



備考　Tc：チャタリング吸収時間

(3) キー入力とLED表示の対応

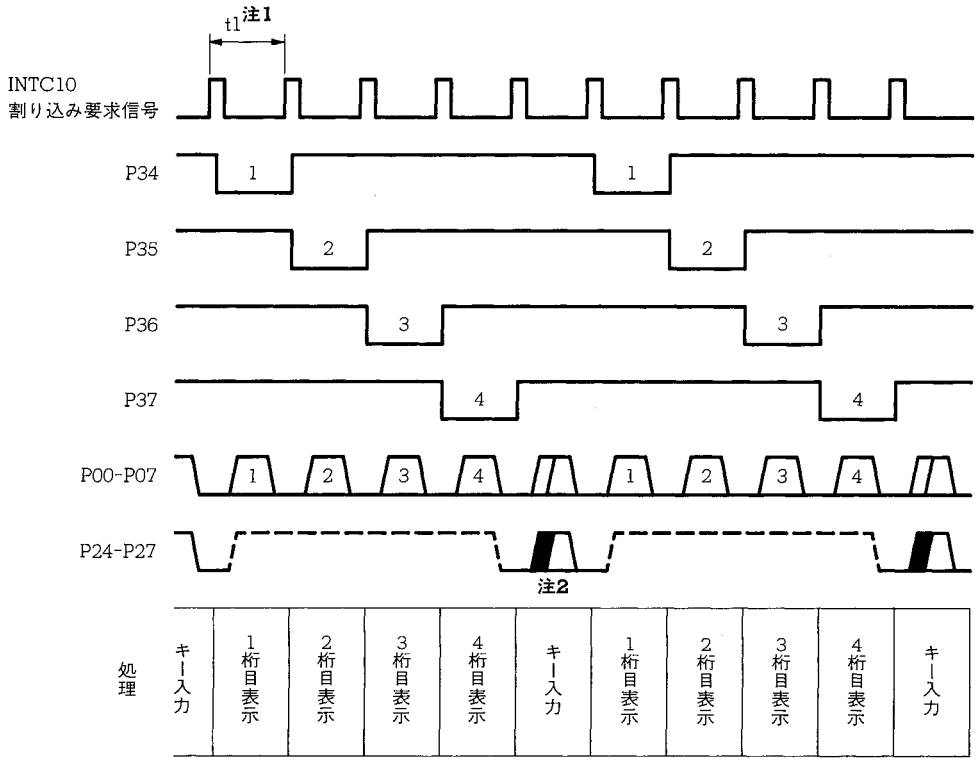
有効なキー入力がされた場合，そのときのキー・コードをセグメント表示データに変換し，LED表示させるようにします。

7.1　ベクタ割り込みによるプログラム例

(1) 動作概要

8ビット・タイマ／カウンタ1のインターバル・タイマ割り込み（INTC10）により，表示とキー入力の処理を時分割で行います。タイミング・チャートを図7-7に示します。

図7-7　表示とキー入力のタイミング

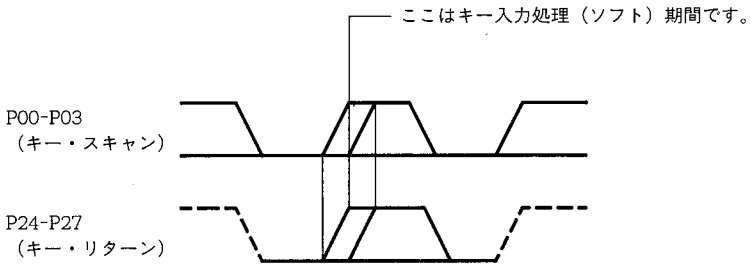


注1. t1＝タイマのカウント・クロック(f_{CLK}/32)周期×(CR10設定値＋1)

＝5.3 μs(分解能)×((188－1)＋1)

≒1.00 ms

注2.



このプログラム例では，5回のタイマ割り込みを1つのサイクル（スキャン・サイクル・タイム）とし，4回を表示処理に，1回をキー入力処理に使用します。この場合の表示デューティは1/5になります。

タイマのインターバル時間は，1.00 ms（=t1）とします。

5回のタイマ割り込み発生で，1回のキー入力処理を行うので，チャタリング吸収時間は1サイクル分の時間のズレ[注]を考慮して次のようになります。

1.00 ms × 5回 × 4サイクル＝20.00 ms（Min.）
1.00 ms × 5回 × 5サイクル＝25.00 ms（Max.）

注　1つのサイクル中のどこでキー入力されるかにより，最大1サイクル分のズレが生じます。

また，表示のスキャン・サイクル・タイムは次のようになります。

1.00 ms × 5回＝5.00 ms（200.0 Hz）

(2) プログラム説明

(a) モジュール説明

表7-2にこのプログラム例のモジュール一覧を示します。

表7－2　モジュール一覧

モジュール	レーベル	処　　理
セグメント・データ変換 ：SG_CONV	SEGCNV	LEDセグメント・データ変換処理
表示／キー入力共通 ：I_CMN	INTC10	INTC10ベクタ割り込み処理
表示関連　：I_DISP	D_INI	表示エリア・イニシャライズ処理
	DISP_I	表示処理
キー入力関連：I_KEY	K_INI	キー・エリア・イニシャライズ処理
	KEY_I	キー入力処理

(b) モジュール処理概要

(i) セグメント・データ変換処理モジュール：SG_CONV

（このモジュールは，**7.2　マクロ・サービスによるプログラム例**と共通で使用できます）。

①　LEDセグメント・データ変換処理［レーベル名称：SEGCNV］

ここでは，アキュムレータの内容（0H-FH）をテーブル参照によりLEDセグメント・データに変換する処理を行っています。

(ii) 表示／キー入力共通モジュール：I_CMN

①　INTC10ベクタ割り込み処理［レーベル名称：INTC10］

ここでは，次の処理を行っています。

- セグメント信号（P00-P07）およびディジット信号（P34-P37）をOFFにすることによりブランキング期間を生成しています。
- タイミング・カウンタ（DSPCNT）の値により表示処理とキー入力処理の切り替えを行います。

(iii) 表示関連モジュール：I_DISP

①　表示エリア・イニシャライズ処理［レーベル名称：D_INI］

ここでは，表示エリアをイニシャライズしています（**表7－4**参照）。

7
214
218A
224
234
244

② **表示処理［レーベル名称：DISP_I］**

ここでは，タイミング・カウンタ（DSPCNT）の値により，示される桁に対するディジット信号（P34-P37）およびセグメント信号（P00-P07）を出力しています。

(iv) キー入力関連モジュール：I_KEY

① **キー・エリア・イニシャライズ処理［レーベル名称：K_INI］**

ここでは，キー・エリアをイニシャライズしています（**表7－5**参照）。

② **キー入力処理［レーベル名称：KEY_I］**

ここでは次の処理を行っています。

- キー・スキャン信号（P00-P03）出力
- キー・リターン信号（P24-P27）入力
- キー・コード生成
- キーの多重押しのチェック
- チャタリング吸収
- ロック・アウト処理

次に詳細を示します。

- キー・スキャン信号（P00-P03）出力

  キー・スキャン信号（P00-P03）の切り替えはP0をシフトさせることにより行います。

- キー・リターン信号入力

  キー・スキャン信号（P00-P03）出力後，ウエイト[注]期間をおいてキー・リターン信号（P24-P27）を取り込みます。

  注　78K/IIシリーズでは最小インストラクション・サイクルが333 ns（f_{CLK}＝6 MHz時）という高速のため，このプログラムでは，キー・スキャン信号出力後，キー・リターン信号入力までの間，約10 μs（外部ROMフェッチ時）のウエイト期間を設定しています。

- キー・コード生成

  取り込んだキー・リターン信号（P24-P27）を1ビット・シフトし，そのときのキャリー状態によりキーが押されたことを判断し，スキャン・キー（Eレジスタ）およびスキャン終了キー（Dレジスタ）を用いて生成します。

●キーの多重押しのチェック

キー・コード生成中，キーが押されたと判断したときにキー・オン・フラグ(KEYON：X.7）をセットし，このフラグの状態により多重キーが入力されたことを判断し，キー入力を無効とします。

●チャタリング吸収

スキャン終了キー（Dレジスタ）とチャタリング・キー（CHTKEY）を比較し，4回内容が一致（チャタリング・カウンタ（CHTCNT）によりカウント）すれば，チャタリングが吸収したとみなします。

●ロック・アウト処理

チャタリング吸収後，キー・オン・フラグ（KEYON）がリセットされている場合は全キーがOFFとなっているので，OFFキー・フラグ（KEYOFF）をセットし，このフラグがセットされているときだけキーを有効にします（チャタリング・キー（CHTKEY）の内容を有効キー（KEYCOD）に格納し，有効キー・フラグ（KEYOK）をセット）。

(c) 使用するワーク・エリア

このプログラムでは，表示／キー入力共通処理用で1バイト，表示処理用で5バイト，キー入力処理用で7バイト＋3ビットのワーク・エリアを使用します。

表7-3に表示／キー入力共通，表7-4に表示関連，表7-5にキー入力関連ワーク・エリアの一覧を示します。

表7-3　表示／キー入力共通ワーク・エリア

［表示／キー入力共通モジュール：I_CMN］

ワーク・エリア名称	配置領域[注]	用　　途	バイト数	初期値
タイミング・カウンタ DSPCNT	SADDR	表示／キー入力タイミング・カウンタ	1	00H

注　配置領域の名称は，次の意味を示します。

SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

7
214
218A
224
234
244

表7-4　表示関連ワーク・エリア

[表示関連モジュール：I_DISP]

ワーク・エリア名称	配置領域[注]	用　　途	バイト数	初期値
DSPDT (LED0-LED3)	RAM	表示データ格納エリア（LED 4 桁分） （この表示データをセグメント信号としてポート0より出力）	4	すべて 00H
DGTWRK	SADDR	ディジット信号生成用ワーク・エリア	1	—

注　配置領域の名称は，次の意味を示します。

RAM　：メモリ・バンク0内の任意のRAM領域

SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

表7-5　キー入力関連ワーク・エリア

[キー入力関連モジュール：I_KEY]

ワーク・エリア名称	配置領域[注]	用　　途	バイト数	初期値
有効キー KEYCOD	SADDR	有効となったキー・コードを格納	1	FFH
チャタリング・キー CHTKEY		チャタリング吸収中のキー・コードを格納		EEH
チャタリング・カウンタ CHTCNT		チャタリング回数カウンタ		04H
OFFキー・フラグ KEYOFF		オフ・チャタリングの吸収完了を確認	1ビット	0
有効キー・フラグ KEYOK		有効キーの入力を確認		
キー・オン・フラグ KEYON（X.7）	レジスタ・バンク1	キー入力の検出を状態として保持		
スキャン終了キー D		キー・スキャンにより検出したキー・コードを格納	1	FFH
スキャン・キー E		キー・スキャン中のキー・コードを格納		00H

注　配置領域の名称は，次の意味を示します。

SADDR　　：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

レジスタ・バンク1：汎用レジスタ領域のバンク1

(3) 入出力パラメータ

表7-6に入力パラメータ，表7-7に出力パラメータの一覧を示します。

表7-6　入力パラメータ

モジュール	パラメータ	固定／可変	用　　途
セグメント・データ変換 ：SG_CONV	Aレジスタ （バンク0）	可変値	表示データ（00H-0FH）を入力

表7-7　出力パラメータ

モジュール	パラメータ	固定／可変	用　　途
セグメント・データ変換 ：SG_CONV	Aレジスタ （バンク0）	可変値	LEDセグメント・データを出力
表示／キー入力共通 ：I_CMN	SEGOFF	固定値	P0への設定値（セグメント信号の出力（P00-P07）をOFFにする）
	DGTOFF		P3への設定値(ディジット信号の出力(P34-P37）をOFFにする）
	DSPCNT	可変値	① タイミング・カウンタとして現在のLEDの表示桁を表す ② キー入力のタイミングを生成
表示関連　：I_DISP	LED0,LED1, LED2, LED3		表示データ（4バイト）格納エリアの各アドレス
キー入力関連：I_KEY	KEYOK		有効キー・フラグ(有効なキー入力があったときにセット）
	KEYCOD		チャタリングの吸収を完了し,有効となったキー・コード（KEYOK＝1のとき，このキー・コードが有効）

7
214
218A
224
234
244

(4) 使用レジスタ

表7－8に使用レジスタの一覧を示します。

表7－8　使用レジスタ

モジュール	レーベル	使用レジスタ	バンク
セグメント・データ変換 :SG_CONV	SEGCNV	A	0
表示／キー入力共通 :I_CMN	INTC10	なし	—
表示関連　:I_DISP	D_INI	A, B, HL	0
	DISP_I	A, B	1
キー入力関連:I_KEY	K_INI	なし	—
	KEY_I	A, X[注], B, C, D, E	1

注　Xレジスタは1ビットのみ使用（X.7）

(5) 使用スタック

この応用例のサブ・ルーチンおよび割り込み処理で使用するスタック・サイズは，9バイトです（ネスティング：4レベル）。

(6) モード・レジスタ設定例

ポート0モード・レジスタ

	7	6	5	4	3	2	1	0	
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	(リセットで00H)
設定例	0	0	0	0	0	0	0	0	

PM0n	P0n端子入出力モード指定 (n=0-7)
00H	出力モード (出力バッファ・オン)
FFH	ハイ・インピーダンス状態(出力バッファ・オフ)
上記以外	設定禁止

ポート3モード・レジスタ

	7	6	5	4	3	2	1	0	
PM3	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30	(リセットでFFH)
設定例	0	0	0	0	1	1	1	1	

PM3n	P3n端子入出力モード指定 (n=0-7)
0	出力モード
1	入力モード

7
214
218A
224
234
244

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	0	0	0	0	0	1	1	

8ビット・タイマ／カウンタ1

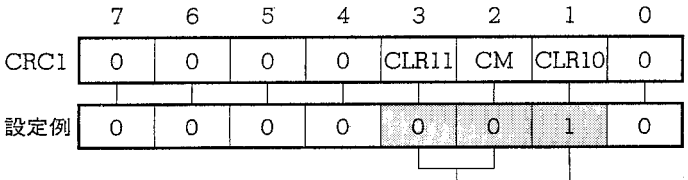
PRS12	PRS11	PRS10	カウント・クロック[Hz]の指定
0	0	0	f_{CLK}/16
0	0	1	
0	1	0	
0	1	1	f_{CLK}/32
1	0	0	f_{CLK}/64
1	0	1	f_{CLK}/128
1	1	0	f_{CLK}/256
1	1	1	f_{CLK}/512

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	カウント・クロック[Hz]の指定
0	0	0	0	f_{CLK}/16
0	0	0	1	
0	0	1	0	
0	0	1	1	f_{CLK}/32
0	1	0	0	f_{CLK}/64
0	1	0	1	f_{CLK}/128
0	1	1	0	f_{CLK}/256
0	1	1	1	f_{CLK}/512
1	1	1	1	外部クロック(CI)

備考　f_{CLK}：内部システム・クロック

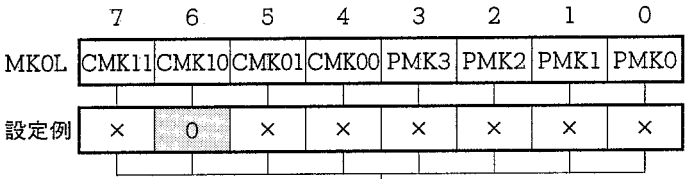
キャプチャ／コンペア・コントロール・レジスタ1



CLR10	TM1とCR10レジスタの内容一致でのクリア動作
0	禁止
1	許可

CLR11	CM	CR11レジスタの動作指定	TM1のクリア動作
0	0	コンペア・レジスタ	禁止
1	0		許可（TM1とCR11レジスタの内容一致のとき）
0	1	キャプチャ・レジスタ	禁止
1	1		許可（TM1の内容をCR11レジスタにキャプチャしたとき）

割り込みマスク・レジスタL



MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

7
214
218A
224
234
244

タイマ・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	0	0	0	0	1	0	0	0	

8ビット・タイマ／カウンタ1

OVF1	TM1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFHから00Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE1	TM1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

CMD2	TM2のカウント動作モード指定
0	通常モード
1	ワンショット・モード

OVF2	TM2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFHから00Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE2	TM2のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

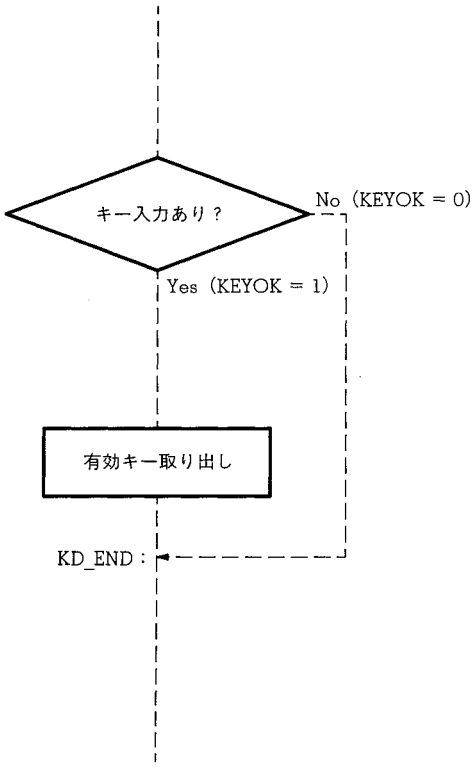
(7) プログラム使用例

キー入力処理と表示処理の使用例を別々に示します。両方の処理を行いたい場合は，組み合わせて行ってください。

(a) キー入力処理

このプログラム例では，有効キー・フラグを調べることによりキー入力の有無を確認します。

```
EXTRN   K_INI
EXTRN   SEGOFF,DSPCNT,KEYCOD    ; parameter
EXTBIT  KEYOK                   ; parameter
        :
        :
CSEG
        :
        :
MOV     P0,#LOW SEGOFF          ; initialize PORT0
MOV     PM0,#00000000B          ; PORT0 <-- output port

MOV     CRC1,#00000010B         ; enable clear TM1 by CR10
MOV     PRM1,#00000011B         ; set prescaler(TM1:fclk/32)
MOV     CR10,#188-1             ; set interval timer (1.00 ms)
MOV     DSPCNT,#0               ; initialize display counter
CALL    !K_INI                  ; initialize keyinput area
CLR1    CMK10                   ; open INTC10 mask flag
MOV     TMC1,#00001000B         ; start timer (TM1)
EI
        :
        :
```




```
;************************************
;*      KEY_DATA CHECK & STORE      *
;************************************
;
        BF      KEYOK,$KD_END           ; check valid key
        CLR1    KEYOK                   ; clear valid key
        MOV     A,KEYCOD                ; read keycode
                :
                :
KD_END:
                :
                :
```

〔メ　モ〕

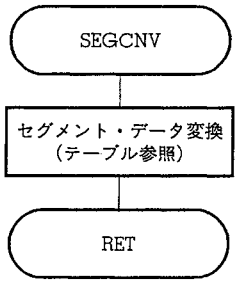
(b) 表示処理

このプログラム例では，表示エリアにセグメント出力データを転送することで，表示を変更します。

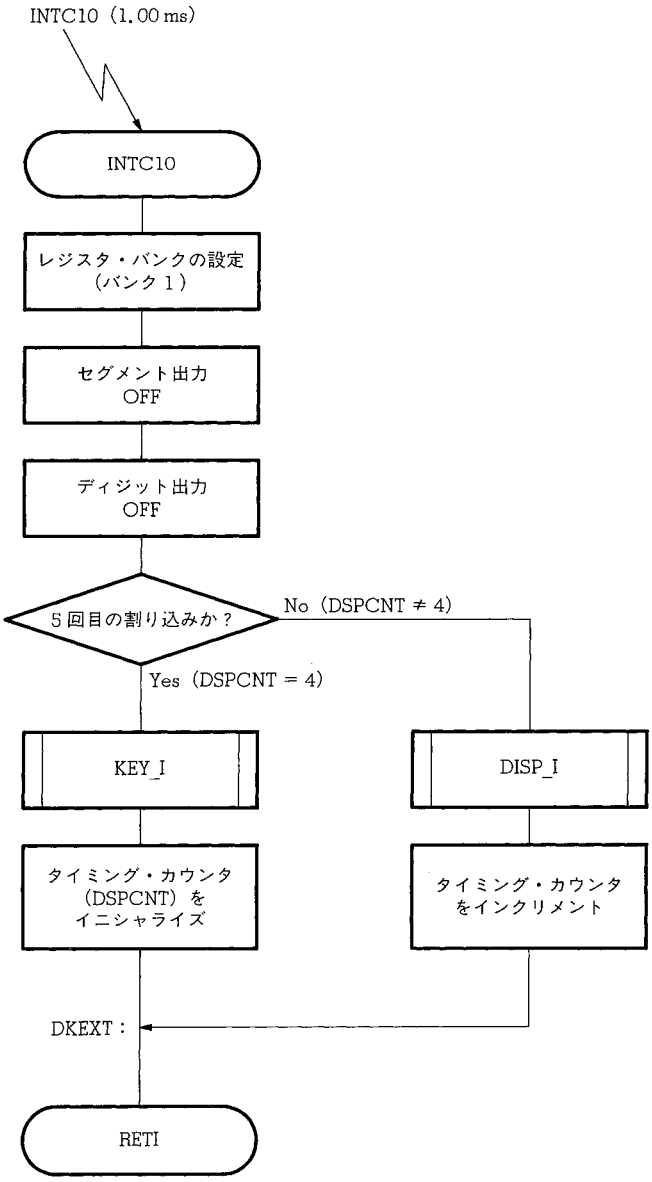
```
        EXTRN   D_INI, SEGCNV           ; package
        EXTRN   SEGOFF, DGTOFF, DSPCNT  ; parameter
        EXTRN   LED0, LED1, LED2, LED3  ; parameter
                :
                :
        CSEG
                :
                :
        MOV     P0, #LOW SEGOFF         ; initialize PORT0
        MOV     PM0, #00000000B         ; PORT0 <-- output port
        MOV     P3, #LOW DGTOFF         ; initialize PORT3
        MOV     PM3, #00001111B         ; P37-P34 <-- output port

        MOV     CRC1, #00000010B        ; enable clear TM1 by CR10
        MOV     PRM1, #00000011B        ; set prescaler(TM1:fclk/32)
        MOV     CR10, #188-1            ; set interval timer (1.00 ms)
        MOV     DSPCNT, #0              ; initialize display counter
        CALL    !D_INI                  ; initialize display area
        CLR1    CMK10                   ; open INTC10 mask flag
        MOV     TMC1, #00001000B        ; start timer (TM1)
        EI
                :
                :
;*************************************
;*      LED_DATA CHANGE & STORE      *
;*************************************
;
        MOV     A, #0
        CALL    !SEGCNV                 ; conversion segment data
        MOVW    HL, #LED0               ; store LED display new data
        XCH     A, [HL]
                :
                :
```

(8) フロー・チャート

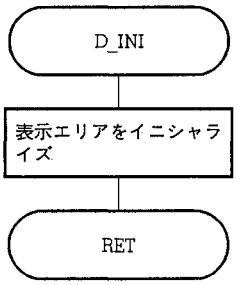
(a) セグメント・データ変更処理モジュール：SG_CONV

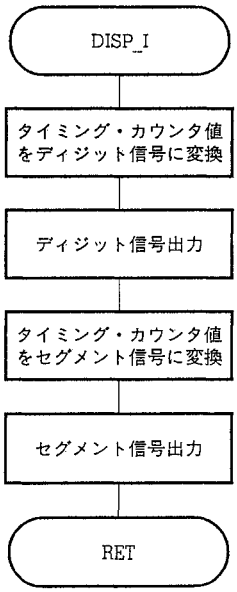


(b) 表示／キー入力共通モジュール：I_CMN



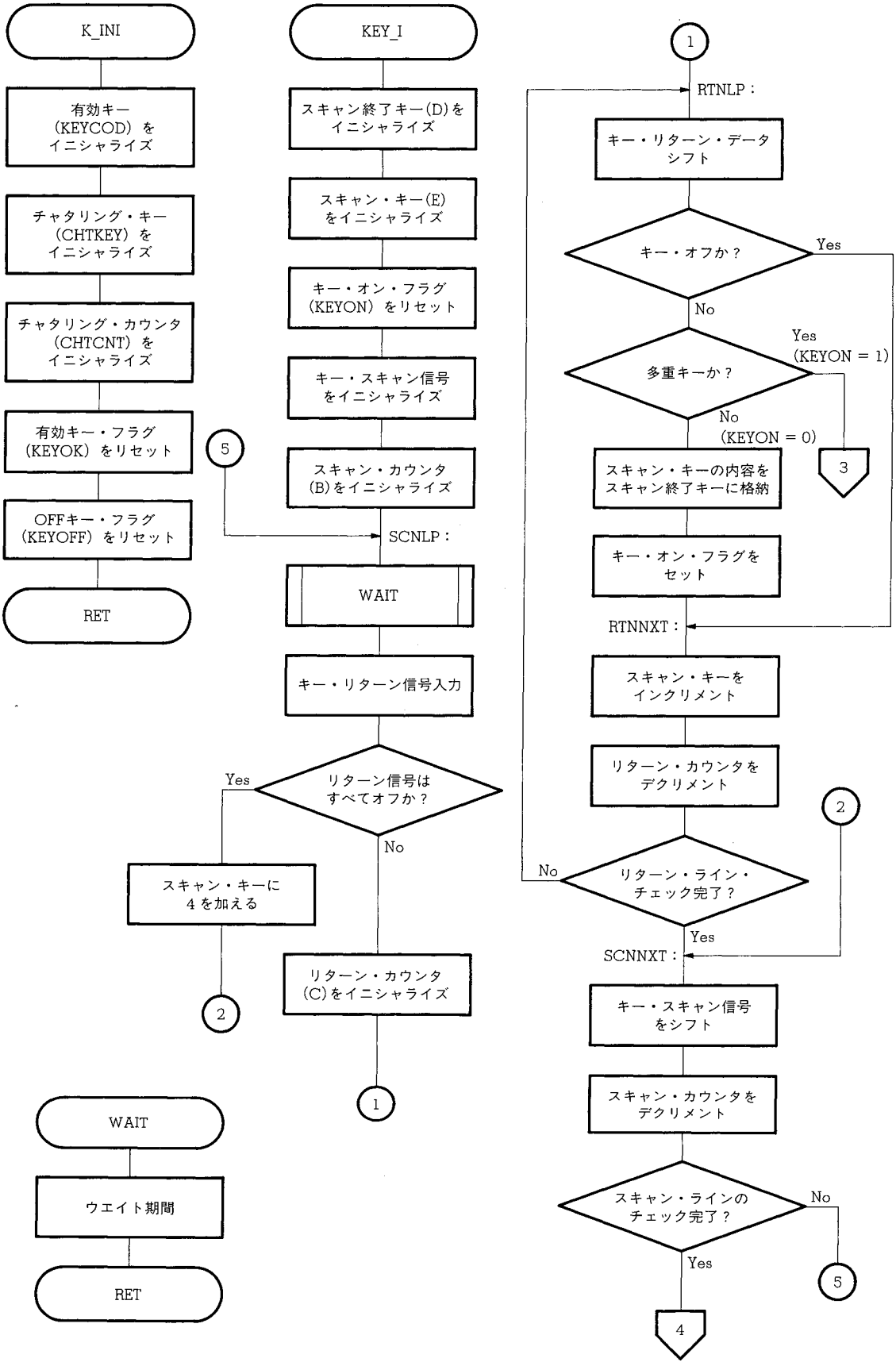
(c) 表示関連モジュール：I_DISP

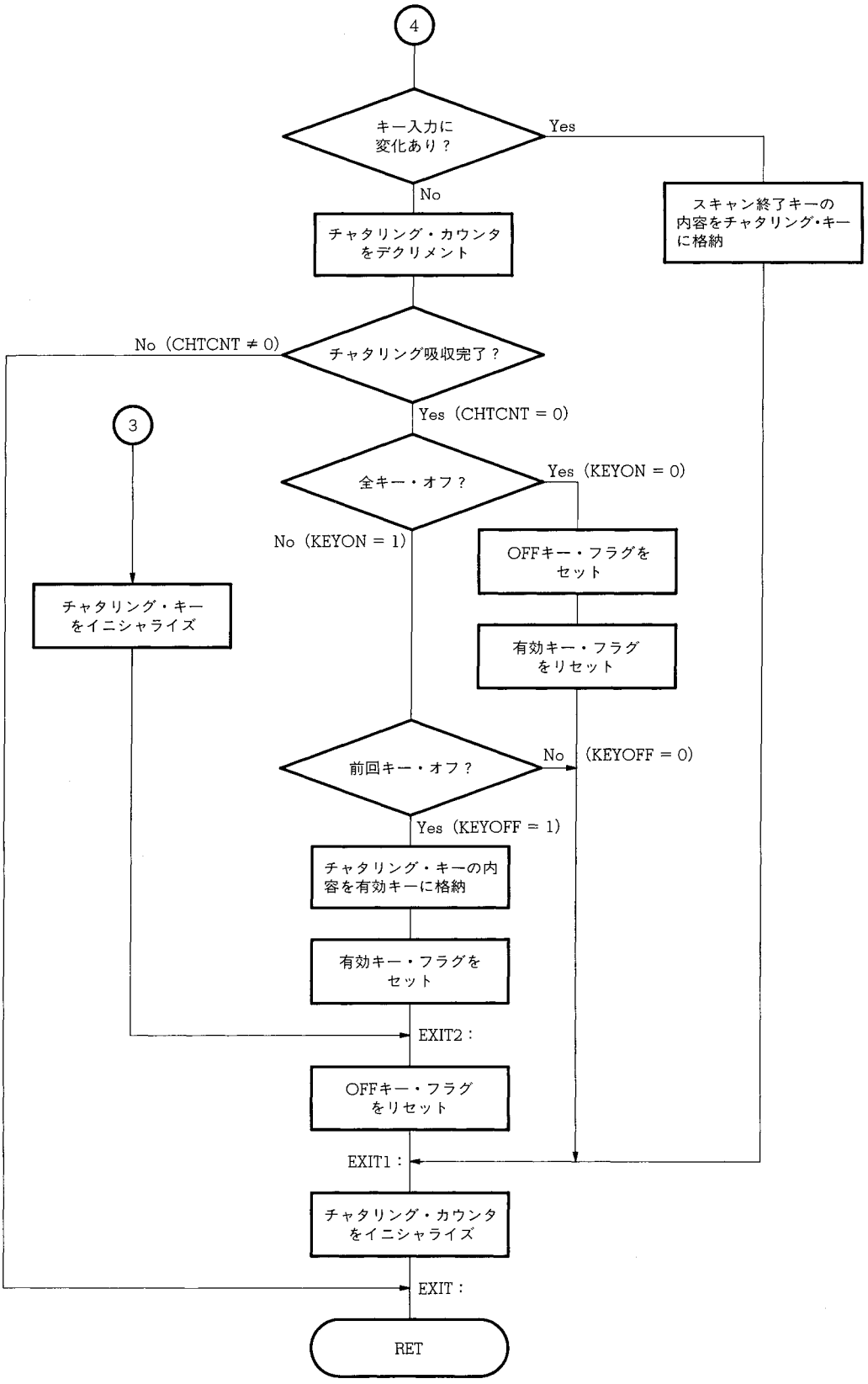




〔メ　モ〕

(d) キー入力関連モジュール：I_KEY





7
214
218A
224
234
244

(9) プログラム・リスト

今回のプログラム例では，割り込みベクタ・アドレスの設定などを行うために，モジュールの先頭でベクタ割り込み定義用のファイル（INTMS.DEF）をインクルードしています（**(b) 表示／キー入力共通モジュール：I_CMN**）。

(a) セグメント・データ変換処理モジュール：SG_CONV

```
        NAME    SG_CONV
;
;****************************************
;*      LED SEGMENT DATA CONVERSION     *
;****************************************
;
        PUBLIC  SEGCNV                  ; package

SGCONV  CSEG
SEGCNV:                                 ; conversion segment data
        MOV     A,SEGTBL[A]
        RET

SEGTBL:                                 ; input code profile
        DB      01110111B               ; (00) 0
        DB      00010100B               ; (01) 1
        DB      10110011B               ; (02) 2
        DB      10110110B               ; (03) 3
        DB      11010100B               ; (04) 4
        DB      11100110B               ; (05) 5
        DB      11100111B               ; (06) 6
        DB      01110100B               ; (07) 7
        DB      11110111B               ; (08) 8
        DB      11110110B               ; (09) 9
        DB      11110101B               ; (0A) A
        DB      11000111B               ; (0B) b
        DB      01100011B               ; (0C) C
        DB      10010111B               ; (0D) d
        DB      11100011B               ; (0E) E
        DB      11100001B               ; (0F) F
;
        END
```

(b) 表示／キー入力共通モジュール：I_CMN

```
$       IC(INTMS.DEF)                   ; interrupt define macro
        NAME    I_CMN
;
;*****************************************
;* KEY_INPUT / LED DISPLAY               *
;*               COMMON PACKAGE          *
;*                           BY INTERRUPT *
;*****************************************
;
        PUBLIC  SEGOFF,DGTOFF,DSPCNT    ; parameter
        EXTRN   DISP_I,KEY_I            ; package
        EXTRN   SEGNO                   ; parameter
;
;       --- symbol define ---
;
SEGOFF  EQU     00000000B               ; segment output off data
DGTOFF  EQU     11110000B               ; digit output off data
;
;       --- for work ---
;
KD_TIMS DSEG    SADDR
DSPCNT: DS      1                       ; timing counter of display & key_in
;
;       --- vector macro define ---
;
        CVENT10                         ; INTC10 vector macro
;
;*********************************************************************
;*      INTC10 INTERRUPT  --> KEY_DATA / LED DISPLAY SELECT          *
;*********************************************************************
;
ICMN    CSEG
INTC10:                                 ; INTC10 (1.00 ms)
        SEL     RB1                     ; change register bank (1)
        MOV     P0,#SEGOFF              ; segment output off (00000000B)
        OR      P3,#DGTOFF              ; digit output off (1111XXXXB)
        CMP     DSPCNT,#LOW SEGNO       ; select KEY_I or DISP_I
        BE      $KEYSEL
        CALL    !DISP_I                 ; LED display_output
        INC     DSPCNT                  ; increment display counter
        BR      DKEXT
KEYSEL:
        CALL    !KEY_I                  ; key_input
        MOV     DSPCNT,#0               ; initialize display counter
DKEXT:
        RETI
;
        END
```

7
214
218A
224
234
244

（c）表示関連モジュール：I_DISP

```
        NAME    I_DISP
;
;**********************************
;*      LED DISPLAY               *
;*                BY INTERRUPT    *
;**********************************
;
        PUBLIC  D_INI,DISP_I            ; package
        PUBLIC  LED0,LED1               ; parameter
        PUBLIC  LED2,LED3,SEGNO         ; parameter
        EXTRN   DSPCNT                  ; parameter
;
;       --- for work ---
;
DISPS1  DSEG
DSPDT:                                  ; segment data top address
LED0:   DS      1                       ; No.0 LED display data address
LED1:   DS      1                       ; No.1
LED2:   DS      1                       ; No.2
LED3:   DS      1                       ; No.3
LEDEND:                                 ; segment data end address

DISPS2  DSEG    SADDR
DGTWRK: DS      1                       ; digit output work
;
;       --- symbol define ---
;
SEGNO   EQU     LEDEND-LED0             ; define segment No.
;
;**********************************
;*      LED DISPLAY INITIALIZE    *
;**********************************
;
IDISP   CSEG
D_INI:                                  ; initialize display area
        MOVW    HL,#DSPDT
        MOV     B,#SEGNO
        MOV     A,#00H
DSPCLR:
        MOV     [HL+],A
        DBNZ    B,$DSPCLR
        RET
```

```
;
;*******************************
;*      LED DISPLAY OUTPUT     *
;*******************************
;
DISP_I:
        MOV     A,DSPCNT                ; display counter -> digit data
                                        ;                 -> segment data
        MOV     B,A
        MOV     A,DGT_D[B]
        MOV     DGTWRK,A
        AND     P3,DGTWRK               ; output digit data
        MOV     A,DSPDT[B]
        MOV     P0,A                    ; output segment data
        RET

DGT_D:
        DB      11101111B               ; LED0 digit data
        DB      11011111B               ; LED1
        DB      10111111B               ; LED2
        DB      01111111B               ; LED3
;
        END
```

7
214
218A
224
234
244

(d) キー入力関連モジュール：I_KEY

```
        NAME    I_KEY
;
;**********************************
;*      KEY_INPUT                 *
;*              BY INTERRUPT      *
;**********************************
;
        PUBLIC  K_INI,KEY_I             ; package
        PUBLIC  KEYOK,KEYCOD            ; parameter
;
;       --- symbol define ---
;
SCNLN   EQU     4                       ; scan line number
SCNINI  EQU     11111110B               ; scan line initial data
RETLN   EQU     4                       ; return line number
RLMSK   EQU     00001111B               ; return data masking
CHTINI  EQU     4                       ; chattering counter initial number
KEYON   EQU     X.7                     ; key on check flag
ALLOFF  EQU     0FFH                    ; keycode off data
NGKEY   EQU     0EEH                    ; multi-key data
;
;       --- for work ---
;
        BSEG
KEYOK   DBIT                            ; valid key flag
KEYOFF  DBIT                            ; all off key flag

KEYS    DSEG    SADDR
KEYCOD: DS      1                       ; last keycode
CHTKEY: DS      1                       ; chattering key
CHTCNT: DS      1                       ; chattering counter
;
;**********************************
;*      KEY INPUT INITIALIZE      *
;**********************************
;
IKEY    CSEG
K_INI:
        MOV     KEYCOD,#ALLOFF          ; initialize last keycode
        MOV     CHTKEY,#NGKEY           ; initialize chattering key
        MOV     CHTCNT,#CHTINI          ; initialize chattering counter
        CLR1    KEYOK                   ; initialize valid key flag
        CLR1    KEYOFF                  ; initialize off key flag
        RET
```

```
;
;**********************
;*      KEY INPUT     *
;**********************
;
KEY_I:
        MOV     D,#ALLOFF               ; initialize Dreg last scan-key
        MOV     E,#00H                  ;            Ereg scan-key
        CLR1    KEYON                   ; clear key on check flag
        MOV     P0,#SCNINI              ; initialize key scanning signal
        MOV     B,#SCNLN
SCNLP:                                  ; start scan loop
        CALL    !WAIT                   ; wait (about 7.5μs )

        MOV     A,P2                    ; input return line
        OR      A,#RLMSK                ; mask return line
        CMP     A,#ALLOFF
        BNE     $RLCHK                  ; all key off ?
        MOV     A,#RETLN                ; then make next scan line's key code
        ADD     E,A
        BR      SCNNXT
RLCHK:
        MOV     C,#RETLN                ; check return line
RTNLP:
        SHL     A,1                     ; key off ?
        BC      $RTNNXT
        BF      KEYON,$SETKEY           ; multi-key ?
        MOV     CHTKEY,#NGKEY           ; initialize chattering key
        BR      EXIT2
SETKEY:
        MOV     D,E                     ; set last scan-key
        SET1    KEYON                   ; set key on check flag
RTNNXT:
        INC     E                       ; make next return line's key code
        DBNZ    C,$RTNLP                ; check all return line
SCNNXT:
        SET1    CY                      ; shift scan line
        ADDC    P0,P0
        DBNZ    B,$SCNLP                ; check all scan line ?

        MOV     A,D                     ; change key ?
        CMP     A,CHTKEY
        BE      $CHTRNG
        MOV     CHTKEY,A                ; set chattering key
        BR      EXIT1
```

7
214
218A
224
234
244

```
CHTRNG:
        DBNZ    CHTCNT,$EXIT            ; absorb chattering ?

        BT      KEYON,$CHKKEY           ; all key off ?
        SET1    KEYOFF                  ; off key in ! set KEYOFF
        CLR1    KEYOK                   ;               clear KEYOK
        BR      EXIT1
CHKKEY:
        BF      KEYOFF,$EXIT1           ; if invalid key , branch 'EXIT1'
        MOV     KEYCOD,CHTKEY           ; valid key in ! set  last keycode
        SET1    KEYOK                   ;                set  KEYOK
EXIT2:
        CLR1    KEYOFF                  ; clear KEYOFF
EXIT1:
        MOV     CHTCNT,#CHTINI          ; initialize chattering counter
EXIT:
        RET
;
;*****************
;*      WAIT     *
;*****************
;
WAIT:                                   ; wait (about 7.5μs)
        NOP
        NOP
        NOP
        RET
;
        END
```

7.2　マクロ・サービスによるプログラム例

(1) 動作概要

８ビット・タイマ／カウンタ１を用いた「表示／キー入力」動作をマクロ・サービスにより行います。マクロ・サービスを３本同時に使用することによって，セグメント信号出力，ディジット信号出力，キー・スキャン信号出力，キー・リターン信号入力の４つの動作を実現します。

表７-９にマクロ・サービスと各動作の対応を示します。

この応用例をμPD78224シリーズで使用する場合，INTP0端子をINTP4端子に，キー・リターン信号入力を他のポートに変更します。

表７-９　マクロ・サービスと各動作の対応

動　　作	ポート	割り込み要求	マクロ・サービス
セグメント信号出力	P00-P07	INTC10	タイプC（MPTインクリメント）
キー・スキャン信号出力	P00-P03		
ディジット信号出力	P34-P37	INTC11	タイプB（メモリ→SFR）
INTP0発生信号出力[注]	P33		
キー・リターン信号入力	P24-P27	INTP0	タイプB（SFR→メモリ）

注　キー・リターン信号入力用トリガINTP0を発生させるために，図７-８の回路のようにしてエッジ検出を行います。

図７-８　INTP0発生回路とタイミング

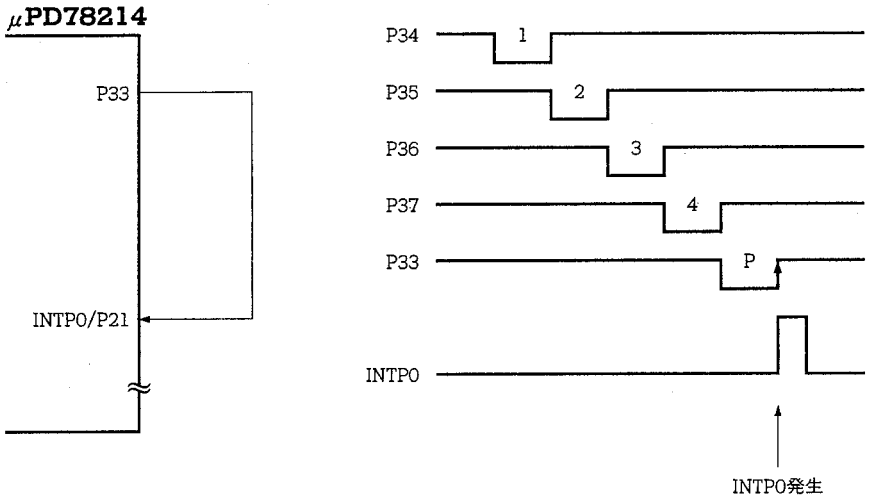
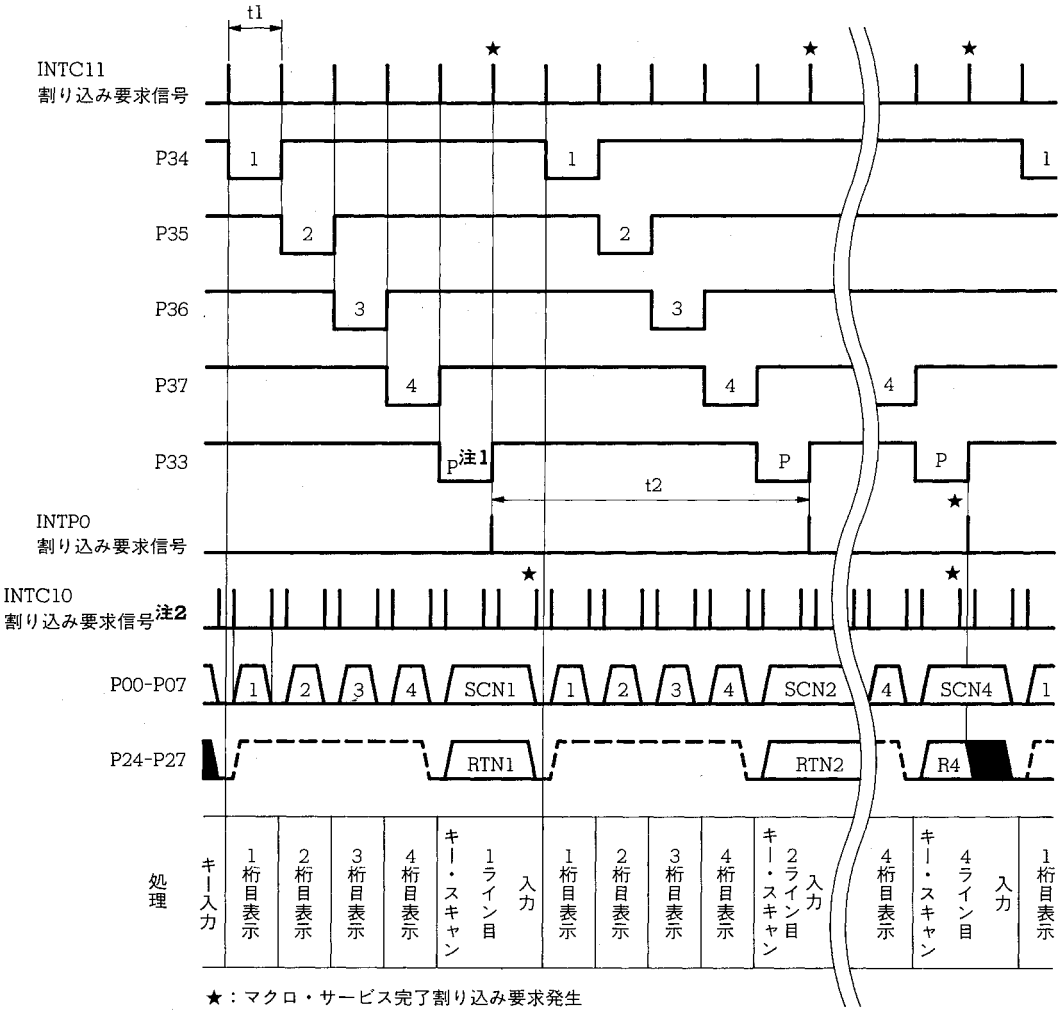


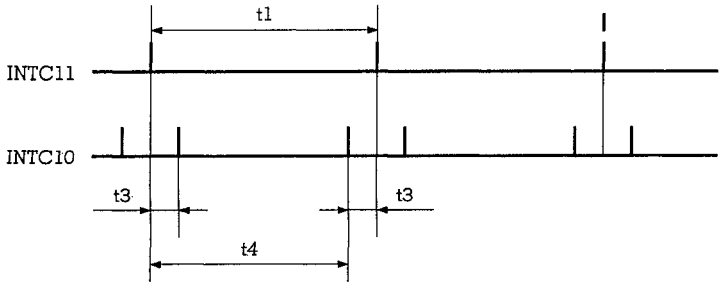
図7-9にマクロ・サービスによる表示とキー入力のタイミングを示します。

図7-9　表示とキー入力のタイミング（マクロ・サービス）



注1. P33の立ち上がりでINTP0を生成し，これによりキー入力のトリガをかけます。キー・スキャン4ラインすべて入力したところでINTP0マクロ・サービス完了割り込みを発生させ，その次のINTC11割り込み要求発生までの期間でキー処理（キー・コード生成，キーの多重押しのチェック，チャタリング吸収，ロック・アウト処理）を行います（図のぬりつぶした部分）。

注2.



t1：各ディジット信号およびINTP0生成用信号の出力期間です。

t1 ＝ タイマのカウント・クロック（f_{CLK}/32）周期×（CR11設定値＋1）

＝ 5.3 μs（分解能）×（(40－1)＋1）

≒ 213.3 μs

t2：1スキャン・ラインのキー入力から次のスキャン・ラインのキー入力までの1スキャン・サイクル期間です。

t2 ＝ t1×6

≒ 1.28 ms

t3：ディジット信号出力からセグメント信号出力までのブランキング期間および，セグメント信号出力後の次のディジット信号出力までのブランキング期間です。

t3 ＝ タイマのカウント・クロック（f_{CLK}/32）周期×（タイプC用タイミング・データ（表示ON））

＝ 5.3 μs（分解能）×5

≒ 26.67 μs

t4：ディジット信号出力から次のブランキング・データ出力までの期間です。

t4 ＝ タイマのカウント・クロック（f_{CLK}/32）周期×（タイプC用タイミング・データ（表示OFF））

＝ 5.3 μs（分解能）×35

≒ 186.67 μs

7
214
218A
224
234
244

次に各割り込み要求ごとの動作の詳細を示します。

[INTC10]

12回のマクロ・サービスごとにマクロ・サービス完了割り込みを発生させます。

（1回目）LED 1桁目のセグメント出力　ON
（2回目）LED全桁　　〃　　OFF（ブランキング時間）
（3回目）LED 2桁目　　〃　　ON
（4回目）LED全桁　　〃　　OFF（ブランキング時間）
（5回目）LED 3桁目　　〃　　ON
（6回目）LED全桁　　〃　　OFF（ブランキング時間）
（7回目）LED 4桁目　　〃　　ON
（8回目）LED全桁　　〃　　OFF（ブランキング時間）
（9，10，11回目）キー・スキャン信号出力[注]
（12回目）LED全桁のセグメント出力　　OFF（ブランキング時間）

注　4スキャン・ラインのうち1スキャン・ラインのみをアクティブにするキー・スキャン信号を作ります。また，INTC10マクロ・サービス完了割り込みごとにキー・スキャン信号をシフトします。

[INTC11]

6回のマクロ・サービスごとにマクロ・サービス完了割り込みを発生させます。

（1回目）LED 1桁目のディジット出力　ON
（2回目）LED 2桁目の　　〃　　〃
（3回目）LED 3桁目の　　〃　　〃
（4回目）LED 4桁目の　　〃　　〃
（5，6回目）INTP0（キー・リターンのタイミング）発生用信号出力

[INTP0]

4回のマクロ・サービスごとにマクロ・サービス完了割り込みを発生させます。

（1回目）キー・リターン信号入力（1スキャン・ライン目）
（2回目）　〃　（2スキャン・ライン目）
（3回目）　〃　（3スキャン・ライン目）
（4回目）　〃　（4スキャン・ライン目）

各マクロ・サービス完了割り込み発生までの時間は，次のようにします。

(INTC10)	t1×5+t4	= 1.25 ms（最初のみ）
	t1×6	= 1.28 ms（2回目以降）
(INTC11)	t1×6	= 1.28 ms
(INTP0)	(t1×6)×4	= 5.12 ms

INTP0のマクロ・サービス完了割り込み発生周期が，キー・スキャン・サイクル・タイムとイコールになります。この間にLED 1桁の表示が4回行われますので表示デューティは次のようになります。

$$\text{表示デューティ} = \frac{Tn}{(Tn+t)\times n} = \frac{4\times(t4-t3)}{5.12\,\text{ms}} = 1/8$$

また，4回のINTP0割り込み要求発生で，1回のキー入力処理が完了するので，チャタリング吸収時間は1サイクル分の時間のズレ[注]を考慮して次のようになります。

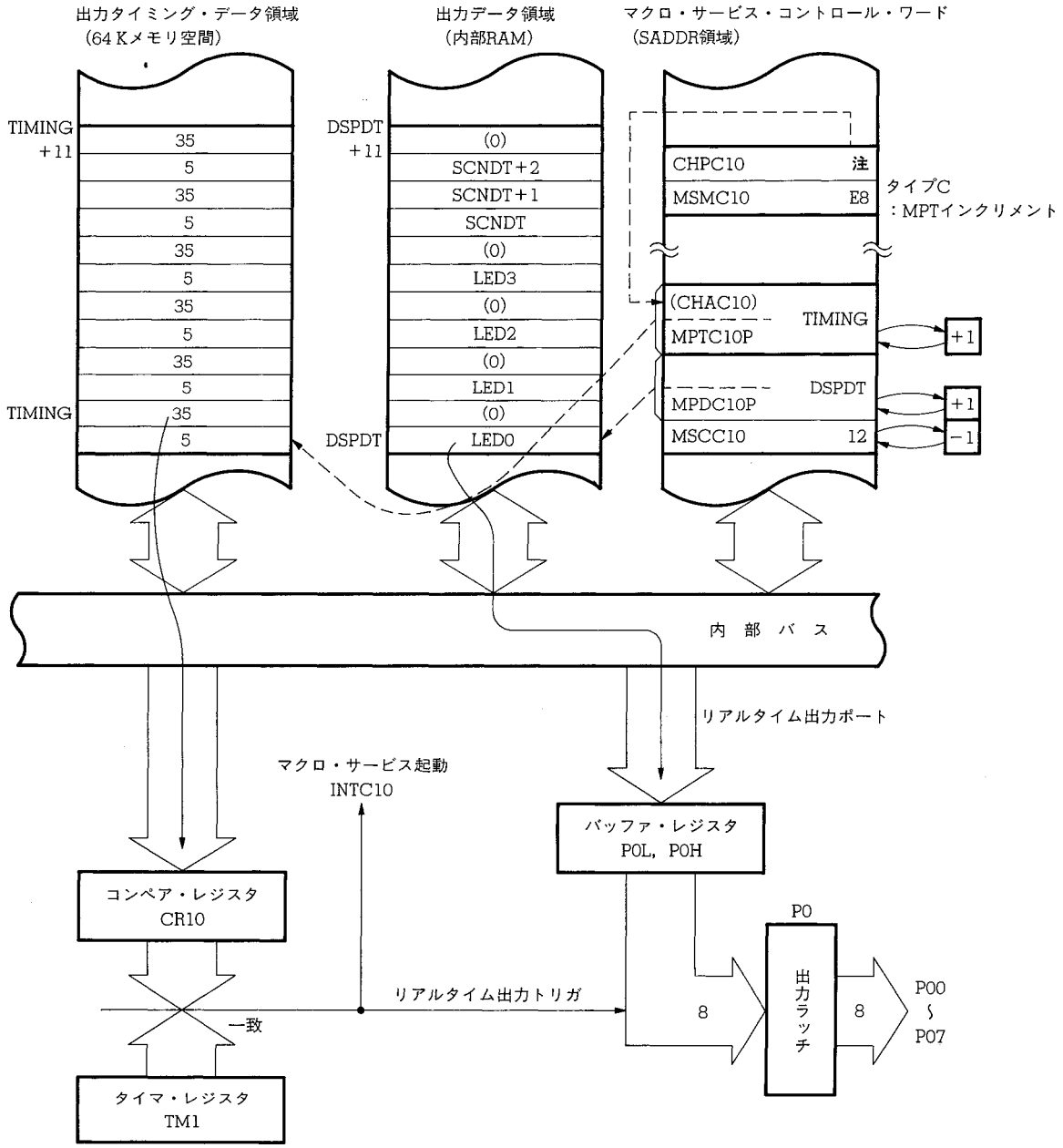
(t1×6)×4×4サイクル = 20.48 ms（Min.）
(t1×6)×4×5サイクル = 25.60 ms（Max.）

注　1つのサイクル中のどこでキー入力されるかにより，最大1サイクル分のズレが生じます。

7
214
218A
224
234
244

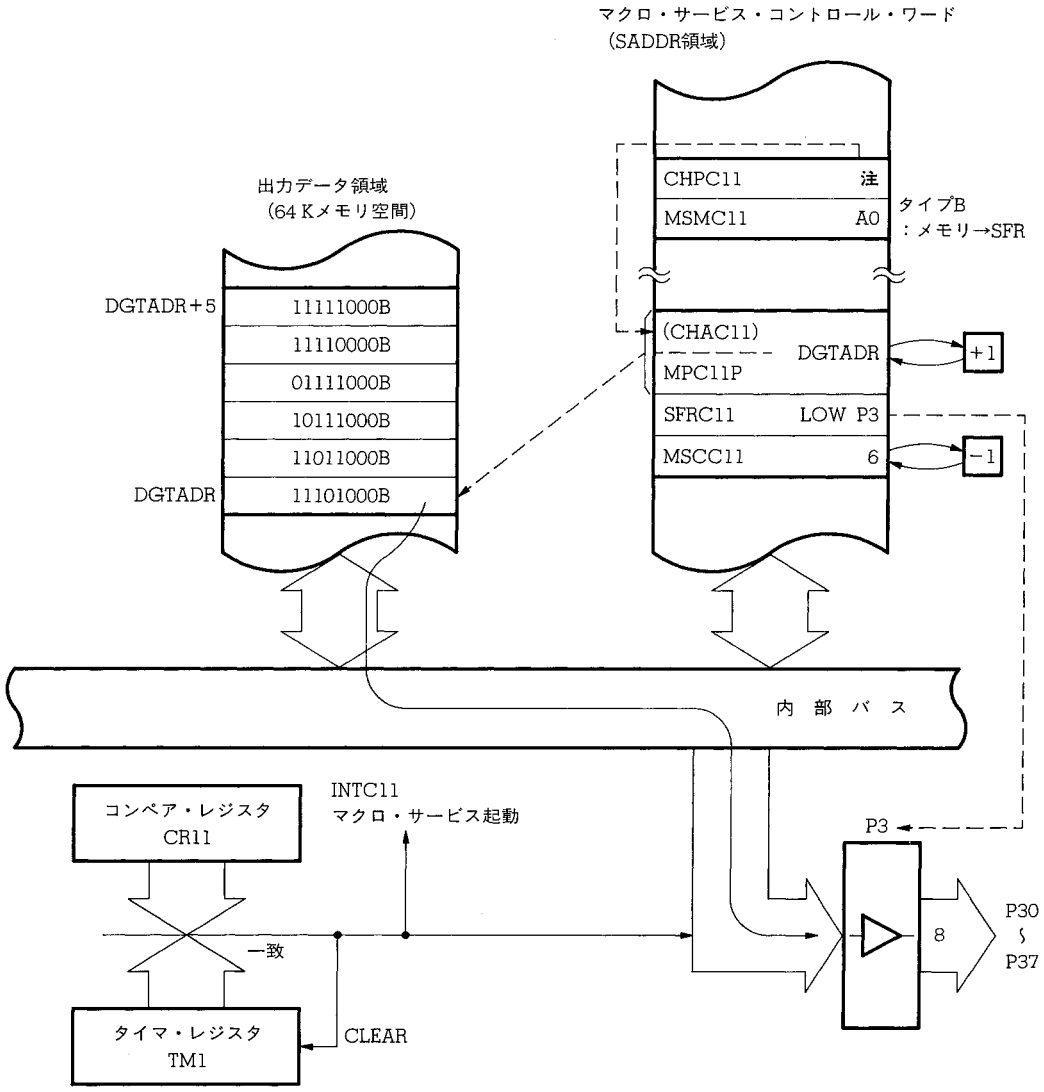
図7－10にマクロ・サービスによるセグメント／スキャン信号出力(INTC10)，図7－11にディジット信号出力（INTC11），図7－12にマクロ・サービスによるキー・リターン信号入力（INTP0）のメモリ・マップと動作ブロック図を示します。

図7－10　マクロ・サービスによるセグメント／スキャン信号出力（INTC10）



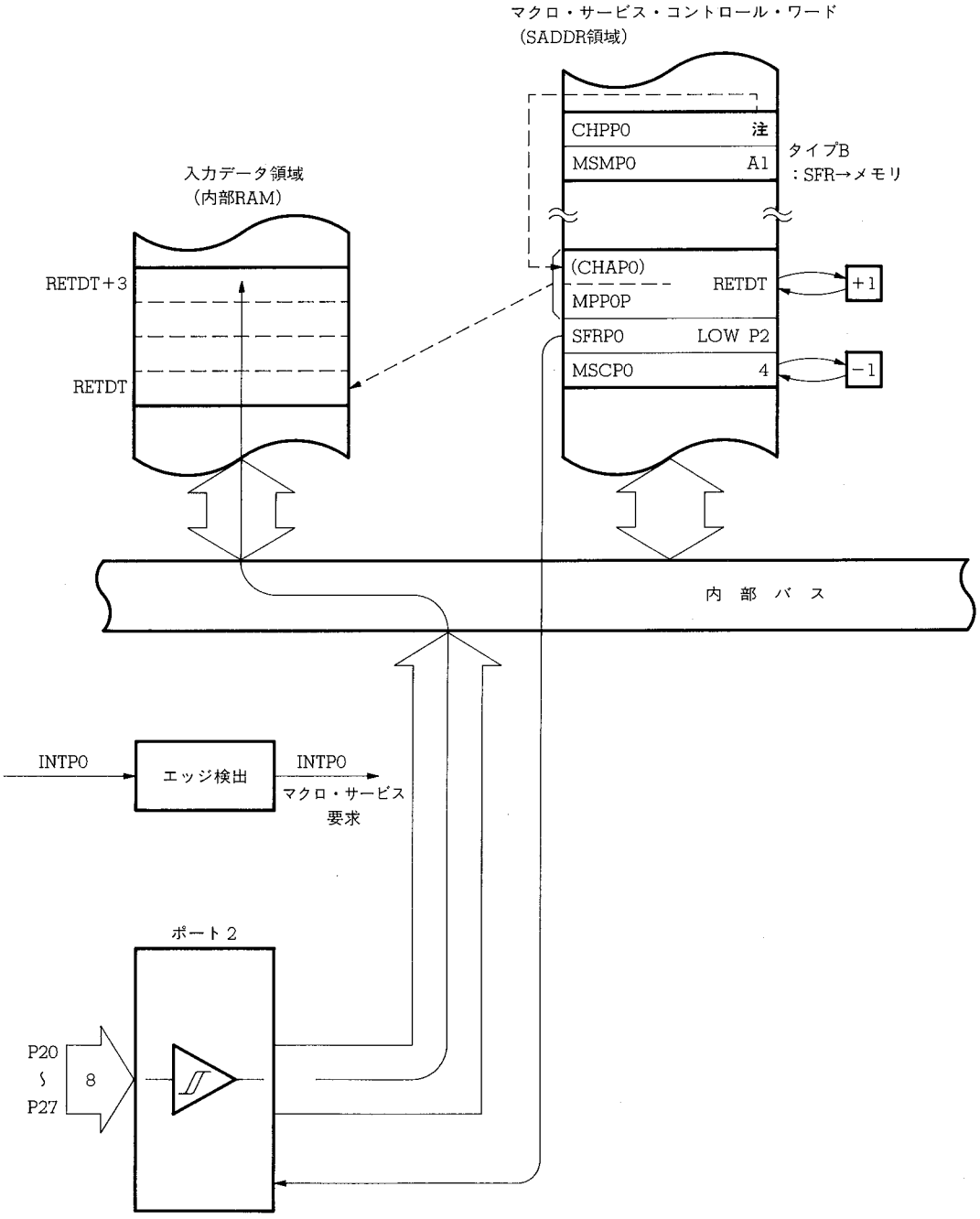
注　CHAC10（＝MPTC10P＋1）の下位バイト

図7-11　マクロ・サービスによるディジット信号出力（INTC11）



注　CHAC11（=MPC11P+1）の下位バイト

図7－12　マクロ・サービスによるキー・リターン信号入力（INTP0）



注　CHAP0（＝MPP0P＋1）の下位バイト

(2) プログラム説明

(a) モジュール説明

表7－10にこの応用例のモジュール一覧を示します。

表7－10　モジュール一覧

モジュール	レーベル	処　　理
セグメント・データ変換 ：SG_CONV	SEGCNV	LEDセグメント・データ変換処理
表示／キー入力関連 ：KD_MCR	KD_INI	INTC10/C11/P0イニシャライズ処理
	INTC10	INTC10マクロ・サービス完了割り込み処理
	INTC11	INTC11マクロ・サービス完了割り込み処理
	INTP0	INTP0マクロ・サービス完了割り込み処理

(b) モジュール処理概要

(i) セグメント・データ変換処理モジュール：SG_CONV

（このモジュールは，**7.1　ベクタ割り込みによるプログラム例**と共通に使用できます）。

① LEDセグメント・データ変換処理［レーベル名称：SEGCNV］

ここでは，アキュームレータの内容(0H-FH)をテーブル参照によりLEDセグメント・データに変換する処理を行っています。

(ii) 表示／キー入力処理モジュール：KD_MCR

① INTC10/C11/P0イニシャライズ処理［レーベル名称：KD_INI］

ここでは，INTC10/C11/P0マクロ・サービス関連のイニシャライズおよび，表示／キー入力関連ワーク・エリアのイニシャライズを行っています（表**7－11**参照）。

② INTC10マクロ・サービス完了割り込み（ベクタ割り込み）処理［レーベル名称：INTC10］

ここでは，キー・スキャン・データの切り替えおよび，INTC10マクロ・サービスの再設定を行っています。

7
214
218A
224
234
244

③ INTC11マクロ・サービス完了割り込み（ベクタ割り込み）処理

［レーベル名称：INTC11］

ここでは，INTC11マクロ・サービスの再設定を行っています。

④ INTP0マクロ・サービス完了割り込み（ベクタ割り込み）処理

［レーベル名称：INTP0］

ここでは次の処理を行っています。

- ●キー・コード生成
- ●キーの多重押しのチェック
- ●チャタリング吸収
- ●ロック・アウト処理
- ●INTP0マクロ・サービスの再設定

次に詳細を示します。

●キー・コード生成

キー・リターン・ポインタ（DEレジスタ）の指すキー・データを1ビット・シフトし，そのときのキャリーの状態によりキーが押されたことを判断し，スキャン・キー（Lレジスタ）およびスキャン終了キー（Hレジスタ）を用いて生成します。

●キーの多重押しのチェック

キー・コード生成中，キーが押されたと判断したときにキー・オン・フラグ（KEYON：X.7）をセットし，このフラグの状態により多重キーが入力されたことを判断し，キー入力を無効とします。

●チャタリング吸収

スキャン終了キー（Hレジスタ）とチャタリング・キー（CHTKEY）を比較し，4回内容が一致（チャタリング・カウンタ（CHTCNT）によりカウント）すれば，チャタリングが吸収したとみなします。

●ロック・アウト処理

チャタリング吸収後，キー・オン・フラグ（KEYON）がリセットされている場合は全キーがOFFとなっているので，OFFキー・フラグ（KEYOFF）をセットし，このフラグがセットされているときだけキーを有効にします（チャタリング・キー（CHTKEY）の内容を有効キー（KEYCOD）に格納し，有効キー・フラグ（KEYOK）をセット）。

(c) 使用するワーク・エリア

この応用例では，表示／キー入力処理用で35バイト＋3ビットのワーク・エリアを使用します。表7－11，表7－12にワーク・エリアの一覧を示します。

表7－11　表示／キー入力関連ワーク・エリア1

[表示／キー入力処理モジュール：KD_MCR]

ワーク・エリア名称	配置領域[注]	用　途	バイト数	初期値
DSPDT (LED0-LED3)	RAM	表示データ格納エリア（LED 4桁分） （1バイトおきにダイナミック表示のブランキング期間用としてオフ・データ（00H）を格納）	8	すべて 00H
スキャン・データ SCNDT	RAM	キー・スキャン・データ格納エリア （1スキャン・ライン分） （最後の1バイトにオフ・データを格納）	4	FEH （3バイト） 00H （1バイト）
スキャン・カウンタ SCNCNT	SADDR	スキャン・ライン・シフト・カウンタ	1	04H
RETDT	RAM	キー・リターン・データ格納エリア （4スキャン・ライン分）	4	FFH
有効キー KEYCOD	SADDR	有効となったキー・コードを格納	1	
チャタリング・キー CHTKEY		チャタリング吸収中のキー・コードを格納		EEH
チャタリング・カウンタ CHTCNT		チャタリング回数カウンタ		04H
OFFキー・フラグ KEYOFF		オフ・チャタリングの吸収完了を確認	1ビット	0
有効キー・フラグ KEYOK		有効キーの入力を確認		
MPTC10P, MPDC10P, MSCC10		INTC10　マクロ・サービス・チャネル （タイプC）	5	TIMING DSPDT 12
MPC11P, SFRC11, MSCC11		INTC11　マクロ・サービス・チャネル （タイプB）	4	DGTADR LOW P3 6
MPP0P, SFRP0, MSCP0		INTP0　マクロ・サービス・チャネル （タイプB）		RETDT LOW P2 4

7
214
218A
224
234
244

注　配置領域の名称は，次の意味を示します。

RAM　：バンク0内の任意のRAM領域

SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

表7-12　表示／キー入力関連ワーク・エリア2

［表示／キー入力処理モジュール：KD_MCR］

ワーク・エリア名称	配置領域[注]	用　途	バイト数	初期値
キー・オン・フラグ KEYON（X.7）	レジスタ・バンク1	キー入力の検出を状態として保持	1ビット	0
スキャン終了キー H		キー・スキャンにより検出したキー・コードを格納	1	FFH
スキャン・キー L		キー・スキャン中のキー・コードを格納		00H

注　配置領域の名称は，次の意味を示します。

レジスタ・バンク1：汎用レジスタ領域のバンク1

(3) 入出力パラメータ

この応用例のパラメータはすべて可変値です。表7-13に入力パラメータ，表7-14に出力パラメータの一覧を示します。

表7-13　入力パラメータ

モジュール	パラメータ	用　途
セグメント・データ変換：SG_CONV	Aレジスタ（バンク0）	表示データ（00H-0FH）を入力

表7－14　出力パラメータ

モジュール	パラメータ	用　　途
セグメント・データ変換 ：SG_CONV	Aレジスタ （バンク0）	LEDセグメント・データを出力
表示／キー入力関連 ：KD_MCR	LED0, LED1, LED2, LED3	表示データ（4バイト）格納エリアの各アドレス
	KEYOK	有効キー・フラグ（有効なキー入力があった場合にセット）
	KEYCOD	チャタリングの吸収を完了し，有効となったキー・コード（KEYOK＝1のとき，このキー・コードが有効）

(4) 使用レジスタ

表7－15に使用レジスタの一覧を示します。

表7－15　使用レジスタ

モジュール	レーベル	使用レジスタ	バンク
セグメント・データ変換 ：SG_CONV	SEGCNV	A	0
表示／キー入力関連 ：KD_MCR	KD_INI	AX, B, HL	0
	INTC10	A	1
	INTC11	なし	—
	INTP0	A, X[注] B, C, DE, H, L	1

注　Xレジスタは1ビットのみ使用（X.7）

(5) 使用スタック

このプログラム例のサブ・ルーチンおよび割り込み処理で使用するスタック・サイズは，5バイトです（ネスティング：2レベル）。

(6) モード・レジスタ設定例

ポート0モード・レジスタ

	7	6	5	4	3	2	1	0
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00
設定例	0	0	0	0	0	0	0	0

（リセットで00H）

PM0n	P0n端子入出力モード指定（n＝0－7）
00H	出力モード（出力バッファ・オン）
FFH	ハイ・インピーダンス状態（出力バッファ・オフ）
上記以外	設定禁止

ポート3モード・レジスタ

	7	6	5	4	3	2	1	0
PM3	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30
設定例	0	0	0	0	0	1	1	1

（リセットでFFH）

PM3n	P3n端子入出力モード指定（n＝0－7）
0	出力モード（出力バッファ・オン）
1	入力モード（出力バッファ・オフ）

キャプチャ／コンペア・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
CRC1	0	0	0	0	CLR11	CM	CLR10	0	（リセットで00H）
設定例	0	0	0	0	1	0	0	0	

CLR10	TM1とCR10レジスタの内容一致でのクリア動作
0	禁止
1	許可

CLR11	CM	CR11レジスタの動作指定	TM1のクリア動作
0	0	コンペア・レジスタ	禁止
1	0		許可（TM1とCR11レジスタの内容一致のとき）
0	1	キャプチャ・レジスタ	禁止
1	1		許可（TM1の内容をCR11レジスタにキャプチャしたとき）

7
214
218A
224
234
244

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	0	0	0	0	0	1	1	

8ビット・タイマ／カウンタ1

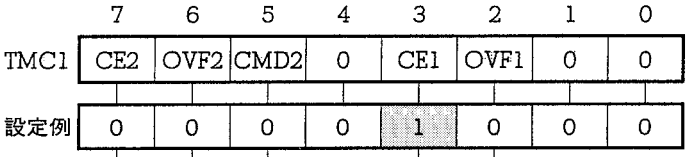
PRS12	PRS11	PRS10	カウント・クロック［Hz］の指定
0	0	0	f_{CLK}/16
0	0	1	
0	1	0	
0	1	1	f_{CLK}/32
1	0	0	f_{CLK}/64
1	0	1	f_{CLK}/128
1	1	0	f_{CLK}/256
1	1	1	f_{CLK}/512

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	カウント・クロック［Hz］の指定
0	0	0	0	f_{CLK}/16
0	0	0	1	
0	0	1	0	
0	0	1	1	f_{CLK}/32
0	1	0	0	f_{CLK}/64
0	1	0	1	f_{CLK}/128
0	1	1	0	f_{CLK}/256
0	1	1	1	f_{CLK}/512
1	1	1	1	外部クロック(CI)

備考　f_{CLK}：内部システム・クロック

タイマ・コントロール・レジスタ1



	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	0	0	0	0	1	0	0	0	

8ビット・タイマ／カウンタ1

OVF1	TM1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFHから00Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE1	TM1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

CMD2	TM2のカウント動作モード指定
0	通常モード
1	ワンショット・モード

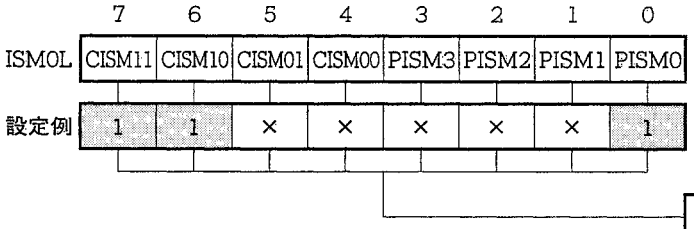
OVF2	TM2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFHから00Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE2	TM2のカウンタ動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

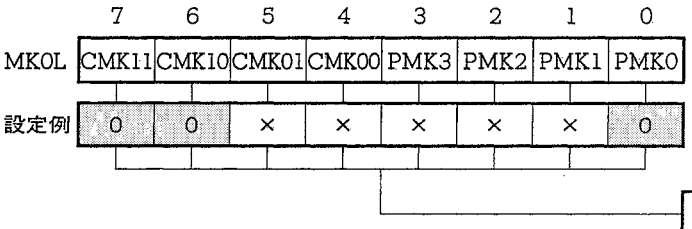
7
214
218A
224
234
244

割り込みサービス・モード・レジスタL



ISM	割り込みサービス・モード・フラグ
0	ベクタ割り込みで処理
1	マクロ・サービスで処理

割り込みマスク・レジスタL



MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

リアルタイム出力ポート・コントロール・レジスタ

	7	6	5	4	3	2	1	0	
RTPC	BYTE	0注	0注	POMH	EXTR	0注	0注	POML	（リセットで00H）
設定例	1	0	0	1	0	0	0	1	

POML	POLの機能指定
0	ポート・モード
1	リアルタイム出力ポート・モード

EXTR	INTP0によるバッファ・レジスタから出力ラッチへのデータ転送の許可
0	許可しない
1	許可する　BYTE=0：POLのみ転送される BYTE=1：POL/Hが転送される

POMH	POHの機能指定
0	ポート・モード
1	リアルタイム出力ポート・モード

BYTE	リアルタイム出力ポート・モードの動作モード
0	4ビット・セパレート・リアルタイム出力ポート
1	8ビット・リアルタイム出力ポート

注　ビット1，2，5，6には必ず“0”を書き込んでください。

7
214
218A
224
234
244

マクロ・サービス・モード・レジスタ（INTC10）

7	6	5	4	3	2	1	0	
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0	（リセットで00H）
設定例 1	1	1	0	1	0	0	0	

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0			リング制御なし	データ転送のみ POL
1	0	0	1				POH
1	0	1	0				自動加算つき POL
1	0	1	1				POH
1	1	0	0			リング制御つき	データ転送のみ POL
1	1	0	1				POH
1	1	1	0				自動加算つき POL
1	1	1	1				POH

マクロ・サービス・モード・レジスタ（INTC11）

7	6	5	4	3	2	1	0	
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0	（リセットで00H）
設定例 1	0	1	0	0	0	0	1	

CHT0	0	1	0	1			
CHT1	1	0	1				
CHT2	0	1	1				

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0		リング制御なし	データ転送のみ	POL
1	0	0	1				POH
1	0	1	0			自動加算つき	POL
1	0	1	1				POH
1	1	0	0		リング制御つき	データ転送のみ	POL
1	1	0	1				POH
1	1	1	0			自動加算つき	POL
1	1	1	1				POH

7
214
218A
224
234
244

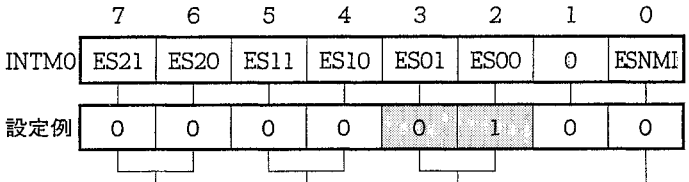
マクロ・サービス・モード・レジスタ（INTPO）

7	6	5	4	3	2	1	0	
CHT2	CHT1	CHT0	0	MOD3	MOD2	MOD1	MOD0	（リセットで00H）
設定例 1	0	1	0	0	0	0	1	

CHT0	0	1	0	1
CHT1	1	0	1	
CHT2	0	1	1	

MOD3	MOD2	MOD1	MOD0	タイプA	タイプB	タイプC	
						MPTL保持	MPTLインクリメント
0	0	0	0	メモリからSFRへデータ転送			
0	0	0	1	SFRからメモリへデータ転送			
1	0	0	0		リング制御なし	データ転送のみ	POL
1	0	0	1				POH
1	0	1	0			自動加算つき	POL
1	0	1	1				POH
1	1	0	0		リング制御つき	データ転送のみ	POL
1	1	0	1				POH
1	1	1	0			自動加算つき	POL
1	1	1	1				POH

外部割り込みモード・レジスタ0



ESNMI	NMI端子入力，検出エッジ指定
0	立ち下がりエッジ
1	立ち上がりエッジ

ES01	ES00	INTP0端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES11	ES10	INTP1端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES21	ES20	INTP2端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

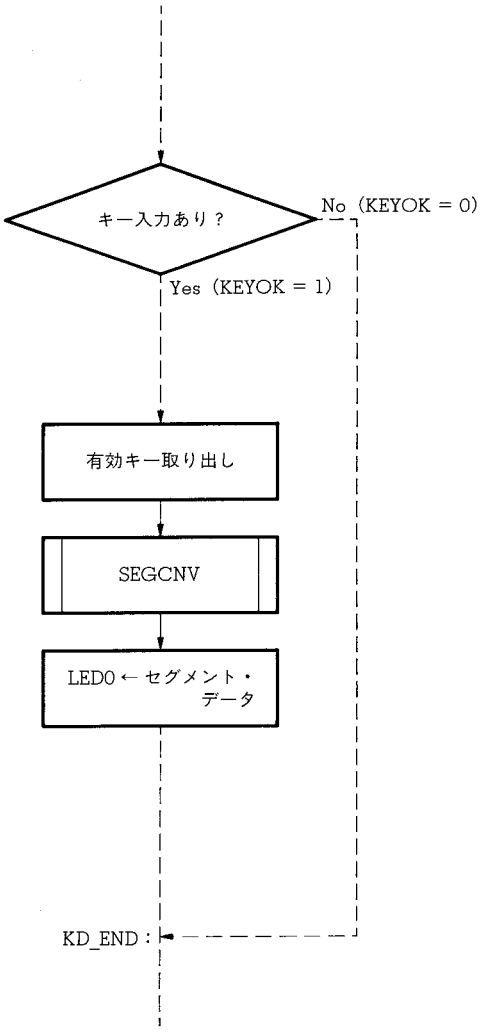
7
214
218A
224
234
244

(7) プログラム使用例

表示／キー入力処理のマクロ・サービスによる使用例を示します。この場合，有効キー・フラグを調べることによりキー入力の有無を確認し，キー入力ありの場合，有効キーのデータをセグメント出力データに変換し，表示エリアに転送することで表示を変更します。

プログラム例として次のように用います。

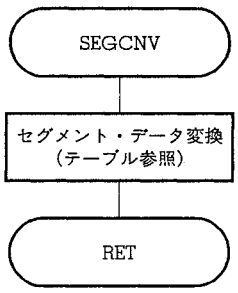
```
EXTRN   KD_INI,SEGCNV                   ; package
EXTRN   KEYCOD,LED0,LED1,LED2,LED3      ; parameter
EXTBIT  KEYOK                           ; parameter
        :
        :
CSEG
        :
        :
MOV     P0,#00000000B           ; initialize PORT0
MOV     PM0,#00000000B          ; PORT0 <-- output port
MOV     P3,#11111000B           ; initialize PORT3
MOV     PM3,#00000111B          ; P37-P33 <-- output port

MOV     CRC1,#00001000B         ; enable clear TM1 by CR11
MOV     PRM1,#00000011B         ; set prescaler (TM1:fclk/32)

CALL    !KD_INI                 ; initialize key & display
                                ;           (INTC10/C11/P0)
MOV     TMC1,#00001000B         ; start timer (TM1)
EI
        :
        :
```



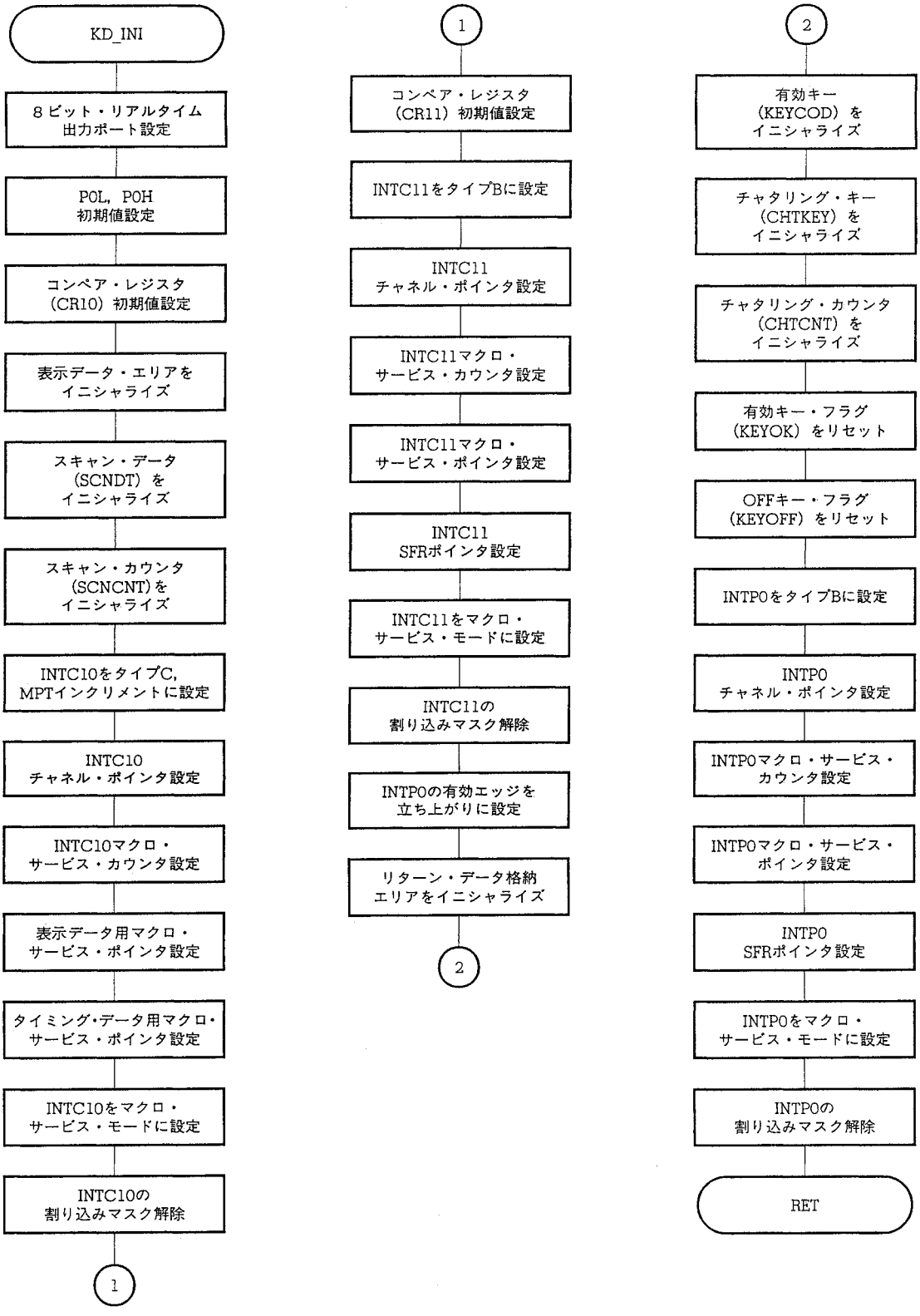

```
;*************************************
;*      KEY_DATA CHECK & STORE       *
;*************************************
;
        BF      KEYOK,$KD_END           ; check valid key
        CLR1    KEYOK                   ; clear valid key
        MOV     A,KEYCOD                ; read keycode
;
;*************************************
;*      LED_DATA CHANGE & STORE      *
;*************************************
;
        CALL    !SEGCNV                 ; conversion segment data
        MOVW    HL,#LED0                ; store LED display new data
        XCH     A,[HL]
                :
                :
KD_END:
                :
                :
```

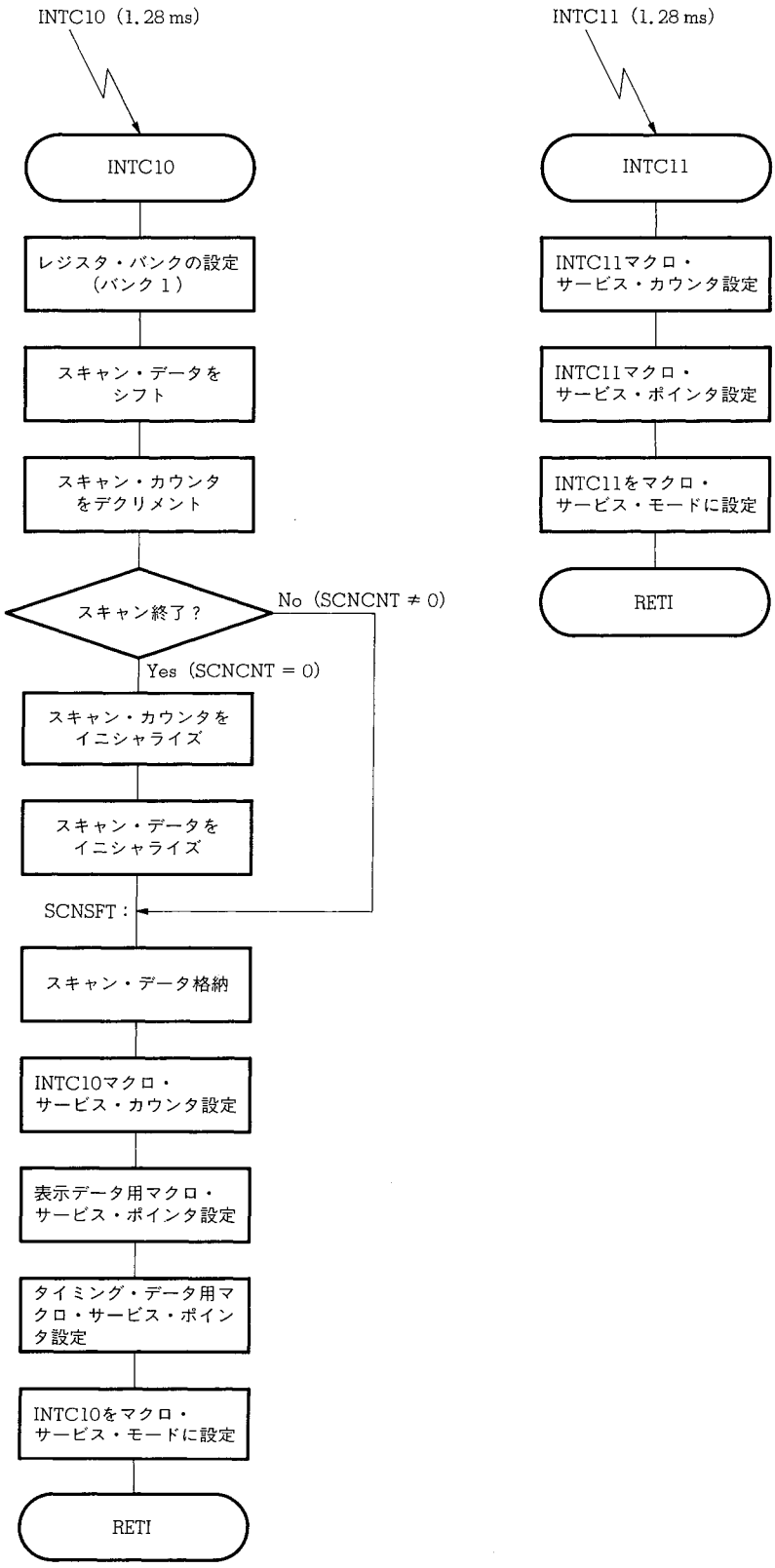
(8) フロー・チャート

(a) セグメント・データ変換処理モジュール：SG_CONV

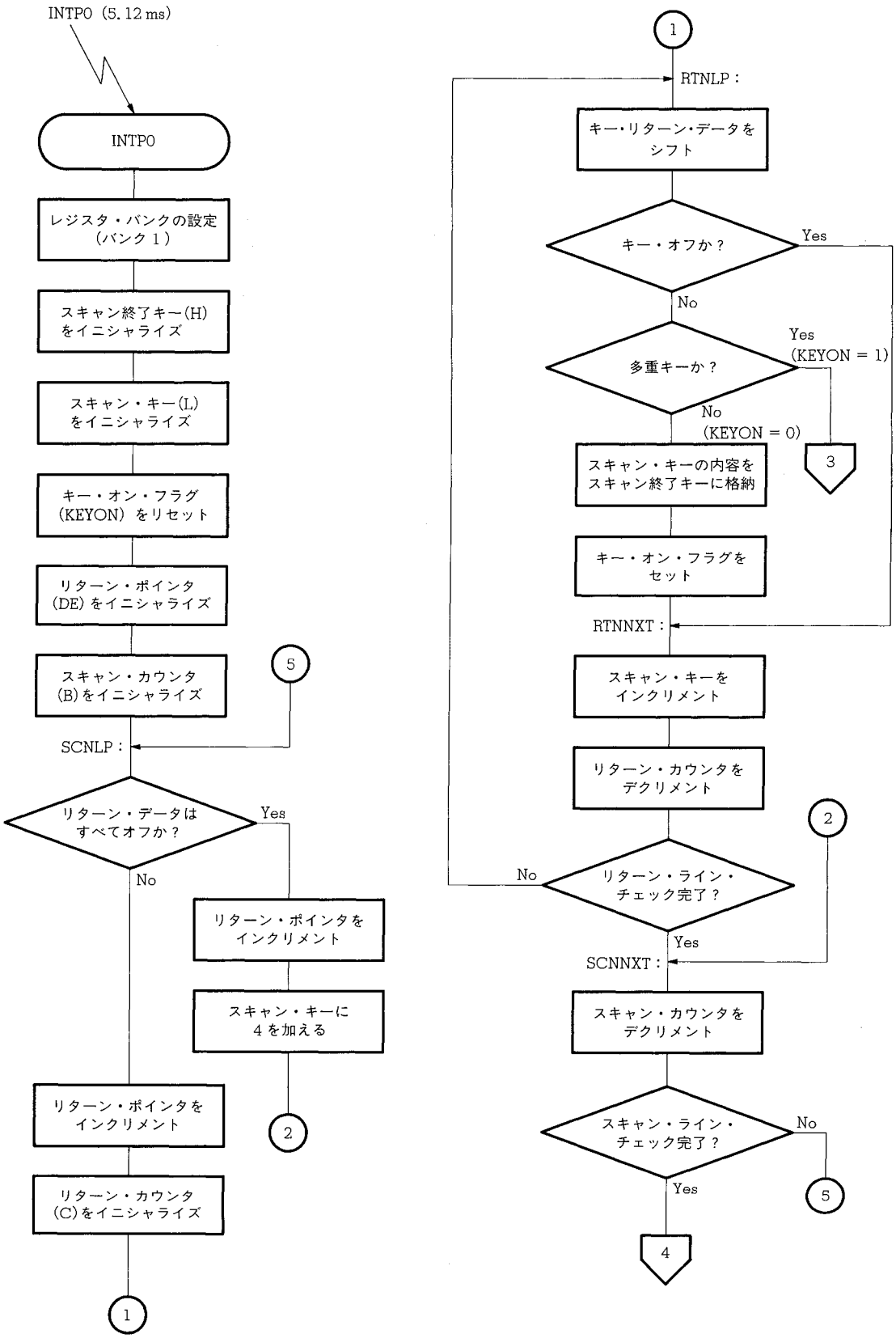


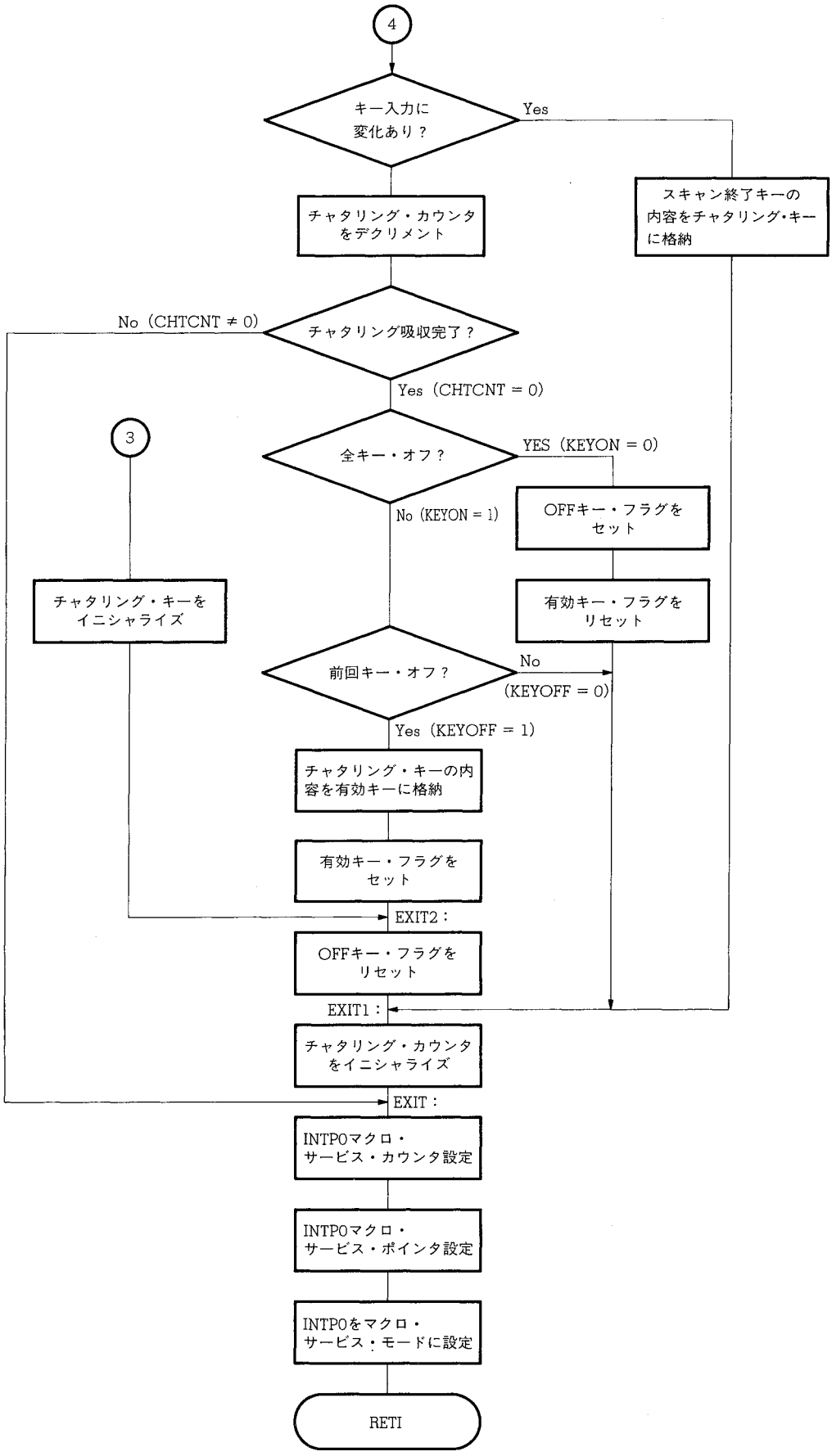
(b) 表示／キー入力共通モジュール：KD_MCR




INTC10（1.28 ms）
INTC10
レジスタ・バンクの設定
（バンク 1）
スキャン・データを
シフト
スキャン・カウンタ
をデクリメント
スキャン終了？
No（SCNCNT ≠ 0）
Yes（SCNCNT = 0）
スキャン・カウンタを
イニシャライズ
スキャン・データを
イニシャライズ
SCNSFT：
スキャン・データ格納
INTC10マクロ・
サービス・カウンタ設定
表示データ用マクロ・
サービス・ポインタ設定
タイミング・データ用マ
クロ・サービス・ポイン
タ設定
INTC10をマクロ・
サービス・モードに設定
RETI
INTC11（1.28 ms）
INTC11
INTC11マクロ・
サービス・カウンタ設定
INTC11マクロ・
サービス・ポインタ設定
INTC11をマクロ・
サービス・モードに設定
RETI

〔メ　モ〕


INTP0（5.12 ms）
INTP0
レジスタ・バンクの設定
（バンク1）
スキャン終了キー(H)
をイニシャライズ
スキャン・キー(L)
をイニシャライズ
キー・オン・フラグ
（KEYON）をリセット
リターン・ポインタ
(DE)をイニシャライズ
スキャン・カウンタ
(B)をイニシャライズ
5
SCNLP：
リターン・データは
すべてオフか？
Yes
No
リターン・ポインタを
インクリメント
スキャン・キーに
4を加える
2
リターン・ポインタを
インクリメント
リターン・カウンタ
(C)をイニシャライズ
1
1
RTNLP：
キー・リターン・データを
シフト
キー・オフか？
Yes
No
多重キーか？
Yes
(KEYON = 1)
No
(KEYON = 0)
3
スキャン・キーの内容を
スキャン終了キーに格納
キー・オン・フラグを
セット
RTNNXT：
スキャン・キーを
インクリメント
リターン・カウンタを
デクリメント
2
No
リターン・ライン・
チェック完了？
Yes
SCNNXT：
スキャン・カウンタを
デクリメント
スキャン・ライン・
チェック完了？
No
5
Yes
4



7
214
218A
224
234
244

(9) プログラム・リスト

今回のプログラム例では，モジュールの先頭でベクタ割り込みおよびマクロ・サービス定義用のファイル(INTMS.DEF)をインクルードしています((b) 表示／キー入力処理モジュール：KD_MCR)。

(a) セグメント・データ変換処理モジュール：SG_CONV

```
        NAME    SG_CONV
;
;****************************************
;*      LED SEGMENT DATA CONVERSION     *
;****************************************
;
        PUBLIC  SEGCNV                  ; package

SGCONV  CSEG
SEGCNV:                                 ; conversion segment data
        MOV     A,SEGTBL[A]
        RET

SEGTBL:                                 ; input code profile
        DB      01110111B               ; (00) 0
        DB      00010100B               ; (01) 1
        DB      10110011B               ; (02) 2
        DB      10110110B               ; (03) 3
        DB      11010100B               ; (04) 4
        DB      11100110B               ; (05) 5
        DB      11100111B               ; (06) 6
        DB      01110100B               ; (07) 7
        DB      11110111B               ; (08) 8
        DB      11110110B               ; (09) 9
        DB      11110101B               ; (0A) A
        DB      11000111B               ; (0B) b
        DB      01100011B               ; (0C) C
        DB      10010111B               ; (0D) d
        DB      11100011B               ; (0E) E
        DB      11100001B               ; (0F) F
;
        END
```

(b) 表示／キー入力処理モジュール：KD_MCR

```
$       IC(INTMS.DEF)                   ; interrupt & macro service define macro
        NAME    KD_MCR
;
;*****************************************
;*      INTC10/C11/P0 MACRO SERVISE      *
;*          INITIALIZE & CONVERSION      *
;*****************************************
;
        PUBLIC  KD_INI                  ; package
        PUBLIC  KEYOK,KEYCOD            ; parameter
        PUBLIC  LED0,LED1,LED2,LED3     ; parameter
;
;       --- symbol define 1 ---
;
SCNNO   EQU     2                       ; scan timing number
SCNLN   EQU     4                       ; scan line number
SCNINI  EQU     11111110B               ; scan initial data
RETLN   EQU     4                       ; return line number
RLMSK   EQU     00001111B               ; return data masking
CHTINI  EQU     4                       ; chattering counter initial data
KEYON   EQU     X.7                     ; key on check flag
ALLOFF  EQU     0FFH                    ; keycode off data
NGKEY   EQU     0EEH                    ; multi-key data

INTIMER EQU     40-1                    ; interval time data (213.3μs)
SEGNO   EQU     4                       ; segment display number
TIMENO  EQU     SEGNO*2+SCNNO*2         ; INTC10 counter data (timing data NO.)
;
;       --- for work ---
;
        BSEG
KEYOK   DBIT                            ; valid key flag
KEYOFF  DBIT                            ; all off key flag

MPDS    DSEG    PAGE
DSPDT:  DS      SEGNO*2                 ; INTC10 segment data (4 line)
SCNDT:  DS      SCNNO*2                 ;        key scan data (1 line only)

KRETS   DSEG
RETDT:  DS      SCNLN                   ; INTP0  return data top address

KEYS    DSEG    SADDR
SCNCNT: DS      1                       ; scan line shift counter
KEYCOD: DS      1                       ; last keycode
CHTKEY: DS      1                       ; chattering key
CHTCNT: DS      1                       ; chattering counter
```

7
214
218A
224
234
244

```
        DSEG_C  C10,'SADDR'             ; INTC10 macro service channel(type C)
        CMCW10                          ;        macro service control word

        DSEG_B  C11,'SADDR'             ; INTC11 macro service channel(type B)
        CMCW11                          ;        macro service control word

        DSEG_B  P0,'SADDR'              ; INTP0  macro service channel(type B)
        PMCW0                           ;        macro service control word
;
;       --- symbol define 2 ---
;
LED0    EQU     DSPDT                   ; No.0 LED display data address
LED1    EQU     DSPDT+2                 ; No.1
LED2    EQU     DSPDT+4                 ; No.2
LED3    EQU     DSPDT+6                 ; No.3
;
;       --- vector macro define ---
;
        CVENT10                         ; INTC10 vector macro
        CVENT11                         ; INTC11 vector macro
        PVENT0                          ; INTP0  vector macro
;
KDMCR1  CSEG
KD_INI:
;
;*****************************************
;*     INTC10 MACRO SERVICE INITIALIZE    *
;*****************************************
;
        MOV     RTPC,#10010001B         ; port0 -> 8bit realtime output port
        MOV     P0L,#00000000B          ; realtime output port first data
        MOV     P0H,#00000000B
        MOV     CR10,#INTIMER           ; set initial interval timer (213.3μs)

        MOVW    HL,#DSPDT               ; initialize LED display area
        MOV     B,#SEGNO
        MOVW    AX,#0000H
DSPINI:
        MOVW    [HL],AX
        INCW    HL
        INCW    HL
        DBNZ    B,$DSPINI

        MOV     A,#SCNINI               ; initialize scan data
        MOVW    HL,#SCNDT
        MOV     [HL+],A
        MOV     [HL+],A
        MOV     [HL+],A
        MOV     A,#00H                  ; initialize off end data
        MOV     [HL],A
        MOV     SCNCNT,#SCNLN           ; initialize scan counter
```

```
        MOV     MSMC10,#11101000B       ; set mode register (TYPE C,increment MPT)
        MOV     CHPC10,#LOW CHAC10      ; set channel pointer
        MOV     MSCC10,#TIMENO          ; set macro service counter
        MOVW    MPDC10P,#DSPDT          ; set pointer for data
        MOVW    MPTC10P,#TIMING         ; set pointer for timing
        SET1    CISM10                  ; set INTC10 macro service mode
        CLR1    CMK10                   ; open INTC10 mask flag
;
;*****************************************
;*     INTC11 MACRO SERVICE INITIALIZE     *
;*****************************************
;
        MOV     CR11,#INTIMER           ; set interval timer (213.3μs)

        MOV     MSMC11,#10100000B       ; set mode register (TYPE B,memory -> SFR)
        MOV     CHPC11,#LOW CHAC11      ; set channel pointer
        MOV     MSCC11,#(SEGNO+SCNNO)   ; set macro service counter
        MOVW    MPC11P,#DGTADR          ; set macro service pointer
        MOV     SFRC11,#LOW P3          ; set SFR pointer (PORT3)
        SET1    CISM11                  ; set INTC11 macro service mode
        CLR1    CMK11                   ; open INTC11 mask flag
;
;*****************************************
;*     INTP0 MACRO SERVICE INITIALIZE     *
;*****************************************
;
        MOV     INTM0,#00000100B        ; INTP0 rize edge mode

        MOVW    HL,#RETDT               ; initialize return data store area
        MOV     B,#SCNLN
        MOV     A,#ALLOFF
RETINI:
        MOV     [HL+],A
        DBNZ    B,$RETINI

        MOV     KEYCOD,A                ; initialize last keycode
        MOV     CHTKEY,#NGKEY           ; initialize chattering key
        MOV     CHTCNT,#CHTINI          ; initialize chattering counter
        CLR1    KEYOK                   ; initialize valid key flag
        CLR1    KEYOFF                  ; initialize all off key flag

        MOV     MSMP0,#10100001B        ; set mode register (TYPE B,SFR -> memory)
        MOV     CHPP0,#LOW CHAP0        ; set channel pointer
        MOV     MSCP0,#SCNLN            ; set macro service counter
        MOVW    MPP0P,#RETDT            ; set macro service pointer
        MOV     SFRP0,#LOW P2           ; set SFR pointer (PORT2)
        SET1    PISM0                   ; set INTP0 macro service mode
        CLR1    PMK0                    ; open INTP0 mask flag
        RET
```

7
214
218A
224
234
244

```
;
;************************************
;*      INTC10 MACRO SERVICE        *
;*      FINISHED INTERRUPT          *
;************************************
;
INTC10:                                 ; INTC10 (1.28 ms)
        SEL     RB1                     ; change register bank (1)
        MOV     A,!SCNDT                ; shift scanning data
        ROL     A,1
        DBNZ    SCNCNT,$SCNSFT
        MOV     SCNCNT,#SCNLN           ; initialize scan counter
        MOV     A,#SCNINI               ; initialize scan data
SCNSFT:
        MOVW    HL,#SCNDT
        MOV     [HL+],A
        MOV     [HL+],A
        MOV     [HL],A

        MOV     MSCC10,#TIMENO          ; restore macro service counter
        MOVW    MPDC10P,#DSPDT          ; restore pointer for data
        MOVW    MPTC10P,#TIMING         ; restore pointer for timing
        SET1    CISM10                  ; set INTC10 macro service mode
        RETI
;
;*********************************
;*      INTC10 TIMING DATA       *
;*********************************
;
MPTS    CSEG    PAGE

TIMING: REPT    SEGNO+SCNNO
        DB      05                      ; interval time  26.7μs
        DB      35                      ;               186.7μs
        ENDM
;
KDMCR2  CSEG
;
;***********************************
;*      INTC11 MACRO SERVICE       *
;*      FINISHED INTERRUPT         *
;***********************************
;
INTC11:                                 ; INTC11 (1.28 ms)
        MOV     MSCC11,#(SEGNO+SCNNO)   ; restore macro service counter
        MOVW    MPC11P,#DGTADR          ; restore macro service pointer
        SET1    CISM11                  ; set INTC11 macro service mode
        RETI
```

```
;
;**************************************
;*      INTC11 DIGIT SIGNAL DATA      *
;**************************************
;
DGTADR:
        DB      11101000B               ; LED No.0
        DB      11011000B               ; LED No.1
        DB      10111000B               ; LED No.2
        DB      01111000B               ; LED No.3
        DB      11110000B               ; INTP0 making
        DB      11111000B
;
;**********************************
;*      INTP0 MACRO SERVICE       *
;*      FINISHED INTERRUPT        *
;**********************************
;
INTP0:                                  ; INTP0 (5.12 ms)
        SEL     RB1                     ; change register bank (1)
        MOV     H,#ALLOFF               ; initialize Hreg last scan-key
        MOV     L,#00H                  ;            Lreg scan-key
        CLR1    KEYON                   ; clear key on check flag
        MOVW    DE,#RETDT               ; check return keydata
        MOV     B,#SCNLN
SCNLP:                                  ; start scan loop
        MOV     A,[DE+]
        OR      A,#RLMSK                ; mask return line
        CMP     A,#ALLOFF
        BNE     $RLCHK                  ; all key off ?
        MOV     A,#RETLN                ; then make next scan line's key code
        ADD     L,A
        BR      SCNNXT
RLCHK:
        MOV     C,#RETLN                ; check return line
RTNLP:
        SHL     A,1                     ; key off ?
        BC      $RTNNXT
        BF      KEYON,$SETKEY           ; multi-key ?
        MOV     CHTKEY,#NGKEY           ; initialize chattering key
        BR      EXIT2
SETKEY:
        MOV     H,L                     ; set last scan-key
        SET1    KEYON                   ; set key on check flag
RTNNXT:
        INC     L                       ; make next return line's key code
        DBNZ    C,$RTNLP                ; check all return line
```

7
214
218A
224
234
244

```
SCNNXT:
        DBNZ    B,$SCNLP                ; check all scan line

        MOV     A,H                     ; change key ?
        CMP     A,CHTKEY
        BE      $CHTRNG
        MOV     CHTKEY,A                ; set chattering key
        BR      EXIT1
CHTRNG:
        DBNZ    CHTCNT,$EXIT            ; absorb chattering ?

        BT      KEYON,$CHKKEY           ; all key off ?
        SET1    KEYOFF                  ; off key in ! set KEYOFF
        CLR1    KEYOK                   ;              clear KEYOK
        BR      EXIT1
CHKKEY:
        BF      KEYOFF,$EXIT1           ; if invalid key , branch 'EXIT1'
        MOV     KEYCOD,CHTKEY           ; valid key in ! set  last keycode
        SET1    KEYOK                   ;                set  KEYOK
EXIT2:
        CLR1    KEYOFF                  ; clear KEYOFF
EXIT1:
        MOV     CHTCNT,#CHTINI          ; initialize chattering counter
EXIT:
        MOV     MSCP0,#SCNLN            ; restore macro service counter
        MOVW    MPP0P,#RETDT            ; restore macro service pointer
        SET1    PISM0                   ; set INTP0 macro service mode
        RETI
;
        END
```

第8章　アナログ・キー入力

アナログ・キー入力は，アナログ入力端子にかかる電圧によって，押されたキーを判断するキー入力方法です。

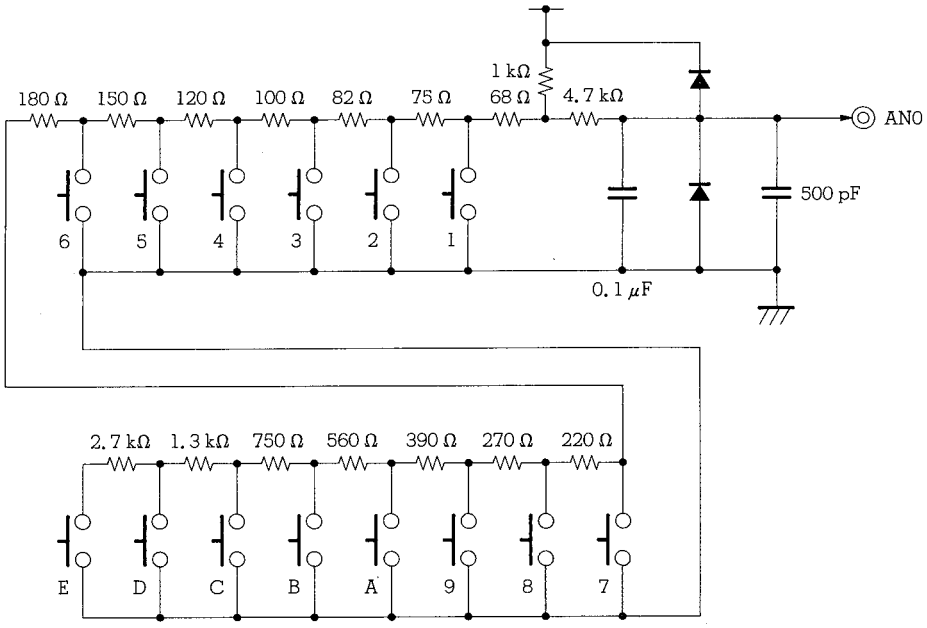
次の二通りの方法でアナログ・キー入力を行います。

- A/Dコンバータによるアナログ・キー入力（μPD78214，78218A，78234，78244シリーズで可）
- ポートTによるアナログ・キー入力（μPD78224シリーズで可）

8.1　A/Dコンバータによるアナログ・キー入力

AN0（μPD78234シリーズではANI0）端子に，14個のキーを接続した場合のプログラム例を示します。図8-1に回路例を示します。

図8-1　アナログ・キーの回路例

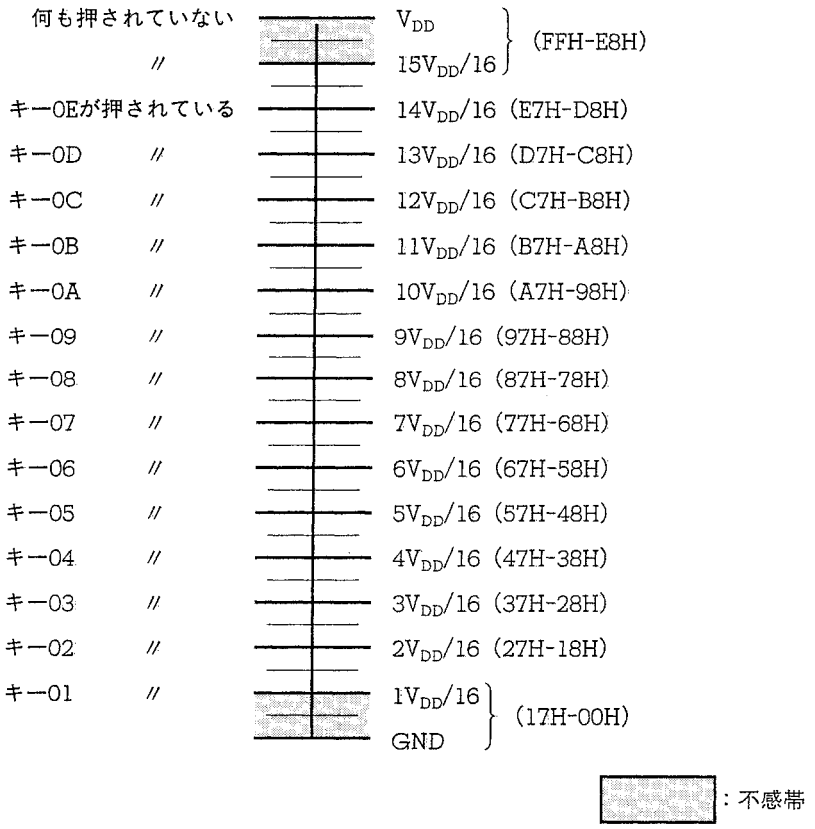


この回路例では，GND，V_{DD}付近に$1/16V_{DD}$の不感帯を設けています。

(1) 動作概要

AN0端子への入力電圧によるキー入力を行います。キーは14個ですが，不感帯の部分も含めると，入力電圧は16分割されることになります。図8-2にキー（キー・コード）と入力電圧（A/D変換値）の関係を示します。

図8-2　キーと入力電圧の関係



キーに変化があった場合のみ，キー・コード（01H-0EH，10H）を取り込みます。

また，キー入力のチャタリング吸収は，キー・コードの4回一致を見ることによって行います。

(2) プログラム説明（「(8) SPDチャート」参照）

(a) 処理概要

(i) イニシャライズ処理［レーベル名称：ANKYINI］

A/Dコンバータとワーク・エリアのイニシャライズを行います。キー・コード格納エリアには，無効キー・コード（EEH）を設定しておきます。

(ii) キー・コード変換処理［レーベル名称：KEYCNV］

A/D変換値を読み出し，キー・コードに変換します。キー・コードに変換の際は，入力電圧の誤差によってキー・コードが変動しないようA/D変換値を補正してから行います。得られたキー・コードによってチャタリング吸収します。

(b) 使用するRAM

このプログラム例では，キー入力用に3バイトと2ビットのRAMを使用します。表8-1に使用RAM一覧を示します。

表8-1　A/Dコンバータによるアナログ・キー入力で使用するRAM一覧

RAM名称	配置領域	用　　途	バイト数	初期値
KEYCODE	SADDR[注]	有効キー・コードを格納	1	不定
PASTDAT		前回のキー・コードを格納	1	不定
CHATCNT		チャタリングのカウント	1	不定
KEYCHGF		キー変化の有無の判断	1ビット	0
CHTCNTF		チャタリング・カウント中であるかの判断	1ビット	1

注　SADDR：ショート・ダイレクト・アドレシングの適用範囲（0FE20H-0FEFFH）

(3) 入出力パラメータ

(a) 入力パラメータ

なし

(b) 出力パラメータ

表8-2　A/Dコンバータによるアナログ・キー入力の出力パラメータ

パラメータ	固定／可変	内　　容
KEYCODE	可変値	有効キー・コード
KEYCHGF	可変値	キー変化フラグ
KEYOFF	固定値	OFFキー・コード

(4) 使用レジスタ

表8-3　A/Dコンバータによるアナログ・キー入力の使用レジスタ

ルーチン	レーベル	レジスタ	バンク
イニシャライズ処理	ANKYINI	なし	—
キー・コード変換処理	KEYCNV	A	0

(5) 使用スタック

このプログラム例のサブルーチン，および割り込み処理で使用するスタック・サイズは2バイトです（ネスティング：1レベル）。

(6) モード・レジスタ設定例

A/Dコンバータ・モード・レジスタ

	7	6	5	4	3	2	1	0	
ADM	CS	TRG	0	FR	ANI2	ANI1	ANI0	MS	(リセットで00H)
設定例	1	0	0	0	0	0	0	1	

ANI2	ANI1	ANI0	MS	A/D変換動作モードの指定	
0	0	0	0	スキャン・モード	ANI0入力をスキャン
0	0	1	0		ANI0, ANI1入力をスキャン
0	1	0	0		ANI0-ANI2入力をスキャン
0	1	1	0		ANI0-ANI3入力をスキャン
1	0	0	0		ANI0-ANI4入力をスキャン
1	0	1	0		ANI0-ANI5入力をスキャン
1	1	0	0		ANI0-ANI6入力をスキャン
1	1	1	0		ANI0-ANI7入力をスキャン
0	0	0	1	セレクト・モード	ANI0入力をセレクト
0	0	1	1		ANI1入力をセレクト
0	1	0	1		ANI2入力をセレクト
0	1	1	1		ANI3入力をセレクト
1	0	0	1		ANI4入力をセレクト
1	0	1	1		ANI5入力をセレクト
1	1	0	1		ANI6入力をセレクト
1	1	1	1		ANI7入力をセレクト

FR	変換速度制御	
0	180クロック	低速変換
1	120クロック	高速変換

TRG	外部端子トリガの制御
0	外部トリガ禁止
1	外部トリガ許可

CS	A/D変換動作制御
0	A/D変換動作停止
1	A/D変換動作開始

8
214
218A
234
244

(7) プログラム使用例

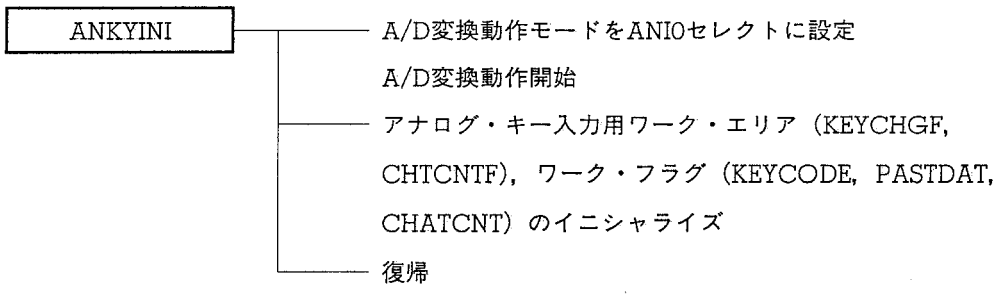
このプログラム例の使用例を示します。

キーに変化があったかを確認して（KEYCHGFのチェック），キー・コードを読み出しています。また，キー入力なしの確認もしています。

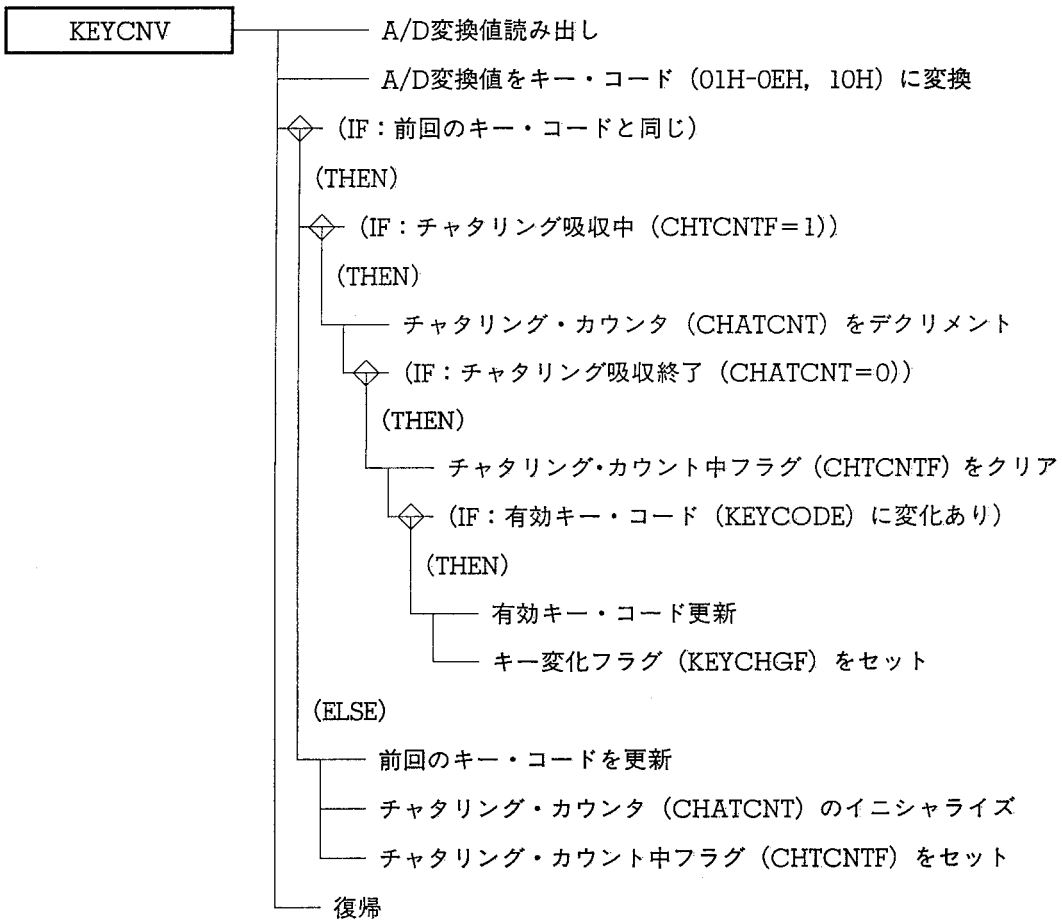
```
EXTRN   ANKYINI,KEYCNV          ; package
EXTRN   KEYOFF                  ; data
EXTRN   KEYCODE                 ; work area
EXTBIT  KEYCHGF                 ; work flag

CSEG
        :
        :
CALL    !ANKYINI                ; initialize of A/D conversion
EI                              ; enable interrupt
        :
        :
    if_bit(KEYCHGF)             ; IF key changed ?
        CLR1    KEYCHGF         ; THEN clear key change flag
        A=KEYCODE               ;    read key code

        if(A==#KEYOFF)          ;    key off ?
        :
        :
        else                    ;    new key
        :
        :
        endif
        :
        :
    endif
        :
        :
```

(8) SPDチャート

●イニシャライズ処理



●キー・コード変換処理



8
214
218A
234
244

(9) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードしています。

```
$       IC(INTMS.DEF)
        NAME    ANKEY
;*************************************************************
;*      ANALOG KEY - INPUT                                   *
;*************************************************************

        PUBLIC  ANKYINI,KEYCNV          ; package
        PUBLIC  KEYCODE                 ; work area
        PUBLIC  KEYCHGF                 ; work flag
        PUBLIC  KEYOFF                  ; data

;       *** define data ***

CNTDATA EQU     4                       ; number of chattering count
NGDATA  EQU     0EEH                    ; key work initial data
KEYOFF  EQU     10H                     ; key off data

;       *** define work area ***

        BSEG
KEYCHGF DBIT                            ; key change flag
CHTCNTF DBIT                            ; chattering counting flag

KYWORKS DSEG    SADDR
KEYCODE:DS      1                       ; key code store area
PASTDAT:DS      1                       ; past key code store area
CHATCNT:DS      1                       ; chattering count area

;*****************************************
;*      initialize routine               *
;*****************************************

ANKYCS  CSEG
ANKYINI:
        ADM=#10000001B                  ; AN0 select mode, start A/D converter
        KEYCODE=#NGDATA                 ; initialize key code
        PASTDAT=#NGDATA                 ; initialize past key code
        CHATCNT=#CNTDATA                ; initialize chattering counter
        CLR1    KEYCHGF
        CLR1    CHTCNTF
        RET
```

```
;*****************************************
;*      conversion key data              *
;*****************************************

KEYCNV:
        A=ADCR                          ; read A/D conversion result
        A+=#08H                         ; revise A/D conversion result
        RORC    A,1
        A>>=3
        A=KEYTBL[A]                     ; refer key code table

        if(A==PASTDAT)                  ; key code does not change ?
            if_bit(CHTCNTF)             ; THEN chattering counting ?
                CHATCNT--
                if(CHATCNT==#0)         ;       chattering count end ?
                    CLR1    CHTCNTF     ;       clear chattering counting flag
                    if(PASTDAT!=KEYCODE);       last key change ?
                        KEYCODE=PASTDAT ;       store last key code
                        SET1    KEYCHGF ;       set key change flag
                    endif
                endif
            endif

        else                            ; ELSE (case of key code change)
            PASTDAT=A                   ;       update past key code
            CHATCNT=#CNTDATA            ;       initialize chattering counter
            SET1    CHTCNTF             ;       set chattering counting flag
        endif

        RET

;       *** KEY DATA TABLE ***

KEYTBL:
        DB      01H,01H,02H,03H,04H,05H,06H,07H
        DB      08H,09H,0AH,0BH,0CH,0DH,0EH,KEYOFF
        DB      KEYOFF

        END
```

8
214
218A
234
244

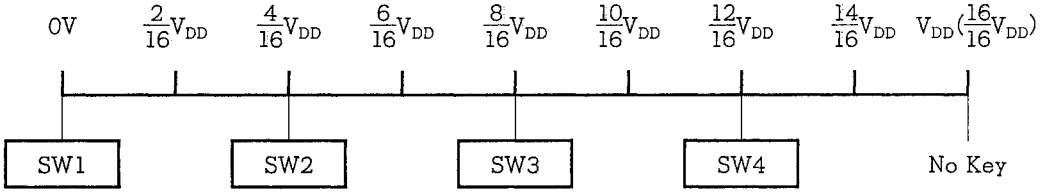
8.2　アナログ・キー入力

ポートTを用いて，1ライン上に配置された4つのキーの入力を行います。

キー入力を行うためには，キーが押されたときに，その固有の電圧がポートTに対して与えられるように回路を構成します。このときの電圧を調べることにより，どのキーが押されているかを判断します。

4つのキーに対する電圧を図8-3に示します。

図8-3　4つのキーに対する電圧



(1) 動作概要

PT0に接続されたキーのキー入力を行います。キー入力は，ロック・アウト方式，チャタリング吸収時間を25 ms程度とし，有効キーがあった場合には，CHNGFLフラグをセット(1)し，レーベル(OLDKEY) に0-4の値でキー・コードを生成します。

図8-4　キー入力で用いるワーク・エリア

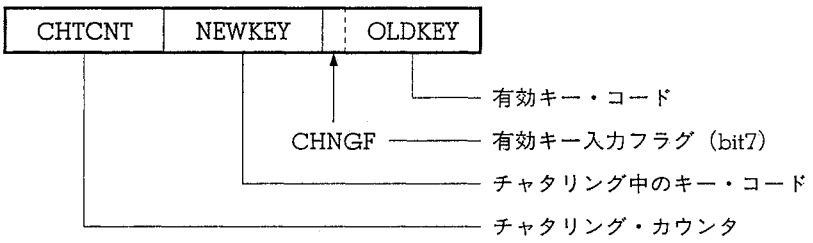


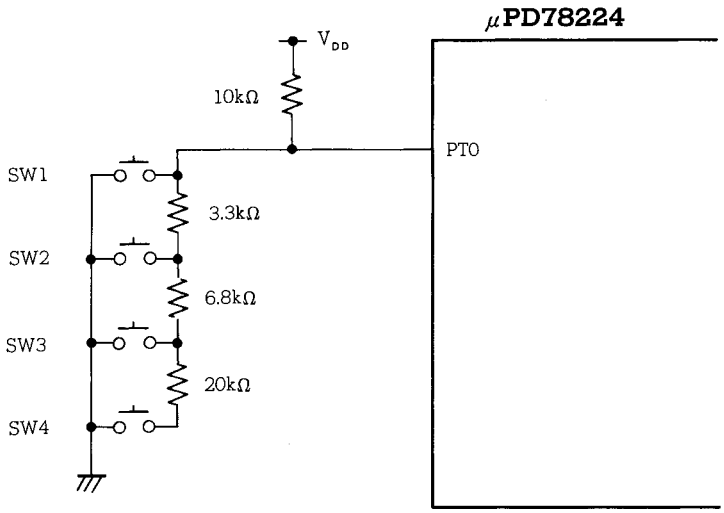
表8-4に各キーに対するキー・コードを示します。

表8-4　各キーに対するキー・コード

SW1	SW2	SW3	SW4	キー入力なし
4	3	2	1	0

図8-3の電圧を各キーに応じて発生させるための回路を図8-5に示します。

図8-5　アナログ・キーの回路例



注意　抵抗は許容差1％の金属皮膜抵抗などを用いてください。

(2) プログラム説明 … (6) プログラム・リスト参照

イニシャライズ処理［レーベル名称：KEY_II］

(a) NEWKEY，OLDKEYにそれぞれ無効キー・コードとして0FHを設定します。

(b) 5.12 msごとにINTC21割り込み要求が発生するようにタイマを設定します。

(c) INTC21のマスクを解除します。

(d) 8ビット・タイマ／カウンタ1のカウントをスタートします。

(e) 割り込みを許可します。

INTC21割り込み処理［レーベル名称：INTC21］

(a) PT0入力電圧を参照するテーブルの先頭アドレスをHLレジスタに設定します。

(b) キー・コードを生成するXレジスタをクリアします。

(c) ポートTの入力電圧とテーブル・データを順検索し，見つかるまでの回数をXレジスタにセットします。

(d) チャタリングの吸収処理を行います。5.12 ms×5回の一致（＝25.6 ms）で吸収を行います。チャタリングを吸収した場合はチャタリング・カウンタをクリアし，キー・コード格納の処理へ進みます。

(e) 前回の有効キーのデータが今回のキー入力と異なったキーが押されていることを示している場合は，今回のキー入力を有効キーとしてメモリに格納しません（ロック・アウトの処理）。ただし，キー・データが何も押されていない状態のキー・コードであれば前回のキー・コードに対して無条件に有効キーとして格納します。有効キーの場合にはキー・コード変更フラグCHNGFLフラグをセット(1)し，NEWKEYに0FHを設定します。

(3) 入出力パラメータ

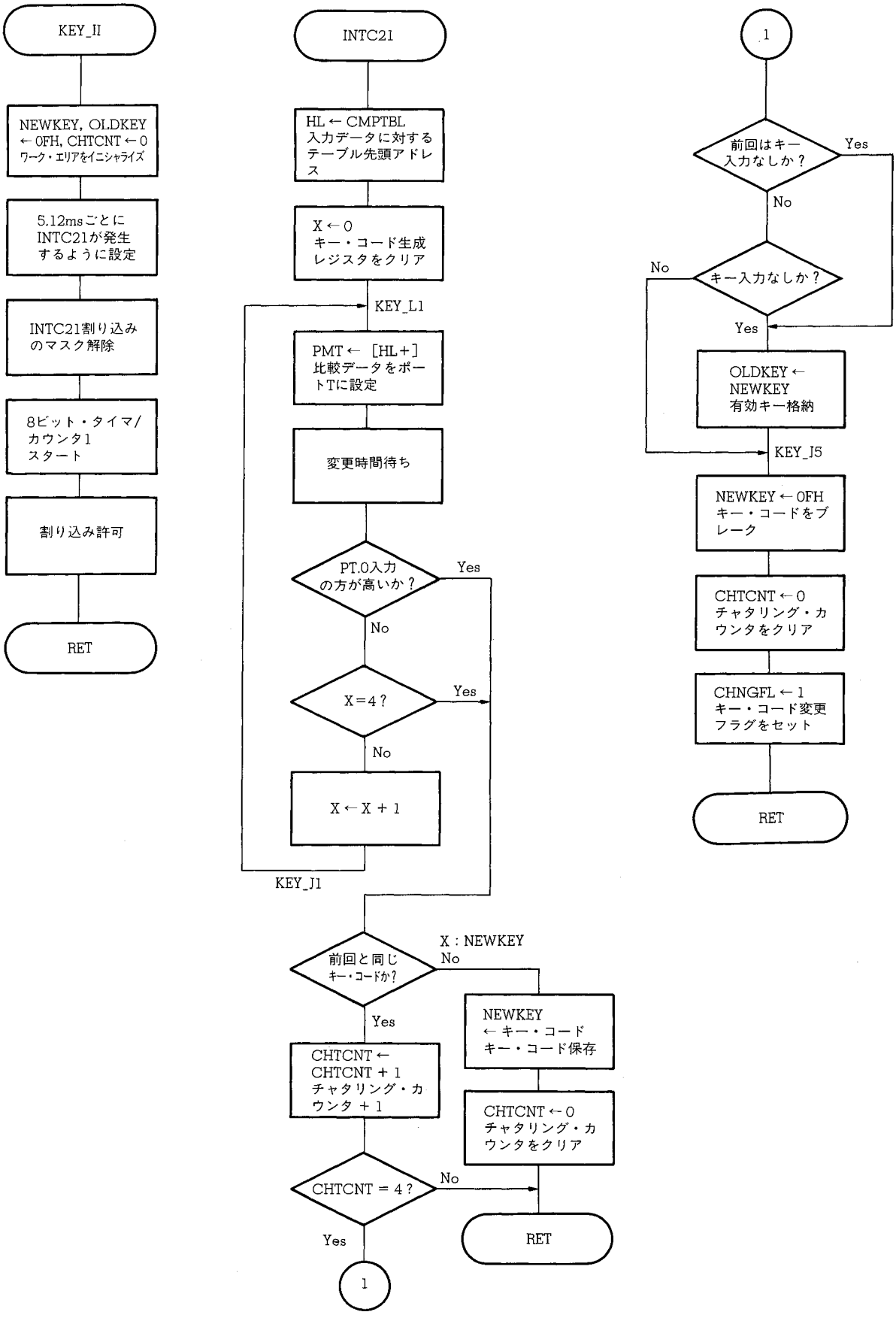
(a) OLDKEY ：有効キーが格納されます。

(b) CHNGFL：有効キーのキー・コードが変化するとセット(1)されるフラグです。

(4) 使用レジスタ

A，X，C，HL

(5) フロー・チャート


KEY_II
NEWKEY, OLDKEY ← 0FH, CHTCNT ← 0
ワーク・エリアをイニシャライズ
5.12msごとに INTC21が発生するように設定
INTC21割り込みのマスク解除
8ビット・タイマ/カウンタ1 スタート
割り込み許可
RET
INTC21
HL ← CMPTBL
入力データに対するテーブル先頭アドレス
X ← 0
キー・コード生成レジスタをクリア
KEY_L1
PMT ← [HL+]
比較データをポートTに設定
変更時間待ち
PT.0入力の方が高いか？
Yes
No
X=4？
Yes
No
X ← X + 1
KEY_J1
前回と同じキー・コードか?
X : NEWKEY
No
Yes
CHTCNT ← CHTCNT + 1
チャタリング・カウンタ + 1
NEWKEY ← キー・コード
キー・コード保存
CHTCNT ← 0
チャタリング・カウンタをクリア
CHTCNT = 4？
No
Yes
RET
1
1
前回はキー入力なしか？
Yes
No
キー入力なしか？
No
Yes
OLDKEY ← NEWKEY
有効キー格納
KEY_J5
NEWKEY ← 0FH
キー・コードをブレーク
CHTCNT ← 0
チャタリング・カウンタをクリア
CHNGFL ← 1
キー・コード変更フラグをセット
RET

(6) プログラム・リスト

このプログラム例では，ソース・プログラムの先頭で，ベクタ・テーブル・アドレスとマクロ・サービスの領域の定義用ファイル（INTMS.DEF）をインクルードします。

```
        NAME    KEYINR
;
;*************************************************************
;* analogue-digital convert on                               *
;*              programable threshold port                   *
;*                      for only uPD78224,78220              *
;*      sampling port                                        *
;*              PT0                                          *
;*      result                                               *
;*              NEWKEY at chattering key code                *
;*              OLDKEY at after chattering key code          *
;*              CHTCNT chattering counter                    *
;*              CHNGFG key code change flag at OLDKEY MSB    *
;*************************************************************
;
        PUBLIC  KEY_II

        EXTRN   INTERV,NEWKEY,OLDKEY,CHTCNT
        EXTBIT  CHNGFL

KYWAIT  EQU     26              ; read result wait time

INTC21VT CSEG   AT 0001CH
        DW      INTC21          ; INTC21
;
;*************************************************************
;       Initialize Sampling late        5.12ms
;               by INTC21
;*************************************************************
;
KEYIN_C CSEG
KEY_II:
        MOV     NEWKEY,#0FH     ; code area clear
        MOV     OLDKEY,#0FH
        MOV     CHTCNT,#0

        MOV     TMC1,#00000000B ; initialize Timer/Counter-2
        MOV     CRC2,#00011000B ; enable clear TM2 by CR21
        MOV     PRM1,#01110000B ; select fclk/512 (TM2)
        MOV     CR21,#LOW(INTERV-1)
                                ; set interval time (5.12ms)
        MOV     MK0H,#11111110B ; open CR21 mask
        OR      TMC1,#10000000B ; timer start
        EI                      ; interrupt enable
        RET
```

```
;
;************************************************************
;       Key Input MAIN
;               at INTC21 interrupt process
;************************************************************
;
INTC21:
        MOVW    HL,#CMPTBL      ; compare date area top address
        MOV     X,#0            ; clear result
KEY_L1:
        MOV     A,[HL+]         ; read compare data
        MOV     PMT,A
        MOV     C,#KYWAIT       ; time delay
KEY_L2:
        DBNZ    C,$KEY_L2
        BT      PT.0,$KEY_J1    ; result check
        MOV     A,X             ; loop end check
        CMP     A,#4
        BZ      $KEY_J1
        INC     X
        BR      KEY_L1
KEY_J1:
        MOV     A,X
        CMP     A,NEWKEY        ; check chattering
        BZ      $KEY_J2
        MOV     CHTCNT,#0
        MOV     NEWKEY,A
        RETI
KEY_J2:
        INC     CHTCNT          ; chattering counter increment
        CMP     CHTCNT,#4
        BZ      $KEY_J3
        RETI
KEY_J3:
        CMP     OLDKEY,#0       ; lock out check
        BZ      $KEY_J4
        CMP     NEWKEY,#0
        BNZ     $KEY_J5
KEY_J4:
        MOV     OLDKEY,NEWKEY
KEY_J5:
        MOV     NEWKEY,#0FH     ; break key code
        MOV     CHTCNT,#0
        SET1    CHNGFL
        RETI
CMPTBL:
        DB      0FH,0CH,08H,04H,01H     ; compare data

        END
```

第９章　内蔵ROMチェック

78K/IIシリーズのうち，μPD78212，78214，78218A，78224，78234，78238，78244はROMを内蔵しています。この内蔵ROMは，プログラム・メモリの0000H番地からマッピングされています。表９－１に内蔵ROMサイズ一覧を示します。

表９－１　内蔵ROMサイズ一覧

デバイス名称	内蔵ROM容量（バイト）	内蔵ROMマッピング・アドレス
μPD78212	8 K	0000H-1FFFH
μPD78214, 78224, 78234 78244	16 K	0000H-3FFFH
μPD78218A, 78238	32 K	0000H-7FFFH

このアプリケーション・ノートでは，内蔵ROMの内容をチェックする方法として，次に示す３種類を取り上げ，それぞれのプログラム例を示します。

（i）チェック・サム
（ii）外部出力
（iii）内部比較

なお，この応用例は，78K/IIシリーズ構造化アセンブラ（ST78K/II）でソース・プログラムが記述されています。

9.1　チェック・サム

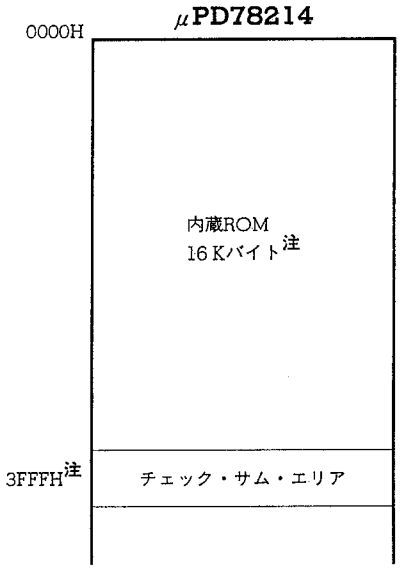
チェック・サムによる内蔵ROMチェックは，内蔵ROMのデータの総和をチェックし，内蔵ROMの内容が正常であるか判断する方法です。

(1) 動作概要

この応用例では，内蔵ROM開始アドレスから内蔵ROM終了アドレス－1までのデータを加算し，その結果を，内蔵ROM終了アドレスにあらかじめ格納しておいたデータと加算することによって，総和のチェックを行います。

この応用例におけるメモリ・マップを図９－１に示します。

図９－１　チェック・サム・プログラムのメモリ・マップ（μPD78214）



注　製品により異なります。**表８－４**を参照してください。

チェック・サム・エリアには，0000H番地から内蔵ROM終了アドレス－1までのデータの総和(X)の２の補数(Y：反転データ＋1)をあらかじめ格納しておきます。チェック・サムの結果，X＋Y＝0ならば内蔵ROMのデータは正常であるということになります。なお，チェック・サム結果はポート０へ出力します。このプログラムでは，チェック結果が正常の場合はFFHを，チェック結果が異常な場合はAAHをポート０へ出力します。

(2) プログラム説明［レーベル名称：SUMROM］… (8) プログラム・リスト参照

① ポート０（チェック・サム結果出力ポート）のイニシャライズを行います。ポート０の初期値には０を設定します。

② アドレス・ポインタのイニシャライズを行います。初期値は内蔵ROM開始アドレスです。HLレジスタをアドレス・ポインタとして使用します。

③ アドレス・ポインタの内容をDレジスタにセットします。Dレジスタをワーク・レジスタとして使用します。

④ アドレス・ポインタが内蔵ROM終了アドレスになるまで，アドレス・ポインタの内容を加算します。

⑤ 加算結果をポート０へ出力します。

加算結果＝０→ROM内容正常

加算結果＝０→ROM内容異常

(3) モード・レジスタ設定例

ポート０モード・レジスタ

	7	6	5	4	3	2	1	0	
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	（リセットでFFH）
設定例	0	0	0	0	0	0	0	0	

PM0	P0モード指定
00H	出力モード
FFH	ハイ・インピーダンス状態
その他	設定禁止

9
214
218A
224
234
244

(4) 入力パラメータ

ADRPS　：チェック開始アドレスです。内蔵ROM開始アドレスとなります。

ADRPED：チェック終了アドレスです。内蔵ROM終了アドレス－1です。

(5) 使用レジスタ

AX, HL, D

(6) プログラム使用例

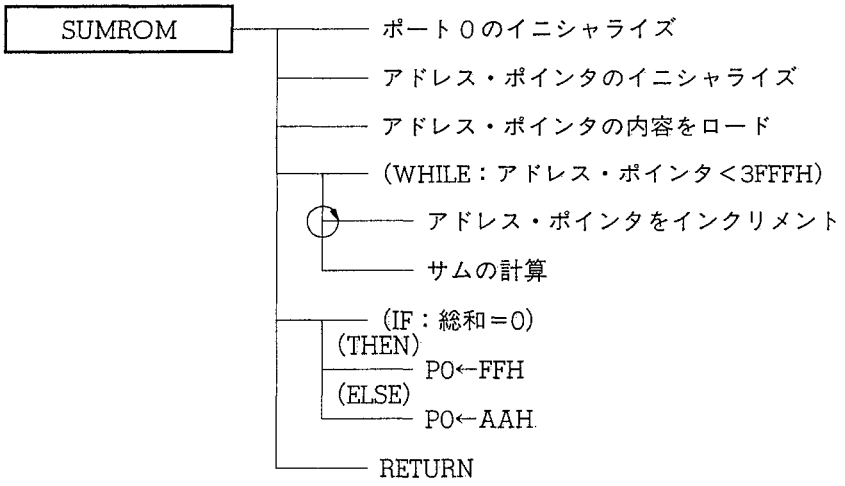
μPD78214の内蔵ROMをチェックするプログラム例を示します。

μPD78214の内蔵ROMは16Kバイトなので，内蔵ROMマッピング・アドレスは0000H-3FFFHです。したがって，チェック開始アドレスは0000H，チェック終了アドレスは3FFFHとなります。またチェック・サム・エリア（3FFFH）には，あらかじめ0000H-3FFFHのデータの総和を定義しておきます（このプログラム例では，41Hと仮定します)。

プログラム例として次のように用います。

```
        PUBLIC  ADRPS,ADRPED
        EXTRN   SUMROM          ; PACKAGE

ADRPS   EQU     0000H           ; ROM check start adress

CKSUMD  EQU     41H             ; check sum data
;
        ORG     3FFFH
ADRPED: DB      CKSUMD          ; check sum area
                :
                :
        CSEG
ROM_S:
        CALL    !SUMROM
                :
                :
```

(7) SPDチャート



(8) プログラム・リスト

```
        NAME    SROM
;
;*************************************************************
;*      internal ROM check by check sum                      *
;*                      for 78214                            *
;*************************************************************
;
        PUBLIC  SUMROM
        EXTRN   ADRPS,ADRPED
;
        CSEG
SUMROM:
        P0=#00H                         ; P0 initialize data set
        PM0=#00H                        ; P0 output mode
        HL=#ADRPS (AX)                  ; set check start address
        D=[HL] (A)
        while(HL<#ADRPED)(AX)
                HL++
                A=D
                A+=[HL]                 ; count sum
                D=A
        endw
        if(D==#0) (A)
                P0=#0FFH
        else
                P0=#0AAH
        endif
        RET
        END
```

9
214
218A
224
234
244

９.２　外部出力

外部出力による内蔵ROMチェックは，内蔵ROMのデータをすべて外部へ出力し，出力したデータが正常なROM内容と一致するかチェックする方法です。

この応用例では，内蔵ROM開始アドレスから内蔵ROM終了アドレスまでのデータを外部へ出力する処理のみを取り上げています。

（１）動作概要

内蔵ROM開始アドレスから内蔵ROM終了アドレスまでのデータを１バイトずつ順番にポート０へ出力します。出力の同期信号として，P34を使用します。P34のレベルをプログラム中で'H'にしたタイミングでデータを出力します。

図９－２　内蔵ROMデータの外部出力動作ブロック図

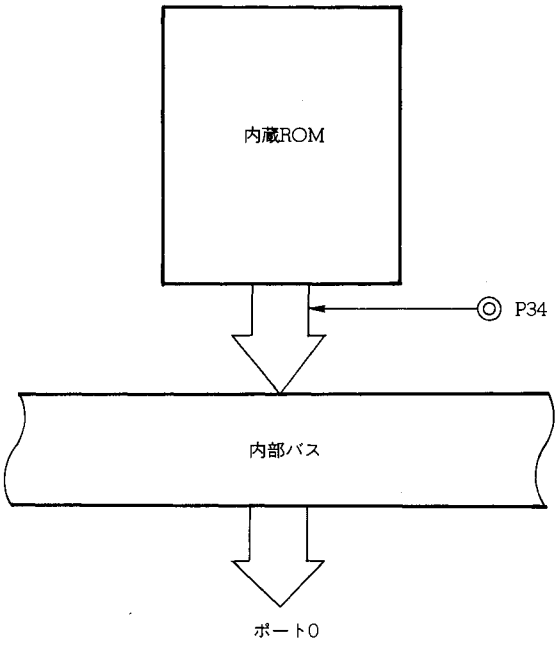
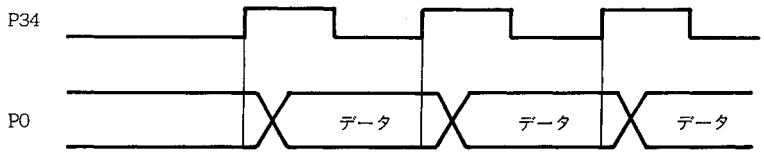


図９－３　ポート０への出力タイミング



P34のハイ・レベル幅とロウ・レベル幅は，ソフトウエアで一定のウエイト周期を作ることによって決定します。

(2) プログラム説明［レーベル名称：EX_ROM］… (8) プログラム・リスト参照

① ポート0（データ出力ポート）のイニシャライズを行います。ポート0の初期値には0を設定します。

② ポート3（P34：出力同期信号）のイニシャライズを行います。ポート3の初期値には0を設定します。

③ アドレス・ポインタのイニシャライズを行います。初期値は内蔵ROM開始アドレスです。HLレジスタをアドレス・ポインタとして使用します。

④ アドレス・ポインタが内蔵ROM終了アドレス＋1になるまでアドレス・ポインタの示すメモリの内容をポート0へ出力します。本プログラムでは，最後のデータ出力終了後にアドレス・ポインタをインクリメントしていますので，データ出力の終了は，内蔵ROMアドレス＋1になった時点で判断します。そのあとポート0にFFHを設定しています。

(3) モード・レジスタ設定例

ポート0モード・レジスタ

	7	6	5	4	3	2	1	0	
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	（リセットでFFH）
設定例	0	0	0	0	0	0	0	0	

PM0	P0モード指定
00H	出力モード
FFH	ハイ・インピーダンス状態
その他	設定禁止

9
214
218A
224
234
244

ポート３モード・コントロール・レジスタ

	7	6	5	4	3	2	1	0	
PMC3	PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30	（リセットで00H）
設定例	0	0	0	0	0	0	0	0	

PMC30	P30端子のコントロール・モード指定
0	入出力ポート・モード
1	RxD入力モード

PMC31	P31端子のコントロール・モード指定
0	入出力ポート・モード
1	TxD出力モード

PMC32	P32端子のコントロール・モード指定
0	入出力ポート・モード
1	$\overline{\text{SCK}}$入出力モード

PMC33	P33端子のコントロール・モード指定
0	入出力ポート・モード
1	SO出力/SB0入出力モード

PMC3n	PM3n端子のコントロール・モード指定（n=4-7）
0	入出力ポート・モード
1	TOm出力モード（m=0-3）

ポート３モード・レジスタ

	7	6	5	4	3	2	1	0	
PM3	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30	（リセットでFFH）
設定例	0	0	0	0	0	0	0	0	

PM3n	P3n端子入出力モード指定（n=0−7）
0	出力モード
1	入力モード

(4) 入力パラメータ

ADRPS：データ出力開始アドレスです。内蔵ROM開始アドレスとなります。

ADRPE：データ出力終了アドレスです。内蔵ROM終了アドレス＋１となります。

(5) 使用レジスタ

AX，BC，HL

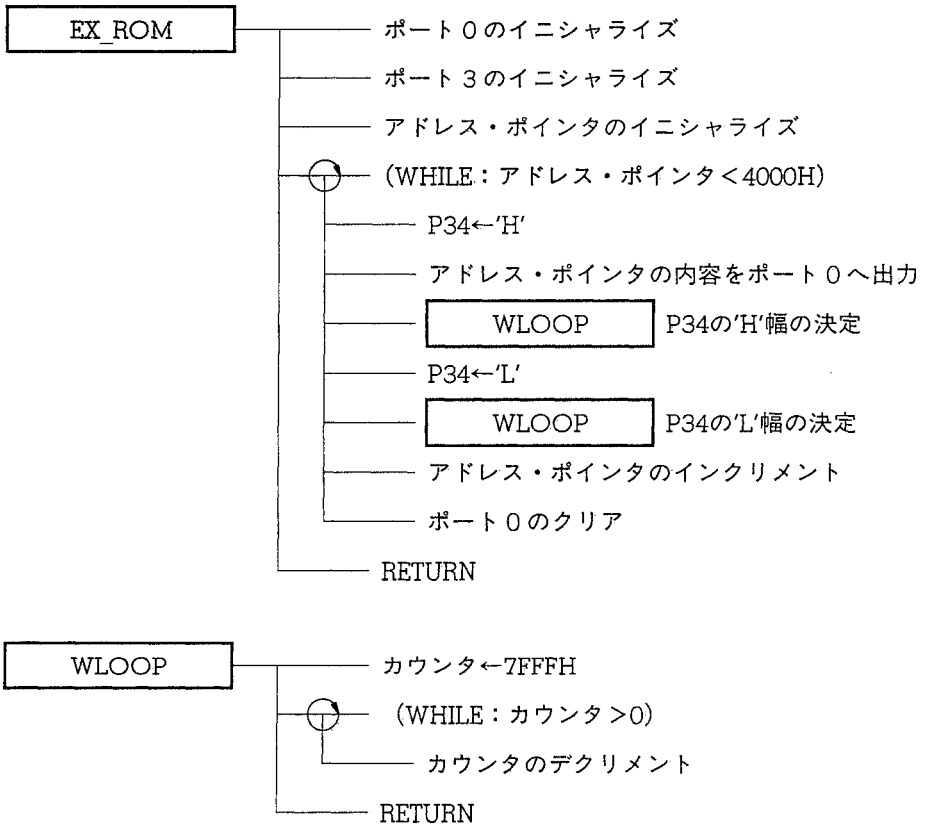
(6) プログラム使用例

μPD78214の内蔵ROMをチェックするプログラム例を示します。μPD78214の内蔵ROMは16 Kバイトなので，内蔵ROMマッピング・アドレスは0000H-3FFFHです。したがって，データ出力開始アドレスは0000H，データ出力終了アドレスは3FFFHとなります。

プログラム例として次のように用います。

```
        PUBLIC  ADRPS,ADRPE
        EXTRN   EX_ROM          ; PACKAGE

ADRPS   EQU     0000H           ; ROM check start address
ADRPE   EQU     4000H           ; ROM check end address
                :
                :
        CSEG
ROM_E:
        CALL    !EX_ROM
                :
                :
```

(7) SPDチャート


EX_ROM
ポート０のイニシャライズ
ポート３のイニシャライズ
アドレス・ポインタのイニシャライズ
(WHILE：アドレス・ポインタ＜4000H)
P34←'H'
アドレス・ポインタの内容をポート０へ出力
WLOOP
P34の'H'幅の決定
P34←'L'
WLOOP
P34の'L'幅の決定
アドレス・ポインタのインクリメント
ポート０のクリア
RETURN
WLOOP
カウンタ←7FFFH
(WHILE：カウンタ＞0)
カウンタのデクリメント
RETURN

(8) プログラム・リスト

```
        NAME    EXROM
;
;*************************************************************
;*      interbal ROM check by data out put                   *
;*                      for 78224                            *
;*************************************************************
;
        PUBLIC  EX_ROM
        EXTRN   ADRPS,ADRPE
        EXTRN   P_OUT
;
SYNCP   EQU     P3.4                    ; data out put synchro port
;
        CSEG
        EX_ROM:
        P0=#00H                         ; P0 initialize data set
        PM0=#00H                        ; P0 output mode
        P3=#00H
        PMC3=#00H
        PM3=#00H
        HL=#ADRPS (AX)                  ; set check start address
        while(HL<#ADRPE)(AX)
                SET1    SYNCP           ; set synchro port
                A=[HL]
                P0=A                    ; data output
                CALL    !WLOOP          ; wait
                CLR1    SYNCP           ; clear synchro port
                CALL    !WLOOP          ; wait
                HL++                    ; incremenat address pointer
                P0=#0FFH                ; clear P0
        endw
        RET
WLOOP:                                  ; wait routine
        BC=#07FFFH
        while(BC>#0)(AX)
                BC--
        endw
        RET
        END
```

9
214
218A
224
234
244

９.３　内部比較

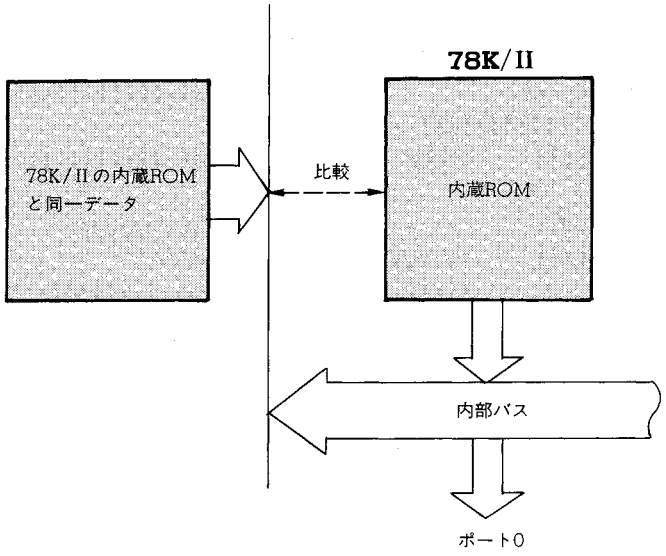
内部比較による内蔵ROMチェックは，外部から入力されたデータと，78K/IIの内蔵ROMデータを比較し，内蔵ROMの内容が正常であるか判断する方法です。

(1) 動作概要

78K/IIの外部に内蔵ROMデータと同一のデータを用意しておき，そのデータを１バイトずつ78K/II側へ入力します。そして，内蔵ROM開始アドレスから内蔵ROM終了アドレスまでのデータと外部からの入力データとの比較を行い，比較結果をポート０へ出力します。このプログラムでは，チェック結果が正常な場合はFFHを，チェック結果が異常な場合はAAHをポート０へ出力します。

図９－４に動作ブロック図を示します。

図９－４　内蔵ROMデータの内部比較の動作ブロック図



この応用例では，外部から１バイト入力されるごとに割り込みが発生するものとし，その割り込み処理中で内蔵ROMデータとの比較処理を行っています。この応用例では，割り込みソースの種類は特に限定していません。

１バイトの比較が終了するごとに比較結果を判断し，フラグにその情報を保持しておきます。全データの比較が終了した時点での比較結果フラグの内容によって，結果をポート０へ出力します。比較結果フラグは，比較結果が不一致となった時点でセット(1)されます。

この応用例において使用するワーク・エリアを表９－２に示します。

表９－２　内部比較プログラムで使用するワーク・エリア

ワーク・エリア名称	用　　途	バイト数
ADRP	比較アドレスを示すポインタを格納する	２バイト
CMPF	比較結果の情報を格納するフラグ	１ビット

(2) プログラム説明

(a) メイン処理［レーベル名称：CMPROM］… (8) プログラム・リスト参照

処理に必要なイニシャライズ，全データの比較終了の判断，結果の出力を行います。

(i) イニシャライズ処理［レーベル名称：CMPINI］

① ポート０（比較結果出力ポート）のイニシャライズを行います。ポート０の初期値は00Hとします。

② アドレス・ポインタ（ADRP）のイニシャライズを行います。初期値は内蔵ROM開始アドレスです。このプログラム例では，割り込み処理で比較処理を行っているため，現在のアドレス・ポインタをRAMに設けたエリア（ADRP）に毎回格納しています。

③ 比較結果フラグ（CMPF）のイニシャライズを行います。初期状態は'0'（比較結果一致状態）です。

9
214
218A
224
234
244

(ii) 比較結果の判断［レーベル名称：CMPEND］

① アドレス・ポインタ（ADRP）が内蔵ROM終了アドレス＋1になるまでウエイトします。アドレス・ポインタは，ウエイト中に発生する割り込み要求処理内で更新されます。このプログラムでは，1バイトの比較終了後にアドレス・ポインタを更新するので，全データの比較終了時のアドレス・ポインタは，内蔵ROMアドレス＋1となります。

② 全データの比較が終了したら，比較結果をポート0へ出力します。

CMPF＝0（全データ一致）　→ROM内容正常

CMPF＝1（不一致データ検出）→ROM内容異常

(b) 割り込み処理（比較処理）［レーベル名称：CMP_IN］

割り込み処理中でこのサブルーチンを呼び出します。ただし，Bレジスタに外部入力データが格納されていることを入力条件とします。

① アドレス・ポインタ（ADRP）をロードします。HLレジスタを介してアドレス・ポインタの更新を行います。

② アドレス・ポインタの示すメモリの内容と外部入力データ（Bレジスタ）を比較します。

③ 比較結果をゼロ・フラグ（Zフラグ）によって判断します。比較結果フラグ（CMPF：初期状態は0）とZフラグの反転の論理和をとることにより，CMPFに比較結果（0あるいは1）が格納されます。CMPFは一度セット（1）されたら，全データの比較が終了するまでセットされたままです。

④ アドレス・ポインタをインクリメントし，ADRPへポインタ値を格納します。

(3) モード・レジスタ設定例

ポート0モード・レジスタ

	7	6	5	4	3	2	1	0	
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	（リセットでFFH）
設定例	0	0	0	0	0	0	0	0	

PM0	P0モード指定
00H	出力モード
FFH	ハイ・インピーダンス状態
その他	設定禁止

(4) 入力パラメータ

ADRPS：比較開始アドレスです。内蔵ROM開始アドレスとなります。

ADRPE：比較終了アドレスです。内蔵ROM終了アドレス＋1となります。

(5) 使用レジスタ

メイン処理　：AX

割り込み処理：AX，HL，B，CY（ビット・アキュームレータ）

備考　割り込み処理の先頭でレジスタ・バンクを切り替えてください。

(6) プログラム使用例

μPD78214の内蔵ROMをチェックするプログラム例を示します。μPD78214の内蔵ROMは16 Kバイトなので，内蔵ROMマッピング・アドレスは0000H-3FFFHです。したがって，比較開始アドレスは0000H，比較終了アドレスは3FFFHとなります。

プログラム例として次のように用います。

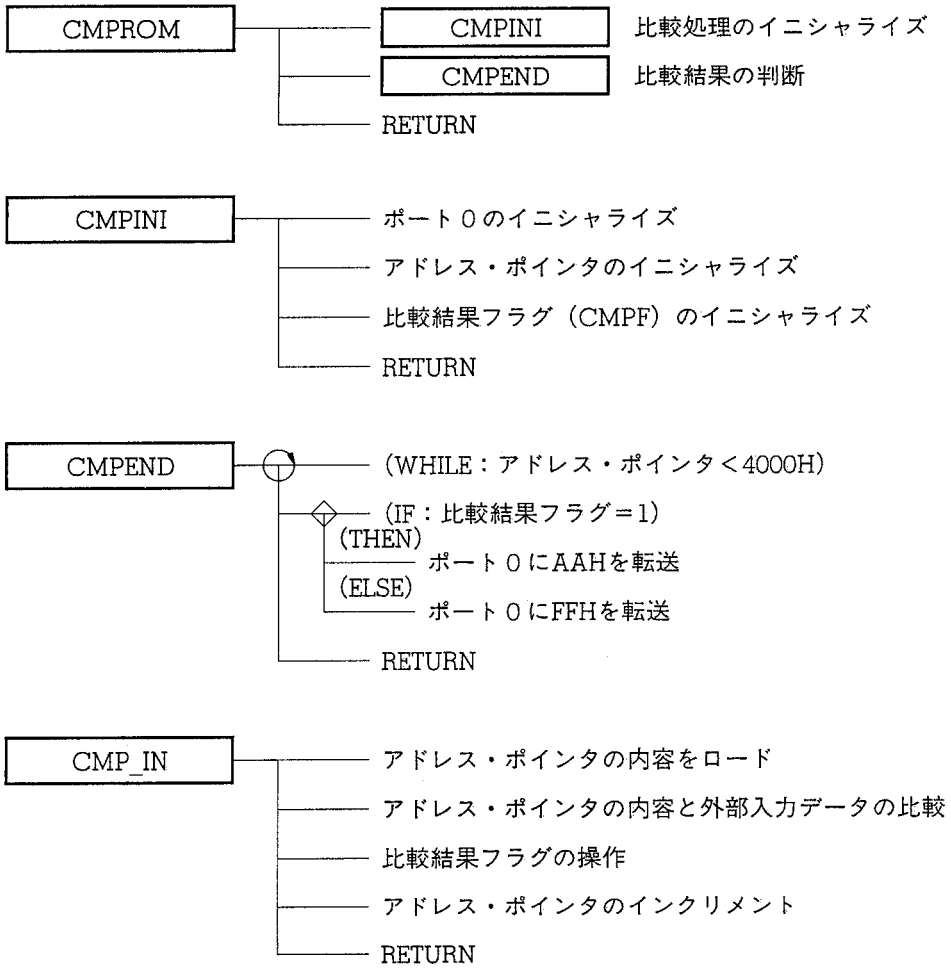
(i) メイン処理

```
        PUBLIC  ADRPS, ADPRS, ADRPE
        PUBLIC  CMPF
        EXTRN   CPMROM          ; PACKAGE

ADRPS   EQU     0000H           ; ROM check start address
ADPRE   EQU     4000H           ; ROM check end address

        BSEG
CMPF    DBIT

        DSEG    SADDR
ADRP:   DS      2

                :
                :
        CSEG
ROM_C:
        CALL    !CMPROM
                :
                :
```

(ii) 割り込み処理

```
        EXTRN   CMP_IN          ; COMPARE PACKAGE
INT:
                :
                :
        CALL    !CMP_IN
                :
                :
        RETI
                :
                :
```

(7) SPDチャート



9
214
218A
224
234
244

(8) プログラム・リスト

```
        NAME    CMPR
;
;*************************************************************
;*      interbal ROM check by internal compare               *
;*                      for 78224                            *
;*************************************************************
;
        PUBLIC  CMPROM, CMP_IN
        EXTRN   ADRP, ADRPS, ADRPE
        EXTRN   SBIINI
        EXTBIT  CMPF
;
ZFLG    EQU     PSW.6
;
CMPROM:
        CSEG
        CALL    !CMPINI         ; compare initialize
        CALL    !SBIINI
        CALL    !CMPEND         ; compare result
        RET
CMPINI:
        P0=#00H                 ; set P0 initialize data
        PM0=#00H                ; P0 output port
        ADRP=#ADRPS             ; address pointer initialize
        CLR1    CMPF            ; compare flag initialize
        RET
;
CMPEND:
        while(ADRP<#ADRPE)(AX)
        endw
        if_bit(CMPF)            ; set CMPF?
                P0=#0AAH
        else
                P0=#0FFH
        endif
        RET
;*************************************************************
;*              ROM data internal compare                    *
;                    B : compare data                       *
;*************************************************************
CMP_IN:
        HL=ADRP (AX)            ; road address pointer
        A=[HL]
        A-=B                    ; compare with external data
        CY=CMPF
        CY|=/ZFLG               ; CY = CY OR (NOT ZFLG)
        CMPF=CY
        HL++
        ADRP=HL (AX)            ; save address pointer
        RET
;
        END
```

第10章　SBIの応用

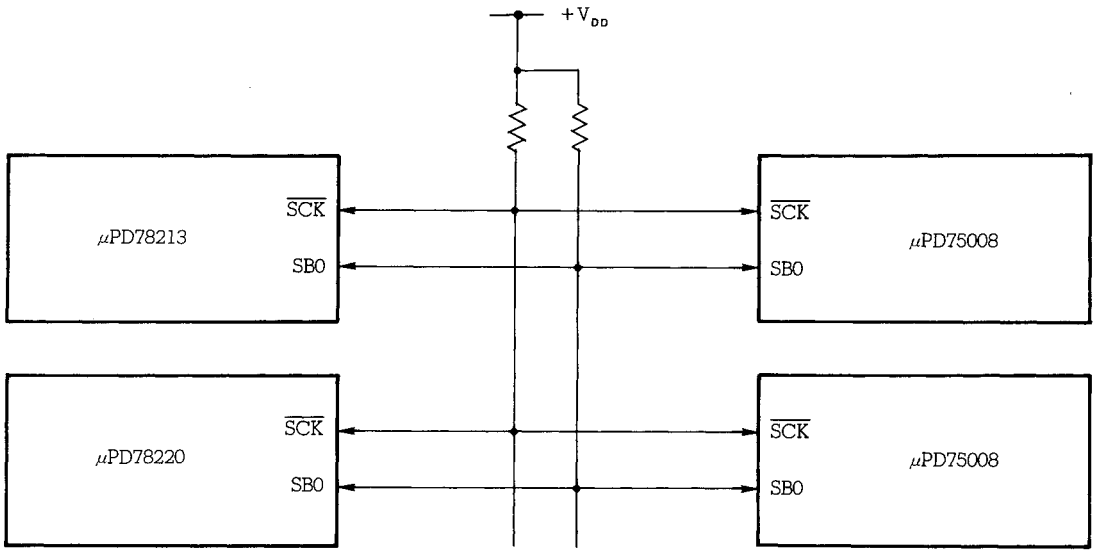
10.1　機能概要

ここでは，SBIの応用例の機能概要について説明します。

10.1.1　接 続 図

この応用例では，μPD78213，78220および２つの75XシリーズのCPUをSBIで接続しています。図10－１にSBIシステム構成例を示します。

図10－１　SBI接続図



10.1.2　アドレス，コマンドの定義

(1) アドレス

この応用例ではアドレスを特に限定していません。よって，バックグランド処理のシステム・イニシャライズ処理中などで，ワークRAMのアドレス格納領域（SVA）にアドレスを設定してください。

```
例  SVA     EQU     0FEC7H                  ; Address Area
              .
              .
              .
            MOV     SVA, #03H
```

(2) コマンド

この応用例で使用しているコマンドを表10－1に示します。動作や詳細については，**10.4　動作説明**を参照してください。

表10－1　コマンド

コマンド	値	意　　味
WRITE	20H	マスタ→スレーブへのデータ転送
READ	21H	スレーブ→マスタへのデータ転送
LWRITE	22H	マスタがデータ数を指定しての，マスタ→スレーブへのデータ転送
LREAD	23H	マスタがデータ数を指定しての，スレーブ→マスタへのデータ転送
MACW	24H	マスタ→スレーブへのマクロ・サービスによるデータ転送
MACR	25H	スレーブ→マスタへのマクロ・サービスによるデータ転送
CHGMST	28H	スレーブへのマスタの権利の受け渡し
DETACH	29H	現在選択されているスレーブを非選択に設定
DSPON	31H	スタティックLEDの表示オン
DSPOFF	32H	スタティックLEDの表示オフ

10.1.3　構成概要

この応用例は，SBIの応用のパッケージと，それを動作させる環境としてのバックグランド処理から構成されています。ここでは，SBIのパッケージのすべてと，その使用例として，バックグランド処理の一部について取り上げます。

この応用例のコマンド処理は，バックグランド処理から呼び出され，コマンド処理内で，すべての動作が完結します。

10.1.4　状態遷移図

図10－2にマスタ，図10－3にスレーブの状態遷移図を示します。

図10－2　マスタの状態遷移図

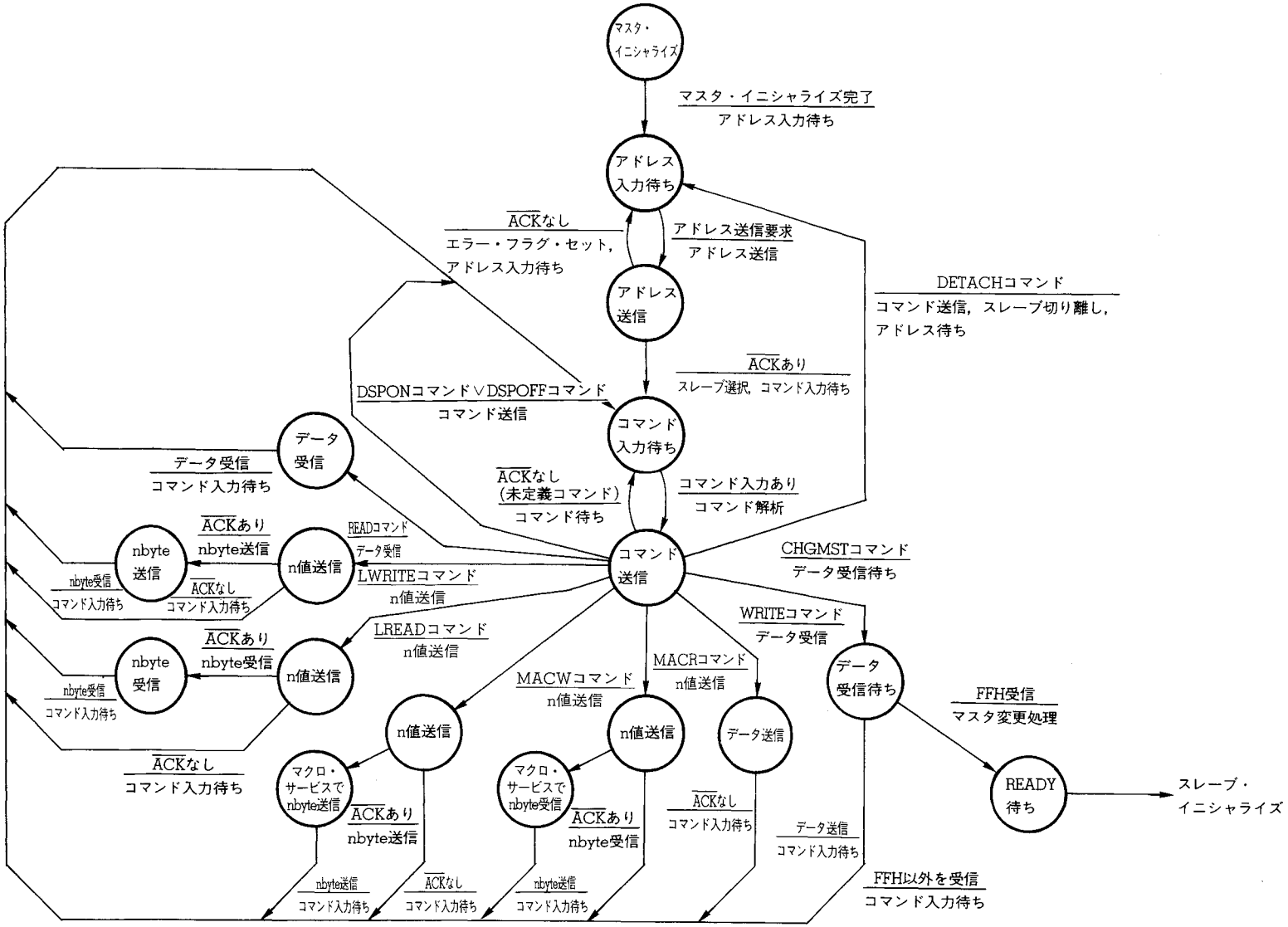
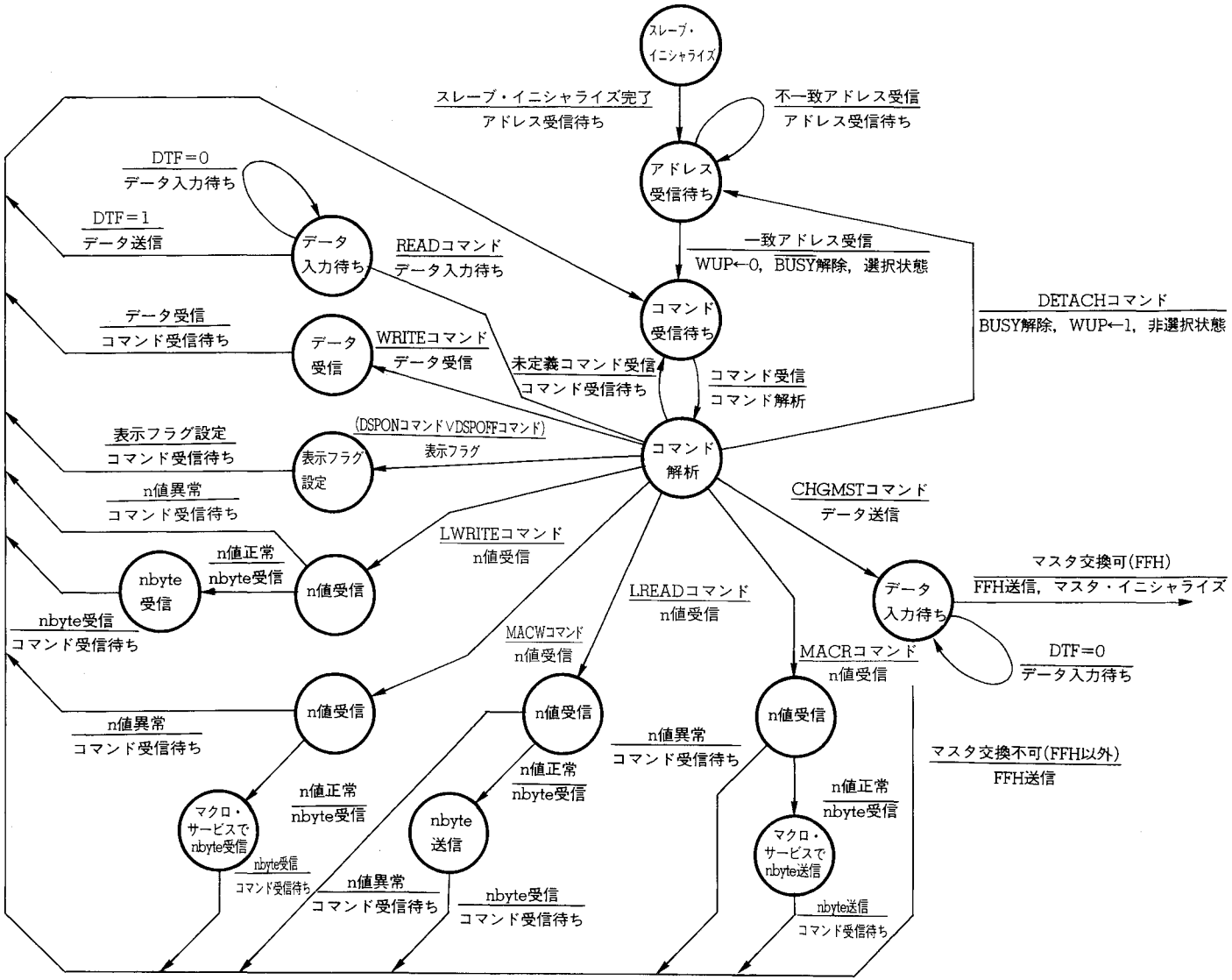


図10-3 スレーブの状態遷移図



10.2　応用プログラムの環境

SBIのパッケージを動作させる環境として，RAM，フラグなどのワーク・エリアおよびインターバル・タイマ割り込み要求を必要とします。ワーク・エリアは再配置が可能です（ただし，ショート・ダイレクト・アドレシングの適用範囲にある必要があります)。したがって，パッケージを使用するモジュール上でパブリック宣言（PUBLIC)，外部参照宣言（EXTRN，EXTBIT）およびEQUの疑似命令で必要なエリアを定義します。以降に，ワーク，シンボル定義，およびインターバル・タイマ割り込みについて解説します。

10.2.1　ワークRAM

この応用例のSBI部分で使用しているワークRAMを表10－2に示します。

表10－2　ワークRAM

レーベル	バイト数	初期値	用　　途
RCVBUF	1	00H	受信バッファ
TRNBUF	1	00H	送信バッファ
WRDATA	1	00H	WRITEコマンド用1バイト送信データ
LPARAM	1	00H	LWRITE，LREAD，MACW，MACRコマンド用送受信データ数
LSTART	2	00H	LWRITE，LREAD，MACW，MACRコマンド用送受信バッファ先頭アドレス
ACKCNT	1	00H	$\overline{\text{ACK}}$待ちカウンタ
MACCNT	1	00H	マクロ・サービス・タイム・アウト・カウンタ
SVA	1	00H	アドレス格納領域
SBIFL0	1	00H	SBI用フラグ・ワーク領域0
SBIFL1	1	00H	SBI用フラグ・ワーク領域1
ETCFLG	1	00H	その他のフラグ・ワーク領域
MSMCSI	1	00H	INTCSI用マクロ・サービス・モード・レジスタ
CHPCSI	1	00H	INTCSI用チャネル・ポインタ

10.2.2　使用フラグ

この応用例のSBI部分で使用している使用フラグを表10－3に示します。

表10－3　使用フラグ

レーベル	ビット	名称	初期値	用　途	0の場合の意味	1の場合の意味
SBIFL0	0	ERRORF	0	転送エラー状態	エラーなし	エラー発生
	1	TREND	0	転送完了状態	非転送完了状態	転送完了状態
	2	ACKWFG	0	$\overline{\text{ACK}}$待ち状態	非$\overline{\text{ACK}}$待ち状態	$\overline{\text{ACK}}$待ち状態
	3	RCVFLG	0	送受信状態	送信	受信
	4	PFLAG	0	パラメータの有無	パラメータなし	パラメータあり
	5	CMDBSY	0	コマンド処理状態	非コマンド処理中	コマンド処理中
	6	未使用	0	未使用	—	—
	7	MSTF	0	マスタ・フラグ	スレーブ	マスタ
SBIFL1	0	DTF	0	データ送信要求	なし	あり
	1	CRF	0	コマンド受信状態	受信なし	受信あり
	2	CTF	0	コマンド送信要求	なし	あり
	3	ATF	0	アドレス送信要求	なし	あり
	4	MACTMF	0	マクロ・サービス タイム・アウト状態	非タイム・アウト待ち	タイム・アウト待ち
	5	未使用	0	未使用	—	—
	6	未使用	0	未使用	—	—
	7	MACF	0	マクロ・サービス 使用状態	通常転送	使用中
—	—	STLEDF	0	スタティックLEDの表示	表示ON	表示OFF

10
214
218A
224
234
244

10.2.3　シンボル定義

10.2.1および**10.2.2**で取り上げたRAMやフラグを用いているため，SBIモジュールでは，アセンブラの疑似命令でパブリック宣言（PUBLIC），外部参照宣言（EXTRN，EXTBIT）を行っています。また，タイマとシリアル転送に割り込みを用いていますので，ベクタ・テーブルの設定が必要です。さらに，sfr. bitのEQU疑似命令での定義を必要とします。

次に，この応用例で記述しているシンボル定義を示します。

―― バックグランド・モジュールでのシンボル定義 ――

```
;
;       --- module define ---
;
        PUBLIC  LPARAM, WRDATA
        PUBLIC  LSTART
        PUBLIC  RCVBUF, TRNBUF
        PUBLIC  ACKCNT, MACCNT
        PUBLIC  SVA
        PUBLIC  SBIFL0, SBIFL1

        PUBLIC  ERRORF                  ; error of Transmit or Receive
        PUBLIC  TREND                   ; Complete of Transmit or Receive
        PUBLIC  ACKWFG                  ; ACK wait
        PUBLIC  RCVFLG                  ; Mode of Transmit or Receive
        PUBLIC  PFLAG                   ; Parameter
        PUBLIC  CMDBSY                  ; Busy with Command Task
        PUBLIC  MSTF                    ; Master or Slave

        PUBLIC  DTF                     ; Request of Data Transmit
        PUBLIC  CRF                     ; Request of Command Receive
        PUBLIC  CTF                     ; Request of Command Transmit
        PUBLIC  ATF                     ; Request of Address Transmit
        PUBLIC  MACF                    ; Transmit by Macro Service Mode
        PUBLIC  MACTMF                  ; Macro Service Time Out Mode

        PUBLIC  STLEDF                  ; Statick LED ON/OFF

        EXTRN   ACKCHK                  ; ACK check (INTC21)
        EXTRN   MACOUT                  ; Time Out of Macro Service
        EXTRN   M_INIS, S_INIS          ; Initialize Package
        EXTRN   TRNADR, TRNCMD, TRNDAT  ; transmint or receive
        EXTRN   RCVDAT, RCVMAC
        EXTRN   M_WR, M_RD, M_LWR, M_LRD    ; command
        EXTRN   M_MACW, M_MACR, M_ON, M_OFF
        EXTRN   M_DTC, M_CMT
        EXTRN   S_WR, S_RD, S_LWR, S_LRD
        EXTRN   S_MACW, S_MACR, S_ON, S_OFF
        EXTRN   S_DTC, S_CMT
```

```
;
;       --- work flags  define ---
;
ERRORF  EQU     SBIFL0.0                ; error of Transmit or Receive
TREND   EQU     SBIFL0.1                ; Complete of Transmit or Receive
ACKWFG  EQU     SBIFL0.2                ; ACK wait
RCVFLG  EQU     SBIFL0.3                ; Mode of Transmit or Receive
PFLAG   EQU     SBIFL0.4                ; Parameter
CMDBSY  EQU     SBIFL0.5                ; Busy with Command Task
MSTF    EQU     SBIFL0.7                ; Master or Slave

DTF     EQU     SBIFL1.0                ; Request of Data Transmit
CRF     EQU     SBIFL1.1                ; Request of Command Receive
CTF     EQU     SBIFL1.2                ; Request of Command Transmit
ATF     EQU     SBIFL1.3                ; Request of Address Transmit
MACF    EQU     SBIFL1.4                ; Transmit by Macro Service Mode
MACTMF  EQU     SBIFL1.7                ; Macro Service Time Out Mode

STLEDF  EQU     ETCFLG.0                ; Static LED ON/OFF
;
;       --- Vector Table define ---
;
        RSTVENT
        CVENT21
```

SBIモジュールでのシンボル定義

```
;
;       --- module define ---
;
        PUBLIC  M_INIS,S_INIS           ; INITIALIZE
        PUBLIC  ACKCHK                  ; ACK CHECK
        PUBLIC  MACOUT                  ; MACRO SERVICE TIME OUT
        PUBLIC  TRNADR,TRNCMD,TRNDAT,RCVDAT,RCVMAC
        PUBLIC  M_WR,S_WR,M_RD,S_RD     ; COMMNDS
        PUBLIC  M_LWR,S_LWR,M_LRD,S_LRD
        PUBLIC  M_MACW,S_MACW,M_MACR,S_MACR
        PUBLIC  M_ON,S_ON,M_OFF,S_OFF
        PUBLIC  M_DTC,S_DTC
        PUBLIC  M_CMT,S_CMT

        EXTRN   LPARAM                  ; Max Length of Long Data
        EXTRN   WRDATA                  ; Write Command's Data
        EXTRN   LSTART                  ; Long Data Area
        EXTRN   RCVBUF                  ; Receive Buffer
        EXTRN   TRNBUF                  ; Transmit Buffer
        EXTRN   ACKCNT                  ; ACK Counter
        EXTRN   MACCNT                  ; Time Out Counter for Macro Service
        EXTRN   SVA                     ; Address Area

        EXTRN   SBIFL0                  ; SBI Work Flags Area 0
        EXTRN   SBIFL1                  ; SBI Work Flags Area 1

        EXTBIT  ERRORF                  ; Error of Transmit or Receive
        EXTBIT  TREND                   ; Complete of Transmit or Receive
        EXTBIT  ACKWFG                  ; ACK wait
        EXTBIT  RCVFLG                  ; Mode of Transmit or Receive
        EXTBIT  PFLAG                   ; Parameter
        EXTBIT  CMDBSY                  ; Busy with Command Task
        EXTBIT  MSTF                    ; Master or Slave
        EXTBIT  MACF                    ; Transmit by Macro Service Mode
        EXTBIT  MACTMF                  ; Macro Service Time Out Mode

        EXTBIT  DTF                     ; Request of Data Transmit
        EXTBIT  CRF                     ; Request of Command Receive
        EXTBIT  CTF                     ; Request of Command Transmit
        EXTBIT  ATF                     ; Request of Address Transmit

        EXTBIT  STLEDF                  ; Static LED ON/OFF
;
;       --- sfr bits define ---
;
SCK     EQU     P3.2
```

10.2.4　インターバル・タイマ割り込み

この応用例では，SBIの動作をモニタするために，$\overline{\text{ACK}}$チェックと，マクロ・サービス・タイム・アウトにインターバル・タイマ割り込みを使用しています。この応用例では，８ビット・タイマ／カウンタ２のINTC21割り込みを1.92 msごとに発生させ，この割り込み処理ルーチン中で，各種フラグの状況に応じて，SBIをモニタするモジュール（ACKCHK，MACOUT）を呼び出しています。

ここでは，1.92 msごとにINTC21割り込み要求を発生させるために，次のようにモード・レジスタの設定を行っています。

① PRM1 … 01100000B
② CRC2 … 00011000B
③ CR21 … 44
④ CMK21 … 0

10.3　基本通信手段

ここでは，マスタ，スレーブ間における基本通信の，ソフトウエアによる実現手段について解説します。

10.3.1　1バイト送受信

(1) マ ス タ

マスタの送信処理では，送信バッファ(TRNBUF)の内容を読み出して，アドレス／コマンド／データの区別をつけて送信します。また，受信処理は，通常はシフト・レジスタへのFFHの書き込みで起動します。

アドレス送信要求とコマンド送信要求は，ATF，CTFフラグのセットで発生します。データ送受信は，常にコマンド処理内で行われ，送信は，あらかじめメモリに設定しておいたデータが送信され，受信は，あらかじめ確保していたメモリに受信データが格納されます。

マスタの送受信用のフラグとして，次のフラグを用いています。

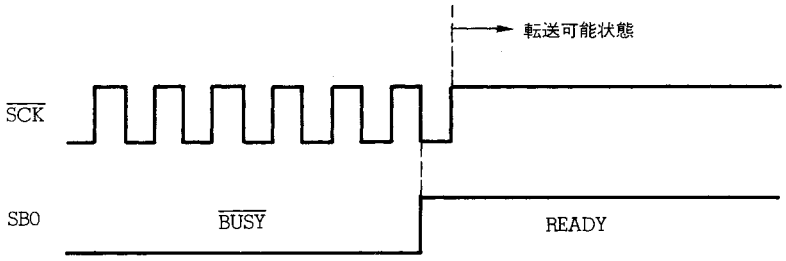
表10－4　マスタの送受信用フラグ

フラグ名称	初期値	用　　途	0の場合の意味	1の場合の意味
MSTF	1	マスタ状態	スレーブ	マスタ
TREND	0	転送完了状態	非転送完了状態	転送完了状態
ERRORF	0	転送エラー状態	エラーなし	エラー発生
ACKWFG	0	$\overline{\text{ACK}}$待ち状態	非$\overline{\text{ACK}}$待ち状態	$\overline{\text{ACK}}$待ち状態
RCVFLG	0	送受信状態	送信	受信
ATF	0	アドレス送信要求	なし	あり
CTF	0	コマンド送信要求	なし	あり

(a) バス・チェック処理

送信動作の前に，データ・ライン（SBO端子）がハイ・レベルか，またクロックが停止しているかを，SBO，$\overline{SCK}$の端子レベルを読むことによってチェックします。

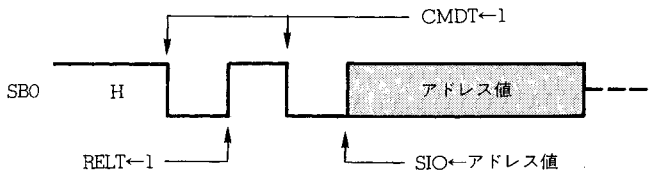
図10－4　バス・チェック



(b) アドレス送信処理

バス・チェック後，コマンド信号，バス・リリース信号，コマンド信号と順次出力して，**(d) データ送信処理**で，アドレス値をシフト・レジスタに書き込みます。

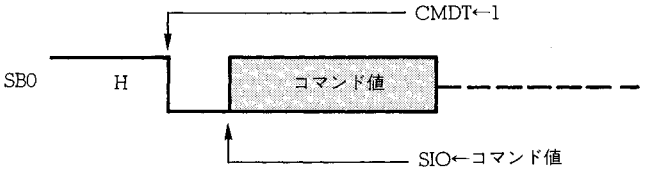
図10－5　アドレス送信



(c) コマンド送信処理

バス・チェック後，コマンド信号を出力して**(d) データ送信処理**で，コマンド値をシフト・レジスタに書き込みます。

図10－6　コマンド送信

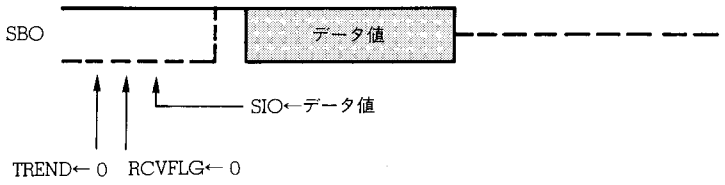


10
214
218A
224
234
244

(d) データ送信処理

TREND←0，RCVFLG←0を行い，データ値をシフト・レジスタに書き込みます。この際，バス・チェックは行いません（$\overline{\text{BUSY}}$中に転送スタートした場合は，READYになり次第，転送がスタートされます）。

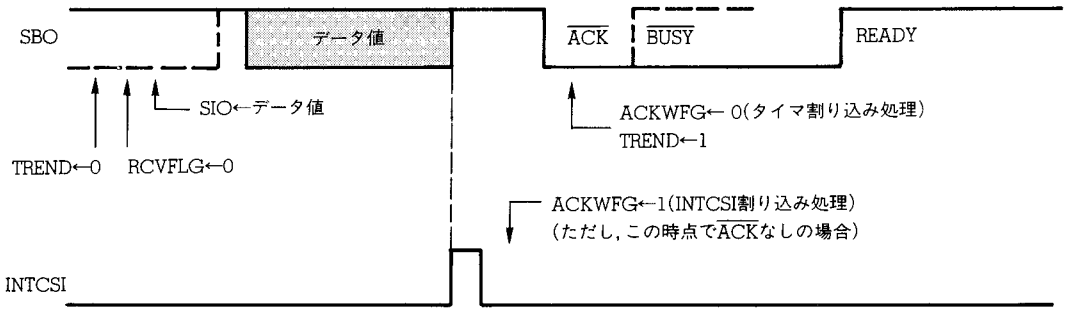
図10－7　データ送信



(e) データ送信終了処理

データ送信起動後，送信終了（$\overline{\text{ACK}}$受信）までウエイトします。

図10－8　送信終了待ち

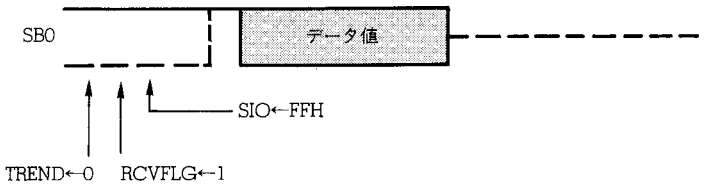


転送完了フラグ（TREND）は，$\overline{\text{ACK}}$検出もしくは，$\overline{\text{ACK}}$タイム・アウトでセットされます（**10.3.3　タイマ割り込み**参照）。

(f) データ受信処理

TREND←0，RCVFLG←1を行い，FFHをシフト・レジスタに書き込みます。この際，バス・チェックは行いません($\overline{\text{BUSY}}$中に転送スタートした場合は，READYになり次第，転送がスタートされます)。

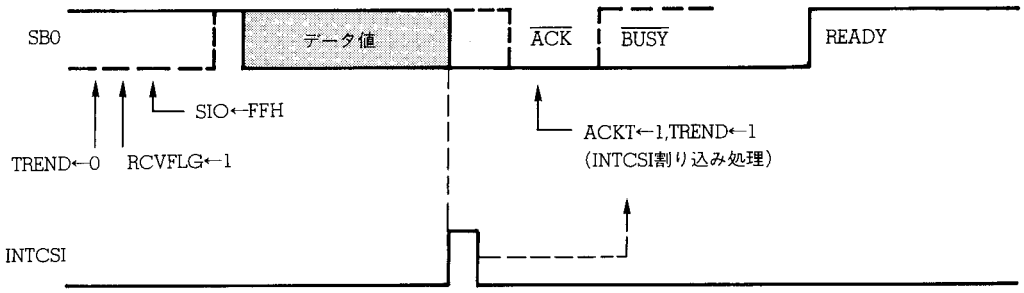
図10－9　データ受信



(g) データ受信終了処理

データ受信起動後，シリアル転送完了割り込み処理で，$\overline{\text{ACK}}$を出力します。

図10－10　受信終了待ち



10
214
218A
224
234
244

(2) スレーブ

スレーブの受信処理は，シリアル転送完了割り込み処理で行われます。割り込み処理内では，$\overline{\text{ACK}}$の送信，ワーク・エリア操作，およびフラグ操作までを行い，アドレス／コマンド受信待ち，またはコマンド処理に移行します。

スレーブの送信処理は，マスタから送信されたコマンド(READ，CHGMST，LREAD，MACRコマンド）によってのみ行われます。この処理にエントリした状態では，バス・ラインは$\overline{\text{BUSY}}$の状態となっています。$\overline{\text{BUSY}}$は，SIOにマスタ送信するデータを書き込んだ時点で解除されます。

スレーブ送信用のフラグとして，次のフラグを用いています。

表10－5　スレーブの送受信用フラグ

フラグ名称	初期値	用　途	0の場合の意味	1の場合の意味
MSTF	0	マスタ状態	スレーブ	マスタ
TREND	0	転送完了状態	非転送完了状態	転送完了状態
ERRORF	0	転送エラー状態	エラーなし	エラー発生
RCVFLG	1	送受信状態	送信	受信
ACKWFG	0	$\overline{\text{ACK}}$待ち状態	非$\overline{\text{ACK}}$待ち状態	$\overline{\text{ACK}}$待ち状態
PFLAG	0	パラメータの有無	パラメータなし	パラメータあり
CRF	0	コマンド受信状態	受信なし	受信あり
DTF	0	データ送信要求	なし	あり
CMDBSY	0	コマンド処理状態	非コマンド処理中	コマンド処理中

10.3.2　シリアル転送完了割り込み

(1) マ ス タ

(a) 送信終了 (RCVFLG=0)

$\overline{\text{ACK}}$の受信をチェックします。$\overline{\text{ACK}}$を受信していれば，TREND←1を行います。$\overline{\text{ACK}}$を受信していなければ，$\overline{\text{ACK}}$待ち状態に設定します（ACKWFG←1)。

(b) 受信終了 (RCVFLG=1)

受信データを，受信バッファ（RCVBUF）に格納します。受信をスレーブに知らせるために，$\overline{\text{ACK}}$を送信し，TREND←1を行います。

(2) スレーブ

(a) 送信終了 (RCVFLG=0)

$\overline{\text{ACK}}$の受信を行います。スレーブでは$\overline{\text{ACK}}$のタイム・アウト処理は行わず，$\overline{\text{ACK}}$を受信するまで待ち続けます。$\overline{\text{ACK}}$を受信した場合は，TREND←1を行います。

(b) 受信終了 (RCVFLG=1)

受信データを，受信バッファ（RCVBUF）に格納します。そのあと，アドレス，コマンド，データに応じた処理を行います。受信内容を判別し，不適当な受信であれば，$\overline{\text{ACK}}$は出力せず，$\overline{\text{BUSY}}$の解除条件の成立のみを行います。

最後に，転送完了フラグをセットし（TREND←1)，バックグランドに戻します。

10
214
218A
224
234
244

10.3.3　タイマ割り込み

タイマ割り込み処理として，$\overline{ACK}$待ち処理と，マクロ・サービスのタイム・アウト処理を行います。

(1) $\overline{ACK}$待ち処理

(a) マ ス タ

INTC21割り込みを用いて$\overline{ACK}$の監視を行います。

INTC21割り込みは，1.92 msごとに発生させ，シリアル転送完了割り込み発生後，非同期で，16回サンプリングする間に$\overline{ACK}$を検出できなければエラーとします。このあと，マスタから強制的に$\overline{ACK}$を出力します。

8ビット・タイマ／カウンタのカウント・クロック＝$f_{CLK}/256$

CR21値＝44

→$\overline{ACK}$待ち時間

＝1.92(ms)×(16−1)−1.92(ms)×16

＝28.80(ms)−30.72(ms)

(b) スレーブ

スレーブでは，スレーブからマスタへデータを送信した場合に$\overline{ACK}$待ち処理が必要です。INTC21による$\overline{ACK}$の監視は行いますが，マスタとは異なり，マスタから$\overline{ACK}$が出力されるまで待ち続けます。

(2) マクロ・サービス・タイム・アウト処理

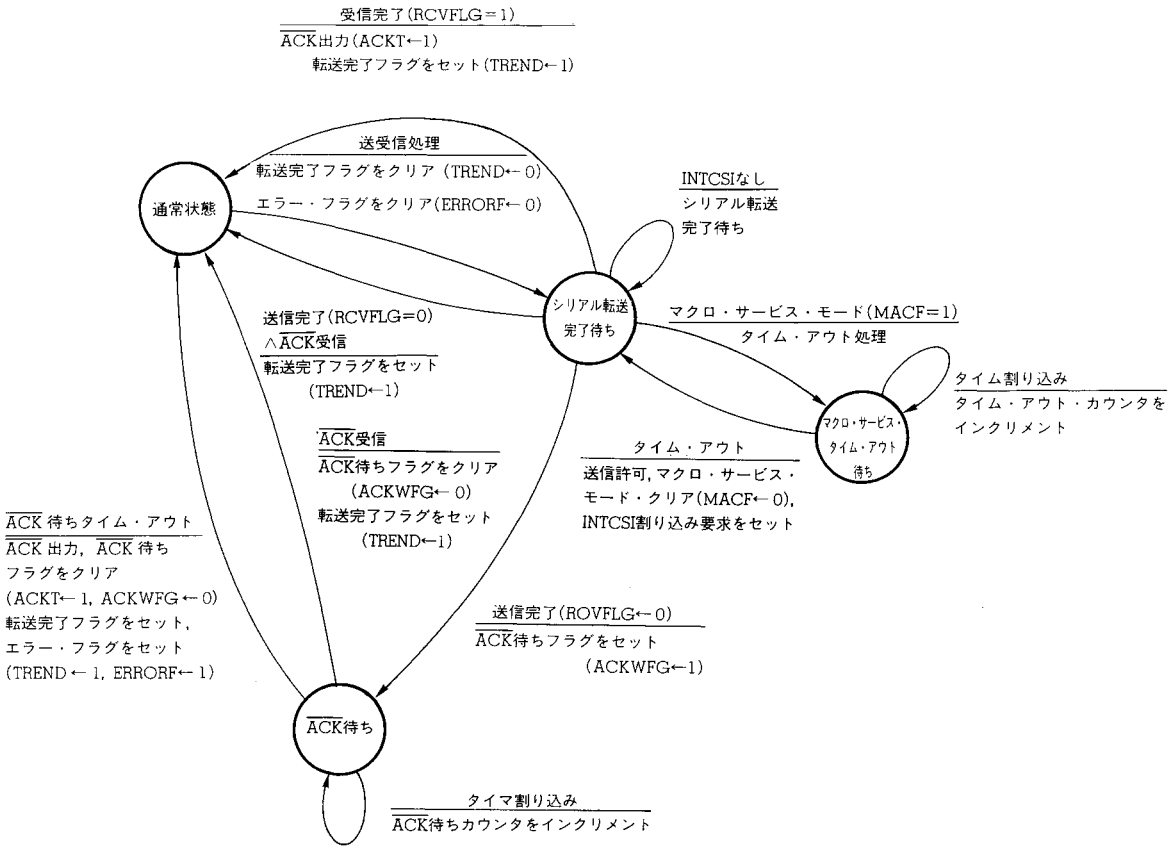
マクロ・サービスによるデータ転送では，マクロ・サービス完了割り込みと，最後のデータ転送時のシリアル転送完了割り込みが連続して，同一割り込みソースで発生し，2回の割り込み要求が1回分しか行われない可能性があります。

ここで，確実に割り込み処理を2回実行させるために，1回目のINTCSI発生時にタイム・アウト待ち状態に設定し，2回目のINTCSIを待ちます。タイム・アウト待ち状態中に，INTC21割り込みを2回カウントしたのち，INTCSI割り込み要求を強制的に発生させています。詳細は**10.4.4　マクロ・サービス・タイム・アウト処理**を参照してください。

10.3.4　1バイト送受信の状態遷移図

図10－11に1バイト送受信の状態遷移図を示します。

図10－11　1バイト送受信の状態遷移図



10.4　動作説明

ここでは，SBIの各モジュールの説明を行っています。

モジュールは，大別してバックグランド処理，1バイト送受信処理および，コマンド処理に分かれています。このうち，実際のシステムに応用する場合には，必ずバックグランド処理と1バイト送受信の処理が必要です（**表10-6　各モジュールの処理概要（1/2）**参照）。

表10-6にこの応用例の各モジュールと処理概要，図10-12にモジュール関連図，および表10-7に，**10.3　基本通信手段**で取り上げたワークの各モジュールでの使用状況を示します。

表10-6　各モジュールの処理概要（1/2）

レーベル		処　理　概　要
マスタ/スレーブ共通	TRNADR	アドレス送信処理
	TRNCMD	コマンド送信処理
	TRNDAT	データ送信処理
	RCVDAT	データ受信処理
	RCVMAC[注]	マクロ・サービス受信起動処理
	ACKCHK	$\overline{\text{ACK}}$受信チェック処理
	MACOUT[注]	マクロ・サービス・タイム・アウト処理
	INTCSI	シリアル転送完了割り込み処理
マスタ	M_ICSI	マスタ側シリアル転送完了割り込み処理
	M_IC21，M_IBT	マスタ側$\overline{\text{ACK}}$チェック処理
	MT_INI	マスタSBI処理エントリ
	M_INIS	マスタ側イニシャライズ処理
スレーブ	S_ICSI	スレーブ側シリアル転送完了割り込み処理
	S_IC21，S_IBT	スレーブ側$\overline{\text{ACK}}$チェック処理
	SL_INI	スレーブSBI処理エントリ
	S_INIS	スレーブ側イニシャライズ処理

注　78K/IIシリーズでマクロ・サービス転送を行う場合にのみ必要な処理です。

表10－6　各モジュールの処理概要（2/2）

レーベル		処　理　概　要
マスタ	M_WR	マスタ側WRITEコマンド処理
	M_RD	マスタ側READコマンド処理
	M_LWR	マスタ側LWRITEコマンド処理
	M_LRD	マスタ側LREADコマンド処理
	M_MACW	マスタ側MACWコマンド処理
	M_MACR	マスタ側MACRコマンド処理
	M_ON	マスタ側DSPONコマンド処理
	M_OFF	マスタ側DSPOFFコマンド処理
	M_CMT	マスタ側CHGMSTコマンド処理
	M_DTC	マスタ側DETACHコマンド処理
スレーブ	S_WR	スレーブ側WRITEコマンド処理
	S_RD	スレーブ側READコマンド処理
	S_LWR	スレーブ側LWRITEコマンド処理
	S_LRD	スレーブ側LREADコマンド処理
	S_MACW	スレーブ側MACWコマンド処理
	S_MACR	スレーブ側MACRコマンド処理
	S_ON	スレーブ側DSPONコマンド処理
	S_OFF	スレーブ側DSPOFFコマンド処理
	S_CMT	スレーブ側CHGMSTコマンド処理
	S_DTC	スレーブ側DETACHコマンド処理

図10－12　モジュール関連図（1/2）

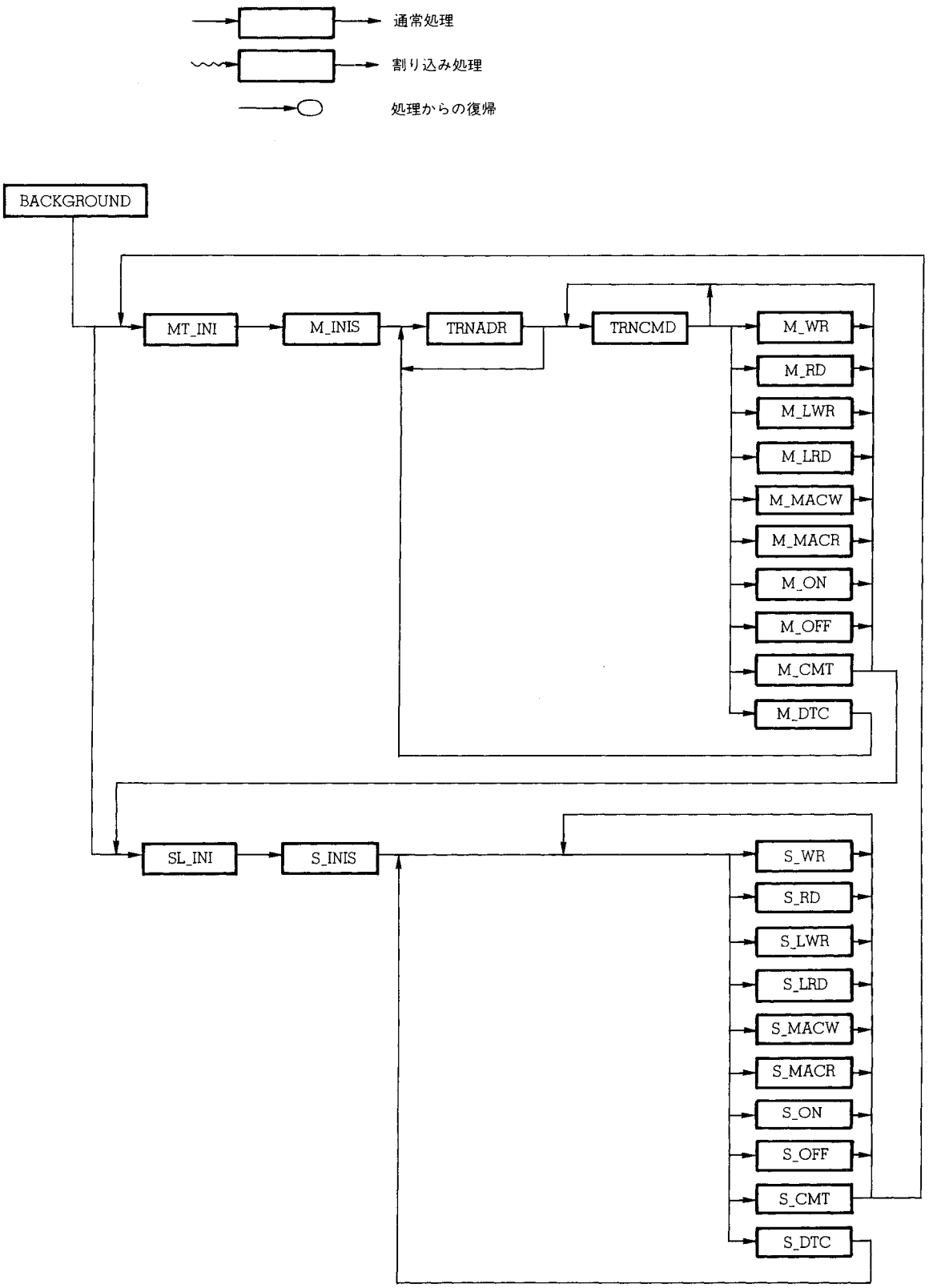


図10－12　モジュール関連図（2/2）

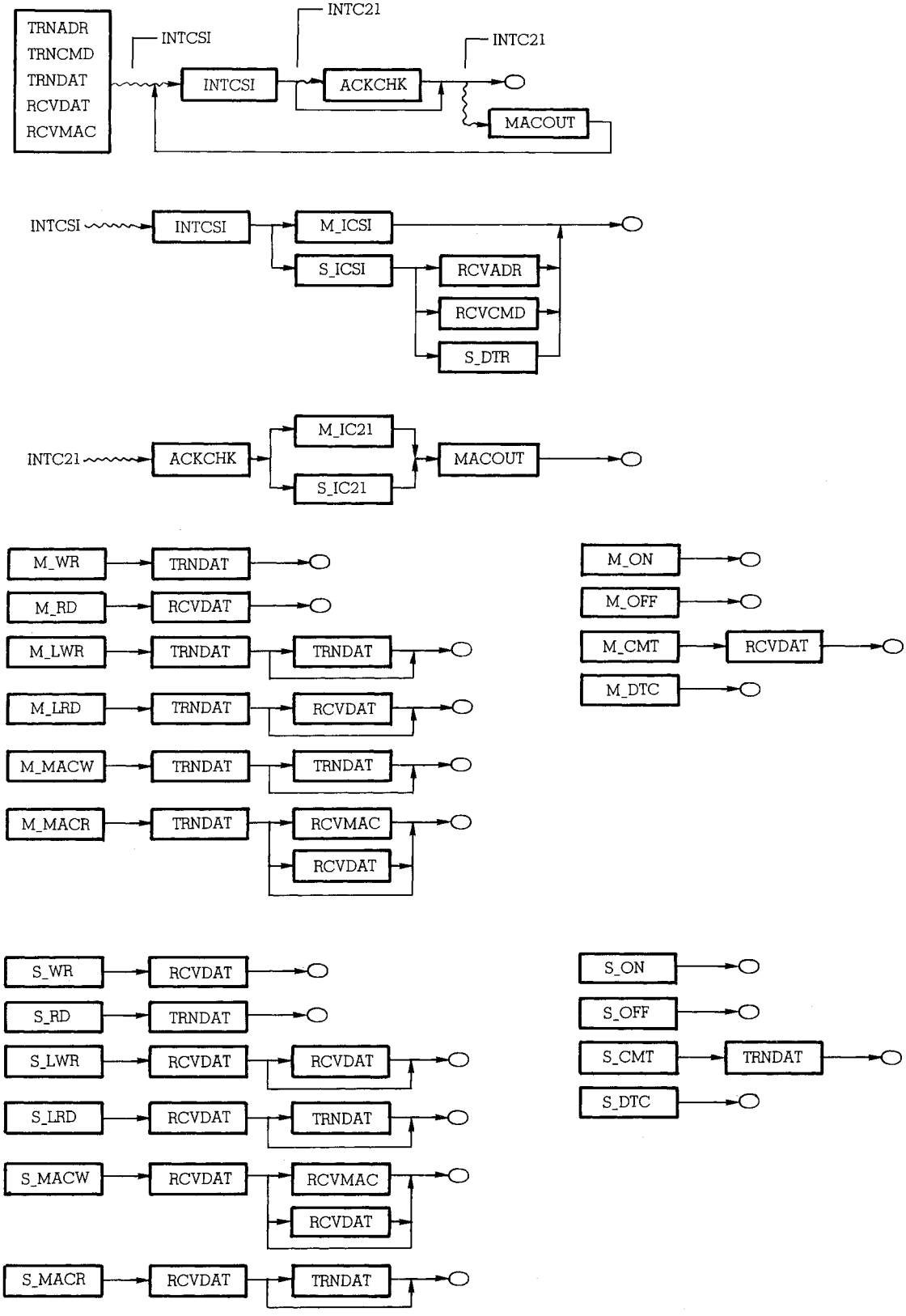


表10－7　ワークの各モジュールでの使用状況（1/2）

レーベル	ライトされる可能性のあるもの		リードされる可能性のあるもの	
	ワークRAM	フラグ	ワークRAM	フラグ
TRNADR TRNCMD TRNDAT	—	RCVFLG TREND ERRORF	TRNBUF	TREND
RCVDAT	TRNBUF	RCVFLG TREND ERRORF	TRNBUF	TREND
RCVMAC	—	RCVFLG TREND ERRORF	TRNBUF	TREND
INTCSI	—	MACTMF	—	MSTF MACF MACTMF
M_ICSI	RCVBUF	ACKWFG TREND	—	RCVFLG
S_ICSI	RCVBUF LPARAM	ERRORF TREND PFLAG ACKWFG CRF SLAVF	RCVBUF LPARAM	RCVFLG CMDBSY PFLAG
M_IC21	ACKCNT	ERRORF ACKWFG TREND	—	ACKWFG
S_IC21	—	ACKWFG TREND	—	ACKWFG
ACKCHK	—	—	—	MASTF
MACOUT	MACCNT	—	MACCNT	MACTMF
MT_INI	—	ATF CTF	TRNBUF	ATF CTF ERRORF
SL_INI	—	CMDBSY CRF TREND	RCVBUF	SLAVF CRF
M_INIS	—	全てのフラグ	—	—
S_INIS	—	全てのフラグ	—	—

注意　サブルーチン呼び出しや，割り込みによるワークRAMやフラグの変化は考慮していませんので，図10－12をあわせて参照してください。

表10－7　ワークの各モジュールでの使用状況（2/2）

レーベル	ライトされる可能性のあるもの		リードされる可能性のあるもの	
	ワークRAM	フラグ	ワークRAM	フラグ
M_WR	TRNBUF	—	WRDATA	—
M_RD	—	—	—	—
M_LWR	TRNBUF	—	LPARAM LSTART	ERRORF
M_LRD	TRNBUF	—	LPARAM LSTART RCVBUF	ERRORF
M_MACW	TRNBUF MSMCSI CHPCSI	MACF	LPARAM LSTART	ERRORF
M_MACR	TRNBUF RCVBUF MSMCSI CHPCSI	MACF	LPARAM LSTART	ERRORF
M_ON	—	—	—	—
M_OFF	—	—	—	—
M_CMT	RCVBUF	—	—	—
M_DTC	—	—	—	—
S_WR	—	—	—	—
S_RD	—	DTF RCVFLG	—	DTF
S_LWR	—	PFLAG	LPARAM LSTART RCVBUF	ERRORF
S_LRD	TRNBUF	PFLAG RCVFLG	LPARAM LSTART	ERRORF
S_MACW	MSMCSI CHPCSI	PFLAG MACF	LPARAM LSTART RCVBUF	ERRORF
S_MACR	TRNBUF MSMCSI CHPCSI	PFLAG RCVFLG MACF	LPARAM LSTART	ERRORF
S_ON	—	STLEDF	—	—
S_OFF	—	STLEDF	—	—
S_CMT	—	DTF RCVFLG	TRNBUF	DTF ERRORF
S_DTC	—	CMDBSY SLVAF	—	—

注意　サブルーチン呼び出しや，割り込みによるワークRAMやフラグの変化は考慮していませんので，図10－12をあわせて参照してください。

10
214
218A
224
234
244

10.4.1　1バイトの送受信処理

アドレス送信，コマンド送信，データの送受信およびマクロ・サービスの受信起動処理を行います。

(1) 動作概要

この処理のエントリ・レーベルは，それぞれの転送指示に応じて表10－8に示すように，5種類あります。

表10－8　1バイト送受信サブルーチン

レーベル	処　理　内　容
TRNADR	マスタからスレーブへのアドレス送信
TRNCMD	マスタからスレーブへのコマンド送信
TRNDAT	マスタおよびスレーブへのデータ送信
RCVDAT	マスタおよびスレーブからのデータ受信
RCVMAC	マスタおよびスレーブのマクロ・サービスによる受信起動

データ送受信のTRNDAT，RCVDATとマクロ・サービスによる受信起動のRCVMACは，マスタ／スレーブともに共通で使用します。

この1バイトの送受信は，送受信の起動と，転送完了のポーリング(TREND＝1)のみを行っていますので，シリアル転送完了割り込み処理と，$\overline{\text{ACK}}$のチェック処理とを組み合わせて使用しています。

(2) プログラム説明 … (6) プログラム・リスト参照

1バイトの送受信処理には，エントリ・レーベルが5つあります。それぞれのレーベルの，プログラム説明開始番号を表10－9に示します。

表10－9　レーベルとプログラム説明開始番号

レーベル	TRNADR	TRNCMD	TRNDAT	RCVDAT	RCVMAC
開始番号	(i)	(v)	(viii)	(ix)	(xiv)

[TRNADR]

(i) READY状態（SBO='H'）をチェックします。

(ii) シリアル・クロックの停止（SCK='H'）をチェックします。

(iii) コマンド・トリガを発行します（CMDT←1）。

(iv) バス・リリースを発行します（RELT←1）。

[TRNCMD]

(v) READY状態（SBO='H'）をチェックします。

(vi) シリアル・クロックの停止（SCK='H'）をチェックします。

(vii) コマンド・トリガを発行します（CMDT←1）。

[TRNDAT]

(viii) 送信モードに設定し（RCVFLG←0），(xi)へ。

[RCVDAT]

(ix) 受信モードに設定します（RCVFLG←1）。

(x) 転送データとしてFFHを設定します。

(xi) 転送完了フラグをクリアします（TREND←0）。

(xii) エラー・フラグをクリアします（ERRORF←0）。

(xiii) シフト・レジスタ（SIO）に送信バッファ（TRNBUF）の値を書き込み，(xviii) へ。

[RCVMAC]

(xiv) 受信モードに設定します（RCVFLG←1）。

(xv) 転送完了フラグをクリアします（TREND←0）。

(xvi) エラー・フラグをクリアします（ERRORF←0）。

(xvii) SIOリードで受信を起動します。

(xviii) 転送完了（TREND=1）までウエイトし，復帰します。

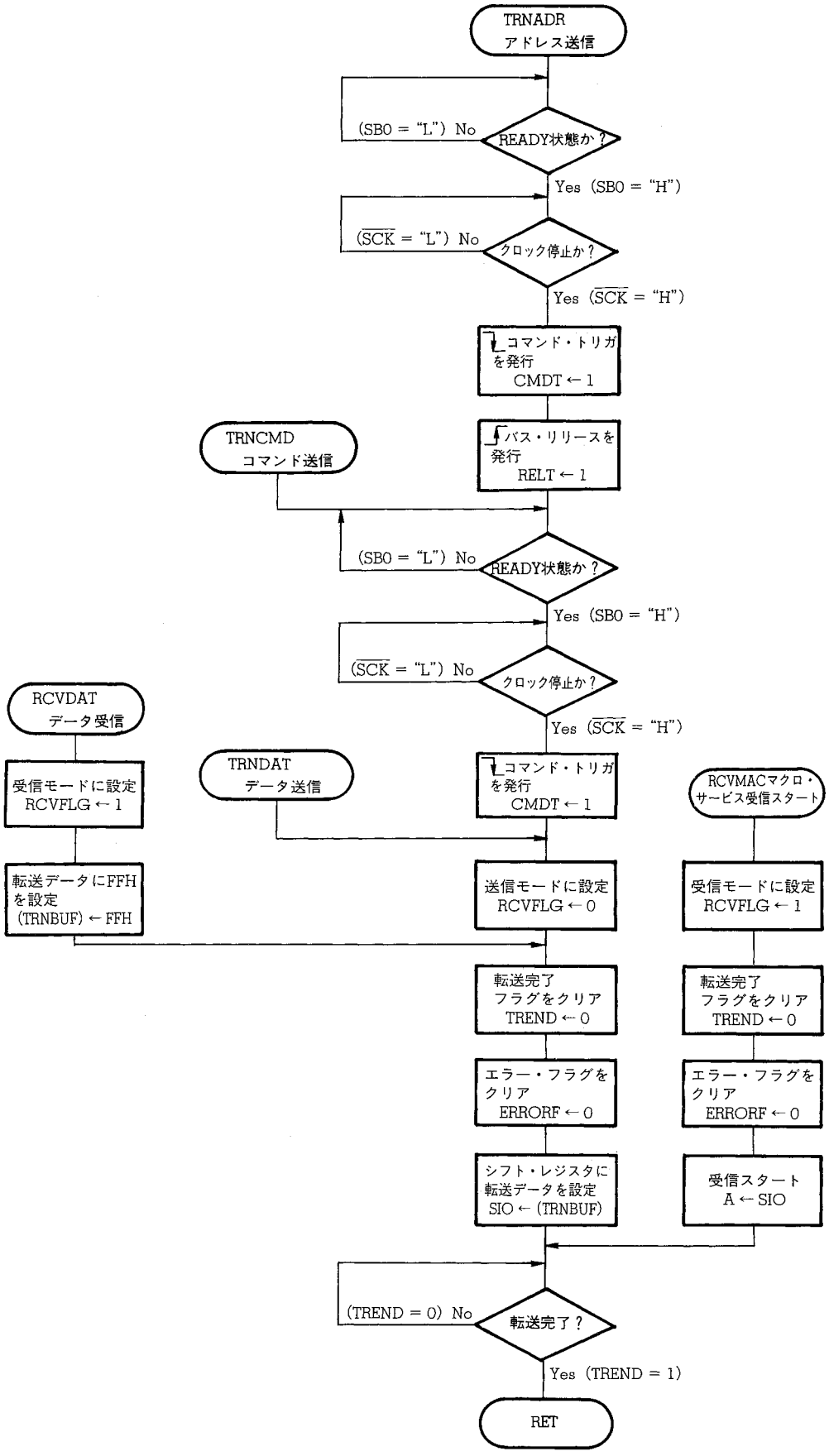
(3) 使用レジスタ

A

(4) 入出力条件

入　力　条　件	出　力　条　件
送信の場合：（TRNBUF）←送信データ 受信の場合：なし	TREND←1 ①　送受信条件 送信の場合：RCVFLG←1 受信の場合：RCVFLG←0 ②　転送結果 正常転送　：ERRORF←0 転送エラー：ERRORF←1 ③　受信バッファの内容 マスタで送信：（RCVBUF）←受信データ スレーブ転送：（RCVBUF）←転送データ

(5) フロー・チャート


TRNADR
アドレス送信
(SBO = "L") No
READY状態か？
Yes (SBO = "H")
(SCK = "L") No
クロック停止か？
Yes (SCK = "H")
コマンド・トリガを発行
CMDT ← 1
TRNCMD
コマンド送信
バス・リリースを発行
RELT ← 1
(SBO = "L") No
READY状態か？
Yes (SBO = "H")
(SCK = "L") No
クロック停止か？
Yes (SCK = "H")
RCVDAT
データ受信
受信モードに設定
RCVFLG ← 1
転送データにFFHを設定
(TRNBUF) ← FFH
TRNDAT
データ送信
コマンド・トリガを発行
CMDT ← 1
送信モードに設定
RCVFLG ← 0
RCVMACマクロ・サービス受信スタート
受信モードに設定
RCVFLG ← 1
転送完了フラグをクリア
TREND ← 0
転送完了フラグをクリア
TREND ← 0
エラー・フラグをクリア
ERRORF ← 0
エラー・フラグをクリア
ERRORF ← 0
シフト・レジスタに転送データを設定
SIO ← (TRNBUF)
受信スタート
A ← SIO
(TREND = 0) No
転送完了？
Yes (TREND = 1)
RET

(6) プログラム・リスト

```
;*************************************************************
;       1 byte Transmit & Receive
;*************************************************************

TRNADR:                                     ; Address
        BF      SB0,$TRNADR
TAD_L1:
        BF      SCK,$TAD_L1
        SET1    CMDT
        SET1    RELT
TRNCMD:                                     ; Command
        BF      SB0,$TRNCMD
TCM_L1:
        BF      SCK,$TCM_L1
        SET1    CMDT
TRNDAT:                                     ; Transmit Data
        CLR1    RCVFLG                      ; Transmit Mode
        BR      TDT_J1
RCVDAT:                                     ; Receive Data
        SET1    RCVFLG                      ; Receive Mode
        MOV     TRNBUF,#0FFH
TDT_J1:
        CLR1    TREND
        CLR1    ERRORF
        MOV     A,TRNBUF
        MOV     SIO,A
        BR      TDT_J2
RCVMAC:                                     ; Macro Service Receive Start
        SET1    RCVFLG
        CLR1    TREND
        CLR1    ERRORF
        MOV     A,SIO
TDT_J2:
        BF      TREND,$TDT_J2
        RET
```

10.4.2　INTCSI割り込み処理

1バイトのシリアル転送完了時の処理として，INTCSI割り込み処理を用いています。

(1) 動作概要

INTCSI割り込み処理は，マスタ動作時の処理と，スレーブ動作時の処理の2つに分かれています。処理の判断はマスタ・フラグ（MSTF）で行います。

表10－10　マスタ・フラグとINTCSI処理の関係

動作	MSTF	RCVFLG	処　理　概　要
マスタ INTCSI	1	0	送信側：$\overline{ACK}$の受信をチェックし，$\overline{ACK}$がない場合には，$\overline{ACK}$待ち状態に設定します。
		1	受信側：$\overline{ACK}$を出力し，受信データを取り込みます。
スレーブ INTCSI	0	0	送信側：$\overline{ACK}$の受信をチェックし，$\overline{ACK}$がない場合には，$\overline{ACK}$待ち状態に設定します。
		1	受信側：アドレス，コマンド，データ処理からなり，コマンド，およびパラメータが有効であるかを判断しています。

(2) プログラム説明 … (6) プログラム・リスト参照

直前の転送がマクロ・サービスの場合は，マクロ・サービス・タイム・アウト状態（MACTMF←1）に設定し，通常のINTCSI処理を行わずに復帰しています。詳細は**10.4.4　マクロ・サービス・タイム・アウト処理**を参照してください。

(a) マスタ転送完了割り込み処理［レーベル名称：M_ICSI］

転送完了割り込みは，送信の場合と，受信の場合があるため，直前の転送方向をRCVFLGで管理しています。

(i) 送信終了の場合（RCVFLG＝0）

① $\overline{ACK}$受信のチェックを行います。

② $\overline{ACK}$を受信していれば（ACKD＝1），転送完了フラグをセット（TREND←1）して，戻ります。

③ $\overline{ACK}$を受信していない場合(ACKD＝0)は，$\overline{ACK}$待ちフラグをセット(ACKWFG←1)して戻ります。

10
214
218A
224
234
244

(ii) 受信完了の場合（RCVFLG＝1）

① シフト・レジスタ（SIO）の値を受信バッファ（RCVBUF）に書き込みます。

② 受信をスレーブに知らせるために，$\overline{\text{ACK}}$を送信します。

③ 転送完了フラグ（TREND）をセット(1)して戻ります。

(b) スレーブ転送完了割り込み処理［レーベル名称：S_ICSI］

スレーブでは，INTCSI割り込み要求が発生すると，まずエラー・フラグ（ERRORF）をクリア(0)し，これからの通信エラー発生に備えます。シフト・レジスタ(SIO)の値を受信バッファ（RCVBUF）に書き込みます。バス・リリース信号の検出，コマンド信号の検出をチェックすることで，アドレス，コマンド，データの判断を行い，それぞれの処理を行います。

(i) アドレス受信処理［レーベル名称：RCVADR］

① アドレスの判別を行います。不一致アドレスを受信した場合は，送受信禁止とし，シフト・レジスタ（SIO）にFFHを設定し，再び送受信許可とし戻ります。

一致アドレスを受信した場合は，転送方向と，コマンド処理中でないことを確認して③へ。

RCVFLG＝0または，CMDBSY＝1の場合は，ERRORF←1として⑤へ。

② ウエイク・アップ状態を解除（WUP←0）し，選択状態とします。

③ $\overline{\text{ACK}}$を出力（ACKT←1）します。

④ 転送完了フラグをセット（TREND←1）して戻ります。

注意　①の不一致アドレスを受信した場合で，いったん送受信禁止にしたあと再び送受信許可にしているのは，IE-78210-R，IE-78220-Rの不具合を回避するためです。実際のデバイスではこの送受信禁止と送受信許可の処理はなくても問題ありません。

(ii) コマンド受信処理［レーベル名称：RCVCMD］

① 転送方向とコマンド処理状態をチェックします。

② コマンド受信待ちでない場合（RCVFLG＝0または，RCVFLG＝1，CMDBSY＝1）は，エラー・フラグをセット（ERRORF←1）し，④へ。

③ コマンド受信可の場合（RCVFLG＝1％ CMDBSY＝0）は，コマンドの判別を行います。未登録コマンドの場合はエラー・フラグをセット（ERRORF←1）して，⑥へ。注

④ $\overline{ACK}$を出力します（ACKT←1）。

⑤ コマンド受信フラグをセットします（CRF←1）。

⑥ 転送完了フラグをセット（TREND←1）して戻ります。

注　**10.1.2　アドレス，コマンドの定義**で取り上げているコマンドでない場合は，エラーとしています。ただし，78K/IIシリーズではテーブル参照命令でチェックし，FFHをターミネータとしているため，FFHは有効コマンドとして認識されます。したがって，コマンドとしてFFHを送信した場合は暴走しますので，送信しないでください。

(iii) データ転送処理［レーベル名称：S_DTR］

データ転送の場合も，送信の場合と，受信の場合があります。

① 送信終了（RCVFLG＝0）の場合は，$\overline{ACK}$受信のチェックを行います。

② $\overline{ACK}$を受信していれば（ACKD＝1），⑦へ。

③ $\overline{ACK}$を受信していない場合は，$\overline{ACK}$待ちフラグをセット（ACKWFG←1）して戻ります。

④ 受信終了の場合（RCVFLG＝1）は，データがパラメータかどうかを判断します。パラメータでない場合（PFLAG＝0）は，⑥へ。

⑤ パラメータが適当な場合（(RCVFLG)＜(LPARAM)）には，連続データ転送コマンドの転送バイト数として，(LPARAM) に受信バッファ（RCVBUF）の値を設定します。パラメータが不適当である場合は，エラー・フラグをセット（ERRORF＝1）して，⑦へ。

⑥ マスタに受信を知らせるために$\overline{ACK}$を出力します。

⑦ パラメータ待ちフラグをクリア（PFLAG←0）します。

⑧ 転送完了フラグをセット（TREND←1）し，戻ります。

10
214
218A
224
234
244

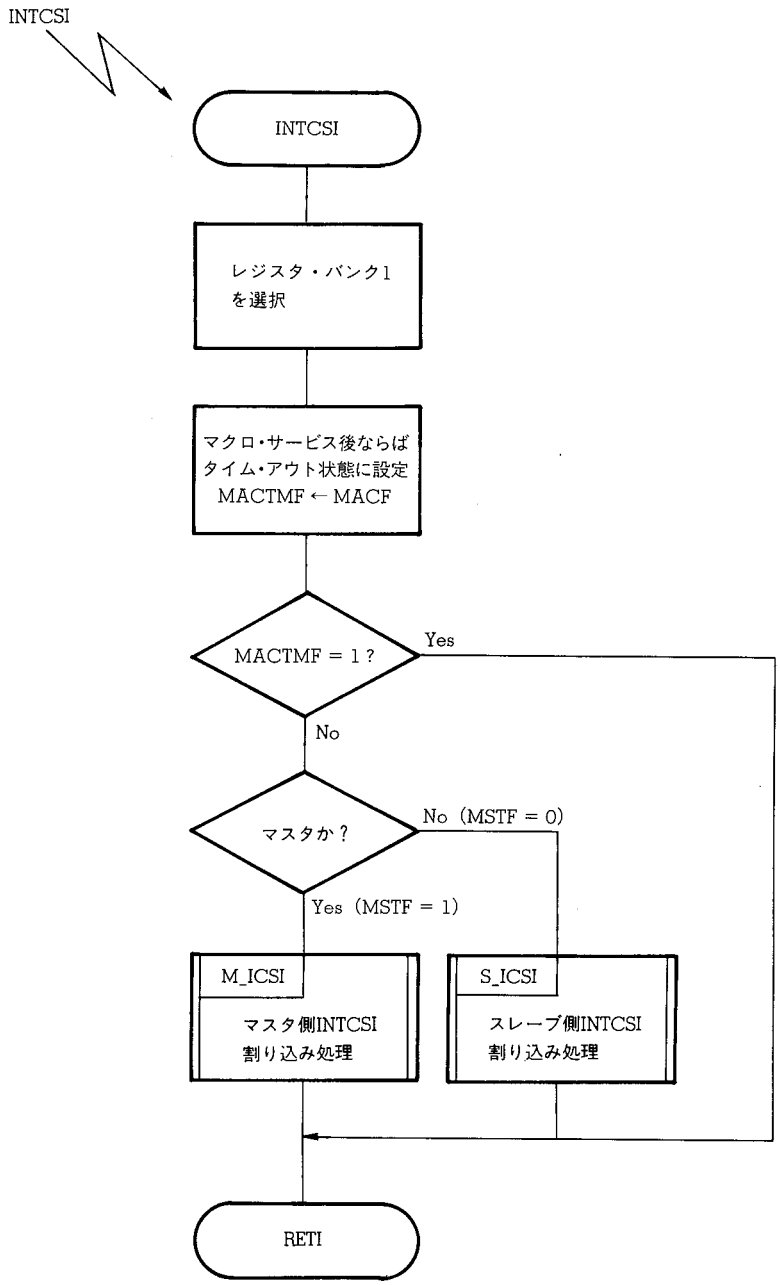
(3) 使用レジスタ

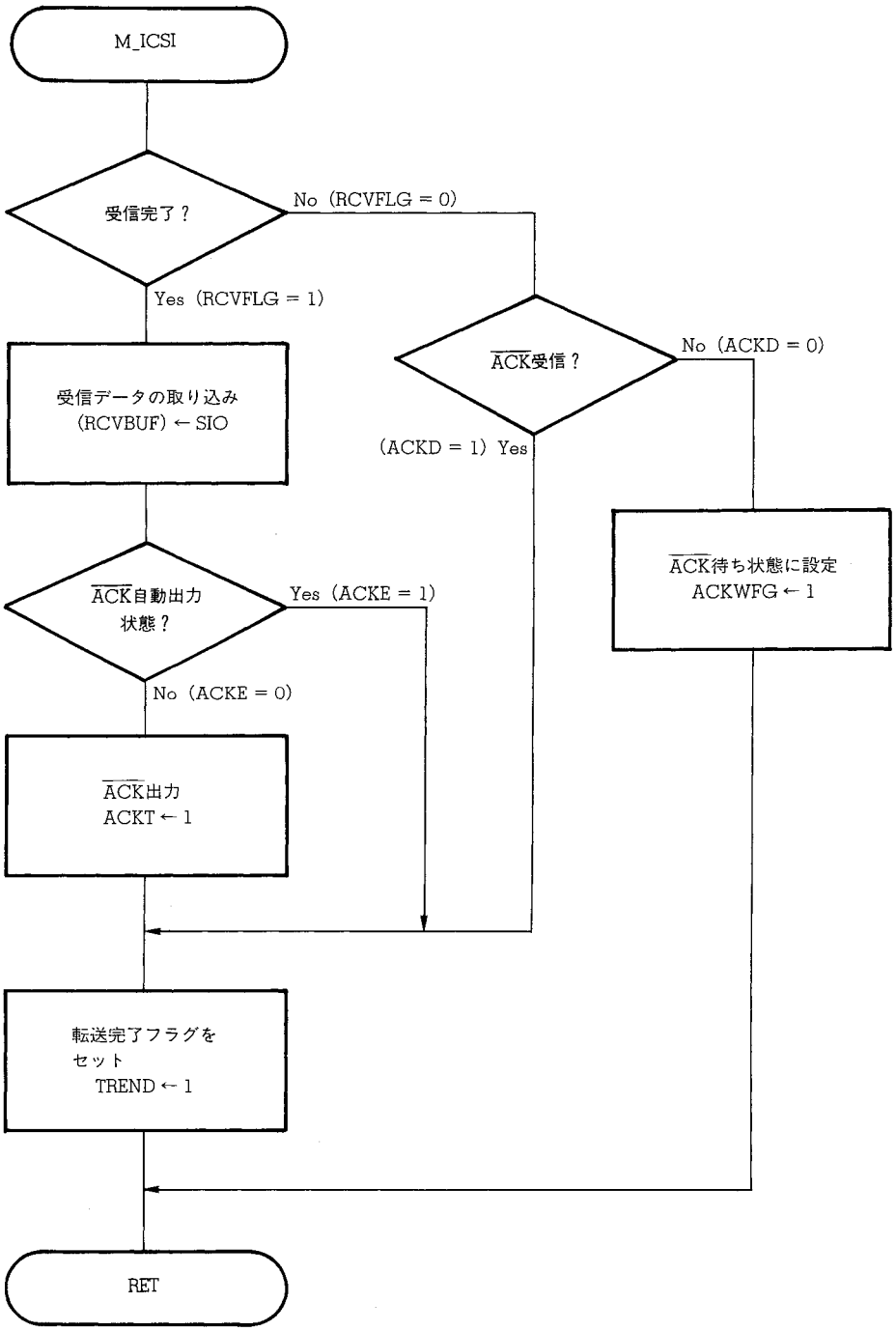
A, X, HL（すべてレジスタ・バンク1）

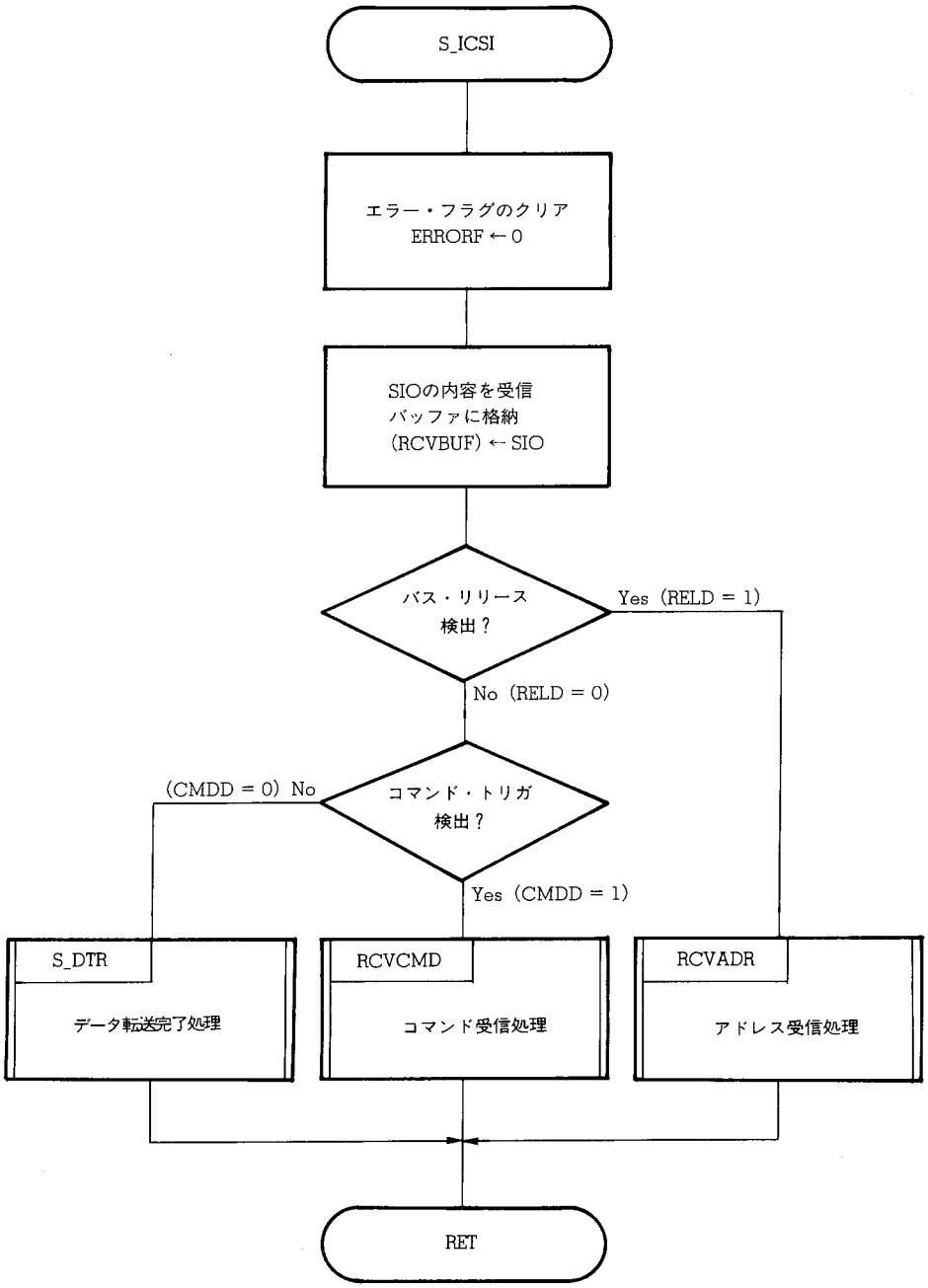
(4) 入出力条件

デバイス	入力条件	出力条件
マスタ	MSTF＝1 INTCSI割り込み許可	送信時：ACKD＝0のときACKWFG←1 受信時：(RCVBUF) ←受信データ
スレーブ	MSTF＝0 INTCSI割り込み許可 マクロ・サービス転送完了の場合は MACF←1	(RCVBUF) ←送受信データ ① 転送結果 正常転送　：ERRORF←0 転送エラー：ERRORF←1 ② マクロ・サービス転送完了の場合 MACTMF←1 ③ $\overline{\text{ACK}}$受信状況 $\overline{\text{ACK}}$なし　：ACKWFG←1 $\overline{\text{ACK}}$受信時，$\overline{\text{ACK}}$自動出力時 ：TREND←1 ④ アドレス受信時 一致アドレス　：WUP←0 不一致アドレス：なし ⑤ 有効コマンド受信時：CRF←1 ⑥ 有効パラメータ受信時 (PFLAG＝1, (LPARAM)＞(RCVBUF)) ：(LPARAM) ←受信データ

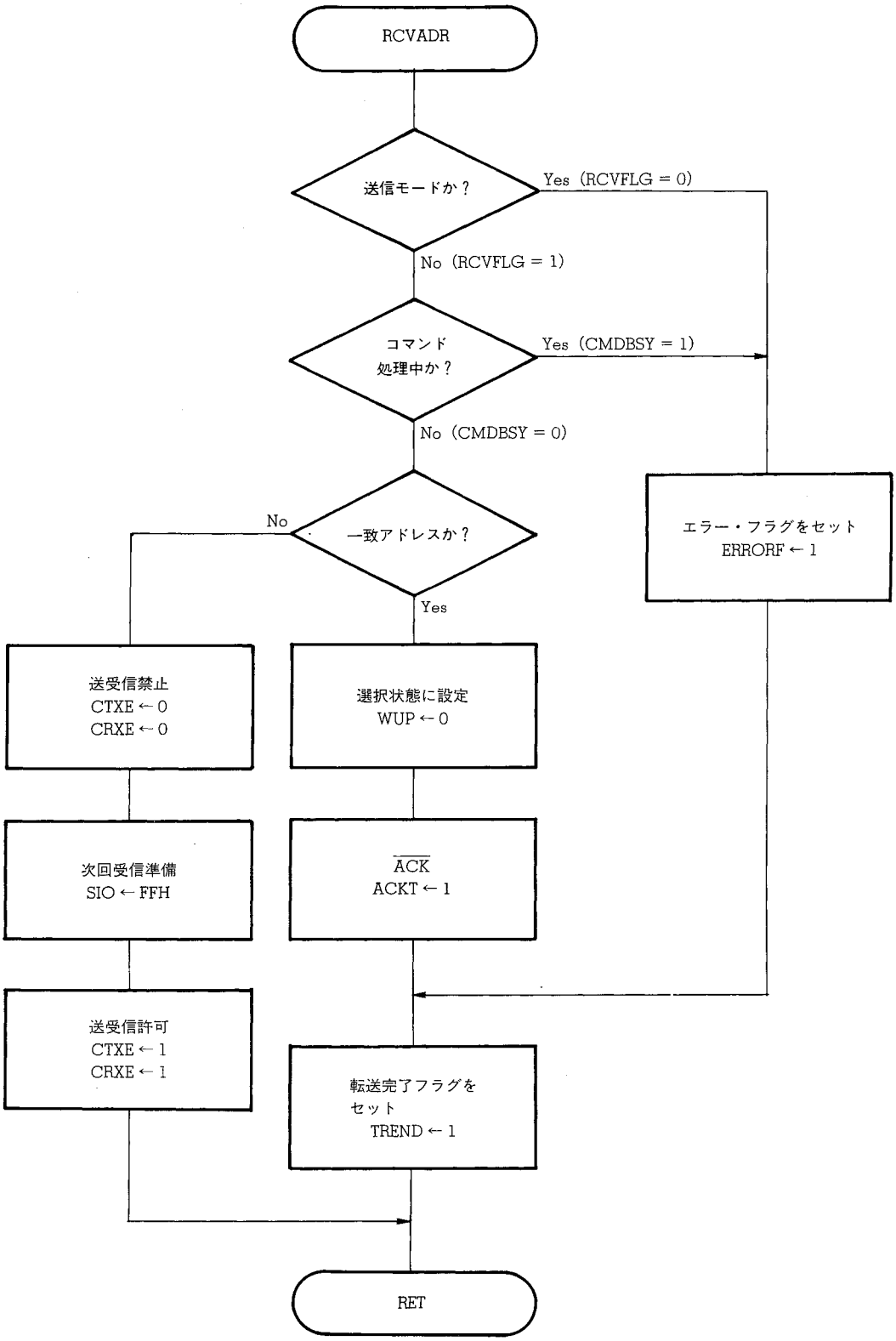
(5) フロー・チャート

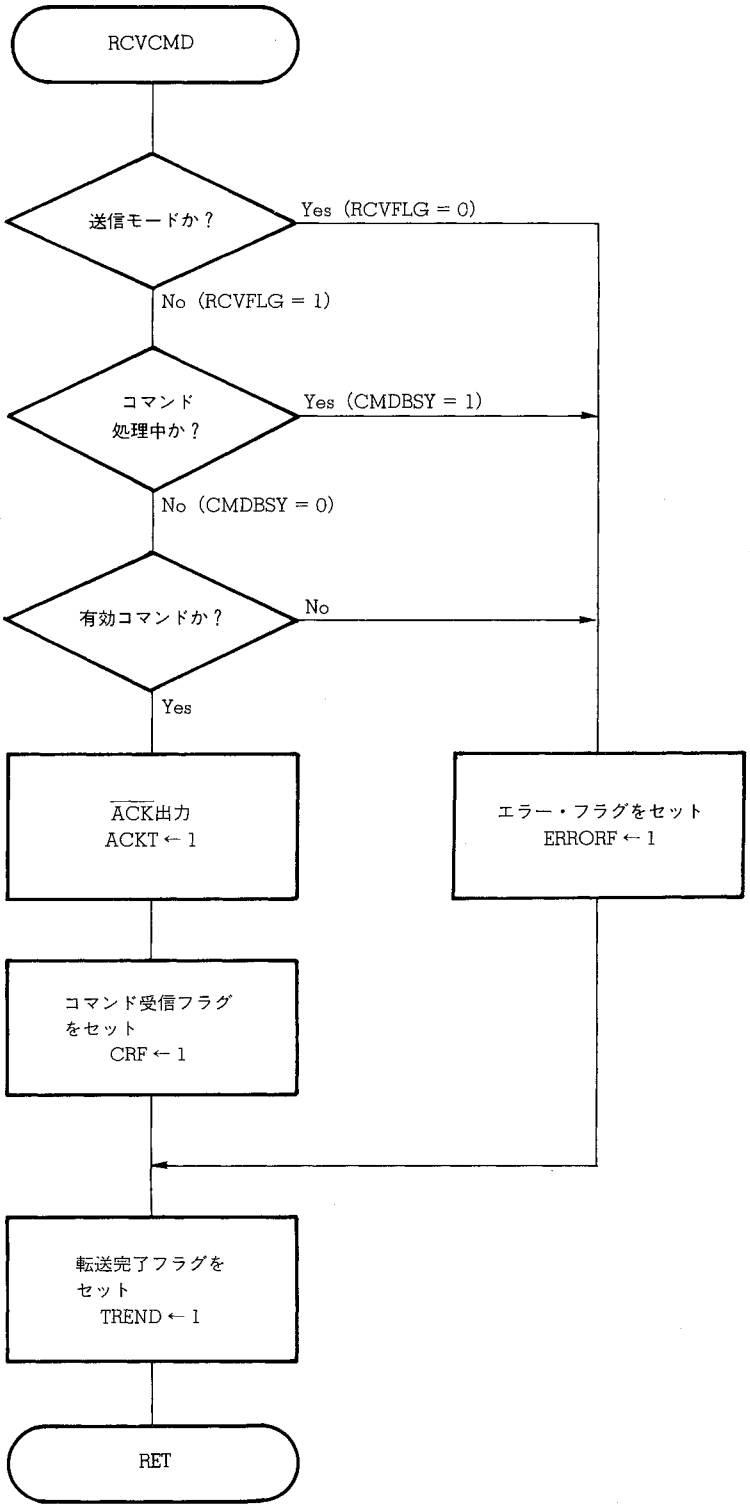

INTCSI
INTCSI
レジスタ・バンク1
を選択
マクロ・サービス後ならば
タイム・アウト状態に設定
MACTMF ← MACF
MACTMF = 1 ?
Yes
No
マスタか ?
No (MSTF = 0)
Yes (MSTF = 1)
M_ICSI
マスタ側INTCSI
割り込み処理
S_ICSI
スレーブ側INTCSI
割り込み処理
RETI


M_ICSI
受信完了？
No（RCVFLG = 0）
Yes（RCVFLG = 1）
受信データの取り込み
（RCVBUF）← SIO
ACK受信？
No（ACKD = 0）
（ACKD = 1）Yes
ACK待ち状態に設定
ACKWFG ← 1
ACK自動出力
状態？
Yes（ACKE = 1）
No（ACKE = 0）
ACK出力
ACKT ← 1
転送完了フラグを
セット
TREND ← 1
RET

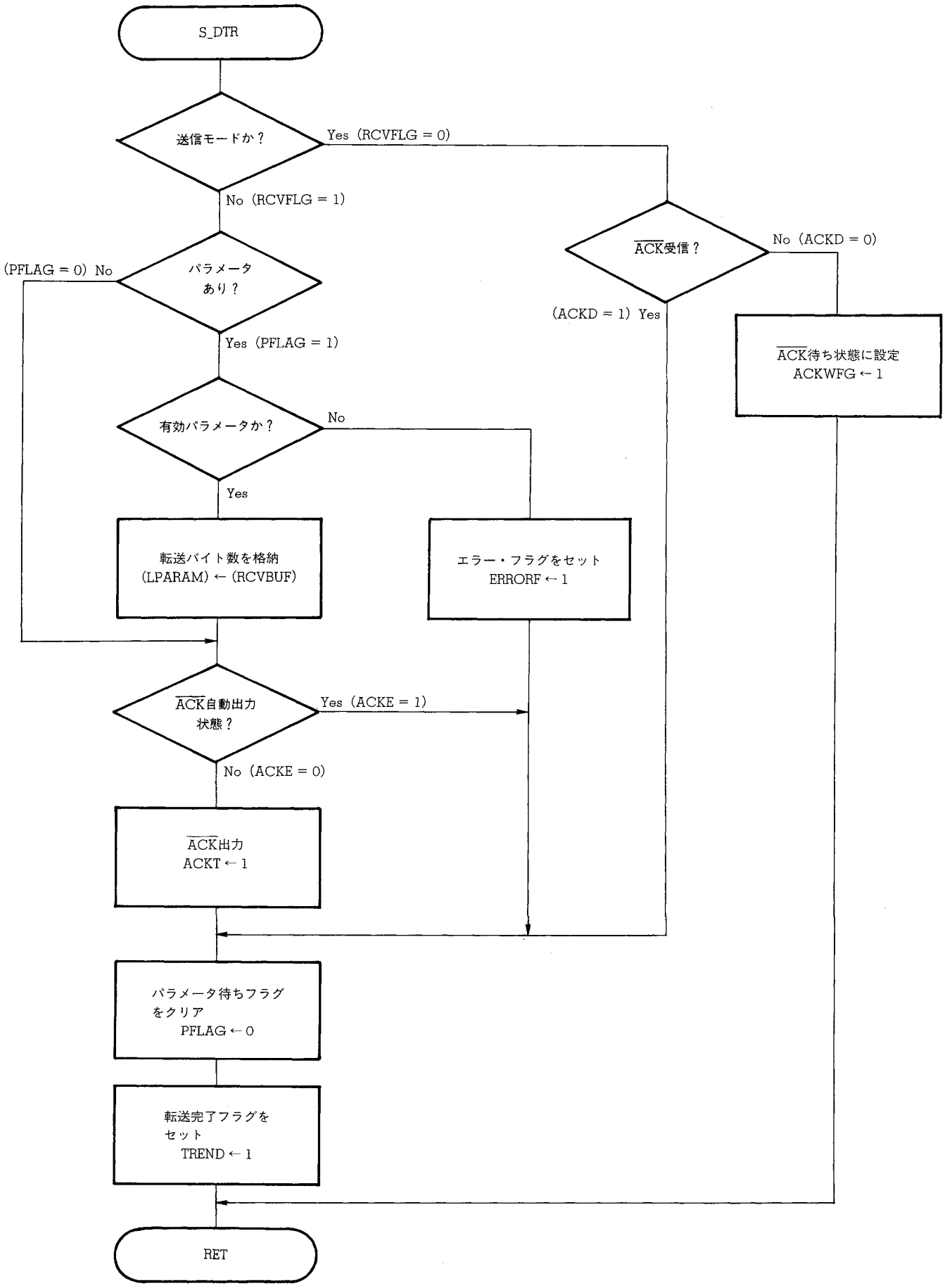


10
214
218A
224
234
244


RCVADR
送信モードか？
Yes（RCVFLG = 0）
No（RCVFLG = 1）
コマンド
処理中か？
Yes（CMDBSY = 1）
No（CMDBSY = 0）
No
一致アドレスか？
Yes
エラー・フラグをセット
ERRORF ← 1
送受信禁止
CTXE ← 0
CRXE ← 0
選択状態に設定
WUP ← 0
次回受信準備
SIO ← FFH
ACK
ACKT ← 1
送受信許可
CTXE ← 1
CRXE ← 1
転送完了フラグを
セット
TREND ← 1
RET



10
214
218A
224
234
244


S_DTR
送信モードか？
Yes（RCVFLG = 0）
No（RCVFLG = 1）
ACK受信？
No（ACKD = 0）
（PFLAG = 0）No
パラメータ
あり？
（ACKD = 1）Yes
ACK待ち状態に設定
ACKWFG ← 1
Yes（PFLAG = 1）
有効パラメータか？
No
Yes
転送バイト数を格納
（LPARAM）←（RCVBUF）
エラー・フラグをセット
ERRORF ← 1
ACK自動出力
状態？
Yes（ACKE = 1）
No（ACKE = 0）
ACK出力
ACKT ← 1
パラメータ待ちフラグ
をクリア
PFLAG ← 0
転送完了フラグを
セット
TREND ← 1
RET

(6) プログラム・リスト

```
;*************************************************************
;       INTCSI INTERRUPT
;*************************************************************

INTCSI:
        SEL     RB1
        MOV1    CY,MACF
        MOV1    MACTMF,CY
        BT      MACTMF,$ICS_J2          ; Complete of Macro Service
        BF      MSTF,$ICS_J1
        CALL    !M_ICSI                 ; Master
        BR      ICS_J2
ICS_J1:
        CALL    !S_ICSI                 ; Slave
ICS_J2:
        RETI
;
;       --- For Master ---
;
M_ICSI:
        BF      RCVFLG,$MIC_J2
        MOV     A,SIO                   ; Receive
        MOV     RCVBUF,A
        BT      ACKE,$MIC_J1
        SET1    ACKT
MIC_J1:
        SET1    TREND
        BR      MIC_J3
MIC_J2:                                 ; Transmit
        BT      ACKD,$MIC_J1
        SET1    ACKWFG
MIC_J3:
        RET
;
;       --- For Slave ---
;
S_ICSI:
        CLR1    ERRORF
        MOV     A,SIO
        MOV     RCVBUF,A
        BT      RELD,$SIC_J1
        BT      CMDD,$SIC_J2
        CALL    !S_DTR                  ; Data
        BR      SIC_J3
SIC_J1:
        CALL    !RCVADR                 ; Address
        BR      SIC_J3
SIC_J2:
        CALL    !RCVCMD                 ; Command
SIC_J3:
        RET
```

10
214
218A
224
234
244

```
;
;       ... ADDRESS ...
;
RCVADR:                                     ; Address
        BF      RCVFLG, $RAD_J1
        BT      CMDBSY, $RAD_J1
        CMP     SVA, RCVBUF
        BNE     $RAD_J3
        CLR1    WUP                         ; Same Address
        SET1    ACKT
        BR      RAD_J2
RAD_J1:
        SET1    ERRORF
RAD_J2:
        SET1    TREND                       ; end of transmit or receive
        BR      RAD_J4
RAD_J3:
        AND     CSIM, #00111111B            ; For Next Address Receive
        MOV     SIO, #0FFH
        OR      CSIM, #11000000B
RAD_J4:
        RET
;
;       ... COMMAND ...
;
RCVCMD:
        BF      RCVFLG, $RCD_J2
        BT      CMDBSY, $RCD_J2
        MOVW    HL, #CMDTBL                 ; Command check
RCD_L1:
        MOV     A, [HL+]
        CMP     A, RCVBUF
        BZ      $RCD_J1
        CMP     A, #0FFH
        BE      $RCD_J2                     ; Error Command Code
        BR      RCD_L1
RCD_J1:
        SET1    ACKT                        ; Active Command
        SET1    CRF
        BR      RCD_J3
RCD_J2:
        SET1    ERRORF
RCD_J3:
        SET1    TREND
        RET
```

```
CMDTBL:
        DB      20H                     ; WRITE
        DB      21H                     ; READ
        DB      22H                     ; LWRITE
        DB      23H                     ; LREAD
        DB      28H                     ; CHGMST
        DB      29H                     ; DETACH

        DB      24H                     ; MACW
        DB      25H                     ; MACR

        DB      31H                     ; DSPON
        DB      32H                     ; DSPOFF

        DB      0FFH                    ; terminater
      ;
      ;       ... DATA ...
      ;
      S_DTR:
              BF      RCVFLG,$SDR_J3
              BF      PFLAG,$SDR_J1
              MOV     A,RCVBUF                ; Check Error Parmeter
              MOV     X,A
              MOV     A,LPARAM
              CMP     A,X
              BC      $SDR_J2
              MOV     LPARAM,RCVBUF
      SDR_J1:
              BT      ACKE,$SDR_J4
              SET1    ACKT
              BR      SDR_J4
      SDR_J2:
              SET1    ERRORF                  ; Parameter error
              BR      SDR_J4
      SDR_J3:
              BF      ACKD,$SDR_J5
      SDR_J4:
              CLR1    PFLAG
              SET1    TREND                   ; end of transmit or receive
              BR      SDR_J6
      SDR_J5:
              SET1    ACKWFG                  ; set ACK wait condition
      SDR_J6:
              RET
```

10.4.3　$\overline{\text{ACK}}$チェック処理

$\overline{\text{ACK}}$の受信の判断，$\overline{\text{ACK}}$受信待ち時間のカウントの処理を行います。

(1) 動作概要

$\overline{\text{ACK}}$チェック処理は，マスタ動作時の処理と，スレーブ動作時の処理の２つに分かれています。処理の判断はマスタ・フラグ（MSTF）で行います。また，$\overline{\text{ACK}}$待ち状態の設定はINTCSI割り込み処理で行っています（ACKWFGフラグの操作）。

表10－11　マスタ・フラグと$\overline{\text{ACK}}$チェック処理の関係

$\overline{\text{ACK}}$チェック動作	MSTF	$\overline{\text{ACK}}$待ちタイム・アウト処理の実行
マスタ側	1	行う
スレーブ側	0	行わない

(a) マスタ側

マスタは$\overline{\text{ACK}}$の受信を，インターバル・タイマ（INTC2）により監視しています。一定期間に$\overline{\text{ACK}}$を検出できなければ，タイム・アウト状態として，エラー・フラグをセット（ERRORF←1）し，マスタから$\overline{\text{ACK}}$を出力しています。

この応用プログラムでは，$\overline{\text{ACK}}$チェックの処理を1.92 msごとに呼び出しています。このときのタイム・アウト時間は，最小28.80 ms，最大30.72 msとなります。

(b) スレーブ側

スレーブでは，$\overline{\text{ACK}}$のタイム・アウト処理は行いません。スレーブがデータを送信した場合は，マスタからの$\overline{\text{ACK}}$を受信するまで，$\overline{\text{ACK}}$待ち状態が継続されます。

(2) プログラム説明

(a) マスタ$\overline{\text{ACK}}$チェック処理［レーベル名称：M_IC21］… (6) プログラム・リスト参照

(i) $\overline{\text{ACK}}$待ち状態かをチェックします。$\overline{\text{ACK}}$待ち状態でない場合(ACKWFG=0)は，戻ります。

(ii) $\overline{\text{ACK}}$の受信をチェックします。$\overline{\text{ACK}}$を受信していれば（ACKD=1），(vii)へ。

(iii) $\overline{\text{ACK}}$を受信していない場合（ACKD=0）は，$\overline{\text{ACK}}$待ちカウンタをインクリメントします（(ACKCNT) ← (ACKCNT)+1)。

(iv) カウント回数が16未満であれば，戻ります。

(v) カウント回数が16の場合は，タイム・アウトとして，エラー・フラグをセット(ERRORF←1)します。

(vi) $\overline{\text{ACK}}$を出力します（ACKT←1)。

(vii) $\overline{\text{ACK}}$待ちカウンタをクリア（(ACKCNT) ←0）します。

(viii) $\overline{\text{ACK}}$待ちフラグをクリア（ACKWFG←0）します。

(ix) 転送完了フラグをセット（TREND←1）して戻ります。

(b) スレーブ$\overline{\text{ACK}}$チェック処理［レーベル名称：S_IC21］

(i) $\overline{\text{ACK}}$待ち状態かをチェックします。$\overline{\text{ACK}}$待ち状態でない場合(ACKWFG=0)は，戻ります。

(ii) $\overline{\text{ACK}}$の受信をチェックします。$\overline{\text{ACK}}$を受信していなければ（ACKD=0）戻ります。

(iii) $\overline{\text{ACK}}$待ちフラグをクリア（ACKWFG←0）します。

(iv) 転送完了フラグをセット（TREND←1）して戻ります。

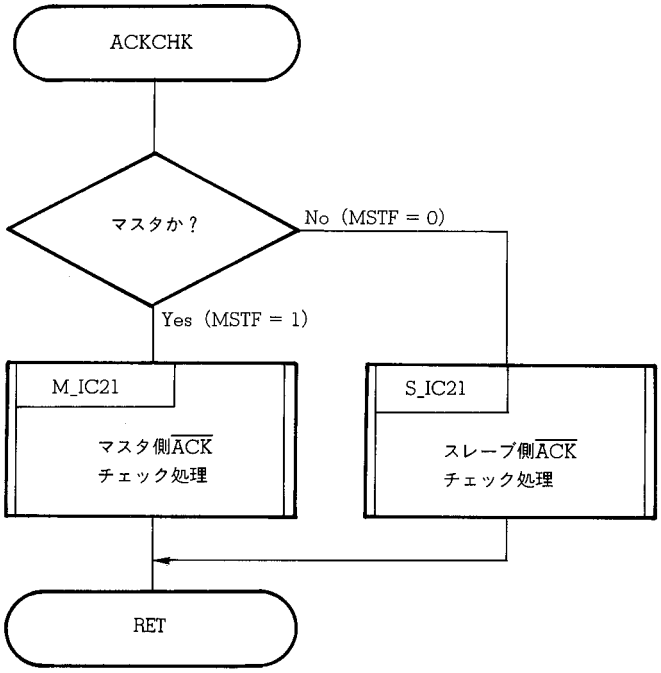
(3) 使用レジスタ

なし

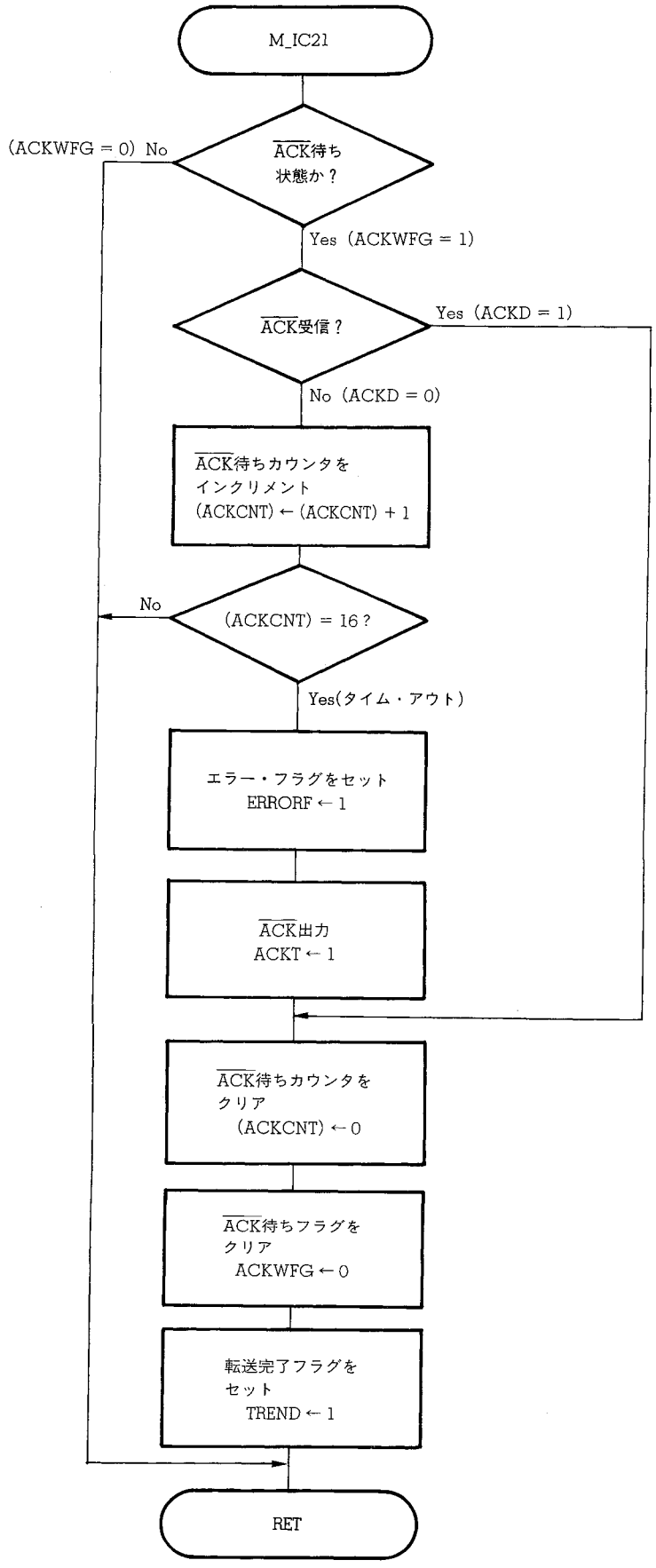
(4) 入出力条件

デバイス	入　力　条　件	出　力　条　件
マスタ	MSTF＝1	(1) $\overline{\text{ACK}}$待ち状態の場合（ACKWFG＝1） ① タイム・アウト時 ERRORF←1 ACKT←1 (ACKCNT) ←0 ACKWFG←0 TREND←1 ② タイム・アウト待ち (ACKCNT) ← (ACKCNT)＋1 ③ $\overline{\text{ACK}}$検出 (ACKCNT) ←0 ACKWFG←0 TREND←1 (2) $\overline{\text{ACK}}$待ち状態でない場合（ACKWFG＝0） なし
スレーブ	MSTF＝0	(1) $\overline{\text{ACK}}$待ち状態の場合（ACKWFG＝1） ① タイム・アウト待ち なし ② $\overline{\text{ACK}}$検出 ACKWFG←0 TREND←1 (2) $\overline{\text{ACK}}$待ち状態でない場合（ACKWFG＝0） なし

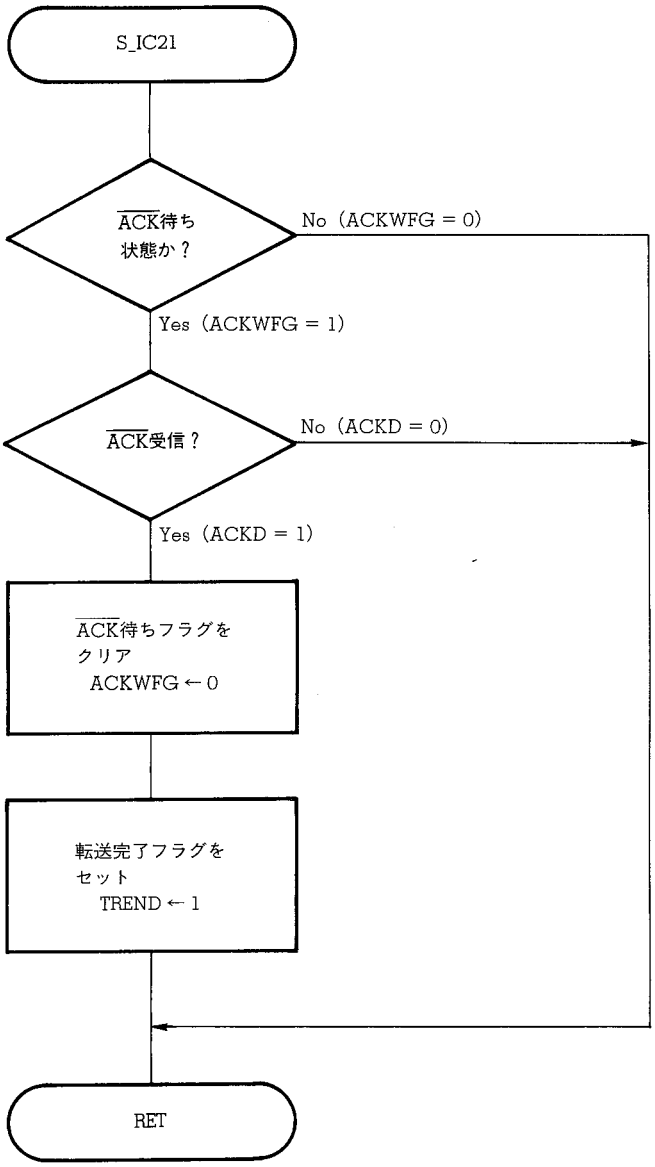
(5) フロー・チャート


ACKCHK
マスタか？
No（MSTF = 0）
Yes（MSTF = 1）
M_IC21
マスタ側$\overline{ACK}$
チェック処理
S_IC21
スレーブ側$\overline{ACK}$
チェック処理
RET

(a) マスタ


M_IC21
(ACKWFG = 0) No
ACK待ち
状態か？
Yes (ACKWFG = 1)
ACK受信？
Yes (ACKD = 1)
No (ACKD = 0)
ACK待ちカウンタを
インクリメント
(ACKCNT) ← (ACKCNT) + 1
No
(ACKCNT) = 16？
Yes(タイム・アウト)
エラー・フラグをセット
ERRORF ← 1
ACK出力
ACKT ← 1
ACK待ちカウンタを
クリア
(ACKCNT) ← 0
ACK待ちフラグを
クリア
ACKWFG ← 0
転送完了フラグを
セット
TREND ← 1
RET

(b) スレーブ


S_IC21
ACK待ち
状態か？
No (ACKWFG = 0)
Yes (ACKWFG = 1)
ACK受信？
No (ACKD = 0)
Yes (ACKD = 1)
ACK待ちフラグを
クリア
ACKWFG ← 0
転送完了フラグを
セット
TREND ← 1
RET

(6) プログラム・リスト

```
;*************************************************************
;       ACK CHECK
;*************************************************************

ACKCHK:
        BF      MSTF,$ACK_J1            ; check master or slave
        CALL    !M_IC21                 ; master
        BR      ACK_J2
ACK_J1:
        CALL    !S_IC21                 ; slave
ACK_J2:
        RET
;
;       --- For Master ---
;
M_IC21:
        BF      ACKWFG,$M21_J2          ; check ACK wait condition
        BT      ACKD,$M21_J1
        INC     ACKCNT
        BF      PSW.4,$M21_J2
        SET1    ERRORF                  ; Time Out
        SET1    ACKT
M21_J1:
        MOV     ACKCNT,#0
        CLR1    ACKWFG
        SET1    TREND                   ; end of tranmit or receive
M21_J2:
        RET
;
;       --- For Slave ---
;
S_IC21:
        BF      ACKWFG,$S21_J1          ; check ACK wait condition
        BF      ACKD,$S21_J1
        CLR1    ACKWFG
        SET1    TREND                   ; end of transmit or receive
S21_J1:
        RET
```

10.4.4　マクロ・サービス・タイム・アウト処理

マクロ・サービスによる転送後の処理です。

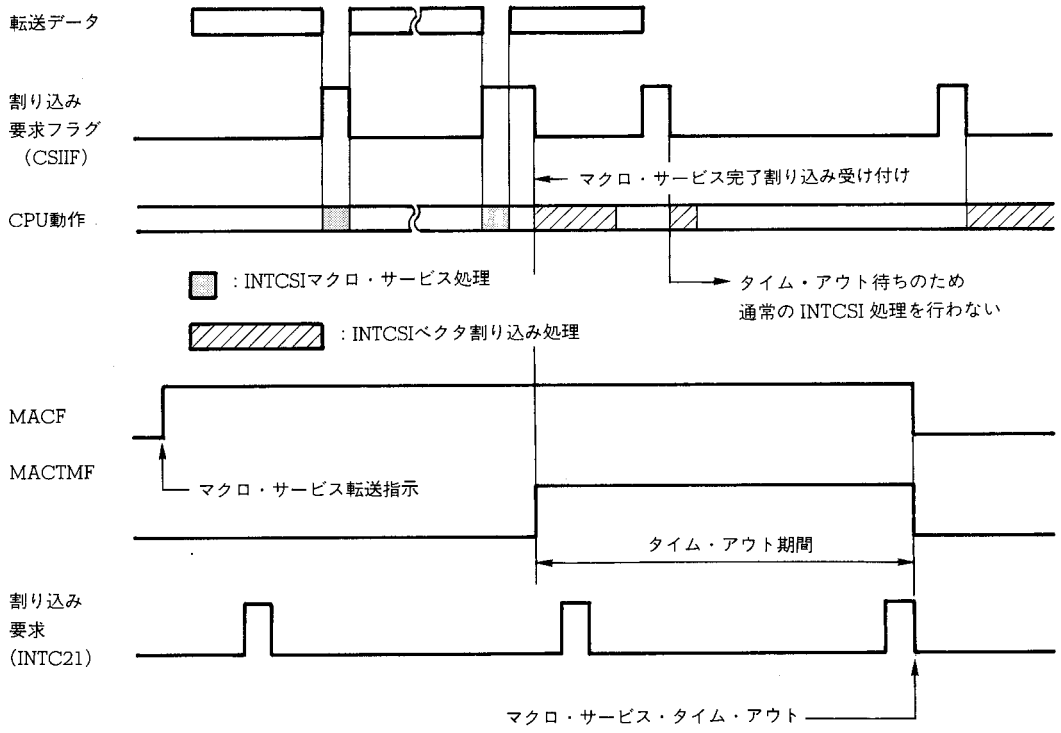
(1) 動作概要

マクロ・サービスを使用する場合，最後のSIOとのマクロ・サービス転送を行ったあとに発生する，マクロ・サービス完了割り込みと，シリアル転送完了割り込みが，転送レートや他の割り込み処理などとの関係で識別できなくなる可能性があります（同一の割り込み要求フラグを使用しているため）。

ここで，確実に2回の割り込み要求を発生させるために，1回目のINTCSI発生時にタイム・アウト状態に設定しています。ここで，$\overline{\text{ACK}}$チェック処理と同一のタイマ割り込み要求（INTC21）を2回カウントしたのちに，INTCSI割り込み要求を強制的に発生させています。

また，INTCSI割り込み処理では，直前の転送がマクロ・サービスの場合(MACF＝1)は，マクロ・サービス・タイム・アウト状態（MACTMF←1）に設定し，通常のINTCSI処理を行わずに復帰しています。

図10－13　マクロ・サービス・タイム・アウト処理



10
214
218A
224
234
244

(2) プログラム説明 … (6) プログラム・リスト参照

(i) タイム・アウト状態をチェックします。タイム・アウト状態でなければ戻ります。

(ii) タイム・アウト・カウンタ（MACCNT）をインクリメントします。

(iii) MACCNT<2ならば戻ります。

(iv) マクロ・サービス転送モード，マクロ・サービス・タイム・アウト状態をクリアします(MACF←0, MACTMF←0)。

(v) タイム・アウト・カウンタをクリアします（MACCNT←0)。

(vi) マクロ・サービス受信では受信のみ許可として使用していますので，送信も許可の状態に設定します。

(vii) INTCSI割り込み要求フラグをセットして戻ります。

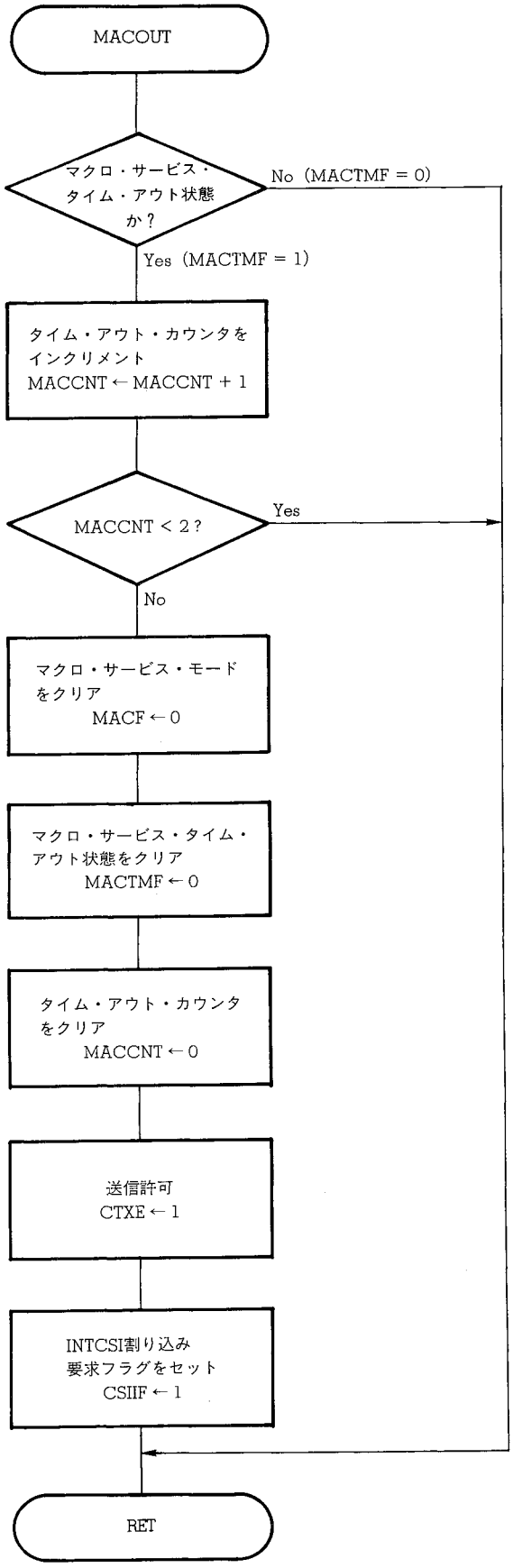
(3) 使用レジスタ

なし

(4) 入出力条件

入　出　力　条　件	出　力　条　件
なし	タイム・アウト処理完了ならば CSIIF＝1，MACF＝0，MACTMF＝0，CTXE＝1

(5) フロー・チャート



10
214
218A
224
234
244

(6) プログラム・リスト

```
;*************************************************************
;       MACRO SERVICE TIME OUT
;*************************************************************

MACOUT:
        BF      MACTMF,$MCT_J1          ; check macro service conditon
        INC     MACCNT
        CMP     MACCNT,#2
        BC      $MCT_J1
        CLR1    MACF                    ; Time Out
        CLR1    MACTMF
        MOV     MACCNT,#0
        SET1    CTXE
        SET1    CSIIF                   ; Set INTCSI
MCT_J1:
        RET
```

10.4.5　バックグランド処理

この応用例では，SBIを動作させるためのバックグランド処理を有しています。この応用例では，転送要求や送信データの設定を割り込み処理で行い，バックグランド処理は単純なループ処理を用いています。SBIの応用プログラムを実際のシステム等で応用される場合は，このループを開くなどの方法で利用してください。

また，SBIの動作をモニタするためのタイマ割り込みのエントリをバックグランドに有していますので，次のような，タイマ割り込みエントリのためのプログラムが必要です。

```
        ORG     0001CH

        DW      I_C21           ; INTC21 vector
                :
                :
I_C21:
        SEL     RB2
        CALL    !ACKCHK
        CALL    !MACOUT
                :
                :
        RETI
                :
                :
```

(1) 動作概要

(a) マスタ

マスタには，イニシャライズ後，アドレスを送信し，選択したスレーブ・デバイスにコマンドを送信し，通信動作を行います。バックグランド処理では，アドレス，コマンドの順に送信し，各コマンド処理へ分岐しています。

(b) スレーブ

スレーブは，マスタから一致アドレスを受信すると，ビジィ解除を行い，コマンド待ち状態になります。有効コマンドを受信したのちに，各コマンド処理へ分岐します。

全体的な流れは，フロー・チャートならびに状態遷移図を参照してください。

10
214
218A
224
234
244

(2) プログラム説明 … (4) プログラム・リスト参照

(a) マスタ・バックグランド処理 [レーベル名称：MT_INI]

(i) マスタのSBIのイニシャライズを行います。

(ii) アドレス送信要求のチェックとしてATFフラグをポーリングします。

(iii) ATFフラグをクリア(0)します。

(iv) アドレス送信を行います。

(v) 転送エラーが発生していれば(ii)へ。

(vi) コマンド送信要求のチェックとしてCTFフラグをポーリングします。

(vii) CTFフラグをクリア(0)します。

(viii) コマンド送信を行います。

(ix) 転送エラーが発生していれば(vi)へ。

(x) 送信したコマンド・コードにより各コマンド処理へ移行します。

(xi) 各コマンド処理が終了次第(vi)へ戻ります。

ただし，CHGMSTコマンド処理で，マスタ交換が可能である場合は，スレーブ処理[SL_INI]へ移行します。また，DETACHコマンド処理では，アドレス入力待ち(ii)へ移行します。

(b) スレーブ・バックグランド処理 [レーベル名称：SL_INI]

(i) スレーブのSBIのイニシャライズを行います。

(ii) 転送完了状態（TREND＝1）までウエイトします。

(iii) 転送完了フラグをクリア（TREND←0）します。

(iv) スレーブとして選択されていなければ(ii)へ戻り，アドレス受信を待ちます。

(v) コマンド処理中フラグをクリア（CMDBSY←0）します。

(vi) 転送完了フラグをクリア（TREND←0）し，コマンド受信に備えます。

(vii) SIO←FFHとし$\overline{\text{BUSY}}$を解除し，コマンド受信を待ちます。

(viii) 転送完了状態（TREND＝1）までウエイトします。

(ix) 転送完了フラグをクリア（TREND←0）します。

(x) 有効コマンドの受信をCRFフラグでチェックします（CRFは転送完了割り込み処理内でセットされます)。

有効コマンドが受信されていなければ（CRF＝0)，(v)へ戻ります。

(xi) 有効コマンドが受信された場合は，CRFをクリア(0)します。

(xii) コマンド処理中フラグをセット（CMDBSY←1）して，各コマンド処理へ移行します。

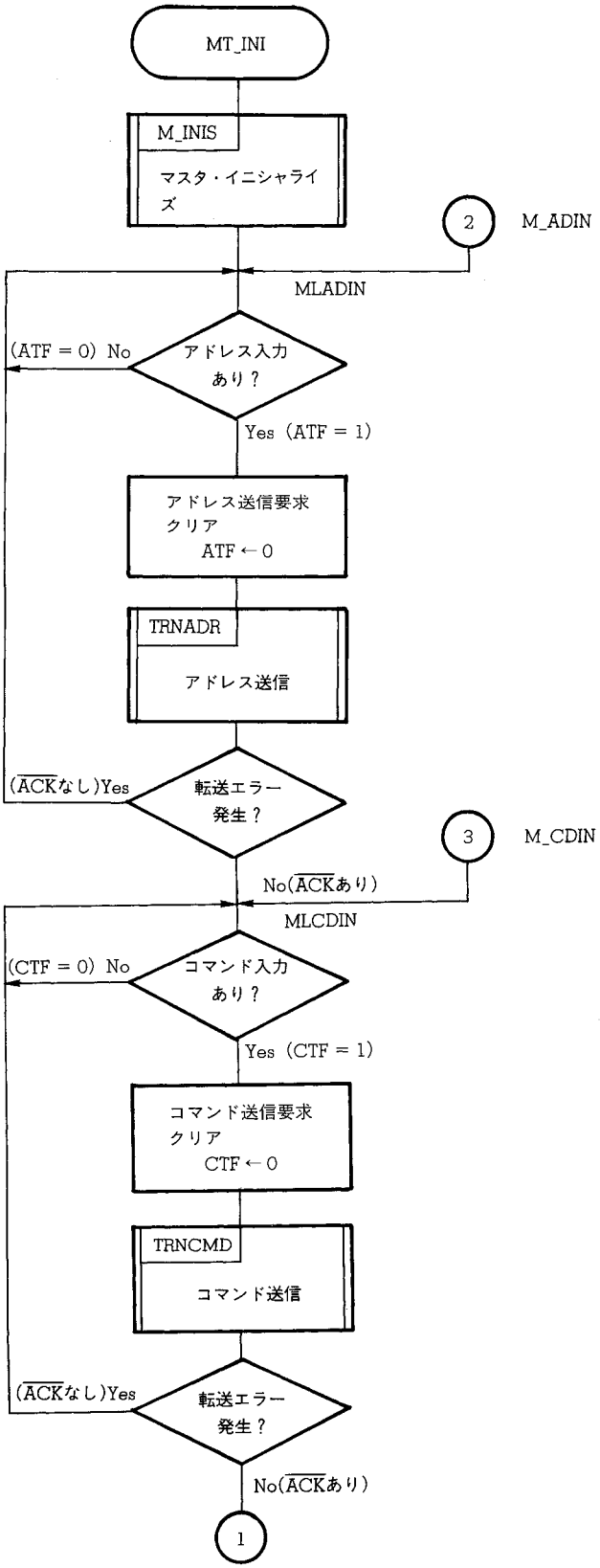
(xiii) 各コマンド処理が終了次第(v)へ戻ります。

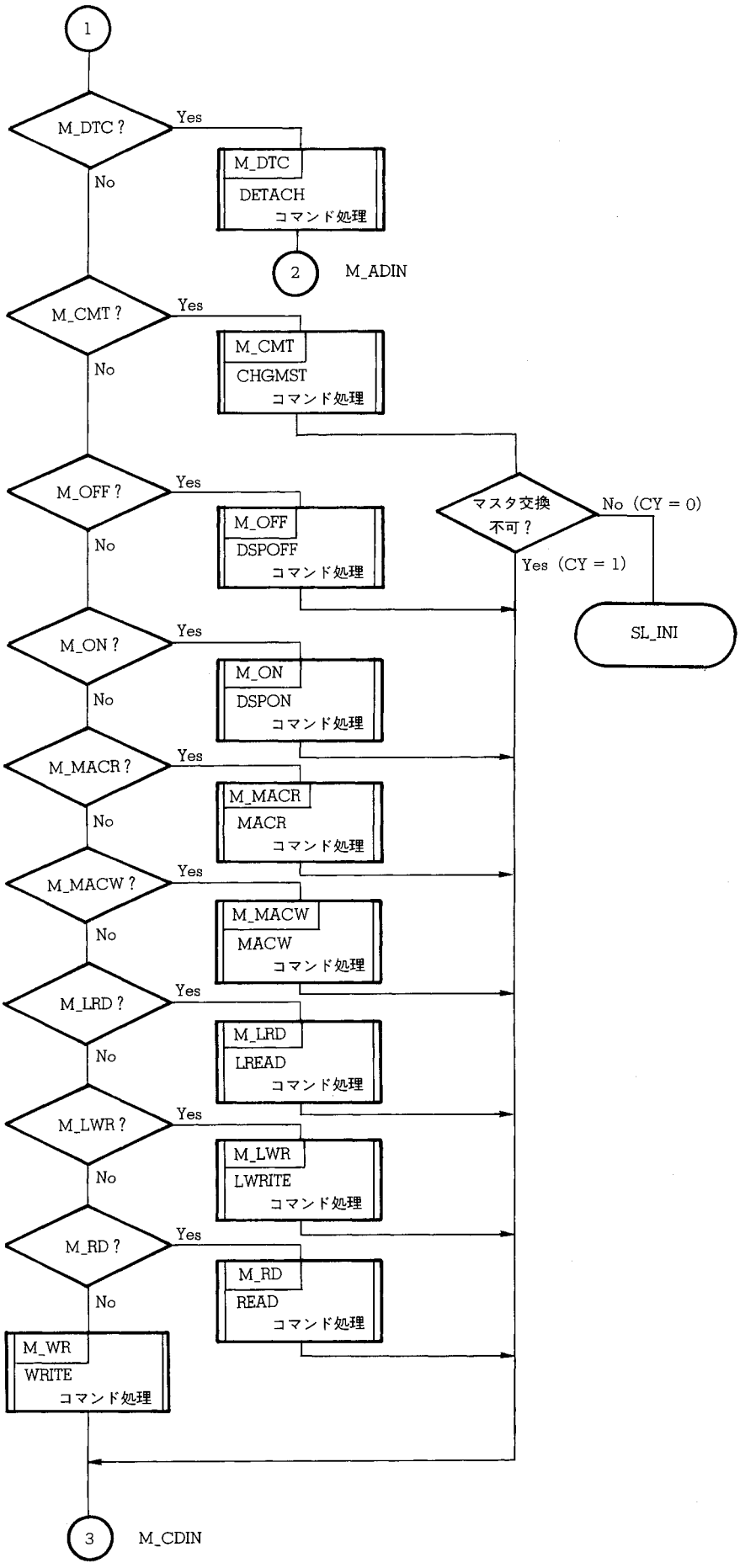
ただし，CHGMSTコマンド処理で，マスタ交換が可能である場合は，マスタ処理[MT_INI]へ移行します。また，DETACHコマンド処理では，アドレス受信待ちとなり，(ii)へ移行します。

〔メ　モ〕

(3) フロー・チャート

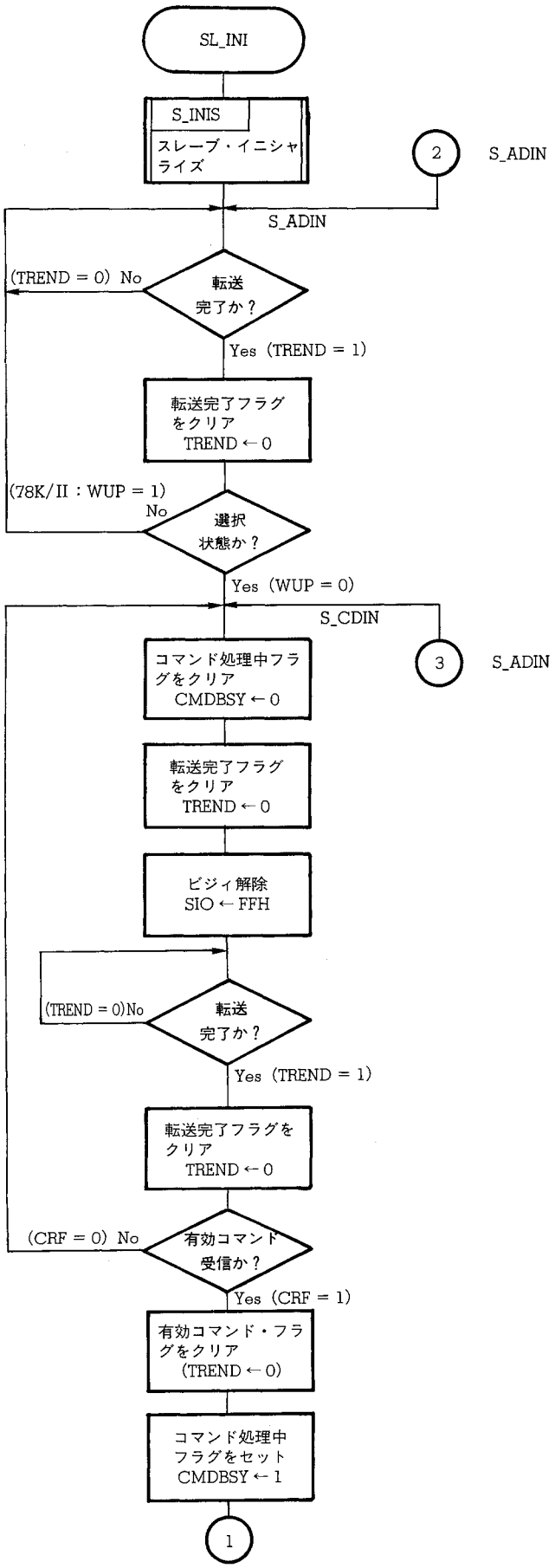
(a) マスタ

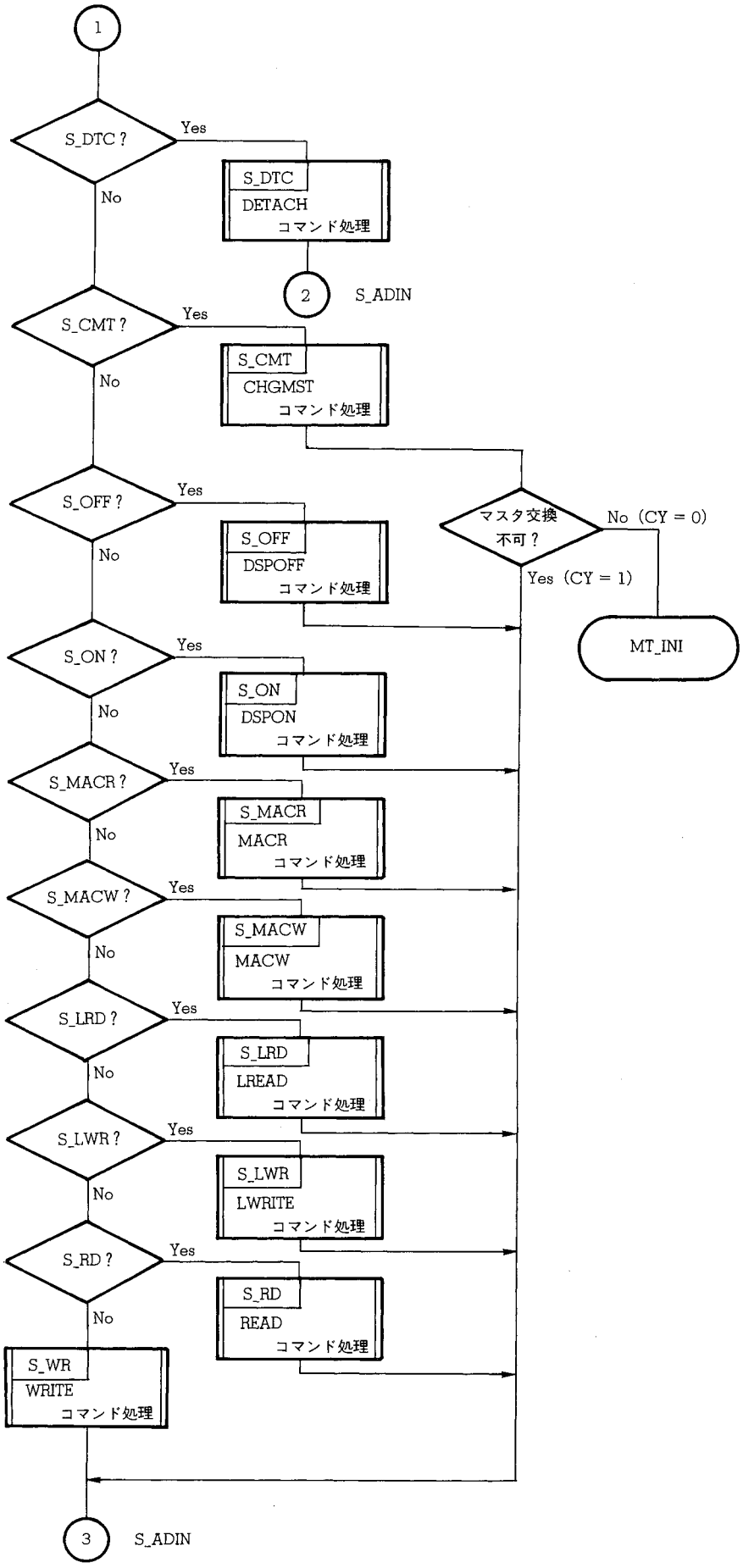




10
214
218A
224
234
244

(b) スレーブ


SL_INI
S_INIS
スレーブ・イニシャライズ
2
S_ADIN
S_ADIN
(TREND = 0) No
転送完了か？
Yes (TREND = 1)
転送完了フラグをクリア
TREND ← 0
(78K/II：WUP = 1)
No
選択状態か？
Yes (WUP = 0)
S_CDIN
3
S_ADIN
コマンド処理中フラグをクリア
CMDBSY ← 0
転送完了フラグをクリア
TREND ← 0
ビジィ解除
SIO ← FFH
(TREND = 0)No
転送完了か？
Yes (TREND = 1)
転送完了フラグをクリア
TREND ← 0
(CRF = 0) No
有効コマンド受信か？
Yes (CRF = 1)
有効コマンド・フラグをクリア
(TREND ← 0)
コマンド処理中フラグをセット
CMDBSY ← 1
1



10
214
218A
224
234
244

(4) プログラム・リスト (SBI関連リストの抜粋)

```
;************************************************************
;       MASTER MAIN (BACK GROUND)
;               : Polling Address
;               : Polling Commands
;               : Go to Commands
;************************************************************

MT_INI:
        CALL    !M_INIS
M_ADIN:                                 ; ADDRESS Wait
        BF      ATF,$M_ADIN
        CLR1    ATF
        CALL    !TRNADR
        BT      ERRORF,$M_ADIN
M_CDIN:                                 ; COMMAND Wait
        BF      CTF,$M_CDIN
        CLR1    CTF
        CALL    !TRNCMD
        BT      ERRORF,$M_CDIN
        MOVW    HL,#M_CDT
MII_L1:
        MOV     A,[HL+]
        CMP     A,TRNBUF
        BZ      $MII_J1
        INCW    HL
        INCW    HL
        BR      MII_L1
MII_J1:
        MOVW    AX,[HL]
        BR      AX
;
;       --- Commands Table ---
;
M_CDT:
        DB      20H                     ; WRITE
        DW      MWR
        DB      21H                     ; READ
        DW      MRD
        DB      22H                     ; LWRITE
        DW      MLWR
        DB      23H                     ; LREAD
        DW      MLRD
        DB      28H                     ; CHGMST
        DW      MCMT
        DB      29H                     ; DETACH
        DW      MDTC

        DB      24H                     ; MACW
        DW      MMACW
        DB      25H                     ; MACR
        DW      MMACR

        DB      31H                     ; DSPON
        DW      MON
        DB      32H                     ; DSPOFF
        DW      MOFF
```

```
;*************************************************************
;       COMMAND PROCESS (BACK GROUND)
;*************************************************************

MWR:
        CALL    !M_WR
        BR      M_CDIN
MRD:
        CALL    !M_RD
        BR      M_CDIN

MLWR:
        CALL    !M_LWR
        BR      M_CDIN
MLRD:
        CALL    !M_LRD
        BR      M_CDIN

MMACW:
        CALL    !M_MACW
        BR      M_CDIN
MMACR:
        CALL    !M_MACR
        BR      M_CDIN

MCMT:
        CALL    !M_CMT
        BC      $MCT_J1
        BR      SL_INI                  ; Change Master OK!
MCT_J1:
        BR      M_CDIN                  ; Change Master BAD!

MDTC:
        CALL    !M_DTC
        BR      M_ADIN

MON:
        CALL    !M_ON
        BR      M_CDIN
MOFF:
        CALL    !M_OFF
        BR      M_CDIN
```

10
214
218A
224
234
244

```
;************************************************************
;       SLAVE (BACK GROUND)
;               : Polling Address Receive
;               : Polling Commands Recieve
;               : Go to Commands
;************************************************************

SL_INI:
        CALL    !S_INIS
S_ADIN:
        BF      TREND,$S_ADIN
        CLR1    TREND
        BT      WUP,$S_ADIN             ; ADDRESS Wait
S_CDIN:
        CLR1    CMDBSY                  ; COMMAND Wait
        CLR1    TREND
        MOV     SIO,#0FFH               ; Clear Busy
ACDWAT:
        BF      TREND,$ACDWAT
        CLR1    TREND
        BF      CRF,$S_CDIN
        CLR1    CRF
        SET1    CMDBSY
        MOVW    HL,#S_CDT
SII_L1:
        MOV     A,[HL+]
        CMP     A,RCVBUF
        BZ      $SII_J1
        INCW    HL
        INCW    HL
        BR      SII_L1
SII_J1:
        MOVW    AX,[HL]
        BR      AX
```

```
;
;       --- Commands Table ---
;
S_CDT:
        DB      20H                     ; WRITE
        DW      SWR
        DB      21H                     ; READ
        DW      SRD
        DB      22H                     ; LWRITE
        DW      SLWR
        DB      23H                     ; LREAD
        DW      SLRD
        DB      28H                     ; CHGMST
        DW      SCMT
        DB      29H                     ; DETACH
        DW      SDTC

        DB      24H                     ; MACW
        DW      SMACW
        DB      25H                     ; MACR
        DW      SMACR

        DB      31H                     ; DSPON
        DW      SON
        DB      32H                     ; DSPOFF
        DW      SOFF
```

```
;*************************************************************
;       COMMAND PROCESS (BACK GROUND)
;*************************************************************

SWR:
        CALL    !S_WR
        BR      S_CDIN
SRD:
        CALL    !S_RD
        BR      S_CDIN

SLWR:
        CALL    !S_LWR
        BR      S_CDIN
SLRD:
        CALL    !S_LRD
        BR      S_CDIN

SMACW:
        CALL    !S_MACW
        BR      S_CDIN
SMACR:
        CALL    !S_MACR
        BR      S_CDIN

SCMT:
        CALL    !S_CMT
        BC      $SCT_J1
        BR      MT_INI                  ; Change Master OK!
SCT_J1:
        BR      S_CDIN                  ; Change Master BAD!

SDTC:
        CALL    !S_DTC
        BR      S_ADIN

SON:
        CALL    !S_ON
        BR      S_CDIN
SOFF:
        CALL    !S_OFF
        BR      S_CDIN
```

10.4.6　イニシャライズ処理

SBIのイニシャライズを行います。

(1) 動作概要

ここでは，この応用例のSBI部分の動作を行うために必要なイニシャライズを行っています。項目を次に示します。

(a) マスタ

(i) ボー・レート　375 kbps

8ビット・タイマ／カウンタ3カウント・クロック　：$f_{CLK}/8$

8ビット・タイマ／カウンタ3コンペア・レジスタCR30：0

(ii) CSIMレジスタ値　11001001B

(iii) SBICレジスタ値　00000001B

(iv) PMC3レジスタ値　00001100B

(v) 割り込み処理イニシャライズ

(vi) フラグ・イニシャライズ

備考　μPD78224シリーズでは，8ビット・タイマ／カウンタ3はボー・レート・ジェネレータになります。

タイマ・コントロール・レジスタ0（TMC0），プリスケーラ・モード・レジスタ0（PRM0），8ビット・タイマ／カウンタ3（μPD78224シリーズではボー・レート・ジェネレータ）のコンペア・レジスタ（CR30），クロック同期式シリアル・インタフェース・モード・レジスタ（CSIM）への操作は，変更せずに使用できます。

(b) スレーブ

(i) SIO初期値　FFH

(ii) CSIMレジスタ値　11101000B

(iii) SBICレジスタ値　10000001B

(iv) PMC3レジスタ値　00001100B

(v) 割り込み処理のイニシャライズ

(vi) フラグ・イニシャライズ

10
214
218A
224
234
244

(2) プログラム説明 … (6) プログラム・リスト参照

(a) マスタ・イニシャライズ処理［レーベル名称：M_INIS］

(i) マスタとしてのイニシャライズ完了まで，割り込みを禁止します。

(ii) スレーブのイニシャライズ完了までウエイトします。

(iii) $\overline{SCK}$，SB0端子の入出力と初期レベルを設定します。

(iv) ボー・レートを375 kbpsに設定します。

(v) RELT←1とし，シリアル出力ラッチ（SO）をハイ・レベルにセットします。

(vi) CSIMレジスタのマスタとしてのイニシャライズを行います。

(vii) $\overline{ACK}$チェックと，マクロ・サービス・タイム・アウトのために用いるインターバル・タイマ（INTC21）のイニシャライズを行います。

(viii) インターバル・タイマ（INTC21）の割り込みを許可します。

(ix) $\overline{SCK}$，SB0端子をコントロール・ポートに設定します。

(x) 送受信を許可します。

ここで，受信禁止から許可（CRXE＝0→CRXE＝1）とすると，受信動作が起動され，スレーブ側から見ると，データが送られた状態となります。ただし，スレーブはアドレス待ち状態ですので，影響はありません。

(xi) INTCSIの割り込みを許可します。

(xii) INTCSI割り込み要求（CSIIF）をポーリングし，セットされていればクリアし，$\overline{ACK}$を出力します。

(xiii) マスタの送受信用にフラグ・エリアをイニシャライズします。

(xiv) 割り込みを許可します。

(b) スレーブ・イニシャライズ処理［レーベル名称：S_INIS］

(i) RELT←1とし，シリアル出力ラッチ（SO）をハイ・レベルにセットします。

(ii) BSYE←1として，ビジィ信号の同期出力を許可します。

(iii) CSIMレジスタのスレーブとしてのイニシャライズを行います。

(iv) $\overline{ACK}$チェックと，マクロ・サービス・タイム・アウトのために用いるインターバル・タイマ（INTC21）のイニシャライズを行います。

(v) ポートのイニシャライズを行います。

(vi) INTCSI割り込みを許可します。

(vii) スレーブの送受信用にフラグ・エリアをイニシャライズします。

(viii) 割り込みを許可します。

(3) 使用レジスタ

マスタ	B
スレーブ	なし

(4) モード・レジスタ，ワーク・フラグ設定一覧

(a) マスタ

クロック同期式シリアル・インタフェース・モード・レジスタ

	7	6	5	4	3	2	1	0	
CSIM	CTXE	CRXE	WUP	0	MOD1	0	CLSI	CLS0	(リセットで00H)
設定値	1	1	0	0	1	0	0	1	

CLS1	CLS0	シリアル・クロックの選択		SCK端子
0	0	外部クロック		入力
0	1	内部クロック	タイマ/カウンタ3出力/2	出力
1	0		$f_{CLK}/32$[注]	
1	1		$f_{CLK}/8$[注]	

MOD1	動作モード選択
0	3線式シリアルI/Oモード
1	SBIモード

WUP	ウエイク・アップ機能の制御
0	すべてのモードでシリアル転送後ごとに，割り込み要求信号を発生
1	SBIモードでアドレスを受信した場合のみ割り込み要求信号を発生

CRXE	受信動作
0	禁止
1	許可

CTXE	送信動作
0	禁止
1	許可

注　f_{CLK}：内部システム・クロック周波数（$f_x/2$）

10
214
218A
224
234
244

シリアル・バス・インタフェース・モード・レジスタ

	7	6	5	4	3	2	1	0	
SBIC	BSYE	ACKD	ACKE	ACKT	CMDD	RELD	CMDT	RELT	（リセットで00H）
設定値	0	0	0	0	0	0	0	1	

RELT	W	バス・リリース信号（REL）のトリガ出力 ⎍
0	出力しない	
1	出力する	

CMDT	W	コマンド信号（CMD）のトリガ出力 ⎎
0	出力しない	
1	出力する	

RELD	R	バス・リリース信号（REL）検出フラグ ⎍
0	バス・リリース信号未検出	
1	バス・リリース信号検出 $\overline{SCK}$がハイ・レベルの期間にシリアル・バスがロウ・レベルからハイ・レベルへ変化	

CMDD	R	コマンド信号（CMD）検出フラグ ⎎
0	コマンド信号未検出	
1	コマンド信号検出 $\overline{SCK}$がハイ・レベルの期間にシリアル・バスがハイ・レベルからロウ・レベルへ変化	

ACKT	W	アクノリッジ信号（$\overline{ACK}$）トリガ出力
0	出力しない	
1	出力する	

ACKE	R/W	アクノリッジ信号（$\overline{ACK}$）許可
0	自動出力を禁止	
1	自動出力を許可	

ACKD	R	アクノリッジ信号（$\overline{ACK}$）の検出
0	検出していない	
1	検出している	

BSYE	R/W	ビジィ信号（$\overline{BUSY}$）の許可，ビジィのクリア
0	自動出力を禁止，ビジィをクリア	
1	自動出力を許可	

フラグ・ワークSBIFL0

	7	6	5	4	3	2	1	0
SBIFL0	MSTF	0	CMDBSY	PFLAG	RCVFLG	ACKWFG	TREND	ERRORF
設定値	1	0	0	0	0	0	0	0

フラグ・ワークSBIFL1

	7	6	5	4	3	2	1	0
SBIFL1	MACF	0	0	MACTMF	ATF	CTF	CRF	DTF
設定値	0	0	0	0	0	0	0	0

(b) スレーブ

クロック同期式シリアル・インタフェース・モード・レジスタ

	7	6	5	4	3	2	1	0	
CSIM	CTXE	CRXE	WUP	0	MOD1	0	CLS1	CLS0	(リセットで00H)
設定値	1	1	1	0	1	0	0	0	

CLS1	CLS0	シリアル・クロックの選択		SCK端子
0	0	外部クロック		入力
0	1	内部クロック	タイマ/カウンタ3出力/2	出力
1	0		$f_{CLK}/32$[注]	
1	1		$f_{CLK}/8$[注]	

MOD1	動作モード選択
0	3線式シリアルI/Oモード
1	SBIモード

WUP	ウエイク・アップ機能の制御
0	すべてのモードでシリアル転送後ごとに，割り込み要求信号を発生
1	SBIモードでアドレスを受信した場合のみ割り込み要求信号を発生

CRXE	受信動作
0	禁止
1	許可

CTXE	送信動作
0	禁止
1	許可

注　f_{CLK}：内部システム・クロック周波数（$f_x/2$）

10
214
218A
224
234
244

シリアル・バス・インタフェース・モード・レジスタ

	7	6	5	4	3	2	1	0	
SBIC	BSYE	ACKD	ACKE	ACKT	CMDD	RELD	CMDT	RELT	（リセットで00H）
設定値	1	0	0	0	0	0	0	1	

RELT	W	バス・リリース信号（REL）のトリガ出力 ⎍
0	出力しない	
1	出力する	

CMDT	W	コマンド信号（CMD）のトリガ出力 ⎎
0	出力しない	
1	出力する	

RELD	R	バス・リリース信号（REL）検出フラグ ⎍
0	バス・リリース信号未検出	
1	バス・リリース信号検出 $\overline{SCK}$がハイ・レベルの期間にシリアル・バスがロウ・レベルからハイ・レベルへ変化	

CMDD	R	コマンド信号（CMD）検出フラグ ⎎
0	コマンド信号未検出	
1	コマンド信号検出 $\overline{SCK}$がハイ・レベルの期間にシリアル・バスがハイ・レベルからロウ・レベルへ変化	

ACKT	W	アクノリッジ信号（$\overline{ACK}$）トリガ出力
0	出力しない	
1	出力する	

ACKE	R/W	アクノリッジ信号（$\overline{ACK}$）許可
0	自動出力を禁止	
1	自動出力を許可	

ACKD	R	アクノリッジ信号（$\overline{ACK}$）の検出
0	検出していない	
1	検出している	

BSYE	R/W	ビジィ信号（$\overline{BUSY}$）の許可，ビジィのクリア
0	自動出力を禁止，ビジィをクリア	
1	自動出力を許可	

フラグ・ワークSBIFL0

	7	6	5	4	3	2	1	0
SBIFL0	MSTF	0	CMDBSY	PFLAG	RCVFLG	ACKWFG	TREND	ERRORF
設定値	0	0	0	0	1	0	0	0

フラグ・ワークSBIFL1

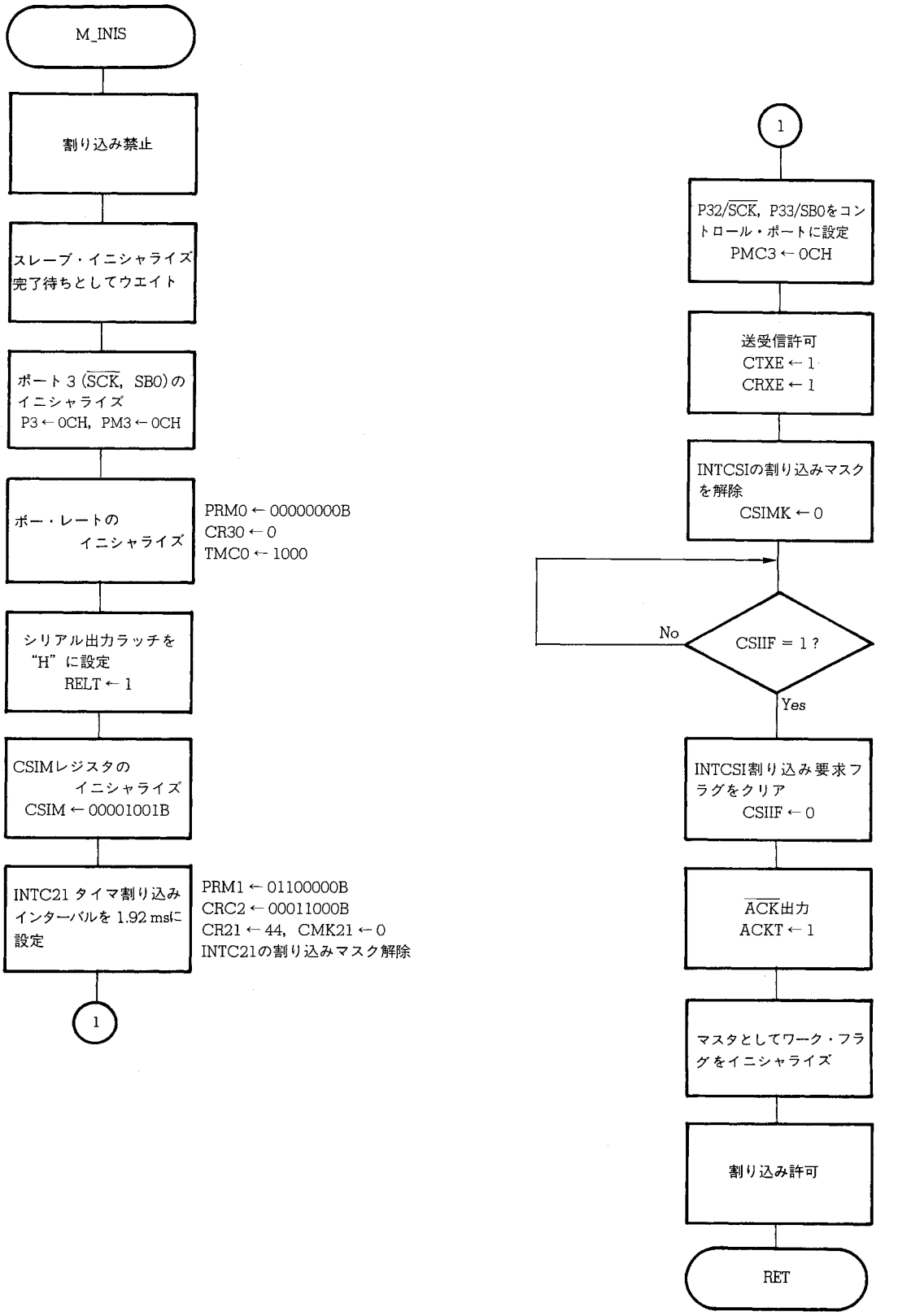
	7	6	5	4	3	2	1	0
SBIFL1	MACF	0	0	MACTMF	ATF	CTF	CRF	DTF
設定値	0	0	0	0	0	0	0	0

〔メ　モ〕

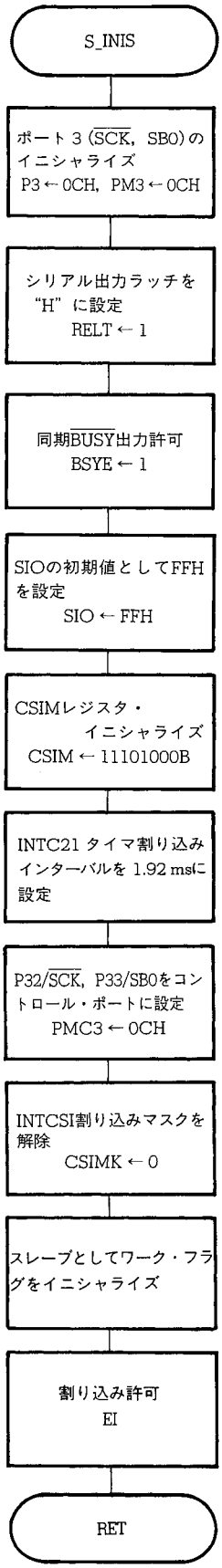
214
218A
224
234
244

(5) フロー・チャート

(a) マスタ



(b) スレーブ



10
214
218A
224
234
244

(6) プログラム・リスト

```
;*************************************************************
;       INITIALIZE PACKAGE
;               : Master & Slave SBI Initialize
;*************************************************************
;
;       --- For Master ---
;
M_INIS:
        DI

        MOV     B,#229                  ; 800us wait
MIN_L1:
        NOP
        NOP
        NOP
        NOP
        DBNZ    B,$MIN_L1

        MOV     P3,#0CH                 ; port initialize
        MOV     PM3,#0CH

        MOV     PRM0,#00000000B         ; SBI initialize
        MOV     CR30,#0
        MOV     TMC0,#10000000B
        SET1    RELT
        MOV     CSIM,#00001001B         ; SBI mode , WUP=0 , TM3 clock

        MOV     PRM1,#01100000B         ; INTC21 1.92ms
        MOV     CRC2,#00011000B
        MOV     CR21,#44
        CLR1    CMK21
        MOV     TMC1,#10000000B

        MOV     PMC3,#0CH
        SET1    CTXE                    ; transmit,receive enable
        SET1    CRXE

        CLR1    CSIMK                   ; First data transmit wait
MIN_L2:
        BTCLR   CSIIF,$MIN_J1
        BR      MIN_L2
MIN_J1:
        SET1    ACKT
        MOV     SBIFL0,#10000000B
        MOV     SBIFL1,#00000000B

        EI
        RET
```

```
;
;       --- For Slave ---
;
S_INIS:
        MOV     P3,#0CH                 ; port initialize
        MOV     PM3,#0CH

        SET1    RELT                    ; SBI initialize
        SET1    BSYE
        MOV     SIO,#0FFH
        MOV     CSIM,#11101000B         ; SBI mode , WUP=1 , ext.clock

        MOV     PRM1,#01100000B         ; INTC21 1.92ms
        MOV     CRC2,#00011000B
        MOV     CR21,#44
        CLR1    CMK21
        MOV     TMC1,#10000000B

        MOV     PMC3,#0CH
        CLR1    CSIMK
        MOV     SBIFL0,#00001000B
        MOV     SBIFL1,#00000000B

        EI
        RET
```

10.4.7　WRITEコマンド処理

マスタ→スレーブへのデータ転送コマンドです。データ数は１バイト固定です。

コマンド値	20H

(1) 動作概要

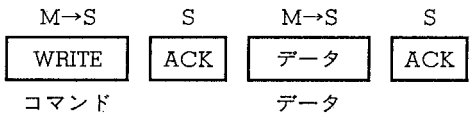
(a) マスタ側

コマンド送信後，WRITEコマンド・データ領域（WRDATA）の値を送信します。

(b) スレーブ側

コマンド受信後は，１バイトのデータを受信し，受信バッファ（RCVBUF）に格納します。

(c) 転送のフォーマット



(2) プログラム説明

(a) マスタWRITEコマンド処理［レーベル名称：M_WR］…（6）プログラム・リスト参照

(i) 送信データとして，WRITEコマンド・データ領域(WRDATA)の値を送信バッファ(TRNBUF)に設定します。

(ii) データ送信のサブルーチン（TRNDAT）を呼び出し，スレーブにデータを送信します。

(b) スレーブWRITEコマンド処理［レーベル名称：S_WR］

(i) データ受信のサブルーチン（RCVDAT）を呼び出し，マスタからデータを受信します。

(3) 使用レジスタ

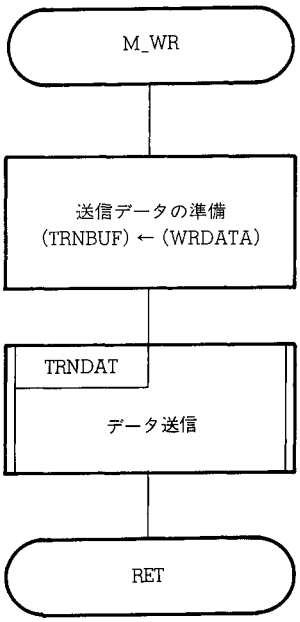
マスタ	A
スレーブ	A

(4) 入出力条件

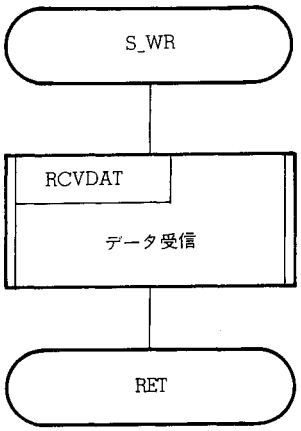
デバイス	入　力　条　件	出　力　条　件
マスタ	(WRDATA) ←送信データ	(TRNBUF) ← (WRDATA)
スレーブ	なし	(RCVBUF) ←受信データ

(5) フロー・チャート

(a) マスタ



(b) スレーブ



10
214
218A
224
234
244

(6) プログラム・リスト

```
;-------------------------------------------------------
;       [[[  WRITE  ]]]
;               Master -> Slave  1byte Data
;               [CMD] A  [DAT] A
;                 M   S    M   S
;-------------------------------------------------------
;
;       ... MASTER ...
;
M_WR:
        MOV     TRNBUF,WRDATA
        CALL    !TRNDAT
        RET
;
;       ... SLAVE ...
;
S_WR:
        CALL    !RCVDAT
        RET
```

10.4.8　READコマンド処理

スレーブ→マスタへのデータ転送コマンドです。データ数は1バイト固定です。

コマンド値	21H

(1) 動作概要

(a) マスタ側

コマンド送信後，スレーブからの$\overline{ACK}$を検出し次第，受信動作を行います。この直後の転送完了割り込みで受け取ったデータを，受信バッファ（RCVBUF）に格納します。

(b) スレーブ側

コマンド受信後，DTFフラグがセットされ次第，送信バッファ(TRNBUF)の内容をマスタに送信します。

(c) 転送のフォーマット

M→S	S	S→M	M
READ	ACK	データ	ACK
コマンド		データ	

(2) プログラム説明 … (6) プログラム・リスト参照

(a) マスタREADコマンド処理［レーベル名称：M_RD］

(i) データ受信のサブルーチン（RCVDAT）を呼び出し，スレーブからデータを受信します。

(b) スレーブREADコマンド処理［レーベル名称：S_RD］

(i) スレーブの転送準備待ちとして，DTFフラグをポーリングします。DTF＝1ならばクリアし，データ送信のサブルーチン（TRNDAT）を呼び出し，マスタにデータを送信します。

(ii) データ送信のサブルーチン（TRNDAT）では，復帰時に送信モード（RCVFLG=0）に設定されているため，受信モード（RCVFLG=1）に設定します。

10
214
218A
224
234
244

(3) 使用レジスタ

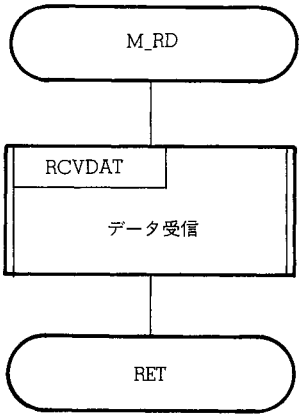
マスタ	A
スレーブ	A

(4) 入出力条件

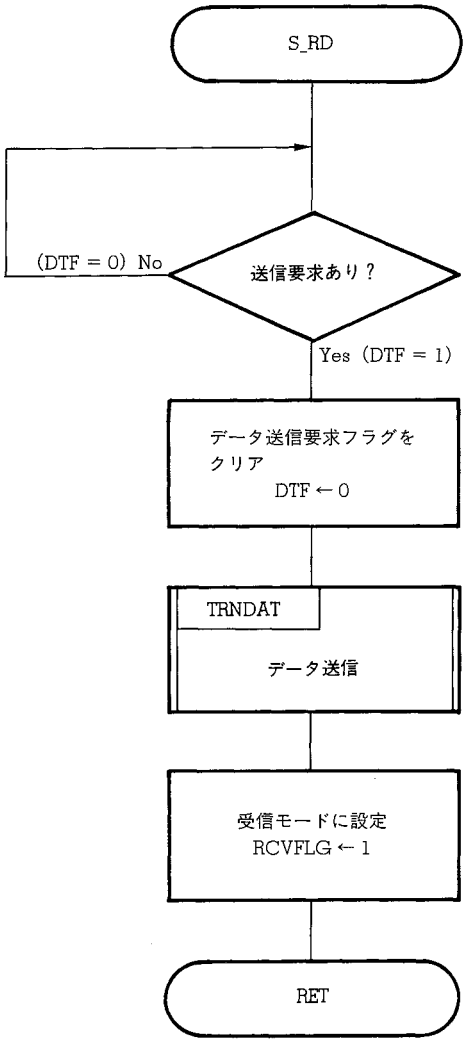
デバイス	入　力　条　件	出　力　条　件
マスタ	なし	(RCVBUF) ←受信データ
スレーブ	DTF←1, (TRNBUF) ←送信データ	DTF←0, RCVFLG←1

(5) フロー・チャート

(a) マスタ



(b) スレーブ



(6) プログラム・リスト

```
;-------------------------------------------------------
;       [[[  READ  ]]]
;               Slave -> Master  1byte Data
;               [CMD] A  [DAT] A
;                 M   S    S   M
;-------------------------------------------------------
;
;       ... MASTER ...
;
M_RD:
        CALL    !RCVDAT
        RET
;
;       ... SLAVE ...
;
S_RD:
        BF      DTF,$S_RD
        CLR1    DTF
        CALL    !TRNDAT
        SET1    RCVFLG
        RET
```

10.4.9　LWRITEコマンド処理

連続データのマスタ→スレーブ転送コマンドです。1-256バイトのデータをマスタ→スレーブへ転送します。転送するデータ数はマスタが指定します。データ数に0を指定した場合，256バイトになります。

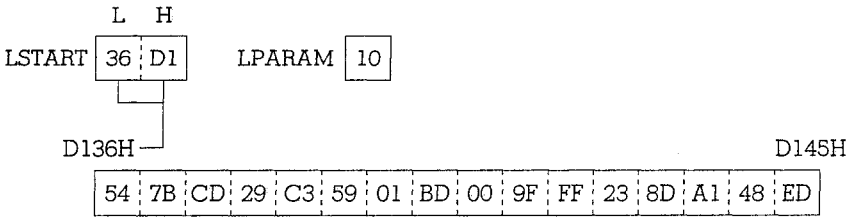
コマンド値	22H

(1) 動作概要

(a) マスタ側

RAM内のパラメータ（LPARAM）で指定されたバイト数分，RAM内の（LSTART）で指定されたアドレス以降のデータを送信します。図10－14に例を示します。

図10－14　LWRITEコマンドによる転送データ



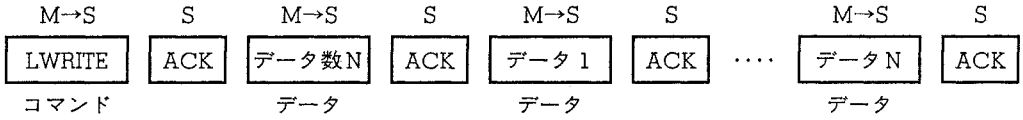
(b) スレーブ側

マスタから指定されたバイト数分のデータの受信を行い，RAM内の(LSTART)で指定されたアドレス以降に受信データを格納します。

データ数の受信時には，(LPARAM)に受信可能バイト数を設定し，パラメータ・フラグ(PFLAG)をセットすることで，バッファ・サイズを越える受信を禁止しています。データ数の受信は，INTCSI割り込み処理で行われ，マスタから指定されたデータ数と（LPARAM）を比較し，受信データが（LPARAM）を越える場合は，$\overline{\text{ACK}}$は出力せず$\overline{\text{BUSY}}$の解除条件のみを成立させています。

注意　転送データ数((LPARAM))はコマンド処理を呼び出す前に設定してください。また（LPARAM）の値はコマンド処理中で破壊されます。

(c) 転送フォーマット



(2) プログラム説明 … (6) プログラム・リスト参照

(a) マスタLWRITEコマンド処理［レーベル名称：M_LWR］

(i) 送信データとして，送信データ数（(LPARAM) の値）を送信バッファ（TRNBUF）に書き込みます。

(ii) データ送信のサブルーチン（TRNDAT）を呼び出し，スレーブに送信データ数を送信します。

(iii) 転送エラーが発生した場合（ERRORF＝1）は，LWRITEコマンド処理を終了して戻ります。

(iv) データ数の送信が正常に実行された場合(ERRORF＝0)は，ポインタとしてHLレジスタに転送データ格納エリアの先頭アドレスを設定します。またカウンタとして，Bレジスタに送信データ数（(LPARAM) の値）を設定します。

(v) ポインタで示される送信データを読み出し，送信バッファ(TRNBUF)に設定します。ポインタはインクリメントします。

(vi) データ送信のサブルーチン（TRNDAT）を呼び出して，スレーブにデータを送信します。

(vii) 転送エラーが発生した場合（ERRORF＝1）は，LWRITEコマンド処理を終了して戻ります。

(viii) データ送信が正常に実行された場合(ERRORF＝0)は，カウンタをデクリメントし，指定した回数の転送が終了するまで(v)-(viii)を繰り返します。

(b) スレーブLWRITEコマンド処理［レーベル名称：S_LWR］

(i) パラメータ待ちフラグ（PFLAG）をセットします。

(ii) データ受信のサブルーチン（RCVDAT）を呼び出し，マスタから受信データ数を受信します。

(iii) 転送エラーが発生した場合（ERRORF＝1）は，LWRITEコマンド処理を終了して戻ります。

(iv) データ数の受信が正常に実行された場合（ERRORF＝0）は，ポインタとしてHLレジスタに転送データ格納エリアの先頭アドレスを設定します。またカウンタとして，Bレジスタに送信データ数（(LPARAM) の値）を設定します。

(v) データ受信のサブルーチン（RCVDAT）を呼び出し，マスタからデータを受信します。

(vi) 転送エラーが発生した場合（ERRORF＝1）は，LWRITEコマンド処理を終了して戻ります。

(vii) データ受信が正常に実行された場合（ERRORF＝0）は，受信データをポインタで示されるメモリに格納します。

(viii) カウンタをデクリメントし，指定した回数の転送が終了するまで(v)-(viii)を繰り返します。

(3) 使用レジスタ

マスタ	A, X, B, HL
スレーブ	A, X, B, HL

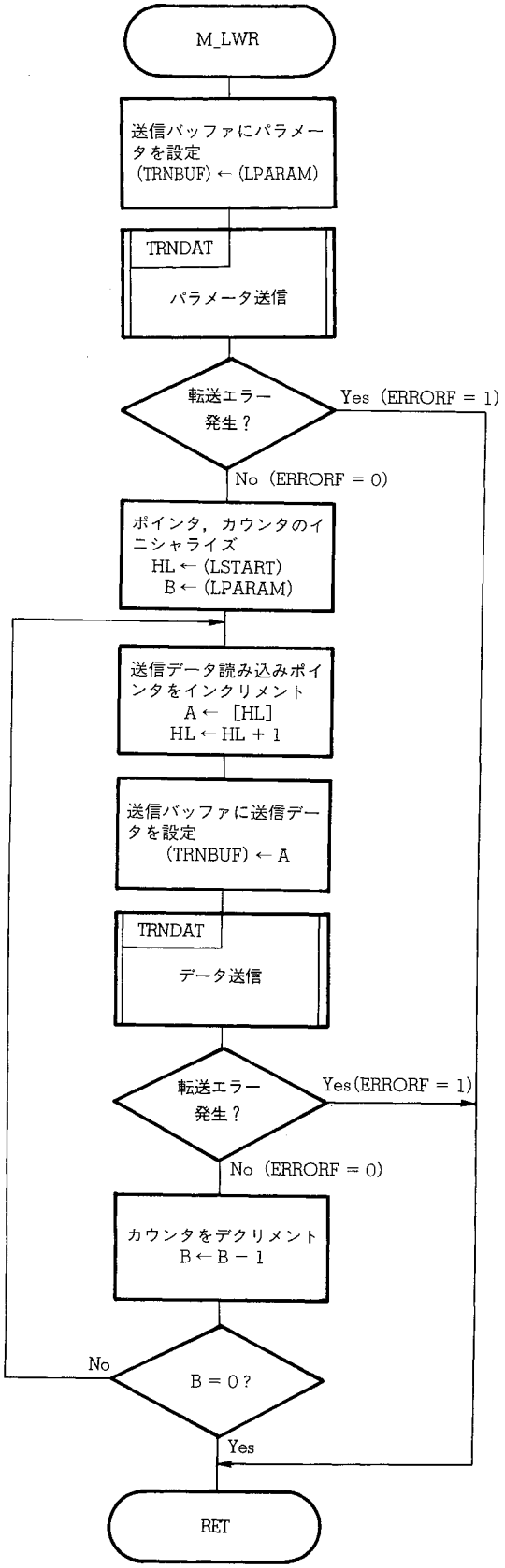
(4) 入出力条件

デバイス	入　力　条　件	出　力　条　件
マスタ	(LSTART) ←データ先頭アドレス (LPARAM) ←送信データ数	(LSTART), (LPARAM)：保持 ①　正常転送　：ERRORF←0 ②　転送エラー：ERRORF←1
スレーブ	(LSTART) ←データ格納先頭アドレス (LPARAM) ←受信可能データ数	①　正常転送： ERRORF←0 (LSTART) 以降に受信データ (LPARAM) ←マスタからの受信データ数 ②　転送エラー： ERRORF←1

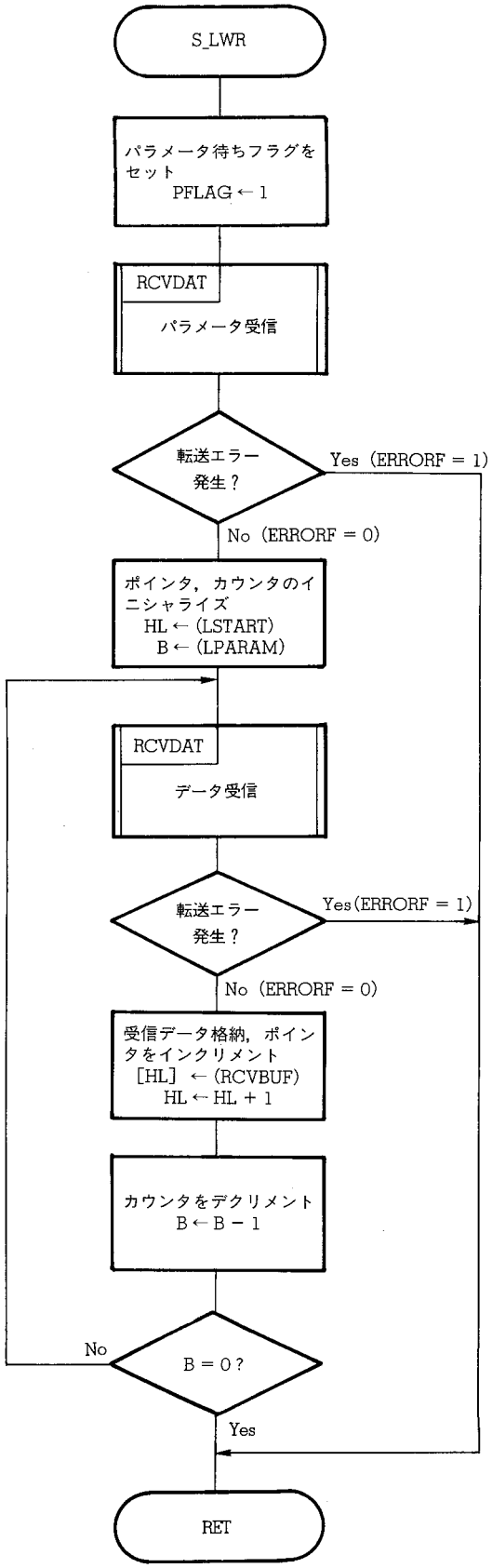
〔メ　モ〕

(5) フロー・チャート

(a) マ ス タ



(b) スレーブ



10
214
218A
224
234
244

(6) プログラム・リスト

```
;-------------------------------------------------------
;       [[[  LWRITE  ]]]
;               Master -> Slave  n byte Data
;               [CMD] A  [DAT] A  [DAT] A ···· [DAT] A
;                 M   S    M   S    M   S        M   S
;-------------------------------------------------------
;
;       ... MASTER ...
;
M_LWR:
        MOV     TRNBUF,LPARAM
        CALL    !TRNDAT
        BT      ERRORF,$MLW_J1

        MOVW    AX,LSTART               ; Pointer,Counter Set
        MOVW    HL,AX
        MOV     A,LPARAM
        MOV     B,A
MLW_L1:                                 ; Read & Transmit Loop
        MOV     A,[HL+]
        MOV     TRNBUF,A
        CALL    !TRNDAT
        BT      ERRORF,$MLW_J1
        DBNZ    B,$MLW_L1
MLW_J1:
        RET
;
;       ... SLAVE ...
;
S_LWR:
        SET1    PFLAG
        CALL    !RCVDAT
        BT      ERRORF,$SLW_J1

        MOVW    AX,LSTART               ; Pointer,Counter Set
        MOVW    HL,AX
        MOV     A,LPARAM
        MOV     B,A
SLW_L1:                                 ; Receive & Write Loop
        CALL    !RCVDAT
        BT      ERRORF,$SLW_J1
        MOV     A,RCVBUF
        MOV     [HL+],A
        DBNZ    B,$SLW_L1
SLW_J1:
        RET
```

10.4.10　LREADコマンド処理

連続データのスレーブ→マスタ転送コマンドです。1-256バイトのデータをスレーブ→マスタへ転送します。転送するデータ数はマスタが指定します。データ数に0を設定した場合，256バイトになります。

コマンド値	23H

(1) 動作概要

(a) マスタ側

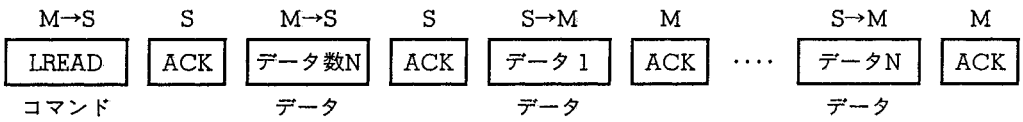
RAM内のパラメータ（LPARAM）で指定されたバイト数分，RAM内の（LSTART）で指定されたアドレス以降に受信データを格納します。RAM上でのデータ構成は，LWRITEコマンドと共通です。

(b) スレーブ側

RAM内の(LSTART)で指定されたアドレス以降のデータを，マスタから指定されたバイト数分マスタへ送信します。データ数のチェック方法はLWRITEコマンドと同じです。

注意　転送データ数((LPARAM))はコマンド処理を呼び出す前に設定してください。また（LPARAM）の値はコマンド処理中で破壊されます。

(c) 転送のフォーマット



(2) プログラム説明 … (6) プログラム・リスト参照

(a) マスタLREADコマンド処理［レーベル名称：M_LRD］

(i) 送信データとして，受信データ数（(LPARAM) の値）を送信バッファ（TRNBUF）に書き込みます。

(ii) データ送信のサブルーチン（TRNDAT）を呼び出し，スレーブに受信データ数を送信します。

(iii) 転送エラーが発生した場合（ERRORF＝1）は，LREADコマンド処理を終了して戻ります。

(iv) データ数の送信が正常に実行された場合(ERRORF＝0)は，ポインタとしてHLレジスタに転送データ格納エリアの先頭アドレスを設定します。またカウンタとして，Bレジスタに送信データ数（(LPARAM) の値）を設定します。

(v) データ受信のサブルーチン（RCVDAT）を呼び出し，スレーブからデータを受信します。

(vi) 転送エラーが発生した場合（ERRORF＝1）は，LREADコマンド処理を終了して戻ります。

(vii) データ受信が正常に実行された場合(ERRORF＝0)は，受信データをポインタで示されるメモリに格納します。

(viii) カウンタをデクリメントし，指定した回数の転送が終了するまで(v)-(viii)を繰り返します。

(b) スレーブLREADコマンド処理［レーベル名称：S_LRD］

(i) パラメータ待ちフラグ（PFLAG）をセットします。

(ii) データ受信のサブルーチン（RCVDAT）を呼び出し，マスタから受信データ数を受信します。

(iii) 転送エラーが発生した場合（ERRORF＝1）は，(ix)へ。

(iv) データ数の送信が正常に実行された場合(ERRORF＝0)は，ポインタとしてHLレジスタに転送データ格納エリアの先頭アドレスを設定します。またカウンタとして，Bレジスタに送信データ数（(LPARAM) の値）を設定します。

(v) ポインタで示される送信データを読み出し，送信バッファ(TRNBUF)に設定します。ポインタはインクリメントします。

(vi) データ送信のサブルーチン（TRNDAT）を呼び出して，マスタにデータを送信します。

(vii) 転送エラーが発生した場合（ERRORF＝1）は，(ix)へ。

(viii) データ送信が正常に実行された場合(ERRORF＝0)は，カウンタをデクリメントし，指定した回数の転送が終了するまで(v)-(viii)を繰り返します。

(ix) データ送信のサブルーチン（TRNDAT）では，復帰時に送信モード（RCVFLG＝0）に設定されているため，受信モード（RCVFLG＝1）に設定し，戻ります。

(3) 使用レジスタ

マスタ	A, X, B, HL
スレーブ	A, X, B, HL

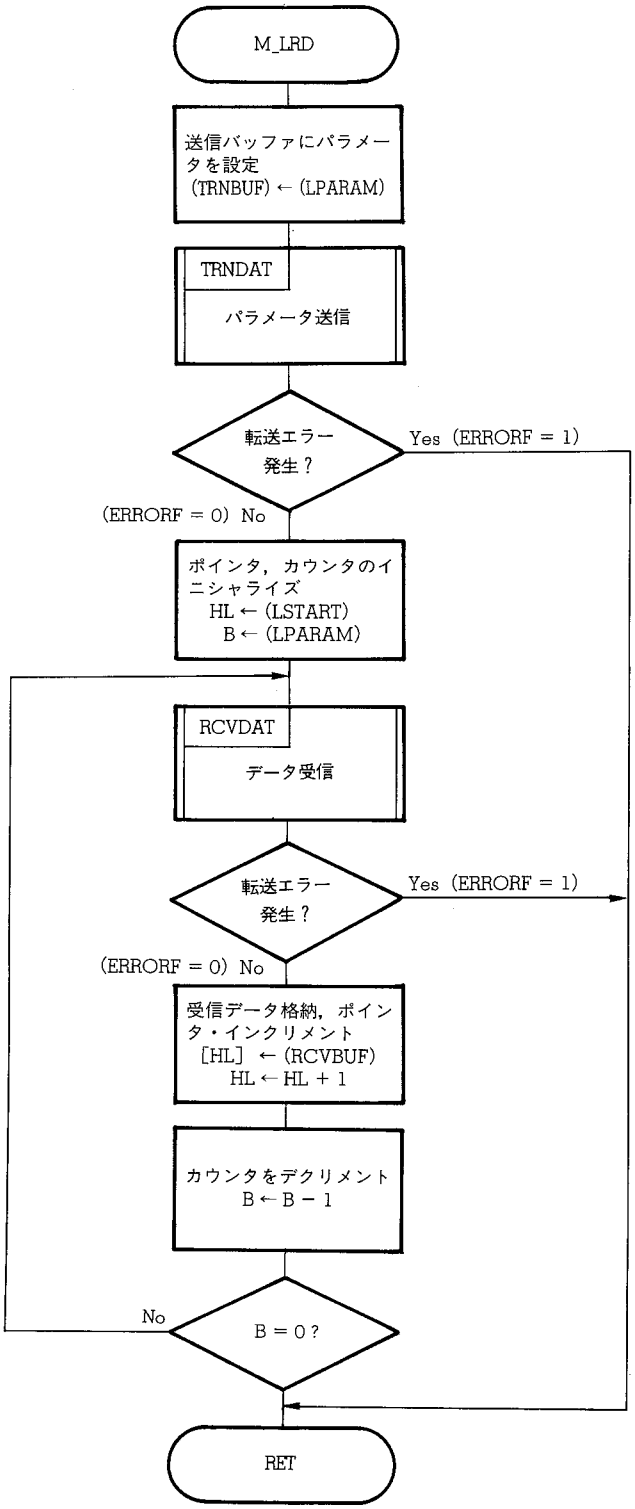
(4) 入出力条件

デバイス	入　力　条　件	出　力　条　件
マスタ	(LSTART) ←データ先頭アドレス (LPARAM) ←受信データ数	(LSTART), (LPARAM) : 保持 ① 正常転送　: ERRORF←0 (LSTART) 以降に受信データ ② 転送エラー : ERRORF←1
スレーブ	(LSTART) ←データ格納先頭アドレス (LPARAM) ←送信可能データ数	(LSTART) : 保持 (LPARAM) ←マスタへの送信データ数 RCVFLG←1 ① 正常転送　: ERRORF←0 ② 転送エラー : ERRORF←1

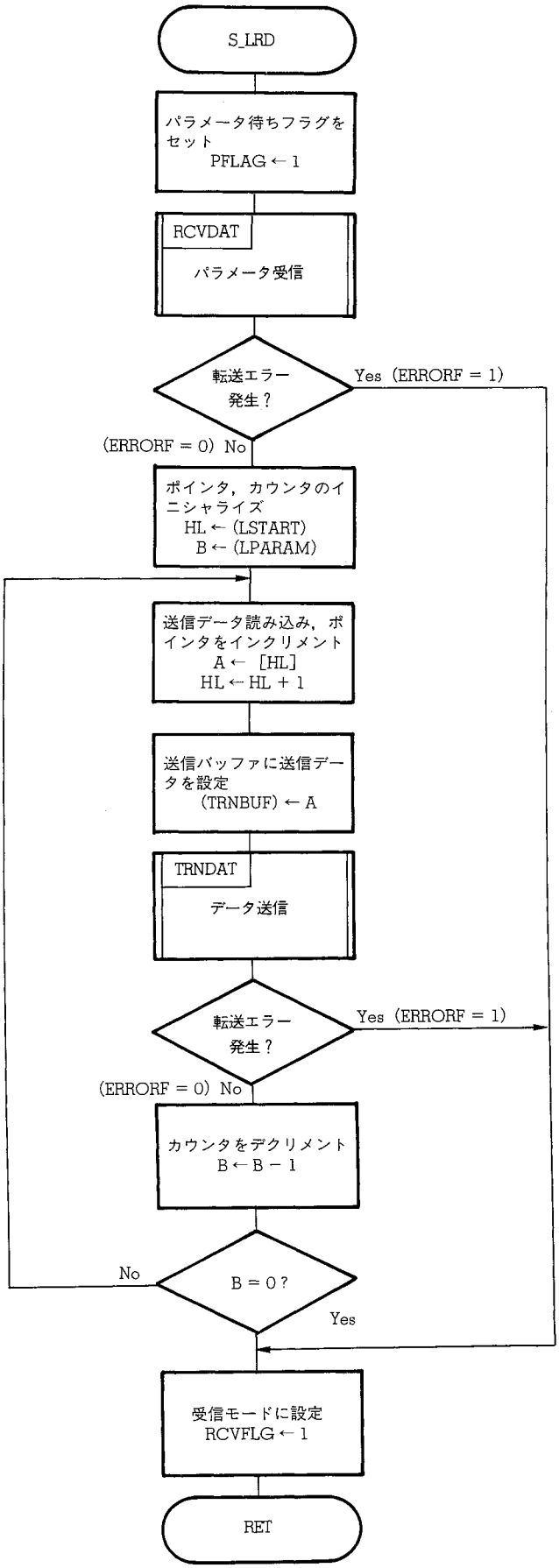
10
214
218A
224
234
244

(5) フロー・チャート

(a) マ ス タ



(b) スレーブ



10
214
218A
224
234
244

(6) プログラム・リスト

```
;-------------------------------------------------------
;       [[[  LREAD  ]]]
;               Slave -> Master  n byte Data
;               [CMD] A  [DAT] A  [DAT] A ···· [DAT] A
;                 M   S    M   S    S   M        S   M
;-------------------------------------------------------
;
;       ... MASTER ...
;
M_LRD:
        MOV     TRNBUF,LPARAM
        CALL    !TRNDAT
        BT      ERRORF,$MLR_J1

        MOVW    AX,LSTART               ; Pointer,Counter Set
        MOVW    HL,AX
        MOV     A,LPARAM
        MOV     B,A
MLR_L1:                                 ; Receive & Write Loop
        CALL    !RCVDAT
        BT      ERRORF,$MLR_J1
        MOV     A,RCVBUF
        MOV     [HL+],A
        DBNZ    B,$MLR_L1
MLR_J1:
        RET
;
;       ... SLAVE ...
;
S_LRD:
        SET1    PFLAG
        CALL    !RCVDAT
        BT      ERRORF,$SLR_J1

        MOVW    AX,LSTART               ; Pointer,Counter Set
        MOVW    HL,AX
        MOV     A,LPARAM
        MOV     B,A
SLR_L1:                                 ; Read & Trasmit Loop
        MOV     A,[HL+]
        MOV     TRNBUF,A
        CALL    !TRNDAT
        BT      ERRORF,$SLR_J1
        DBNZ    B,$SLR_L1
SLR_J1:
        SET1    RCVFLG
        RET
```

10.4.11　DETACHコマンド処理

現在選択されているスレーブを非選択状態にするコマンドです。

コマンド値	29H

(1) 動作概要

(a) マスタ側

現在選択されているデバイスを非選択状態とします。したがって，マスタのバックグランド処理としては，アドレス送信要求待ちの状態になります。

(b) スレーブ側

非選択状態に設定されます。したがって，スレーブのバックグランド処理としては，アドレス受信待ちの状態になります。

(c) 転送のフォーマット

M→S	S
DETACH	ACK
コマンド	

(2) プログラム説明 … (6) プログラム・リスト参照

(a) マスタDETACHコマンド処理［レーベル名称：M_DTC］

(i) マスタはDETACHコマンド送信後，バックグランド処理にて，アドレス入力待ち状態へ移行します。

(b) スレーブDETACHコマンド処理［レーベル名称：S_DTC］

(i) DETACHコマンド受信後，受信動作を禁止するために，INTCSI割り込みを禁止(マスク)します。

(ii) SIO←FFHによりビジィを解除します。

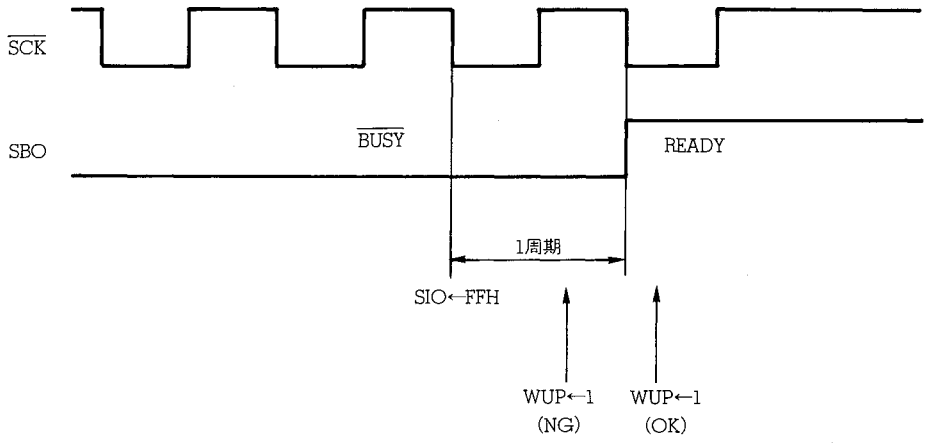
(iii) 一定時間ウエイトします。

(iv) WUP←1とし，非選択状態とします。

(v) INTCSI割り込みを許可します。

(vi) コマンド処理中フラグ（CMDBSY）をクリア(0)し，スレーブのアドレス受信待ち状態に移行します。

図10-15　ウエイク・アップ設定（WUP←1）タイミングの注意事項



(3) 使用レジスタ

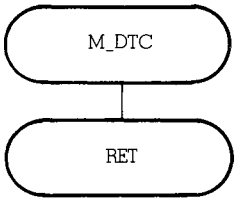
マスタ	なし
スレーブ	なし

(4) 入出力条件

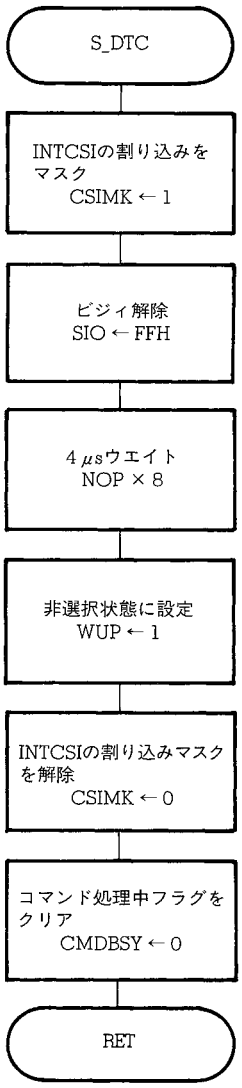
デバイス	入　力　条　件	出　力　条　件
マスタ	なし	なし
スレーブ	なし	ウエイク・アップ状態（WUP＝1） CMDBSY←0

(5) フロー・チャート

(a) マスタ



(b) スレーブ



10
214
218A
224
234
244

(6) プログラム・リスト

```
;-------------------------------------------------------
;       [[[  DETACH  ]]]
;               Master -> Slave Only Command
;               [CMD] A
;                 M   S
;-------------------------------------------------------

M_DTC:
        RET
S_DTC:
        SET1    CSIMK
        MOV     SIO,#0FFH               .; Clear Busy
        NOP                              ; WUP wait
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        SET1    WUP
        CLR1    CSIMK
        CLR1    CMDBSY
        RET
```

10.4.12　CHGMSTコマンド処理

現在選択されているスレーブにマスタの権利を受け渡すコマンドです。

コマンド値	28H

(1) 動作概要

(a) 動作

現在接続されている，マスタとスレーブ間でマスタ／スレーブの交換を行います。このコマンド実行前にマスタであったデバイスは，非選択状態のスレーブとなります。また，スレーブであったデバイスはスレーブを選択していないマスタとなります。

マスタ変更のタイミングは，データ転送終了後，データ・ラインのREADYを確認して変更します。

マスタに変更されたデバイスは，イニシャライズの終了後に，スレーブに変更されたデバイスのイニシャライズ完了待ちのために，ウエイトする必要があります。このプログラムでは，マスタ交換の可否までを行っており，バックグランド処理で，イニシャライズ処理を呼び出すことにより，イニシャライズ処理の中でウエイトしています。

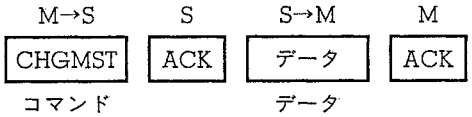
(b) マスタ変換の可否

マスタからCHGMSTコマンドを受信したスレーブは，DTFフラグのポーリングを行い，セットされたときの（WRDATA）の値をマスタに送信します。この送信データの値で，マスタ交換の可否を選択しています。

表10－12　マスタ交換の可否

スレーブ送信データ	マスタ交換の可否
FFH	可
FFH以外	不可

(c) 転送のフォーマット



(2) プログラム説明 … (6) プログラム・リスト参照

(a) マスタCHGMSTコマンド処理［レーベル名称：M_CMT］

(i) データ受信のサブルーチン（RCVDAT）を呼び出し，スレーブからデータを受信します。

(ii) 受信データがFFHならば，マスタ交換が可能であり，READYを確認後スレーブのイニシャライズ処理に移行します。

(iii) 受信データがFFH以外の場合は，マスタ交換不可であり，マスタのコマンド入力待ち状態に戻ります。

(b) スレーブCHGMSTコマンド処理［レーベル名称：S_CMT］

(i) スレーブの転送準備待ちとして，DTFフラグをポーリングします。DTF＝1ならばクリアします。ここで，送信データがFFHならば，マスタ交換が可能であり，データ送信のサブルーチン（TRNDAT）を呼び出し，マスタにデータ送信を行います。

送信データがFFHでない場合は，(iv)へ。

(ii) 転送エラーが発生した場合（ERRORF＝1）は，(v)へ。

(iii) データ送信が正常に実行された場合（ERRORF＝0）は，BSYE←0としてビジィ状態を解除し，マスタのイニシャライズ処理へ移行します。

(iv) 送信データがFFH以外の場合は，マスタ交換不可であり，データ送信のサブルーチン（TRNDAT）を呼び出し，マスタにデータ送信を行います。

(v) データ送信のサブルーチン（TRNDAT）では，復帰時に送信モード（RCVFLG＝0）に設定されているため，受信モード（RCVFLG＝1）に設定し，戻ります。

(vi) SIO←FFHによりビジィ状態を解除し，スレーブのコマンド受信待ちの状態に戻ります。

(3) 使用レジスタ

マスタ	A
スレーブ	A

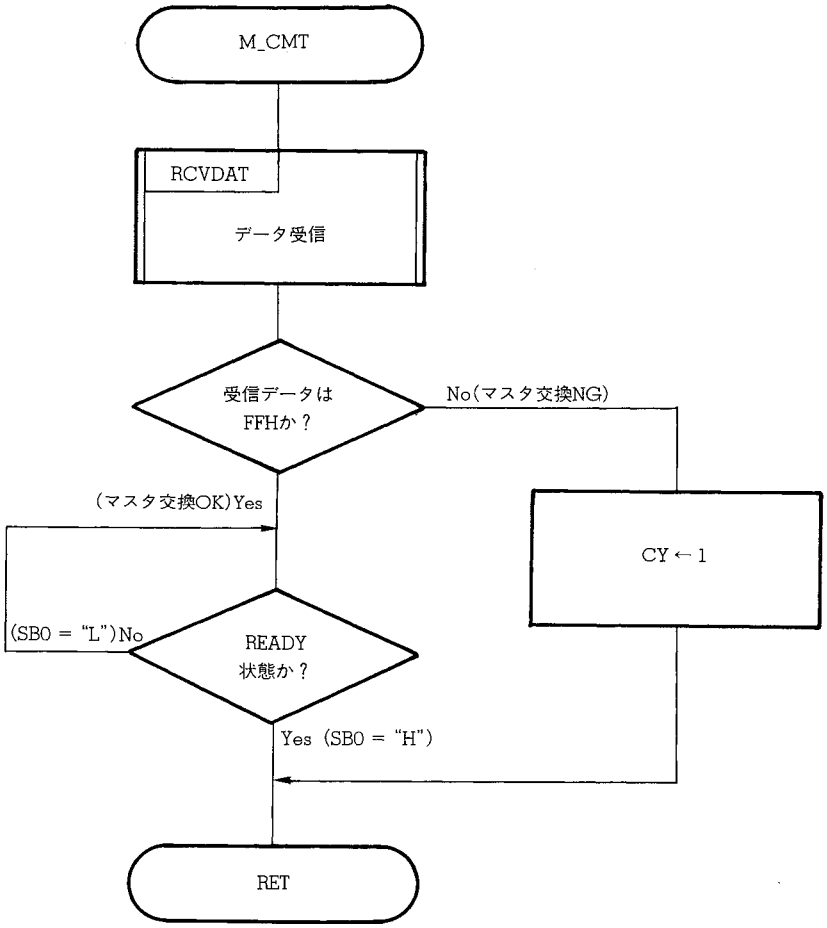
(4) 入出力条件

デバイス	入　力　条　件	出　力　条　件
マスタ	なし	マスタ・チェンジ可： (RCVBUF) ←FFH, CY←0 マスタ・チェンジ不可：CY＝1 (RCVBUF) ←FFH以外, CY←1
スレーブ	① DTF←1 ② マスタ・チェンジ可 ：(TRNBUF) ←FFH マスタ・チェンジ不可 ：(TRNBUF) ←FFH以外	マスタ・チェンジ可 ：CY←0 マスタ・チェンジ不可 ：CY←1, RCVFLG←1

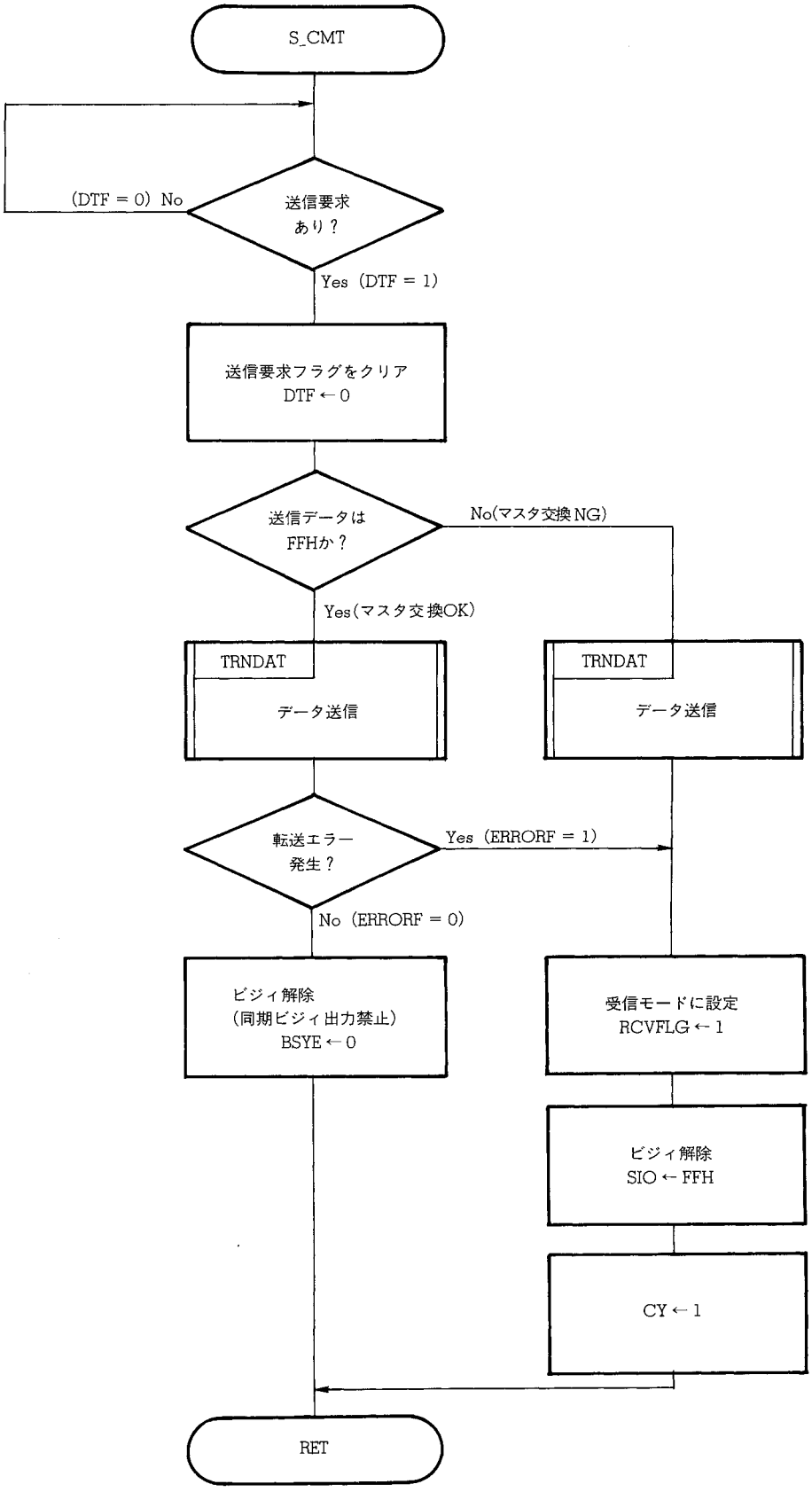
10
214
218A
224
234
244

(5) フロー・チャート

(a) マ ス タ



(b) スレーブ



10
214
218A
224
234
244

(6)プログラム・リスト

```
;-------------------------------------------------------
;       [[[  CHGMST  ]]]
;               Master -> Slave Only Command
;               [CMD] A
;                 M   S
;-------------------------------------------------------
;
;       ... MASTER ...
;
M_CMT:
        CALL    !RCVDAT
        CMP     RCVBUF,#0FFH            ; Check Master Change OK?
        BZ      $MCM_J1
        SET1    CY
        BR      MCM_J2
MCM_J1:
        BF      SBO,$MCM_J1
MCM_J2:
        RET
;
;       ... SLAVE ...
;
S_CMT:
        BF      DTF,$S_CMT
        CLR1    DTF
        CMP     TRNBUF,#0FFH            ; Check Master Change OK?
        BNZ     $SCM_J1
        CALL    !TRNDAT
        BT      ERRORF,$SCM_J2
        CLR1    BSYE
        BR      SCM_J3
SCM_J1:
        CALL    !TRNDAT
SCM_J2:
        SET1    RCVFLG
        MOV     SIO,#0FFH
        SET1    CY
SCM_J3:
        RET
```

10.4.13　MACWコマンド処理

78Kシリーズの特徴であるマクロ・サービスの機能を用いた，連続データのマスタ→スレーブ転送コマンドです。マクロ・サービスはタイプAを使用しています。タイプAでは，バッファ領域が内蔵RAMに確保されますので，使用しているワーク領域によって，一度に転送できるデータ数に限りがあります。

75XシリーズはMACWコマンドをLWRITEコマンドで代替します。

コマンド値	24H

(1) 動作概要

(a) マスタ側

マクロ・サービス・タイプAを用いて，RAM内のパラメータ（LPARAM）で指定されたバイト数分，RAM内の(LSTART)で指定されたアドレス以降のデータを送信します。ただし，1バイトのみの転送の場合には，マクロ・サービスを用いず，通常転送を行っています。

RAM上でのデータ構成はLWRITEコマンドと共通です。

ここで，1バイト目の送信は，1バイト送受信のサブルーチン（TRNDAT）を用いており，1バイト目の転送完了割り込みからマクロ・サービスでSIOへの書き込みを行います。したがって，チャネル・ポインタとマクロ・サービス・カウンタを表10－13のように設定しています。

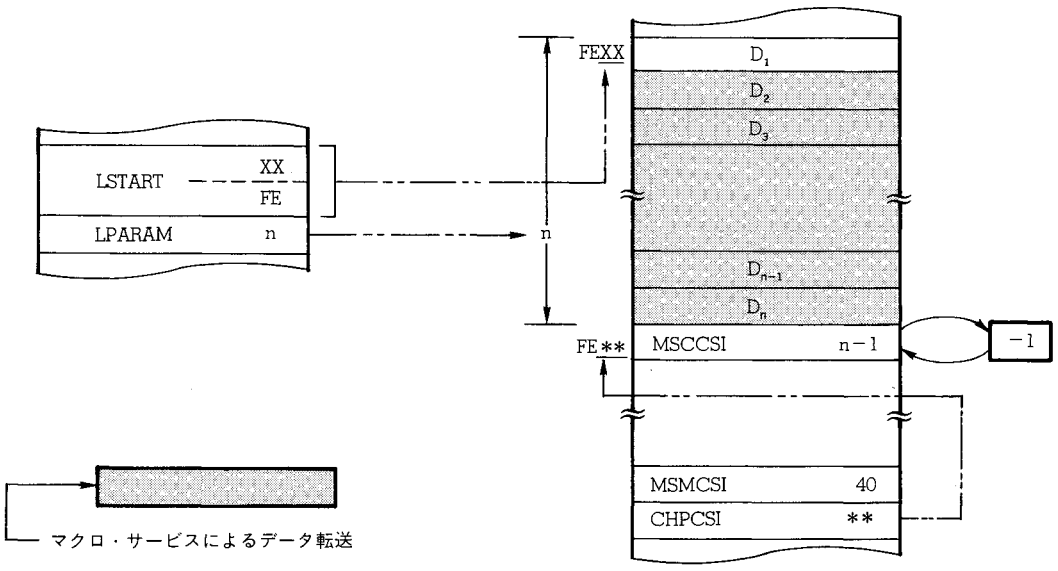
表10－13　MACWコマンドのチャネル・ポインタとマクロ・サービス・カウンタ（マスタ）

チャネル・ポインタ	マクロ・サービス・カウンタ
(LSTART)＋（LPARAM）の下位8ビット	(LPARAM)－1

また，マクロ・サービス・コントロール・ワードとマクロ・サービス・バッファの関係を図10－16に示します。

10
214
218A
224
234
244

図10－16　MACWコマンドのメモリ構成（マスタ）



(b) スレーブ側

マクロ・サービス・タイプAを用いて，マスタから指定されたバイト数分のデータ受信を行い，RAM内の(LSTART)で指定されたアドレス以降に受信データを格納します。ただし，1バイトのみの転送の場合には，マクロ・サービスを用いず，通常転送を行っています。

データ数のチェック方法はLWRITEコマンドと同じです。

注意　転送データ数((LPARAM))はコマンド処理を呼び出す前に設定してください。また（LPARAM）の値はコマンド処理中で破壊されます。

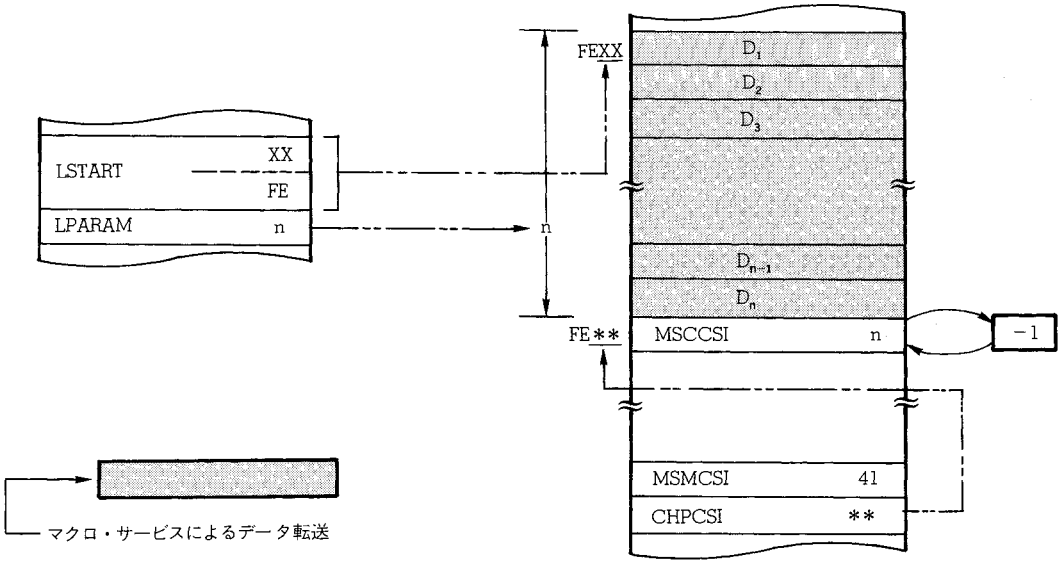
ここでマスタとは異なり，マクロ・サービスによる受信を行うために，受信のみ許可（CTXE＝0，CRXE＝1）に設定し，SIO読み出しでビジィ解除を行います。また，$\overline{\text{ACK}}$の自動出力を許可（ACKE←1）しています。1バイト目の受信は，1バイト送受信のサブルーチン(RCVMAC)を用いており，1バイト目の転送完了割り込みからマクロ・サービスでSIOからの読み出しを行います。したがって，チャネル・ポインタとマクロ・サービス・カウンタを表10－14のように設定しています。

表10－14　MACWコマンドのチャネル・ポインタとマクロ・サービス・カウンタ（スレーブ）

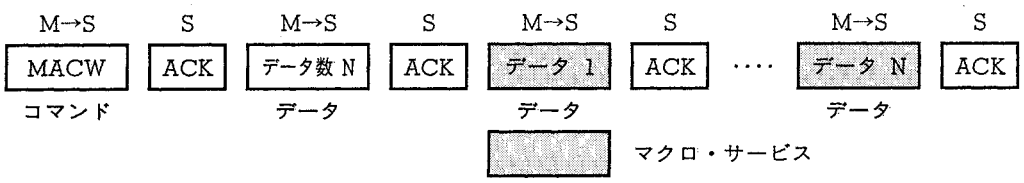
チャネル・ポインタ	マクロ・サービス・カウンタ
(LSTART)＋（LPARAM）の下位8ビット	(LPARAM)

また，マクロ・サービス・コントロール・ワードとマクロ・サービス・バッファの関係を図10－17に示します。

図10－17　MACWコマンドのメモリ構成（スレーブ）



（c）転送のフォーマット



(2) プログラム説明 … (6) プログラム・リスト参照

(a) マスタMACWコマンド処理［レーベル名称：M_MACW］

(i) 送信データ数をスレーブに送信します。転送エラーの場合はMACWコマンドを終了して戻ります。

(ii) 送信データ数が1バイトの場合は，(LSTART) で示されるメモリの内容を送信して戻ります。

(iii) マクロ・サービス・モード・レジスタに40H（タイプA，mem→sfr）を設定します。

(iv) チャネル・ポインタを設定します。

(v) マクロ・サービス・カウンタを設定します。

(vi) INTCSI割り込みサービス・モードをマクロ・サービスに設定します。

(vii) MACFフラグをセットし，1バイト目のデータを，TRNDATサブルーチンで送信して戻ります。

(b) スレーブMACWコマンド処理［レーベル名称：S_MACW］

(i) パラメータ待ちフラグ（PFLAG）をセットします。

(ii) データ受信のサブルーチン（RCVDAT）を呼び出し，マスタから受信データ数を受信します。

(iii) 転送エラーが発生した場合（ERRORF＝1）は，MACWコマンド処理を終了して戻ります。

(iv) 送信データ数が1バイトの場合は，1バイト受信を行い，受信データを (LSTART) で示されるメモリに格納して戻ります。

(v) マクロ・サービス・モード・レジスタに41H（タイプA，sfr→mem）を設定します。

(vi) チャネル・ポインタを設定します。

(vii) マクロ・サービス・カウンタを設定します。

(viii) INTCSI割り込みサービス・モードをマクロ・サービスに設定します。

(ix) $\overline{\text{ACK}}$の自動出力を許可します。

(x) 送信を禁止します（CTXE←0）。

(xi) MACFフラグをセットし，マクロ・サービスによる受信を起動します。

(xii) $\overline{\text{ACK}}$の自動出力を禁止して戻ります。

(3) 使用レジスタ

マスタ	A, X, DE, HL
スレーブ	A, X, HL

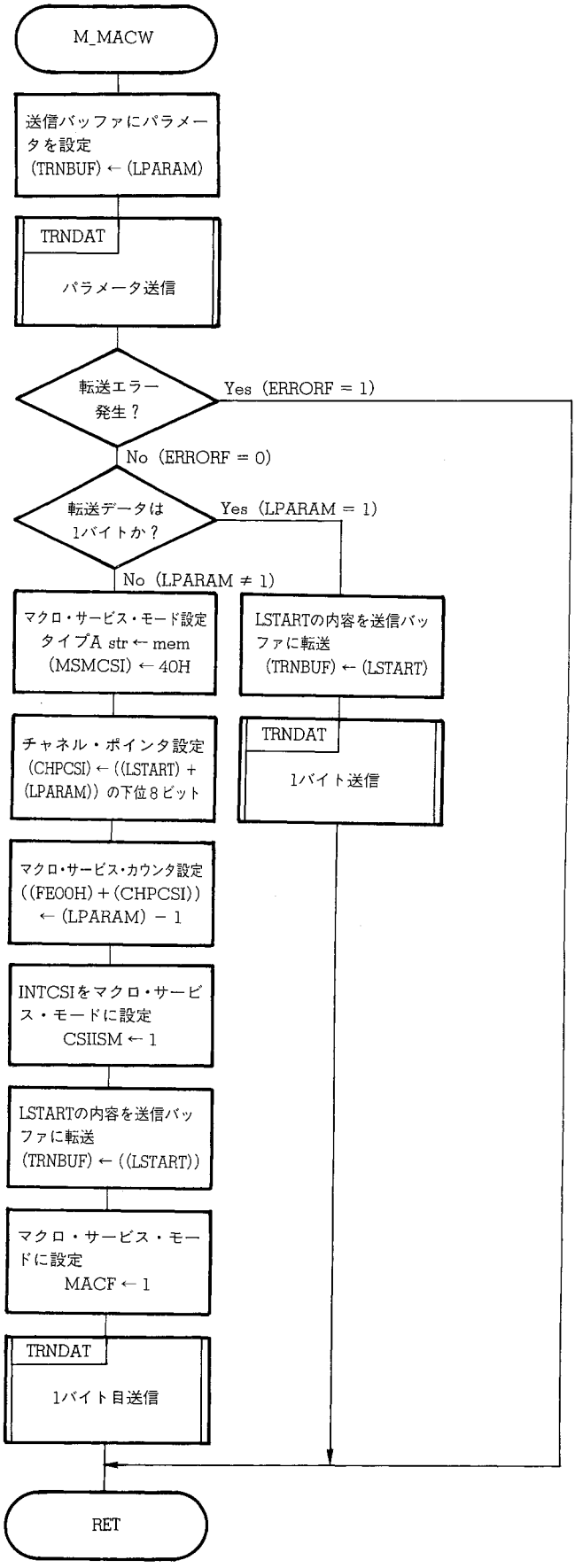
(4) 入出力条件

デバイス	入　力　条　件	出　力　条　件
マスタ	(LSTART) ←データ先頭アドレス (LPARAM) ←送信データ数	(LSTART), (LPARAM) : 保持 ①　正常転送　: ERRORF←0 ②　転送エラー : ERRORF←1
スレーブ	(LSTART) ←データ格納先頭アドレス (LPARAM) ←受信可能データ数	①　正常転送 : ERRORF←0 (LSTART) 以降に受信データ (LPARAM) ←マスタからの受信データ数 ②　転送エラー : ERRORF←1

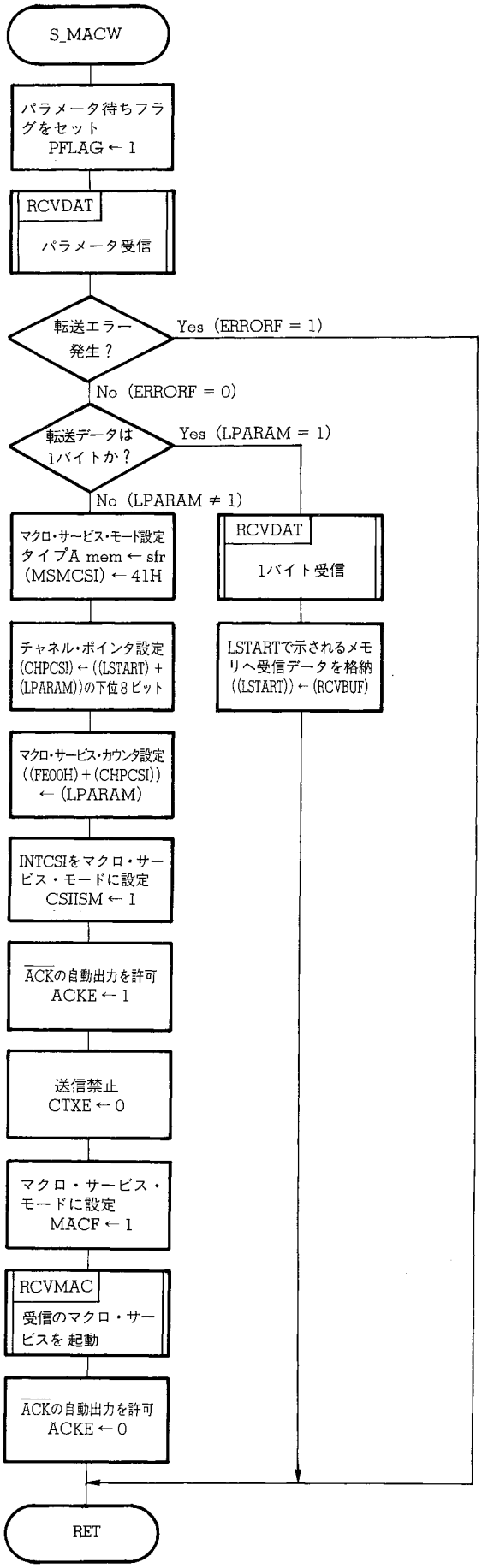
10
214
218A
224
234
244

(5) フロー・チャート

(a) マスタ



(b) スレーブ



10
214
218A
224
234
244

(6) プログラム・リスト

```
;-------------------------------------------------------
;       [[[  MACW  ]]]
;               Master -> Slave  n byte Data
;                       by Macro Service
;               [CMD] A  [DAT] A  [DAT] A ···· [DAT] A
;                 M   S    M   S    M   S        M   S
;-------------------------------------------------------
;
;       ... MASTER ...
;
M_MACW:
        MOV     TRNBUF,LPARAM
        CALL    !TRNDAT
        BT      ERRORF,$MMW_J2
        CMP     LPARAM,#1
        BE      $MMW_J1                 ; For Only 1 byte

        MOV     MSMCSI,#01000000B       ; type A , memory -> sfr

        MOVW    AX,LSTART
        MOVW    HL,AX
        MOVW    DE,AX

        MOV     A,LPARAM                ; Chanel Pointer
        ADD     A,L                     ;  (CHPCSI)=LOW((LSTART)+(LPARAM))
        MOV     CHPCSI,A

        MOV     L,A                     ; Macro Service Counter
        MOV     A,LPARAM                ;   MSCCSI =(LSTART)+(LPARAM)
        DEC     A                       ;  (MSCCSI)=(LPARAM)-1
        MOV     [HL],A

        SET1    CSIISM
        MOV     A,[DE]
        MOV     TRNBUF,A
        SET1    MACF
        CALL    !TRNDAT                 ; Macro Service Start
        BR      MMW_J2
MMW_J1:
        MOVW    AX,LSTART               ; For Only 1 byte
        MOVW    HL,AX                   ; No Macro Service
        MOV     A,[HL]
        MOV     TRNBUF,A
        CALL    !TRNDAT
MMW_J2:
        RET
```

```
;
;       ... SLAVE ...
;
S_MACW:
        SET1    PFLAG
        CALL    !RCVDAT
        BT      ERRORF,$SMW_J2
        CMP     LPARAM,#1
        BE      $SMW_J1                 ; For Only 1 byte

        MOV     MSMCSI,#01000001B       ; type A , sfr -> memory

        MOVW    AX,LSTART
        MOVW    HL,AX

        MOV     A,LPARAM                ; Chanel Pointer
        ADD     A,L                     ;  (CHPCSI)=LOW(LSTART)+(LPARAM))
        MOV     CHPCSI,A

        MOV     L,A                     ; Macro Service Counter
        MOV     A,LPARAM                ;   MSCCSI =(LSTART)+(LPARAM)
        MOV     [HL],A                  ;  (MSCCSI)=(LPARAM)

        SET1    CSIISM
        SET1    ACKE
        CLR1    CTXE
        SET1    MACF
        CALL    !RCVMAC                 ; Macro Service Start
        CLR1    ACKE
        BR      SMW_J2
SMW_J1:
        MOVW    AX,LSTART               ; For Only 1 byte
        MOVW    HL,AX                   ; No Macro Service
        CALL    !RCVDAT
        MOV     A,RCVBUF
        MOV     [HL],A
SMW_J2:
        RET
```

10
214
218A
224
234
244

10.4.14　MACRコマンド処理

78Kシリーズの特徴であるマクロ・サービスの機能を用いた，連続データのスレーブ→マスタ転送コマンドです。連続データのマスタ→スレーブ転送コマンドです。マクロ・サービスはタイプAを使用しています。タイプAでは，バッファ領域が内蔵RAMに確保されますので，使用しているワーク領域によって，一度に転送できるデータ数に限りがあります。

75XシリーズはMACRコマンドをLREADコマンドで代替します。

コマンド値	25H

(1) 動作概要

(a) マスタ側

マクロ・サービス・タイプAを用いて，RAM内のパラメータ（LPARAM）で指定されたバイト数分，RAM内の（LSTART）で指定されたアドレス以降に受信データを格納します。ただし，1バイトのみの転送の場合には，マクロ・サービスを用いず，通常転送を行っています。

RAM上でのデータ構成はLWRITEコマンドと共通です。

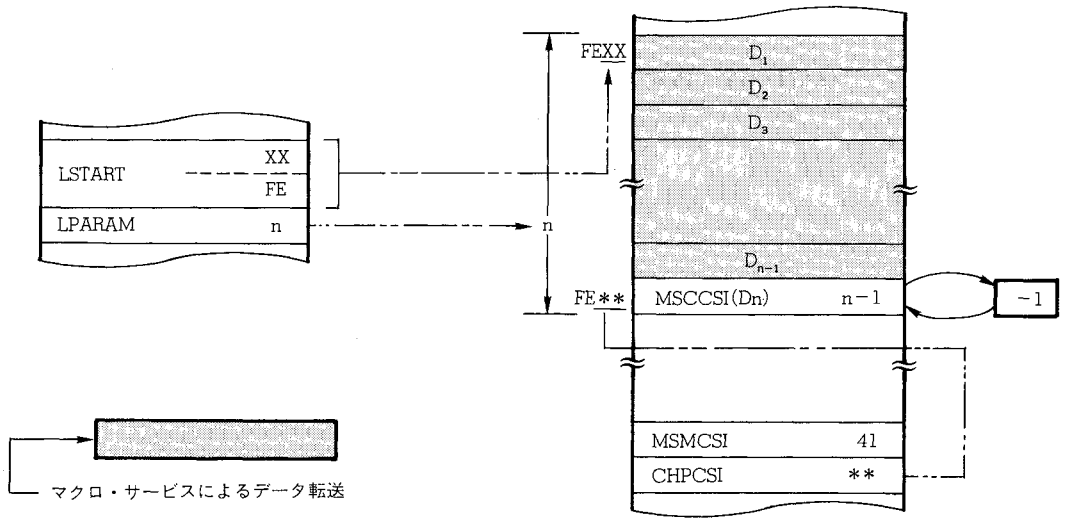
マクロ・サービスによる受信を行うために，受信のみ許可（CTXE=0，CRXE=1）に設定し，また，$\overline{\text{ACK}}$の自動出力を許可(ACKE←1)しています。ここで，最後のデータ受信は，INTCSIベクタ割り込みを用いてメモリに格納しています。したがって，チャネル・ポインタとマクロ・サービス・カウンタを表10－15のように設定しています。

表10－15　MACRコマンドのチャネル・ポインタとマクロ・サービス・カウンタ（マスタ）

チャネル・ポインタ	マクロ・サービス・カウンタ
(LSTART)＋（LPARAM)－1の下位8ビット	(LPARAM)－1

また，マクロ・サービス・コントロール・ワードとマクロ・サービス・バッファの関係を図10－18に示します。

図10－18　MACRコマンドのメモリ構成（マスタ）



(b) スレーブ側

マクロ・サービス・タイプAを用いて，RAM内の(LSTART)で指定されたアドレス以降のデータをマスタから指定されたバイト数分マスタへ送信します。ただし，1バイトのみの転送の場合には，マクロ・サービスを用いず，通常転送を行っています。

データ数のチェック方法はLWRITEコマンドと同じです。

注意　転送データ数((LPARAM))はコマンド処理を呼び出す前に設定してください。また（LPARAM）の値はコマンド処理中で破壊されます。

ここで，1バイト目の送信は，1バイト送受信のサブルーチン（TRNDAT）を用いており，1バイト目の転送完了割り込みからマクロ・サービスでSIOへの書き込みを行います。したがって，チャネル・ポインタとマクロ・サービス・カウンタを表10－16のように設定しています。

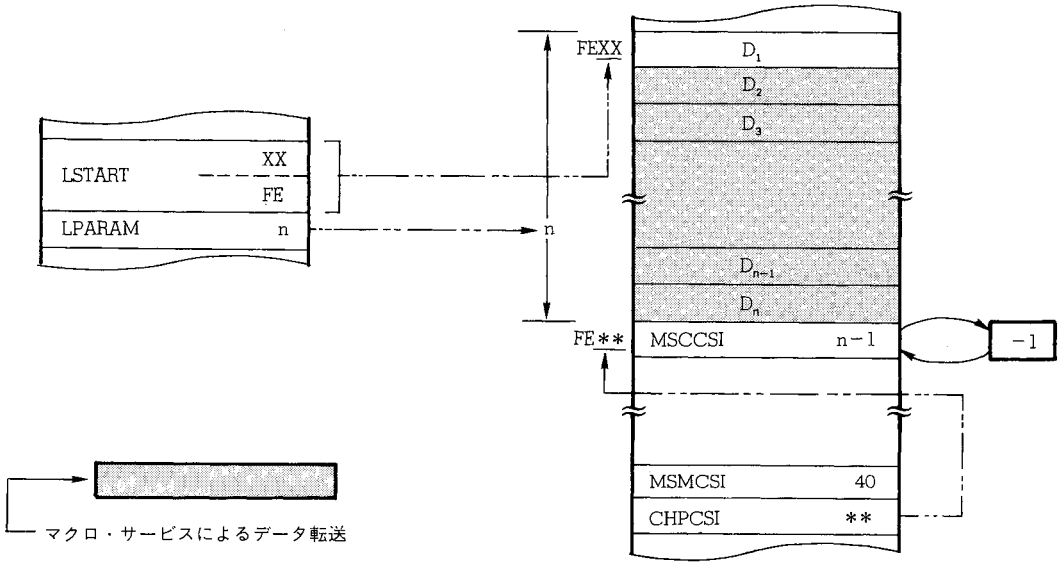
表10－16　MACRコマンドのチャネル・ポインタとマクロ・サービス・カウンタ（スレーブ）

チャネル・ポインタ	マクロ・サービス・カウンタ
(LSTART)＋（LPARAM）の下位8ビット	(LPARAM)－1

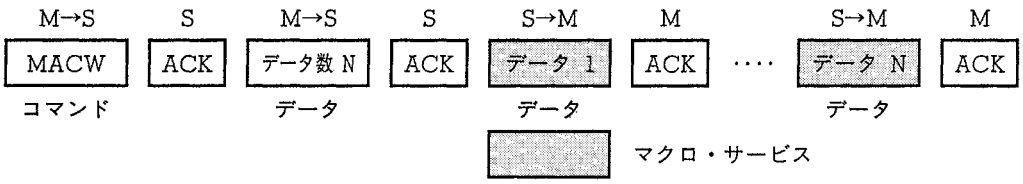
10
214
218A
224
234
244

また，マクロ・サービス・コントロール・ワードとマクロ・サービス・バッファの関係を図10－19に示します。

図10－19　MACRコマンドのメモリ構成（スレーブ）



（c）転送のフォーマット



(2) プログラム説明 … (6) プログラム・リスト参照

(a) マスタMACRコマンド処理［レーベル名称：M_MACR］

(i) 送信データ数をスレーブに送信します。転送エラーの場合はMACRコマンドを終了して戻ります。

(ii) 送信データ数が1バイトの場合は，1バイト受信を行い，受信データを（LSTART）で示されるメモリに格納して戻ります。

(iii) マクロ・サービス・モード・レジスタに41H（タイプA，sfr→mem）を設定します。

(iv) チャネル・ポインタを設定します。

(v) マクロ・サービス・カウンタを設定します。

(vi) INTCSI割り込みサービス・モードをマクロ・サービスに設定します。

(vii) $\overline{\text{ACK}}$の自動出力を許可します。

(viii) 送信を禁止します（CTXE→0）。

(ix) MACFフラグをセットし，マクロ・サービスによる受信を起動します。

(x) 最後の受信データを（LSTART）＋（LPARAM）で示されるメモリに格納します。

(xi) $\overline{\text{ACK}}$の自動出力を禁止し，戻ります。

(b) スレーブMACRコマンド処理［レーベル名称：S_MACR］

(i) パラメータ待ちフラグ（PFLAG）をセットします。

(ii) データ受信のサブルーチン（RCVDAT）を呼び出し，マスタから受信データ数を受信します。

(iii) 転送エラーが発生した場合（ERRORF＝1）は，MACRコマンド処理を終了し(x)へ。

(iv) 送信データ数が1バイトの場合は，（LSTART）で示されるメモリの内容を送信後(x)へ。

(v) マクロ・サービス・モード・レジスタに40H（タイプA，mem→sfr）を設定します。

(vi) チャネル・ポインタを設定します。

(vii) マクロ・サービス・カウンタを設定します。

(viii) INTCSI割り込みサービス・モードをマクロ・サービスに設定します。

(ix) MACFフラグをセットし，1バイト目のデータを，TRNDATサブルーチンで送信します。

(x) 受信モードに設定（RCVFLG←1）して戻ります。

10
214
218A
224
234
244

(3) 使用レジスタ

マスタ	A, X, HL
スレーブ	A, X, DE, HL

(4) 入出力条件

デバイス	入　力　条　件	出　力　条　件
マスタ	(LSTART) ←データ格納先頭アドレス (LPARAM) ←受信データ数	(LSTART), (LPARAM) : 保持 ① 正常転送　: ERRORF←0 (LSTART) 以降に受信データ ② 転送エラー : ERRORF←1
スレーブ	(LSTART) ←データ先頭アドレス (LPARAM) ←送信可能データ数	(LSTART) : 保持 (LPARAM) ←マスタへの送信データ数 RCVFLG←1 ① 正常転送　: ERRORF←0 ② 転送エラー : ERRORF←1

〔メ　モ〕

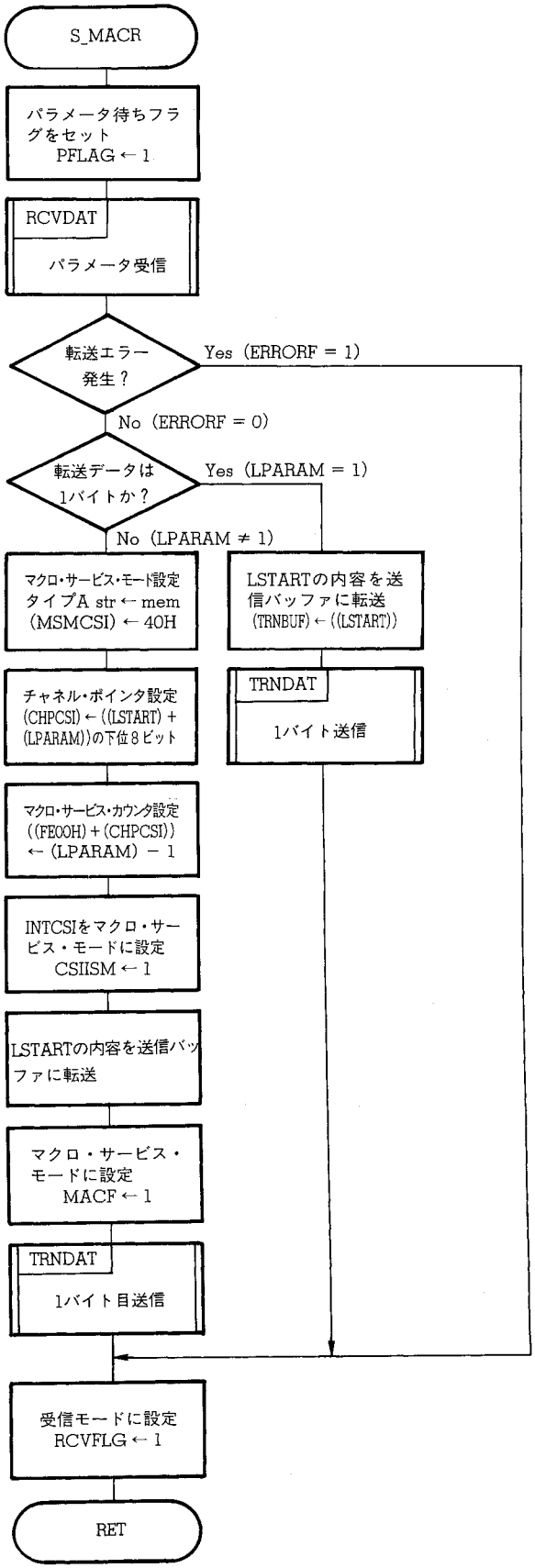
214
218A
224
234
244

(5) フロー・チャート

(a) マスタ



(b) スレーブ



10
214
218A
224
234
244

(6) プログラム・リスト

```
;-------------------------------------------------------
;       [[[  MACR  ]]]
;               Slave -> Master  n byte Data
;                       by Macro Service
;               [CMD] A  [DAT] A  [DAT] A ···· [DAT] A
;                 M   S    M   S    S   M        S   M
;-------------------------------------------------------
;
;       ... MASTER ...
;
M_MACR:
        MOV     TRNBUF,LPARAM
        CALL    !TRNDAT
        BT      ERRORF,$MMR_J2
        CMP     LPARAM,#1
        BE      $MMR_J1                 ; For Only 1 byte

        MOV     MSMCSI,#01000001B       ; type A , sfr -> memory

        MOVW    AX,LSTART
        MOVW    HL,AX

        MOV     A,LPARAM                ; Chanel Pointer
        ADD     A,L                     ;  (CHPCSI)=LOW(LSTART)+(LPARAM)-1)
        DEC     A
        MOV     CHPCSI,A

        MOV     L,A                     ; Macro Service Counter
        MOV     A,LPARAM                ;   MSCCSI =(LSTART)+(LPARAM)
        DEC     A                       ;  (MSCCSI)=(LPARAM-1)
        MOV     [HL],A

        SET1    CSIISM
        SET1    ACKE
        CLR1    CTXE
        SET1    MACF
        CALL    !RCVMAC                 ; Macro Service Start
        CLR1    ACKE
        MOV     A,RCVBUF
        MOV     [HL],A
        BR      MMR_J2
MMR_J1:
        MOVW    AX,LSTART               ; For Only 1 byte
        MOVW    HL,AX                   ; No Macro Service
        CALL    !RCVDAT
        MOV     A,RCVBUF
        MOV     [HL],A
MMR_J2:
        RET
```

```
;
;       ... SLAVE ...
;
S_MACR:
        SET1    PFLAG
        CALL    !RCVDAT
        BT      ERRORF,$SMR_J2
        CMP     LPARAM,#1
        BE      $SMR_J1                 ; For Only 1 byte

        MOV     MSMCSI,#01000000B       ; type A , memory -> sfr

        MOVW    AX,LSTART
        MOVW    HL,AX
        MOVW    DE,AX

        MOV     A,LPARAM                ; Chanel Pointer
        ADD     A,L                     ;  (CHPCSI)=LOW((LSTART)+(LPARAM))
        MOV     CHPCSI,A

        MOV     L,A                     ; Macro Service Counter
        MOV     A,LPARAM                ;   MSCCSI =(LSTART)+(LPARAM)
        DEC     A                       ;  (MSCCSI)=(LPARAM)-1
        MOV     [HL],A

        SET1    CSIISM
        MOV     A,[DE]
        MOV     TRNBUF,A
        SET1    MACF
        CALL    !TRNDAT                 ; Macro Service Start
        BR      SMR_J2
SMR_J1:
        MOVW    AX,LSTART               ; For Only 1 byte
        MOVW    HL,AX                   ; No Macro Service
        MOV     A,[HL]
        MOV     TRNBUF,A
        CALL    !TRNDAT
SMR_J2:
        SET1    RCVFLG
        RET
```

10
214
218A
224
234
244

10.4.15　DSPON/DSPOFFコマンド処理

スレーブに対する，1ビットのフラグ操作指令などは，コマンドのみで行うことができます。ここでは応用例として，スレーブ側のLEDのオン，オフを行う例を示しました。

コマンド値	31H

(1) 動作概要

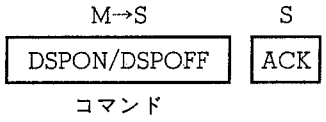
(a) STLEDFの管理

スレーブでは，LEDのON/OFFをSTLEDFフラグで管理しています。このコマンド処理では，STLEDFのセット／クリアを行います。

表10－17　スタティックLEDの表示状態

STLEDF	スタティックLED表示状態
0	ON
1	OFF

(b) 転送のフォーマット



(2) プログラム説明

(a) マスタDSPON/DSPOFFコマンド処理［レーベル名称：M_ON，M_OFF］

(i) マスタはDSPON/DSPOFFコマンド送信後，バックグランド処理にて，コマンド入力待ち状態へ移行します。

(b) スレーブDSPON/DSPOFFコマンド処理［レーベル名称：S_ON，S_OFF］

(i) DSPONコマンドならば，STLEDF←0，DSPOFFコマンドならば，STLEDF←1とし，戻ります。

(3) 使用レジスタ

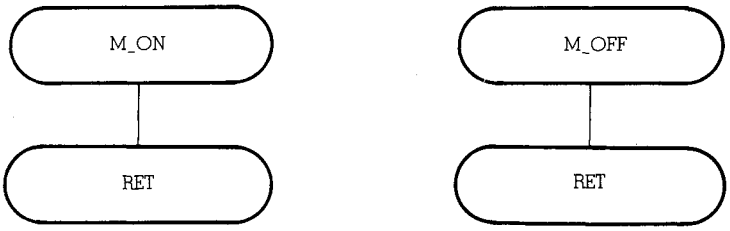
なし

(4) 入出力条件

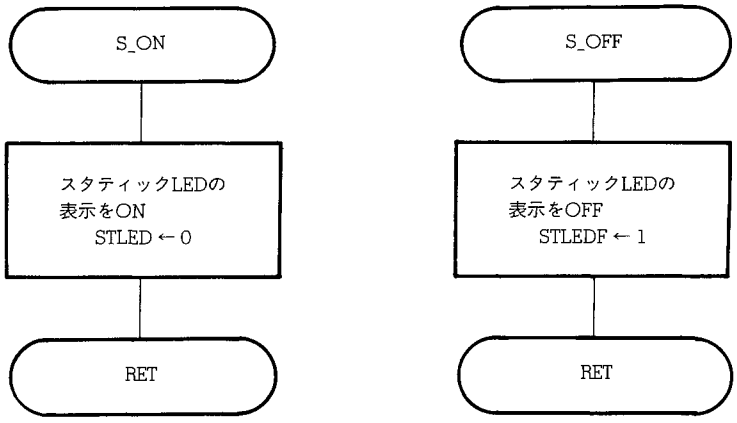
デバイス	入　力　条　件	出　力　条　件
マスタ	なし	なし
スレーブ	なし	STLEDFフラグの変化

(5) フロー・チャート

(a) マスタ



(b) スレーブ



10
214
218A
224
234
244

(6) プログラム・リスト

```
;-------------------------------------------------------
;       [[[  DSPON/DSPOFF  ]]]
;               Master -> Slave Only Command
;               [CMD] A
;                 M   S
;-------------------------------------------------------

M_ON:
        RET
S_ON:
        CLR1    STLEDF
        RET
M_OFF:
        RET
S_OFF:
        SET1    STLEDF
        RET
```

10.5　SBI使用上の注意事項

(1) IE-78210-RとIE-78220-Rでは，スレーブで不一致アドレスを受信した場合，いったん送受信を禁止状態(CTXE←0, CRXE←0)にし，以後の受信のために再び送受信許可の状態(CTXE←1, CRXE←1)に設定してください。

不一致アドレス受信後，送受信許可のままで使用すると以後の受信でINTCSI割り込み要求が発生しません。

(2) マクロ・サービスを使用する場合，最後のSIOとのマクロ・サービス転送を行ったあとに発生する，マクロ・サービス完了割り込みと，シリアル転送完了により発生する割り込みが，転送レートや他の割り込み処理などとの関係で識別できなくなる可能性があります（同一の割り込み要求フラグを使用しているため)。したがって，ソフトウエアなどによる何らかの方法で回避してください。

第11章　ホームバス・システムへの応用

11.1　概　　要

ホームバス・システムは，家庭内の家電機器，AV（Audio Visual）機器，セキュリティ関連機器や，社会のさまざまな機器を共通の伝送路に接続して，家庭内外から，これらの情報をアクセスできるシステムです。

現在，通信規格として，ET-2101（日本電子機械工業会（EIAJ））が定められています。

この応用例では，ホームバス・プロトコル・コントローラ（以降HBPCと略す）として，フィジカル・レイヤ，データリンク・レイヤ，ネットワーク・レイヤ（各プロトコルについては**11.2　通信プロトコル**参照）と呼ばれる部分の制御を，ET-2101の制御チャネル・プロトコルに準拠して行います。

ターゲットCPUは，内部ROM　32Kバイト，内部RAM　1KバイトのμPD78218Aとしています。なお，このプログラム例は，高速内部ROMフェッチ・モードでのみ動作します。

11.2　通信プロトコル

この応用例に関係あるフィジカル・レイヤ，データリンク・レイヤ，ネットワーク・レイヤのプロトコル項目を次に示します。

表11－1　ホームバス・システムのプロトコル項目

レイヤ構成	制御チャネル・プロトコル
フィジカル・レイヤ	電気特性 　制御信号の伝送速度 　伝送方式／伝送波形 論理条件 　制御方式 　同期方式 　制御信号のフォーマット 　パケット優先 　競合でのビット照合 　衝突検出 　短電文割り込み
データリンク・レイヤ	誤り検出 誤り制御 電文長コード 制御コード 同報／個別種別 アドレス・コード割り当て
ネットワーク・レイヤ	ヘッダ 共通コマンド（一部） 通信シーケンス（一部）

11.2.1　制御信号と制御方式

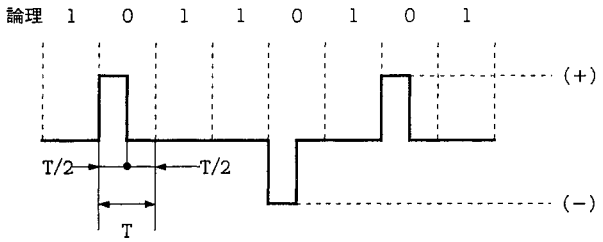
(1) 制御信号

伝送速度：9600 bps±0.13 %

伝送波形：AMI（Alternate Mark Inversion）方式，デューティ比50 %

論理　　：負論理

図11－1　制御信号の基本構成

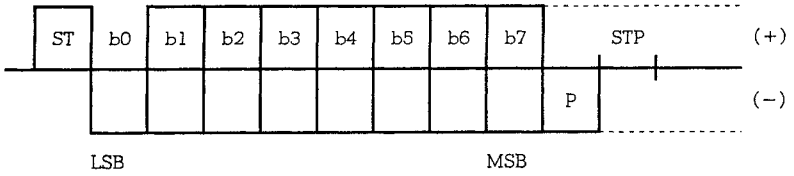


(2) 制御方式

同期方式　　：調歩同期

キャラクタ構成：11ビット（スタート・ビット1，データ8，パリティ1，ストップ1）

図11－2　制御信号の基本構成



スタート・ビット送出：（+）側

データ送出　　：LSBファースト（負論理）

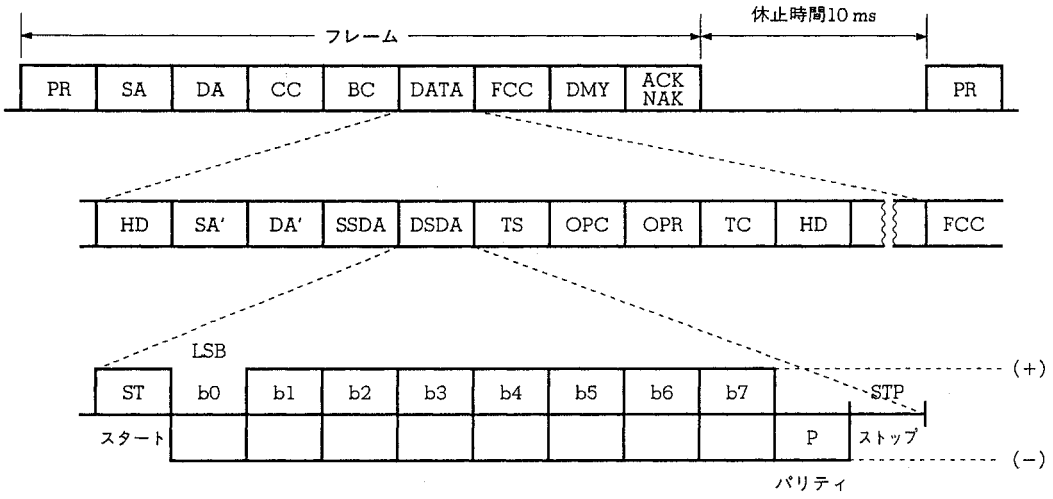
パリティ　　：偶数パリティ

キャラクタ間隔　　：ストップ・ビットと次キャラクタとの間隔をあけない

11
214
218A
234
244

(3) 制御信号の基本フォーマット

図11－3　制御信号のフォーマット



PR　　：優先コード
SA　　：自己アドレス
DA　　：相手アドレス
CC　　：制御コード
BC　　：電文長コード
DATA　：データ，オペレーション・コード，オペランドなど
FCC　　：チェック・コード
DMY　　：ダミー
ACK/NAK：ACK/NAKコード

DATA
- HD　：ヘッダ・コード
- SA'　：自己サブアドレス
- DA'　：相手サブアドレス
- SSDA：自己サブデバイス・アドレス
- DSDA：相手サブデバイス・アドレス
- TS　：テーブル・セレクタ
- OPC　：オペレーション・コード
- OPR　：オペランド
- TC　：ターミネーション・コード［1DH］

11.2.2　休止時間，休止期間

休止時間，休止期間は，フレーム（1フレームは，優先コードからACK/NAKまでのデータ）とフレームの間に設ける時間で，この期間，データ伝送は行いません（禁止）。

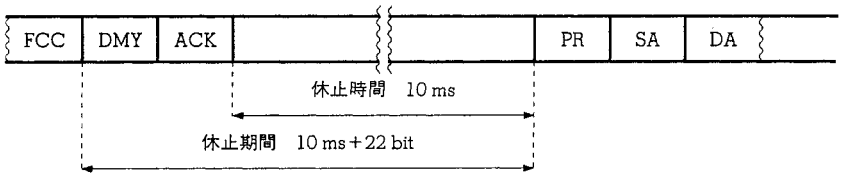
次の場合，送受信終了後，休止時間（10 ms）に入ります。

- 同報でない電文の送受信が終了
- 短電文割り込み，被短電文割り込みのない電文の送受信が終了

次の場合，送受信終了後，休止期間（10 ms＋22 bit）に入ります。

- 同報の電文の送受信が終了
- 短電文割り込み，被短電文割り込みが入った

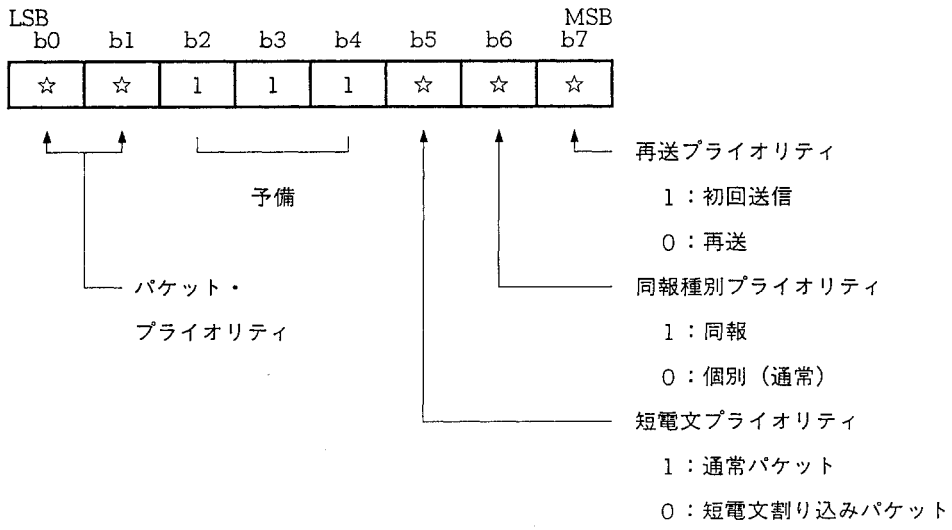
図11－4　休止時間と休止期間のタイミング



11.2.3　パケット優先

同時に起きる電文送信の競合制御を行うために，パケット（1パケットは，スタート・ビットからストップ・ビットまでの11ビット構成）の優先順位を優先コード（PR）によって決定します。

図11－5　優先コードのビット割り付け



11
214
218A
234
244

(1) パケット・プライオリティ

表11－2　パケット・プライオリティ

優先レベル	b0b1	内　　容	電　文	備　　考
1	00	システム特権パケット	短	初期化，復旧時など
2	01	回線交換パケット	短	回線接続のためのシーケンス
3	10	短電文通常パケット	短	セキュリティ，負荷制御，一般
4	11	長電文通常パケット	長	パソコンなどのデータ

(2) 短電文プライオリティ

短電文割り込み(**11.2.6　短電文割り込み**参照)のためにブレーク信号を送出した端末は，休止期間後に送出する短電文パケットにおいて，このビットを０として通常フレームより優先させることができます。

(3) 同報種別プライオリティ

個別電文フレームは，同報電文フレームより優先します。

(4) 再送プライオリティ

フレームの送信を行った端末は，ACK/NAKコードの受信によって伝送誤りを検出した場合，再送が可能です。この際，このビットを０とすることによって，初回送信のフレームより優先させることができます。

11.2.4　競合でのビット照合と衝突検出

(1) 競合時の動作

複数の端末から送信されたパケットの競合時，優先コードと自己アドレス・コード部で，ビットごとに送信データと受信データを照合することによって，勝ち残りパケットを決定します。

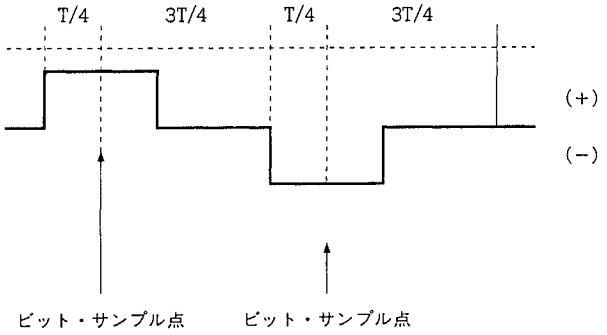
(2) 衝突検出

ビット単位で送信データと受信データの不一致を検出した時を，衝突検出（競合負け）とみなします。この場合，ただちに送信を中止し，受信に移ります。中断したデータの送信は送信可能になってから再開します（再送回数は保持）。

〈衝突検出点〉

各ビットの開始時刻から1/4ビット時刻（1/9600×1/4＝26 μs）経過点

図11－6　ビット・サンプル点



備考　受信データの取り込みも，このビット・サンプル点で行います。

11
214
218A
234
244

11.2.5　同期回復

(1) 同期回復時間

休止時間の終了２ビット前からバスを監視し，複数IFU間の同期を回復します。

同期回復監視時間（TF）：２ビット相当時間（1/9600×2＝208 μs）

(2) 同期回復後の動作

送信：TF時間に他の端末が送信を開始した場合は，これに同期して送信を開始します。

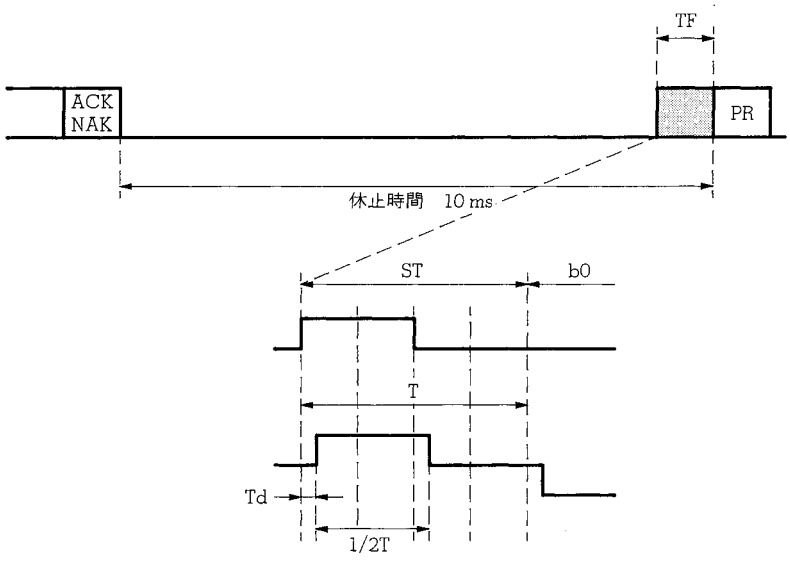
TF時間に他の端末が送信を開始しない場合は，休止時間（10 ms）後に送信します。

受信：休止時間経過後，受信可能な状態に入ります。

(3) 送信遅れ許容時間

同期回復監視時間内に他の端末が送信を開始してから，これに同期して送信を行う場合，1/8ビット時刻（1/9600×1/8＝13 μs）以内に送信を開始しなければなりません。

図11－7　同期回復のタイミング



Td＝13 μs以内

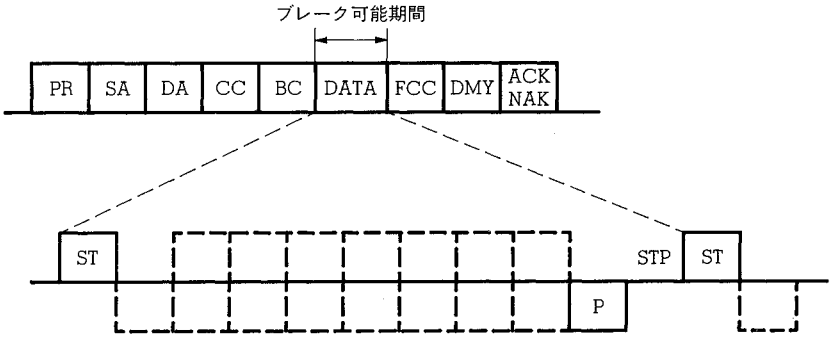
11.2.6　短電文割り込み

長電文フレーム通信中に短電文フレームの送信要求があった場合，長電文のデータ・フィールド中に割り込みをかけることができます。

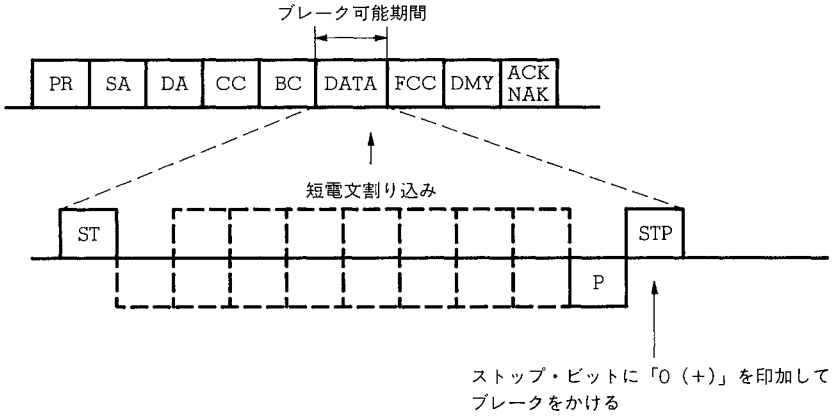
(1) ブレーク信号

図11－8　ブレーク信号

〈短電文割り込みがない場合のバス上の信号〉



〈短電文割り込みがある場合のバス上の信号〉



(2) ブレーク後の処理

休止期間（10 ms＋22 bit）に入ります。休止期間後，送信を中断した電文と割り込み電文の送信が行われますが，競合の結果，割り込み電文が勝ち残ります。

11
214
218A
234
244

11.2.7　誤り検出

(1) 誤り検出の方法

次の方法でフレーム伝送上でのデータ化けやデータ落ちによる伝送エラーの検出を行います。

- ●パリティ・ビット　：偶数パリティ…パケット単位
- ●チェック・コード　：チェック・サム…フレーム単位

上記方法によるエラー・チェックの結果をACK/NAKコードで示します。

(2) ACK/NAKコード

ACK/NAKコードは，受信端末が送信端末に返送します。

表11-3　ACK/NAKとエラー

種別	コード	結　果
ACK	06H	エラーなし
NAK	15H ACK/NAK以外	エラーあり
無応答	—	エラーあり

(3) エラー再送

送信結果がエラーの場合，送信端末は，休止時間後にフレームの再送を行います。

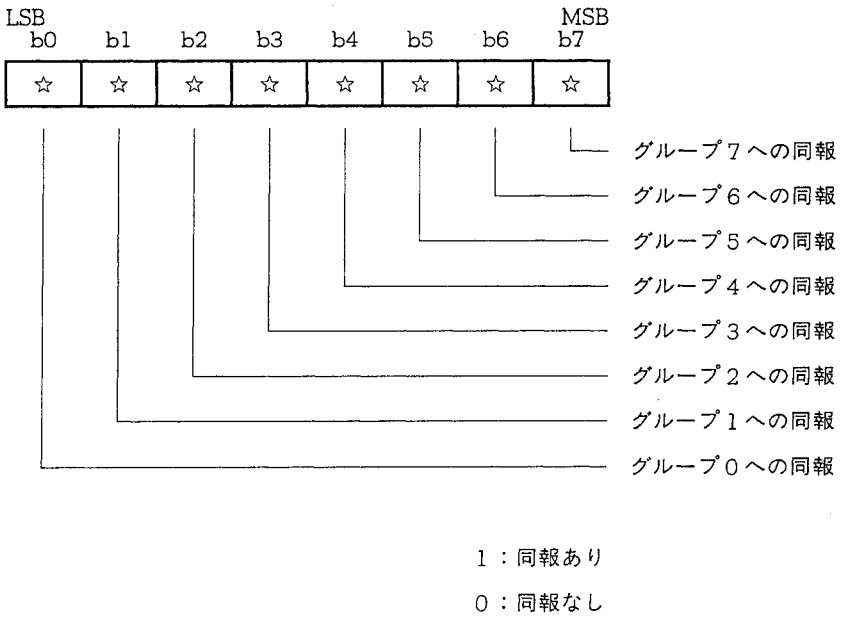
再送は3回まで可能で，再送回数は制御コードの再送フレーム・ナンバ（b6，b7）で示します（**11.2.9　制御コード**参照）。

11.2.8 同　　報

複数の端末に対して同時に電文の送信を行うことで，一斉同報とグループ同報があります。同報であることは，優先コードのb6＝1で示します。

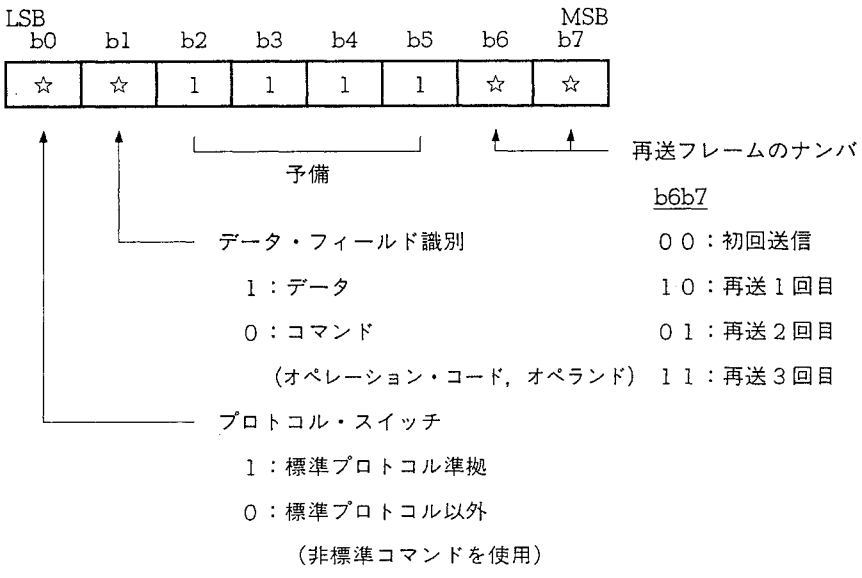
また，一斉同報，グループ同報の指定は，相手アドレス（DA）コードによって行います。

図11－9　同報時の相手アドレスのビット割り付け



11.2.9 制御コード

図11－10　制御コードのビット割り付け



11
214
218A
234
244

11.2.10　電文長コード

データ・フィールドのキャラクタ数を指定します。キャラクタ数は最大256個指定でき，01H-FFHは1-255，00Hは256を表します。

電文長は，キャラクタ数によって長電文と短電文に分けられます。

表11-4　短電文と長電文の区別

長電文	17-256キャラクタの電文
短電文	16キャラクタ以下の電文

11.2.11　チェック・コード

フレームの伝送エラー検出のために，自己アドレスからデータ・フィールドの最後のキャラクタまでの和の2の補数の値として，フレームの最後に送出します。チェック・コードは，計算結果の下位1バイトの値をとります。

11.2.12　ダ ミ ー

この期間はバス・アイドルの状態で，電文は存在しません。この期間をチェック・コードの計算に割り当てることも可能です。

11.2.13　アドレス・コードの割り当て

表11－5　アドレス・コードの割り当て

下位4ビット ＼ 上位4ビット	0	8	4	C	2	A	6	E	1	9	5	D	3	B	7	F
0	★	NT用（規格）注														（グループ0）
8	I群（コミュニケーション（CT）関連機器）															
4																（グループ1）
C																
2	II群（ハウス・キーピング（HK）関連機器）															（グループ2）
A																
6																（グループ3）
E																
1	III群（AVC関連機器）															（グループ4）
9																
5																（グループ5）
D																
3	IV群（予備，拡張用）															（グループ6）
B																
7																（グループ7）
F																

注　NT：ネットワーク・ターミネーション

備考1.　ホームバス・コントローラ（HBC：★印）のアドレス：［00H］

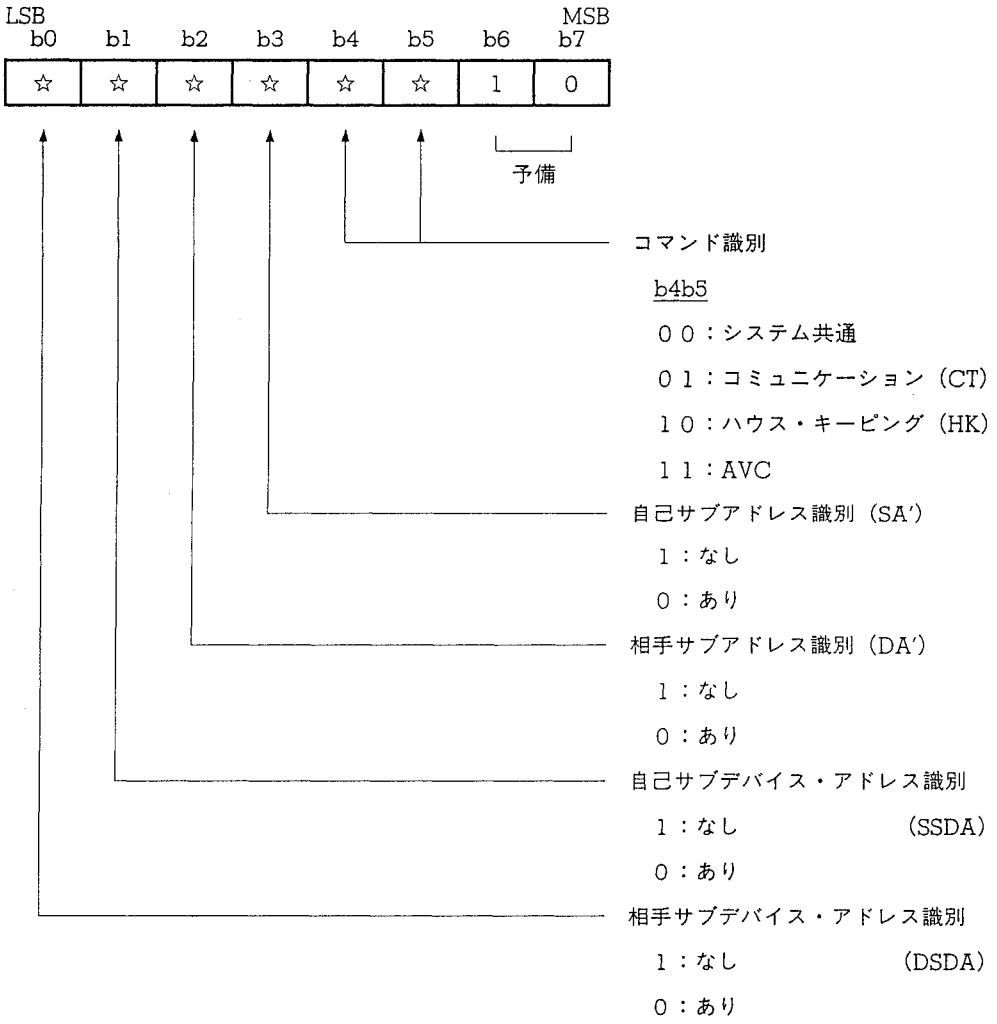
2.　自己アドレスのコード：［00H］-［FFH］

3.　相手アドレスのコード：［00H］-［FFH］

11
214
218A
234
244

11.2.14　ヘッダ・コード

図11－11　ヘッダ・コードのビット割り付け



11.2.15　システム共通コマンド

(1) コマンドの基本形

(a) OPC

(b) OPC＋OPR

OPC：オペレーション・コード

OPR：オペランド

(2) オペレーション・コード

上位4ビット＝8，9：オペランドなし

上位4ビット＝A，B，C，D，E，F：オペランドあり

(3) 必須，自由採用コマンド

送信元
- 必須　　…必ず送信しなければならない
- 自由採用…送信してもしなくてもよい

受信元
- 必須　　…無視してはならない
- 自由採用…無視してもよい

11
214
218A
234
244

(4) システム共通コマンド一覧

表11－6　システム共通コマンド一覧

下位4ビット	上位4ビット														
	オペランド						オペレーション・コード								
	0	1	2		6	7	8	9	A	B	C	D	E	F	
0							システム・リセット	回線捕捉不可	登録開始	情報チャネル使用問い合わせ	着信	相手ビジィ	ダイヤル・イン着信		ツイスト・ペア・ケーブル用
1							チャネル要求	回線使用終了	登録要求	情報チャネル使用中	着信応答	相手呼び出し中	ダイヤル・イン着信		
2							チャネル使用不可	OK	登録確認	電話回線状態通知	着信応答確認	ダイヤル・トーン受信	透過転送		
3							チャネル使用終了	NG	チャネル使用許可No.指定付き	情報チャネル接続	着信終了	2DT受信			
4							チャネル使用許可	ダミー	チャネル要求No.指定付き	発信要求	属性要求	モデム使用要求			
5							情報チャネル使用一斉問い合わせ		チャネル使用終了No.指定付き	回線捕捉可	属性応答	モデム使用終了			
6							情報チャネル接続可		状態アナウンス	ダイヤル	極性反転	保留			
7							情報チャネル接続不可		情報チャネル解放	ダイヤル	相手応答	保留解除		注	
8							システム・リセット		登録開始	情報チャネル使用問い合わせ					同軸ケーブル用
9							チャネル要求		登録要求	情報チャネル使用中					
A							チャネル使用不可		登録確認				透過転送		
B							チャネル使用終了		チャネル使用許可No.指定付き	情報チャネル接続					
C							チャネル使用許可		チャネル要求No.指定付き						
D		TC							チャネル使用終了No.指定付き						
E							情報チャネル接続可		状態アナウンス						
F							情報チャネル接続不可		情報チャネル解放					注	

オペランドなし　　オペランドあり

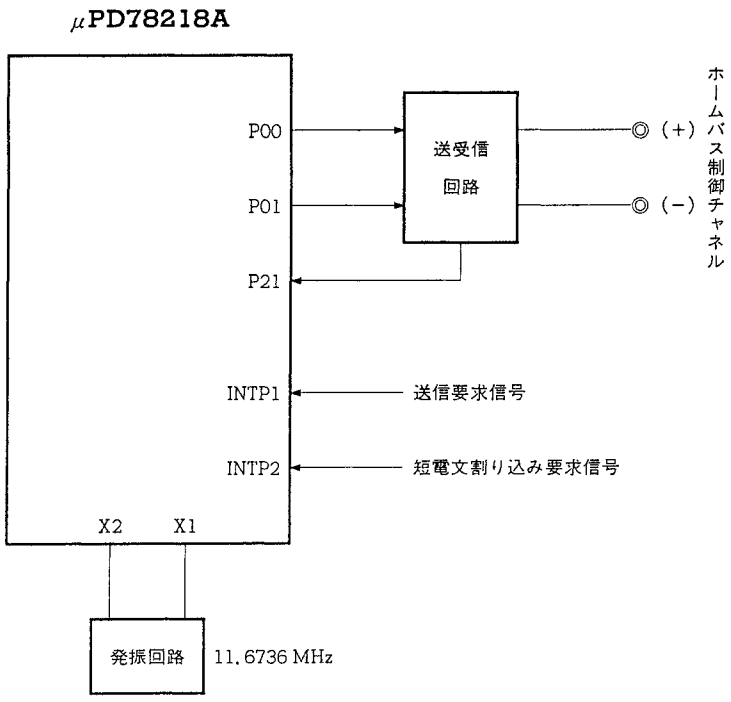
注　コマンド拡張用コマンド

備考　：必須コマンド

11.3　ハードウエア構成

11.3.1　システム構成

図11－12　システム構成図



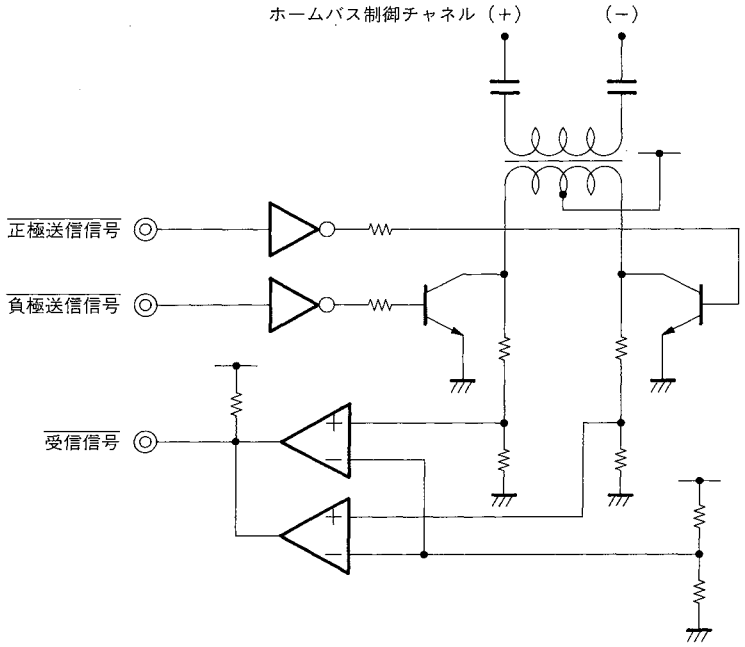
このシステム例では，伝送速度9600 bpsを高精度で実現するために，動作クロックを11.6736 MHz（内部システム・クロック＝5.8368 MHz）としています。

11
214
218A
234
244

11.3.2　送受信回路

この応用例において使用している送受信部分の回路を図11－13に示します。

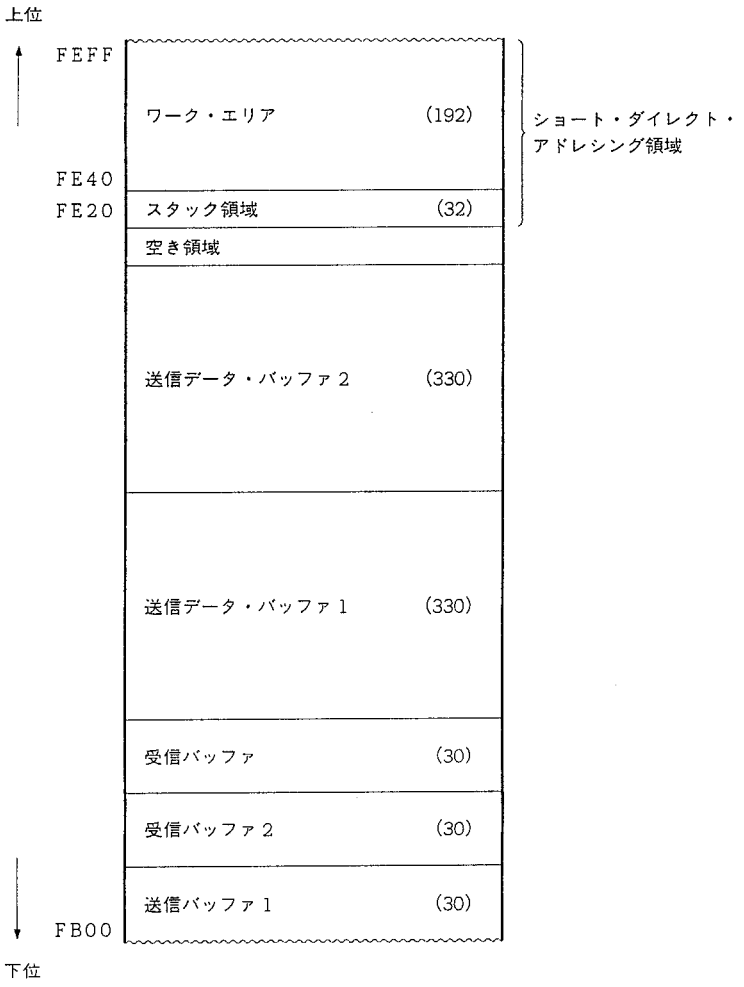
図11－13　送受信回路



11.3.3　内部RAMの割り付け

内部RAM　FB00H-FEFFH（1024バイト）を次のような形で使用します。

図11－14　内部RAMの割り付け



備考　（　）内の数字はバイト数

11
214
218A
234
244

11.4　動作概要

11.4.1　ホームバス・システムとしての動作

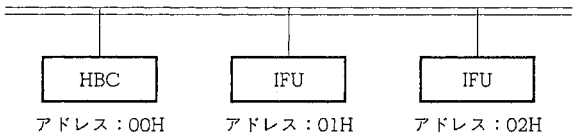
このプログラム例では，3台のHBPCが動作するシステムとなっています。

ホームバス・システムでは，一般的にホームバス・コントローラ（以降HBCと称す）が主体となって端末機器（以降IFUと称す）の情報を管理したり，システムの異常検出時にはIFUに指令を出したりします。

このプログラム例では，1台がHBC（自己アドレス：00H）として，2台がIFU（自己アドレス：01H，02H）として動作する形になっています。

またシステム・リセット直後には，HBCとIFUがシステム立ち上げのシーケンス（ネットワーク・レイヤに相当）を実行します。

図11－15　本プログラム例のホームバス・システム



11.4.2　送受信の実現方法

(1) 送　　信

リアルタイム出力ポートを使用します。

P00を＋側の出力ポートとし，P01を－側の出力ポートとしますので，バッファ・レジスタP0Lの下位2ビットが波形データとなります。このデータを8ビット・タイマ／カウンタ1の割り込みINTC10のタイミングで，P00，P01より出力することによって送信を実現します。

送信データの伝送速度は9600 bpsですので，1ビットの幅は1/9600≒104 μsとなります。また**11.2.1 (1) 制御信号**に示すように，AMI波形のデューティ比は50％ですので，1ビットあたり52 μsのタイミングで2回の出力を行います。

8ビット・タイマ／カウンタ1のカウント・クロックをf_{CLK}/16とし，コンペア・レジスタ（CR10）に設定する値は，次のようになります。

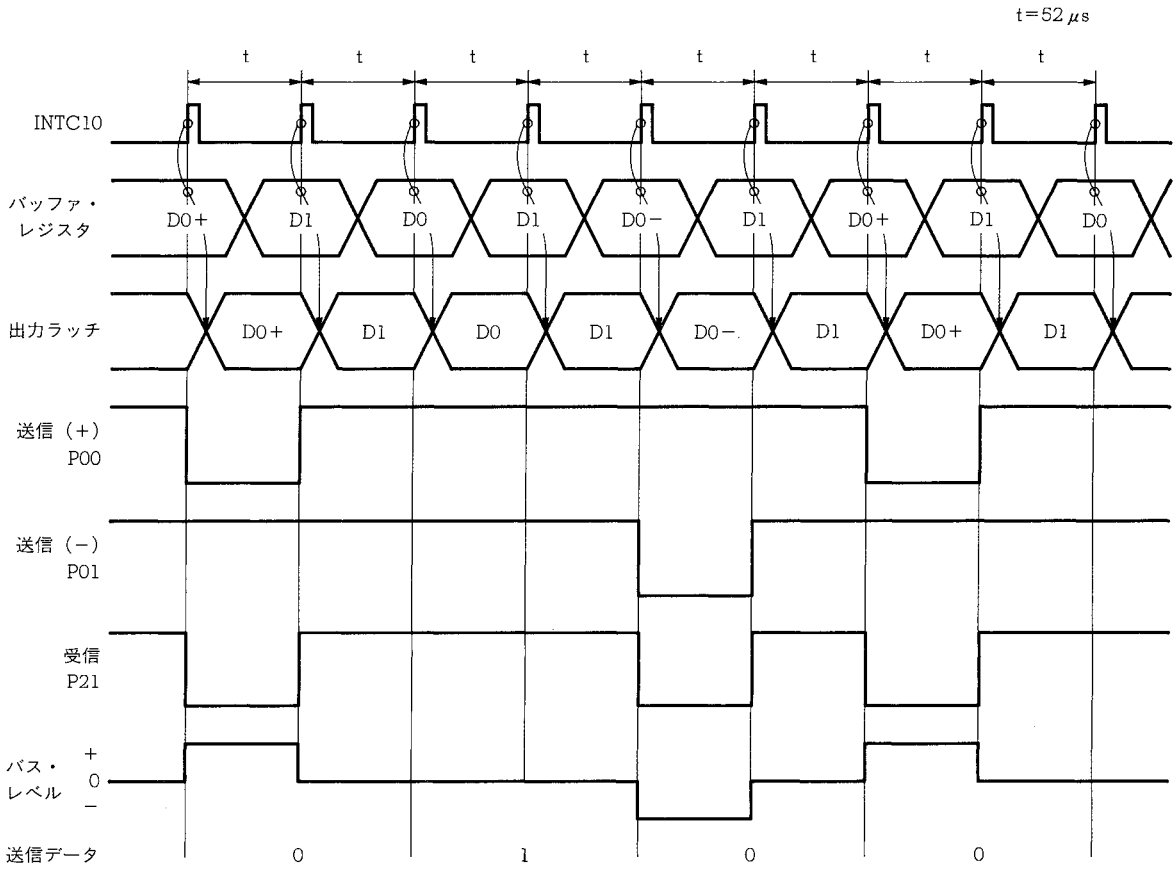
$$CR10=(1/9600)/2\div\frac{1}{11.6736\times10^{6}/2/16}-1$$
$$=19-1$$
$$=18$$

(2) 受　　信

ホームバス上の信号が，INTP0エッジ割り込み端子(INTP0/P21)に入力されますので，INTP0エッジ検出でスタート・ビットを検出し，次のビット以降はINTC10の発生タイミング（52 μs）でP21のレベルを読むことによって，受信を実現します。

送受信のタイミング・チャートを図11－16に示します。

図11－16　送受信のタイミング・チャート



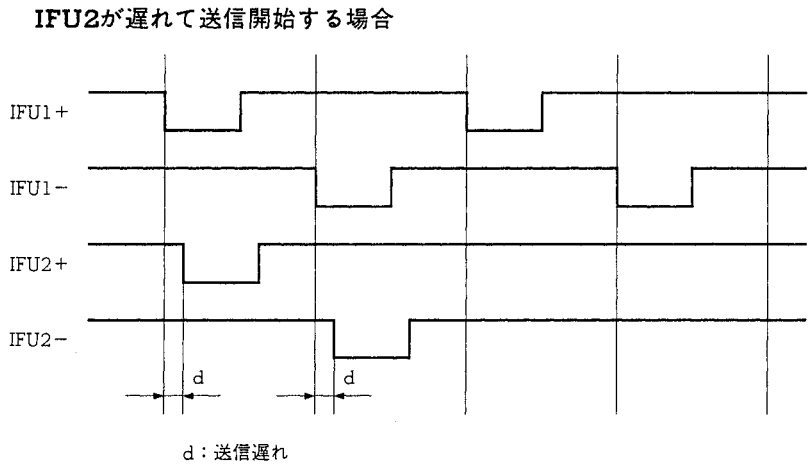
(3) 競合送信

他の送信に競合して送信を行う場合は，INTP0トリガによるリアルタイム・ポート出力を使用します。

スタート・ビット(INTP0立ち下がりエッジ)を検出すると同時に，リアルタイム出力ポートのバッファ・レジスタ（POL）の内容が出力されます。この場合，送信遅れ時間は，規格で定められている許容時間　13 μsに対し，実際のエッジ入力から78K/II内部で有効エッジと判断されるまでの12クロック（約2 μs：内部システム・クロック＝5.8368 MHzの場合）となります。

11
214
218A
234
244

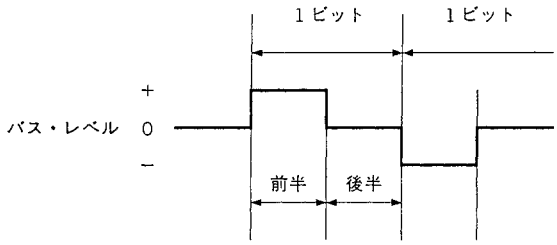
図11－17　競合送信のタイミング・チャート



11.4.3　送信波形データの形式

送信波形データは，1ビットにつき前半ビットの出力データ（オンデューティ・データ：以降前半データ）と後半ビットの出力データ（オフデューティ・データ：以降後半データ）が必要となります。

図11－18　1ビットの構成



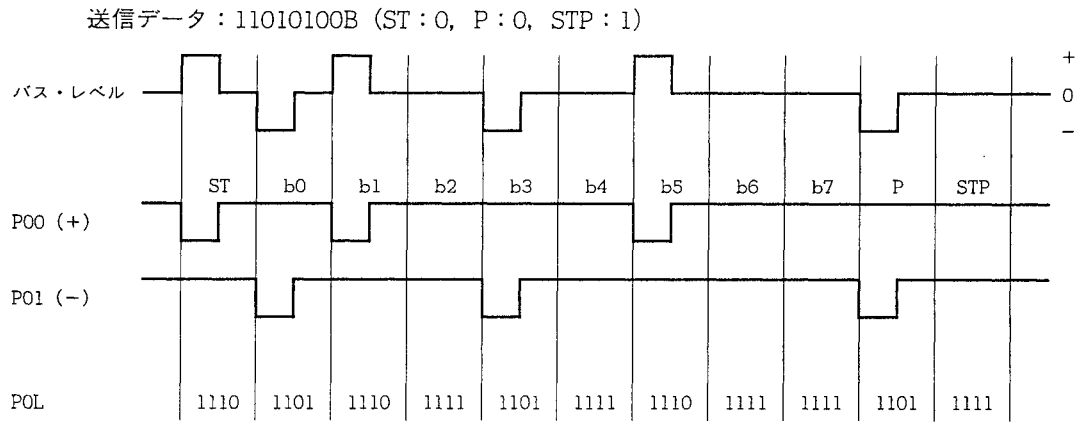
また，AMI波形ではアクティブ・データ（0）を＋側，－側より交互に出力しますので，前半データは，送信データに応じて＋，－，0レベルの3種類のデータが必要です。

＋レベルは＋側（P00）にロウ・レベルを出力しますので××××××10B，－レベルは－側（P01）にロウ・レベルを出力しますので××××××01B，0レベルは＋側（P00），－側（P01）ともハイ・レベルを出力しますので××××××11Bとなります。

後半ビットは必ず0レベルですので，出力データは××××××11Bとなります。

送信データと送信波形データ（リアルタイム・ポート出力データ）の対応を図11－19に示します。

図11－19　送信データと送信波形データの対応



備考1. P0Lデータは前半ビット・データです。後半データは，すべて1111Bとします。

2. 上位4ビット（P0H側）は1111Bとしますが，出力時にはマスクされます。

11.4.4　データ・バッファの構成

(1) 送受信バッファ

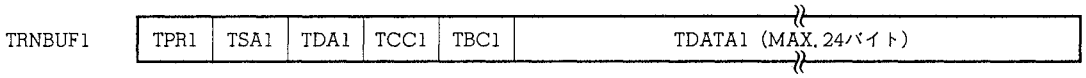
このプログラム例では，優先コード（PR）からチェック・コード（FCC）まで，あわせて30バイトのデータ送信，受信が可能です。したがって，30バイトの送信バッファ，受信バッファを持ちます。

また送信中の短電文割り込みを可能としていますので，短電文割り込み時に元の送信データを保持しておくために，送信バッファを2つ設けています。

送信データ，受信データとも，優先コード（PR）からチェック・コード（FCC）までの格納場所は固定としています。次に送受信バッファの名称と構成を示します。

図11－20　送受信バッファの名称と構成

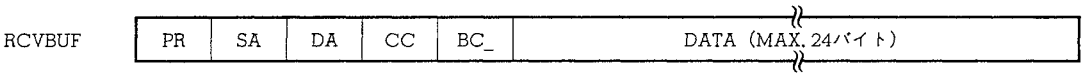
(a) 送信バッファ1



(b) 送信バッファ2（短電文割り込み用）



(c) 受信バッファ



11
214
218A
234
244

チェック・コード（FCC）は，データ（DATA）の最後尾に置くものとし，電文長コード（BC）によってその位置は変化します。

(2) 送受信データ・バッファ

送信データ1ビットにつき1バイトの前半データが必要ですので，スタート・ビットからストップ・ビットまでの11ビットで送信波形データは11バイトとなります。したがって，送信データ・バッファの大きさは送信バッファ長（30バイト）×11＝330バイトとなります。

11.4.5　送信要求，短電文割り込み要求の発行

送信要求はINTP1割り込み要求とします。また短電文割り込み要求はINTP2割り込み要求とします。これらは送信が終了するまで取り下げません。

11.4.6　休止時間，休止期間のカウント方法

(1) 同期回復時間を除いた時間のカウント

8ビット・タイマ／カウンタ3のカウント・クロックを$f_{CLK}/512$としてカウントします。この場合のコンペア・レジスタ（CR30）に設定する値は次のようになります。

(a) 通常の通信終了

$$10\,\text{ms}-2\,\text{ビット}=9.792\,\text{ms}$$

$$\text{CR30}=9.792\,\text{ms}\div\frac{1}{11.6736\,\text{MHz}/2/512}-1$$

$$\fallingdotseq 112-1$$

$$=111$$

(b) 同報終了，短電文割り込みによる送受信中断

$$10\,\text{ms}+22\,\text{ビット}-2\,\text{ビット}=12.083\,\text{ms}$$

$$\text{CR30}=12.083\,\text{ms}\div\frac{1}{11.6736\,\text{MHz}/2/512}-1$$

$$\fallingdotseq 138-1$$

$$=137$$

(2) 同期回復時間のカウント

8ビット・タイマ／カウンタ3のカウント・クロックを$f_{CLK}/16$として208 μsをカウントします。コンペア・レジスタ（CR30）に設定する値は次のようになります。

$$10\,\text{ms}-9.792\,\text{ms}=208\,\mu\text{s}$$
$$\text{CR30}=208\,\mu\text{s}\div\frac{1}{11.6736\,\text{MHz}/2/16}-1$$
$$\fallingdotseq 76-1$$
$$=75$$

11.4.7　エラー検出

次の通信エラーを検出します。

(1) 受信側が検出するエラー

パリティ・エラー　　：受信パリティと計算パリティが不一致
チェック・サム・エラー：受信チェック・コードと計算チェック・コードが不一致
フレーミング・エラー　：ストップ・ビットが検出されない（データ（DATA）部は除く）
タイム・アウト・エラー：フレーム受信中，定時間内にスタート・ビットが検出されない
　　（電文長コードより受信フレームの電文長が短い場合も含む）
ダミー・データ・エラー：ダミー期間中にデータが検出された
　　（電文長コードより受信フレームの電文長が長い場合も含む）

(2) 送信側が検出するエラー

NAKエラー　　：ACKコード（06H）以外を受信
無応答エラー　　：ACK/NAKデータが返送されない
送信エラー　　：再送不可能
オーバ・データ・エラー：電文長コードが24を越えている

11.4.8　受信タイム・アウト

1フレームのデータ受信は，1パケットごとに同期を取り直すために，毎回スタート・ビットを検出しながら行います。前パケットのストップ・ビットの先頭から，次のデータのスタート・ビットを検出するまでの時間は，正常に通信が行われている場合は，1ビット時間(104 μs)ですが，10 ms間スタート・ビットが来なかった場合は，受信タイム・アウトとします。

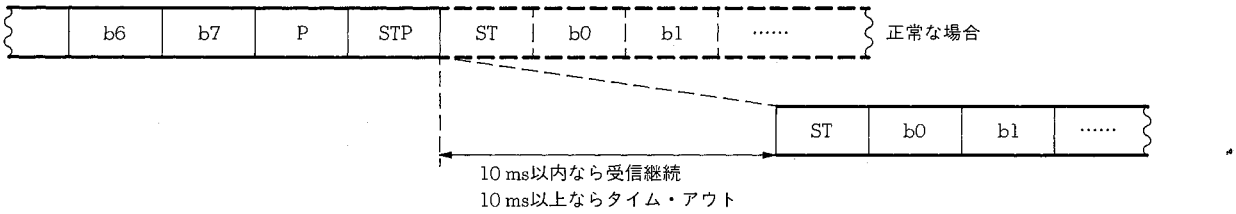
その後，休止時間相当時間だけバスを監視し，同期回復を行います。

タイム・アウト時間は，8ビット・タイマ／カウンタ3のカウント・クロックを$f_{CLK}/512$としてカウントします。

11
214
218A
234
244

$$CR30 = 10\,ms \div \frac{1}{11.6736\,MHz/2/512} - 1$$
$$\fallingdotseq 114 - 1$$
$$= 113$$

図11－21　受信タイム・アウトの判定タイミング



11.4.9　コマンド処理

(1) システム共通コマンド

システム共通コマンドのうち必須コマンド（同軸ケーブル用コマンドと，ツイスト・ペア・ケーブル用コマンド）については，すべてコマンド・コード（オペレーション・コード）をプログラム中で定義しています。

またこのプログラムでは，これらのうち優先レベル1のコマンド（システム特権パケット）について処理を行っています。優先レベル1のコマンドを次に示します。

80H：システム・リセット[注]　(HBC→端末)
A0H：登録開始　(HBC→端末)
A1H：登録要求　(端末→HBC)
A2H：登録確認　(HBC→端末)

注　端末側の処理は規定しません。

(2) HBC側の処理

自己アドレスが00Hの場合，次のコマンド処理を行います。

表11－7　HBC用のコマンドとデータ・フォーマット

コマンド・コード[注1]	機　　能	データ・フォーマット（DATA）[注2]
80H	システム・リセット	[80]
A0H	登録開始	[A0]，[60－67（アドレス・グループ）]
A2H	登録確認	[A2]，[60-67（情報チャネル状態）]， [60-66（電話回線１の状態）]， [70-76（電話回線２の状態）]， [60-6F（端末属性（基本））]， [70-7F（端末属性（詳細））]， [60-6F（回線属性：回線１）]， [70-7F（回線属性：回線２）]

注1．ツイスト・ペア・ケーブル用のみ記載。

2．詳細はET-2101を参照。

登録開始データと登録確認データは例としてプログラムで定義しておき，送信前に送信バッファに転送しています。

(3) IFU側の処理

自己アドレスが01H，02Hの場合，次のコマンド処理を行います。

表11－8　IFU用のコマンドとデータ・フォーマット

コマンド・コード	機　　能	データ・フォーマット（DATA）
A1H	登録要求	[A1]，[60-6F（端末属性（基本））]， [70-7F（端末属性（詳細））]， [60-6F（回線属性：回線１）]， [70-7F（回線属性：回線２）]

登録要求データは，例としてプログラムで定義しておき，送信前に送信バッファに転送しています。

(4) システム立ち上げシーケンス

システム立ち上げ時には，HBCとIFUの間で機器情報などの登録を行います。

処理は，ET-2101の通信シーケンスに従います。

図11－22にアドレス・グループが異なる場合，図11－23にアドレス・グループが同じ場合のシーケンスを示します。

11
214
218A
234
244

図11－22　異なるアドレス・グループのシーケンス

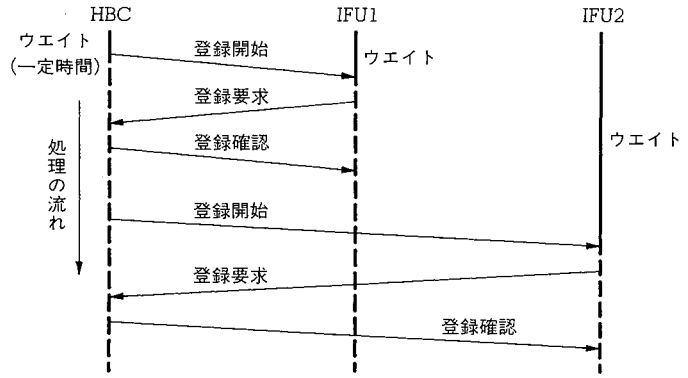
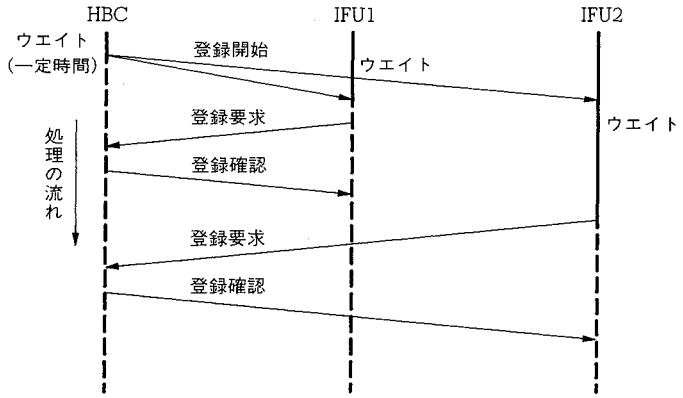
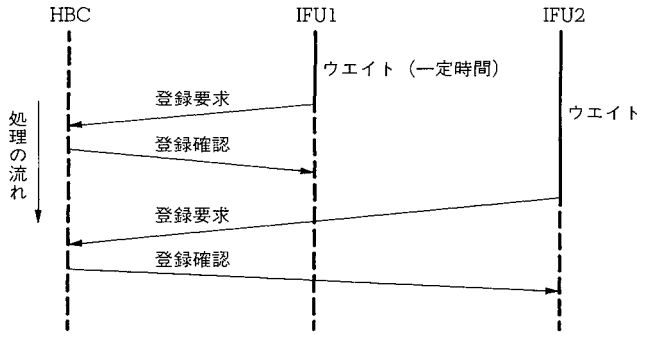


図11－23　同アドレス・グループのシーケンス



またIFUは，一定時間ウエイト後，登録開始コマンド・フレームを受信しない場合，自分から登録要求を出します。この場合のシーケンスを図11－24に示します。

図11－24　IFUが先に登録要求を出す場合のシーケンス



11.5　制御プログラム

11.5.1　処理概要と状態遷移

送受信中の処理は，送信，受信，監視の３つのモードに分かれます。これらの処理は，すべて送受信のタイミングを生成するINTC10割り込み処理で行います。

このプログラム例では，送受信もしくは監視中は，送信要求，短電文割り込み要求の割り込みしか受け付けていませんが，実際のシステムにおいて他の割り込みを使用する場合は，INTC10処理の受け付けが遅れないよう，その受け付けタイミングと処理時間に注意する必要があります。

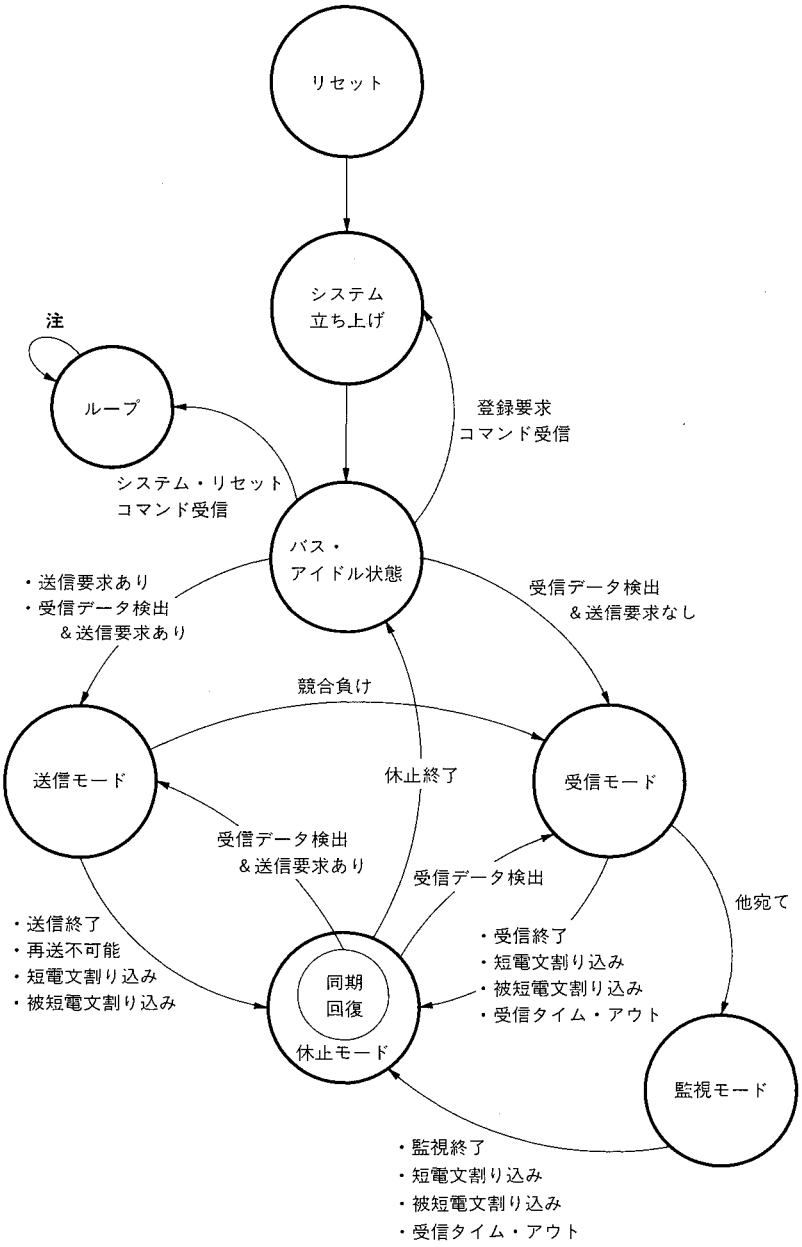
休止時間中は，休止モードとします。休止終了２ビット前から行う同期回復中も休止モードとし，この期間は競合送信と受信を可能とします。同期回復以外の休止モードでは，送信波形データの作成や通信以外のメイン処理も行います。

何も通信が行われていない状態をバス・アイドル状態とします。この状態のときは，同期回復以外の休止モードのときと同様の処理が可能で，さらに競合送信以外の送信開始はこの状態のときのみ可能とします。

図11－25に状態遷移図，図11－26に概要フロー・チャートを示します。

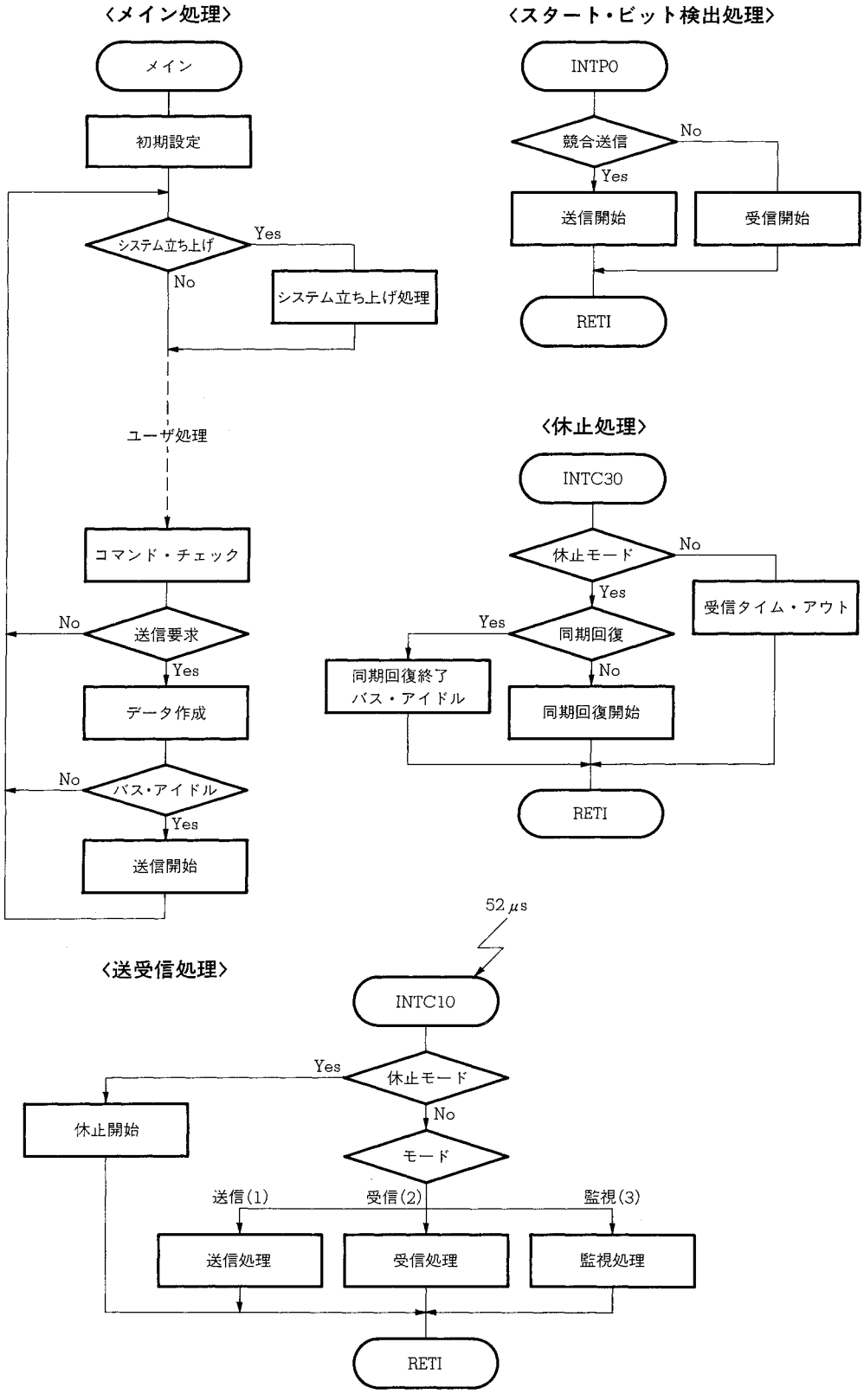
11
214
218A
234
244

図11－25　状態遷移図



注　実際のシステムでは，リセットに戻すなどの処理が必要です。

図11－26　概要フロー・チャート



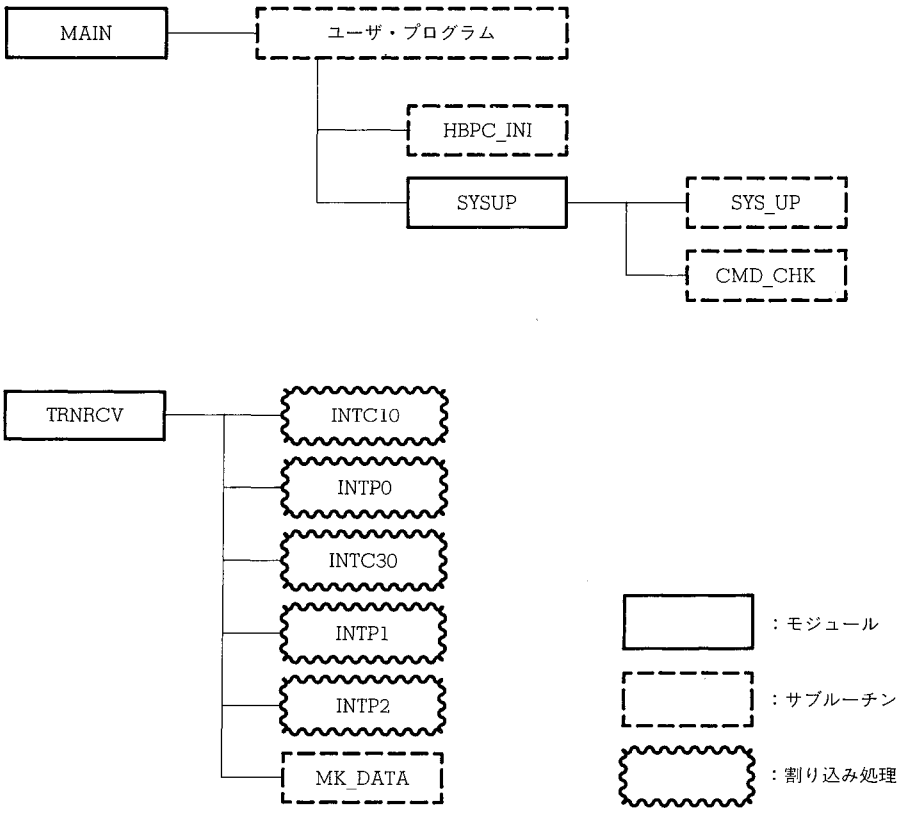
11
214
218A
234
244

11.5.2　モジュール構成

このプログラム例は，メイン（MAIN），システム立ち上げ（SYSUP），送受信（TRNRCV）の３つのモジュールから構成されます。

図11－27にモジュールとサブルーチンの構成を示します。

図11－27　モジュール，サブルーチン構成図



11.5.3　ワークRAM，ワーク・フラグ

(1) ワークRAM，ワーク・フラグ一覧

ワーク・メモリは，すべてショート・ダイレクト・アドレシング領域に配置します。

表11-9　ワークRAM一覧

シンボル	名　　称	サイズ	初 期 値
MODE	モード番号	1バイト	00H
DATCNT	データ・カウンタ	1バイト	1
BITCNT	ビット・カウンタ	1バイト	11
RSTCNT	リセット時カウンタ	1バイト	RSTWAITI
RCVDAT	一時受信バッファ	1バイト	—
CHKSAV	受信中チェック・サム	1バイト	00H
TRBFPNT	送信バッファ・ポインタ	2バイト	TRNDAT1
CHRCNT	電文長カウンタ	1バイト	0
PRCODE	優先コード・ワーク	1バイト	—
CTLCODE	制御コード・ワーク	1バイト	—
ERROR	エラー・フラグ・エリア	1バイト	00000000
SYSUPMOD	システム立ち上げモード番号	1バイト	0
SELFADRS	自己アドレス	1バイト	—
GRPCNT	アドレス・グループ・カウンタ	1バイト	0
BITSAV	受信ビット・セーブ	1ビット	—
PRTYSAV	パリティ・ビット・セーブ	1ビット	0

11
214
218A
234
244

表11－10　ワーク・フラグ一覧

シンボル	名　称（…フラグ）	初期	シンボル	名　称（…フラグ）	初期
TRNENDF	送信終了	1	ACPTINTF	短電文割り込み受け付け	0
RCVENDF	受信終了	1	CLRMDATF	データ作成クリア	0
MKDATF	送信データ作成終了	0	DTCLSHF	衝突検出（競合負け）	0
PAUSEF	休止モード	0	DTINTF	短電文割り込み検出	0
PAUSE2F	休止期間	0	INPUTDAT	データ入力終了	0
BITF	ビット判断	0	TRNINTF	送信中割り込み	0
INTCHRF	短電文割り込み中	0	TRNRDYF	送信準備	0
RQTRNF	送信要求	0	SYNCF	同期回復中	0
RQTRN2F	送信要求予備	0	CMPTTRF	競合送信中	0
RQINTF	短電文割り込み要求	0	IC30F	INTC30処理中	0
BUSIDLF	バス・アイドル	1	CMDCHKF	コマンド・チェック	0
TOSELFF	自分宛て	0	EP0TRGF	INTP0トリガ出力許可	0
SYSRSTF	システム・リセット要求	0	PCCHKF	PCチェック	0
RGSTRQF	登録要求（途中）	0	OVDATERF	オーバ・データ・エラー	0
SYSUPF	システム立ち上げ	0			

(2) ワークRAM，ワーク・フラグ説明

表11－11　ワークRAM説明（1/4）

RAM名称	内容（用途，値の意味など）
MODE	【用　途】 送信，受信，監視モードの番号を格納 【値の意味】 0：モード外 1：送信モード 2：受信モード 3：監視モード 【値の更新条件】 1フレームの通信終了，衝突検出，他宛て電文受信
DATCNT	【用　途】 通信データの種類（1パケット単位）を示す番号を格納 【値の意味】 1：優先コード（PR） 2：自己アドレス（SA） 3：相手アドレス（DA） 4：制御コード（CC） 5：電文長コード（BC） 6：データ（DATA） 7：チェック・コード（FCC） 8：ダミー（DMY） 9：ACK/NAK 10：フレーム終了 【値の更新条件】 1パケットの通信終了　→　インクリメント
BITCNT	【用　途】 通信ビットの位置（1パケット内）を示す番号を格納 【値の意味】 11：スタート・ビット 10－3：データ 2：パリティ・ビット 1：ストップ・ビット 0：1パケット終了 【値の更新条件】 1ビットの通信終了　→　デクリメント

11
214
218A
234
244

表11－11　ワークRAM説明（2/4）

RAM名称	内容（用途，値の意味など）
RSTCNT	【用　途】 IFUセット時のウエイトのカウント数を格納 【値の意味】 0　　：ウエイト中でない 1－100：ウエイト中 【値の更新条件】 →　INTC30要求発生時にデクリメント
RCVDAT	【用　途】 受信データ（スタート，パリティ，ストップ・ビットを除く８ビット）を一時的に格納 【値の意味】 — 【値の更新条件】 １ビット受信。ただし８ビット更新(１ビット受信ごとにシフト)で，１パケット分のデータ更新となる。
CHKSAV	【用　途】 受信中に計算したチェック・コードの下位１バイトを格納 【値の意味】 — 【値の更新条件】 １パケット（８ビット）受信後，前値と加算
TRBFPNT	【用　途】 現在の送信波形データ・エリアのアドレスを格納 【値の意味】 — 【値の更新条件】 １ビット送信　→　インクリメント
CHRCNT	【用　途】 電文長のカウント数を格納 【値の意味】 — 【値の更新条件】 データ（DATA）において１パケットのデータ受信終了　→　インクリメント

表11－11　ワークRAM説明（3/4）

RAM名称	内容（用途，値の意味など）
PRCODE	【用　途】 プライオリティ（パケット・プライオリティ，短電文プライオリティ，同報種別プライオリティ，再送プライオリティ）読み出しのために優先コードを格納 【値の意味】 — 【値の更新条件】 優先コード（PR）を1パケット受信（送信モード時も優先コードは受信）
CTLCODE	【用　途】 再送回数のカウントのために制御コードを格納 【値の意味】 0：再送なし 1：再送回数1回 2：再送回数2回 3：再送回数3回 【値の更新条件】 ACK/NAK受信時に受信エラー検出＆再送回数が3回未満
ERROR	【用　途】 エラー状態をビットごとに格納 【値の意味】 ERROR.0：パリティ・エラー ERROR.1：チェック・サム・エラー ERROR.2：フレーミング・エラー ERROR.3：タイム・アウト・エラー ERROR.4：ダミー・データ・エラー ERROR.5：NAKエラー ERROR.6：無応答エラー ERROR.7：送信エラー 1：エラーあり，0：エラーなし 【値の更新条件】 上記エラーが発生し，検出した時点

11
214
218A
234
244

表11－11　ワークRAM説明（4/4）

RAM名称	内容（用途，値の意味など）
SYSUPMOD	【用　途】 システム立ち上げモード時のシーケンス（登録開始，登録要求，登録確認，登録終了）番号を格納 【値の意味】 0：登録開始 1：登録要求 2：登録確認 3：登録終了 【値の更新条件】 システム立ち上げモード時，1シーケンス終了
SELFADRS	【用　途】 自己アドレスを格納 【値の意味】 00H　　：ホームバス・コントローラ（HBC） 01H－02H：端末（IFU） 【値の更新条件】 リセット後1回のみ。その後変更しない。
GRPCNT	【用　途】 登録するアドレス・グループ数を示す。 【値の意味】 0：アドレス・グループ数0 1：アドレス・グループ数1 2：アドレス・グループ数2 【値の更新条件】 HBCモード時，システム立ち上げの一連のシーケンス（登録開始，登録要求，登録確認，登録終了）終了＆全グループに登録開始コマンドを送信していない
BITSAV	【用　途】 受信データ（1ビット）を格納 【値の意味】 — 【値の更新条件】 1ビット受信
PRTYSAV	【用　途】 受信中に計算（前のパリティとXOR）したパリティ・ビットを格納 【値の意味】 — 【値の更新条件】 1ビット受信

表11－12　ワーク・フラグ説明（1/8）

フラグ名称	内容（用途，値の意味など）
TRNENDF	【用　途】 送信終了を示す 【値の意味】 0：送信が終了していない 1：送信終了 【値の更新条件】 0：送信終了チェック後 1：送信が終了し，休止時間経過後
RCVENDF	【用　途】 受信終了を示す 【値の意味】 0：受信が終了していない 1：受信終了 【値の更新条件】 0：受信終了チェック後 1：受信が終了し，休止時間経過後
MKDATF	【用　途】 データ作成終了を示す 【値の意味】 0：データ作成が終了していない 1：データ作成終了 【値の更新条件】 0：データ作成クリア・フラグ（CLRMDATF）＝1 1：データ作成終了（MK_DATAサブルーチンから復帰）
PAUSEF	【用　途】 休止モードであることを示す 【値の意味】 0：休止モードでない（送信モード，受信モード，監視モード，バス・アイドル状態） 1：休止モード 【値の更新条件】 0：休止時間経過（同期回復終了） 1：1フレームの通信終了，短電文割り込み検出，短電文割り込み要求あり

11
214
218A
234
244

表11－12　ワーク・フラグ説明（2/8）

フラグ名称	内容（用途，値の意味など）
PAUSE2F	【用　途】 休止モードで，休止期間（10 ms＋22 bit）をカウントすることを示す 【値の意味】 0：休止期間をカウントしない 1：休止期間をカウントする 【値の更新条件】 0：休止期間のカウント・データをCR30に設定後 1：短電文割り込み検出，短電文割り込み要求あり
BITF	【用　途】 通信が，1ビットの前半か後半かを示す 【値の意味】 0：後半 1：前半 【値の更新条件】 0：発生したINTC10で後半データを送信（受信，監視） 1：発生したINTC10で前半データを送信（受信，監視）
INTCHRF	【用　途】 次のINTC10のタイミングで，短電文割り込みのためのブレーク信号を出力することを示す 【値の意味】 0：短電文割り込み要求なし，ブレーク信号出力不可 1：ブレーク信号出力許可 【値の更新条件】 0：割り込み電文の送信が終了し，休止モード解除時 1：短電文割り込み要求あり
RQTRNF	【用　途】 送信要求の有無を示す 【値の意味】 0：送信要求なし 1：送信要求あり 【値の更新条件】 0：1フレームのデータ送信が終了 （個別：DATCNT＝9，同報：DATCNT＝7） 1：送信要求発生（INTP1エッジ検出）

表11－12　ワーク・フラグ説明（3/8）

フラグ名称	内容（用途，値の意味など）
RQTRN2F	【用　途】 送信要求がすでにあるときに，さらに送信要求が発生したことを示す（送信要求が２レベル） 【値の意味】 ０：送信要求が２レベルでない １：送信要求が２レベル 【値の更新条件】 ０：先に発生した送信要求が送信終了によってクリア １：送信要求がすでにあり，さらに送信要求が発生
RQINTF	【用　途】 短電文割り込み要求の有無を示す 【値の意味】 ０：短電文割り込み要求なし １：短電文割り込み要求あり 【値の更新条件】 ０：１フレームの割り込み電文の送信が終了 １：短電文割り込み要求発生（INTP2エッジ検出）
BUSIDLF	【用　途】 バスがアイドル状態であることを示す 【値の意味】 ０：バス・アイドル状態でない １：バス・アイドル状態 【値の更新条件】 ０：送信または受信を開始 １：休止時間（期間）終了
TOSELFF	【用　途】 受信フレームが自分宛であることを示す 【値の意味】 ０：受信フレームが自分宛でない １：受信フレームが自分宛 【値の更新条件】 ０：電文長コードを受信 １：受信した相手アドレスと自己アドレス（SELFADRS）が一致

11
214
218A
234
244

表11－12　ワーク・フラグ説明（4/8）

フラグ名称	内容（用途，値の意味など）
SYSRSTF	【用　途】 システム・リセット要求の有無を示す 【値の意味】 0：システム・リセット要求なし 1：システム・リセット要求あり 【値の更新条件】 0：―（ユーザ・プログラムでクリア） 1：受信したオペコード（OPC）がシステム・リセット・コマンド（80H）
RGSTRQF	【用　途】 システム立ち上げ終了後（通常モード）の登録要求の有無を示す 【値の意味】 0：システム立ち上げ終了後の登録要求なし 1：システム立ち上げ終了後の登録要求あり 【値の更新条件】 0：メイン処理で登録要求が受け付けられた（通常モードがシステム立ち上げモードに設定された） 1：受信したオペコード（OPC）が登録要求コマンド（A1H）
SYSUPF	【用　途】 システム立ち上げ中であることを示す 【値の意味】 0：システム立ち上げ中でない（通常モード） 1：システム立ち上げ中 【値の更新条件】 0：システム立ち上げシーケンス（全グループの登録処理）終了 1：リセット直後，あるいは通常モードで登録要求発生
ACPTINTF	【用　途】 短電文割り込みの受け付け（データ作成，送信開始許可）中であることを示す 【値の意味】 0：短電文割り込み受け付け中でない 1：短電文割り込み受け付け中 【値の更新条件】 0：割り込み電文の送信が終了し，休止モード解除後 1：短電文割り込みのためのブレーク信号出力後

表11－12　ワーク・フラグ説明（5/8）

フラグ名称	内容（用途，値の意味など）
CLRMDATF	【用　途】 データ作成終了フラグ（MKDATF）のクリア許可／不可を示す 【値の意味】 0：データ作成終了フラグのクリア不可 1：データ作成終了フラグのクリア許可 【値の更新条件】 0：1フレームのデータ送信終了（ACK/NAK，FCCのストップ・ビットの後半）このフラグ＝1 1：1フレームのデータ送信終了（ACK/NAK，FCCのストップ・ビットの前半）で，送信中短電文割り込み中，競合負け中でない
DTCLSHF	【用　途】 送信中にデータ衝突があった（競合負けした）ことを示す 【値の意味】 0：データ衝突なし 1：データ衝突あり 【値の更新条件】 0：－（ユーザ・プログラムでクリア） 1：送信モード時，優先コード，自己アドレス部分で，送信ビットと受信ビットに不一致が検出された
DTINTF	【用　途】 短電文割り込みが検出されたことを示す 【値の意味】 0：短電文割り込みが検出されていない 1：短電文割り込み検出 【値の更新条件】 0：－（ユーザ・プログラムでクリア） 1：長電文のストップ・ビットの位置で，ブレーク信号が検出された
INPUTDAT	【用　途】 キー入力によるデータ設定が終了したことを示す 【値の意味】 0：キー入力によるデータ設定中，あるいはデータ設定していない 1：キー入力によるデータ設定終了 【値の更新条件】 0：キー入力中，本フラグ＝1 1：キー入力によるデータ設定終了（キー入力ルーチンでセット）

11
214
218A
234
244

表11－12　ワーク・フラグ説明（6/8）

フラグ名称	内容（用途，値の意味など）
TRNINTF	【用　途】 送信中に短電文割り込み要求があったことを示す 【値の意味】 0：送信中に短電文割り込み要求がない 1：送信中に短電文割り込み要求がある 【値の更新条件】 0：短電文割り込み要求クリア時で，本フラグ＝1 1：送信モード時，パリティ・ビット部で短電文割り込み要求がある
TRNRDYF	【用　途】 休止時間に送信を行うことを示す 【値の意味】 0：休止時間後に送信を行わない 1：休止時間後に送信を行う 【値の更新条件】 0：次の送信の準備（送信ポインタの設定など）完了 1：衝突検出，短電文割り込み検出，短電文割り込み要求，再送
SYNCF	【用　途】 同期回復中であることを示す 【値の意味】 0：同期回復中でない 1：同期回復中 【値の更新条件】 0：同期回復終了 1：同期回復開始
CMPTTRF	【用　途】 競合送信中であることを示す 【値の意味】 0：競合送信中でない 1：競合送信中 【値の更新条件】 0：INTPO処理中で競合送信中のチェック終了 1：競合送信処理（INTPO）中

表11－12　ワーク・フラグ説明（7/8）

フラグ名称	内容（用途，値の意味など）
IC30F	【用　途】 INTC30処理中であることを示す。INTC30処理中の多重割り込み時に使用 【値の意味】 0：INTC30処理中でない 1：INTC30処理中 【値の更新条件】 0：INTC30処理終了（RETI） 1：INTC30処理開始（割り込み受け付け直後）
CMDCHKF	【用　途】 コマンド・チェックを行うことを示す 【値の意味】 0：コマンド・チェックを行わない 1：コマンド・チェックを行う 【値の更新条件】 0：コマンド・チェック処理実行 1：受信モードで休止時間終了時
EP0TRGF	【用　途】 INTP0トリガによるリアルタイム出力許可／禁止の判断 【値の意味】 0：INTP0トリガによるリアルタイム出力禁止 1：INTP0トリガによるリアルタイム出力許可 【値の更新条件】 0：送信，競合送信開始 1：RTPCレジスタにてINTP0トリガによるリアルタイム出力許可に設定
PCCHKF	【用　途】 メイン処理で送信開始処理を開始していないかの判断 【値の意味】 0：メイン処理で送信開始処理を開始していない 1：メイン処理で送信開始処理を開始している 【値の更新条件】 0：メイン・ループの先頭に分岐 1：メイン処理の送信開始処理を開始

11
214
218A
234
244

表11－12　ワーク・フラグ説明（8/8）

フラグ名称	内容（用途，値の意味など）
OVDATERF	【用　途】 オーバ・データ・エラー（電文長が24を越えている）検出の判断 【値の意味】 0：オーバ・データ・エラーなし 1：オーバ・データ・エラーあり 【値の更新条件】 0：休止時間終了時 1：送信波形データ作成中に電文長が24を越えている

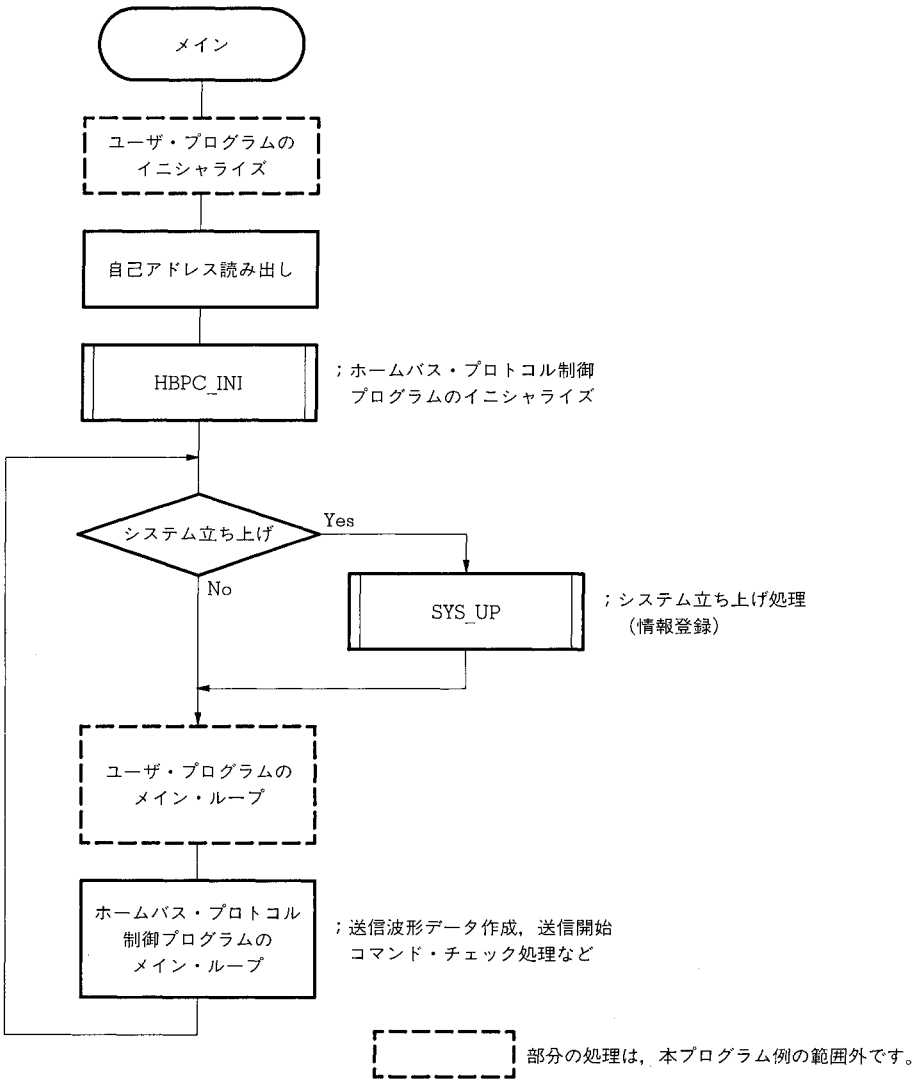
11.5.4　プログラム説明

(1) プログラム使用方法

このプログラム例は，ホームバス・プロトコルの制御部分のみですので，実際に使用する場合は，ユーザ・プログラムにこのプログラム例を組み込む形となります。この場合のユーザ・プログラムとは，他のHBPCとコマンドのやりとりを行ったり，外部に接続したホストや端末機器とのやりとりを行ったりするプログラムを示します。

図11－28にこのプログラムを組み込んだ場合のメイン処理のフロー・チャート例を示します。

図11－28　フロー・チャート例



このほかに，INTC10，INTP0，INTP1，INTP2割り込み処理が必要です。

11
214
218A
234
244

(2) ホームバス・プロトコル制御プログラムのメイン・ループ

本処理を，ユーザ・プログラムのメイン・ループに組み込んでください。

次の内容の処理を行っています。

○コマンド処理
○送信波形データ作成処理
○送信開始処理

(a) コマンド処理

ここでコマンド処理を行うか行わないかをチェックし，行う場合はコマンドの種類をチェックしたあと，コマンド処理を行います。

次の場合は，コマンド・チェックを行いません。

- システム立ち上げ処理中（SYSUPF＝1）
- 受信終了でない（RCVENDF＝0）
- コマンド・チェック・フラグ（CMDCHKF）＝0

上記以外の場合は，コマンド・チェック・サブルーチン（CMD_CHK）を呼び出します。

コマンド・チェックの結果，受信したコマンドがシステム・リセット・コマンド（80H）の場合（SYSRSTF＝1）は，その場で永久ループしています。実際のシステムでは，イニシャライズ処理に分岐するなどの処理を行ってください。

受信したコマンドが登録要求コマンド（A1H）の場合（RGSTRQF＝1：HBCのみにあり得る）は，途中で登録要求が来たとみなし，このあと登録確認コマンド・フレームを返送します。

(b) 送信波形データ作成処理（レーベル：ACPT_INT）

次の条件が両立している場合，送信バッファ（TRNBUF1もしくはTRNBUF2）の内容からチェック・コード（FCC）を生成し，データ作成サブルーチン（MK_DATA）を呼び出します。

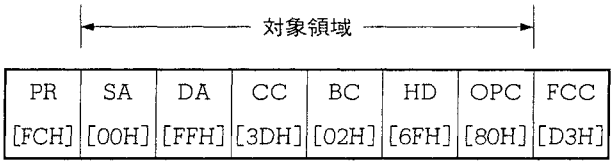
〈条件〉

- 送信要求（RQTRNF＝1）もしくは短電文割り込み要求（RQINTF＝1）あり
- データ入力終了（INPUTDAT＝1）

データ入力終了フラグ（INPUTDAT）は，システム立ち上げ処理内の送信，短電文割り込みによる送信，エラー再送，競合負け後の送信の場合にはプログラム中でセット（1）されますが，通常の送信（バス・アイドル中に送信要求があった）の場合にはセットされません。したがって，この場合には，ユーザ・プログラムで外部からの送信バッファへのデータ設定終了時，INPUTDATをセットしてください。

チェック・コード生成は，次のように行っています。

例　電文長が２バイトのフレームのチェック・コード（FCC）の計算例



$$
\begin{aligned}
\text{計算式} &= \overline{SA+DA+CC+BC+HD+OPC}+1 \\
&= \overline{[00H]+[FFH]+[3DH]+[02H]+[6FH]+[80H]}+1 \\
&= \overline{[022DH]}+1 \\
&= [FDD2H]+1 \\
&= [FDD3H]
\end{aligned}
$$

上記の計算結果の下位１バイトの値をチェック・コードとします。

FCC＝［D3H］

（c）送信開始処理（レーベル：BUS_IDL）

バス・アイドル状態（BUSIDLF＝1）の場合，送信開始処理を行います。

本処理に入るとすぐに，INTP0トリガによるリアルタイム出力を許可とします。以降，INTP0トリガによるリアルタイム出力を禁止（INTC10割り込み要求信号でのみリアルタイム出力）とするまでは，他のIFUが送信を開始（自分がINTP0エッジ検出）すると，これに競合して送信を開始します。

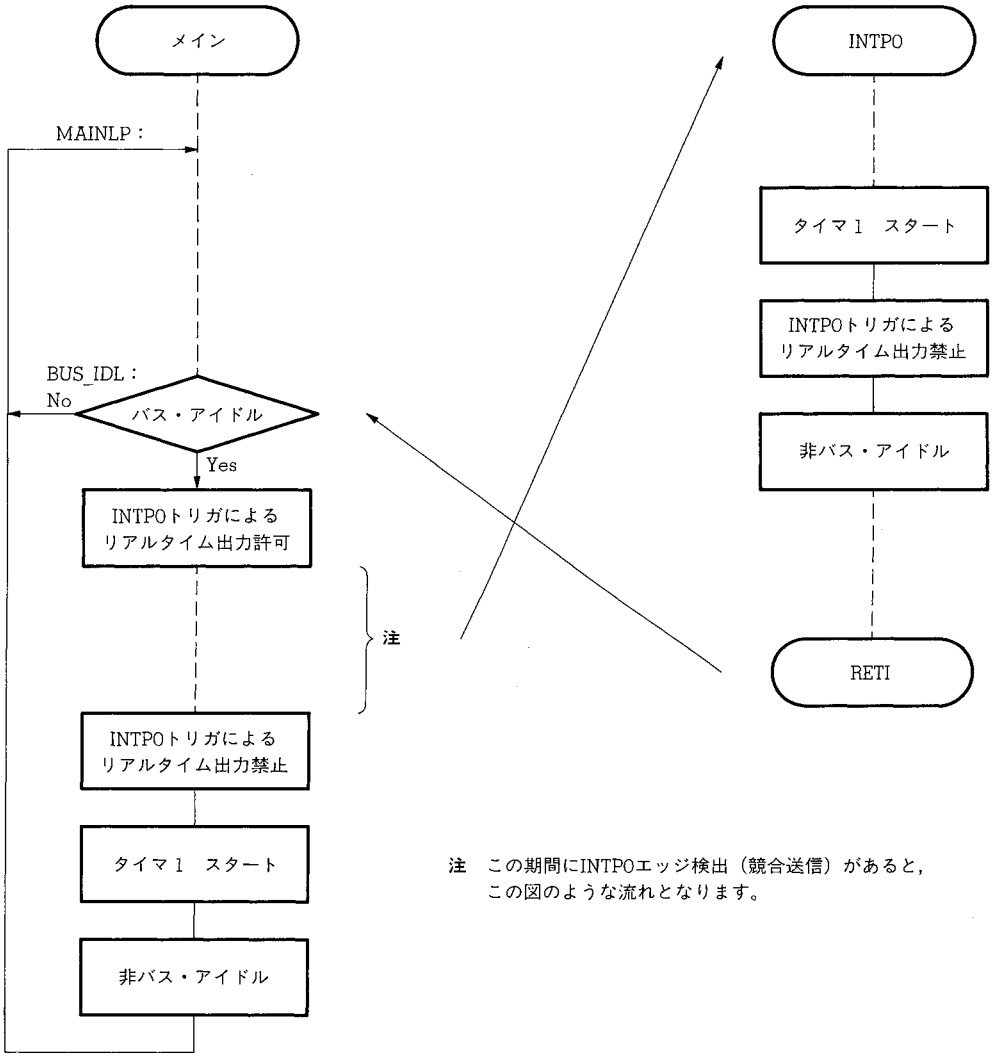
また，この期間に他のIFUが送信を開始しない場合は，タイマ１のカウントを開始し，非バス・アイドル状態として，送信を開始します。

INTP0トリガによって競合送信する場合は，INTP0割り込み処理で送信開始処理を行うことになり，残りのメインの本処理内には復帰せず，一律に本処理の先頭のバス・アイドル判定処理（BUS_IDL：）に復帰します。この場合すでに送信が開始され，バス・アイドル状態ではないので，送信開始処理には分岐せず，メイン・ループの先頭（MAINLP）へ分岐します。

この処理の流れを図11－29のフロー・チャートに示します。

11
214
218A
234
244

図11－29　送信開始処理の流れ



注　この期間にINTP0エッジ検出（競合送信）があると，
この図のような流れとなります。

この処理の先頭では，スタート・ビット検出処理（INTP0）で復帰先を判断できるよう，PCチェック・フラグ（PCCHKF）をセット(1)しています。

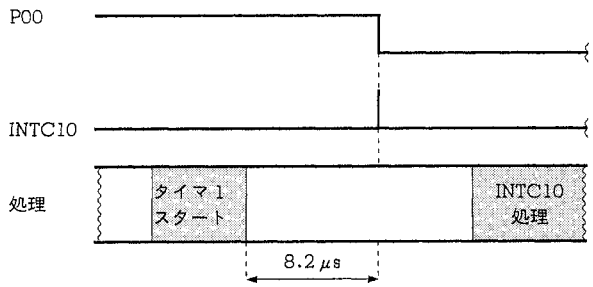
また本処理内では，タイマ1のコンペア・レジスタ（CR10）に2を設定しています。これは，1回目のTM1とCR10の一致後，INTC10処理内でCR10値を52 μsに相当する18に書き換える命令（MOV CR10, #18）を実行する前に，2回目の一致信号が発生してしまわないよう，次の時間（必要時間）＋余裕時間を確保するためです。

必要時間＝INTC10割り込み処理受け付け時間（18クロック）
＋MOV CR10, #18の実行時間（5クロック）
≒3.9 μs

これにより，タイマ１をスタートしてから一発目の送信波形出力までの時間(図11－30参照)は，次のようになります。

$$\frac{1}{11.6736\times10^{6}/2/16}\times(2+1)\fallingdotseq8.2\,\mu s$$

図11－30　タイマ１起動から波形出力までのタイミング

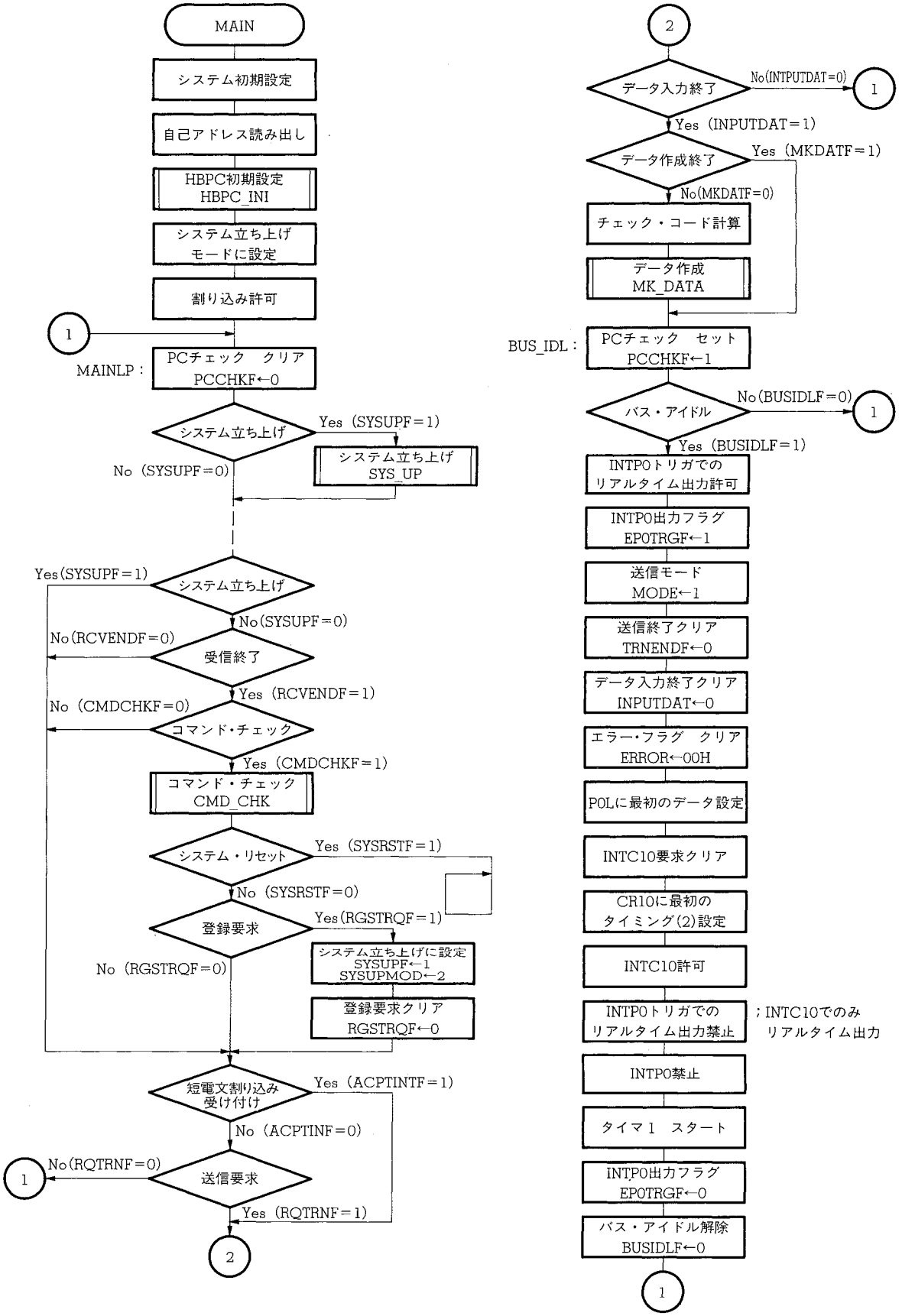


ただし，タイマの分周による１カウント目の誤差を考慮すると，実際には約5.5μsから8.2μsとなります。

なお処理の順序は，このプログラム例の順番としてください。

11
214
218A
234
244

(d) 詳細フロー・チャート


MAIN
システム初期設定
自己アドレス読み出し
HBPC初期設定
HBPC_INI
システム立ち上げ
モードに設定
割り込み許可
1
MAINLP：
PCチェック　クリア
PCCHKF←0
システム立ち上げ
Yes（SYSUPF＝1）
システム立ち上げ
SYS_UP
No（SYSUPF＝0）
Yes(SYSUPF＝1)
システム立ち上げ
No(SYSUPF＝0)
No(RCVENDF＝0)
受信終了
Yes（RCVENDF＝1）
No（CMDCHKF＝0）
コマンド・チェック
Yes（CMDCHKF＝1）
コマンド・チェック
CMD_CHK
システム・リセット
Yes（SYSRSTF＝1）
No（SYSRSTF＝0）
登録要求
Yes(RGSTRQF＝1)
No（RGSTRQF＝0）
システム立ち上げに設定
SYSUPF←1
SYSUPMOD←2
登録要求クリア
RGSTRQF←0
短電文割り込み
受け付け
Yes（ACPTINTF＝1）
No（ACPTINF＝0）
No(RQTRNF＝0)
送信要求
Yes（RQTRNF＝1）
1
2
2
データ入力終了
No(INTPUTDAT=0)
1
Yes（INPUTDAT＝1）
データ作成終了
Yes（MKDATF＝1）
No(MKDATF=0)
チェック・コード計算
データ作成
MK_DATA
BUS_IDL：
PCチェック　セット
PCCHKF←1
バス・アイドル
No(BUSIDLF＝0)
1
Yes（BUSIDLF＝1）
INTP0トリガでの
リアルタイム出力許可
INTP0出力フラグ
EP0TRGF←1
送信モード
MODE←1
送信終了クリア
TRNENDF←0
データ入力終了クリア
INPUTDAT←0
エラー・フラグ　クリア
ERROR←00H
POLに最初のデータ設定
INTC10要求クリア
CR10に最初の
タイミング(2)設定
INTC10許可
INTP0トリガでの
リアルタイム出力禁止
；INTC10でのみ
リアルタイム出力
INTP0禁止
タイマ1　スタート
INTP0出力フラグ
EP0TRGF←0
バス・アイドル解除
BUSIDLF←0
1

(3) ホームバス・プロトコル制御プログラムのイニシャライズ処理

サブルーチン名：HBPC_INI

ホームバス・プロトコル制御プログラムのイニシャライズを行います。

(a) 送信波形出力ポートのイニシャライズ

ポート0の初期値はFFHとし，初期出力レベルをハイ・レベル（バス上は0レベルを出力）にします。

(b) 割り込みのイニシャライズ

このプログラム例では，表11－13に示すタイマ割り込みと，表11－14に示すエッジ割り込みを使用しています。

表11－13　使用するタイマ割り込み

割り込みソース	用途		CR10/CR30	周期	優先順位	処理形態	マスク設定
INTC10	送受信タイミングの生成		18	52 μs $f_{CLK}/16$	低位	ベクタ	動作時解除
INTC30	休止時間のカウント		111	9.79 ms $f_{CLK}/512$	低位	ベクタ	動作時解除
	休止期間のカウント		137	12.08 ms $f_{CLK}/512$	低位	ベクタ	動作時解除
	同期回復時間のカウント		75	208.3 μs $f_{CLK}/16$	低位	ベクタ	動作時解除
	受信タイム・アウト時間のカウント		113	10 ms $f_{CLK}/512$	低位	ベクタ	動作時解除
	システム立ち上げ時のウエイト	HBC	227	20 ms $f_{CLK}/512$	低位	フラグ・チェック	マスク状態
		IFU	113	10 ms $f_{CLK}/512$	低位	フラグ・チェック	マスク状態

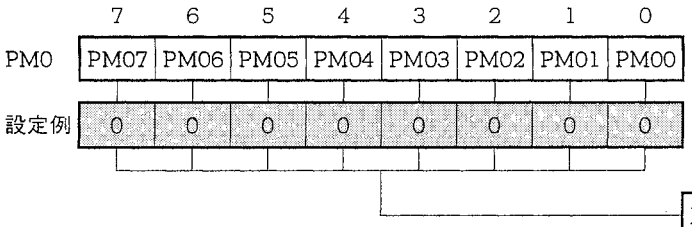
11
214
218A
234
244

表11－14　使用するエッジ割り込み

割り込みソース	用　　途	有効エッジ	優先順位	処理形態	マスク設定
INTP0	競合送信開始，受信開始	立ち下がり	高位	ベクタ	動作時解除
INTP1	送信要求発行	立ち下がり	低位	ベクタ	イニシャライズ時解除
INTP2	短電文割り込み要求発行	立ち下がり	低位	ベクタ	イニシャライズ時解除

(c) モード・レジスタ設定例

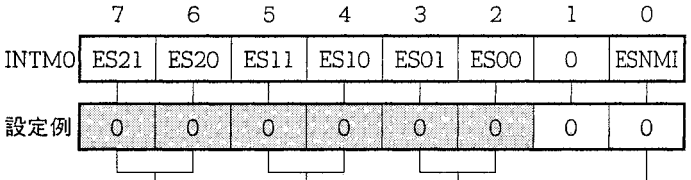
ポート0モード・レジスタ



PM0n	P0n端子入出力モード指定（n＝0-7）
00H	出力モード（出力バッファ・オン）
FFH	ハイ・インピーダンス状態(出力バッファ・オフ)
上記以外	設定禁止

11
214
218A
234
244

外部割り込みモード・レジスタ0



	7	6	5	4	3	2	1	0	
INTM0	ES21	ES20	ES11	ES10	ES01	ES00	0	ESNMI	（リセットで00H）
設定例	0	0	0	0	0	0	0	0	

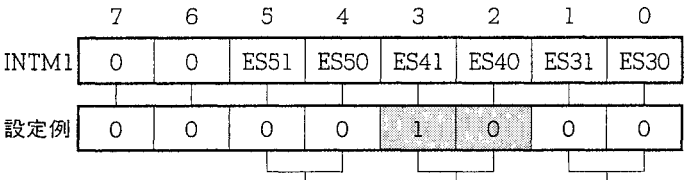
ESNMI	NMI端子入力，検出エッジ指定
0	立ち下がりエッジ
1	立ち上がりエッジ

ES01	ES00	INTP0端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES11	ES10	INTP1端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES21	ES20	INTP2端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

外部割り込みモード・レジスタ 1



	7	6	5	4	3	2	1	0	
INTM1	0	0	ES51	ES50	ES41	ES40	ES31	ES30	（リセットで00H）
設定例	0	0	0	0	1	0	0	0	×：操作しません

ES31	ES30	INTP3端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	立ち下がり，立ち上がり両エッジ

ES41	ES40	INTP4端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	INTC30を選択
1	1	設定禁止

ES51	ES50	INTP5端子入力，検出エッジ指定
0	0	立ち下がりエッジ
0	1	立ち上がりエッジ
1	0	設定禁止
1	1	

11
214
218A
234
244

優先順位指定フラグ・レジスタL

	7	6	5	4	3	2	1	0	
PR0L	CPR11	CPR10	CPR01	CPR00	PPR3	PPR2	PPR1	PPR0	（リセットでFFH）
設定例	1	1	1	1	1	1	1	0	

PR	優先順位指定フラグ
0	高優先順位レベル
1	低優先順位レベル

キャプチャ／コンペア・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
CRC1	0	0	0	0	CLR11	CM	CLR10	0	（リセットで00H）
設定例	0	0	0	0	0	0	1	0	

CLR10	TM1とCR10レジスタの内容一致でのクリア動作
0	禁止
1	許可

CLR11	CM	CR11レジスタの動作指定	TM1のクリア動作
0	0	コンペア・レジスタ	禁止
1	0		許可（TM1とCR11レジスタの内容一致のとき）
0	1	キャプチャ・レジスタ	禁止
1	1		許可（TM1の内容をCR11レジスタにキャプチャしたとき）

プリスケーラ・モード・レジスタ1

	7	6	5	4	3	2	1	0	
PRM1	PRS23	PRS22	PRS21	PRS20	0	PRS12	PRS11	PRS10	（リセットで00H）
設定例	0	0	0	0	0	0	0	0	

8ビット・タイマ／カウンタ1

PRS12	PRS11	PRS10	タイマ／カウンタ1のカウント・クロック周波数の指定
0	0	0	$f_{CLK}/16$[注]
0	0	1	
0	1	0	
0	1	1	$f_{CLK}/32$
1	0	0	$f_{CLK}/64$
1	0	1	$f_{CLK}/128$
1	1	0	$f_{CLK}/256$
1	1	1	$f_{CLK}/512$

8ビット・タイマ／カウンタ2

PRS23	PRS22	PRS21	PRS20	タイマ／カウンタ3のカウント・クロック周波数の指定
0	0	0	0	$f_{CLK}/16$
0	0	0	1	
0	0	1	0	
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	1	1	1	外部クロック（CI）

注　f_{CLK}：内部システム・クロック周波数（$f_{XX}/2$）

11
214
218A
234
244

割り込みマスク・レジスタL

	7	6	5	4	3	2	1	0	
MK0L	CMK11	CMK10	CMK01	CMK00	PMK3	PMK2	PMK1	PMK0	（リセットでFFH）
設定例	1	1	1	1	1	0	0	1	

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

(d) バッファ内容クリア

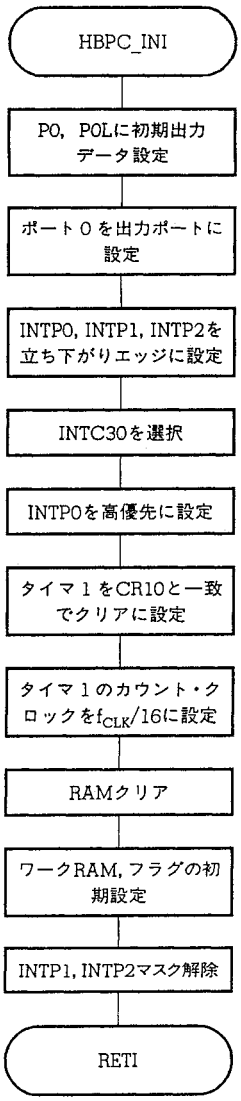
送信バッファ1（TRNBUF1），送信バッファ2（TRNBUF2），受信バッファ（RCVBUF），送信データ・バッファ1（TRNDAT1），送信データ・バッファ2（TRNDAT2）を0でクリアします。

各バッファの割り付け位置は，**11.3.3　内部RAMの割り付け**を参照してください。

(e) ワークRAM，ワーク・フラグのイニシャライズ

初期設定値は，**11.5.3　ワークRAM，ワーク・フラグ**を参照してください。

(f) 詳細フロー・チャート



11
214
218A
234
244

(4) システム立ち上げ処理

サブルーチン名：SYS_UP

システム立ち上げシーケンスを実行します。システム立ち上げシーケンスでやりとりするコマンドと通信シーケンスは，**11.4.9　コマンド処理**を参照してください。

システム立ち上げ処理は，HBC側とIFU側に分かれます。

本処理の先頭で自己アドレスを判定し，自己アドレスが00HならHBC，00H以外なら(このプログラム例では01H，02H) IFUの処理へ分岐します。

登録開始と登録要求コマンド・フレームは，同報電文として送信します。この場合の送信先のアドレス・グループと相手アドレスは，表11－15のようになります。

表11－15　アドレスとアドレス・グループの対応

アドレス	グループ	相手アドレス（DA）	プライオリティ
00H	0	00000001B	1
01H	4	00010000B	3
02H	2	00000100B	2

なお本処理は，HBCがIFUよりあとに立ち上がった場合や，違うコマンドを受信した場合などを想定していませんので，実際のシステムに応用する場合は，これらのことを考慮する必要があります。

(a) HBC側の処理

システム立ち上げモード番号(SYSUPMOD)に従って，それぞれのコマンド送受信シーケンスに分岐します。

システム立ち上げモード番号とコマンド送受信シーケンスの対応を表11－16に示します。

表11－16　HBC側のシステム立ち上げモード番号とコマンド送受信シーケンスの対応

モード番号	シーケンス
0	登録開始コマンド・フレーム送信開始
1	登録要求コマンド・フレーム受信開始
2	登録確認コマンド・フレーム送信開始
3	登録終了

(i) 登録開始コマンド・フレーム送信開始 (SYSUPMOD=0)

- 他のIFUが通信していないことをチェックするために10 msウエイトします。
  ウエイト中に受信データが検出された場合は，再度10 msウエイトします。
- 登録開始の送信データ・テーブル (RGSTSTRT) のデータと，送信アドレス・グループ・データを送信バッファ1 (TRNBUF1) へ転送します。この処理は，データ転送のマクロ (DATMOV) で行っています。
- メイン・ループで送信開始が行えるように，送信要求フラグ (RQTRNF)，データ入力終了フラグ (INPUTDAT) をセット(1)します。

(ii) 登録要求コマンド・フレーム受信開始 (SYSUPMOD=1)

- 登録開始コマンド・フレームの送信が終了するまでウエイトします。
- 登録要求コマンド・フレームの受信が遅れた場合に，次のシーケンスで登録要求コマンド・フレーム受信終了のチェックが正常に行えるように，受信終了フラグ (RCVENDF) をクリア(0)します。

(iii) 登録確認コマンド・フレーム送信開始 (SYSUPMOD=2)

- 登録要求コマンド・フレームの受信が終了するまでウエイトします。
- 登録確認の送信データ・テーブル (RGSTCNFM) のデータと，送信アドレス・グループ・データを送信バッファ1 (TRNBUF1) へ転送します。この処理は，データ転送のマクロ (DATMOV) で行っています。
- メイン・ループで送信開始が行えるように，送信要求フラグ (RQTRNF)，データ入力終了フラグ (INPUTDAT) をセット(1)します。

(iv) 登録終了 (SYSUPMOD=3)

- 登録確認コマンド・フレームの送信が終了するまでウエイトします。
- 同グループの他のアドレスのIFUから登録要求があるまでウエイトします。
  登録要求がある場合は，システム立ち上げモード番号 (SYSUPMOD) を“2”に書き換え，再度登録確認コマンド・フレームの送信開始を行います。
  登録要求がない場合は，10 msでタイム・アウトとします。
- 他グループの登録が済んでいない場合は，システム立ち上げモード番号を“0”に書き換え，再度登録開始コマンド・フレームの送信から始めます。
  全グループの登録が終了した場合は，システム立ち上げモードを解除 (SYSUPF←0) します。

(b) IFU側の処理

HBC側の処理と同様に，システム立ち上げモード番号(SYSUPMOD)に従って，それぞれのコマンド送受信シーケンスに分岐します。

11
214
218A
234
244

システム立ち上げモード番号とコマンド送受信シーケンスの対応を表11－17に示します。

表11－17　IFU側のシステム立ち上げモード番号とコマンド送受信シーケンスの対応

モード番号	シーケンス
0	登録開始コマンド・フレーム受信開始
1	登録要求コマンド・フレーム送信開始
2	登録確認コマンド・フレーム受信開始
3	登録終了

(i) 登録開始コマンド・フレーム受信開始（SYSUPMOD＝0）

受信許可(INTP0マスク解除)にして，1sウエイトします。ウエイト中に受信データが検出された場合は，登録開始コマンド・フレームの受信を開始したものとみなし，登録要求コマンド・フレーム送信開始シーケンスに移行します。

ウエイト中に受信データが検出されない場合は，自分が途中で立ち上がったものとみなし，登録要求コマンド・フレーム送信開始シーケンスに移行して，自分から登録要求を出します。

(ii) 登録要求コマンド・フレーム送信開始（SYSUPMOD＝1）

- 登録開始コマンド・フレームの受信が終了するまでウエイトします。

  登録開始コマンド・フレームが受信されず，自分から登録要求を出す場合は，受信終了フラグ(RCVENDF)が1にセットされたままなので，ウエイトせずに次の処理に入ります。
- 登録要求の送信データ・テーブル(RGSTRQST)のデータを，送信バッファ1(TRNBUF1)へ転送します。この処理は，データ転送のマクロ（DATMOV）で行っています。

(iii) 登録確認コマンド・フレーム受信開始（SYSUPMOD＝2）

登録要求では，同アドレス・グループ内なら複数のIFUが競合送信を行います。競合で負けた，あるいは競合に間に合わなかったIFUは，ここで他の登録終了まで待ちます。

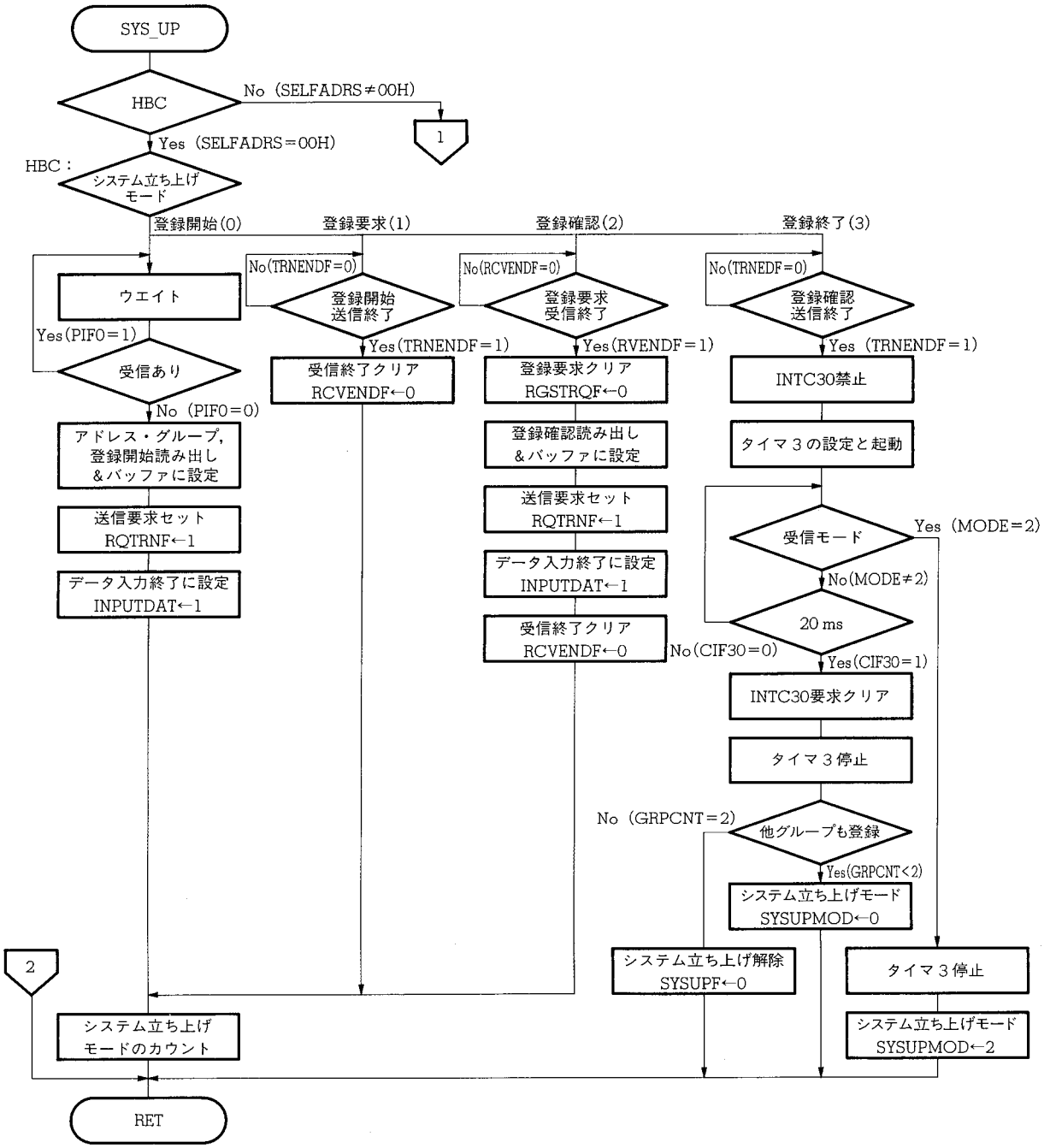
バス・アイドルになったら，その時点から10 ms間，バス・アイドルであることをチェックします。

10 msのバス・アイドル期間を確認したあとメインに戻り，再度登録要求を送信します。

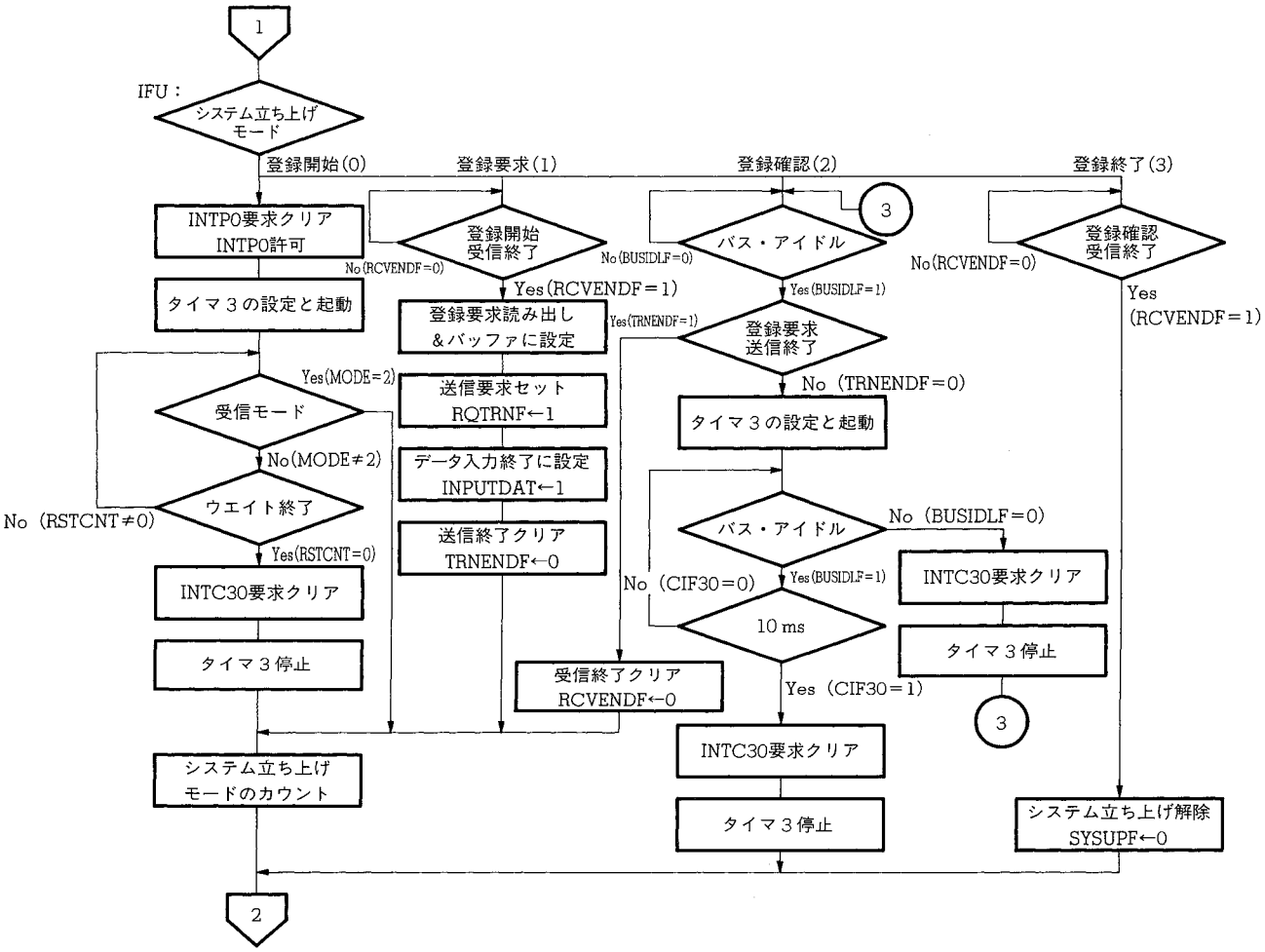
(iv) 登録終了（SYSUPMOD＝3）

システム立ち上げフラグ（SYSUPF）をクリア(0)して，システム立ち上げ処理を終了し，通常のメイン・ループに戻ります。

（c）詳細フロー・チャート



11
214
218A
234
244


1
IFU：
システム立ち上げ
モード
登録開始(0)
登録要求(1)
登録確認(2)
登録終了(3)
INTP0要求クリア
INTP0許可
タイマ３の設定と起動
受信モード
Yes(MODE=2)
No(MODE≠2)
ウエイト終了
No（RSTCNT≠0）
Yes(RSTCNT=0)
INTC30要求クリア
タイマ３停止
システム立ち上げ
モードのカウント
2
登録開始
受信終了
No(RCVENDF=0)
Yes(RCVENDF＝1)
登録要求読み出し
＆バッファに設定
送信要求セット
RQTRNF←1
データ入力終了に設定
INPUTDAT←1
送信終了クリア
TRNENDF←0
受信終了クリア
RCVENDF←0
バス・アイドル
No(BUSIDLF=0)
Yes(BUSIDLF=1)
3
登録要求
送信終了
Yes(TRNENDF=1)
No（TRNENDF＝0）
タイマ３の設定と起動
バス・アイドル
No（BUSIDLF＝0）
Yes(BUSIDLF=1)
No（CIF30＝0）
10 ms
Yes（CIF30＝1）
INTC30要求クリア
タイマ３停止
INTC30要求クリア
タイマ３停止
3
登録確認
受信終了
No(RCVENDF=0)
Yes
(RCVENDF＝1)
システム立ち上げ解除
SYSUPF←0

(5) コマンド・チェック処理

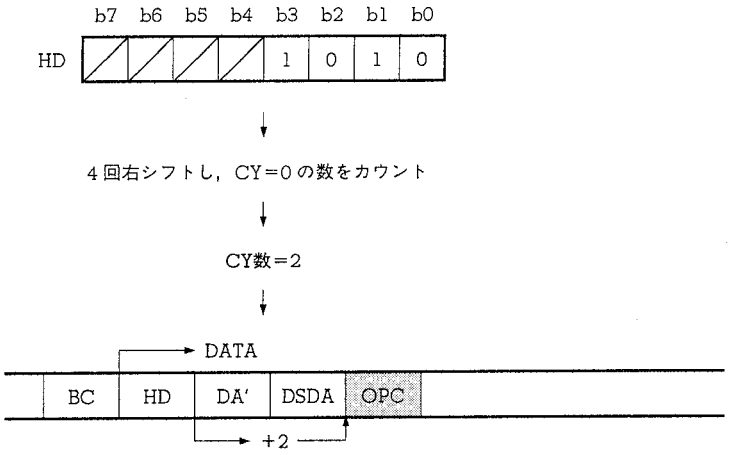
サブルーチン名：CMD_CHK

本処理では，受信データが登録要求コマンド(A1H)，あるいはシステム・リセット・コマンド(80H)であるかのチェックを行います。

直前の制御コード(CC)をチェックし，受信データがデータ・フレームの場合は，コマンド・チェックを行いません。

受信データがコマンド・フレームの場合は，データ（DATA）中のヘッダ・コード（HD）をチェックし，オペレーション・コード（OPC）の位置を検索します。ヘッダ・コードの下位4ビットの内の"1"の数をカウントし，ヘッダ・コードの置かれているアドレスに加算すると，オペレーション・コードの位置が求められます。

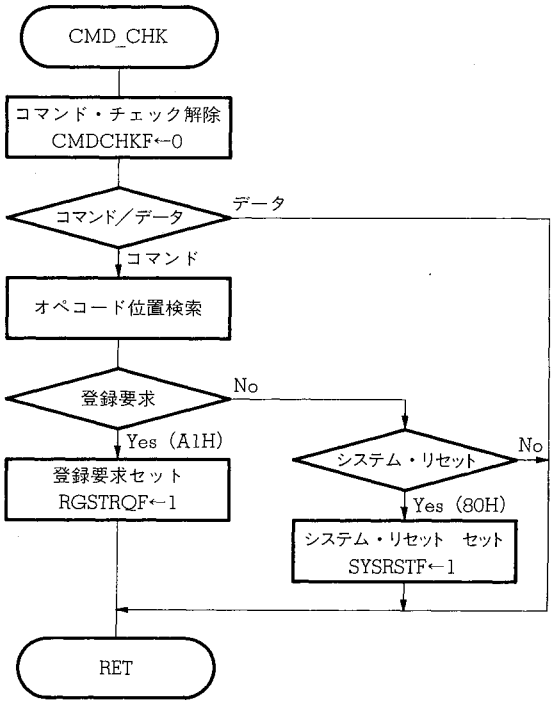
例　ヘッダ・コード（HD）が××××1010Bの場合



オペレーション・コード(コマンド)が登録要求コマンド(A1H)の場合は，登録要求フラグ(RGSTRQF)をセット(1)し，システム・リセット・コマンド(80H)の場合は，システム・リセット・フラグ(SYSRSTF)をセット(1)します。

11
214
218A
234
244

(a) 詳細フロー・チャート


CMD_CHK
コマンド・チェック解除
CMDCHKF←0
コマンド/データ
データ
コマンド
オペコード位置検索
登録要求
No
Yes (A1H)
登録要求セット
RGSTRQF←1
システム・リセット
No
Yes (80H)
システム・リセット　セット
SYSRSTF←1
RET

(6) INTC10割り込み処理

割り込み処理名：INTC10

この処理は，52 μs（1/2ビット）ごとに受け付けます。

送信の場合，最初にこの処理に入った時点でのCR10の値は2（(**c**) **送信開始処理**参照）ですので，CR10に18（52 μs）を設定します。2回目以降，および受信の場合はCR10＝18ですが，その判断をここに入れると，処理時間がきつくなりますので，毎回CR10←18の処理を行う形にしています。

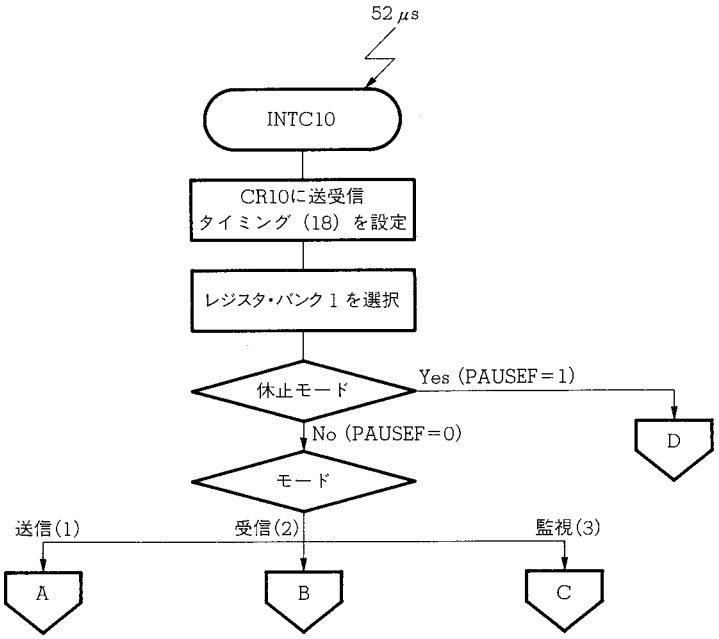
次の4つのモードに分けて送受信の処理を行います。

○送信モード
○受信モード
○監視モード
} モード番号（MODE）でモードを判定します。また，処理の基本構成は同様です。

○休止モード…フラグでモードを判断します。

送信，受信，監視処理では，カウンタ（DATCNT，BITCNT）によって，現在，通信フォーマット中のどの部分のデータを処理しているのか，およびどのビットを処理しているのかを表しています。DATCNT，BITCNTの値と対応する状態については，**11.5.3　ワークRAM，ワーク・フラグ**を参照してください。

INTC10割り込み処理受け付けから，各モードの分岐するまでのフロー・チャートを次に示します。



11
214
218A
234
244

(a) 送信，受信，監視モードの共通処理

次の場合は，データの送信あるいは受信から26 μs経過時点でP21のレベルを読み出します。

- 送信時の競合チェック
- 受信データの読み出し
- 被短電文割り込みチェック

26 μsのタイマのカウント値は9.5ですが，カウント値を読み出して26 μs相当の値と比較する方法で26 μsを検証していますので，読み出し，比較，分岐などの処理にかかる時間を考慮して，比較値は8としています。

$$26\times10^{6}\div\frac{1}{11.6736\times10^{6}/2/16}\fallingdotseq 9.5$$

(b) 送信モード

モード番号（MODE）＝1の場合，送信処理を行います。

ビット判断フラグ（BITF）によって，ビットの前半処理を行うか，後半処理を行うかを判断します。

(i) ビット前半の処理

次の処理を行います。

- 後半ビット・データの設定
- 競合チェック
- 短電文割り込みチェック
- 被短電文割り込みチェック
- 各データごとの処理

この処理に入るたびに，ビット・カウンタ（BITCNT）を－1していきます。

① 後半ビット・データの設定

後半ビット・データ（FFH）をバッファ・レジスタ（POL）へ設定します。

次のINTC10発生時には，P00（＋），P01（－）ともハイ・レベル（バス上は0レベル）を出力します。

② 競合チェック

優先コード（PR）および自己アドレス（SA）部（DATCNT＝1，2）で，スタート・ビットを除き1ビットごとに，送信データ（P0Lの内容）と受信データ（P21のレベル）を照合します。波形出力から26 μs経過点でP21のレベルを読み，送信データとレベルが異なっていれば（データが衝突していれば），競合負けしたとみなします。この場合，データ衝突フラグ（競合負けフラグ）（DTCLSHF）をセット(1)し，受信モードに設定します（MODE←2）。

なお，競合負けするまでの送信データは，競合負けして受信モードに移行することを考慮し，一時受信バッファ(RCVDAT)に格納します(スタート・ビット，パリティ・ビット，ストップ・ビットを除きます)。ビット・データの一時受信バッファへの格納方法は，**(c) 受信モード**を参照してください。

③ 短電文割り込みチェック

データ(DATA)部(DATCNT=6)で，長電文(優先コード・ワークPRCODE=××××××11B)の場合に行います。

パリティ・ビット（BITCNT＝2）の位置で短電文割り込み要求の有無をチェックします。短電文割り込み要求がある（RQINTF＝1）場合は，短電文割り込み中フラグ（INTCHRF）をセット(1)し，割り込み電文の送信準備を行えるようTRNRDYF，CLRMDATFをセット(1)します。

④ 被短電文割り込みチェック

データ(DATA)部(DATCNT=6)で，長電文(優先コード・ワークPRCODE=××××××11B)の場合に行います。

ストップ・ビット（BITCNT＝1）の位置でP21のレベルを読み，P21がロウ・レベルならそれをブレーク信号とみなし，被短電文割り込みあり，P21がハイ・レベルなら被短電文割り込みなしと判断します。

短電文割り込みありの場合は，このあと割り込み電文と競合して送信が行えるように，送信準備フラグ（TRNRDYF）をセット(1)します。

またブレーク信号のあとは休止期間に入るため，フレーム終了とみなし，データ・カウンタ（DATCNT）に10を設定します。

⑤ 各データごとの処理

ビット・カウンタ（BITCNT)＝0の場合，1パケット終了とみなし，この処理を行います。

データ・カウンタ（DATCNT）の値に応じて，各処理へ分岐します。

INTC10処理から復帰時には，BITCNTに11（初期値）を設定し，DATCNTを＋1します。

11
214
218A
234
244

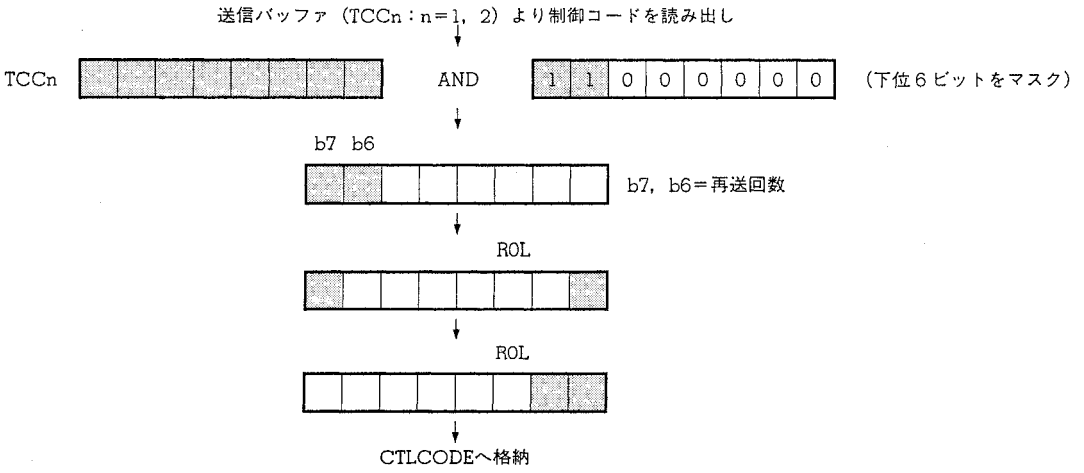
○優先コード（PR），自己アドレス（SA）

競合負けした場合に，引き続き受信動作に入れるよう，データを受信バッファ（PR，SA）に格納します。

○制御コード（CC）

ビット6，7の再送回数を演算して，ワークRAM（CTLCODE）に格納します。

〈再送回数のCTLCODEへの格納方法〉



○電文長コード（BC）

電文長カウンタ（CHRCNT）に0を初期設定します。電文長カウンタはアップ・カウンタとします。

○データ（DATA）

電文長をカウントします。電文長カウンタ（CHRCNT）値と送信バッファ内の電文長コード（TBCn：n＝1，2）値を比較し，電文の終了を判断します。

電文長が24を越えている場合，送信バッファ外のデータを送信してしまわないよう，比較値を最大の25とします。

DATAでは，電文長をカウント中はデータ・カウンタ（DATCNT）を更新しません（電文長カウント終了時に変更します）。

○チェック・コード（FCC）

電文が同報の場合は，ダミー(DMY)，ACK/NAK部はありませんので，ここでフレーム終了とするために，データ・カウンタ(DATCNT)に9を設定します(INTC10処理から復帰前に＋1されます)。

また送信要求フラグ（RQTRNF）や短電文割り込み要求フラグ（RQINTF）をクリアします。次の送信要求がある場合（RQTRN2F＝1）は，送信要求予備フラグ（RQTRN2F）をクリアし，送信要求フラグ（RQTRNF）はセットしたままとします。そして送信準備フラグ（TRNRDYF）をセットし，次の送信に備えます。

(ii) ビット後半の処理

次の処理を行います。

- ダミー部での送信停止
- 前半ビット・データ（送信波形データ，ブレーク・データ）の設定
- ACK/NAK受信待ち処理の起動
- フレーム終了処理

① ダミー部での送信停止

ダミー（DMY）部（DATCNT＝8）では，次に発生するINTC10より，リアルタイム出力（送信）を禁止します。

② 前半ビット・データの設定

短電文割り込み中でない（INTCHRF＝0）場合は，送信データ・バッファから送信波形データを読み出し，バッファ・レジスタ（POL）へ設定します。送信データ・バッファのアドレスは，送信バッファ・ポインタ（TRBFPNT）が示します。

短電文割り込み中（INTCHRF＝1）で，まだ短電文割り込みを受け付けていない（ACPTINTF＝0）場合は，ブレーク・データ（FEH）をバッファ・レジスタ（POL）へ設定します。

短電文割り込み中で，すでに短電文割り込みを受け付けている（ACPTINTF＝1）場合は，短電文割り込み中でない場合と同様の処理を行います。この場合，送信バッファ・ポインタの示すアドレスは，送信データ・バッファ２（TRNDAT2）のアドレスとなります。

③ ACK/NAK受信待ち処理の起動

ACK/NAK部（DATCNT＝9）で，１パケットの送信が終了（BITCNT＝11）した場合は，次のフレームよりACK/NAKデータを受信します。

受信モード（MODE←2）に設定し，送信に使用していたタイマ１を停止すると，以降ACK/NAK受信待ちとなります。受信データのスタート・ビットを検出するためINTP0を許可し，受信タイム・アウト（10 ms）をカウントするため，タイマ３を起動します。

この期間にスタート・ビット（INTP0立ち下がりエッジ）を検出すると，次回から発生するINTC10では受信処理を行います。

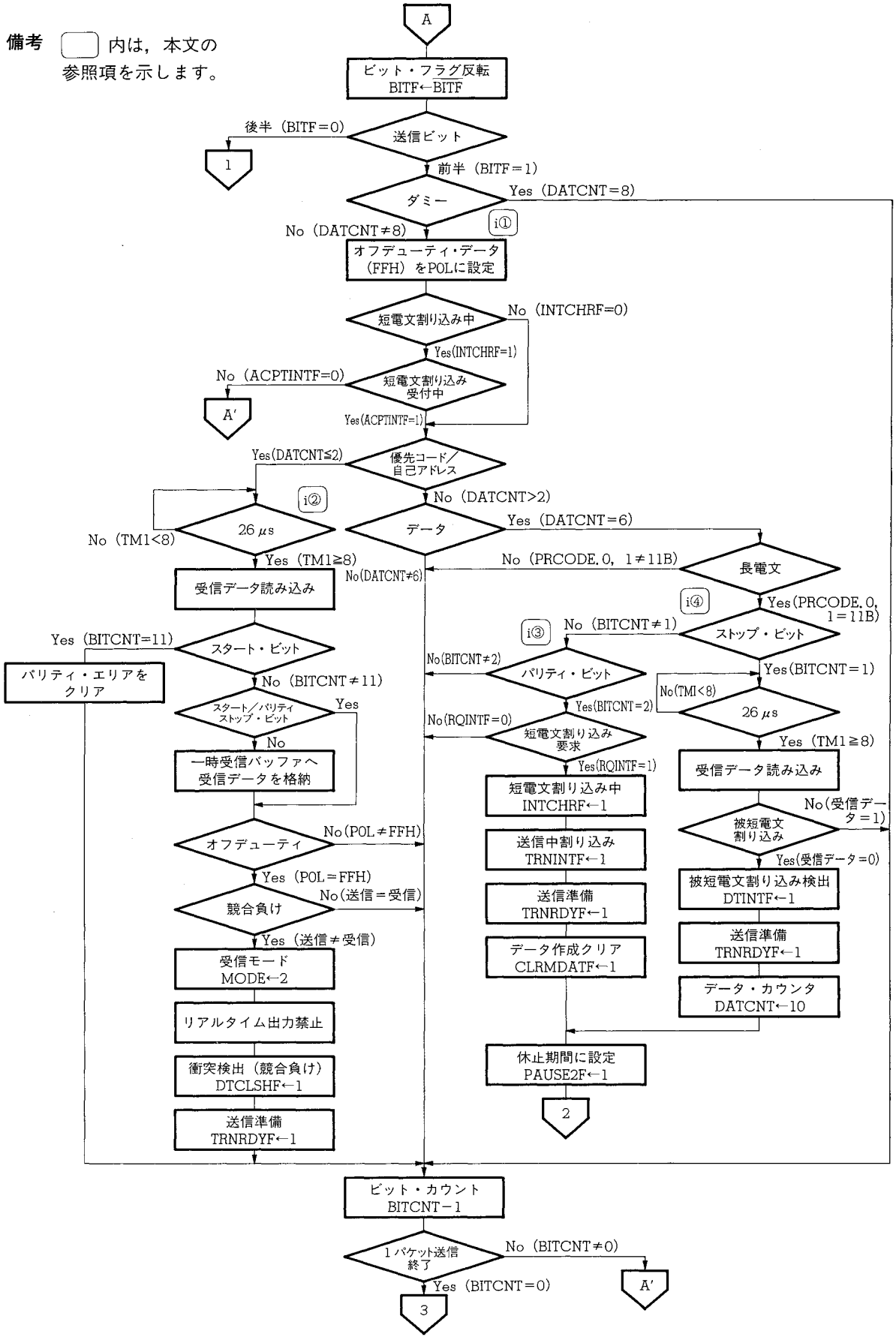
11
214
218A
234
244

④ フレーム終了処理

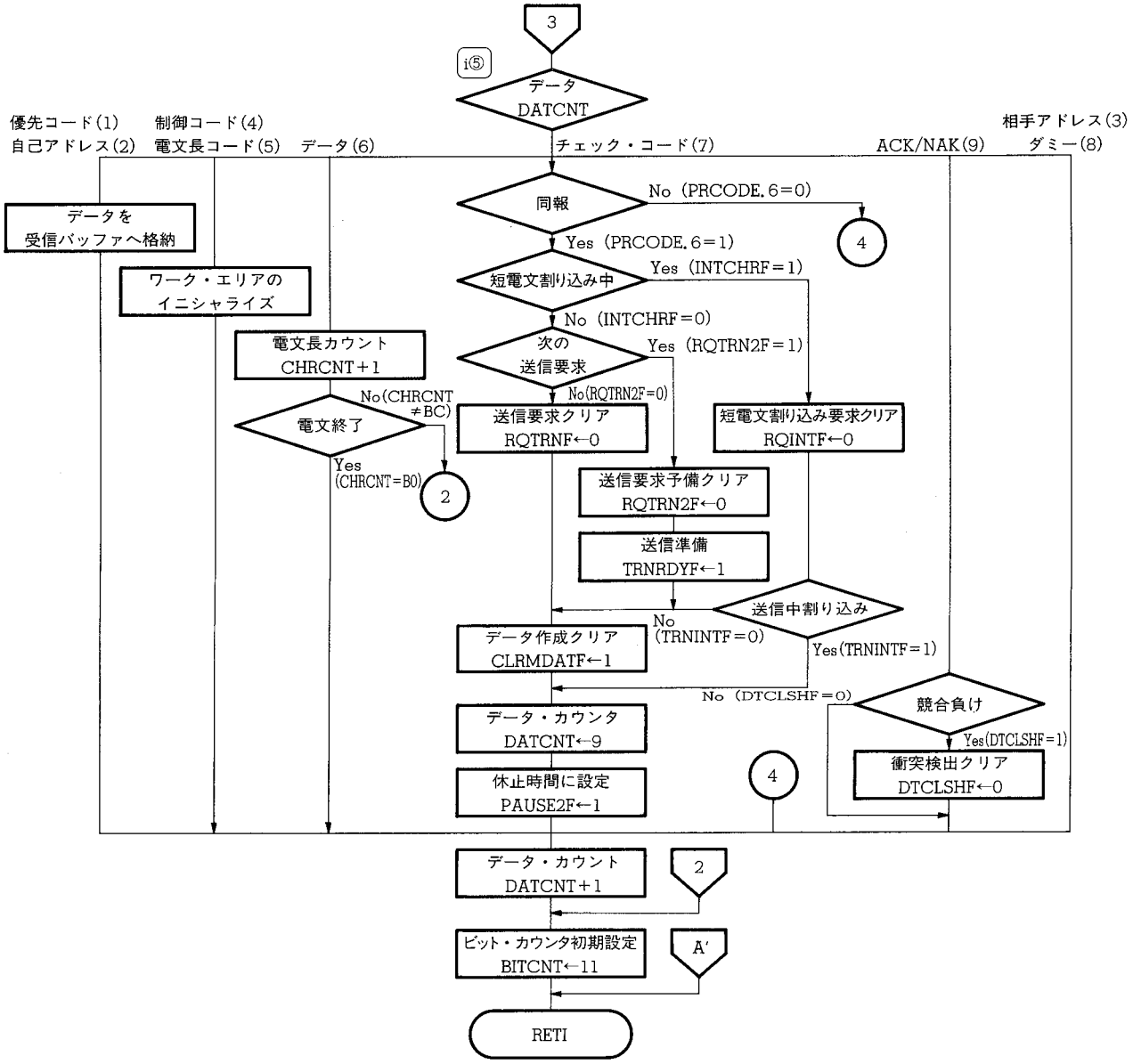
- 同報電文の送信(ACK/NAKを受信しない)，あるいは個別電文の受信(ACK/NAKを送信する）で，フレーム終了（DATCNT＝10）の場合，この処理に入ります。
- 送信準備フラグ(TRNRDYF)＝1なら，次の送信の準備を行います。データ入力終了フラグ（INPUTDAT）をセット(1)し，送信バッファ・ポインタを設定します。
- 終了した電文が送信で，短電文割り込み中(INTCHRF＝1)の場合が，短電文割り込みが終了したとみなし，短電文割り込み関係のフラグ（INTCHRF，ACPTINTF，TRNINTF）をクリア(0)します。
- 次回の競合送信に備えて，バッファ・レジスタ（POL）にスタート・ビット・データ（FEH）を設定します。
- 休止モードに設定します。次回に発生するINTC10の処理で，休止時間のカウントを開始します。

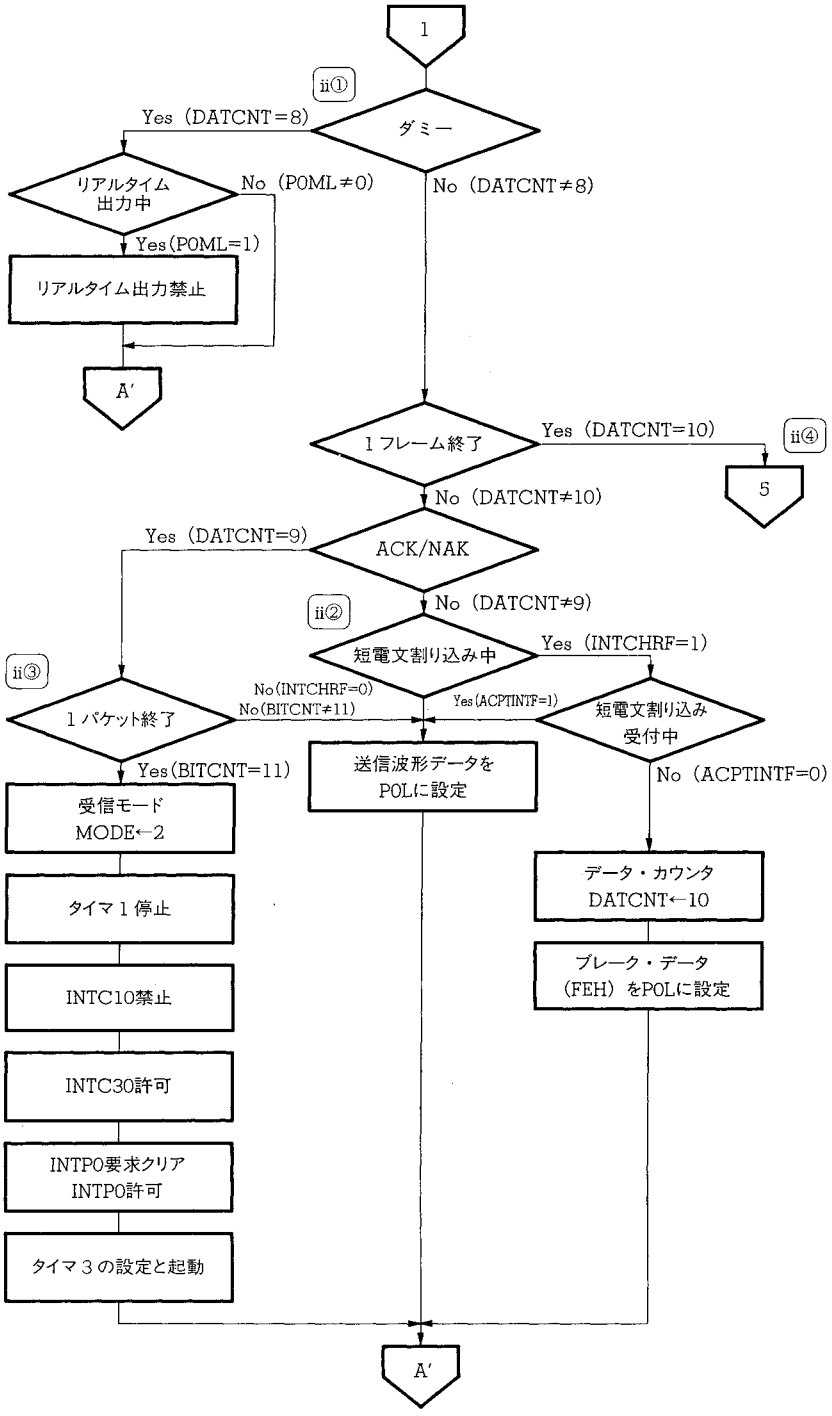
(iii) 詳細フロー・チャート

備考　□ 内は，本文の参照項を示します。

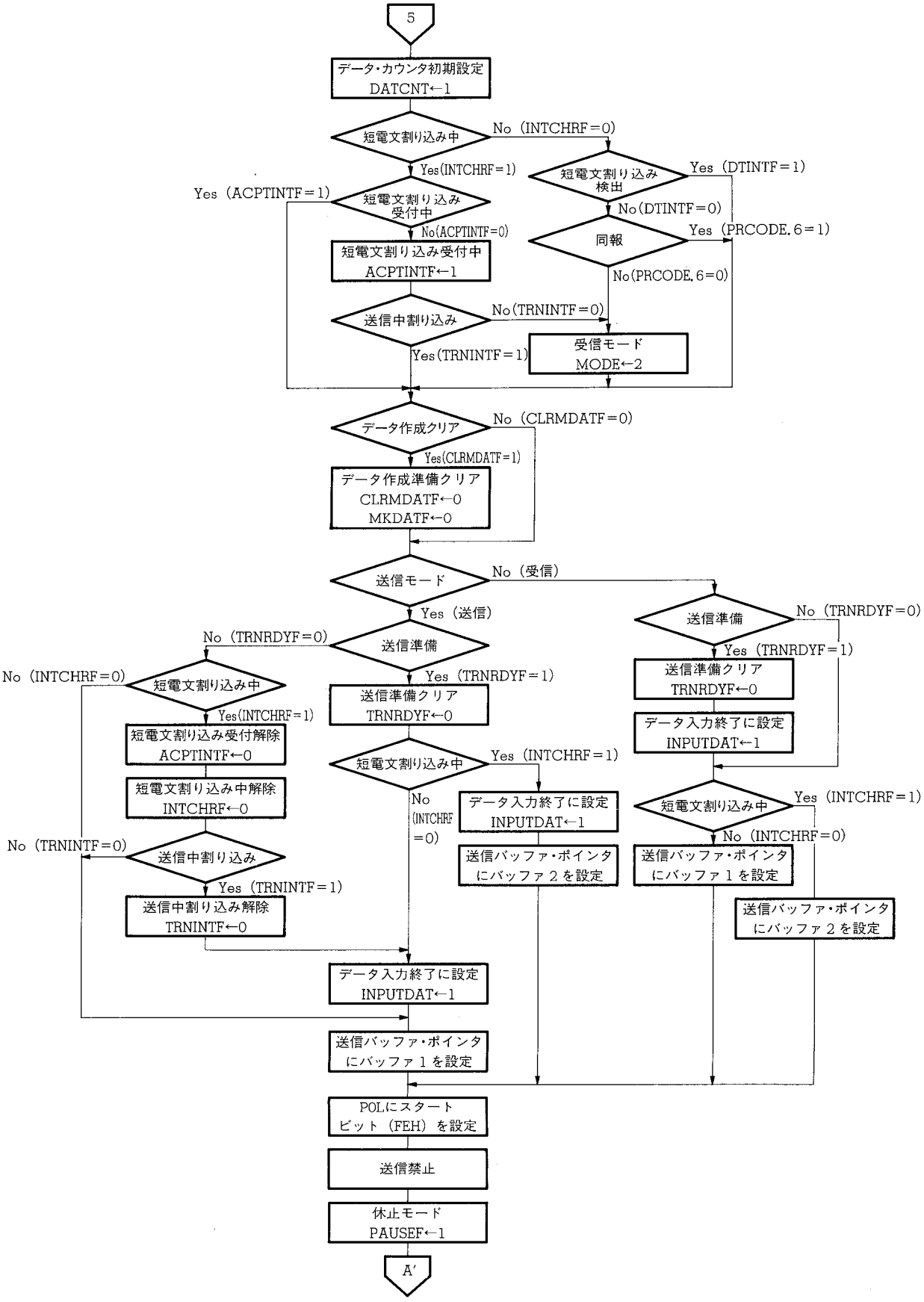


11
214
218A
234
244


3
i⑤
データ
DATCNT
優先コード(1)
自己アドレス(2)
制御コード(4)
電文長コード(5)
データ(6)
チェック・コード(7)
ACK/NAK(9)
相手アドレス(3)
ダミー(8)
同報
No (PRCODE.6=0)
4
Yes (PRCODE.6=1)
データを
受信バッファへ格納
短電文割り込み中
Yes (INTCHRF=1)
ワーク・エリアの
イニシャライズ
No (INTCHRF=0)
次の
送信要求
Yes (RQTRN2F=1)
電文長カウント
CHRCNT+1
No(RQTRN2F=0)
No(CHRCNT
≠BC)
電文終了
送信要求クリア
RQTRNF←0
短電文割り込み要求クリア
RQINTF←0
Yes
(CHRCNT=B0)
2
送信要求予備クリア
RQTRN2F←0
送信準備
TRNRDYF←1
送信中割り込み
No
(TRNINTF=0)
Yes(TRNINTF=1)
データ作成クリア
CLRMDATF←1
No (DTCLSHF=0)
競合負け
Yes(DTCLSHF=1)
データ・カウンタ
DATCNT←9
衝突検出クリア
DTCLSHF←0
休止時間に設定
PAUSE2F←1
4
データ・カウント
DATCNT+1
2
ビット・カウンタ初期設定
BITCNT←11
A'
RETI



11
214
218A
234
244


5
データ・カウンタ初期設定
DATCNT←1
短電文割り込み中
No（INTCHRF＝0）
Yes(INTCHRF＝1)
短電文割り込み
検出
Yes（DTINTF＝1）
No(DTINTF＝0)
同報
Yes（PRCODE.6＝1）
No(PRCODE.6＝0)
Yes（ACPTINTF＝1）
短電文割り込み
受付中
No(ACPTINTF=0)
短電文割り込み受付中
ACPTINTF←1
送信中割り込み
No(TRNINTF＝0)
Yes(TRNINTF＝1)
受信モード
MODE←2
データ作成クリア
No（CLRMDATF＝0）
Yes(CLRMDATF=1)
データ作成準備クリア
CLRMDATF←0
MKDATF←0
送信モード
No（受信）
Yes（送信）
送信準備
No（TRNRDYF＝0）
Yes（TRNRDYF＝1）
送信準備クリア
TRNRDYF←0
データ入力終了に設定
INPUTDAT←1
短電文割り込み中
Yes（INTCHRF＝1）
No（INTCHRF＝0）
短電文割り込み受付解除
ACPTINTF←0
短電文割り込み中解除
INTCHRF←0
送信中割り込み
No（TRNINTF＝0）
Yes（TRNINTF＝1）
送信中割り込み解除
TRNINTF←0
No
(INTCHRF
＝0)
送信バッファ・ポインタ
にバッファ２を設定
送信バッファ・ポインタ
にバッファ１を設定
POLにスタート
ビット（FEH）を設定
送信禁止
休止モード
PAUSEF←1
A'

(c) 受信モード

モード番号（MODE）＝2の場合，受信処理を行います。

ビット判断フラグ（BITF）によって，ビットの前半処理を行うか，後半処理を行うかを判断します。

(i) ビット前半の処理

次の処理を行います。

- 受信データの取り込みとパリティ・ビットの計算
- パリティ・チェックと短電文割り込みチェック
- ストップ・ビットのチェックと被短電文割り込みチェック
- チェック・サムの計算
- 各データごとの処理

① 受信データの取り込みとパリティ・ビット計算

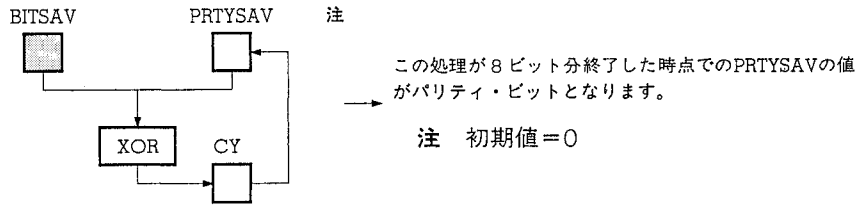
スタート・ビットを除く受信データを取り込みます（スタート・ビットの取り込みはINTP0割り込み処理で行います）。

受信データを読み出し，受信ビット・セーブ（BITSAV）エリアに保存します。

1パケット終了時は，パリティ・ビット／ストップ・ビットを除く8ビット・データを一時受信バッファ（RCVDAT）に格納します。

また1ビットの受信ごとにパリティを計算し，パリティ・ビット・セーブ・エリア（PRTYSAV）に格納します。

〈パリティ・ビットの計算方法〉

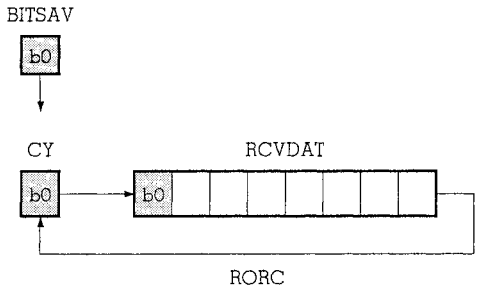


注　初期値＝0

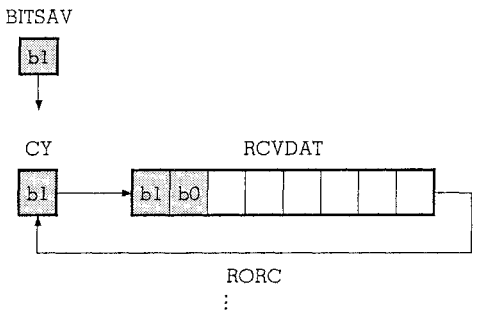
11
214
218A
234
244

〈ビット・データの一時受信バッファへの格納方法〉

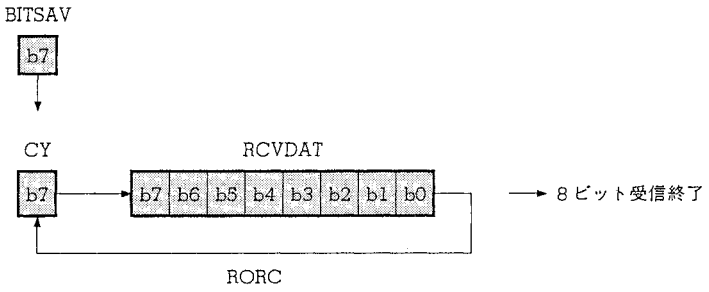
［ビット０の受信］



［ビット１の受信］



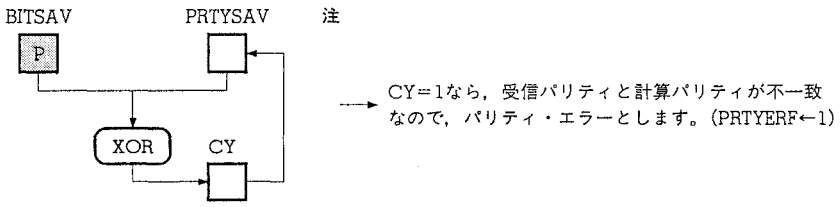
［ビット７の受信］



②　パリティ・チェックと短電文割り込みチェック

パリティ・ビット（BITCNT＝2）の位置で，パリティ・チェックと短電文割り込みチェックを行います。

〈パリティ・エラーのチェック方法〉



短電文割り込みチェックは，送信の場合と同様です。

③　ストップ・ビットのチェックと被短電文割り込みチェック

データ（DATA）部（DATCNT＝6）でない場合，ストップ・ビット位置（BITCNT＝1）でストップ・ビットのチェックを行います。受信レベルが1なら正常，受信レベルが0ならフレーミング・エラーとします（FRMERF←1）。

データ部の場合，ストップ・ビット位置では被短電文割り込みチェックを行います。受信ビット・セーブ（BITSAV）の値が0ならそれをブレーク信号とみなし，被短電文割り込みがあったとします。受信ビット・セーブの値が1なら，被短電文割り込みなしとします。

④　チェック・サムの計算

自己アドレス（DATCNT＝2）部からデータ（DASTA＝6）部の範囲で，1パケットの受信が終了した場合に行います。

受信中チェック・サム・セーブ・エリア（CHKSAV）（初期値は00H）の値に，一時受信バッファ（RCVDAT）の値を加算し，加算結果の下位8ビットをCHKSAVに格納します。

データ・カウンタがチェック・コード（FCC）部（DATCNT＝7）になるまでこれを繰り返します。

⑤　各データごとの処理

ビット・カウンタ（BITCNT）＝0の場合，1パケット終了とみなし，この処理を行います。

データ・カウンタ（DATCNT）の値に応じて，各処理へ分岐します。

INTC10処理から復帰時には，BITCNTに11（初期値）を設定し，DATCNTを＋1します。

また，優先コード（PR）からチェック・コード（FCC）までの受信データを受信バッファ（RCVBUF）に格納します。受信バッファのフォーマットは，**11.4.4　データ・バッファの構成**を参照してください。

11
214
218A
234
244

○相手アドレス（DA）

受信データ（RCVDAT）と自分のアドレス（SELFADRS）を比較し，自分宛ての電文であるかチェックします。電文が同報の場合は，アドレス・グループ・テーブルより，自分のアドレス・グループを読み出し，その値と比較します。

○電文長コード（BC）

受信した電文長コードが24を越えている場合，電文長を24として受信バッファ（BC_）に格納します。

受信中の電文が自分宛てでない場合は，モード番号（MODE）に3を書き込み，監視モードに設定します。次回発生するINTC10より，監視処理を行うことになります。

○チェック・コード（FCC）

受信途中に計算したチェック・サム（CHKSAV）の2の補数を，チェック・コードとして受信バッファ（FCC）に格納します。

受信チェック・コードと計算チェック・コードを比較し，不一致ならチェック・サム・エラーとします（CKSUMERF←1）。

電文が同報の場合はダミー（DMY），ACK/NAK部はありませんので，ここでフレーム終了とするために，データ・カウンタ（DATCNT）に9を設定します。

○ダミー（DMY）

受信エラーのチェックを行います。

図11－31　エラー・フラグ・エリアのフォーマット（受信）

	b7	b6	b5	b4	b3	b2	b1	b0
ERROR	TRNERF	NOANSERF	NAKERF	DMYDTERF	TMOUTERF	FRMERF	CKSUMERF	PRTYERF

受信エラー・フラグ

エラー内容（ERROR値）に応じて，ACK，NAK，無応答データ・テーブルの先頭アドレスを送信バッファ・ポインタ（TRBFPNT）に設定します。

データ・テーブルには，ACKデータ（06H），NAKデータ（15H），無応答データ（FFH）に対応する送信波形データ（11個）を定義してあります。ただし無応答の場合は，P00，P01ともハイ・レベルを出力しますので，スタート・ビットからストップ・ビットまでをすべてFFHとしています。

表11－18　受信エラーとERROR値，参照テーブルの対応

エラー有無	エラー内容（エラー・フラグ名称）	ERROR値	テーブル種類	Pr注
なし	—	××000000B	ACKデータ・テーブル（ACKDAT）	1
あり	フレーミング・エラー（FRMERF）	××××1××B	無応答データ・テーブル（NOANSDAT）	2
	ダミー・データ・エラー（DMYDTERF）	×××1××××B		
	パリティ・エラー（PRTYERF）	×××0×0×1B	NAKデータ・テーブル（NAKDAT）	3
	チェック・サム・エラー（CKSUMERF）	×××0×01×B		

注　優先順位（チェック順序）を表しています。たとえば，フレーミング・エラーとチェック・サム・エラーの両方が発生している場合は，無応答データを設定します。

○ACK/NAK

送信モードで電文を送信したあと，ACK/NAK受信部でこの処理に入ります。送信エラーをチェックし，エラーありの場合は再送回数をカウント・アップし，再送の準備を行います。

図11－32　エラー・フラグ・エリアのフォーマット（送信）

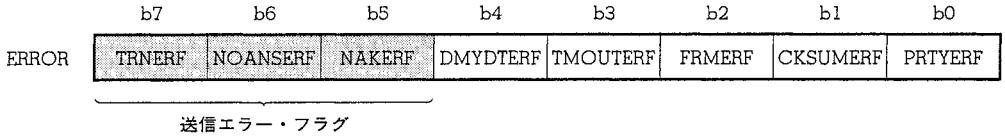


表11－19　送信エラーとERROR値，参照テーブルの対応

エラー有無	エラー内容（エラー・フラグ名称）	条　件	後処理	Pr注1
なし	—	受信データ　06H	要求フラグ・クリア→通常処理	2
あり	NAKエラー（NAKERF）	受信データ　15H	再送可否の判定	3
	無応答エラー（NOANSERF）	受信せず	再送可否の判定	1
	送信エラー（TRNERF）	再送不可	要求フラグ・クリア→通常処理注2	4

注1．優先順位（処理順序）を表しています。たとえば，無応答エラーの場合は，そのあとエラーあり／なしの判定はしません。

2．実際のシステムでは，このあとエラー・フラグ（TRNERF）をチェックし，何らかの処理を行ってください。

11
214
218A
234
244

(ii) ビット後半の処理

次の処理を行います。

- ダミー部での受信許可
- ACK/NAK送信処理の起動
- 短電文割り込み時のブレーク信号出力準備
- 受信タイム・アウト起動処理
- フレーム終了処理

① ダミー部での受信許可

ダミー（DMY）部（DATCNT＝8）の場合，ダミー・データ・エラーを検出できるようINTP0を許可しておきます。

② ACK/NAK送信処理の起動

ダミー部（DATCNT＝8）かつ1パケット終了（BITCNT＝11）の場合，ACK/NAKデータをバッファ・レジスタ（POL）に設定し，送信処理を起動します。

1パケットが終了していない（BITCNT≠11）場合は，ACK/NAK受信中ということになります。

③ 短電文割り込み時のブレーク信号出力準備

短電文割り込み中（INTCHRF＝1）で，まだブレーク信号を出力していない（ACPTINTF＝0）場合は，ブレーク信号出力準備を行います。ブレーク信号出力後，フレーム終了となるよう，データ・カウンタ（DATCNT）に10を設定します。そしてブレーク・データ（FEH）をバッファ・レジスタ（POL）に設定し，送信処理を起動します。

④ 受信タイム・アウト起動処理

1パケットの受信が終了（BITCNT＝11）した場合は，受信タイム・アウト時間（10 ms）のカウントを開始します。受信用のタイマ1を停止し，タイム・アウト・カウント用のタイマ3をスタートします。そしてタイム・アウト・カウント中のスタート・ビットを検出できるよう，INTP0を許可しておきます。

⑤ フレーム終了処理

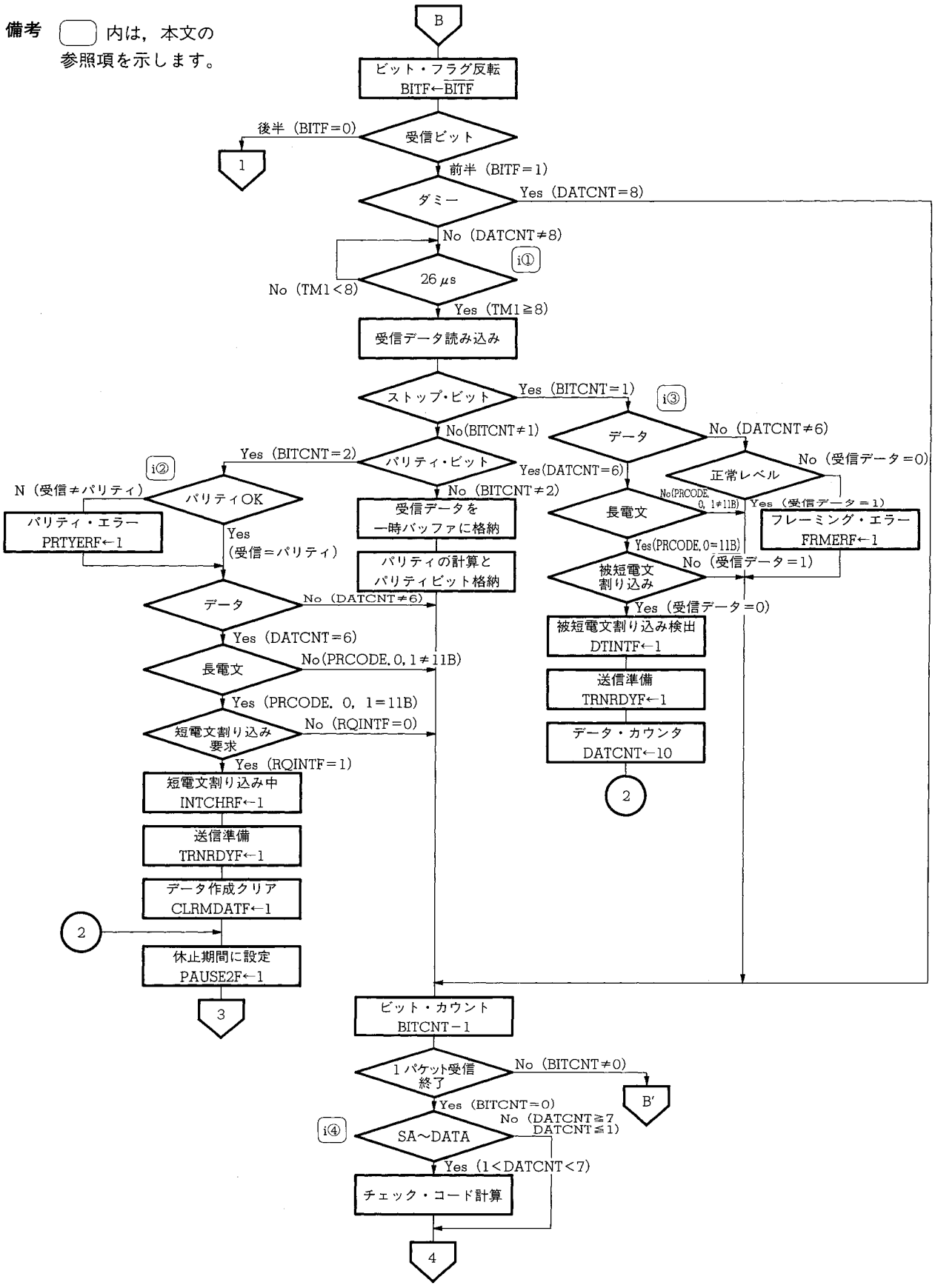
- 同報電文受信（ACK/NCKを送信しない），あるいは個別電文送信（ACK/NAKを受信する）の場合で，フレーム終了（DATCNT＝10）の場合はこの処理に入ります。
- 同報電文でない場合は，送信モード（MODE＝1）に戻します。

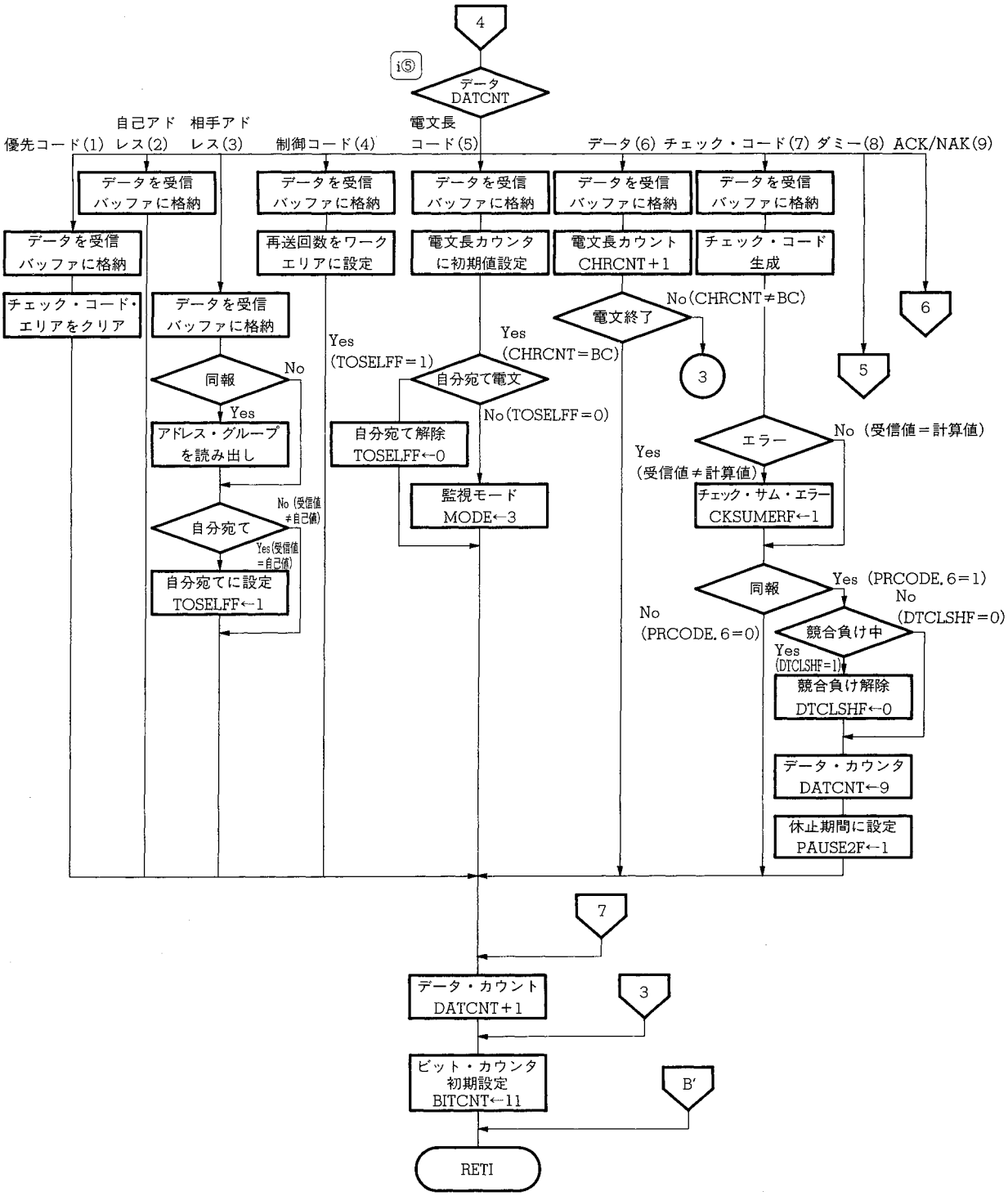
- 送信準備フラグ(TRNRDYF)＝1なら，次の送信の準備を行います。データ入力終了フラグ（INPUTDAT）をセット(1)し，送信バッファ・ポインタ（TRBFPNT）を設定します。
- 終了した電文が送信で，短電文割り込み中(INTCHRF＝1)の場合が，短電文割り込みが終了したとみなし，短電文割り込み関係のフラグ(INTCHRF, ACPTINTF, TRNINTF）をクリア(0)します。
- 次回の競合送信に備えて，バッファ・レジスタ（POL）にスタート・ビット・データ（FEH）を設定します。
- 休止モードに設定します。次回に発生するINTC10の処理で，休止時間のカウントを開始します。

11
214
218A
234
244

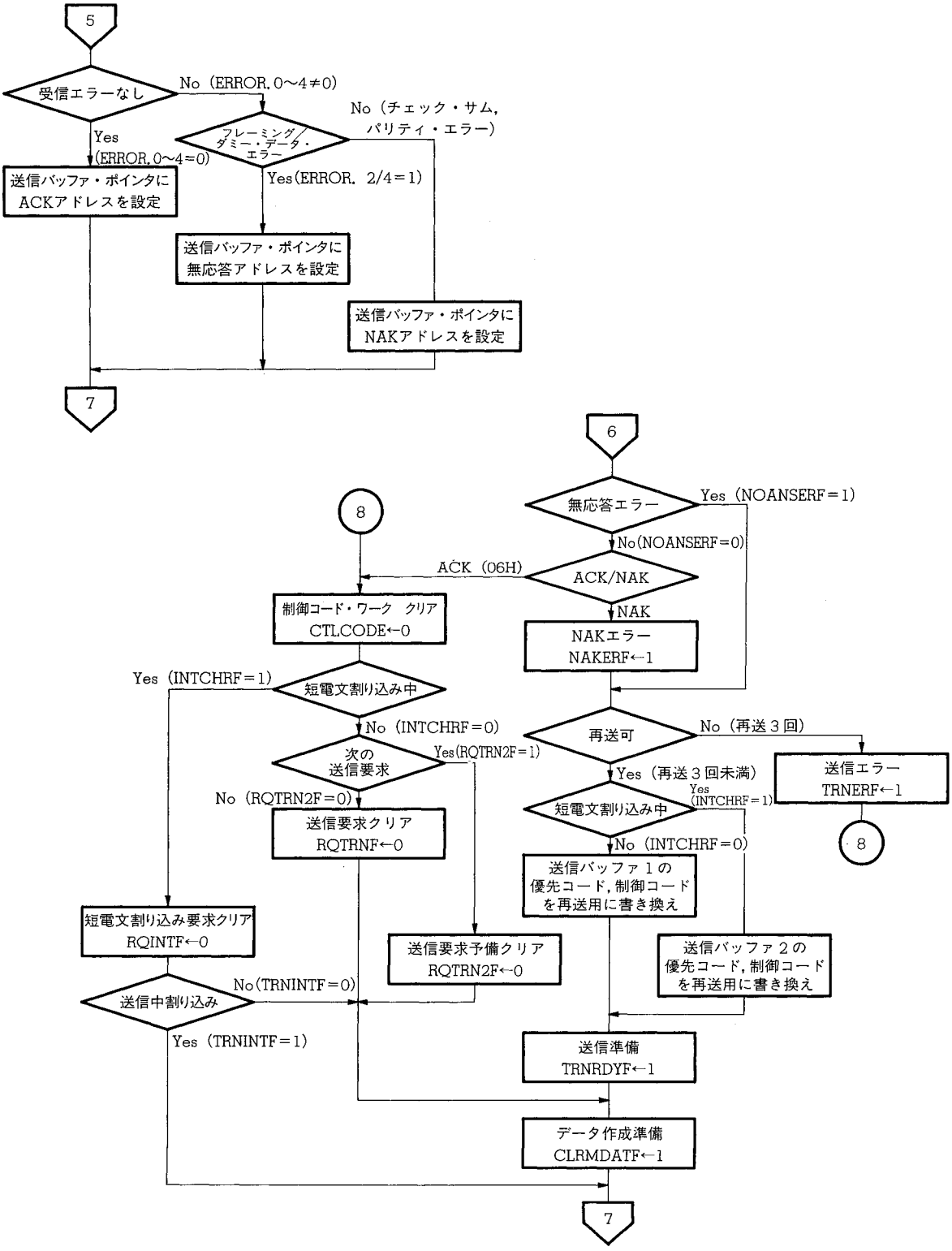
(iii) 詳細フロー・チャート

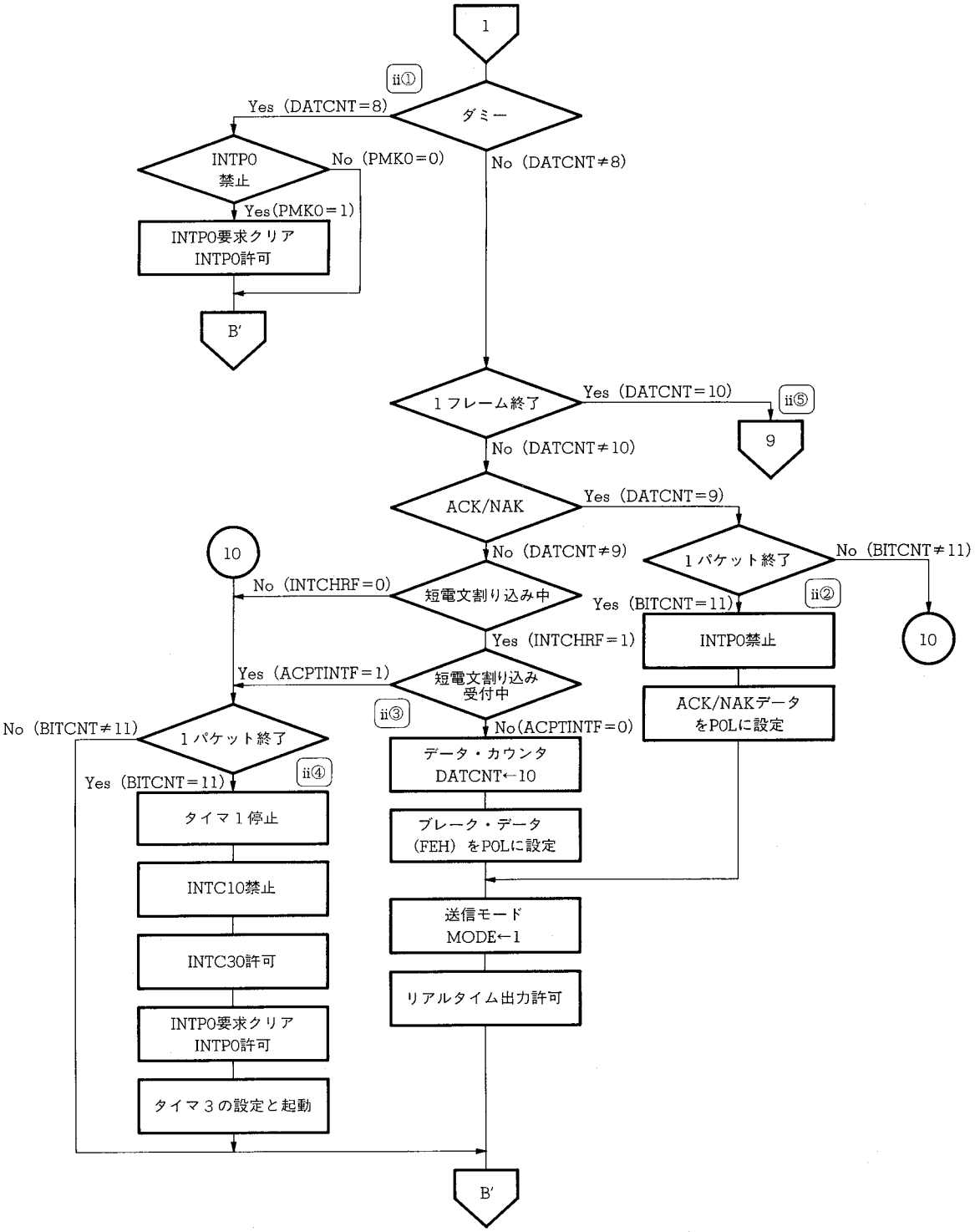
備考 ( ) 内は，本文の参照項を示します。



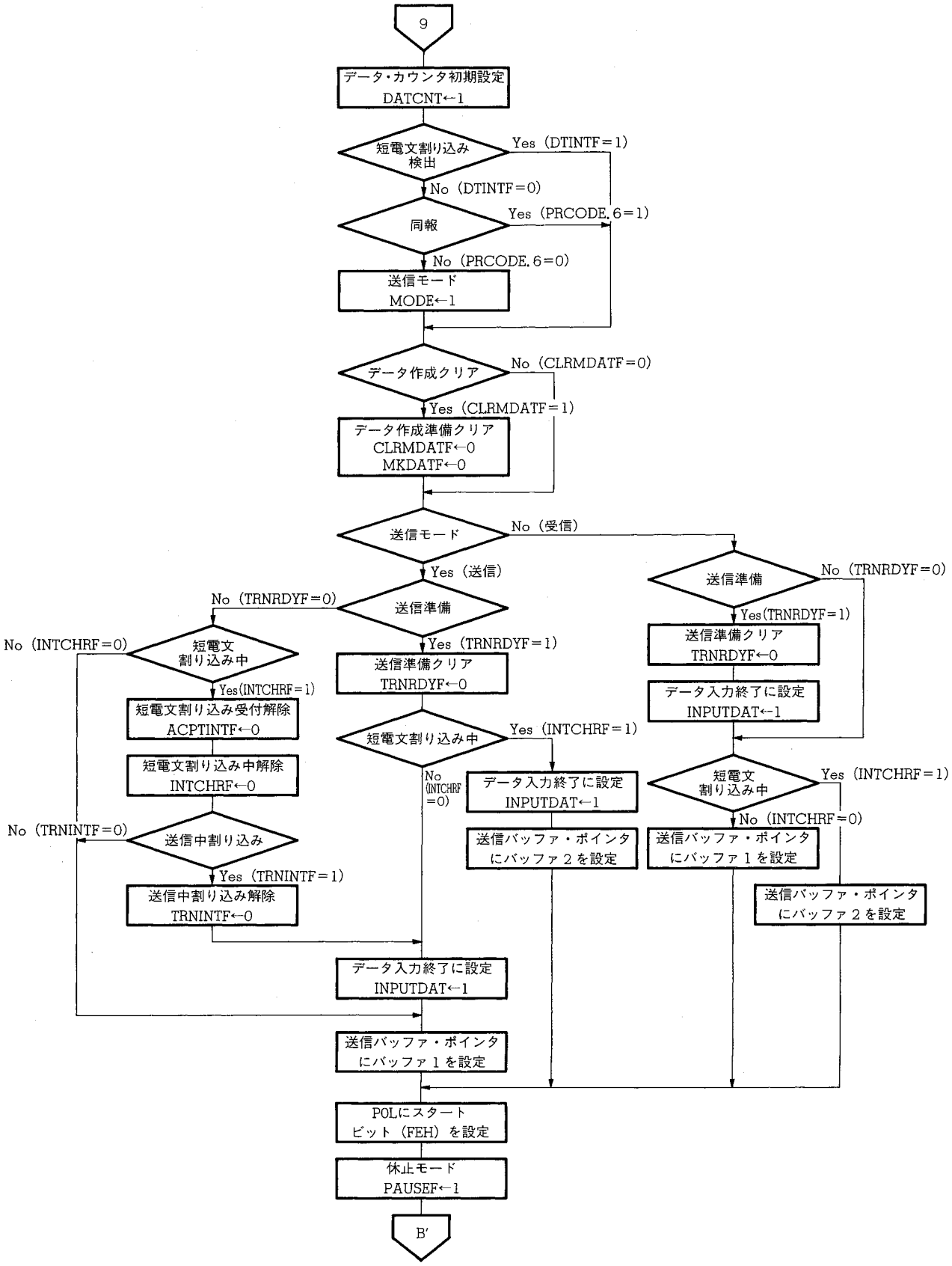


11
214
218A
234
244


5
受信エラーなし
No（ERROR, 0～4≠0）
No（チェック・サム，
パリティ・エラー）
フレーミング／
ダミー・データ・
エラー
Yes
(ERROR, 0～4＝0)
Yes(ERROR, 2/4＝1)
送信バッファ・ポインタに
ACKアドレスを設定
送信バッファ・ポインタに
無応答アドレスを設定
送信バッファ・ポインタに
NAKアドレスを設定
7
6
無応答エラー
Yes（NOANSERF＝1）
No(NOANSERF＝0)
ACK（06H）
ACK/NAK
NAK
NAKエラー
NAKERF←1
8
制御コード・ワーク　クリア
CTLCODE←0
Yes（INTCHRF＝1）
短電文割り込み中
No（INTCHRF＝0）
再送可
No（再送３回）
次の
送信要求
Yes(RQTRN2F＝1)
Yes（再送３回未満）
送信エラー
TRNERF←1
No（RQTRN2F＝0）
Yes
(INTCHRF＝1)
短電文割り込み中
送信要求クリア
RQTRNF←0
8
No（INTCHRF＝0）
送信バッファ１の
優先コード，制御コード
を再送用に書き換え
短電文割り込み要求クリア
RQINTF←0
送信要求予備クリア
RQTRN2F←0
送信バッファ２の
優先コード，制御コード
を再送用に書き換え
No(TRNINTF＝0)
送信中割り込み
Yes（TRNINTF＝1）
送信準備
TRNRDYF←1
データ作成準備
CLRMDATF←1
7



11
214
218A
234
244


9
データ・カウンタ初期設定
DATCNT←1
短電文割り込み検出
Yes（DTINTF＝1）
No（DTINTF＝0）
同報
Yes（PRCODE.6＝1）
No（PRCODE.6＝0）
送信モード
MODE←1
データ作成クリア
No（CLRMDATF＝0）
Yes（CLRMDATF＝1）
データ作成準備クリア
CLRMDATF←0
MKDATF←0
送信モード
No（受信）
Yes（送信）
送信準備
No（TRNRDYF＝0）
Yes（TRNRDYF＝1）
送信準備クリア
TRNRDYF←0
短電文割り込み中
No（INTCHRF＝0）
Yes(INTCHRF＝1)
短電文割り込み受付解除
ACPTINTF←0
短電文割り込み中解除
INTCHRF←0
送信中割り込み
No（TRNINTF＝0）
Yes（TRNINTF＝1）
送信中割り込み解除
TRNINTF←0
短電文割り込み中
Yes（INTCHRF＝1）
No (INTCHRF＝0)
データ入力終了に設定
INPUTDAT←1
送信バッファ・ポインタ
にバッファ２を設定
送信準備
No（TRNRDYF＝0）
Yes(TRNRDYF＝1)
送信準備クリア
TRNRDYF←0
データ入力終了に設定
INPUTDAT←1
短電文割り込み中
Yes（INTCHRF＝1）
No（INTCHRF＝0）
送信バッファ・ポインタ
にバッファ１を設定
送信バッファ・ポインタ
にバッファ２を設定
データ入力終了に設定
INPUTDAT←1
送信バッファ・ポインタ
にバッファ１を設定
POLにスタート
ビット（FEH）を設定
休止モード
PAUSEF←1
B′

(d) 監視モード

モード番号（MODE）＝3の場合，監視処理を行います。

受信電文が自分宛てでない場合，データ（DATA）部より監視モードに入ります。したがって，優先コード（PR）から電文長コード（BC）まで（DATCNT＝1～5）は，この処理を行うことはありません。

ビット判断フラグ（BITF）によって，ビットの前半処理を行うか，後半処理を行うかを判断します。

(i) ビット前半の処理

次の処理を行います。

- ●短電文割り込みチェック
- ●被短電文割り込みチェック
- ●各データ（一部）の処理

① 短電文割り込みチェック

送信モードと同様の処理となります。

② 被短電文割り込みチェック

送信モードと同様の処理となります。

③ 各データ（一部）の処理

ビット・カウンタ（BITCNT）＝0の場合，1パケット終了とみなし，この処理を行います。

○データ（DATA）

電文長をカウントし，電文終了（CHRCNT＝BC_）の場合，データ・カウンタ（DATCNT）を更新します。

○チェック・コード（FCC）

電文が同報の場合は，ダミー(DMY)，ACK/NAK部はありませんので，ここでフレーム終了とするため，データ・カウンタ（DATCNT）に9を設定します。

(ii) ビット後半の処理

次の処理を行います。

- ●短電文割り込み時のブレーク信号出力準備
- ●受信タイム・アウト監視の起動処理
- ●フレーム終了処理

11
214
218A
234
244

①　短電文割り込み時のブレーク信号出力準備

短電文割り込み中（INTCHRF＝1）で，まだブレーク信号を出力していない（ACPTINTF＝0）の場合は，ブレーク信号出力準備を行います。ブレーク信号出力後，フレーム終了となるよう，データ・カウンタ（DATCNT）に10を設定します。そしてブレーク・データ（FEH）をバッファ・レジスタ（POL）に設定し，送信処理を起動します。

②　受信タイム・アウト監視の起動処理

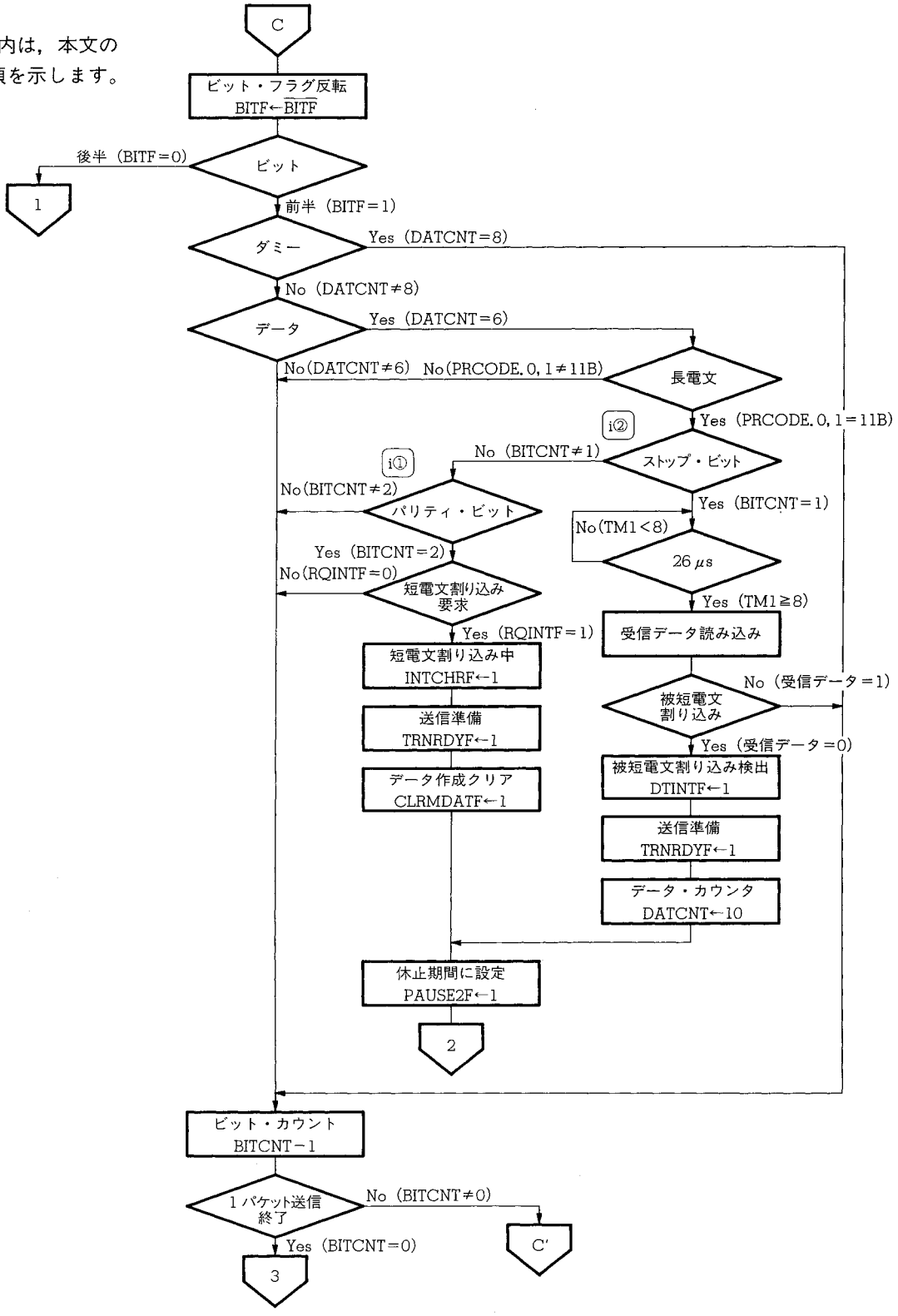
1パケットの監視が終了（BITCNT＝11）した場合は，受信タイム・アウト監視時間（10 ms）のカウントを開始します。監視用のタイマ1を停止し，タイム・アウト・カウント用のタイマ3をスタートします。そしてタイマ・アウト・カウント中のスタート・ビットを検出できるよう，INTP0を許可しておきます。

③　フレーム終了処理

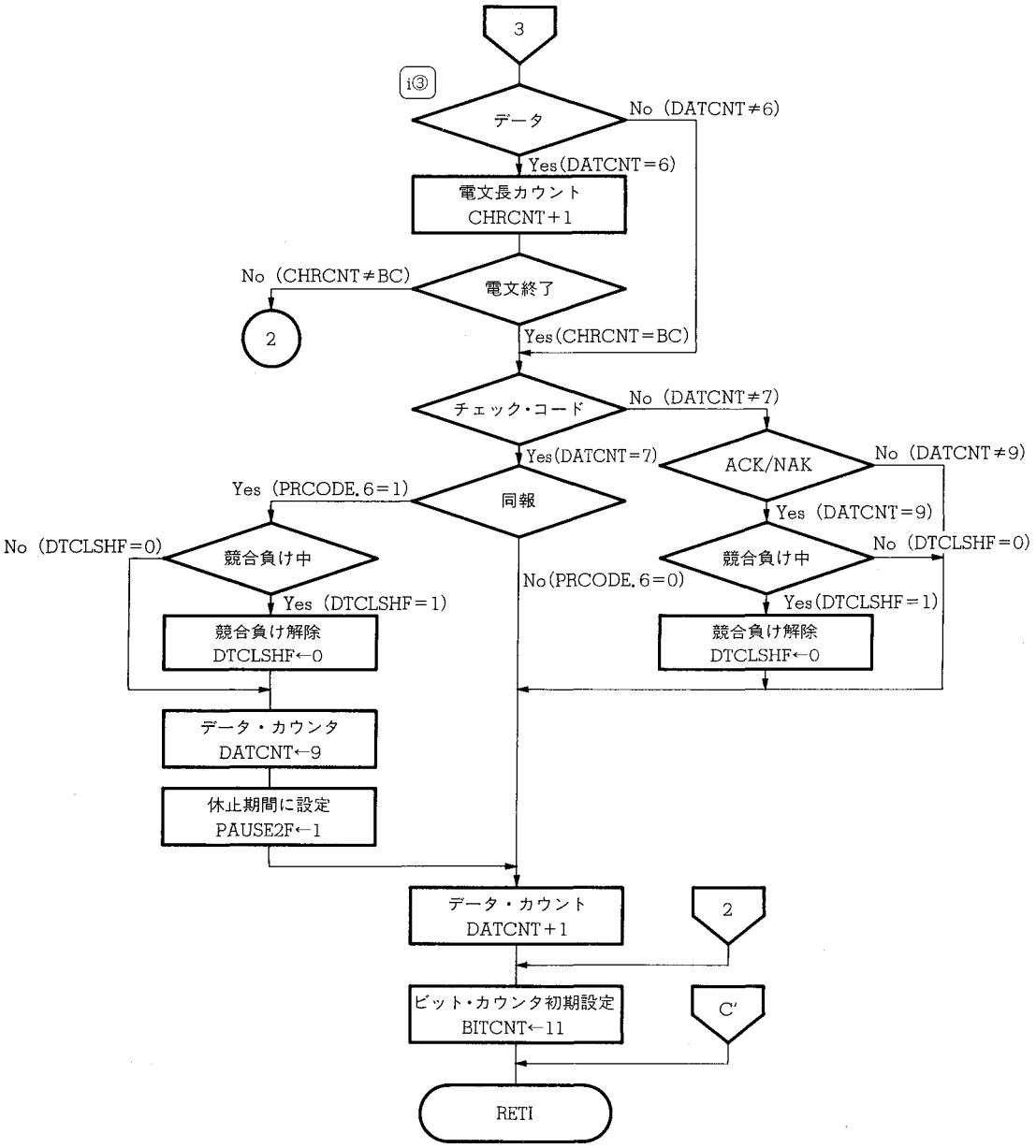
- 送信準備フラグ(TRNRDYF)＝1なら，次の送信の準備を行います。データ入力終了フラグ（INPUTDAT）をセット(1)し，送信バッファ・ポインタを設定します。
- 終了した電文が送信で，短電文割り込み中（INTCHRF＝1）の場合が，短電文割り込みが終了したとみなし，短電文割り込み関係のフラグ（INTCHRF，ACPTINTF，TRNINTF）をクリア(0)します。
- 次回の競合送信に備えて，バッファ・レジスタ（POL）にスタート・ビット・データ（FEH）を設定します。
- 休止モードに設定します。次回に発生するINTC10の処理で，休止時間のカウントを開始します。

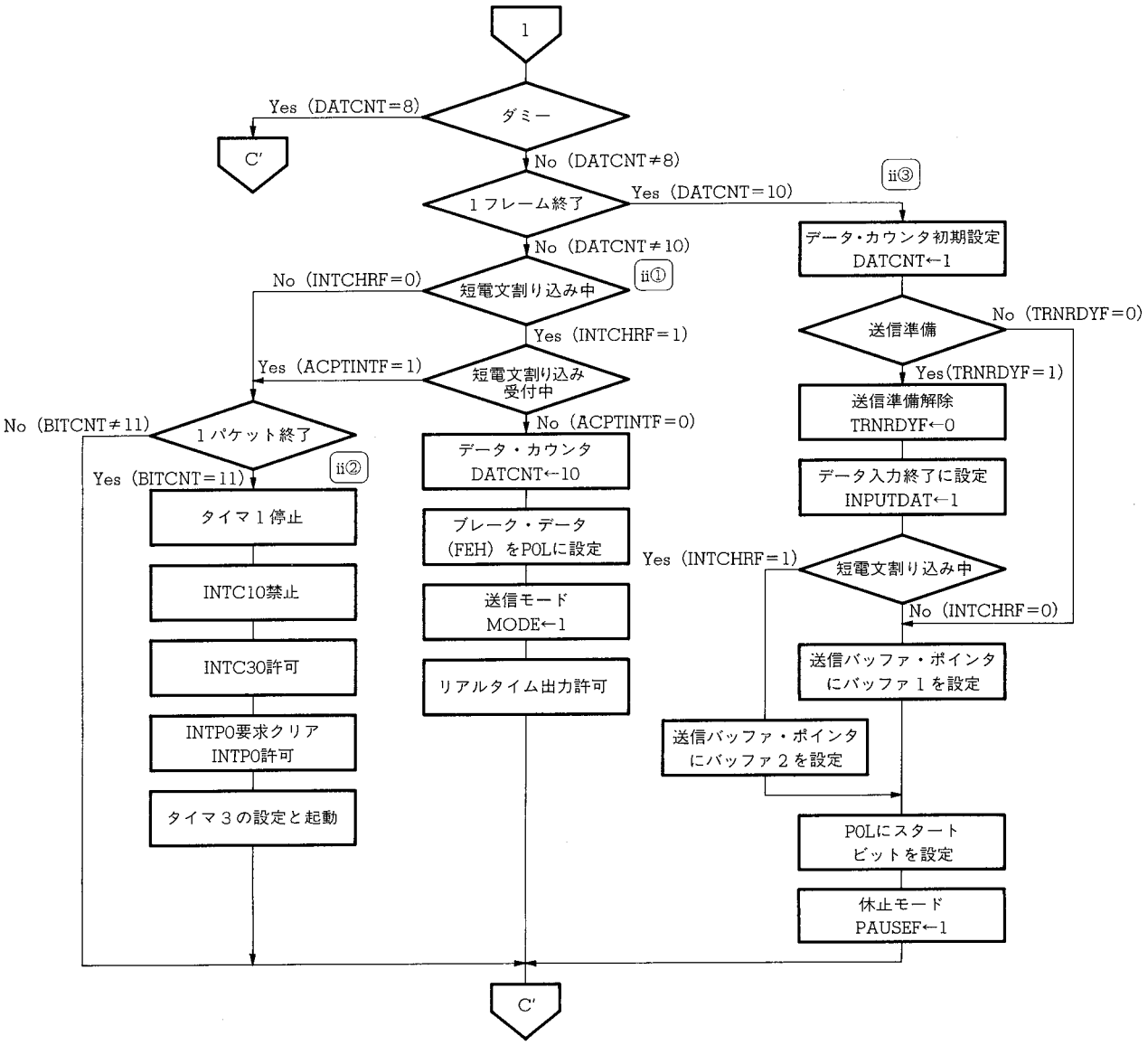
(iii) 詳細フロー・チャート

備考　□ 内は，本文の参照項を示します。



11
214
218A
234
244


3
i③
データ
No（DATCNT≠6）
Yes(DATCNT＝6)
電文長カウント
CHRCNT＋1
No（CHRCNT≠BC）
電文終了
2
Yes(CHRCNT＝BC)
チェック・コード
No（DATCNT≠7）
Yes(DATCNT＝7)
ACK/NAK
No（DATCNT≠9）
Yes（PRCODE.6＝1）
同報
Yes（DATCNT＝9）
No（DTCLSHF＝0）
競合負け中
No（DTCLSHF＝0）
No(PRCODE.6＝0)
Yes（DTCLSHF＝1）
競合負け中
Yes(DTCLSHF＝1)
競合負け解除
DTCLSHF←0
競合負け解除
DTCLSHF←0
データ・カウンタ
DATCNT←9
休止期間に設定
PAUSE2F←1
データ・カウント
DATCNT＋1
2
ビット・カウンタ初期設定
BITCNT←11
C'
RETI



11
214
218A
234
244

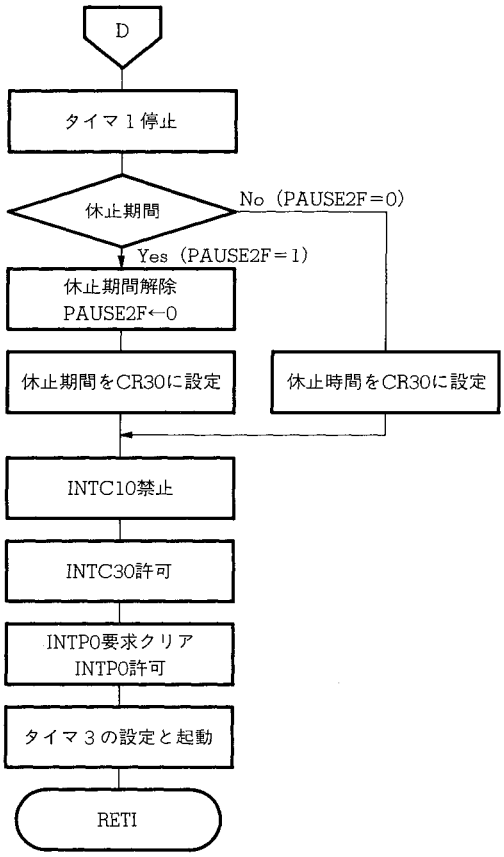
(e) 休止モード

休止フラグ（PAUSEF）＝1の場合は，休止時間のカウントを開始します。今まで使用していたタイマ1を停止し，休止時間カウント用のタイマ3を起動します。

終了した電文が同報，短電文割り込み，被短電文割り込みの場合は，休止期間（10 ms＋22ビット）をカウントします。この場合，休止期間フラグ（PAUSE2F）＝1となっていますので，PAUSE2Fによって，休止時間（10 ms），休止期間（10 ms＋22ビット）を選択します。

なお，この処理でカウントする時間は，同期回復に入るまでとします。

(i) 詳細フロー・チャート



(7) INTP0割り込み処理

割り込み処理名　：　INTP0

他のIFUが送信したスタート・ビットを検出すると，この処理に入ります。
処理は，次の２つに分かれます。

○競合送信開始
○スタート・ビット受信

なおINTP0割り込みは，多重割り込みする場合があります。

(a) 競合送信開始

次の条件で，この処理に入ります。

●同期回復中（SYNCF＝1），データ作成終了（MKDATF＝1）
●バス・アイドル（BUSIDLF＝1），データ作成終了

いずれもINTP0トリガによるリアルタイム出力が行われていない（EP0TRGF＝0）の場合は，この処理から抜け，スタート・ビット受信処理に入ります。

次の処理を行います。

○スタート・ビット送信
○プログラム・カウンタの変更

①　スタート・ビット送信

タイマ１を起動し，リアルタイム出力のモードをINTP0トリガによる出力禁止(INTC10でのみ出力）に設定します。

以降の処理は，メインの送信開始とINTC10の前半ビット処理を合わせた形となります。

ビット判断フラグ(BITF)を反転しますので，次回に発生するINTC10では，後半ビットの処理と行うことになります。したがって，後半ビット・データ(FFH)の設定や，送信バッファ・ポインタ（TRBFPNT）の更新などもこの処理内で行います。

②　プログラム・カウンタの変更

(2) ホームバス・プロトコル制御プログラムのメイン・ループ　(c) 送信開始処理にあるように，メイン処理の送信開始処理に入ってから本処理を行った場合(PCCHKF＝1），元

11
214
218A
234
244

のプログラム・カウンタへ復帰すると正常動作しませんので，メインの復帰先を送信開始前（バス・アイドルの判断（BUS_IDL：））に変更します。

またメイン処理で，すでに送信バッファ・ポインタ（TRBFPNT）を更新している場合は，本処理においても重複して更新していますので，送信バッファ・ポインタを１つ戻しています。

(b) スタート・ビット受信

次の条件で，この処理に入ります。

- バス・アイドルでない（BUSIDLF＝0）（同期回復を除く休止中も含む）
- データ作成が終了していない（MKDATF＝0）

受信データ・フレームのすべてのパケットのスタート・ビット検出で，次の処理を行います。

- ○スタート・ビットのレベル・チェック
- ○ダミー・データ・エラーのチェック
- ○受信開始処理

①　スタート・ビットのレベル・チェック

P21がロウ・レベルの場合は正常とみなし，引き続き受信開始処理を行います。

P21がハイ・レベルの場合はノイズを検出したとみなし，元の状態に復帰します。

②　ダミー・データ・エラーのチェック

ダミー部(DATCNT＝8)でこの処理に入った場合は，ダミー期間中にデータがあったと判断し，ダミー・データ・エラーとします（DMYDTERF←1）。

③　受信開始処理

フレーム最初のデータ受信の場合(DATCNT＝1)は，受信モードに設定します(MODE←2)。

フレーム最初のデータ受信でない場合(DATCNT≠1)は，すでに受信モードとなっていますので，受信モードへの設定処理は飛ばします。

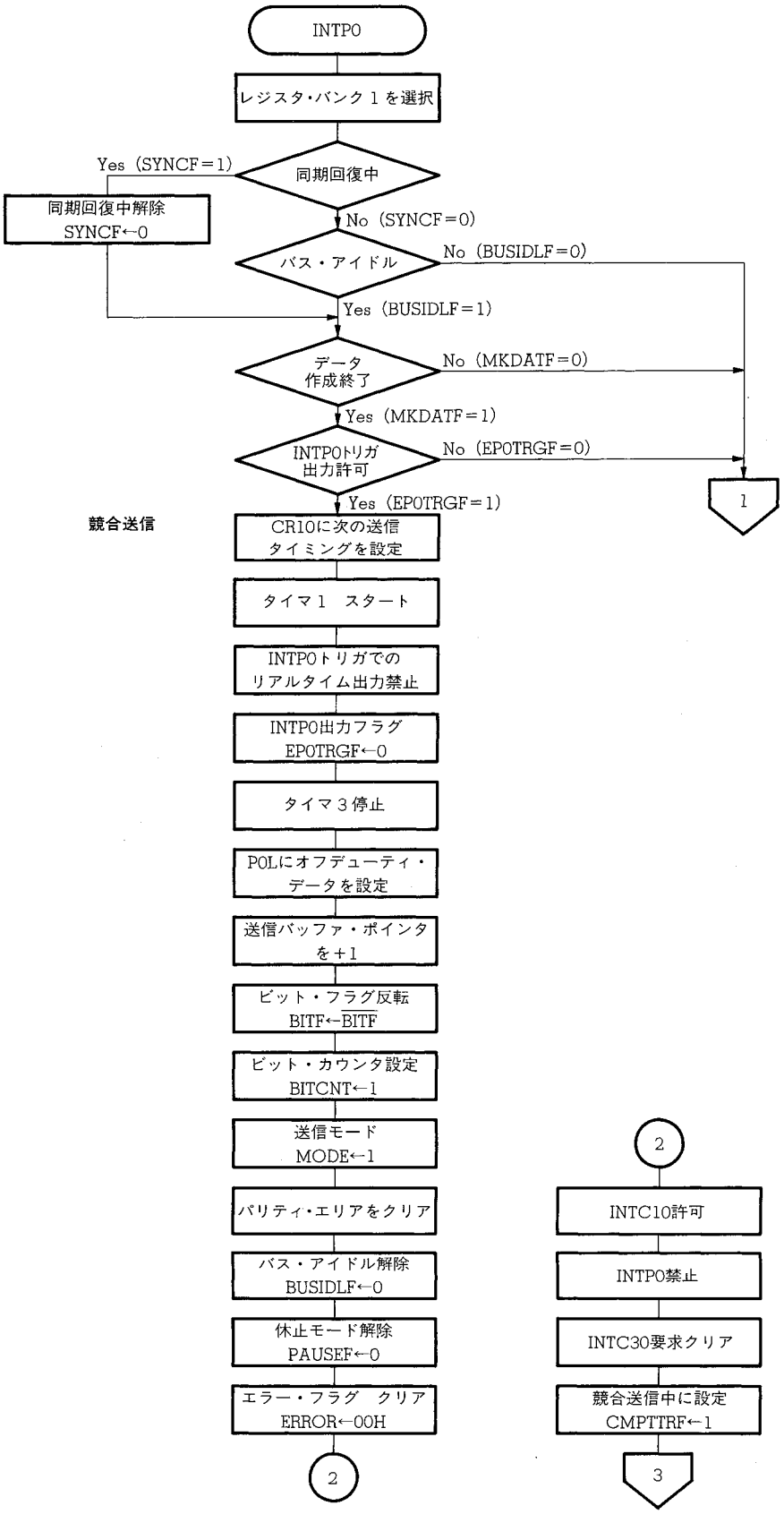
いずれの場合も，このあとタイマ１を起動し，受信開始とします。この際，ビット判断フラグ(BITF)を反転していますので，次に発生するINTC10では，後半ビットを受信することになります。

(c) 共通処理

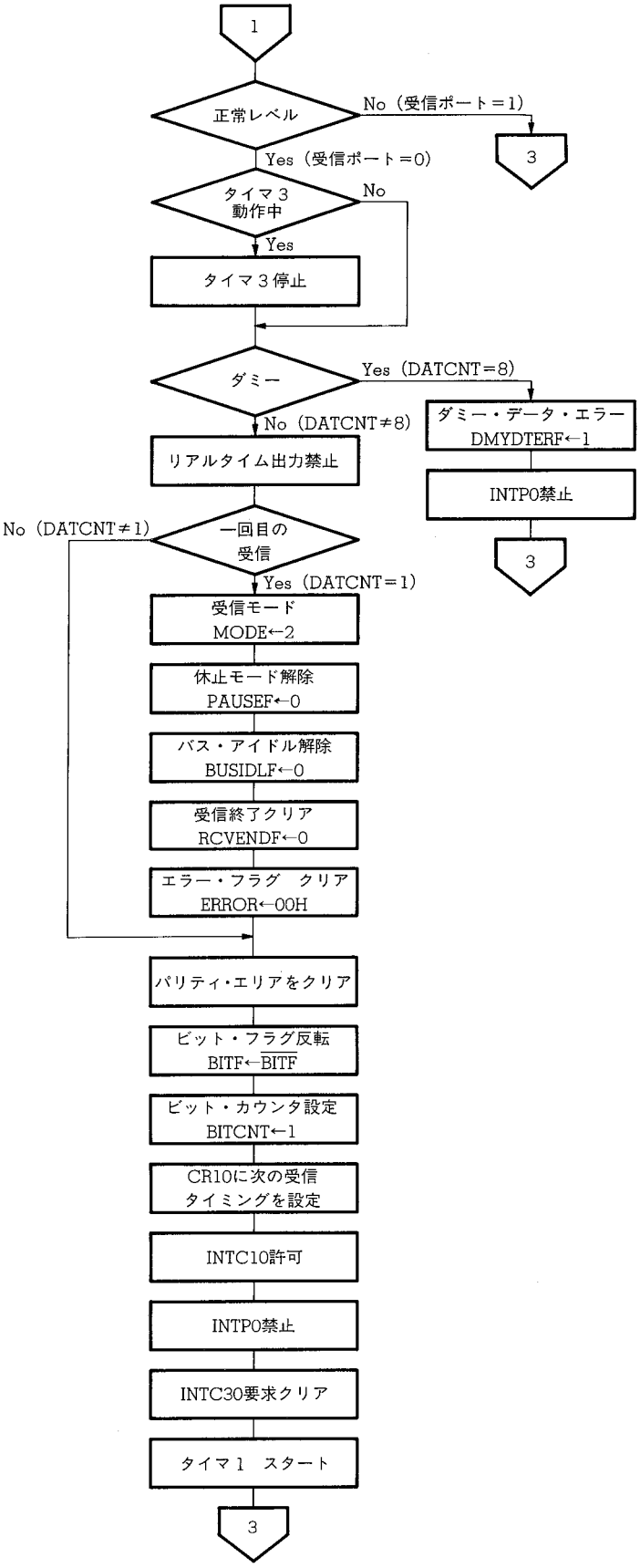
INTC30の同期回復終了処理内で，多重割り込み処理としてINTP0を受け付けた場合(IC30F＝1)は，再びINTC30処理に復帰すると，送信もしくは受信を停止してしまいます(INTC30同期回復終了処理参照)。したがって，この場合はスタック・ポインタを3つ進めて，割り込み1レベル分スタックを戻し，メイン処理に復帰するようにしています。

11
214
218A
234
244

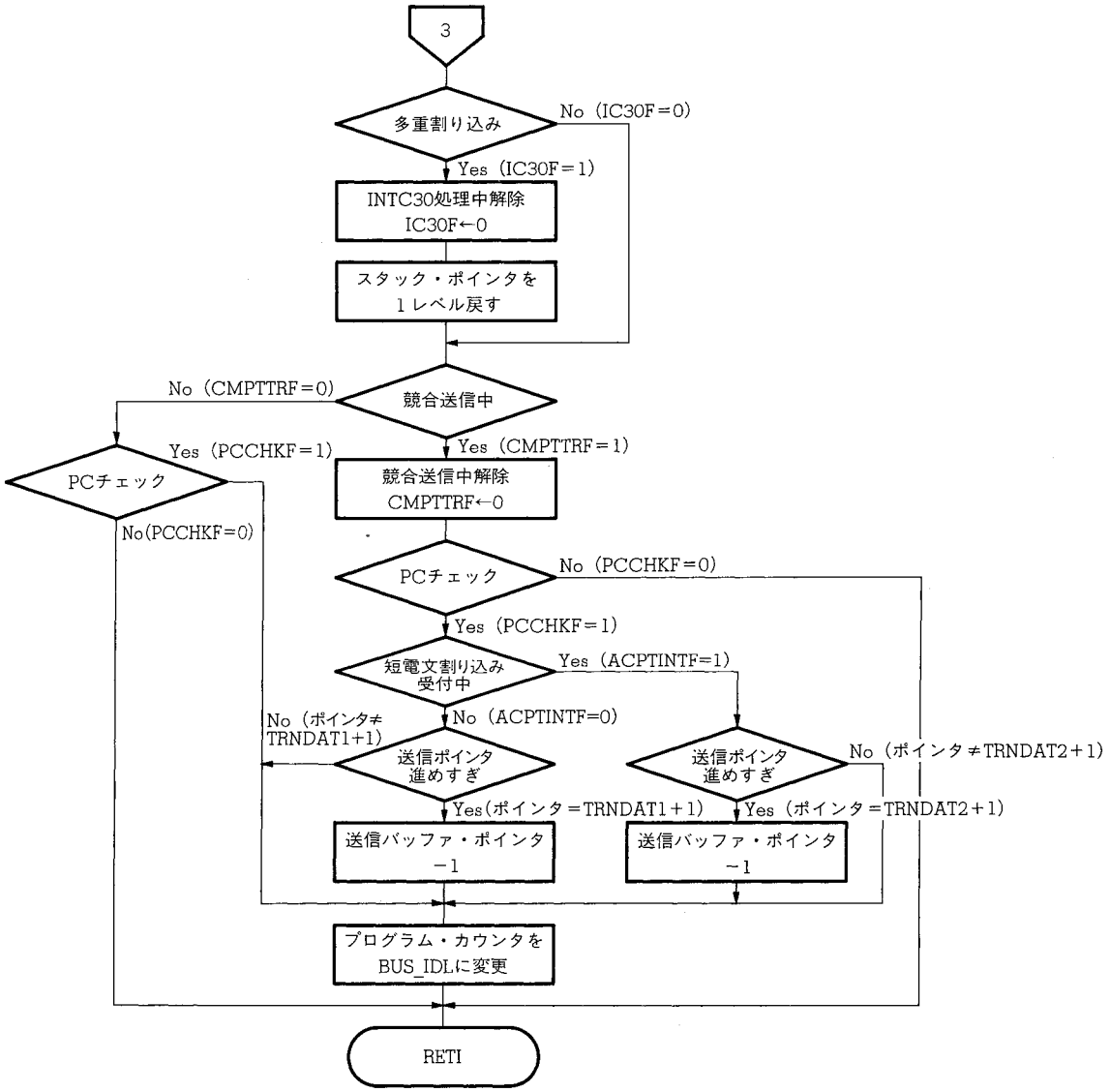
(d) 詳細フロー・チャート


INTP0
レジスタ・バンク1を選択
同期回復中
Yes (SYNCF=1)
同期回復中解除
SYNCF←0
No (SYNCF=0)
バス・アイドル
No (BUSIDLF=0)
Yes (BUSIDLF=1)
データ
作成終了
No (MKDATF=0)
Yes (MKDATF=1)
INTP0トリガ
出力許可
No (EP0TRGF=0)
Yes (EP0TRGF=1)
1
競合送信
CR10に次の送信
タイミングを設定
タイマ1　スタート
INTP0トリガでの
リアルタイム出力禁止
INTP0出力フラグ
EP0TRGF←0
タイマ3停止
P0Lにオフデューティ・
データを設定
送信バッファ・ポインタ
を+1
ビット・フラグ反転
BITF←BITF
ビット・カウンタ設定
BITCNT←1
送信モード
MODE←1
パリティ・エリアをクリア
バス・アイドル解除
BUSIDLF←0
休止モード解除
PAUSEF←0
エラー・フラグ　クリア
ERROR←00H
2
2
INTC10許可
INTP0禁止
INTC30要求クリア
競合送信中に設定
CMPTTRF←1
3

スタート・ビット受信



11
214
218A
234
244


3
多重割り込み
No（IC30F＝0）
Yes（IC30F＝1）
INTC30処理中解除
IC30F←0
スタック・ポインタを
1レベル戻す
No（CMPTTRF＝0）
競合送信中
Yes（CMPTTRF＝1）
Yes（PCCHKF＝1）
PCチェック
競合送信中解除
CMPTTRF←0
No(PCCHKF=0)
PCチェック
No（PCCHKF＝0）
Yes（PCCHKF＝1）
短電文割り込み
受付中
Yes（ACPTINTF=1）
No（ポインタ≠
TRNDAT1+1）
No（ACPTINTF=0）
送信ポインタ
進めすぎ
送信ポインタ
進めすぎ
No（ポインタ≠TRNDAT2＋1）
Yes(ポインタ＝TRNDAT1＋1)
Yes（ポインタ＝TRNDAT2＋1）
送信バッファ・ポインタ
－1
送信バッファ・ポインタ
－1
プログラム・カウンタを
BUS_IDLに変更
RETI

(8) INTC30割り込み処理

割り込み処理名：INTC30

処理は，次の３つに分かれます。

○受信タイム・アウト
○同期回復開始
○同期回復終了

(a) 受信タイム・アウト

休止中でない（PUSEF＝０）場合にこの処理に入ると，受信タイム・アウトとします（TMOUTERF←１）。

データ・カウンタ（DATCNT）＝９の場合は，ACK/NAK受信タイム・アウトとなり，無応答エラーとして（NOANSERF←１），タイマ１を起動します。このあと発生するINTC10では，受信モードのACK/NAK部の処理を行います。

データ・カウンタ≠９の場合は，自己アドレス（SA）からチェック・コード（FCC）までのデータ受信タイム・アウトとなり，ここでフレーム終了として休止モードに設定します（PAUSEF←１）。

(b) 同期回復開始

休止モード（PAUSEF＝１）で，同期回復中でない（SYNCF＝０）の場合は，同期回復を開始します。

同期回復中にスタート・ビットを検出した場合，これに競合して送信できるよう，INTP0トリガによるリアルタイム出力を許可にしておきます。ただし，これは送信波形データの作成が終了している（MKDATF＝１）の場合のみです。

(c) 同期回復終了

次の処理を行います。

○多重割り込みの許可
○送信，受信終了フラグのセット
○モードのクリア

11
214
218A
234
244

①　多重割り込みの許可

休止モード(PAUSEF＝1)で，同期回復中(SYNCF＝1)の場合は，同期回復を終了します。

INTP0トリガによるリアルタイム出力が許可になっており，この処理を行っている最中にスタート・ビットが検出された場合，この処理から復帰後にINTP0を受け付けるのでは，すでにP00にロウ・レベルを出力しているにもかかわらず，競合送信開始処理に入るのが保留されます。この場合，スタート・ビット幅が保留された時間分だけ長くなってしまいます。これを避けるため，本処理内では多重割り込みを許可としています。

②　送信，受信終了フラグのセット

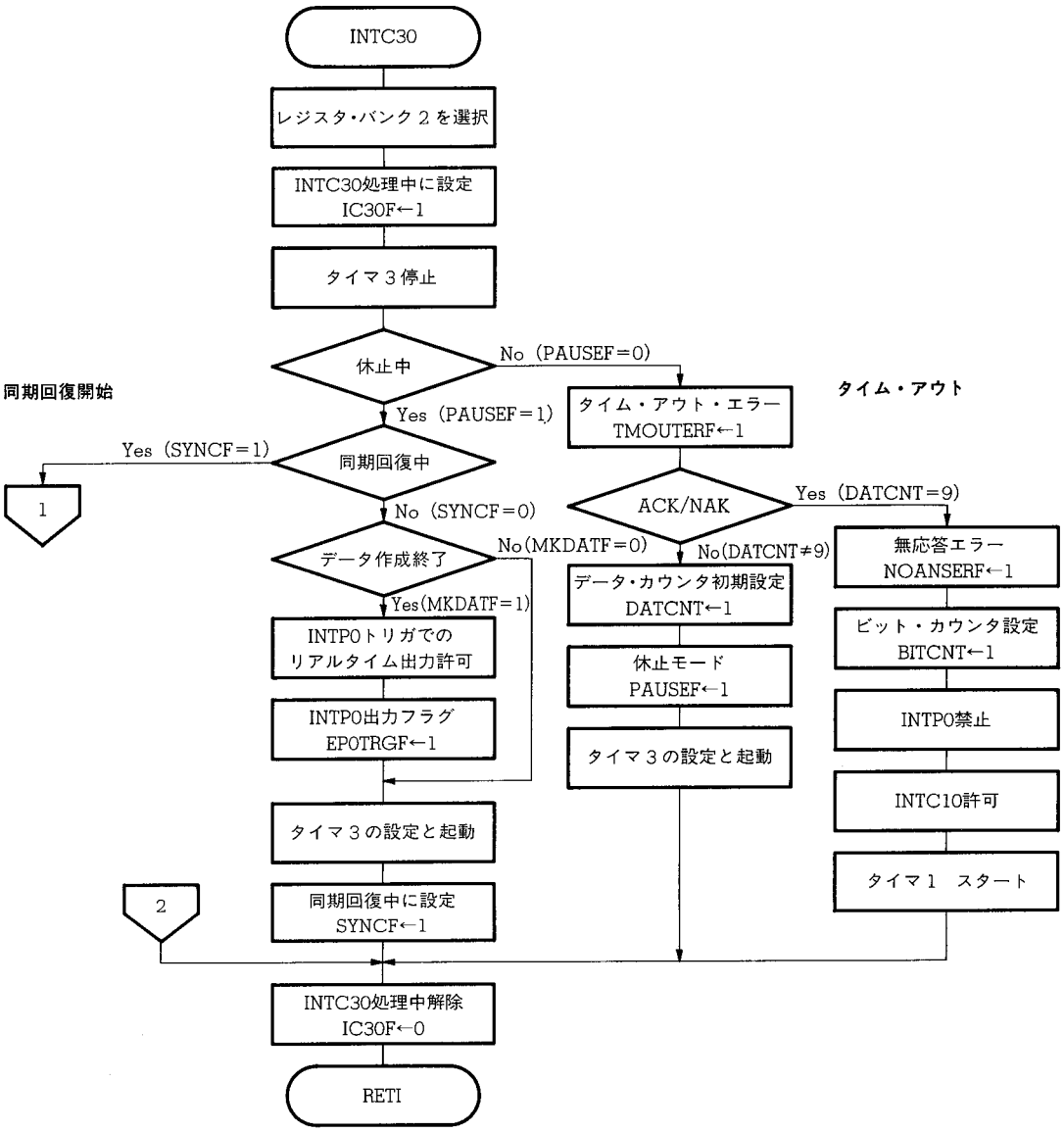
通信モード（モード番号（MODE））に応じて，送信終了フラグ（TRNENDF），もしくは受信終了フラグ（RCVENDF）をセット(1)します。ただし，監視モード（MODE＝3）の場合はいずれもセットされません。

また，競合負けし，途中で受信モードに移行している場合は，受信終了フラグがセットされます（送信終了フラグはクリアされたままとなります)。

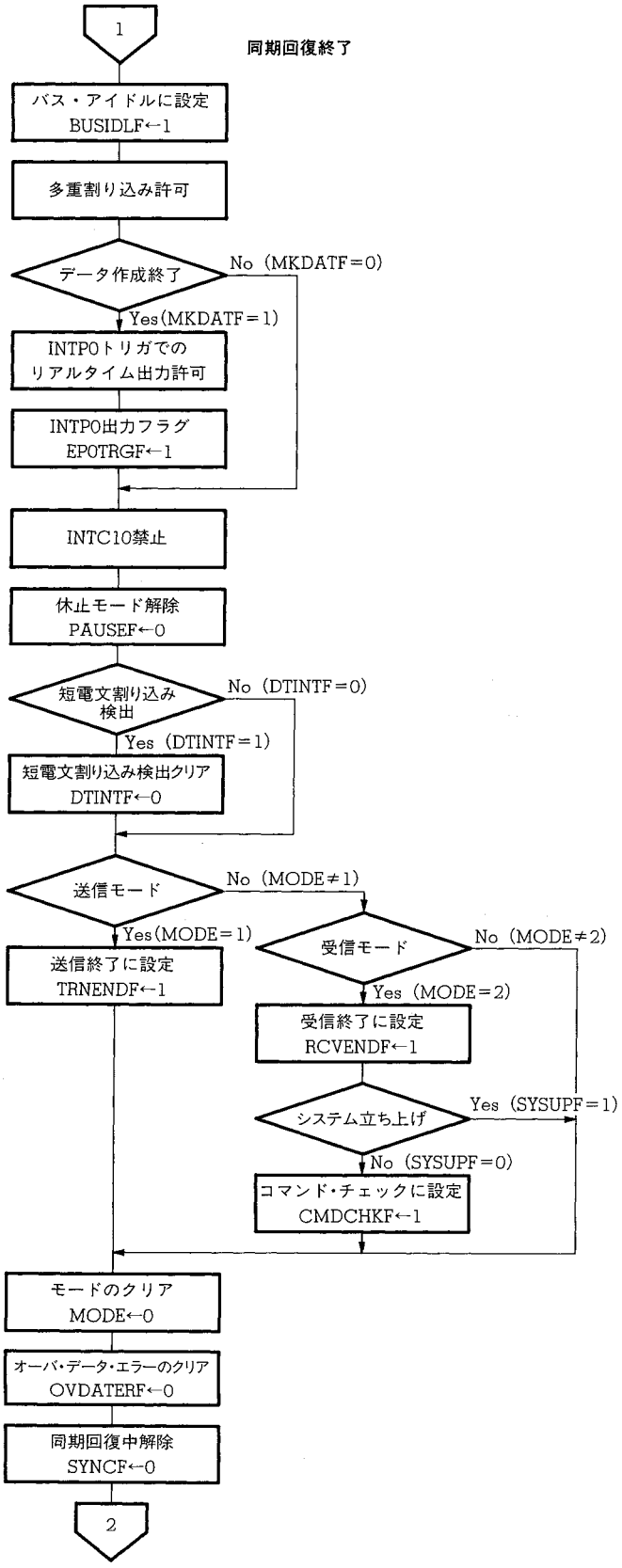
③　モードのクリア

同期回復が終了し，バス・アイドル状態に戻るときは，モード番号（MODE）を00Hとします。

(d) 詳細フロー・チャート



11
214
218A
234
244


1
同期回復終了
バス・アイドルに設定
BUSIDLF←1
多重割り込み許可
データ作成終了
No（MKDATF＝0）
Yes(MKDATF＝1)
INTP0トリガでの
リアルタイム出力許可
INTP0出力フラグ
EP0TRGF←1
INTC10禁止
休止モード解除
PAUSEF←0
短電文割り込み
検出
No（DTINTF＝0）
Yes（DTINTF＝1）
短電文割り込み検出クリア
DTINTF←0
送信モード
No（MODE≠1）
Yes(MODE＝1)
受信モード
No（MODE≠2）
送信終了に設定
TRNENDF←1
Yes（MODE＝2）
受信終了に設定
RCVENDF←1
システム立ち上げ
Yes（SYSUPF＝1）
No（SYSUPF＝0）
コマンド・チェックに設定
CMDCHKF←1
モードのクリア
MODE←0
オーバ・データ・エラーのクリア
OVDATERF←0
同期回復中解除
SYNCF←0
2

(9) INTP1割り込み処理

割り込み処理名：INTP1

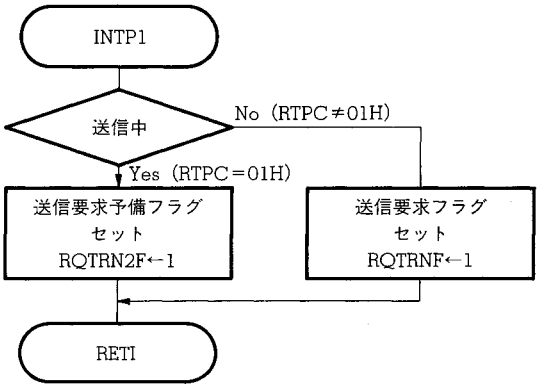
送信要求処理を行います。

自分が送信中（リアルタイム出力中）の場合は，すでに送信要求が出されていますので，送信要求予備フラグ（RQTRN2F）をセット(1)します。

上記以外の場合は，送信要求フラグ（RQTRNF）をセットします。

他のIFUと通信中でも，送信要求フラグはセットされますので，次の送信データを用意しておくことが可能です。

(a) 詳細フロー・チャート



11
214
218A
234
244

(10) INTP2割り込み処理

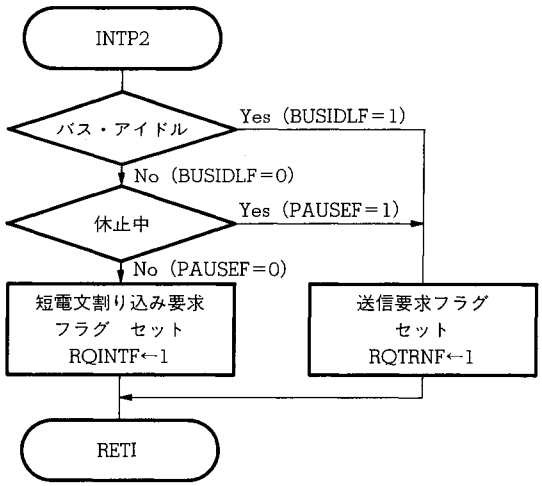
割り込み処理名：INTP2

短電文割り込み要求処理を行います。

バス・アイドル中（BUSIDLF＝1），休止中（PAUSEF＝1）の場合は，送信要求フラグ（RQTRNF）をセット(1)します。

上記以外（通信中）の場合は，短電文割り込み要求フラグ（RQINTF）をセット(1)します。

(a) 詳細フロー・チャート



(11) データ作成処理

サブルーチン名：MK_DATA

〈入力条件〉

Bレジスタ：データ数（フレームの長さ）

HLレジスタ：送信バッファ（TRNBUF1，TRNBUF2）の先頭アドレス

DEレジスタ：送信データ・バッファ（TRNDAT1，TRNDAT2）の先頭アドレス

〈使用レジスタ〉

AX

送信波形データを作成します。送信波形データの形式は，**11.4.3　送信波形データの形式**を参照してください。

① スタート・ビット（FEH）を送信データ・バッファに格納します。

② 送信データ（ビット）が1の場合は，送信データ・バッファにFFHを格納します。

送信データが0の場合は，極性フラグ(POLEF)により，+側，−側を判定し，FEH，FDHのいずれかを送信データ・バッファに格納します。

1ビットの生成ごとにパリティ・ビットを生成します。生成方法は，**(b) 受信モード　(i) ビット前半の処理**と同様です。

以上の処理を8回繰り返し，8ビット分のデータを作成します。

③ パリティ・データを送信データ・バッファに格納します。

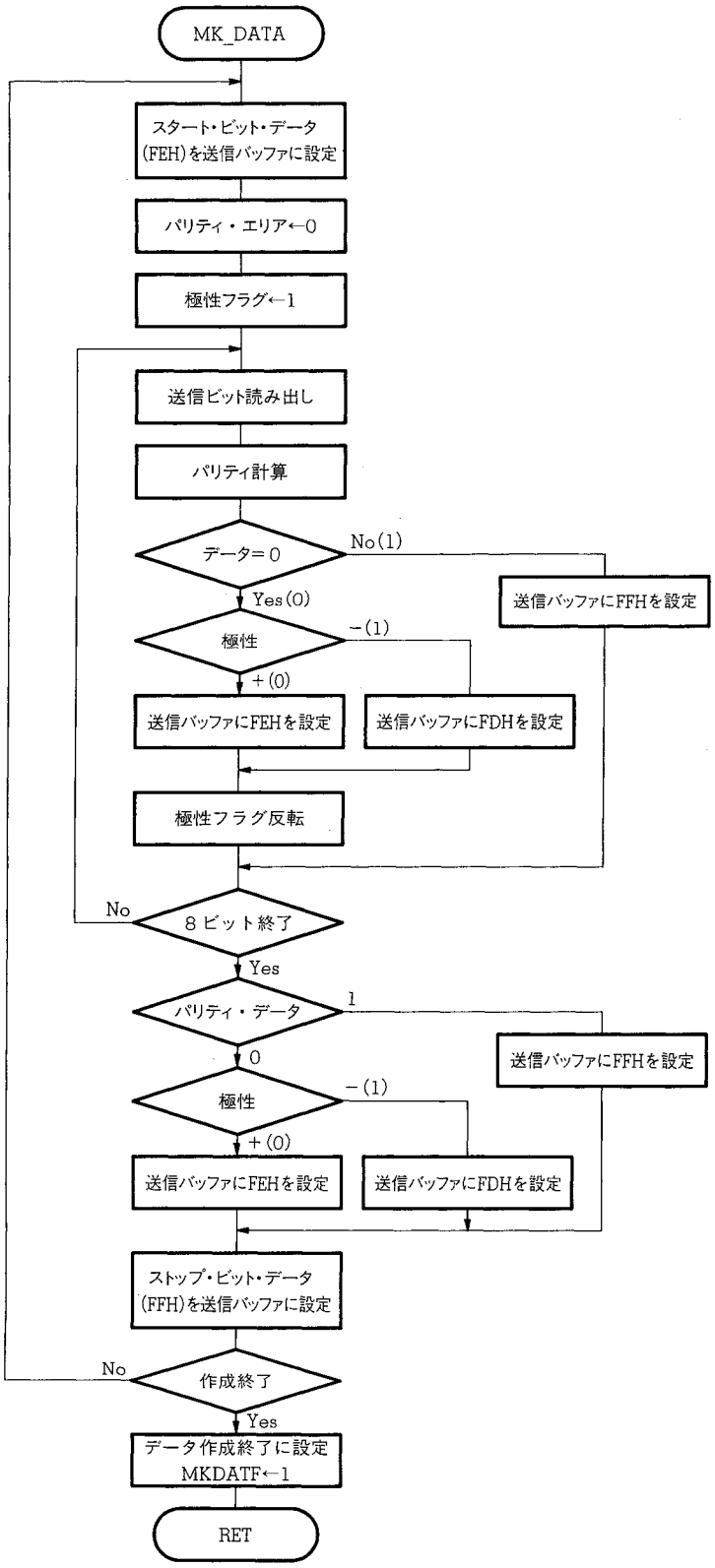
④ ストップ・ビット・データを送信データ・バッファに格納します。

⑤ Bレジスタで示すデータ数の分だけ，①〜④を繰り返します。

⑥ データ終了フラグ（MKDATF）をセット(1)します。

11
214
218A
234
244

（a）詳細フロー・チャート


MK_DATA
スタート・ビット・データ(FEH)を送信バッファに設定
パリティ・エリア←0
極性フラグ←1
送信ビット読み出し
パリティ計算
データ＝0
No(1)
送信バッファにFFHを設定
Yes(0)
極性
－(1)
＋(0)
送信バッファにFEHを設定
送信バッファにFDHを設定
極性フラグ反転
No
8ビット終了
Yes
パリティ・データ
1
0
送信バッファにFFHを設定
極性
－(1)
＋(0)
送信バッファにFEHを設定
送信バッファにFDHを設定
ストップ・ビット・データ(FFH)を送信バッファに設定
No
作成終了
Yes
データ作成終了に設定
MKDATF←1
RET

(12) 操作するモード・レジスタ

通信プログラム中で操作するモード・レジスタの設定を示します。

(a) リアルタイム出力ポート

リアルタイム出力ポート・コントロール・レジスタ

	7	6	5	4	3	2	1	0	
RTPC	BYTE	0注	0注	POMH	EXTR	0注	0注	POML	(リセットで00H)
設定例	0	0	0	0	0/1	0	0	0/1	0/1：プログラム中で切り替えながら使用しています

POML	POLの動作指定
0	ポート・モード
1	リアルタイム出力ポート・モード

EXTR	INTPOによるバッファ・レジスタから出力ラッチへのデータ転送の許可
0	許可しない
1	許可する　BYTE=0：POLのみ転送される BYTE=1：POL/Hが転送される

POMH	POHの機能指定
0	ポート・モード
1	リアルタイム出力ポート・モード

BYTE	リアルタイム出力ポート・モードの動作モード
0	4ビット・セパレート・リアルタイム出力ポート
1	8ビット・リアルタイム出力ポート

注　ビット1, 2, 5, 6には必ず“0”を書き込んでください。

11
214
218A
234
244

(b) 8ビット・タイマ／カウンタ1（割り込みソース：INTC10）

タイマ・コントロール・レジスタ1

	7	6	5	4	3	2	1	0	
TMC1	CE2	OVF2	CMD2	0	CE1	OVF1	0	0	（リセットで00H）
設定例	0	0	0	0	0/1	0	0	0	0/1：プログラム中で切り替えながら使用しています

8ビット・タイマ／カウンタ1

OVF1	TM1のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

CE1	TM1のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ2

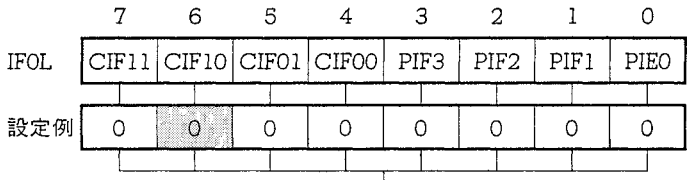
CMD2	TM2のカウント動作モード指定
0	通常モード
1	ワンショット・モード

OVF2	TM2のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー(FFHから00Hへのカウント・アップ)

備考　このビットはソフトウエアでのみリセットされます。

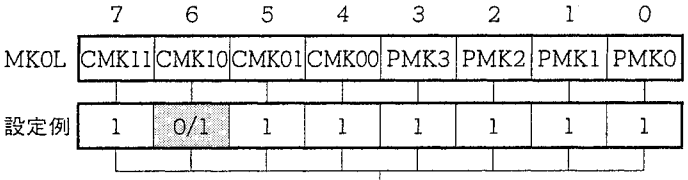
CE2	TM2のカウント動作制御
0	クリアしたままカウント動作停止
1	カウント動作許可

割り込み要求フラグ・レジスタL



IF	割り込み要求フラグ
0	割り込み要求が発生していない
1	割り込み要求発生

割り込みマスク・レジスタL



MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

11
214
218A
234
244

(c) エッジ割り込み（割り込みソース：INTPO）

割り込み要求フラグ・レジスタL

	7	6	5	4	3	2	1	0	
IF0L	CIF11	CIF10	CIF01	CIF00	PIF3	PIF2	PIF1	PIE0	（リセットで00H）
設定例	0	0	0	0	0	0	0	0	

IF	割り込み要求フラグ
0	割り込み要求が発生していない
1	割り込み要求発生

割り込みマスク・レジスタL

	7	6	5	4	3	2	1	0	
MK0L	CMK11	CMK10	CMK01	CMK00	PMK3	PMK2	PMK1	PMK0	（リセットでFFH）
設定例	1	1	1	1	1	1	1	0/1	0/1：プログラム中で切り替えながら使用しています

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

(d) 8ビット・タイマ／カウンタ3（割り込みソース：INTC30）

タイマ・コントロール・レジスタ0

	7	6	5	4	3	2	1	0	
TMC0	CE3	0	0	0	CE0	OVF0	0	0	（リセットで00H）
設定例	0/1	0	0	0	0	0	0	0	0/1：プログラム中で切り替えながら使用しています

16ビット・タイマ／カウンタ

OVF0	TM0のオーバフロー・フラグ
0	オーバフローなし
1	オーバフロー（FFFFHから0000Hへのカウント・アップ）

備考　このビットはソフトウエアでのみリセットされます。

CE0	TM0のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

8ビット・タイマ／カウンタ・ユニット3

CE3	TM3のカウント動作制御
0	クリアしたまま，カウント動作停止
1	カウント動作許可

11
214
218A
234
244

プリスケーラ・モード・レジスタ0

	7	6	5	4	3	2	1	0	
PRM0	PRS33	PRS32	PRS31	PRS30	0	0	0	0	（リセットで00H）
設定例	0/1	0/1	0/1	0/1	0	0	0	0	0/1：プログラム中で切り替えながら使用しています

8ビット・タイマ／カウンタ3　　f_{CLK}＝6 MHz

PRS33	PRS32	PRS31	PRS30	カウント・クロック[Hz]の指定
0	0	0	0	$f_{CLK}/8$
0	0	0	1	
0	0	1	0	$f_{CLK}/16$
0	0	1	1	$f_{CLK}/32$
0	1	0	0	$f_{CLK}/64$
0	1	0	1	$f_{CLK}/128$
0	1	1	0	$f_{CLK}/256$
0	1	1	1	$f_{CLK}/512$
1	×	×	×	設定禁止

備考　f_{CLK}：内部システム・クロック

割り込み要求フラグ・レジスタH

	7	6	5	4	3	2	1	0	
IF0H	CSIIF	STIF	SRIF	SERIF	CIF20	PIF5	PIF4	CIF21	（リセットで00H）
	0	0	0	0	0	0	0	0	

IF	割り込み要求フラグ
0	割り込み要求が発生していない
1	割り込み要求発生

割り込みマスク・レジスタH

	7	6	5	4	3	2	1	0	
MK0H	CSIMK	STMK	SRMK	SERMK	CMK20	PMK5	PMK4	CMK21	（リセットでFFH）
設定例	1	1	1	1	1	1	0/1	1	0/1：プログラム中で切り替えながら使用しています

MK	割り込みマスク・フラグ
0	割り込み処理許可
1	割り込み処理保留

11
214
218A
234
244

(13) 注意事項

(a) プログラミング上の注意

通信中は，ユーザ・プログラム中で処理時間が長い割り込み処理（26 μs程度以上）は入れないようにしてください。INTC10割り込み処理の受け付けが遅れると，受信タイミングが遅れることになります。

通信中のユーザ・プログラムの割り込み処理時間が長くなる場合は，INTC10の多重割り込みを許可にしてください。

(b) 動作上の注意

(i) 実現していない動作

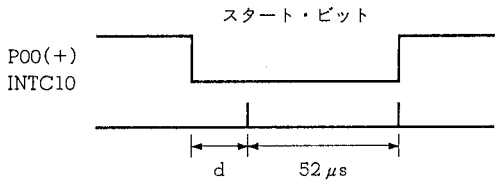
本プログラム例では，短電文割り込みによる登録要求を実現していません。したがって，IFUが途中で立ち上がった際，長電文の通信中であっても，登録要求コマンド・フレームの送信を短電文割り込みで行うことはできません。

(ii) 特異動作

① 次の命令実行タイミングでINTP0エッジ（スタート・ビット）が検出されると，競合送信の際，スタート・ビットが約12 μs長くなります(52 μs→64 μs)。

```
                メイン処理
                ⋮
TR_MOD：                                ；送信開始処理
                ⋮
        MOV   RTPC, #00000001B
        SET1  PMK0
        OR    TMC1, #00001000B
        CLR1  EP0TRGF
        CLR1  BUSIDLF
        BR    MAINLP
```

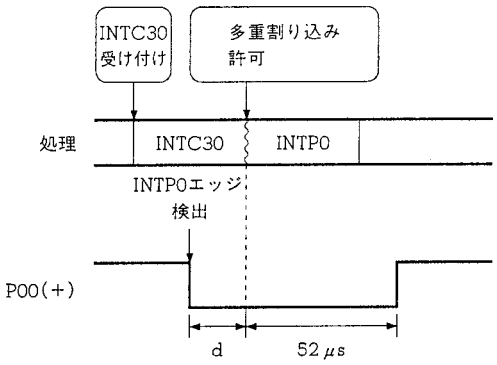
自分が送信開始処理に入り，RTPC, #00000001Bという命令を実行中にINTP0エッジを検出すると，競合送信(INTP0トリガによるリアルタイム出力)が行われます。しかし，次の命令でINTP0をマスクしていますので，INTP0割り込み処理(競合送信処理)は受け付けず，プログラム上ではメイン処理で送信開始することになります。したがって，スタート・ビットは次の図のようになります。



d＝SET1 PMK0, OR TMC1, #00001000Bの実行時間＋一発目の波形出力までの時間（8.2 μs）

＝約12 μs

② 同期回復終了処理において，多重割り込みを許可する前にスタート・ビット（INTP0エッジ）を検出すると，競合送信の際，INTP0エッジ検出からINTP0割り込み処理を受け付けるまでの時間（割り込み保留時間）だけスタート・ビット幅が長くなります。



dの最大時間は，次のようになります。

24クロック（INTC30割り込み受付時間）
＋38クロック（INTC30処理の最初からEI命令実行までの時間）
＝62クロック（約14.7 μs）

11
214
218A
234
244

11.5.5　プログラム・リスト

(1) メイン処理（モジュール名：**MAIN**　本プログラムに関係ある部分のみ）

```
$       TITLE   ('78K/II HOME-BUS CONTROL PROGRAM')
$       PROCESSOR(218A)
$       INCLUDE(INTMS.DEF)
;*************************************************************
;*************************************************************
;**      78K/II HOME-BUS CONTROL PROGRAM                    **
;**                                                         **
;**              TARGET : uPD78218A                         **
;*************************************************************
;*************************************************************

        NAME    MAIN
;*************************************************************
;*      MAIN ROUTINE                                         *
;*              *DEFINE MACRO                                *
;*              *INITIALIZE ROUTINE                          *
;*              *MAIN LOOP ROUTINE                           *
;*************************************************************

        PUBLIC  MODE,DATCNT,BITCNT,TRBFPNT
        PUBLIC  SYSUPMOD,SELFADRS,GRPCNT,RSTCNT
        PUBLIC  TRNENDF,RCVENDF,PAUSEF,PAUSE2F,BITF,INTCHRF
        PUBLIC  RQTRNF,RQTRN2F,RQINTF,MKDATF,BUSIDLF
        PUBLIC  TOSELFF,SYSRSTF,RGSTRQF,ACPTINTF
        PUBLIC  DTINTF,DTCLSHF,CLRMDATF,TRNINTF,INPUTDAT
        PUBLIC  SYSUPF,TRNRDYF,SYNCF,EP0TRGF
        PUBLIC  CMPTTRF,IC30F,CMDCHKF,PCCHKF
        PUBLIC  ADRSGRP,GRPBIT,GRPBIT_R
        PUBLIC  BUS_IDL

        EXTRN   SYS_UP,MK_DATA,CMD_CHK
        EXTRN   TRNBUF1,TRNDAT1,TRNBUF2,TRNDAT2,RCVBUF
        EXTRN   ERROR,BC_,DATA,CTLCODE,CHRCNT,RCVDAT
        EXTRN   TRN,RCV,WATCH,FRSTTIM,MAXDAT
        EXTRN   TPR1,TSA1,TBC1,TPR2,TSA2,TBC2
        EXTBIT  PRTYERF,CKSUMERF,FRMERF
        EXTBIT  TMOUTERF,DMYDTERF,NAKERF
        EXTBIT  NOANSERF,TRNERF,OVDATERF

;*****  define stack area  *****

STACKS  DSEG    SADDR
        DS      32
STACK:
```

```
;*****  define work area  *****

M_WORKS  DSEG   SADDR
MODE:    DS     1                    ; mode of transmit/receive/watch
DATCNT:  DS     1                    ; data counter (data kind)
BITCNT:  DS     1                    ; bit counter (bit kind)
TRBFPNT: DS     2                    ; transmit buffer pointer
SELFADRS:DS     1                    ; self address area
SYSUPMOD:DS     1                    ; system-up mode NO.
GRPCNT:  DS     1                    ; address group counter
RSTCNT:  DS     1                    ; wait counter just reset for IFU

;*****  define work flag  *****

         BSEG
TRNENDF  DBIT                         ; end of transmit
RCVENDF  DBIT                         ; end of receive
BUSIDLF  DBIT                         ; bus idle
MKDATF   DBIT                         ; end of making data
PAUSEF   DBIT                         ; pause time
PAUSE2F  DBIT                         ; pause term
BITF     DBIT                         ; bit of before/after
INTCHRF  DBIT                         ; interrupting of short frame
RQTRNF   DBIT                         ; request of transmission
RQTRN2F  DBIT                         ; reserve request of transmission
RQINTF   DBIT                         ; request of interrupt of short frame
TOSELFF  DBIT                         ; frame to self
SYSRSTF  DBIT                         ; system reset
SYSUPF   DBIT                         ; system up
RGSTRQF  DBIT                         ; request of registing
ACPTINTF DBIT                         ; accept short frame interrupt
CLRMDATF DBIT                         ; clear making data flag
DTINTF   DBIT                         ; detect short frame interrupt
DTCLSHF  DBIT                         ; detect data clash
TRNINTF  DBIT                         ; short frame interrupt in transmission
INPUTDAT DBIT                         ; end input data
TRNRDYF  DBIT                         ; transmissive ready
SYNCF    DBIT                         ; recovering syncronism
CMPTTRF  DBIT                         ; competitive transmission
IC30F    DBIT                         ; in INTC30 routine
CMDCHKF  DBIT                         ; command check
EP0TRGF  DBIT                         ; INTP0 trigger enable
PCCHKF   DBIT                         ; program counter check
```

11
214
218A
234
244

```
;*****  define vector table  *****

        RSTVENT                         ; RESET vector table

;---------------------------------------
;       DEFINE MACRO
;---------------------------------------

;*****  setting parameter for making data *****

PARASET MACRO TSA,TBC,TRNBUF,TRNDAT
        LOCAL   M_CHKLP,DAT_CNT

        MOV     A,!TBC                  ; read character count
        CMP     A,#MAXDAT+1             ; DATA size over ?
        BC      $DAT_CNT                ;
        MOV     A,#MAXDAT               ; set max size
        SET1    OVDATERF                ; set over data error flag
DAT_CNT:
        ADD     A,#5                    ; PR - DATA count
        MOV     B,A                     ;
        DEC     A                       ;
        MOV     C,A                     ;
        MOVW    HL,#TSA                 ; set transmissive SA address
        MOV     A,#0                    ;
M_CHKLP:
        ADD     A,[HL+]                 ;
        DBNZ    C,$M_CHKLP              ; end calculate transmissive FCC ?
        XOR     A,#0FFH                 ; reverse sum data
        INC     A                       ; A=FCC
        MOV     TRNBUF[B],A             ; save transmissive FCC
        INC     B                       ; set data count
        MOVW    HL,#TRNBUF              ; set source address
        MOVW    DE,#TRNDAT              ; set destination address

        ENDM
```

```
MAINS   CSEG
RST:
;---------------------------------------
;       INITIALIZE ROUTINE
;---------------------------------------

;*****  memory environment initialize  *****

        MOVW    SP,#STACK               ; stack pointer
        MOV     MM,#10000000B           ; high speed fetch, single-chip mode
        OR      PM6,#00000000B          ; memory bank 0
                :
                : (ユーザ・プログラムのイニシャライズ処理)
                :
;*****  read self address *****
                :
                : (自己アドレスをAレジスタに設定)
                :
        MOV     SELFADRS,A              ; save self address

;*****  HBPC initialize *****

        CALL    !HBPC_INI               ; HBPC initialize sub-routine

;*****  ready system-up routine *****

        SET1    SYSUPF                  ; set system-up mode
        MOV     SYSUPMOD,#0             ; set sequence of starting register

        EI                              ; enable all interrupt

;---------------------------------------
;       MAIN LOOP
;---------------------------------------
MAINLP:
        CLR1    PCCHKF                  ; clear PC check
        BF      SYSUPF,$NORMAL          ; system-up mode ?
        CALL    !SYS_UP                 ; call sysem-up routine

;*****  NORMAL MODE  *****

NORMAL:
                :
                :
                : (ユーザ・プログラムのメイン・ループ)
                :
                :
                :
```

11
214
218A
234
244

```
        BT      SYSUPF,$ACPT_INT        ; system-up mode ?
        BF      RCVENDF,$ACPT_INT       ; receive end ?
        BF      CMDCHKF,$ACPT_INT       ; command check ?
        CALL    !CMD_CHK                ; call command check routine
        BF      SYSRSTF,$CHK_RGRQ       ; system reset ?
SYSRSTLP:
        BR      SYSRSTLP                ; loop forever
CHK_RGRQ:
        BF      RGSTRQF,$ACPT_INT       ; request of registing ?
        SET1    SYSUPF                  ; THEN set ststem-up mode
        MOV     SYSUPMOD,#2             ;      set request of registing mode
        CLR1    RGSTRQF                 ;      clear request of register
ACPT_INT:
        BT      ACPTINTF,$M_DATA        ; accept short frame interrupt ?
        BT      RQTRNF,$M_DATA          ; ELSE request of transmission ?
        BR      MAINLP                  ;
M_DATA:
        BT      INPUTDAT,$CLR_IPDT      ; end data-input ?
        BR      MAINLP                  ;
CLR_IPDT:
        BT      MKDATF,$BUS_IDL         ; end making data ?
        BT      ACPTINTF,$INT_CHKD      ; accepting interrupt ?

                                        ; THEN
        PARASET TSA1,TBC1,TRNBUF1,TRNDAT1 ; MACRO (set parameter for MK_DATA)

        BR      MKDAT                   ;

INT_CHKD:
        PARASET TSA2,TBC2,TRNBUF2,TRNDAT2 ; MACRO

MKDAT:
        CALL    !MK_DATA                ; ELSE making data
BUS_IDL:
        SET1    PCCHKF                  ; set PC check
        BT      BUSIDLF,$TR_MOD         ; bus idle ?
        BR      MAINLP                  ;
TR_MOD:
        MOV     RTPC,#00001001B         ; INTP0 trigger, P0L realtime output
        SET1    EP0TRGF                 ; set INTP0 trigger enable flag
        MOV     MODE,#TRN               ; ELSE set transmission mode
        CLR1    TRNENDF                 ; clear transmission end flag
        CLR1    INPUTDAT                ;
        MOV     ERROR,#0                ; clear all error flag
        MOVW    AX,TRBFPNT              ; read transmissive buffer pointer data
        MOVW    HL,AX                   ;
        MOV     A,[HL+]                 ;
        MOV     P0L,A                   ; set first transmissive data
        MOVW    AX,HL                   ;
        MOVW    TRBFPNT,AX              ; save transmissive buffer pointer
        CLR1    CIF10                   ; clear request of INTC10
```

```
        MOV     CR10,#FRSTTIM           ; set first timing data to CR10
        CLR1    CMK10                   ; enable INTC10
        MOV     RTPC,#00000001B         ; enable realtime output
        SET1    PMK0                    ; disable INTP0
        OR      TMC1,#00001000B         ; start TM1
        CLR1    EP0TRGF                 ; clear INTP0 trigger enable flag
        CLR1    BUSIDLF                 ; clear bus-idle flag
        BR      MAINLP                  ;

;---------------------------------------
;       HBPC INITIALIZE SUB-ROUTINE
;---------------------------------------
HBPC_INI:
        MOV     P0,#0FFH                ; set P0 initial data to P0
        MOV     P0L,#0FFH               ; set P0L initial data to P0L
        MOV     PM0,#00000000B          ; set P0 output mode

        MOV     INTM0,#00000000B        ; INTP0,INTP1,INTP2 falling edge
        MOV     INTM1,#00001000B        ; select INTC30
        MOV     PR0L,#11111110B         ; INTP0 high priority, other low
        MOV     PR0H,#11111111B         ;

        MOV     CRC1,#00000010B         ; enable clear TM1 by matching CR10
        MOV     PRM1,#00000000B         ; set TM1 prescaler fclk/16

;*****  RAM clear  *****

        MOVW    HL,#TRNBUF1             ; set transmit buffer first address
RMCLRLP:
        MOV     A,#0                    ; full with 0
        MOV     [HL+],A
        MOVW    AX,HL                   ;
        CMPW    AX,#(0FB00H+(0FE20H-0FB00H));
        BNZ     $RMCLRLP                ;

;*****  work area initialize  *****

;       << work RAM >>

        MOV     MODE,#0                 ; mode of transmit/receive/watch
        MOV     DATCNT,#1               ; data counter (data kind)
        MOV     BITCNT,#11              ; bit counter (bit kind)
        MOVW    AX,#TRNDAT1             ; transmit buffer pointer
        MOVW    TRBFPNT,AX              ;
        MOV     ERROR,#0                ; error flag area
        MOV     GRPCNT,#0               ; address group count
```

11
214
218A
234
244

```
;       << work FLAG >>

        CLR1    MKDATF                  ; end of making data
        CLR1    PAUSEF                  ; pause time
        CLR1    PAUSE2F                 ; pause term
        CLR1    BITF                    ; bit of before/after
        CLR1    INTCHRF                 ; interrupting of short frame
        CLR1    RQTRNF                  ; request of transmission
        CLR1    RQTRN2F                 ; reserve request of transmission
        CLR1    RQINTF                  ; request of interrupt of short frame
        CLR1    TOSELFF                 ; frame to self
        CLR1    SYSRSTF                 ; system reset
        CLR1    SYSUPF                  ; system up
        CLR1    RGSTRQF                 ; request of registing
        CLR1    ACPTINTF                ; accept short frame interrupt
        CLR1    DTINTF                  ; detect short frame interrupt
        CLR1    DTCLSHF                 ; detect data clash
        CLR1    CLRMDATF                ; clear make data flag
        CLR1    TRNINTF                 ; short frame interrupt when transmit
        CLR1    INPUTDAT                ; end input data
        CLR1    TRNRDYF                 ; transmissive ready
        CLR1    SYNCF                   ; recovering syncronism
        CLR1    CMPTTRF                 ; competitive transmission
        CLR1    IC30F                   ; in INTC30 routine
        CLR1    CMDCHKF                 ; command check
        CLR1    EP0TRGF                 ; INTP0 trigger enable
        CLR1    OVDATERF                ; over data error
        SET1    TRNENDF                 ; end of transmit
        SET1    RCVENDF                 ; end of receive
        SET1    BUSIDLF                 ; bus idle

;*****  open mask interrupt *****

        CLR1    PMK1                    ; enable INTP1
        CLR1    PMK2                    ; enable INTP2

        RET                             ;
```

```
;---------------------------------------
;       DEFINE ROM DATA
;---------------------------------------

;*****  ADDRESS GROUP *****

ADRSGRP:                                ; SELF ADDRESS
        DB      0                       ; 00H
        DB      2                       ; 02H
        DB      4                       ; 01H

GRPBIT:                                 ; ADDRESS GROUP
        DB      00000001B               ; 0
        DB      00000100B               ; 2
        DB      00010000B               ; 4

GRPBIT_R:                               ; ADDRESS GROUP for receive
        DB      00000001B               ; 0
        DB      00010000B               ; 4
        DB      00000100B               ; 2

        END
```

11

214

218A

234

244

(2) システム立ち上げ処理（モジュール名：SYSUP）

```
$       TITLE   ('78K/II HOME-BUS CONTROL PROGRAM')
$       PROCESSOR(218A)
$       INCLUDE(INTMS.DEF)

        NAME    SYSUP
;*************************************************************
;*      *SYSTEM UP ROUTINE HBC SIDE                          *
;*                         IFU SIDE                          *
;*************************************************************

        PUBLIC  SYS_UP,CMD_CHK

        EXTRN   SELFADRS,SYSUPMOD,TRNBUF1,GRPCNT
        EXTRN   MODE,ADRSGRP,GRPBIT,RSTCNT
        EXTRN   TRN,RCV,TSA1,TDA1,TDATA1,SA,DATA,CC
        EXTRN   RQRGCMD0,SRSTCMD0
        EXTBIT  RQTRNF,INPUTDAT,TRNENDF,RCVENDF,BUSIDLF
        EXTBIT  PAUSEF,SYSUPF,RGSTRQF,CMDCHKF,SYSRSTF

;*****  define data *****

CIF30    EQU    IF0H.1                  ; INTC30 request flag
CMK30    EQU    MK0H.1                  ; INTC30 mask flag

RSTWAITH EQU    4170                    ; wait time after just reset HBC
RSTWAITI EQU    100                     ;                            IFU
RCVWAIT1 EQU    228                     ; wait receiving for HBC
RCVWAIT2 EQU    114                     ; wait receiving for IFU
GRPNUM   EQU    2                       ; count number of address group

;---------------------------------------
;       DEFINE MACRO
;---------------------------------------

;*****  move ROM data to trans-buffer  *****

DATMOV  MACRO   DATADRS
        LOCAL   DATMOVLP

        MOV     A,!DATADRS              ; read move data count
        MOV     B,A
        MOVW    HL,#(DATADRS+1)
        MOVW    DE,#TRNBUF1
DATMOVLP:
        MOV     A,[HL+]
        MOV     [DE+],A                 ; move data
        DBNZ    B,$DATMOVLP

        ENDM
```

```
SUPS    CSEG
;---------------------------------------
;       SYSTEM UP ROUTINE
;---------------------------------------

SYS_UP:
        CMP     SELFADRS,#00H           ; self address is 00H ?
        BZ      $HBC                    ; ELSE IFU side
        BR      IFU                     ;

;*****  HBC SIDE *****
HBC:
        MOV     A,SYSUPMOD              ; read system-up mode NO. in HBC side
        ADD     A,A                     ;
        MOV     X,A                     ;
        MOV     A,#00H                  ;
        ADDW    AX,#H_SUPADR            ;
        MOVW    HL,AX                   ;
        MOVW    AX,[HL]                 ;
        BR      AX                      ; branch each system-up mode

;       << transmission of starting register >>
;       << SYSUPMODE=0 >>

H_STRGST:
        MOVW    BC,#RSTWAITH            ; WAIT 10ms
WAITLP:
        DECW    BC                      ;
        MOVW    AX,BC                   ;
        CMPW    AX,#0                   ;
        BNZ     $WAITLP                 ;
        BTCLR   PIF0,$H_STRGST          ; receive data ?

        DATMOV  RGSTSTRT                ; move ROM data to trans-buffer

        MOV     A,SELFADRS              ; read self address
        MOV     !TSA1,A                 ; save self address

        MOV     A,GRPCNT                ;
        MOV     B,A                     ;
        INC     B                       ;
        MOV     A,GRPBIT[B]             ; read address group bit
        MOV     !TDA1,A                 ; destination address
        MOV     A,ADRSGRP[B]            ; read address group number
        OR      A,#01100000B            ;
        MOV     !(TDATA1+2),A           ; TDATA1+2 : address group save area
        INC     GRPCNT                  ; count address group
        SET1    RQTRNF                  ; set request of transmission flag
        SET1    INPUTDAT                ; set input end flag
        BR      H_SUPCNT
```

11
214
218A
234
244

```
;       << receiving of request register >>
;       << SYSUPMODE=1 >>

H_RQRGST:
        BF      TRNENDF,$H_RQRGST       ; end transmission ? (start register)
        CLR1    RCVENDF                 ; clear receiving end flag
        BR      H_SUPCNT

;       << transmission of confirming register >>
;       << SYSUPMODE=2 >>

H_CFRGST:
        BF      RCVENDF,$H_CFRGST       ; end receiving ? (request register)
        CLR1    RGSTRQF                 ;       clear request of register

        DATMOV  RGSTCNFM                ; move ROM data to trans-buffer

        MOV     B,#4                    ;
        MOVW    HL,#DATA+2              ; 2nd address of request register data
        MOVW    DE,#TDATA1+5            ; 5th address of confirm register data
SAVCFDT:
        MOV     A,[HL+]                 ;
        MOV     [DE+],A                 ; save confirm register data
        DBNZ    B,$SAVCFDT              ; end saving 4 - 7 byte ?

        MOV     A,SELFADRS              ; read self address
        MOV     !TSA1,A                 ; self address save area
        MOV     A,!SA                   ; read destination address
        MOV     !TDA1,A                 ; save destination address
        SET1    RQTRNF                  ; set request of transmission flag
        SET1    INPUTDAT                ; set input end flag
        CLR1    TRNENDF                 ; clear transmission end flag
H_SUPCNT:
        INC     SYSUPMOD                ; count system-up mode
        BR      RET_SUP                 ;

;       << end of register >>
;       << SYSUPMOD=3 >>

H_EDRGST:
        BF      TRNENDF,$H_EDRGST       ; end transmission ? (confirm register)
        SET1    CMK30                   ; disable INTC30
        MOV     CR30,#RCVWAIT1          ; set receiving wait time
        MOV     PRM0,#01110000B         ; set TM3 prescaler fclk/512
        OR      TMC0,#10000000B         ; start TM3
RV_MOD1?:
        CMP     MODE,#RCV               ; receiving mode ?
        BZ      $SYSUPINI               ;
        BF      CIF30,$RV_MOD1?         ; timeout ? THEN clear INTC30 request
        CLR1    CIF30                   ; clear INTC30 request flag
        AND     TMC0,#01111111B         ; THEN stop TM3
```

```
        MOV     A,#GRPNUM               ; read address group number
        CMP     A,GRPCNT                ; regist another group too ?
        BZ      $H_SCNSL                ;
        MOV     SYSUPMOD,#0             ;
        BR      RET_SUP                 ;
H_SCNSL:
        CLR1    SYSUPF                  ; cansel system-up mode
        BR      RET_SUP                 ;
SYSUPINI:
        AND     TMC0,#01111111B         ; THEN stop TM3
        MOV     SYSUPMOD,#2             ;
        BR      RET_SUP                 ;
```

11

214

218A

234

244

```
;*****  IFU SIDE *****

IFU:
        MOV     A,SYSUPMOD              ; read system-up mode NO. in IFU side
        ADD     A,A                     ;
        MOV     X,A                     ;
        MOV     A,#00H                  ;
        ADDW    AX,#I_SUPADR            ;
        MOVW    HL,AX                   ;
        MOVW    AX,[HL]                 ;
        BR      AX                      ; branch each system-up mode

;       << receiving of starting register >>
;       << SYSUPMOD=0 >>

I_STRGST:
        MOV     RSTCNT,#RSTWAITI        ; initialize reset wait counter : 1s
        CLR1    PIF0                    ; clear INTP0 request flag
        CLR1    PMK0                    ; enable INTP0
        SET1    CMK30                   ; disable INTC30
        MOV     CR30,#RCVWAIT2          ; set wait time
        MOV     PRM0,#01110000B         ; set TM3 prescaler fclk/512
        OR      TMC0,#10000000B         ; start TM3
RV_MOD2?:
        CMP     MODE,#RCV               ; receiving mode ?
        BZ      $I_SUPCNT               ;
        BTCLR   CIF30,$DEC_CNT          ; timeout ? THEN clear INTC30 request
        BR      RV_MOD2?                ;
DEC_CNT:
        DBNZ    RSTCNT,$RV_MOD2?        ; reset syncronise count end ?
        CLR1    CIF30                   ; clear INTC30 request flag
        AND     TMC0,#01111111B         ; THEN stop TM3
        BR      I_SUPCNT                ;

;       << transmission of request register >>
;       << SYSUPMOD=1 >>

I_RQRGST:
        BF      RCVENDF,$I_RQRGST       ; end receiving ?

        DATMOV  RGSTRQST                ; move ROM data to trans-buffer

        MOV     A,SELFADRS              ; read selg address
        MOV     !TSA1,A                 ; self address save area
        SET1    RQTRNF                  ; set request of transmision
        SET1    INPUTDAT                ; set data input end
        CLR1    TRNENDF                 ; clear transmission end falg
        BR      I_SUPCNT                ;
```

```
;       << receiving of confirming register >>
;       << SYSUPMOD=2 >>

I_CFRGST:
        BF      BUSIDLF,$I_CFRGST       ; bus idle ?
I_SETTM3:
        BT      TRNENDF,$I_CLRRCV       ; end transmission ?
        SET1    CMK30                   ; disable INTC30
        MOV     CR30,#RCVWAIT2          ; set wait time
        MOV     PRM0,#01110000B         ; set TM3 prescaler fclk/512
        OR      TMC0,#10000000B         ; start TM3
I_CHKTRN:
        BT      BUSIDLF,$CHK_IDL        ; bus idle ?
        CLR1    CIF30                   ; clear INTC30 request flag
        AND     TMC0,#01111111B         ; THEN stop TM3
        BR      I_CFRGST                ;
CHK_IDL:
        BF      CIF30,$I_CHKTRN         ; timeout ? THEN clear INTC30 request
        CLR1    CIF30                   ; clear INTC30 request flag
        AND     TMC0,#01111111B         ; THEN stop TM3
        BR      RET_SUP                 ;
I_CLRRCV:
        CLR1    RCVENDF                 ; clear receiving end flag

I_SUPCNT:
        INC     SYSUPMOD                ; count system-up mode
        BR      RET_SUP                 ;

;       << end of register >>
;       << SYSUPMOD=3 >>

I_EDRGST:
        BF      RCVENDF,$I_EDRGST       ; end receiving ? (confirm register)
        CLR1    SYSUPF                  ; cansel system-up mode

RET_SUP:
        RET
```

11
214
218A
234
244

```
;*****************************************
;       COMMAND CHECK ROUTINE
;*****************************************

CMD_CHK:
        CLR1    CMDCHKF                 ; clear command check flag
        MOV     A,!CC                   ; read CC
        BT      A.1,$EDCMDCHK           ; command or data ?
        MOV     A,!DATA                 ; IF command, THEN read headder code
        MOV     C,#4                    ;
        MOV     B,#0                    ;
OPPOSCHK:
        SHR     A,1                     ;
        BC      $CY_NOCNT               ;
        INC     B                       ; serch OP code position
CY_NOCNT:
        DBNZ    C,$OPPOSCHK             ;
        MOV     A,!RQRGCMD0             ;
        CMP     A,DATA+1[B]             ; request regist command ?
        BNZ     $CHK_RST                ;
        SET1    RGSTRQF                 ; set system up
        BR      EDCMDCHK                ;
CHK_RST:
        MOV     A,!SRSTCMD0             ;
        CMP     A,DATA+1[B]             ; system reset command ?
        BNZ     $EDCMDCHK               ;
        SET1    SYSRSTF                 ; set system reset
EDCMDCHK:
        RET                             ;

;---------------------------------------
;       DEFINE ROM DATA
;---------------------------------------

;*****  ADDRESS OF SYSTEM-UP MODE ROUTINE *****
;       << HBC SIDE >>

H_SUPADR:
        DW      H_STRGST                ;
        DW      H_RQRGST                ;
        DW      H_CFRGST                ;
        DW      H_EDRGST                ;

;*****  ADDRESS OF SYSTEM-UP MODE ROUTINE *****
;       << IFU SIDE >>

I_SUPADR:
        DW      I_STRGST                ;
        DW      I_RQRGST                ;
        DW      I_CFRGST                ;
        DW      I_EDRGST                ;
```

```
;*****  transmissive data of starting register *****

RGSTSTRT:
        DB      8                       ; number of transmissive data
        DB      11111100B               ; PR
        DB      0FFH                    ; SA (dummy)
        DB      0FFH                    ; DA (dummy)
        DB      00111101B               ; CC
        DB      3                       ; BC
        DB      01001111B               ; DATA
        DB      0A0H, 0FFH              ; DATA (address group is dummy)

;*****  transmissive data of request register *****

RGSTRQST:
        DB      11                      ; number of transmissive data
        DB      11111100B               ; PR
        DB      0FFH                    ; SA (dummy)
        DB      00000001B               ; DA (HBC)
        DB      00111101B               ; CC
        DB      6                       ; BC
        DB      01001111B               ; DATA
        DB      0A1H, 60H, 71H, 60H, 72H    ; DATA

;*****  transmissive data of confirming register *****

RGSTCNFM:
        DB      10                      ; number of transmissive data
        DB      10111100B               ; PR
        DB      0FFH                    ; SA (dummy)
        DB      0FFH                    ; DA (dummy)
        DB      00111101B               ; CC
        DB      9                       ; BC
        DB      01001111B               ; DATA
        DB      0A2H, 60H, 60H, 70H         ; DATA

        END
```

11
214
218A
234
244

(3) 送受信処理（モジュール名：TRNRCV）

```
$       TITLE   ('78K/II HOME-BUS CONTROL PROGRAM')
$       PROCESSOR(218A)
$       INCLUDE(INTMS.DEF)

        NAME    TRNRCV
;*************************************************************
;*      *INTC10 INTERRUPT                                    *
;*              TRANSMIT/RECEIVE ROUTINE                     *
;*      *INTC30 INTERRUPT                                    *
;*              ENTER RECOVERING SYNCRONISM                  *
;*              END RECOVERING SYNCRONISM                    *
;*              TIMEOUT ROUTINE                              *
;*      *INTP0 INTERRUPT                                     *
;*              DETECT START BIT ROUTINE                     *
;*      *INTP1 INTERRUPT                                     *
;*              REQUEST OF TRANSMISSION ROUTINE              *
;*      *INTP2 INTERRUPT                                     *
;*              REQUEST OF SHORT FRAME INTERRUPT ROUTINE*
;*************************************************************

        PUBLIC  MK_DATA
        PUBLIC  TRNBUF1,TRNDAT1,TRNBUF2,TRNDAT2,RCVBUF
        PUBLIC  BC_,DATA,ERROR,CTLCODE
        PUBLIC  TRN,RCV,WATCH,FRSTTIM,TMOUTTIM,MAXDAT
        PUBLIC  PRTYERF,CKSUMERF,FRMERF,NOANSERF
        PUBLIC  TMOUTERF,DMYDTERF,NAKERF,TRNERF,OVDATERF
        PUBLIC  TPR1,TSA1,TDA1,TBC1,TDATA1,TPR2,TSA2,TBC2,SA
        PUBLIC  CC,CHRCNT,RCVDAT
        PUBLIC  RQRGCMD0,SRSTCMD0

        EXTRN   MODE,DATCNT,BITCNT,TRBFPNT
        EXTRN   SELFADRS,GRPCNT,GRPBIT_R
        EXTRN   BUS_IDL
        EXTBIT  TRNENDF,RCVENDF,PAUSEF,PAUSE2F,BITF,INTCHRF
        EXTBIT  RQTRNF,RQTRN2F,RQINTF,MKDATF,BUSIDLF,INPUTDAT
        EXTBIT  TOSELFF,SYSRSTF,RGSTRQF,ACPTINTF,CMDCHKF
        EXTBIT  DTINTF,DTCLSHF,TRNINTF,CLRMDATF,SYSUPF
        EXTBIT  TRNRDYF,SYNCF,CMPTTRF,IC30F
        EXTBIT  EP0TRGF,PCCHKF
```

```
;*****  define data *****

RCVPORT  EQU    P2.1                ; data receive port
CMK30    EQU    MK0H.1              ; INTC30 mask flag
CIF30    EQU    IF0H.1              ; INTC30 request flag

TRRVTIM  EQU    19-1                ; trans/receive timing      :52us
READTIM  EQU    9-1                 ; data read timing          :26us
FRSTTIM  EQU    2                   ; from trigger to start transmission
                                    ;                           :8.1us
TMOUTTIM EQU    114-1               ; timeout interval : 10ms(fclk/512)
PAUSTIM1 EQU    112-1               ; pause time 1     :  9.79ms(fclk512)
PAUSTIM2 EQU    138-1               ; pause time 2     : 12.08ms(fclk512)
SYNCTIM  EQU    76-1                ; recovering syncronism time
                                    ;                  : 208.3us(fclk/16)
START    EQU    11111110B           ; start bit data
OFFDUTY  EQU    11111111B           ; off duty data
BREAK    EQU    11111110B           ; break data

BUFSIZE  EQU    30                  ; transmissive/receiving buffer size
MAXDAT   EQU    BUFSIZE-6           ; max size of DATA
ACKCOD   EQU    06H                 ; ack code
TRNTIMS  EQU    3                   ; re-transmit time number

TRN      EQU    1                   ; mode NO. of transmission
RCV      EQU    2                   ;             receiving
WATCH    EQU    3                   ;             watching

PRD      EQU    1                   ;
SAD      EQU    2                   ;
DAD      EQU    3                   ;
CCD      EQU    4                   ;
BCD      EQU    5                   ;
DATAD    EQU    6                   ;
FCCD     EQU    7                   ;
DMYD     EQU    8                   ;
ACKD_    EQU    9                   ;
```

11
214
218A
234
244

```
;*****  define transmit/receive buffer  *****

BUFFS   DSEG
TRNBUF1:                                ; transmit buffer 1
TPR1:   DS      1                       ;       PR area
TSA1:   DS      1                       ;       SA area
TDA1:   DS      1                       ;       DA area
TCC1:   DS      1                       ;       CC area
TBC1:   DS      1                       ;       BC area
TDATA1: DS      25                      ;       DATA area(include FCC area)

TRNBUF2:                                ; transmit buffer 2
TPR2:   DS      1                       ;       PR area
TSA2:   DS      1                       ;       SA area
TDA2:   DS      1                       ;       DA area
TCC2:   DS      1                       ;       CC area
TBC2:   DS      1                       ;       BC area
TDATA2: DS      25                      ;       DATA area(include FCC area)

RCVBUF:                                 ; receive buffer
PR:     DS      1                       ;       PR area
SA:     DS      1                       ;       SA area
DA:     DS      1                       ;       DA area
CC:     DS      1                       ;       CC area
BC_:    DS      1                       ;       BC area
DATA:   DS      25                      ;       DATA area(include FCC area)

TRNDAT1:DS      BUFSIZE*11              ; transmissive data area 1
TRNDAT2:DS      BUFSIZE*11              ; transmissive data area 2

;*****  define work area *****

        BSEG
BITSAV  DBIT                            ; receiving bit save area
PRTYSAV DBIT                            ; parity bit save area
OVDATERF DBIT                           ; over data error

I_WORKS DSEG    SADDR
RCVDAT: DS      1                       ; temporary area of receiving data
CHKSAV: DS      1                       ;                   check code
PRCODE: DS      1                       ; work area of PR
CHRCNT: DS      1                       ;             BC(character count)
CTLCODE: DS     1                       ;             CC
ERROR:  DS      1                       ; error flag area
```

```
;*****  define ERROR flag *****

PRTYERF   EQU    ERROR.0                ; parity error
CKSUMERF  EQU    ERROR.1                ; check sum error
FRMERF    EQU    ERROR.2                ; framing error
TMOUTERF  EQU    ERROR.3                ; timeout error
DMYDTERF  EQU    ERROR.4                ; dummy data error
NAKERF    EQU    ERROR.5                ; nak error
NOANSERF  EQU    ERROR.6                ; no answer error
TRNERF    EQU    ERROR.7                ; transmit error

;*****  define vector table  *****

          CVENT10                       ; INTC10 vector table
          CVENT30                       ; INTC30 vector table
          PVENT0                        ; INTP0 vector table
          PVENT1                        ; INTP1 vector table
          PVENT2                        ; INTP2 vector table

INTS      CSEG
;---------------------------------------
;         INTC10 INTERRUPT
;             TRANSMIT/RECEIVE ROUTINE
;                 per 52us
;---------------------------------------

INTC10:
          MOV     CR10,#TRRVTIM         ; set transmission/receiving timing
          SEL     RB1                   ; select register bank 1
          BF      PAUSEF,$TRN_RCV       ; pause start ?
          BR      PAUSMOD               ;
TRN_RCV:
          MOV     A,MODE                ; branch each mode
          DEC     A                     ;
          ADD     A,A                   ;
          MOV     X,A                   ;
          MOV     A,#00H                ;
          ADDW    AX,#MODADR            ; refer address of mode
          MOVW    HL,AX                 ;
          MOVW    AX,[HL]               ;
          BR      AX                    ; branch trn/rcv/wch mode
```

11
214
218A
234
244

```
;*****  transmissive mode *****

TRNMODE:
        NOT1    BITF                    ; set opposite flag
        BT      BITF,$T_BEFORE          ; before/after half bit ?
        BR      T_AFTER                 ;

;       << before half bit >>

T_BEFORE:
        CMP     DATCNT,#DMYD            ; dummy ?
        BNZ     $T_SETTRD               ;
        BR      T_BITCNT                ;
T_SETTRD:
        MOV     A,POL                   ; save POL data to temporary register
        MOV     C,A                     ;
        MOV     POL,#OFFDUTY            ; set off duty data to POL
        BF      INTCHRF,$T_CHKPR        ; interrupting short frame ?
        BT      ACPTINTF,$T_CHKPR       ; accepting interrupt ?
        BR      T_RETC10                ;
T_CHKPR:
        CMP     DATCNT,#DAD             ; PR or SA ?
        BC      $T_SAVBIT               ;
        BR      T_CHKDAT                ;

T_SAVBIT:
        MOV     A,TM1                   ;
        CMP     A,#READTIM              ; passed 26us ?
        BC      $T_SAVBIT               ;
        MOV1    CY,RCVPORT              ; read receiving port
        MOV1    BITSAV,CY               ;
        CMP     BITCNT,#11              ; start bit ?
        BNZ     $T_PRTY                 ;
        CLR1    PRTYSAV                 ; clear parity save area
        BR      T_BITCNT                ;
T_PRTY:
        CMP     BITCNT,#3               ; parity/stop bit ?
        BC      $CHKCLASH               ; ELSE
        MOV1    CY,BITSAV               ; parity check
        XOR1    CY,PRTYSAV              ;
        MOV1    PRTYSAV,CY              ; save parity bit
        MOV     A,RCVDAT                ; read receiving data before
        MOV1    CY,BITSAV               ;
        RORC    A,1                     ;
        MOV     RCVDAT,A                ; save receiving data to temporary area

CHKCLASH:
        MOV     A,C                     ;
        OR      A,#11111100B            ;
        CMP     A,#OFFDUTY              ; off duty data ?
        BNZ     $T_BITCNT               ; ELSE not check data clashing
```

```
        SET1    CY                      ;
        XOR1    CY,BITSAV               ; check data clashing
        BNC     $T_BITCNT               ; clash data ?
        MOV     MODE,#RCV               ; THEN set receiving mode
        MOV     RTPC,#00000000B         ; stop realtime output
        SET1    DTCLSHF                 ; set data clash
        SET1    TRNRDYF                 ; set transmit ready
        BR      T_BITCNT                ;

T_CHKDAT:
        CMP     DATCNT,#DATAD           ; data field ?
        BNZ     $T_BITCNT               ;

;       << check short frame interrupt >>

        MOV     A,PRCODE                ;
        OR      A,#11111100B            ;
        CMP     A,#11111111B            ; long frame ?
        BNZ     $T_BITCNT               ;
        CMP     BITCNT,#1               ; stop bit ?
        BNZ     $T_CKPRTY               ;
T_WAIT26:
        MOV     A,TM1                   ;
        CMP     A,#READTIM              ; passed 26us ?
        BC      $T_WAIT26               ;
        MOV1    CY,RCVPORT              ; read receiving data
        BC      $T_BITCNT               ; short frame interrupt ?
        SET1    TRNRDYF                 ; set transmit ready
        SET1    DTINTF                  ; set detecting short frame interrupt
        MOV     DATCNT,#10              ; THEN set end data to data counter
        BR      T_PAUS2                 ;

T_CKPRTY:
        CMP     BITCNT,#2               ; parity bit ?
        BNZ     $T_BITCNT               ;
        BF      RQINTF,$T_BITCNT        ; THEN request of short frame interrupt?
        SET1    INTCHRF                 ; THEN set short frame interrupting
        SET1    TRNINTF                 ;      set interrupt in transmission
        SET1    TRNRDYF                 ; transmissive ready
        SET1    CLRMDATF                ; set clear flag of making data end
T_PAUS2:
        SET1    PAUSE2F                 ; set pause term
        BR      T_BITINI                ;

T_BITCNT:
        DEC     BITCNT                  ;
        BZ      $T_EACHDT               ; end 1 packet ?
        BR      T_RETC10                ;
```

11
214
218A
234
244

```
;       << branch each routine by data >>
T_EACHDT:
        MOV     A,DATCNT                ; read data kind
        DEC     A                       ;
        ADD     A,A                     ;
        MOV     X,A                     ;
        MOV     A,#00H                  ;
        ADDW    AX,#T_DATADR            ;
        MOVW    HL,AX                   ;
        MOVW    AX,[HL]                 ;
        BR      AX                      ; branch each data routine

;       << each data routine >>

;       < PR >
T_PR:                                   ; DATCNT=1
        MOV     A,RCVDAT                ; read receiving data
        MOV     !PR,A                   ; save PR to receiving buffer
        BT      ACPTINTF,$INT_PR        ; accepting interrupt ?
        MOV     A,!TPR1                 ; read PR from transmissive buffer 1
        BR      SAVPR                   ;
INT_PR:
        MOV     A,!TPR2                 ; read PR from transmissive buffer 2
SAVPR:
        MOV     PRCODE,A                ; save PR to work area
        BR      T_DATCNT                ;

;       < SA >
T_SA:                                   ; DATCNT=2
        MOV     A,RCVDAT                ; read receiving data
        MOV     !SA,A                   ; save SA to receiving buffer
        BR      T_DATCNT                ;

;       < DA >
T_DA:                                   ; DATCNT=3
        BR      T_DATCNT                ;

;       < CC >
T_CC:                                   ; DATCNT=4
        BT      ACPTINTF,$INT_CC        ; accepting interrupt ?
        MOV     A,!TCC1                 ; read CC from transmissive buffer1
        BR      SAVCTL                  ;
INT_CC:
        MOV     A,!TCC2                 ; read CC from transmissive buffer2
SAVCTL:
        AND     A,#11000000B            ;
        ROL     A,2                     ; only re-transmit time
        MOV     CTLCODE,A               ; save CC to work area
        BR      T_DATCNT                ;
```

```
;       < BC >
T_BC:                                   ; DATCNT=5
        MOV     CHRCNT,#0               ; initialize character counter
        BR      T_DATCNT                ;

;       < DATA >
T_DATA:                                 ; DATCNT=6
        INC     CHRCNT                  ; count length of character
        BT      ACPTINTF,$INT_DAT       ; accepting interrupt ?
        MOV     A,!TBC1                 ; read BC from transmissive buffer1
        BR      CHK_MAX                 ;
INT_DAT:
        MOV     A,!TBC2                 ; read BC from transmissive buffer2
CHK_MAX:
        CMP     A,#MAXDAT+1             ; DATA size over ?
        BC      $CHK_CHR                ;
        MOV     A,#MAXDAT               ; set max size
CHK_CHR:
        CMP     A,CHRCNT                ; end character ?
        BNZ     $T_BITINI               ;
        BR      T_DATCNT                ;

;       < FCC >
T_FCC:                                  ; DATCNT=7
        BF      PRCODE.6,$T_DATCNT      ; announce together ?
        BT      INTCHRF,$T_CLRINT       ; interrupting short frame ?
        BT      RQTRN2F,$T_CLRTR2       ; next request of transmission ?
        CLR1    RQTRNF                  ; clear request of transmission
        BR      T_SETMDT                ;
T_CLRINT:
        CLR1    RQINTF                  ; clear request of interrupt
        BF      TRNINTF,$T_SETMDT       ; interrupt in transmission ?
        BR      T_SETCNT                ;
T_CLRTR2:
        CLR1    RQTRN2F                 ; clear second request of transmission
        SET1    TRNRDYF                 ; set transmissive ready flag
T_SETMDT:
        SET1    CLRMDATF                ; set clear flag of making data end
T_SETCNT:
        MOV     DATCNT,#ACKD_           ; set ACK data to data counter
        SET1    PAUSE2F                 ; set pause term
        BR      T_DATCNT                ;

;       < DMY >
T_DMY:
        BR      T_DATCNT                ;
```

11
214
218A
234
244

```
;       < ACK/NAK >
T_ACK:
        BTCLR   DTCLSHF,$T_DATCNT       ; defeat by clashed (detect clashing )?
T_DATCNT:
        INC     DATCNT                  ; count data kind
T_BITINI:
        MOV     BITCNT,#11              ; set bit counter

        BR      T_RETC10                ; return
```

```
        << after half bit >>

T_AFTER:
        CMP     DATCNT,#DMYD            ; dummy ?
        BNZ     $T_CHKEND               ;
        BF      P0ML,$BR_RET            ; IF in realtime output
        MOV     RTPC,#00000000B         ;    THEN stop realtime output
BR_RET:
        BR      T_RETC10                ;
T_CHKEND:
        CMP     DATCNT,#10              ; end 1 frame ?
        BZ      $T_FRMEND               ;
        CMP     DATCNT,#ACKD_           ; ack/nak ?
        BZ      $T_PKTEND               ;
OUT_ACK:
        BF      INTCHRF,$SET_AFTR       ; short frame interrupt ?
        BT      ACPTINTF,$SET_AFTR      ; accepting interrupt ?
        MOV     DATCNT,#10              ;   THEN count data kind
        MOV     P0L,#BREAK              ;        set break data to P0L
        BR      T_RETC10                ;
SET_AFTR:
        MOVW    AX,TRBFPNT              ;    ELSE set transmissive data to P0L
        MOVW    HL,AX                   ;
        MOV     A,[HL+]                 ;
        MOV     P0L,A                   ;
        MOVW    AX,HL                   ;
        MOVW    TRBFPNT,AX              ;
        BR      T_RETC10                ;
T_PKTEND:
        CMP     BITCNT,#11              ; end 1 packet ?
        BNZ     $SET_AFTR               ;
        MOV     MODE,#RCV               ; set to receiving mode
        MOV     CR30,#TMOUTTIM          ; set timeout time
        AND     TMC1,#11110111B         ; stop TM1
        SET1    CMK10                   ; disable INTC10
        CLR1    CMK30                   ; enable INTC30
        CLR1    PIF0                    ; clear request of INTP0
        CLR1    PMK0                    ; enable INTP0
        MOV     PRM0,#01110000B         ; set TM3 prescaler fclk/512
        OR      TMC0,#10000000B         ; start TM3
        BR      T_RETC10                ;
```

11
214
218A
234
244

```
;       << end of 1 frame >>

T_FRMEND:
        MOV     DATCNT,#PRD             ; set initial data(PR) to data counter
        BF      INTCHRF,$T_RETMOD       ; interrupting short frame ?
        BT      ACPTINTF,$T_CRMKDF      ; accepting interrupt ?
        SET1    ACPTINTF                ; set accept interrupt
        BF      TRNINTF,$T_RCV          ; interrupt in transmissing ?
        BR      T_CRMKDF                ;
T_RETMOD:
        BT      DTINTF,$T_CRMKDF        ; detect short frame interrupt ?
        BT      PRCODE.6,$T_CRMKDF      ; announce together ?
T_RCV:
        MOV     MODE,#RCV               ; ELSE set receiving mode
T_CRMKDF:
        BF      CLRMDATF,$T_MODE        ; enable clearing making data end flag?
        CLR1    CLRMDATF                ; THEN clear        "
        CLR1    MKDATF                  ; clear end flag of making data
T_MODE:
        CMP     MODE,#TRN               ;
        BZ      $TPTRNMOD               ;
        BR      TPRCVMOD                ;

TPTRNMOD:
        BTCLR   TRNRDYF,$TTRNRDY1       ; transmissive ready ?
        BF      INTCHRF,$TTRETPNT       ; interrupting short frame ?
        CLR1    ACPTINTF                ; cansel accepting of short frame
        CLR1    INTCHRF                 ; clear interrupting of short frame
        BF      TRNINTF,$TTRETPNT       ; interrupt in transmitting ?
        CLR1    TRNINTF                 ;
        BR      T_SETIN                 ;
TTRNRDY1:
        BT      INTCHRF,$TSETBUF2       ; interrupting short frame ?
T_SETIN:
        SET1    INPUTDAT                ; set data input end flag
TTRETPNT:
        MOVW    TRBFPNT,#TRNDAT1        ; return pointer to trans-data buffer1
        BR      STP_TRN                 ;
TSETBUF2:
        SET1    INPUTDAT                ; set data input end flag
        MOVW    TRBFPNT,#TRNDAT2        ; return pointer to trans-data buffer2
        BR      STP_TRN

TPRCVMOD:
        BF      TRNRDYF,$T_RETPNT       ; transmissive ready ?
        CLR1    TRNRDYF                 ; clear transmissive ready flag
        SET1    INPUTDAT                ; set data input end flag
T_RETPNT:
        BT      INTCHRF,$TRCVBUF2       ; short frame interrupt ?
        MOVW    TRBFPNT,#TRNDAT1        ; return pointer to trans-data buffer1
        BR      STP_TRN                 ;
```

```
TRCVBUF2:
        MOVW    TRBFPNT,#TRNDAT2        ; return pointer to trans-data buffer2
STP_TRN:
        MOV     POL,#START              ; set first transmissive data
        MOV     RTPC,#00000000B         ; disable realtime output
        SET1    PAUSEF                  ; set pause mode

T_RETC10:
        BR      RETC10                  ; return
```

11
214
218A
234
244

```
;*****  receiving mode *****

RCVMODE:
        NOT1    BITF                    ; set opposite flag
        BT      BITF,$R_BEFORE          ; before/after half bit ?
        BR      R_AFTER                 ;

;       << before half bit >>

R_BEFORE:
        CMP     DATCNT,#DMYD            ; dummy ?
        BNZ     $CK_NOANS               ;
        BR      R_BITCNT                ;
CK_NOANS:
        BF      NOANSERF,$R_WAIT26      ; no answer error ?
        BR      R_EACHDT                ;
R_WAIT26:
        MOV     A,TM1                   ;
        CMP     A,#READTIM              ; passed 26us ?
        BC      $R_WAIT26               ;
        MOV1    CY,RCVPORT              ; read receiving data
        MOV1    BITSAV,CY               ; save receiving data to temporary bit

        CMP     BITCNT,#1               ; stop bit ?
        BNZ     $PRTYCHK                ;
        CMP     DATCNT,#DATAD           ; data field ?
        BNZ     $CHKSTP                 ;
        MOV     A,PRCODE                ; read priority code
        AND     A,#00000011B            ;
        CMP     A,#00000011B            ; long frame ?
        BNZ     $R_BITCNT               ;
        MOV1    CY,BITSAV               ; read receiving data
        BC      $R_BITCNT               ; short frame interrupt ?
        SET1    DTINTF                  ; set detecting interrupt flag
        SET1    TRNRDYF                 ; set transmissive ready flag
        MOV     DATCNT,#10              ; THEN set end data to data counter
        BR      R_PAUS2                 ;
CHKSTP:
        BT      BITSAV,$R_BITCNT        ;
        SET1    FRMERF                  ; set framing error flag
        BR      R_BITCNT                ;
```

```
;       << calculation of parity bit >>

PRTYCHK:
        CMP     BITCNT,#2               ; parity bit ?
        BZ      $PRERCHK                ; THEN check parity
        MOV1    CY,BITSAV               ; read receiving bit
        XOR1    CY,PRTYSAV              ; calculate parity
        MOV1    PRTYSAV,CY              ; save parity bit
        MOV     A,RCVDAT                ; ELSE read receiving data before part
        MOV1    CY,BITSAV               ;
        RORC    A,1                     ;
        MOV     RCVDAT,A                ; save receiving data to temporary area
        BR      R_BITCNT                ;

;       << parity error check >>

PRERCHK:
        MOV1    CY,BITSAV               ;
        XOR1    CY,PRTYSAV              ; receiving data = parity data ?
        BNC     $R_CHKINT               ;
        SET1    PRTYERF                 ; ELSE set parity error flag
R_CHKINT:
        CMP     DATCNT,#DATAD           ; THEN data field ?
        BNZ     $R_BITCNT               ;
        MOV     A,PRCODE                ; THEN
        OR      A,#11111100B            ;
        CMP     A,#11111111B            ;       long frame ?
        BNZ     $R_BITCNT               ;
        BF      RQINTF,$R_BITCNT        ; THEN
        SET1    INTCHRF                 ; ELSE set interrupting short frame
        SET1    TRNRDYF                 ; transmissive ready
        SET1    CLRMDATF                ; set clear flag of making data end
R_PAUS2:
        SET1    PAUSE2F                 ; set pause term
        BR      R_BITINI                ;

R_BITCNT:
        DEC     BITCNT                  ; count bit number
        BZ      $R_CALCHK               ; end 1 packet ?
        BR      R_RETC10                ;
R_CALCHK:
        CMP     DATCNT,#FCCD            ; range of PR - DATA ?
        BNC     $R_EACHDT               ;
        CMP     DATCNT,#SAD             ;
        BC      $R_EACHDT               ;
        ADD     CHKSAV,RCVDAT           ; calculate check code
```

11
214
218A
234
244

```
;       << branch each routine by data >>

R_EACHDT:
        MOV     A,DATCNT                ; read data kind
        DEC     A                       ;
        ADD     A,A                     ;
        MOV     X,A                     ;
        MOV     A,#00H                  ;
        ADDW    AX,#R_DATADR            ;
        MOVW    HL,AX                   ;
        MOVW    AX,[HL]                 ;
        BR      AX                      ; branch each data routine

;       << each data routine >>

;       < PR >
R_PR:                                   ; DATCNT=1
        MOV     A,RCVDAT                ; read receiving data
        MOV     !PR,A                   ; save PR to receiving buffer
        MOV     PRCODE,A                ; save PR to work area
        MOV     CHKSAV,#0               ; clear check sum save area
        BR      R_DATCNT                ;

;       < SA >
R_SA:                                   ; DATCNT=2
        MOV     A,RCVDAT                ; read receiving data
        MOV     !SA,A                   ; save SA to receiving buffer
        BR      R_DATCNT                ;

;       < DA >
R_DA:                                   ; DATCNT=3
        MOV     A,RCVDAT                ; read receiving data
        MOV     !DA,A                   ; save DA to receiving buffer
        BF      PRCODE.6,$CHK_SELF      ; announce together ?
        MOV     A,SELFADRS              ; read self address
        MOV     A,GRPBIT_R[A]           ; read address group bit
        MOV     B,A                     ;
        MOV     A,RCVDAT                ; read receiving data
        AND     A,B                     ;
        CMP     A,B                     ; to self address group ?
        BZ      $R_TOSELF               ;
        BR      R_DATCNT                ;
CHK_SELF:
        CMP     A,SELFADRS              ; to self ?
        BZ      $R_TOSELF               ;
        BR      R_DATCNT                ;
R_TOSELF:
        SET1    TOSELFF                 ; set to-self flag
        BR      R_DATCNT                ;
```

```
;       < CC >
R_CC:                                   ; DATCNT=4
        MOV     A,RCVDAT                ; read receiving data
        MOV     !CC,A                   ; save CC to receiving buffer
        AND     A,#11000000B            ;
        ROL     A,2                     ; transmission time
        MOV     CTLCODE,A               ; save CC to work area
        BR      R_DATCNT                ;

;       < BC >
R_BC:                                   ; DATCNT=5
        MOV     A,RCVDAT                ; read receiving data
        CMP     A,#MAXDAT+1             ; DATA size over ?
        BC      $R_SAVBC                ;
        MOV     A,#MAXDAT               ; set max size
R_SAVBC:
        MOV     !BC_,A                  ; save BC to receiving buffer
        MOV     CHRCNT,#0               ; initialize character counter
        BF      TOSELFF,$R_SETWT        ; IF to self, THEN clear to-self flag
        CLR1    TOSELFF                 ; clear to-self flag
        BR      R_DATCNT                ;
R_SETWT:
        MOV     MODE,#WATCH             ; set watching mode
        BR      R_DATCNT                ;

;       < DATA >
R_DATA:                                 ; DATCNT=6
        MOV     A,CHRCNT                ;
        MOV     B,A                     ;
        MOV     A,RCVDAT                ; read receiving data
        MOV     DATA[B],A               ; save DATA to receiving buffer
        INC     CHRCNT                  ; count length of character
        MOV     A,!BC_                  ;
        CMP     A,CHRCNT                ; end character ?
        BNZ     $R_BIT                  ;
        BR      R_DATCNT                ;
R_BIT:
        BR      R_BITINI                ;
```

11
214
218A
234
244

```
;       < FCC >
R_FCC:                                  ; DATCNT=7
        MOV     A,!BC_                  ; read character count
        MOV     B,A                     ;
        MOV     A,RCVDAT                ; read receiving data
        MOV     DATA[B],A               ; save FCC to receiving buffer
        XOR     CHKSAV,#0FFH            ; reverse calculative data
        INC     CHKSAV                  ; calculative data +1
        CMP     RCVDAT,CHKSAV           ; receiving data = calculative data ?
        BZ      $CHK_ANN                ;
        SET1    CKSUMERF                ; set check-sum error
CHK_ANN:
        BT      PRCODE.6,$R_CLRCLS      ; announce together ?
        BR      R_DATCNT                ; ELSE
R_CLRCLS:
        BTCLR   DTCLSHF,$R_SETACK       ; defeat by clashed (detect clashing )?
R_SETACK:
        MOV     DATCNT,#ACKD_           ; THEN
        SET1    PAUSE2F                 ; set pause term
        BR      R_DATCNT                ;

;       < DMY >
R_DMY:                                  ; DATCNT=8
        MOV     A,ERROR                 ;
        AND     A,#00011111B            ; check receiving error
        CMP     A,#00000000B            ; no receiving error ?
        BNZ     $CHK_RVER               ;
        MOVW    AX,#ACKDAT              ; THEN set ack data
        BR      ACKPNT                  ;
CHK_RVER:
        MOV     A,ERROR                 ;
        AND     A,#00010100B            ; check framing & dummy data
        CMP     A,#00000000B            ; check framing or dummy data error ?
        BZ      $PRYSUMER               ;
        MOVW    AX,#NOANSDAT            ; set no-answer data
        BR      ACKPNT                  ;
PRYSUMER:
        MOVW    AX,#NAKDAT              ; set nak data
ACKPNT:
        MOVW    TRBFPNT,AX              ; set ack/nak data address
        BR      R_DATCNT                ;
```

```
;       < ACK/NAK >
R_ACK:
        BT      NOANSERF,$CK_TRAGN      ; no answer error ?
        CMP     RCVDAT,#ACKCOD          ; ELSE receiving data is ack code ?
        BZ      $CLR_CTL                ;
        SET1    NAKERF                  ; IF nak, set nak error
CK_TRAGN:
        INC     CTLCODE                 ; increment re-transmit time
        CMP     CTLCODE,#TRNTIMS+1      ; transmited 3 times ?
        BC      $TRNAGN                 ; enable transmit again ?
        SET1    TRNERF                  ; set transmission error
        BR      CLR_CTL                 ;

;       << transmit again >>

TRNAGN:
        BT      INTCHRF,$TRAGNINT       ; interrupting short frame ?
        MOV     A,!TPR1                 ; ELSE read PR1
        CLR1    A.7                     ; set PR re-transmit mode
        MOV     !TPR1,A                 ;
        MOV     A,!TCC1                 ; read CC1
        AND     A,#00111111B            ; mask re-transmit time
        MOV     B,A                     ;
        MOV     A,CTLCODE               ; read re-transmit time
        ROR     A,2                     ;
        OR      A,B                     ; re-transmit time U CC1
        MOV     !TCC1,A                 ; save new CC1
        BR      TRN_RDY2                ;

TRAGNINT:
        MOV     A,!TPR2                 ; ELSE read PR2
        CLR1    A.7                     ; set PR re-transmit mode
        MOV     !TPR2,A                 ;
        MOV     A,!TCC2                 ; read CC2
        AND     A,#00111111B            ; mask re-transmit time
        MOV     B,A                     ;
        MOV     A,CTLCODE               ; read re-transmit time
        ROR     A,2                     ;
        OR      A,B                     ; re-transmit time U CC2
        MOV     !TCC2,A                 ; save new CC2
TRN_RDY2:
        SET1    TRNRDYF                 ; set re-transmissive flag
        BR      R_SETMDT                ;

CLR_CTL:
        MOV     CTLCODE,#0              ; clear CC work area
        BT      INTCHRF,$R_CLRINT       ;
        BT      RQTRN2F,$R_CLRTR2       ;
        CLR1    RQTRNF                  ; clear request of transmission
        BR      R_SETMDT                ;
```

11
214
218A
234
244

```
R_CLRINT:
        CLR1    RQINTF                  ; clear request of interrupt of frame
        BF      TRNINTF,$R_SETMDT       ; interrupt in transmission ?
        BR      R_DATCNT                ;
R_CLRTR2:
        CLR1    RQTRN2F                 ; clear second request of transmission
        SET1    TRNRDYF                 ; set transmissive ready flag
R_SETMDT:
        SET1    CLRMDATF                ; set clear flag of making data end
        BR      R_DATCNT                ;

R_DATCNT:
        INC     DATCNT                  ; count data kind
R_BITINI:
        MOV     BITCNT,#11              ; set bit counter
        BR      R_RETC10                ; return
```

```
;       << after half bit >>

R_AFTER:
        CMP     DATCNT,#DMYD            ; dummy ?
        BNZ     $R_ENDCHK               ;
        BT      PMK0,$R_DMYP0           ; prohibit INTP0 ?
        BR      R_RETC10                ;
R_DMYP0:
        CLR1    PIF0                    ; clear INTP0 request flag
        CLR1    PMK0                    ; enable INTP0
        BR      R_RETC10                ;
R_ENDCHK:
        CMP     DATCNT,#10              ; end of 1 frame ?
        BZ      $R_FRMEND               ; THEN branch frame end
        CMP     DATCNT,#ACKD_           ; ack/nak ?
        BZ      $R_CHGPNT               ;
        BR      R_SETBRK                ;
R_CHGPNT:
        CMP     BITCNT,#11              ; end 1packet ?
        BNZ     $R_PKTEND               ;
        SET1    PMK0                    ; prohibit INTP0
        MOVW    AX,TRBFPNT              ; THEN set ack/nak data to P0L
        MOVW    HL,AX                   ;
        MOV     A,[HL+]                 ;
        MOV     P0L,A                   ;
        MOVW    AX,HL                   ;
        MOVW    TRBFPNT,AX              ;
        BR      R_SETTRN                ;
R_SETBRK:
        BF      INTCHRF,$R_PKTEND       ; ELSE interrupting short frame ?
        BT      ACPTINTF,$R_PKTEND      ; accepting short frame interrupt ?
        MOV     DATCNT,#10              ;    THEN reset data kind
        MOV     P0L,#BREAK              ; set break data to P0L
R_SETTRN:
        MOV     MODE,#TRN               ; set transmissive mode
        MOV     RTPC,#00000001B         ; enable realtime output
        BR      R_RETC10                ;
R_PKTEND:
        CMP     BITCNT,#11              ; end 1 packet ?
        BZ      $R_TMOUT                ;
        BR      R_RETC10                ;
R_TMOUT:
        MOV     CR30,#TMOUTTIM          ; set timeout time
        AND     TMC1,#11110111B         ; stop TM1
        SET1    CMK10                   ; disable INTC10
        CLR1    CMK30                   ; enable INTC30
        CLR1    PIF0                    ; clear request of INTP0
        CLR1    PMK0                    ; enable INTP0
        MOV     PRM0,#01110000B         ; set TM3 prescaler fclk/512
        OR      TMC0,#10000000B         ; start TM3
        BR      R_RETC10                ;
```

11
214
218A
234
244

```
;       << end of 1 frame >>

R_FRMEND:
        MOV     DATCNT,#PRD             ; set initial data to data counter
        BT      DTINTF,$R_CRMKDF        ; detect short frame interrupt ?
        BT      PRCODE.6,$R_CRMKDF      ; announce together ?
        MOV     MODE,#TRN               ; ELSE set transmissive mode
R_CRMKDF:
        BF      CLRMDATF,$R_MODE        ; enable clearing making data end flag?
        CLR1    CLRMDATF                ; THEN clear        "
        CLR1    MKDATF                  ; clear end flag of making data
R_MODE:
        CMP     MODE,#TRN               ;
        BZ      $RPTRNMOD               ;
        BR      RPRCVMOD                ;

RPTRNMOD:
        BTCLR   TRNRDYF,$RTRNRDY1       ; transmissive ready ?
        BF      INTCHRF,$RTRETPNT       ; interrupting short frame ?
        CLR1    ACPTINTF                ; cansel accepting of short frame
        CLR1    INTCHRF                 ; clear interrupting of short frame
        BF      TRNINTF,$RTRETPNT       ; interrupt in transmitting ?
        CLR1    TRNINTF                 ;
        BR      R_SETIN                 ;
RTRNRDY1:
        BT      INTCHRF,$RSETBUF2       ; interrupting short frame ?
R_SETIN:
        SET1    INPUTDAT                ; set data input end flag
RTRETPNT:
        MOVW    TRBFPNT,#TRNDAT1        ; return pointer to trans-data buffer1
        BR      R_SETPS                 ;
RSETBUF2:
        SET1    INPUTDAT                ; set data input end flag
        MOVW    TRBFPNT,#TRNDAT2        ; return pointer to trans-data buffer2
        BR      R_SETPS                 ;
RPRCVMOD:
        BF      TRNRDYF,$R_RETPNT       ; transmissive ready ?
        CLR1    TRNRDYF                 ; clear transmissive ready flag
        SET1    INPUTDAT                ; set data input end flag
R_RETPNT:
        BT      INTCHRF,$RRCVBUF2       ; short frame interrupt ?
        MOVW    TRBFPNT,#TRNDAT1        ; return pointer to trans-data buffer1
        BR      R_SETPS                 ;
RRCVBUF2:
        MOVW    TRBFPNT,#TRNDAT2        ; return pointer to trans-data buffer2

R_SETPS:
        MOV     POL,#START              ; set first transmissive data
        SET1    PAUSEF                  ; set pause mode
R_RETC10:
        BR      RETC10                  ;
```

```
;*****  watching mode *****

WCHMODE:
        NOT1    BITF                    ; set opposite flag
        BT      BITF,$W_BEFORE          ; before/after half bit ?
        BR      W_AFTER                 ;

;       << before half bit >>

W_BEFORE:
        CMP     DATCNT,#DMYD            ; dummy ?
        BZ      $W_BITCNT               ;
        CMP     DATCNT,#DATAD           ; data field ?
        BNZ     $W_BITCNT               ; ELSE count bit

;       << check short frame interrupt >>

        MOV     A,PRCODE                ; THEN long frame ?
        AND     A,#00000011B            ;
        CMP     A,#00000011B            ;   ELSE count bit
        BNZ     $W_BITCNT               ;
        CMP     BITCNT,#1               ;   THEN stop bit ?
        BNZ     $W_CKPRTY               ;      ELSE check parity bit
W_WAIT26:
        MOV     A,TM1                   ;      THEN waiting 26us
        CMP     A,#READTIM              ; passed 26us ?
        BC      $W_WAIT26               ;
        MOV1    CY,RCVPORT              ; read receiving data
        MOV1    BITSAV,CY               ; save receiving data
        BT      BITSAV,$W_BITCNT        ; interrupt of short frame ?
        SET1    DTINTF                  ; set detect interrupt flag
        SET1    TRNRDYF                 ; set transmit ready
        MOV     DATCNT,#10              ; THEN set frame end data
        BR      W_PAUS2                 ;

W_CKPRTY:
        CMP     BITCNT,#2               ; parity bit ?
        BNZ     $W_BITCNT               ; ELSE count bit
        BF      RQINTF,$W_BITCNT        ; request of short frame interrupt ?
        SET1    INTCHRF                 ; THEN set interrupting short frame
        SET1    TRNRDYF                 ; transmissive ready
        SET1    CLRMDATF                ; set clear flag of making data end
W_PAUS2:
        SET1    PAUSE2F                 ; set pause term
        BR      W_BITINI                ;

W_BITCNT:
        DEC     BITCNT                  ; count bit counter
        BZ      $W_CHKFCC               ; end transmission of 1 packet ?
        BR      W_RETC10                ;
```

11
214
218A
234
244

```
W_CHKFCC:
        CMP     DATCNT,#DATAD           ; data field ?
        BNZ     $W_FCC                  ;
        INC     CHRCNT                  ; count length of character
        MOV     A,!BC_                  ;
        CMP     A,CHRCNT                ; end character ?
        BNZ     $W_BITINI               ;
W_FCC:
        CMP     DATCNT,#FCCD            ; check code ?
        BNZ     $W_ACK                  ;
        BTCLR   DTCLSHF,$W_CHKANN       ; detect data clashed ?
W_CHKANN:
        BF      PRCODE.6,$W_DATCNT      ; THEN announce together ?
        MOV     DATCNT,#ACKD_           ; set data of ack/nak part
        SET1    PAUSE2F                 ; set pause term
        BR      W_DATCNT                ;
W_ACK:
        CMP     DATCNT,#ACKD_           ; ack/nak ?
        BNZ     $W_DATCNT               ;
        BTCLR   DTCLSHF,$W_DATCNT       ; detect data clashed ?
W_DATCNT:
        INC     DATCNT                  ; count data kind
W_BITINI:
        MOV     BITCNT,#11              ; reset bit counter
        BR      W_RETC10                ;
```

```
;       << after half bit >>

W_AFTER:
        CMP     DATCNT,#DMYD            ; dummy ?
        BNZ     $W_CHKEND               ;
        BR      W_RETC10
W_CHKEND:
        CMP     DATCNT,#10              ; end of 1 frame ?
        BZ      $W_FRMEND               ; THEN branch frame end
        BF      INTCHRF,$W_PKTEND       ; ELSE interrupting short frame ?
        BT      ACPTINTF,$W_PKTEND      ; accepting short frame interrupt ?
        MOV     DATCNT,#10              ;    THEN reset data kind
        MOV     P0L,#BREAK              ; set break data to P0L
        MOV     MODE,#TRN               ; set transmissive mode
        MOV     RTPC,#00000001B         ; enable rialtime output
        BR      W_RETC10                ;

W_PKTEND:
        CMP     BITCNT,#11              ; end 1 packet ?
        BZ      $W_TMOUT                ;
        BR      W_RETC10                ;
W_TMOUT:
        MOV     CR30,#TMOUTTIM          ; set timeout time
        AND     TMC1,#11110111B         ; stop TM1
        SET1    CMK10                   ; disable INTC10
        CLR1    CMK30                   ; enable INTC30
        CLR1    PIF0                    ; clear request of INTP0
        CLR1    PMK0                    ; enable INTP0
        MOV     PRM0,#01110000B         ; set TM3 prescaler fclk/512
        OR      TMC0,#10000000B         ; start TM3
        BR      W_RETC10

;       << end of 1 frame >>

W_FRMEND:
        MOV     DATCNT,#1               ; set initial data to data counter
        BF      TRNRDYF,$W_RETPNT       ; transmissive ready ?
        CLR1    TRNRDYF                 ;
        SET1    INPUTDAT                ; set data input end flag
        BT      INTCHRF,$WSETBUF2       ; interrupting short frame ?
W_RETPNT:
        MOVW    TRBFPNT,#TRNDAT1        ; return pointer to trans-data buffer1
        BR      W_SETPS                 ;
WSETBUF2:
        MOVW    TRBFPNT,#TRNDAT2        ; return pointer to trans-data buffer2
W_SETPS:
        MOV     P0L,#START              ; set first transmissive data
        SET1    PAUSEF                  ; set pause mode

W_RETC10:
        BR      RETC10                  ;
```

11
214
218A
234
244

```
;*****  entering pause mode *****

PAUSMOD:
        AND     TMC1,#11110111B         ; stop TM1
        BTCLR   PAUSE2F,$SET_PS2        ; pause time 2 ?
        MOV     CR30,#PAUSTIM1          ; ELSE set puase time 1 :
        BR      CLR_C30                 ;
SET_PS2:
        MOV     CR30,#PAUSTIM2          ; THEN set puase time 2 :
CLR_C30:
        SET1    CMK10                   ; disable INTC10
        CLR1    CMK30                   ; enable INTC30
        CLR1    PIF0                    ; clear request of INTP0
        CLR1    PMK0                    ; enable INTP0
        MOV     PRM0,#01110000B         ; set TM3 prescaler fclk/512
        OR      TMC0,#10000000B         ; start TM3

RETC10:
        RETI
```

```
;---------------------------------------
;       DETECT START BIT ROUTINE
;               by INTP0 interrupt
;---------------------------------------

INTP0:
        SEL     RB1                     ; select register bank 1
        BTCLR   SYNCF,$CMPT_TRN         ; recovering syncronism ?
        BF      BUSIDLF,$STRT_RCV       ; bus idle ?
CMPT_TRN:
        BF      MKDATF,$STRT_RCV        ; making data end ?
        BF      EP0TRGF,$STRT_RCV       ; INTP0 trigger enable ?
        MOV     CR10,#TRRVTIM-3         ; set first pulse width
        OR      TMC1,#00001000B         ; start TM1
        MOV     RTPC,#00000001          ; enable realtime output
        CLR1    EP0TRGF                 ; clear INTP0 trigger enable flag
        AND     TMC0,#01111111B         ; stop TM3
        MOV     P0L,#OFFDUTY            ; set off duty data to P0L
        MOVW    AX,TRBFPNT              ;
        INCW    AX                      ; increment transmissive buffer pointer
        MOVW    TRBFPNT,AX              ;
        NOT1    BITF                    ; reverse before/after bit flag
        DEC     BITCNT                  ; decrement bit counter
        MOV     MODE,#TRN               ; set transmissive mode
        CLR1    PRTYSAV                 ; clear parity save area
        CLR1    INPUTDAT                ; clear data input end flag
        CLR1    BUSIDLF                 ; clear bus idle flag
        CLR1    PAUSEF                  ; clear pause mode
        MOV     ERROR,#0                ; clear all error flag
        CLR1    CMK10                   ; enable INTC10
        SET1    PMK0                    ; disable INTP0
        CLR1    CIF30                   ; clear INTC30 request flag
        SET1    CMPTTRF                 ; set conpetitive transmission flag
        BR      RETP0                   ;
```

11
214
218A
234
244

```
STRT_RCV:
        BF      RCVPORT,$R_STPTM3       ; invalid revel ?
        BR      RETP0                   ;
R_STPTM3:
        MOV     A,TMC0                  ;
        BF      A.7,$R_DMYCHK           ; TM3 counting ?
        AND     TMC0,#01111111B         ; stop TM3
R_DMYCHK:
        CMP     DATCNT,#DMYD            ; dummy ?
        BNZ     $STP_RTP                ;
        SET1    DMYDTERF                ; if detect data, set dummy data error
        SET1    PMK0                    ; disable INTP0
        BR      RETP0                   ;
STP_RTP:
        MOV     RTPC,#00000000B         ; stop realtime output
        CMP     DATCNT,#PRD             ; first detection ? (PR)
        BNZ     $STRTBIT                ; THEN
        MOV     MODE,#RCV               ; set receiving mode
        CLR1    PAUSEF                  ; clear pause mode
        CLR1    BUSIDLF                 ; clear bus idle flag
        CLR1    RCVENDF                 ; clear receive end flag
        MOV     ERROR,#0                ; clear all error flag
STRTBIT:                                ; ELSE
        CLR1    PRTYSAV                 ; initialize parity data save area
        NOT1    BITF                    ; reverse before/after bit flag
        DEC     BITCNT                  ; decrement bit counter
        MOV     CR10,#TRRVTIM-10        ; set first pulse width
        CLR1    CMK10                   ; enable INTC10
        SET1    PMK0                    ; disable INTP0
        CLR1    CIF30                   ; clear INTC30 request flag
        OR      TMC1,#00001000B         ; start TM1

RETP0:
        BF      IC30F,$CHK_CMPT         ; in INTC30 routine ?
        CLR1    IC30F                   ; clear going out INTC30 routine flag
        INCW    SP                      ; SP+3
        INCW    SP                      ;
        INCW    SP                      ;
CHK_CMPT:
        BF      CMPTTRF,$R_PCCHK        ; competitive transmission ?
        CLR1    CMPTTRF                 ; clear competitive transmission flag
        BF      PCCHKF,$RETIP0          ; PC check ?
        BT      ACPTINTF,$CHK_PNT2      ; accepting short frame interrupt ?
        CMP     TRBFPNT,#LOW(TRNDAT1+1) ; transmissive data pointer 1 is next ?
        BZ      $RWT_SP                 ;
        BR      DEC_PNT                 ;
CHK_PNT2:
        CMP     TRBFPNT,#LOW(TRNDAT2+1) ; transmissive data pointer 2 is next ?
        BZ      $RWT_SP                 ;
```

```
DEC_PNT:
        MOVW    AX,TRBFPNT              ;
        DECW    AX                      ; decrement transmissive pointer
        MOVW    TRBFPNT,AX              ;
        BR      RWT_SP                  ;
R_PCCHK:
        BF      PCCHKF,$RETIP0          ; PC check ?
RWT_SP:
        MOVW    AX,SP                   ; rewrite stack pointer
        MOVW    HL,AX                   ;
        MOVW    AX,#BUS_IDL             ;
        MOVW    [HL],AX                 ;
RETIP0:
        RETI                            ;
```

11

214

218A

234

244

```
;---------------------------------------
;       *ENTER RECOVERING SYNCRONISM
;       *END RECOVERING SYNCRONISM
;       *TIMEOUT ROUTINE
;               by INTC30 interrupt
;---------------------------------------

INTC30:
        SEL     RB2                     ; select register bank 2
        SET1    IC30F                   ; go into INTC30 routine
        AND     TMC0,#01111111B         ; stop TM3
        BF      PAUSEF,$TMOUT           ; pause mode ?
        BT      SYNCF,$SYNC_END         ; recovering syncronism ?
        BF      MKDATF,$SET_TM3         ; making data end ?

;*****  enter recovering syncronism *****

        MOV     RTPC,#00001001B         ; INTP0 trigger, P0L realtime output
        SET1    EP0TRGF                 ; set INTP0 trigger enable flag
SET_TM3:
        MOV     PRM0,#00100000B         ; set TM3 prescaler fclk/16
        MOV     CR30,#SYNCTIM           ; set recovering syncronism time
        OR      TMC0,#10000000B         ; start TM3
        SET1    SYNCF                   ; set recovering syncronism flag
        BR      RETC30                  ;

;*****  end recovering syncronism *****

SYNC_END:
        SET1    BUSIDLF                 ; set bus idle
        EI                              ; enable multiple interrupt
        BF      MKDATF,$CLR_PAUS        ; making data end ?
        MOV     RTPC,#00001001B         ; INTP0 trigger, P0L realtime output
        SET1    EP0TRGF                 ; set INTP0 trigger enable flag
CLR_PAUS:
        SET1    CMK10                   ; disable INTC10
        CLR1    PAUSEF                  ; clear pausing mode
        BF      DTINTF,$P_BRMOD         ; detect short frame interrupt
        CLR1    DTINTF                  ; clear detecting short frame interrupt
P_BRMOD:
        CMP     MODE,#TRN               ;
        BZ      $P_TRNMOD               ;
        CMP     MODE,#RCV               ;
        BZ      $P_RCVMOD               ;
        BR      CLR_MOD                 ;
```

```
P_TRNMOD:
        SET1    TRNENDF                 ; set end of transmission
        BR      CLR_MOD                 ;
P_RCVMOD:
        SET1    RCVENDF                 ; set end of receiving
        BT      SYSUPF,$CLR_MOD         ; system-up mode ?
        SET1    CMDCHKF                 ; set command check flag
CLR_MOD:
        MOV     MODE,#0                 ; clear mode
        CLR1    OVDATERF                ; clear over data error flag
        CLR1    SYNCF                   ; clear recovering syncronism flag
        BR      RETC30                  ;

;*****  receiving timeout *****

TMOUT:
        SET1    TMOUTERF                ; set timeout error
        CMP     DATCNT,#ACKD_           ; ack/nak ?
        BZ      $ACKTMOUT               ;
        MOV     DATCNT,#PRD             ; initialize data counter
        SET1    PAUSEF                  ; set pause mode
        MOV     PRM0,#01110000B         ; set TM3 prescaler fclk/512
        MOV     CR30,#PAUSTIM1          ; set pause time 1
        OR      TMC0,#10000000B         ; start TM3
        BR      RETC30                  ;
ACKTMOUT:
        SET1    NOANSERF                ; set no answer error
        MOV     BITCNT,#1               ;
        SET1    PMK0                    ; disable INTP0
        CLR1    CMK10                   ; enable INTC10
        OR      TMC1,#00001000B         ; start TM1
RETC30:
        CLR1    IC30F                   ; go out INTC30 routine
        RETI                            ;
```

11
214
218A
234
244

```
;---------------------------------------
;       REQUEST OF TRANSMISSION
;               by INTP1 interrupt
;---------------------------------------

INTP1:
        CMP     RTPC,#00000001B         ; transmitting ?
        BZ      $SET_RQ2                ;
        SET1    RQTRNF                  ; set second request of transmission
        BR      RETRQTR                 ;
SET_RQ2:
        SET1    RQTRN2F                 ; set request of transmission
RETRQTR:
        RETI                            ;

;---------------------------------------
;       REQUEST OF
;       SHORT FRAME INTERRUPT ROUTINE
;               by INTP2 interrupt
;---------------------------------------

INTP2:
        BT      BUSIDLF,$SETRQ2         ; bus idle ?
        BT      PAUSEF,$SETRQ2          ; pausing ?
        SET1    RQINTF                  ; set request of short frame interrupt
        BR      RETRQINT                ;
SETRQ2:
        SET1    RQTRNF                  ; set request of tramsmission
RETRQINT:
        RETI                            ;
```

```
;-----------------------------------------
;       MAKING TRANSMISSIVE DATA ROUTINE
;       INPUT : B <- data count
;               HL <- transmit buffer address
;               DE <- data buffer address
;-----------------------------------------

;*****  define work register *****

PARITY  EQU     X.0                     ; parity bit
BIT     EQU     X.1                     ; transmissive bit
POLEF   EQU     X.2                     ; judge pole of transmission flag

MK_DATA:

;       << start bit >>

        MOV     A,#11111110B            ; set start bit data
        MOV     [DE+],A                 ;
        CLR1    PARITY                  ; parity area <-- 0
        SET1    POLEF                   ; clear pole flag
        MOV     A,[HL+]                 ; read transmissive bit
        PUSH    BC                      ;
        MOV     B,A                     ;
        MOV     C,#8                    ; initialize bit counter

;       << data >>

RD_TRBIT:
        SHR     B,1                     ;
        MOV1    BIT,CY                  ; save transmissive bit
        XOR1    CY,PARITY               ; culculate parity bit
        MOV1    PARITY,CY               ; save parity bit
        BF      BIT,$JD_POLE0           ;
        MOV     A,#11111111B            ; set off duty data to transmit buffer
        MOV     [DE+],A                 ;
        BR      BITEND                  ;
JD_POLE0:
        BT      POLEF,$POLE0            ; pole ?
        MOV     A,#11111110B            ; pole is +
        MOV     [DE+],A                 ; save transmissive bit
        BR      RVSPOLE                 ;
POLE0:
        MOV     A,#11111101B            ; pole is -
        MOV     [DE+],A                 ; save transmissive bit
RVSPOLE:
        NOT1    POLEF                   ; reverse pole flag
BITEND:
        DBNZ    C,$RD_TRBIT             ; end 8 bit ?
```

11
214
218A
234
244

```
;       << parity bit >>

        BF      PARITY,$JD_POLE1        ; parity bit ?
        MOV     A,#11111111B            ; set off duty data to transmit buffer
        MOV     [DE+],A                 ;
        BR      SETSTP                  ;
JD_POLE1:
        BT      POLEF,$POLE1            ; pole ?
        MOV     A,#11111110B            ; pole is +
        MOV     [DE+],A                 ; save transmissive bit
        BR      SETSTP                  ;
POLE1:
        MOV     A,#11111101B            ; pole is -
        MOV     [DE+],A                 ; save transmissive bit

;       << stop bit >>

SETSTP:
        MOV     A,#11111111B            ; set stop bit data
        MOV     [DE+],A                 ;
        POP     BC                      ;
        DBNZ    B,$MK_DATA              ; end making data ?
        SET1    MKDATF                  ; set end making data flag
        RET                             ;
```

```
;---------------------------------------
;       DEFINE MODE ADDRESS
;               *TRANSMISSION
;               *RECEIVING
;               *WATCHING
;---------------------------------------

;*****  communication mode  *****

MODADR:
        DW      TRNMODE                 ; transmissive mode
        DW      RCVMODE                 ; receiving mode
        DW      WCHMODE                 ; watching mode

;---------------------------------------
;       DEFINE ENTRY ADDRESS OF DATA
;               *TRANSMISSION
;               *RECEIVING
;               *WATCHING
;---------------------------------------

;*****  transmissive mode *****

T_DATADR:
        DW      T_PR                    ;
        DW      T_SA                    ;
        DW      T_DA                    ;
        DW      T_CC                    ;
        DW      T_BC                    ;
        DW      T_DATA                  ;
        DW      T_FCC                   ;
        DW      T_DMY                   ;
        DW      T_ACK                   ;

;*****  receiving mode *****

R_DATADR:
        DW      R_PR                    ;
        DW      R_SA                    ;
        DW      R_DA                    ;
        DW      R_CC                    ;
        DW      R_BC                    ;
        DW      R_DATA                  ;
        DW      R_FCC                   ;
        DW      R_DMY                   ;
        DW      R_ACK                   ;
```

11
214
218A
234
244

```
;*****  ack/nak data *****

;       << ACK data >>
ACKDAT:
        DB      11111110B               ; start bit
        DB      11111101B               ;
        DB      11111111B               ;
        DB      11111111B               ;
        DB      11111110B               ;
        DB      11111101B               ;
        DB      11111110B               ;
        DB      11111101B               ;
        DB      11111110B               ;
        DB      11111101B               ; parity bit
        DB      11111111B               ; stop bit

;       << NAK data >>
NAKDAT:
        DB      11111110B               ; start bit
        DB      11111111B               ;
        DB      11111101B               ;
        DB      11111111B               ;
        DB      11111110B               ;
        DB      11111111B               ;
        DB      11111101B               ;
        DB      11111110B               ;
        DB      11111101B               ;
        DB      11111111B               ; parity bit
        DB      11111111B               ; stop bit

;       << no answer data >>
NOANSDAT:
        DB      11111111B               ; start bit
        DB      11111111B               ;
        DB      11111111B               ;
        DB      11111111B               ;
        DB      11111111B               ;
        DB      11111111B               ;
        DB      11111111B               ;
        DB      11111111B               ;
        DB      11111111B               ;
        DB      11111111B               ; parity bit
        DB      11111111B               ; stop bit
```

```
;*****  system command table *****
;       -- except free command --

;       << for twist pare cable >>
SRSTCMD0:DB      80H            ; system reset
         DB      81H            ; request channel
         DB      82H            ; impossible to use channel
         DB      83H            ; end using channel
         DB      84H            ; possible to use channel
         DB      86H            ; possible to connect information channel
         DB      87H            ; impossible to connect information channel
         DB      90H            ; impossible to catch line
         DB      91H            ; end using line
STRGCMD0:DB     0A0H            ; start to register
RQRGCMD0:DB     0A1H            ; request to register
CFRGCMD0:DB     0A2H            ; confirm register
         DB     0A3H            ; impossible to use channel with NO.
         DB     0A4H            ; request channel with NO.
         DB     0A5H            ; end using channel with NO.
         DB     0A6H            ; announce condition
         DB     0A7H            ; release condition channel
         DB     0B0H            ; inquiry of using information channel
         DB     0B1H            ; using announce channel
         DB     0B2H            ; notice condition of telephon line
         DB     0B3H            ; connect information channel
         DB     0A4H            ; request to send
         DB     0B5H            ; possible to catch line
         DB     0B6H            ; dial
         DB     0B7H            ; dial
         DB     0C0H            ; receive
         DB     0C1H            ; answer for receiving
         DB     0C2H            ; confirm anser fo receiving
         DB     0C3H            ; end receiving
         DB     0C4H            ; request attribute
         DB     0C5H            ; answer attribute

;       << for same axis cable >>
SRSTCMD1:DB      88H            ; system reset
         DB      89H            ; request channel
         DB      8AH            ; impossible to use channel
         DB      8BH            ; end using channel
         DB      8CH            ; possible to use channel
         DB      8EH            ; possible to connect information channel
         DB      8FH            ; impossible to connect information channel
STRGCMD1:DB     0A8H            ; start to register
RQRGCMD1:DB     0A9H            ; request to register
CFRGCMD1:DB     0AAH            ; confirm register
         DB     0ABH            ; impossible to use channel with NO.
         DB     0ACH            ; request channel with NO.
         DB     0ADH            ; end using channel with NO.
         DB     0AEH            ; announce condition
```

11
214
218A
234
244

```
DB      0AFH            ; release condition channel
DB      0B9H            ; using announce channel
DB      0BBH            ; connect information channel

END
```

付録A　78K/IIシリーズのプログラムの留意事項

A.1　ベクタ割り込み処理の留意事項

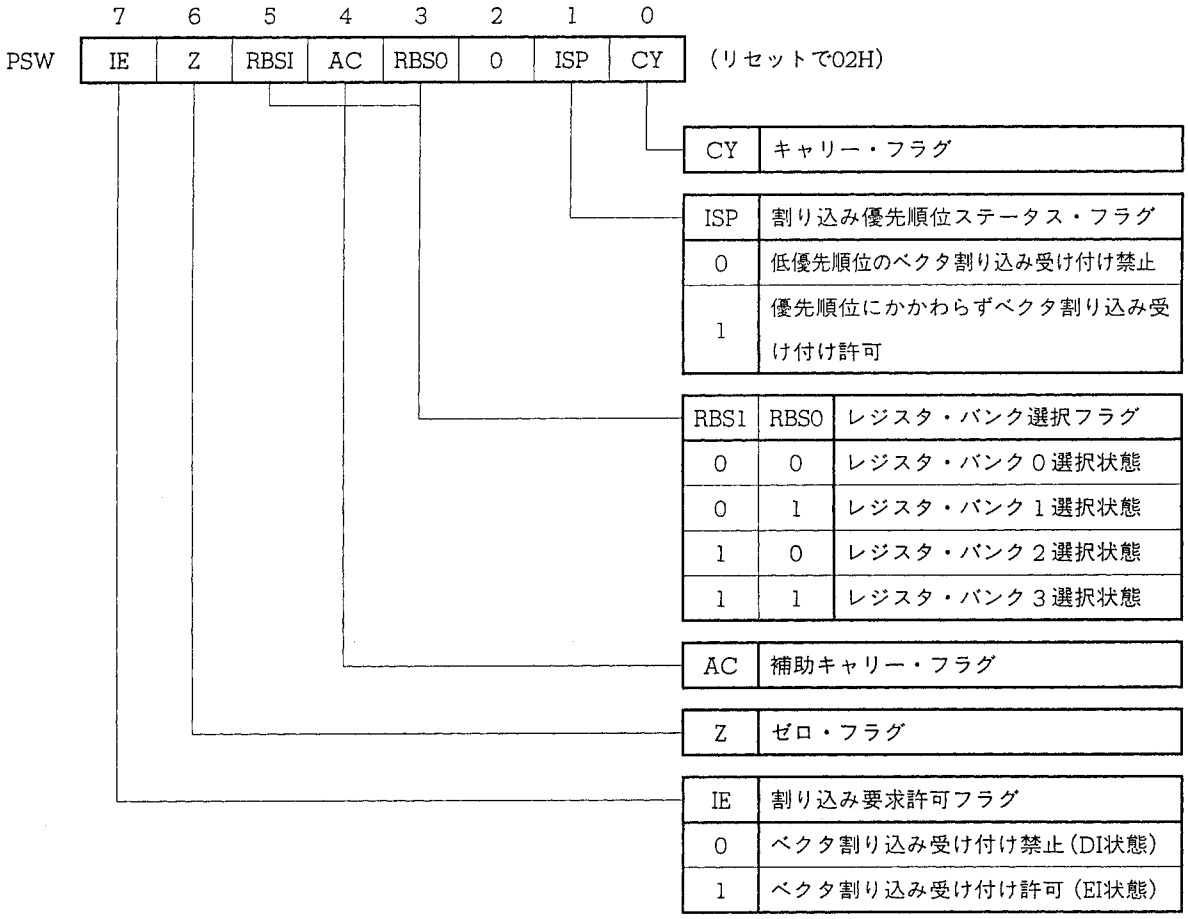
78K/IIシリーズでは図A－1に示したPSW（プログラム・ステータス・ワード）内に次の情報を含んでいます。

(a) 割り込み要求許可フラグ（IE）

(b) 割り込み優先順位ステータス・フラグ（ISP）

(c) レジスタ・バンク選択フラグ（RBS0，RBS1）

図A－1　プログラム・ステータス・ワード



付

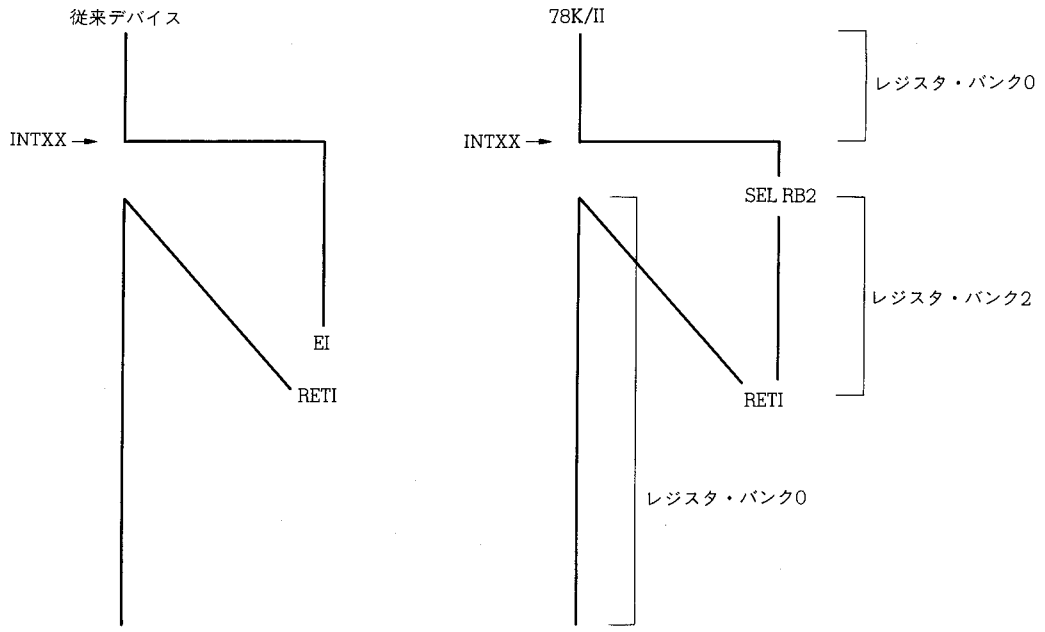
ベクタ割り込みの受け付けによりプログラム・カウンタ（PC），PSWがスタックに退避されます。このとき，割り込み許可フラグ(IE)は0となりベクタ割り込みの受け付けが禁止の状態となります。ベクタ割り込みサービス・プログラムが終了し，元のプログラムに復帰するためには“RETI”命令を実行します。

従来のデバイスの場合，次からのベクタ割り込みの受け付けを可能状態とするために，通常“EI”命令を“RETI”命令の前に実行します。

78K/IIシリーズの場合には“RETI”命令によってPC，PSWがスタックから復帰されますので，自動的に割り込みが許可状態となります。

また，ベクタ割り込み受け付け時にレジスタ・バンクを切り替えて対応した場合であっても，“RETI”命令によって元のレジスタ・バンクに戻ります。

図A－2　ベクタ割り込みとPSWの関係



次に割り込み優先順位ステータス・フラグ（ISP）は，高優先順位のベクタ割り込み（PROL/Hで指定したもの）が発生すると，PSWが退避されたのち，ISP←0となり，低優先順位のベクタ割り込みが受け付け禁止の状態となります。“RETI”命令で元のプログラムに復帰したときに優先順位にかかわらず，ベクタ割り込みの受け付けが可能となりますが，ISP←1とすることにより高優先順位のベクタ割り込み処理内でも優先順位にかかわらず（ただし“EI”命令を実行した場合：IE＝1）ベクタ割り込みの受け付けが可能です。

A.2　78K/IIシリーズの外部拡張データ・メモリのアクセス方法の留意事項

μPD78224シリーズでは，アクセスするメモリに相当するオペランドに“&”を付加することにより，あらかじめP6（ポート6）の出力ラッチに設定しておいた値がアドレス・バスの上位4ビットに出力されます。

その他の製品では，上記の方法に加えて，すべてのメモリ空間に共通のアドレシング（ダイレクト・アドレシング，レジスタ・インダイレクト・アドレシング，インデクスト・アドレシング，ベース・アドレシング）によって，PM6（ポート6モード・レジスタ）の下位4ビットに設定された値が，常にアドレス・バスの上位4ビットとして出力されます。

注意　$\overline{\text{RESET}}$解除後，PM6はOF×Hに初期化され，下位4ビットが不定となります。したがって，MMレジスタのビット6（MM6）をセット(1)して，外部拡張データ・メモリのアクセスを許可する前に，PM6レジスタの下位4ビットに“0”を設定するなどの初期化が必要です。

付

A.3　ポート0とリアルタイム出力ポートのアクセスの注意点

ポート0をリアルタイム出力ポートに設定した場合，ポート0への直接アクセスによって出力ラッチの内容を書き換えることができません。このときの出力ラッチの書き換えは，バッファ・レジスタからのハードウエアによる転送によってのみ行われます。

付録B　78K/IIシリーズ製品一覧

この章では，78K/IIシリーズ製品の機能を一覧表にして示します。

また，詳細につきましては，個別のユーザーズ・マニュアルを参照してください。

シリーズ名		μPD78214シリーズ			μPD78218Aシリーズ		μPD78224シリーズ	
項目＼品名		μPD78212	μPD78213	μPD78214 (μPD78P214)	μPD78217A	μPD78218A (μPD78P218A)	μPD78220	μPD78224 (μPD78P224)
基本命令数		65（78K/IIシリーズ共通）						
最小命令実行時間		333 ns	500 ns	333 ns	500 ns	333 ns	500 ns	333 ns
PUSH PSW命令の実行時間(クロック数)		スタック・エリアが内部デュアル・ポートRAMの場合：5または7 上記以外：7または9			スタック・エリアが内部デュアル・ポートRAMの場合：6 上記以外：8		スタック・エリアが内部デュアル・ポートRAMの場合：5または7 上記以外：7または9	
動作温度範囲と電源電圧範囲		−40〜+85°C：V_{DD}＝+5 V±10 % μPD78P218A：V_{DD}＝+5 V±6 %					−40〜+85°C： V_{DD}＝+5 V±5 % −10〜+70°C： V_{DD}＝+5 V±10 %	
汎用レジスタ		8ビット×8×4バンク						
バンク・レジスタ		P6とPM6					P6のみ	
内蔵メモリ	ROM	8 Kバイト	なし	16 Kバイト	なし	32 Kバイト	なし	16 Kバイト
	EEPROM	なし						
	RAM	384バイト	512バイト		1024バイト		640バイト	
メモリ空間		プログラム・メモリ空間：64 Kバイト，データ・メモリ空間：1 Mバイト						
I/O端子	入力	14					8	
	出力	12					12	20
	入出力	28	10	28	10	28	25	35
	合計	54	36	54	36	54	45	63
注 付加機能付き端子	プルアップ抵抗付き端子	34	16	34	16	34	なし	
	LEDダイレクト・ドライブ出力	16	0	16	0	16	8	
	トランジスタ・ダイレクト・ドライブ出力	8					なし	
P0		8ビット出力ポート						
P1		—					8ビット入出力ポート	
P2		8ビット入力ポート						
P3		8ビット入出力ポート						
P4		8ビット入出力ポート	—	8ビット入出力ポート	—	8ビット入出力ポート	—	8ビット入出力ポート
P5		8ビット入出力ポート	—	8ビット入出力ポート	—	8ビット入出力ポート	—	8ビット出力ポート
P6		4ビット出力ポート＋4ビット入出力ポート	4ビット出力ポート＋2ビット入出力ポート	4ビット出力ポート＋4ビット入出力ポート	4ビット出力ポート＋2ビット入出力ポート	4ビット出力ポート＋4ビット入出力ポート	4ビット出力ポート＋2ビット入出力ポート	4ビット出力ポート＋4ビット入出力ポート
P7		6ビット入力ポート					7ビット入出力ポート	

注　付加機能付き端子は，I/O端子の中に含まれています。

(1/3)

μPD78234シリーズ				μPD78244シリーズ	
μPD78233	μPD78234	μPD78237	μPD78238 (μPD78P238)	μPD78243	μPD78244
65（78K/IIシリーズ共通）					
500 ns	333 ns	500 ns	333 ns	500 ns	333 ns
スタック・エリアが内部デュアル・ポートRAMの場合：6 上記以外：8					
−40〜+85°C：V_{DD}＝+5 V±10 %				−10〜+70°C： V_{DD}＝+5 V±10 %	
8ビット×8×4バンク					
P6とPM6					
な　し	16 Kバイト	な　し	32 Kバイト (32 K/16 Kバイト注)	な　し	16 Kバイト
な　し				512バイト	
640バイト		1024バイト	1024バイト (1024バイト／ 640バイト注)	512バイト	
プログラム・メモリ空間：64 Kバイト，データ・メモリ空間：1 Mバイト					
16				14	
12					
18	36	18	36	10	28
46	64	46	64	36	54
24	42	24	42	16	34
8	24	8	24	0	16
8					
8ビット出力ポート					
8ビット入出力ポート				—	
8ビット入力ポート					
8ビット入出力ポート					
—	8ビット 入出力ポート	—	8ビット 入出力ポート	—	8ビット 入出力ポート
—	8ビット 入出力ポート	—	8ビット 入出力ポート	—	8ビット 入出力ポート
4ビット 出力ポート + 2ビット 入出力ポート	4ビット 出力ポート + 4ビット 入出力ポート	4ビット 出力ポート + 2ビット 入出力ポート	4ビット 出力ポート + 4ビット 入出力ポート	4ビット 出力ポート + 2ビット 入出力ポート	4ビット 出力ポート + 4ビット 入出力ポート
8ビット入力ポート				6ビット入力ポート	

注　ソフトウエアにより設定

付

シリーズ名		μPD78214シリーズ			μPD78218Aシリーズ		μPD78224シリーズ	
項目＼品名		μPD78212	μPD78213	μPD78214 (μPD78P214)	μPD78217A	μPD78217A (μPD78P218A)	μPD78220	μPD78224 (μPD78P224)
PWM出力		なし						
コンパレータ		なし					4ビット×8	
A/Dコンバータ		8ビット×8					なし	
	変換時間の選択	動作周波数に応じて選択					—	
	AV_{REF}入力電圧範囲	3.4 V～V_{DD}			3.6 V～V_{DD}		—	
	入力電圧に関する制限	ADMレジスタのANI0-ANI2ビットで選択されている端子のみ，常時0VからAV_{REF}端子電圧まで			A/D変換の対象となっている端子について，A/D変換中のみ0V からAV_{REF}端子電圧まで		—	
D/Aコンバータ		なし						
タイマ／カウンタ	16ビット・タイマ／カウンタ	1						
	8ビット・タイマ／カウンタ	3					2	
	トグル出力	4						
	PWM / PPG出力	あり					なし	
	ワンショット・パルス	なし			あり		なし	
	割り込みソース	7					5	
外部SFR領域		0FFD0H-0FFDFHの16バイト					なし	
シリアル・インタフェース	UART	1チャネル						
	CSI	1チャネル（SBI対応）						
	BRG用タイマ	あり（タイマ／カウンタ3と兼用）					あり	
	専用ボー・レート・ジェネレータ	あり					なし	
	外部ボー・レート・クロック入力	あり					なし	
リアルタイム出力ポート		4ビット×2または8ビット×1						

(2/3)

μPD78234シリーズ				μPD78244シリーズ	
μPD78233	μPD78234	μPD78237	μPD78238 (μPD78P238)	μPD78243	μPD78244
12ビット×2				なし	
なし					
8ビット×8					
任意に選択可能				動作周波数に応じて選択	
3.4 V～V_{DD}				3.6 V～V_{DD}	
A/D変換の対象となっている端子について，A/D変換中のみ， 0 VからAV$_{REF}$端子電圧まで					
8ビット×2				なし	
1					
3					
4					
あり					
あり					
7					
0FFD0H-0FFDFHの16バイト					
1チャネル					
1チャネル（SBI対応）					
あり（タイマ／カウンタ3と兼用）					
あり					
あり					
4ビット×2または8ビット×1					

付

シリーズ名	μPD78214シリーズ			μPD78218Aシリーズ		μPD78224シリーズ	
項目＼品名	μPD78212	μPD78213	μPD78214 (μPD78P214)	μPD78217A	μPD78217A (μPD78P218A)	μPD78220	μPD78224 (μPD78P224)
割り込み	2レベル（プログラマブル），ベクタ／マクロ・サービス						
外部	7					8	
内部	12					9	
マクロ・サービス使用可能割り込み数	15					6	
マクロ・サービス・カウンタのビット数	8ビットのみ			8ビット／16ビットの選択が可能 (ただし，タイプAを除く)		8ビットのみ	
マクロ・サービス・タイプCのMPD，MPTのインクリメント	下位8ビットのみインクリメント (上位は変化せず)			16ビットでインクリメント		下位8ビットのみインクリメント (上位は変化せず)	
マクロ・サービスの実行時間	μPD78214シリーズ，μPD78224シリーズは同一です。 μPD78218Aシリーズ，μPD78234シリーズおよびμPD78244シリーズは同一です。 実行時間はモードによって異なります。各製品のユーザーズ・マニュアルで比較してください。						
マクロ・サービス・タイプAのメモリからSFRへのデータ転送時の制限事項	転送データがD0H-DFHのときに発生			転送元バッファ(メモリ)のアドレスが0FED0H-0FEDFHのときに発生		転送データがD0H-DFHのときに発生	
スタンバイ機能	HALT/STOPモード						
STOPモード解除時の発振安定時間	固定			2種類の時間から選択が可能		固定	
疑似SRAMリフレッシュ機能	あり（リフレッシュ・パルス幅は，$1/f_{CLK}$）					あり (リフレッシュ・パルス幅は，$1.5/f_{CLK}$)	
メモリ・アクセスに関する制限	なし					リフレッシュ機能使用時は，FC80H-FDFFHのアクセス不可	
ROMレス・モードの設定	$\overline{EA}$端子= ロウ・レベル	—	$\overline{EA}$端子= ロウ・レベル	—	$\overline{EA}$端子= ロウ・レベル	—	$\overline{EA}$端子= ロウ・レベル
パッケージ	●64ピン・プラスチック・シュリンクDIP (750 mil) ●68ピン・プラスチックQFJ：μPD78212を除く ●64ピン・プラスチックQFP（本体14×14 mm） ●74ピン・プラスチックQFP（本体20×20 mm） ●64ピン・プラスチックQUIP：μPD78212を除く ●64ピン・セラミック窓付きシュリンクDIP ：μPD78P214のみ			●64ピン・プラスチック・シュリンクDIP (750 mil) ●64ピン・プラスチックQFP (本体14×14 mm) ●64ピン・セラミック窓付きシュリンクDIP ：μPD78P218Aのみ		●84ピン・プラスチックQFJ ●94ピン・プラスチックQFP (本体20×20 mm)	

(3/3)

μPD78234シリーズ				μPD78244シリーズ	
μPD78233	μPD78234	μPD78237	μPD78238 (μPD78P238)	μPD78243	μPD78244
2レベル（プログラマブル），ベクタ／マクロ・サービス					
7					
12				14	
15					
8ビット／16ビットの選択が可能（ただし，タイプAを除く）					
16ビットでインクリメント					
μPD78214シリーズ，μPD78224シリーズは同一です。 μPD78218Aシリーズ，μPD78234シリーズおよびμPD78244シリーズは同一です。 実行時間はモードによって異なります。各製品のユーザーズ・マニュアルで比較してください。					
転送データがD0H-DFHのときに発生				転送元バッファ（メモリ）のアドレスが0FED0H-0FEDFHのときに発生	
HALT/STOPモード					
2種類の時間から選択が可能					
あ　り（リフレッシュ・パルス幅は，$1/f_{CLK}$）					
な　し					
—	MODE端子＝ ハイ・レベル	—	MODE端子＝ ハイ・レベル （設定不可）	—	$\overline{EA}$端子＝ ロウ・レベル
●84ピン・プラスチックQFJ ●80ピン・プラスチックQFP（本体14×14 mm） ●94ピン・プラスチックQFP（本体20×20 mm） ●94ピン・セラミックWQFN：μPD78P238のみ				●64ピン・プラスチック・シュリンクDIP（750 mil） ●64ピン・プラスチックQFP（本体14×14 mm）	

付

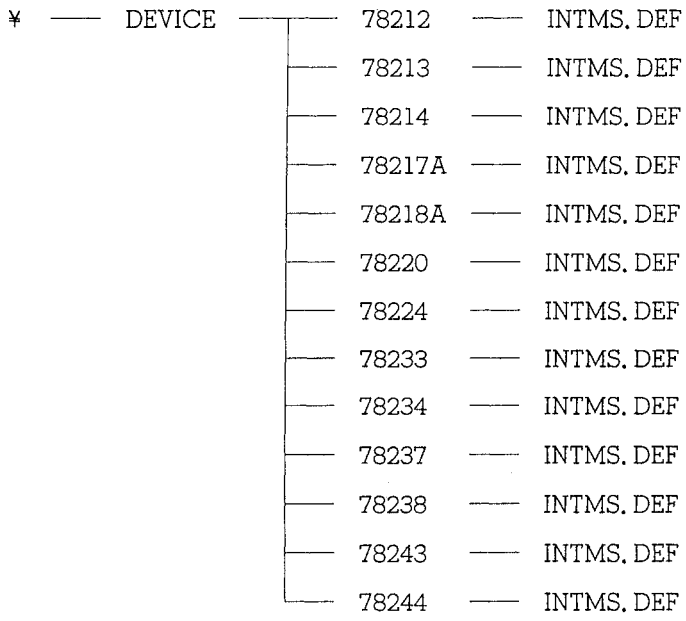
付録C　RA78K/II

C.1　RA78K/II添付ファイルについて

このアプリケーション・ノートでも使用していますが，バージョン4.00以上のアセンブラには，アセンブラ・パッケージのプログラムのほかに，アセンブル時にインクルードして使用することのできるテキスト・ファイルが添付されています。

添付ファイルは，割り込みのベクタ・テーブルやマクロ・サービスで使用する領域を確保するマクロを定義したファイル（INTMS. DEF）です。

これらのファイルは，¥DEVICEというディレクトリ中にある各デバイス用のディレクトリ内に用意してあります。



各ファイルの詳細な使用方法は，アセンブラ・パッケージに添付されている資料 **RA78K/II アセンブラ・パッケージ添付ファイルの使用方法について（SUD-T-0319-2）**を参照してください。

C.1.1　INTMS.DEF

INTMS.DEFは，割り込みおよびマクロ・サービスで使用するメモリ領域の確保などを行うマクロを定義したファイルです。マクロを参照することで領域の確保などを行います。これらを使用することにより，絶対アドレスを気にすることなしに割り込みやマクロ・サービスが使用できます。

マクロは，定義する内容により３種類に分類されます。

(1) ベクタ・テーブル定義用マクロ

ベクタ・テーブルの定義を行います。各割り込みごとにマクロが定義されています。

(2) マクロ・サービス・コントロール・ワード用領域確保マクロ

マクロ・サービス・コントロール・ワード用の領域確保とレーベルの定義を行います。各割り込み（マクロ・サービスが使用できるものに限る）ごとにマクロが定義されています。

(3) マクロ・サービス・チャネル用領域確保マクロ

マクロ・サービス・チャネル用の領域確保とレーベルの定義を行います。各マクロ・サービスのタイプごとにマクロが定義されています。

付録D　プログラム，デバイス，内蔵周辺ハードウエアの対応表

項目			対象デバイス					内蔵周辺ハードウエア												構造化アセンブラ対応
								タイマ／カウンタ				SBI	リアルタイム出力ポート	A/Dコンバータ	ポートT	割り込み				
																マクロ・サービス			ベクタ割り込み	
			μPD78214シリーズ	PD78218Aシリーズ	μPD78224シリーズ	μPD78234シリーズ	μPD78244シリーズ	16ビット・タイマ／カウンタ	8ビット・タイマ／カウンタ1	8ビット・タイマ／カウンタ2	8ビット・タイマ／カウンタ3					タイプA	タイプB	タイプC		
第2章　タイマ／カウンタのプログラム応用例	2.1　マクロ・サービスによるフリー・ランニング・インターバル・タイマ		○	○		○	○			○						○	○			
	2.2　高分解能PWM出力		○	○		○	○			○						○	△			
	2.3　高分解能キャプチャ機能		○	○		○	○		△	○						△	○			○
	2.4　積分型A/Dコンバータ		○	○	○	○	○	○											○	○
	2.5 DTMF信号の生成	2.5.2　ベクタ割り込みを使用したプログラム例	○	○		○	○			○									○	○
		2.5.3　マクロ・サービスを使用したプログラム例	○	○		○	○			○							○			○
第3章　マクロ・サービスの応用例	3.1　プログラマブル・イベント・カウンタ		○	○	△	○	○									△注3	○	○		○
	3.2　パルス周期の測定	3.2.1　16ビット・タイマ／カウンタを使用したプログラム例	○	○	△	○	○	○									○			○
		3.2.2　8ビット・タイマ／カウンタ1を使用したプログラム例	○	○	△	○	○		○	△	△					△注4	○			○
	3.3　アナログ波形出力		○	○	△	○	○	△注2	△	△注2	○注2						○			○
	3.4　階段波形出力によるA/D変換	3.4.1　ベクタ割り込みを使用したプログラム例	○	○	△	○	○	△注2	△	△注2	○注2						○		○	○
		3.4.2　マクロ・サービスを使用したプログラム例	○	○	△	○	○	△注2	△	△注2	○注2						○			○
	3.5　リアルタイム出力ポートによるPWM出力		○	○	○	○	○						○					○		○
第4章　ウォッチドッグ・タイマ			○	○	○	○	○												○	○
第5章　リアルタイム出力ポートによるステッピング・モータの制御	5.1　高分解能タイミングによる制御		○	○	○	○	○						○					○		○
	5.2　2軸同時制御		○	○	○	○	○						○					○		
第6章　リアルタイム出力ポートによるインバータの制御				○		△注1							○					○		
第7章　表示／キー入力	7.1　ベクタ割り込みによるプログラム例		○	○	○	○	○											○	○	
	7.2　マクロ・サービスによるプログラム例		○	○	△	○	○										○	○		
第8章　アナログ・キー入力	8.1　A/Dコンバータによるアナログ・キー入力		○	○		○	○							○						
	8.2　アナログ・キー入力				○										○				○	
第9章　内蔵ROMチェック			○	○	○	○	○													○
第10章　SBIの応用			○	○	○	○	○					○				○	△		○	
第11章　ホームバス・システムへの応用			△	○		△	△						○							

備考　○：そのまま使用できます。
　　△：若干の修正で使用できます。

注1.　μPD78238の場合はそのまま使用できます
2.　μPD78224シリーズの場合は使用できません
3.　INTP0端子を使用する場合のみ可能です（μPD78224シリーズを除く）
4.　INTP4端子とTM2を使用する場合のみ可能です（μPD78224シリーズを除く）

付

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］ 78K/IIシリーズ　アプリケーション・ノート　応用編

（IEA-700C（第4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 半応技本部員，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 半導体応用技術本部インフォメーションセンター
FAX：(044)548-7900

キリトリ

お問い合わせは，最寄りのNECへ

【営業関係お問い合わせ先】

半導体第一販売事業部 半導体第二販売事業部 半導体第三販売事業部	〒108-01　東京都港区芝五丁目7番1号（NEC本社ビル）	東　京	(03)3454-1111　(大代表)
中部支社 半導体第一販売部 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 名古屋	(052)222-2170 (052)222-2190
関西支社 半導体第一販売部 半導体第二販売部 半導体第三販売部	〒540　大阪市中央区城見一丁目4番24号（NEC関西ビル）	大　阪 大　阪 大　阪	(06) 945-3178 (06) 945-3200 (06) 945-3208

北海道支社	札　幌 (011)231-0161	太田支店	太　田 (0276)46-4011	富山支店	富　山 (0764)31-8461
東北支社	仙　台 (022)267-8740	宇都宮支店	宇都宮 (028)621-2281	三重支店	津　 (0592)25-7341
岩手支店	盛　岡 (019)651-4344	小山支店	小　山 (0285)24-5011	京都支社	京　都 (075)344-7824
山形支店	山　形 (0236)23-5511	長野支社	松　本 (0263)35-1662	神戸支社	神　戸 (078)333-3854
郡山支店	郡　山 (0249)23-5511	甲府支店	甲　府 (0552)24-4141	中国支社	広　島 (082)242-5504
いわき支店	いわき (0246)21-5511	埼玉支社	大　宮 (048)641-1411	鳥取支店	鳥　取 (0857)27-5311
長岡支店	長　岡 (0258)36-2155	立川支社	立　川 (0425)26-5981	岡山支店	岡　山 (086)225-4455
土浦支店	土　浦 (0298)23-6161	千葉支社	千　葉 (043)238-8116	四国支社	高　松 (0878)36-1200
水戸支店	水　戸 (029)226-1717	静岡支社	静　岡 (054)255-2211	新居浜支店	新居浜 (0897)32-5001
神奈川支社	横　浜 (045)324-5524	北陸支社	金　沢 (0762)23-1621	松山支店	松　山 (089)945-4149
群馬支店	高　崎 (0273)26-1255	福井支店	福　井 (0776)22-1866	九州支社	福　岡 (092)271-7700

【本資料に関する技術お問い合わせ先】

半導体ソリューション技術本部 マイクロコンピュータ技術部	〒210　川崎市幸区塚越三丁目484番地	川　崎	(044)548-7924
半導体販売技術本部 東日本販売技術部	〒108-01　東京都港区芝五丁目7番1号（NEC本社ビル）	東　京	(03)3798-9619
半導体販売技術本部 中部販売技術部	〒460　名古屋市中区錦一丁目17番1号（NEC中部ビル）	名古屋	(052)222-2125
半導体販売技術本部 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)