ユーザーズ・マニュアル

NEC

V53™

16ビット・マイクロプロセッサ

μPD70236
μPD70236(A)

資料番号　U10109JJ8V0UM00（第8版）
発行年月　May 1996 P

CMOSデバイスの一般的注意事項

①静電気対策（MOS全般）

注意 MOSデバイス取り扱いの際は静電気防止を心がけてください。

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

また，MOSデバイスを実装したボードについても同様の扱いをしてください。

②未使用入力の処理（CMOS特有）

注意 CMOSデバイスの入力レベルは固定してください。

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

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

③初期化以前の状態（MOS全般）

注意 電源投入時，MOSデバイスの初期状態は不定です。

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

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

M7 94.11

本版で改訂された主な箇所

箇　　所	内　　容
P.23,24	表２－４　特定状態での各端子の状態　修正
P.117	図５－２２　バス・ホールド（通常動作時）　修正
P.118	図５－２３　バス・ホールド（バス待ち動作時）　修正
P.266	図６－１０４（ｉ）ホールド状態→ＣＰＵサイクル→ホールド状態 修正
P.272	表７－３　各端子の状態　修正
P.275	表７－４　ＨＡＬＴモード時の端子状態（周辺機能非動作の場合） 修正
P.278	表７－５　ＳＴＯＰモード時の端子状態（周辺機能非動作の場合） 修正
P.447	１０．１６．２（４）レジスタとイミーディエト・データをレジスタへ について未定義コードに関する説明を追加
P.453	１０．１６．３（４）レジスタとイミーディエト・データをレジスタへ について未定義コードに関する説明を追加
P.459	１０．１６．４（４）レジスタとイミーディエト・データをレジスタへ について未定義コードに関する説明を追加

本文欄外の★印は，本版で改訂された主な箇所を示しています．

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

はじめに

対象者　このマニュアルは，μPD70236，70236(A)(別名称V53)の機能を理解し，それを用いたアプリケーション・システムを設計するユーザのエンジニアを対象とします．

目的　このマニュアルは，次の構成に示すμPD70236，70236(A)の持つハードウエア機能をユーザに理解していただくことを目的とします．

構成　このマニュアルは，大きく分けて次の内容で構成しております．

- 概説
- 端子機能
- CPU
- 内蔵周辺デバイス
- スタンバイ機能
- リセット機能
- 命令の説明

読み方　このマニュアルの読者には，電気，論理回路，マイクロコンピュータの一般知識を必要とします．なお，このマニュアルではμPD70236，70236(A)という製品名をV53の名称に統一して説明してあります．

μPD70216(別名称V50™)をすでに経験しているユーザ

→ V53は，V50を高速，高機能化した製品です．“**1.6 V50との相違点一覧**”で違っている部分を確認し，それらの説明を中心にお読みください．また，第6章では，各節に“**V50に対する変更箇所**”という項を設けて違いを説明していますので，その項をまずお読みください．

μPD70136(別名称V33™)をすでに経験しているユーザ

→ V53は，V33相当のCPUを内蔵しています．命令セットはフルコンパチブルです．

内蔵周辺デバイスに関する説明(**第2，3，6－8章**)を中心にお読みください．

命令ニモニックの意味を調べたいとき

→ **付録A　命令ニモニック・リスト**を利用してください．

ニモニックが分かっている命令の機能を調べたいとき

→ **付録B　命令索引(アルファベット順)**を利用してください．

レジスタ名が分かっていて，レジスタの詳細を確認するとき

→ **付録C　レジスタ索引(アルファベット順)**を利用してください．

一通りV53の機能を理解しようとするとき

→ 目次に従ってお読みください．

電気的特性を知りたいとき

→別冊のデータ・シートを参照してください。

各種機能の応用例を知りたいとき

→別冊のアプリケーション・ノートを参照してください。

凡　　例　データ表記の重み　：左が上位桁，右が下位桁

アクティブ・ロウの表記　：$\overline{\times\times\times}$（端子，信号名称に上線）

メモリ・マップのアドレス　：上部－上位，下部－下位

注　：本文中につけた注の説明

注　意　：気をつけて読んでいただきたい内容

備　考　：本文の補足説明

数の表記　：２進数…××××または××××B

１０進数…××××

１６進数…××××H

２のべき数を示す接頭語（アドレス空間，メモリ容量）：

K（キロ）　$2^{10} = 1024$

M（メガ）　$2^{20} = 1024^2$

信号の状態　：H…ハイ・レベル

L…ロウ・レベル

H／L…ハイ・レベルまたはロウ・レベル

Hi－Z…ハイ・インピーダンス

関連資料一覧

製品名		資料名		資料番号
V53		データ・シート		U10119J
		ユーザーズ・マニュアル		このマニュアル
		アプリケーション・ノート ハードウエア編		IEA-702
		Q&A集		U10875J
		レジスタ活用表		IEM-5581
μPD72291 （浮動小数点演算用コプロセッサ）		データ・シート		IC-7604
		ユーザーズ・マニュアル		IEM-5092
		アプリケーション・ノート インタフェース・ライブラリ編		IEA-714
インターツール	RA70116-I SP70116-I	ユーザーズ・マニュアル	言語編	EEU-861
			操作編	EEU-869
リアルタイムOS	RX136	ユーザーズ・マニュアル	基礎編	EEU-707
			テクニカル編	EEU-719
マルチタスク・ディバッガ	RX-DB116,136	ユーザーズ・マニュアル	基礎編	EEU-726
			テクニカル編	EEU-776
		アプリケーション・ノート ターゲット依存部移植の手法		EEA-609

目　次

図の目次

図番号　　タイトル，ページ

図番号	タイトル，ページ

図番号	タイトル，ページ

図番号	タイトル，ページ

表の目次

表番号　　タイトル，ページ

第1章　概　　説

μPD70236（別名称V53）は，CMOS VLSIによる『Vシリーズ™』の16ビット・ファミリのなかで，最上位機種に位置付けられる，高性能，高機能なマイクロプロセッサです．

1.1　Vシリーズと V53の位置付け

Vシリーズとは，『CMOSテクノロジ』を，全面的に採用した，NECオリジナル高性能マイクロプロセッサの一連の製品系列です．

Vシリーズ16ビット・マイクロプロセッサ・ファミリは，従来品とのソフトウエアの継承性を保ちつつ，さらに上位展開を図ったファミリです．内部データ・バス幅はすべて16ビットですが，外部データ・バス幅は8ビット固定のもの，16ビット固定のもの，8／16ビット切り替え可能なものの3種類があります．

V25™，V35™ファミリはシングルチップ・マイクロコンピュータです．また，V55PI™はV25，V35とソフトウエア上位コンパチブルで，V25，V35に対し高機能化，高性能化を図っています．

V20HL™，V30HL™は，V20™，V30™を高速化，低消費電力化した製品です．

V40™，V50は，CPUにV20，V30を使用し，さらに汎用周辺機能を内蔵した製品です．また，V40HL™，V50HL™は，CPUにV20HL，V30HLを使用し，V40，V50に対し高速化，低消費電力化を図っています．

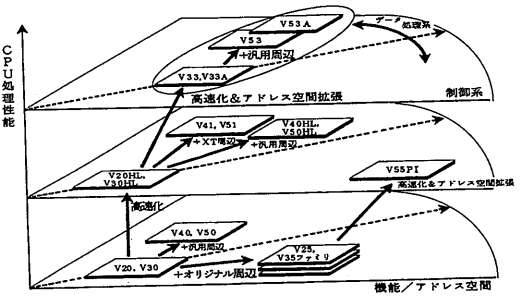
V41™，V51™は，CPUにV20HL，V30HLを使用し，IBM PC／XT™ 用の周辺機能を内蔵した製品です．

V33，V33A™は，V20，V30とソフトウエア上位コンパチブルで，性能を約4倍に高速化しています．

このような製品展開において，V53，V53Aは，高速CPU V33，V33Aを核とし，さらに汎用周辺LSIを取り込むことで，V40，V50の上位機種となっています．V53，V53Aの外部データ・バス幅は8ビット，16ビットをダイナミックに切り替えられます．

Vシリーズの製品展開を，図1－1に示します．

図1－1　16ビットVシリーズの製品展開



1.2 V53の特徴

V53は周辺機能内蔵の高速16ビット・マイクロプロセッサであり，以下の特徴を備えています．

(1) **高性能16ビットCPU搭載**

- V33相当

  動作周波数　最大16MHz

  2クロック／1バス・サイクル

- 16Mバイトのメモリ空間，64Kバイトの I／O空間
- 強力な命令セット

  V20，V30，V40，V50のネイティブ・モードとソフトウエア上位コンパチブル

- ダイナミック・バス・サイジング機能
- 未定義命令トラップ機能

  未定義コードをフェッチした場合，割り込みを発生（**5.6　割り込み機能**参照）

(2) **クロック・ジェネレータ内蔵**

- インストラクション・サイクル時間可変機能（内部動作周波数を変化させることが可能）
- CLKOUT出力（＝内部動作周波数（可変）），PCLKOUT出力（＝入力クロック周波数の1／4固定）

(3) **プログラマブル・ウエイト・コントロール・ユニット内蔵**

以下のサイクルに対して，0－7ウエイトの挿入が可能

- 外部I／Oバス・サイクル
- DMAバス・サイクル
- リフレッシュ・バス・サイクル
- 3分割した1Mバイト・メモリ空間に対するリード／ライト・サイクル
- 3分割した16Mバイト・メモリ空間に対するリード／ライト・サイクル

(4) **ダイナミックRAMリフレッシュ機能**

- 16ビット・リフレッシュ・アドレス
- 4クロック／1バス・サイクル

(5) **タイマ／カウンタ・ユニット内蔵**

- μPD71054相当（最大　10MHz）
- 3チャネル：TCTL0－TCTL2入力およびTOUT0－TOUT2出力あり

(6) **割り込みコントロール・ユニット内蔵**

- μPD71059相当（ベクタ・モードのみ）
- μPD71059をカスケード接続可能
- INTP0－INTP7入力あり

(7) **DMAコントロール・ユニット内蔵**

- μPD71071モード，μPD71037モードをソフトウエアで切り替え可能
- 24ビット・アドレス・バス
- 4チャネル

(8) シリアル・コントロール・ユニット内蔵

・μPD71051（調歩同期）相当

・専用ボー・レート・ジェネレータ内蔵

(9) バス・アービトレーション・ユニット内蔵

・V50相当

・優先順位

BUSLOCK付きCPU＞REFU（最高優先）＞DMAU＞HLDRQ＞CPU＞REFU（最低優先）

(10) コプロセッサ接続機能

μPD72291接続可能

(11) スタンバイ機能

	μPD70236	μPD70236(A)
HALTモード（一部クロック停止）	40mA（MAX.）	40mA（MAX.）
STOPモード（完全クロック停止）	3mA（MAX.）	5mA（MAX.）
インストラクション・サイクル時間可変時	13mA×fx〔MHz〕+40mA（MAX.）（fxは内部動作周波数）	15mA×fx〔MHz〕+40mA（MAX.）（fxは内部動作周波数）

(12) μPD70236(A)

・μPD70236に比べてより厳しい品質保証プログラムを適用している商品です.

（NECではこれを品質水準の分類において，特別水準と称しています.）

・μPD70236に比べて動作温度範囲が広くなっています。

1.3 オーダ情報と品質水準

(1) オーダ情報

オーダ名称	パッケージ	最大動作周波数(MHz)
μPD70236GD−10−5BB	120ピン・プラスチックQFP (□28mm)	10
μPD70236GD−12−5BB	〃	12.5
μPD70236GD−16−5BB	〃	16
μPD70236R−10	132ピン・セラミックPGA	10
μPD70236R−12	〃	12.5
μPD70236R−16	〃	16
μPD70236GD(A)−10−5BB	120ピン・プラスチックQFP (□28mm)	10
μPD70236GD(A)−12−5BB	〃	12.5

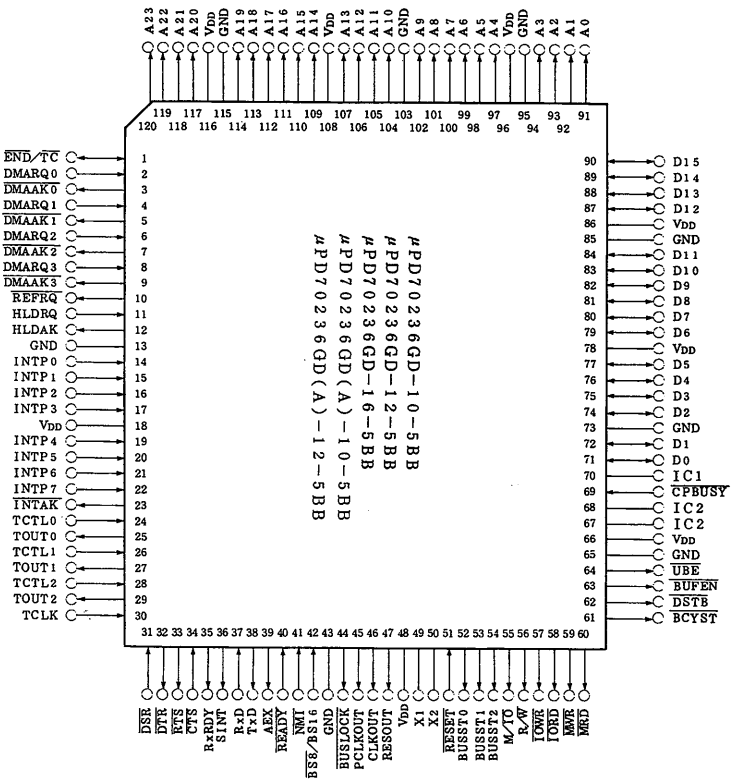
(2) 品質水準

オーダ名称	パッケージ	品質水準
μPD70236GD−10−5BB	120ピン・プラスチックQFP (□28mm)	標準水準
μPD70236GD−12−5BB	〃	〃
μPD70236GD−16−5BB	〃	〃
μPD70236R−10	132ピン・セラミックPGA	〃
μPD70236R−12	〃	〃
μPD70236R−16	〃	〃
μPD70236GD(A)−10−5BB	120ピン・プラスチックQFP (□28mm)	特別水準
μPD70236GD(A)−12−5BB	〃	〃

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。

1.4 端子接続図

(1) 120ピン・プラスチックQFP(□28mm)(Top View)

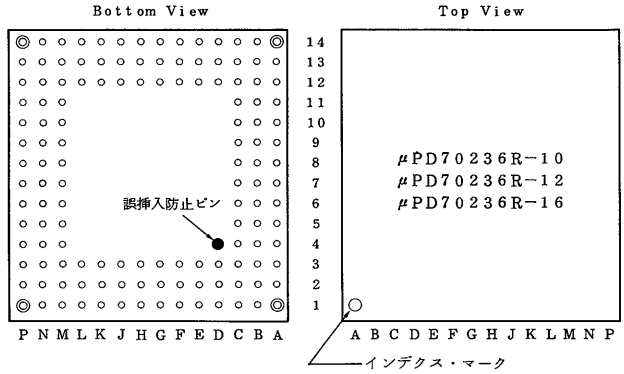


IC : Internally Connected

注意 1. IC1 : オープンにしてください.

2. IC2 : グランドに接続してください.

(2) 132ピン・セラミックPGA



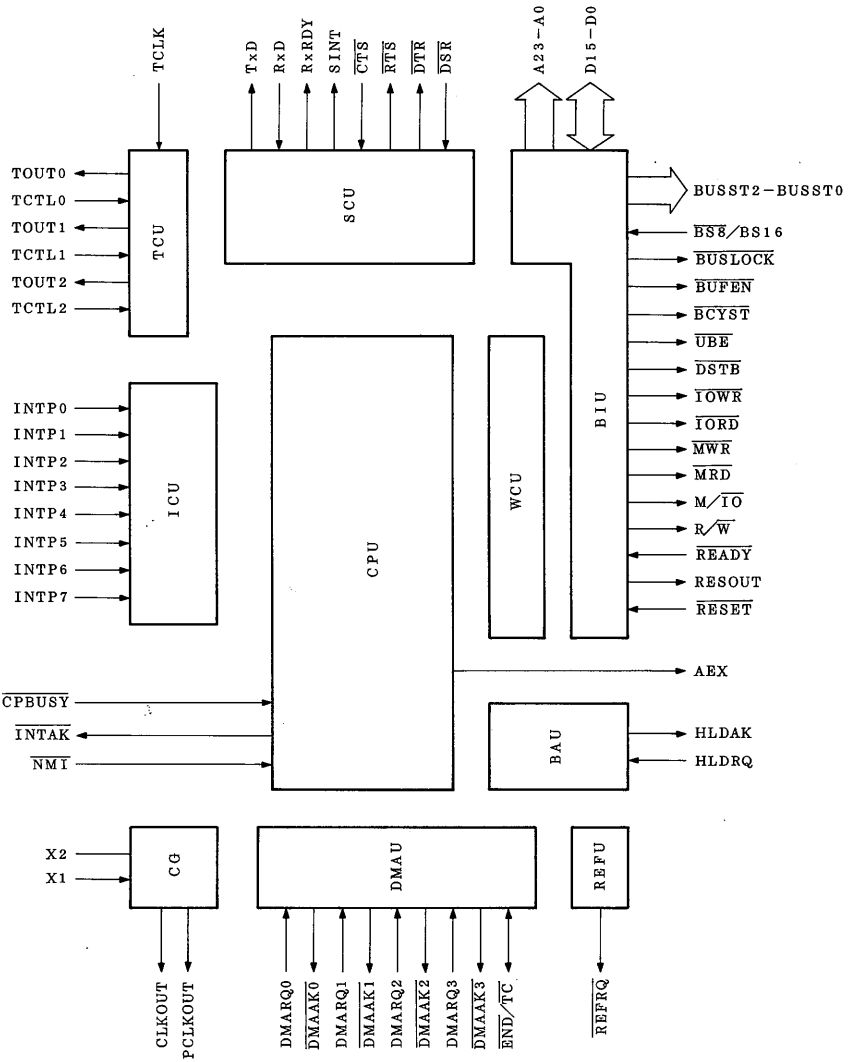
備考　誤挿入防止ピンは，ピン数に含みません．

番号	名　称	番号	名　称	番号	名　称	番号	名　称	番号	名　称	番号	名　称
A1	A22	B9	A9	D3	DMARQ0	H1	INTP2	L13	GND	N7	$\overline{\text{BUSLOCK}}$
A2	A20	B10	A5	D12	D14	H2	INTP3	L14	IC2	N8	RESOUT
A3	GND	B11	GND	D13	IC1	H3	V_{DD}	M1	TOUT0	N9	X2
A4	A19	B12	A2	D14	D11	H12	GND	M2	TCTL2	N10	BUSSTO
A5	A16	B13	IC1	E1	HLDRQ	H13	D2	M3	TCLK	N11	R/$\overline{\text{W}}$
A6	A14	B14	D12	E2	$\overline{\text{DMAAK3}}$	H14	D3	M4	$\overline{\text{DTR}}$	N12	$\overline{\text{IORD}}$
A7	A12	C1	$\overline{\text{DMAAK2}}$	E3	DMARQ2	J1	INTP4	M5	RxRDY	N13	$\overline{\text{BCYST}}$
A8	A11	C2	$\overline{\text{DMAAK0}}$	E12	V_{DD}	J2	INTP5	M6	AEX	N14	$\overline{\text{UBE}}$
A9	NC	C3	IC1	E13	D10	J3	INTP7	M7	GND	P1	$\overline{\text{DSR}}$
A10	A8	C4	A23	E14	D8	J12	IC1	M8	V_{DD}	P2	$\overline{\text{CTS}}$
A11	A6	C5	IC1	F1	NC	J13	D1	M9	BUSST1	P3	SINT
A12	A4	C6	A18	F2	HLDAK	J14	NC	M10	IC1	P4	TxD
A13	A3	C7	V_{DD}	F3	$\overline{\text{REFRQ}}$	K1	INTP6	M11	$\overline{\text{MRD}}$	P5	$\overline{\text{READY}}$
A14	A0	C8	GND	F12	D9	K2	$\overline{\text{INTAK}}$	M12	IC1	P6	$\overline{\text{BS8}}$/BS16
B1	$\overline{\text{DMARQ1}}$	C9	A7	F13	D7	K3	TCTL1	M13	$\overline{\text{BUFEN}}$	P7	PCLKOUT
B2	$\overline{\text{END}}$/$\overline{\text{TC}}$	C10	V_{DD}	F14	D6	K12	V_{DD}	M14	IC2	P8	CLKOUT
B3	A21	C11	A1	G1	INTP1	K13	$\overline{\text{CPBUSY}}$	N1	TOUT1	P9	X1
B4	V_{DD}	C12	D15	G2	INTP0	K14	D0	N2	IC1	P10	$\overline{\text{RESET}}$
B5	A17	C13	D13	G3	GND	L1	TCTL0	N3	$\overline{\text{RTS}}$	P11	BUSST2
B6	A15	C14	GND	G12	V_{DD}	L2	IC1	N4	IC1	P12	M/$\overline{\text{IO}}$
B7	A13	D1	DMARQ3	G13	D5	L3	TOUT2	N5	RxD	P13	$\overline{\text{IOWR}}$
B8	A10	D2	$\overline{\text{DMAAK1}}$	G14	D4	L12	$\overline{\text{DSTB}}$	N6	$\overline{\text{NMI}}$	P14	$\overline{\text{MWR}}$

注意 1.　IC1：オープンにしてください．

2.　IC2：グランドに接続してください．

1.5 内部ブロック図



CPU ：中央演算処理ユニット
CG ：クロック・ジェネレータ
BIU ：バス・インタフェース・ユニット
BAU ：バス・アービトレーション・ユニット
WCU ：ウエイト・コントロール・ユニット
REFU：リフレッシュ・コントロール・ユニット
TCU ：タイマ／カウンタ・ユニット
SCU ：シリアル・コントロール・ユニット
ICU ：割り込みコントロール・ユニット
DMAU：DMAコントロール・ユニット

1.6 V50との相違点一覧

V53の，V50に対する主な相違点を表1－1に示します．

表1－1 V50とV53の相違点一覧

項　　目	V50（μPD70216）	V53（μPD70236）
CPU性能（相対値）	1.0	約4.0
メモリ空間	1Mバイト	通常アドレス・モード時：1Mバイト 拡張アドレス・モード時：16Mバイト
アドレス・バス	・20ビット ・データ・バスとマルチプレクス	・24ビット ・データ・バスとセパレート
μPD8080AFエミュレーション機能	あり	なし
未定義命令コード・トラップ	しない	する（割り込みを発生）
割り込み入力	・INTP1－INTP7（7本） ・TOUT0，TOUT1，SCU端子とマルチプレクスあり	・INTP0－INTP7（8本） ・ほかの端子とマルチプレクスなし
DMAコントロール・ユニット	・μPD71071モード ・20ビット・アドレス	・μPD71071モードとμPD71037モードを切り替え可能 ・24ビット・アドレス
シリアル・コントロール・ユニット	・1チャネル ・SRDY出力 ・専用ボー・レート・ジェネレータなし ・RS－232－C制御端子なし	・1チャネル ・RxRDY，SINT出力 ・専用ボー・レート・ジェネレータあり ・RS－232－C制御端子あり
タイマ／カウンタ・ユニット	TOUT1，TOUT2，TCTL2端子あり	TOUT0，TOUT1，TOUT2，TCTL0，TCTL1，TCTL2端子あり
内部I／O再配置機能	16ビット・バウンダリに再配置可能	8ビット・バウンダリ，または16ビット・バウンダリに再配置機能を切り替え可能
バス・サイジング機能	なし	あり（8ビット↔16ビット可変）
スタンバイ機能	HALTモードのみ	HALTモード，STOPモード インストラクション・サイクル時間変更可能
リフレッシュ・アドレス	9ビット	16ビット
プログラマブル・ウエイト	0－3ウエイト	0－7ウエイト

1．7　μPD70236とμPD70236(A)の違い

項目＼品名	μPD70236	μPD70236(A)
品質水準	標準水準	特別水準
動作周囲温度（T_A）	−10〜+70°C	−40〜+85°C
動作周波数（fx）	2〜10／12.5／16MHz	2.5〜10／12.5MHz
電源電圧（V_{DD}）	5V±10%	4.5〜5.25V
DC特性	低レベル入力電圧，リーク電流が異なります．	
AC特性	バス・タイミングが異なります．	

［メ　モ］

第2章　端子機能

2.1　端子機能一覧

表2－1　端子機能一覧（1／2）

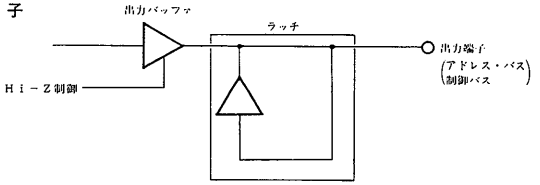
端　子　名	入　出　力	機　　能
A23－A0	3ステート出力	アドレス・バス
D15－D0	3ステート入出力	データ・バス
$\overline{\text{UBE}}$	3ステート出力	データ・バス上位バイト・イネーブル
R/$\overline{\text{W}}$	3ステート出力	リード／ライト選択
M/$\overline{\text{IO}}$	3ステート出力	メモリ／(I/O)選択
BUSST2－BUSST0	3ステート出力	バス・ステータス
$\overline{\text{BCYST}}$	3ステート出力	バス・サイクル・スタート
$\overline{\text{DSTB}}$	3ステート出力	データ・ストローブ
$\overline{\text{MRD}}$	3ステート出力	メモリ・リード
$\overline{\text{MWR}}$	3ステート出力	メモリ・ライト
$\overline{\text{IORD}}$	3ステート出力	I/Oリード
$\overline{\text{IOWR}}$	3ステート出力	I/Oライト
$\overline{\text{BUFEN}}$	3ステート出力	バッファ・イネーブル
$\overline{\text{BUSLOCK}}$	出　力	バス・ロック表示
$\overline{\text{READY}}$	入　力	バス・サイクル終了
$\overline{\text{BS8}}$/BS16	入　力	データ・バス幅指定
AEX	出　力	アドレス拡張モード表示
$\overline{\text{REFRQ}}$	出　力	リフレッシュ要求
HLDRQ	入　力	バス・ホールド要求
HLDAK	出　力	バス・ホールド許可
$\overline{\text{NMI}}$	入　力	ノンマスカブル割り込み要求
$\overline{\text{CPBUSY}}$	入　力	コプロセッサ・ビジィ
$\overline{\text{RESET}}$	入　力	リセット
RESOUT	出　力	システム・リセット出力
X2, X1	入　力	クリスタル／外部クロック
CLKOUT	出　力	システム・クロック出力
PCLKOUT	出　力	外部I/O用クロック出力
TCLK	入　力	タイマ・クロック

表２－１　端子機能一覧（２／２）

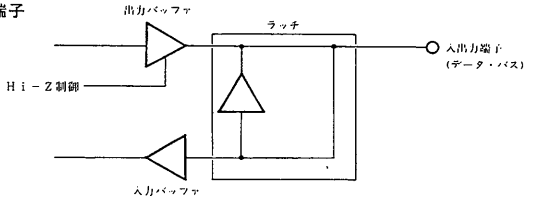
端　子　名	入 出 力	機　　能
TCTL0－TCTL2	入　力	タイマ・コントロール
TOUT0－TOUT2	出　力	タイマ出力
INTP0－INTP7	入　力	マスカブル割り込み要求
$\overline{\text{INTAK}}$	出　力	割り込みアクノリッジ
TxD	出　力	シリアル送信データ
RxD	入　力	シリアル受信データ
RxRDY	出　力	シリアル受信可
SINT	出　力	シリアル割り込み要求
$\overline{\text{RTS}}$	出　力	リクエスト・トゥ・センド
$\overline{\text{CTS}}$	入　力	クリア・トゥ・センド
$\overline{\text{DTR}}$	出　力	データ・ターミナル・レディ
$\overline{\text{DSR}}$	入　力	データ・セット・レディ
DMARQ0－DMARQ3	入　力	DMA要求
$\overline{\text{DMAAK0}}$－$\overline{\text{DMAAK3}}$	出　力	DMAアクノリッジ
$\overline{\text{END}}/\overline{\text{TC}}$	入出力	DMAサービス強制終了入力／ DMAサービス完了出力
V_{DD}	－	正電源供給端子
GND	－	グランド電位端子
IC2, IC1	－	内部接続端子

図２－１　ラッチ構成図

出力端子



入出力端子



2.2 端子機能説明

(1) A23－A0（Address Bus）……3ステート出力

外部の主記憶装置や入出力装置をアクセスする際の実アドレスを出力します．16Mバイトのメモリ空間と64Kバイト（予約領域を含む）のI／O空間をアクセスできます．

- リセット時，ホールド時，DMAカスケード接続時は，ハイ・インピーダンス．
- 割り込みアクノリッジ・サイクルでは不定．カスケード接続時はA2－A0にカスケード・アドレスを出力．
- アドレス拡張しない場合，A23－A20はロウ・レベルを出力．
- I／Oアクセス時，A23－A16はロウ・レベルを出力．
- 割り込みベクタ・テーブル・アクセス時もアドレス拡張機能は有効．

(2) D15－D0（Data Bus）……3ステート入出力

外部の主記憶装置や入出力装置をアクセスする際の書き込みデータおよび読み出しデータを入出力します．ライト・バス・サイクル以外では入力状態になっています．また，ライト・バス・サイクルでは，T1クロックの立ち上がりからライト・バス・サイクルの終了した次のクロックの立ち上がりまで出力します．

(3) $\overline{\text{UBE}}$（Upper Byte Enable）……3ステート出力

外部の主記憶や入出力装置をアクセスする際に，データ・バスの上位8ビット（D15—D8）に対するアクセスを含む場合にT1クロックの立ち下がりで変化します．データ・バスの下位8ビット（D7－D0）に対してはA0により制御します．$\overline{\text{UBE}}$，A0とバス・アクセス動作の関係を表2－2に示します．

表2－2 $\overline{\text{UBE}}$，A0とバス・アクセス動作

$\overline{\text{UBE}}$	A0	動作
0	0	16ビット・アクセス
0	1	上位8ビット・アクセス
1	0	下位8ビット・アクセス
1	1	バス・サイジング時の第2サイクル

(4) R／$\overline{\text{W}}$（Read／Write）……3ステート出力

実行中のアクセス・サイクルが，リード・サイクルであるか，あるいはライト・サイクルであるかを示します．バス・サイクルの起動されている間有効で，ハイ・レベルの場合はリード・サイクル，ロウ・レベルの場合はライト・サイクルを示します．割り込みアクノリッジ・サイクルではハイ・レベルを出力します．また，バス・ホールド時にはハイ・インピーダンス状態になります．T1クロックの立ち下がりで変化します．

(5) M／$\overline{\text{IO}}$（Memory／IO）……3ステート出力

実行中のアクセスがメモリに対するものか，その他（入出力装置，コプロセッサなど）に対す

るものかを示します．BUSST1，BUSST0信号（(6)項参照）との組み合わせにより，アクセスの種類が規定されます．バス・ホールド時はハイ・インピーダンス状態になります．T1クロックの立ち下がりで変化します．

(6) BUSST2－BUSST0（Bus Status）……3ステート出力

実行中のバス・サイクルの種類を表す情報をエンコードして出力します．バス・ホールド時はハイ・インピーダンス状態になります．M／$\overline{IO}$，R／$\overline{W}$信号との組み合わせにより，次表のような動作を示します．

表2－3　V53のバス・サイクル一覧

M／$\overline{IO}$	R／$\overline{W}$	BUSST2	BUSST1	BUSST0	バス・サイクル
0	1	0	0	0	割り込みアクノリッジ・サイクル（from SLAVE）
0	1	1	0	0	割り込みアクノリッジ・サイクル（from ICU）
0	1	0	0	1	外部I／Oリード・サイクル
0	1	1	0	1	内部I／Oリード・サイクル
0	0	0	0	1	外部I／Oライト・サイクル
0	0	1	0	1	内部I／Oライト・サイクル
0	1	0	1	0	コプロセッサ・リード・サイクル
0	0	0	1	0	コプロセッサ・ライト・サイクル
0	0	0	1	1	ホールト・アクノリッジ・サイクル
1	1	0	0	0	命令フェッチ・サイクル
1	1	1	0	0	リフレッシュ・サイクル
1	1	0	0	1	CPUメモリ・リード・サイクル
1	1	1	0	1	DMAリード転送サイクル
1	0	0	0	1	CPUメモリ・ライト・サイクル
1	0	1	0	1	DMAライト転送サイクル
1	1	0	1	0	コプロセッサ用メモリ・リード・サイクル
1	0	0	1	0	コプロセッサ用メモリ・ライト・サイクル
1	1	1	1	1	DMAカスケード

(a) 割り込みアクノリッジ・サイクル（from SLAVE）

ICUのスレーブ端子に対する2回目の割り込みアクノリッジ・サイクルにおいて出力します．このサイクルでは，外部割り込みコントローラから出力されるデータを割り込みベクタとして処理を進めます．

バス・サイジングは無効です．

プログラマブル・ウエイト，$\overline{READY}$はともに有効です．

(b) 割り込みアクノリッジ・サイクル（from ICU）

1回目の割り込みアクノリッジ・サイクル，およびICUの非スレーブ端子に対する2回

目の割り込みアクノリッジ・サイクルにおいて出力します．非スレーブ端子に対する２回目の割り込みアクノリッジ・サイクルでは，内蔵ＩＣＵから出力されるデータを割り込みベクタとして処理を進めます．

バス・サイジングは無効です．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに有効です．

(c) 外部Ｉ／Ｏリード・サイクル

ＩＮ命令の実行において，外部Ｉ／Ｏ領域をリードする場合に出力します．

バス・サイジングは有効です．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに有効です．

(d) 内部Ｉ／Ｏリード・サイクル

ＩＮ命令の実行において，内部Ｉ／Ｏ領域をリードする場合に出力します．

バス・サイジングは無効です．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに無効ですが，アドレス拡張テーブルとアドレス拡張フラグを除く内部Ｉ／Ｏ領域サイクルには自動的に２クロックのウエイト・ステートが挿入されます．

(e) 外部Ｉ／Ｏライト・サイクル

ＯＵＴ命令の実行において，外部Ｉ／Ｏ領域に対しライトする場合に出力します．

バス・サイジングは有効です．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに有効です．

(f) 内部Ｉ／Ｏライト・サイクル

ＯＵＴ命令の実行において，内部Ｉ／Ｏ領域に対しライトする場合に出力します．

バス・サイジングは無効です．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに無効ですが，アドレス拡張テーブルとアドレス拡張フラグを除く内部Ｉ／Ｏ領域サイクルには自動的に２クロックのウエイト・ステートが挿入されます．

(g) コプロセッサ・リード・サイクル

コプロセッサ命令実行時，コプロセッサに対するリード・アクセスであることを示します．

バス・サイジングは無効となり，データ・バスは常に１６ビットとなります．

プログラマブル・ウエイトは無効ですが，$\overline{\text{READY}}$は有効です．

(h) コプロセッサ・ライト・サイクル

コプロセッサ命令実行時，コプロセッサに対するライト・アクセスであることを示します．

バス・サイジングは無効となり，データ・バスは常に１６ビットとなります．

プログラマブル・ウエイトは無効ですが，$\overline{\text{READY}}$は有効です．

(i) ホールト・アクノリッジ・サイクル

ＨＡＬＴ命令実行時に出力します．このバス・サイクルでは$\overline{\text{DSTB}}$端子（(8)を参照）はアクティブとなりません．

バス・サイジングは無効です．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに無効です．

(j) 命令フェッチ・サイクル

命令のフェッチ・アクセスであることを示します．

バス・サイジングは有効です．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに有効です．

(k) リフレッシュ・サイクル

リフレッシュ・サイクルであることを示します．

バス・サイジングは無効です．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに有効です．

(l) CPUメモリ・リード・サイクル

CPUがメモリ上のデータをリードする場合に出力します．

バス・サイジングは有効です．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに有効です．

(m) DMAリード転送サイクル

DMAのリード転送（メモリ→I／O）サイクルにおいて出力します．

バス・サイジングは無効です．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに有効です．

(n) CPUメモリ・ライト・サイクル

CPUがメモリへデータをライトする場合に出力します．

バス・サイジングは有効です．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに有効です．

(o) DMAライト転送サイクル

DMAのライト転送（I／O→メモリ）サイクルにおいて出力します．

バス・サイジングは無効です．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに有効です．

(p) コプロセッサ用メモリ・リード・サイクル

コプロセッサ（μPD72291）のためにメモリのデータをリードする場合に出力します．メモリから読み出されたデータはそのままコプロセッサに引き取られます．

バス・サイジングは有効ですが，コプロセッサは常に16ビット・バスとして動作するため正常な動作ができなくなります．このため，このサイクルでは$\overline{\text{BS8}}$／BS16を常にハイ・レベルとしてください．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに有効です．

(q) コプロセッサ用メモリ・ライト・サイクル

コプロセッサ（μPD72291）のためにメモリへデータをライトする場合に出力します．CPUはデータ・バスを駆動せず，代わりにコプロセッサがデータ・バスを駆動し，メモリへ書き込みます．

バス・サイジングは有効ですが，コプロセッサは常に16ビット・バスとして動作するため正常な動作ができなくなります．このため，このサイクルでは$\overline{\text{BS8}}$／BS16を常にハイ・レベルとしてください．

プログラマブル・ウエイト，$\overline{\text{READY}}$はともに有効です．

(r) DMAカスケード

スレーブのDMAコントローラに対しバスを開放し，DMAカスケード状態にあることを示します．

(7) $\overline{\text{BCYST}}$ (Bus Cycle Start Strobe) ……3ステート出力

バス・サイクルの開始を示します．バス・サイクルに起動直後の1クロック間のみロウ・レベルを出力します．バス・ホールド時はハイ・インピーダンス状態になります．

(8) $\overline{\text{DSTB}}$ (Data Strobe) ……3ステート出力

リードおよびライト動作のストローブ信号です．HALT命令が実行されたことを示すホールト・アクノリッジ・サイクルではアクティブとなりません．バス・ホールド時はハイ・インピーダンス状態になります．

リード時とライト時とでは出力タイミングが異なります．また，内部I／Oリード／ライト・サイクル，DMAサイクルおよびリフレッシュ・サイクルのときは，アクティブとなりません．

(9) $\overline{\text{BUSLOCK}}$ (Bus Lock) ……出力

CPU自身以外のバス使用者（たとえばDMAコントローラやほかのプロセッサ）に対するバスの使用禁止を示します．割り込みアクノリッジの1回目のサイクルの開始から2回目のサイクルの開始までと，BUSLOCKプリフィクス付きの命令の実行中にアクティブとなります．この信号がアクティブであるバス・ロック期間中はCPU以外のいかなるバス要求も受け付けられません．

(10) $\overline{\text{READY}}$ (Ready) ……入力

バス・サイクルを低速なメモリやI／O制御デバイスのアクセス・タイムに合わせるために，バス・サイクルを延長するかどうかを制御します．HALT命令が実行されたことを示すホールト・アクノリッジ・サイクル，および内部I／O領域アクセス・サイクルでは，入力レベルを無視します．

なお，$\overline{\text{READY}}$入力のセットアップ／ホールド時間を守るようにしてください．守らない場合の動作は保証できません．

(11) $\overline{\text{BS8}}$／BS16 (Bus Size 8Bits／16Bits) ……入力

外部データ・バスを8ビット，16ビットいずれにするかを制御します．バス・サイクルのT2ないし最後のTWクロックの立ち上がりでサンプリングします．ハイ・レベルの場合は，データ・バスの16ビットが有効なデータ・バスとして，ロウ・レベルの場合はデータ・バスの下位8ビットが有効なデータ・バスとして動作します．

例外として，ホールト・アクノリッジ・サイクル，割り込みアクノリッジ・サイクル，コプロセッサ・アクセス・サイクル，DMAサイクル，リフレッシュ・サイクルでは8ビットに設定しても無視され，16ビット・データ・バスとして動作します．

また，コプロセッサ用データ・アクセス・サイクルで8ビット・バス幅の指定を行った場合の動作は保証できません．

(12) AEX (Address Extension) ……出力

実アドレス生成を標準モードで行うか，拡張モードで行うかを示します．この端子はXAフラグの内容を出力します．ハイ・レベルの場合は24ビットの実アドレスが生成され，ロウ・レベルの場合は，20ビットの実アドレスが生成されます．

この端子は，BRKXA命令による最初のフェッチ・サイクルからアクティブになり，RETXA命令による最初のフェッチ・サイクルからインアクティブになります．

(13) HLDRQ (Hold Request) ……入力

外部デバイスがV53に対し，アドレス・バス，データ・バス，制御バスの開放を要求する信号で，CLKOUTに対して非同期入力が可能です．この端子がハイ・レベルになると，CPUは実行中のバス・サイクルがあるとその終了後に，なければすぐにアドレス・バス，データ・バス，制御バスをハイ・インピーダンス状態にし，HLDAK端子をアクティブにしてバスを開放します．

ただし，$\overline{\text{BUSLOCK}}$端子がロウ・レベルを出力している場合には，入力を受け付けません．

(14) HLDAK (Hold Acknowledge) ……出力

V53がHLDRQ信号を受け付けて，アドレス／データ・バス，制御バスをハイ・インピーダンスにして外部デバイスにバスを開放したことを示すアクノリッジ信号です．

(15) $\overline{\text{NMI}}$ (Non-maskable Interrupt Request) ……入力

マスク不可能な割り込み要求信号です．PSW内の割り込み許可フラグの状態とは無関係に常に受け付けられます．

この入力は，各クロック・サイクルの立ち下がりでサンプリングされます．この値のハイ・レベルからロウ・レベルへのエッジを検出することで，$\overline{\text{NMI}}$が受け付けられます．$\overline{\text{NMI}}$が受け付けられると，実行中の命令終了後割り込み処理に入ります．この処理が開始されると，RETI命令で終了するまではNMI処理のネスティングは行われません．NMI処理中の$\overline{\text{NMI}}$要求は保留され，その処理が終了してから，保留されていたNMIの処理を開始します．ただし，NMI処理中のスタンバイ状態は，$\overline{\text{NMI}}$入力で解除できます．その場合，NMI割り込みを発生し，NMI処理ルーチンに入ります．

この信号はHALTモード／STOPモードの解除にも使用できます．

(16) $\overline{\text{CPBUSY}}$ (Co-processor Busy) ……入力

この端子はコプロセッサの状態を示す信号を入力します．この端子のレベルがロウ・レベルの場合，コプロセッサがビジィ状態であると認識します．コプロセッサを接続しない場合は，GNDに接続してください．

(17) $\overline{\text{RESET}}$ (Reset) ……入力

プロセッサを初期化する信号で，CLKOUTに対して非同期入力が可能です．ロウ・レベルを6クロック以上保ち，ハイレベルに戻すと，プロセッサがリセットされます．

ただし，振動子による発振の場合STOPモードの解除の際の$\overline{\text{RESET}}$入力または，電源投入直後の$\overline{\text{RESET}}$入力は発振安定時間の期間，アクティブにしておく必要があります．

(18) RESOUT (Reset Output) ……出力

非同期の$\overline{\text{RESET}}$入力信号を内部クロックに同期化したアクティブ・ハイの信号として出力します．この信号はシステム・リセットとして使用できます．

(19) $\overline{\text{REFRQ}}$ (Refresh Request) ……出力

この信号は，リフレッシュ・サイクルのT2，TW，T3ステートでアクティブとなり，リフレッシュ・サイクルであることを外部に知らせます．

(20) $\overline{\text{MRD}}$ (Memory Read) ……3ステート出力

メモリからの読み出しサイクルにおいてアクティブとなるリード信号です．

この信号はCPUによるメモリ・リード・サイクルのほか，DMAのリード転送においても

出力します．

リフレッシュ・サイクルではハイ・レベルを出力します．

(21) $\overline{\text{MWR}}$ (Memory Write) ……3ステート出力

メモリへの書き込みサイクルにおいてアクティブとなるライト信号です．

この信号はCPUによるメモリ・ライト・サイクルのほか，DMAのライト転送にも出力しますが，DMAサイクルでの$\overline{\text{MWR}}$の出力タイミングには，拡張ライトと通常ライトの2種類あります．

(22) $\overline{\text{IORD}}$ (I／O Read) ……3ステート出力

I／Oからの読み出しサイクルにおいてアクティブとなるリード信号です．

この信号はCPUによるI／Oリード・サイクルのほか，DMAのライト転送においても出力します．

ただし，CPUによる内部I／Oからのリード・サイクルでは出力しません．

(23) $\overline{\text{IOWR}}$ (I／O Write) ……3ステート出力

I／Oへの書き込みサイクルにおいてアクティブとなるライト信号です．

この信号はCPUによるI／Oライト・サイクルのほか，DMAのリード転送時にも出力しますが，DMAサイクルでの$\overline{\text{IOWR}}$の出力タイミングには，拡張ライトと通常ライトの2種類あります．

なお，CPUによる内部I／Oライト・サイクルでは出力しません．

(24) $\overline{\text{BUFEN}}$ (Buffer Enable) ……3ステート出力

この信号は外部バッファの出力イネーブル信号として使用される信号です．リード・サイクルと割り込みアクノリッジ・サイクル，ライト・サイクルおよびコプロセッサ・サイクルでアクティブとなります．

ただし，DMAサイクル時，内部I／Oへのアクセス時はアクティブにはなりません．

(25) X1，X2 (Crystal) ……入力

内部クロック・ジェネレータを使用する場合は，X1，X2端子に動作周波数の2倍のクリスタルを接続します．

外部クロック・ジェネレータを使用する場合は，X1端子に動作周波数の2倍の方形波を入力し，X2端子にはX1の逆相（インバータ出力）を入力します．

(26) CLKOUT (Clock Output) ……出力

X1，X2に加えられたクロック周波数を分周した方形波クロックを出力します．このクロックのデューティは約50％です．出力周波数はCPU内部の動作周波数（発振周波数の 1／2，1／4，1／8，1／16）と同じです．

(27) PCLKOUT (Peripheral Clock Output) ……出力

X1，X2に加えられたクロック周波数を4分周した方形波クロックを出力します．このクロックのデューティは約50％です．

(28) TCLK (Timer Clock) ……入力

TCUの外部クロック入力端子です．TCUへの入力クロックは，この端子からの外部クロックを使用するか，内部クロックを分周したクロックを使用するかをイニシャライズ時に選択します．

(29) TOUT0－TOUT2（Timer Output）……出力

TCUが有する3つのカウンタの出力です．TCUで設定する6つのモードにより出力状態が異なります．

(30) TCTL0－TCTL2（Timer Control）……入力

TCUが有する3つのカウンタの制御入力です．TCUで設定する6つのモードにより制御機能が異なります．

(31) INTP0－INTP7（Interrupt from Peripherals）……入力

内蔵割り込みコントロール・ユニット（ICU）の非同期割り込み要求入力です．入力信号は，エッジ・トリガ（立ち上がり）／レベル・トリガ（ハイ・レベル）の選択ができます．受け付け優先順位は固定／回転を選択できます．

また，これらの割り込み要求入力は，HALT／STOPモードの解除にも使用できます．

(32) $\overline{\text{INTAK}}$（Interrupt Acknowledge）……出力

マスカブル割り込みに対するアクティブ・ロウのアクノリッジ信号です．

(33) TxD（Transmit Data）……出力

シリアル送信データの出力です．送信データがないときにハイ・レベル（マーキング）となります．送信データがセットされると，スタート・ビットを自動的に出力し，続いてセットされたデータをシリアルに出力します．各データの最後にはパリティ・ビットとストップ・ビットが付加されます．

(34) RxD（Receive Data）……入力

シリアル受信データの入力です．受信データがないときにハイ・レベル（マーキング）が入力され，スタート・ビットを検出すると，シリアル・データの受信を開始します．

(35) RxRDY（Receive Ready）……出力

SCUが1キャラクタ分のデータを受信し，そのデータが受信データ・バッファに転送されたとき，すなわち受信データの読み出しが可能になったときにハイ・レベルを出力します．

(36) $\overline{\text{RTS}}$（Request To Send）……出力

汎用の出力端子です．この端子の状態は，シリアル・コマンド・レジスタ（SCM）のビット5により設定できます．

また，外部へのデータ送信要求としても使用できます．

(37) $\overline{\text{CTS}}$（Clear To Send）……入力

シリアル送信制御用入力です．シリアル・コマンド・レジスタ（SCM）のビット0（TE）が“1”となっているときにロウ・レベルにすると，SCUは送信可能となります．送信中にハイ・レベルにすると，そのとき書き込まれているデータをすべて送出してから送信動作を停止し，TxD端子はハイ・レベルになります．

(38) $\overline{\text{DTR}}$（Data Terminal Ready）……出力

汎用の出力端子です．この端子の状態は，シリアル・コマンド・レジスタ（SCM）のビット1により設定できます．

また，SCUがレディ状態であることを外部に対して示す信号としても使用できます．

(39) $\overline{\text{DSR}}$（Data Set Ready）……入力

汎用の入力端子です．この端子の状態は，シリアル・ステータス・レジスタ（SST）のビッ

ト７の読み出しによって知ることができます．

また，ＳＣＵに対しすでにデータの準備ができていることを伝える信号としても使用できます．

(40) ＳＩＮＴ（Ｓｅｒｉａｌ Ｉｎｔｅｒｒｕｐｔ）……出力

送信バッファが空で，かつ送信割り込みが非マスクのとき，または，受信バッファに読み出すべきデータがあり，かつ受信割り込みが非マスクのときにＳＣＵからの割り込み要求としてアクティブになります．

この端子は，受信割り込みマスク・レジスタでマスクすることにより，送信許可信号（ＴｘＲＤＹ）としても使用できます．

(41) ＤＭＡＲＱ０－ＤＭＡＲＱ３（ＤＭＡ Ｒｅｑｕｅｓｔ）……入力

内蔵ＤＭＡＵのチャネル０－チャネル３のアクティブ・ハイのＤＭＡリクエスト信号で，ＣＬＫＯＵＴに対して非同期入力が可能です．

ＤＭＡＲＱは，対応する$\overline{\text{DMAAK}}$がアクティブになるまでハイ・レベルを保持しなければなりません．また，ＤＭＡＲＱの取り下げは，対応する$\overline{\text{DMAAK}}$がアクティブの期間に行う必要があります．ただし，確実にこのＤＭＡサイクル中にＤＭＡ要求を取り下げるためには，Ｔ３ステートのＣＬＫＯＵＴの立ち上がりでＤＭＡＲＱをサンプリングするので，それまでにＤＭＡＲＱをインアクティブにしてください．

(42) $\overline{\text{DMAAK0}}$－$\overline{\text{DMAAK3}}$（ＤＭＡ Ａｃｋｎｏｗｌｅｄｇｅ）……出力

内蔵ＤＭＡＵのチャネル０－チャネル３のアクティブ・ロウのＤＭＡアクノリッジ信号です．

(43) $\overline{\text{END}}$／$\overline{\text{TC}}$（ＥＮＤ／Ｔｅｒｍｉｎａｌ Ｃｏｕｎｔ）……入出力

内蔵ＤＭＡＵのＤＭＡ転送終了に関するアクティブ・ロウの入出力端子です．

- $\overline{\text{END}}$入力… ＤＭＡ転送中にロウ・レベルのパルスが入力されると，ＤＭＡサービス中であっても転送中のバス・サイクル終了後にサービスを終了します．
- $\overline{\text{TC}}$出力…… 転送中のチャネルのカウント・レジスタが“０”になり，指定した転送回数を終了するとロウ・レベルのパルスを出力します．

なお，この端子はオープンドレーン出力ですので，外部にプルアップ抵抗を接続してください．

(44) V_{DD}（Ｐｏｗｅｒ）

５Ｖ±１０％の正電源端子です．

V_{DD}端子はすべて共通の電源に接続してください．

(45) ＧＮＤ（Ｇｒｏｕｎｄ）

グランド電位（０Ｖ）端子です．

ＧＮＤ端子はすべて共通のグランドに接続してください．

注意　本製品はＣＭＯＳプロセスにより製造されています．したがって，入力機能のある端子は，入力レベルが中間電位にならないように（たとえばリセット直後など），必ずハイ・レベルかロウ・レベルに確定するようにしてください．

2.3 特定状態での端子状態

特定状態（バス・ホールド時，スタンバイ・モード時，リセット時，DMAカスケード時）での各端子の状態を次に示します．

表2－4 特定状態での各端子の状態（1／2）

端子名	入出力	バス・ラッチ注1	特定状態				CLKOUTに対する非同期入力
			バス・ホールド時	スタンバイ・モード時	リセット時	DMAカスケード時	
A23－A0	3ステート出力	○	Hi－Z	L	Hi－Z	Hi－Z	
D15－D0	3ステート入出力	○	Hi－Z	注2	Hi－Z	Hi－Z	
$\overline{UBE}$	3ステート出力	○	Hi－Z	H	Hi－Z	Hi－Z	
R／$\overline{W}$	3ステート出力	○	Hi－Z	L	Hi－Z	H	
M／$\overline{IO}$	3ステート出力	○	Hi－Z	L	Hi－Z	H	
BUSST2－BUSST0	3ステート出力	○	Hi－Z	注3	Hi－Z	H	★
$\overline{BCYST}$	3ステート出力	○	Hi－Z	注4	Hi－Z	Hi－Z	★
$\overline{DSTB}$	3ステート出力	○	Hi－Z	H	Hi－Z	Hi－Z	
$\overline{MRD}$	3ステート出力	○	Hi－Z	H	Hi－Z	Hi－Z	
$\overline{MWR}$	3ステート出力	○	Hi－Z	H	Hi－Z	Hi－Z	
$\overline{IORD}$	3ステート出力	○	Hi－Z	H	Hi－Z	Hi－Z	
$\overline{IOWR}$	3ステート出力	○	Hi－Z	H	Hi－Z	Hi－Z	
$\overline{BUFEN}$	3ステート出力	○	Hi－Z	H	Hi－Z	Hi－Z	
$\overline{BUSLOCK}$	出力	×	注5	注5	H	H	
$\overline{READY}$	入力	×	－	－	－	－	不可
$\overline{BS8}$／BS16	入力	×	－	－	－	－	不可
AEX	出力	×	注6	注6	H／L	注6	
$\overline{REFRQ}$	出力	×	H	注7	H	H	
HLDRQ	入力	×	－	－	－	－	可
HLDAK	出力	×	H	H／L	L	L	
$\overline{NMI}$	入力	×	－	－	－	－	可
$\overline{CPBUSY}$	入力	×	－	－	－	－	可
$\overline{RESET}$	入力	×	－	－	－	－	可
RESOUT	出力	×	L	L	H	L	
X2，X1	入力	×	－	－	－	－	－
CLKOUT	出力	×	非固定	注8	非固定	非固定	
PCLKOUT	出力	×	非固定	注8	非固定	非固定	
TCLK	入力	×	－	－	－	－	可
TCTL0－TCTL2	入力	×	－	－	－	－	可
TOUT0－TOUT2	出力	×	非固定	非固定	非固定	非固定	

（注の説明は次ページにあります）

表２－４　特定状態での各端子の状態（２／２）

端子名	入出力	バス・ラッチ注1	特定状態				CLKOUTに対する非同期入力
			バス・ホールド時	スタンバイ・モード時	リセット時	DMAカスケード時	
INTP0－INTP7	入力	×	－	－	－	－	可
$\overline{\text{INTAK}}$	出力	×	H	H	H	H	
TxD	出力	×	非固定	非固定	H	非固定	
RxD	入力	×	－	－	－	－	可
RxRDY	出力	×	非固定	非固定	H	非固定	
SINT	出力	×	非固定	非固定	L	非固定	
$\overline{\text{RTS}}$	出力	×	非固定	非固定	H	非固定	
$\overline{\text{CTS}}$	入力	×	－	－	－	－	可
$\overline{\text{DTR}}$	出力	×	非固定	非固定	H	非固定	
$\overline{\text{DSR}}$	入力	×	－	－	－	－	可
DMARQ0－DMARQ3	入力	×	－	－	－	－	可
$\overline{\text{DMAAK0}}$－$\overline{\text{DMAAK3}}$	出力	×	H	非固定	H	非固定	
$\overline{\text{END}}$／$\overline{\text{TC}}$	入出力	×	Hi－Z	非固定	Hi－Z	非固定	不可
V_{DD}	－	×	－	－	－	－	
GND	－	×	－	－	－	－	
IC2，IC1	－	×	－	－	－	－	

注1. ○：ラッチを内蔵しています．

×：ラッチを内蔵していません．

図２－１　ラッチ構成図を参照してください．

ラッチを内蔵している端子は，Hi－Z時，外部からドライブされるまで，Hi－Zとなる以前の状態を保持しています．したがって，これらの端子をプルアップまたはプルダウンする必要はありません．

また，Hi－Z状態で外部から端子レベルを反転させるには，ラッチ反転電流（I_{ILH}，I_{ILL}）以上のドライブ能力が必要です．

なお，電源投入直後のラッチ・レベルは不定です。したがって，ラッチを内蔵している端子に電源投入直後からあるレベルを期待する場合は，必要に応じてプルアップまたはプルダウンしてください．

2. ホールト・アクノリッジ・サイクルの最初の２クロック間は不定，以後はHi－Z．

3. BUSST2　：L

BUSST1，0：H

4. ホールト・アクノリッジ・サイクルの最初の１クロック間はL，以後はH．

5. 次のいずれかの場合はL，それ以外はH.
   - ホールド時にBUSLOCKプリフィクス付きの命令を実行.
   - BUSLOCKプリフィクス付きのHALT命令を実行.
6. アドレス拡張モード時はH，非拡張モード時はL.
7. HALTモード時は非固定，STOPモード時はH.
8. HALTモード時は非固定，STOPモード時はL.

備考1. H：ハイ・レベル
L：ロウ・レベル
H／L：ハイまたはロウ・レベル
Hi-Z：ハイ・インピーダンス
非固定：有効値を出力

2. スタンバイ・モード時とは，HALTモード時／STOPモード時の両方を指します（第7章　スタンバイ機能参照）

2.4 各端子の入出力回路と未使用時の処理

各端子（IC2, IC1, GND, V_{DD} は除く）の入出力回路タイプと未使用時の処理方法を表2－5に示します.

また，各入出力回路タイプの回路図を図2－2に示します.

表2－5 各端子の入出力回路タイプと未使用時の処理（1／2）

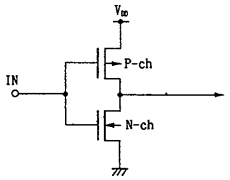
端子名称	入出力	入出力回路タイプ	未使用時の推奨接続方法
A23−A0	3ステート出力	4L	オープン
D15−D0	3ステート入出力	5L	オープン
$\overline{UBE}$	3ステート出力	4L	オープン
R/$\overline{W}$	3ステート出力	4L	オープン
M/$\overline{IO}$	3ステート出力	4L	オープン
BUSST2−BUSST0	3ステート出力	4L	オープン
$\overline{BCYST}$	3ステート出力	4L	オープン
$\overline{DSTB}$	3ステート出力	4L	オープン
$\overline{BUSLOCK}$	出力	3	オープン
$\overline{READY}$	入力	1	プルダウン抵抗を介してGNDに接続
$\overline{BS8}$/BS16	入力	1	プルアップ抵抗を介してV_{DD}に接続またはプルダウン抵抗を介してGNDに接続
AEX	出力	3	オープン
HLDRQ	入力	1	プルダウン抵抗を介してGNDに接続
HLDAK	出力	3	オープン
$\overline{NMI}$	入力	1	プルアップ抵抗を介してV_{DD}に接続
$\overline{CPBUSY}$	入力	1	プルダウン抵抗を介してGNDに接続
$\overline{RESET}$	入力	2	−
RESOUT	出力	3	オープン
$\overline{REFRQ}$	出力	3	オープン
$\overline{MRD}$	3ステート出力	4L	オープン
$\overline{MWR}$	3ステート出力	4L	オープン
$\overline{IORD}$	3ステート出力	4L	オープン
$\overline{IOWR}$	3ステート出力	4L	オープン
$\overline{BUFEN}$	3ステート出力	4L	オープン

表２－５　各端子の入出力回路タイプと未使用時の処理（２／２）

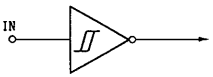
端子名称	入出力	入出力回路タイプ	未使用時の推奨接続方法
X1, X2	入　力	－	－
CLKOUT	出　力	3	オープン
PCLKOUT	出　力	3	オープン
TCLK	入　力	1	プルアップ抵抗を介してV_{DD}に接続またはプルダウン抵抗を介してＧＮＤに接続
TOUT0－TOUT2	出　力	3	オープン
TCTL0－TCTL2	入　力	1	プルアップ抵抗を介してV_{DD}に接続またはプルダウン抵抗を介してＧＮＤに接続
INTP0－INTP7	入　力	1	プルダウン抵抗を介してＧＮＤに接続
$\overline{\text{INTAK}}$	出　力	3	オープン
TxD	出　力	3	オープン
RxD	入　力	1	プルダウン抵抗を介してＧＮＤに接続
RxRDY	出　力	3	オープン
$\overline{\text{RTS}}$	出　力	3	オープン
$\overline{\text{CTS}}$	入　力	1	プルアップ抵抗を介してV_{DD}に接続またはプルダウン抵抗を介してＧＮＤに接続
$\overline{\text{DTR}}$	出　力	3	オープン
$\overline{\text{DSR}}$	入　力	1	プルアップ抵抗を介してV_{DD}に接続またはプルダウン抵抗を介してＧＮＤに接続
SINT	出　力	3	オープン
DMARQ0－DMARQ3	入　力	1	プルダウン抵抗を介してＧＮＤに接続
$\overline{\text{DMAAK0}}$－$\overline{\text{DMAAK3}}$	出　力	1	オープン
$\overline{\text{END}}$／$\overline{\text{TC}}$	入出力	１３－Ｃ	個別にプルアップ抵抗を介してV_{DD}に接続

図２－２　タイプ別入出力回路図（１／２）

タイプ1

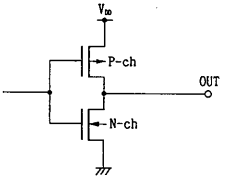


タイプ2

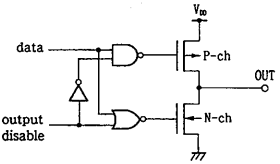


ヒステリシス特性を有するシュミット・トリガ入力となっています。

タイプ3



タイプ4



出力ハイ・インピーダンス（P-ch, N-chともにオフ）とすることができるプッシュプル出力となっています。

タイプ4L

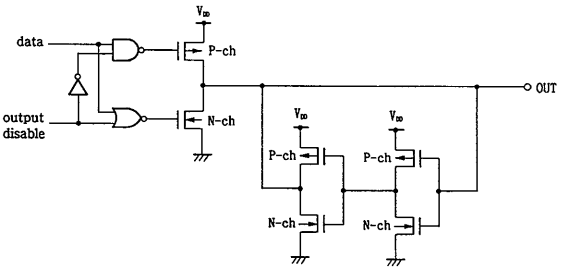
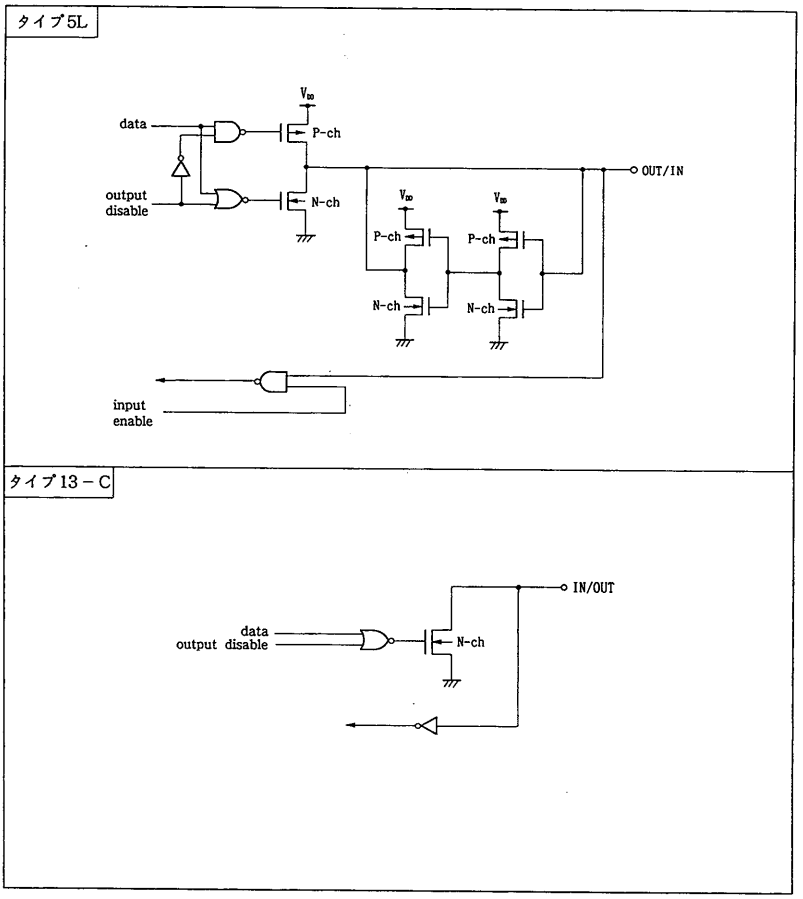


図２－２　タイプ別入出力回路図（２／２）



［メ　　モ］

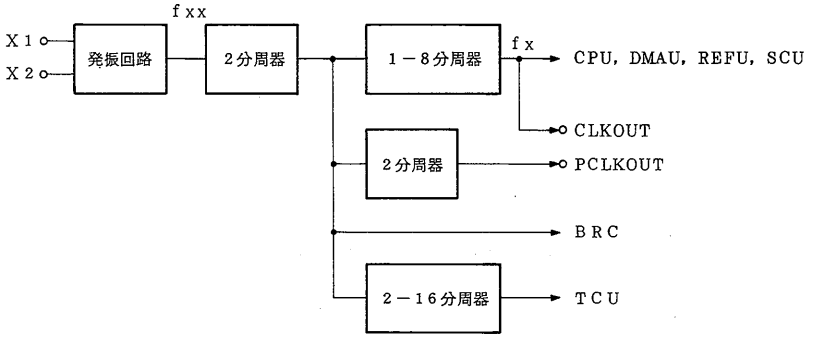
第3章　内部ブロック構成

3.1　CPU

μPD70136（V33）と同等の機能を有し，V20，V30，V40，V50のネイティブ・モードと上位コンパチブルな命令セットを有しています．また，アドレス空間を16Mバイトに拡張できます（詳細は第4章参照）．

3.2　CG(クロック・ジェネレータ)

X1，X2端子に接続されたクリスタルおよび発振器の1／2，1／4，1／8，1／16の周波数のクロックを発生し，CPU動作クロックとして供給するとともに，CLKOUT端子として外部へ出力します．また，発振周波数の1／4固定周波数のクロックをPCLKOUT端子に出力します（詳細は6.9節参照）．



3.3　BIU(バス・インタフェース・ユニット)

アドレス・バス，データ・バス，制御バスの端子を制御します．これらのバスはCPU，DMAU，REFUの3つによって使用されます．

3.4　BAU(バス・アービトレーション・ユニット)

V53内部のバス使用権の調停を行います．バス使用権の優先順位は次のとおりです（詳細は6.8節参照）．

BUSLOCK付きCPU＞最高優先のREFU＞DMAU＞HLDRQ＞通常のCPU＞最低優先のREFU

3.5 WCU(ウエイト・コントロール・ユニット)

メモリ・サイクル，I／Oサイクル，DMAサイクル，リフレッシュ・サイクルに対し，0－7のウエイトの自動挿入機能を有します．メモリ空間は，1Mバイト空間の3分割，および16Mバイト空間の3分割が可能です（詳細は6.2節参照）．

3.6 REFU(リフレッシュ・コントロール・ユニット)

16ビットのリフレッシュ・アドレスと，リフレッシュ・サイクルであることを示すリフレッシュ信号（$\overline{\text{REFRQ}}$）を発生し，DRAMのリフレッシュ動作をサポートします（詳細は6.3節参照）．

3.7 TCU(タイマ／カウンタ・ユニット)

μPD71054と同等の機能を有し，3チャネルの16ビット・タイマ／カウンタを内蔵しています（詳細は6.6節参照）．

3.8 SCU(シリアル・コントロール・ユニット)

μPD71051から同期モードを除いたものとほぼ同等の機能を有するシリアル・コントローラを内蔵し，RS－232－Cプロトコルをサポートします．さらに専用のボー・レート・ジェネレータを内蔵しています（詳細は6.7節参照）．

3.9 ICU(割り込みコントロール・ユニット)

μPD71059からCALLモード（8085モード）を除いたものと同等の機能を有しています．8レベルの外部割り込み入力端子を有し，カスケード接続によりさらに外部割り込み入力を拡張可能です（詳細は6.4節参照）．

3.10 DMAU(DMAコントロール・ユニット)

μPD71071およびμPD71037と同等の機能を有しています．システムI／O領域で設定することにより，いずれかの動作モードを選択します（詳細は6.5節参照）．

第4章　メモリとI／Oの構成

4.1　メモリ構成とアクセス方法

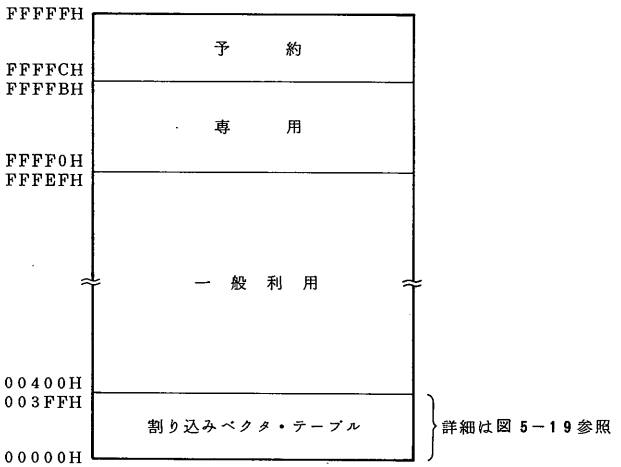
V53のメモリ・アクセスには，通常アドレス・モードと拡張アドレス・モードがあります．リセット直後は通常アドレス・モードとなります．モード切り替えについては「5.3.2アドレス空間拡張機能」を参照してください．

4.1.1　通常アドレス・モード

このモードでは，1Mバイト（512Kワード）までのメモリをアクセスでき，20ビット・アドレス・バスより出力されるアドレス情報（A19－A0）によってアドレスされます．

図4－1にメモリ・マップを示します．0H－3FFHの1Kバイトには，割り込みベクタ・テーブルが割り付けられています．ただし，システムによって使用しないテーブル・エリアはほかの目的に使用可能です．FFFF0H－FFFFBHの12バイトは，CPUがリセット・スタート等で自動的に使用しますので，ほかの目的に使用できません．FFFFCH－FFFFFHの4バイトは，将来の使用のため予約されていますので，ユーザは使用できません．

図4－1　通常アドレス・モード時のメモリ・マップ



メモリにストアされる要素としては，命令コード，割り込み開始アドレス，スタック・データ，一般変数等があり，バイト単位のものとワード単位のものがあります．

これらの要素に対して命令によって生成されるアドレスとしては，割り込み開始アドレス（割り込みベクタ・テーブル）のストアされているエリアが常に偶数（Ａ０＝０）でアドレスされるのを除けば，偶数（Ａ０＝０）と奇数（Ａ０＝１）の両方が考えられます．Ｖ５３のワード・データのアクセスは，そのアドレスが偶数でも奇数でも可能な設計となっており，命令の生成アドレスとして偶数，奇数の両方を許しています．

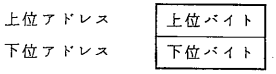
各メモリ要素のアドレス，データ構成を表４－１に示します．

表４－１　各メモリ要素のアドレス，データ構成

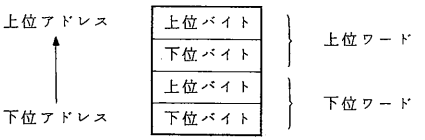
メモリ要素	アドレス	データ構成
命令コード	偶数／奇数	１／２／３／４／５／６バイト
割り込みベクタ・テーブル	偶数	２ワード／ベクタ
スタック	偶数／奇数	ワード
一般変数	偶数／奇数	バイト／ワード／ダブル・ワード

ワード・データとダブル・ワード・データは次に示すようにとらえられます．

ワード・データの構成

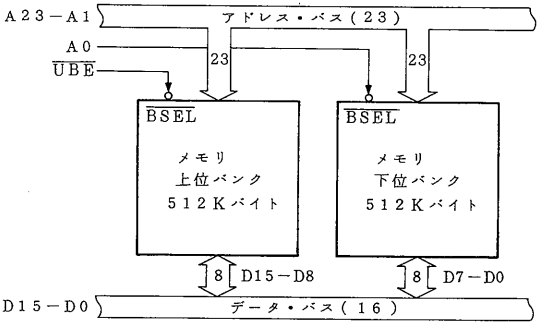


ダブル・ワード・データの構成



一方，メモリとのインタフェースは，図４－２のようになっています．

図４－２　メモリとのインタフェース



データ・バス幅は16ビットのため，本来16ビットのワード・データを1バス・サイクルで転送する能力を持っているわけですが，これは命令によって生成されたアドレスが偶数（A0＝0）のときだけで，奇数（A0＝1）のときはワード・データの転送に2バス・サイクルを必要とします．

図4－2において，A0はアクティブ・ロウでメモリの下位バンクのバイト・データをイネーブルにし，アドレス・バスからの情報とは別に$\overline{\text{UBE}}$（Upper Byte Enable）信号が出力されて，これもアクティブ・ロウでメモリの上位バンクのバイト・データをイネーブルにします．

奇数アドレスのワード・データをアクセスしようとする場合，初めのバス・サイクルで$\overline{\text{UBE}}$＝0，A0＝1となって上位バイトだけがアクセスされ，続いて自動的に$\overline{\text{UBE}}$＝1となり，アドレス情報の下位16ビット（A15－A0）がインクリメント（＋1）されて，すなわちA0＝0となって次のアドレスの下位バイトがアクセスされます．

偶数アドレスのワード・データは，$\overline{\text{UBE}}$＝0，A0＝0によって1バス・サイクルでアクセスされます．

これらのことは表4－2にまとめられています．

表4－2　データ・アクセス

オペランド	$\overline{\text{UBE}}$	A0	バス・サイクル数
偶数アドレスのワード	0	0	1
奇数アドレスのワード	0	1	2
	1	0	
偶数アドレスのバイト	1	0	1
奇数アドレスのバイト	0	1	1

命令コードのアクセス（プリフェッチ）に対しては，通常，ワード単位に行います．ただし，奇数アドレスへのブランチが起こった場合，その奇数アドレスの1バイトだけがフェッチされ，それ以降は再びワード単位にフェッチされます．

割り込みベクタ・テーブルのアクセスは，ベクタ番号（0－255）からベクタ・テーブルのアドレスが生成されるとき，必ず偶数アドレスが生成されますので，常に偶数アドレスのワード・データとして行われます．したがって，1つの割り込みに対するベクタ・テーブルのアクセスは，セグメント・ベースとオフセットの2ワードに対して常に2バス・サイクルで行われます．

メモリ・アクセスのための1つのバス・サイクルは2クロックを必要とします．したがって，奇数アドレスのワード・データを一度アクセスするごとに，偶数アドレスのワード・データをアクセスするのに比べて命令実行時間が2クロック余分に必要となります．命令の説明でワード・データ・アクセス回数が1以上ある場合，その命令を実行する際にこのことが適用されます．

メモリからメモリへのワード・データ転送の場合，ソースからの読み出しのためとデスティネーションへの書き込みとに2回のメモリ・アクセスが必要なため，両方とも奇数アドレスのとき実行時間は最大となります．

これらの奇数アドレスの問題は，スタック操作に対しても同様に起こります．割り込み処理でレ

ジスタ等が自動的にスタックに退避されますが，これらはすべてワード・データになっていますので，奇数アドレスで処理された場合，バス・サイクル数が2倍になり，割り込み応答時間を遅くしますので考慮が必要です．

以上のことから，ワード・データをアクセスする場合，プログラマがチェックできるものはすべて偶数アドレスに配置できるように考慮すべきです．

例　MOV　reg, mem命令の実行時間（クロック数）

バイト・データのとき	5
ワード・データのとき	7：奇数アドレス
	5：偶数アドレス

これはワード・データ・アクセスが1回行われる例です．

4．1．2　拡張アドレス・モード

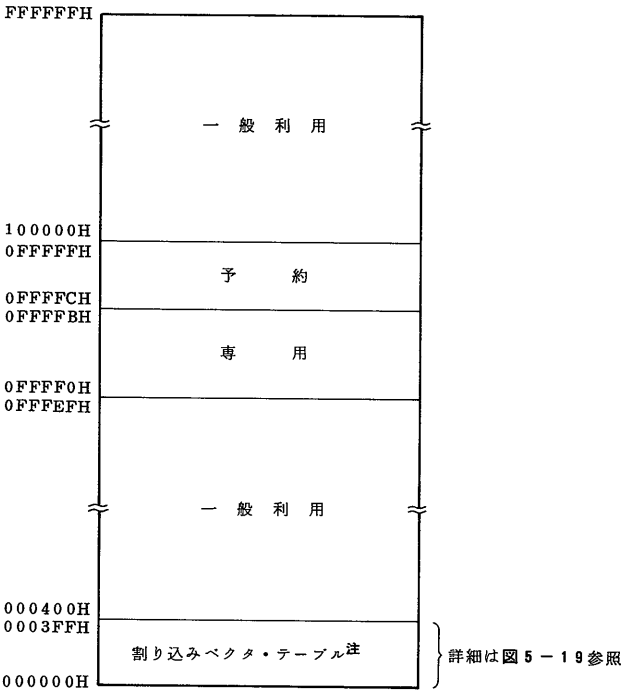
このモードでは16Mバイト（8Mワード）までのメモリをアクセスでき，24ビット・アドレス・バスより出力されるアドレス情報（A23－A0）によってアドレスされます．

図4－3にメモリ・マップを示します．0H→3FFHの1Kバイトには，割り込みベクタ・テーブルの下位14ビット（A13－A0）が割り付けられています．ただし，システムによって使用しないテーブル・エリアはほかの目的に使用可能です．なお，割り込みベクタ・テーブルの上位10ビット（A23－A14）は，アドレス変換テーブルのページ・レジスタ1（PGR1）の値で決まります．

FFFF0H－FFFFBHの12バイトは，CPUがリセット・スタート等で内部で使用しますので，ほかの目的には使用できません．FFFFCH－FFFFFHの4バイトは，将来の使用のために予約されていますので，使用できません．

アクセス方法は，通常アドレス・モードと同様です．

図4－3　拡張アドレス・モード時のメモリ・マップ例（PGR1＝000Hのとき）



注　アドレス変換テーブルのPGR1によって，16Mバイトのメモリ空間の任意領域に割り付けられます．

4.2 I／O構成とアクセス方法

V53は，64Kバイト（32Kワード）までのI／Oを，メモリとは独立なエリアでアクセスできます．

I／Oはアドレス・バスの下位16ビットより出力されるI／Oアドレス情報によってアドレスされます．

図4－4にI／Oマップを示します．

このうち，システムI／O領域（上位256バイトのI／O空間）は内部I／O領域として使用するため，外部I／Oをマッピングできません．

I／Oアドレスに対しては，メモリのようにセグメント方式は適用されていません．

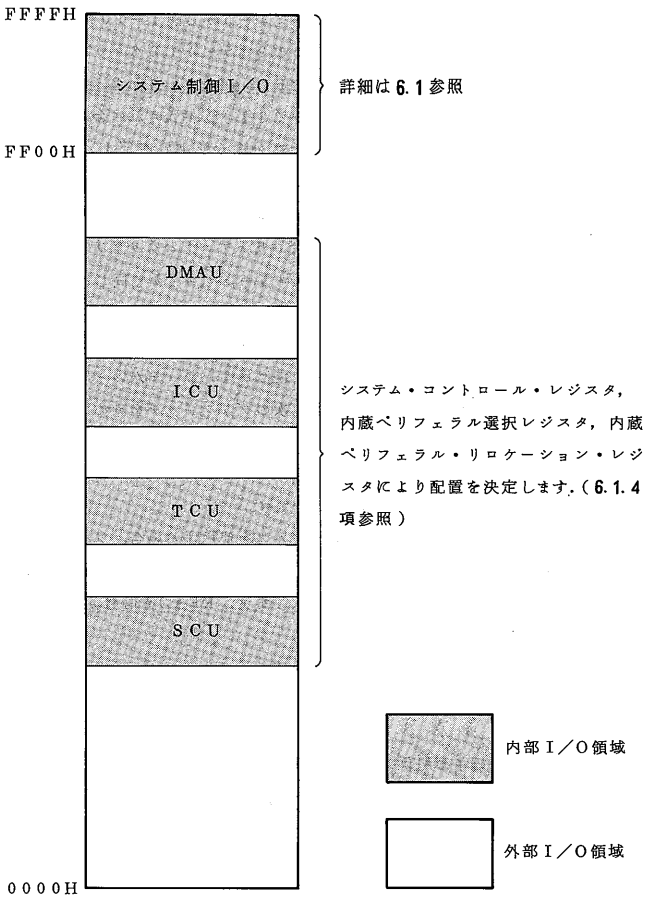
外部I／Oアクセスおよび内部I／Oアクセスタイミングでは，アドレス・バスの上位8ビット（A23－A16）には，すべて0が出力されます．

CPUとI／Oとのデータのやりとりは，バイトまたはワード単位の両方が可能で，8ビットI／Oデバイスおよび16ビットI／Oデバイスの両方が接続可能です．ただし，メモリの場合と同様に，ワード・データ・アクセスのとき偶数アドレスなら1バス・サイクル，奇数アドレスなら2バス・サイクルとなっています．

なお，バス・サイジング機能を使用しないで8ビットI／Oデバイスをアクセスする際は，I／OアドレスのうちA0はデバイス選択にのみ使用し，A1より上位の値をデバイス選択と一つのデバイス内のいくつかのレジスタ選択に使用します．つまり，偶数アドレスのI／Oデバイスはその内部レジスタもすべて偶数で，奇数アドレスのI／Oデバイスはその内部レジスタもすべて奇数で選択されるようにします．

また，外部I／O領域と内部I／O領域にまたがったワード・タイプのI／Oアクセスは禁止です．

図4－4　I／Oマップ



注意　V53の内蔵I／Oへのアクセスは，μPD71071モード時のDMAUおよびページ・レジスタを除きバイト・タイプのIN／OUT命令で行ってください．

4.3 メモリ，I／Oのリード／ライト・タイミング

メモリおよびI／Oリード／ライト・タイミングを図4－5から図4－10に示します．

メモリの1バス・サイクルは，基本的にT1，T2の2ステート（2クロック）で構成されています．1ステート（1クロック）は，16MHz動作時で62.5nsです．外部I／Oサイクルは，T1，T2の2ステート後，自動的に6アイドル・ステート（TI）が挿入され，合計8ステートになります．ページ・レジスタおよびXAMレジスタ・アクセスを除く通常の内部I／Oサイクルは T1，T2の2ステート後，自動的に2ウエイト・ステート（TW）および6アイドル・ステートが挿入され，合計10ステートになります．ページ・レジスタおよびXAMレジスタをアクセスする内部I／Oサイクルは，2ウエイト・ステートは挿入されず合計8ステートとなります．

V53では，命令コードのフェッチもメモリのデータ・リードもまったく同じタイミング（図4－5）で処理されます．

ある命令の実行に内部処理時間が多くかかる場合，その命令の実行を開始したあと，V53はキューに対するプリフェッチをキューがいっぱいになるまで続けます．キューがいっぱいになっても命令実行中でキューから命令コードをフェッチしないとき，次のプリフェッチをやめて，T2ステートの次にアイドル・ステート（TI）を自動的に挿入します．TIは，そのとき実行中の命令を終了し，次の命令コードをキューからフェッチするまで続けて挿入されます．

このあと，T1とバス・サイクルのステートを進めます．

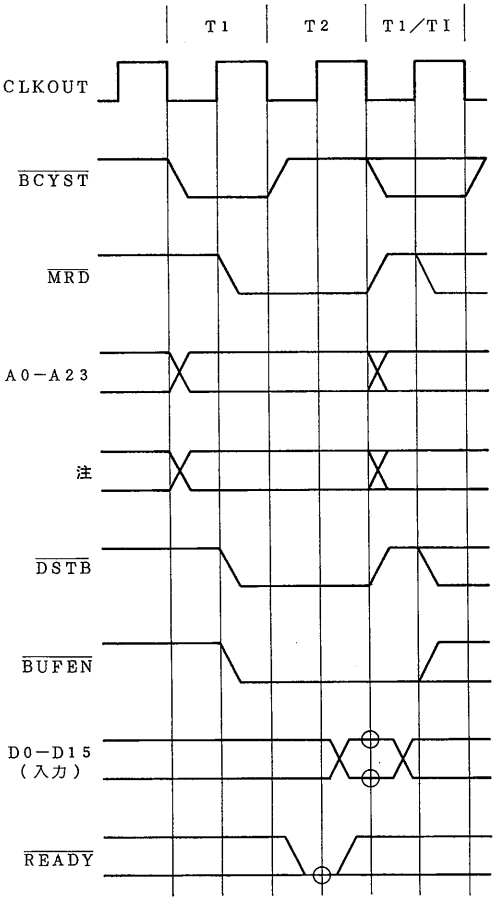
また，アクセス・タイムの遅いメモリやI／Oデバイスを使用する場合，メモリやI／Oデバイス側から送られてくる$\overline{\text{READY}}$信号を検出し，$\overline{\text{READY}}$信号がハイ・レベルであればT2ステートの次にT1に進まず，ウエイト・ステート（TW）を挿入します．$\overline{\text{READY}}$信号がロウ・レベルになるとTW，T1とバス・サイクルのステートを進めます．

TWの挿入されている間，各信号は同じレベルに保持され，リード／ライト・タイミングが拡張されます．

備考　WCUと$\overline{\text{READY}}$端子の関係および内部I／Oサイクルのウエイト・ステートの自動挿入については，**6.2.3 WCUと$\overline{\text{READY}}$端子の関係**および**6.2.4 バス・サイクル**を参照してください．

図4－5　CPUメモリ・リード（1／2）

(a) ウエイトなし



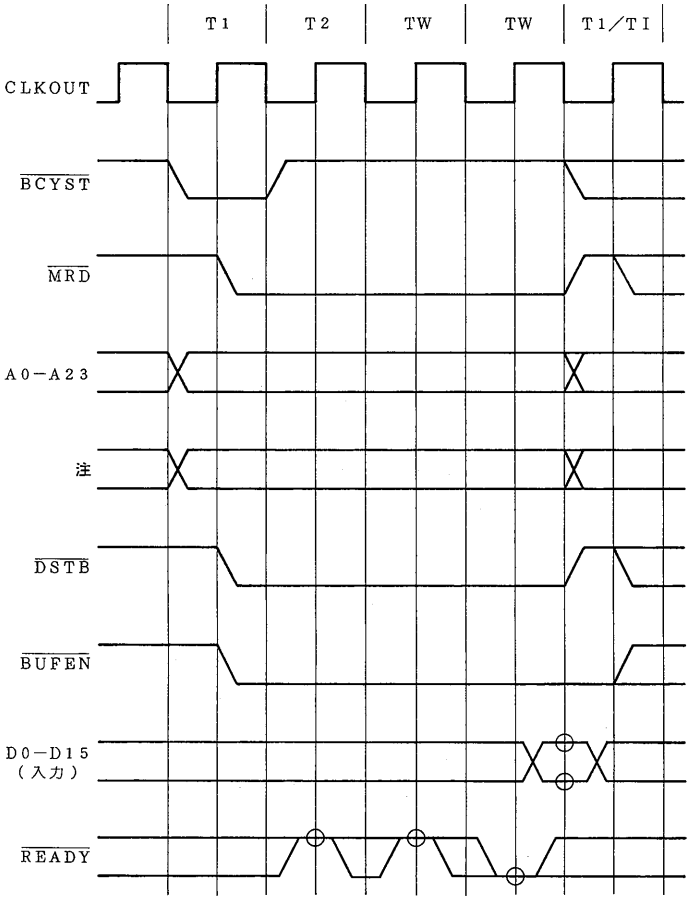
注　R／W，M／IO，BUSST0－BUSST2，UBE，AEX

備考　○印はWCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUとREADY端子の関係は，6.2.3 WCUとREADY端子の関係を参照してください．

図４－５　ＣＰＵメモリ・リード（２／２）

(b)　２ウエイト時



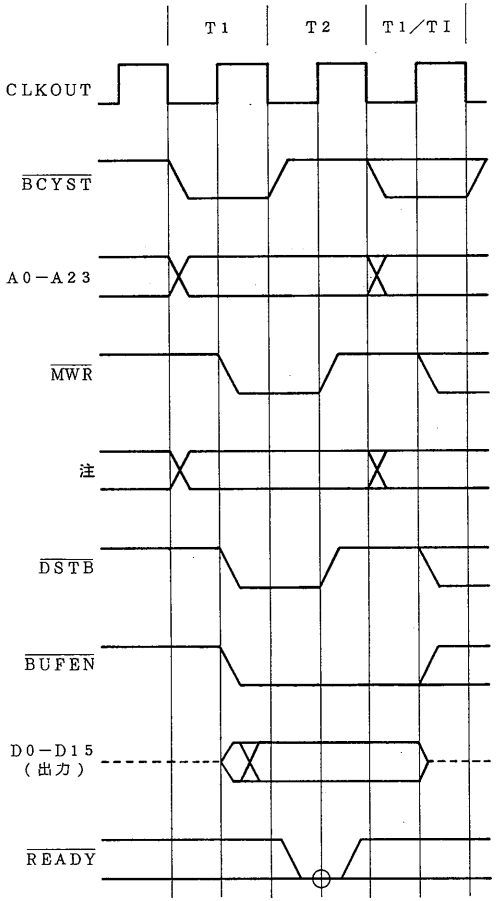
注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

備考　○印はWCUにウエイト・サイクル０が設定されているときに，サンプリングされるタイミングです．

WCUと$\overline{READY}$端子の関係は，6.2.3 WCUと$\overline{READY}$端子の関係を参照してください．

図4－6　CPUメモリ・ライト（1／2）

(a) ウエイトなし



注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

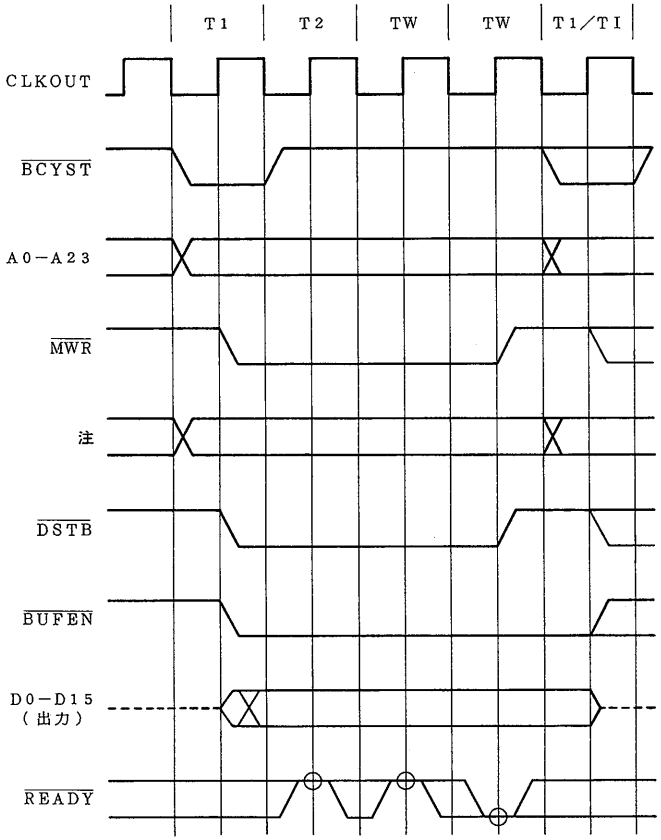
備考1.　○印はWCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUと$\overline{READY}$端子の関係は，6.2.3 WCUと$\overline{READY}$端子の関係を参照してください．

2.　破線はハイ・インピーダンスを示します．

図４－６　ＣＰＵメモリ・ライト（２／２）

(b)　２ウエイト時



注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

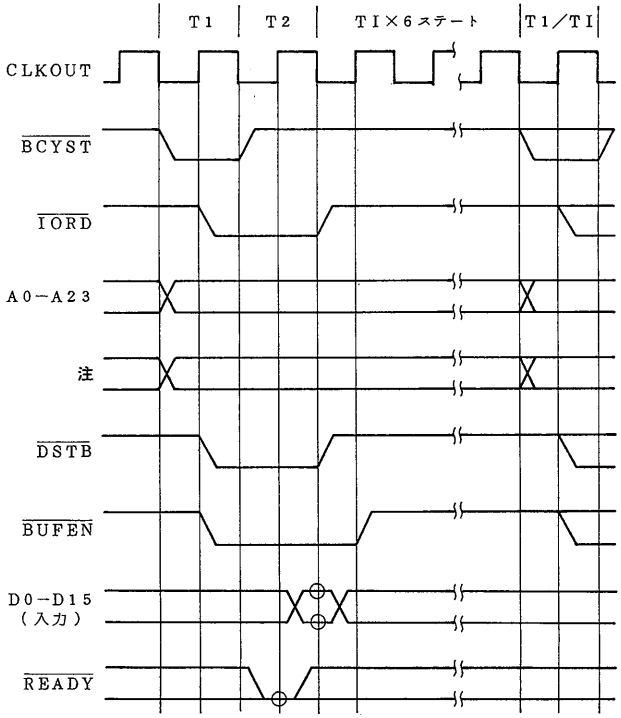
備考 1.　○印はWCUにウエイト・サイクル０が設定されているときに，サンプリングされるタイミングです．

WCUと$\overline{READY}$端子の関係は，6.2.3　WCUと$\overline{READY}$端子の関係を参照してください．

2.　破線はハイ・インピーダンスを示します．

図４－７　外部Ｉ／Ｏリード（１／２）

(a) ウエイトなし



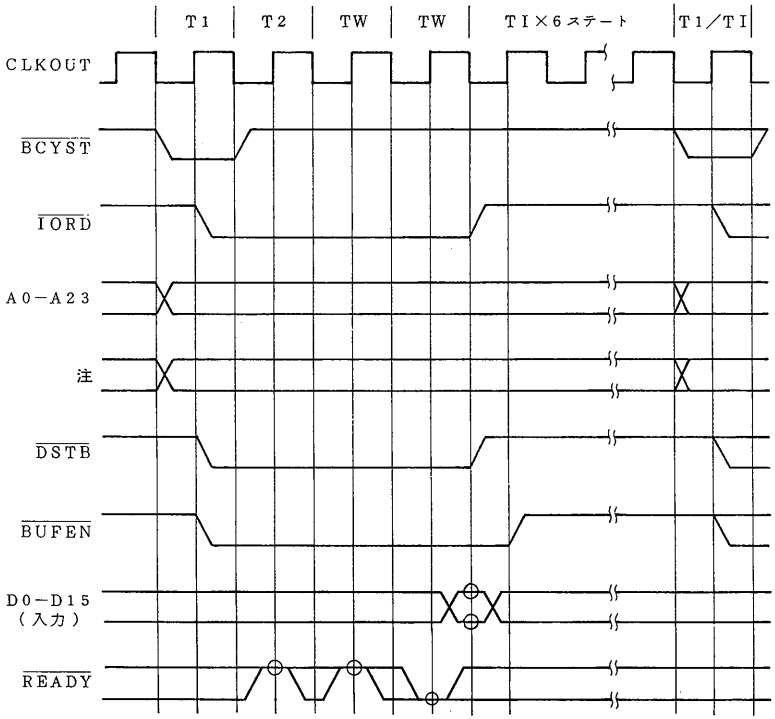
注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

備考　○印はWCUにウエイト・サイクル０が設定されているときに，サンプリングされるタイミングです．

WCUと$\overline{READY}$端子の関係は，6.2.3 WCUと$\overline{READY}$端子の関係を参照してください．

図４－７　外部Ｉ／Ｏリード（２／２）

(b)　２ウエイト時



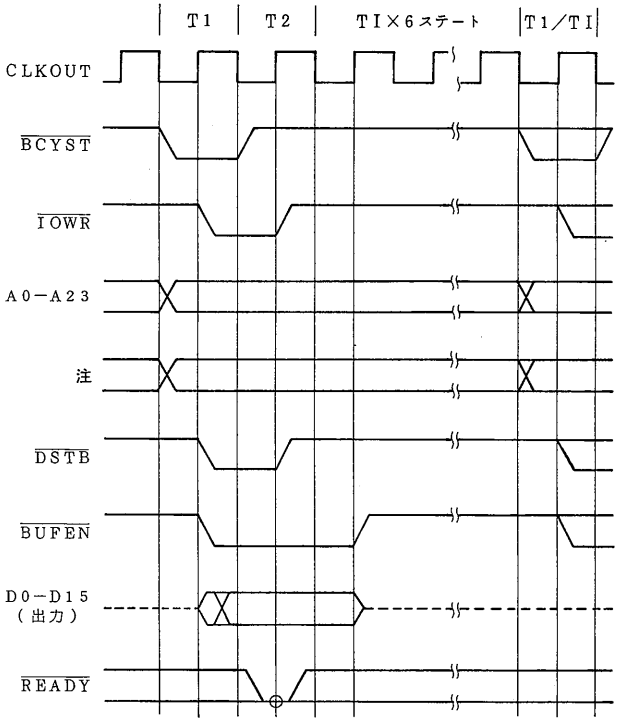
注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

備考　○印はWCUにウエイト・サイクル０が設定されているときに，サンプリングされるタイミングです．

WCUと$\overline{READY}$端子の関係は，**6.2.3 WCUと$\overline{READY}$端子の関係**を参照してください．

図４－８　外部Ｉ／Ｏライト（１／２）

(a) ウエイトなし



注　R／W，M／IO，BUSST0－BUSST2，UBE，AEX

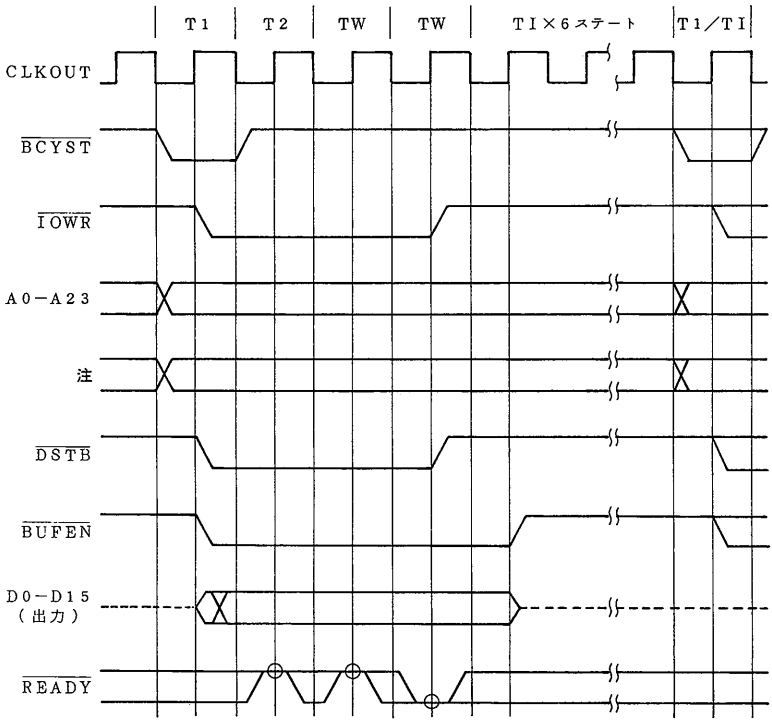
備考 1. ○印はWCUにウエイト・サイクル０が設定されているときに，サンプリングされるタイミングです．

WCUとREADY端子の関係は，6.2.3 WCUとREADY端子の関係を参照してください．

2. 破線はハイ・インピーダンスを示します．

図４－８　外部Ｉ／Ｏライト（２／２）

(b)　２ウエイト時



注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

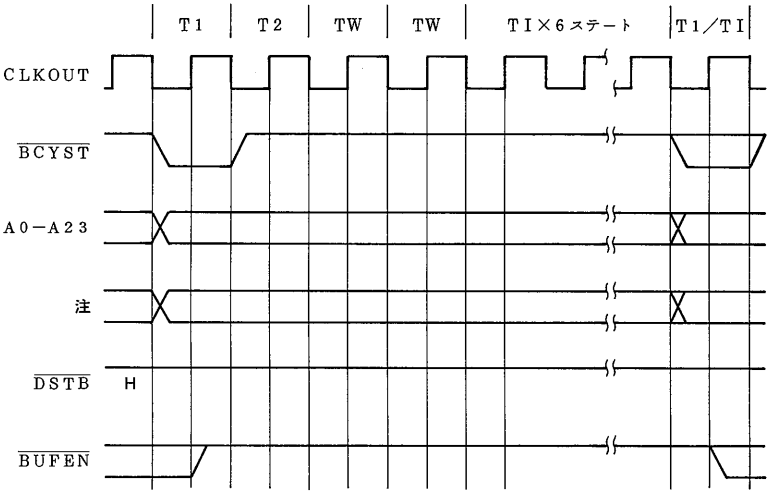
備考１．○印はWCUにウエイト・サイクル０が設定されているときに，サンプリングされるタイミングです．

WCUと$\overline{READY}$端子の関係は，6.2.3 WCUと$\overline{READY}$端子の関係を参照してください．

２．破線はハイ・インピーダンスを示します．

図４－９　内部Ｉ／Ｏリード（１／２）

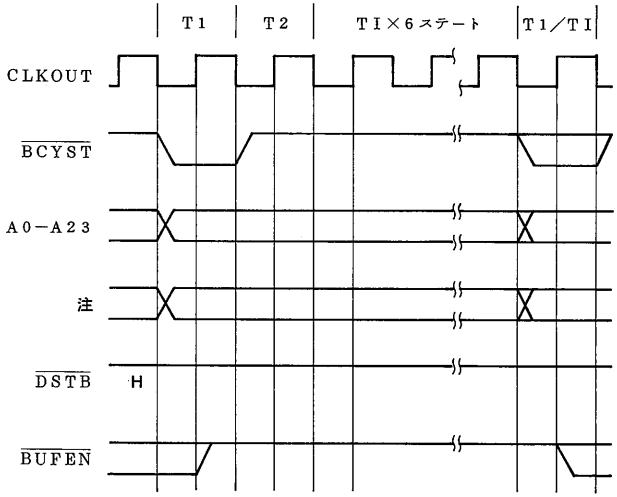
(a)　通常アクセス時



注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

図４－９　内部Ｉ／Ｏリード（２／２）

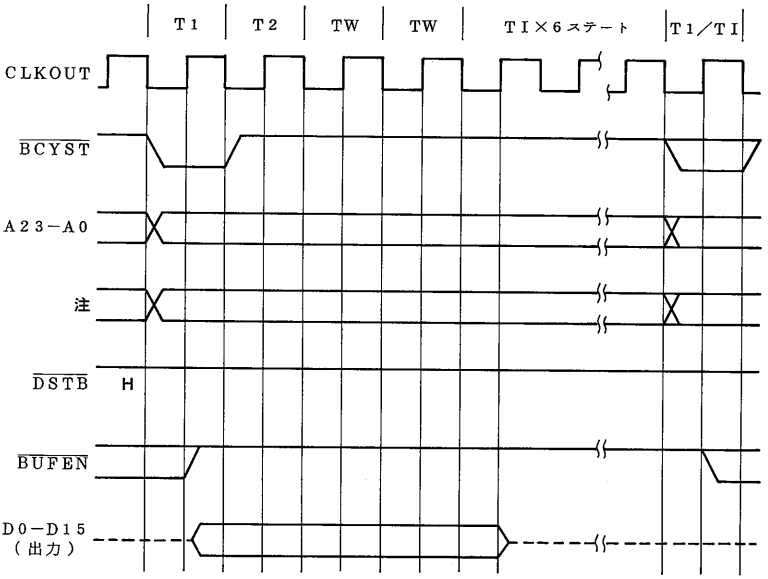
(b)　ページ・レジスタ，XAMレジスタ・アクセス時



注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

図４－１０　内部Ｉ／Ｏライト（１／２）

(a) 通常アクセス時

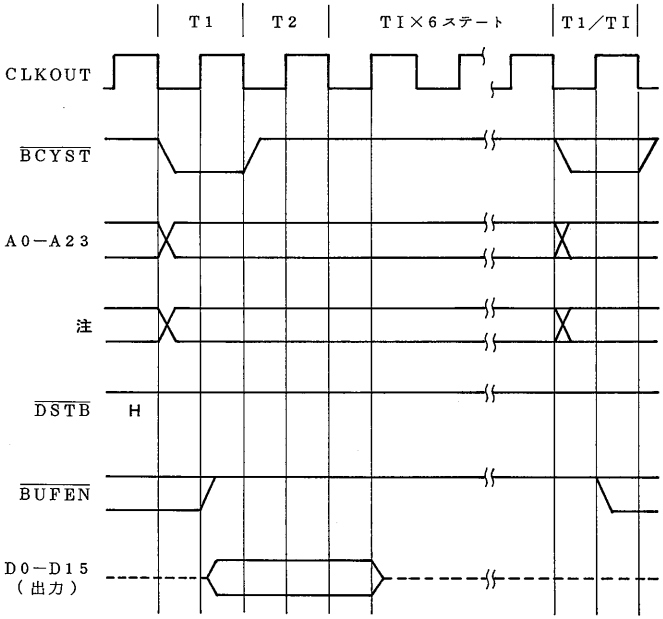


注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

備考　破線はハイ・インピーダンスを示します．

図4－10　内部I／Oライト（2／2）

(b)　ページ・レジスタ，XAMレジスタ・アクセス時



注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

備考　破線はハイ・インピーダンスを示します．

第５章　CPU

Ｖ５３のＣＰＵはＶ３３と同機能であり，Ｖ３０（１０ＭＨｚ動作時）に比べ約４倍（１６ＭＨｚ動作時）の高性能となっています．

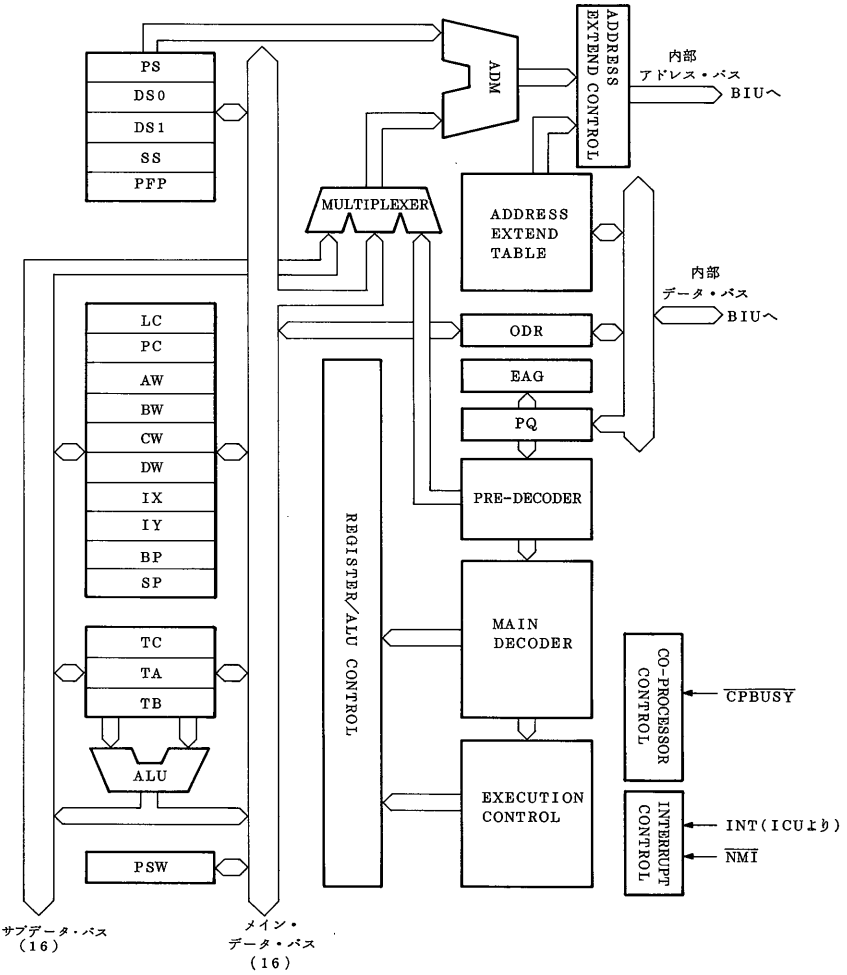
５．１　特　　徴

Ｖ５３のＣＰＵは，以下の特徴を有しています．

- Ｖ３３相当
  Ｖ５０（１０ＭＨｚ動作時）の約４倍の高性能（１６ＭＨｚ動作時）
- ２クロック／１バス・サイクル
- アドレス拡張機能
  １Ｍ↔１６Ｍバイト・メモリ空間の変更が可能です．
- Ｖ３３フルコンパチブルの命令セット
  Ｖ５０の命令セットに，アドレス拡張命令（ＢＲＫＸＡ，ＲＥＴＸＡ）を追加し，８０８０ＡＦエミュレーション命令を削除したものとなっています．
- 未定義命令に対するトラップ機能
  本ユーザーズ・マニュアルで定義していない命令コードに対し，ベクタ番号１２２の割り込みを発生します．

5.2 CPU内部

図5－1　CPUブロック図



５．２．１　ＣＰＵ内部構成

(1) ＰＣ（プログラム・カウンタ）

現在のプログラム・セグメント（ＰＳレジスタの内容をベース値とする６４Ｋバイトのメモリ空間）内で，現在実行している命令の先頭アドレス（セグメント・ベースに対するオフセット値）を格納している１６ビット・レジスタです．

ＰＣは，通常（分岐命令ではない場合），現在実行中の命令の先頭アドレスを指しており，次の命令の実行直前に更新します．分岐を伴う命令を実行した場合は，分岐先アドレスがセットされます．

(2) ＰＦＰ（プリフェッチ・ポインタ）

現在のプログラム・セグメント（ＰＳレジスタの内容をベース値とする６４Ｋバイトのメモリ空間）内で，次にプリフェッチする命令コードのアドレス（セグメント・ベースに対するオフセット値）を格納している１６ビット・レジスタです．

ＰＦＰは，通常（分岐命令ではない場合），プリフェッチした命令コードの次のアドレスを指しており，プリフェッチするたびにインクリメント（＋１）されます．分岐を伴う命令を実行した場合は，分岐先アドレスがセットされ，ＰＣと同じ内容になります．

ＰＦＰは，常にＰＳ（プログラム・セグメント・レジスタ）とともに使用されます．

(3) ＰＱ（プリフェッチ・キュー）

Ｖ５３は８バイト命令キュー（ＦＩＦＯ）を持っており，プリフェッチする命令コードを最大８バイト分までストアすることができます．

ブランチ，コール，リターン，ブレーク命令実行時や外部割り込み処理時には，キューの内容はクリアされ，新たなロケーションの命令がプリフェッチされます．

通常，Ｖ５３はキューに１ワード（２バイト）以上の空きがあるとプリフェッチを行います．

いくつかの連続して実行される命令の平均実行時間が，１つ１つの命令の命令コードのプリフェッチに必要なクロック数をある程度越えていれば，１つの命令を実行終了したときに，続いてEXUが実行できる命令コードがキューに用意されているようになり，外部メモリからのフェッチ時間を命令実行時間から除けるようになります．この結果，１命令ごとにフェッチ，実行を行うＣＰＵに比べて処理速度が向上できます．

キューの効果は，上述のブランチ命令実行時のようにキューがクリアされる命令が多いほど，また実行時間の短い命令が続いたときなどに減少します．

(4) セグメント・レジスタ（ＰＳ，ＳＳ，ＤＳ０，ＤＳ１）

ＣＰＵは，メモリ空間を６４Ｋバイト単位の論理セグメントに分解し，同時に４つのセグメントまで管理しています．各セグメントは，それぞれ次の４つのセグメント・レジスタで開始アドレスが指定されます．

ＰＳ　（プログラム・セグメント）
ＳＳ　（スタック・セグメント）
ＤＳ０（データ・セグメント０）
ＤＳ１（データ・セグメント１）

論理セグメント内でのオフセットは，特定のレジスタまたは実効アドレスが指定するようになっています．オフセットとセグメント・レジスタの組み合わせを表に示します．オフセットがPFPレジスタの場合，SPレジスタの場合，およびプリミティブ・ブロック転送命令，プリミティブ入力命令，BCDストリング命令，INS命令時でのIYレジスタの場合は，組み合わせることのできるセグメント・レジスタはそれぞれPS，SS，DS1に固定されており，ほかのセグメント・レジスタは使用できません．その他のオフセットでは，ディフォールトのセグメント・レジスタ以外に任意のセグメント・レジスタをセグメント・オーバライド・プリフィクスによって指定することができます．

表5－1　オフセットとセグメント・レジスタの組み合わせ

セグメント・レジスタ / オフセット	ディフォールト	オーバライト
PFPレジスタ	PS	不可
SPレジスタ	SS	不可
実効アドレス（BPベース）	SS	PS，DS0，DS1
実効アドレス（非BPベース）	DS0	PS，SS，DS1
命令群AでのIXレジスタ	DS0	PS，SS，DS1
命令群BでのIYレジスタ	DS1	不可

命令群A：プリミティブ・ブロック転送命令，プリミティブ出力命令，BCDストリング命令，EXT命令

命令群B：プリミティブ・ブロック転送命令，プリミティブ入力命令，BCDストリング命令，INS命令

(5) 汎用レジスタ（AW，BW，CW，DW）

4本の16ビット・レジスタ（AW，BW，CW，DW）から構成されます．各レジスタは，16ビット・レジスタとしても，2本の8ビット・レジスタ（AL，AH，BL，BH，CL，CH，DL，DH）としても使用できます．このように上位8ビット，下位8ビットに分割して使用することで，バイト・データを対象とする処理を効率良く実行できます．

各16ビット・レジスタ，および8ビット・レジスタの一部は，プログラマが命令の中で自由に指定できる汎用レジスタであるとともに，特定の命令処理においては次のような特別な役割を持っています．

AW：ワード・タイプの乗算，除算命令における下位ワード用データ・レジスタ
　　ワード・タイプの入出力命令におけるデータ・レジスタ
　　データ交換命令におけるデータ・レジスタ
　　可変長ビット・フィールド操作命令におけるデータ・レジスタ

BW：TRANS命令におけるベース・レジスタ
　　実行アドレス生成におけるベース・レジスタ

CW：ブロック・データ操作命令およびループ命令におけるループ・カウンタ

DW：ワード・タイプの乗算，除算命令における上位ワード用データ・レジスタ
　　入出力命令におけるポート指定用レジスタ

AL：バイト・タイプの乗算，除算命令における下位バイト用データ・レジスタ
　　バイト・タイプの入出力命令におけるデータ・レジスタ
　　データ変換命令におけるデータ・レジスタ
　　BCDローテート命令におけるデータ・レジスタ
　　TRANS命令におけるデータ・レジスタ

AH：バイト・タイプの乗算，除算命令における上位バイト用データ・レジスタ

CL：シフト，ローテート命令における繰り返し回数指定用レジスタ
　　ビット・データ操作命令におけるビット・アドレス・レジスタ
　　BCDストリング演算命令における桁数指定用レジスタ

(6) ポインタ（SP，BP）

2本の16ビット・レジスタ（BP，SP）から構成されます．各レジスタは，命令中で参照することもできますが，メモリ・データ参照時には，インデクス・レジスタとして用いられます．また，次のような特別な役割を持っています．

BP：メモリ・オペランド参照時の実効アドレス生成用ベース・レジスタ

SP：スタック・ポインタ

(7) インデクス・レジスタ（IX，IY）

2本の16ビット・レジスタ（IX，IY）から構成されます．各レジスタは，命令中で参照することもできますが，メモリ・データ参照時には，実効アドレス生成用のインデクス・レジスタとして用いられます．また，特定の命令処理においては次のような特別な役割を持っています．

IX：ブロック・データ操作命令におけるソース・オペランド用アドレス・レジスタ
　　可変長ビット・フィールド操作命令におけるベース・レジスタ
　　BCDストリング演算命令におけるソース・オペランド用アドレス・レジスタ

IY：ブロック・データ操作命令におけるデスティネーション・オペランド用アドレス・レジスタ
　　可変長ビット・フィールド操作命令におけるベース・レジスタ
　　BCDストリング演算命令におけるデスティネーション・オペランド用アドレス・レジスタ

(8) プログラム・ステータス・ワード（PSW）

プログラム・ステータス・ワードは，6種のステータス・フラグと3種のコントロール・フラグで構成されます．

ステータス・フラグ
・V　（オーバフロー）

・S　（サイン）
・Z　（ゼロ）
・AC（補助キャリー）
・P　（パリティ）
・CY（キャリー）

コントロール・フラグ
・DIR（方向）
・IE　（割り込み許可）
・BRK（ブレーク）

これらのフラグがスタック処理されるときは，次に示すようなワード・イメージで操作されます．

図5－2　PSWフォーマット

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
1	1	1	1	V	DIR	IE	BRK	S	Z	0	AC	0	P	1	CY	PSW

ステータス・フラグは，各種命令実行の結果（データ値）に従って自動的にセット，リセットされます．
CYフラグは，命令によって直接にセット，リセット，反転が可能です．
コントロール・フラグは，命令によってセット，リセットされ，CPUの動作を制御します．

(a) CY（キャリー・フラグ）

(i) 2進加減算

バイト演算の場合には，演算の結果ビット7からのキャリーまたはボローがあったときセットされ，なければリセットされます．

ワード演算の場合には，演算の結果ビット15からのキャリーまたはボローがあったときセットされ，なければリセットされます．

インクリメント，デクリメント命令では変化しません．

(ii) 論理演算

演算結果にかかわらずリセットされます．

(iii) 2進乗算

符号なしバイト演算の結果，AHが0ならばリセットされ，0以外ならばセットされます．

符号付きバイト演算の結果，AHがALのサイン拡張になっていればリセットされ，それ以外ではセットされます．

符号なしワード演算の結果，DWが0ならばリセットされ，0以外ならばセットされます．

符号付きワード演算の結果，DWがAWのサイン拡張になっていればリセットされ，それ以外ではセットされます．

8ビット・イミーディエト演算の場合は，積が16ビット以内のときはリセットされ，16ビットを越える場合にはセットされます．

(iv) 2進除算

不　定

(v) シフト／ローテート

CYを含むシフトおよびローテートの場合は，CYへシフトされるビットが1の場合はセットされ，0の場合はリセットされます．

(b) P（パリティ・フラグ）

(i) 2進加減算，論理演算，シフト

演算の結果を構成するビットの下位8ビットのうち，1のビットの数が偶数個のときセットされ，奇数個のときリセットされます．

結果がオール0のときにはセットされます．

(ii) 2進乗除算

不　定

(c) AC（補助キャリー・フラグ）

(i) 2進加減算

バイト演算の場合，下位4ビットから上位4ビットへのキャリーまたは上位4ビットから下位4ビットへのボローがあるときセットされ，それ以外のときリセットされます．

ワード演算では，下位バイトについてバイト演算の場合と同様の動作を行います．

(ii) 論理演算，2進乗除算，シフト／ローテート

不　定

(d) Z（ゼロ・フラグ）

(i) 2進加減算，論理演算，シフト／ローテート

バイト演算の場合には結果の8ビットが，ワード演算の場合には結果の16ビットがすべて0の場合にセットされ，それ以外のときにはリセットされます．

(ii) 2進乗除算

不　定

(e) S（サイン・フラグ）

(i) 2進加減算，論理演算，シフト／ローテート

バイト演算の場合，結果のビット7が1のときセットされ，0のときリセットされます．

ワード演算の場合，結果のビット15が1のときセットされ，0のときセットされます．

(ii) 2進乗除算

不　定

(f) V（オーバフロー・フラグ）

(i) 2進加減算

バイト演算の場合，ビット7からのキャリーとビット6からのキャリーが違っていればセットされ，同じであればリセットされます．

ワード演算の場合，ビット15からのキャリーとビット14からのキャリーが違っていればセットされ，同じであればリセットされます．

例1.　１２７＋１２７の場合

```
  01111111
+ 01111111
----------
  11111110
```

V＝1，CY＝0

例2.　（－1）＋（－1）の場合

```
  11111111
+ 11111111
----------
  11111110
```

V＝0，CY＝1

(ii)　2進乗算

符号なしバイト演算の結果，AHが0ならばリセットされ，0以外ならばセットされます．

符号付きバイト演算の結果，AHがALのサイン拡張になっていればリセットされ，それ以外ではセットされます．

符号なしワード演算の結果，DWが0ならばリセットされ，0以外ならばセットされます．

符号付きワード演算の結果，DWがAWのサイン拡張になっていればリセットされ，それ以外ではセットされます．

8ビット・イミーディエト演算の場合は，積が16ビット以内のときにはリセットされ，16ビットを越える場合にはセットされます．

(iii)　2進除算

リセットされます．

(iv)　論理演算

リセットされます．

(v)　シフト／ローテート

左1ビット・シフト／ローテートの場合，演算結果において

CY＝最上位ビットのとき　リセット

CY≒最上位ビットのとき　セット

されます．

右1ビット・シフト／ローテートの場合，演算結果において

最上位ビット＝最上位の次の下位ビットのとき　リセット

最上位ビット≒最上位の次の下位ビットのとき　セット

されます．

多ビット・シフト／ローテートの場合は不定となります．

(g)　BRK（ブレーク・フラグ）

PSWの一部としてスタックに退避されている状態でのみ，メモリ操作命令によってセットでき，セット後にPSWにリストアされると効果を持ちます．

BRKフラグがセットされていれば，命令を1つ実行するとソフトウエア割り込み（割り込みベクタ1）が自動的に発生し，1命令ずつのトレースが可能となります．

(h) ＩＥ（割り込み許可フラグ）

ＥＩ命令でセットされてＩＣＵ割り込みを許可状態にし，ＤＩ命令でリセットされてＩＣＵ割り込みを禁止状態にします．

(i) ＤＩＲ（方向フラグ）

ＳＥＴ１　ＤＩＲ命令でセットされ，ＣＬＲ１　ＤＩＲ命令でリセットされます．

ＤＩＲフラグがセットされていると，ブロック転送／入出力系命令において上位アドレスから下位アドレスへ向かって処理を行い，リセットされていると，下位アドレスから上位アドレスへ向かって処理を行います．

(9) ＴＡ／ＴＢ（テンポラリ・レジスタ／シフタＡ／Ｂ）

ＴＡ／ＴＢは，乗除算およびシフト／ローテート（ＢＣＤローテート含む）命令に用いられる１６ビット・テンポラリ・レジスタ／シフタです．

乗除算命令実行時には，ＴＡ＋ＴＢの３２ビット・テンポラリ・レジスタ／シフタとして働き，シフト／ローテート命令実行時には，ＴＢのみが１６ビット・テンポラリ・レジスタ／シフタとして働きます．

(10) ＴＣ（テンポラリ・レジスタＣ）

ＴＣは，乗除算その他の内部処理に用いられる１６ビット・テンポラリ・レジスタです．

(11) ＡＬＵ（算術論理演算回路）

ＡＬＵは，フル・アダーと論理演算回路で構成されており，算術演算（加減乗除，インクリメント，デクリメント，補数演算）と論理演算（テスト，ＡＮＤ，ＯＲ，ＸＯＲ，およびビット単位のテスト，セット，クリア，反転）を行います．

(12) ＬＣ（ループ・カウンタ）

ＬＣは，リピート・プリフィクス命令（ＲＥＰ，ＲＥＰＣなど）によって制御されるプリミティブ・ブロック転送／入出力命令（ＭＯＶＢＫ，ＯＵＴＭなど）のループ回数や，多ビット・シフト／ローテート命令のシフト数をカウントする１６ビット・レジスタです．

(13) ＯＤＲ（オペランド・データ・レジスタ）

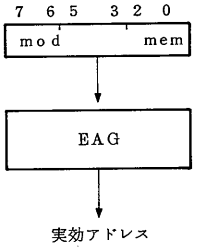
ＯＤＲは，オペランド・データを一時保持するためのテンポラリ・レジスタです．ＯＤＲにデータ転送を行うことでライト動作を終了し，外部データ・バスからＯＤＲにデータが転送されるのを確認してリード動作を終了します．

(14) ＡＤＭ（アドレス修飾回路）

ＡＤＭは，物理アドレスの生成（セグメント・レジスタとＰＦＰまたはＤＰとの加算）およびＰＦＰのインクリメントを行います．

(15) EAG（実効アドレス発生回路）

EAGは，メモリ・アクセス時に必要な実効アドレス計算を高速で行う回路です．すべてのアドレシング・モードに対して，2クロックで計算を終了します．



命令のオペランドの指定されているバイト（2バイト目または3バイト目）を取り込み，メモリ・アクセスが必要な場合，該当するレジスタ操作に関する制御信号を発生し，実効アドレスを計算します．

さらに，必要に応じて，バス・サイクル（メモリ・リードなど）を起動します．

5．2．2 パイプライン動作

(1) 基本動作

V53のパイプラインは，フェッチ，デコード，実行の3ステージで構成されています．ただし，分岐などによるキュー・フラッシュや，フェッチとオペランド・リード／ライトの競合，外部バス・サイクルのウエイト数などにより，必ずしも以下のように動作するとはかぎりません．

(a) フェッチ・ステージ

命令コードのプリフェッチは，8バイトの命令キューに対し2クロックの外部バス・サイクルを用いて行います．

(b) デコード・ステージ

命令キューに格納された命令コードを16ビットずつデコードします．デコードは16ビットあたり1クロックで行います．前半の半クロックで，命令コードを8ビット単位でアラインし，後半の半クロックでデコードを行います．命令コード長によって，クロック数が異なります．1，2バイト命令では1クロック，3，4バイト命令では2クロック，5，6バイト命令では3クロックかかります．

命令コードに含まれるイミーディエト・データ，およびディスプレースメントの獲得も行います．

1段のデコード済み命令バッファを持っており，次の実行ステージに移れば1命令分先行してデコードできます．

(c) 実行ステージ

最小2クロックで処理を行います．基本命令でメモリ・アクセスが必要であれば，実効アドレス計算を行います．リード・アクセスは実行ステージで行うため，リード・バッファは持っ

ていません．

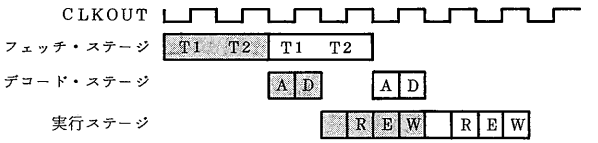
ライト・アクセスは，バス・サイクルを起動した時点で実行ステージの処理を終了します．

(2) 理想的な動作

基本命令（reg－reg'→reg，mem→reg，reg→mem，分岐）については，パイプライン処理サイクルは次のようになっています．

(a) reg－reg'→regタイプ

このタイプの命令コード長は2バイト（または1バイト）で，フェッチ・ステージは2クロック，デコード・ステージは1クロック，実行ステージは2クロックで処理できます．また，連続して処理される場合は2クロックで終了します．



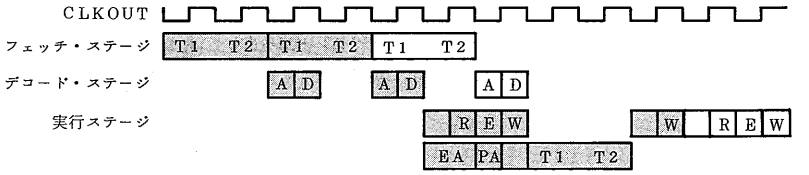
備考　T1，T2：バス・サイクル　　R：オペランド読み出し　　：現在の命令

A：オペコードのアライン　　E：実行処理　　：次の命令

D：デコード　　W：結果書き戻し

(b) mem→regタイプ

このタイプの命令コード長は2－4バイトで，多くの場合フェッチ・ステージでは2バス・サイクルを必要とします．したがって，フェッチ・ステージは4クロック，デコード・ステージは2クロック，実行ステージは5クロックで処理できます．



備考　T1，T2：バス・サイクル　　PA：物理アドレス計算　　：現在の命令

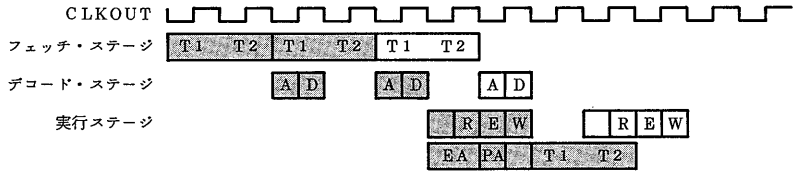
A：オペコードのアライン　　R：オペランド読み出し　　：次の命令

D：デコード　　E：実行処理

EA：実効アドレス計算　　W：結果書き戻し

(c) reg→memタイプ

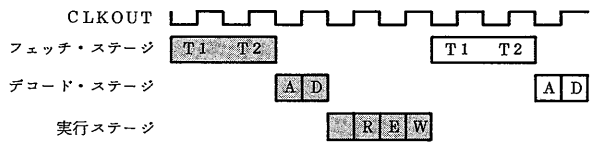
mem→regタイプと同じように，命令コード長は2－4バイトです．フェッチ・ステージは4クロック，デコード・ステージは2クロック，実行ステージは3クロックで処理できます．



備考　T1，T2：バス・サイクル　　PA：物理アドレス計算　　：現在の命令
A：オペコードのアライン　　R：オペランド読み出し　　：次の命令
D：デコード　　E：実行処理
EA：実効アドレス計算　　W：結果書き出し

(d) 分岐タイプ

最も頻繁に実行される条件付き分岐命令（Bcc）の場合は，命令コード長は2バイトです．フェッチ・ステージは2クロック，デコード・ステージは1クロック，実行ステージは2クロックで処理できますが，フェッチ・ステージがフラッシュされるため，分岐命令が連続する場合は5クロックで処理します．



備考　T1，T2：バス・サイクル　　R：オペランド読み出し　　：現在の命令
A：オペコードのアライン　　E：実行処理　　：次の命令
D：デコード　　W：結果書き戻し

5.3 アドレス空間

5.3.1 論理アドレスと物理アドレス

(1) アクセス方法

V53は24ビット・アドレス・バスを備えており，16メガ・バイトまでのメモリ空間をアクセスすることができます．

しかし，プログラマがハードウエアに直接対応した16Mバイトのアドレス（これを物理アドレスと呼びます）を管理しながらプログラムを作成することは不可能に近いものです．

そこで，V53では，プログラム作成の際に物理アドレスに直接依存しない小単位（64Kバイト）の論理アドレス空間の集合としてとらえるようにセグメント方式をとっています．

セグメントにはプログラム，スタック，データ0，データ1の4種類があり，それぞれ16ビット・セグメント・レジスタ（PS，SS，DS0，DS1）で決定される論理セグメント先頭アドレスからのオフセット値によってアドレスが指定されます．

セグメント・レジスタ / オフセット	ディフォールト	オーバライド
PFPレジスタ	PS	不可
SPレジスタ	SS	不可
実効アドレス（BPベース）	SS	PS, DS0, DS1
実効アドレス（非BPベース）	DS0	PS, SS, DS1
命令群AでのIXレジスタ	DS0	PS, SS, DS1
命令群BでのIYレジスタ	DS1	不可

命令群A：プリミティブ・ブロック転送命令，プリミティブ出力命令，BCDストリング命令，EXT命令

命令群B：プリミティブ・ブロック転送命令，プリミティブ入力命令，BCDストリング命令，INS命令

次に各セグメントの働きを示します．

(a) プログラム・セグメント

プログラム・セグメント・レジスタ（PS）によってこのセグメントの先頭アドレスが決定され，プリフェッチ・ポインタ（PFP）によって先頭アドレスからのオフセットが指定されます．

このセグメントには，命令コード，テーブル・データ等が配置されます．

セグメント・オーバライド・プリフィクス（PS：）を用いれば，一般変数およびソース・ブロック・データとしてこのセグメント内のデータをアクセスすることができます．

(b) スタック・セグメント

スタック・セグメント・レジスタ（SS）によってこのセグメントの先頭アドレスが決

定され，スタック・ポインタ（ＳＰ）によって先頭アドレスからのオフセットが指定されます．

割り込みやサブルーチン処理の際に，リターン・アドレス（ＰＳ，ＰＣの内容）やＰＳＷ，汎用レジスタ等の内容の退避およびパラメータの受け渡しエリアとして利用されます．

セグメント・オーバライド・プリフィクス（ＳＳ：）を用いれば，一般変数，ソース・ブロック・データとしてこのセグメント内のデータをアクセスすることができます．

一般変数のアドレシングにおいて，ベース・レジスタとしてＢＰが指定されたとき，セグメント・レジスタとしてＳＳが自動的に使用され，実効アドレスによってオフセットが指定されるため，一般変数のようにスタック・セグメント内のデータを操作することができます．

(c) **データ・セグメント０**

データ・セグメント０レジスタ（ＤＳ０）によってこのセグメントの先頭アドレスが決定され，実効アドレスによって先頭アドレスからのオフセットが指定されます．

このセグメントは，一般変数の格納エリアに利用されます．

ブロック転送，ＢＣＤストリング演算命令等の実行の際は，ソース・ブロック・データをストアするのに利用されます．ただし，この場合，オフセットはＩＸレジスタの内容によって決定されます．

ベース・レジスタとしてＢＰを指定したとき，ＳＳがセグメント・レジスタとして自動的に選択されますが，“ＤＳ０：”オーバライド・プリフィクスを用いれば，データ・セグメント０を使用することができます．

(d) **データ・セグメント１**

データ・セグメント１レジスタ（ＤＳ１）によってこのセグメントの先頭アドレスが決定され，ＩＹレジスタによって先頭アドレスからのオフセットが指定されます．

ブロック転送，ＢＣＤストリング演算命令等の実行の際，デスティネーション・ブロック・データをストアするのに利用されます．

セグメント・オーバライド・プリフィクス（ＤＳ１：）を用いれば，一般変数（オフセットは実効アドレスによって決定される）やソース・ブロック・データ（オフセットはＩＸによって決定される）としてこのセグメント内のデータをアクセスすることができます．

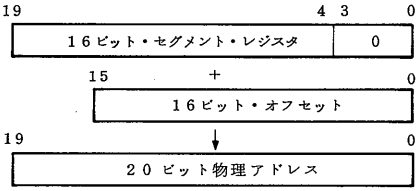
プログラムは，使用される（ディフォールトのもの，またはオーバライド指定したもの）セグメント・レジスタの内容とその値からのオフセット値（すなわち，セグメント・レジスタの値を０番地と考えれば，そのセグメント内のアドレスは，オフセット値がそのまま論理アドレスとなります）に着目するだけで，プログラム作成を行うことができます．

こうして論理アドレスで指定されるセグメントの集合として作られたいくつかのプログラムは，個々にコンパイル／アセンブルされて複数のオブジェクト・モジュールとなります．個々のオブジェクト・モジュールは，セグメント名，サイズ，内容区分，制御情報等を持っており，リンカへの指示となります．

複数のオブジェクト・モジュールはリンカにかけられ，物理アドレスに対応したセグメント・ベ

ース等が指定されて実際のメモリにロードできる状態になります．

セグメント・レジスタおよびオフセットと物理アドレスとの関係は次の通りです．



すなわち，セグメント・レジスタの内容を16倍したものにオフセット値を加えたものが物理アドレスとなります．ここで，セグメント・レジスタの内容およびオフセット値は，符号なしデータとして扱われます．

あるプログラムが，その中でセグメント・ベースを変更するような命令を実行していなければ，（ブランチや変数参照等がセグメント内），そのプログラム内のアドレスはすべてセグメント・レジスタの値からのオフセットだけで決定できるため，セグメント・レジスタの内容をそのプログラムをロードしようとしているメモリの物理アドレスに合わせるだけで，メモリ内の任意空間にロードできます．

このことを利用すれば，フロッピィ・ディスク等の外部ファイルにあるプログラムを実行させるのに，OSがメモリの領域およびセグメント・レジスタの管理を行って，空いている任意のメモリ領域にプログラムをロードして実行させることができます．

このように，別のファイルにばらばらにまたは集合して置いてあるプログラムを，そのプログラムを実行させようとするたびに，そのときたまたま空いているメモリ空間に配置することをダイナミック・リロケーションといいます．

5．3．2　アドレス空間拡張機能

(1)　概　　略

V53は，EAG（実効アドレス発生回路）により生成された20ビット長の物理アドレスを，24ビット長の拡張アドレスに拡張する機能を有しています（図5－3参照）．拡張方式としては，アドレス・リロケーション方式を採ります．この拡張／非拡張の切り替えは，命令実行によるモードの切り替えによって行います（(4)項参照）．

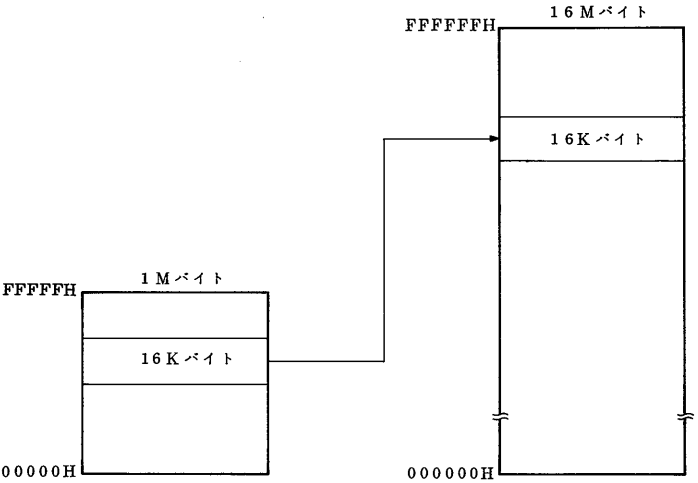
V53は内部I／O領域（FF00H－FF7FH）に64本のページ・レジスタより構成されるアドレス変換テーブルを有しています．アドレス拡張時は20ビット長の物理アドレスの上位6ビットでこのテーブルをアクセスし，10ビットの拡張アドレスに変換します．

リロケーション単位は1ページ16Kバイトで，16Mバイト（64ページ）までメモリ空間を拡張できます．

拡張アドレス・モードが設定されていない場合は，20ビットの物理アドレスをそのまま出力し，上位4ビット（A20－A23）は“0”を出力します．

なお，I／Oサイクル，DMAサイクル，リフレッシュ・サイクルに対しては，このアドレス拡張機能は無効です．

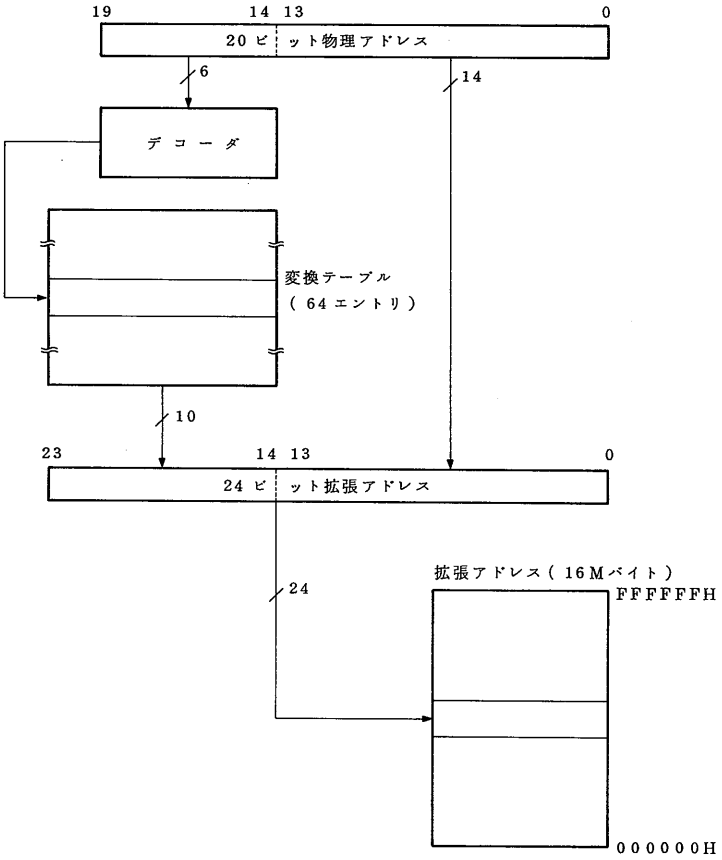
図5－3　アドレス・リロケーション概念図



(2) アドレス変換方式

V53のアドレス拡張機能はリロケーション方式により実現しています．つまり，20ビットの物理アドレスの上位6ビットにより，64エントリのアドレス変換テーブルをアクセスし，10ビットの拡張アドレスに置き換えます．下位14ビットはそのまま使用するため，16Kバイト単位でアドレス拡張を制御できます．

図5－4　アドレス変換方式



(3) アドレス変換テーブル

V53は内部I／O領域のFF00H－FF7FHに64エントリのアドレス変換テーブルを持っており，64本のページ・レジスタ（PGR1－PGR64）が割り付けられています．

(a) アドレス変換

アドレス変換テーブルは，20ビット物理アドレスの上位6ビットを入力し，64本のページ・レジスタ（PGR1－PGR64）より1本を選択し，OUT命令により書き込まれたD9－D0までの10ビット・データを拡張アドレスの上位10ビットとします．

表5－2　ページ・レジスタの選択

A19	A18	A17	A16	A15	A14	選択されるページ・レジスタ	A19	A18	A17	A16	A15	A14	選択されるページ・レジスタ
0	0	0	0	0	0	PGR1	1	0	0	0	0	0	PGR33
0	0	0	0	0	1	PGR2	1	0	0	0	0	1	PGR34
0	0	0	0	1	0	PGR3	1	0	0	0	1	0	PGR35
0	0	0	0	1	1	PGR4	1	0	0	0	1	1	PGR36
0	0	0	1	0	0	PGR5	1	0	0	1	0	0	PGR37
0	0	0	1	0	1	PGR6	1	0	0	1	0	1	PGR38
0	0	0	1	1	0	PGR7	1	0	0	1	1	0	PGR39
0	0	0	1	1	1	PGR8	1	0	0	1	1	1	PGR40
0	0	1	0	0	0	PGR9	1	0	1	0	0	0	PGR41
0	0	1	0	0	1	PGR10	1	0	1	0	0	1	PGR42
0	0	1	0	1	0	PGR11	1	0	1	0	1	0	PGR43
0	0	1	0	1	1	PGR12	1	0	1	0	1	1	PGR44
0	0	1	1	0	0	PGR13	1	0	1	1	0	0	PGR45
0	0	1	1	0	1	PGR14	1	0	1	1	0	1	PGR46
0	0	1	1	1	0	PGR15	1	0	1	1	1	0	PGR47
0	0	1	1	1	1	PGR16	1	0	1	1	1	1	PGR48
0	1	0	0	0	0	PGR17	1	1	0	0	0	0	PGR49
0	1	0	0	0	1	PGR18	1	1	0	0	0	1	PGR50
0	1	0	0	1	0	PGR19	1	1	0	0	1	0	PGR51
0	1	0	0	1	1	PGR20	1	1	0	0	1	1	PGR52
0	1	0	1	0	0	PGR21	1	1	0	1	0	0	PGR53
0	1	0	1	0	1	PGR22	1	1	0	1	0	1	PGR54
0	1	0	1	1	0	PGR23	1	1	0	1	1	0	PGR55
0	1	0	1	1	1	PGR24	1	1	0	1	1	1	PGR56
0	1	1	0	0	0	PGR25	1	1	1	0	0	0	PGR57
0	1	1	0	0	1	PGR26	1	1	1	0	0	1	PGR58
0	1	1	0	1	0	PGR27	1	1	1	0	1	0	PGR59
0	1	1	0	1	1	PGR28	1	1	1	0	1	1	PGR60
0	1	1	1	0	0	PGR29	1	1	1	1	0	0	PGR61
0	1	1	1	0	1	PGR30	1	1	1	1	0	1	PGR62
0	1	1	1	1	0	PGR31	1	1	1	1	1	0	PGR63
0	1	1	1	1	1	PGR32	1	1	1	1	1	1	PGR64

(b) ページ・レジスタ（PGR1−PGR64）

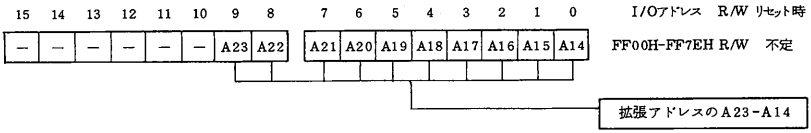
各ページ・レジスタはI／O空間のFF00H−FF7EHに割り付けられている64本の16ビット・レジスタです．ワード・タイプのIN／OUT命令でリード／ライトします．

ページ・レジスタの有効ビットはいずれも下位10ビット（D9−D0）です．有効ビット以外の上位ビット（D15−D10）は，リード時は“0”が読み出され，ライト時は無視されます．リセット入力の影響は受けません．

注意 拡張アドレス・モード（XA=1）時は，ページ・レジスタをアクセスしないでください．

表5−3 ページ・レジスタ一覧

ページ・レジスタ（PGR1−PGR64）



I/Oアドレス	ページ・レジスタ	I/Oアドレス	ページ・レジスタ	I/Oアドレス	ページ・レジスタ
FF00	PGR 1	FF2C	PGR23	FF58	PGR45
FF02	PGR 2	FF2E	PGR24	FF5A	PGR46
FF04	PGR 3	FF30	PGR25	FF5C	PGR47
FF06	PGR 4	FF32	PGR26	FF5E	PGR48
FF08	PGR 5	FF34	PGR27	FF60	PGR49
FF0A	PGR 6	FF36	PGR28	FF62	PGR50
FF0C	PGR 7	FF38	PGR29	FF64	PGR51
FF0E	PGR 8	FF3A	PGR30	FF66	PGR52
FF10	PGR 9	FF3C	PGR31	FF68	PGR53
FF12	PGR10	FF3E	PGR32	FF6A	PGR54
FF14	PGR11	FF40	PGR33	FF6C	PGR55
FF16	PGR12	FF42	PGR34	FF6E	PGR56
FF18	PGR13	FF44	PGR35	FF70	PGR57
FF1A	PGR14	FF46	PGR36	FF72	PGR58
FF1C	PGR15	FF48	PGR37	FF74	PGR59
FF1E	PGR16	FF4A	PGR38	FF76	PGR60
FF20	PGR17	FF4C	PGR39	FF78	PGR61
FF22	PGR18	FF4E	PGR40	FF7A	PGR62
FF24	PGR19	FF50	PGR41	FF7C	PGR63
FF26	PGR20	FF52	PGR42	FF7E	PGR64
FF28	PGR21	FF54	PGR43		
FF2A	PGR22	FF56	PGR44		

注意 ページ・レジスタ（PGR1−PGR64）へは，常に偶数番地に対するワード・アクセスを行ってください．

(4) 拡張アドレス・モードの設定／解除

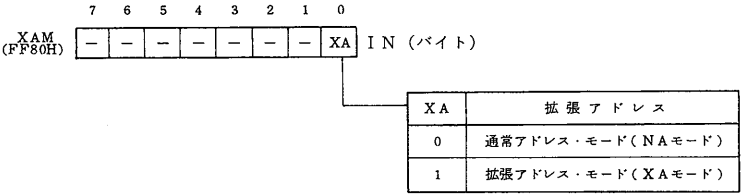
拡張アドレス・モード（XAモード）の設定／解除は

- BRKXA命令　（設定）
- RETXA命令　（解除）

により行います．

拡張アドレス・モードに設定されているか否かは，XAMレジスタ（FF80H）のビット0（XAフラグ）により知ることが可能です．

図5－5　拡張アドレス・モード・レジスタ（XAM）



XA	拡張アドレス
0	通常アドレス・モード（NAモード）
1	拡張アドレス・モード（XAモード）

備考　－：不定

(a) XAフラグの意味

XAフラグ	動作
0	通常アドレス・モード
1	拡張アドレス・モード

(b) XAフラグ設定／解除命令の動作

命　令	動作
BRKXAn	ベクタnをリードし，分岐（XAフラグを“1”にセット）
RETXAn	ベクタnをリードし，分岐（XAフラグを“0”にクリア）

注意　アドレス拡張／アドレス非拡張は，BRKXA，RETXAの実行による分岐先アドレスのフェッチ・サイクルから有効となります．

(c) XAフラグのリード／ライト

XAフラグはバイト・タイプのIN命令により，リードのみ可能です．

(d) リセット

リセットによりXAフラグのみが，“0”（通常アドレス・モード）となります．

(5) 拡張アドレス・モードに関する初期化

$\overline{\text{RESET}}$信号により，XAフラグおよびアドレス変換テーブルが以下の状態になります．

・XAフラグ：“0”（通常アドレス・モード）

・アドレス変換テーブル：

電源投入時…… 不定です．

通常リセット時…… リセット前の状態を保持しています．

備考　通常リセットとは，V53がすでに動作中であり，かつ電源電圧V_{DD}が定格の範囲内であるときに$\overline{\text{RESET}}$信号をアクティブにすることです．

5.4 ダイナミック・バス・サイジング機能

V53は外部16ビット・データ・バスとして設計されていますが，8ビット・データ・バス・システムに対しても接続可能なようにダイナミック・バス・サイジング機能を取り入れています．

V53のダイナミック・バス・サイジング機能とは，ダイナミック・バス・サイジング用信号（$\overline{BS8}$／BS16）により，データ・バス幅を8ビットまたは16ビットに切り替える機能です．

このバス・サイジング機能はCPUメモリ・アクセス，外部I／Oアクセスの両方に有効であり，$\overline{BS8}$／BS16端子がロウ・レベルであることがサンプリングされると，下位8ビット・データ・バスのみが有効となります．

偶数番地へのワード・アクセスの際は通常の第1バス・サイクルに引き続き，第2バス・サイクルを起動して上位8ビット・データのリード／ライトをD7－D0を用いて行います．

この第1バス・サイクルと第2バス・サイクルは連続で，間に別のバス・サイクル（DMAサイクルなど）が挿入されることはありません．

一方，$\overline{BS8}$／BS16端子がハイ・レベルであることがサンプリングされると，16ビット・データ・バスとなります．

表5－4に非サイジング時およびサイジング時のデータ・バス入出力を示します．また，図5－6，5－7にメモリおよび外部I／Oアクセスの偶数番地に対するワード・アクセス時のバス・サイジング・タイミング図を示します．

なお，DMAサイクル，リフレッシュ・サイクルでは$\overline{BS8}$／BS16端子の指定は意味を持たず，バス・サイジング機能は無効です．

バス・サイジング無効時は，データ・バスは常に16ビット・バスとして機能します．

表5－5に，バス・サイジングと$\overline{BS8}$／BS16端子のサンプリングの有無を，バス・サイクルごとに示します．

表5－4　データ・バスの入出力

(a)　ライト動作

バイト／ワード	アドレス	A0	$\overline{UBE}$	サイクル	非サイジング（$\overline{BS8}$／BS16＝1）		サイジング（$\overline{BS8}$／BS16＝0）	
					D15－D8	D7－D0	D15－D8	D7－D0
バイト	偶数	0	1	第1	不定	下位バイト	不定	下位バイト
	奇数	1	0	第1	下位バイト	下位バイト	下位バイト	下位バイト
ワード	偶数	0	0	第1	上位バイト	下位バイト	上位バイト	下位バイト
		1	1	第2	サイクルなし	サイクルなし	上位バイト	上位バイト
	奇数	1	0	第1	下位バイト	下位バイト	下位バイト	下位バイト
		0	1	第2	下位バイト	上位バイト	下位バイト	上位バイト

表5－4　データ・バスの入出力

(b)　リード動作

バイト／ワード	アドレス	A0	$\overline{UBE}$	サイクル	非サイジング（$\overline{BS8}$／BS16＝1）		サイジング（$\overline{BS8}$／BS16＝0）	
					D15－D8	D7－D0	D15－D8	D7－D0
バイト	偶数	0	1	第1	—	下位バイト	—	下位バイト
	奇数	1	0	第1	下位バイト	—	—	下位バイト
ワード	偶数	0	0	第1	上位バイト	下位バイト	—	下位バイト
		1	1	第2	サイクルなし	サイクルなし	—	上位バイト
	奇数	1	0	第1	下位バイト	—	—	下位バイト
		0	1	第2	—	上位バイト	—	上位バイト

備考　－は，V53が該当データを取り込まないことを示します．

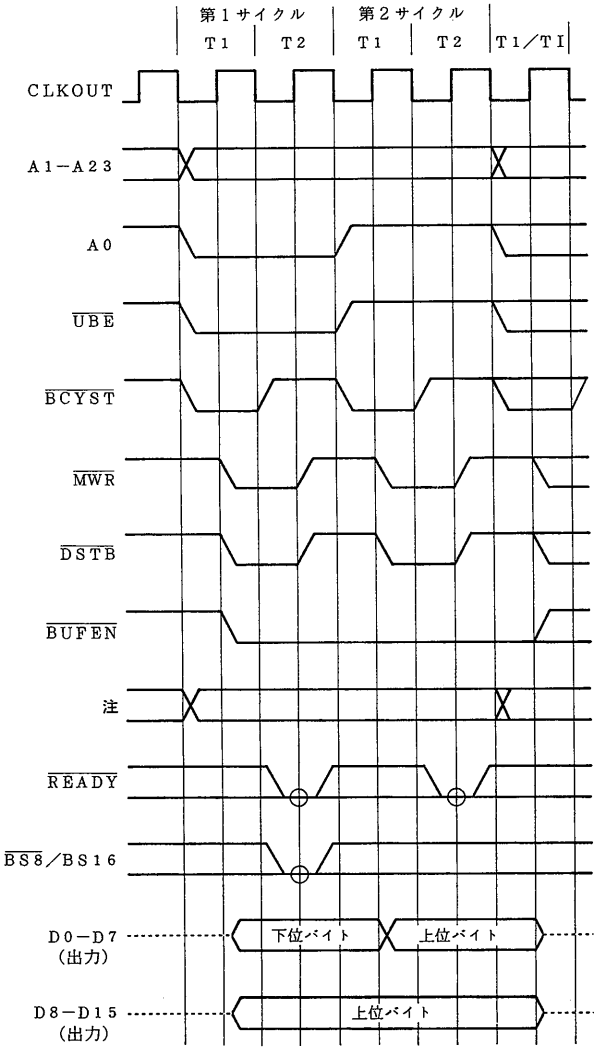
表5－5　バス・サイクルごとのバス・サイジングとサンプリングの有無

バス・サイクル	D15－D0端子 有効なバス・サイズ	$\overline{BS8}$／BS16端子 サンプリング
割り込みアクノリッジ・サイクル（スレーブ）	8ビット	－
割り込みアクノリッジ・サイクル（ICU）	8ビット	－
外部I／Oリード・サイクル	8／16ビット	○
内部I／Oリード・サイクル	8／16ビット	－
外部I／Oライト・サイクル	8／16ビット	○
内部I／Oライト・サイクル	8／16ビット	－
コプロセッサ・リード・サイクル	16ビット	－
コプロセッサ・ライト・サイクル	16ビット	－
ホールト・アクノリッジ・サイクル	Hi－Z（意味なし）	－
命令フェッチ・サイクル	8／16ビット	○
リフレッシュ・サイクル	Hi－Z（意味なし）	－
CPUメモリ・リード・サイクル	8／16ビット	○
DMAリード転送サイクル	Hi－Z（転送するI／Oに依存）	－
CPUメモリ・ライト・サイクル	8／16ビット	○
DMAライト転送サイクル	Hi－Z（転送するI／Oに依存）	－
コプロセッサ用メモリ・リード・サイクル	16ビット	○注
コプロセッサ用メモリ・ライト・サイクル	16ビット	○注
DMAカスケード	Hi－Z（意味なし）	－

注　8ビット・バス・サイズに指定した場合は，正常に動作しません．

図5－6　ダイナミック・バス・サイジング（メモリ・アクセス時）（1／4）

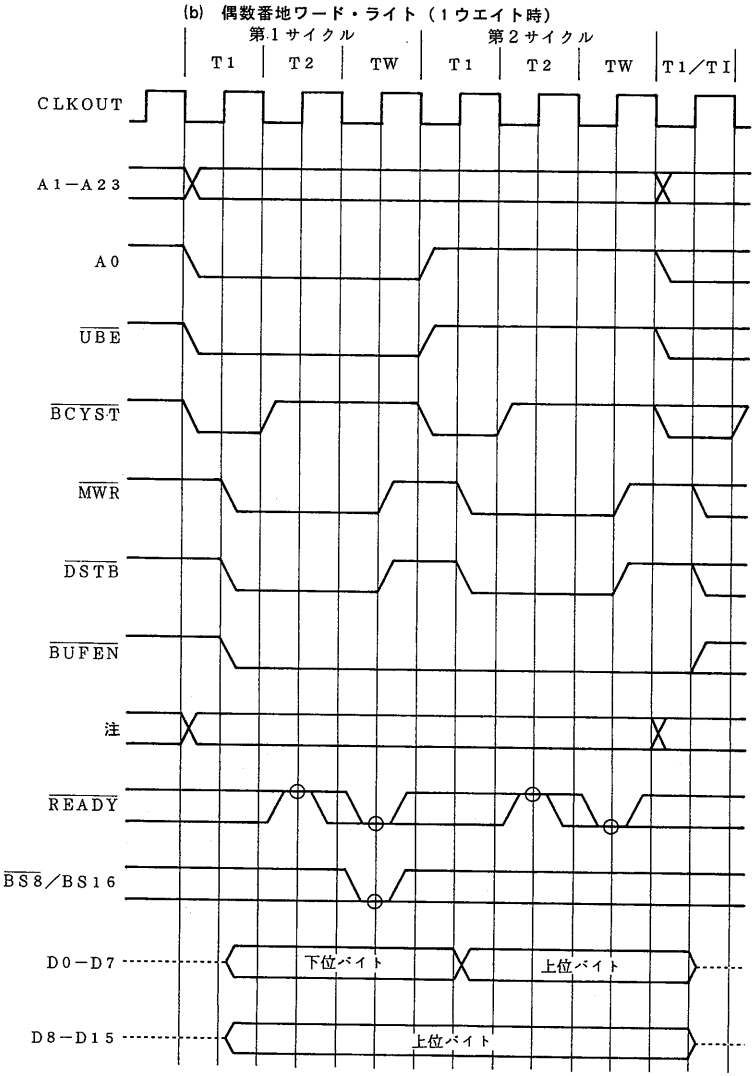
(a)　偶数番地ワード・ライト（ウエイトなし）



注　R／W，M／IO，BUSST0－BUSST2，AEX

備考　○印はWCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．　WCUとREADY端子の関係は，**6.2.3 WCUとREADY端子の関係**を参照してください．

図5－6　ダイナミック・バス・サイジング（メモリ・アクセス時）（2／4）

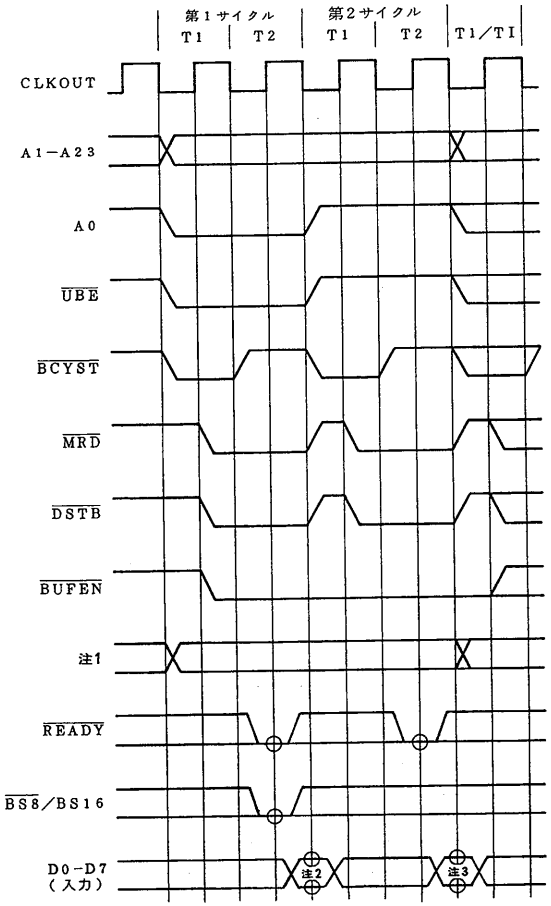


注　R／W, M／IO, BUSST0－BUSST2, AEX

備考　○印はWCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．WCUとREADY端子の関係は，6.2.3 WCUとREADY端子の関係を参照してください．

図5－6　ダイナミック・バス・サイジング（メモリ・アクセス時）（3／4）

(c)　偶数番地ワード・リード（ウエイトなし）



注 1.　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，AEX

2.　下位バイト

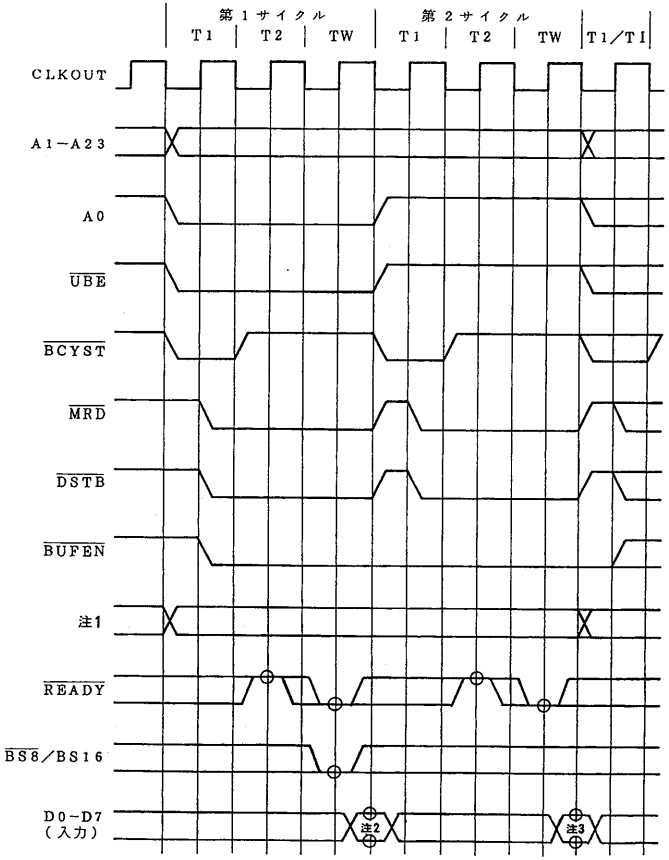
3.　上位バイト

備考　○印はWCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUと$\overline{READY}$端子の関係は，6.2.3　WCUと$\overline{READY}$端子の関係を参照してください．

図5－6　ダイナミック・バス・サイジング（メモリ・アクセス時）（4／4）

(d)　偶数番地ワード・リード（1ウエイト時）



注 1.　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，AEX

2.　下位バイト

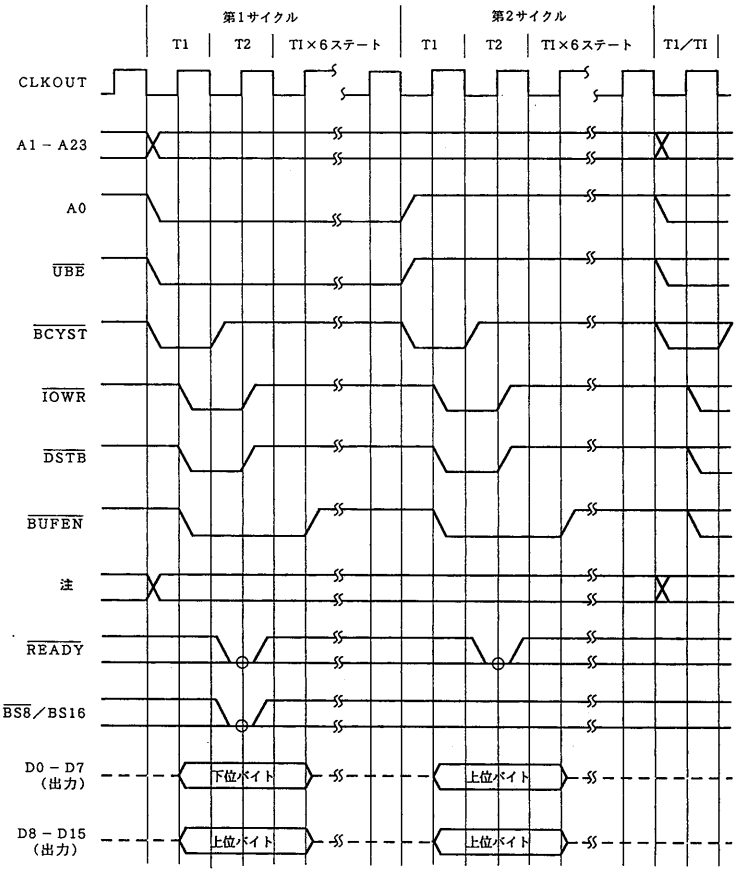
3.　上位バイト

備考　○印はWCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUと$\overline{READY}$端子の関係は，6.2.3　WCUと$\overline{READY}$端子の関係を参照してください．

図5－7　ダイナミック・バス・サイジング（外部I／Oアクセス時）（1／4）

(a)　偶数番地ワード・ライト（ウエイトなし）



注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

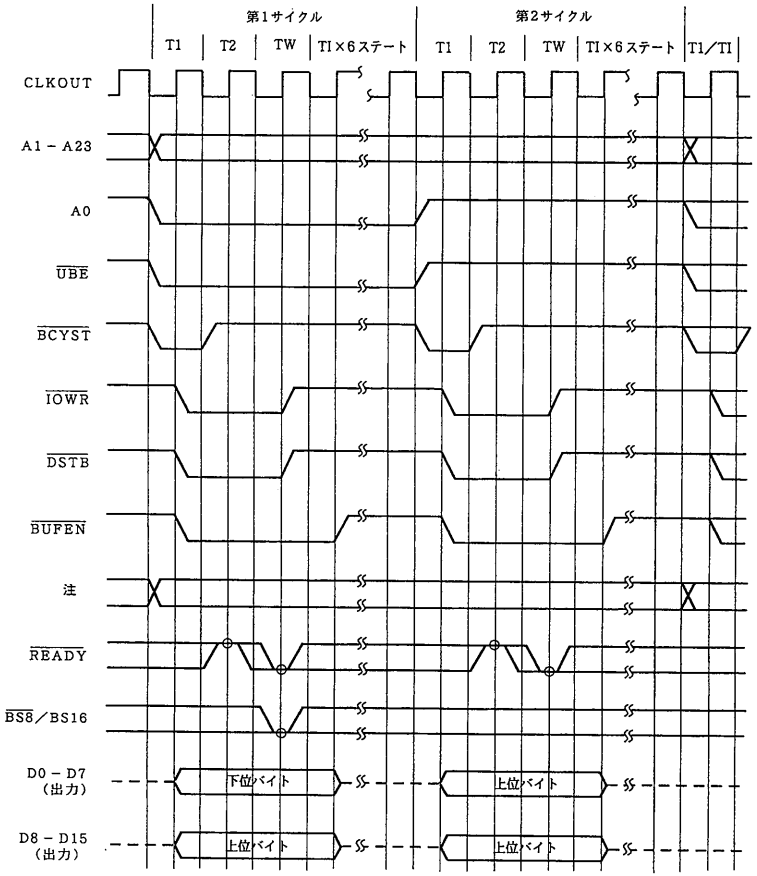
備考 1.　○印はWCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUと$\overline{READY}$端子の関係は，6.2.3　WCUと$\overline{READY}$端子の関係を参照してください．

2.　破線はハイ・インピーダンスを示します．

図5－7　ダイナミック・バス・サイジング（外部I／Oアクセス時）（2／4）

(b) 偶数番地ワード・ライト（1ウエイト時）



注　R／W，M／IO，BUSST0－BUSST2，UBE，AEX

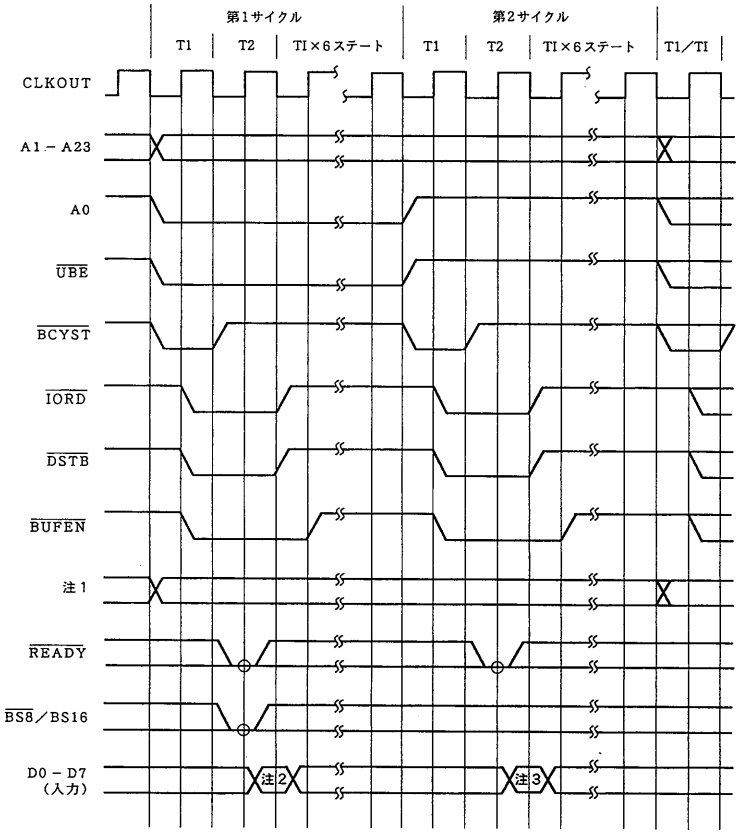
備考1.　○印はWCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUとREADY端子の関係は，6.2.3　WCUとREADY端子の関係を参照してください．

2.　破線はハイ・インピーダンスを示します．

図5－7　ダイナミック・バス・サイジング（外部I／Oアクセス時）（3／4）

(c) 偶数番地ワード・リード（ウエイトなし）



注 1. R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

2. 下位バイト

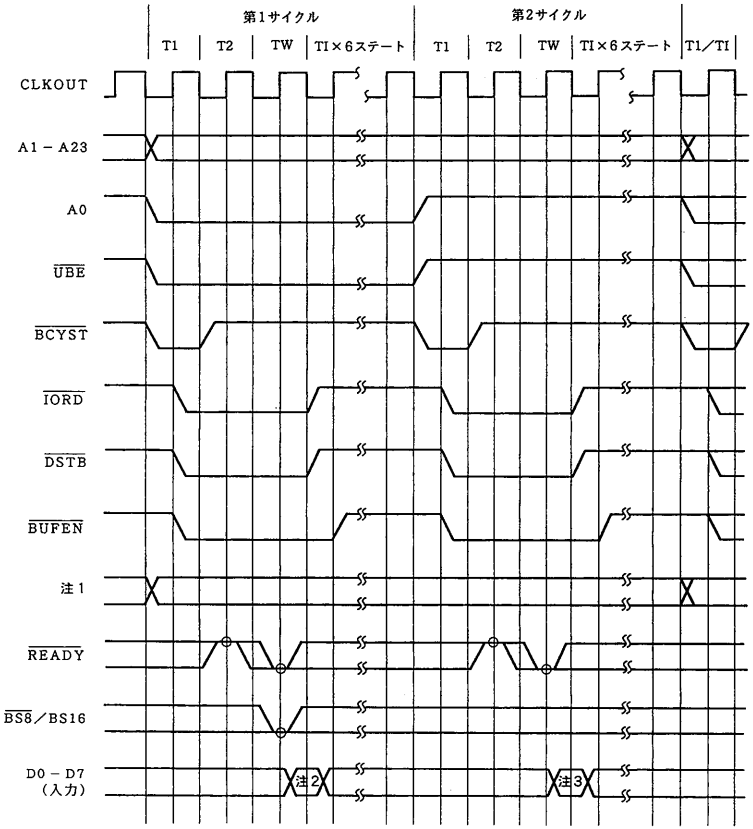
3. 上位バイト

備考　○印にWCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUと$\overline{READY}$端子の関係は，6.2.3　WCUと$\overline{READY}$端子の関係を参照してください．

図5－7　ダイナミック・バス・サイジング（外部I／Oアクセス時）（4／4）

(d)　偶数番地ワード・リード（1ウエイト時）



注1.　R／W，M／IO，BUSST0－BUSST2，UBE，AEX

2.　下位バイト

3.　上位バイト

備考　○印はWCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUとREADY端子の関係は，6.2.3　WCUとREADY端子の関係を参照してください．

5.5　コプロセッサ・インタフェース機能

V53には，浮動小数点演算を実行する専用コプロセッサとしてμPD72291が用意されており，きわめて容易に接続できます．

5.5.1　コプロセッサの識別

V53は，リセット時に$\overline{\text{CPBUSY}}$端子をサンプリングして，μPD72291が接続されているか否かを以下のように判断します．

表5－6　コプロセッサ接続の認識

リセット時の$\overline{\text{CPBUSY}}$状態	認　　識	FPO1，FPO2命令の処理
0	コプロセッサ不在	コプロセッサ不在例外を発生
1	μPD72291接続	実　　行

(1)　**μPD72291が接続されている場合**

V53は，FPO1またはFPO2命令をフェッチすると，その命令の第1バイトと第2バイトをμPD72291に書き込みます．そして，μPD72291に演算処理をまかせ，必要に応じてμPD72291に対する補助的な処理（実効アドレス計算，物理アドレスの生成，およびメモリ・リード／ライト・サイクルの起動）のみを行います．

FPO1命令とFPO2命令は，機能上の差はなくコードの種類が異なるだけです．

また，実際のアセンブリ言語の記述においては，FPO1やFPO2というニモニックを使用するよりもμPD72291の持つ各命令に対するニモニックを使用するのが一般的です．

V53は，FPO1命令やFPO2命令をフェッチしたとき，その命令がメモリ・アクセスを要求していればメモリ・リード・サイクルを起動しますが，そのとき読み出されるデータはμPD72291が使用し，V53は一切取り込みません（コプロセッサ用メモリ・リード・サイクル）．

また，μPD72291がメモリ・ライト・サイクルを必要とする場合，V53はメモリ・ライト・サイクルを起動しますが，このときのデータ・バスはμPD72291が駆動し，V53はデータを出力しません（コプロセッサ用メモリ・ライト・サイクル）．

V53は，POLL命令をフェッチすると，μPD72291と同期をとるため$\overline{\text{CPBUSY}}$端子を2クロックごとにサンプリングし，$\overline{\text{CPBUSY}}$端子がハイ・レベルになるまで実行を待ちます．

(2) μPD72291が接続されていない場合

V53は，FPO1，FPO2，POLL命令をフェッチすると，割り込み許可フラグ（ＩＥ）の状態に関係なくベクタ番号１３０の割り込みを発生します．

FPO1，FPO2，POLL命令フェッチ

↓

ベクタ１３０のテーブル・リード

↓

割り込み先の命令コードのフェッチ

↓

PC，PS[注]，PSWの退避

↓

コプロセッサ不在処理ルーチンへ（内容はユーザが作成）

注　FPO1，FPO2，POLL命令の先頭番地

５．５．２　μＰＤ７２２９１インタフェース

μPD72291とV53とのインタフェースについて説明します．

(1) システム構成例

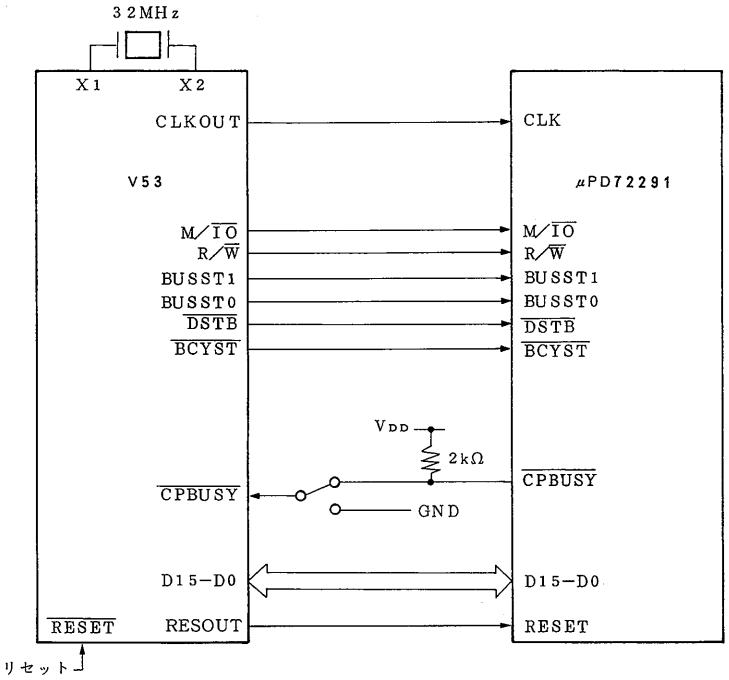
図５－８に，V53とμPD72291とを接続したシステム構成例を示します．μPD72291とV53は，外部回路を追加することなく接続できます．

μPD72291の入出力信号（CLK，RESET，BUSST1，BUSST0，M/$\overline{IO}$，R/$\overline{W}$，$\overline{DSTB}$，D15－D0）は，図５－８に示すように，V53と直結します．μPD72291のCLK端子には，V53のCLKOUT信号を入力してください．

μPD72291の$\overline{CPBUSY}$信号は，V53の$\overline{CPBUSY}$信号に接続し，抵抗（2kΩ程度）でV_{DD}にプルアップしてください．

μPD72291のソケットだけを用意してμPD72291を接続しない場合は，V53の$\overline{CPBUSY}$信号にスイッチを付け，GNDに接続できるようにします．V53は，リセット時に$\overline{CPBUSY}$信号をサンプリングし，$\overline{CPBUSY}$信号がロウ・レベルならば，以後μPD72291用命令を実行しようとするとコプロセッサ不在例外を発生します．

図5－8　V53とμPD72291の接続例



(2)　バス・サイクル

ここでは，V53が発行するコプロセッサ関連のバス・サイクルについて説明します．

V53が発行するバス・サイクルには，メモリやI／Oに対するアクセスのためのバス・サイクルのほかに，コプロセッサに関するバス・サイクルがあります．データがどこからどこへ転送されるかは，バス・ステータス信号群（M／$\overline{IO}$，R／$\overline{W}$，BUSST2，BUSST1，BUSST0）によって示されます．

表5－7にV53のバス・ステータス信号群と，μPD72291に関する4つのバス・サイクルの関係を示します．

表５－７　バス・ステータス信号とコプロセッサ関連のバス・サイクル

M／$\overline{IO}$	R／$\overline{W}$	BUSST2	BUSST1	BUSST0	バス・サイクル	データの転送方向
0	1	0	1	0	コプロセッサ・リード	COP → CPU
0	0				コプロセッサ・ライト	CPU → COP
1	1				コプロセッサ用メモリ・リード	メモリ → COP
1	0				コプロセッサ用メモリ・ライト	COP → メモリ

備考　COPは，コプロセッサの略です．

次に，表５－７に示した各バス・サイクルについて説明します．

(a)　コプロセッサ・リード・サイクル

μPD72291のステータス・ワード・ポート（STWP）から，ステータスを読み出すときに使用します．V53が出力するアドレス（A23－A0）は意味を持ちません（V53は000008Hを出力します）．データ・バスはμPD72291が駆動します．

バス・サイクルは，２クロックです．

図５－９にコプロセッサ・リード・サイクルを示します．

(b)　コプロセッサ・ライト・サイクル

μPD72291のコマンド・ワード・ポート（CMWP）に，命令を書き込むときに使用します．V53が出力するアドレス（A23－A0）は意味を持ちません（V53は000000Hを出力します）．データ・バスはV53が駆動します．

バス・サイクルは，２クロックです．

図５－１０にコプロセッサ・ライト・サイクルを示します．

(c)　コプロセッサ用メモリ・リード・サイクル

メモリ・データをμPD72291のソース・オペランド・ワード・ポート（SOPWP）に転送するときに使用します．V53が出力するアドレス（A23－A0）はメモリ・アドレスです．データ・バスは，メモリが駆動します．

バス・サイクルは，３クロックかかります（V53が自動的にバス・サイクルを１クロック延長します）．

図５－１１にコプロセッサ用メモリ・リード・サイクルを示します．

(d)　コプロセッサ用メモリ・ライト・サイクル

μPD72291のデスティネーション・オペランド・ワード・ポート（DOPWP）から演算結果をメモリに転送するときに使用します．V53が出力するアドレス（A23－A0）はメモリ・アドレスです．データ・バスは，μPD72291が駆動します．

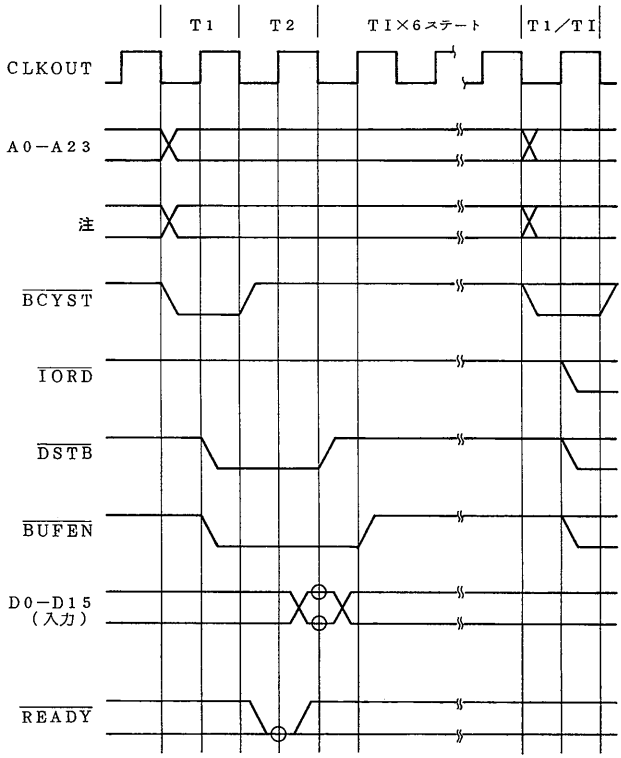
バス・サイクルは，３クロックかかります（V53が自動的にバス・サイクルを１クロック延長します）．

図５－１２にコプロセッサ用メモリ・ライト・サイクルを示します．

備考　CMWP，STWP，SOPWP，DOPWPは，μPD72291内蔵のポートです．

注意　μPD72291用命令を実行する場合，メモリ・オペランドを必ず偶数アドレスに置いてください．また，ダイナミック・バス・サイジングを16ビットに指定してください．これは，μPD72291が1バス・サイクルで16ビット・データをアクセスするため，8ビットにサイジングされたメモリをオペランドに指定すると正常動作できなくなるからです．

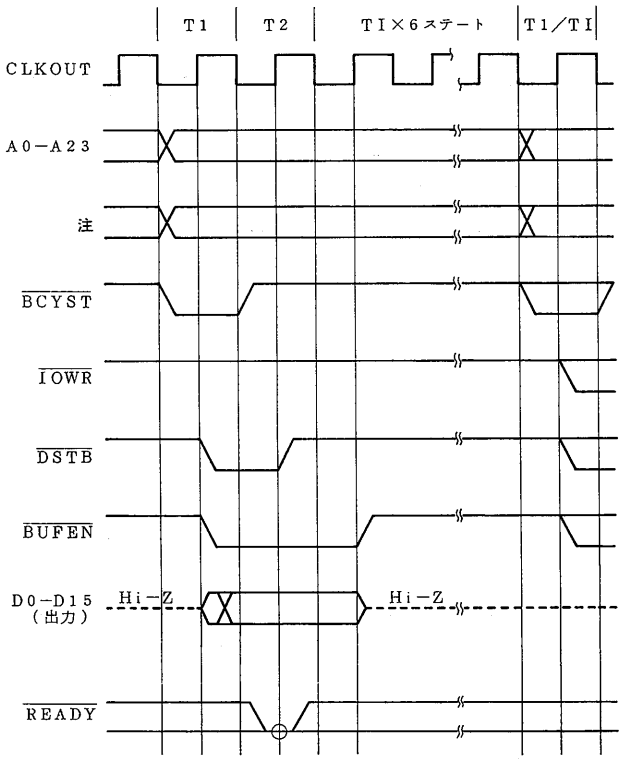
図５－９　コプロセッサ・リード・サイクル



注　R／W，M／IO，BUSST0－BUSST2，UBE，AEX

備考　○印はサンプリングされるタイミングです．

図5－10　コプロセッサ・ライト・サイクル

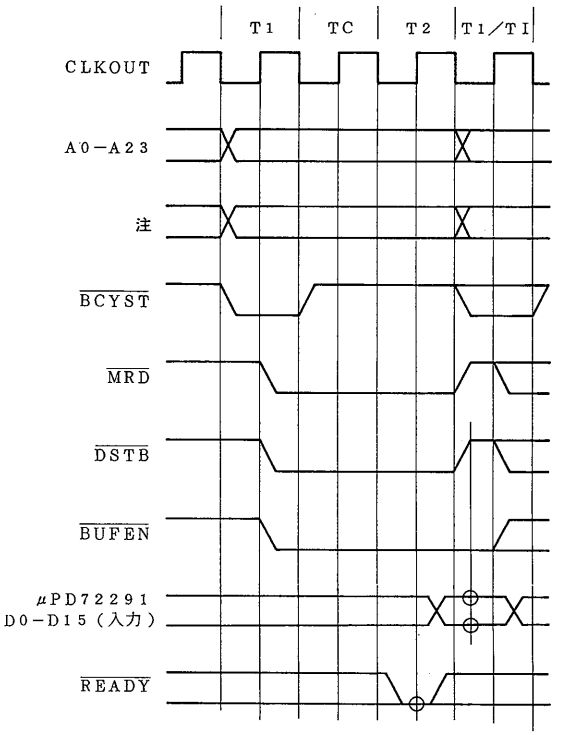


注　R／W，M／IO，BUSST0－BUSST2，UBE，AEX

備考　○印はサンプリングされるタイミングです．

図５－１１　コプロセッサ用メモリ・リード・サイクル（１／２）

(a)　ウエイトなし

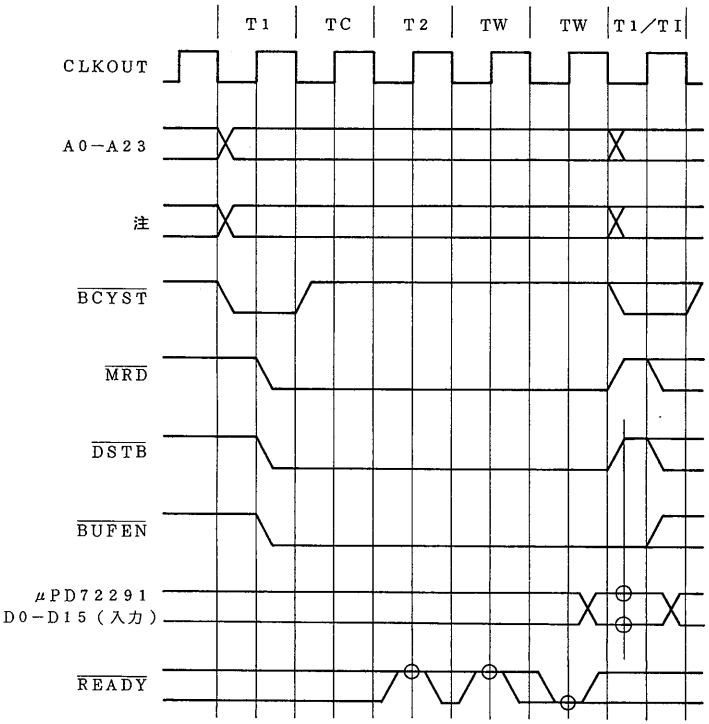


注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

備考　○印はサンプリングされるタイミングです．

図5－11　コプロセッサ用メモリ・リード・サイクル（2／2）

(b)　2ウエイト時

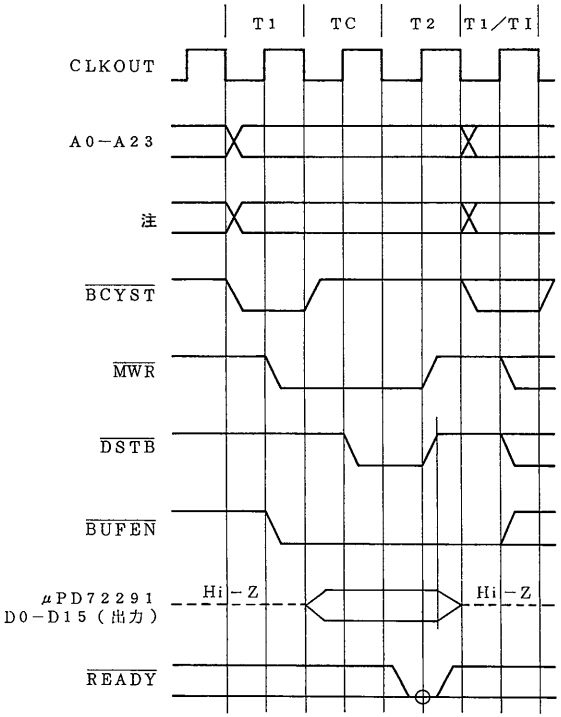


注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

備考　○印はサンプリングされるタイミングです．

図５－１２　コプロセッサ用メモリ・ライト・サイクル（１／２）

(a)　ウエイトなし

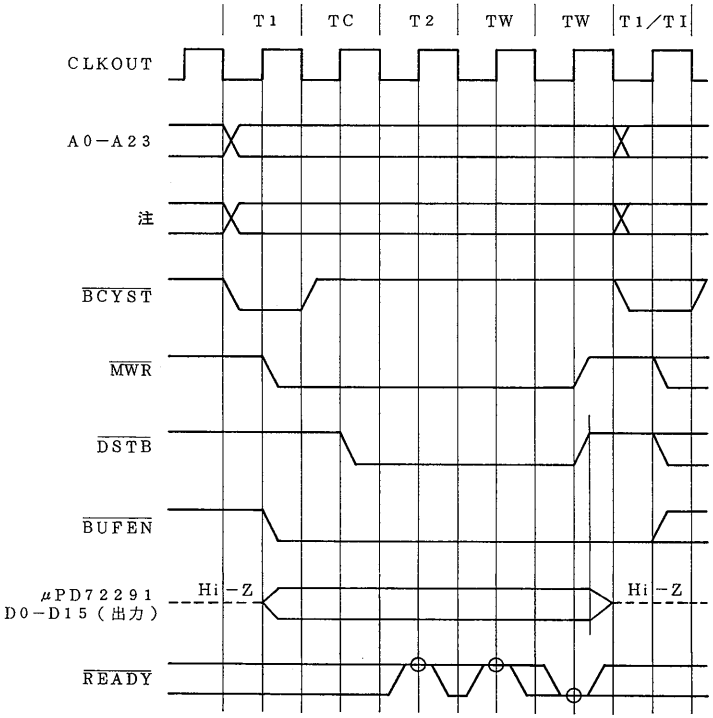


注　R／W，M／IO，BUSST0－BUSST2，UBE，AEX

備考　○印はサンプリングされるタイミングです．

図５－１２　コプロセッサ用メモリ・ライト・サイクル（２／２）

(b)　２ウエイト時



注　R／$\overline{W}$，M／$\overline{IO}$，BUSST0－BUSST2，$\overline{UBE}$，AEX

備考　○印はサンプリングされるタイミングです．

5．5．3　ソフトウエア・インタフェース

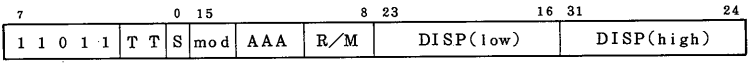
(1)　命令フォーマット

μPD72291用の命令は，V53の命令セットのうちFPO1，FPO2と呼ぶコプロセッサ用命令フォーマットを用います．FPO1は命令の第1バイトがD8H−DFH，FPO2は第1バイトが66H−67Hです．

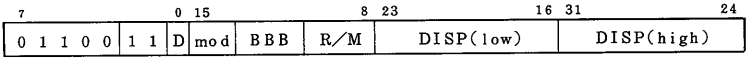
図5−13に，μPD72291の命令フォーマットを示します．

図5−13　μPD72291の命令フォーマット

(a)　命令フォーマット：FPO1



(b)　命令フォーマット：FPO2



各命令フォーマットの詳細については「μPD72291ユーザーズ・マニュアル」を参照してください．

(2)　コプロセッサ・プロトコル

以下にμPD72291で命令を実行させるときの手順（コプロセッサ・プロトコル）を示します．

コプロセッサ・プロトコルは，V53が自動的に実行するものであり，ユーザはコプロセッサ・プロトコルを特に意識する必要はありません．

μPD72291の命令は，以下の4種のコプロセッサ・プロトコルで実行されます．

① 並行動作型　：　オペランドがレジスタだけの命令，または，
　　メモリ・オペランド（32または64ビット）をリードする命令
② 比較命令型　：　比較命令 … FCMP，FCMPE，FCMPA，FCMPAE
③ メモリ・ライト型：　メモリ・オペランド（32または64ビット）にライトする命令
④ 特殊転送型　：　96ビット・オペランドを，リードまたはライトする命令
　　… FMOVRT

以下，各種コプロセッサ・プロトコルについて説明します．

① 並行動作型プロトコル

オペランドがレジスタだけの命令，オペランドのない命令，またはメモリ・オペランド（32または64ビット）をリードする命令で実行されるコプロセッサ・プロトコルです．ただし，比較命令はここには含みません．

（例： FADD FR0, FR3 , FMOV FR0, mem , FINIT ,
FMOV FR3, FR0 ）

並行動作とは，μPD72291が演算実行中にV53が次の命令を実行することを言います．

並行動作型プロトコルは，以下のように実行されます．

(a) V53 ：ソース・オペランドがメモリならば，メモリ・データをSOPWPに転送します．
ソース・オペランドがレジスタ（FRn）ならば，何もしません．

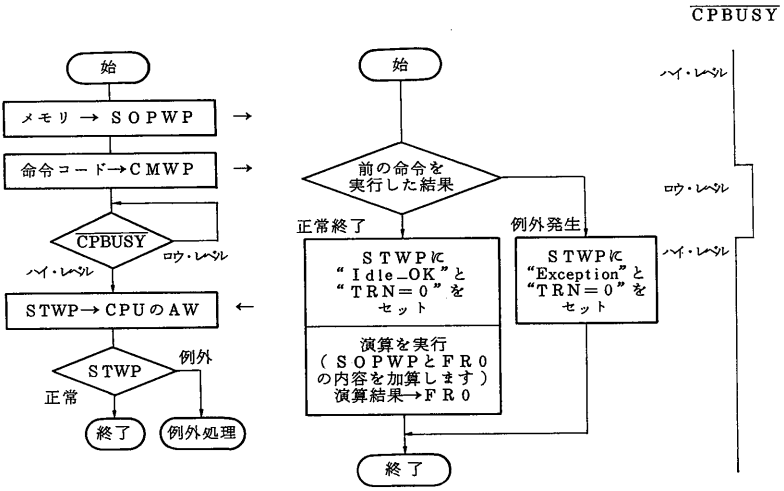
(b) V53 ：μPD72291の命令コードをCMWPに書き込みます．
μPD72291 ：$\overline{\text{CPBUSY}}$信号をロウ・レベルにします．

(c) V53 ：$\overline{\text{CPBUSY}}$信号がハイ・レベルになるまで待ちます．
μPD72291 ：(i)前の命令の実行が終了後，$\overline{\text{CPBUSY}}$信号をハイ・レベルにします．
(ii)前の命令を実行した結果，
・正常終了（命令が実行可能の場合）：
STWPに“Idle_OK”と“TRN=0”をセットします．
CMWPに書き込まれた命令を実行します．
・例外発生（以前実行した命令での例外，または，CMWPの値が未定義）：
STWPに“例外”と“TRN=0”をセットします．
CMWPに書き込まれた命令の実行を中止します．

(d) V53 ：(i)前の命令のSTWPを読み出し，V53のAWレジスタに転送します．
(ii)前の命令を実行した結果，
・正常終了（STWPのStatus=“Idle_OK”の場合）：
命令を終了します．
・例外発生（STWPのStatus=“例外”の場合）：
例外処理を行います．

注意 ただし，FMOD，FREM命令では演算終了まで$\overline{\text{CPBUSY}}$信号がハイ・レベルになりません．

図５－１４　並行動作型プロトコルの例（ＦＡＤＤ　ＦＲ０，ｍｅｍ命令の場合）



② 比較演算型プロトコル

比較命令（FCMP，FCMPE，FCMPA，FCMPAE）で実行されるコプロセッサ・プロトコルです．オペランドがレジスタだけのときと，メモリ・オペランド（32または64ビット）をリードするときがあります．

（例： FCMP FR0，FR3 ，FCMPE FR0，mem）

比較命令型プロトコルは，以下のように実行されます．

(a) V53 ：ソース・オペランドがメモリならば，メモリ・データをSOPWPに転送します．

ソース・オペランドがレジスタ（FRn）ならば，何もしません．

(b) V53 ：μPD72291の命令コードをCMWPに書き込みます．

μPD72291 ：$\overline{\text{CPBUSY}}$信号をロウ・レベルにします．

(c) V53 ：$\overline{\text{CPBUSY}}$信号がハイ・レベルになるまで待ちます．

μPD72291 ：(i)前の命令の実行終了後，比較命令を実行します．

(ii)比較命令の実行終了後に，$\overline{\text{CPBUSY}}$信号をハイ・レベルにします．

(iii)比較命令を実行した結果，

・正常終了（比較命令で例外が発生しなかった場合）：

STWPに“Affect_Flag”と“TRN=0”をセットします．

・例外発生（以前実行した命令または比較命令で例外が発生した場合）：

STWPに“例外”と“TRN=0”をセットします．

(d) V53 ：(i)STWPを読み出し，V53のAWレジスタに転送します．

(ii)比較命令を実行した結果，

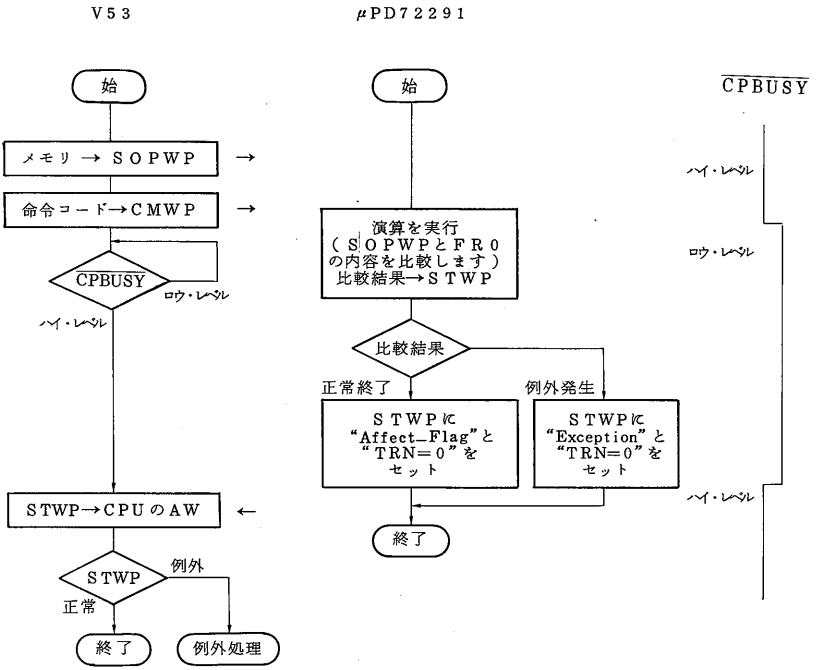
・正常終了（STWPのStatus＝“Affect_Flag”の場合）：

命令を終了します．

・例外発生（STWPのStatus＝“例外”の場合）：

例外処理を行います．

図５－１５　比較命令型プロトコルの例（ＦＣＭＰ　ＦＲ０，ｍｅｍ命令の場合）



③ メモリ・ライト型プロトコル

メモリ・オペランド（３２または６４ビット）にライトする命令で実行されるコプロセッサ・プロトコルです．

（例： FMOV mem, FR0 , FNEG mem, FR0 ）

メモリ・ライト型プロトコルは，以下のように実行されます．

(a) V５３ ：μPD７２２９１の命令コードをCMWPに書き込みます．

μPD７２２９１ ：$\overline{\text{CPBUSY}}$信号をロウ・レベルにします．

(b) V５３ ：$\overline{\text{CPBUSY}}$信号がハイ・レベルになるまで待ちます．

μPD７２２９１ ：(i)前の命令の実行が終了後，CMWPに書き込まれた命令を実行します．

(ii)現在の命令の実行が終了後，$\overline{\text{CPBUSY}}$信号をハイ・レベルにします．

(iii)現在の命令の実行した結果，

・正常終了（例外が発生しなかった場合）：

STWPに“Idle_OK”と“TRN＝１”をセットします．

・例外発生（以前実行した命令または今の命令で例外が発生した場合）：

STWPに“例外”と“TRN＝１またはTRN＝０”をセットします．

(c) V５３ ：(i)STWPを読み出し，V５３のAWレジスタに転送します．

(ii)STWPのTRNの値により以下を実行します．

・TRN＝１： DOPWP→メモリの転送を行います．

・TRN＝０： 何もしません．

(d) V５３ ：現在の命令の実行した結果，

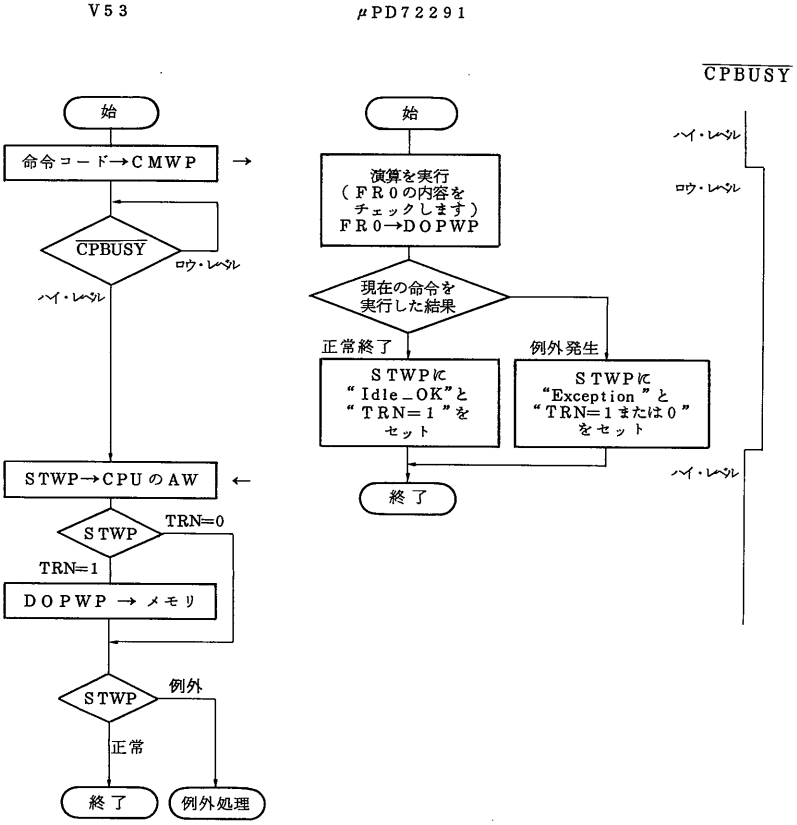
・正常終了（STWPのStatus＝“Idle_OK”の場合）：

命令を終了します．

・例外発生（STWPのStatus＝“例外”の場合）：

例外処理を行います．

図５－１６　メモリ・ライト型プロトコルの例（FMOV　mem，FR0命令の場合）



④ 特殊転送型プロトコル

FMOVRT命令で実行されるコプロセッサ・プロトコルです．メモリ・オペランド（96ビット）をリードまたはライトする命令です．

（例： FMOVRT FR2，mem ，FMOVRT mem，FR7 ）

特殊転送型プロトコルは，以下のように実行されます．

(a) V53 ：μPD72291の命令コードをCMWPに書き込みます．

μPD72291 ：$\overline{\text{CPBUSY}}$信号をロウ・レベルにします．

(b) V53 ：$\overline{\text{CPBUSY}}$信号がハイ・レベルになるまで待ちます．

μPD72291 ：(i)前の命令の実行が終了後，$\overline{\text{CPBUSY}}$信号をハイ・レベルにします．

(ii)前の命令を実行した結果，

・正常終了（命令が実行可能の場合）：

STWPに“Transfer_String”と“TRN=1”をセットします．

CMWPに書き込まれた命令を実行します．

・例外発生（以前実行した命令での例外）：

STWPに“例外”と“TRN=0”をセットします．

CMWPに書き込まれた命令の実行を中止します．

(c) V53 ：(i)STWPを読み出し，V53のAWレジスタに転送します．

(ii)STWPのTRNの値により以下を実行します．

・TRN=1：FMOVRT FRn，mem：メモリ→SOPWPを行います．

FMOVRT mem，FRn：DOPWP→メモリを行います．

・TRN=0：何もしません．

μPD72291 ：STWPのTRNの値により以下を実行します．

・TRN=1：FMOVRT FRn，mem：SOPWP→レジスタを行います．

FMOVRT mem，FRn：レジスタ→DOPWPを行います．

・TRN=0：何もしません．

(d) V53 ：STWPの値によって，以下の動作を行います．

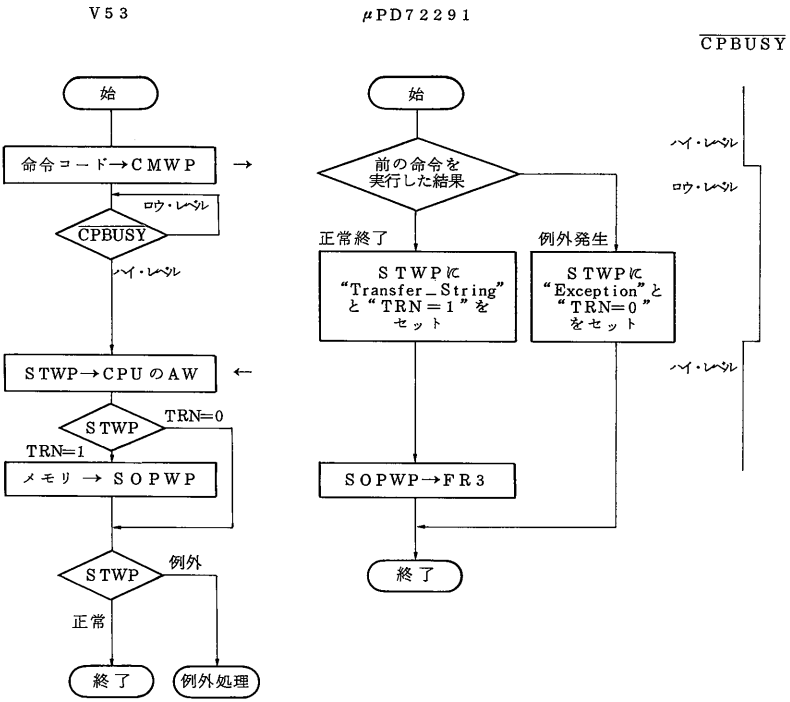
・正常終了（STWPのStatus=“Transfer_String”の場合）：

命令を終了します．

・例外発生（STWPのStatus=“例外”の場合）：

例外処理を行います．

図５－１７　特殊転送型プロトコルの例（FMOVRT　FR3，mem命令の場合）



5．5．4　並行動作

μPD72291命令のプロトコルは，μPD72291での演算が終わらないうちに，V53が次の命令に進むことができます．V53とμPD72291が並行して動作できるため，これを並行動作とよびます．μPD72291は，この並行動作によって高性能を得ています．

並行動作を実現するために，μPD72291では，コマンド・ポート（CMWP）とソース・オペランド・ワード・ポート（SOPWP）を2段バッファにし，2命令分（現在実行中の命令と次に実行予定の命令）の命令コードとデータを保持しています．

並行動作は，並行動作型プロトコルを実行する命令で行われます．

並行動作する命令には，以下のようなものがあります．

① デスティネーション・オペランドがレジスタの命令（比較命令，FMOVRT命令を除く）

例：　FADD　FR0，mem　，FCOS　FR0，FR3　，FMOV　FR4，FR0

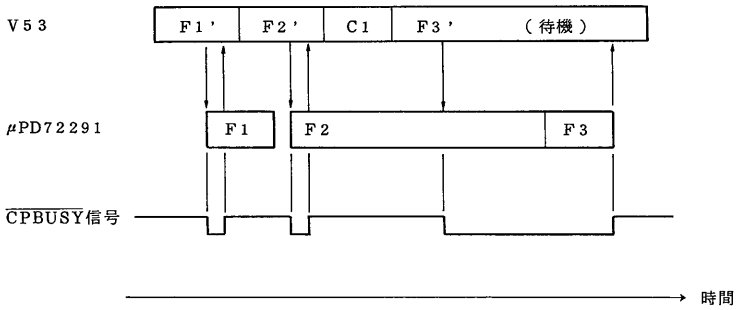
② オペランドがない命令

例：　FINIT　，FRPOP

次に，簡単なプログラム例をあげて，並行動作時のＶ５３とμＰＤ７２２９１の動作を説明します．

```
F1 : FMOV  FR0,mem1 ; FR0  ←mem1        （並行動作型プロトコル）
F2 : FDIV  FR0,mem2 ; FR0  ←FR0/mem2    （並行動作型プロトコル）
C1 : INC   mem3     ; mem3←mem3+1       （ＣＰＵの命令）
F3 : FMOV  mem4,FR0 ; mem4←FR0          （メモリ・ライト型プロトコル）
```

図５－１８　並行動作の実行例



備考　F1, F2, F3　：μＰＤ７２２９１の命令
　　C1　：Ｖ５３の命令
　　F1', F2', F3'　：コプロセッサ・プロトコル
　　↓　：μＰＤ７２２９１の命令→CMWP
　　↑　：ＳＴＷＰ→Ｖ５３

図５－１８において，命令Ｆ１とＦ２が並行動作型プロトコルで実行されます．

命令Ｆ１とＦ２のように，並行動作型プロトコルの命令が連続する場合は，Ｖ５３でのプロトコルの実行とμＰＤ７２２９１での演算がパイプライン的に処理されます．そのため，Ｖ53がμＰＤ７２２９１の演算終了まで待つよりも，高速に処理できます．もし，命令Ｆ１のμＰＤ７２２９１での演算が長い場合は，Ｖ５３はＦ１の演算が終了するまで待たされます．

命令Ｃ１はＶ５３が単独で実行できる命令であり，この期間にμＰＤ７２２９１はＦ２の命令の演算を実行します．このように，Ｖ５３とμＰＤ７２２９１が同時に２命令を実行することによって，処理を高速化できます．

命令Ｆ３は並行動作できない命令です．Ｆ２命令の演算とＦ３命令の演算が終了するまでＶ５３は待たされます．

5.6 割り込み機能

V53では命令の実行が終了するか，割り込み受け付け処理が終了した時点で，割り込み要求の有無がチェックされます．要求が受け付けられればその時点で実行中の命令シーケンスを打ちきり，割り込み受け付け処理を行います．

V53の割り込みは，表5－8に示すように外部割り込み要求によって発生するハードウエア割り込みと，命令の実行によって発生するソフトウエア割り込みに分類できます．

表5－8 割り込みソース一覧

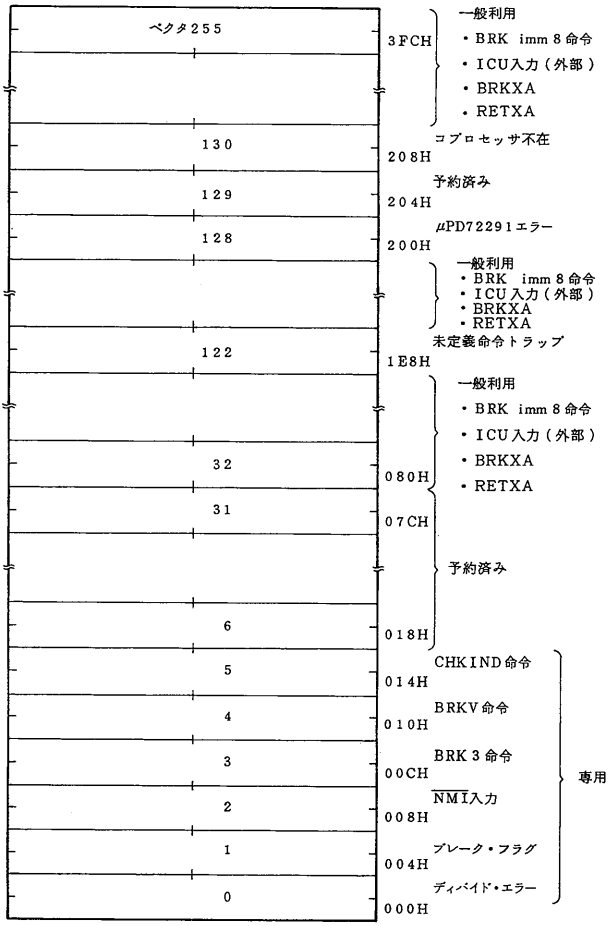
分類	割り込みソース	クロック注	ベクタ番号	優先順位
ハードウエア割り込み	$\overline{\text{NMI}}$	15	2	1
	ICU	32	32－255	2
ソフトウエア割り込み	BRKフラグ（シングル・ステップ）	15	1	3
	DIV／DIVUのディバイド・エラー		0	4
	BRK3		3	
	BRKV		4	
	CHKIND境界オーバ		5	
	未定義命令トラップ		122	
	μPD72291エラー		128	
	（予約済）	－	129	
	コプロセッサ不在例外	15	130	
	BRK imm8		32－255	

注 CPUが割り込み要求を受け付けてから割り込み処理ルーチンに分岐するまでの最小クロック数（ノー・ウエイト，バス・サイズ16ビット時）．

割り込みベクタとは，CPUが2回目の割り込みアクノリッジ・サイクルでICU，または外部の割り込みコントローラから受け取るもので，それをさらに4倍して得られた値が割り込みベクタ・テーブルのアドレスとなります．CPUはそのアドレスのメモリ（割り込みベクタ・テーブル）から分岐先のPC，PSをリードし，割り込み処理ルーチンへ分岐します．

図5－19に割り込みベクタ・テーブルを示します．

図５－１９　割り込みベクタ・テーブル



5．6．1　ハードウエア割り込み

外部からの割り込み要求には，ICU（6.4節参照）が発生するマスカブル割り込み要求と，$\overline{\text{NMI}}$入力によるノンマスカブル割り込み要求の2種類があります．

(1) マスカブル割り込み

ICUがマスカブル割り込み要求を発生した場合のCPUの動作は，PSWの割り込み許可フラグ（IE）の状態により異なります．

IEフラグが“0”にクリアされていると，CPUはその割り込み要求を無視して順次命令を実行し続けます．

IEフラグが“1”にセットされていると，現在実行中の命令を終了したあとで割り込み要求を受け付け，割り込み受け付け処理に移ります．

割り込み受け付け処理では，割り込みベクタをICUまたは外部割り込みコントローラから供給するために割り込みアクノリッジ・サイクルを発行するのがマスカブル割り込みの特徴です．

また，割り込み要求はCPU内部でラッチされないので，CPUが割り込み受け付け処理に入り最初の割り込みアクノリッジ・サイクルを発行するまで保持する必要があります．

(2) ノンマスカブル割り込み

もう一つの外部割り込み要因としての$\overline{\text{NMI}}$入力は，通常電源の急激な変動（瞬時停電），メモリ・エラーやバス・エラーなどシステムの破局的な事象をマイクロプロセッサに伝達するために用いられます．$\overline{\text{NMI}}$入力によって要求される割り込みは，ソフトウエアによるマスクが不可能であるためにノンマスカブル割り込みと呼ばれます．

ノンマスカブル割り込みのベクタは2に固定されているため，割り込みベクタを外部から供給する必要はありません．したがって，ノンマスカブル割り込み受け付け時は，割り込みアクノリッジ・サイクルを発行しません．

5．6．2　ソフトウエア割り込み

ソフトウエア割り込みはCPUの内部の要因で発生するものであり，命令の実行に同期して発生します．BRK，BRKV命令などの割り込み命令によって発生するもの，DIV，DIVU，CHKIND命令の実行時にエラーを検出して発生するもの，ブレーク・フラグ（BRK）の指定によって発生するもの（シングルステップ割り込み），およびコプロセッサ用命令（FPO1，FPO2）の実行によって発生するものの4種類に分類できます．

いずれの場合も，ノンマスカブル割り込みと同様に割り込み許可フラグ（IE）によって割り込みの受け付けを禁止することはできません．また，ノンマスカブル割り込みと同時に，割り込みアクノリッジ・サイクルは発行されません．

ソフトウエア割り込みの特徴をまとめると次のようになります．

- 割り込みベクタは，命令コード中に含まれている（BRK命令）か，あらかじめ決められています．
- 割り込みが受け付けられても，割り込みアクノリッジ・サイクルを発行しません．
- ブレーク割り込みを除いて，割り込みの受け付けをマスクすることはできません．

5．6．3　割り込み受け付け処理

割り込みを受け付けると，割り込みベクタの引き取りかたに違いがあるほかは，CPUは以下のようなシーケンスで共通の受け付け処理を行います．

・割り込みベクタを得る．

↓

・レジスタ類（PC，PSおよびPSW）をスタックに退避させる．

↓

・マスカブル割り込みおよびシングル・ステップ割り込みを禁止する（PSW中のIE，BRKフラグをクリア）．

↓

・割り込みベクタ・テーブルから，割り込みタイプに応じた割り込み処理ルーチンの先頭アドレスを得る．

↓

・割り込み要求のチェック後，割り込み処理ルーチンに制御を移す（PSおよびPCのセット）．

以下，各処理の動作をシーケンスごとに具体的に述べます．

これら(1)から(5)の動作は，特にプログラムの実行と断らないかぎりはハードウエアが自動的に行います．

(1)　割り込みベクタの取得

割り込みベクタは，0－255の値を取ります．マスカブル割り込みとその他の割り込みでは取得方法が異なります．

マスカブル割り込みでは，割り込みを受け付けると2回の割り込みアクノリッジ・サイクルが発行されます．1回目のサイクルは，内蔵ICU，または外部に接続された割り込みコントロール・ユニットに対して，マスカブル割り込みが発生したことを通知するために発行されるものです．2回目のサイクルで，CPUは8ビットの割り込みベクタを内蔵ICUまたはデータ・バスからリードします．

マスカブル割り込み以外では，割り込みベクタは命令コードの中に含まれているか，割り込みの種類によってあらかじめ決まっており，割り込みベクタ取得のための割り込みアクノリッジ・サイクルは発行されません．

(2)　レジスタの退避

割り込み処理ルーチンの実行が終了したときに割り込みを受け付けた時点の状態を完全に復元できるように，PSW，PS，PCをスタックに順番に退避します．これら以外のレジスタ類は，必要に応じて割り込み処理ルーチンの先頭でプログラムの実行によって退避させる必要があります．退避されたレジスタ類（PSW，PS，PC）は，割り込み処理ルーチンの最後で，RETI命令を実行することにより元の状態に戻ります．プログラムで退避させたレジスタは，RETI命令の実行前にプログラムで復帰させてください．

(3) IEフラグとBRKフラグのクリア

割り込み要求は，割り込み受け付け処理が終了する際に再度チェックされます．したがって，割り込み受け付け処理の終了までに同一の割り込み要求が解除されない場合，この割り込み要求は永久に受け付け続けられてしまいます．この不具合を避けるため，割り込みを受け付けるとPSW中のIEフラグとBRKフラグが“0”にクリアされて，マスカブル割り込みとブレーク割り込みの受け付けは禁止されます．

(4) 割り込みベクタ・テーブルのアクセス

割り込み処理ルーチンのアドレスは，割り込みベクタによって決定される割り込みベクタ・テーブルのエントリに格納されています．各エントリは，図5－20に示すように16ビットのオフセット・アドレスと16ビットのベース・アドレスからなる4バイトで構成されます．オフセット・アドレスはPCに，ベース・アドレスはPSにセットされます．

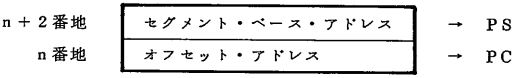
割り込みベクタ・テーブルは，図5－19に示すようにメモリ空間において最下位の1Kバイトに配置されています．nである割り込みベクタのエントリは，（4×n）番地から始まります．

エントリ0－5はノンマスカブル割り込みおよびソフトウエア割り込み専用であり，またエントリ6－31は将来のために予約されているのでユーザが直接使用することはできません．

CPUは，エントリの格納アドレスを計算したあと，オフセット・アドレスおよびベース・アドレスをワード・データとしてリードします．

ユーザが割り込みを用いる場合は，あらかじめ割り込みベクタ・テーブルを初期化してください．

図5－20　割り込みベクタ・テーブル・エントリ



(5) 割り込み処理ルーチンの起動

割り込みベクタ・テーブルから得たオフセット・アドレスをPCに，ベース・アドレスをPSに転送することで，割り込み処理プロシージャに制御を移します．

このとき，命令の実行終了時と同様に，再度割り込み要求の有無がチェックされます．

注意1.　ソフトウエア割り込みのうち，次の要因で発生するものは，割り込みのかかった命令の先頭番地を示すPC，PSの値を，スタックに退避します．

- **ディバイド・エラー割り込み**
- **CHKIND境界エラー割り込み**
- **未定義命令コード・トラップ**
- **コプロセッサ不在割り込み**
- **μPD72291エラー割り込み**

上記以外の要因で発生する割り込みでは，割り込みのかかった命令の次の命令の

先頭番地を示すPC，PSの値を，スタックに退避します．

2. NMI処理ルーチン中は，RETI命令を実行するまで，ほかのNMI割り込みを受け付けません．ただし，NMI処理ルーチンにおいて，HALT命令を実行し，スタンバイ・モードとなっている状態では，NMI割り込みを受け付けます．この場合，スタンバイ・モードを解除し，NMI割り込み（ベクタ2）を発生します．

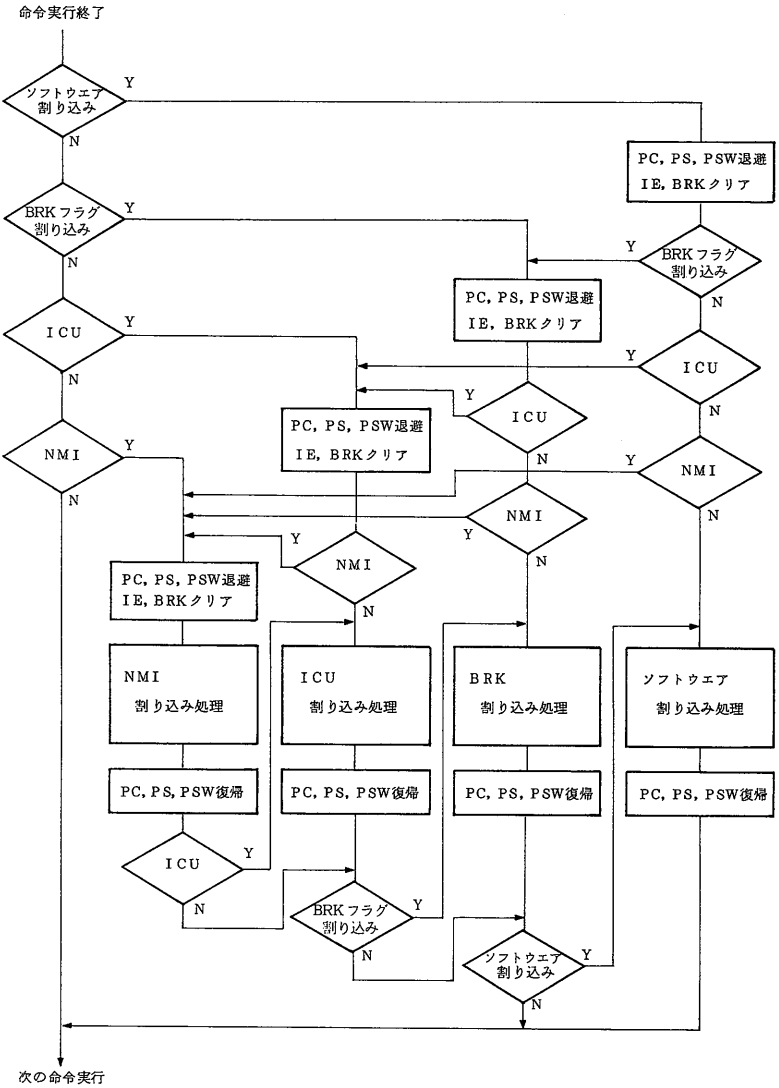
5．6．4　割り込みの優先順位

V53における割り込みの優先順位は，以下のような4つのレベルに分類されます．

(1) $\overline{\text{NMI}}$入力
(2) ICU入力
(3) BRK（ブレーク・フラグ）
(4) ソフトウエア割り込み
- 未定義命令トラップ
- BRK3命令
- BRK　imm8　命令
- ディバイド・エラー
- BRKV命令オーバフロー
- CHKIND境界オーバ
- μPD72291エラー
- コプロセッサ不在例外

割り込み受け付け順位 …… (4)＞(3)＞(2)＞(1)
割り込みプロシージャ処理順位…… (1)＞(2)＞(3)＞(4)

図５－２１　割り込み処理シーケンス



5．6．5　BRKフラグ（シングルステップ）割り込み

V53には，プログラムのディバグなどに有効なシングルステップ割り込み機能が用意されています．この割り込みは，PSWのビット8のBRKフラグによって制御されます．ただし，BRKフラグを直接セット，リセットする命令はなく，PSWをスタックに退避した状態で操作し，それをPSWに復帰することによって間接的にセット，リセット処理を行います．

BRKフラグがセットされると，次の1命令を実行したあと，ベクタ1で指定される割り込みルーチン（モニタ・プログラムなど）が起動され，そのとき割り込み許可フラグ（IE）といっしょにBRKフラグもリセットされます．

したがって，いったんベクタ1割り込みが起動されると，割り込みルーチンの命令は1つずつではなく，ほかの割り込みと同様に連続して実行されます．ここで内部レジスタやフラグの状態，メモリの内容等をチェックしたり，ダンプしたりすることができます．

この割り込みルーチンの中で，シングルステップの回数などをチェックし，シングルステップ動作を終了しても良ければ，メモリ操作命令でスタック内のBRKフラグをリセットしてリターンします．こうするとメイン・ルーチンに戻ったあと，連続して命令が実行されるようになります．

BRKフラグを操作せずに戻れば，スタックに退避されていたBRK＝1がPSWにリストアされ，メイン・ルーチンの1命令を実行したあと，再びベクタ1割り込みを起こします．

5．6．6　割り込みが受け付けられないタイミング

次のタイミングでは，割り込みは受け付けられません．

表5－9　割り込み受け付け不可タイミング

割り込みを受け付けないタイミング	受け付けられない割り込み
MOV reg, sreg／MOV sreg, reg／MOV mem, sreg MOV sreg, mem／POP sreg の各命令とその次の命令の間	ICU, $\overline{\text{NMI}}$, ブレーク・フラグ
セグメント・オーバライド・プリフィクス命令（PS:, SS:, DS0:, DS1:）とその次の命令の間	ICU, $\overline{\text{NMI}}$, ブレーク・フラグ
リピート・プリフィクス命令（REP, RENZ, REPC, REPNC）とその次の命令の間	ICU, $\overline{\text{NMI}}$, ブレーク・フラグ
バス・ロック・プリフィクス命令（BUSLOCK）とその次の命令の間	ICU, $\overline{\text{NMI}}$, ブレーク・フラグ
（IE＝0状態 → EI命令実行 → IE＝1状態）における，EI命令とその次の命令の間	ICU
（IE＝0状態 → POP PSW命令実行 → IE＝1状態）における．POP PSW命令とその次の命令の間	ICU
（IE＝0状態 → RETI命令実行 → IE＝1状態）における，RETI命令とその次の命令の間	ICU

5．6．7　ブロック処理命令実行中の割り込み処理

プリミティブ・ブロック転送／比較，入出力命令実行途中で外部割り込み（NMIまたはEI状態でのICU）が発生した場合，CPUはそれを受け付け，該当の割り込みアドレスにブランチします．こうして起動された割り込み処理ルーチンの先頭で，ブロック・データのカウンタとして働いているCWレジスタをスタックに退避しておき，割り込み処理ルーチンの終わりでCWを回復させたあと，RETI命令で元のルーチンに戻れば，中断していたブロック処理を再開することができます．

このときブロック処理命令の前にプリフィクスが置かれていた場合，3種までそれらを記憶し，割り込み処理ルーチンから戻ったときにプリフィクスの置かれているアドレスに戻れるように，リターン・アドレスを修正（1種類のプリフィクスに対して－1番地）して退避させるようになっています．

これらの機能を有効に利用するには，あるブロック処理命令の前に置かれるプリフィクスの合計が3個を越えないようにする必要があります．

〔良い例〕

```
        BUSLOCK
        REPC
NMI → CMPBKB  SS：src－block，dst－block
```

この例では，NMI割り込み処理から戻ったあと，BUSLOCK，REPC，SS：ともに有効に働きます．

〔悪い例〕

```
        BUSLOCK
        REP
        REPC
NMI → CMPBK  SS：src－block，dst－block
```

この例では，プリフィクスの種類は3（リピート・プリフィクスはすべて同種）として受けとめられ，3番地分のリターン・アドレス修正が行われますが，実際にプリフィクスは4番地を占めていますので，割り込み処理ルーチンからBUSLOCKには戻れず，REPに戻ってしまいます．

５．７　バス・ホールド

V５３は，次のようなバス・ホールド機能を持っています．

HLDRQに外部からハイ・レベルを入力することにより，外部デバイスがバスの開放を要求していることを知らせます．HLDRQがハイ・レベルであることを検出すると，CPUは実行中のバス・サイクルがあるとその終了後に，なければすぐに，A0−A23，$\overline{\text{UBE}}$，D0−D15，R／$\overline{\text{W}}$，M／$\overline{\text{IO}}$，BUSST0−BUSST2，$\overline{\text{BCYST}}$，$\overline{\text{DSTB}}$，$\overline{\text{MRD}}$，$\overline{\text{MWR}}$，$\overline{\text{IORD}}$，$\overline{\text{IOWR}}$，$\overline{\text{BUFEN}}$の各出力をハイ・インピーダンス状態にします．さらに，HLDAKをアクティブにして外部デバイスにバスを開放したことを示し，ホールド状態に移ります．ホールド状態の間は，CPUのバス・アクセス要求などを停止し，バスを使用しない内蔵周辺デバイスのみを動作させます．ホールド状態では，CPUはHLDRQをサンプリングし，ロウ・レベルを検出するとHLDAKをインアクティブにします．これにより，外部デバイスに対しバスを開放していないことを示し，ホールド状態を終了します．

HLDRQのバス使用権の優先順位は次のようになっています．

REFU（最高優先）＞DMAU＞HLDRQ＞CPU＞REFU（最低優先）

バスのアイドル時（TI）や，CPUバス・サイクルおよび最低優先のリフレッシュ・サイクル中に外部デバイスからバス・ホールド要求（HLDRQ）が発生した場合は，バス・サイクル終了後ただちにHLDRQが受け付けられ，バスが開放されます．しかし，バス・ホールド中にDMA要求や，最高優先のリフレッシュ要求が発生した場合，V５３はHLDAKをインアクティブにしてバス使用権の返還を要求します．その場合，外部デバイスはHLDRQをインアクティブにしてバス使用権を返還する必要があります．V５３内部の優先順位の高いバス・マスタは，HLDRQが取り下げられるまで待たされています．この状態をバス待ち動作といいます．ただし，HLDRQがインアクティブになるまで，ほかのバス・マスタがバスを使用することはありません．このように，バス返還要求のためHLDAKを強制的にインアクティブにする場合のHLDAK信号のハイ・レベル幅は，最小の場合で１クロック分となります．

HALTモード（7.5　HALTモード参照）中でも，バス・ホールドの要求は受け付けられます．HLDRQを受け付けると，ただちにHLDAKを発生し，バス・ホールド状態になります．バス・ホールド要求がなくなると，CPUは再びバスの使用権を得ますが，再びHALTモードに戻ります．

CPUがBUSLOCKプリフィクス付きの命令を実行中，および$\overline{\text{BUSLOCK}}$がアクティブの期間は，バス・ホールドの要求は受け付けません．

表5－10　バス・ホールド時の端子状態

端子名	入出力	端子状態	端子名	入出力	端子状態
A23－A0	3ステート出力	Hi－Z	RESOUT	出力	L
D15－D0	3ステート入出力	Hi－Z	X2, X1	入力	－
$\overline{UBE}$	3ステート出力	Hi－Z	CLKOUT	出力	非固定
R/$\overline{W}$	3ステート出力	Hi－Z	PCLKOUT	出力	非固定
M/$\overline{IO}$	3ステート出力	Hi－Z	TCLK	入力	－
BUSST2－BUSST0	3ステート出力	Hi－Z	TCTL0－TCTL2	入力	－
$\overline{BCYST}$	3ステート出力	Hi－Z	TOUT0－TOUT2	出力	非固定
$\overline{DSTB}$	3ステート出力	Hi－Z	INTP0－INTP7	入力	－
$\overline{MRD}$	3ステート出力	Hi－Z	$\overline{INTAK}$	出力	H
$\overline{MWR}$	3ステート出力	Hi－Z	TxD	出力	非固定
$\overline{IORD}$	3ステート出力	Hi－Z	RxD	入力	－
$\overline{IOWR}$	3ステート出力	Hi－Z	RxRDY	出力	非固定
$\overline{BUFEN}$	3ステート出力	Hi－Z	SINT	出力	非固定
$\overline{BUSLOCK}$	出力	注1	$\overline{RTS}$	出力	非固定
$\overline{READY}$	入力	－	$\overline{CTS}$	入力	－
$\overline{BS8}$/$\overline{BS16}$	入力	－	$\overline{DTR}$	出力	非固定
AEX	出力	注2	$\overline{DSR}$	入力	－
$\overline{REFRQ}$	出力	H	$\overline{DMARQ0}$－$\overline{DMARQ3}$	入力	－
HLDRQ	入力	－	$\overline{DMAAK0}$－$\overline{DMAAK3}$	出力	H
HLDAK	出力	H	$\overline{END}$/$\overline{TC}$	入出力	Hi－Z
$\overline{NMI}$	入力	－	V_{DD}	－	－
$\overline{CPBUSY}$	入力	－	GND	－	－
$\overline{RESET}$	入力	－	IC2, IC1	－	－

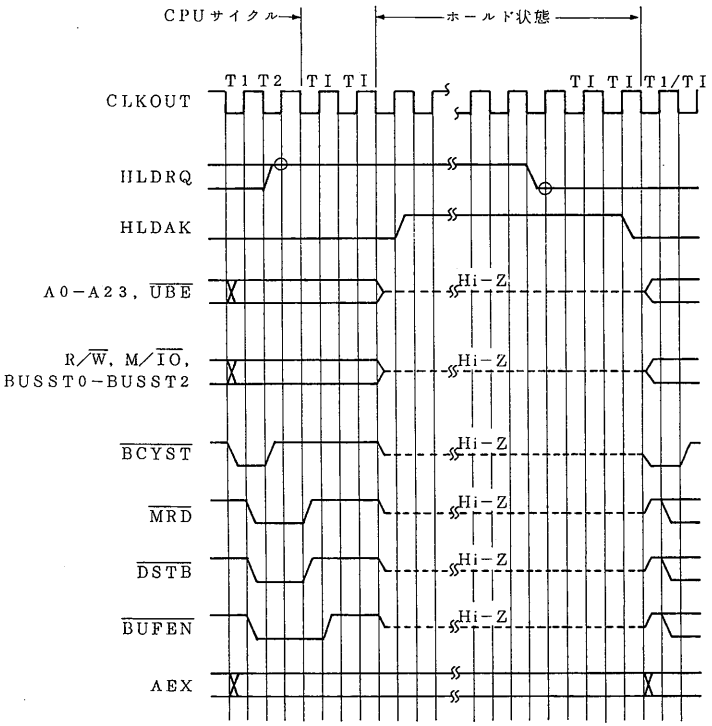
注1.　次のいずれかの場合はL，それ以外はH.

・ホールド時にBUSLOCKプリフィクス付きの命令を実行.

・BUSLOCKプリフィクス付きのHALT命令を実行.

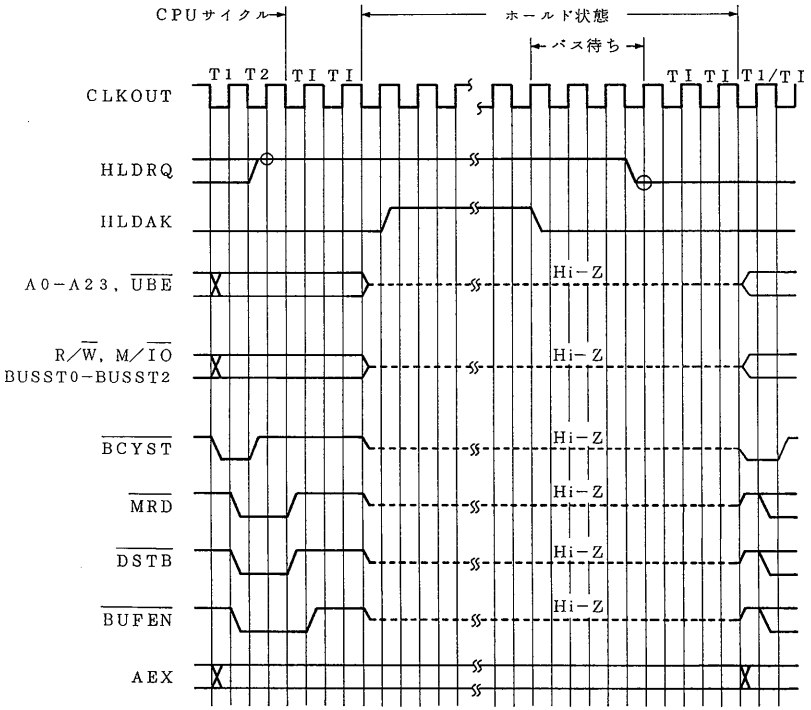
2.　アドレス拡張モード時はH，非拡張モード時はL.

図５－２２　バス・ホールド（通常動作時）　★



備考　○印はサンプリングされるタイミングです．

★ 図5－23　バス・ホールド（バス待ち動作時）



備考　○印はサンプリングされるタイミングです．

5.8 バス・ロック

CPU以外のバス使用者（たとえばDMAUやほかのプロセッサ）に対するバスの使用禁止をするためには，$\overline{\text{BUSLOCK}}$端子をアクティブにしてバス・ロック状態にします．$\overline{\text{BUSLOCK}}$端子は次に示す期間アクティブになります．

- BUSLOCKプリフィクス付きの命令の実行中．
- 割り込みアクノリッジの第1サイクルの開始から第2サイクルの開始まで．

バス・ロック期間中は，CPU以外のいかなるバス要求も受け付けられません．したがって，バス・ロック期間中はホールド要求，DMA要求は無視され，リフレッシュ要求は保留となります．なお，CPUアクセスである命令コードのプリフェッチは行われます．

[メ　モ]

第６章　内蔵周辺デバイス

Ｖ５３は内部Ｉ/Ｏ 領域としてＦＦ００Ｈ－ＦＦＦＦＨ番地を予約しており，システム制御用のＩ/Ｏ が割り付けられています．Ｖ５３を使用するにあたって，まずこれらのＩ/Ｏ を正しく初期化する必要があります．

６．１　システム制御Ｉ／Ｏ

６．１．１　システムＩ／Ｏ領域

システムＩ／Ｏ領域とは６４Ｋバイト（００００Ｈ－ＦＦＦＦＨ）のＩ／Ｏ空間のうち最上位の２５６バイト（ＦＦ００Ｈ－ＦＦＦＦＨ）を占める領域を指します．このシステムＩ／Ｏ領域にはＶ５３の初期化に必要な各種レジスタやＲＥＦＵ，ＷＣＵ内のシステム制御Ｉ／Ｏがマッピングされています．

以下にシステムＩ／Ｏ領域にマッピングされているレジスタと，その主な機能概要を示します．

(1) ＳＣＴＬ（システム・コントロール・レジスタ）
- 内蔵ペリフェラル・リロケーション・アドレスの８ビット・バウンダリ／１６ビット・バウンダリ切り替え機能
- 内蔵ＤＭＡＵをμＰＤ７１０７１モード／μＰＤ７１０３７モードとするかの切り替え機能
- μＰＤ７１０３７モード時にＡ１５からＡ１６へのキャリー伝搬制御機能
- μＰＤ７１０３７モード時にＡ１９からＡ２０へのキャリー伝搬制御機能
- ＳＣＵクロックのボー・レート・ジェネレータ／ＴＯＵＴ切り替え機能

(2) ＯＰＳＥＬ（内蔵ペリフェラル選択レジスタ）
- 内蔵Ｉ／Ｏの使用可／不可設定機能

(3) 内蔵ペリフェラル・リロケーション・レジスタ（ＯＰＨＡ、ＤＵＬＡ、ＩＵＬＡ、ＴＵＬＡ、ＳＵＬＡ）
- 内蔵Ｉ／Ｏのリロケーション・アドレス設定機能

(4) ＷＭＢ０（プログラマブル・ウエイト・メモリ領域設定レジスタ０）
- 自動ウエイト挿入を行う１６Ｍバイト・メモリ空間の３分割機能

(5) ＷＭＢ１（プログラマブル・ウエイト・メモリ領域設定レジスタ１）
- 自動ウエイト挿入を行う１Ｍバイト・メモリ空間３分割機能

(6) ＷＣＹ０（プログラマブル・ウエイト・サイクル数設定レジスタ０）
- １６Ｍバイトの上位メモリ空間アクセスに対する自動挿入ウエイト数設定機能

(7) ＷＣＹ１（プログラマブル・ウエイト・サイクル数設定レジスタ１）
- １６Ｍバイトの中位，下位メモリ空間アクセスに対する自動挿入ウエイト数設定機能

(8) ＷＣＹ２（プログラマブル・ウエイト・サイクル数設定レジスタ２）
- １Ｍバイトの中位，下位メモリ空間アクセスに対する自動挿入ウエイト数設定機能

(9) ＷＣＹ３（プログラマブル・ウエイト・サイクル数設定レジスタ３）
- 外部Ｉ／Ｏサイクルに対する自動挿入ウエイト数設定機能
- １Ｍバイトの上位メモリ空間アクセスに対する自動挿入ウエイト数設定機能

(10) WCY4（プログラマブル・ウエイト・サイクル数設定レジスタ4）

- DMAサイクルに対する自動挿入ウエイト数設定機能
- リフレッシュ・サイクルに対する自動挿入ウエイト数設定機能

(11) WAC（プログラマブル・ウエイト・メモリ・アドレス・コントロール・レジスタ）

- 自動ウエイト挿入を行う1Mバイト・メモリ空間のA23－A20設定機能

(12) RFC（リフレッシュ・コントロール・レジスタ）

- リフレッシュ・サイクルの使用可／不可設定機能
- リフレッシュ・バス幅の設定機能
- リフレッシュ間隔設定機能

(13) SBCR（スタンバイ・コントロール・レジスタ）

- HALT命令実行によるHALTモード／STOPモード選択機能
- STOPモード解除時の発振安定時間設定機能
- 内部クロック分周数設定機能

(14) TCKS（タイマ・クロック選択レジスタ）

- TCUクロックの内部クロック／TCLK選択機能
- TCUクロックの内部クロック分周数設定機能

(15) BADR（バンク・アドレス・レジスタ）

- バンク・レジスタの下位アドレスA7－A2，またはA7－A3，A0を設定

(16) BSEL（バンク選択レジスタ）

- バンク・レジスタの選択チャネルを設定

(17) BRC（ボー・レート・カウンタ）

- SCUのボー・レート・カウンタ

(18) PGR64－PGR1（ページ・レジスタ64－1）

- アドレス拡張用64エントリ・テーブル

(19) XAM（アドレス拡張モード・レジスタ）

- アドレス拡張フラグ

表6－1にシステムI／O領域を示します．

表６－１　システムＩ／Ｏ領域一覧

Ｉ／Ｏアドレス	レジスタ名	操　　作	
FFFFH	予　約	－	
FFFEH	SCTL	リード／ライト	
FFFDH	OPSEL	リード／ライト	
FFFCH	OPHA	リード／ライト	
FFFBH	DULA	リード／ライト	
FFFAH	IULA	リード／ライト	
FFF9H	TULA	リード／ライト	
FFF8H	SULA	リード／ライト	
FFF7H	予　約	－	
FFF6H	WCY4	リード／ライト	6.2 WCU参照
FFF5H	WCY3	リード／ライト	
FFF4H	WCY2	リード／ライト	
FFF3H	WMB1	リード／ライト	
FFF2H	RFC	リード／ライト	6.3 REFU参照
FFF1H	SBCR	リード／ライト	第７章　スタンバイ機能参照
FFF0H	TCKS	リード／ライト	6.6 TCU参照
FFEFH－FFEEH	予　約	－	
FFEDH	WAC	リード／ライト	6.2 WCU参照
FFECH	WCY0	リード／ライト	
FFEBH	WCY1	リード／ライト	
FFEAH	WMB0	リード／ライト	
FFE9H	BRC	リード／ライト	6.7 SCU参照
FFE8H	予　約	－	
FFE7H－FFE2H	予　約	－	
FFE1H	BADR	リード／ライト	6.5 DMAU参照
FFE0H	BSEL	リード／ライト	
FFDFH－FF81H	予　約	－	
FF80H	XAM	リード	第５章　CPU参照
FF7FH－FF00H	PGR64－PGR1	リード／ライト	

注意　このシステムＩ／Ｏ領域へのアクセスをはじめＶ５３の内蔵Ｉ／Ｏへのアクセスは基本的にバイト・タイプのＩＮ／ＯＵＴ命令で行ってください．ただし，μＰＤ７１０７１モード時のＤＭＡＵ内レジスタでワード・アクセスが可能なレジスタがあります．
また，ページ・レジスタへのアクセスはワード・タイプのＩＮ／ＯＵＴ命令で行ってください．

6．1．2　SCTL（システム・コントロール・レジスタ）

SCTLは，以下の機能を有し内蔵ペリフェラルを制御するレジスタです．

(1) **内蔵ペリフェラル・リロケーション・アドレスの8ビット・バウンダリ／16ビット・バウンダリ切り替え機能…IOAG**

μPD70236は内蔵ペリフェラルとしてDMAU，ICU，TCU，SCUの4種のI／Oを有しており，そのI／Oアドレスはリロケーション・レジスタによって設定されます．その際に最下位ビットA0を固定として，16ビット・バウンダリに配置するか，あるいは，最下位ビットを固定せずに，8ビット・バウンダリに配置するかを決定します．

(2) **内蔵DMAUをμPD71071モード／μPD71037モードとするかの切り替え機能…DMAM**

DMAUにはμPD71071モードとμPD71037モードとがあり，どちらのモードで使用するかを決定します．DMAUを使用する前には必ずこのビットを設定する必要があります．

(3) **μPD71037モード時にA15からA16へのキャリー伝搬制御機能…CE0**

DMAUをμPD71037モードで使用する場合に，DMAアドレスのA15からA16へキャリーを伝搬するか，しないかを決定します．

(4) **μPD71037モード時にA19からA20へのキャリー伝搬制御機能…CE1**

DMAUをμPD71037モードで使用する場合に，DMAアドレスのA19からA20へキャリーを伝搬するか，しないかを決定します．

(5) **SCUクロックのボー・レート・ジェネレータ／TOUT1切り替え機能…SC**

SCUの送受信クロックとして，専用のボー・レート・ジェネレータを使用するか，TCUのTOUT1出力を使用するかを決定します．

図6－1　SCTL（システム・コントロール・レジスタ）

I/Oアドレス		7	6	5	4	3	2	1	0
FFFEH	SCTL	－	－	－	SC	CE1	CE0	DMAM	IOAG

IOAG	機　　能
0	内蔵I/Oアドレスを奇数/偶数に固定する（16ビット・バウンダリ）
1	内蔵I/Oアドレスを連続にする（8ビット・バウンダリ）

DMAM	DMAUモード指定
0	DMAUをμPD71071モードにする
1	DMAUをμPD71037モードにする

CE0	機　　能
0	μPD71037モード時にキャリーをA16に伝搬しない
1	μPD71037モード時にキャリーをA16に伝搬する

CE1	機　　能
0	μPD71037モード時にキャリーをA20に伝搬しない
1	μPD71037モード時にキャリーをA20に伝搬する

SC	SCUの入力クロック指定
0	SCUの入力クロックにTOUT1を使用
1	SCUの入力クロックにボー・レート・ジェネレータを使用

6．1．3　OPSEL（内蔵ペリフェラル選択レジスタ）

OPSELは，以下の機能を有するレジスタです．

- 内蔵DMAUの使用可/不可設定機能
- 内蔵ICUの使用可/不可設定機能
- 内蔵TCUの使用可/不可設定機能
- 内蔵SCUの使用可/不可設定機能

使用不可の状態とはそのI/O領域に対するプログラミング不可を意味します．すなわち，各周辺に対するプログラミングを行う前にこのレジスタによって使用可の状態にしておく必要があります．

図6－2　OPSEL（内蔵ペリフェラル選択レジスタ）

I/Oアドレス	7	6	5	4	3	2	1	0
FFFDH OPSEL	－	－	－	－	SS	TS	IS	DS

DS	内蔵DMAU使用指定
0	内蔵DMAU使用不可
1	内蔵DMAU使用可

IS	内蔵ICU使用指定
0	内蔵ICU使用不可
1	内蔵ICU使用可

TS	内蔵TCU使用指定
0	内蔵TCU使用不可
1	内蔵TCU使用可

SS	内蔵SCU使用指定
0	内蔵SCU使用不可
1	内蔵SCU使用可

注意　OPSELを設定する前または読み出す前に内蔵ペリフェラル・リロケーション・レジスタ（OPHA，DULA，IULA，TULA，SULA）を設定してください．

6．1．4　内蔵ペリフェラル・リロケーション・レジスタ（OPHA，DULA，IULA，TULA，SULA）

内蔵ペリフェラル・リロケーション・レジスタにはOPHA，DULA，IULA，TULA，SULAの5つがあり，DMAU，ICU，TCU，SCUの各内蔵ペリフェラルに対するI／Oアドレスを決定します．

OPHAはこれら4つの周辺に共通なレジスタで各I／Oアドレスの上位8ビットを設定します．すなわち，これら4つの周辺はOPHAレジスタで設定された256バイト内のI/O空間に配置されます．次にこの256バイト内のどこに各周辺を配置するかをDULA，IULA，TULA，SULAの下位アドレス・レジスタが設定します．下位アドレスは，前述のSCTL内のIOAGビットの値によって設定可能なビットが変わります．

図6－3に内蔵ペリフェラル・リロケーション・レジスタの詳細を示します．

注意1.　I／O領域はDMAUが16バイト，ICU，TCU，SCUが4バイトを占めますので，それぞれが重ならないようにしてください．

2. 前述したようにDMAU以外へのアクセスはバイト・タイプのIN/OUT命令で行ってください.

3. 内蔵ペリフェラルI/O領域，およびシステムI/O領域以外のI/O空間は，外部I/O領域として使用することができます.

4. OPHAにはFFHを設定しないでください.

図6－3　内蔵ペリフェラル・リロケーション・レジスタ

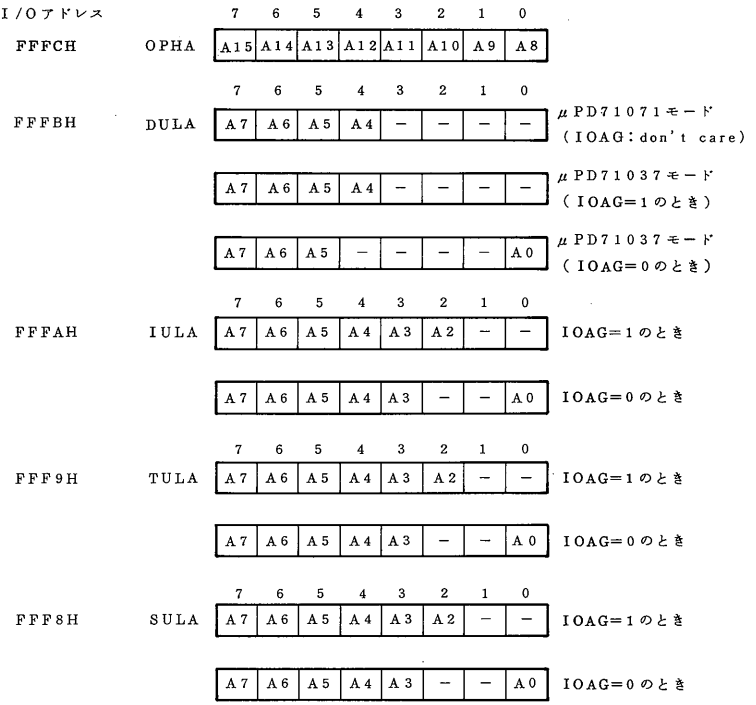
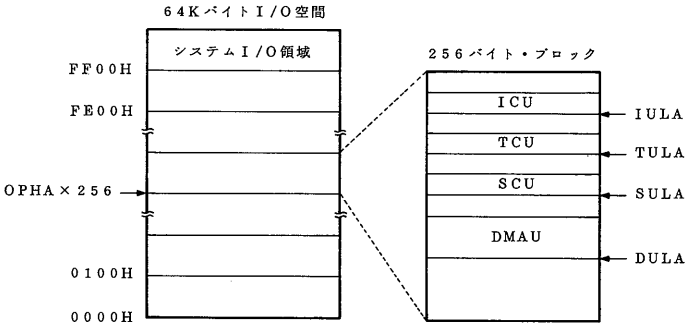


図６－４　内蔵ペリフェラル・リロケーション概念図



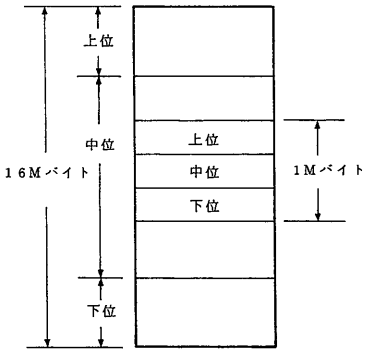
6.2 WCU(ウエイト・コントロール・ユニット)

WCUは，CPU，DMAU，REFUの各バス・サイクルに対し，0－7クロック分のウエイト・ステート（TW）を自動挿入する機能を持っています．通常，低速のメモリやI/Oを接続する場合には$\overline{\text{READY}}$入力を制御することによってウエイト・ステートを挿入し，バス・サイクルを引き延ばします．しかし，0－7クロック以下のTWで十分な場合はこのWCUを設定するだけで済み，外部READY制御回路は不要になります．

6.2.1 特　徴

- CPUメモリ・バス・サイクルに対する0－7ウエイトの自動設定
  - 16Mバイトのメモリ空間を3分割し，それぞれ独立にウエイト指定可能
  - 特定の1Mバイトのメモリ空間を3分割し，それぞれ独立にウエイト指定可能
  - （16Mバイトのメモリ空間に対する設定より優先されます）
- 外部I/Oサイクルに対する0－7ウエイトの自動設定
- DMAサイクルに対する0－7ウエイトの自動設定
- リフレッシュ・サイクルに対する0－7ウエイトの自動設定
- 割り込みアクノリッジ・サイクルに対する2－7ウエイトの自動設定

図6－5　メモリ空間の分割



6．2．2　V50に対する変更箇所

・16Mバイトのメモリ空間に対する3分割ウエイト機能

（1Mバイト・メモリ空間に対する3分割ウエイト機能も有効です．その設定は，16Mバイト・メモリ空間に対する設定よりも優先順位は高くなっています）

・設定可能ウエイト数

V50：0－3ウエイト

V53：0－7ウエイト

・システムI／O領域の変更（6.1　システム制御I／O参照）

V50：WCY1，WCY2，WMB

V53：WCY0，WCY1，WCY2，WCY3，WCY4，WMB0，WMB1，WAC

16Mバイトまでメモリ空間を拡張するため，16Mバイトを3分割してウエイトを設定できるようにしています．

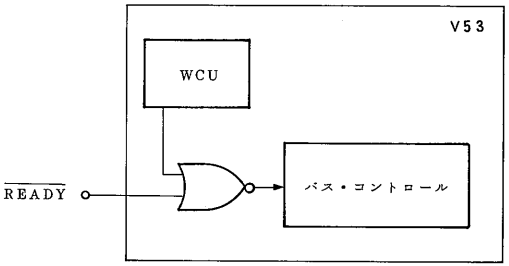
また，アドレス拡張を使用しないアプリケーションのため，あるいはもっと細かくウエイトを設定できるよう，従来の1Mバイトに対する3分割も可能としています．

6．2．3　WCUとREADY端子の関係

0－7クロック以上のウエイト・サイクルが必要な場合には，このWCUとREADY信号を組み合わせて使用することができます．WCUの設定値によるウエイト・サイクルとREADY制御によるウエイト・サイクルは，論理和の形で挿入され，どちらか多い方のウエイト・サイクル数だけ挿入されます．

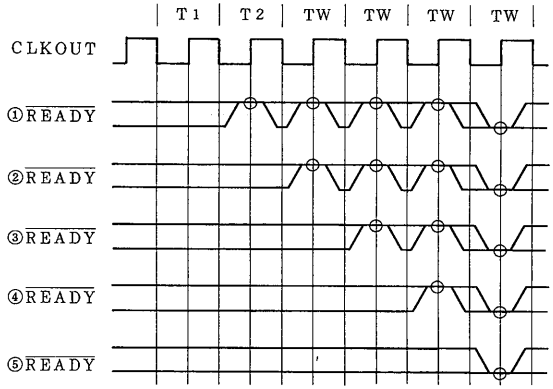
図6－6　WCUとREADY制御

(a)　WCUとREADY端子の関係



備考　WCUおよびREADY端子は，ロウ・レベルでレディ状態です．また，バス・コントロールは，ハイ・レベルでレディ状態です．

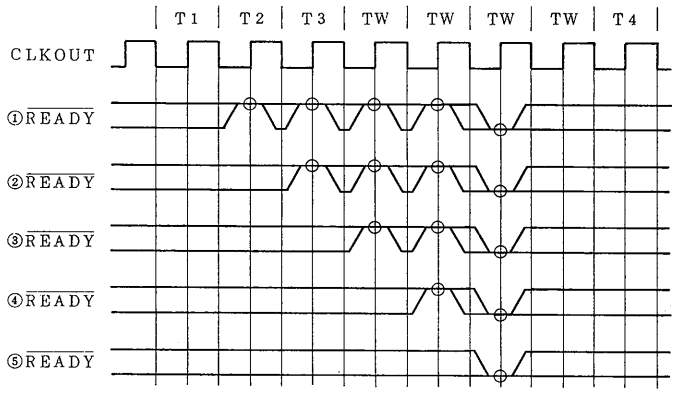
(b) CPUサイクルのバス・タイミング例



①…WCUにウエイト数0を設定したとき
②…WCUにウエイト数1を設定したとき
③…WCUにウエイト数2を設定したとき
④…WCUにウエイト数3を設定したとき
⑤…WCUにウエイト数4を設定したとき

備考　○印はサンプリングされるタイミングです.

(c) DMAサイクル，リフレッシュ・サイクルのバス・タイミング例



①…WCUにウエイト数0を設定したとき
②…WCUにウエイト数1を設定したとき
③…WCUにウエイト数2を設定したとき
④…WCUにウエイト数3を設定したとき
⑤…WCUにウエイト数4を設定したとき

備考　○印はサンプリングされるタイミングです.

6．2．4　バス・サイクル

$\overline{\text{READY}}$のサンプルと，WCUによるプログラマブル・ウエイトについて，バス・サイクルごとに以下の表に示します．

バス・サイクル	$\overline{\text{READY}}$端子のサンプリング	プログラマブル・ウエイトの設定
外部I／Oリード・サイクル	○	○注1
外部I／Oライト・サイクル	○	
割り込みアクノリッジ・サイクル（スレーブ）	○	
割り込みアクノリッジ・サイクル（ICU）	○	
命令フェッチ・サイクル	○	○（領域指定可能）
CPUメモリ・リード・サイクル	○	
CPUメモリ・ライト・サイクル	○	
コプロセッサ用メモリ・リード・サイクル	○	
コプロセッサ用メモリ・ライト・サイクル	○	
DMAリード転送サイクル	○	○
DMAライト転送サイクル	○	
リフレッシュ・サイクル	○	○
内部I／Oリード・サイクル	×注2	×
内部I／Oライト・サイクル	×注2	×
コプロセッサ・リード・サイクル	○	×
コプロセッサ・ライト・サイクル	○	×
ホールト・アクノリッジ・サイクル	×	×
DMAカスケード	×	×

備考　○：有効　×：無効

注 1.　割り込みアクノリッジ・サイクルの場合，0ウエイト設定時は2ウエイト，1ウエイト設定時は3ウエイトとなります．

2.　PGR1－PGR64，XAMレジスタへのアクセスは0ウエイトで行われますが，そのほかのレジスタへのアクセスでは，自動的に2クロックのウエイト・ステートが挿入されます．

6．2．5　メモリ／外部I／Oサイクル

V53のメモリ空間は16Mバイトあり，システム構成時にメモリ空間によってはウエイトを必要としない場合もあり，全メモリ空間に対し同一のウエイト数を挿入することは，システムの性能の低下にもつながります．そのため，WCUでは16Mバイトのメモリ空間を3分割し，それぞれの領域に対し，0－7のウエイト数を独立に設定できるようになっています．また，さらに16Mバイト空間の任意の1Mバイト空間を3分割可能になっています．ここで設定されるウエイト数は，CPUによる命令フェッチ・サイクル，オペランド・アクセス・サイクル，外部I／Oアクセス・サイクルに対し有効になります．

メモリ／外部Ｉ／Ｏサイクルに対するプログラマブル・ウエイト制御は以下のレジスタによって行います．

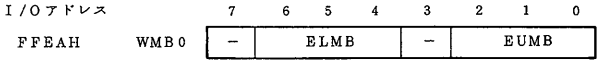
表６－２　ＷＣＵのレジスタ

レジスタ	Ｉ／Ｏアドレス	操　作
WMB0	FFEAH	リード／ライト
WCY1	FFEBH	リード／ライト
WCY0	FFECH	リード／ライト
WAC	FFEDH	リード／ライト
WMB1	FFF3H	リード／ライト
WCY2	FFF4H	リード／ライト
WCY3	FFF5H	リード／ライト
WCY4	FFF6H	リード／ライト

(1)　WMB0（プログラマブル・ウエイト・メモリ領域設定レジスタ０）

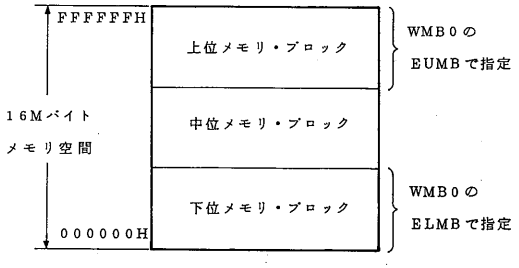
WMB0は16Mバイトのメモリ空間を下位／中位／上位の３つのメモリ・ブロックに分割します．ELMBとEUMBフィールドにより，下位ブロック，上位ブロックを設定し，その中間が中位ブロックになります．

図６－７　WMB0レジスタ



ELMB，EUMB	16Mバイト・メモリ空間の下位，上位メモリ・ブロック・サイズ
0　0　0	1Mバイト
0　0　1	2Mバイト
0　1　0	3Mバイト
0　1　1	4Mバイト
1　0　0	5Mバイト
1　0　1	6Mバイト
1　1　0	7Mバイト
1　1　1	8Mバイト

図６－８　WMB0による16Mバイト・メモリ空間3分割の概念図



(2) WCY0（プログラマブル・ウエイト・サイクル数設定レジスタ0）

WCY0はWMB0で設定した16Mバイトの上位メモリ・ブロックに対して0－7のウエイト数の設定を行います．

図6－9 WCY0レジスタ

I/Oアドレス		7	6	5	4	3	2	1	0
FFECH	WCY0	－	－	－	－	－	EUMW		

EUMW	16Mバイト中の上位メモリ・ブロック・アクセス時のウエイト挿入指定
0 0 0	ウエイトを自動挿入しない
0 0 1	1ウエイトを自動挿入する
0 1 0	2ウエイトを自動挿入する
0 1 1	3ウエイトを自動挿入する
1 0 0	4ウエイトを自動挿入する
1 0 1	5ウエイトを自動挿入する
1 1 0	6ウエイトを自動挿入する
1 1 1	7ウエイトを自動挿入する

(3) WCY1（プログラマブル・ウエイト・サイクル数設定レジスタ1）

WCY1はWMB0で設定した16Mバイトの中位，下位メモリ・ブロックに対して0－7のウエイト数の設定を行います．

図6－10 WCY1レジスタ

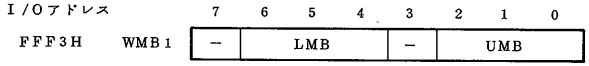
I/Oアドレス		7	6	5	4	3	2	1	0
FFEBH	WCY1	－	EMMW			－	ELMW		

EMMW, ELMW	16Mバイト中の中位，下位メモリ・ブロック・アクセス時のウエイト挿入指定
0 0 0	ウエイトを自動挿入しない
0 0 1	1ウエイトを自動挿入する
0 1 0	2ウエイトを自動挿入する
0 1 1	3ウエイトを自動挿入する
1 0 0	4ウエイトを自動挿入する
1 0 1	5ウエイトを自動挿入する
1 1 0	6ウエイトを自動挿入する
1 1 1	7ウエイトを自動挿入する

(4) WMB1（プログラマブル・ウエイト・メモリ領域設定レジスタ1）

WMB1は1Mバイトのメモリ空間を下位/中位/上位の3つのメモリ・ブロックに分割します．LMBとUMBフィールドにより，下位ブロック，上位ブロックを設定し，その中間が中位ブロックになります．

図6－11　WMB1レジスタ

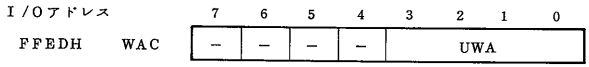


LMB, UMB	1Mバイト・メモリ空間の下位，上位メモリ・ブロック・サイズ
0 0 0	32Kバイト
0 0 1	64Kバイト
0 1 0	96Kバイト
0 1 1	128Kバイト
1 0 0	192Kバイト
1 0 1	256Kバイト
1 1 0	384Kバイト
1 1 1	512Kバイト

(5) WAC（プログラマブル・ウエイト・メモリ・アドレス・コントロール・レジスタ）

WACは16Mバイト・メモリ空間の上位4ビット・アドレスを設定し，WMB1で設定する1Mバイトのメモリ空間を決定します．

図6－12　WACレジスタ



UWA：1Mバイトの自動ウエイト設定可能メモリ空間を指定する上位4ビット

1Mバイト・メモリ領域の16Mバイト空間内でのアドレスは，下図のようになります．

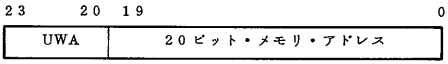
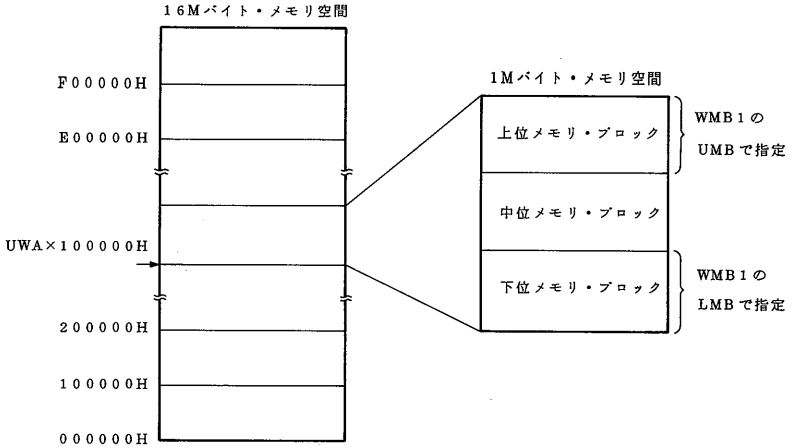


図６－１３　WAC，WMB１ による１Mバイト・メモリ空間３分割の概念図



備考　UWAはWACレジスタの下位４ビットです．

(6)　WCY２（プログラマブル・ウエイト・サイクル数設定レジスタ２）

WCY２はWMB１で設定した１Mバイトの中位，下位メモリ・ブロックに対して０－７のウエイト数の設定を行います．

図６－１４　WCY２レジスタ

I/Oアドレス		7	6	5	4	3	2	1	0
FFF4H	WCY２	－	MMW			－	LMW		

MMW，LMW	１Mバイト中の中位，下位メモリ・ブロック・アクセス時のウエイト挿入指定
0　0　0	ウエイトを自動挿入しない
0　0　1	１ウエイトを自動挿入する
0　1　0	２ウエイトを自動挿入する
0　1　1	３ウエイトを自動挿入する
1　0　0	４ウエイトを自動挿入する
1　0　1	５ウエイトを自動挿入する
1　1　0	６ウエイトを自動挿入する
1　1　1	７ウエイトを自動挿入する

(7) WCY3（プログラマブル・ウエイト・サイクル数設定レジスタ3）

WCY3はWMB1で設定した1Mバイトの上位メモリ・ブロック，または外部I／Oバス・サイクルに対して0－7のウエイト数の設定を行います．

注意　内部I／O領域に対するリード／ライト・サイクルでは無効です．割り込みアクノリッジ・サイクルでは有効になります．

図6－15　WCY3レジスタ

I/Oアドレス		7	6	5	4	3	2	1	0
FFF5H	WCY3	－	IOW			－	UMW		

IOW 注	外部I／Oサイクル時のウエイト挿入指定	割り込みアクノリッジ・サイクル時のウエイト挿入指定
0 0 0	ウエイトを自動挿入しない	2ウエイトを自動挿入する
0 0 1	1ウエイトを自動挿入する	3ウエイトを自動挿入する
0 1 0	2ウエイトを自動挿入する	2ウエイトを自動挿入する
0 1 1	3ウエイトを自動挿入する	3ウエイトを自動挿入する
1 0 0	4ウエイトを自動挿入する	4ウエイトを自動挿入する
1 0 1	5ウエイトを自動挿入する	5ウエイトを自動挿入する
1 1 0	6ウエイトを自動挿入する	6ウエイトを自動挿入する
1 1 1	7ウエイトを自動挿入する	7ウエイトを自動挿入する

UMW	1Mバイト中の上位メモリ・ブロック・アクセス時のウエイト挿入指定
0 0 0	ウエイトを自動挿入しない
0 0 1	1ウエイトを自動挿入する
0 1 0	2ウエイトを自動挿入する
0 1 1	3ウエイトを自動挿入する
1 0 0	4ウエイトを自動挿入する
1 0 1	5ウエイトを自動挿入する
1 1 0	6ウエイトを自動挿入する
1 1 1	7ウエイトを自動挿入する

注　IOWが000，001のときは，外部I/Oサイクルと割り込みアクノリッジ・サイクルとでは挿入されるウエイトが異なります．

6．2．6　DMA／リフレッシュ・サイクル

DMAサイクルおよびリフレッシュ・サイクルに対するウエイト数の設定はWCY4で行います．

(1)　WCY4（プログラマブル・ウエイト・サイクル数設定レジスタ4）

WCY4はDMAサイクル，リフレッシュ・サイクルに対して0－7のウエイト数の設定を行います．DMAサイクル，リフレッシュ・サイクルには，メモリ空間の分割は無効です．

図6－16　WCY4レジスタ

I/Oアドレス		7	6	5	4	3	2	1	0
FFF6H	WCY4	－	DMAW			－	RFW		

DMAW	DMAサイクル時のウエイト挿入指定
0 0 0	ウエイトを自動挿入しない
0 0 1	1ウエイトを自動挿入する
0 1 0	2ウエイトを自動挿入する
0 1 1	3ウエイトを自動挿入する
1 0 0	4ウエイトを自動挿入する
1 0 1	5ウエイトを自動挿入する
1 1 0	6ウエイトを自動挿入する
1 1 1	7ウエイトを自動挿入する

RFW	リフレッシュ・サイクル時のウエイト挿入指定
0 0 0	ウエイトを自動挿入しない
0 0 1	1ウエイトを自動挿入する
0 1 0	2ウエイトを自動挿入する
0 1 1	3ウエイトを自動挿入する
1 0 0	4ウエイトを自動挿入する
1 0 1	5ウエイトを自動挿入する
1 1 0	6ウエイトを自動挿入する
1 1 1	7ウエイトを自動挿入する

6．2．7　WCU設定例

レジスタ	設定値 76543210	設定内容
WMB0	−100−101	16Mバイト空間の3分割
WMB1	−011−010	1Mバイト空間の3分割
WCY0	----−100	⑦
WCY1	−101−011	①，②，⑥
WCY2	−010−001	③，④
WCY3	−111−011	⑤，⑧，⑨
WCY4	−101−010	⑩，⑪
WAC	----−0111	1Mバイト空間の開始アドレス

① 000000H−4FFFFFHのメモリ空間に対する
命令フェッチ，オペランド・アクセス
…… 6ウエイト

② 500000H−6FFFFFHのメモリ空間に対する
命令フェッチ，オペランド・アクセス
…… 5ウエイト

③ 700000H−71FFFFHのメモリ空間に対する
命令フェッチ，オペランド・アクセス
…… 1ウエイト

④ 720000H−7E7FFFHのメモリ空間に対する
命令フェッチ，オペランド・アクセス
…… 2ウエイト

⑤ 7E8000H−7FFFFFHのメモリ空間に対する
命令フェッチ，オペランド・アクセス
…… 3ウエイト

⑥ 800000H−9FFFFFHのメモリ空間に対する
命令フェッチ，オペランド・アクセス
…… 5ウエイト

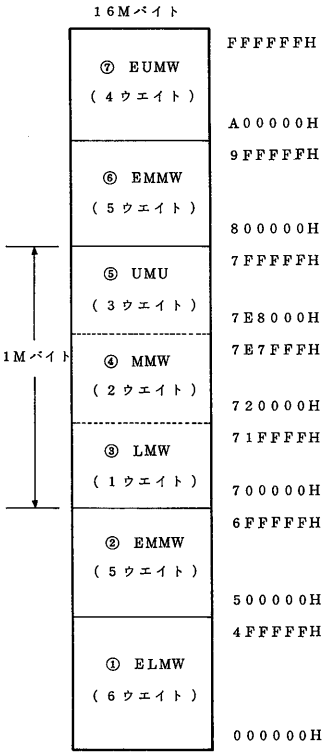
⑦ A00000H−FFFFFFHのメモリ空間に対する
命令フェッチ，オペランド・アクセス
…… 4ウエイト

⑧ 外部I/Oバス・サイクル
…… 7ウエイト

⑨ 割り込みアクノリッジ・バス・サイクル
…… 7ウエイト

⑩ DMAバス・サイクル
…… 5ウエイト

⑪ リフレッシュ・バス・サイクル
…… 2ウエイト



注意　上図①−⑦は項左の①−⑦と対応しています。

6.3 REFU(リフレッシュ・コントロール・ユニット)

6.3.1 特　徴

REFUは，外部DRAMのリフレッシュ動作に必要なリフレッシュ・サイクルを発生します．リフレッシュの許可／禁止，およびリフレッシュ間隔をRFCレジスタによって設定します．

- 最低優先リフレッシュ／最高優先リフレッシュ
- 7リフレッシュ・キュー内蔵
- 16ビット・リフレッシュ・アドレス（$\overline{\text{RAS}}$オンリー・リフレッシュ用アドレス）
- 4クロック／1バス・サイクル

6.3.2 V50に対する変更箇所

リフレッシュ・アドレス出力をA15－A0の16ビットとしています．

6.3.3 リフレッシュ・アドレス

リフレッシュ・アドレスはA15－A0のアドレスを出力します．1回のリフレッシュ・サイクルごとに，リフレッシュ・アドレスを1または2ずつインクリメントし，次のリフレッシュ・アドレスを生成します．

また，リフレッシュ・サイクルでの上位アドレス（A23－A16）は，ロウ・レベルを出力します．

このリフレッシュ・アドレスは動作中のリセットで影響を受けません．また，パワーオン時のリフレッシュ・アドレスは不定です．

6.3.4 RFC(リフレッシュ・コントロール・レジスタ)

RFCはシステムI/O領域にマッピングされており，以下の機能を有します．

(1) リフレッシュ許可／禁止

リフレッシュの許可／禁止はRFEビット(ビット7)で行います．RFEビットはリセットで初期化されませんので，パワーオン時に許可になっている場合も禁止になっている場合もあります．したがって，パワーオン・リセットから144(16×9)クロック後にリフレッシュ・サイクルが起動する場合もあります．つまり，この期間内にRFEビットを0にすることで，リフレッシュ・サイクルを1回も実行しないでリフレッシュ禁止にすることができます．

(2) リフレッシュ間隔

リフレッシュ間隔は，RTMビット(ビット4－0)で設定します．リフレッシュ間隔は個個のシステムによって下記の式に従って最適な値を設定してください．

リフレッシュ間隔(クロック)＝16×タイマ・ファクタ(N)

たとえば，16MHz動作時にタイマ・ファクタN＝20に設定すると，320クロックに1回の割合でリフレッシュ要求が発生します．すなわち，320×62.5ns＝20μsに1回の割合でリフ

レッシュ・サイクルが起動されることになります．ただし，後述するようにほかのバス・マスタがバスを占有している場合には，リフレッシュ要求が待たされる場合があります．

リフレッシュ間隔の最低設定値はN＝1であり，リセット時にはN＝9に初期化されます．

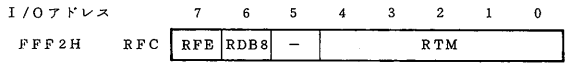
(3) **アドレス更新制御**

リフレッシュ・アドレスを1ずつインクリメントするか，2ずつインクリメントするかを，RDB8ビット（ビット6）で選択します．

表6－3　RFCのアドレスと操作

レジスタ	I／Oアドレス	操　作
RFC	FFF2H	リード／ライト

図6－17　RFC（リフレッシュ・コントロール・レジスタ）



RFE	リフレッシュ許可／禁止
0	リフレッシュ禁止
1	リフレッシュ許可

RDB8	リフレッシュ・アドレス更新制御
0	リフレッシュ・アドレスを2ずつインクリメント $\overline{UBE}$：ロウ・レベル出力
1	リフレッシュ・アドレスを1ずつインクリメント $\overline{UBE}$：偶数アドレス時はハイレベル，奇数アドレス時はロウ・レベル

RTM	タイマ・ファクタの設定
00000	N（タイマ・ファクタ）＝1
00001	N（タイマ・ファクタ）＝2
00010	N（タイマ・ファクタ）＝3
00011	N（タイマ・ファクタ）＝4
00100	N（タイマ・ファクタ）＝5
00101	N（タイマ・ファクタ）＝6
≈ ⋮ ≈	⋮ ≈
11110	N（タイマ・ファクタ）＝31
11111	N（タイマ・ファクタ）＝32

6．3．5　リフレッシュ・サイクル

リフレッシュ・サイクルは図6－18に示すように，4クロック／1バス・サイクルで構成され，プログラマブル・ウエイトまたは外部$\overline{\text{READY}}$制御により，引き延ばすこともできます．リフレッシュ・サイクルの特徴は，アクティブ・ロウの$\overline{\text{REFRQ}}$信号を伴うことです．$\overline{\text{MRD}}$信号，$\overline{\text{DSTB}}$信号および$\overline{\text{BUFEN}}$信号はインアクティブのままとなります．

このようなリフレッシュ・サイクルは，前項で説明したリフレッシュ間隔をもってREFUからBAUへリフレッシュ要求が送られます．もちろん，バスがアイドル（TI）状態のときには，すぐにこの要求は受け付けられますが，ほかのバス・マスタ（CPU，DMAU，外部デバイス）がバスを占有している場合には，リフレッシュ要求はいったんリフレッシュ・キューに蓄えられます．すなわち，通常のリフレッシュ要求は，最低優先のバス使用権しか有しておらず，ほかのバス・マスタからバスの制御を奪い取ることはできません．

しかし，あまり長い間リフレッシュ要求が保留されると，DRAMのリフレッシュ期間を守れなくなる可能性があります．そのため，リフレッシュ要求が7回分保留されると（リフレッシュ・キューの内容が7になると），REFUは最高優先のバスの使用権を与えられます．REFUは最高のバス使用権が与えられると，ほかのバス・マスタに対しバスの返還要求を出し，リフレッシュ・キューの内容が3になるまで連続して4（または5）回リフレッシュ・サイクルを実行します．通常は最高優先のリフレッシュ・サイクルは4回連続して発行されますが，リフレッシュ・サイクル実行中に次のリフレッシュ要求が発生する場合もありますので，その場合は，5回連続して実行されます．

また，リフレッシュ・キューはリフレッシュ要求を7つまでしか蓄えることができないため，ほかのバス・マスタがバスを返還せず，リフレッシュ・キューの内容が7のまま次のリフレッシュ要求が発生した場合は，以降のリフレッシュ要求はすべて無効となります．ただし，リフレッシュ・キューの内容は7のままで，リフレッシュ・アドレスも更新されません（リフレッシュ・アドレスはリフレッシュ・サイクルが実行されないかぎり更新されません）．

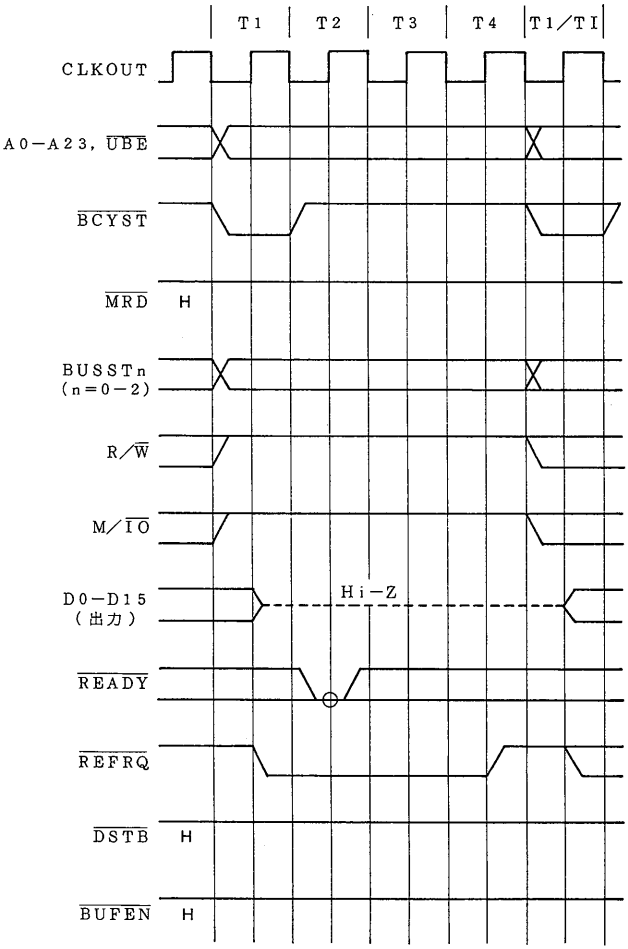
リフレッシュ・キューにリフレッシュ要求が保留されているとき，RFEビットを0（リフレッシュ禁止）にしても，リフレッシュ・キューの内容が0になるまでリフレッシュ・サイクルを実行します．

リフレッシュ・キューはリセットによりクリアされます．

バスの使用権，およびアービトレーションに関しては，「6.8 BAU」の節で詳しく説明します．

図6－18　リフレッシュ・サイクル（1／2）

(a)　ウエイトなし

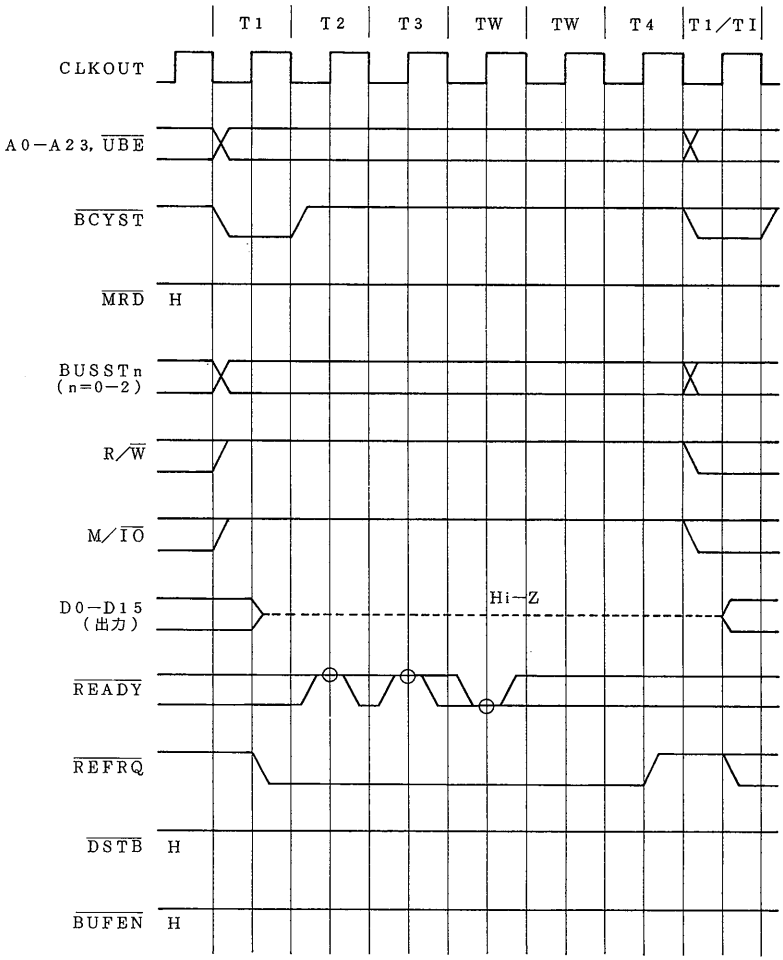


備考　○印はWCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUとREADY端子の関係は，6.2.3 WCUとREADY端子の関係を参照してください．

図6－18　リフレッシュ・サイクル（2／2）

(b)　2ウエイト時



備考　○印は，WCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUとREADY端子の関係は，6.2.3 WCUとREADY端子の関係を参照してください．

6.4 ICU(割り込みコントロール・ユニット)

ICUは8本までの割り込み要求を調停し，その中の1つをCPUに伝えます．

機能的にはμPD71059からCALLモード(8085モード)，カスケード接続時のスレーブ機能を取り除いたものとなっています．

6.4.1 特　徴

- μPD71059相当
  - カスケード接続によりチャネル拡張可能
  - エッジ・トリガ/レベル・トリガ選択可能
  - 割り込み要求は個々にマスク可能
  - 割り込み要求の優先順位がプログラマブル
  - ポーリング動作可能
- 8割り込み要求入力(INTP0－INTP7)
- 外部8入力端子あり
- $\overline{\text{INTAK}}$出力端子あり

注意　V53はスタンバイ・モード時の消費電流低減のため，INTP0－INTP7端子に，プルアップ抵抗を内蔵していません．μPD71059やV40，V50とは異なります．

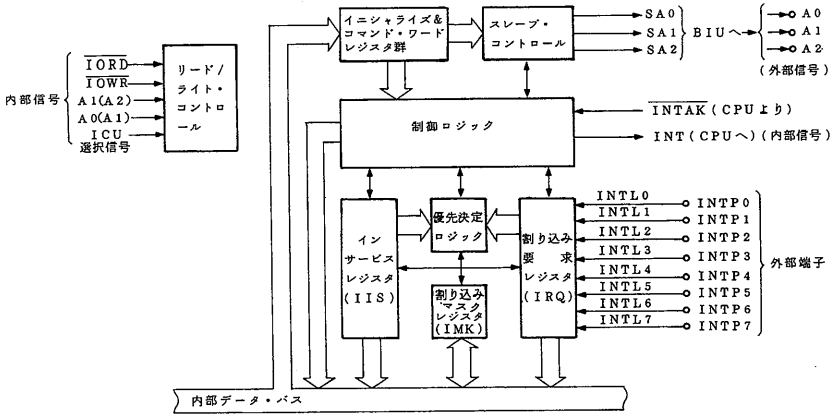
6.4.2 V50に対する変更箇所

V50では，内蔵ICUへの割り込み入力は外部端子(INTP1－INTP7)，タイマ出力(TOUT0，TOUT1)，シリアル割り込み要求(SINT)より選択できましたが，V53では外部端子(INTP0－INTP7)のみとしています．

このため，タイマ出力3本，およびシリアル割り込み要求3本を端子に出力しています(マルチプレクスなし)．

また，INTP0－INTP7端子は，プルアップ抵抗を内蔵していません．

6．4．3　ＩＣＵ内部ブロック図



6．4．4　アドレシング

ＩＣＵの内部レジスタは，システムＩ/Ｏ領域で設定したＯＰＨＡとＩＵＬＡ，およびアドレス信号（Ａ０またはＡ１）によってアドレスされます．ＩＯＡＧ＝１のときのＡ１またはＩＯＡＧ＝０のときのＡ２は，０，１どちらでも構いません．（ＩＯＡＧはＳＣＴＬレジスタのビット０）．

表６－４　ＩＣＵ内部レジスタのアドレス

	上位アドレス	下位アドレス			レジスタ，コマンド	操　作
ＩＯＡＧ＝１のとき	Ａ１５－Ａ８（ＯＰＨＡ）	Ａ７　Ａ６　Ａ５　Ａ４　Ａ３　Ａ２（ＩＵＬＡ）		－　０	ＩＲＱ/ＩＩＳ/ＩＰＯＬ	リード
					ＩＩＷ１/ＩＰＦＷ/ＩＭＤＷ	ライト
				－　１	ＩＭＫＷ	リード/ライト
					ＩＩＷ２/ＩＩＷ３/ＩＩＷ４	ライト
ＩＯＡＧ＝０のとき	Ａ１５－Ａ８（ＯＰＨＡ）	Ａ７　Ａ６　Ａ５　Ａ４　Ａ３（ＩＵＬＡ）	－　０	Ａ０	ＩＲＱ/ＩＩＳ/ＩＰＯＬ	リード
					ＩＩＷ１/ＩＰＦＷ/ＩＭＤＷ	ライト
			－　１		ＩＭＫＷ	リード/ライト
					ＩＩＷ２/ＩＩＷ３/ＩＩＷ４	ライト

備考　－は０でも１でもかまいません．

ICUは，複数のレジスタに対しアドレス信号は1本しかないため，レジスタの読み出し/書き込みを行うには一定のシーケンスが必要です．たとえば，初期化の際は，4つのイニシャライズ・シーケンスのうちから使用目的に合わせて1つ選んで，IIW1，IIW2，IIW3，IIW4を書き込みます．

書き込み可能なレジスタにはIPFW，IMDW，IMKWがあります．IMKWはイニシャライズ・シーケンス後は，A0(A1)=1として書き込めますが，IPFW，IMDWはどちらもA0(A1)=0なので，書き込むデータのビット4，3の値で区別します．

レジスタの中で読み出せるのは，IRQ，IIS，IMKWの3つです．IMKWはA0(A1)=1としていつでも読み出せますが，IRQ，IISはどちらもA0(A1)=0なので，読み出しを行う前にIMDWでどちらを読み出すかをあらかじめ指定してから読み出します．

表6－5にICUのレジスタ，コマンド・アドレスについてまとめます．

表6－5　ICUのレジスタ，コマンド・アドレス

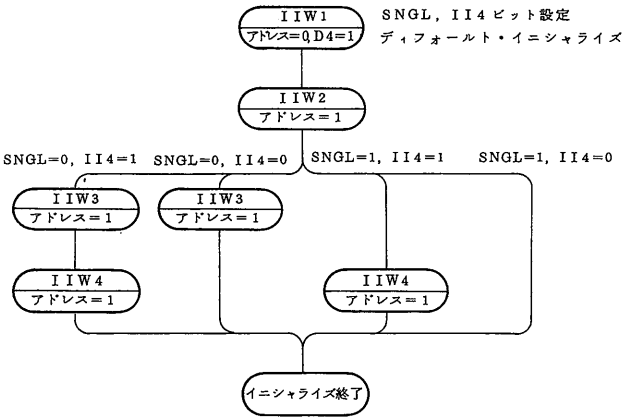
	A0(A1)	A0(A1)のほかの条件	動作
リード	0	IMDWでIRQを設定	CPU←IRQデータ
		IMDWでIISを設定	CPU←IISデータ
		ポーリング・フェーズ注	CPU←IPOL
	1	—	CPU←IMKW
ライト	0	ビット4＝1	CPU→IIW1
		ビット4，3＝00	CPU→IPFW
		ビット4，3＝01	CPU→IMDW
	1	イニシャライズ・シーケンス中	CPU→IIW2
			CPU→IIW3
			CPU→IIW4
		イニシャライズの後	CPU→IMKW

注　ポーリング・フェーズではIRQまたはIISに優先してIPOLのポーリング・データが読み出されます．

6．4．5　ＩＣＵの初期化

ＩＣＵは $\overline{\text{RESET}}$ 信号によって初期化されません．そのため，パワーオン時には必ずＩＣＵを初期化してください．初期化は，イニシャライズ・シーケンスに従ってＩＩＷ1，ＩＩＷ2，ＩＩＷ3，ＩＩＷ4を書き込むことによって行われますが，図６－１９に示すように４通りのイニシャライズ・シーケンスがあります．

図６－１９　イニシャライズ・シーケンス



備考　アドレスとはＡ1またはＡ0のことです．

イニシャライズ・シーケンスは，Ａ0（Ａ1）＝0に対し，ビット４が１であるデータを書き込んだ時点からスタートし，そのとき書き込んだ値がＩＩＷ1になります．そして，そのＩＩＷ1中のＳＮＧＬビット，ＩＩ４ビットの値によって４つのイニシャライズ・シーケンスのうちの１つが選択されます（図６－２０参照）．また，ＩＩＷ1を書き込んだ時点でディフォールト・イニシャライズ[注]が実行されます．

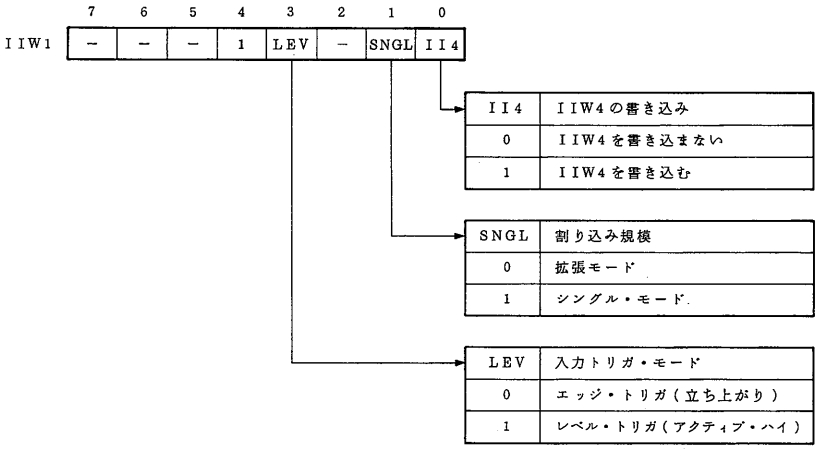
注　ディフォールト・イニシャライズは次のように実行されます．

(1)　エッジ・トリガ・モードの場合，ＩＲＱレジスタがクリア（＝０）されます．
(2)　ＩＩＳ，ＩＭＫＷレジスタがクリア（＝０）されます．
(3)　チャネル０が最高優先となります．
(4)　通常ネスト・モードが設定されます．
(5)　ＩＲＱレジスタが読み出しレジスタに設定されます．
(6)　ＩＩＷ4がクリア（＝０）されます．

6．4．6　レジスタ／コマンド

(1)　IIW1（割り込みイニシャライズ・ワード1）

図6－20　IIW1レジスタ

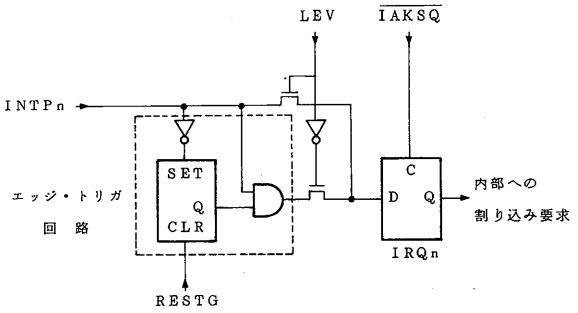


(a)　LEV

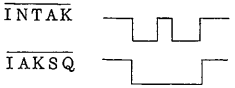
LEV ビットは，割り込み要求のトリガ・モードを設定します．

レベル・トリガの場合はINTPn入力が“1”ならば何度でも割り込み要求として認められますが，エッジ・トリガの場合は立ち上がりエッジがなければ割り込み要求として認められません．すなわち，エッジ・トリガでは，一度割り込みが受け付けられると，INTPn入力をいったん“0”にしてから再び“1”にしないと割り込み要求としてみなされません．

図6－21　割り込み要求入力回路

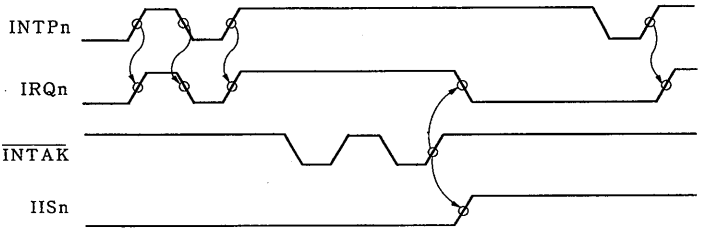


備考1.　IAKSQ は，割り込みアクノリッジ・シーケンスにおいてロウ・レベルとなる信号です．



2.　RESTGは，IIW1の書き込み時と IAKSQ の立ち上がり時にハイ・レベルとなる信号です．F／Fをクリアするとロウ・レベルになります．

図6－22　割り込み要求入力タイミング例



(b)　SNGL

SNGL ビットは，割り込み規模（カスケード接続の有無）を指定します．つまり，内蔵のICUのみを使用する場合は“1”にしてシングル・モードに設定し，INTPn端子にμPD71059などの割り込みコントローラをカスケード接続する場合には“0”にして拡張モードに設定します．

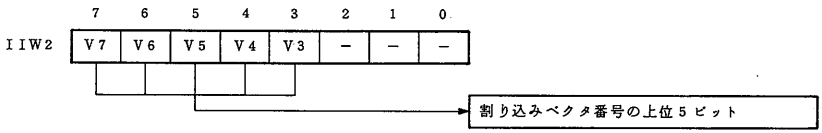
“0”に設定した場合は，イニシャライズ・シーケンスにおいてIIW3を書き込みます．

(c) II4

II4ビットは，イニシャライズ・シーケンスにおいてIIW4を書き込むか否かを指定します．IIW4を書き込まない場合は，IIW4はディフォールト・イニシャライズにより00Hにクリアされたままとなります．

(2) IIW2（割り込みイニシャライズ・ワード2）

図6－23 IIW2レジスタ



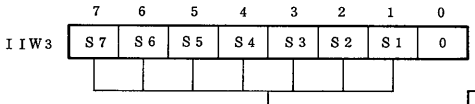
V7－V3は，8ビット割り込みベクタの内の上位5ビットを設定します．
残り下位3ビットは割り込みの受け付けられたレベルに応じて出力されます．

図6－24 割り込みベクタ番号の発生

割り込みレベル	7	6	5	4	3	2	1	0
INTP0	V7	V6	V5	V4	V3	0	0	0
INTP1	V7	V6	V5	V4	V3	0	0	1
INTP2	V7	V6	V5	V4	V3	0	1	0
INTP3	V7	V6	V5	V4	V3	0	1	1
INTP4	V7	V6	V5	V4	V3	1	0	0
INTP5	V7	V6	V5	V4	V3	1	0	1
INTP6	V7	V6	V5	V4	V3	1	1	0
INTP7	V7	V6	V5	V4	V3	1	1	1

(3) IIW3（割り込みイニシャライズ・ワード3）

図6－25　IIW3レジスタ



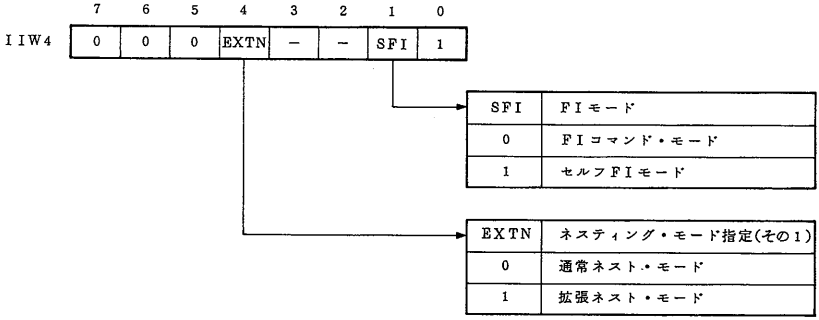
Sn	スレーブ接続状況
0	INTPnは非スレーブ接続
1	INTPnはスレーブ接続

このIIW3は，カスケード接続時のみ有効となります．すなわち，INTPnが，スレーブ（割り込みチャネルを拡張するために外部に接続する割り込みコントローラ）としてμPD71059などをカスケード接続している場合に，該当するSnを“1”にします．たとえばINTP5にスレーブが接続されている場合は，S5を“1”に設定します．このときICUはベクタを出力せず，スレーブのμPD71059が2回目の割り込みアクノリッジ・サイクルでデータ・バスのD0－D7にベクタを出力しなければなりません．そのあとのCPUの動作は通常の割り込み処理と同じです．

ただし，INTP0端子にはスレーブの割り込みコントローラを接続できません．

(4) IIW4（割り込みイニシャライズ・ワード4）

図6－26　IIW4レジスタ



SFI	FIモード
0	FIコマンド・モード
1	セルフFIモード

EXTN	ネスティング・モード指定(その1)
0	通常ネスト・モード
1	拡張ネスト・モード

(a) EXTN

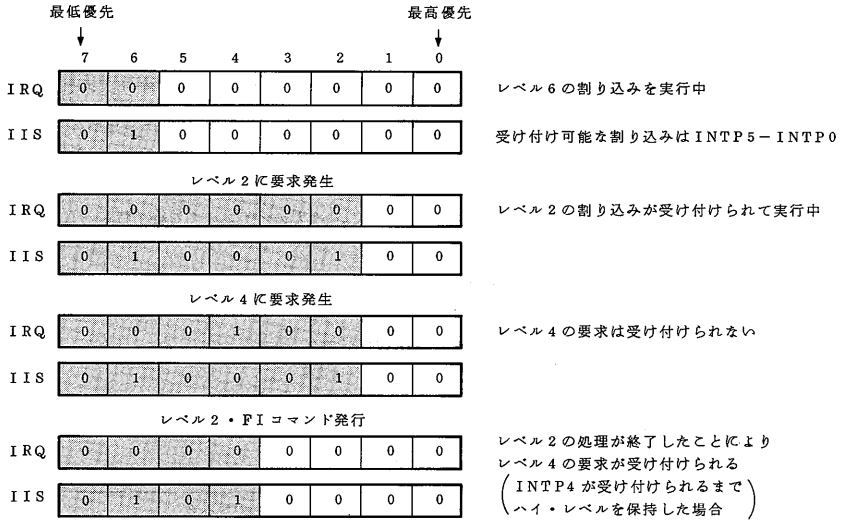
EXTNビットは，ネスティング・モードの設定を行います．ネスティング・モードには，通常ネスト・モード，拡張ネスト・モード，例外ネスト・モードの3種類があります．例外ネスト・モードにはIMDWレジスタで設定します（（7）項参照）．

・通常ネスト・モード

このモードは，実行中の割り込みレベル（すなわち，IISレジスタの該当ビットが“1”である割り込みレベル）よりも優先順位の高い割り込み要求しか受け付けません．ただし，セルフFIモードではIISの該当ビットが割り込み受け付けとほとんど同時にクリア（0）されてしまうため，どのレベルの割り込み要求も優先順位の高い順に受け付けられます．

IRQは割り込み要求の発生しているレベルを示すレジスタで，IISは現在サービス中の割り込みレベルを示すレジスタです（図6－34，35参照）．

図6－27　通常ネスト・モード



備考　□は受け付けられないレベルを示します．

・拡張ネスト・モード

このモードは後述する拡張モードのときに意味をもちます．拡張ネスト・モードではIIW3で指定したスレーブのμPD71059からの割り込み要求にかぎり，同じレベルであっても割り込み要求を受け付けます．たとえば，あるスレーブのμPD71059の1つの割り込みが実行中に，同じスレーブのμPD71059からより優先順位の高い割り込みであっても，通常ネスト・モードでは受け付けられません．これでは，スレーブのμPD71059に設定した優先順位の意味がなくなるからです．

注意　拡張モードにおけるFIコマンドの発行は，次のような手順で行ってください．FIコマンドとは，割り込み処理の終了をICUに知らせるためにCPUが発行するコマンドです（（6）項参照）．

スレーブのμPD71059からの割り込みの場合は，まずそのスレーブのμPD71059に対してFIコマンドを発行します．さらに，スレーブのμPD71059内のIISレジスタを調べ，スレーブのμPD71059にまだサービス中の割り込みがあるかどうかをチェックします．スレーブのμPD71059にサービス中の割り込みがない場合に，ICUに対してもFIコマンドを発行し

ます．スレーブのμPD71059が接続されているチャネル以外からの割り込みの場合は，シングル・モードと同様にＩＣＵに対してもＦＩコマンドを発行します．

(b) ＳＦＩ

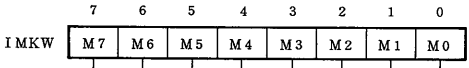
ＳＦＩビットはＦＩモードの設定を行います．この２つのモードは，割り込み処理の終了をＩＣＵに知らせるためのＩＩＳの該当ビットのクリア動作が，次のように異なります．

ＦＩコマンド・モードでは，ＦＩコマンドの発行によりＩＩＳの該当ビットをクリアします．

セルフＦＩモードでは，１回目の割り込みアクノリッジ・サイクル終了時に，ＩＩＳの該当ビットを自動的にクリアします．

(5) ＩＭＫＷ（割り込みマスク・ワード）

図６－２８　ＩＭＫＷレジスタ

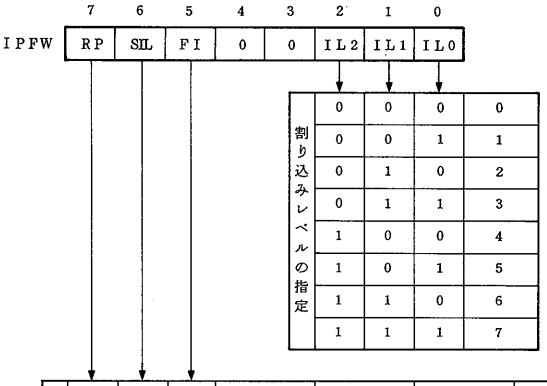


Mn	要求のマスク
0	INTPnをマスクしない
1	INTPnをマスクする

このＩＭＫＷを設定することによって，割り込み要求をマスクすることができます．すなわち割り込み要求の発生によりＩＲＱがセットされても，ＩＭＫＷの該当するビットがセットされていると，ＩＣＵはＣＰＵに割り込みを要求しません．また，例外ネスト・モードではＩＩＳもマスクします（図６－３２参照）．

(6) IPFW (割り込みプライオリティ，フィニッシュ・ワード)

図6－29　IPFWレジスタ



優先順位回転とFIコマンド							
0	0	1	FIコマンド	レベル指定無し	回転無し	通常FIコマンド
1	0	1			回転有り	通常回転FIコマンド
0	1	1		レベル指定有り	回転無し	指定FIコマンド
1	1	1			回転有り	指定回転FIコマンド
0	1	0	非FIコマンド		回転無し	ノー・オペレーション
1	1	0			回転有り	指定回転コマンド
0	0	0		レベル指定無し	回転無し	セルフFIモード回転リセット
1	0	0			回転有り	セルフFIモード回転セット

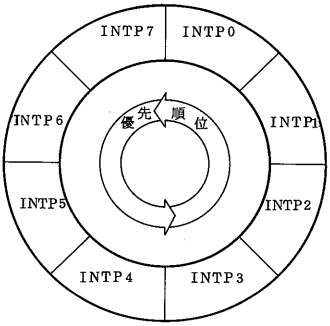
このIPFWの書き込みによって，割り込み要求優先順位の設定／変更と，割り込み処理の終了宣言（FIコマンド発行；IISの該当ビットをクリア）を同時に行います．

FIビットを“1”とすると，IISnビットのクリアを行うFIコマンドとなり，RP，SIL，IL2－IL0ビットにより優先順位，割り込みレベルの指定を行います．

FIビットを“0”とすると，IL2－IL0ビットにより回転を行う指定回転コマンド，あるいはセルフFIモード回転セット／リセットを指定するコマンドになります．

RPビットは優先順位の回転／固定を決めるビットで，“1”のとき回転となり，レベル指定がある場合は，IL0－IL2ビットで指定された割り込みレベルを最低優先にします．また，レベル指定がない場合には，そのときの割り込みレベルを最低優先にします．優先順位は初期化終了時にはINTP7が最低で，INTP0が最高優先になっています．優先順位を回転させた場合には，図6－30に示すようにINTPnを最低優先にするとINTP(n+1)が最高優先になり，以下図に示すリング状に優先順位が決まります(n=7の場合はn+1=0)．

図6－30　INTL要求の優先順位（回転時）



FIビットを“1”にして以下に説明するコマンドを発行すると，レベル指定のある場合にはそのレベルのIISnビットをクリアし，レベル指定のない場合にはその時点で最も優先順位の高いIISnビットをクリアします．

以下に個々のコマンドを説明します．

(a)　通常FIコマンド

	7	6	5	4	3	2	1	0
IPFW＝	0	0	1	0	0	－	－	－

・優先順位

このコマンドの発行によって優先順位は変りません．

・動作説明

現在サービス中の割り込みの中で，最も優先順位の高い割り込みのIISnビットをクリアします．

(b)　指定FIコマンド

	7	6	5	4	3	2	1	0
IPFW＝	0	1	1	0	0	IL2	IL1	IL0

・優先順位

このコマンドの発行によって優先順位は変りません．

・動作説明

IL2－IL0で指定されたレベルのIISnビットをクリアします．

(c) 通常回転ＦＩコマンド

	7	6	5	4	3	2	1	0
IPFW=	1	0	1	0	0	−	−	−

・優先順位

このコマンドで終了させた割り込みレベルを最低優先にします．

・動作説明

現在サービス中の割り込みのなかで，最も優先順位の高い割り込みのＩＩＳｎビットをクリアします．

(d) 指定回転ＦＩコマンド

	7	6	5	4	3	2	1	0
IPFW=	1	1	1	0	0	IL2	IL1	IL0

・優先順位

ＩＬ２－ＩＬ０で指定された割り込みレベルを最低優先にします．

・動作説明

ＩＬ２－ＩＬ０で指定されたレベルのＩＩＳｎビットをクリアします．

(e) 指定回転コマンド

	7	6	5	4	3	2	1	0
IPFW=	1	1	0	0	0	IL2	IL1	IL0

・優先順位

ＩＬ２－ＩＬ０で指定された割り込みレベルを最低優先にします．

・動作説明

このコマンドの発行によってＩＩＳｎビットは影響を受けません．

(f) セルフＦＩモード

		7	6	5	4	3	2	1	0
セルフFI回転付加	IPFW=	1	0	0	0	0	−	−	−
セルフFI回転非付加	IPFW=	0	0	0	0	0	−	−	−

前述したＩＩＷ４のＳＦＩビットが“１”に設定されていると，ＩＣＵはセルフＦＩモードになります．

・優先順位

回転付加の場合は，割り込みアクノリッジ・シーケンス終了時に，そのシーケンスで

受け付けられた割り込みのレベルを最低優先にします．

回転非付加の場合，優先順位は変化しません．

・動作説明

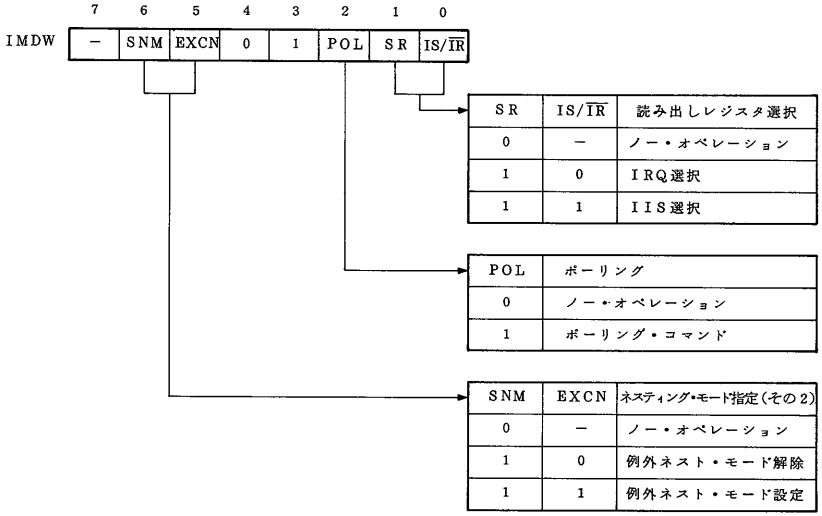
セルフFIモードでは，1回目の割り込みアクノリッジ・パルスの立ち下がりでIISの該当ビットがセットされますが，立ち上がりですぐにリセットされるため，外見上，IISはセットされないように見えます．そのため，IPFWでFIコマンドを発行する必要はありません．

セルフFIモードでのIPFWの発行は優先順位の回転／非回転の設定にだけ必要となります．

セルフFIモードではIISの該当ビットがすぐにクリアされてしまいますので，CPUが割り込み許可状態になっていると，次々に割り込みが受け付けられてしまいます．そのため，スタック・オーバフローを起こしやすいのでネスティングの管理に注意してください．

(7) IMDW (割り込みモード・ワード)

図6－31　IMDWレジスタ



SR	IS/$\overline{IR}$	読み出しレジスタ選択
0	－	ノー・オペレーション
1	0	IRQ選択
1	1	IIS選択

POL	ポーリング
0	ノー・オペレーション
1	ポーリング・コマンド

SNM	EXCN	ネスティング・モード指定(その2)
0	－	ノー・オペレーション
1	0	例外ネスト・モード解除
1	1	例外ネスト・モード設定

(a) SNM, EXCN

SNMビットとEXCNビットとはペアで用いられ，例外ネスト・モードの設定／解除を行います．SNMが“0”の場合は何の設定もせず，イニシャライズ・シーケンスで設定されたネスト・モードがそのまま有効になります．

・例外ネスト・モード

ＩＩＷ4の項で説明したように，割り込みのネスティングのモードには通常ネスト・モードと拡張ネスト・モードがありますが，どちらの場合もサービス中の割り込みよりも優先順位の低い割り込みは受け付けられません．そこで，優先順位の低い割り込みも受け付けたい場合にこの例外ネスト・モードを使用します．

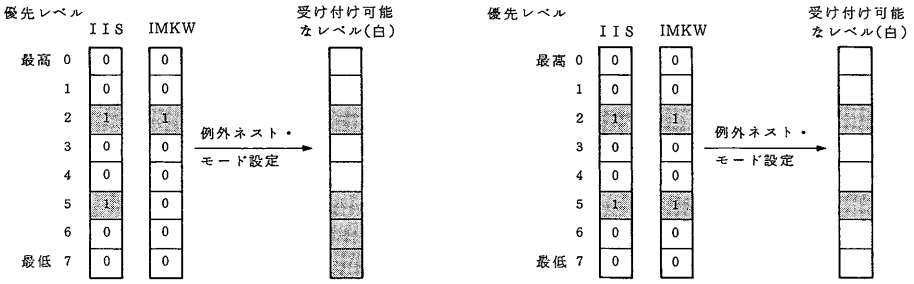
例外ネスト・モードではＩＭＫＷでマスクする対象がＩＲＱとＩＩＳの２つになります．すなわち，ＩＭＫＷで指定したレベルのＩＩＳのビットは“０”であるように見えるためそのレベルよりも優先順位の低い割り込み要求も受け付けるようになります．ただし，ＩＲＱもマスクされていますので，そのレベルに対する再割り込みは禁止されます．

以下に，例外ネスト・モード設定手順の一例を示します．

(i) ＩＩＳを読み出す．
(ii) 読み出した値をＩＭＫＷに書き込む．
(iii) 例外ネスト・モードを設定する．

この例では，現在サービスされていないすべての割り込みを受け付け許可にします．
図６－３２に，例外ネスト・モードでの割り込み要求受け付けレベルの例を示します．

図６－３２　例外ネスト・モード例



ＩＭＫＷのビット２のみを１にした場合は，ＩＮＴＰ６，ＩＮＴＰ７の要求は受け付けられない．

ＩＭＫＷをＩＩＳと同じにした場合は実行中以外のすべての要求が受け付けられる．

注意　例外ネスト・モードではサービス中のＩＩＳビットがマスクされているため“通常ＦＩコマンド”の発行では正常に動作しません．そのため，ＦＩコマンドとして“指定ＦＩコマンド”を発行してください．例外ネスト・モードを解除したあとは通常ネスト・モードを発行しても問題ありません．

(b) POL

POLビットはポーリング動作をするときに設定します．

以下，ポーリングについて説明します．

・ポーリング

ポーリングはCPUが割り込みシーケンスを発行せずに割り込み処理を行いたいとき，または現在サービスをしなければならない要求はどれかという情報を得たいときに使用します．

ポーリングを行う場合，CPUはDI命令を実行し，割り込み禁止状態にする必要があります．次に，POLビットを1としたIMDWを書き込んで，ポーリング・コマンドを発行します．このコマンドを発行してから，次にCPUがICUから読み出すまでの間，ICUはポーリング・フェーズに入ります．ポーリング・フェーズ中にA0，A1＝“0”とした読み出しを行えば，IRQ，IISではなく，割り込みポーリング・レジスタ（IPOL）からポーリング・データが読み出されます．そして，ICUはポーリング・フェーズを終了します．

注意　ポーリング・フェーズ中にA0，A1＝“1”とした読み出しを行えば，IMKWが読み出されますが，その時点でICUはポーリング・フェーズは終了してしまいます．このため，ネスティングが乱れる場合がありますので，ポーリング・フェーズでは必ずポーリング・データを読み出してください．

図6－33　IPOLレジスタ

	7	6	5	4	3	2	1	0
IPOL	INT	0	0	0	0	PL2	PL1	PL0

読み出したポーリング・データのINTビットが1ならば，ICUはPL2－PL0で示されるICUレベルに相当するIISレジスタのビットをセットし，その割り込みがサービスされたとみなします．したがって，CPUは読み出されたポーリング・データによって必要な処理を行う必要があります．その場合，IISレジスタの該当ビットがセットされていますので，処理の終了時にはFIコマンドを発行し，そのビットをクリアしてください．

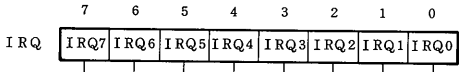
(c) SR，IS／$\overline{IR}$

SRビットとIS／$\overline{IR}$ビットはペアで用いられ，アドレスを0としたときに，読み出すレジスタを指定します．

SR＝0のときは，レジスタの内容は読み出せません．

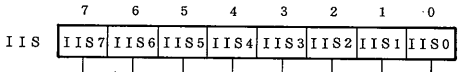
SR＝1，IS／$\overline{IR}$＝0とするとIRQレジスタが，SR＝1，IS／$\overline{IR}$＝1とすると，IISレジスタが指定されます．

図6－34　IRQレジスタのフォーマット



IRQn	INTPnの割り込み要求状態
0	割り込み要求なし
1	割り込み要求あり

図6－35　IISレジスタのフォーマット



IISn	INTPnの割り込み処理状態
0	割り込み処理は実行していない
1	割り込み処理実行中

6．4．7　割り込みシーケンス

ここでは，シングル・モードおよび拡張モードにおける割り込みシーケンスをフロー・チャートを用いて説明します．

(1) シングル・モード

図6－36に，シングル・モードにおける割り込みシーケンスを示します．シングル・モードでは周辺回路が出した割り込み要求をICUが受け付け，CPUに対し割り込み要求を発生し，さらにCPUがその割り込み要求を受け付けて処理を行います．

以下に割り込みシーケンスの順に説明します．

(a) 周辺回路が割り込み要求を発生します．この割り込み要求信号（INTPn）は，V53からの割り込みアクノリッジ信号（$\overline{\text{INTAK}}$）がアクティブになるまで，レベル/エッジのトリガ・モードに関係なく，アクティブ・レベルを保持してください．

(b) INTPnに割り込み要求が発生すると，IRQのビットが1にセットされます．ICUはすべてのIRQビットの中で最高優先のINTPnを受け付けます．

(c) ICUはINTPnを受け付けると，CPUに対し割り込み要求（INT）を発生します．

(d) CPUはICUからの割り込み要求を受け付けると，現在実行中の命令終了時に1回目の割り込みアクノリッジ（$\overline{\text{INTAK}}$）パルスをアクティブとします．前述のように，INTPnはこの時点までアクティブ・レベルを保持しなければなりません．

(e) ICUはCPUから$\overline{\text{INTAK}}$パルスを受け取ると，割り込みアクノリッジ・シーケンスに入ります．

(f) CPUは2回目の$\overline{\text{INTAK}}$パルスをアクティブとし，ICUに対し割り込みベクタ番号を要求します．

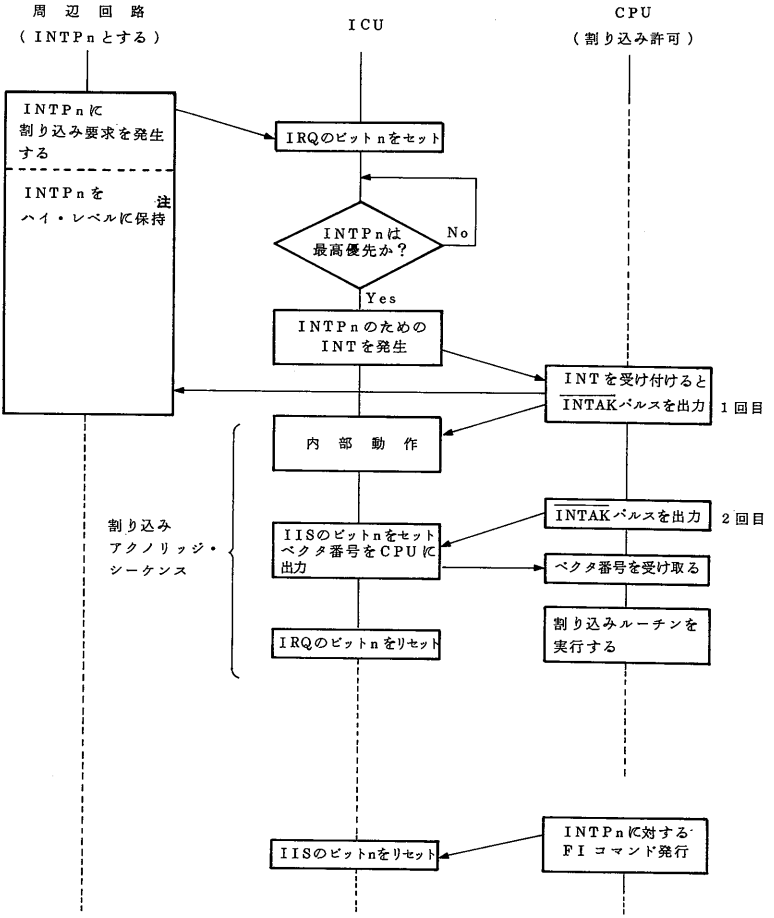
(g) ICUは2回目の$\overline{\text{INTAK}}$パルスに合わせて，8ビットの割り込みベクタをアドレスの下位8ビットに出力します．また，IISのビットnを1にセットし，INTPnに対する割り込み処理中であることを示し，ICUの割り込みアクノリッジ・シーケンスは終了します．ただし，セルフFIモードでは割り込みアクノリッジ・シーケンス終了時にIISのビットnはクリアされます．

(h) CPUは割り込みベクタを受け取り，INTPnに対する割り込み処理を実行します．処理の終了時には，ICUに対し処理終了を告げるFIコマンドを発行します．ただし，セルフFIモードの場合は必要ありません．

(i) ICUはFIコマンドを受け，IISのビットnをクリアします．これでINTPnに対する割り込み処理は完了します．

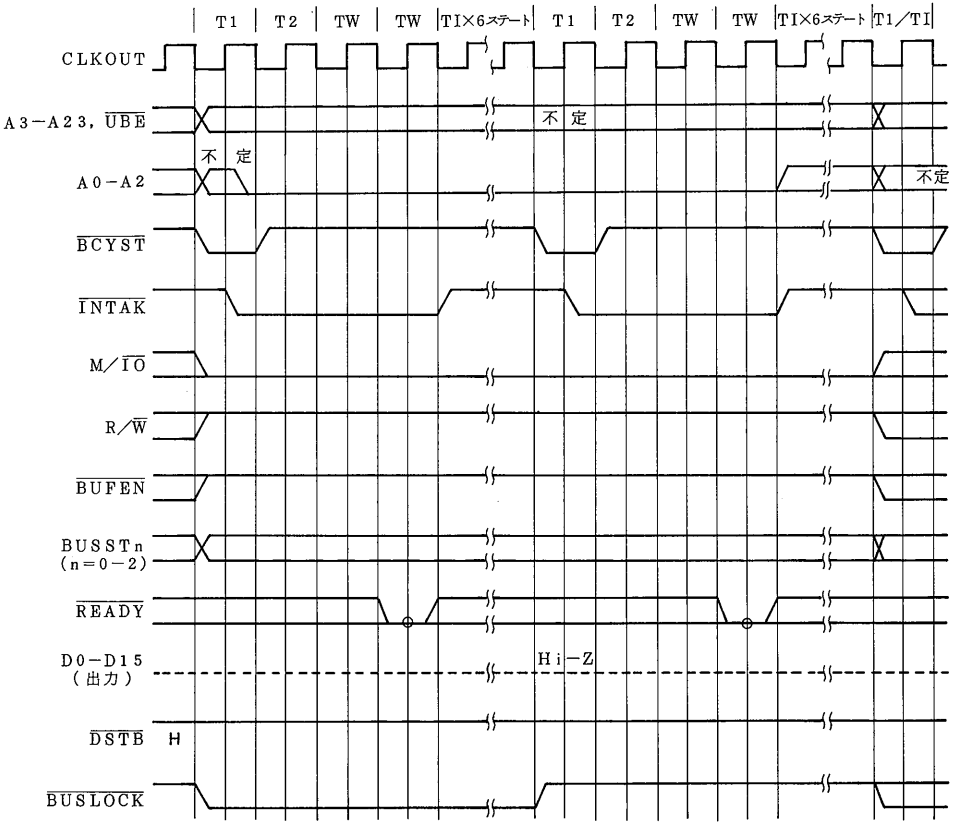
IISのクリアにより，受け付けた割り込みが終了したものと見なされ，保留してあった割り込みや新規に発生した割り込みを許可し実行できる状態とします．

図6－36　割り込みシーケンス



注　INTPnはその割り込み要求に対する最初の $\overline{\text{INTAK}}$ パルスが入力されるまではハイ・レベルに保つ必要があります．

図６－３７　割り込みアクノリッジ・サイクル（シングル・モード）



備考 1.　○印はサンプリングされるタイミングです．
2.　IORD，IOWR，MRD，MWRはすべてインアクティブです．

(2) 拡張モード

V53は外部からの割り込み要求を最大8本まで入力できますが，それ以上必要な場合は外部にμPD71059などの割り込みコントローラをカスケード接続してチャネル数を拡張することができます．

図6－38にμPD71059を2個カスケード接続したときの接続図を示します．スレーブの割り込みコントローラからのINT出力をICUのINTPnの1本に接続します．拡張モードのときはイニシャライズ時などのレジスタ/コマンドの設定がシングル・モードのときと異なってきますから注意して設定してください．

拡張モードでのICUの動作は2種類あります．1つはスレーブを通さない要求の場合ですが，これはシングル・モードの場合と同じ動作をします．もう1つはスレーブを通しての要求の場合でシングル・モードとは異なった動作をします．

スレーブを通しての要求の場合，CPUにベクタ番号を与えるのはICUではなくスレーブとなりますので，ICUはベクタ番号を出力すべきスレーブはどれであるかを指示します．これはA2－A0に出力されるスレーブ・アドレス信号により行います（**図6－40**参照）．

図6－39に拡張モードのV53のINTPn端子にμPD71059を接続した場合の，μPD71059のINTPiに発生した要求が受け付けられる過程を示します．

以下に，シングル・モードとの主な相違点を示します．

(a) 1回目の割り込みアクノリッジ・サイクル時に，A2－A0にスレーブ・アドレスを出力します．このスレーブ・アドレスはスレーブのμPD71059の接続されているチャネルに応じて出力されます．もちろんIIW3でスレーブの接続されているチャネルを正しく設定しなければなりません．

たとえば，チャネル5にスレーブが接続されている場合は，A2－A0に5Hというアドレスが出力されます．

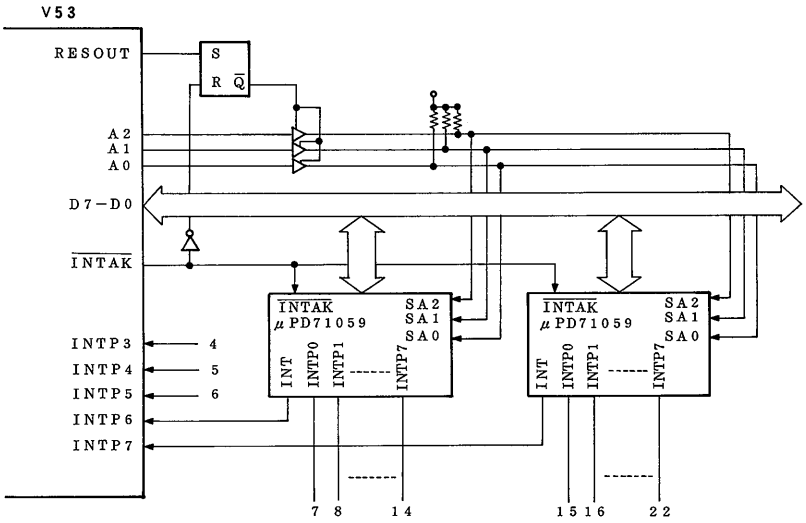
ただし，INTP0端子にはスレーブの割り込みコントローラを接続できません．

カスケード・チャネル	A2	A1	A0
INTP1	0	0	1
INTP2	0	1	0
INTP3	0	1	1
INTP4	1	0	0
INTP5	1	0	1
INTP6	1	1	0
INTP7	1	1	1

(b) 割り込みベクタはICUではなく，スレーブのμPD71059が下位8ビットのデータ・バスに出力します．

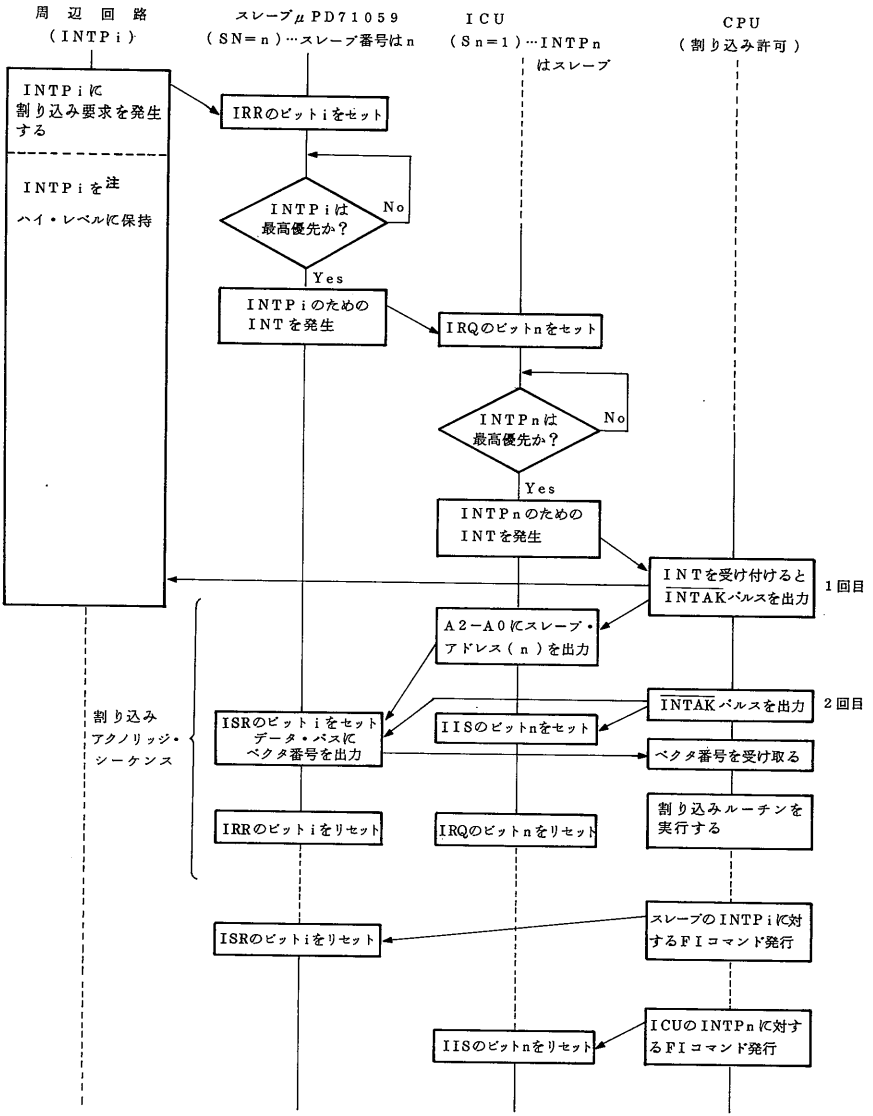
注意　拡張モードにおいても，スレーブのμPD71059の接続されていないチャネルに対するINPTnの割り込みシーケンスは，シングル・モードのときと同じです．

図6－38　拡張モードでのスレーブ接続例



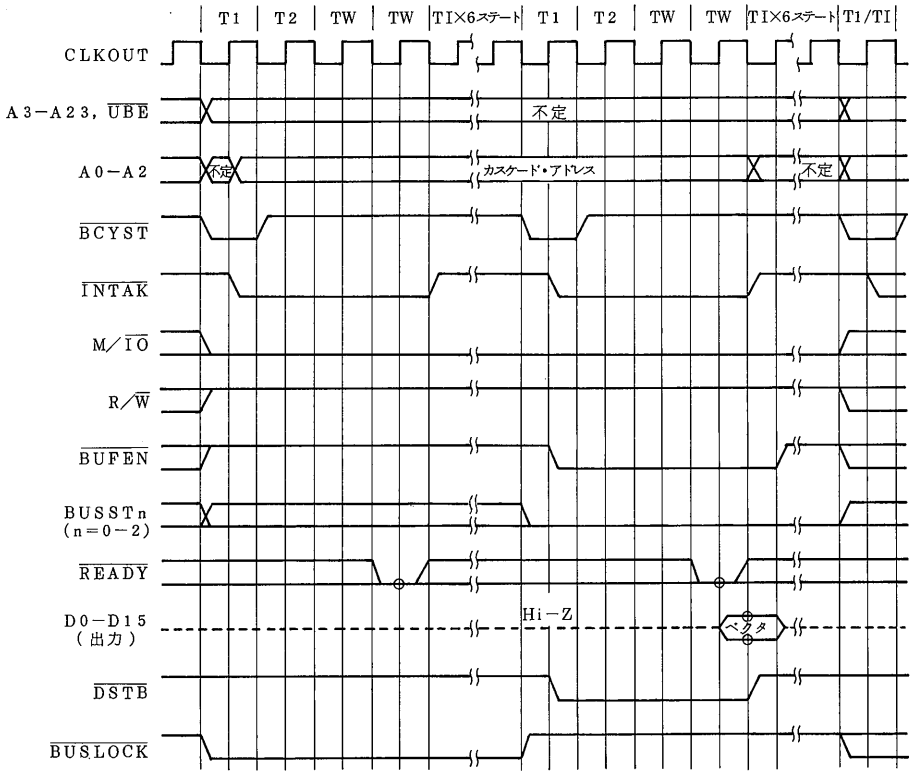
注意　この接続例では，アドレス・バスA0，A1，A2とスレーブのμPD71059のSA0，SA1，SA2のライン間に，3ステートのバッファが必要です．これはμPD71059のSA0，SA1，SA2は初期化されるまで入出力の状態が不定であり，アドレス・バスと直結すると，アドレス情報に何らかの影響を与える可能性があるからです．

図6－39　拡張モードでの割り込みシーケンス



注　INTPiはその割り込み要求に対する最初の $\overline{\text{INTAK}}$ パルスが入力されるまではハイ・レベルを保つ必要があります．

図６－４０　割り込みアクノリッジ・サイクル（拡張モード）



備考 1.　○印はサンプリングされるタイミングです．
2.　IORD，IOWR，MRD，MWRはすべてインアクティブです．

6．4．8　その他使用上の注意事項

(1)　INTPn端子

V53はスタンバイ・モード時の消費電流低減のため，INTPn端子はプルアップ抵抗を内蔵していません．μPD71059やV40，V50とは異なります．

(2)　不完全割り込み

これまで，INTPn入力はCPUからの割り込みアクノリッジ・パルス（$\overline{\text{INTAK}}$）が返るまで保持するように説明してきましたが，これはINTPnを途中でインアクティブにすると，割り込みがまったく受け付けられないか，あるいは不完全割り込みが発生するためです．

不完全割り込みとは，上述のような場合ICUがあたかもレベル7の割り込みがあったかのように動作するにもかかわらず，IISのビット7はセットされない状態を意味します．このため，不完全割り込みの発生する可能性のある場合は，レベル7の割り込み処理ルーチンに不完全割り込み処理ルーチンを入れる必要があります．不完全割り込みかどうかは，IISのビット7がセットされているかどうかで判断できます．また，不完全割り込み処理のときは，FIコマンドを発行しないでください．

6.5 DMAU(DMAコントロール・ユニット)

DMAUは4本のDMAチャネルをもち，2種類のLSIμPD71071，μPD71037の機能（サブセット）を提供します．

6.5.1 特　徴

- 2つの動作モード（μPD71071モード，μPD71037モード）
- 24ビット長のアドレス・レジスタ
- 16ビット長のカウント・レジスタ
- 4つの独立なDMAチャネル
- 4クロック/1バス・サイクル
- バイト転送/ワード転送選択可能
- 3種類の転送モード（チャネルごとに設定可能）
  シングル転送モード，ディマンド転送モード，ブロック転送モード
- 2種類のバス・モード（全チャネル共通；μPD71037モードではバス・リリース・モードのみ）
  バス・リリース・モード
  バス・ホールド・モード
- チャネルごとにDMA要求をマスク可能
- オート・イニシャライズ機能
- 転送アドレスのインクリメント/デクリメント
- 2種類のチャネル優先順位（固定優先順位/回転優先順位）
- 転送終了時の$\overline{TC}$出力
- $\overline{END}$入力によるサービスの強制終了
- カスケード接続可能
- I/O－メモリ，メモリーI/O転送可能

6.5.2 V50に対する変更箇所

- μPD71037動作モードの追加
- DMARQ3，$\overline{DMAAK3}$を直接端子に出力（マルチプレクスなし）
- アドレス・レジスタの拡張
  V50：20ビット
  V53：24ビット

6．5．3　DMAU内部ブロック図

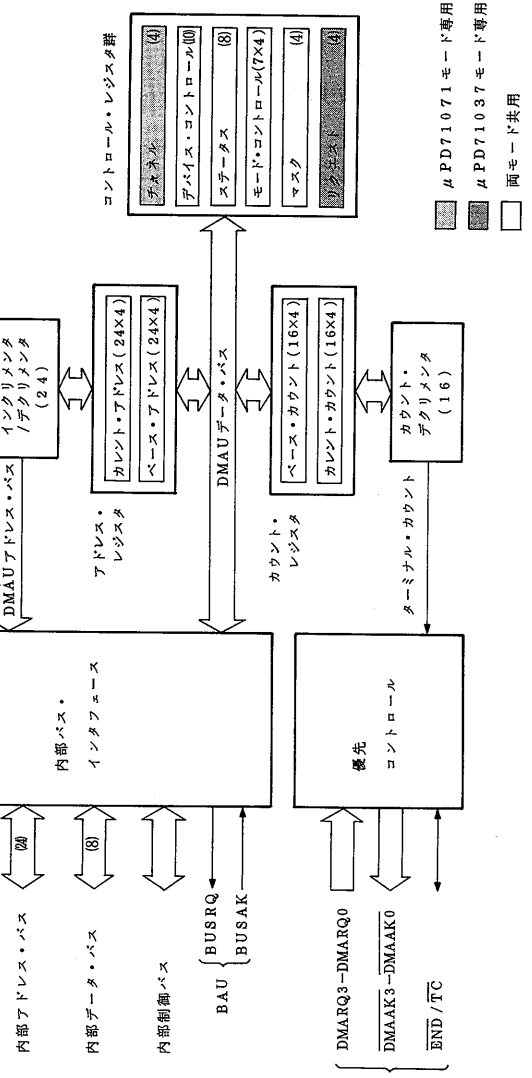


図6－41　DMAU内部ブロック図

6．5．4　μPD71071モードとμPD71071の相違点

以下の機能を除き，内蔵DMAUのμPD71071モードは，μPD71071と同等の機能を有します．

表6－6　μPD71071モードとμPD71071の違い

機　　能	μPD71071モード	μPD71071
ソフトウエア・リクエスト	なし	あり
メモリーメモリ転送	なし	あり
DMARQアクティブ・レベル	ハイ	ハイまたはロウ
DMAAKアクティブ・レベル	ロウ	ハイまたはロウ
バス・サイクル	4クロック	4または3クロック

6．5．5　μPD71037モードとμPD71037の相違点

以下の機能を除き，内蔵DMAUのμPD71037モードは，μPD71037と同等の機能を有します．

表6－7　μPD71037モードとμPD71037の違い

機　　能	μPD71037モード	μPD71037
メモリーメモリ転送	なし	あり
DMARQアクティブ・レベル	ハイ	ハイまたはロウ
DMAAKアクティブ・レベル	ロウ	ハイまたはロウ
バス・サイクル	4クロック	3または2クロック

6．5．6　μPD71037モードとμPD71071モードの相違点

各内部レジスタは，両モードで共通に使われます．ただし，チャネル・レジスタはμPD71071モード専用で，リクエスト・レジスタはμPD71037モード専用です（図6－41参照）．モードを切り替えた場合，各レジスタの内容はモード遷移後も保持されています．また，遷移後のモードでの未使用ビットは，モード遷移前の状態を保持しています．

μPD71037モードとμPD71071モードの相違点を，以下に示します．

(1) 転送の単位

μPD71071モードではバイト転送／ワード転送の選択ができますが，μPD71037モードでは常にバイト転送に固定されています．

(2) チャネル選択

μPD71037モードではチャネル・レジスタがないため，各コマンド発行時にチャネル指定をする方法を採っています．モード・コントロール・レジスタの場合は，書き込むデータの

一部でチャネルを選択します．また，アドレス・レジスタ，カウント・レジスタの場合では，アクセスするＩ／Ｏアドレスによってチャネルを選択します．

(3) ベース／カレント・レジスタのアクセス

各チャネルのアドレス・レジスタおよびカウント・レジスタは，おのおのベース・レジスタカレント・レジスタのペアで構成されています．モードによりアクセス対象が次のように異なります．

(a) μPD71071モード

チャネル・レジスタにより次のように切り替え可能

- リード時：カレント・レジスタのみ
- ライト時：ベース・レジスタ，カレント・レジスタ両方同時
- リード／ライト時：ベース・レジスタのみ

(b) μPD71037モード

- リード時：カレント・レジスタのみ
- ライト時：ベース・レジスタ，カレント・レジスタ両方同時

(4) ソフトウエアDMAリクエスト

μPD71037モードには，μPD71071モードに存在しないリクエスト・レジスタがあり，ソフトウエアでのDMA要求の発生ができます．

(5) バス・モード

μPD71071モードには，バス・リリース・モードとバス・ホールド・モードがあります．

μPD71037モードにはバス・モードの選択はなく，常にμPD71071モードのバス・リリース・モードでのみ動作します．

(6) DMAUロケーション・アドレス

μPD71071モードの場合，DMAUのＩ／Ｏアドレスは常に連続アドレスに配置されますが，μPD71037モードでは，システムＩ／Ｏ領域にあるＳＣＴＬレジスタによって連続アドレスまたは，奇数／偶数アドレスへの固定が選択できます．

(7) そ　の　他

基本的なコマンドは両モードで同じですが，μPD71071モードにあってμPD71037モードにはないコマンドや，その逆のものもあります．また，各コマンドのＩ／Ｏアドレスは，μPD71071モードとμPD71037モードとの間で互換性はありません．μPD71037モードにおける全コマンドは，バイト・タイプのＩＮ／ＯＵＴ命令で行わなければなりません．

6.5.7 μPD71071モード

システムＩ／Ｏ領域のＳＣＴＬレジスタ内のDMAMビットを“0”に設定することによって，DMAUはμPD71071モードに設定されます．また，リセット直後にもμPD71071モードに設定されます．

(1) アドレシング

DMAUの内部レジスタはシステムI/O領域で設定したOPHAとDULAおよびアドレス信号（A3－A0）によってアドレスされます．

DMAUは表6－8に示すように計24本のレジスタを有しています．このうち，DBA，DCA，DBC，DCC，DMDの各レジスタは各チャネルに1本ずつあるため，どのチャネルのレジスタをアクセスするかを，チャネル・レジスタ（DCH）であらかじめ設定してからアクセスします．

表6－8　DMAUレジスタ構成（μPD71071モード時）

レジスタ名	ビット・サイズ
チャネル・レジスタ（DCH）	5ビット
ベース・アドレス・レジスタ（DBA）	24ビット×4
カレント・アドレス・レジスタ（DCA）	24ビット×4
ベース・カウント・レジスタ（DBC）	16ビット×4
カレント・カウント・レジスタ（DCC）	16ビット×4
モード・コントロール・レジスタ（DMD）	7ビット×4
デバイス・コントロール・レジスタ（DDC）	5ビット
ステータス・レジスタ（DST）	8ビット
マスク・レジスタ（DMK）	4ビット

表6－9　DMAU内部レジスタのアドレス（μPD71071モード時）

上位アドレス	下位アドレス		レジスタ	操作
A15－A8 (OPHA)	A7 A6 A5 A4 (DULA)	0 0 0 0	DICM	ライト
		0 0 0 1	DCH	リード/ライト
		0 0 1 0	DBC/DCC（下位バイト）	リード/ライト
		0 0 1 1	DBC/DCC（上位バイト）	リード/ライト
		0 1 0 0	DBA/DCA（下位バイト）	リード/ライト
		0 1 0 1	DBA/DCA（中位バイト）	リード/ライト
		0 1 1 0	DBA/DCA（上位バイト）	リード/ライト
		0 1 1 1	予約	－
		1 0 0 0	DDC（下位バイト）	リード/ライト
		1 0 0 1	DDC（上位バイト）	リード/ライト
		1 0 1 0	DMD	リード/ライト
		1 0 1 1	DST	リード
		1 1 0 0	予約	－
		1 1 0 1	予約	－
		1 1 1 0	予約	－
		1 1 1 1	DMK	リード/ライト

備考　IOAGビットの値には影響されません．

(2) μPD71071モード時のコマンド

DMAUのレジスタをリード/ライトするコマンドの発行は，システムI/O領域で設定したアドレスに対するI/O入出力命令で行い，コマンドの選択はA3－A0の4ビットで行います．コマンドのアドレスとはA3－A0の値のことを指します．

表6－10　DMAUコマンド・アドレス（μPD71071モード時）

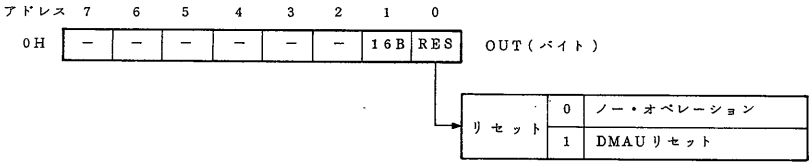
アドレス	操　作	コ　マ　ン　ド
0H	W(B)	イニシャライズ（DICM）
1H	R(B)	チャネル・レジスタ・リード（DCH）
	W(B)	チャネル・レジスタ・ライト（DCH）
2H	R/W	カウント・レジスタ・リード/ライト
3H	R/W	（DBC/DCC）
4H	R/W	アドレス・レジスタ・リード/ライト
5H	R/W	（DBA/DCA）
6H	R/W(B)	
8H	R/W	デバイス・コントロール・レジスタ・リード/ライト
9H	R/W	（DDC）
0AH	R/W(B)	モード・コントロール・レジスタ・リード/ライト（DMD）
0BH	R(B)	ステータス・レジスタ・リード（DST）
0FH	R/W(B)	マスク・レジスタ・リード/ライト（DMK）

注意　（B）の付いているコマンドは，バイトIN/OUT命令で行うこと，
この表に示されているアドレスと操作の組み合わせ以外は禁止．

(3) 初　期　化

DMAUは，リセット信号（$\overline{\text{RESET}}$）によって表6－11のように初期化されます．また，下記のイニシャライズ・コマンド（DICM）の発行によっても同様に初期化されます．すなわち，ハードウエア，ソフトウエアのどちらでも初期化できます．

図6－42　イニシャライズ・コマンド・フォーマット



RESビットをセットすると，DMAUの内部レジスタが表6－11のように初期化されます．初期化終了後，RESビットは自動的にクリアされます．

μPD71071では，16Bビットにより8ビット・データ・バス，16ビット・データ・バスの選択を行いますが，V53では，16Bビットは0，1のどちらでも常に16ビット・データ・バスとなります．　なお，データ転送単位（バイト転送/ワード転送）は，μPD71071同様モード・コントロール・レジスタのW/$\overline{\text{B}}$ビットにより設定できます．

表6－11　リセットによるレジスタの初期化

レジスタ名	初期化内容
アドレス・レジスタ	変化なし
カウント・レジスタ	変化なし
チャネル・レジスタ	7 6 5 4 3 2 1 0 － － － 0 0 0 0 1　（CH0選択）
モード・コントロール・レジスタ	全ビット・クリア
デバイス・コントロール・レジスタ	全ビット・クリア
ステータス・レジスタ	全ビット・クリア
マスク・レジスタ	全ビット・セット（全チャネル・マスク）

(4) レジスタ／コマンド

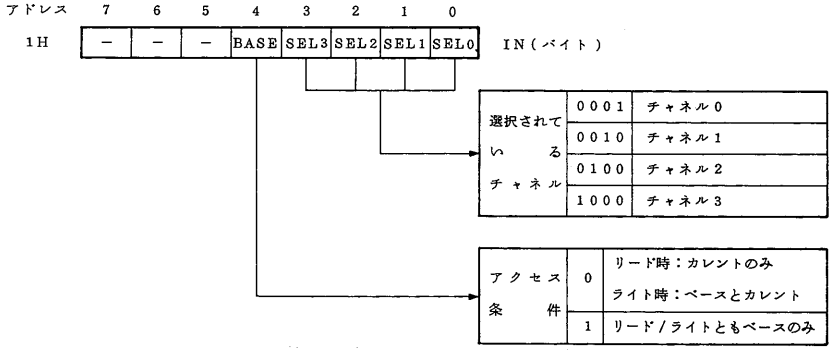
(a) DCH（DMAチャネル・レジスタ）

このレジスタは，4本あるDMAチャネルの中からCPUによるプログラミングの対象となる1本のチャネルを選択します．すなわち，各チャネルがそれぞれ有しているアドレス・レジスタ，カウント・レジスタ，モード・コントロール・レジスタを設定する直前にこのチャネル・レジスタを設定して，どのチャネルのレジスタに対してプログラミングするかを指定します．

このレジスタは，読み出し時と書き込み時でフォーマットが異なります．

(i) DCH読み出し時

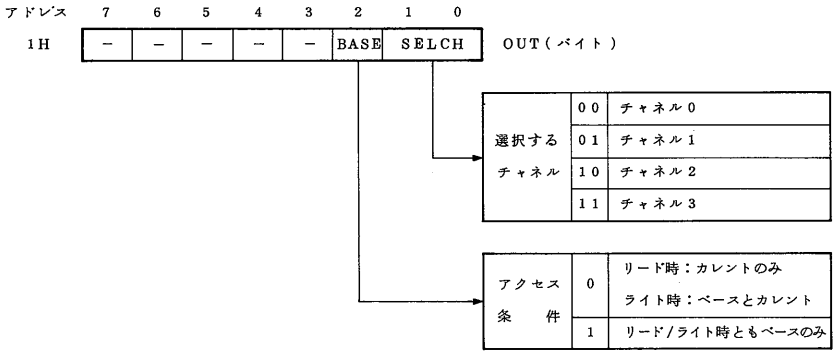
図6－43　チャネル・レジスタ・リード・コマンド・フォーマット



・BASEビットは，アドレス／カウント・レジスタに対するリード／ライト時のアクセス対象を示します．セットされていれば，読み出し／書き込みともベース・レジスタが選択されます．また，クリアされていれば，読み出し時はカレント・レジスタが，書き込み時はベース／カレント・レジスタ両方が選択されます．
・SELビットは，現在どのチャネルが選択されているかを示します．

(ii) DCH書き込み時

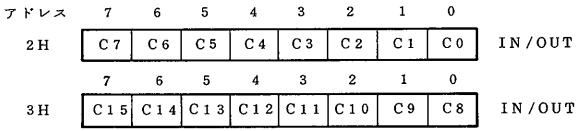
図6－44 チャネル・レジスタ・ライト・コマンド・フォーマット



・BASEビットは，DCHリード時と同様に，SELCHビットで選択するチャネルのアドレス／カウント・レジスタに対するアクセス対象を指定します．セットされていれば，読み出し／書き込みともベース・レジスタが選択されます．また，クリアされていれば，読み出し時はカレント・レジスタが，書き込み時はベース／カレント・レジスタ両方が選択されます．
・SELCHビットは，どのチャネルを選択するかを設定します．

(b) DBC／DCC（DMAベース／カレント・カウント・レジスタ）

図6－45 カウント・レジスタ・リード／ライト・コマンド・フォーマット



カウント・レジスタは，各チャネルごとにベース・カウント・レジスタとカレント・カウント・レジスタの2組のレジスタで構成されています．

ベース・カウント・レジスタ／カレント・カウント・レジスタとも同じアドレスに割り付けられています．リード／ライト時のベース／カレント・レジスタの選択はチャネル・レジスタで行います．

カレント・カウント・レジスタは，1回のDMA転送ごとにデクリメント（－1）されます．

ベース・カウント・レジスタは，新たなカウント値が設定されるまで以前に設定された値を保持し，オートイニシャライズ（(5)項参照）時にはこの値がカレント・カウント・レジスタに転送されます．

また，このレジスタにはワード・タイプのIN/OUT命令でリード／ライトを行えます．

(c) DBA／DCA（DMAベース／カレント・アドレス・レジスタ）

図6－46 アドレス・レジスタ・リード／ライト・コマンド・フォーマット

アドレス	7	6	5	4	3	2	1	0	
4H	A7	A6	A5	A4	A3	A2	A1	A0	IN/OUT
	7	6	5	4	3	2	1	0	
5H	A15	A14	A13	A12	A11	A10	A9	A8	IN/OUT
	7	6	5	4	3	2	1	0	
6H	A23	A22	A21	A20	A19	A18	A17	A16	IN/OUT（バイト）

アドレス・レジスタは，各チャネルごとにベース・アドレス・レジスタとカレント・アドレス・レジスタの2組のレジスタで構成されています．

ベース・アドレス・レジスタ／カレント・アドレス・レジスタとも同じアドレスに割り付けられています．リード／ライト時のベース／カレント・レジスタの選択はチャネル・レジスタで行います．

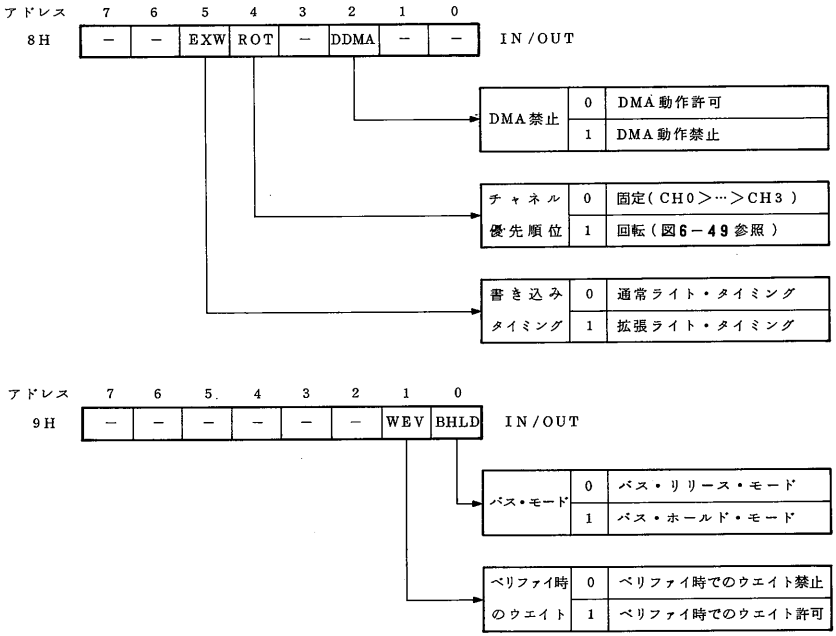
カレント・アドレス・レジスタは，1回のDMA転送ごとに更新されます（ワード転送時は±2，バイト転送時は±1）．

ベース・アドレス・レジスタは，新たなカウント値が設定されるまで以前に設定された値を保持し，オートイニシャライズ時にはこの値がカレント・アドレス・レジスタに転送されます．

このレジスタに対するリード／ライトは，下位2バイト（アドレス4H，5H）に対してはワード・タイプのIN/OUT命令で行えますが，上位1バイト（アドレス6H）に対してはバイト・タイプのIN/OUT命令で行ってください．

(d) DDC(DMAデバイス・コントロール・レジスタ)

図6－47　デバイス・コントロール・レジスタ・リード／ライト・コマンド・フォーマット

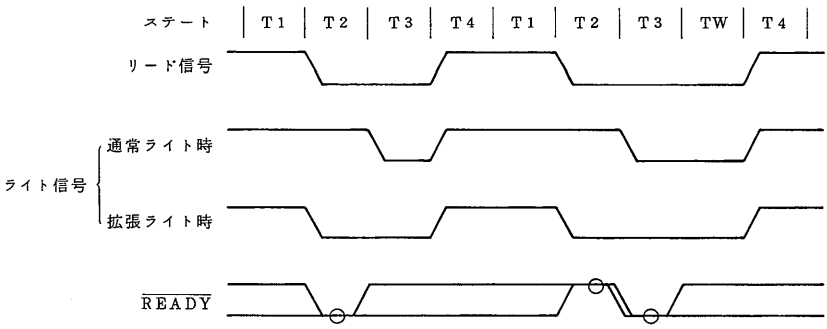


このレジスタは，全チャネルに共通のモードや，DMA動作の許可／禁止などの制御を行います．このレジスタには，ワード・タイプのIN/OUT命令でリード/ライトを行えます．

(i)　EXWビットは図6－48に示すようにライト信号の出力タイミングを設定します．

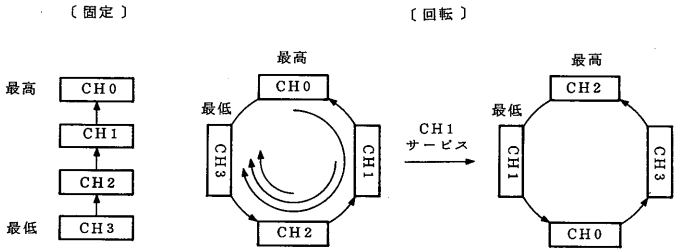
・通常ライト時：T3，TWの間アクティブ
・拡張ライト時：T2，T3，TWの間アクティブ

図６－４８　拡張ライト・タイミング



(ii) ROTビットをセットすると回転優先順位になります．図6－49に示すように固定優先順位では常にチャネル0が最高優先で，チャネル3が最低優先ですが，回転優先順位ではサービスの終了したチャネルが最低優先となり，特定のチャネルによるサービスの独占を防ぎます．

図6－49　DMA優先順位



(iii) DDMAビットをセットすると，DMA動作が一時的に停止します．内部的には，DDMA＝“1”の間BAUに対するバス要求信号を出力しません．その後再びDDMA＝“0”としてDMA動作を許可状態にすると，DMA動作を禁止する前の状態でDMA動作を再開できます．DMA動作禁止状態でDMAUをプログラミングすることも可能です．

(iv) WEVビットはベリファイ転送時に外部$\overline{\text{READY}}$信号，およびプログラマブル・ウエイトによるウエイト・ステートの挿入の許可／禁止を制御します．

(v) BHLDビットはDMA転送のバス・モードを設定します．バス・モードは，バス使用権の獲得／返還に関するモードです．

・バス・リリース・モード

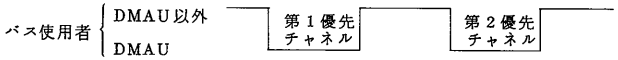
バス・リリース・モードでは，1回のサービス終了ごとにバス使用権をCPUに返還します．このため，複数のDMA要求が同時に発生する場合でも一度バス使用権を返還するため，最初のサービスと次のサービスの間には他のバス・サイクルが割り込むことができます．このように1つのDMAサイクルが終了するごとにバス使用権を返還するため，DMA要求に対する応答が遅くなります．

・バス・ホールド・モード

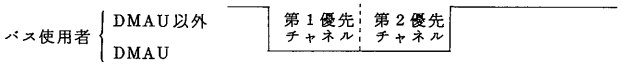
バス・ホールド・モードでは，1回のサービスが終了してもまだほかにDMA要求があれば，バス使用権を保持したまま引き続いてサービスを行うことができます．ただし，同じチャネルが連続してサービスされることはありません．このように，バス・ホールド・モードはバス・リリース・モードに比較し，効率の良い転送ができます．

図6－50　バス・モードによるバス使用権の違い

(a)　バス・リリース・モード



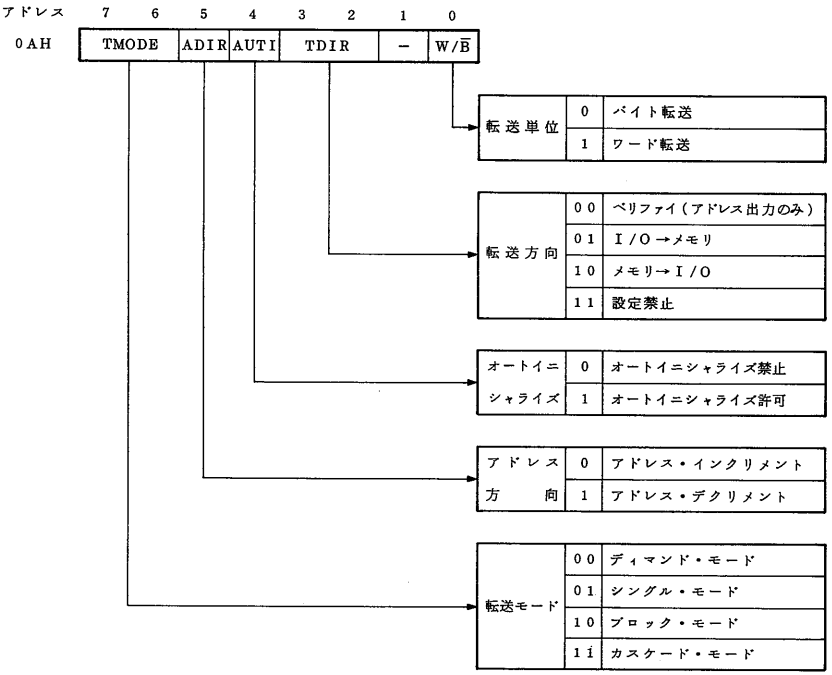
(b)　バス・ホールド・モード



備考　要求は2本あったものとします．

(e)　DMD（DMAモード・コントロール・レジスタ）

図6－51　モード・コントロール・レジスタ・リード／ライト・コマンド・フォーマット



このレジスタによって，各チャネルごとの動作モードを設定します．

(i) TMODEビットは，メモリーI/O間のDMA転送モードを指定します．

各転送モードについては6.5.9項を，カスケード接続については（5）項を参照してください．

(ii) ADIRビットはカレント・アドレス・レジスタの更新方向を指定します．

・アドレス・インクリメント

ワード転送時：＋2，バイト転送時：＋1

・アドレス・デクリメント

ワード転送時：－2，バイト転送時：－1

(iii) AUTIビットをセットすることによって，オートイニシャライズ機能が有効になります．

オートイニシャライズ機能については，6.5.10項を参照してください．

(iv) TDIRビットはメモリーI/O間の転送方向を指定します．転送方向には次の3種類があります．

・リード転送……メモリ→I/O

$\overline{MRD}$ と $\overline{IOWR}$ を同時に出力します．

・ライト転送……I/O→メモリ

$\overline{IORD}$ と $\overline{MWR}$ を同時に出力します．

・ベリファイ転送……転送を行いません．

アドレスを出力するだけで，リード/ライト制御信号は出力しません．

フロッピィ・ディスクのCRCチェックなど，DMAAK信号のみ必要なベリファイ動作に使用します．

(v) W/$\overline{B}$ビットは，DMA転送をワード単位で行うか，バイト単位で行うかを指定します．

ワード転送の場合，偶数番地から始まる2バイトを1ワードとして扱い，16ビット・データ・バスを用いて1回で転送します．転送アドレスが奇数の場合は，自動的にアドレスを1だけデクリメントしてからワード転送を行うため，最後の1バイトが転送できなくなります．したがって，転送開始アドレスは必ず偶数番地に設定してください．

表6－12にバイト/ワード転送時のカレント・カウント・レジスタ，カレント・アドレス・レジスタの更新を，表6－13にDMA転送時のA0，$\overline{UBE}$ 信号の状態を示します．

表6－12　カレント・レジスタの更新

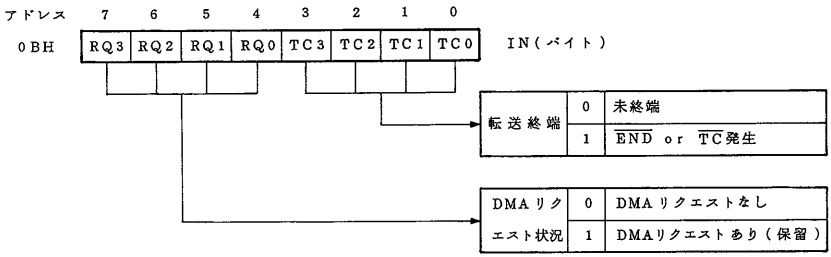
レジスタ	バイト転送	ワード転送
カレント・アドレス	± 1	± 2
カレント・カウント	－ 1	－ 1

表6－13　DMA動作時のA0，$\overline{UBE}$

バイト/ワード	A0	$\overline{UBE}$
バイト	0	1
	1	0
ワード	0	0

(f)　DST（DMAステータス・レジスタ）

図6－52　ステータス・レジスタ・リード・コマンド・フォーマット



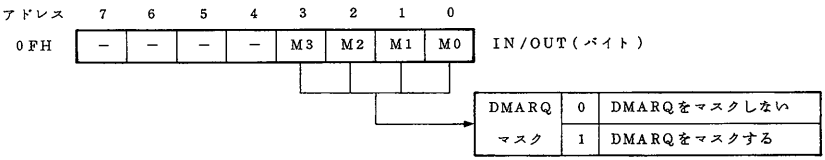
このレジスタは各チャネルのDMA要求の状態，$\overline{TC}$ または $\overline{END}$入力の発生状態を保持しています．

(i)　RQ3－RQ0ビットは各チャネルのDMAリクエストの状態を示します．マスク・レジスタによるマスク状態に関係なく，DMARQ端子の状態（アクティブならば"1"）を示します．そのため，マスクによって現在受け付け保留中のDMA要求を調べることができます．

(ii)　TC3－TC0ビットは，各チャネルの $\overline{TC}$ または $\overline{END}$ 入力の発生状態を示します．これらのビットはこのレジスタを読み出すごとに自動的にクリアされます．

(g)　DMK（DMAマスク・レジスタ）

図6－53　マスク・レジスタ・リード／ライト・コマンド・フォーマット



このレジスタをセットすることによって，任意のチャネルのDMA要求をマスクすることができます．

また，ターミナル・カウント発生時および$\overline{\text{END}}$入力による強制終了時にも，該当チャネルのマスク・ビットがセットされます．ただし，オートイニシャライズが設定されているチャネルでは，マスク・ビットはセットされません．

(5) カスケード接続

DMAUは4本の独立なDMAチャネルを内蔵していますが，それ以上にDMAチャネルを増設したいときは，外部にμPD71071などのDMAコントローラをカスケード接続することができます．

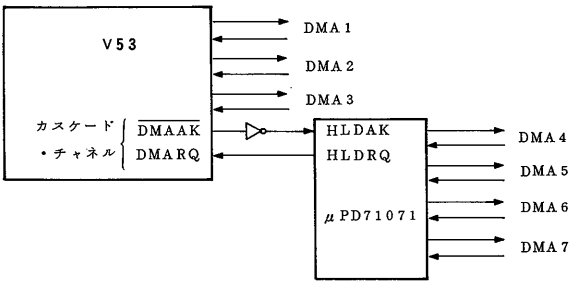
μPD71071をカスケード接続する場合は以下のようにします．

(a) μPD71071のHLDRQ，HLDAK端子をV53の任意のチャネルのDMARQn，$\overline{\text{DMAAK}}$n端子に接続します（図6－54参照）．

(b) μPD71071の接続されたチャネルを，DMDレジスタでカスケード・モードに設定します．

また，カスケード・モードに設定されたチャネルのDMAサービス中はDDCレジスタの設定に関係なく，常にバス・リリース・モードとして動作します．すなわち，カスケード・チャネルのサービス中はほかのチャネルのDMA要求が割り込むことができません．ただし，カスケード・チャネルのサービスが終了すると，再びDDCの設定によるバス・モードに戻ります．

図6－54　カスケード接続例



6．5．8　μPD71037モード

システムI／O領域のSCTLレジスタ内のDMAMビットを“1”に設定することによって，DMAUはμPD71037モードに設定されます．

リセット直後は，μPD71037モードではなくμPD71071モードに設定されています．

(1) アドレシング

μPD71037モード時のDMAU内部レジスタは，システムI／O領域で設定したOPHAとDULAおよびアドレス信号（A3－A0またはA4－A1）によってアドレスされます．μPD71037モード時の内部レジスタ・アドレスを表6－14に示します．

また，μPD71037モードでは，各コマンド発行時に設定対象のチャネルを指定するため，チャネル・レジスタ（DCH）はありません．

表6－14　DMAU内部レジスタのアドレス（μPD71037モード時）

	上位アドレス	下位アドレス		レジスタ，コマンド	操作
IOAG＝1 のとき	A15－A8 （OPHA）	A7 A6 A5 A4	0 0 0 0	アドレス・レジスタ（チャネル0）	リード／ライト
			0 0 1 0	〃 （チャネル1）	リード／ライト
			0 1 0 0	〃 （チャネル2）	リード／ライト
			0 1 1 0	〃 （チャネル3）	リード／ライト
			0 0 0 1	カウント・レジスタ（チャネル0）	リード／ライト
			0 0 1 1	〃 （チャネル1）	リード／ライト
		（DULA）	0 1 0 1	〃 （チャネル2）	リード／ライト
			0 1 1 1	〃 （チャネル3）	リード／ライト
			1 0 0 0	ステータス・レジスタ／コマンド・レジスタ	リード／ライト
			1 0 0 1	リクエスト・レジスタ	ライト
			1 0 1 0	シングル・マスク・レジスタ	ライト
			1 0 1 1	モード・レジスタ	ライト
			1 1 0 0	クリア・バイト・ポインタF／F	ライト
			1 1 0 1	イニシャライズ	ライト
			1 1 1 0	クリア・マスク・レジスタ	ライト
			1 1 1 1	オール・マスク・レジスタ	ライト
IOAG＝0 のとき	A15－A8 （OPHA）	A7 A6 A5	0 0 0 0	アドレス・レジスタ（チャネル0）	リード／ライト
			0 0 1 0	〃 （チャネル1）	リード／ライト
			0 1 0 0	〃 （チャネル2）	リード／ライト
			0 1 1 0	〃 （チャネル3）	リード／ライト
			0 0 0 1	カウント・レジスタ（チャネル0）	リード／ライト
			0 0 1 1	〃 （チャネル1）	リード／ライト
			0 1 0 1	〃 （チャネル2）	リード／ライト
		（DULA）	0 1 1 1 A0	〃 （チャネル3）	リード／ライト
			1 0 0 0	ステータス・レジスタ／コマンド・レジスタ	リード／ライト
			1 0 0 1	リクエスト・レジスタ	ライト
			1 0 1 0	シングル・マスク・レジスタ	ライト
			1 0 1 1	モード・レジスタ	ライト
			1 1 0 0	クリア・バイト・ポインタF／F	ライト
			1 1 0 1	イニシャライズ	ライト
			1 1 1 0	クリア・マスク・レジスタ	ライト
			1 1 1 1	オール・マスク・レジスタ	ライト
—	1 1 1 1 1 1 1 1	1 1 1 0 0 0 0 0		BSEL	リード／ライト
	1 1 1 1 1 1 1 1	1 1 1 0 0 0 0 1		BADR	リード／ライト
IOAG＝1 のとき	A15－A8 （OPHA）	A7 A6 A5 A4 A3 A2 （BADR）	BNKn (BSEL)	バンク・レジスタ（チャネルn）	リード／ライト
IOAG＝0 のとき	A15－A8 （OPHA）	A7 A6 A5 A4 A3 （BADR）	BNKn (BSEL) A0	バンク・レジスタ（チャネルn）	リード／ライト

備考 1.　BSEL，BADRのアドレスはIOAG，OPHA，DULAに関係なくFFE0H，FFE1Hです．

2.　n＝0－3です．

(2) μPD71037モード・コマンド一覧

表6－15　μPD71037モード・コマンド一覧

A3 (A4)	A2 (A3)	A1 (A2)	A0 (A1)	操作	F/F注2	コマンド名	略号
0	channel注1		0	リード	0	リード・カレント・アドレス・レジスタ（下位バイト）	DRCA0－DRCA3
					1	リード・カレント・アドレス・レジスタ（上位バイト）	
				ライト	0	ライト・ベース＆カレント・アドレス・レジスタ（下位バイト）	DWBCA0－DWBCA3
					1	ライト・ベース＆カレント・アドレス・レジスタ（上位バイト）	
0	channel注1		1	リード	0	リード・カレント・カウント・レジスタ（下位バイト）	DRCC0－DRCC3
					1	リード・カレント・カウント・レジスタ（上位バイト）	
				ライト	0	ライト・ベース＆カレント・カウント・レジスタ（下位バイト）	DWBCC0－DWBCC3
					1	ライト・ベース＆カレント・カウント・レジスタ（上位バイト）	
1	0	0	0	リード		リード・ステータス・レジスタ	DRST
				ライト		ライト・コマンド・レジスタ	DWC
1	0	0	1	ライト		ライト・リクエスト・レジスタ	DWRQ
1	0	1	0	ライト		ライト・シングル・マスク・レジスタ	DWSM
1	0	1	1	ライト		ライト・モード・レジスタ	DWM
1	1	0	0	ライト		クリア・バイト・ポインタF／F	DCBP
1	1	0	1	ライト		イニシャライズ	DINT
1	1	1	0	ライト		クリア・マスク・レジスタ	DCM
1	1	1	1	ライト		ライト・オール・マスク・レジスタ	DWAM

注1. channel＝00　なら　チャネル0選択
channel＝01　なら　チャネル1選択
channel＝10　なら　チャネル2選択
channel＝11　なら　チャネル3選択

2. F／Fはバイト・ポインタF／Fのことです．
バイト・ポインタF／Fとは，アドレス・レジスタ，カウント・レジスタをアクセスするときに，上位／下位バイトを指定するフリップ・フロップです．
リセット，およびクリア・バイト・ポインタF／Fコマンドの実行で“0”になり，アドレス・レジスタ，カウント・レジスタのリード／ライトを行うたびに反転します．したがって，アドレス・レジスタ，カウント・レジスタをリード／ライトするときは，下位，上位と続けて行ってください．

(3) μPD71037モード・コマンドの詳細

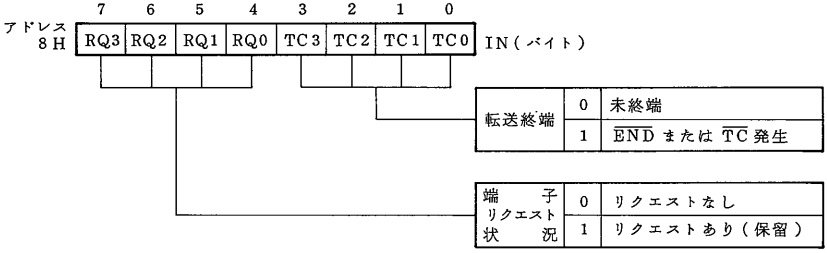
以下に各制御コマンドの詳細なフォーマットを示します．

なお，アドレスとは，SCTLレジスタのIOAGビットが“1”のときのA3−A0，またはIOAGビットが“0”のときのA4−A1のことです．

(a) リード・ステータス・レジスタ

各チャネルについて，DMA要求や，ターミナル・カウントの発生状態，および$\overline{\text{END}}$入力の有無などの情報を保持しています．

図6−55 リード・ステータス・レジスタ



(i) RQ3−RQ0

DMA要求（DMARQ端子入力）の状態を示します．たとえ該当チャネルがマスクされていても，同じチャネルのDMARQ入力がアクティブであるかぎりこれらのビットはセットされます．これらのビットをサンプリングすることで，マスクにより保留されているDMAリクエストの検出が行えます．

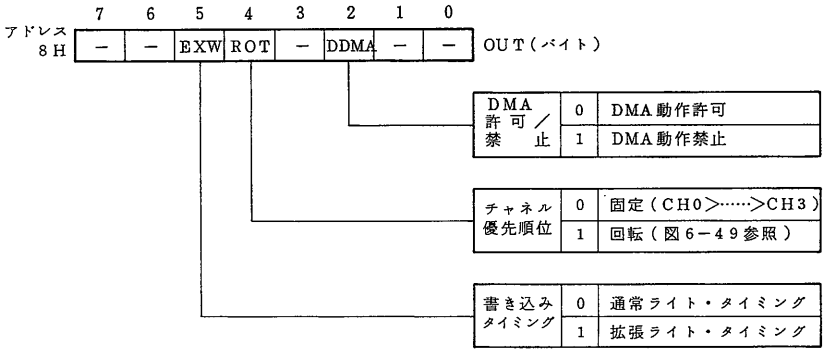
(ii) TC3−TC0

これらのビットは，各チャネルにおける$\overline{\text{END}}$入力およびターミナル・カウントの発生状態を示します．チャネル内部でターミナル・カウントが発生するか，あるいは外部から$\overline{\text{END}}$入力があれば該当チャネルのビットがセットされます．ステータス・リード・レジスタを読み出すごとに，これらビットはすべてリセットされます．

(b) ライト・コマンド・レジスタ

DMA転送の許可／禁止，優先順位決定，ライト・タイミングを決定します．

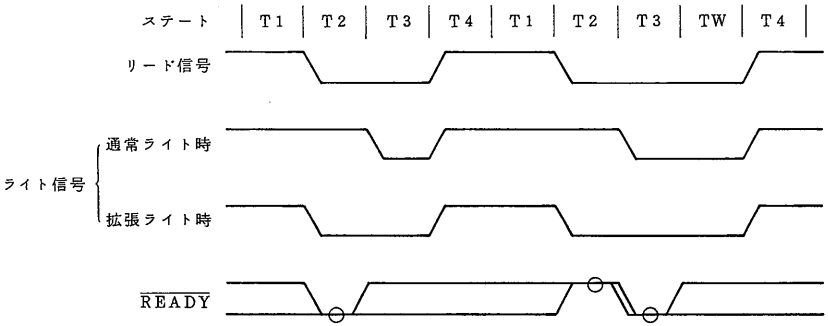
図6－56 ライト・コマンド・レジスタ



(i) EXW

このビットがセットされていると，DMAUはライト信号の出力をリード信号と同じタイミングで行います（拡張ライト）．図6－57に拡張ライト・タイミングを示します．

図6－57 拡張ライト・タイミング



(ii) ROT

このビットをセットすると，DMAチャネルの優先順位の決定を回転ネスト・モードで行います．

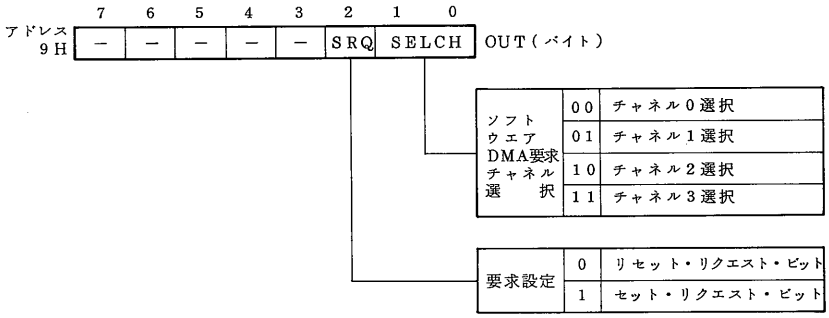
(iii) DDMA

DMA動作を禁止します．このビットがセットされている間，DMAUは有効なDMA要求を受け付けていてもホストCPUに対するHLDRQ信号を出力しません．DDMAビットは，DMAUのプログラミング中に誤ったDMA転送が行われるのを防止します．

(c) ライト・リクエスト・レジスタ

ソフトウエアDMAリクエストの制御を指定します.

図6－58 ライト・リクエスト・レジスタ



(i) SRQ

ビット0，1で選択されるチャネルについて，ソフトウエアDMAリクエストの制御を行います．1で該当チャネルのリクエスト・ビットをセット，0でリセットします．

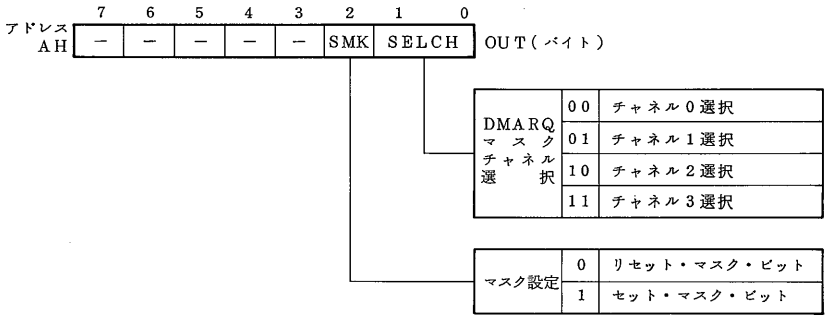
(ii) SELCH

ソフトウエアDMA要求制御を行う対象となるチャネルを指定します．

(d) ライト・シングル・マスク・レジスタ

各チャネルのマスクのセット，リセットを制御します．

図6－59 ライト・シングル・マスク・レジスタ



(i) SMK

ビット0，1で指定されたチャネルについてDMA要求（DMARQ端子入力）をマスクします．1ならマスクをセット，0の場合はマスクを解除します．

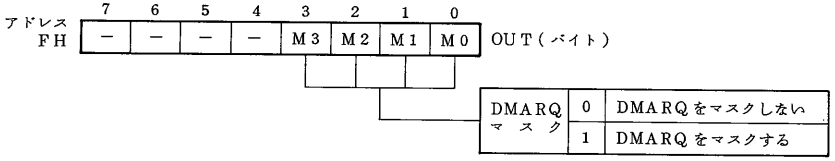
(ii) SELCH

DMAリクエスト・マスクの対象となるチャネルを選択します．

(e) ライト・オール・マスク・レジスタ

全チャネルのマスクのセット，リセットを指定します．

図6－60 ライト・オール・マスク・レジスタ

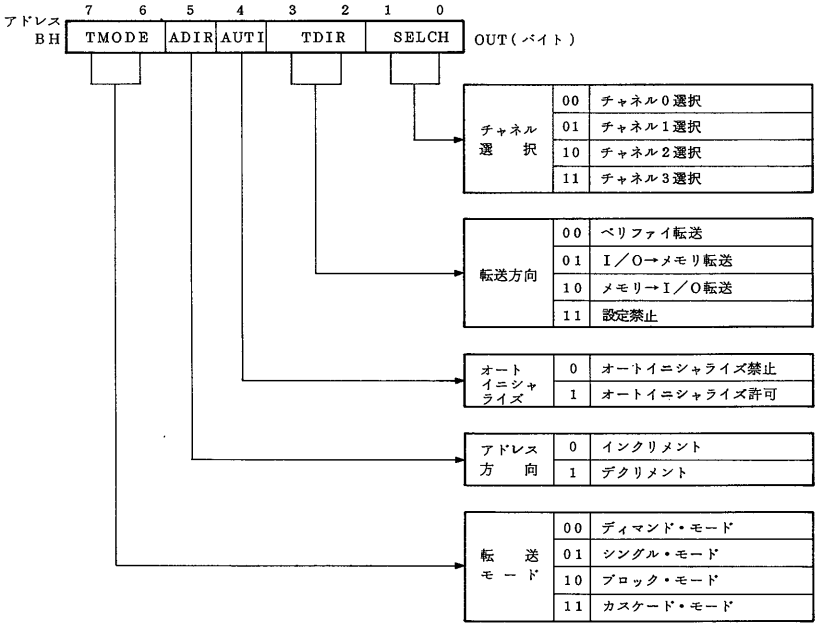


M3－M0ビットは4つのDMAチャネルについて，マスクのセット／リセットを指定します．1に設定すると該当チャネルのマスクがセットされ，そのチャネルのDMA要求（DMARQ端子入力）は禁止されます．0の場合はマスクを解除します．

(f) ライト・モード・レジスタ

各チャネルごとに，DMA転送のモードを指定します．

図6－61　ライト・モード・レジスタ



(i)　TMODE

DMA転送モードを指定します．

各転送モードについては，**6.5.9**項を参照してください．

また，外部にDMAコントローラをカスケード接続する場合は，カスケード・モードに指定してください．

(ii)　ADIR

カレント・アドレス・レジスタの値を更新する際，インクリメントするかデクリメントするかを指定します．0ならば1バイト転送するごとに1つインクリメント，1ならば1つデクリメントします．

(iii)　AUTI

このビットをセットすることでオートイニシャライズを指定します．オートイニシャライズ機能については，**6.5.10**項を参照してください．

(iv)　TDIR

転送方向を指定します．

(v)　SELCH

ビット7－2でモード指定を行う対象となるDMAチャネルを指定します．

(g) クリア・バイト・ポインタF/F，イニシャライズ，クリア・マスク・レジスタ

クリア・バイト・ポインタF/F，イニシャライズ，クリア・マスク・レジスタの3つのコマンドでは，書き込むデータは意味を持ちません．どんな値でもバイト・タイプのOUT命令で書き込めば，そのコマンドは有効になります．

コ　マ　ン　ド	アドレス	機　　能
クリア・バイト・ポインタF/F	CH	バイト・ポインタF/Fをクリアします．アドレス/カウント・レジスタをアクセスする際は，このコマンドを発行してから行ってください．
イニシャライズ	DH	DMAUを初期化します（$\overline{\text{RESET}}$入力時と同じ状態になります）．
クリア・マスク・レジスタ	EH	全チャネルのマスクをクリアし，DMA要求の受け付けを許可します．

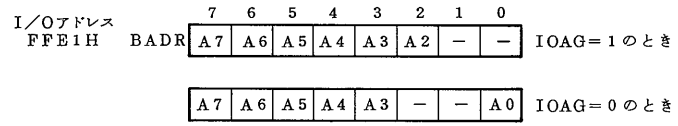
(4) バンク・レジスタ

V53は，μPD71037モード時にチャネル3－チャネル0の拡張DMAアドレスA23－A16を設定するバンク・レジスタ（BNKR3－BNKR0）を内蔵しています．μPD71037モードで，このレジスタを読み出し/書き込みする際のI/Oアドレスは，システムI/O領域内のOPHAとバンク・アドレス・レジスタ（BADR）で行います．このバンク・レジスタはリロケーション可能なレジスタです．

(a) バンク・レジスタのI/Oアドレス

バンク・レジスタ（BNKR3－BNKR0）のI/Oアドレスの上位8ビット（A15－A8）は，OPHAにより設定され，下位8ビット（A7－A0）はBADRで設定されます．ほかの内蔵ペリフェラル・リロケーション・レジスタと同様に，SCTL内のIOAGビットの値によって設定可能なビットが変わります．

図6－62　バンク・アドレス・レジスタ



(b) バンク選択レジスタ

OPHAとBADRの各レジスタにより設定されたアドレスによりバンク・レジスタのI/Oアドレスが決定されますが，さらに，A1/A0，またはA2/A1によって選択されるバンク・レジスタもプログラマブルになっています．

図6－63　バンク選択レジスタ

I／Oアドレス		7	6	5	4	3	2	1	0
FFE0H	BSEL	BNK3		BNK2		BNK1		BNK0	

BNK3，BNK2，BNK1，BNK0	チャネル3，2，1，0のバンク・レジスタ（BNKR3－BNKR0）のアドレス指定
00	A1（A2）＝0，A0（A1）＝0
01	A1（A2）＝0，A0（A1）＝1
10	A1（A2）＝1，A0（A1）＝0
11	A1（A2）＝1，A0（A1）＝1

(c)　バンク・レジスタ（BNKR0－BNKR3）

図6－64　バンク・レジスタ（BNKR0－BNKR3）

7	6	5	4	3	2	1	0
A23	A22	A21	A20	A19	A18	A17	A16

このレジスタはμPD71071モードの使用後は不定となります．バンク・レジスタはシステムI／O領域のOPSELによってDMAUが使用可の状態になっており，かつ，μPD71037モード時にのみI／O空間上に存在します．μPD71071モードでは内部I／O空間上には存在しません．ただし，BADR，BSELの両レジスタはDMAUのモードに関わらず，システムI／O空間上に存在します．バンク・レジスタはμPD71071モード時のアドレス・レジスタ上位8ビットを使用しています．

(d)　キャリーの伝搬制御

バンク・レジスタ使用時のA15からA16へのキャリー，またはA19からA20へのキャリーの伝搬制御は，システムI／O領域のSCTL内のCE0，CE1ビットで行います．

CE0	機　　能
0	μPD71037モード時にキャリーをA16に伝搬しない
1	μPD71037モード時にキャリーをA16に伝搬する

CE1	機　　能
0	μPD71037モード時にキャリーをA20に伝搬しない
1	μPD71037モード時にキャリーをA20に伝搬する

6．5．9　DMAUの転送モード

μPD71071モード時の転送モードは，DMDレジスタのビット7，6（TMODE）により指定され，μPD71037モード時の転送モードはライト・モード・レジスタ・コマンドのビット7，6（TMODE）により指定されます．μPD71071モード時にもμPD71037モード時も各転送モードの動作は同じです．

ただし，μPD71037モードではバス・モードは常にバス・リリースに固定です．

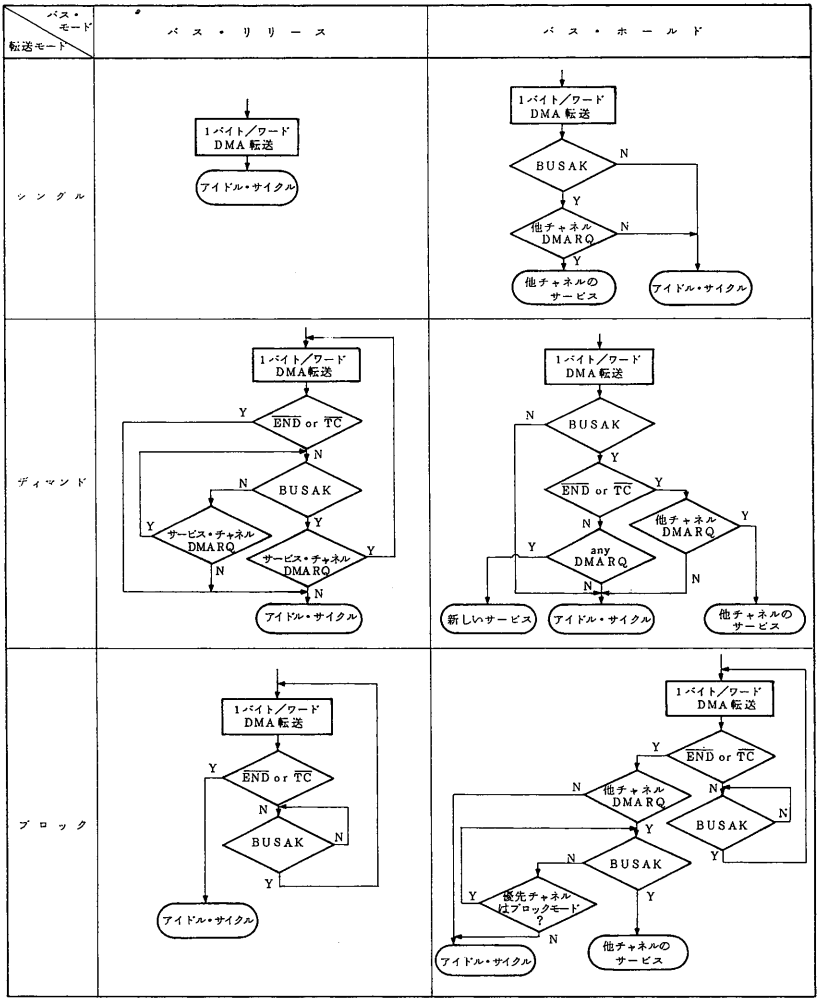
表6－16に転送モードの種類と転送終了条件を示します．

表6－16　転送モードとDMAサービス終了条件

転送モード	サービス終了条件
シングル・モード	毎1バイト／1ワード転送完了
ディマンド・モード	○外部からの$\overline{\text{END}}$入力 ○ターミナル・カウントの発生 ○サービス・チャネルDMARQの取り下げ ○より高優先のDMARQ発生（バス・ホールド・モード）
ブロック・モード	○外部からの$\overline{\text{END}}$入力 ○ターミナル・カウントの発生

複数のDMA要求があった場合，上記転送モードとバス・モードの組み合わせにより多少動作が異なります．転送モードとバス・モードの組み合わせによるそれぞれの場合の動作フローを表6－17に示します．

表6－17　転送モードとバス・モードの組み合わせによる各DMA動作



備考　ＢＵＳＡＫとは，Ｖ５３内部の信号で，Ｖ５３のＤＭＡＵ専用のＨＬＤＡＫ信号です．

6．5．10　オートイニシャライズ

オートイニシャライズ機能は，カレント・カウント・レジスタが“0”になりターミナル・カウント（$\overline{TC}$）が発生した場合，あるいは$\overline{END}$入力があった場合に，アドレスおよびカウント値を自動的に初期化する機能です．DMDレジスタによって各チャネルごとに設定できます．

オートイニシャライズが設定されているチャネルのサービス中に$\overline{END}$入力または$\overline{TC}$が発生すると，次の動作が行われます．

・カレント・アドレス・レジスタにベース・アドレス・レジスタの内容が，カレント・カウント・レジスタにベース・カウント・レジスタの内容が自動的に転送されます．
・マスク・レジスタの該当ビットはセットされません（オートイニシャライズが設定されていないチャネルは，マスク・レジスタの該当ビットがセットされます）．

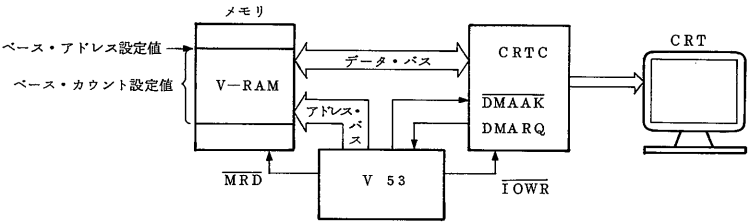
オートイニシャライズ機能は以下に示すようなDMA転送を行う場合に特に有効性が発揮されます．

(1)　メモリの一定エリアの繰り返し入出力

例えばCRTコントローラとV－RAMの間のDMA転送が考えられます．この場合，ベース・レジスタとカレント・レジスタに同一値を初期設定してオートイニシャライズにすれば，以後CPUは何の操作をしなくてもDMA転送が繰り返し実行されます．

図6－65　オートイニシャライズ応用（1／2）

(a)　CRTコントローラとV－RAM間のDMA転送



(2)　複数個のメモリ・エリアの連続転送

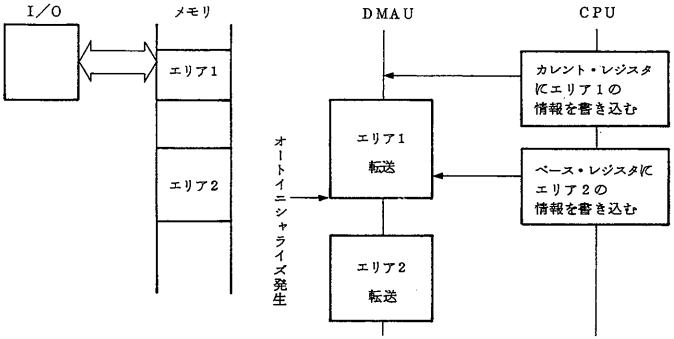
アドレス／カウント・レジスタへの書き込みをベース・レジスタのみに対して行うことができますのでオートイニシャライズ機能により次のようなことができます．

バス・リリース・モードでのシングルまたはディマンド・モード動作時に，複数個の連続したあるいは不連続なメモリ・エリアを連続して転送しようとする場合，ある1つのエリアの転送中に次のエリアの情報をベース・レジスタに設定すると，転送中のエリアのターミナル・カ

ウント時にオートイニシャライズ機能により次のエリアの情報がカレント・レジスタに転送され転送を連続に実行することができます．このようにオートイニシャライズ機能を用いると，CPUがDMAUへデータを書き込むタイミングに余裕ができます．

図６－６５　オートイニシャライズ応用（２／２）

(b)　複数個のメモリ・エリアの連続転送



６．５．１１　DMAサイクル

DMAUの動作は大きくDMAサイクルとアイドル・サイクルに分けられます．アイドル・サイクルではDMAU以外のバス・マスタがバスを使用することができますので，この間にDMAUに対するプログラミングを済ませます．

DMAサイクルは，４クロック／１バス・サイクルです．図６－６６にDMAタイミング例を示します．

DMA転送中の$\overline{\text{UBE}}$，A０の状態を表６－１８に示します．$\overline{\text{UBE}}$は，CPUサイクルと同じように，奇数アドレスを含むバス・サイクルでアクティブになります．

表６－１８　DMA動作時の$\overline{\text{UBE}}$，A０

(a)　μPD７１０３７モード

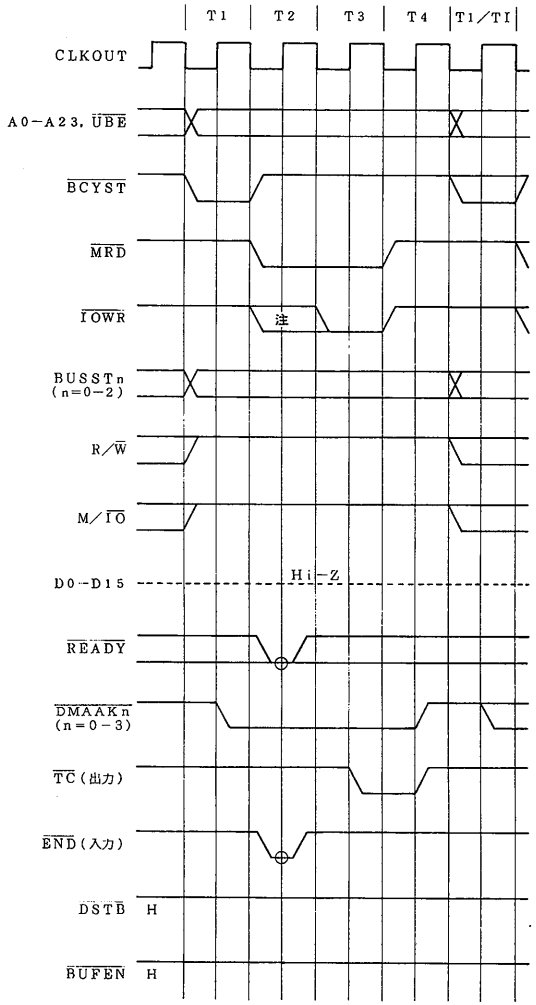
$\overline{\text{UBE}}$	A０	動作	
		転送単位	ｱｸｾｽ・ｱﾄﾞﾚｽ
0	1	バイト	奇数
1	0	バイト	偶数

(b)　μPD７１０７１モード

$\overline{\text{UBE}}$	A０	動作	
		転送単位	ｱｸｾｽ・ｱﾄﾞﾚｽ
0	0	ワード	偶数
1	0	バイト	偶数
0	1	バイト	奇数

図6－66　DMAタイミング（1／5）

(a)　メモリ→I／O転送（ウエイトなし）



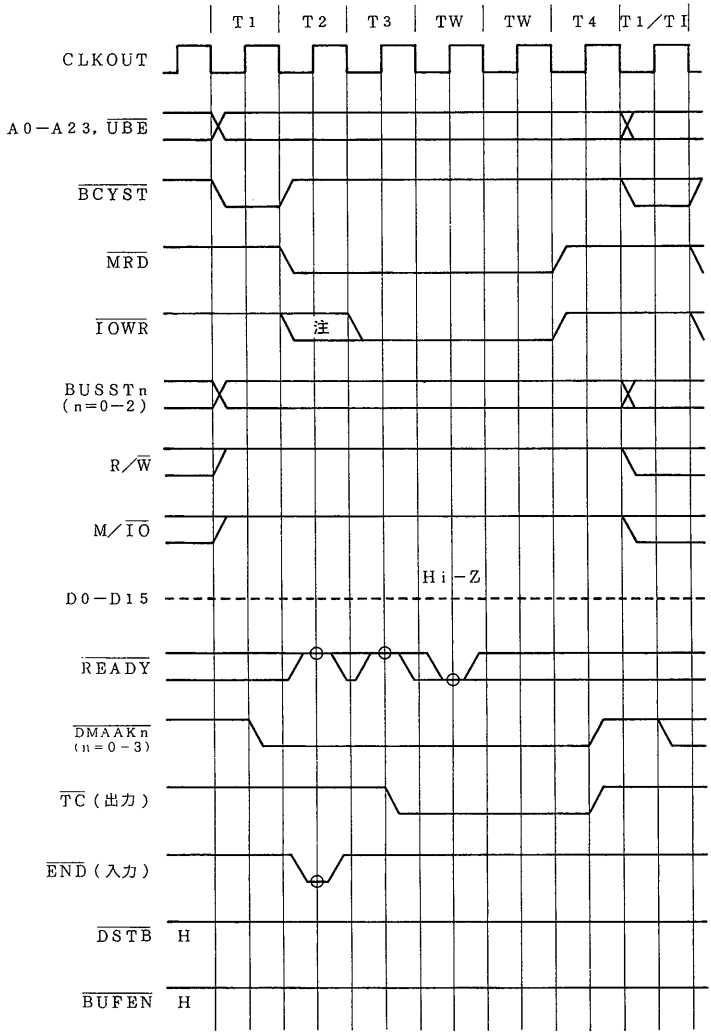
注　拡張ライト時はロウ・レベルを出力します．

備考　○印は，WCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUとREADY端子の関係は，6.2.3 WCUとREADY端子の関係を参照してください．

図6－66　DMAタイミング（2／5）

(b)　メモリ→I／O転送（2ウエイト時）



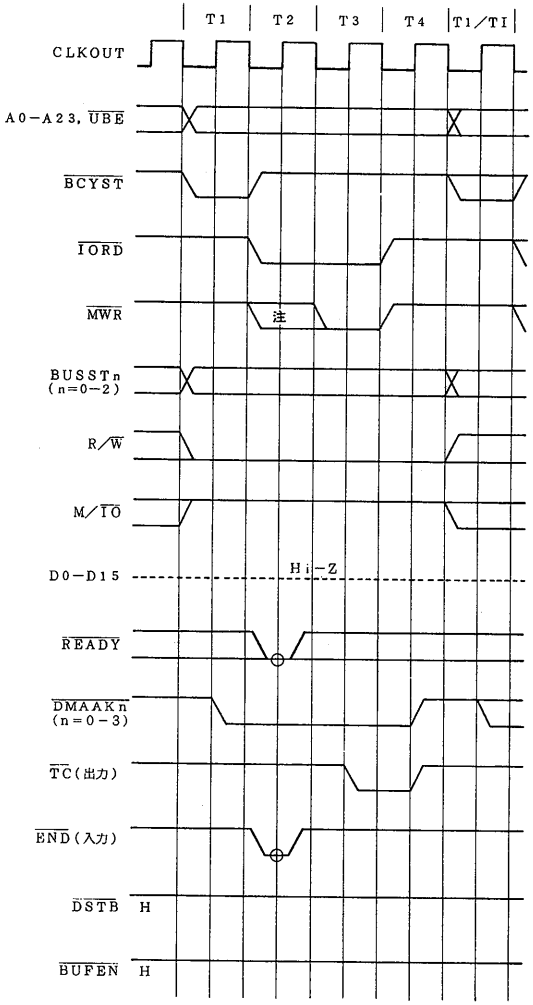
注　拡張ライト時はロウ・レベルを出力します．

備考　○印は，WCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUとREADY端子の関係は，6.2.3 WCUとREADY端子の関係を参照してください．

図6－66　DMAタイミング（3／5）

(c)　I／O→メモリ転送（ウエイトなし）



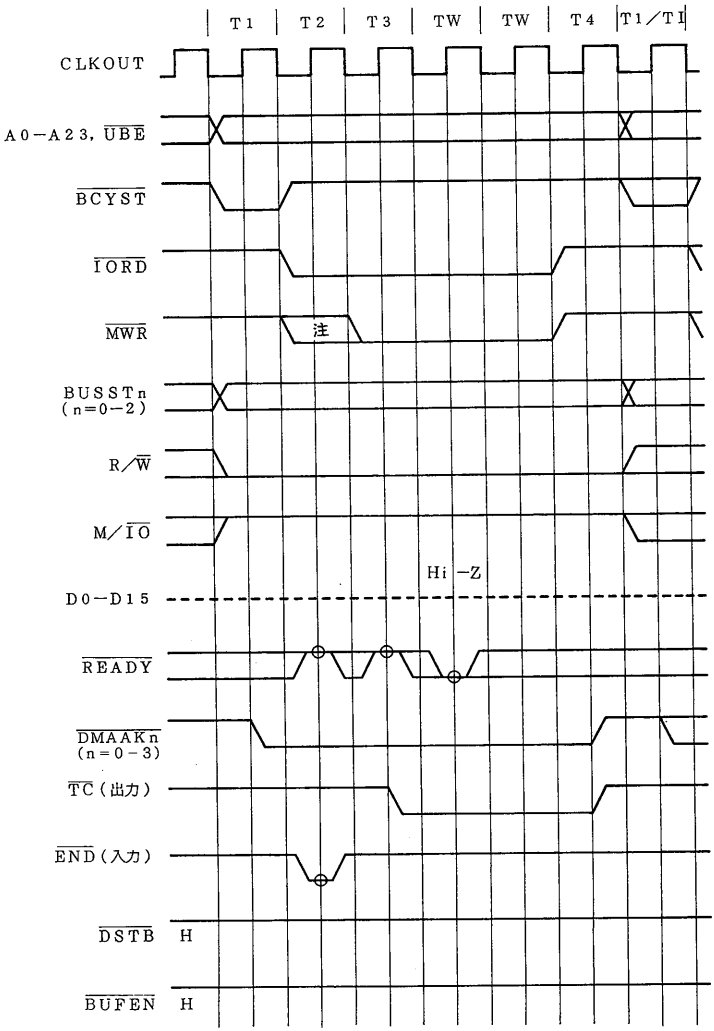
注　拡張ライト時はロウ・レベルを出力します．

備考　○印は，WCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUとREADY端子の関係は，6.2.3　WCUとREADY端子の関係を参照してください．

図6－66　DMAタイミング（4／5）

(d)　I／O→メモリ転送（2ウエイト時）



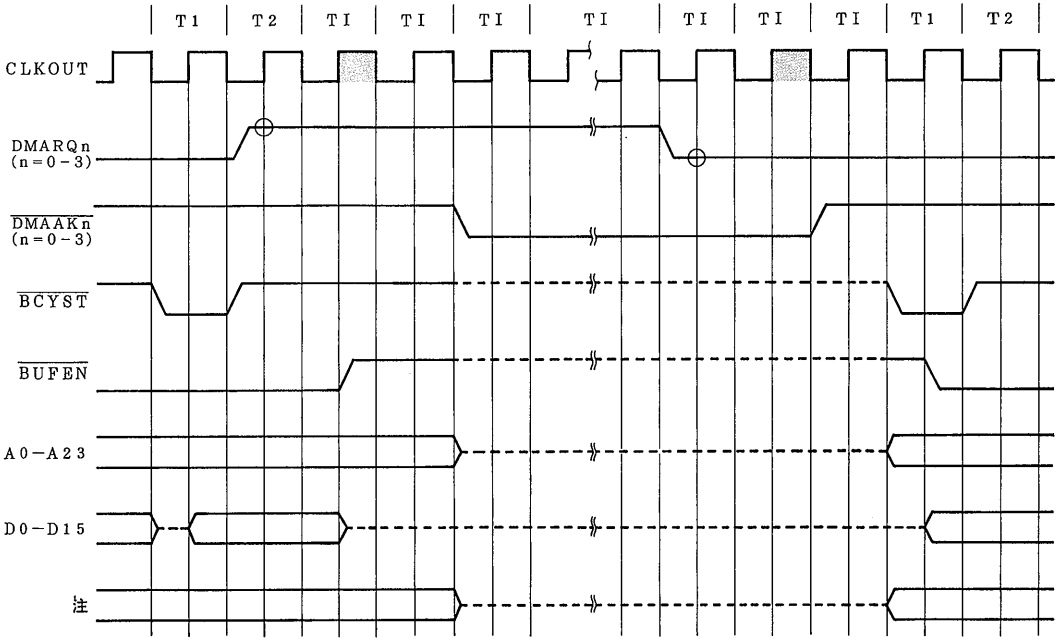
注　拡張ライト時はロウ・レベルを出力します．

備考　○印は，WCUにウエイト・サイクル0が設定されているときに，サンプリングされるタイミングです．

WCUとREADY端子の関係は，6.2.3 WCUとREADY端子の関係を参照してください．

図6－66　DMAタイミング（5／5）

(e) カスケード・モード



注　UBE, MRD, MWR, IOWR, IORD

備考 1. ▒▒は，バス使用権の切り替えを行うタイミングを示します．

2. ○印はサンプリングされるタイミングです．

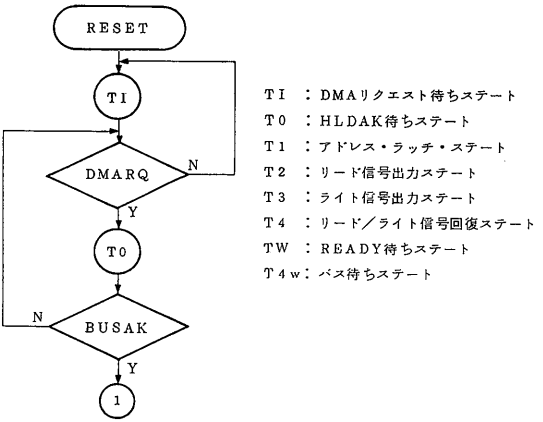
3. 破線はハイ・インピーダンスを示します．ただし，出力ラッチを内蔵してますので，ハイ・インピーダンス時には，外部からドライブされるまで，ハイ・インピーダンスとなる以前の状態を保持しています．

6．5．12　DMAU状態遷移図

DMAUの状態遷移図を以下に示します．μPD71071モード時もμPD71037モード時も同じです．

ただし，μPD71037モードではバス・モードは常にバス・リリースに固定です．

図6－67　アイドル・サイクル



備考　BUSAKとは，V53内部の信号で，V53のDMAU専用のHLDAK信号です．

図6－68　DMAサイクル（カスケード・モード）

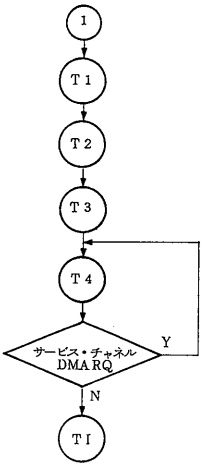
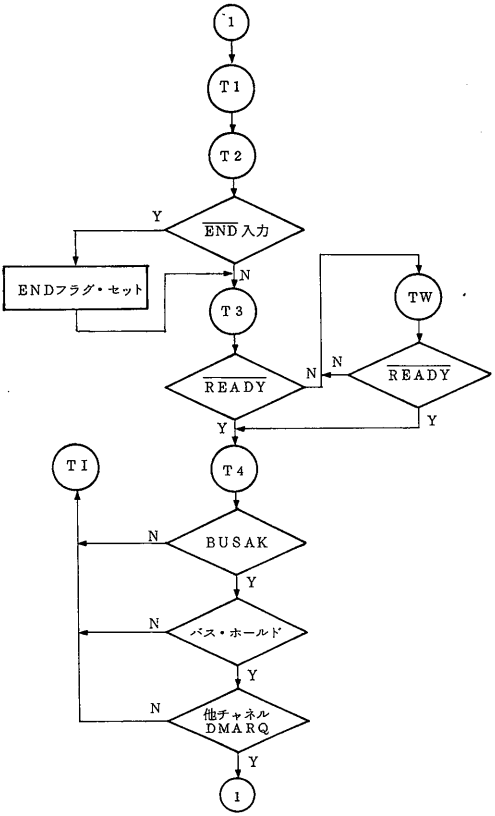
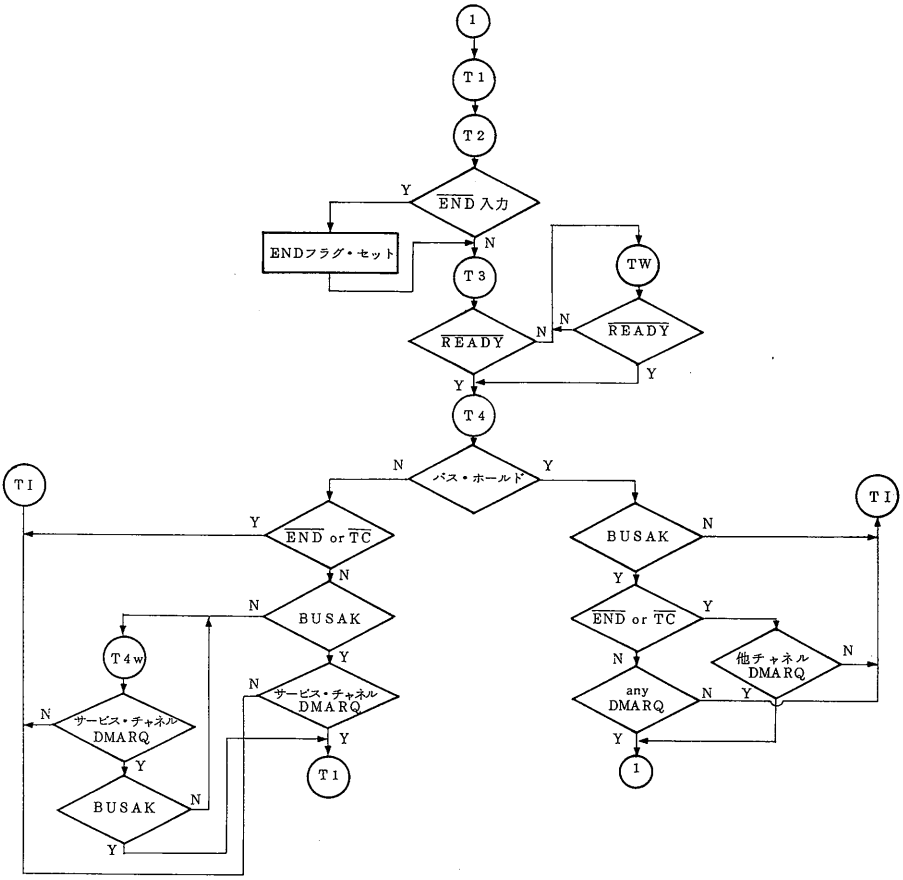


図 6－69　DMA サイクル・シングル・モード



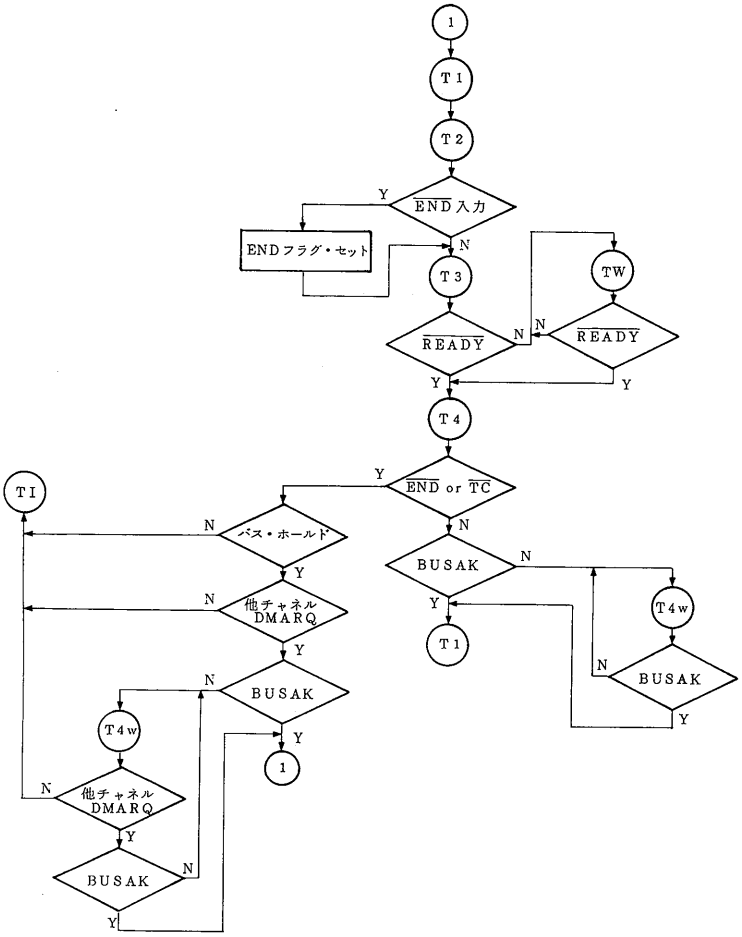
備考　ＢＵＳＡＫとは，Ｖ５３内部の信号で，Ｖ５３のＤＭＡＵ専用のＨＬＤＡＫ信号です．

図6－70　DMAサイクル・ディマンド・モード



備考　ＢＵＳＡＫとは，Ｖ５３内部の信号で，Ｖ５３のＤＭＡＵ専用のＨＬＤＡＫ信号です．

図６－７１　DMAサイクル・ブロック・モード



備考　ＢＵＳＡＫとは，Ｖ５３内部の信号で，Ｖ５３のＤＭＡＵ専用のＨＬＤＡＫ信号です．

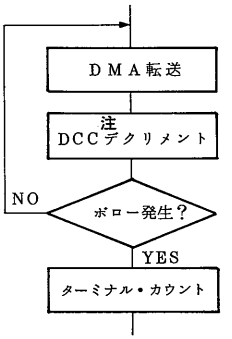
6．5．13　その他使用上の注意事項

(a)　ターミナル・カウントと転送数

カレント・カウント・レジスタは，DMA転送のたびにデクリメントされて転送回数を数えています．ターミナル・カウントは，図6－72のフロー・チャートが示すように，カウント・レジスタが“0”になったときではなく，ボローが発生したときに起こります．

したがって，ターミナル・カウントが発生して$\overline{TC}$を出力するまでに行われるDMA転送の回数は，カレント・カウント・レジスタに設定された値よりも1回多くなります．

図6－72　ターミナル・カウントの発生



注　カレント・カウント・レジスタ

(b)　バス待ち動作

通常，DMAUは最も高いバス使用権を有しており，ほかのバス・マスタによるバス要求によってDMAサービスが中断させられることはありません．しかし，REFUを使用している場合，リフレッシュ要求が長時間待たされると，REFUのバス使用権はDMAUよりも高くなり，DMAUに対しバスの返還を要求します．

その場合，DMAUは現在実行中のDMAサイクルが終了次第，DMAサービスを一時中断してREFUにバスを明け渡します．REFUはたまっていたリフレッシュ要求をまとめて4または5回実行したあと，再びDMAUにバスの使用権を戻します．

このようにして，REFUを使用している場合は，DMAUの転送モード（シングル，ディマンド，ブロック），バス・モード（バス・リリース，バス・ホールド）に関係なくDMAサービスが一時的に中断される場合があります．

これがバス待ち動作ですが，一時的にDMAサービスが中断される以外はDMA機能にはなんの影響も与えません．

6.6 TCU(タイマ／カウンタ・ユニット)

TCUは3組のカウンタを有し，機能的にはμPD71054と同じです．

6.6.1 特　徴

- タイマ3チャネル
- TOUT0－TOUT2端子出力
- TCTL0－TCTL2端子入力
- TCLK端子入力
- TOUT1をSCUのクロックとして使用可能
- 16ビット・カウンタ×3
- プログラマブルな6つのカウント・モード
- バイナリ／BCDカウント
- マルチプル・ラッチ・コマンド
- カウント・ラッチ・コマンド
- 内部／外部入力クロック選択可能
- 10MHz動作可能

6.6.2 V50に対する変更箇所

- TOUT0，1出力端子を追加
- TCTL0，1入力端子を追加

TCUの汎用性を高めるために，出力信号（TOUT），制御信号（TCTL）を3チャネルすべてに用意しています（マルチプレクスなし）．

6．6．3　TCU内部ブロック図

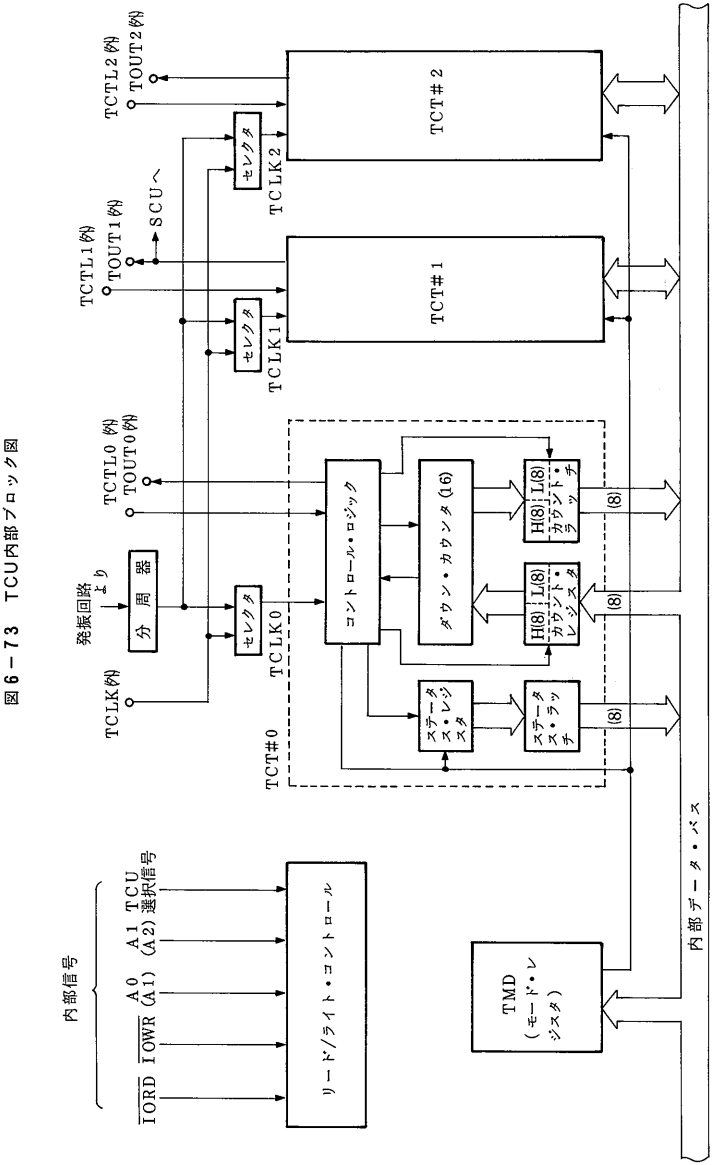


図 6－73　TCU内部ブロック図

6．6．4　アドレシング

TCUの内部レジスタは，システムI/O領域で設定したOPHAとTULAおよびアドレス信号（A1,A0またはA2,A1）によってアドレスされます．

表6－19　TCU内部レジスタのアドレス

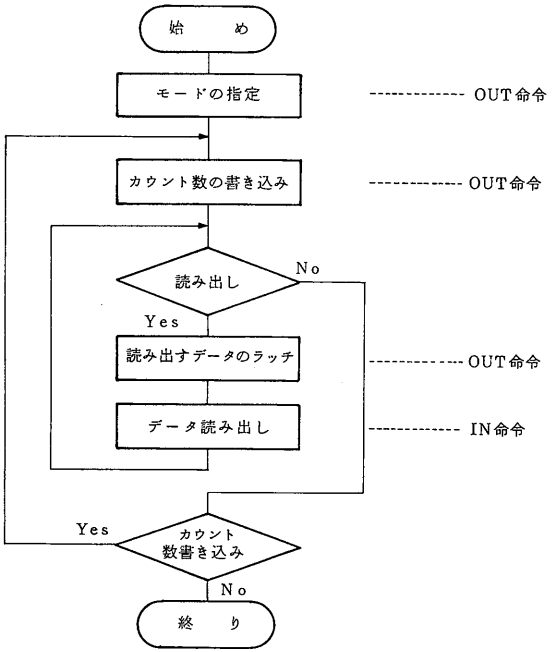
	上位アドレス	下位アドレス			レジスタ	操作
IOAG=1のとき	A15—A8（OPHA）	A7 A6 A5 A4 A3 A2（TULA）		0 0	TCT#0	リード/ライト
					TST0	リード
				0 1	TCT#1	リード/ライト
					TST1	リード
				1 0	TCT#2	リード/ライト
					TST2	リード
				1 1	TMD	ライト
IOAG=0のとき	A15—A8（OPHA）	A7 A6 A5 A4 A3（TULA）	0 0	A0	TCT#0	リード/ライト
					TST0	リード
			0 1		TCT#1	リード/ライト
					TST1	リード
			1 0		TCT#2	リード/ライト
					TST2	リード
			1 1		TMD	ライト
—	11111111	1 1 1 1 0 0 0 0			TCKS	リード/ライト

備考　TCKSのアドレスは，IOAG，OPHA，TULAの内容に関係なくFFF0Hです．

6．6．5　TCUの操作手順

TCUは電源投入後，$\overline{\text{RESET}}$によりリセットが行われないため 未定義状態となっています．そのため，使用する場合は，目的のカウンタをプログラムして，動作モードを指定する必要があります．一度あるモードにプログラムされたカウンタは，再びそのカウンタがモード指定されるまではそのモードで動作します．カウンタにカウント数を書き込み，それがダウン・カウンタに転送されると新しいカウントが開始されます．カウントの途中で現在のカウンタの値であるカウント・データや，そのカウンタの状態を示すステータスを読み出すこともできます．

図6－74　基本操作手順



6．6．6　TCUのレジスタ，コマンド

TCUのレジスタのリード／ライト，コマンドの発行は，システムI／O領域で設定したアドレスに対するI／O入出力命令で行い，レジスタ等の選択はA1,A0またはA2,A1で行います．

表6－20　TCUレジスタ／コマンド・アドレス

A1 (A2)	A0 (A1)	レジスタ／コマンド	操　作
0	0	TCT#0	リード／ライト
		TST0	リード
0	1	TCT#1	リード／ライト
		TST1	リード
1	0	TCT#2	リード／ライト
		TST2	リード
1	1	TMD	ライト
注		TCKS	リード／ライト

注　TCKSのアドレスはFFF0Hに固定です．

TMDへの書き込みはTCU内の各カウンタの動作モード（カウント・モード，バイナリ／BCD，リード／ライト・モード）の設定と，カウンタの値をラッチするためのコマンド（カウント・ラッチ・コマンド，マルチプル・ラッチ・コマンド）の発行を行います．

TCT#2－TCT#0は各カウンタへのカウント数の書き込み／カウント・データの読み出しに用います．通常，カウント・データの読み出しは，その前にカウント・ラッチ・コマンドまたは，マルチプル・ラッチ・コマンドを発行して，目的のカウンタのカウント・データをラッチしてから行います．

TST2－TST0のリードは各カウンタのステータス情報の読み出しを行います．ステータスの読み出しは，マルチプル・ラッチ・コマンドで目的のカウンタのステータスをラッチしたあとに行います．

1つのカウンタについて，ステータスとカウント・データの両方がラッチされている場合，最初の読み出しでステータスが得られ，その次の読み出しでカウント・データが得られます．

(1) TMD (タイマ・モード・レジスタ)

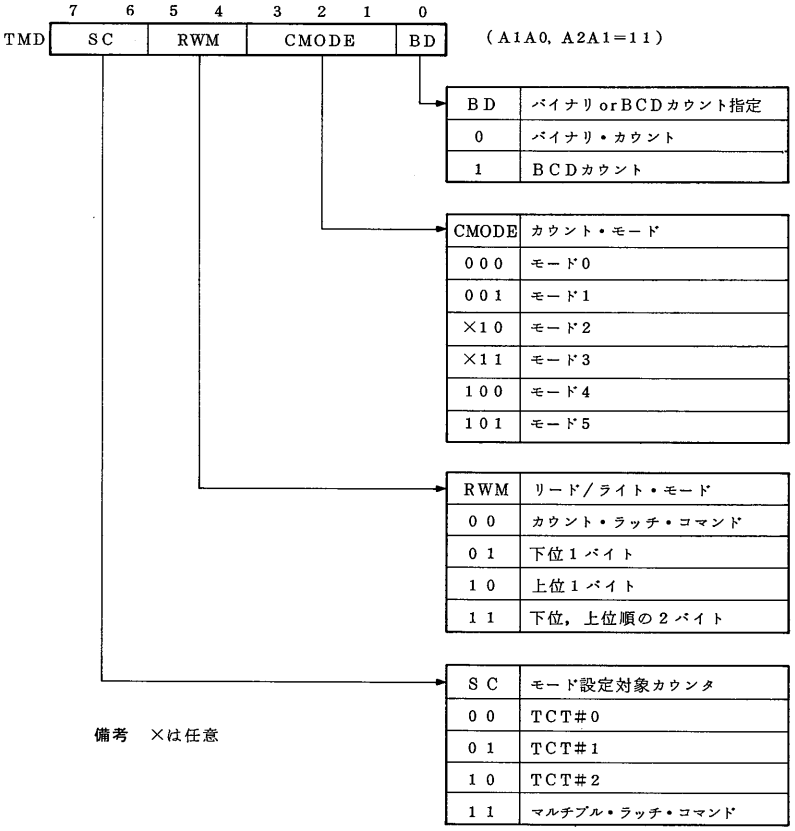
TMDレジスタに書き込むデータには，各カウンタの動作モードを設定するモード・ワードのほかに，カウンタの値をラッチするための次のようなコマンドがあります．

- カウント・ラッチ・コマンド
- マルチプル・ラッチ・コマンド

それぞれ，(4)，(5)項で説明します．

カウンタを動作させるには，まずTMDにモード・ワードを書き込んで，各カウンタごとにモードを設定します．

図6－75　モード・ワード



備考　×は任意

(a) SC

SCビットは，モード設定対象カウンタ（TCT#2－TCT#0）またはマルチプル・ラッチ・コマンドの指定を行います．カウンタ指定を行った場合は指定されたカウンタに対してのみRWM，CMODE，BDの各ビットの設定が有効になります．

マルチプル・ラッチ・コマンドを指定した場合は，ビット5－0の意味は異なります．マルチプル・ラッチ・コマンドについては(5)，(6)項を参照してください．

(b) RWM

RWMビットはカウント・レジスタへの書き込み，カウント・ラッチの読み出しを指定するリード／ライト・モード，またはカウント・ラッチ・コマンドの指定を行います．カウント・ラッチ・コマンドについては，(4)項を参照してください．

(c) CMODE

CMODEビットはカウント・モード0－5を指定します，

各カウント・モードの動作は，6.6.7を参照してください．

(d) BD

BDビットはバイナリ・カウント，BCDカウントの指定を行います．バイナリの場合は0000H－FFFFHのカウント数の設定が可能で，2進カウント動作を行います．

BCDの場合は0－9999のカウント数の設定が可能で，10進カウント動作を行います．

(2) TCKS (タイマ・クロック選択レジスタ)

TCKSはTCU内の3本のカウンタ(TCT#0－TCT#2)に供給するクロック(TCLK0－TCLK2)をTCLK端子から入力するか，または内部クロック(X1，X2端子に接続した水晶の発振周波数を分周したクロック)にするかを選択します．

TCKSのアドレスは，システムI/O領域のFFF0H番地に固定されています．ほかのTCU内部レジスタとは異なります．

図6－76　TCKS(タイマ・クロック選択レジスタ)

I/Oアドレス		7	6	5	4	3	2	1	0
FFF0H	TCKS	－	－	－	CS2	CS1	CS0	PS	

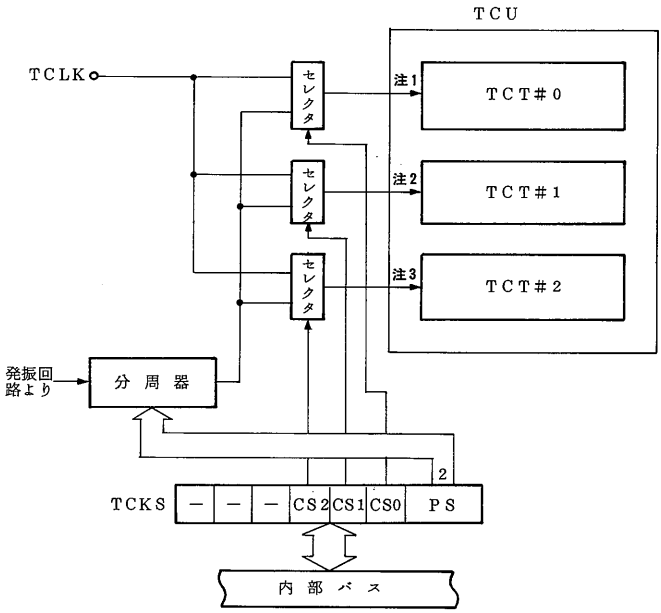
CS2	TCUチャネル2のクロック選択
0	内部クロック使用
1	TCLK端子入力使用

CS1	TCUチャネル1のクロック選択
0	内部クロック使用
1	TCLK端子入力使用

CS0	TCUチャネル0のクロック選択
0	内部クロック使用
1	TCLK端子入力使用

PS	内部クロック使用時の周波数指定
0 0	$TCLK_n = fxx/4$
0 1	$TCLK_n = fxx/8$
1 0	$TCLK_n = fxx/16$
1 1	$TCLK_n = fxx/32$

図 6 － 7 7　TCUへのクロック供給



注 1.　TCLK0

2.　TCLK1

3.　TCLK2

(3) TCT#0－TCT#2（タイマ・カウント・レジスタ#0－#2）

各チャネルにはそれぞれ16ビット長のカウント・レジスタが1本ずつあり，その書き込み，読み出しはモード・ワードで設定したリード／ライト・モードに従って行います．下位1バイトおよび上位1バイト・モードに設定した場合は1回の書き込みでそれぞれカウント・レジスタの下位バイト，または上位バイトに書き込まれます．この場合，残りの上位バイト，下位バイトは“00H”になります．下位，上位2バイト・モードでは1回目の書き込みで下位バイトに書き込み，続いて2回目の書き込みで同じアドレスに上位バイトを書き込みます．

表6－21　カウント・レジスタへの書き込み

リード／ライト・モード	書き込み回数	カウント・レジスタ	
		上位バイト	下位バイト
下位1バイト	1	00H	××H
上位1バイト	1	××H	00H
下位・上位2バイト	2	××H（2回目）	××H（1回目）

備考　××Hは任意値（有効データ）です．

カウンタからの読み出しの場合も基本的に書き込みと同じです．下位，上位2バイト・モードでは，1回目の読み出しで下位バイトを読み出したあと，続いて2回目の読み出しで同じアドレスから上位バイトを読み出します．

カウンタからの読み出し手順としては，次の3つがあります．

(a)　カウンタからの直接読み出し
(b)　カウント・ラッチ・コマンド後の読み出し
(c)　マルチプル・ラッチ・コマンド後の読み出し

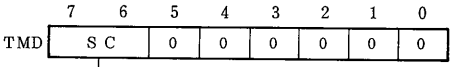
カウンタからの直接読み出しの場合は，ダウン・カウンタの追従状態にあるカウント・ラッチを読み出すので，読み出し動作中にその値が変化することもあり，得られる値が不正確になります．正確な値を得るにはTCLKnやTCTL入力を操作してカウント停止状態で読み出す必要があります．通常は(b)および(c)の方法で読み出します．(c)の方法ではカウント・データのほかにステータスも読み出せます．

表6－22　カウンタからの読み出し

A1／A2	A0／A1	読み出し対象
0	0	TCT#0，TST0
0	1	TCT#1，TST1
1	0	TCT#2，TST2

(4) カウント・ラッチ・コマンド

図6－78　カウント・ラッチ・コマンド・フォーマット



SC	ラッチ対象カウンタ
0 0	TCT#0
0 1	TCT#1
1 0	TCT#2

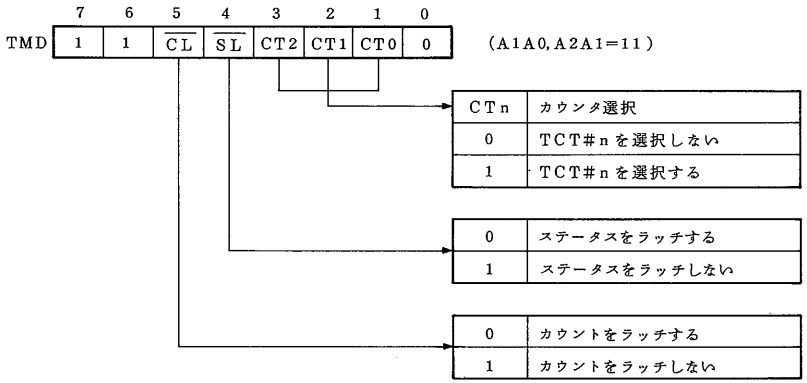
カウント・ラッチ・コマンドの発行は，ビット5－0が全ビット“0”であるデータをTMDに書き込むことにより行います．

カウント・ラッチ・コマンドを発行すると，SCビットで選択されたダウン・カウンタの内容がカウント・ラッチにラッチされます．ラッチされたカウント・データはそれが読み出されるか，もしくは新たなモード設定が行われるまで保持されます．このカウント・ラッチ・コマンドにより，コマンド発行時での正確なカウント・データをカウント動作に影響を与えずに読み出せます．

カウント・ラッチ・コマンドを発行して，そのカウンタを読み出さないうちに再び同じカウンタに対してカウント・ラッチ・コマンドを発行した場合は，後者のコマンドは無視され，前にラッチされている値が保持されます．ラッチされているカウント・データを読み出せば，ラッチは解除され，もとのダウン・カウンタの追従状態に戻ります．

(5) マルチプル・ラッチ・コマンド

図6－79　マルチプル・ラッチ・コマンド・フォーマット



マルチプル・ラッチ・コマンドの発行は，ビット7，6が“11”であるデータをTMDに書き込むことにより行います．

マルチプル・ラッチ・コマンドを発行すると，選択されたカウンタすべてのカウント・データ，およびステータスがカウント・ラッチ，ステータス・ラッチにラッチされます．ラッチされた状態でカウンタを読み出せば，ラッチされた状態でのカウント・データおよびステータスをカウンタの動作に影響を与えずに読み出すことができます．

(a) $\overline{CL}$

$\overline{CL}$ビットは選択されたカウンタすべてのカウント・データをラッチする／しないを指定します．

(b) $\overline{SL}$

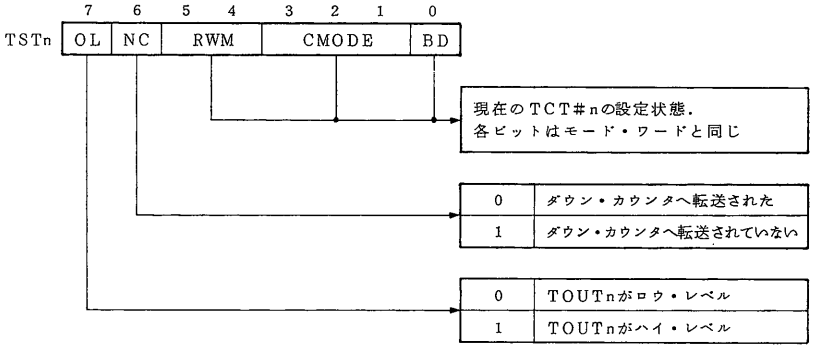
$\overline{SL}$ビットは選択されたカウントすべてのステータスをラッチする／しないを指定します．

(c) CT2－CT0

CT2－CT0の3ビットはラッチする対象となるカウンタを指定します．複数のカウンタのカウント・データ，ステータスを同時にラッチすることができます．

ステータスは，マルチプル・ラッチ・コマンド発行時のカウンタの動作状態を示しています．ラッチされたステータスのフォーマットを図6－80に示します．

図6－80　ステータス・フォーマット



OLビットは，マルチプル・ラッチ・コマンド発行時点でのカウンタの出力状態を示しています．

NCビットはカウント無効フラグで，最も新しく書き込まれたカウント数がダウン・カウンタへ転送されたかどうかを示しています．

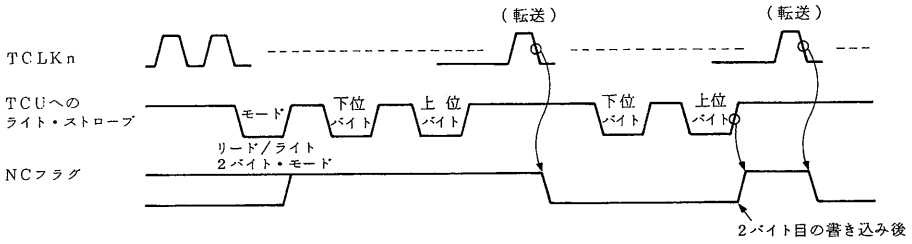
BD，CMODE，RWMの各ビットはカウンタの設定状態を示しており，モード・ワードで設定した値と同じです．

表6－23　NCフラグの変化

カウンタ（への）動作	NCフラグ
モード・ワード（対応するカウンタへの）の書き込み	1
カウント・レジスタへのカウント数の書き込み 注	1
カウント・レジスタからダウンカウンタへのカウント数の転送	0

注　リード／ライトが2バイト・モードの場合には，2バイト目が書き込まれた時点でフラグが1になります．

図6－81　NCフラグの変化例



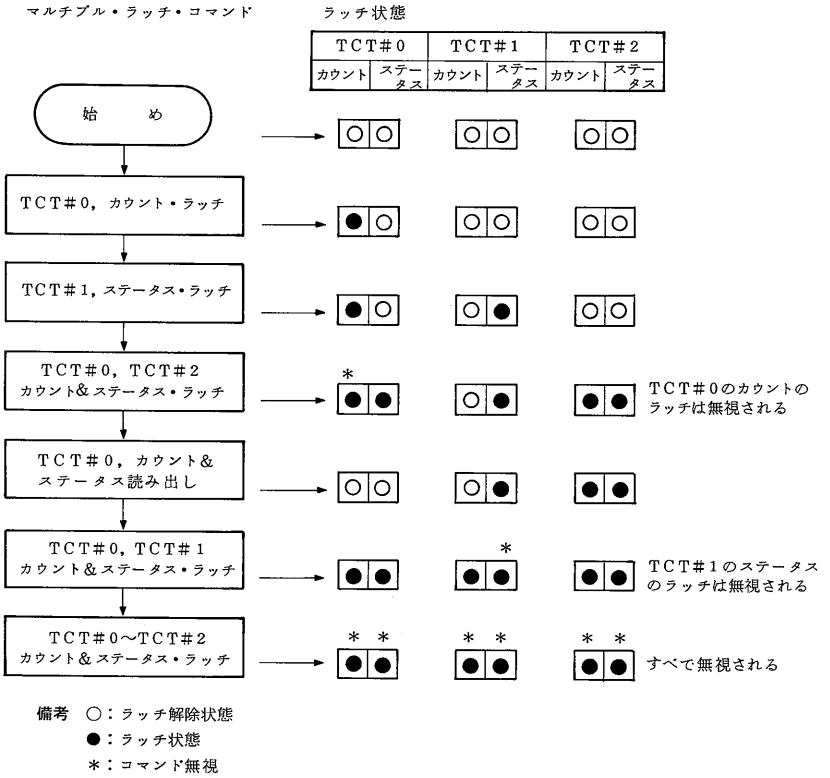
(6) マルチプル・ラッチ・コマンド発行時の注意事項

マルチプル・ラッチ・コマンドでラッチしたカウント・ラッチ，ステータス・ラッチは，それが読み出されるか，もしくは新たにモード設定がなされるまではデータを保持しています．

すなわち，ラッチされてから，まだ読み出されていないうちに再びマルチプル・ラッチ・コマンドを発行しても，そのコマンドは無視されます．また，カウント・ラッチ，ステータス・ラッチはどちらも一度読み出されると，ラッチは解除されます．図6－82にマルチプル・ラッチ・コマンドの実行例を示します．

マルチプル・ラッチ・コマンドではカウント・データ，ステータスのどちらが先にラッチされたとしても，最初の読み出しでは常にステータスが読み出されます．次の1回，または2回（リード／ライト・モードにより異なる）の読み出しでカウント・データが読み出されます．さらに読み出しを読けると，ラッチされていないダウン・カウンタの追従状態のカウント値が読み出されます．

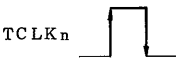
図6－82　マルチプル・ラッチ・コマンド実行例



6．6．7　カウント・モード

ここでは，モード・ワードのCMODEビットで設定する６つのカウント・モードについて説明します．各モードの説明で使われる用語の意味は次のとおりです．

・TCLKnパルスとは，TCLKn入力の立ち上がりから次の立ち下がりまでです．
・トリガとは，TCTL入力の立ち上がりエッジです．



TCLKnパルス

トリガ

・TCTLは，TCLKnの立ち上がりごとにサンプリングされています．サンプリングには，レベル検出と，立ち上がり（トリガ）検出とがあります．
・TOUT初期状態とは，モード・ワードでカウント・モードを設定すると即座に決定されるTOUT出力の状態です．
・カウント数転送とは，カウント・レジスタからダウン・カウンタへのカウント数の転送のことです．
・デクリメントは，ダウン・カウンタの動作で，TCLKnパルスの立ち下がり時に行われます．
・カウント・ゼロとは，ダウン・カウンタの内容が“０”になった状態です．
・LBは，カウント数の下位バイトです．
・HBは，カウント数の上位バイトです．

各カウント・モードでの動作例（タイミング図）は，TCT#２を，リード／ライト１バイト・モード，バイナリ・カウントで実行した場合です．TCTL２信号が省略されている場合は，TCTL2入力はハイ・レベルとします．TOUT信号の下の１６進数は，その時点でのカウント値を表しています．“？”記号は，モード指定直後では不定値，それ以外の場合は前のカウントの続きであることを示しています．各モードでの最大カウント数の設定値は“０”です．

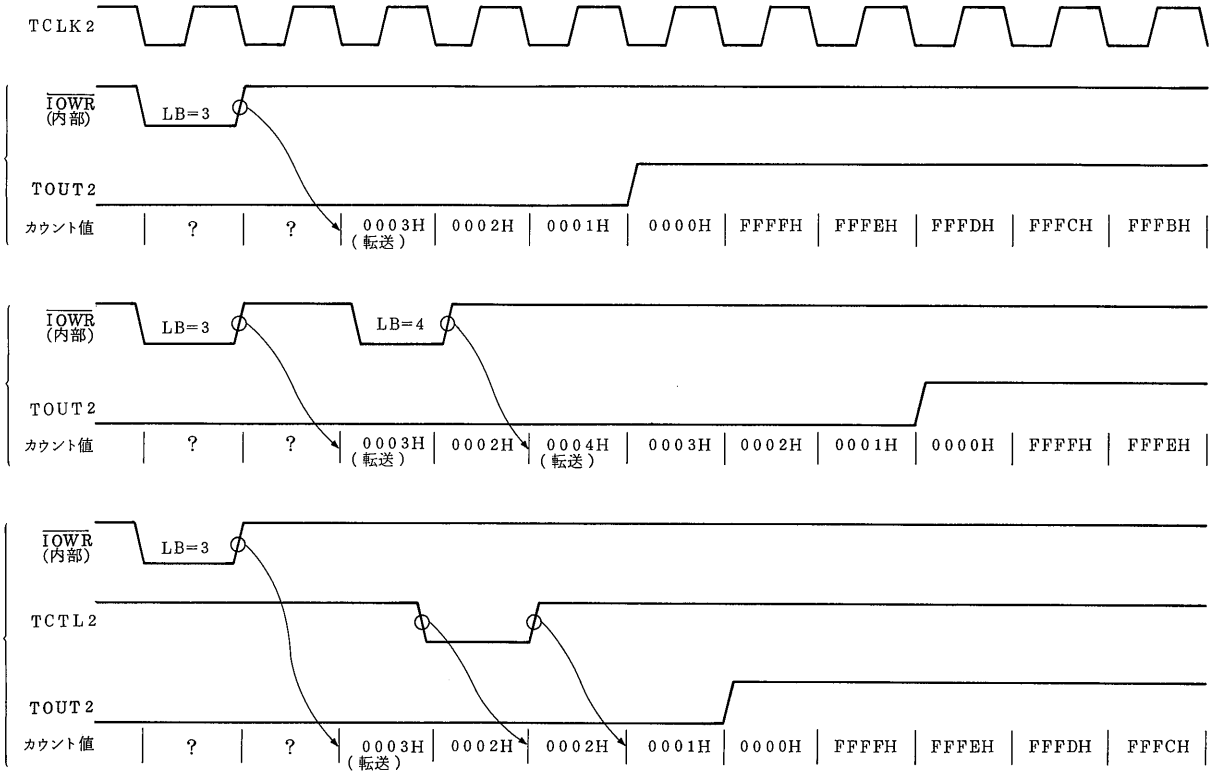
(1) モード0：カウント終了信号の出力

指定したカウントが終了すると同時にTOUT端子はロウ・レベルからハイ・レベルに変わります．

表6－24　モード0動作

項目		内容
TOUT初期状態		ロウ・レベル
TCTL入力	ハイ・レベル	カウント状態
	ロウ・レベル	カウント禁止状態
カウント数書き込み		TOUT端子がロウ・レベルになります（TCLKnパルスに無関係）． リード／ライトが2バイト・モードの場合は，1バイト目を書くとカウントが禁止され，TOUT端子がロウ・レベルになります．
カウント数転送とカウント		TCTL＝ハイ・レベル時に書き込まれた場合 カウント数が書き込まれた次のTCLKnパルスで転送が行われます． デクリメントは転送の次のTCLKnパルスから開始されますので，カウント数がNならば，TOUT端子は（N＋1）・TCLKnパルスの間ロウ・レベルになります． TCTL＝ロウ・レベル時に書き込まれた場合 カウント数が書き込まれた次のTCLKnパルスで転送が行われます． TCTLがハイ・レベルになると，その次のTCLKnパルスからデクリメントが開始されますので，カウント数がNならば，TOUT端子はN・TCLKnパルスの間ロウ・レベルになります．
カウント・ゼロ		TOUT端子がハイ・レベルになります．カウント動作は止まらずに，バイナリならばFFFFH，BCDならば9999へとカウント・ダウンされていきます．
最小カウント数		1

図６－８３　モード０動作例



(2) モード1：コントロール端子リトリガブル・ワンショット

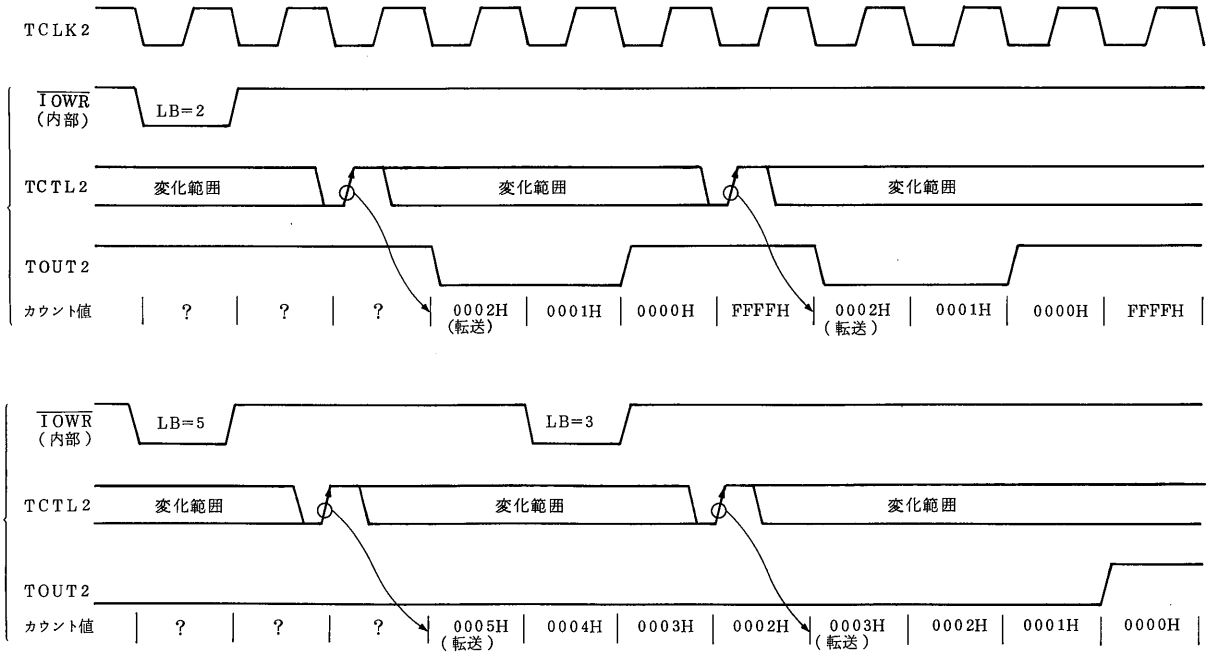
指定した長さのロウ・レベル・ワンショット・パルスがTOUT端子に出力されます．TCTL入力により再トリガが可能です．

表6－25　モード1動作

項目		内容
TOUT初期状態		ハイ・レベル
TCTL入力	トリガ注	トリガの次のTCLKnパルスでカウント数転送が起こります．
カウント数書き込み		現在の動作に影響せずに書き込まれます．
カウント数転送とカウント		トリガがあると，次のTCLKnパルスで転送が行われ，同時にTOUT端子がロウ・レベルになり，ワンショット・パルスが開始されます．デクリメントは次のTCLKnパルスから開始されますので，カウント数がNならば，TOUT端子のワンショット出力はN・TCLKnパルスの間続きます．
カウント・ゼロ		TOUT端子がハイ・レベルになります．カウント動作は止まらずに，バイナリならFFFFH，BCDなら9999へとカウント・ダウンされていきます．
最小カウント数		1

注　モード指定直後でまだカウント数が書かれていない状態，およびリード／ライト・2バイト・モードで1バイトしか書かれていない状態ではトリガは無視されます．

図 6－84　モード 1 動作例



(3) モード2：レート・ジェネレータ

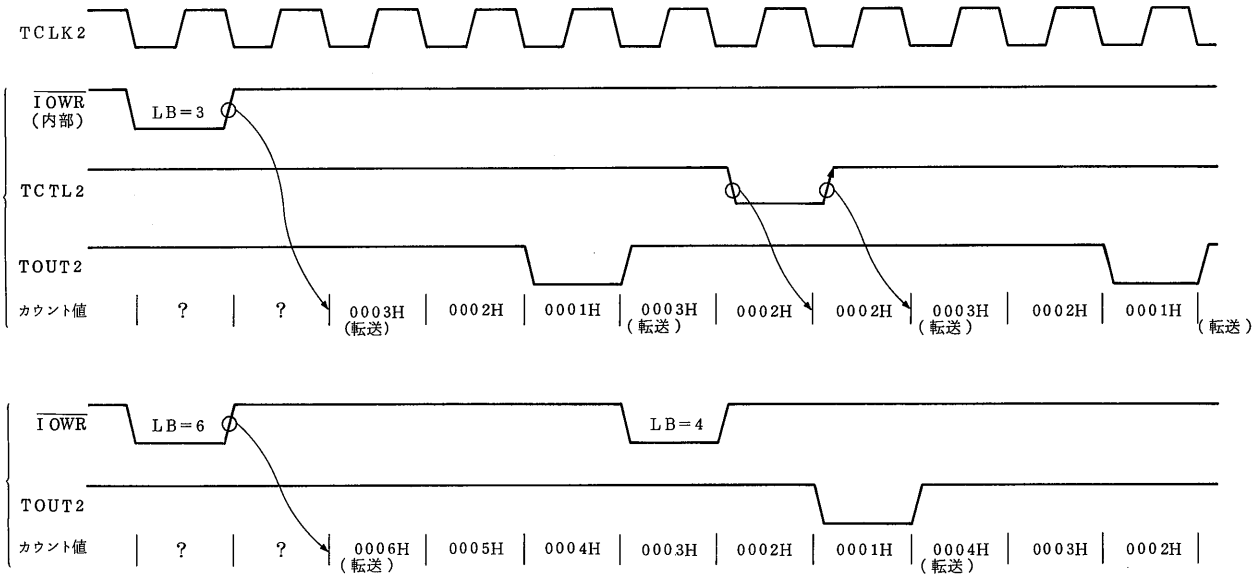
指定したカウントの最後の1TCLKnパルス分だけTOUT端子がロウ・レベルになる動作を周期的に行う分周カウンタです．

表6－26　モード2動作

項目		内容
TOUT初期状態		ハイ・レベル
TCTL入力	ハイ・レベル	カウント状態
	ロウ・レベル	カウント禁止状態．TOUT端子がロウ・レベルのときにTCTLがロウ・レベルになると，TOUT端子はハイ・レベルになります．(TCLKnパルスに無関係)
	トリガ注	トリガの次のTCLKnパルスでカウント数転送が起こります．
カウント数書き込み		現在の動作に影響せずに書き込まれます．
カウント数転送とカウント		モード指定に続くカウント数の書き込みの次のTCLKnパルスで転送されます．その後はカウント数がデクリメントされてカウント値が1になった次のTCLKnパルスで転送が行われます．また，トリガがあればその次のTCLKnパルスでも転送が行われます． TOUT端子は，ダウン・カウンタの内容が1になると同時に1TCLKnの期間だけロウ・レベルになり，再びハイ・レベルに戻ります．よって，カウント数がNならば動作は，N・TCLKnパルスを周期として繰り返されます．
カウント・ゼロ		このモードでは起こりません．
最小カウント数		2

注　モード指定直後でまだカウント数が書かれていない状態，およびリード/ライト・2バイト・モードで1バイトしか書かれていない状態ではトリガは無視されます．

図6－85　モード2動作例



(4) モード3：方形波ジェネレータ

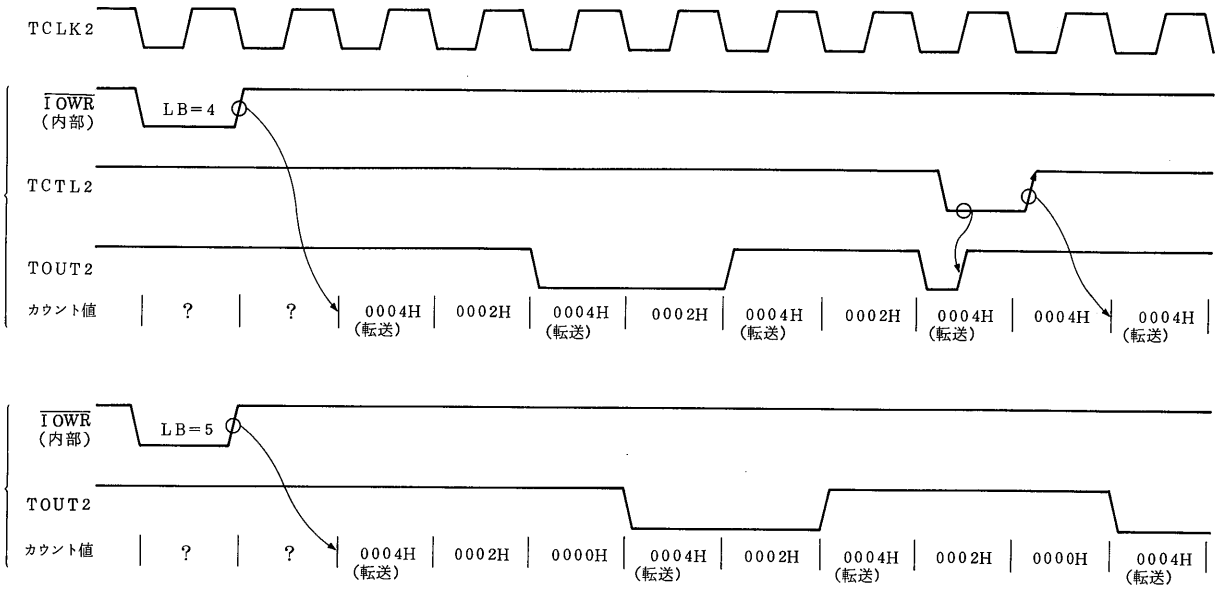
モード2と同様の分周カウンタですが，デューティが異なります．

表6－27　モード3動作

項目		内容
TOUTの初期状態		ハイ・レベル
TCTL入力	ハイ・レベル	カウント状態
	ロウ・レベル	カウント禁止状態．TOUT端子がロウ・レベルのときにTCTLがロウ・レベルになると，TOUT端子はハイ・レベルになります．（TCLKnパルスは無関係）
	トリガ注	トリガの次のTCLKnパルスでカウント数転送が起こります．
カウント数書き込み		現在の動作に影響しませんが，現在の方形波の半周期終了時にカウント数転送が起こり，同時にTOUT端子がロウ・レベルになります．
カウント数転送とカウント		モード指定に続くカウント数の書き込みの次のTCLKnパルスで転送されます．その後は現在の半周期終了時に転送が起こり，TOUT端子出力が反転します．また，トリガの次のTCLKnパルスでも転送が行われます． カウント数Nが偶数か，奇数かによって動作が異なり，偶数の場合は，転送されてから2ずつデクリメントされ，カウント値が2になると，次のTCLKnパルスで，転送が起こりTOUT端子の状態が反転します．これを半周期として以後この動作が繰返し行われます． 奇数の場合は，N－1が転送され，2ずつデクリメントされます．TOUT端子がハイ・レベルである半周期はカウント値が0の状態までですが，次のTCLKnパルスで再びN－1が転送され，TOUTがロウ・レベルになった半周期はカウント値が2の状態までしか続きません．このためハイ・レベルの状態の方が1TCLKn分だけ長くなっています．
カウント・ゼロ		カウント数が奇数の場合にのみ起こり得ます．
最小カウント数		2

注　モード指定直後でまだカウント数が書かれていない状態，およびリード/ライト・2バイト・モードで1バイトしか書かれていない状態ではトリガは無視されます．

図 6－86　モード 3 動作例



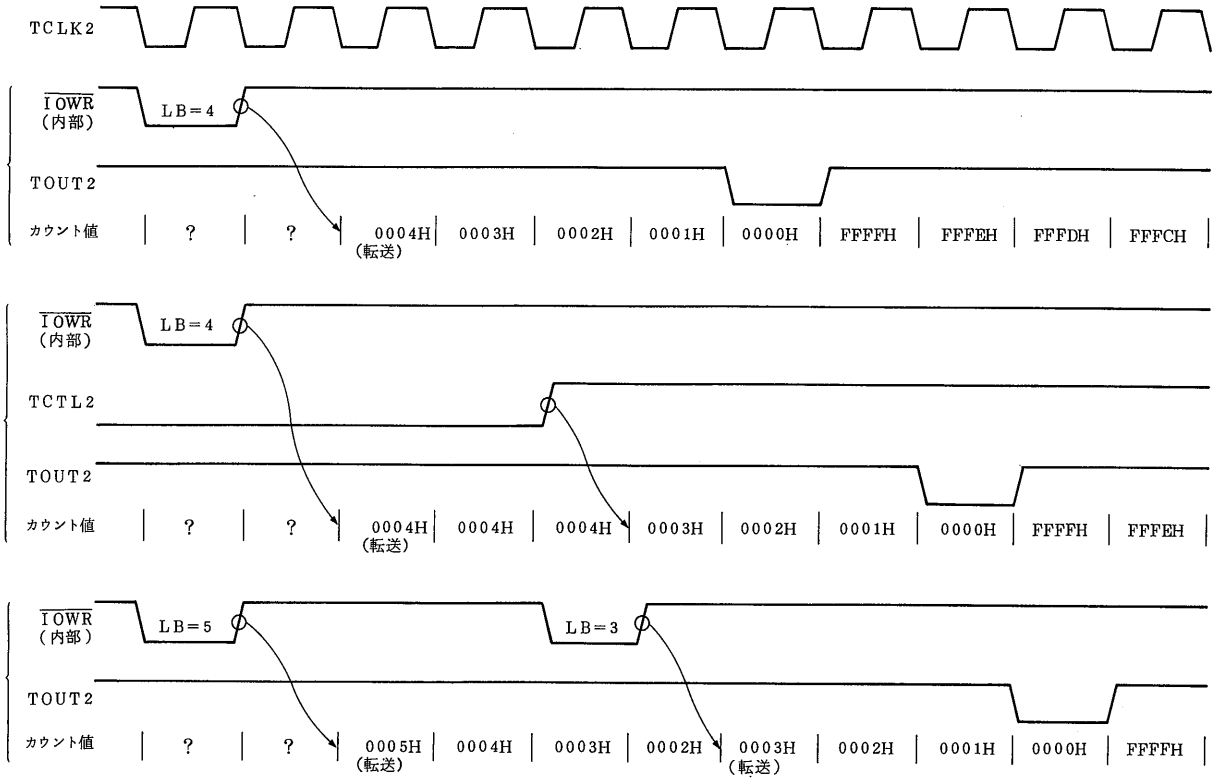
(5) モード4：ソフトウエア・トリガード・ストローブ

指定したカウントが終了すると1TCLKnパルス分だけTOUT端子がロウ・レベルになります．この機能はカウント数転送1回につき1度だけ働きます．

表6－28　モード4動作

項目		内容
TOUT初期状態		ハイ・レベル
TCTL入力	ハイ・レベル	カウント状態
	ロウ・レベル	カウント禁止状態
カウント数書き込み		カウント数が書き込まれると，次のTCLKnパルスでカウント数が転送されます．リード／ライトが2バイト・モードの場合は，2バイト目が書き込まれたときに上記の動作が起こります．
カウント数転送とカウント		カウント数が書き込まれた次のTCLKnパルスで転送が行われます．TCTLがハイ・レベルならば，その次のTCLKnパルスからデクリメントが開始され，TCTLがロウ・レベルならばTCTLがハイ・レベルになった次のTCLKnパルスからデクリメントされます．
カウント・ゼロ		TOUT端子が1TCLKnの間ロウ・レベルになり，再びハイ・レベルに戻ります．カウント動作は止まらずに，バイナリならFFFFH，BCDなら9999へカウント・ダウンされていきます．
最小カウント数		1

図6－87　モード4動作例



(6) モード5：ハードウエア・トリガード・ストローブ（リトリガブル）

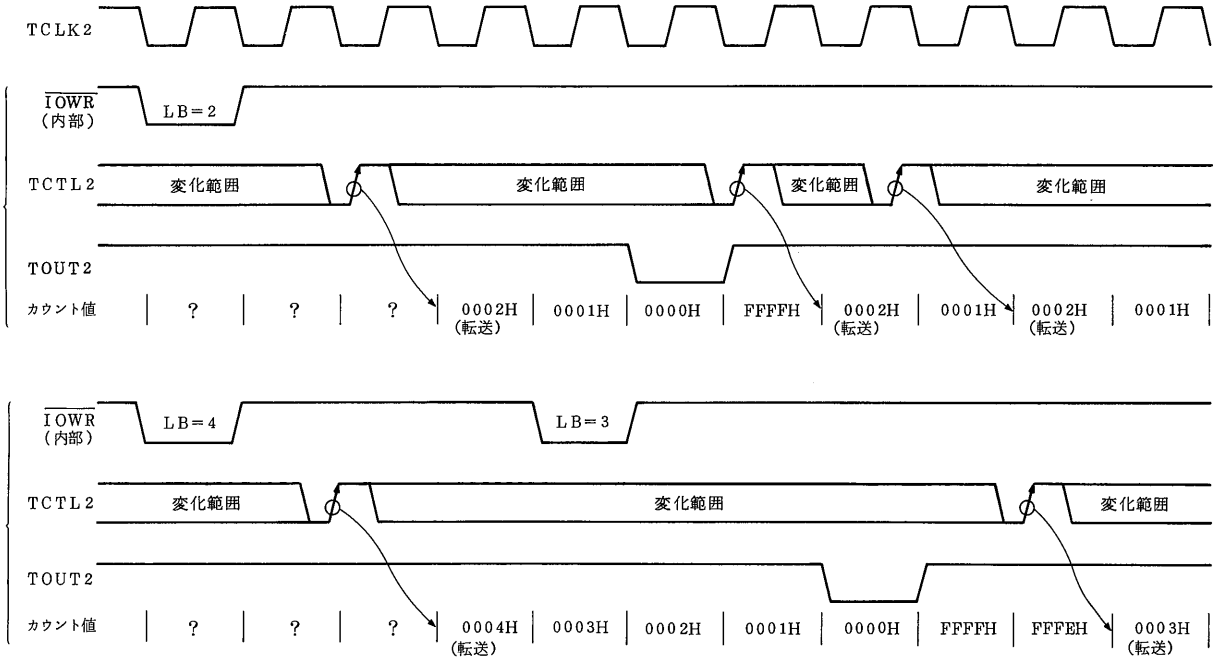
モード4と同じ動作ですが，ＴＣＴＬ入力により開始され，再トリガ可能です．この機能はカウント数転送１回につき１度だけ働きます．

表６－２９　モード５動作

項目		内容
ＴＯＵＴ初期状態		ハイ・レベル
ＴＣＴＬ入力	トリガ注	トリガの次のＴＣＬＫｎパルスでカウント数転送が起こります．
カウント数書き込み		現在の動作に影響せずに書き込まれます．
カウント数転送とカウント		トリガがあると，次のＴＣＬＫｎパルスで転送が行われます．デクリメントは転送の次のＴＣＬＫｎパルスから開始されます．ので，カウント数がＮならばＴＯＵＴ端子は，トリガから（Ｎ＋1)ＴＣＬＫｎパルスの間はロウ・レベルになりません．
カウント・ゼロ		ＴＯＵＴ端子が１ＴＣＬＫｎの間ロウ・レベルになり，再びハイ・レベルに戻ります．カウント動作は止まらずに，バイナリならＦＦＦＦＨ，ＢＣＤなら９９９９へとカウント・ダウンされていきます．
最小カウント数		1

注　モード指定直後でまだカウント数が書かれていない状態，およびリード／ライト・２バイト・モードで１バイトしか書かれていない状態ではトリガは無視されます．

図 6－88　モード 5 動作例



6．6．8　注意事項

カウント・モードの切り替えに関して次のことに注意してください．

任意のモードを設定しているカウンタに，再度，モード指定（モード2またはモード3のみ）を行う場合は，TCLKnパルスが必要です．

このTCLKnパルスは，前モードのカウント数設定と新しいモードの設定の間に1パルス（立ち上がり，次に立ち下がるパルス）以上行ってください．

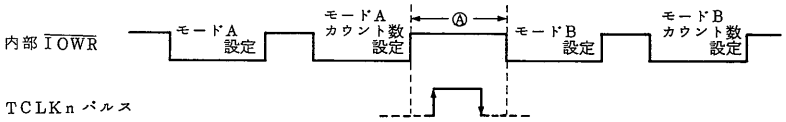
入力しなかった場合には，新しいモードの動作がカウント値を誤る場合があります．

モード切り替えは次のどちらかの方法で行ってください．

・図6－89のⒶ区間に最低1パルス入力する周期のクロックを使用する

・使用クロックの立ち上がり，立ち下がりが最低1回生じるように新モード設定を遅らせる

図6－89　モード切り替えタイミング



備考　モードA：任意のモード

モードB：モード2，またはモード3

6.7 SCU(シリアル・コントロール・ユニット)

SCUは，調歩同期式のシリアル通信機能を提供します．コマンド体系はμPD71051に似ていますが，μPD71051ではコントロール・ワード・レジスタだったものがSCM(シリアル・コマンド・レジスタ)とSMD(シリアル・モード・レジスタ)の2つに分かれています．

6.7.1 特　徴

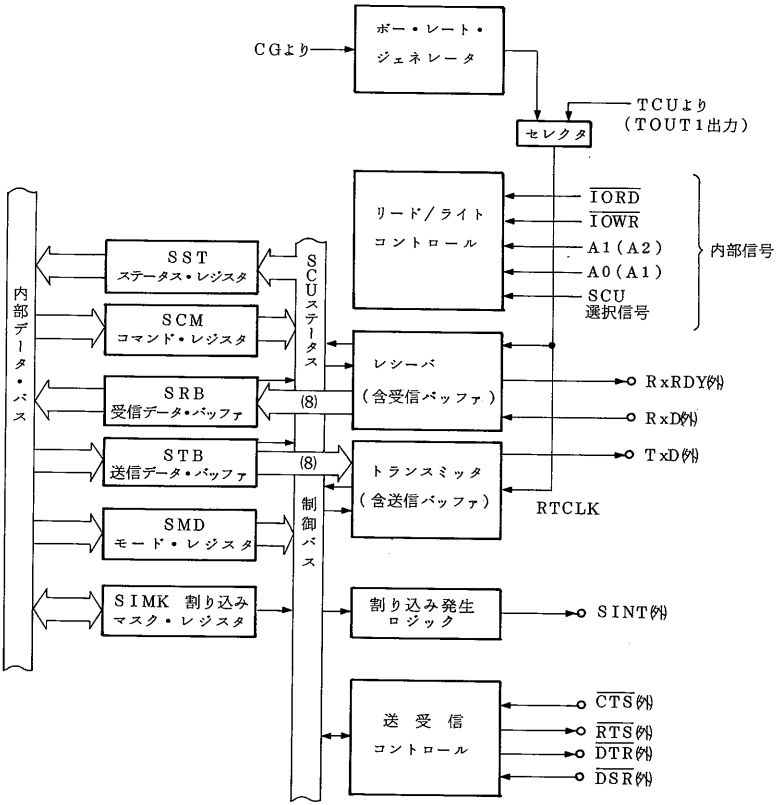
- RS−232−Cプロトコル・サポート($\overline{RTS}$, $\overline{CTS}$, $\overline{DTR}$, $\overline{DSR}$端子内蔵)
- 専用ボー・レート・ジェネレータ内蔵(内部クロック使用)
- ボー・レート・ジェネレータ出力またはタイマ出力を送受信クロックとして選択可能
- 調歩同期のみ
- 入出力端子のマルチプレクスなし
- シリアル割り込み出力(SINT)あり
- クロック・レート：ボー・レート×16，ボー・レート×64
- ボー・レート：DC−500Kbps
- キャラクタ長：7, 8ビット
- 送信ストップ・ビット：1, 2ビット
- ブレーク送信可能
- 自動ブレーク検出あり
- 全二重ダブル・バッファ方式
- パリティ付加／チエック可能
- エラー検出：パリテイ，オーバラン，フレーミングの3種

6.7.2 V50に対する変更箇所

- SRDY → RxRDY, SINT
- RS−232−Cプロトコル・サポート
- ボー・レート・ジェネレータ内蔵
  入力クロックは内部クロック(クロック・ジェネレータより)

内蔵タイマ／カウンタの汎用性を高めるため，SCUは専用のボー・レート・ジェネレータを内蔵しています．また，V40, V50とのコンパチビリティを保つため，TCLKn入力によるボー・レートを設定できるように内蔵タイマ出力と，ボー・レート・ジェネレータ出力から送受信クロックを選択できるようにしています．

6．7．3　ＳＣＵ内部ブロック図



6．7．4　アドレシング

SCUの内部レジスタは，システムI/O領域で設定したOPHAとSULAおよびアドレス信号（A1/A0またはA2/A1）によってアドレスされます．

ただし，BRCのアドレスはシステムI/O領域のFFE9Hに固定されており，OPHAとSULAの内容によるリロケーションはできません．

表6－30　SCU内部レジスタのアドレス

	上位アドレス	下位アドレス			レジスタ	操作
IOAG=1のとき	A15－A8（OPHA）	A7 A6 A5 A4 A3 A2（SULA）	0 0		SRB	リード
					STB	ライト
			0 1		SST	リード
					SCM	ライト
			1 0		SMD	ライト
			1 1		SIMK	リード/ライト
IOAG=0のとき	A15－A8（OPHA）	A7 A6 A5 A4 A3（SULA）	0 0	A0	SRB	リード
					STB	ライト
			0 1		SST	リード
					SCM	ライト
			1 0		SMD	ライト
			1 1		SIMK	リード/ライト
－	1 1 1 1 1 1 1 1	1 1 1 0 1 0 0 1			BRC	リード/ライト

備考　BRCのアドレスは，IOAG，OPHA，SULAの内容に関係なくFFE9Hです．

6．7．5　SCUの初期化

SCUはパワーオン時，またはリセット信号によって以下のような状態に初期化されます．

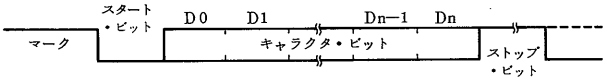
- ボー・レート・ファクタ：×64
- キャラクタ長：7ビット
- パリティ・ビット　：なし
- ストップ・ビット　：1ビット
- 送受信禁止
- ブレーク検出なし
- エラーなし
- バッファ非レディ
- $\overline{\text{RTS}}$，$\overline{\text{DTR}}$端子　：ハイ・レベル

6．7．6　シリアル・データ・フォーマット

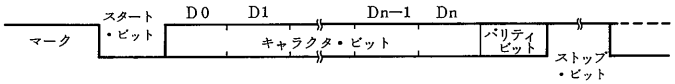
図6－90にSCUが扱うシリアル・データ・フォーマットを示します．CPUがSCUから受けたり，SCUに渡すデータは，キャラクタ・ビットの部分です．キャラクタ・ビットをはさんでいるスタート・ビット，パリティ・ビット，ストップ・ビットは，シリアル通信を実現するための制御情報であり，SCUが自動的に付加（送信時）／削除（受信時）します．

図6－90　シリアル・データ・フォーマット

(a)　パリティなし



(b)　パリティあり



n：6または7
ストップ・ビット：1ビットまたは2ビット

以下に各データ状態を説明します．

(1)　マーク（ハイ・レベル）

データ送信を行っていない状態ではTxD端子はハイ・レベル（マーキング状態）になっています．

(2)　スタート・ビット（ロウ・レベル）

シリアル・データの開始を示す1ビット長のロウ・レベルです．

(3)　キャラクタ・ビット

送信，受信において実際にデータとして扱われる部分で，7または8ビットに設定されます．

(4)　パリティ・ビット

パリティ許可の場合には，キャラクタ・ビットとパリティ・ビットを合わせたビットの中で，1であるビットの数が偶数（偶数パリティ），または奇数（奇数パリティ）になるようにパリティ・ビットが付加／チェックされます．

例	キャラクタ（8ビット）	パリティ・ビット
偶数パリティ	10100101	0
奇数パリティ	10100101	1

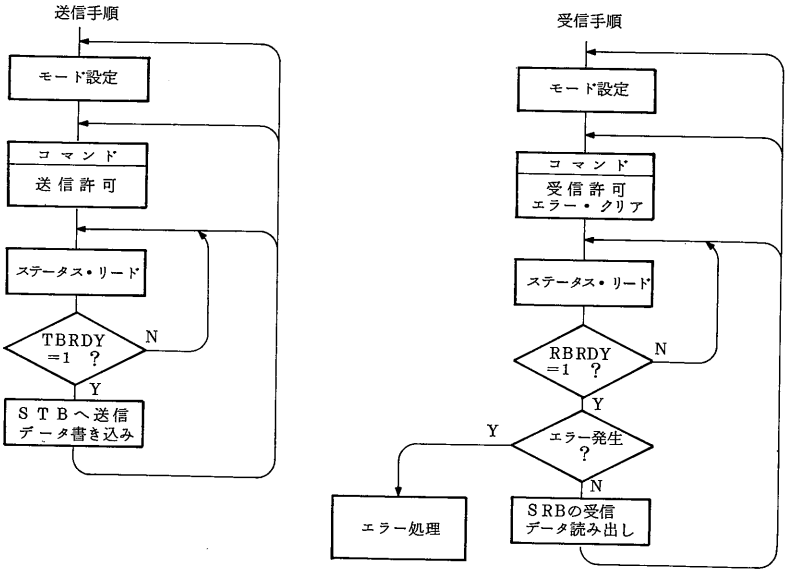
(5)　ストップ・ビット

シリアル・データの終了を示す，1または2ビット長の“1”の状態です．

6．7．7　SCUの操作手順

SCUでシリアル通信を行うための操作手順を図6－91に示します．

図6－91　SCU操作手順



送信を行うには，送信許可にしたあと，STB（送信データ・バッファ）が空であることをステータスで確認してから，送信データをSTBへ書き込みます．

受信を行うには，受信許可にすると同時にステータス中のエラー・フラグをクリアしておきます．そして，SRB（受信データ・バッファ）に受信データが格納されたことをステータスで確認してから，受信データをSRBから読み出します．

図6－91では，ステータスを読み出し，TBRDY，RBRDYフラグを調べながら送受信を行っていますが，割り込みを利用して行うこともできます．

6．7．8　SCUのレジスタ，コマンド

SCUのレジスタのリード／ライト，コマンドの発行は，システムI/O領域で設定したアドレスに対するI/O入出力命令で行い，レジスタ等の選択はA1，A0またはA2，A1で行います．

表6－31　SCUレジスタ，コマンド・アドレス

A1（A2）	A0（A1）	レジスタ	操作
0	0	SRB	リード
		STB	ライト
0	1	SST	リード
		SCM	ライト
1	0	SMD	ライト
1	1	SIMK	リード／ライト
注		BRC	リード／ライト

注　BRCのアドレスはFFE9Hに固定です．

SRBは受信データ・バッファで，CPUはこのバッファから受信データを読み出すことができます．

STBは送信データ・バッファで，CPUは送信するデータをこのバッファに書き込みます．

SSTは，通信状況を示すステータス・レジスタです．SRB，STBの情報や受信エラー情報を持っています．

SCMはコマンド・レジスタで，送信／受信の許可／禁止，エラー・フラグのクリア，ブレーク送信，$\overline{\text{RTS}}$，$\overline{\text{DTR}}$端子の制御を指示します．

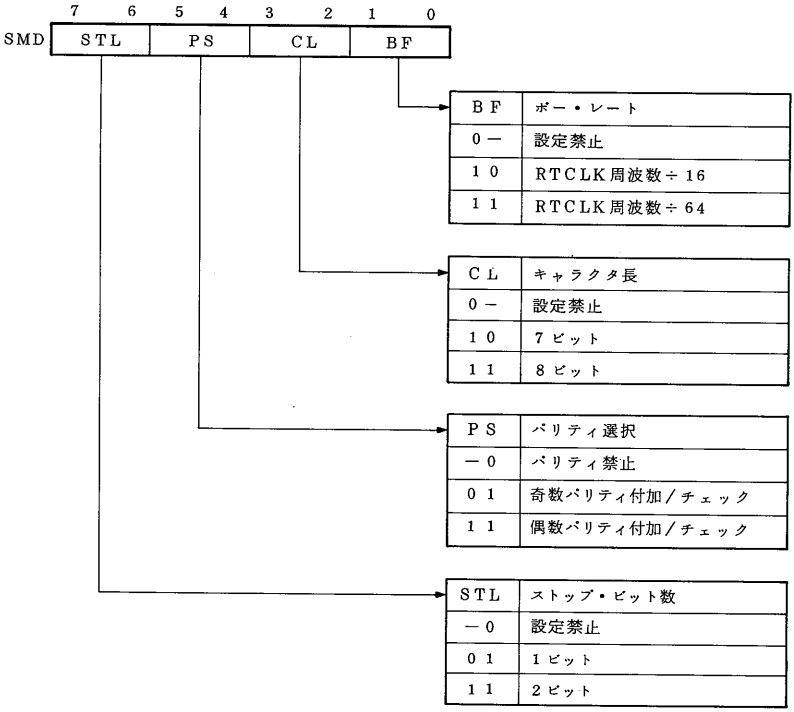
SMDはシリアル・データの設定を行うモード・レジスタです．ボー・レイト，キャラクタ長，パリティ，ストップ・ビット数の設定を行います．キャラクタ長を7ビットにした場合，SRB，STBの下位7ビットが有効になります．

SIMKは，SCUが発生する割り込み要求のマスク制御を行うシリアル割り込みマスク・レジスタです．マスクすると割り込みは発生しなくなります．

BRCは，SCUの専用ボー・レート・ジェネレータの8ビット分周カウンタです．

(1) SMD (シリアル・モード・レジスタ)

図6－92 SMDレジスタ



備考 －は任意

(a) STL

STLビットは，送信時のストップ・ビットの数を1または2に設定します．

(b) PS

PSビットは，パリティ・ビットの設定を行います．パリティ禁止にすると，送信時のパリティ付加，受信時のパリティ・チェックは行われません．パリティ許可の場合には，偶数パリティか奇数パリティかを選択します．

(c) CL

CLビットは，1キャラクタのビット数の設定で，7ビットまたは8ビットの選択を行います．7ビットを選択した場合，SCUはCPUの書き込んだ8ビット・データの下位7ビットを受け取り，SCUがCPUへ出力するデータは上位1ビットが“0”に固定されます．

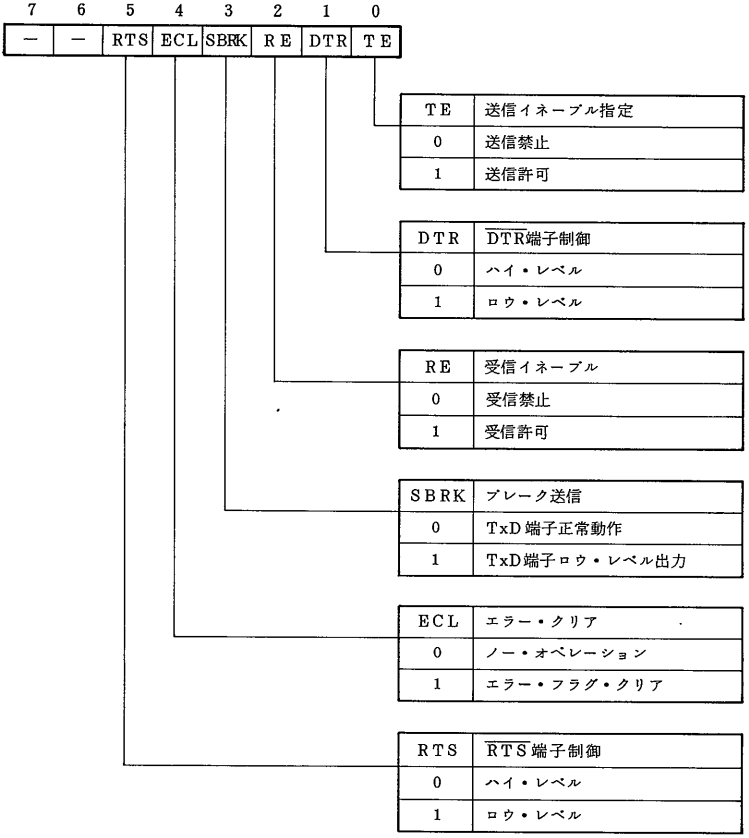
(d) BF

BFビットは，RTCLKの周波数とボー・レートとの関係を規定します．送受信クロックに対してボー・レートを1/16にするのか，1/64にするのか選択します．

ボー・レート設定値については，**6.7.9項**を参照してください．

(2) SCM (シリアル・コマンド・レジスタ)

図6－93 SCMレジスタ



(a) RTS

RTSビットは，汎用の出力端子$\overline{\text{RTS}}$の制御に用います．RTSが“1”ならば$\overline{\text{RTS}}$はロウ・レベル，RTSが“0”ならば$\overline{\text{RTS}}$はハイ・レベルになります．

(b) ECL

ECLビットを“1”にすると，ステータス・レジスタ中のエラー・フラグ(PE,OVE,FE)がすべてクリアされます．特に受信許可にする際には，ECL＝1としてエラー・フラグをクリアする必要があります．

(c) SBRK

SBRKビットは，ブレーク状態の送信に用います．ブレーク送信では，データに関係なくTxD端子にはロウ・レベルが出力されます．また，ブレーク送信機能は送信禁止状態においても有効です．

(d) RE

REビットは受信の禁止／許可を制御します．

(e) DTR

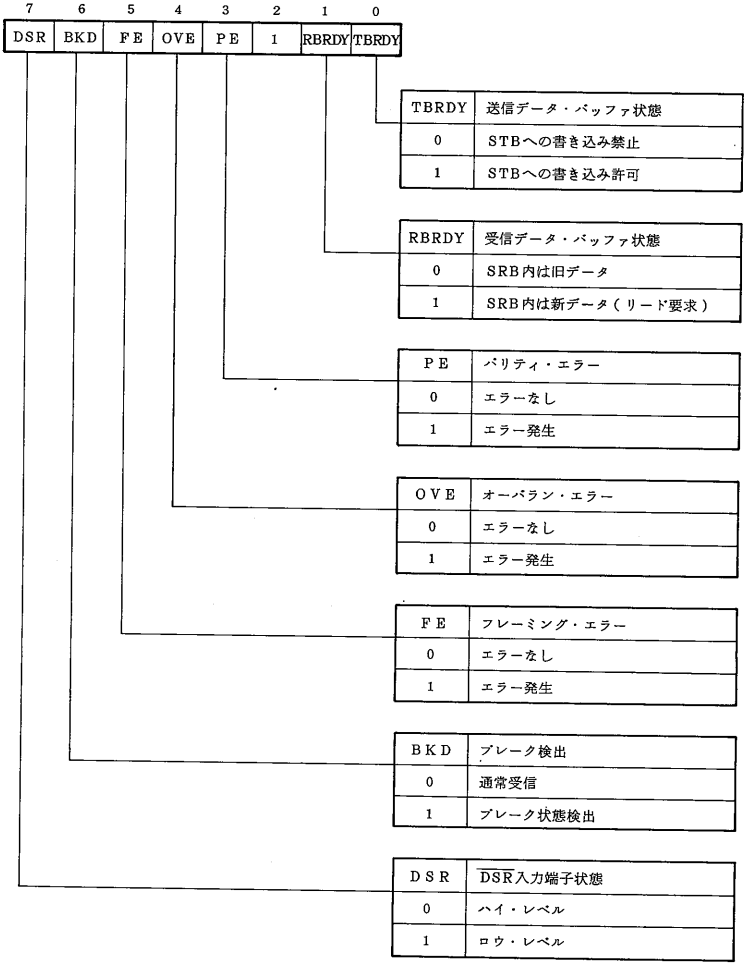
DTRビットは，汎用の出力端子$\overline{\text{DTR}}$の制御に用います．DTRが“1”ならば$\overline{\text{DTR}}$はロウ・レベル，DTRが“0”ならば$\overline{\text{DTR}}$はハイ・レベルになります．

(f) TE

TEビットは，送信の許可／禁止を制御します．送信禁止の場合，TEを“0”にする時点で送信バッファにデータがあると，そのデータは送信されず，バッファに保持されます．その後，再び送信許可になった時点でバッファ内に保持されていたデータが送信されます．送信禁止状態では，TxD端子はハイ・レベル(マーク状態)になっています．このため，書き込んだデータをすべて送信して送信禁止にする場合は，SSTレジスタのTBRDY＝1に設定後，TE＝0に設定してください．

(3) ＳＳＴ（シリアル・ステータス・レジスタ）

図６－９４　ＳＳＴレジスタ



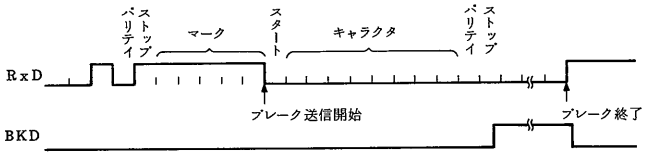
(a) DSR

DSRビットは汎用の入力端子$\overline{DSR}$の状態を示しています．$\overline{DSR}$がロウ・レベルのとき，“1”となります．

(b) BKD

BKDビットは，ブレーク状態を検出したときに“1”となります．すなわち，スタート・ビットからストップ・ビットまで連続してロウ・レベルが入力されたときに発生します．一度セットされたBKDビットはRxDがハイ・レベルにもどるか，またはリセットされるまでクリアされません．図6－95に，RxDに対するBKDビットの動作を示します．

図6－95　ブレーク入力とその検出



(c) FE

FEビットは，ストップ・ビットがあるべきタイミングでハイ・レベルが検出されなかったときに“1”となり，フレーミング・エラーの発生を示します．フレーミング・エラーはブレーク状態の受信時に発生するほか，送信側と受信側のクロックがずれている場合や，伝送路中でデータが変化したときなどに発生します．

(d) OVE

OVEビットは，受信時にCPUによる受信データの読み出しが遅れたときに“1”となり，オーバラン・エラーの発生を示します．このときSRB内の旧データが新データの書き込みによって失われます．

(e) PE

PEビットはパリティが有効なときの受信時に，パリティ・チェックでエラーが発生すると“1”になります．

(f) RBRDY

RBRDYビットは，1キャラクタ分のデータを受信し，そのデータが受信データ・バッファSRBに転送されたとき，すなわち，受信データをCPUが読出し可能になったときに1になります．CPUがSRBを読み出したときにRBRDYは“0”にクリアされます．

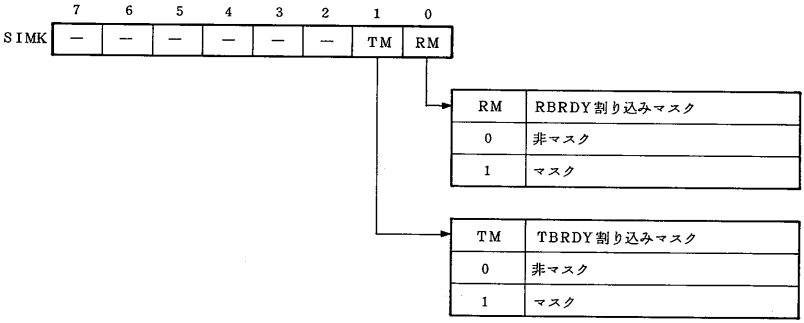
(g) TBRDY

TBRDYビットは，送信データ・バッファSTBが空であるとき，すなわち，CPUからの送信データの書き込みが可能なときに1となります．CPUがSTBにデータを書き込

むとTBRDYは“0”になりクリアされます．なお，TBRDY＝0の状態で送信データを書き込むと，STB内にあるまだ送信されていないデータは破壊されてしまいます．また，送信禁止状態ではTBRDYは“1”にはなりません．

(4) SIMK（シリアル割り込みマスク・レジスタ）

図6－96　SIMKレジスタ



このレジスタは，SCUからの割り込み要求の要因となるRBRDYとTBRDYを独立にマスクします．

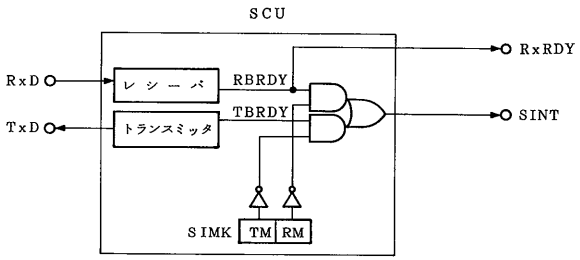
(a) TM

TMビットを“1”にセットすると，TBRDY＝1となっても割り込みは発生しません．

(b) RM

RMビットを“1”にセットすると，RBRDY＝1となっても割り込みは発生しません．

図6－97　TM，RMビットとSINTの関係



6．7．9　送受信ボー・レート

SCUの送受信ボー・レートは，送受信クロック（RTCLK）信号と，SMDレジスタのBFビットによって決定されます．

RTCLKは，専用ボー・レート・ジェネレータ出力とTCUのTOUT１出力とから選択できます．その指定は，システムI/O領域のSCTLレジスタで行います．

TOUT1出力を使用する場合は，TCUのTCT#１動作モードをモード３の方形波ジェネレータに設定してください（**6.6.7**項参照）．

(1)　BRC（ボー・レート・カウンタ）

BRCはSCUの専用ボー・レート・ジェネレータの８ビット分周カウンタです．内部クロック（発振周波数の１/２固定）の分周数を設定します．

BRCは，システムI/O領域のFFE9H番地に割り付けられています．

図6－98　BRC（ボー・レート・カウンタ）

I/Oアドレス	7	6	5	4	3	2	1	0
FFE9H　BRC	D7	D6	D5	D4	D3	D2	D1	D0

(2)　ボー・レート設定

ボー・レートは，次式より決まります．

(a)　ボー・レート・ジェネレータ使用時

$$\text{ボー・レート} = \frac{\text{発振周波数（Hz）}}{\text{BF}\times\text{BRC設定値}} \times \frac{1}{2}$$

備考　BRCレジスタとBRC設定値の関係は次のとおりです．

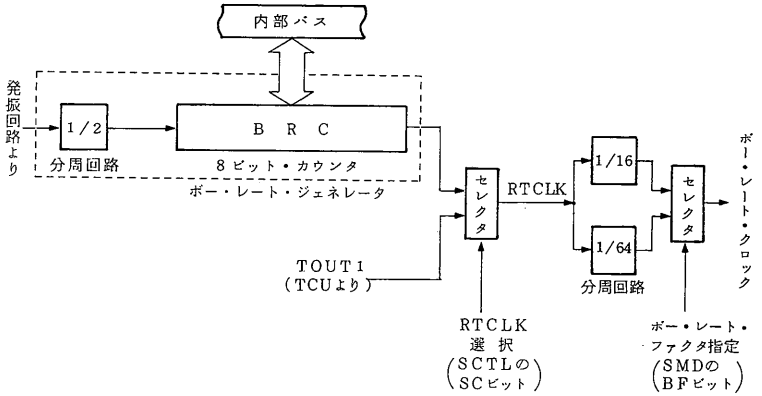
BRCレジスタ	BRC設定値
00H	2
01H	2
02H	2
03H	3
04H	4
⋮	⋮
FFH	255

(b)　TCU使用時

$$\text{ボー・レート} = \frac{\text{TCLK1周波数（Hz）}}{\text{BF}\times\text{TCT\#1設定値}}$$

備考　BF（ボー・レート・ファクタ）はSMDレジスタのBFビットで指定され，16か64の値をとります．

図6－99　ボー・レート・クロック生成図



(3) ボー・レート設定例

ボー・レート・ジェネレータ使用時のボー・レート設定例と，TOUT1出力使用時のボー・レート設定例を表6－32に示します．

表6－32　ボー・レート設定例（1／2）

(a) ボー・レート・ジェネレータ使用時

発振周波数〔X1,X2〕	24.576MHz		29.4912MHz	
ボー・レート・ファクタ	16	64	16	64
ボー・レート〔Baud〕	BRC設定カウント数			
1200	—	160	—	192
2400	—	80	—	96
4800	160	40	192	48
9600	80	20	96	24
19200	40	10	48	12
38400	20	5	24	6

表6－32　ボー・レート設定例（2／2）

(b)　TOUT1出力使用時

TCT#1 入力クロック〔MHz〕	4.9152注		7.3728注	
ボー・レイト・ファクタ RTCLK／ボーレイト	16	64	16	64
ボー・レイト〔Baud〕	TCT#1設定カウント数			
110	2793	698	4189	1047
150	2048	512	3072	768
300	1024	256	1536	384
600	512	128	768	192
1200	256	64	384	96
2400	128	32	192	48
4800	64	16	96	24
9600	32	8	48	12
19200	16	4	24	6
38400	8	—	12	—

注　外部よりの入力クロックです．

備考　TCT#1：モード3に設定

(4)　ボー・レート設定時の注意

SCUは，専用のボー・レート・ジェネレータ，またはTCUから送受信クロック（RTCLK）を入力しますが，内部クロックによって表6－33のように最大ボー・レートが規定されます．

表6－33に示した値以上のボー・レートには設定しないでください．

表6－33　内部クロックと最大ボー・レート

内部クロック	最大ボー・レート
16 MHz	500Kbps
8 MHz	250Kbps
4 MHz	125Kbps
2 MHz	62.5Kbps

6．7．10　シリアル送信／受信

ここでは，SCUによるシリアル・データの送信／受信動作の基本フローについて説明します．

(1) シリアル送信動作

(a) 送信禁止状態ではTxD端子はハイ・レベル（マーク状態）となっており（ブレーク送信時を除く），このときTBRDYは常に“0”です．

(b) SCMにより，送信許可にするとTBRDY=“1”となって，送信データの書き込みが可能になります．

(c) CPUが送信データをSTBに書き込むと，データはいったんトランスミッタへ転送され，スタート・ビット，パリティ・ビット，ストップ・ビットが付加されてTxD端子からシリアル・データとなって送出されます．

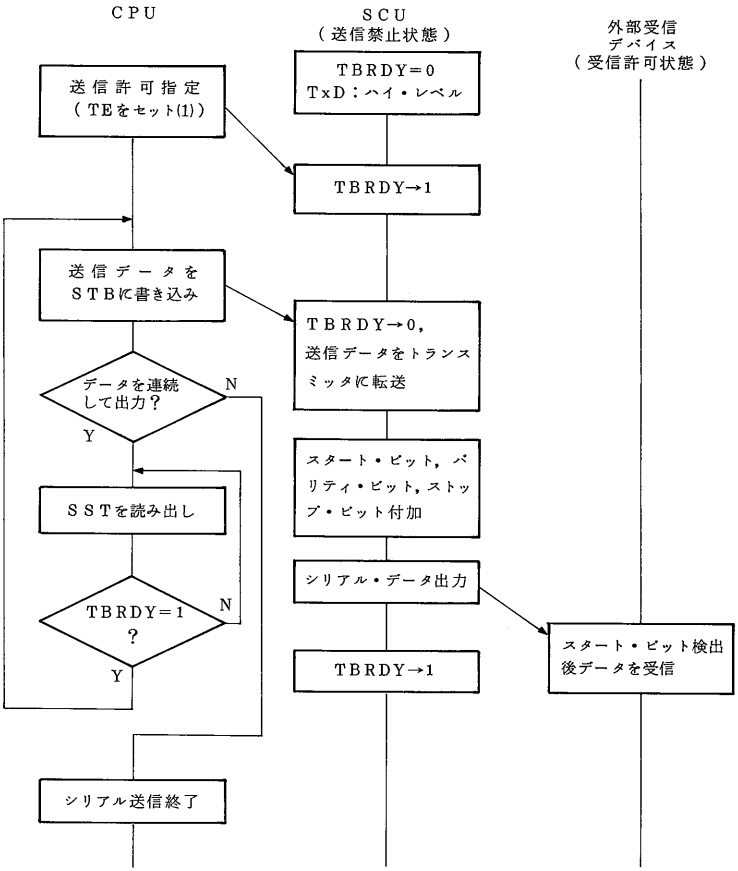
(d) 続けてデータを送信する場合は，TBRDY＝1となっていることを確認してから送信データを書き込みます．TBRDY＝0のときは，前のデータがまだ送信バッファに残っているので，次のデータを書き込んではいけません．

(e) $\overline{\text{CTS}}$とTEビットは同一の動作を行います．

$\overline{\text{CTS}}$とTEビットは，TxD端子そのものを制御しているのではなく，送信データ・バッファ（STB)からトランスミッタへの転送を制御しています．したがって，トランスミッタへ転送されたデータは送信中に送信禁止にしても送信されます．

連続に送信させる場合などで，トランスミッタに転送されたデータの送信が終了していない状態で送信データ・バッファ（STB）にデータを書き込み，TEビットを0にして送信禁止にすると，送信中のデータを送信後送信は停止し送信データ・バッファ（STB）の内容は保持されます．送信許可後，保持されている送信データ・バッファ（STB）内のデータは，トランスミッタに転送され送信されます．

図6－100　シリアル送信動作



備考　TEはSCMのビット0，TBRDYはSSTのビット0です．

(2) シリアル受信動作

(a) データ受信中でないときはＲｘＤ端子はハイ・レベルになっていなければなりません．

(b) 受信許可状態でＲｘＤ端子にロウ・レベルが検出されると，レシーバはこれが有効なスタート・ビットであるかを判別するために立ち下がりから1/2ビット後の位置でＲｘＤ入力のレベルをサンプリングします．このとき，ロウ・レベルが検出されると有効なスタート・ビットとみなし，受信が開始されます．反対にハイ・レベルが検出された場合にはスタート・ビットとはみなさずに，再びロウ・レベルの入力待ち状態に戻ります．

(c) スタート・ビットが検出されたあとは，キャラクタ・ビット，パリティ・ビット，ストップ・ビットのサンプリングが行われ，データが受信されます．

(d) 1キャラクタ分のデータが受信されると，データはレシーバからＳＲＢへ転送され，ＲＢＲＤＹ＝1となってＣＰＵにデータの読み出しを要求します．

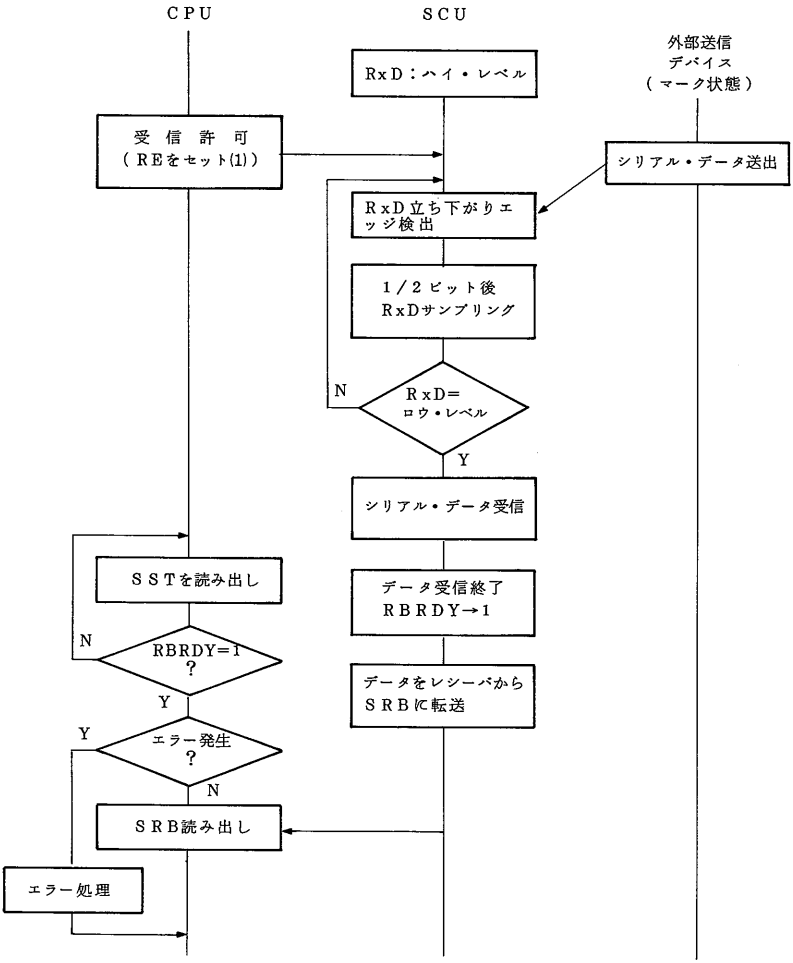
(e) ＲＢＲＤＹ＝“1”の状態でＣＰＵがデータを読み出さないでいると，次のデータがＳＲＢへ転送され，前のデータが消されます．このときＯＶＥ＝1となり，オーバラン・エラーが発生したことを示します．

(f) パリティが有効な場合には，パリティ・チェックを行い，エラーが検出されると，ＰＥ＝1になります．

(g) ストップ・ビットのあるべきタイミングでハイ・レベルが検出されると，次のデータのスタート・ビット待ちとなりますが，ハイ・レベルが検出されないと，ＦＥ＝1となり，フレーミング・エラーが発生したことを示します．

データ受信時にいかなるエラーが発生しても受信動作は停止しませんので，エラー発生時はソフトウエアで対処する必要があります．

図6－101　シリアル受信動作



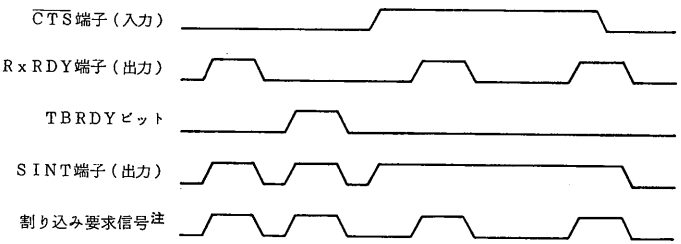
備考　REはSCMのビット1，RBRDYはSSTのビット1です．

6．7．11　注意事項

$\overline{CTS}$端子による送信禁止制御に関して次のことに注意してください．

SIMKレジスタのTMビット＝0のときに，$\overline{CTS}$端子をインアクティブ（ロウ・レベル→ハイ・レベル：送信禁止にする）にするとTBRDYビット＝0，RBRDYビット＝0にもかかわらず，SINT端子がアクティブになります．

$\overline{CTS}$端子とSINT端子の関係は次のとおりです．



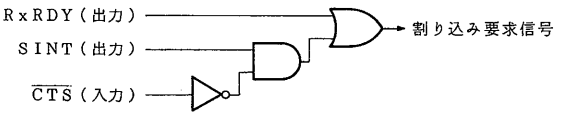
注　下記(3)の論理回路例を参照してください．

したがって，次のいずれかの方法で$\overline{CTS}$端子による送信禁止制御を行ってください．

(1)　$\overline{CTS}$端子入力をロウ・レベル固定とし，送信禁止の制御はTEビットで行ってください．ただし，外部端子から，送信禁止の制御はできません．

(2)　SIMKレジスタのTMビット＝1にすることでマスクしてください（このとき，TBRDYビットをポーリングして送信制御してください）．

(3)　外付け回路を設けてください．論理回路例を次に示します．



上記の回路例の場合，次の点に注意してください．

- SIMKレジスタのRMビット＝1にしてください．
- この回路におけるロジックの遅延は合計10nsと見込んでおり，10ns以上の場合，$\overline{CTS}$入力をCLKOUTの立ち下がりで同期する必要があります．
- 割り込み要求信号をINTPn端子に入力する場合，不完全割り込みが発生する可能性があります．

6.8 BAU(バス・アービトレーション・ユニット)

BAUは，バス・マスタ間でバス使用権の調停を行います．

バス・マスタ（バスの制御権を有することができるもの）には次の4つがあります．

優先順位に従い，以下の4つがバス・マスタとしてバス使用権を得ます．

表6－34　バス・マスター覧

バス・マスタ	バス・サイクル
CPU	プログラムのフェッチ， データのリード/ライト
DMAU	DMAサイクル
REFU	リフレッシュ・サイクル
外部バス・マスタ （HLDRQ端子入力）	外部デバイスが駆動する バス・サイクル

6.8.1 V50に対する変更箇所

内部I/Oのアクセス時は，2クロックのウエイト・ステートを挿入します．

6.8.2 優先順位

各バス・マスタ間の優先順位は以下のようになっています．

高い	CPU（BUSLOCKプリフィクス付きのとき）
↑	REFU（最高優先：一定要求数に達したとき）
	DMAU
	HLDRQ端子
↓	CPU（通常のCPUサイクル）
低い	REFU（最低優先：サイクル・スチール）

REFUには2種類の優先順位があります．通常は最低優先で，完全にバスがアイドル状態でなければリフレッシュ・サイクルを起動できません．しかし，保留されたリフレッシュ要求が7つ以上になると，最高優先になり，バス使用中のバス・マスタに対してバスの明け渡しを要求します．

ただし，CPUがBUSLOCKプリフィクス付きの命令を実行中は，ほかのいかなるバス・マスタもバスを使用できなくなるため，リピート付きのブロック命令など，実行時間の長い命令にBUSLOCKを付ける際にはリフレッシュ・サイクルがダイナミックRAMのリフレッシュ期間を越えないようにしてください．

6．8．3　バス待ち動作

前項のように，各バス・マスタには優先順位がつけられています．優先順位の低いバス・マスタがバスを使用しているときに，より優先順位の高いバス・マスタのバス使用要求が起きた場合，優先順位の低いバス・マスタはただちにバスを開放しなくてはなりません．

CPU，DMAU，REFUなどのV53に内蔵されているバス・マスタは，通常は図6－102に示すように現在実行中のバス・サイクルが終了次第，バスを開放します．しかし，HLDRQ端子に接続した外部バス・マスタやカスケードの外部DMAコントローラなどの場合は，図6－103のようになります．V53は，アクノリッジ信号（HLDAK）をインアクティブにしてバスの返還を要求しますので，バス使用権を有している外部バス・マスタは，それを受けてバス・ホールド要求信号（HLDRQ）を取り下げてバスを開放するようにしてください．V53内部の優先順位の高いバス・マスタは，バス・ホールド要求信号が取り下げられるまで待たされています．

このような状態をバス待ち動作といいます．この状態が長すぎるとDRAMのリフレッシュ動作などが影響を受けます．

図6－102　内部バス・サイクル

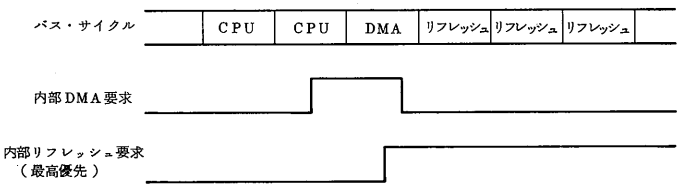
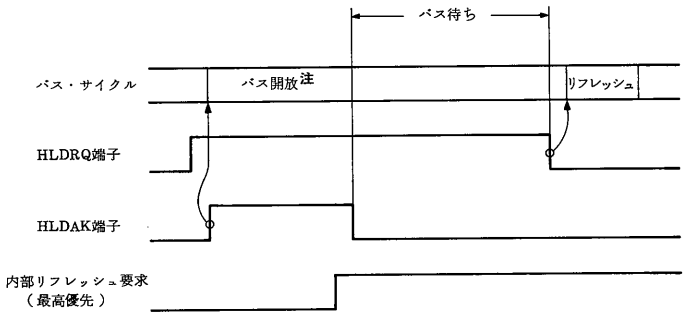


図6－103　バス待ち動作



注　V53がバスを開放し，バス使用権が与えられた外部バス・マスタがバスを使用可能な期間．

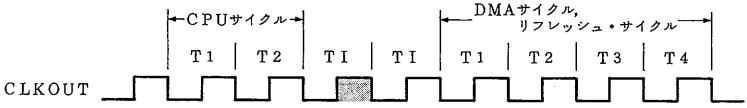
6．8．4　バス使用権の切り替えタイミング

バス使用権の切り替えは，各バス・サイクルおよびアイドル・ステート（バス・サイクルの始まる直前のTIステートは除く）で行われます．各CPUバス・サイクルでは，HLDRQ，DMARQのサンプリングをT2ステートの立ち上がりで行います．

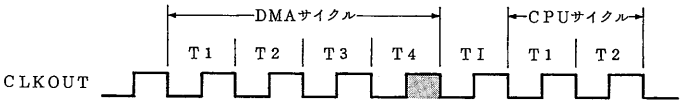
CPUサイクルでは，バス・サイクルに続くアイドル・ステートでバス使用権の切り替えを行います．DMAサイクルではT4ステートでバス使用権の切り替えを行います．リフレッシュ・サイクルではT3ステートでバス使用権の切り替えを行います．ただし，CPUサイクル→CPUサイクルまたはDMAサイクル→DMAサイクルのように同一のバス・サイクルが連続する場合，バス・サイクルの間にはアイドル・ステートは挿入されません．図6－104にバス使用権の切り替えタイミングを示します．

図6－104　バス使用権の切り替えタイミング（1／3）

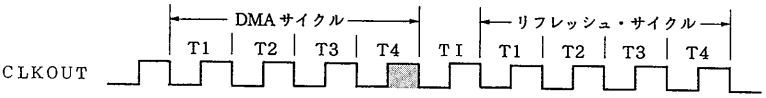
(a)　CPUサイクル→DMAサイクル，リフレッシュ・サイクル



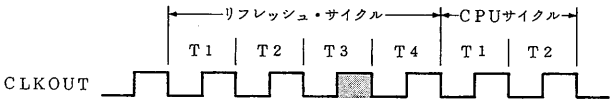
(b)　DMAサイクル→CPUサイクル



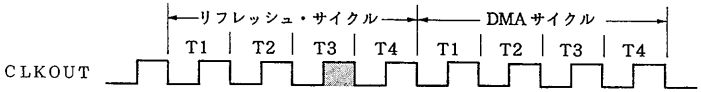
(c)　DMAサイクル→リフレッシュ・サイクル



(d)　リフレッシュサイクル→CPUサイクル



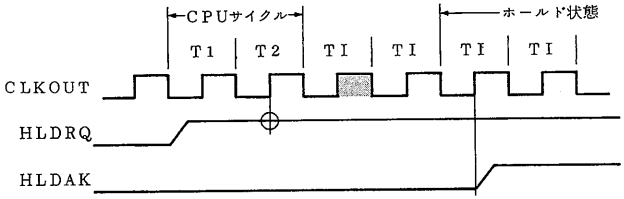
(e)　リフレッシュ・サイクル→DMAサイクル



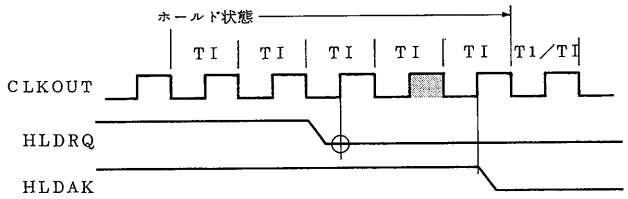
備考　▒▒はバス使用権の切り替えを行うタイミングを示します．

図６－１０４　バス使用権の切り替えタイミング（２／３）

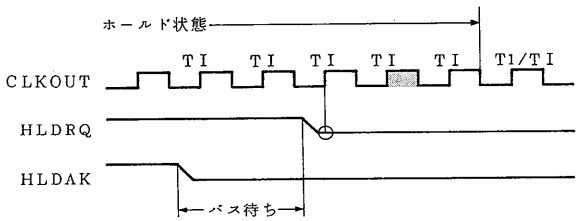
(f)　ＣＰＵサイクル→ホールド状態



(g)　ホールド状態→ＣＰＵサイクル



(h)　ホールド状態→バス待ち



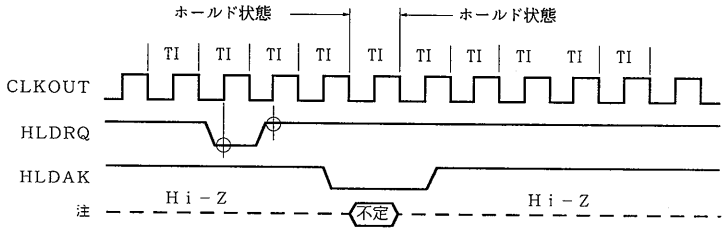
備考１．▒▒はバス使用権の切り替えを行うタイミングを示します．

２．○印はサンプリングされるタイミングです．

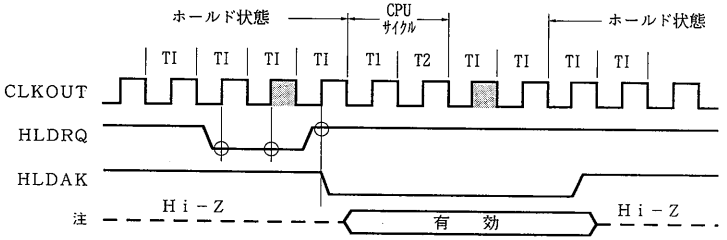
図６－１０４　バス使用権の切り替えタイミング（３／３）

(i)　ホールド状態→ＣＰＵサイクル→ホールド状態

★　・ＨＬＤＲＱのインアクティブを１クロックだけサンプリングした場合（ＣＰＵサイクル起動しない）



・ＨＬＤＲＱのインアクティブを２クロックだけサンプリングした場合（ＣＰＵサイクル起動）



注　A0－A23,R／$\overline{W}$,M／$\overline{IO}$,BUSST0－BUSST2,$\overline{UBE}$

備考 1.　▒▒はバス使用権の切り替えを行うタイミングを示します．

2.　○印はサンプリングされるタイミングです．

6．8．5　注意事項

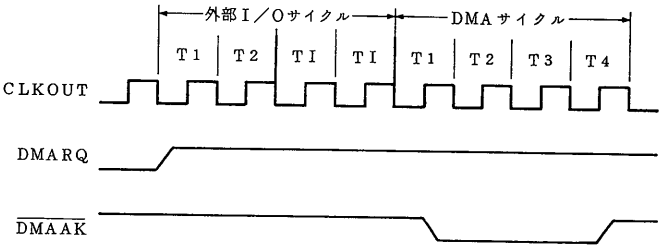
次の３つのサイクルの直後に，アービトレーションが起こるようなバス・サイクルを実行する場合，アイドル・サイクル数は２になります（通常は６サイクル）．

・外部Ｉ／Ｏサイクル
・内部Ｉ／Ｏサイクル
・２回目の割り込みアクノリッジ・サイクル
・コプロセッサ・サイクル

対象となるバス・サイクルは次のとおりです．

・DMAサイクル
・最高優先リフレッシュ・サイクル
・バス・ホールド・サイクル

例　外部Ｉ／Ｏサイクル直後にDMAサイクルが挿入され，外部Ｉ／Ｏサイクルのアイドル・サイクル数が２となる例を以下に示します．



6．9　CG(クロック・ジェネレータ)

CGブロックは，X1,X2端子に接続されたクリスタル／発振器の周波数の１／２のクロック信号を発生し，CLKOUT端子に供給します．また，発振周波数の１／４固定クロック信号をPCLKOUT端子に発生します．

CLKOUTと同一クロックをCPU，DMAU，REFU，SCUに内部クロックとして供給するとともに，発振周波数の１／２のクロックをボー・レート・カウンタに，また，１／４，１／８，１／16,1/32（TCKSで選択）のクロックをタイマに供給します．

さらに，インストラクション・サイクル時間可変機能として，コマンドにより，内部クロックの周波数を下げ，消費電力を低減できます．

6．9．1 特　徴

- X1，X2入力
- CLKOUT出力
- PCLKOUT出力
- インストラクション・サイクル時間可変機能あり

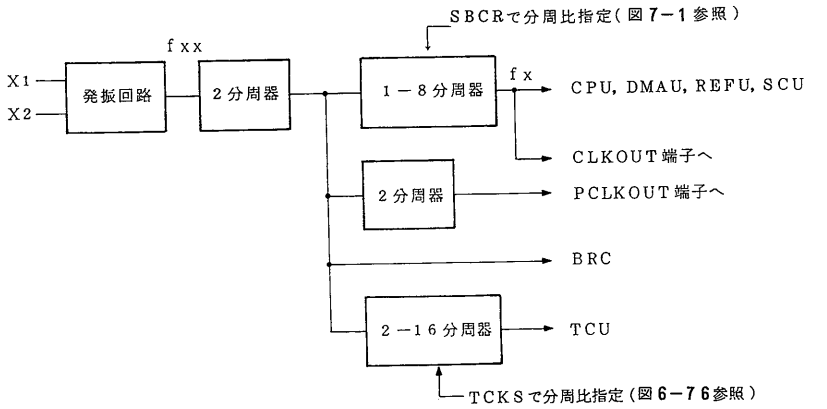
6．9．2 V50に対する変更箇所

- PCLKOUT（入力の1／4）出力端子追加
- インストラクション・サイクル時間可変機能の追加

低速デバイスに対するクロック供給のためにPCLKOUTを採用しています．また，HALT命令によらずに，低消費電力化を図るためにインストラクション・サイクル時間可変機能を採用しています．

によらずに，低消費電力化

図6－105　CG内部ブロック図



注意　インストラクション・サイクル時間可変機能により，内部クロックの分周数を変更した場合は，CLKOUT端子の状態も同時に変化します（入力クロックの1／2－1／16）．インストラクション・サイクル時間可変機能により，内部クロックの分周数を変更しても，PCLKOUT端子は変化しません．常に発振周波数の1／4です．

第7章　スタンバイ機能

μPD70236はスタンバイ機能として，HALT命令によるプログラム待機中のスタンバイ機能と，インストラクション・サイクル時間を切り替えることによるプログラム実行中のスタンバイ機能を備えています．

7.1　特　　徴

・HALTモード

HALT命令の実行によって，CPU内部（HALTモード解除用回路を除く）へのクロックを停止します．

・STOPモード

HALT命令の実行によって，CPUおよび内部I／Oへのクロックをすべて停止します．

・インストラクション・サイクル時間可変機能

内部システム・クロック周波数を，発振周波数の1／2，1／4，1／8，1／16に分周指定可能です．

7.2　V50に対する変更箇所

システムI／O領域にSBCR（スタンバイ・コントロール・レジスタ）をもち，HALT命令により，HALTモードに入るか，STOPモードに入るかを切り替えることができます．

また，発振安定時間の設定，内部クロック分周数の設定を行うことができます．

7.3 SBCR(スタンバイ・コントロール・レジスタ)

V53のスタンバイ機能の制御は，システムI／O領域内のSBCRレジスタによって行います．

表7－1　SBCRのアドレスと操作

レジスタ	I／Oアドレス	操　作
SBCR	FFF1H	リード／ライト

図7－1　SBCR（スタンバイ・コントロール・レジスタ）

I／Oアドレス　FFF1H　SBCR

7	6	5	4	3	2	1	0
－	－	－	CLKC		WT		STOP

STOP	HALTモード／STOPモードの切り替え
0	HALT命令の実行によりHALTモードに入る
1	HALT命令の実行によりSTOPモードに入る

WT	発振安定時間指定
00	発振安定時間＝2^{21}／発振周波数（65.53ms）
01	発振安定時間＝2^{20}／発振周波数（32.76ms）
10	発振安定時間＝2^{19}／発振周波数（16.38ms）
11	発振安定時間＝2^{18}／発振周波数（8.19ms）

備考　（　）内は発振周波数＝32MHz時の値です．

CLKC	内部クロック周波数（fx）の指定
00	fx＝発振周波数×1／2
01	fx＝発振周波数×1／4
10	fx＝発振周波数×1／8
11	fx＝発振周波数×1／16

(a) ＣＬＫＣ

インストラクション・サイクル時間の設定を行ないます．発振周波数に対して1／2，1／4，1／8，1／16のクロックから1つを選択し，これをCLKOUTおよびCPU，DMAU，REFU，SCUに内部クロックとして供給します．

(b) ＷＴ

発振安定時間とは，割り込み（$\overline{\text{NMI}}$入力，ＩＮＴＰｎ入力）によって，ＳＴＯＰモードを解除するときに，振動子の発振を再開させ，発振が安定するまでの時間をいいます．発振が安定したのちに割り込み処理ルーチンに分岐します．

Ｖ５３は，この発振安定時間指定による期間，動作を待たせることができます．表７－２に発振周波数による発振安定時間を示します．

発振安定時間は，振動子の種類，特性により適切な時間に設定してください．

表７－２　発振周波数による発振安定時間

ＷＴ		発振安定時間［ｍｓ］			
		f_{xx}＝32MHz	f_{xx}＝25MHz	f_{xx}＝20MHz	f_{xx}＝10MHz
００	$2^{21}/f_{xx}$	65.53	83.88	104.85	209.71
０１	$2^{20}/f_{xx}$	32.76	41.94	52.42	104.85
１０	$2^{19}/f_{xx}$	16.38	20.97	26.21	52.42
１１	$2^{18}/f_{xx}$	8.19	10.48	13.10	26.21

(c) ＳＴＯＰ

ＳＴＯＰビットは，ＨＡＬＴ命令を実行したときに入るスタンバイ機能のモードを設定します．

7.4 ホールト・アクノリッジ・サイクル

表7－3　各端子の状態

端子名	入出力	端子状態	端子名	入出力	端子状態
A23-A0	3ステート出力	L	RESOUT	出力	L
D15-D0	3ステート入出力	注1	X2, X1	入力	－
$\overline{UBE}$	3ステート出力	H	CLKOUT	出力	非固定
R/$\overline{W}$	3ステート出力	L	PCLKOUT	出力	非固定
M/$\overline{IO}$	3ステート出力	L	TCLK	入力	－
★ BUSST2-BUSST0	3ステート出力	注2	TCTL0-TCTL2	入力	－
★ $\overline{BCYST}$	3ステート出力	注3	TOUT0-TOUT2	出力	非固定
$\overline{DSTB}$	3ステート出力	H	INTP0-INTP7	入力	－
$\overline{MRD}$	3ステート出力	H	$\overline{INTAK}$	出力	H
$\overline{MWR}$	3ステート出力	H	TxD	出力	非固定
$\overline{IORD}$	3ステート出力	H	RxD	入力	－
$\overline{IOWR}$	3ステート出力	H	RxRDY	出力	非固定
$\overline{BUFEN}$	3ステート出力	H	SINT	出力	非固定
$\overline{BUSLOCK}$	出力	注4	$\overline{RTS}$	出力	非固定
$\overline{READY}$	入力	－	$\overline{CTS}$	入力	－
$\overline{BS8}$/BS16	入力	－	$\overline{DTR}$	出力	非固定
AEX	出力	注5	$\overline{DSR}$	入力	－
$\overline{REFRQ}$	出力	非固定	$\overline{DMARQ0}$-$\overline{DMARQ3}$	入力	－
HLDRQ	入力	－	$\overline{DMAAK0}$-$\overline{DMAAK3}$	出力	非固定
HLDAK	出力	H/L	$\overline{END}$/$\overline{TC}$	入出力	非固定
$\overline{NMI}$	入力	－	V_{DD}	－	－
$\overline{CPBUSY}$	入力	－	GND	－	－
$\overline{RESET}$	入力	－	IC2, IC1	－	－

注1. ホールト・アクノリッジ・サイクルの最初の2サイクル間は不定，以後はHi－Z.

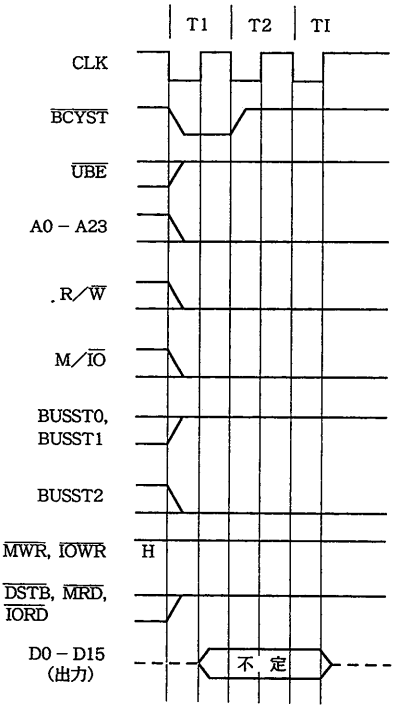
2. BUSST2　：L
   BUSST1, 0：H

3. ホールト・アクノリッジ・サイクルの最初の1クロック間はL，以後はH.

4. 次のいずれかの場合はL，それ以外はH.
   - ホールド時にBUSLOCKプリフィクス付きの命令を実行.
   - BUSLOCKプリフィクス付きのHALT命令を実行.

5. アドレス拡張モード時はH，非拡張モード時はL.

図7－2　ホールト・アクノリッジ・サイクル



7.5 HALTモード

7．5．1 HALTモードの設定

V53はSBCRレジスタのSTOPビット＝“0”に設定し，HALT命令を実行すると，HALTモードに入ります．

7．5．2 HALT状態

HALT状態においては，CPU回路のうち，HALT状態の解除に必要な機能に関する回路，およびバス・ホールド制御機能に関する回路にのみ内部クロックを供給し，ほかのCPU回路にはクロックを停止します．この結果，通常動作時の約1／5に消費電力を低減できます．ただし，HALT中も割り込み機能，DMA機能，タイマ／カウンタ機能，バス・ホールド機能およびリフレッシュ機能などの周辺機能にはクロックが供給されており，HALT中の動作を可能にしています．各周辺機能が動作していない場合の出力端子は表7－4に示すような所定の状態に固定されます．

表7－4　ＨＡＬＴモード時の端子状態（周辺機能非動作の場合）

端子名	入出力	端子状態	端子名	入出力	端子状態
A23－A0	3ステート出力	L	RESOUT	出力	L
D15－D0	3ステート入出力	注1	X2, X1	入力	－
$\overline{UBE}$	3ステート出力	H	CLKOUT	出力	非固定
R/$\overline{W}$	3ステート出力	L	PCLKOUT	出力	非固定
M/$\overline{IO}$	3ステート出力	L	TCLK	入力	－
BUSST2－BUSST0	3ステート出力	注2	TCTL0－TCTL2	入力	－
$\overline{BCYST}$	3ステート出力	注3	TOUT0－TOUT2	出力	非固定
$\overline{DSTB}$	3ステート出力	H	INTP0－INTP7	入力	－
$\overline{MRD}$	3ステート出力	H	$\overline{INTAK}$	出力	H
$\overline{MWR}$	3ステート出力	H	TxD	出力	非固定
$\overline{IORD}$	3ステート出力	H	RxD	入力	－
$\overline{IOWR}$	3ステート出力	H	RxRDY	出力	非固定
$\overline{BUFEN}$	3ステート出力	H	SINT	出力	非固定
$\overline{BUSLOCK}$	出力	注4	$\overline{RTS}$	出力	非固定
$\overline{READY}$	入力	－	$\overline{CTS}$	入力	－
$\overline{BS8}$/BS16	入力	－	$\overline{DTR}$	出力	非固定
AEX	出力	注5	$\overline{DSR}$	入力	－
$\overline{REFRQ}$	出力	非固定	$\overline{DMARQ0}$－$\overline{DMARQ3}$	入力	－
HLDRQ	入力	－	$\overline{DMAAK0}$－$\overline{DMAAK3}$	出力	非固定
HLDAK	出力	H／L	$\overline{END}$/$\overline{TC}$	入出力	非固定
$\overline{NMI}$	入力	－	V_{DD}	－	－
$\overline{CPBUSY}$	入力	－	GND	－	－
$\overline{RESET}$	入力	－	IC2, IC1	－	－

★
★

注 1. ホールト・アクノリッジ・サイクルの最初の2サイクル間は不定，以後はＨｉ－Ｚ.

2. BUSST2　：L

BUSST1，0：H

3. ホールト・アクノリッジ・サイクルの最初の1クロック間はＬ，以後はＨ.

4. 次のいずれかの場合はＬ，それ以外はＨ.

・ホールド時にBUSLOCKプリフィクス付きの命令を実行.

・BUSLOCKプリフィクス付きのHALT命令を実行.

5. アドレス拡張モード時はＨ，非拡張モード時はＬ.

7．5．3　HALTモードの解除

一度設定されたHALT状態を解除するには，次のような外部信号による３種類の方法があります．

(1)　$\overline{\text{RESET}}$入力による解除

$\overline{\text{RESET}}$入力により，V53は初期化され，改めて設定されるPS（FFFFH）と，PC（0000H）により，FFFF0H番地の命令をフェッチすることから通常の動作が再開されます．

(2)　割り込み（$\overline{\text{NMI}}$入力またはINTPn入力）による解除

受け付け可能なマスカブル割り込み要求の発生，または$\overline{\text{NMI}}$入力によりHALTモードは解除され，CPUは割り込み処理プログラムの実行を開始します．割り込み処理プログラムの最後でRETI命令を実行すると，PC，PSおよびPSWがスタックから復帰し，HALT命令の次の命令から命令の実行を再開します．

割り込み禁止状態（IE＝0のとき）の場合は，INTPnを５クロック以上アクティブにすると，HALTモードは解除されHALT命令の次の命令から実行を再開します．

INTPnがマスクされている場合（割り込みマスク・ワード・レジスタ（IMKW）のM0－M7ビットがセット（１）されている場合）は，INTPn入力によってHALTモードを解除することはできません．

(3)　HLDRQ入力による解除

HLDRQ入力（バス・ホールド要求）があると，ただちにバス・ホールドの応答信号（HLDAK）を発生し，アドレス／データ・バスおよび制御信号をハイ・インピーダンス状態にして，バスの使用権を要求があったバス・マスタに引き渡します．バス・ホールド要求がなくなると，CPUは再びバスの使用権を得ますが，再びスタンバイ状態に戻ります．したがって，バス・ホールドの要求は実質的なスタンバイ状態の解除にはなりません．

7.6 STOPモード

7．6．1 STOPモードの設定

V53はSBCRレジスタのSTOPビット＝“1”に設定し，HALT命令を実行するとSTOPモードにはいります．

7．6．2 STOP状態

STOP状態においては，内部クロックを完全に停止します．この結果，消費電流は3mAに低減されます．STOP中はTCU（TCLK入力使用時）を除き，全機能が停止します．各周辺機能が動作していない場合の出力端子は，HALTモードと同じく表7－5に示すような所定の状態に固定されます．

また，各周辺機能が動作している場合の出力端子は，STOPモード設定のHALT命令実行直前のバス状態を保持します．

なお，STOPモードは，X1，X2端子に発振子を接続した場合に使用してください．

表7－5　STOPモード時の端子状態（周辺機能非動作の場合）

端子名	入出力	端子状態	端子名	入出力	端子状態
A23-A0	3ステート出力	L	RESOUT	出力	L
D15-D0	3ステート入出力	注1	X2, X1	入力	－
$\overline{UBE}$	3ステート出力	H	CLKOUT	出力	L
R/$\overline{W}$	3ステート出力	L	PCLKOUT	出力	L
M/$\overline{IO}$	3ステート出力	L	TCLK	入力	－
★ BUSST2-BUSST0	3ステート出力	注2	TCTL0-TCTL2	入力	－
★ $\overline{BCYST}$	3ステート出力	注3	TOUT0-TOUT2	出力	非固定
$\overline{DSTB}$	3ステート出力	H	INTP0-INTP7	入力	－
$\overline{MRD}$	3ステート出力	H	$\overline{INTAK}$	出力	H
$\overline{MWR}$	3ステート出力	H	TxD	出力	非固定
$\overline{IORD}$	3ステート出力	H	RxD	入力	－
$\overline{IOWR}$	3ステート出力	H	RxRDY	出力	非固定
$\overline{BUFEN}$	3ステート出力	H	SINT	出力	非固定
$\overline{BUSLOCK}$	出力	注4	$\overline{RTS}$	出力	非固定
$\overline{READY}$	入力	－	$\overline{CTS}$	入力	－
$\overline{BS8}$/$\overline{BS16}$	入力	－	$\overline{DTR}$	出力	非固定
AEX	出力	注5	$\overline{DSR}$	入力	－
$\overline{REFRQ}$	出力	H	$\overline{DMARQ0}$-$\overline{DMARQ3}$	入力	－
HLDRQ	入力	－	$\overline{DMAAK0}$-$\overline{DMAAK3}$	出力	非固定
HLDAK	出力	H/L	$\overline{END}$/$\overline{TC}$	入出力	非固定
$\overline{NMI}$	入力	－	V_{DD}	－	－
$\overline{CPBUSY}$	入力	－	GND	－	－
$\overline{RESET}$	入力	－	IC2, IC1	－	－

注1. ホールト・アクノリッジ・サイクルの最初の2サイクル間は不定，以後はHi－Z.

2. BUSST2　：L

   BUSST1, 0：H

3. ホールト・アクノリッジ・サイクルの最初の1クロック間はL，以後はH.

4. 次のいずれかの場合はL，それ以外はH.

   ・ホールド時にBUSLOCKプリフィクス付きの命令を実行.

   ・BUSLOCKプリフィクス付きのHALT命令を実行.

5. アドレス拡張モード時はH，非拡張モード時はL.

7．6．3　STOPモードの解除

一度設定されたSTOP状態を解除するには，次の3種類の方法があります．

(1) $\overline{\text{RESET}}$入力による解除

$\overline{\text{RESET}}$入力により，V53は初期化され，改めて設定されるPS（FFFFH）と，PC（0000H）により，FFFF0H番地の命令をフェッチすることから通常の動作が再開されます．$\overline{\text{RESET}}$入力は発振安定時間の期間アクティブにしておく必要があります（$\overline{\text{RESET}}$を途中でインアクティブにした場合，その後の動作は保証されません）．

(2) マスカブル割り込み（INTPn）による解除

受け付け可能な割り込み要求をINTPn端子に入力すると，STOPモードは解除されます．

割り込み許可状態の場合は，STOPモードが解除されると割り込み処理に入りますが，このとき，INTPn端子は割り込みアクノリッジ・サイクルが発行されるまでアクティブに保つ必要があります．途中でインアクティブにした場合は，次の2つの状態のどちらかになります．

(a) STOPモードからHALTモードに移る．

(b) HALT命令の次の命令から実行を再開する．

割り込み禁止状態の場合は，STOPモードが解除され，発振安定時間ののちにHALT命令の次の命令から実行を再開します．このときINTPn端子は発振安定時間の間十分アクティブに保つ必要があります．INTPnを途中でインアクティブにした場合は，STOPモードからHALTモードに移る場合があります．

INTPnがマスクされている場合（割り込みマスク・ワード・レジスタ（IMKW）のM0－M7ビットがセット（1）されている場合），STOPモードを解除することはできません．

(3) ノンマスカブル割り込みによる解除

$\overline{\text{NMI}}$端子がアクティブ（ロウ・レベル）であると，STOPモードが解除されて，発振安定時間ののちにNMI処理を行います．ただし，NMI端子は発振安定時間の間（NMIの8番地のベクタ・リード・サイクルまで）アクティブ・レベルを保持しておく必要があります．途中でインアクティブにした場合は，STOPモードからHALTモードに入ることがあります．

注意　HALT命令を実行する前から$\overline{\text{NMI}}$端子をアクティブにしておくと，STOPモードには入りません．

7.7　インストラクション・サイクル時間可変機能

V53はSBCRレジスタに内部クロックの分周数を設定することにより，動作周波数を低くして，消費電力を低減する機能を有しています．

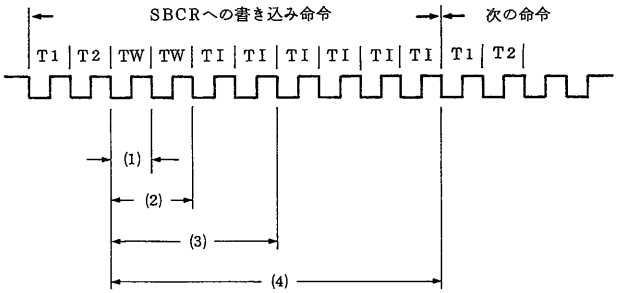
表7－6　分周数と消費電流の関係

発振周波数	分周数（CLKC）注1	内部クロック（fx）	消費電流（MAX.値）注2
32 MHz	1／2　（00）	16 MHz	248 mA
32 MHz	1／4　（01）	8 MHz	144 mA
32 MHz	1／8　（10）	4 MHz	92 mA
32 MHz	1／16　（11）	2 MHz	66 mA

注1. 内部クロックがV53の最低動作周波数（2MHz）以下にならないように分周数を設定してください．ただし，μPD70236（A）の場合，最低動作周波数は2.5 MHzです．CLKCはSBCRレジスタのビット4，3です．

2. μPD70236の場合　消費電流　＝　13 mA　×　fx（MHz）＋　40 mA
μPD70236（A）の場合　消費電流　＝　15 mA　×　fx（MHz）＋　40 mA

注意　動作周波数を設定してから実際に動作周波数が変化するタイミングは不定ですが，下図のようにSBCRへの書き込みサイクルのT2の次のクロックから8クロック以内に変化します．



(1)　分周数1／16からほかの分周数を設定した場合に動作周波数が変化する範囲
(2)　分周数1／8からほかの分周数を設定した場合に動作周波数が変化する範囲
(3)　分周数1／4からほかの分周数を設定した場合に動作周波数が変化する範囲
(4)　分周数1／2からほかの分周数を設定した場合に動作周波数が変化する範囲

第8章　リセット機能

$\overline{\text{RESET}}$端子に6クロック以上のロウ・レベルを入力したのちにハイ・レベルに戻すと，V53はリセットされます．

8.1　CPUのリセット動作

8.1.1　リセットによる初期値

CPUはリセットされると，表8－1のように初期化され，FFFF0H番地より命令のプリフェッチを開始します．

表8－1　CPUのリセット

対　象	初　期　値	備　考
PFP	0000H	スタート・アドレス：FFFF0H番地
PC	0000H	
PS	FFFFH	
SS	0000H	
DS0	0000H	
DS1	0000H	
PSW	V DIR IE BRK 上位 1 1 1 1 0 0 0 0 S Z AC P CY 下位 0 0 0 0 0 0 0 0	
キュー	クリア	
XAM	7 6 5 4 3 2 1 0 － － － － － － － 0	XAフラグ：通常アドレス・モード
PGR1－ PGR64	15 14 13 12 11 10 9 8 － － － － － － × × 7 6 5 4 3 2 1 0 × × × × × × × ×	ページ・レジスタ：不定

備考　×：リセット直前の状態を保持

8．1．2　リセット直後のタイミング

リセット直後の動作は，次のような２つの動作があります．

(1)　通常のプリフェッチ（HLDRQ入力なし）

図８－１のように，RESOUTがインアクティブになってから５クロックのちに，最初のプリフェッチを開始します．

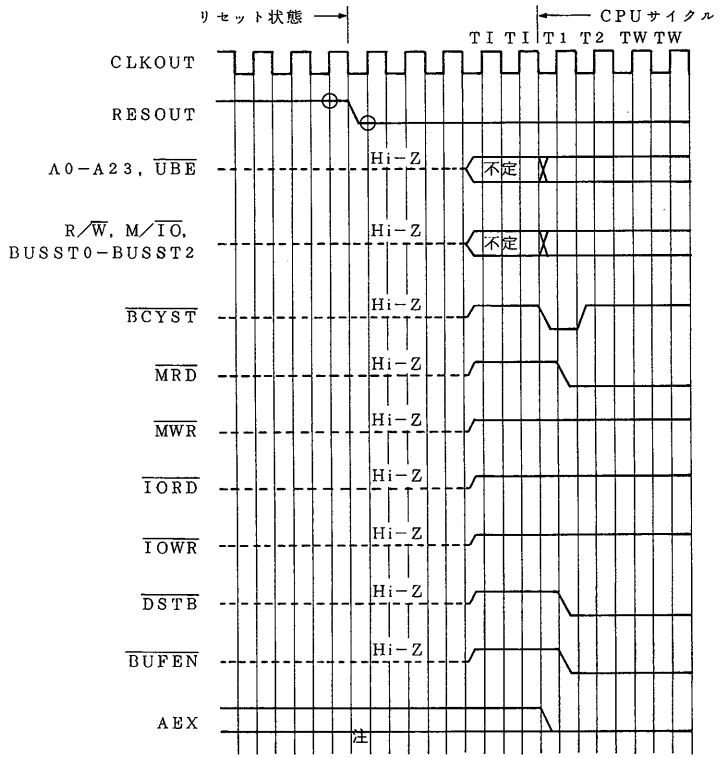
なお，NMIによる外部要因は，このタイミングに影響しません．

(2)　バス・ホールド時（HLDRQ入力あり）

HLDRQがアクティブのままリセットを解除した場合，図８－２のように，ただちにHLDAKを返しホールド状態になります．

ただし，HLDRQはリセット解除1.5クロック以前にアクティブにする必要があります．

図８－１　リセット直後のタイミング（通常のプリフェッチ）

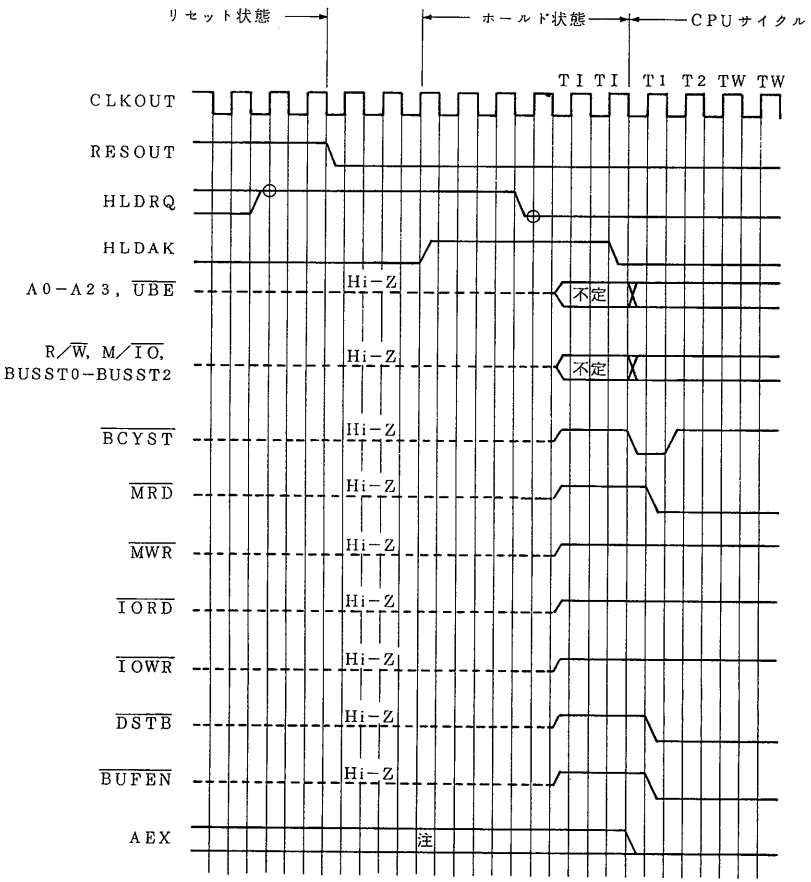


注　パワーオン・リセット時は不定，通常リセット時は直前の値を保持します．

備考１．　バス・ホールド，リセットによってハイ・インピーダンス状態になる端子はラッチを内蔵しているため，外部からドライブされないかぎり直前のレベルを保持しています．

２．　○印はサンプリングされるタイミングです．

図8－2　リセット直後のタイミング（バス・ホールド時）



注　パワーオン・リセット時は不定，通常リセット時は直前の値を保持します．

備考1．バス・ホールド，リセットによってハイ・インピーダンス状態になる端子はラッチを内蔵しているため，外部からドライブされないかぎり直前のレベルを保持しています．

2．○印はサンプリングされるタイミングです．

8．1．3　リセット時の端子状態

端子名	入出力	端子状態	端子名	入出力	端子状態
A0-A23	3ステート出力	Hi－Z	RESOUT	出力	H
D0-D15	3ステート入出力	Hi－Z	X1, X2	入力	－
$\overline{UBE}$	3ステート出力	Hi－Z	CLKOUT	出力	非固定
R/$\overline{W}$	3ステート出力	Hi－Z	PCLKOUT	出力	非固定
M/$\overline{IO}$	3ステート出力	Hi－Z	TCLK	入力	－
BUSST0-BUSST2	3ステート出力	Hi－Z	TCTL0-TCTL2	入力	－
$\overline{BCYST}$	3ステート出力	Hi－Z	TOUT0-TOUT2	出力	非固定
$\overline{DSTB}$	3ステート出力	Hi－Z	INTP0-INTP7	入力	－
$\overline{MRD}$	3ステート出力	Hi－Z	$\overline{INTAK}$	出力	H
$\overline{MWR}$	3ステート出力	Hi－Z	TxD	出力	H
$\overline{IORD}$	3ステート出力	Hi－Z	RxD	入力	－
$\overline{IOWR}$	3ステート出力	Hi－Z	RxRDY	出力	H
$\overline{BUFEN}$	3ステート出力	Hi－Z	SINT	出力	L
$\overline{BUSLOCK}$	出力	H	$\overline{RTS}$	出力	H
$\overline{READY}$	入力	－	$\overline{CTS}$	入力	－
BS8/BS16	入力	－	$\overline{DTR}$	出力	H
AEX	出力	H／L	$\overline{DSR}$	入力	－
$\overline{REFRQ}$	出力	H	$\overline{DMARQ0}$-$\overline{DMARQ3}$	入力	－
HLDRQ	入力	－	$\overline{DMAAK0}$-$\overline{DMAAK3}$	出力	H
HLDAK	出力	L	$\overline{END/TC}$	入出力	Hi－Z
$\overline{NMI}$	入力	－	V_{DD}	－	－
$\overline{CPBUSY}$	入力	－	GND	－	－
$\overline{RESET}$	入力	－	IC1, IC2	－	－

8.2 内蔵I／Oのリセット動作

内蔵I／Oもリセットで初期化されるものがあります．表8－2に初期化されるI／Oの一覧表を示します．この一覧表に載っていないI／Oについてはリセット直前の状態が保持されますが，パワーオン時は不定となります．

表8－2 内蔵I／Oのリセット（1／3）

(a) システムI／O領域（1／2）

対象	初期値	備考
SCTL	7 6 5 4 3 2 1 0 － － － 0 0 0 0 0	・16ビット・バウンダリ ・μPD71071モード選択 ・μPD71037モード時にA16へキャリー伝搬をしない ・μPD71037モード時にA20へキャリー伝搬をしない ・SCUの入力クロック：TOUT1
OPSEL	7 6 5 4 3 2 1 0 － － － － 0 0 0 0	DMAU，ICU，TCU，SCUは使用不可
WCY0	7 6 5 4 3 2 1 0 － － － － － 1 1 1	16Mバイト中の上位メモリ・ブロック・アクセス時のウエイト挿入：7ウエイト
WCY1	7 6 5 4 3 2 1 0 － 1 1 1 － 1 1 1	16Mバイト中の中位，下位メモリ・ブロック・アクセス時のウエイト挿入：7ウエイト
WCY2	7 6 5 4 3 2 1 0 － 1 1 1 － 1 1 1	1Mバイト中の中位，下位メモリ・ブロック・アクセス時のウエイト挿入：7ウエイト
WCY3	7 6 5 4 3 2 1 0 － 1 1 1 － 1 1 1	・外部I／Oアクセス時，割り込みアクノリッジ・サイクル時のウエイト挿入：7ウエイト ・1Mバイト中の上位メモリ・ブロック・アクセス時のウエイト挿入：7ウエイト
WCY4	7 6 5 4 3 2 1 0 － 1 1 1 － 1 1 1	・DMAサイクル時のウエイト挿入：7ウエイト ・リフレッシュ・サイクル時のウエイト挿入：7ウエイト
WMB0	7 6 5 4 3 2 1 0 － 1 1 1 － 1 1 1	16Mバイト・メモリ空間の上位，下位メモリ・ブロック・サイズ：8Mバイト
WMB1	7 6 5 4 3 2 1 0 － 1 1 1 － 1 1 1	1Mバイト・メモリ空間の上位，下位メモリ・ブロック・サイズ：512Kバイト
WAC	7 6 5 4 3 2 1 0 － － － － 0 0 0 0	1Mバイト・メモリ領域の16Mバイト空間内でのアドレスの上位4ビット：0000
TCKS	7 6 5 4 3 2 1 0 － － － 0 0 0 0 0	TCUのTCT#0－TCT#2のクロックは発振周波数を4分周した内部クロック

表８－２　内蔵Ｉ／Ｏのリセット（２／３）

(a)　システムＩ／Ｏ領域（２／２）

対　象	初　期　値	備　考
RFC	7 6 5 4 3 2 1 0 × 0 — 0 1 0 0 0	・リフレッシュ許可／禁止：不定 ・タイマ・ファクタ（Ｎ）＝9 ・リフレッシュ・アドレスを２ずつインクリメント ・$\overline{\text{UBE}}$：ロウ・レベル出力
SBCR	7 6 5 4 3 2 1 0 — — — 0 0 0 0 0	・ＨＡＬＴ命令の実行によりＨＡＬＴモードに入る ・発振安定時間＝2^{21}／発振周波数 ・内部クロック周波数（ｆｘ）＝発振周波数×１／２
BRC	7 6 5 4 3 2 1 0 0 0 0 0 0 0 0 0	ＢＲＣ設定値＝２

(b)　SCU

対　象	初　期　値	備　考
SMD	7 6 5 4 3 2 1 0 0 1 0 0 1 0 1 1	・ボー・レート：×６４ ・キャラクタ長：７ビット ・パリティ：なし ・ストップ・ビット数：１ビット
SCM	7 6 5 4 3 2 1 0 — — 0 0 0 0 0 0	・送信禁止 ・受信禁止
SIMK	7 6 5 4 3 2 1 0 — — — — — — 1 1	・ＲＢＲＤＹ割り込み：マスク ・ＴＢＲＤＹ割り込み：マスク
SST	7 6 5 4 3 2 1 0 × 0 0 0 0 1 0 0	・送信データ・バッファ状態：非Ｒｅａｄｙ ・受信データ・バッファ状態：非Ｒｅａｄｙ ・エラー：なし ・ブレーク検出：なし

備考　×：リセット直前の状態を保持

表8－2　内蔵I／Oのリセット（3／3）

(c)　DMAU

対　象	初　期　値	備　考
DCH	7 6 5 4 3 2 1 0 － － － 0 0 0 0 1	・DMAチャネル0選択 ・アクセス条件：カレントのみ（リード時） ベースとカレント（ライト時）
DMD	7 6 5 4 3 2 1 0 0 0 0 0 0 0 － 0	・ディマンド・モード，アドレス増 ・オートイニシャライズ禁止 ・ベリファイ転送，バイト転送
DDC	［上位］ 7 6 5 4 3 2 1 0 － － － － － － 0 0	・バス・リリース・モード ・ベリファイ時のウエイト・ステート禁止
	［下位］ 7 6 5 4 3 2 1 0 － － 0 0 － 0 － －	・DMA動作：許可 ・チャネル優先順位：固定（CH0＞…＞CH3） ・書き込みタイミング：通常ライト・タイミング
DST	7 6 5 4 3 2 1 0 0 0 0 0 0 0 0 0	・転送終端：未終端 ・DMAリクエスト状況：リクエストなし
DMK	7 6 5 4 3 2 1 0 － － － － 1 1 1 1	全チャネルDMA要求マスク

[メ　モ]

第９章　アドレス生成

９．１　命令アドレス

命令が実行されるごとに自動的にインクリメントされる以外に，命令の実行順を制御するいくつかの方法があります．それらを以下に示します．

９．１．１　ダイレクト・アドレシング

命令バイト中の２バイトまたは４バイト・イミーディエト・データが直接ＰＣまたはＰＳとＰＣの両方にロードされ，ブランチ・アドレスとなります．

次の命令を実行する際に用いられます．

```
CALL  far-proc
CALL  memptr16
CALL  memptr32
BR    far-label
BR    memptr16
BR    memptr32
```

９．１．２　レラティブ・アドレシング

命令バイト中の１バイトまたは２バイト・イミーディエト・データが符号付きディスプレースメント値となってＰＣに加算され，ブランチ・アドレスとなります．

８ビット・ディスプレースメントのときは，符号拡張されて１６ビット・データとなってＰＣに加算されます．

加算されるときのＰＣの内容は，次の命令の先頭アドレスを示しています．

次の命令を実行する際に用いられます．

```
CALL  near-proc
BR    near-label
BR    short-label
条件付きブランチ命令　short-label
```

９．１．３　レジスタ・アドレシング

命令バイト中のレジスタ指定フィールド（３ビット）で指定される任意の１６ビット・レジスタの内容が，ブランチ・アドレスとしてＰＣにロードされます．

データの場合と異なり，８個のすべての１６ビット・レジスタ（AW, BW, CW, DW, IX, IY,

SP，BP）を使用することができます．

次の命令を実行する際に用いられます．

記述例

```
CALL  regptr16        CALL  AW
BR    regptr16        BR    BW
```

9．1．4　レジスタ・インダイレクト・アドレシング

命令バイト中のレジスタ指定フィールドで指定される１６ビット・レジスタ（IX，IY，BW）でアドレスされるメモリの内容（ワードまたはダブルワード）がブランチ・アドレスとしてPC（またはPCとPSの両方）にロードされます．

記　述　例

```
CALL  memptr16      CALL  WORD  PTR 〔IX〕
CALL  memptr32      CALL  DWORD PTR 〔IY〕
BR    memptr16      BR    WORD  PTR 〔BW〕
BR    memptr32      BR    DWORD PTR 〔IX〕
```

備考　WORD PTRと指定されたものはmemptr16の命令コードが，DWORD PTRと指定されたものはmemptr32の命令コードが，アセンブラによって生成されます．

9．1．5　インデクスト・アドレシング

命令バイト中の１バイトまたは２バイト・イミーディエト・データが符号付きディスプレースメント値となって，インデクス・レジスタとして働く１６ビット・レジスタ（IXまたはIY）に加算され，その結果がアドレスするメモリの内容（ワードまたはダブル・ワード）がブランチ・アドレスとしてPCにロードされます．

次の命令を実行する際に用いられます．

記　述　例

```
CALL  memptr16      CALL  var〔IX〕〔2〕
CALL  memptr32      CALL  var〔IY〕
BR    memptr16      BR    var〔IY〕
BR    memptr32      BR    var〔IX+4〕
```

備考　変数varがワード属性を持つ場合はmemptr16の命令コードが，ダブル・ワード属性を持つ場合はmemptr32の命令コードがアセンブラによって生成されます．

9．1．6　ベースト・アドレシング

命令バイト中の1バイトまたは2バイト・イミーディエト・データが符号付きディスプレースメント値となって，ベース・レジスタとして働く16ビット・レジスタ（BPまたはBW)に加算され，その結果がアドレスするメモリの内容（ワードまたはダブル・ワード）がブランチ・アドレスとしてPCにロードされます.

次の命令を実行する際に用いられます.

記　述　例

```
CALL  memptr16     CALL  var〔BP+2〕
CALL  memptr32     CALL  var〔BP〕
BR    memptr16     BR    var〔BW〕〔2〕
BR    memptr32     BR    var〔BP〕
```

備考　変数varがワード属性を持つ場合はmemptr16の命令コードが，ダブル・ワードの属性を持つ場合はmemptr32の命令コードがアセンブラによって生成されます.

9．1．7　ベースト・インデクスト・アドレシング

命令バイト中の1バイトまたは2バイト・イミーディエト・データが符号付きディスプレースメント値となり，この値とベース・レジスタとして働く16ビット・レジスタ（BPまたはBW)およびインデクス・レジスタとして働く16ビット・レジスタ（IXまたはIY）の3つが加算され，その結果がアドレスするメモリの内容（ワードまたはダブル・ワード）が，ブランチ・アドレスとしてPCにロードされます.

次の命令を実行する際に用いられます.

```
CALL  memptr16
CALL  memptr32
BR    memptr16
BR    memptr32
```

記述例

```
CALL  var〔BP〕〔IX〕
CALL  var〔BW+2〕〔IY〕
BR    var〔BW〕〔2〕〔IX〕
BR    var〔BP+4〕〔IY〕
```

備考　変数varがワードの属性を持つ場合はmemptr16の命令コードが，ダブル・ワードの属性を持つ場合はmemptr32の命令コードがアセンブラによって生成されます.

９．２　メモリ・オペランド・アドレス

命令を実行する際に操作対象となるレジスタやメモリなどをアドレスする方法として，次に示すいくつかの方法があります．

９．２．１　レジスタ・アドレシング

命令バイト中のレジスタ指定フィールド（reg＝3ビット・フィールド，sreg＝2ビット・フィールド）の内容が操作対象となるレジスタをアドレスします．

regの場合は，同じく命令バイト中のワードかバイトかを指定する１ビット（W）と組みになって８種のワード・レジスタ（AW，BW，CW，DW，BP，SP，IX，IY）と８種のバイト・レジスタ（AL，AH，BL，BH，CL，CH，DL，DH）を指定します．

sregの場合は，４種のセグメント・レジスタ（PS，SS，DS0，DS1）を指定します．

また，命令のオペレーション・コードが特定のレジスタを指定する場合もあります．

次に示すオペランド記述形式を持つ命令を実行する際に用いられます．

記述形式	記述方法
reg	AW，BW，CW，DW，SP，BP，IX，IY， AL，AH，BL，BH，CL，CH，DL，DH
reg16	AW，BW，CW，DW，SP，BP，IX，IY
reg8	AL，AH，BL，BH，CL，CH，DL，DH
sreg	PS，SS，DS0，DS1
acc	AW，AL

記述例

```
MOV     reg，reg'の場合
MOV     BP，SP
MOV     AL，CL
```

９．２．２　イミーディエト・アドレシング

命令バイト中の１バイトまたは２バイト・イミーディエト・データがそのまま操作対象となります．

次に示すオペランド記述形式を持つ命令を実行する際に用いられます．

記述形式	記述方法
imm	8／16ビット・イミーディエト・データ
imm16	16　ビット　〃
imm8	8　ビット　〃
pop-value	16　ビット　〃

immの場合，８ビットか１６ビットかの区別は，オペランドに記述されたimm の値または同時に記述される別のオペランドの属性をアセンブラが判断して決定し，ワード／バイト指定ビットWを決定します．

記述例

MOV reg,immの場合

```
MOV AL,5 ;バイト

MUL reg16,reg16',imm16の場合
MUL AW,BW,1000H
```

９．２．３ ダイレクト・アドレシング

命令バイト中のイミーディエト・データが，操作対象となるメモリをアドレスします．
次のオペランド記述形式を持つ命令を実行する際に用いられます．

記述形式	記述方法
mem	８ビットまたは１６ビット・メモリ・データを指定する１６ビット変数
dmem	〃
imm4	ビット・フィールド・データのビット長を示す４ビット変数

記述例

MOV mem,immの場合

```
MOV WORD_VAR,2000H

MOV acc,dmemの場合
MOV AL,BYTE_VAR
```

９．２．４ レジスタ・インダイレクト・アドレシング

命令バイト中のメモリ指定フィールド(mod,mem)によって指定される１６ビット・レジスタ(IX,IY,BW)が，操作対象となるメモリをアドレスします．
次に示すオペランド記述形式を持つ命令を実行する際に用いられます．

記述形式	記述方法
mem	〔IX〕,〔IY〕,〔BW〕

記述例

SUB mem,regの場合

```
SUB 〔IX〕,AW
```

9．2．5　オートインクリメント／デクリメント・アドレシング

レジスタ・インダイレクト・アドレシングに属するものですが，ディフォールト・レジスタの内容で操作対象をアドレスしたあと，そのレジスタの内容を自動的にインクリメント／デクリメント（バイト処理なら＋1／－1，ワード処理なら＋2／－2）します．

つまり，このアドレシング機能を用いれば，次のバイト／ワード・オペランド処理のためのアドレス更新が自動的に行われるわけです．

インクリメントとデクリメントの区別は，方向フラグ（DIR）によって行われ，DIR=0 ならインクリメント，1ならデクリメントです．

このアドレシングは，すべてディフォールト・レジスタに対して行われ，次に示すオペランド記述形式を持つ命令を実行する際に用いられます．

記述形式	ディフォールト・レジスタ
dst-block	IY
src-block	IX

このアドレシングとバイト／ワード・オペランドの繰り返し処理回数をカウントするカウンタ（CW）が組み合わされて，ブロック・データ処理の制御に用いられます．

9．2．6　インデクスト・アドレシング

命令バイト中の1バイトまたは2バイト・イミーディエト・データが符号付きディスプレースメント値となって，インデクス・レジスタとして働く16ビット・レジスタ（IXまたはIY）に加算され，その結果が操作対象となるメモリ・オペランドをアドレスします．

このアドレシングは，アレイ・タイプのデータをアクセスするのに有効で，ディスプレースメントがアレイの開始アドレスを指し，インデクス・レジスタの内容がそこから何番目のアレイかを決定します．

次に示すオペランド記述形式を持つ命令を実行する際に用いられます．

記述形式	記述方法
mem	var〔IX〕,var〔IY〕
mem16	〃
mem8	〃

記述例

```
TEST    mem,immの場合
TEST    BYTE_VAR〔IX〕,7FH
TEST    BYTE_VAR〔IX+8〕,7FH
TEST    WORD_VAR〔IX〕〔8〕,7FFFH
```

備考　変数varがバイト属性を持つ場合はバイト・オペランドが指定され，ワード属性を持つ場合はワード・オペランドが指定され，それぞれに該当する命令コードがアセンブラによって生成されます．

9．2．7　ベースト・アドレシング

命令バイト中の1バイトまたは2バイト・イミーディエト・データが符号付きディスプレースメント値となって，ベース・レジスタとして働く16ビット・レジスタ（BPまたはBW）に加算され，その結果が操作対象となるメモリ・オペランドをアドレスします．

このアドレシングは，メモリの複数箇所に置かれる構造タイプのデータをアクセスするのに有効で，ベース・レジスタが各構造の開始アドレスを指し，ディスプレースメントが各構造内の1要素を選択します．

次に示すオペランド記述形式を持つ命令を実行する際に用いられます．

記述形式	記述方法
mem	var〔BP〕，var〔BW〕
mem16	〃
mem8	〃

記述例

```
SHL     mem，1の場合
SHL     BYTE_VAR〔BP〕，1
SHL     WORD_VAR〔BP+2〕，1
SHL     BYTE_VAR〔BP〕〔4〕，1
```

備考　変数varがバイト属性を持つ場合はバイト・オペランドが指定され，ワード属性を持つ場合はワード・オペランドが指定され，それぞれに該当する命令コードがアセンブラによって生成されます．

9．2．8　ベースト・インデクスト・アドレシング

命令バイト中の1バイトまたは2バイト・イミーディエト・データが符号付きディスプレースメント値となり，この値とベース・レジスタとして働く16ビット・レジスタ（BPまたはBW）およびインデクス・レジスタとして働く16ビット・レジスタ（IXまたはIY）が加算され，その結果が操作対象となるメモリ・オペランドをアドレスします．

このアドレシングは，ベース・レジスタの内容とインデクス・レジスタの内容を両方変化させて1つのデータを指すことができるため，アレイ・タイプを含んだ構造タイプのデータをアクセスするのに非常に有効です．すなわち，ベース・レジスタで各構造の先頭アドレスを指し，ディスプレースメント値がそこからアレイ・データの先頭アドレスまでのオフセット分を示し，アレイ・データの何番目かをインデクス・レジスタが指すというようなことができます．

次に示すオペランド記述形式を持つ命令を実行する際に用いられます.

記述形式	記述方法
mem	var〔ベース・レジスタ〕〔インデクス・レジスタ〕
mem16	〃
mem8	〃

記述例

```
PUSH    mem16の場合
PUSH    WORD_VAR[BP][IX]
PUSH    WORD_VAR[BP+2][IX+6]
PUSH    WORD_VAR[BP][4][IX][8]
```

9．2．9　ビット・アドレシング

命令バイト中の3／4ビット・イミーディエト・データまたはCLレジスタの下位3／4ビットが操作対象の8／16ビット・レジスタまたはメモリの1つのビットを指定します.

このアドレシングを用いた命令によれば，レジスタ／メモリの特定の1ビットだけを，ほかのビット内容を意識せずにテスト（0，1の判定）／セット／クリア／反転することができます．つまり，AND命令やOR命令を使用してセット，リセットするときのように，1ビットを操作するのにバイト／ワード・データを用意しなければならないということは起こりません.

次に示すオペランド記述形式を持つ命令を実行する際に用いられます.

記述形式	記述方法
imm4	ワード・オペランドのビット番号
imm3	バイト・オペランドのビット番号
CL	CL

記述例

```
TEST1   reg8, CL
TEST1   AL,CL
NOT1    reg8,imm3
NOT1    CL,5
CLR1    mem16,CL
CLR1    WORD_VAR[IX],CL
SET1    mem16,imm4
SET1    WORD_VAR[BP],9
```

第10章 命令の説明

この章ではV53の命令の説明をします．各命令は次の項目に分けて説明します．

① 記述形式
② 命令語形式
③ バイト数
④ クロック数
⑤ 16ビット・ワード転送回数
⑥ 機能
⑦ フラグの動作
⑧ 記述例

①記述形式，⑥機能，⑦フラグの動作の項目では説明上いくつもの識別子を用いています．表10－1から表10－4に識別子とその意味を，表10－5から表10－7にメモリ・アドレシング，汎用レジスタ／セグメント・レジスタの選択について示します．

⑧記述例の項目はリロケータブル・アセンブラRA70136の記述方法により示しています．

④クロック数は実行ユニットが命令実行に必要とする時間です．この命令実行クロック数にはプリフェッチ時間，プリデコード時間，バス使用のための待ち時間などは含んでいません．条件は次のようになります．

(1) メモリ・アクセスは，0ウエイトを想定しています．つまり1バス・サイクルは2クロックです．
(2) I／Oアクセスは0ウエイトを想定しています．
(3) 奇数アドレスへのワード・データの場合，バス・サイクルを2回起動します．
奇数アドレス，偶数アドレスでそれぞれ場合分けしています．
(4) プリミティブ・ブロック転送命令，プリミティブ入出力命令はリピート・プリフィクスも含んでいます．
(5) 通常アドレス・モードを想定しています．

備考1． バス・サイジング機能は，16ビット指定を想定しています．8ビット指定を行った場合，クロック数は偶数番地へのワード・データに対してのみバス・サイクルを2倍にする必要があります．
2． ウエイト挿入時のクロック数は，（16ビット・ワード転送回数）×（ウエイト数）を加える必要があります．

表10－1　オペランド・タイプの凡例

識　別　子	説　　明
reg, reg′	8／16ビット汎用レジスタ
reg8, reg8′	8ビット汎用レジスタ
reg16, reg16′	16ビット汎用レジスタ
dmem	8／16ビット・メモリ・ロケーション
mem	8／16ビット・メモリ・ロケーション
mem8	8ビット・メモリ・ロケーション
mem16	16ビット・メモリ・ロケーション
mem32	32ビット・メモリ・ロケーション
imm	0－FFFFHの範囲の定数
imm3	0－7Hの範囲の定数
imm4	0－FHの範囲の定数
imm8	0－FFHの範囲の定数
imm16	0－FFFFHの範囲の定数
acc	レジスタAWまたはAL
sreg	セグメント・レジスタ
src-table	256バイト変換テーブルの名称
src-block	レジスタIXでアドレスされるブロックの名称
dst-block	レジスタIYでアドレスされるブロックの名称
near-proc	現在のプログラム・セグメント内のプロシージャ
far-proc	別のプログラム・セグメント内のプロシージャ
near-label	現在のプログラム・セグメント内のレーベル
short-label	命令の終わりから－128～＋127バイトの範囲のレーベル
far-label	別のプログラム・セグメント内のレーベル
memptr16	制御が移されようとしている現在のプログラム・セグメント内のロケーションのオフセットを含むワード
memptr32	制御が移されようとしている別のプログラム・セグメント内のロケーションのオフセットとセグメント・ベース・アドレスを含むダブルワード
regptr16	制御が移されようとしている別のプログラム・セグメント内のロケーションのオフセットを含む16ビット汎用レジスタ
pop-value	スタックから捨てるバイト数（0－64K，通常は偶数）
fp-op	外部の浮動小数点演算用コプロセッサの命令コードを判別するイミーディエト値

表10－1　オペランド・タイプの凡例（続き）

識　別　子	説　　明
R	レジスタ・セット
DS1-spec	DS1，またはDS1にASSUMEしてあるセグメント名／グループ名
Seg-spec	任意のセグメント・レジスタ名，またはセグメント・レジスタにASSUMEしてあるセグメント名／グループ名
〔　　〕	省略可能

表10－2　命令語の凡例

識　別　子	説　　明
W	バイト／ワード指定ビット（0：バイト，1：ワード）．ただし，s＝1のときは，W＝1であってもサイン拡張のバイト・データを16ビット・オペランドとします．
reg，reg′	レジスタ・フィールド（000-111）
mem	メモリ・フィールド（000-111）
mod	モード・フィールド（00-10）
(disp-low)	オプション16ビット・ディスプレースメント下位バイト
(disp-high)	〃　上位バイト
disp-low	PCレラティブ加算用16ビット・ディスプレースメント下位バイト
disp-high	〃　上位バイト
imm3	3ビット・イミーディエト・データ
imm4	4ビット　〃
imm8	8ビット　〃
imm16-low	16ビット・イミーディエト・データの下位バイト
imm16-high	〃　上位バイト
addr-low	16ビット・ダイレクト・アドレスの下位バイト
addr-high	〃　上位バイト
sreg	セグメント・レジスタ指定ビット（00-11）
s	サイン拡張指定ビット（1：サイン拡張あり，0：サイン拡張なし）
offset-low	PCにロードされる16ビット・オフセット・データの下位バイト
offset-high	〃　上位バイト
seg-low	PSにロードされる16ビット・セグメント・データの下位バイト
seg-high	〃　上位バイト
pop-value-low	スタックの捨てるバイト数を指定する16ビット・データの下位バイト
pop-value-high	〃　上位バイト
disp8	PCにレラティブ加算される8ビット・ディスプレースメント

表10－2　命令語の凡例（続き）

識　別　子	説　　明
X XXX YYY ZZZ	外部浮動小数点演算用コプロセッサの命令コードを判別するためのデータ

表10－3　オペレーション説明上の凡例

識　別　子	説　　明
AW	アキュームレータ（16ビット）
AH	〃　（上位バイト）
AL	〃　（下位バイト）
BW	レジスタBW（16ビット）
CW	レジスタCW（　〃　）
CL	〃　（下位バイト）
DW	レジスタDW（16ビット）
BP	ベース・ポインタ（16ビット）
SP	スタック・ポインタ（16ビット）
PC	プログラム・カウンタ（16ビット）
PSW	プログラム・ステータス・ワード（16ビット）
IX	インデクス・レジスタ（ソース）（16ビット）
IY	〃　（デスティネーション）（16ビット）
PS	プログラム・セグメント・レジスタ（16ビット）
DS1	データ・セグメント1・レジスタ（16ビット）
DS0	データ・セグメント0・レジスタ（16ビット）
SS	スタック・セグメント・レジスタ（16ビット）
AC	補助キャリー・フラグ
CY	キャリー・フラグ
P	パリティ・フラグ
S	サイン・フラグ
Z	ゼロ・フラグ
DIR	方向フラグ
IE	割り込み許可フラグ
V	オーバフロー・フラグ
BRK	ブレーク・フラグ

表10－3　オペレーション説明上の凡例（続き）

識　別　子	説　　明
(…)	（ ）内で示されるメモリの内容
disp	ディスプレースメント（8／16ビット）
temp	テンポラリ・レジスタ（8／16／32ビット）
temp1, 2	テンポラリ・レジスタ（16ビット）
ext-disp8	8ビット・ディスプレースメントをサイン拡張した16ビット
TA	テンポラリ・レジスタA（16ビット）
TB	テンポラリ・レジスタB（16ビット）
TC	テンポラリ・レジスタC（16ビット）
tmpcy	テンポラリ・キャリー・フラグ（1ビット）
seg	イミーディエト・セグメント・データ（16ビット）
offset	イミーディエト・オフセット・データ（16ビット）
←	転送方向
＋	加　算
－	減　算
×	乗　算
÷	除　算
%	モジュロ
∧	論理積
∨	論理和
⊻	排他的論理和
××H	16進数2けたの数値
××××H	16進数4けたの数値

表10－4　フラグの動作の凡例

識　別　子	説　　明
（ブランク）	変化なし
0	0にクリアされる
1	1にセットされる
×	結果に従ってセットまたはクリアされる
U	不　定
R	以前に退避した値がリストアされる

表10－5　メモリ・アドレシング

mem ＼ mod	0 0	0 1	1 0
0 0 0	BW+IX	BW+IX+disp8	BW+IX+disp16
0 0 1	BW+IY	BW+IY+disp8	BW+IY+disp16
0 1 0	BP+IX	BP+IX+disp8	BP+IX+disp16
0 1 1	BP+IY	BP+IY+disp8	BP+IY+disp16
1 0 0	IX	IX+disp8	IX+disp16
1 0 1	IY	IY+disp8	IY+disp16
1 1 0	ダイレクト・アドレス	BP+disp8	BP+disp16
1 1 1	BW	BW+disp8	BW+disp16

表10－6　8／16ビット汎用レジスタの選択

reg, reg'	W＝0	W＝1
0 0 0	AL	AW
0 0 1	CL	CW
0 1 0	DL	DW
0 1 1	BL	BW
1 0 0	AH	SP
1 0 1	CH	BP
1 1 0	DH	IX
1 1 1	BH	IY

表10－7　セグメント・レジスタの選択

sreg	
0 0	DS1
0 1	PS
1 0	SS
1 1	DS0

10．1　データ転送命令

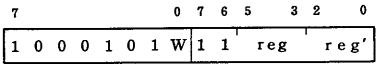
10．1．1　MOV(Move)

(1) レジスタからレジスタへ

① 記述形式

MOV reg,reg′

② 命令語形式



③ バイト数

2

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

reg←reg′

第1オペランドで指定される8／16ビット・レジスタに，第2オペランドで指定される8／16ビット・レジスタの内容を転送します．

⑦ フラグの動作

なし

⑧ 記述例

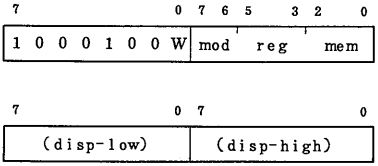
MOV BP,SP

(2) レジスタからメモリへ

① 記述形式

MOV mem,reg

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき 3

W＝1のとき 5：奇数アドレス

3：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

(mem)←reg

第1オペランドで指定される8／16ビット・メモリに，第2オペランドで指定される8／16ビット・レジスタの内容を転送します．

⑦ フラグの動作

なし

⑧ 記述例

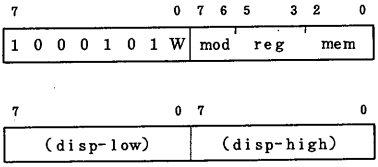
MOV 〔BP〕〔IX〕,AW

(3) メモリからレジスタへ

① 記述形式

MOV reg,mem

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき 5

W＝1のとき 7：奇数アドレス

5：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

reg←(mem)

第1オペランドで指定される8／16ビット・レジスタに，第2オペランドで指定される8／16ビット・メモリの内容を転送します．

⑦ フラグの動作

なし

⑧ 記述例

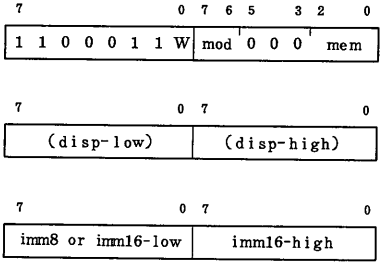
MOV AW,〔BW〕〔IY〕

(4) イミーディエト・データをメモリへ

① 記述形式

MOV mem, imm

② 命令語形式



③ バイト数

3／4／5／6

④ クロック数

W＝0のとき 3

W＝1のとき 5：奇数アドレス

3：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

(mem)← imm

第1オペランドでアドレスされる8／16ビット・メモリに，第2オペランドで指定される8／16イミーディエト・データを転送します．

⑦ フラグの動作

なし

⑧ 記述例

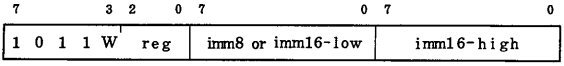
MOV 〔BP〕〔IX〕,0000H

(5) イミーディエト・データをレジスタへ

① 記述形式

MOV reg,imm

② 命令語形式



③ バイト数

2／3

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg←imm

第1オペランドで指定される8／16ビット・レジスタに，第2オペランドで指定される8／16イミーディエト・データを転送します.

⑦ フラグの動作

なし

⑧ 記述例

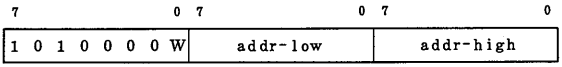
MOV BP,8000H

(6) メモリからアキュームレータへ

① 記述形式

MOV acc,dmem

② 命令語形式



③ バイト数

3

④ クロック数

W＝0のとき　5

W＝1のとき　7：奇数アドレス

5：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機　能

W＝0のとき　AL←(dmem)

W＝1のとき　AW←(dmem+1,dmem)

第1オペランドで指定されるアキュームレータ(AL／AW)に，第2オペランドでアドレスされるメモリの内容を転送します．

⑦ フラグの動作

なし

⑧ 記述例

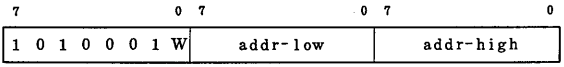
MOV AL,〔2000H〕

(7) アキュームレータからメモリへ

① 記述形式

MOV dmem,acc

② 命令語形式



③ バイト数

3

④ クロック数

W＝0のとき　3

W＝1のとき　5：奇数アドレス

3：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機　能

W＝0のとき　(dmem)←AL

W＝1のとき　(dmem+1,dmem)←AW

第1オペランドでアドレスされる8／16ビット・メモリに，第2オペランドで指定されるアキュームレータ(AL／AW)の内容を転送します．

⑦ フラグの動作

なし

⑧ 記述例

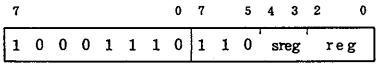
MOV 〔3000H〕, AW

(8) レジスタからセグメント・レジスタへ

① 記述形式

MOV sreg,reg16

② 命令語形式



③ バイト数

2

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

sreg←reg16

第1オペランドで指定されるセグメント・レジスタ（PSを除く）に，第2オペランドで指定される16ビット・レジスタの内容を転送します．

なお，この命令と次の命令との間では，ハードウエア割り込み（マスカブル割り込み，ノンマスカブル割り込み）要求およびシングル・ステップ・ブレークは受け付けられません．

⑦ フラグの動作

なし

⑧ 記述例

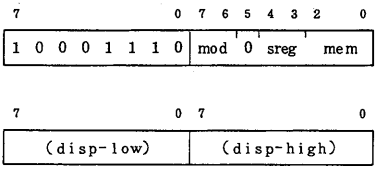
MOV SS,AW

(9) メモリからセグメント・レジスタへ

① 記述形式

MOV sreg,mem16

② 命令語形式



③ バイト数

2／3／4

④ クロック数

7：奇数アドレス

5：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

sreg←(mem16)

第1オペランドで指定されるセグメント・レジスタ(PSを除く)に，第2オペランドでアドレスされる16ビット・メモリの内容を転送します．

⑦ フラグの動作

なし

⑧ 記述例

MOV DS0,SEG[BW][IX]

⑨ 注 意

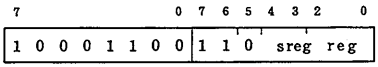
この命令と次の命令との間では，ハードウエア割り込み（マスカブル割り込み，ノンマスカブル割り込み）要求およびシングル・ステップ・ブレークは受け付けられません．

(10) セグメント・レジスタからレジスタへ

① 記述形式

MOV reg16,sreg

② 命令語形式



③ バイト数

2

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg16←sreg

第1オペランドで指定される16ビット・レジスタに，第2オペランドで指定されるセグメント・レジスタの内容を転送します．

⑦ フラグの動作

なし

⑧ 記述例

MOV AW,DS1

⑨ 注 意

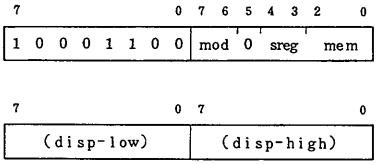
この命令と次の命令との間では，ハードウエア割り込み（マスカブル割り込み，ノンマスカブル割り込み）要求およびシングル・ステップ・ブレークは受け付けられません．

(11) セグメント・レジスタからメモリへ

① 記述形式

MOV mem16,sreg

② 命令語形式



③ バイト数

2／3／4

④ クロック数

5：奇数アドレス

3：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

(mem16)←sreg

第1オペランドでアドレスされる16ビット・メモリに，第2オペランドで指定されるセグメント・レジスタの内容を転送します．

⑦ フラグの動作

なし

⑧ 記述例

MOV 〔IX〕,PS

⑨ 注 意

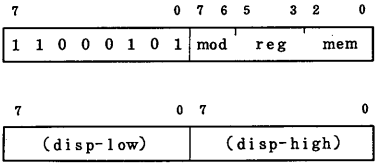
この命令と次の命令との間では，ハードウエア割り込み（マスカブル割り込み，ノンマスカブル割り込み）要求およびシングル・ステップ・ブレークは受け付けられません．

(12) 32ビット・メモリから16ビット・レジスタとDS0へ

① 記述形式

MOV DS0,reg16,mem32

② 命令語形式



③ バイト数

2／3／4

④ クロック数

14：奇数アドレス

10：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

reg16 ←(mem32)

DS0 ←(mem32+2)

第3オペランドでアドレスされる32ビット・メモリの下位16ビット(32ビット・ポインタ変数のオフセット・ワード)を，第2オペランドで指定される16ビット・レジスタに，上位16ビット(セグメント・ワード)をセグメント・レジスタDS0に転送します.

⑦ フラグの動作

なし

⑧ 記述例

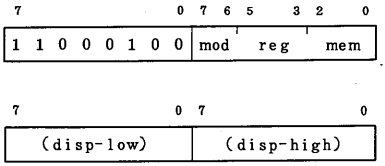
MOV DS0,BW,DWORD_VAR[IY]

(13) 32ビット・メモリから16ビット・レジスタとDS1へ

① 記述形式

MOV DS1,reg16,mem32

② 命令語形式



③ バイト数

2／3／4

④ クロック数

14：奇数アドレス

10：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

reg16 ←(mem32)

DS1 ←(mem32+2)

第3オペランドでアドレスされる32ビット・メモリの下位16ビット(32ビット・ポインタ変数のオフセット・ワード)を，第2オペランドで指定される16ビット・レジスタに，上位16ビット(セグメント・ワード)をセグメント・レジスタDS1に転送します.

⑦ フラグの動作

なし

⑧ 記述例

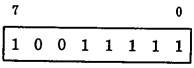
MOV DS1,BW,DWORD_VAR〔IX〕

(14) ＰＳＷからＡＨへ

① 記述形式

MOV AH, PSW

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ＡＨ←Ｓ，Ｚ，×，ＡＣ，×，Ｐ，×，ＣＹ

ＰＳＷのうち，Ｓ，Ｚ，ＡＣ，Ｐ，ＣＹの各フラグをＡＨレジスタに転送します．ＡＨのビット５，３，そして１は不定となります．

⑦ フラグの動作

なし

⑧ 記述例

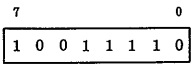
MOV AH, PSW

(15) ＡＨからＰＳＷへ

① 記述形式

MOV PSW, AH

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

Ｓ，Ｚ，×，ＡＣ，×，Ｐ，×，ＣＹ←ＡＨ

ＡＨレジスタのビット７，６，４，２，そして０を，それぞれＰＳＷのＳ，Ｚ，ＡＣ，Ｐ，ＣＹフラグに転送します．

⑦ フラグの動作

V	S	Z	AC	P	CY
	×	×	×	×	×

⑧ 記述例

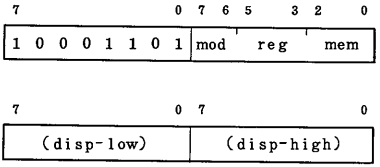
MOV PSW, AH

10．1．2 LDEA(Load Effective Address to register)

① 記述形式

LDEA reg16,mem16

② 命令語形式



③ バイト数

2／3／4

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg16←mem16

第1オペランドで指定される16ビット汎用レジスタに，第2オペランドで生成される実効アドレス（オフセット）をロードします．

TRANS命令やブロック命令でオペランドを指定するために自動的に用いられるレジスタ等にオペランド・アドレスの先頭値をセットするときなどに用いられます．

⑦ フラグの動作

なし

⑧ 記述例

LDEA AW,WORD_VAR〔BP〕

10．1．3 TRANS／TRANSB（Translate Byte）

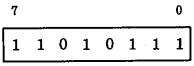
① 記述形式

TRANS src-table

TRANS （オペランドなし）

TRANSB （オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

5

⑤ 16ビット・ワード転送回数

1

⑥ 機　能

AL←（BW＋AL）

レジスタBWとレジスタALによってアドレスされる256バイトの変換テーブルの1バイトをレジスタALに転送します．このときレジスタBWはテーブルの先頭アドレスを指し，先頭アドレスから256バイト内のオフセット値をレジスタALが指定します．

⑦ フラグの動作

なし

⑧ 記述例

TRANS SIN_TBL

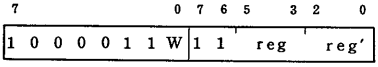
10．1．4　XCH（Exchange）

(1)　レジスタとレジスタ

①　記述形式

XCH　reg，reg′

②　命令語形式



③　バイト数

2

④　クロック数

3

⑤　１６ビット・ワード転送回数

なし

⑥　機　能

reg↔reg′

第１オペランドで指定される８／１６ビット・レジスタの内容と，第２オペランドで指定される８／１６ビット・レジスタの内容を交換します．

⑦　フラグの動作

なし

⑧　記述例

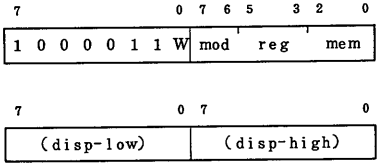
XCH　AW，BW

⑵ メモリとレジスタ

① 記述形式

XCH mem,reg or XCH reg,mem

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　8

W＝1のとき　12：奇数アドレス

8：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能

(mem)↔reg

第1オペランドでアドレスされる8／16ビット・メモリの内容と，第2オペランドで指定される8／16ビット・レジスタの内容を交換します．

⑦ フラグの動作

なし

⑧ 記述例

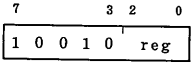
XCH WORD_VAR[BP],CW

(3) アキュームレータとレジスタ

① 記述形式

XCH AW, reg16 or XCH reg16, AW

② 命令語形式



③ バイト数

1

④ クロック数

3

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

AW↔reg16

第1オペランドで指定されるアキュームレータ（AWのみ）の内容と，第2オペランドで指定される16ビット・レジスタの内容を交換します．

⑦ フラグの動作

なし

⑧ 記述例

XCH AW, DW

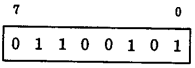
10.2　リピート・プリフィクス

10.2.1　REPC(Repeat while Carry)

① 記述形式

REPC（オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

CW≒0の間，続くバイトのプリミティブ・ブロック転送命令を実行し，CWをデクリメント（－1）します．

プリミティブ・ブロック転送命令の結果，CY≒1になるとループを抜けます．

CWのチェックは，プリミティブ・ブロック転送命令の実行前，すなわちREPC命令実行の直前の状態に対して行われます．したがって，最初CW=0でREPC命令実行に入ると，続くプリミティブ・ブロック転送命令は1回も実行されずにその次の命令に移ります．

CYのチェックは，あくまで続くプリミティブ・ブロック転送命令の結果に対して行われ，最初REPC命令に入る直前の内容は関係ありません．

⑦ フラグの動作

なし

⑧ 記述例

```
REPC  CMPBKW
```

⑨ 注　意

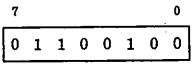
この命令と次の命令との間では，ハードウエア割り込み（マスカブル割り込み，ノンマスカブル割り込み）要求およびシングル・ステップ・ブレークは受け付けられません．

10．2．2 REPNC（Repeat while Not Carry）

① 記述形式

REPNC（オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ＣＷ≒０の間，続くバイトのプリミティブ・ブロック転送命令を実行し，ＣＷをデクリメント（－１）します．

プリミティブ・ブロック転送命令の結果，ＣＹ＝１になるとループを抜けます．

CWのチェックは，プリミティブ・ブロック転送命令の実行前，すなわちREPNC命令実行直前の状態に対して行われます．したがって，最初CW=0でREPNC命令実行に入ると，続くプリミティブ・ブロック転送命令は１回も実行されずにその次の命令に移ります．

ＣＹのチェックは，あくまで続くプリミティブ・ブロック転送命令の結果に対して行われ，最初REPNC命令に入る直前の内容は関係ありません．

⑦ フラグの動作

なし

⑧ 記述例

REPNC　CMPMB

⑨ 注　意

この命令と次の命令との間では，ハードウエア割り込み（マスカブル割り込み，ノンマスカブル割り込み）要求およびシングル・ステップ・ブレークは受け付けられません．

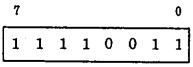
10．2．3 REP／REPE／REPZ（Repeat／Repeat while Equal／Repeat while Zero）

① 記述形式

REP　　（オペランドなし）

REPE／REPZ（オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ＣＷ≒０の間，続くバイトのブロック転送／比較／入出力命令を実行し，ＣＷをデクリメント（－１）します．

ＲＥＰはＭＯＶＢＫ，ＬＤＭ，ＳＴＭ，ＯＵＴＭ，ＩＮＭの各命令とともに用いられ，Ｚの値に関係なくＣＷ≒０の間くり返し処理を行います．

REPZおよびREPEはCMPBK，CMPMの各命令とともに用いられ，各ブロック命令による比較の結果Ｚ≒１になるか，またはＣＷ＝０になるとループを抜けます．

ＣＷのチェックは，ブロック命令の実行前，すなわちREP／REPE／REPZ 命令実行直前の状態に対して行われます．したがって，最初ＣＷ＝０でREP／REPE／REPZ 命令実行に入ると，続くブロック命令は１回も実行されずにその次の命令に移ります．

Ｚのチェックは，あくまで続くブロック命令の結果に対して行われ，最初REPE／REPZ命令に入る直前の内容は関係ありません．

⑦ フラグの動作

なし

⑧ 記述例

```
REP  MOVBKW
REPZ CMPBKW
```

⑨ 注　意

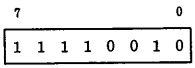
この命令と次の命令との間では，ハードウエア割り込み（マスカブル割り込み，ノンマスカブル割り込み）要求およびシングル・ステップ・ブレークは受け付けられません．

10．2．4 REPNE／REPNZ（Repeat while Not Equal／Repeat while Not Zero）

① 記述形式

REPNE／REPNZ （オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

CW≒0の間，続くバイトのブロック比較命令（CMPBK，CMPM）を実行し，CWをデクリメント（－1）します．

ブロック比較命令の結果，Z≒0になるか，またはCW＝0になるとループを抜けます．

CWのチェックは，ブロック比較命令の実行前，すなわちREPNE／REPNZ命令実行直前に対して行われます．したがって，最初CW＝0でREPNE／REPNZ命令実行に入ると，続くブロック比較命令は1回も実行されずにその次の命令に移ります．

Zのチェックは，あくまで続くブロック比較命令の結果に対して行われ，最初REPNE／REPNZ命令に入る直前の内容は関係ありません．

⑦ フラグの動作

なし

⑧ 記述例

```
REPNE  CMPMB
REPNZ  CMPBKW
```

⑨ 注　意

この命令と次の命令との間では，ハードウエア割り込み（マスカブル割り込み，ノンマスカブル割り込み）要求およびシングル・ステップ・ブレークは受け付けられません．

10.3　プリミティブ・ブロック転送命令

10.3.1　MOVBK／MOVBKB／MOVBKW(Move Block／Move Block Byte／Move Block Word)

① 記述形式

(repeat) MOVBK　〔DS1-spec:〕dst-block,〔Seg-spec:〕src-block

(repeat) MOVBKB (オペランドなし)

(repeat) MOVBKW (オペランドなし)

② 命令語形式

7							0
1	0	1	0	0	1	0	W

③ バイト数

1

④ クロック数

繰り返す場合	W＝0のとき	6／rep
	W＝1のとき	10／rep：奇数，奇数アドレス
		8／rep　：奇数，偶数アドレス
		6／rep　：偶数，偶数アドレス
1回の場合	W＝0のとき	6
	W＝1のとき	10：奇数，奇数アドレス
		8　：奇数，偶数アドレス
		6　：偶数，偶数アドレス

⑤ 16ビット・ワード転送回数

繰り返す場合　2／rep

1回の場合　2

⑥ 機　能

W＝0のとき　(IY)←(IX)

DIR＝0：IX←IX＋1，IY←IY＋1

DIR＝1：IX←IX－1，IY←IY－1

W＝1のとき　(IY＋1，IY)←(IX＋1，IX)

DIR＝0：IX←IX＋2，IY←IY＋2

DIR＝1：IX←IX－2，IY←IY－2

ＩＸでアドレスされるブロックを，ＩＹでアドレスされるブロックに，バイトまたはワード・データの繰り返しで転送します．

ＩＸ，ＩＹレジスタは，次のバイト／ワード転送のために１バイト／ワード転送ごとに自動的にインクリメント（＋１／＋２），またはデクリメント（－１／－２）されます．ブロックの方向は，方向フラグ（ＤＩＲ）の状態によって決定されます．

バイトかワードかの指定は，MOVBKを用いる場合はオペランドの属性によって行われ，MOVBKBまたはMOVBKWを用いる場合は，それぞれバイト，ワード・タイプに直接指定されます．

デスティネーション・ブロックは常にセグメント・レジスタＤＳ１で指されるセグメント内にロケートされる必要があり，セグメント・オーバライドはできません．一方，ソース・ブロックは，ディフォールト・セグメント・レジスタはＤＳ０となっていますが，セグメント・オーバライド可能で，任意のセグメント・レジスタで指されるセグメント内にロケートできます．

⑦ フラグの動作

なし

⑧ 記述例

```
MOVBK  BYTE_VAR,BYTE_VAR
MOVBK  WORD_VAR,WORD_VAR
```

10．3．2 CMPBK／CMPBKB／CMPBKW（Compare Block ／Compare Block Byte／Compare Block Word）

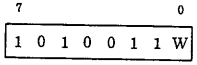
① 記述形式

（repeat） CMPBK 〔Seg-spec:〕src-block,〔DS1-spec:〕dst-block

（repeat） CMPBKB （オペランドなし）

（repeat） CMPBKW （オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

繰り返す場合　W＝0のとき　12／rep－1

W＝1のとき　16／rep－1：奇数，奇数アドレス

14／rep－1：奇数，偶数アドレス

12／rep－1：偶数，偶数アドレス

1回の場合　W＝0のとき　11

W＝1のとき　15：奇数，奇数アドレス

13：奇数，偶数アドレス

11：偶数，偶数アドレス

⑤ 16ビット・ワード転送回数

繰り返す場合　2／rep

1回の場合　2

⑥ 機　能

W＝0のとき　（IX）－（IY）

DIR＝0：IX←IX＋1，IY←IY＋1

DIR＝1：IX←IX－1，IY←IY－1

W＝1のとき　（IX＋1，IX）－（IY＋1，IY）

DIR＝0：IX←IX＋2，IY←IY＋2

DIR＝1：IX←IX－2，IY←IY－2

IXでアドレスされるブロックからIYでアドレスされるブロックをバイトまたはワード・データ単位に繰り返し減算を行い，結果をフラグに反映させます．

IX，IYは，次のバイト／ワード処理のために1バイト／ワード・データが処理されるごとに自動的にインクリメント（＋1／＋2）またはデクリメント（－1／－2）されます．ブロックの方向は，方向フラグ（DIR）の状態によって決定されます．

バイトかワードかの指定は，CMPBKを用いる場合はオペランドの属性によって行われ，CMPBKBまたはCMPBKWを用いる場合はそれぞれバイト，ワード・タイプに直接指定されます．

デスティネーション・ブロックは常にセグメント・レジスタＤＳ１で指されるセグメント内にローケートされる必要があり，セグメント・オーバライドはできません．一方，ソース・ブロックは，ディフォールト・セグメント・レジスタはＤＳ０となっていますが，セグメント・オーバライド可能で，任意のセグメント・レジスタで指されるセグメント内にローケートできます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

⑧ 記述例

```
CMPBK  BYTE_VAR, BYTE_VAR
```

10．3．3 CMPM／CMPMB／CMPMW(Compare Multiple／Compare Multiple Byte／Compare Multiple Word)

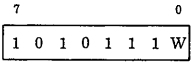
① 記述形式

（repeat） CMPM 〔DS1-spec：〕dst-block

（repeat） CMPMB（オペランドなし）

（repeat） CMPMW（オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

繰り返す場合　W＝0のとき　10／rep－1

W＝1のとき　12／rep－1：奇数アドレス

10／rep－1：偶数アドレス

1回の場合　W＝0のとき　9

W＝1のとき　11：奇数アドレス

9　：偶数アドレス

⑤ 16ビット・ワード転送回数

繰り返す場合　1／rep

1回の場合　1

⑥ 機　能

W＝0のとき　AL－（IY）

DIR＝0：IY←IY＋1

DIR≧1：IY←IY－1

W＝1のとき　AW－（IY＋1，IY）

DIR＝0：IY←IY＋2

DIR＝1：IY←IY－2

アキュームレータ（AL／AW）からIYでアドレスされるブロックをバイトまたはワード・データ単位に繰り返し減算を行い，結果をフラグに反映させます．

IYは次のバイト／ワード処理のために1バイト／ワード・データが処理されるごとに自動的にインクリメント（＋1／＋2）またはデクリメント（－1／－2）されます．ブロックの方向は，方向フラグ（DIR）の状態によって決定されます．

バイトかワードかの指定は，CMPMを用いる場合はオペランドの属性によって行われ，CMPMBまたはCMPMWを用いる場合はそれぞれバイト，ワード・タイプに直接指定され

ます.

デスティネーション・ブロックは，常にセグメント・レジスタＤＳ１で指されるセグメント内にローケートされる必要があり，セグメント・オーバライドはできません.

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

⑧ 記述例

```
REPC  CMPM  BYTE_VAR,BYTE_VAR
REPNC CMPMW
REPZ  CMPMB
```

10．3．4 LDM／LDMB／LDMW（Load Multiple／Load Multiple Byte／Load Multiple Word）

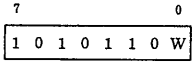
① 記述形式

（repeat） LDM 〔Seg-spec:〕src-block

（repeat） LDMB （オペランドなし）

（repeat） LDMW （オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

繰り返す場合　W＝0のとき　3／rep＋2

W＝1のとき　5／rep＋2：奇数アドレス

3／rep＋2：偶数アドレス

1回の場合　W＝0のとき　5

W＝1のとき　7：奇数アドレス

5：偶数アドレス

⑤ 16ビット・ワード転送回数

繰り返す場合　1／rep

1回の場合　1

⑥ 機　能

W＝0のとき　AL←（IX）

DIR＝0：IX←IX＋1

DIR＝1：IX←IX－1

W＝1のとき　AW←（IX＋1，IX）

DIR＝0：IX←IX＋2

DIR＝1：IX←IX－2

IXでアドレスされるブロックをバイトまたはワード単位にアキュームレータ（AL／AW）に繰り返し転送します．

IXは次のバイト／ワード処理のために1バイト／ワード・データが処理されるごとに自動的にインクリメント（＋1／＋2）またはデクリメント（－1／－2）されます．ブロックの方向は，方向フラグ（DIR）の状態によって決定されます．

バイトかワードかの指定は，LDMを用いる場合はオペランドの属性によって行われ，LDMBまたはLDMWを用いる場合はそれぞれバイト，ワード・タイプに直接指定されます．

ソース・ブロックは，ディフォールト・セグメント・レジスタはＤＳ０となっていますが，セグメント・オーバライド可能で，任意のセグメント・レジスタで指されるセグメント内にロケートできます．

⑦ フラグの動作

なし

⑧ 記述例

```
REP  LDM  DS1:BYTE_VAR;DS1セグメント
REP  LDMB              ;DS0セグメント
```

10．3．5 STM／STMB／STMW(Store Multiple／Store Mutiple Byte／Store Multiple Word)

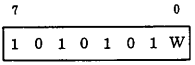
① 記述形式

(repeat) STM 〔DS1-spec：〕dst-block

(repeat) STMB （オペランドなし）

(repeat) STMW （オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

繰り返す場合　W＝0のとき　3／rep

W＝1のとき　5／rep：奇数アドレス

3／rep：偶数アドレス

1回の場合　W＝0のとき　3

W＝1のとき　5：奇数アドレス

3：偶数アドレス

⑤ 16ビット・ワード転送回数

繰り返す場合　1／rep

1回の場合　1

⑥ 機　能

W＝0のとき　(IY)←AL

DIR＝0：IY←IY＋1

DIR＝1：IY←IY－1

W＝1のとき　(IY＋1，IY)←AW

DIR＝0：IY←IY＋2

DIR＝1：IY←IY－2

AL／AWをIYでアドレスされるブロックにバイト／ワード・データ単位に繰り返し転送します．

IYは次のバイト／ワード処理のために1バイト／ワード・データが処理されるごとに自動的にインクリメント(＋1／＋2)またはデクリメント(－1／－2)されます．ブロックの方向は，方向フラグ(DIR)の状態によって決定されます．

バイトかワードかの指定は，STMを用いる場合はオペランドの属性によって，STMBまたはSTMWを用いる場合は，それぞれバイト，ワード・タイプに直接指定されます．

デスティネーション・ブロックは，常にセグメント・レジスタＤＳ１で指されるセグメント内にロケートされる必要があり，セグメント・オーバライドはできません．

⑦ フラグの動作

なし

⑧ 記述例

```
REP  STM  DS1:WORD_VAR        ;ＤＳ１セグメント
REP  STMB                     ;ＤＳ１セグメント
```

10．4　ビット・フィールド操作命令

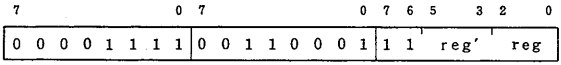
10．4．1　INS(Insert Bit Field)

(1)　レジスタ

①　記述形式

INS　reg8,reg8'

②　命令語形式



③　バイト数

3

④　クロック数

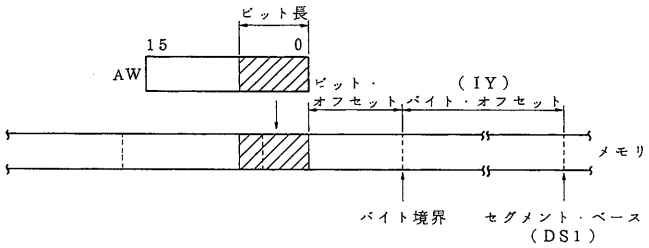
39－77：奇数アドレス

37－69：偶数アドレス

⑤　16ビット・ワード転送回数

2または4

⑥　機　能



AWレジスタの16ビット・データのうち，第2オペランドの8ビット・レジスタで指定される長さの下位ビット・データを，セグメント・レジスタDS1とインデクス・レジスタIYでアドレスされるバイト・オフセットおよび第1オペランドの8ビット・レジスタで指定されるビット・オフセットによって決定されるメモリ領域へ転送します．

転送終了後，IYおよび第1オペランドで指定された8ビット・レジスタは，次のビット・フィールドを示すために自動的に次のごとく更新されます．

```
reg8（第１オペランド）← reg8（第１オペランド）＋reg8'（第２オペランド）
＋１
if reg8（第１オペランド）＞１５  then
    {
     reg8（第１オペランド）← reg8（第１オペランド）－１６
     IY ← IY＋２
    }
```

ビット・オフセット（最長１５ビット）を指定する第１オペランドの８ビット・レジスタの値は，０－１５のみ有効です．また，ビット長（最長１６ビット）を指定する第２オペランドの８ビット・レジスタの値は，０－１５のみが有効で，０が１ビット長を，１５が１６ビット長を指定します．

ビット・フィールド・データは，メモリのバイト境界にまたがることができます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	U	U	U

⑧ 記述例

```
INS DL,CL
```

⑨ 注 意

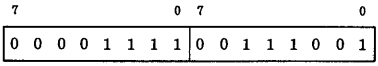
第１，第２オペランドとして使用される８ビット・レジスタの上位４ビットは０にして下さい．

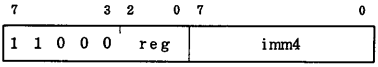
(2) イミーディエト・データ

① 記述形式

INS reg8, imm4

② 命令語形式





③ バイト数

4

④ クロック数

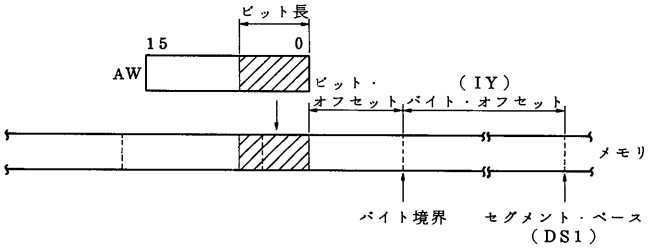
39－77：奇数アドレス

37－69：偶数アドレス

⑤ 16ビット・ワード転送回数

2または4

⑥ 機 能



AWレジスタの16ビット・データのうち，第2オペランドの4ビット・イミーディエト・データで指定される長さの下位ビット・データを，セグメント・レジスタDS1とインデクス・レジスタIYでアドレスされるバイト・オフセットおよび第1オペランドの8ビット・レジスタで指定されるビット・オフセットによって決定されるメモリ領域へ転送します．

転送終了後，IYおよび第1オペランドで指定された8ビット・レジスタは，次のビット・フィールドを示すために自動的に次のごとく更新されます．

```
reg8 ← reg8 + imm4 + 1
if reg8 >15 then
    {
     reg8 ← reg8 - 16
     IY ← IY+2
    }
```

ビット・オフセット（最長１５ビット）を指定する第１オペランドの８ビット・レジスタの値は，０－１５のみ有効です．また，ビット長（最長１６ビット）を指定する第２オペランドのイミーディエト・データ値は，０－１５のみが有効で，０が１ビット長を，１５が１６ビット長を指定します．

ビット・フィールド・データは，メモリのバイト境界にまたがることができます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	U	U	U

⑧ 記述例

```
INS DL,12
```

⑨ 注　意

第１オペランドとして使用される８ビット・レジスタの上位４ビットは０にして下さい．

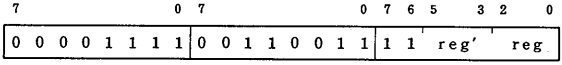
10．4．2 EXT(Extract Bit Field)

(1) レジスタ

① 記述形式

EXT reg8,reg8'

② 命令語形式



③ バイト数

3

④ クロック数

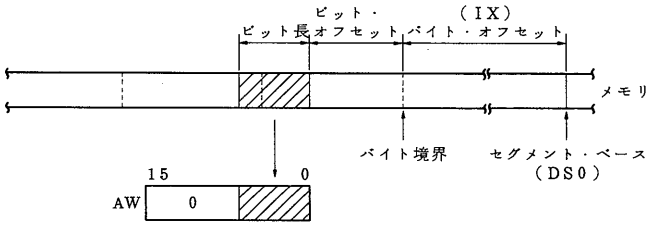
３３－６３：奇数アドレス

２９－６１：偶数アドレス

⑤ １６ビット・ワード転送回数

１または２

⑥ 機 能



セグメント・レジスタＤＳ０とインデクス・レジスタＩＸでアドレスされるバイト・オフセットおよび第１オペランドの８ビット・レジスタで指定されるビット・オフセットによって決定されるメモリ領域から，第２オペランドの８ビット・レジスタによって指定されるビット長のビット・フィールド・データをＡＷレジスタへロードします．このときＡＷの余りの上位ビットには０がロードされます．

転送終了後，ＩＸおよび第１オペランドで指定された８ビット・レジスタは，次のビット・フィールドを示すために自動的に次のごとく更新されます．

```
reg8（第１オペランド）← reg8（第１オペランド）＋reg8'（第２オペランド）
＋１
if reg8（第１オペランド）＞１５  then
    {
     reg8（第１オペランド）← reg8（第１オペランド）－１６
     IX ← IX＋２
    }
```

ビット・オフセット（最長１５ビット）を指定する第１オペランドの８ビット・レジスタの値は０－１５のみ有効です．また，ビット長（最長１６ビット）を指定する第２オペランドの８ビット・レジスタの値は，０－１５のみが有効で，０が１ビット長を，１５が１６ビット長を指定します．

ビット・フィールド・データは，メモリのバイト境界にまたがることができます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	U	U	U

⑧ 記述例

```
EXT  CL,DL
```

⑨ 注　意

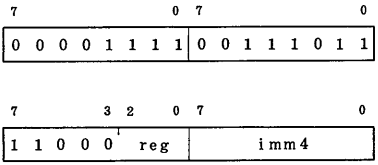
第１，第２オペランドとして使用される８ビット・レジスタの上位４ビットは０にして下さい．

(2) イミーディエト・データ

① 記述形式

EXT reg8,imm4

② 命令語形式



③ バイト数

4

④ クロック数

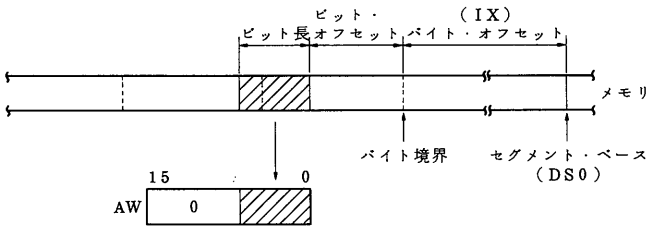
３３－６３：奇数アドレス

２９－６１：偶数アドレス

⑤ １６ビット・ワード転送回数

１または２

⑥ 機　能



セグメント・レジスタＤＳ０とインデクス・レジスタＩＸでアドレスされるバイト・オフセットおよび第１オペランドの８ビット・レジスタで指定されるビット・オフセットによって決定されるメモリ領域から，第２オペランドの４ビット・イミーディエト・データによって指定されるビット長のビット・フィールド・データをAWレジスタへロードします．このときAWの余りの上位ビットには０がロードされます．

転送終了後，ＩＸおよび第１オペランドで指定された８ビット・レジスタは，次のビット・フィールドを示すために自動的に次のごとく更新されます．

```
reg8 ← reg8 + imm4 + 1
if reg8 >15 then
    {
     reg8 ← reg8 − 16
     IX ← IX+2
    }
```

ビット・オフセット（最長１５ビット）を指定する第１オペランドの８ビット・レジスタの値は０－１５のみ有効です．また，ビット長（最長１６ビット）を指定する第２オペランドのイミーディエト・データ値は，０－１５のみが有効で，０が１ビット長を，１が１６ビット長を指定します．

ビット・フィールド・データは，メモリのバイト境界にまたがることができます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	U	U	U

⑧ 記述例

```
EXT CL,8
```

⑨ 注 意

第１オペランドとして使用される８ビット・レジスタの上位４ビットは０にして下さい．

10．5　入出力命令

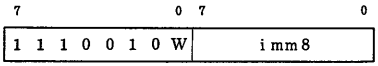
10．5．1　IN（Input）

(1) 直接指定I／Oデバイス

① 記述形式

IN　acc，imm8

② 命令語形式



③ バイト数

2

④ クロック数

W＝0のとき　5

W＝1のとき　7：奇数アドレス

5：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機　能

W＝0のとき　AL←（imm8）

W＝1のとき　AW←（imm8＋1，imm8）

第1オペランドで指定されるアキュームレータ（AL／AW）に，第2オペランドで指定されるI／Oデバイスの内容を入力します．I／Oアドレスの上位8ビット（A8－A15）には0が出力されます．

⑦ フラグの動作

なし

⑧ 記述例

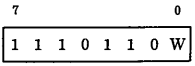
IN　AL，20H

⑵ ＤＷ間接指定Ｉ／Ｏデバイス

① 記述形式

ＩＮ　ａｃｃ，ＤＷ

② 命令語形式



③ バイト数

1

④ クロック数

Ｗ＝０のとき　５

Ｗ＝１のとき　７：奇数アドレス

５：偶数アドレス

⑤ １６ビット・ワード転送回数

1

⑥ 機　能

Ｗ＝０のとき　ＡＬ←（ＤＷ）

Ｗ＝１のとき　ＡＷ←（ＤＷ＋１，ＤＷ）

第１オペランドで指定されるアキュームレータ（ＡＬ／ＡＷ）に，レジスタＤＷで指定されるＩ／Ｏデバイスの内容を入力します．

⑦ フラグの動作

なし

⑧ 記述例

ＩＮ　ＡＬ，ＤＷ

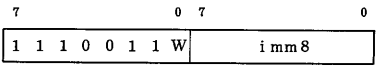
10．5．2 OUT（Output）

(1) 直接指定Ｉ／Ｏデバイス

① 記述形式

ＯＵＴ　ｉｍｍ８，ａｃｃ

② 命令語形式



③ バイト数

２

④ クロック数

Ｗ＝０のとき　３

Ｗ＝１のとき　５：奇数アドレス

３：偶数アドレス

⑤ １６ビット・ワード転送回数

１

⑥ 機　能

Ｗ＝０のとき　（ｉｍｍ８）←ＡＬ

Ｗ＝１のとき　（ｉｍｍ８＋１，ｉｍｍ８）←ＡＷ

第１オペランドで指定されるＩ／Ｏデバイスに，第２オペランドで指定されるアキュームレータ（ＡＬ／ＡＷ）の内容を入力します．Ｉ／Ｏアドレスの上位８ビット（Ａ８－Ａ１５）には０が出力されます．

⑦ フラグの動作

なし

⑧ 記述例

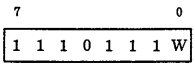
ＯＵＴ　３０Ｈ，ＡＷ

⑵ ＤＷ間接指定Ｉ／Ｏデバイス

① 記述形式

ＯＵＴ　ＤＷ，ａｃｃ

② 命令語形式



③ バイト数

1

④ クロック数

Ｗ＝０のとき　３

Ｗ＝１のとき　５：奇数アドレス

３：偶数アドレス

⑤ １６ビット・ワード転送回数

1

⑥ 機　能

Ｗ＝０のとき　（ＤＷ）←ＡＬ

Ｗ＝１のとき　（ＤＷ＋１，ＤＷ）←ＡＷ

レジスタＤＷの内容で指定されるＩ／Ｏデバイスに，第２オペランドで指定されるアキュームレータ（ＡＬ／ＡＷ）の内容を出力します．

⑦ フラグの動作

なし

⑧ 記述例

ＯＵＴ　ＤＷ，ＡＷ

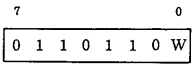
10.6　プリミティブ入出力命令

10.6.1　INM(Input Multiple)

① 記述形式

(repeat) INM 〔DS1-spec:〕dst-block,DW

② 命令語形式



③ バイト数

1

④ クロック数

繰り返す場合　W＝0のとき　8／rep＋4

W＝1のとき　14／rep＋8：奇数奇数アドレス

12／rep＋8：奇数偶数アドレス（I／Oが奇数）

10／rep＋4：　〃　（メモリが奇数）

8／rep＋4　：偶数偶数アドレス

1回の場合　W＝0のとき　12

W＝1のとき　22：奇数奇数アドレス

20：奇数偶数アドレス（I／Oが奇数）

14：　〃　（メモリが奇数）

12：偶数偶数アドレス

⑤ 16ビット・ワード転送回数

繰り返す場合　2／rep

1回の場合　2

⑥ 機　能

W＝0のとき　(IY)←(DW)

DIR＝0：IY←IY＋1

DIR＝1：IY←IY－1

W＝1のとき　(IY＋1,IY)←(DW＋1,DW)

DIR＝0：IY←IY＋2

DIR＝1：IY←IY－2

DWでアドレスされるI／Oデバイスの内容を，IYでアドレスされるメモリに転送します．繰り返し転送回数は，ペアで使用されるリピート・プリフィクスのREP命令によって制御されます．繰り返し転送の際，DWの内容（I／Oデバイスのアドレス）は固定ですが，IYは次のバイト／ワード転送のために1バイト／ワード・データが転送さ

れるごとに自動的にインクリメント（＋１／＋２）またはデクリメント（－１／－２）されます．ブロックの方向は，方向フラグ（ＤＩＲ）の状態によって決定されます．

バイトかワードかの指定は，オペランドの属性によって行われます．

ＩＮＭ命令は，リピート・プリフィクスのＲＥＰとともに使用されます．

デスティネーション・ブロックは，常にセグメント・レジスタＤＳ１で指されるセグメント内にローケートされる必要があり，セグメント・オーバライドはできません．

⑦ フラグの動作

なし

⑧ 記述例

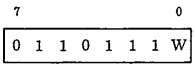
```
REP  INM  BYTE_VAR,DW
```

10．6．2 OUTM(Output Multiple)

① 記述形式

(repeat) OUTM DW,〔 Seg-spec：〕src-block

② 命令語形式



③ バイト数

1

④ クロック数

繰り返す場合　W＝0のとき　12／rep－6

W＝1のとき　22／rep－6：奇数奇数アドレス

20／rep－6：奇数偶数アドレス（I／Oが奇数）

14／rep－6：〃（メモリが奇数）

12／rep－6：偶数偶数アドレス

1回の場合　W＝0のとき　6

W＝1のとき　16：奇数奇数アドレス

14：奇数偶数アドレス（I／Oが奇数）

8　：〃（メモリが奇数）

6　：偶数偶数アドレス

⑤ 16ビット・ワード転送回数

繰り返す場合　2／rep

1回の場合　2

⑥ 機　能

W＝0のとき　（DW）←（IX）

DIR＝0：IX←IX＋1

DIR＝1：IX←IX－1

W＝1のとき　（DW＋1，DW）←（IX＋1，IX）

DIR＝0：IX←IX＋2

DIR＝1：IX←IX－2

IXでアドレスされるメモリの内容を，DWでアドレスされるI／Oデバイスに転送します．繰り返し転送回数は，ペアで使用されるリピート・プリフィクスのREP命令によって制御されます．繰り返し転送の際，DWの内容（I／Oデバイスのアドレス）は固定ですが，IXは次のバイト／ワード転送のために1バイト／ワード・データが転送されるごとに自動的にインクリメント（＋1／＋2）またはデクリメント（－1／－2）されます．ブロックの方向は，方向フラグの内容によって決定されます．

バイトかワードかの指定は，オペランドの属性によって行われます．

ＯＵＴＭ命令は，リピート・プリフィクスのＲＥＰとともに使用されます．

ソース・ブロックは，ディフォールト・セグメント・レジスタはＤＳ０となっていますが，セグメント・オーバライド可能で，任意のセグメント・レジスタで指されるセグメント内にローケートできます．

⑦ フラグの動作

なし

⑧ 記述例

```
REP  OUTM DW,DS1:BYTE_VAR
```

10．7　加減算命令

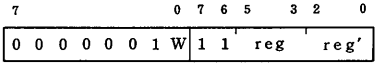
10．7．1 ADD（Add）

(1) レジスタ＋レジスタをレジスタへ

① 記述形式

ＡＤＤ　ｒｅｇ，ｒｅｇ′

② 命令語形式



③ バイト数

２

④ クロック数

２

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ｒｅｇ←ｒｅｇ＋ｒｅｇ′

第１オペランドで指定される８／１６ビット・レジスタの内容と第２オペランドで指定される８／１６ビット・レジスタの内容を加算し，結果を第１オペランドで指定されるレジスタにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

⑧ 記述例

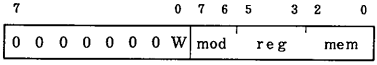
ＡＤＤ　ＡＷ，ＢＷ

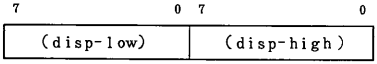
⑵ メモリ＋レジスタをメモリへ

① 記述方式

ＡＤＤ　ｍｅｍ，ｒｅｇ

② 命令語形式





③ バイト数

２／３／４

④ クロック数

Ｗ＝０のとき　７

Ｗ＝１のとき　１１：奇数アドレス

７：偶数アドレス

⑤ １６ビット・ワード転送回数

２

⑥ 機　能

（ｍｅｍ）←（ｍｅｍ）＋ｒｅｇ

第１オペランドでアドレスされる８／１６ビット・メモリの内容と第２オペランドで指定される８／１６ビット・レジスタの内容を加算し，結果を第１オペランドでアドレスされるメモリにストアします．

⑦ フラグの動作

Ｖ	Ｓ	Ｚ	ＡＣ	Ｐ	ＣＹ
×	×	×	×	×	×

⑧ 記述例

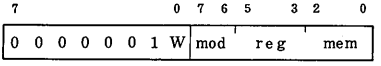
ADD WORD_VAR,AW

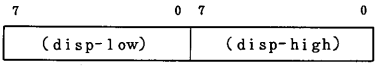
⑶ レジスタ＋メモリをレジスタへ

① 記述形式

ＡＤＤ　ｒｅｇ，ｍｅｍ

② 命令語形式





③ バイト数

２／３／４

④ クロック数

Ｗ＝０のとき　６

Ｗ＝１のとき　８：奇数アドレス

６：偶数アドレス

⑤ １６ビット・ワード転送回数

１

⑥ 機　能

ｒｅｇ←ｒｅｇ＋（ｍｅｍ）

第１オペランドで指定される８／１６ビット・レジスタの内容と第２オペランドでアドレスされる８／１６ビット・メモリの内容を加算し，結果を第１オペランドで指定されるレジスタにストアします．

⑦ フラグの動作

Ｖ	Ｓ	Ｚ	ＡＣ	Ｐ	ＣＹ
×	×	×	×	×	×

⑧ 記述例

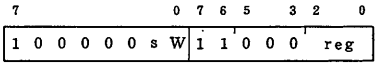
```
ADD AW,WORD_VAR
```

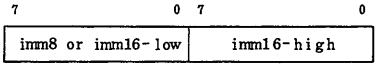
(4) レジスタ＋イミーディエト・データをレジスタへ

① 記述形式

ＡＤＤ　ｒｅｇ，ｉｍｍ

② 命令語形式





③ バイト数

３／４

④ クロック数

２

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ｒｅｇ←ｒｅｇ＋ｉｍｍ

第１オペランドで指定される８／１６ビット・レジスタの内容と第２オペランドで指定される８／１６ビット・イミーディエト・データを加算し，結果を第１オペランドで指定されるレジスタにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

⑧ 記述例

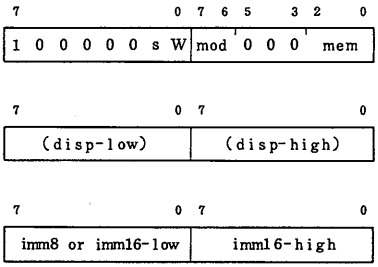
ＡＤＤ　ＡＬ，１０

(5) メモリ＋イミーディエト・データをメモリへ

① 記述形式

ＡＤＤ　ｍｅｍ，ｉｍｍ

② 命令語形式



③ バイト数

３／４／５／６

④ クロック数

Ｗ＝０のとき　７

Ｗ＝１のとき　１１：奇数アドレス

７：偶数アドレス

⑤ １６ビット・ワード転送回数

２

⑥ 機　能

（ｍｅｍ）←（ｍｅｍ）＋ｉｍｍ

第１オペランドでアドレスされる８／１６ビット・メモリの内容と第２オペランドで指定される８／１６ビット・イミーディエト・データを加算し，結果を第１オペランドでアドレスされるメモリにストアします．

⑦ フラグの動作

Ｖ	Ｓ	Ｚ	ＡＣ	Ｐ	ＣＹ
×	×	×	×	×	×

⑧ 記述例

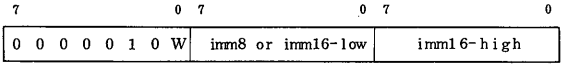
```
ADD BYTE_VAR[BP],100
```

(6) アキュームレータ＋イミーディエト・データをアキュームレータへ

① 記述形式

ＡＤＤ　ａｃｃ，ｉｍｍ

② 命令語形式



③ バイト数

２／３

④ クロック数

２

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

Ｗ＝０のとき　ＡＬ←ＡＬ＋ｉｍｍ８

Ｗ＝１のとき　ＡＷ←ＡＷ＋ｉｍｍ１６

第１オペランドで指定されるアキュームレータ（ＡＬ，ＡＷ）の内容と第２オペランドで指定される８／１６ビット・イミーディエト・データを加算し，結果を第１オペランドで指定されるアキュームレータにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

⑧ 記述例

ＡＤＤ　ＡＬ，３

10．8　BCD演算命令

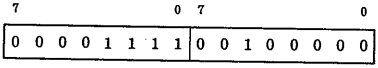
10．8．1 ADD4S(Add Nibble String)

① 記述形式

ADD4S 〔DS1-spec:〕dst-string,〔Seg-spec:〕src-string

ADD4S （オペランドなし）

② 命令語形式



③ バイト数

2

④ クロック数

18×n＋2　　n：BCDけた数の1／2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

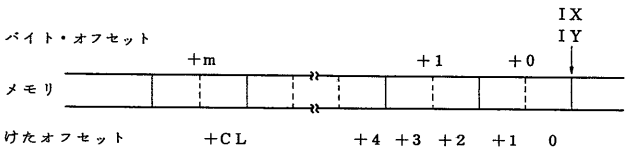
BCDストリング(IY,CL)←BCDストリング(IY,CL)

＋BCDストリング(IX,CL)

IXでアドレスされるパックトBCDストリングとIYでアドレスされるパックトBCDストリングを加算し，結果をIYでアドレスされるストリングにストアします．ストリング長（BCDけた数）は，CL（CLの内容がdならばdけた）によって決定され1－254けたまで可能です．

デスティネーション・ストリングは，常にセグメント・レジスタDS1で指されるセグメント内にローケートされる必要があり，セグメント・オーバライドはできません．一方，ソース・ストリングは，ディフォールト・セグメント・レジスタはDS0となっていますが，セグメント・オーバライド可能で，任意のセグメント・レジスタで指されるセグメント内にローケートできます．

パックトBCDストリングのフォーマットを次に示します．



⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	×	U	U	×

⑧ 記述例

⑨ 注 意

ＢＣＤストリング命令は常に偶数桁単位で動作します．このため桁数として偶数を指定したときは演算結果，および各フラグは正しく動きますが，桁数として奇数を指定した場合は奇数＋１の偶数桁演算を実行し，演算結果，および各フラグは偶数桁の演算結果を示しますので，桁数として奇数を指定する場合は次のようにしてください．

桁数が奇数のときのＢＣＤ加算命令は，最上位バイトの上位４ビットを“０”にしてから実行してください．この結果 キャリーは最上位バイトのビット４に示され，フラグには反映されません．

10．8．2 SUB4S(Subtract Nibble String)

① 記述形式

SUB4S 〔DS1-spec：〕dst-string，〔Seg-spec：〕src-string

SUB4S （オペランドなし）

② 命令語形式

7							0	7							0
0	0	0	0	1	1	1	1	0	0	1	0	0	0	1	0

③ バイト数

2

④ クロック数

18×n＋2　　n：BCDけた数の1／2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

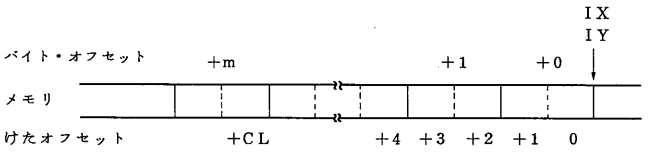
BCDストリング(IY，CL)←BCDストリング(IY，CL)

－BCDストリング(IX，CL)

IYでアドレスされるパックトBCDストリングからIXでアドレスされるパックトBCDストリングを減算し，結果をIYでアドレスされるストリングにストアします．ストリング長(BCDけた数)は，CL(CLの内容がdならばdけた)によって決定され，1－254けたまで可能です．

デスティネーション・ストリングは，常にセグメント・レジスタDS1で指されるセグメント内にローケートされる必要があり，セグメント・オーバライドはできません．一方，ソース・ストリングは，ディフォールト・セグメント・レジスタはDS0となっていますが，セグメント・オーバライド可能で，任意のセグメント・レジスタで指されるセグメント内にローケートできます．

パックトBCDストリングのフォーマットを次に示します．



⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	×	U	U	×

⑧ 記述例

⑨ 注　意

ＢＣＤストリング命令は常に偶数桁単位で動作します．このため桁数として偶数を指定したときは演算結果，および各フラグは正しく動きますが，桁数として奇数を指定した場合は奇数＋１の偶数桁演算を実行し，演算結果，および各フラグは偶数行の演算結果を示しますので，桁数として奇数を指定する場合は次のようにしてください．

桁数が奇数のときのＢＣＤ減算命令は，最上位バイトの上位４ビットを“０”にしてから実行してください．この結果ボローが生じた場合は最上位バイトの上位４ビットは“９”になります．

10．8．3 CMP4S(Compare Nibble String)

① 記述形式

CMP4S 〔DS1-spec：〕dst-string,〔Seg-spec：〕src-string

CMP4S （オペランドなし）

② 命令語形式

7							0	7							0
0	0	0	0	1	1	1	1	0	0	1	0	0	1	1	0

③ バイト数

2

④ クロック数

１４×ｎ＋２　　ｎ：ＢＣＤけた数の１／２

⑤ １６ビット・ワード転送回数

なし

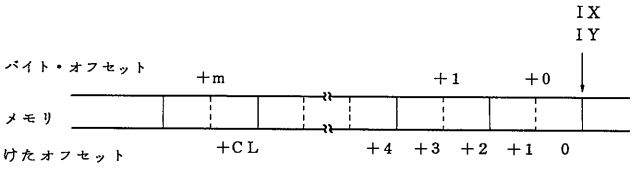
⑥ 機　能

ＢＣＤストリング（ＩＹ，ＣＬ）－ＢＣＤストリング（ＩＸ，ＣＬ）

ＩＹでアドレスされるパックトＢＣＤストリングからＩＸでアドレスされるパックトＢＣＤストリングを減算し，結果はストアされずフラグのみ影響を受けます．ストリング長（ＢＣＤけた数）は，ＣＬ（ＣＬの内容がｄならばｄけた）によって決定され，1-254けたまで可能です．

デスティネーション・ストリングは，常にセグメント・レジスタＤＳ１で指されるセグメント内にロケートされる必要があり，セグメント・オーバライドはできません．一方，ソース・ストリングは，ディフォールト・セグメント・レジスタはＤＳ０となっていますが，セグメント・オーバライド可能で，任意のセグメント・レジスタで指されるセグメント内にロケートできます．

パックトＢＣＤストリングのフォーマットを次に示します．



⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	×	U	U	×

⑧ 記述例

⑨ 注　意

ＢＣＤストリング命令は常に偶数桁単位で動作します．このため桁数として偶数を指定したときは演算結果，および各フラグは正しく動きますが，桁数として奇数を指定した場合は奇数＋１の偶数桁演算を実行し，演算結果，および各フラグは偶数桁の演算結果を示しますので，桁数として奇数を指定する場合は次のようにしてください．

桁数が奇数のときのＢＣＤ比較命令は，最上位バイトの上位４ビットを“０”にしてから実行してください．

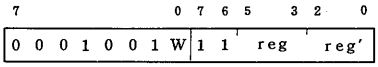
10．7．2 ADDC(Add with Carry)

(1) レジスタ＋レジスタをレジスタへ

① 記述形式

ＡＤＤＣ　ｒｅｇ，ｒｅｇ′

② 命令語形式



③ バイト数

２

④ クロック数

２

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ｒｅｇ←ｒｅｇ＋ｒｅｇ′＋ＣＹ

第１オペランドで指定される８／１６ビット・レジスタの内容と第２オペランドで指定される８／１６ビット・レジスタの内容をキャリー・フラグの内容も含めて加算し，結果を第１オペランドで指定されるレジスタにストアします．

⑦ フラグの動作

Ｖ	Ｓ	Ｚ	ＡＣ	Ｐ	ＣＹ
×	×	×	×	×	×

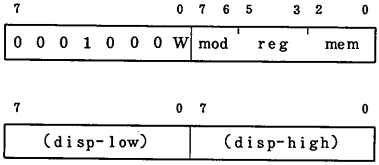
⑧ 記述例

⑵ メモリ＋レジスタをメモリへ

① 記述形式

ADDC mem,reg

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　7

W＝1のとき　11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能

(mem)←(mem)＋reg＋CY

第1オペランドでアドレスされる8／16ビット・メモリの内容と第2オペランドで指定される8／16ビット・レジスタの内容をキャリー・フラグの内容も含めて加算し，結果を第1オペランドでアドレスされるメモリにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

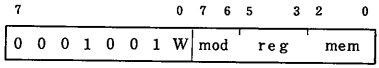
⑧ 記述例

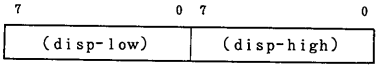
(3) レジスタ＋メモリをレジスタへ

① 記述形式

ＡＤＤＣ　ｒｅｇ，ｍｅｍ

② 命令語形式





③ バイト数

２／３／４

④ クロック数

Ｗ＝０のとき　６

Ｗ＝１のとき　８：奇数アドレス

６：偶数アドレス

⑤ １６ビット・ワード転送回数

１

⑥ 機　能

ｒｅｇ←ｒｅｇ＋（ｍｅｍ）＋ＣＹ

第１オペランドで指定される８／１６ビット・レジスタの内容と第２オペランドでアドレスされる８／１６ビット・メモリの内容をキャリー・フラグの内容も含めて加算し，結果を第１オペランドで指定されるレジスタにストアします．

⑦ フラグの動作

Ｖ	Ｓ	Ｚ	ＡＣ	Ｐ	ＣＹ
×	×	×	×	×	×

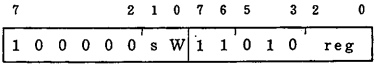
⑧ 記述例

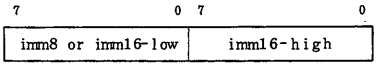
(4) レジスタ+イミーディエト・データをレジスタへ

① 記述形式

ＡＤＤＣ　ｒｅｇ，ｉｍｍ

② 命令語形式





③ バイト数

３／４

④ クロック数

２

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ｒｅｇ←ｒｅｇ＋ｉｍｍ＋ＣＹ

第１オペランドで指定される８／１６ビット・レジスタの内容と第２オペランドで指定される８／１６ビット・イミーディエト・データをキャリー・フラグの内容も含めて加算し，結果を第１オペランドで指定されるレジスタにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

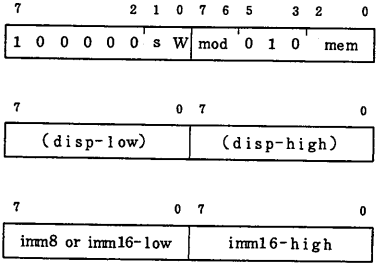
⑧ 記述例

⑸ メモリ＋イミーディエト・データをメモリへ

① 記述形式

ＡＤＤＣ　ｍｅｍ，ｉｍｍ

② 命令語形式



③ バイト数

３／４／５／６

④ クロック数

Ｗ＝０のとき　７

Ｗ＝１のとき　１１：奇数アドレス

７：偶数アドレス

⑤ １６ビット・ワード転送回数

２

⑥ 機　能

（ｍｅｍ）←（ｍｅｍ）＋ｉｍｍ＋ＣＹ

第１オペランドでアドレスされる８／１６ビット・メモリの内容と第２オペランドで指定される８／１６ビット・イミーディエト・データをキャリー・フラグの内容も含めて加算し，結果を第１オペランドでアドレスされるメモリにストアします．

⑦ フラグの動作

Ｖ	Ｓ	Ｚ	ＡＣ	Ｐ	ＣＹ
×	×	×	×	×	×

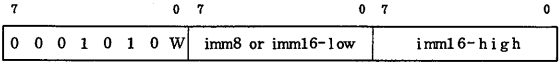
⑧ 記述例

⑹ アキュームレータ＋イミーディエト・データをアキュームレータへ

① 記述形式

ＡＤＤＣ　ａｃｃ，ｉｍｍ

② 命令語形式



③ バイト数

２／３

④ クロック数

２

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

Ｗ＝０のとき　ＡＬ←ＡＬ＋ｉｍｍ８＋ＣＹ

Ｗ＝１のとき　ＡＷ←ＡＷ＋ｉｍｍ１６＋ＣＹ

第１オペランドで指定されるアキュームレータ（ＡＬ／ＡＷ）の内容と第２オペランドで指定される８／１６ビット・イミーディエト・データをキャリー・フラグの内容も含めて加算し，結果を第１オペランドで指定されるアキュームレータにストアします．

⑦ フラグの動作

Ｖ	Ｓ	Ｚ	ＡＣ	Ｐ	ＣＹ
×	×	×	×	×	×

⑧ 記述例

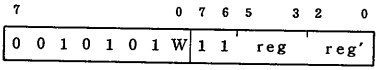
10．7．3 SUB（Subtract）

(1) レジスター レジスタをレジスタへ

① 記述形式

SUB reg，reg′

② 命令語形式



③ バイト数

2

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg←reg－reg′

第1オペランドで指定される8／16ビット・レジスタの内容から第2オペランドで指定される8／16ビット・レジスタの内容を減算し，結果を第1オペランドで指定されるレジスタにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

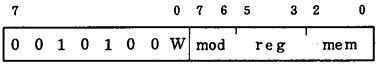
⑧ 記述例

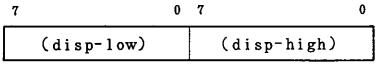
(2) メモリーレジスタをメモリへ

① 記述形式

SUB mem,reg

② 命令語形式





③ バイト数

2／3／4

④ クロック数

W＝0のとき 7

W＝1のとき 11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

(mem)←(mem)－reg

第1オペランドでアドレスされる8／16ビット・メモリの内容から第2オペランドで指定される8／16ビット・レジスタの内容を減算し，結果を第1オペランドでアドレスされるメモリにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

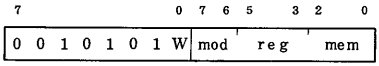
⑧ 記述例

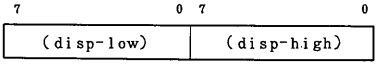
⑶ レジスタ－メモリをレジスタへ

① 記述形式

ＳＵＢ　ｒｅｇ，ｍｅｍ

② 命令語形式





③ バイト数

２／３／４

④ クロック数

Ｗ＝０のとき　６

Ｗ＝１のとき　８：奇数アドレス

６：偶数アドレス

⑤ １６ビット・ワード転送回数

１

⑥ 機　能

ｒｅｇ←ｒｅｇ－（ｍｅｍ）

第１オペランドで指定される８／１６ビット・レジスタから第２オペランドでアドレスされる８／１６ビット・メモリの内容を減算し，結果を第１オペランドで指定されるレジスタにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

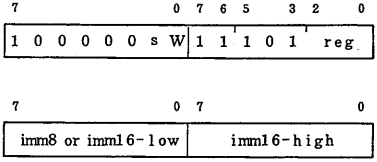
⑧ 記述例

(4) レジスターイミーディエト・データをレジスタへ

① 記述形式

SUB reg, imm

② 命令語形式



③ バイト数

3／4

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg←reg－imm

第1オペランドで指定される8／16ビット・レジスタの内容から第2オペランドで指定される8／16ビット・イミーディエト・データを減算し，結果を第1オペランドで指定されるレジスタにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

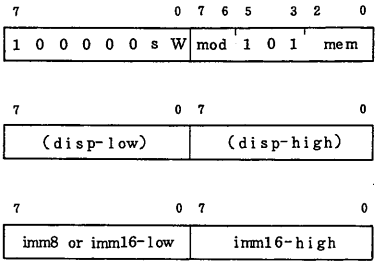
⑧ 記述例

(5) メモリーイミーディエト・データをメモリへ

① 記述形式

SUB mem, imm

② 命令語形式



③ バイト数

3／4／5／6

④ クロック数

W＝0のとき 7

W＝1のとき 11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

(mem)←(mem)－imm

第1オペランドでアドレスされる8／16ビット・メモリの内容から第2オペランドで指定される8／16ビット・イミーディエト・データを減算し，結果を第1オペランドでアドレスされるメモリにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

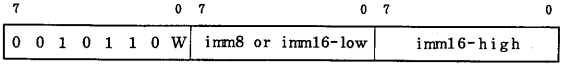
⑧ 記述例

(6) アキュームレーターイミーディエト・データをアキュームレータへ

① 記述形式

SUB acc,imm

② 命令語形式



③ バイト数

2／3

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

W＝0のとき AL←AL－imm8

W＝1のとき AW←AW－imm16

第1オペランドで指定されるアキュームレータ（AL，AW）の内容から第2オペランドで指定される8／16ビット・イミーディエト・データを減算し，結果を第1オペランドで指定されるアキュームレータにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

⑧ 記述例

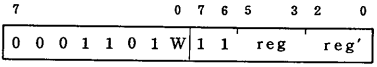
10．7．4 SUBC(Subtract with Carry)

(1) レジスター レジスタをレジスタへ

① 記述形式

SUBC reg，reg′

② 命令語形式



③ バイト数

2

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg← reg－ reg′＋CY

第1オペランドで指定される8／16ビット・レジスタの内容から第2オペランドで指定される8／16ビット・レジスタの内容をキャリー・フラグの内容も含めて減算します.

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

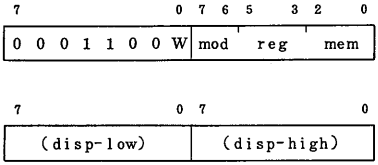
⑧ 記述例

⑵ メモリーレジスタをメモリへ

① 記述形式

SUBC mem,reg

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき 7

W＝1のとき 11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

(mem)←(mem)－reg－CY

第1オペランドでアドレスされる8／16ビット・メモリの内容から第2オペランドで指定される8／16ビット・レジスタの内容をキャリー・フラグの内容も含めて減算し，結果を第1オペランドでアドレスされるメモリにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

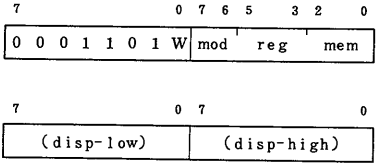
⑧ 記述例

(3) レジスターメモリをレジスタへ

① 記述形式

SUBC reg,mem

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき 6

W＝1のとき 8：奇数アドレス

6：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

reg←reg－(mem)－CY

第1オペランドで指定される8／16ビット・レジスタの内容から第2オペランドで指定される8／16ビット・レジスタの内容をキャリー・フラグの内容も含めて減算し，結果を第1オペランドで指定されるレジスタにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

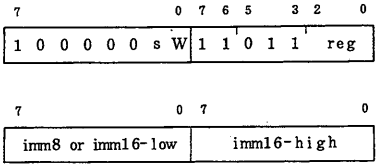
⑧ 記述例

(4) レジスターイミーディエト・データをレジスタへ

① 記述形式

SUBC reg,imm

② 命令語形式



③ バイト数

3／4

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg←reg－imm－CY

第1オペランドで指定される8／16ビット・レジスタの内容から第2オペランドで指定される8／16ビット・レジスタの内容をキャリー・フラグの内容も含めて減算し，結果を第1オペランドで指定されるレジスタにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

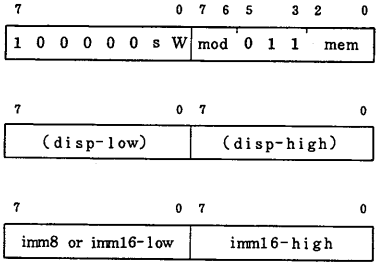
⑧ 記述例

⑸ メモリーイミーディエト・データをメモリへ

① 記述形式

SUBC mem, imm

② 命令語形式



③ バイト数

3／4／5／6

④ クロック数

W＝0のとき 7

W＝1のとき 11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

(mem)←(mem)－imm－CY

第1オペランドでアドレスされる8／16ビット・メモリの内容から第2オペランドで指定される8／16ビット・イミーディエト・データの内容をキャリー・フラグの内容も含めて減算し，結果を第1オペランドでアドレスされるメモリにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

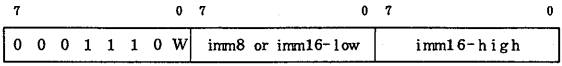
⑧ 記述例

(6) アキュームレーターイミーディエト・データをアキュームレータへ

① 記述形式

SUBC acc,imm

② 命令語形式



③ バイト数

2／3

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

W＝0のとき AL←AL－imm8－CY

W＝1のとき AW←AW－imm16－CY

第1オペランドで指定されるアキュームレータ（AL／AW）の内容から第2オペランドで指定される8／16ビット・イミーディエト・データをキャリー・フラグの内容も含めて減算し，結果を第1オペランドで指定されるアキュームレータにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

⑧ 記述例

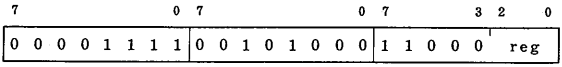
10．8．4 ROL4 (Rotate Left Nibble)

(1) 8ビット・レジスタ

① 記述形式

ＲＯＬ４　ｒｅｇ８

② 命令語形式



③ バイト数

3

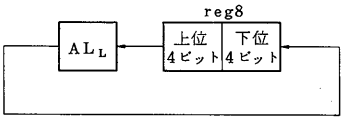
④ クロック数

9

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能



オペランドで指定される８ビット・レジスタのバイト・データを２けたのパックトBCDとして扱い，ＡＬレジスタの下位４ビット（ＡＬ$_L$）を介して，１けた分左に回転します．この命令の結果，ＡＬの上位４ビットは保証されません．

⑦ フラグの動作

なし

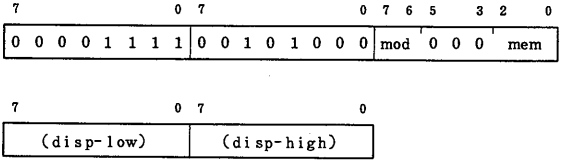
⑧ 記述例

⑵ 8ビット・メモリ

① 記述形式

ROL4 mem8

② 命令語形式



③ バイト数

3／4／5

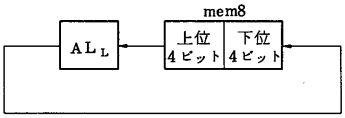
④ クロック数

15

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能



オペランドでアドレスされる8ビット・メモリのバイト・データを2けたのパックトBCDとして扱い，ALレジスタの下位4ビット（AL$_L$）を介して，1けた分左に回転します．

この命令の結果，ALの上位4ビットは保証されません．

⑦ フラグの動作

なし

⑧ 記述例

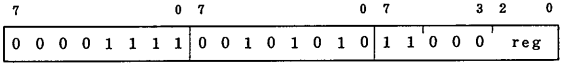
10．8．5 ROR4（Rotate Right Nibble）

(1) 8ビット・レジスタ

① 記述形式

ＲＯＲ４　ｒｅｇ８

② 命令語形式



③ バイト数

３

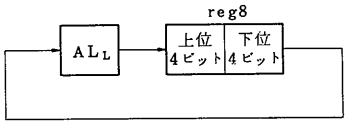
④ クロック数

１３

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能



オペランドで指定される８ビット・レジスタのバイト・データを２けたのパックトＢＣＤとして扱い，ＡＬレジスタの下位４ビット（ＡＬ$_L$）を介して，１けた分右に回転します．

この命令の結果，ＡＬの上位４ビットは保証されません．

⑦ フラグの動作

なし

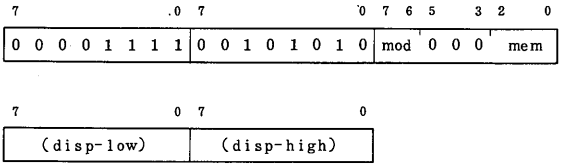
⑧ 記述例

⑵ 8ビット・メモリ

① 記述形式

ＲＯＲ４　ｍｅｍ８

② 命令語形式



③ バイト数

３／４／５

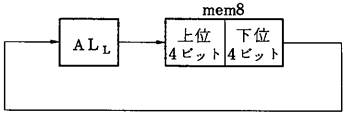
④ クロック数

１９

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能



オペランドでアドレスされる８ビット・メモリのバイト・データを２けたのパックＢＣＤとして扱い，ＡＬレジスタの下位４ビット（ＡＬ$_L$）を介して，１けた分右に回転します．

この命令の結果，ＡＬの上位４ビットは保証されません．

⑦ フラグの動作

なし

⑧ 記述例

10．9　増減命令

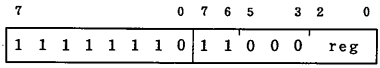
10．9．1　INC(Increment)

(1)　8ビット・レジスタ

①　記述形式

INC　reg8

②　命令語形式



③　バイト数

2

④　クロック数

2

⑤　16ビット・ワード転送回数

なし

⑥　機　能

reg←reg8＋1

オペランドで指定される8ビット・レジスタの内容をインクリメント(＋1)します.

⑦　フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	

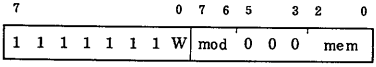
⑧　記述例

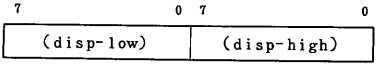
(2) メモリ

① 記述形式

ＩＮＣ　ｍｅｍ

② 命令語形式





③ バイト数

２／３／４

④ クロック数

Ｗ＝０のとき　７

Ｗ＝１のとき　１１：奇数アドレス

７：偶数アドレス

⑤ １６ビット・ワード転送

２

⑥ 機　能

（ｍｅｍ）←（ｍｅｍ）＋１

オペランドでアドレスされる８／１６ビット・メモリの内容をインクリメント（＋１）します。

⑦ フラグの動作

Ｖ	Ｓ	Ｚ	ＡＣ	Ｐ	ＣＹ
×	×	×	×	×	

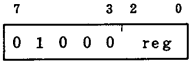
⑧ 記述例

⑶ １６ビット・レジスタ

① 記述形式

ＩＮＣ　ｒｅｇ１６

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ｒｅｇ１６←ｒｅｇ１６＋１

オペランドで指定される１６ビット・レジスタの内容をインクリメント（＋１）します．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	

⑧ 記述例

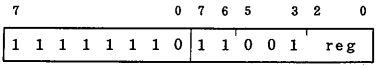
10．9．2 DEC（Decrement）

(1) 8ビット・レジスタ

① 記述形式

DEC reg8

② 命令語形式



③ バイト数

2

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg8←reg8－1

オペランドで指定される8ビット・レジスタの内容をデクリメント（－1）します.

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	

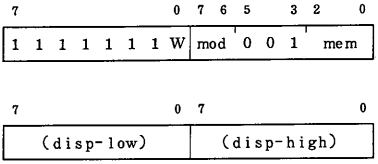
⑧ 記述例

(2) メモリ

① 記述形式

DEC mem

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき 7

W＝1のとき 11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

(mem)←(mem)－1

オペランドでアドレスされる8／16ビット・メモリの内容をデクリメント(－1)します.

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	

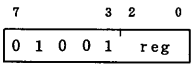
⑧ 記述例

(3) 16ビット・レジスタ

① 記述形式

DEC reg16

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg16←reg16－1

オペランドで指定される16ビット・レジスタの内容をデクリメント(－1)します.

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	

⑧ 記述例

10．10　乗算命令

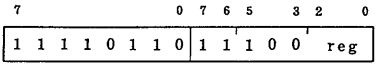
10．10．1　MULU(Multiply Unsigned)

(1)　8ビット・レジスタ

①　記述形式

MULU　reg8

②　命令語形式



③　バイト数

2

④　クロック数

8

⑤　16ビット・ワード転送回数

なし

⑥　機　能

AW←AL×reg8

AH＝0のとき　CY←0，V←0

AH≒0のとき　CY←1，V←1

レジスタALの内容とオペランドで指定される8ビット・レジスタの内容との符号なし乗算を行い，倍長結果をレジスタALとAHにストアします．結果の上位半分（AH）がゼロでないとき，キャリー・フラグとオーバフロー・フラグがセットされます．AHは拡張レジスタです．

⑦　フラグの動作

V	S	Z	AC	P	CY
×	U	U	U	U	×

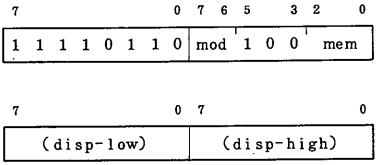
⑧　記述例

⑵ 8ビット・メモリ

① 記述形式

ＭＵＬＵ　ｍｅｍ８

② 命令語形式



③ バイト数

２／３／４

④ クロック数

１２

⑤ １６ビット・ワード転送回数

１

⑥ 機　能

ＡＷ←ＡＬ×（ｍｅｍ８）

ＡＨ＝０のとき　ＣＹ←０，Ｖ←０

ＡＨ≠０のとき　ＣＹ←１，Ｖ←１

レジスタＡＬの内容とオペランドでアドレスされる８ビット・メモリの内容との符号なし乗算を行い，倍長結果をレジスタＡＬとＡＨにストアします．結果の上位半分（ＡＨ）がゼロでないとき，キャリー・フラグとオーバフロー・フラグがセットされます．ＡＨは拡張レジスタです．

⑦ フラグの動作

Ｖ	Ｓ	Ｚ	ＡＣ	Ｐ	ＣＹ
×	Ｕ	Ｕ	Ｕ	Ｕ	×

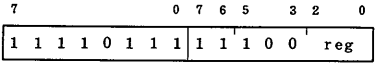
⑧ 記述例

⑶ １６ビット・レジスタ

① 記述形式

ＭＵＬＵ　ｒｅｇ１６

② 命令語形式



③ バイト数

２

④ クロック数

１２

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ＤＷ，ＡＷ←ＡＷ×ｒｅｇ１６

ＤＷ＝０のとき　ＣＹ←０，Ｖ←０

ＤＷ≒０のとき　ＣＹ←１，Ｖ←１

レジスタＡＷの内容とオペランドで指定される１６ビット・レジスタの内容との符号なし乗算を行い，倍長結果をレジスタＡＷとＤＷにストアします．結果の上位半分（ＤＷ）がゼロでないとき，キャリー・フラグとオーバフロー・フラグがセットされます．

ＤＷは拡張レジスタです．

⑦ フラグの動作

Ｖ	Ｓ	Ｚ	ＡＣ	Ｐ	ＣＹ
×	Ｕ	Ｕ	Ｕ	Ｕ	×

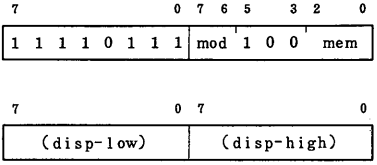
⑧ 記述例

(4) 16ビット・メモリ

① 記述形式

MULU mem16

② 命令語形式



③ バイト数

2／3／4

④ クロック数

18：奇数アドレス

16：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

DW，AW←AW×(mem16)

DW＝0のとき CY←0，V←0

DW≠0のとき CY←1，V←1

レジスタAWとオペランドでアドレスされる16ビット・メモリとの符号なし乗算を行い，倍長結果をレジスタAWとDWにストアします．結果の上位半分DWがゼロでないとき，キャリー・フラグとオーバフロー・フラグがセットされます．DWは拡張レジスタです．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	U	U	U	U	×

⑧ 記述例

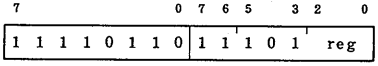
10．10．2 MUL(Multiply Signed)

(1) 8ビット・レジスタ

① 記述形式

MUL reg8

② 命令語形式



③ バイト数

2

④ クロック数

8

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

AW←AL×reg8

AH＝ALのサイン拡張のとき CY←0，V←0

AH≒ALのサイン拡張のとき CY←1，V←1

レジスタALとオペランドで指定される8ビット・レジスタとの符号付き乗算を行い，倍長結果をレジスタALとAHにストアします．結果の上位半分AHが下位半分ALの符号拡張でないとき，キャリー・フラグとオーバフロー・フラグがセットされます．AHは拡張レジスタです．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	U	U	U	U	×

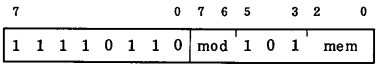
⑧ 記述例

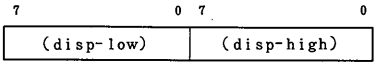
⑵ 8ビット・メモリ

① 記述形式

ＭＵＬ　ｍｅｍ8

② 命令語形式





③ バイト数

２／３／４

④ クロック数

１２

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ＡＷ←ＡＬ×（ｍｅｍ8）

ＡＨ＝ＡＬのサイン拡張のとき　ＣＹ←０，Ｖ←０

ＡＨ≠ＡＬのサイン拡張のとき　ＣＹ←１，Ｖ←１

レジスタＡＬとオペランドでアドレスされる８ビット・メモリとの符号付き乗算を行い，倍長結果をレジスタＡＬとＡＨにストアします．結果の上位半分ＡＨが下位半分のＡＬの符号拡張でないとき，キャリー・フラグとオーバフロー・フラグがセットされます．ＡＨは拡張レジスタです．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	U	U	U	U	×

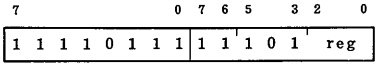
⑧ 記述例

(3) 16ビット・レジスタ

① 記述形式

MUL reg16

② 命令語形式



③ バイト数

2

④ クロック数

12

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

DW，AW←AW× reg16

DW＝AWのサイン拡張のとき CY←0，V←0

DW≒AWのサイン拡張のとき CY←1，V←1

レジスタAWとオペランドで指定される16ビット・レジスタとの符号付き乗算を行い，倍長結果をレジスタAWとDWにストアします．結果の上位半分DWが下位半分AWの符号拡張でないとき，キャリー・フラグとオーバフロー・フラグがセットされます．DWは拡張レジスタです．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	U	U	U	U	×

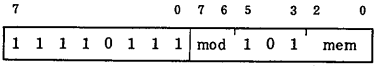
⑧ 記述例

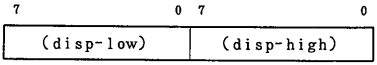
(4) 16ビット・メモリ

① 記述形式

MUL　mem16

② 命令語形式





③ バイト数

2／3／4

④ クロック数

18：奇数アドレス

16：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機　能

DW←AW×(mem16)

DW＝AWのサイン拡張のとき　CY←0，V←0

DW≒AWのサイン拡張のとき　CY←1，V←1

レジスタAWとオペランドでアドレスされる16ビット・メモリとの符号付き乗算を行い，倍長結果をレジスタAWとDWにストアします．結果の上位半分DWが下位半分AWの符号拡張でないとき，キャリー・フラグとオーバフロー・フラグがセットされます．

DWは拡張レジスタです．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	U	U	U	U	×

⑧ 記述例

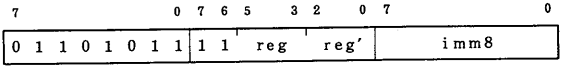
⑸ １６ビット・レジスタ×８ビット・イミーディエト・データを１６ビット・レジスタへ

① 記述形式

ＭＵＬ　ｒｅｇ１６，ｒｅｇ１６′，ｉｍｍ８

ＭＵＬ　ｒｅｇ１６，ｉｍｍ８

② 命令語形式



③ バイト数

３

④ クロック数

１２

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ｒｅｇ１６←ｒｅｇ１６′×ｉｍｍ８

積≦１６ビット：ＣＹ←０，Ｖ←０

積＞１６ビット：ＣＹ←１，Ｖ←１

第２オペランド（２オペランド記述のときは第１オペランド）で指定される16ビット・レジスタと第３オペランド（２オペランド記述のときは第２オペランド）で指定される８ビット・イミーディエト・データとの符号付き乗算を行い，結果を第１オペランドで指定される１６ビット・レジスタにストアします．

ソース・レジスタとデスティネーション・レジスタが同じで良い場合は，２オペランド記述が可能です．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	U	U	U	U	×

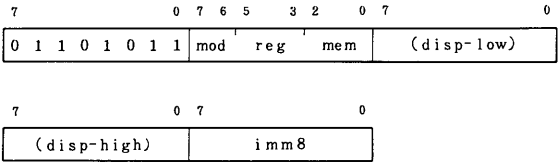
⑧ 記述例

⑹ １６ビット・メモリ×８ビット・イミーディエト・データを１６ビット・レジスタへ

① 記述形式

ＭＵＬ　ｒｅｇ１６，ｍｅｍ１６，ｉｍｍ８

② 命令語形式



③ バイト数

３／４／５

④ クロック数

１８：奇数アドレス

１６：偶数アドレス

⑤ １６ビット・ワード転送回数

1

⑥ 機　能

ｒｅｇ１６←（ｍｅｍ１６）×ｉｍｍ８

積≦１６ビット：ＣＹ←０，Ｖ←０

積＞１６ビット：ＣＹ←１，Ｖ←１

第２オペランドでアドレスされる１６ビット・メモリと第３オペランドで指定される８ビット・イミーディエト・データとの符号付き乗算を行い，結果を第１オペランドで指定される１６ビット・レジスタにストアします．

⑦ フラグの動作

Ｖ	Ｓ	Ｚ	ＡＣ	Ｐ	ＣＹ
×	Ｕ	Ｕ	Ｕ	Ｕ	×

⑧ 記述例

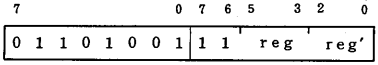
⑺ １６ビット・レジスタ×１６ビット・イミーディエト・データを１６ビット・レジスタへ

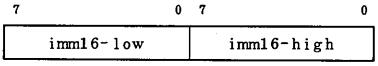
① 記述形式

ＭＵＬ　ｒｅｇ１６，ｒｅｇ１６′，ｉｍｍ１６

ＭＵＬ　ｒｅｇ１６，ｉｍｍ１６

② 命令語形式





③ バイト数

４

④ クロック数

１２

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ｒｅｇ１６←ｒｅｇ１６′×ｉｍｍ１６

積≦１６ビット：ＣＹ←０，Ｖ←０

積＞１６ビット：ＣＹ←１，Ｖ←１

第２オペランド（２オペランド記述のときは第１オペランド）で指定される１６ビット・レジスタと第３オペランド（２オペランド記述のときは第２オペランド）で指定される１６ビット・イミーディエト・データとの符号付き乗算を行い，結果を第１オペランドで指定される１６ビット・レジスタにストアします．

ソース・レジスタとデスティネーション・レジスタが同じで良い場合は，２オペランド記述が可能です．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	U	U	U	U	×

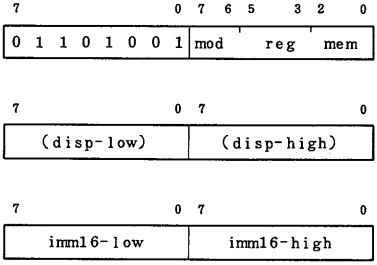
⑧ 記述例

⑧ １６ビット・メモリ×１６ビット・イミーディエト・データを１６ビット・レジスタへ

① 記述形式

ＭＵＬ　ｒｅｇ１６，ｍｅｍ１６，ｉｍｍ１６

② 命令語形式



③ バイト数

４／５／６

④ クロック数

１８：奇数アドレス

１６：偶数アドレス

⑤ １６ビット・ワード転送回数

1

⑥ 機　能

ｒｅｇ１６←（ｍｅｍ１６）×ｉｍｍ１６

積≦１６ビット：ＣＹ←０，Ｖ←０

積＞１６ビット：ＣＹ←１，Ｖ←１

第２オペランドでアドレスされる１６ビット・メモリと第３オペランドの１６ビット・イミーディエト・データとの符号付き乗算を行い，結果を第１オペランドで指定される１６ビット・レジスタにストアします．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	U	U	U	U	×

⑧ 記述例

10．11　除算命令

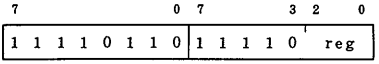
10．11．1　DIVU(Divide Unsigned)

(1)　8ビット・レジスタ

①　記述形式

DIVU　reg8

②　命令語形式



③　バイト数

2

④　クロック数

11

⑤　16ビット・ワード転送回数

なし

⑥　機　能

temp←AW

temp÷reg8≦FFHのとき

AH←temp％reg8

AL←temp÷reg8

temp÷reg8＞FFHのとき

TA←(001H,000H)

TC←(003H,002H)

SP←SP−2,(SP+1,SP)←PSW

IE←0,BRK←0

SP←SP−2,(SP+1,SP)←PS

PS←TC

SP←SP−2,(SP+1,SP)←PC

PC←TA

16ビット・レジスタAWをオペランドで指定される8ビット・レジスタで割り(符号なし除算)，商をレジスタALにストアし，剰余をレジスタAHにストアします．商がデスティネーション・レジスタALの容量FFHを越えた場合は，ベクタ0割り込みが発生され，商と剰余は不定となります．このことは，特に0で割ったときに起こります．このときスタックに退避するアドレスは，この命令の先頭アドレスです．

整数でない商は整数に切り縮められます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	U	U	U

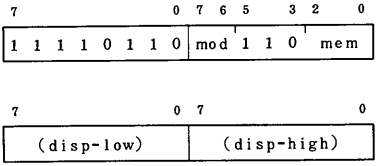
⑧ 記述例

⑵ 8ビット・メモリ

① 記述形式

DIVU mem8

② 命令語形式



③ バイト数

2／3／4

④ クロック数

15

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

```
temp←AW
  temp÷(mem8)≦FFHのとき
    AH←temp%(mem8)
    AL←temp÷(mem8)

  temp÷(mem8)＞FFHのとき
      TA←(001H,000H)
      TC←(003H,002H)
      SP←SP−2,(SP+1,SP)←PSW
      IE←0,BRK←0
      SP←SP−2,(SP+1,SP)←PS
      PS←TC
      SP←SP−2,(SP+1,SP)←PC
      PC←TA
```

16ビット・レジスタAWをオペランドで指定される8ビット・メモリで割り(符号なし除算)，商をレジスタALにストアし，剰余をレジスタAHにストアします．商がデスティネーション・レジスタALの容量FFHを越えた場合は，ベクタ0割り込みが発生され，商と剰余は不定となります．このことは，特に0で割ったときに起こります．このときスタックに退避するアドレスは，この命令の先頭アドレスです．

整数でない商は整数に切り縮められます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	U	U	U

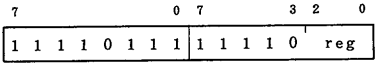
⑧ 記述例

(3) 16ビット・レジスタ

① 記述形式

DIVU reg16

② 命令語形式



③ バイト数

2

④ クロック数

19

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

temp←DW,AW

temp÷reg16≦FFFFHのとき

DW←temp％reg16

AW←temp÷reg16

temp÷reg16＞FFFFHのとき

TA←(001H,000H)

TC←(003H,002H)

SP←SP−2,(SP+1,SP)←PSW

IE←0,BRK←0

SP←SP−2,(SP+1,SP)←PS

PS←TC

SP←SP−2,(SP+1,SP)←PC

PC←TA

16ビット・レジスタDWとAWをオペランドで指定される16ビット・レジスタで割り(符号なし除算),商をAWにストアし,剰余をDWにストアします.商がデスティネーション・レジスタAWの容量FFFFHを越えた場合は,ベクタ0割り込みが発生され,商と剰余は不定となります,このことは,特に0で割ったときに起こります.このときスタックに退避するアドレスは,この命令の先頭アドレスです.

整数でない商は整数に切り縮められます.

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	U	U	U

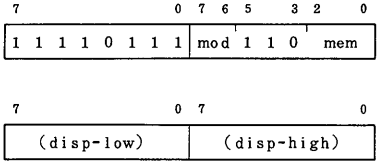
⑧ 記述例

(4) 16ビット・メモリ

① 記述形式

DIVU mem16

② 命令語形式



③ バイト数

2／3／4

④ クロック数

25：奇数アドレス

23：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

temp←DW,AW

temp÷(mem16)≦FFFFHのとき

DW←temp%(mem16)

AW←temp÷(mem16)

temp÷(mem16)＞FFFFHのとき

TA←(001H,000H)

TC←(003H,002H)

SP←SP−2,(SP+1,SP)←PSW

IE←0,BRK←0

SP←SP−2,(SP+1,SP)←PS

PS←TC

SP←SP−2,(SP+1,SP)←PC

PC←TA

16ビット・レジスタDWとAWをオペランドで指定される16ビット・メモリで割り(符号なし除算)，商をAWにストアし，剰余をDWにストアします．商がデスティネーション・レジスタAWの容量FFFFHを越えた場合は，ベクタ0割り込みが発生され，商

と剰余は不定となります．このことは，特に０で割ったときに起こります．このときスタックに退避するアドレスは，この命令の先頭アドレスです．

整数でない商は整数に切り縮められます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	U	U	U

⑧ 記述例

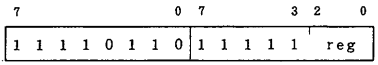
10．11．2　DIV(Divide Signed)

(1)　8ビット・レジスタ

① 記述形式

DIV　reg8

② 命令語形式



③ バイト数

2

④ クロック数

17

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

temp←AW

temp÷reg8＞0でtemp÷reg8≦7FHまたは

temp÷reg8＜0でtemp÷reg8＞0−7FH−1のとき

AH←temp％reg8

AL←temp÷reg8

temp÷reg8＞0でtemp÷reg8＞7FHまたは

temp÷reg8＜0でtemp÷reg8≦0−7FH−1のとき

商と剰余は不定

TA←(001H,000H)

TC←(003H,002H)

SP←SP−2,(SP+1,SP)←PSW

IE←0,BRK←0

SP←SP−2,(SP+1,SP)←PS

PS←TC

SP←SP−2,(SP+1,SP)←PC

PC←TA

16ビット・レジスタAWをオペランドで指定される8ビット・レジスタで割り(符号付き除算)，商を8ビット・レジスタALにストアし，剰余をレジスタAHにストアします．正の商の最大値は＋127(7FH)で，負の商の最小値は−128(80H)です．商が正で最大値を越えたとき，または商が負で最小値より小さくなったときは，商と剰余は不定

となり，ベクタ0割り込みが発生されます．このことは特に0で割ったときに起こります．

このときスタックに退避するアドレスは，この命令の先頭アドレスです．

整数でない商は整数に切り縮められ，剰余は被除数と同じ符号を持ちます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	U	U	U

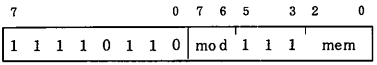
⑧ 記述例

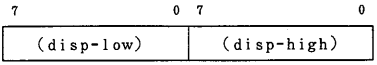
(2) 8ビット・メモリ

① 記述形式

DIV mem8

② 命令語形式





③ バイト数

2／3／4

④ クロック数

20

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

temp←AW

temp÷(mem8)＞0でtemp÷(mem8)≦7FHまたは

temp÷(mem8)＜0でtemp÷(mem8)＞0－7FH－1のとき

AH←temp％(mem8)

AL←temp÷(mem8)

temp÷(mem8)＞0でtemp÷(mem8)＞7FHまたは

temp÷(mem8)＜0でtemp÷(mem8)≦0－7FH－1のとき

商と剰余は不定

TA←(001H,000H)

TC←(003H,002H)

SP←SP－2,(SP+1,SP)←PSW

IE←0,BRK←0

SP←SP－2,(SP+1,SP)←PS

PS←TC

SP←SP－2,(SP+1,SP)←PC

PC←TA

16ビット・レジスタAWをオペランドで指定される8ビット・メモリで割り(符号付き除算),商を8ビット・レジスタALにストアし,剰余をレジスタAHにストアします.

正の商の最大値は＋１２７（７ＦＨ）で，負の商の最小値は－１２８（８０Ｈ）です．商が正で最大値を越えたとき，または商が負で最小値より小さくなったときは，商と剰余は不定となり，ベクタ０割り込みが発生されます．このことは特に０で割ったときに起こります．このときスタックに退避するアドレスは，この命令の先頭アドレスです．

整数でない商は整数に切り縮められ，剰余は被除数と同じ符号を持ちます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	U	U	U

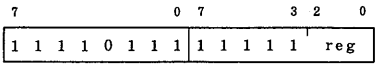
⑧ 記述例

(3) 16ビット・レジスタ

① 記述形式

DIV reg16

② 命令語形式



③ バイト数

2

④ クロック数

24

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

temp←DW,AW

temp÷reg16＞0でtemp÷reg16≦7FFFHまたは

temp÷reg16＜0でtemp÷reg16＞0−7FFFH−1のとき

DW←temp％reg16

AW←temp÷reg16

temp÷reg16＞0でtemp÷reg16＞7FFFHまたは

temp÷reg16＜0でtemp÷reg16≦0−7FFFH−1のとき

商と剰余は不定

TA←(001H,000H)

TC←(003H,002H)

SP←SP−2,(SP+1,SP)←PSW

IE←0,BRK←0

SP←SP−2,(SP+1,SP)←PS

PS←TC

SP←SP−2,(SP+1,SP)←PC

PC←TA

16ビット・レジスタDWとAWをオペランドで指定される16ビット・レジスタで割り(符号付き除算),商を16ビット・レジスタAWにストアし,剰余を16ビット・レジスタDWにストアします.正の商の最大値は+32,767(7FFFH)で,負の商の最小値は−32,768(8000H)です. 商が正で最大値を越えたとき,または商が負で最小値より小さくなったときは,商と剰余は不定となり,ベクタ0割り込みが発生されます.この

ことは特に0で割ったときに起こります．このときスタックに退避するアドレスは，この命令の先頭アドレスです．

整数でない商は整数に切り縮められ，剰余は被除数と同じ符号を持ちます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	U	U	U

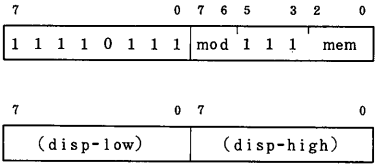
⑧ 記述例

(4) 16ビット・メモリ

① 記述形式

DIV mem16

② 命令語形式



③ バイト数

2／3／4

④ クロック数

30：奇数アドレス

28：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

temp←DW,AW

temp÷(mem16)＞0でtemp÷(mem16)≦7FFFHまたは

temp÷(mem16)＜0でtemp÷(mem16)＞0−7FFFH−1のとき

DW←temp％(mem16)

AW←temp÷(mem16)

temp÷(mem16)＞0でtemp÷(mem16)＞7FFFHまたは

temp÷(mem16)＜0でtemp÷(mem16)≦0−7FFFH−1のとき

商と剰余は不定

TA←(001H,000H)

TC←(003H,002H)

SP←SP−2,(SP+1,SP)←PSW

IE←0,BRK←0

SP←SP−2,(SP+1,SP)←PS

PS←TC

SP←SP−2,(SP+1,SP)←PC

PC←TA

１６ビット・レジスタＤＷとＡＷをオペランドで指定される１６ビット・メモリで割り（符号付き除算），商を１６ビット・レジスタＡＷにストアし，剰余を１６ビット・レジスタＤＷにストアします．正の商の最大値は＋３２,７６７（７ＦＦＨ）で，負の商の最小値は－３２,７６８（８０００Ｈ）です．商が正で最大値を越えたとき，または商が負で最小値より小さくなったときは，商と剰余は不定となり，ベクタ０割り込みが発生されます．このことは特に０で割ったときに起こります．このときスタックに退避するアドレスは，この命令の先頭アドレスです．

整数でない商は整数に切り縮められ，剰余は被除数と同じ符号を持ちます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	U	U	U

⑧ 記述例

10．12　BCD補正命令

10．12．1　ADJBA(Adjust Byte Add)

① 記述形式

ADJBA（オペランドなし）

② 命令語形式

7							0
0	0	1	1	0	1	1	1

③ バイト数

1

④ クロック数

4

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

AL∧0FH>9またはAC＝1のとき

AL←AL＋6

AH←AH＋1

AC←1

CY←AC

AL←AL∧0FH

2つのアンパック10進数の加算のAL内の結果を1つのアンパック10進数に補正します．上位4ビットはゼロになります．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	×	U	×

⑧ 記述例

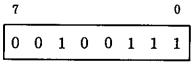
ADJBA

10．12．2　ADJ4A（Adjust Nibble Add）

① 記述形式

ADJ4A（オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

AL∧0FH＞9またはAC＝1のとき

　AL←AL＋6

　AC←1

AL＞99HまたはCY＝1のとき

　AL←AL＋60H

　CY←1

2つのパック10進数の加算のAL内の結果を1つのパック10進数に補正します．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	×	×	×

⑧ 記述例

ADJ4A

10．12．3　ADJBS(Adjust Byte Subtract)

① 記述形式

ADJBS（オペランドなし）

② 命令語形式

7							0
0	0	1	1	1	1	1	1

③ バイト数

1

④ クロック数

4

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

AL∧０FH＞９またはAC＝１のとき

AL←AL－６

AH←AH－１

AC←１

CY←AC

AL←AL∧０FH

２つのアンパック１０進数の減算のＡＬ内の結果を１つのアンパック１０進数に補正します．上位４ビットはゼロになります．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	U	U	×	U	×

⑧ 記述例

ADJBS

10．12．4　ADJ4S(Adjust Nibble Subtract)

① 記述形式

ADJ4S(オペランドなし)

② 命令語形式

7							0
0	0	1	0	1	1	1	1

③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

AL∧0FH＞9またはAC＝1のとき

AL←AL－6

AC←1

AL＞99HまたはCY＝1のとき

AL←AL－60H

CY←1

2つのパック10進数の減算のAL内の結果を1つのパック10進数に補正します.

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	×	×	×

⑧ 記述例

ADJ4S

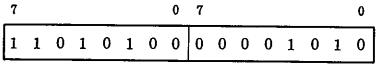
10．13　データ変換命令

10．13．1　CVTBD(Convert Binary to Decimal)

① 記述形式

CVTBD(オペランドなし)

② 命令語形式



③ バイト数

2

④ クロック数

12

⑤ 1.6ビット・ワード転送回数

なし

⑥ 機　能

AH←AL÷0AH

AL←AL％0AH

レジスタALの2進数8ビットを，2けたのアンパック10進数に変換します．

レジスタAHはレジスタALを10で割った商で置き換えられ，次にレジスタALがその乗算の剰余で置き換えられます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	U

⑧ 記述例

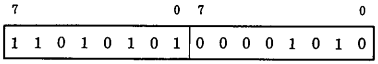
CVTBD

10．13．2　CVTDB(Convert Decimal to Binary)

① 記述形式

CVTDB(オペランドなし)

② 命令語形式



③ バイト数

2

④ クロック数

8

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

AL←AH×0AH+AL

AH←0

レジスタAHおよびALの2けたのアンパック10進数を，16ビットの2進数に変換します．

レジスタALは，AHに10を掛けた結果にALを加えたもので置き換えられ，レジスタAHは0で置き換えられます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	U

⑧ 記述例

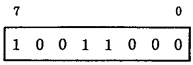
CVTDB

10．13．3　CVTBW(Convert Byte to Word)

① 記述形式

CVTBW(オペランドなし)

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

AL＜80Hのとき　　AH←0

AL≧80Hのとき　　AH←FFH

レジスタAL内のバイトの符号をレジスタAHに拡張します．バイト除算を実行する前に，あるバイトから倍長(ワード)の被除数を得るのに有効です．

⑦ フラグの動作

なし

⑧ 記述例

CVTBW

10．13．4　CVTWL（Convert Word to Long Word）

① 記述形式

CVTWL（オペランドなし）

② 命令語形式

7							0
1	0	0	1	1	0	0	1

③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

AW＜8000Hのとき　　DW←0

AW≧8000Hのとき　　DW←FFFFH

レジスタAW内のワードの符号をレジスタDWに拡張します．ワード除算を実行する前に，あるワードから倍長（ダブルワード）の被除数を得るのに有効です．

⑦ フラグの動作

なし

⑧ 記述例

CVTWL

10．14　比較命令

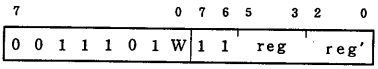
10．14．1　CMP（Compare）

(1) レジスタとレジスタ

① 記述形式

CMP　reg，reg′

② 命令語形式



③ バイト数

2

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

reg－reg′

第1オペランドで指定される8／16ビット・レジスタの内容から第2オペランドで指定される8／16ビット・レジスタの内容を減算します．結果はストアせず，フラグにのみ影響を与えます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

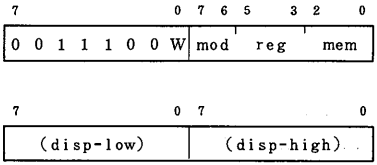
⑧ 記述例

(2) メモリとレジスタ

① 記述形式

CMP mem,reg

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W=0のとき 6

W=1のとき 8：奇数アドレス

6：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

(mem)－reg

第1オペランドでアドレスされる8／16ビット・メモリの内容から第2オペランドで指定される8／16ビット・レジスタの内容を減算します．結果はストアせず，フラグにのみ影響を与えます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

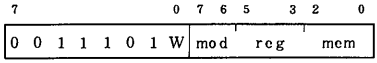
⑧ 記述例

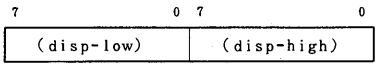
⑶ レジスタとメモリ

① 記述形式

CMP reg,mem

② 命令語形式





③ バイト数

2／3／4

④ クロック数

W＝0のとき 6

W＝1のとき 8：奇数アドレス

6：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

reg−(mem)

第1オペランドで指定される8／16ビット・レジスタの内容から第2オペランドでアドレスされる8／16ビット・メモリの内容を減算します．結果はストアせず，フラグにのみ影響を与えます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

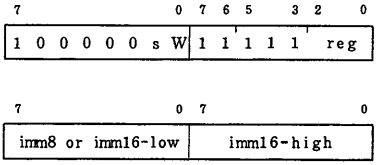
⑧ 記述例

(4) レジスタとイミーディエト・データ

① 記述形式

CMP reg，imm

② 命令語形式



③ バイト数

3／4

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg－imm

第1オペランドで指定される8／16ビット・レジスタの内容から第2オペランドで指定される8／16ビット・イミーディエト・データの内容を減算します．結果はストアせず，フラグにのみ影響を与えます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

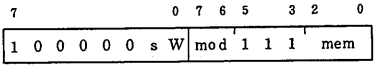
⑧ 記述例

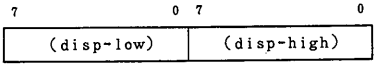
(5) メモリとイミーディエト・データ

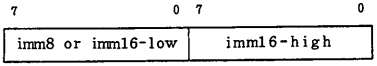
① 記述形式

CMP mem, imm

② 命令語形式







③ バイト数

3／4／5／6

④ クロック数

W＝0のとき 6

W＝1のとき 8：奇数アドレス

6：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

(mem)－imm

第1オペランドでアドレスされる8／16ビット・メモリの内容から第2オペランドで指定される8／16ビット・イミーディエト・データを減算します．結果はストアせず，フラグにのみ影響を与えます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

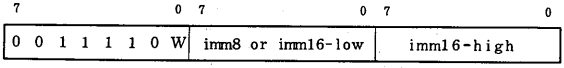
⑧ 記述例

(6) アキュームレータとイミーディエト・データ

① 記述形式

CMP acc, imm

② 命令語形式



③ バイト数

2／3

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

W＝0のとき　AL－imm8

W＝1のとき　AW－imm16

第1オペランドで指定されるアキュームレータ（AL／AW）の内容から第2オペランドで指定される8／16ビット・イミーディエト・データを減算します．結果はストアせず，フラグにのみ影響を与えます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	×

⑧ 記述例

10．15　補数演算命令

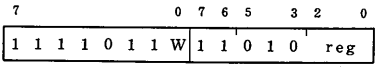
10．15．1　NOT(Not)

(1) レジスタ

① 記述形式

NOT reg

② 命令語形式



③ バイト数

2

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

reg←$\overline{\text{reg}}$

オペランドで指定される8／16ビット・レジスタの各ビットを反転し(1の補数をとる)，結果を指定されたレジスタにストアします．

⑦ フラグの動作

なし

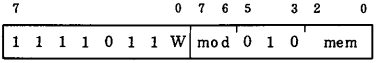
⑧ 記述例

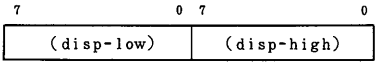
(2) メモリ

① 記述形式

NOT mem

② 命令語形式





③ バイト数

2／3／4

④ クロック数

W＝0のとき　7

W＝1のとき　11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能

(mem)←$\overline{(mem)}$

オペランドでアドレスされる8／16ビット・メモリの各ビットを反転し(・1の補数をとる)，結果をアドレスされたメモリにストアします．

⑦ フラグの動作

なし

⑧ 記述例

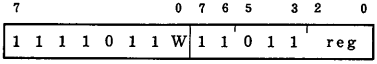
10．15．2　NEG(Negate)

(1) レジスタ

① 記述形式

NEG　reg

② 命令語形式



③ バイト数

2

④ クロック数

2

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

reg←$\overline{\text{reg}}$＋1

オペランドで指定される８／１６ビット・レジスタの２の補数をとります.

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	1*

＊：実行前のregが0のときは0

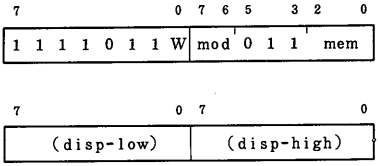
⑧ 記述例

(2) メモリ

① 記述形式

NEG mem

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　7

W＝1のとき　11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能

(mem)←$\overline{(mem)}$＋1

オペランドでアドレスされる8／16ビット・メモリの2の補数をとります.

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	×	×	1*

＊：実行前の(mem)が0のときは0

⑧ 記述例

10．16　論理演算命令

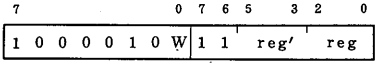
10．16．1　TEST (Test)

(1) レジスタとレジスタ

① 記述形式

TEST reg,reg'

② 命令語形式



③ バイト数

2

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

reg∧reg'

第1オペランドで指定される8／16ビット・レジスタと第2オペランドで指定される8／16ビット・レジスタとの論理積をとります．結果はストアされず，フラグのみが影響を受けます．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

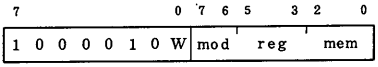
⑧ 記述例

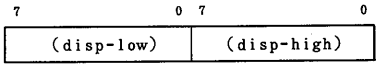
(2) メモリとレジスタ

① 記述形式

TEST mem,reg or TEST reg,mem

② 命令語形式





③ バイト数

2／3／4

④ クロック数

W＝0のとき 6

W＝1のとき 8：奇数アドレス

6：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

(mem)∧reg

第1オペランドで指定される8／16ビット・レジスタと第2オペランドでアドレスされる8／16ビット・メモリとの論理積をとるか，または，第1オペランドでアドレスされる8／16ビット・メモリと第2オペランドで指定される8／16ビット・レジスタとの論理積をとります．結果はストアされず，フラグのみが影響を受けます．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

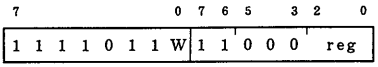
⑧ 記述例

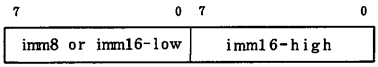
(3) レジスタとイミーディエト・データ

① 記述形式

TEST reg, imm

② 命令語形式





③ バイト数

3／4

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg ∧ imm

第1オペランドで指定される8／16ビット・レジスタと第2オペランドで指定される8／16ビット・イミーディエト・データとの論理積をとります．結果はストアされず，フラグのみが影響を受けます．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

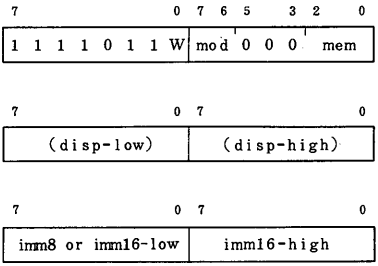
⑧ 記述例

(4) メモリとイミーディエト・データ

① 記述形式

TEST mem, imm

② 命令語形式



③ バイト数

3／4／5／6

④ クロック数

W＝0のとき 6

W＝1のとき 8：奇数アドレス

6：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

(mem) ∧ imm

第1オペランドでアドレスされる8／16ビット・メモリと第2オペランドで指定される8／16ビット・イミーディエト・データとの論理積をとります．結果はストアされず，フラグのみが影響を受けます．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

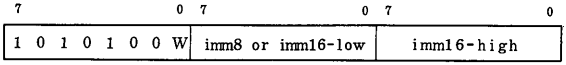
⑧ 記述例

(5) アキュームレータとイミーディエト・データ

① 記述形式

TEST acc, imm

② 命令語形式



③ バイト数

2／3

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

W＝0のとき AL∧imm8

W＝1のとき AW∧imm16

第1オペランドで指定されるアキュームレータ（AL／AW）と第2オペランドで指定される8／16ビット・イミーディエト・データとの論理積をとります．結果はストアされず，フラグのみが影響を受けます．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

⑧ 記述例

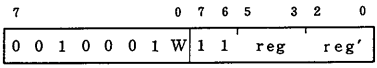
10．16．2　AND（And）

(1)　レジスタとレジスタをレジスタへ

①　記述形式

AND　reg，reg′

②　命令語形式



③　バイト数

2

④　クロック数

2

⑤　１６ビット・ワード転送回数

なし

⑥　機　能

reg←reg∧reg′

第１オペランドで指定される８／１６ビット・レジスタと第２オペランドで指定される８／１６ビット・レジスタとの論理積をとり，結果を第１オペランドで指定されたレジスタにストアします．ＣＹとＶフラグはクリアされ，ＡＣフラグは不定となります．

⑦　フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

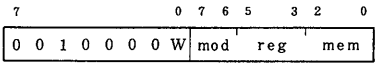
⑧　記述例

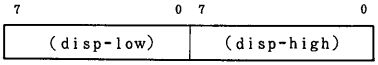
(2) メモリとレジスタをメモリへ

① 記述形式

AND mem,reg

② 命令語形式





③ バイト数

2／3／4

④ クロック数

W＝0のとき　7

W＝1のとき　11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能

(mem)←(mem)∧reg

第1オペランドでアドレスされる8／16ビット・メモリと第2オペランドで指定される8／16ビット・レジスタとの論理積をとり，結果を第1オペランドでアドレスされたメモリにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

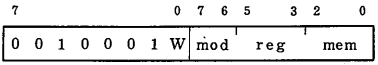
⑧ 記述例

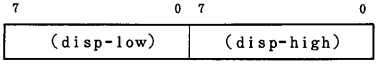
(3) レジスタとメモリをレジスタへ

① 記述形式

AND reg,mem

② 命令語形式





③ バイト数

2／3／4

④ クロック数

W＝0のとき　6

W＝1のとき　8：奇数アドレス

6：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機　能

reg←reg∧(mem)

第1オペランドで指定される8／16ビット・レジスタと第2オペランドでアドレスされる8／16ビット・メモリとの論理積をとり，結果を第1オペランドで指定されたレジスタにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

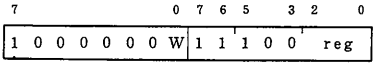
⑧ 記述例

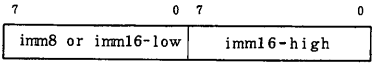
(4) レジスタとイミーディエト・データをレジスタへ

① 記述形式

AND reg，imm

② 命令語形式





③ バイト数

3／4

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg←reg∧imm

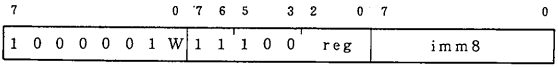
第1オペランドで指定される8／16ビット・レジスタと第2オペランドで指定される8／16ビット・イミーディエト・データとの論理積をとり，結果を第1オペランドで指定されたレジスタにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

⑧ 記述例

備考 アセンブラ，コンパイラによっては，次に示すような未定義のコードが生成されることがあります． ★



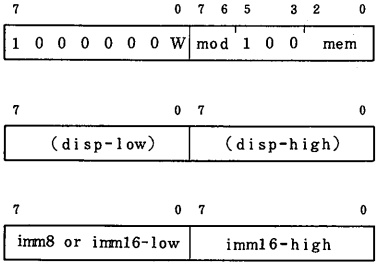
このような場合でも正常に動作します．ただし，エミュレータによっては，この未定義命令に対する逆アセンブル機能，ライン・アセンブル機能がサポートされていない場合があるので注意してください．

(5) メモリとイミーディエト・データをメモリへ

① 記述形式

AND mem, imm

② 命令語形式



③ バイト数

3／4／5／6

④ クロック数

W＝0のとき　7

W＝1のとき　11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能

(mem)←(mem)∧imm

第1オペランドでアドレスされる8／16ビット・メモリと第2オペランドで指定される8／16ビット・イミーディエト・データとの論理積をとり，結果を第1オペランドでアドレスされたメモリにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

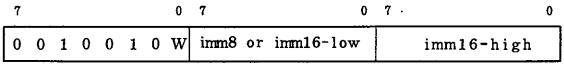
⑧ 記述例

(6) アキュームレータとイミーディエト・データをアキュームレータへ

① 記述形式

AND acc,imm

② 命令語形式



③ バイト数

2／3

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

W＝0のとき AL←AL∧imm8

W＝1のとき AW←AW∧imm16

第1オペランドで指定されるアキュームレータ（AL／AW）と第2オペランドで指定される8／16ビット・イミーディエト・データとの論理積をとり，結果を第1オペランドで指定されたアキュームレータにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

⑧ 記述例

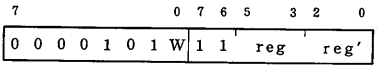
10．16．3　OR(Or)

(1) レジスタとレジスタをレジスタへ

① 記述形式

OR reg,reg'

② 命令語形式



③ バイト数

2

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

reg←reg∨reg'

第1オペランドで指定される8／16ビット・レジスタと第2オペランドで指定される8／16ビット・レジスタとの論理和をとり，結果を第1オペランドで指定されたレジスタにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

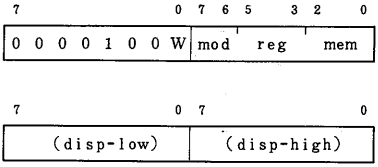
⑧ 記述例

(2) メモリとレジスタをメモリへ

① 記述形式

OR mem,reg

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　7

W＝1のとき　11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能

(mem)←(mem)∨reg

第1オペランドでアドレスされる8／16ビット・メモリと第2オペランドで指定される8／16ビット・レジスタとの論理和をとり，結果を第1オペランドでアドレスされるメモリにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

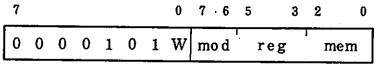
⑧ 記述例

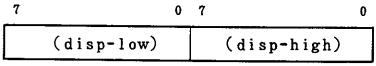
(3) レジスタとメモリをレジスタへ

① 記述形式

OR reg,mem

② 命令語形式





③ バイト数

2／3／4

④ クロック数

W＝0のとき 6

W＝1のとき 8：奇数アドレス

6：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

reg← reg∨(mem)

第1オペランドで指定される8／16ビット・レジスタと第2オペランドでアドレスされる8／16ビット・メモリとの論理和をとり，結果を第1オペランドで指定されたレジスタにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

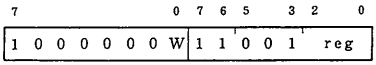
⑧ 記述例

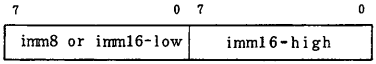
(4) レジスタとイミーディエト・データをレジスタへ

① 記述形式

OR　reg，imm

② 命令語形式





③ バイト数

3／4

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

reg←reg∨imm

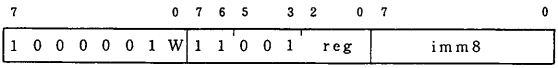
第1オペランドで指定される8／16ビット・レジスタと第2オペランドで指定される8／16ビット・イミーディエト・データとの論理和をとり，結果を第1オペランドで指定されたレジスタにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

⑧ 記述例

備考　アセンブラ，コンパイラによっては，次に示すような未定義のコードが生成されることがあります． ★



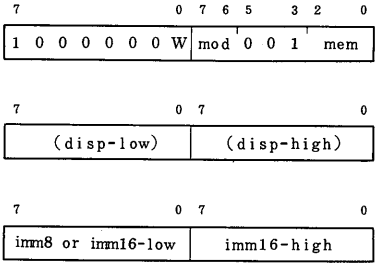
このような場合でも正常に動作します．ただし，エミュレータによっては，この未定義命令に対する逆アセンブル機能，ライン・アセンブル機能がサポートされていない場合があるので注意してください．

(5) メモリとイミーディエト・データをメモリへ

① 記述形式

OR mem, imm

② 命令語形式



③ バイト数

3／4／5／6

④ クロック数

W＝0のとき　7

W＝1のとき　11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能

(mem)←(mem)∨imm

第1オペランドでアドレスされる8／16ビット・メモリと第2オペランドで指定される8／16ビット・イミーディエト・データとの論理和をとり，結果を第1オペランドでアドレスされたメモリにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

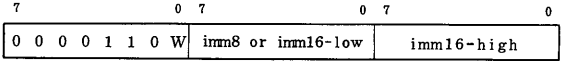
⑧ 記述例

(6) アキュームレータとイミーディエト・データをアキュームレータへ

① 記述形式

OR acc,imm

② 命令語形式



③ バイト数

2／3

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

W＝0のとき AL←AL∨imm8

W＝1のとき AW←AW∨imm16

第1オペランドで指定されるアキュームレータ（AL／AW）と第2オペランドで指定される8／16ビット・イミーディエト・データとの論理和をとり，結果を第1オペランドで指定されたアキュームレータにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

⑧ 記述例

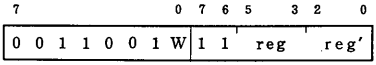
10．16．4　XOR（Exclusive Or）

(1) レジスタとレジスタをレジスタへ

① 記述形式

XOR reg,reg′

② 命令語形式



③ バイト数

2

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

reg←reg∀reg′

第1オペランドで指定される8／16ビット・レジスタと第2オペランドで指定される8／16ビット・レジスタとの排他的論理和をとり，結果を第1オペランドで指定されるレジスタにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

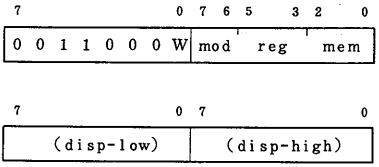
⑧ 記述例

(2) メモリとレジスタをメモリへ

① 記述形式

XOR mem,reg

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　7

W＝1のとき　11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能

(mem)←(mem)∀reg

第1オペランドでアドレスされる8／16ビット・メモリと第2オペランドで指定される8／16ビット・レジスタとの排他的論理和をとり，結果を第1オペランドでアドレスされるメモリにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

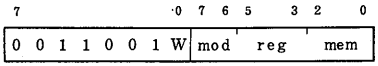
⑧ 記述例

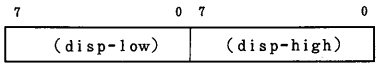
(3) レジスタとメモリをレジスタへ

① 記述形式

XOR reg,mem

② 命令語形式





③ バイト数

2／3／4

④ クロック数

W＝0のとき 6

W＝1のとき 8：奇数アドレス

6：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

reg←reg∀(mem)

第1オペランドで指定される8／16ビット・レジスタと第2オペランドでアドレスされる8／16ビット・メモリとの排他的論理和をとり，結果を第1オペランドで指定されるレジスタにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

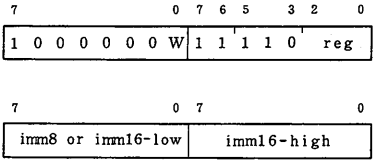
⑧ 記述例

(4) レジスタとイミーディエト・データをレジスタへ

① 記述形式

XOR reg，imm

② 命令語形式



③ バイト数

3／4

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

reg← reg ∀ imm

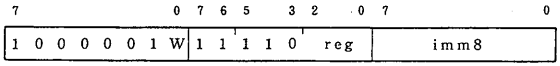
第1オペランドで指定される8／16ビット・レジスタと第2オペランドで指定される8／16ビット・イミーディエト・データとの排他的論理和をとり，結果を第1オペランドで指定されたレジスタにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

⑧ 記述例

備考　アセンブラ，コンパイラによっては，次に示すような未定義のコードが生成されることがあります． ★



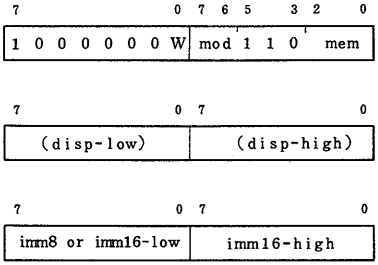
このような場合でも正常に動作します．ただし，エミュレータによっては，この未定義命令に対する逆アセンブル機能，ライン・アセンブル機能がサポートされていない場合があるので注意してください．

(5) メモリとイミーディエト・データをメモリへ

① 記述形式

XOR mem, imm

② 命令語形式



③ バイト数

3／4／5／6

④ クロック数

W＝0のとき 7

W＝1のとき 11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

(mem)←(mem) ∀ imm

第1オペランドでアドレスされる8／16ビット・メモリと第2オペランドで指定される8／16ビット・イミーディエト・データとの排他的論理和をとり，結果を第1オペランドでアドレスされるメモリにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

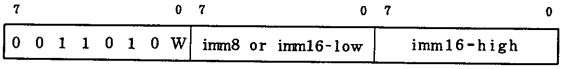
⑧ 記述例

⑥ アキュームレータとイミーディエト・データをアキュームレータへ

① 記述形式

XOR acc,imm

② 命令語形式



③ バイト数

2／3

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

W＝0のとき AL←AL∀imm8

W＝1のとき AW←AW∀imm16

第1オペランドで指定されるアキュームレータ（AL／AW）と第2オペランドで指定される8／16ビット・イミーディエト・データとの排他的論理和をとり，結果を第1オペランドで指定されるアキュームレータにストアします．CYとVフラグはクリアされ，ACフラグは不定となります．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	0

⑧ 記述例

10．17　ビット操作命令

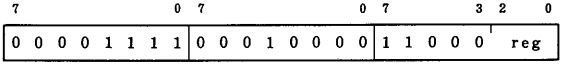
10．17．1　TEST1(Test Bit)

(1)　8ビット・レジスタのビットCL

①　記述形式

TEST1　reg8,CL

②　命令語形式



③　バイト数

3

④　クロック数

4

⑤　16ビット・ワード転送回数

なし

⑥　機　能

reg8のビットNO.CL＝0のとき　Z←1

reg8のビットNO.CL＝1のとき　Z←0

第1オペランドで指定される8ビット・レジスタのCLレジスタで指定されるビットが0ならばZフラグがセット(1)され，1ならばリセット(0)されます．CLの値は下位3ビット(0−7)のみ有効です．

⑦　フラグの動作

V	S	Z	AC	P	CY
0	U	×	U	U	0

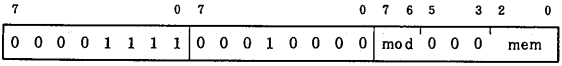
⑧　記述例

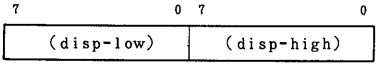
(2) 8ビット・メモリのビットCL

① 記述形式

TEST1 mem8,CL

② 命令語形式





③ バイト数

3／4／5

④ クロック数

8

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

(mem8)のビットNO.CL＝0のとき Z←1

(mem8)のビットNO.CL＝1のとき Z←0

第1オペランドで指定される8ビット・メモリのCLレジスタで指定されるビットが0ならばZフラグがセット(1)され，1ならばリセット(0)されます．CLの値は下位3ビット(0－7)のみ有効です．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	U	×	U	U	0

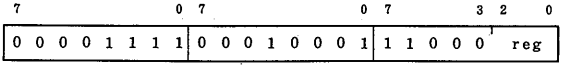
⑧ 記述例

⑶ １６ビット・レジスタのビットＣＬ

① 記述形式

TEST1 reg16,CL

② 命令語形式



③ バイト数

3

④ クロック数

4

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

reg16のビットNO.CL＝０のとき　Z←1

reg16のビットNO.CL＝１のとき　Z←0

第１オペランドで指定される１６ビット・レジスタのＣＬレジスタで指定されるビットが０ならばＺフラグがセット(1)され，１ならばリセット(０)されます．ＣＬの値は下位４ビット(０−１５)のみ有効です．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	U	×	U	U	0

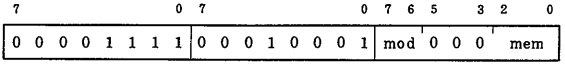
⑧ 記述例

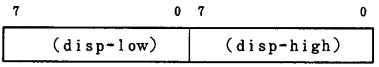
(4) １６ビット・メモリのビットＣＬ

① 記述形式

TEST1 mem16,CL

② 命令語形式





③ バイト数

３／４／５

④ クロック数

１０：奇数アドレス

８：偶数アドレス

⑤ １６ビット・ワード転送回数

1

⑥ 機　能

（mem16）のビットNO.CL＝０のとき　Z←1

（mem16）のビットNO.CL＝１のとき　Z←0

第１オペランドで指定される１６ビット・メモリのＣＬレジスタで指定されるビットが０ならばＺフラグがセット(1)され，１ならばリセット(0)されます．ＣＬの値は下位４ビット（0－15）のみ有効です．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	U	×	U	U	0

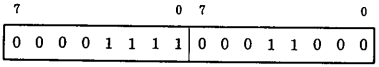
⑧ 記述例

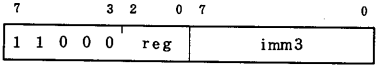
(5) 8ビット・レジスタのビットimm3

① 記述形式

TEST1 reg8,imm3

② 命令語形式





③ バイト数

4

④ クロック数

4

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg8のビットNO.imm3＝0のとき Z←1

reg8のビットNO.imm3＝1のとき Z←0

第1オペランドで指定される8ビット・レジスタの第2オペランドで指定される3ビット・イミーディエト・データで指定されるビットが0ならばZフラグがセット(1)され，1ならばリセット(0)されます．命令の4バイト目のイミーディエト・データは下位3ビットのみ有効です．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	U	×	U	U	0

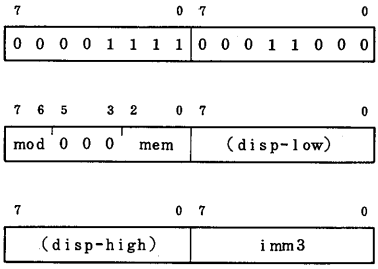
⑧ 記述例

⑥ 8ビット・メモリのビット imm3

① 記述形式

TEST1 mem8, imm3

② 命令語形式



③ バイト数

4／5／6

④ クロック数

8

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

(mem8)のビットNO. imm3＝0のとき Z←1

(mem8)のビットNO. imm3＝1のとき Z←0

第1オペランドでアドレスされる8ビット・メモリの第2オペランドで指定された3ビット・イミーディエト・データで指定されるビットが0ならばZフラグがセット(1)され，1ならばリセット(0)されます．命令の最終バイトのイミーディエト・データは下位3ビットのみ有効です．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	U	×	U	U	0

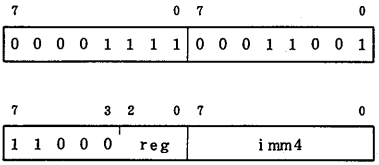
⑧ 記述例

(7) 16ビット レジスタのビットimm4

① 記述形式

TEST1 reg16,imm4

② 命令語形式



③ バイト数

4

④ クロック数

4

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg16のビットNO.imm4＝0のとき Z←1

reg16のビットNO.imm4＝1のとき Z←0

第1オペランドで指定される16ビット・レジスタの，第2オペランドで指定される4ビット・イミーディエト・データで指定されるビットが0ならばZフラグがセット(1)され，1ならばリセット(0)されます．命令の4バイト目のイミーディエト・データは下位4ビットのみ有効です．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	U	×	U	U	0

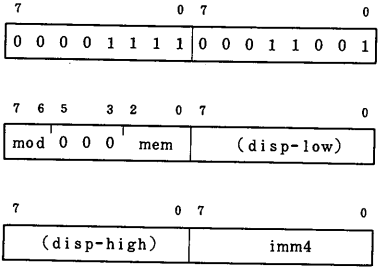
⑧ 記述例

(8) 16ビット・メモリのビット imm4

① 記述形式

TEST1 mem16, imm4

② 命令語形式



③ バイト数

4／5／6

④ クロック数

10：奇数アドレス

8：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

(mem16)のビットNO. imm4＝0のとき Z←1

(mem16)のビットNO. imm4＝1のとき Z←0

第1オペランドでアドレスされる16ビット・メモリの，第2オペランドで指定された4ビット・イミーディエト・データで指定されるビットが0ならばZフラグがセット(1)され，1ならばリセット(0)されます．命令の最終バイトのイミーディエト・データは，下位4ビットのみ有効です．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	U	×	U	U	0

⑧ 記述例

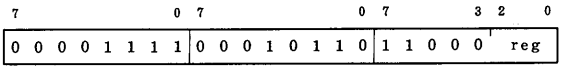
10．17．2　NOT1(Not Bit)

(1)　8ビット・レジスタのビットＣＬ

①　記述形式

NOT1　reg8,CL

②　命令語形式



③　バイト数

3

④　クロック数

4

⑤　１６ビット・ワード転送回数

なし

⑥　機　能

reg8のビットNO.CL ← $\overline{\text{reg8のビットNO.CL}}$

第１オペランドで指定される８ビット・レジスタの，ＣＬレジスタで指定されるビットが反転されます．ＣＬの値は下位３ビットのみ有効です．

⑦　フラグの動作

なし

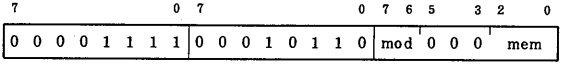
⑧　記述例

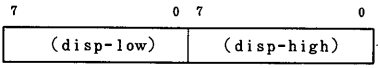
(2) 8ビット・メモリのビットCL

① 記述形式

NOT1 mem8,CL

② 命令語形式





③ バイト数

3／4／5

④ クロック数

9

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

(mem8)のビットNO.CL←$\overline{\text{(mem8)のビットNO.CL}}$

第1オペランドで指定される8ビット・メモリの，CLレジスタで指定されるビットが反転されます．CLの値は下位3ビットのみ有効です．

⑦ フラグの動作

なし

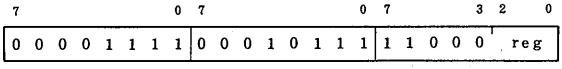
⑧ 記述例

(3) 16ビット・レジスタのビットCL

① 記述形式

NOT1 reg16,CL

② 命令語形式



③ バイト数

3

④ クロック数

4

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg16のビットNO.CL← $\overline{\text{reg16のビットNO.CL}}$

第1オペランドで指定される16ビット・レジスタの，CLレジスタで指定されるビットが反転されます．CLの値は下位4ビットのみ有効です．

⑦ フラグの動作

なし

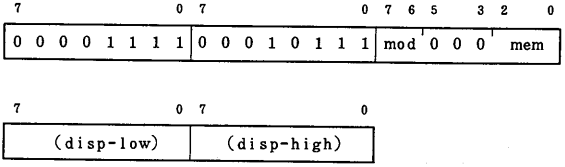
⑧ 記述例

(4) 16ビット・メモリのビットCL

① 記述形式

NOT1 mem16,CL

② 命令語形式



③ バイト数

3／4／5

④ クロック数

13：奇数アドレス

9：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

(mem16)のビットNO.CL←$\overline{\text{(mem16)のビットNO.CL}}$

第1オペランドでアドレスされる16ビット・メモリの，CLレジスタで指定されるビットが反転されます．CLの値は下位4ビットのみ有効です．

⑦ フラグの動作

なし．

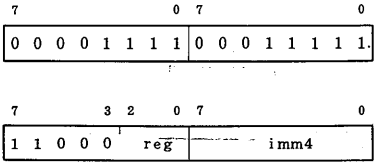
⑧ 記述例

(7) １６ビット・レジスタのビットｉｍｍ４

① 記述形式

NOT1　reg16，imm4

② 命令語形式



③ バイト数

4

④ クロック数

4

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

reg16のビットNO.imm4←$\overline{\text{reg16のビットNO.imm4}}$

第１オペランドで指定される１６ビット・レジスタの，第２オペランドに記述された４ビット・イミーディエト・データで指定されるビットが反転されます．命令の４バイト目のイミーディエト・データは，下位４ビットのみ有効です．

⑦ フラグの動作

なし

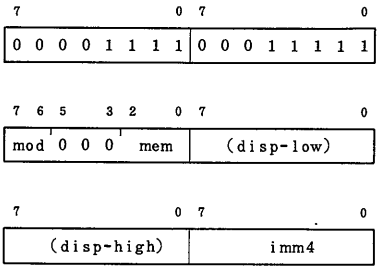
⑧ 記述例

(8) 16ビット・メモリのビットimm4

① 記述形式

NOT1 mem16,imm4

② 命令語形式



③ バイト数

4／5／6

④ クロック数

13：奇数アドレス

9：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

(mem16)のビットNO.imm4 ← $\overline{\text{(mem16)のビットNO.imm4}}$

第1オペランドでアドレスされる16ビット・メモリの，第2オペランドの4ビット・イミーディエト・データで指定されるビットが反転されます．命令の最終バイトのイミーディエト・データは，下位4ビットのみ有効です．

⑦ フラグの動作

なし

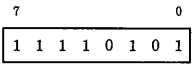
⑧ 記述例

(9) キャリー・フラグ

① 記述形式

NOT1 CY

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

$CY \leftarrow \overline{CY}$

CYフラグを反転します.

⑦ フラグの動作

V	S	Z	AC	P	CY
					×

⑧ 記述例

NOT1 CY

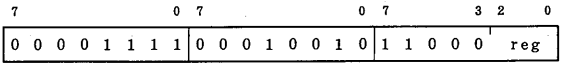
10．17．3　CLR1(Clear Bit)

(1)　8ビット　レジスタのビットCL

①　記述形式

CLR1　reg8,CL

②　命令語形式



③　バイト数

3

④　クロック数

4

⑤　16ビット・ワード転送回数

なし

⑥　機　能

reg8のビットNO.CL←0

第1オペランドで指定される8ビット・レジスタの，CLレジスタで指定されるビットがクリア(0)されます．CLの値は，下位3ビットのみ有効です．

⑦　フラグの動作

なし

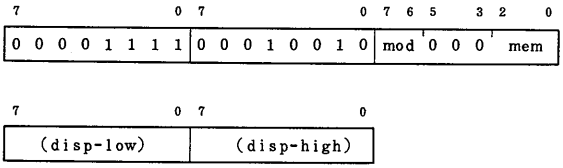
⑧　記述例

⑵ 8ビット・メモリのビットＣＬ

① 記述形式

ＣＬＲ1 mem8,ＣＬ

② 命令語形式



③ バイト数

３／４／５

④ クロック数

９

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

（mem8）のビットNO.ＣＬ←０

第１オペランドでアドレスされる８ビット・メモリの，ＣＬレジスタで指定されるビットがクリア(0)されます．ＣＬの値は，下位３ビットのみ有効です．

⑦ フラグの動作

なし

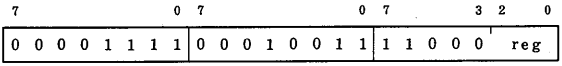
⑧ 記述例

⑶ １６ビット・レジスタのビットＣＬ

① 記述形式

CLR1 reg16,CL

② 命令語形式



③ バイト数

3

④ クロック数

4

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

reg16のビットNO.CL← 0

第１オペランドで指定される１６ビット・レジスタの，ＣＬレジスタで指定されるビットがクリア(0)されます．ＣＬの値は，下位４ビットのみ有効です．

⑦ フラグの動作

なし

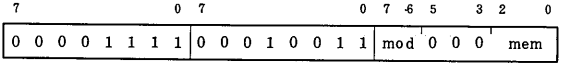
⑧ 記述例

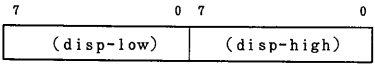
(4) 16ビット・メモリのビットCL

① 記述形式

CLR1 mem16,CL

② 命令語形式





③ バイト数

3／4／5

④ クロック数

13：奇数アドレス

9：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

(mem16)のビットNO.CL←0

第1オペランドでアドレスされる16ビット・メモリの，CLレジスタで指定されるビットがクリア(0)されます．CLの値は，下位4ビットのみ有効です．

⑦ フラグの動作

なし

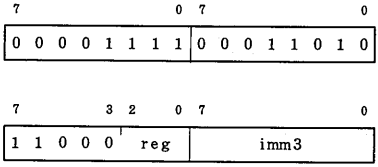
⑧ 記述例

(5) 8ビット・レジスタのビットimm3

① 記述形式

CLR1 reg8, imm3

② 命令語形式



③ バイト数

4

④ クロック数

4

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg8のビットNO. imm3 ← 0

第1オペランドで指定される8ビット・レジスタの，第2オペランドに記述された3ビット・イミーディエト・データで指定されるビットがクリア(0)されます．命令の4バイト目のイミーディエト・データは，下位3ビットのみ有効です．

⑦ フラグの動作

なし

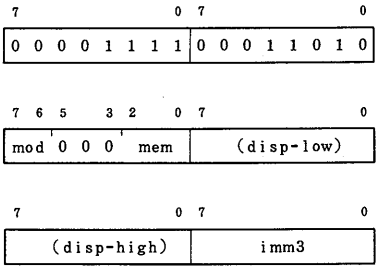
⑧ 記述例

(6) 8ビット・メモリのビットimm3

① 記述形式

CLR1 mem8,imm3

② 命令語形式



③ バイト数

4／5／6

④ クロック数

9

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

(mem8)のビットNO.imm3 ← 0

第1オペランドで指定される8ビット・メモリの，第2オペランドに記述された3ビット・イミーディエト・データで指定されるビットがクリア(0)されます．命令の最終バイトのイミーディエト・データは，下位3ビットのみ有効です．

⑦ フラグの動作

なし

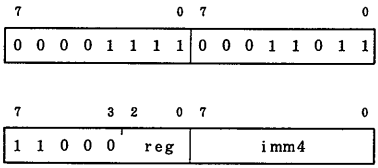
⑧ 記述例

(7) 16ビット・レジスタのビットimm4

① 記述形式

CLR1 reg16,imm4

② 命令語形式



③ バイト数

4

④ クロック数

4

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg16のビットNO.imm4←0

第1オペランドで指定される16ビット・レジスタの，第2オペランドに記述された4ビット・イミーディエト・データで指定されるビットがクリアされます．命令の4バイト目のイミーディエト・データは，下位4ビットのみ有効です．

⑦ フラグの動作

なし

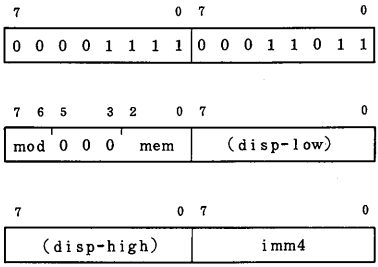
⑧ 記述例

(8) 16ビット・メモリのビットimm4

① 記述形式

CLR1 mem16,imm4

② 命令語形式



③ バイト数

4／5／6

④ クロック数

13：奇数アドレス

9：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

(mem16)のビットNO.imm4←0

第1オペランドでアドレスされる16ビット・メモリの，第2オペランドに記述された4ビット・イミーディエト・データで指定されるビットがクリア(0)されます．命令の最終バイトのイミーディエト・データは，下位4ビットのみ有効です．

⑦ フラグの動作

なし

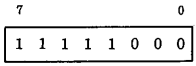
⑧ 記述例

(9) キャリー・フラグ

① 記述形式

CLR1 CY

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

CY←0

CYフラグをクリアします.

⑦ フラグの動作

V	S	Z	AC	P	CY
					0

⑧ 記述例

(10) 方向フラグ

① 記述形式

CLR1 DIR

② 命令語形式

7							0
1	1	1	1	1	1	0	0

③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

DIR←0

DIRフラグをクリアします.

MOVBK, CMPBK, CMPM, LDM, STM, INM, OUTMの各命令実行時に,インデクス・レジスタIX, IYをオートインクリメントするように設定します.

⑦ フラグの動作

DIR
0

⑧ 記述例

CLR1 DIR

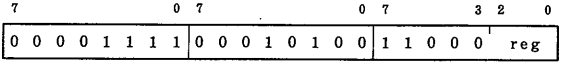
10．17．4 SET1(Set Bit)

(1) 8ビット・レジスタのビットCL

① 記述形式

SET1 reg8,CL

② 命令語形式



③ バイト数

3

④ クロック数

4

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg8のビットNO.CL←1

第1オペランドで指定される8ビット・レジスタの，CLレジスタで指定されるビットがセット(1)されます．CLの値は，下位3ビットのみ有効です．

⑦ フラグの動作

なし

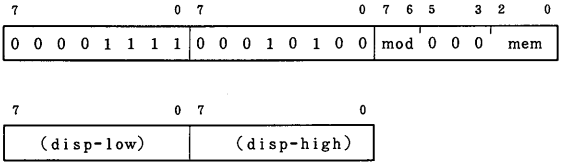
⑧ 記述例

(2) 8ビット・メモリのビットＣＬ

① 記述形式

SET1 mem8,CL

② 命令語形式



③ バイト数

3／4／5

④ クロック数

9

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

(mem8)のビットNO.CL←1

第１オペランドで指定される８ビット・メモリの，ＣＬレジスタで指定されるビットがセット(1)されます．ＣＬの値は，下位３ビットのみ有効です．

⑦ フラグの動作

なし

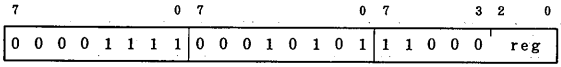
⑧ 記述例

⑶ １６ビット・レジスタのビットＣＬ

① 記述形式

SET1 reg16,CL

② 命令語形式



③ バイト数

3

④ クロック数

4

⑤ １６ビット・ワード転送回数

なし

⑥ 機 能

reg16のビットNO.CL←1

第１オペランドで指定される１６ビット・レジスタの，ＣＬレジスタで指定されるビットがセット(1)されます．ＣＬの値は，下位４ビットのみ有効です．

⑦ フラグの動作

なし

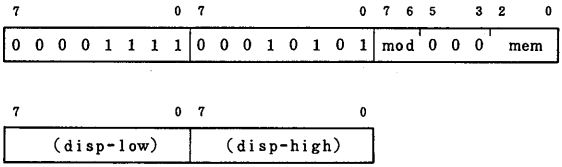
⑧ 記述例

(4) １６ビット・メモリのビットＣＬ

① 記述形式

SET1 mem16,CL

② 命令語形式



③ バイト数

３／４／５

④ クロック数

１３：奇数アドレス

９：偶数アドレス

⑤ １６ビット・ワード転送回数

２

⑥ 機　能

（mem16）のビットNO.CL←１

第１オペランドでアドレスされる１６ビット・メモリの，ＣＬレジスタで指定されるビットがセット（1）されます．ＣＬの値は，下位４ビットのみ有効です．

⑦ フラグの動作

なし

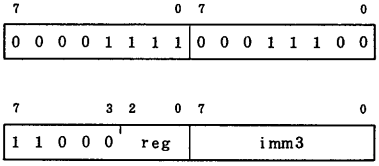
⑧ 記述例

(5) 8ビット・レジスタのビットimm3

① 記述形式

SET1 reg8,imm3

② 命令語形式



③ バイト数

4

④ クロック数

4

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

reg8のビットNO.imm3←1

第1オペランドで指定される8ビット・レジスタの，第2オペランドで記述された3ビット・イミーディエト・データで指定されるビットがセット(1)されます．命令の4バイト目のイミーディエト・データは，下位3ビットのみ有効です．

⑦ フラグの動作

なし

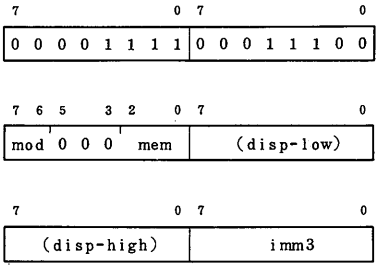
⑧ 記述例

(6) 8ビット・メモリのビットimm3

① 記述形式

SET1 mem8,imm3

② 命令語形式



③ バイト数

4／5／6

④ クロック数

9

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

(mem8)のビットNO.imm3←1

第1オペランドでアドレスされる8ビット・メモリの，第2オペランドに記述された3ビット・イミーディエト・データで指定されるビットがセット(1)されます．命令の最終バイトのイミーディエト・データは，下位3ビットのみ有効です．

⑦ フラグの動作

なし

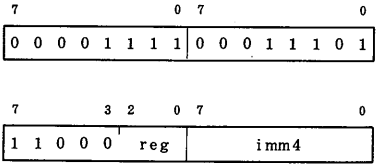
⑧ 記述例

(7) １６ビット・レジスタのビットｉｍｍ４

① 記述形式

ＳＥＴ１　ｒｅｇ１６，ｉｍｍ４

② 命令語形式



③ バイト数

４

④ クロック数

４

⑤ １６ビット・ワード転送回数

なし

⑥ 機　能

ｒｅｇ１６のビットＮＯ．ｉｍｍ４←１

第１オペランドで指定される１６ビット・レジスタの，第２オペランドに記述された４ビット・イミーディエト・データで指定されるビットがセット（1）されます．

命令の４バイト目のイミーディエト・データは，下位４ビットのみ有効です．

⑦ フラグの動作

なし

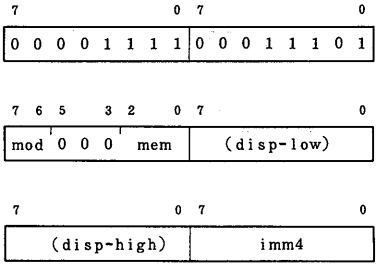
⑧ 記述例

(8) １６ビット・メモリのビットｉｍｍ４

① 記述形式

SET1 mem16，imm4

② 命令語形式



③ バイト数

４／５／６

④ クロック数

１３：奇数アドレス

９：偶数アドレス

⑤ １６ビット・ワード転送回数

２

⑥ 機　能

（mem16）のビットNO. imm4←１

第１オペランドでアドレスされる１６ビット・メモリの，第２オペランドに記述された４ビット・イミーディエト・データで指定されるビットがセット（1）されます．命令の最終バイトのイミーディエト・データは，下位４ビットのみ有効です．

⑦ フラグの動作

なし

⑧ 記述例

(9) キャリー・フラグ

① 記述形式

SET1 CY

② 命令語形式

③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

CY←1

CYフラグをセットします.

⑦ フラグの動作

V	S	Z	AC	P	CY
					1

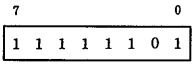
⑧ 記述例

(10) 方向フラグ

① 記述形式

SET1 DIR

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

DIR←1

DIRフラグをセットします.

MOVRK, CMPBK, CMPM, LDM, STM, INM, OUTMの各命令実行時, インデクス・レジスタIX, IYをオートデクリメントするように設定します.

⑦ フラグの動作

DIR
1

⑧ 記述例

SET1 DIR

10．18　シフト命令

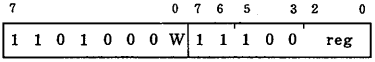
10．18．1　SHL（Shift Left）

(1)　レジスタ1ビット

①　記述形式

SHL　reg,1

②　命令語形式



③　バイト数

2

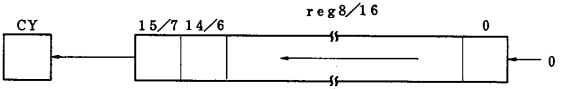
④　クロック数

2

⑤　16ビット・ワード転送回数

なし

⑥　機　能



第1オペランドで指定される8／16ビット・レジスタを1ビット左シフトします．指定されたレジスタのLSBには0がロードされ，MSBのデータはCYフラグにシフトされます．シフト後符号ビットが元のままであればVフラグがクリアされます．

⑦　フラグの動作

V	S	Z	AC	P	CY
×	×	×	U	×	×

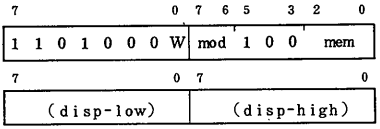
⑧　記述例

(2) メモリ1ビット

① 記述形式

SHL mem,1

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　7

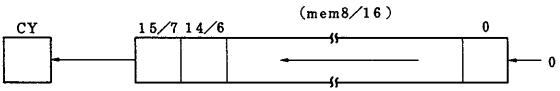
W＝1のとき　11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリを1ビット論理的または算術的に左シフトします．アドレスされたメモリのLSBには0がロードされ，MSBのデータはCYフラグにシフトされます．シフト後符号ビット（ビット7または15）が元のままであればVフラグがクリアされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	U	×	×

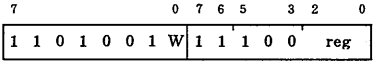
⑧ 記述例

(3) レジスタ可変ビット

① 記述形式

SHL reg, CL

② 命令語形式



③ バイト数

2

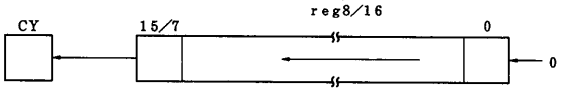
④ クロック数

2＋n （n：シフト数）

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能



第1オペランドで指定される8/16ビット・レジスタをレジスタCLに等しいビット数分左シフトします．指定されたレジスタのLSBには0がロードされ，MSBのデータはCYフラグにシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	×

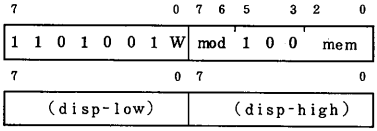
⑧ 記述例

(4) メモリ可変ビット

① 記述形式

SHL mem, CL

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　6＋n

W＝1のとき　10＋n：奇数アドレス

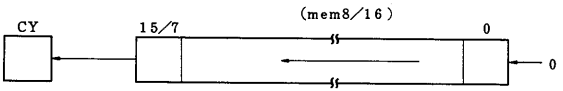
6＋n：偶数アドレス

（n：シフト数）

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリをレジスタCLに等しいビット数分左シフトします．アドレスされたメモリのLSBには0がシフトされ，MSBのデータはCYフラグにシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	×

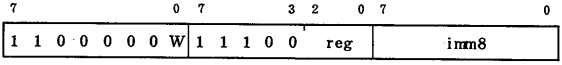
⑧ 記述例

(5) レジスタ・マルチビット

① 記述形式

SHL reg,imm8

② 命令語形式



③ バイト数

3

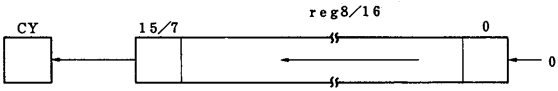
④ クロック数

2＋n　　n：シフト数

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8/16ビット・レジスタが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分左へシフトされます．8/16ビット・レジスタのLSBには0が入力され，MSBのデータはCYフラグにシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	×

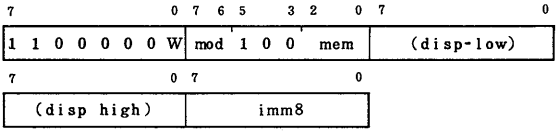
⑧ 記述例

(6) メモリ・マルチビット

① 記述形式

SHL mem, imm8

② 命令語形式



③ バイト数

3／4／5

④ クロック数

W＝0のとき　6＋n　　n：シフト数

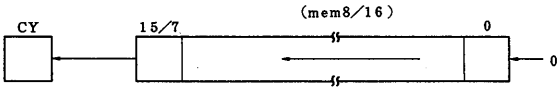
W＝1のとき　10＋n：奇数アドレス

6＋n：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分左へシフトされます．8／16ビット・メモリのLSBには0がロードされ，MSBのデータはCYフラグにシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	×

⑧ 記述例

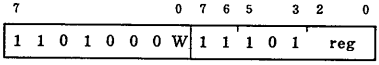
10．18．2　SHR（Shift Right）

(1)　レジスタ1ビット

①　記述形式

SHR　reg,1

②　命令語形式



③　バイト数

2

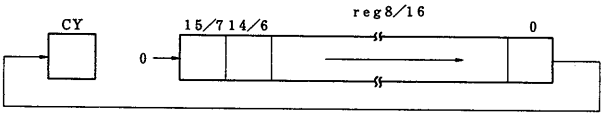
④　クロック数

2

⑤　16ビット・ワード転送回数

なし

⑥　機　能



第1オペランドで指定される8／16ビット・レジスタを1ビット論理的に右シフトします．指定されたレジスタのMSBには0がロードされ，LSBのデータはCYフラグにシフトされます．シフト後符号ビット（ビット7または15）が元のままであればVフラグがクリアされます．

⑦　フラグの動作

V	S	Z	AC	P	CY
×	×	×	U	×	×

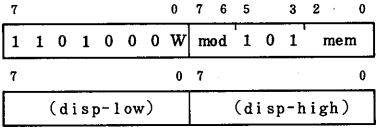
⑧　記述例

(2) メモリ１ビット

① 記述形式

SHR mem,1

② 命令語形式



③ バイト数

２／３／４

④ クロック数

W＝０のとき　７

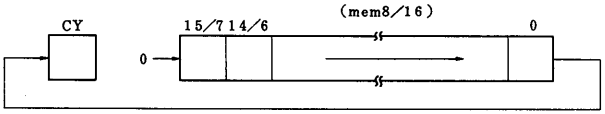
W＝１のとき　１１：奇数アドレス

７：偶数アドレス

⑤ １６ビット・ワード転送回数

２

⑥ 機　能



第１オペランドでアドレスされる8／16ビット・メモリを１ビット論理的に右シフトします．アドレスされたメモリのMSBには０がロードされ，LSBのデータはCYフラグにシフトされます．シフト後符号ビット（ビット７または１５）が元のままであればVフラグがクリアされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×	×	×	U	×	×

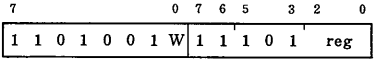
⑧ 記述例

(3) レジスタ可変ビット

① 記述形式

SHR reg,CL

② 命令語形式



③ バイト数

2

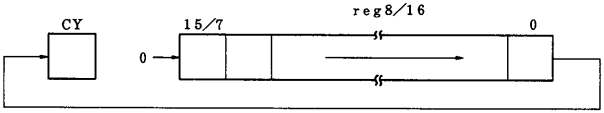
④ クロック数

2＋n （n：シフト数）

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能



第1オペランドで指定される8/16ビット・レジスタをレジスタCLに等しいビット数分論理的に右シフトします．指定されたレジスタのMSBには0がロードされ，LSBのデータはCYフラグにシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	×

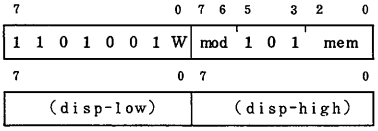
⑧ 記述例

(4) メモリ可変ビット

① 記述形式

SHR mem, CL

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　6＋n

W＝1のとき　10＋n：奇数アドレス

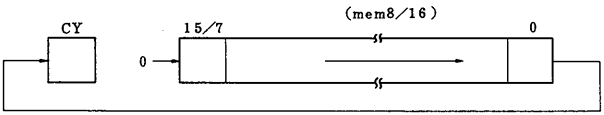
6＋n：偶数アドレス

（n：シフト数）

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリをレジスタCLに等しいビット数分論理的に右シフトします．アドレスされたメモリのMSBには0がロードされ，LSBのデータはCYフラグにシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	×

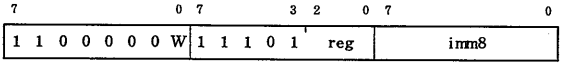
⑧ 記述例

(5) レジスタ・マルチビット

① 記述形式

SHR reg,imm8

② 命令語形式



③ バイト数

3

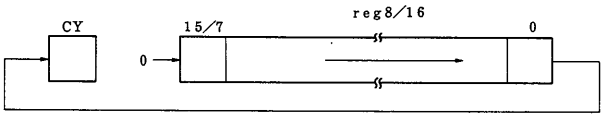
④ クロック数

2＋n　　n：シフト数

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8/16ビット・レジスタが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分右へシフトされます．8/16ビット・レジスタのMSBには0がロードされ，LSBのデータはCYフラグにシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	×

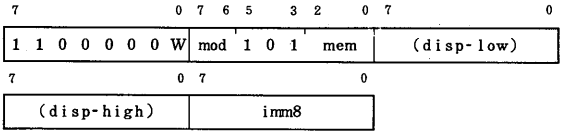
⑧ 記述例

(6) メモリ・マルチビット

① 記述形式

SHR mem, imm8

② 命令語形式



③ バイト数

3／4／5

④ クロック数

W＝0のとき　6＋n　　n：シフト数

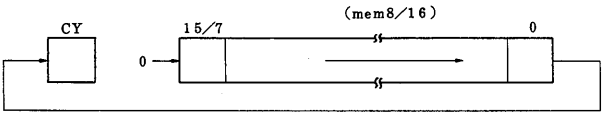
W＝1のとき　10＋n：奇数アドレス

6＋n：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドで指定される8/16ビット・メモリが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分右へシフトされます．8/16ビット・メモリのMSBには0がロードされ，LSBのデータはCYフラグにシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	×

⑧ 記述例

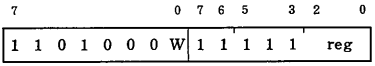
10．18．3　SHRA（Shift Right Arithmetic）

(1) レジスタ1ビット

① 記述形式

SHRA reg,1

② 命令語形式



③ バイト数

2

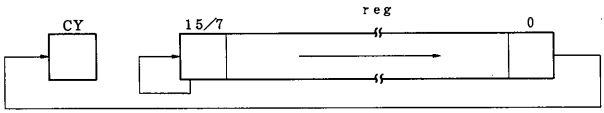
④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8／16ビット・レジスタを1ビット算術的に右シフトします．指定されたレジスタのMSBには元と同じ値のビットがシフトされ，シフト後も符号は変化しません．LSBのデータはCYフラグにシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	×

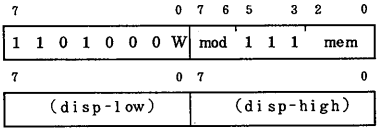
⑧ 記述例

(2) メモリ1ビット

① 記述形式

SHRA mem,1

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　7

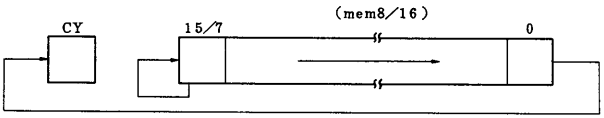
W＝1のとき　11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリを1ビット算術的に右シフトします．アドレスされたメモリのMSBには元と同じ値のビットがシフトされ，シフト後も符号は変化しません．LSBのデータはCYフラグにシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
0	×	×	U	×	×

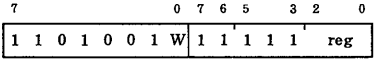
⑧ 記述例

(3) レジスタ可変ビット

① 記述形式

SHRA reg,CL

② 命令語形式



③ バイト数

2

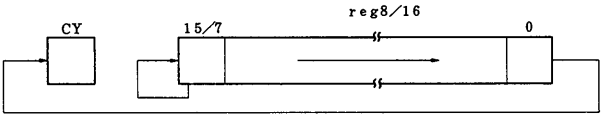
④ クロック数

2＋n （n：シフト数）

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能



第1オペランドで指定される8/16ビット・レジスタをレジスタCLに等しいビット数分，算術的に右シフトします．指定されたレジスタのMSBには元と同じ値のビットがシフトされ，シフト後も符号は変化しません．LSBのデータはCYフラグにシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	×

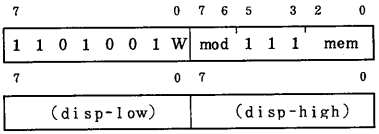
⑧ 記述例

(4) メモリ可変ビット

① 記述形式

SHRA mem.CL

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　6＋n

W＝1のとき　10＋n：奇数アドレス

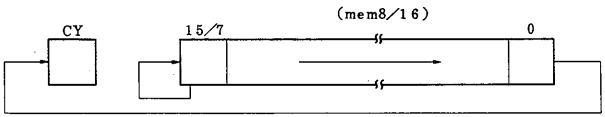
6＋n：偶数アドレス

（n：シフト数）

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8/16ビット・メモリをレジスタCLに等しいビット数分，算術的に右シフトします．アドレスされたメモリのMSBには元と同じ値のビットがシフトされ，シフト後も符号は変化しません．LSBのデータはCYフラグにシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	×

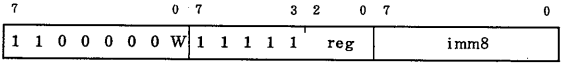
⑧ 記述例

(5) レジスタ・マルチビット

① 記述形式

SHRA reg,imm8

② 命令語形式



③ バイト数

3

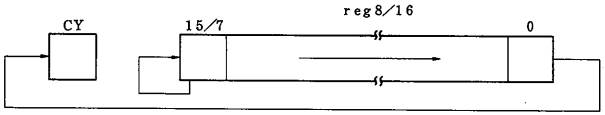
④ クロック数

2＋n　　n：シフト数

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8/16ビット・レジスタが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分，算術的に右シフトされます．8/16ビット・レジスタのLSBのデータはCYフラグにシフトされますが，MSBには元の値が再ロードされて変化せず，符号は保存されます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	×

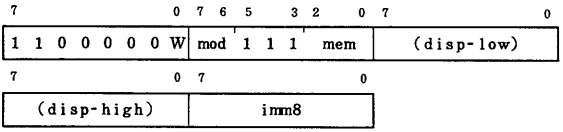
⑧ 記述例

(6) メモリ・マルチビット

① 記述形式

SHRA mem, imm8

② 命令語形式



③ バイト数

3／4／5

④ クロック数

W＝0のとき　6＋n　　n：シフト数

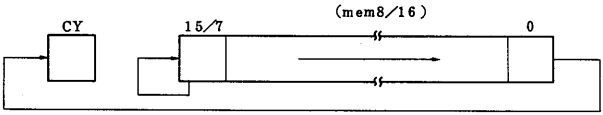
W＝1のとき　10＋n：奇数アドレス

6＋n：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分，算術的に右シフトされます．8／16ビット・メモリのLSBのデータはCYフラグにシフトされますが，MSBには元の値が再ロードされて変化せず，符号は保存されます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U	×	×	U	×	×

⑧ 記述例

10．19　ローテート命令

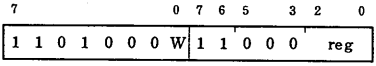
10．19．1　ROL（Rotate Left）

(1)　レジスタ1ビット

①　記述形式

ROL　reg,1

②　命令語形式



③　バイト数

2

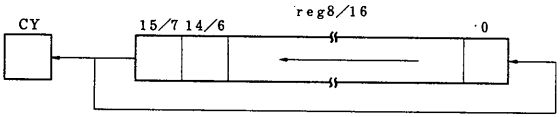
④　クロック数

2

⑤　16ビット・ワード転送回数

なし

⑥　機　能



第1オペランドで指定される8／16ビット・レジスタを1ビット左回転します．指定されたレジスタのMSBが変化したときはVフラグがセットされ，元のままのときはクリアされます．

⑦　フラグの動作

V	S	Z	AC	P	CY
×					×

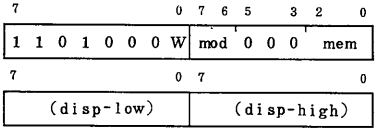
⑧　記述例

⑵ メモリ１ビット

① 記述形式

ROL mem,1

② 命令語形式



③ バイト数

２／３／４

④ クロック数

W＝０のとき　７

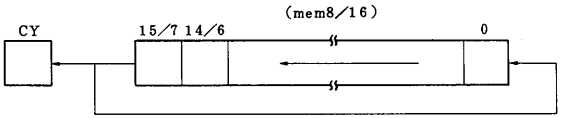
W＝１のとき　１１：奇数アドレス

７：偶数アドレス

⑤ １６ビット・ワード転送回数

２

⑥ 機　能



第１オペランドでアドレスされる８／１６ビット・メモリを１ビット左回転します．アドレスされたメモリのＭＳＢが変化したときはＶフラグがセットされ，元のままのときはクリアされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×					×

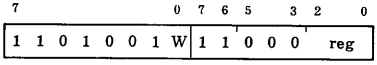
⑧ 記述例

⑶ レジスタ可変ビット

① 記述形式

ROL reg,CL

② 命令語形式



③ バイト数

2

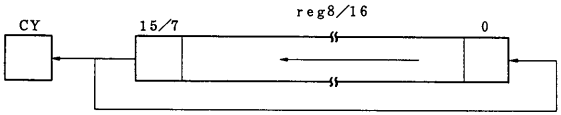
④ クロック数

2＋n　　n：シフト数

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8/16ビット・レジスタをレジスタCLに等しいビット数分左回転します.

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

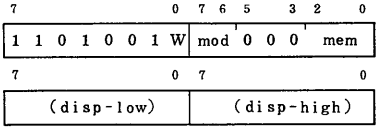
⑧ 記述例

(4) メモリ可変ビット

① 記述形式

ROL mem, CL

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　6＋n

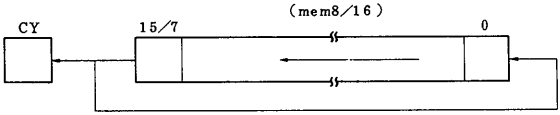
W＝1のとき　10＋n：奇数アドレス

6＋n：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8/16ビット・メモリをレジスタCLに等しいビット数分左回転します.

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

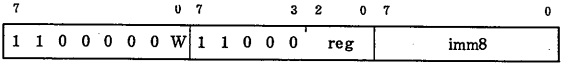
⑧ 記述例

⑸ レジスタ・マルチビット

① 記述形式

ROL reg, imm8

② 命令語形式



③ バイト数

3

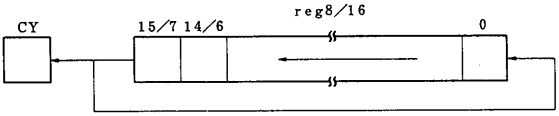
④ クロック数

2＋n　　n：シフト数

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8/16ビット・レジスタが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分，左回転されます．8/16ビット・レジスタのMSBのデータはCYフラグにシフトされる一方，自身のLSBにロードされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

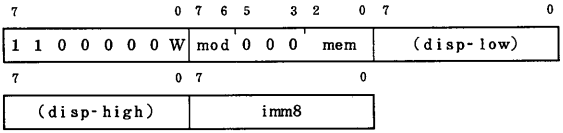
⑧ 記述例

⑹ メモリ・マルチビット

① 記述形式

ROL mem, imm8

② 命令語形式



③ バイト数

3／4／5

④ クロック数

W＝0のとき　6＋n　　n：シフト数

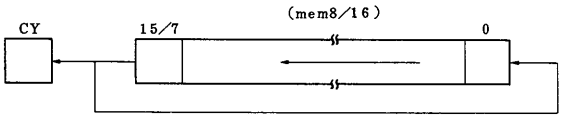
W＝1のとき　10＋n：奇数アドレス

6＋n：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分，左回転されます．8／16ビット・メモリのMSBのデータは，CYフラグにシフトされる一方，自身のLSBにロードされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

⑧ 記述例

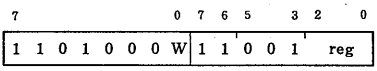
10．19．2　ROR（Rotate Right）

(1) レジスタ1ビット

① 記述形式

ROR reg,1

② 命令語形式



③ バイト数

2

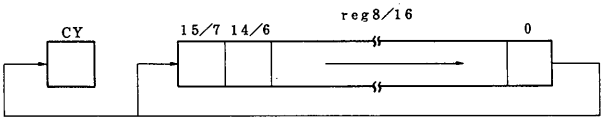
④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8/16ビット・レジスタを1ビット右回転します．指定されたレジスタのMSBが変化したときはオーバフロー・フラグがセットされ，元のままのときはクリアされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×					×

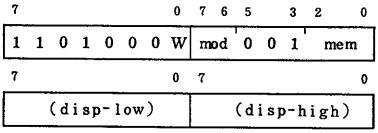
⑧ 記述例

② メモリ1ビット

① 記述形式

ROR mem, 1

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　7

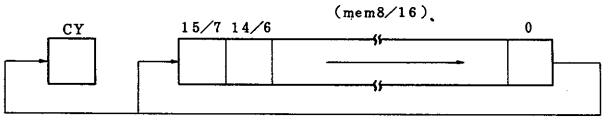
W＝1のとき　11：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリを1ビット右回転します．アドレスされたメモリのMSBが変化したときはオーバフロー・フラグがセットされ，元のままのときはクリアされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×					×

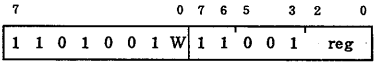
⑧ 記述例

⑶ レジスタ可変ビット

① 記述形式

ROR reg,CL

② 命令語形式



③ バイト数

2

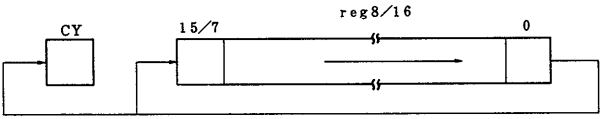
④ クロック数

2＋n （n：シフト数）

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能



第1オペランドで指定される8/16ビット・レジスタをレジスタCLに等しいビット数分右回転します.

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

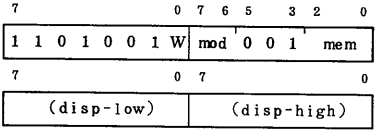
⑧ 記述例

(4) メモリ可変ビット

① 記述形式

ROR mem, CL

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　6＋n

W＝1のとき　10＋n：奇数アドレス

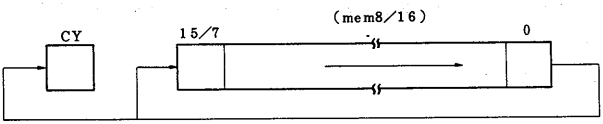
6＋n：偶数アドレス

（n：シフト数）

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリをレジスタCLに等しいビット数分右回転します.

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

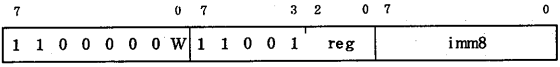
⑧ 記述例

(5) レジスタ・マルチビット

① 記述形式

ROR reg, imm8

② 命令語形式



③ バイト数

3

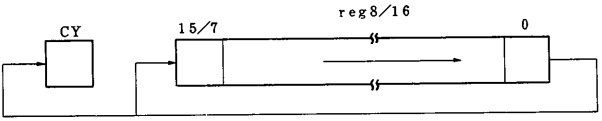
④ クロック数

2＋n　　n：シフト数

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8/16ビット・レジスタが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分，右回転されます．8/16ビット・レジスタのLSBのデータは，自身のMSBにシフトされる一方，CYフラグにもシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

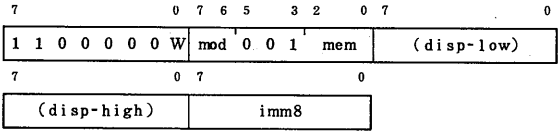
⑧ 記述例

(6) メモリ・マルチビット

① 記述形式

ROR mem, imm8

② 命令語形式



③ バイト数

3／4／5

④ クロック数

W＝0のとき　6＋n　　n：シフト数

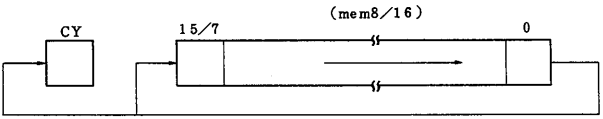
W＝1のとき　10＋n：奇数アドレス

6＋n：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分，右回転されます．8／16ビット・メモリのLSBのデータは，自身のMSBにシフトされる一方，CYフラグにもシフトされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

⑧ 記述例

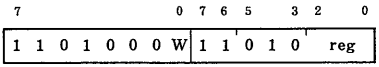
10．19．3　ROLC(Rotate Left with Carry)

(1)　レジスタ1ビット

①　記述形式

ROLC　reg,1

②　命令語形式



③　バイト数

2

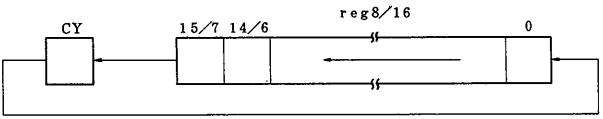
④　クロック数

2

⑤　16ビット・ワード転送回数

なし

⑥　機　能



第1オペランドで指定される8/16ビット・レジスタをCYフラグを通して1ビット左回転します．指定されたレジスタのMSBが変化したときはVフラグがセットされ，元のままのときはクリアされます．

⑦　フラグの動作

V	S	Z	AC	P	CY
×					×

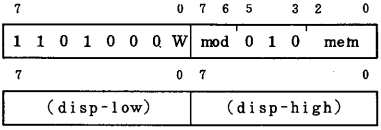
⑧　記述例

⑵ メモリ１ビット

① 記述形式

ROLC mem,1

② 命令語形式



③ バイト数

２／３／４

④ クロック数

W＝０のとき　７

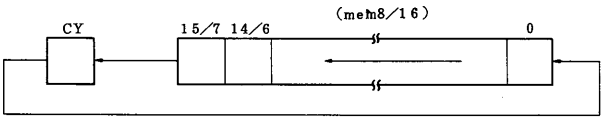
W＝１のとき　１１：奇数アドレス

７：偶数アドレス

⑤ １６ビット・ワード転送回数

２

⑥ 機　能



第１オペランドでアドレスされる８／１６ビット・メモリをCYフラグを通して１ビット左回転します．アドレスされたメモリのMSBが変化したときはVフラグがセットされ，元のままのときはクリアされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×					×

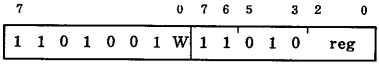
⑧ 記述例

(3) レジスタ可変ビット

① 記述形式

ROLC reg,CL

② 命令語形式



③ バイト数

2

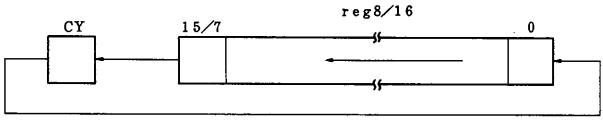
④ クロック数

2＋n　　n：シフト数

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8/16ビット・レジスタをCYフラグを通してレジスタCLに等しいビット数分左回転します.

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

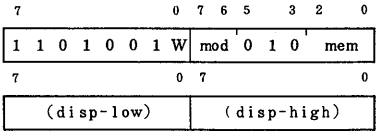
⑧ 記述例

(4) メモリ可変ビット

① 記述形式

ROLC mem, CL

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　6＋n　　n：シフト数

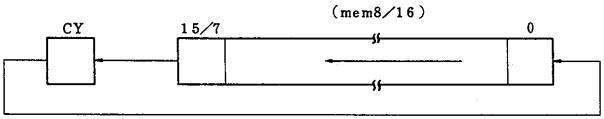
W＝1のとき　10＋n：奇数アドレス

6＋n：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリをCYフラグを通してレジスタCLに等しいビット数分左回転します.

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

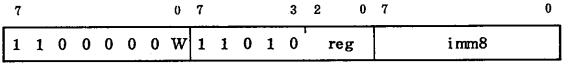
⑧ 記述例

(5) レジスタ・マルチビット

① 記述形式

ROLC reg, imm8

② 命令語形式



③ バイト数

3

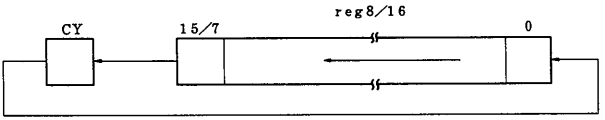
④ クロック数

2＋n　　n：シフト数

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8/16ビット・レジスタが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分，CYフラグを通して左回転されます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

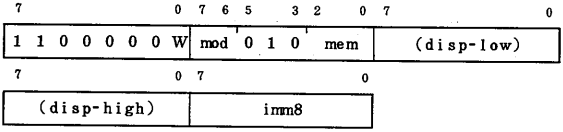
⑧ 記述例

⑹ メモリ・マルチビット

① 記述形式

ROLC mem, imm8

② 命令語形式



③ バイト数

3／4／5

④ クロック数

W＝0のとき　6＋n　　n：シフト数

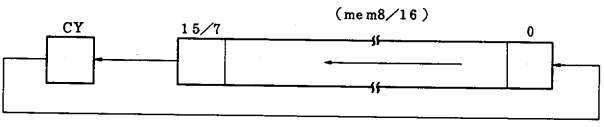
W＝1のとき　10＋n：奇数アドレス

6＋n：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分，CYフラグを通して左回転されます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

⑧ 記述例

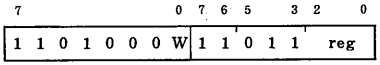
10．19．4　RORC（Rotate Right with Carry）

(1) レジスタ1ビット

① 記述形式

RORC　reg,1

② 命令語形式



③ バイト数

2

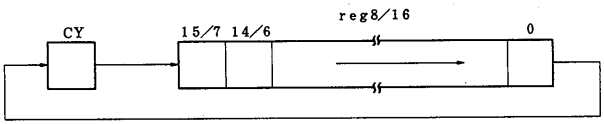
④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8/16ビット・レジスタをCYフラグを通して1ビット右回転します．指定されたレジスタのMSBが変化したときはVフラグがセットされ，元のままのときはクリアされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×					×

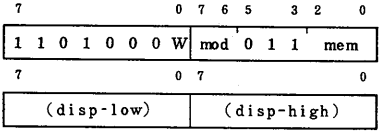
⑧ 記述例

⑵ メモリ１ビット

① 記述形式

RORC mem, 1

② 命令語形式



③ バイト数

２／３／４

④ クロック数

W＝０のとき　７

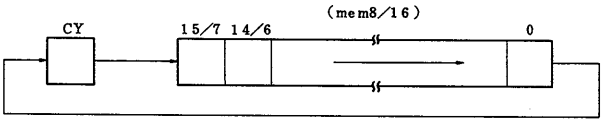
W＝１のとき　１１：奇数アドレス

７：偶数アドレス

⑤ １６ビット・ワード転送回数

２

⑥ 機　能



第１オペランドでアドレスされる８/１６ビット・メモリをCYフラグを通して１ビット右回転します．アドレスされたメモリのMSBが変化したときはVフラグがセットされ，元のままのときはクリアされます．

⑦ フラグの動作

V	S	Z	AC	P	CY
×					×

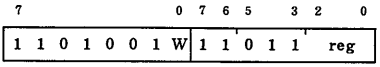
⑧ 記述例

⑶ レジスタ可変ビット

① 記述形式

RORC reg,CL

② 命令語形式



③ バイト数

2

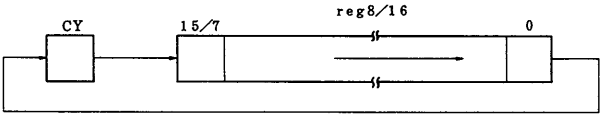
④ クロック数

2＋n　　n：シフト数

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8/16ビット・レジスタをCYフラグを通してレジスタCLに等しいビット数分右回転します.

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

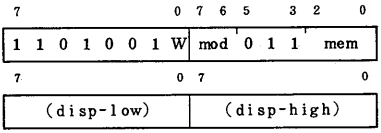
⑧ 記述例

(4) メモリ可変ビット

① 記述形式

RORC mem, CL

② 命令語形式



③ バイト数

2／3／4

④ クロック数

W＝0のとき　6＋n　　n：シフト数

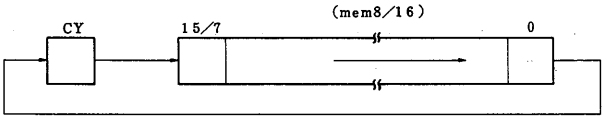
W＝1のとき　10＋n：奇数アドレス

6＋n：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能



第1オペランドでアドレスされる8／16ビット・メモリをキャリー・フラグを通してレジスタCLに等しいビット数分右回転します.

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

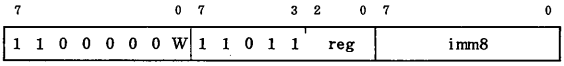
⑧ 記述例

(5) レジスタ・マルチビット

① 記述形式

RORC reg,imm8

② 命令語形式



③ バイト数

3

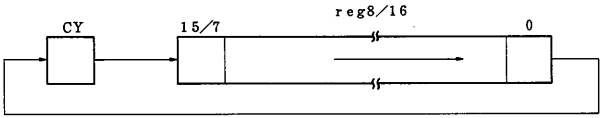
④ クロック数

2＋n　　n：シフト数

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能



第1オペランドで指定される8/16ビット・レジスタが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分，CYフラグを通して右回転されます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

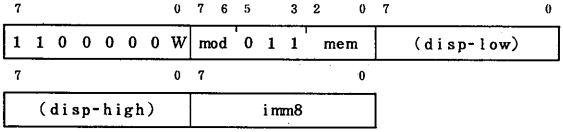
⑧ 記述例

(6) メモリ・マルチビット

① 記述形式

RORC mem, imm8

② 命令語形式



③ バイト数

3／4／5

④ クロック数

W＝0のとき　6＋n　　n：シフト数

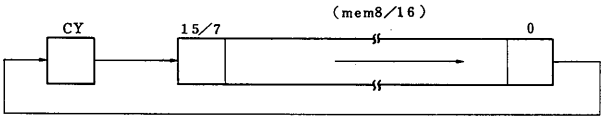
W＝1のとき　10＋n：奇数アドレス

6＋n：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能



第1オペランドでアドレスされる8／16ビット・メモリが，第2オペランドに記述された8ビット・イミーディエト・データで指定されるビット数分，CYフラグを通して右回転されます．

⑦ フラグの動作

V	S	Z	AC	P	CY
U					×

⑧ 記述例

10．20　サブルーチン制御命令

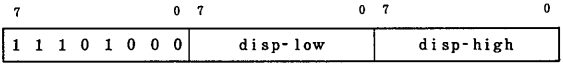
10．20．1　CALL（Call）

(1) セグメント内レラティブ

① 記述形式

CALL near-proc

② 命令語形式



③ バイト数

3

④ クロック数

9：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機　能

SP←SP−2

(SP+1，SP)←PC

PC←PC＋disp

PCがスタックに退避されます．次にPCに16ビット相対アドレスがロードされます．この命令によって，この命令の置かれているセグメント内の任意のアドレスをコール可能です．

⑦ フラグの動作

なし

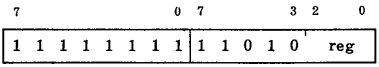
⑧ 記述例

⑵ セグメント内レジスタ

① 記述形式

CALL regptr16

② 命令語形式



③ バイト数

2

④ クロック数

9：奇数アドレス

7：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

SP←SP−2

(SP+1, SP)←PC

PC←regptr16

PCがスタックに退避されます．次にオペランドで指定される16ビット・レジスタの値（オフセット）がPCにロードされます．

この命令の置かれているセグメント内の任意のアドレスをコール可能です．

⑦ フラグの動作

なし

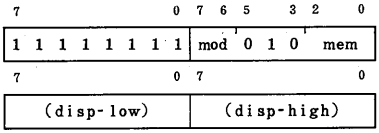
⑧ 記述例

(3) セグメント内メモリ

① 記述形式

CALL memptr16

② 命令語形式



③ バイト数

2／3／4

④ クロック数

15：奇数アドレス

11：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

TA←(memptr16+1,memptr16)

SP←SP−2,(SP+1,SP)←PC

PC←TA

PCがスタックに退避されます．次にオペランドでアドレスされる16ビット・メモリの内容(オフセット)がPCにロードされます．

この命令の置かれているセグメント内の任意のアドレスをコール可能です．

⑦ フラグの動作

なし

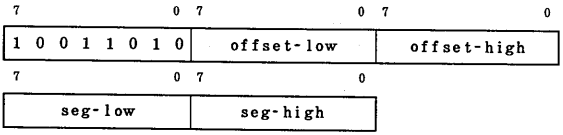
⑧ 記述例

(4) セグメント外ダイレクト

① 記述形式

CALL far-proc

② 命令語形式



③ バイト数

5

④ クロック数

13：奇数アドレス

9：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

SP←SP−2

(SP+1, SP)←PS

PS←seg

SP←SP−2

(SP+1, SP)←PC

PC←offset

PSとPCがスタックに退避されます．次にPSには命令の4，5バイト目が，PCには命令の2，3バイト目がロードされます．この命令によって，任意のセグメントの任意のアドレスをコール可能です．

⑦ フラグの動作

なし

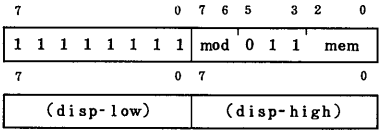
⑧ 記述例

⑸ セグメント外メモリ

① 記述形式

CALL memptr32

② 命令語形式



③ バイト数

2／3／4

④ クロック数

23：奇数アドレス

15：偶数アドレス

⑤ 16ビット・ワード転送回数

4

⑥ 機 能

TA←(memptr32+1,memptr32)

TB←(memptr32+3,memptr32+2)

SP←SP−2,(SP+1,SP)←PS

PS←TB

SP←SP−2,(SP+1,SP)←PC

PC←TA

PSとPCがスタックに退避されます．次にオペランドでアドレスされる32ビット・メモリの上位2バイトがPSに，下位2バイトがPCにロードされます．この命令によって任意のセグメントの任意のアドレスをコール可能です．

⑦ フラグの動作

なし

⑧ 記述例

10．20．2　RET(Return from Procedure)

(1) セグメント内

① 記述形式

RET（オペランドなし）

② 命令語形式

7							0
1	1	0	0	0	0	1	1

③ バイト数

1

④ クロック数

12：奇数アドレス

10：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機　能

PC←(SP＋1,SP)

SP←SP＋2

PCをスタックからリストアします．セグメント内コールからのリターンに使用されます．(3)のRETとの区別は，アセンブラが自動的に行います．

⑦ フラグの動作

なし

⑧ 記述例

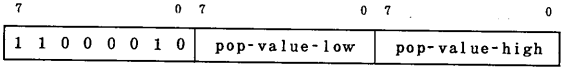
RET

⑵ セグメント内SP加算

① 記述形式

RET pop-value

② 命令語形式



③ バイト数

3

④ クロック数

12：奇数アドレス

10：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

PC←(SP+1,SP)

SP←SP+2

SP←SP+pop-value

PCがスタックからリストアされます．次にオペランドで指定される16ビット・ポップ値が加算されます．

この命令は，PCに続いて退避してあったパラメータが不要となったとき，不要なパラメータの分だけSPの値をとばすのに有効です．

セグメント内コールからのリターンに使用されます．(4)のRET pop-valueとの区別は，アセンブラが自動的に行います．

⑦ フラグの動作

なし

⑧ 記述例

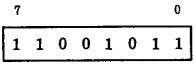
RET 8

(3) セグメント外

① 記述形式

RET（オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

16：奇数アドレス

12：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機　能

PC←(SP＋1, SP)

PS←(SP＋3, SP＋2)

SP←SP＋4

PCとPSがスタックからリストアされます．セグメント外コールからのリターンに使用されます．(1)のRETとの区別は，アセンブラが自動的に行います．

⑦ フラグの動作

なし

⑧ 記述例

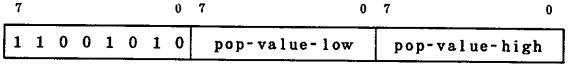
RET

(4) セグメント外SP加算

① 記述形式

RET pop-value

② 命令語形式



③ バイト数

3

④ クロック数

16：奇数アドレス

12：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

PC←(SP+1,SP)

PS←(SP+3,SP+2)

SP←SP+4

SP←SP+pop-value

PCとPSがスタックからリストアされます．次にオペランドで指定される16ビット・ポップ値がSPに加算されます．この命令は，PCとPSに続いて退避してあったパラメータが不要となったとき，SPの値をとばすのに有効です．

セグメント外コールからのリターンに使用されます．(2)のRET pop-valueとの区別は，アセンブラが自動的に行います．

⑦ フラグの動作

なし

⑧ 記述例

10.21　スタック操作命令

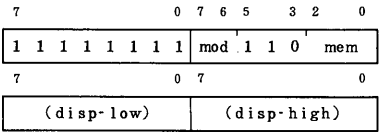
10.21.1　PUSH(Push)

(1)　16ビット・メモリ

①　記述形式

PUSH　mem16

②　命令語形式



③　バイト数

2／3／4

④　クロック数

9：奇数アドレス

5：偶数アドレス

⑤　16ビット・ワード転送回数

2

⑥　機　能

(SP−1, SP−2)←(mem16+1, mem16)

SP←SP−2

オペランドでアドレスされる16ビット・メモリの内容をスタックに退避します.

⑦　フラグの動作

なし

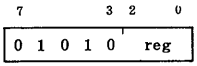
⑧　記述例

⑵ １６ビット・レジスタ

① 記述形式

PUSH reg16

② 命令語形式



③ バイト数

1

④ クロック数

5：奇数アドレス

3：偶数アドレス

⑤ １６ビット・ワード転送回数

1

⑥ 機　能

(SP−1, SP−2)←reg16

SP←SP−2

オペランドで指定される１６ビット・レジスタをスタックに退避します.

⑦ フラグの動作

なし

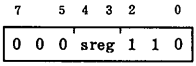
⑧ 記述例

(3) セグメント・レジスタ

① 記述形式

PUSH sreg

② 命令語形式



③ バイト数

1

④ クロック数

5：奇数アドレス

3：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

(SP−1, SP−2)←sreg

SP←SP−2

オペランドで指定されるセグメント・レジスタをスタックに退避します.

⑦ フラグの動作

なし

⑧ 記述例

(4) プログラム・ステータス・ワード

① 記述形式

PUSH PSW

② 命令語形式

7							0
1	0	0	1	1	1	0	0

③ バイト数

1

④ クロック数

5：奇数アドレス

3：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

(SP−1, SP−2)←PSW

SP←SP−2

PSWをスタックに退避します.

⑦ フラグの動作

なし

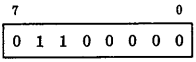
⑧ 記述例

⑸ レジスタ・セット

① 記述形式

PUSH R

② 命令語形式



③ バイト数

1

④ クロック数

36：奇数アドレス

20：偶数アドレス

⑤ 16ビット・ワード転送回数

8

⑥ 機 能

temp←SP

(SP−1, SP−2)←AW

(SP−3, SP−4)←CW

(SP−5, SP−6)←DW

(SP−7, SP−8)←BW

(SP−9, SP−10)←temp

(SP−11, SP−12)←BP

(SP−13, SP−14)←IX

(SP−15, SP−16)←IY

SP←SP−16

8個の16ビット・レジスタ(AW, BW, CW, DW, SP, BP, IX, IY)をスタックに退避します.

⑦ フラグの動作

なし

⑧ 記述例

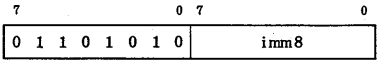
PUSH R

⑹ 8ビット・イミーディエト・データ

① 記述形式

PUSH imm8

② 命令語形式



③ バイト数

2

④ クロック数

5：奇数アドレス

3：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

(SP−1, SP−2)←imm

SP←SP−2

オペランドに記述された8ビット・イミーディエト・データがサイン拡張されて，16ビット・データとしてSPでアドレスされるスタックに退避されます.

⑦ フラグの動作

なし

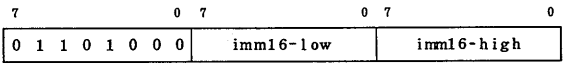
⑧ 記述例

(7) 16ビット・イミーディエト・データ

① 記述形式

PUSH imm16

② 命令語形式



③ バイト数

3

④ クロック数

5：奇数アドレス

3：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

(SP−1, SP−2) ← imm

SP ← SP−2

オペランドに記述された16ビット・イミーディエト・データが，SPでアドレスされるスタックに退避されます．

⑦ フラグの動作

なし

⑧ 記述例

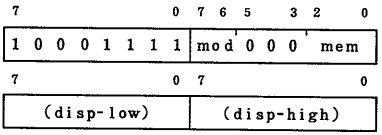
10．21．2　POP（Pop）

(1)　16ビット・メモリ

①　記述形式

POP　mem16

②　命令語形式



③　バイト数

2／3／4

④　クロック数

9：奇数アドレス

5：偶数アドレス

⑤　16ビット・ワード転送回数

2

⑥　機　能

SP←SP＋2

（mem16）←（SP−1，SP−2）

オペランドでアドレスされる16ビット・メモリに，スタックの内容を転送します．

⑦　フラグの動作

なし

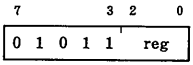
⑧　記述例

(2) 16ビット・レジスタ

① 記述形式

POP reg16

② 命令語形式



③ バイト数

1

④ クロック数

7：奇数アドレス

5：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

SP←SP＋2

reg16←(SP−1, SP−2)

オペランドで指定される16ビット・レジスタに，スタックの内容を転送します．

⑦ フラグの動作

なし

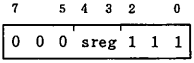
⑧ 記述例

(3) セグメント・レジスタ

① 記述形式

POP sreg

② 命令語形式



③ バイト数

1

④ クロック数

7：奇数アドレス

5：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

SP←SP＋2

sreg←(SP−1, SP−2)

オペランドで指定されるセグメント・レジスタ(PSは除く)に，スタックの内容を転送します．

⑦ フラグの動作

なし

⑧ 記述例

⑨ 注 意

この命令と次の命令との間では，ハードウエア割り込み（マスカブル割り込み，ノンマスカブル割り込み）要求およびシングル・ステップ・ブレークは受け付けられません．

(4) プログラム・ステータス・ワード

① 記述形式

POP PSW

② 命令語形式

③ バイト数

1

④ クロック数

7：奇数アドレス

5：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

SP←SP＋2

PSW←(SP−1, SP−2)

PSWにスタックの内容を転送します.

⑦ フラグの動作

V	DIR	IE	BRK	S	Z	AC	P	CY
R	R	R	R	R	R	R	R	R

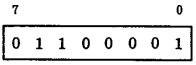
⑧ 記述例

⑸ レジスタ・セット

① 記述形式

POP R

② 命令語形式



③ バイト数

1

④ クロック数

38：奇数アドレス

22：偶数アドレス

⑤ 16ビット・ワード転送回数

7

⑥ 機 能

IY←(SP+1,SP)

IX←(SP+3,SP+2)

BP←(SP+5,SP+4)

temp←(SP+7,SP+6)

BW←(SP+9,SP+8)

DW←(SP+11,SP+10)

CW←(SP+13,SP+12)

AW←(SP+15,SP+14)

SP←SP+16

SPでアドレスされるスタックの内容が，8個の16ビット・レジスタ(AW,BW,CW,DW,BP,SP,IX,IY)に回復されます.

⑦ フラグの動作

なし

⑧ 記述例

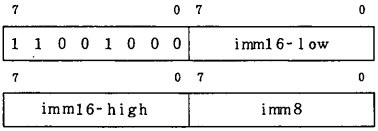
POP R

10．21．3　PREPARE（Prepare New Stack Frame）

① 記述形式

PREPARE　imm16，imm8

② 命令語形式



③ バイト数

4

④ クロック数

imm8＝0のとき　15

imm8≧1のとき　17＋12（imm8－1）：奇数アドレス

15＋ 8（imm8－1）：偶数アドレス

⑤ 16ビット・ワード転送回数

imm8＝0のとき　1

imm8≧1のとき　imm8×2

⑥ 機　能

（SP－1，SP－2）←BP

SP←SP－2

temp←SPを行ったあと，

imm8＞0のとき次の動作を，

（SP－1，SP－2）←（BP－1，BP－2）

SP←SP－2　　＊1

BP←BP－2

“imm8－1”回繰り返したあとに，

（SP－1，SP－2）←temp

SP←SP－2　　＊2

を実行します．

このあと次の処理を行います．

BP←temp

SP←SP－imm16

imm8＝1のとき，＊1の動作を行いません．

imm8＝0のとき，＊1および＊2の動作を行いません．

この命令は，ブロック構造の高級言語（たとえばPascal，Adaなど）において必要な“スタック・フレーム”を生成するために用いられます．スタック・フレームには，その

プロシージャから参照してもよい変数を含むフレームを指すポインタ群とローカル変数の領域が含まれます．

この命令は，ローカル変数の領域確保と，グローバル変数への参照を可能とするため，フレーム・ポインタのコピーを行います．第1オペランドに記述された16ビット・イミーディエト・データでローカル変数用として確保する領域の大きさ（バイト単位）が指定され，第2オペランドに記述された8ビット・イミーディエト・データでプロシージャ・ブロックの深さ（この深さをレキシカル・レベルといいます）が示されます．

この命令によって生成されるフレームのベース・アドレスは，ベース・ポインタBPにセットされます．

まずBPをスタックへ退避します．これは，プロシージャの終了時点で，コールした側のプロシージャのBPを復帰するためです．次に，コールされたプロシージャから参照可能な範囲のフレーム・ポインタ（退避されたBP）をスタックに積みます．参照可能な範囲は，そのプロシージャのレキシカル・レベルより1減じた値となります．

レキシカル・レベルが1以上の場合には，自分自身のフレーム・ポインタもスタックへ積みます．これは，このプロシージャからコールされたプロシージャにおいてフレーム・ポインタのコピーを行う場合に，コールしたプロシージャのフレーム・ポインタもコピーするためです．

次にBPに新しいフレーム・ポインタの値をセットし，そのプロシージャで使用されるローカル変数の領域をスタックに確保します．つまり，SPをローカル変数分だけ減らします．

⑦ フラグの動作

なし

⑧ 記述例

注意　PREPARE命令の次に下記のいずれかの命令を実行させないでください．

```
ADJ4A    ADJ4S    ADJBA    ADJBS    CVTBD    CVTDB    DIV
DIVU     MOV DS0,reg16,mem32     MOV DS1,reg16,mem32
MUL      MULU     PUSH R
```

実行させる場合は，PREPARE命令と上記命令との間にNOP命令を挿入してください．

例　PREPARE命令の次にPUSH　R命令を実行させたい場合

```
PREPARE  10,5
NOP
PUSH  R
```

10．21．4　DISPOSE(Dispose a Stack Frame)

① 記述形式

DISPOSE（オペランドなし）

② 命令語形式

7							0
1	1	0	0	1	0	0	1

③ バイト数

1

④ クロック数

8：奇数アドレス

6：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機　能

SP←BP

BP←(SP＋1,SP)

SP←SP＋2

この命令は，PREPARE命令で生成されたスタック・フレームを1フレーム分リリースします．

BPには，1つ前のフレームを指すポインタ値をロードし，SPには，フレームの最下位を示すポインタ値をロードします．

⑦ フラグの動作

なし

⑧ 記述例

10．22　ブランチ命令

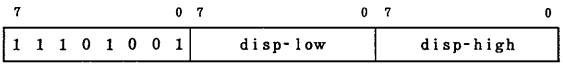
10．22．1　BR（Branch）

(1) セグメント内レラティブ

① 記述形式

BR near-label

② 命令語形式



③ バイト数

3

④ クロック数

7

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

PC←PC＋disp

現在のPCの値に16ビット・ディスプレースメントを加えた値をロードします．

ブランチ・アドレスがこの命令の置かれているセグメント内にあれば，アセンブラが自動的にこの命令を生成します．

⑦ フラグの動作

なし

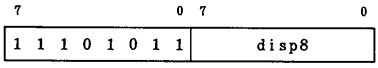
⑧ 記述例

⑵ ショート・レラティブ

① 記述形式

BR short-label

② 命令語形式



③ バイト数

2

④ クロック数

7

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

PC←PC＋ext-disp8

現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をPCにコードします．

ブランチ・アドレスが，この命令の置かれているセグメント内でかつ±127バイト内にあれば，アセンブラが自動的にこの命令を生成します．

⑦ フラグの動作

なし

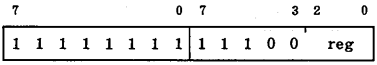
⑧ 記述例

(3) セグメント内レジスタ

① 記述形式

BR regptr16

② 命令語形式



③ バイト数

2

④ クロック数

7

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

PC← regptr16

オペランドで指定される16ビット・レジスタの内容をPCにロードします.

この命令の置かれているセグメント内の任意のアドレスにブランチできます.

⑦ フラグの動作

なし

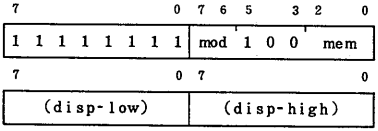
⑧ 記述例

(4) セグメント内メモリ

① 記述形式

BR memptr16

② 命令語形式



③ バイト数

2／3／4

④ クロック数

13：奇数アドレス

11：偶数アドレス

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

PC←(memptr16)

オペランドでアドレスされる16ビット・メモリの内容がPCにロードされます.

この命令の置かれているセグメント内の任意のアドレスにブランチできます.

⑦ フラグの動作

なし

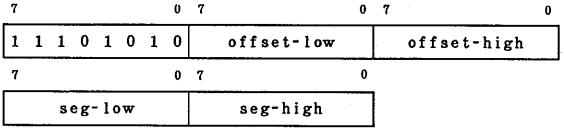
⑧ 記述例

(5) セグメント外ダイレクト

① 記述形式

BR far-label

② 命令語形式



③ バイト数

5

④ クロック数

7

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

PS←seg

PC←offset

PCに命令の2，3バイト目の16ビット・オフセット・データを，PSに命令の4，5バイト目の16ビット・セグメント・データをロードします.

任意のセグメントの任意のアドレスにブランチ可能です.

⑦ フラグの動作

なし

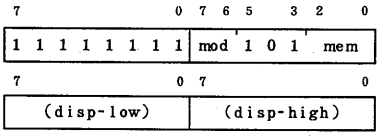
⑧ 記述例

(6) セグメント外メモリ

① 記述形式

BR memptr32

② 命令語形式



③ バイト数

2／3／4

④ クロック数

17：奇数アドレス

13：偶数アドレス

⑤ 16ビット・ワード転送回数

2

⑥ 機 能

PS←(memptr32＋3, memptr32＋2)

PC←(memptr32＋1, memptr32)

オペランドによってアドレスされる32ビット・メモリの上位2バイトをPSに，下位2バイトをPCにロードします．

任意のセグメントの任意のアドレスにブランチ可能です．

⑦ フラグの動作

なし

⑧ 記述例

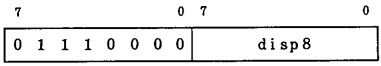
10.23　条件付きブランチ命令

10.23.1　BV(Branch if Overflow)

① 記述形式

BV　short-label

② 命令語形式



③ バイト数

2

④ クロック数

V＝1のとき　6

V＝0のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

V＝1のとき　PC←PC＋ext-disp8

Vフラグが1のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

⑦ フラグの動作

なし

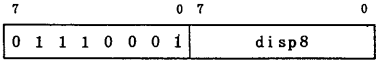
⑧ 記述例

10.23.2 BNV (Branch if Not Overflow)

① 記述形式

BNV short-label

② 命令語形式



③ バイト数

2

④ クロック数

V＝0のとき 6

V＝1のとき 3

⑤ 16ビット・ワード転送回数

なし

⑥ 機 能

V＝0のとき PC←PC＋ext-disp8

Vフラグが0のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

⑦ フラグの動作

なし

⑧ 記述例

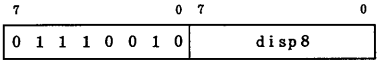
10．23．3　BC／BL（Branch if Carry／Lower）

① 記述形式

BC　short-label

BL　short-label

② 命令語形式



③ バイト数

2

④ クロック数

CY＝1のとき　6

CY＝0のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

CY＝1のとき　PC←PC＋ext-disp8

CYフラグが1のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

⑦ フラグの動作

なし

⑧ 記述例

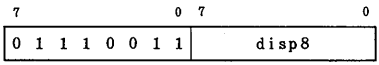
10．23．4　BNC／BNL（Branch if Not Carry／Not Lower）

① 記述形式

BNC　short-label

BNL　short-label

② 命令語形式



③ バイト数

2

④ クロック数

CY＝0のとき　6

CY＝1のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

CY＝0のとき　PC←PC＋ext-disp8

CYフラグが0のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

⑦ フラグの動作

なし

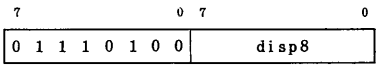
⑧ 記述例

10．23．5　BE／BZ（Branch if Equal／Zero）

① 記述形式

BE short-label or BZ short-label

② 命令語形式



③ バイト数

2

④ クロック数

Z＝1のとき　6

Z＝0のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

Z＝1のとき　PC←PC＋ext-disp8

Zフラグが1のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

⑦ フラグの動作

なし

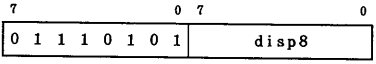
⑧ 記述例

10．23．6　BNE／BNZ（Branch if Not Equal／Not Zero）

① 記述形式

BNE short-label or BNZ short-label

② 命令語形式



③ バイト数

2

④ クロック数

Z＝0のとき　6

Z＝1のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

Z＝0のとき　PC←PC＋ext-disp8

Zフラグが0のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

⑦ フラグの動作

なし

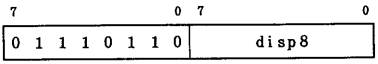
⑧ 記述例

10．23．7　BNH（Branch if Not Higher）

① 記述形式

BNH　short-label

② 命令語形式



③ バイト数

2

④ クロック数

CY∨Z＝1のとき　6

CY∨Z＝0のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

CY∨Z＝1のとき　PC←PC＋ext-disp8

CYとZフラグの論理和が1のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

⑦ フラグの動作

なし

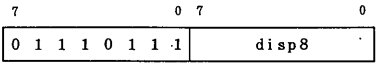
⑧ 記述例

10．23．8　BH（Branch if Higher）

① 記述形式

BH short-label

② 命令語形式



③ バイト数

2

④ クロック数

CY∨Z＝0のとき　6

CY∨Z＝1のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

CY∨Z＝0のとき　PC←PC＋ext-disp8

CYとZフラグの論理和が0のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

⑦ フラグの動作

なし

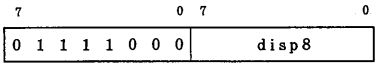
⑧ 記述例

10．23．9　BN(Branch if Negative)

① 記述形式

BN　short-label

② 命令語形式



③ バイト数

2

④ クロック数

S＝1のとき　6

S＝0のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

S＝1のとき　PC←PC＋ext-disp8

Sフラグが1のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

⑦ フラグの動作

なし

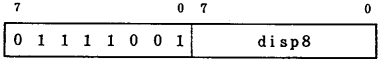
⑧ 記述例

10．23．10　BP（Branch if Positive）

① 記述形式

BP　short-label

② 命令語形式



③ バイト数

2

④ クロック数

S＝0のとき　6

S＝1のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

S＝0のとき　PC←PC＋ext-disp8

Sフラグが0のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

⑦ フラグの動作

なし

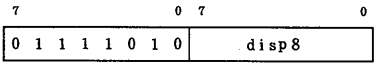
⑧ 記述例

10．23．11　BPE(Branch if Parity Even)

① 記述形式

BPE short-label

② 命令語形式



③ バイト数

2

④ クロック数

P＝1のとき　6

P＝0のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

P＝1のとき　PC←PC＋ext-disp8

Pフラグが1のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

⑦ フラグの動作

なし

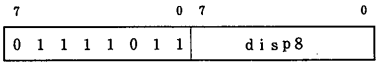
⑧ 記述例

10．23．12　BPO(Branch if Parity Odd)

① 記述形式

BPO short-label

② 命令語形式



③ バイト数

2

④ クロック数

P＝0のとき　6

P＝1のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

P＝0のとき　PC←PC＋ext-disp8

Pフラグが0のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

⑦ フラグの動作

なし

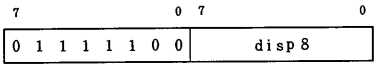
⑧ 記述例

10．23．13　BLT(Branch if Less Than)

① 記述形式

BLT　short-label

② 命令語形式



③ バイト数

2

④ クロック数

S∀V＝1のとき　6

S∀V＝0のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

S∀V＝1のとき　PC←PC＋ext-disp8

SとVフラグの排他的論理和が1のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

条件不成立のときは次の命令に進みます．

⑦ フラグの動作

なし

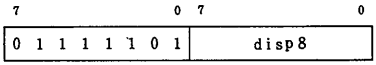
⑧ 記述例

10．23．14　BGE(Branch if Greater Than or Equal)

① 記述形式

BGE short-label

② 命令語形式



③ バイト数

2

④ クロック数

S⊻V＝0のとき　6

S⊻V＝1のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

S⊻V＝0のとき　PC←PC＋ext-disp8

SとVフラグの排他的論理和が0のとき，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

条件不成立のときは次の命令に進みます．

⑦ フラグの動作

なし

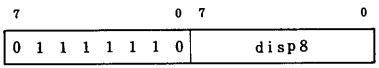
⑧ 記述例

10．23．15　BLE(Branch if Less Than or Equal)

① 記述形式

BLE　short-label

② 命令語形式



③ バイト数

2

④ クロック数

(S⊻V)∨Z＝1のとき　6

(S⊻V)∨Z＝0のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

(S⊻V)∨Z＝1のとき　PC←PC＋ext-disp8

SとVフラグの排他的論理和をとった結果とZフラグの論理和が1のとき，現在のPCの値に8ビット・ディスプレースメント(実際にはサイン拡張された16ビット)を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

条件不成立のときは次の命令に進みます．

⑦ フラグの動作

なし

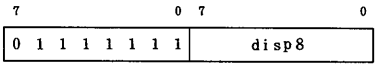
⑧ 記述例

10．23．16　BGT(Branch if Greater Than)

① 記述形式

BGT　short-label

② 命令語形式



③ バイト数

2

④ クロック数

(S∀V)∨Z＝0のとき　6

(S∀V)∨Z＝1のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

(S∀V)∨Z＝0のとき　PC←PC＋ext-disp8

SとVフラグの排他的論理和をとった結果とZフラグの論理和が0のとき，現在のPCの値に8ビット・ディスプレースメント(実際にはサイン拡張された16ビット)を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

条件不成立のときは次の命令に進みます．

⑦ フラグの動作

なし

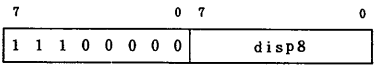
⑧ 記述例

10．23．17　DBNZNE(Decrement and Branch if Not Zero and Not Equal)

① 記述形式

DBNZNE　short-label

② 命令語形式



③ バイト数

2

④ クロック数

CW≒0でZ＝0のとき　6

上記以外のとき　　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

CW←CW−1

CW≒0でZ＝0のとき　PC←PC＋ext-disp8

16ビット・レジスタCWをデクリメントしたあと，CWが0でなくZフラグがクリアされていれば，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

条件不成立のときは次の命令に進みます．

⑦ フラグの動作

なし

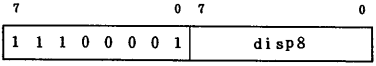
⑧ 記述例

10．23．18　DBNZE(Decrement and Branch if Not Zero and Equal)

① 記述形式

DBNZE short-label

② 命令語形式



③ バイト数

2

④ クロック数

CW≒0でZ＝1のとき　6

上記以外のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

CW←CW－1

CW≒0でZ＝1のとき　PC←PC＋ext-disp8

16ビット・レジスタCWをデクリメント（－1）し，その結果CWが0でなく，かつZフラグがセットされていれば，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

条件不成立のときは次の命令に進みます．

⑦ フラグの動作

なし

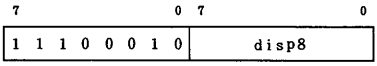
⑧ 記述例

10．23．19　DBNZ（Decrement and Branch if Not Zero）

① 記述形式

DBNZ　short-label

② 命令語形式



③ バイト数

2

④ クロック数

CW≒0　とき　6

CW＝0のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

CW←CW－1

CW≒0のとき　PC←PC＋ext-disp8

16ビット・レジスタCWをデクリメント（－1）し，その結果CWが0でなければ，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

条件不成立のときは次の命令に進みます．

⑦ フラグの動作

なし

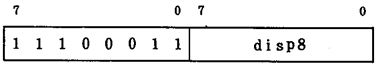
⑧ 記述例

10．23．20　BCWZ（Branch if CW equals Zero）

① 記述形式

BCWZ　short-label

② 命令語形式



③ バイト数

2

④ クロック数

CW＝0のとき　6

CW≒0のとき　3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

CW＝0のとき　PC←PC＋ext-disp8

16ビット・レジスタCWが0ならば，現在のPCの値に8ビット・ディスプレースメント（実際にはサイン拡張された16ビット）を加えた値をロードします．

この命令の置かれているセグメント内でかつ，±127バイトのアドレス内にブランチできます．

条件不成立のときは次の命令に進みます．

⑦ フラグの動作

なし

⑧ 記述例

10.24 割り込み命令

10.24.1 BRK(Break)

(1) ベクタ3

① 記述形式

BRK 3

② 命令語形式

7							0
1	1	0	0	1	1	0	0

③ バイト数

1

④ クロック数

24：奇数アドレス

18：偶数アドレス

⑤ 16ビット・ワード転送回数

5

⑥ 機 能

TA←(00DH,00CH)

TC←(00FH,00EH)

SP←SP−2,(SP+1,SP)←PSW

IE←0,BRK←0

SP←SP−2,(SP+1,SP)←PS

PS←TC

SP←SP−2,(SP+1,SP)←PC

PC←TA

PSW,PS,PCをスタックに退避し,IEとBRKフラグをリセット(0)します.

次に割り込みベクタ・テーブルのベクタ3の下位2バイトをPCに,上位2バイトをPSにロードします.

⑦ フラグの動作

IE	BRK
0	0

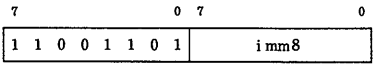
⑧ 記述例

⑵ イミーディエト・データ

① 記述形式

BRK imm8 (≒3)

② 命令語形式



③ バイト数

2

④ クロック数

24：奇数アドレス

18：偶数アドレス

⑤ 16ビット・ワード転送回数

5

⑥ 機 能

TA←(imm8×4+1, imm8×4)

TC←(imm8×4+3, imm8×4+2)

SP←SP−2, (SP+1, SP)←PSW

IE←0, BRK←0

SP←SP−2, (SP+1, SP)←PS

PS←TC

SP←SP−2, (SP+1, SP)←PC

PC←TA

PSW, PS, PCをスタックに退避し, IEとBRKフラグをリセット(0)します.

次に, 8ビット・イミーディエト・データで指定される割り込みベクタ・テーブル(4バイト)の下位2バイトをPCに, 上位2バイトをPSにロードします.

⑦ フラグの動作

IE	BRK
0	0

⑧ 記述例

10．24．2　BRKV(Break if Overflow)

① 記述形式

BRKV（オペランドなし）

② 命令語形式

7							0
1	1	0	0	1	1	1	0

③ バイト数

1

④ クロック数

V＝1のとき　26：奇数アドレス

　　20：偶数アドレス

V＝0のとき　8

⑤ 16ビット・ワード転送回数

5

⑥ 機　能

V＝1のとき　TA←(011H,010H)

　　TC←(013H,012H)

　　SP←SP−2,(SP+1,SP)←PSW

　　IE←0,BRK←0

　　SP←SP−2,(SP+1,SP)←PS

　　PS←TC

　　SP←SP−2,(SP+1,SP)←PC

　　PC←TA

Vフラグがセットされていれば，PSW,PS,PCをスタックに退避し，IEとBRKフラグをリセット(0)します．次に割り込みベクタ・テーブルのベクタ4の下位2バイトをPCに，上位2バイトをPSにロードします．

Vフラグがリセットされていれば次の命令に進みます．

⑦ フラグの動作

IE	BRK
0	0

⑧ 記述例

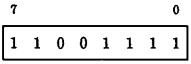
BRKV

10．24．3　RETI（Return from Interrupt）

① 記述形式

RETI（オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

19：奇数アドレス

13：偶数アドレス

⑤ 16ビット・ワード転送回数

3

⑥ 機　能

PC←（SP＋1，SP）

PS←（SP＋3，SP＋2）

PSW←（SP＋5，SP＋4）

SP←SP＋6

PC，PS，PSWにスタックの内容をリストアします．割り込み処理から戻るときに用います．

⑦ フラグの動作

V	DIR	IE	BRK	S	Z	AC	P	CY
R	R	R	R	R	R	R	R	R

⑧ 記述例

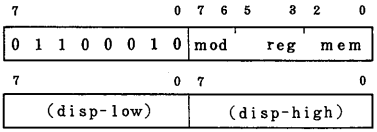
RETI

10．24．4　CHKIND（Check Index）

① 記述形式

CHKIND reg16,mem32

② 命令語形式



③ バイト数

2／3／4

④ クロック数

割り込み条件成立のとき　30－32：奇数アドレス

24－26：偶数アドレス

割り込み条件不成立のとき　14：奇数アドレス

12：偶数アドレス

⑤ 16ビット・ワード転送回数

割り込み条件成立のとき　7

割り込み条件不成立のとき　2

⑥ 機　能

（mem32）＞reg16 または（mem32＋2）＜reg16のとき

TA←（015H，014H）

TC←（017H，016H）

SP←SP－2，（SP＋1，SP）←PSW

IE←0，BRK←0

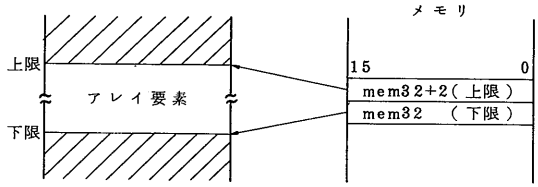
SP←SP－2，（SP＋1，SP）←PS[注]

PS←TC

SP←SP－2，（SP＋1，SP）←PC[注]

PC←TA

注　CHKIND命令の先頭番地．



アレイ・タイプのデータ構造において，要素を指定するインデクス値が定義域内にあるか否かをチェックするための命令です．インデクスが定義域を越える場合には，BRK 5を起動します．

定義域値は，あらかじめメモリ中の2ワード（1ワード目は下限値，2ワード目は上限値）にセットしておきます．

インデクス値は，アレイ操作プログラムが使用しているレジスタ（任意の16ビット・レジスタ）を対象とします．

⑦　フラグの動作

割り込み条件成立のとき

IE	BRK
0	0

割り込み条件不成立のとき

なし

⑧　記述例

10．25　CPU制御命令

10．25．1　HALT（Halt）

① 記述形式

HALT（オペランドなし）

② 命令語形式

7							0
1	1	1	1	0	1	0	0

③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

ホールト状態にします．

ホールト状態は次の3つの要因によって解除されます．

・リセット入力

・マスカブル割り込み要求入力

・ノンマスカブル割り込み要求入力

⑦ フラグの動作

なし

⑧ 記述例

HALT

10．25．2　POLL（Poll and wait）

① 記述形式

POLL （オペランドなし）

② 命令語形式

③ バイト数

1

④ クロック数

2＋2×n　　n：$\overline{\text{CPBUSY}}$端子サンプリング回数

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

・コプロセッサ接続時

$\overline{\text{CPBUSY}}$端子がインアクティブ（ハイ・レベル）になるまでCPUをウエイト状態にします．

・コプロセッサ非接続時

コプロセッサ不在割り込み（ベクタ130）を発生します．このときスタックに退避するアドレスは，この命令の先頭アドレスです．

⑦ フラグの動作

なし

⑧ 記述例

POLL

⑨ 注　意

POLL命令の直前にBUSLOCK命令を置くことはできません．

10．25．3　DI（Disable Interrupt）

① 記述形式

DI （オペランドなし）

② 命令語形式

7							0
1	1	1	1	1	0	1	0

③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

IE←0

IEフラグをリセットし，外部マスカブル割り込み入力（INT）を禁止します．

この命令によって外部ノンマスカブル割り込み入力（NMI）およびソフトウエア割り込み命令は禁止されません．

⑦ フラグの動作

IE
0

⑧ 記述例

DI

10．25．4　EI（Enable Interrupt）

① 記述形式

EI （オペランドなし）

② 命令語形式

7							0
1	1	1	1	1	0	1	1

③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

IE←1

IEフラグをセットし，外部マスカブル割り込み入力（INT）を許可します．

ただし，実際に割り込み許可状態になるのはEIに続く1命令を実行したあとです．

⑦ フラグの動作

IE
1

⑧ 記述例

EI

10．25．5　BUSLOCK（Bus Lock Prefix）

① 記述形式

BUSLOCK （オペランドなし）

② 命令語形式

7							0
1	1	1	1	0	0	0	0

③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

V53に対して,この命令に続く1命令を実行している間バス・ロック信号($\overline{\text{BUSLOCK}}$)を出力します.

リピート・プリフィクスの付いたブロック処理命令に対してこの命令が用いられれば,ブロック処理が終了するまで$\overline{\text{BUSLOCK}}$信号が出力され続けます.

⑦ フラグの動作

なし

⑧ 記述例

```
BUSLOCK
REP
MOVBKB
```

⑨ 注　意

・BUSLOCK命令をPOLL命令の直前に置くことはできません.

・この命令と次の命令との間では，ハードウエア割り込み（マスカブル割り込み，ノンマスカブル割り込み）要求およびシングル・ステップ・ブレークは受け付けられません.

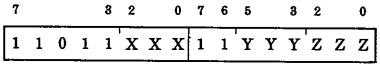
10．25．6　FPO1（Floating Point Operation 1）

(1) register

① 記述形式

FPO1 fp-op

② 命令語形式



③ バイト数

2

④ クロック数

定義できません

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

外部に接続される浮動小数点演算用コプロセッサの動作を指定する命令です．CPUがこの命令をフェッチしたとき自身でも何もせず，演算処理はコプロセッサにまかせます．

⑦ フラグの動作

なし

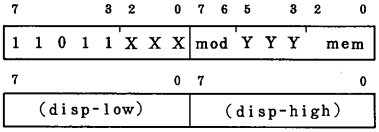
⑧ 記述例

⑵ メモリ

① 記述形式

FPO1 fp-op,mem

② 命令語形式



③ バイト数

2／3／4

④ クロック数

定義できません

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

data bus←(mem)

外部に接続される浮動小数点演算用コプロセッサの動作を指定する命令です．ＣＰＵがこの命令をフェッチしたとき演算処理はコプロセッサにまかせ，必要に応じてコプロセッサに対する補助的な処理（実効アドレス計算，物理アドレス生成，そしてメモリ・リード・サイクルの起動）のみを行います．

ＣＰＵは，ＣＰＵが起動するメモリ・リード・サイクルにおいて，データ・バス上のリード・データは一切取り込みません．

⑦ フラグの動作

なし

⑧ 記述例

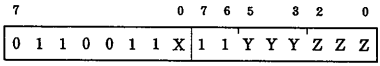
10．25．7　FPO2（Floating Point Operation 2）

(1) register

① 記述形式

FPO2 fp-op

② 命令語形式



③ バイト数

2

④ クロック数

定義できません

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

外部に接続される浮動小数点演算用コプロセッサの動作を指定する命令です．CPUがこの命令をフェッチしたとき自身では何もせず，演算処理はコプロセッサにまかせます．

⑦ フラグの動作

なし

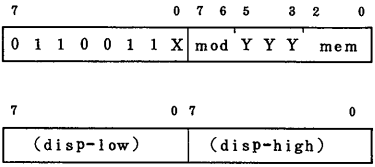
⑧ 記述例

(2) メモリ

① 記述形式

FPO2 fp-op , mem

② 命令語形式



③ バイト数

2／3／4

④ クロック数

定義できません

⑤ 16ビット・ワード転送回数

1

⑥ 機 能

data bus←(mem)

外部に接続される浮動小数点演算用コプロセッサの動作を指定する命令です．CPUがこの命令をフェッチしたとき演算処理はコプロセッサにまかせ，必要に応じてコプロセッサに対する補助的な処理（実効アドレス計算，物理アドレス生成，そしてメモリ・リード・サイクルの起動）のみを行います．

⑦ フラグの動作

なし

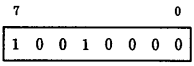
⑧ 記述例

10．25．8　NOP（No Operation）

① 記述形式

NOP （オペランドなし）

② 命令語形式



③ バイト数

1

④ クロック数

3

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

何もせず3クロック費やします.

⑦ フラグの動作

なし

⑧ 記述例

NOP

10.26　セグメント・オーバライド・プリフィクス

これは本来の命令とは異なりますが，機械語を生成します．

① 記述形式

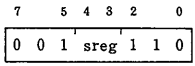
DS0：

DS1：

PS：

SS：

② 命令語形式



③ バイト数

1

④ クロック数

2

⑤ 16ビット・ワード転送回数

なし

⑥ 機　能

セグメント・オーバライド可能なメモリ・オペランドをアクセスする際に，オペランドに付記してそのとき使用されるセグメント・レジスタを指定します．この命令を直接記述しなくとも，ASSUME（アセンブラ擬似命令）を使用することによって，セグメント・オーバライド指定をアセンブラに行わせることができます．

⑦ フラグの動作

なし

⑧ 記述例

```
REP MOVBKB BYTE_VAR , SS:BYTE_VAR
```

⑨ 注　意

この命令と次の命令との間では，ハードウエア割り込み（マスカブル割り込み，ノンマスカブル割り込み）要求およびシングル・ステップ・ブレークは受け付けられません．

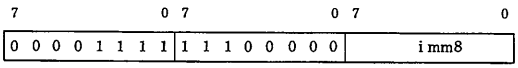
10．27　拡張アドレス・モード専用命令

10．27．1　BRKXA(Break Extend Address Mode)

① 記述形式

BRKXA　imm8

② 命令語形式



③ バイト数

3

④ クロック数

12

⑤ 16ビット・ワード転送回数

2

⑥ 機能

temp1←(imm8×4＋1，imm8×4)

temp2←(imm8×4＋3，imm8×4＋2)

XA←1

PC←temp1

PS←temp2

拡張アドレス・モードを設定します。

命令により指定された割り込みベクタ・テーブルのエントリに格納されているアドレスに制御を移し，XAMレジスタ(内部I／Oポート：FF80H)のビット0(XAフラグ)を1にセットします.

通常アドレス・モードで実行したときは，通常アドレス・モードのアドレス上のベクタ・テーブルを読み込んだあと，拡張アドレス・モードに移り，先に読み込んだアドレスにジャンプします.

拡張アドレス・モードで実行したときは，拡張アドレス・モードのアドレス上のベクタ・テーブルを読み込んだあと，このベクタ・テーブルのアドレスにジャンプします.

割り込みアクノリッジ・サイクルは実行されません.

スタックへのPC，PS，PSWの退避を行いません.

⑦ フラグの動作

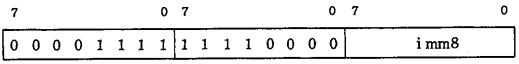
なし

10．27．2 RETXA(Return from Extend Address Mode)

① 記述形式

RETXA imm8

② 命令語形式



③ バイト数

3

④ クロック数

12

⑤ 16ビット・ワード転送回数

2

⑥ 機能

temp1 ← (imm8 × 4 + 1, imm8 × 4)

temp2 ← (imm8 × 4 + 3, imm8 × 4 + 2)

XA ← 0

PC ← temp1

PS ← temp2

拡張アドレス・モードを解除する命令です.

命令により指定された割り込みベクタ・テーブルのエントリに格納されているアドレスに制御を移し，XAMレジスタ（内部I／Oポート：FF80H）のビット0（XAフラグ）を0にリセットします.

通常アドレス・モードで実行したときは，通常アドレス・モードのアドレス上のベクタ・テーブルを読み込んだあと，このベクタ・テーブルのアドレスにジャンプします.

拡張アドレス・モードで実行したときは，拡張アドレス・モードのアドレス上のベクタ・テーブルを読み込んだあと，通常アドレス・モードに移り，先に読み込んだアドレスにジャンプします.

割り込みアクノリッジ・サイクルは実行されません.

スタックへのPC，PS，PSWの退避を行いません.

⑦ フラグ動作

なし

［メ　　モ］

付録A　命令ニモニック・リスト

ADD	Add
ADD4S	Add Nibble String
ADDC	Add with Carry
ADJ4A	Adjust Nibble Add
ADJ4S	Adjust Nibble Subtract
ADJBA	Adjust Byte Add
ADJBS	Adjust Byte Subtract
AND	And
BC	Branch if Carry
BCWZ	Branch if CW equals Zero
BE	Branch if Equal
BGE	Branch if Greater Than or Equal
BGT	Branch if Greater Than
BH	Branch if Higher
BL	Branch if Lower
BLE	Branch if Less Than or Equal
BLT	Branch if Less Than
BN	Branch if Negative
BNC	Branch if Not Carry
BNE	Branch if Not Equal
BNH	Branch if Not Higher
BNL	Branch if Not Lower
BNV	Branch if Not Overflow
BNZ	Branch if Not Zero
BP	Branch if Positive
BPE	Branch if Parity Even
BPO	Branch if Parity Odd
BR	Branch
BRK	Break
BRKXA	Break Extend Address Mode
BRKV	Break if Overflow
BUSLOCK	Bus Lock
BV	Branch if Owerflow
BZ	Branch if Zero

CALL	Call
CHKIND	Check Index
CLR1	Clear Bit
CMP	Compare
CMP4S	Compare Nibble String
CMPBK	Compare Block
CMPBKB	Compare Block Byte
CMPBKW	Compare Block Word
CMPM	Compare Multiple
CMPMB	Compare Multiple Byte
CMPMW	Compare Multiple Word
CVTBD	Convert Binary to Decimal
CVTBW	Convert Byte to Word
CVTDB	Convert Decimal to Binary
CVTWL	Convert Word to Long Word
DBNZ	Decrement and Branch if Not Zero
DBNZE	Decrement and Branch if Not Zero and Equal
DBNZNE	Decrement and Branch if Not Zero and Not Equal
DEC	Decrement
DI	Disable Interrupt
DISPOSE	Dispose
DIV	Divide Signed
DIVU	Divide Unsigned
DS0:	Data Segment 0
DS1:	Data Segment 1
EI	Enable Interrupt
EXT	Extract Bit Fieid
FPO1	Floating Point Operation 1
FPO2	Floating Point Operation 2
HALT	Halt
IN	Input
INC	Increment
INM	Input Multiple
INS	Insert Bit Field
LDEA	Load Effective Address
LDM	Load Multiple
LDMB	Load Multiple Byte
LDMW	Load Multiple Word

MOV	Move
MOVBK	Move Block
MOVBKB	Move Block Byte
MOVBKW	Move Block Word
MUL	Multiply Signed
MULU	Multiply Unsigned
NEG	Negate
NOP	No Operation
NOT	Not
NOT1	Not Bit
OR	Or
OUT	Output
OUTM	Output Multiple
POLL	Poll and wait
POP	Pop
PREPARE	Prepare
PS:	Program Segment
PUSH	Push
REP	Repeat Block Operation
REPC	Repeat Block Operation While Carry
REPE	Repeat Block Operation While Equal
REPNC	Repeat Block Operation While Not Carry
REPNE	Repeat Block Operation While Not Equal
REPNZ	Repeat Block Operation While Not Zero
REPZ	Repeat Block Operation While Zero
RET	Return
RETXA	Return from Extend Address Mode
RETI	Return from Interrupt
ROL	Rotate Left
ROL4	Rotate Left Nibble
ROLC	Rotate Left with Carry
ROR	Rotate Right
ROR4	Rotate Right Nibble
RORC	Rotate Right with Carry
SET1	Set Bit
SHL	Shift Left Logical
SHR	Shift Right Logical
SHRA	Shift Right Arithmetic

SS:	Stack Segment
STM	Store Multiple
STMB	Store Multiple Byte
STMW	Store Multiple Word
SUB	Subtract
SUB4S	Subtract Nibble String
SUBC	Subtract with Carry
TEST	Test
TEST1	Test Bit
TRANS	Translate
TRANSB	Translate Byte
XCH	Exchange
XOR	Exclusive-Or

付録B　命令索引（アルファベット順）

［メ　モ］

付録C　レジスタ索引(アルファベット順)

略　号	名　称	ユニット	ページ
BADR	バンク・アドレス・レジスタ	DMAU(37)	197
BNKR0 -BNKR3	バンク・レジスタ	DMAU(37)	198
BRC	ボー・レート・カウンタ	SCU	254
BSEL	バンク選択レジスタ	DMAU(37)	198
DBA	DMAベース・アドレス・レジスタ	DMAU(71)	181
DBC	DMAベース・カウント・レジスタ	DMAU(71)	180
DCA	DMAカレント・アドレス・レジスタ	DMAU(71)	181
DCBP	クリア・バイト・ポインタF／F	DMAU(37)	197
DCC	DMAカレント・カウント・レジスタ	DMAU(71)	180
DCH	DMAチャネル・レジスタ	DMAU(71)	179
DCM	クリア・マスク・レジスタ	DMAU(37)	197
DDC	DMAデバイス・コントロール・レジスタ	DMAU(71)	182
DICM	DMAイニシャライズ・コマンド・レジスタ	DMAU(71)	178
DMD	DMAモード・コントロール・レジスタ	DMAU(71)	185
DMK	DMAマスク・レジスタ	DMAU(71)	187
DRCA0 -DRCA3	リード・カレント・アドレス・レジスタ	DMAU(37)	191
DRCC0 -DRCC3	リード・カレント・カウント・レジスタ	DMAU(37)	191
DRST	リード・ステータス・レジスタ	DMAU(37)	192
DST	DMAステータス・レジスタ	DMAU(71)	187
DULA	内蔵ペリフェラル・リロケーション・レジスタ	システムI／O	127
DWAM	ライト・オール・マスク・レジスタ	DMAU(37)	195
DWBCA0 -DWBCA3	ライト・ベース＆カレント・アドレス・レジスタ	DMAU(37)	191
DWBCC0 -DWBCC3	ライト・ベース＆カレント・カウント・レジスタ	DMAU(37)	191
DWC	ライト・コマンド・レジスタ	DMAU(37)	193
DWM	ライト・モード・レジスタ	DMAU(37)	196
DWRQ	ライト・リクエスト・レジスタ	DMAU(37)	194
DWSM	ライト・シングル・マスク・レジスタ	DMAU(37)	194
IIS	割り込みインサービス・レジスタ	ICU	163

略　号	名　称	ユニット	ページ
IIW1	割り込みイニシャライズ・ワード1レジスタ	ICU	150
IIW2	割り込みイニシャライズ・ワード2レジスタ	ICU	152
IIW3	割り込みイニシャライズ・ワード3レジスタ	ICU	153
IIW4	割り込みイニシャライズ・ワード4レジスタ	ICU	154
IMDW	割り込みモード・ワード・レジスタ	ICU	160
IMKW	割り込みマスク・ワード・レジスタ	ICU	156
IPFW	割り込みプライオリティ，フィニッシュ・ワード・レジスタ	ICU	157
IPOL	割り込みポーリング・レジスタ	ICU	162
IRQ	割り込み要求レジスタ	ICU	163
IULA	内蔵ペリフェラル・リロケーション・レジスタ	システムI／O	127
OPHA	内蔵ペリフェラル・リロケーション・レジスタ	システムI／O	127
OPSEL	内蔵ペリフェラル選択レジタ	システムI／O	126
PGR1 −PGR64	ページ・レジスタ	CPU	71
RFC	リフレッシュ・コントロール・レジスタ	REFU	142
SBCR	スタンバイ・コントロール・レジスタ	CG	270
SCM	シリアル・コマンド・レジスタ	SCU	249
SCTL	システム・コントロール・レジスタ	システムI／O	125
SIMK	シリアル割り込みマスク・レジスタ	SCU	253
SMD	シリアル・モード・レジスタ	SCU	247
SRB	シリアル受信データ・バッファ	SCU	246
SST	シリアル・ステータス・レジスタ	SCU	251
STB	シリアル送信データ・バッファ	SCU	246
SULA	内蔵ペリフェラル・リロケーション・レジスタ	システムI／O	127
TCKS	タイマ・クロック選択レジスタ	TCU	220
TCTn	タイマ・カウント・レジスタ	TCU	222
TMD	タイマ・モード・レジスタ	TCU	218,223 224
TSTn	タイマ・ステータス・レジスタ	TCU	225
TULA	内蔵ペリフェラル・リロケーション・レジスタ	システムI／O	127
WAC	プログラマブル・ウエイト・メモリ・アドレス・コントロール・レジスタ	WCU	136
WCY0	プログラマブル・ウエイト・サイクル数設定レジスタ0	WCU	135
WCY1	プログラマブル・ウエイト・サイクル数設定レジスタ1	WCU	135
WCY2	プログラマブル・ウエイト・サイクル数設定レジスタ2	WCU	137
WCY3	プログラマブル・ウエイト・サイクル数設定レジスタ3	WCU	138
WCY4	プログラマブル・ウエイト・サイクル数設定レジスタ4	WCU	139
WMB0	プログラマブル・ウエイト・メモリ領域設定レジスタ0	WCU	133

略　号	名　称	ユニット	ページ
WMB1	プログラマブル・ウエイト・メモリ領域設定レジスタ1	WCU	136
XAM	拡張アドレス・モード・レジスタ	CPU	72

備考　DMAU(71)：μPD71071モード時のDMAU
　　DMAU(37)：μPD71037モード時のDMAU

［メ　モ］

付録D　V40, V50と置き換える際の注意点

V53は，V40，V50と比べるといくつかの動作上の相違点があります．ここでは，V40，V50のシステムをV53に置き換える際に，特に注意すべき点について説明します．

D．1　CPU動作の相違点

D．1．1　割り込み時にスタックへ退避するアドレス

(1)　V40，V50の場合

常に割り込みのかかった命令の次の命令の先頭アドレスを示すPC，PSの値をスタックに退避します．

(2)　V53の場合

外部割り込みでは，V40，V50と同じ動作をします．

ソフトウエア割り込みのうちの以下の割り込みでは，割り込みのかかった命令の先頭アドレスを示すPC，PSの値をスタックに退避します．

- ディバイド・エラー割り込み
- CHKIND境界エラー割り込み
- 未定義命令コード・トラップ
- コプロセッサ不在割り込み
- μPD72291エラー割り込み

このほかのソフトウエア割り込みでは，割り込みのかかった命令の次の命令の先頭アドレスを示すPC，PSの値をスタックに退避します．

D．1．2　割り込み受け付けタイミング(INT入力)

RETI命令またはPOP　PSW命令によりDI状態からEI状態に移る場合，その命令と次の命令の間における割り込み受け付け動作が，次のように異なります．

- V40，V50の場合 …　INT入力による割り込みは受け付けられる
- V53の場合　　…　INT入力による割り込みは受け付けられない

D．1．3　NMI多重割り込み

(1)　V40，V50の場合

NMI処理ルーチン中であっても，NMI割り込みを受け付けます（NMI多重割り込み可能）．

(2)　V53の場合

NMI処理ルーチン中は，RETI命令を実行するまで，NMI処理のネスティングは行われません．NMI処理中の$\overline{\text{NMI}}$要求は保留され，その処理が終了してから，保留されていたNMIの処理を開始します．ただし，NMI処理中のスタンバイ状態は，$\overline{\text{NMI}}$入力で解除できます．その場合，NMI割り込みを発生し，NMI処理ルーチンに入ります．

D．1．4　例外割り込みシーケンスの実行順序

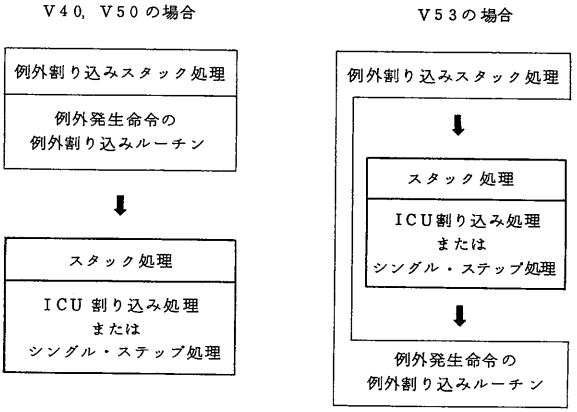
例外割り込みが多重に発生した場合，例外割り込みシーケンスの実行順序が次のように異なります．

◇EI命令→例外発生命令実行時のICU入力

◇例外発生命令実行時のシングル・ステップ割り込み

対象となる例外は，次の命令です．

- BRK3
- BRKV
- CHKIND 境界オーバ
- コプロセッサ不在
- BRK imm8



この相違点は，割り込み処理ルーチンを実行する順番のみの違いであり，それぞれの処理ルーチンの動作結果は同じになります．

D.2　命令実行動作の相違点

命　　令	V40，V50の場合	V53の場合
プリフィクス	・最大3つまで記憶可能です． ・同一種類のプリフィクスを2つ以上つけた場合は，最も命令に近いプリフィクスのみが有効で，それ以外は無効です．	・異種類のプリフィクスならばいくつでも覚えられます（ただし，全部で3種類です）． ・同一種類のプリフィクスは，2つ以上つけないでください．
DIV （符号付き除算）	商が80H（バイト演算時）または8000H（ワード演算時）になると，DIV命令ディバイド・エラーを発生します．	商が80H（バイト演算時）または8000H（ワード演算時）になるときも，正常に演算します．
ADJ4A，注1 ADJ4S	9AH≦AL≦9FHの場合は，AC=0のときのみ上位4ビットの補正を行います． 〔命令の機能〕 ・AL∧0FH＞9またはAC=1のとき，ALレジスタの下位4ビットを補正 ・AL＞9FHまたはCY=1のとき，ALレジスタの上位4ビットを補正 ・99H＜AL＜A0Hかつ AC=0のとき，ALレジスタの上位4ビットを補正	9AH≦AL≦9FHの場合は，常に上位4ビットの補正を行います． 〔命令の機能〕 ・AL∧0FH＞9またはAC=1のとき，ALレジスタの下位4ビットを補正 ・AL＞99HまたはCY=1のとき，ALレジスタの上位4ビットを補正
CVTBD， CVTDB	命令コードの2バイト目が0AHでなくても正常に計算します．	命令コードの2バイト目が0AH以外ならば，誤動作します．
PUSH SP POP SP	（SP－2）← SP－2 注2 SP←（SP）	（SP－2）← SP SP←（SP）

備考　プリフィクスの種類は次の3種類です．

・リピート　… REPC, REPNC, REPZ, REPNZ

・セグメント・オーバライド … PS:, DS0:, DS1:, SS:

・バス・ロック　… BUSLOCK

注1．この相違点は，10進データ以外の演算結果に対して補正を行った場合に，違いとして現われます（10進データどうしの演算においては，9AH≦AL≦9FHでかつAC=1にはなりません）．

2．SPレジスタに対してPUSH命令，POP命令の組み合わせを実行すると，SPレジスタには命令実行前の値から（－2）された値が格納されます．

命　　令	V40，V50の場合	V53の場合
POP　R	POPサイクルを7回起動します. 〔命令の機能〕 IY←(SP+1, SP) IX←(SP+3, SP+2) BP←(SP+5, SP+4) BW←(SP+9, SP+8) DW←(SP+11, SP+10) CW←(SP+13, SP+12) AW←(SP+15, SP+14) SP←SP+16	POPサイクルを8回起動します. 〔命令の機能〕 IY←(SP+1, SP) IX←(SP+3, SP+2) BP←(SP+5, SP+4) temp←(SP+7, SP+6)注 BW←(SP+9, SP+8) DW←(SP+11, SP+10) CW←(SP+13, SP+12) DW←(SP+15, SP+14) SP←SP+16
POLL	$\overline{\text{POLL}}$端子がロウ・レベルになるまでウエイトします.	・コプロセッサ不在時 コプロセッサ不在例外を発生します. ・μPD72291接続時 $\overline{\text{CPBUSY}}$端子がロウ・レベルになるまでウエイトします.
PREPARE	制限事項なし	直後に実行できる命令に制限があります.
BRKEM	あり	なし
BRKXA	なし	あり
RETXA	なし	あり

注　この動作は，まったく意味をもちません.

D.3　未定義命令

・V40，V50の場合

未定義命令コードをフェッチした場合の動作は，保証できません．

・V53の場合

表D－1に示す未定義コード一覧表のコードに対し，ベクタ番号122の未定義命令コード・トラップを発生します．この表にない未定義コードに対しては，正常動作は期待できません．

表D－1　未定義コード一覧表

1バイト目	2バイト目
0FH	00H－0FH，21H，23H－25H，27H， 29H，2BH－30H，32H，34H－38H， 3AH，3CH－CEH，D0H－DFH， E1H－EFH，F1H－FFH
63H	－
8DH	C0H－FFH
C4H	C0H－FFH
C5H	C0H－FFH
C0H，C1H，D0H，D1H	30H－37H，70H－77H，B0H－B7H，F0H－F7H
D2H，D3H	30H－37H，70H－77H，B0H－B7H，F0H－F7H
D6H	－
F6H，F7H	08H－0FH，48H－4FH，88H－8FH，C8H－CFH
FEH，FFH	38H－3FH，78H－7FH，B8H－BFH，F8H－FFH

D.4　内蔵周辺デバイスの相違点

（1／2）

内蔵周辺デバイス	項　　目	V40，V50	V53
WCU	設定可能ウエイト数	0－3ウエイト	0－7ウエイト
	システムI／O領域	WCY1，WCY2，WMB	WCY0，WCY1，WCY2，WCY3，WCY4，WMB0，WMB1
	メモリ空間の3分割機能	1Mバイト空間を分割可能	16Mバイト空間と特定の1Mバイト空間を分割可能
REFU	リフレッシュ・アドレス	A0－A8（9ビット）	A0－A15（16ビット）
ICU	割り込み入力	・INTP1－INTP7（TOUT1，SCUからの割り込み要求とマルチプレクス） ・TOUT0	INTP0－INTP7
	INTP0－INTP7端子のプルアップ抵抗	あり	なし
	スレーブ・アドレス	A10－A8	A2－A0
DMAU	μPD71037動作モード	なし	あり
	アドレス	20ビット	24ビット
	DMARQ3，$\overline{\text{DMAAK3}}$	TxD，RxDとマルチプレクス	あり
TCU	TOUT0	ICUへ接続	あり
	TOUT1	$\overline{\text{INTAK}}$，SRDYとマルチプレクス	あり
	TCTL0	なし	あり
	TCTL1	なし	あり
SCU	制御端子	SRDY（INTAK，TOUT1とマルチプレクス）	SINT，RxRDY
	専用ボー・レート・ジェネレータ	なし	あり
	RS－232－C制御端子	なし	あり（$\overline{\text{RTS}}$，$\overline{\text{CTS}}$，$\overline{\text{DTR}}$，$\overline{\text{DSR}}$）
CG	PCLKOUT	なし	あり
	スタンバイ機能	HALTモード	・HALTモード，STOPモード ・インストラクション・サイクル時間変更可能
	CLKOUTの出力	X1の立ち下がりに同期	X1の立ち上がりに同期

内蔵周辺デバイス	項　　目	V40, V50	V53
BIU	出力ラッチ機能	なし	あり
	I／Oアクセスのリカバリ・サイクル	なし	6クロック
	CLKOUTに対するレディ入力	READYは非同期入力可能	$\overline{\text{READY}}$は同期化が必要

［メ　モ］

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］ V53 ユーザーズ・マニュアル

（U10109JJ8V0UM00（第8版））

［お名前など］（さしつかえのない範囲で）

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

1．ご評価（各欄に○をご記入ください）

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

2．わかりやすい所（第　章，第　章，第　章，第　章，その他 ）

理由 ［ ］

3．わかりにくい所（第　章，第　章，第　章，第　章，その他 ）

理由 ［ ］

4．ご意見，ご要望

5．このドキュメントをお届けしたのは

NEC 販売員，特約店販売員，NEC半導体ソリューション技術本部員，
その他（ ）

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

NEC 半導体インフォメーションセンター
FAX：(044)548-7900

キリトリ

お問い合わせは，最寄りのNECへ

【営業関係お問い合わせ先】

半導体第一販売事業部 半導体第二販売事業部 半導体第三販売事業部	〒108-01 東京都港区芝五丁目7番1号（NEC本社ビル）	東京	(03)3454-1111 （大代表）
中部支社 半導体販売部	〒460 名古屋市中区錦一丁目17番1号（NEC中部ビル）	名古屋	(052)222-2170
関西支社 半導体第一販売部 関西支社 半導体第二販売部 関西支社 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	宇都宮支店 宇都宮	(028)621-2281	富山支店 富山	(0764)31-8461
東北支社 仙台	(022)261-5511	小山支店 小山	(0285)24-5011	三重支店 津	(0592)25-7341
岩手支店 盛岡	(0196)51-4344	長野支社 長野	(026)235-1444	京都支社 京都	(075)344-7824
山形支店 山形	(0236)23-5511	松本支店 松本	(0263)35-1666	神戸支社 神戸	(078)333-3854
郡山支店 郡山	(0249)23-5511	上諏訪支店 諏訪	(0266)53-5350	中国支社 広島	(082)242-5504
いわき支店 いわき	(0246)21-5511	甲府支店 甲府	(0552)24-4141	鳥取支店 鳥取	(0857)27-5311
長岡支店 長岡	(0258)36-2155	埼玉支社 大宮	(048)641-1411	岡山支店 岡山	(086)225-4455
土浦支店 土浦	(0298)23-6161	立川支社 立川	(0425)26-5981	四国支社 高松	(0878)36-1200
水戸支店 水戸	(0292)26-1717	千葉支社 千葉	(043)238-8116	新居浜支店 新居浜	(0897)32-5001
神奈川支社 横浜	(045)324-5511	静岡支社 静岡	(054)255-2211	松山支店 松山	(089)945-4111
群馬支店 高崎	(0273)26-1255	北陸支社 金沢	(0762)23-1621	九州支社 福岡	(092)271-7700
太田支店 太田	(0276)46-4011	福井支店 福井	(0776)22-1866	北九州支店 北九州	(093)541-2887

【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)548-8890
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）