１０．割り込みルーチン（割り込みコントローラ，タイマ）

割り込み（Interrupt）とは，あるプログラムを実行中にＣＰＵが何等かの信号を受けた時に，その実行を中止し，全く別のプログラムで処理を行い，その処理が終了すると再び元のプログラムに実行を移すような動作をいい，制御プログラムで基本となるプログラム処理である。割り込みには「ソフトウェア割り込み（Software Interrupt）」と「ハードウェア割り込み（Hardware Interrupt）」があり，いずれもサブルーチンと同様な記述形式のプログラムである。サブルーチンの最後が「RET」で終わるのに対し，割り込みルーチンでは「IRET」で終了することを除けば，割り込みプログラムは普通のサブルーチンとほとんど変わらない（サブルーチンでは「CALL」および「RET」命令によってスタックとの間で「CS」と「IP」の値を退避，復帰するのに対し，割り込みでは「FL」，「CS」および「IP」の値を退避，復帰する）。特に「内部割り込み（Internal Interrupt）」はサブルーチンと同様にメインプログラム中に「INT #」という形式で記述されるので，割り込みプログラム移行処理が異なる以外はサブルーチンと全く同じであると考えてよい。しかし，ハードウェア割り込みではかなり様子が異なり，ＣＰＵの処理状態だけでなく，その割り込みを制御する「プログラマブル割り込みコントローラ８２５９（PIC：Programmable Interrupt Controller）」と呼ばれるＬＳＩの動作が重要であり，これを理解しなければ，割り込みを理解したとはいえない。ハードウェア割り込みは通常，ＴＴＬレベルの負論理信号をＣＰＵに対して送信することで行われ，その信号をコンピュータ内部で発生する場合を「内部ハードウェア割り込み（Internal Hardware Interrupt）」，コンピュータ外部で発生する場合を「外部ハードウェア割り込み（External Hardware Interrupt）」という。本システムの制御プログラムでは内部および外部ハードウェア割り込みを行っており，前者は「プログラマブル・インターバルタイマー８２５３（PIT：Programmable Interval Timer）」と呼ばれるＬＳＩ（このチップは卒業研究で行っているパルス幅変調「PWM」制御のパルスパターンを発生するためにも使用している）を利用し，後者は商用電源に同期した割り込み信号をＴＴＬ回路で作成している。この章では，これらの割り込みについて関連する全てのＬＳＩおよび処理状態について順次説明する。

［ソフトウェア割り込み］

ＰＣ９８０１のＤＯＳには多くの内部割り込みが用意されているが，本プログラムで使用している「キャラクタのディスプレイ表示」用割り込みを説明する。この割り込みはデータ領域に記述した「エスケープシーケンスコード（Escape Sequence Code）」によってカーソルを指定位置に移動，色の指定，文字の表示を行い，BASIC の「LOCATE」，「COLOR」および「PRINT」と同等の処理を行うことができる。

次に示すプログラム（「EX101.C」と「EX101A.ASM」）は画面の１０行（Ｙ方向）２０桁（Ｘ方向）に緑で「Power Electronics」を，１５行４０桁に黄色で「パワーエレク

トロニクス」を表示するものである。これはあらかじめＤＯＳシステムに用意されているサブルーチンを利用したものであり，どういう処理を行うかを指定されたレジスタに値を格納して「INT 0DCH」を実行する。

```
/* EX101.C */
main()
{
disp1();
}
```

その形式は

```
MOV AH,1
MOV CL,10H
MOV DX,OFFSET DGROUP:ﾗﾍﾞﾙ
INT 0DCH
```

```
;EX101A.ASM
DGROUP   group  mdata
         segment word public 'data'
         assume  ds:DGROUP
MOJI1    DB 1BH,"[32m",1BH,"[10;20H"
         DB "POWER ELECTRONICS","$"
MOJI2    DB 1BH,"[33m",1BH,"[15;40H"
         DB "パワーエレクトロニクス"
         DB 1BH,"[37m","$"
mdata    ends

_TEXT    segment byte public 'CODE'
         assume  cs:_TEXT,ds:DGROUP
         public  _disp1
_disp1   proc near
         MOV  AH,1
         MOV  CL,10H
         MOV  DX,OFFSET DGROUP:MOJI1
         INT  0DCH

         MOV  AH,1
         MOV  CL,10H
         MOV  DX,OFFSET DGROUP:MOJI2
         INT  0DCH
         RET
_disp1   endp
_TEXT    ends
         end
```

である。ラベルはデータ領域に「DB：（Define Byte）バイト文字列」で定義したエスケープ文字列を含むデータである。エスケープ文字のコードデータは「1BH」であり，これに続く文字列が操作を表す。文字列は全て「“」で囲んで記述し，文字列の終りを「$」で指示する。エスケープデータに続く文字列の意味は次のとおりである。

```
1BH,“[Cm”
```

これは「色」の指定を行うコードであり，「Ｃ」の値で決定する。対応する色は

```
30：黒    31：赤    32：緑    33：黄
34：青    35：紫    36：水色  37：白
```

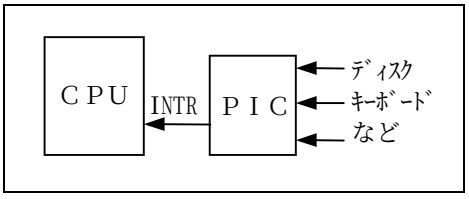
であり，省略時は白となる。また，数値が４０台はその色のリバースとなる。カーソル移動は

```
1BH,“[Y;XH”
```

の形式で指示し，Ｙ行Ｘ列へ移動する。ただし，ディスプレイ上では左上隅が Y=1,X=1 で右下隅が Y=25,X=80 となっている（BASIC の「LOCATE」命令では左上隅が Y=0,X=0 で右下隅が Y=24,X=79 である）。

［ハードウェア割り込み］

上記のソフトウェア割り込みでは「INT #」のようにソースプログラム中に命令を記述するが，ハードウェア割り込みでは「割り込み信号（Interrupt Signal）」と呼ばれる信号がＣＰＵに対して送信された時点で「割り込みプログラム」へ移行しなければならない。したがって，ソースプログラム中に「割り込みプログラム」の開始アドレスを記述することはできない。ハードウェア割り込みは例えばハードディスクやマウス，キーボード入力など様々な割り込みがあり，これらに対応した割り込みプログラムの実行開始アドレスはあらかじめ設定されている「割り込みベクタ（Interrupt Vector）」と呼ば

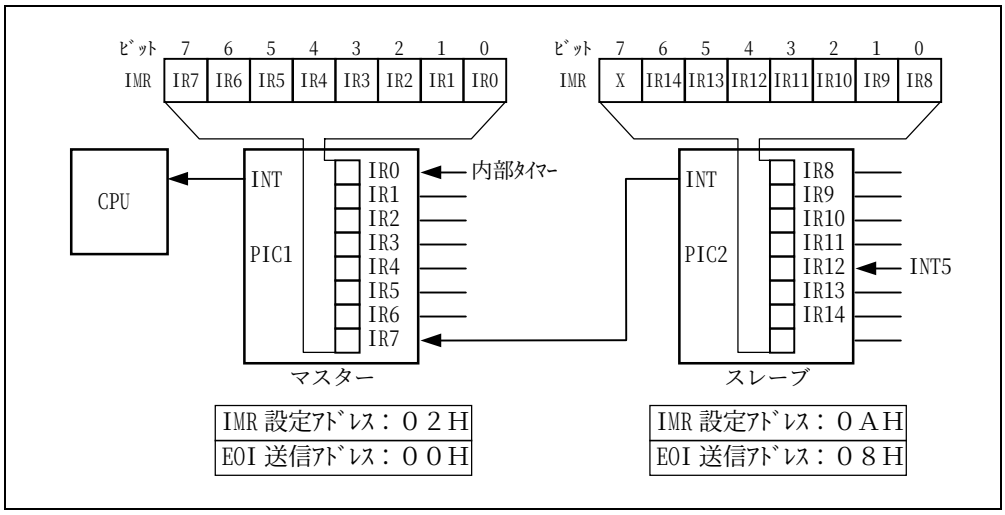


れるシステムＲＡＭ領域に保存する。幾つかの割り込みに対してそれらの優先順位のモード設定や割り込み要求のマスク（Mask：許可，不許可を行うこと）およびベクタアドレスの指定などはＰＩＣ（８２５９）が行い，ＣＰＵをコントロールしている。すべての割り込み信号はまずＰＩＣへ送られ，設定状態に応じてＣＰＵに割り込み信号（INTR：Interrupt Request）を送信する。本システムでは１つあるいは２つのハードウェア割り込みを使用しており，１つの内部ハードウェア割り込み信号はコンピュータ内部にあるＰＩＴ（８２５３：内部タイマー）で発生し，もう１つの外部ハードウェア割り込み信号は別に作成した回路によって発生し，共にＰＩＣへ送られる。

［プログラマブル割り込みコントローラ（ＰＩＣ）８２５９］

ＰＣ９８０１（１６ビット）コンピュータのＰＩＣの構成は下図のようになっており，２つのＰＩＣがカスケード（Cascade）接続されている。ＰＩＣには８つの割り込み入力端子があり，優先順位がある。ＰＩＣ２の割り込み出力信号はＰＩＣ１の入力となっており，ＰＩＣ１をマスタ（Master），ＰＩＣ２をスレーブ（Slave）と呼んでいる。ＰＩＣにはそれぞれの信号の許可，不許可を設定する「割り込みマスクレジスタ（IMR：Interrupt Mask Register）」があり，対応するビットを「０」に設定することで「許可」を行う。２つのＰＩＣの入力構成は表のようであり，本システムでは割り込み要求番号「IR0」と「IR12」を使用するので，これらの設定について説明する。

割り込みレベル	デバイス	割り込み要求番号	割り込み名
0	ＰＩＣ１	IR0	内部ﾀｲﾏｰ(8253)
1	マスタ	IR1	ｷｰﾎﾞｰﾄﾞ
2		IR2	CRTC(V)ﾃﾞｨｽﾌﾟﾚｲ
3		IR3	INT0
4		IR4	RS-232C
5		IR5	INT1
6		IR6	INT2
7		IR7	PIC2 ｽﾚｰﾌﾞ
8	ＰＩＣ２	IR8	ｾﾝﾄﾛ･ﾌﾟﾘﾝﾀ
9	スレーブ	IR9	INT3
10		IR10	INT4
11		IR11	8inch FD
12		IR12	INT5
13		IR13	INT6
14		IR14	8087



内部タイマーによる割り込みを行う場合には，マスターの「IR0」を許可する必要があるので，マスターIMR レジスタにその設定データを出力する。

```
MOV AL,11111110B              ;11111110B=FEH
OUT 02H,AL
```

この設定は１度行えば，書き直されるまでこの状態を保持する。マスターＰＩＣは IR0 に割り込み信号が送られてくると，マスク状態を調べ許可されていればＣＰＵに対して割り込み信号を送信し，その割り込みに対応した割り込みプログラムの先頭アドレスを格納した割り込みベクタアドレスを出力する。ＣＰＵが割り込み信号を受けると割り込みベクタから割り込みプログラムのアドレスを取り込み，割り込み処理を行い，処理終了後，次の割り込み信号を待つ。このとき，ＰＩＣはＣＰＵへの割り込み信号送信によって次の IR0 割り込み信号を受け付けないようにセットされるので，これをリセットするためにマスターＰＩＣに対して，割り込み終了（EOI：End Of Interrupt）データ「20H」を次のように EOI 送信アドレスへ出力する。

```
MOV AL,20H
OUT 00H,AL
```

割り込みを順次行う場合には，このデータを毎回出力する必要がある。

一方，外部ハードウェア割り込みは「INT5」を使用しており，この場合にはスレーブおよびマスターを通して割り込み信号が送信されるので，両方に対してマスクの設定および EOI データ送信を行う。すなわち，IMR 設定は

```
MOV AL,01111111B              ;01111111B=7FH
OUT 02H,AL
MOV AL,11101111B              ;11101111B=EFH
OUT 0AH,AL
```

となる。ただし，マスターにおいて「IR0」と「IR7」を共に許可する時は「7EH」のデータを出力する。EOI データ送信は次のようである。

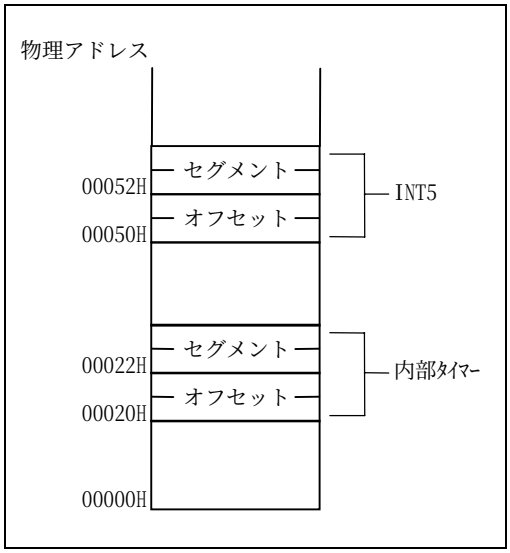
```
MOV AL,20H
OUT 00H,AL
OUT 08H,AL
```

ＰＩＣのＩ／Ｏアドレスは８ビットデコード（８ビットでアドレスを指定）である。
なお，このようにユーザーが作成したアプリケーションプログラムによってＰＩＣのＩＭＲを変更した時には，ＤＯＳシステム設定を変えることになるので，プログラムによってあらかじめこれらレジスタの内容を保存しておき，アプリケーションプログラム終了時に元の設定に戻すようにしないと，ＤＯＳ上でキーが受け付けない等の暴走の可能性がある。

［割り込みベクタ］

ハードウェア割り込みでは，全ての割り込み要求番号に対して割り込みプログラムが記憶されているメモリの先頭物理アドレスを格納する「割り込みベクタ領域」を参照する。割り込みベクタは物理アドレスの最下位アドレス，すなわち「00000H」から割り当てられ

ている。本システムで使用する「内部タイマー（IR0）」と「INT5（IR12）」の割り込みに対するベクタ物理アドレス（全てのベクタアドレスはマニュアル参照）は図のように設定されており，それぞれ「コードセグメント」と「オフセット」を格納する２ワードで構成される。これらはＣＰＵが割り込み許可を行う前に設定しなければならない。



例えば,内部割り込みのベクタ設定は次のように行う。通常，割り込みプログラムはメインプログラムと同一セグメント「CS」内に作成する。割り込みルーチンの先頭に記述したラベルを「IROU:」とすれば右のように作成する。エクストラセグメントを使用し，これに「0000H」を入れる。「OFFSET」演算子によって割り込みプログラムの先頭番地オフセットを得る。セグメントオーバーライドによる「MOV」命令で割り込みベクタへ値を格納する。プログラムのセグメントは「CS」であるので，これをベクタのセグメント領域に格納する。また，「INT5」に対しても同様に設定を行う。なお，内部割り込みによるプログラミングは後で簡単な例によって示す。

ベクタアドレスの設定

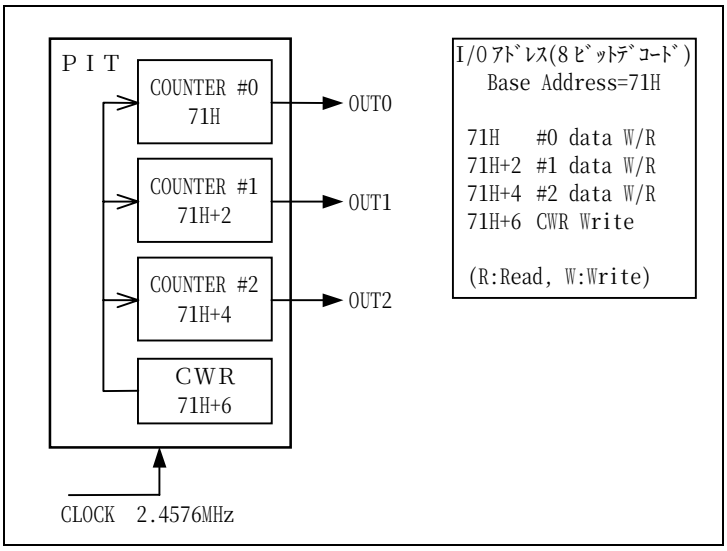
```
XOR  AX,AX
MOV  ES,AX
MOV  AX,OFFSET IROU
MOV  BX,0020H
MOV  ES:[BX],AX
MOV  AX,CS
ADD  BX,2
MOV  ES:[BX],AX
```

［プログラマブル・インターバルタイマ（ＰＩＴ）８２５３］

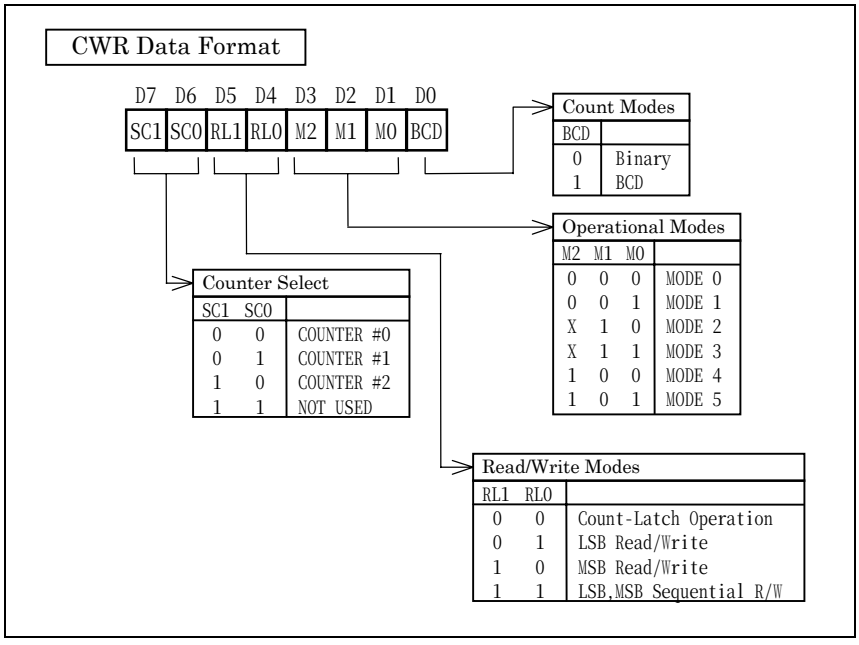
ハードウェア割り込みではＰＩＣに割り込み信号を与えなければならない。この信号はコンピュータ内部に装備されているＰＩＴ（内部タイマとよぶ）によって発生する。このタイマはクロック（Clock）と呼ばれる一定周波数の方形波信号（「０」と「１」の繰り返し信号）の周期数（クロック数）を数え，何等かの出力信号を発生するＬＳＩである。ユーザーはタイマが数えるクロック数をデータとして与え，タイマはデータを受け取った時点から数え始める。そのため，タイマはカウンタ（「数える」の意味）とも言われる。

１つのＰＩＴチップには３つの同じカウンタあり，それぞれを独立して動作させることができる。また，それぞれには「GATE」端子があり，これによってカウント状態をコントロールできる（本システムでは使用していない）。さらに，それぞれは６つの動作モードを有しており，ユーザーは目的に合ったモードでＰＩＴをカウントさせる。ここでは，モード０と呼ばれる動作モードを使用しており，このモードはカウントデータが送られた時点でカウントを開始し（出力は０「Low」），カウントが終了すると出力を１「High」

に変化するようなモードである。この出力信号がマスターＰＩＣの「IR0」へ送られる。ＰＩＣの割り込みはカウンタ出力が「０」から「１」へ変化した時点で発生する（このような割り込み信号発生をエッジトリガ（Edge Trigger）という）。内部タイマと同じＰＩＴは本制御プログラムの目的であるパルスパターン発生に用いているため（コンピュータ外部に作成している：このＰＩＴをここでは外部タイマとよぶ），その基本動作を理解しておかなければならない。



ＰＩＴを使用するにはまず，初期設定を行う。内部タイマのＩ／Ｏアドレスは８ビットデコードでベースアドレス（Base Address）すなわち最下位のＩ／Ｏアドレスは「71H」に設定されている。ＰＩＴは１個で４つのアドレスを占有し，それぞれ２アドレスづつずれている。カウンタには＃０，＃１，＃２と番号が付いており，内部タイマとしてユーザーが利用できるのは，カウンタ＃０である（他の２つはシステムが使用している）。カウンタ＃０がカウントするデータを送るＩ／Ｏアドレスは「71H」であり，また，モード設



定を行う「コントロールワードレジスタ（CWR：Control Word Register）」のアドレスは「71H+6」番地である。３つのカウンタのモード設定は，全て「CWR」へデータを書き込むことによって行われる。

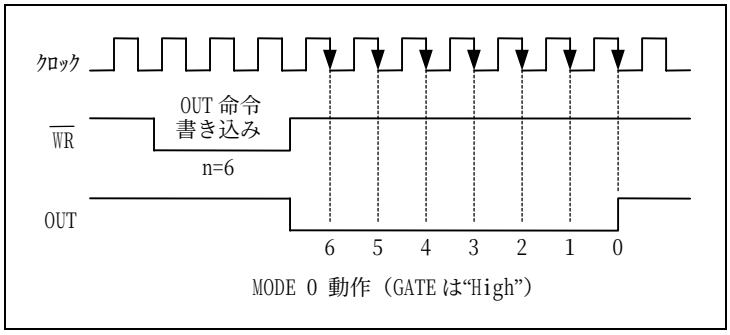
本システムでは次のように設定する。

```
Counter Select      :COUNTER #0
Read/Write Modes    :LSB,MSB Sequential R/W
Operational Modes   :MODE 0
Count Modes         :Binary
```

データフォーマットから，上記設定のデータは

００１１００００Ｂ＝３０Ｈ

となり，これを「OUT」命令で「77H」番地（CWR）へ出力すればカウンタ＃０が設定できる。なお，内部タイマのＩ／Ｏアドレスもまた８ビットデコードである。ここで「Sequential R/W」とは１６ビットのカウントデータを上位８ビットと下位８ビットに分け，まず下位８ビット，次に上位８ビットを出力することである（これは８２５３データバスが８ビットとなっているためである）。例えば，６カウントを行う場合には，次のようにプログラムし，その動作は右のようである。




```
MOV  AX,6
OUT  71H,AL
MOV  AL,AH
OUT  71H,AL
```

カウンタは２番目の「OUT」命令データを受け取った時点からカウントを開始し，指定カウントを終了するとその出力を「High」とする。８２５９およびＣＰＵが割り込みを許可していれば，この時点で割り込み処理に移行する。

上図からわかるようにカウント値が大きいほど出力の「Low」区間が長くなる。制御プログラムの１つの処理は何等かの規則に従ってこのカウント値を求めることであり，コンピュータ外部に接続したこの８２５３を組み合わせて任意の幅のパルスを作成する。外部ＰＩＴは１６ビットデコード（アドレス）としているが，モードの設定やデータ出力は８ビットの場合と同じである。なお，外部ＰＩＴのベースアドレスは「50D0H，50D1H，60D0H，60D1H」あるいは「70D0H，70D1H，80D0H，80D1H」などに設定している。

内部および外部ＰＩＴへのクロックはシステムクロックを用いており，コンピュータを10MHz（厳密には 9.8304MHz であり，通常この周波数で行う）で動作する時には，その４分周（４分の１）した周波数 2.4576MHz である。

［割り込みプログラム例］

割り込み処理を行う例として「ディスプレイ中央に０から９までを１秒間隔で順次表示する」プログラム（「EX102.C」「EX102A.ASM」）を示す。この処理内容は別として記述方法はアプリケーションプログラム作成の基本となるのでよく理解されたい。

```
/* EX102.C */
main()
{
dint();
}
```

```
;EX102A.ASM
DGROUP   group  mdata
mdata    segment word public 'data'
         assume  ds:DGROUP
CNT      DB 75
DISPD    DB 30H
PICD     DB ?
mdata    ends

_TEXT    segment byte public 'CODE'
         assume  cs:_TEXT,ds:DGROUP
         public  _dint
_dint    proc near
         CLI
;- Vector Setting -
         XOR  AX,AX
         MOV  ES,AX
         MOV  AX,OFFSET IROU
         MOV  BX,0020H
         MOV  ES:[BX],AX
         MOV  AX,CS
         ADD  BX,2
         MOV  ES:[BX],AX
;- PIC Setting -
         IN   AL,02H
         MOV  PICD,AL
         MOV  AL,0FEH
         OUT  02H,AL
;- PIT Setting -
         MOV  AL,00110000B
         OUT  71H+6,AL
;- PIT Start -
         MOV  AX,8000H
         OUT  71H,AL
         MOV  AL,AH
         OUT  71H,AL

;- Main Loop -
         STI
MLP:     IN   AL,41H
         CMP  AL,0E0H
         JNE  SKP
         JMP  KST
SKP:     MOV  CX,2000H
LP:      LOOP LP
         JMP  MLP
```

```
;- Exit -
KST:     CLI
         MOV  AL,PICD
         OUT  02H,AL
         STI
         RET

;- Interrupt Routine -
IROU:    CLI
         PUSH AX
         PUSH BX
         PUSH CX
         PUSH ES
;- PIT Start -
         MOV  AX,8000H
         OUT  71H,AL
         MOV  AL,AH
         OUT  71H,AL

;- Judge -
         DEC  CNT
         JZ   DISP
         JMP  ISKP
;- Display -
DISP:    MOV  CNT,75
         MOV  CL,DISPD
         INC  CL
         CMP  CL,39H
         JBE  DSKP
         MOV  CL,30H
DSKP:    MOV  DISPD,CL
         MOV  AX,0A000H
         MOV  ES,AX
         MOV  BX,0A0H*10+80
         MOV  ES:[BX],CL
;- Exit of INT.Rou. -
ISKP:    MOV  AL,20H
         OUT  00H,AL
         POP  ES
         POP  CX
         POP  BX
         POP  AX
         STI
         IRET
_dint    endp
_TEXT    ends
         end
```

ＰＩＴへのクロック周波数は 2.4576MHz であるから，１秒では「2457600」カウントしなければならないが，１６ビットでは「65536」カウントが限界であるから，これを分割して割り込みカウントを「2457600＝75×32768（8000H）」，すなわち「8000H」カウントごとに行い，７５回目の割り込みで１秒となるようにプログラムを行う。データ領域の「CNT」は割り込み回数の変数，「DISPD」はアスキーコードのデータ（30H～39Hまで変える），「PICD」はマスタＰＩＣの IMR データ保存用変数である。

メインルーチンではまず，ベクタ設定，ＰＩＣ設定，ＰＩＴ設定を行い，ＰＩＴのカウントを開始し，その後は無限ループを実行して割り込みを待つ。無限ループではキーボー

ドからのキー入力を調べて（後述），「STOP」キーによってこのループから抜け出し，終了するようにしている。終了時にはＰＩＣのマスク状態を元に戻して戻る。

一方，割り込みルーチンでは，まず，ＣＰＵの割り込みを不許可（CLI）にし，割り込みで使用するレジスタの値をスタックに退避（PUSH）する。「PUSH」命令は，先に説明したスタック領域に指定したレジスタの値を保存する命令である。ハードウェア割り込みではメインルーチンの何処で割り込みが発生するのかわからないため，割り込みで使用したレジスタの値を保存しておかないと，割り込みを終了してメインルーチンに戻った時に暴走する。スタックからレジスタの値を復帰する命令は「POP」である。「POP」する場合には最後に「PUSH」した値から復帰されるので（このような動作を FIFO：（Fast-In and Fast-Out）という），「PUSH」とは順番が逆になる。また，「PUSH」と「POP」の命令数は必ず一致させなければならない（一致していない時は「Stack Overflow」のエラーとなり，停止する）。レジスタ退避後，次の割り込み信号を発生させるためにＰＩＴを動作させる。割り込み処理中にＰＩＴはカウントを行うので，割り込みプログラムは次の割り込み信号が発生するまでに処理を終り，メインルーチンへ戻らなければならない。

表示は７５回に１度行うので，「CNT」を判別し，キャラクタコードを指定表示内のデータとするために判別を行う。割り込みの最後にはマスタＰＩＣに「EOI」を送信し，ＣＰＵ割り込みを許可（STI）して，「IRET」でメインルーチンへ戻る。

このような処理状態から，基本的な割り込みプログラムの構成は右図のようになる。

```
;- 割り込みルーチン -
ﾗﾍﾞﾙ:    CLI
          レジスタ退避
          （内部ﾀｲﾏ動作）

          「割り込み処理」

          PIC へ EOI 送信
          レジスタ復帰
          STI
          IRET
```

キーボードからの入力はキーボードインターフェース「８２５１」と呼ばれるＬＳＩが管理しており，そのＩ／Ｏアドレス「41H」から８ビットデータを取り込むことにより，どのキーが押されているかを調べることができる。プログラムでは頻繁にこのＬＳＩからデータを取り込まないように時間待ち「LOOP 命令」を行っている。

「Make」はキーを押下されている時のデータであり，「Break」はキーが離された時のデータである。例えば，「STOP」キーを押している間は「60H」であり，離した時（その後も何等かのキーが押されるまで）は「E0H」となる。キーとデータとの関係は次ページ参照。

上位４ビット（Make）

下位4ビット	0	1	2	3	4	5	6	7
0	ESC	Q タ	F ハ	< , ネ 、	―	.	STOP	SHIFT
1	! 1 ヌ	W テ	G キ	> 。 ル 。	／		COPY	CAPS
2	“ 2 フ	E ィ イ	H ク	? / ・ メ	7		f･1	ｶﾅ
3	# 3 ァ ア	R ス	J マ	― ロ	8		f･2	GRPH
4	$ 4 ゥ ウ	T カ	K ノ	Δ	9		f･3	CTRL
5	% 5 ェ エ	Y ン	L リ	XFER	＊		f･4	
6	& 6 ォ オ	U ナ	+ ; レ	ROLL UP	4		f･5	
7	‘ 7 ャ ヤ	I ニ	* : ケ	ROLL DOWN	5		f･6	
8	(8 ュ ユ	O ラ	} 」 」 ム	INS	6		f･7	
9	) 9 ョ ヨ	P セ	Z ッ ツ	DEL	＋		f･8	
A	0 ヲ ワ	~ @ 、	X サ	↑	1		f･9	
B	= ー ホ	{ [。 「	C ソ	←	2		f･10	
C	` ^ ヘ	ﾘﾀｰﾝ	V ヒ	→	3			
D	\ ｜ ー	A チ	B コ	↓	＝			
E	BS	S ト	N ミ	HOME CLR	0			
F	TAB	D シ	M モ	HELP	,			
	8	9	A	B	C	D	E	F

上位４ビット（Break）