[第7章]

制御，計測などに対応できるよう豊富な割り込み源を活用する

PICの割り込み処理について理解する

本章では，制御に不可欠な割り込み処理についてタイマなどを利用しながらその動作を確認します．割り込み処理はわかってしまえば思いのほか簡単で，容易に効率的な仕組みが実現できることを実感できます．また，PICに内蔵されている各機能はそれぞれ割り込み要求が出せます．本来がコントローラですから，リアルタイム処理は得意中の得意です．

7-1　割り込み処理は定められた手順による緊急対応

割り込み処理は，外部からの入力またはタイマ/カウンタのオーバフローなど内蔵された機能によるイベント発生などによって，何時発生するか予測できないが，発生したときには即時に対応しなければならない処理方法として用意されています(**図7-1**)．

前章のタイマの解説では，カウント・アップしてフラグがONになるまで，フラグのチェックを繰り返

図7-1　割り込み処理はそれぞれ割り込み源と決められた外部要因，PIC内のコンポーネント(タイマ，通信機能，入出力機能)**など，メイン・プログラムの実行と無関係に割り込み元の都合に合わせて緊急対応で処理が行われる**

して待っていました．プログラムは待つという仕事だけしかできなかったわけです．タイマをスタートして，ほかに仕事をしていても，フラグがONになったときに実行中のプログラムにフラグがONになったことを知らせる仕組みがあれば，ほかの仕事をしながら待っていることができます．このように待つだけとか処理に時間がかかる事象では，フラグがONになったときに特別なプログラムが呼び出されるようになっていれば，ほかの仕事をしていても連絡を受けると即応できます．

この仕組みが「割り込み処理」で，コンピュータのリアルタイム処理として一般に利用されている技術です．具体的な例として，タイマ2を使用して割り込み処理のプログラムを作りながら説明します．

7-2　タイマ2について

タイマ2は，**図7-2**に示すように各4ビットのプリスケーラとポストスケーラ，8ビットのカウンタ・レジスタをもち，全体で16ビットのカウンタ/タイマを構成できます．

また，タイマ2はPR2（Period Register2）と呼ばれる周期レジスタをもっているので，任意の一定間隔の周期をもつインターバル・タイマが容易に実現できます．そのため，タイマ2は単独のタイマ以外に次章で説明するモータの回転数制御や明るさ制御などの実現方法の一つであるPWMなどにも利用されま

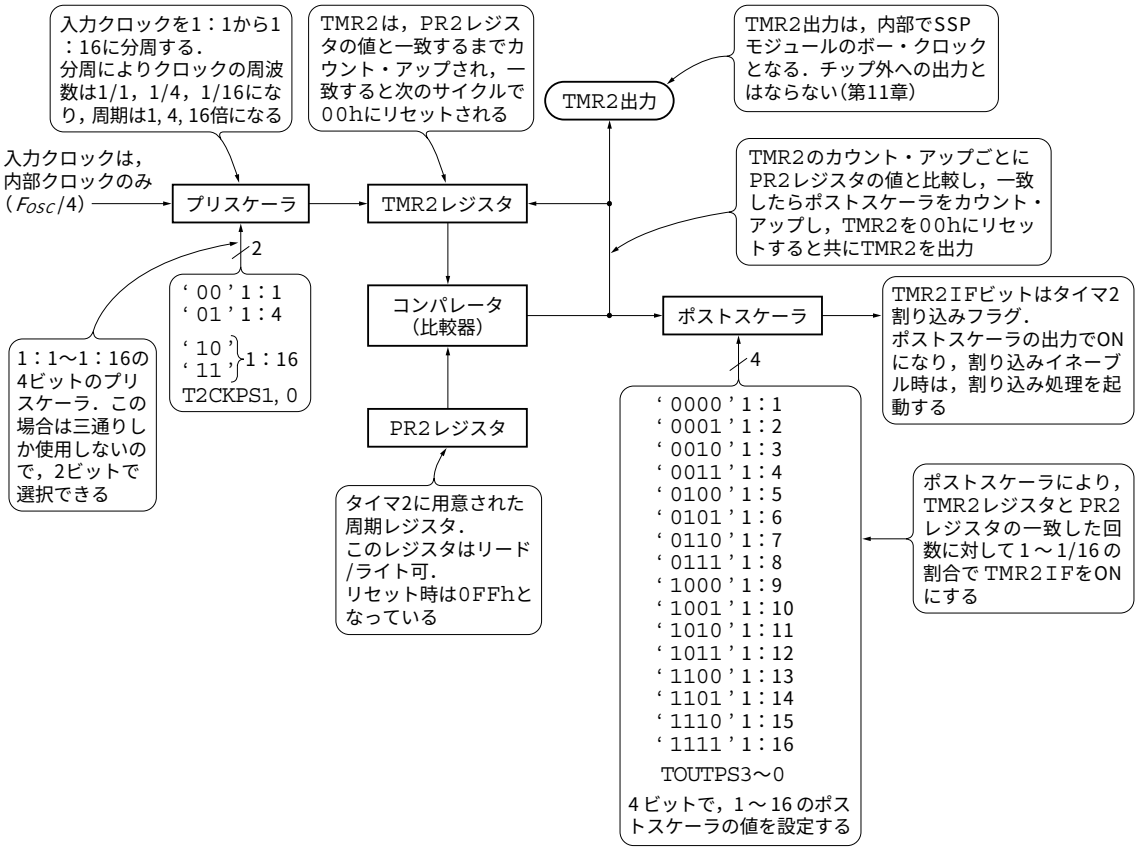

図7-2　タイマ2のブロック図
TMR2レジスタ，PR2レジスタは読み書き可，PR2レジスタにはタイマ2でカウントさせたい値をセットする．

す．PWMについては第8章にゆずり，ここではタイマ単独の特徴をまとめました．

◆8ビットのタイマ・レジスタ(TMR2)，周期レジスタ(PR2)をもつ
◆TMR2(バンク0)，PR2(バンク1)は読み書き可能
◆ソフトウェアでプログラム可能なプリスケーラ(1：1，1：4，1：16の3種)
◆ソフトウェアでプログラム可能なポストスケーラ(1：1～1：16の16種)
◆TMR2とPR2の一致によりタイマ2割り込み処理が行える

最初に，タイマ2の特徴である周期レジスタについて説明します．周期レジスタPR2は8ビットのレジスタで，カウント・アップされるTMR2レジスタとカウント・アップごとに比較され，コンパレータが一致を検出すると，次のサイクルで，ポストスケーラをカウント・アップし，TMR2レジスタをクリアします．クリアされたTMR2は再度PR2に設定された値までカウント・アップされ，クリアされることを繰り返します．

ポストスケーラの出力は，PIR1レジスタにあるTMR2IFビット(TMR2とPR2レジスタの一致割り込みフラグ・ビット)をONにします．この際，PIE1レジスタのTMR2IEビットがONであればタイマ2からこの時点で割り込みが発生します．この機能を使って，ある一定間隔の割り込みを発生させるプログラムを作り，テスト・ボードのタイマの処理を変更してみます．

割り込みを可能にするための処理は，前段にもう一つGIE，PEIEビットをONにする処理が必要です．では，割り込み処理の全体の様子を見てみます．

この印は頁右上に略語の語源の説明があります

7-3　割り込みについて処理の実施はレジスタのビットのON/OFFで制御

図7-3に16F87XAの割り込み源を示します．各割り込み源ごとに割り込み処理を行うか，否かを制御できるようになっています．これらの制御は，INTCONレジスタ，PIR1，PIR2レジスタ，PIE1，PIE2レジスタで行っています．

この図に示されるように，割り込み処理が行えるようにするためには，総元締めにあたるINTCONレジスタのGIE(Global Interrupt Enable)ビットをON(1)にしなければなりません．これにより，CPUに対する割り込みのゲートを開くことができます．このゲートを開くとタイマ0，RBポートについてはINTCONレジスタの該当する割り込みイネーブル・ビットをONにすることで割り込み処理が行えます．

その他のタイマ1，2，A-D変換などの割り込み処理については，PEIE(Peripheral Interrupt Enable)ビットをON(1)にして，INTCONレジスタで制御している以外の割り込み処理のゲートを開く必要があります．後は，個別の割り込み源について用意されている割り込みイネーブル・ビットをONにすると，該当する割り込み源に対する割り込み処理が使用できるようになります．

● フラグのON/OFFはGIE，PEIEの影響を受けない

INTCON，PIR1，PIR2の各レジスタの割り込み発生フラグは，そのフラグが発生する事象が生じたときにGIE，PEIEの設定の如何に関わらずフラグをONにします．したがって，割り込み処理ルーチンを使わず，このフラグ・ビットをクリアしておき，フラグがONになるのを待ってそれぞれの処理が完了したことを知る方法があります．それが第6章でタイマの時間待ちをした方法で，ポーリング処理といいます．

GIE，PEIEはCPUへの割り込み処理信号の通り道のゲートなので，閉じていても開いていても，各割り込み発生源の割り込みは，このゲートとは関係なく発生しています．また，各割り込みイネーブル・

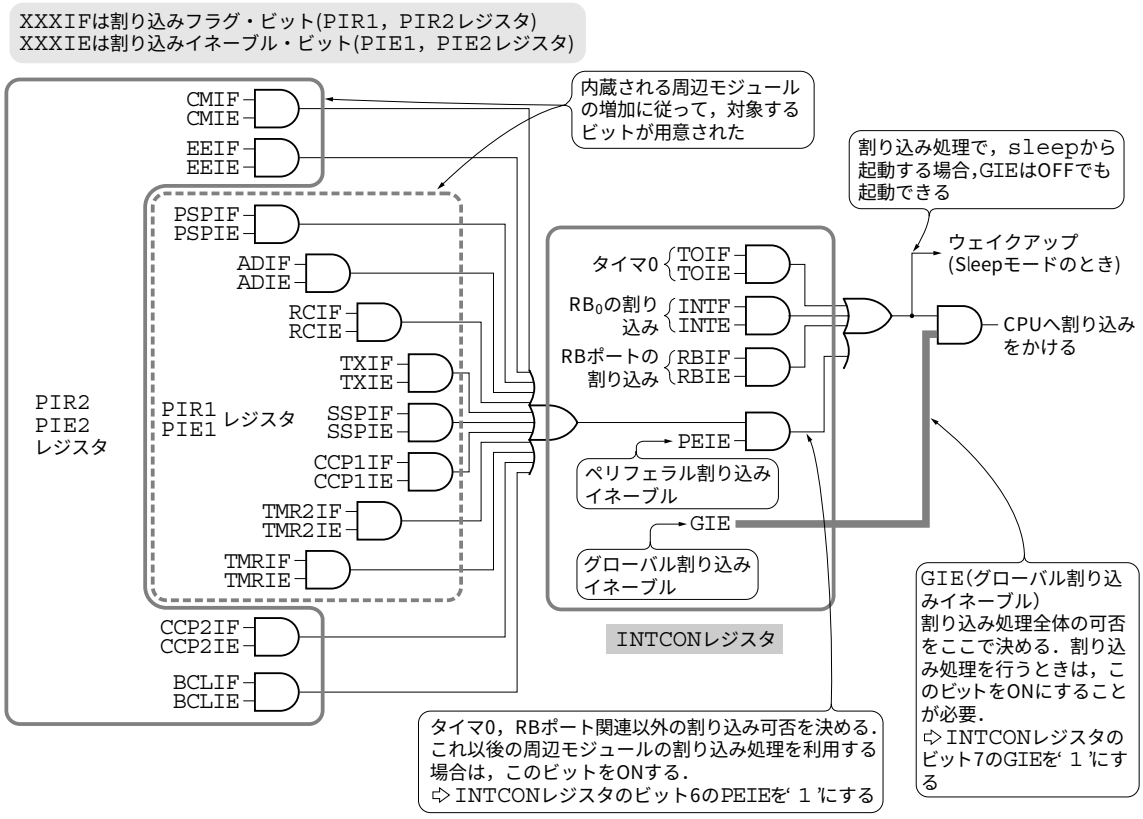

図7-3(2)　16F87XAの割り込み源

ビットも同様に割り込みフラグからCPUへの割り込み信号のゲートの役割を果たします．

● 割り込み関連のベースになるINTCONレジスタ(Appendix 7　p.321参照)

INTCONレジスタには，**図7-3**に示すようにGIE，PEIEでの割り込み処理の許可/禁止のほか，タイマ0，RB_0，RBポートに対する割り込みのイネーブルとフラグが割り当てられています．このINTCONレジスタは，バンク0，1，2，3のどのバンクでプログラムが動いていても同じINTCONレジスタのビットがアクセスできるようになっています．

● フラグ・ビットのPIR1，PIR2と割り込みイネーブル・ビットのPIE1，PIE2

図7-4に16F87XAの割り込み関連レジスタの内容を示します．PIE1/PIR1，PIE2/PIR2レジスタは，それぞれ後についた番号ごとの組み合わせで，フラグ・ビット・レジスタと割り込みイネーブル・ビット・レジスタが対になっています．基本的な使用方法は，PIR1，PIR2の該当する割り込みフラグ・ビットをクリアし，次にPIE1，PIE2の該当するレジスタの割り込みイネーブル・ビットをONにします．

以降，具体的な例としてタイマ2について説明します．ほかの周辺機能も，このタイマ例を参考にすれば，同様に扱えます．

16F87Xのデータ・シートでは，PIE2/PIR2のb_6をOFF(0)に指定しておくよう注意がありました．

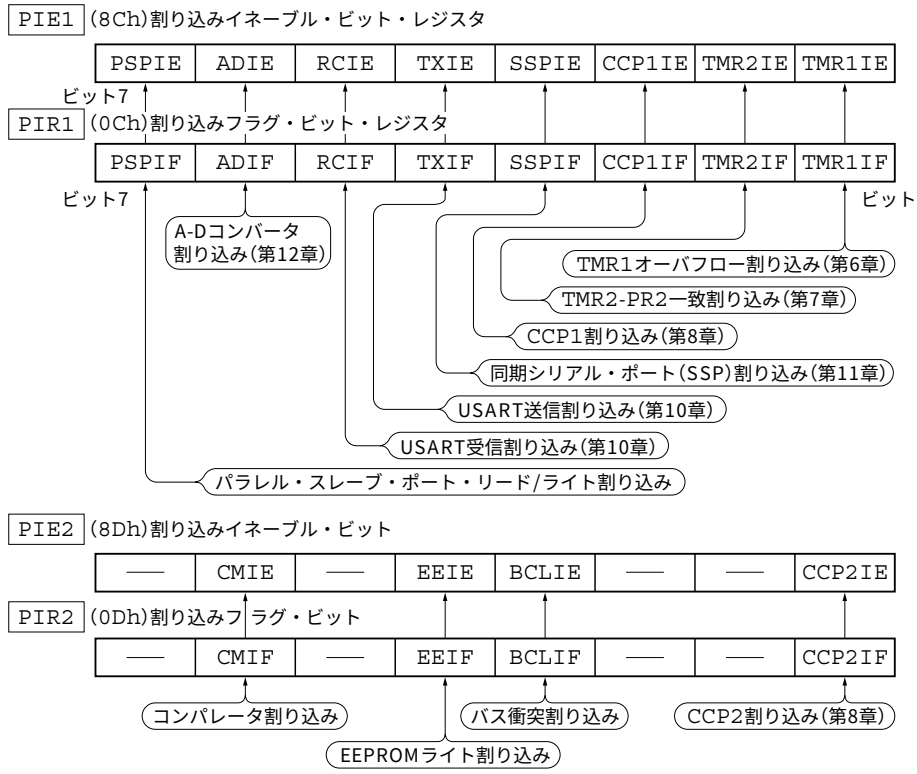

図7-4　16F87XAの割り込み関連レジスタ
INTCONレジスタはAppendix 7を参照.

16F87XÀではこのビットにコンパレータの割り込み機能(CMIE，CMIF)が割り当てられています.
16F87Xではコンパレータの機能がありません.

7-4　割り込み処理時の実行シーケンス

図7-5に示す図では，プログラムの実行中(ここでは0025h番地の命令)に割り込み要求が生じたという状況を想定しています.①で示すこの時点で，GIEは'0'になり，割り込みが禁止されます.これにより，割り込み処理中に新たな割り込み処理が発生しないようになります.

併せて，実行中のプログラムを中断し，②で示す，次に実行する命令のアドレス(0026h)をハードウェア・スタックに保存します.

その後，③で示す割り込み処理ルーチンの具体的な処理を開始します.

詳しくみていきましょう.

● 割り込み処理は，0004hから始め，割り込みフラグをクリアしRETFIEで終わる

割り込み処理ルーチンに必要な基本的な事項を**図7-6**に示します.

① レジスタ保存用のワーク・エリアの設定

変数の定義域でWレジスタやSTATUSレジスタなどの主要な変数は，以降の割り込みプログラムの中

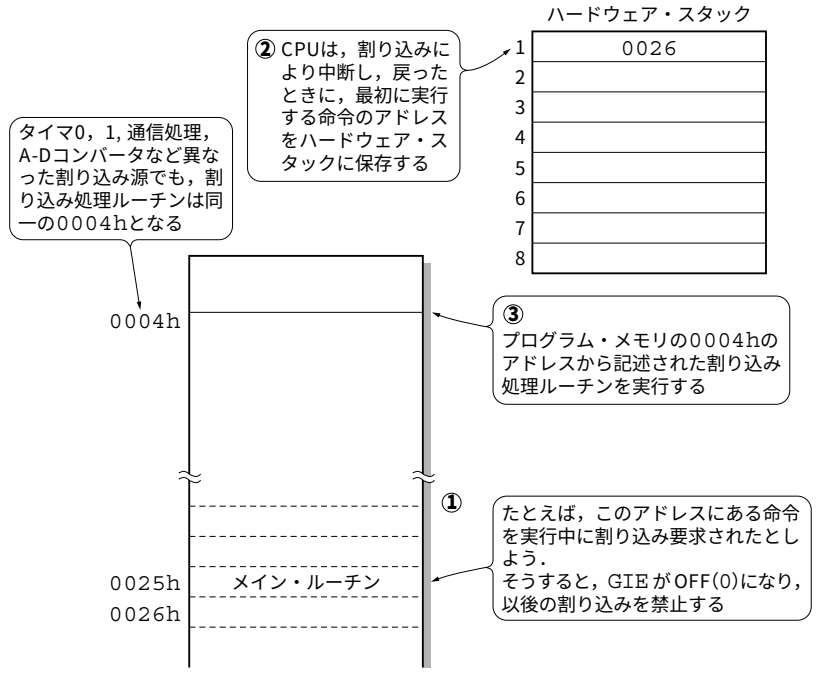

図7-5　割り込み処理の開始手順

でも使われて内容が変わってしまう可能性が高いので，入口のところで保存しておきます．その割り込み処理中保存しておくための変数を定義します(①)．このほかに，メインの処理で使用していて割り込み処理でも使用しメインの処理に戻ったときに影響を与えるものがあったら，同様な方法で保存します．Wレジスタはデータの処理などで多用され，STATUSレジスタもバンク選択，Z，Cなど演算結果などでフラグの状態がプログラムの中で参照されるので，この二つは特別なことがない限り保存が必要です．

PCLATHレジスタは，プログラムがページ0以外のページ1，2，3を使用しているときに保存が必要です．PCLATHはその値を保存すると共に，PCLATHレジスタをクリアして0ページを指定しないと割り込み処理ルーチンの中で，call，goto命令を使用したときにプログラムが暴走します．FSRレジスタの保存は，割り込み処理の中で間接アドレッシング を使用しない場合は保存不要です．

レジスタの保存領域はバンク0の70h以降の領域に指定します．16F87XAなどの368バイトのRAMをもつデバイスでは，どのバンクからアクセスしてもこのアドレスにアクセスできるので，レジスタの保存に適しています．

② 割り込み処理ルーチン

ユーザは，具体的な割り込み処理のルーチンを0004h番地から記述します．PICは割り込みを検出すると，GIEをOFFにして，割り込み禁止状態にします．この処理により，割り込み処理中に再度割り込みが要求されることが防げます．次にPICは，割り込み処理が終わったときに，中断された処理を継続するためのリターン・アドレスをハードウェア・スタックに保存します．ここまではPICが自動で行います．次に，0004h番地のアドレスから書かれたプログラムを割り込み処理ルーチンとして実行します．

割り込み処理ルーチンの最初に記述する処理は，①で用意した保存領域に，レジスタの内容をこわさないように，保存していきます(②)．Wレジスタを最初に保存しているのは，その後の処理でWレジスタ