[第6章]

メモリ・アクセス

● 概要

AVRマイコンのメモリ配置については，「1-3　メモリ構造 」のところで述べましたが，AVRマイコンを使いこなすにはメモリ操作について熟知しておく必要があります．ここでは，メモリのアクセス方法について解説します．

6-1　レジスタのアクセス

命令によって，アクセス範囲に制限があるので注意が必要です．**図6-1**に従って命令を使い分けてください．命令を間違えてもアセンブル時にエラーにならない場合がありますが，その場合の結果は保証されないため，制限に従って使い分ける必要があります．

■ 汎用レジスタと標準I/Oレジスタ間のデータ転送

IN命令/OUT命令を使用します．

標準I/Oアドレスはデータ・メモリ・アドレスではなくI/Oアドレスで指定します．たとえば，データ・メモリ・アドレス0x0020はI/Oアドレス0x00です．また，標準I/Oレジスタ・アドレスはインクルード・ファイル内においてレジスタ名で定義されているので，なるべくそちらを使用するようにします．

使用例

```
IN    R16, PINB   ; 標準I/OレジスタPINB(I/Oアドレス0x03)の内容を汎用
                  ; レジスタR16に転送
OUT   PORTB,R16   ; 汎用レジスタR16の内容を標準I/OレジスタPORTB(I/O
                  ; アドレス0x05)に転送
```

■ 汎用レジスタと拡張I/Oレジスタ間，または汎用レジスタと内蔵SRAM間のデータ転送

LDS命令/STS命令を使用します．

拡張I/Oレジスタ・アドレスはインクルード・ファイル内においてレジスタ名で定義されているので，なるべくそちらを使用するようにします．

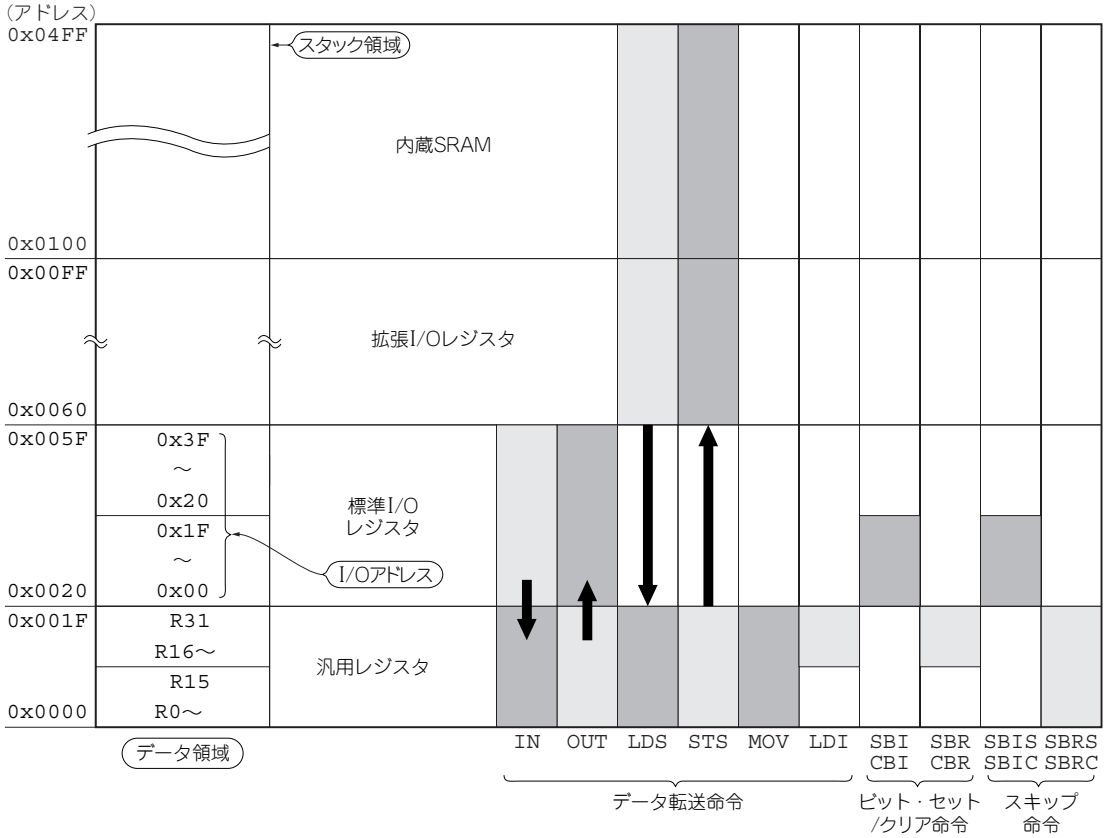

図6-1　命令別のレジスタ・アクセス範囲

使用例

```
LDS   R16, TIMSK0  ; 拡張I/OレジスタTIMSK0の内容を汎用レジスタR16に転送
LDS   R16, 0x0100  ; データ・メモリ・アドレス0x0100の内容を汎用レジスタ
                     R16に転送
STS   TIMSK0, R16  ; 汎用レジスタR16の内容を拡張I/OレジスタTIMSK0に転送
STS   0x0100, R16  ; 汎用レジスタR16の内容をデータ・メモリ・アドレス
                     0x0100に転送
```

■ 汎用レジスタ(R0～R31)間のデータ転送

MOV命令を使用します.

使用例

```
MOV   R0, R16      ; 汎用レジスタR16の内容を汎用レジスタR0に転送
```

[第7章]

割り込みについて

● 概要

割り込みとは，マイコンがプログラムで記述された順番に処理を実行している最中に発生する予定外の処理をいいます．予定外とはいっても，割り込みが発生した際の処理についてはあらかじめプログラムとして記述しておく必要があります．

割り込みの説明に，仕事中に電話がかかってくるたとえがよく使われます．電話がかかってくると，仕事を中断して電話に応対し，終わったら中断したところから仕事に戻ります．これと同様のことをマイコンでも行います．

7-1　割り込みベクタ・テーブルの定義

割り込みを利用するには，割り込みベクタ・テーブルを定義する必要があります．各割り込みは，その割り込みが発生した際に実行するプログラム・アドレスが「割り込みプログラム・アドレス表(**表7-1**)」のとおり決まっています．たとえばATmega48/88における外部割り込み1(INT1)のプログラム・アドレスは0x0002になります．プログラム・アドレスには，その割り込み処理を記載したラベル名へのジャンプ命令を記述します．

割り込みベクタ・テーブルの開始アドレスは，プログラムの先頭番地(0x0000)になります．0x0000は電源ONなどによるリセットが発生した場合のプログラム・アドレスで，ここにメイン・ルーチンへのジャンプ命令を記述します．

ATmega168ではRJMP命令でアクセスできるメモリ領域を超えた位置に割り込みルーチンを置くことが可能であり，その場合は2ワード命令であるJMP命令を使用するため，割り込みプログラム・アドレスを2番地ごとに割り振っています．ATmega168においてもRJMP命令は使用できますが，その場合でもプログラム・アドレスは必ず2番地ごととなります(「5-5　無条件ジャンプ」参照)．

リスト7-1は割り込みベクタ・テーブルの定義例です．

たとえば，タイマ2のオーバフロー割り込みが発生した場合，マイコンはプログラム・アドレス0x0009を実行します．ここにはRJMP　　IOVF2と記述されているため，IOVF2というラベルの割り込み処理へジャンプします．

表7-1　割り込みプログラム・アドレス表

ベクタ番号	プログラム・アドレス		名　称	発生要因
	ATmega48/88	ATmega168		
1	0x0000	0x0000	RESET	各種リセット要因(「1-2　マイコンの動作」参照)
2	0x0001	0x0002	INT0	外部割り込み要求 0 発生
3	0x0002	0x0004	INT1	外部割り込み要求 1 発生
4	0x0003	0x0006	PCINT0	ピン変化割り込み要求 0 発生
5	0x0004	0x0008	PCINT1	ピン変化割り込み要求 1 発生
6	0x0005	0x000A	PCINT2	ピン変化割り込み要求 2 発生
7	0x0006	0x000C	WDT	ウォッチドッグ・タイマのタイムアウト
8	0x0007	0x000E	TIMER2 COMPA	タイマ / カウンタ 2 出力比較レジスタ A 一致
9	0x0008	0x0010	TIMER2 COMPB	タイマ / カウンタ 2 出力比較レジスタ B 一致
10	0x0009	0x0012	TIMER2 OVF	タイマ / カウンタ 2 オーバフロー
11	0x000A	0x0014	TIMER1 CAPT	タイマ / カウンタ 1 キャプチャ
12	0x000B	0x0016	TIMER1 COMPA	タイマ / カウンタ 1 出力比較レジスタ A 一致
13	0x000C	0x0018	TIMER1 COMPB	タイマ / カウンタ 1 出力比較レジスタ B 一致
14	0x000D	0x001A	TIMER1 OVF	タイマ / カウンタ 1 オーバフロー
15	0x000E	0x001C	TIMER0 COMPA	タイマ / カウンタ 0 出力比較レジスタ A 一致
16	0x000F	0x001E	TIMER0 COMPB	タイマ / カウンタ 0 出力比較レジスタ B 一致
17	0x0010	0x0020	TIMER0 OVF	タイマ / カウンタ 0 オーバフロー
18	0x0011	0x0022	SPI STC	SPI 送出完了
19	0x0012	0x0024	USART RX	USART 受信完了
20	0x0013	0x0026	USART UDRE	USART 送信データ・レジスタ空
21	0x0014	0x0028	USART TX	USART 送信完了
22	0x0015	0x002A	ADC	A-D 変換完了
23	0x0016	0x002C	EE RDY	EEPROM 操作可
24	0x0017	0x002E	ANALOG COMP	アナログ・コンパレータ　割り込み発生条件満足
25	0x0018	0x0030	TWI	TWI 要求処理完了
26	0x0019	0x0032	SPM READY	SPM 命令 操作可

リスト7-1　割り込みベクタ・テーブルの定義例

```
;---------------------------------------
; 割り込みベクタ定義例
;---------------------------------------

.CSEG                       ;コード・セグメント
        RJMP  MAIN          ;リセット→0番地に飛んで最初から実行
.ORG    0x0001
        RJMP  EXT_INT0      ;外部割り込み0の処理へ
.ORG    0x0009
        RJMP  IOVF2         ;タイマ2の処理へ
.ORG    0x000D
        RJMP  IOVF1         ;タイマ1の処理へ
.ORG    0x0010
        RJMP  IOVF0         ;タイマ0の処理へ
```

プログラム(コード・セグメント)の先頭を意味するアセンブラ擬似命令

プログラム・アドレスをセットするアセンブラ擬似命令

[第8章]

各機能の使い方

8-1 8ビット・タイマ（タイマ0，タイマ2）

本章ではAVRの各機能を解説する．タイマ，外部割り込み，ピン変化割り込み，A-D変換，アナログ・コンパレータ，PWM，USART，TWI，SPI．これらの機能を一つだけ，もしくは複数組み合わせて利用することで，AVRの処理能力を生かした多くのアプリケーションを実現することが可能になる．

● 概要

タイマとは時間を生成する機能であり，一般的な動作は次のとおりです．

- カウンタに初期値をセットする
- タイマをスタートさせると，カウンタが初期値から一定時間ごとにカウント・アップされていく
- カウンタが最大値となった次のカウント・アップのタイミングでオーバフローのフラグがセットされる

タイマによって生成される時間はスタートからカウンタ・オーバフローまでの時間であり，生成時間＝（最大カウント数－カウンタ初期値）×カウント・アップ時間となります．8ビット・タイマにおける最大カウント数は$2^8 = 256$であり，カウンタ初期値は0～255の範囲で設定します．

カウント・アップの時間は，プリスケーラと呼ばれるマイコンのクロックの倍数で設定します．たとえば，**図8-1-1**のようにカウンタの初期値を120，カウント・アップ時間を（t_{up}）とした場合，生成時間（T）は，

$$T = (256 - 120) \times t_{up}$$

となります．

プログラムで「カウンタ初期値」，「カウント・アップ時間」を設定することで，タイマ・スタートからカウンタ・オーバフローまでの時間を変更することが可能です．

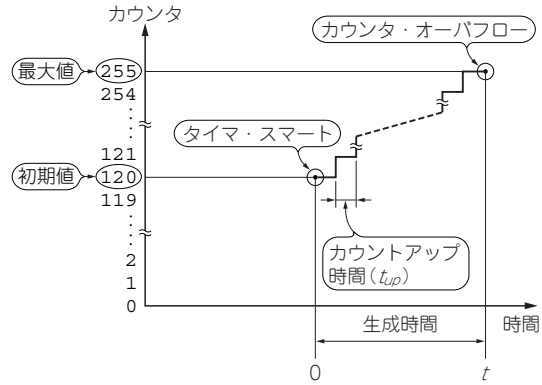

図8-1-1　8ビット・タイマの動作

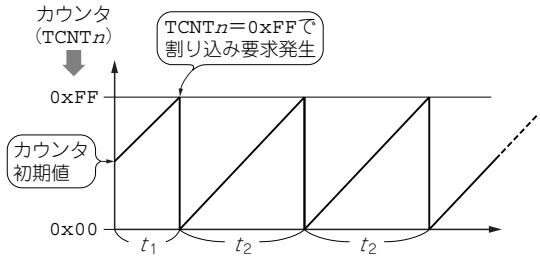

図8-1-2　ノーマル・モードの動作

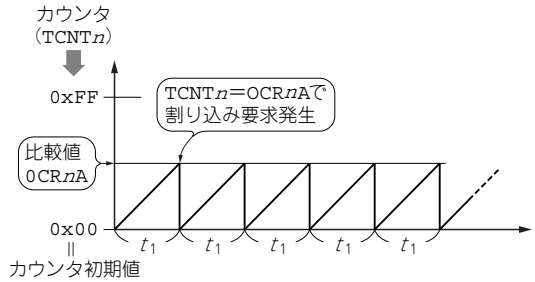

図8-1-3　CTCモードの動作

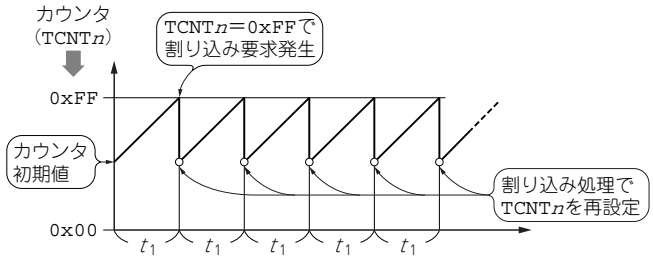

図8-1-4　ノーマル・モード動作における同一時間の取得方法

AVRマイコンのタイマの動作には，「ノーマル・モード」と，「CTCモード」があります．その他に「PWMモード」がありますが，それについては「8-7　PWM」で説明します．タイマ0使用例 ①および ② にノーマル・モード，使用例 ③ および ④ にCTCモードのプログラム例を記載しているので参照ください．

冒頭でタイマとは時間を生成する機能であると述べましたが，別の言い方をすると時々刻々変化するカウンタの値と希望の値との比較を行う機能であり，希望の値のことを「比較値」と呼ぶことにします．

8ビット・タイマでは，ノーマル・モードでの比較値はカウンタの最大値である0xFFです．それに対してCTCモードでの比較値は，レジスタOCR*n*Aにセットする値になります(*n*はタイマ0の場合は0，タイマ2の場合は2)．

ノーマル・モードでは，タイマ・スタートからオーバフロー発生までの希望の時間を得るために，それに見合う初期値をカウンタにセットしてタイマをスタートさせます．カウンタが0xFFとなった次のカウント・アップで割り込み要求が発生するとともに，カウンタは0x00にクリアされてカウントが継続します．

最初の割り込み要求は希望の時間(t_1)で発生しますが，その後の割り込み要求はカウンタが0x00から0xFFまでカウントしてから発生(t_2)します(**図8-1-2**)．

CTCモードではレジスタOCR*n*Aに希望の値をセットし，カウントを0x00からスタートさせます．カウンタがOCR*n*Aの値となった次のカウント・アップで割り込み要求が発生するとともに，カウンタは0x00にクリアされてカウントが継続します(**図8-1-3**)．

ノーマル・モードにおいて希望の時間を連続して得るには，割り込み要求が発生するたびにプログ

[第8章]

各機能の使い方
8-3 外部割り込み

● 概要

外部割り込みとは，特定のピンに変化があったときに発生する割り込みをいいます．タイマと並んでよく使用される割り込みで，スイッチ入力が一つか二つの場合は外部割り込みを使用するのが一般的です．

スイッチなどの入力があるシステムでは，この外部割り込みを使用することで常に入力を監視する必要がなくなり，その間にほかの処理を実行することが可能となるため，システム全体の効率が上がります．

広い意味では別章のピン変化割り込みも外部割り込みですが，ピン変化割り込みがポート単位の割り込みであるのに対し，この章ではピン単位で割り込みができるINT_nについて説明します．

INT_nのnは0, 1, 2, …であり，マイコンの規模が大きくなるほど数が大きくなります．たとえば，ATtiny45ではINT_0のみ，ATmega88ではINT_0およびINT_1の二つとなります．

外部割り込みピン(INT_0，INT_1)の入力に変化があり，外部割り込み制御レジスタAの割り込み発生条件と一致した場合は，外部割り込み要求フラグ・レジスタ(EIFR)の，変化した外部割り込みピ

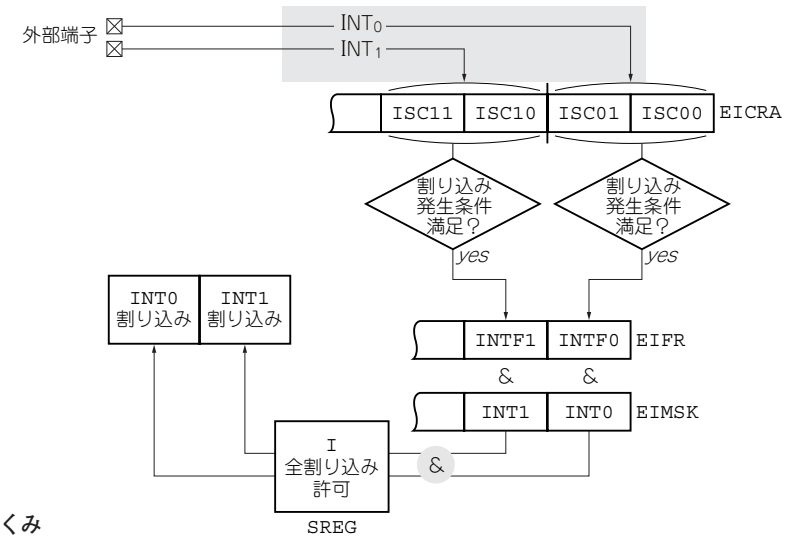

図8-3-1　外部割り込み発生のしくみ

ンに対応するビットがセットされます．そして，外部割り込みマスク・レジスタの該当ビットが有効で，かつ全割り込みが許可されていれば，対応する割り込みベクタが実行されます(**図8-3-1**)．

● 外部割り込み利用方法

使用にあたり，下記の設定を行います．

▸ 割り込み発生条件(EICRA)．
▸ 外部割り込みマスク設定(EIMSK)．

● 外部割り込み関連レジスタ

図8-3-2に関連レジスタの内容を示します．

● 外部割り込みの使用例

外部割り込み0(INT_0)に接続されたスイッチをSW_0，外部割り込み1(INT_1)に接続されたスイッチをSW_1とし，SW_0が押されたら1秒間LED_0を点灯，SW_1が押されたら1秒間LED_1を点灯する事例を**図8-3-3**を用いて説明します．実行例を**写真8-3-1**に示します．

プログラムの解説

メイン・ルーチン

■ 初期処理
▸ 全割り込みを禁止する．
▸ 入出力ポートの設定をする．

LED制御用にPB_0，PB_1を使用しますが，LEDのカソード側をマイコンのピンに接続するので出力'0'(Low)で点灯，'1'(High)で消灯状態になります．最初は消灯させるために'1'(High)にします．

▸ 外部割り込み関連レジスタをセットする．
▸ ユーザ・フラグ(R_FLAG1)をクリアする．
▸ 全割り込みを許可する．

■ 主処理
▸ ユーザ・フラグ内の外部割り込み0フラグ，外部割り込み1フラグをクリアする．
▸ ループに入り割り込みが発生するのを待つ．
▸ スイッチ(SW_0 or SW_1)が押されると割り込みがかかり，外部割り込み0フラグ or 外部割り込み1フラグがセットされてループを抜ける．

割り込みを使用することで，メイン・ルーチンはいつ押されるかわからないスイッチを気にせずに処理を行うことが可能となります．

また，ユーザ・フラグを使用する理由ですが，割り込み処理はメイン・ルーチンやサブルーチンで何らかの処理中にいきなり発生するため，場合によっては非常にまずい事態を引き起こす恐れがあります．そのため，割り込み発生時にはその割り込みが発生したというフラグ(ユーザ・フラグ)だけを立てておき，メイン・ルーチンやサブルーチンで，実行中の処理が一段落したときにフラグを見て必要な処理を行うという手法を採るようにしています．

[第8章]

各機能の使い方

8-5 A-D変換

● 概要

A-D変換とは，基準となる電圧に対して測定する電圧がどれくらいかを比率で求める機能です．マイコンの入力として取り込む事象(音/光/温度/重量など)は連続して変化するアナログ量ですが，これをマイコン内部で処理するには基準値との比較であるディジタル量に変換する必要があり，通常は基準電圧との比較になります．

測定結果はビット数が多いほど精度(分解能)が高くなり，たとえば8ビットでは256段階(0～255)で，10ビットでは1024段階(0～1023)で表すことが可能です．基準電圧が5 Vの場合は，8ビットでは5 V ÷ 256 = 0.02 V，10ビットでは5 V ÷ 1024 = 0.005 Vの精度となります．AVRマイコンでは，初期状態では10ビット精度ですが，8ビット精度への変更が可能です(**図8-5-1**)．

● 設計上の注意点

A-D変換の機能を利用するには，マイコン用電源(V_{CC})とは別にA-D変換用の電源(AV_{CC})を供給する必要があります．AV_{CC}はV_{CC}±0.3V以内となるようにします．

通常はV_{CC}をAV_{CC}に供給してかまいませんが，マイコンが発生するノイズの影響で基準電圧がふらつくのを低減するためにLCフィルタの挿入が推奨されています．それほどシビアな精度が要求されない趣味などの用途では，とくにLCフィルタを挿入しなくても問題ありませんが，スリープ・モードのA-D変換ノイズ低減モードを利用することでノイズ低減を図ることは可能です．

測定する電圧の端子(A_{in})はプルアップなしの入力として初期化します．

測定結果は，測定電圧が基準電圧以上になる場合は最大値(10ビット精度では最大値1023)で頭打ちとなり測定できません．したがって，測定電圧として印加されるV_{in}が基準電圧を超える可能性が

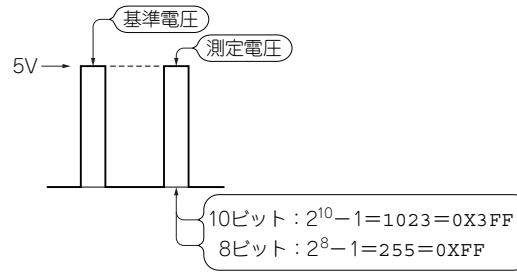

図8-5-1　A-D変換におけるビット数による精度(分解能)の違い
ビット数が多いほど高精度の測定が可能．

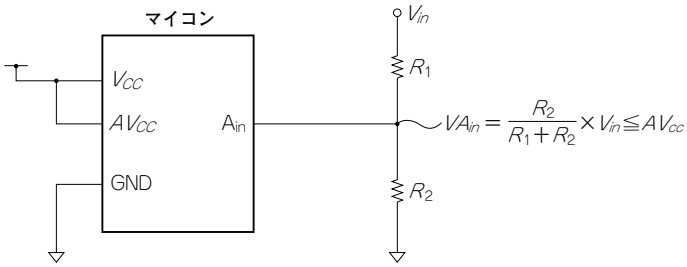

図8-5-2　A-D変換における分圧
A_{in}への入力はAV_{cc}以下となるようにする．

ある場合は，抵抗による分圧で基準電圧を超えないようにします(**図8-5-2**)．また，A_{in}にかかる電圧(VA_{in})が入力ピンの許容電圧を超える恐れがある場合は，保護回路を設けるとよいでしょう．

基準電圧は下記の3種類から選択します．

▸外部リファレンス電圧(A_{ref})．
外部から基準となる電圧をマイコンに供給します．

▸内部リファレンス電圧．
マイコン内部で生成される基準電圧(1.1 V)を利用します．

▸アナログV_{CC}(AV_{CC})．
A-D変換用の電源(AV_{CC})を利用します．

● A-D変換の方法

使用に先立ち，初期設定として下記のレジスタを設定します．

▸ADMUX(A-Dマルチプレクサ選択レジスタ)．
▸ADCSRA(A-D変換 制御/ステータス・レジスタA)．
▸ADCSRB(A-D変換 制御/ステータス・レジスタB)．

初期設定例を下記に示します．

```
LDI         R_TEMP1,0x60      ;AVCC/左詰/ADC0
STS         ADMUX,R_TEMP1
LDI         R_TEMP1,0x8B      ;変換許可/割り込み許可/(CLK÷8)
STS         ADCSRA,R_TEMP1
LDI         R_TEMP1,0x00      ;設定項目なし
STS         ADCSRB,R_TEMP1
```

● A-D変換関連レジスタ

図8-5-3に関連レジスタを示します．

● A-D変換の使用例

電源電圧を5 V，基準電圧をAV_{CC}とし，ADC_0から入力する電圧値によって，次のようにLED_0～LED_2を点灯させます．ただしA-D変換割り込みは使用せず，分解能は8ビットとします．

[第8章]

各機能の使い方
8-6 アナログ・コンパレータ

● 概要

コンパレータとは，二つの電圧の大小を比較する機能です．比較するという点でA-D変換と似ていますが，A-D変換が基準電圧との比率を求めるのに対して，コンパレータは大小判定だけなので取り扱いが容易です．

コンパレータは基準値をあらかじめ決定し，その値になったら何がしらのイベントを発生させる場合に多く用いられます．たとえば，昼夜判別式の街灯は，昼夜が変わる値を決めてそれを基準値とし昼間は「比較値＞基準値」で消灯，夕方になり周囲が暗くなると「比較値＜基準値」となって点灯するという具合です．

図8-6-1のように，AIN_0が比較器の＋側，AIN_1が比較器の－側になります．比較結果は，後述するACSRレジスタのACOビットに入ります．

図8-6-2はアナログ・コンパレータの機能を表したブロック図です．ACBGは後述するACSRレジスタのビットで，AIN_0に入力される外部リファレンス電圧を基準とするか，バンド・ギャップ・リファレンスと呼ばれる内部リファレンス電圧(1.1 V)を基準とするかを設定します．

外部リファレンス電圧は文字どおり外部から入力する電圧ですが，正しい比較を行うためには電源電圧(V_{CC})を超えないようにします．ACME，ADENは基準電圧と比較する電圧を選択するためのビットです．詳しくは後述する「コンパレータ入力選択について」を参照ください．

● 設計上の注意点

比較用電圧の端子はプルアップなし入力で初期化します．

● アナログ・コンパレータの利用方法

使用にあたり，次の設定を行います．

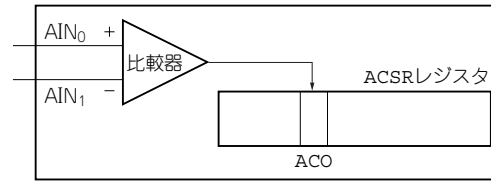

図8-6-1　アナログ・コンパレータの動作
AIN_0＞AIN_1の場合，ACOは‘1’になる．
AIN_0≦AIN_1の場合，ACOは‘0’になる．

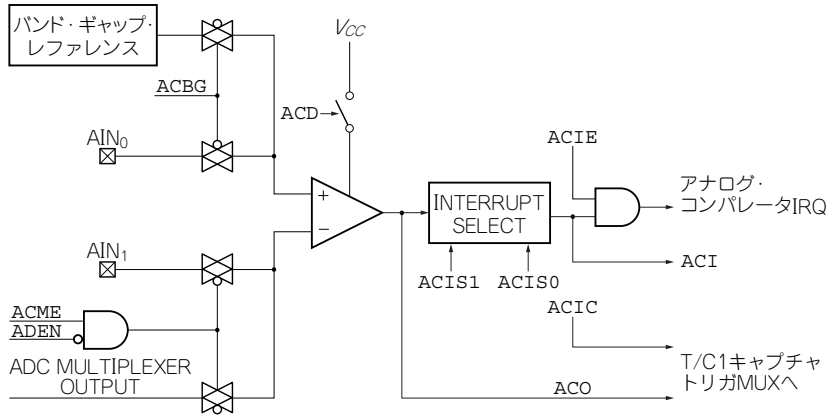

図8-6-2　アナログ・コンパレータのブロック図

- 基準電圧選択(ACSR)
- コンパレータ入力選択(ADCSRB)
- 割り込みタイミング選択
- 割り込み設定

● アナログ・コンパレータ関連レジスタ

図8-6-3に関連レジスタを示します.

● コンパレータ入力選択について

基準電圧と比較するための入力(アナログ・コンパレータ反転入力)を，ADCSRBレジスタのACMEビット，ADCSRAレジスタのADENビット，ADMUXレジスタのMUX2～MUX0ビットの組み合わせにより設定します(**表8-6-1**)．ADCSRAレジスタのADENビットとADMUXレジスタのMUX2～MUX0ビットについては，「8-5　A-D変換」を参照ください.

表8-6-1　コンパレータ入力の選択

ACME	ADEN	MUX2 ～ MUX0	アナログ・コンパレータ反転入力
0	x	xxx	AIN_1
1	1	xxx	AIN_1
1	0	000	ADC_0
1	0	001	ADC_1
1	0	010	ADC_2
1	0	011	ADC_3
1	0	100	ADC_4
1	0	101	ADC_5
1	0	110	ADC_6
1	0	111	ADC_7

[第8章]

各機能の使い方
8-7 PWM

● 概要

PWMとは，Pulse Width Modulationの略で，直訳するとパルス幅変調となります．これは，周期におけるONとOFFの比率を変えて制御する方法で，モータ制御や調光(光の明るさを可変する)などに使用されます．たとえば，LEDでは，電流をパルス状にして点灯時間を100%から50%にした場合，見かけ上電流値を半分にしたのと同じように見えます(**図8-7-1**)．

AVRマイコンのPWMには，「高速PWMモード」と「フェーズ・コレクトPWMモード」という2種類のモードがあります．ただし，調光やモータ制御などにおいてどちらも同じような出力を得ることが可能であり，プログラムの組みやすさにも大差はないため，どちらのモードを使用してもかまわないと思います．

■ 高速PWMモード

後述の**図8-7-3**に示すようなカウント値をプロットした形をカウント波形と呼び，それがノコギリ波です．

高速という言葉が使われるのは，波形がノコギリ波であることでフェーズ・コレクトPWMモードと比較して周期が半分になり，周波数が倍になることからきています．

■ フェーズ・コレクトPWMモード

カウント波形が三角波です．

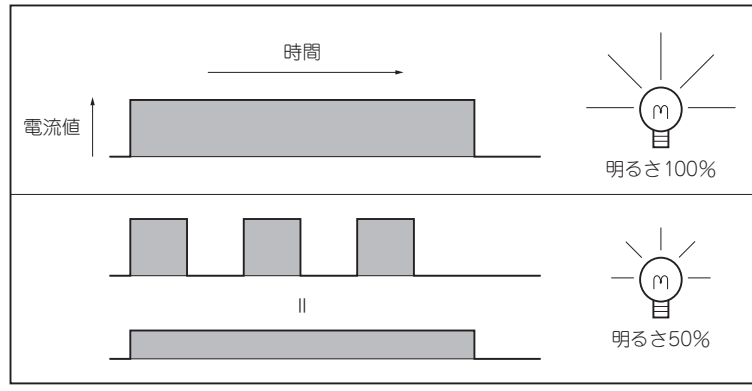

図8-7-1　PWM方式による調光

● 設計上の注意点

PWM信号の出力ピンはOC*n*AおよびOC*n*Bです．*n*はタイマ/カウンタ*n*の*n*です．つまり，タイマ0のPWM機能を使用する場合の出力はOC0AおよびOC0Bとなり，タイマ2のPWM機能を使用する場合の出力はOC2AおよびOC2Bとなります．

動作モード

カウント波形の形状を，TCCR0Bのビット3およびTCCR0Aのビット1，0からなるWGM02～WGM00により設定する．

アドレス	I/Oアドレス	レジスタ名	ビット7	ビット6	ビット5	ビット4	ビット3	ビット2	ビット1	ビット0
0x45	0x25	TCCR0B	FOC0A	FOC0B	—	—	WGM02	CS02	CS01	CS00
初期値			0	0	0	0	0	0	0	0
Read/Write			W	W	R	R	R/W	R/W	R/W	R/W

アドレス	I/Oアドレス	レジスタ名	ビット7	ビット6	ビット5	ビット4	ビット3	ビット2	ビット1	ビット0
0x44	0x24	TCCR0A	COM0A1	COM0A0	COM0B1	COM0B0	—	—	WGM01	WGM00
初期値			0	0	0	0	0	0	0	0
Read/Write			R/W	R/W	R/W	R/W	R	R	R/W	R/W

番号	WGM02～WGM00	動作モード	TOP値	OCR0x更新タイミング	TOV0セットタイミング
⓪	000	ノーマル	0xFF	即時	0xFF
①	001	TOP値固定フェーズ・コレクトPWM	0xFF	TOP	0x00
②	010	CTC	OCR0A	即時	0xFF
③	011	TOP値固定高速PWM	0xFF	BOTTOM	0xFF
④	100	—	—	—	—
⑤	101	フェーズ・コレクトPWM	OCR0A	TOP	0x00
⑥	110	—	—	—	—
⑦	111	高速PWM	OCR0A	BOTTOM	0xFF

高速PWMモードには，番号③（TOP値固定高速PWM動作）と，番号⑦（高速PWM動作）の2種類がある．

番号③はカウンタのTOP値が0xFFと決まっている．

番号⑦はカウンタのTOP値はOCR0Aの値であり，ユーザが任意に設定できる．

フェーズ・コレクトPWM動作には，番号①（TOP値固定フェーズ・コレクトPWM動作）と，番号⑤（フェーズ・コレクトPWM動作）の2種類がある．

番号①はカウンタのTOP値が0xFFと決まっている．

番号⑤はカウンタのTOP値はOCR0Aの値であり，ユーザが任意に設定できる．

出力条件

TCCR0Aのビット7，6からなるCOM0A1，COM0A0およびビット5，4からなるCOM0B1，COM0B0により設定する．

COM0A1，COM0A0は出力ピンOC0Aに影響を与える．

COM0B1，COM0B0は出力ピンOC0Bに影響を与える．

COM0A1	COM0A0	動　作
0	0	標準ポート動作
0	1	WGM02=0：標準ポート動作 WGM02=1：一致時OC0Aトグル出力
1	0	一致時Low，BOTTOMでHigh出力
1	1	一致時High，BOTTOMでLow出力

COM0B1	COM0B0	動　作
0	0	標準ポート動作
0	1	—
1	0	一致時Low，BOTTOMでHigh出力
1	1	一致時High，BOTTOMでLow出力

動作モードが番号③（TOP値固定高速PWM動作）では，出力条件のCOM0A1，COM0A0およびCOM0B1，COM0B0が1，0または1，1のときに意味をもつ．

PWM信号の出力ピンはOC*n*AおよびOC*n*Bであることは前述したが，PWM出力として片方しか使用しない場合は，使用しないほうを標準ポート動作（すなわち0，0）に設定する．

図8-7-2　PWM関連レジスタ

[第8章]

各機能の使い方
8-9 TWI(I²C)

● 概要

TWIとはTwo Wire Interface の略で，2線式シリアル・インターフェースとも呼ばれます．またフィリップス社のI²Cインターフェースと互換性があります．

データ線(SDA)とクロック線(SCL)の2本の信号線に，最大128のデバイスを接続することが可能です(**図8-9-1**)．別章のSPI方式に比べて通信速度は遅いですが，メモリ，リアルタイム・クロックICのほか，多くのデバイスに採用されている方式です．主に基板上に配置したデバイス間で使用するのが一般的で，仮にケーブルで接続するとしても通信可能な距離はせいぜい数m程度と考えてください．

TWIバスに接続されるデバイスは，データを送出するマスタとデータを受けるスレーブに分かれます．ある瞬間だけ見ると，マスタとスレーブは1対1です(特殊な使い方としてマスタがほかの全デバイスをスレーブにする機能があるが，本書では省略)．

TWIの動作には下記のものがあります．

(1) スタート・コンディション(開始条件)
(2) リピーテッド・スタート・コンディション(再送開始条件)
(3) SLA＋W(コントロール・バイト)
(4) SLA＋R(コントロール・バイト)
(5) ストップ・コンディション(停止条件)
(6) データ転送(TWDRレジスタ・セット)

スレーブを識別するために，7ビットでデバイス・アドレスが設定されます．デバイス・アドレスは，ユーザが任意に決めることができる場合と，デバイス側による制限がある場合(たとえば7ビットのうち5ビットがデバイス側で決められていて，2ビットしか指定できないなど)があります．

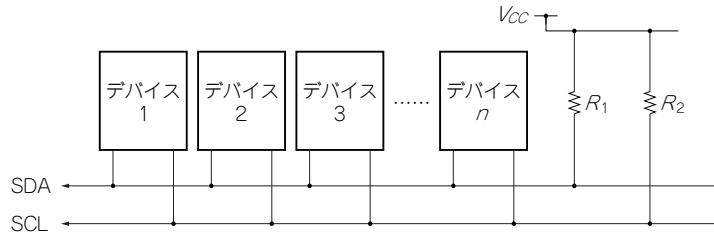

図8-9-1　TWIの接続

TWIバスが空いていればどのデバイスもマスタになることが可能であり，マスタとなったデバイスはスレーブとするデバイスのアドレスを送出し，スレーブがそれに応答することで，1対1の通信が始まります．

図8-9-2は信号線の状態遷移です．マスタとなるデバイスはTWIバスが空いている(SCLが“H”状態)ことを確認し，スタート・コンディションを送出してバスを占有します．スレーブとの通信が完了すると，ストップ・コンディションを送出してバスを開放します．

スタート・コンディションとは，SCLが“H”の状態でSDAを“H”→“L”にした場合を指します．ストップ・コンディションとは，SCLが“H”の状態でSDA を“L”→“H”にした場合を指します(**図8-9-2**)．

通信のパケットは8ビットですが，スレーブからのACK/NACK受信のため，マスタは9ビット目のSCLクロックを送出します．マスタからパケットを受け取ったスレーブは，ACK(正常受信)もしくはNACK(異常受信)を返します．

具体的には，マスタがSCLクロックに同期して8ビットを送信(スレーブが8ビット受信)した後，9ビット目のSCLクロックでスレーブがSDAを“L”にすることで，ACKを返したことになります．“H”のままだとNACKを返したことになります(**図8-9-3**)．

スタート・コンディション(およびリピーテッド・スタート・コンディション)の直後のパケットは，コントロール・バイトと呼ばれます．コントロール・バイトは，7ビットのデバイス・アドレスとW/Rを示す1ビットで構成されます．W(WRITE)の値は‘0’で，マスタがスレーブに対してコマンドやデータの転送を行うことを意味し，SLA+Wと表記します．R(READ) の値は‘1’で，マスタがスレーブに対してデータの要求を行うことを意味し，SLA＋Rと表記します．

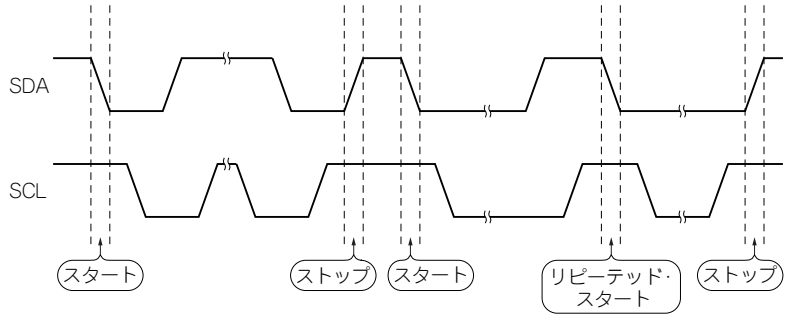

図8-9-2　SDAとSCLの状態変化

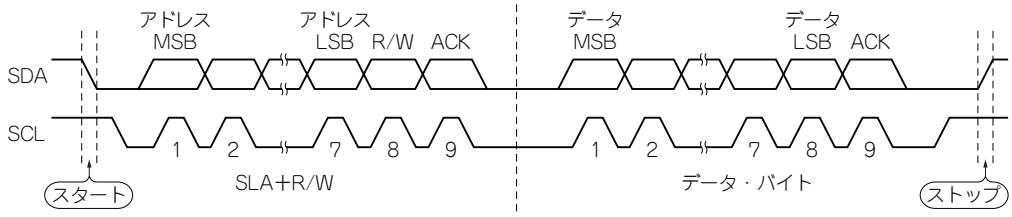

図8-9-3　TWIの通信パケット

[第8章]

各機能の使い方
8-10 SPI

● 概要

SPIとはSerial Peripheral Interface の略で，3線式インターフェースとも呼ばれます．SPI方式を採用するデバイスにはEEPROM，センサなどがあり，マイコン同士のデータのやりとりにも使用されるなど，TWI方式とともにデバイス間の非常にポピュラなインターフェースとして使用されています．

TWIは通信速度の上限が400 kHzであるのに対し，SPIでは最大でシステム・クロックの1/2までと高速な通信が可能であるため，スピードが必要なところでメリットが生かせます．主に基板上に配置したデバイス間で使用するのが一般的で，仮にケーブルで接続するとしても通信可能な距離はせいぜい数m程度と考えてください．

SPIはMISO，MOSI，SCKの3線とスレーブ・セレクト信号$\overline{SS}$で構成されます．スレーブ・デバイスに対し，TWIではデバイスごとにデバイス・アドレスを設定し，マスタがアドレス指定することで1対1の通信が成立していました．SPIではデバイス・アドレス設定は不要ですが，代わりにスレーブ・セレクト信号でデバイスを指定する必要があります．

図8-10-1に示すように，ポイントはマスタ→スレーブへ1バイト転送されると同時に，スレーブ→マスタへ1バイト転送されることです．つまり，マスタからのSCKのクロックに同期して，双方の8ビット・データが押し出され，マスタとスレーブのデータが入れ替わります．

AVRマイコンがマスタの場合，SPDRレジスタに1バイトのデータをセットするとSPI機能が自動

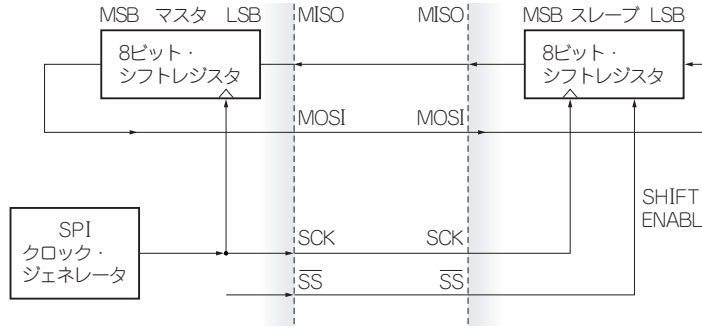

図8-10-1　SPIの動作

的に送出を行います．送出完了はSPSRレジスタのSPIFビットで確認します．SPIFビットはSPDRへのデータ・セットによりクリアされ，送出完了によってセットされます．

通常AVRマイコンに接続されるデバイスがスレーブになるためマイコン側がマスタですが，マイコン同士の接続の場合はマスタとスレーブのペアになります．マスタ/スレーブの設定は，SPCRのMSTRビットで定義します．セット‘1’でマスタ，クリア‘0’でスレーブです．

SPI関連のピンは，使用時に自動的に機能に適した値にされるので，とくに初期設定の必要はありません．

● SPIにおけるデータ交換例

SPI機能を使用するデバイスの通信例として，マスタからの要求に対してスレーブがデータを送出する例を紹介します(あくまでも例ということで，デバイスによって通信内容は異なる)．

マスタからの開始コードに対するスレーブからの応答(了解コード)を確認後，マスタはデータ要求を意味する要求コードを送信し，スレーブからデータを受け取るという一連の流れです．スレーブはデータを送る準備が整ったら，データ・コードに続けてデータを送出するものとします．

やり取りのようすを**図8-10-2**に示します．

(1) スレーブはマスタからの開始コードを待つ
(2) マスタが開始コードを送信する
(3) スレーブが開始コードを受信する
(4) マスタはスレーブからの了解コードを待つ
(5) スレーブが了解コードを送信する
(6) マスタが了解コードを受信する
(7) スレーブはマスタからの要求コードを待つ
(8) マスタが要求コードを送信する
(9) スレーブが要求コードを受信し，データの準備に入る
(10) マスタはスレーブからのデータ・コードを待つ
(11) スレーブがデータを送る準備ができたことを示すデータ・コードを送信する
(12) マスタがデータ・コードを受信する
(13) スレーブがデータ・コードに続けてデータを送信する
(14) マスタがデータを受信する

前述しましたが，SPIとはマスタとスレーブ間のデータ交換であり，マスタがデータを送出しないとスレーブからデータが送られてこないため，データ交換の主導権はマスタが握っています．

データのやりとりはマスタのSCKから送出されるクロックに同期して行われますが，このクロックはマスタのSPDRデータ・レジスタへのデータ・セットで送出されます．また，スレーブは必要な処理をするための時間が必要なため，マスタは繰り返しデータ(コード)を送って期待値が返されたかどうかを調べなければなりません．期待値が返るまでは，スレーブへ送った直前のデータがそのままマスタへ帰ってきます．当然スレーブから返される期待値を識別するために，マスタは期待値以外のデータ(ダミー・コード)を送る必要があります(本書の例では，マスタはデータ・コードを受信す