Ｒ８Ｃ１５マイコン周辺Ｉ／Ｏ

ＳＦＲレジスタ設定と使用例

2005/10/30　Ver0.91（暫定版）

ＡＴ－Ｌａｂｏｒａｔｏｒｙ

このドキュメント内容は AT-Laboratory の許可なく、一部分を含む転載、複製は出来ません。
このドキュメントの著作は AT-Laboratory にあります。
サンプルソフトウェア記述は個人使用を除き、営利目的に使用する場合には
AT-Laboratory の許可が必要です。

Ｒ８Ｃ１５マイコン内蔵周辺Ｉ／Ｏ

このドキュメントでは、R8C15 の周辺 I/O 機能と使用例を解説します。

1. 端子機能
2. プログラマブル入出力ポート
   ① ポートＰ１
   ② ポートＰ３
   ③ ポートＰ４
3. タイマー
   ① タイマーＸ
   ② タイマーＺ
   ③ タイマーＣ
4. シリアルインターフェース
   ① ＵＡＲＴ０
   ② ＵＡＲＴ１
5. ＡＤコンバータ
6. 外部割り込み端子
   ① ＩＮＴ０割り込み
   ② ＩＮＴ１割り込み
   ③ ＩＮＴ３割り込み
   ④ キー入力割り込み
7. チップセレクト付きクロック同期シリアルＩ／Ｏ（ＳＳＵ）
8. フラッシュメモリ制御
9. CPU 動作機能設定

SFR 定義ファイル “ R8C15.h ” について

このドキュメント中で使用している SFR レジスタ名、ビット名は、” R8C15.h ” のレジスタ定義に基づいて記述しています。 ” R8C15.h ” は、AT ラボラトリが作成したもので、ルネサス社の定義名とは異なります。ルネサス社の ” sfr815.h ” を使用する場合には R8C15 ハードウェアマニュアルを参照してください。
AT ラボラトリ製 SFR 定義ファイル ” R8C15.h ” はルネサス社の定義ファイルと記述が異なります。
レジスタ名はすべて半角大文字で、SFR_レジスタ名、とレジスタ名の前部に ” SFR_ ” が付加されます。このレジスタ名に関しては、ルネサス R8C15 ハードウェアマニュアルに準拠しています。
また、レジスタのビット定義は、SFR_グループ名_ビット名、と周辺 I/O グループ名に統一して作成してあります。
周辺 I/O グループは、タイマＸは “ SFR_TX_ ”、タイマＺは “ SFR_TZ_ ”、タイマＣは “ SFR_TC_ ”、ＡＤコンバータは“ SFR_AD_ ”、ＵＡＲＴ０は“ SFR_U0_ ”、ＵＡＲＴ１は“ SFR_U1_ ”などのようにして、レジスタ名を使用せず、グループと機能で表現できるようにしてあります。
このビット名はルネサス表記とは異なります。出来るだけビット表記が統一出来るように変更しています。例えば、ルネサス表記ではエッジ極性は POL と PL が混在していますが、当表記では POL は、エッジの立上り/立下り選択、PL は片/両エッジ選択と統一してあります。
また互換性のために SFR_レジスタ名_ビット名（R8C15 ハードウェアマニュアル記述）も使用できます。ただし、SFR 名記述に関してはすべて半角大文字で記述してください。
さらに、ei(),di(),nop()がこのヘッダーファイル内でマクロ定義されています。

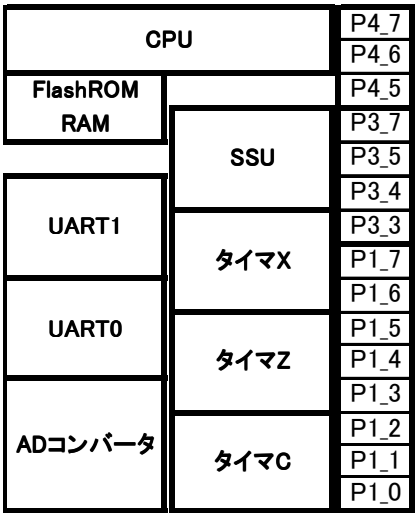
このドキュメントの記述例では、NC30WA のコンパイラオプション MVT(MakeVectorTable)を ON 状態で使用することを前提としています。従って、割り込み定義で#pragma interrupt で(vect=ベクタ番号)と表記してあります。MVT オプションを OFF で使用する場合は、(vect=xx)を削除して、別途割り込みベクタテーブルを変更記述してください。

PG_AID_R8 をインストールすると、自動的に上記の項目が有効になります。

１.端子機能

R8C15 は 20 ピンのパッケージで、電源端子の他、各端子にプログラマブル入出力ポートが割り当てられています。また、R8C15 の内部には、タイマー、シリアルインターフェース、AD コンバータ、割り込みコントローラなどが搭載されていて、ポート端子に重複して各機能の端子が割り当てられています。端子のピンは限られていますので、特定の内部 I/O を使用する場合には、その端子に割り当てられている、プログラマブル入出力ポートは使用が限定されます。

R8C15内部ブロック



端子機能

プログラマブルI/O	割り込み	タイマーX	タイマーZ	タイマーC	ADコンバータ	UART	SSU
P4_7 XOUT							
P4_6 XIN							
P4_5	INT0-		(INT0-)			RXD1	
P3_7		CNTR0-				TXD1	SSO
P3_5				CMP12			SSCK
P3_4				CMP11			SCS-
P3_3	INT3-		CMP10/(INT3-)				SSI
P1_7	INT10-	CNTR00/(INT10-)					
P1_6						CLK0	
P1_5	INT11-	(INT11-)				RXD0	
P1_4						TXD0	
P1_3	KI3-		TZOUT		AN11		
P1_2	KI2-			CMP02	AN10		
P1_1	KI1-			CMP01	AN9		
P1_0	KI0-			CMP00	AN8		

R8C15端子

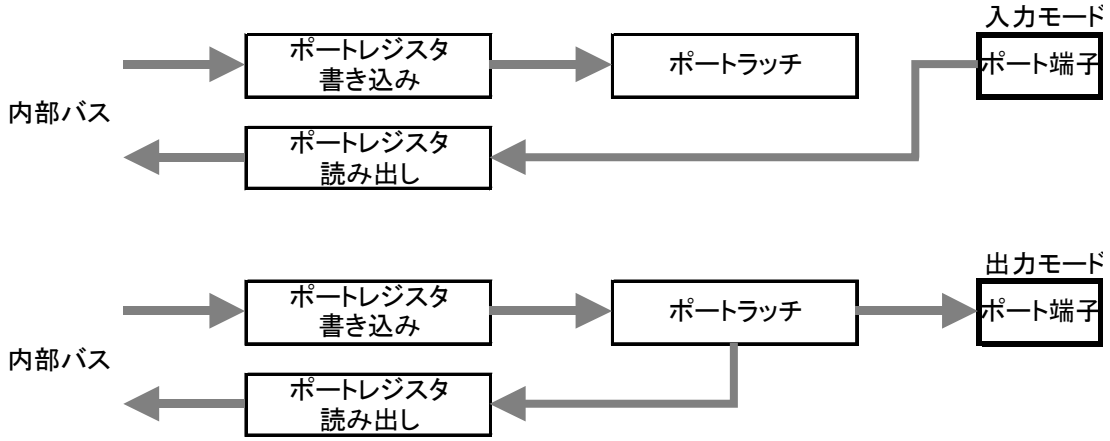
	CMP12	SSCK	P3_5	1	R8C15	20	P3_4		CMP11	SCS-
TXD1	CNTR0-	SSO	P3_7	2		19	P3_3	INT3-	CMP10	SSI
			RES-	3		18	P1_0	KI0-	CMP00	AN8
		XOUT	P4_7	4		17	P1_1	KI1-	CMP01	AN9
			GND	5		16	AVCC			
		XIN	P4_6	6		15	P1_2	KI2-	CMP02	AN10
			VCC	7		14	P1_3	KI3-	TZOUT	AN11
			MODE	8		13	P1_4			TXD0
RXD1		INT0-	P4_5	9		12	P1_5	INT11-		RXD0
	CNTR00	INT10-	P1_7	10		11	P1_6			CLK0

P4_6/P4_7 は外部発振子接続端子と共用しています。従って外部発振子を使用した場合、P4_6/P4_7 は使用できません。P4_6/P4_7 端子入力を使用する場合は、内蔵オシレータを使用してください。
シリアルデバッグまたはシリアルフラッシュＲＯＭ書き込みを行う場合には、P4_5/P3_7 端子を各 TxD1/RxD1 端子としてホスト通信に占有するため、使用できません。

各周辺 I/O 機能や端子機能は SFR レジスタにより機能の設定が行えます。
SFR とはスペシャルファンクションレジスタの頭文字で、周辺 I/O ポート制御レジスタ群を意味します。
SFR レジスタはバイト単位（一部はワード単位）、またはビット単位で設定します。
R8C15 では SFR 領域はメモリマップ上の(0000h～01FFh)の範囲に配置されていて各レジスタはメモリと同じように読み書き（アクセス）します。ただ、いちいちアドレスで指定して読み書きするのは効率的ではないので、SFR レジスタ名と SFR アドレス・ビットの対応が定義されている、SFR 定義ファイルを使用して SFR レジスタ名やビット名で周辺 I/O 制御レジスタをアクセスするようにプログラムで記述します（そのほうが見やすいため）。

2.プログラマブル入出力ポート

R8C15 ではプログラマブル入出力ポートはＰ１・Ｐ３・Ｐ４があります。
プログラマブル入出力ポートは、ビット単位（端子単位）のデジタル“Ｈ”、“Ｌ”レベルの入出力制御が出来ます。
各ポートにはポートレジスタ、ポート方向レジスタ、プルアップレジスタがあります。
ポート方向レジスタを設定すると、各ポート端子を入力として使用するか、出力として使用するかの“入出力方向”を制御します。ポート方向レジスタは各ポート端子対応のビット単位で入出力の設定ができます。
また、ポート方向レジスタはＣＰＵのリセット後は値は“０”になり、すべての端子が入力設定となります。
通常の場合、出力端子として設定したい端子のみ、ポート方向レジスタの対応ビットを“１”の出力モードに設定します。（入出力方向が固定ならば、入力モードの設定は省略できます）
入力モードのポートレジスタを読み出すと、端子に対応するビットのポート端子に印加されている電圧レベル（ロジックレベル）の“Ｌ”=0 /“Ｈ”=1 が読み出せます。
また､入力モードのポートレジスタに対して書き込みを行うと､書き込んだ値はポートラッチに書き込まれますが､ポート端子のレベルには影響しません。
出力モードのポートレジスタへ書き込みを行うと、ポートラッチにその値が書き込まれ、そのレベルの 0=“Ｌ” / 1=“Ｈ”がビットに対応するポート端子から出力されます。
出力モードのポートレジスタを読み込むとポートラッチの内容を読み出します。

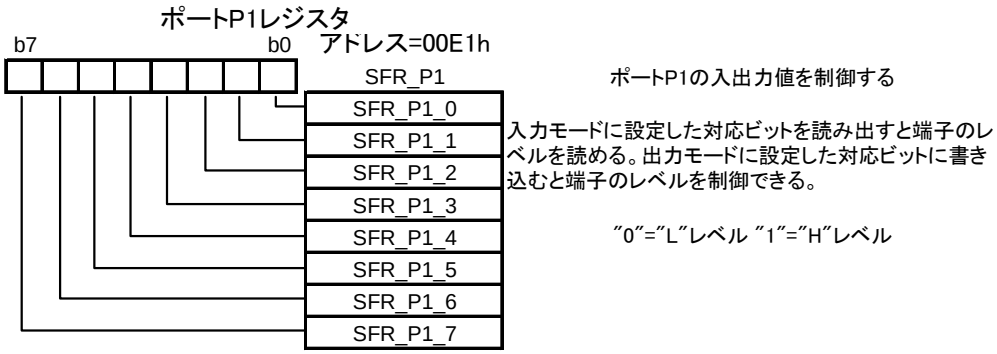


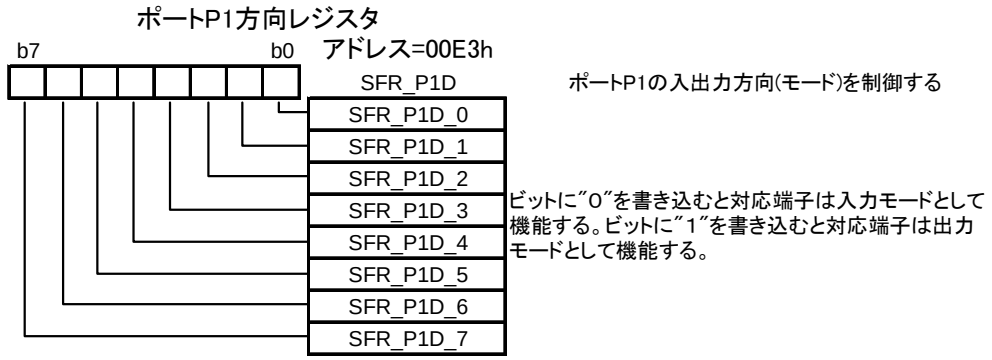
P4_6/P4_7 端子は、内部オシレータを使用する場合のみ、使用可能です。
その場合、P4_6/P4_7 端子は入力専用です。ポートラッチは存在しません。

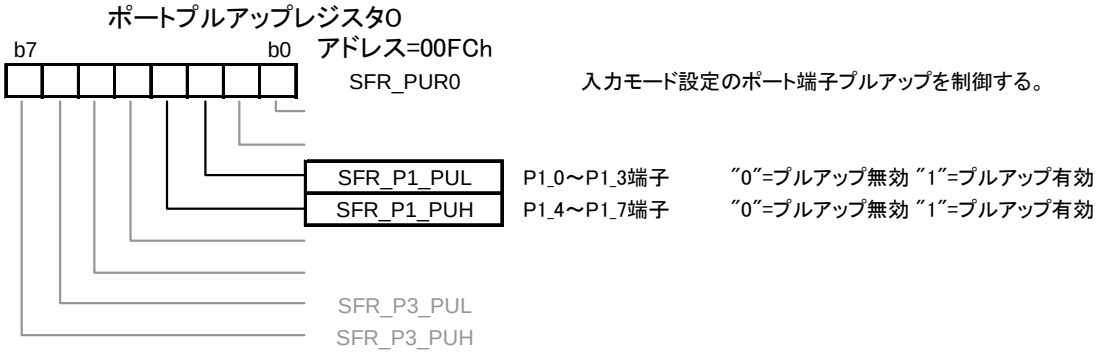
リセット後、端子を“Ｌ”レベルにせずに初期値“Ｈ”レベルから設定したい場合があります。そのときは、対応端子を外部でプルアップして（内蔵プルアップはリセット時 OFF なため使用できない)、ポート方向レジスタを出力に設定する前（入力モード）でポートレジスタの対応ビットに“１”を書き（ポートラッチに書き込まれる)、次いでポート方向レジスタに“１”を書き込み出力モードにします。これにより、リセット時に不用意なパルスが発生しない負論理出力が実現できます。
ただし、同じポートの複数の端子を操作する場合は、ポートレジスタへの書き込みは”SFR_P1 |= 00000110b;”のようにビット単位でなくバイト単位で記述してください。
ビット単位での操作はそのポートをまず読んで対応ビットを操作してポートに書き込むという動作（リードモディファイライト）を行うので、入力モードではポートラッチ値を書き込んだでも読み込み時は端子状態を読んでしまいます。したがって、入力モード時、”SFR_P1_1=1; SFR_P1_2=1; SFR_P1D_1=1; SFR_P1D_2=1;”のようにポートレジスタへ複数回ビット操作を行った場合、最後のビット操作のみが有効となってしまい、意図した動作になりません。したがって、入力モード時複数ビットのポートレジスタ設定には、バイト単位で一回のみで書き込みを行ってください。

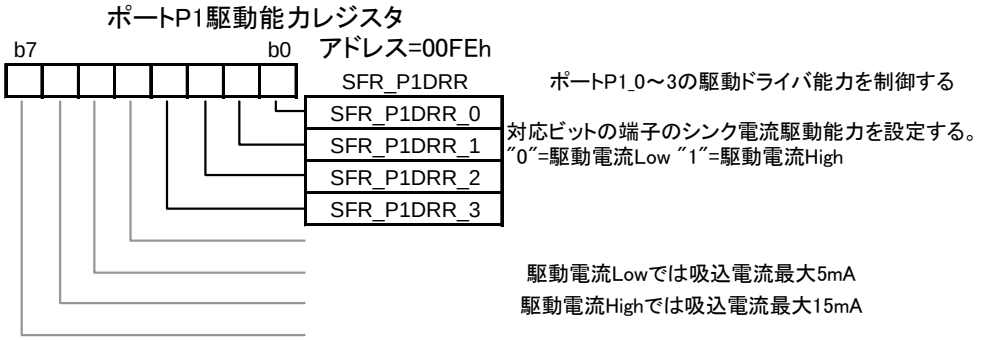
割込み入力やタイマ出力、AD 入力、シリアル入出力などを使用する場合、その周辺 I/O に定義する端子の対応するポート方向レジスタのビットは“０”の入力に設定してください。(CMP00～CMP12 出力を除く）
このとき、対応するポート入力を行うと、対応ビットに端子状態（端子入出力レベル）を読み込むことが出来ます。
また、それらの周辺Ｉ／Ｏ機能端子を使用する場合、入力端子に対してポートプルアップレジスタで設定したプルアップ指定は有効です。ＵＡＲＴのＴｘＤ端子をオープンドレイン出力で使用する場合、この端子をプルアップレジスタで指定してプルアップすることは出来ません（出力端子に対してはプルアップできないので）ご注意ください。

2-1 ポートＰ１







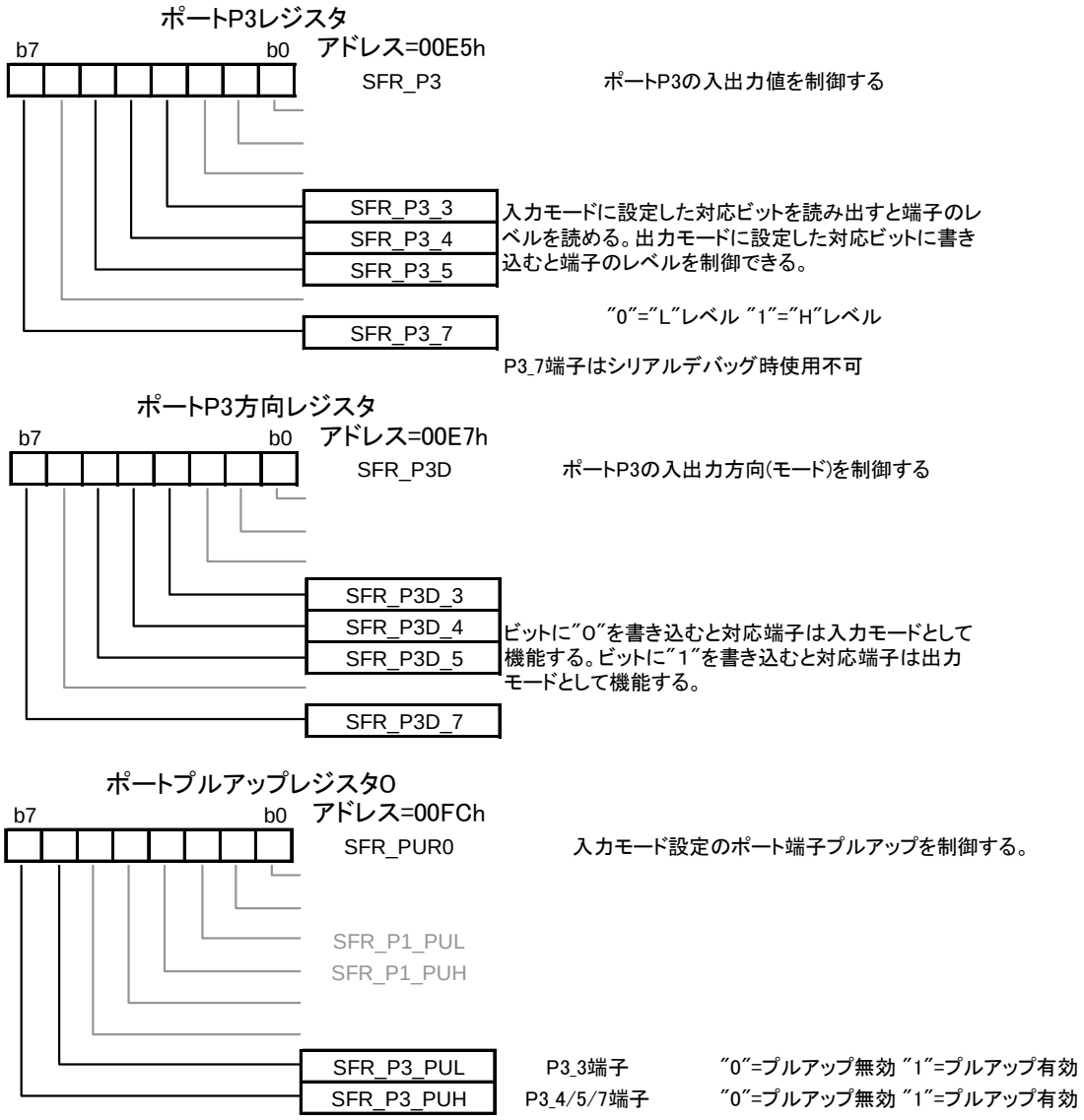


設定例

```
// ポート P1_1 P1_2 を出力に設定して出力レベルを P1_1=“L” P1_2=“H”にします。
        SFR_P1D_1=1;      // P1_1 を出力に設定
        SFR_P1D_2=1;      // P1_2 を出力に設定
        SFR_P1_1=0;       // P1_1 の出力を“L”
        SFR_P1_2=1;       // P1_2 の出力を“H”
```

2-2 ポートＰ３

ポートＰ３のうち P3_7 はシリアルデバッグ時に RxD1 端子として使用されます。シリアルデバッグは R8C の最も強力な武器です。P3_7 はできるだけ使用せず、シリアルデバッグにあけておくことをお勧めします。

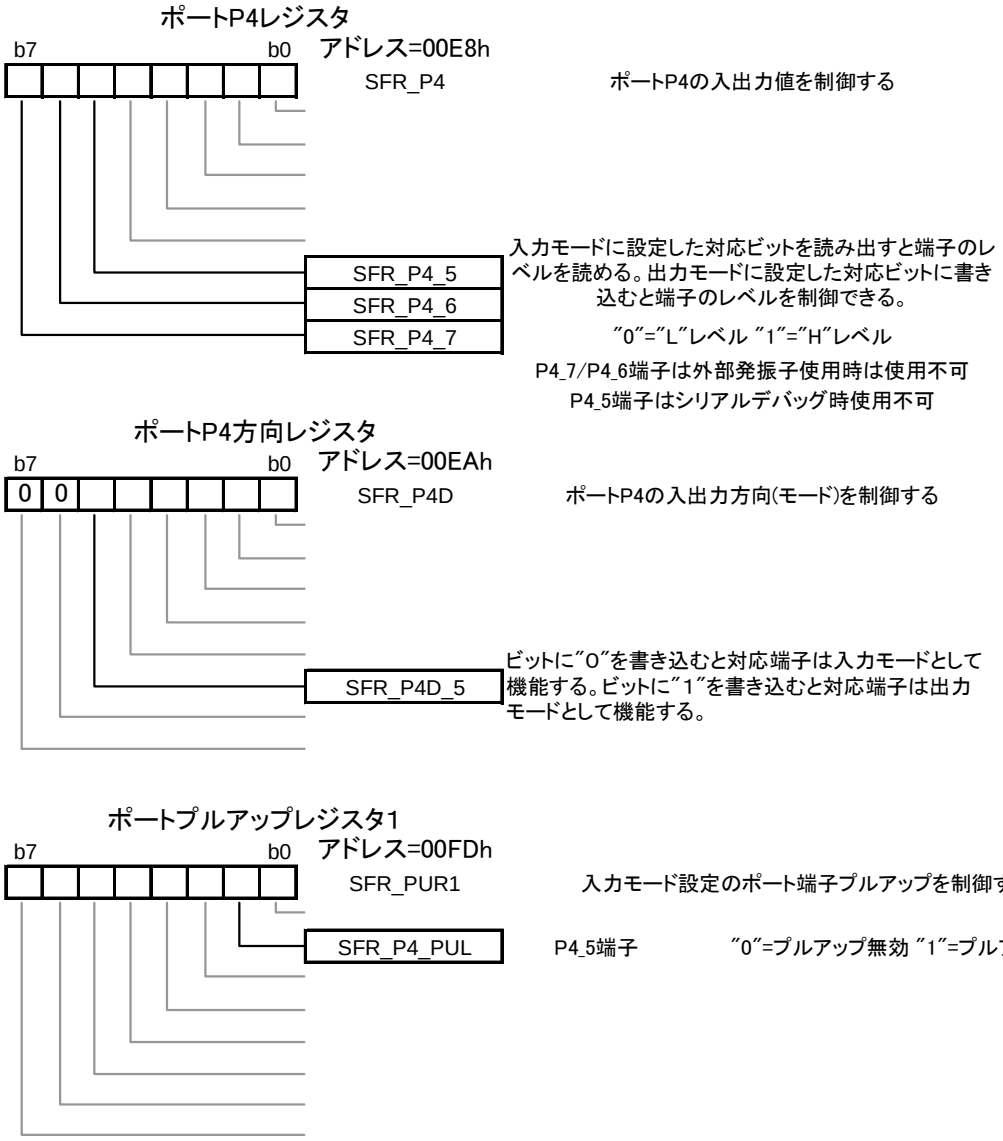


設定例

```
// ポート P3_3 にプッシュスイッチが付いている仮定で、P3_3 を入力モード、プルアップ有効にして
// スイッチが押されているかチェックする。
        SFR_P3_PUL=1;     // P3_3 端子プルアップ有効
//      SFR_P3D_3=0;      // P3_3 を入力に設定（この行は省略可）
                …
        if (SFR_P3_3==0) // P3_3 が“L”か？(スイッチが押されているか？)
        {
                // P3_3 が“L”の場合の処理
        }


// ポート P3_4 P3_5 を出力にして”H”状態に初期化してから、P3_5 のみ“L”にする
        SFR_P3 |=0b00110000;      // P3_4 P3_5 出力ラッチを“H”にする
        SFR_P3D |=0b00110000;     // P3_4 P3_5 を出力に設定（出力は“H”状態）
                …
        SFR_P3_5=0;               // P3_5 を”L”にする
```

2-3 ポートＰ４

ポートＰ４のうち P4_5 はシリアルデバッグ時に TxD1 端子として使用されます。シリアルデバッグは R8C の最も強力な武器です。P4_5 はできるだけ使用せず、シリアルデバッグにあけておくことをお勧めします。
P4_6/P4_7 をプログラマブル入出力ポートとして使用する場合には発振子を外部に接続せず、内部オシレータを使用することで P4_6/P4_7 を入力ポートとして使用することができます。が、シリアルデバッグおよびシリアルでのフラッシュ ROM 書き込みは出来なくなりますので、出来るだけ使用しない事をお勧めします。
ということで、ポートＰ４は使用しないほうが良いと思います。(ポートが足りない場合の最後の手段ということで)



3.タイマ

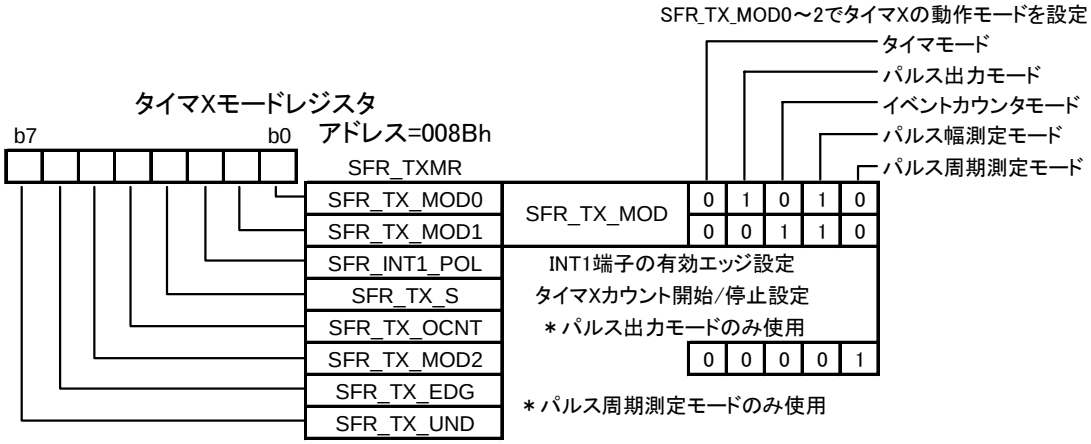
R8C15 にはタイマＸ、タイマＺ、タイマＣの３つのタイマを搭載しています。
それぞれのタイマで機能が異なります。

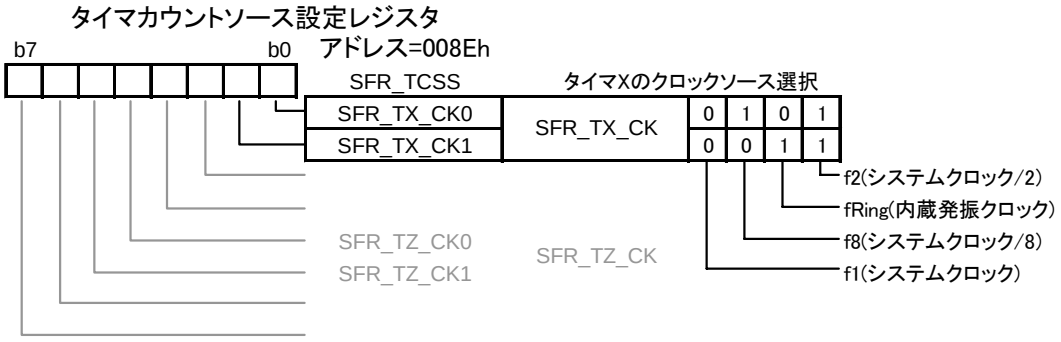
3-1 タイマＸ

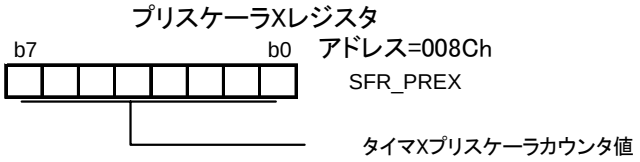
タイマＸは動作モードとして、タイマモード、パルス出力モード、イベントカウンタモード、パルス幅測定モード、パルス周期測定モードがあります。

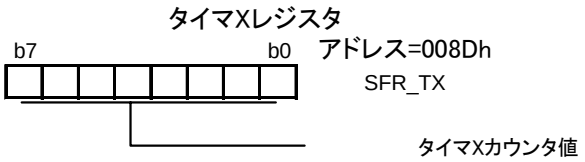
- タイマモード　　連続して内部カウントソースをダウンカウントする
- パルス出力モード　　連続して内部カウントソースをダウンカウントし、アンダーフローで出力を反転したパルスを発生する
- イベントカウンタモード　外部パルスをダウンカウントする
- パルス幅測定モード　　外部パルスの“Ｌ”幅または“Ｈ”幅を測定する
- パルス周期測定モード　　外部パルスの立上り～立上りまたは立下り～立下りの間隔を測定する

レジスタによりモードを設定し、また、モードにより設定するレジスタ値が異なります。







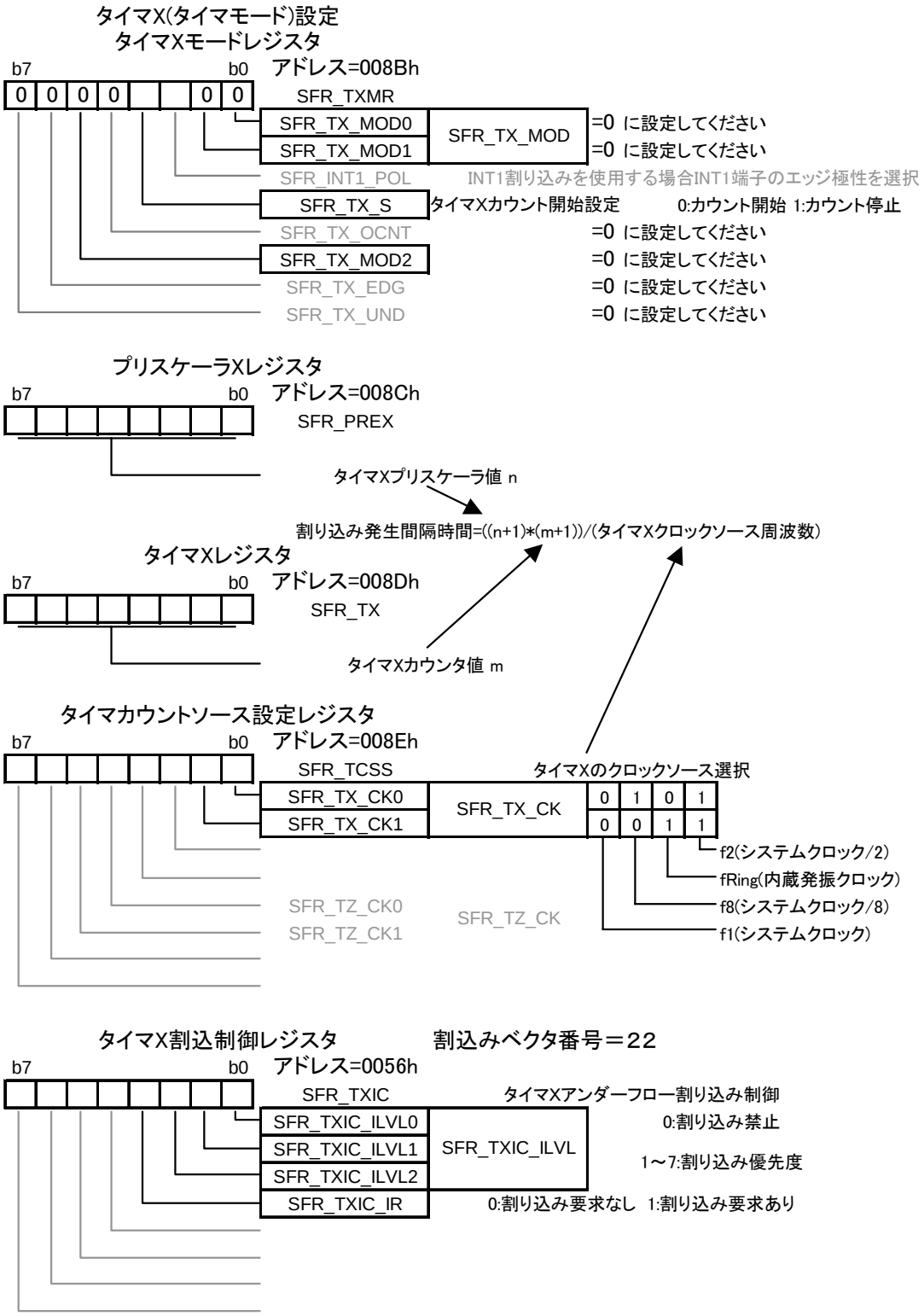


タイマX割込制御レジスタ
割込みベクタ番号＝22
b7
b0
アドレス=0056h
SFR_TXIC
タイマXアンダーフロー割り込み制御
SFR_TXIC_ILVL0
SFR_TXIC_ILVL1
SFR_TXIC_ILVL2
SFR_TXIC_ILVL
0:割り込み禁止
1～7:割り込み優先度
SFR_TXIC_IR
0:割り込み要求なし 1:割り込み要求あり

UART0送受信制御レジスタ
b7
b0
アドレス=00B0h
SFR_UCON
SFR_U0_IRS
SFR_U0_RRM
SFR_U0_SEL0
SFR_U0_SEL1
SFR_INT1_SEL
INT1入力端子選択
0:INT10-(P1_7)端子選択 1:INT11-(P1_5)端子選択

3-1-1 タイマＸ（タイマモード）

タイマモードは指定したタイマＸクロックソースをタイマＸレジスタ値、プリスケーラＸレジスタ値で分周して、その分周比による一定時間隔でタイマＸアンダーフロー割り込みを発生させて使用します。
通常は、割り込み制御によるインターバルタイマとして使用します。
INT1 端子入力、CNTR00/CNTR01/CNTR0-出力端子は使用しませんので各端子はプログラマブル入出力端子として使用できます。

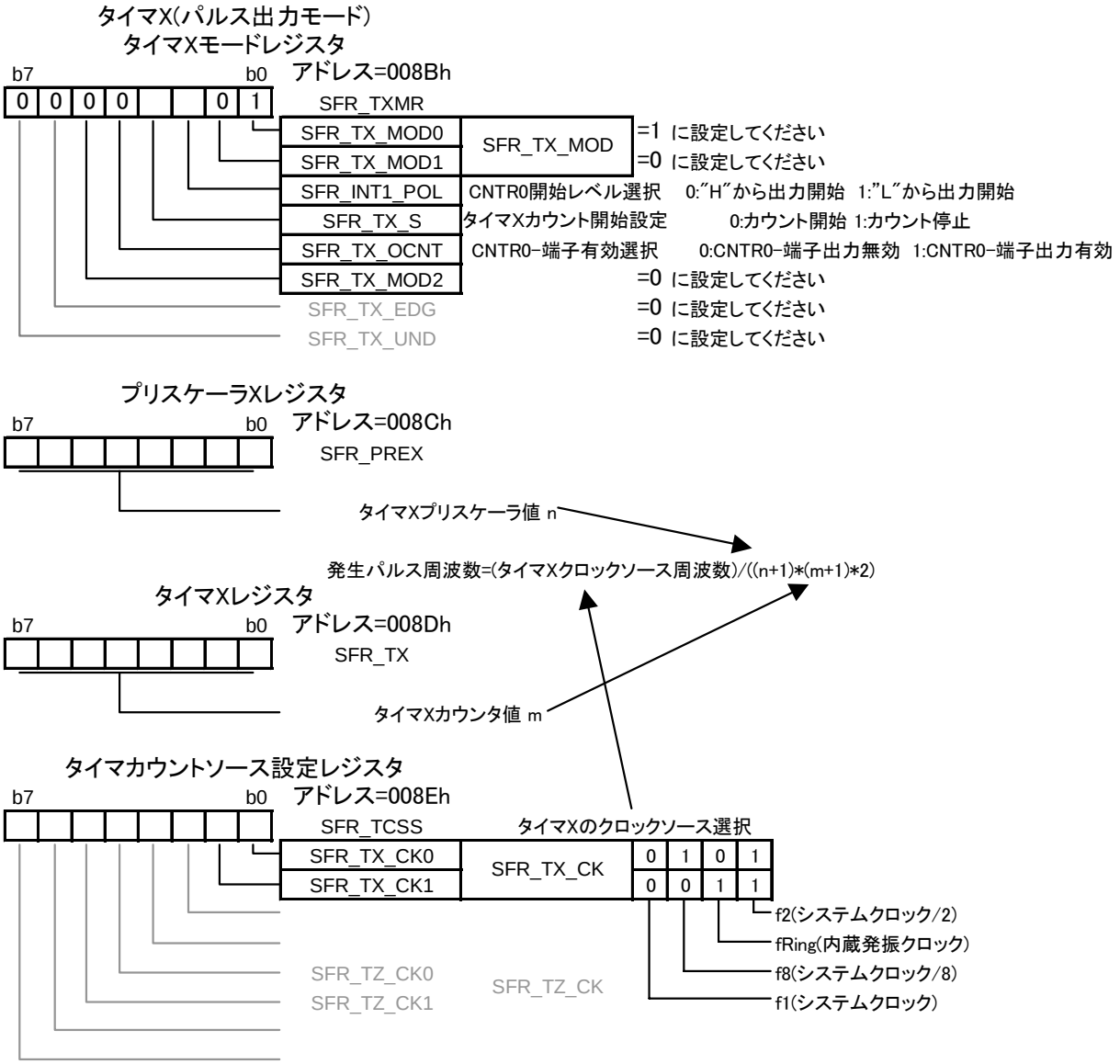


記述例

```
// １m秒間隔でタイマ X アンダーフロー割り込みを発生させる。
// タイマーX 初期化
void tx_init(void)
{
        SFR_TXMR=0;
        SFR_TX_CK=1;      // タイマ X クロックソース= f8 (20MHz/8=2.5MHz)
        SFR_PREX=250-1;   //
        SFR_TX=10-1;      // インターバル時間 t=(250*10)/2500000=1mS
        SFR_TXIC=6;       // 割り込みレベル 6（割込みレベル値は任意）
        SFR_TX_S=1;       // タイマ X カウントスタート
}

// タイマ X アンダーフロー割り込み（1mS インターバル割込み）
void tx_int(void)
#pragma interrupt tx_int (vect=22)
{
        // インターバル割り込み処理を記述する
}

void main(void)
{
        tx_init();        // タイマ X 初期化
        ei();             // 割り込み許可

        // メイン処理を記述

}
```

3-1-2 タイマＸ（パルス出力モード）

パルス出力モードは指定したタイマＸクロックソースをタイマＸレジスタ値、プリスケーラＸレジスタ値で分周して、その分周比によりタイマＸアンダーフロー割り込みを発生させ、同時に CNTR00 端子出力を反転します。このモードでは CNTR00 端子(P1_7 端子)は必ず出力されます。また CNTR0 を反転した CNTR0-端子(P3_7 端子)出力は設定により出力できます。



UART 送受信制御レジスタ SFR_UCON の CNTR0 信号端子選択ビットは入力すなわち INT1 入力のみに有効で、このモードではこのビットにかかわらず、必ず CNTR00 端子(P1_7 端子)から出力されます。

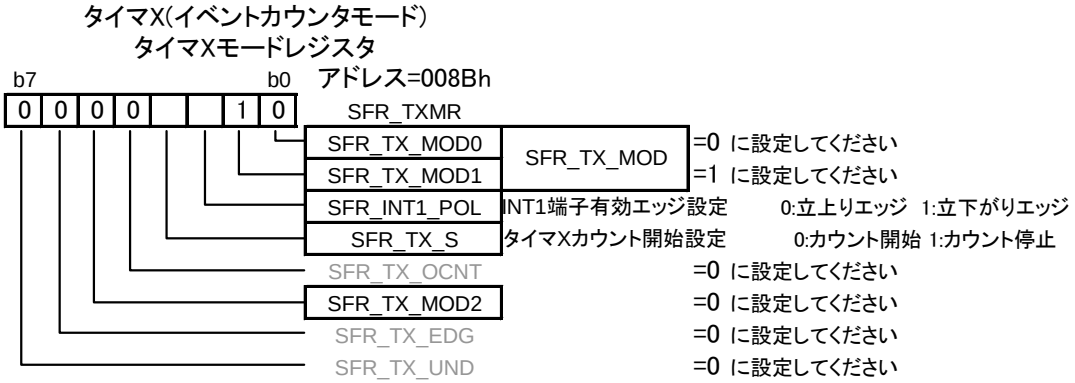
CNTR00 端子を出力として使用するので、ポートＰ１方向レジスタの P1_7 対応ビットを“０”=入力に設定してください。

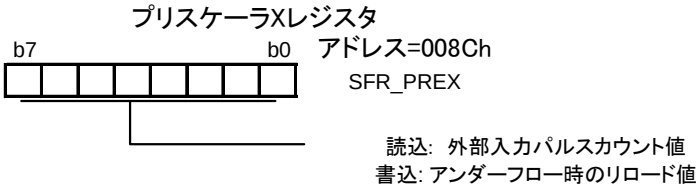
設定例

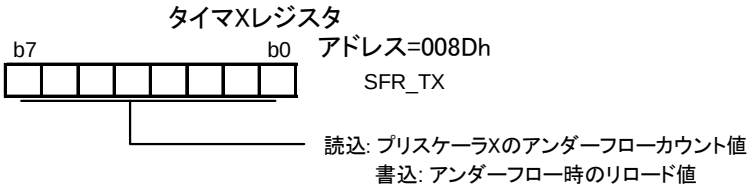
```
// CNTR00(P1_7)端子に 1Hz のクロックパルスを出力する
//      SFR_P1D_7=0;      // P1_7 を入力端子に設定（この行は省略可）
        SFR_TXMR=1;       // タイマ X パルス出力モード CNTR0-端子無効
        SFR_TX_CK=2;      // タイマ X クロックソース= fRing (125KHz 低速オシレータ)
        SFR_PREX=0xf9;    //
        SFR_TX=0xf9;      // パルス周波数 f=125KHz/(250*250*2)=1Hz
        SFR_TX_S=1;       // タイマ X カウントスタート(CNTR0 端子からパルス出力開始)
```

3-1-3 タイマＸ （イベントカウンタモード）

イベントカウンタモードは、INT1 端子の指定エッジをプリスケーラ X カウンタとタイマ X カウンタでダウンカウントします。（タイマＸのカウントソースに INT1 端子入力が使用されます。）プリスケーラ X レジスタ、タイマ X レジスタに書き込んだ値は初期、または各カウンタがアンダーフローしたときにプリロードされます。またタイマ X カウンタがアンダーフローすると、タイマ X アンダーフロー割り込みが発生します。







このイベントカウンタモードでは、タイマカウントソース設定レジスタの設定は必要ありません。（設定は無効です）

設定例

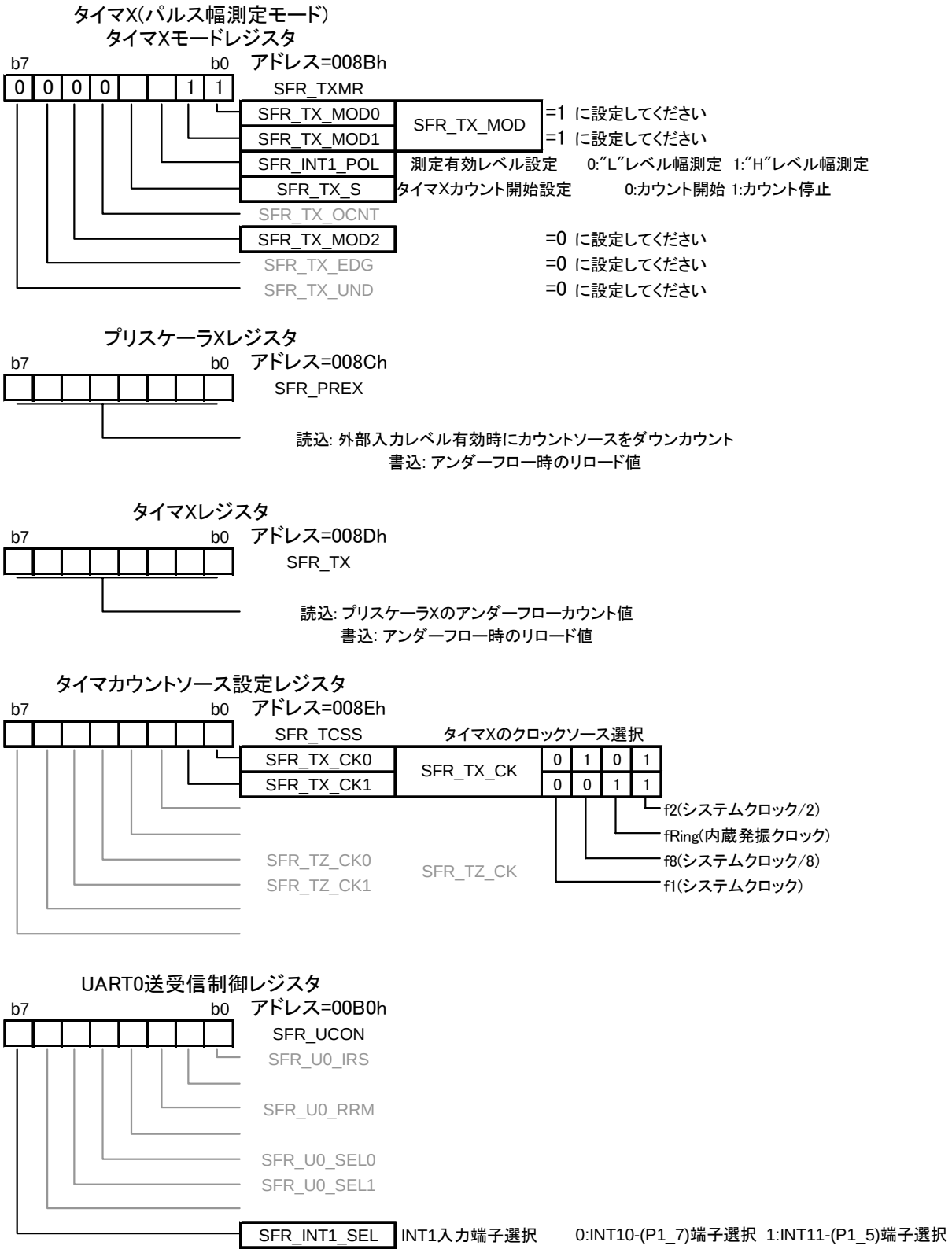
```
// INT1-端子入力の立下がりエッジをカウントして任意なときにカウント値を読み取る
        unsigned int event_count;
//      SFR_P1D_7=0;            // P1_7(INT1-)端子入力設定(この行は省略可能)
        SFR_TXMR=0b00000110;    // タイマ X イベントカウンタモード INT1:立下りエッジ
//      SFR_INT1_SEL=0;         // INT10(P1_7)端子有効（この行は省略可能）
        SFR_PREX=0xff;
        SFR_TX=0xff;            // PREX/TX を-1 に初期化
        SFR_TX_S=1;             // タイマ X カウントスタート
                ...
                ...
                ...
        event_count=0xffff*((unsigned int *)&SFR_PREX);   // カウント値取り出し
        SFR_PREX=0xff;
        SFR_TX=0xff;            // PREX/TX を-1 に初期化
```

記述例

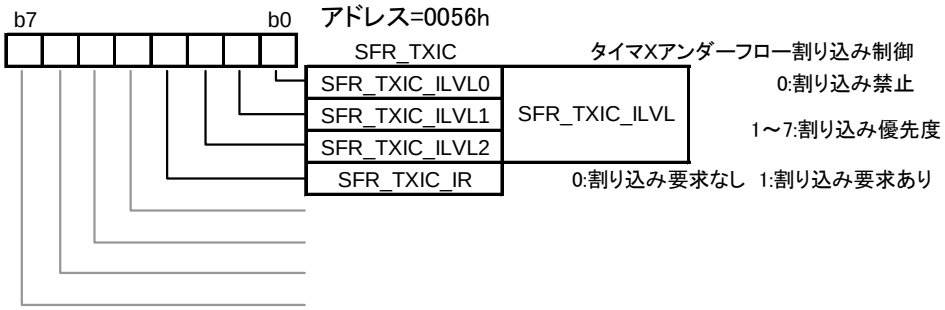
```
// INT1-端子入力の立上がりエッジ１００個検出でタイマ X アンダーフロー割り込みを発生させる。
// タイマーX 初期化
void tx_init(void)
{
//      SFR_P1D_7=0;            // P1_7(INT1-)端子入力設定(この行は省略可能)
        SFR_TXMR=0b00000010;    // タイマＸイベントカウンタモード INT1:立上りエッジ
//      SFR_INT1_SEL=0;         // INT10(P1_7)端子有効（この行は省略可能）
        SFR_PREX=0;     //
        SFR_TX=100-1;   // 100
        SFR_TXIC=6;     // 割り込みレベル 6
        SFR_TX_S=1;     // タイマ X カウントスタート
}

// タイマ X アンダーフロー割り込み（イベント１００回）
void tx_int(void)
#pragma interrupt tx_int (vect=22)
{
        // INT1-端子の立下りエッジ１００個目で割り込み発生
        // 割り込み処理を記述する
}

void main(void)
{
        tx_init();      // タイマ X 初期化
        ei();           // 割り込み許可

        // メイン処理を記述

}
```

3-1-4 タイマＸ （パルス幅測定モード）

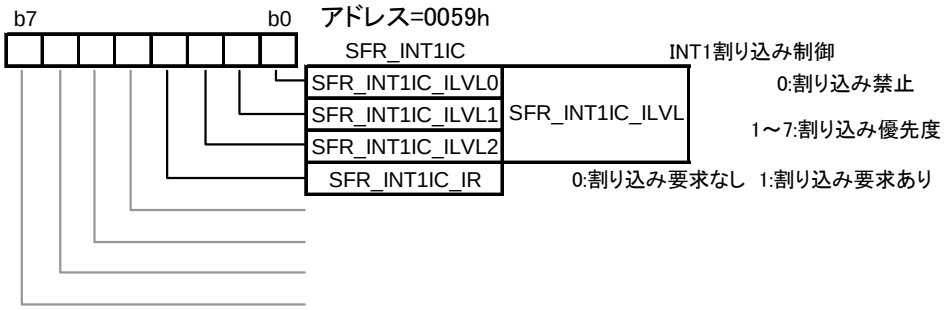
パルス幅測定モードは INT1 端子の指定レベルでカウントソースをゲートしてをプリスケーラ X カウンタとタイマ X カウンタでダウンカウントします。プリスケーラ X レジスタ、タイマ X レジスタに書き込んだ値は初期、または各カウンタがアンダーフローしたときにリロード（再設定）されます。またタイマ X カウンタがアンダーフローすると、タイマ X アンダーフロー割り込みが発生します。



タイマX割込制御レジスタ　　割込みベクタ番号＝22



INT1割込制御レジスタ　　割込みベクタ番号＝25



記述例

```
//
void tx_init(void)
{
//      SFR_P1D_7=0;            // P1_7(INT1-)端子入力設定(この行は省略可能)
        SFR_TXMR=0b00000011;    // タイマＸパルス幅測定モード INT1 端子“ L ”レベル幅測定
//      SFR_INT1_SEL=0;         // INT10(P1_7)端子有効（この行は省略可能）
        SFR_TX_CK=1;            // タイマ X クロックソース= f8 (20MHz/8=2.5MHz)
        SFR_PREX=0xff;
        SFR_TX=0xff;            // PREX/TX を-1 に初期化
        SFR_INT1IC=5;           // INT1 割り込みレベル５
        SFR_TXIC=5;             // タイマＸ割り込みレベル５
        SFR_TX_S=1;             // タイマ X カウントスタート
}

// パルス幅値格納変数をグローバルで定義
unsigned int p_width;

//
void int1_int(void)
#pragma interrupt int1_int (vect=25)
{
        // パルス検出割り込み
        p_width=0xffff-*((unsigned int *)&SFR_PREX);      //16 ビット値でカウント値を取り込み
        SFR_PREX=0xff;
        SFR_TX=0xff;            // PREX/TX を-1 に初期化
}

//
void tx_int(void)
#pragma interrupt (vect=22)
{
        // パルス幅オーバーフロー処理

}

void main(void)
{
        tx_init();      //
        ei();

        // メイン処理

}
```

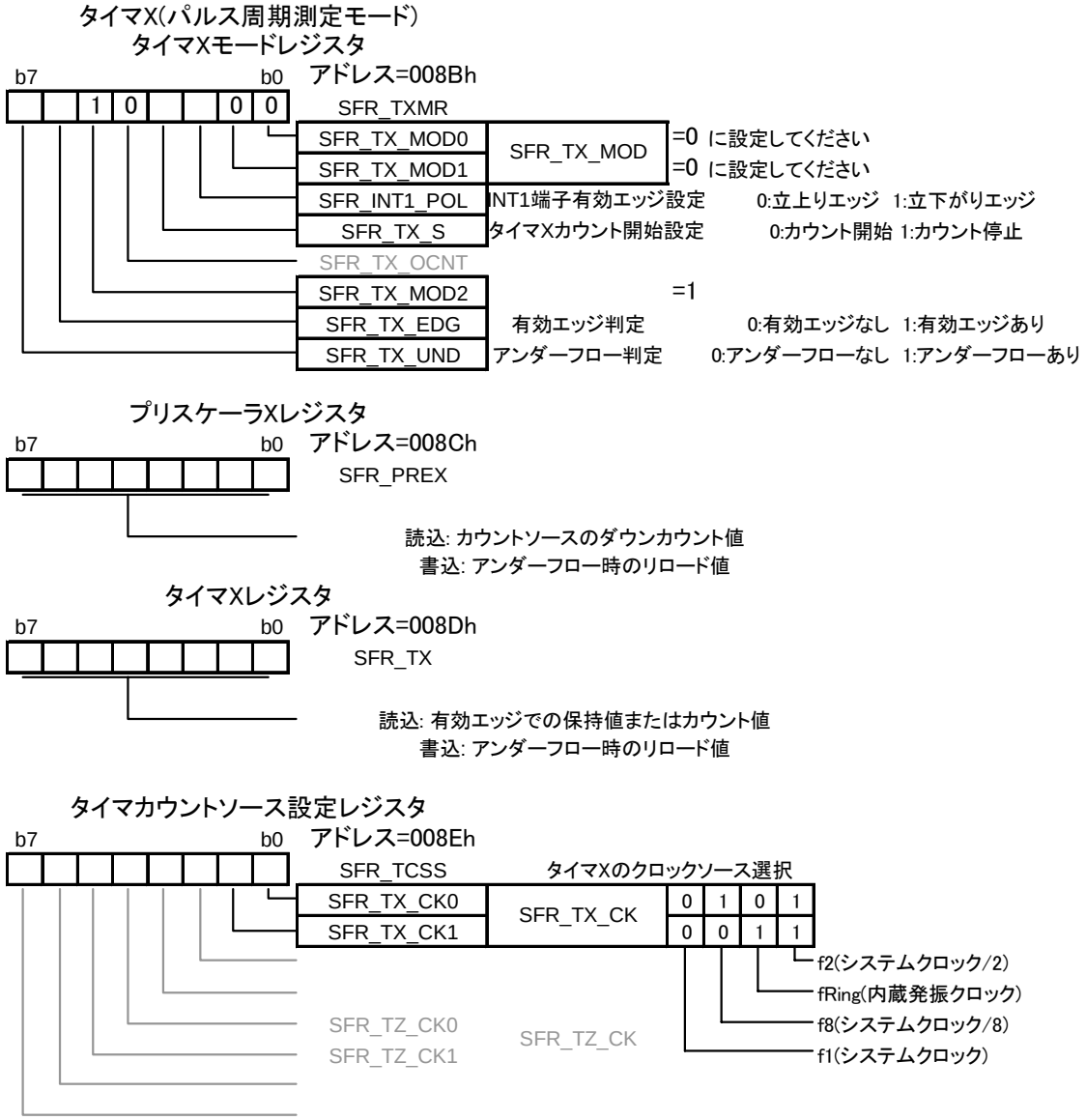
注意）上記のパルス幅測定では被測定パルスレベル（上記では“Ｌ”）でないレベルの幅が短い（割込みレスポンス＋α　1.5uS(20MHz 動作時)以下の間隔の）場合は正常動作しない。

3-1-5 タイマＸ （パルス周期測定モード）

パルス周期測定モードは INT1 端子入力の立上りエッジ～立上りエッジまたは立下りエッジ～立下りエッジの間のクロックソースのダウンカウント値を読み出します。INT1 端子有効エッジ検出後のプリスケーラＸカウンタのアンダーフロー時にタイマーＸカウンタの値は読み出しバッファ（タイマＸレジスタ）に、タイマＸレジスタが読込まれるまで保持されます（プリスケーラＸカウント値は保持されません）。同時にタイマＸ割り込みが発生します。また、このとき（実際には次のプリスケーラＸカウンタアンダーフロー時）にタイマーＸカウンターはリロードされ、ダウンカウントを継続します。

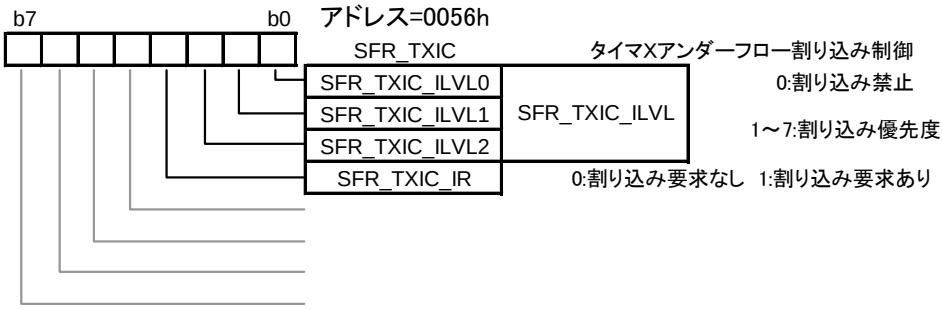
タイマーＸカウンタがアンダーフローすると、タイマーＸ割り込みが発生します。タイマーＸ割り込みが発生したとき、SFR_TX_EDG/SFR_TX_UND フラグを参照することで、INT1 端子有効エッジでの割り込みか、アンダーフローでの割り込みかが、判断できます。

詳細は R8C15 ハードウェアマニュアルを参照してください。

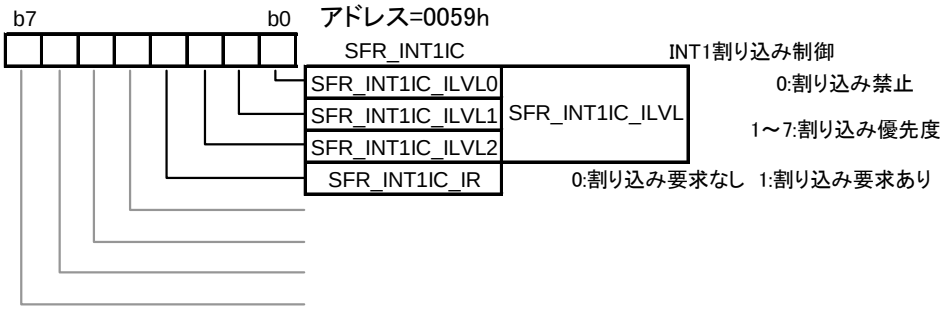


UART0送受信制御レジスタ

タイマX割込制御レジスタ　　割込みベクタ番号＝22



INT1割込制御レジスタ　　割込みベクタ番号＝25



記述例

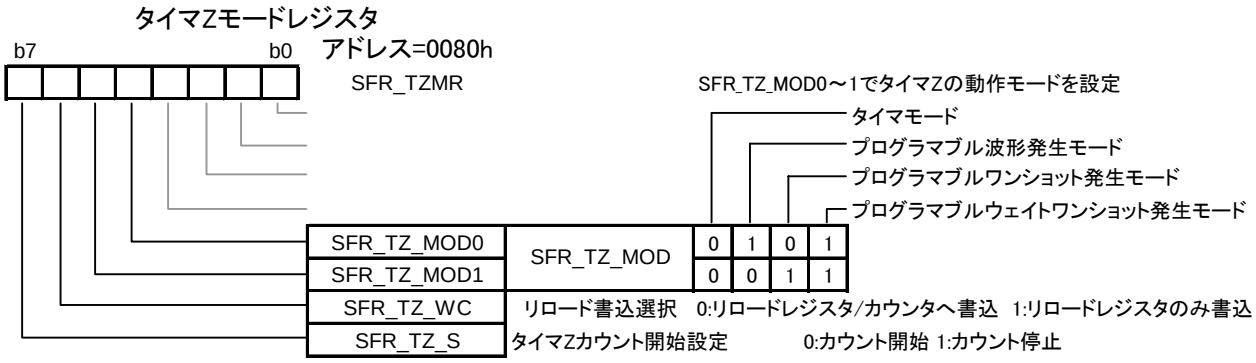
```
//
void tx_init(void)
{
//      SFR_P1D_7=0;            // P1_7(INT1-)端子入力設定(この行は省略可能)
        SFR_TXMR=0b00100100;    // タイマＸパルス周期測定モード INT1 端子立下りエッジ測定
        SFR_INT1_SEL=1;         // INT11(P1_5)端子有効
        SFR_TX_CK=1;            // タイマＸクロックソース= f8 (20MHz/8=2.5MHz)
        SFR_PREX=0x63;          // (99+1)/2.5MHz=100uS
        SFR_INT1IC=5;           // INT1 割り込みレベル５
        SFR_TXIC=5;             // タイマＸ割り込みレベル５
        SFR_TX_S=1;             // タイマＸカウントスタート
}

// グローバル変数定義
unsigned char pulse_prod;       // パルス幅計測結果格納 *100uS
unsigned char pulse_top;
unsigned char overflow_count;
unsigned char pulse_temp;

//
void int1_int(void)
#pragma interrupt int1_int (vect=25)
{
        pulse_temp=SFR_TX;
        if (pulse_temp<pulse_top)
                overflow_count--;
        if (overflow_count!=0)
                pulse_prod=0;           // 測定不能
        else
                pulse_prod=pulse_temp-pulse_top;  // 測定結果
        pulse_top=pulse_temp;
        overflow_count=0;
}

//
void tx_int(void)
#pragma interrupt tx_int (vect=22)
{
        overflow_count++;
        if (overflow_count==0)
                overflow_count=0xff;
}

//
void main(void)
{
        overflow_count=0xff;
        pulse_top=0;
        pulse_prod=0;
        tx_init();
        ei();           // 割込み許可

        // メイン処理
}
```

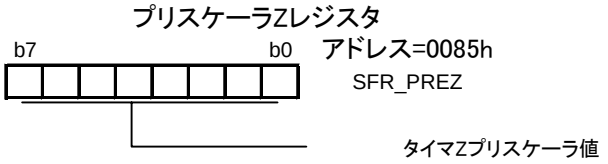
3-2 タイマＺ

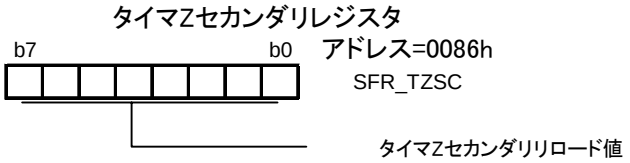
タイマＺは動作モードとして、タイマモード、プログラマブル波形発生モード、プログラマブルワンショット発生モード、プログラマブルウェイとワンショット発生モードがあります。

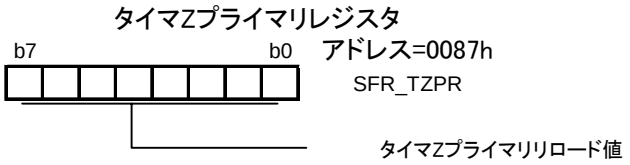
- タイマモード　連続して内部カウントソースをダウンカウントする。
- プログラマブル波形発生モード　連続して内部カウントソースをプライマリ値・セカンダリ値を交互にダウンカウントし、アンダーフローで出力を反転したパルスを発生する。
- プログラマブルワンショット発生モード　外部トリガによりプライマリ値で設定した単発のパルスを発生する。パルスは１回のみ外部トリガから内部カウントソースをダウンカウントしアンダーフローまで出力される。
- プログラマブルウェイトワンショット発生モード　外部トリガによりプライマリ値で設定した時間待った後、セカンダリ値で設定した単発のパルスを発生する。外部トリガによりプライマリ設定値分ダウンカウントしてアンダーフローしたら、そこからセカンダリ設定値分ダウンカウントしアンダーフローするまでのパルスが１回出力される。

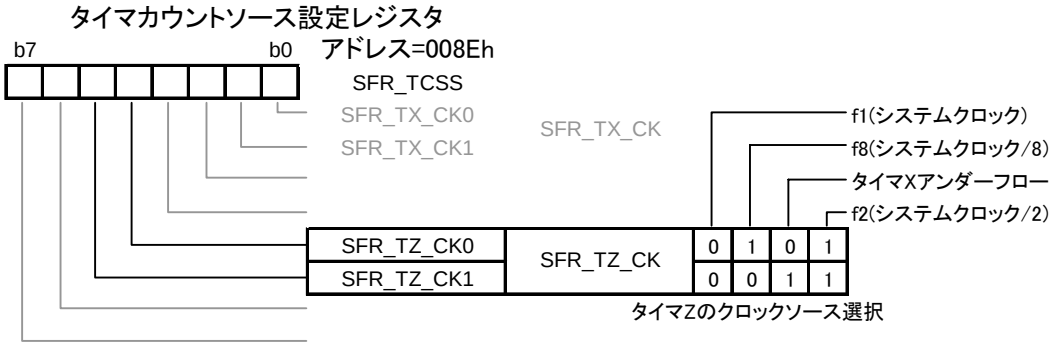
レジスタによりモードを設定し、また、モードにより設定するレジスタ値が異なります。

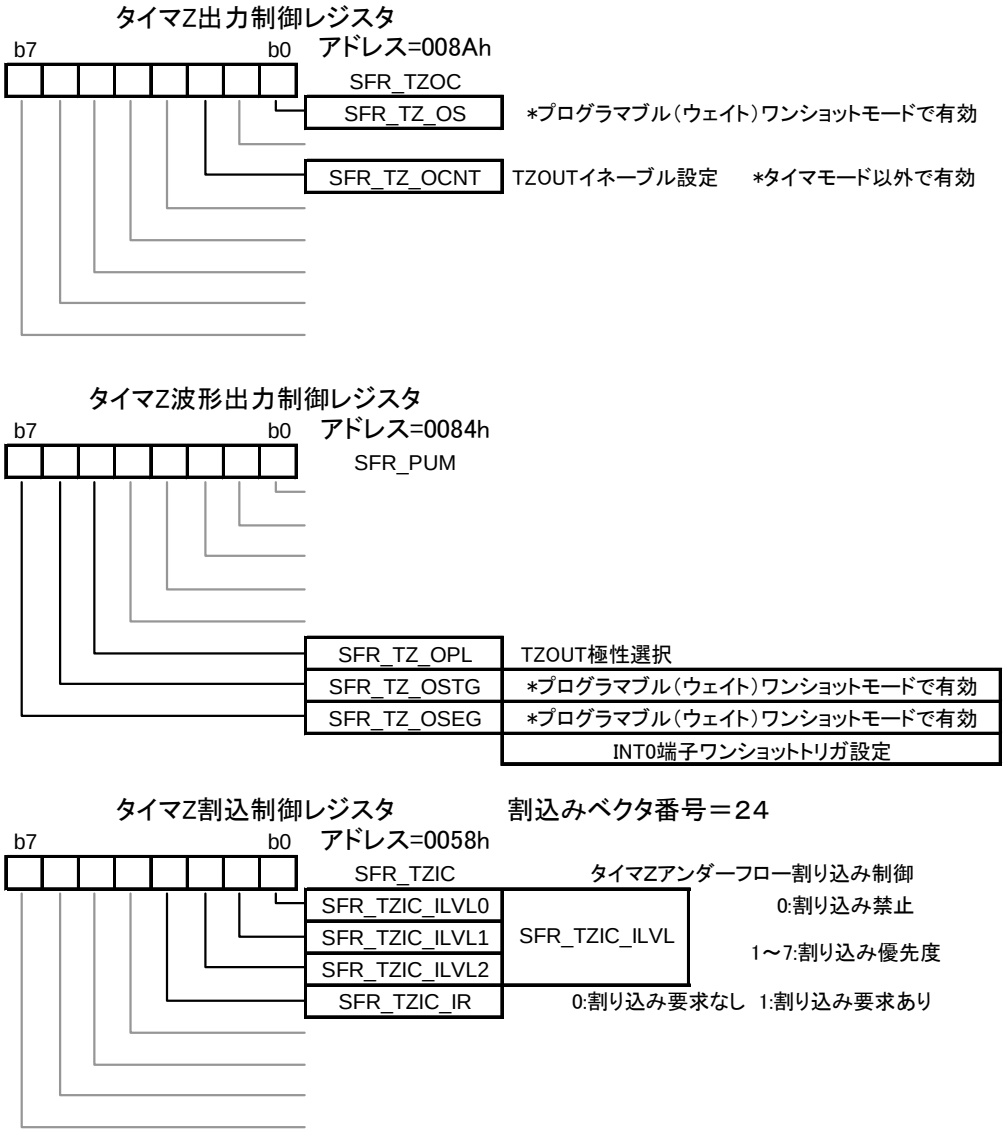










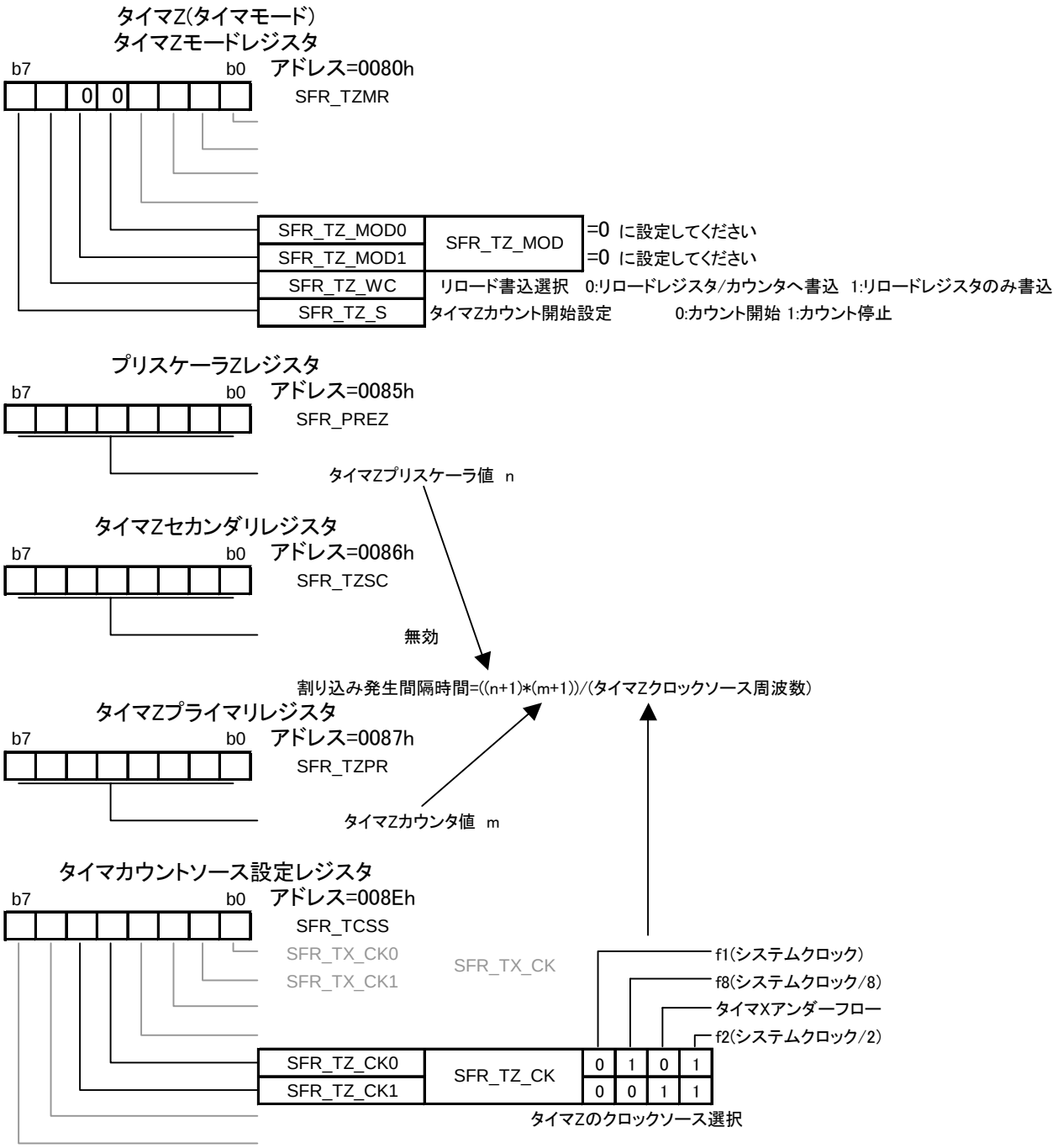


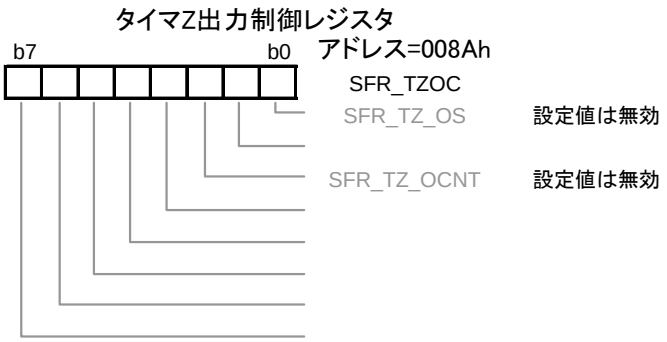
タイマＺセカンダリレジスタ書き込みの際の注意
タイマＺセカンダリレジスタ(SFR_TZSC)書込み後に、タイマＺプライマリレジスタ(SFR_TZPR)を書き込んでください。(タイマＺセカンダリレジスタ(SFR_TZSC)書込み後、タイマＺプライマリレジスタ(SFR_TZPR)を書き込まないと書き込まれたタイマＺセカンダリレジスタが有効になりません)。タイマＺセカンダリレジスタ書込み値は、タイマＺプライマリレジスタ書込み後の次のプライマリ期間で値が有効になります。

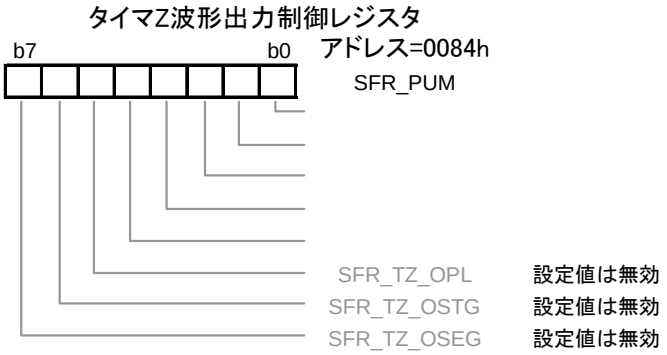
タイマ Z ではタイマモード以外で SFR_TZ_OCNT により TZOUT 出力の有効無効を設定できます。
SFR_TZ_OCNT=0 の場合 TZOUT 出力は有効です。SFR_TZ_OCNT=1 の場合、TZOUT 出力はタイマ Z 出力ではなく、P1_3 ポートラッチの値が出力されます。(TZOUT 端子出力を使用するために、P1_3 端子は入力設定(SFR_P1D_3=0)にしているはずですが、この状態の場合は常に端子出力モードとして扱われます。)

3-2-1 タイマＺ（タイマモード）

タイマモードは指定したタイマＺクロックソースをタイマＺプライマリレジスタ値、プリスケーラＺレジスタ値で分周して、その分周比による一定時間でタイマＺアンダーフロー割り込みを発生します。
通常は、割り込みによるインターバルタイマとして使用します。
タイマモードでは TZOUT 端子は使用されませんので、プログラマブル入出力ポート P1_3 端子として使用できます。







タイマモードでは、タイマＺセカンダリレジスタ(SFR_TZSC)、タイマＺ出力制御レジスタ(SFR_TZOC)、タイマＺ波形出力レジスタ(SFR_PUM)は使用しません。（設定の必要はありません。）

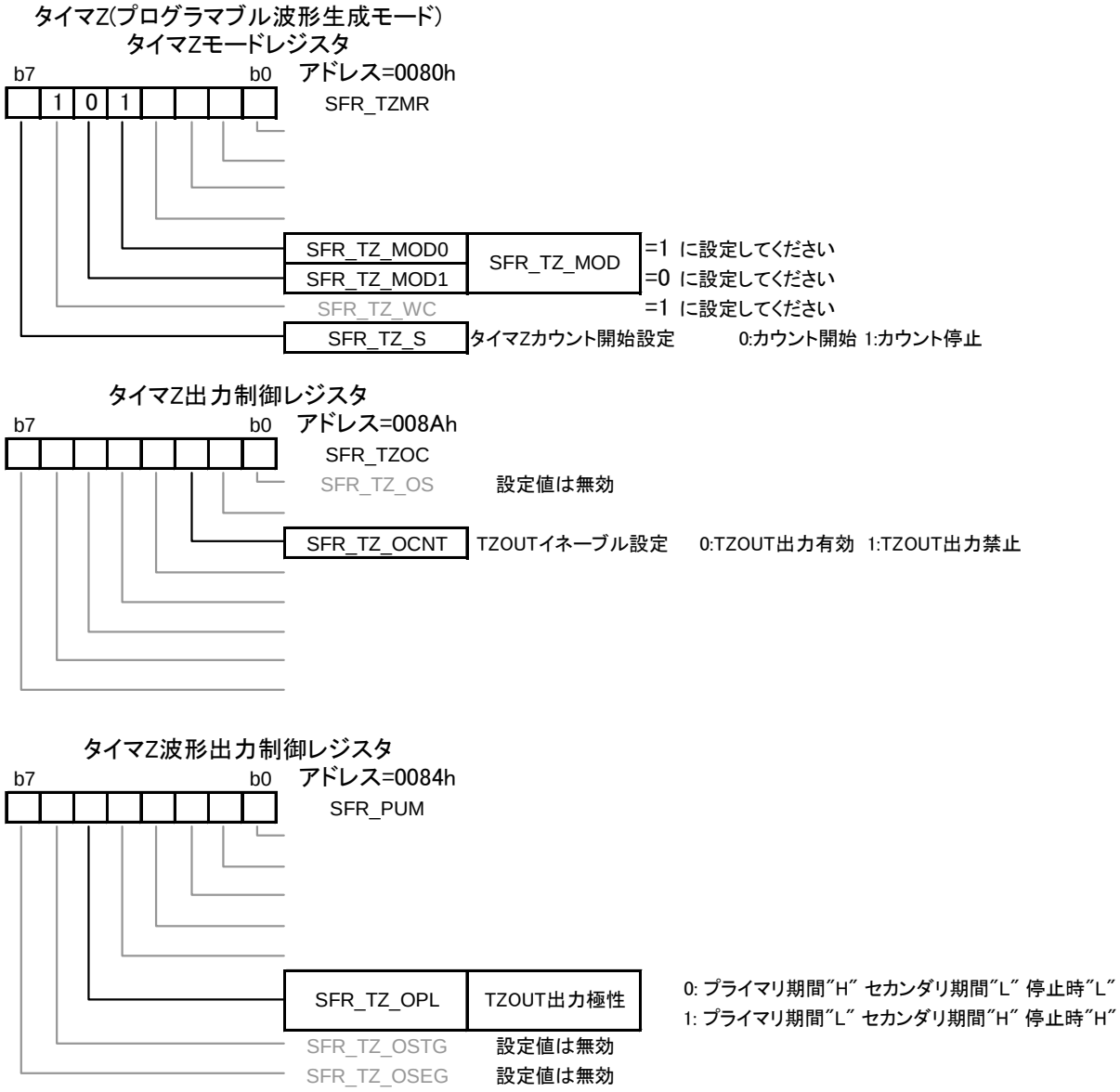
TZOUT 端子は使用されません。プログラマブル入出力ポートの P1_3 端子として使用可能です。

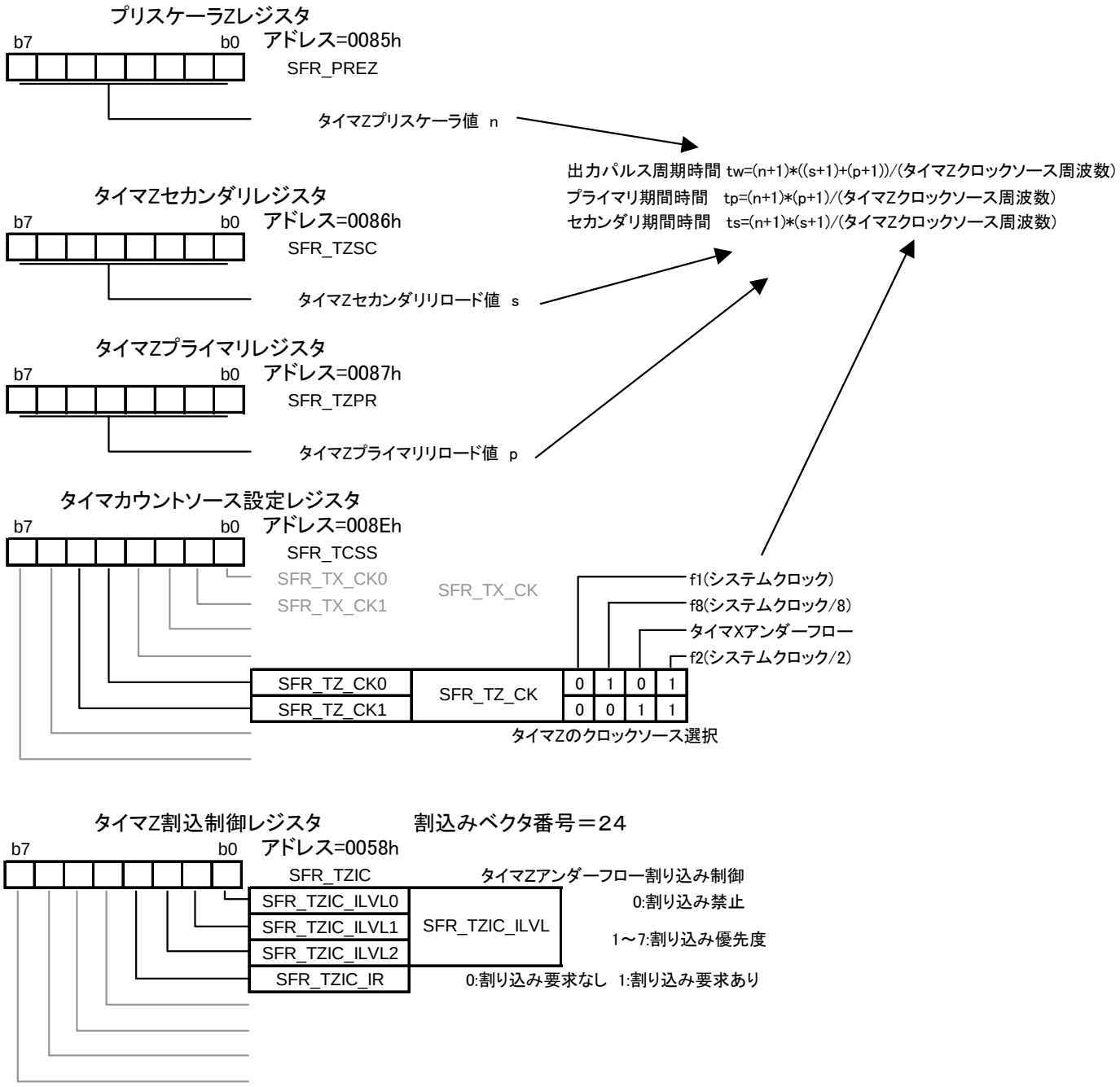
記述例

```
// １m秒間隔でタイマＺアンダーフロー割り込みを発生させる。
// タイマーＺ初期化
void tz_init(void)
{
        SFR_TZMR=0;
        SFR_TZ_CK=1;      // タイマＺクロックソース= f8 (20MHz/8=2.5MHz)
        SFR_PREZ=250-1;   //
        SFR_TZPR=10-1;    // インターバル時間 t=(250*10)/2500000=1mS
        SFR_TZIC=6;       // 割り込みレベル 6
        SFR_TZ_S=1;       // タイマＺカウントスタート
}

// タイマＺアンダーフロー割り込み（1mS インターバル割込み）
void tz_int(void)
#pragma interrupt tz_int (vect=24)
{
        // 割り込み処理を記述する
}

void main(void)
{
        tz_init();        // タイマＺ初期化
        ei();             // 割り込み許可

        // メイン処理を記述

}
```

3-2-2 タイマＺ（プログラマブル波形生成モード）

プログラマブル波形生成モードは指定したタイマＺクロックソースをプリスケーラＺレジスタ値で分周し、そのアンダーフローでタイマＺプライマリレジスタ値（プライマリリロード値）とタイマＺセカンダリレジスタ値（セカンダリリロード値）を交互にリロードして連続してダウンカウントを行います。そのときプライマリ期間とセカンダリ期間で反転したパルスを TZOUT 端子(P1_3 端子)から出力します。パルスの極性は TZOUT 出力極性選択により設定できます。パルスの“Ｈ”幅と“Ｌ”幅を任意に設定できるので PWM（パルス幅変調）制御に向いています。





TZOUT 端子を出力として使用するので、ポートＰ１方向レジスタの P1_3 対応ビットを“０”=入力に設定してください。（必用に応じて）
SFR_TZ_OCNT により TZOUT 出力の有効無効を設定できます。SFR_TZ_OCNT=0 の場合 TZOUT 出力は有効です。SFR_TZ_OCNT=1 の場合、TZOUT 出力はタイマ Z 出力ではなく、P1_3 ポートラッチの値が出力されます。（TZOUT 端子出力を使用するために、P1_3 端子は入力設定(SFR_P1D_3=0)にしているはずですが、この状態の場合は常に端子出力モードとして扱われます。）

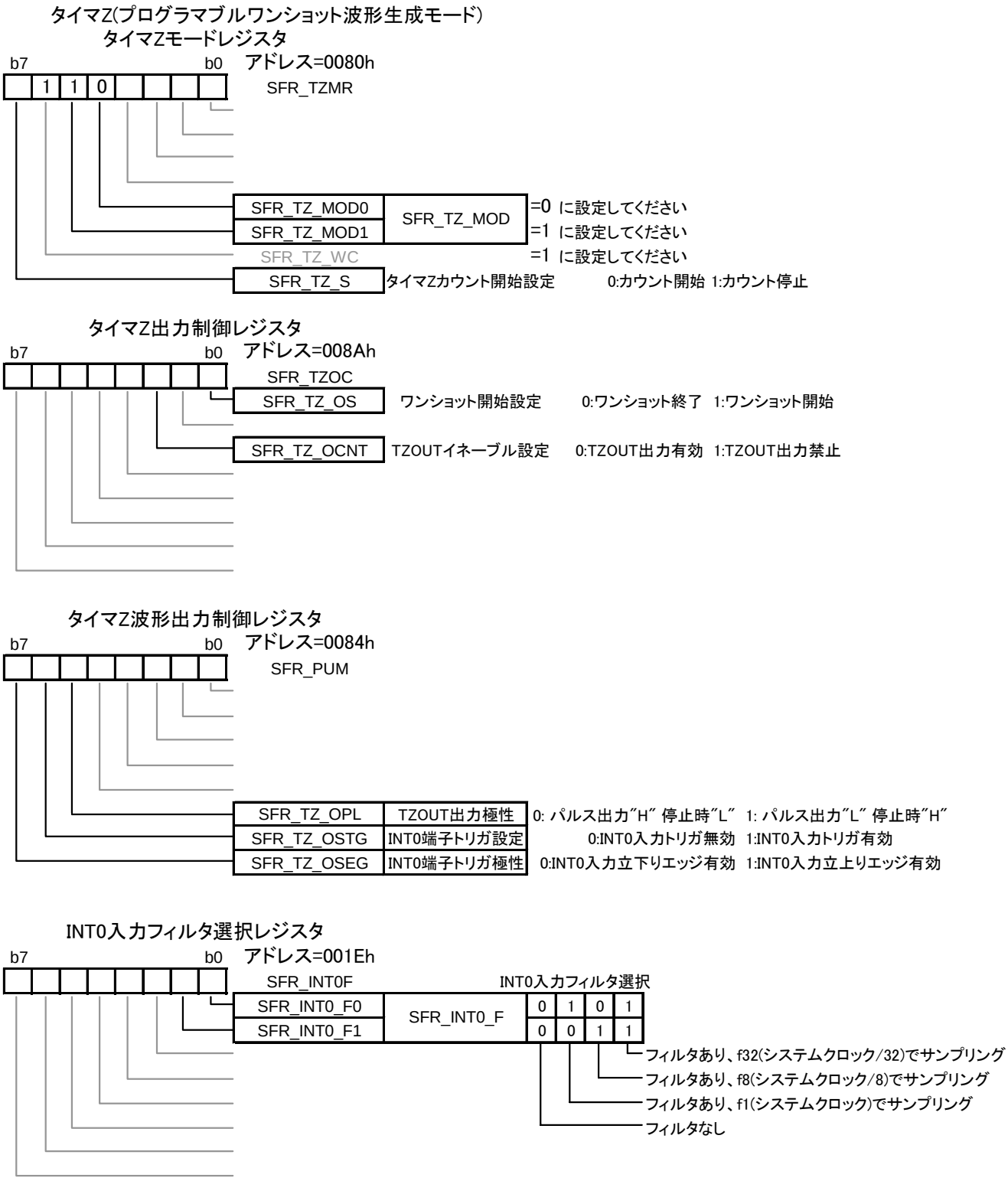
タイマＺセカンダリレジスタ(SFR_TZSC)書込み後に、タイマＺプライマリレジスタ(SFR_TZPR)を書き込んでください。タイマＺセカンダリレジスタ書込み値は、タイマＺプライマリレジスタ書込み後の次のプライマリ期間で値が有効になります。

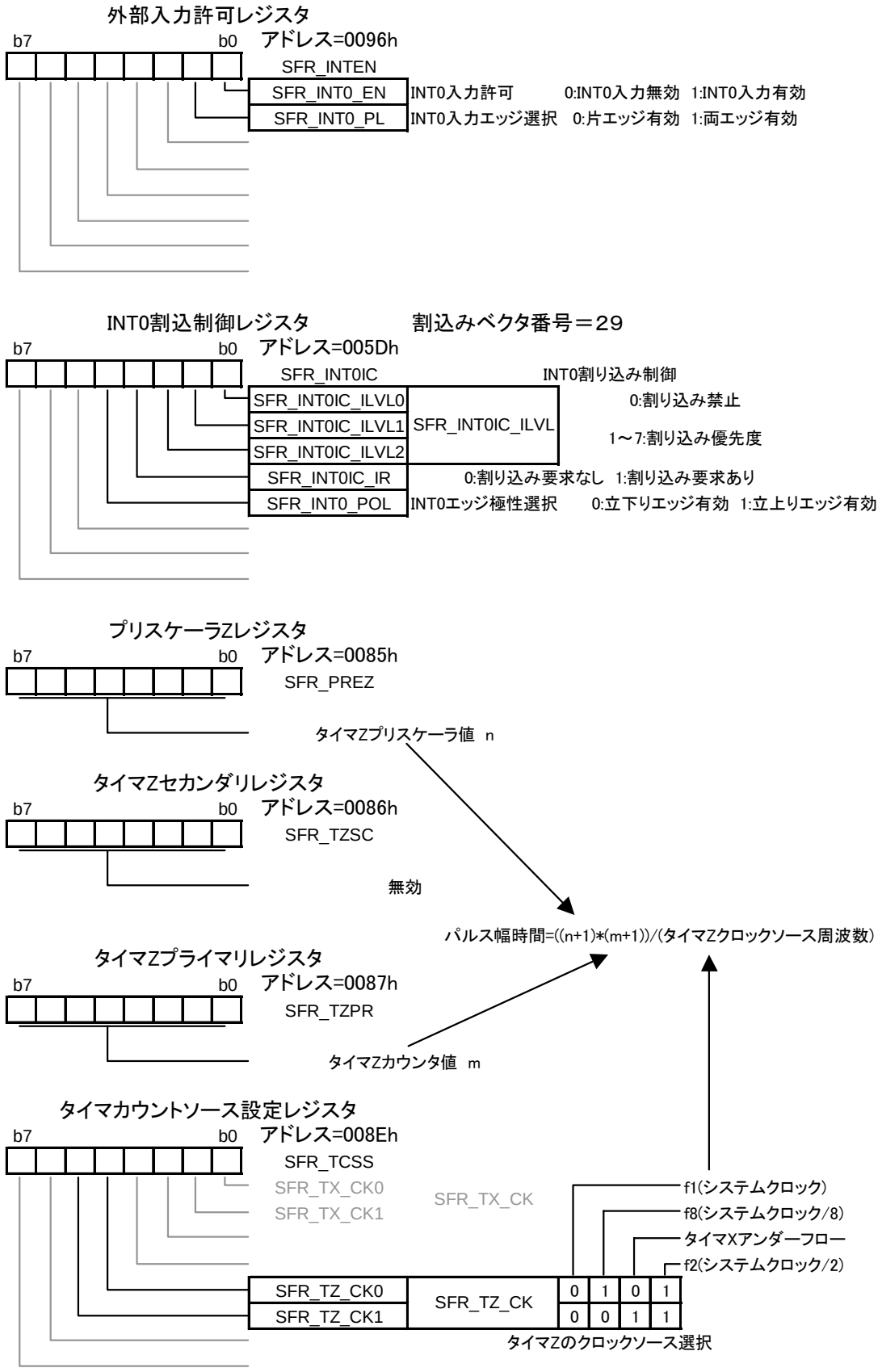
設定例

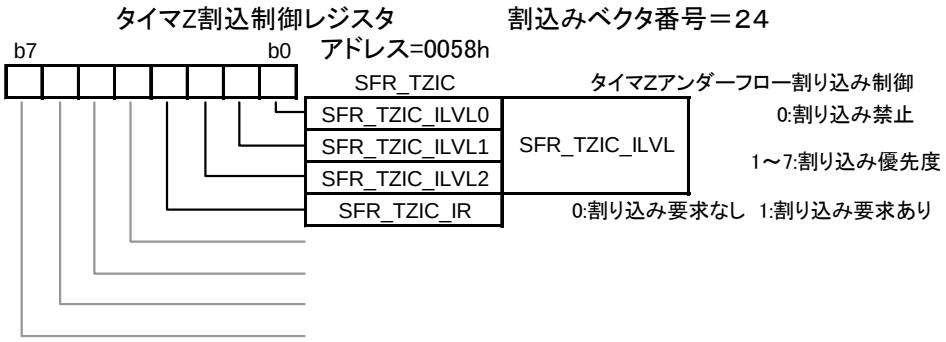
```
// TZOUT 端子(P1_3 端子)にピエゾブザーを取り付け、約 4KHz でブザーを鳴動する
//      SFR_P1D_3=0;            // P1_3 端子を入力設定（この行は省略可能）
        SFR_TZMR=01010000b;     // タイマＺプログラマブル波形生成モード
        SFR_TZ_CK=1;            // タイマＺクロックソース=f8(20MHz/8=2.5MHz)
//      SFR_TZ_OCNT=0;          // TZOUT 端子有効（この行は省略可能）
        SFR_TZ_OPL=0;           // プライマリ“Ｈ”セカンダリ“Ｌ”
        SFR_PREZ=15;
        SFR_TZSC=18;
        SFR_TZPR=19;            // TZOUT 周波数=2.5MHz/((15+1)*(18+19+2))=4006Hz
        SFR_TZ_S=1;             // タイマＺスタート（ブザー鳴動開始）


// TZOUT 端子(P1_3 端子)で PWM 出力制御を行う
// PWM の周期は 1.024mS、分解能は 256 とします。ただし PWM の 0%と 100%は設定できません。
#define PWM_RES 256             // PWM 分解能(設定は最大２５６まで)

unsigned char pwm_value;        // pwm 値(0～254) 0=1/256 254=255/256  0%と 100%は設定できない
        pwm_value=0;            // PWM 初期値
//      SFR_P1D_3=0;            // P1_3 端子を入力設定（この行は省略可能）
        SFR_TZMR=01010000b;     // タイマＺプログラマブル波形生成モード
        SFR_TZ_CK=1;            // タイマＺクロックソース=f8(20MHz/8=2.5MHz)
//      SFR_TZ_OCNT=0;          // TZOUT 端子有効（この行は省略可能）
        SFR_TZ_OPL=1;           // プライマリ“Ｌ”セカンダリ“Ｈ”
        SFR_PREZ=9;             // PWM 周期= ((9+1)*PWM_RES)/2.5MHz=1.024mS
        SFR_TZSC=PWM_RES-2-pwm_value;
        SFR_TZPR=pwm_value;     // PWM 初期値設定(最小値 1/256 pw=4uS)
        SFR_TZ_S=1;             // タイマＺスタート

          …

        pwm_value=127;          // PWM 値は 0 から PWM_RES-2 まで(0-254 の範囲)設定できる。
                                // 255 が設定された場合は正常動作しない。
        SFR_TZSC=PWM_RES-2-pwm_value;
        SFR_TZPR=pwm_value;     // 設定した次の TZPR 値リロードタイミング時に TZSC/TZPR 値が有効
```

3-2-3 タイマＺ（プログラマブルワンショット発生モード）

プログラマブルワンショット発生モードは、プログラムでのトリガまたは、INT0 端子(P4_5 端子)の外部トリガにより、単発の指定時間のパルスを発生します。トリガを検出すると指定したタイマＺクロックソースをプリスケーラＺレジスタ値とタイマＺプライマリレジスタ値でダウンカウントし、アンダーフローすると、タイマＺ割り込みが発生し、ダウンカウントを停止します。また同時に、トリガが入ると（選択クロックソースの立下り）、タイマーＺカウンタのアンダーフローまでの間、パルスを TZOUT 端子(P1_3 端子)から出力します。パルスの極性は TZOUT 出力極性選択により設定できます。




外部入力許可レジスタ
b7
b0
アドレス=0096h
SFR_INTEN
SFR_INT0_EN
INT0入力許可
0:INT0入力無効 1:INT0入力有効
SFR_INT0_PL
INT0入力エッジ選択 0:片エッジ有効 1:両エッジ有効
INT0割込制御レジスタ
割込みベクタ番号＝29
b7
b0
アドレス=005Dh
SFR_INT0IC
INT0割り込み制御
SFR_INT0IC_ILVL0
SFR_INT0IC_ILVL1
SFR_INT0IC_ILVL2
SFR_INT0IC_ILVL
0:割り込み禁止
1～7:割り込み優先度
SFR_INT0IC_IR
0:割り込み要求なし 1:割り込み要求あり
SFR_INT0_POL
INT0エッジ極性選択
0:立下りエッジ有効 1:立上りエッジ有効
プリスケーラZレジスタ
b7
b0
アドレス=0085h
SFR_PREZ
タイマZプリスケーラ値 n
タイマZセカンダリレジスタ
b7
b0
アドレス=0086h
SFR_TZSC
無効
パルス幅時間=((n+1)*(m+1))/(タイマZクロックソース周波数)
タイマZプライマリレジスタ
b7
b0
アドレス=0087h
SFR_TZPR
タイマZカウンタ値 m
タイマカウントソース設定レジスタ
b7
b0
アドレス=008Eh
SFR_TCSS
SFR_TX_CK0
SFR_TX_CK1
SFR_TX_CK
f1(システムクロック)
f8(システムクロック/8)
タイマXアンダーフロー
f2(システムクロック/2)
SFR_TZ_CK0
SFR_TZ_CK1
SFR_TZ_CK
0 1 0 1
0 0 1 1
タイマZのクロックソース選択



TZOUT 端子を出力として使用するので、ポートＰ１方向レジスタの P1_3 対応ビットを“０”=入力に設定してください。(必用に応じて)
SFR_TZ_OCNT により TZOUT 出力の有効無効を設定できます。SFR_TZ_OCNT=0 の場合 TZOUT 出力は有効です。SFR_TZ_OCNT=1 の場合、TZOUT 出力はタイマ Z 出力ではなく、P1_3 ポートラッチの値が出力されます。(TZOUT 端子出力を使用するために、P1_3 端子は入力設定(SFR_P1D_3=0)にしているはずですが、この状態の場合は常に端子出力モードとして扱われます。)

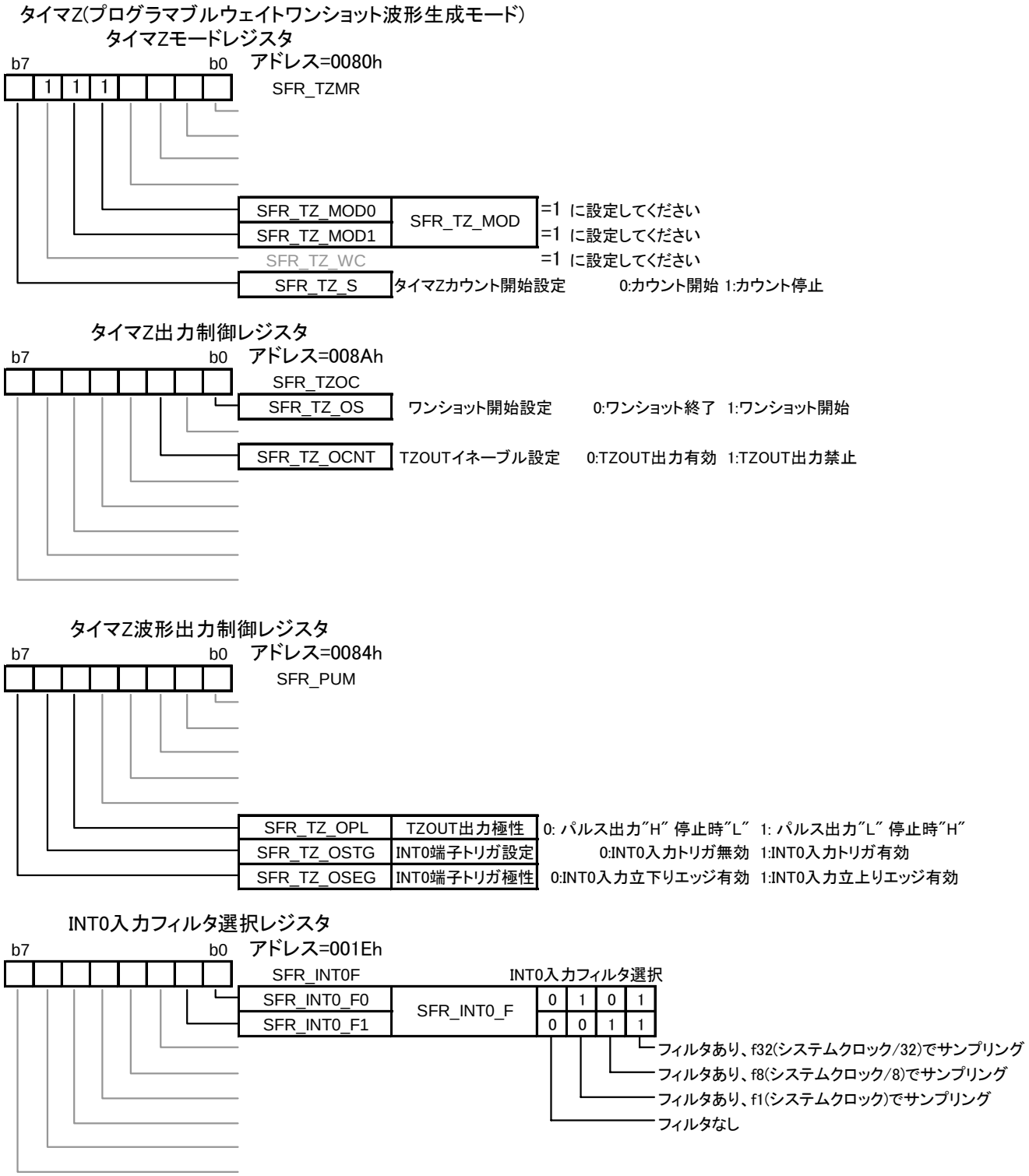
パルス出力カウント中にトリガが入った場合無視されます。リトリガ（延長トリガ）はできません。

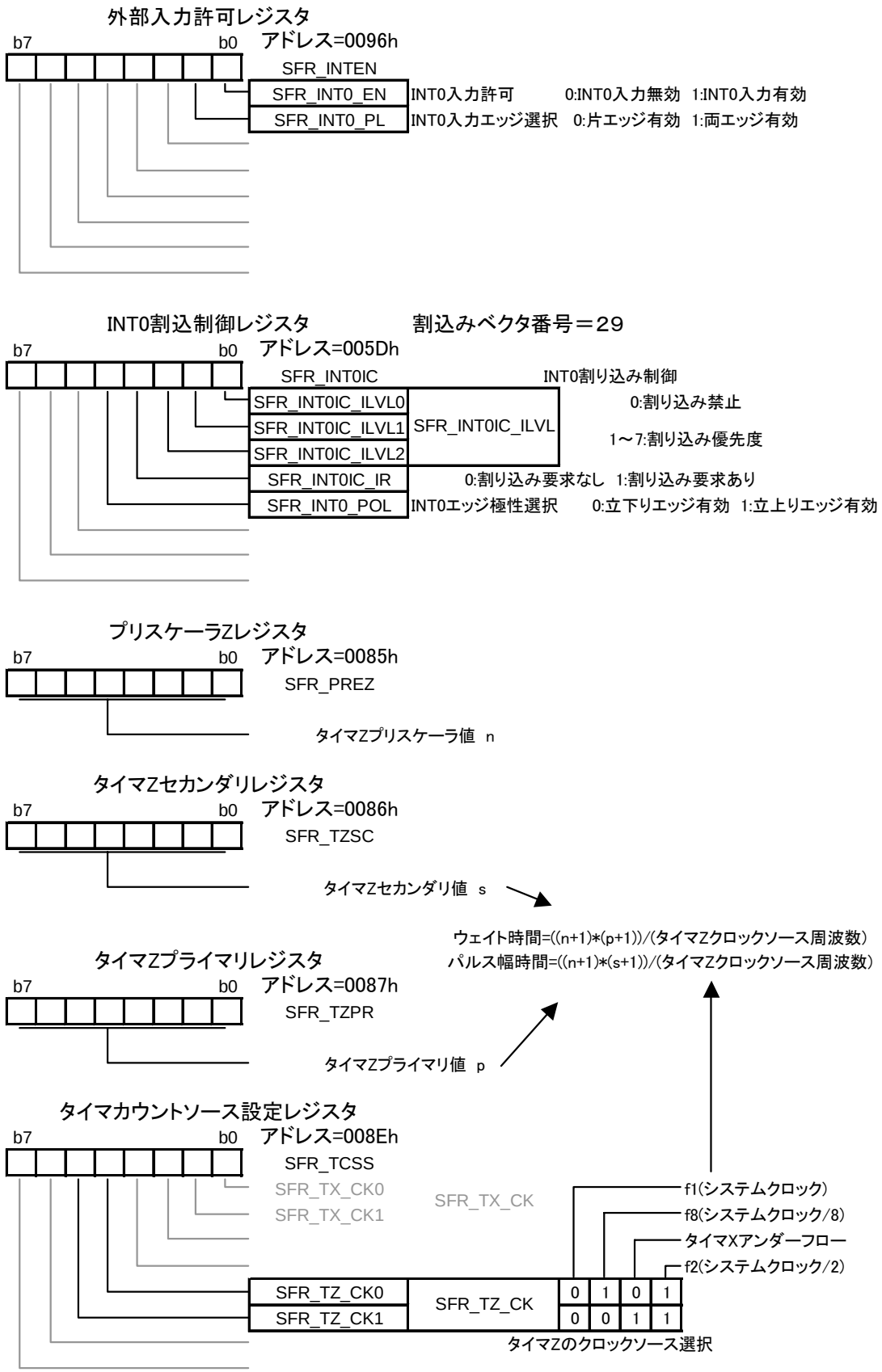
設定例

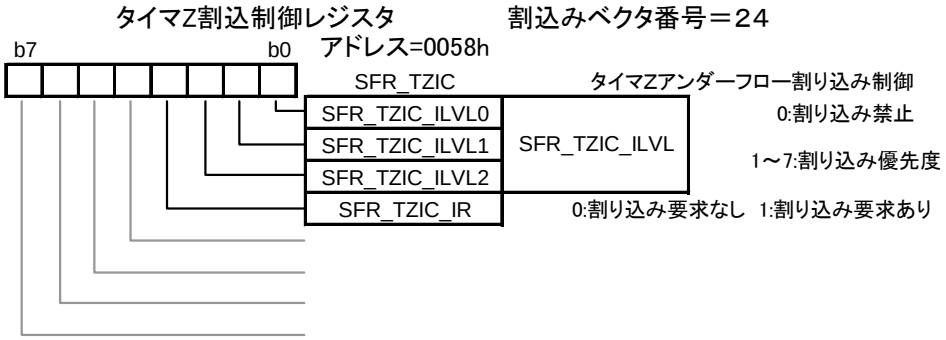
```
// INT0 端子(P4_5 端子)による外部トリガの立下りで、TZOUT 端子(P1_3 端子)に“L”パルスを 500mS 出力する
// （UART1 の RS232C の受信で LED1_3 を一定時間点灯する）
//      SFR_P1D_3=0;            // P1_3 端子を入力設定（この行は省略可能）
        SFR_TXMR=0;             // タイマＸタイマモード
        SFR_TX_CK=1;            // タイマＸクロックソース=f8
        SFR_PREX=0xF9;
        SFR_TX=9;               // タイマＸ間隔=(249+1)*(9+1)/(20MHz/8)=1mS
        SFR_TX_S=1;             // タイマＸスタート

        SFR_TZMR=0x60;          // タイマＺプログラマブルワンショット発生モード
        SFR_TZ_CK=2;            // タイマＺクロックソース=（タイマＸアンダーフロー）
//      SFR_TZ_OCNT=0;          // TZOUT 端子有効（この行は省略可能）
        SFR_TZ_OPL=1;           // TZOUT “L”パルス出力
        SFR_TZ_OSTG=1;          // ワンショット INT0 トリガ有効
//      SFR_TZ_OSEG=0;          // トリガ立下りエッジ有効（この行は省略可能）
        SFR_INT0_EN=1;          // INT0 入力有効
//      SFR_INT0_F=0;           // INT0 入力デジタルフィルタなし(この行は省略可能)
        SFR_PREZ=0x63;
        SFR_TZPR=5-1;           // パルス幅=((99+1)*(5))*1mS=500mS
        SFR_TZ_S=1;             // タイマＺスタート
// 以降 UART1 側のデータ受信があると（TXD の立下り検出で）TZOUT 端子に 500mS の負論理パルスを発生する
```

3-2-4 タイマＺ（プログラマブルウェイトワンショット発生モード）

プログラマブルウェイトワンショット発生モードは、プログラムでのトリガまたは、INT0(P4_5 端子)端子の外部トリガにより､プライマリレジスタで指定時間待った後､単発のセカンダリレジスタ指定時間のパルスを発生します。トリガを検出すると指定したタイマＺクロックソースをプリスケーラＺレジスタ値とタイマＺプライマリレジスタ値でダウンカウントし、アンダーフローした後、さらに、指定したタイマＺクロックソースをプリスケーラＺレジスタ値とタイマＺセカンダリレジスタ値でダウンカウントし、アンダーフローするとタイマＺ割り込みが発生し、ダウンカウントを停止します。そのとき、セカンダリカウント期間に、パルスを TZOUT 端子(P1_3 端子)から出力します。パルスの極性は TZOUT 出力極性選択により設定できます。




外部入力許可レジスタ
b7
b0
アドレス=0096h
SFR_INTEN
SFR_INT0_EN
INT0入力許可
0:INT0入力無効 1:INT0入力有効
SFR_INT0_PL
INT0入力エッジ選択 0:片エッジ有効 1:両エッジ有効
INT0割込制御レジスタ
割込みベクタ番号＝29
b7
b0
アドレス=005Dh
SFR_INT0IC
INT0割り込み制御
SFR_INT0IC_ILVL0
SFR_INT0IC_ILVL1
SFR_INT0IC_ILVL2
SFR_INT0IC_ILVL
0:割り込み禁止
1～7:割り込み優先度
SFR_INT0IC_IR
0:割り込み要求なし 1:割り込み要求あり
SFR_INT0_POL
INT0エッジ極性選択
0:立下りエッジ有効 1:立上りエッジ有効
プリスケーラZレジスタ
b7
b0
アドレス=0085h
SFR_PREZ
タイマZプリスケーラ値 n
タイマZセカンダリレジスタ
b7
b0
アドレス=0086h
SFR_TZSC
タイマZセカンダリ値 s
ウェイト時間=((n+1)*(p+1))/(タイマZクロックソース周波数)
パルス幅時間=((n+1)*(s+1))/(タイマZクロックソース周波数)
タイマZプライマリレジスタ
b7
b0
アドレス=0087h
SFR_TZPR
タイマZプライマリ値 p
タイマカウントソース設定レジスタ
b7
b0
アドレス=008Eh
SFR_TCSS
SFR_TX_CK0
SFR_TX_CK1
SFR_TX_CK
f1(システムクロック)
f8(システムクロック/8)
タイマXアンダーフロー
f2(システムクロック/2)
SFR_TZ_CK0
SFR_TZ_CK1
SFR_TZ_CK
0 1 0 1
0 0 1 1
タイマZのクロックソース選択



TZOUT 端子を出力として使用するので、ポートＰ１方向レジスタの P1_3 対応ビットを“ 0 ”=入力に設定してください。(必用に応じて)
SFR_TZ_OCNT により TZOUT 出力の有効無効を設定できます。SFR_TZ_OCNT=0 の場合 TZOUT 出力は有効です。SFR_TZ_OCNT=1 の場合、TZOUT 出力はタイマ Z 出力ではなく、P1_3 ポートラッチの値が出力されます。(TZOUT 端子出力を使用するために、P1_3 端子は入力設定(SFR_P1D_3=0)にしているはずですが、この状態の場合は常に端子出力モードとして扱われます。)

タイマＺセカンダリレジスタ(SFR_TZSC)書込み後に、タイマＺプライマリレジスタ(SFR_TZPR)を書き込んでください。タイマＺセカンダリレジスタ書込み値は、タイマＺプライマリレジスタ書込み後の次のプライマリ期間で値が有効になります。
ウェイトカウント中またはパルス出力カウント中にトリガが入った場合無視されます。リトリガ（延長トリガ）はできません。

設定例

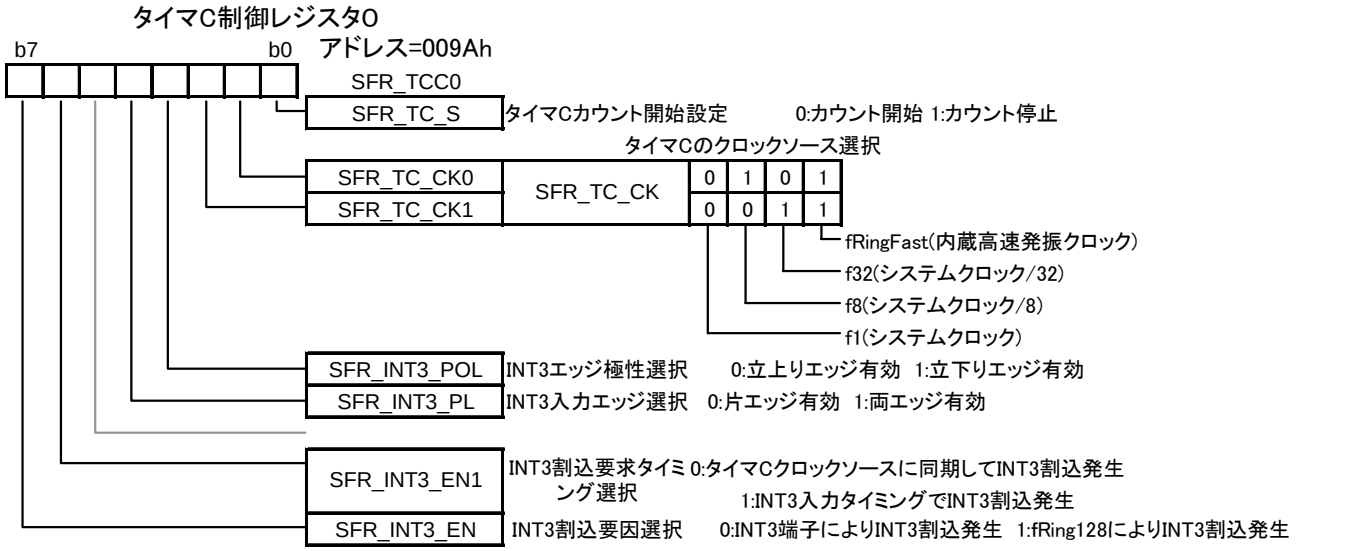
```
//  プログラムによるトリガで 3 秒待ち、1 秒の負論理パルスを出力する。
//      SFR_P1D_3=0;            // P1_3 端子を入力設定（この行は省略可能）
        SFR_TXMR=0;             // タイマＸタイマモード
        SFR_TX_CK=1;            // タイマＸクロックソース=f8
        SFR_PREX=0xF9;
        SFR_TX=9;               // タイマＸ間隔=(249+1)*(9+1)/(20MHz/8)=1mS
        SFR_TX_S=1;             // タイマＸスタート

        SFR_TZMR=0x70;          // タイマＺプログラマブルウェイトワンショット発生モード
        SFR_TZ_CK=2;            // タイマＺクロックソース=（タイマＸアンダーフロー）
//      SFR_TZ_OCNT=0;          // TZOUT 端子有効（この行は省略可能）
        SFR_TZ_OPL=1;           // TZOUT “Ｌ”パルス出力
//      SFR_INT0_EN=0;          // INT0 入力無効（この行は省略可能）
        SFR_PREZ=0x63;
        SFR_TZSC=9;             // パルス幅=((99+1)*(9+1))*1mS=1S
        SFR_TZPR=0x1D;          // 待ち時間=((99+1)*(29+1))*1mS=3S
        SFR_TZ_S=1;             // タイマＺスタート

        SFR_TZ_OS=1;            // トリガ（ここから３秒待ってから 1 秒のパルスを出力）
        while(SFR_TZ_OS!=0)     // ウェイトワンショット完了？
                ;
```

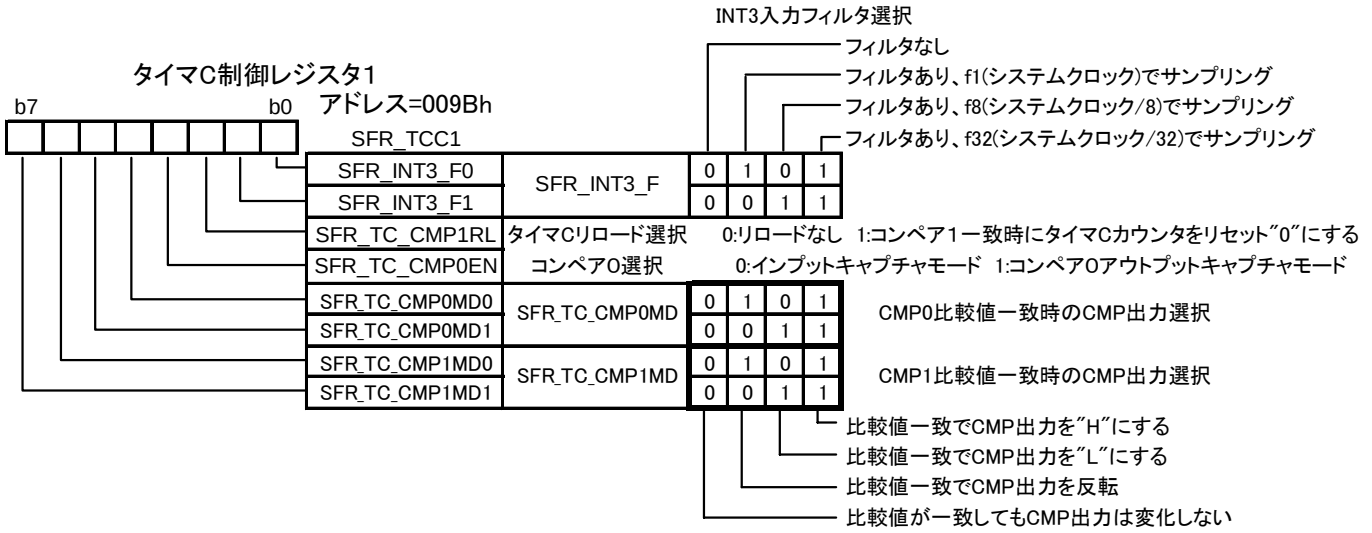
3-3 タイマC

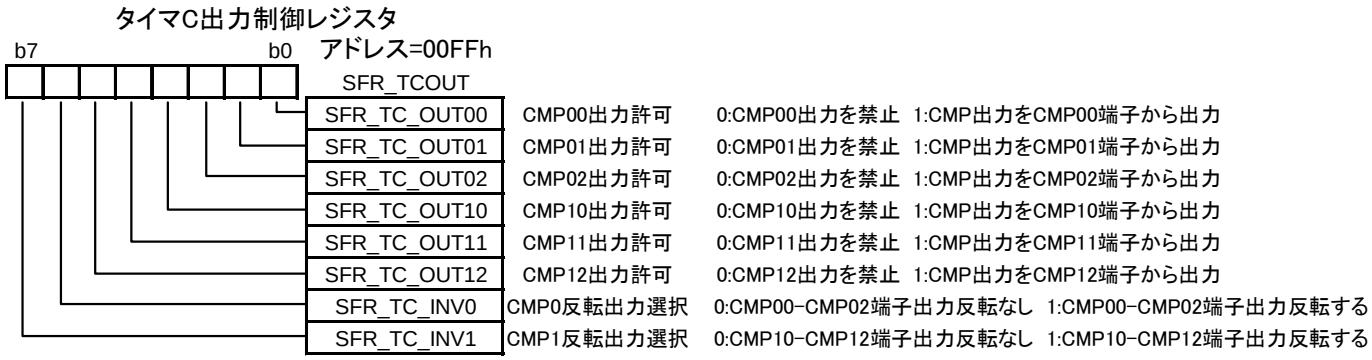
タイマCは動作モードとして、インプットキャプチャモード、アウトプットコンペアモードがあります。

- **インプットキャプチャモード**
- **アウトプットコンペアモード**

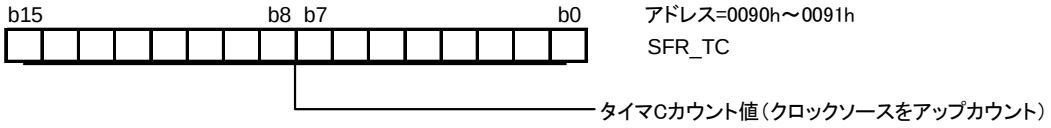
レジスタによりモードを設定し、また、モードにより設定するレジスタ値が異なります。



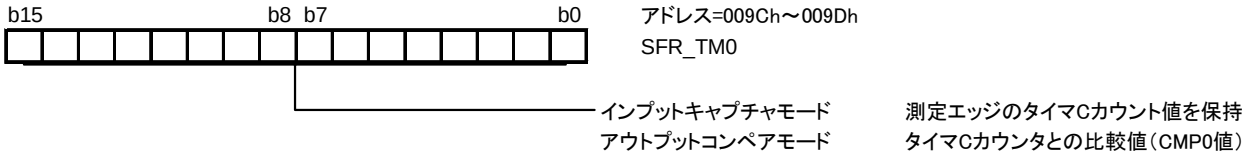




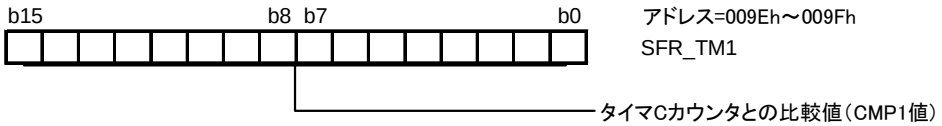
タイマCレジスタ



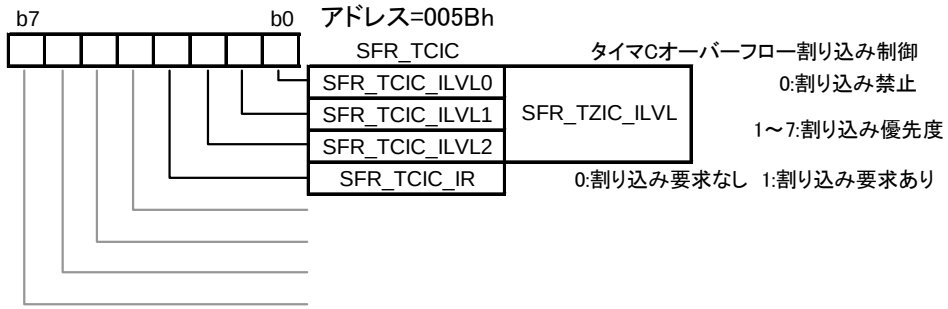
キャプチャ・コンペア0レジスタ



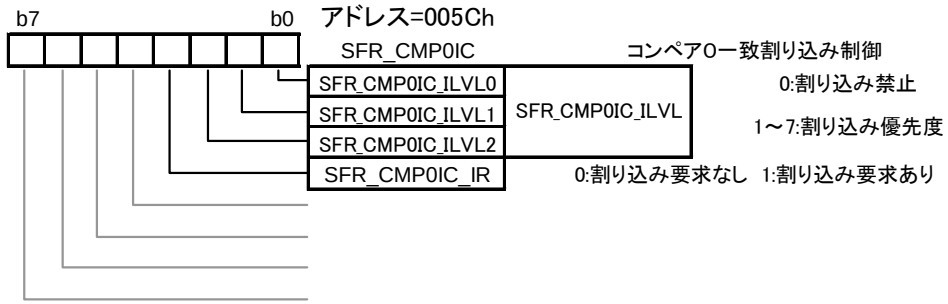
コンペア1レジスタ



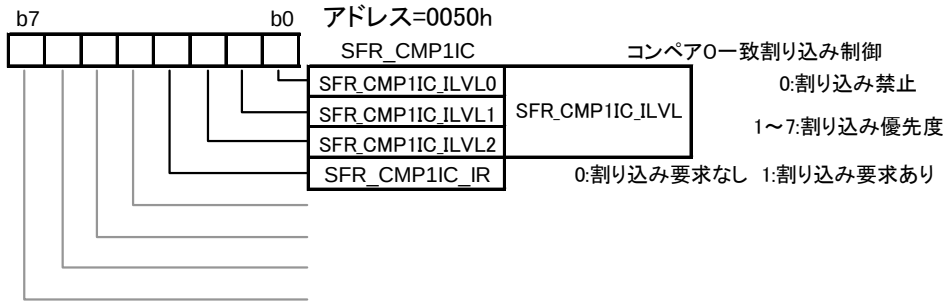
タイマC割込制御レジスタ　　割込みベクタ番号＝27



コンペア0割込制御レジスタ　　割込みベクタ番号＝28

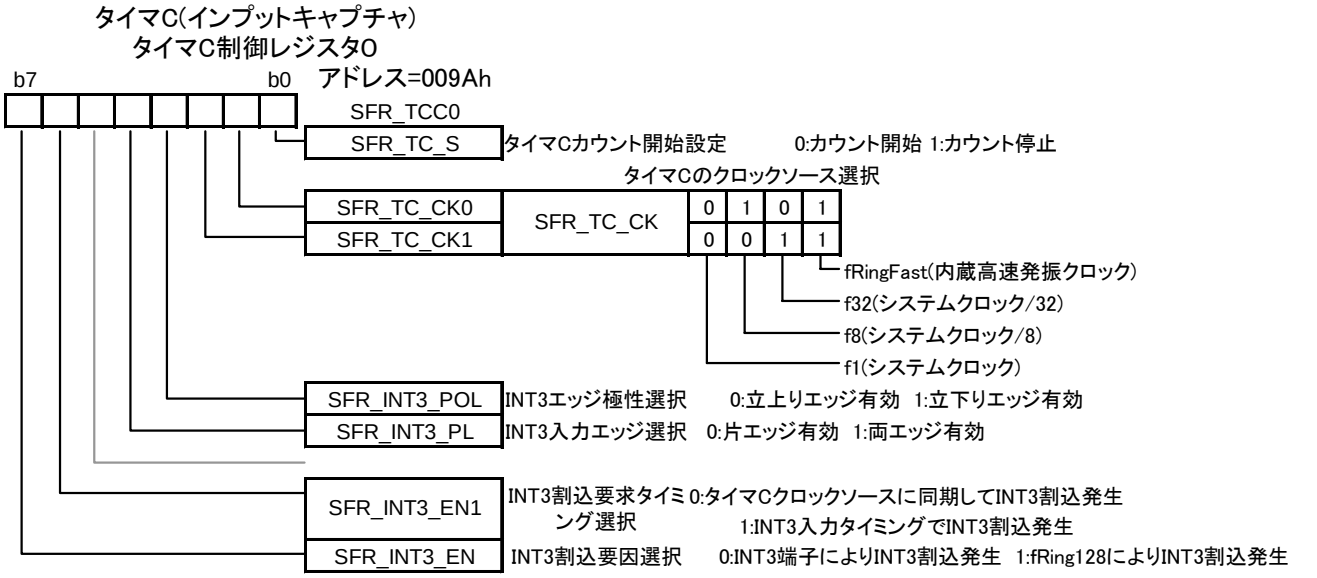


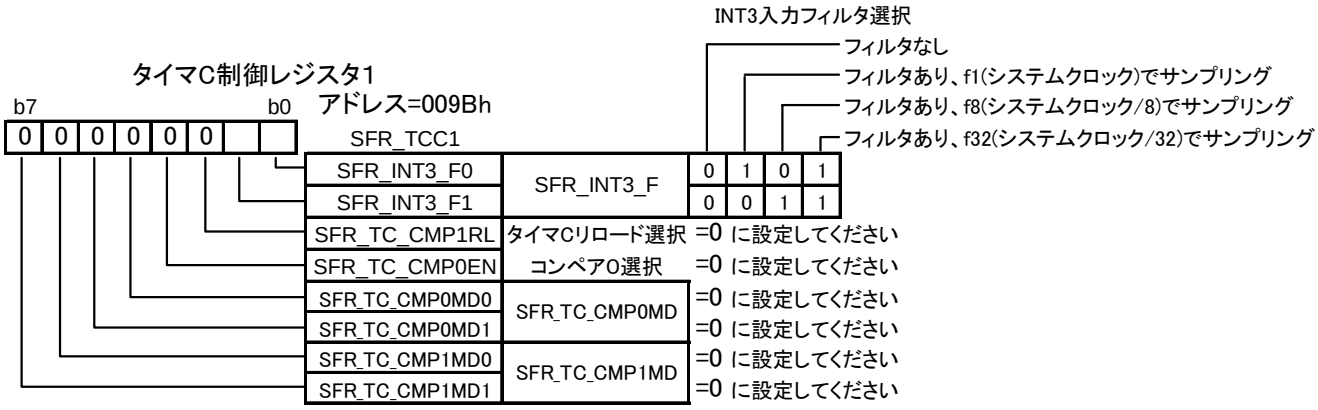
コンペア1割込制御レジスタ　　割込みベクタ番号＝16

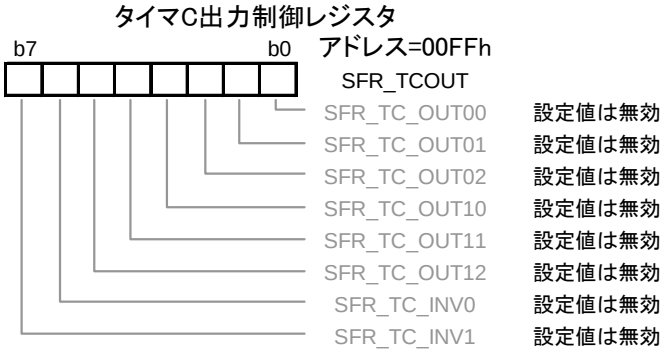


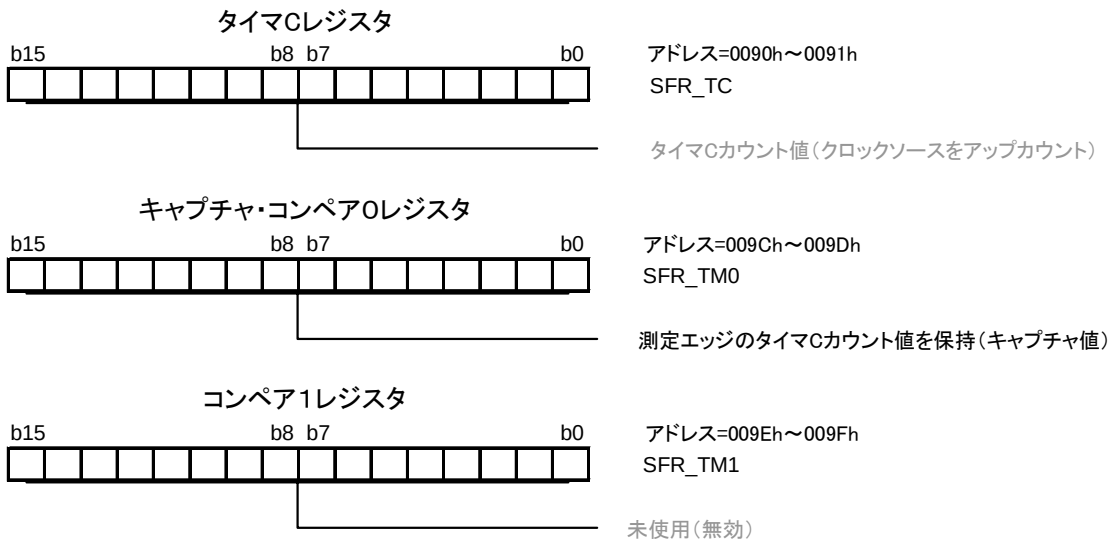
3-3-1 タイマＣ（インプットキャプチャモード）

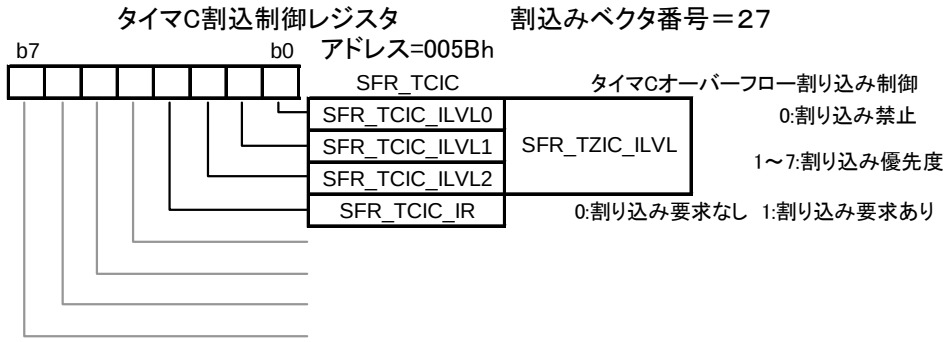
インプットキャプチャーモードは、16 ビットベースタイマ（選択したクロックソースをカウントアップする）であるタイマＣカウンタ値を INT3 端子(P3_3 端子)外部入力トリガエッジまたは fRing128（内部オシレータ）によりキャプチャー（その時点のタイマＣカウント値を保持）します。外部入力トリガ検出と、タイマＣオーバーフローで割り込みを発生します。
外部パルスの間隔時間や幅時間を測定するのに使用します。











タイマＣカウント停止を行うと(SFR_TC_S=0)タイマＣカウント値は“０”にリセットされます。
タイマＣカウント開始(SFR_TC_S=1)でタイマＣカウンタは“０”から選択したクロックソースでカウントアップされます。カウントアップされた、タイマＣカウンタは 16 ビット(0FFFFh)をオーバーするとタイマＣオーバーフロー割り込みを発生して、カウント値を“０”にし、カウントを継続します。

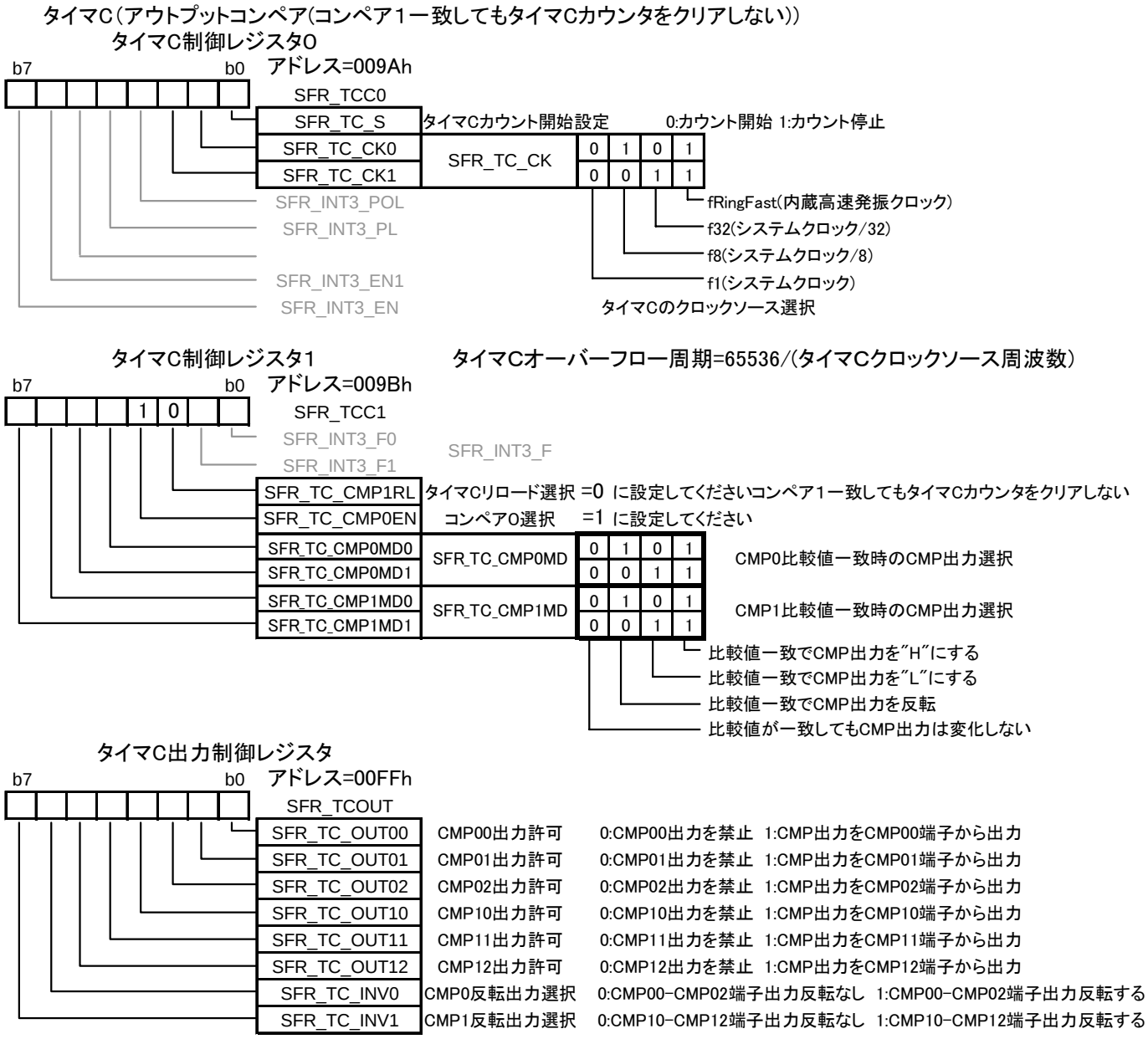
CMP00～CMP12 端子は使用しません。任意にプログラマブル入出力ポート(P1_0/P1_1/P1_2/P3_3/P3_4/P3_4)端子として使用できます。

記述例

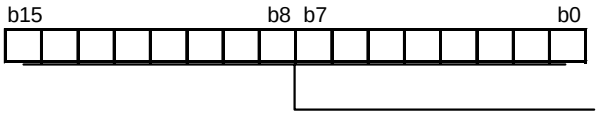
```
// INT3 端子の"L"パルス幅を 400nS 単位で最大 26mS までとして計測する。
// タイマＣインプットキャプチャモード初期化
void tc_init(void)
{
//      SFR_P3D_3=0;            // P3_3(INT3-)端子入力設定（この行は省略可能）
        SFR_TCC0=0b01001000;    // タイマＣインプットキャプチャ INT3 立下りエッジ有効
        SFR_TC_CK=1;            // タイマＣクロックソース=f8 (20MHz/8=2.5MHz)
//      SFR_INT3_F=0;           // INT3 デジタルフィルタなし（この行は省略可能）
        SFR_INT3IC=6;           // 割込みレベル６
        SFR_TCIC=6;             // 割込みレベル６
        SFR_TC_S=1;             // タイマＣカウント開始
}

// グローバル変数宣言
unsigned int pulse_width;       // パルス"L"幅測定値(*400nS) max:26.214mS 計測不能時:0xffff
unsigned int ov_count;          // オーバーフローカウント
unsigned int count_value;
unsigned int prev_count_value;

// タイマＣオーバーフロー割込み
void tc_int(void)
#pragma interrupt tc_int (vect=27)
{
        if (ov_count!=0xffff)   // ov_count が 0xffff ならそれ以上カウントしない
                ov_count++;     // オーバーフロー(65536/2.5MHz=26.2144mS)をカウントする
}

// キャプチャー入力エッジ検出(INT3)割込み
void int3_int(void)
#pragma interrupt int3_int (vect=26)
{
        count_value=SFR_TM0;    // カウント値読み込み
        if (count_value>prev_count_value)
                pulse_width=(ov_count==0)?(count_value-prev_count_value):0xffff;
        else
                pulse_width=(ov_count<=1)?(count_value-prev_count_value):0xffff;
        ov_count=0;     // ov_count クリア
        prev_count_value=count_value;   // 現 TM0 値保存
}

void main(void)
{
        prev_count_value=0;     //
        pulse_width=0xffff;     // 計測値初期化（インバリッド設定）
        tc_init();
        ei();

        // メイン処理を記述

}
```

3-3-2 タイマＣ（アウトプットコンペアモード）

アウトプットコンペアモードでは、16 ビットベースタイマ（選択したクロックソースをカウントアップする）であるタイマＣカウンタ値をコンペア 0 値とコンペア 1 値の 2 つの比較値と比較して一致すると各コンペア０一致割込み、コンペア１一致割込みを発生すると同時に、指定された一致時の各ＣＭＰ出力選択で、ＣＭＰ出力を変化させることができます。ＣＭＰ出力は CMP00 端子～CMP12 端子の切り替えができます。
タイマＣカウンタがオーバーフローするとタイマＣオーバーフロー割り込みを発生します。
アウトキャプチャモードは位相を変化させるパルス生成やパルス幅変調（ＰＷＭ）出力などに使用できます。
コンペア１レジスタ値一致時にタイマＣカウンタを“０”に(クリア)することができます。
コンペア１一致時にタイマＣカウンタをクリアすることで、タイマＣカウンタの１周期をコンペア１レジスタ値で任意に設定できます。コンペア１一致時にタイマＣカウンタをクリアしない場合は、タイマＣカウンタの１周期時間は 65536/(クロックソース周波数)固定です。



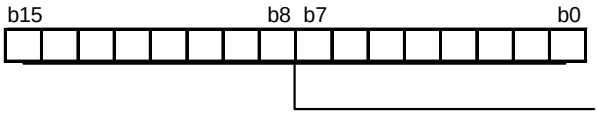
タイマCレジスタ



アドレス=0090h～0091h
SFR_TC

タイマCカウント値（クロックソースをアップカウント）

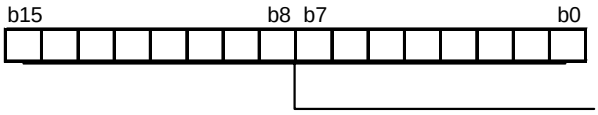
キャプチャ・コンペア0レジスタ



アドレス=009Ch～009Dh
SFR_TM0

タイマC比較値0（コンペア0値）

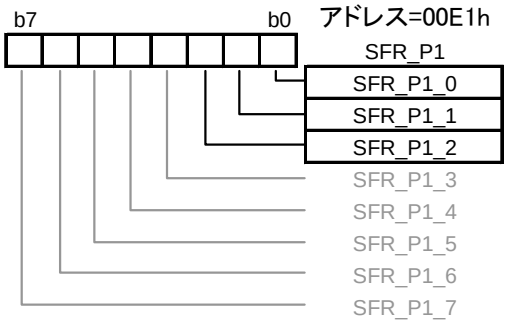
コンペア1レジスタ



アドレス=009Eh～009Fh
SFR_TM1

タイマC比較値1（コンペア1値）
タイマCカウンタクリア値

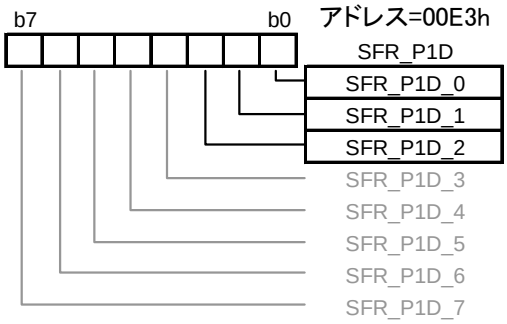
ポートP1レジスタ



アドレス=00E1h

CMP00端子出力　SFR_P1_0=1かつSFR_TC_OUT00=1かつCMP出力=1ならCMP00端子="H"
CMP01端子出力　SFR_P1_1=1かつSFR_TC_OUT01=1かつCMP出力=1ならCMP01端子="H"
CMP02端子出力　SFR_P1_2=1かつSFR_TC_OUT02=1かつCMP出力=1ならCMP02端子="H"
ただしSFR_TC_INV0=1ならば上記出力は反転される

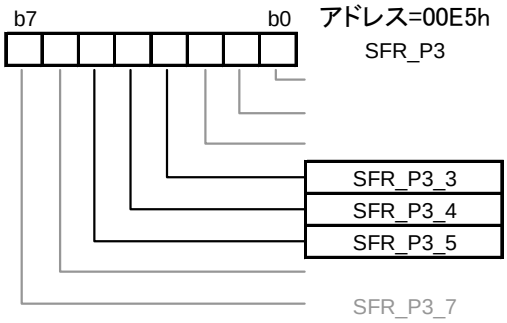
ポートP1方向レジスタ



アドレス=00E3h

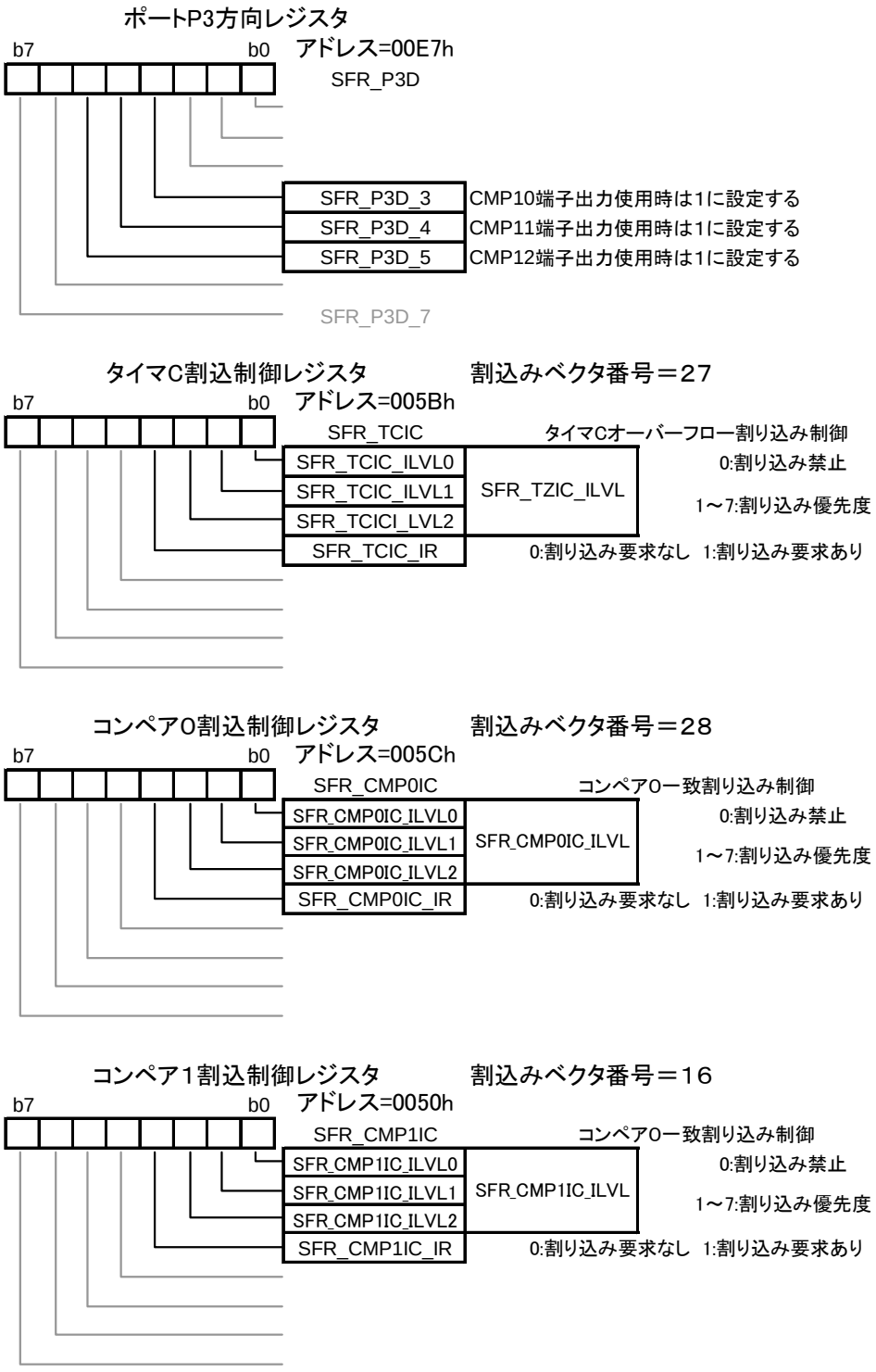
CMP00端子出力使用時は1に設定する
CMP01端子出力使用時は1に設定する
CMP02端子出力使用時は1に設定する

ポートP3レジスタ



アドレス=00E5h

CMP10端子出力　SFR_P3_3=1かつSFR_TC_OUT10=1かつCMP出力=1ならCMP10端子="H"
CMP11端子出力　SFR_P3_4=1かつSFR_TC_OUT11=1かつCMP出力=1ならCMP11端子="H"
CMP12端子出力　SFR_P3_5=1かつSFR_TC_OUT12=1かつCMP出力=1ならCMP12端子="H"
ただしSFR_TC_INV1=1ならば上記出力は反転される



アウトプットコンペアリロードなしモード(SFR_TC_CMP1RL=0)の時、コンペア１レジスタ TM1 の値がコンペア０レジスタ TM0 の値より小さい場合コンペア０の出力設定動作を行わないようです、コンペア１を使用しない場合は、値を 0xFFFF にして（リセットでの初期値は 0xFFFF です）使用してください。

コンペア１レジスタ(TM0)値はリセットでの初期値は“０”ですが、SFR_TC_CMP0EN ビットを“１”にすると“0xFFFF”になります。SFR_TC_CMP0EN ビットを“１”にした後で、コンペア１レジスタ(TM0)値を設定してください。

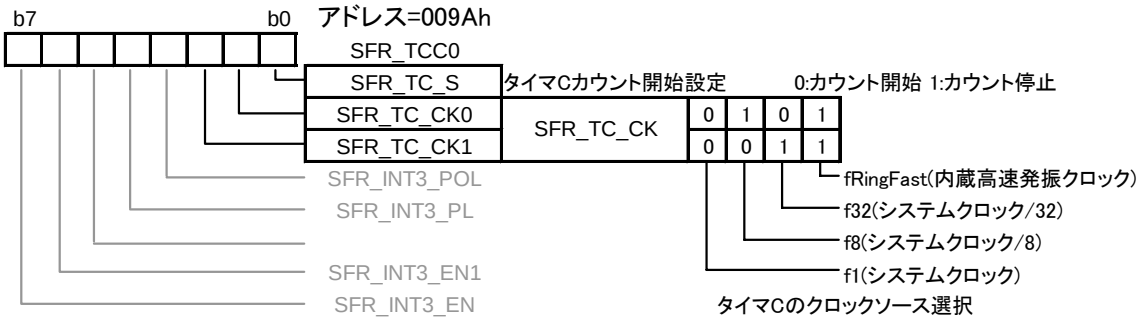
設定例

// タイマＣアウトプットコンペアモードで 1 周期(104.8576mS)の中で TM0 設定値一致で CMP01 出力を"L"にして
// TM1 設定値一致で CMP01 出力を"H"にする。周期固定で任意パルスの幅、位相を TM0/TM1 で変化させせます。

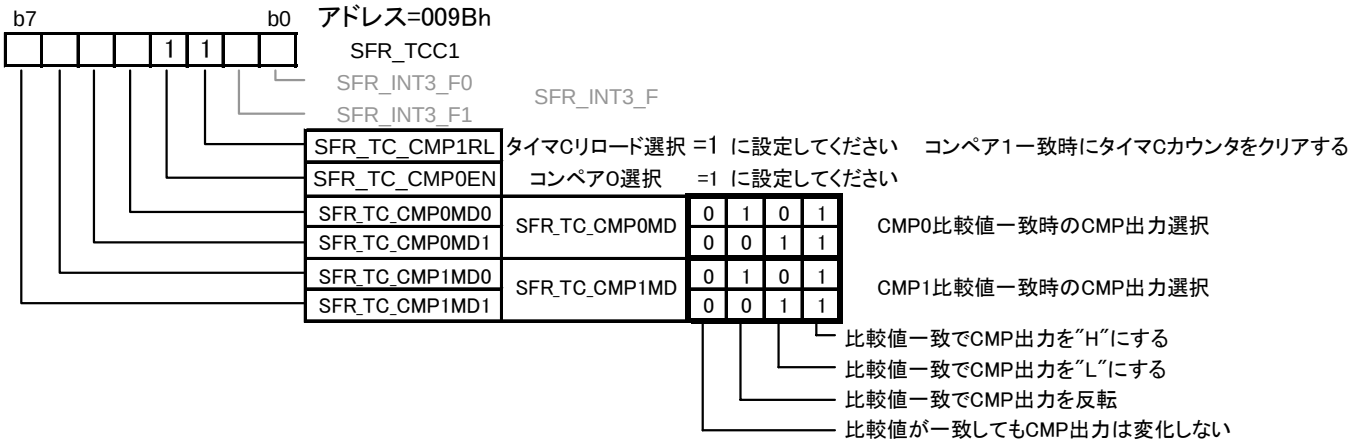
```
SFR_P1_1=1;                 // P1_1(CMP01)端子出力"H"設定
SFR_P1D_1=1;                // P1_1(CMP01)端子出力設定
SFR_TCC0=0b00000000;        // タイマＣアウトプットコンペア
SFR_TC_CK=2;                // タイマＣクロックソース=f32 (20MHz/32=625KHz)
SFR_TC_CMP1RL=0; // CMP1 一致リロードなし
SFR_TC_CMP0EN=1; // アウトプットコンペア０有効
SFR_TC_CMP0MD=3; // CMP0 一致で CMP 出力"H"
SFR_TC_CMP1MD=2; // CMP1 一致で CMP 出力"L"
SFR_TC_OUT01=1;             // CMP01 出力有効
SFR_TC_INV0=1;              // CMP0x 出力反転

SFR_TM0=0x1000;
SFR_TM1=0xC000;             // 必ず TM0<TM1 であること

SFR_TC_S=1;                 // タイマＣカウント開始

SFR_TM0=0x8000;             // 必ず TM0<TM1 であること
SFR_TM1=0x9000;             // TM0/TM1 を変更するとパルス位相と幅が変化する。
```

タイマC（アウトプットコンペア(コンペア1一致でタイマCカウンタをクリアする)）

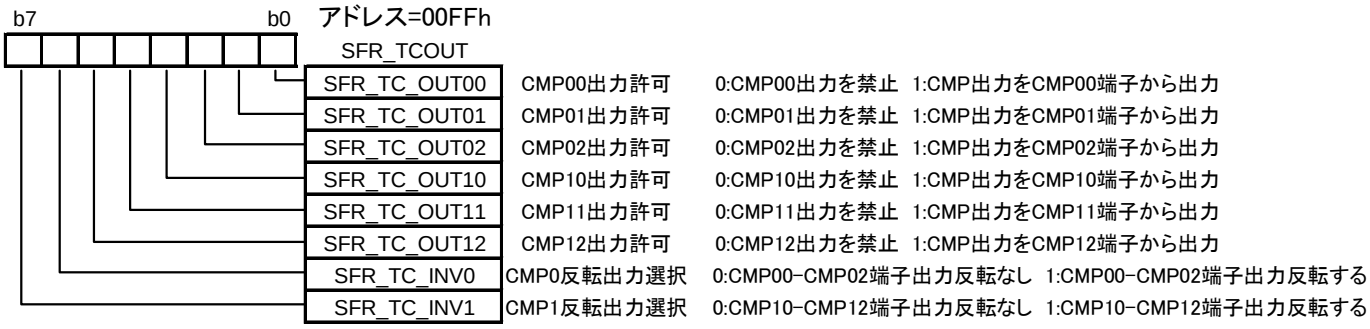
タイマC制御レジスタ0



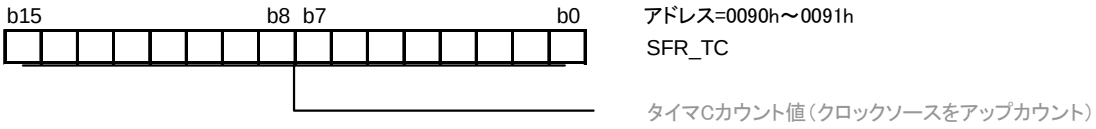
タイマC制御レジスタ1　　タイマC周期=(コンペア1レジスタ値−1)/(タイマCクロックソース周波数)



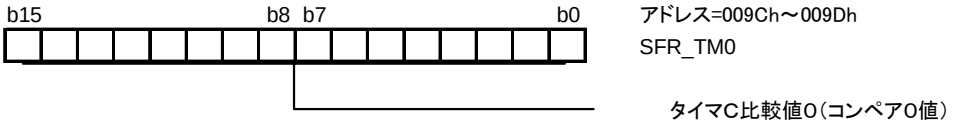
タイマC出力制御レジスタ



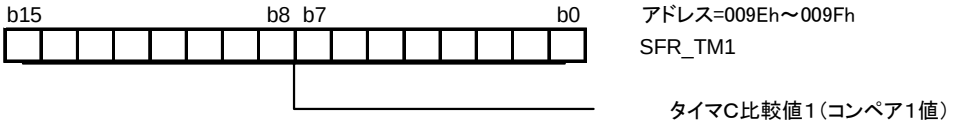
タイマCレジスタ



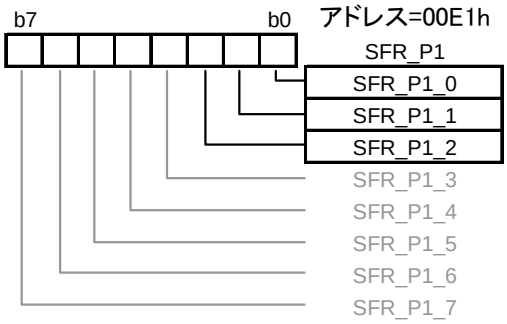
キャプチャ・コンペア0レジスタ



コンペア1レジスタ

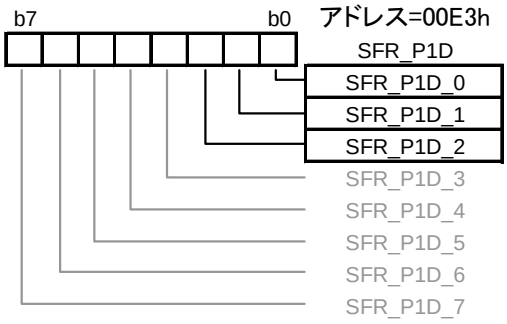


ポートP1レジスタ



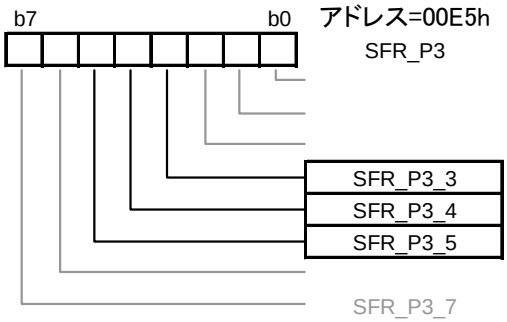
CMP00端子出力　SFR_P1_0=1かつSFR_TC_OUT00=1かつCMP出力=1ならCMP00端子="H"
CMP01端子出力　SFR_P1_1=1かつSFR_TC_OUT01=1かつCMP出力=1ならCMP01端子="H"
CMP02端子出力　SFR_P1_2=1かつSFR_TC_OUT02=1かつCMP出力=1ならCMP02端子="H"
ただしSFR_TC_INV0=1ならば上記出力は反転される

ポートP1方向レジスタ



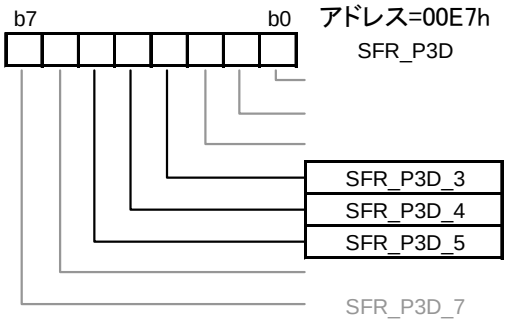
CMP00端子出力使用時は1に設定する
CMP01端子出力使用時は1に設定する
CMP02端子出力使用時は1に設定する

ポートP3レジスタ

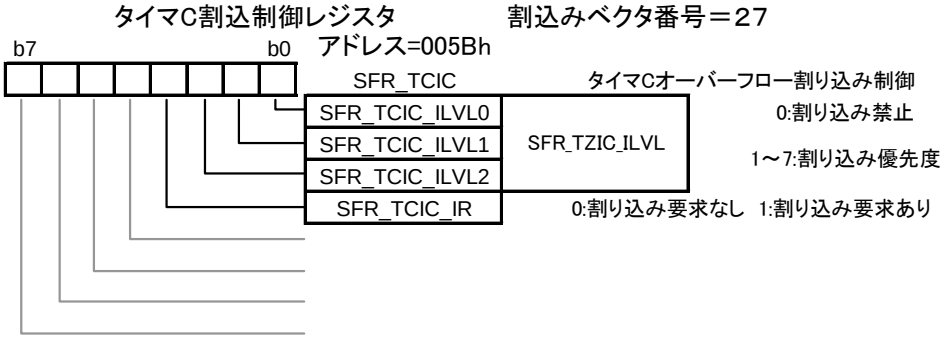


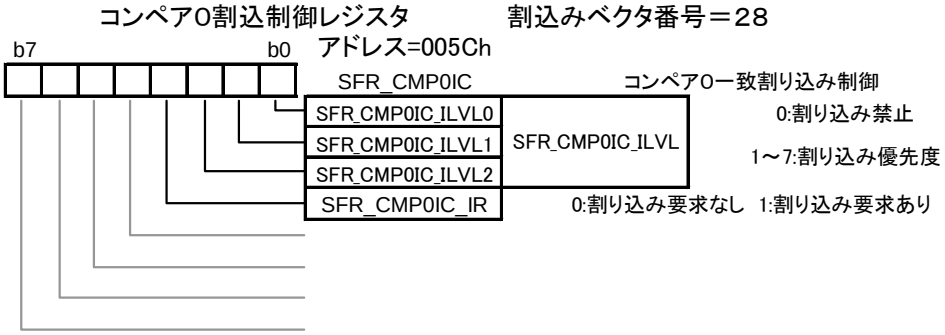
CMP10端子出力　SFR_P3_3=1かつSFR_TC_OUT10=1かつCMP出力=1ならCMP10端子="H"
CMP11端子出力　SFR_P3_4=1かつSFR_TC_OUT11=1かつCMP出力=1ならCMP11端子="H"
CMP12端子出力　SFR_P3_5=1かつSFR_TC_OUT12=1かつCMP出力=1ならCMP12端子="H"
ただしSFR_TC_INV1=1ならば上記出力は反転される

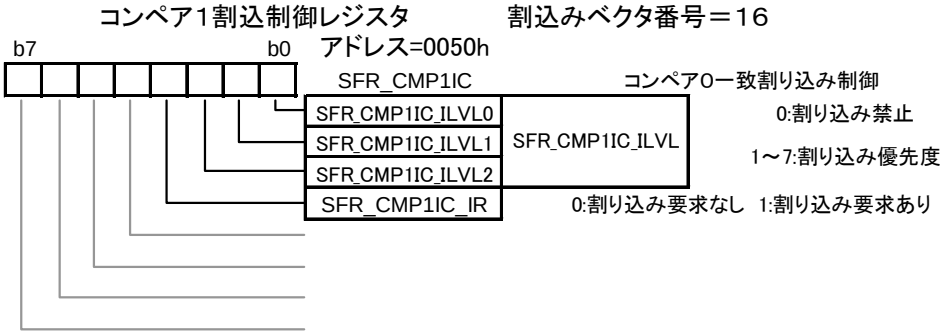
ポートP3方向レジスタ



CMP10端子出力使用時は1に設定する
CMP11端子出力使用時は1に設定する
CMP12端子出力使用時は1に設定する







コンペア１レジスタ(TM0)値はリセットでの初期値は“０”ですが、SFR_TC_CMP0EN ビットを“１”にすると“0xFFFF”になります。SFR_TC_CMP0EN ビットを“1”にした後で、コンペア１レジスタ(TM0)値を設定してください。

記述例

```
// タイマＣアウトプットコンペアモードで PWM 変調を制御します。
// TM0 設定で幅を 0.1%～100%まで 0.1%単位で設定できます。PWM 周期は 1.6uS*1000=1.6mS です。
        SFR_P1_2=1;             // P1_2(CMP02)端子出力"H"設定
        SFR_P1D_2=1;            // P1_2(CMP02)端子出力設定
        SFR_TCC0=0b00000000;    // タイマＣアウトプットコンペア
        SFR_TC_CK=2;            // タイマＣクロックソース=f8（20MHz/32=625KHz)
        SFR_TC_CMP1RL=1;        // CMP1 一致リロードあり
        SFR_TC_CMP0EN=1;        // アウトプットコンペア０有効
        SFR_TC_CMP0MD=2;        // CMP0 一致で CMP 出力"L"
        SFR_TC_CMP1MD=3;        // CMP1 一致で CMP 出力"H"
        SFR_TC_INV0=1;          // CMP0x 出力反転
        SFR_TC_OUT02=1;         // CMP02 出力有効

        SFR_TM0=0;              // PWM 幅最小に初期値設定
        SFR_TM1=0x3E7;          // PWM 周期=(999+1)/625KHz=1.6mS

        SFR_TC_S=1;             // タイマＣカウント開始

        SFR_TM0=0;              // PWM 幅=(0+1)/1000=0.1% =1.6uS

        SFR_TM0=0x1F3;          // PWM 幅=(499+1)/1000=50%

        SFR_TM0=0x3E7;          // PWM 幅=(999+1)/1000=100%

// PWM 幅を０％（全"H"）の場合は TM0 値の設定では出来ません。
        SFR_CMP1MD=2;           // PWM 幅=0%の設定

// 戻す
        SFR_TC_CMP1MD=3;        // PWM 幅=0%以外の設定
        SFR_TM0=0x1F3;          // PWM 幅=(499+1)/1000=50%
```

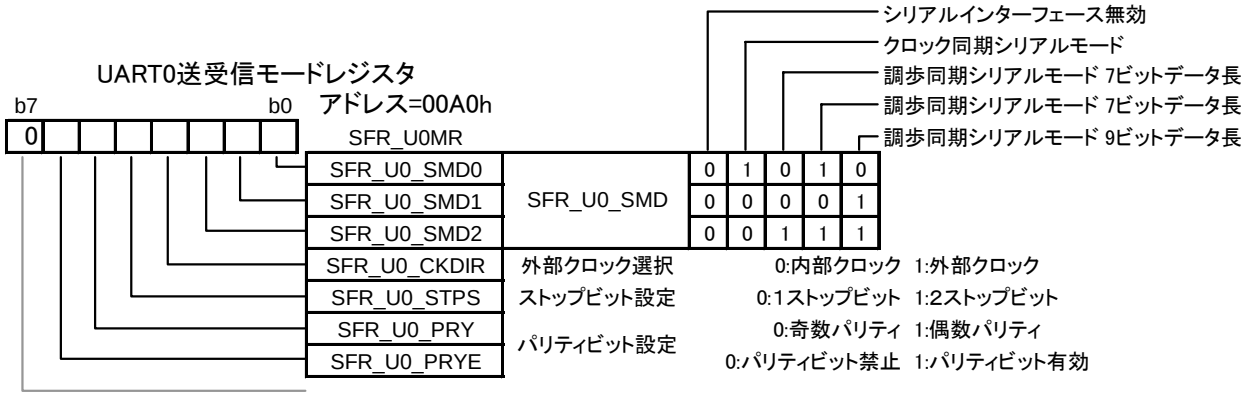
4 シリアルインターフェース

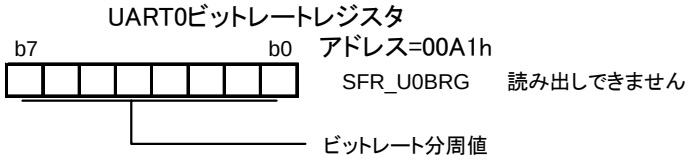
シリアルインターフェース（ＵＡＲＴ）は、Ｒ８Ｃ１５マニュアル上では１チャンネル搭載とありますが、実際には２チャンネル搭載しています。ＵＡＲＴ０は、クロック同期、調歩同期のモードを持ったシリアルインターフェースです。一方、ＵＡＲＴ１は、クロック同期モードは持っておらず、調歩同期モード専用です。また、ＵＡＲＴ１はシリアルデバッグ、Ｅ８デバッグ、シリアルフラッシュ書込みを兼ねていて、デバッグを行う時はＵＡＲＴ１はデバッグ通信に占有するため、ユーザーは使用できません。

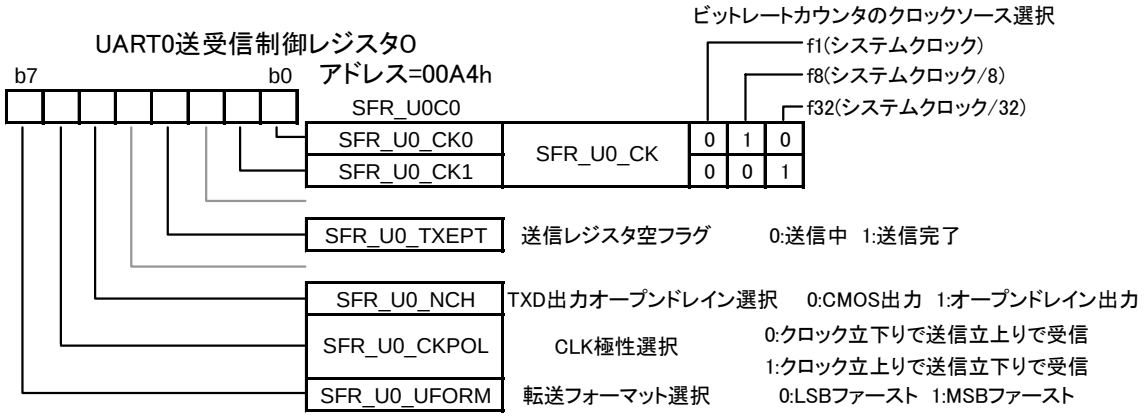
クロック同期シリアルモードは、送受信データ信号のほか送受信タイミング用クロックを用いてシリアル送受信を行います。送受信タイミングクロックは、内部で設定したクロックを出力する（マスター）または外部クロックを入力する(スレーブ)を選択します。

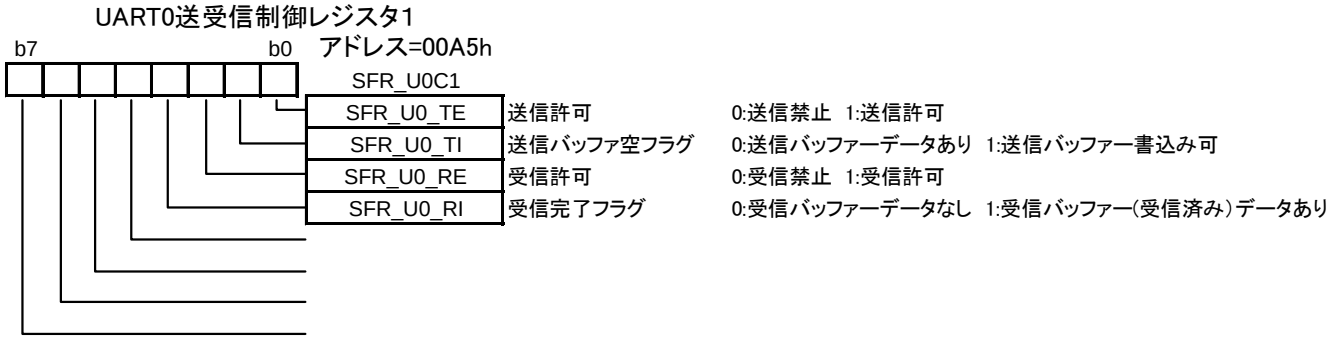
調歩同期シリアルモードは、ＲＳ２３２Ｃ通信のような送受信データ信号のみ（ハンドシェーク信号は別として）で、設定されたビットレート（ボーレート）クロックとシリアルデータのスタートビット・ストップビットにより、内部で同期をとり、シリアルデータ送受信を行います。

4-1 ＵＡＲＴ０

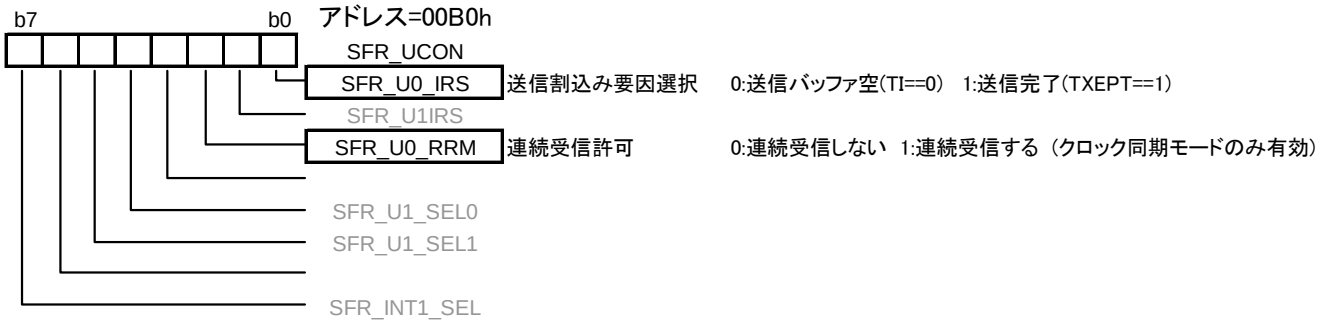




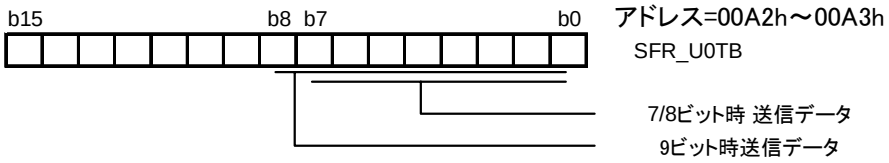




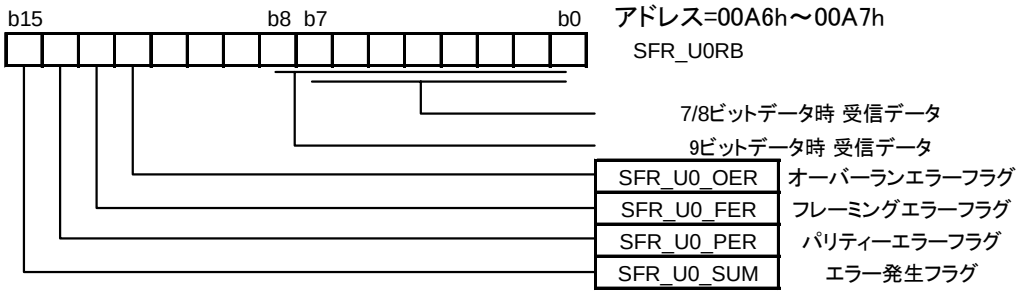
UART0送受信制御レジスタ2



UART0送信バッファーレジスタ

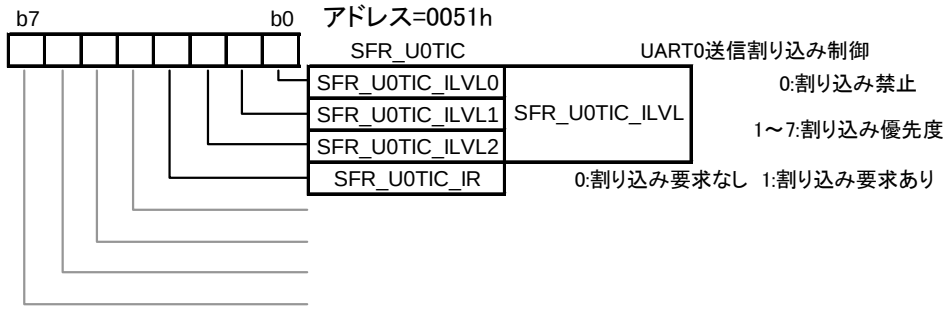


UART0受信バッファーレジスタ

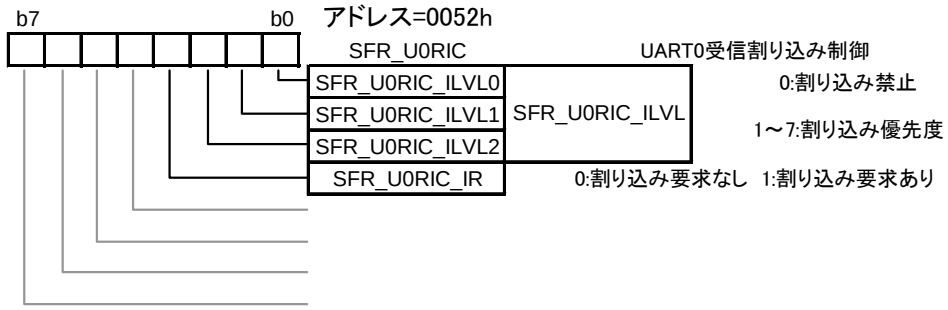


受信バッファーレジスタは常に１６ビット（ワード）単位で読み出してください。（もし、バイト単位で読み出す場合には必ず（必要がなくても）上位８ビットを読み出してください。）

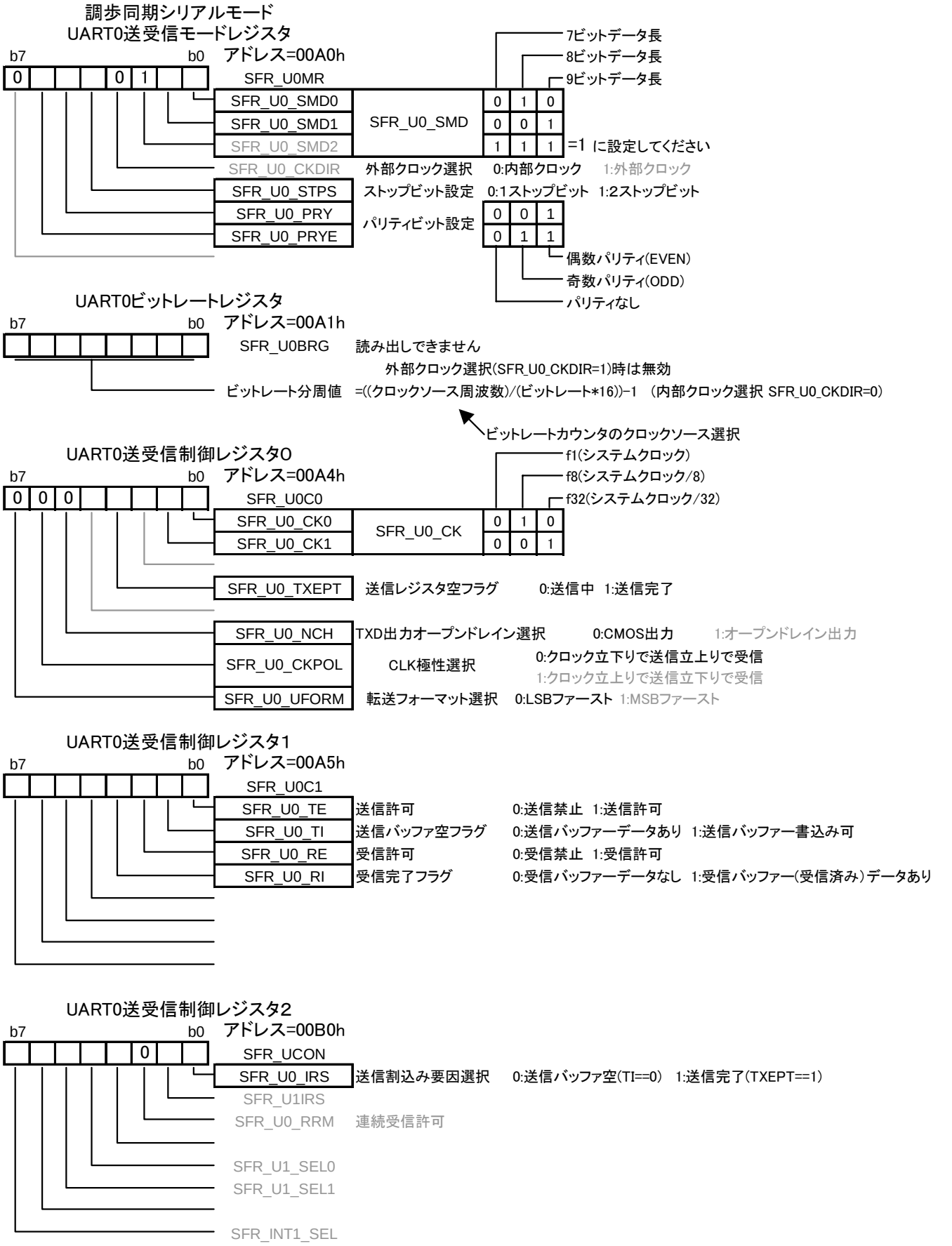
UART0送信割込制御レジスタ　　割込みベクタ番号＝17

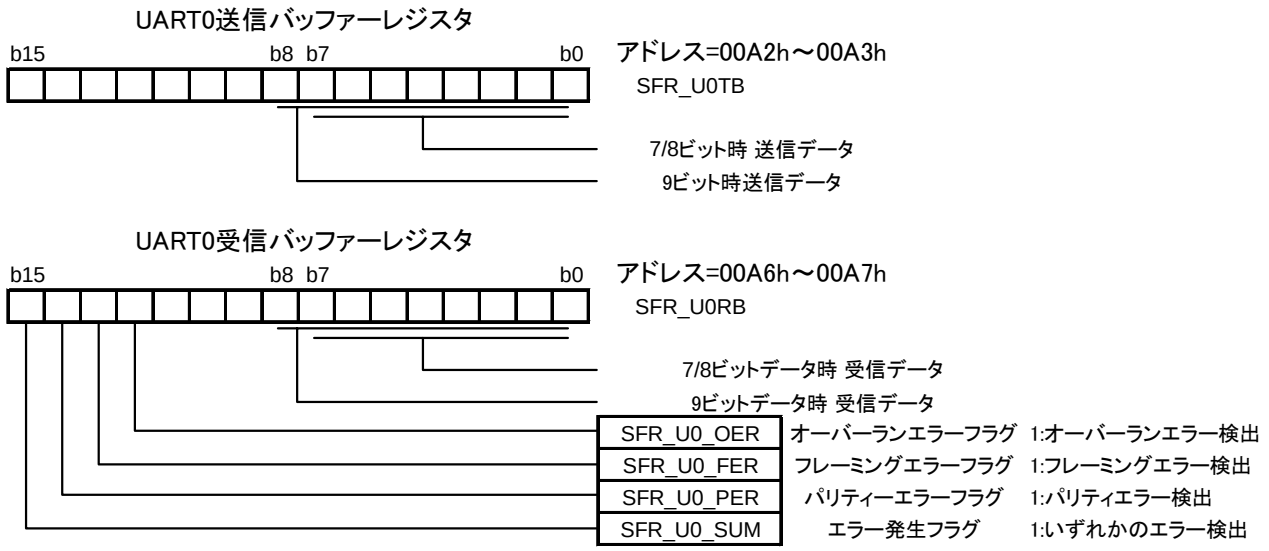


UART0受信割込制御レジスタ　　割込みベクタ番号＝18

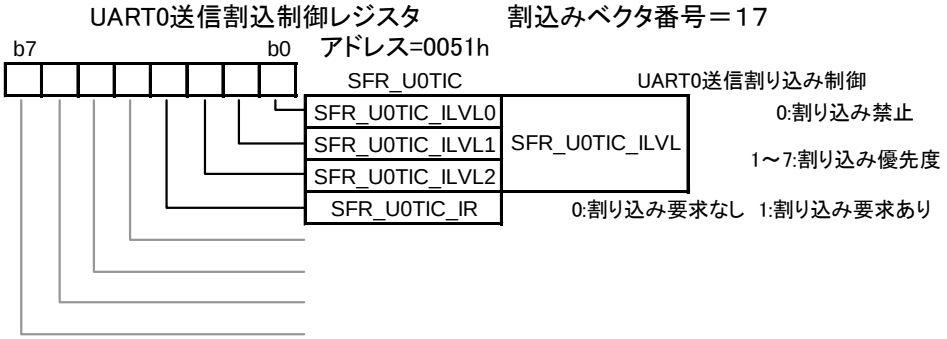


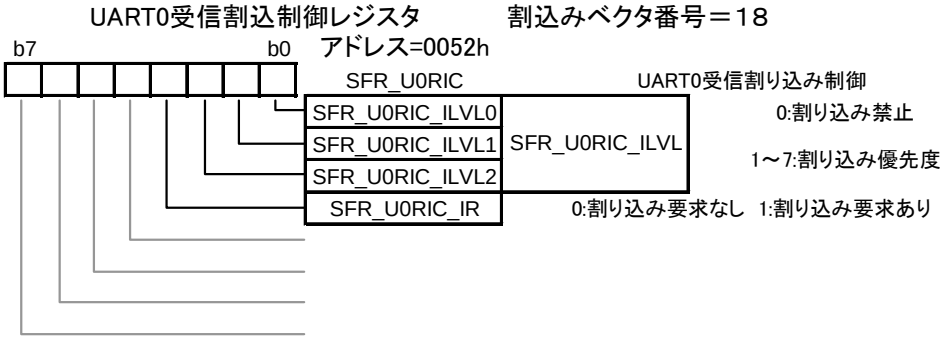
4-1-1 ＵＡＲＴ０（調歩同期シリアルモード）





受信バッファーレジスタは常に１６ビット（ワード）単位で読み出してください。（もし、バイト単位で読み出す場合には必ず上位８ビットを読み出してください。）下位８ビットのみを読み出した場合、受信完了フラグ(SFR_U0_RI)が“０”にクリアされません。





ＵＡＲＴの調歩同期モードでの通信フォーマット・ボーレート設定値早見表

SFR_UxMR設定値									
	PRYE	PRY	STPS			SMD1	SMD0		
b7							b0	HEX	
0	0	0	0	0	1	0	0	0x04	7ビットデータ パリティなし ストップビット1
0	1	0	0	0	1	0	0	0x44	7ビットデータ 奇数パリティ ストップビット1
0	1	1	0	0	1	0	0	0x64	7ビットデータ 偶数パリティ ストップビット1
0	0	0	1	0	1	0	0	0x14	7ビットデータ パリティなし ストップビット2
0	1	0	1	0	1	0	0	0x54	7ビットデータ 奇数パリティ ストップビット2
0	1	1	1	0	1	0	0	0x74	7ビットデータ 偶数パリティ ストップビット2
0	0	0	0	0	1	0	1	0x05	8ビットデータ パリティなし ストップビット1
0	1	0	0	0	1	0	1	0x45	8ビットデータ 奇数パリティ ストップビット1
0	1	1	0	0	1	0	1	0x65	8ビットデータ 偶数パリティ ストップビット1
0	0	0	1	0	1	0	1	0x15	8ビットデータ パリティなし ストップビット2
0	1	0	1	0	1	0	1	0x55	8ビットデータ 奇数パリティ ストップビット2
0	1	1	1	0	1	0	1	0x75	8ビットデータ 偶数パリティ ストップビット2
0	0	0	0	0	1	1	0	0x04	9ビットデータ パリティなし ストップビット1
0	1	0	0	0	1	1	0	0x44	9ビットデータ 奇数パリティ ストップビット1
0	1	1	0	0	1	1	0	0x64	9ビットデータ 偶数パリティ ストップビット1
0	0	0	1	0	1	1	0	0x14	9ビットデータ パリティなし ストップビット2
0	1	0	1	0	1	1	0	0x54	9ビットデータ 奇数パリティ ストップビット2
0	1	1	1	0	1	1	0	0x74	9ビットデータ 偶数パリティ ストップビット2

	システムクロック 20MHz時			システムクロック 8MHz時		
ボーレート	SFR_UxC0 設定値	SFR_UxBRG 設定値	実際のボーレート	SFR_UxC0 設定値	SFR_UxBRG 設定値	実際のボーレート
300	0x02	0x81	300	0x01	0xCF	300
600	0x02	0x40	601	0x01	0x67	601
1200	0x01	0x81	1202	0x01	0x33	1202
2400	0x01	0x40	2404	0x00	0xCF	2404
4800	0x01	0x20	4735	0x00	0x67	4808
9600	0x00	0x81	9615	0x00	0x33	9615
14400	0x00	0x56	14368	0x00	0x22	14286
19200	0x00	0x40	19231	0x00	0x19	19231
28800	0x00	0x2A	29070	0x00	0x10	29412
31250	0x00	0x27	31250	0x00	0x0F	31250
38400	0x00	0x20	37879	0x00	0x0C	38462
57600	0x00	0x15	56818	0x00	0x08	55556
115200	0x00	0x0A	113636	0x00	0x03	125000

上記の SFR_UxMR/SFR_UxC0/SFR_UxBRG はＵＡＲＴ０では SFR_U0MR/SFR_U0C0/SFR_U0BRG、ＵＡＲＴ１では SFR_U1MR/SFR_U1C0/SFR_U1BRG と指定してください。

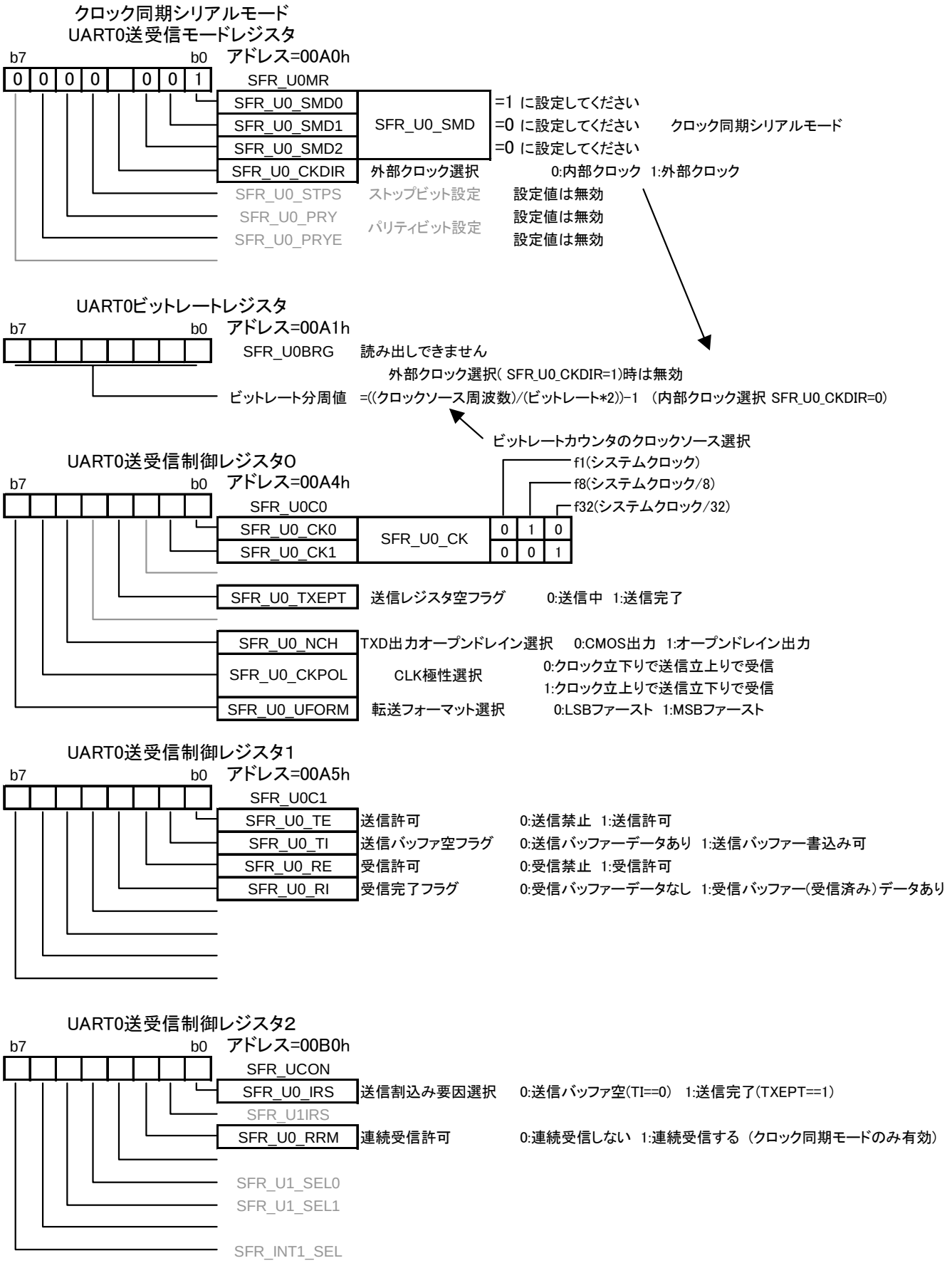
設定例

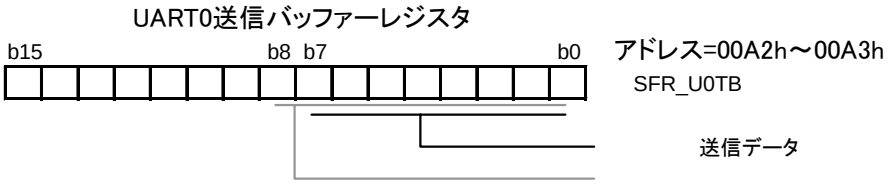
```
// 調歩同期 19200bps 8 ビットデータ長パリティなしストップビット 2 に設定　割込み未使用
// システムクロック＝２０ＭＨｚの場合
//      SFR_P1D &=0b11001111;     // P1_4 P1_5 端子を入力に設定（この行は省略可能）
        SFR_U0C0=0x00;            //
        SFR_U0BRG=0x40;           // 19200bps
        SFR_U0MR=0x05;            // 8 ビットデータ、パリティなし、ストップ１
        SFR_U0_TE=1;              // 送信許可
        SFR_U0_RE=1;              // 受信許可

unsigned char data;       // 送受信データ値

// データ受信
        while(SFR_U0_RI==0)       // データ受信済み？
                ;
        data=SFR_U0RB & 0xff;     // データ受信（上位 8 ビットのエラーフラグをマスク）

// データ送信
        while(SFR_U0_TI==0)       // 送信バッファー書込み可？
                ;
        SFR_U0TB=data;            // 送信バッファーに送信データ書き込み
```

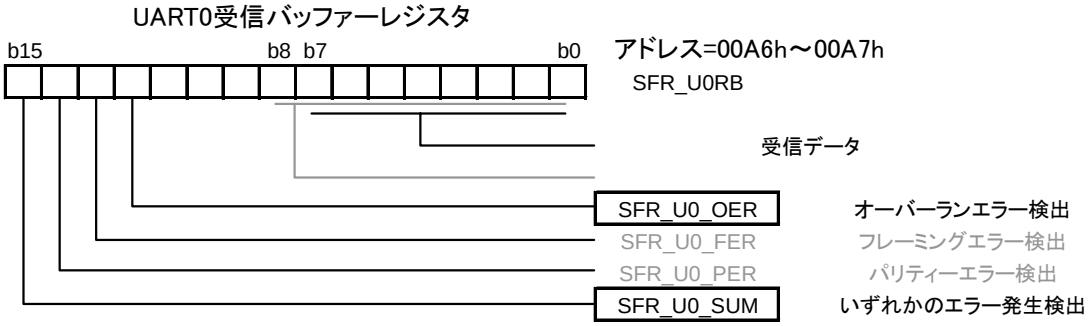
記述例

```
// 調歩同期 115200bps 8 ビットデータ長パリティなしストップビット 2 に設定　受信に割込み使用
// システムクロック＝２０ＭＨｚの場合
void UART0_init(void)
{
//      SFR_P1D &=0b11001111;     // P1_4 P1_5 端子を入力に設定（この行は省略可能）
        SFR_U0C0=0x00;            // 調歩同期モード
        SFR_U0BRG=0x0A;           // 115200bps
        SFR_U0MR=0x05;            // 8 ビットデータ、パリティなし、ストップ１
        SFR_U0RIC=6;              // ＵＡＲＴ０受信割込みレベル６
//      SFR_U0TIC=0;              // ＵＡＲＴ０送信割込み禁止（この行は省略可能）
        SFR_U0_TE=1;              // 送信許可
        SFR_U0_RE=1;              // 受信許可
}

unsigned int rx_data;     // 受信データ値
// データ受信割込み
void UART0_rx_int(void)
#pragma interrupt UART0_rx_int (vect=18)
{
        rx_data=SFR_U0RB;         // データ受信（上位 8 ビットにエラーフラグを含む）
        // データ受信処理
}

void main(void)
{
        UART0_init();             // UART0 初期化
        ei();                     // 割込み許可
        // メイン処理を記述
}
```

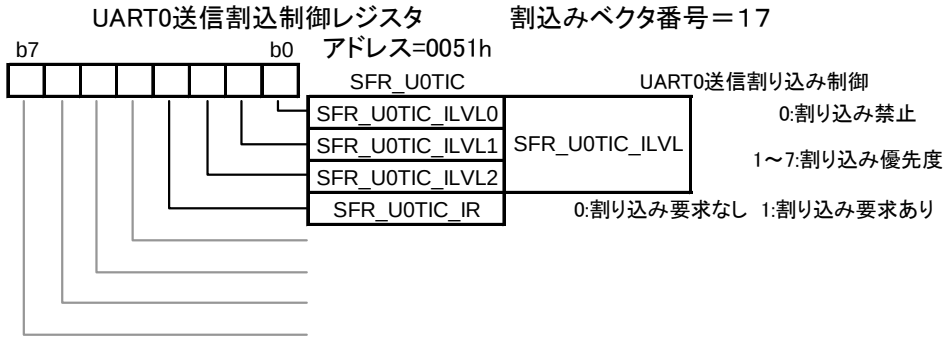
4-1-2 ＵＡＲＴ０（クロック同期シリアルモード）

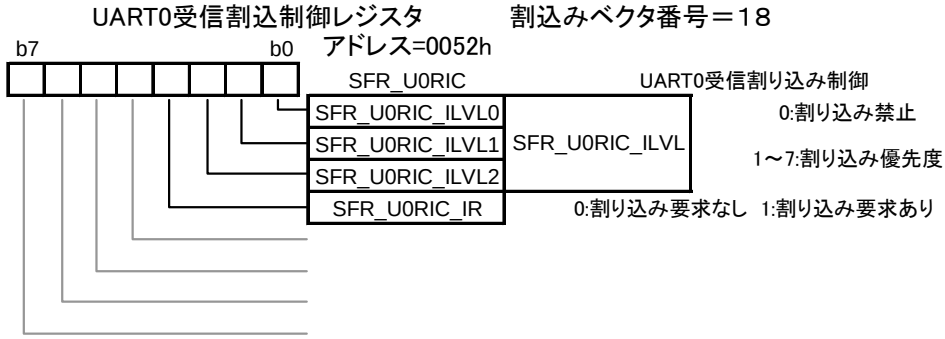






受信バッファーレジスタは常に１６ビット（ワード）単位で読み出してください。（もし、バイト単位で読み出す場合には必ず（必要がなくても）上位８ビットを読み出してください。）下位８ビットのみを読み出した場合、受信完了フラグ(SFR_U0_RI)が“０”にクリアされません。



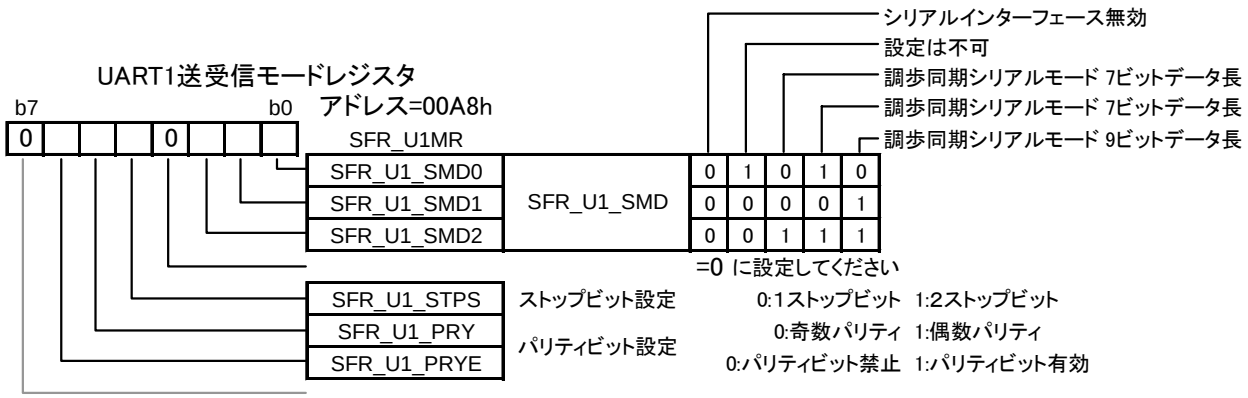


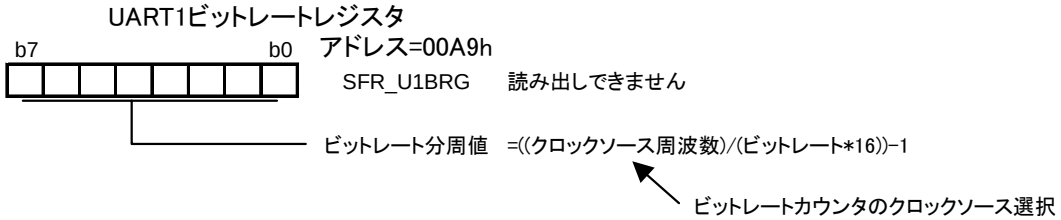
設定例

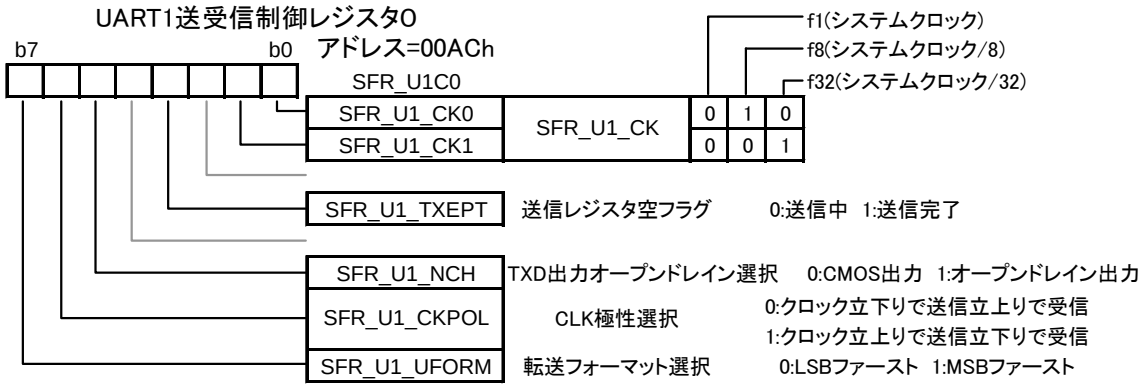
```
// クロック同期 内部クロック 1MHz に設定　割込み未使用
// システムクロック＝２０ＭＨｚの場合
//      SFR_P1D &=0b10001111;     // P1_4 P1_5 P1_6 端子を入力に設定（この行は省略可能）
        SFR_U0MR=0b00000001;      // クロック同期モード 内部クロック選択
        SFR_U0C0=0b10000000;      // MSB ファースト クロック立下り送信/立上り受信 クロックソース f1
        SFR_U0BRG=10-1;           // シリアルクロック　(20MHz)/(10*2)=１ＭＨｚ設定
//      SFR_U0_IRS=0;             // 送信バッファ空で割り込み発生
        SFR_U0_RRM=0;             // 連続受信しない
        SFR_U0_TE=1;              // 送信許可
        SFR_U0_RE=0;              // 受信禁止

unsigned char data;       // 送受信データ値

// データ送信
        while(SFR_U0_TI==0)       // 送信バッファー書込み可？
                ;
        SFR_U0TB=data;            // 送信バッファーに送信データ書き込み

// データ受信
        while(SFR_U0_TI==0)       // ダミー
                ;
        while(SFR_U0_TXEPT==0)    // データ送信中？
                ;
        SFR_U0_RE=1;              // 受信許可
        SFR_U0_TB=0xff;           // ダミーデータ送信
        while(SFR_U0_RI==0)       // データ受信済み？
                ;
        data=SFR_U0RB & 0xff;     // データ受信（上位 8 ビットのエラーフラグをマスク）
        SFR_U0_RE=0;              // 受信禁止
```

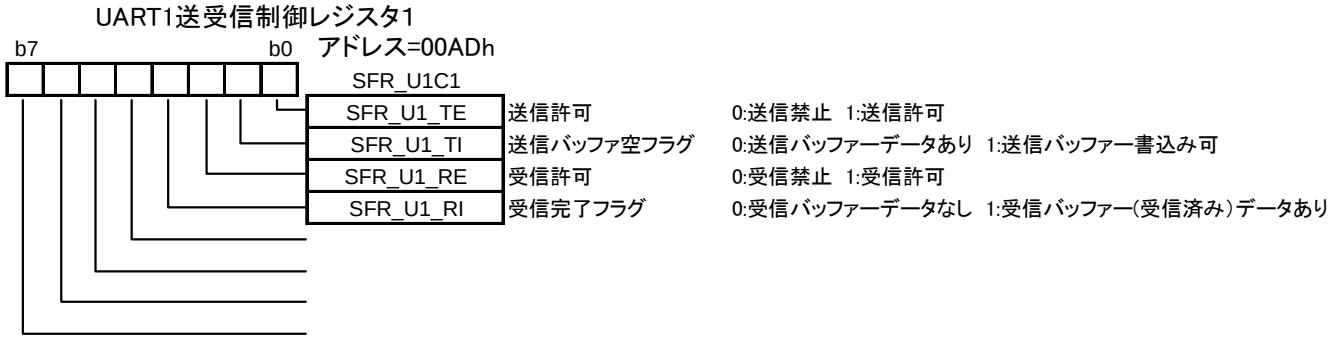
経験でのこと）
SFR_U0_TI を確認して SFR_U0TB にデータを書き込んで SFR_U0_TXEPT を確認して送信終了と判断すると、まだ送信終了前だったということがあります。実際に SFR_U0_TI と SFR_U0_TXEPT のフラグが変化するのは内部シリアルクロックに同期したタイミングのようです。そのため、上記のように SFR_U0_TXEPT 確認前にダミーとして SFR_U0_TI の確認を必要ないのに入れています。連続送信時、SFR_U0_TI 確認を行う際にも同じことが言えます。連続送信時には、SFR_U0_TI の確認を必要がなくとも、2 回行ってから SFR_U0TB への書き込みを行うと、回避できます。
このことは、非同期シリアルでの送信でも同じことが言えます。

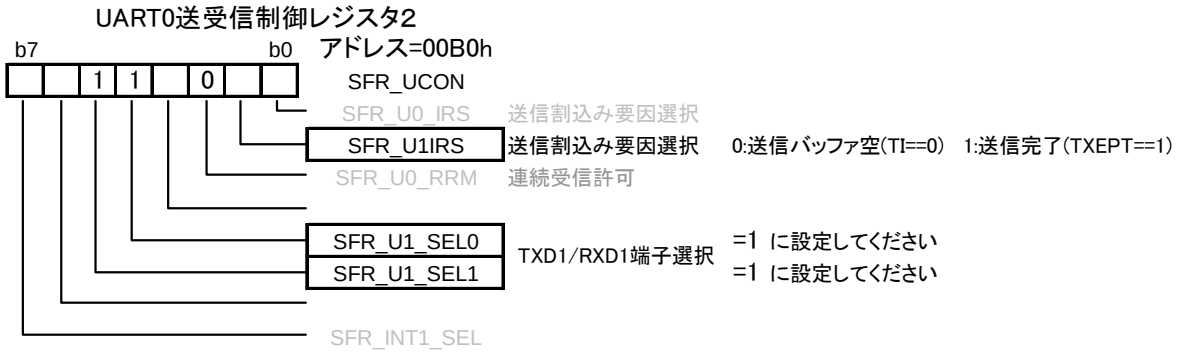
4-2 ＵＡＲＴ１

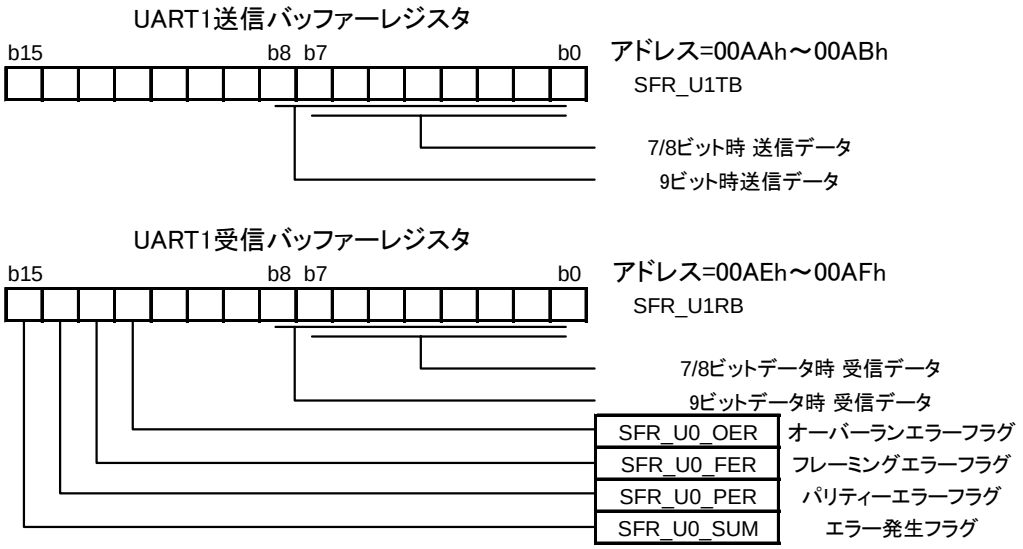

UART1送受信モードレジスタ
b7
b0
アドレス=00A8h
SFR_U1MR
0
0
SFR_U1_SMD0
SFR_U1_SMD1
SFR_U1_SMD2
SFR_U1_SMD
0 1 0 1 0
0 0 0 0 1
0 0 1 1 1
シリアルインターフェース無効
設定は不可
調歩同期シリアルモード 7ビットデータ長
調歩同期シリアルモード 7ビットデータ長
調歩同期シリアルモード 9ビットデータ長
=0 に設定してください
SFR_U1_STPS
ストップビット設定
0:1ストップビット 1:2ストップビット
SFR_U1_PRY
SFR_U1_PRYE
パリティビット設定
0:奇数パリティ 1:偶数パリティ
0:パリティビット禁止 1:パリティビット有効


UART1ビットレートレジスタ
b7
b0
アドレス=00A9h
SFR_U1BRG
読み出しできません
ビットレート分周値 =((クロックソース周波数)/(ビットレート*16))-1
ビットレートカウンタのクロックソース選択

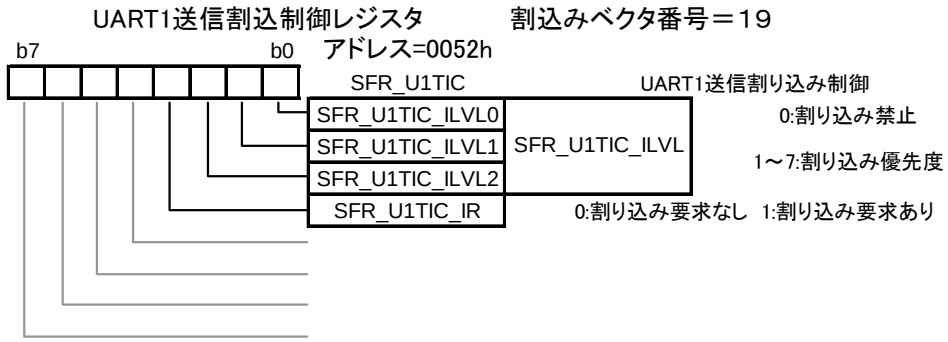

UART1送受信制御レジスタ0
b7
b0
アドレス=00ACh
SFR_U1C0
SFR_U1_CK0
SFR_U1_CK1
SFR_U1_CK
0 1 0
0 0 1
f1(システムクロック)
f8(システムクロック/8)
f32(システムクロック/32)
SFR_U1_TXEPT
送信レジスタ空フラグ
0:送信中 1:送信完了
SFR_U1_NCH
TXD出力オープンドレイン選択
0:CMOS出力 1:オープンドレイン出力
SFR_U1_CKPOL
CLK極性選択
0:クロック立下りで送信立上りで受信
1:クロック立上りで送信立下りで受信
SFR_U1_UFORM
転送フォーマット選択
0:LSBファースト 1:MSBファースト

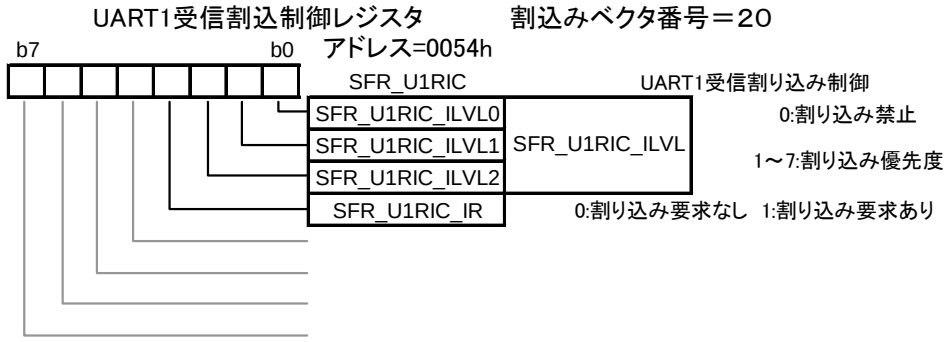

UART1送受信制御レジスタ1
b7
b0
アドレス=00ADh
SFR_U1C1
SFR_U1_TE
送信許可
0:送信禁止 1:送信許可
SFR_U1_TI
送信バッファ空フラグ
0:送信バッファーデータあり 1:送信バッファー書込み可
SFR_U1_RE
受信許可
0:受信禁止 1:受信許可
SFR_U1_RI
受信完了フラグ
0:受信バッファーデータなし 1:受信バッファー(受信済み)データあり


UART0送受信制御レジスタ2
b7
b0
アドレス=00B0h
SFR_UCON
1 1 0
SFR_U0_IRS
送信割込み要因選択
SFR_U1IRS
送信割込み要因選択
0:送信バッファ空(TI==0) 1:送信完了(TXEPT==1)
SFR_U0_RRM
連続受信許可
SFR_U1_SEL0
SFR_U1_SEL1
TXD1/RXD1端子選択
=1 に設定してください
=1 に設定してください
SFR_INT1_SEL

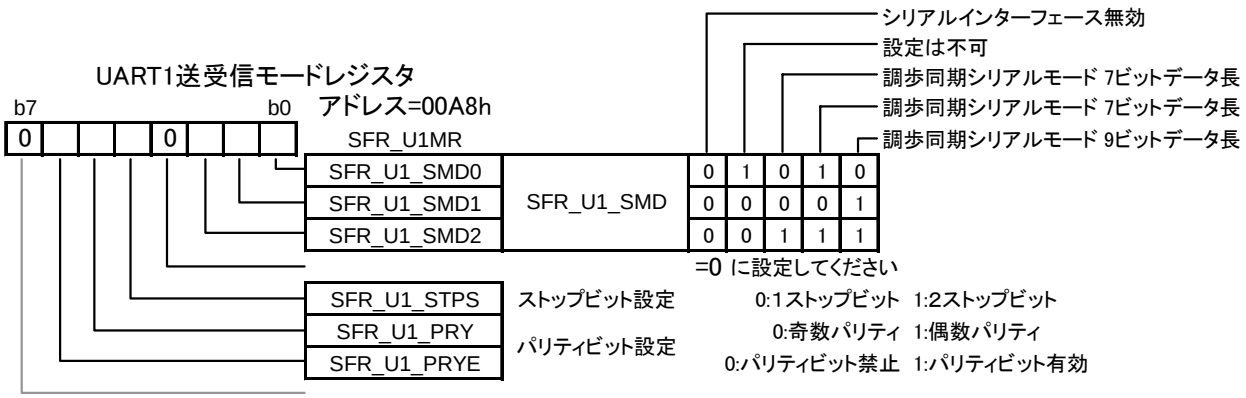


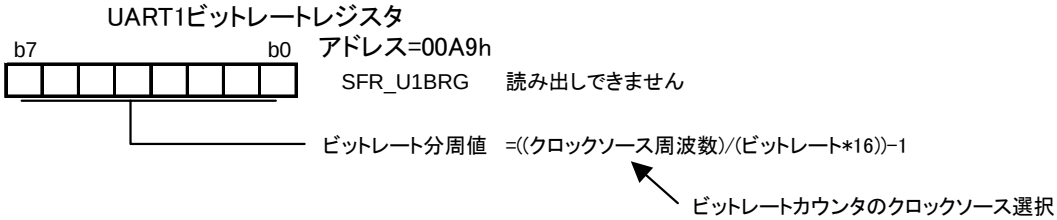
受信バッファーレジスタは常に１６ビット（ワード）単位で読み出してください。（もし、バイト単位で読み出す場合には必ず（必要がなくても）上位８ビットを読み出してください。）

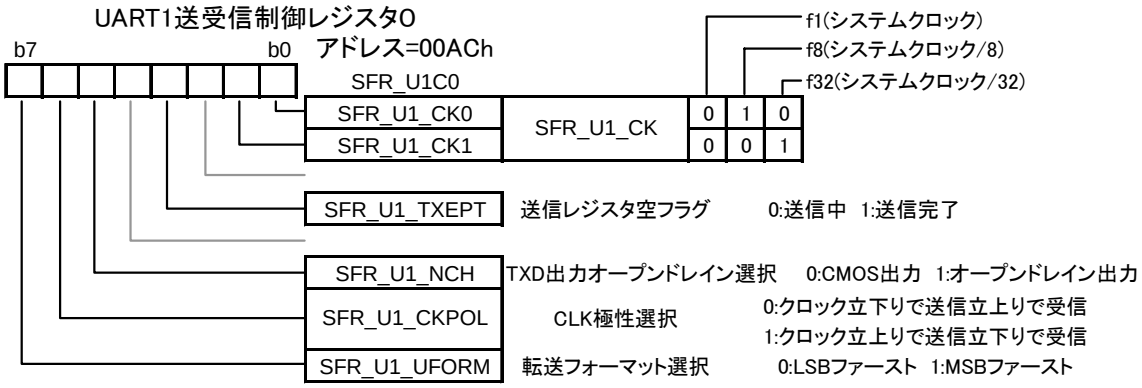


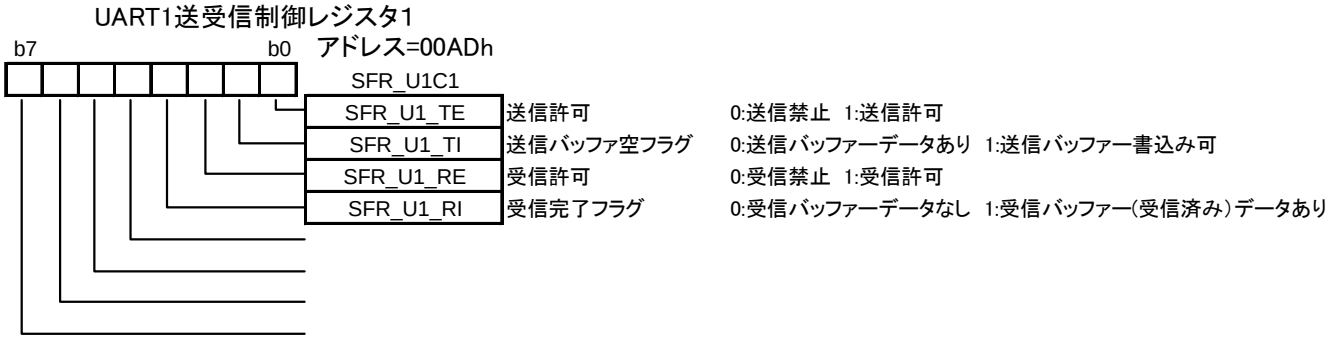


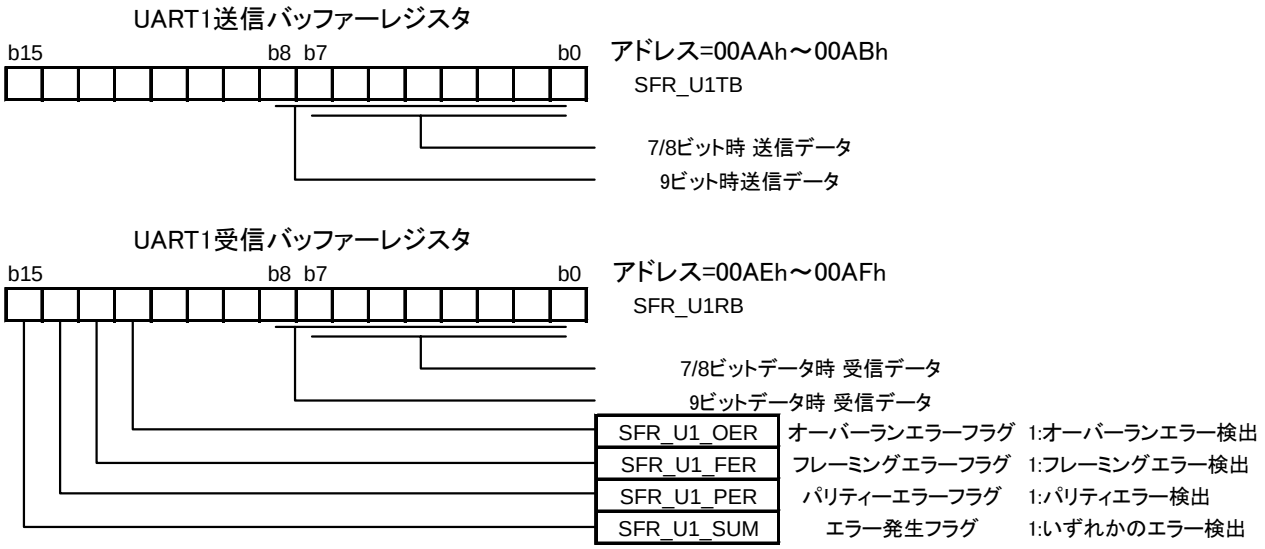
4-2-1 ＵＡＲＴ１（調歩同期シリアルモード）



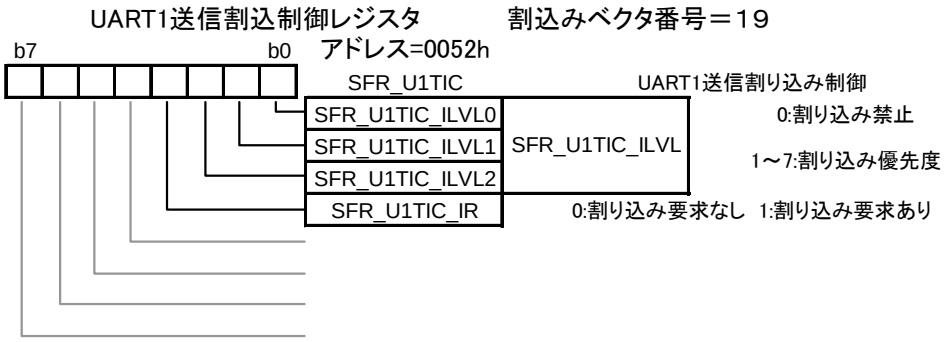


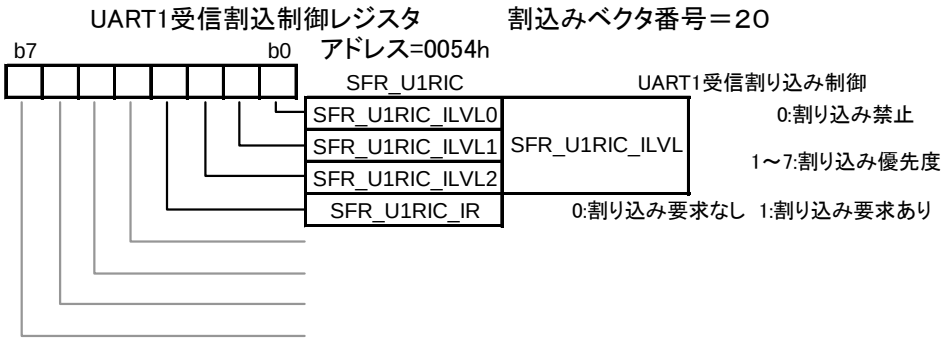






受信バッファーレジスタは常に１６ビット（ワード）単位で読み出してください。（もし、バイト単位で読み出す場合には必ず上位８ビットを読み出してください。）下位８ビットのみを読み出した場合、受信完了フラグ(SFR_U1_RI)が“０”にクリアされません。





設定例

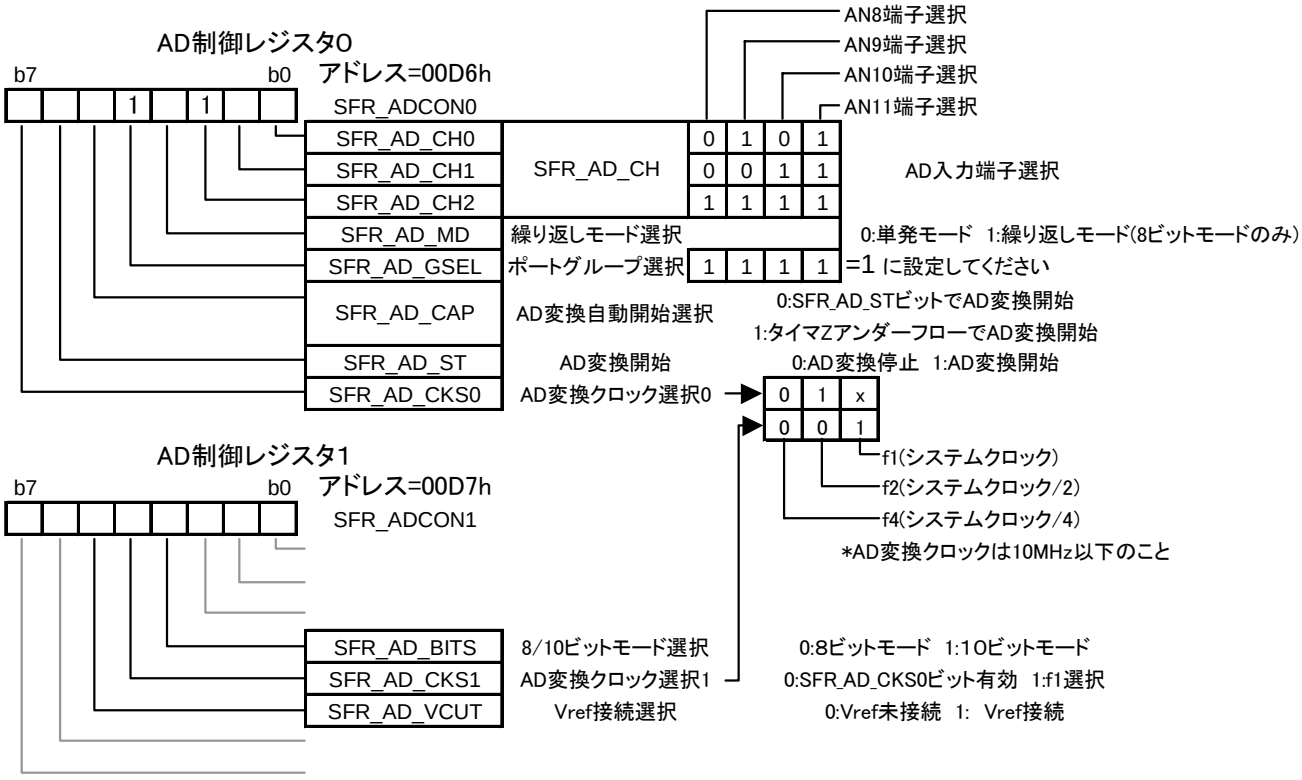
```
// 調歩同期 19200bps 8 ビットデータ長パリティなしストップビット 2 に設定　割込み未使用
// システムクロック＝２０ＭＨｚの場合
//      SFR_P4D_5=0;            // P4_5 端子(RXD1 端子)を入力に設定（この行は省略可能）
//      SFR_P3D_7=0;            // P3_7 端子(TXD1 端子)を入力に設定（この行は省略可能）
        SFR_UCON=0x30;          // RXD1/TXD1 端子有効
        SFR_U1C0=0x00;          //
        SFR_U1BRG=0x40;         // 19200bps
        SFR_U1MR=0x05;          // 8 ビットデータ、パリティなし、ストップ１
        SFR_U1_TE=1;            // 送信許可
        SFR_U1_RE=1;            // 受信許可

unsigned char data;     // 送受信データ値

// データ受信
        while(SFR_U1_RI==0)     // データ受信済み？
                ;
        data=SFR_U1RB & 0xff;   // データ受信（上位 8 ビットのエラーフラグをマスク）

// データ送信
        while(SFR_U1_TI==0)     // 送信バッファー書込み可？
                ;
        SFR_U1TB=data;          // 送信バッファーに送信データ書き込み
```

記述例

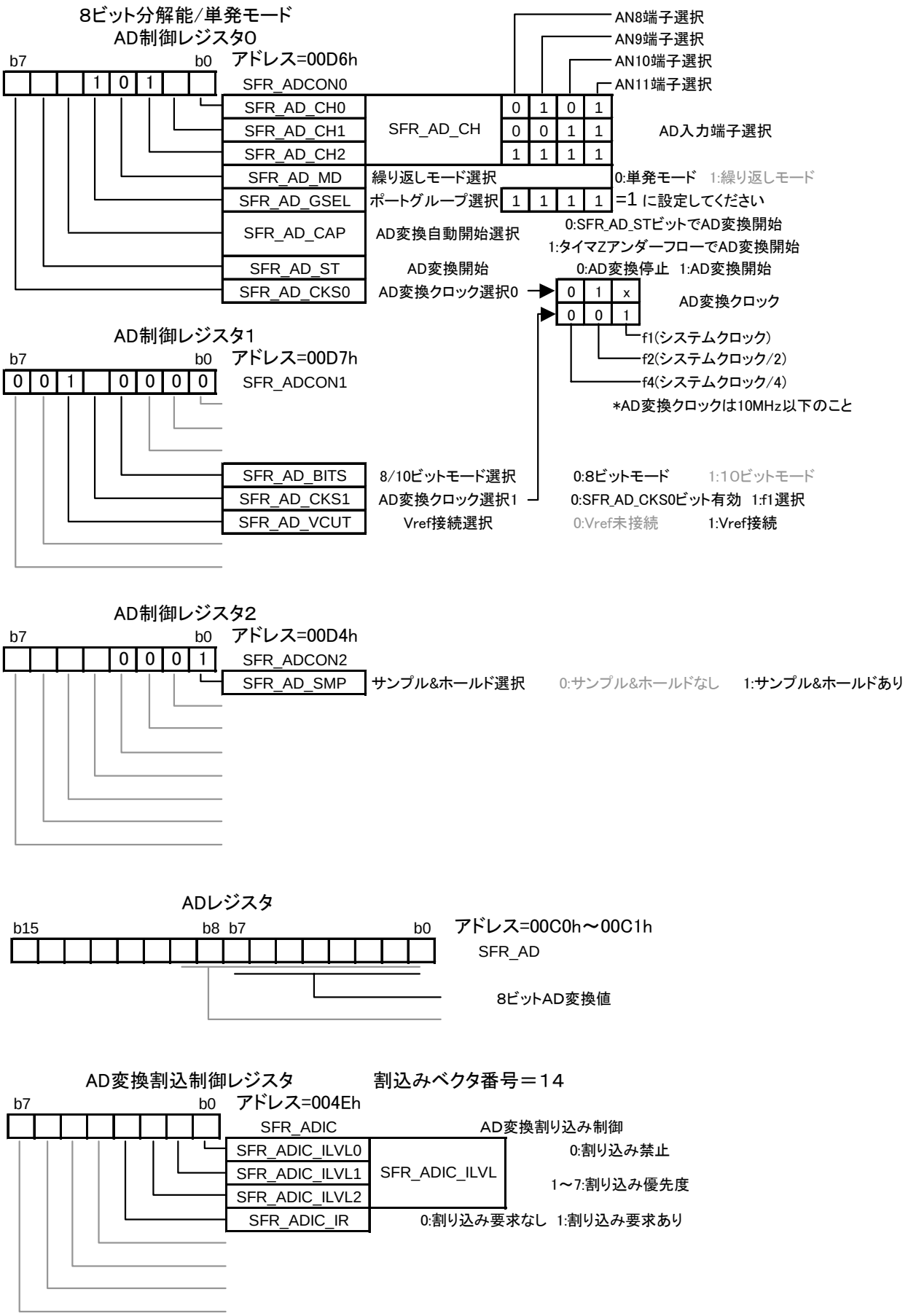
```
// 調歩同期 115200bps 8 ビットデータ長パリティなしストップビット 2 に設定　受信に割込み使用
// システムクロック＝２０ＭＨｚの場合
void UART1_init(void)
{
//      SFR_P4D_5=0;            // P4_5 端子(RXD1 端子)を入力に設定（この行は省略可能）
//      SFR_P3D_7=0;            // P3_7 端子(TXD1 端子)を入力に設定（この行は省略可能）
        SFR_UCON=0x30;          // RXD1/TXD1 端子有効
        SFR_U1C0=0x00;          // 調歩同期モード
        SFR_U1BRG=0x0A;         // 115200bps
        SFR_U1MR=0x05;          // 8 ビットデータ、パリティなし、ストップ１
        SFR_U1RIC=6;            // ＵＡＲＴ１受信割込みレベル６
//      SFR_U1TIC=0;            // ＵＡＲＴ１送信割込み禁止（この行は省略可能）
        SFR_U1_TE=1;            // 送信許可
        SFR_U1_RE=1;            // 受信許可
}

unsigned int rx_data;   // 受信データ値
// データ受信割込み
void UART1_rx_int(void)
#pragma interrupt UART1_rx_int (vect=20)
{
        rx_data=SFR_U1RB;       // データ受信（上位 8 ビットにエラーフラグを含む）
        // データ受信処理
}

void main(void)
{
        UART1_init();           // UART1 初期化
        ei();                   // 割込み許可
        // メイン処理を記述
}
```

５．ＡＤコンバータ

外部からのアナログ入力電圧値をデジタル数値に変換します。
R8C15 では 4 本のアナログ入力端子(AN8 端子(P1_0 端子)～AN11 端子(P1_3 端子))をアナログマルチプレクサで選択し（切り替え）て、各端子の逐次比較アナログデジタル変換が可能です。アナログ入力部にはサンプルアンドホールドを備えていて、サンプルアンドホールドを有効にすると変換速度が向上します。(通常はサンプルアンドホールド有効状態で使用します)。このときの AD 変換時間は、８ビットモード時に最小 2.8uS(AD 変換クロック*28)、１０ビットモード時最小 3.3uS(AD 変換クロック*33)です。
ＡＤ変換の分解能は８ビットと１０ビットが選択できます。８ビット分解能の場合にはフルスケール電圧値を２５６分割、１０ビット分解能の場合にはフルスケール電圧値を１０２４分割した値を比較した値をデジタル値で示します。変換精度に関しては R8C15 ハードウェアマニュアルを参照してください。
８ビットモードでは繰り返しモードが選択できます。繰り返しモードでは 1 本の選択したアナログ入力端子を繰り返して変換を行います。単発モードでは、変換開始指令（プログラムまたはタイマＺアンダーフロー信号）により変換を行います。



5-1-1 ＡＤコンバータ（8 ビット分解能/単発モード）



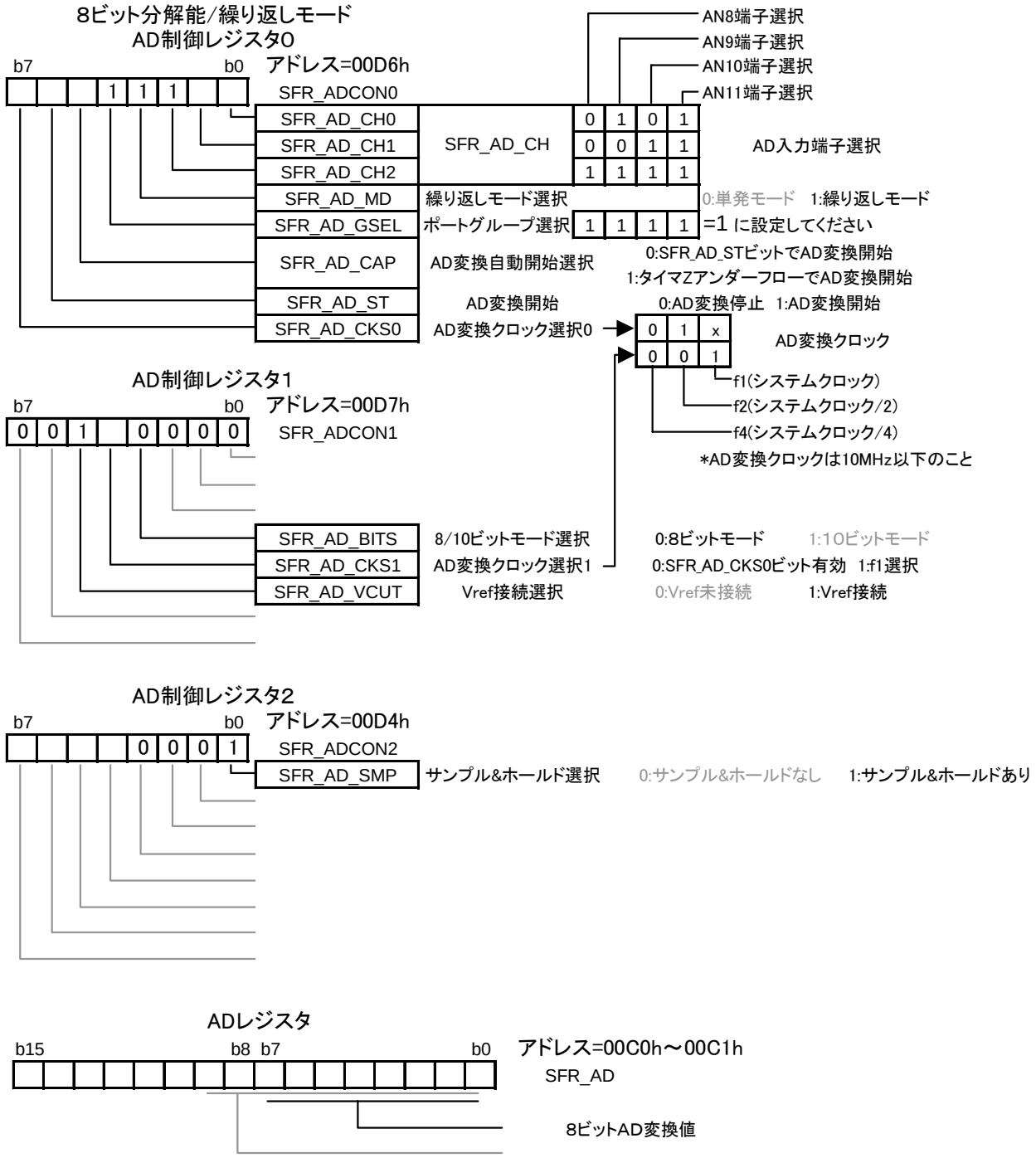
設定例

```
// ＡＤコンバータ 8 ビット分解能単発モードで使用する
        unsigned char value;
        SFR_ADCON0=0b10010100;    // 変換クロック f2 単発モード AN8 端子選択
        SFR_ADCON1=0b00100000;    // 8 ビットモード CKS0 有効 Vref 接続
        SFR_ADCON2=0b00000001;    // サンプル&ホールド有効
        SFR_AD_ST=1;              // AD 変換開始
        while (SFR_AD_ST==1)      // 変換終了待ち
                ;
        value=SFR_AD &0xff;       // AD 変換値読み込み
```

記述例

```
// 100uS 間隔でタイマＺアンダーフローにより正確なサンプリング間隔で 8 ビット AD 変換行う。
// ＡＤ初期化
void ad_init(void)
{
        SFR_ADCON0=0b10110100;    // 変換クロック f2 キャプチャー単発モード AN8 端子選択
        SFR_ADCON1=0b00100000;    // 8 ビットモード CKS0 有効 Vref 接続
        SFR_ADCON2=0b00000001;    // サンプル&ホールド有効
        SFR_ADIC=7;               // AD 割込みレベル７
        SFR_AD_ST=1;              // AD 変換開始
}

// タイマーＺ初期化
void tz_init(void)
{
        SFR_TZMR=0;
        SFR_TZ_CK=1;     // タイマＺクロックソース= f8 (20MHz/8=2.5MHz)
        SFR_PREZ=125-1;  //
        SFR_TZPR=2-1;    // インターバル時間 t=(125*2)/2500000=100uS
        SFR_TZ_S=1;      // タイマＺカウントスタート
}

// グローバル変数宣言
unsigned char value;              // AD 変換結果格納

// ＡＤ変換終了割り込み
void ad_int(void)
#pragma interrupt ad_int (vect=14)
{
        value=SFR_AD & 0xff;      // AD 変換結果読み込み(下位 8 ビット有効)
}

void main(void)
{
        ad_init();                // ＡＤ初期化
        tz_init();                // タイマＺ初期化
        ei();            // 割り込み許可

        // メイン処理を記述

}
```

5-1-2 ＡＤコンバータ（8 ビット分解能/繰り返しモード）

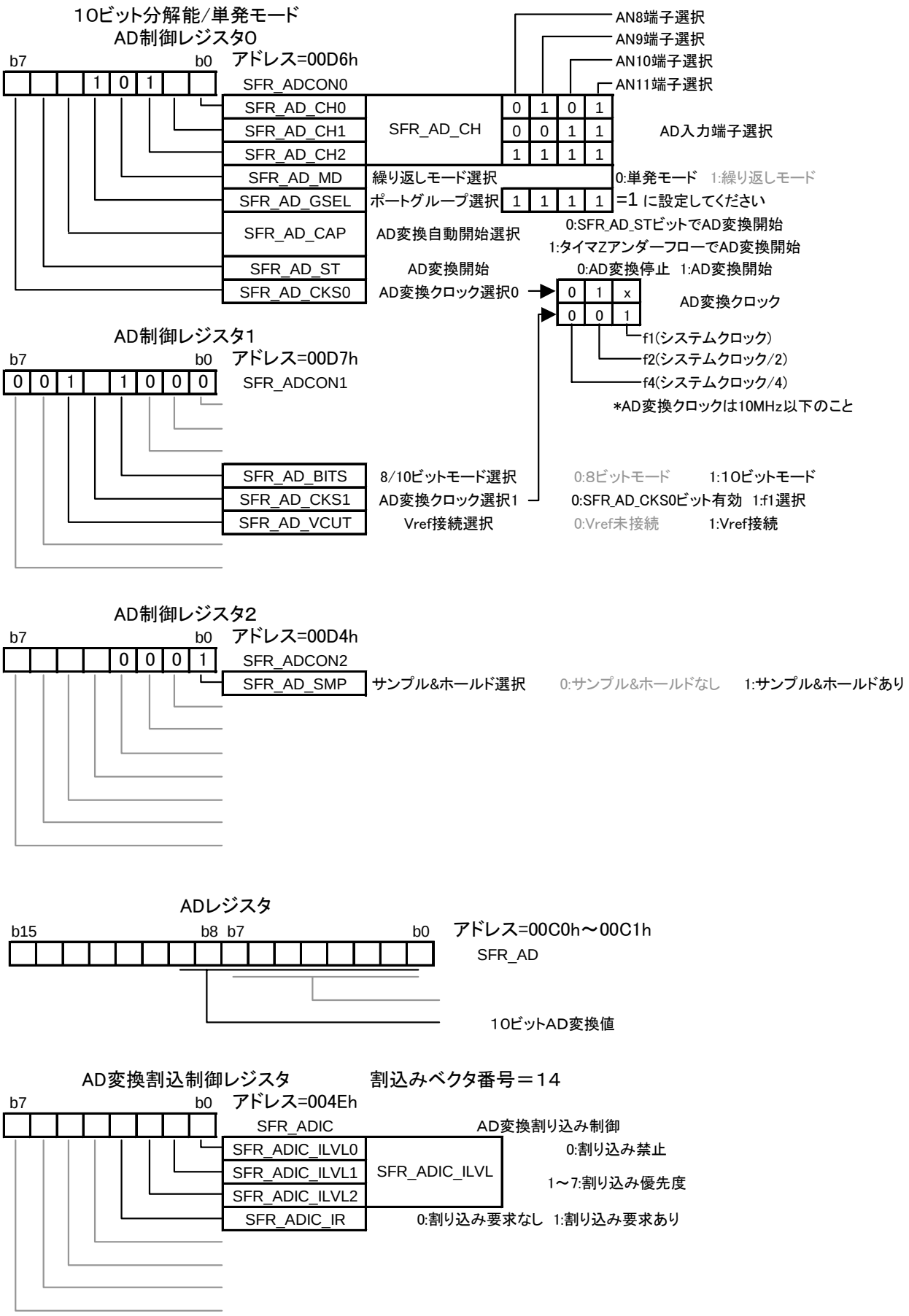


繰り返しモードの場合、AD 変換終了割り込みは発生しません。SFR_ADIC_IR ビットも当然“1”にはなりません。従って、最初に SFR_AD_ST を“1”にして、最初に AD 変換値を読み出すときは、８ビットモード時の AD 変換時間である 2.8uS(AD 変換クロック*28)以上たってから AD 変換値を読み出すようにしてください。

設定例

```
// ＡＤコンバータ 8 ビット分解能繰り返しモードで使用する
        unsigned char value;
        SFR_ADCON0=0b10011100;    // 変換クロック f2 繰り返しモード AN8 端子選択
        SFR_ADCON1=0b00100000;    // 8 ビットモード CKS0 有効 Vref 接続
        SFR_ADCON2=0b00000001;    // サンプル&ホールド有効
        SFR_AD_ST=1;              // AD 変換開始
                        …
                                  // 最初の変換終了待ち(省略する場合は AD 値読込まで 2.8uS 以上待つこと)

        while(1)
        {
                        …
                value=SFR_AD &0xff;       // AD 変換値読み込み
                        …
        }
```

5-1-3 ＡＤコンバータ（１０ビット分解能/単発モード）



設定例

```
// ＡＤコンバータ 10 ビット分解能単発モードで使用する
        unsigned char value;
        SFR_ADCON0=0b10010100;    // 変換クロック f2 単発モード AN8 端子選択
        SFR_ADCON1=0b00101000;    // 10 ビットモード CKS0 有効 Vref 接続
        SFR_ADCON2=0b00000001;    // サンプル&ホールド有効
        SFR_AD_ST=1;              // AD 変換開始
        while (SFR_AD_ST==1)      // 変換終了待ち
                ;
        value=SFR_AD &0x3ff;      // AD 変換値読み込み
```

記述例

```
// 500uS 間隔でタイマＺアンダーフローにより正確なサンプリング間隔で 10 ビット AD 変換行う。
// ＡＤ初期化
void ad_init(void)
{
        SFR_ADCON0=0b10110100;    // 変換クロック f2 キャプチャー単発モード AN8 端子選択
        SFR_ADCON1=0b00101000;    // 10 ビットモード CKS0 有効 Vref 接続
        SFR_ADCON2=0b00000001;    // サンプル&ホールド有効
        SFR_ADIC=7;               // AD 割込みレベル７
        SFR_AD_ST=1;              // AD 変換開始
}

// タイマーＺ初期化
void tz_init(void)
{
        SFR_TZMR=0;
        SFR_TZ_CK=1;      // タイマＺクロックソース= f8 (20MHz/8=2.5MHz)
        SFR_PREZ=250-1;   //
        SFR_TZPR=5-1;     // インターバル時間 t=(250*5)/2500000=500uS
        SFR_TZ_S=1;       // タイマＺカウントスタート
}

// グローバル変数宣言
unsigned int value;               // AD 変換結果格納

// ＡＤ変換終了割り込み
void ad_int(void)
#pragma interrupt ad_int (vect=14)
{
        value=SFR_AD & 0x3ff;     // AD 変換結果読み込み(下位 10 ビット有効)
}

void main(void)
{
        ad_init();                // ＡＤ初期化
        tz_init();                // タイマＺ初期化
        ei();             // 割り込み許可

        // メイン処理を記述

}
```

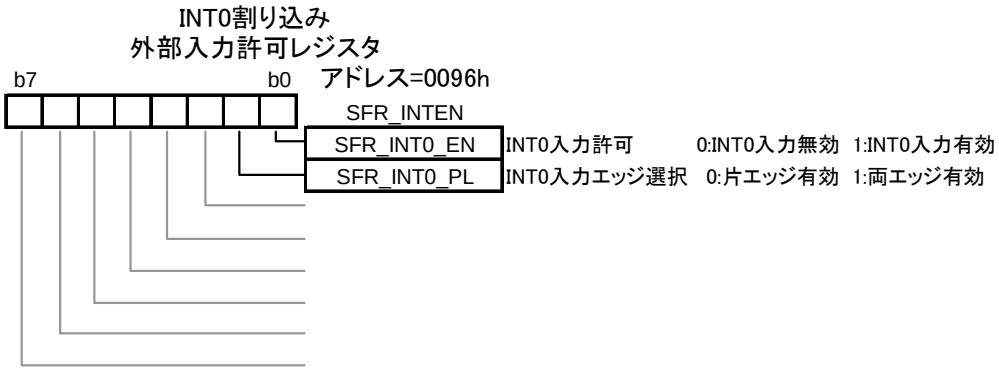
6.外部割り込み端子

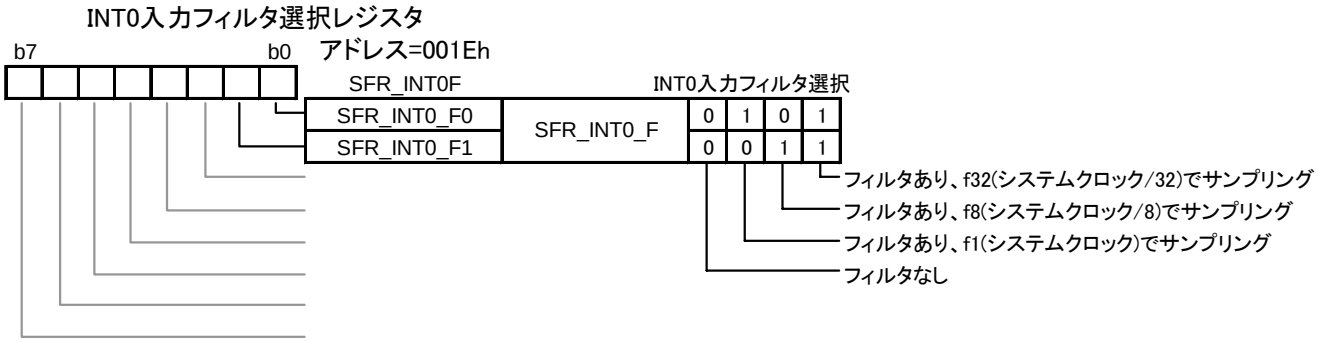
外部割込み入力は、外部信号により割り込みを起動します。
外部割込み入力には、ＩＮＴ０割込み、ＩＮＴ１割込み、ＩＮＴ３割り込み、キー入力割込みが使用できます。

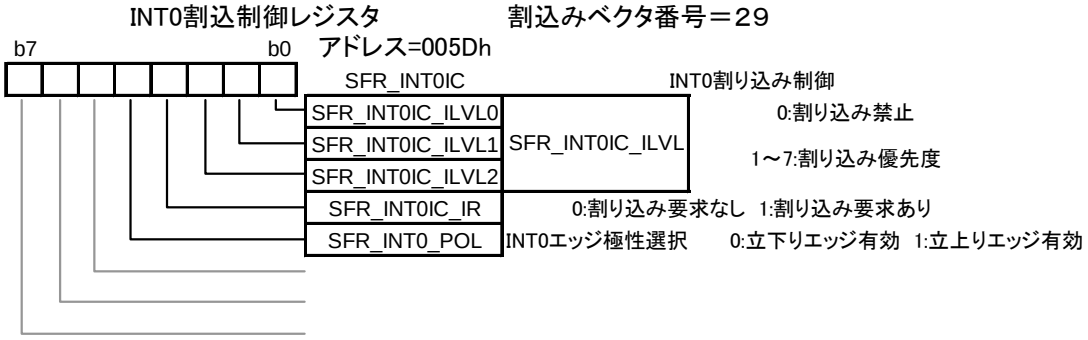
6-1 ＩＮＴ０割込み

ＩＮＴ０割り込みはＩＮＴ０割り込みを有効に設定すると、INT0-端子(P4_5 端子)信号の立上りエッジ、立下りエッジ、または立上り立下り両エッジで発生します。
信号ノイズ対策のデジタルフィルタが選択できます。デジタルフィルタを使用すると入力信号の短いグリッジやバウンズなどを除去することができます。デジタルフィルタは、指定したサンプリングクロックで３回のレベルが一致したとき割り込みが発生します。従って、デジタルフィルタを選択すると信号のエッジではなくサンプリング時間の３倍でレベル検出をすることになります（従って信号から割り込みまで遅延します)。サンプリング時間はせいぜい 50uS 程度なので、スイッチやリレーなどのメカニカルなチャタリング防止には利用できません。

INT0-端子はタイマＺトリガ入力にも使用されます。







ＩＮＴ０割り込みを有効にするには、ＩＮＴ０割込みレベル(SFR_INT0IC_ILVL)を 0 以外に設定して、SFR_INT0_EN を“1”にして、割り込みを許可（ei()または_asm(“ fset I”)）します。

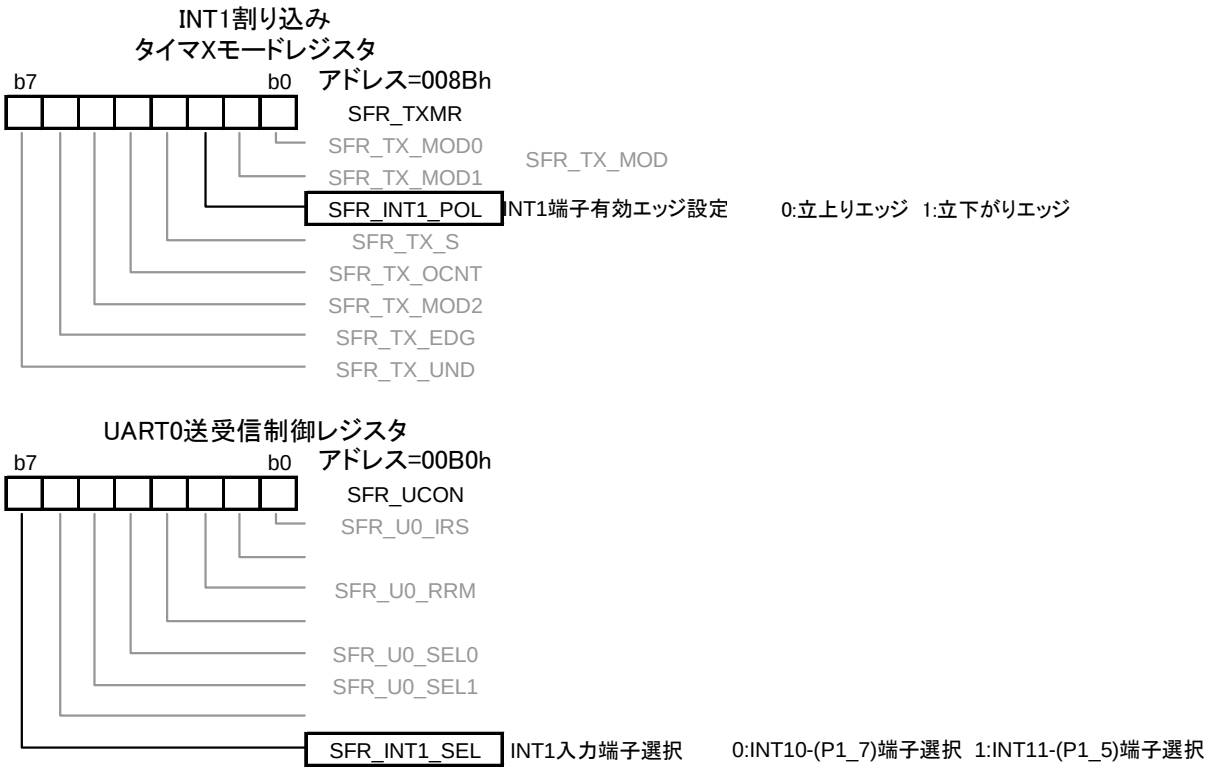
記述例

```
// ＩＮＴ０割込みを立下りエッジで発生する。
void int0_init(void)
{
//      SFR_P4D_5=0;            // P4_5(INT0-)端子入力設定（この行は省略可能）
//      SFR_INT0F=0;            // デジタルフィルタ無効（この行は省略可能）
        SFR_INTEN=0b00000001;   // INT0 端子有効 片エッジ選択
        SFR_INT0IC=7;           // INT0 割込みレベル７ 立下りエッジ有効
}

// ＩＮＴ０割り込み INT0-端子信号立下り検出
void int0_int(void)
#pragma interrupt int0_int (vect=29)
{
        // INT0 割込み処理を記述

}

void main(void)
{
        int0_init();            // INT0 割込み初期化
        ei();                   // 割り込み許可

        // メイン処理を記述

}
```

6-2 ＩＮＴ１割込み

ＩＮＴ１割り込みは、有効入力端子を INT10-端子(P1_7)端子と INT11-端子(P1_5 端子)から選択できます。
ＩＮＴ１割り込みを有効に設定すると、INT1 有効端子信号の立上りエッジまたは、立下りエッジでＩＮＴ１割り込みが発生します。

INT1-端子はタイマＸ入力にも使用されます。



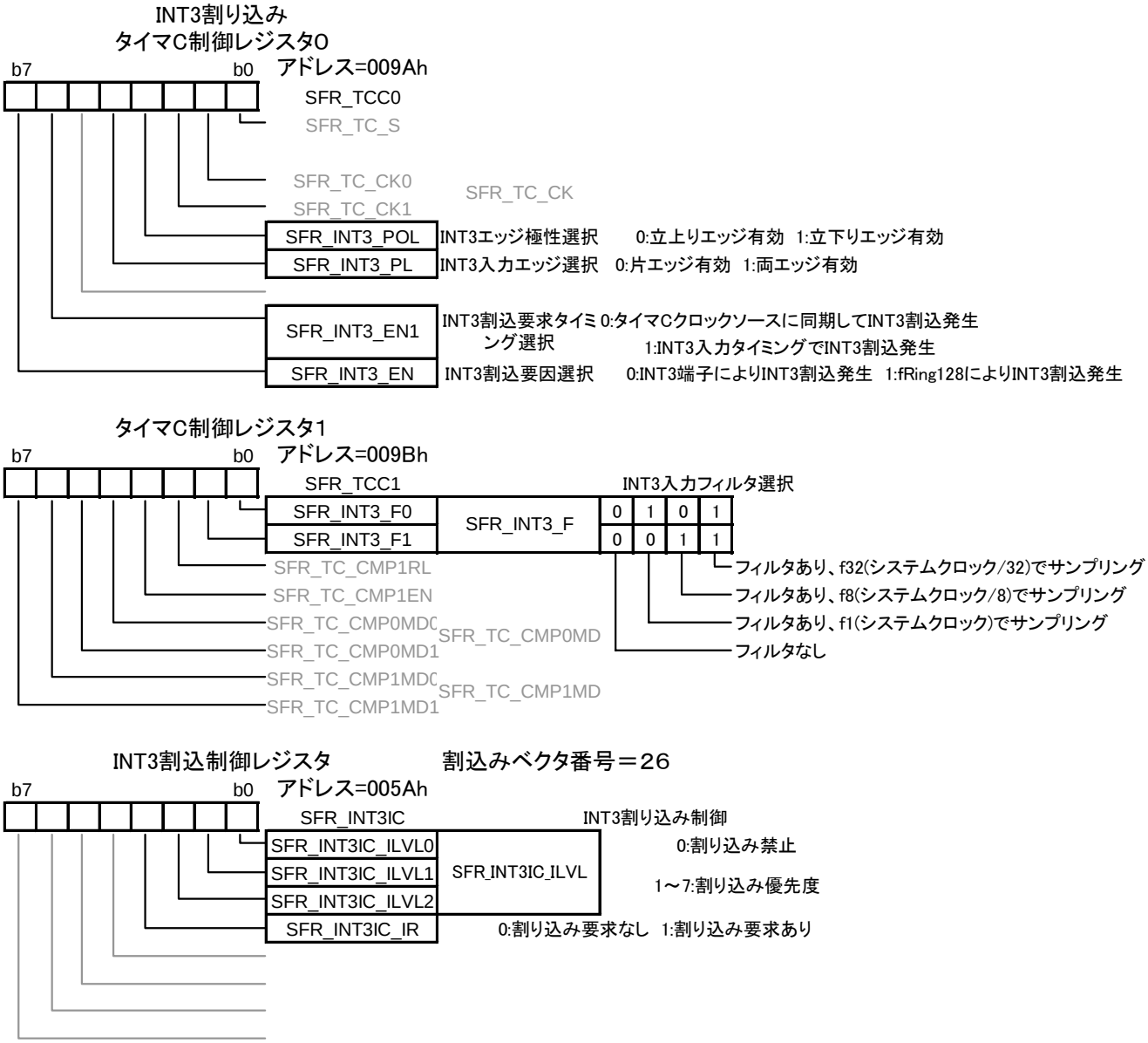
ＩＮＴ１割り込みを有効にするには、INT1 入力端子を選択して、ＩＮＴ１割込みレベル(SFR_INT1IC_ILVL)を 0 以外に設定して、割り込みを許可（ei()または_asm(“ fset I”)）します。

記述例

```
// ＩＮＴ１割込みを INT11-端子(P1_5)を選択して、立下りエッジで発生する。
void int1_init(void)
{
//      SFR_P1D_5=0;            // P1_5(INT11-)端子入力設定（この行は省略可能）
        SFR_INT1_SEL=1;         // INT11-端子(P1_5 端子)選択
        SFR_INT1_POL=1;         // INT1 立ち下がりエッジ有効
        SFR_INT1IC=7;           // INT1 割込みレベル７
}

// ＩＮＴ１割り込み INT11-端子信号立下り検出
void int1_int(void)
#pragma interrupt int1_int (vect=25)
{
        // INT1 割込み処理を記述

}

void main(void)
{
        int1_init();            // INT0 割込み初期化
        ei();                   // 割り込み許可

        // メイン処理を記述

}
```

6-3 ＩＮＴ３割込み

ＩＮＴ３割り込みはＩＮＴ３割り込みを有効に設定すると、INT3-端子(P3_3 端子)信号の立上りエッジ、立下りエッジ、または立上り立下り両エッジで発生します。
信号ノイズ対策のデジタルフィルタが選択できます。デジタルフィルタを使用すると入力信号の短いグリッジやバウンズなどを除去することができます。デジタルフィルタは、指定したサンプリングクロックで３回のレベルが一致したとき割り込みが発生します。従って、デジタルフィルタを選択すると信号のエッジではなくサンプリング時間の３倍でレベル検出をすることになります（従って信号から割り込みまで遅延します)。サンプリング時間はせいぜい 50uS 程度なので、スイッチやリレーなどのメカニカルなチャタリング防止には利用できません。
fRing（低速または高速内蔵オシレータ発振）の 128 分周の時間で定期的にＩＮＴ３割り込みを発生することも可能です。
INT3-端子はタイマＣ入力にも使用されます。



INT3-端子によるＩＮＴ３割り込みを有効にするには、SFR_INT3_EN を“０”に、SFR_INT3_EN1 を“１”に設定し、ＩＮＴ３割込みレベル(SFR_INT0IC_ILVL)を 0 以外に設定して、割り込みを許可（ei()または_asm(“ fset I ”)）します。

記述例

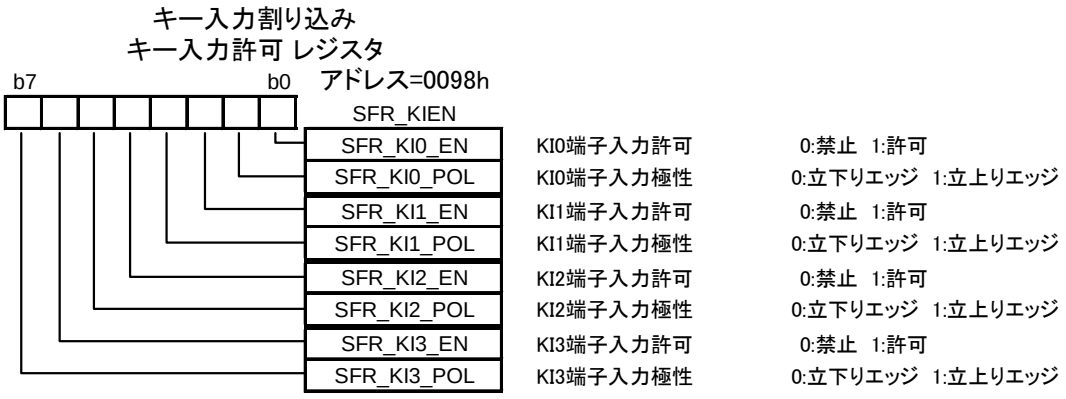
```
// ＩＮＴ３割込みを立上り立下り両エッジで発生する。
void int3_init(void)
{
//      SFR_P3D_3=0;            // P3_3(INT3-)端子入力設定（この行は省略可能）
//      SFR_INT3F=0;            // デジタルフィルタ無効（この行は省略可能）
        SFR_INT3_EN=0;          // INT3-端子有効
        SFR_INT3_EN1=1;         //
        SFR_INT3_POL=0;         // 両エッジの場合は“ 0 ”に設定
        SFR_INT3_PL=1;          // 両エッジ有効
        SFR_INT3IC=7;           // INT0 割込みレベル７ 立下りエッジ有効
}

// ＩＮＴ３割り込み INT3-端子信号立上り立下りエッジ検出
void int3_int(void)
#pragma interrupt int3_int (vect=26)
{
        // INT3 割込み処理を記述

}

void main(void)
{
        int3_init();            // INT3 割込み初期化
        ei();                   // 割り込み許可

        // メイン処理を記述

}
```

経験でのこと）

INT0/INT1/INT3 のいずれか２つの有効エッジが同時に入った場合、片方を検出しない場合がありました。このとき、インプットフィルターを f32 で入れて、検出にディレイをかけて取れるようにしたことがありました。

6-4 キー入力割込み

キー入力割り込みは、有効入力端子を KI0-端子(P1_0)端子、KI1-端子(P1_1)端子、KI2-端子(P1_2)端子、KI3-端子(P1_3)端子から同時に複数選択できます。
各端子有効を設定して、キー入力割り込みを有効に設定すると、有効各端子信号の立上りエッジまたは、立下りエッジでキー入力割り込みが発生します。
キー入力割り込みでは複数端子を有効としている場合、割込みがどの端子からかかったかは判断できません。
また、キー入力とありますが、実際にキースイッチなどをつけた場合にはチャタリング対策をソフトウェアまたは外付け回路で行う必用があります。



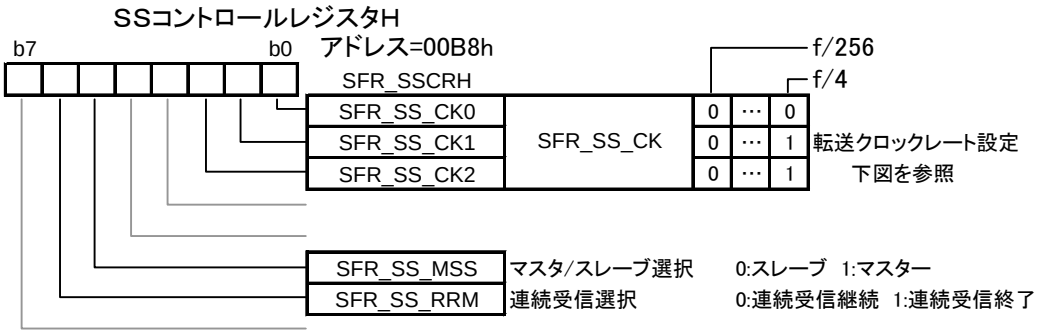
キー入力割り込みを使用する場合は、有効端子を許可して、キー入力割込みレベル(SFR_KUPIC_ILVL)を 0 以外に設定して、割り込みを許可（ei()または_asm(“ fset I”)）します。

記述例

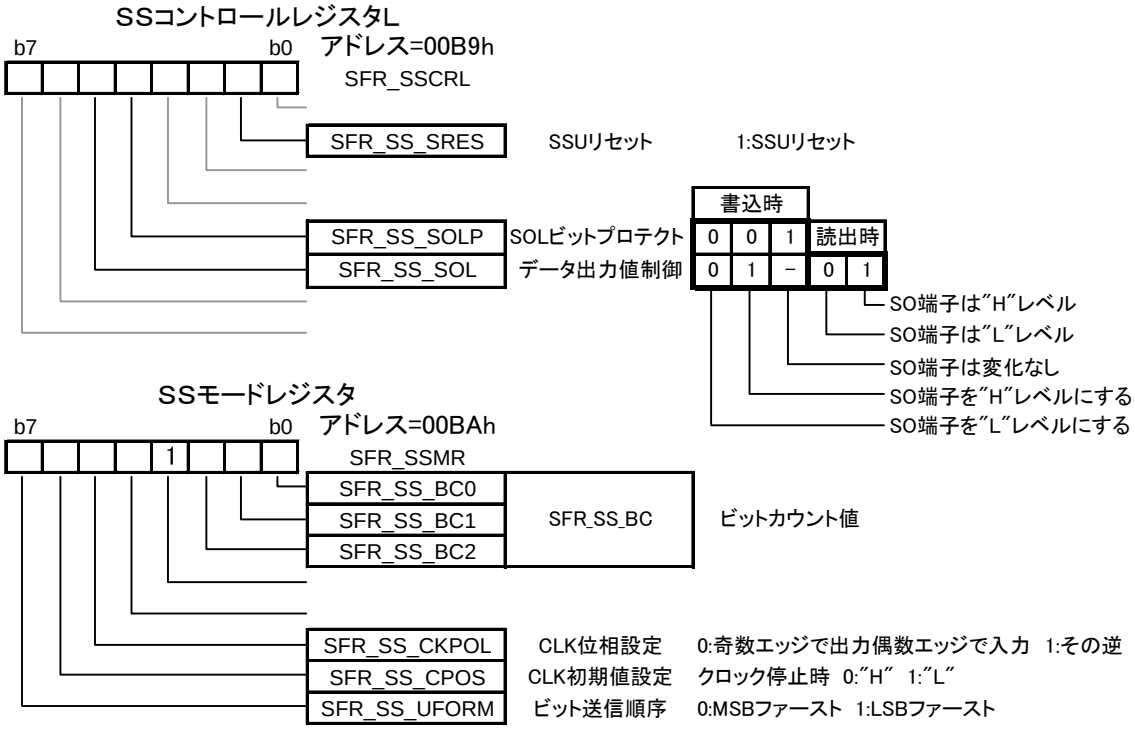
```
// キー入力割込みを使用して立下りエッジで発生する。
void key_init(void)
{
//      SFR_P1D &=0b11110000;     // P1_0(KI0-)端子～P1_3(KI3-)端子入力設定（この行は省略可能）
        SFR_KIEN=0b01010101;      // KI0-～KI3-端子割込み有効　全端子立下りエッジ選択
        SFR_KUPIC=2;              // キー入力割込みレベル２
}

// キー入力割り込み キー入力端子信号立下りエッジ検出
void key_int(void)
#pragma interrupt key_int (vect=13)
{
        // キー入力割込み処理を記述

}

void main(void)
{
        key_init();               // キー入力割込み初期化
        ei();                     // 割り込み許可

        // メイン処理を記述

}
```

７． チップセレクト付きクロック同期シリアルＩ／Ｏ（ＳＳＵ）

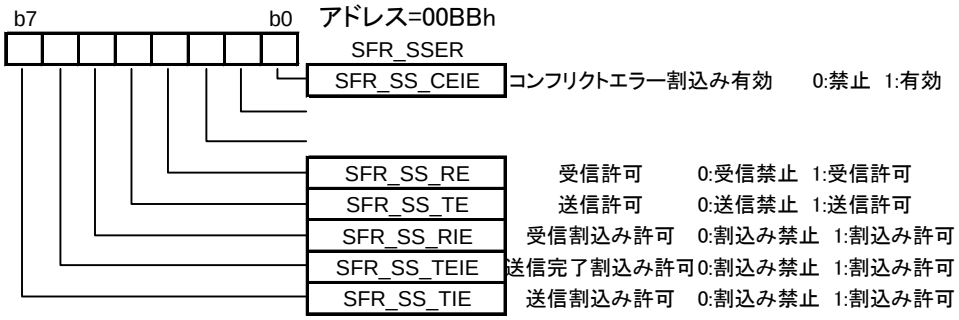
ＳＳＵはチップセレクト制御可能な同期クロックシリアルインターフェースです。一般的にＳＰＩと呼ばれるシリアルインターフェースを行います。



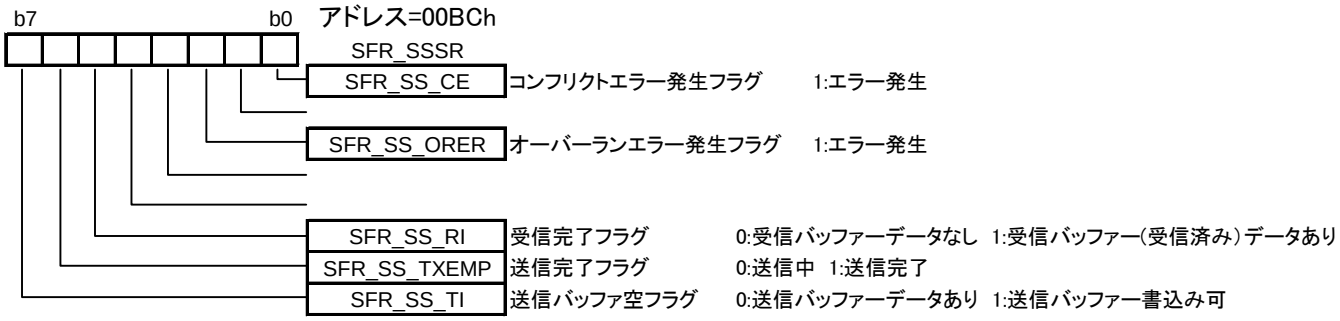
SFR_SS_CK2	SFR_SS_CK1	SFR_SS_CK0	転送レート(Hz) 分周比	20MHz時	8MHz時
0	0	0	f/256	78,125	31,250
0	0	1	f/128	156,250	62,500
0	1	0	f/64	312,500	125,000
0	1	1	f/32	625,000	250,000
1	0	0	f/16	1,250,000	500,000
1	0	1	f/8	2,500,000	1,000,000
1	1	0	f/4	5,000,000	2,000,000
1	1	1	–	–	–



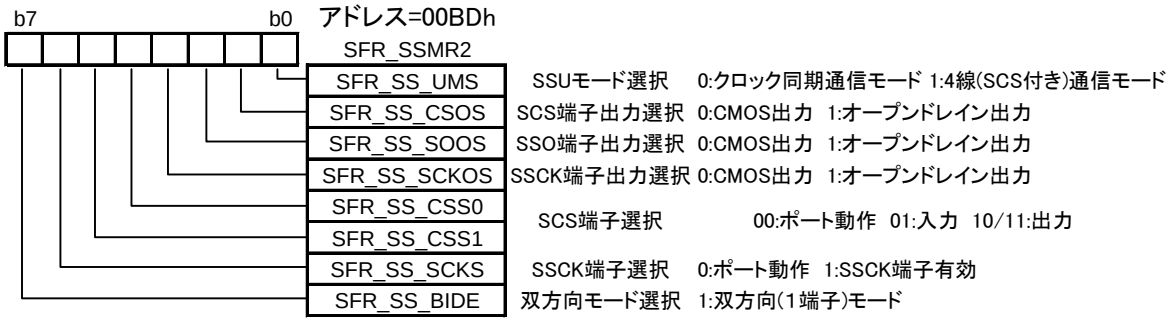
SSイネーブルレジスタ



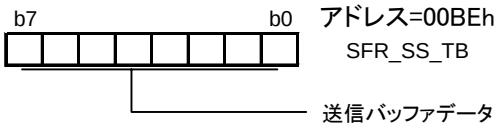
SSステータスレジスタ



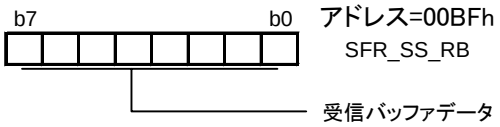
SSモードレジスタ2



SS送信データレジスタ



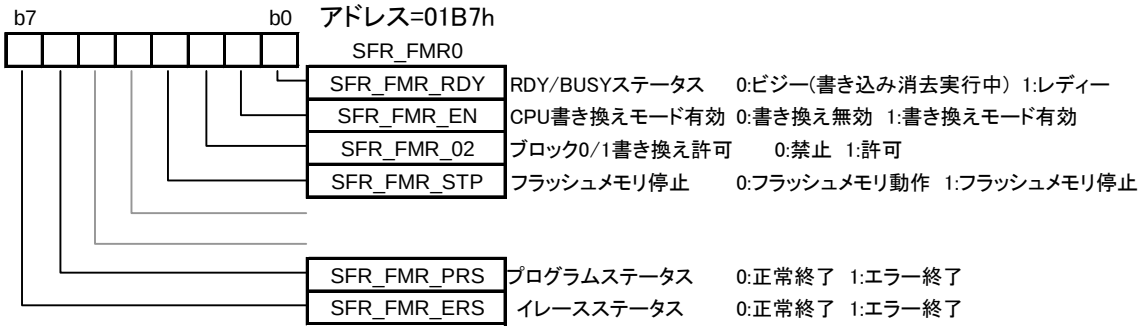
SS受信データレジスタ



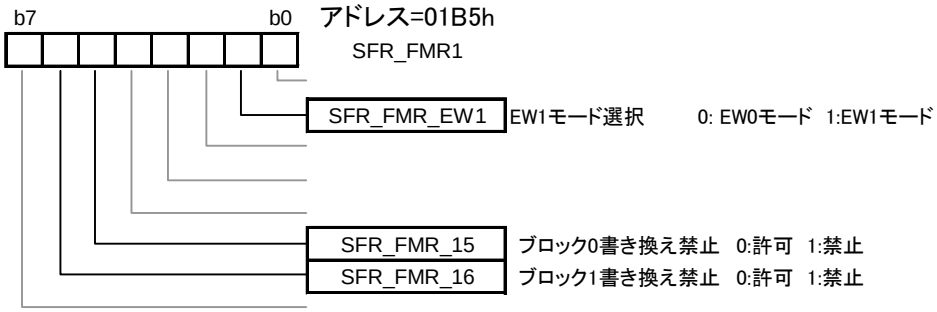
SSU割込制御レジスタ　　割込みベクタ番号＝15

8. フラッシュメモリ制御

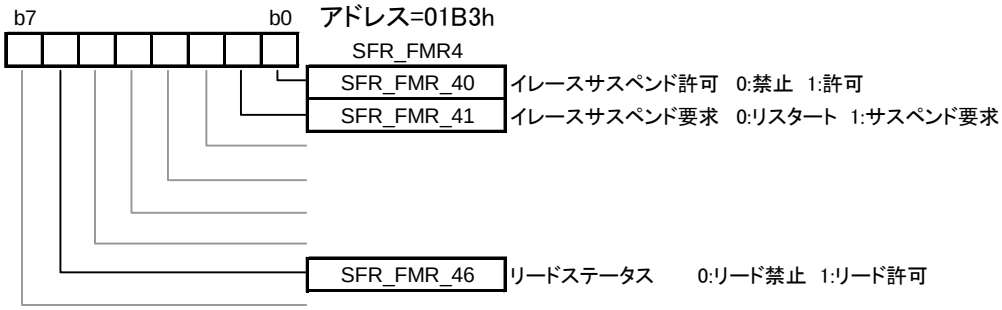
フラッシュメモリ制御レジスタ0



フラッシュメモリ制御レジスタ1



フラッシュメモリ制御レジスタ4

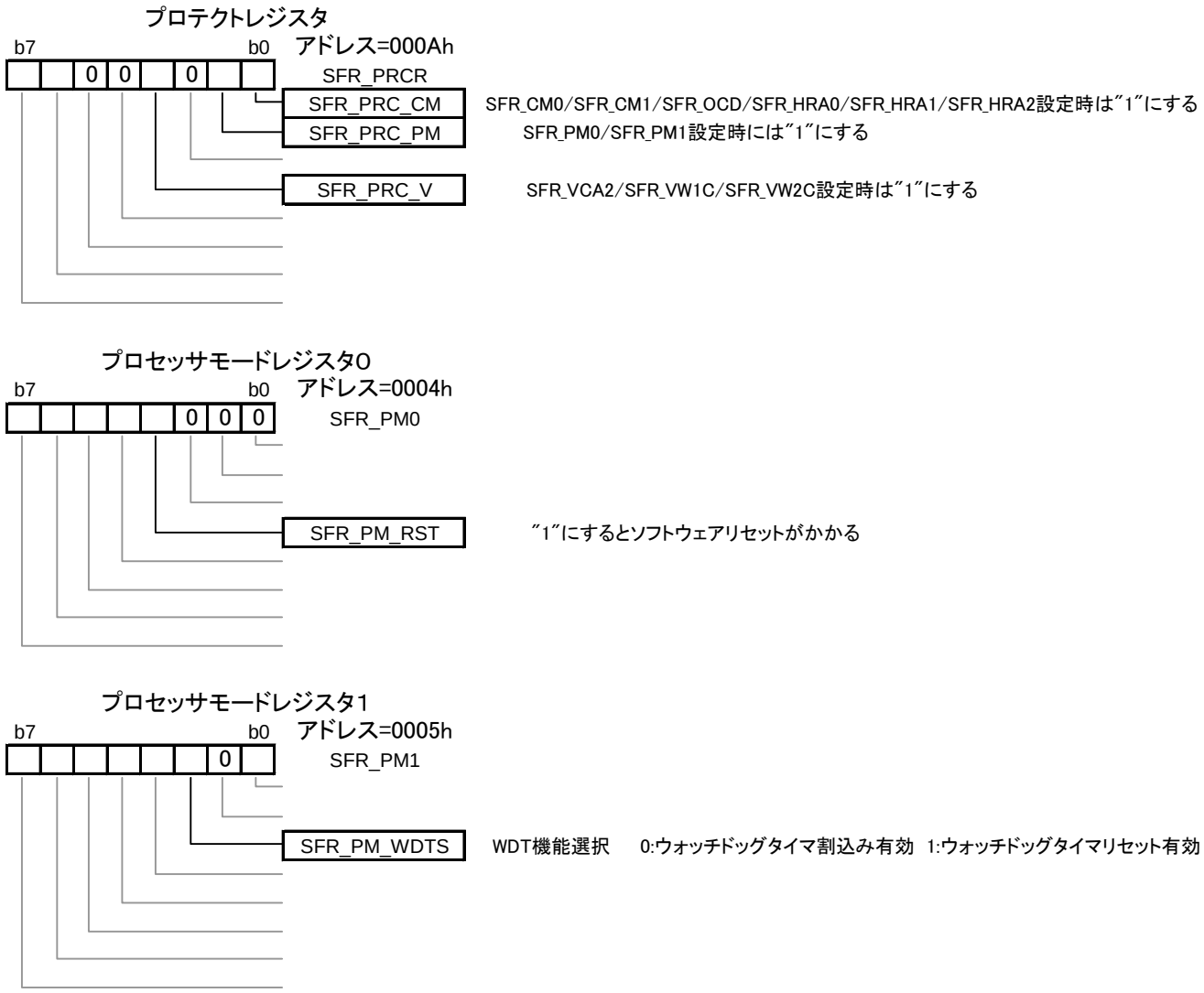


記述例

```
//
#define BLOCKA_TOP ((unsigned char *)0x2400)            // データフラッシュ BLOCK_A アドレス
#define BLOCKB_TOP ((unsigned char *)0x2800)            // データフラッシュ BLOCK_B アドレス

// フラッシュ ROM 書き込み 1 バイト
void write_flash(unsigned int adrs,unsigned char data)
{
        asm(" pushc flg");
        di();

        SFR_FMR_EN=0;
        SFR_FMR_EN=1;               // Flash 書き込み有効
        SFR_FMR_02=0;
        SFR_FMR_02=1;               // Block0/1 Enable
        SFR_FMR_EW1=0;
        SFR_FMR_EW1=1;              // EW1 モード設定

        *((unsigned char *)adrs)=0x40;                          // 書き込みコマンド
        *((unsigned char *)adrs)=data;
        while(SFR_FMR_RDY==0)
                        ;
        if (SFR_FMR_PRS)
                *((unsigned char *)adrs)=0x50;                  // クリアステータスコマンド
        SFR_FMR_EN=0;               // Flash 書き込み解除
        asm(" popc flg");
}

// フラッシュ ROM ブロック消去
void erase_flash(unsigned int adrs)
{
        asm(" pushc flg");
        di();

        SFR_FMR_EN=0;
        SFR_FMR_EN=1;               // Flash 書き込み有効
        SFR_FMR_02=0;
        SFR_FMR_02=1;               // Block0/1 Enable
        SFR_FMR_EW1=0;
        SFR_FMR_EW1=1;              // EW1 モード設定

        *((unsigned char *)adrs)=0x20;
        *((unsigned char *)adrs)=0xD0;                          // 消去コマンド
        while(SFR_FMR_RDY==0)
                ;
        if (SFR_FMR_ERS)
                *((unsigned char *)adrs)=0x50;                  // クリアステータスコマンド
        SFR_FMR_EN=0;               // Flash 書き込み解除

        asm(" popc flg");
}
```

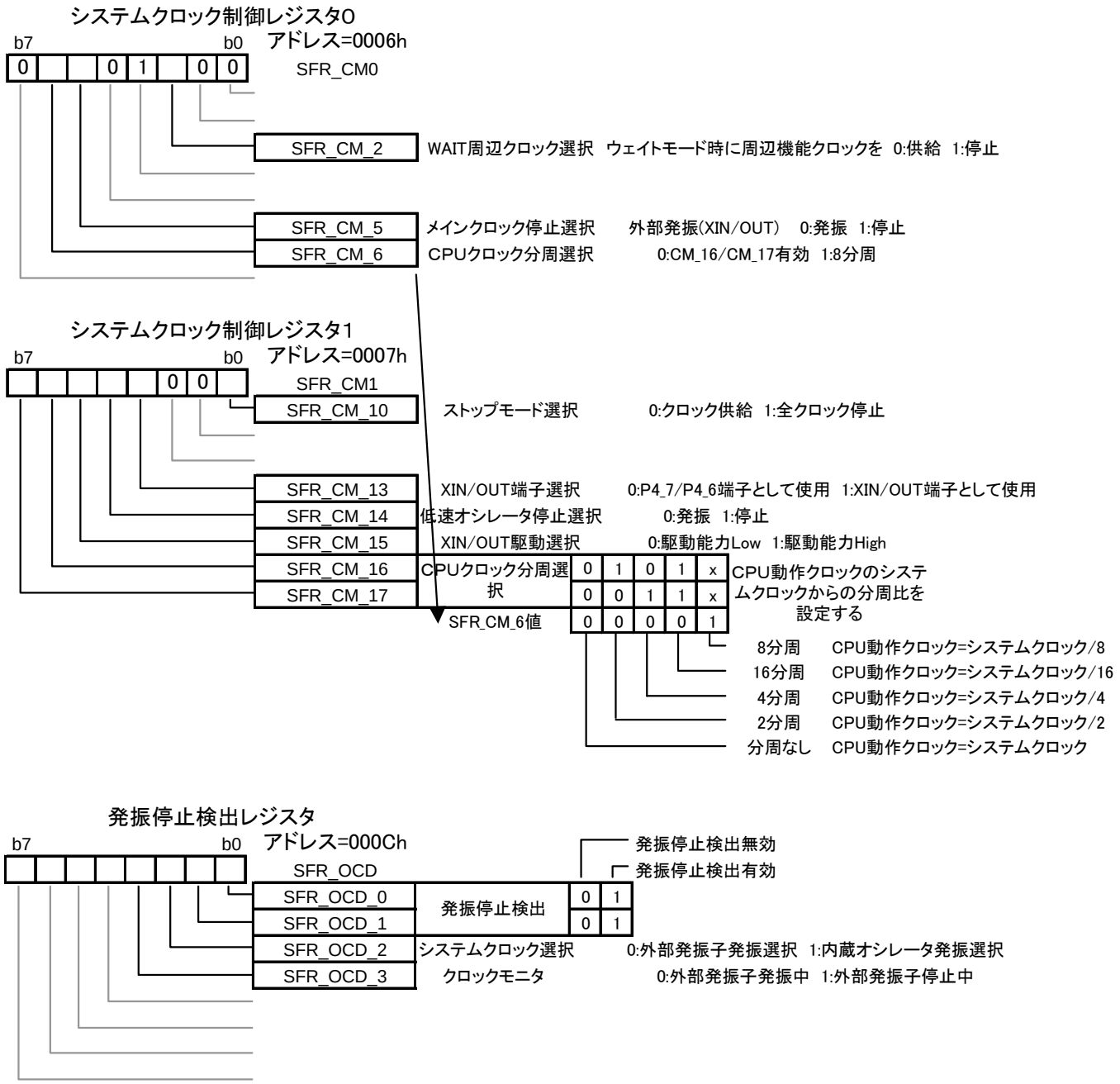
注意）消去書き込み実行中にデバッグでのノンマスカブル割り込みが入った場合は正常動作しないので、デバッグ時はフリーランで行ってください。

9. ＣＰＵ動作設定

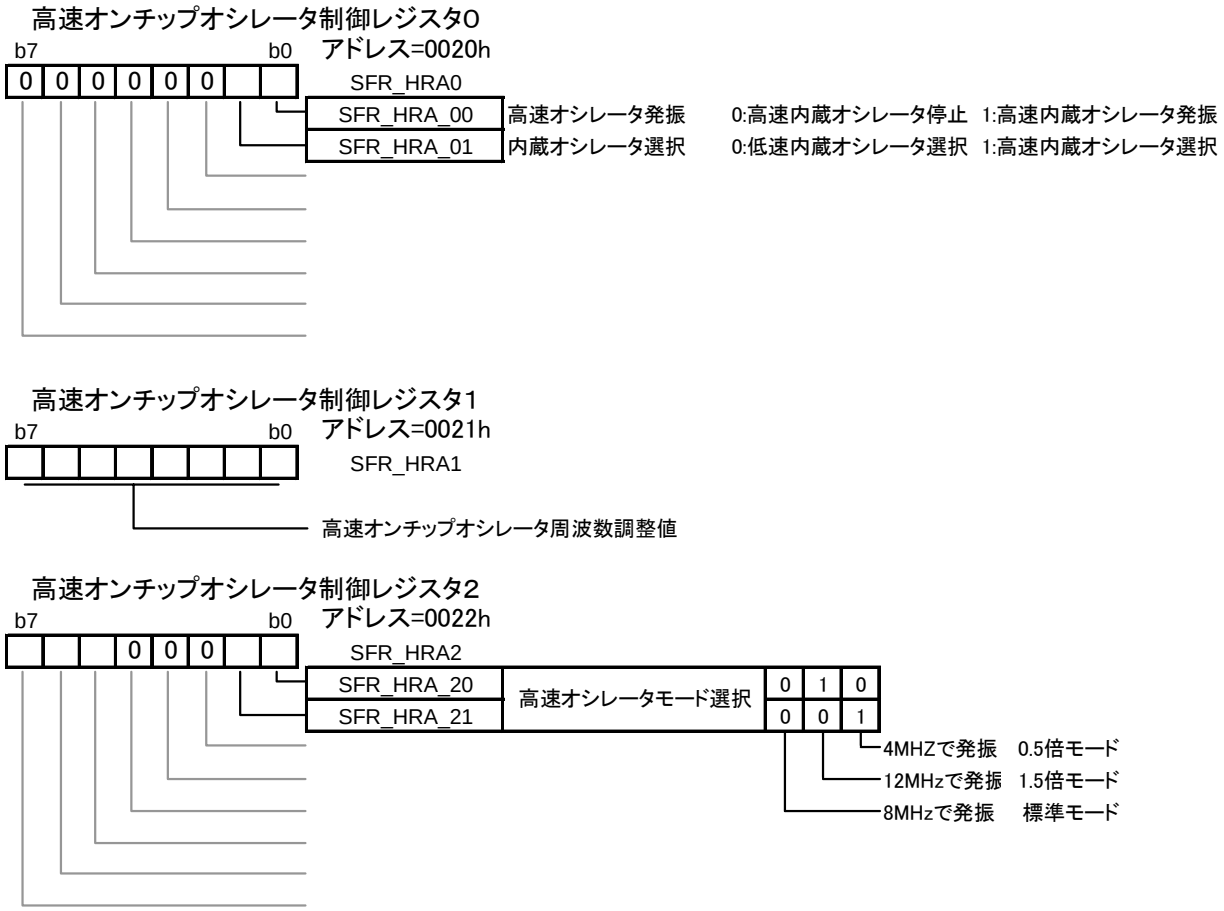


SFR_PM0/SFR_PM1 レジスタを設定する場合、SFR_PRC_PM=1 で PM レジスタのプロテクト解除を行ってから各レジスタを操作してください。
操作後に SFR_PRC_PM=0 を行ってください。

システムクロック設定関連レジスタ



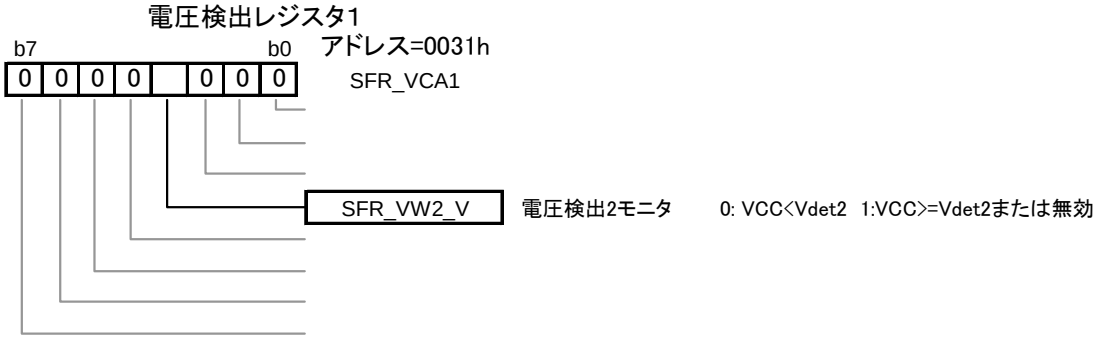
リセット直後は低速オシレータ発振で 8 分周の設定になっています。
外部発振子を使用する場合には、外部発振子を発振させ(SFR_CM_13=1 SFR_CM_15=1 SFR_CM_5=0)、発振安定を待ってからシステムクロックに外部発振子を選択(SFR_OCD_2=0)します。その後に、CPU クロック分周比を分周なし(SFR_CM_16=SFR_CM_17=0 SFR_CM_6=0)に設定します。

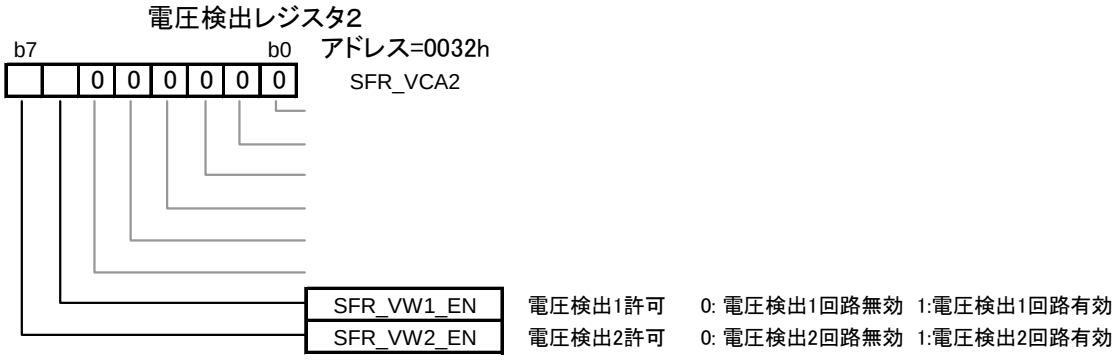


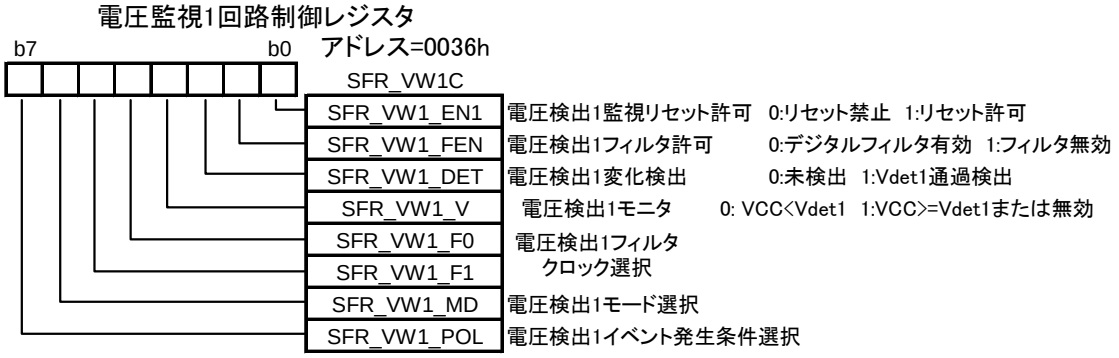
SFR_HRA1 の値は、出荷時に 8MHz に調整されています。従ってチップにより初期値は異なります。
この初期値（調整値）はブート ROM エリアに書かれているようです。

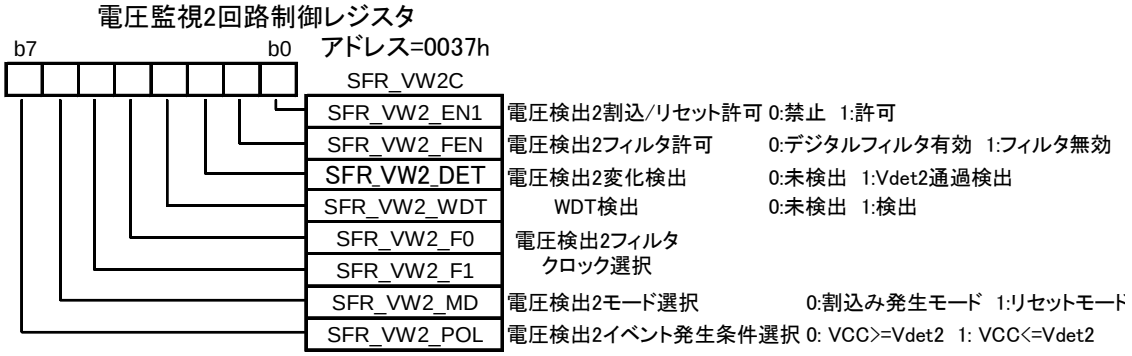
SFR_CM0/SFR_CM1/SFR_OCD/SFR_HRA0/SFR_HRA2 レジスタを設定する場合、SFR_PRC_CM=1 でレジスタのプロテクト解除を行ってから各レジスタを操作してください。
操作後に SFR_PRC_CM=0 を行ってください。

電圧監視関連レジスタ



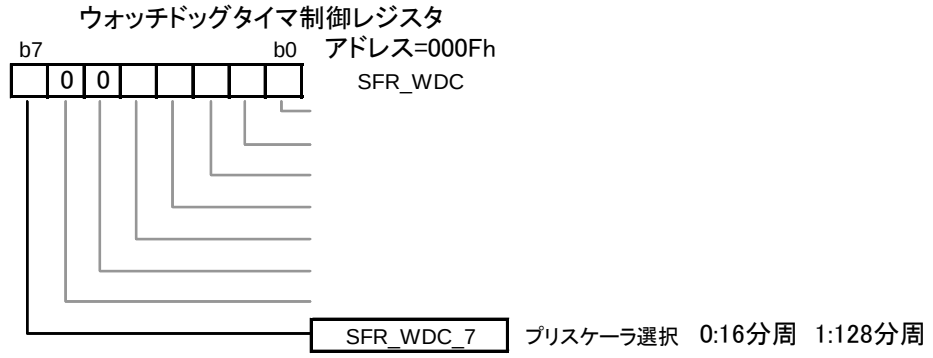


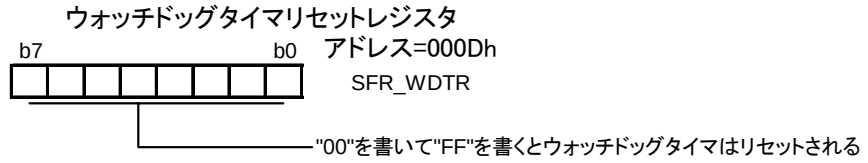


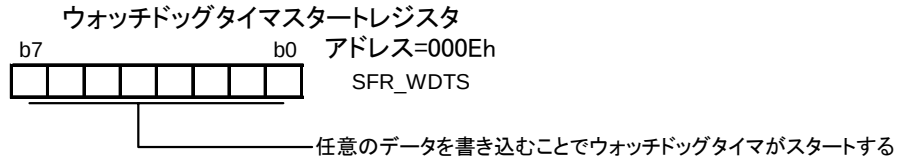


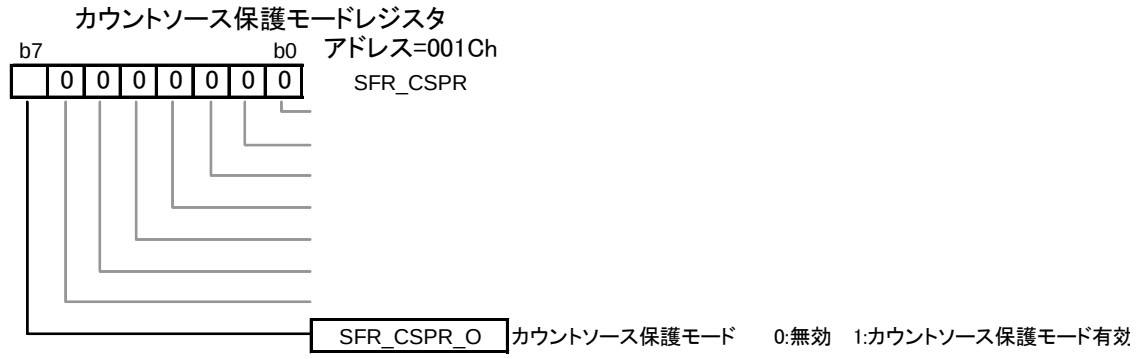
SFR_VCA2/SFR_VW1C/SFR_VW2C レジスタを設定する場合、SFR_PRC_V=1 でレジスタのプロテクト解除を行ってから各レジスタを操作してください。
操作後に SFR_PRC_V=0 を行ってください。

ウォッチドッグ関連レジスタ









オプション機能レジスタはフラッシュ ROM 最後尾のバイトです。
固定ベクタ中のリセットベクタの最上位(31-25 ビット)部分をうっかり“ 0 ”にしてしまうとウォッチドッグが起動してしまい、ウォッチドッグ前提のプログラミングをしていないと、正常動作しないので注意が必要です。