SH7450 グループ/SH7451 グループ
レジスタ定義ヘッダファイル

R01AN0190JJ0102
Rev.1.02
2010.09.01

要旨

この資料では、SH7450 グループ/SH7451 グループのレジスタ定義ヘッダファイルの記載内容と使用例を説明します。

動作確認デバイス

SH74504（R5F74504KBG）

SH74513（R5F74513KBG）

目次

1. レジスタ定義ヘッダファイルの説明

この資料は、以下のマイコンとコーディングツールの利用に適用します。

- マイコン　　：SH74504(R5F74504KBG)、SH74513(R5F74513KBG)
- コンパイラ　　：SH SERIES C/C++ Compiler V.9.03.02 (以下、SHC コンパイラと示す)
- MISRA C ルールチェッカ　：SQMlint V.1.03

1.1 レジスタ定義ヘッダファイルの構成

レジスタ定義ヘッダファイル(以下、SFR ヘッダファイルと示す)構成を、下記に示します。

(1) 製品ハードウェアマニュアルに記載のあるレジスタを定義。
(2) レジスタおよびビットのシンボル名は、製品ハードウェアマニュアル(Rev.1.00)に記載の名称を基本に使用。
(3) レジスタのアクセスサイズ(ビットフィールドアクセス含む)は、レジスタの最大サイズを基本に定義。
(4) レジスタのアドレスは、P4 領域(特権モード)のアドレスで定義。
(5) 8/16/32 ビットのアクセスサイズは各々、BYTE/WORD/LONG と示す。

ただし、SFR ヘッダファイルの使い勝手の向上のために、製品ハードウェアマニュアルに記載しているレジスタ名/ビット名以外に SFR ヘッダファイル専用に定義しているものがあります。詳細は「2.2　特殊なレジスタ定義方法」を参照してください。

1.2 MISRA C 対応について

この SFR ヘッダファイルは、MISRA C:1998 のルールに沿って記述しています。特徴と準拠について以下に示します。※MISRA ルールについては、SQMlint V.1.03 のメッセージと概要を示します。

(1) 特徴について

下記 MISRA C:1998 のルール 13 に準拠するために、符号無し 8/16/32 ビットのデータ型を_U1/_U2/_U4 と typedef で定義し使用しています。

・[MISRA C:1998 ルール 13(推奨)]
The basic types of char, int, short, long, float and double should not be used, but specific-length equivalents should be typedef'd for the specific compiler, and these type names used in the code.
(概要)char, int, long, float および double という基本のデータ型は使用せず、かわりに、個々のコンパイラに対してデータ長を typedef で定義することを推奨する。

(2) 準拠について

SFR ヘッダファイルでは、MISRA C:1998 での以下の 3 つのルール(ルール 110/111/45)に準拠していません。各ルールの概要と、準拠していない理由を以下に示します。

・[MISRA C:1998 ルール 110(必要)] Unions shall not be used to access the sub-parts of larger data types.
(概要)より大きなデータ型の一部分にアクセスするために共用体を使用してはならない。
(未準拠理由)レジスタを周辺機能毎に構造体で定義しており、ビットフィールドアクセスを含め可能なアクセスサイズで使用できる様に記載しているため、準拠していません。

・[MISRA C:1998 ルール 111(必要)]
Bit fields shall only be defined to be of type unsigned int or signed int.
(概要)ビットフィールドは, unsigned int 型又は signed int 型だけで定義しなければならない。
(未準拠理由)ルネサスエレクトロニクス製 SHC コンパイラは、ビットフィールドを定義したデータ型でレジスタをアクセスするコードを生成するため、準拠していません。

・[MISRA C:1998 ルール 45(必要)] Type casting from any type to or from pointers shall not be used.
(概要)全ての型からポインタへ、またはポインタからの全ての型へのキャストは使用してはならない。
(未準拠理由)レジスタは、周辺機能毎に構造体で定義しており、この構造体の先頭にアドレスを数値でマクロ定義(#define)しているため、準拠していません。

上記の準拠していない項目については、ルネサスエレクトロニクス製 SHC コンパイラでは、意図どおりのコードを生成することを確認していますので、問題ありません。

2. 使用方法の説明

SFR ヘッダファイルの基本的な使用方法と特殊な定義方法を使用している各レジスタについて説明します。

2.1 基本的な使用方法

SFR ヘッダファイルの基本的な使用方法を説明します。

2.1.1 基本様式

SFR ヘッダファイルを使用した基本的なレジスタのアクセスおよびビットフィールドのアクセスは、下記の様式となります。

(1) レジスタをアクセスする基本様式
　モジュール名(※1).レジスタシンボル名.アクセスサイズ(※2)

(2) ビットフィールドをアクセスする基本様式
　モジュール名(※1).レジスタシンボル名.BIT.ビットシンボル名

(※1)モジュール名　：define マクロで定義しています。
　詳細は、SFR ヘッダファイルの最後部にあります define 定義の部分をご確認ください。
(※2)アクセスサイズ　：レジスタサイズを基本に定義しています。
　BYTE or WORD or LONG で表記しています。
　詳細につきましては、製品ハードウェアマニュアルまたは、SFR ヘッダファイルの内容をご確認ください。

2.1.2 基本的な使用例

SFR ヘッダファイルの基本的な使用方法について TRAPA 例外レジスタ(TRA)を例に説明します。

TRA は、H'FF00 0020 番地に配置している 32 ビットのレジスタです。

TRAPA例外レジスタ（TRA）　　<P4領域アドレス：H'FF00 0020番地>

ビット：	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値：	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット：	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	TRACODE								—	—
リセット後の値：	0	0	0	0	0	0	不定	不定	不定	不定	不定	不定	不定	不定	0	0

<リセット後の値：不定>

ビット	シンボル	リセット後の値	R	W	説　明
31～10	—	すべて0	0	0	予約ビット 読み出すと常に"0"が読み出されます。書き込む値も常に"0"にしてください。
9～2	TRACODE	不定	R	W	TRAPAコード TRAPA命令の8ビットイミディエイトデータが設定されます。
1，0	—	全て0	0	0	予約ビット 読み出すと常に"0"が読み出されます。書き込む値も常に"0"にしてください。

(1) **構成内容の定義**

TRA は、例外処理レジスタの一部として struct st_exp 構造体でレジスタ定義しています。

```
/***********************************************************************************
 * Exception Handling (EXP)                                                        *
 ***********************************************************************************/
struct st_exp                                   /* struct exp          */
{                                               /*                     */
    union                                       /*                     */
    {                                           /* TRA                 */
        _U4 LONG;                               /*  Long Access        */
        struct                                  /*                     */
        {                                       /*  Bit Access         */
            _U4 reserved1:22;                   /*   Reserved Bits     */
            _U4 TRACODE:8;                      /*   TRACODE[9:2]      */
            _U4 reserved2:2;                    /*   Reserved Bits     */
        }BIT;                                   /*                     */
    }TRA;                                       /*                     */

    union                                       /*                     */
    {                                           /* EXPEVT              */
        _U4 LONG;                               /*  Long Access        */
        struct                                  /*                     */
        {                                       /*  Bit Access         */
            _U4 reserved1:20;                   /*   Reserved Bits     */
            _U4 EXPCODE:12;                     /*   EXPCODE[11:0]     */
        }BIT;                                   /*                     */
    }EXPEVT;                                    /*                     */

    union                                       /*                     */
    {                                           /* INTEVT              */
        _U4 LONG;                               /*  Long Access        */
        struct                                  /*                     */
        {                                       /*  Bit Access         */
            _U4 reserved1:18;                   /*   Reserved Bits     */
            _U4 INTCODE:14;                     /*   INTCODE[13:0]     */
        }BIT;                                   /*                     */
    }INTEVT;                                    /*                     */

    _U1 reserved1[0x2EFFD8];

    union                                       /*                     */
    {                                           /* EXPMASK             */
        _U4 LONG;                               /*  Long Access        */
        struct                                  /*                     */
        {                                       /*  Bit Access         */
            _U4 reserved1:27;                   /*   Reserved Bits     */
            _U4 MMCAW:1;                        /*   MMCAW[4:4]        */
            _U4 reserved2:3;                    /*   Reserved Bits     */
            _U4 RTEDS:1;                        /*   RTEDS[0:0]        */
        }BIT;                                   /*                     */
    }EXPMASK;                                   /*                     */

};
                                    :
                                    :
                                    :

/***********************************************************************************
 *  Defines                                                                        *
 ***********************************************************************************/
#define EXP         (*(volatile struct st_exp *)0xFF000020ul)       /* Exception Handling (EXP)              */
```

(2)　レジスタサイズのアクセス

TRA にレジスタサイズ(32 ビット)でアクセスする場合は、EXP.TRA.LONG を使用します。

例) TRA にレジスタサイズ(32 ビット)でアクセスする場合

```
EXP.TRA.LONG = 0x000003FCul;

unsigned long temp;
temp = EXP.TRA.LONG;
```

(3)　ビットフィールドのアクセス

TRA の TRACODE にビットフィールドでアクセスする場合は、EXP.TRA.BIT.TRACODE を使用します。

ルネサスエレクトロニクス製 SHC コンパイラは、ビットフィールドを定義したデータ型でレジスタをアクセスするコードを生成しますので、本ケースでは、32 ビットでレジスタアクセスするコードを生成します。ルネサスエレクトロニクス製 SHC コンパイラ以外のコンパイラでは、ビットフィールド定義のデータ型にかかわらず、8 ビットでレジスタをアクセスするコードを生成する場合がありますのでご注意ください。

例) TRA の TRACODE にビットフィールドでアクセスする場合

```
EXP.TRA.BIT.TRACODE = 0xFFul;

unsigned long temp;
temp = EXP.TRA.BIT.TRACODE;
```

2.1.3　ビットフィールドでアクセスする時の注意事項

ビットフィールドでアクセスして書き込みを行う場合、対象となるビットフィールド以外のフィールドには、読み出した値をそのまま書き戻します。フィールドの値を読み出してから書き戻すまでの間にレジスタの値が変化している場合は、意図しない変更を行う可能性があり、ビットフィールドのアクセスを使用してはいけない場合があります。

例えば、INTREQ レジスタの特定フラグを 0 クリアする場合、ビットフィールドのアクセスでは、0 クリアする対象以外のフラグには読み出した値をそのまま書き戻します。このため、読み出してから書き戻すまでの間に割り込み要求が発生してクリア対象以外のフラグの状態が変化した場合(レジスタを読み出す時にフラグは 0 であったが、書き戻す時にフラグが 1 に変化している場合)、その割り込み要求のフラグも 0 クリアすることになります。したがって、下記の例に示すようにレジスタサイズでアクセスして、クリア対象以外のフラグを 0 クリアしないようにしてください。

例)エッジ検出の場合の INTREQ レジスタの IRQ7 ビットをクリアする場合

```
temp = INTC.INTREQ.LONG;
if (temp & 0x01000000ul) {
    INTC.INTREQ.LONG = 0xFE000000ul;
}
```

製品のハードウェアマニュアルをご確認の上、ハードウェア制限事項に対して違反することがないよう十分に注意してご使用ください。

2.2　特殊なレジスタ定義方法

製品のハードウェアマニュアルに記載のレジスタ名/ビット名以外に、SFR ヘッダファイルで独自に定義しているものについて説明します。ご使用の際はご注意ください。

(1) ビットシンボル名の先頭に ”_” を付加するレジスタ

次のレジスタは、モジュール名(define マクロ名)の定義とビットシンボル名の重複を回避するため、ビットシンボル名の先頭に “_” (アンダースコア)を付加した形式で定義しています。

- 割り込み優先順位設定レジスタ 0～12(INT2PRI0～INT2PRI12)
- 割り込み要因レジスタ 00(INT2A00)
- 割り込み要因レジスタ 01(INT2A01)
- 割り込み要因レジスタ 10(INT2A10)
- 割り込み要因レジスタ 11(INT2A11)
- 割り込みマスクレジスタ 0(INT2MSKR)
- 割り込みマスクレジスタ 1(INT2MSKR1)
- 割り込みマスククリアレジスタ 0(INT2MSKCR)
- 割り込みマスククリアレジスタ 1(INT2MSKCR1)
- A/Di 変換値加算回数選択レジスタ(AD0ADC、AD1ADC)(i=0, 1)
- モジュールストップレジスタ 0(MSTPCR0)の一部のシンボル名(DRO、DRI0、DRI1、DRI2、PDAC)

例) 割り込み優先順位設定レジスタ 0(INT2PRI10)の定義

```
union                                   /*                      */
{                                       /* INT2PRI10            */
    _U4 LONG;                           /*  Long Access         */
    struct                              /*                      */
    {                                   /*  Bit Access          */
        _U4 reserved1:3;                /*   Reserved Bits      */
        _U4 _CAN0:5;                    /*   _CAN0[28:24]       */
        _U4 reserved2:3;                /*   Reserved Bits      */
        _U4 _CAN1:5;                    /*   _CAN1[20:16]       */
        _U4 reserved3:3;                /*   Reserved Bits      */
        _U4 _CAN2:5;                    /*   _CAN2[12:8]        */
        _U4 reserved4:3;                /*   Reserved Bits      */
        _U4 _CAN3:5;                    /*   _CAN3[4:0]         */
    }BIT;                               /*                      */
}INT2PRI10;                             /*                      */
```

(2) ビットシンボル名を定義しないレジスタ

次のレジスタは、タイマ TOU の動作モードによってビット数が変動するため、構造体によるビットフィールドを使用しない形式で定義しています。

- TOUnm カウンタ(TOnmCNT)
- TOUnm リロードレジスタ(TOnmRLD)(n=0～4、m=0～7)

例) TOU0 カウンタ(T000CNT)と TOU0 リロードレジスタ(T000RLD)の定義

```
union                                   /*                      */
{                                       /* T000CNT              */
    _U4 LONG;                           /*  Long Access         */
}T000CNT;                               /*                      */

union                                   /*                      */
{                                       /* T000RLD              */
    _U4 LONG;                           /*  Long Access         */
}T000RLD;                               /*                      */
```

次のレジスタは、右詰め/左詰めフォーマットがあり、また加算モード対応のチャネルについては、ビット数も変動するため構造体によるビットフィールドを使用しない形式で定義しています。

- A/D0 データレジスタ 0～15(AD0DR0～AD0DR15)
- A/D1 データレジスタ 0～7(AD1DR0～AD1DR7)
- A/D0 データレジスタ DIAG0(AD0DRD)
- A/D1 データレジスタ DIAG1(AD1DRD)

(3) SPiDR レジスタ

SPiDR レジスタへのリード/ライトは、RSPIi データコントロールレジスタ(SPiDCR)の RSPI ロングワードアクセス/ワードアクセス設定ビット(SPLW)の設定によってアクセスサイズが異なりますが、どちらのアクセスサイズにも対応できるようにワードアクセス用の構造体定義を追加しています。

- RSPIi データレジスタ(SPiDR)(i=0～2)

例) RSPI0 データレジスタ(SP0DR)の定義

```
  union                                  /*                      */
  {                                      /* SP0DR                */
      _U4 LONG;                          /*  Long Access         */
      struct                             /*                      */
      {                                  /*  Bit Access          */
          _U4 SPD31:1;                   /*   SPD31[31:31]       */
          _U4 SPD30:1;                   /*   SPD30[30:30]       */
          _U4 SPD29:1;                   /*   SPD29[29:29]       */
          _U4 SPD28:1;                   /*   SPD28[28:28]       */
          _U4 SPD27:1;                   /*   SPD27[27:27]       */
          _U4 SPD26:1;                   /*   SPD26[26:26]       */
          _U4 SPD25:1;                   /*   SPD25[25:25]       */
          _U4 SPD24:1;                   /*   SPD24[24:24]       */
          _U4 SPD23:1;                   /*   SPD23[23:23]       */
          _U4 SPD22:1;                   /*   SPD22[22:22]       */
          _U4 SPD21:1;                   /*   SPD21[21:21]       */
          _U4 SPD20:1;                   /*   SPD20[20:20]       */
          _U4 SPD19:1;                   /*   SPD19[19:19]       */
          _U4 SPD18:1;                   /*   SPD18[18:18]       */
          _U4 SPD17:1;                   /*   SPD17[17:17]       */
          _U4 SPD16:1;                   /*   SPD16[16:16]       */
          _U4 SPD15:1;                   /*   SPD15[15:15]       */
          _U4 SPD14:1;                   /*   SPD14[14:14]       */
          _U4 SPD13:1;                   /*   SPD13[13:13]       */
          _U4 SPD12:1;                   /*   SPD12[12:12]       */
          _U4 SPD11:1;                   /*   SPD11[11:11]       */
          _U4 SPD10:1;                   /*   SPD10[10:10]       */
          _U4 SPD9:1;                    /*   SPD9[9:9]          */
          _U4 SPD8:1;                    /*   SPD8[8:8]          */
          _U4 SPD7:1;                    /*   SPD7[7:7]          */
          _U4 SPD6:1;                    /*   SPD6[6:6]          */
          _U4 SPD5:1;                    /*   SPD5[5:5]          */
          _U4 SPD4:1;                    /*   SPD4[4:4]          */
          _U4 SPD3:1;                    /*   SPD3[3:3]          */
          _U4 SPD2:1;                    /*   SPD2[2:2]          */
          _U4 SPD1:1;                    /*   SPD1[1:1]          */
          _U4 SPD0:1;                    /*   SPD0[0:0]          */
      }BIT;                              /*                      */
      struct                             /*                      */
      {                                  /*  Bit Access          */
          _U2 SPD;                       /*   SPD[31:16]         */
          _U2 reserved1;                 /*   Reserved Bits      */
      }WORD;                             /*                      */
  }SP0DR;                                /*                      */
```

(4) CiMCTLj レジスタ

送信モード/受信モードでレジスタ内のビット役割と名称が異なるため、送受信の双方に対応できるよう、任意の構造体名でビットフィールドを定義しています。

- CANi メッセージ制御レジスタ j(CiMCTLj)(i=0～4、j=0～63)

例) CAN0 メッセージ制御レジスタ 0(COMCTL0)の定義

```
union                                   /*                      */
{                                       /* COMCTL0              */
    _U1 BYTE;                           /*  Byte Access         */
    struct                              /*                      */
    {                                   /*  Bit Access          */
        _U1 reserved1:1;                /*   Reserved Bits      */
        _U1 RECREQ:1;                   /*   RECREQ[6:6]        */
        _U1 reserved2:3;                /*   Reserved Bits      */
        _U1 MSGLOST:1;                  /*   MSGLOST[2:2]       */
        _U1 INVALDATA:1;                /*   INVALDATA[1:1]     */
        _U1 NEWDATA:1;                  /*   NEWDATA[0:0]       */
    }BIT;                               /*                      */
    struct                              /*                      */
    {                                   /*  Bit Access          */
        _U1 reserved1:1;                /*   Reserved Bits      */
        _U1 RECREQ:1;                   /*   RECREQ[6:6]        */
        _U1 reserved2:3;                /*   Reserved Bits      */
        _U1 MSGLOST:1;                  /*   MSGLOST[2:2]       */
        _U1 INVALDATA:1;                /*   INVALDATA[1:1]     */
        _U1 NEWDATA:1;                  /*   NEWDATA[0:0]       */
    }BIT_RECEIVE;                       /*                      */
}COMCTL0;                               /*                      */
```

例) CAN0 メッセージ制御レジスタ 32(COMCTL32)の定義

```
union                                   /*                      */
{                                       /* COMCTL32             */
    _U1 BYTE;                           /*  Byte Access         */
    struct                              /*                      */
    {                                   /*  Bit Access          */
        _U1 TRMREQ:1;                   /*   TRMREQ[7:7]        */
        _U1 RECREQ:1;                   /*   RECREQ[6:6]        */
        _U1 reserved1:1;                /*   Reserved Bits      */
        _U1 ONESHOT:1;                  /*   ONESHOT[4:4]       */
        _U1 reserved2:1;                /*   Reserved Bits      */
        _U1 TRMABT:1;                   /*   TRMABT[2:2]        */
        _U1 TRMACTIVE:1;                /*   TRMACTIVE[1:1]     */
        _U1 SENTDATA:1;                 /*   SENTDATA[0:0]      */
    }BIT_TRANSMIT;                      /*                      */
    struct                              /*                      */
    {                                   /*  Bit Access          */
        _U1 TRMREQ:1;                   /*   TRMREQ[7:7]        */
        _U1 RECREQ:1;                   /*   RECREQ[6:6]        */
        _U1 reserved1:1;                /*   Reserved Bits      */
        _U1 ONESHOT:1;                  /*   ONESHOT[4:4]       */
        _U1 reserved2:1;                /*   Reserved Bits      */
        _U1 MSGLOST:1;                  /*   MSGLOST[2:2]       */
        _U1 INVALDATA:1;                /*   INVALDATA[1:1]     */
        _U1 NEWDATA:1;                  /*   NEWDATA[0:0]       */
    }BIT_RECEIVE;                       /*                      */
}COMCTL32;                              /*                      */
```

(5) CiCLKR と CiBCR レジスタ

CiCLKR レジスタへのバイトアクセス、CiBCR レジスタと CiCLKR レジスタを結合したロングワードアクセスに対応できるよう、CiBCR レジスタ内の 1 バイトを CiCLKR レジスタとして定義しています。

- CANi クロック選択レジスタ(CiCLKR)
- CANi ビットコンフィグレーションレジスタ(CiBCR)(i=0～4)

例) CAN0 ビットコンフィグレーションレジスタ(C0BCR)の定義

```
  union                                  /*                      */
  {                                      /* C0BCR                */
      _U4 LONG;                          /*  Long Access         */
      struct                             /*                      */
      {                                  /*  Bit Access          */
          _U4 TSEG1:4;                   /*   TSEG1[31:28]       */
          _U4 reserved1:2;               /*   Reserved Bits      */
          _U4 BRP:10;                    /*   BRP[25:16]         */
          _U4 reserved2:2;               /*   Reserved Bits      */
          _U4 SJW:2;                     /*   SJW[13:12]         */
          _U4 reserved3:1;               /*   Reserved Bits      */
          _U4 TSEG2:3;                   /*   TSEG2[10:8]        */
          _U4 reserved4:7;               /*   Reserved Bits      */
          _U4 CCLKS:1;                   /*   CCLKS[0:0]         */
      }BIT;                              /*                      */
      struct                             /*                      */
      {                                  /*  Bit Access          */
          _U1 reserved1;                 /*   Reserved Bits      */
          _U1 reserved2;                 /*   Reserved Bits      */
          _U1 reserved3;                 /*   Reserved Bits      */
          _U1 C0CLKR;                    /*   C0LKR[7:0]         */
      }BYTE;                             /*                      */
  }C0BCR;                                /*                      */
```

(6) CiMBj レジスタ

CiMBj は、複数のレジスタを結合した構成でレジスタを定義しているため、メールボックスレジスタ内の各レジスタシンボルを個別の名称で定義しています。さらに CAN のチャネルごとに 64 メールボックス分を配列で定義しています。メールボックスのアクセスは、CiMBj の j と同じ番号の配列要素にアクセスしてください。例えば COMB0 の DLC にアクセスする場合は、CAN0.COMB[0].BIT.DLC を使用します。

- CANi メールボックスレジスタ j(CiMBj)(i=0～4、j=0～63)

例) CAN0 メールボックスレジスタ 0～63(COMB0～COMB63)の定義

```
union                                   /*                      */
{                                       /* COMB                 */
    _U1 cmbx[16];                       /*                      */
    struct                              /*                      */
    {                                   /*                      */
        _U4 CMBID;                      /*   CMBID              */
        _U4 CMB_1;                      /*   CMB_1              */
        _U4 CMB_2;                      /*   CMB_2              */
        _U4 CMB_3;                      /*   CMB_3              */
    }LONG;                              /*                      */
    struct                              /*                      */
    {                                   /*                      */
        _U2 CMBID_0;                    /*   CMBID_0            */
        _U2 CMBID_1;                    /*   CMBID_1            */
        _U2 CMBDLC;                     /*   CMBDLC             */
        _U2 DATA0;                      /*   DATA0              */
        _U2 DATA2;                      /*   DATA2              */
        _U2 DATA4;                      /*   DATA4              */
        _U2 DATA6;                      /*   DATA6              */
        _U2 TSP;                        /*   TSP                */
    }WORD;                              /*                      */
    struct                              /*                      */
    {                                   /*                      */
        _U1 MID0;                       /*   MID0               */
        _U1 MID1;                       /*   MID1               */
        _U1 MID2;                       /*   MID2               */
        _U1 MID3;                       /*   MID3               */
        _U1 RESERVED1;                  /*   RESERVED1          */
        _U1 DLC;                        /*   DLC                */
        _U1 DATA0;                      /*   DATA0              */
        _U1 DATA1;                      /*   DATA1              */
        _U1 DATA2;                      /*   DATA2              */
        _U1 DATA3;                      /*   DATA3              */
        _U1 DATA4;                      /*   DATA4              */
        _U1 DATA5;                      /*   DATA5              */
        _U1 DATA6;                      /*   DATA6              */
        _U1 DATA7;                      /*   DATA7              */
        _U1 TSH;                        /*   TSH                */
        _U1 TSL;                        /*   TSL                */
    }BYTE;                              /*                      */
    struct                              /*                      */
    {                                   /*                      */
        _U4 IDE:1;                      /*   IDE[31:31]         */
        _U4 RTR:1;                      /*   RTR[30:30]         */
        _U4 reserved1:1;                /*   Reserved Bits      */
        _U4 SID:11;                     /*   SID[28:18]         */
        _U4 EID:18;                     /*   EID[17:0]          */
        _U4 reserved2:12;               /*   Reserved Bits      */
        _U4 DLC:4;                      /*   DLC[19:16]         */
        _U4 DATA0:8;                    /*   DATA0[15:8]        */
        _U4 DATA1:8;                    /*   DATA1[7:0]         */
        _U4 DATA2:8;                    /*   DATA2[31:24]       */
        _U4 DATA3:8;                    /*   DATA3[23:16]       */
        _U4 DATA4:8;                    /*   DATA4[15:8]        */
        _U4 DATA5:8;                    /*   DATA5[7:0]         */
        _U4 DATA6:8;                    /*   DATA6[31:24]       */
        _U4 DATA7:8;                    /*   DATA7[23:16]       */
        _U4 TSH:8;                      /*   TSH[15:8]          */
        _U4 TSL:8;                      /*   TSL[7:0]           */
    }BIT;                               /*                      */
}COMB[64];                              /*                      */
```

(7) 配列で定義するレジスタ

次のレジスタは、連続するレジスタを配列で union 定義しています。対象レジスタの番号は、配列の要素数と一致しませんのでご注意ください。対象レジスタを参照する際はレジスタ番号-1 の配列要素にアクセスしてください。例えば PDIRTA1 にアクセスする場合は、PDA.PDIRTA[0].BYTE を使用します。

- PDAC 変調 A 立ち上がり出力時間レジスタ 1～120(PDIRTA1～PDIRTA120)
- PDAC 変調 A 立ち下がり出力時間レジスタ 1～120(PDIFTA1～PDIFTA120)
- PDAC 変調 B 立ち上がり出力時間レジスタ 1～200(PDIRTB1～PDIRTB200)
- PDAC 変調 B 立ち下がり出力時間レジスタ 1～200(PDIFTB1～PDIFTB200)
- PDAC 変調 C 立ち上がり出力時間レジスタ 1～600(PDIRTC1～PDIRTC600)
- PDAC 変調 C 立ち下がり出力時間レジスタ 1～600(PDIFTC1～PDIFTC600)

例) PDAC 変調 A 立ち上がり出力時間レジスタ 1～120(PDIRTA1～120)の定義

```
union                                   /*                      */
{                                       /* PDIRTA               */
    _U1 BYTE;                           /*  Byte Access         */
    struct                              /*                      */
    {                                   /*  Bit Access          */
        _U1 RTA:8;                      /*   RTA[7:0]           */
    }BIT;                               /*                      */
}PDIRTA[120];                           /*                      */
```

- FlexRay データセクションライトレジスタ 1～64(FRWRDS1～FRWRDS64)
- FlexRay データセクションリードレジスタ 1～64(FRRDDS1～FRRDDS64)

配列で定義しています。対象レジスタの番号は、配列の要素数と一致しませんのでご注意ください。対象レジスタを参照する際はレジスタ番号-1 の配列要素にアクセスしてください。例えば FRWRDS1 にアクセスする場合は、FR.FRWRDS[0].LONG を使用します。

(8) ダミーアクセス領域

ダミーアクセス領域にバイト/ワード/ロングワードアクセスに対応するために、BYTE/WORD/LONG の名前で変数を定義しています。

- ダミーアクセス領域(DUMMYHPB1)　H'FFA0 0000～H'FFA0 0003 番地
- ダミーアクセス領域(DUMMYHPB0)　H'FFFF 5020～H'FFFF 5023 番地

例) ダミーアクセス領域(DUMMYHPB0 と DUMMYHPB1)の定義

```
union                                   /*                      */
{                                       /* DUMMYHPB1            */
    _U4 LONG;                           /*  Long Access         */
    _U2 WORD;                           /*  Word Access         */
    _U1 BYTE;                           /*  Byte Access         */
}DUMMYHPB1;                             /*                      */
```

および

```
union                                   /*                      */
{                                       /* DUMMYHPB0            */
    _U4 LONG;                           /*  Long Access         */
    _U2 WORD;                           /*  Word Access         */
    _U1 BYTE;                           /*  Byte Access         */
}DUMMYHPB0;                             /*                      */
```

各レジスタの特殊な定義内容詳細につきましては、SFR ヘッダファイルをご確認ください。

3. 参考ドキュメント

- **ハードウェアマニュアル**
  SH7450 グループ、SH7451 グループ　ハードウェアマニュアル Rev.1.00(RJJ09B0470-0100)
  (最新版はルネサス エレクトロニクスのホームページから入手してください)

- **ソフトウェアマニュアル**
  SH-4A 拡張機能ソフトウェアマニュアル(RJJ09B0235-0100)
  (最新版はルネサス エレクトロニクスのホームページから入手してください)

ホームページとサポート窓

- ルネサス エレクトロニクスホームページ
  http://japan.renesas.com/
- お問合せ先
  http://japan.renesas.com/inquiry

改訂記録	SH7450 グループ/SH7451 グループ レジスタ定義ヘッダファイル

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2009.10.26	–	初版発行
1.01	2010.07.09	–	SFR ヘッダファイル変更（MDCR レジスタ追加）
		–	英文 AN 改訂に伴うスタイル統一
1.02	2010.09.01	–	ドキュメントスタイルとページ番号の修正
		–	ドキュメント番号を「RJJ06B1111-0101」から 「R01AN0190JJ0102」に変更

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本文を参照してください。なお、本マニュアルの本文と異なる記載がある場合は、本文の記載が優先するものとします。

1. 未使用端子の処理

【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。

CMOS製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI周辺のノイズが印加され、LSI内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。

2. 電源投入時の処置

【注意】電源投入時は，製品の状態は不定です。

電源投入時には、LSIの内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。

外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。

同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。

3. リザーブアドレスのアクセス禁止

【注意】リザーブアドレスのアクセスを禁止します。

アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレスがあります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。

4. クロックについて

【注意】リセット時は、クロックが安定した後、リセットを解除してください。

プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。

リセット時､外部発振子(または外部発振回路)を用いたクロックで動作を開始するシステムでは､クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子(または外部発振回路)を用いたクロックに切り替える場合は､切り替え先のクロックが十分安定してから切り替えてください。

5. 製品間の相違について

【注意】型名の異なる製品に変更する場合は、事前に問題ないことをご確認下さい。

同じグループのマイコンでも型名が違うと、内部メモリ、レイアウトパターンの相違などにより、特性が異なる場合があります。型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
3. 当社製品を改造、改変、複製等しないでください。
4. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   標準水準： コンピュータ、OA機器、通信機器、計測機器、AV機器、家電、工作機械、パーソナル機器、産業用ロボット

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害などを生じさせないようお客様の責任において冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、詳細につきましては製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

注1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

注2. 本資料において使用されている「当社製品」とは、注1において定義された当社の開発、製造製品をいいます。

RENESAS

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口

http://www.renesas.com

※営業お問合せ窓口の住所・電話番号は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス販売株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル）　(03)5201-5307

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：http://japan.renesas.com/inquiry

Colophon 1.0