```
/************************************************************************/
/*                          IIC(Inter IC Bus)の制御                     */
/************************************************************************/
/* ※割り込みは不使用                                                   */

#include <machine.h>
#include <stdlib.h>
#include "typedefine.h"
#include "iodefine.h"

/************************************************************************/
/* ICCR1のCKS2～CKS0にセットするコード **********************************/
                                /* φ=20MHz          */
#define RATE_714    (0)         /* φ/ 28 = 714kHz  */
#define RATE_500    (1)         /* φ/ 40 = 500kHz  */
#define RATE_417    (2)         /* φ/ 48 = 417kHz  */
#define RATE_313    (3)         /* φ/ 64 = 313kHz  */
#define RATE_250    (4)         /* φ/ 80 = 250kHz  */
#define RATE_200    (5)         /* φ/100 = 200kHz  */
#define RATE_179    (6)         /* φ/112 = 179kHz  */
#define RATE_156    (7)         /* φ/128 = 156kHz  */
#define RATE_357    (8)         /* φ/ 56 = 357kHz  */
//#define RATE_250  (9)         /* φ/ 80 = 250kHz  */
#define RATE_208    (10)        /* φ/ 96 = 208kHz  */
//#define RATE_156  (11)        /* φ/128 = 156kHz  */
#define RATE_125    (12)        /* φ/160 = 125kHz  */
#define RATE_100    (13)        /* φ/200 = 100kHz  */
#define RATE_89     (14)        /* φ/224 = 89.3kHz */
#define RATE_78     (15)        /* φ/256 = 78.1kHz */

#define RATE_USE    RATE_250    /* 使用する転送速度 */

/************************************************************************/

#define READ_MODE   (0)
#define WRITE_MODE  (1)
#define TRYMAX_IIC  (3)

/************************************************************************/
/* ソフトウェアタイマの時間                                             */

#define WAIT_100mS (80000)              /* 約100mS */
#define WAIT_50mS  (40000)              /* 約 50mS */
#define WAIT_10mS  ( 8000)              /* 約 10mS */
#define WAIT_1mS   (  800)              /* 約  1mS */
#define WAIT_500uS (  400)              /* 約500uS */
#define WAIT_200uS (  160)              /* 約200uS */
#define WAIT_100uS (   80)              /* 約100uS */

/************************************************************************/
#pragma abs8(bySlaveID)
static BYTE bySlaveID;                  /* 相手（スレーブ）のアドレス */


#pragma section
/************************************************************************/
/*                          IICインタフェース初期化                     */
/************************************************************************/
/* 備考 : H8/3694では、P57,P56はICCR1のICE=1でSCL,SDA端子になる。       */

void Init_IIC2(void)
{
    /* IICリセット */
    IIC2.ICCR2.BIT.IICRST = 1;
    IIC2.ICCR2.BIT.IICRST = 0;
    /* 送受信モード, 転送速度設定 */
    IIC2.ICCR1.BYTE = 0x80 | RATE_USE;  /* ICE=1:バスイネーブル、RCVD=0:受信継続、MST=0,TRS=0:スレーブ受信モード、CKS=4:転送クロック(RATE_USE) */
    /* 転送モード設定 */
    IIC2.ICMR.BYTE = 0x30;              /* MLS=0:MSBファースト、WAIT=0:DATAとACK連続転送、BCWP=BC2～BC0の書き込み可、BC2～BC0:９ビット転送 */
    IIC2.ICMR.BIT.BCWP = 1;             /* BCWP=BC2～BC0の書き込み禁止 */
    /* 割り込みモード設定 */
    IIC2.ICIER.BYTE = 0x00;             /* TIE=0:送信割り込み禁止、TEIE=0:送信終了割り込み禁止、RIE=0:受信割り込み禁止、NAKIE=0:NAK受信割り込み禁止 */
                                        /* STIE=0:停止条件検出割り込み禁止、ACKE=0:アクノリッジを無視する */
    /* 全ステータスクリア */
    if(IIC2.ICSR.BYTE);                 /* ICSR読み込み */
    IIC2.ICSR.BYTE = 0x00;              /* ACKB=0,全ステータスビットクリア */
    /* 転送フォーマット選択 */
    IIC2.SAR.BYTE  = 0x00;              /* 自局アドレス=0x00、FS=0:IICフォーマット選択 */
}

```

```
/*****************************************************************************/
/*                          スレーブアドレスの設定                           */
/*****************************************************************************/

void SetSlaveAddr_IIC2(BYTE bySlaveAddr)
{
    /* （相手）スレーブアドレス保存 */
    bySlaveID = bySlaveAddr;
}

/*****************************************************************************/
/*                IICインタフェースによる送受信（割り込み不使用）            */
/*****************************************************************************/

/* スタートコンディション＋スレーブアドレス  ********************************/

static BOOL Start_IIC2(BYTE byFirst, BYTE byWrite)
{
    WORD    wWaitCnt;
    BOOL    bResult;
    BYTE    byAddr,byTry;

    /* 最初のみバスの開放を待つ（BUSYなら待つ） */
    if(byFirst != 0){
        byTry = 20;
        wWaitCnt = WAIT_10mS;
        while(IIC2.ICCR2.BIT.BBSY != 0){
            if(--wWaitCnt == 0){
                if(--byTry == 0){
                    return FALSE;
                }
                wWaitCnt = WAIT_10mS;
            }
        }
    }

    /* R/Wを加えたスレーブアドレス作成 */
    byAddr = bySlaveID;
    if(byWrite == READ_MODE)
        byAddr |= 0x01;
    /* スレーブアドレス送信 */
    bResult = FALSE;
    byTry = 0;
    do{
        /* マスター送信モード指定 */
        IIC2.ICCR1.BYTE = 0xB0 | RATE_USE;      /* ICE=1:バスイネーブル、RCVD=0:受信継続、MST=1,TRS=1:マスター送信モード、CKS=4:転送クロック(250kbps) */
        /* 開始条件発行 */
        IIC2.ICCR2.BYTE = 0xBD;                 /* BBSY=1,SCP=0:開始条件発行、SDAO=1,SDAOP=1:SDA出力制御なし、IICRST=0:IICリセット無効 */
        /* 送信エンプティを待つ */
        wWaitCnt = WAIT_500uS;
        while(IIC2.ICSR.BIT.TDRE == 0){
            if(--wWaitCnt == 0)
                return FALSE;
        }
        /* スレーブアドレス送信 */
        IIC2.ICDRT = byAddr;
        /* 送信終了を待つ */
        wWaitCnt = WAIT_500uS;
        while(IIC2.ICSR.BIT.TEND == 0){
            if(--wWaitCnt == 0)
                goto _RETRY;
        }
        /* ACK'0'受信をチェック */
        if(IIC2.ICIER.BIT.ACKBR == 0){
            bResult = TRUE;
            break;
        }

_RETRY:
        wWaitCnt = WAIT_1mS;
        while(--wWaitCnt != 0);
    }while(++byTry < 10);

    return bResult;
}

/* ストップコンディション  ***************************************************/
/* 備考: RTS命令で10ステートかかるので、RTC-8564NBのtBUF(1.3uS)は確保される。 */

static BOOL Stop_IIC2(void)
{
    WORD    wWaitCnt;
```

```
163     BOOL    bResult;
164
165     /* 停止条件検出フラグクリア */
166     IIC2.ICSR.BIT.STOP = 0;
167     /* 停止条件発行 */
168     IIC2.ICCR2.BYTE = 0x3D;                         /* BBSY=0,SCP=0:停止条件発行、SDAO=1,SDAOP=1:SDA出力制御
    なし、IICRST=0:IICリセット無効 */
169     /* 停止条件の検出を待つ */
170     bResult = TRUE;
171     wWaitCnt = WAIT_500uS;
172     while(IIC2.ICSR.BIT.STOP == 0){
173         if(--wWaitCnt == 0){
174             bResult = FALSE;
175             break;
176         }
177     }
178     /* 送信終了フラグをクリア */
179     IIC2.ICSR.BIT.TEND = 0;
180     /* スレーブ受信モードに設定 */
181     IIC2.ICCR1.BYTE = 0x80 | RATE_USE;          /* ICE=1:バスイネーブル、RCVD=0:受信継続、MST=0,TRS=0:ス
    レーブ受信モード、CKS=4:転送クロック(250Kbps) */
182
183     return bResult;
184 }
185
186
187 /* データブロック送信  ******************************************************/
188
189 static BOOL SendBlock_IIC2(WORD wSize, const BYTE *pbyData)
190 {
191     WORD    wWaitCnt;
192
193     while(wSize > 1){
194         /* データ送信 */
195         IIC2.ICDRT = *(pbyData++);
196         --wSize;
197         /* 送信エンプティを待つ */
198         wWaitCnt = WAIT_500uS;
199         while(IIC2.ICSR.BIT.TDRE == 0){
200             if(--wWaitCnt == 0)
201                 return FALSE;
202         }
203     }
204     /* 最終データ送信 */
205     IIC2.ICDRT = *pbyData;
206 //  --wSize;
207     /* 送信終了を待つ */
208     wWaitCnt = WAIT_500uS;
209     while(IIC2.ICSR.BIT.TEND == 0){
210         if(--wWaitCnt == 0)
211             return FALSE;
212     }
213     /* ACK'0'受信をチェック */
214     if(IIC2.ICIER.BIT.ACKBR == 1){
215         return FALSE;
216     }
217
218     return TRUE;
219 }
220
221 /*  データブロック受信  ******************************************************/
222
223 static BOOL RecvBlock_IIC2(WORD wSize, BYTE *pbyBuff)
224 {
225     WORD    wWaitCnt;
226     BYTE    byDummy;
227     BYTE    byCCR;
228
229     /* 0バイト受信は無効 */
230     if(wSize == 0)  return FALSE;
231
232     /* 割り込みマスク */
233     byCCR = get_ccr();
234     set_imask_ccr(1);
235     /* トランスミットエンドフラグをクリア */
236     IIC2.ICSR.BIT.TEND = 0;
237     /* マスター受信モード指定 */
238     IIC2.ICCR1.BYTE = 0xA0 | RATE_USE;          /* ICE=1:バスイネーブル、RCVD=0:受信継続、MST=1,TRS=0:マ
    スター受信モード、CKS=4:転送クロック(250kbps) */
239     /* トランスミットデータエンプティフラグをクリア */
240     IIC2.ICSR.BIT.TDRE = 0;
241     /* 1バイトのみの受信では最初が最後となる */
242     if(wSize == 1){
243         /* ACKデータに'1'を設定 */
```

```
244         IIC2.ICIER.BIT.ACKBT = 1;
245         /* 受信ディセーブル */
246         IIC2.ICCR1.BIT.RCVD = 1;
247     }
248     else{
249         /* ACKデータに'0'を設定 */
250         IIC2.ICIER.BIT.ACKBT = 0;
251         /* 受信イネーブル */
252         IIC2.ICCR1.BIT.RCVD = 0;
253     }
254     /* 受信データ読み込み（最初のデータはダミー） */
255     byDummy = IIC2.ICDRR;
256     /* 割り込みマスク復帰 */
257     set_ccr(byCCR);
258 
259     /* 最初の受信データを待つ */
260     wWaitCnt = WAIT_1mS;
261     while(IIC2.ICSR.BIT.RDRF == 0){
262         if(--wWaitCnt == 0)
263             return FALSE;
264     }
265 
266     /* 最後以外の受信データを読み込み */
267     while(wSize > 1){
268         /* 最後－１番目の受信 */
269         if(wSize == 2){
270             /* ACKデータに'1'を設定 */
271             IIC2.ICIER.BIT.ACKBT = 1;
272             /* 受信ディセーブル */
273             IIC2.ICCR1.BIT.RCVD = 1;
274         }
275         /* 受信データ読み込み */
276         *(pbyBuff++) = IIC2.ICDRR;
277         --wSize;
278         /* データ受信を待つ */
279         wWaitCnt = WAIT_500uS;
280         while(IIC2.ICSR.BIT.RDRF == 0){
281             if(--wWaitCnt == 0)
282                 return FALSE;
283         }
284 
285     }
286 
287     /* 最後の受信データを読み込み */
288     *(pbyBuff++) = IIC2.ICDRR;
289 //  --wSize;
290 
291     return TRUE;
292 }
293 
294 
295 /*****************************************************************************/
296 /*                              １バイトデータリード                         */
297 /*****************************************************************************/
298 
299 /* １バイトのアドレス指定（RTC用） ******************************************/
300 
301 BOOL ReadByte_A8_IIC2(BYTE byRdAddr, BYTE *pRxBuff)
302 {
303     int     nTryCntr;
304     BOOL    bResult;
305 
306     bResult = FALSE;
307     nTryCntr = 0;
308     do{
309         /* スタートコンディション */
310         if(Start_IIC2(1, WRITE_MODE)){
311             /* ターゲットアドレス送信 */
312             if(SendBlock_IIC2(1, &byRdAddr)){
313                 /* スタートコンディション再開 */
314                 if(Start_IIC2(0, READ_MODE)){
315                     /* データ受信 */
316                     if(RecvBlock_IIC2(1, pRxBuff)){
317                         bResult = TRUE;
318                         break;
319                     }
320                 }
321             }
322         }
323     }while(++nTryCntr < TRYMAX_IIC);
324     /* ストップコンディション */
325     Stop_IIC2();
326 
327     return bResult;
```

```
}

/* ２バイトのアドレス指定（EEPROM用） ****************************************/

BOOL ReadByte_A16_IIC2(WORD wRdAddr, BYTE *pRxBuff)
{
    int     nTryCntr;
    BOOL    bResult;
    BYTE    byAddrH,byAddrL;

    /* 16bitアドレスを8bitずつにする */
    byAddrL = (BYTE)wRdAddr;
    byAddrH = (BYTE)(wRdAddr >> 8);
    /* 読み込みシーケンス */
    bResult = FALSE;
    nTryCntr = 0;
    do{
        /* スタートコンディション */
        if(Start_IIC2(1, WRITE_MODE)){
            /* ターゲットアドレス送信（High） */
            if(SendBlock_IIC2(1, &byAddrH)){
                /* ターゲットアドレス送信（Low） */
                if(SendBlock_IIC2(1, &byAddrL)){
                    /* スタートコンディション再開 */
                    if(Start_IIC2(0, READ_MODE)){
                        /* データ受信 */
                        if(RecvBlock_IIC2(1, pRxBuff)){
                            bResult = TRUE;
                            break;
                        }
                    }
                }
            }
        }
    }while(++nTryCntr < TRYMAX_IIC);
    /* ストップコンディション */
    Stop_IIC2();

    return bResult;
}

/*****************************************************************************/
/*                              ページデータリード                           */
/*****************************************************************************/

/* １バイトのアドレス指定（RTC用） ********************************************/

BOOL ReadPage_A8_IIC2(BYTE byRdAddr, WORD wSize, BYTE *pRxBuff)
{
    int     nTryCntr;
    BOOL    bResult;

    bResult = FALSE;
    nTryCntr = 0;
    do{
        /* スタートコンディション */
        if(Start_IIC2(1, WRITE_MODE)){
            /* ターゲットアドレス送信 */
            if(SendBlock_IIC2(1, &byRdAddr)){
                /* スタートコンディション再開 */
                if(Start_IIC2(0, READ_MODE)){
                    /* データ受信 */
                    if(RecvBlock_IIC2(wSize, pRxBuff)){
                        bResult = TRUE;
                        break;
                    }
                }
            }
        }
    }while(++nTryCntr < TRYMAX_IIC);
    /* ストップコンディション */
    Stop_IIC2();

    return bResult;
}

/* ２バイトのアドレス指定（EEPROM用） ****************************************/

BOOL ReadPage_A16_IIC2(WORD wRdAddr, WORD wSize, BYTE *pRxBuff)
{
    int     nTryCntr;
    BOOL    bResult;
    BYTE    byAddrH,byAddrL;

```

```
412     /* 16bitアドレスを8bitずつにする */
413     byAddrL = (BYTE)wRdAddr;
414     byAddrH = (BYTE)(wRdAddr >> 8);
415     /* 読み込みシーケンス */
416     bResult = FALSE;
417     nTryCntr = 0;
418     do{
419         /* スタートコンディション */
420         if(Start_IIC2(1, WRITE_MODE)){
421             /* ターゲットアドレス送信 (High)  */
422             if(SendBlock_IIC2(1, &byAddrH)){
423                 /* ターゲットアドレス送信 (Low)  */
424                 if(SendBlock_IIC2(1, &byAddrL)){
425                     /* スタートコンディション再開 */
426                     if(Start_IIC2(0, READ_MODE)){
427                         /* データ受信 */
428                         if(RecvBlock_IIC2(wSize, pRxBuff)){
429                             bResult = TRUE;
430                             break;
431                         }
432                     }
433                 }
434             }
435         }
436     }while(++nTryCntr < TRYMAX_IIC);
437     /* ストップコンディション */
438     Stop_IIC2();
439 
440     return bResult;
441 }
442 
443 /*****************************************************************************/
444 /*                              １バイトデータライト                         */
445 /*****************************************************************************/
446 
447 /* １バイトのアドレス指定 (RTC用)  ***************************************/
448 
449 BOOL WriteByte_A8_IIC2(BYTE byWrAddr, const BYTE *pRxBuff)
450 {
451     int     nTryCntr;
452     BOOL    bResult;
453 
454     bResult = FALSE;
455     nTryCntr = 0;
456     do{
457         /* スタートコンディション */
458         if(Start_IIC2(1, WRITE_MODE)){
459             /* ターゲットアドレス送信 */
460             if(SendBlock_IIC2(1, &byWrAddr)){
461                 /* データ送信 */
462                 if(SendBlock_IIC2(1, pRxBuff)){
463                     bResult = TRUE;
464                     break;
465                 }
466             }
467         }
468     }while(++nTryCntr < TRYMAX_IIC);
469     /* ストップコンディション */
470     Stop_IIC2();
471 
472     return bResult;
473 }
474 
475 /* ２バイトのアドレス指定 (EEPROM用)  ************************************/
476 
477 BOOL WriteByte_A16_IIC2(WORD wWrAddr, const BYTE *pRxBuff)
478 {
479     int     nTryCntr;
480     BOOL    bResult;
481     BYTE    byAddrH,byAddrL;
482 
483     /* 16bitアドレスを8bitずつにする */
484     byAddrL = (BYTE)wWrAddr;
485     byAddrH = (BYTE)(wWrAddr >> 8);
486     /* 書き込みシーケンス開始 */
487     bResult = FALSE;
488     nTryCntr = 0;
489     do{
490         /* スタートコンディション */
491         if(Start_IIC2(1, WRITE_MODE)){
492             /* ターゲットアドレス送信 (High)  */
493             if(SendBlock_IIC2(1, &byAddrH)){
494                 /* ターゲットアドレス送信 (Low)  */
495                 if(SendBlock_IIC2(1, &byAddrL)){
```

```
                    /* データ送信 */
                    if(SendBlock_IIC2(1, pRxBuff)){
                        bResult = TRUE;
                        break;
                    }
                }
            }
        }
    }while(++nTryCntr < TRYMAX_IIC);
    /* ストップコンディション */
    Stop_IIC2();

    return bResult;
}


/*****************************************************************************/
/*                          ページデータライト                               */
/*****************************************************************************/

/* １バイトのアドレス指定（RTC用） ******************************************/

BOOL WritePage_A8_IIC2(BYTE byWrAddr, WORD wSize, const BYTE *pRxBuff)
{
    int     nTryCntr;
    BOOL    bResult;

    bResult = FALSE;
    nTryCntr = 0;
    do{
        /* スタートコンディション */
        if(Start_IIC2(1, WRITE_MODE)){
            /* ターゲットアドレス送信 */
            if(SendBlock_IIC2(1, &byWrAddr)){
                /* データ送信 */
                if(SendBlock_IIC2(wSize, pRxBuff)){
                    bResult = TRUE;
                    break;
                }
            }
        }
    }while(++nTryCntr < TRYMAX_IIC);
    /* ストップコンディション */
    Stop_IIC2();

    return bResult;
}

/* ２バイトのアドレス指定（EEPROM用） ***************************************/

BOOL WritePage_A16_IIC2(WORD wWrAddr, WORD wSize, const BYTE *pRxBuff)
{
    int     nTryCntr;
    BOOL    bResult;
    BYTE    byAddrH,byAddrL;

    /* 16bitアドレスを8bitずつにする */
    byAddrL = (BYTE)wWrAddr;
    byAddrH = (BYTE)(wWrAddr >> 8);
    /* 書き込みシーケンス開始 */
    bResult = FALSE;
    nTryCntr = 0;
    do{
        /* スタートコンディション */
        if(Start_IIC2(1, WRITE_MODE)){
            /* ターゲットアドレス送信（High） */
            if(SendBlock_IIC2(1, &byAddrH)){
                /* ターゲットアドレス送信（Low） */
                if(SendBlock_IIC2(1, &byAddrL)){
                    /* データ送信 */
                    if(SendBlock_IIC2(wSize, pRxBuff)){
                        bResult = TRUE;
                        break;
                    }
                }
            }
        }
    }while(++nTryCntr < TRYMAX_IIC);
    /* ストップコンディション */
    Stop_IIC2();

    return bResult;
}

```