```
/*****************************************************************************/
/***                シリアルフラッシュメモリのアクセス制御                   ***/
/*****************************************************************************/

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


#pragma section
#pragma bit_order right

/*****************************************************************************/
/* 書き込みバッファ                                                          */

static BYTE abyWrBuff[FLWBUFFSZ];       /* 書き込み用バッファ */

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

#define FLASH_RD_DATA_CMD       (0x0B)      /* 5pass(High Speed) Read Command */
#define FLASH_WR_DATA_CMD       (0x02)      /* Byte/Page Write Command        */
#define FLASH_WR_AAI_CMD        (0xAD)      /* AutoAddressIncrement(AAI)Word-Program (SST only)☆ */
#define FLASH_RD_STAT_CMD       (0x05)      /* Status Register Read  Command  */
#define FLASH_WR_STAT_CMD       (0x01)      /* Status Register Write Command  */
#define FLASH_RD_ID_CMD         (0x9F)      /* JEDEC Device ID Read Command   */
#define FLASH_ERASE_4K_CMD      (0x20)      /*  4K bytes Block Erase Command ☆ */
#define FLASH_ERASE_32K_CMD     (0x52)      /* 32K bytes Block Erase Command ☆ */
#define FLASH_ERASE_64K_CMD     (0xD8)      /* 64K bytes Block Erase Command  */
#define FLASH_ERASE_CHIP_CMD    (0xC7)      /* Chip(Bulk) Erase Command       */
#define FLASH_WR_ENABLE_CMD     (0x06)      /* Write Enable  Command          */
#define FLASH_WR_DISABLE_CMD    (0x04)      /* Write Disable Command          */
/* ### ☆印はM25Pxx(Micron)にないコマンド。 ### */

/* シリアルフラッシュメモリのコマンド（１サイクルコマンドのみ） */
static const BYTE WrEnableCmd  = FLASH_WR_ENABLE_CMD;
static const BYTE WrDisableCmd = FLASH_WR_DISABLE_CMD;
static const BYTE ReadStatCmd  = FLASH_RD_STAT_CMD;
static const BYTE ReadIdCmd    = FLASH_RD_ID_CMD;
static const BYTE EraseChipCmd = FLASH_ERASE_CHIP_CMD;

/*****************************************************************************/
/*              シリアルフラッシュメモリのステータス読み込み                 */
/*****************************************************************************/
/* 引数 : wFlashNum = フラッシュメモリの選択（０のみ指定可能）               */
/*      : pbyStat = ステータス格納アドレス（ステータスは１バイト）           */
/* 戻値 : SubReadStatus_sFlash() ==  TRUE : 正常終了                         */
/*      :                        == FALSE : 異常終了                         */

static BOOL SubReadStatus_sFlash(WORD wFlashNum, BYTE *pbyStat)
{
    int     nTry;
    BOOL    bResult;

    /* コマンド送信、レスポンス受信 */
    nTry = 3;
    do{
        /* フラッシュメモリの選択とSPI通信開始 */
        Enable_SPI0(SPI_ENABLE, wFlashNum);
        /* ステータス読み込みコマンド送信 */
        bResult = Tranceive_SPI0(1, 1, &ReadStatCmd, pbyStat);
        /* SPI通信停止 */
        Enable_SPI0(SPI_DISABLE, 0);
        if(bResult) break;
    }while(--nTry != 0);

    return bResult;
}

/*****************************************************************************/
/*                シリアルフラッシュメモリのステータス書き込み               */
/*****************************************************************************/
/* 注意 : 自動的に書込み禁止状態（ステータスのb1=0）になる。                 */
/* 引数 : wFlashNum = フラッシュメモリの選択（０のみ指定可能）               */
/*      : byStat = 書き込むステータス（ステータスは１バイト）                */
/*      :          bit5～bit2のみ有効。                                      */
/* 戻値 : SubWriteStatus_sFlash() ==  TRUE : 正常終了                        */
/*      :                         == FALSE : 異常終了                        */

static BOOL SubWriteStatus_sFlash(WORD wFlashNum, BYTE byStat)
{
    int     nTry;
    BYTE    abyCmd[2];
    BYTE    byReady;
```

```
 85     BOOL    bResult;
 86
 87     /* ライトイネーブル **********************/
 88     if(!SubWriteEnable_sFlash(wFlashNum))
 89         return FALSE;
 90
 91     /* ステータス書き込み ********************/
 92     abyCmd[0] = FLASH_WR_STAT_CMD;
 93     abyCmd[1] = (BYTE)(byStat & 0x3C);
 94     nTry = 3;
 95     do{
 96         /* フラッシュメモリの選択とSPI通信開始 */
 97         Enable_SPI0(SPI_ENABLE, wFlashNum);
 98         /* ステータス書き込みコマンド送信 */
 99         bResult = Tranceive_SPI0(2, 0, &abyCmd[0], NULL);
100         /* SPI通信停止 */
101         Enable_SPI0(SPI_DISABLE, 0);
102         if(bResult) break;
103     }while(--nTry != 0);
104
105     /* ステータス読み込み ********************/
106     nTry = 100;
107     do{
108         /* フラッシュメモリの選択とSPI通信開始 */
109         Enable_SPI0(SPI_ENABLE, wFlashNum);
110         /* ステータス読み込みコマンド送信 */
111         bResult = Tranceive_SPI0(1, 1, &ReadStatCmd, &byReady);
112         /* SPI通信停止 */
113         Enable_SPI0(SPI_DISABLE, 0);
114         if(bResult){
115             if((byReady & 0x01) == 0)    /* READY? */
116                 break;
117         }
118     }while(--nTry != 0);
119
120     return bResult;
121 }
122
123 /*****************************************************************************/
124 /*                  シリアルフラッシュメモリのREADYを確認する                */
125 /*****************************************************************************/
126 /* 引数 : wFlashNum = フラッシュメモリの選択（０のみ指定可能）               */
127 /*      : bWrCheck = TRUEで書き込み可能かどうかを調べる                      */
128 /* 戻値 : SubCheckReady_sFlash() ==  TRUE : READY（正常）                    */
129 /*      :                        == FALSE : NOT READY（異常終了）            */
130
131 static BOOL SubCheckReady_sFlash(WORD wFlashNum, BOOL bWrCheck)
132 {
133     int     nTry;
134     BYTE    byStat;
135     BOOL    bResult;
136
137     /* READY状態になるまで戻らない */
138     byStat = 0x00;
139     nTry = 0;
140     while(1){
141         bResult = SubReadStatus_sFlash(wFlashNum, &byStat);
142         if(bResult){
143             if((byStat & 0x01) == 0)     /* b0=0:READY */
144                 break;
145             nTry = 0;
146         }
147         else{
148             ++nTry;
149             if(nTry >= 3)
150                 return FALSE;
151         }
152     }
153     /* 書き込み可能かどうかを調べる */
154     if(bWrCheck){
155         if((byStat & 0x02) == 0)         /* b1=0:書込み禁止 */
156             return FALSE;
157     }
158
159     return TRUE;
160 }
161
162 /*****************************************************************************/
163 /*                  シリアルフラッシュメモリの全ビットイレース               */
164 /*****************************************************************************/
165 /* 引数 : wFlashNum = フラッシュメモリの選択（０のみ指定可能）               */
166 /* 戻値 : BulkErase_sFlash() ==  TRUE : 正常終了                             */
167 /*      :                    == FALSE : 異常終了                             */
168
```

```
169 BOOL BulkErase_sFlash(WORD wFlashNum)
170 {
171     int     nTry;
172     BOOL    bResult;
173 
174     /* フラッシュのREADYを確認する */
175     if((bResult = SubCheckReady_sFlash(wFlashNum, FALSE)) == FALSE)
176         return bResult;
177     /* 書き込み許可にする */
178     if((bResult = SubWriteEnable_sFlash(wFlashNum)) == FALSE)
179         return bResult;
180 
181     /* コマンド送信実施 */
182     nTry = 3;
183     do{
184         /* フラッシュメモリの選択とSPI通信開始 */
185         Enable_SPI0(SPI_ENABLE, wFlashNum);
186         /* イレースコマンド送信 */
187         bResult = Tranceive_SPI0(1, 0, &EraseChipCmd, NULL);
188         /* SPI通信停止 */
189         Enable_SPI0(SPI_DISABLE, 0);
190         if(bResult) break;
191     }while(--nTry != 0);
192 
193     return bResult;
194 }
195 
196 /*****************************************************************************/
197 /*                    シリアルフラッシュメモリにページ書き込み               */
198 /*****************************************************************************/
199 /* 引数 : wFlashNum = フラッシュメモリの選択（０のみ指定可能）               */
200 /*      : wWrSize   = 書き込みバイト数（最大256バイト）                      */
201 /*      : dwWrAddr  = 書き込むフラッシュメモリのアドレス                     */
202 /*      : pbyData   = 書き込みデータのアドレス                               */
203 /* 戻値 : WritePage_sFlash() ==  TRUE : 正常終了                             */
204 /*      :                    == FALSE : 異常終了                             */
205 
206 BOOL WritePage_sFlash(WORD wFlashNum, WORD wWrSize, DWORD dwWrAddr, const BYTE *pbyData)
207 {
208     BYTE    *pBuff;
209     int     nTry;
210     BOOL    bResult;
211     WORD    wSize;
212 
213     /* フラッシュのREADYを確認 */
214     if((bResult = SubCheckReady_sFlash(wFlashNum, FALSE)) == FALSE)
215         return bResult;
216     /* 書き込み許可にする */
217     if((bResult = SubWriteEnable_sFlash(wFlashNum)) == FALSE)
218         return bResult;
219 
220     /* 書き込みコマンドとアドレス部を準備 */
221     pBuff = &abyWrBuff[0];
222     *(pBuff++) = FLASH_WR_DATA_CMD;                 /* Byte/Page Program Command */
223     *(pBuff++) = (BYTE)(dwWrAddr >> 16);
224     *(pBuff++) = (BYTE)(dwWrAddr >> 8);
225     *(pBuff++) = (BYTE)dwWrAddr;
226     wSize = 4;
227     /* 書き込みデータをバッファに転送 */
228     while(wWrSize != 0){
229         *(pBuff++) = *(pbyData++);
230         wSize++;
231         --wWrSize;
232     }
233     /* 書き込み実施 */
234     nTry = 3;
235     do{
236         /* フラッシュメモリの選択とSPI通信開始 */
237         Enable_SPI0(SPI_ENABLE, wFlashNum);
238         /* 書き込みコマンド送信 */
239         bResult = Tranceive_SPI0(wSize, 0, &abyWrBuff[0], NULL);
240         /* SPI通信停止 */
241         Enable_SPI0(SPI_DISABLE, 0);
242         if(bResult) break;
243     }while(--nTry != 0);
244     /* フラッシュメモリを書き込み禁止にする */
245     SubWriteDisable_sFlash(wFlashNum);
246 
247     return bResult;
248 }
249 
250 /*****************************************************************************/
251 /*                  シリアルフラッシュメモリからページ読み込み               */
252 /*****************************************************************************/
```

```
/* 引数 : wFlashNum = フラッシュメモリの選択（０のみ指定可能）                */
/*      : wRdSize   = 読み込むバイト数（最大256バイト）                      */
/*      : dwRdAddr  = 読み込むフラッシュメモリのアドレス                     */
/*      : pbyBuff   = 読み込んだデータの格納アドレス                         */
/* 戻値 : ReadPage_sFlash() ==  TRUE : 正常終了                              */
/*      :                   == FALSE : 異常終了                              */

BOOL ReadPage_sFlash(WORD wFlashNum, WORD wRdSize, DWORD dwRdAddr, BYTE *pbyBuff)
{
    int     nTry;
    BOOL    bResult;
    BYTE    abyCmd[6];

    /* フラッシュのREADYを確認 */
    if((bResult = SubCheckReady_sFlash(wFlashNum, FALSE)) == FALSE)
        return bResult;

    /* 読み込みコマンド送信ブロックの作成 */
    abyCmd[0] = FLASH_RD_DATA_CMD;                  /* 5pass(high speed) read command  */
    abyCmd[1] = (BYTE)(dwRdAddr >> 16);
    abyCmd[2] = (BYTE)(dwRdAddr >> 8);
    abyCmd[3] = (BYTE)dwRdAddr;
    abyCmd[4] = 0x00;                               /* Dummy bit */
    /* コマンド送信実施 */
    nTry = 3;
    do{
        /* フラッシュメモリの選択とSPI通信開始 */
        Enable_SPI0(SPI_ENABLE, wFlashNum);
        /* データ読み込みコマンド送信 */
        bResult = Tranceive_SPI0(5, wRdSize, &abyCmd[0], pbyBuff);
        /* SPI通信停止 */
        Enable_SPI0(SPI_DISABLE, 0);
        if(bResult) break;
    }while(--nTry != 0);

    return bResult;
}

/*****************************************************************************/
/*                      シリアルフラッシュメモリ制御準備                     */
/*****************************************************************************/
/* ※シリアルフラッシュメモリを使う前に最初に１回だけ呼ぶこと。              */
/* 引数 : なし                                                               */
/* 戻値 : Prepare_sFlash() ==  TRUE : 正常終了                               */
/*      :                  == FALSE : 異常終了                               */

BOOL Prepare_sFlash(void)
{
    /* SPI I/F初期化 */
    Init_SPI0();
    /* SPI通信停止 */
    Enable_SPI0(SPI_DISABLE, 0);

    return TRUE;
}
```