```
/************************************************************************/
/************************************************************************/
/***                                                                  ***/
/***                    キー・マトリクス入力（プログラム例）          ***/
/***                         MatrixKeyBoard.c                         ***/
/***                                                                  ***/
/************************************************************************/
/************************************************************************/
/* 注意 : (1)キーマトリクスは４行×８列（ビット）であるとする。          */
/*      : (2)キーの行は、ポート６のb3～b0の４ビットの出力を使うものとする。 */
/*      : (3)キーの列は、ポート４のb7～b0の８ビットの入力を使うものとする。 */
/*      : (4)複数のキーが押された場合、最初に見つけた１つのキーのみしか    */
/*      :    検知しない。                                               */

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

#pragma section
/************************************************************************/

#define SCANINT (1)                         /* スキャン周期（1mS）          */
#define LINEMAX (4)                         /* マトリクスの行数（キーの行数＝４） */
#define CHAT_TIME (48/LINEMAX/SCANINT)      /* チャタリングキャンセル時間（48mS） */
#define RPT_STTIM (1000/LINEMAX/SCANINT)    /* リピート開始時間（1000mS）   */
#define RPT_INTTM (100/LINEMAX/SCANINT)     /* リピート間隔（100mS）        */

static int  st_nScanLine;                   /* スキャン中のキー行（0～LINEMAX-1） */
static BYTE st_byNowIn,st_byLastIn;
static BYTE st_byValidIn,st_byKeyState;
static BYTE st_byChatCnt;
static const BYTE st_abyScanPos[LINEMAX] = {0x01, 0x02, 0x04, 0x08};    /* b0～b3の行位置を示す */


/************************************************************************/
/*                              初期化                                  */
/************************************************************************/
/* 注意 : 最初に１回呼ぶこと。                                          */

void Init_Keyin(void)
{
    st_nScanLine = 0;
    st_byNowIn = 0;
    st_byLastIn = 0;
    st_byValidIn = 0;
    st_byChatCnt = 0;
}

/************************************************************************/
/*                            キースキャン                              */
/************************************************************************/
/* 注意 : (1)SCANINT (=1mS) 周期割り込みから呼ぶこと。                   */
/*      : (2)スキャン過程で最初に見つけたキーのみ有効。                 */
/*      :    複数のキーを押していても検知できない。                     */
/*      : (3)st_byLastIn,st_byNowIn,st_byValidIn,st_byKeyStateの構造    */
/*      :    [76543210]                                                 */
/*      :     ||||||||                                                  */
/*      :     |00|||||                                                  */
/*      :     |  |||bbb:列番号（ビット番号:0～7）                       */
/*      :     |  cc:行番号（0～3）                                      */
/*      :     v:"1"で有効                                              */

void Scan_Keyboard(void)
{
    BYTE    byInData;
    BYTE    byTmpLine,byTmpCnt;

    /* 入力ポート４から、st_nScanLineで選択中行のキー状態を取り込む */
    byInData = PORT4.PORT.BYTE;
    /* スキャン一巡の間に押しているキーが見つかれば以降は無視する */
    if((byInData != 0x00) && (st_byNowIn == 0)){
        byTmpLine = (BYTE)((st_nScanLine << 3) & 0x18);
        byTmpCnt  = 0;
        do{
            if((byInData & 0x01) != 0){
```

```
                st_byNowIn = (BYTE)((byTmpLine | byTmpCnt) | 0x80); /* b7='1'は有効を示す */
                break;
            }
            byInData >>= 1;
        }while(++byTmpCnt < 8);
    }
    /* 次の行のキーを選択 */
    st_nScanLine++;
    if(st_nScanLine >= LINEMAX){
        st_nScanLine = 0;
        /* 前回と変化している場合はキーの変化とする */
        if(st_byLastIn != st_byNowIn){
            st_byLastIn = st_byNowIn;
            st_byKeyState = 0;
            st_byChatCnt  = 0;
        }
        /* 前回と同じ場合はキーが安定しているとする */
        else{
            if(st_byChatCnt != 255){
                st_byChatCnt++;
                /* チャタリングキャンセル処理 */
                if(st_byChatCnt == CHAT_TIME){
                    if(st_byLastIn != 0){       /* ONしている */
                        st_byValidIn  = st_byLastIn;
                        st_byKeyState = st_byLastIn;
                    }
                }
                /* オートリピート処理（下記に指定したキー以外はリピートしない） */
                else if(st_byChatCnt == RPT_STTIM){
                    if(st_byLastIn != 0){       /* ONしている */
                        switch(st_byLastIn){
                        /* ↓オートリピートさせたいキーがあれば、case 0x??の部分に16進数で列挙する */
                        case 0x??:
                         :    :
                        case 0x??:
                            st_byValidIn = st_byLastIn;
                            st_byChatCnt = RPT_STTIM - RPT_INTTM;
                            break;
                        default:
                            break;
                        }
                    }
                }
            }
        }
        st_byNowIn = 0;
    }
    /* 次の行のスキャンパターンを、出力ポート６に出力 */
    PORT6.DR.BYTE = (BYTE)((PORT6.DR.BYTE & 0xF0) | st_abyScanPos[st_nScanLine]);
}

/*****************************************************************************/
/*                      キーのコードを変換する                               */
/*****************************************************************************/

static const BYTE abyKeyCodeTbl[LINEMAX*8]={
    /* Scan Line : 0行目 */
    KEY_00,                     /* b0:'A' */
    KEY_01,                     /* b1:'B' */
    KEY_02,                     /* b2:'C' */
    KEY_03,                     /* b3:'D' */
    KEY_04,                     /* b4:'E' */
    KEY_05,                     /* b5:'F' */
    KEY_06,                     /* b6:'G' */
    KEY_07,                     /* b7:'H' */
    /* Scan Line : 1行目 */
    KEY_10,                     /* b0:'I' */
    KEY_11,                     /* b1:'J' */
    KEY_12,                     /* b2:'K' */
    KEY_13,                     /* b3:'L' */
    KEY_14,                     /* b4:'M' */
    KEY_15,                     /* b5:'N' */
    KEY_16,                     /* b6:'O' */
    KEY_17,                     /* b7:'P' */
    /* Scan Line : 2行目 */
    KEY_20,                     /* b0:'Q' */
    KEY_21,                     /* b1:'R' */
    KEY_22,                     /* b2:'S' */
```

```
    KEY_23,                     /* b3:'T' */
    KEY_24,                     /* b4:'U' */
    KEY_25,                     /* b5:'V' */
    KEY_26,                     /* b6:'W' */
    KEY_27,                     /* b7:'X' */
    /* Scan Line : 3行目 */
    KEY_30,                     /* b0:'Y' */
    KEY_31,                     /* b1:'Z' */
    KEY_32,                     /* b2:'0' */
    KEY_33,                     /* b3:'1' */
    KEY_34,                     /* b4:'2' */
    KEY_35,                     /* b5:'3' */
    KEY_36,                     /* b6:'4' */
    KEY_37,                     /* b7:'5' */
};

static BYTE Conv_Keycode(BYTE byKeyPtn)
{
    return (abyKeyCodeTbl[(byKeyPtn & 0x1F)]);
}

/*****************************************************************************/
/*                        新たに押されたキーを読み込む                       */
/*****************************************************************************/
/* 引数 : pbyKey = キーコード格納アドレス (コード＝KEY_xx)                   */
/* 戻値 : Get_Key() == FALSE : 押されていない                                */
/*      :           ==  TRUE : 押された (*pbyKey←キーコード格納)             */

BOOL Get_Key(BYTE* pbyKey)
{
    if((st_byValidIn & 0x80) == 0){
        *pbyKey = 0;
        return FALSE;
    }
    *pbyKey = Conv_Keycode(st_byValidIn);
    st_byValidIn = 0;

    return TRUE;
}

/*****************************************************************************/
/*                        現在のキー押下状態を読み込む                       */
/*****************************************************************************/
/* 引数 : pbyKey = キーコード格納アドレス (コード＝KEY_xx)                   */
/* 戻値 : Check_Key() == 0x00 : 押されていない                               */
/*      :             != 0x00 : 押されたキーコード                           */

BYTE Check_Key(void)
{
    if((st_byKeyState & 0x80) == 0)
        return 0;
    return Conv_Keycode(st_byKeyState);
}

/* End fo File */
```