```
/*****************************************************************************
    Event and Cyclic Task's for
        sample project using rtm_msp430/MSP430F2013

        May      25,1998    New version on H8/3048F
        August    6,2006    for MSP430
        August    6,2006    looks like running!
        August    8,2006

    Copyright (C) 1998, 2006 Kenji Arai/JH1PJL
    All rights reserved. Permission is granted to use, modify,or redistribute
    this software so long as it is not sold or exploited for profit.

    THIS SOFTWARE IS PROVIDED AS IS AND WITHOUT WARRANTY OF ANY KIND,
    EITHER EXPRESSED OR IMPLIED.
*****************************************************************************/
/* -----< Include Files >--------------------------------------------------- */
#include    <msp430x20x3.h>
#include    "rtm_msp430.h"

/* -----< Function Prototype >---------------------------------------------- */
void main( void );

void ed0_main( void );
void ed1_main( void );
void ed2_main( void );
void ed3_main( void );
void cy0_init( void );
void cy0_main( void );
void cy1_init( void );
void cy1_main( void );
void cy2_init( void );
void cy2_main( void );
void cy3_init( void );
void cy3_main( void );
void cy4_init( void );
void cy4_main( void );
void cy5_init( void );
void cy5_main( void );
void cy6_init( void );
void cy6_main( void );
void cy7_init( void );
void cy7_main( void );

void general_init( void );

/* -----< Function Prototype (Extern) >------------------------------------- */


/* -----< Constant data in ROM >-------------------------------------------- */
extern char *const ROM_copyright =
"Arai,Kenji/JH1PJL(c)2006 kenjia@sannet.ne.jp <MSP430F2013 sample project>";

#if defined (STD)
/*      Cyclic Task Period                  */
const unsigned char cyclic_period[8] = {
    T_500MS,    /* cyclic task no.0 */ // ????
    T_500MS,    /* cyclic task no.1 */ // ????
    T_100MS,    /* cyclic task no.2 */ // ????
    T_200MS,    /* cyclic task no.3 */ // ????
    T_500MS,    /* cyclic task no.4 */ // NOT USE
    T_500MS,    /* cyclic task no.5 */ // NOT USE
```

```
    T_500MS,    /* cyclic task no.6 */ // NOT USE
    T_500MS     /* cyclic task no.7 */ // NOT USE
};
//      ONLY FOR POWER ON INITIALIZE
const unsigned char cyclic_init_period[8] = {
    T_500MS,        /* cyclic task no.0 */
    T_200MS+T_20MS, /* cyclic task no.1 */
    T_200MS+T_50MS, /* cyclic task no.2 */
    T_200MS+T_10MS, /* cyclic task no.3 */
    T_200MS,        /* cyclic task no.4 */
    T_200MS+T_100MS,/* cyclic task no.5 */
    T_500MS+T_6MS,  /* cyclic task no.6 */
    T_500MS+T_10MS  /* cyclic task no.7 */
};
#endif

#if defined (OPT)
/*      Cyclic Task Period                  */
const unsigned char cyclic_period[8] = {
    T_50MS, /* cyclic task no.0 */ // ????
    T_500MS,    /* cyclic task no.1 */ // ????
    T_100MS,    /* cyclic task no.2 */ // ????
    T_200MS,    /* cyclic task no.3 */ // ????
    T_500MS,    /* cyclic task no.4 */ // NOT USE
    T_500MS,    /* cyclic task no.5 */ // NOT USE
    T_500MS,    /* cyclic task no.6 */ // NOT USE
    T_500MS     /* cyclic task no.7 */ // NOT USE
};
//      ONLY FOR POWER ON INITIALIZE
const unsigned char cyclic_init_period[8] = {
    T_50MS,         /* cyclic task no.0 */
    T_200MS+T_20MS, /* cyclic task no.1 */
    T_200MS+T_50MS, /* cyclic task no.2 */
    T_200MS+T_12MS, /* cyclic task no.3 */
    T_200MS,        /* cyclic task no.4 */
    T_200MS+T_100MS,/* cyclic task no.5 */
    T_500MS+T_8MS,  /* cyclic task no.6 */
    T_500MS+T_12MS  /* cyclic task no.7 */
};
#endif

/*-----< RAM assign >-------------------------------------------------------- */

/*-----< RAM assign (External data) >---------------------------------------- */

/*****************************************************************************
************* Event driven task and Cyclic task ******************************
*****************************************************************************/

/*****************************************************************************
    Event Driven Tasks
*****************************************************************************/
/*---------------------------------------------------------------------------
    ED0 Task --- ???????????
---------------------------------------------------------------------------*/
void ed0_main(void)
{
    ;
}

/*---------------------------------------------------------------------------
    ED1 Task --- ???????????
```

```
-------------------------------------------------------------------------------*/
void ed1_main(void)
{
    ;
}

/*-------------------------------------------------------------------------------
    ED2 Task --- Not use
-------------------------------------------------------------------------------*/
void ed2_main(void){ ; }

/*-------------------------------------------------------------------------------
    ED3 Task --- Not use
-------------------------------------------------------------------------------*/
void ed3_main(void) { ; }

/*******************************************************************************
    Cyclic Tasks
*******************************************************************************/
/*-------------------------------------------------------------------------------
    CY0 Task --- ?????????
-------------------------------------------------------------------------------*/
static int flg;

void cy0_init( void )
{
    tim_usr0 = T_1S_16B;
    flg = 0;
}

void cy0_main( void )
{
  if (tim_usr0 == 0){
    tim_usr0 = T_1S_16B;
    if (flg){
        flg = 0;
        P1OUT |= 0x01;
    } else {
        flg = 1;
        P1OUT &= ~0x01;
    }
  }
}

/*-------------------------------------------------------------------------------
    CY1 Task --- ???????
-------------------------------------------------------------------------------*/
void cy1_init( void )
{
    ;
}

void cy1_main( void )
{
    ;
}

/*-------------------------------------------------------------------------------
    CY2 Task --- ???????
-------------------------------------------------------------------------------*/
void cy2_init( void )
{
```

```
    ;
}

void cy2_main( void )
{
    ;
}

/*-------------------------------------------------------------------------
    CY3 Task  --- Not use
-------------------------------------------------------------------------*/
void cy3_init( void ) { ; }
void cy3_main( void ) { ; }
/*-------------------------------------------------------------------------
    CY4 Task  --- Not use
-------------------------------------------------------------------------*/
void cy4_init( void ) { ; }
void cy4_main( void ) { ; }
/*-------------------------------------------------------------------------
    CY5 Task  --- Not use
-------------------------------------------------------------------------*/
void cy5_init( void ) { ; }
void cy5_main( void ) { ; }
/*-------------------------------------------------------------------------
    CY6 Task  --- Not use
-------------------------------------------------------------------------*/
void cy6_init( void ) { ; }
void cy6_main( void ) { ; }
/*-------------------------------------------------------------------------
    CY7 Task  --- Not use
-------------------------------------------------------------------------*/
void cy7_init( void ) { ; }
void cy7_main( void ) { ; }

/*************************************************************************
    Subroutine
*************************************************************************/
/********** General  initialize ******************************************/
void general_init( void )
{
    ;
}

/********** Port initialize **********************************************/
void port_init ( void )
{
    P1DIR = 0xFF;             // All P1.x outputs
    P1OUT = 0;                // All P1.x reset
    P2DIR = 0xFF;             // All P2.x outputs
    P2OUT = 0;                // All P2.x reset
}

/*************************************************************************
    Watchdog timer interrupt
*************************************************************************/
#pragma vector=WDT_VECTOR
__interrupt void watchdog_timer (void)
{
    LPM3_EXIT;                // Clear LPM3 bits from 0(SR)
    req_cyctsk();             // Request to start RTM
}
```

```
/*****************************************************************************
    Main routine
*****************************************************************************/
void main( void )
{
    /********** RTM ************************************************************/
    rtm_msp430_init();
    /********** SCI ************************************************************/
    //;
    /********** Interrupt ******************************************************/
    //__disable_interrupt();  // Disable interrupts
    /********** Timer **********************************************************/
    //reset_timer();
    /********** others *********************************************************/
    general_init();

    /********** Initialize each task *******************************************/
    cy0_init();
    cy1_init();
    cy2_init();
    cy3_init();
    cy4_init();

    /********** Hardware related initialize ************************************/
    BCSCTL1 |= DIVA_0;        // ACLK/1
    BCSCTL3 |= LFXT1S_2;      // VLOCLK=10 selected
    WDTCTL = WDT_ADLY_1_9;    // WDT 1.9mS(@32KHz) interval timer
    IE1 |= WDTIE;             // Enable WDT interrupt

    port_init();              // Port initialize

    /********** Interrupt ******************************************************/
    __enable_interrupt();   // Enable interrupts

    /********** Main loop ******************************************************/
    while(1)
    {
        rtm_core();           // Start dispatch routine
        LPM3;                 // Enter LPM3
        _NOP();
    }
}
```