```
1 //*****************************************************************************
  *****
2 //*
  *
3 //*                             I2C ROUTINES
  *
4 //*                             REQUIRES GLOBAL
  *
5 //* TXByteCtr,*PTxData, RXByteCtr,*PRxData,TX_I2CByteCtr,*PTx_I2CData,
  RX_I2CByteCtr,*PRx_I2CData *
6 //*****************************************************************************
  *****
7 #include <msp430.h>
8 #include "stdio.h"
9 #include "stdlib.h"
10 #include "string.h"
11 #include "stdint.h"
12 #include "stdlib.h"
13
14 #ifndef __MSP430FR2311
15 #define __MSP430FR2311
16 #include "MSP430FR2311.h"
17 #endif
18 #ifndef init_h
19 #define init_h
20 #include "init.h"
21 #endif
22 #ifndef tempsensor_h
23 #define tempsensor_h
24 #include "tempsensor.h"
25 #endif
26 #ifndef i2c_h
27 #define i2c_h
28 #include "i2c.h"
29 #endif
30
31 extern unsigned int TXByteCtr,RXByteCtr,TX_I2CByteCtr,RX_I2CByteCtr;
32 extern unsigned char *PRxData,
  *PTxData,RX_DONE,error_flag,testi2c,IC2_on,*PTx_I2CData,*PRx_I2CData;
33
34
35
36
37 void init_i2c(void)
38 {
39
40     // Configure Pins for I2C
41     P1SEL0 |= BIT2 | BIT3;                            // I2C pins
42     PM5CTL0 &= ~LOCKLPM5;
43      UCB0CTLW0 |= UCSWRST;                             // put eUSCI_B in reset state
44      UCB0CTLW0 |= UCMODE_3 | UCMST;                    // I2C master mode, SMCLK
45     UCB0CTLW0 &= ~UCSYNC;
46     UCB0BRW = 14;                                   // fSCL = SMCLK/120 = ~100kHz
47     UCB0I2COA0 = MST_ADD ;                           // Slave Address is 068h
48     UCB0CTL1 &= ~UCSWRST;                           // Clear SW reset, resume operation
49  // UCB0IE |= UCTXIE | UCNACKIE;                            // Enable TX interrupt
50
```

```
51 }
52
53
54 void i2c_xmit( unsigned char *dat,unsigned char add,unsigned char byte)
55 {
56     UCB0I2CSA =add;                      // i2c Address
57     TX_I2CByteCtr = byte;                        // bytes to xmit
58     PTx_I2CData =dat;                    // Start of TX buffer
59     RX_DONE=0;
60     UCB0IFG &= ~UCRXIFG;                                    // Remain in LPM0 until master
61     error_flag = 0;                           //initialize error flag
62     IC2_on=1;                                 //set up for ic2
63   // str_xmit_i2c();
64 //    __delay_cycles(20);                    // Delay required between transaction
65      UCB0IE &= UCRXIE;
66     UCB0IE |= UCTXIE;                         // CLEAR TX interrupt
67      while (UCB0CTL1 & UCTXSTP);              // Ensure stop condition got sent
68      __no_operation();
69      UCB0CTLW0 |= UCTR + UCTXSTT;          // I2C TX, start condition
70      while(TX_I2CByteCtr !=0);
71      while (UCB0CTL1 & UCTXSTP);              // Ensure stop condition got sent
72      __no_operation();
73     while (UCB0STAT & UCBBUSY);            // make sure buss in not busy
74
75 }
76 void i2c_rcv( unsigned char *dat,unsigned char add,unsigned char byte)
77 {
78     RX_I2CByteCtr=byte;    //no of bytes to read
79
80       UCB0I2CSA =add;                     // i2c Address
81    UCB0IE &= ~UCTXIE;                         // CLEAR TX interrupt
82    PRx_I2CData = dat;   // RX array start address
83   UCB0IFG &= ~UCRXIFG;                  // Clear USCI_B0 TX int flag
84 //  UCB0IE |= UCRXIE;                          // Enable RX interrupt
85   UCB0IE |= UCSTPIE + UCSTTIE + UCRXIE;
86   while (UCB0CTL1 & UCTXSTP);             // Ensure stop condition got sent
87   UCB0CTL1 &= ~UCTR;               //SET TO RECEIVER
88  __no_operation();
89   UCB0CTL1 |= UCTXSTT;     //Enable i2c start condition
90   while (UCB0CTL1 & UCTXSTT);   //Ensure stART condition got sent
91   while (RX_I2CByteCtr !=0);
92   while (UCB0STAT & UCBBUSY);             // make sure buss in not busy
93   testi2c=0;
94 }
95 void str_xmit_i2c(void)
96 {
97      __delay_cycles(20);                    // Delay required between transaction
98        UCB0IE &= UCRXIE;
99      UCB0IE |= UCTXIE;                          // CLEAR TX interrupt
100     while (UCB0CTL1 & UCTXSTP);              // Ensure stop condition got sent
101      __no_operation();
102     UCB0CTLW0 |= UCTR + UCTXSTT;          // I2C TX, start condition
103     while(TX_I2CByteCtr !=0);
104     while (UCB0CTL1 & UCTXSTP);              // Ensure stop condition got sent
105     __no_operation();
106  //   while (UCB0STAT & UCBBUSY);            // make sure buss in not busy
107 }
```

```
108
109
110 void wait_ic2_cmd(unsigned char *rbuff)
111 {
112    *(rbuff+0)=0x00;
113    *(rbuff+1)=0x00;
114    *(rbuff+2)=0x00;
115
116     RX_DONE=0;
117     PRxData = (unsigned char *)rbuff;              // Start of RX buffer
118     RXByteCtr = 0;                                // Clear RX byte count
119 //     __bis_SR_register(LPM0_bits + GIE);        // Enter LPM0, enable interrupts
120                                                   // Remain in LPM0 until master
121                                                   // finishes TX
122  __bis_SR_register( GIE);                         // Enter LPM0, enable interrupts
123                                                   // Remain in LPM0 until master
124                                                   // finishes TX
125   while((RX_DONE==0x00) ) ;
126
127
128 }
129 void init_ic2_xmit(void)
130 {
131      UCB0CTL1 &= ~UCTR;                           //SET TO XMI
132      UCB0IE &= ~UCTXIE;                           // CLEAR TX interrupt
133      UCB0IFG &= ~UCRXIFG;                         // Clear USCI_B0 TX int flag
134      UCB0CTL1 |= UCSWRST;                         // Enable SW reset
135      UCB0CTLW0 = UCMODE_3 ;                        // I2C Slave, synchronous mode
136      UCB0I2COA0 = MST_ADD;                        // Own Address is 048h
137      UCB0CTL1 &= ~UCSWRST;                        // Clear SW reset, resume operation
138      UCB0IE |= UCSTPIE + UCSTTIE + UCTXIE;        // Enable STT, STP & RX interrupt
139  }
140 void init_ic2_rcv(void)
141 {
142   UCB0CTL1 &= ~UCTR;                              //SET TO RECEIVER
143   UCB0CTL1 |= UCSWRST;                            // Enable SW reset
144   UCB0CTLW0 = UCMODE_3 ;                           // I2C Slave, synchronous mode
145   UCB0I2COA0 = MST_ADD;                           // Own Address is 048h
146   UCB0CTL1 &= ~UCSWRST;                           // Clear SW reset, resume operation
147   UCB0IE |= UCSTPIE + UCSTTIE + UCRXIE;           // Enable STT, STP & RX interrupt
148 }
149
150 //*****************************************************************************
151 //*                                                                           *
152 //*                           END  I2C ROUTINES                               *
153 //*                                                                           *
154 //*****************************************************************************
155
```