MSP430FR5969 I2C 测试

实例代码：

```
//  MSP430FR59xx Demo - eUSCI_B0 I2C Master RX multiple bytes from MSP430 Slave
//
//  Description: This demo connects two MSP430's via the I2C bus. The master
//  reads 5 bytes from the slave. This is the MASTER CODE. The data from the slave
//  transmitter begins at 0 and increments with each transfer.
//  The USCI_B0 RX interrupt is used to know when new data has been received.
//  ACLK = n/a, MCLK = SMCLK = BRCLK =  DCO = 1MHz
//
//    *****used with "MSP430FR59xx_euscib0_i2c_11.c"****
//
//                                /|\  /|\
//              MSP430FR5969      10k  10k     MSP430F5969
//                 slave         |    |        master
//             -----------------   |    |  -----------------
//           -|XIN  P1.6/UCB0SDA|<-|----+->|P1.6/UCB0SDA  XIN|-
//            |                 | |       |                 | 32kHz
//           -|XOUT             | |       |             XOUT|-
//            |     P1.7/UCB0SCL|<-+------>|P1.7/UCB0SCL     |
//            |                 |          |             P1.0|--> LED
//
//   P. Thanigai
//   Texas Instruments Inc.
//   Feb 2012
//   Built with CCS V5.5
//******************************************************************************
#include <msp430.h>

volatile unsigned char RXData;

int main(void)
{
  WDTCTL = WDTPW | WDTHOLD;

  // Configure GPIO
  P1OUT &= ~BIT0;                         // Clear P1.0 output latch
  P1DIR |= BIT0;                          // For LED
  P1SEL1 |= BIT6 | BIT7;                  // I2C pins

  // Disable the GPIO power-on default high-impedance mode to activate
  // previously configured port settings
  PM5CTL0 &= ~LOCKLPM5;

  // Configure USCI_B0 for I2C mode
  UCB0CTLW0 |= UCSWRST;                   // Software reset enabled
  UCB0CTLW0 |= UCMODE_3 | UCMST | UCSYNC;   // I2C mode, Master mode, sync
  UCB0CTLW1 |= UCASTP_2;                  // Automatic stop generated
                                          // after UCB0TBCNT is reached
  UCB0BRW = 0x0008;                       // baudrate = SMCLK / 8
  UCB0TBCNT = 0x0005;                     // number of bytes to be received
  UCB0I2CSA = 0x0048;                     // Slave address
  UCB0CTL1 &= ~UCSWRST;
  UCB0IE |= UCRXIE | UCNACKIE | UCBCNTIE;

  while (1)
  {
    __delay_cycles(2000);
    while (UCB0CTL1 & UCTXSTP);           // Ensure stop condition got sent
    UCB0CTL1 |= UCTXSTT;                  // I2C start condition

    __bis_SR_register(LPM0_bits | GIE);     // Enter LPM0 w/ interrupt
  }
}

#if defined(__TI_COMPILER_VERSION__) || defined(__IAR_SYSTEMS_ICC__)
#pragma vector = USCI_B0_VECTOR
__interrupt void USCI_B0_ISR(void)
#elif defined(__GNUC__)
void __attribute__ ((interrupt(USCI_B0_VECTOR))) USCI_B0_ISR (void)
#else
#error Compiler not supported!
#endif
{
  switch(__even_in_range(UCB0IV, USCI_I2C_UCBIT9IFG))
  {
    case USCI_NONE:          break;         // Vector 0: No interrupts
    case USCI_I2C_UCALIFG:   break;         // Vector 2: ALIFG
    case USCI_I2C_UCNACKIFG:                // Vector 4: NACKIFG
      UCB0CTL1 |= UCTXSTT;                  // I2C start condition
```

```
    break;
  case USCI_I2C_UCSTTIFG:  break;         // Vector 6: STTIFG
  case USCI_I2C_UCSTPIFG:  break;         // Vector 8: STPIFG
  case USCI_I2C_UCRXIFG3:  break;         // Vector 10: RXIFG3
  case USCI_I2C_UCTXIFG3:  break;         // Vector 12: TXIFG3
  case USCI_I2C_UCRXIFG2:  break;         // Vector 14: RXIFG2
  case USCI_I2C_UCTXIFG2:  break;         // Vector 16: TXIFG2
  case USCI_I2C_UCRXIFG1:  break;         // Vector 18: RXIFG1
  case USCI_I2C_UCTXIFG1:  break;         // Vector 20: TXIFG1
  case USCI_I2C_UCRXIFG0:                 // Vector 22: RXIFG0
    RXData = UCB0RXBUF;                   // Get RX data
    __bic_SR_register_on_exit(LPM0_bits); // Exit LPM0
    break;
  case USCI_I2C_UCTXIFG0:  break;         // Vector 24: TXIFG0
  case USCI_I2C_UCBCNTIFG:                // Vector 26: BCNTIFG
    P1OUT ^= BIT0;                        // Toggle LED on P1.0
    break;
  case USCI_I2C_UCCLTOIFG: break;         // Vector 28: clock low timeout
  case USCI_I2C_UCBIT9IFG: break;         // Vector 30: 9th bit
  default: break;
 }
}
```

由于前面介绍UART学习的时候已经对端口操作，时钟操作做了详细的介绍，这里就只介绍I2C的内容。

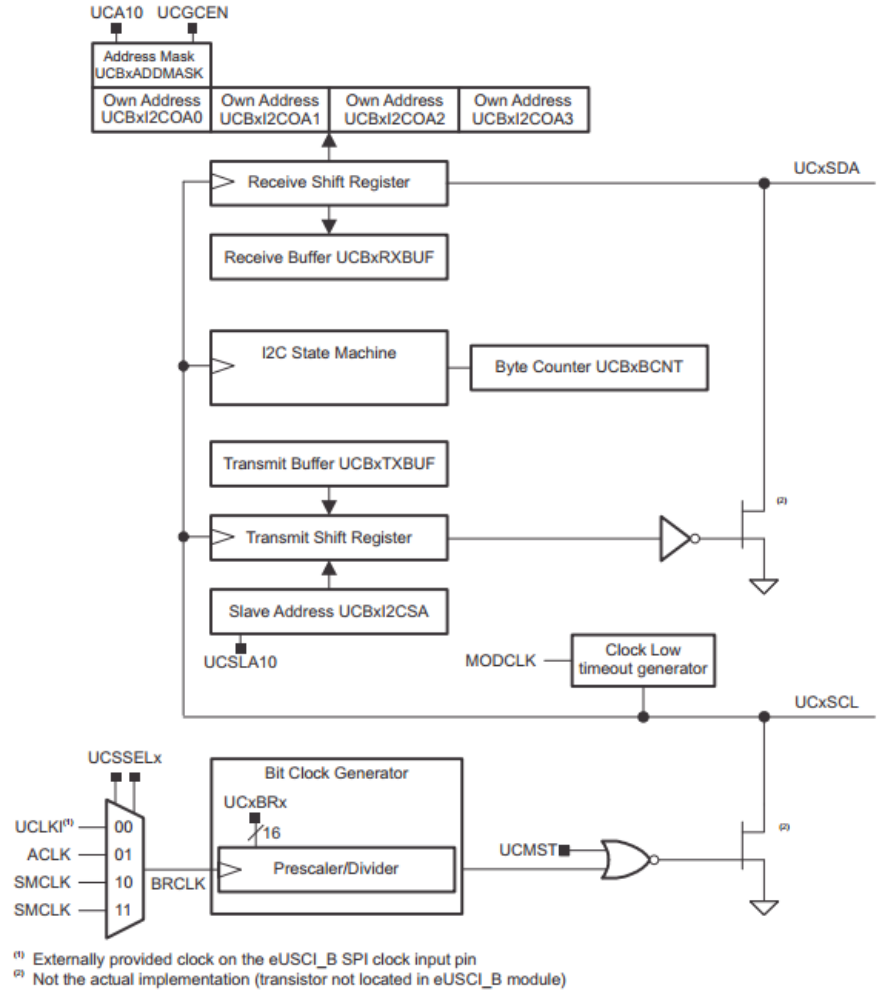

Figure 26-1. eUSCI_B Block Diagram – I²C Mode

1、配置USCI_B0为I2C模式。

（1）在配置或者重新配置eUSCI_B模块时应该先将UCSWRST置1以避免出现配置错误，配置完之后释放复位。

```
UCB0CTLW0 |= UCSWRST;                     // Software reset enabled
UCB0CTL1 &= ~UCSWRST;
```

Table 26-4. UCBxCTLW0 Register Description

Bit	Field	Type	Reset	Description
15	UCA10	RW	0h	Own addressing mode select. Modify only when UCSWRST = 1. 0b = Own address is a 7-bit address. 1b = Own address is a 10-bit address.
14	UCSLA10	RW	0h	Slave addressing mode select 0b = Address slave with 7-bit address 1b = Address slave with 10-bit address
13	UCMM	RW	0h	Multi-master environment select. Modify only when UCSWRST = 1. 0b = Single master environment. There is no other master in the system. The address compare unit is disabled. 1b = Multi-master environment
12	Reserved	R	0h	Reserved
11	UCMST	RW	0h	Master mode select. When a master loses arbitration in a multi-master environment (UCMM = 1), the UCMST bit is automatically cleared and the module acts as slave. 0b = Slave mode 1b = Master mode
10-9	UCMODEx	RW	0h	eUSCI_B mode. The UCMODEx bits select the synchronous mode when UCSYNC = 1. Modify only when UCSWRST = 1. 00b = 3-pin SPI 01b = 4-pin SPI (master or slave enabled if STE = 1) 10b = 4-pin SPI (master or slave enabled if STE = 0) 11b = I2C mode
8	UCSYNC	RW	1h	Synchronous mode enable. For eUSCI_B always read and write as 1.
7-6	UCSSELx	RW	3h	eUSCI_B clock source select. These bits select the BRCLK source clock. These bits are ignored in slave mode. Modify only when UCSWRST = 1. 00b = UCLKI 01b = ACLK 10b = SMCLK 11b = SMCLK
5	UCTXACK	RW	0h	Transmit ACK condition in slave mode with enabled address mask register. After the UCSTTIFG has been set, the user needs to set or reset the UCTXACK flag to continue with the I2C protocol. The clock is stretched until the UCBxCTL1 register has been written. This bit is cleared automatically after the ACK has been send. 0b = Do not acknowledge the slave address 1b = Acknowledge the slave address

Table 26-4. UCBxCTLW0 Register Description (continued)

Bit	Field	Type	Reset	Description
4	UCTR	RW	0h	Transmitter/receiver 0b = Receiver 1b = Transmitter
3	UCTXNACK	RW	0h	Transmit a NACK. UCTXNACK is automatically cleared after a NACK is transmitted. Only for slave receiver mode. 0b = Acknowledge normally 1b = Generate NACK
2	UCTXSTP	RW	0h	Transmit STOP condition in master mode. Ignored in slave mode. In master receiver mode, the STOP condition is preceded by a NACK. UCTXSTP is automatically cleared after STOP is generated. This bit is a don't care, if automatic UCASTPx is different from 01 or 10. 0b = No STOP generated 1b = Generate STOP
1	UCTXSTT	RW	0h	Transmit START condition in master mode. Ignored in slave mode. In master receiver mode, a repeated START condition is preceded by a NACK. UCTXSTT is automatically cleared after START condition and address information is transmitted. Ignored in slave mode. 0b = Do not generate START condition 1b = Generate START condition
0	UCSWRST	RW	1h	Software reset enable. 0b = Disabled. eUSCI_B released for operation. 1b = Enabled. eUSCI_B logic held in reset state.

（2）设置模式。

```
UCB0CTLW0 |= UCMODE_3 | UCMST | UCSYNC;   // I2C mode, Master mode, sync
```

其中，UCMODE_3 = 0x0600, UCMST = 0x0800, UCSYNC = 0x0100。

注意：eUSCI_B时钟源默认是SMCLK。

（3）设置停止条件

Table 26-5. UCBxCTLW1 Register Description

Bit	Field	Type	Reset	Description
15-9	Reserved	R	0h	Reserved
8	UCETXINT	RW	0h	Early UCTXIFG0. Only in slave mode. When this bit is set, the slave addresses defined in UCxI2COA1 to UCxI2COA3 must be disabled. Modify only when UCSWRST = 1. 0b = UCTXIFGx is set after an address match with UCxI2COAx and the direction bit indicating slave transmit 1b = UCTXIFG0 is set for each START condition
7-6	UCCLTO	RW	0h	Clock low time-out select. Modify only when UCSWRST = 1. 00b = Disable clock low time-out counter 01b = 135000 MODCLK cycles (approximately 28 ms) 10b = 150000 MODCLK cycles (approximately 31 ms) 11b = 165000 MODCLK cycles (approximately 34 ms)
5	UCSTPNACK	RW	0h	The UCSTPNACK bit allows to make the eUSCI_B master acknowledge the last byte in master receiver mode as well. This does not conform to the I2C specification and should only be used for slaves that automatically release the SDA after a fixed packet length. Modify only when UCSWRST = 1. 0b = Send a not acknowledge before the STOP condition as a master receiver (conform to I2C standard) 1b = All bytes are acknowledged by the eUSCI_B when configured as master receiver
4	UCSWACK	RW	0h	This bit selects whether sending an ACK of the address is triggered by the eUSCI_B module or is controlled by software. 0b = The address acknowledge of the slave is controlled by the eUSCI_B module 1b = The user needs to trigger the sending of the address ACK by issuing UCTXACK
3-2	UCASTPx	RW	0h	Automatic STOP condition generation. In slave mode, only settings 00b and 01b are available. Modify only when UCSWRST = 1. 00b = No automatic STOP generation. The STOP condition is generated after the user sets the UCTXSTP bit. The value in UCBxTBCNT is a don't care. 01b = UCBCNTIFG is set with the byte counter reaches the threshold defined in UCBxTBCNT 10b = A STOP condition is generated automatically after the byte counter value reached UCBxTBCNT. UCBCNTIFG is set with the byte counter reaching the threshold. 11b = Reserved

Table 26-5. UCBxCTLW1 Register Description (continued)

Bit	Field	Type	Reset	Description
1-0	UCGLITx	RW	0h	Deglitch time 00b = 50 ns 01b = 25 ns 10b = 12.5 ns 11b = 6.25 ns

```
UCB0CTLW1 |= UCASTP_2;              // Automatic stop generated
                                    // after UCB0TBCNT is reached
```

其中，UCASTP_2 = 0x0008。

（4）设置设置比特率。

26.4.3 UCBxBRW Register

eUSCI_Bx Bit Rate Control Word Register

Figure 26-19. UCBxBRW Register

15	14	13	12	11	10	9	8
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw
7	6	5	4	3	2	1	0
UCBRx							
rw	rw	rw	rw	rw	rw	rw	rw

Can be modified only when UCSWRST = 1.

Table 26-6. UCBxBRW Register Description

Bit	Field	Type	Reset	Description
15-0	UCBRx	RW	0h	Bit clock prescaler. Modify only when UCSWRST = 1.

```
UCB0BRW = 0x0008;                   // baudrate = SMCLK / 8
```

f BitClock = fBRCLK/UCBRx

（5）设置接收字节计数器阈值

26.4.5 UCBxTBCNT Register

eUSCI_Bx Byte Counter Threshold Register

Figure 26-21. UCBxTBCNT Register

15	14	13	12	11	10	9	8
Reserved							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
UCTBCNTx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Can be modified only when UCSWRST = 1.

Table 26-8. UCBxTBCNT Register Description

Bit	Field	Type	Reset	Description
15-8	Reserved	R	0h	Reserved
7-0	UCTBCNTx	RW	0h	The byte counter threshold value is used to set the number of I2C data bytes after which the automatic STOP or the UCSTPIFG should occur. This value is evaluated only if UCASTPx is different from 00. Modify only when UCSWRST = 1.

```
UCB0TBCNT = 0x0005;                // number of bytes to be received
```

（6）设置从机地址。

26.4.14 UCBxI2CSA Register

eUSCI_Bx I2C Slave Address Register

Figure 26-30. UCBxI2CSA Register

15	14	13	12	11	10	9	8
Reserved						I2CSAx	
r-0	r0	r0	r0	r0	r0	rw-0	rw-0
7	6	5	4	3	2	1	0
I2CSAx							
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

Table 26-17. UCBxI2CSA Register Description

Bit	Field	Type	Reset	Description
15-10	Reserved	R	0h	Reserved
9-0	I2CSAx	RW	0h	I2C slave address. The I2CSAx bits contain the slave address of the external device to be addressed by the eUSCIx_B module. It is only used in master mode. The address is right justified. In 7-bit slave addressing mode, bit 6 is the MSB and bits 9-7 are ignored. In 10-bit slave addressing mode, bit 9 is the MSB.

```
UCB0I2CSA = 0x0048;                // Slave address
```

（7）设置中断使能。

Table 26-18. UCBxIE Register Description

Bit	Field	Type	Reset	Description
15	Reserved	R	0h	Reserved
14	UCBIT9IE	RW	0h	Bit position 9 interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
13	UCTXIE3	RW	0h	Transmit interrupt enable 3 0b = Interrupt disabled 1b = Interrupt enabled
12	UCRXIE3	RW	0h	Receive interrupt enable 3 0b = Interrupt disabled 1b = Interrupt enabled
11	UCTXIE2	RW	0h	Transmit interrupt enable 2 0b = Interrupt disabled 1b = Interrupt enabled
10	UCRXIE2	RW	0h	Receive interrupt enable 2 0b = Interrupt disabled 1b = Interrupt enabled
9	UCTXIE1	RW	0h	Transmit interrupt enable 1 0b = Interrupt disabled 1b = Interrupt enabled
8	UCRXIE1	RW	0h	Receive interrupt enable 1 0b = Interrupt disabled 1b = Interrupt enabled
7	UCCLTOIE	RW	0h	Clock low time-out interrupt enable. 0b = Interrupt disabled 1b = Interrupt enabled
6	UCBCNTIE	RW	0h	Byte counter interrupt enable. 0b = Interrupt disabled 1b = Interrupt enabled
5	UCNACKIE	RW	0h	Not-acknowledge interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
4	UCALIE	RW	0h	Arbitration lost interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
3	UCSTPIE	RW	0h	STOP condition interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled

Table 26-18. UCBxIE Register Description (continued)

Bit	Field	Type	Reset	Description
2	UCSTTIE	RW	0h	START condition interrupt enable 0b = Interrupt disabled 1b = Interrupt enabled
1	UCTXIE0	RW	0h	Transmit interrupt enable 0 0b = Interrupt disabled 1b = Interrupt enabled
0	UCRXIE0	RW	0h	Receive interrupt enable 0 0b = Interrupt disabled 1b = Interrupt enabled

```
 UCB0IE |= UCRXIE | UCNACKIE | UCBCNTIE;
```

其中，UCRXIE = 0x0001,UCNACKIE = 0x0020,UCBCNTIE = 0x0040。

2、在配置完USCI_B0后，需要先检测下现有的传输有没有完成。完成后发送开始位进行数据传输。

```
   while (UCB0CTL1 & UCTXSTP);             // Ensure stop condition got sent
   UCB0CTL1 |= UCTXSTT;                    // I2C start condition
```

这里我遇到了一个问题，导致我停顿了一些时间，那就是从机地址与R/W读写位的关系。

Table 26-4. UCBxCTLW0 Register Description (continued)

Bit	Field	Type	Reset	Description
4	UCTR	RW	0h	Transmitter/receiver 0b = Receiver 1b = Transmitter
3	UCTXNACK	RW	0h	Transmit a NACK. UCTXNACK is automatically cleared after a NACK is transmitted. Only for slave receiver mode. 0b = Acknowledge normally 1b = Generate NACK
2	UCTXSTP	RW	0h	Transmit STOP condition in master mode. Ignored in slave mode. In master receiver mode, the STOP condition is preceded by a NACK. UCTXSTP is automatically cleared after STOP is generated. This bit is a don't care, if automatic UCASTPx is different from 01 or 10. 0b = No STOP generated 1b = Generate STOP
1	UCTXSTT	RW	0h	Transmit START condition in master mode. Ignored in slave mode. In master receiver mode, a repeated START condition is preceded by a NACK. UCTXSTT is automatically cleared after START condition and address information is transmitted. Ignored in slave mode. 0b = Do not generate START condition 1b = Generate START condition
0	UCSWRST	RW	1h	Software reset enable. 0b = Disabled. eUSCI_B released for operation. 1b = Enabled. eUSCI_B logic held in reset state.

这里可以知道，MSPFR5969默认是对从机读操作的，所以代码里面没有对这操作，后面的从机code就有说明。

```
UCB0CTLW0 |= UCTR | UCTXSTT;             // I2C TX, start condition
```

我之前用逻辑分析依抓到地址不是0x48,而是0x91，是因为硬件I2C默认会将7位地址左移一位，然后读写位作为最低位。

0x48 = 1001000

0x91 = 10010001 , R/W = 1为读,0为写。

3、这里分析下低功耗模式。

用户手册1.4节有对工作模式的详细说明。里面有这样一幅图：

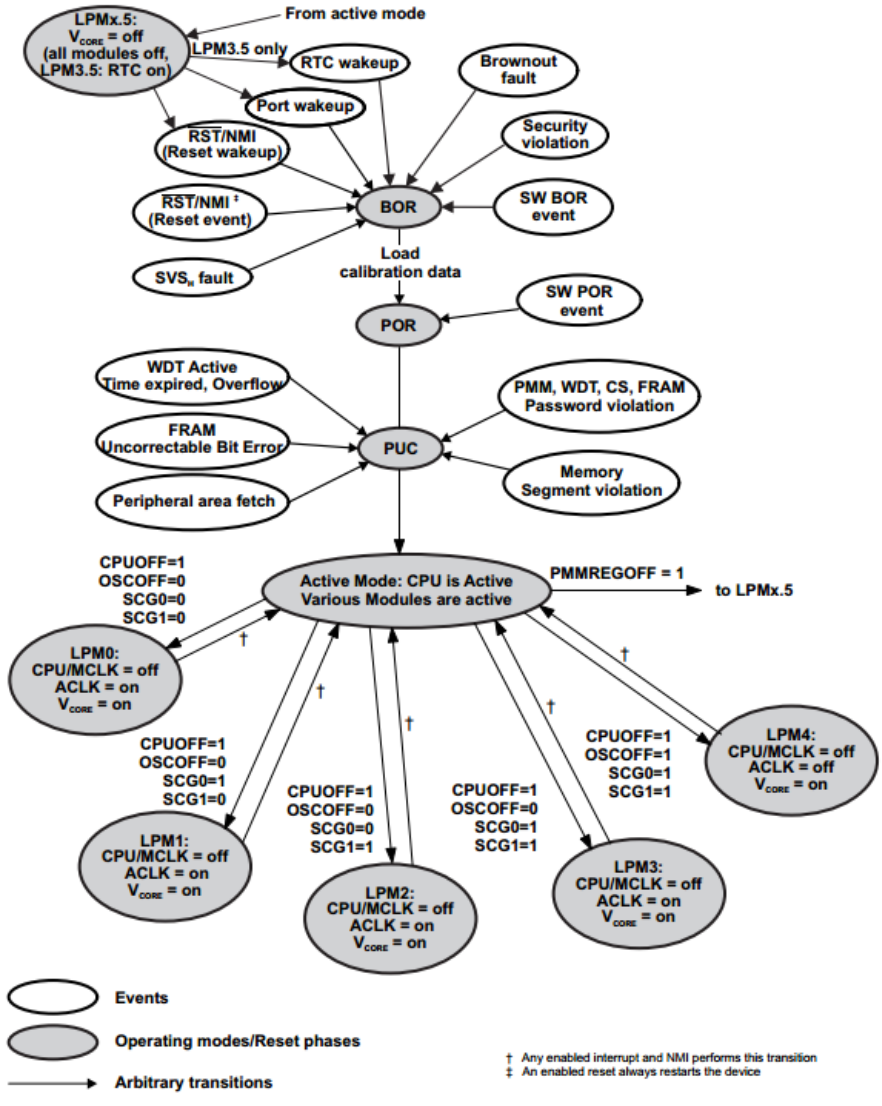

Figure 1-5. Operation Modes

可以很清楚的看到在各个模式下的工作状况。

我将原文的翻译直接摘录一部分：

MSP430系列设计用于超低功耗应用，并使用不同的工作模式，如图1-5所示。

操作模式考虑到三种不同的需求：

•超低功耗

•速度和数据吞吐量
•最小化单个外围电流消耗
LPM0至LPM4的低功耗模式通过SR中的CPUOFF，OSCOFF，SCG0和SCG1位进行配置。在SR中包括CPUOFF，OSCOFF，SCG1和SCG1模式控制位的优点是在中断服务程序期间将当前操作模式保存到堆栈中。如果在中断服务程序期间保存的SR值未被更改，则程序流程将返回到上一个操作模式。程序流程可以通过在中断服务程序中操作堆栈内的保存的SR值来返回到不同的操作模式。当设置任何modecontrol位时，所选的操作模式将立即生效。使用任何禁用时钟的外设将被禁用，直到时钟变为活动状态。外设也可以通过其各自的控制寄存器设置来禁用。所有I / O端口引脚，RAM和寄存器都不变。通过所有使能的中断可以从LPM0到LPM4进行写操作。
进入LPMx.5（LPM3.5或LPM4.5）时，电源管理模块（PMM）的稳压器被禁用。所有RAM和寄存器内容都将丢失。虽然I / O寄存器内容丢失，但是I / O引脚状态被锁定在LPMx.5条目上。有关详细信息，请参见数字I / O章节。通过电源序列，RST事件或特定的I / O可以从LPM4.5唤醒。通过电源序列，RST事件，RTC事件或特定的I / O可以从LPM3.5唤醒。
注意：
TEST / SBWTCK引脚用于通过Spy-Bi-Wire或JTAG调试协议实现外部开发工具与器件的连接。当TEST / SBWTCK为高电平时，通常启用连接。连接启用后，设备进入调试模式。在调试模式下，进入和从低功耗模式唤醒时间可能与正常操作相比。使用连接到开发工具的设备使用低功耗模式时，请注意实时行为！
有关详细信息，请参阅EEM章节。

4.3.3 Status Register (SR)

The 16-bit Status Register (SR, also called R2), used as a source or destination register, can only be used in register mode addressed with word instructions. The remaining combinations of addressing modes are used to support the constant generator. Figure 4-9 shows the SR bits. Do not write 20-bit values to the SR. Unpredictable operation can result.

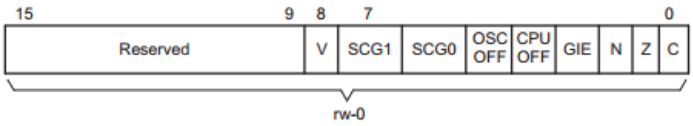

Figure 4-9. SR Bits

Table 4-1 describes the SR bits.

Table 4-1. SR Bit Description

Bit	Description
Reserved	Reserved
V	Overflow. This bit is set when the result of an arithmetic operation overflows the signed-variable range. `ADD(.B), ADDX(.B,.A), ADDC(.B), ADDCX(.B.A), ADDA` — Set when: positive + positive = negative; negative + negative = positive; otherwise reset `SUB(.B), SUBX(.B,.A), SUBC(.B),SUBCX(.B,.A), SUBA, CMP(.B), CMPX(.B,.A), CMPA` — Set when: positive – negative = negative; negative – positive = positive; otherwise reset
SCG1	System clock generator 1. This bit may be used to enable or disable functions in the clock system depending on the device family; for example, DCO bias enable or disable.
SCG0	System clock generator 0. This bit may be used to enable or disable functions in the clock system depending on the device family; for example, FLL enable or disable.
OSCOFF	Oscillator off. When this bit is set, it turns off the LFXT1 crystal oscillator when LFXT1CLK is not used for MCLK or SMCLK. In FRAM devices, CPUOFF must be 1 to disable the cyrstal oscillator.
CPUOFF	CPU off. When this bit is set, it turns off the CPU and requests a low-power mode according to the settings of bits OSCOFF, SCG0, and SCG1.
GIE	General interrupt enable. When this bit is set, it enables maskable interrupts. When it is reset, all maskable interrupts are disabled.
N	Negative. This bit is set when the result of an operation is negative and cleared when the result is positive.
Z	Zero. This bit is set when the result of an operation is 0 and cleared when the result is not 0.
C	Carry. This bit is set when the result of an operation produced a carry and cleared when no carry occurred.

```
__bis_SR_register(LPM0_bits | GIE);     // Enter LPM0 w/ interrupt
```
这条代码用于进入LPM0模式，并开启通用中断使能，允许从中断模式唤醒。
```
__bic_SR_register_on_exit(LPM0_bits); // Exit LPM0
```
这条代码用于退出LPM0模式。
几乎所有的低功耗模式都可以这样调用进入和退出。
4、中断子程序
其实MSP430的中断程序官方给出的例程很完整了，按照你自己的应用改一下就能用，具体的我没有深究，就是按我自己的要求修改了下，都可以正常运行。这里有一点需要注意的就是：

Table 26-3. eUSCI_B Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	UCBxCTLW0	eUSCI_Bx Control Word 0	Read/write	Word	01C1h	Section 26.4.1
00h	UCBxCTL1	eUSCI_Bx Control 1	Read/write	Byte	C1h	
01h	UCBxCTL0	eUSCI_Bx Control 0	Read/write	Byte	01h	
02h	UCBxCTLW1	eUSCI_Bx Control Word 1	Read/write	Word	0000h	Section 26.4.2
06h	UCBxBRW	eUSCI_Bx Bit Rate Control Word	Read/write	Word	0000h	Section 26.4.3
06h	UCBxBR0	eUSCI_Bx Bit Rate Control 0	Read/write	Byte	00h	
07h	UCBxBR1	eUSCI_Bx Bit Rate Control 1	Read/write	Byte	00h	
08h	UCBxSTATW	eUSCI_Bx Status Word	Read	Word	0000h	Section 26.4.4
08h	UCBxSTAT	eUSCI_Bx Status	Read	Byte	00h	
09h	UCBxBCNT	eUSCI_Bx Byte Counter Register	Read	Byte	00h	
0Ah	UCBxTBCNT	eUSCI_Bx Byte Counter Threshold Register	Read/Write	Word	00h	Section 26.4.5
0Ch	UCBxRXBUF	eUSCI_Bx Receive Buffer	Read/write	Word	00h	Section 26.4.6
0Eh	UCBxTXBUF	eUSCI_Bx Transmit Buffer	Read/write	Word	00h	Section 26.4.7
14h	UCBxI2COA0	eUSCI_Bx I2C Own Address 0	Read/write	Word	0000h	Section 26.4.8
16h	UCBxI2COA1	eUSCI_Bx I2C Own Address 1	Read/write	Word	0000h	Section 26.4.9
18h	UCBxI2COA2	eUSCI_Bx I2C Own Address 2	Read/write	Word	0000h	Section 26.4.10
1Ah	UCBxI2COA3	eUSCI_Bx I2C Own Address 3	Read/write	Word	0000h	Section 26.4.11
1Ch	UCBxADDRX	eUSCI_Bx Received Address Register	Read	Word		Section 26.4.12
1Eh	UCBxADDMASK	eUSCI_Bx Address Mask Register	Read/write	Word	03FFh	Section 26.4.13
20h	UCBxI2CSA	eUSCI_Bx I2C Slave Address	Read/write	Word	0000h	Section 26.4.14
2Ah	UCBxIE	eUSCI_Bx Interrupt Enable	Read/write	Word	0000h	Section 26.4.15
2Ch	UCBxIFG	eUSCI_Bx Interrupt Flag	Read/write	Word	0002h	Section 26.4.16
2Eh	UCBxIV	eUSCI_Bx Interrupt Vector	Read	Word	0000h	Section 26.4.17

这里显示MSPFR5969可以设置本机地址，也就是说一个MSPFR5969可以作为四个从机。