Exam 1 – Intro to MicroComputers

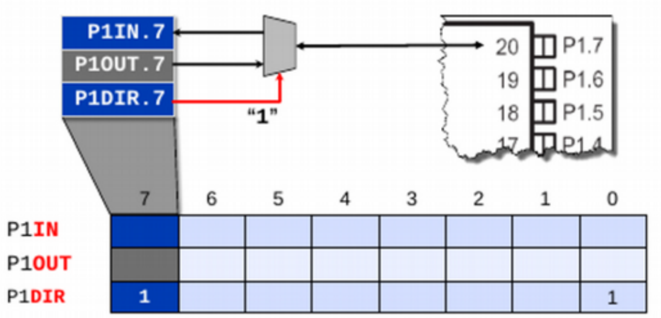
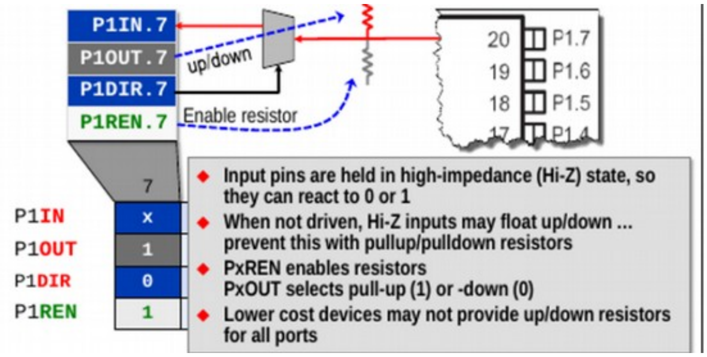


Illustration 1



- Input pins are held in high-impedance (Hi-Z) state, so they can react to 0 or 1
- When not driven, Hi-Z inputs may float up/down ... prevent this with pullup/pulldown resistors
- PxREN enables resistors
  PxOUT selects pull-up (1) or -down (0)
- Lower cost devices may not provide up/down resistors for all ports

Illustration 2

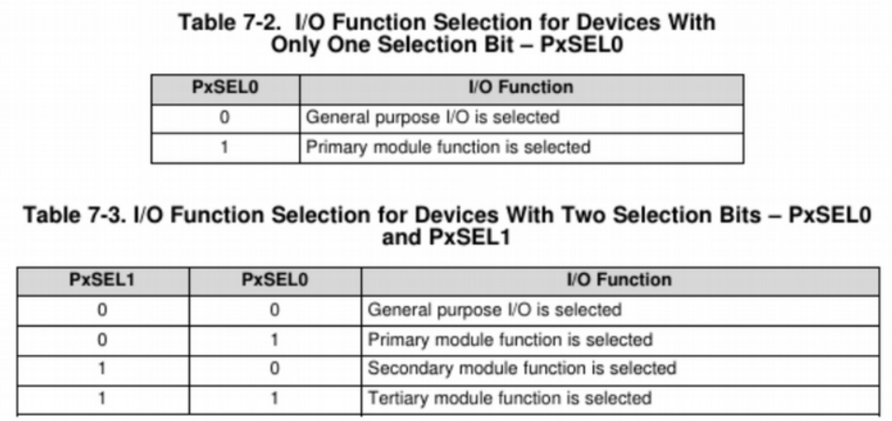


Table 7-2. I/O Function Selection for Devices With Only One Selection Bit – PxSEL0

PxSEL0	I/O Function
0	General purpose I/O is selected
1	Primary module function is selected

Table 7-3. I/O Function Selection for Devices With Two Selection Bits – PxSEL0 and PxSEL1

PxSEL1	PxSEL0	I/O Function
0	0	General purpose I/O is selected
0	1	Primary module function is selected
1	0	Secondary module function is selected
1	1	Tertiary module function is selected

Illustration 3

Exam 1 – Intro to MicroComputers

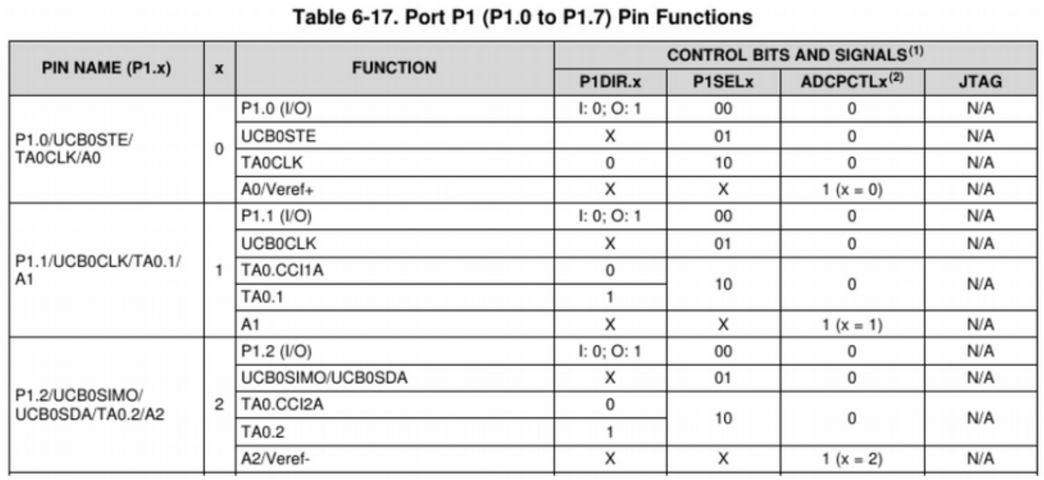
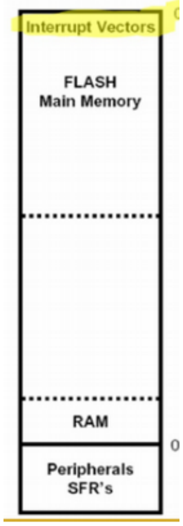


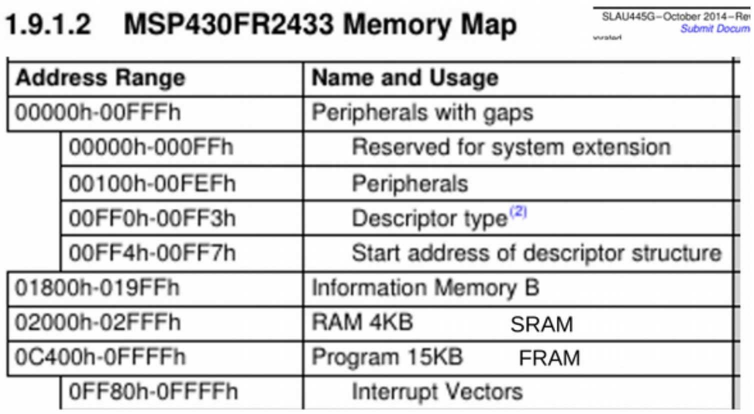
Table 6-17. Port P1 (P1.0 to P1.7) Pin Functions

PIN NAME (P1.x)	x	FUNCTION	CONTROL BITS AND SIGNALS[1]			
			P1DIR.x	P1SELx	ADCPCTLx[2]	JTAG
P1.0/UCB0STE/ TA0CLK/A0	0	P1.0 (I/O)	I: 0; O: 1	00	0	N/A
		UCB0STE	X	01	0	N/A
		TA0CLK	0	10	0	N/A
		A0/Veref+	X	X	1 (x = 0)	N/A
P1.1/UCB0CLK/TA0.1/ A1	1	P1.1 (I/O)	I: 0; O: 1	00	0	N/A
		UCB0CLK	X	01	0	N/A
		TA0.CCI1A	0	10	0	N/A
		TA0.1	1			
		A1	X	X	1 (x = 1)	N/A
P1.2/UCB0SIMO/ UCB0SDA/TA0.2/A2	2	P1.2 (I/O)	I: 0; O: 1	00	0	N/A
		UCB0SIMO/UCB0SDA	X	01	0	N/A
		TA0.CCI2A	0	10	0	N/A
		TA0.2	1			
		A2/Veref-	X	X	1 (x = 2)	N/A

Illustration 4

Interrupts & Memory Map





1.9.1.2 MSP430FR2433 Memory Map

SLAU445G–October 2014–Re
Submit Docum

Address Range	Name and Usage
00000h-00FFFh	Peripherals with gaps
00000h-000FFh	Reserved for system extension
00100h-00FEFh	Peripherals
00FF0h-00FF3h	Descriptor type[2]
00FF4h-00FF7h	Start address of descriptor structure
01800h-019FFh	Information Memory B
02000h-02FFFh	RAM 4KB SRAM
0C400h-0FFFFh	Program 15KB FRAM
0FF80h-0FFFFh	Interrupt Vectors

Illustration 5

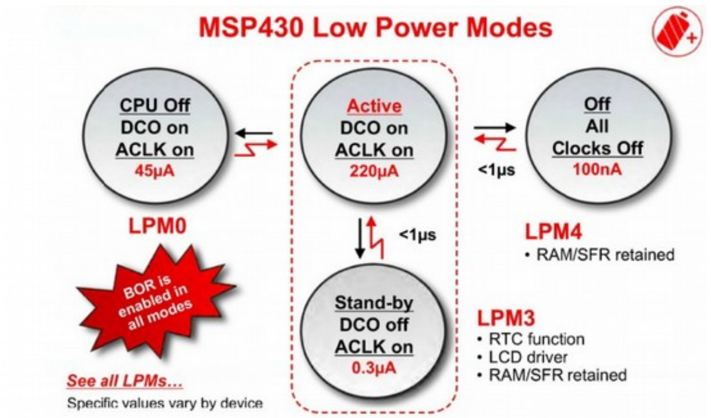


Illustration 6

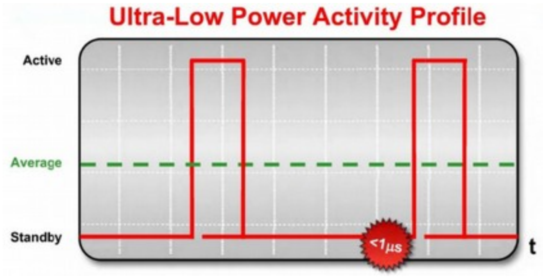


Illustration 7

Waiting for an Event: Family Vacation

Polling

Interrupts

Illustration 8

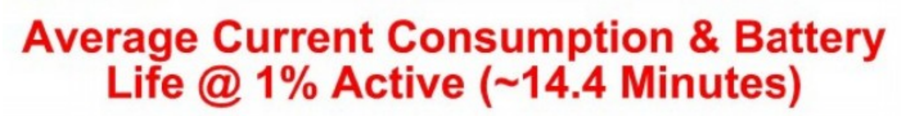
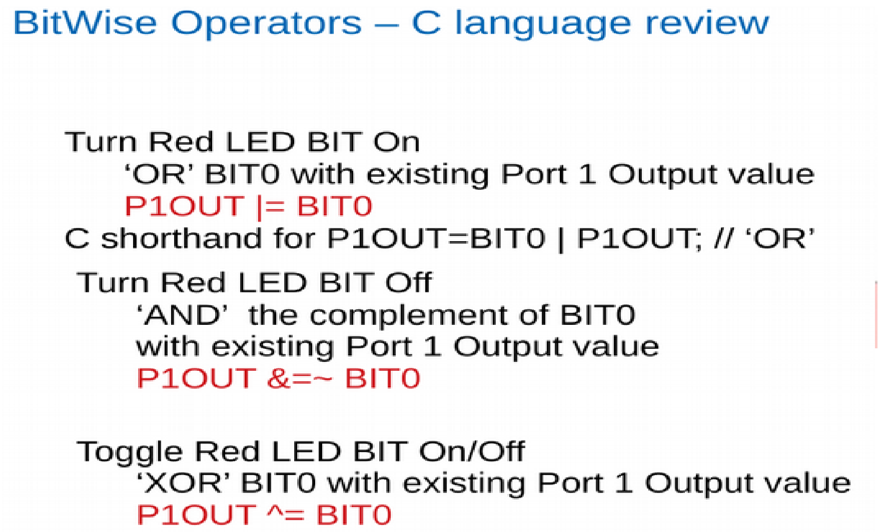

Average Current Consumption & Battery Life @ 1% Active (~14.4 Minutes)

Illustration 9

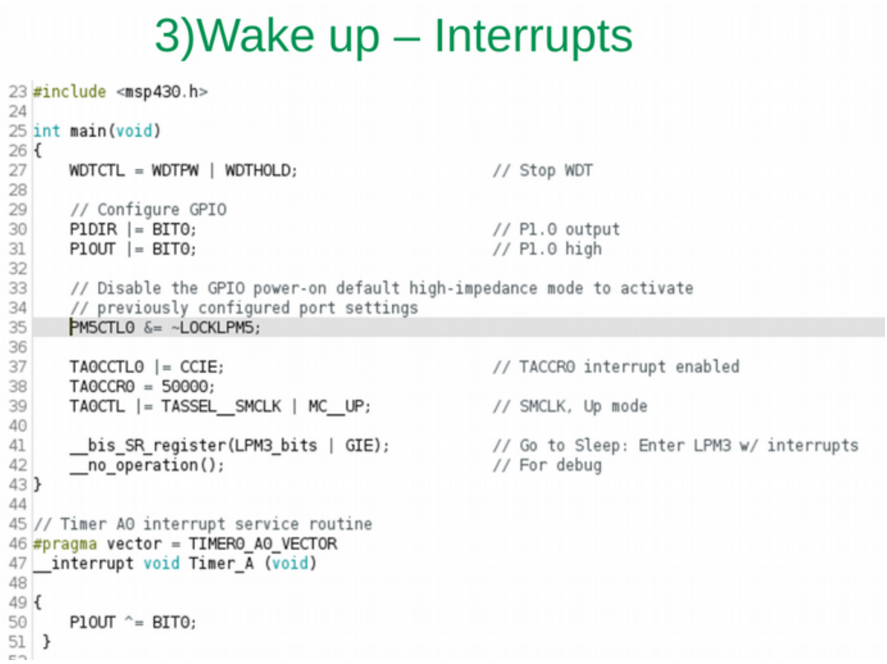

BitWise Operators – C language review

Turn Red LED BIT On
'OR' BIT0 with existing Port 1 Output value
P1OUT |= BIT0
C shorthand for P1OUT=BIT0 | P1OUT; // 'OR'

Turn Red LED BIT Off
'AND' the complement of BIT0
with existing Port 1 Output value
P1OUT &=~ BIT0

Toggle Red LED BIT On/Off
'XOR' BIT0 with existing Port 1 Output value
P1OUT ^= BIT0

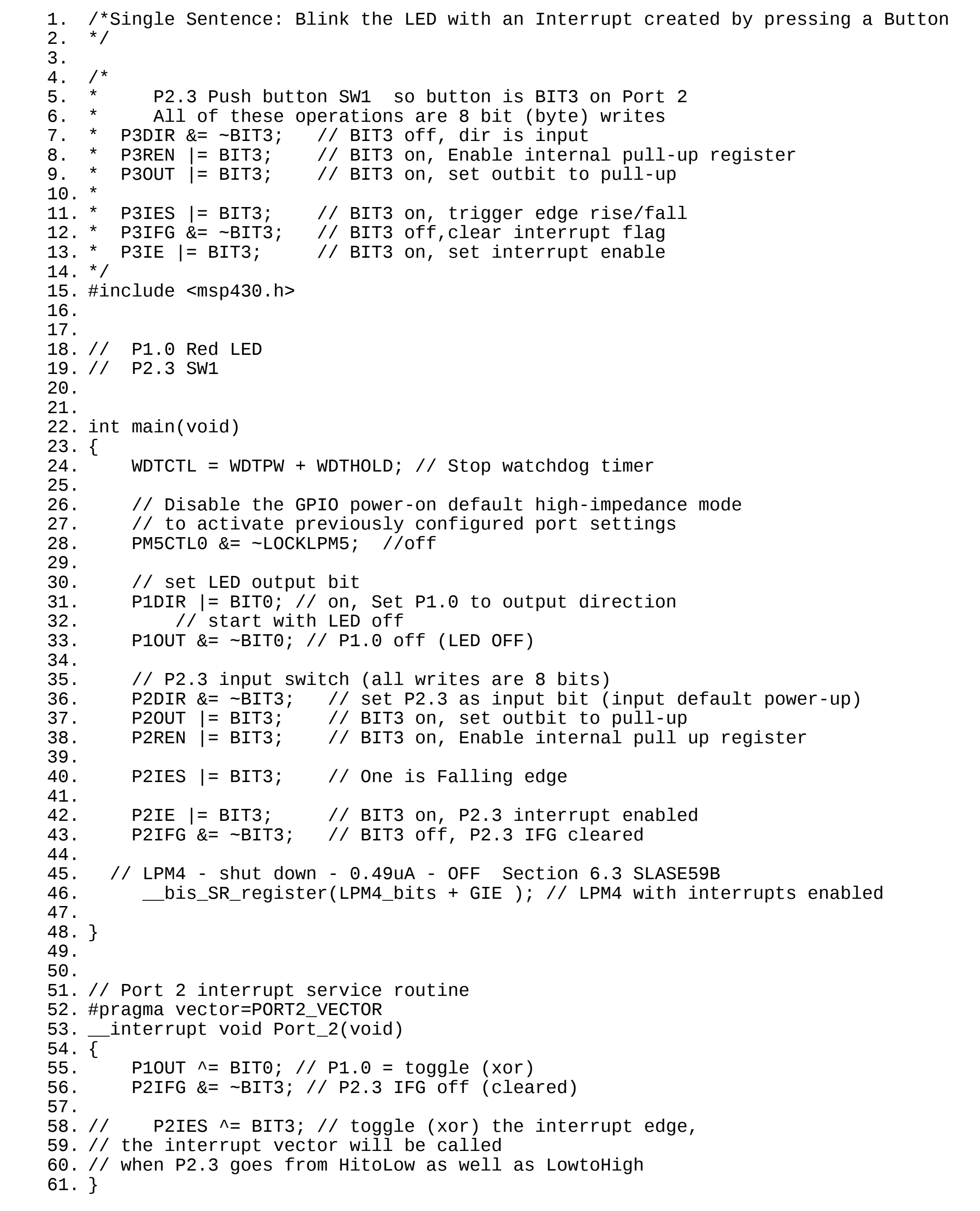
Illustration 10



3)Wake up – Interrupts

```
#include <msp430.h>

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

    // Configure GPIO
    P1DIR |= BIT0;                              // P1.0 output
    P1OUT |= BIT0;                              // P1.0 high

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

    TA0CCTL0 |= CCIE;                           // TACCR0 interrupt enabled
    TA0CCR0 = 50000;
    TA0CTL |= TASSEL__SMCLK | MC__UP;           // SMCLK, Up mode

    __bis_SR_register(LPM3_bits | GIE);         // Go to Sleep: Enter LPM3 w/ interrupts
    __no_operation();                           // For debug
}

// Timer A0 interrupt service routine
#pragma vector = TIMER0_A0_VECTOR
__interrupt void Timer_A (void)

{
    P1OUT ^= BIT0;
 }
```

Illustration 12



```
/*Single Sentence: Blink the LED with an Interrupt created by pressing a Button
*/

/*
*     P2.3 Push button SW1  so button is BIT3 on Port 2
*     All of these operations are 8 bit (byte) writes
*  P3DIR &= ~BIT3;   // BIT3 off, dir is input
*  P3REN |= BIT3;    // BIT3 on, Enable internal pull-up register
*  P3OUT |= BIT3;    // BIT3 on, set outbit to pull-up
*
*  P3IES |= BIT3;    // BIT3 on, trigger edge rise/fall
*  P3IFG &= ~BIT3;   // BIT3 off,clear interrupt flag
*  P3IE |= BIT3;     // BIT3 on, set interrupt enable
*/
#include <msp430.h>


//  P1.0 Red LED
//  P2.3 SW1


int main(void)
{
    WDTCTL = WDTPW + WDTHOLD; // Stop watchdog timer

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

    // set LED output bit
    P1DIR |= BIT0; // on, Set P1.0 to output direction
        // start with LED off
    P1OUT &= ~BIT0; // P1.0 off (LED OFF)

    // P2.3 input switch (all writes are 8 bits)
    P2DIR &= ~BIT3;   // set P2.3 as input bit (input default power-up)
    P2OUT |= BIT3;    // BIT3 on, set outbit to pull-up
    P2REN |= BIT3;    // BIT3 on, Enable internal pull up register

    P2IES |= BIT3;    // One is Falling edge

    P2IE |= BIT3;     // BIT3 on, P2.3 interrupt enabled
    P2IFG &= ~BIT3;   // BIT3 off, P2.3 IFG cleared

  // LPM4 - shut down - 0.49uA - OFF  Section 6.3 SLASE59B
     __bis_SR_register(LPM4_bits + GIE ); // LPM4 with interrupts enabled

}


// Port 2 interrupt service routine
#pragma vector=PORT2_VECTOR
__interrupt void Port_2(void)
{
    P1OUT ^= BIT0; // P1.0 = toggle (xor)
    P2IFG &= ~BIT3; // P2.3 IFG off (cleared)

//    P2IES ^= BIT3; // toggle (xor) the interrupt edge,
// the interrupt vector will be called
// when P2.3 goes from HitoLow as well as LowtoHigh
}
```

Illustration 13