Lab2

Contents

[Ex1 – Clock configuration 1](#_Toc147345862)

[Ex2 – Digital I/O 8](#_Toc147345863)

[Ex3 – Interrupts 10](#_Toc147345864)

[Ex4 – UART 13](#_Toc147345865)

[Ex5 – Timer I 16](#_Toc147345866)

[Ex6 – Timer II 19](#_Toc147345867)

# Ex1 – Clock configuration

1. Setup SMCLK to run on DCO

A screenshot of a computer

Description automatically generated

A diagram of a computer

Description automatically generated

Figure - family guide p. 72

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Configure the DCO to run at 8 MHz

A screenshot of a computer

Description automatically generated

1. Setup SMCLK with a divider of 32

A diagram of a computer

Description automatically generated

A screenshot of a document

Description automatically generated

1. Configure P3.4 as output, set it to output SMCLK

A diagram of a diagram

Description automatically generated

Figure - datasheet pg. 2

A diagram of a circuit

Description automatically generated

Figure - datasheet pg. 81

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

# Ex2 – Digital I/O

1. Configure PJ.0, PJ.1, PJ.2, PJ.3, P3.4, P3.5, P3.6, P3.7 as digital outputs

A screenshot of a computer

Description automatically generated

1. Set the LEDs 1 to 8 to output 10010011

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Value** | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| **LED** | LED8 | LED7 | LED6 | LED5 | LED4 | LED3 | LED2 | LED1 |
| **Reg** | P3.7 | P3.6 | P3.5 | P3.4 | PJ.3 | PJ.2 | PJ.1 | PJ.0 |

A diagram of electrical wiring

Description automatically generated

Figure - board datasheet pg. 17

A screenshot of a computer

Description automatically generated

1. Write a program to blink the LEDs that are 0 in step 2. Use a delay loop in the main infinite loop to visibly blink the LED

**int** **main**(**void**)

{

**int** i;

WDTCTL = WDTPW | WDTHOLD; // stop watchdog timer

// Configure PJ.0, PJ.1, PJ.2, PJ.3, P3.4, P3.5, P3.6, and P3.7 as digital outputs

PJDIR |= BIT3 + BIT2 + BIT1 + BIT0;

P3DIR |= BIT7 + BIT6 + BIT5 + BIT4;

// Set the LEDs 1 to 8 to output 10010011

PJOUT |= BIT1 + BIT0;

PJOUT &= ~(BIT3 + BIT2);

P3OUT |= BIT7 + BIT4;

P3OUT &= ~(BIT6 + BIT5);

**while** (1) {

PJOUT ^= (BIT3 + BIT2);

P3OUT ^= (BIT6 + BIT5);

**for** (i = 0; i < 20000; i++) {

\_NOP();

}

}

**return** 0;

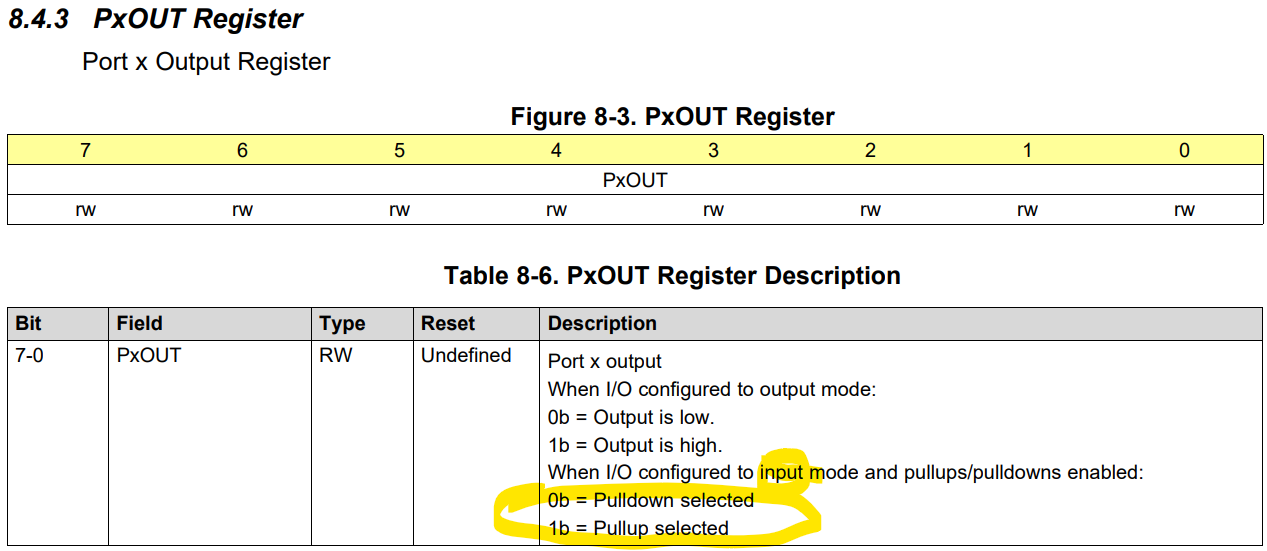
}

# Ex3 – Interrupts

1. Configure P4.0 as digital input (SW1)

A screenshot of a computer

Description automatically generated



Set P4OUT.0 to select pullup

1. Enable internal pull up resistors for the switch S1 connected to P4.0 on the EXP board

A close-up of a register

Description automatically generated

PULLUP: open = 1, close = 0

1. Set P4.0 to get interrupted from a rising edge

Interrupted at the release of S1 switch (button)

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

// enable global interrupt

\_EINT();

1. Configure P3.7 as output (LED8)

// configure P3.7 as output (connected to LED8)

P3DIR |= BIT7;

1. Write an ISR to toggle LED8 when S1 provides a rising edge

A screenshot of a computer

Description automatically generated

A table with text on it

Description automatically generated

Figure - user guide pg. 45

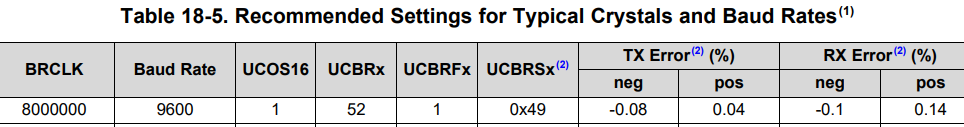
# Ex4 – UART

A diagram of a computer program

Description automatically generated

1. Configure the UART to operate at 9600, 8, N, 1

Baud rate = 9600 bits/s



Data size = 8 bits

Parity = None

# stop bits = 1

1. Set up P2.0 and P2.1 for UART communications

// Configure ports for UART

P2SEL0 &= ~(BIT0 + BIT1);

P2SEL1 |= BIT0 + BIT1;

1. Write a program to periodically transmit the letter ‘a’ to the serial port

while (1)

{

// Periodically transmit an "a" character

while (!(UCA0IFG & UCTXIFG));

UCA0TXBUF = 'a';

for (i=0;i<20000;i++)

\_NOP();

}

1. Check the transmission using the CCS terminal in debug (or PuTTY)

CCS terminal -> COM3

**Serial terminal, COM3, 9600, 8, None, 1, UTF-8**

1. Enable UART receive interrupt. Enable global interrupt

UCA0CTLW0 |= UCSWRST; // Put the UART in software reset

UCA0CTLW0 |= UCSSEL0; // Run the UART using ACLK

UCA0MCTLW = UCOS16 + UCBRF0 + 0x4900; // Baud rate=9600 from 8 MHz clk

UCA0BRW = 52;

UCA0CTLW0 &= ~UCSWRST; // release UART for operation

UCA0IE |= UCRXIE; // Enable UART Rx interrupt

// Global interrupt enable

\_EINT();

1. Set up an interrupt service routine so that when a single byte is received, the same byte is transmitted by back (or echoed) to the serial port. Check using CCS terminal/PuTTY.

#pragma vector = USCI\_A0\_VECTOR

\_\_interrupt void USCI\_A0\_ISR(void)

{

unsigned char RxByte = 0;

RxByte = UCA0RXBUF; // Get the new byte from the Rx buffer

while (!(UCA0IFG & UCTXIFG)); // Wait until the previous Tx is finished

UCA0TXBUF = RxByte; // Echo back the received byte

while (!(UCA0IFG & UCTXIFG)); // Wait until the previous Tx is finished

UCA0TXBUF = RxByte + 1; // Echo back the received byte + 1

if (RxByte == 'j') PJOUT |= BIT0; // Turn on LED1 when a 'j' is received

else if (RxByte == 'k') PJOUT &= ~BIT0; // Turn off LED1 when a 'k' is received

}

1. In addition to echoing, also transmit the next byte in the ASCII table. Check again using PuTTY
2. Add code to turn on LED1 when a ‘j’ is received and turn off LED1 when ‘k’ is received

# Ex5 – Timer I

1. Set up Timer B in the “up count” mode.
2. Configure TB1.1 to produce a 500 Hz square wave. You may need to use frequency dividers when setting up the clock and the timer. Output on P3.4. Verify using an oscilloscope. LED5 should also be lit.

Default clock, TB1CCR0 = 2000 -> 250 MHz (Mode 0)

A drawing of a triangle

Description automatically generated

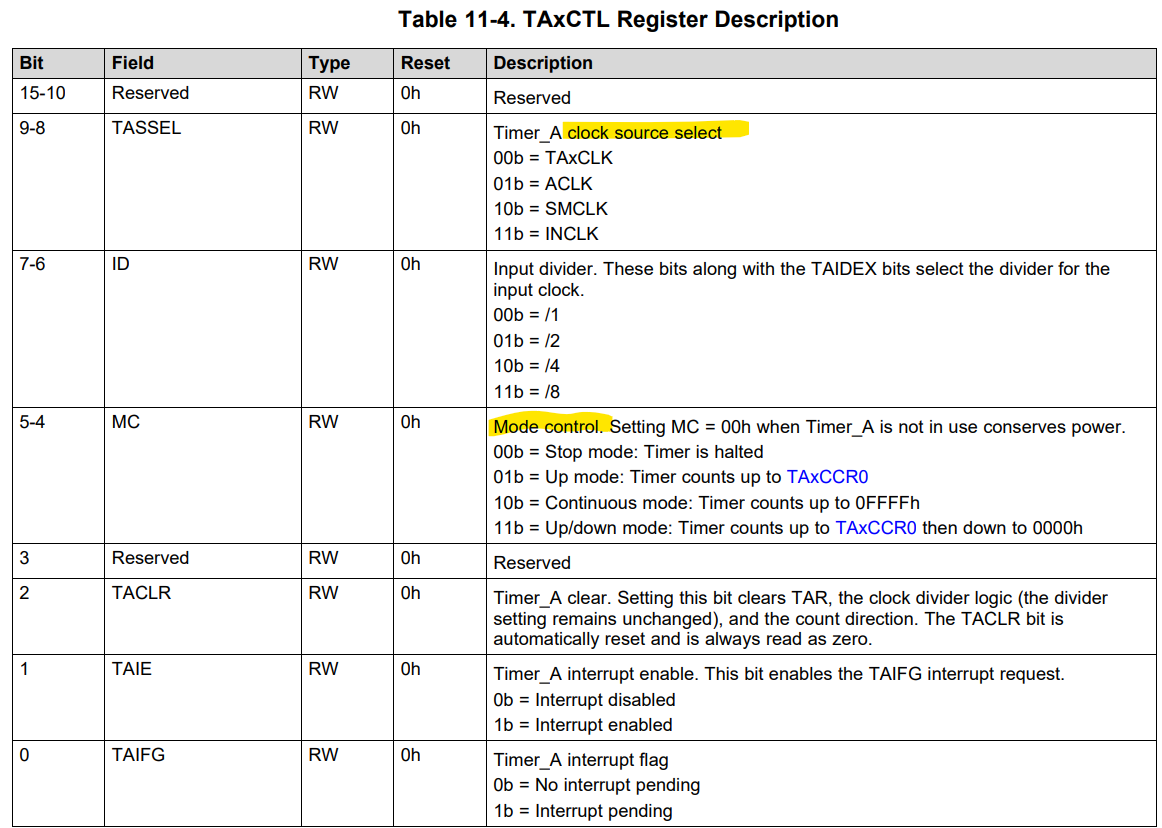
Want 500 MHz -> TB1CCTL1 |= OUTMOD\_7; TB1CCTL2 |= OUTMOD\_7;

TB1CCR1 = 1000; // 50% duty cycle

TB1CCR2 = 500; // 25% duty cycle

A diagram of a output

Description automatically generated



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Figure - device datasheet pg 73

1. Configure TB1.2 to produce a 500 Hz square wave at 25% duty cycle. Output on P3.5. Verify using an oscilloscope. LED6 should be lit. Verify that LED6 is dimmer than LED5.

# Ex6 – Timer II

1. Set up timer A to measure the length of time of a pulse from a rising edge to a falling edge.
2. Connect the timer output from the previous exercise to the input of this timer.
3. Using the debugger to check the measured 16-bit value