

CPE 325: Intro to Embedded Computer System

Lab07

Using Timers and PWM for MSP430FG4618

Submitted by: Caleb Keller

Date of Experiment: 10/12/2022

Report Deadline: 10/12/2022

Demonstration Deadline: 10/13/2022

Introduction

Write a brief discussion on what the lab is about. (Use the tutorial and write in your own words. DO NOT copy text).

In this lab, I am writing C code to perform specific tasks on LED1 on the M30fG618 board. The first part of this project will be implementing a PWM signal to make LED1 “breath”. For the second part, I will be implementing code to use Timer B to trigger the buzzer and LED1 at the same time.

Theory

Write short notes on each topic discussed in lab.

Topic 1: Watchdog Timer

- a.) Performs a controlled-system restart after a software issue arises.
- b.) Works on a time interval--- if timeout occurs a reset occurs
- c.) Watchdog can be used as an interval timer to generate an interrupt after a selected time interval.

Topic 2: Timer A and Timer B

- a.) Timers are used to raise interrupt requests regularly.
- b.) Timers can be used to perform periodic tasks.
- c.) Specific timers have different modes for counting. For example, Timer B supports 4 modes: STOP, UP, Continuous, and UP/DOWN.

Results & Observation

Program 1:

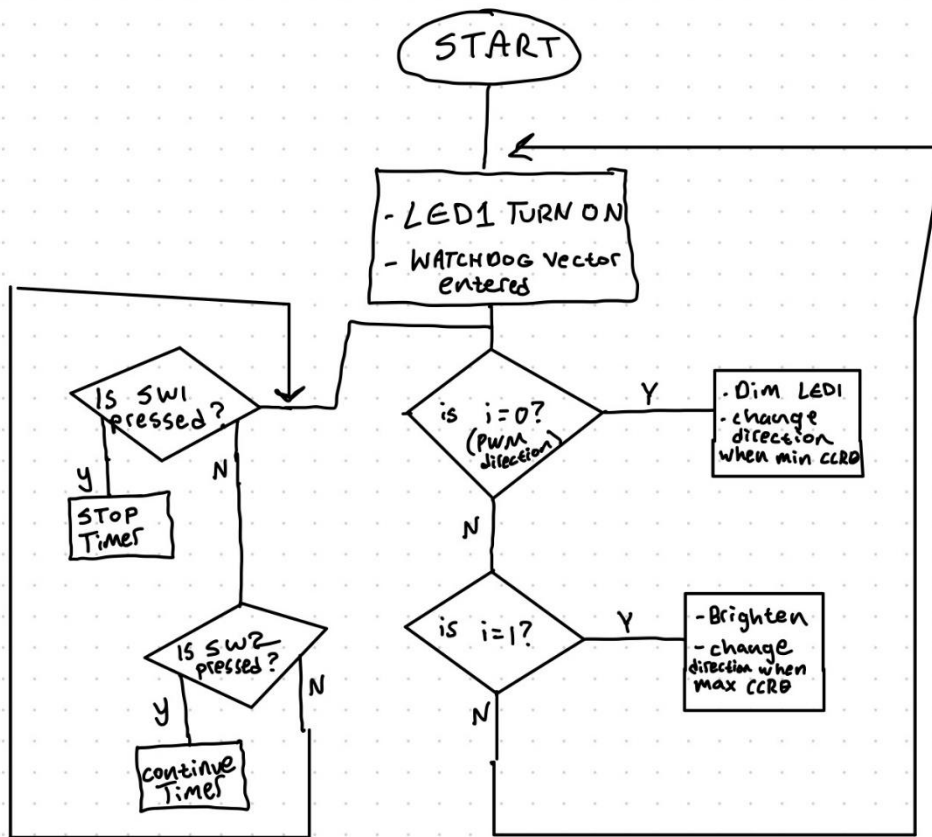
Program Description:

Explain your approach in solving the problem.

To perform the given tasks for this program, I began by writing C code to cause the LED1 “breath” for an interval of 6 seconds. Once the LED reached the CCR0 value (light is brightest) it begins to fade back to off. When specified switches are pressed, the LED either stops breathing or returns back to its breathing period.

Program 1 Flowchart

PART 1

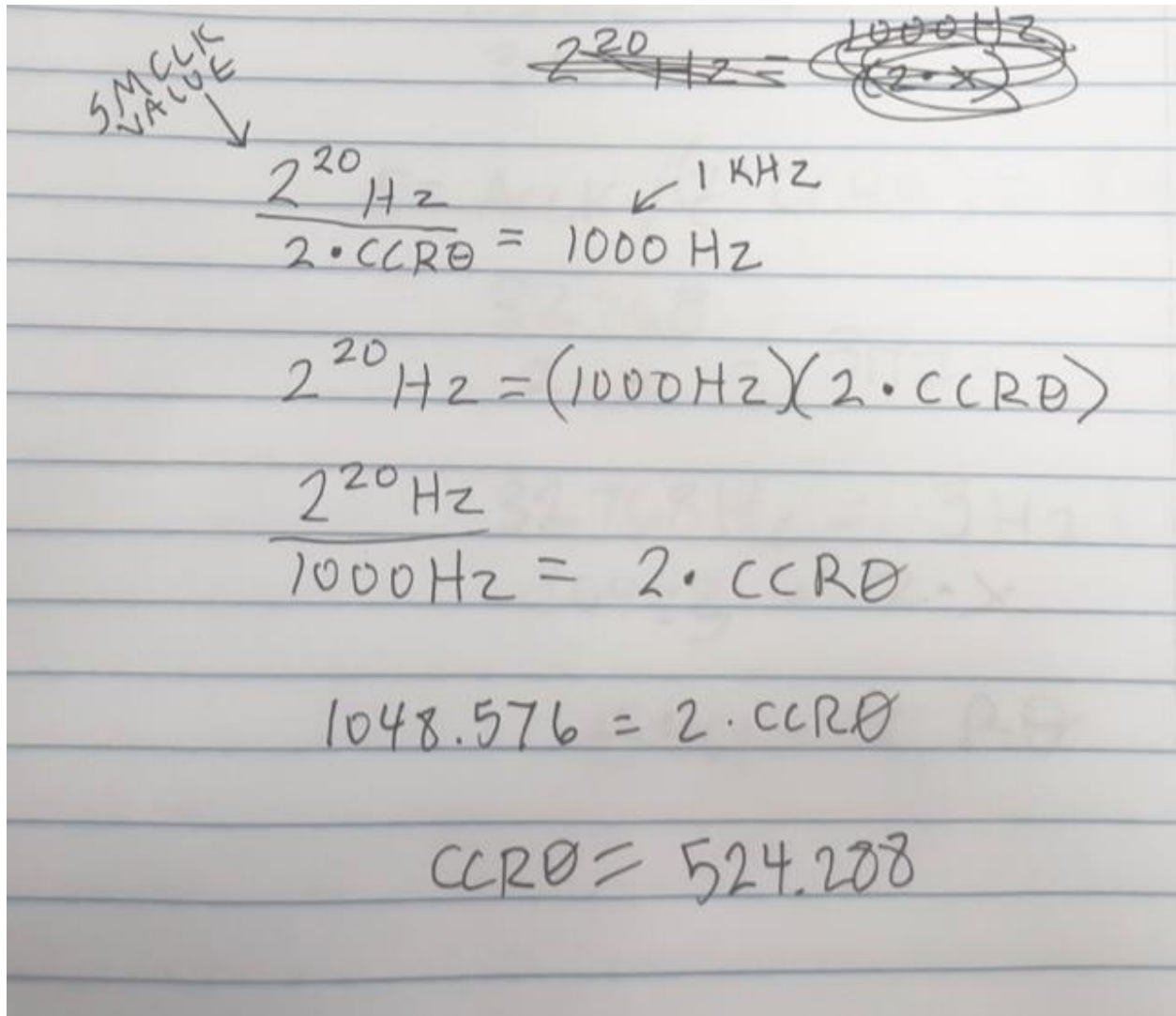


Program 2:

Program Description:

Explain your approach in solving the problem.

In this program, I implemented C code to cause the LED1 and the buzzer to turn on for a timer period of 1 sec ON and 1 sec OFF. The buzzer frequency is set to 1KHz. The buzzer makes a sound alongside the lighting of LED1. I calculated the CCR0 value according to the frequency formula given in the tutorial. (I will attach my method below that I used to calculate the value.)

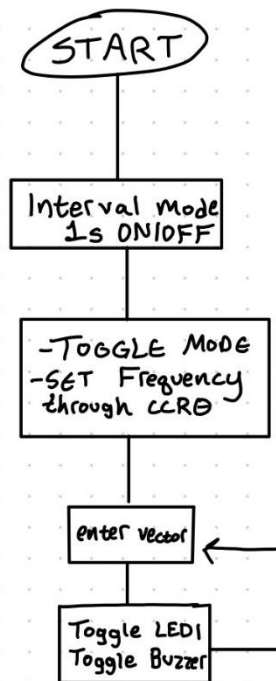


The image shows handwritten calculations on lined paper. At the top left, '5MCLK' is written above 'VALUE' with an arrow pointing down. At the top right, there is a crossed-out equation: $\frac{2^{20}}{2 \cdot 112} = \frac{1000 \text{ Hz}}{(2 \cdot x)}$. Below this, the main calculation starts with $\frac{2^{20} \text{ Hz}}{2 \cdot \text{CCR0}} = 1000 \text{ Hz}$, where '1 KHz' is written above '1000 Hz' with an arrow pointing to it. The next line is $2^{20} \text{ Hz} = (1000 \text{ Hz}) (2 \cdot \text{CCR0})$. This is followed by $\frac{2^{20} \text{ Hz}}{1000 \text{ Hz}} = 2 \cdot \text{CCR0}$. Then, $1048.576 = 2 \cdot \text{CCR0}$. Finally, the result is $\text{CCR0} = 524.288$.

$$\frac{2^{20} \text{ Hz}}{2 \cdot \text{CCR0}} = 1000 \text{ Hz}$$
$$2^{20} \text{ Hz} = (1000 \text{ Hz}) (2 \cdot \text{CCR0})$$
$$\frac{2^{20} \text{ Hz}}{1000 \text{ Hz}} = 2 \cdot \text{CCR0}$$
$$1048.576 = 2 \cdot \text{CCR0}$$
$$\text{CCR0} = 524.288$$

Program 2 Flowchart

PART 2



Conclusion

While working on this program, I ran into many challenges and issues. I was unable to get part 1 of the program to dim the LED. I tried to increment and decrement the CCR1 value to shift the brightness back and forth to no avail. In part 2 of the program, I learned several things about the specialty selections for a port, as well as how to set the sound for the buzzer to be activated.

Appendix

Your first code goes here, if any. Make sure you use a 1X1 table for this.

(Note: Make sure the code is readable, have comments. Also reduce spacing between lines to avoid lengthy reports.

Program 1 source code

```
/*
 * Lab 7 P1
 * Author: Caleb Keller
 * Program Description: LED1 "breathes" for 3 seconds and
 * fades for 3 seconds and is manipulated by the switches
 *
 */

#include <msp430.h>

#define SW1 BIT0&P1IN // SW1
#define SW2 BIT1&P1IN // SW2

static int i = 1; // PWM

int main(void)
{
    WDTCTL = WDT_ADLY_1000; // 1s interval timer
    IE1 = WDTIE; // enable WDT interrupt
    _EINT(); // enable interrupts
    P2DIR |= BIT2; // P2.2 (LED1) set up as output
    P2SEL |= BIT2;
    P2OUT &= ~BIT2; // LED1 is OFF

    P1IE |= BIT0; // P1.0 Interrupt Enabled sw1
    P1IES |= BIT0; // P1.0 h2l edge
    P1IFG &= ~BIT0; // P1.0 IFG cleared

    P1IE |= BIT1; // P1.1 Interrupt Enabled sw2
    P1IES |= BIT1; // P1.1 h2 edge
    P1IFG &= ~BIT1; // P1.1 IFG cleared

    TB0CCTL0 = CCIE; // TB0 output is in toggle mode LED1
    TB0CCTL1 = CCIE;
    TB0CCR0 = 5461; // PWM time period
    TB0CCR1 = 5461;
    TB0CCTL1 = OUTMOD_7; // TB0 output reset/set
```

```

    TB0CTL = TBSSSEL_2 | MC_1; // ACLK is clock source, UP/DOWN mode
    _BIS_SR(LPM0); // enter low power mode 0
}
#pragma vector = WDT_VECTOR
__interrupt void watchdog(void)
{
    if(i == 1) // if LED1 is at 0 low
    {
        TB0CCR0++; // start going back up
        if(TB0CCR1 >= TB0CCR0) // if greater than CCR0 value change direction
        {
            i = 1; // change direction
        }
    }
    if(i == 0) // from top value ready to go back down
    {
        TB0CCR0--; // dimming
        if(TB0CCR1 == 0) // if LED1 reaches 0 and is off
        {
            i = 0; // change direction
        }
    }
}

}
#pragma vector = TIMERB1_VECTOR
__interrupt void timerISR(void)
{
    P2OUT &= ~BIT2;
}
#pragma vector = PORT1_VECTOR
__interrupt void Por1_ISR (void)
{
    if((SW1) == 0 && (SW2) != 0)
    {
        IE1 &= ~WDTIE; // stop watchdog timer
    }
    if((SW1) != 0 && (SW2) == 0)
    {
        IE1 |= WDTIE;
    }
}
}

```

Program 2 Source Code

```

/*
 * Lab 7 P2

```

```

* Author: Caleb Keller
* Program Description: blinks LED1 and sounds the buzzer at the same time
* in 1 second intervals
*
*/

#include <msp430.h>

void main(void)
{
    // watchdog setup
    WDTCTL = WDT_ADLY_1000; // 1s interval timer
    IE1 |= WDTIE; // enable WDT interrupt
    _EINT(); // enable interrupts
    // LED1 setup
    P2DIR |= BIT2; // set P2.2 to output direction
    P2OUT |= BIT2; // LED1 is ON
    // Buzzer setup
    P3DIR |= BIT5; // set P3.5 to output direction for buzzer
    P3SEL |= BIT5; // special function P2.2 output *TB4*
    // timer B setup
    TBCTL = MC_0;
    TBCTL |= TBSSEL_2; // SMCLK as clock
    TBCCTL4 |= OUTMOD_4; // TB output is in toggle mode
    TBCCR0 = 524; // CCR0 calculated by SMCLK(2^20)/(2*CCR0) = frequency(1KHz =
1000Hz)
    TBCTL |= MC_1; // UP mode
    _BIS_SR(LPM0_bits + GIE); // enter low power mode 0
}
#pragma vector = WDT_VECTOR
__interrupt void watchdog_timer(void)
{
    P2OUT ^= BIT2; // toggle LED1
    P3SEL ^= BIT5; // toggle special function (buzzer)
}
/*#pragma vector = TIMERB1_VECTOR
__interrupt void timerISR2(void)
{
    P2OUT ^= BIT2;
    TBCCTL1 &= ~CCIFG;
}
*/

```

Make sure to submit a single pdf file only