

EGR 226: Microcontroller Programming and Applications

Winter 2021

Instructor: Prof. Trevor Ekin

Lab 3: Introduction to MSP432

MSP432 Introduction

Jake Carlson

February 10, 2021

Contents

1. Objectives	4
2. Equipment	4
3. Introduction	4
3.1. Part 1: Exploring the capabilities of the MSP432	4
3.2. Part 2: Using CCS to compile, download, and debug an application	4
3.3. Part 3: Flashing the LED at various rates	4
4. Procedure	5
4.1. Part 1: Exploring the capabilities of the MSP432	5
4.1.1. Discussing the Steps	5
4.1.2. GUI Example	5
4.2. Part 2: First program in Code Composer Studio	5
4.2.1. Discussing the Steps	5
4.2.2. Simple Program Example	5
4.3. Part 3: Modifying Simple Program to Flash LED	6
4.3.1. Discussing the Steps	6
4.3.2. Modified Program Example	6
5. Results/Discussion	6
5.1. Lab Questions	6
6. Conclusion and Future Work	7
Appendices	8
A. Source Code: Part 3 main.c	8

List of Figures

List of Tables

1.	Laboratory Equipment Usage	4
----	--------------------------------------	---

1. Objectives

The objective of this lab was to use Code Composer Studio Development System for the first time and become familiar with the program. This process is also used to install and run firmware using the MSP432 MCU. Another goal of this lab was to explore the use of breakpoints in the debugging process and use the Graphical User Interface for the first time.

2. Equipment

Table 1: Laboratory Equipment Usage

Part	Description	Model	Measured Value	Notes
Code Composer Studio	Texas Instruments programming environment	Version 9.3.0	<i>N/A</i>	<i>N/A</i>
MSP432	Texas Instruments Microcontroller	<i>MSP-EXP432P401R</i>	<i>N/A</i>	<i>N/A</i>

3. Introduction

3.1. Part 1: Exploring the capabilities of the MSP432

In the first section of the lab both the MSP432 Launchpad and the GUI interface were used in order to perform the out of box demo. This part of the lab shows many of the different features of using an embedded system interface.

3.2. Part 2: Using CCS to compile, download, and debug an application

In the second part of the lab TI Code Composer Studio was used to to run a simple program given in the instructions. This program is used to blink an LED light on the MSP432.

3.3. Part 3: Flashing the LED at various rates

In the third part of the lab TI Code Composer Studio was used to modify the given program from part two. The goal of this modification was to produce a delay that can be changed for the LED on and off.

4. Procedure

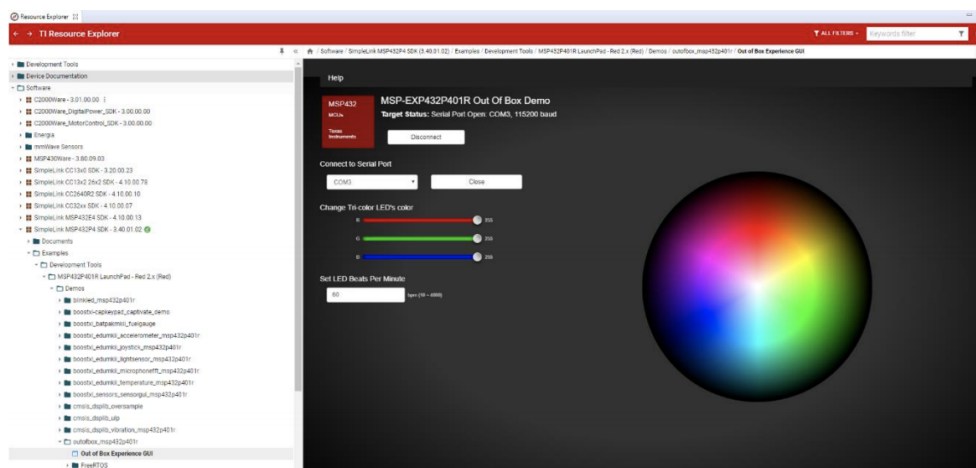
4.1. Part 1: Exploring the capabilities of the MSP432

4.1.1. Discussing the Steps

The first step in Part 1 was to open CCS for the first time and plug in the MSP432 into the computer. After this a program called Out Of The Box Experience was used and in CCS the color of the blinking LED was changed using the Graphical User Interface program.

4.1.2. GUI Example

This example shows the Graphical User Interface and all the different colors the LED can possibly put out.



4.2. Part 2: First program in Code Composer Studio

4.2.1. Discussing the Steps

The first step in Part 2 of the lab was to start a new CCS project including the MSP432 Family and MSP432P401R as the Target. Then in main.c a simple program was copied from the instructions that makes the LED blink. The project was then built and debugged. After this the program will run and the LED on the board blinked.

4.2.2. Simple Program Example

This example shows the simple program that was used in CCS to make the LED blink.

```

1 #include "msp.h"
2
3 void main(void)
4 {
5     // stop watchdog timer (WDT is used to break you out of unwanted loops, more on this later in the semester)
6     WDT_A->CTL = WDT_A_CTL_PW | WDT_A_CTL_HOLD;
7
8     // Configure GPIO for output on P1.0 LED1 on MSP432 Launchpad
9     P1->DIR = BIT0;
10
11     // Temporary variable for loop-maintenance
12     int i;
13     while(1){
14         P1->OUT ^= BIT0; // Toggle LED status
15         for(i=20000; i>0; i--); // Crude delay
16     }
17 }

```

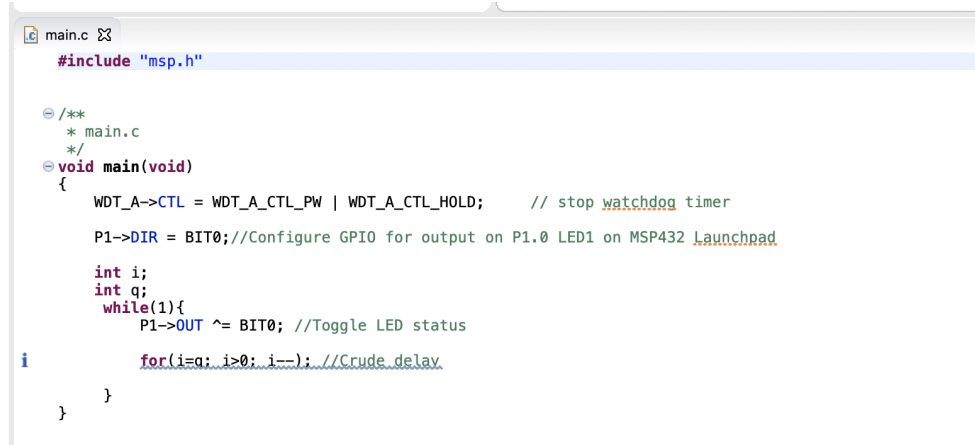
4.3. Part 3: Modifying Simple Program to Flash LED

4.3.1. Discussing the Steps

In Part 3 of the lab the original simple program was modified to make the LED flash after a fixed amount of time. In order to do this a new variable was created and used in the same for loop so that the variable can be changed in the debugger tool to change the blinking LED.

4.3.2. Modified Program Example

This example shows the new for loop using the variable q which can be adjusted in the variable tab of the debugger tool.



```

main.c
#include "msp.h"

/**
 * main.c
 */
void main(void)
{
    WDT_A->CTL = WDT_A_CTL_PW | WDT_A_CTL_HOLD; // stop watchdog timer

    P1->DIR = BIT0; // Configure GPIO for output on P1.0 LED1 on MSP432 Launchpad

    int i;
    int q;
    while(1){
        P1->OUT ^= BIT0; // Toggle LED status
        for(i=q; i>0; i--); // Crude delay
    }
}

```

5. Results/Discussion

5.1. Lab Questions

1. Why do you think TI decided to use an Eclipse foundation for CCS? Describe the major features of the Eclipse IDE platform. What other devices are supported by an Eclipse based IDE?

Ti decided to use an Eclipse foundation for CCS because Eclipse supports multiple languages. It also has unlimited customization and extensions for the MSP432 family. Eclipse also uses an integrated development system and supports numerous devices such as CCS, gumtree, Kalypso, Polyspace, PyDev, Threadsafe, and many more.

2. How is a software breakpoint established in Code Composer Studio? How does this differ from a hardware breakpoint?

A software breakpoint is established by double clicking in front of the line number or the pull down breakpoint tab at the top of the screen. A hardware breakpoint is limited by the device the user is using whereas a software breakpoints are unlimited.

3. What is a watchpoint? How does it differ from a standard breakpoint?

A watchpoint is a special kind of hardware breakpoint used for memory read, write, and read or write. This differs from a standard breakpoint because a standard breakpoint is line specific whereas a watchpoint is event specific.

6. Conclusion and Future Work

In conclusion this lab showcased the endless possibilities that CCS and the MSP432 can achieve. This lab familiarized the user with the new programs needed to succeed and was helpful to set up the MSP432 for the first time. The objectives of this lab were accomplished by running firmware to interface with the LED, exploring the GUI, and learning how to work through Code Composer Studio. A major part of this lab was learning to use the debugging tool to set and adjust breakpoints and exploring the variable tab. Overall, these objectives were completed.

Appendices

A. Source Code: Part 3 main.c

```
#include "msp.h"

/**
 * main.c
 */
void main(void)
{
    WDT_A->CTL = WDT_A_CTL_PW | WDT_A_CTL_HOLD; // stop
    watchdog timer

    P1->DIR = BIT0; //Configure GPIO for output on P1.0 LED1 on
    MSP432 Launchpad

    int i;
    int q;
    while(1){
        P1->OUT ^= BIT0; //Toggle LED status

        for(i=q; i>0; i--); //Crude delay
    }
}
```