Experiment 2: General Purpose Input Output

09.10.2018
Res. Asst. Abdullah Cihan Ak
akab@itu.edu.tr

Сила приносит свободу,
Побеждай и станешь звездой,
А может, обретёшь покой.

Ария , Колизей

1 Introduction

This lab aims to help students to gain more experience in the MSP430 Education Board, MSP430G2553 microcontroller and its assembly language. Students are recommended both to read the supplementary material **"Supplementary_Chapter_6_General_Purpose_IO"** on Ninova. Also, if preferred, they could bring their own computers to the laboratory on which Texas Instruments Code Composer Studio IDE is installed.

2 Part 1

The general purpose input and output (GPIO) using the ports of MSP430G2553 (i.e. Port 1 and 2) can be performed by configuring and reading/setting the corresponding registers of the selected port. The following two instructions read P1.2 and conditionally branch depending on the state of the button.

```
1            ;read the switch at P1.2 and set flags
             bit.b    #00000100b,&P1IN
3            jnz    ON
```

The following two instructions clear and set LED 5 respectively.

```
1            ;read the switch at P1.2 and set flags
             bic.b    #00010000b,&P1OUT;clear P1.4
3            bis.b    #00010000b,&P1OUT;set P1.4
```

Remember; bic and bis instructions are bit clear and bit set instructions respectively. bit instruction is bit test instruction. Note that bic and bis uses masks and they are different than mov instruction. In MSP430 bit test compares two operands and sets status bits.

Note that, this document is written to describe the objectives of the experiment. You are required to know(learn) how to use the MSP430. Examples in the first part are not sufficient enough to successfully complete the experiment. They can only give you some insights. You should read necessary supplementary material before coming to laboratory and in the laboratory.

3 Part 2

Write an assembly program that toggles the LED P2.2 and P2.3 using BUTTON P1.5. LED P2.2 and LED P2.3 should be inverse of each other all the time. If the LED P2.2 is on, it turns off LED P2.2 and turns on LED P2.3 when the BUTTON P1.5 is pressed. If the LED P2.2 is off, it turns on LED P2.2 and turns off LED P2.3 when the BUTTON P1.5 is pressed. If you hold the BUTTON, state of the LED should not keep changing.

4 Part 3

Write an assembly program that counts how many times the push button P2.1 is pressed. In this manner, you should first **"define a variable in memory rather than using an accumulator"**. to store how many times the button is pressed. Your program should display the value of the variable on the Port 1. Note that the value should not continue to increase if you keep the push button pressed. (You should check how to declare a variable can be declared and initialized in the data section. Please check MSP430_Introduction document.)

5 Part 4

In this part you are asked to improve your code and add another functionality to your assembly program in part 3. Utilize another push button (P2.2) that gets the One's Complement of the value shown in Port 1.