Introduction to Programming the MSP430*

adrian valenzuela

This work is produced by The Connexions Project and licensed under the
Creative Commons Attribution License †

Abstract

This is a basic tutorial on how to program the basic digital peripherals on the MSP430 on the MSP430F16x Lite Development Board.

1 Configuring Digital I/O

Digital I/O such as the LEDs and pushbuttons are configured by modifying a several registers pertaining to the port that they are attached to. Check the datasheet to find the respective port number of the peripheral you which to control. For more detailed information about Digital I/O on the MSP430 check **Chapter 9: Digital I/O** of the User's Guide[1] .

First, we must assign the direction of the corresponding I/O pins. This is done by setting the direction register, **PxDIR**, with the appropriate bit. By default, all I/O pins are assigned to be inputs.

- Bit = 0: The port pin is switched to input direction.
- Bit = 1: The port pin is switched to output direction.

On the MSP430F16x Lite Development Board[2] the three LEDs are located on Ports 1 and 2. The port number will correspond the to x value in registers such as **PxIN**, **PxOUT**, or **PxDIR**. Therefore, if we wanted to define the direction of a pin on port 2 we would write to **P2DIR**.

Exercise 1 ***(Solution on p. 3.)***

How do we switch the three pins (P1.7, P2.2, and P2.3) corresponding to the LEDs to be outputs?

Output pins may be toggled using the **PxOUT** register. LEDs are turned on by setting their corresponding register bits low.

Exercise 2 ***(Solution on p. 3.)***

How would be turn on the three LEDs without modifying any other bits in the register?

Since all I/O registers are set as inputs by default we do not have to set the direction of the push buttons. Each time an input is toggled a bit in **PxIN** will be modified.

Exercise 3 ***(Solution on p. 3.)***

Write a couple different polling schemes for detecting if BUTTON_1 was pushed.

*Version 1.4: Jul 7, 2007 10:33 pm -0500

†http://creativecommons.org/licenses/by/1.0

[1] http://focus.ti.com/lit/ug/slau049e/slau049e.pdf

[2] "The MSP430F16x Lite Development Board" <http://cnx.org/content/m12796/latest/>

NOTE: PxIN bits corresponding to the push buttons are high by default (i.e. the button is not depressed.)

Exercise 4 ***(Solution on p. 3.)***

Now we will write a program that lights up one of the LEDs and will light up a different LED once BUTTON_2 is pressed. The LED sequence should go as follows: red, green, yellow, repeat.

Create a new project in CrossStudio and make sure you select the correct processor, the MSP430F169.

Include the correct header file by adding the following line at the top of the main.c file.

```
#include <msp430x16x.h>
```

It may be helpful to define some macros for commonly used register values. For example, if we add `#define red_on ~0x04` to the top of the file (after the `#include`) we may call `red_on` every time we wanted the value ~0x04. Similarly, you may write a function to turn a light on or off.

Complete the program.

Solutions to Exercises in this Module

Solution to Exercise (p. 1)

```
P1DIR |= 0x80;
P2DIR |= 0x0C;
```

0x0C in hex is equivalent to 0b00001100 in binary. Similarly, 0x80 = 0b10000000. You may refer to the module about Binary and Hexadecimal Notation[3] to learn how to do this conversion or you could use the Windows Calculator (**Start -> Run... -> Calc**) to do it much more quickly. We can now easily see that P1.7, P2.2, and P2.3 are set to 1 which makes them outputs.

> NOTE: It is helpful to use `|=` instead of `=` because it won't overwrite bits that are not set to 1 in the mask.

Solution to Exercise (p. 1)

```
P1DIR &= ~0x80;
P2DIR &= ~0x0C;
```

This will turn on all three LEDs assuming they had already been set to be outputs.

> NOTE: ~0x0C = 0xF3 = 0b11110011

Solution to Exercise (p. 1)

`while(!(P2IN&0x01));` or `if (!(P2IN&0x01));`

Solution to Exercise (p. 2)

```
#include <msp430x16x.h>

void main(void)
{
  P1DIR |= 0x80;              // Set P1.7 as an output pin
  P2DIR |= 0x0C;              // Set P2.2 & P2.3 as output pins

  P1OUT |= 0x80;
  P2OUT |= 0x0C;              // Turn off all LEDs

  while(1){
    P1OUT |= 0x80;            // turn off yellow LED
    P2OUT &= ~0x04;            // turn on red LED

    while(P2IN&0x02);         // waits here while button isn't depressed
    while(!(P2IN&0x02));      // waits here while button is pushed in

    P2OUT |= 0x04;            // turn off red LED
```

[3] "The MSP430F16x Lite Development Board" <http://cnx.org/content/m12796/latest/>

```
    P2OUT &= ~0x08;           // turn on green LED

    while(P2IN&0x02);
    while(!(P2IN&0x02));

    P2OUT |= 0x08;            // turn off green LED
    P1OUT &= ~0x80;           // turn on yellow LED

    while((P2IN&0x02));
    while(!(P2IN&0x02));
  }
}
```