ECE2049: Homework 3

Due: Tuesday, 16 June 2020 by 2pm EDT

Submission notes:

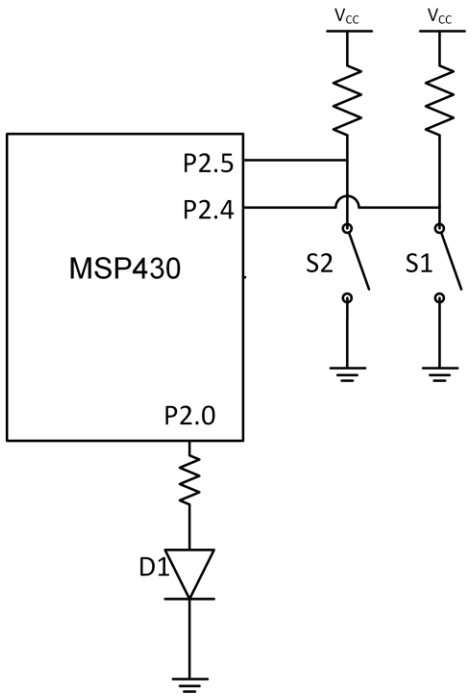
- For full credit, please show your work and denote your answers with a circle or a box.
- Always write or draw your diagrams neatly! We cannot be expected to GUESS what you meant to write! Some problems (such as those involving code) must be typed to be graded—the others may be handwritten (neatly!) or typed.
- Points for each problem are as indicated. Some portions of problems are marked as "BONUS," which count as extra credit.

1. Using our notes from lecture, answer the following questions about memory on the MSP430F5529. Your answers should be at most one or two sentences.

- **a.** The MSP430 uses both RAM and Flash memory. Why does it need both? (In other words, why not just use one or the other?)
- **b.** When you download code from CCS to the MSP430, is the code stored in RAM or Flash? How do you know?
- **c.** True or False: Each peripheral connected to the MSP430 is assigned its own region in the memory map—to interact with these peripherals, the CPU uses special machine code instructions to access each region. **Explain your reasoning.**

(Continued on the next page)

2. (5 pts) In this example, two switches and a blue LED are connected to an MSP430F5529 running the code shown below. Answer the following questions about the configuration.

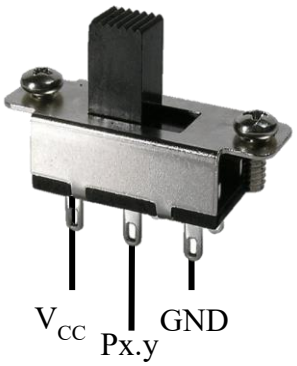



```
#include <msp430.h>
void configure_io(void)
{
    P2SEL &= ~(BIT5|BIT4|BIT0);
    P2DIR &= ~(BIT5|BIT4);
    P2DIR |=  (BIT0);
}

void main(void)
{
    char val, s;

    configure_io();

    while(1) {
        val = P2IN;
        s = (val & 0x60) >> 5;

        if(s > 1) {
            P2OUT &= ~BIT0;
        } else {
            P2OUT |= BIT0;
        }
    }
}
```

a. This configuration example code does not use the P2REN register to enable internal pull-up/pull-down resistors. Does it need to be configured in this case? Explain why or why not.

b. In the example main(), what happens to the LED (on or off) if `val = 0x7A`? What happens if `val = 0x1C`?

3. (10 pts) Say you want to connect four slide-switches like the one shown at the right to P7.1 and P3.5-3. The switches operate such that when the switch is slid to the right, the pin is connected to GND (0V), and when slid to the left is connects to V_{CC} (3.3V).



a. Complete the two functions `config_switches()` and `read_switches()` based on the skeleton below. Your code must be typed—you can find this example as a C file on the course website as switches.c.

```
#include <msp430.h>

// Function prototypes
void config_switches(void);
char read_switches(void);

// Example main() to demonstrate how the functions are used--no need to modify it.
void main(void)
{
      char val;
      WDTCTL = WDTPW + WDTHOLD;   // Stop watchdog timer
      config_switches();

      while(1)
      {
            val = read_switches();
            // Assume something with val happens here...
      }
}

void config_switches(void)
{
      // Configure switches here!
}

// Return a value between 0-Fh corresponding
// to the value of the switches, with the values
// of each switch in the following bit positions:
// MSB                              LSB
// Bits 7-4    Bit 3    2       1     0
// 0           P7.1  P3.5  P3.4    P3.3
char read_switches(void)
{
      char ret_val = 0;

      // Read switches and place the output
      // into the appropriate bit here!

      return ret_val;
}
```

b. Assuming that your program has properly configured the slide switches, what should the function `read_switches()` return given the following register values?
`P7IN = 0x33, P7OUT = 0x42, P3IN = 0x4C, P3OUT = 0xDD`

4. **BONUS (5 pts)**: Say you are you using a different microprocessor that exposes the memory bus so that you can add new memory-mapped peripherals to it. Assume that you attach your peripheral device and that it has one value that the CPU can read at address 0x1104. You may assume that this processor has a 16-bit data bus, like the MSP430.

   How would you write code to read the value at this address? Like all register definitions in C, you can do this with a single `#define` statement. Complete the definition below, which includes an example of how the register should be used.

```
#define MY_REG   (/* Fill in your definition here! */)

void main(void)
{
      int val;
      val = MY_REG; // Read the value of the peripheral at address 0x1104

      // . . .
}
```