1300 Henley Court
Pullman, WA 99163
509.334.6306
www.store.digilentinc.com

Lab 1a: Microprocessor I/O

Revised March 9, 2017
This manual applies to Unit 1, Lab 1a

1 Objectives

1. How to develop a plan for a software-based microprocessor design.
2. Understand the management of basic microprocessor digital inputs and outputs.
3. Become familiar with the circuits included on the Basys MX3 trainer board.

2 Basic Knowledge

1. Fundamentals of digital logic.
2. How to interpret a schematic diagram and electric circuits.
3. How to write a computer program using the C language.
4. How to launch a Microchip MPLAB X project.

3 Equipment List

3.1 Hardware

1. Basys MX3 trainer board
2. Standard USB A to micro-B cable
3. Workstation computer running Windows 10 or higher, MAC OS, or Linux

In addition, we suggest the following instruments:

4. Digilent Analog Discovery 2

3.2 Software

The following programs must be installed on your development workstation:

1. Microchip MPLAB X® v3.35 or higher
2. XC32 Cross Compiler
3. PLIB Peripheral Library

4 Project Takeaways

1. Know how to generate a microprocessor development project using MPLAB X.
2. Know how to configure the Microchip PIC32 processor pins as either digital inputs or outputs.
3. Know how to write a C program for a specific application using an embedded system.

5 Fundamental Concepts

This lab focuses on control of PIC32MX370 I/O pins. It consists of a simple example that demonstrates the syntax, as well as a mini project. The mini project is designed to demonstrate the process for approaching software-based design of an open-ended problem and to apply the design approach given only the performance specifications.

6 Problem Statement

In this lab, you will be using specified switches to turn specified RED, BLUE, and GREEN colors of the tri-color LED8 either on or off using SW0, SW1, and SW2, respectively.

7 Background Information

7.1 Control Registers

In this lab, we will make use of the TRISx, LATx, and PORTx control registers. TRISx registers simply control the data direction flow through port I/O pins, e.g., whether a PORTx I/O pin is an input or an output. If the data direction bit is a "1", then the corresponding PORTx I/O pin is an input, and vice versa.

LATx registers (PORTx data latch) hold data written to port I/O pins. A write to a LATx register latches data to the corresponding port I/O pins and those I/O port pins configured as outputs are updated. A read from a LATx resister reads the data held in the PORTx data latch, not directly from the port I/O pins (this distinction will come into play in Lab 1b).

PORTx registers are the registers that allow I/O pins to be accessed. A write to a PORTx register writes to the corresponding LATx register (PORTx data latch) and those I/O port pins configured as outputs are updated. This is effectively the same as writing to a LATx register. A read from a PORTx register reads the synchronized signal applied to the port I/O pins.

7.2 Configuring the PIC32 for Digital Inputs

As noted from Appendix E Table E.1 in the Unit 1 description, the eight slide switches are connected to various pins on differing I/O ports of the PIC32 processor. This then requires that PIC32 pins connected to the switches must be configured as inputs either individually or in groups if multiple pins are connected to a common PIC32 port. Notice that some of these pins have an "AN" designation in the Alt column of Appendix E Table E.1. These pins are analog capable. To be used as a digital input, port pins that can be used as analog inputs must have the analog input disabled as well as be set as an input by setting the corresponding bit in the TRIS (tristate) register to a 1. To be used as an output, this bit must be set to 0. For example, to program SW5, SW6, and SW7 as digital inputs, the following statements must be included in the hardware initialization. The order of these statements is not critical.

Listing 7.1. Instructions for Setting Analog Inputs to be Digital Inputs for SW5 Through SW7

```
    TRISBbits.TRISB9  = 1; // Set Port B bit 9 for input mode (3.3V tolerant only)
    ANSELBbits.ANSB9  = 0; // Disable analog input
    TRISBbits.TRISB10 = 1; // Set Port B bit 10 for input mode (3.3V tolerant only)
    ANSELBbits.ANSB10 = 0; // Disable analog input
    TRISBbits.TRISB11 = 1; // Set Port B bit 11 for input mode (3.3V tolerant only)
    ANSELBbits.ANSB11 = 0; // Disable analog input
```

If multiple I/O pins on a common processor port are to be simultaneously set, then the code in Listing 7.2 can be used. The bits do not need to be in succession and the ordering of the two statements is not critical. Only the bit positions designated as “1” will be set or cleared and all other bit positions will be unchanged by the “SET” and “CLR” instructions.

Listing 7.2. Instructions for Setting Analog Inputs to be Digital Inputs for SW5 Through SW7

```
TRISBSET =  0b0000111000000000; // Set Port B bits 9 - 11 for input mode (3.3V only)
ANSELBCLR = 0b0000111000000000;   // Disable analog input for Port B bits 9 – 11
```

Note: If the instruction *TRISB = 0b0000111000000000;* is used, all bit positions of the TRISB register will be set according to the bit positions of the 1s and 0s.

If the peripheral library function is used, the statement *PORTSetPinsDigitalIn(IOPORT_B, BIT_9 | BIT_10 | BIT_11);* will both disable the analog input and set the specified TRIS register bits. This instruction is equivalent to the two statements in Listing 7.2.

Other PIC I/O pins that serve solely as digital inputs require only the instruction to set the appropriate TRIS bit. For example, to program the PIC32 to an input for SW0, initialize using the statement *TRISFbits.TRISF3 = 1;*.

Figure A.1 shows that seven of the switches (SW0 through SW6) are isolated from the PIC32 processor I/O pins by a 10K ohm resistor. This means the PIC32 pins could be used as outputs with no ill effect on the operations of the PIC32 processor. Note also that SW7 (PIC32 pin RB9) is connected via a voltage divider to the Analog Discovery 2 Trig2 pin. Therefore, SW7 can be used as a manual trigger for the Analog Discovery 2 by setting RB9 as an input, or used as a software-controlled trigger by setting RB9 as an output and setting SW7 off.

7.3 Configuring the PIC32 for Digital Outputs

Figure A.1 shows the portion of the Basys MX3 schematic for connection to the tri-color LED. The corresponding PIC32 pins connected to the tri-color LED must be configured as outputs. If output pins are analog capable, they must be set for digital mode operation as described above.

8 Lab 1a

8.1 Requirements

This exercise uses the Basys MX3 switches SW0 through SW2 to control the red, blue, and green LEDs on the tri-colored LED labeled LED8 according to the following rules (refer to the schematic diagrams in Appendix A of this lab for additional details):

1. SW0 is to turn the RED color of LED8 on or off.
2. SW1 is to turn the BLUE color of LED8 on or off.
3. SW2 is to turn the GREEN color of LED8 on or off.
4. All switch combinations are allowable.

8.2 Design Phase

This design follows Table 6.1 and the concept maps shown in Figs. 6.1 and 6.2 of the Unit 1 text. Functionally, this design consists of three tasks: hardware initialization, reading the state of slide switches, and setting the outputs to the tri-color LED. After executing the required initialization, the process of reading switches and setting outputs is repeated as fast as the processor can execute the code. As long as all processor inputs and outputs are completed at least once within the infinite loop, the program will meet the above project requirements.

8.3 Construction Phase

1. Write a problem statement that lists the requirements.
2. List the resources that you will be using in this design.
3. Sketch a flow diagram for this program that identifies what is to be done in the initialization section and what will be done in the infinite loop.
4. Write a test plan that you will use to validate your software design. (See Unit 1)
5. From the schematic diagram for the Basys MX3 trainer board:
   a. Determine and record the port and bit numbers assigned to switch outputs for SW0 through SW2.
   b. Determine and record the port and bit number assigned to the RED, GREEN, and BLUE LEDs on the tri-colored LED labeled LED8.
   c. Determine which pins can be set as analog input or digital I/O.
6. Create a new project for the Basys MX3 trainer board. Name this project Lab1a.
7. Copy the “config_bits.h” file to the project folder
8. Add the “config_bits.h” file to the project list of Header Files.
9. Create a new “mainfile” to the project. Name this file “main.c”.
10. Add config_bits.h to the top of main.c.
11. Create a function called main by writing *int main(void) { … }*
12. Within the initialization section of the main function:
    a. Set the PORT D TRIS register to set the bits for LED8_R, LED8_G, and LED8_R as digital outputs.
    b. Set the PORT F TRIS register to set the bits SW0 through SW2 as digital inputs.
13. Write the code inside the infinite loop so that SW0 through SW2 are used to control the ON/OFF state of RED, GREEN, and BLUE LEDS respectively.

8.4 Testing

1. Verify that all LED colors are off when the three switches are off.

2. Verify that the specified colors RED, BLUE, and GREEN individually light on and off using SW0, SW1, and SW2.
3. Verify that multiple colors simultaneously light when switch pairs are on. Record the observed colors of the pair color combinations.
4. Verify that the tri-color LED appears to be bluish white when all three switches are on.

9 Questions

1. Listing 7.1 and Listing 7.2 present two equivalent methods of setting PIC32 pins as inputs.
   a. Do a quick internet search and find out what "magic number" refers to in software programming.
   b. Which listing, 7.1 or 7.2, uses magic numbers?
   c. What are the advantages and disadvantages of using magic numbers?
2. Most of the PIC32 processor outputs have a resistor connected in series with the end circuit or device. Using section 31 of the PIC32MX370 datasheet (see Reference 2 in section 10), answer the following questions relative to output drive capability. Assume that the processor is powered with VDD equal to 3.3V.
   a. What is the maximum source current for processor pins and what are the stipulated conditions?
   b. What is the maximum sink current for the processor pins and what are the stipulated conditions?
   c. What is the maximum output (source) current to each LED0 through LED7?
   d. What is the maximum input (sink) current from each of the 7-segment LEDs? What is the maximum current that can be supplied by the PIC32 to all I/O pins?
3. Using the schematic diagram shown in Fig. A.3 for LED0 through LED7 on the Basys MX3 and Figs. 32-1 presented in Reference 2, when Vdd is 3.3V:
   a. Estimate the output voltage when RA0 pin is set high. Estimate the current output when RA0 pin is set high, assuming that the forward diode voltage drop is 0.7 V.

10 References

1. Best coding practices, https://en.wikipedia.org/wiki/Best_coding_practices
2. PIC32MX330/350/370/430/450/470 Family Data Sheet

Appendix A: Schematic Drawings

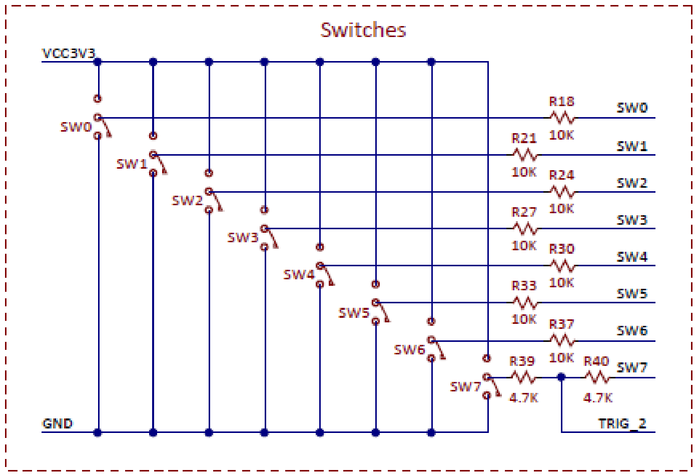

Figure A.1. Slide switch schematic diagram.

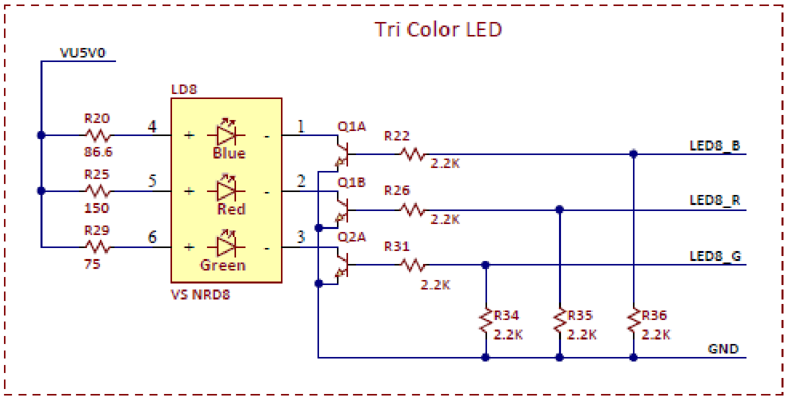

Figure A.1. Schematic of PIC32 interconnection with the Tri Color LED.

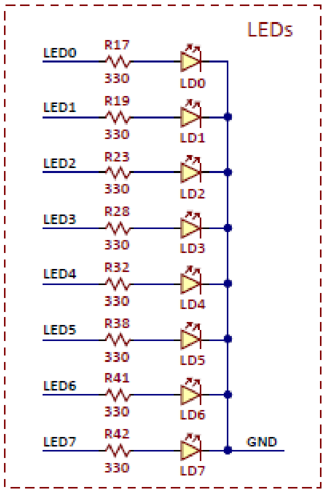

Figure A.3. Basys MX3 LEDs.

Appendix B: Integrated Development Environment (IDE) Setup – Essential Elements of a MPLAB X Project

1. Microchip MPLAB X IDE and XC32 C compiler must be installed on a PC or Linux computer.
2. All applications execute from a MPLAB X project. Instructions on how to create a project can be found at http://www.siriusmicro.com/mplabx-c.html. Although you should follow the general flow of the tutorial, some specifics need to be altered as follows:
    a. The processer type is a "32-bit MCU".
    b. The particular processor is a "PIC32MX370F512L".
    c. The hardware debugging tool is "MCU Alpha One".
    d. The C compiler will be "XC32(v1.xx)".
    e. The developer selects the project name and location as needed.
3. Configure the processor: "Configuration Bits - are a collection of special bits that can only be modified at program time. Configuration bits are "read" during reset and enable or disable hardware features in the microcontroller."[1] Sorting out the proper configurations can be daunting for beginning developers. Hence, the labs associated with the Microchip PIC32MX270F512L microprocessor installed on the Basys MX3 processor board will use a header file called *config_bits.h* that is included only with the file containing the function "main".
4. The software must have a function called "main" that contains an initialization section and a run section.
    a. The initialization section configures executes only once and sets up processor resources for the particular application. Typically, I/O pins are configured to be analog or digital. If the pin is declared as digital, it is configured as input. Other resources such as timers, serial ports, and interrupts are also configured in this section.
    b. The run section must contain an infinite software loop that, under no condition, must ever end. Typically this infinite loop is written as *for(;;){ … } or while(1){ …}*. The embedded application executes from the infinite loop or from interrupts that preempt this loop.

[1] View and Set Configuration Bits, http://microchip.wikidot.com/mplabx:view-and-set-configuration-bits