LIGHT EMITTING DIODES (LEDS)

Revision 04.04.07

____

Class

____

Instructor / Professor

2813 Industrial Lane - Garland, TX 75041

Phone: 972-926-9303
http://www.axman.com
email: sales@axman.com

CONTENTS

1 REQUIREMENTS

1.1 Hardware

To complete this lab, **the following hardware is required**:

- Axiom CML12C32 Development Kit
- Project Board Development Kit
- PC running Windows OS

1.2 Software

The CML12C32 & Project Board used in this experiment comes with all the software needed to complete this project.

There are many additional utilities included on the boards support CD that can make developing your own projects easier. The CD contains example source code, documentation and experiments for all Axiom development boards. You can also download the latest versions of the software and documentation free from our web site at: www.axman.com.

Also included is an integrated development environment, called AxIDE, for communicating with the board (via the serial port) and for reading and writing its flash memory. To complete this Lab, you should have this program installed on a PC running Microsoft Windows (95/98/2000/XP).

NOTE: This lab does not teach you how to use the AxIDE terminal interface or the Serial Monitor program to modify memory and upload programs. It assumes you're already familiar with these procedures. Refer to your board manual for details on installing and using this software, including a tutorial for using AxIDE.

CAUTION

Devices used in this lab are static sensitive and easily damaged by mishandling. Use caution when installing wires and devices onto the board to prevent bending the leads.

Experiments should be laid out in an orderly fashion. Start your lab time with the bench clean and free of metal objects and leave the lab area in a clean condition by picking up loose parts, wires and small objects.

2 GETTING STARTED

This lab project will show you how to add Light Emitting Diodes, LEDS, as output indicators for the microcontroller on your Axiom development board. In this example, four LEDS are used

An LED is a solid state device that, when current is forced though it, will emit a light. A port pin on the microcontroller will output a +5 volt signal high, which is applied to the anode of the LED. The cathode of the LED is connected though a resistor to ground. This resistor is a current limiting resistor for the port which limits the current flow on the port to its rated value, preventing damage by over heating.

LED intensity is depended on current flow. Driving LEDS at a higher intensity requires external drivers rated for the LED being used.

LEDS are good indicators for appliances, machinery, cars, alarms and many other products to indicate power on/off, alarm conditions, etc. They come in several colors such as red, green, yellow, orange and blue. By toggling an LED on and off at different rates, a single LED can be used to indicate many different conditions.

3 LAB PROCEDURE

This lab is arranged in a series of simple steps. Each step should be completed before moving on to the next one, which builds on prior ones. Repeat each step as many times as necessary to become familiar with it. You will find it easier to complete more complex experiments after mastering the simple ones.

As an aid to keeping track of location it's a good idea to mark each step as it's completed, since the experiment will fail if anything is skipped.

3.1 Description

This example uses PORT A on the C32 microcontroller in single chip mode. This is a multiple functional, bi-directional port. Bits 0-3 on PORT A are configured as outputs using DDRA (address $0002) and are used to drive the four LEDS. You see in the C32 microcontroller reference manual that PORT A is located at address $0000. Writing directly to this port will change the level of each pin. Writing a binary one will turn the LED on and writing a binary zero will turn it off.

3.2 Detailed Steps

This section describes how to build the LED project and test it with the monitor running on the CML12C32. In the next section, you'll see how to write a simple program to control the LEDS.

> **NOTE**: To complete these steps you must be familiar with modifying register contents on your board. For example, to write $0F to the DDRA register with the Serial monitor, select <address> then enter data. Type Help for more commands.
>
> You can use a different monitor or debugger if you prefer, such as the GNU GDB.

1. Verify power is NOT applied to development boards.
2. Connect CML-12C32 board MCU_PORT connector to the Project board EVBU connector.
3. Move jumper JP7 on Project Board to position 2 & 3. Install JP6 and ENABLE jumpers.
4. Configure the CML12C32 board jumpers to their default positions (NO_AUTO = On, MEM_EN = disabled).
5. Apply power to the CML-12C32 and Project Board. See Project Board manual.
6. Write $0F to the DDRA register. This configures PORT A bits 0 – 3 as outputs.
7. Write $00 to PORTA then verify all LEDS are off. Writing 0 forces all outputs low, thus removing the drive from all the LEDS.
8. Write $01 to PORTA then verify LED 1 is on. This forces PORTA bit 0 high, thus applying a drive to LED 1.
9. Write $02 to PORTA then verify LED 2 is on. This forces PORTA bit 1 high, thus applying a drive to LED 2.
10. Write $04 to PORTA then verify LED 3 is on. This forces PORTA bit 2 high, thus applying a drive to LED 3.
11. Write $08 to PORTA then verify LED 4 is on. This forces PORTA bit 3 high, thus applying a drive to LED 4.

You see how easy LEDS can be added to and controlled by a microcontroller. By using the four bits of port A as drive for the four LEDS, each LED can be directly set on or off. Any combination of LEDS can be set - all on at once or any combination. Try different combinations yourself.

In product development, you can assign each LED a function. One LED as a "RUN" indicator, the other as a "FAULT" indicator, for example. Another could be a cycle indicator like the "RINSE" indicator on a washing machine.

Most LEDS are not bright enough to view at very strong light levels. Normally a driver is provided between the microcontroller and LEDS. This increases the current, which increases the LED intensity.

4 LED PROGRAM

The previous section described how to control LEDS manually using a debugger. While this method is useful for testing and experimenting, once the hardware is working you'll want to write a software program to control the LEDS.

This section describes how to write such a program in assembly language. The source code is listed at the end of this section. Both source code and assembled executable for this example can also be downloaded from the Axiom web site: **www.axman.com**.

If viewing this on your PC, you can copy and paste the source code below into a text editor (such as notepad) then save and assemble it using AxIDE. Refer to the owner's manual of your board for instructions on creating software and running programs for your development board.

4.1 Program Description

The program first sets the direction register of PORT A to output mode. Using equates LED1, LED2, LED3, and LED4 as a mask, bits 0,1,2 and 3 of the port A data register are cleared. This turns all LEDS off.

Next, mask bit "LED1" is used to set LED1 on. The LED is made visual by calling a delay routine. This delay is long enough for a human to see the LED as being on. The next step uses the same mask bit "LED1" to turn LED1 off. The remaining steps turn LEDS 2, 3 and 4 on and off in the same way.

Finally, the program jumps back to the beginning and repeats until you press RESET or remove power.

4.2 Running the Program

1. Upload the assembled program LEDS3D.S19 into the RAM on your board. This program starts at address $0800, which is internal memory on the CML12C32. The source code for this program is shown below.

2. Execute the program by typing `call 0800` in the terminal and pressing <enter>.

3. The LEDS should begin flashing on and off in sequence.

4.3 LED Program Source Code

```
;  Blinking LEDS
;
;  Hardware - 4 LEDS with cathode's connected by four resistors to ground
;  Led anode's connected to Port A bits 0 thru 3.

PORTA:      equ  $0000     ; port A data
DDRA:       equ  $0002     ; port A direction
LED1:       equ  $01       ; LED 1 select
LED2:       equ  $02       ; LED 2 select
LED3:       equ  $04       ; LED 3 select
LED4:       equ  $08       ; LED 4 select

            org $0800

; Setup port A
MAIN:
            movb  #$0F,DDRA   ; bits 0-3 as outputs
            bclr  PORTA,LED1+LED2+LED3+LED4   ; all bits low
            jsr   DELAY
; LED one
            bset  PORTA,LED1    ; LED one On
            jsr   DELAY
            bclr  PORTA,LED1    ; LED one Off
            jsr   DELAY
; LED two
            bset  PORTA,LED2    ; LED two On
            jsr   DELAY
            bclr  PORTA,LED2    ; LED two Off
            jsr   DELAY
; LED three
            bset  PORTA,LED3    ; LED three On
            jsr   DELAY
            bclr  PORTA,LED3    ; LED three Off
            jsr   DELAY
; LED four
            bset  PORTA,LED4    ; LED four On
            jsr   DELAY
            bclr  PORTA,LED4    ; LED four Off
            jsr   DELAY

            jmp  MAIN           ; start over

; Delay subroutine
DELAY:
            ldab #$0F
DELAYL1:
            ldx  #$ffff
DELAYL:
            dbne  X,DELAYL

            dbne  B,DELAYL1
            rts
```

5 QUIZ

Answer the following questions based on the example presented in this lab.

1. Where is the program LEDS3D.S19 located in memory?
A. External memory **B**. Internal memory **C**.Eprom **D**. Rom

2. Writing $04 to PORTA, turns which LED on?
A. LED1 **B**. LED2 **C**. LED3 **D**. LED4

3. What causes an LED to emit light?
A. Current **B**. Resistor **C**. Voltage **D**. Diode

4. Is PORT A?
A. Input **B**. Output **C**. Bi-Directional **D**. All of these

5. How many LEDS can be on at one time?
A. 1 **B**. 2 **C**. 3 **D**. 4

BONUS QUESTION: What is DDRA?
A. Data Register **B**. Program Counter **C**. Direction Register **D**. Timer