|  |  |  |
| --- | --- | --- |
| **EXP NO: 7** | **STUDY OF MPLAB X IDE** | **DATE:** |

**AIM:**

To study the PIC MPLAB X IDE and its tutorial.

**INTRODUCTION:**

MPLAB X IDE is a software program that is used to develop applications for Microchip microcontrollers and digital signal controllers. This development tool is called an Integrated Development Environment, or IDE, because it provides a single integrated “environment” to develop code for embedded microcontrollers.

**AN OVERVIEW OF EMBEDDED SYSTEMS:**

An embedded system is typically a design that uses the power of a small microcontroller, like the Microchip PIC microcontroller (MCU) or PIC digital signal controller (DSC). These microcontrollers combine a microprocessor unit (like the CPU in a personal computer) with some additional circuits called peripherals, plus some additional circuits, on the same chip to make a small control module requiring few other external devices. This single device can then be embedded into other electronic and mechanical devices for low-cost digital control.

**COMPONENTS OF THE MICROCONTROLLER:**

The PIC MCU has on-chip program memory in figure 1 for the firmware, or coded instructions, to run a program. A Program Counter (PC) is used to address program memory, including Reset and interrupt addresses. A hardware stack is used with call and return instructions in code, so it works with, but is not part of, program memory. Device data sheets describe the details of program memory operation, vectors and the stack.

**REGISTERS:**

The microcontroller also has data or “file register” memory. This memory consists of Special Function Registers (SFRs) and General Purpose Registers (GPRs) as shown in figure 3. SFRs are registers used by the CPU and peripheral functions for controlling the desired operation of the device. GPRs are for storage of variables that the program will need for computation or temporary storage. Some microcontrollers have additional data EEPROM memory. As with program memory, device data sheets describe the details of data memory use and operation.

**PERIPHERALS:**

In addition to memory, the microcontroller has a number of peripheral device circuits on the same chip in figure 3. Some peripheral devices are called input/output (I/O) ports. I/O ports are pins on the microcontroller that can be used as outputs and driven high or low to send signals, blink lights, drive speakers – just about anything that can be sent through a wire. Often these pins are bidirectional and can also be configured as inputs, allowing the program to respond to an external switch, a sensor, or to communicate with some external device.

To design such a system, choose which peripherals are necessary for the application. The following is a list of common peripherals:

• Analog-to-Digital Converters (ADCs) allow microcontrollers to connect to sensors and receive changing voltage levels.

• Serial communication peripherals that allow streaming communications over a few wires to another microcontroller, to a local network, or to the Internet.

• Peripherals on the PIC MCU called “timers” accurately measure signal events and generate and capture communications signals, produce precise waveforms, even automatically reset the microcontroller if it gets “hung” or lost due to a power glitch or hardware malfunction.

• Other peripherals detect when the external power is dipping to dangerous levels, so that the microcontroller can store critical information and safely shut down before power is completely lost.

**COMPONENTS OF MPLAB X IDE:**

MPLAB X IDE includes:

• a full-featured programmer’s text editor that also serves as a window into the debugger.

• a project manager (visible as the Projects window) that provides integration and communication between the IDE and the language tools.

• a number of assembler/linker suites for the development of firmware for your project’s device.

• a debugger engine that provides breakpoints, single stepping, Watches windows and all the features of a modern debugger. The debugger works in conjunction with debug tools, both software and hardware.

• a software simulator for all PIC MCU and PIC DSC devices. The simulator is actually composed of several device-specific simulator executables. MPLAB X IDE decides which one to use based on your project’s device.

**CREATING AND SETTING UP A PROJECT:**

The following information discusses setting up a project in MPLAB X IDE, which is required to develop your application code.

Create a New Project MPLAB X IDE is project-based, so you must set up a project to work on your application. Create a new project by performing either of the following actions:

• On the Start page, click the Learn & Discover tab, “Dive In” section, and the “Create New Project” link

• From the menu bar on top of the window, click File>New Project. The New Project Wizard will launch to guide you through creating a new project.

**STEP 1** **Choose a project category:**

For this tutorial, choose “Microchip Embedded”. Secondly, choose a project type. For this tutorial, choose “Standalone Project”. Click Next> to move to the next dialog. Figure 4.

**STEP 2** **Choosing your device**:

In this case **PIC16F877A**. When you are done, click Next>. Figure 5.

**STEP 3** **Select the tool:**

Tool support for the selected device is signified by the coloured circles (lights) in front of the tool name. If you cannot see the colours, mouse over a light to pop up text about support. For some tools, there are two lights next to the tool name, where the first light is the left-most light and the second light is to the right of the first. For the hardware tools, you will notice that a serial number (SN) is specified below any tool that is connected to your computer. This allows you to select from several connected hardware tools. Select your tool and then click Next>. Figure 6.

**STEP 4** **It appears only:**

if MPLAB REAL ICE in-circuit emulator is selected as the tool. For the MPLAB REAL ICE in-circuit emulator, you may specify a plug-in board to use. A plug-in board is the circuit board that is inserted into the emulator’s driver board slot.

**STEP 5 Select the language tool:**

Either a C compiler or assembler. The version and installation location of a language tool is displayed beneath that tool. This allows you to select from several installed language tools. Select your tool and then click Next>.Figure 7.

**STEP 6** **Select the project:**

name, location and other project options. Enter the project name MyProject. By default, projects will be placed in:

C:\Documents and Settings\UserName\MPLABXProject

If the Project Location does not point to the default location, browse to the appropriate location. Check “Set as main project” to make this your main project. Leave “Use project location as project folder” unchecked. When you are done, select Finish to complete new project creation. Figure 8.

**ADD A NEW FILE TO THE PROJECT:**

For this, you will create a new file containing the code you need to compile and execute.

1. In the Projects window, right click on the Source Files folder of the project tree and select New>C Main File.

2. In the “New C Main File” window, name the file and then click Finish. You should now see the file listed under Source Files, and a new Editor window with the file name on the tab.

3. Delete the code in the Editor window and write the code.

**RUNNING AND DEBUGGING CODE:**

The following information discusses using the MPLAB X IDE to run or debug your code.

Build a Project:

For MPLAB X IDE, it is not necessary to build the project first and then run or debug. Building is part of the run and debug processes. For initial development or major changes, however, you may want to make sure that the project builds before attempting to run or debug.

To build a project:

• In the Projects window, right click on the project name and select “Build”. You may also select “Clean and Build” to remove intermediary files before building.

• Click on the “Build Project” or “Clean and Build Project” toolbar icon. Build progress will be visible in the Output window, located in the Task Pane. For this tutorial, the code should build successfully. Figure 9.

**RUN CODE:**

After the code builds successfully, you can attempt to run the application. Click on the “Make and Program Device Project” icon (or select Run>Run Project) to run your program. The lights on the demo board should be flickering.

Run progress will be visible in the Output window as well. Use the Hold in Reset button to toggle between device Reset and running. You can add a “Run Project” icon to the toolbar if you wish (View>Toolbars>Customize).

**DEBUG RUN CODE:**

For this, the code used has been tested and runs successfully. However, your own code may need to be debugged as you develop your application.

To Debug Run the tutorial code, click on the “Debug Project” icon (or select Debug>Debug Project or Debug>Step Into) to begin a debug session. Debug Run progress will be visible in the Output window.

To halt your application code: Click on the “Pause” icon (or select Debug>Pause) to halt your program execution.

To run your code again: Click on the “Continue” icon (or select Debug>Continue) to start your program execution again.

To end execution of your code: Click on the “Finish Debugger Session” icon (or select Debug>Finish Debugger Session) to end your program execution.

**SAMPLE PROGRAM:**

**The LED Flash Program:**

#include<pic16f877x.h>  
void delay(int);  
sbit a=PB^2;  
sbit b=PB^3;  
sbit c=PB^4;  
sbit d=PB^5;  
void main()  
{

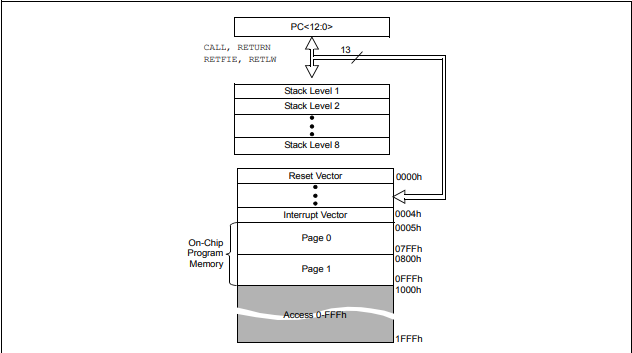
TRISB=0x00;  
a=b=c=d=0x00;  
delay(10);  
a=b=c=d=0xFF;}  
void delay(int a )  
{  
unsigned char c;  
for(c=0;c<a;c++)  
for(c=0;c<250;c++);  
}

**OUTPUT:**

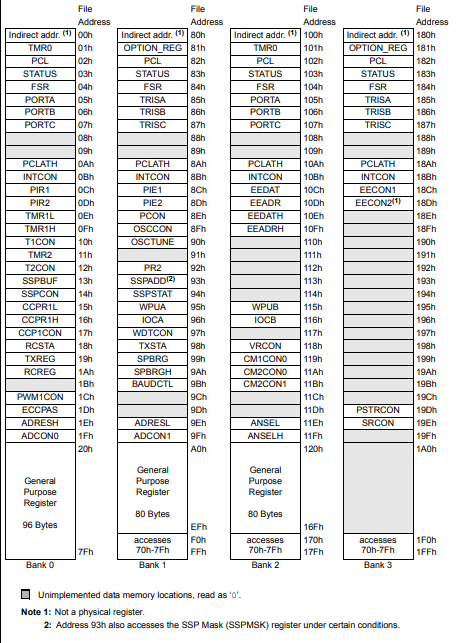
The microcontroller generates the output logic pulses so that the LED light is switched ON and OFF at certain intervals.

**RESULT:**

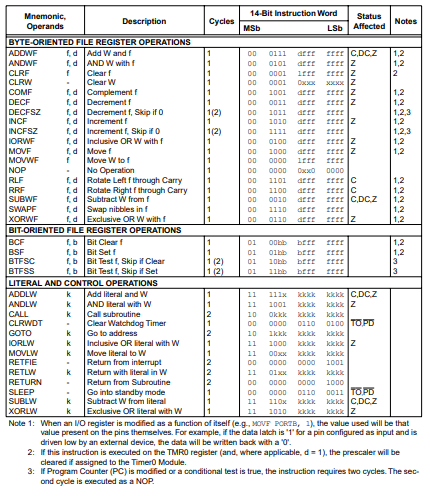
Thus, the PIC microcontroller MPLAB X IDE and tutorial is studied and sample program is executed successfully.

****

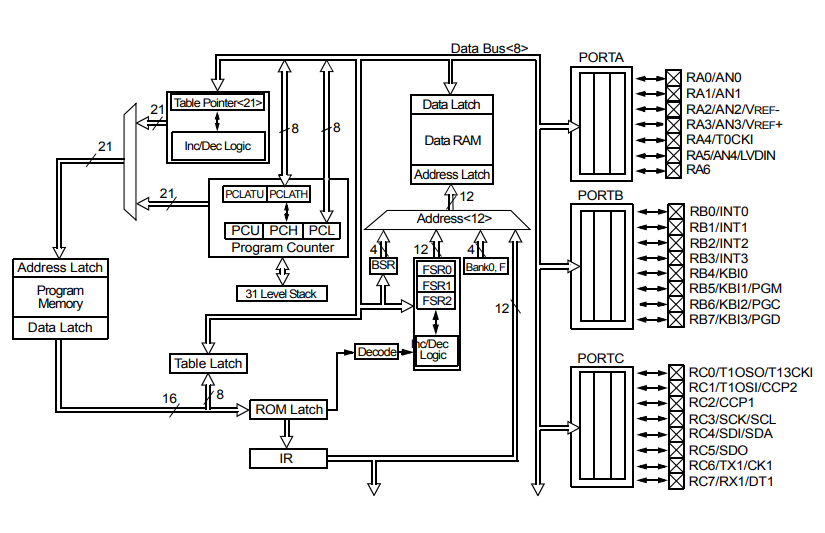
**Figure 1**

****

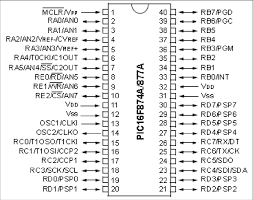
**Figure 2(a)**

****

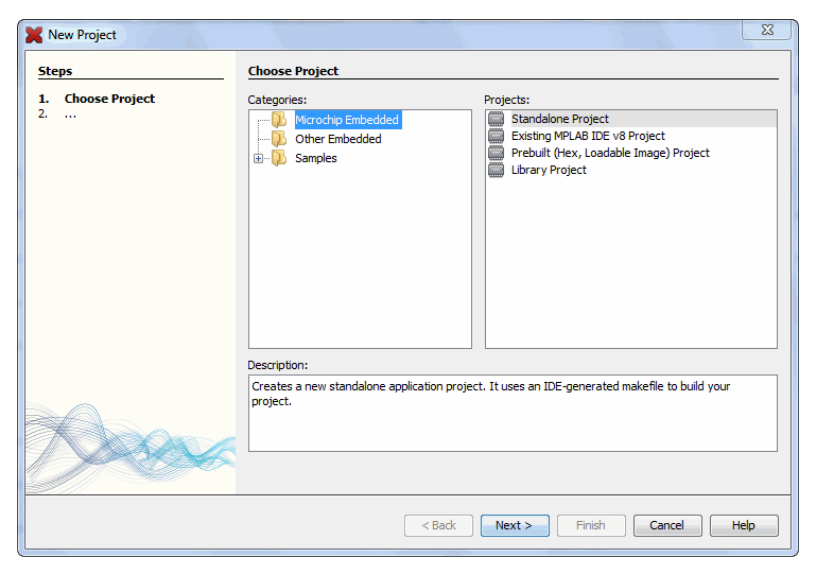
**Figure 2(b)**

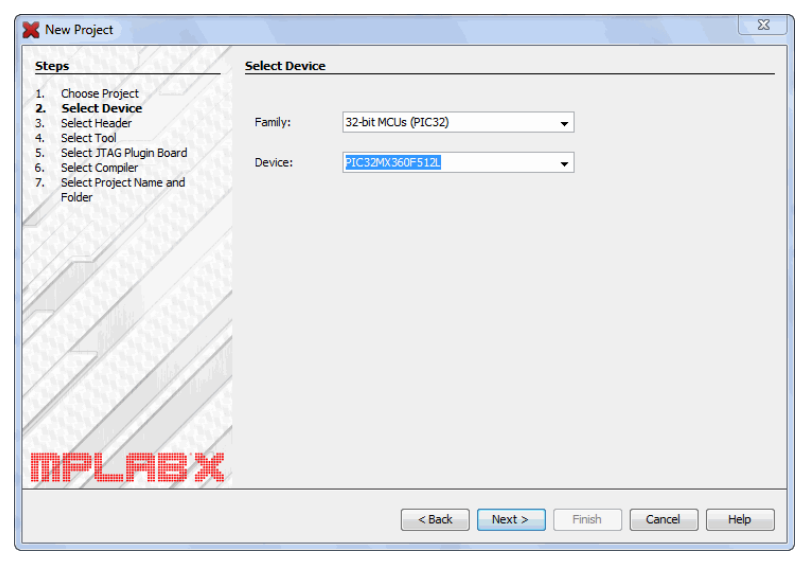
****

**Figure 3(a)**

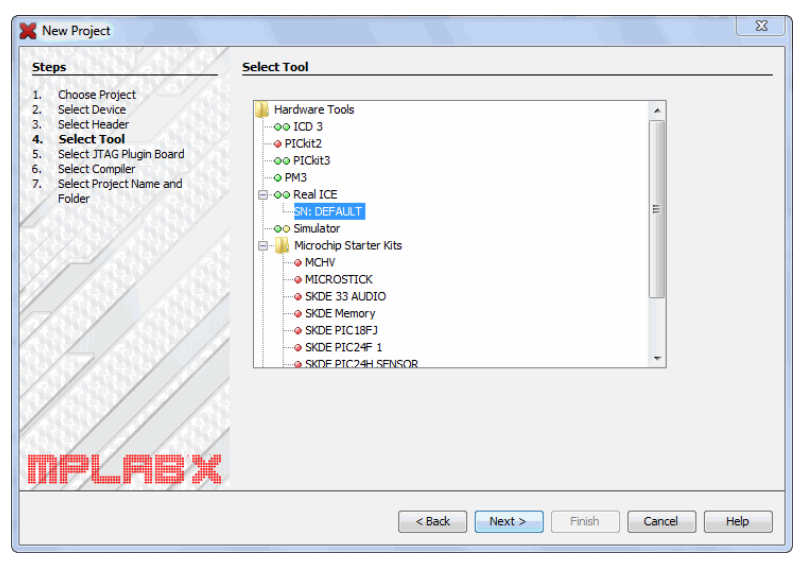
****

**Figure 3(b)**

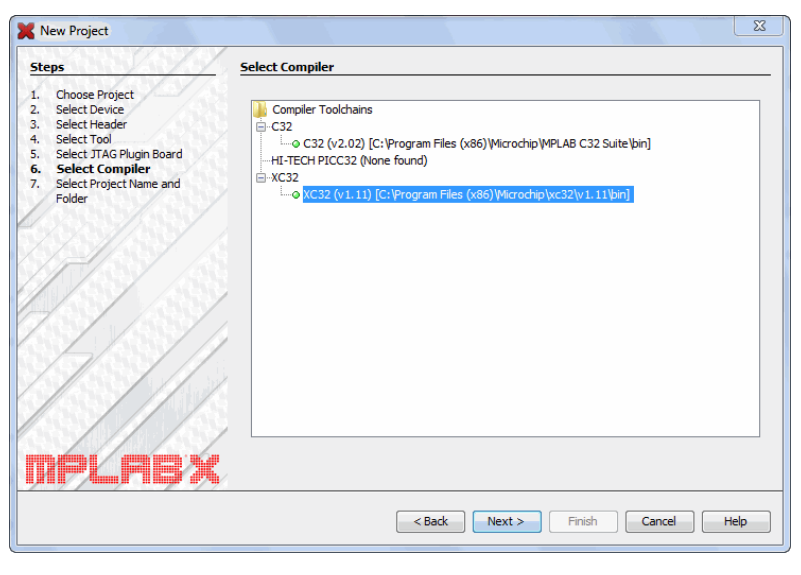
****

**Figure 4**

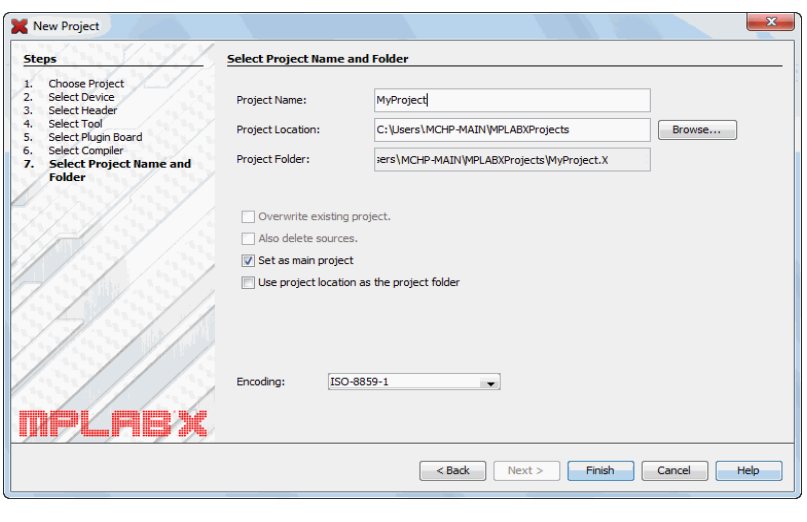
**Figure 5**

****

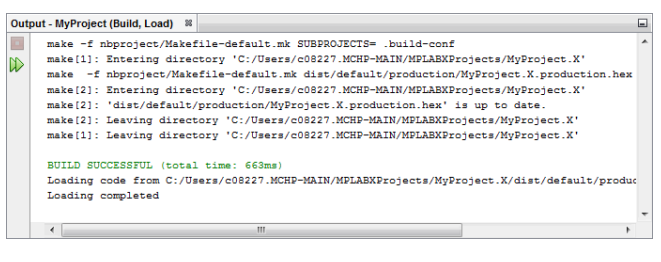
**Figure 6**

****

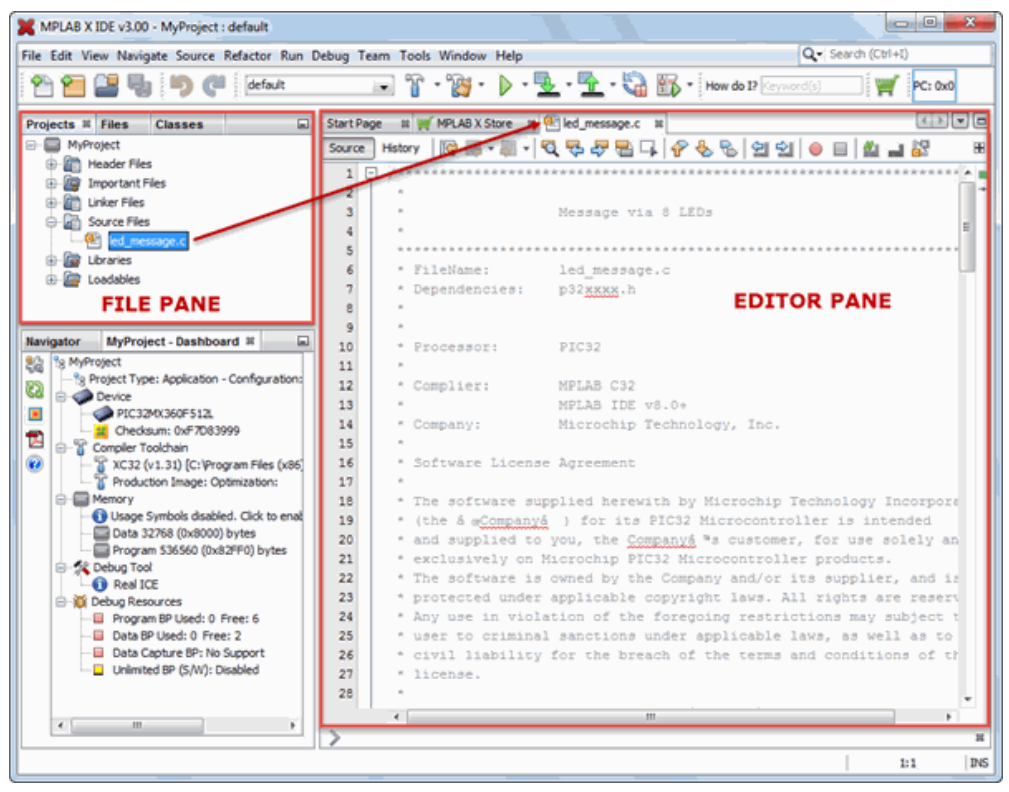
**Figure 7**

****

**Figure 8**

****

**Figure 9**

****