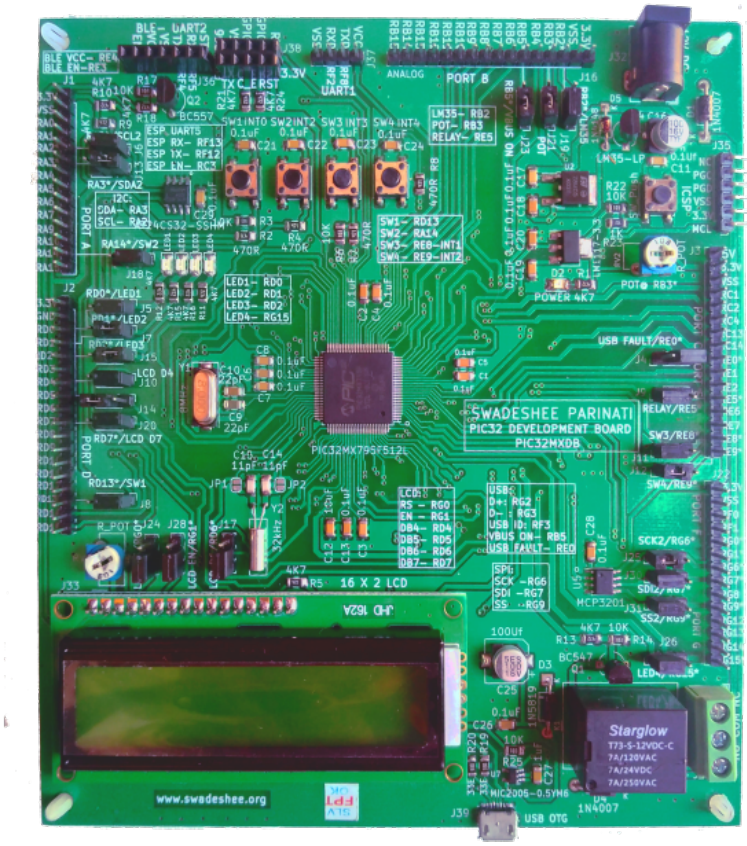
PIC 32MX DEVELOPMENT BOARD

USER'S GUIDE
Version 1.0



www.swadeshee.org parinatie@gmail.com

Table of Contents

1 Introduction

Thank you for purchasing the Swadeshee Parinati PIC32MX Development Board. This board provides a low-cost, modular development system for Microchip's PIC32MX 32-bit microcontroller. The Development Board comes pre-loaded with demonstration software for the user to explore the new features of the PIC32MX. It is also expandable through pin headers for external interface, which allows the user to extend its functionality. The PIC32MX Development Board also supplies on-board circuitry and consists of bootloader firmware for flash programming with USB port.

2 PIC32MX Development Board Functionality & Features

A representation of the layout of the PIC32MX Development Board is shown in Figure 1-1. The board includes these key features, as indicated in the diagram:

1. PIC32MX795F512L 32-bit microcontroller
2. USB Bootloader for Flash programming
3. Red power-indicator LED
4. Regulated +5V and +3.3V power supply for powering the board via USB or external DC source.
5. On-board crystal for precision microcontroller clocking (8 MHz)
6. On-board 32kHz crystal for RTC applications
7. USB connectivity
8. 2X16 LCD display
9. Four push-button switches for user-defined inputs and additional push button for microcontroller reset
10. Four user-defined indicator LEDs
11. 6-pin ICSP header for flash programmer interface
12. Dedicated pin headers for BLE and ESP8266 modules (Modules not included)
13. On-board Potentiometer for ADC input and LM35 temperature sensor
14. On-board MCP3201 12-bit A/D converter with SPI interface
15. On-board EEPROM with I2C interface
16. Jumpers for shared I/O pins with on-board peripherals
17. On-board 12V SPDT relay for power load switching

3 Host Computer Requirements

To communicate with and program the development board, the following hardware and software requirements must be met:

• PC-compatible system
• An available USB port on PC or powered USB hub
• Windows OS

4 Software Installation

MPLABX IDE:

1. Download the latest version of MPLABX IDE from:
https://www.microchip.com/mplab/mplab-x-ide

2. Install the IDE in your computer

XC32 C Compiler:

1. Download the latest version of the XC32 C compiler from:
https://www.microchip.com/mplab/compilers

2. Install XC32 compiler in your system.

5 Creating a Project

1. Launch the MPLAB X IDE

2. Under file menu click on **New Project** or use hotkey **Ctrl+Shift+N**

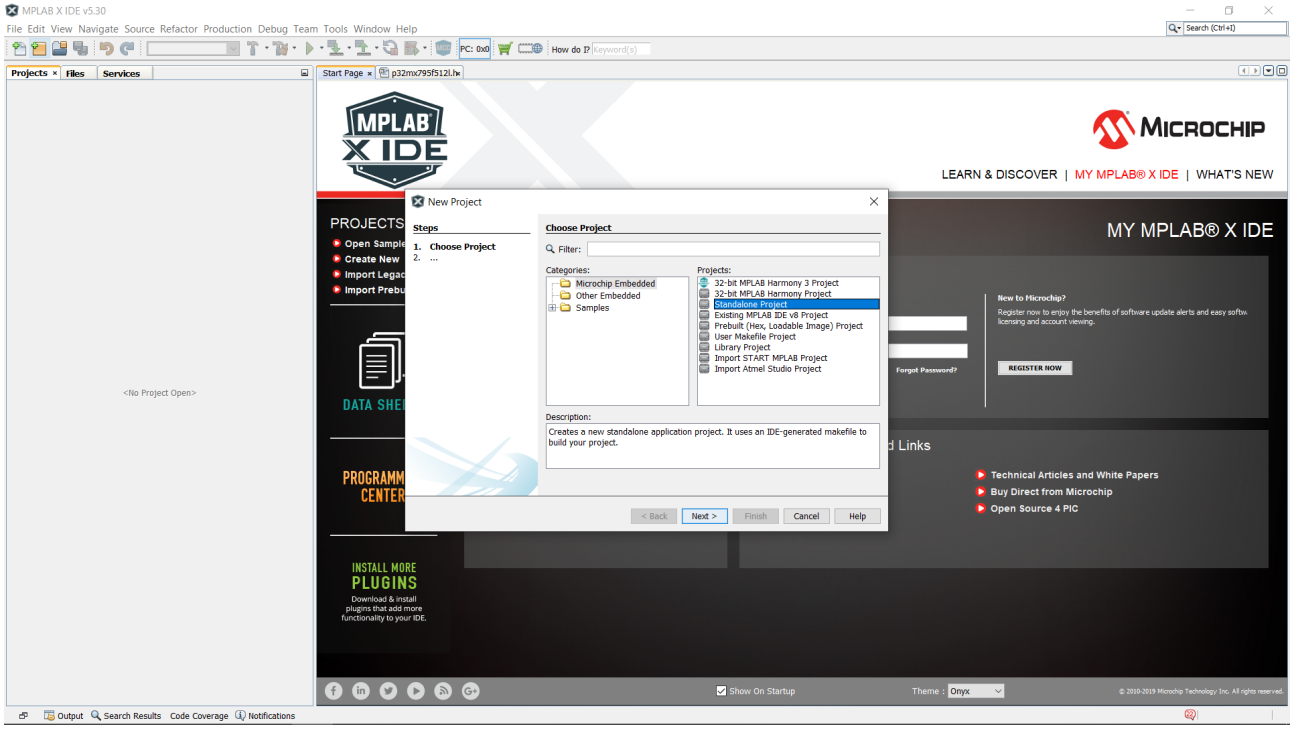


Fig. 5.1

3. Select **Microchip Embedded** under categories and **Standalone Projec**t. Clik Next. **(Refer Fig. 5.1)**

4.In device, type **PIC32MX795F512L** and click Next. **(Refer Fig. 5.2)**

5. Under hardware tools, select Simulator if the target board has Bootloader. If there is no bootloader, select ICD3/4 or PICKit3/4 as per your needs. Click Next. **(Refer Fig. 5.3)**

6. Under Compiler toolchains, select the latest version of XC32 and click Next. **(Refer Fig. 5.4)**

7. Enter Project name without spaces, select a project folder and tick on Set as Main Project and click Finish. **(Refer Fig. 5.5)**

8. Under Projects, a number of directories are listed on the left side of the window. **(Refer Fig. 5.6)**

9. Right click on Source files and select New-> C Main File. **(Refer Fig. 5.6)**

10. In new window which appears, give a file name and keep extension as c and hit finish. **(Refer Fig. 5.7)**

11. Under Window-> Target Memory View, select Configuration Bits. **(Refer Fig. 5.8)**

12. Select the configuration bits as per requirement. Recommended bit configuration is shown in Fig. 5.9. Click **Generate Source code to output**.

13. Copy the generated code and paste it to the top of the main file.**(Refer Fig. 5.10)**

14. Write the code to blink the LED. (LED blink code is given in Section 6)

15. Save the file and click on **Build Project** or press F11.**(Refer Fig. 5.11)**

16. Check if Build is successful. **(Refer Fig. 5.12)**

17. Dashboard gives details about the memory usage.**(Refer Fig. 5.13)**

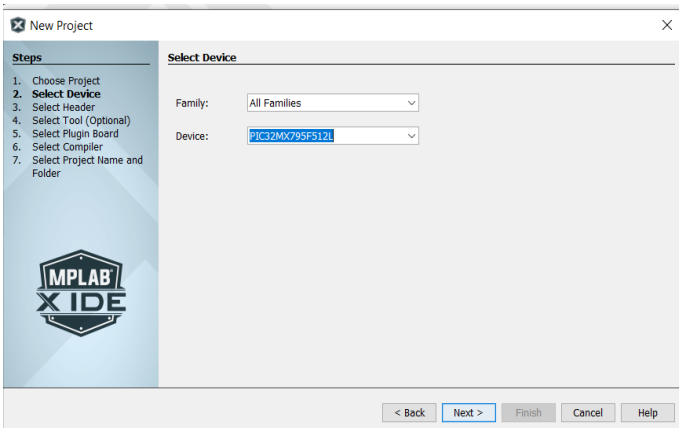


Fig. 5.2

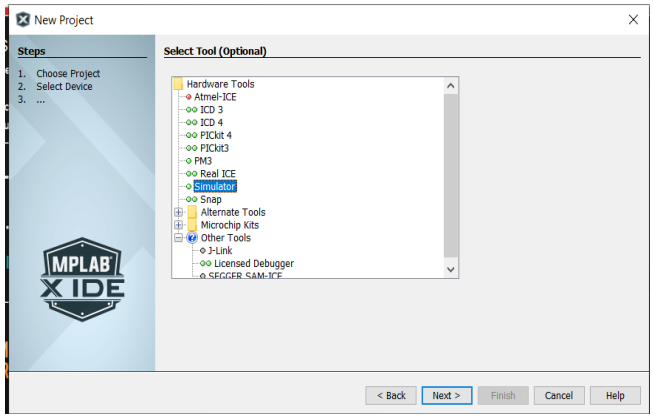
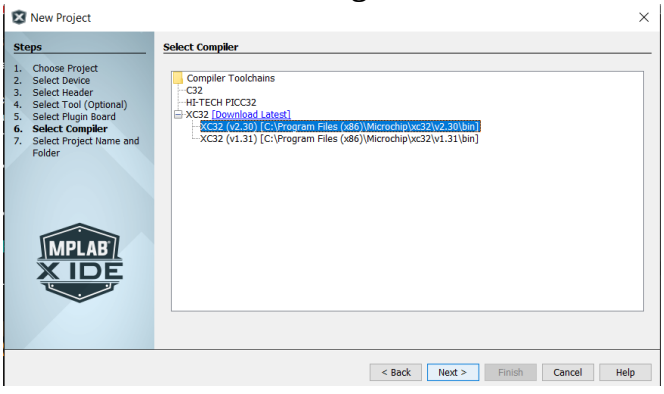


Fig. 5.3



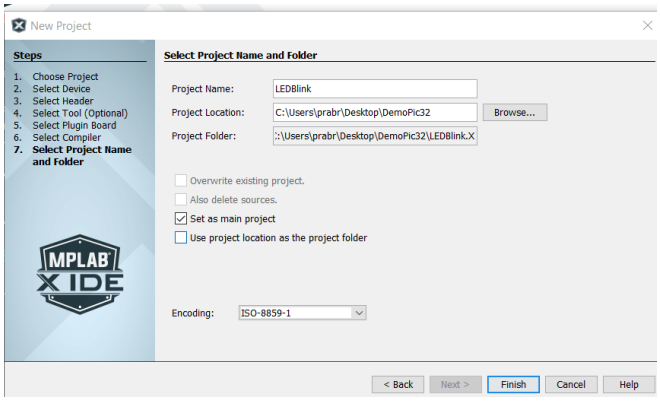
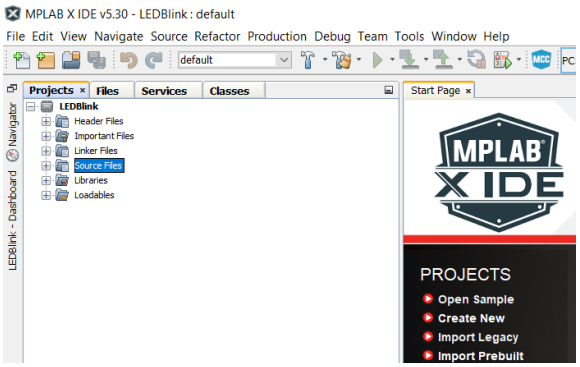


Fig. 5.4

Fig. 5.5



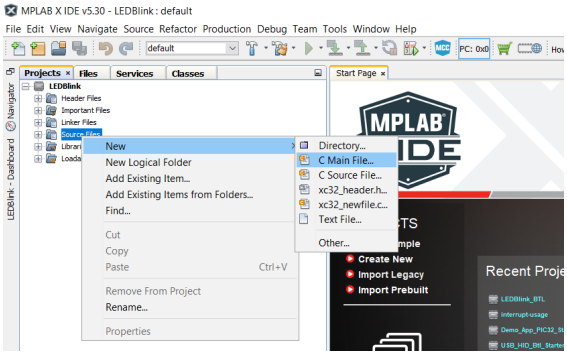


Fig. 5.6

Fig. 5.7

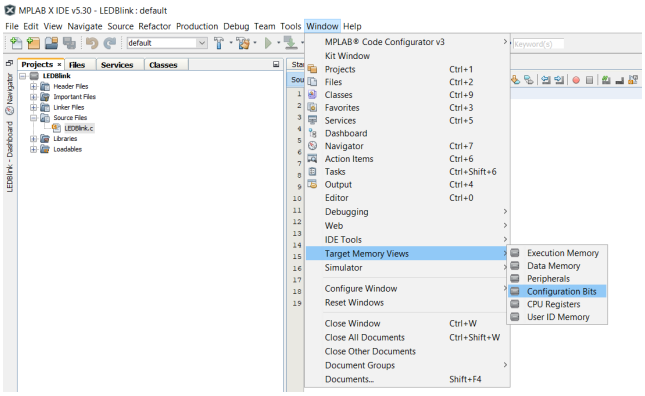


Fig. 5.8

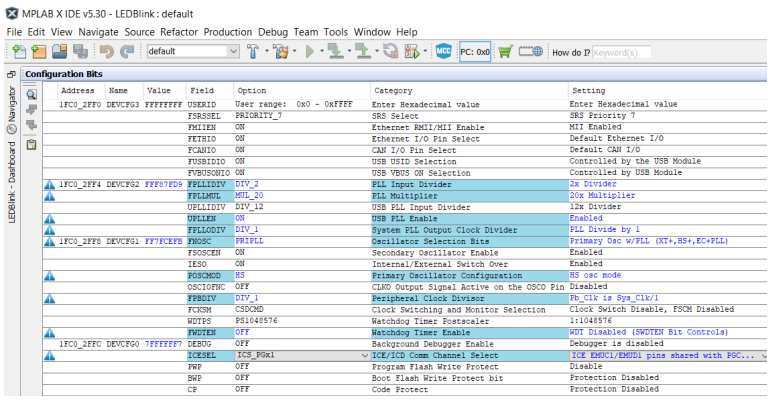


Fig. 5.9

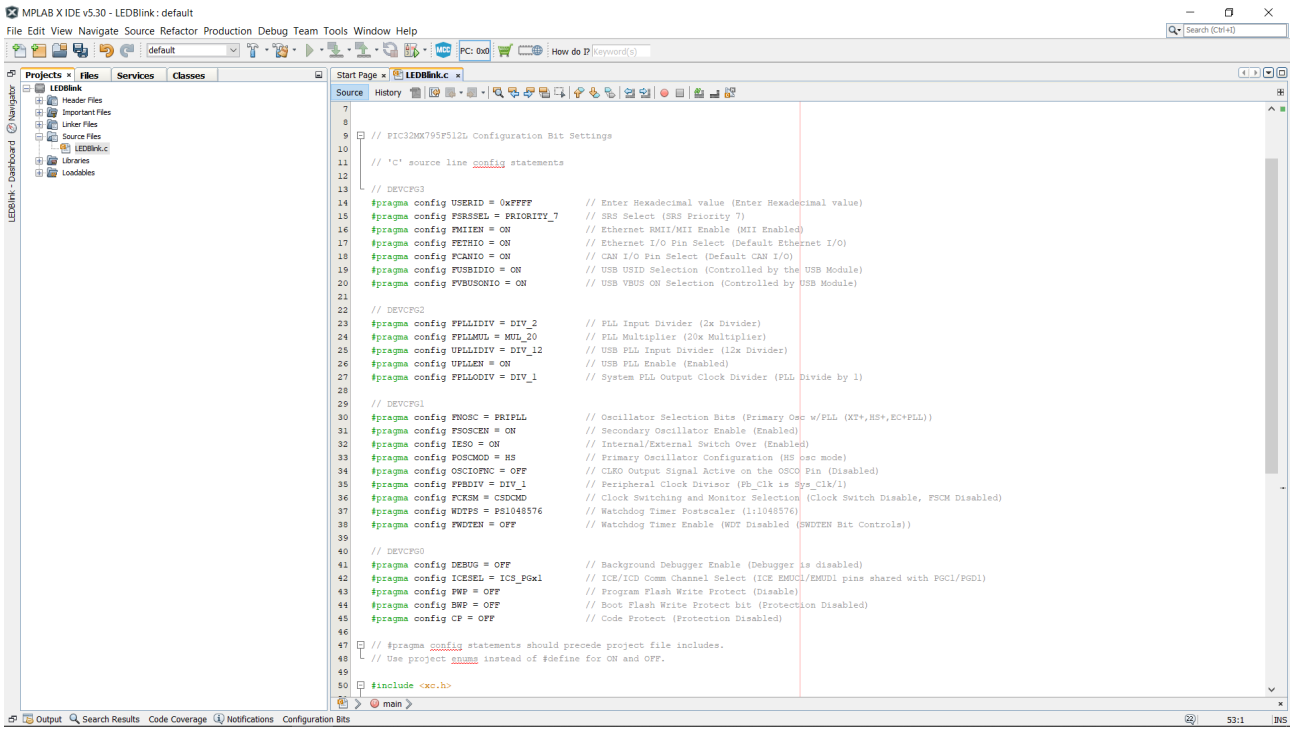


Fig. 5.10

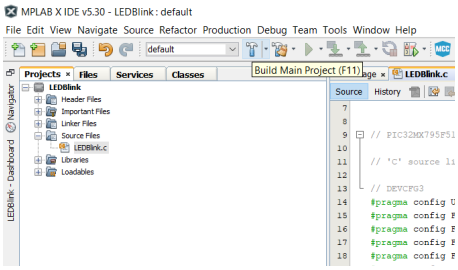


Fig. 5.11

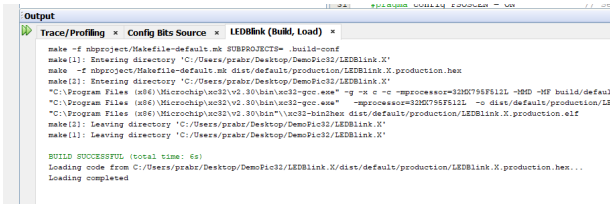


Fig. 5.12

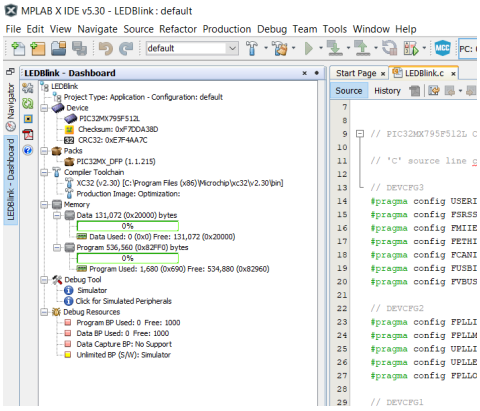


Fig. 5.13

6 LED Blink Code

```
/*
 * File:   LEDBlink.c
 * Author: prabhu
 *
 * Created on 4 January, 2020, 5:16 PM
 */


// PIC32MX795F512L Configuration Bit Settings

// 'C' source line config statements

// DEVCFG3
#pragma config USERID = 0xFFFF         // Enter Hexadecimal value (Enter Hexadecimal value)
#pragma config FSRSSEL = PRIORITY_7     // SRS Select (SRS Priority 7)
#pragma config FMIIEN = ON             // Ethernet RMII/MII Enable (MII Enabled)
#pragma config FETHIO = ON             // Ethernet I/O Pin Select (Default Ethernet I/O)
#pragma config FCANIO = ON             // CAN I/O Pin Select (Default CAN I/O)
#pragma config FUSBIDIO = ON           // USB USID Selection (Controlled by the USB Module)
#pragma config FVBUSONIO = ON          // USB VBUS ON Selection (Controlled by USB Module)

// DEVCFG2
#pragma config FPLLIDIV = DIV_2        // PLL Input Divider (2x Divider)
#pragma config FPLLMUL = MUL_20        // PLL Multiplier (20x Multiplier)
#pragma config UPLLIDIV = DIV_12       // USB PLL Input Divider (12x Divider)
#pragma config UPLLEN = ON             // USB PLL Enable (Enabled)
#pragma config FPLLODIV = DIV_1        // System PLL Output Clock Divider (PLL Divide by 1)

// DEVCFG1
#pragma config FNOSC = PRIPLL          // Oscillator Selection Bits (Primary Osc w/PLL
(XT+,HS+,EC+PLL))
#pragma config FSOSCEN = ON            // Secondary Oscillator Enable (Enabled)
#pragma config IESO = ON               // Internal/External Switch Over (Enabled)
#pragma config POSCMOD = HS            // Primary Oscillator Configuration (HS osc mode)
#pragma config OSCIOFNC = OFF          // CLKO Output Signal Active on the OSCO Pin (Disabled)
#pragma config FPBDIV = DIV_1          // Peripheral Clock Divisor (Pb_Clk is Sys_Clk/1)
#pragma config FCKSM = CSDCMD          // Clock Switching and Monitor Selection (Clock Switch
Disable, FSCM Disabled)
#pragma config WDTPS = PS1048576       // Watchdog Timer Postscaler (1:1048576)
#pragma config FWDTEN = OFF            // Watchdog Timer Enable (WDT Disabled (SWDTEN Bit
Controls))

// DEVCFG0
#pragma config DEBUG = OFF             // Background Debugger Enable (Debugger is disabled)
#pragma config ICESEL = ICS_PGx1       // ICE/ICD Comm Channel Select (ICE EMUC1/EMUD1 pins
shared with PGC1/PGD1)
#pragma config PWP = OFF               // Program Flash Write Protect (Disable)
#pragma config BWP = OFF               // Boot Flash Write Protect bit (Protection Disabled)
#pragma config CP = OFF                // Code Protect (Protection Disabled)

// #pragma config statements should precede project file includes.
// Use project enums instead of #define for ON and OFF.

#include <xc.h>
```

```
#define LED1            LATDbits.LATD0
#define TRIS_LED1       TRISDbits.TRISD0
#define LED1_SET()      LATDSET = _LATD_LATD0_MASK;
#define LED1_CLR()      LATDCLR = _LATD_LATD0_MASK;
#define LED1_INV()      LATDINV = _LATD_LATD0_MASK;
void delay(uint16_t ms)
{
    uint16_t i,j;
    for(j=0;j<=ms;j++)
    {
        for(i=0;i<=8888;i++);
    }
}

#include <stdio.h>
#include <stdlib.h>

/*
 *
 */
int main(int argc, char** argv)
{
    TRIS_LED1=0; // Set RD0 as output
    while(1)
    {
        LED1_INV();
        delay(500);
    }

    return (EXIT_SUCCESS);
}
```

7 Program the device using Flash Programmer.

Note: If the target board is loaded with Bootloader, using Flash Programmer may overwrite Bootloader.

1. After successful build of the project, under file menu select Project Properties. (Refer Fig. 7.1)

2. In Project Properties window, under categories select Conf:[default]. Under Hardware tools: Select the ICD3/4 or PicKit3/4 as per your need if it is not selected while creating the project. Click OK. (Refer Fig. 7.2)

3. Connect the flash programmer to the target board. (Ex. ICSP header in PicKit4.). Supply the power to the target board and click Make and Program Device Main Project. (Refer Fig. 7.3)

4. Wait for the programmer to flash the memory. LED should blink in the target board after flash is successful. Check if the LED jumper is placed properly in the target board if LED does not blink.

Note: ICD or PicKit programmer might take it long to download the code to the target device for the first time.

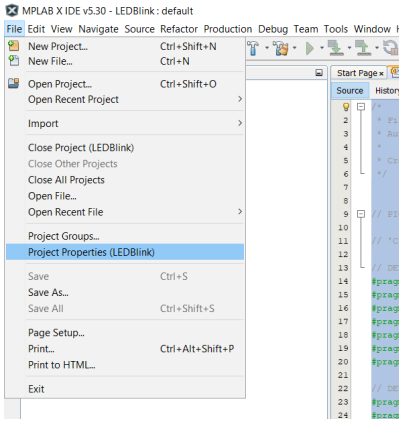


Fig. 7.1

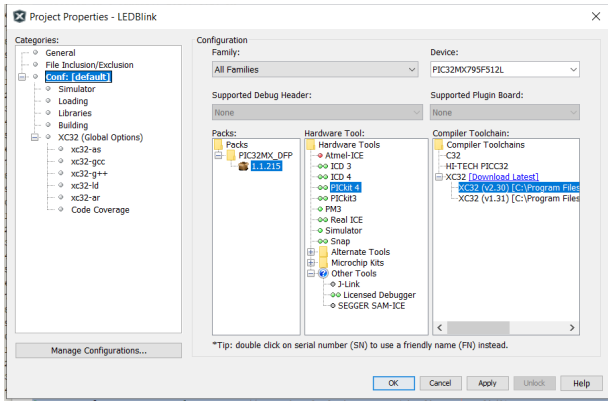


Fig. 7.2

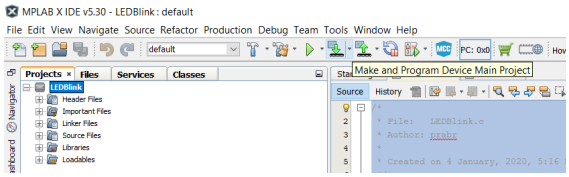


Fig. 7.3

8 Using the Bootloader Program

1. If the target board comes up with a bootloader, we need to add the custom linker file provided by the board vendor. Parinati PIC32MX795 development board has on-board USB bootloader. Copy the app_32MX795F512L.ld linker file to the project folder. (Refer Fig. 8.1)

2. In MPLABX IDE, under project folders, right click on linker files group and select Add existing Item.

3. In the Add file pop-up, select the copied linker file and click on **Select**.

4. **IMPORTANT:** The bootloader decides the configuration bits and we should not set the configuration bits in our application. So delete all #pragma lines which sets the configuration bits from the main program.

5. The code reduces to:

```
/*
 * File:   LEDBlink.c
 * Author: prabr
 *
 * Created on 4 January, 2020, 5:16 PM
 */


// PIC32MX795F512L Configuration Bit Settings

// 'C' source line config statements
```

```
#include <xc.h>

#define LED1            LATDbits.LATD0
#define TRIS_LED1       TRISDbits.TRISD0
#define LED1_SET()      LATDSET = _LATD_LATD0_MASK;
#define LED1_CLR()      LATDCLR = _LATD_LATD0_MASK;
#define LED1_INV()      LATDINV = _LATD_LATD0_MASK;
void delay(uint16_t ms)
{
    uint16_t i,j;
    for(j=0;j<=ms;j++)
    {
        for(i=0;i<=8888;i++);
    }
}

#include <stdio.h>
#include <stdlib.h>

/*
 *
 */
int main(int argc, char** argv)
{
    TRIS_LED1=0; // Set RD0 as output
    while(1)
    {
        LED1_INV();
        delay(500);
    }

    return (EXIT_SUCCESS);
}
```

6. Build the project.

7. Now connect the Parinati PIC32MX795 development board to the computer. Press and hold the switch SW1 and press Reset button. (Place the Jumper J8 on SW1 position in Parinati PIC32MX795 development board).

8. On Windows machine it appears as HID compliant vendor-defined device with VID 04D8 and PID 003C. (Refer Fig. 8.4)

9. Run the PC Program PIC32UBL.exe on windows computer. Enable USB option with 04D8 and PID 003C. (Refer Fig. 8.5)

10. Click on Connect.(Refer Fig. 8.6)

11. Click on Erase and wait for Erase confirmation.(Refer Fig. 8.7)

12. Click on Load hex file and browse for hex file in the project folder: \\LEDBlink.X\dist\default\production

Click on open. (Refer Fig. 8.8)

13. Click on Program, Verify and then Run Application. (Refer Fig. 8.9)

14. LED should blink in the Target Board. (Place the Jumper J5 on LED1 position in Parinati PIC32MX795 development board).

For more details and documentation on PIC32 Bootloader, visit:
https://www.microchip.com/wwwAppNotes/AppNotes.aspx?appnote=en554836

Check the 1388 Application Note: http://ww1.microchip.com/downloads/en/AppNotes/01388B.pdf

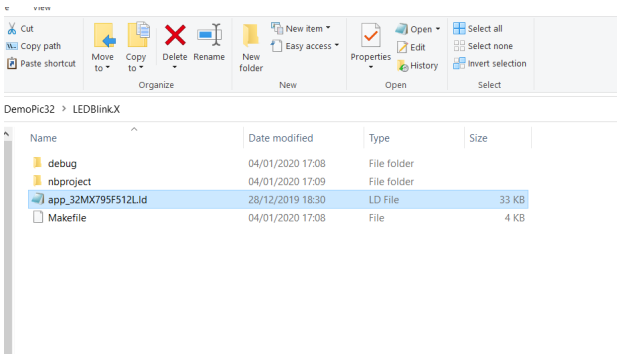


Fig. 8.1

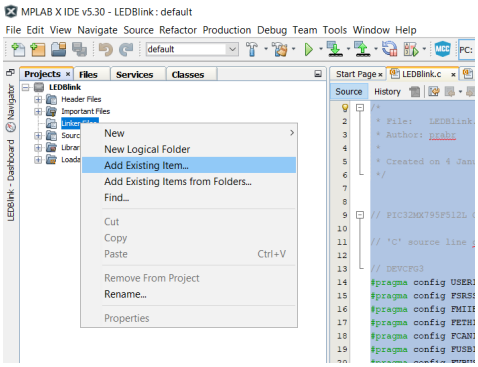


Fig. 8.2

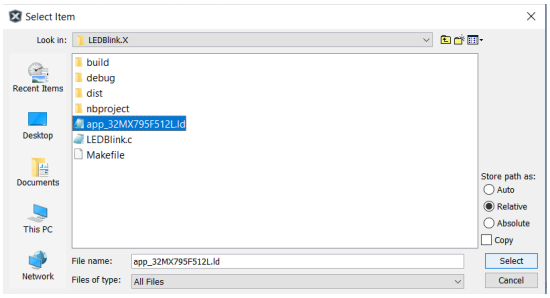


Fig. 8.3

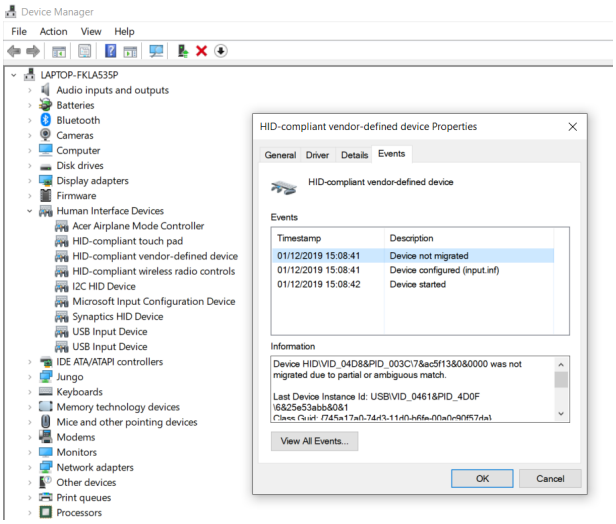


Fig. 8.4

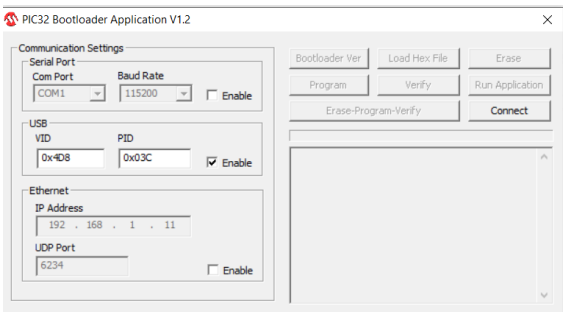


Fig. 8.5

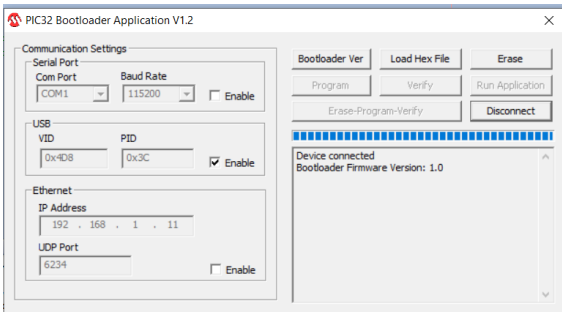


Fig. 8.6

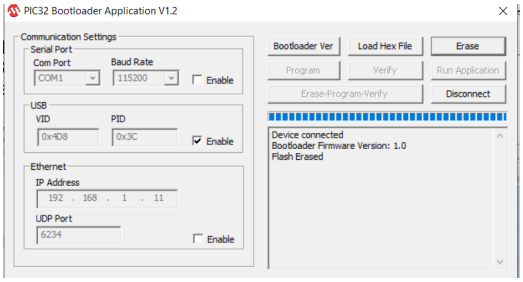


Fig. 8.7

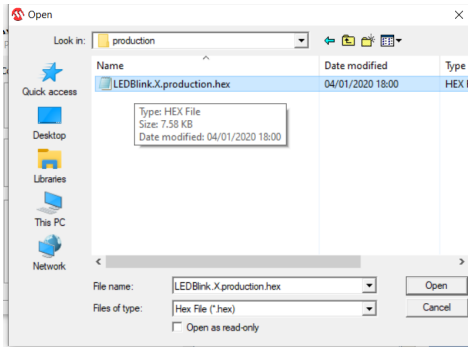


Fig. 8.8

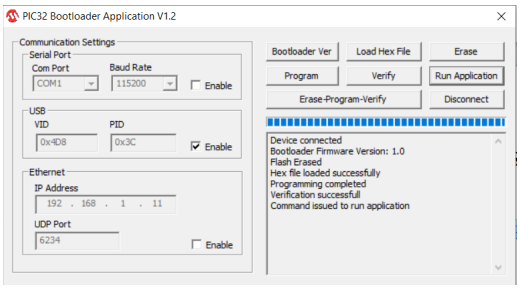


Fig. 8.9

9 Hardware Features

The key features of Swadeshee Parinati PIC32 development board are given in detail in this section:

1. **PIC32MX795F512L 32-bit microcontroller**
   The development board has PIC32MX795F512L permanently soldered on it to provide the computational resources to implement your embedded requirements.

2. **USB Bootloader for Flash programming**
   The development board comes pre-loaded with USB bootloader to easily program the PIC32MX795F512L without the need of hardware programmers.
   Note: Bootloader Courtesy- AN1388 Microchip Inc.

3. **Red power-indicator LED**
   The indicator LED glows if the board has sufficient power and voltage supply.

4. **Regulated +5V and +3.3V power supply** for powering the board via USB or external DC source.
   There are two ways to supply power to the development board:
   • USB bus power through USB port.
   • 12V DC input to DC Jack.

5. **On-board crystal for precision microcontroller clocking (8 MHz)**

The development board has 8 MHz external crystal oscillator for accurate clock timing. PIC32MX795F512L has internal 8MHz and 32kHz oscillators with PLL capability to run at 80MHz maximum system clock.

6. **On-board 32kHz crystal for RTC applications**
   For accurate applications based on real-time clock, the board has on-board 32 kHz external oscillator. To use 32 kHz external oscillator, you need to solder the pads JP1 and JP2.

7. **USB connectivity**
   The Micro USB B-type connector enables us to build a lot of possibilities around USB as PIC32MX795F512L has USB OTG support in addition to USB host and device capabilties.
   USB uses following pins:
   D+ connected RG2
   D- connected to RG3
   USB ID connected to RF3
   VBUS ON connected to RB5 through jumper J23
   USB FAULT connected to RE0 through jumper J4

8. **2X16 LCD display**
   2x16 LCD will help in building simple user interface. Jumpers J10, J14, J20, J24, J28 and J17 must be suitably placed to used LCD pins.
   The pins used by LCD are:
   D4 connected to RD4 through jumper J10
   D5 connected to RD5 through jumper J14
   D6 connected to RD6 through jumper J17
   D7 connected to RD7 through jumper J20
   EN connected to RG1 through jumper J28
   RS connected to RG0 through jumper J24

9. **Four push-button switches** for user-defined inputs and additional push button for microcontroller reset
   - SW1: Active-low switch connected to RD13 though jumper J8
   - SW2: floating switch connected to RA14 though jumper J18
   - SW3: Active-low switch connected to RE8 though jumper J11
   - SW4: floating switch connected to RE9 though jumper J12

   SW1 and SW3 has on-board pull-ups. SW2 and SW4 does not have pull-up resistors. SW2 and SW4 must be used with CN pins with internal weak pull-ups enabled. Use jumpers J8, J18, J11 and J12 to connect the switches to processor pins.

10. **Four user-defined indicator LEDs**
    LED1- active high connected to RD0
    LED2- active high connected to RD1
    LED2- active high connected to RD2
    LED3- active high connected to RG15
    Use jumpers J5, J7, J15 and J26 to connect the LEDs to processor pins.

11. **6-pin ICSP header** for flash programmer interface

The board supports ICSP flash programming using PicKit3 or PicKit4. ICSP header is available on J35.
When ICSP is not used, following two pins can be used for GPIO:
PGC- RB0
PGD- RB1
Note: Swadeshee Parinati PIC32 development board has pre-loaded USB bootloader for flash programming. So using ICSP header for flash programming will overwrite the existing bootloader code.

12. **Dedicated pin headers for BLE and ESP8266 modules** (Modules not included)
The development board has headers for ESP8266 and BLE modules such as HM10 or HC05 to be connected to establish a wireless connectivity. With this the board can be used for IoT projects.

13. **On-board Potentiometer for ADC input and LM35 temperature sensor**
To work with ADC, the board has one potentiometer and LM35 temperature sensor.
Potentiometer is connected to RB3 through jumper J21
LM35 is connected to RB2 through jumper J19.

14. **On-board MCP3201 12-bit A/D converter with SPI interface**
For testing SPI communication, MCP3201 is provided on-board which gets analog input from LM35 sensor. MCP3201 is 12-bit ADC with Vref supplied from 3.3V regulated output.
For SPI communication:
SCK is connected to RG6 through Jumper J25
SDI is connected to RG7 through Jumper J30
SS2 is connected to RG9 through Jumper J31

15. **On-board EEPROM with I2C interface**
The development board has 24C32 EEPROM to store non-volatile data. It makes use of 2-wire I2C protocol.
SDA is connected to RA3 with a pull-up resistor
SCL is connected to RA2 with a pull-up resistor.
Use jumpers J6 and J13 for 2-wire communications.

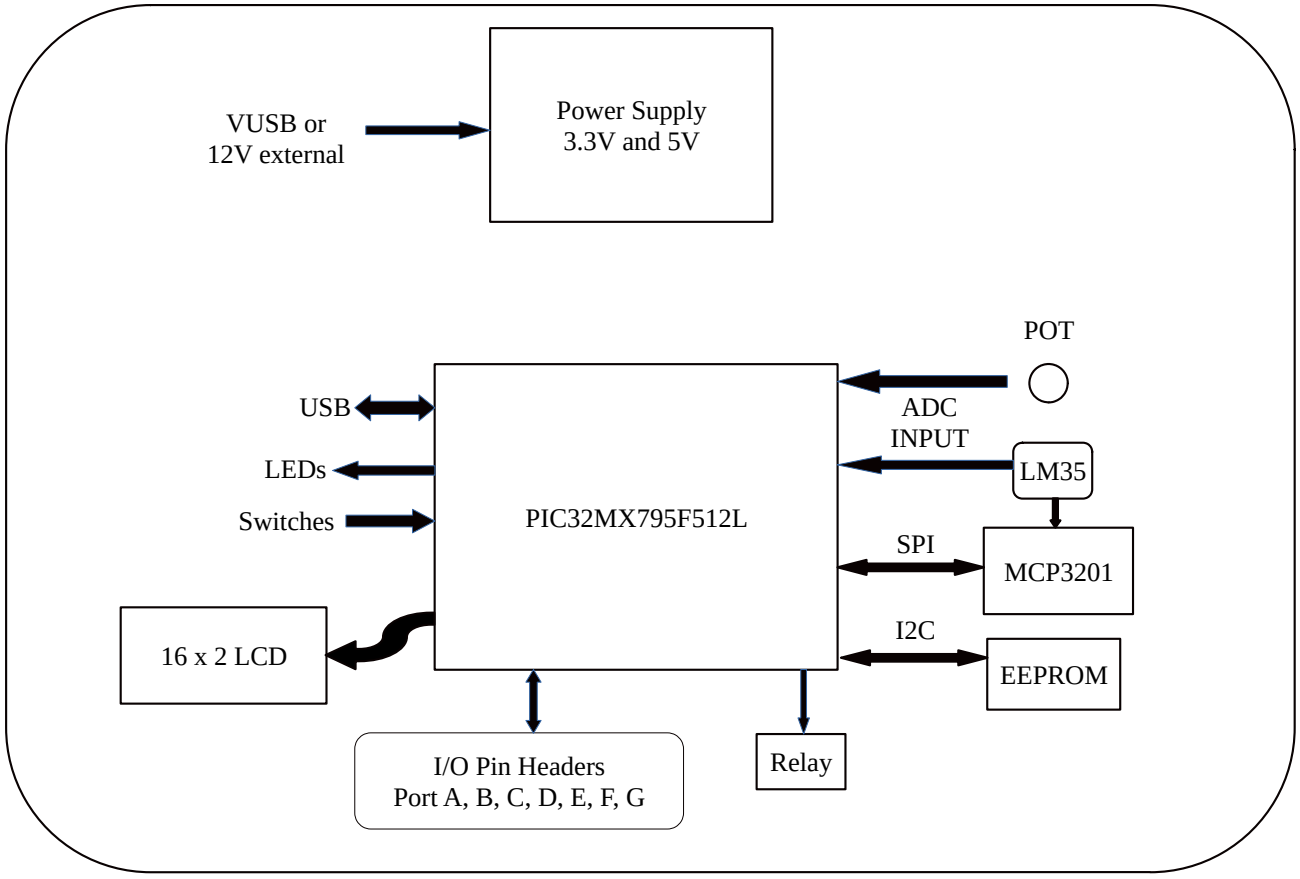
16. **Jumpers for shared I/O pins with on-board peripherals**
All the I/O pins are available to the users Jof the development board. Most of I/O pins are available at Headers J1, J2, J32, J35, J3, J22. Few pins are available at J36, J37 and J3. The pins marked * on PCB are multiplexed with few on-board peripherals. These multiplexed I/O pins are connected to through 3 pin headers that allow us to use with on-board peripherals or external hardware.

17. **On-board 12V SPDT relay for power load switching**

The board has a 12V SPDT relay for switching power line loads such as Lamps and DC pumps. The relay is driven by an active-high relay drive circuit connected to RE5 through jumper J9.

10 Block Diagram

Figure: High-level block diagram of PIC32MX795F512L Development board



11 Board Schematics

12 Sample Codes

13 Online resources & Courses

https://www.udemy.com/user/74fe0e9f-2797-4fb8-b425-982f74d2c914/

14 References

https://github.com/parinatie/PIC32MX