

## BÀI 1. LẬP TRÌNH VI ĐIỀU KHIỂN PIC VÀ I/O PORT

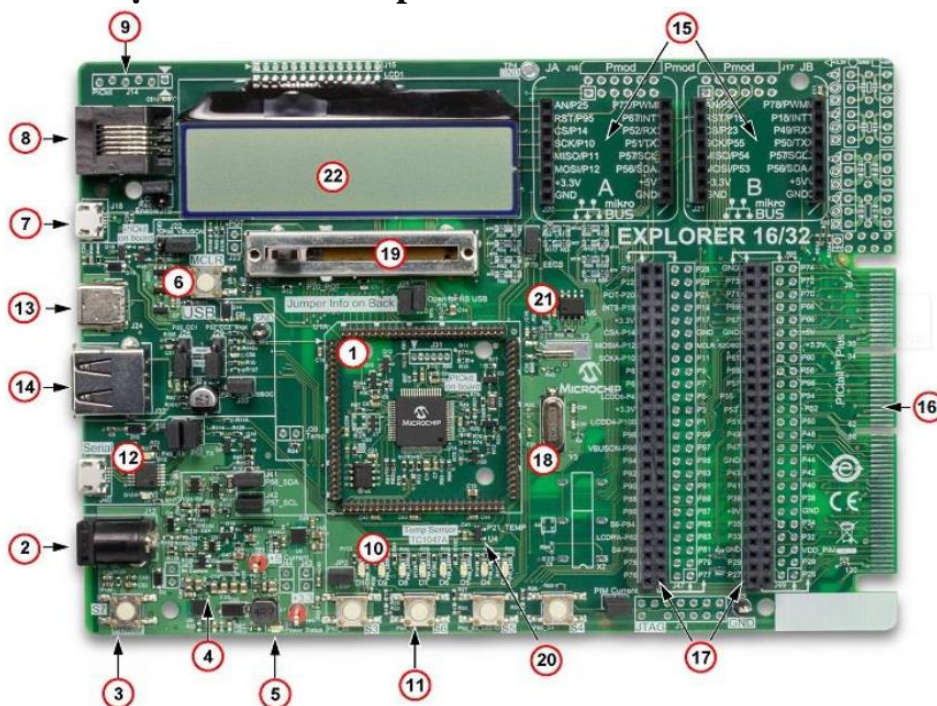
### 1. Mục đích

Qua bài học sinh viên có thể đạt được các kiến thức sau:

- Cách thức sử dụng board PIC Explorer 16/32 trong quá trình học tập.
- Cách debug chương trình bằng công cụ MPLAB-X.
- Cách giao sử dụng I/O port của PIC24F và giao tiếp với các thiết bị ngoại vi như nút nhấn, led trên board PIC Explorer 16/32.

### 2. Tóm tắt nội dung lý thuyết

#### 2.1. Giới thiệu board PIC Explorer 16/32



Hình 2.1. Board PIC Explorer 16/32 dùng PIC24FJ1024GB610.

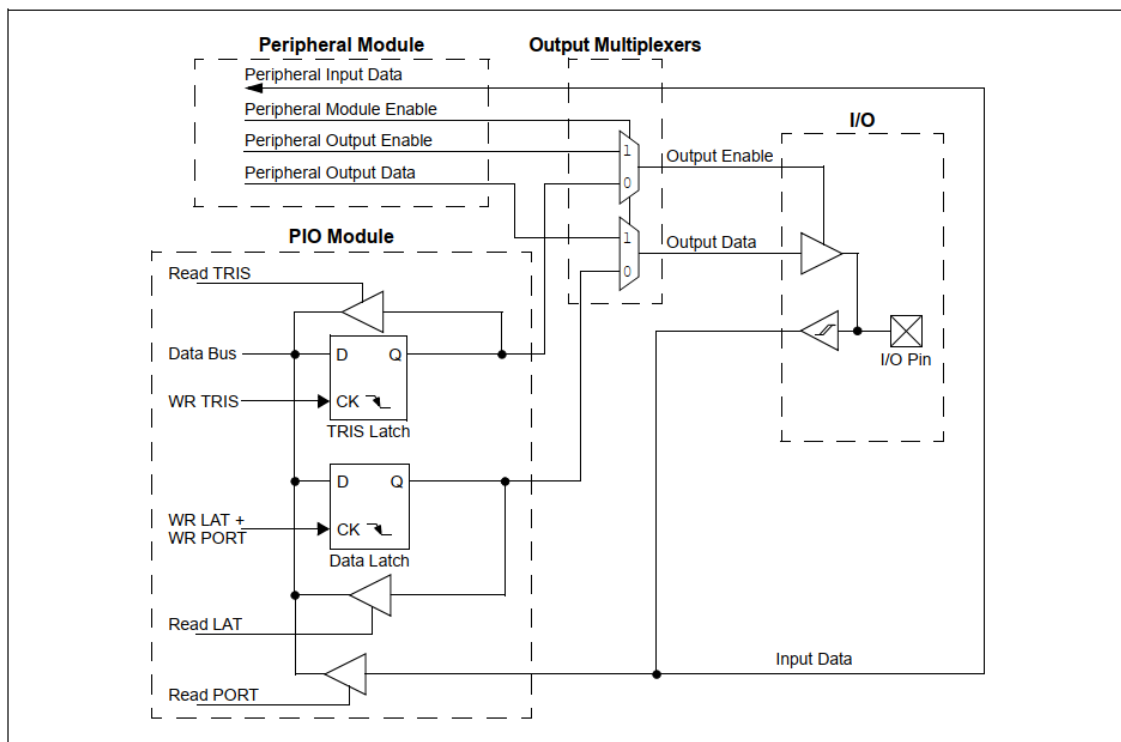
Board PIC Explorer 16/32 là board thực tập cho phép sinh viên làm quen với lập trình Vi điều khiển PIC cũng như thiết kế các ứng dụng liên quan từ đơn giản đến phức tạp. Các thành phần trên board Pic Explorer 16/32 bao gồm:

1. Vi điều khiển PIC24FJ1024GB610.
2. Cổng cắm 8-15V cung cấp điện thế ngõ vào +3.3V và +5V cho board.
3. Nút nguồn.
4. Mạch chuyển đổi nguồn đầu vào thành +3.3/5V cho mạch.
5. LED thông báo trạng thái cấp nguồn.
6. Nút bấm MCLR để reset mạch.
7. Cổng USB micro-B kết nối với PICkit On Board (PKOB) cho phép nạp code và debug chương trình.

8. Khe cắm In-Circuit Debugger (ICD).
9. 6 chân để kết nối với bộ nạp code rời như PICKit 3.
10. 8 LEDs báo hiệu dành cho ứng dụng.
11. 11 nút nhấn dành cho ứng dụng.
12. Chip chuyển đổi USB sang UART và I2C MCP2221A.
13. Cổng USB type C dành cho phát triển ứng dụng trên USB.
14. Cổng USB type A dành cho phát triển ứng dụng host USB.
15. 2 mikroBUS dùng để kết nối với board mở rộng.
16. Socket cắm vào PCI.
17. 100 chân để kết nối với các thiết bị ngoại vi.
18. Thạch anh 8Mhz và Real-Time Calendar/Clock (RTCC) 32.768 Khz.
19. Biến trở 10Kohm dùng cho ADC.
20. Cảm biến nhiệt độ TC1047A.
21. EEPROM nối tiếp.
22. LCD (2 dòng, 16 cột).

## 2.2. I/O port trên PIC24F

Mục đích chung I/O pins có thể được coi là đơn giản nhất của thiết bị ngoại vi. Chúng cho phép các PIC MCU theo dõi và kiểm soát các thiết bị khác bên ngoài. Để thêm nhiều tính linh hoạt và chức năng đầu vào một thiết bị, một số chân được ghép với nhiều chức năng thay thế. Các chức năng này phụ thuộc vào những tính năng ngoại vi là trên thiết bị. Nói chung, khi một thiết bị ngoại vi đang hoạt động, pin có thể không được sử dụng như một mục đích chung I/O pin. Hình 2.2 thể hiện sơ đồ khối của một cổng I/O thông thường.



Hình 2.2. Sơ đồ khối của IO port.

### 2.3. Thanh ghi điều khiển IO port

IO port trên PIC24F cho phép nó có thể đóng vai trò là chân digital (input/output) hay chân analog tùy vào việc cấu hình các thanh ghi của IO port. Trên PIC24FJ128GA010 có 7 được đặt tên là PORTA, PORTB, PORTC, PORTD, PORTE, PORTF, PORTG. Trong đó, mỗi PORT đều có 4 thanh ghi cơ bản.

- ANSx: Thanh ghi cấu hình là chân analog hay digital.
- TRISx: Thanh ghi hướng dữ liệu.
- PORTx: Thanh ghi I/O port.
- LATx: Thanh ghi LAT.
- ODCx: Thanh ghi I/O open-drain.

Với “x” là các kí từ từ A → G tương ứng cho PORTA → PORTG.

#### 2.3.1. Thanh ghi ANSx

Các bit trên thanh ghi ANSx cho phép cấu hình chân IO là chân analog hay digital. Nếu các bit của thanh ghi ANSx là “1” thì chân tương ứng của PORTx là chân analog, nếu ANSx là “0” thì chân tương ứng của PORTx được cấu hình là chân digital.

#### 2.3.2. Thanh ghi TRISx

Các bit trên thanh ghi TRISx xác định xem mỗi pin gắn liền với cổng I/O là đầu vào hoặc đầu ra tương ứng. Nếu các bit Tris cho một I/O pin là ‘1’, thì pin đó là một đầu vào. Nếu các bit Tris cho một I/O pin là ‘0’, pin đó được cấu hình cho một đầu ra.

#### 2.3.3. Thanh ghi PORTx

Thanh ghi PORT dùng để đọc dữ liệu đầu vào trên các I/O pin nếu nó là đầu vào, hoặc dùng để ghi dữ liệu ra các I/O pin nếu nó là đầu ra.

#### 2.3.4. Thanh ghi LATx

Thanh ghi LATx được gắn với một IO pin để loại bỏ các lỗi xảy ra trong quá trình thực hiện các câu lệnh chuyển đổi giữa read và ghi. Việc đọc giá trị trên thanh ghi sẽ đọc dữ liệu được chốt trong các flipflop tương ứng thay vì giá trị trên các pin. Ngược lại, việc ghi giá trị vào thanh LAT sẽ tương tự như việc ghi vào thanh ghi PORT.

#### 2.3.5. Thanh ghi ODCx

Các pin của IO port có thể được cấu hình như là một chân digital output bình thường hay là một chân open-drain output. Việc này được điều khiển bởi các bit trong thanh ghi ODC tương ứng. Nếu bit là “1” thì sẽ cho phép nó là một chân open-drain output, ngược lại nếu là “0” thì nó sẽ là một chân digital output bình thường.

## 3. Nội dung thực hành

### 3.1. Các bước thực hiện cấu hình cho IO Port

#### 1. Cấu hình thanh ghi ANSx.

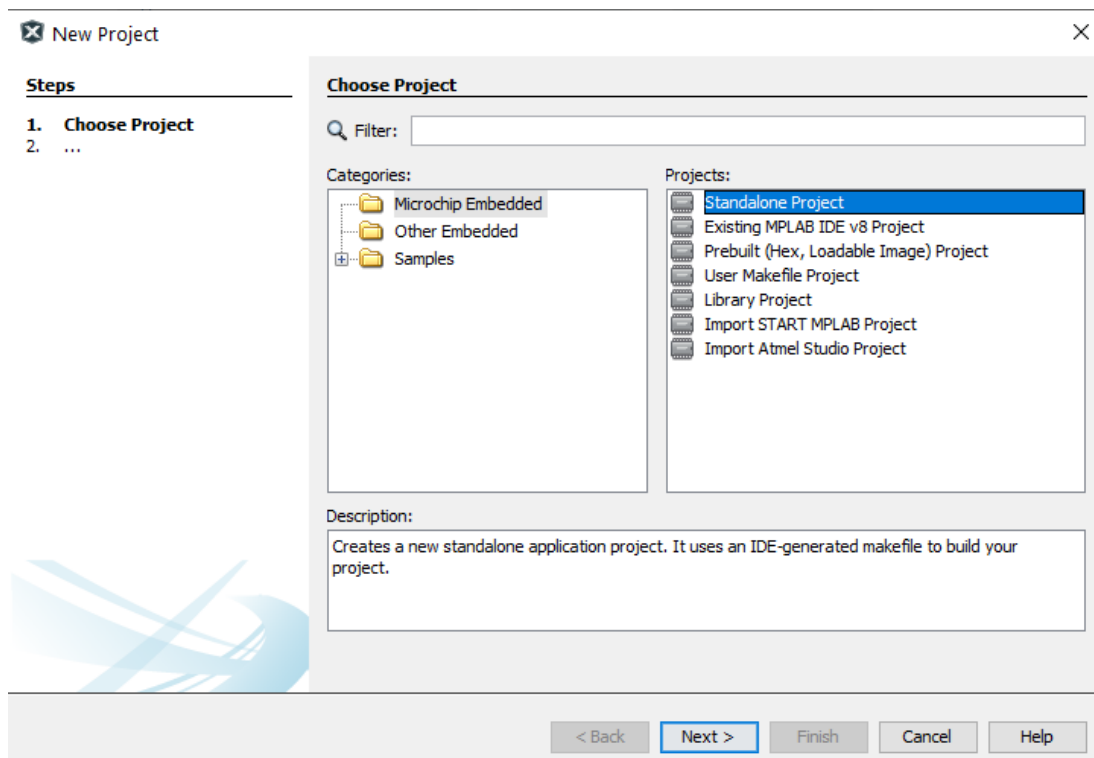
Ví dụ:

+ Nếu muốn cấu hình các chân IO của Port A là digital thì: → **ANSA = 0;**

- + Nếu muốn cấu hình các chân IO của Port A là analog thì:  $\rightarrow \text{ANSA} = 0\text{xff}$ ;
- 2. Cấu hình thanh ghi **TRISx**.  
Ví dụ:
  - + Nếu muốn cấu hình Port A là input thì:  $\rightarrow \text{TRISA} = 0\text{xff}$ ;
  - + Nếu muốn cấu hình Port A là output thì:  $\rightarrow \text{TRISA} = 0$ ;
- 3. Đọc hoặc ghi dữ liệu với thanh ghi **PORTx**.  
Ví dụ:
  - + Đọc giá trị từ Port A thì :  $\rightarrow \text{data} = \text{PORTA}$ ;
  - + Ghi giá trị vào Port A thì :  $\rightarrow \text{PORTA} = \text{data}$ ;

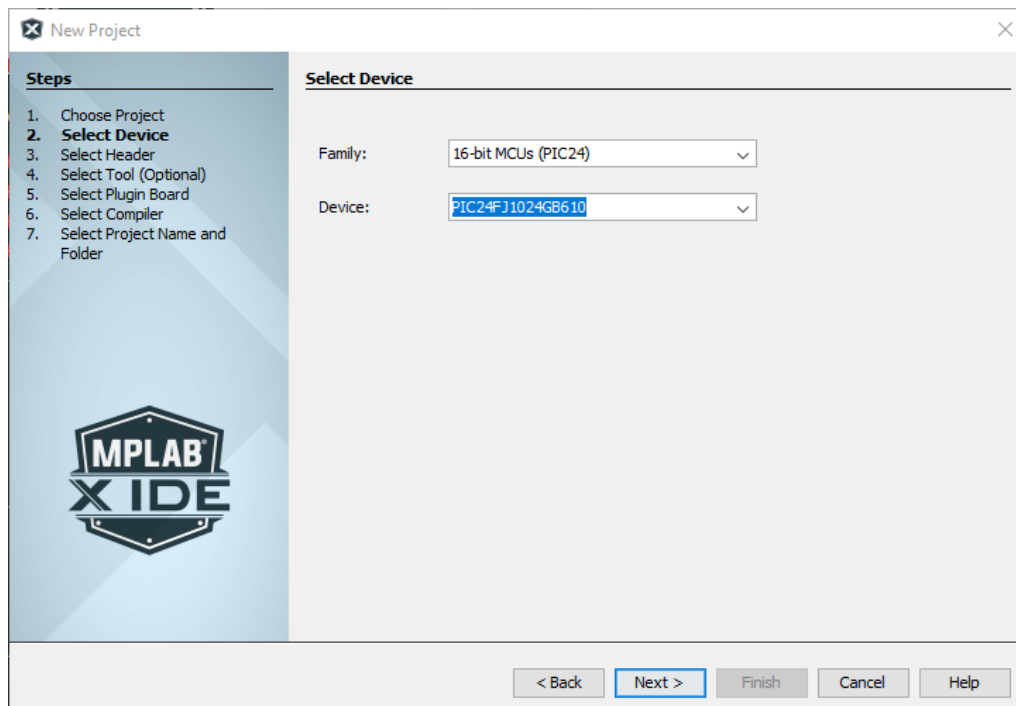
### 3.2. Các bước thực hành

Tạo project mới cho PIC24FJ1024GB610 trên phần mềm MPLAB-X, cấu hình các tham số, đặt tên là Lab1 và lưu ở đường dẫn ổ đĩa D theo các hình từ 2.3 đến hình 2.7 bên dưới.



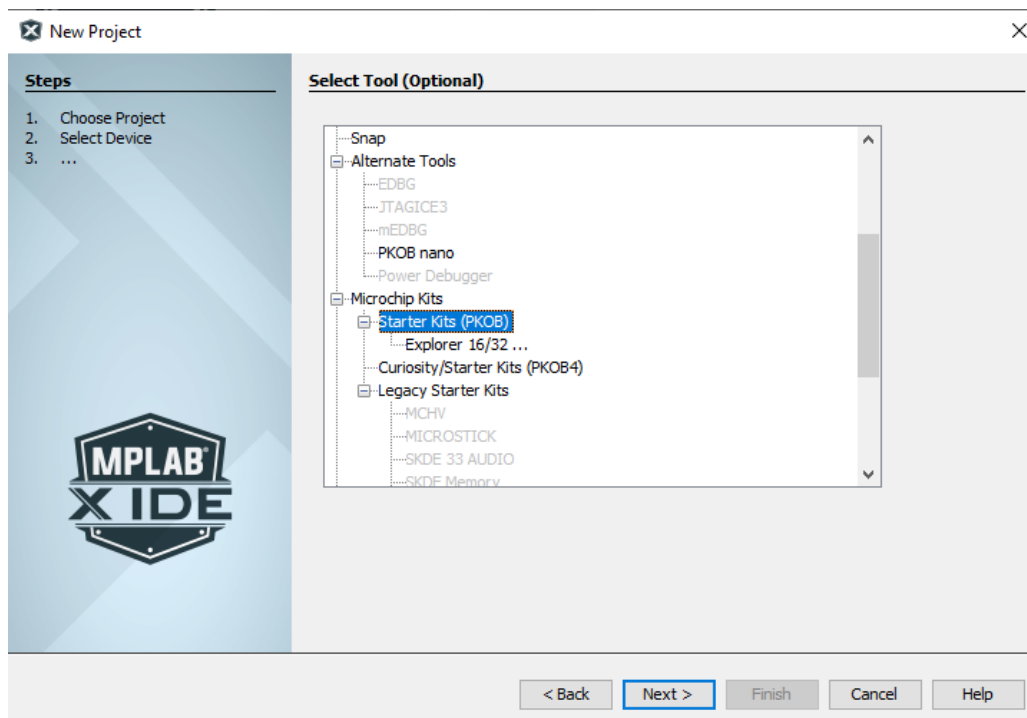
Hình 2.3. Chọn lại project.

Chọn **Device** là **PIC24FJ1024GB610** như hình bên dưới:



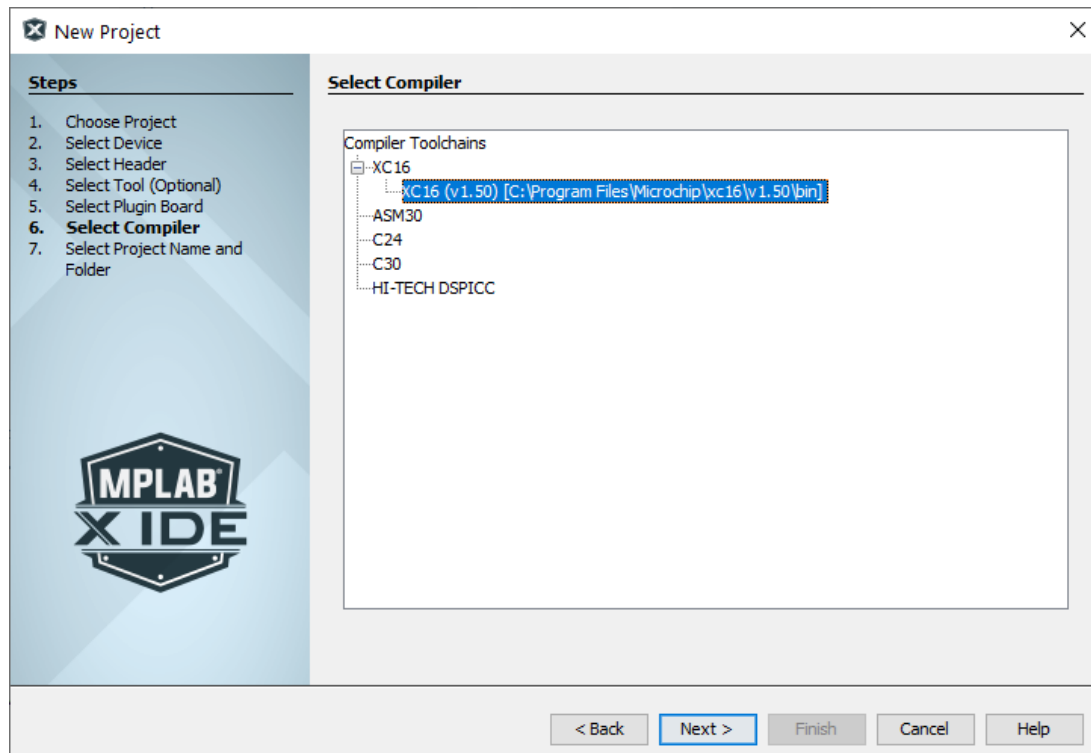
Hình 2.4. Chọn thiết bị là PIC24F.

Chọn Công cụ nạp code (**Select Tool**) là **Starter Kits (PKOB)** như hình bên dưới (lưu ý phải kết nối board với máy tính, nếu chưa kết nối thì bỏ qua bước này):



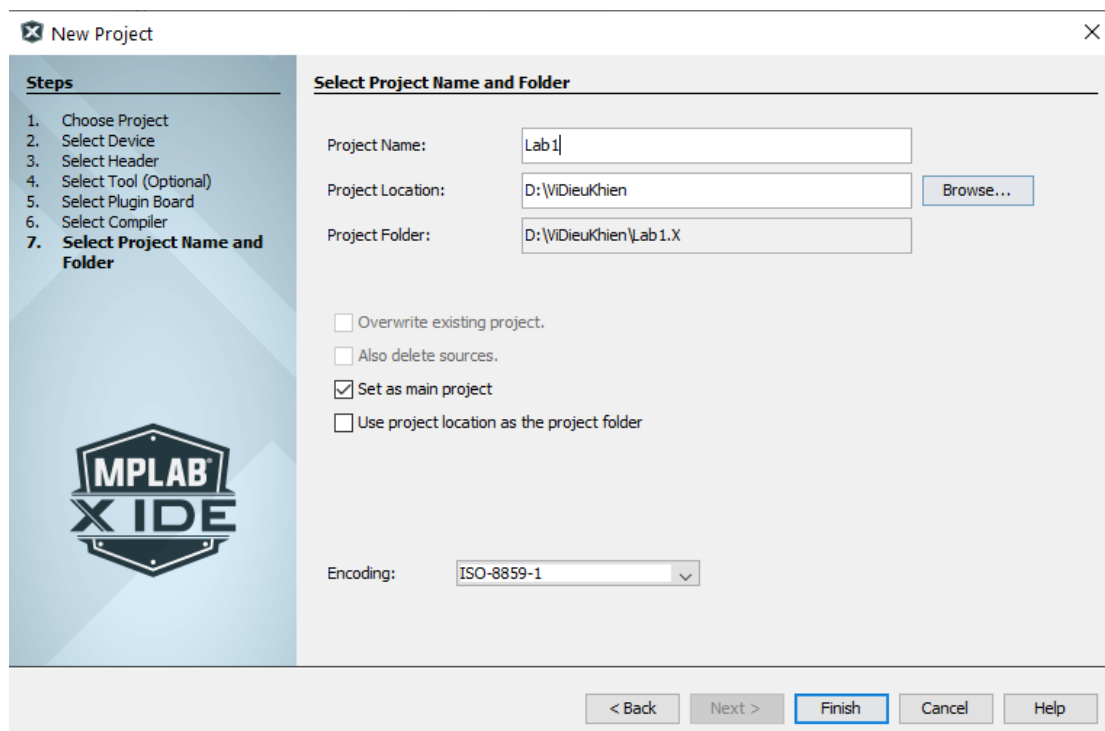
Hình 2.5. Chọn Starter Kits (PKOB) ở phần công cụ.

Chọn **Compiler** là **XC16** như hình bên dưới :



Hình 2.6. Chọn trình biên dịch là xc16.

Đặt tên project là **Lab1** và lưu ở đường dẫn như hình bên dưới:



Hình 2.7. Đặt tên và đường dẫn lưu project.

Thêm file main.c với nội dung là đoạn code bên dưới vào project.

```
void delay(int t){
    int i,temp;
    temp = t;
    for(i = t; i > 0; i--){
        while(temp > 0){
            temp--;
        }
    }
}

int main(void) {
    ANSA = 0; // Các chân của PORTA là chân digital
    TRISA = 0; // PORTA là đầu ra.
    PORTA = 0; // Các chân của PORTA bằng 0.
    while(1){
        delay(10000);
        PORTA = 0xff;
        delay(10000);
        PORTA = 0;
    }
}
```

Để có thể sử dụng xuất dữ liệu PORTA ra LED, sinh viên cần tắt chức năng JTAG bằng cách bỏ dấu check tại Configuration Bits set in code và chọn giá trị JTAG port is **Disabled**.

Để vào Configuration Bits ta chọn **Window → Target Memory Views → Configuration Bits** như hình 2.9.





Bookmarks	Debugger Console	Notifications	Terminal	Usages	Variables	Output	Configuration Bits	Configuration Bits
	Address	Name	Value	Field	Option	Category		Setting
	0ABF00	FSEC	FFFFFF	BWRP	OFF	Boot Segment Write-Protect bit		Boot Segment may i
				BSS	DISABLED	Boot Segment Code-Protect Level bits		No Protection (ot
				BSEN	OFF	Boot Segment Control bit		No Boot Segment
				GWRP	OFF	General Segment Write-Protect bit		General Segment m
				GSS	DISABLED	General Segment Code-Protect Level bits		No Protection (ot
				CWRP	OFF	Configuration Segment Write-Protect bit		Configuration Seg
				CSS	DISABLED	Configuration Segment Code-Protect Level bits		No Protection (ot
				AIVTDIS	OFF	Alternate Interrupt Vector Table bit		Disabled AIVT
	0ABF10	FBSLIM	FFFFFF	BSLIM	User range: 0x0 - 0x1FFF	Boot Segment Flash Page Address Limit bits		Enter Hexadecimal
	0ABF18	FOSCSEL	FFFF0A	FNOSC	PRI	Oscillator Source Selection		Primary Oscillato
				PLLMODE	PLL96DIV2	PLL Mode Selection		96 MHz PLL. (8 MH
				IESO	OFF	Two-speed Oscillator Start-up Enable bit		Start up with use
	0ABF1C	FOSC	FFFFFF	POSCMD	HS	Primary Oscillator Mode Select bits		HS Crystal Oscill
				OSCIOFCN	OFF	OSC2 Pin Function bit		OSC2 is clock out
				SOSCSEL	ON	SOSC Power Selection Configuration bits		SOSC is used in c
				PLLSS	PLL_PRI	PLL Secondary Selection Configuration bit		PLL is fed by the
				IOLWAY	ON	Peripheral pin select configuration bit		Allow only one re
				FCKSM	CSDCMD	Clock Switching Mode bits		Both Clock switch
	0ABF20	FWDI	FFFFFF	WDTPS	PS32768	Watchdog Timer Postscaler bits		1:32,768
				FWPSA	PRI28	Watchdog Timer Prescaler bit		1:28
				FWDTEN	ON	Watchdog Timer Enable bits		WDT Enabled
				WINDIS	OFF	Watchdog Timer Window Enable bit		Watchdog Timer in
				WDTWIN	WIN25	Watchdog Timer Window Select bits		WDT Window is 25%
				WDTMUX	WDTCLK	WDT MUX Source Select bits		WDT clock source
				WDTCLK	LPRC	WDT Clock Source Select bits		WDT uses LPRC
	0ABF24	FPOR	FFFFFB	BOREN	ON	Brown Out Enable bit		Brown Out Enable
				LPCFG	ON	Low power regulator control		Retention Sleep c
				DNVREN	ENABLE	Downside voltage protection Enable bit		Downside protecti
	0ABF28	FICD	FFFFDF	ICS	PGD1	ICD Communication Channel Select bits		Communicate on PG
				JTAGEN	OFF	JTAG Enable bit		JTAG is disabled
				BTSWP	OFF	BOOTSWP Disable		BOOTSWP instructi
	0ABF2C	FDEVOPT1	FFFFFF	ALTCMP1	DISABLE	Alternate Comparator Input Enable bit		C1INC, C2INC, and
				TMPRPIN	OFF	Tamper Pin Enable bit		TMPRN pin functio
				SOSCHP	ON	SOSC High Power Enable bit (valid only when SOSCSEL = 1		Enable SOSC high
				ALTVPREF	ALTREFEN	Alternate Voltage Reference Location Enable bit		VREF+ and CVREF+

Memory

Configuration Bits

Format

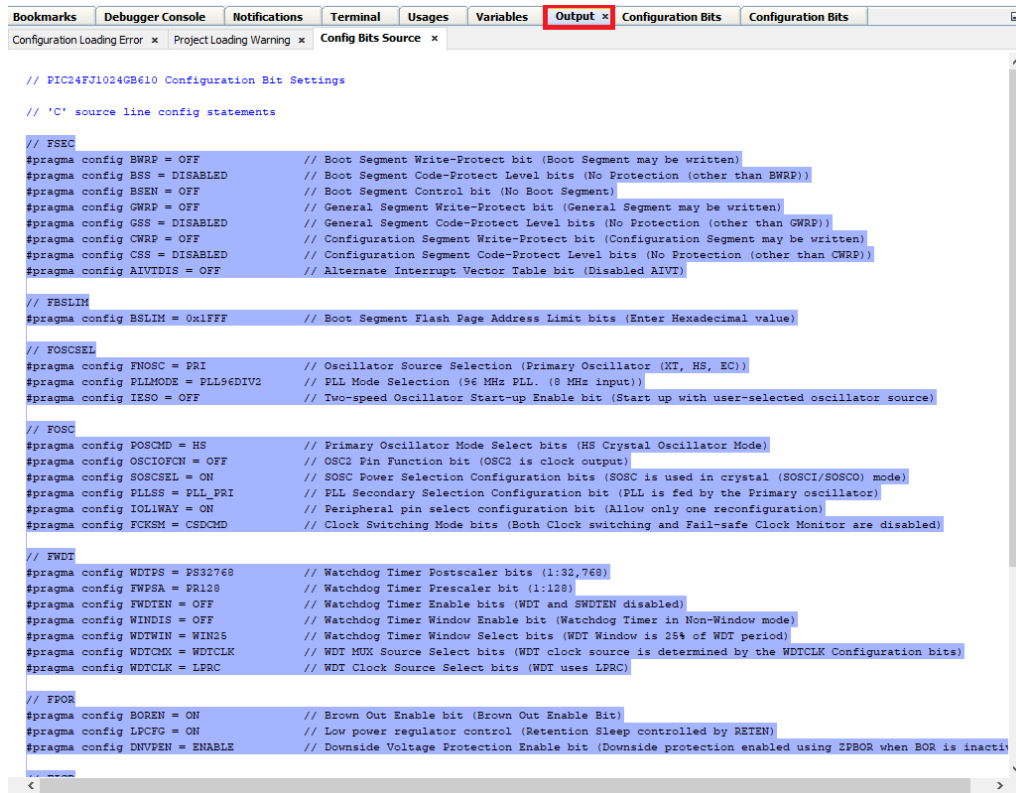
Single

Generate Source Code to Output

Hình 2.9. Cấu hình các bit.



Sau khi nhấn **Generate**, copy đoạn code cấu hình như trong hình 2.10 và dán vào đoạn code đã có trong file main.c.



```
// PIC24FJ1024GB610 Configuration Bit Settings

// 'C' source line config statements

// FSEC
#pragma config BWRP = OFF           // Boot Segment Write-Protect bit (Boot Segment may be written)
#pragma config BSS = DISABLED       // Boot Segment Code-Protect Level bits (No Protection (other than BWRP))
#pragma config BSEN = OFF           // Boot Segment Control bit (No Boot Segment)
#pragma config GWRP = OFF           // General Segment Write-Protect bit (General Segment may be written)
#pragma config GSS = DISABLED       // General Segment Code-Protect Level bits (No Protection (other than GWRP))
#pragma config CWRP = OFF           // Configuration Segment Write-Protect bit (Configuration Segment may be written)
#pragma config CSS = DISABLED       // Configuration Segment Code-Protect Level bits (No Protection (other than CWRP))
#pragma config AIVDIS = OFF         // Alternate Interrupt Vector Table bit (Disabled AIVT)

// FBSLIM
#pragma config BSLIM = 0x1FFF       // Boot Segment Flash Page Address Limit bits (Enter Hexadecimal value)

// FOSCSEL
#pragma config FNOOSC = PRI         // Oscillator Source Selection (Primary Oscillator (XT, HS, EC))
#pragma config PLLMODE = PLL96DIV2  // PLL Mode Selection (96 MHz PLL. (8 MHz input))
#pragma config IESO = OFF           // Two-speed Oscillator Start-up Enable bit (Start up with user-selected oscillator source)

// FOSC
#pragma config POSCMD = HS           // Primary Oscillator Mode Select bits (HS Crystal Oscillator Mode)
#pragma config OSCIOFNC = OFF        // OSC2 Pin Function bit (OSC2 is clock output)
#pragma config SOSCSSEL = ON        // SOSC Power Selection Configuration bits (SOSC is used in crystal (SOSCI/SOSCO) mode)
#pragma config PLLSS = PLL_PRI       // PLL Secondary Selection Configuration bit (PLL is fed by the Primary oscillator)
#pragma config IOL1WAY = ON         // Peripheral pin select configuration bit (Allow only one reconfiguration)
#pragma config FCKSM = CSDCMD        // Clock Switching Mode bits (Both Clock switching and Fail-safe Clock Monitor are disabled)

// FWD
#pragma config WDTPS = PS32768      // Watchdog Timer Postscaler bits (1:32,768)
#pragma config FWPSA = PR128        // Watchdog Timer Prescaler bit (1:128)
#pragma config FWDTEN = OFF          // Watchdog Timer Enable bits (WDT and SWDTEN disabled)
#pragma config WNDIS = OFF           // Watchdog Timer Window Enable bit (Watchdog Timer in Non-Window mode)
#pragma config WDTWIN = WIN25       // Watchdog Timer Window Select bits (WDT Window is 25% of WDT period)
#pragma config WDTCKM = WDTCLK       // WDT MUX Source Select bits (WDT clock source is determined by the WDTCLK Configuration bits)
#pragma config WDTCLK = LPRC         // WDT Clock Source Select bits (WDT uses LPRC)

// FPOR
#pragma config BOREN = ON            // Brown Out Enable bit (Brown Out Enable Bit)
#pragma config LPCFG = ON            // Low power regulator control (Retention Sleep controlled by RETEN)
#pragma config DNVEN = ENABLE        // Downside Voltage Protection Enable bit (Downside protection enabled using ZPBOR when BOR is inactive)
```

Hình 2.10. Đoạn code cấu hình.

Sau khi copy ta được kết quả như hình 2.11.



```
#include "xc.h"
#include <p24fj1024gb610.h>

// FSEC
#pragma config BWRP = OFF           // Boot Segment Write-Protect bit (Boot Segment may be written)
#pragma config BSS = DISABLED       // Boot Segment Code-Protect Level bits (No Protection (other than BWRP))
#pragma config BSEN = OFF           // Boot Segment Control bit (No Boot Segment)
#pragma config GWRP = OFF           // General Segment Write-Protect bit (General Segment may be written)
#pragma config GSS = DISABLED       // General Segment Code-Protect Level bits (No Protection (other than GWRP))
#pragma config CWRP = OFF           // Configuration Segment Write-Protect bit (Configuration Segment may be written)
#pragma config CSS = DISABLED       // Configuration Segment Code-Protect Level bits (No Protection (other than CWRP))
#pragma config AIVDIS = OFF         // Alternate Interrupt Vector Table bit (Disabled AIVT)

// FBSLIM
#pragma config BSLIM = 0x1FFF       // Boot Segment Flash Page Address Limit bits (Enter Hexadecimal value)

// FOSCSEL
#pragma config FNOOSC = PRI         // Oscillator Source Selection (Primary Oscillator (XT, HS, EC))
#pragma config PLLMODE = PLL96DIV2  // PLL Mode Selection (96 MHz PLL. (8 MHz input))
#pragma config IESO = OFF           // Two-speed Oscillator Start-up Enable bit (Start up with user-selected oscil

// FOSC
#pragma config POSCMD = HS           // Primary Oscillator Mode Select bits (HS Crystal Oscillator Mode)
#pragma config OSCIOFNC = OFF        // OSC2 Pin Function bit (OSC2 is clock output)
#pragma config SOSCSSEL = ON        // SOSC Power Selection Configuration bits (SOSC is used in crystal (SOSCI/SOSCO) mode)
#pragma config PLLSS = PLL_PRI       // PLL Secondary Selection Configuration bit (PLL is fed by the Primary oscil
#pragma config IOL1WAY = ON         // Peripheral pin select configuration bit (Allow only one reconfiguration)
#pragma config FCKSM = CSDCMD        // Clock Switching Mode bits (Both Clock switching and Fail-safe Clock Monitor
```

Hình 2.11. Đoạn code cấu hình đã được đưa vào file main.c

Sau khi kết nối, tiến hành chạy project bằng cách nhấn chuột phải vào **Lab1** → **Run**,

hoặc nhấn 1 trong 2 button  này trên thanh công cụ.

#### 4. Bài tập

##### a. Bài tập chuẩn bị ở nhà

1. Dựa trên các file trên tài liệu tham khảo, hãy cho biết các LED và nút nhấn được kết nối với board Explorer 16-32 thông qua các PORT gì?

##### b. Bài tập trên lớp

1. Tiến hành viết code để đọc giá trị từ nút nhấn S3 và S4, khi S3 được nhấn thì sẽ đếm lên, S4 được nhấn thì sẽ đếm xuống và giá trị đếm sẽ được xuất ra LED.
2. Tiến hành viết code để đọc giá trị từ nút nhấn S3 và S4, khi S3 được nhấn thì LED sẽ sáng dần, S4 được nhấn thì LED sẽ tối dần.

#### TÀI LIỆU THAM KHẢO

1. Explorer 16-32 Development Board User's Guide.pdf
2. PIC24FJ1024GB610 family datasheet.pdf
3. Explorer\_16\_32\_Schematics\_R6\_3.pdf.
4. PIC24FJ1024GB610 Plug-In Module (PIM) Information Sheet.pdf
5. dsPIC24 FRM IO Ports with Interrupt-on-Change (IOC) (70005186a).pdf