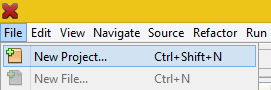
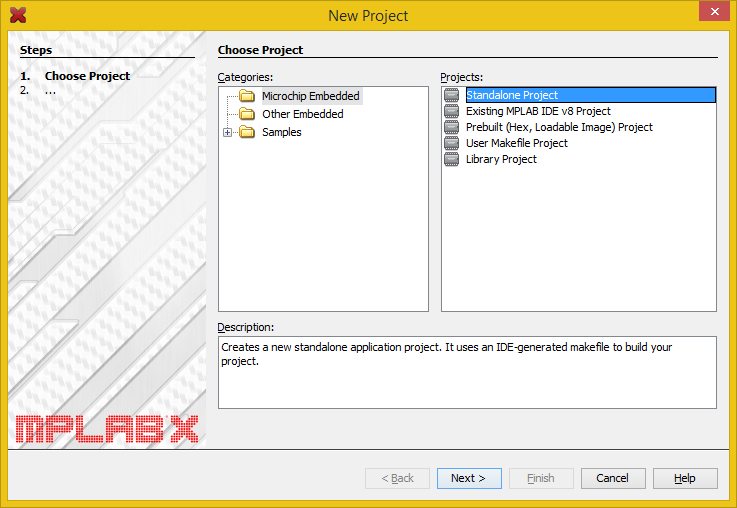
Bài tập về nhà tuần 3

Lập trình mô phỏng mạch nháy LED

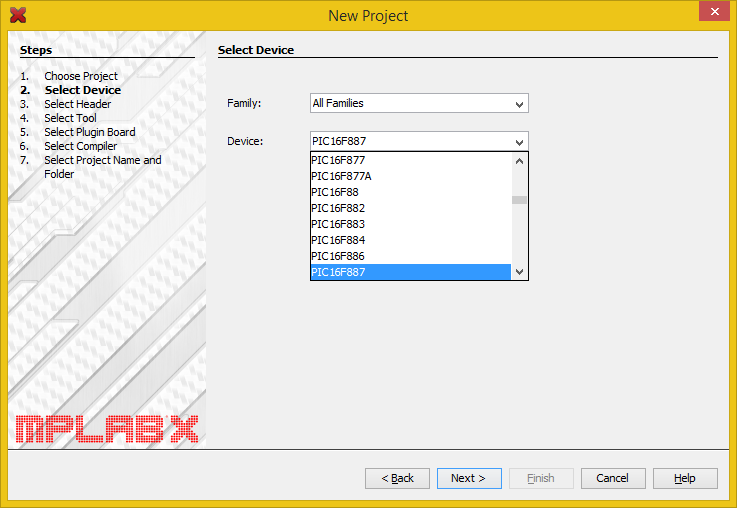
# Tạo project mới trên MPLABX



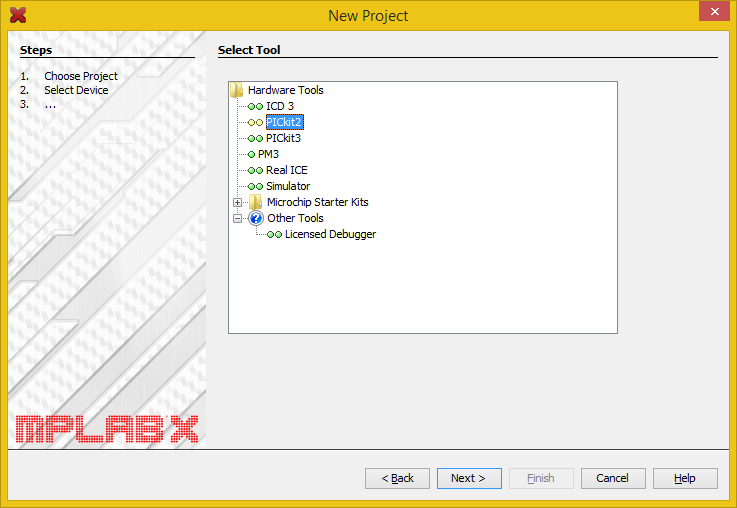
Sau khi khởi động MPLABX, vào **File > New Project...** để tạo project mới.



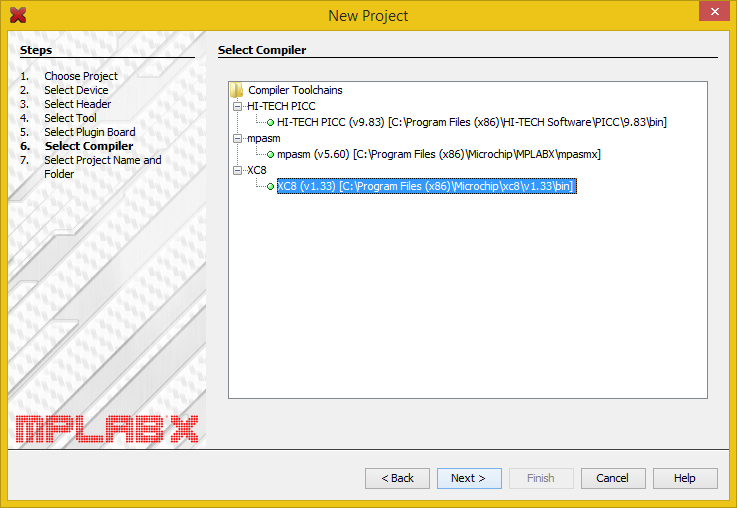
Trong giao diện tạo project mới, chọn loại project **Microchip Embedded > Standalone Project** rồi bấm **Next**.



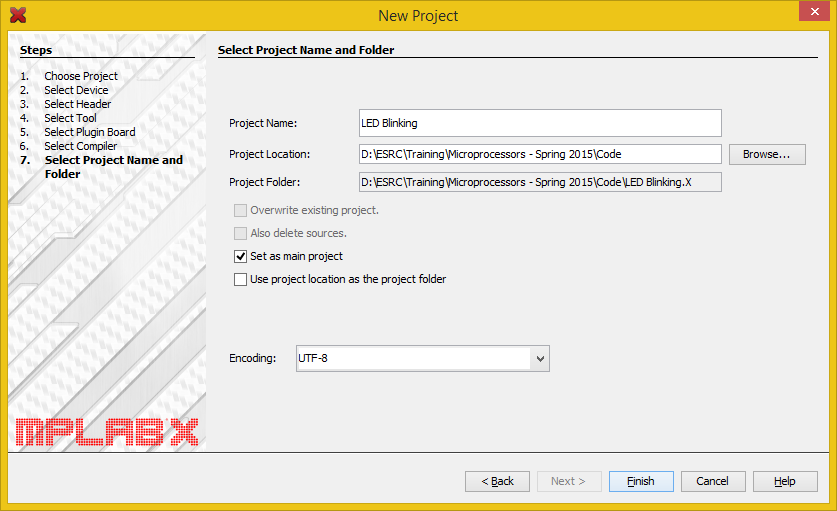
Trong giao diện lựa chọn thiết bị, chọn vi điều khiển 8-bit dòng trung (**Mid-Range 8-bit MCUs**), chip **PIC16F887** rồi bấm **Next**.



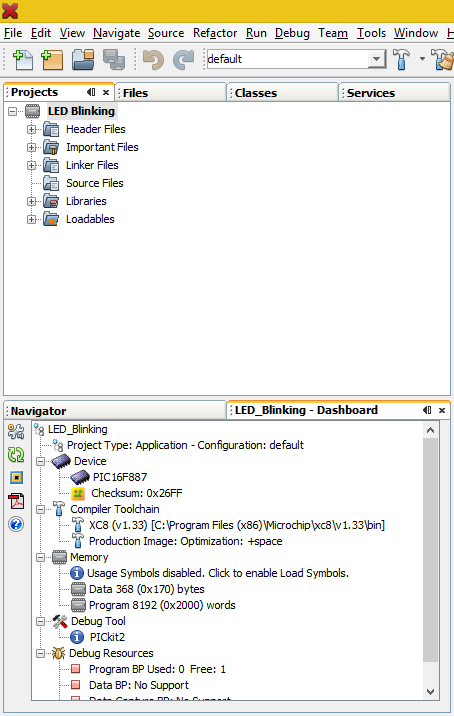
Trong giao diện chọn mạch nạp và debug, chọn **PICkit2[[1]](#footnote-1)** rồi bấm **Next**.



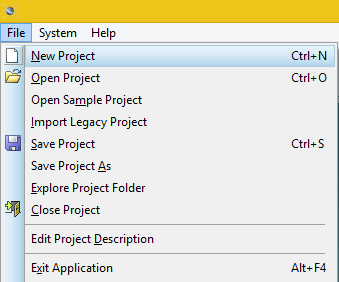
Trong giao diện chọn trình biên dịch, chọn **XC8** rồi bấm **Next**. Nếu XC8 đã được cài thành công, đường dẫn tới thư mục của XC8 sẽ được hiển thị như trong hình.



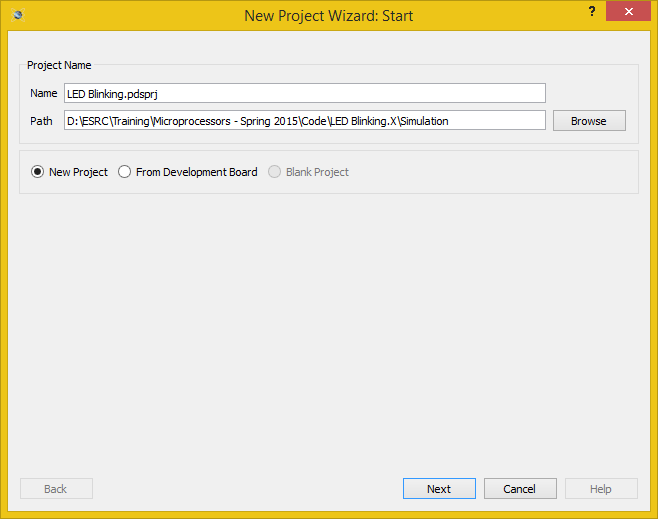
Cuối cùng, nhập các thông tin về project cần tạo, bao gồm tên, thư mục chứa project. Ở đây ta tạo project với tên **LED Blinking** ở một thư mục nào đó, sau đó bấm **Finish**.

Sau khi project đã được tạo thành công, ta có thể sẵn sàng lập trình cho PIC. Các thông tin cơ bản về project được hiển thị ở **Dashboard** và các file chứa trong project được hiển thị ở **Projects**.

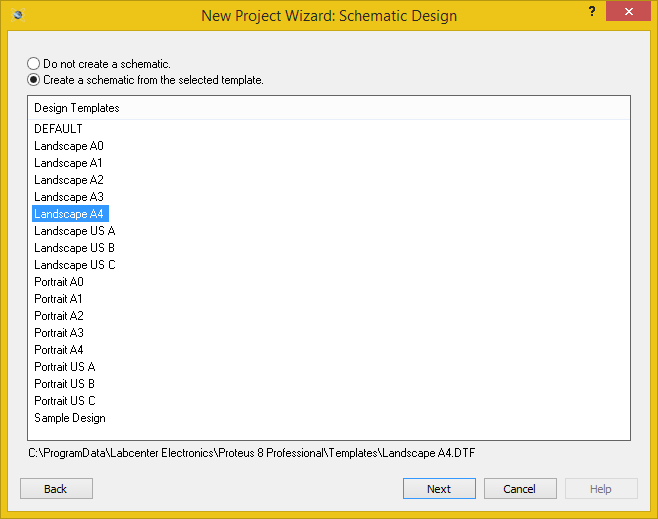
# Tạo project mô phỏng trên Proteus



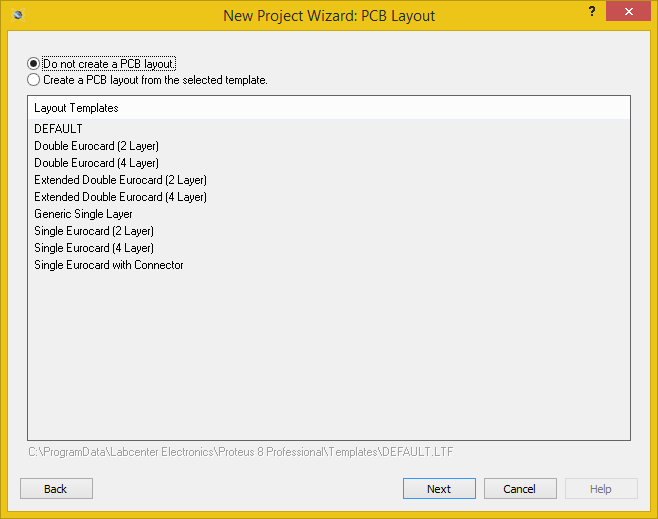
Sau khi khởi động vào Proteus, vào **File > New Project** để tạo project mới.



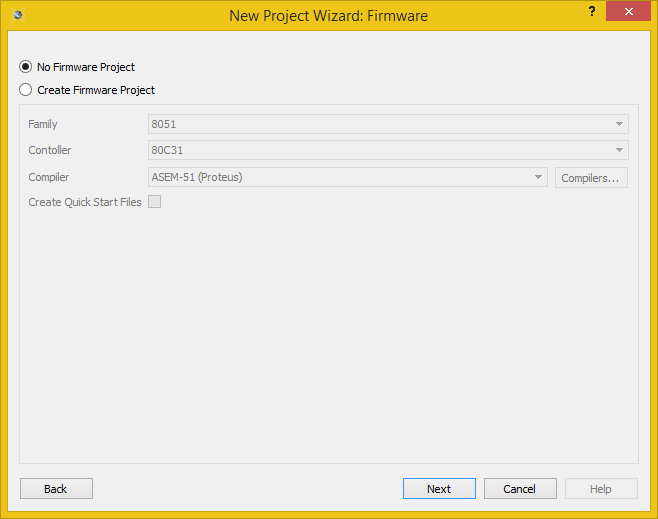
Trong giao diện hiện ra, đặt tên project là **LED Blinking** rồi chọn đường dẫn tới thư mục nào đó. Ở đây ta tạo một thư mục con **Simulation** trong thư mục project của MPLABX đã tạo, sau đó bấn **Next**.



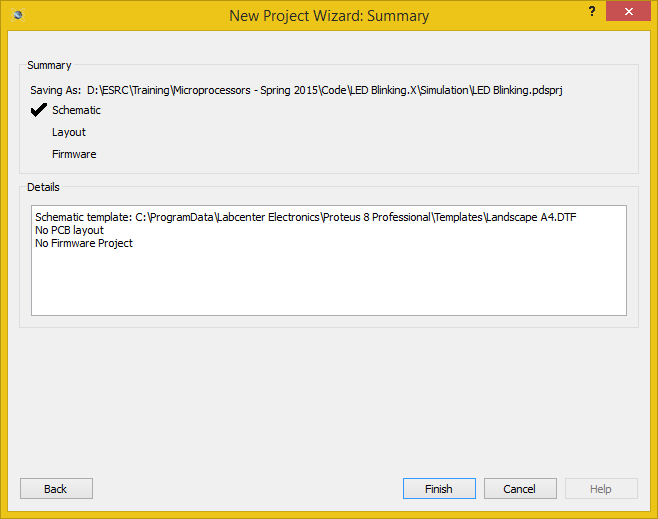
Trong giao diện chọn thiết kế sơ đồ nguyên lý, chọn **Create a schematic from the selected template** và chọn giấy A4 ngang **Landscape A4** rồi bấm **Next**.



Do không sử dụng chức năng vẽ mạch in trong Proteus, trong phần tiếp theo chúng ta chọn **Do not create a PCB layout** rồi bấm **Next**.

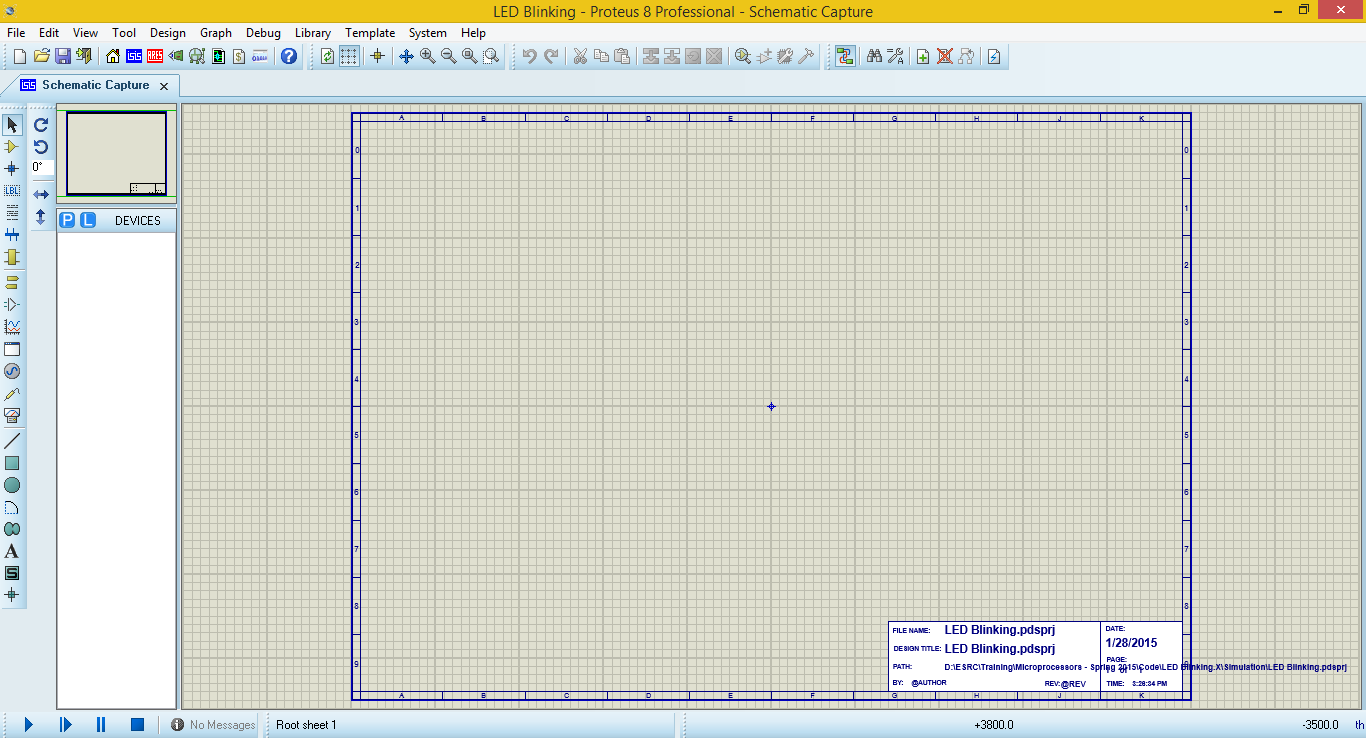


Mặc dù Proteus cũng có chức năng hỗ trợ lập trình, nhưng do đã sử dụng MPLABX, ta bỏ qua phần này bằng cách chọn **No Firmware Project** rồi bấm **Next**.

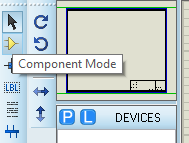


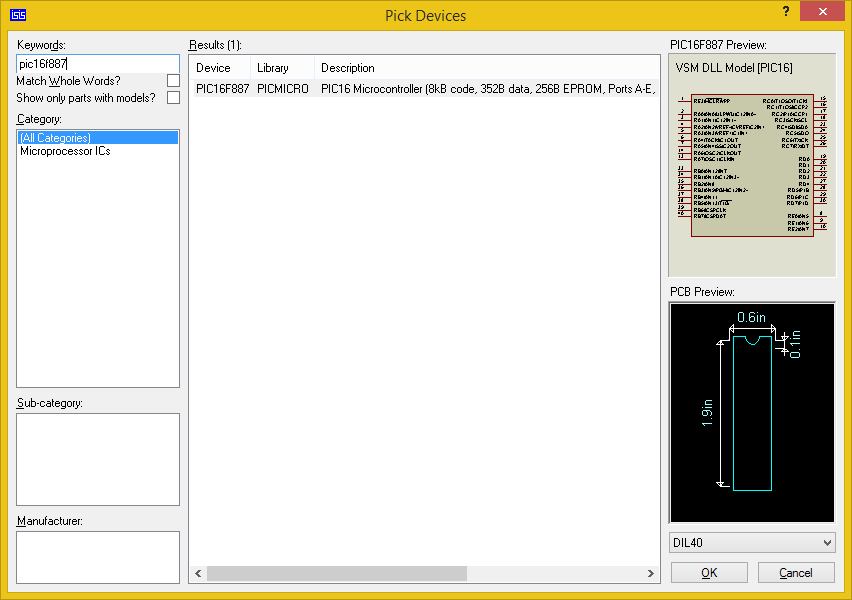
Cuối cùng Proteus hiển thị thông tin tổng hợp về project đã được tạo (có sơ đồ nguyên lý, không sơ đồ mạch in, không lập trình firmware). Bấm **Finish** để hoàn thành.

# Thiết kế mạch mô phỏng trên Proteus



Sau khi tạo project trên Proteus, giao diện chương trình sẽ hiển thị sơ đồ nguyên lý của mạch.

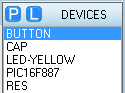
Bấm nút  để chuyển sang Component Mode, sau đó bấm nút  để mở giao diện lấy linh kiện.

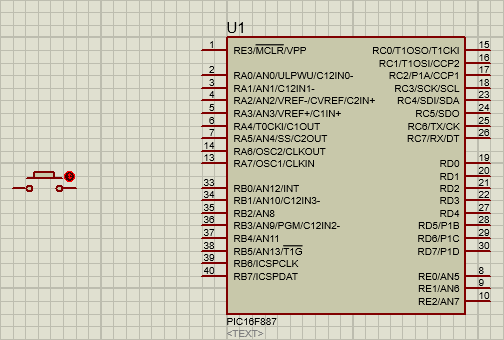


Trong cửa sổ lấy linh kiện, gõ tên linh kiện cần tìm kiếm vào phần **Keywords**. Ta lần lượt lấy các linh kiện sau:

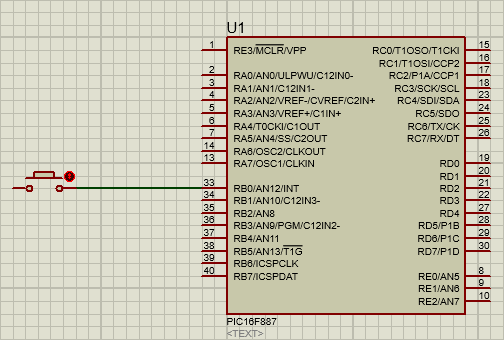
* Vi điều khiển PIC: **PIC16F887**
* Điện trở: **RES**
* Tụ điện: **CAP**
* Đèn LED vàng: tìm với từ khoá **LED** và tìm linh kiện có tên **LED-YELLOW** trong danh sách.
* Nút bấm: **BUTTON**

Với mỗi linh kiện khi đã tìm được, nháy kép vào nó trong danh sách **Results** để thêm vào danh sách linh kiện sử dụng. Sau khi đã lấy đủ linh kiện, bấm **OK** để quay lại giao diện thiết kế mạch.

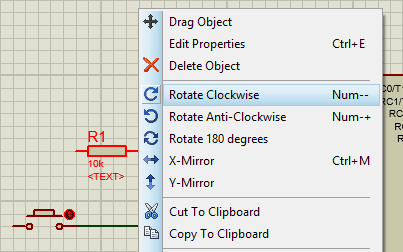
Sau khi lấy linh kiện xong, các linh kiện đã chọn được hiển thị trong danh sách linh kiện. Để đặt linh kiện vào mạch, kích chuột vào linh kiện tương ứng trong danh sách, kích chuột lên sơ đồ mạch rồi kích vào vị trí muốn đặt linh kiện.



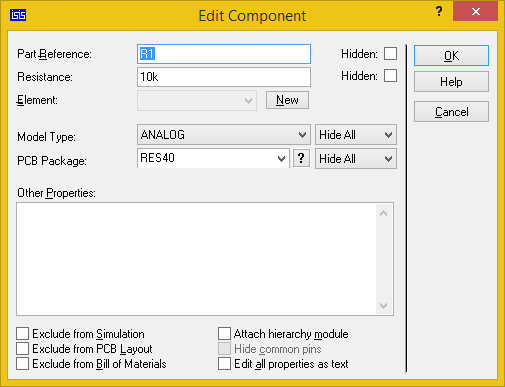
Đặt linh kiện vào mạch giống như hình (nút bấm có vị trí ngang với chân RB0 của PIC).



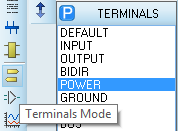
Thực hiện nối dây giữa nút bấm và chân RB0 của PIC bằng cách bấm chuột vào 2 đầu muốn nối dây. Lưu ý di chuyển chuột đến gần chân linh kiện Proteus sẽ hiển thị một chấm vuông, khi đó bấm chuột Proteus sẽ nối dây.

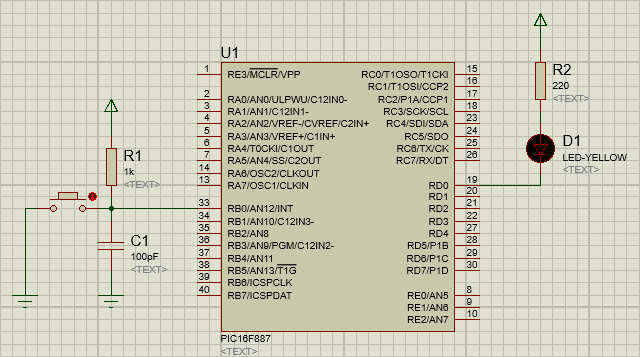


Đôi khi linh kiện cần xoay (ví dụ như điện trở), ta thực hiện bấm chuột phải vào linh kiện đó rồi chọn **Rotate Clockwise**, **Rotate Anti-Clockwise**, v.v... (có biểu tượng minh hoạ).



Để thay đổi tên, giá trị linh kiện và nhiều thông số khác, nháy kép vào linh kiện cần sửa để Proteus hiển thị cửa sổ **Edit Component**.

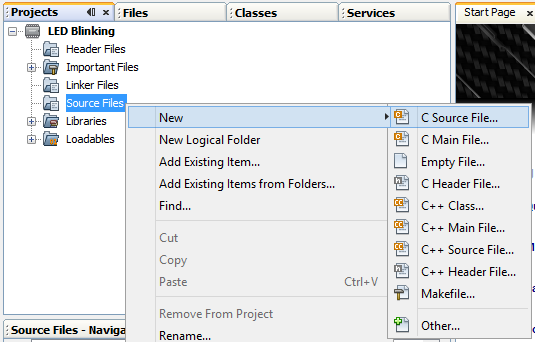
Nguồn, đất... cũng giống như các linh kiện khác, nhưng được sắp xếp ở **Terminals Mode**.



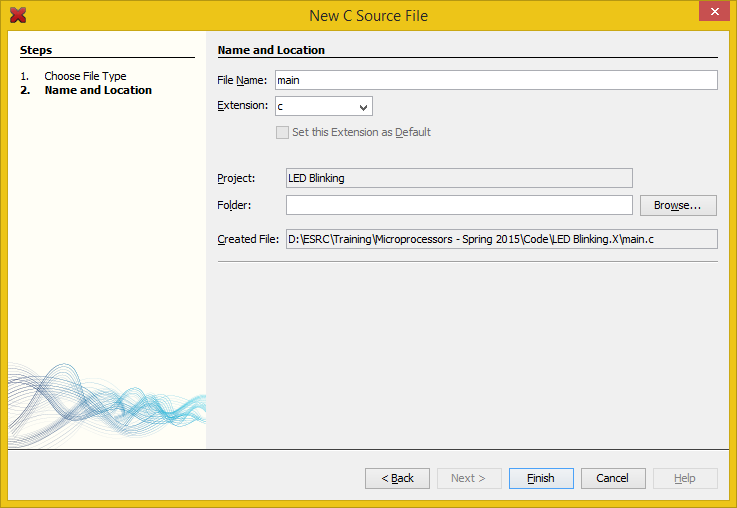
Với các công cụ trên ta có thể dễ dàng thiết kế được mạch mô phỏng như hình vẽ. Ở đây chân RB0/AN12/INT được kết nối với nút bấm, chân RD0 được dùng để điều khiển đèn LED.

# Lập trình điều khiển nháy LED

## Tạo file mã nguồn đầu tiên



Bấm chuột phải vào project **LED Blinking**, nhóm **Source Files**, vào **New > C Source File...** để tạo một file mã nguồn C mới.



Đặt tên file mã nguồn mới là **main** rồi bấm **Finish**.

Mã nguồn một chương trình C cho PIC sử dụng XC8 có dạng như sau:

01 #include <xc.h>

02

03 int main()

04 {

05 while (1)

06 {

07 //

08 }

09

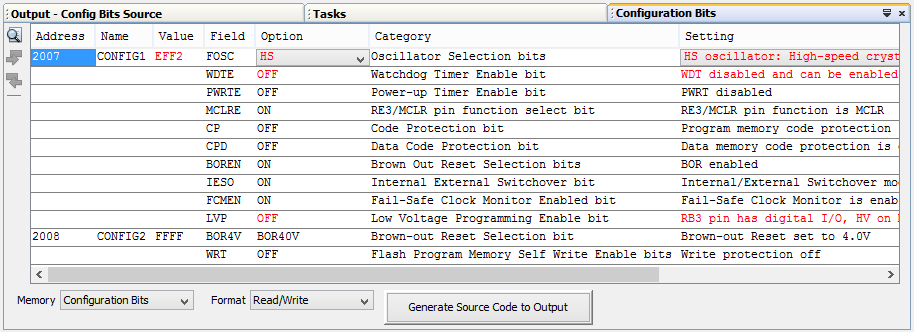
10 return 0;

11 }

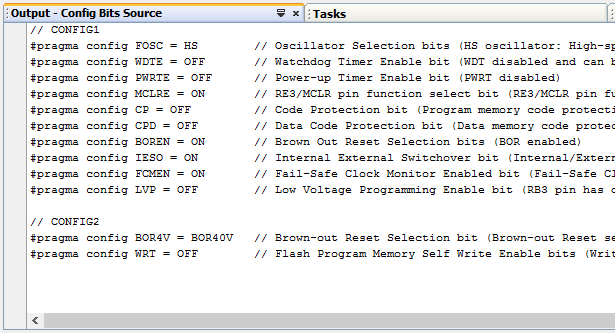
12

## Cấu hình cho PIC

Trước khi lập trình cho PIC, chúng ta phải thực hiện cấu hình cho PIC. MPLABX hỗ trợ tạo mã cấu hình tự động, bằng cách vào **Window > PIC Memory Views > Configuration Bits**.



MPLABX sẽ mở ra giao diện cho phép set các bit cấu hình cho PIC và sinh mã nguồn tự động. Chức năng của từng bit cấu hình được ghi rất rõ ràng. Ở đây mình set bit như trên hình.



Sau khi set bit xong, bấm **Generate Source Code to Output**. Khi đó đoạn mã cấu hình sẽ được tự động sinh ra. Chép đoạn mã này vào đầu chương trình.

## Xuất tín hiệu ra chân IC

Lấy và xuất tín hiệu ra chân của IC (được gọi là GPIO) là thao tác cơ bản nhất trong lập trình vi điều khiển, được thực hiện thông qua các thanh ghi giá trị và thanh ghi điều khiển.

Đầu tiên, muốn xuất tín hiệu ra chân D0, ta thiết lập chế độ xuất cho chân D0:

TRISDbits.TRISD0 = 0;

Khi đó để đưa tín hiệu mức 1, thực hiện lệnh:

PORTDbits.RD0 = 1;

Để thực hiện nháy LED, ở mỗi bước lặp ta phải đảo giá trị tại chân D0 rồi cho vi điều khiển trễ đi 1s. Để làm được điều này đầu tiên ta phải khai báo tần số hoạt động của mạch ở đầu chương trình (1 MHz):

#define \_XTAL\_FREQ 1000000L

Sau đó dùng lệnh sau để trễ đi 1s:

\_\_delay\_ms(1000);

Trên đây là các lệnh sử dụng để thực hiện các chức năng trong PIC. Sử dụng các lệnh trên kết hợp với kiến thức lập trình C cơ bản để hoàn thành chương trình nháy LED:

01 #include <xc.h>

02

03 /\* Cau hinh PIC \*/

04 /\* Khai bao tan so \*/

05

06 int main()

07 {

08 /\* Khoi tao \*/

09

10 while (1)

11 {

12 /\* Nhay LED \*/

13 }

14

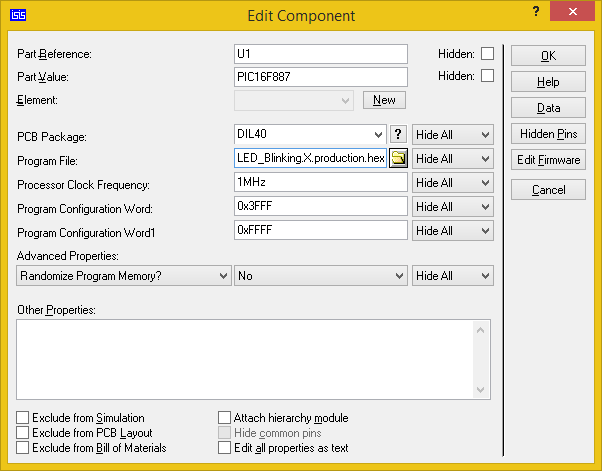
15 return 0;

16 }

17

Sau khi lập trình xong, tiến hành biên dịch chương trình bằng nút . Chương trình sau khi build sẽ nằm ở thư mục dist/default/production.

# Mô phỏng mạch nháy LED trên Proteus



Nháy kép vào con PIC trên sơ đồ mạch, chọn file hex được sinh ra sau khi biên dịch từ phần trước rồi bấm OK. File chương trình nằm ở:

dist\default\production\LED\_Blinking.X.production.hex

Bấm nút  để chạy mô phỏng mạch. Kết quả cho thấy đèn LED liên tục bật tắt mỗi 1s.

# Điều khiển chức năng nháy LED bằng nút bấm

Mỗi khi nút được bấm, tín hiệu trên chân RB0 của PIC thay đổi từ 1 về 0 và ngược lại khi nút được nhả ra. Để bắt được sự kiện, trong vi xử lý có cơ chế ngắt ngoài trên chân RB0. Lưu ý với PIC16F887, chân RB0 ngoài chức năng GPIO, ngắt ngoài còn có chức năng ADC, vì thế cần tắt chế độ Analog của chân RB0 thì chân mới có thể thực hiện ngắt.

# Mở rộng nháy nhiều LED

Mỗi LED được nối vào một chân của PIC, tương ứng với một bit trên thanh ghi giá trị và thanh ghi điều khiển GPIO. Chương trình có thể dễ dàng được mở rộng cho nhiều LED.

1. Hầu hết các mạch nạp PIC bán ngoài thị trường đều là mạch nạp PICkit2 hoặc bản sao của PICkit2 [↑](#footnote-ref-1)