

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA ĐIỆN TỬ VIỄN THÔNG



BÁO CÁO ĐỒ ÁN MÔN HỌC
THỰC HÀNH VI ĐIỀU KHIỂN

HK2 - NĂM HỌC: 2021-2022

TP. HỒ CHÍ MINH, NGÀY 16 THÁNG 07 NĂM 2022

MỤC LỤC

CHƯƠNG 1: THIẾT KẾ PHẦN CỨNG THIẾT BỊ	3
1.1. Giới thiệu:	3
1.2. Giới thiệu phần mềm Proteus:	3
1.3. Thiết kế sơ đồ khối:	6
1.4. Khảo sát các khối:	7
1.4.1 Khối xử lý trung tâm:	7
1.4.2 Khối nguồn:	9
1.4.3 Khối đầu vào:	10
1.4.3.1 Tổng quan cảm biến TC1047A:	10
1.4.3.2 Nguyên lý hoạt động:	10
1.4.4 Khối xung:	11
1.4.5 Khối hiển thị:	12
1.4.6 Khối Hoạt Động:	13
1.5. Sơ đồ nguyên lý mạch:	15
CHƯƠNG 2: XÂY DỰNG PHẦN MỀM	17
2.1. Giới thiệu phần mềm lập trình MPLAB X IDE	17
2.2 Lưu đồ giải thuật của mạch:	22
2.3 Thiết kế code chương trình thiết bị:	23
CHƯƠNG 3: KẾT QUẢ, HƯỚNG PHÁT TRIỂN VÀ BÀI HỌC	30
3.1. Kết quả thực hiện:	30
3.3. Kiến thức học được:	30
BÁO CÁO KỸ NĂNG	31



**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TP.HCM
KHOA ĐIỆN TỬ - VIỄN THÔNG**



**BÁO CÁO ĐỒ ÁN MÔN HỌC
VI ĐIỀU KHIỂN
HK2 - NĂM HỌC: 2021-2022**

**Tên đồ án: ĐIỀU KHIỂN TỐC ĐỘ CẢNH QUẠT THÔNG QUA NHIỆT ĐỘ
VÀ NÚT NHẤN, SỬ DỤNG PIC24FJ128GA010**

STT	HỌ VÀ TÊN	MSSV
1	Trần Ngọc Tài	19200467
2	Nguyễn Lê Minh Hiệp	20200029
3	Đặng Nguyễn Phát Đạt	20200156
4	Đinh Nguyễn Đăng Khoa	20200233
5	Lê Vĩnh Phú	20200306
6	Nguyễn Đình Quý	20200325
7	Phạm Văn Trường	20200389
8	Huỳnh Nguyễn Anh Tuấn	20200397
9	Trần Quang Tuấn	20200408

CHƯƠNG 1: THIẾT KẾ PHẦN CỨNG THIẾT BỊ

1.1. Giới thiệu:

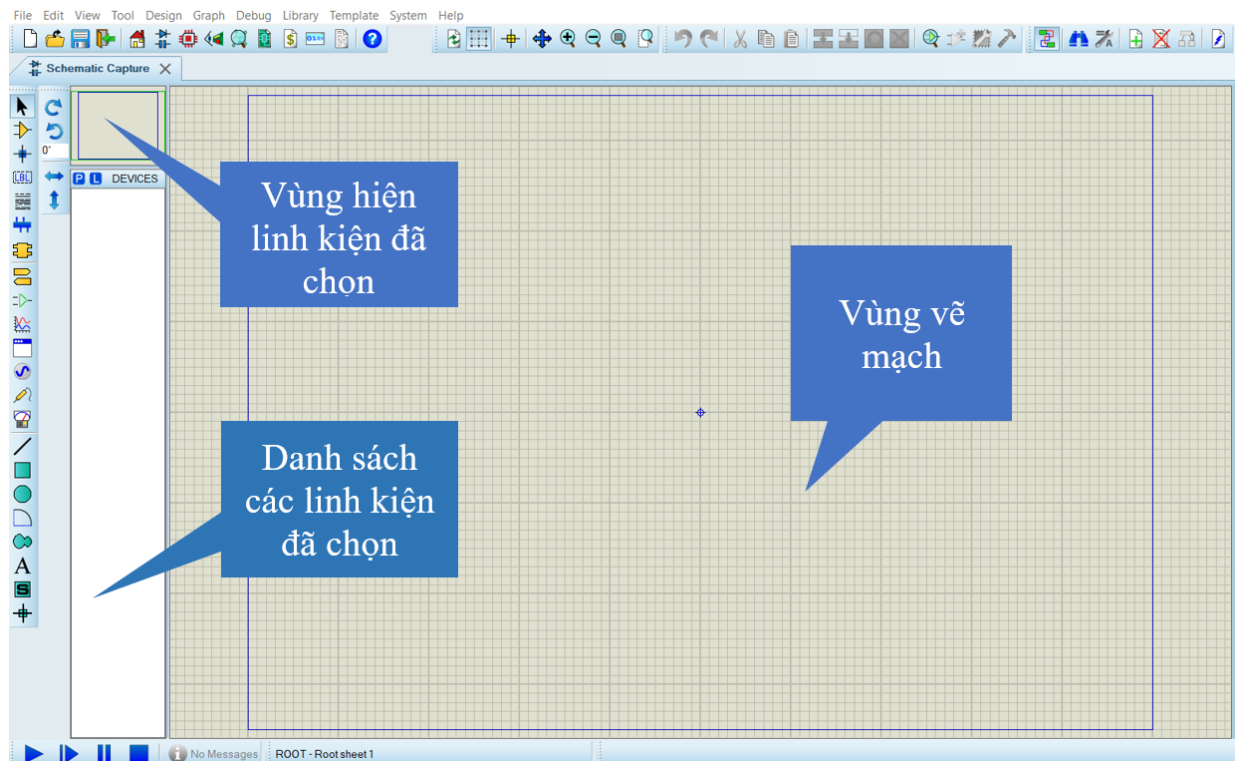
Với việc sử dụng đơn giản, thiết bị được ứng dụng để đo nhiệt độ để điều khiển tốc độ cánh quạt giúp người dùng có thể theo dõi được nhiệt độ của không khí tại nhà, văn phòng hay đặc biệt hơn ứng dụng vào nông nghiệp và với công dụng điều khiển tốc độ cánh quạt ta có thể ứng dụng vào nhiều thứ như: thiết bị tản nhiệt, quạt tản nhiệt, hệ thống thoáng khí, thông gió,

1.2. Giới thiệu phần mềm Proteus:










Proteus là phần mềm của hãng Labcenter dùng để vẽ sơ đồ nguyên lý, mô phỏng và thiết kế mạch điện. Gói phần mềm gồm có phần mềm chính :

- ISIS dùng để vẽ sơ đồ nguyên lý và mô phỏng.
- ARES dùng để thiết kế mạch in.

Vẽ sơ đồ nguyên lý với ISIS



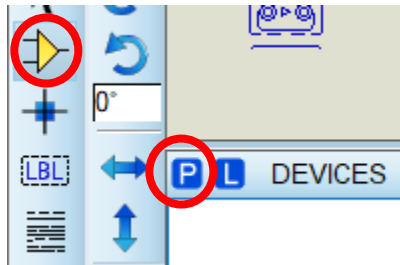
ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

STT	Kí hiệu	Chức năng
1		Section mode: Chức năng này để chọn linh kiện
2		Component mode: Dùng để lấy linh kiện trong thư viện linh kiện
3		Đặt label cho wire
4		Bus
5		Terminal: Chứa Power, Ground
6		Graph: Dùng để vẽ dạng sóng, datasheet, trở kháng
7		Generator Mode: Chứa các nguồn điện, nguồn xung, nguồn dòng
8		Voltage Probe Mode: Dùng để đo điện thế tại 1 điểm trên mạch, đây là 1 dụng cụ chỉ có 1 chân và không có thật trong thực tế Curent Probe mode: Dùng để đo chiều và độ lớn của dòng điện tại 1 điểm trên wire
9		Virtual Instrument Mode: Chứa các dụng cụ đo dòng và áp, các dụng cụ này được mô phỏng như trong thực tế

Cách lấy linh kiện

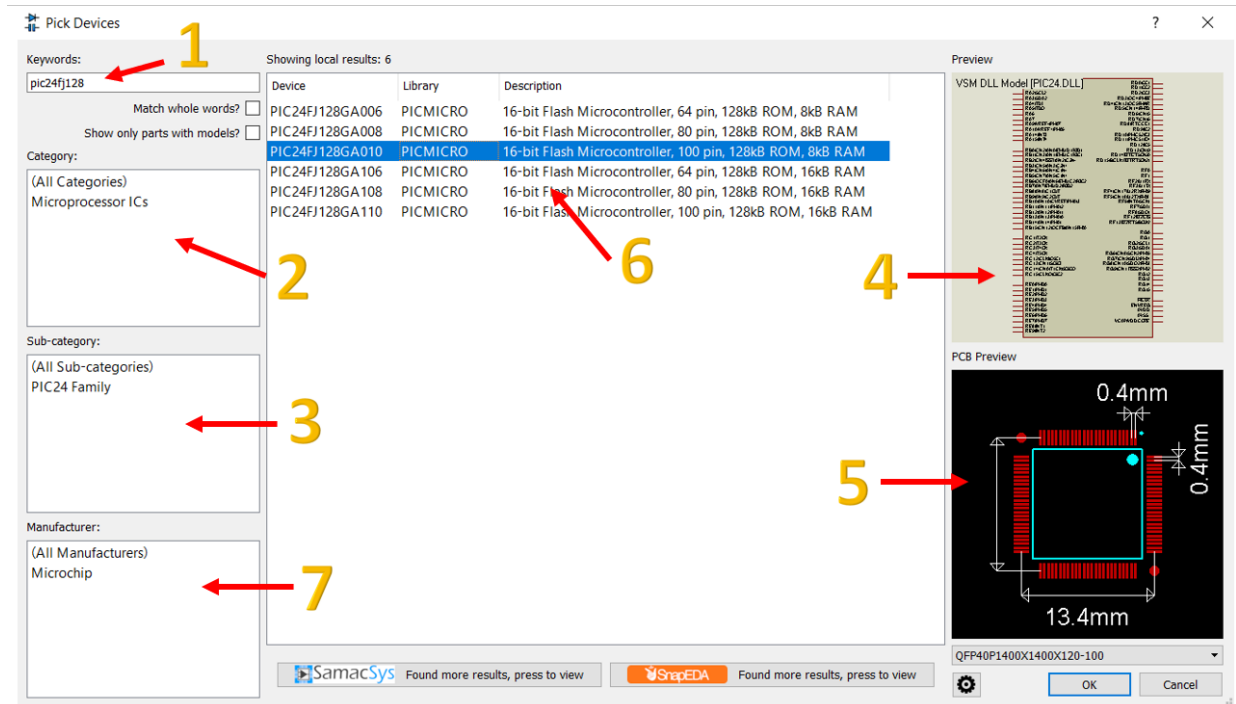
Để lấy linh kiện, nhìn vào phía trái của chương trình và thực hiện như sau:

- Bấm vào biểu tượng Component Mode
- Sau đó bấm vào chữ P hoặc ấn phím tắt P trên bàn phím



ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

Khung chương trình Pick Devices hiện ra

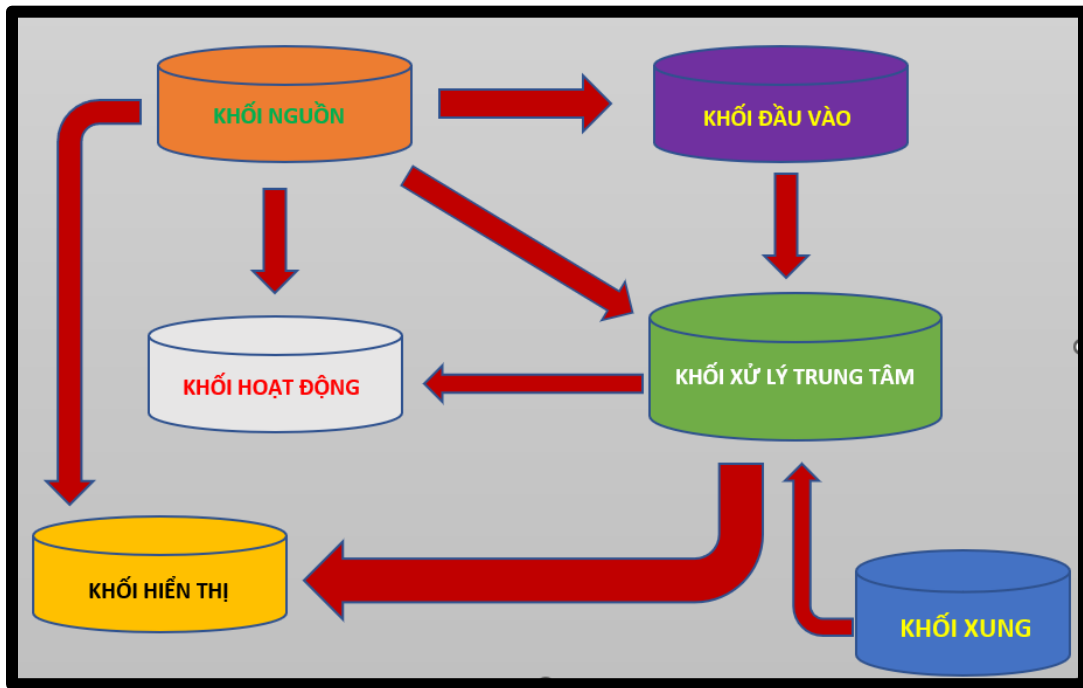


1. Là ô tìm kiếm linh kiện, chỉ cần gõ từ khóa vào, ví dụ như muốn tìm BJT 2N2222 thì gõ 2N2222 như hình vẽ (không phân biệt chữ hoa và chữ thường).
2. Là các nhóm linh kiện liên quan đến từ khóa cần tìm.
3. Là nhóm con của linh kiện, ví dụ như transistor thì có BJT, FET
4. Là ký hiệu (Schematic) trên sơ đồ nguyên lý
5. Là hình dáng trên sơ đồ mạch in (PCB), ví dụ như BJT có nhiều kiểu đóng gói như TO18, TO220, vv ...
6. Là kết quả của việc tìm kiếm linh kiện. Double Click vào linh kiện cần lấy, lập tức linh kiện sẽ được bổ sung vào “bàn làm việc” là vùng màu trắng phỉ bên trái . Xem hình dưới
7. Là tên nhà sản xuất

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

1.3. Thiết kế sơ đồ khối:

Với các yêu cầu đặt ra ở phần đầu, thiết bị bao gồm các khối như sau:



Chức năng và nhiệm vụ của từng khối:

- Khối nguồn: Có chức năng cung cấp điện áp cho các thiết bị hoạt động
- Khối đầu vào (cảm biến): Có chức năng gửi giá trị nhiệt độ và độ ẩm đo được về cho pic.
- Khối xung thạch anh: Có chức năng tạo xung.
- Khối xử lý trung tâm: Có chức năng giải mã tín hiệu của DH11 gửi về và xuất giá trị ra LCD
- Khối hiển thị: Có chức năng hiển thị kết quả đếm dạng số thập phân.
- Khối hoạt động: Có chức năng điều khiển cánh quạt.

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

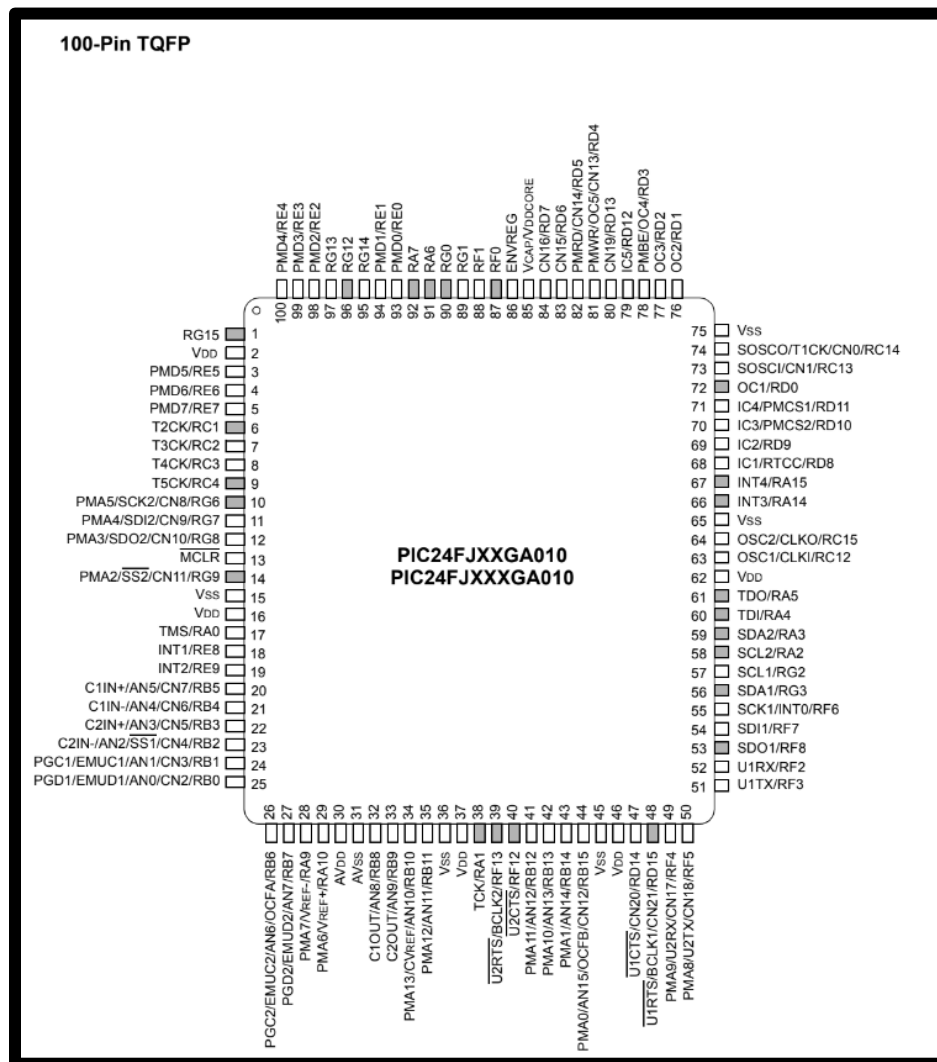
1.4. Khảo sát các khối:

1.4.1 Khối xử lý trung tâm:

KHẢO SÁT VI ĐIỀU KHIỂN PIC24FJ128GA010

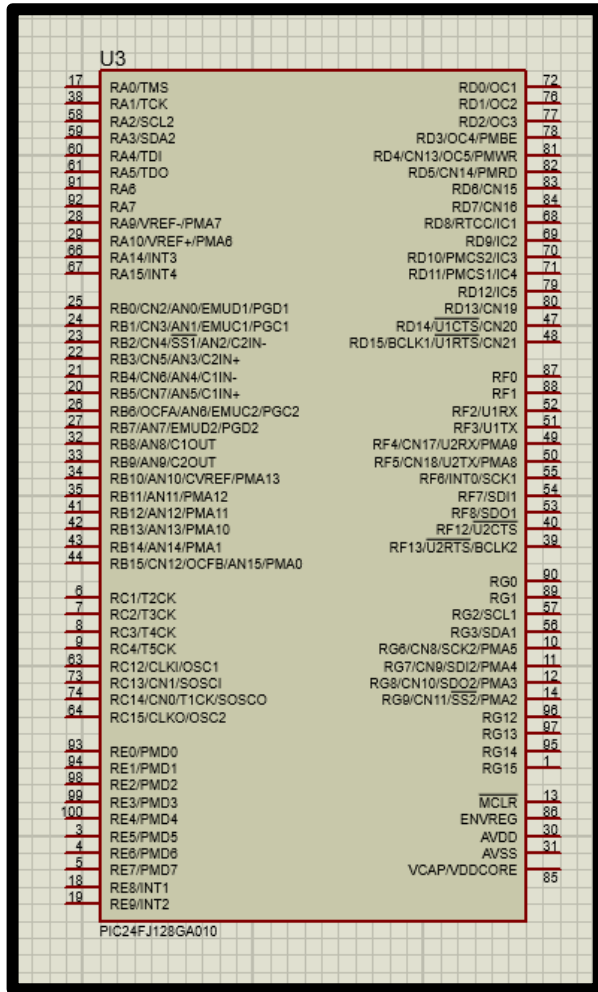
Giới thiệu PIC24FJ128GA010:

- Đây là một dòng vi điều khiển khá thông dụng, có nhiều chức năng, phù hợp với nhiều ứng dụng.
- PIC24FJ128GA010 là loại vi điều khiển 16bit có kiến trúc Harvard và kiến trúc tối ưu hóa trình biên dịch C của hãng microchip.
- Sơ đồ chân với chip loại cắm 100 chân:

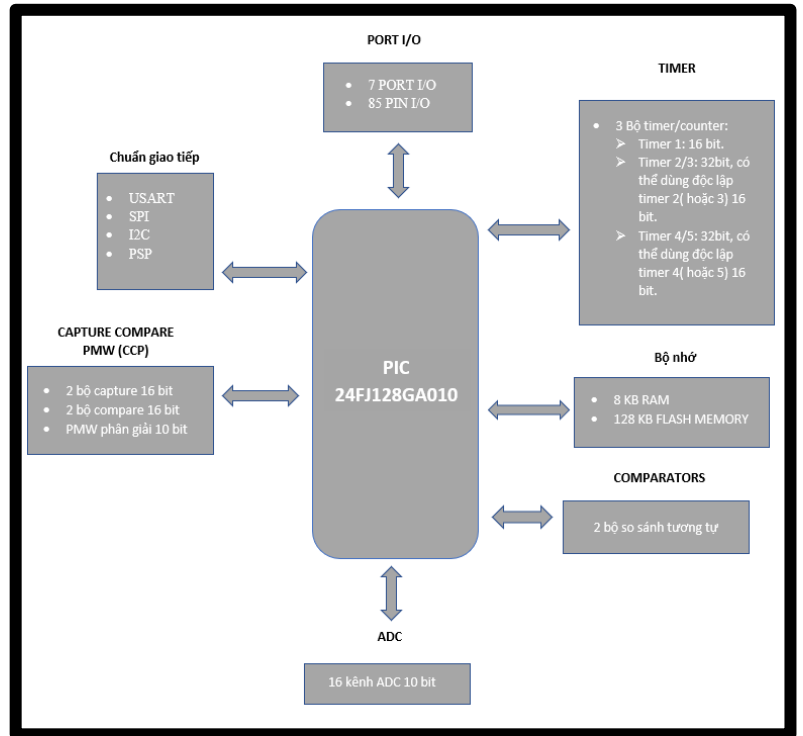


ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

- Sơ đồ chân trong proteus:



- Sơ đồ chức năng:



Nạp chương trình cho PIC:

Để nạp được chương trình cho PIC có 2 cách:

- Nạp trực tiếp dùng mạch nạp: Có 2 loại mạch nạp hay được sử dụng đó là PICKIT và BURNE. Mạch nạp Pickit là hàng chính hãng, độ ổn định cao tuy nhiên chỉ nạp được cho các dòng PIC và DSPIC. Mạch BurnE thì có thể nạp cho rất nhiều loại khác nhau cả PIC và AVR. Tuy nhiên là hàng việt nam sản xuất, độ ổn định có lẽ không cao bằng.
- Nạp qua Bootloader: Có thể tải chương trình Bootloader về và nạp cho Pic 1 lần, sau đó có thể nạp chương trình qua cổng UART rất tiện lợi.

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

Phần mềm lập trình:

Có rất nhiều phần mềm có thể lập trình cho vi điều khiển Pic:

- MikroC PRO for PIC 5.61.
- CCS.
- MPLab.

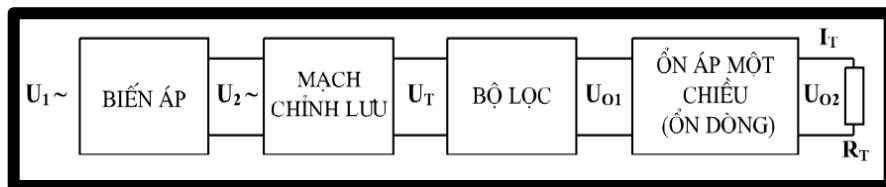
Ngoài ra còn có một số phần mềm hỗ trợ:

- Virtual Serial Port (dùng để tạo cổng COM ảo, dùng khi học về UART).
- Driver PL2303v2: (driver dây USB to COM).
- MH Terminal (Dùng để truyền nhận dữ liệu qua cổng COM).
- Phần mềm nạp chương trình cho PIC (Mạch nạp Pickit V2).
- Phần mềm nạp chương trình cho PIC (Mạch nạp BurnE).
- Phần mềm mô phỏng Protus 8.8.

1.4.2 Khởi nguồn:

Nguồn một chiều có nhiệm vụ cung cấp năng lượng một chiều cho các mạch và các thiết bị điện tử hoạt động. Năng lượng một chiều của nó tổng quát được lấy từ nguồn xoay chiều của lưới điện thông qua một quá trình biến đổi được thực hiện trong nguồn một chiều. Ta sử dụng IC số mắc song song nên áp ra là 5V biến đổi từ nguồn 220V.

Sơ đồ khối nguồn:



Ta có thể dùng cục sạc điện thoại bởi nó đã làm các bước như sơ đồ trên và dùng 2 dây cắm vào external của mạch.

Tuy nhiên, do dùng external giúp mạch có thể đa dạng nguồn vào, thay vì dùng nguồn AC 220 rồi qua các bước hình 1.5, ta có thể dùng nguồn pin tiểu, pin vuông hay pin sạc để cấp nguồn cho mạch hoạt động hay dùng sạc dự phòng. Trong đồ án này, theo dự định chúng em sẽ sử dụng pin sạc Con Ó PVC R6P - Size AA để làm khối nguồn cho đồ án này.

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN



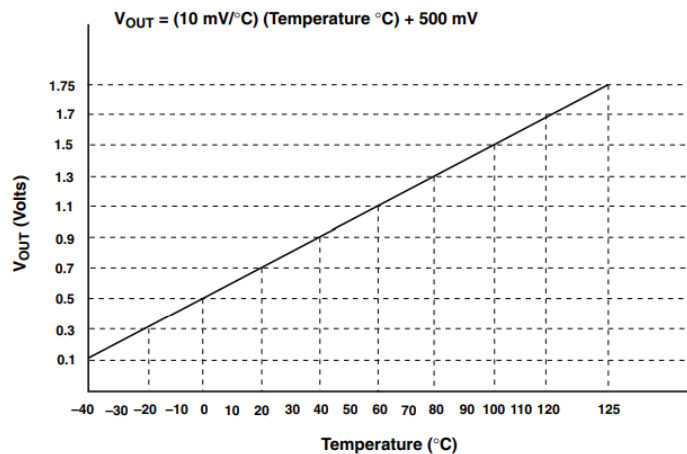
1.4.3 Khỏi đầu vào:

1.4.3.1 Tổng quan cảm biến TC1047A:

TC1047A là một cảm biến kỹ thuật số chuyên dụng dùng để biến đổi tín hiệu nhiệt độ thành tín hiệu điện. Cảm biến này có thể dễ dàng giao tiếp với bất kỳ bộ vi điều khiển vi nào như Arduino, Raspberry Pi, ... để ngay lập tức đo nhiệt độ trong khoảng -40°C đến 125°C .

1.4.3.2 Nguyên lý hoạt động:

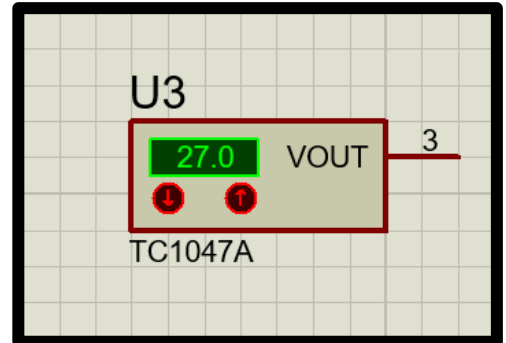
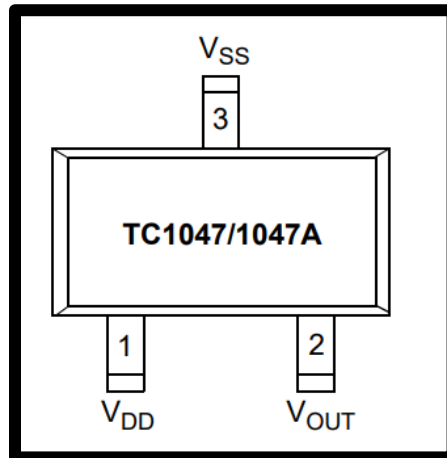
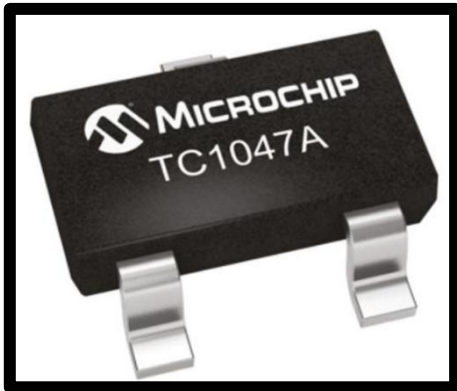
TC1047A có điện áp ngõ ra thay đổi tuyến tính với nhiệt độ tính bằng độ C. Sơ đồ dưới đây sẽ cho ta thấy mối liên quan giữa nhiệt độ với điện áp ngõ ra của TC1047A, độ dốc nhiệt độ được cố định ở $10\text{mV}/^{\circ}\text{C}$ và điện áp ngõ ra ở 0°C là 500mV :



ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

- Khi được cấp nguồn TC1047A ngay lập tức hoạt động và biến đổi thông số nhiệt mà nó thu được bằng cảm biến nhiệt độ bên trong mình thành thông số điện áp và đưa ra ngõ ra.
- Sơ đồ chân:

Số chân	Tên chân	Mô tả
1	V_{DD}	Điện áp cung cấp ngõ vào
2	V_{OUT}	Phần đầu cuối ngõ ra cảm biến nhiệt độ
3	V_{SS}	Phần đầu cuối đất



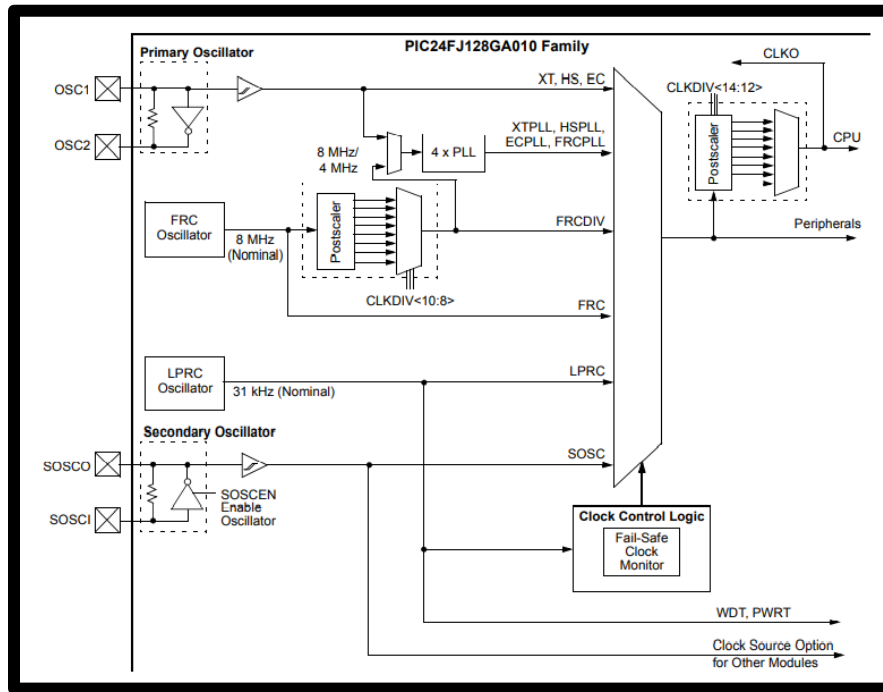
1.4.4 Khối xung:

Thạch anh là bộ dao động khá ổn định để tạo ra tần số dao động cho vi điều khiển. Đa số các mạch điều khiển đèn Led đều dùng thạch anh có thể là Thạch anh 12Mhz, 24Mhz ... mỗi loại sẽ cho ra 1 xung nhịp khác nhau.

- Thạch anh sử dụng rất rộng rãi, hầu như ở đâu cũng có và giá thành thì nó cũng rất rẻ, khoảng 2k/1 con.
- Thạch anh trong điện tử đa phần để tạo ra tần số được ổn định vì tần số của thạch anh tạo ra rất ít bị ảnh hưởng bởi nhiệt độ hơn là các mạch dao động RC....
- Trong Vi điều khiển bắt buộc phải có thạch anh (trừ các loại có dao động nội) vì xét chi tiết thì VDK có CPU, timer, ... CPU bao gồm các mạch logic và mạch logic muốn hoạt động cũng cần có xung clock, còn timer thì gồm các dãy FF cũng cần phải có xung để đếm. Tùy loại VDK mà bao nhiêu xung clock thì ứng với 1 chu kỳ máy, và với mỗi xung clock VDK sẽ đi làm 1 công việc nhỏ ứng với lệnh đang thực thi.

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

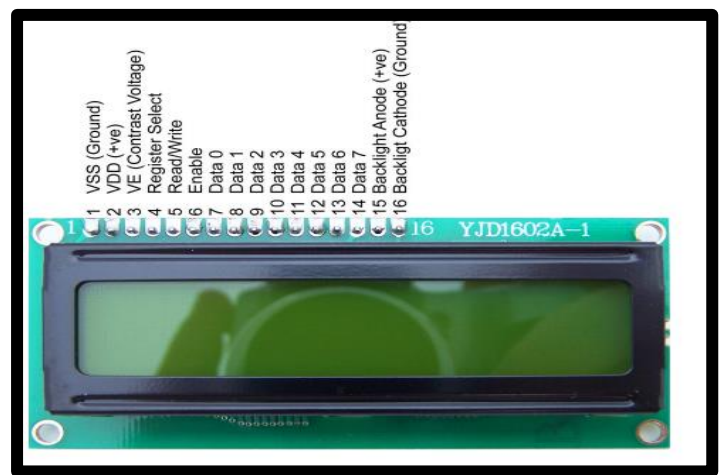
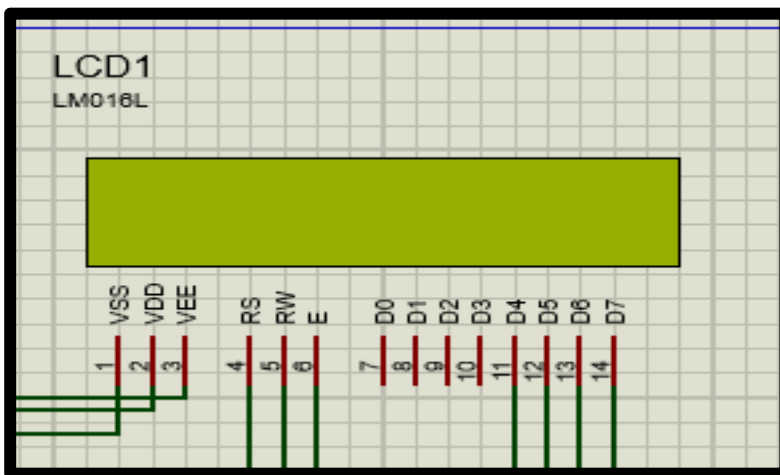
- Ở đây nhóm sẽ dùng khối xung thạch anh có trong PIC24FJ128GA010:



1.4.5 Khôi hiển thị:

Giới thiệu LCD 16×2

Thông số kỹ thuật:



ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

- LCD 16×2 được sử dụng để hiển thị trạng thái hoặc các thông số.
- LCD 16×2 có 16 chân trong đó 8 chân dữ liệu (D0 – D7) và 3 chân điều khiển (RS, RW, EN).
- 5 chân còn lại dùng để cấp nguồn và đèn nền cho LCD 16×2.
- Các chân điều khiển giúp ta dễ dàng cấu hình LCD ở chế độ lệnh hoặc chế độ dữ liệu.
- Chúng còn giúp ta cấu hình ở chế độ đọc hoặc ghi.
- LCD 16×2 có thể sử dụng ở chế độ 4bit hoặc 8bit tùy theo ứng dụng ta đang làm.

1.4.6 Khởi Hoạt Động:

Như đã nói ở trên sau khi nhận được tín hiệu từ cảm biến TC1047A, PIC sẽ điều khiển cổng ngoại vi của mình (RD0/OC1 hay chân 72) để điều chỉnh tốc độ cánh quạt và LED dựa trên số liệu nó nhận được và quạt sẽ được nối với một khóa K dùng để ngắt điện khi nhiệt độ quá 40°C.

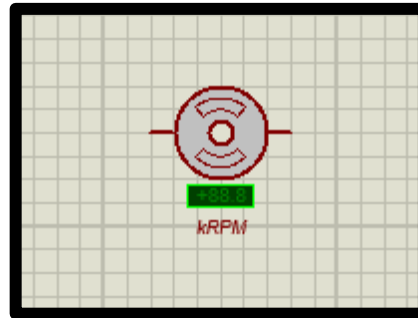
Thực tế tại em sử dụng quạt 12V với thông số:

- Điện áp hoạt động: DC12V
- Dòng tải: 0.25A
- Kích thước: 80 x 80 x 25mm

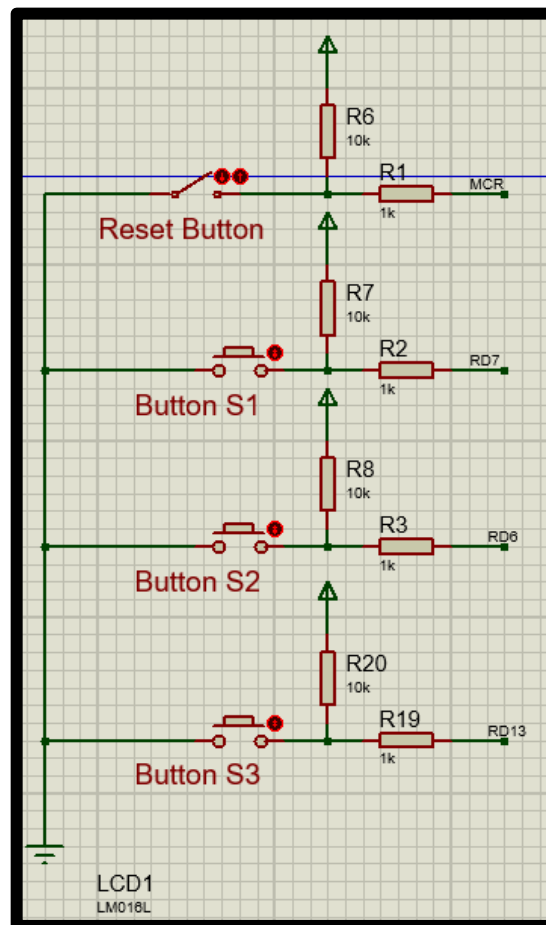


ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

Trong mô phỏng chúng em dùng quạt DC:



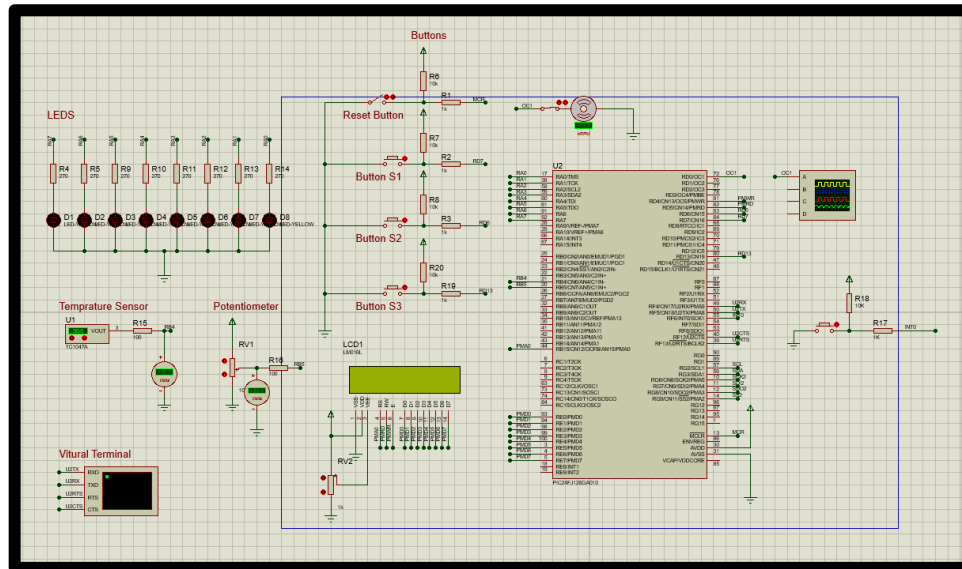
Ngoài ra nhóm em còn làm thêm 3 nút nhấn để điều khiển trực tiếp tốc độ cánh quạt, các nút này sẽ được nối vào các chân PORT D của PIC và lấy nguồn từ khối nguồn:



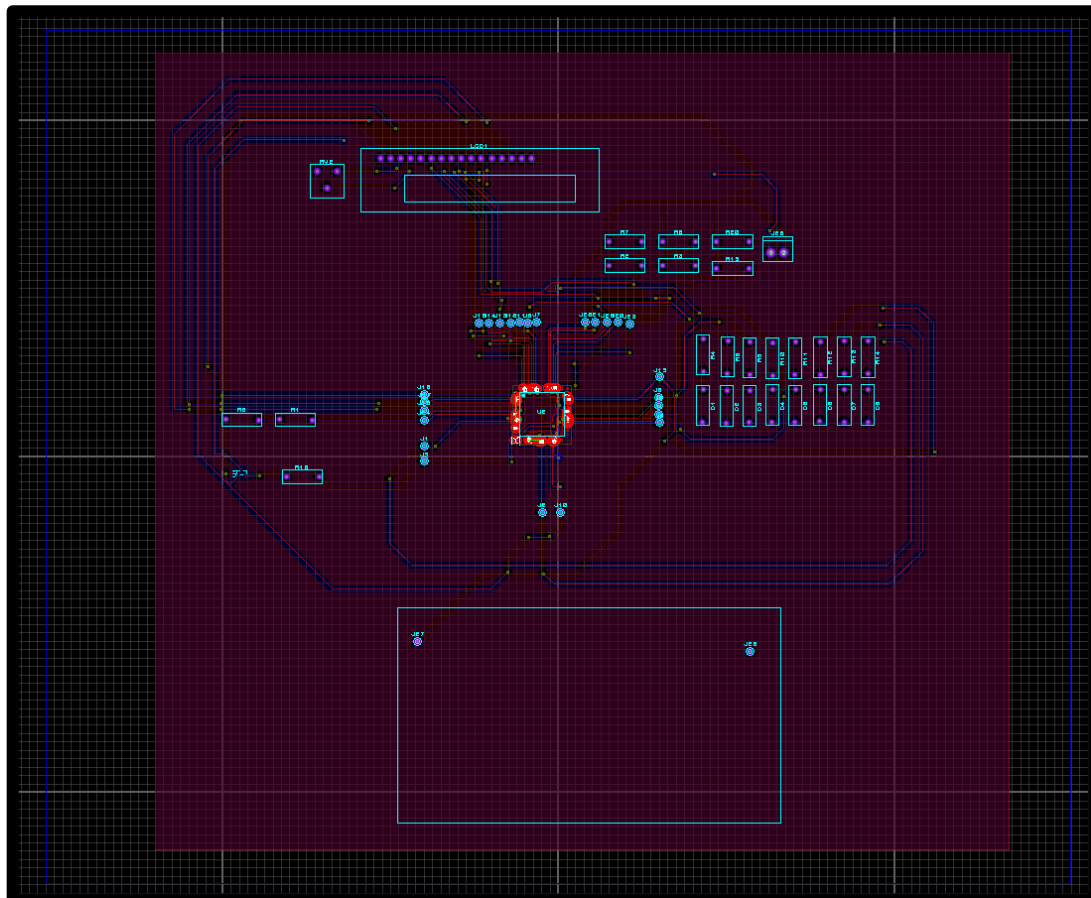
ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

1.5. Sơ đồ nguyên lý mạch:

Proteus:



PCB:

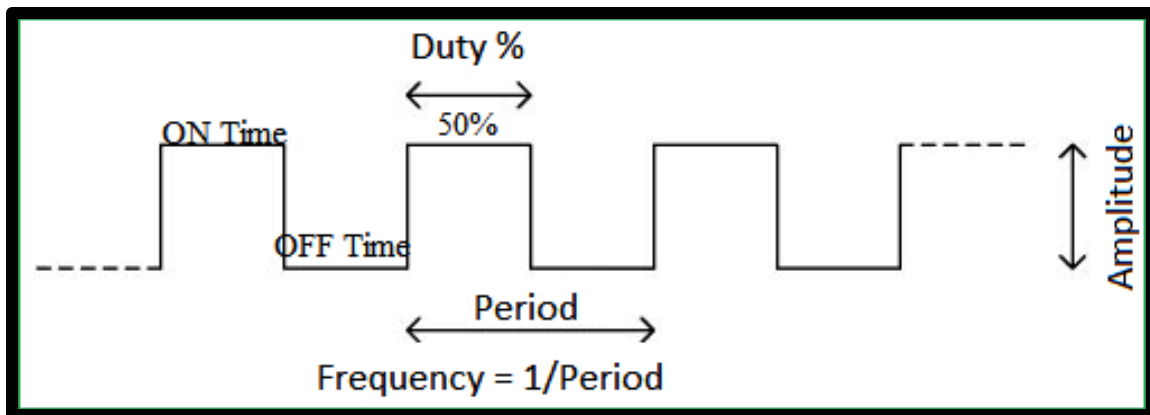
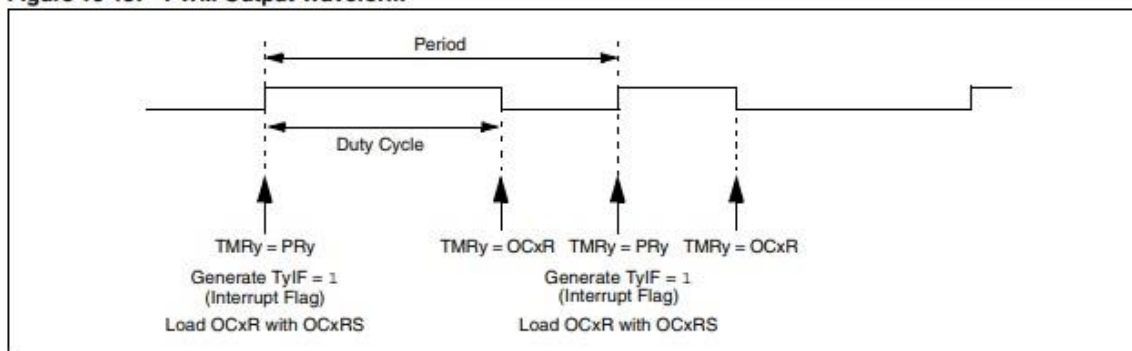


ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

1.6. PWM là gì?

- PWM là tín hiệu số thường được sử dụng trong điều khiển mạch điện
- Tín hiệu được đặt ở mức cao (5v) và thấp (0v) trong một tốc độ và thời gian xác định trước
- Khoảng thời gian mà tín hiệu ở mức cao thì được gọi là thời gian "on time" và khoảng thời gian tín hiệu ở mức thấp được gọi là "Off time"
- Và có 2 tham số quan trọng của PWM đó là:
 - DutyCycle của PWM: Phần trăm thời gian mà tín hiệu PWM vẫn ở mức CAO (on time) được gọi là chu kỳ nhiệm vụ. Nếu tín hiệu luôn BẬT, nó đang ở trong chu kỳ làm việc 100% và nếu nó luôn tắt thì đó là chu kỳ làm việc 0%. $\text{Duty Cycle} = \text{Turn ON time} / (\text{Turn ON time} + \text{Turn OFF time})$
 - Tần số của PWM

Figure 16-15: PWM Output Waveform



CHƯƠNG 2: XÂY DỰNG PHẦN MỀM

2.1. Giới thiệu phần mềm lập trình MPLAB X IDE



MPLAB IDE là phần mềm cho hệ điều hành Windows, thuộc nhóm phần mềm Software được phát triển bởi NA. là phần mềm được hỗ trợ bởi Microchip, dùng để soạn thảo code cho các ứng dụng của PIC. Để lập trình PIC trên phần mềm MPLAB phải cài đặt trình biên dịch C.

Microchip đã phát triển trình biên dịch C16, C18, C30, C32 (C16 cho dòng vi điều khiển 16F, C18 cho dòng vi điều khiển 18F, C30 cho dòng vi điều khiển DsPIC30F). Hiện nay Microchip đã ngừng hỗ trợ MPLAB, chỉ có MPLAB IDE X là được Microchip phát triển. Cùng với đó là các trình biên dịch: (Tháng 9/2012 Microchip cho ra đời 3 phiên bản nâng cấp của Hitech C và C18, C30, C32):

XC8 cho dòng vi điều khiển PIC 8bit như PIC10, PIC12, PIC16, PIC18.

XC16 cho dòng vi điều khiển PIC 16bit như dSPIC30F, dSPIC33F- XC 32 cho dòng vi điều khiển PIC 32 bit.

Ưu điểm: Tương tác cấp độ phần cứng, tính mở trong phương pháp lập trình, dễ dàng phát triển, kết nối với các phương pháp lập trình khác. cho phép người lập trình tối ưu hóa về code. Người lập trình có thể tính toán được thời gian hoạt thực thi chương trình một cách chính xác. Rất tương thích với môi trường lập trình MPLAB và vi điều khiển PIC

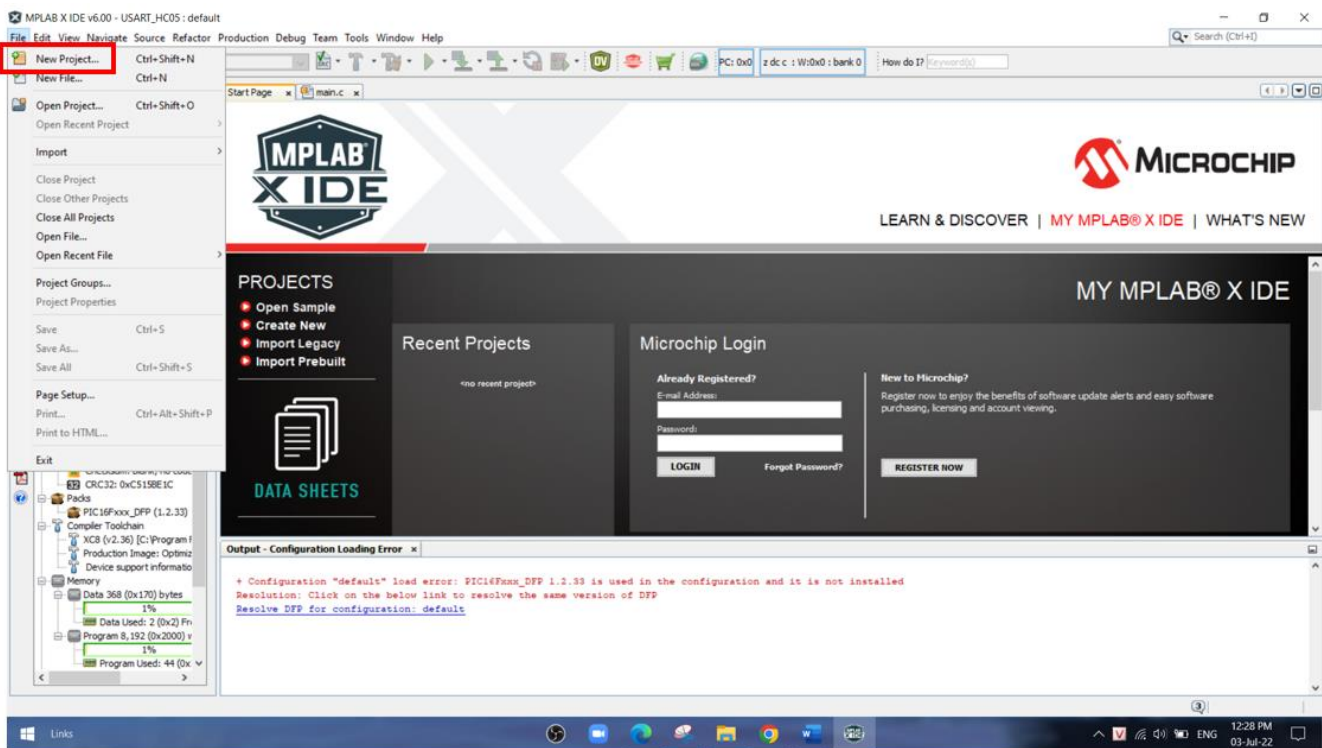
Nhược điểm: Đòi hỏi người dùng không những có kiến thức về C, người lập trình còn phải hiểu rõ về cấu trúc máy tính, bộ nhớ... Hỗ trợ ít thư viện, nên người sử dụng phải tự xây dựng các thư viện.

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

MPLAB IDE (*Integrated Development Environment*) cung cấp cho người dùng một chương trình để phát triển các ứng dụng cho vi điều khiển Microchip nhúng. Thông qua môi trường phát triển, người dùng có thể viết mã cho hệ thống nhúng. Họ cũng có thể chỉnh sửa và gỡ lỗi mã, để đảm bảo rằng các ứng dụng sẽ được tích hợp vào vi điều khiển sẽ thực thi mà không glitches. Basically, người dùng được cung cấp với công cụ mà sẽ cho phép họ thiết kế các mạch vi điều khiển, dựa trên cụ thể vi điều khiển sẽ được sử dụng bởi các nhà phát triển. Khi mạch đã được tạo ra, sau đó người dùng có thể tiếp tục phát triển các chương trình cơ sở, đó là chương trình sẽ kiểm soát như thế nào phần cứng sẽ thực hiện các ứng dụng tích hợp

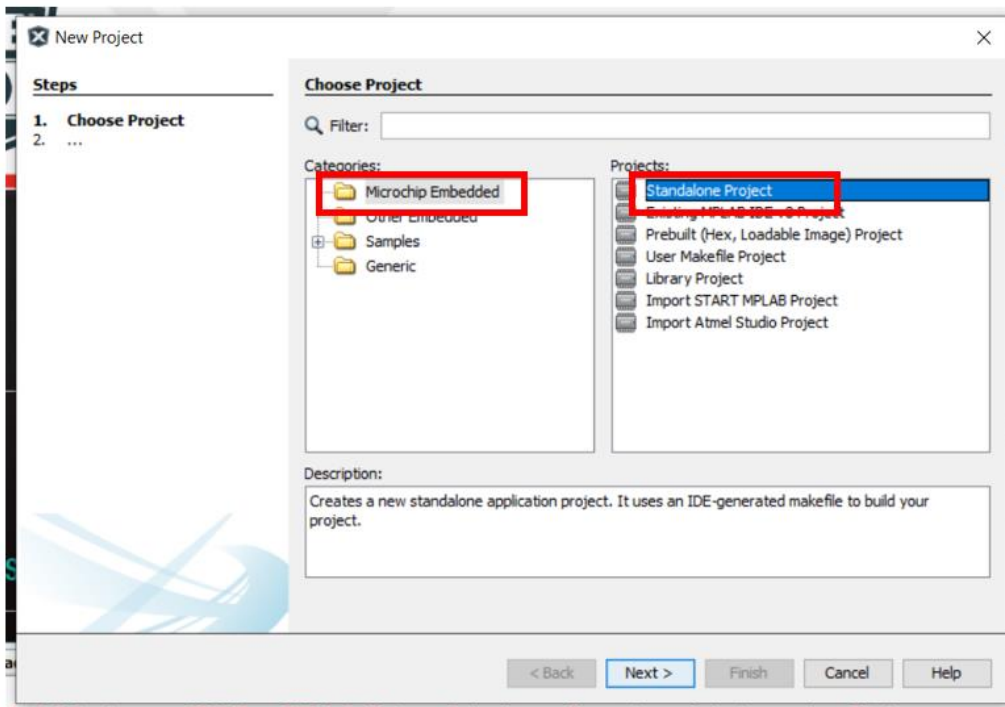
Cách tạo Project dùng PIC24FJ128GA010:

Ở màn hình phần mềm MPLAB-X, chọn File → New Project

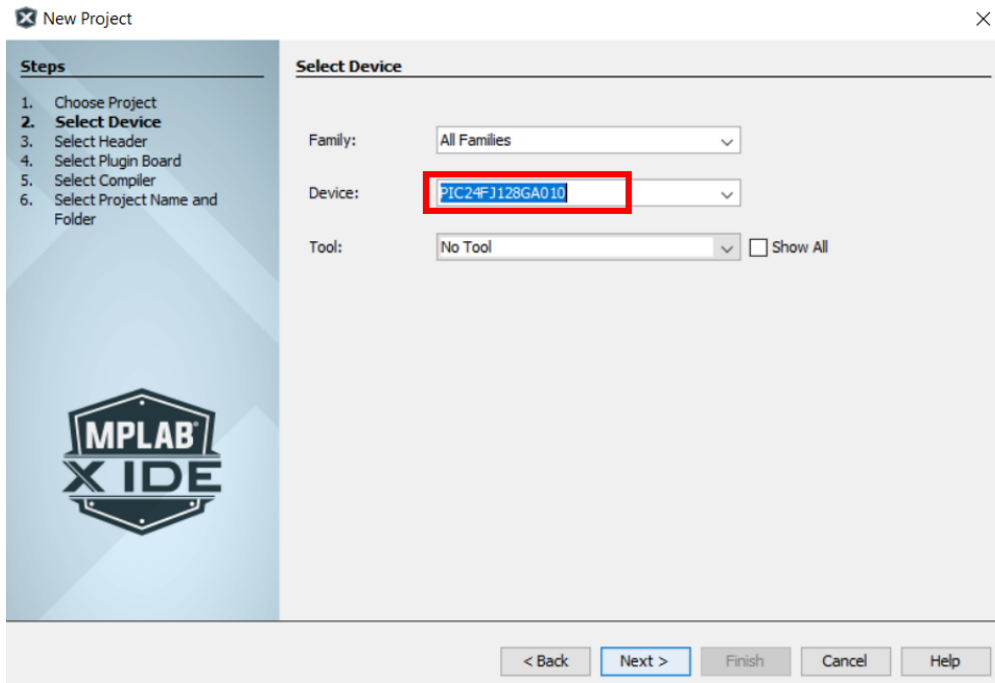


ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

Thiết lập phân loại project là Microchip Embedded và Standalone Project, sau đó chọn Next.

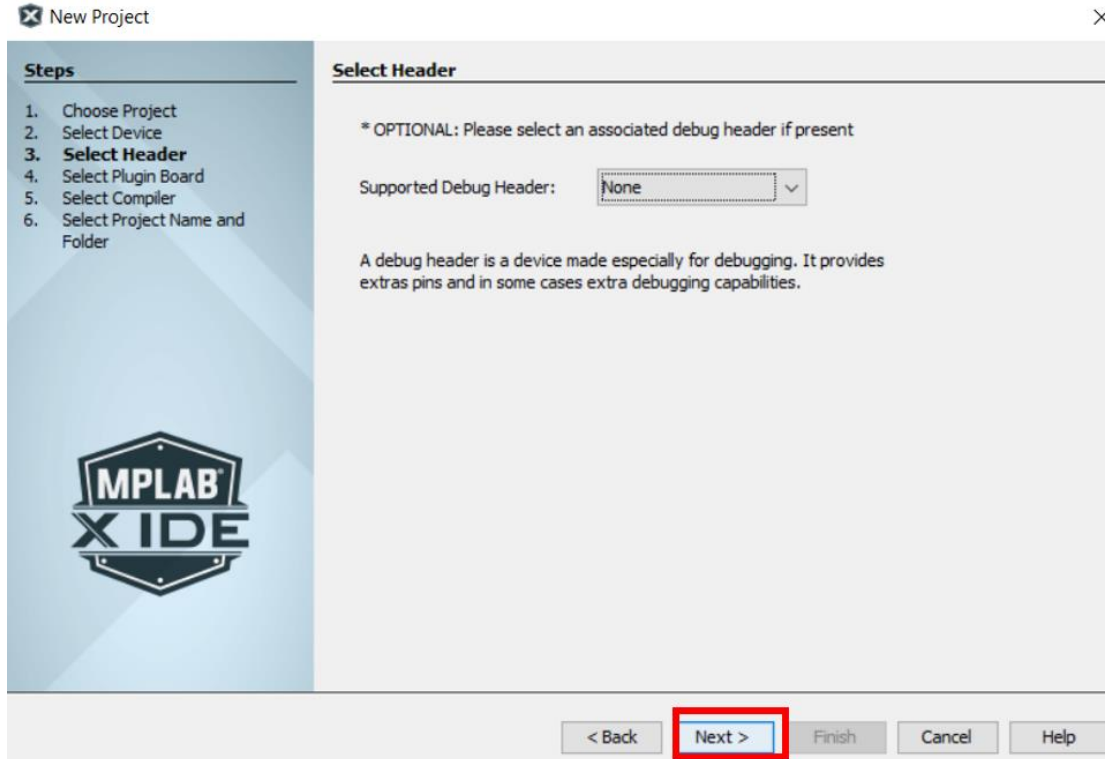


Chọn thiết bị (device) theo họ (family) là All Families thiết bị là PIC24FJ128GA010, sau đó chọn Next.

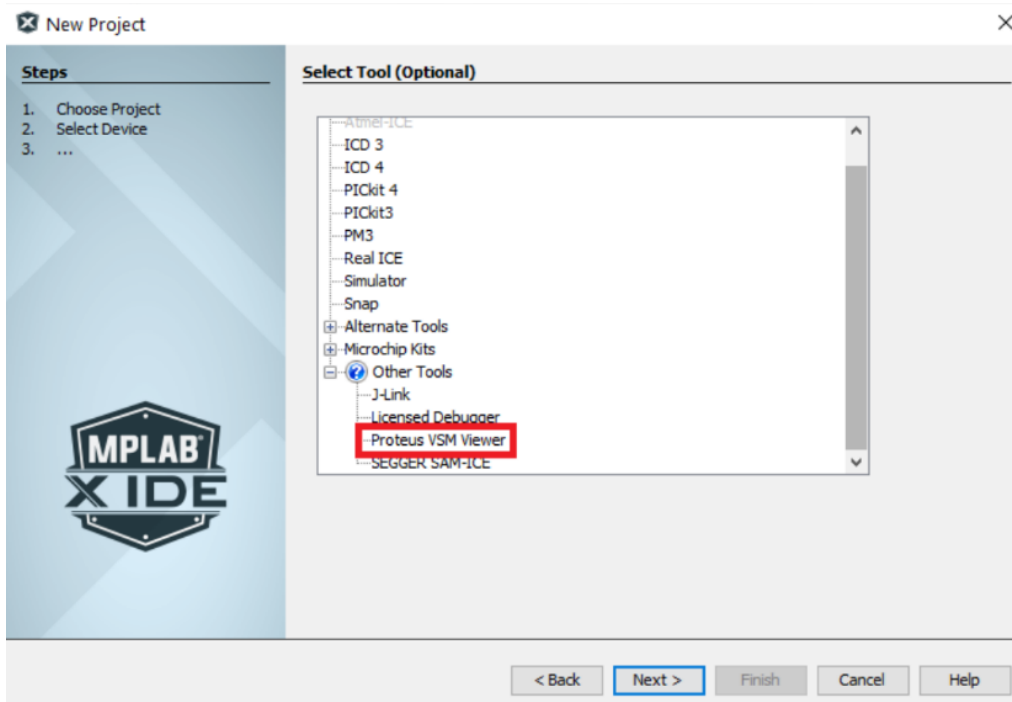


ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

Chọn Next ở bước chọn header

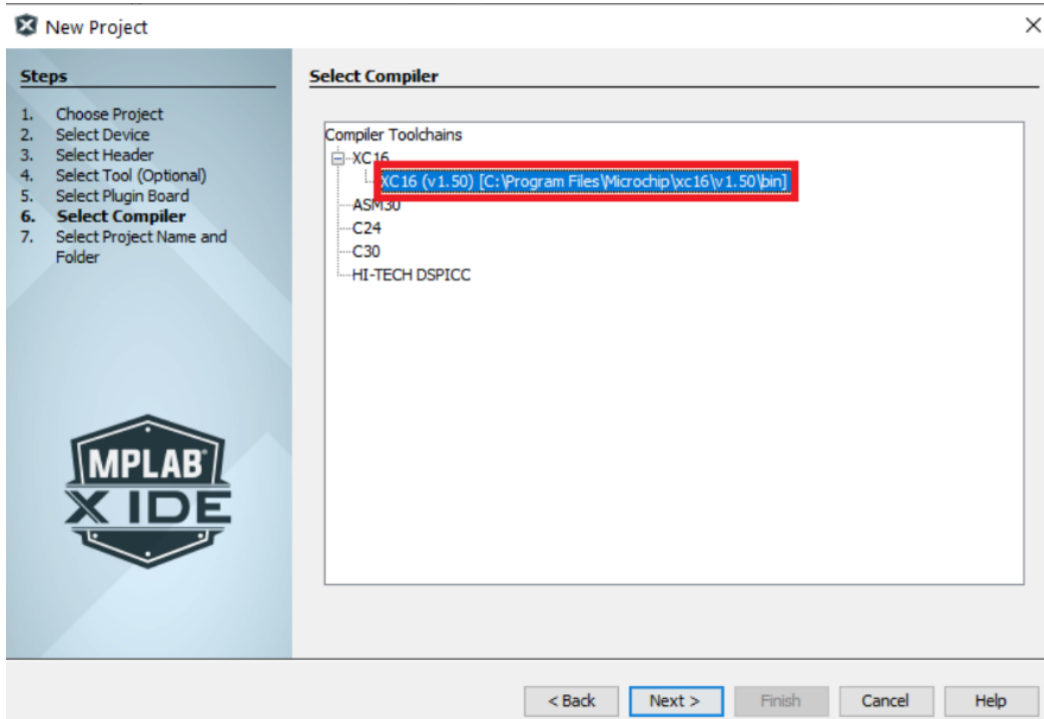


Trong cửa sổ chọn công cụ, chọn công cụ là Proteus VSM Viewer, nhấn Next để tiếp tục.

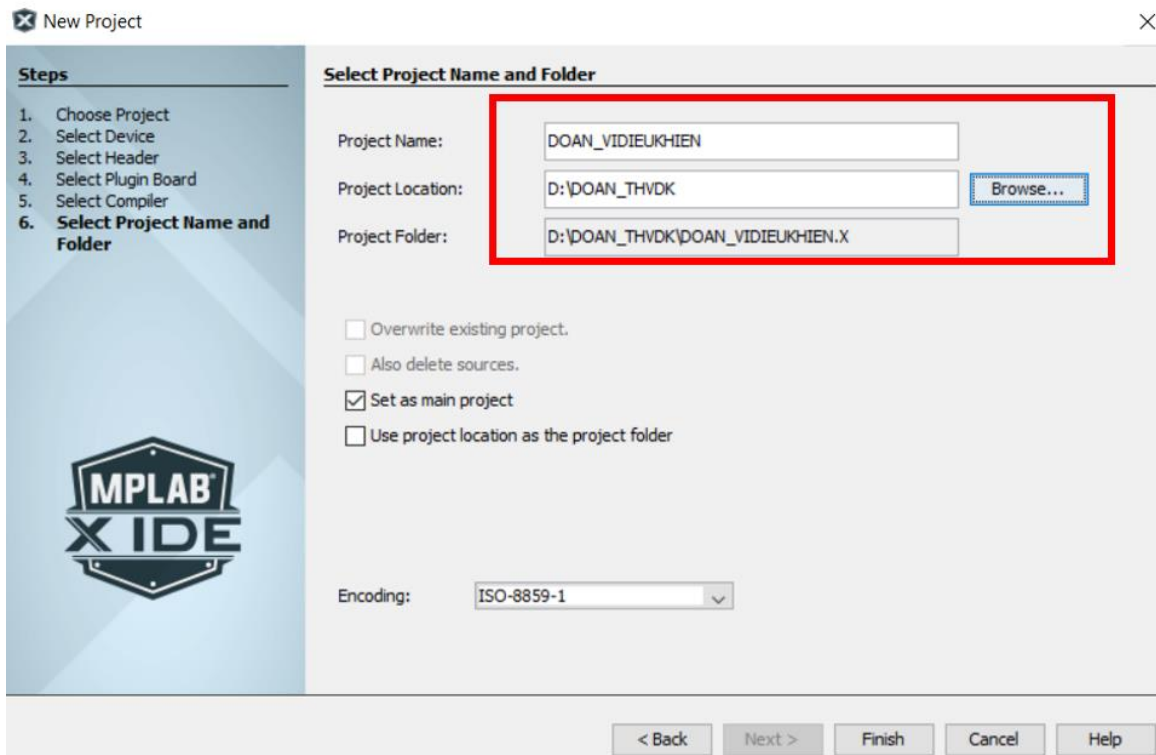


ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

Chọn trình biên dịch là XC16, sau đó nhấn Next.

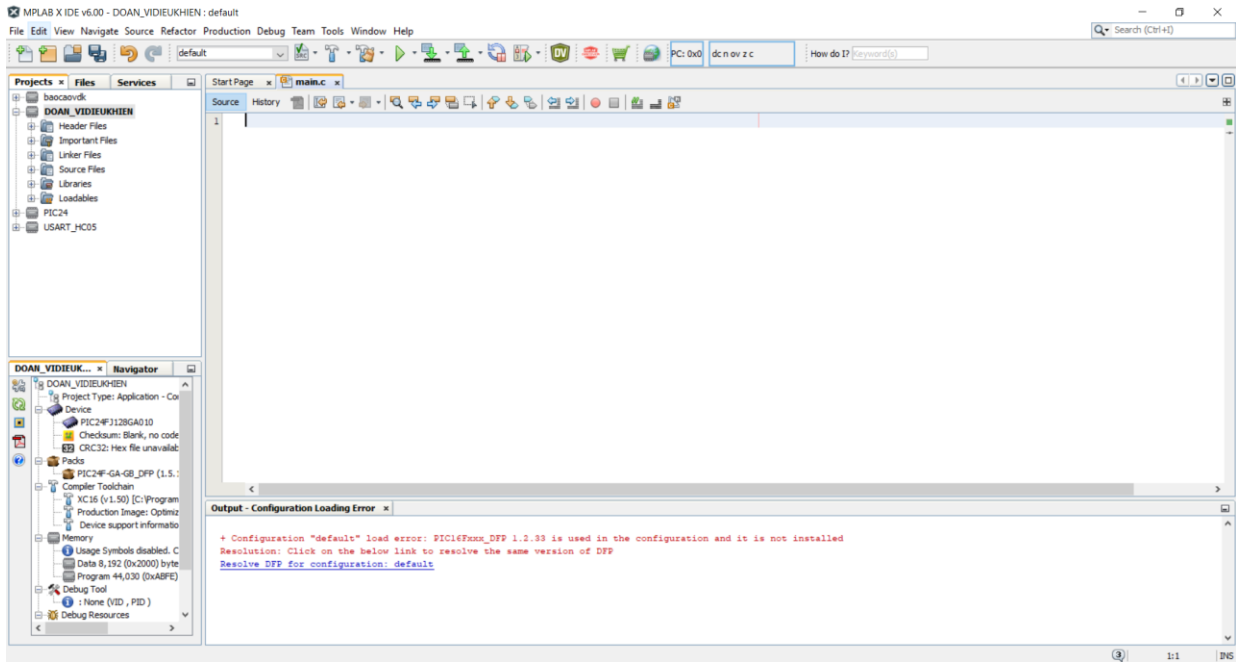


Đặt tên cho project , và chọn đường dẫn, nhấn Finish để hoàn tất quá trình tạo project.

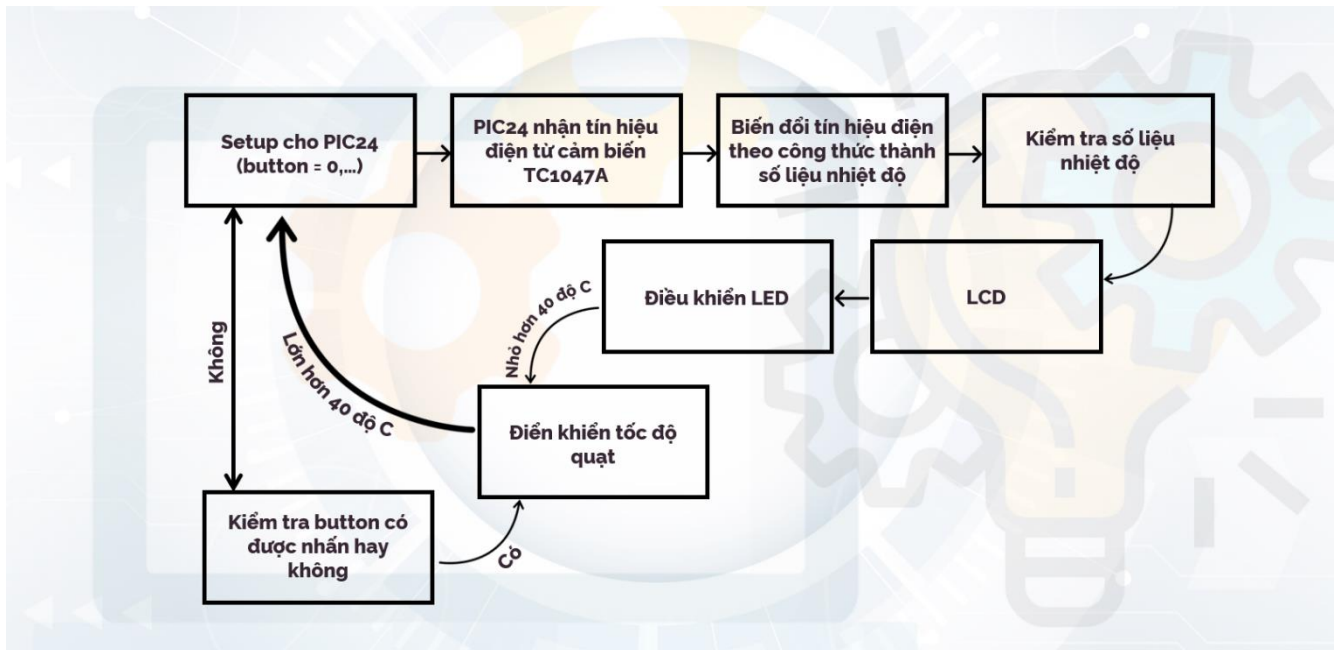


ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

Giao diện làm việc của MPLAB IDE



2.2 Lưu đồ giải thuật của mạch



ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

2.3 Thiết kế code chương trình thiết bị:

Từ lưu đồ trên và những kiến thức đã được học từ các buổi Thực hành Vi Điều Khiển, nhóm chúng em đã xây dựng lên chương trình Code cho thiết bị đo nhiệt độ, độ ẩm:

<pre>// CONFIG2 #pragma config POSCMOD = HS #pragma config OSCIOFNC = OFF #pragma config FCKSM = CSDCMD #pragma config FNOSC = PRI #pragma config IESO = ON // CONFIG1 #pragma config WDTPS = PS32768 #pragma config FWPSA = PR128 #pragma config WINDIS = ON #pragma config FWDTEN = ON #pragma config ICS = PGx2 #pragma config GWRP = OFF #pragma config GCP = OFF #pragma config JTAGEN = OFF</pre>	<p>+ Chọn chế độ hoạt động cho PIC</p>
<pre>#include <xc.h> #include <xc.h> #include "p24FJ128GA010.h" #include <stdio.h> #include <string.h> #define POT 5 #define TEM 4 // 10k potentiometer connected to AN5 input #define CTS_RF12 // Clear To Send, input, HW handshake #define RTS_RF13 // Request To Send, output, HW handshake #define BRATE 68 #define U_ENABLE 0x8008 #define U_TX 0x0400</pre>	<p>+ Khai báo thư viện sử dụng cho chương trình</p> <p>+Chiết áp 10k kết nối với đầu vào AN5</p>

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

<pre>#define BUF_SIZE 128 #define LCDDATA 1 // RS = 1 ; access data register #define LCDCMD 0 // RS = 0 ; access command register #define LCD_COMMAND_CLEAR_SCREEN 0x01 #define LCD_COMMAND_RETURN_HOME 0x02 #define PMDATA PMDIN1 // PMP data buffer #define putLCD(d) LCDwrite(LCDDATA, (d)) #define LCDcmd(c) LCDwrite(LCDCMD, (c)) #define LCDhome() LCDwrite(LCDCMD, 2) #define LCDclr() LCDwrite(LCDCMD, 1) #define delay_32ms() TMR1 = 0; while(TMR1 < 2000); #define delay_162us() TMR1 = 0; while(TMR1 < 100); #define delay_48us() TMR1 = 0; while(TMR1 < 3); #define DELAY() TMR1=0; while(TMR1<9000)</pre>	
<pre>char tr[BUF_SIZE]; int index = 0; int flag = 0; float temperature = 0.0; int count = 0;</pre>	<p>- Khai báo kiểu dữ liệu cho các biến trong chương trình: +cờ = 0 +nhiệt độ = 0 +biến tạm = 0</p>
<pre>void InitializeTimer2For_PWM(void) { T2CONbits.TON = 0; T2CONbits.TCS = 0; T2CONbits.TGATE = 0; T2CONbits.TCKPS = 0b00;</pre>	<p>+Tắt Timer +Giá trị Prescale</p>

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

<pre> TMR2 = 0x00; PR2 = 0xFFFF; T2CONbits.TON = 1; } </pre>	<p>+Đặt chu kì xung PWM</p> <p>+Bật Timer 2</p>
<pre> void Init_PWM(void) { OC1CONbits.OCM = 0b000; // Clear whatever values are in OC1R = 0x0F; OC1RS = 0x0F; OC1CONbits.OCTSEL = 0; OC1R = 0x0F; OC1CONbits.OCM = 0b110; } </pre>	<p>-Hàm khởi chạy biến tạo xung:</p> <p>+Xóa tất cả giá trị</p> <p>+Khởi tạo dutycycle ba đầu</p>
<pre> void SetDutyCycle_PWM(unsigned int DutyCycle) { OC1RS = DutyCycle; } </pre>	<p>-Đặt dutycycle cho PWM</p>
<pre> void LCDinit(void) { // PMP initialization PMCON = 0x83BF; PMMODE = 0x3FF; PMAEN = 0x0001; // init TMR1 T1CON = 0x8030; // wait for >30ms TMR1 = 0; while(TMR1<625); PMADDR = LDCMD; PMDATA = 0b00111000; TMR1 = 0; while(TMR1<1); PMDATA = 0b00001100; TMR1 = 0; while(TMR1<1); PMDATA = 0b00000001; } </pre>	<p>-Hàm khởi tạo LCD</p>

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

<pre>TMR1 = 0; while(TMR1<26); PMDATA = 0b00000110; TMR1 = 0; while(TMR1<26); }</pre>	
<pre>void LCDwrite(int addr, char c) { PMADDR = addr; PMDATA = c; TMR1 = 0; while(TMR1<10); } // LCDwrite</pre>	-Viết lên LCD
<pre>void putsLCD(char *s) { while(*s) putLCD(*s++); } //putsLCD</pre>	
<pre>void initU2(void){ U2BRG = BRATE; U2MODE = U_ENABLE; U2MODEbits.BRGH = 1; U2STA = U_TX; _TRISF13 = 0; _TRISF12 = 1; RTS = 1; } // initU2</pre>	
<pre>int putU2(int c) { while (CTS); while (U2STAbits.UTXBF); U2TXREG = c; return c; } // putU2</pre>	-Ghi lên Virtual Terminal
<pre>void putsU2(char *s) { while(*s) putU2(*s++); } // putsU2</pre>	

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

<pre>void INT0_Interrupt_Init(){ INTCON2bits.INT0EP = 1; _INT0IP = 5; _INT0IF = 0; _INT0IE = 1; }</pre>	-Khởi tạo nút Interrupt
<pre>void _ISR _INT0Interrupt(void){ _INT0IF = 0; putsU2("Wassup\n\r"); putsU2("Nhiet do day: \r\n"); int temp = (int)(temperature*10); putU2(temp/100+48); putU2((temp%100)/10+48); putU2('.'); putU2((temp%100)%10+48); putsU2("do C\r\n"); }</pre>	-Hiển thị lên Virtual Terminal
<pre>void initADC() { AD1PCFGbits.PCFG4 = 0; AD1CON1 = 0x00E0; AD1CSSL = 0; AD1CON2 = 0; AD1CON3 = 0x1F02; AD1CON1bits.ADON = 1; }</pre>	-Khởi tạo cảm biến
<pre>int readADC(int ch) { AD1CHS = ch; AD1CON1bits.SAMP = 1; while (!AD1CON1bits.DONE); return ADC1BUF0; } // readADC</pre>	-Hàm đọc ADC +chọn kênh đầu vào tương tự +bắt đầu lấy mẫu +đợi quá trình chuyển đổi hoàn tất +đọc kết quả chuyển đổi
<pre>int main() { char s[BUF_SIZE]; TRISA = 0; TRISDbits.TRISD6 = 1; TRISDbits.TRISD13 = 1;</pre>	-Hàm chính: +Cấu hình cho PORTA là đầu ra +Chân RD6 là ngõ vào +Chân SD13 là ngõ vào

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

<pre> LCDinit(); initU2(); putsU2("DO AN DIEU KHIEN CANH QUAT!\n\r"); INT0_Interrupt_Init(); initADC(); InitializeTimer2For_PWM(); Init_PWM() int code; while(1){ Timer1_Init(); code = readADC(4); temperature = (5*(float)(code)*1000/1024-500)/10; if (temperature < 31) { PORTA = 0; } else if(temperature >= 31 && temperature <= 33){ PORTA = 15; } else if(temperature > 33){ PORTA = 255; } if (temperature >= 40){ LCDhome(); putsLCD("!WARN HIGH TEMP!"); PORTA = 255; PORTA = 0; } else { int temp = (int)(temperature*10); LCDhome(); putsLCD("!TEMPER: "); putLCD(temp/100+48); </pre>	<p>+Tính toán nhiệt độ</p> <p>+Nhiệt độ lớn hơn 30 thì các chân PORTA =0(LED tắt)</p> <p>+Nhiệt độ >= 31 và <=33 thì các chân RA0 -> RA7 bằng 1 (4 LED đầu tiên sáng)</p> <p>+Nhiệt độ >33 thì tắt cả các chân PORTA=1 (tắt cả các led đều sáng)</p> <p>+Nhiệt độ >=40 in ra lệnh cảnh báo lên LCD và led chớp tắt liên tục</p> <p>+Cảnh báo nhiệt độ quá nóng</p> <p>-Nhiệt độ < 40 in nhiệt độ lên LCD</p> <p>+Clear màn hình, cập nhật nhiệt độ theo thời gian thực</p>
---	---

<pre> putLCD((temp%100)/10+48); putLCD('.'); putLCD((temp%100)%10+48); putsLCD("doC"); } if(PORTDbits.RD7 == 0){ unsigned int DutyCycle = 0x000F; SetDutyCycle_PWM(DutyCycle); PORTA = 0; } else if(PORTDbits.RD6 == 0){ unsigned int DutyCycle = 0x0FFF; SetDutyCycle_PWM(DutyCycle); } else if(PORTDbits.RD13 == 0){ unsigned int DutyCycle = 0xFFFF; SetDutyCycle_PWM(DutyCycle); } else if (temperature < 31){ unsigned int DutyCycle = 0x000F; SetDutyCycle_PWM(DutyCycle); } else if(temperature >= 31 && temperature <= 33){ unsigned int DutyCycle = 0x1FFF; SetDutyCycle_PWM(DutyCycle); } else if(temperature > 33){ unsigned int DutyCycle = 0xFFFF; SetDutyCycle_PWM(DutyCycle); } } delays(3000); } </pre>	<p>+Nếu nút RD7 được nhấn giữ thì quạt tắt</p> <p>+Nếu nút RD6 được nhấn thì quạt quay tốc độ vừa</p> <p>+Nếu nút RD13 được nhấn thì quạt quay tốc độ nhanh</p> <p>+Nếu không có nút nào được nhấn thì tốc độ quạt sẽ được điều chỉnh dựa theo nhiệt độ (< 31 độ thì quạt tắt)</p> <p>+Từ 31 – 33 độ thì quạt quay tốc độ vừa</p> <p>+Trên 33 độ quạt quay tốc độ nhanh</p>
---	--

CHƯƠNG 3: KẾT QUẢ, HƯỚNG PHÁT TRIỂN VÀ BÀI HỌC

3.1. Kết quả thực hiện:

Về phần mô phỏng trên phần mềm PROTEUS:

- Nhóm đã hoàn thành việc thiết kế phần cứng trên Proteus, lập trình phần mềm với MPLAB IDE, khởi tạo và mô phỏng được hoạt động của PIC24FJ128GA010 dùng để điều khiển quạt.

Về phần thiết kế PCB và lắp mạch thực tế:

- Nhóm đã nghiên cứu và thảo luận nhưng vẫn chưa thể hoàn thành mạch PCB đạt yêu cầu. Đối với mạch thực tế thì cần bản PCB cụ thể, cũng như PIC24 không bán ở Việt Nam nên nhóm không có mạch thực tế để DEMO.

3.2. Hướng phát triển:

- Nhóm sẽ cố gắng mua được PIC24FJ128GA010 để hoàn thành 100% đồ án
- Đưa đồ án vào thực tế để sử dụng trong gia đình, lớp học...
- Phát triển thêm hệ thống phun sương cho quạt, thêm những chế độ điều khiển, tùy biến cho hệ thống để phù hợp với các mùa trong năm và các vùng khí hậu khác nhau.

3.3. Kiến thức học được:

- Sử dụng được các phần mềm như Proteus, MBLAB IDE
- Hiểu về các chân và chức năng của vi điều khiển PIC.
- Hiểu về các ngoại vi như LCD, TC1047A,... Khi kết nối với PIC.
- Kiến thức cần thiết khác từ PIC để dùng cho việc thực hiện các đồ án sau này.

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

BÁO CÁO KỸ NĂNG

1. Kỹ năng làm việc nhóm và quản lý dự án

Kế hoạch thực hiện:

Nhiệm vụ	Người thực hiện	Bắt đầu	Kết quả
Họp triển khai	Anh Tuấn	26/05/2022	
Ý tưởng và Duyệt mục tiêu	Cả nhóm	27/05/2022	Đã chốt được phân công và bắt đầu triển khai
Thiết kế phần cứng	Đăng Khoa, Phát Đạt, Minh Hiệp	27/05/2022	Hoàn thành Proteus. PCB chưa hoàn thiện
Xây dựng Code	Ngọc Tài, Đình Quý, Quang Tuấn	29/05/2022	Tương đối đầy đủ, đáp ứng được chức năng đề ra, nhưng cần thêm tối ưu
Nạp Code và Chạy mô phỏng	Cả nhóm	09/06/2022	Thành công, mạch chạy đúng chức năng đề ra
Word	Vĩnh Phú, Văn Trường, Anh Tuấn	16/06/2022	Hoàn thành
PowerPoint	Ngọc Tài	29/06/2022	Hoàn thành
Sửa chữa lỗi sai và tổng hợp	Ngọc Tài	31/06/2022	Hoàn thành

Kế hoạch thực hiện đồ án

ĐỒ ÁN THỰC HÀNH VI ĐIỀU KHIỂN

Tự đánh giá các kỹ năng/kiến thức

Nhóm tự đánh giá tổng hợp các kỹ năng, kiến thức đạt được qua đồ án môn học

STT	Các kỹ năng	Đánh giá ^(*)
1	Kỹ năng làm việc nhóm, giao tiếp, hành xử chuyên nghiệp, khả năng lãnh đạo và làm việc độc lập	B
2	Kỹ năng tư duy phản biện	D
3	Kỹ năng thuyết trình	B
4	Giao tiếp kỹ thuật (viết báo cáo kỹ thuật)	A
5	Kỹ năng tư duy sáng tạo	C
6	Kỹ năng quản lý dự án/thời gian thực hiện dự án	C
7	Hình thành nội dung, xác định vấn đề và kỹ năng giải quyết vấn đề	B
8	Kiến thức, thực nghiệm qua đồ án môn học	B

^(*)Ghi chú: Đánh giá theo mức A/B/C/D (A: Rất tốt, B: Tốt, C: Trung bình, D: Chưa tốt)