موژان میرجلیلی ۹۸۳۱۱۴۰

مفهوم PWM و کاربردهای آن چیست؟

PWM مخفف Pulse Width Modulation است و تکنیکی برای کنترل ولتاژ است. روشی برای تنظیم توان الکتریکیِ داده‌شده به بار، با تغییردادن زمان قطع و وصل شدن منبع توان به بار (در هر سیکل) است. PWM کاربردهای گوناگونی دارد. بخش اصلی PWM، یک سیگنال کنترلی به شکل موج مربعی (پالس) است، به‌طوری‌که دورهٔ کاری (duty cycle) پالس‌ها، در هر دورهٔ تناوب موج (هرسیکل)، قابل تنظیم است. دورهٔ کاری، نسبت مدتِ بالابودن موج مربعی به دورهٔ تناوب آن است و برحسب درصد بیان می‌شود. در واقع این سیگنال، قطع و وصل شدن منبع توان به بار را تعیین می‌کند (مثلاً با کنترل باز و بسته شدن یک سوئیچ الکترونیکی). در مبحث طراحی منابع تغذیه و کنترل سطح ولتاژ، مدولاسیون پهنای پالس، روشی برای کنترل توان بدون نیاز به دفع یا اتلاف هرگونه توان در راه‌انداز (driver) است. در واقع PWM تکنیکی است که به کمک آن می‌توان مقدار ولتاژ و بنابراین، مقدار توان را کنترل کرد.

پاسخ به پرسش‌های دستورچیست؟

- **ابزار دیگری همانند سروو موتور به‌نام stepper وجود دارد. در رابطه با نحوهٔ عملکرد و تفاوت آن با سروو توضیحاتی ارائه دهید.**
  - ✓ داشتن مدار الکترونیکی فیدبک که معمولاً روی موتور دیده می‌شود.
  - ✓ بازده بیش‌تر.
  - ✓ قدرت نامی بیش‌تر.
  - ✓ کاربرد بیش‌تر در صنعت.
  - ✓ داشتن کابل تغذیه مجزا از کابل سیگنالینگ (کینتیکس).
  - ✓ امکان تنظیم زاویه، سرعت زاویه‌ای و شتاب زاویه‌ای.
  - ✓ دَوران یکنواخت (ریپل گشتاور کم).
  - ✓ انواع مختلف الکتروموتور از قبیل موتور بدون جاروبک دی‌سی یا موتور القایی.
  - ✓ کنترل بُرداری.
  - ✓ سروو موتور برای سرعت‌های بالاتر و گشتاورهای بالا مناسب است.
- **در بخش آخر شرح آزمایش به fundamental period و duty cycle اشاره شده است. در رابطه با هر یک توضیح دهید.**

Fudamental period ک.م.م. تمام دوره‌های زمانی است یا همان دوره تناوب اصلی.

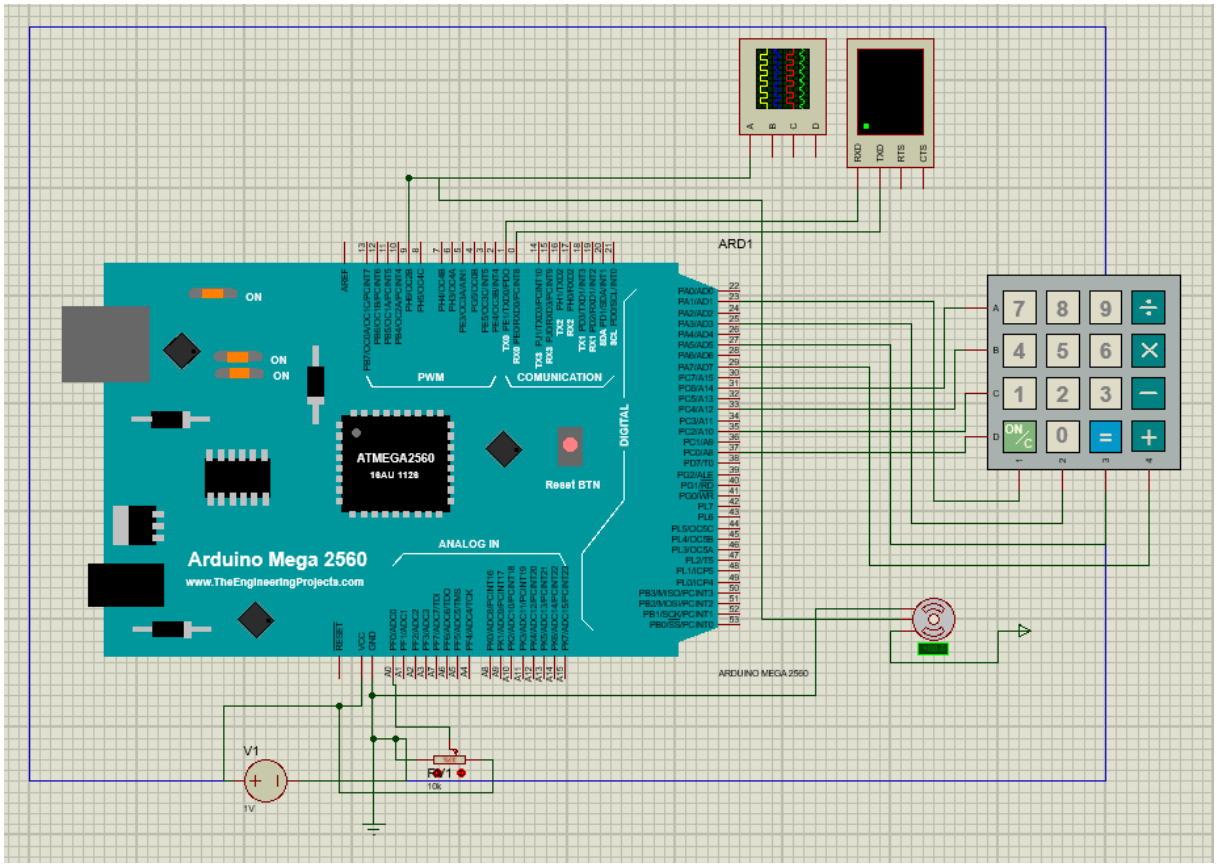
Duty cycle نسبت زمان کاری ماشین به کلِ بازهٔ زمان است. در PWM نسبت مدتِ بالابودن موج مربعی به دورهٔ تناوب آن است و برحسب درصد بیان می‌شود.

کاربردهای سروو موتور چیست؟

سروو موتور متشکل از یک موتور الکتریکی ساده است. سروو موتور، عملگر دورانی یا عملگر خطی است که امکان کنترل دقیق موقعیت زاویه‌ای یا خطی، سرعت و شتاب را فراهم می‌کند. سروو موتورها شامل یک موتور مناسب به همراه یک سنسور خاص برای بازخورد موقعیت (Position Feedback) هستند. سروو موتورها هم‌چنین شامل یک کنترلرِ تقریباً پیچیده هستند که معمولاً خود یک واحد مجزای طراحی شده برای آن‌ها می‌باشد. سروو موتورها کلاس خاصی از موتورها نیستند. با این‌که معمولاً از عبارت سروموتور برای اشاره به موتورهایی استفاده می‌شود که برای استفاده در سیستم‌های کنترل حلقه ـ بسته مناسب هستند.

هرجا که اکثریت الکترو موتورها را می‌توان استفاده کرد انواع سروموتورها را نیز می‌توان استفاده کرد. قیمت این نوع دستگاه‌ها بالا است بنابراین در جایی که نیاز به دقت زیاد، سرعت بالا و پاسخ سریع داریم بیش‌تر از این نوع الکتروموتور استفاده می‌شود. این نوع الکتروموتور در دستگاه‌های CNC فلز، دستگاه CNC چوب، طلا، دستگاه‌های پزشکی، تزریق پلاستیک، دستگاه‌های چاپ، دستگاه‌های تولید قطعه‌های الکترونیکی و نساجی به وفور استفاده شده‌است.

کدهای موردنیاز برای برنامه‌ریزی بُرد چیست؟



q1 §

```
#include <Servo.h>

Servo servo;

int degree;
void setup() {
  servo.attach(9,1000,2000);  // attaches the servo on pin 9 to the servo object
}

void loop() {
  for (degree = 0; degree <= 90; degree ++) {
    servo.write(degree+90);
    delay(50);
  }
  for (degree = 90; degree >= 0; degree --) {
    servo.write(degree+90);
    delay(50);
  }
}
```

q2 §

```
#include <Keypad.h>
#include <Servo.h>

const byte ROWS = 4; //four rows
const byte COLS = 4; //three columns
char keys[ROWS][COLS] = {
  {'7','8','9', '/'},
  {'4','5','6', '*'},
  {'1','2','3', '-'},
  {'o','0','=', '+'}
};
byte rowPins[ROWS] = {31, 33, 35, 37}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {23, 25, 27, 29}; //connect to the column pinouts of the keypad

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );
Servo servo;

String degree;

void setup() {
  Serial.begin(9600);
  servo.attach(9,1000,2000);
}

void loop() {
  char key = keypad.getKey();

  if (key){
    if (key >= '0' && key <= '9')
    {
      degree += key;
    }
    else if (key == '=')
    {
      servo.write(degree.toInt()/2);
      degree = "";
    }

    Serial.println(degree);
  }
}
```

q3 §

```
#include <Servo.h>

Servo servo;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  servo.attach(9, 1000, 2000);

}

int ServoMap(int x){
  return map(x, -180, 180, 1000, 2000);
}

String incoming = "";

void loop() {
  // put your main code here, to run repeatedly:
  if(Serial.available() > 0){
    incoming = Serial.readString();
    Serial.println(-1 * incoming.toInt());
    servo.writeMicroseconds(ServoMap(-1 * incoming.toInt()));
  }

}
```

q4 §

```
#include <Servo.h>

Servo servo;

void setup() {
  servo.attach(9);
  Serial.begin(9600);
}

void loop() {
  int degree;
  degree = analogRead(A0);
  degree = map(degree, 0, 1023, 0, 180);
  servo.write(degree);
  Serial.println(degree);
  delay(1000);
}
```

توضیح در مورد ورودی آنالوگ و تحلیل آن در آردوینو و تابع مورد استفاده در این آزمایش:

- **AnalogRead()**

analogRead() در آردوینو داده را از پین آنالوگِ مشخص‌شده می‌خواند. برد آردوینو حاوی یک مبدل آنالوگ است. به این معنا که این مبدل، ولتاژهای ورودی بین ۰ تا ۵ ولت را به مقداری صحیح (integer) بین ۰ تا ۱۰۲۳، تبدیل خواهد کرد. محدودهٔ ورودی و رزولوشن را می‌توانیم با استفاده از analogReference() تغییر دهیم. اگر پین ورودی آنالوگ به چیزی متصل نباشد، مقدار بازگردانده‌شده توسط analogRead()، طبق چند فاکتور درحال تغییر و نوسان خواهد بود.

تعریف مختصر توابع موردنیاز از کتابخانهٔ servo.h مانند:

- **Attach()**

شمارهٔ پین را از ورودی گرفته و آن را برای کار با سروو موتور آماده می‌کند.

- **Write()**

یک زاویه می‌گیرد و اهرم موتور را تا رسیدن به آن زاویه می‌چرخاند.

- **Read()**

زاویهٔ فعلی موتور یا همان آخرین عدد Write شده در موتور را برمی‌گرداند.

- **WriteMicroseconds()**

اهرم موتور را برحسب عدد ورودی گرفته شده، تنظیم می‌کند (دقت بالاتری از Write دارد).

- **ReadMicroseconds()**

زاویهٔ فعلی اهرم موتور را برمی‌گرداند (دقت بالاتری از Read دارد).