Điều khiển Step Motor

Bởi:

DKS Group

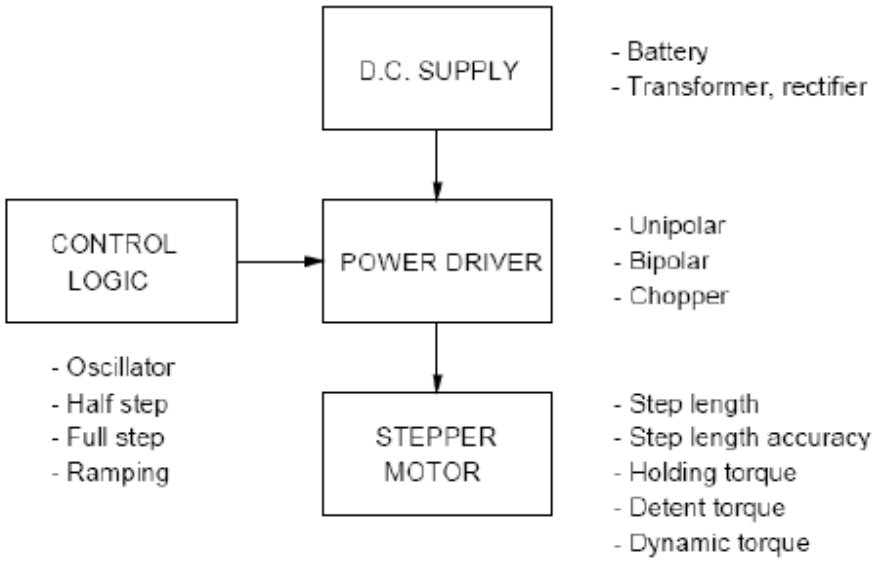
Lý thuyết

Giới thiệu về động cơ bước

Động cơ bước thực chất là một động cơ đồng bộ dùng để biến đổi các tín hiệu điều khiển dưới dạng các xung điện rời rạc kế tiếp nhau thành các chuyển động góc quay hoặc các chuyển động của roto và có khả năng cố định roto vào những vị trí cần thiết. Động cơ bước làm việc được là nhờ có bộ chuyển mạch điện tử đưa các tín hiệu điều khiển vào stato theo một thứ tự và một tần số nhất định. Tổng số góc quay của roto tương ứng với số lần chuyển mạch, cũng như chiều quay và tốc độ quay của roto, phụ thuộc vào thứ tự chuyển đổi và tần số chuyển đổi. Khi một xung điện áp đặt vào cuộn dây stato (phần ứng) của động cơ bước thì roto (phần cảm) của động cơ sẽ quay đi một góc nhất định, góc ấy là một bước quay của động cơ. Khi các xung điện áp đặt vào các cuộn dây phần ứng thay đổi liên tục thì roto sẽ quay liên tục. (Nhưng thực chất chuyển động đó vẫn là theo các bước rời rạc).

Hệ thống điều khiển động cơ bước

Một hệ thống có sử dụng động cơ bước có thể được khái quát theo sơ đồ sau.



D.C.SUPPLY: Có nhiệm vụ cung cấp nguồn một chiều cho hệ thống. Nguồn một chiều này có thể lấy từ pin nếu động cơ có công suất nhỏ. Với các động cơ có công suất lớn có thể dùng nguồn điện được chỉnh lưu từ nguồn xoay chiều.

CONTROL LOGIC: Đây là khối điều khiển logic. Có nhiệm vụ tạo ra tín hiệu điều khiển động cơ. Khối logic này có thể là một nguồn xung, hoặc có thể là một hệ thống mạch điện tử. Nó tạo ra các xung điều khiển. Động cơ bước có thể điều khiển theo cả bước hoặc theo nửa bước.

POWER DRIVER: Có nhiệm vụ cấp nguồn điện đã được điều chỉnh để đưa vào động cơ. Nó lấy điện từ nguồn cung cấp và xung điều khiển từ khối điều khiển để tạo ra dòng điện cấp cho động cơ hoạt động.

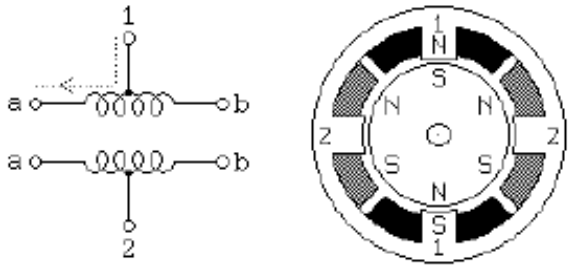
STEPPER MOTOR: Động cơ bước. Các thông số của động cơ gồm có: Bước góc, sai số bước góc, mômen kéo, mômen hãm, mômen làm việc.Đối với hệ điều khiển động cơ bước, ta thấy đó là một hệ thống khá đơn giản vì không hề có phần tử phản hồi. Điều này có được vì động cơ bước trong quá trình hoạt động không gây ra sai số tích lũy, sai số của động cơ do sai số trong khi chế tạo. Việc sử dụng động cơ bước tuy đem lai độ chính xác chưa cao nhưng ngày càng được sử dụng phổ biến. Vì công suất và độ chính xác của bước góc đang ngày càng được cải thiện.

Bước góc của động cơ bước được chế tạo theo bảng tiêu chuẩn sau:

Step angle	Steps per revolution
0.9°	400
1.8°	200
3.6°	100
3.75°	96
7.5°	48
15.0°	24

Nguyên tăc điều khiển động cơ bước đơn cực

Động cơ bước đơn cực, (có thể là động cơ vĩnh cửu hoặc động cơ hỗn hợp) có 5,6 hoặc 8 dây ra thường được quấn như sơ đồ dưới. Khi dùng, các đầu nối trung tâm thường được nối vào cực dương nguồn cấp, và hai đầu còn lại của mỗi mấu lần lượt nối đất để đảo chiều từ trường tạo bởi quận đó.



Động cơ đơn cực

Tín hiệu điều khiển. Điều khiển đủ bước (full step) :

Winding 1a 10001000100010001000100010001

Winding 1b 00100010001000100010001000100

Winding 2a 01000100010001000100010001000

Winding 2b 00010001000100010001000100010

time --->

Winding 1a 11001100110011001100110011001

Winding 1b 00110011001100110011001100110

Winding 2a 01100110011001100110011001100

Winding 2b 10011001100110011001100110011

time --->

Điều khiển nửa bước (half step)

Winding 1a 11000001110000011100000111

Winding 1b 00011100000111000001110000

Winding 2a 01110000011100000111000001

Winding 2b 00000111000001110000011100

time --->

Mạch điều khiển động cơ bước

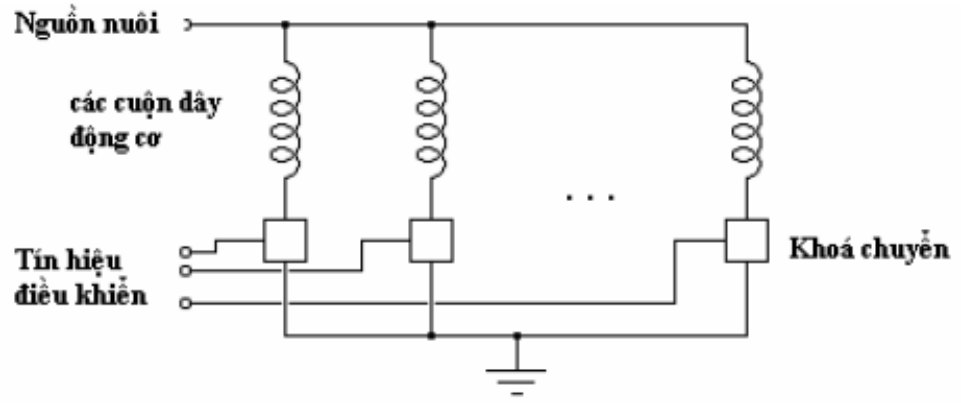
Mạch điều khiển động cơ bước bao gồm một số chức năng sau đây:

Tạo các xung với những tần số khác nhau.

Chuyển đổi các phần cho phù hợp với thứ tự kích từ.

Làm giảm các dao động cơ học.

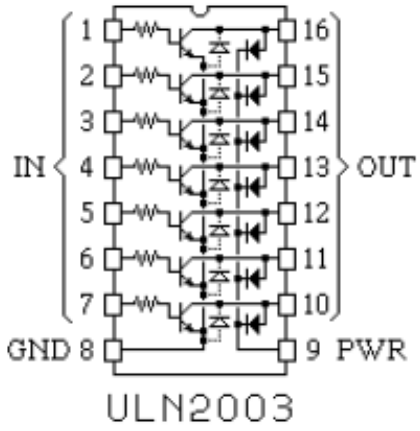
Đầu vào của mạch điều khiển là các xung. Thành phần của mạch là các bán dẫn, vi mạch. Kích thích các phần của động cơ bước theo thứ tự 1-2-3-4 do các transistor công suất T1 đến T4 thực hiện.Với việc thay đổi vị trí bộ chuyển mạch, động cơ có thể quay theo chiều kim đồng hồ hoặc ngược lại.



Điện áp được cấp qua các khoá chuyển để nuôi các cuộn dây, tạo ra từ trường làm quay rotor. Các khoá ở đây không cụ thể, có thể là bất cứ thiết bị đóng cắt nào điều khiển được như rơle, transitor công suất... Tín hiệu điều khiển có thể được đưa ra từ bộ điều khiển như vi mạch chuyên dụng, máy tính. Với động cơ nhỏ có dòng cỡ 500 mili Ampe, có thể dùng IC loại dãy darlington collector hở như :

ULN2003, ULN2803 (Allegro Microsystem)

DS2003 (National Semiconductor), MC1413 (Motorola)



IC họ ULN200x có đầu vào phù hợp TTL, các đầu emitor được nối với chân 8. Mỗi transitor darlington được bảo vệ bởi hai diode. Một mắc giữa emitor tới collector chặn điện áp ngược lớn đặt lên transitor. Diode thứ hai nối collector với chân 9. Nếu chân 9 nối với cực dương của cuộn dây, tạo thành mạch bảo vệ cho transitor.

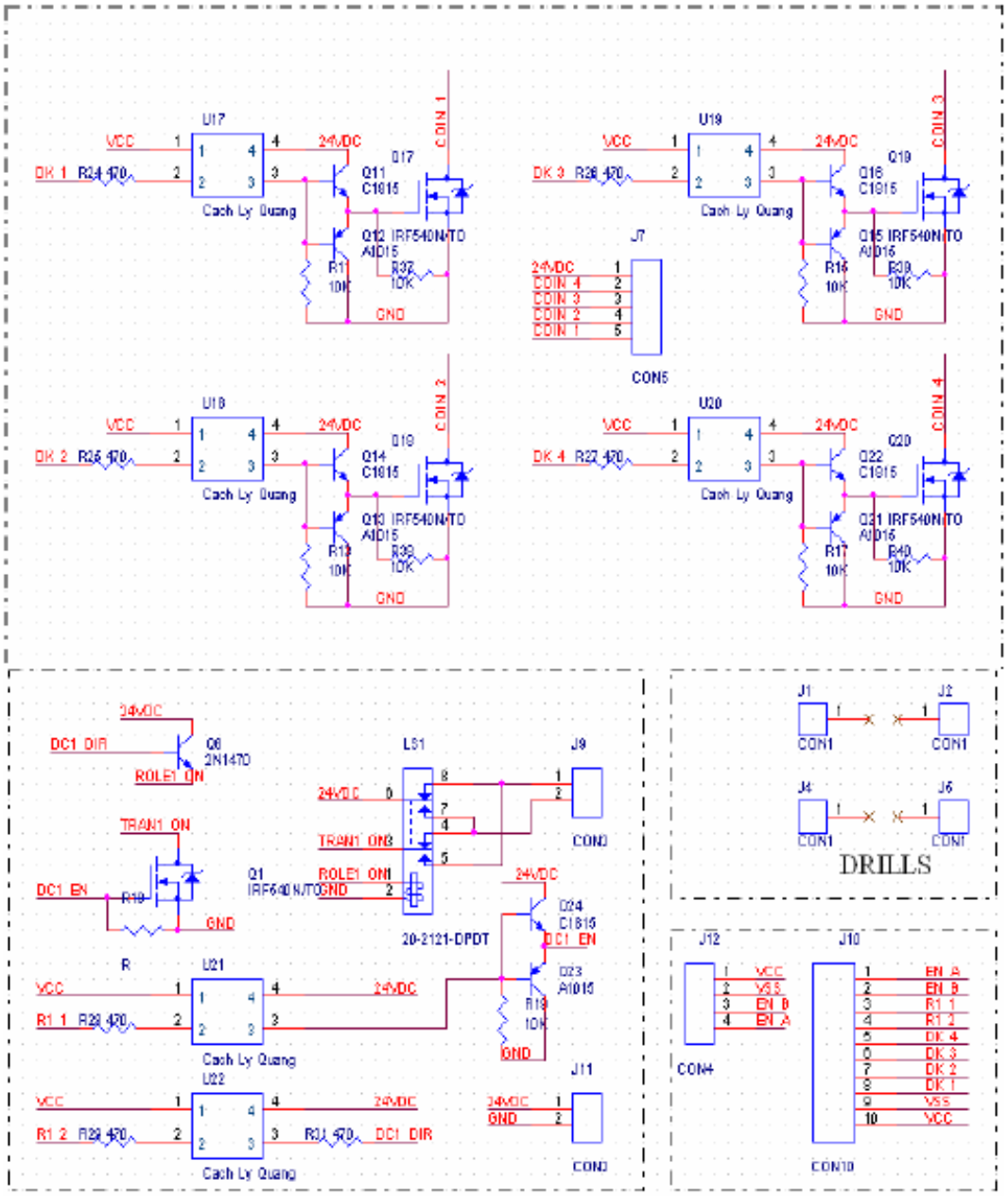
Với các động cơ lớn có dòng > 0.5A các IC họ ULN không đáp ứng được ta có thể dùng các Tranzitor trường(IRF).Một số loại IRF thông dụng:

IRF540 tranzitor ngược có thể chịu dòng đến 20A

IRF640 tranzitor ngược có thể chịu dòng đến 18A

IRF250 tranzitor ngược có thể chịu dòng đến 30A .

Sơ đồ mạch được thiết kế như sau:



Code

```
#include <mega16.h>

#include <delay.h>

// Khai bao bien

unsigned char stepA[] = {0xFF,0xFE,0xFD,0xFB,0xF7}, stepB[]
= {0xFF,0xEF,0xDF,0xBF,0x7F},
```

```
stepC[] = {0xFF,0xEF,0xDF,0xBF,0x7F};

unsigned char indexA, indexB, indexC;

unsigned char n_data;

unsigned char n_step=10;

unsigned int n_step3=5000,n_i;

//-------------------

// Declare your global variables here void main(void)

{

// Declare your local variables here

// Input/Output Ports initialization

// Port A initialization

// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In

// State7=T State6=T State5=T State4=T State3=T State2=T
State1=T State0=T PORTA=0xFF;

DDRA=0xFF;

// Port B initialization

// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In

// State7=T State6=T State5=T State4=T State3=T State2=T
State1=T State0=T PORTB=0xFF;

DDRB=0xFF;

// Port C initialization
```

```
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In

// State7=T State6=T State5=T State4=T State3=T State2=T
State1=T State0=T PORTC=0xFF;

DDRC=0xFF;

// Port D initialization

// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In
Func1=In Func0=In

// State7=T State6=T State5=T State4=T State3=T State2=T
State1=T State0=T

PORTD=0xFF;

DDRD=0xFF;

// Timer/Counter 0 initialization

// Clock source: System Clock

// Clock value: Timer 0 Stopped

// Mode: Normal top=FFh

// OC0 output: Disconnected

TCCR0=0x00;

TCNT0=0x00;

OCR0=0x00;

// Timer/Counter 1 initialization

// Clock source: System Clock

// Clock value: Timer 1 Stopped

// Mode: Normal top=FFFFh
```

```
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
```

```
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;
// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;
// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;
// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80; SFIOR=0x00;
while (1)
{
// Place your code here
if(indexA ++>3) indexA = 1;
if(indexB ++>3) indexB = 1;
```

```
if(indexC ++>3) indexC = 1;

PORTA = stepA[indexA] & stepB[indexB];

PORTC = stepC[indexC];

//------- delay_ms(500);

}}
```