

Lecture 9 objectives

- Stepper motor basic theory
- Stepper motors, full step versus half step algorithm,
- Stepper interface electronics (2N2222, L293, IRF540)

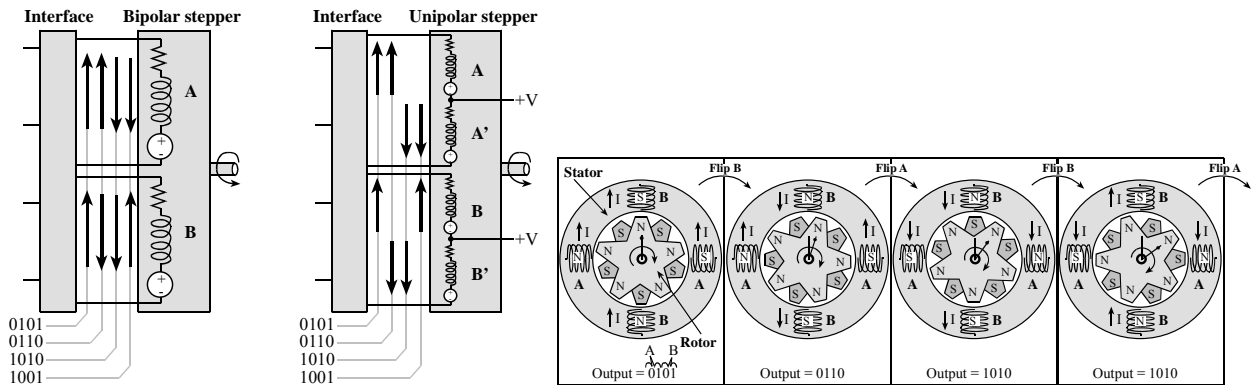
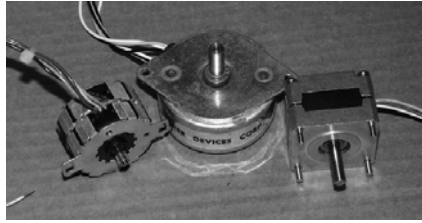
8.6. Stepper Motors

Advantages:

- 1
- 2
- 3

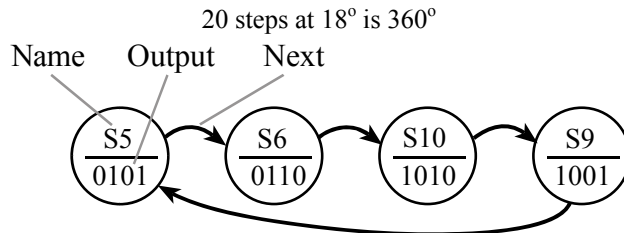
Disadvantages:

- 1
- 2
- 3



Initial stable state

To rotate this stepper by 18°, the interface flips the direction of one of the currents.



8.6.1. Stepper Motor Example

With 200 steps/revolution each new output will cause the motor to rotate 1.8°.

If the time in between outputs is fixed at Δt seconds,
then the shaft rotation speed will be $0.005/\Delta t$ in rps.

Interface design

unipolar (5-wire or 6-wire) versus bipolar (4-wire)

full step 0101, 0110, 1010, 1001, ...

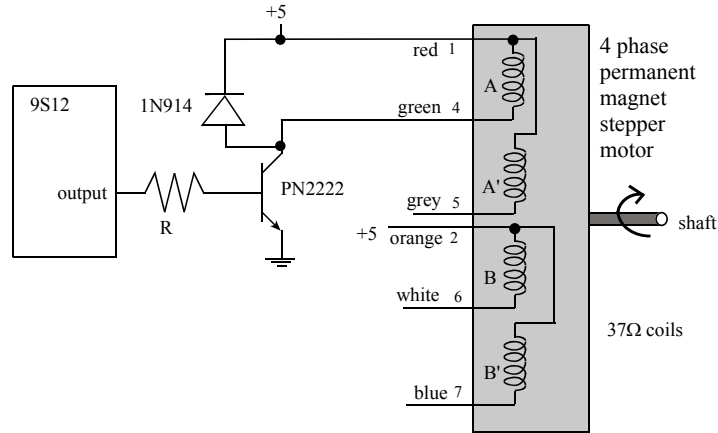
half-step 0101, 0100, 0110, 0010, 1010, 1000, 1001, 0001, ...

coil voltage and current

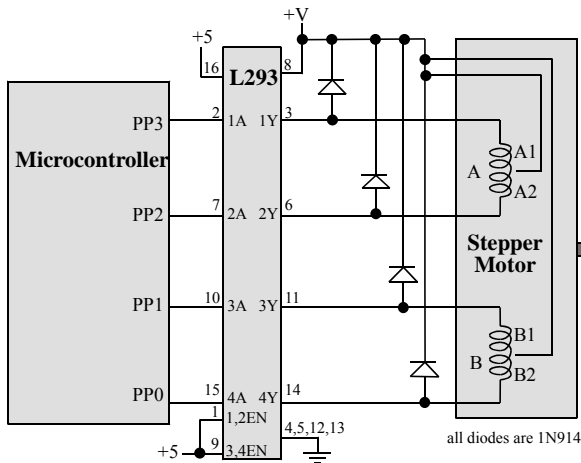
voltage => choose power supply

current => choose driver such that driver $I_{OL} > I_{coil}$

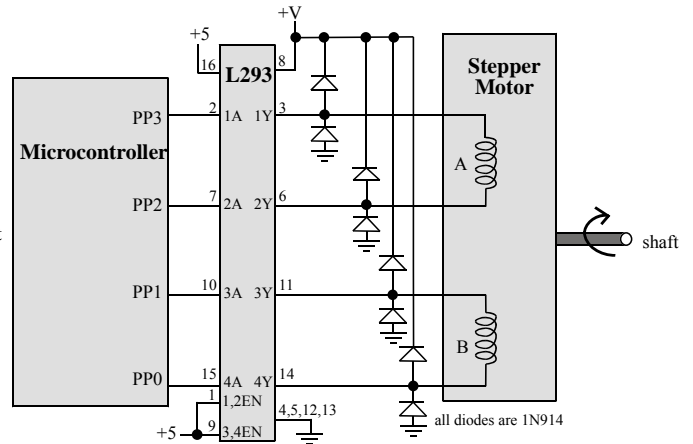
7406 unipolar stepper motors up to 40 mA
75492 unipolar stepper motors up to 250 mA



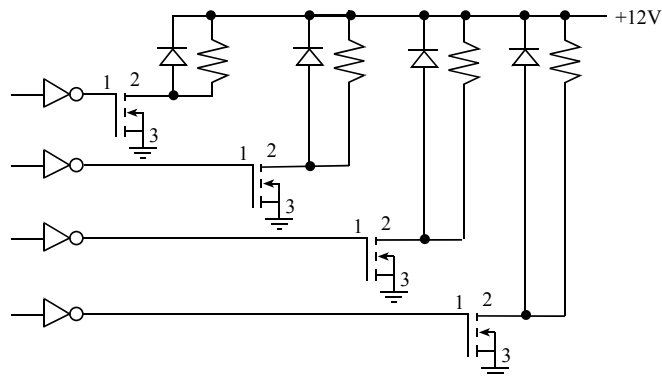
2N2222 unipolar stepper motors up to 500 mA



L293 unipolar stepper motors (1A per output)



L293 bipolar stepper motors (1A per output)

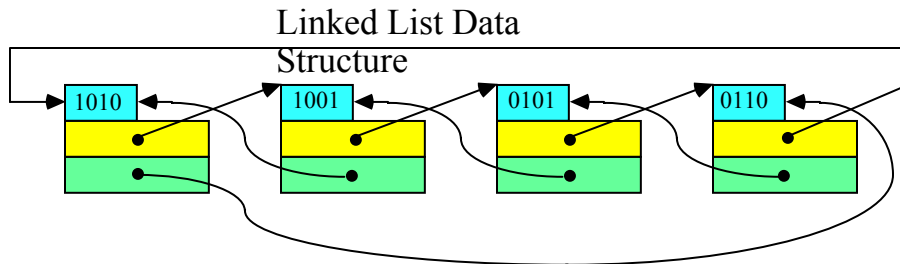


IRF540 unipolar stepper motors (28A per output)

We define the active state of the coil when current is flowing. The basic operation is summarized in the following table

output	A	A'	B	B'
10	activate	deactivate	activate	deactivate
9	activate	deactivate	deactivate	activate
5	deactivate	activate	deactivate	activate
6	deactivate	activate	activate	deactivate

Table 8.13. Stepper motor sequence.



A double circular linked list used to control the stepper motor.

```

const struct State{
    unsigned char Out; // Output for this state
    const struct State *Next[2]; // CW or CCW
};
typedef const struct State StateType;
typedef StateType * StatePtr;
unsigned short POS;
/* between 0 and 199 representing shaft angle */
#define clockwise 0 // Next index*/
#define counterclockwise 1 // Next index*/
full step 0101, 0110, 1010, 1001,...
half step 0101,0100,0110,0010,1010,1000,1001,0001,...
StateType fsm[4]={
    { 5,{&fsm[1],&fsm[3]}},
    { 6,{&fsm[2],&fsm[0]}},
    {10,{&fsm[3],&fsm[1]}},
    { 9,{&fsm[0],&fsm[2]}}
};
StatePtr Pt; /* Current State */
void CW(void){
    Pt = Pt->Next[clockwise]; // circulates
    PTP = Pt->Out; // step motor
    POS++;
    if(POS==200){
        POS=0; // shaft angle
    }
}
void CCW(void){
    Pt = Pt->Next[counterclockwise]; // circulates
    PTP = Pt->Out; // step motor
    if(POS==0){
        POS = 199;
    }
    else{
        POS--;
    }
    // shaft angle
}
half-step 0101,0100,0110,0010,1010,1000,1001,0001,...
StateType fsm[8]={
    { 5,{&fsm[1],&fsm[7]}},

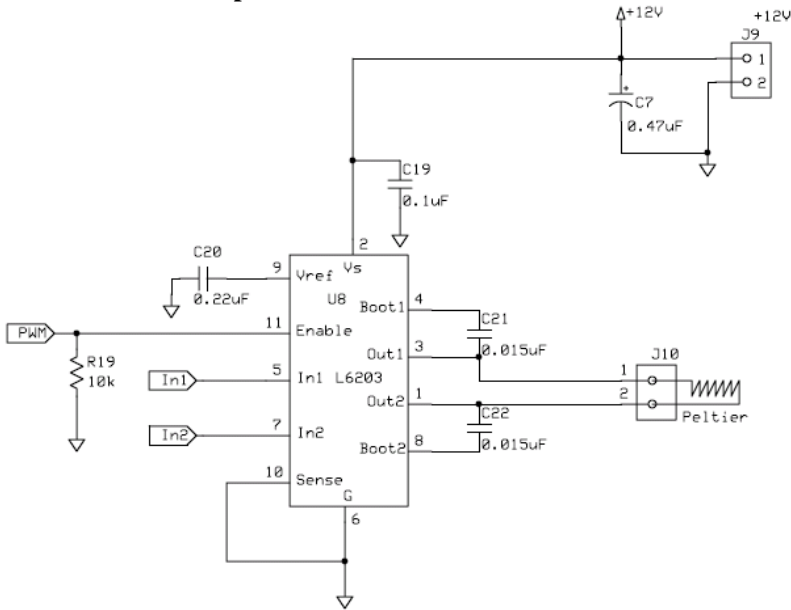
```

```

    { 4, {&fsm[2], &fsm[0]}},
    { 6, {&fsm[3], &fsm[1]}},
    { 2, {&fsm[4], &fsm[2]}},
    {10, {&fsm[5], &fsm[3]}},
    { 8, {&fsm[6], &fsm[4]}},
    { 9, {&fsm[7], &fsm[5]}},
    { 1, {&fsm[0], &fsm[6]}}
};
StatePtr Pt; /* Current State */
void CW(void){
    Pt = Pt->Next[clockwise]; // circulates
    PTP = Pt->Out; // step motor
    POS++;
    if(POS==400){
        POS=0; // shaft angle
    }
}
void CCW(void){
    Pt = Pt->Next[counter-clockwise]; // circulates
    PTP = Pt->Out; // step motor
    if(POS==0){
        POS = 399;
    }
    else{
        POS--;
    }
    // shaft angle
}
}

```

L6203 can handle 5A peak or 4A RMS current



Peltier Temperature Controller

- In1 = 1, In2 = 0, PWM duty cycle sets amplitude of heating
- In1 = 0, In2 = 1, PWM duty cycle sets amplitude of cooling
- In1 = 0, In2 = 0, off

For a Brushed DC motor Controller

- In1 = 1, In2 = 0, PWM duty cycle sets amplitude of spin CW
- In1 = 0, In2 = 1, PWM duty cycle sets amplitude of spin CCW
- In1 = 0, In2 = 0, off