

## ECE/CS 5780/6780: Embedded System Design

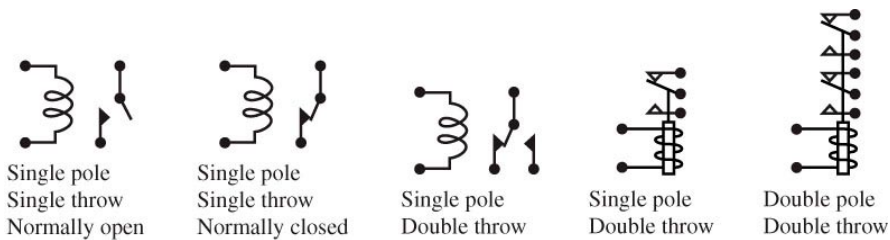
Scott R. Little

Lecture 17: Relays and Motors

## Introduction to Relays

- A relay is a device that responds to a small current or voltage change by activating a switches or other devices.
- Used to remotely switch signals or power.
- Input control usually electrically isolated from output.
- Input signal determines whether switch is open or closed.

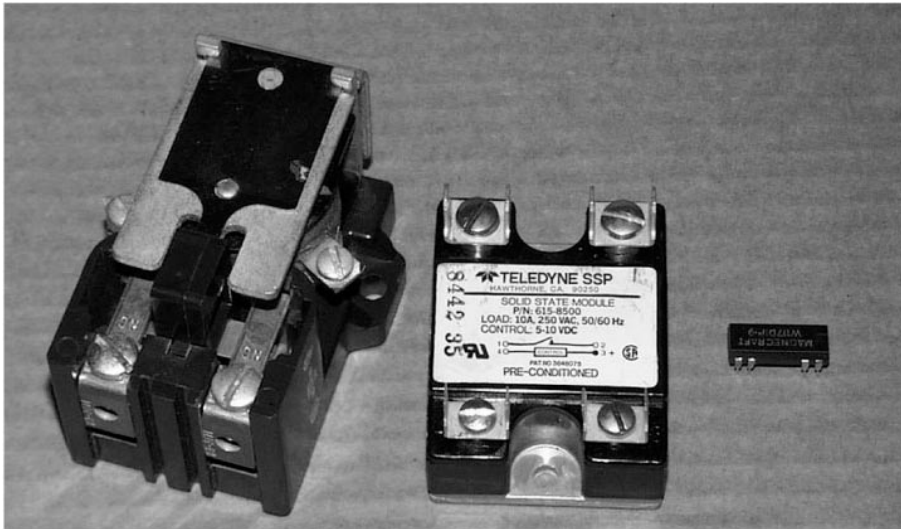
## Various Relay Configurations



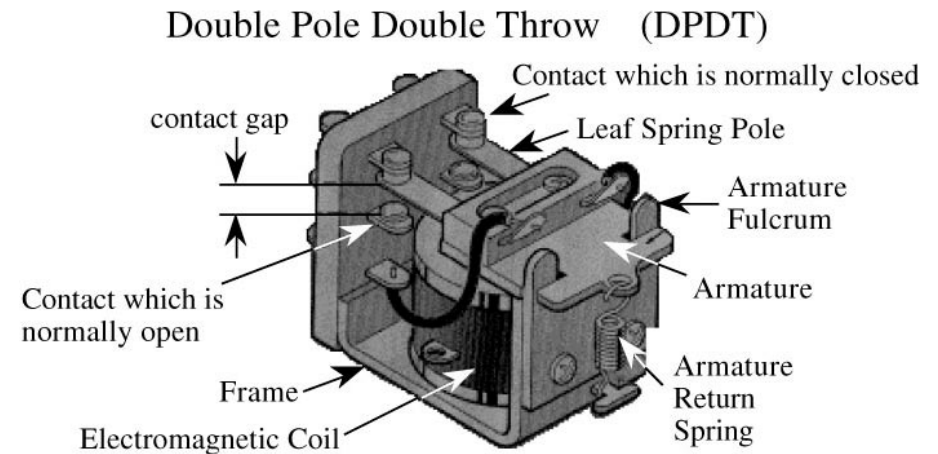
## Types of Relays

- Classic general-purpose relays have EM coils and can switch power.
- Solid-state relays (SSR) have input-triggered semiconductor switches.
- Reed relay has an EM coil and can switch low level DC signals.
- The bilateral switch uses CMOS, FET, or biFET transistors (technically not a relay but behaves similarly).

## Types of Relays



## Drawing of an EM Relay



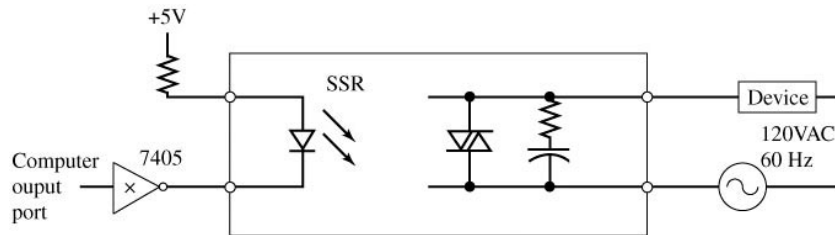
## Electromagnetic Relay Basics

- Input circuit is an EM coil with an Iron Core.
- Output switch includes two sets of silver or silver-alloy contacts (*poles*).
- One set is fixed to the relay *frame*, and other is located at end of leaf spring poles connected to the *armature*.
- Contacts held in “normally closed” position by the armature return spring.
- When input circuit energizes EM coil, a “pull-in” force is applied to the armature and “normally closed” contacts break while “normally open” contacts are made.

## Solid State Relays

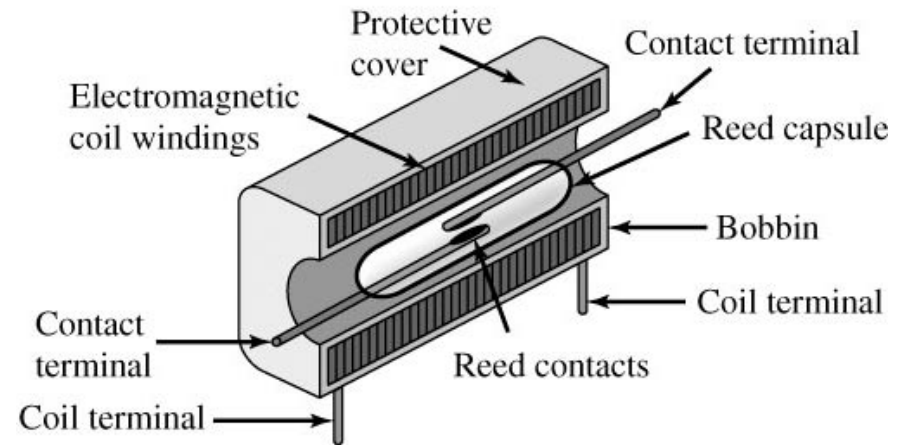
- Developed to solve limited life expectancy and contact bounce problems since they have no moving parts.
- Also, faster, insensitive to vibrations, reduced EMI, quieter, and no contact arcing.
- Optocoupler provides isolation between the input circuit (pseudocoil) and the triac (pseudocontact).
- Signal from phototransistor triggers the output triac so that it switches the load current.
- Zero-voltage detector triggers triac only when AC voltage is zero, reducing surge currents when triac is switched.
- Once triggered, triac conducts until next zero crossing.

## Solid State Relays

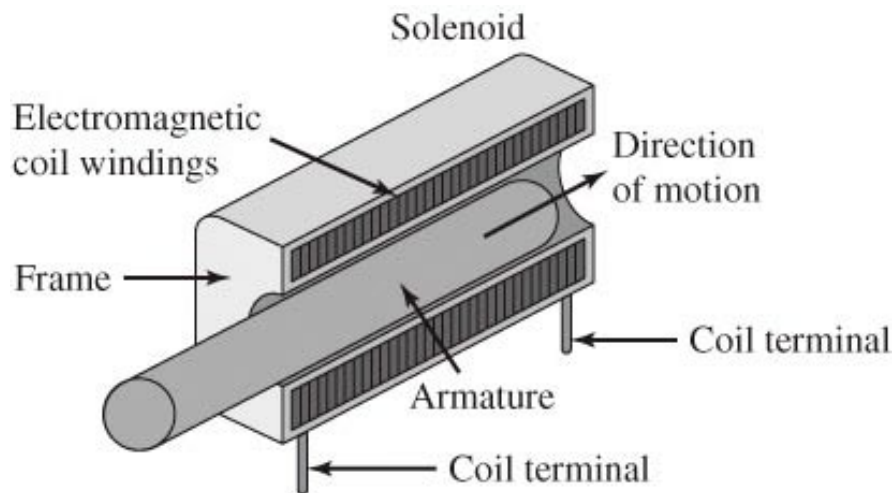


## Reed Relays

### Single Pole Single Throw (SPST) Reed Relay



## Solenoids



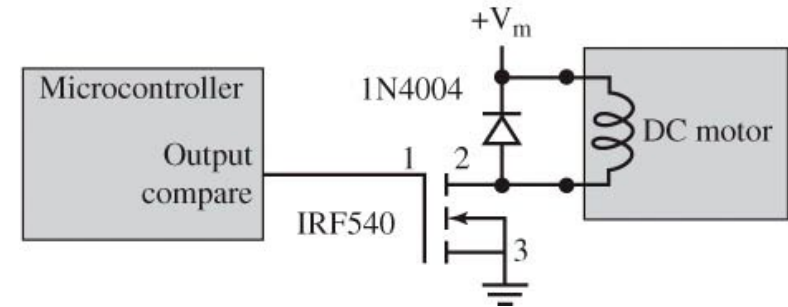
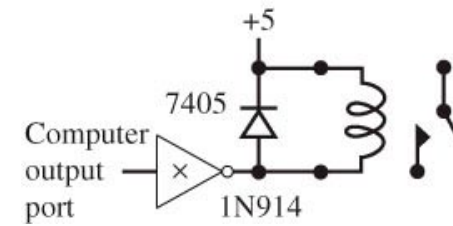
## Pulse-Width Modulated DC Motors

- DC motor also has frame that remains motionless and an armature that moves in this case in a circular manner.
- When current flows through EM coil, magnetic force created that causes rotation of the shaft.
- Brushes positioned between frame and armature used to alternate the current direction through the coil so that a DC current generates a continuous rotation of the shaft.
- When current removed, shaft is free to rotate.
- Pulse-width modulated DC motor activated with fixed magnitude current but duty cycle varied to control speed.

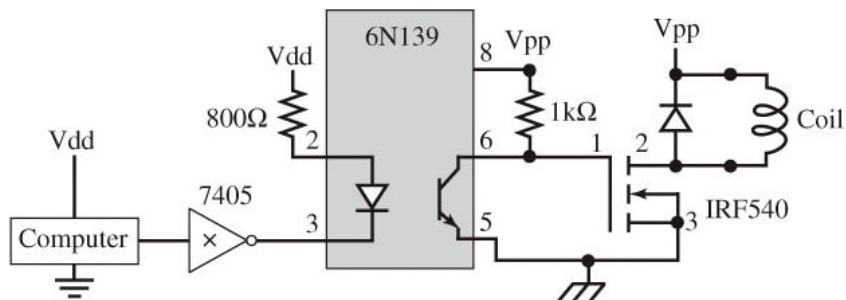
## Interfacing EM Relays, Solenoids, and DC Motors

- Interface circuit must provide sufficient current and voltage to activate the device.
- In off state, input current should be zero.
- Due to inductive nature of the coil, huge back electromotive force (EMF) when coil current is turned off.
- Due to high speed transistor switch, there is a large  $di/dt$  when the coil is deactivated (activation also but smaller).
- Voltages can range from 50 to 200V.
- To protect the driver electronics, a snubber diode is added to suppress the back EMF.

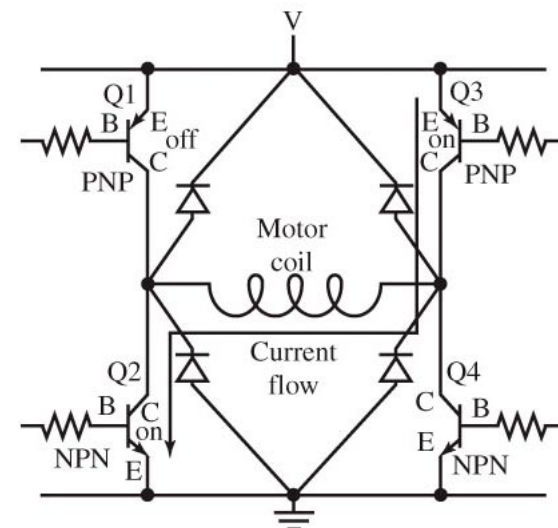
## Relay and Motor Interfaces



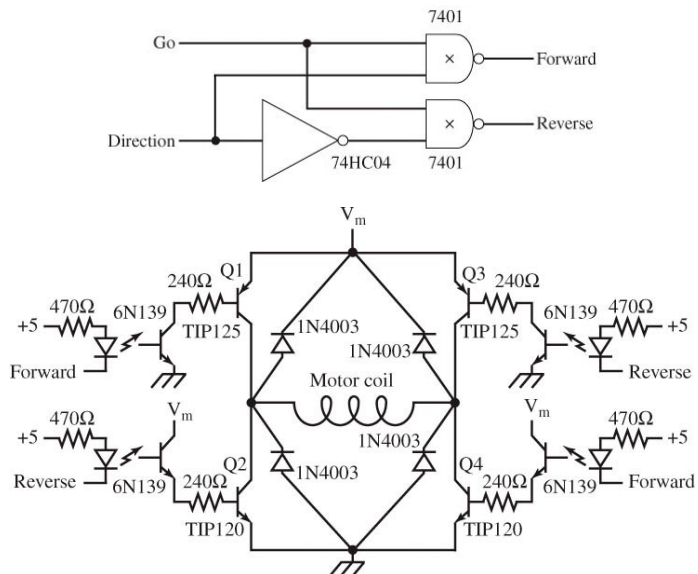
## Isolated Interfaces



## H-Bridge



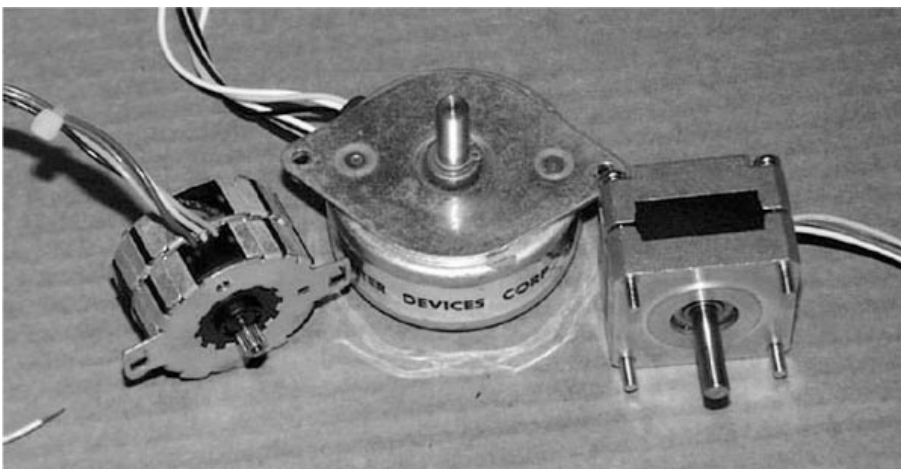
## Isolated H-Bridge with Direction Control



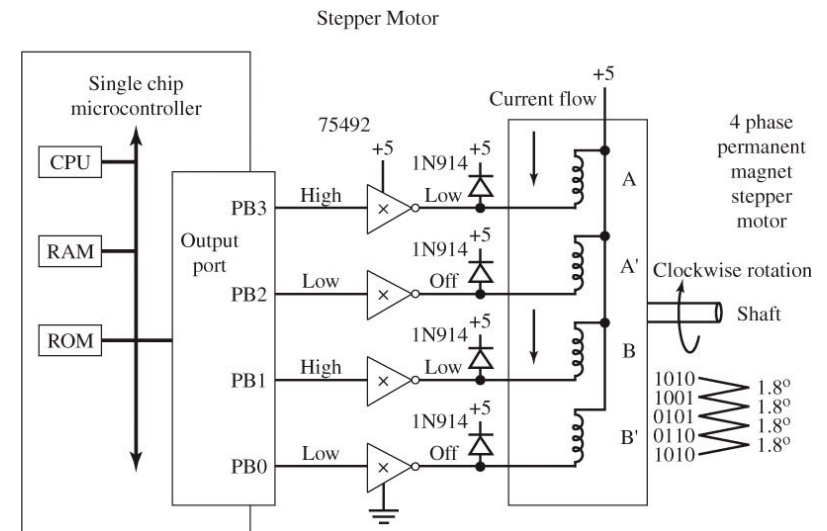
## Stepper Motors

- Very popular due to inherent digital interface.
- Easy to control both position and velocity in an open-loop fashion.
- Though more expensive than ordinary DC motors, system cost is reduced as they require no feedback sensors.
- Can also be used as shaft encoders to measure both position and speed.

## Stepper Motors



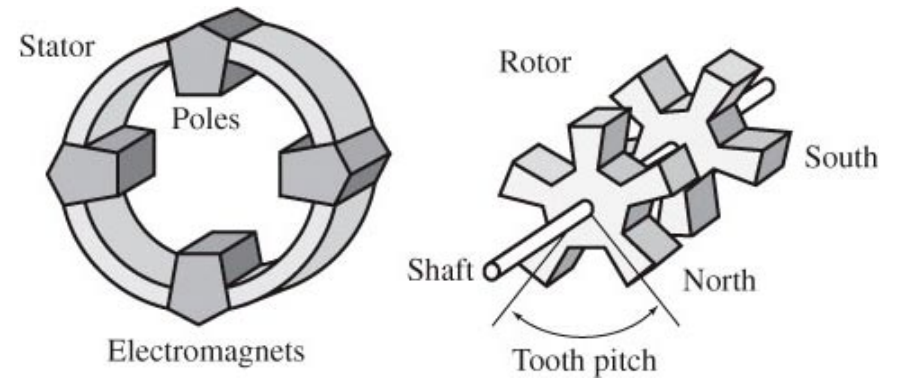
## Simple Stepper Motor Interface



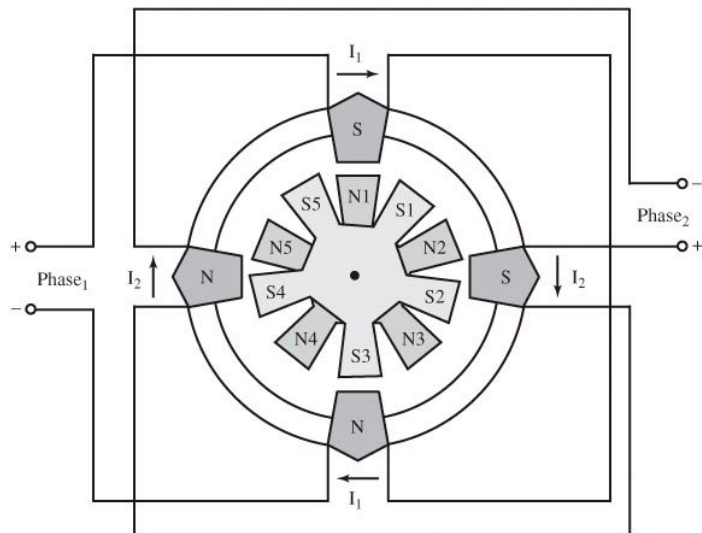
## Stepper Motor Sequence

PortB	A	A'	B	B'
10	Activate	deactivate	activate	deactivate
9	Activate	deactivate	deactivate	activate
5	Deactivate	activate	deactivate	activate
6	Deactivate	activate	activate	deactivate

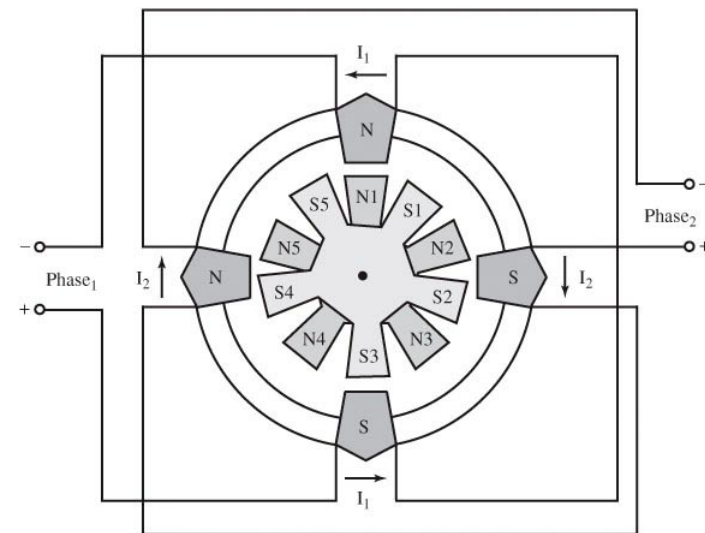
## Stepper Motor Basic Operation



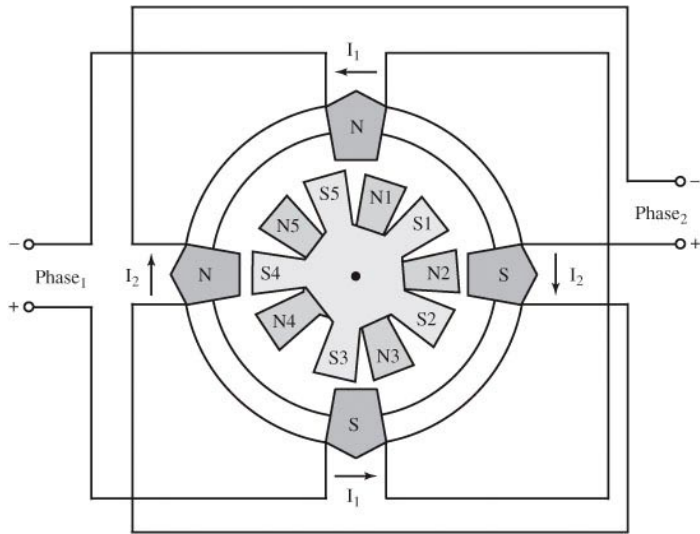
## Stepper Motor Basic Operation (cont)



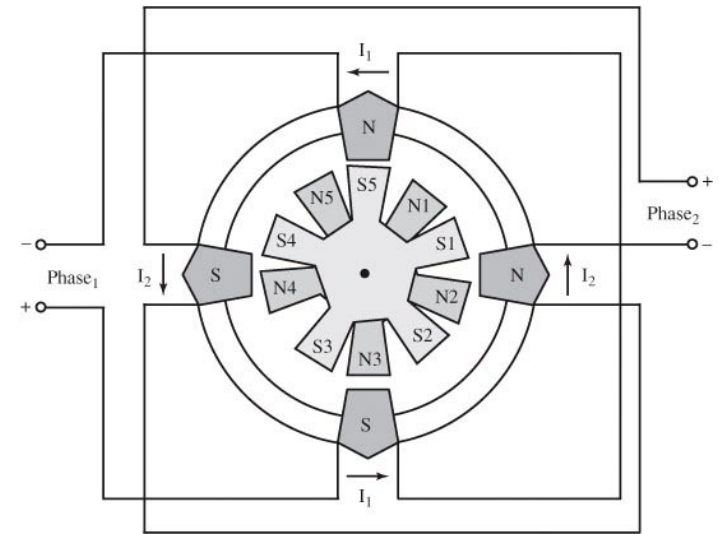
## Stepper Motor Basic Operation (cont)



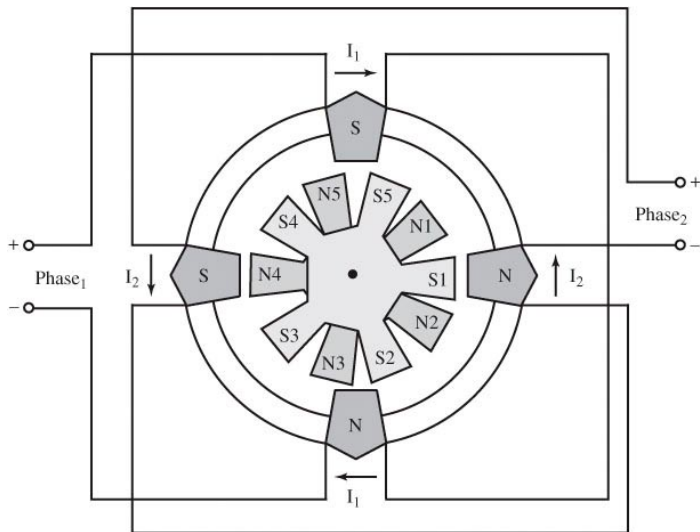
## Stepper Motor Basic Operation (cont)



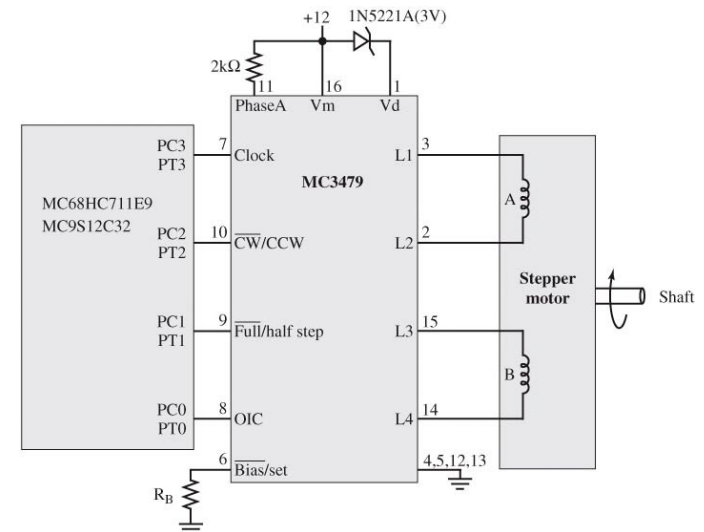
## Stepper Motor Basic Operation (cont)



## Stepper Motor Basic Operation (cont)



## Bipolar Stepper Motor Interface



## Slip

- A *slip* is when computer issues a sequence change, but the motor does not move.
- Occurs if load on shaft exceeds available torque of motor.
- Can also occur if computer changes output too fast.
- If initial shaft angle known and motor never slips, computer can control shaft angle and speed without position sensor.

## Stepper Motor Sequence



## Data Structures to Control Stepper Motor

```
const struct State{
    unsigned char Out;           // Output
    const struct State *Next[2]; // CW/CCW
};
typedef struct State StateType;
typedef StateType *StatePtr;
#define clockwise 0 // Next index
#define counterclockwise 1 // Next index
StateType fsm[4]={
    {10, {&fsm[1], &fsm[3]}},
    { 9, {&fsm[2], &fsm[0]}},
    { 5, {&fsm[3], &fsm[1]}},
    { 6, {&fsm[0], &fsm[2]}}};
unsigned char Pos; // between 0 and 199
StatePtr Pt; // Current State
```

## Ritual to Control Stepper Motor

```
void Init(void){
    Pos = 0;
    Pt = &fsm[0];
    DDRB = 0xFF;
}
```



## Helper Functions to Control Stepper Motor

```

void CW(void){
    Pt = Pt->Next[clockwise]; // circular
    PORTB = Pt->Out;          // step motor
    if(Pos==199){             // shaft angle
        Pos = 0;              // reset
    }else{
        Pos++;                // CW
    }
}
void CCW(void){
    Pt = Pt->Next[counterclockwise];
    PORTB = Pt->Out;          // step motor
    if(Pos==0){               // shaft angle
        Pos = 199;           // reset
    }else{
        Pos--;                // CCW
    }
}

```

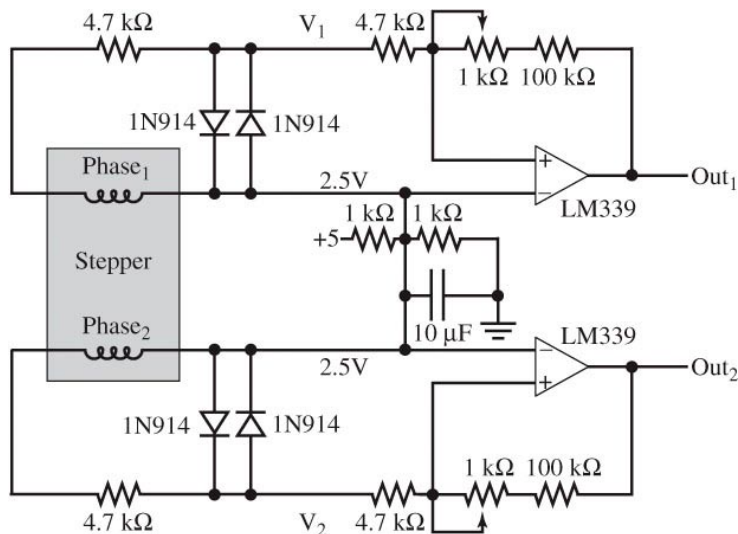
## High-Level Control of Stepper Motor

```

void Seek(unsigned char desired){
    short CWsteps;
    if((CWsteps=desired-Pos)<0){
        CWsteps+=200;
    } // CW steps is 0 to 199
    if(CWsteps>100){
        while(desired!=Pos){
            CCW();
        }
    }
    else{
        while(desired!=Pos){
            CW();
        }
    }
}

```

## Stepper Motor as Shaft Position Sensor



## Timing of Stepper Motor as Shaft Position Sensor

