CS/ECE 6780/5780

Al Davis

Today's topics:

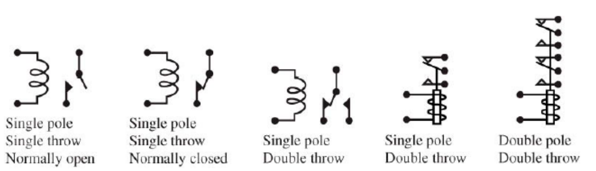
- Relays & Motors
  - prelude to 5780 Lab 9

Relays

- **Common embedded system problem**
  - digital control: relatively small I & V levels
  - controlled device requires significantly higher power
- **Solution**
  - amplify the control power
  - use the control signal to activate a switch
    - » switch turns on/off bigger power source
- **Electrically controlled switches**
  - transistor
    - » can be used as a switch but it's really an amplifier since it has gain
      - MOS – voltage controlled, BIPOLAR – current controlled
  - relay
    - » control induces magnetic field in coil
    - » magnetic field moves a mechanical switch
      - bounce problem?
        - – usually not a concern for outputs to non-digital gizmos like motors

Poles and Throws

- **Terminology used for switches**
  - relay is just an electrically controlled switch
    - » pole – controlled
    - » throw – contact point
    - » relay difference – magnetic movement of pole
      - difference in where the switch is when switch/magnet is off
        - – off state usually controlled by a spring

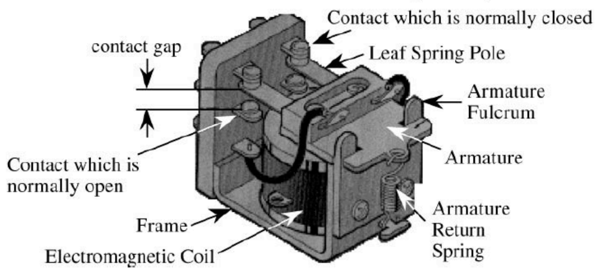


Relay Types

- **Basic issue is size**
  - control power
    - » reed relays – smallish power
      - common in ES designs
    - » general purpose – large'ish power
      - you have lots of them in your car

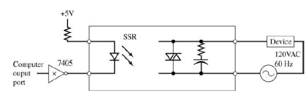
Mechanical DPDT Illustrated

Double Pole Double Throw (DPDT)



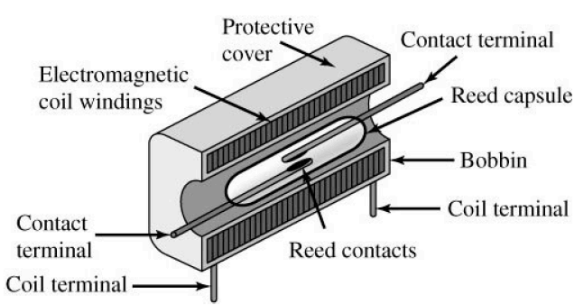
Solid State Relays

- **Improvement on mechanical relay problems**
  - contact bounce and arcing limit lifetime
  - sensitive to vibrations, EMI issues
  - slow movement of large mechanical pole
- **Optocoupler**
  - provides electrical isolation between input (pseudocoil) and output triac (pseudocontact)
    - » particularly important in driving large inductive loads
  - zero-voltage detector triggers triac
    - » reduces surge currents when triac is switched
  - once triggered
    - » triac conducts until next zero crossing

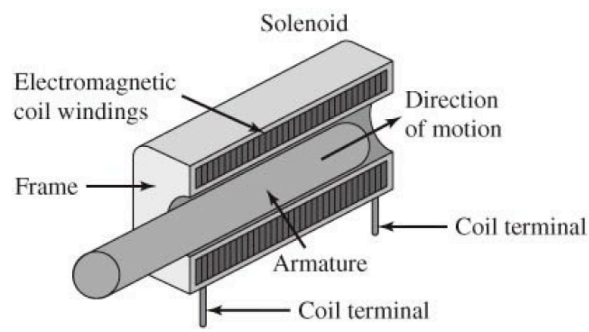


Reed Relays

Single Pole Single Throw (SPST) Reed Relay



Solenoids



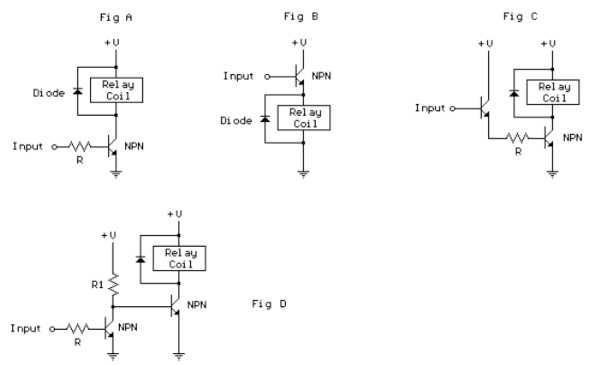
Interfacing to Inductive Loads

- **Interface circuit**
  - **must provide sufficient current and voltage to activate the device**
    - » **common error**
      - **"my microcontroller puts out 5v but at the device it's only 200 mV"**
      - **what's the problem?**

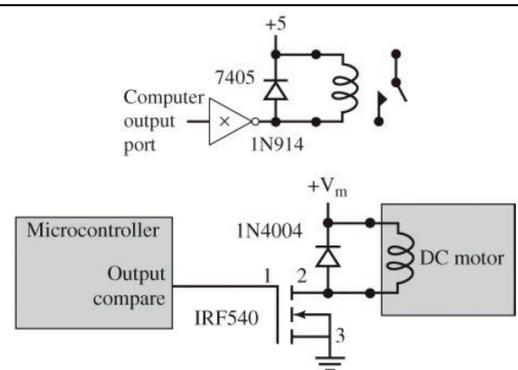
Interfacing to Inductive Loads

- **Interface circuit**
  - **must provide sufficient current and voltage to activate the device**
    - » **common error**
      - **"my microcontroller puts out 5v but at the device it's only 200 mV"**
      - **what's the problem?**
      - **Ohm's law**
        - – current, impedance and voltage are related
        - – microcontroller can't provide enough current so voltage is similarly low
  - **in off state current should be zero**
  - **BEWARE**
    - » **large L → huge back EMF when coil is turned off**
      - fast digital switch causes large di/dt
      - 50 – 200V back is common
    - » **it will destroy your controller**
      - isolation or buffering is required
        - – optoisolator
        - – or snub diode
        - – etc.

Relay Control Examples

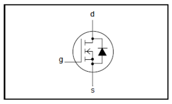


Relay & Motor Interfaces



IRF 540 Power Transistor

SYMBOL



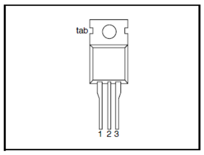
QUICK REFERENCE DATA

$V_{DSS} = 100$ V
$I_D = 23$ A
$R_{DS(ON)} \leq 77$ mΩ

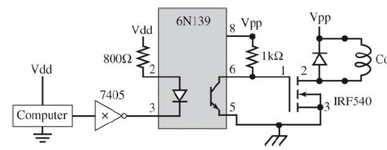
PINNING

PIN	DESCRIPTION
1	gate
2	drain[1]
3	source
tab	drain

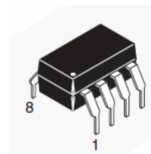
SOT78 (TO220AB)

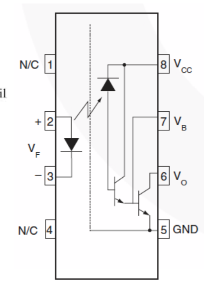


Isolated Interfaces



.5 mA in ➔ 1 A out
2000% current transfer ratio 5V logic compatible (TTL, CMOS)

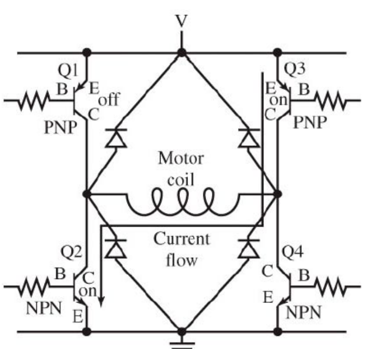




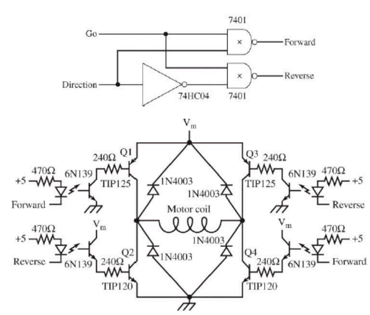
6N138 / 6N139

split Darlington photodetector

Typical H-Bridge Motor Control

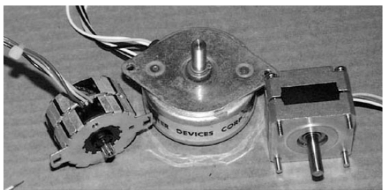


Isolated H-Bridge w/ Direction Control



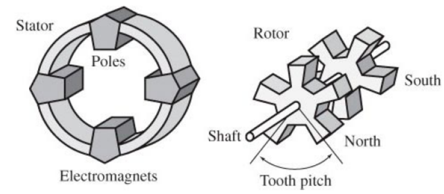
Stepper Motors

- **Popular due to inherent digital interface**
  - **easy to control both position and velocity in an open-loop fashion**
  - **more expensive than simple DC motor**
    - » **still not too bad since may not require feedback sensors**
  - **can be used as shaft encoders**
    - » **measure both position and speed**



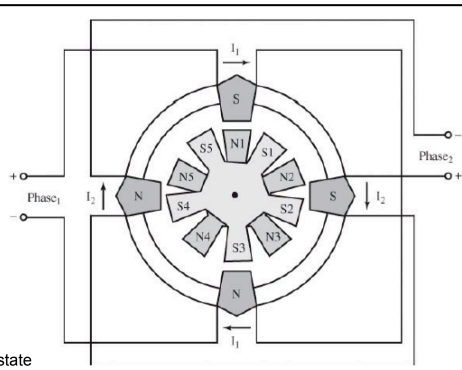
Stepper Motor Basics

- **Stator**
  - **stationary frame with electromagnet poles**
- **Rotor**
  - **teeth are permanent magnets alternating south and north pole teeth**



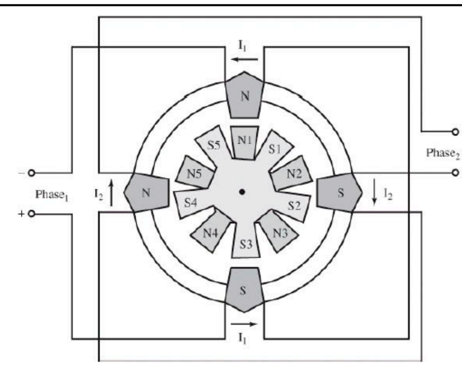
360 degrees/(4 poles * 5 teeth) = 18 degrees per step

2 Phase Operation



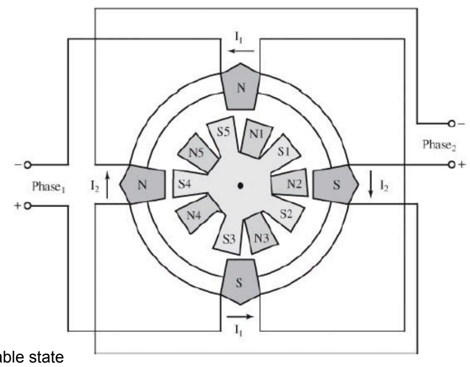
stable state

2 Phase Operation



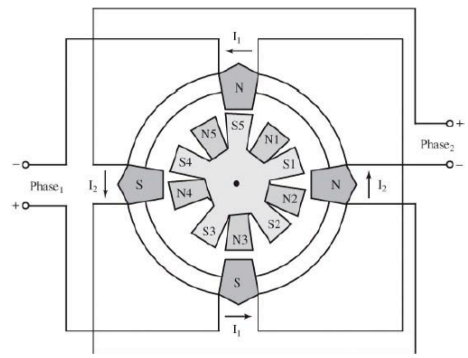
reverse phase 1 polarity – unstable state – closest stable state?

2 Phase Operation



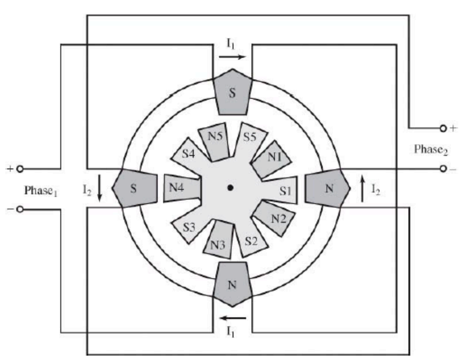
next stable state

2 Phase Operation

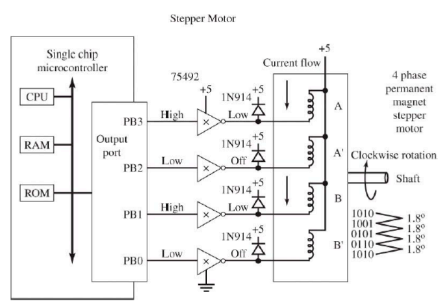


reverse polarity of phase 2 and movement continues

Continue by Reversing Phase 1



Simple Interface

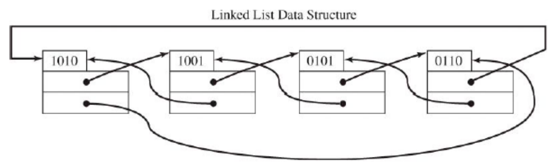


note this motor has 200 steps – hence 1.8 degrees
Port B output is (10, 9, 5, 6)*
reverse direction? (6, 5, 9, 10)*

Slip & Torque Issues

- **Slip**
  - **command issued but motor doesn't move**
  - **causes**
    - » **motor torque insufficient to drive mechanical load**
    - » **or if computer change is too fast**
      - • **➔ magnetic field is too weak**
- **IF no slip can be guaranteed**
  - **then computer knows the shaft position**
    - » **and doesn't need a sensor**

Stepper Motor Sequence



Control Data Structures (FSM)

```
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
```


Init Ritual

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

Helper Functions

```
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

```
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();
    }
  }
}
```


Concluding Remarks

- **Lots of types of electrical motors**
  - **stepper & DC are most common in inexpensive ES's**
- **Beware when driving inductive loads**
  - **back EMF has to be controlled**
    - » **snub diode is cheap**
    - » **optical isolation is even more secure**
- **5780 students**
  - **lab 9 will get you provide an introduction**
  - **stepper motor kits available for checkout**