Design Rules for NPN Transistors in Saturated Mode

1. Normally, $V_c > V_e$
2. Input current flows from base to emitter, output current flows from collector to emitter.
3. Have maximum values for: $I_b$, $I_c$, $V_{ce}$, and $I_c \cdot V_{ce}$.
4. Act like current amplifiers: $I_c = h_{fe} \cdot I_b$.
5. Activate if $V_b > V_c + V_{bc(SAT)}$, where $V_{bc(sat)}$ is >0.6V.

Design Rules for PNP Transistors in Saturated Mode

1. Normally, $V_c > V_e$
2. Input current flows from emitter to base, output current flows from emitter to collector.
3. Have maximum values for: $I_b$, $I_c$, $V_{ce}$, and $I_c \cdot V_{ce}$.
4. Act like current amplifiers: $I_c = h_{fe} \cdot I_b$.
5. Activate if $V_b < V_c - V_{bc(SAT)}$, where $V_{bc(sat)}$ is >0.6V.
Introduction to Relays

- A relay is a device that responds to a small current or voltage change by activating a switch 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.

Types of Relays

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

Various Types of Relays

- Single Pole Single Throw Normally Open
- Single Pole Single Throw Normally Closed
- Single Pole Double Throw
- Double Pole Double Throw

Drawing of an EM Relay

- Double Pole Double Throw (DPDT)
- Contact which is normally closed
- Leaf Spring Pole
- Armature
- Fulcrum
- Frame
- Electromagnetic Coil
- Armature Return Spring
- Contact which is normally open
- contact gap
Electromagnetic Relay Basics

- Input circuit is an EM coil with an Iron Core.
- Output switch includes two sets of silver or silver-alloy contacts (called 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.

Various Relay Configurations

<table>
<thead>
<tr>
<th>Form</th>
<th>Activation sequence</th>
<th>Deactivation sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Make 1, Break 1</td>
<td>Make 1</td>
</tr>
<tr>
<td>B</td>
<td>Break 1</td>
<td>Make 1</td>
</tr>
<tr>
<td>C</td>
<td>Break 1, Make 2</td>
<td>Break 2, Make 1</td>
</tr>
<tr>
<td>D</td>
<td>Make 1, Break 2</td>
<td>Make 2, Break 1</td>
</tr>
<tr>
<td>E</td>
<td>Break 1, Make 2, Break 3</td>
<td></td>
</tr>
</tbody>
</table>

Relay Parameters
(See Table 8.10)

Reed Relays

Single Pole Single Throw (SPST) Reed Relay
- Bobbin
- Coil Terminal
- Reed Contacts
- Protective Cover
- Contact Terminal
- Electromagnetic Coil Windings
- Reed Capsule
- Contact Terminal
- Magnetic attractive force
**Solenoids**

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 \( \frac{di}{dt} \) 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 Interfaces**
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.
**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.
- Used in disk drives and printers.
- Can also be used as shaft encoders to measure both position and speed.

---

**Stepper Motor Sequence**

<table>
<thead>
<tr>
<th>PortB</th>
<th>A</th>
<th>A'</th>
<th>B</th>
<th>B'</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Activate</td>
<td>deactivate</td>
<td>activate</td>
<td>deactivate</td>
</tr>
<tr>
<td>9</td>
<td>Activate</td>
<td>deactivate</td>
<td>deactivate</td>
<td>activate</td>
</tr>
<tr>
<td>5</td>
<td>Deactivate</td>
<td>activate</td>
<td>deactivate</td>
<td>activate</td>
</tr>
<tr>
<td>6</td>
<td>Deactivate</td>
<td>activate</td>
<td>activate</td>
<td>deactivate</td>
</tr>
</tbody>
</table>

---

**Simple 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.
Helper Functions to Control Stepper Motor

```c
const struct State
{  unsigned char Out; /* Output for this state */
   const struct State *Next[2]; /* Next state */
} stateType;

unsigned char POS;/* b/w 0 and 199, shaft angle */
#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]} }; 

StatePtr Pt; /* Current State */
```

Slide 25

Linked List to Control Stepper Motor

```c
; Linked list stored in EEPROM
S10: dc b 10 ; Output pattern
    dc w S9 ; Next if CW
    dc w S6 ; Next if CCW
S9: dc b 9
    dc w S10
    dc w S5
S5: dc b 5
    dc w S9
    dc w S6
S6: dc b 6
    dc w S5
    dc w S10

; Global variables stored in RAM
POS: ds 1 ; 0 <= POS <= 199
PT: ds 2 ; to current state
```

Slide 27

Helper Functions to Control Stepper Motor

```c
void CW(void){
   Pt=Pt->Next[clockwise];
   PORTB=Pt->Out;
   if (POS++==200) POS=0;
}
void CCW(void){
   Pt=Pt->Next[counterclockwise];
   PORTB=Pt->Out;
   if (POS==0) POS=199;
   else POS--;
}
void Init(void){
   POS=0;
   Pt=&fsm[0];
}
```

Slide 26

Helper Functions to Control Stepper Motor

```c
Init: clr POS
    ldx #S’0
    stx PT
    rts
```

Slide 28
Helper Functions to Control Stepper Motor

; Move 1.8 degrees clockwise

CW:  
1dx PT ; current state
1dx 1,X ; next clockwise
stx PT ; update pointer
1dlaa ,X ; output pattern
staa PORTB ; set phase control
1dlaa POS ; update position
inca ; clockwise
cmpa #200
blo OK1 ; 0<=POS<=199
cra
OK1: staa POS
rts

Helper Functions to Control Stepper Motor

; Move 1.8 degrees counterclockwise

CCW:  
1dx PT ; current state
1dx 3,X ; next CCW
stx PT ; update pointer
1dlaa ,X ; output pattern
staa PORTB ; set phase control
1dlaa POS ; update position
deca ; CCW direction
cmpa #255
bne OK2 ; 0<=POS<=199
1dlaa #199
OK2: staa POS
rts

High-Level Control of Stepper Motor

void SEEK(unsigned char New){  
  int CWsteps,i;
  if((CWsteps=New-POS)<0)
    CWsteps+=200;
  if(CWsteps>100)
    for(i=CWsteps;i<200;i++)
      CCW();
  else
    for(i=0;i<CWsteps;i++)
      CW(); }

High-Level Control of Stepper Motor

; Reg B=desired 0<=RegB<=199
SEEK:  
  pshb ; Save desired
  tsy
  subb POS ; Go CW or CCW?
  beq DONE ; Skip if equal
  bhi HIGH ; Desired>POS?
  ; Desired<POS
  negb ; (POS-Desired)
  cmpb #100
  bne GOCCW ; Go CCW if
High-Level Control of Stepper Motor

; Desired<POS and POS-Desired<100
GOCW: bsr CW ; Reg A current
    cmpa ,Y
    bne GOCW ; POS=Desired?
    bra DONE
HIGH: cmpb #100 ; (Desired-POS)
    blo GOCW ; Go CW if
; Desired>POS and Desired-POS<100
GOCCW: bsr CCW ; Reg A current
    cmpa ,Y
    bne GOCCW ; POS=Desired?
DONE: pulb
    rts ; Return

Stepper Motor Basic Operation

18
Stepper Motor Basic Operation (cont)

Effective Approach to Changing Motor Speed

(See Tables 8.14 and 8.15)
Timing of Stepper Motor as Shaft Position Sensor

Slide 45

(clockwise)

(counterclockwise)