Memory transfer

```
MOV R0, #30H       ; Move 30H into R0
MOV R1, #40H       ; Move 30H into R0
MOV R3, #05H       ; Set counter as 5
AGAIN: MOV A, @R0  ; Move content of R0 into A
MOV @R1, A
INC R0                 ; Increment R0
INC R1            ; Increment R1
DJNZ R3, AGAIN
END
```

LED FLash Program

```
ORG     0000H
SJMP    0030H
ORG 0030H
GO:
MOV P1, #00H
ACALL DELAY
MOV P1, #0FFH
ACALL DELAY
AJMP GO

DELAY:
MOV     R1,#0F0H      ; delay as per user
MOV     R2,#0F0H
MOV     R3,#10H

D1:
DJNZ    R1,D1
MOV     R1,#0A0H
DJNZ    R2,D1              ;decrement and jump if not zero R2
MOV     R2,#0A0H
DJNZ    R3,D1              ;decrement and jump if not zero R3
RET

END
```

Stepper motor

```
;program for stepper motor thr.port 0 of 89v51RD2,
;lines used p0.0 to p0.3
;motor rotates in forward direction
;to change the direction use "Reverse" look up table

ORG    0000h
SJMP   0030H

ORG    0030H


MOV    P0,#80H   ; initialise port 0
MOV    P2,#80H   ;initialise port 2
ACALL  DELAY      ;delay

START1:
MOV    DPTR,#REVERSE  ;load forward direction table
MOV    R7,#08H             ; steps
NEXT:  CLR    A

MOVC   A,@A+DPTR       ; ini. A for signal.
INC    DPTR
MOV    P2,A                 ;out stepper signal to p2
MOV    P0,A         ; out stepper signal  to p0
ACALL  DELAY
DEC    R7
CJNE   R7,#00,NEXT       ; keep in step loop
SJMP   START1           ; keep in continous loop

DELAY:
MOV    R1,#0F2H         ; delay as per user
MOV    R2,#0F2H              ; motor speed is depend on this delay
MOV    R3,#0f8H              ; if you want slow speed increase delay

D1:
DJNZ   R1,D1
MOV    R1,#010H
DJNZ   R2,D1
MOV    R2,#010H
DJNZ   R3,D1
RET

FORWARD: DB 01H,03H,02H,06H,04H,0CH,08H,09H  ; forward direction  table
REVERSE: DB 09H,08H,0CH,04H,06H,02H,03H,01H  ;revese direction table
END
```

LED, Buzzer, Relay, Pushbuttons with PIC

```
#include <p18f4520.h>
#pragma config FOSC = HS
#pragma config WDT = OFF
#pragma config LVP = OFF
#pragma config PBADEN = OFF


#define lrbit PORTBbits.RB4   //SW0 interfaced to RB4
#define rlbit PORTBbits.RB5   //SW1
#define buzzer PORTCbits.RC2
#define relay PORTCbits.RC1

void MsDelay (unsigned int time)
{
    unsigned int i, j;
    for (i = 0; i < time; i++)
        for (j = 0; j < 275; j++);/*Calibrated for a 1 ms delay in MPLAB*/
}

void main()
{
  unsigned char val=0;
  INTCON2bits.RBPU=0;   //To Activate the internal pull on PORTB
    ADCON1 = 0x0F;              //To disable the all analog inputs

    TRISBbits.TRISB4=1;  //To configure RB4 as input
    TRISBbits.TRISB5=1;       //To configure RB5 as input

    TRISCbits.TRISC1=0;   //To configure RC1 (relay)  as output
    TRISCbits.TRISC2=0;       //To cofigure RC2 (buzzer) as output

    TRISD = 0x00;             // To cofigure PORTD (LED) as output

    PORTD = 0x00;              //Initial Value for LED
    buzzer = 0;                //Initial Value for Buzzer
    relay = 0;               //Initial Value for Relay

while (1)
    {
    if (!(lrbit)) // if (lrbit == 0)             //To check whether SW0 is pressed
    val = 1;                  // Latch the SWO
    if (!(rlbit))            //To check whether SW1 is pressed
    val = 2;                  // Latch the SW1

    if (val == 1)
    {
        buzzer = 1;
        relay = 1;
        PORTD = PORTD >>1;  //Shift left by 1 bit
            if (PORTD == 0x00)
                PORTD = 0x80; // Make the MSB bit eqaul to 1
        MsDelay(250);
    }
    if (val == 2)
    {
        buzzer = 0;
        relay = 0;
        PORTD = PORTD<<1; //Shift right by 1 bit
            if (PORTD == 0x00)
                PORTD = 0x01;  // Make the LSB bit eqaul to 1
        MsDelay(250);
    }
    }

}
```

Square wave

```
#include<p18f458.h>
#define mybit PORTBbits.RB4
#pragma config OSC=HS
#pragma config PWRT=OFF
#pragma config WDT=OFF
#pragma config DEBUG=ON
#pragma config LVP=OFF

void main(void);
void Timerdelay(void);
void main()
{
TRISBbits.TRISB4=0;
while(1)
{
mybit^=0;
mybit = ~mybit;
Timerdelay();
}
}
void Timerdelay()
{
T0CON =0x07;
TMR0H = 0xFF;
TMR0L = 0xF0;
INTCONbits.TMR0IF =1;
T0CONbits.TMR0ON =1;
while(INTCONbits.TMR0IF==0)
INTCONbits.TMR0IF=0;
T0CONbits.TMR0ON =0;
}
```

Generation of PWM signal using CCP module

```
#include<p18f4550.h>

#pragma config FOSC = HS
#pragma config WDT = OFF
#pragma config LVP = OFF
#pragma config PBADEN = OFF

void myMsDelay (unsigned int time)
{
        unsigned int i, j;
        for (i = 0; i < time; i++)
                for (j = 0; j < 275; j++);/*Calibrated for a 1 ms delay in MPLAB*/
}

void main()
{
        TRISCbits.TRISC2 = 0 ;          // Set PORTC, 2 as output
  TRISCbits.TRISC6 = 0 ;
        TRISCbits.TRISC7 = 0 ;
        PR2 = 0x7F;                     // set PWM period to Maximum value
  CCPR1L = 0x12;                  // Initalise PWM duty cycle to 00
  CCP1CON = 0x3C;                 // Configure CCP1CON as explained above.
        T2CON = 0x07;

                myMsDelay(50);
                CCPR1L = 0x7F;
                myMsDelay(50);
PORTCbits.RC6 = 1;
  PORTCbits.RC7 = 0;
 while(1)
        {
 CCPR1L = 0x0F;
                myMsDelay(50);
CCPR1L = 0x1F;
                myMsDelay(50);


                CCPR1L = 0x2F;
                myMsDelay(50);
                CCPR1L = 0x3F;
                myMsDelay(50);
                CCPR1L = 0x4F;
                myMsDelay(50);
                CCPR1L = 0x5F;
                myMsDelay(50);
                CCPR1L = 0x6F;
        }

}
```