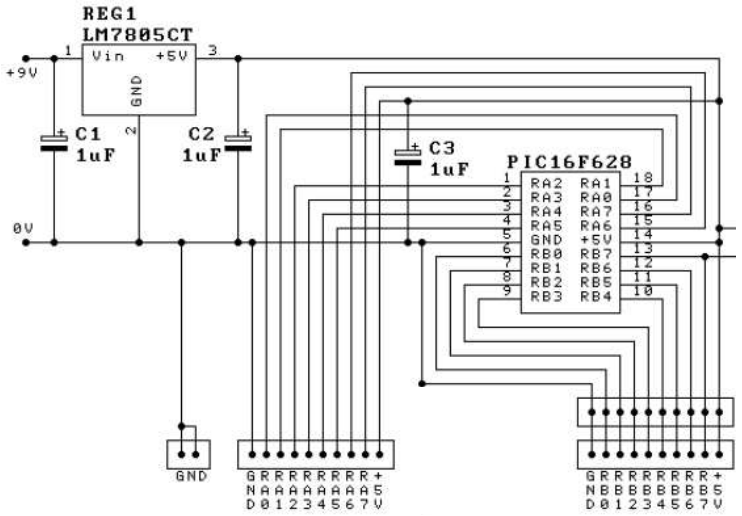
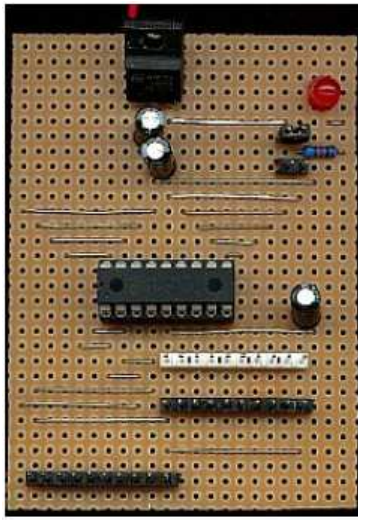
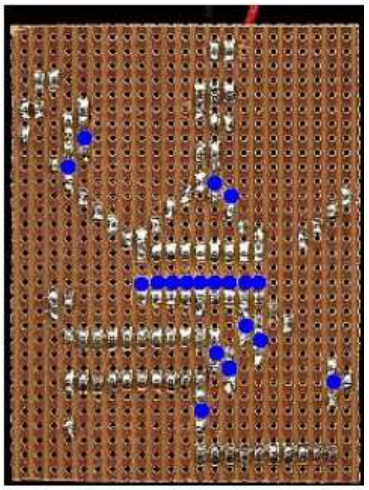
Simple PIC Application



Utilize internal oscillator

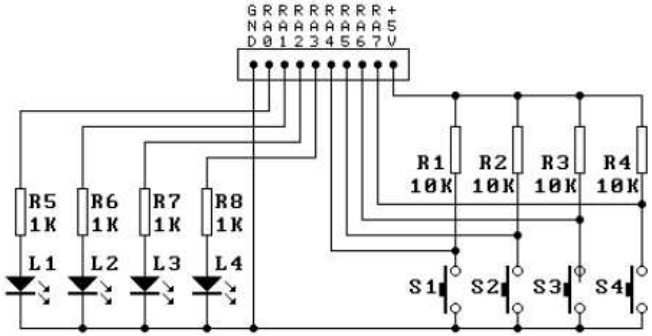
Simple PIC Application

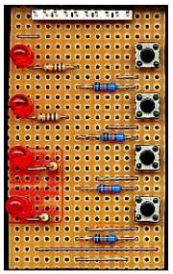




Simple PIC Application

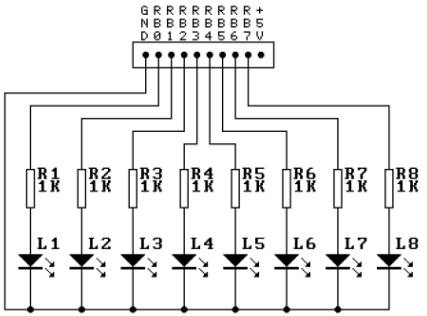
I/O Circuit

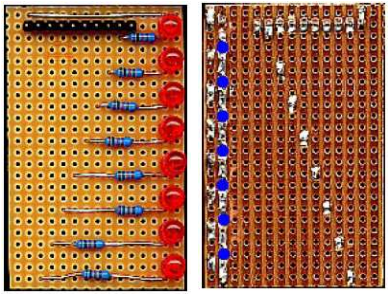




Simple PIC Application

Display Circuit





Simple PIC Application

```
LIST p=16F628                ;tell assembler what chip we are using
#include <p16f628.inc>       ;include the defaults for the chip

        org   0x0000
        movlw 0x07
        movwf CMCON          ;turn comparators off
        bsf   STATUS, RP0    ;select bank 1
        movlw b'00000000'    ;set PortB all outputs
        movwf TRISB
        movwf TRISA          ;set PortA all outputs
        bcf   STATUS, RP0    ;select bank 0
Loop    movlw 0xff
        movwf PORTA          ;set all bits on
        movwf PORTB
        nop                  ;make up the time taken by the goto
        nop                  ;giving a square wave output
        movlw 0x00
        movwf PORTA
        movwf PORTB          ;set all bits off
        goto  Loop           ;go back and do it again
        end
```

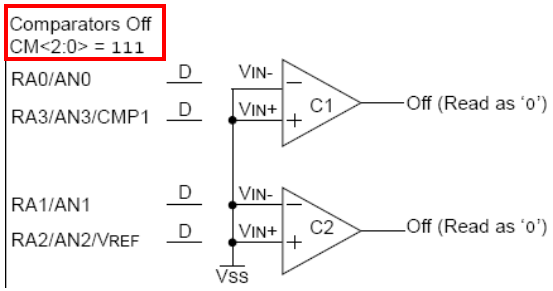
See MPLAB Demo

Simple PIC Application

CMCON – COMPARATOR CONFIGURATION REGISTER (ADDRESS: 01Fh)

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit 7							bit 0

```
movlw   0x07
movwf   CMCON
```



Simple PIC Application

STATUS – STATUS REGISTER (ADDRESS: 03h, 83h, 103h, 183h)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit 7							bit 0

RP<1:0>: Register Bank Select bits (used for direct addressing)

00 = Bank 0 (00h-7Fh)
01 = Bank 1 (80h-FFh)
10 = Bank 2 (100h-17Fh)
11 = Bank 3 (180h-1FFh)

```
bsf     STATUS,         RP0
```

Simple PIC Application

STATUS – STATUS REGISTER (ADDRESS: 03h, 83h, 103h, 183h)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit 7							bit 0

Z: Zero bit
1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is not zero

DC: Digit Carry/$\overline{\text{Borrow}}$ bit (ADDWF, ADDLW, SUBLW, SUBWF instructions) (for $\overline{\text{Borrow}}$ the polarity is reversed)
1 = A carry-out from the 4th low order bit of the result occurred
0 = No carry-out from the 4th low order bit of the result

C: Carry/$\overline{\text{Borrow}}$ bit (ADDWF, ADDLW, SUBLW, SUBWF instructions)
1 = A carry-out from the Most Significant bit of the result occurred
0 = No carry-out from the Most Significant bit of the result occurred

Interrupt

The PIC16F627A/628A/648A has 10 sources of interrupt:

- **External Interrupt RB0/INT**
- TMR0 Overflow Interrupt
- **PORTB Change Interrupts (pins RB<7:4>)**
- Comparator Interrupt
- USART Interrupt TX
- USART Interrupt RX
- CCP Interrupt
- TMR1 Overflow Interrupt
- TMR2 Match Interrupt
- Data EEPROM Interrupt

Interrupt

- The Interrupt Control register (INTCON) records individual interrupt requests in flag bits.
- It also has individual and global interrupt enable bits.
- A Global Interrupt Enable bit, GIE (INTCON<7>) enables or disables all interrupts.
- Individual interrupts can be disabled through their corresponding enable bits in INTCON register.
- GIE is cleared on Reset.

Interrupt

INTCON – INTERRUPT CONTROL REGISTER (ADDRESS: 0Bh, 8Bh, 10Bh, 18Bh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
bit 7							bit 0

GIE: Global Interrupt Enable bit
1 = Enables all un-masked interrupts
0 = Disables all interrupts

PEIE: Peripheral Interrupt Enable bit
1 = Enables all un-masked peripheral interrupts
0 = Disables all peripheral interrupts

T0IE: TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

INTE: RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

Interrupt

INTCON – INTERRUPT CONTROL REGISTER (ADDRESS: 0Bh, 8Bh, 10Bh, 18Bh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
bit 7							bit 0

RBIE: RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

T0IF: TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in s/w)
0 = TMR0 register did not overflow

INTF: RB0/INT External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in s/w)
0 = The RB0/INT external interrupt did not occur

RBIF: RB Port Change Interrupt Flag bit
1 = When at least one of RB<7:4> pins changes state (must be cleared in s/w)
0 = None of the RB<7:4> pins have changed state

Interrupt

OPTION_REG – OPTION REGISTER (ADDRESS: 81h, 181h)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0

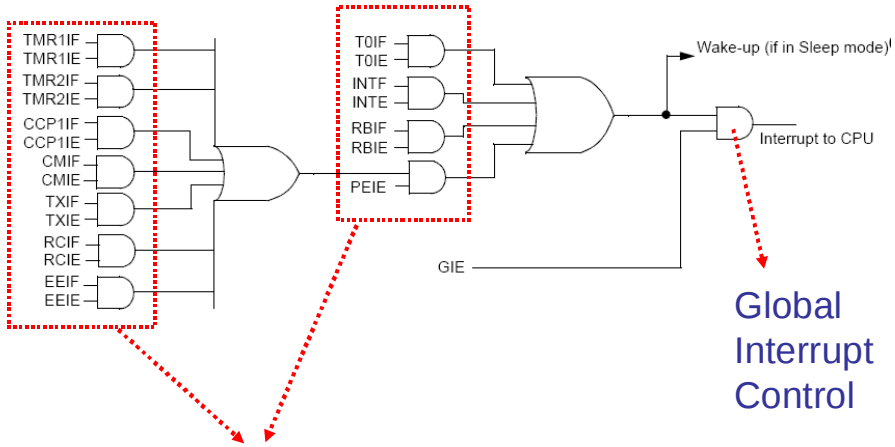
bit 6 **INTEDG**: Interrupt Edge Select bit

1 = Interrupt on rising edge of RB0/INT pin

0 = Interrupt on falling edge of RB0/INT pin

Bank 0		Bank 1		Bank 2		Bank 3	
Indirect addr.(1)	00h	Indirect addr.(1)	80h	Indirect addr.(1)	100h	Indirect addr.(1)	180h
TMR0	01h	OPTION	81h	TMR0	101h	OPTION	181h
PCL	02h	PCL	82h	PCL	102h	PCL	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
FSR	04h	FSR	84h	FSR	104h	FSR	184h
PORTA	05h	TRISA	85h		105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
	07h		87h		107h		187h
	08h		88h		108h		188h
	09h		89h		109h		189h
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch		10Ch		18Ch
	0Dh		8Dh		10Dh		18Dh
TMR1L	0Eh	PCON	8Eh		10Eh		18Eh
TMR1H	0Fh		8Fh		10Fh		18Fh
T1CON	10h		90h				
TMR2	11h		91h				

Interrupt Logic



Interrupt Service Routine

PIC defines interrupt vector at 004h

```
                org 0x000
                goto start

                org 0x004
                goto ISR

start:          ;main program starts here
                  ⋮

loop:           goto loop       ;keep main program running

ISR:            ;put interrupt service routine here
                  ⋮
                RETFIE
```

Interrupt Example

```
            w       equ     0
            f       equ     1
            porta   equ     0x05
            portb   equ     0x06
            intcon  equ     0x0b
            cmcon   equ     0x1f
            org     0x000
            goto    start
            org     0x004
            goto    int_serv
start       movlw   b'01111111' ;select Ext. Int. Rising Edge
            option              ;this inst. is not recommended
            bcf     intcon,1    ;clear INT Flag
            bsf     intcon,7    ;enable GIE
            bsf     intcon,4    ;enable Ext. Int.
            clrf    porta       ;initialize PORTA
            movlw   0x07        ;turn off analog comparator
            movwf   cmcon       ;
            movlw   b'11111111' ;assign PORTA as input
            tris    porta
            movlw   b'00000001' ;assign PORTB.0 as input
            tris    portb
            clrf    portb
test        btfss   porta,0     ;check PORTA.0
            goto    clrb_1      ;if PORTA.0 = 0, goto clrb_1
            bsf     portb,1     ;if PORTA.0 = 1, PORTB.1 = 1
            goto    test        ;check PORTA.0 again
clrb_1      bcf     portb,1     ;PORTB.1 = 0
            goto    test        ;check PORTA.0 again
int_serv    bsf     portb,2     ;if Ext. Int. occurs
loop        goto    loop        ;trap in ISR, reset required
            end
```

See Demo