16F628A megszakítás kezelése

A 'megszakítás' azt jelenti, hogy a program normális, szekvenciális futása valamilyen külső hatás miatt átmenetileg felfüggesztődik, és a vezérlést egy külön rutin, a megszakításkezelő kapja meg. Miután a megszakításkezelő végzett, a program a futását - mintha semmi se történt volna - folytatja.

A külső hatások tipikusan: jelváltozás valamelyik bemeneten, a hardver számláló / időzítő 255-ről 0-ra váltása (túlcsordulása), vagy pld. az A/D konverter jelzése, hogy készen van a konverzió.

A megszakítás vezérlő feladata, hogy:
- legyenek megszakítások
- egyesével, vagy globálisan lehessen a megszakításokat engedélyezni, vagy letiltani. Kezdetben valószínűleg nem fogsz a megszakításokkal foglalkozni, de hamar elérkezik ennek is az ideje. Ha nem kell az interrupt, gondoskodj arról, hogy az INTCON regiszter 00-ban legyen. Ezzel minden megszakítást letiltottál.

INTCON REGISTER (ADDRESS 0Bh, 8Bh)

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	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
bit 7							bit 0

bit 7 **GIE:** Global Interrupt Enable bit **BSF INTCON, 7**
1 = Enables all unmasked interrupts
0 = Disables all interrupts

bit 6 **EEIE**: EE Write Complete Interrupt Enable bit
1 = Enables the EE Write Complete interrupts
0 = Disables the EE Write Complete interrupt

bit 5 **T0IE**: TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt **BSF INTCON, 5**
0 = Disables the TMR0 interrupt

bit 4 **INTE**: RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

bit 3 **RBIE**: RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt **BSF INTCON, 3**
0 = Disables the RB port change interrupt

bit 2 **T0IF**: TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software) **BCF INTCON, 3**
0 = TMR0 register did not overflow

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

bit 0 **RBIF**: RB Port Change Interrupt Flag bit **BCF INTCON, 0**
1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

GCB programban:

IntOff
Tiltja a mikrovezérlő megszakítás kezelést

IntOn
Engedélyezi a mikrovezérlő megszakítás kezelését

Példa 1

This example shows if an event occurs the microcontroller will be program to jump to the interrupt vector and the program will not know the event type, it will simple execute the Interrupt subroutine. This code is not intended as a meaningful solution and intended to show the functionality only. An LED is attached to PORTB.1 via a suitable resistor. It will light up when the Interrupt event has occurred.

```
#chip 16f628a, 4
dir PORTB.1 out
Set PORTB.1 Off

'Note: there is NO On Interrupt handler
InitTimer1 Osc, PS1_8
SetTimer 1, 1
StartTimer 1
'Manually set Timer1Overflow to the overflow event
'this will event will be handled by the Interrupt sub routine
TMR1IE = 1
end

Sub Interrupt
  Set PORTB.1 On
  TMR1IF = 0
End Sub
```

Példa 2

Any events that are not dealt with by On Interrupt will result in the code in the Interrupt subroutine executing. This example shows the operation of two interrupt handlers - is not intended as a meaningful solution.

LEDs are attached to PORTB.1 and PORTB.2 via suitable resistors. They will light up when the Interrupt events occur.

```
#chip 16f628a, 4
On Interrupt Timer1Overflow call Overflowed

dir PORTB.1 out
Set PORTB.1 Off

dir PORTB.2 out
Set PORTB.2 Off

InitTimer1 Osc, PS1_8
SetTimer 1, 1
StartTimer 1
```

```
InitTimer2 PS2_16, PS2_16
SetTimer 2, 255
StartTimer 2

'Manually set Timer2Overflow to create a second event
'this will event will be handled by the Interrupt sub routine
TMR2IE = 1
end

Sub Interrupt
  Set PORTB.2 On
  TMR2IF = 0
End Sub

Sub Overflowed
  Set PORTB.1 On
  TMR1IF = 0
End Sub
```

Példa 3

This code implements four flashing LEDs. This is based on the Microchip PIC Low Pin Count Demo Board.

The example program will blink the four red lights in succession. Press the Push Button Switch, labeled **SW1**, and the sequence of the lights will reverse. Rotate the potentiometer, labeled **RP1**, and the light sequence will blink at a different rate.

This implements an interrupt for the switch press, reads the analog port and set the LEDs.

Demonstration program:

```
#chip 18F14K22, 32
#config OSC = IRC, MCLRE_OFF

'Works with the low count demo board

'Set the input pin direction
    #define SwitchIn1 PORTa.3
    Dir SwitchIn1 In

#define LedPort PORTc
    DIR PORTC OUT

'Setup the ADC  pin direction
    Dir PORTA.0 In
    dim ADCreading as word


'Setup the input pin direction
    #define IntPortA PORTA.1
    Dir IntPortA In

'Variable and constants
    #define intstate as byte
    intstate = 0
    #define minwait 1

    dim ccount as byte
    dim leddir as byte

    ccount = 8
    leddir = 0
```

```
        SET PORTC = 15
        WAIT 1 S

        SET PORTC = 0

    'Setup the Interrupt
        Set IOCA.3 on
        Dir porta.3 in
        On Interrupt PORTABCHANGE Call Setir

    'Set initial LED direction
        setLedDirection

    DO FOREVER


        INTON
        ADCreading = ReadAD10(AN0)
        if ADCreading < minwait then ADCreading = minwait

        'Set LEDs
        Set PortC = ccount
        wait ADCreading ms

        if leddir = 0 then
          rotate ccount left simple
          'Restart LED position
          if ccount = 16 then
              ccount = 128
          end if

        end if

        if leddir = 1 then
          rotate ccount Right simple
          'Restart LED position
          if ccount = 128 then
            ccount = 8
          end if

        end if
        'Reset interrupt - this may be been reset so set to zero so interrupt
can operate.
        intstate = 0

    Loop


    'Interrupt routine.
    sub Setir

        if IntPortA  = 0 and intstate = 0 then
          intstate = 1
          wait while SwitchIn1 = 0
          setLedDirection
        end if


    end sub

    sub setLedDirection

      'Set LED values
```

```
    select case leddir

      case 0
        leddir = 1
        ccount  = 8

      case 1
        leddir = 0
        ccount = 1

      end select

  End Sub
```

REGISTER 4-3: 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

bit 7 **GIE**: Global Interrupt Enable bit
1 = Enables all un-masked interrupts
0 = Disables all interrupts

bit 6 **PEIE**: Peripheral Interrupt Enable bit
1 = Enables all un-masked peripheral interrupts
0 = Disables all peripheral interrupts

bit 5 **T0IE**: TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

bit 4 **INTE**: RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

bit 3 **RBIE**: RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **T0IF**: TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

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

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