Les Entrées Digitales interruptibles

Contenu

Interrupt Port

Dans les exemples précédents, le programme tourne en boucle et contrôle à chaque itération de celle-ci, l'état de la broche associée par exemple à un bouton (polling). Cette méthode utilise beaucoup de temps processeur pour un évènement qui peut avoir lieu qu'une fois par heure...

Les interruptions associées aux ports d'entrée ***Interrupt ports***, permettent d'utiliser une méthode qui sera exécutée lorsque l'état du bouton sera modifié (par exemple lorsque la broche est à l'état bas).

Les *Interrupt ports* ne sont pas utilisables avec toutes les entrées digitales !

Nous pouvons définir une interruption (appel de méthode) sur plusieurs changements d'état d'une broche:

- état haut,
- état bas,
- changement d'état.

L'utilisation la plus courante est le changement d'état. Le changement de bas en haut ou de haut en bas va générer un flanc (*signal edge*) montant (*high edge*, de 0 à 1) ou descendant (*low edge*, de 1 à 0).

Les modes *InterruptEdgeLevelHigh* et *InterruptEdgeLevelLow* provoque des exceptions !

Dans l'exemple ci-dessous, nous utilisons un flanc descendant pour détecter une pression sur le bouton. La méthode « **IntButton_OnInterrupt** » s'exécute automatiquement lorsque l'on appuie sur le bouton.

```
1. using System;
2. using System.Threading;
3. using Microsoft.SPOT;
4. using Microsoft.SPOT.Hardware; // Définitions pour les IO digitales
5.
6. namespace change_this_to_your_namespace
7. {
8.   public class Program
9.   {
10.    public static void Main()
11.    {
12.      InterruptPort IntButton = new InterruptPort(EMX.Pin.IOxx, // choix de l'entrée
13.          false, // Filtre anti-rebonds
14.          Port.ResistorMode.PullUp, // choix de la rés. d'entrée
15.          Port.InterruptMode.InterruptEdgeLow); // choix du mode de détection
16.    // Associe le gestionnaire d'interruption à l'évènement
```

```
17.     IntButton.OnInterrupt += new NativeEventHandler(IntButton_OnInterrupt);
18.   // … suite du code …
19.     Thread.Sleep(Timeout.Infinite);
20.   }
21.
22.   static void IntButton_OnInterrupt(uint port, uint state, DateTime time)
23.   {
24.     Debug.Print("Button Pressed");
25.   }
26. }
27.}
```

Lors de la création du nouvel objet InterruptPort, le second argument indique si le filtre anti-rebonds est activé (true) ou désactivé (false). Cela est généralement nécessaire lorsque vous utilisez la fonction d'interruption pour éviter les rebonds sur le bouton. La non activation de cette fonction peut provoquer le déclencher de plusieurs événements, même si le bouton n'est pressé qu'une fois!

Dans le gestionnaire d'événements (... *OnInterrupt (uint port, uint state ...),* l'argument ***state*** est l'état de la broche **après** le flanc de transition, c'est-à-dire:

- à *true* après un flanc montant;
- à false après un flanc descendant.

Syntaxe et paramètres de la méthode InterruptPort()

```
public InterruptPort     (
                           Cpu.Pin portId,
                           bool glitchFilter,
                           Port.ResistorMode resistor,
                           Port.InterruptMode interrupt
                         )
```

Paramètres

portId	Identifie la broche du port d'entrée. Type: Microsoft.SPOT.Hardware.Cpu.Pin
glitchFilter	Permet d'activer le filtre anti-rebonds. Type: System.Boolean
Resistor	Définit l'état par défaut de la résistance d'entrée du port. Type: Microsoft.SPOT.Hardware.Port.ResistorMode
Interrupt	Détermine le mode de déclenchement de l'interruption du port. Type: Microsoft.SPOT.Hardware.Port.InterruptMode

Enumeration Port.InterruptMode

Member name	Description
InterruptNone	ne génère pas d'interruption.
InterruptEdgeLow	déclenche l'interruption sur le front descendant (falling edge).
InterruptEdgeHigh	déclenche l'interruption sur le front montant (rising edge).
InterruptEdgeBoth	déclenche l'interruption sur les deux fronts.
InterruptEdgeLevelHigh	déclenche l'interruption lorsque le niveau d'entrée est à l'état Haut.
InterruptEdgeLevelLow	déclenche l'interruption lorsque le niveau d'entrée est à l'état bas.

Autres méthodes

Nom	Description
`public void ClearInterrupt()`	Efface l'interruption courante sur le port d'interruption.
`public override void DisableInterrupt()`	Désactive les interruptions sur cet objet InterruptPort
`public override void EnableInterrupt()`	Active les interruptions sur cet objet InterruptPort
`public bool Read()`	Retourne la valeur booléenne de l'état de à l'entrée du port. (Hérité de Port).
`public virtual string ToString()`	`Retourne une chaîne qui représente l'objet actuel. (Hérité de Object.).`

Propriétés

Nom	Description
GlitchFilter	Obtient ou définit l'état du filtre anti-rebonds du port d'entrée. (Hérité de InputPort.)
Id	Obtient l'identificateur (ID) pour un port. (Hérité de Port.)
Interrupt	Obtient ou définit le mode d'interruption du port.
Resistor	Obtient ou définit le mode de la résistance d'entrée. Vous définissez la valeur du mode de résistance initiale dans le constructeur InputPort. (Hérité de InputPort.)