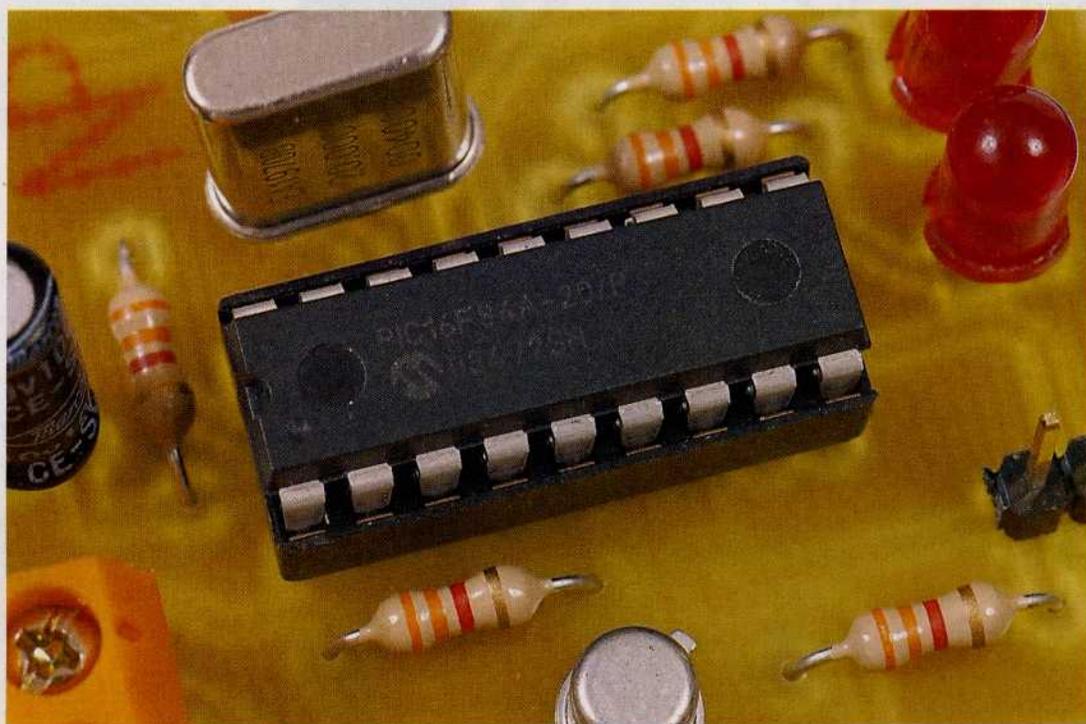


À la découverte des microcontrôleurs PIC

(Huitième partie)



Nous allons, dans ce numéro, aborder l'étude des interruptions sur le microcontrôleur PIC. Comme nous le verrons tout au long de cet article, les interruptions vont nous offrir des possibilités de gestion d'un programme par rapport à un événement extérieur.

Imaginez vous tranquillement assis dans votre fauteuil à regarder votre film préféré lorsque soudain, le téléphone sonne.

Le premier réflexe qui vous vient à l'esprit est sans doute de mettre le magnétoscope en mode pause afin de ne rien manquer du film merveilleux que vous regardez et d'aller répondre à l'appel téléphonique.

Une fois votre conversation terminée, vous revenez devant votre écran et vous remettez le magnétoscope en mode lecture. Nous pourrions dire que vous avez été interrompu pendant votre occupation, vous avez traité le plus urgent.

Pour le programme d'un microcontrôleur qui se déroule "tranquillement", le principe est le même. Une tâche de fond s'exécute, celle-ci peut être assimilée à votre programme principal,

par exemple vous faites clignoter une led à une fréquence de 1 Hz. Soudain, l'utilisateur appuie sur le bouton "Fréquence rapide" (c'est notre appel téléphonique), que va faire votre programme ?

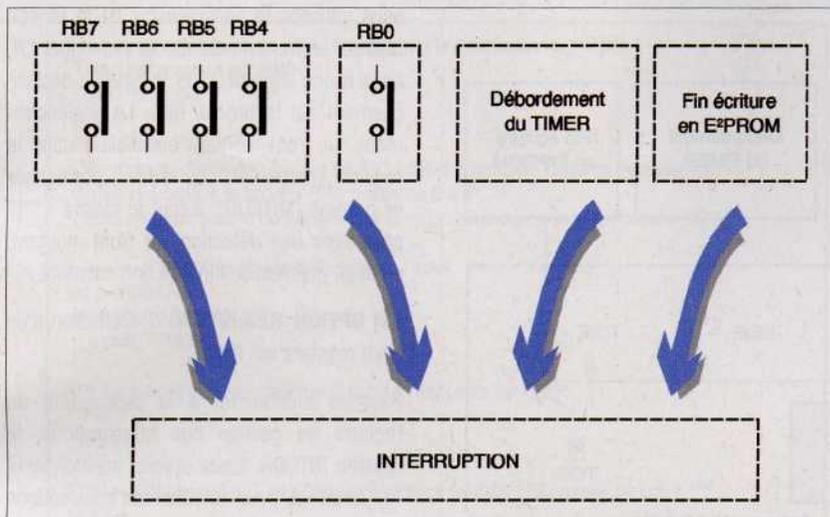
En fait, il fait comme vous avez fait précédemment, c'est à dire que quel que soit l'endroit où l'on se situe dans le programme (cela correspond au film que vous visualisez dans l'exemple imagé), il exécute le plus urgent, il détecte l'appui sur le bouton et enregistre cette information. Ensuite, l'appui étant mémorisé, le programme "revient" à l'endroit où il s'était arrêté auparavant et la led clignote plus vite (une fois que vous avez répondu au téléphone vous remettez le magnétoscope en lecture).

Pour terminer cette introduction, on peut dire qu'une interruption est une

action extérieure obligeant un programme à traiter un sous programme prioritaire. Une fois ce sous programme terminé, le microcontrôleur revient au programme principal à l'endroit où il l'a quitté avant l'action extérieure. On dit également que le programme est "dérouté" vers un sous-programme d'interruption.

1 L'appel téléphonique est prioritaire





2 Quatre type d'interruptions possibles sur le PIC 16F84

Il existe sur le PIC de la famille 16F84 plusieurs types d'événements pouvant provoquer une interruption, ils sont au nombre de quatre. Sur d'autres PIC tel que le PIC16F877, on trouvera 15 sources d'interruptions, ces sources d'IT sont en rapport bien sûr avec les différentes fonctionnalités ainsi que le nombre de registres internes propres à chaque PIC :

Pour le PIC 16F84 on aura une interruption sur :

- Un changement d'état sur la broche RB0
- Un changement d'état sur une des broches RB4 à RB7
- La fin de l'écriture en E_PROM
- Le débordement du Timer interne

Comme nous venons de le voir, quatre sources différentes d'événements peuvent déclencher une interruption et le programmeur (c'est à dire vous), aura la possibilité de valider telle ou telle source d'interruption. Cette possibilité se définit dans un registre du PIC, le registre INTCON (INTerrupt CONtrol). Dans ce registre interne de huit bits, il existe quatre bits permettant d'autoriser une ou plusieurs sources d'interruptions, trois bits permettant d'indiquer d'où vient l'interruption ainsi qu'un bit permettant de valider globalement les interruptions.

Vous l'avez peut être remarqué, le nom des bits est significatif, les noms se terminant par F (comme flag) sont les indicateurs, les noms

se terminant par E (enabled) sont les bits permettant de valider la source d'interruption.

Rôle des bits du registre INTCON

Bit GIE (Global Interrupt Enabled bit)

Ce huitième bit du registre permet de valider les interruptions. C'est une validation générale, cela signifie que si ce bit est positionné à "1", alors le programme principal pourra être dérivé par une interruption. Dans le cas contraire ("0"), alors aucun événement ne pourra provoquer une IT (interruption), toutes les sources susceptibles de dérouter le programme sont donc inhibées.

Bit EEIE (Eeprom write complète Interrupt Enabled bit)

Ce bit permet de valider une interruption issue de la fin d'écriture en E_PROM. Si ce bit est positionné à "1", alors à chaque fois que vous allez écrire en E_PROM et à condition que le bit GIE vu précédemment soit également positionné à "1", une interruption sera générée et le programme principal sera dérivé. Cela peut par exemple permettre de savoir si l'écriture E_PROM s'est bien déroulée.

Bit TOIE (Timer 0 Interrupt Enabled bit)

Si ce bit est positionné à "1", alors un débordement du TIMER interne provoquera une interruption. Le TIMER interne est un registre de huit bits qui s'incrémente au rythme de l'horloge. Dès que la valeur de ce registre

atteint 255 et passe à 0 et si le bit GIE vu précédemment est également positionné à "1", alors une IT est générée et le programme principal est dérivé vers le sous-programme d'interruption.

Bit INTE (INTerrupt pin Enabled bit)

Ce cinquième bit valide une interruption suite au changement d'état sur la broche RB0. La broche RB0 peut changer d'état soit sur un front montant, soit sur un front descendant, cette option se définit dans le registre OPTION à l'aide du 6^{ème} bit (INTEDG). Comme pour les autres sources d'interruptions, il faut bien sûr que le bit GIE soit également à "1".

Bit RBIE (RB port change INTerrupt Enabled bit)

Ce quatrième bit permet de valider une interruption suite au changement d'état sur une des broches RB4, RB5, RB6 ou RB7. Comme pour les autres sources d'interruptions, il faut bien sûr que le bit GIE soit également positionné à "1".

Avant de poursuivre sur le rôle des bits du registre INTCON, nous allons parler très brièvement de l'adresse de notre sous-programme d'interruption. En effet, depuis le début du cours, nous avons appris que le programme principal se dérivait mais sans savoir exactement à quelle adresse mémoire. En fait, pour toutes les sources d'IT, il n'y a qu'une seule adresse, c'est l'adresse 04. Cela signifie que pour n'importe quelle source d'interruption (débordement TIMER, changement d'état sur la broche RB0, ou bien RB4-RB7 ou bien encore une fin d'écriture en E_PROM), le programme se "branche" à l'adresse 04 en mémoire.

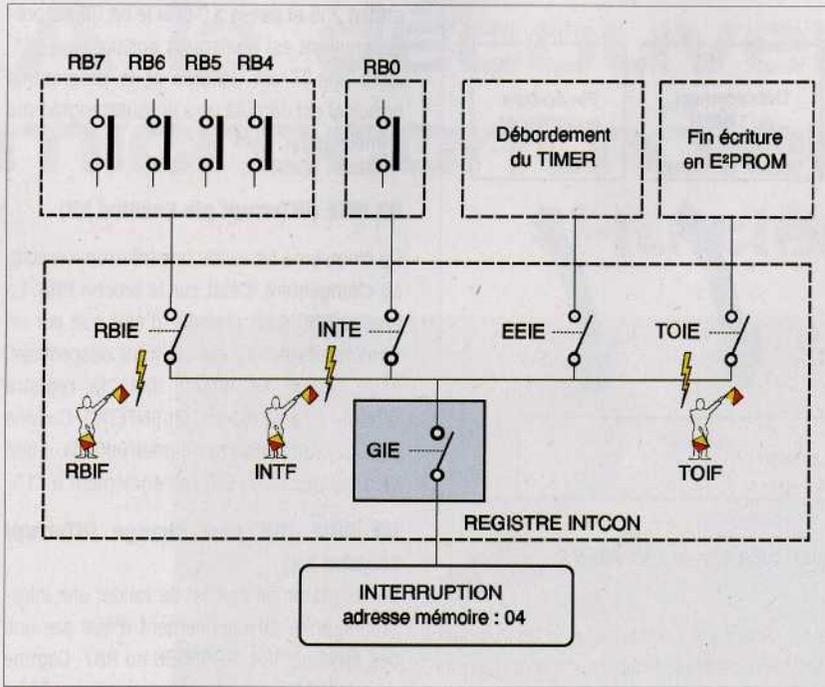
Vous l'avez compris, les flags (les 3 derniers bits du registre INTCON), vont renseigner l'utilisateur sur l'origine de la source d'IT responsable du déroutement du programme. Sans eux, nous ne pourrions savoir d'où provient l'interruption. Dans le sous-programme, nous pourrions alors tester ces différents bits et provoquer tel ou tel traitement selon la source d'IT.

Bit TOIF (Timer 0 Interrupt Flag bit)

Ce bit du registre INTCON sera à "1" lorsqu'une interruption sera issue du débordement du Timer interne. Attention, ce bit n'est pas remis à zéro automatiquement, c'est au programmeur (à vous...) de réaliser cette action (dans le sous programme d'IT bien sûr).

3 Vue du registre INTCON

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
GIE	EEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF



4 Vue générale du cheminement d'une Interruption

Bit INTF (INTerrupt pin Flag bit)

Ce bit du registre INTCON sera à "1" lorsqu'une interruption sera issue du changement d'état de la broche RBO.

Attention, comme cité précédemment, ce bit n'est pas remis à zéro automatiquement. C'est au programmeur (à vous...) de réaliser cette action (dans le sous programme d'IT bien sûr).

Bit RBIF (poRt B Interrupt Flag bit)

Ce bit du registre INTCON sera à "1" lorsqu'une interruption sera issue du changement d'état d'une des broches RB4 à RB7. Comme cité précédemment, il est possible de tester ce bit et celui-ci n'est pas remis à zéro automatiquement. C'est au programmeur (à vous...) de réaliser cette action (dans le sous programme d'IT bien sûr).

Vous l'avez peut être remarqué, il n'y a pas d'indicateur (flag) sur la fin d'écriture en E_PROM, mais sachant que nous avons les trois autres indicateurs, il est facilement concevable de faire la déduction.

En fait, si nous utilisons les quatre sources d'interruptions, il faudra faire trois tests de flags (RBIF, INTF, TOIF) et si aucun d'entre eux n'est à "1", alors l'interruption proviendra de la fin d'écriture en EPROM. Un exemple de programme illustrant ces tests sera fourni en fin d'article.

Intégration dans un programme de la gestion d'une interruption

Le programme commencera toujours par les éternelles directives d'assemblage que nous avons étudiées dans les précédentes leçons, celles-ci ne seront pas détaillées. Comme vous le savez, le programme d'IT se trouvera toujours à l'adresse 4 en mémoire, donc il faudra commencer par déclarer l'adresse de début du programme principal (voir fichier source **Figure 5**) pour "sauter" l'adresse 4 réservée à l'IT, avec une instruction du type :

```
ORG 0 'adresse 0 (dès alimentation du PIC ou RESET)
Goto init 'on saute au programme principal qui commencera à l'étiquette "init"
```

permettant de faire un saut au programme principal grâce à l'instruction "GOTO" dès la mise sous tension du PIC (vecteur Reset). Dans notre programme principal, nous allons déclarer si besoin est la direction des registres TRISA et TRISB (le port B doit être déclaré en entrée pour les broches correspondantes si on utilise des IT sur RBO et RB4 à RB7) et également le positionnement des bits du registre INTCON qui, comme nous l'avons vu, va permettre de configurer les sources d'IT.

Dans le fichier source fournis (**Figure 5**),

nous utilisons le basculement de la broche RBO sur un front montant pour provoquer l'IT, nous allons donc déclarer le type de déclenchement sur la broche RBO. Le déclenchement sur front montant se déclare dans le registre interne OPTION, le bit à positionner se nomme "INTEDG", il faut le mettre à "1" pour avoir une détection sur front montant, c'est ce que réalise l'instruction suivante :

```
bsf OPTION_REG,INTEDG ; Détection d'un front montant sur RBO
```

Passons maintenant à la description du registre de gestion des interruptions, le registre INTCON. Nous devons positionner à "1" le bit GIE pour autoriser les interruptions en général puis nous devons également mettre à "1" le bit "INTE" pour valider la source d'IT en provenance de la broche RBO. Voici les deux instructions qui permettent de configurer le registre INTCON :

```
bsf INTCON,INTE ; on autorise l'IT sur RBO
bsf INTCON,GIE ; on autorise les interruptions
```

Désormais, tout front montant en provenance de la broche RBO sera pris en compte. Passons maintenant au sous-programme d'interruption. Il faut dans un premier temps passer l'adresse du sous programme, la directive ORG 4 se chargera de réaliser cette opération, puis vient ensuite les instructions qui seront exécutées dans cette routine d'interruption.

```
ORG 4
bsf PORTB,7 ; on allume la led connectée sur rb7
bcf INTCON,INTF ; on remet à 0 le bit du registre d'IT qui est passé à 1
RETFIE ; retour d'interruption
```

Comme déjà précisé, il ne faut pas oublier de remettre le flag concernant la provenance de l'interruption (ici le bit INTF puisque nous avons choisi de ne travailler que sur une seule source d'IT avec la broche RBO) à zéro, puis, quelque chose de fondamental, le retour d'interruption avec l'instruction "RETFIE" afin que le programme se repositionne à l'endroit du programme principal où a eu lieu l'événement (l'interruption).

Dans les précédentes leçons, nous avons vu que l'adresse courante contenue dans le compteur de programme (PC) au moment de l'interruption est sauvegardée dans la pile

```

;--- Application avec un PIC : Gestion d'une interruption sur RB0 ---
; Titre : Interruption sur RB0
; Date : 12-2004
; Auteur : P.M
; PIC utilisé : PIC 16 F 84
; Ce montage d'initiation à base de PIC 16F84 permet de tester le déroulement
; d'une IT. Lorsque la broche RB0 passe de 0 à 1
; ( front montant ) alors on génère une IT et on allume une led (sur broche RB7)

```



```

;----- Directive d'assemblage pour PLAB ---
list p=16f84A
#include p16f84A.inc
__config H'3FF9'

```

```

;**** Le programme principal commence à l'étiquette init ****
ORG 0
goto init

```

```

;**** Le programme d'IT se déclenche lorsque l'entrée RB0 passe de 0 à 1 ***
ORG 4

```

```

;***** Programme d'interruption *****

```

```

bsf PORTB,7 ; on allume la led connectée sur rb7

bcf INTCON,INTF ; on remet à 0 le bit du registre d'IT qui est passé à 1

RETFIE ; retour d'interruption

```



```

;***** Programme d'INIT *****
init

```

```

bsf STATUS,5 ; on met à 1 le 5ème bit du registre status pour accéder
; à la 2ème page mémoire pour config trisb

```

```

MOVLW B'00000001' ; rb0, en entrée ( rb0 sera la broche utilisée pour l'IT )
MOVWF TRISB

```

```

bcf STATUS,5 ; on remet à 0 le 5ème bit du registre status pour
; accéder à la 1ère page mémoire
bsf OPTION_REG,INTEDG ; Le passage de 0 à 1 sur RB0 provoque une IT sur un
; front montant

```

```

bsf INTCON,INTE ; on autorise l'IT sur RB0
bsf INTCON,GIE ; on autorise les interruptions
clrf PORTB

```

```

;***** Programme principal en rebouclage *****
debut

```

```

GOTO debut ; boucle d'attente impulsion sur RB0
;***** Fin du programme *****
end

```

5 Programme type de gestion d'une IT déclenchée par RB0

puis est restituée grâce à l'instruction RETFIE (Figure 6). Notez bien que seul le compteur de programme est sauvegardé automatiquement, pour sauvegarder par exemple le registre de travail W et le registre STATUS ce qui peut être optionnel dans certains cas, il faudra insérer les instructions correspondantes au début du programme d'IT, puis il faudra restaurer ces informations en fin de sous-programme d'IT. Voici un exemple de sauvegarde de ces deux registres : Nous avons déclaré auparavant deux variables

nommées `w_temp` et `status_temp`. L'utilisation de l'instruction SWAPF permet de n'affecter aucun flag du registre d'état, ainsi on peut sauvegarder ce registre sans le modifier auparavant ce qui paraît évident.

En début de sous-programme d'IT

```

movwf w_temp ; sauvegarde du registre W dans w_temp
swapf STATUS,w ; sauvegarde du registre status dans w

```

```

movwf status_temp ; sauvegarde du registre status dans status_temp

```

instructions du sous-programme d'IT

En fin de sous-programme d'IT avant RETFIE :

```

swapf status_temp,w ; on remet l'ancien status dans w
movwf STATUS ; restauration du registre status
swapf w_temp,f ; Inversion L et H de l'ancien contenu du registre W
swapf w_temp,w ; Restauration du registre de travail W

```

RETFIE

Test de la source d'Interruption

Dans le cas où vous voulez utiliser les quatre sources d'IT (en positionnant à "1" les bits RBIE, INTE, EEIE et TOIE correspondants du registre INTCON) et que vous souhaitez savoir d'où provient l'interruption afin de réaliser des traitements différents, il faudra faire un test des flags d'interruptions (RBIF,INTF,TOIF). Voici un exemple de test de ces indicateurs :

```

ORG 0
Goto init

```

```

ORG 4

```

;Adresse 04 début de sous programme d'IT

```

btfsc INTCON,INTE ; tester si interruption sur RB0 autorisée
btfss INTCON,INTF ; on teste ensuite si flag déclenché (IT en cours)
goto int_timer ; sinon on teste la source d'IT suivante

```

;traitement de l'interruption --- instructions pour IT sur RB0

```

bcf INTCON,INTF ; on efface le flag d'IT sur RB0
RETFIE ; retour au programme principal

```

```

int_timer

```

```

btfsc INTCON,TOIE ; tester si interruption sur TIMER autorisée
btfss INTCON,TOIF ; on teste ensuite si flag déclenché (IT en cours)
goto int_rb47 ; sinon on teste la source d'IT suivante

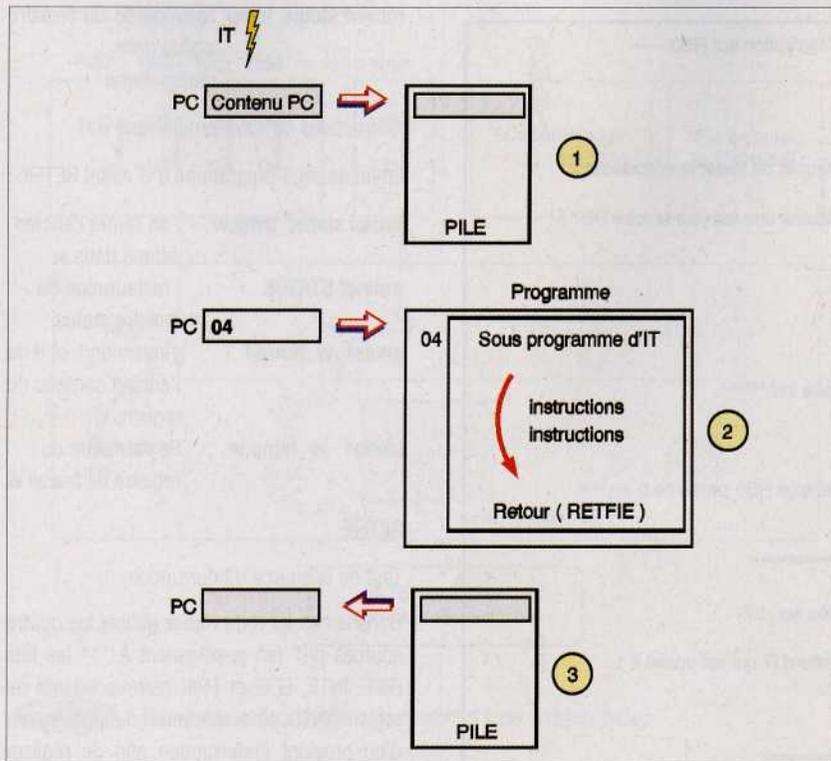
```

;traitement de l'interruption --- instructions pour IT sur TIMER

```

bcf INTCON,TOIF ; on efface le flag d'IT

```



6 Sauvegarde de l'adresse courante

RETFIE ; retour au programme principal

int_rb47

btfsc INTCON,RBIE ; tester si interruption sur RB4-RB7 autorisée

btfss INTCON,RBIF ; on teste ensuite si flag déclenché (IT en cours)

goto int_eprom ; sinon on teste la source d'IT suivante

;traitement de l'interruption --- instructions pour IT sur RB4, RB5, RB6 ou RB7

bcf INTCON,RBIF ; on efface le flag d'IT sur RB4-RB7

RETFIE ; retour au programme principal

int_eprom

;traitement de l'interruption --- instructions pour IT sur fin écriture E_PROM

RETFIE ; retour au programme principal

;Phase d'initialisation

init

bsf INTCON,INTE ; on autorise l'IT sur RB0

bsf INTCON,RBIE ; on autorise l'IT sur RB4-RB7

bsf INTCON,TOIE ; on autorise l'IT sur le TIMER

bsf INTCON,EEIE ; on autorise l'IT sur la fin écriture sur E_PROM

bsf INTCON,GIE ; on autorise les interruptions

debut

goto debut

Pour conclure ce chapitre

Avec cette huitième partie, nous avons abordé les interruptions et leur rôle primordial dans un programme. Si vous avez bien compris cette introduction aux IT, il vous sera facile d'appréhender d'autres microcontrôleurs PIC dans lesquels le principe des IT reste identique.

Vous pouvez télécharger le programme d'IT sur RBO sur le site de l'auteur.

Cette utilisation des interruptions telle que nous venons de la voir est présente dans notre vie courante, lorsque l'on frappe sur les touches d'un clavier d'ordinateur par exemple...

P. MAYEUX

<http://perso.libertysurf.fr/p.may>

ADDITIF

Article " **Serrure à cartes** " (version 2 Septembre 2004 page 76 - EP n° 286)

Précisions sur la connexion de l'afficheur LCD

Il peut y avoir plusieurs types de connexions, sur ce type d'afficheurs :

- en long sur la longueur du module

1.2.3...14

- ou sur le côté du module.

14 13
12 11
.....
2 1

Dans tous les cas le brochage par rapport au numéro de broche est identique et conforme au tableau suivant :

- 1 : GND masse du montage
- 2 : 5 V du montage
- 3 : Réglage contraste - brancher une résistance de 330 ohms à la masse
- 4 : RS - pin 26 du pic
- 5 : RW\ - relier à la masse
- 6 : E - pin 25 du pic
- 7...10 : inutilisé
- 11...14 : datas 4 à 7, pattes 21 à 24 du PIC (respectivement)

Dans le cas où celui-ci est équipé d'un rétroéclairage par led, branchez la patte 15 au 5 V avec une résistance de 100 ohms, et la patte 16 à la masse.

g.samblancat@free.fr