

## LAB 5

Le but du LAB5 : comment compter le nombre d'évènement réaliser par un capteur qui se trouve dans un processus industriel ou n'importe.

Parmi les appareils qui utilisent l'opération du comptage : le fréquencemètre, le tachymètre et quelques décodeurs de claviers qui utilisent les fréquences vocales comme le DTMF (DUAL-TONE MULTI-FREQUENCY)... et dans l'industrie, on utilise cette opération pour réaliser des temporisations et de compter le nombre de pièces passées devant un capteur de proximité...

Donc, après cette petite introduction sur l'utilité de cette opération, on essaye à travers le LAB5 de réaliser cette dernière par notre  $\mu$ C PIC16F84A.

Comment représenter un évènement par un signal électrique ?

Tout simplement, un évènement arrive à partir d'une source ; c'est une impulsion soit positive ou négative comme indique la figure suivante :



Impulsion positive



Impulsion négative

Que remarquez-vous ?

On remarque que les deux impulsions se composent par deux fracs, pour l'impulsion positive le franc montant au début puis un autre franc descendant à la fin, et pour l'impulsion négative, c'est l'inverse.



Impulsion positive



Impulsion négative

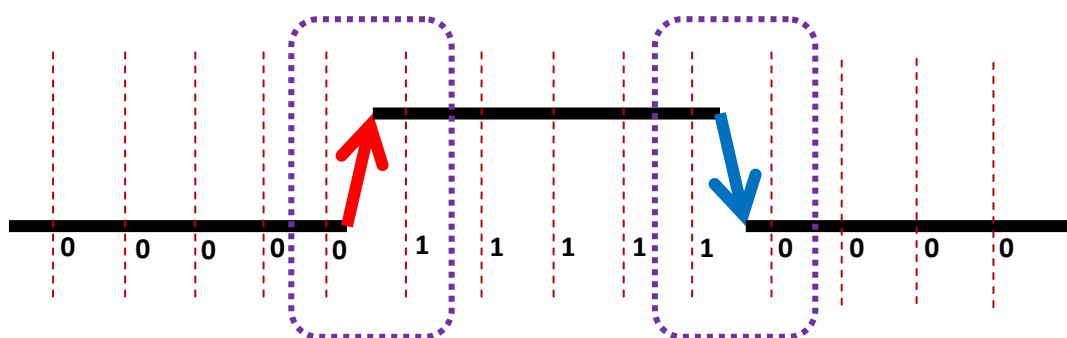
A votre avis mes chers étudiants, comment détecter cette impulsion (notre évènement) ?

La plus part des étudiants qui assistent au cours « les principales périphériques de PIC16F84A » me disent que la patte RB0/INT comporte une logique capable de détecter ce genre de signaux.

Très bien, d'accord, c'est juste !

Mais, comment faire par une autre patte qui ne comporte pas la logique de détection ?

Un ZOOM-IN sur l'impulsion avec échantillonnage :



On remarque clairement que, au moment de détecter un franc montant l'échantillon à l'instant « n-1 » est un « 0 » et l'échantillon à l'instant « n » est un « 1 », et la même histoire avec le franc descendant mais en inverse.

Comment traduire cette remarque en algorithme ?

Pour le franc montant :

```
If (OLD_VALUE=0) and (CURRENT_VALUE=1) then
    // On a détecté un franc montant
End If
```

Pour le franc descendant :

```
If (OLD_VALUE=1) and (CURRENT_VALUE=0) then
    // On a détecté un franc descendant
End If
```

=====

Soit le pseudo-code suivant :

```
// Initialization
OLD_VALUE ← 1;
CNT ← 0;

While (True)
    CURRENT_VALUE ← RA4;
    If (OLD_VALUE=1) and (CURRENT_VALUE=0) then
        CNT ← CNT+1;
    End If
    OLD_VALUE ← CURRENT_VALUE;
    Call DELAY_20ms;
End While
```

Quel type de franc ce pseudo-code arrive à détecter ?

Quelle est la fréquence d'échantillonnage utilisée par ce pseudo-code ?

Qu'est ce qu'il fait le pseudo-code au-dessus ?

D'après l'état initial de « OLD\_VALUE » et la condition dans le « IF », le franc détecté par le pseudo-code est le franc descendant.

La fréquence d'échantillonnage utilisée par le pseudo-code est :  $F_e \approx 1/20\text{ms} = 50\text{Hz}$ .

Le pseudo-code au-dessus fait le comptage des évènements en termes de franc descendant qui arrivent à travers la patte RA4.

=====

Maintenant, on va aller à notre LAB :

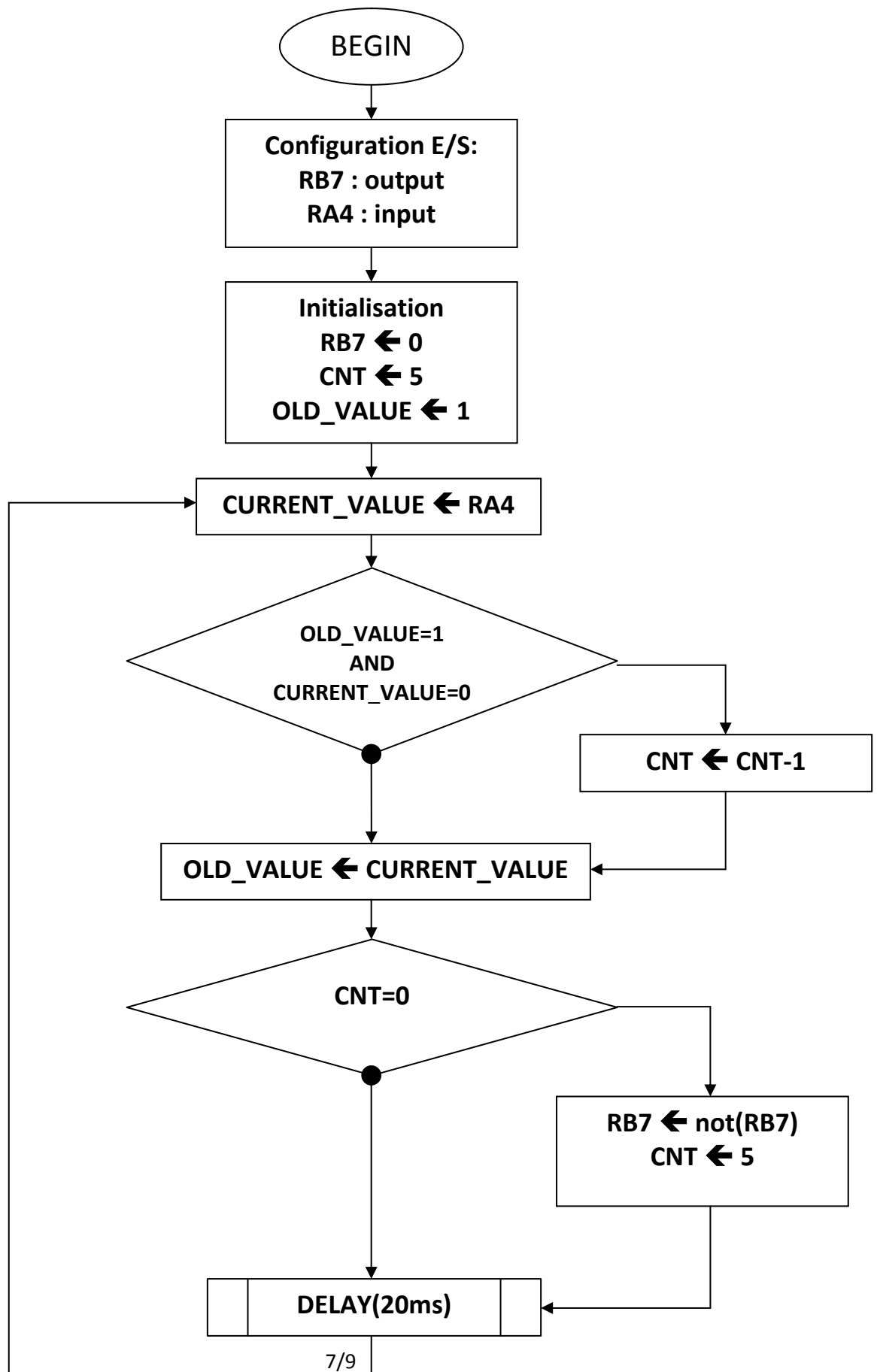
Comment modifier le pseudo-code au-dessus pour réaliser notre LAB5 qui demande de compter 5 impulsions arrive à travers la patte RA4. Si le cas inverser l'état d'une LED relier à RB7.

## Le pseudo-code modifié :

```
// Initialisation
OLD_VALUE ← 1;
CNT ← 5;
RB7 ← 0; // LED (OFF)

While (True)
    CURRENT_VALUE ← RA4;
    If (OLD_VALUE=1) and (CURRENT_VALUE=0) then
        CNT ← CNT-1;
    End If
    OLD_VALUE ← CURRENT_VALUE;
    // Traitement de la LED
    If (CNT=0) then
        RB7 ← not(RB7); // inversion de l'état LED
        CNT ← 5; // Initialisation de CNT pour un autre cycle
    End if
    Call DELAY_20ms;
End While
```

Traduction en organigramme sur  $\mu C$  :



## LAB 5 partie 2

Le même travail mais avec le module TIMER0.

D'après l'étude qu'on a faite sur le module TIMER0, ce module travaille en deux modes, temporisateur et compteur d'évènements.

Dans notre LAB5, ce dernier demande de compter les impulsions qui attaquent la patte RA4, donc on va configurer le module TIMER0 en mode compteur d'évènement par la sélection de la source d'incrémentation externe RA4 sans passé par le pré-diviseur.

Pour ne pas comparer à chaque fois la valeur du registre TMR0 avec 5, on préfère de tester le positionnement du flag TOIF si le TMR0 passe de la valeur 255 vers 0.

Quelle est la valeur qu'il faut charger au début dans le registre TMR0 qui assure le positionnement du flag TOIF après 5 impulsions ?



Tout simplement, c'est le suivant :

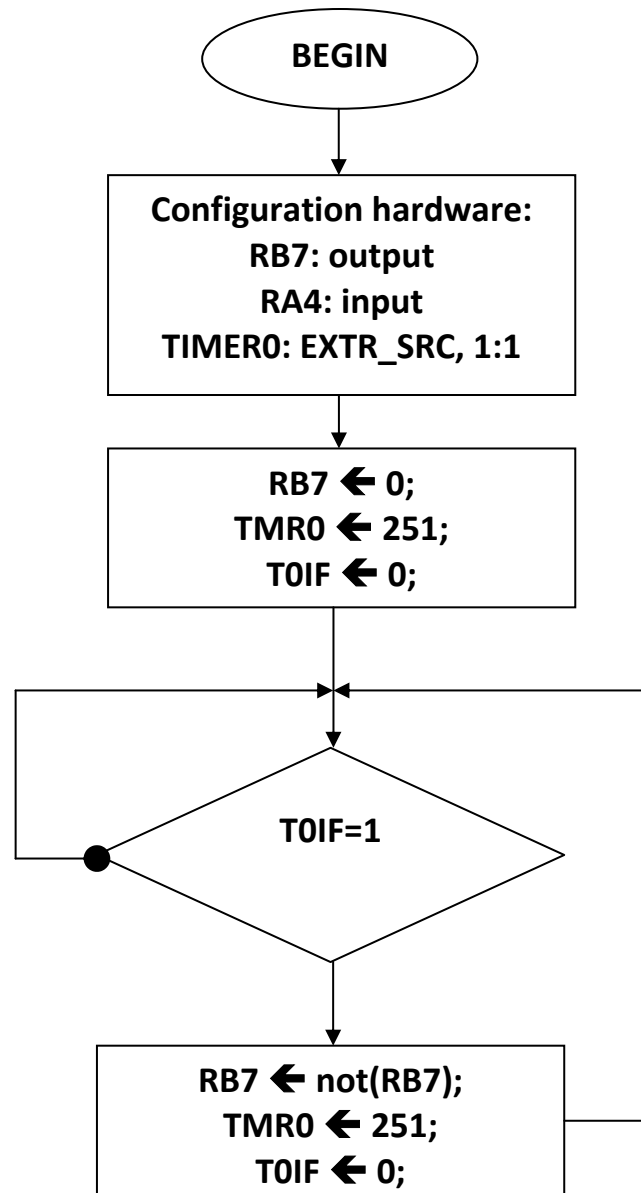
$$N_{\text{impulsions}} = (256 - \text{TMRO\_INIT}) * \text{PRESCALER}$$

$$5 = (256 - \text{TMRO\_INIT}) * 1 \Rightarrow \text{TMRO\_INIT} = 256 - 5 = 251$$

Pseudo-code de la solution sans interruption :

```
// Initialization
RB7 ← 0;
TMRO ← 251;
T0IF ← 0;

While (True)
  If (T0IF=1) then
    RB7 ← not(RB7);
    TMRO ← 251;
    T0IF ← 0;
  End If
End While
```



**Comparer entre la  
solution SOFT et la  
solution TIMER0**

***A vous mes chers étudiants de faire sous  
interruption.***