

## II.1 Introduction:

Les PICs sont des microcontrôleurs fournis par Microchip® à architecture RISC (Reduce Instructions Construction Set), c'est-à-dire leur jeu d'instructions réduit, avec une architecture Harvard. L'avantage est que plus on réduit le nombre d'instructions, plus leur décodage sera rapide ce qui augmente la vitesse de fonctionnement du microcontrôleur.

La famille des PICs est subdivisée en 5 grandes familles:

- La famille Base-Line, qui utilise des mots d'instructions codés sur 12 bits.
- la famille Mid-Range, qui utilise des mots d'instructions codés sur 14 bits.
- la famille High-End, qui utilise des mots d'instruction codés sur 16 bits.
- la famille améliorée (enhanced), qui utilise des mots d'instruction codés sur 24 bits.
- Les dsPICs qui ont été développés pour le traitement de signal et les applications avancées [6].

Il existe en particulier une petite famille appartenant à la famille High-end désignée spécialement au commandement de puissance et les moteurs électriques (DC, AC.....etc.), elle est caractérisée par sa richesse en ressources matérielles (PWM amélioré, interface d'encodeur optique.....etc.), son référence est 18Fxx31.

La table suivante représente un bilan comparatif entre les membres de cette famille:

PIC	FLASH	RAM	EEPROM	E/S	Interface QEI	PWM amélioré	Convertisseur A/N
18F2331	8 KO (4K mots)	768 octet	256 octet	24	oui	6 ch	5
18F2431	16 KO (8K mots)	768 octet	256 octet	24	oui	6 ch	5
18F4331	8 KO (4K mots)	768 octet	256 octet	36	oui	8ch	9
18F4431	16 KO (8K mots)	768 octet	256 octet	36	oui	8ch	9

**Tableau 2.1:** bilan comparatif entre la famille de conversion de puissance.

Nous traitons ici le PIC18F2431

### II.1.1 CARACTERISTIQUES GENERALES DU PIC 18F2431:

- 6 chaînes du PWM amélioré désignées pour la conversion d'énergie.
- Une interface du codeur optique qui capte la vitesse, la position et le sens de rotation.
- Une mémoire programme de type EEPROM flash de 8K mots (16Kbits).
- Une RAM donnée de 768 octets.
- Une mémoire EEPROM de 256 octets.
- trois ports d'entrée sortie, A (8 bits), B (8 bits), C (8 bits).
- 5 entrées du convertisseur analogique/numérique de 10 bits très rapide (200ksps).
- Plusieurs sources d'horloge atteignant jusqu'à 40 MHz.
- Plusieurs modes de l'économie d'énergie (SLEEP, IDLE).
- Deux modules de comparaison / Capture / PWM: CCP1 et CCP2.
- Plusieurs modes de communication: USART amélioré, RS-232.
- 4 Timers principales.
- Un chien de garde étendu.

- 22 sources d'interruption avec ses niveaux de priorité.
- 75 jeux d'instructions.
- Largeur du mot d'instruction est 16bits ,8 bits largeur du bus donnes.
- détecteur programmable du Voltage basse.
- Programmation par mode ICSP (In Circuit Serial Programming) [6].

### II.1.2 Brochage :

Le PIC 18F2431 contient 28 broches, 24 pins d'entrées sorties distribuer en portes : A, B et C

28-Pin SDIP, SOIC

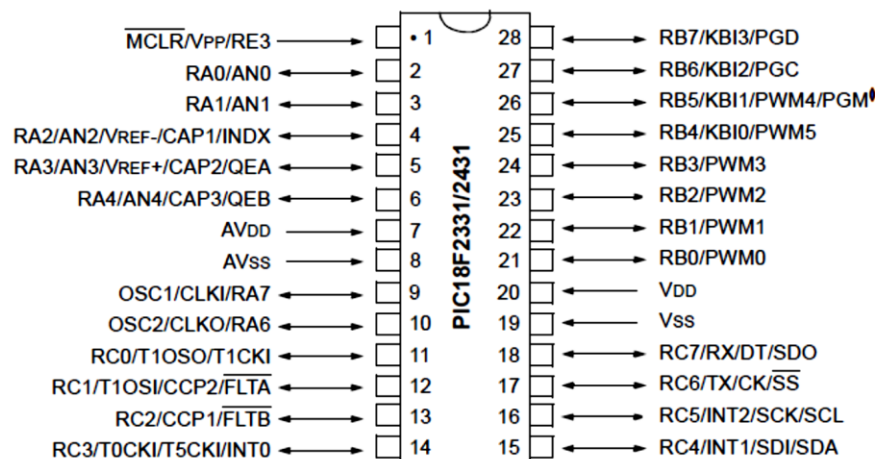


Figure2.1:boitier et brochage du PIC18F2431.

## II.2 Architecture du PIC 18F2431:

### II.2.1 Mémoire

Les mémoires sont des parts indispensables dans les PICs, pour installer le programme à exécuter ainsi les données, ils sont divisés en trois parties:

#### II.2.1.1 Mémoire Programmable

Une mémoire contenant le programme à exécuter par le microcontrôleur. Cette mémoire a la particularité de sauvegarder en permanence les informations qu'elle contient même en absence de tension. Elle est électriquement effaçable, alors on peut la reprogrammer plusieurs fois (100,000 fois). La capacité de cette mémoire est de 8 KO.

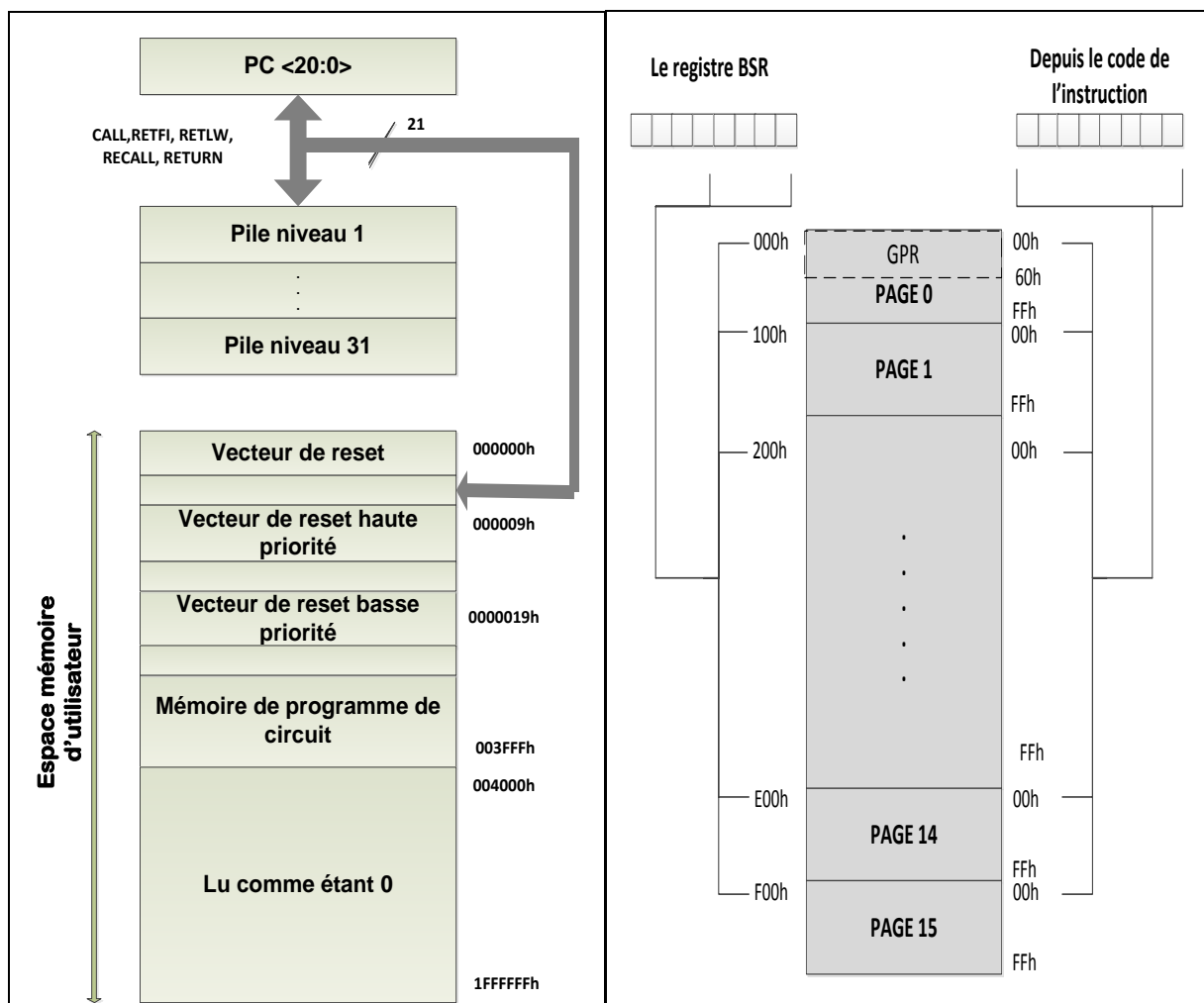
#### II.2.1.2 EEPROM

La mémoire EEPROM (Electrical Erasable Programmable Read Only Memory) est constituée de 256 octets, qu'on peut lire et écrire depuis notre programme. Ces octets sont conservés après une coupure de courant et sont très utiles pour conserver des paramètres semi-permanents. Leur

utilisation implique une procédure spéciale, car ce n'est pas de la RAM, mais bien une ROM de type spéciale. On peut réécrire jusqu'à un million de fois.

### II.2.1.3 RAM

Une mémoire vive également appelée RAM (Random Access Memory). Cette mémoire permet de sauvegarder temporairement des informations. Il est à noter que le contenu d'une RAM n'est sauvegardé que pendant la phase d'alimentation du circuit. La mémoire RAM disponible du PIC 18F2431 est de 368 octets. Elle est organisée en 16 pages, dans la première page (PAGE 0) nous trouvons les registres de contrôle du microcontrôleur (GPR), le reste sont des cases mémoires libres [7].



**Figure 2.2:** organisation du mémoire FLASH et la RAM.

### II.2.2 Les Interruptions:

Une interruption est un événement non excepté provoque l'arrêt du programme principal pour aller exécuter une procédure d'interruption. A la fin de cette procédure, le PIC reprend le programme principal à l'endroit où il l'a laissé.

Les interruptions dans la famille 18F est divisée en 2 catégories: interruptions haute priorité et interruption basse priorité, une interruption haute priorité peut arrêter l'exécution d'une interruption basse priorité (le contraire c'est faux), A chaque interruption sont associés 4 bits, un bit d'autorisation, validation, drapeau et priorité...etc.

Le PIC 18F2431 contient 22 sources d'interruptions, l'adresse des interruptions de haute priorité se trouve dans le vecteur 008, pour les interruptions de basse priorité 018.

Les sources d'interruption dans notre PIC sont:

- Interruption par le module EPWM.
- débordement ou changement de sens à l'interface de l'encodeur optique.
- Entrée d'interruption externe via les pins RCi/INTi.
- Débordement de l'un des TIMERS (TIMER1, TIMER2, TIMER3).
- Changement d'état sur un ou plusieurs pins de PORTB.
- Changement d'état de l'un des comparateurs analogiques.
- Réception ou émission de donnée par USART.
- Fin de conversion analogique numérique.
- Ecriture dans EEPROM est terminée
- Activité du circuit CAPTURE/COMPARE/PWM.
- Activité sur le bus I2C (détection, fin de l'opération).
- Emission ou réception de données sur le port série synchrone (MSSP).

### II.2.2.1 Registres de contrôle:

- **PIE1** (PERIPHERAL INTERRUPT ENABLE) :  
Leur fonction est valider le processus de l'interruption, si le bit associé à une interruption est vaut 0 c'est-à-dire que cette interruption est invalide (masqué).
- **PIRx** (PERIPHERAL INTERRUPT REQUEST):  
Lorsque une interruption est déclenchée le bit associé à cette interruption est mise à 1, on l'appelle le drapeau de l'interruption, la remise à 0 de ce bit se fait par le logiciel.
- **IPRx** (PERIPHERAL INTERRUPT PRIORITY) :  
Chaque bit de ce registre définit la priorité de l'interruption associée, 1 pour interruption prioritaire.
- **INTCON, INTCON2 et INTCON3 (Interrupt Contrôle):** ces registres contrôlent (activation + drapeau + priorité) les interruptions non-périphériques telles que l'interruption externe par pin INTx, Timer 0, changement en port B.

### II.2.2.2 Principe d'interruption:

Selon l'utilisation de la priorité on distingue deux modes principales:

#### II.2.2.2.1 Interruption en mode non hiérarchisé:

Ce mode valide par mise à 0 du bit IPEN du registre RCON, ça veut dire que toutes les interruptions ont la même priorité, le déroulement de ce mode est comme suite:

- le bit de validation général d'interruption GIE du registre INTCON est automatiquement mise à 0 afin d'interdire la prise en compte de toute nouvelle interruption.
- le contenu du compteur ordinal PC est sauvegardé dans la pile.
- le contenu des registres BSR, STATUS et WREG est sauvegardé dans les registres BSRS, STATUSS et WREGS.

- le contenu du compteur ordinal PC chargé par l'adresse 008, ce qui signifie que toutes les interruptions se traitent dans un seul programme.

#### II.2.2.2 Interruption en mode hiérarchisé :

Ce mode prend en compte les niveaux de priorité (haute et basse) de l'interruption qui ont par défaut haute priorité c'est-à-dire ses bits associés au registre IPLx(InterruptPriorityLevel) est mise à 1. Pour mettre en service ce mode de fonctionnement il faut positionner 1 au bit IPEN du registre RCON, ainsi GIEH et/ou GIEL. Le déroulement de ce mode est comme suit :

- Le bit de validation générale d'interruption GIEH ou GIEL est mis à 0 selon le niveau de priorité de l'interruption afin d'interdire la prise en compte des nouvelles interruptions.
- le contenu du compteur ordinal PC est sauvegardé dans la pile.
- le contenu des registres BSR, STATUS et WREG est sauvegardé dans les registres BSRS, STATUSS et WREGS.
- le contenu du compteur ordinal PC chargé par l'adresse 008 s'il s'agit d'une interruption de haute priorité, et 018 s'il s'agit d'une interruption de basse priorité ce qui signifie que les interruptions prioritaires se traitent dans un seul programme(008) et les interruptions de basse priorité se traitent dans un autre programme(018).

Lorsque le CPU arrive à l'instruction **RETFIE** (**RE**TUrnFromIntErrupt), il charge l'adresse où elle s'arrête et les registres précédemment sauvegardés et continue leur exécution normale [7].

#### II.2.2.3 Exemple :

```
void main()
{
    INTCON=0xC0; // activer les interruptions
    PIE3=0xC;    //activer l'interruption du débordement en compteur de position.
} //main

void interrupt() // la sous-routine d'interruption
{
    if(PIR3.IC2QEIF) //débordement du compteur de position
    {
        if(QEICON.UP_DOWN) count ++ ; //si sens avant +1
        if(!QEICON.UP_DOWN) count -- ; //si sens avant -1
    } //interrupt
    PIR3=0; //effacer le flag d'interruption
}
```

### II.3 Convertisseur analogique/Numérique(DAC):

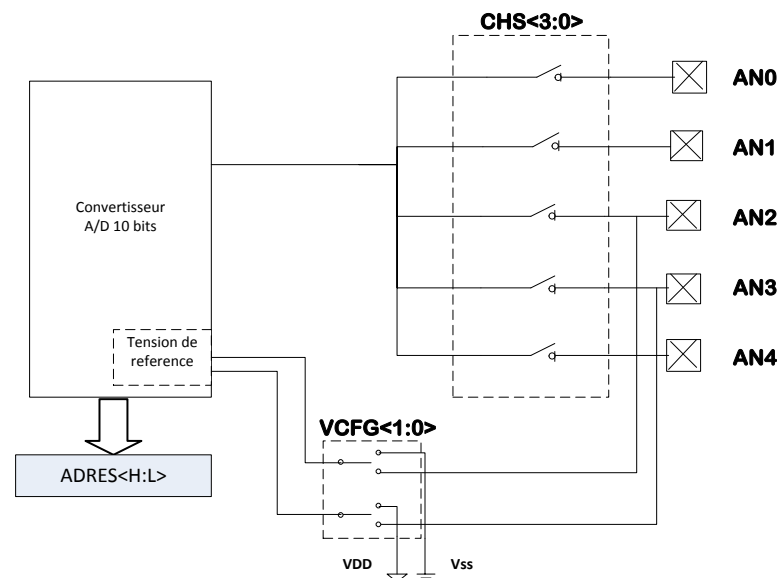
Le convertisseur Analogique-Numérique est un modèle à approximations successives précédé d'un échantillonneur/bloqueur et d'un multiplexeur à plusieurs entrées,

#### II.3.1 Caractéristique du DAC:

- conversion rapide atteint à plus de 200K échantillons par seconde.
- 10 bits de Résolution.
- 2 modules échantillonneurs/bloqueurs.
- Mode d'échantillonnage simultané ou séquentielle optionnelle.
- 4 mots de stockage temporaire(Buffer) de type FIFO.

Selon la configuration choisie au moyen de bits appropriés de son registre de contrôle il peut utiliser diverses source de tension de référence. La tension de référence haute peut être ainsi la tension d'alimentation du PIC ou  $V_{DD}$ , ou bien une tension externe appliqué à l'entrée analogique AN3, tandis que l'entrée analogique basse peut être la masse du PIC  $V_{SS}$ , ou encore appliqué à l'entrée analogique AN2. Toutes les entrées analogiques sont partagées avec les E/S logique.

Le résultat numérique de conversion placer sur le registre ADRESH et ADRESL (Analog Digital RESult), si le buffer est activé on doit sélectionner quel buffer on le met dans le registre ADRES.



**Figure 2.3:** schéma équivalent du convertisseur analogique numérique.

### II.3.2 Les registres de contrôle:

➤ **ADCON0**: contrôle générale du l'ADC

--	--	ACONV	ACSCH	ACMOD1	ACMOD0	GO/DONE	ADON
----	----	-------	-------	--------	--------	---------	------

**ADON**:

Ce bit doit être à 1 pour valider le convertisseur.

**GO/DONE**:

Le fait de mettre 1 a ce bit démarre la conversion, le passage à 0 signifie résultat prêt à lire.

**ACMOD<1:0>**: sélection des chaînes à faire l'auto-conversion séquentielle ou simultanée

Si **ADSCH**=1 (échantillonner plusieurs chaînes):

00: échantillonner groupe A ensuite groupe B (séquentiel).

01: échantillonner les groupes A ensuite B ensuite C ensuite D (séquentiel).

10: échantillonner groupe A et groupe B (simultanément).

11: échantillonner les groupes A, B, C et D simultanément.

Si **ADSCH**=0 (échantillonner un seul chaîne):

00: groupe A ; 01: groupe B ; 10: groupe C ; 11: groupe D ;

*Remarque: les groupes trouvés dans le registre ADCHS.*

**ACONV**: auto conversion bouclé (1) ou bien une seul fois (0).

➤ **ADCON 1**: contrôle générale du l'ADC

VCFG1	VCFG0	—	FIFOEN	BFEMT	BFOVFL	ADPNT1	ADPNT0
-------	-------	---	--------	-------	--------	--------	--------

**VCFG<1:0>**: sélection les tensions de référence de la conversion:

00 = VREF+ = AVDD, VREF- = AVSS.

01 = VREF+ = VREF+ externe, VREF- = AVSS.

10 = VREF+ = AVDD, VREF- = VREF- externe.

11 = VREF+ = VREF- externe, VREF- = VREF- externe.

**FIFOEN**: activer l'utilisation de l'espace du Buffer (1: activé ; 0: désactivé).

**BFEMT**: espace Buffer est vierge (1: vierge ; 0: occupé ;).

**BFOVFL**: Débordement du Buffer

1 = le résultat A/D est écrasé.

0 = le résultat A/D n'est pas écrasé.

**ADPNT<1:0>**: le pointeur Buffer (indique la location prochaine à lire).

00 = Buffer 0 ; 01 = Buffer 1 ; 10 = Buffer 2 ; 11 = Buffer 3.

➤ **ADCON2: contrôle générale du l'ADC**

ADFM	ACQT3	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
------	-------	-------	-------	-------	-------	-------	-------

ADFM: justification de de résultat d'ADC

1 = justifie à droite ; 0 = justifie à gauche.

**ACQT<3:0>**: sélectionner le temps d'acquisition du convertisseur A/D

0000 = démarrage immédiatement lorsque GO/DONE est mise à 1.

0001 = 2 TAD

0010 = 4 TAD

0011 = 6 TAD

...

1111 = 64 TAD

**ADCS<2:0>**:sélection le clock de conversion A/D

000 = FOSC/2

001 = FOSC/8

010 = FOSC/32

011 = FRC/4(2)

100 = FOSC/4

101 = FOSC/16

110 = FOSC/64

111 = FRC (Oscillateur RC interne)

➤ **ADCHS**:sélection les membres (pins) du groupe

GDSEL1	GDSEL0	GBSEL1	GBSEL0	GCSEL1	GCSEL0	GASEL1	GASEL0
--------	--------	--------	--------	--------	--------	--------	--------

**GDSEL<1:0>**: sélectionner les entres du groupe D

00 =AN3; les autres cas réserves.

**GBSEL<1:0>**: sélectionner les entres du groupe B

00 =AN1 les autres cas réserves;

**GCSEL<1:0>**: sélectionner les entres du groupe C

00 =AN2 ; les autres cas réserves.

**GASEL<1:0>**: sélectionner les entres du groupe A

00 =AN0 ; 01 =AN4 ; les autres cas réserves;

➤ **ANSEL0**:sélection des entrées analogiques



—	—	—	ANS4	ANS3	ANS2	ANS1	ANS0
---	---	---	------	------	------	------	------

**ANSx:** chaîne analogique ou entrée/sortie logique

1: chaîne analogique ; 0:entrée/sortie logique

### II.3.3 Exemple:

ADCON0=0x20;

ADCON1=0;// sélectionner AN0 comme entrée à lire.

ADCON2=0x8;//  $V_{ref+}=VDD$ ;  $V_{ref-}=VSS$ , Buffer désactivé.

ANSEL0=1;// clock de conversion = 12 T<sub>cyc</sub>

result = ADRSH\*256+ADRSL;//obtenir le résultat de conversion !

## II.4 Le module PWM amélioré (EPWM):

Le module EPWM (EnhancedPulse Width Modulation), ou bien modulation de largeur d'impulsion améliorée est module intéressant à la commande des moteurs électriques, il est désigné afin de faciliter la tâche de génération du PWM spécialement à la conversion de puissance comme les hacheurs, onduleurs ...etc. On l'utilise généralement dans la commande des moteurs électriques(DC, AC, BLDC ...etc.).

### II.4.1 Caractéristique du module PWM amélioré:

- 6 sorties PWM peuvent être complémentaires ou indépendantes.
- 3 générateurs des rapports cycliques indépendants.
- possibilité de gérer des temps morts entre chaque 2 broches complémentaire.
- Un compteur séparable et flexible avec des différents modes.
- La polarité des broches PWM programmable.
- Commande extérieure par pin (PWM Fault).
- Possibilité de commander manuellement.
- 14 bits résolution du PWM.
- Programmation d'événements pouvant synchroniser une conversion ADC.

### II.4.2 Principe de fonctionnement de PWM:

Même si nombreuses fonctionnalités de la commande ont été ajoutées à la fonction PWM de base, le principe de fonctionnement de mode EPWM est resté identique, Le module contient:

- Un compteur de 12 bits qui se situe dans le registre PTMR, associé à un prescaler et un postcaler.
- 3 registres contiennent les rapports cycliques du PWM: **PDC1, PDC2 et PDC3.**

Le principe de génération du PWM est la comparaison entre ces deux registres: le compteur PTMR et le registre PDCx:

- Si  $PTMR < PDCx$ : la sortie PWM (2x) = 1 (5v), PWM (2x+1) = 0 (0v).
- Si  $PTMR > PDCx$ : la sortie PWM (2x) = 0 (0v), PWM (2x+1) = 1 (5v).

Comme la figure ci-suivant, le rapport cyclique est dépend directement au registre PDCx, la résolution maximale c'est 14 bits c'est-à-dire, pour  $\alpha=100\%$  il faut placer  $2^{14}=0x3FFF$ , et pour 50% on met  $2^7=0x1FFF$

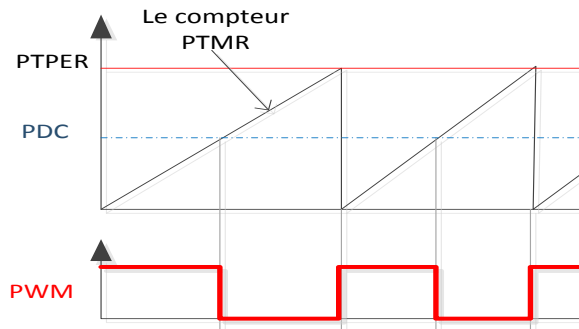


Figure 2.4: principe de génération du PWM.

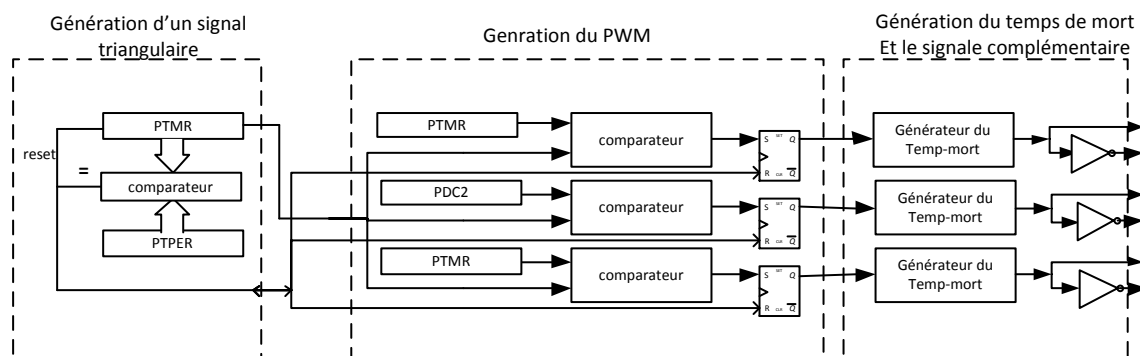


Figure 2.5: modèle équivalent du module EPWM.

### II.4.3 Les modes PWM:

Selon les besoins on peut régler le compteur du PWM:

**II.4.3.1 Mode compteur libre (dent de scie):** c'est le même que le mode PWM classique, le compteur PTMR compte en avant jusqu'il atteint à la valeur de PTPER, et refait pour toujours:

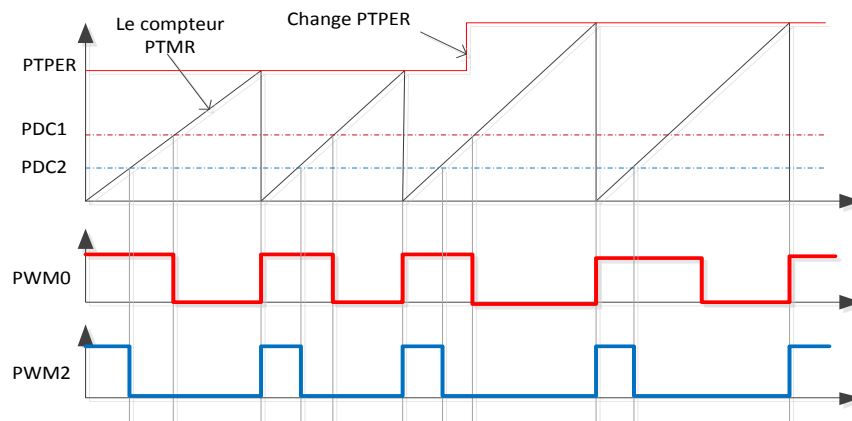
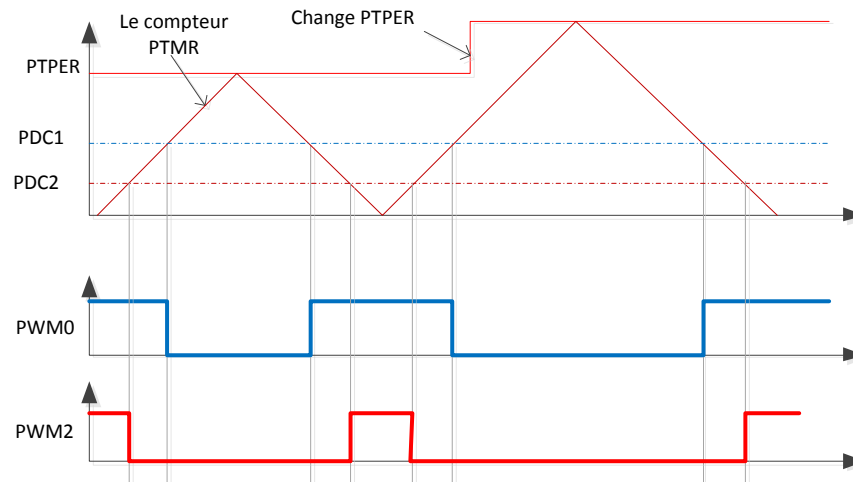


Figure 2.6: PWM en mode libre.

### II.4.3.2 Mode compteur/décompteur (PWM triangulaire):

Dans ce mode le PTMR compte jusqu'à la valeur de PTPER, ensuite il change le sens de comptage c'est-à-dire il décrémente, lorsqu'il arrive à la valeur zéro, et continue comme ainsi de suite:



**Figure 2.7:** PWM en mode compteur/décompteur

### II.4.4 Le Période du PWM:

Le période du PWM en mode compteur libre:

$$T_{PWM} = \frac{(PTPER + 1)}{FOSC (PTMRPS / 4)} \quad (2.1)$$

Le période du PWM en mode compteur /décompteur:

$$T_{PWM} = \frac{(2 * PTPER)}{FOSC (PTMRPS / 4)} \quad (2.2)$$

Ou:

TPWM: le période de PWM.

FOSC: fréquence d'oscillation.

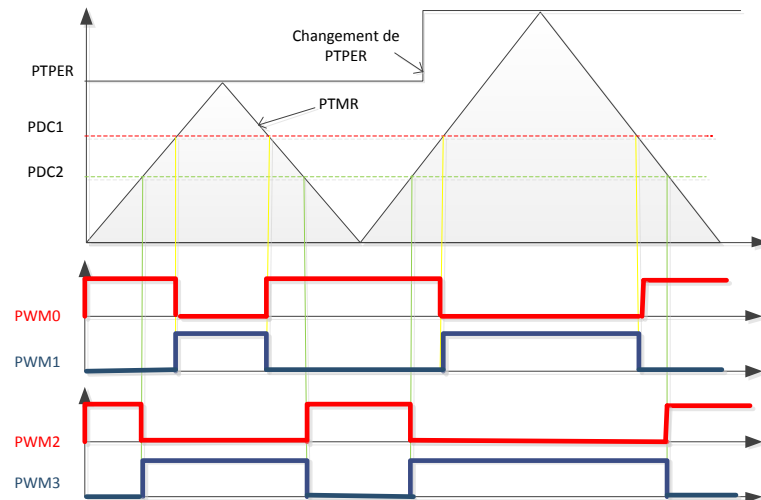
PTPER: la valeur du période du compteur.

PTMRPS: le prescaler du compteur.

### II.4.5 Le mode complémentaire:

La majorité du commande de la conversion de puissance exige deux signaux PWM complémentaire, c'est-à-dire une le deuxième signal PWM est l'inverse du premier. On profiter cette astuce pour faciliter la tâche de contrôler le hacheur, bien qu'il existe 6 sorties de PWM ils ont arrangé en mode complémentaire comme ci-de suite:

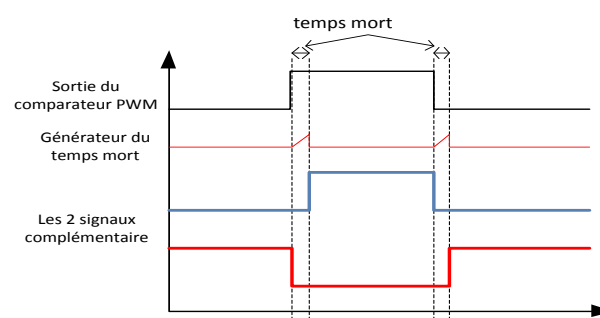
- PWM0  $\leftrightarrow$  PWM1 ;
- PWM2  $\leftrightarrow$  PWM3 ;
- PWM4  $\leftrightarrow$  PWM5 ;



**Figure 2.8:** PWM en mode complémentaire.

#### II.4.6 gestion de temps mort:

Dans les circuits de conversion de puissance qui utilisent le PWM en mode complémentaire (deux signaux inversés), afin de contrôler le pont, il est très important d'insérer un temps mort car les MOSFETs ne sont des interrupteurs idéaux, puisqu'il prend un certain temps pour ouvrir ou fermer son chaîne, pour éviter le court-circuit dû aux caractéristiques ON/OFF du MOSFET, le module EPWM nous permet de programmer un temps mort, c'est un compteur de 6 bits, il démarre un décompte avec le front montant du comparateur lorsqu'il devient nul, il donne la main à mettre 1 à la première sortie (PWMH), lorsque le front descendant du comparateur est arrivé le compteur refait le décompte mais cette fois pour déclencher la deuxième sortie (PWML), comme présenté sur la figure 2.9:



**Figure 2.9:** génération du temps mort.

Il est noté que lorsqu'on active le temps mort, il faut tenir en compte que la valeur maximale du rapport cyclique n'atteint pas à 100%, mais selon le temps de mort choisie.

#### II.4.7 Autre fonctionnalité:

- Le PWM améliorer peut également mettre une synchronisation entre lui-même et le convertisseur analogique numérique.
- Il possède aussi une technique activer ou désactiver la sortie PWM à l'aide d'un pin extérieur appelé PWM FAULT.
- On peut commander manuellement les sorties PWM, on configure le registre OVDCONDSuit on modifiant le registre OVDCONS.
- On peut également activer le PWM en mode libre pour un seul coup, avec une interruption générée [6].

#### II.4.8 Les registres de contrôle:

➤ **PWMCON1**: contrôle le PWM

—	PWMEN2	PWMEN1	PWMEN0	—	PMOD2	PMOD1	PMOD0
---	--------	--------	--------	---	-------	-------	-------

**PWMEN2:PWMEN0**: choisit les sorties PWM

000 = le module PWM désactivé. Tous les E/S PWM sont des E/S logiques.

001 = PWM1.

010 = PWM0 et PWM1.

011 = PWM0, PWM1, PWM2 et PWM3.

100 = PWM0, PWM1, PWM2, PWM3, PWM4 et PWM5.

101 = Tous les E/S PWM sont activés.

110 = PWM1, PWM3.

111 = PWM1, PWM3, PWM5.

**PMODx**: choisit les sorties complémentaires

0: les sorties (PWMx, PWMx+1) sont indépendantes.

1: les sorties (PWMx, PWMx+1) sont complémentaires.

➤ **PTCON0**: contrôle de la base de temps (période)

PTOPS3	PTOPS2	PTOPS1	PTOPS0	PTCKPS1	PTCKPS0	PTMOD1	PTMOD0
--------	--------	--------	--------	---------	---------	--------	--------

**PTOPS3:PTOPS0**: Postscaler du compteur du PWM:

0000 = 1:1 Postscale.

0001 = 1:2 Postscale.

.

.

1111 = 1:16 Postscale.

**PTCKPS1:PTCKPS0:** Prescaler du compteur du PWM

00 = 1:1 prescaler( $F_{osc}/4$ ).

01 = 1:4 prescaler( $F_{osc}/16$ ).

10 = 1:16 prescaler( $F_{osc}/64$ ).

11 = 1:64 prescaler( $F_{osc}/256$ ).

**PTMOD1:PTMOD0:** sélection du mode du PWM

00 = compteur du PWM en mode libre.

01 = compteur du PWM en mode libre un seul coup.

10 = compteur du PWM en mode compteur/ décompteur.

11 = compteur du PWM en mode compteur/ décompteur avec des interruptions.

➤ **PTCON1:** contrôle de la base de temps (période)

PTEN	PTDIR	—	—	—	—	—	—
------	-------	---	---	---	---	---	---

**PTEN:** activation du compteur PWM

1 = compteur est ON.

0 = compteur est OFF.

**PTDIR:** direction du compteur

1 = compteur PWM incrémente .

0 = compteur PWM décrémenté .

### II.4.9 Exemple:

PTCON0 = 0x00 ; //compteur PWM(prescaler,postscaler,mode PWM libre).

PWMCON1 = 0x00 ; // spéciale événement désactivé, mise à jour du rapport cyclique active.

DTCON = 0x4f ; //insérer un temps mort entre les 2 sortie inversé.

PTPERH = 0x0F ; //période compteur de PWM (12bits)

PTPERL = 0xFF ; //(low byte PWM periode);

PWMCON0 = 0x30 ; //activer PWM(0,1,2 et 4) en mode complémentaire.

PTCON1 = 0x80 ; //démarrage de compteur (en indiquons la direction).

// charger les rapports cycliques

PDC0H = 0x3F ; //rapport cyclique du (PWM0 et PWM1)=100%

PDC0L = 0xFF ; // (low byte PWM duty cycle)

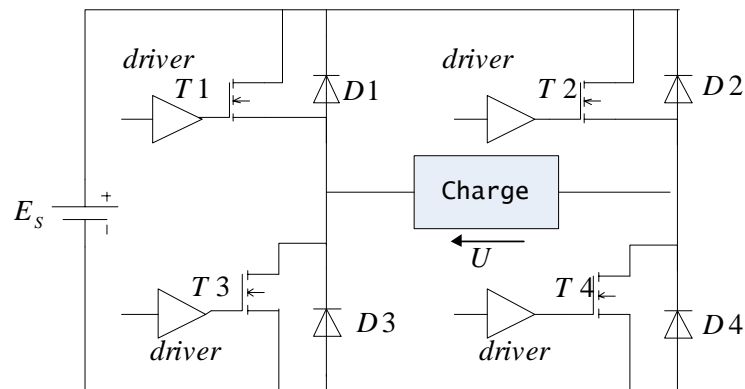
PDC1H = 0x1F ; //second duty cycle(PWM2 et PWM3)=50%

PDC1L = 0xFF ; //

### II.4.10 Application:Hacheur en pont

Le hacheur en pont est un convertisseur DC/DC réglable, il constitue globalement de 4 interrupteurs (mosfet) commandées par le PWM, 2 guident le courant dans le sens direct et les

autres deux guident dans le sens inverse, afin d'éviter la chute de tension les mosfet nécessite un driver de mosfet:



**Figure 2.10:** Schémas de principe du hacheur à quatre quadrants.

Le choix d'un hacheur réversible en courant et en tension permet le fonctionnement dans les quatre quadrants du plan couple/vitesse avec récupération d'énergie dans les phases

Décélération. Il permet aussi de contrôler la tension de sortie par la modification du rapport cyclique  $\alpha$  suivant l'équation suivante [3]:

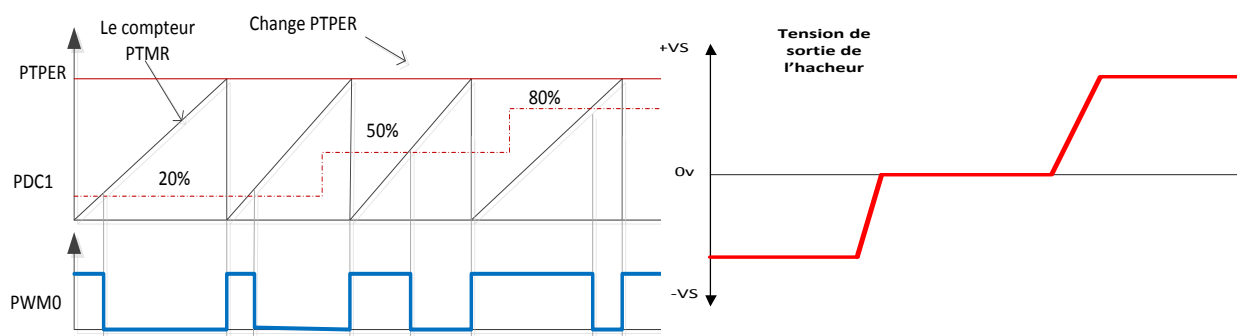
$$\bar{U} = (2\alpha - 1)E_s \quad (2.3)$$

Le modèle du hacheur en pont est donné par l'équation suivante:

$$U = (S_1 - S_2)E_s \quad (2.4)$$

Tel que  $S_1$  c'est l'état logique des interruptions  $T_1$  et  $T_4$  et  $S_2$  c'est l'état logique des

Interruptions  $T_2$  et  $T_3$ .



**Figure 2.11:** caractéristique de tension de sortie de hacheur en fonction du rapport cyclique.

## II.5 Interface du codeur optique(QEI):

C'est un module intégré dans le microcontrôleur désigné spécialement pour lire l'entrée de l'encodeur optique, ceci est très important et efficace pour la commande de la vitesse ou la position.

### II.5.1 Caractéristique du module QEI:

- 3 entrées QEA, QEB et INDX ;
- Compteur de 16 bits(POSCNT) pour calcul de position avec haute précision.
- Détection de sens de rotation avec interruption en cas de changement de direction.
- Filtrage intégré optionnelle.
- Précision de calcul x2 et x4.
- Mesurage de vitesses d'une façon précise.

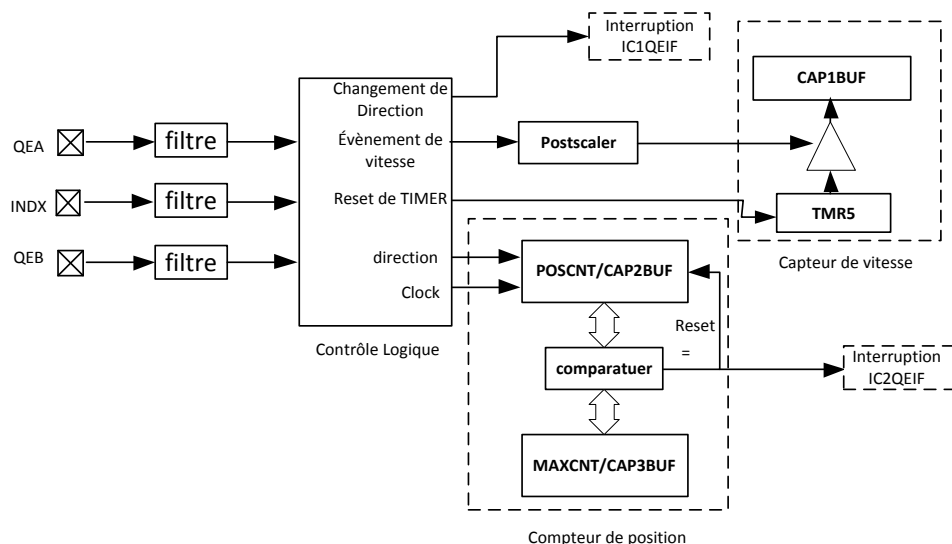


Figure 2.12: schéma équivalent au QEI.

### II.5.2 Constitution de L'interface QEI:

#### II.5.2.1 Module de contrôle logique:

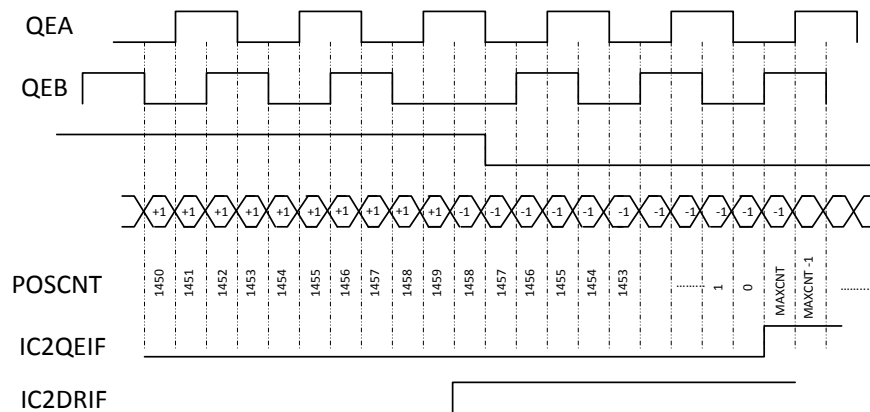
Leur objectif est détecter quel est le front anticipé des entrées QEA ou QEB, en effet il génère le sens de rotation, il génère aussi des impulsions vers le compteur de position (POSCNT), ainsi le signal de déclenche de module capture.

#### II.5.2.2 Module de la position:

Il fonctionne comme un intégrateur des distance traversé, dépend de la mode sélectionné le compteur incrément avec les impulsions de QEA, ou QEA et QEB, la reset de ce compteur se fait ou bien il arrive à la valeur maximal situe dans le registre MAXCNT ou avec l'impulsion qui vin du pin INDX.



Pour le mode x2 le compteur de la position POSCNT augmente avec le front montant et descendant du pin QEA. Pour le mode x4 le compteur de la position POSCNT augmente avec le front montant et descendant des pins QEA et QEB.



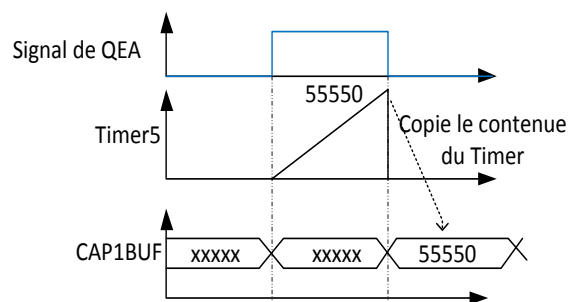
**Figure 2.13:** Timing du compteur de position (précision x4)

### II.5.2.3 Module de la vitesse:

L'idée principale de ce module est calculer la durée d'une impulsion c'est-à-dire la fréquence, leur mode de fonctionnement est le suivant:

Au moment de l'apparition de l'événement déclencheur sur le pin concerné, la valeur du timer5 contenue dans les registres TMR5H et TMR5L est copiée dans les registres CAP1BUFH et CAP1BUFL. On déduit ce cas-là le timer 5 donne une valeur Inversif a la valeur de la vitesse, afin de calculer la vitesse il faut inverser cette valeur

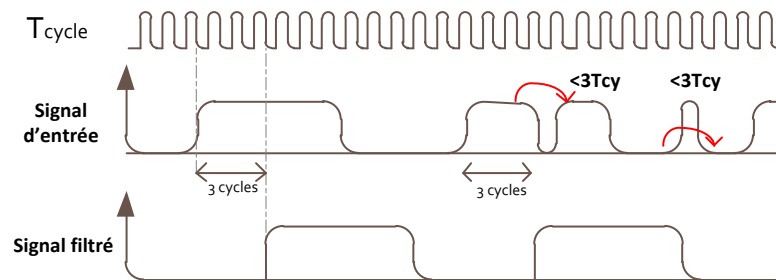
L'évènement selon nous choisis sert le front montant ou descendant, dans le cas où la vitesse est très grande on peut calculer la durée du 1, 4, 16 ou 64 impulsions [8].



**Figure 2.14:** technique de capture la vitesse.

### II.5.3 Le filtre d'entrée:

En réalité, le signal intervenant de l'encodeur optique sont associés à un bruit, du aux moteur et l'interférence des signaux entrant au PIC, ce bruit influence sur les résultats du capteur, pour cela un relecteur de bruit a été placé aux 3 entrées de l'encodeur optique, leur idée de base est tester la largeur de l'impulsion si il est inférieur à  $T_{CYC.FILTRE} = N * T_{CY}$ . Puisque le signal bruit est caractérisé par son fréquence élevée par rapport aux signaux du codeur optique [6]



**Figure 2.15:** filtrage de l'entrée de l'encodeur optique.

## II.5.4 Les registres de contrôle:

### ➤ QEICON:contrôle générale

$\overline{\text{VELM}}$	ERROR	UP / $\overline{\text{DOWN}}$	QEIM2	QEIM1	QEIM0	PDEC1	PDEC0
--------------------------	-------	-------------------------------	-------	-------	-------	-------	-------

**$\overline{\text{VELM}}$** :Activation du mode capture de vitesse

0: capteur vitesse est active.

1: capteur vitesse est active.

**ERROR**: compteur de position

1: compteur de position POSCNT débordé.

0: pas de débordement.

**UP /  $\overline{\text{DOWN}}$** :Direction du moteur

1: sens direct.

0: sens inverse.

**QEIM<2:0>**:sélectionner le mode de comptage de position

000:QEI désactivée.

001: précisionx2,reset avec impulsion de pin INDX.

010:précisionx2, reset quand POSCNT=MAXCNT.

101:précisionx4, reset avec impulsion de pin INDX.

110:précisionx2, reset quand POSCNT=MAXCNT.

**PDEC<1:0>**:réduction du ratio de vitesse

00: mesurer la vitesse chaque une impulsion.

01: mesurer la vitesse chaque 4impulsions.

10: mesurer la vitesse chaque 16 impulsions.

11: mesurer la vitesse chaque 64 impulsions.

### ➤ DFLTCON: registre de contrôle de filtre numérique:

—	FLT4EN	FLT3EN	FLT2EN	FLT41EN	FLTCK1	FLTCK1	FLTCK1
---	--------	--------	--------	---------	--------	--------	--------

**FLT4EN**: activation de filtre au pin T5CKI

1 = activé.

0 = désactivé.

**FLT3EN:**activation de filtre au pin CAP3/QEB

1 = activé.

0 = désactivé.

**FLT2EN:** activation de filtre au pin CAP2/QEA

1 = activé.

0 = désactivé.

**FLT1EN:**activation de filtre au pin CAP1/INDX

1 = activé.

0 = désactivé.

**FLTCK<2:0>:**sélection de l'horloge du filtre

000 =  $1T_{\text{cycles}}^{\text{d'horloge}}$ .

001 =  $2T_{\text{cycles}}^{\text{d'horloge}}$ .

010 =  $4T_{\text{cycles}}^{\text{d'horloge}}$ .

011 =  $16T_{\text{cycles}}^{\text{d'horloge}}$ .

100 =  $32T_{\text{cycles}}^{\text{d'horloge}}$ .

101 =  $64T_{\text{cycles}}^{\text{d'horloge}}$ .

110 =  $128T_{\text{cycles}}^{\text{d'horloge}}$ .

111 =Inutilisable.

**Les autres registres:**

- **VELRH, VELRL (CAP1BUFH:CAP1BUFL):** c'est dont lequel on met la valeur capturé.
- **POSCNTH, POSCNTL (CAP2BUFH:CAP2BUFL):**le compteur de position.
- **MAXCNTL, MAXCNTH (CAP3BUFH:CAP3BUFL):** le période du compteur de position
- **TMR5H, TMRL:** c'est le compteur du mode capture dans le module QEI.
- **PR5H, PR5L:** le période du TIMER5.

### II.5.5 Exemple:

QEICON = 0x28; //(resolution\*2;reset par MAXCNT;mesure vitesse active).

CAP1CON = 0x40; //capture 1 seul impulsion .

T5CON = 0x19; //active le TIMER en mode compteur async. pour capturer le vitesse.

CAP2BUFH =0; //reset position counter

CAP2BUFL =0; //

CAP3BUFH =0xFF; //set MAXCNT, le periode du POSCNT dans valeur maximal.

CAP3BUFL =0xFF; //

while(true)

```
{
    vitesse =    constant1/(VELRH*256+VELRL);
    position=    (POSCNTH*256+POSCNTL)+(count*MAXCNT);
}
```

## II.6 Autre fonctionnalité du PIC:

### II.6.1 Reset:

Un reset est une mise à zéro du microcontrôleur, elle efface le contenu de RAM, et redémarrer l'exécution du programme de début, un reset provoqué par:

- Par action sur la patte  $\overline{\text{MCLR}}$  (memory clear).
- Par le débordement du registre Watchdog.
- Détection d'une chute anormale de la tension d'alimentation.
- Suite à l'instruction « RESET ».
- Saturation de la pile [6].

### II.6.2 Oscillateur:

Dans le PIC 18F2431 il existe plusieurs sources, qu'on peut exploiter comme un oscillateur :

- Cristal/céramique oscillateur via les pins OSC1 et OSC2 (LP, XT, HS, HSPLL).
- Avec un réseau RC externe via le pin RA4(RC), ou RA4 et RA6 (RCIO).
- Oscillateur interne (INTIO).
- Horloge externe via CLKIN CLKO (EC, ECIO) [6].

### II.6.3 Alimentation:

Les tensions qui peuvent être appliquées aux pins VDD et VSS vont: De 4,5V à 6V pour la gamme standard F. mais c'est logiquement de choisir une tension de 5V L'intensité du courant consommé peut aller jusqu'à 25mA.

### II.6.4 Gestion d'énergie:

La conservation d'énergie a une grande importance surtout dans les applications dont leur source est la batterie, La gamme 18F offre une variété des modes de conservation d'énergie:

- Mode SLEEP: ce mode consiste à désactiver la source d'oscillation principale, c'est-à-dire tous les fonctionnalités sont arrêt, ce mode activé par la mise à 0 du bit IDLEN suit l'instruction SLEEP.

- Mode IDLE: ce mode consiste à contrôler sélectivement la source d'oscillateurs des périphériques PIC, ce mode activé par la mise à 1 du bit IDLEN ensuit l'instruction SLEEP [8].

### **II.6.5 Programmation du microcontrôleur:**

Une fois l'algorithme est prêt, on choisit un compilateur, il existe plusieurs compilateurs tel que : MikroC (mikroelectronica©), MPLABC (Microchip©), CCSPCW (computer costume service©)...etc.

Après la compilation on charge le fichier (.hex) à notre PIC à l'aide d'un circuit programmeur.

Dans ce travail on a utilisé MikroC.

### **II.7 Conclusion:**

Le microcontrôleur PIC 18F2431 représente un bon milieu d'implanter notre commande, surtout avec ses caractéristiques précédemment décrits, on peut aller plus loin à la commande des machines électriques.