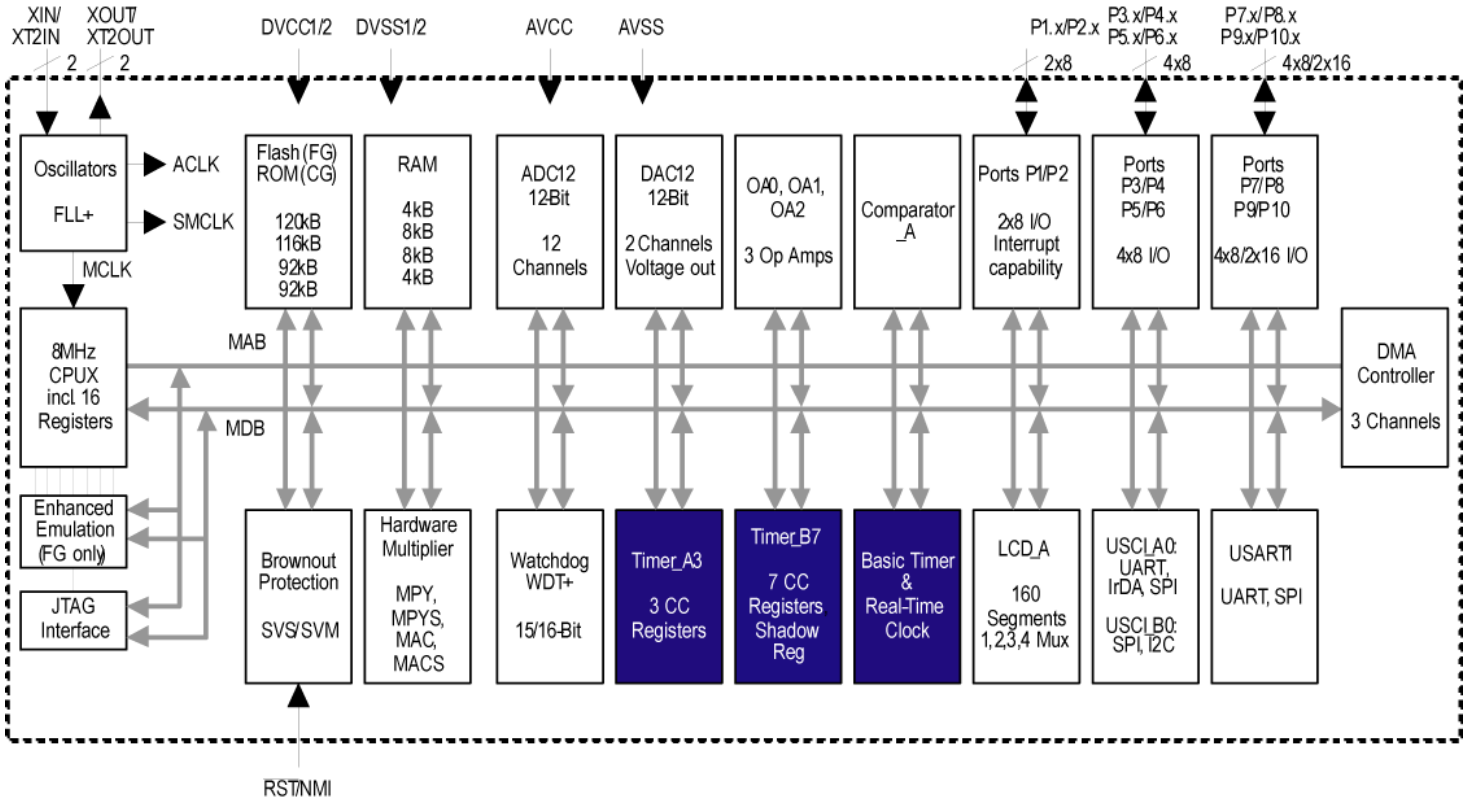
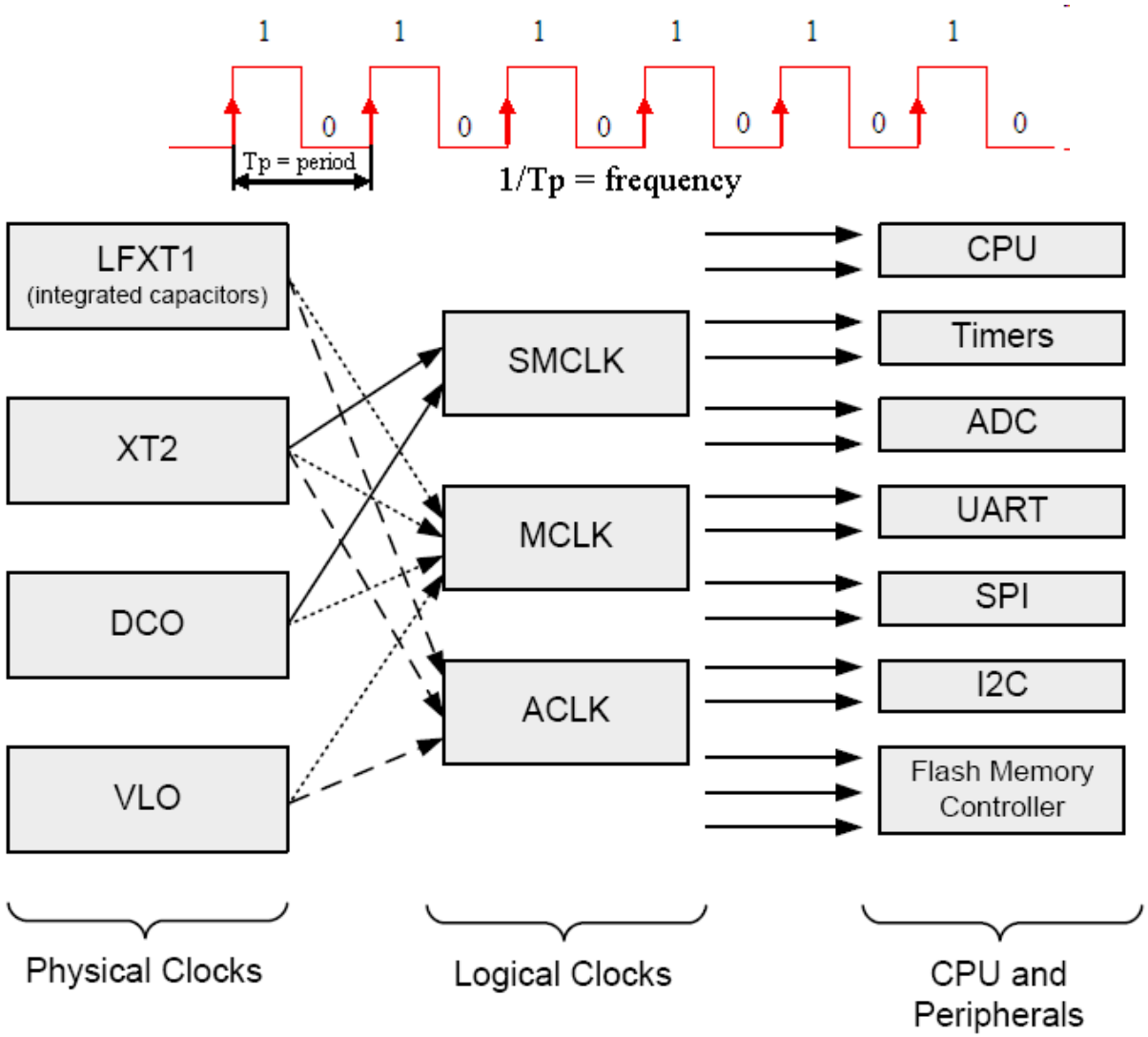
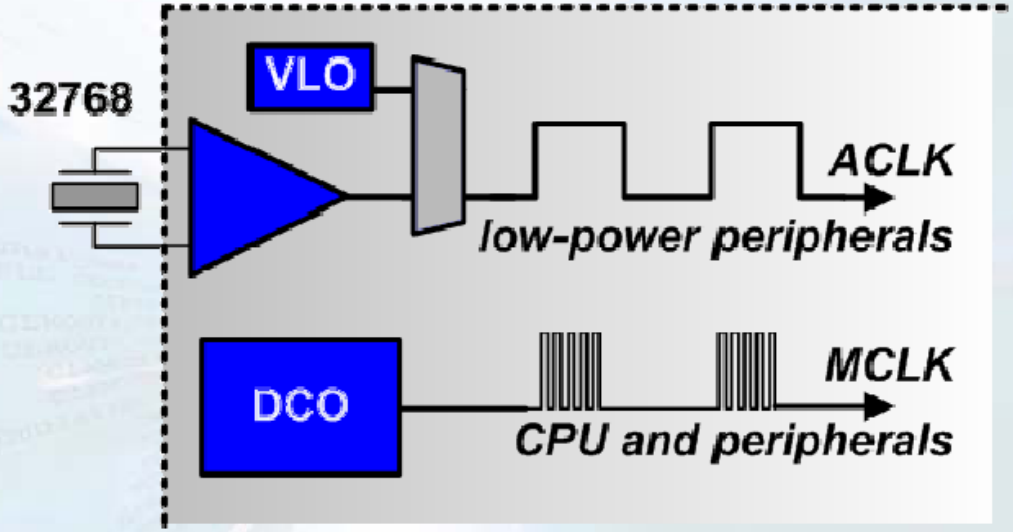
Horloges et Timers



Sources, Horloges et Timers



Ultra-low Power Clock System



- Always-on low-frequency ACLK
- On-demand high-speed DCO
- DCO on and *stable* in <1µs

© 2006 Texas Instruments Inc, Slide 6

VLO

- Very Low (frequency) Oscillator - présent sur le MSP430F2012

internal very low power, low frequency oscillator (VLO)

PARAMETER		TEST CONDITIONS	T_A	VCC	MIN	TYP	MAX	UNIT
f_{VLO}	VLO frequency		-40–85°C	2.2 V/3 V	4	12	20	kHz
			105°C	2.2 V/3 V			22	
df_{VLO}/dT	VLO frequency temperature drift	(see Note 1)	I: -40–85°C T: -40–105°C	2.2 V/3 V		0.5		%/°C
df_{VLO}/dV_{CC}	VLO frequency supply voltage drift	(see Note 2)	25°C	1.8V – 3.6V		4		%/V

DCO

- Digitally Controlled Oscillator

DCO Calibration Data (provided from factory in flash info memory segment A)			
DCO Frequency	Calibration Register	Size	Address
1 MHz	CALBC1_1MHz	byte	010FFh
	CALDCO_1MHz	byte	010FEh
8 MHz	CALBC1_8MHz	byte	010FDh
	CALDCO_8MHz	byte	010FCh
12 MHz	CALBC1_12MHz	byte	010FBh
	CALDCO_12MHz	byte	010FAh
16 MHz	CALBC1_16MHz	byte	010F9h
	CALDCO_16MHz	byte	010F8h

```
BCSCTL1 = CALBC1_16MHZ;          // DCO = 16MHz
DCOCTL = CALDCO_16MHZ;
```

Horloges

ACLK : Auxiliary clock.

- Horloge basse fréquence (économique) sélectionnable en software par les périphériques.

MCLK : Master clock.

- Horloge haute fréquence, utilisée par le CPU et le système d’opération.

SMCLK : Sub-main clock.

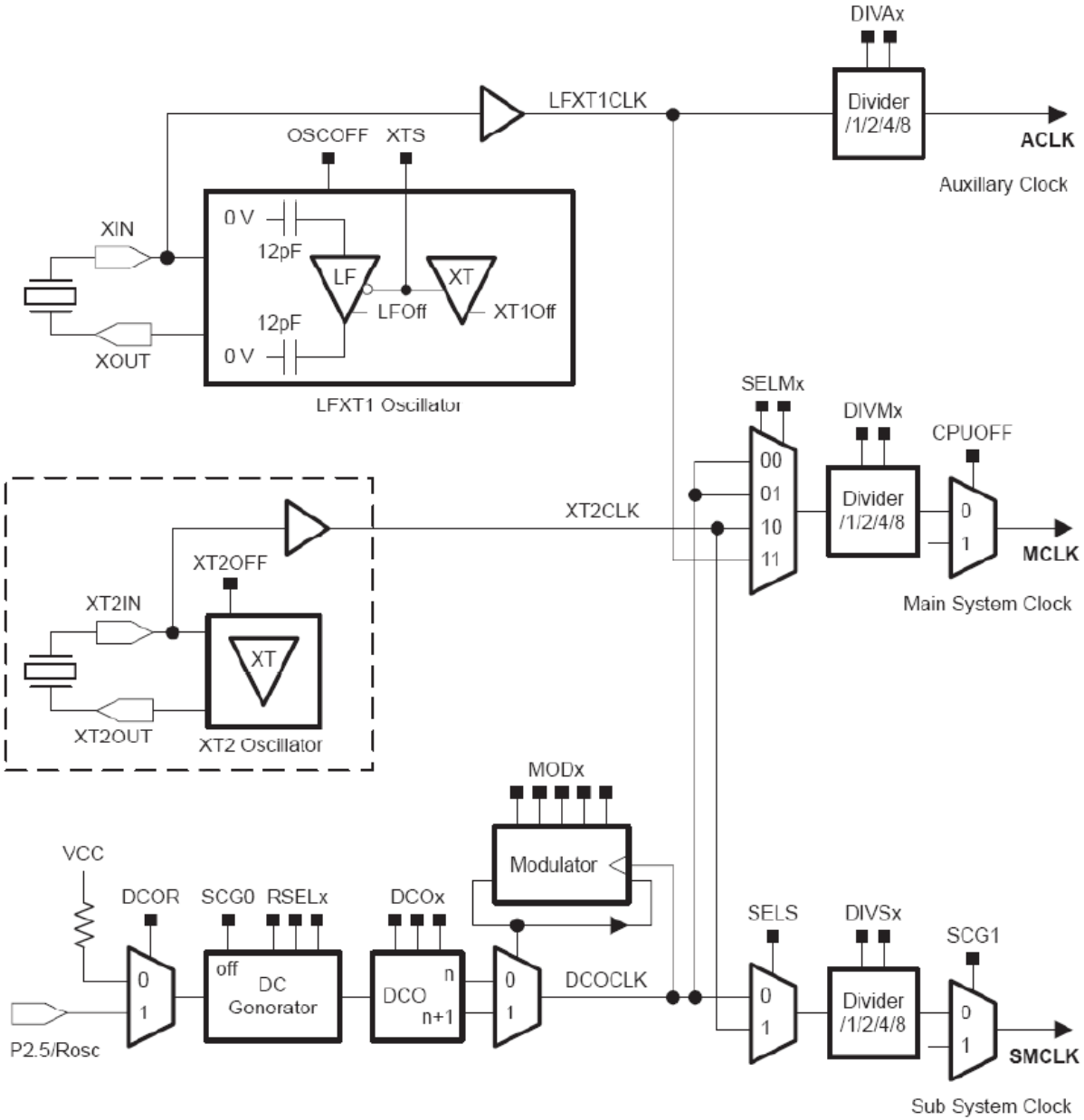
- Horloge haute fréquence, sélectionnable en software pour cadencer les périphériques.

Horloges

- Pour des performances low-power optimales, l'horloge **ACLK** peut être configurée pour fonctionner en général avec un cristal low-power 32’768 Hz (mais seulement environ 12 kHz sur le MSP430F2012), fournissant un temps de base stable pour le système et les modes d'operation low-power.
- L'horloge **MCLK** peut être configurée pour fonctionner avec le l’oscillateur on-chip (DCO) et peut être activée seulement lorsque requis par des interruptions.
- L'horloge **SMCLK** peut être configurée pour fonctionner à partir d'un cristal ou du DCO, en fonction des besoins de périphériques.

- Les horloges ont des détecteurs de mauvais fonctionnement et peuvent émettre une interruption dans ce cas.

Périphérique : les Timers



Les Timers

Les MSP430 de grande taille (ex. FG4617) possèdent plusieurs Timers pour des usages différents :

- Watchdog Timer
- Timer de base
- Timer A
- Timer B

Timer A: Caractéristiques principales

- Timer_A est un timer/compteur de 16 bits avec des registres de capture et de comparaison, présent en général sur tous les types de MSP430.
- Timer_A supporte multiples captures / comparaisons, sortie PWM, et compteur d'intervalles.
- Timer_A a également des possibilités étendues d'interruption.
- Des interruptions peuvent être générées par des dépassement du compteur ou par chacun des registres de comparaison et de capture.

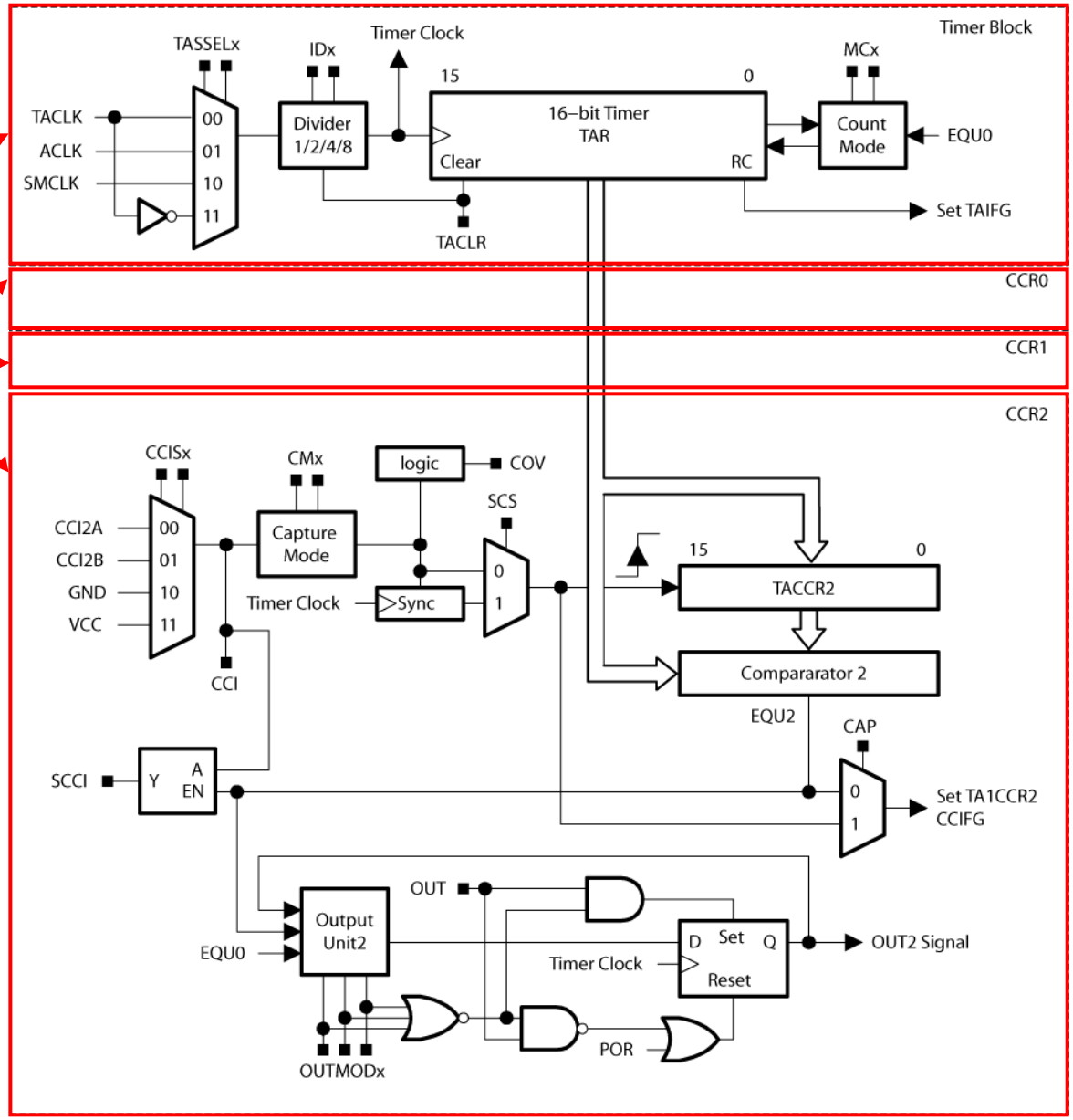
Les dispositifs de Timer_A incluent :

- ⇨ Timer / compteur de 16 bits asynchrone avec quatre modes opératoire.
- ⇨ Base de temps (horloge) sélectionnable et configurable.
- ⇨ Trois ou cinq registres de comparaisons et de captures configurables.
- ⇨ Sorties configurables avec des possibilités de modulation de largeurs d'impulsions (PWM).
- ⇨ Verrouillage asynchrones des entrée et des sorties.
- ⇨ Vecteur d'interruption (registre) pour un décodage rapide des interruptions du Timer_A.

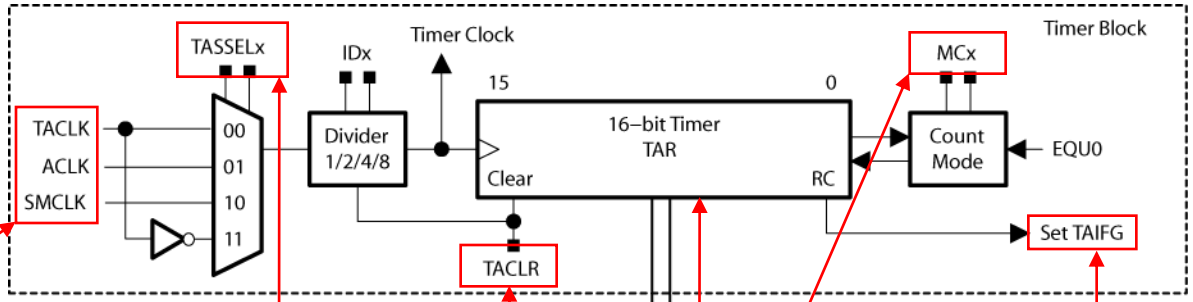
Timer A : Caractéristiques principales

Fonction Timer « simple »

3 modules de capture et de comparaison (2 sur le F2012)



Timer A : Caractéristiques principales



Mise à zéro du timer

Flag indiquant une interruption due à un dépassement

Sélection de la base de temps

Source de la base de temps (horloge de référence)

Timer / compteur avec incrémentation ou décrémentation (selon le mode sélectionné) à chaque flanc montant de l'horloge Timer Clock
Peut être lu ou écrit.
En mode timer peut générer une interruption en cas de dépassement

00	STOP
01	Comptage positif jusqu'à la valeur dans TACCR0
10	Continu, comptage jusqu'au maximum 0xFFFF
11	Comptage et décomptage entre 0 et la valeur dans TACCR0

Timer A : Caractéristiques principales

Défini le mode capture ou le mode comparaison

Indique si un nouvel événement a eu lieu avant lecture du précédent

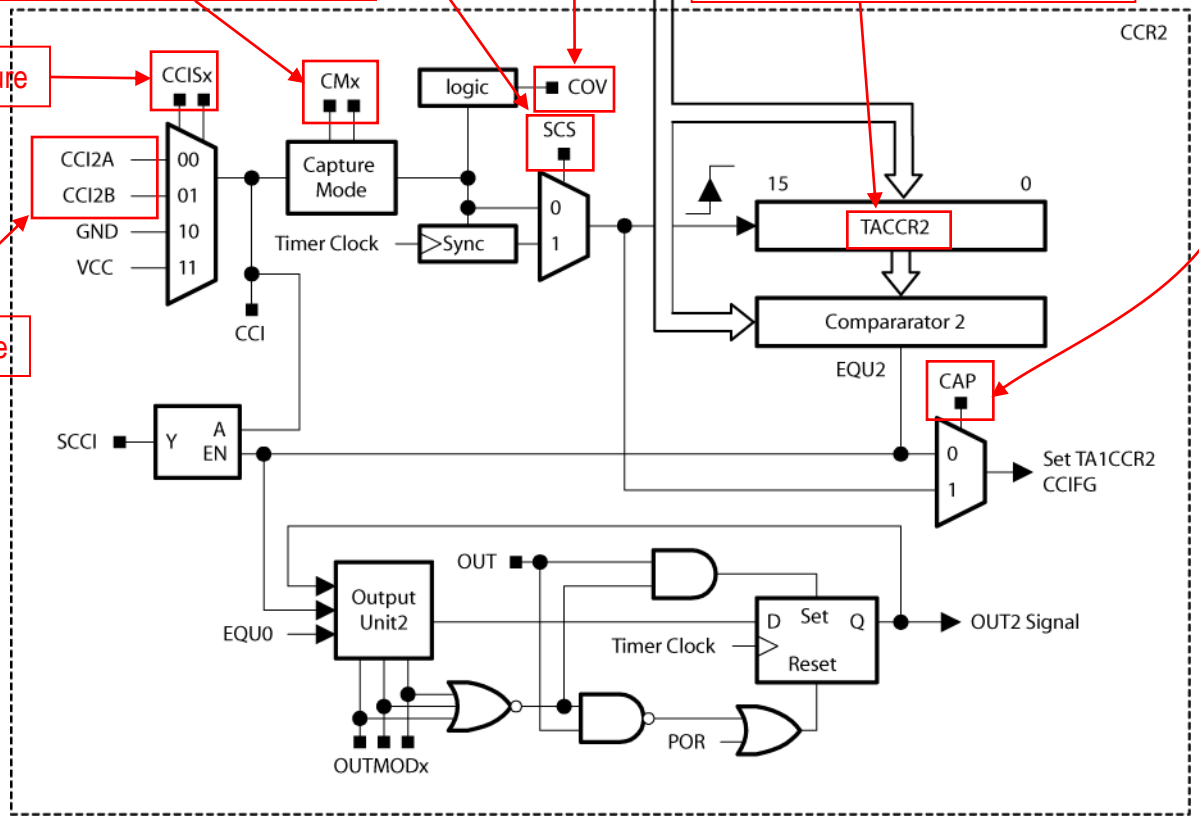
Synchronisation de l'événement avec l'horloge du timer

Mémorisation de la valeur du timer lorsqu'un événement apparaît

Type d'évènement (flanc montant , descendant ou les deux)

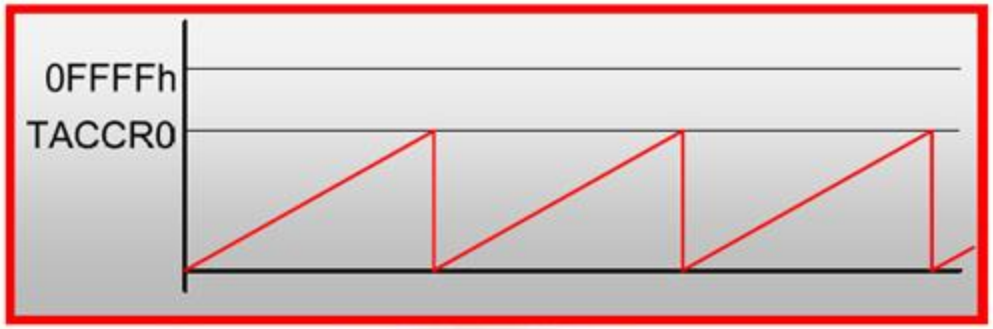
Sélection du signal de capture

Signaux d'entrée pour capture



Timer A: Count Modes

Up

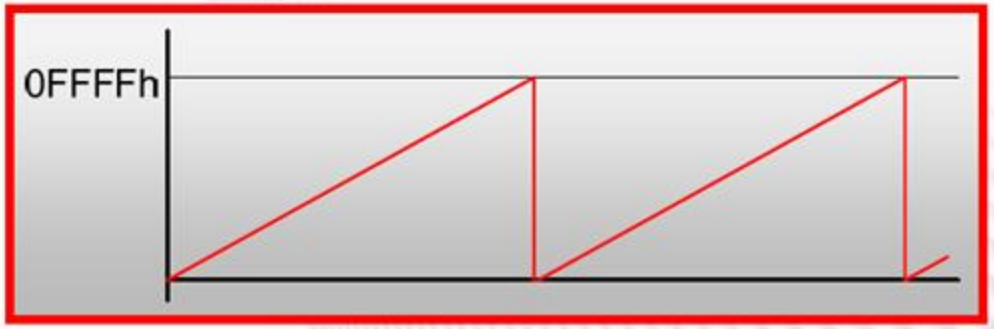


Up to value specified by CCR0, rolls over to 0000, back up to CCR0 value, etc.

Up/ Down

Up to value in CCR0, count down to 0000, back up to value in CCR0, etc.

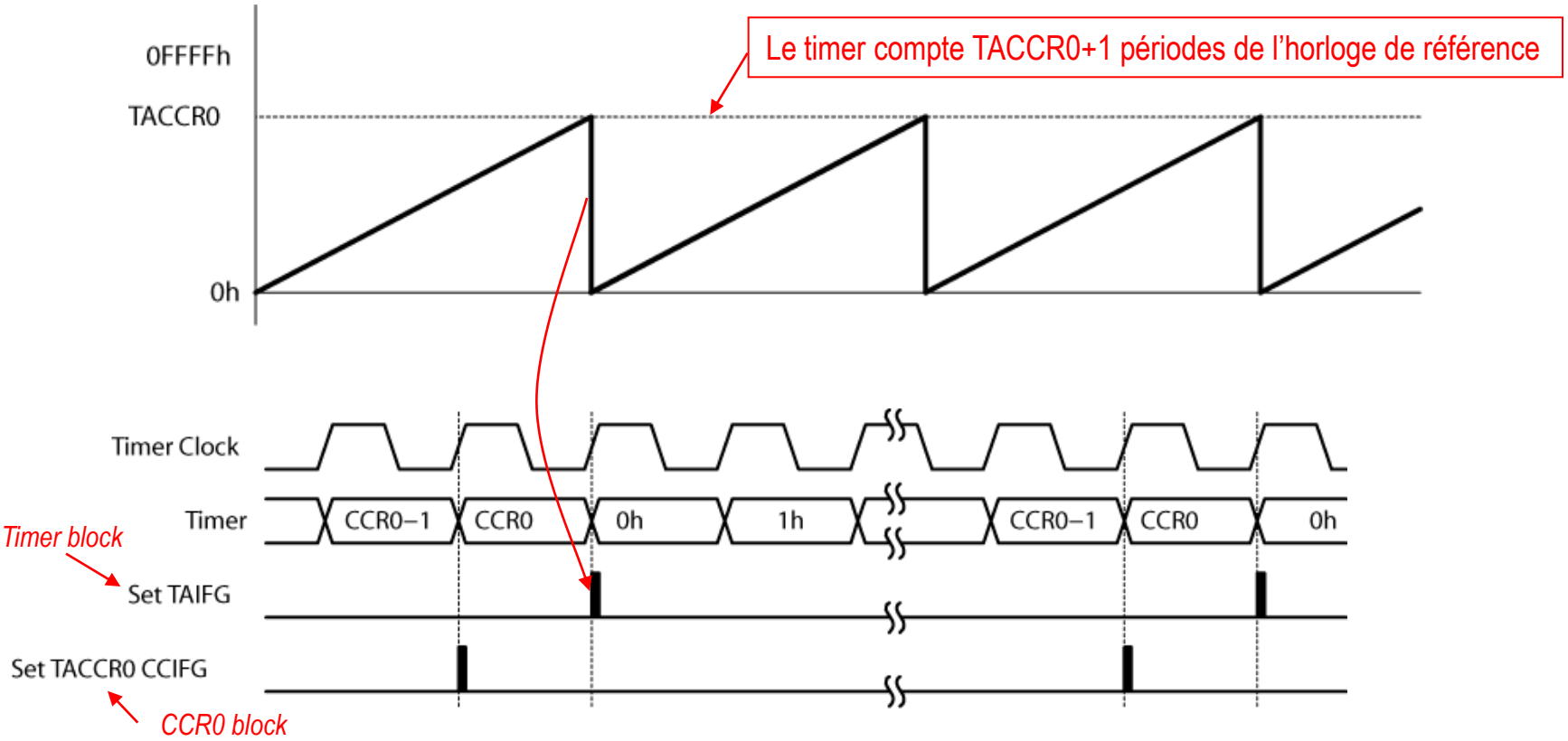
Continuous



Up to FFFF, rolls over to 0000, back up to FFFF, etc.

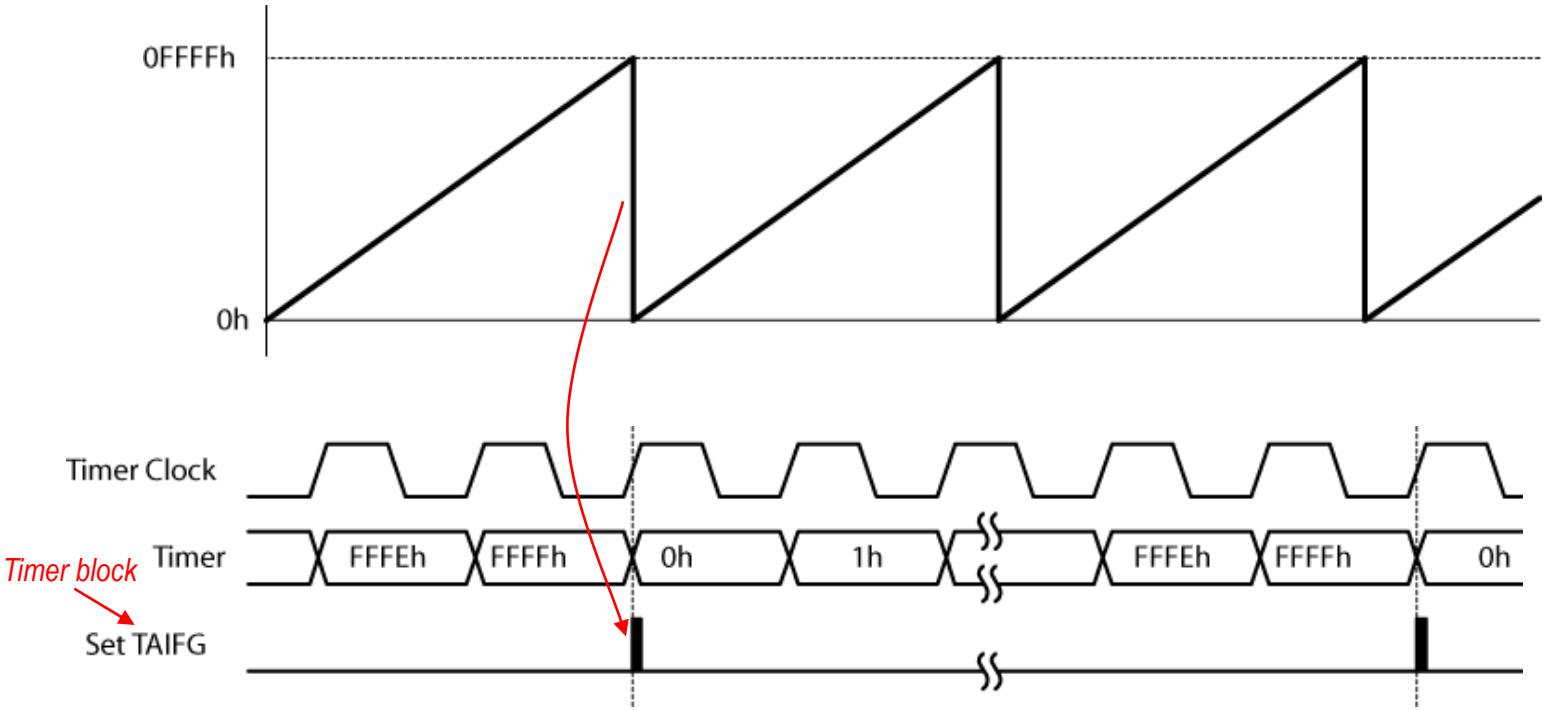
Timer A : Mode de comptage (UP Mode) : MCx = 01

Le timer compte jusqu'à la valeur mémorisée dans le registre TACCR0 puis revient à 0x0



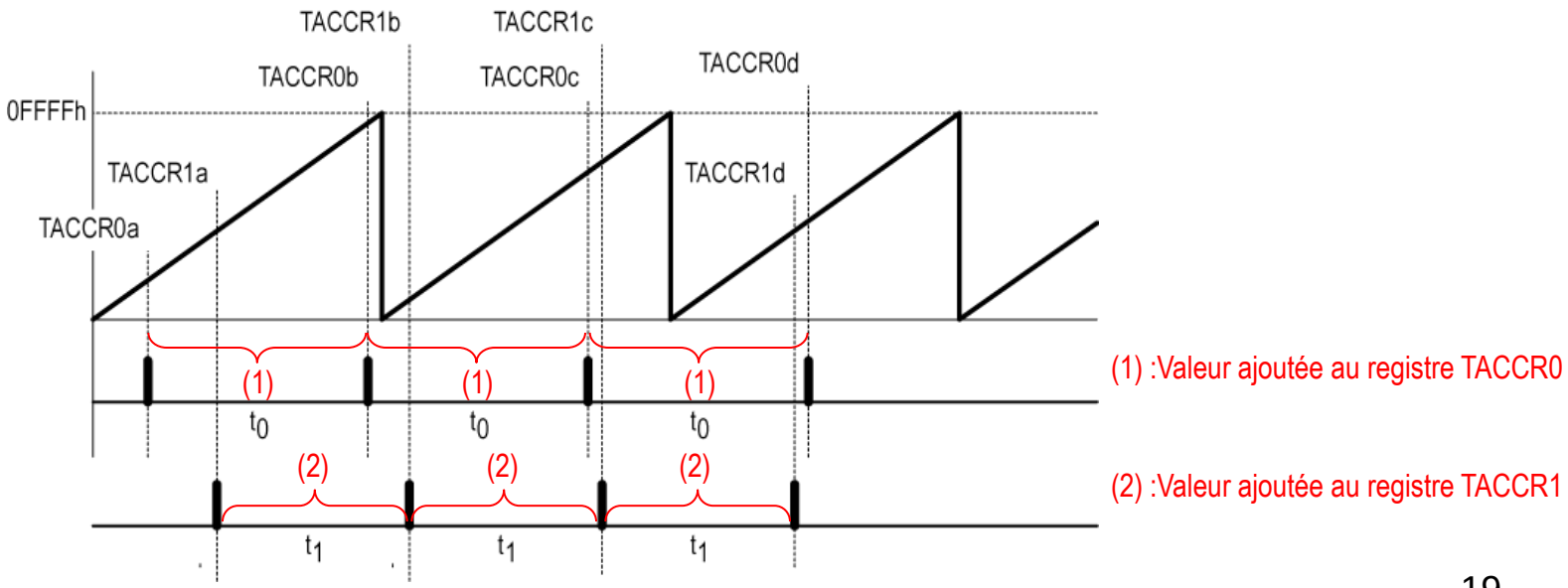
Timer A : Mode de comptage continu (Continuous Mode) : MCx = 10

Le timer compte jusqu'à la valeur maximum (0xFFFF) puis revient à 0x0



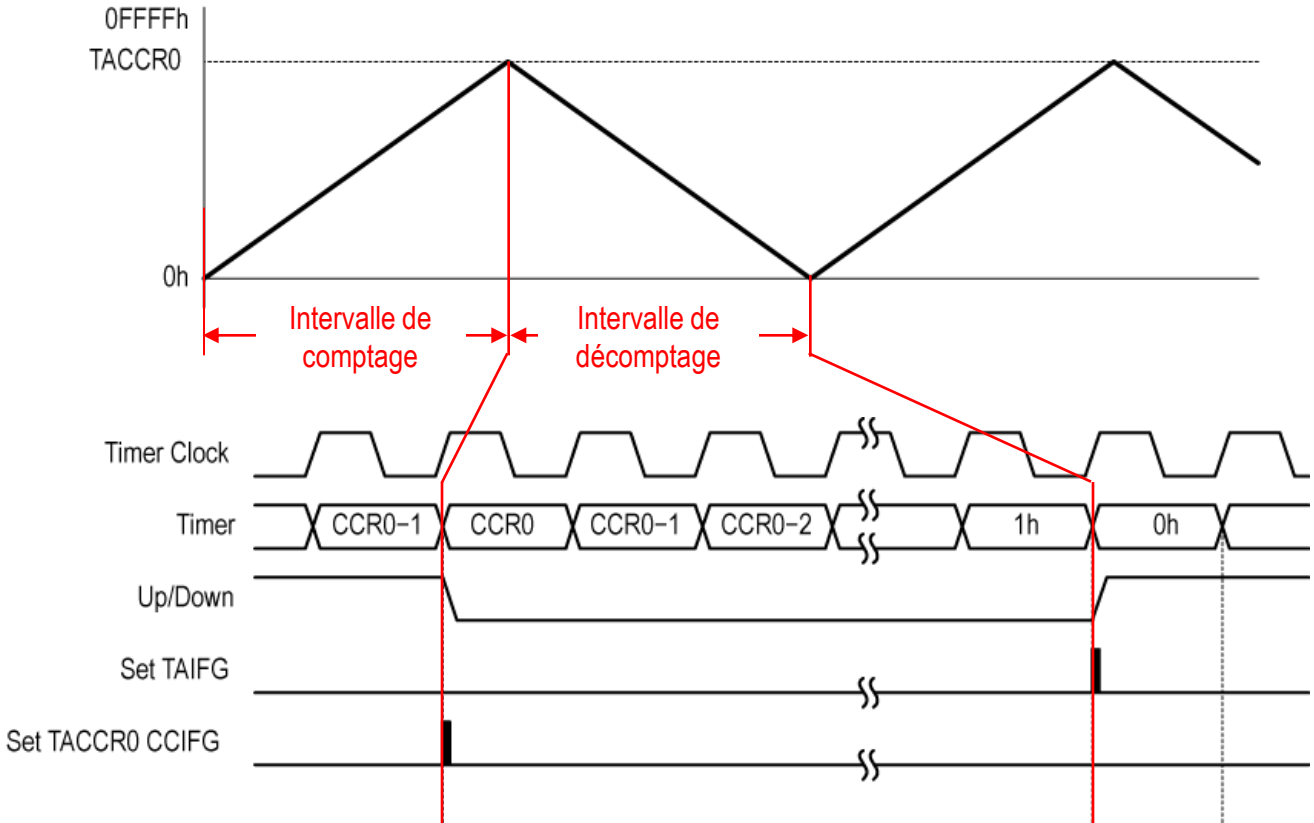
Timer A : Mode de comptage continu (Continuous Mode) : MCx = 10

- Le mode continu peut être employé pour générer des intervalles de temps.
- Chaque fois qu'un intervalle est terminé, une interruption est activée.
- Le prochain intervalle est ajouté au TACCRx dans la routine d'interruption.
- Jusqu'à trois (Timer_A3) intervalles de temps peuvent être générés en utilisant les registres de comparaison / capture TACCR0 à TACCR2.
- Une éventuelle gigue sur la latence d'exécution de la routine d'interruption n'a aucun impact sur la valeur temporelle de l'intervalle.



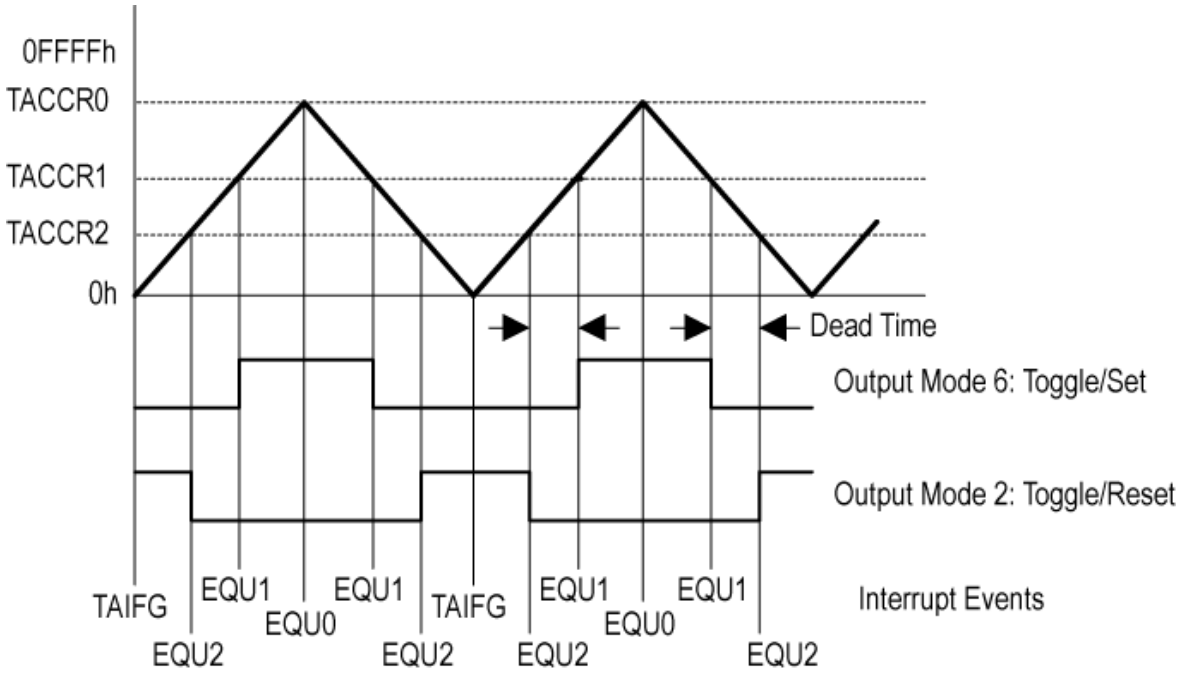
Timer A : Mode Up / Down : MCx = 11

- Le mode compteur / décompteur est employé si la période du timer doit être différente de 0FFFFh et si la génération symétrique d'impulsions est nécessaire.
- Le timer compte jusqu'à la valeur de chargée dans le registre de comparaison / capture TACCR0 puis décompte jusqu'à 0x0.
- La période est deux fois la valeur dans TACCR0.



Timer A : Utilisation du mode Up / Down : MCx = 11

Le mode up/down permet de réaliser une commande de branche d'un pont en H ou d'un onduleur avec contrôle du temps d'antichevauchement, c'est-à-dire un temps pendant lequel aucun des deux signaux n'est actif.

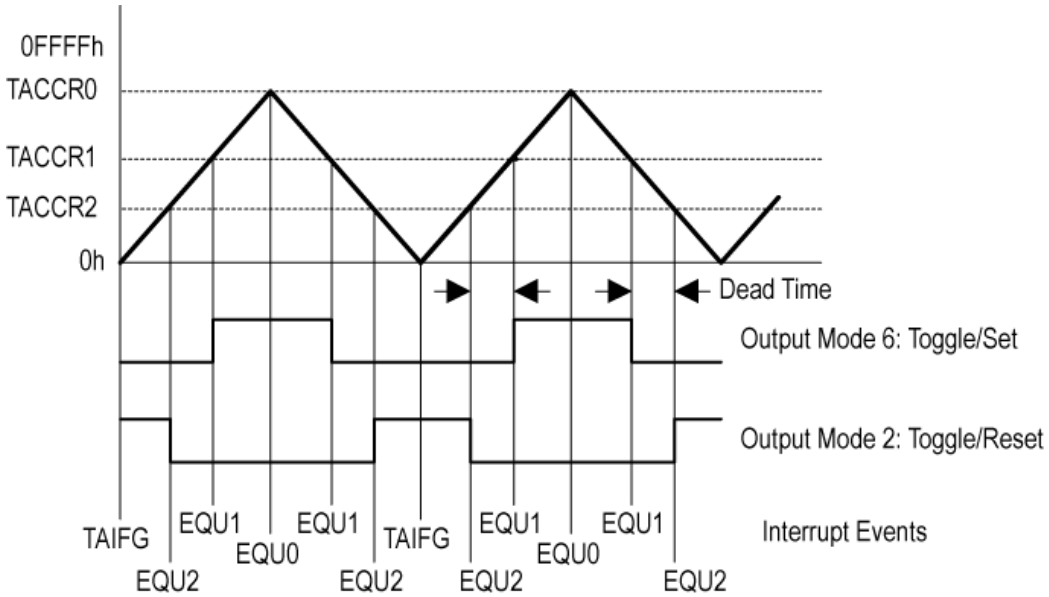


Timer A : Utilisation du mode Up / Down : MCx = 11

Le temps d'antichevauchement, c'est-à-dire le temps pour lequel les deux signaux de sortie sont à l'état bas peut être défini comme :

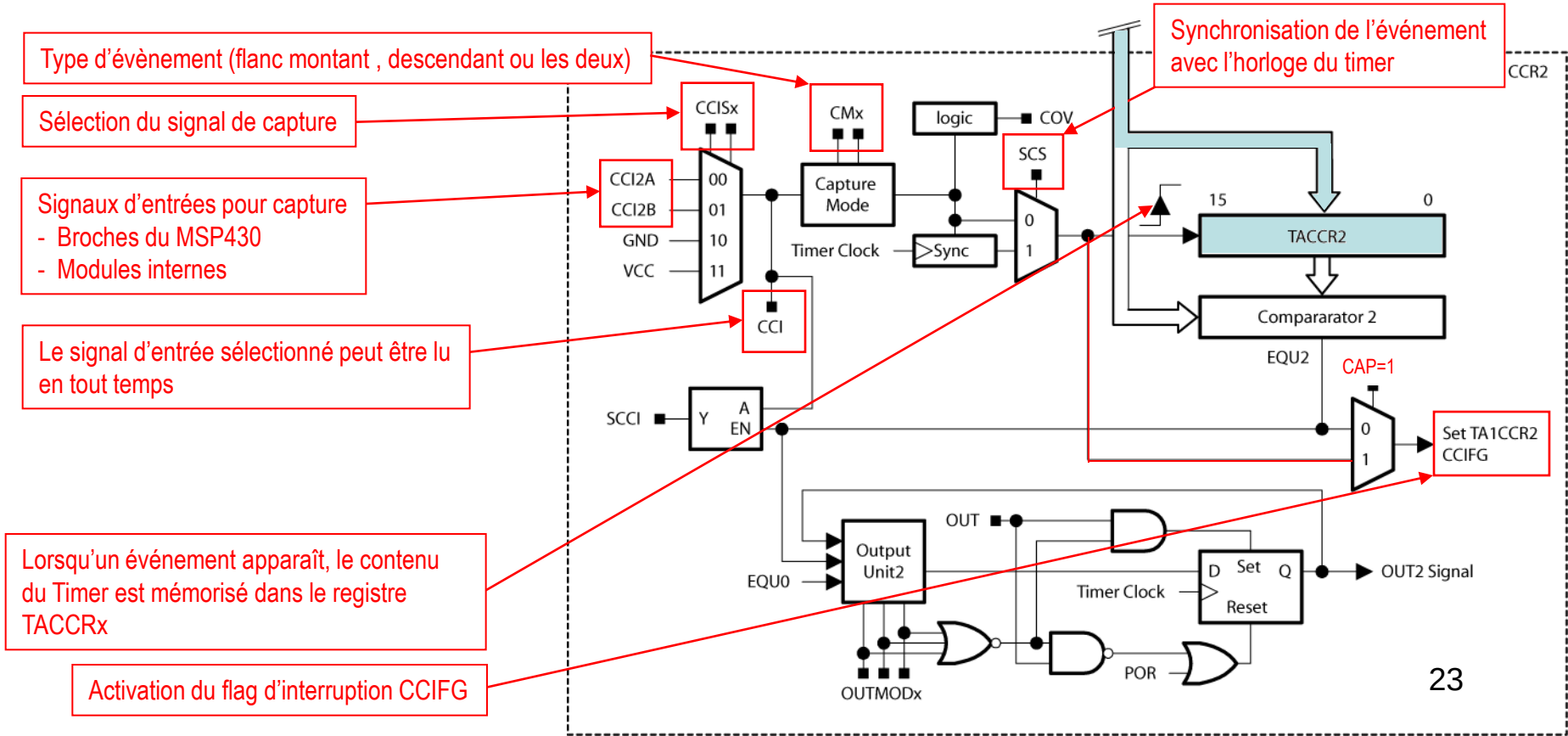
$$t_{dead} = h_{clock} \times (TACCR1 - TACCR2)$$

Les registres TACCRx ne sont pas bufférisé. La mise à jour est faite directement lors de L'écriture. Le maintient du temps mort n'est donc pas automatique.



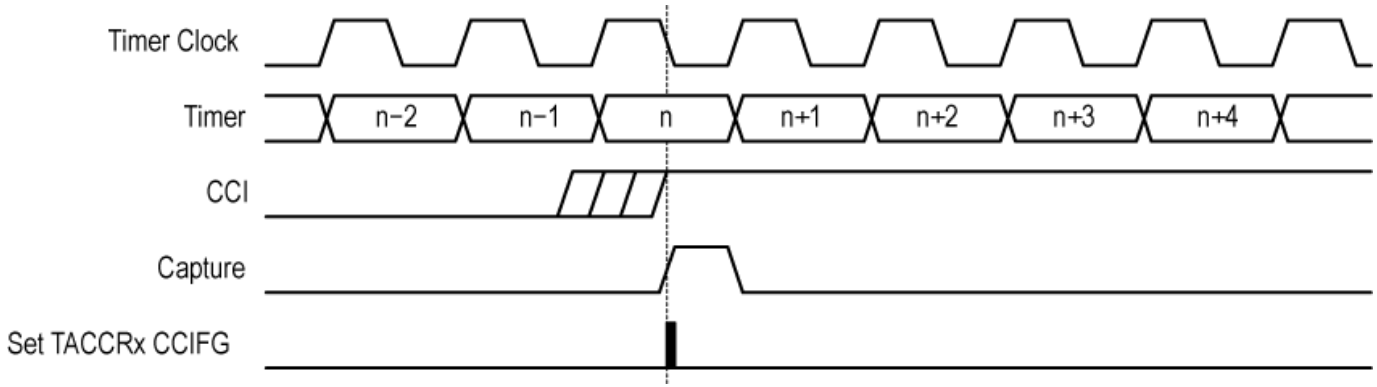
Timer A : Mode capture : CAP=1

Le mode capture est sélectionné lorsque CAP=1. Il permet d'enregistrer des événements temporels comme par exemple une position (en vue d'une mesure de vitesse) ou permet de réaliser une mesure de temps.

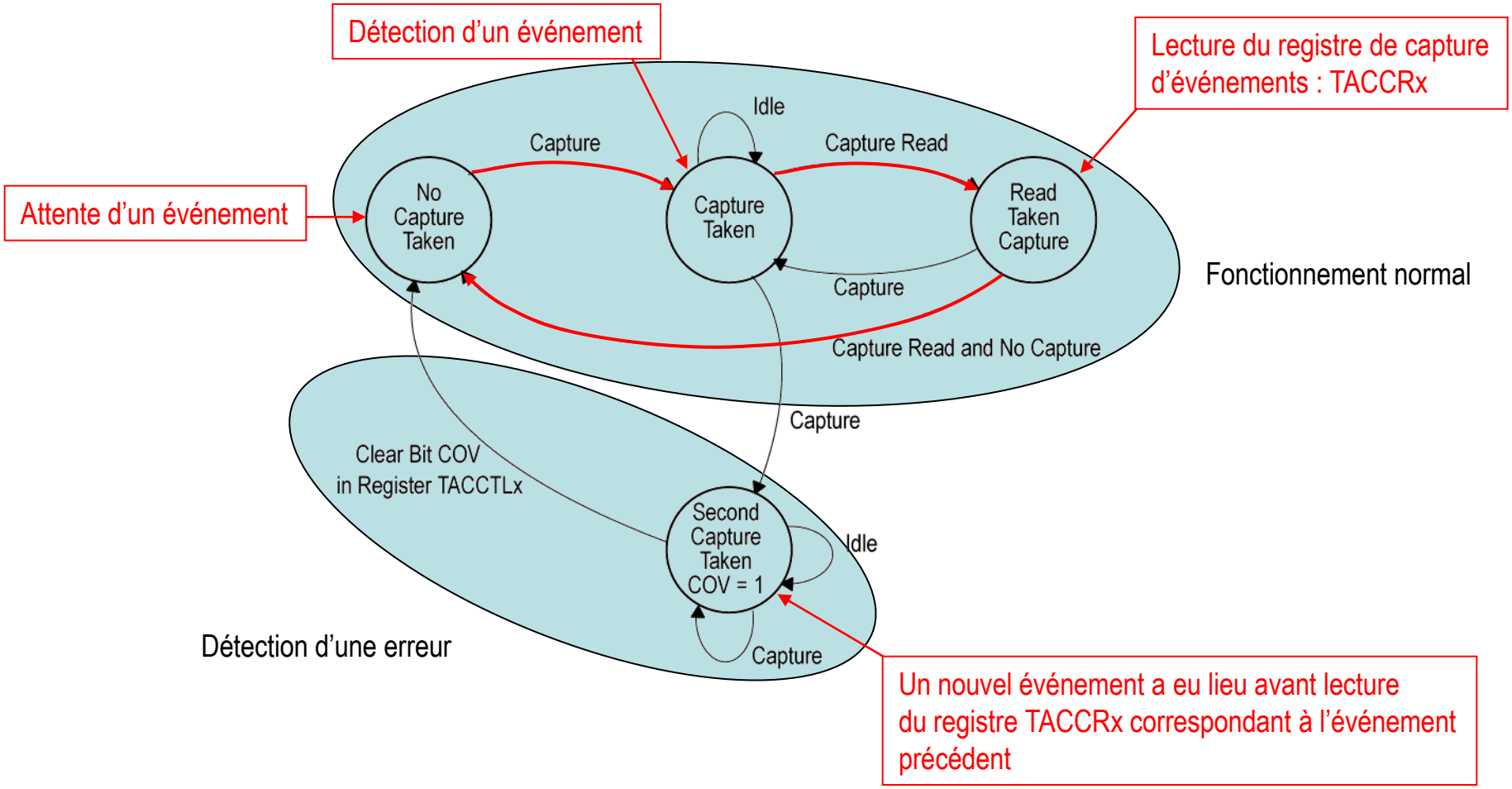


Timer A : Mode capture : CAP=1

Il est possible de synchroniser le signal de capture avec l'horloge de base du Timer.



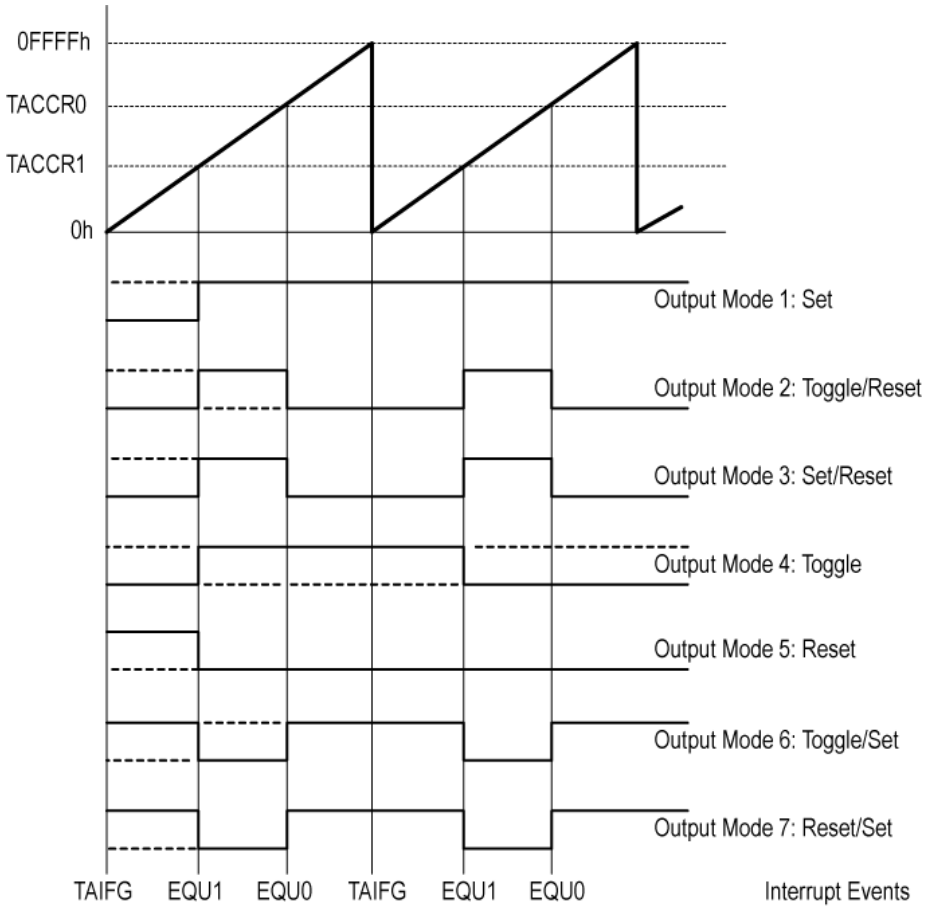
Timer A : Mode capture : CAP=1



Timer A : fonction de comparaison : CAP=0

OUTMODx	Mode	Description
000	Sortie	Le signal de sortie OUTx est défini par le bit OUT. Le signal OUTx est mis à jour dès que le bit OUT est modifié
001	Set	Le signal de sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TACCRx. Il y reste jusqu'à ce qu'un autre mode soit sélectionné
010	Toggle/Reset	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TACCRx La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TACCR0
011	Set/Reset	La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TACCRx La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TACCR0
100	Toggle	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TACCRx La période du signal est le double de la période du Timer
101	Reset	Le signal de sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TACCRx. Il y reste jusqu'à ce qu'un autre mode soit sélectionné
110	Toggle/Set	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TACCRx La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TACCR0
111	Reset/Set	La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TACCRx La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TACCR0

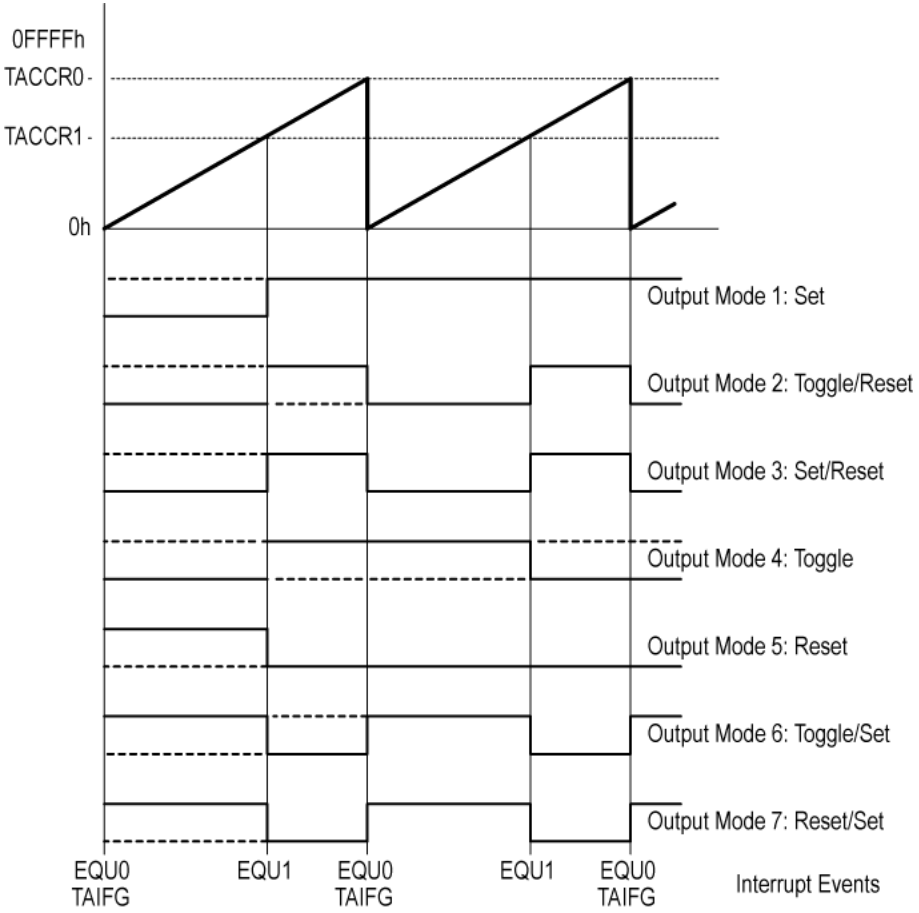
Exemple : Timer en mode continu
(Continuous Mode : MC1=10)



Timer A : fonction de comparaison : CAP=0

OUTMODx	Mode	Description
000	Sortie	Le signal de sortie OUTx est défini par le bit OUT. Le signal OUTx est mis à jour dès que le bit OUT est modifié
001	Set	Le signal de sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TACCRx. Il y reste jusqu'à ce qu'un autre mode soit sélectionné
010	Toggle/Reset	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TACCRx La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TACCR0
011	Set/Reset	La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TACCRx La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TACCR0
100	Toggle	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TACCRx La période du signal est le double de la période du Timer
101	Reset	Le signal de sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TACCRx. Il y reste jusqu'à ce qu'un autre mode soit sélectionné
110	Toggle/Set	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TACCRx La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TACCR0
111	Reset/Set	La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TACCRx La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TACCR0

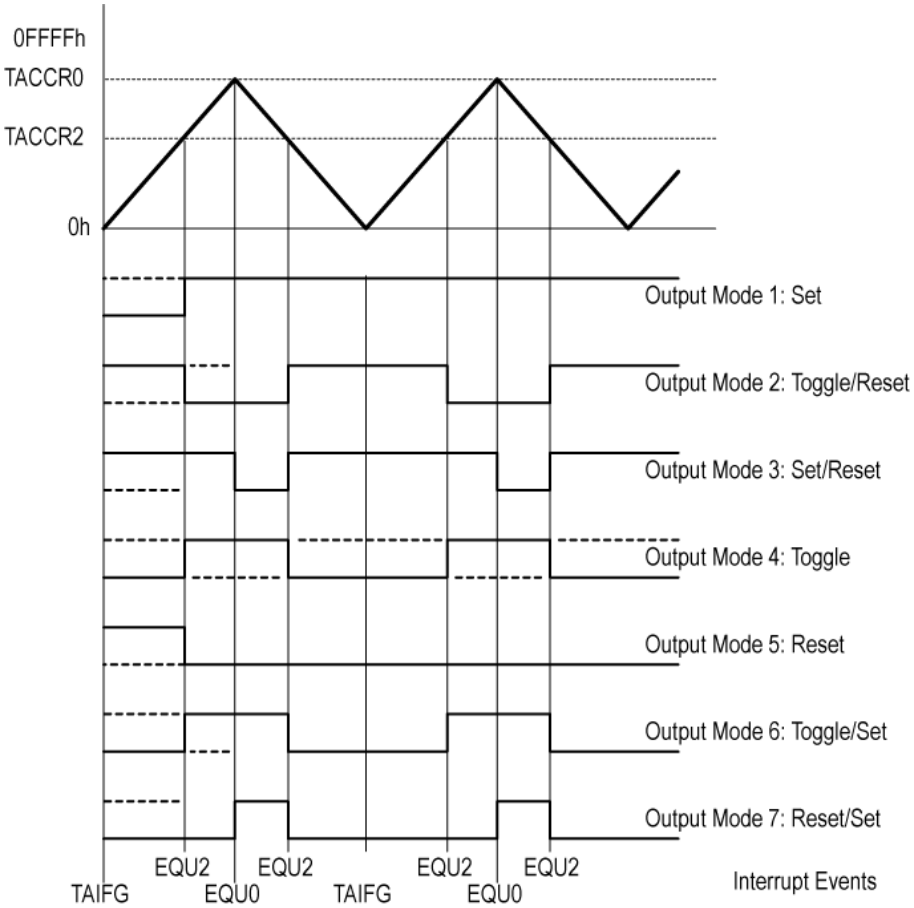
Exemple : Timer en mode comptage
(UP Mode : MC1=01)



Timer A : fonction de comparaison : CAP=0

OUTMODx	Mode	Description
000	Sortie	Le signal de sortie OUTx est défini par le bit OUT. Le signal OUTx est mis à jour dès que le bit OUT est modifié
001	Set	Le signal de sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TACCRx. Il y reste jusqu'à ce qu'un autre mode soit sélectionné
010	Toggle/Reset	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TACCRx La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TACCR0
011	Set/Reset	La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TACCRx La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TACCR0
100	Toggle	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TACCRx La période du signal est le double de la période du Timer
101	Reset	Le signal de sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TACCRx. Il y reste jusqu'à ce qu'un autre mode soit sélectionné
110	Toggle/Set	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TACCRx La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TACCR0
111	Reset/Set	La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TACCRx La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TACCR0

Exemple : Timer en mode compteur/décompteur (UP/Down Mode : MC1=11)



Timer A : Vecteurs d'Interruptions associés au Timer A

Deux vecteurs d'interruption sont associés au Timer A:

- TACCR0 pour TACCR0 CCIFG
- TAIV pour tous les autres flags CCIFG et TAIFG

En mode capture, n'importe quel flag CCIFG est activé lorsqu'une valeur du Timer est capturée dans le registre associé TACCRx.

En mode comparaison, n'importe quel flag CCIFG est activé si la valeur de comptage atteint la valeur contenue dans le registre TACCRx.

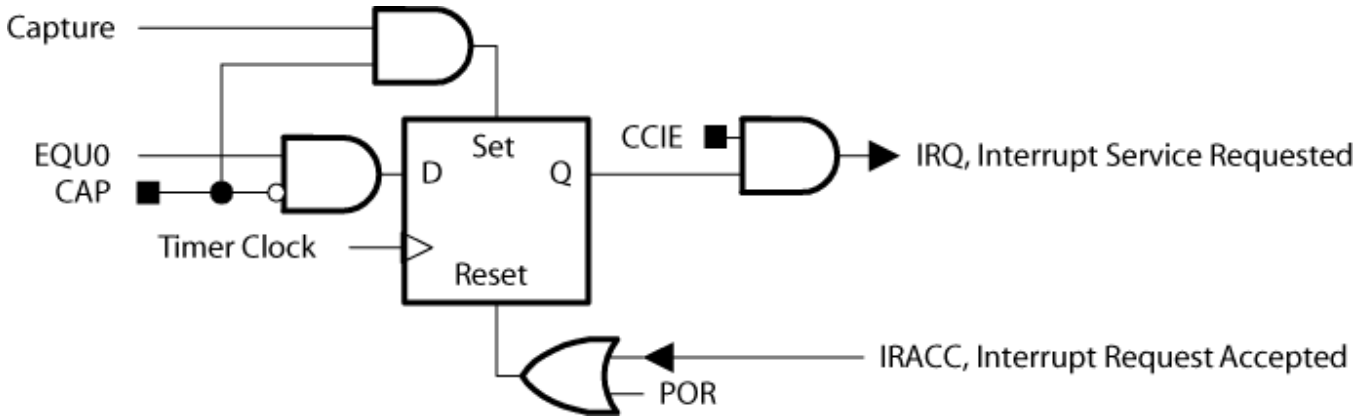
Par software il et également possible d'activer ou de désactiver n'importe quel flag CCIFG.

Tous les flags CCIFG provoquent une interruption lorsque leurs bits de masquage correspondants CCIE at le bit de masquage général GIE sont activés (mis à 1).

Timer A : Vecteurs d'Interruptions associés au Timer A

Interruption TACCR0

Le flag TACCR0 CCIFG a la plus haute priorité du Timer A et possède un vecteur d'interruption.



Le flag TACCR0 CCIFG est automatiquement remis à 0 (reset) lorsque l'interruption TACCR0 est exécutée.

Timer A : Vecteurs d'Interruptions associés au Timer A

Registre de génération du vecteur d'interruption TAIV

- Les flags TACCR1 CCIFG, TACCR2 CCIFG, et TAIFG sont combinés de manière à ne former qu'un seul vecteur d'interruption.
- Le registre du vecteur d'interruption TAIV est utilisé pour déterminer quel flag est à la source d'une demande d'interruption.
- L'interruption de priorité la plus haute défini la valeur dans le registre TAIV.
- Cette valeur peut être évaluée et ajoutée au compacteur de programme (PC) de manière à entrer automatiquement dans la routine d'interruption correspondante.
- La désactivation des interruptions du Timer A n'affecte pas le contenu du registre TAIV.
- Un accès en lecture et en écriture du registre TAIV annule automatiquement l'interruption pendante de priorité la plus haute.
- Si un second flag d'interruption est activé, une seconde interruption est immédiatement générée.

Par exemple :

- Si les flags TACCR1 et TACCR2 CCIFG sont activés, lorsque la routine d'interruption accède au registre TAIV, le flag TACCR1 CCIFG est automatiquement forcé à 0.
- Après l'instruction de retour d'interruption (RETI), le flag TACCR2 CCIFG génère immédiatement une seconde interruption.

Timer A : liste des registres

Registre	Nom	Type de registre	Adresse	Etat initial
Registre de contrôle	TACTL	Lecture / Ecriture	0x0160	Reset avec POR
Registre interne du Timer	TARx	Lecture / Ecriture	0x0170	Reset avec POR
Contrôle capture/comparaison 0	TACCTL0	Lecture / Ecriture	0x0162	Reset avec POR
Capture/comparaison 0	TACCR0	Lecture / Ecriture	0x0172	Reset avec POR
Contrôle capture/comparaison 1	TACCTL1	Lecture / Ecriture	0x0164	Reset avec POR
Capture/comparaison 1	TACCR1	Lecture / Ecriture	0x0174	Reset avec POR
Contrôle capture/comparaison 2	TACCTL2	Lecture / Ecriture	0x0166	Reset avec POR
Capture/comparaison 2	TACCR2	Lecture / Ecriture	0x0176	Reset avec POR
Vecteur d'interruption	TAIV	Lecture / Ecriture	0x012E	Reset avec POR

Chaque registre contient plusieurs champs qui ont chacun leur fonction de définition et contrôle.

Timer A : registre de contrôle **TACTL** (16 bits)

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

7	6	5	4	3	2	1	0
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	w-(0)	rw-(0)	rw-(0)

TASSELx TACTL[9 8] : Horloge de référence
00 : TACLK
01 : ACLK
10 : SMCLK
11 : $\overline{\text{TACLK}}$

IDx TACTL [7 6] : Diviseur de l'horloge de référence
00 : /1
01 : /2
10 : /4
11 : /8

MCx TACTL [5 4] : contrôle de mode
00 : Stop mode, le Timer est arrêté (low power)
01 : Up mode: Le Timer compte jusqu'à TACCR0
10 : Continuous mode : Le Timer compte jusqu'à 0FFFFh
11 : Up/down mode : Le Timer compte jsuqu'à TACCR0 puis décompte jusqu'à 0000h

TACLR TACTL [2] : Reset du compteur : TAR=0, IDx=00 et MCx=00

TAIE TACTL[1] : Contrôle les interruptions (TAIFG).
0 : interruptions désactivées
1 : interruption actives

TAIFG TACTL[0] : flag d'état des interruptions
0 : pas d'interruption pendante
1 : interruption pendante

Timer A : registre interne du Timer **TARx**

15	14	13	12	11	10	9	8
TARx							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

7	6	5	4	3	2	1	0
TARx							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

TARx [15 0] :
Contient la valeur interne du Timer A

Timer A : registre de comparaison et de capture **TACCRx**

15	14	13	12	11	10	9	8
TACCRx							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

7	6	5	4	3	2	1	0
TACCRx							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

TACCRx [15 0] :
Compare mode : TACCRx contient la valeur de comparaison avec le registre interne du Timer A TAR.
Capture mode : Le contenu du registre interne du Timer A TAR est copié dans TACCRx lors de la capture d'un événement

Timer A : registre de contrôle de capture/comparaison **TACCTLx**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)

7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CMx TACCTLx[15 14] : en mode capture
00 : pas de capture
01 : capture sur flanc montant
10 : capture sur flanc descendant
11 : capture sur les deux flancs

CCISx TACCTLx[13 12] : sélection du signal d'entrée
00 : CCIxA (x=0 → P1.0, x=1 → P1.2, x=2 → P2.0)
01 : CCIxB (x=0 → P1.1, x=1 → CAOUT, x=2 → ACLK)
10 : GND
11 : VCC

SCS TACCTLx[11] : Synchronisation du signal d'entrée avec l'horloge de référence.
0 : pas de synchronisation (capture asynchrone)
1 : synchronisation (capture synchrone)

SCCI TACCTLx[1] : Signal d'entrée de capture/comparaison synchronisé (latch) avec le signal EQUx

CAP TACCTL[8] : Contrôle du mode capture/comparaison
0 : mode comparaison
1 : mode capture

Timer A : registre de contrôle de capture/comparaison **TACCTLx**

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)

7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

OUTMODx TACCTLx[7 5] : Mode de sortie
Les modes 2, 3, 6 et 7 ne sont pas utilisés pour TACCR0 car EQUx=EQU0
000 : Sortie
001 : Set
010 : Toggle/reset
011 : Set/reset
100 : Toggle
101 : Reset
110 : Toggle/set
111 : Reset/set

CCIE TACCTLx[4] : Activation des interruptions de capture/comparaison (flag CCIFG).
0 : interruption désactivée
1 : interruption activée

CCI TACCTLx[3] : Signal d'entrée de capture/comparaison

OUT TACCTL[2] : Contrôle du bit de sortie pour le mode OUTMODx=000.
0 : sortie forcée à 0
1 : sortie forcée à 1

COV TACCTLx[1] : dépassement en mode capture (COV doit être remis à 0 par software).
0 : pas de dépassement
1 : détection d'un dépassement

CCIFG TACCTL[0] : flag d'interruption de capture/comparaison
0 : pas d'interruption pendante
1 : interruption pendante

Timer A : registre du vecteur d'interruption **TAIV**

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
r0	r0	r0	r0	r0	r0	r0	r0

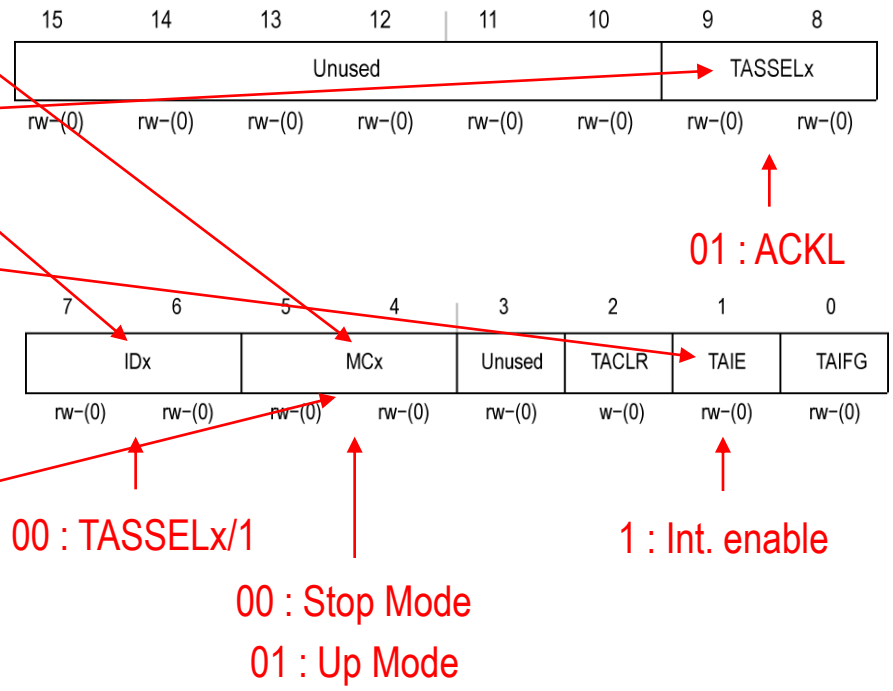
7	6	5	4	3	2	1	0
0	0	0	0	TAIVx			0
r0	r0	r0	r0	r−(0)	r−(0)	r−(0)	r0

TAIV [15 0] : Vecteur d'interruption du Timer A
0000 : pas d'interruption pendante
0010 : interruption du bloc CC1 de capture/comparaison, TACCR1 flag CCIFG (interruption de priorité la plus élevée)
0100 : interruption du bloc CC2 de capture/comparaison, TACCR2 flag CCIFG
1010 : Dépassement (flag d'interruption TAIFG)

Exemple, initialisation d'une interruption du Timer A

```
void timerA_init()              // initialisation timer A
{
  TACTL_bit.TAMC = 0;           // Stop Timer A
  TACTL_bit.TASSEL = 1;         // CLK source : ACLK
  TACTL_bit.TAID = 0;           // ACLK / 1
  TACTL_bit.TAIE = 1;           // Interrupt enable
  TACCTL0_bit.CCIE = 1;         // Interrupt compare
  TACCR0 = 1000;                // Valeur max du compteur
  TACTL_bit.TAMC = 1;           // Compteur en up mode
}
```

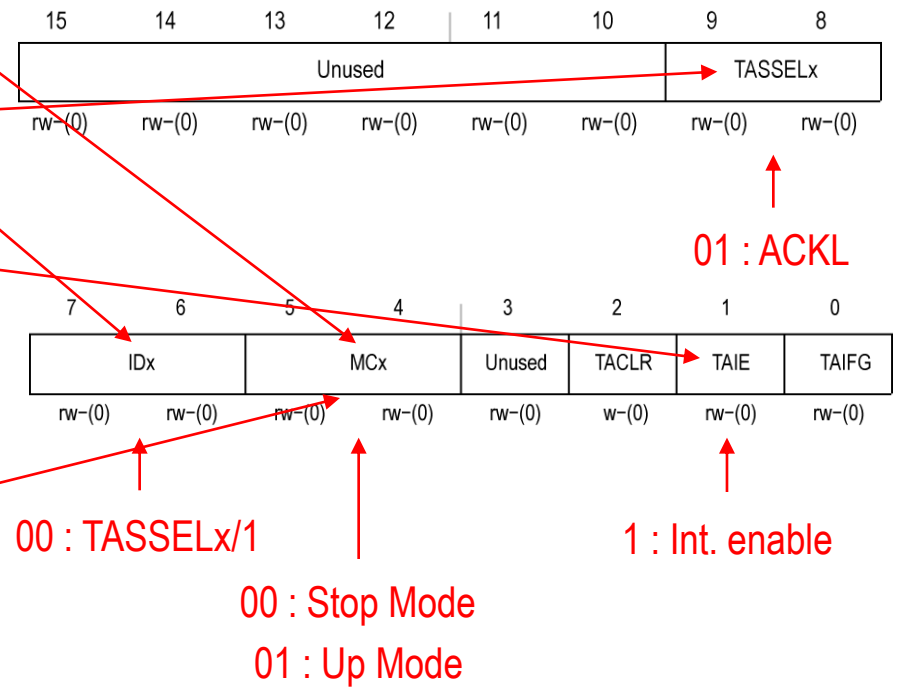
Registre TACTL



Exemple, initialisation d'une interruption du Timer A – listing alternatif

```
void timerA_init()          // initialisation timer A
{
  TACTL |= TACLR;           // Reset Timer A
  TACTL |= TASSEL _1;       // CLK source = ACLK
  TACTL |= ID_0;            // ACLK / 1
  TACTL |= TAIE;            // Interrupt  enable
  TACCTL0 |= CCIE;          // Interrupt  compare
  TACCR0 = 1000;            // Valeur max du compteur
  TACTL |= MC_1;            // Compteur en up mode
}
```

Registre TACTL



Exemple, initialisation d'une interruption du Timer A

```
void timerA_init()              //initialisation timer A
{
   TACTL_bit.TAMC = 0;          // Stop Timer A
   TACTL_bit.TASSEL = 1;        // CLK source : ACLK
   TACTL_bit.TAID = 0;          // ACLK / 1
   TACTL_bit.TAIE = 1;          // Interrupt enable
   TACCTL0_bit.CCIE = 1;        // Interrupt compare
   TACCR0 = 1000;               // Valeur max du compteur
   TACTL_bit.TAMC = 1;          // Compteur en up mode
}
```

Registre TACCTL0

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)

7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

1 : Int. sur comparaison

Registre TACCR0

15	14	13	12	11	10	9	8
TACCR0							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

15	14	13	12	11	10	9	8
TACCR0							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Timer B : Caractéristiques principales

- Timer_B est un timer/compteur de 16 bits avec sept registres de capture et de comparaison
- Timer_B supporte multiples captures / comparaisons, sortie PWM, et compteur d'intervalles.
- Timer_B a également des possibilités étendues d'interruption.
- Des interruptions peuvent être générées par des dépassement du compteur ou par chacun des registres de comparaison et de capture.

Les dispositifs de Timer_B incluent :

- ⇨ Timer / compteur de 8, 10, 12 ou 16 bits asynchrone avec quatre modes opératoire.
- ⇨ Base de temps (horloge) sélectionnable et configurable.
- ⇨ Sept registres de comparaisons et de captures configurables.
- ⇨ Sorties configurables avec des possibilités de modulation de largeurs d’impulsions (PWM).
- ⇨ Double comparateur avec mémorisation et chargement synchronisé.
- ⇨ Vecteur d'interruption (registre) pour un décodage rapide des interruptions du Timer_B.

Timer B : Caractéristiques principales

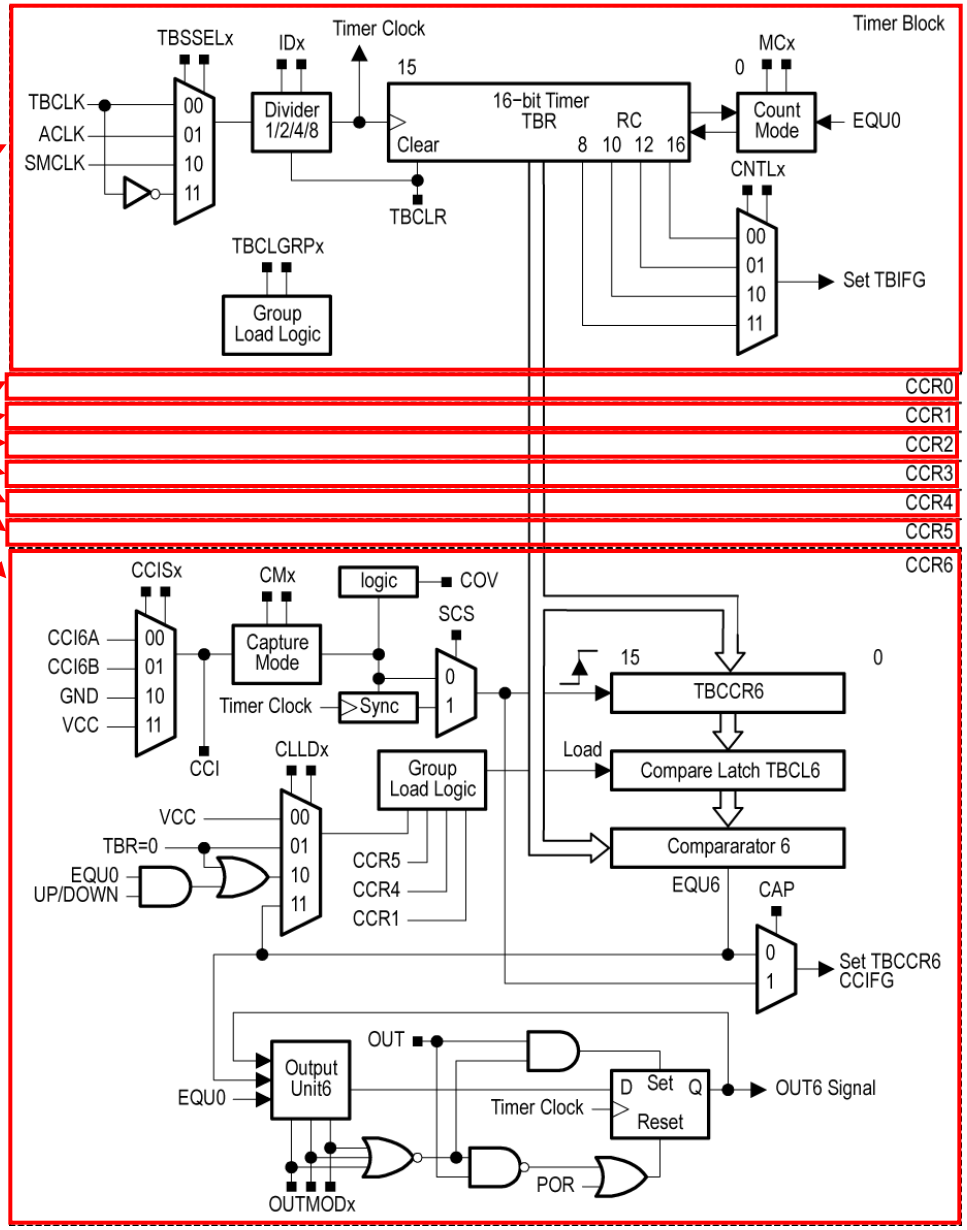
Similarités et différences avec le Timer A :

- Nombre de bit du Timer B programmable de 8, 10, 12, 16 bits.
- Base de temps (horloge) sélectionnable et configurable.
- Les registres de comparaison/capture sont doublés et peuvent être chargés de manière synchrone, ils peuvent aussi être groupés.
- Les sorties du Timer B peuvent être mises en haute impédance (tristate).
- Double comparateur avec mémorisation et chargement synchronisé.
- Le flag SCCI (état du signal d'entrée synchronisé) n'est pas implémenté dans le Timer B.

Timer B7 : Caractéristiques principales

Fonction Timer de base

7 modules de capture et de comparaison



Timer B7 : Caractéristiques principales

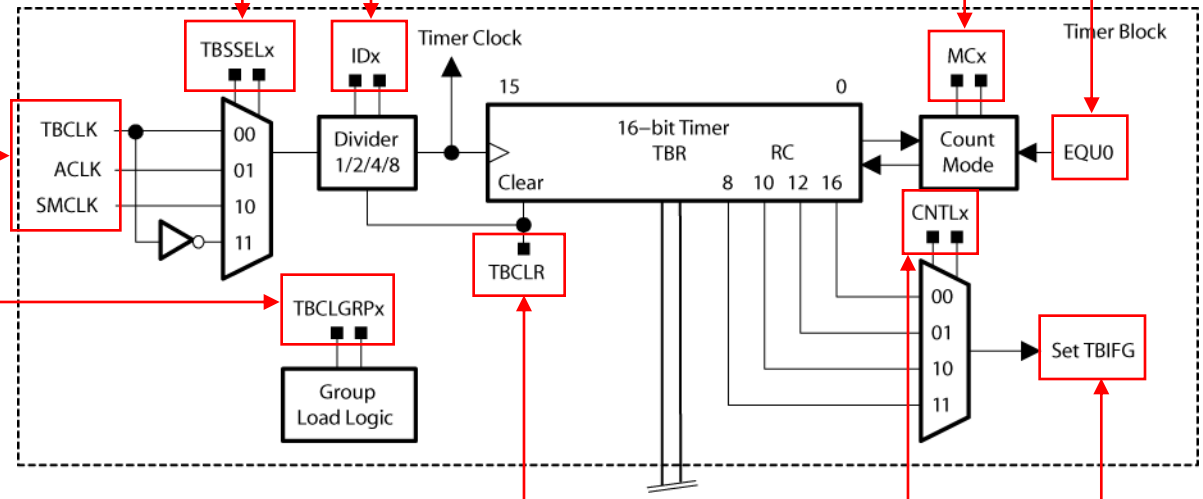
Flag indiquant l'état de comparaison pour TBCCR0

00 STOP
01 Comptage positif jusqu'à la valeur dans TBCCR0
10 Continu, comptage jusqu'au maximum 0xFFFF
11 Comptage et décomptage entre 0 et la valeur dans TBCCR0

Sélection de la base de temps

Choix de la valeur de division de la base de temps

Source de la base de temps (horloge de référence)



Mise à zéro du timer

Sélection de la taille (nb de bits) du Timer

Flag indiquant une interruption due à un dépassement

Timer B7 : Caractéristiques principales

Synchronisation de l'événement avec l'horloge du timer

Indique si un nouvel événement a eu lieu avant lecture du précédent

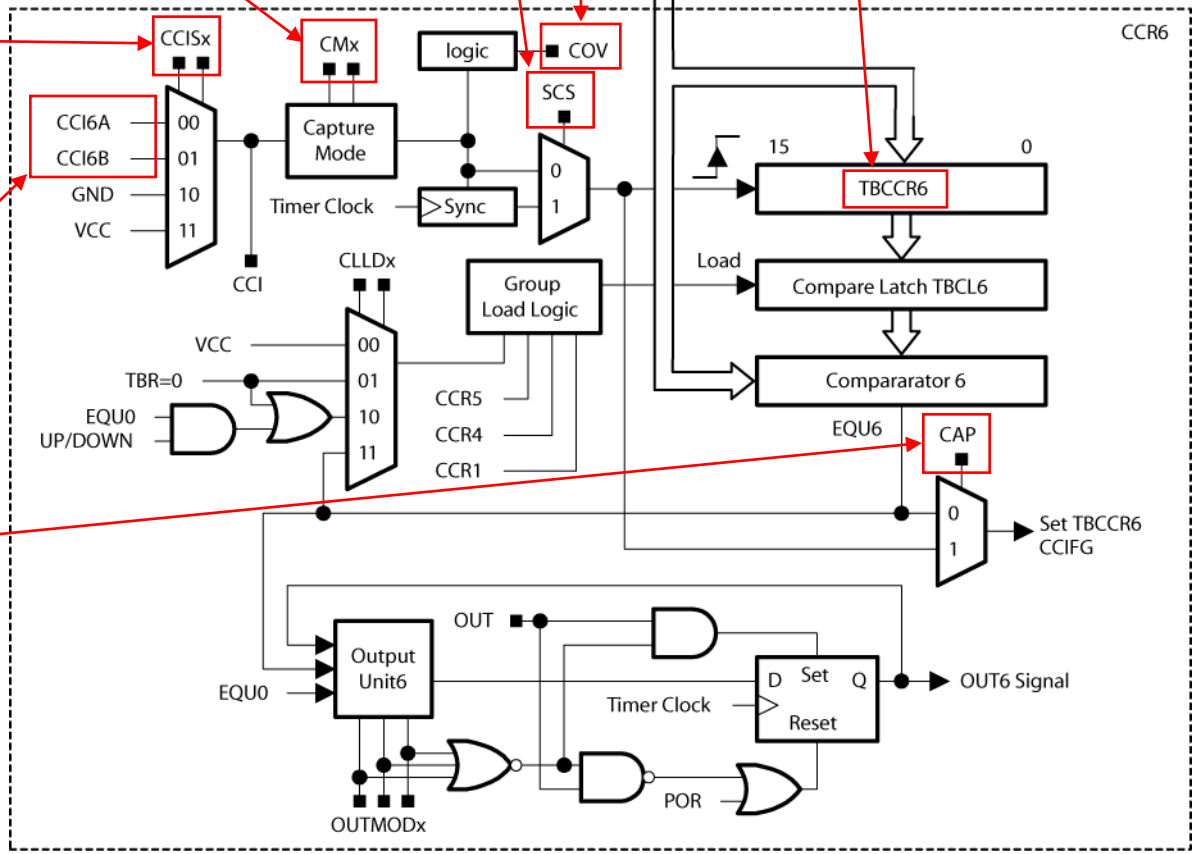
Mémorisation de la valeur du timer lorsqu'un événement apparaît

Type d'évènement (flanc montant , descendant ou les deux)

Sélection du signal de capture

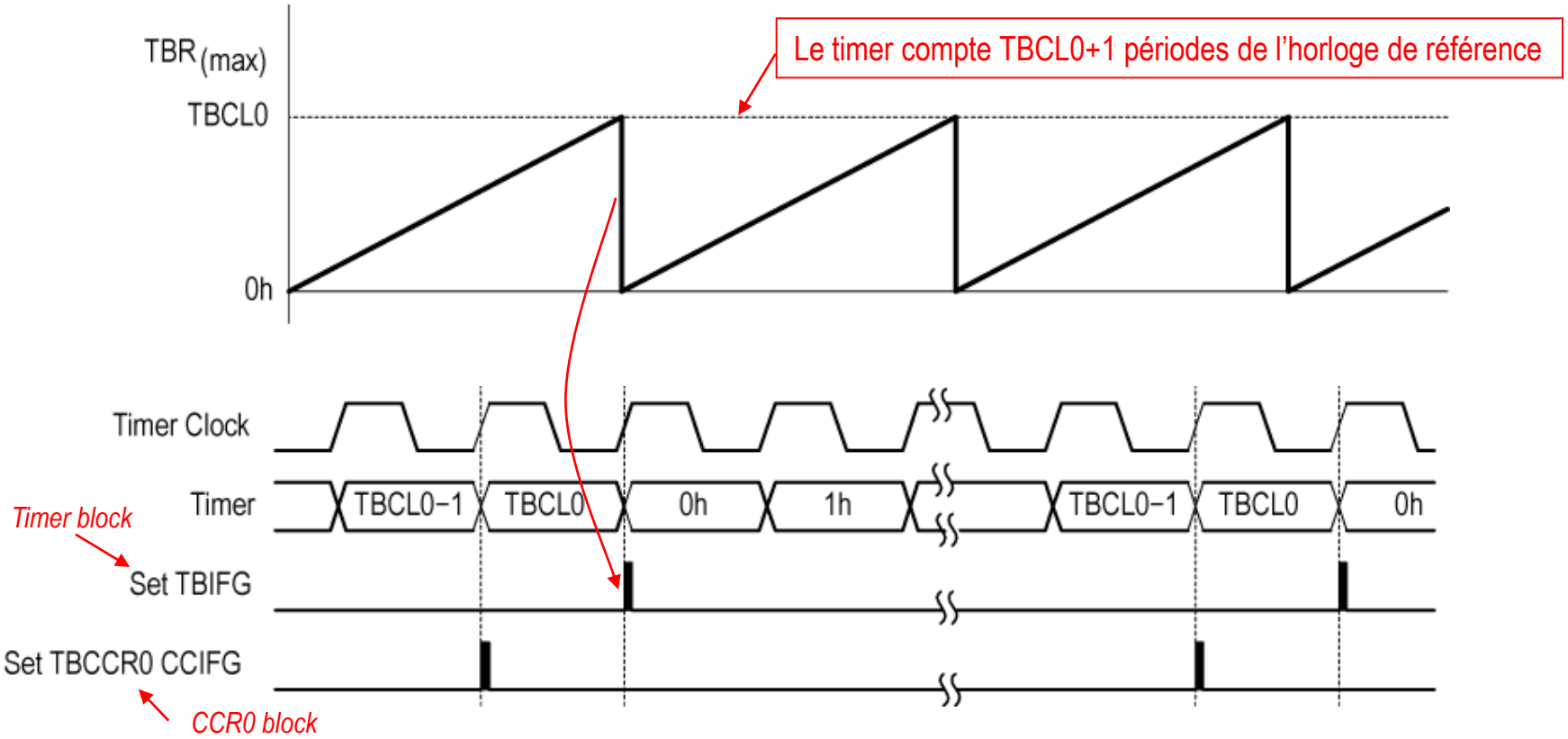
Signaux d'entrée pour capture

Défini le mode capture ou le mode comparaison



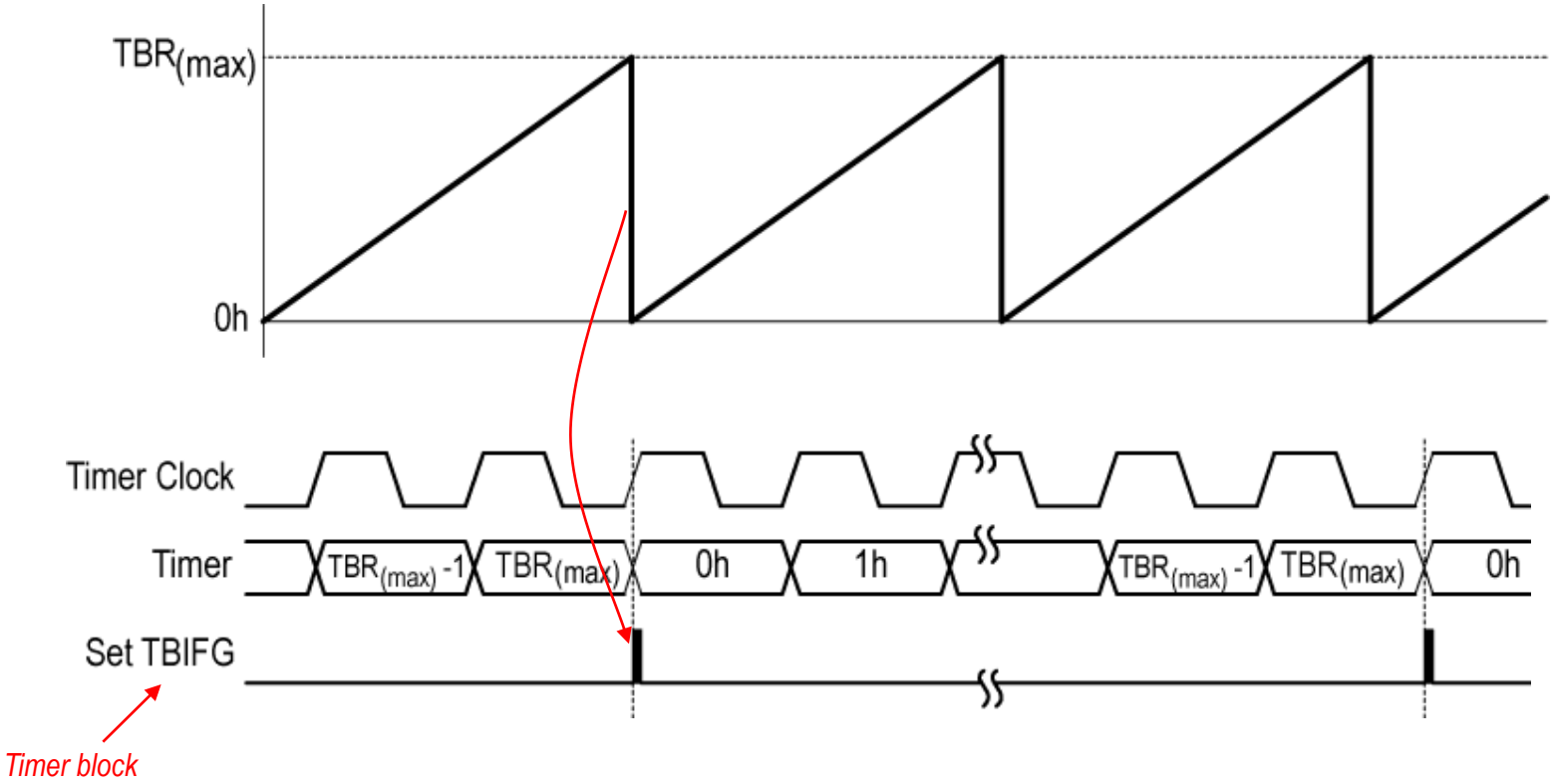
Timer B : Mode de comptage (UP Mode) : MCx = 01

Le timer compte jusqu'à la valeur mémorisée dans le registre TBCL0 puis revient à 0x0



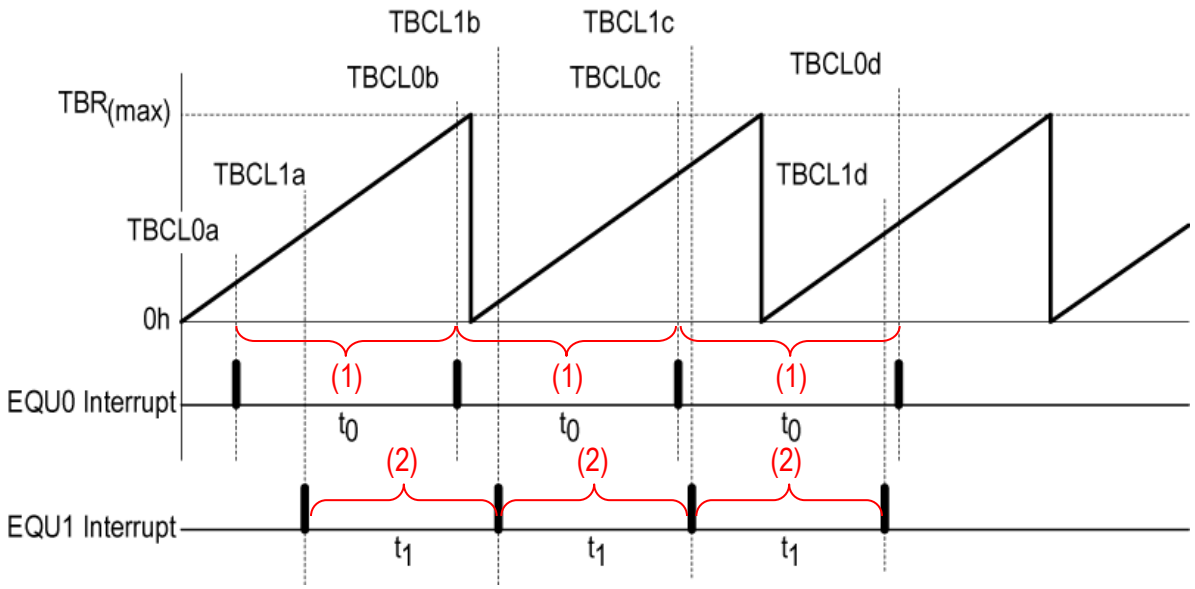
Timer B : Mode de comptage continu (Continuous Mode) : MCx = 10

Le timer compte jusqu'à la valeur maximum (0xFF, 0x3FF, 0xFFF, 0xFFFF) puis revient à 0x0



Timer B : Mode de comptage continu (Continuous Mode) : MCx = 10

- Le mode continu peut être employé pour générer des intervalles de temps.
- Chaque fois qu'un intervalle est terminé, une interruption est activée.
- Le prochain intervalle est ajouté au TBCLx dans la routine d'interruption.
- Jusqu'à 7 (Timer_B7) intervalles de temps peuvent être générés en utilisant les registres de comparaison / capture TBCL0 à TBCL6.
- Une éventuelle gigue sur la latence d'exécution de la routine d'interruption n'a aucun impact
- sur la valeur temporelle de l'intervalle.

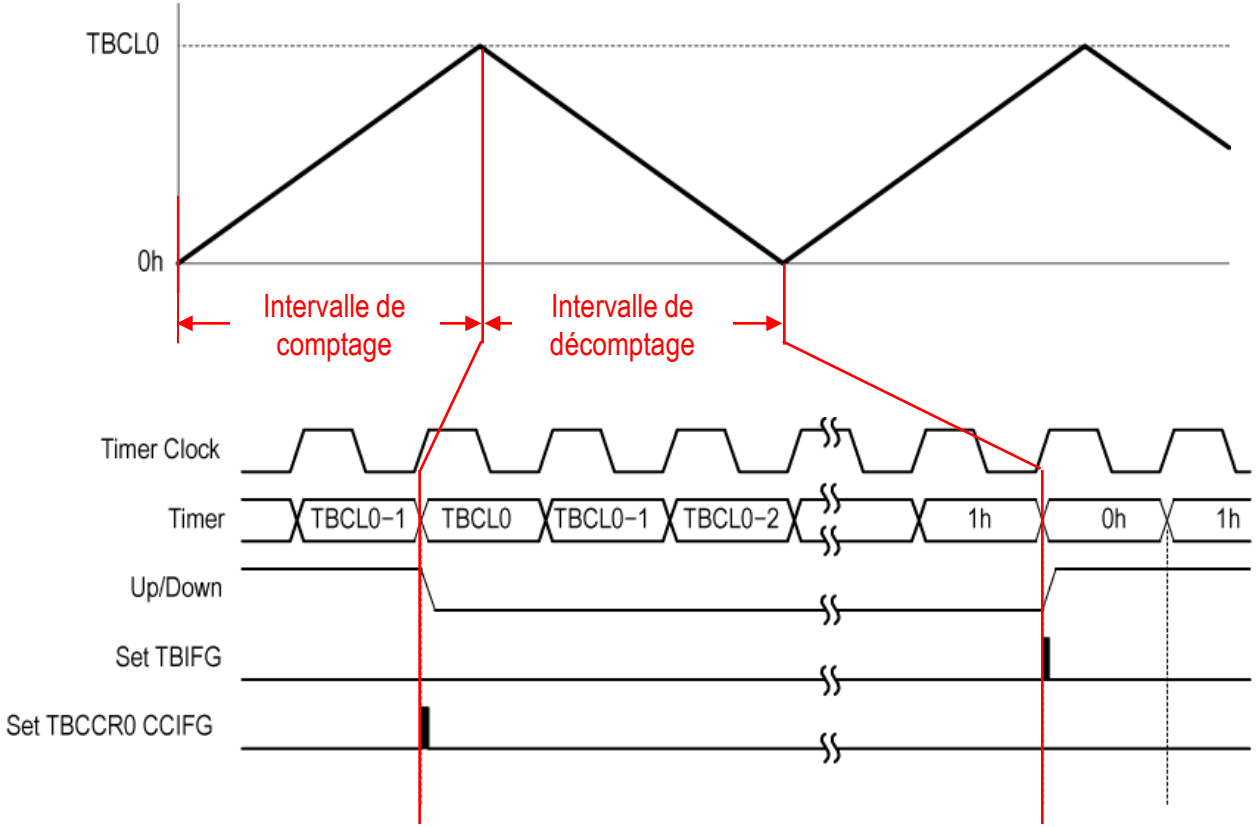


(1) :Valeur ajoutée au registre TBCCR0

(2) :Valeur ajoutée au registre TBCCR1

Timer B : Mode Up / Down : MCx = 11

- Le mode compteur / décompteur est employé si la période du timer doit être différente de $TBR_{(max)}$ et si la génération symétrique d'impulsions est nécessaire.
- Le timer compte jusqu'à la valeur de chargée dans le registre de comparaison / capture TBCL0 puis décompte jusqu'à 0x0.
- La période est deux fois la valeur dans TBCL0.

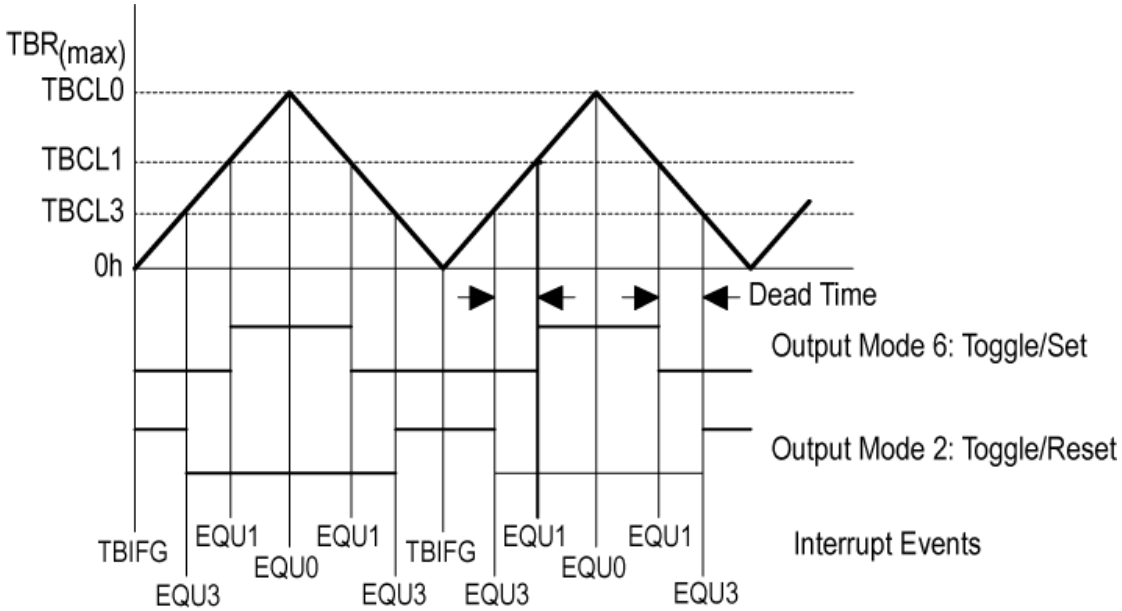


Timer B : Utilisation du mode Up / Down : MCx = 11

Le temps d'antichevauchement, c'est-à-dire le temps pour lequel les deux signaux de sortie sont à l'état bas peut être défini comme :

$$tdead = hclock \times (TBCL1 - TBCL3)$$

La possibilité de charger simultanément des groupes de registres de comparaison permet d'assurer le temps mort (différence majeur avec Timer A).



Timer B : Mode capture : CAP=1

Le mode capture est sélectionné lorsque CAP=1. Il permet d'enregistrer des événements temporels comme par exemple une position (en vue d'une mesure de vitesse) ou permet de réaliser une mesure de temps.

Type d'évènement (flanc montant , descendant ou les deux)

Sélection du signal de capture

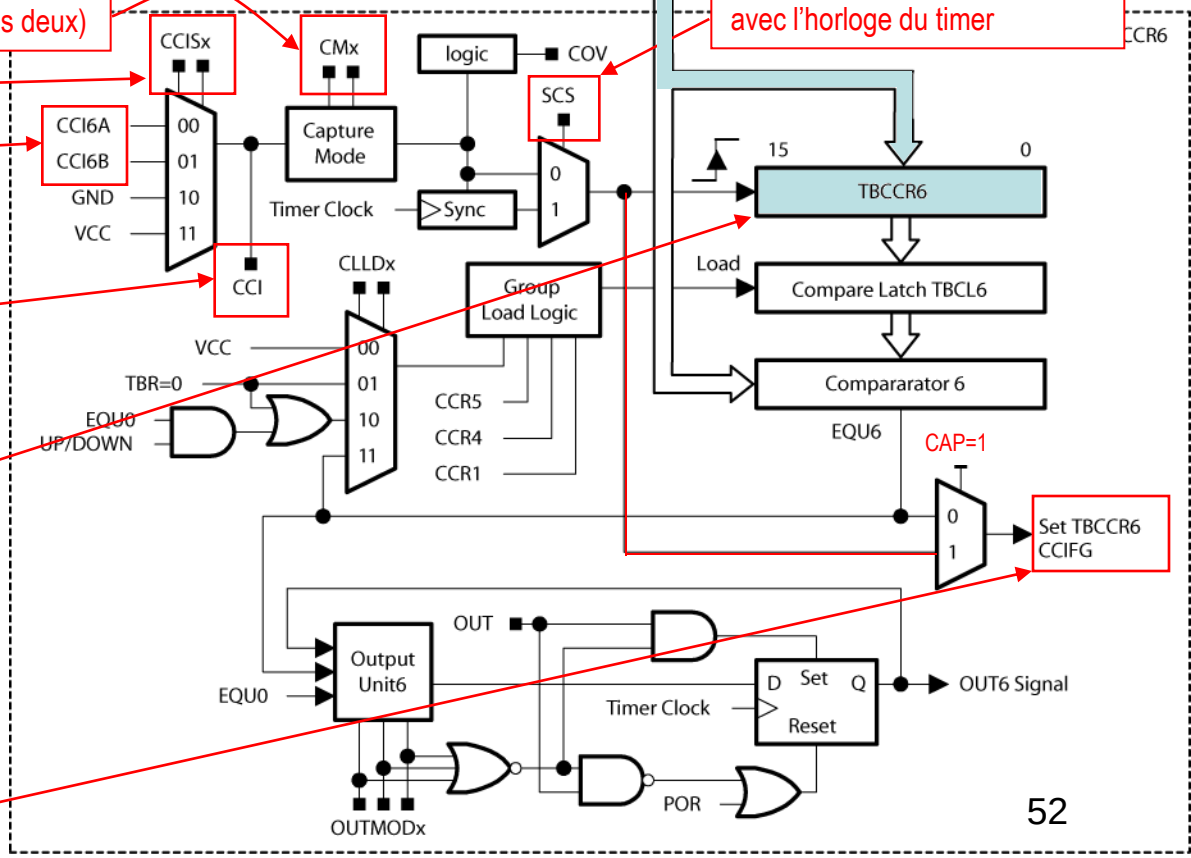
Signaux d'entrées pour capture
- Broches du MSP430
- Modules internes

Le signal d'entrée sélectionné peut être lu en tout temps

Lorsqu'un événement apparaît, le contenu du Timer est mémorisé dans le registre TBCCRx

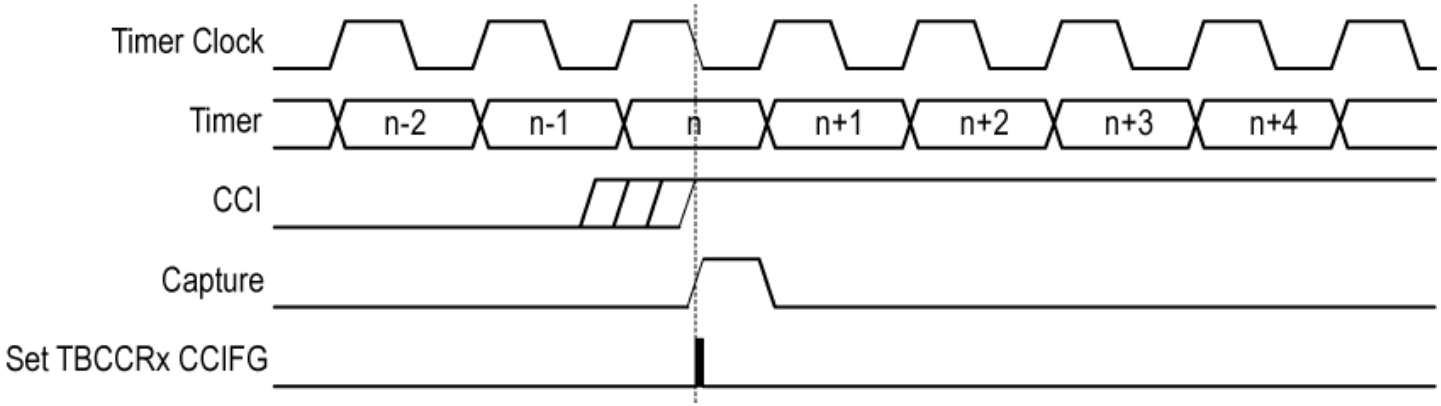
Activation du flag d'interruption CCIFG

Synchronisation de l'événement avec l'horloge du timer

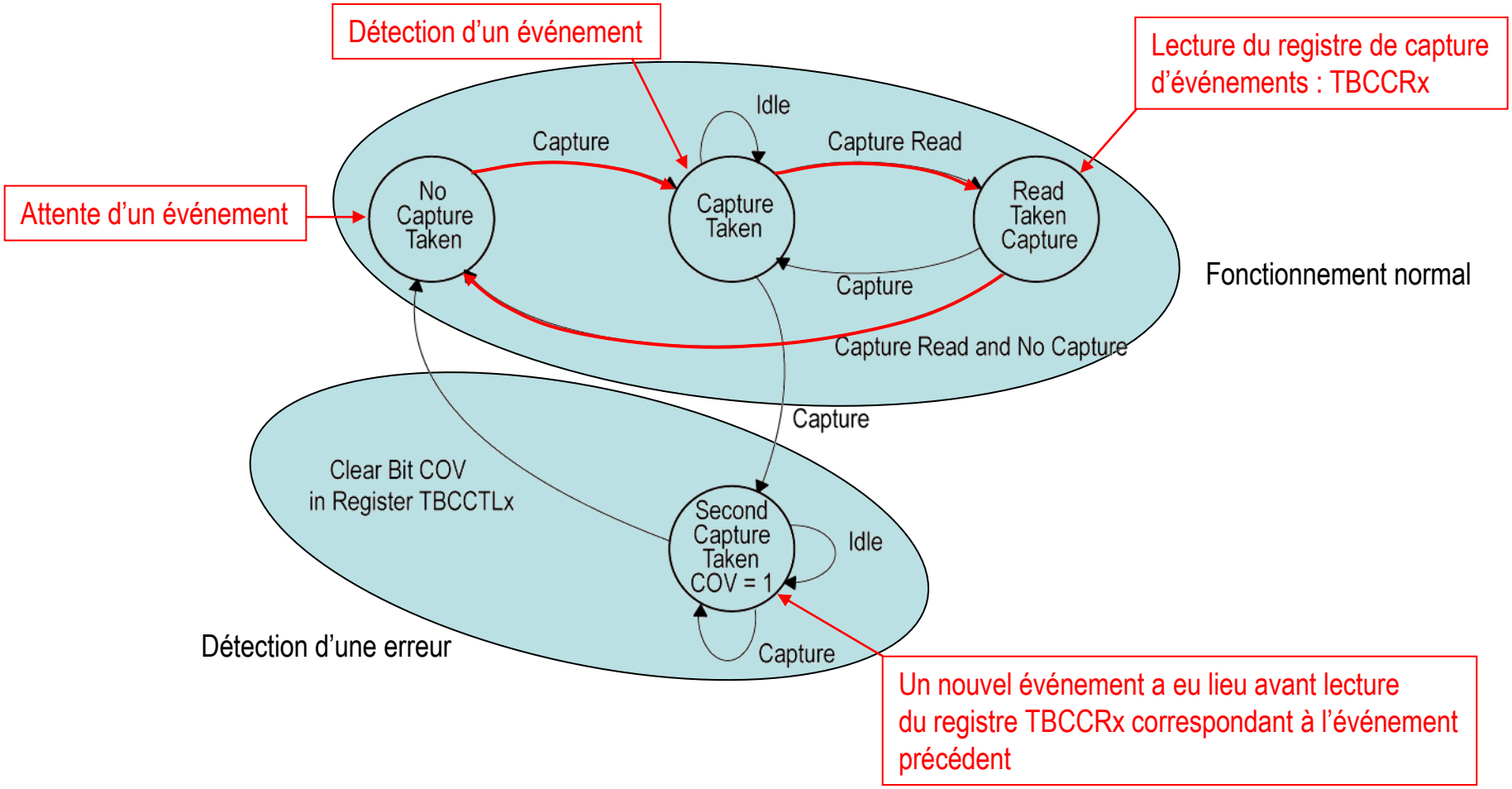


Timer B : Mode capture : CAP=1

Il est possible de synchroniser le signal de capture avec l'horloge de base du Timer.
Bien que pas nécessaire, il est vivement recommander de travailler de manière synchrone.



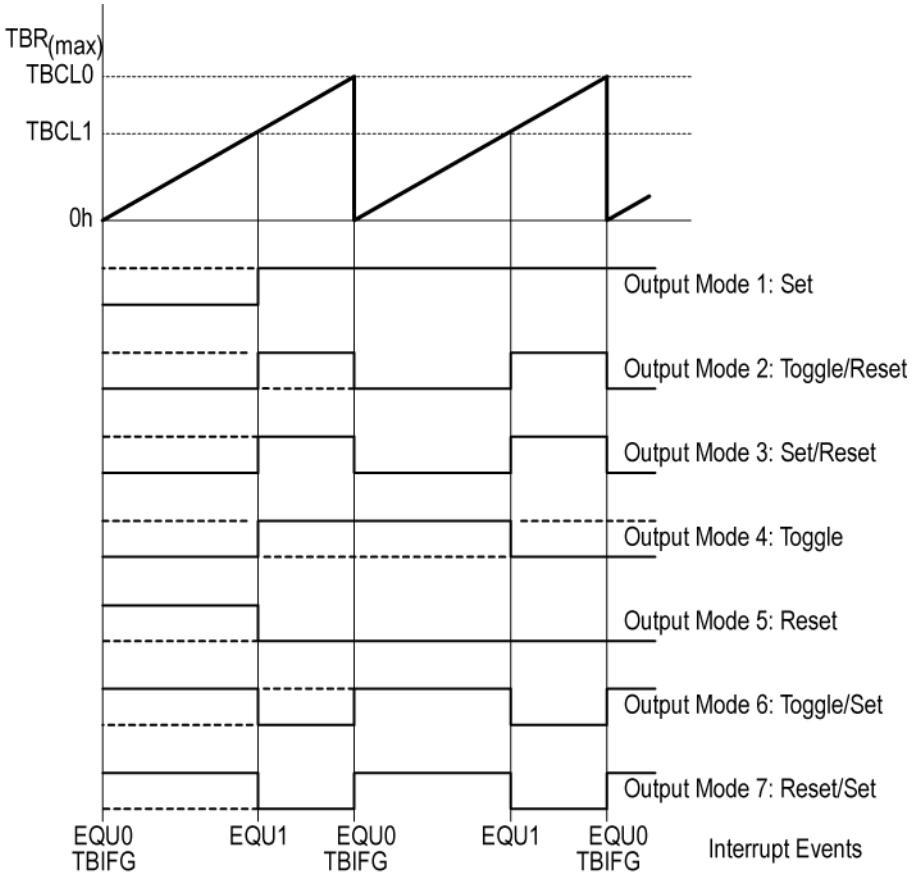
Timer B : Mode capture : CAP=1



Timer B : Mode comparaison : CAP=0

OUTMODx	Mode	Description
000	Sortie	Le signal de sortie OUTx est défini par le bit OUT. Le signal OUTx est mis à jour dès que le bit OUT est modifié
001	Set	Le signal de sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TBCLx. Il y reste jusqu'à ce qu'un autre mode soit sélectionné
010	Toggle/Reset	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TBCLx La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TBCL0
011	Set/Reset	La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TBCLx La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TBCL0
100	Toggle	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TBCLx La période du signal est le double de la période du Timer
101	Reset	Le signal de sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TBCLx. Il y reste jusqu'à ce qu'un autre mode soit sélectionné
110	Toggle/Set	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TBCLx La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TBCL0
111	Reset/Set	La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TBCLx La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TBCL0

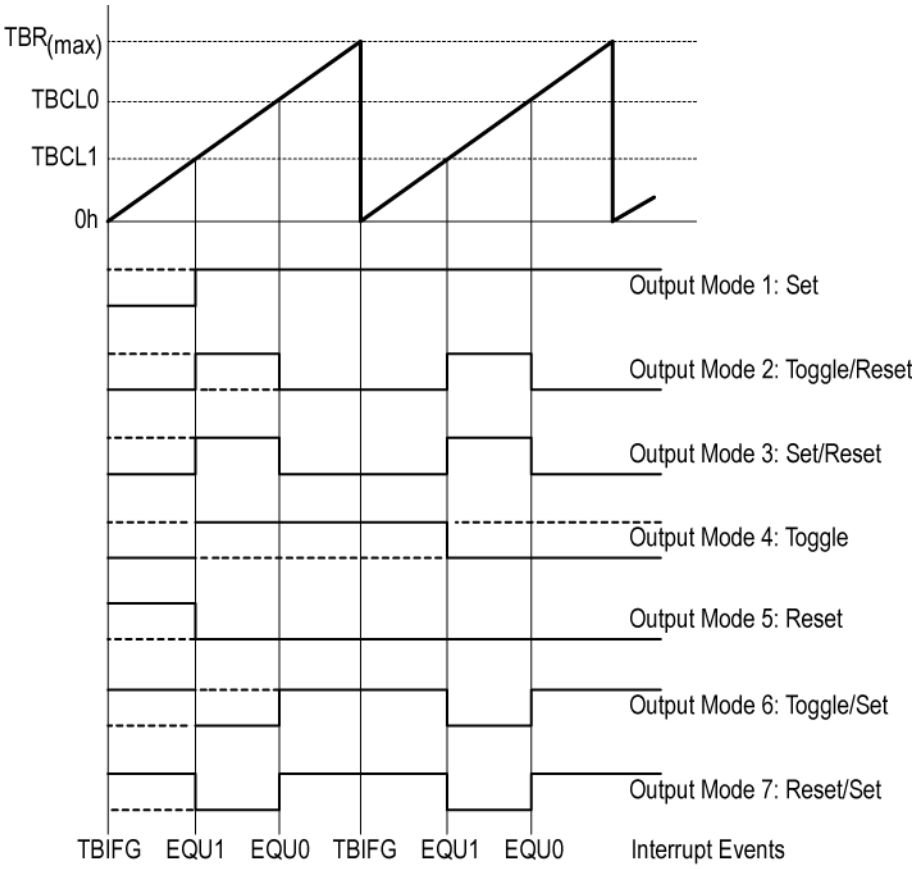
Exemple : Timer en mode continu
(Continuous Mode : MC1=10)



Timer B : Mode comparaison : CAP=0

OUTMODx	Mode	Description
000	Sortie	Le signal de sortie OUTx est défini par le bit OUT. Le signal OUTx est mis à jour dès que le bit OUT est modifié
001	Set	Le signal de sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TBCLx. Il y reste jusqu'à ce qu'un autre mode soit sélectionné
010	Toggle/Reset	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TBCLx La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TBCL0
011	Set/Reset	La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TBCLx La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TBCL0
100	Toggle	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TBCLx La période du signal est le double de la période du Timer
101	Reset	Le signal de sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TBCLx. Il y reste jusqu'à ce qu'un autre mode soit sélectionné
110	Toggle/Set	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TBCLx La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TBCL0
111	Reset/Set	La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TBCLx La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TBCL0

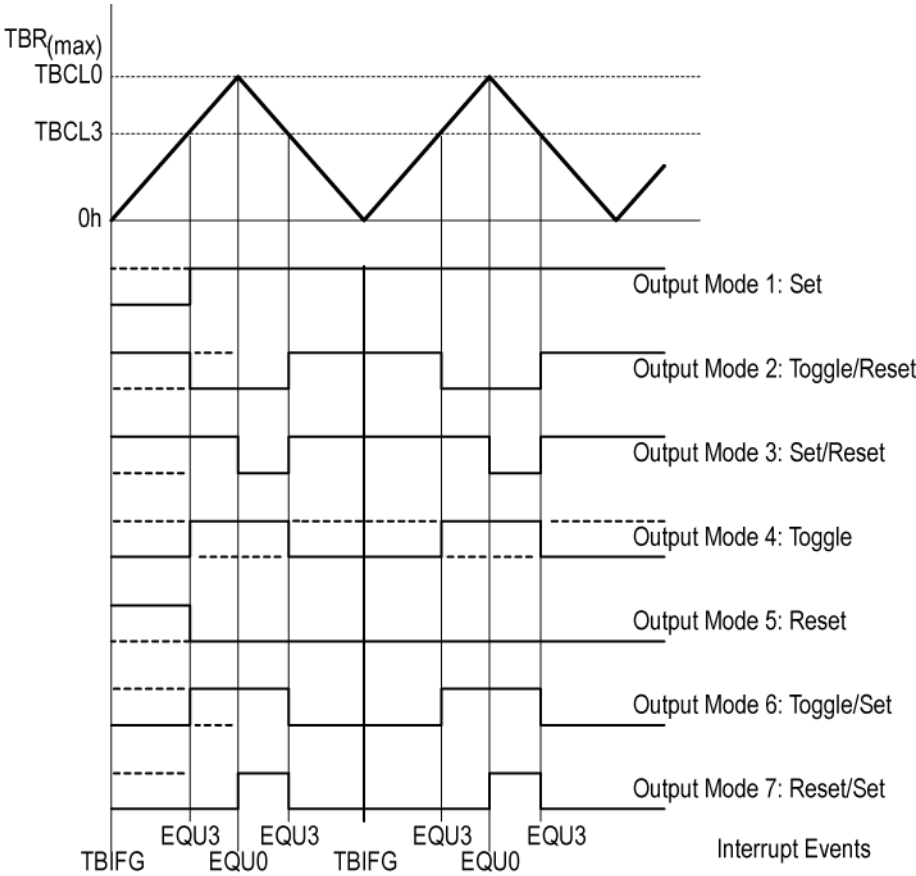
Exemple : Timer en mode comptage (UP Mode : MC1=01)



Timer B : Mode comparaison : CAP=0

OUTMODx	Mode	Description
000	Sortie	Le signal de sortie OUTx est défini par le bit OUT. Le signal OUTx est mis à jour dès que le bit OUT est modifié
001	Set	Le signal de sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TBCLx. Il y reste jusqu'à ce qu'un autre mode soit sélectionné
010	Toggle/Reset	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TBCLx La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TBCL0
011	Set/Reset	La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TBCLx La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TBCL0
100	Toggle	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TBCLx La période du signal est le double de la période du Timer
101	Reset	Le signal de sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TBCLx. Il y reste jusqu'à ce qu'un autre mode soit sélectionné
110	Toggle/Set	La sortie OUTx est inversée dès que le registre du Timer à atteint la valeur définie dans TBCLx La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TBCL0
111	Reset/Set	La sortie OUTx est forcée à 0 dès que le registre du Timer à atteint la valeur définie dans TBCLx La sortie OUTx est forcée à 1 dès que le registre du Timer à atteint la valeur définie dans TBCL0

Exemple : Timer en mode compteur/décompteur (UP/Down Mode : MC1=11)



Timer B : Vecteurs d'Interruptions associés au Timer B

16.2.6 Timer_B Interrupts

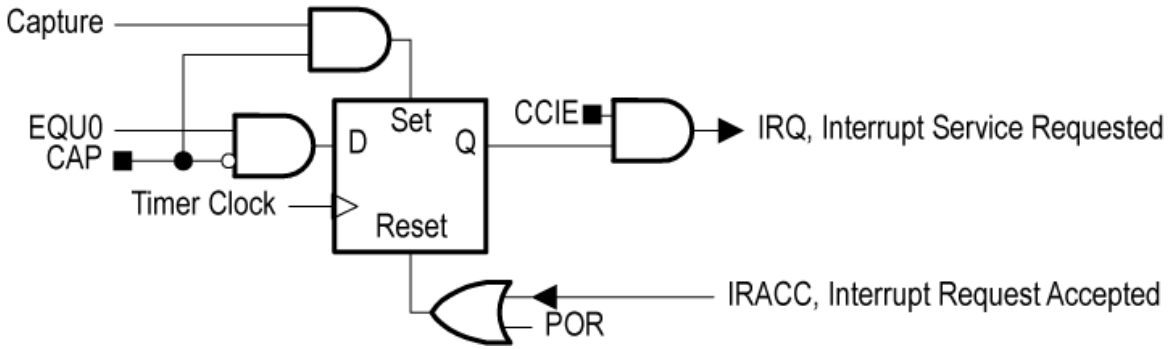
Two interrupt vectors are associated with the 16-bit Timer_B module:

- TBCCR0 interrupt vector for TBCCR0 CCIFG
- TBIV interrupt vector for all other CCIFG flags and TBIFG

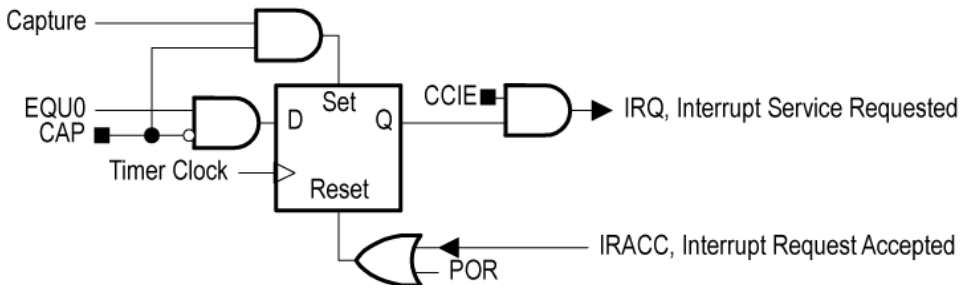
In capture mode, any CCIFG flag is set when a timer value is captured in the associated TBCCRx register. In compare mode, any CCIFG flag is set when TBR *counts* to the associated TBCLx value. Software may also set or clear any CCIFG flag. All CCIFG flags request an interrupt when their corresponding CCIE bit and the GIE bit are set.

TBCCR0 Interrupt Vector

The TBCCR0 CCIFG flag has the highest Timer_B interrupt priority and has a dedicated interrupt vector as shown in Figure 16–15. The TBCCR0 CCIFG flag is automatically reset when the TBCCR0 interrupt request is serviced.



Timer B7 : Caractéristiques principales



TBIV, Interrupt Vector Generator

The TBIFG flag and TBCCRx CCIFG flags (excluding TBCCR0 CCIFG) are prioritized and combined to source a single interrupt vector. The interrupt vector register TBIV is used to determine which flag requested an interrupt.

The highest priority enabled interrupt (excluding TBCCR0 CCIFG) generates a number in the TBIV register (see register description). This number can be evaluated or added to the program counter to automatically enter the appropriate software routine. Disabled Timer_B interrupts do not affect the TBIV value.

Any access, read or write, of the TBIV register automatically resets the highest pending interrupt flag. If another interrupt flag is set, another interrupt is immediately generated after servicing the initial interrupt. For example, if the TBCCR1 and TBCCR2 CCIFG flags are set when the interrupt service routine accesses the TBIV register, TBCCR1 CCIFG is reset automatically. After the `RETI` instruction of the interrupt service routine is executed, the TBCCR2 CCIFG flag will generate another interrupt.

Timer B7 : Caractéristiques principales

TBIV, Interrupt Handler Examples

The following software example shows the recommended use of TBIV and the handling overhead. The TBIV value is added to the PC to automatically jump to the appropriate routine.

The numbers at the right margin show the necessary CPU clock cycles for each instruction. The software overhead for different interrupt sources includes interrupt latency and return-from-interrupt cycles, but not the task handling itself. The latencies are:

- Capture/compare block CCR0 — 11 cycles
- Capture/compare blocks CCR1 to CCR6 — 16 cycles
- Timer overflow TBIFG — 14 cycles

The following software example shows the recommended use of TBIV for Timer_B3.

```
; Interrupt handler for TBCCR0 CCIFG.                    Cycles
CCIFG_0_HND
          ...       ; Start of handler Interrupt latency  6
          RETI                                            5


; Interrupt handler for TBIFG, TBCCR1 and TBCCR2 CCIFG.
TB_HND    ...                 ; Interrupt latency          6
          ADD    &TBIV,PC     ; Add offset to Jump table   3
          RETI                ; Vector  0: No interrupt    5
          JMP    CCIFG_1_HND  ; Vector  2: Module 1        2
          JMP    CCIFG_2_HND  ; Vector  4: Module 2        2
          RETI                ; Vector  6
          RETI                ; Vector  8
          RETI                ; Vector 10
          RETI                ; Vector 12

TBIFG_HND                     ; Vector 14: TIMOV Flag
          ...                 ; Task starts here
          RETI                                            5

CCIFG_2_HND                   ; Vector 4: Module 2
          ...                 ; Task starts here
          RETI                ; Back to main program      5

; The Module 1 handler shows a way to look if any other
; interrupt is pending: 5 cycles have to be spent, but
; 9 cycles may be saved if another interrupt is pending
CCIFG_1_HND                   ; Vector 6: Module 3
          ...                 ; Task starts here
          JMP    TB_HND       ; Look for pending ints     2
```

Timer B7 : liste des registres

Registre	Nom	Type de registre	Adresse	Etat initial
Registre de contrôle	TBCTL	Lecture / Ecriture	0x0180	Reset avec POR
Registre interne du Timer	TBR	Lecture / Ecriture	0x0190	Reset avec POR
Contrôle capture/comparaison 0	TBCCTL0	Lecture / Ecriture	0x0182	Reset avec POR
Capture/comparaison 0	TBCCR0	Lecture / Ecriture	0x0192	Reset avec POR
Contrôle capture/comparaison 1	TBCCTL1	Lecture / Ecriture	0x0184	Reset avec POR
Capture/comparaison 1	TBCCR1	Lecture / Ecriture	0x0194	Reset avec POR
Contrôle capture/comparaison 2	TBCCTL2	Lecture / Ecriture	0x0186	Reset avec POR
Capture/comparaison 2	TBCCR2	Lecture / Ecriture	0x0196	Reset avec POR
Contrôle capture/comparaison 3	TBCCTL3	Lecture / Ecriture	0x0188	Reset avec POR
Capture/comparaison 3	TBCCR3	Lecture / Ecriture	0x0198	Reset avec POR
Contrôle capture/comparaison 4	TBCCTL4	Lecture / Ecriture	0x018A	Reset avec POR
Capture/comparaison 4	TBCCR4	Lecture / Ecriture	0x019A	Reset avec POR
Contrôle capture/comparaison 5	TBCCTL5	Lecture / Ecriture	0x018C	Reset avec POR
Capture/comparaison 5	TBCCR5	Lecture / Ecriture	0x019C	Reset avec POR
Contrôle capture/comparaison 6	TBCCTL6	Lecture / Ecriture	0x018E	Reset avec POR
Capture/comparaison 6	TBCCR6	Lecture / Ecriture	0x019E	Reset avec POR
Vecteur d’interruption	TBIV	Lecture / Ecriture	0x011E	Reset avec POR

Timer B7 : registre de contrôle TBCTL

Timer_B Control Register TBCTL

15	14	13	12	11	10	9	8
Unused	TBCLGRPx		CNTLx		Unused	TBSSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

7	6	5	4	3	2	1	0
IDx		MCx		Unused	TBCLR	TBIE	TBIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	w-(0)	rw-(0)	rw-(0)

TBCLGRP TBCTL[14 13] : groupe
00 : Chaque TBCLx est chargé de manière indépendante
01 : TBCL1 et TBCL2 (TBCCR1 CLLDx bits contrôle la mise à jour)
TBCL3 et TBCL4 (TBCCR3 CLLDx bits contrôle la mise à jour)
TBCL5 et TBCL6 (TBCCR5 CLLDx bits contrôle la mise à jour)
TBCL0 indépendant
10 : TBCL1,TBCL2 et TBCL3 (TBCCR1 CLLDx bits contrôle la mise à jour)
TBCL4,TBCL5 et TBCL6 (TBCCR4 CLLDx bits contrôle la mise à jour)
TBCL0 indépendant
11 : TBCL1,TBCL2,TBCL3,TBCL4,TBCL5 et TBCL6 (TBCCR1 CLLDx bits contrôle la mise à jour)

CNTLx TACTL [12 11] : Taille du compteur
00 : 16 bits, TBR[max]=0xFFFF
01 : 12 bits, TBR[max]=0xFFF
10 : 10 bits, TBR[max]=0x3FF
11 : 8 bits, TBR[max]=0xFF

TBSSELx TBCTL[9 8] : Horloge de référence
00 : TBCLK
01 : ACLK
10 : SMCLK
11 : $\overline{\text{TBCLK}}$

Timer B7 : registre de contrôle TBCTL

Timer_B Control Register TBCTL

15	14	13	12	11	10	9	8
Unused	TBCLGRPx		CNTLx		Unused	TBSSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

7	6	5	4	3	2	1	0
IDx		MCx		Unused	TBCLR	TBIE	TBIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	w-(0)	rw-(0)	rw-(0)

TBSSELx TBCTL[9 8] : Horloge de référence
00 : TBCLK
01 : ACLK
10 : SMCLK
11 : $\overline{\text{TBCLK}}$

IDx TBCTL [7 6] : Diviseur de l'horloge de référence
00 : /1
01 : /2
10 : /4
11 : /8

MCx TACTL [5 4] : contrôle de mode
00 : Stop mode, le Timer est arrêté (low power)
01 : Up mode: Le Timer compte jusqu'à TBCL0
10 : Continuous mode : Le Timer compte jusqu'à TBCLx
11 : Up/down mode : Le Timer compte jusqu'à TBCL0 puis décompte jusqu'à 0000h

TBCLR TBCTL [2] : Reset du compteur : TBR=0, IDx=00 et MCx=00

TBIE TBCTL[B] : Contrôle les interruptions (TBIFG).
0 : interruptions désactivées
1 : interruption actives

TBIFG TBCTL[0] : flag d'état des interruptions
0 : pas d'interruption pendante
1 : interruption pendante

Timer B7 : registre interne TBRx

TBR, Timer_B Register

15	14	13	12	11	10	9	8
TBRx							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

7	6	5	4	3	2	1	0
TBRx							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

TBRx	Bits 15-0	Timer_B register. The TBR register is the count of Timer_B.

Timer B7 : registre de capture/comparaison TBCCRx

TBCCRx, Timer_B Capture/Compare Register x

15	14	13	12	11	10	9	8
TBCCRx							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

7	6	5	4	3	2	1	0
TBCCRx							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

TBCCRx	Bits 15-0	Timer_B capture/compare register. Compare mode: Compare data is written to each TBCCRx and automatically transferred to TBCLx. TBCLx holds the data for the comparison to the timer value in the Timer_B Register, TBR. Capture mode: The Timer_B Register, TBR, is copied into the TBCCRx register when a capture is performed.

Timer B7 : registre de contrôle capture/comparaison TBCCTLx

TBCCTLx, Capture/Compare Control Register

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	CLLDx		CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CMx TBCCTLx[15 14] : en mode capture
00 : pas de capture
01 : capture sur flanc montant
10 : capture sur flanc descendant
11 : capture sur les deux flancs

CCISx TBCCTLx[13 12] : sélection du signal d'entrée
00 : CCIxA
01 : CCIxB
10 : GND
11 : VCC

CLLDx TBCCTL[10 9] : Contrôle du chargement du registre de comparaison
00 : TBCLx chargé par écriture dans TBCCRx
01 : TBCLx chargé lorsque TBR vaut 0
10 : TBCLx chargé lorsque TBR vaut 0 (up ou continuous mode)
TBCLx chargé lorsque TBR vaut TBCL0 ou 0 (up/down mode)
11 : TBCLx chargé par écriture dans TBCCRx

CAP TBCCTL[8] : Contrôle du mode capture/comparaison
0 : mode comparaison
1 : mode capture

Timer B7 : registre de contrôle capture/comparaison TBCCTLx

TBCCTLx, Capture/Compare Control Register

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	CLLDx		CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

OUTMODx TBCCTLx[7 5] : Mode de sortie
Les modes 2, 3, 6 et 7 ne sont pas utilisés pour TBCL0 car EQUx=EQU0
000 : Sortie
001 : Set
010 : Toggle/reset
011 : Set/reset
100 : Toggle
101 : Reset
110 : Toggle/set
111 : Reset/set

CCI TBCCTLx[3] : Signal d'entrée de capture/comparaison

OUT TBCCTL[2] : Contrôle du bit de sortie pour le mode OUTMODx=000.
0 : sortie forcée à 0
1 : sortie forcée à 1

COV TBCCTLx[1] : dépassement en mode capture
(COV doit être remis à 0 par software).
0 : pas de dépassement
1 : détection d'un dépassement

Timer B7 : registre du vecteur d'interruption TBIV

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
r0	r0	r0	r0	r0	r0	r0	r0

7	6	5	4	3	2	1	0
0	0	0	0	TBIVx			0
r0	r0	r0	r0	r−(0)	r−(0)	r−(0)	r0

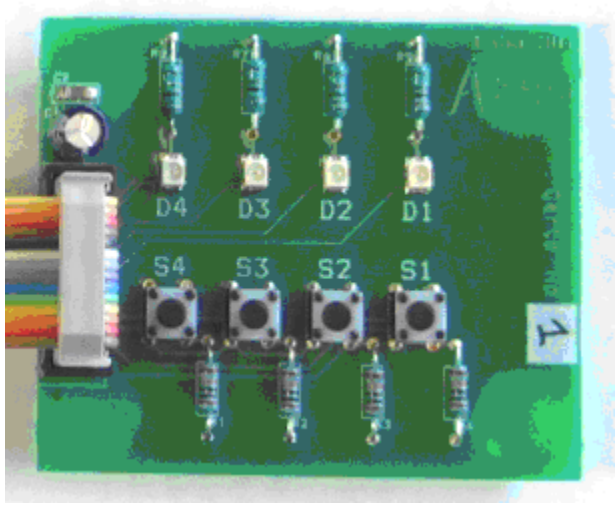
TBIVx Bits 15-0 Timer_B interrupt vector value

TBIV Contents	Interrupt Source	Interrupt Flag	Interrupt Priority
00h	No interrupt pending	–	
02h	Capture/compare 1	TBCCR1 CCIFG	Highest
04h	Capture/compare 2	TBCCR2 CCIFG	
06h	Capture/compare 3†	TBCCR3 CCIFG	
08h	Capture/compare 4†	TBCCR4 CCIFG	
0Ah	Capture/compare 5†	TBCCR5 CCIFG	
0Ch	Capture/compare 6†	TBCCR6 CCIFG	
0Eh	Timer overflow	TBIFG	Lowest

† MSP430x4xx devices only

Périphérique : les Timers

L'exemple suivant illustre un Timer_A dans un programme qui fait clignoter les quatre LEDs de P1.4 à P1.7 à une fréquence de 1 Hz. On utilise le ACLK et le VLO.



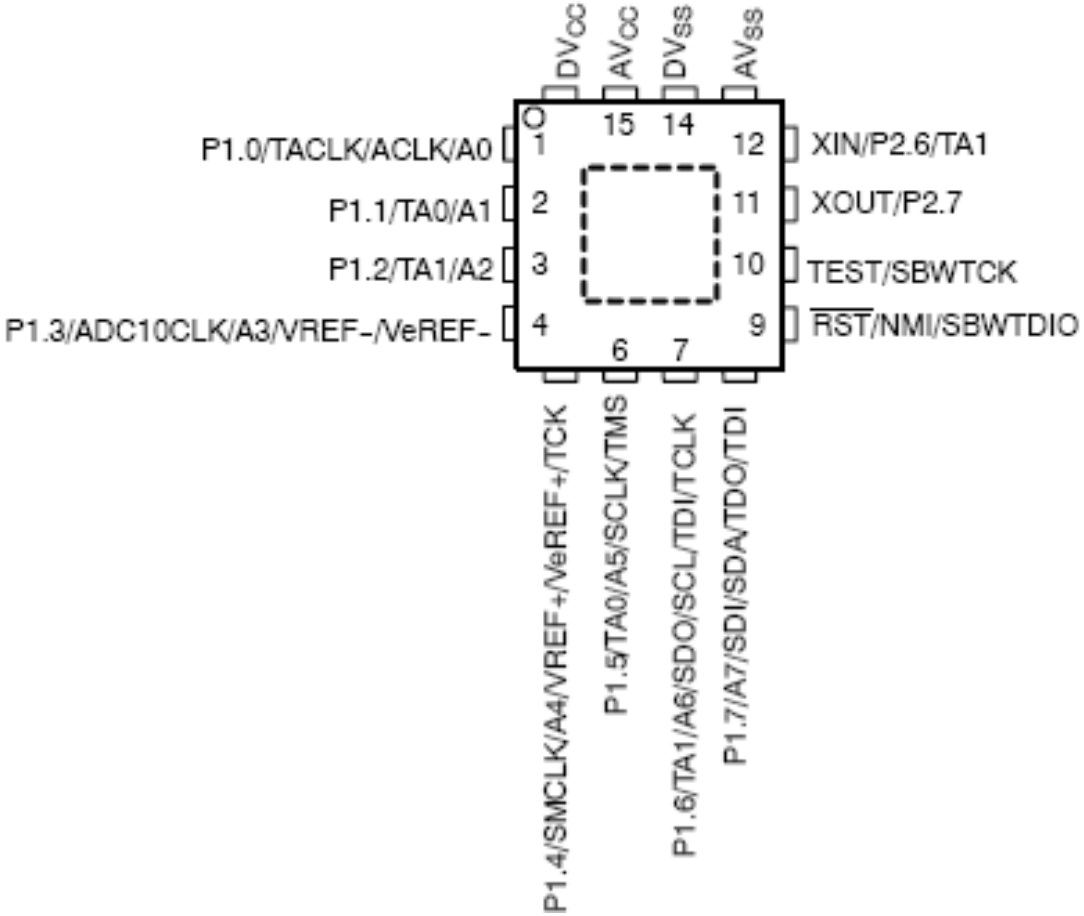

```
#include "io430.h"
#include "intrinsics.h"
int main( void )
{
  float freq;
  // Stop watchdog timer to prevent time out reset
  WDTCTL = WDTPW + WDTHOLD; // Stop WDT

  P1DIR |= 0xF0; // P1.4 à P1.7 output
  P1OUT = 0;
  TACCTL0 = CCIE;  // CCR0 interrupt enabled

  // Registres de contrôle de l'horloge
  BCSCTL1 |= DIVA_0;                        // ACLK/1
  BCSCTL3 |= LFXT1S_2;                      // ACLK = VLO
  freq = 2; // Hz
  TACCR0 = 12000 / freq - 1;
  TACTL = TASSEL_1 + MC_1;                  // ACLK, upmode
  __enable_interrupt();

  while (1);
}
// Timer_A TACCR0 interrupt vector handler
#pragma vector= TIMERA0_VECTOR
__interrupt void int_timerA(void)
{
  P1OUT ^= 0xF0;      // Toggle P1.x
}
```

Contrôle d’un moteur par un PWM

(Modulation de Largeur d’Impulsions - MLI)

On configure les timers pour produire sur les broches de sortie un signal pulsé avec le rapport cyclique (*duty cycle*) désiré.

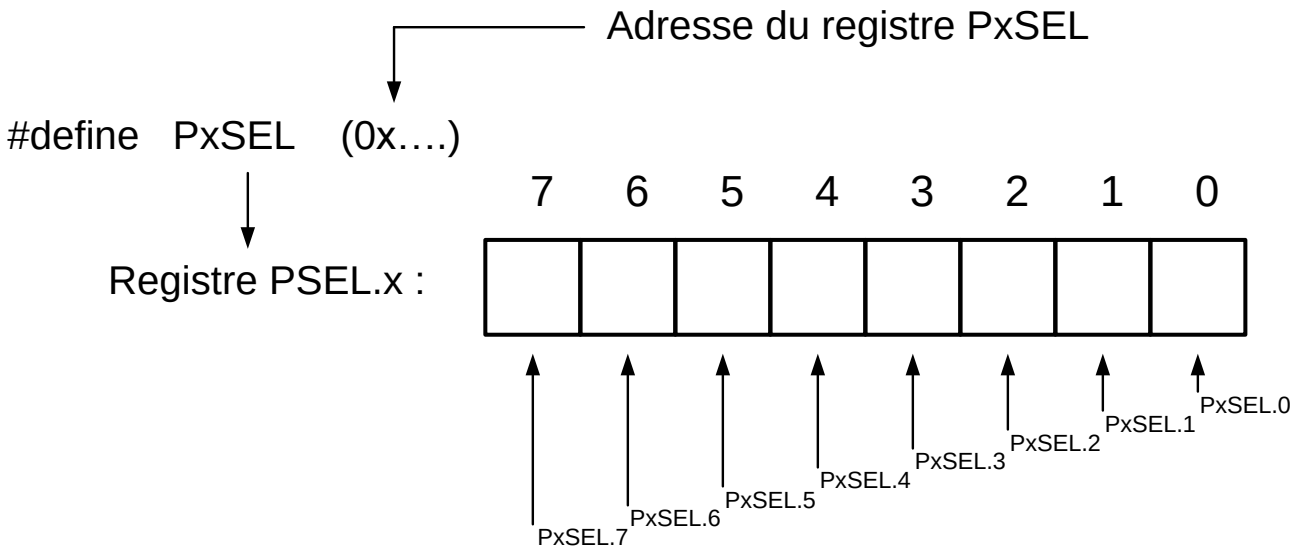
Pinout du MSP430F2012 avec des sorties pour les timers



Registre de sélection de fonction : PxSEL

Chaque broche du port P1 peut être multiplexées avec des modules d'autres périphériques. Par conséquent chaque bit des registres PxSEL permet de sélectionner un parmi plusieurs fonctions.

Bit = 0 : la broche est configurée en entrée – sortie
Bit = 1 : la broche est configurée pour une entrée – sortie d'un module spécifique



Registre de sélection de fonction : PxSEL

Assigner 1 à PxSEL.y ne définit pas automatiquement la direction (entrée ou sortie) de la broche y du port x.

Le bit correspondant du registre PxDIR.y doit donc être programmé en fonction des exigences du module pouvant être lié à la broche Px.y.

Par exemple

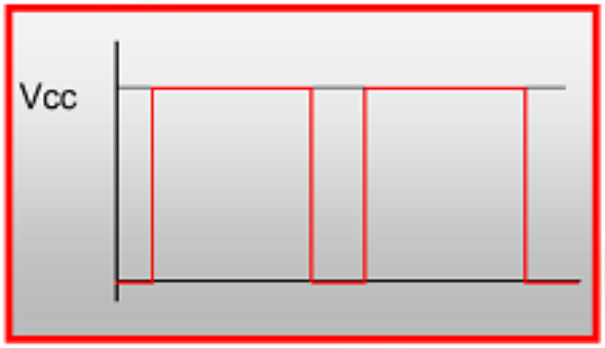
```
// sortie de Timer A1 sur la broche P1.2
P1SEL_bit.P1SEL_2=1;        // sélection de Timer A1 en sortie sur P1.2
P1DIR_bit. P1DIR_2=1;       // assigne P1.2 en sortie
```

ou

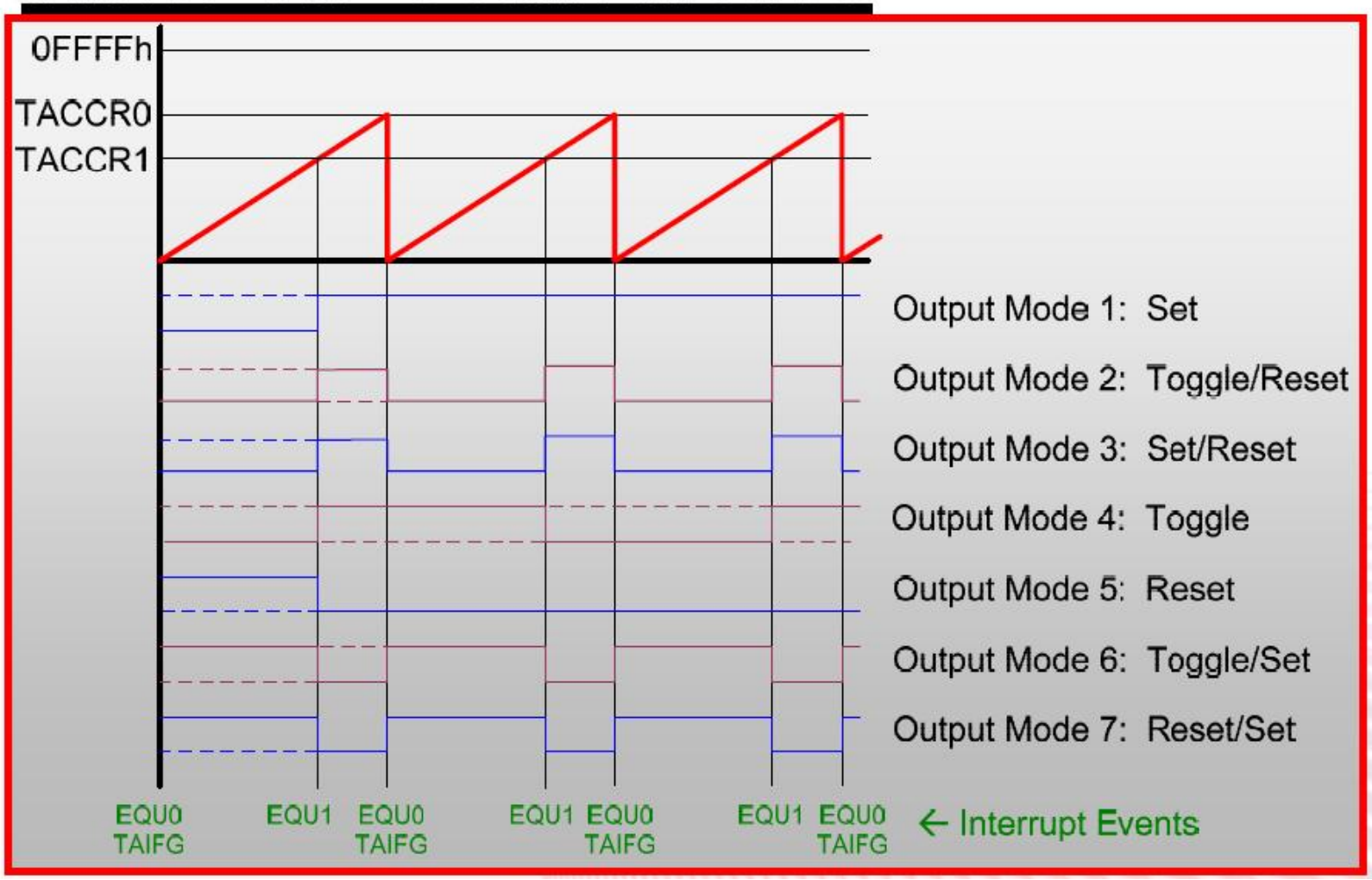
```
// sortie de Timer A1 sur la broche P1.2
P1SEL &=BIT2;               // sélection de Timer A1 en sortie sur P1.2
P1DIR &=BIT2;               // assigne P1.2 en sortie
```

Génération d'un PWM par le Timer_A1 sur la broche P1.2

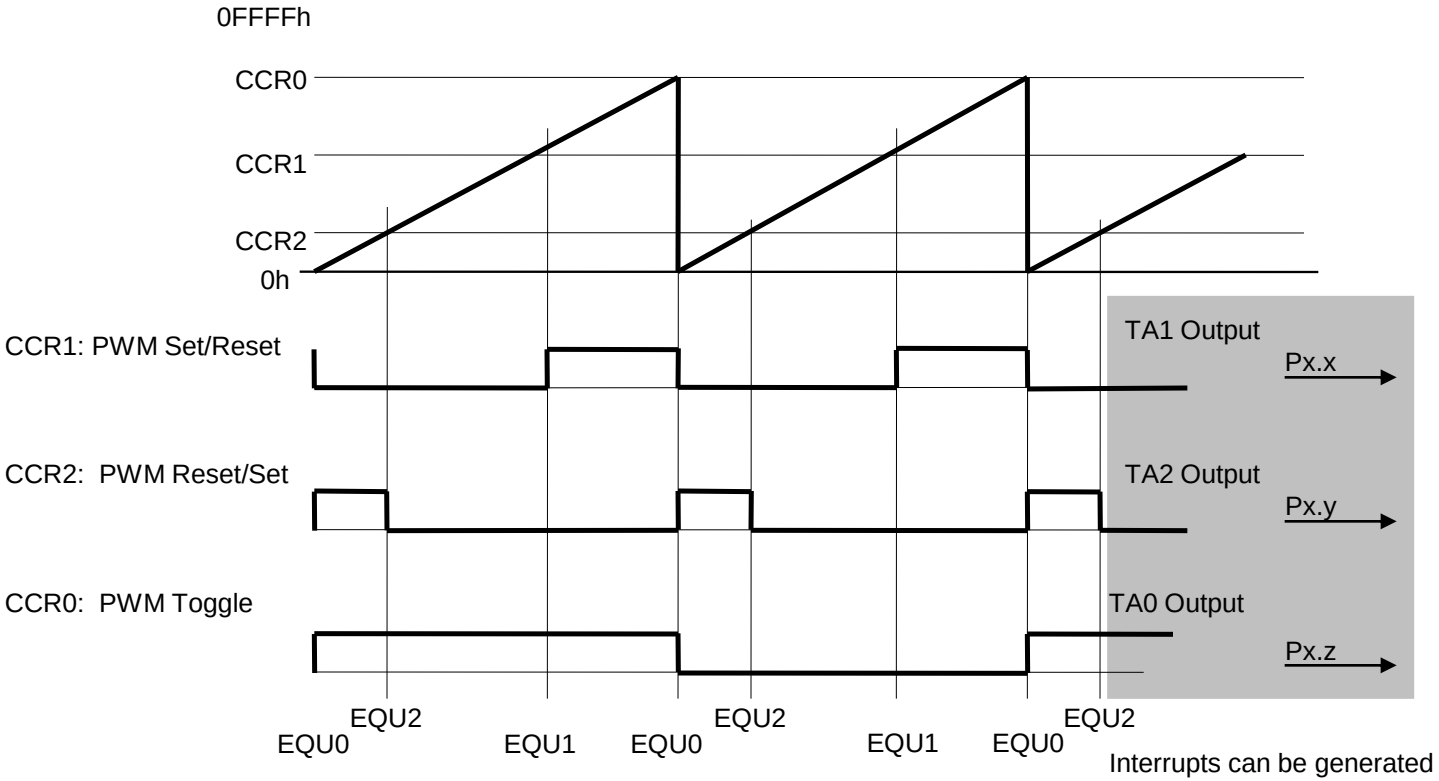
```
void main(void)
{
  WDTCTL = WDTPW + WDTHOLD;                    // Stop WDT

  P1DIR |= 0x04;                               // P1.2 output
  P1SEL |= 0x04;                               // P1.2 TA1 option

  TACTL = TASSEL_2 + MC_1;                     // SMCLK, up mode

  CCR0 = 512-1;                                // Période du PWM
  CCTL1 = OUTMOD_7;                            // CCR1 reset/set
  CCR1 = 51;                                   // CCR1 PWM duty cycle

  while (1);
}
```



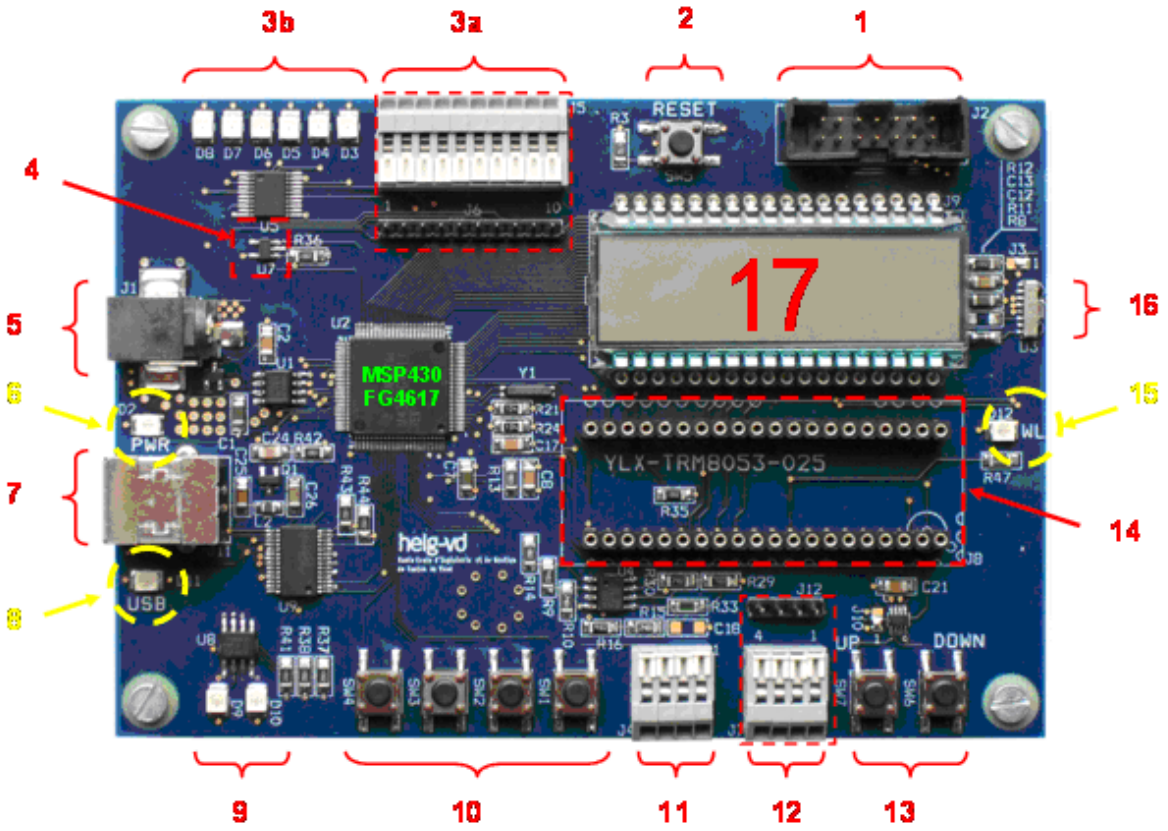
Timer A: Count Modes



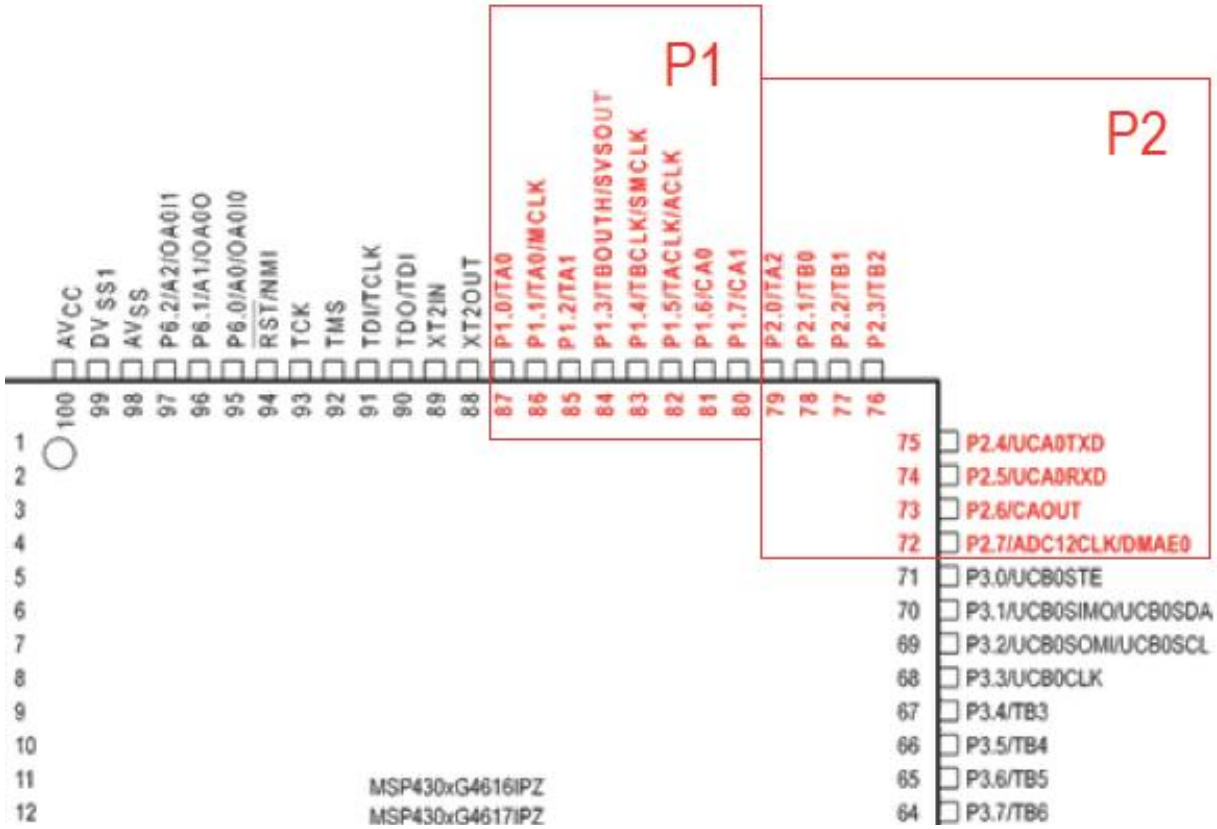
Selon le type de MSP430, trois ou plus différentes sorties PWM peuvent être générés par la sélection de CCR0, CCR1, CCR2, etc.



- Exemples pour la carte d'expérimentation IAI



Pinout du MSP430FG4617 avec des sorties pour les timers



L'exemple suivant illustre un simple chronomètre affichant minutes, secondes et centièmes de secondes:

```
#include "io430.h"
#include "intrinsics.h"
#include "LCD.h"
#include "stdio.h"

#define GO 1
#define STOP 0

int mm=0, ss=0, cc=0, mode=GO;

int main( void )
{
  char strDisp[8+1];
  float freq;
  // Stop watchdog timer to prevent time out reset
  WDTCTL = WDTPW + WDTHOLD; // Stop WDT

  P2DIR |= 0x01; // P2.0 output
  P2OUT = 0;
  CCTL0 = CCIE;  // CCR0 interrupt enabled

  // fréquence = 32768/CCR0 [Hz] port ACLK
  // fréquence = 32*32768/CCR0 [Hz] pout SMCLK
  freq = 100; // Hz
  CCR0 = 32*32768/freq - 1;
  TACTL = TASSEL_2 + MC_1;  // SMCLK, upmode

  P1IES &= 0x00;   // interrupt P1.x au flancs montants
  P1IE  |= 0x01;   // enable interrupt P1.0
  P1IFG  = 0;
```

Suite du programme *main.c* simple chronomètre affichant minutes, secondes et centièmes de secondes:

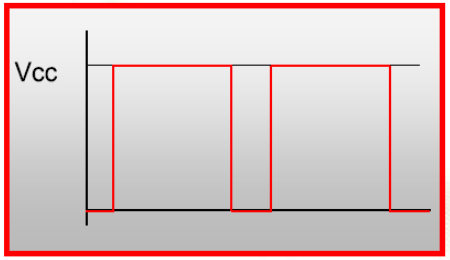
```
. . .

  __enable_interrupt();

  LCD_init();   // initialise et efface LCD

  while (1)
  {
    if (mode == GO)
    { sprintf(strDisp, "%02d %02d %02d", mm, ss, cc);
      LCD_print(strDisp);
    }
  }
}
```

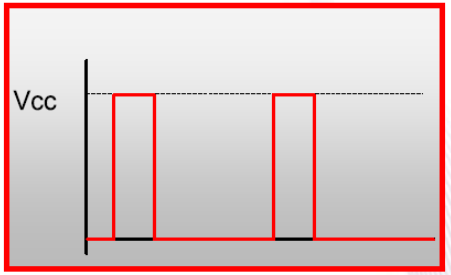
Et ensuite … les routines d'interruption:

- **Interruption par le timer**
- **Interruptions par le port 1**

```
// Timer_A TACCR0 interrupt vector handler

#pragma vector= TIMERA0_VECTOR
__interrupt void int_timerA(void)
{
  cc++;  // ajouter un centième de seconde
  if (cc == 100)
  { cc = 0;
    ss++;  // ajouter une seconde
    P2OUT ^= 0xCF;      // Toggle P2.x toutes les demi-secondes
  }
  if (cc == 50)
    P2OUT ^= 0xCF;      // Toggle P2.x
  if (ss == 60)
  { ss = 0;
    mm++;  // ajouter une minute
  }
  if (mm == 60)  mm=0;
}
```

```
// routine d'interruption de P1
// si on presse P1.0 une fois le chrono s'arrête, si on re-presse il recommence

#pragma vector=PORT1_VECTOR
__interrupt void Port1_ISR (void)
{
  if (P1IFG == 0x01)
  { if (mode == GO)
    { CCTL0 = 0;     // CCR0 interrupt disabled
      mode = STOP;
    }
    else
    { CCTL0 = CCIE;  // CCR0 interrupt enabled
      mode = GO;
    }
    P1IFG = 0;
  }
}
```





Génération de deux PWM

(modulation de largeur d'impulsions - MLI)

par le Timer_B sur les broches P2.2 et P2.3

```
int main( void )
{
  // Stop watchdog timer to prevent time out reset
  WDTCTL = WDTPW + WDTHOLD;

  TBCTL = TBSSEL_2 + TBCLR;              // SMCLK, Clear Tar
  TBCCR0 = 512-1;                        // PWM Period
  TBCTL |= MC_1;                         // Start Timer_A in up mode

  TBCCTL1 = OUTMOD_7;                    // CCR1 reset/set pour TB1
  TBCCR1 = 32;                           // CCR1 PWM duty cycle sur port 2.2, max 511

  TBCCTL2 = OUTMOD_7;                    // CCR2 reset/set pour TB2
  TBCCR2 = 64;                           // CCR2 PWM duty cycle sur port 2.3, max 511

  P2DIR |= 0xCF;                         // P2.x  output
  P2SEL |= 0x0C;                         // Sélectionne P2.2-3 sur TB1/TB2
  P2OUT |= 0xC0;  // signal continu sur les deux derniers LEDs, pour comparaison


  while(1);
}
```