Timer

1 Utilité

Un timer sert de base temps qui permet

- de générer des évènements au bout d'une certaine durée basée sur cette base de temps
- de compter une durée de temps entre deux évènements

Exemple 1 : pour fonctionner un chronomètre ou une montre nécessite un timer pour incrémenter une variable, l'heure si c'est une montre ou la durée si c'est une chronomètre.

- Il nécessite de générer une évènement toute les secondes (ou centièmes de seconde selon la précision désirée).
- Un évènement se produit à intervalle régulier (ici les secondes)
- Lorsque cet évènement se produit, le logiciel modifie l'heure courante (variable) et l'affichage sur l'écran

Exemple 2 : votre application embarquée souhaite faire clignoter une led avec une période de 1 seconde et un rapport cyclique de 1/2. Le rapport cyclique est défini par le temps ou le signal de commande est à l'état haut sur la période. Pour une led cela corresponde au rapport où la led est 'ON' sur la période définie par la durée 'ON' plus la durée 'OFF'. Ici un rapport cyclique de 1/2 et une période de 1 seconde signifie : une demis seconde la led est allumée ('ON'), une demi seconde la led est éteinte ('OFF').
Il est possible d'utiliser un Timer qui mettra à jour une variable, ou plus généralement un drapeau (flag) d'une variable (un bit de la variable) toute le 1/2 seconde. Le logiciel pourra se contenter de basculer le signal de sortie lorsque cette variable change). En pseudo code cela donnerai :

LED à ON
attendre que le flag passe à 1
LED à OFF
repositionner le flag à 0
attendre que le flag passe à 1
LED à ON
….

Ce type d'attente active qui consiste à attendre qu'un drapeau change d'état s'appelle la **scrutation**. Cette méthode est très utilisée dans le monde des pilotes de périphériques.

Le rôle du timer ici est de générer un évènement à intervalle de temps régulier (activation du flag) de manière autonome (sans opération logicielle). Pour un timer matériel, le drapeau sera un bit d'un registre de périphérique mappé en mémoire.

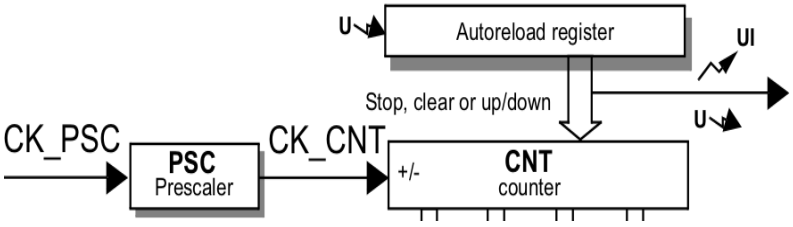
2 Principe de fonctionnement du timer de STM32

2.1 Coeur de timer

Le coeur du timer est constitué de 3 éléments

- un compteur appelé **CNT**
- un registre de base de comptage appelé **ARR** pour 'auto-reload register'
- un prescaler appelé **PSC**

Schéma tiré de la doc :



1.1. Le registre compteur CNT est l'élément central. C'est un simple compteur numérique qui s'incrémente ou se décrémente au rythme d'un signal d'horloge appelée **CK_CNT**. Ce signal d'horloge est la base de temps, on aura CNT = CNT+1 à chaque période de ce signal.
Il convient, s'il on souhaite avoir une base de temps correcte de bien connaître la fréquence de ce signal d'horloge.

1.2. Associé au compteur le registre ARR permet de fixe les bornes de comptage/décomptage, à savoir :

- Si le compteur compte ('upcounting mode'), CNT comptera de 0 à ARR (CNT prériodes). Ainsi lorsque CNT sera égal à ARR, le coup suivant il passera à 0 et recommencera à compter (CNT = CNT+1). L'évènement CNT==ARR → CNT==0 est appelé **overflow**
- Si le compteur décompte ('downcounting mode', il décomptera de ARR à 0. Ainsi lorsque CNT sera égal à 0, au coup d'horloge suivant il passera de 0 à ARR et recommencera à décompter (CNT=CNT-1) L'évènement CNT==0 → CNT==ARR est appelé **underflow**

Lorsque qu'il y a overflow (ou underflow) un évènement est généré appelé 'update event' (**UEV** ou U).

Il existe également un mode appelé 'centered aligned mode' où le compteur compte de 0 a ARR-1 puis décompte de ARR à 0. Un évènement UEV est généré lors du passage de ARR-1 à ARR et de 0 à 1.

1.3. Le préscaler PSC permet de modifier la fréquence du signal CK_CNT. Ainsi la fréquence du signal d'horloge CK_CNT est la fréquence du signal d'horloge CK_PSC divisée par PSC+1.

Exemple :

pour revenir à l'exemple d'un évènement généré toute les 1/2 secondes. Si l'horloge est une horloge à 42MHZ dérivée d'un quartz. Il est possible de générer un évènement (UEV) toute les 1/2 seconde en se plaçant en mode upcounting et en plaçant la valeur 42E6/2 -1 = 21E6 -1 dans le registre ARR. Ainsi l'evènement UEV sera produit toutes les demi-seconde. Cet évènement se traduit par la mise à 1 du bit UIF (bit 0) du registre 'status register' (TIMx_SR) du timer.

Note :

- Le drapeau UIF doit être remis à 0 par le logiciel (écriture d'un 0 sur ce bit)
- Certain TIM ont leur compteur CNT sur 32 bits, d'autres sur 16 bit. Dans ce dernier cas, la valeur 21E6 ne peut pas être codée sur 16 bit, il faut alors utiliser le prescaler pour réduire une première fois la fréquence (exemple : 10E3 dans le prescaler et 21E3-1 dans ARR)

2.2 Horloge

L'horloge permettant d'alimenter le prescaler (CK_PSC) peut être choisi parmi

- Horloge interne (**CK_INT**) qui est l'horloge qui alimente le fonctionnement de tout le timer, elle correspond à l'horloge du bus périphérique auquel est connecté le timer.
- Horloge externe (mode 1) : c'est une pin externe (Tix) configurée en entrée qui alimente l'horloge CK_PSC
- Horloge externe (mode 2) : c'est encore une pin externe (ETR) qui alimente l'horloge (avec quelque options de configuration en plus)
- Signal interne issu d'un autre timer (Internal trigger inputs (ITRx)), permet de chaîner les timers.

2.3 Capture/compare Mode

Les timers du STM32 disposent également de 2 à 4 registres appelés capture compare mode register (**CCRx**). Ces registres permettent :

- capturer la valeur du compteur principal CNT (i.e. copier la valeur courante du compteur) sur un évènement
- comparer la valeur courante du compteur principal CNT à la valeur du regitre CCMx pour générer une action

Dans le premier cas (input capture mode ou PWM input mode), les évènements sont typiquement le passage de 0 à 1 ou de 1 à 0 (ou les deux) d'un signal (externe (pin) ou interne). Destiné à mesurer des signaux numérique (durées, périodes, fréquences)

Dans le second cas (output compare mode ou PWM mode), les actions sont la mise à 0 ou à 1 ou bien encore le changement d'état d'une pin. Cela permet de générer automatiquement des signaux sur ces pins.

Exemple d'utilisation :

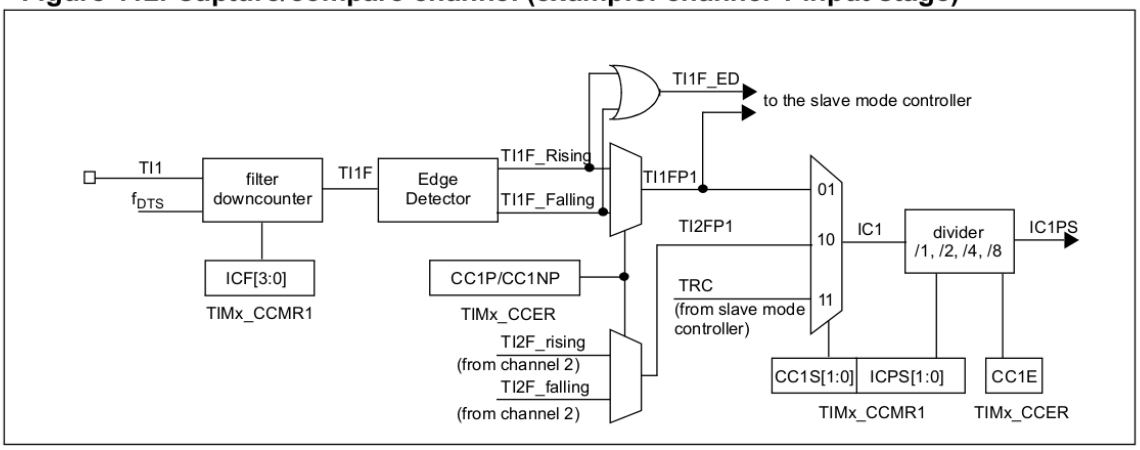
- On souhaite calculer la période d'un signal externe, on peut configurer le timer pour mémoriser la valeur du compteur CNT dans un des registre CCRx lorsqu'il y a passage du signal externe de l'état bas à l'état haut.
- On souhaite générer un signal pour piloter la vitesse d'un moteur. Cette vitesse est gérée par un signal ON/OFF et est proportionnelle au rapport cyclique (temps ON sur temps ON + temps OFF). On utilise le principe de comparaison : Le compteur principale tourne en boucle sur une durée égale à la période, le signal de sortie est généré automatiquement : état haut tant que le compteur est inférieur à la valeur présente dans le registre CCRx puis bas jusqu'à la remise à zéro (valeur de ARR) du compteur principal

2.3.1 Input capture mode

Dans ce mode la valeur du compteur CNT, est copié dans un des registre CCRx sur changement d'état d'un signal. On peut prendre l'exemple du chanel 1 permettant de capture CNT dans CCR1 (ou CCR2). L'étage d'entrée est présenté dans le schéma ci-dessous :

Figure 112. Capture/compare channel (example: channel 1 input stage)

Le



signal d'entrée, peut être filtré (pour gérer d'éventuelle oscillation sur un changement d'état) puis on peut sélectionner si l'on souhaite une capture sur un front montant (rising) ou un front descendant (falling) ou les deux, puis un prescaler (pour capturer à chaque front ou 1 front sur 2, 4 ou 8). Le signal IC1PS permettra d'effectuer la capture proprement dite.

2.3.2 Output compare mode

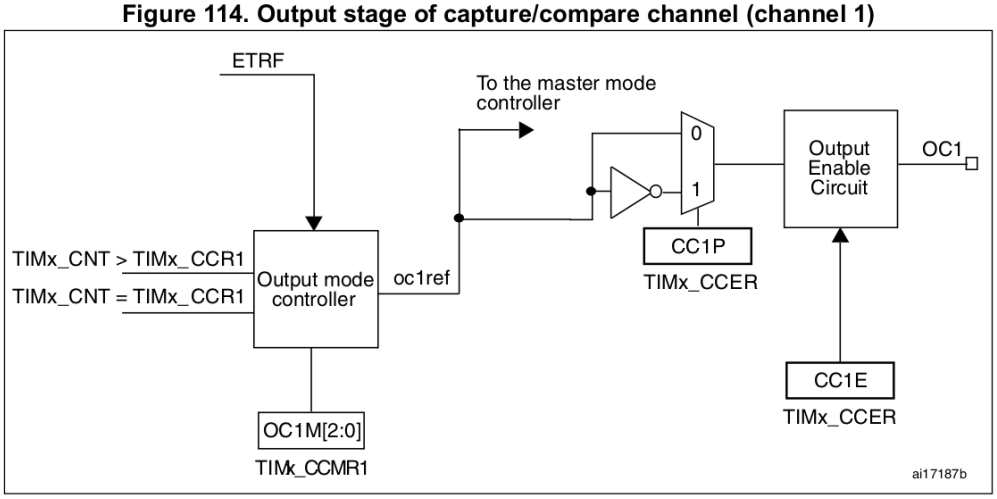
Ce mode est utilisé pour contrôler la génération d'un signal de sortie ou pour indiquer d'une période de temps est arrivée à échéance.

Dans ce mode le compteur principal CNT est comparé à la valeur d'un des registre CCRx. Lorsqu'il y a égalité (match),le signal interne OCxREF peut être placé à l'état haut ou à l'état bas ou être inversé (toggle). Ce signal (ou son inverse) peut alors être utilisé pour piloter une pin externe du canal associé (OCx) et ainsi générer un signal sur cette pin.

Il est par exemple possible que le signal OCx soit maintenu à l'état haut tant que le compteur principal CNT est inférieur à la valeur présente dans CCRX et maintenu à l'état bas tant qu'il est supérieure à la valeur présente dans CCRx.

Schéma de l'étage se sortie (exemple pour la OC1 (TIMX_CH1)) :

Figure 114. Output stage of capture/compare channel (channel 1)



2.3.3 shadow register

Les registre ARR, et CCR sont associé à des registre dit 'shadow'. Ces registres servent de tampon permettant de modifier indépendamment les registres associé (ARR et CCR) sans forcément modifier immédiatement le comportement du timer. Par exemple pour le registre ARR en mode upcounter, C'est le 'shadow register' qui est comparé au compteur principal CNT. Si l'on souhaite changer la valeur de remise à zéro du compteur principal, on ne modifie que le registre ARR, le registre shadow est chargé au moment où il y a overflow et l'évènement UEV. Le shadow register est alors chargé avec la nouvelle valeur.

Pour les registre CCR, la capture est effectuée dans le shadow register puis copié dans le registre CCR. En mode comparaison, la valeur dans le CCR est copié dans le shadow register qui est ensuite comparé au compteur.

Schéma complet pour le channel1 du timer

Figure 113. Capture/compare channel 1 main circuit

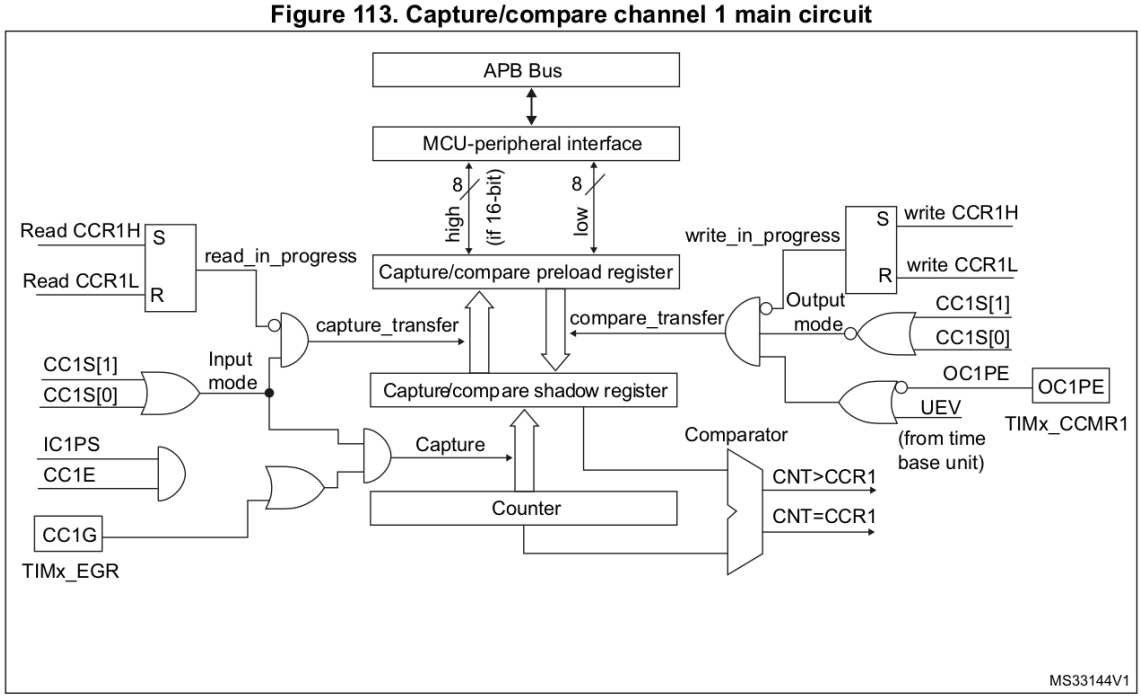
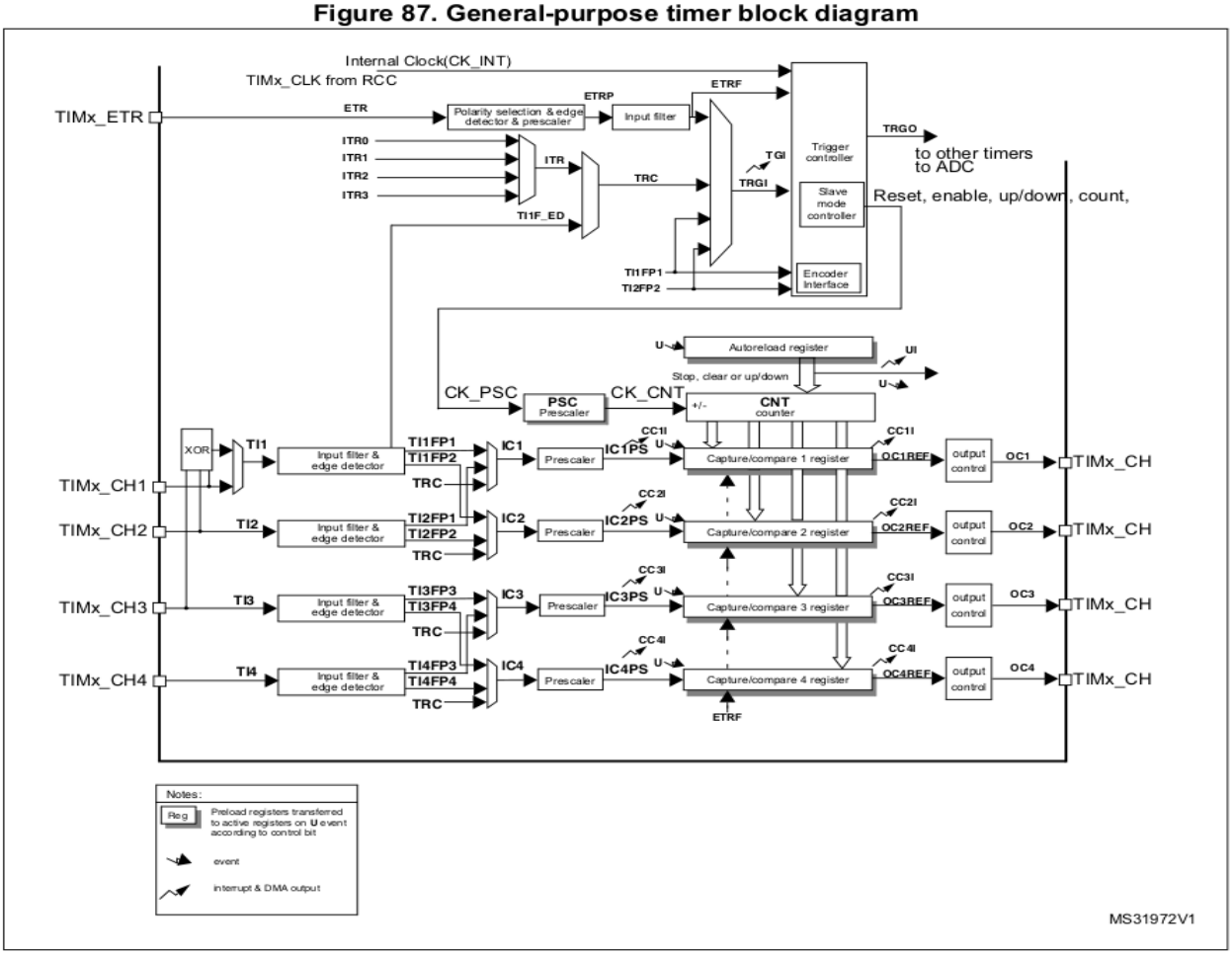


Schéma complet du timer :

Figure 87. General-purpose timer block diagram



2.4 Drapeaux

Chaque évènement (overflow, underflow, capture, match, …) activent des drapeaux dans le registre d'état des timers ('status register'). Ces drapeaux peuvent être utilisé pour :

- effectuer du polling ou scrutation pour attendre l'évènement : boucle d'attente active en attendant que le drapeau soit activé
- générer des requêtes d'interruptions (vu plus tard)

2.5 Autres modes et exemple d'utilisation

D'autres modes d'utilisation sont possibles pour les timers, certains directement dérivés des modes capture et compare :

- PWM (pulse width modulation) en entrée (mesure de période ou d'impulsion) ou sortie conversion numérique analogique sous forme d'un signal dont on modifie le rapport cyclique en fonction des valeurs présente dans les registre CCR
- mesure de roue codeuse (codeur incrémental) : utilisé pour déterminer si le sens et la vitesse de rotation d'une roue.
- External trigger : le compteur peut être mis à zéro, arrêter ou démarrer un compteur sur un signal externe dit 'trigger'

Des exemples de fonctionnement et de configuration sont donnés dans la documentation « STM32F401 reference manual ».