GIF-3002 Périphériques, Timers, PWM, DAC et ADC

Ce document présente les timers, les PWM, les DACs et les ADCs. Il s'inscrit dans la séquence de cours sur les périphériques communs des systèmes microprocesseurs.

1.1 Timers

Les minuteries (timers) sont des composantes fondamentales dans les systèmes microcontrôleurs.

Tous les timers sont des valeurs qui augmentent ou décrémentent en fonction d'une horloge et, parfois, d'une transition sur une broche d'entrée/sortie, c'est-à-dire un compteur d'évènement. Un timer qui incrémente incrémentera jusqu'à ce qu'il atteigne une valeur prédéfinie ou jusqu'à ce qu'il fasse un débordement, ce qui est plus fréquent. Un timer qui décrémente décrémentera jusqu'à ce qu'il atteigne une valeur ou lorsqu'il atteindra 0, ce qui est plus fréquent.

Lorsque les timers expirent (atteignent 0, débordent ou atteignent une valeur prédéterminée), les timers peuvent générer des interruptions. Lorsqu'ils expirent, ils peuvent soit arrêter, soit être re-initialisés (reload) automatiquement avec une valeur prédéterminée.

L'horloge des timers est souvent une horloge dérivée de l'horloge du microprocesseur. Il s'agira habituellement de celle du microprocesseur divisée par une constante additionnée à un registre de contrôle du timer (horloge timer = horloge système / (K + RegistreTimerClock).

La plupart des timers ont un mode capture : quand un évènement se produit sur une broche prédéterminée du microcontrôleur, la valeur du timer est capturée dans un registre réservé à cette fin.

Bref, il y a plusieurs façons d'utiliser une timer :

- Libre (free run) : le timer est incrémenté ou décrémenté librement et il continue même lorsqu'il déborde.
- Un coup (one shot) : le timer est programmé pour compter un temps fini, puis s'arrêter et générer une interruption.
- Générateur d'horloge (périodic) : le timer est programmé pour généré une interruption à intervalle fixe.

1.1.1 Programmation de Timers

Souvent les timers ont un ou plusieurs registres qui permet de configurer le timer : déterminer son mode d'opération, la direction des comptes, déterminer la nature du compteur (horloge ou évènements) et plus. Les timers ont aussi des registres pour déterminer la fréquence d'horloge du timer, pour déterminer les valeurs de reload du timer, pour décoller ou arrêter le timer ou déterminer quand et si le timer déclenchera des interruptions…

Un élément important à considérer lorsque l'on programme un Timer est le temps maximal et le temps minimal que peut mesurer le timer. Par exemple, un timer 16 bits, sur un microprocesseur ayant une horloge de 8192kHz (~8MHz) pour les périphériques et un diviseur d'horloge sur 8 bits (prescale, 0 à 255) pourra compter de 0 à 65535 avec un intervalle entre les unités variant de ~0.12us (1/8192kHz) à ~31.13us (255/8192kHz). Ce Timer ne peut pas avoir une période plus grande que 2.04ms (255/8192kHz*65536).

Les timers roulent de manière autonome. Cela peut causer des erreurs dans vos programmes. Par exemple, si on veut effectuer une tacĥe lorsque la valeur d'un timer est égale à X, faire une comparaison d'égalité (==) de la valeur du timer avec X dans le programme principal peut donner un résultat désastreux : le timer peut incrémenter deux fois avant que vous ne fassiez la comparaison et l'égalité peut ne jamais devenir vraie… Lorsqu'on compare des valeurs de timers avec des variables dans le main, il faut utiliser des inégalités (>=, >, <=, <)!

De par leur nature, les timers débordent. Ne pas en tenir compte est courir à la catastrophe. Par exemple, si on compare la valeur d'un timer 16-bits avec un entier sur 32-bits, il est possible, si l'entier 32-bits devient supérieur à 65536 que la condition reliée au timer ne se produise jamais… La meilleure façon de mesurer un intervalle de temps avec des timers est d'utiliser des variables ayant toutes la même taille et de faire des soustractions : en notation complément 2, le résultat de la soustraction sera correct, même si le timer a débordé.

1.1.2 Exemple de Timer : le timer du 8051

La figure ci-dessous illustre un timer du 8051, une architecture de microcontrôleur encore très présente dans l'industrie malgré son âge[1]. Le timer du 8051 est présenté ici comme exemple, en raison de sa simplicité relative.

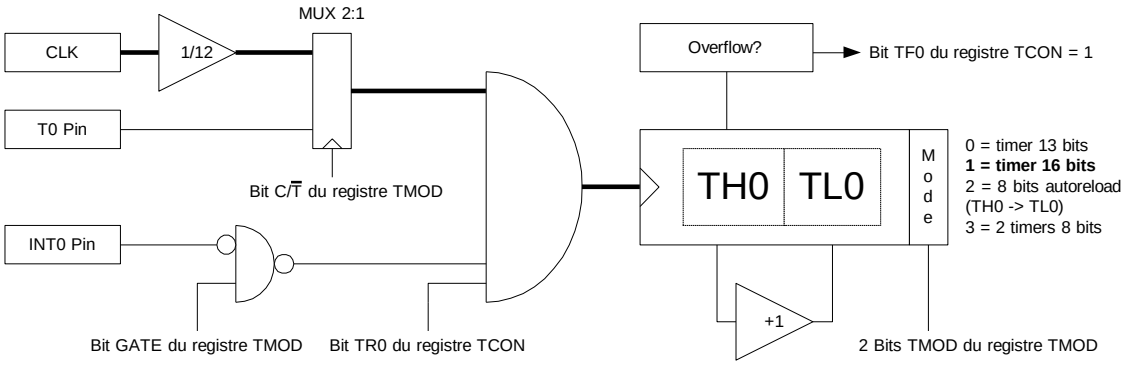


Figure 1 – Timer du 8051

[1] Le 8051 d'Intel est apparu officiellement en 1980. Il a été très apprécié de l'industrie et plusieurs cœurs de microcontrôleurs ont été construits avec son architecture (exemple : 8052!). En 2008, ~20% des microcontrôleurs avaient une architecture basée sur le 8051, mais ce pourcentage est en baisse rapide.

1.1.2.1 Modes d'opération

Le timer du 8051 a 4 modes d'opération: timer 13 bits, timer 16 bits, timer 8 bits avec auto-reload et 2 timers 8 bits indépendants. Le mode d'opération est déterminé par les 2 bits TMOD du registre TMOD.

Dans les modes 13 bits et 16 bits, le timer part de 0 ou de la valeur écrite dans TH0, TL0. Il est ensuite incrémente jusqu'à ce qu'il atteigne la valeur maximum (2^13 – 1 = 8191 sur 13 bits et 65535 sur 16 bits). Lorsque la valeur maximum est atteinte, le bit TF0 (Timer overFlow) du registre TCON devient 1. Il est aussi possible de déclencher une interruption dans ce cas.

En mode auto-reload, TL0 (la partie la moins significative de la valeur du timer 0) est incrémenté jusqu'à franchir 255 (sur 8 bits). Après 255, un overflow, et, possiblement, une interruption, TL0 est automatiquement rechargé avec la valeur de TH0.

En mode 2 timers 8 bits, le 8051 a deux timers relativement indépendant, mais de 8 bits.

1.1.2.2 Registres de contrôle

Quel que soit le mode d'opération, le timer peut être incrémenté selon une ou l'autre des conditions suivantes :

1) Le timer est incrémenté automatiquement à tous les 12 coups d'horloge du microcontrôleur
2) Le timer est incrémenté lorsqu'une transition montante apparaît sur la broche T0.

Par ailleurs, le timer incrémentera uniquement si les conditions suivantes sont respectées :

- Le timer est décollé : le bit TR (timer running) du registre TCON est 1.
- Il n'y a pas de GATE sur le signal incrémentant l'horloge. La GATE est soit désactivée, soit activée, mais ouverte.

Le terme GATE se traduit littéralement par barrière. En électronique, une GATE est une porte logique ET accompagnée d'un signal qui peut empêcher la transmission d'un signal de l'autre côté de la porte (barrière ouverte –> X ET 1 = X, barrière fermée –> X ET 0 = 0)

1.1.3 Exemple de timer : le timer du STM32F407

Les timers de la datasheet du STM32F407 seront présentés en classe. Ces timers supportent tous les éléments mentionnés ci-dessus.

1.2 Pulse Width Modulator (PWM)

Un PWM génère des ondes rectangulaires dont la fréquence et le duty cycle (duty cycle = Ton/T sur la figure ci-dessous) sont déterminées par des registres.

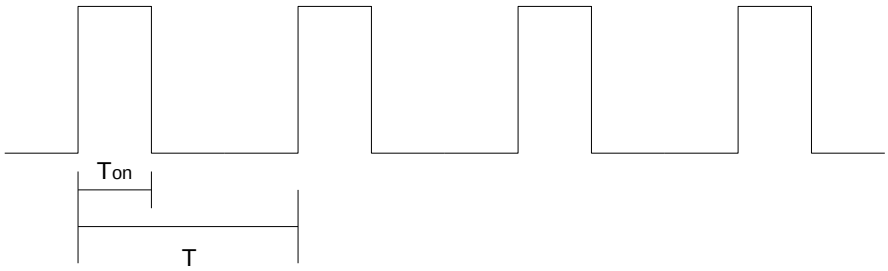


Figure 2 - Sortie d'un PWM

Un PWM est souvent configuré par deux registres qui sont, soit la fréquence et le duty cycle, soit le temps HIGH et le temps LOW.

Les PWMs sont utilisés dans plusieurs applications : générer des signaux de communication, générer des tensions DC, contrôler la puissance fournie à des appareils, les moteurs, etc.

1.3 Digital to Analog Converter (DAC)

Un DAC convertit un signal digital en signal analogique. Les deux propriétés les plus importantes des DACs sont la précision (le nombre de bits du DAC) et la vitesse du DAC (le nombre d'échantillons différents que peut produire le DAC par seconde.

Habituellement, la sortie d'un DAC est déterminée par l'entrée du DAC (digitale) et un voltage de référence qui est la valeur maximum que peut produire le DAC. Si on suppose un DAC 10bits par exemple avec une tension de référence de 2.5V, mettre une entrée de 256 (256 sur 1024 = 2^10) produira une tension de sortie du DAC de 0.75V (2.5V * 256 / 1024).

D'un point de vue matériel, le signal de sortie d'un DAC est souvent amplifié en voltage et en courant. Les DACs ne fournissent habituellement pas beaucoup de puissance et leur voltage de sortie est très souvent limité par le voltage d'alimentation du microcontrôleur.

Par ailleurs, la tension de référence et la tension d'alimentation du DAC sont souvent filtrées pour réduire le bruit à la sortie du DAC. Un DAC 12 bits avec une tension de référence de 3.3V peut, en théorie, faire varier le voltage de sortie avec des pas de $3.3V/2^{12}$. Toutefois, dans la pratique, diverses raisons dont le bruit, font en sorte que le nombre de bits effectifs du DAC est plutôt autour de 10 bits…

D'un point de vue logiciel, il faut habituellement écrire des registres de contrôles pour activer le DAC, puis, simplement, écrire le registre de sortie du DAC pour changer la

tension sur la broche dédiée au DAC (il faut évidemment aussi écrire un registre de « fonction » de la broche pour que la broche soit contrôlée par le DAC).

Enfin, le DAC est souvent associé au contrôleur de DMA. Si on veut générer une onde spécifique avec un DAC (comme un sinus ou un triangle), on peut configurer un timer pour générer une interruption période à fréquence d'échantillonnage voulue. Puis, il suffit d'envoyer un point de l'onde spécifique à chaque interruption. Ces opérations sont complexes et peuvent être remplacées par le contrôleur de DMA : on prépare un tableau de points en mémoire représentant l'onde à générer et on associe le contrôleur de DMA avec un timer…

1.3.1 Le DAC sommateur

Les premières versions de DAC étaient des DAC avec ampli-ops sommateurs. Il s'agit d'un circuit avec amplificateur opérationnel classique dans lequel la sortie est la somme pondérée de toutes les entrées. On pondère chaque entrée pour représenter un bit différent en ayant un facteur 2 entre chaque résistance :

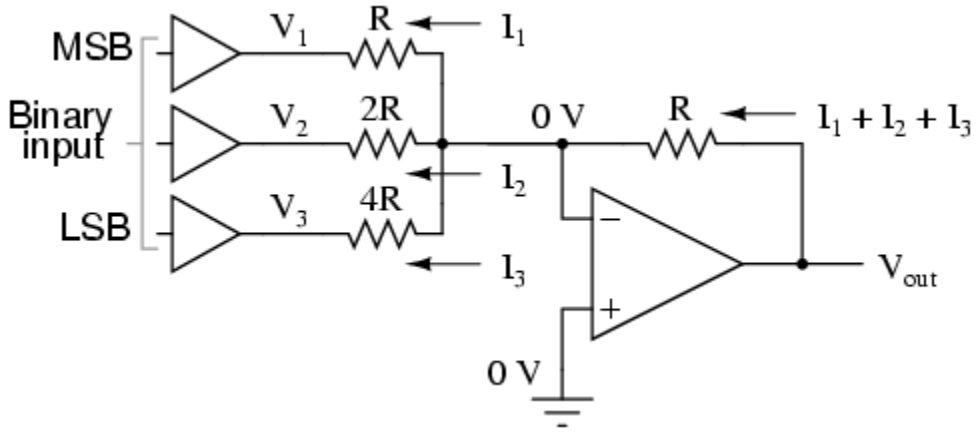


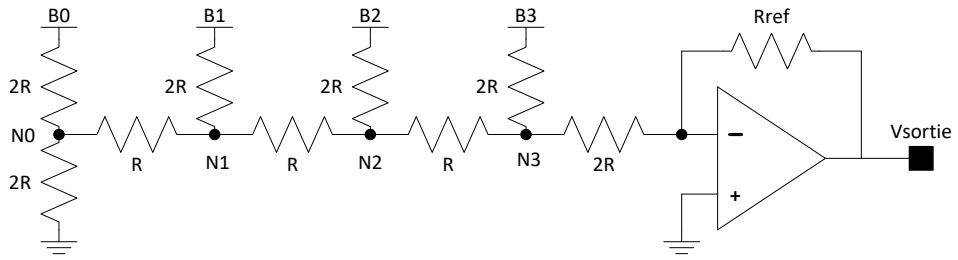
Figure 3 – DAC sommateur. Tiré de http://www.allaboutcircuits.com/vol_4/chpt_13/1.html

Dans le circuit ci-dessus, il n'y a pas de courant qui passe dans l'amplificateur opérationnel et la rétroaction fait en sorte que la sortie, Vout, s'ajuste pour que la tension sur le borne – de l'amplificateur soit à peu près égale à la tension sur la borne + (on considère que les deux valeurs sont égales et que le gain de l'amplificateur est infini dans les calculs). Le courant causé par V1 est V1/R où V1 est une valeur binaire : 3.3V ou 0V par exemple. Le courant causé par V2 est V2/2R. Le courant causé par V3 est V3/4R et le courant causé par Vn est $Vn/2^{n-1}R$. Ces courants s'additionnent à la sortie de telle sorte que $Vout = -1 * (V1/R + V2/2R + V3/4R + \ldots + Vn/2^{n-1}R)$. V1 est donc le bit valant le plus cher et Vn est celui qui contribue le moins à la sortie…

1.3.2 Le DAC R2R

Le DAC sommateur n'est pas implémenté dans la pratique, parce qu'il faut plusieurs valeurs différentes de résistances. Cela cause de l'imprécision, en plus de complexifier le montage.

La plupart des DACs qu'on retrouve dans l'industrie sont des DACs R2R implémentés comme suit :



À l'instar du DAC sommateur, le DAC R2R utilise un amplificateur opérationnel pour convertir une valeur digitale en tension analogique. Comme pour ce dernier, le voltage de sortie est proportionnel à Rref * [la somme des contributions en courants de chaque bit]. Chaque bit (B0 à B3 dans la figure) produit un courant qui circule vers les autres bits, le ground et la broche – de l'ampli. C'est la portion de courant qui se dirige vers la broche – qui détermine le poids du bit.

Une idée permet de comprendre le DAC R2R simplement : la résistance équivalente parallèle à gauche ou à droite est toujours 2R. De ce fait, le courant se divise toujours par 2 à chaque nœud et la contribution de chaque bit, en courant sur la broche -, est deux fois plus petite lorsqu'on s'éloigne d'un nœud.

> *Rappel : selon le principe de superposition des sources, si on veut regarder la contribution en courant/voltage d'une source (un bit dans le cas présent), il faut court-circuiter toutes les autres sources de tension (les autres bits) afin de faire le calcul…*

Si B3 vaut « 1 » ou 3.3V, dans la figure ci-dessus par exemple :
- Le courant I tiré de B3 sera 3.3V/3R.
- Le courant provenant de B3 sera divisé en deux au nœud N3 : la résistance équivalente à gauche est 2R et celle à droite est aussi 2R.
- La contribution en courant de B3 sur la broche – de l'ampli sera $I/2^1$.

Si B1 vaut « 1 » ou 3.3V, dans la figure ci-dessus par exemple :
- Le courant I tiré de B1 sera 3.3V/3R.
- Le courant provenant de B1 sera divisé en deux au nœud N1 : la résistance équivalente à gauche est 2R et celle à droite est aussi 2R.
- Le courant provenant de B1-N1 sera divisé en deux au nœud N2 : la résistance équivalente vers B2 est 2R et celle à droite est aussi 2R.
- Le courant provenant de B1-N1-N2 sera encore divisé en deux au nœud N3.
- Au final, la contribution en courant de B1 sur la broche – de l'ampli sera $I/2^3$.

1.4 Analog To Digital Converter (ADC)

Un ADC convertit un signal analogique en signal digital. Les deux propriétés les plus importantes des ADCs sont la précision (le nombre de bits de l'ADC) et la vitesse d'échantillonnage de l'ADC.

Habituellement, les valeurs possibles lues par un ADC sont entre 0 et un voltage de référence qui doit être stable. L'ADC lit des valeurs avec une précision maximale étant du voltage de référence divisé par 2^nombre de bits de l'ADC. Si on suppose un ADC 10bits par exemple avec une tension de référence de 2.5V, une unité de la sortie digitale de l'ADC vaudra 0.00244V (2.5V/1024).

D'un point de vue matériel, le signal à l'entrée d'un ADC est presque toujours filtré en fonction de la fréquence d'échantillonnage de l'ADC (fréquence de Nyquist, pour éviter le recouvrement spectral). Dans la plupart des cas, un offset DC est également fourni aux signaux AC pour que les signaux AC inférieurs à 0 ne soient pas perdus.

D'un point de vue logiciel, les ADCs doivent être configurés, puis on lit la valeur de sortie de l'ADC à partir d'un registre. Par ailleurs, les ADCs sont souvent multiplexés : plusieurs broches peuvent conduire un signal externe vers l'ADC. Cela permet d'utiliser un seul ADC, souvent coûteux en quantité de matériel, pour mesurer plusieurs signaux. La contrepartie est de divisé la vitesse d'échantillonnage de l'ADC parmi plusieurs canaux.

Enfin, les ADC utilisent souvent le DMA pour transférer les échantillons de l'ADC à la mémoire, de la même façon que les DACs.

1.4.1 ADC FLASH, ADC par approximation successive et ADC suiveur

Les exemples d'ADC suivant seront présentés à partir du site http://www.allaboutcircuits.com/vol_4/chpt_13/1.html (http://www.allaboutcircuits.com/textbook/digital/chpt-13/digital-analog-conversion/):

- ADC parallèle\FLASH
- ADC avec approximation successive
- ADC suiveur

À travers ces exemples, il sera montré les compromis qui doivent être fait entre la précision, la vitesse et le coût des ADCs.