



Piscine embarquée

Module01 : Timers

contact@42chips.fr

Résumé: Tic Tac, Tic Tac, Tic Tac.

Version: 1

Chapitre I

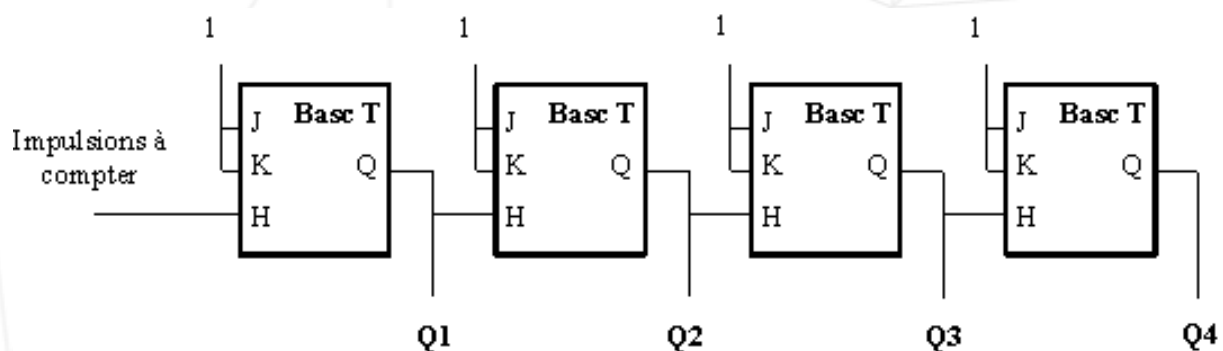
Préambule

En électronique, un compteur est un circuit intégré numérique destiné à compter le nombre d'impulsions appliquées à son entrée.

Il est composé d'un certain nombre de [bascules](#) D, T ou JK.

Le compteur le plus simple est obtenu en mettant en cascade une série de bascules T, le signal à compter étant appliqué à l'entrée de la première bascule ; la sortie de cette bascule pilote l'entrée de la deuxième bascule et ainsi de suite.

Le résultat du comptage apparaît sous forme de nombre binaire, la première bascule indiquant le chiffre le moins significatif.



La capacité du compteur, c'est le nombre maximum d'impulsions qu'il peut totaliser. Elle vaut 2^N pour un compteur binaire.

Si l'on dépasse la capacité, le compteur revient à 0 et se remet à compter.

Mais bon, trêve de plaisanteries, voyons maintenant ce qui nous intéresse le plus. Comment le faire sur [minecraft](#)

Chapitre II


Consignes générales

Sauf contradiction explicite, les consignes suivantes seront valables pour tous les exercices.

- Le langage utilisé pour ce projet est le C.
- Il n'est pas nécessaire de coder à la norme de 42.
- Les exercices sont très précisément ordonnés du plus simple au plus complexe. En aucun cas nous ne prendrons en compte ni n'évaluerons un exercice complexe si un exercice plus simple n'est pas parfaitement réussi.
- Vos exercices seront évalués par des responsables de l'association 42Chips.
- Vous ne devez laisser aucun autre fichier que ceux explicitement spécifiés par les énoncés des exercices dans votre répertoire lors de la peer-évaluation.
- Toutes les réponses à vos questions techniques se trouvent dans les **datasheets** ou sur Internet. À vous d'utiliser et d'abuser de ces sujets pour comprendre comment réaliser votre exercice.
- Vous devez utiliser la datasheet du microcontrôleur qui vous est fourni et commenter les parties importantes de votre programme en renseignant où vous avez trouvé les indices dans le document, et, si nécessaire, expliquer votre démarche. Ne faites pas des pavés non plus. Il faut que cela reste clair.
- Vous avez une question ? Demandez à votre voisin de droite ou de gauche. Vous pouvez demander sur le salon dédié dans le Discord de la piscine ou en dernier recours à un staff.

Chapitre III

Tic & tac

	Exercice : 00
Clignotant	
Dossier de rendu : <i>ex00/</i>	
Fichiers à rendre : <i>Makefile, *.c, *.h</i>	
Fonctions Autorisées : <i>avr/io.h</i>	

- Vous devez écrire un programme qui permet d'allumer et éteindre la LED D2 (PB1) à une fréquence d'environ 1 Hz.
- Vous devez écrire un code qui permet d'attendre plusieurs centaines de millisecondes et qui sera inséré dans la boucle infinie du programme (pas de TIMER hardware).
- Le changement d'état de la LED doit être fait avec une unique opération bitwise, il ne faut pas utiliser de condition (if else).
- Vous devez utiliser uniquement les registres AVR (DDRX , PORTX, PINX).




1Hz == (allumé 0.5sec et éteint 0.5sec).

L'objectif est de vous faire allumer une LED. Il n'y a pas besoin d'être super précis.




Vous devez à chaque fois expliquer la fonction et les valeurs assignées aux registres en commentaire ! L'exercice sera considéré comme invalide si votre main return.

	Exercice : 01
Timer1	
Dossier de rendu : <i>ex01/</i>	
Fichiers à rendre : <i>Makefile, *.c, *.h</i>	
Fonctions Autorisées : <i>avr/io.h</i>	


- Vous devez écrire un programme qui permet d'allumer et éteindre la LED D2 (PB1) à une fréquence de 1 Hz.
- Vous devez configurer les registres du `Timer1` pour commander la LED.
- La boucle infinie du programme doit rester vide.
- Et vous ne devez pas utiliser `PORTX`.



Vous devez à chaque fois expliquer la fonction et les valeurs assignées aux registres en commentaire !

	Exercice : 02
Rapport cyclique	
Dossier de rendu : <i>ex02/</i>	
Fichiers à rendre : <i>Makefile, *.c, *.h</i>	
Fonctions Autorisées : <i>avr/io.h</i>	

- Vous devez écrire un programme qui permet d'allumer et éteindre la LED D2 (PB1) à une fréquence de 1 Hz et avec un [rapport cyclique](#) de 10 %.
- Vous devez configurer les registres du `Timer1` pour commander la LED.
- La boucle infinie de votre programme doit rester vide.
- Et vous ne devez pas utiliser `PORTX`.

	Exercice : 03
Le cycle de la vie	
Dossier de rendu : <i>ex03/</i>	
Fichiers à rendre : <i>Makefile, *.c, *.h</i>	
Fonctions Autorisées : <i>avr/io.h, util/delay.h</i>	

- Vous devez écrire un programme qui permet d'allumer et éteindre la LED D2 (PB1) à une fréquence de 1 Hz et avec un **rapport cyclique** variable de 10 à 100 %.
- Vous devez configurer les registres du **Timer1** pour commander la LED.
- Le rapport cyclique de la LED doit être contrôlé comme suit :
 - Presser sur le bouton SW1 l'incrémente de 10 %.
 - Presser sur le bouton SW2 le décrémente de 10 %.



Le delay est seulement autorisé pour débouner les boutons
Toute utilisation autre résultera en un exercice invalide