



Piscine embarquée

Day 02 : Interruptions

[contact@42chips.fr](mailto:contact@42chips.fr)

*Résumé: OBJECTION!*

# Chapitre I

## Préambule

Une étude américaine menée sur des cadres de hauts rangs a révélé qu'en moyenne, un cadre est interrompu toutes les 8 minutes, et l'interruption dure 3 minutes.

Cela peut être dû à un visiteur inattendu, un appel téléphonique, un email ou à quelque chose qui pose problème...

En ajoutant qu'il faut quelques minutes pour retrouver sa concentration et se replonger dans son travail, on commence à percevoir la situation réelle.

Sur une période de 13 minutes, l'employé passe 5 minutes sur une activité inattendue et souvent non rentable, ce qui représente 40 % de son temps.

On comprend mieux pourquoi certaines personnes avancent si vite, pendant que d'autres subissent ces aléas.

Simplement en gérant son temps et en limitant les interruptions, on peut économiser jusqu'à 40 % de notre productivité.

Sur une base de 40 heures par semaine, cela donne 752 heures par an, ou encore 94 jours de travail !

# 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

## ISR

|   |               |
|---|---------------|
|      | Exercice : 00 |
| Interruption extérieure   |               |
| Dossier de rendu : <i>ex00/</i>   |               |
| Fichiers à rendre : <b>Makefile</b> , <b>*.c</b> , <b>*.h</b>                         |               |
| Fonctions Autorisées : <b>avr/io.h</b> , <b>util/delay.h</b> , <b>avr/interrupt.h</b> |               |

- Vous devez écrire un programme qui change l'état de la LED D1 lorsque l'on appuie sur le bouton SW1.
- Vous devez utiliser les interruptions pour lire l'état du bouton. Il est interdit de lire les registres PINx.



## Exercice : 01


## Timer0 interruption et PWM

Dossier de rendu : *ex01/*Fichiers à rendre : **Makefile**, **\*.c**, **\*.h**Fonctions Autorisées : **avr/io.h**, **util/delay.h**, **avr/interrupt.h**

- Vous devez configurer le **Timer0** pour déclencher une interruption périodique qui fait varier le rapport cyclique de la LED PB1 contrôlée par le **Timer1**.
- La fréquence du **Timer1** doit être assez élevée pour ne plus voir la LED clignoter.
- N'hésitez pas à utiliser plusieurs registres Timer pour réaliser cet exercice.
- Le rapport cyclique doit varier en boucle de 0 % à 100 % puis de 100 % à 0 % en 1 seconde.

# Chapitre IV

## Bonus : Multiplex !

|   |               |
|---|---------------|
|                        | Exercice : 02 |
| Compteur binaire  |               |
| Dossier de rendu : <i>ex02/</i>   |               |
| Fichiers à rendre : <code>Makefile</code> , <code>*.c</code> , <code>*.h</code>                         |               |
| Fonctions Autorisées : <code>avr/io.h</code> , <code>util/delay.h</code> , <code>avr/interrupt.h</code> |               |

- Vous devez écrire un programme qui :
  - chaque fois que vous pressez le bouton SW1 incrémente une valeur ;
  - chaque fois que vous pressez le bouton SW2 décrémente une valeur ;
  - et l'affiche en permanence cette valeur sur les LEDs D1 D2 D3 et D4 en binaire.
- Vous devez utiliser les interrupts et ne rien avoir dans votre boucle de main.



Si les 4 LEDs étaient sur les GPIO PB0, PB1, PB2, PB3, cet exercice serait plus simple.

Malheureusement la LED D4 est sur PB4 au lieu de PB3 car PB3 est utilisé pour autre chose.

Il va falloir manipuler des bits.