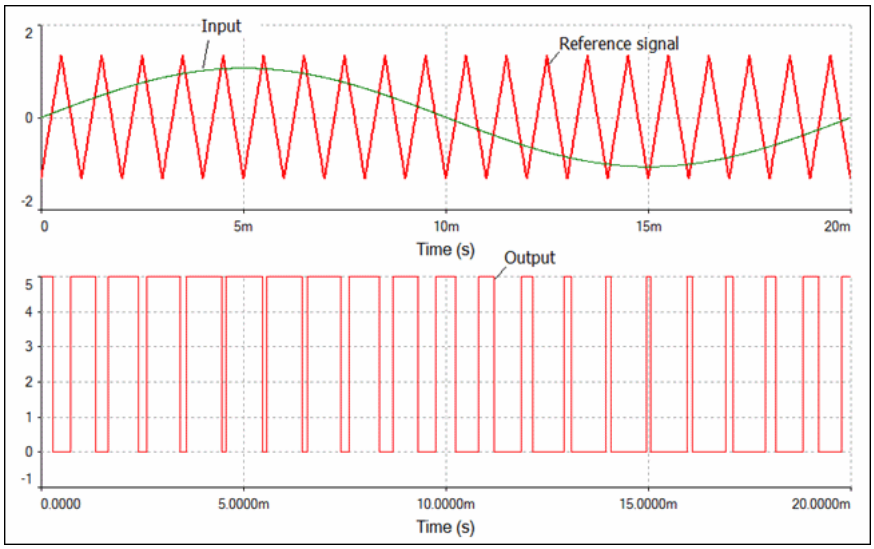
STM32 TIMER

Objectifs : Mise en œuvre des PWM (Pulse Width Modulation)

Matériel : Ce TP utilise une NUCLEO-F411RE, mais n'importe quelle autre carte NUCLEO convient.
Logiciel : MBED

Le signal PWM (MLI, Modulation de largeur d'impulsion) est un signal de fréquence constante et de rapport cyclique variable. Il est mis en œuvre dans des fonctions telles que :

La synthèse vocale ou associée à un filtre passe bas permet la synthèse de signaux audios.
La commande en vitesse d'un moteur à courant continu, ou ce dernier fait naturellement office de filtre passe bas.
La commande en position d'un servomoteur
La génération de signaux aléatoires ou périodiques, pour un onduleur par exemple.



www.multisim.com

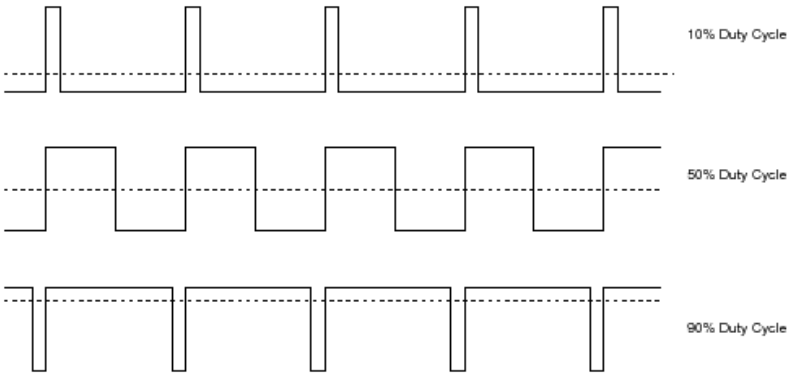
Tension sinusoïdale modulée PWM à l'aide d'un signal triangulaire et d'un comparateur.
La valeur moyenne est récupérée simplement par un filtre passe bas.

Valeur moyenne d'un signal rectangulaire :

La valeur moyenne d'un signal rectangulaire dépend du rapport cyclique (duty Cycle) ainsi que de la tension maximum (Vmax)

Le rapport cyclique η=th/T

th représente la durée de l'état haut et T la période.
Si th=0, η=0
Si th=T, η=1
donc 0≤η≤1



Valeur moyenne :

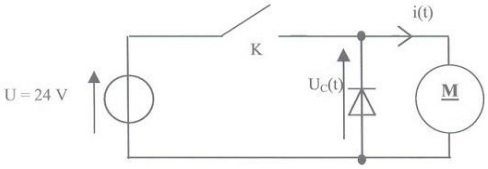
Vm=Vmax.th/T = Vmax. η

En retirant la fréquence porteuse F=1/T avec un filtre passe bas, il reste la valeur moyenne du signal.
Le filtre passe bas peut être électronique ou mécanique (cas d'un moteur et de son inertie) ou optique (cas de l'œil humain qui filtrera les fréquence au delà de 30Hz).

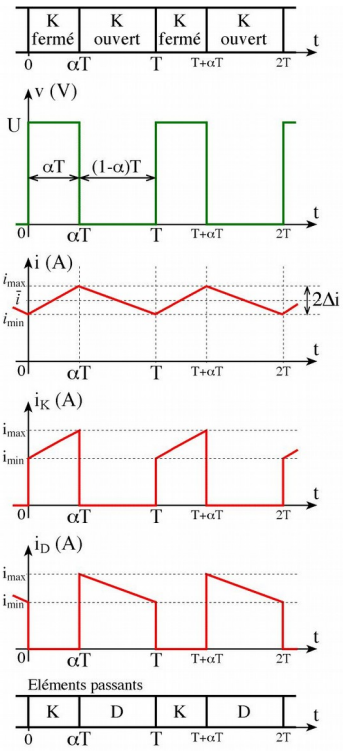
La génération d'un signal par PWM est particulièrement avantageux du point de vue de la consommation de la commande. En effet il sera produit par un transistor MOSFET qui ne consomme pratique rien en mode triode (le transistor se comporte comme une résistance de très faible valeur) et rien du tout en mode bloqué.

Exemple de structure d'un hacheur permettant la commande d'un moteur à courant continu par PWM.

K est un interrupteur électronique, généralement un MOSFET.



K fermé le courant passe de l'alimentation vers le moteur.
K ouvert l'énergie emmagasinée dans l'inductance du moteur s'évacue sous forme d'un courant traversant la diode (dite de roue libre)



U représente le signal de commande PWM.

i représente le courant dans le moteur
ik représente le courant dans le transistor
id représente le courant dans la diode

MBED propose une bibliothèque PWM très simple d'emploi, qui permet de définir la période et le rapport cyclique d'un signal PWM. *Attention tous les GPIO ne supportent pas le mode PWM.*

https://os.mbed.com/docs/v5.7/reference/pwmout.html

Pour tester le mode PWM, créer un nouveau projet avec comme modèle "Output a PWM signal"

Tester ce programme qui contrôle l'intensité lumineuse sur la LED verte

```
#include "mbed.h"
PwmOut mypwm(LED1); // Attention tous les GPIO ne supportent pas le PWM

int main() {
    mypwm.period_ms(10);
    mypwm.pulsewidth_ms(3);
    printf("pwm set to %.2f %%\n", mypwm.read() * 100);
    while(1) ;
}
```

STM32 TIMER

Exercice 1:

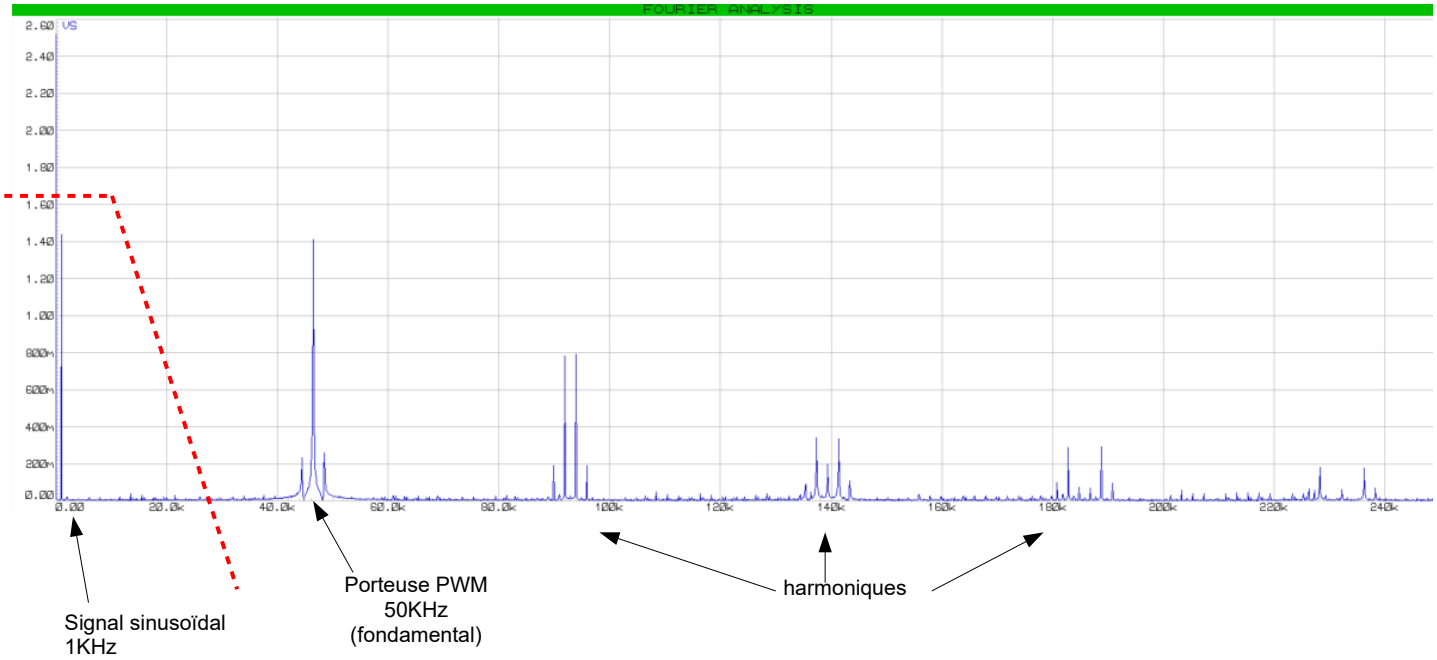
Réaliser un programme en C (C++), faisant varier l'intensité de la LED verte de +10% toutes les 100mS. La période PWM sera de 1mS.
La LED est allumée durant l'état haut de la sortie. La puissance lumineuse restituée est proportionnelle au rapport cyclique η=th/T (th= temps état haut, T est la période ici 1KHz). La persistance rétinienne fait office de filtre passe bas. (Il est admis que l'œil humain ne perçoit pas les variations de lumière inférieure à 30mS)

Exercice 2:

Un signal PWM peut permettre de faire clignoter automatiquement une LED, avec les mêmes structures que pour l'exercice, la fréquence est maintenant de 1Hz et le rapport cyclique est de ½ (th=T/2). Une fois le périphérique configuré et activé, le clignotement est automatique et n'est plus géré par le logiciel, le programme comportera une boucle infinie vide.

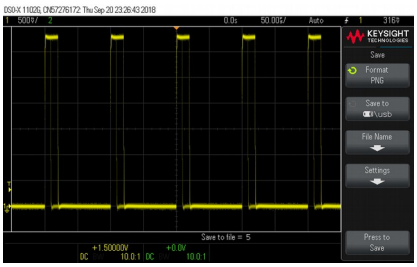
Exercice 3 :
La fonction PWM permet la synthèse de signaux, la valeur moyenne du signal PWM étant directement proportionnelle au rapport cyclique, celle-ci peut être récupérée par un simple filtre passe-bas. Ainsi il est possible de générer un signal analogique dont la fréquence est très inférieure à celle de la PWM, en filtrant la fréquence PWM on récupère les fréquences de modulation.

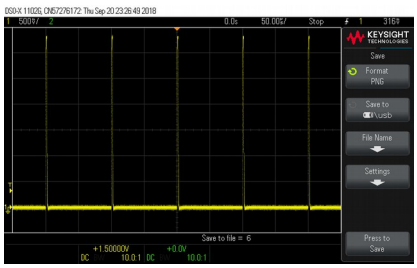
Voici un exemple de spectre d'un signal PWM de fréquence 50KHz modulé par une sinusoïde de fréquence 1KHz

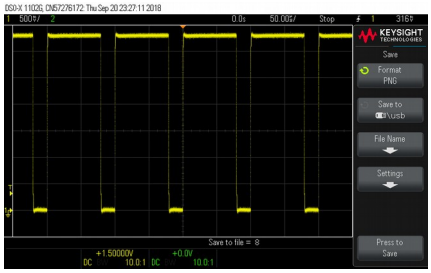


En plaçant sur ce signal un filtre passe bas (en rouge) éliminant la porteuse PWM (et donc toutes les harmoniques) , on récupère le signal sinusoïdal modulant.

Exemples de signaux PWM

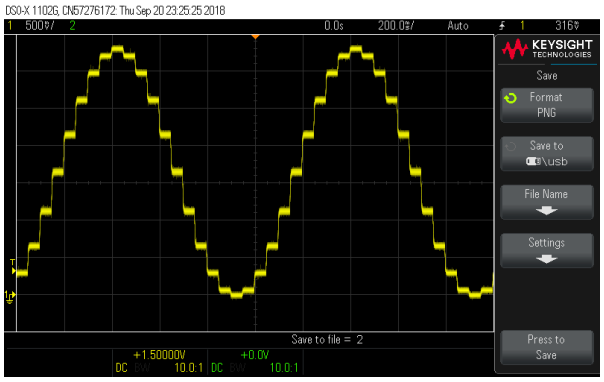




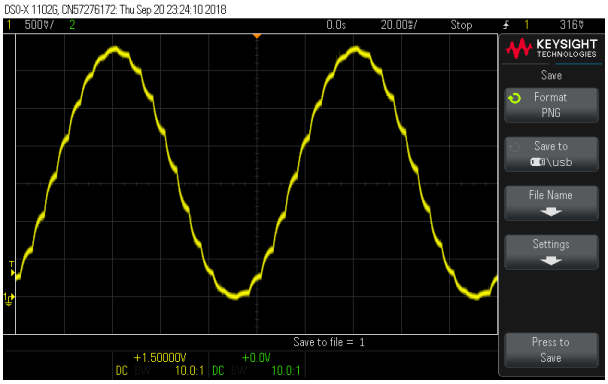


Exemple de signaux après filtrage passe bas

fc =40Khz



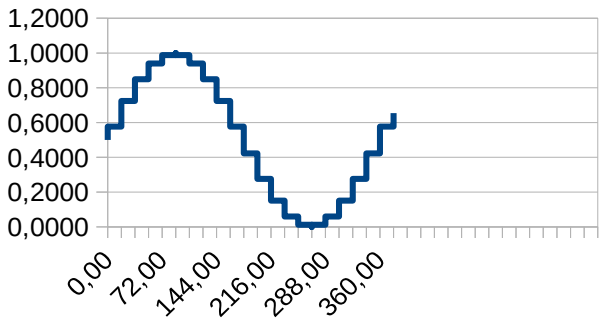
fc =10Khz



La sortie du signal PWM s'effectuera sur un connecteur Arduino de la carte Nucleo et sera reliée à un filtre RC passe bas (prendre C=100nF) du premier ordre et de fréquence de coupure 100Hz . (On rappelle que ωc=1/RC),

La fréquence du signal PWM sera de 1KHz.

A l'aide d'un tableur on génère 20 valeurs comprises entre 0uS et 1000uS suivant une fonction sinusoïdale.
Dans le programme, ces valeurs entières seront introduites dans un tableau qui servira à produire les PWM successives.



Le programme utilisera le tableau de vingt valeurs et mettra à jour la PWM (méthode pulsewidth_us) périodiquement de manière à produire une onde sinusoïdale de fréquence 10Hz.
La temporisation entre deux valeurs PWM sera réalisée avec une fonction wait dans un premier temps.

Vérifier le résultat à l'aide d'un oscilloscope.

Interpréter le résultat, indiquer comment l'améliorer.

Améliorer le programme en remplaçant la méthode wait par un objet Ticker. En quoi cette seconde solution est elle préférable ?

On remarque dans le tableau ci-après une répétition des valeurs de th. Comment améliorer le programme en réduisant la taille du tableau à 10 valeurs tout en conservant la même résolution ?

angle en radians	sinus	Th PWM (uS)
0,0000	0,0000	500
0,3142	0,3090	655
0,6283	0,5878	794
0,9425	0,8090	905
1,2566	0,9511	976
1,5708	1,0000	1000
1,8850	0,9511	976
2,1991	0,8090	905
2,5133	0,5878	794
2,8274	0,3090	655
3,1416	0,0000	500
3,4558	-0,3090	345
3,7699	-0,5878	206
4,0841	-0,8090	95
4,3982	-0,9511	24
4,7124	-1,0000	0
5,0265	-0,9511	24
5,3407	-0,8090	95
5,6549	-0,5878	206
5,9690	-0,3090	345
6,2832	0,0000	500