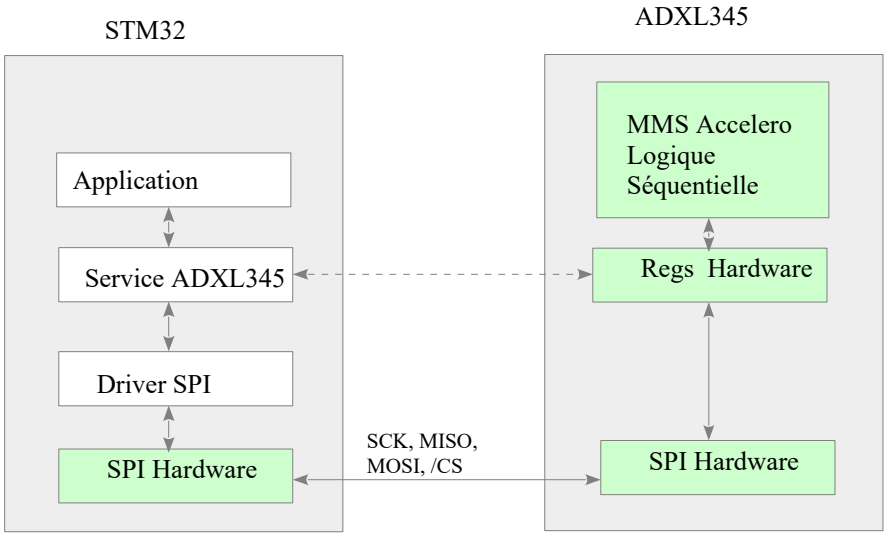
Quelques informations sur l'accéléromètre ADXL345...

Le composant permet d'obtenir l'accélération à laquelle est soumise la masselotte mobile embarquée dans le composant.
Il peut être vu comme un « périphérique externe » au STM32. Il est donc doté de registres de ***configuration*** et de registres d'***utilisation*** (*Regs Hardware* sur la figure ci-dessous). L'accès à ces registres se fait par l'intermédiaire d'un bus SPI (cf cours dédié UART/SPI). On peut représenter la communication par le schéma suivant :



Le principe d'accès aux registres (géré par le SPI)

Gérer l'accéléromètre c'est donc accéder aux registres (cf documentation) de l'ADXL345 en écriture ou en lecture. Les registres sont organisés à la manière d'une mémoire : adresse / donnée. Le principe de lecture / écriture se fait en deux temps :

- le STM32 envoie l'adresse sur 6 bits, plus 2 bits dont l'un explicite si le prochain octet est à lire ou à écrire (R/W). Le format 8 bits est le suivant : | R/W MB A5 A4 A3 A2 A1 A0| . Le bit M permet de faire une lecture ou écriture en multi bytes (voir datasheet ADXL345),

- le STM32 envoie ou lit la (les) donnée(s) , selon les deux bits particuliers R/W et MB du précédent octet d'adressage.

La gestion de la ligne /CS se fait « à la main ». En effet, la lib SPI configure la pin /CS en sortie pushpull, donc contrôlée par ODR. C'est donc au niveau service qu'il faut baisser la ligne lors de tout transfert (mise à 0 au début d'un accès simple ou multiple et mise à 1 à la fin), cf datasheet.

La couche Service ADXL345

A ce niveau on gère la configuration du composant et son utilisation (ici essentiellement, la lecture des 3 axes).

La configuration :
Comme souvent, le composant permet beaucoup de cas d'utilisations, et donc pas mal de registres à gérer.

Afin de faciliter la tâche, voici une présélection des registres importants à configurer :

- ***POWER_CTL*** (@ 0x2D)
  Permet de gérer les mode *low power* typiquement. Dans l'application, on peut ignorer cet aspect d'économie d'énergie (dans une première étape) et placer uniquement le bit *Measure* à '1' pour lancer les mesures régulièrement,

- ***BW_RATE*** (@ 0x2C)
  Permet de fixer le nombre de mesures d'accélération faites par seconde sur les 3axes. Ce n'est pas critique dans l'application voilier. Un maximum de 100Hz semble raisonnable,

- ***DATA_FORMAT*** (@ 0x31)
  Ce registre permet de régler la résolution de mesure. Même si dans notre application il est inutile d'avoir une grande précision, on peut partir sur une « Full résolution », en +/-16g, mais tout autre réglage est possible. Le bit *Justify*, permet d'aligner les 16 bits à droite ou à gauche. Un alignement à droite, classique, est conseillé.

L'utilisation :
Il s'agira ici de lire les 3 axes (ou seulement 2). Les registres utiles sont DATAX0, DATAX1,…. DATAZ1. Chaque axe est donc géré par 2 octets.
On utilisera ici à profit une lecture multi bytes.