Kirthika MURUGENDRAN - Dignon TRAN

TP5 - Liaison et bus, consommation électrique

Le jour où le TP a été réalisé, les connexions WiFi de l'ENSEA ne fonctionnaient pas (wifi-ensea et eduroam). Pour certaines recherches, notamment la documentation sur quelques commandes, cela a pris un peu de temps.

9.1. Mesure de la consommation électrique.

9.1.1. Préparation

La mesure du courant se fait à l'aide d'une **résistance de shunt** entre load et supply. C'est l'une des méthodes que l'on peut utiliser pour mesurer un courant. La résistance de shunt est branchée en parallèle avec l'élément à mesurer. Elle a une valeur assez faible, ce qui permet de mesurer la "vraie" valeur du courant (qui peut être importante).
L'inconvénient peut être la perte de puissance par dissipation de chaleur (effet Joule) à cause de la résistance de shunt, qui est assez faible, devant un courant qui peut être important.
On peut également mesurer le courant en utilisant un ampèremètre branché en série avec le circuit électrique

On mesure les éléments passifs connectés à Load et Supply, qui les alimentent, et qui passent par la résistance de shunt, puisque l'on utilise la méthode avec la résistance de shunt pour mesurer le courant.

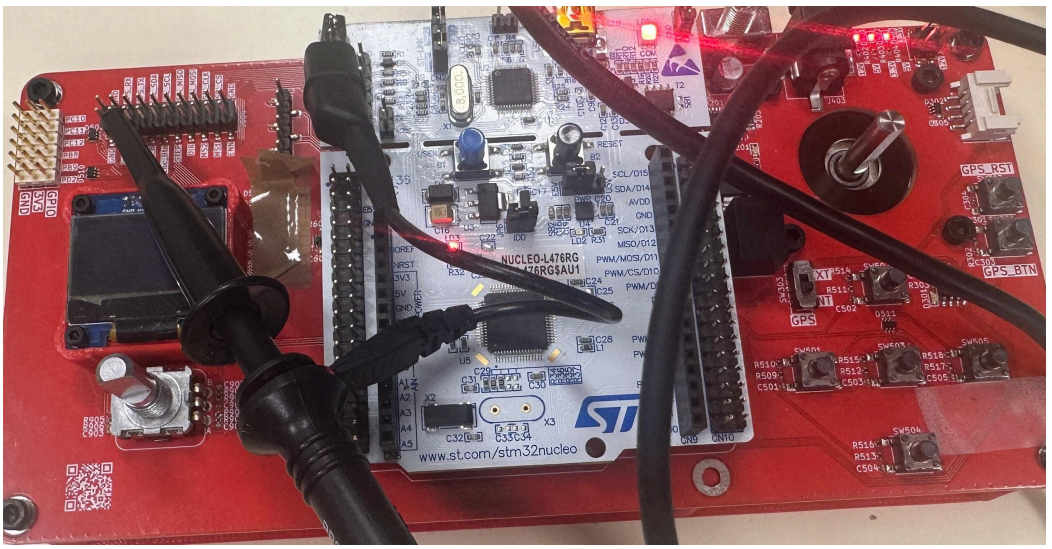
On attend un courant I qui vaut quelques mA. La chute de tension aux bornes de la résistance de shunt vaut $V_{shunt} = V_{load} - V_{supply} = R_{shunt} I_{load}$.
Pour la tension de sortie, on a la formule suivante dans la figure 9.1 : $V_{out} = (I_{load} * R_{shunt}) * Gain + V_{ref}$. On a un gain de 200 et une tension de référence $V_{ref} = 0V$ car reliée à la masse GND (on remarque de plus que $I_{load} * R_{shunt} = V_{load} - V_{supply}$).
En prenant $I_{load} = 100mA$ et $R_{shunt} = 10m\Omega$ par exemple (probablement des valeurs proches de ce que l'on pourrait mesurer car suffisamment basses), on a $V_{shunt} = 0.2V$, en sorte de l'amplificateur différentiel.

La grandeur I_{mes} est une grandeur analogique, et doit être convertie en signal numérique pour pouvoir être traitée par la carte Nucléo. On recourt à la CAN (conversion analogique numérique). On code la valeur de I_{mes} sur 12 bits. On a donc $2^{12} = 4096$ niveaux possibles pour coder le signal analogique en signal numérique (donc des valeurs comprises entre 0 et 4095). On a une résolution de $Q = \frac{3.3V}{4096} = 0.805mV$. On devrait lire une valeur de tension "numérisée", représentative de I_{mes}, autour de $\frac{0.2V}{0.805mV} = 248$. L'ordre de grandeur à lire sur le signal numérisé est de 248.

9.1.2. Manipulation de base

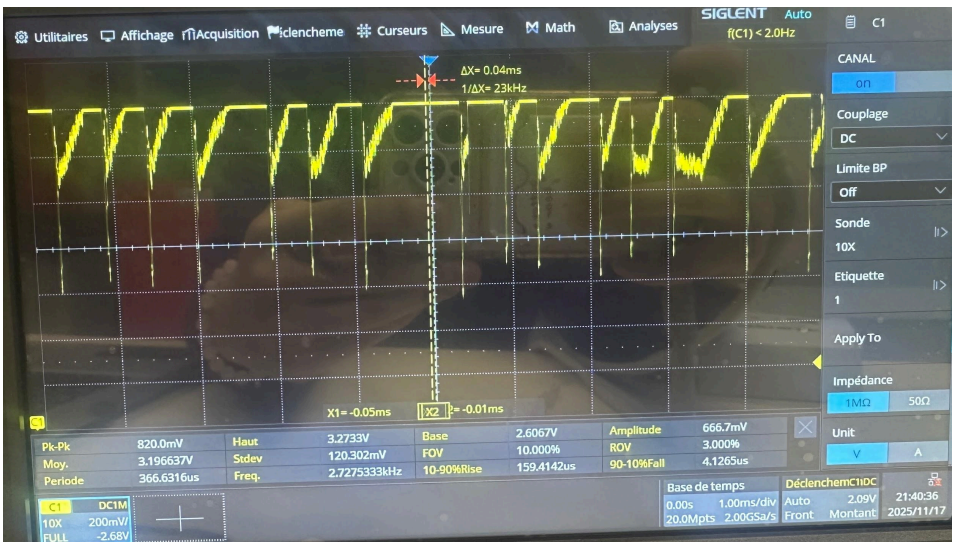
La carte doit être éteinte au moment des branchements pour éviter un court-circuit. On l'allume une fois les branchements réalisés. (Il faut à chaque fois éteindre la carte avant de manipuler les connexions.)

Kirthika MURUGENDRAN - Dignon TRAN

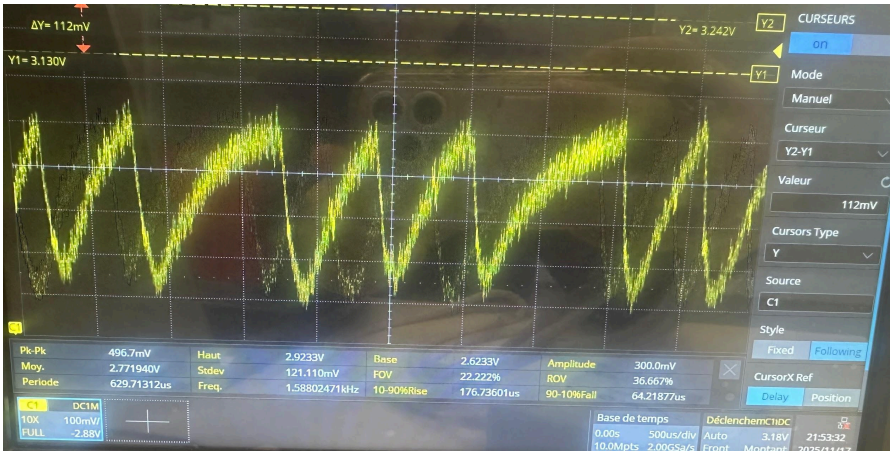


On effectue le branchement demandé et on observe le signal sur l'oscilloscope.

On observe ce signal lorsque les leds sont éteintes, ou alors on observe le signal ci-dessous statique.



On observe ce signal lorsque les leds sont allumés qui varie constamment, alors qu'on est censé observer un signal carré. Ce signal ne convient pas. Malgré le trigger en mode "normal" et le réglage manuel du trigger, le signal ne se stabilise pas. C'est probablement à cause des différents bruits autour du branchement, à cause de la carte ou de la sonde qui peut être défectueuse. Autre hypothèse possible, le code exécuté est erroné. Étant donné que la valeur de la tension mesurée varie autour de 0.4 V, le signal est assez faible et est sujet aux bruits. Au moment où on allume les LEDs, le signal est décalé vers le haut. On suppose qu'il y a un offset qui est appliqué au moment où les LEDs sont allumées.



C'est peut-être aussi à cause du code qu'on observe ce signal. On a commenté le *HAL_PWR_EnterSLEEPMode()* de la fonction *loop()* et on observe le signal ci-dessus et quand la LED est éteinte, on ne voit plus de signal.

On a passé plus d'une heure sur cette partie à cause du signal non satisfaisant.

9.1.3. Mode power down

Nous avons utilisé le site https://controllerstech.com/low-power-modes-in-stm32/ qui explique et compare les différents modes.

Mode	HAL_PWR_EnterSLEEPMode ()	HAL_PWR_EnterSTOP Mode()	HAL_PWR_EnterSTANDBY Mode()
Horloges	Actives	Eteintes pour la plupart	Toutes éteintes
RAM	Conservée	Conservée	Perdue
Consommation	faible	très faible	minimale
Précisions	Les périphériques (timers, USART, etc.) restent actifs. Utile pour redémarrer rapidement une opération	Utile pour les redémarrages périodiques des opérations	Utile si l'appareil doit rester éteint pendant une longue période. On redémarre à partir d'un *reset*

```
File download complete
Time elapsed during download operation: 00:00:01.226


Verifying...



Time elapsed during verifying operation: 00:00:00.231

File download complete
Time elapsed during download operation: 00:00:01.227


Verifying...



Time elapsed during verifying operation: 00:00:00.228
```

On observe qu'avec *HAL_PWR_EnterSLEEPMode(),* la console prend *231 ms* pour répondre.

Avec *HAL_PWR_EnterSTOPMode()*, on obtient un temps de réponse plus court : *228 ms*.

```
File download complete
Time elapsed during download operation: 00:00:01.246


Verifying...



Time elapsed during verifying operation: 00:00:00.227
```

Et avec *HAL_PWR_EnterSTANDBYMode()*, le temps de réponse est encore plus court (*227 ms*), ce qui est cohérent avec la documentation selon laquelle **le temps de réponse décroît selon le mode choisi**.

On choisit le 2e mode qui semble être le meilleur choix parmi les 3 qui est un compromis entre le temps de réponse, la précision et la consommation.

9.2. Liaison et bus

9.2.1. Le composant BMM150

On a exécuté le programme après avoir complété *readOneRegister()*, mais on n'observait rien. On a dû mettre la ligne *HAL_PWR_EnterSLEEPMode()* de la *loop()* en commentaire car c'est à cause de cela que la console renvoyait "Target is not responding".
On a finalement compris l'origine du problème : on a oublié d'ajouter le programme qui activait le printf…
On observe les mesures suivantes :

```
Mesure courant : 3045    2.453846        30.673077
Mesure courant : 3220    2.594872        32.435898
Mesure courant : 2949    2.376483        29.706043
Mesure courant : 3211    2.587619        32.345238
Mesure courant : 3350    2.699634        33.745419
Mesure courant : 3380    2.723809        34.047619
Mesure courant : 3279    2.642418        33.030220
Mesure courant : 3037    2.447399        30.592493
Mesure courant : 2998    2.415971        30.199635
Mesure courant : 3321    2.676264        33.453300
Mesure courant : 3301    2.660146        33.251831
```

Quand les LEDs étaient allumées, on mesure 40 mA (on suppose que les valeurs sont mesurées en mA). Et quand on appuyait sur le codeur rotatif pour éteindre les LEDs, les valeurs de courant tournaient autour de 30 mA.

```
Mesure courant : ****  TP5 : liaisons et bus, consommation electrique****
BMM150 connection failedBTN pressed
BTN pressed
BTN pressed
BTN pressed
BTN pressed
BTN pressed
```

On constate que ça fonctionne bien. Le prof a dit que ce n'était pas normal d'observer "BTN pressed" après que la connexion a été interrompue, mais nous a laissées avancer dans le TP (on devrait voir "BMM150 connected" dès qu'on appuie à nouveau sur le codeur pour rétablir la connexion).

9.2.2. Détection de l'angle par rapport au Nord magnétique

Cette partie a été entamée mais n'a pas pu être terminée pendant le TP. Voici quelques observations pendant l'écriture du code pour l'affichage.
On note plusieurs erreurs au niveau du programme au moment où on essaye de *printf()* l'angle sur l'écran OLED. L'erreur nous indique qu'il manque un point-virgule avant un crochet, après la déclaration d'une fonction, ce qui est bizarre puisqu'après le nom de la fonction, on ajoute des parenthèses et enfin le crochet…

9.3. Le projet complet (optionnel)

On n'a pas eu le temps de traiter cette partie.

Conclusion : Ce TP nous a permis d'apprendre les différentes manières d'utiliser une carte de manière économique en termes d'électricité avec les différents modes qui existent. Chaque mode a ses propres avantages et inconvénients : il faut donc faire un compromis pour pouvoir avoir la meilleure utilisation possible de la carte en fonction de ce que l'on souhaite faire. On a également pu revoir comment faire des mesures sur l'oscilloscope. Malgré les signaux insatisfaisants qu'on a pu observer, cela ne nous a pas freinées dans la compréhension du sujet. Cependant, cela nous a fait perdre du temps, et nous n'avons pas pu avancer rapidement dans le TP. Il y avait également plusieurs erreurs d'inattention qui auraient pu être évitées (notamment le code activant le *printf()*…), et nous faire gagner du temps.
Mais on garde une note positive sur ce dernier TP de microcontrôleur.