TP2

Le but de ce TP va être pour nous d'effectuer une liaison entre une carte Launchpad équipée d'un microcontrôleur MSP430G2553 et une carte Launchpad équipée d'un microcontrôleur MSP430G2231, via une communication SPI

Pour se faire nous commençons donc par étudier les documentations techniques de chacune des deux launchpad à notre disposition :

1.1 Partie microcontrôleur MSP430G2553

Concernant la recherche des broches disponibles pour réaliser une communication SPI via la launchpad équipée d'un microcontrôleur MSP430G2553, nous constatons les éléments suivants :

Les broches suivantes permettent de paramétrer le microcontrôleur en « master » pour un échange avec un système « slave » :

- 1.1
- 1.2
- 1.6
- 1.7

1.2 Partie microcontrôleur MSP430G2231

Nous avons ensuite recherché les broches disponibles pour réaliser une communication SPI avec le MSP430G2231. Pour se faire nous parcourons alors la datasheet à la recherche de cette information.

Nous constatons alors que les broches suivantes nous permettent de paramétrer le microcontrôleur en « slave » pour un échange avec un système « master » :

- 1.5
- 1.6
- 1.7

1.3 Échange de données entre les deux cartes

Afin d'échanger des données entre les launchpad nous nous sommes basés sur le code que nous avions réalisé au cours de notre premier TP et nous nous somme aidés des exemples mis à notre disposition afin de pouvoir reprendre notre code et d'y réaliser les modifications demandées.

Durant cette partie nous avons rencontrés quelques difficultés dû au fait que nous ne disposions que d'un jeu de launchpad fonctionnel. Vous pourrez trouver en pièce jointe à ce compte rendu le code que nous avons pu réaliser afin de faire communique les MSP entre eux.

Concernant notre code nous avons commencé par nous questionner sur la configuration de la communication SPI coté MSP430G2553. Nous avons alors défini les points suivants afin de paramétrer la communication :

- Le signal d'horloge sera transmis par la broche 1.5.
- Le MISO sera transmis par la broche 1.6.
- Le MOSI sera transmis par la broche 1.7.

Nous avons ensuite réfléchi aux actions à effectuer pour en arriver aux choix suivants :

- Le caractère '1' allume la led 1.0 du MSP430G2231.
- Le caractère '2' allume la led 1.6 duMSP430 G2231.
- Le caractère '0' éteint les deux leds.
- Afficher un message dans la fenêtre putty de l'opération effectuée afin d'informer l'utilisateur.

Du coté MSPG2231 nous avons choisi de définir les points suivants concernant la communication

- Le signal d'horloge sera transmis par la broche 1.5.
- Le MOSI sera transmis par la broche 1.6.
- Le MISO sera transmis par la broche 1.7.

Nous avons ensuite défini les différents cas auxquels nous avions réfléchis précédemment dans le code coté MSP430G2231 :

- Le caractère '1' allume la led 1.0.
- Le caractère '2' allume la led 1.6.
- Le caractère '0' éteint les deux leds.

1.4 Application

Dans cette partie, nous avons réalisés un diagramme fonctionnel en vue de la réalisation futur d'un PCB destiné à la mise en œuvre du MSP430G2231. Le diagramme que nous avons réalisé est visible sur la figure ci-dessous :

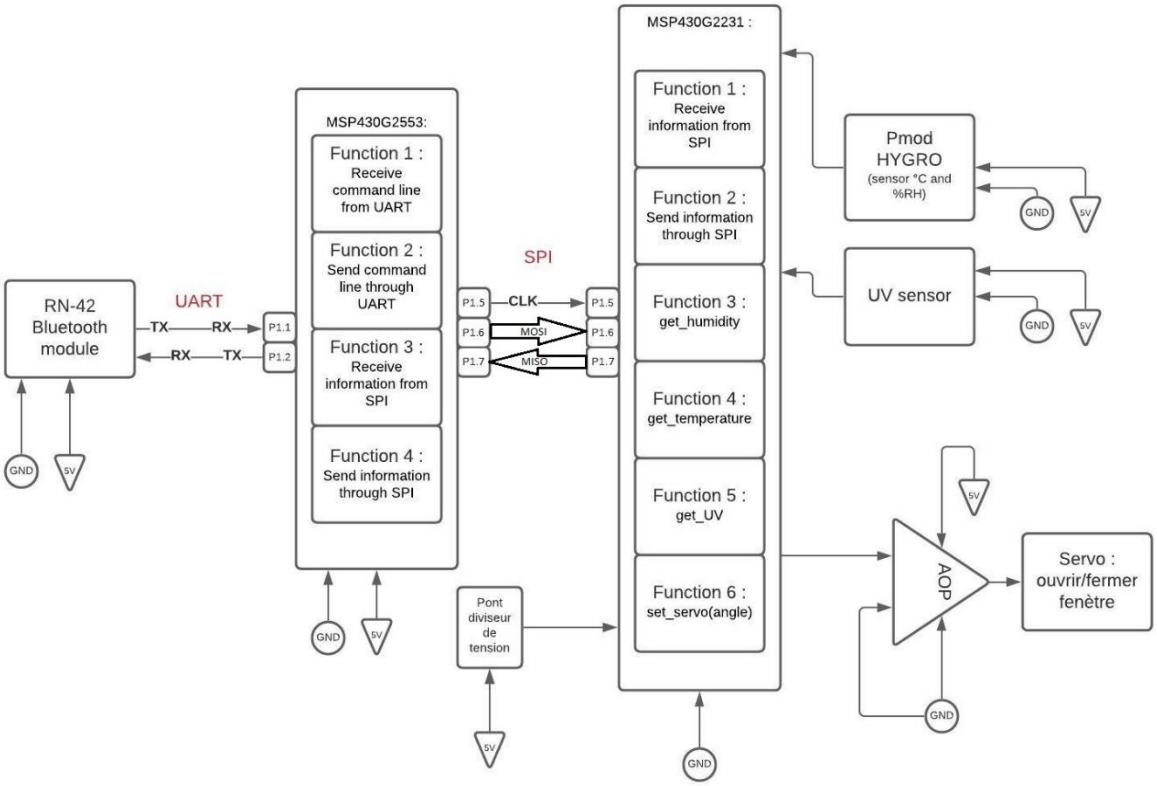

Diagramme fonctionnel

L'étude des différentes documentations techniques ainsi que la réalisation du diagramme fonctionnel du PCB nous a amener à divers problèmes. Cela impliquant donc que nous nous avons été amenés à réfléchir sur ces derniers. Nous avons ainsi pu constater que les pins présents permettant de communiquer en SPI sont les même que ceux permettant de communiquer en I2C. De ce fait, si nous utilisons une connexion par BUS SPI nous ne disposerions plus de suffisamment de pins pour connecter notre capteur. Nous pensons donc passer la connexion entre les MSP en I2C afin de libérer un pin supplémentaire. Cela nous permettrait de connecter le capteur HYGRO sur la launchpad équipée du microcontrôleur MSP430G223 par l'intermédiaire d'un bus I2C.

Ainsi le MSP430G2553 le master de deux slaves, cependant cela modifierait nôtre diagramme fonctionnel de la manière suivante : le MSP430G2553 sera master du MSP430G2231 et master de HYGRO.

Nous pensons qu'un fonctionnement comme celui-ci permettrais d'éviter d'avoir à utiliser simultanément du SPI de l'I2C dans le même projet, ce qui aurait posé problèmes si nous voulions les utiliser en même temps

Cependant cela a l'inconvénient de ne pas avoir tous les capteurs/actionneurs déportés sur la même carte (en projet réel, cela aurait été plus pratique).