Projet M2 LSE 2016

Pilotage automatique d'un voilier via un bus CAN Arduino

Auteur:
Romain Le Forestier
M2LSE

Encadrant:
Goulven Guillou

Sommaire

1 Présentation du projet

1.1 Sujet

Aujourd'hui nous disposons d'un ensemble de noeuds Arduino associés en un bus CAN permettant de récupérer un ensemble de données issues de capteurs positionnés sur un voilier et en particulier de décoder des données au format propriétaire SeaTalk. Le projet a pour objectif d'utiliser ces données pour commander via le bus SeaTalk (en se faisant passer pour une télécommande) un vérin de pilote automatique. Le projet comporte le développement d'une interface simple pour PC (PC qui sera raccordé à un noeud) pour d'une part visualiser certaines données (type tableau de bord) et pour d'autre part contrôler le pilote via un algorithme de contrôle/commande (qui pourra être un simple PID dans un premier temps) le noeud ne servant que d'interface entre le PC et le bus SeaTalk. Ce même noeud devra pouvoir également se substituer au PC, c'est-à-dire faire tourner un algorithme de pilotage de manière autonome.

1.2 Déroulement du projet

Le projet devait se dérouler de la manière suivante, premièrement réaliser l'interfaçage entre les cartes Arduino et le matériel Raymarine du pilote automatique existant sur le voilier. Les cartes Arduino proviennent d'un projet précédent et devront être réutilisées.

Ensuite, implémenter une interface pour permettre la communication entre un PC et une carte Arduino qui servira d'interface entre le réseau CAN et le pc. L'interface devra permettre une communication bidirectionnelle afin d'implanter un pilote sur l'ordinateur basé sur les données disponibles sur le réseau de carte Arduino et Raymarine et ensuite envoyer des données au pilote du bateau basé sur ces données.

1.3 Projet similaire

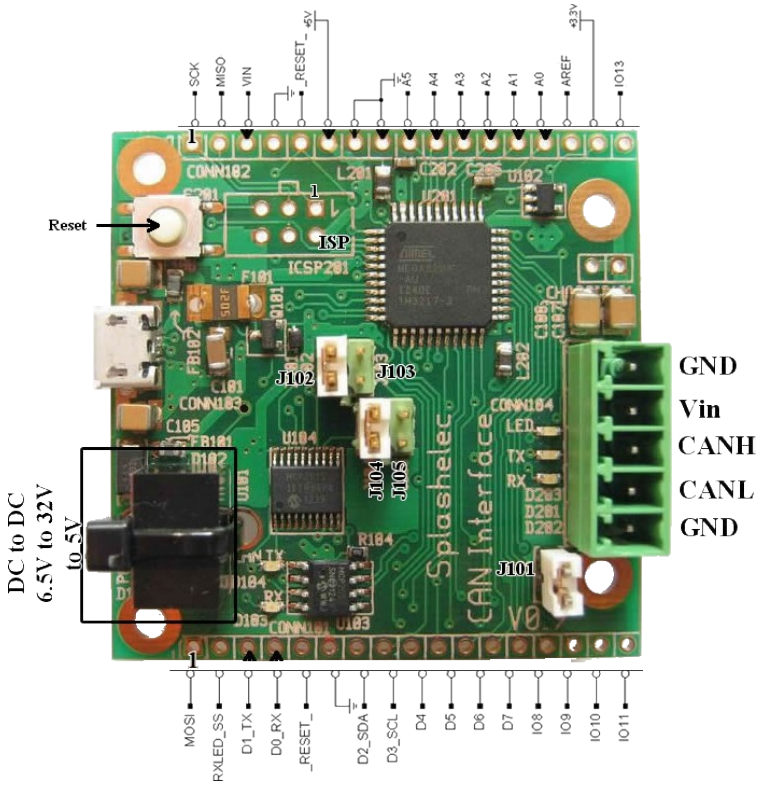
Ce projet c'est inspiré d'autre réalisation de pilote déja existante, le projet [Freeboard] apportait une approche intéressante car il impliquait la réalisation d'une interface entre une carte Arduino et du matériel Raymarine avec un ordinateur pour controller le pilote automatique.

2 Présentation du matériel utilisé

2.1 Carte Arduino

2.1.1 Présentation de la carte

Les cartes Arduino utilisées pour ce projet sont celle utilisée lors du projet de TER effectuer précédemment, le projet de TER consistait a réaliser un réseau CAN entre les cartes Arduino le projet et les sources du TER sont disponible sur un [dépot Github TER].



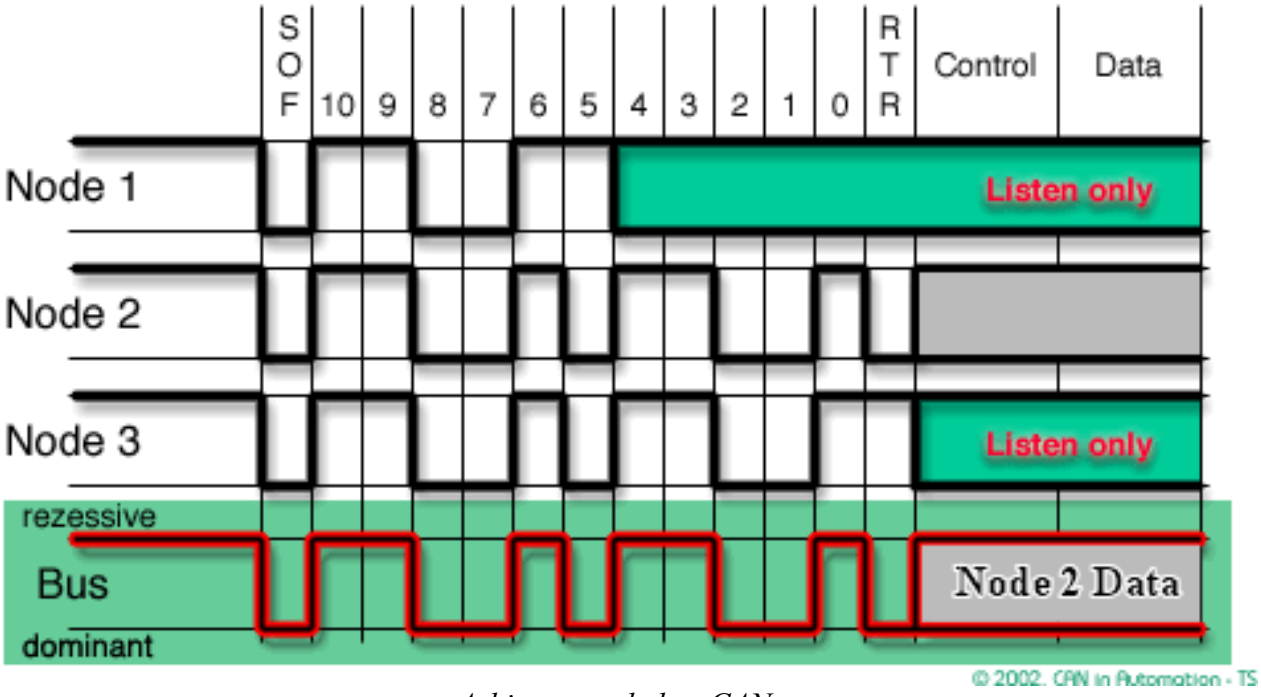
Carte Arduino utilisé pour le projet

Les cartes Arduino sont basées sur un microcontrôleur Atmel mega32u4, correspondant au processeur utilisé sur les cartes Arduino Leonardo. Un contrôleur de bus CAN, le MCP2515, est installé sur la carte et permet la réalisation du réseau de carte Arduino via le bus CAN. Pour la réalisation du réseau ont utilisera la librairie CAN utilisé lors du projet de TER.

2.1.2 Présentation du fonctionnement du protocole CAN

Le protocole CAN est un bus de terrain développer pour le secteur automobile par Bosch. Le bus utilise 2 fils pour communiqué, c'est un bus bidirectionnel sans maitre avec une vitesse

maximale de 1 Mb/s . Un seul noeud du réseau peut émettre à la fois ce qui correspond à un principe de broadcast, un noeud qui parle les autres écoutes et récupère les données s'ils en ont besoin. Les messages sur le bus sont différentiés par un identifiant de tram dans le champ d'arbitrage, car le noeud qui émet avec la plus petite valeur d'identifiant sera prioritaire, chaque noeud écoutant ce qui est émis quand il écrit, les noeuds avec une valeur d'identifiant supérieur à celui qui émet passeront en mode "écoute" et attendront que le bus soit de nouveau libre pour émettre.



Arbitage sur le bus CAN

Le 0 sur le bus CAN est dominant ce qui permet de différentier les identifiants des messages lors d'une phase d'arbitrage. Sur cette image représentant une phase d'arbitrage entre 3 noeuds sur un bus CAN, on peut remarquer que le noeud 2 remporte l'arbitrage, car il a la valeur d'identifiant la plus faible. On peut également constater que la valeur des autres noeuds est écrasée par la valeur prioritaire et que chaque noeud passe en mode écoute dès qu'il détecte qu'il a perdu la phase d'arbitrage sur le bus.

Pour mettre en oeuvre une communication sur le bus on constate donc que chaque message doit avoir un identifiant unique afin de différentier chaque noeud et grandir que 2 noeuds n'utilise pas le même identifiant en même temps se qui invaliderait la phase d'arbitrage. Pour ce faire lors du projet de TER nous avions réaliser un fichier header pour la communication CAN qui était commun a tous les noeuds ce qui garantissait que 2 noeuds n'utilisent pas le même identifiant et permettait également de garantir que chaque noeud puisse récupère la bonne donné en lecture. Pour le projet nous allons donc ajouter de nouvel identifiant. On utilisait le mode standard de CAN ce qui permettait de codé jusqu'a 2^{11} identifiant de message différent, la librairie utilisait permettait également d'utiliser le mode étendu, qui permettait de codé 2^{29} identifiant, mais cette quantité n'est pour le moment pas nécessaire pour le réseau que l'on souhaite réaliser car il ne comporte que 5 noeux.

2.2 Carte d'interface SeaTalk

Pour rendre notre réseaux de carte Arduino compatible avec le materiel Raymarine embarqué sur un voilier, il faut réaliser une carte d'interface quiu convertie la sortie série de la carte Arduino en un signal compatible avec le language des réseaux Raymarine le SeaTalk.

2.2.1 Le protocole SeaTalk

Le protocole SeaTalk est un language de communication série inventé par Raymarine vers 1989 pour réaliser une communication entre un pilote de voilier et les instruments de bords. C'est un protocole série propriétaire qui a été en partie documenter par [Thomas Knauf] sur son site. Cette documentation nous a servis de référence pour interpréter les tram émise par le pilote

3 Annexe

3.1 Référence

[Freeboard] http://www.42.co.nz/freeboard/

[dépot Github TER] https://github.com/surpriserom/Protocole_SimpleCan_Arduino

[Thomas Knauf] http://www.thomasknauf.de/seatalk.htm