ELE542 – Systèmes ordinés en temps réel

Projet de session

Robot mobile

1.0 - Objectifs

- Utiliser et programmer un microcontrôleur dans un système embarqué en temps réel.
- Se familiariser avec l'architecture de base d'un programme qui traite des évènements synchrones et asynchrones en temps réel.
- Se familiariser avec un système d'exploitation multitâche de systèmes embraqués en temps réel (RTOS).
- Se familiariser avec le fonctionnement d'un microcontrôleur et de ses périphériques internes.
- Se familiariser avec les entrées/sorties et protocoles de communication habituellement utilisés dans les systèmes embarqués.

2.0 – Description

Le but du projet est d'utiliser le microcontrôleur ATMega32 pour faire fonctionner un petit robot mobile en mode « téléguidage » et avec détection d'obstacle. Les déplacements du robot sont décidés par des commandes de téléguidage tandis que le robot détecte et évite les collisions avec les obstacles de façon autonome.

2.1 – Description du système

Le robot est composé d'une plateforme à quatre roues motrices, de deux sonars piézoélectrique et d'un module de communication RS232-Bluetooth. Le module de communication permet au robot de recevoir des commandes de téléguidage en provenance d'un ordinateur. La figure 1 ci-dessous donne une représentation schématique du circuit d'interconnections entre les composantes du robot et le microcontrôleur ATMega32.

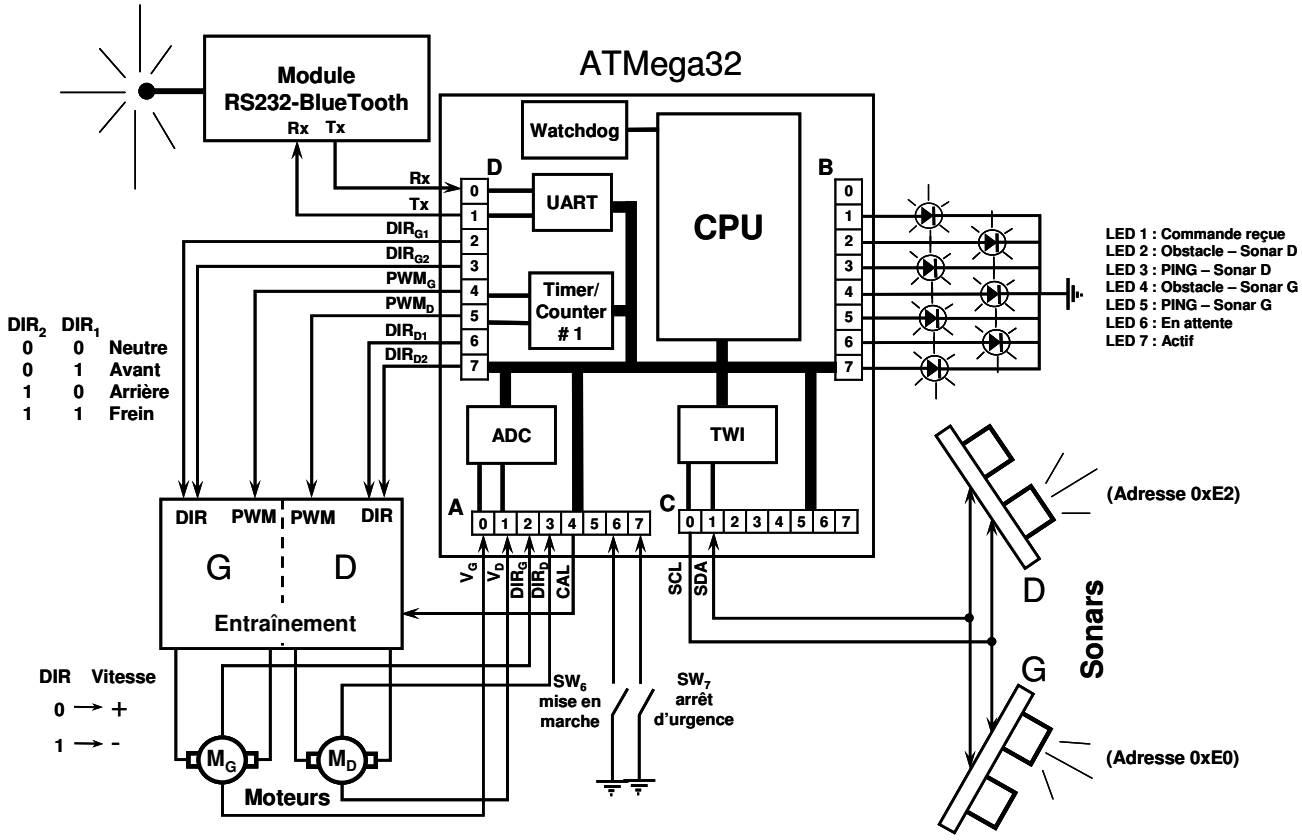


Fig. 1 – Schéma d'interconnections entre le robot et le microcontrôleur ATMega32

La plupart des fonctionnalités du microcontrôleur sont misent à contribution dans ce montage. Par exemple :

- **Chronomètre/compteur (Timer/counter)** :

  Génère un signal PWM qui contrôlera la vitesse des moteurs du robot, selon un mode d'opération qu'on retrouve très fréquemment dans les systèmes embarqués. En fait, la plupart des chronomètres/compteurs possèdent une fonction PWM qui sert dans ce type d'application.

- **Convertisseur A/D** :

  Vérifie la vitesse des moteurs par une mesure du courant, ce qui permet l'utilisation d'une boucle de rétroaction ou simplement d'un test diagnostique pour éviter les dépassements.

- **Module d'interface à deux fils (TWI - Two Wire Interface)** :

  Ce module utilise le protocole I2C qu'on retrouve comme unique mode de communication pour de nombreux périphériques. Le module servira pour la communication avec les deux capteurs de type sonar.

- **Module de communication série UART** :

  Servira à la réception des commandes de téléguidage et la transmission des informations. Ici aussi, cette interface de communication est très fréquemment utilisée dans les systèmes embarqués.

- **Ports parallèles du microcontrôleur** :

  Serviront pour le contrôle de la direction des moteurs ainsi que pour les lumières d'états et les interrupteurs de fonction. Encore une fois, c'est une utilisation typique d'une interface qu'on retrouve dans tous les systèmes embarqués.

- **Watchdog** :

  Le module Watchdog est habituellement utilisé pour détecter les problèmes de blocage du système. Dans le cas présent, il servira à vérifier que le lien de communication entre le téléguidage et le robot est actif en permanence. Si pour un raison ou une autre ce lien est brisé, alors le Watchdog redémarre immédiatement le système.

2.2 – Description de l'environnement de travail

La carte de développement STK500 est conçue pour être programmée à partir d'un ordinateur de type PC et est supportée par de nombreux outils logiciels disponibles gratuitement, tel que le compilateur GCC et l'environnement de développement AVRStudio. De plus, l'environnement de travail est conçu pour permettre aux étudiants qui le désirent de faire leur travail à la maison, moyennant l'achat de la carte de développement STK500 vendue à la COOP. Finalement, l'environnement de travail comprend un logiciel servant au téléguidage du robot et un logiciel de simulation du robot.

Logiciel de téléguidage :

Le logiciel de téléguidage est une application qui roule sur le PC et qui fait le lien entre une petite manette de jeu (Logitech Precision II) ou la souris et la carte de développement qui contrôle le robot. Le logiciel reçoit les commandes en provenance de la manette de jeu ou la souris et les transmet à la carte STK500 via le lien série.

En temps normal, le robot reçoit les commandes par lien Bluetooth grâce à un module RS232-Bluetooth installé sur l'ordinateur et un autre module RS232-Bluetooth connecté au port série de la carte STK500. Par contre, lors du développement, le lien Bluetooth peut est remplacé par un simple câble série qui relie l'ordinateur directement à la carte STK500.

De plus, le logiciel de téléguidage fournit une petite console qui permet d'afficher des messages transmis par le robot via le lien série et qui peut servir pour transmettre des messages de déverminage.

Logiciel de simulation :

Le logiciel de simulation émule le comportement du robot tels que le déplacement et la détection d'obstacles ainsi qu'un environnement virtuel composé d'obstacles et/ou de parcours tel qu'un labyrinthe. Plus précisément, le logiciel de simulation reçoit les commandes des moteurs (PWM, direction) produit par la carte STK500 et les convertit en déplacement dans l'environnement virtuel. Simultanément, le logiciel de simulation détecte la présence d'obstacles virtuels et émule le fonctionnement des capteurs sonars en transmettant les signaux appropriés vers la carte STK500 à la place des vrais capteurs.

Pour s'interconnecter à la carte STK500, le logiciel de simulation utilise uniquement des périphériques qu'on retrouve sur la plupart des ordinateurs personnels. Par exemple, les commandes de vitesse des moteurs (PWM) sont lues par les canaux gauche/droite du microphone (Line In) de la carte de son de l'ordinateur. Tandis que les commandes de moteurs (commandes de direction) ainsi que la communication I2C avec les capteurs sonars sont obtenus à l'aide du port parallèle de l'ordinateur. Grâce au logiciel de simulation du robot, le développement au laboratoire est fait principalement à l'aide d'un ordinateur PC et de la carte de développement STK500.

3.0 – Partie 1 : Système embarqué avant-plan/arrière-plan

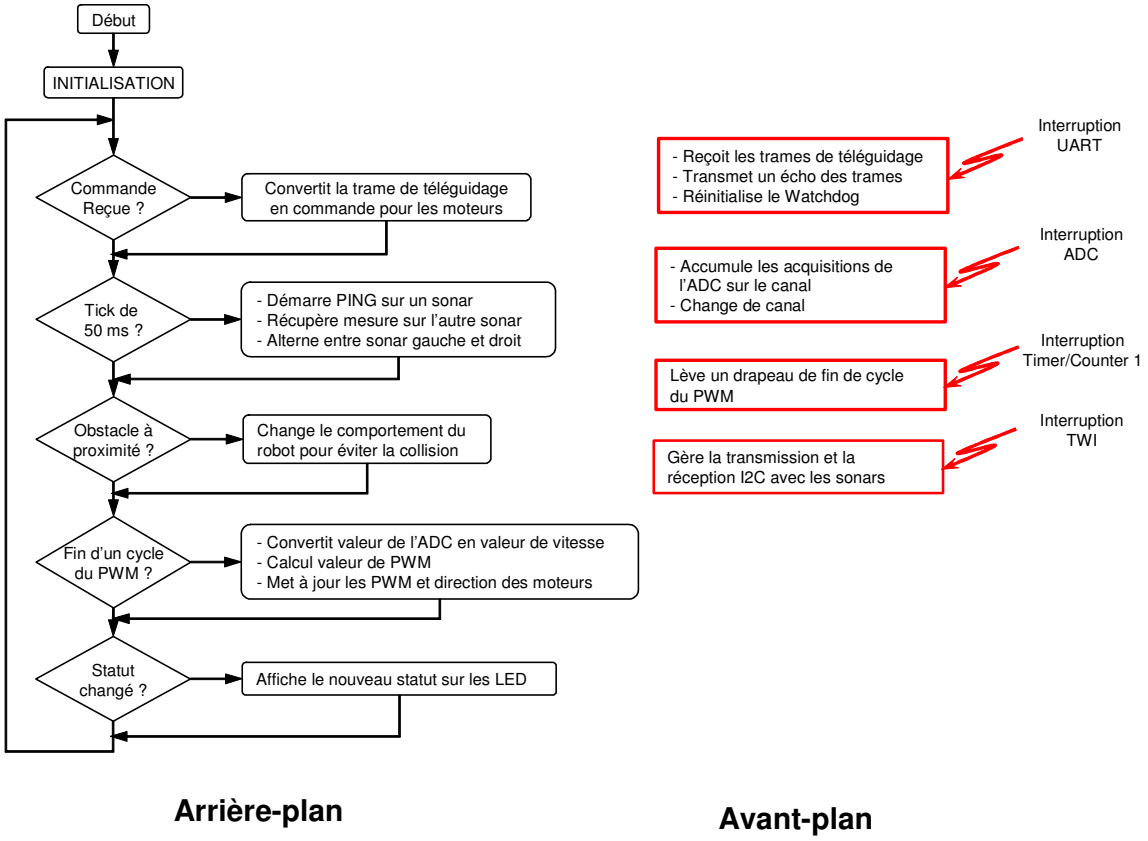
Le but de cette partie du projet est de développer un programme selon l'architecture avant-plan/arrière-plan qui gère les différentes composantes du robot. Selon cette architecture, tous les évènements synchrones ou asynchrones sont traités par des interruptions, tels que :

- l'arrivée d'une commande,
- fin d'un cycle du PWM,
- acquisitions de l'ADC,
- communication I2C.

Tandis que le gros du traitement est effectué dans le programme principal et les fonctions que celui-ci appelle. C'est-à-dire, l'arrière-plan sera responsable du traitement suivant :

- la conversion des trames de téléguidage,
- la conversion des acquisitions de l'ADC,
- la conversion des commandes en PWM,
- la gestion des sonars et détection des obstacles,
- gestion de l'affichage du statut du robot par les LED,
- gestion des interrupteurs de mise en marche et d'arrêt d'urgence.

La figure ci-dessous donne un aperçu (incomplet) de l'architecture générale du programme.



3.1 – Spécifications

- **Commandes de téléguidage** :

  - Chaque trame de téléguidage est composée de 3 données de 1 Byte :

Commande	**Vitesse**	**Angle**
0xF1 – commande normale 0xF0 – arrêt obligatoire	Valeur = 0 à 200 (vitesse = -100% à 100%)	Valeur = 0 à 180 (angle = 0 à 360^{o})

  - Les trames sont transmises continuellement par le téléguidage.
  - Le robot doit transmettre un écho de chaque donnée de trame reçue immédiatement.
  - La communication se fait par le UART selon le protocole suivant :
    - vitesse = 9600 Baud - 1 bit d'arrêt - Mode : Interruption
    - 8 bits de données - pas de parité

- **Envoie d'une trame de « debug »** :

  - Le téléguidage permet de recevoir des trames de « debug » et de les afficher dans l'espace blanc sous le contrôle d'angle et de vitesse. Pour ce faire, il suffit d'envoyer une chaine de caractère débutant par 0xFE et se terminant par 0xFF.
  - Pour éviter les problèmes lors de cette transmission, il faut cesser temporairement de transmettre l'écho de chaque donnée reçue.

- **Détection d'un bris de lien de téléguidage** :

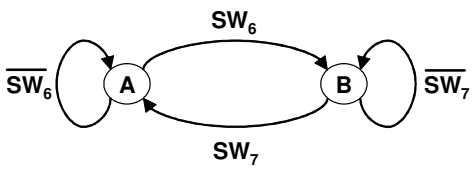
  - Le robot doit détecter dés que possible un bris de communication avec le téléguidage; c'est-à-dire, lorsque le robot ne reçoit pas de commandes de téléguidage pendant un certain intervalle de temps.
  - Le module Watchdog est particulièrement bien adapté pour cette tâche de surveillance. Il est programmé pour redémarrer le système (RESET) après un certain délai à moins qu'il ne soit réinitialisé avant la fin du délai.
  - Le Watchdog doit être configuré avec un délai judicieusement choisit et devra être réinitialisé à l'arrivée de chaque commande de téléguidage. Ainsi, si les

commandes de téléguidage tardent à arriver, le Watchdog redémarrera le système et le robot sera replacé dans son état initial, en attente d'être activé.

- **Affichage de l'état du robot par les LED** :
  - Les LED de la carte STK500 servent à afficher certaines informations sur l'état du robot.
  - La définition de chacune de ces LED est donnée dans le tableau ci-après :

# LED	Signification
0	Pas utilisée
1	Commande de téléguidage reçue (clignote)
2	Obstacle détecté par le sonar droit
3	PING envoyé par le sonar droit (clignote)
4	Obstacle détecté par le sonar gauche
5	PING envoyé par le sonar gauche (clignote)
6	Robot en attente (État A – voir interrupteurs)
7	Robot actif (État B – voir interrupteurs)

- **Interrupteurs de mise en marche et d'arrêt d'urgence** :
  - L'interrupteur # 6 permet d'activé le robot
  - L'interrupteur # 7 force un arrêt d'urgence du robot
  - La logique de fonctionnement de la mise en marche et de l'arrêt d'urgence est donnée par la machine à états finis ci-dessous :



État A : - Robot en attente
- Tous les systèmes sont inactifs

État B : - Robot actif
- Tous les systèmes sont en fonction

- **Calibration des moteurs** :
  - À cause de la variabilité des composantes du robot (tolérance des composantes, charge de la batterie, …), les mesures de vitesse des moteurs doivent être calibrées lors de l'initialisation du système.

- La calibration se fait en mesurant les vitesses maximum (positive et négative) et la vitesse nulle de chacun des deux moteurs (gauche et droite) de la façon suivante :

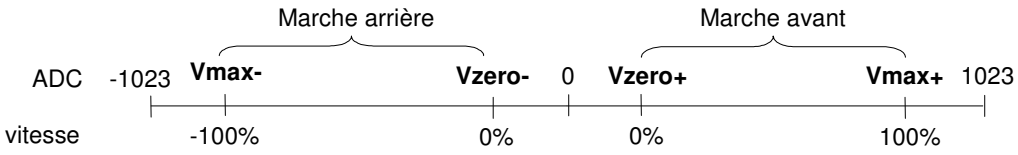
CAL	DIR_2	DIR_1	PWM	Mesure
1	0	1	0	**Vmax+**
0	0	1	0	**Vzero+**
0	1	0	0	**Vzero-**
1	1	0	0	**Vmax-**

où CAL = bit de calibration
DIR_1-DIR_2 = bits de direction du moteur } voir figure 1
PWM = signal de PWM du moteur

- La calibration est effectuée en prenant la moyenne d'une série de mesure pour chaque vitesse.
- Les deux signaux de PWM des moteurs doivent être maintenus à zéro pendant la procédure de calibration.
- Le bit de calibration (CAL) est maintenu à zéro pendant le fonctionnement normal du robot.

- **Mesure de la vitesse des moteurs** :
  - La vitesse des moteurs gauche et droit est mesurée par sur les canaux 0 et 1 de l'ADC, respectivement.
  - La configuration de l'ADC est la suivante :
    - Fréquence de l'horloge de l'ADC = 125 kHz
    - Mode d'opération = Freerun
    - Référence de voltage = AREF
    - Mode de traitement des acquisitions = Interruption
  - La routine d'interruption accumule les mesures dans une variable pour chaque moteur, dont la moyenne sera calculée plus tard lorsque la valeur de la vitesse sera utilisée pour réguler le PWM (voir Régulation de la vitesse des moteurs).

- Puisque l'ADC fournit une valeur entre 0 et 1023 et que les bits A2 et A3 nous permettent d'interpréter cette valeur comme positive ou négative, le rapport entre les valeurs fournies par l'ADC et la vitesse du moteur est le suivant :



- **Régulation de la vitesse des moteurs** :
  - Les vitesses de rotation des moteurs sont contrôlées par deux signaux de PWM générés par le chronomètre/compteur # 1 (Timer/Counter # 1). En effet, ce chronomètre/compteur de 16 bits est en fait double et peut générer 2 signaux de PWM indépendants avec une excellente résolution.
  - Le chronomètre/compteur # 1 doit être configuré de la façon suivante :

    f_{PWM} = 200 Hz

    PWM_B = moteur gauche

    PWM_A = moteur droit

    <u>Mode d'opération recommandé</u> :
    - Fast PWM
    - Interruption de débordement
    - Niveau haut au débordement
    - Niveau bas à la comparaison
  - Le sens de rotation des moteurs est contrôlé par 2 bits de direction (DIR_2, DIR_1) pour chaque moteur, selon la logique suivante :

DIR_2	DIR_1	Sens de rotation
0	0	Neutre
0	1	Marche avant
1	0	Marche arrière
1	1	Frein

  - Pour que les mouvements du robot correspondent aux commandes de téléguidage, la vitesse des moteurs doit être régulée par la fonction **CalculPWM** qui sera fournie. Le prototype de cette fonction est donné ci-dessous :

void **CalculPWM** (float Vitesse, float Angle, float Vg, float Vd, float *PWMg, float *PWMd)

où $-1.0 \leq \text{Vitesse} \leq 1.0$, $0.0 \leq \text{Angle} \leq 2\pi$ } Provient des commandes de téléguidage

$-1.0 \leq \text{Vg} \leq 1.0$, $-1.0 \leq \text{Vd} \leq 1.0$ } Provient des mesures de l'ADC

$-1.0 \leq \text{PWMg} \leq 1.0$, $-1.0 \leq \text{PWMd} \leq 1.0$ } Valeurs de PWM retournés par la fonction

Note : Les valeurs de vitesse et de PWM sont exprimées en %.

- Ainsi, la procédure pour réguler le comportement du robot est la suivante :

À tous les 5 ms

1 – Calculer la moyenne des échantillons recueillit par l'ADC.

2 – Exécuter la fonction **CalculPWM**.

3 – Mettre à jour les PWM et les bits de direction des moteurs.

- **Détection d'obstacle avec les sonars** :

- La communication avec les sonars est établie à l'aide du module TWI et selon le protocole de communication I2C. Ainsi, le TWI devras être configuré de la façon suivante :

Mode d'opération = - Traitement par interruption $f_{TWI} = 10$ kHz

- Mode maître

- pas d'adresse « esclave »

- Les deux sonars sont connectés sur le même bus TWI et ont leur adresse propre :

	Adresse	
	En lecture	**En écriture**
Sonar gauche	0xE1	0xE0
Sonar droit	0xE3	0xE2

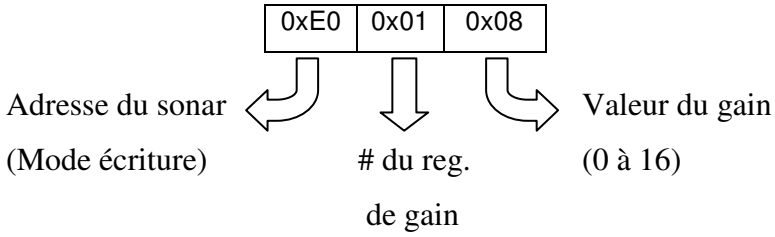
Note : En fait, le bit LSB de l'adresse détermine si une opération d'écriture (LSB = 0) ou de lecture (LSB = 1) sera effectuée par le maître (TWI) vers l'esclave (sonar).

- En plus de leur adresse personnelle, les sonars possèdent les 4 registres internes suivants :

No. registre	Fonction	
	En lecture	**En écriture**
0	# de révision - logiciel	Commande
1	inutilisé	Gain max
2	Mesure - MSB	Portée
3	Mesure - LSB	inutilisé

Gain : La valeur du gain détermine la sensibilité du micro du sonar. Un gain élevé rend le sonar plus sensible à des échos faibles tels que des échos lointains et des échos sur des surfaces molles. Par contre, cela rend aussi le sonar sujet aux mesures erronées.

La séquence de communication pour modifier la valeur du gain est la suivante :



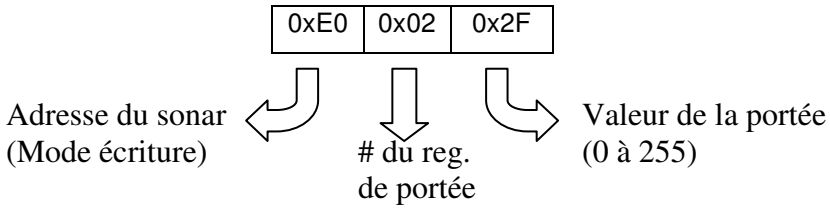
Portée : La valeur de la portée détermine le temps maximum d'écoute du sonar. Les échos qui reviennent après ce temps d'écoute sont ignorés. Ainsi, selon la vitesse du son dans l'air (environ 350 m/s), ce temps d'écoute détermine une durée de vol aller-retour et donc une distance maximum. La formule de calcul de la portée est la suivante :

Durée d'écoute (en µs) ≈ (Reg. de portée + 1) * 255 µs

ou

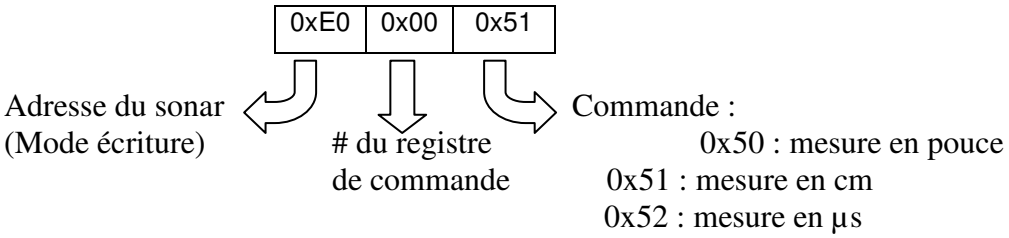
Portée (en cm) = (Reg. de portée + 1) * 4.3 cm

La séquence de communication pour modifier la valeur de la portée est la suivante :

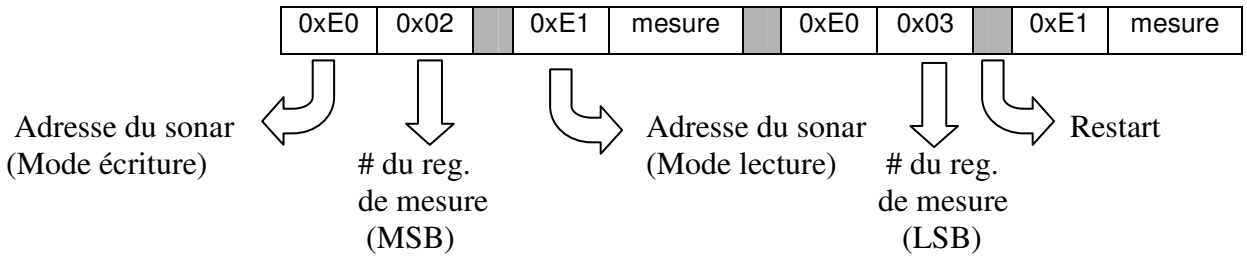


Commande : Le registre de commande permet de déclencher un PIng du sonar et d'effectuer une mesure selon un de 3 modes.

La séquence de communication pour modifier la valeur de la portée est la suivante :



Mesure : Le résultat d'un PING du sonar est fournit sous la forme d'un nombre non-signé de 16 bits, répartit dans 2 registres. Ces 2 registres doivent être lut séparément. La séquence de communication pour lire des registres exige que le numéro de registre soit transmit d'abord au sonar, avant de lire sa valeur :



- Puisque le robot possède 2 sonars et pour éviter qu'ils s'interfèrent l'un l'autre, les sonars devront faire leur PING en alternance avec un délai suffisant entre les

deux. Ainsi, la configuration et la gestion recommandées des sonars sont données ci-dessous :

- **Configuration** : Gain = 8 à 12

  Portée = 1.5 m à 3 m

  2 options : - portée fixe

  - portée proportionnelle à la vitesse

- Gestion : À tous les 50 ms

1 – Déclenche un PING dans un sonar,

2 – Récupère la mesure de l'autre sonar,

3 – Alterne les sonars.