Département de génie informatique et génie logiciel

INF1900
Projet initial de système embarqué

Rapport final de projet

Projet dans SimulIDE

Équipe No 5365

Section de laboratoire 2

Alexandre Gélinas
Ioana Daria Danciu
Maxence Sigouin
Vincent Grenier

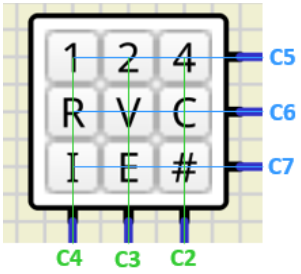
21 avril 2021

1. Description de la structure du code et de son fonctionnement

Les capteurs de distance

Les capteurs de distances permettent de déterminer la distance séparant le robot de ses obstacles hypothétiques. Les capteurs retournent des valeurs en voltage allant jusqu'à environ 2,5 volts, nous avons donc ajusté le voltage de référence des convertisseurs analogiques-numériques en conséquence, pour obtenir une valeur plus précise. Nous avons laissé la valeur retournée par le CAN interne sur 10 bits pour cette même raison. Pour déterminer une relation entre la distance détectée par les capteurs et la valeur qu'ils retournent, nous avons utilisé excel. Nous avons entré les valeurs retournées par les capteurs à plusieurs distances allant de 10 à 100 cm, et avons utilisé l'outil de courbe de tendances pour obtenir une fonction qui permet de retrouver la distance à partir de la valeur retournée par le convertisseur. Cependant, nous n'arrivions pas à faire des exposants dans le code, car pour une raison quelconque la valeur retournée par la fonction que nous utilisons n'était pas correcte. Nous avons donc linéarisé l'équation, en trouvant la relation entre l'inverse des valeurs retournées par les convertisseurs et la distance. Les formules que nous avons trouvées permettent de retrouver des distances qui sont environ à 3% de différence de la distance initiale dans les pires cas.

Le clavier



Le clavier sert à contrôler le comportement du robot. Afin de savoir quelle touche du clavier a été appuyée, nous avons séparé notre port en deux et fait deux lectures de celui-ci. En effet, le clavier peut être représenté sous forme de matrice, dont les colonnes équivalent à la première moitié du port, alors que l'autre moitié correspond aux lignes. Lors de la première lecture, nous rendons la première moitié des broches égale à 1, alors que la deuxième reste à 0. Ensuite, on inverse les bits pour la deuxième lecture. Ainsi, seule une partie du port émet un signal lors d'une lecture et lorsqu'on appuie sur l'une des touches, ce signal est transmis à l'une des broches qui était nulle auparavant. Par exemple, si l'on appuie sur le bouton « 1 », C4 passe de 0 à 1 lors de la première lecture et C5 passe de 0 à 1 lors de la deuxième. Il est donc ensuite possible d'obtenir le bouton appuyé à l'aide des deux ports qui ont changé de valeur.

Les interruptions

_____La première interruption est celle du clavier. Lorsqu'un bouton du clavier est appuyé, une interruption est lancée. Pour ce faire, nous regardons les broches du clavier pour savoir s'il y en a une qui est activée. Le bouton activé est ensuite sauvegardé. Nous remettons les broches aux valeurs initiales.

La deuxième interruption est celle du chrono. Lorsque la valeur du chrono est au maximum, l'interruption est appelée afin de répartir le chrono et de continuer la mesure du temps.

Il y a une troisième interruption qui est activée par le bouton poussoir, dont il est question un peu plus loin dans le rapport.

L'afficheur 7 segments

L'afficheur à 7 segments doit afficher le pourcentage de vitesse des moteurs, soit en décimal ou en hexadécimal. Notre classe Moteur a une méthode qui retourne la vitesse de ceux-ci, mais la valeur va de 0 à 255. La valeur est donc transformée en pourcentage à l'aide d'une règle de trois. Notre classe robot a un attribut affichage, qui indique si l'affichage se fait en hexadécimal ou en décimal. Chaque moteur a deux afficheurs, un pour les dizaines et un pour les unités. Nous faisons une division entière, par 10 pour afficher en décimal et par 16 pour afficher en hexadécimal, pour obtenir les dizaines à afficher et un modulo par les mêmes valeurs pour obtenir les unités. Les afficheurs ont un décodeur qui permet d'activer les bonnes entrées pour afficher, donc la valeur à afficher est passée directement au décodeur, avec une valeur qui indique l'afficheur à modifier.

Le bouton poussoir

____Le bouton-poussoir permet de passer du mode détection au mode démarrage. Si le bouton est appuyé, nous regardons l'antirebond et nous passons à la prochaine opération. La valeur est ainsi obtenue par interruption sur INT0.

Objectifs du robot

Notre classe robot a un attribut qui lui permet de savoir l'opération qu'il doit effectuer, par exemple détection ou manœuvre. Au départ, le robot doit effectuer une séquence de démarrage tel que décrit dans l'énoncé. Il est donc initialisé avec l'attribut démarrage, et alterne ensuite entre les deux autres pour le reste de l'exécution du programme. Le robot passe donc en mode détection suite au mode démarrage. Dans ce mode, il affiche la distance mesurée par les trois capteurs à une certaine fréquence. Cette fréquence est enregistrée dans un autre attribut du robot, qui lui permet de savoir lorsqu'il doit afficher les distances et lorsqu'il ne doit pas le faire. Lorsque le bouton poussoir est appuyé, le robot passe en mode manœuvre conformément à l'énoncé. Comme nous devions afficher la manœuvre associée aux différentes distances dans le mode détection, le robot sait laquelle il doit effectuer à chaque fois qu'il lit les capteurs. Il effectue donc la manœuvre associée aux distances mesurées par les trois capteurs. Les vitesses de chaque manœuvre sont données en pourcentage dans l'énoncé, mais les valeurs que nous devons passer aux moteurs sont sur une plage de 0 à 255. Nous avons donc recours à une fonction qui effectue une règle de trois pour déterminer la vitesse à passer à chaque moteur à partir du pourcentage. Nous ne faisons ensuite que changer la vitesse des moteurs et introduire des délais.

2. Expérience de travail à distance

Lors de ce laboratoire, nous avons uniquement communiqué grâce à Discord : lors de chaque séance de laboratoire, ou lors de nos rencontres en dehors de celles-ci, nous nous sommes appelés et avons partagé nos écrans afin de travailler tous ensemble et de s'aider plus facilement. Lors du premier travail pratique en équipe, nous nous sommes réparti les tâches, de façon à ce que chacun d'entre nous s'occupe de certains fichiers dans la librairie ou du Makefile. Cependant, nous nous promenions d'une tâche à l'autre, afin de s'assurer de la cohérence de notre travail.

Par la suite, lors des prochains travaux pratiques, nous avons adopté une organisation d'équipe plus centralisée : un membre de l'équipe, qui avait plus de connaissances sur le sujet, partageait son écran et décidait de la structure et du contenu du code, tandis que les autres l'aidaient à le garnir. Avant la rencontre avec le professeur, nous nous sommes appelés en ouvrant nos caméras afin de discuter. Cela nous a permis de développer notre énergie de solidarité ainsi que nos liens affectifs.

Lorsque le projet fut entamé, nous avons décidé de retourner à une organisation plus décentralisée, car cela donnait la chance à tous les membres de l'équipe d'écrire le code en tant que tel, ce qui garantissait la compréhension de tout le monde. De plus, lors de chaque étape, nous avons séparé l'équipe en deux et fait une rotation de partenaires, afin que nous puissions apprendre à mieux nous connaître. Par exemple, deux personnes travaillaient sur les capteurs, tandis que les deux autres s'occupaient du clavier. Par la suite, les équipes étaient encore changées et deux personnes arrangeaient les interruptions, alors que les autres codaient les manœuvres. À la fin de chaque étape, nous faisions une récapitulation de ce que nous avions fait dans nos équipes et fixions une nouvelle date d'échéance pour la prochaine partie du projet. Nous avons procédé ainsi jusqu'à la fin du projet.