BoucharD luka, renaud émile

Élaborer l'installation et l'intégration de systèmes ordinés

247-616-LI , gr.1

Bras robotisé :

**Six axes de liberté**

Travail présenté à

M. Yves Roy

Département de technologie du génie électrique

Cégep Limoilou

Le 6 novembre 2019

Table des matières

[Description : 2](#_Toc23935941)

[Démonstration : 2](#_Toc23935942)

[Planification : 2](#_Toc23935943)

[Analyse des risques : 2](#_Toc23935944)

[Requis client: 2](#_Toc23935945)

[Requis matériel : 3](#_Toc23935946)

[Requis logiciel : 4](#_Toc23935947)

[Architecture système : 5](#_Toc23935948)

[Architecture Matérielle: 5](#_Toc23935949)

[Architecture logicielle : 5](#_Toc23935950)

# Description :

Le projet consiste à construire un bras robotisé ayant 6 axes de rotation. Il ressemblera à ceux que l’on retrouve dans les usines, mais en modèle réduit. Il pourra être programmé simplement pour faire des tâches répétitives. Le bras pourra être contrôlé grâce à une interface graphique ou une interface console. Il sera équipé d’une pince pour prendre des objets.

# Démonstration :

Lors de la démonstration, le bras va bouger tous ses axes un à un pour montrer l’ampleur des mouvements qu’il peut faire. Il va ensuite bouger autour d’un point imaginaire fixe. Il va avoir un mini-programme préprogrammé qui va faire des mouvements à répétition comme dans une usine. Il sera aussi contrôlé à l’aide de l’interface manuellement par un être humain.

# Planification :

## Semaine 1 et 2:

Elles seront consacrées à la réalisation des schémas des pilotes des moteurs et de la carte d’interfaçage des périphériques du BeagleBone Black, pour l’écran LCD et la communication du BeagleBone Black avec les axes du bras. Aussi, il sera possible de tester le schéma des pilotes des moteurs sur un « protoboard ». Une fois les circuits testés, une première version des PCB pourra être conçu et commandé.

## Semaine 3 :

Elle sera consacrée au montage initial du bras axe par axe et de l’écriture d’un programme de contrôle de moteur. Il sera possible de tester chaque axe et de voir si le moteur peut être contrôlé. Le bras devrait fonctionner de façon très rudimentaire (contrôle sans suivi). Il n’y aura donc aucun capteur de position.

## Semaine 4:

Elle sera consacrée à la mise en marche et l’installation de tous les capteurs du bras. Le code associé à la lecture des capteurs devra être écrit. Des codes d’erreurs seront attribués aux diverses erreurs que nous trouverons lors de nos essais.

## Semaine 5 et 6 :

Elles seront consacrées à la communication LIN. Il faudra mettre en marche un protocole de communication stable nous permettant d’interagir avec les pilotes des moteurs. En parallèle, il sera possible de faire le programme d’interface graphique du LCD.

## Semaine 7 et 8 :

Elles seront consacrées à la finalisation de l’interface graphique du LCD et à l’écriture du programme final de contrôle des moteurs pas à pas.

## Semaine 9 et 10:

Elles seront consacrées à l’écriture du programme interface de contrôle des moteurs pas à pas (les rendre plus intelligents).

## Semaine 11 et 12:

Ces semaines seront réservées comme semaines de réserve au cas où il y aurait un imprévu. Advenant le cas où il n’y a pas d’imprévus, il sera possible d’ajouter une interface graphique PC avec ROS. Le rapport pourra aussi être complété lors de ces deux semaines.

# Livrables :

* 6 PCB de pilotes des moteurs.
* 1 PCB d’interfaçage des périphériques du BeagleBone Black.
* 6 axes du bras robotisé.
* 1 interface graphique pour l’écran LCD.
* 1 programme de contrôle des moteurs.
* 1 programme de gestion des capteurs.
* 1 programme de gestions des erreurs.
* 1 programme de communication LIN.
* 1 programme de contrôle intégral du bras robotisé.
* 1 schéma des pilotes des moteurs
* 1 schéma de l’interfaçage des périphériques du BeagleBone Black.
* 1 circuit d’alimentation fonctionnel dans le système.
* 1 interface graphique pour le PC (bonus)
* 1 rapport complet
* Acheter le matériel nécessaire à la mise en marche du bras.

# Analyse des risques :

Au cours du projet, il y a divers imprévus qui peuvent survenir. Il est possible que la livraison des pièces prend plus de temps qu’il avait été prévu. Si nous n’avons pas accès à certaines ressources matérielles, il est fort probable que nous ne puissions pas progresser dans l’horaire que nous nous étions fixé. Dans la même ordre d’idée, il est à considérer que certaines des pièces se brisent lors du transport

* Communication lin non fonctionnel
* Surchauffe des moteurs ou des pilotes des moteurs
* Interférence magnétique
* Pièce mécanique qui brise
* PCB brisé pendant le transport
* Grève scolaire
* Tempête
* PCB non fonctionnel
* Oublier de sauvegarder un fichier

# Requis client:

Le bras doit avoir 6 axes de rotation. Il sera muni d’une pince qui pourra soulever un poids d’au moins 250 grammes. S’il doit répéter un même mouvement plusieurs fois, il devra exécuter ces mouvements avec une imprécision d’au maximum 1 cm. Grâce à une interface, le bras sera programmable en GCode. Il y aura des capteurs de positions permettant de savoir si les axes du bras sont à la bonne position. Il y aura un système de code d’erreur permettant de décoder les diverses erreurs que les capteurs du bras nous envoient. Il doit avoir une interface usagée sur ordinateur ou directement sur le bras. Le bras devra être le plus silencieux possible. Chacun des axes du bras doivent être muni de leur propre contrôleur, pour faciliter la communication. Il est primordial d’utiliser la communication LIN pour parler avec les moteurs pas à pas (voir la Figure 1).

|  |  |
| --- | --- |
| Référence | Requis |
| 1 | Les 6 axes du robot doivent bouger |
| 2 | Le bras doit être capable de soulever une charge de 250 grammes. |
| 3 | Le bras doit pouvoir répéter un mouvement avec une imprécision d’au plus 1 cm |
| 4 | Le bras doit pouvoir être programmer facilement pour créer une séquence de mouvement |
| 5 | Le bras doit être muni d’une pince |
| 6 | Le bras doit corriger les erreurs qu’il fait dans les mouvements à l’aide de capteur de positions. |
| 7 | Le bras doit se brancher facilement à un ordinateur |
| 8 | Le bras doit être le plus silencieux possible |
| 9 | Les moteurs doivent être équipé de leur propre contrôleur. |
| 10 | Les moteurs doivent se parler en LIN bus |
| 11 | Interface graphique pour contrôler manuellement les axes. |
| 12 | Le bras robotisé sera muni d’un système de codes d’erreur. |

Figure : Tableau des requis clients

# Requis matériel :

Le projet se base sur le bras à 6 axes WE-R2.4. Il aura 3 moteurs NEMA 17 pour les 3 plus grosses articulations. Les trois autres articulations seront munies de moteur pas à pas aussi, mais de plus petits moteurs 35 mm pour des questions d’optimisation de l’espace disponible. Pour contrôler la pince, un moteur DC sera utilisé. Afin de savoir la position des moteurs, ils seront chacun munis d’encodeur de position magnétique. Chaque axe du bras à son propre module. Ils possèdent donc tous un microcontrôleur STM8. Pour permettre au bras d’avoir une fonction « homing », il sera muni d’un capteur à effet Hall. Pour refroidir les pilotes des moteurs et le moteur pas à pas, des ventilateurs seront installé près des PCB et des moteurs. Il y aura aussi plusieurs capteurs de température, pour s’assurer que la température est bien régulée (changer la vitesse des ventilateurs). Des dissipateurs de chaleurs seront aussi utilisé à des fins de refroidissement. Un BeagleBone Black sera utilisé pour contrôler les modules de chaque axe. Il sera connecté à un écran qui permettra d’afficher en temps réel la position du bras ainsi que les diverses codes d’erreur. (voir la Figure 2)

|  |  |  |
| --- | --- | --- |
| Identification | Référence | Requis matériel |
| M1 | 1 | Moteur NEMA 17 pour les moteurs de puissance |
| M2 | 2 | Moteur pas à pas (35mm) pour les petites articulations |
| M3 | 3 | Contrôleur de moteur intégré sur chaque moteur |
| M4 | 4 | 1 PCB par moteur (contrôleur + encodeur) |
| M5 | 5 | Encodeur de position magnétique pour moteur pas à pas |
| M6 | 6 | STM8 comme contrôleur sur chaque pilote de moteur (QFP or SSOP) |
| M7 | 7 | Capteur à effet Hall pour le retour à l’origine des moteurs |
| M8 | 8 | Capteur de température (TO-92-3) |
| M9 | 9 | Mini ventilateur pour chaque pilote de moteur (refroidissement actif ) |
| M10 | 10 | Dissipateur de chaleur pour les pilotes de moteur (refroidissement passif) |
| M11 | 11 | Moteur DC pour la pince |
| M12 | 12 | Écran TFT LCD 7’’ |
| M13 | 13 | Bras WE-R2.4 |
| M14 | 14 | 1 PCB pour l’interface du BeagleBone Black et de ses périphériques. |
| M15 | 15 | 1 BeagleBone Black |

Figure  : Tableau des requis matériels

# Requis logiciel :

(voir la Figure 3)

|  |  |  |
| --- | --- | --- |
| Identification | Référence | Requis logiciel |
| S1 | 1 | Logiciel ROS (Bonus) |
| S2 | 2 | Contrôleur de moteur programmable via ST-Link |
| S3 | 3 | Utilisation d’un système de gestion de fichiers GIT |
| S4 | 4 | 1 pilote de direction des moteurs pas à pas |
| S5 | 5 | 1 pilote de capteur à effet Hall |
| S6 | 6 | 1 pilote de capteur de température |
| S7 | 7 | 1 pilote d’encodeur magnétique |
| S8 | 8 | Interface graphique en C# pour le LCD |
| S9 | 9 | 1 service de communication LIN |
| S10 | 10 | 1 pilote pour l’écran LCD |
| S11 | 11 | 1 pilote de contrôle des ventilateurs |
| S12 | 12 | 1 interface de gestion des codes d’erreurs |
| S13 | 13 | 1 service du TMC2130 (pilote de moteur) |
| S14 | 14 | 1 interface pour l’écran LCD |
| S15 | 15 | 1 pilote de synchronisation d’horloge |

Figure  : Tableau des requis logiciels

# Architecture système :

(voir la Figure 4)

<https://www.draw.io/?lightbox=1&highlight=0000ff&edit=_blank&layers=1&nav=1#G1Wr0yj4o1g_6H85Xn3UtQ55oE_WySvxa3>

Figure  : Schéma de l’architecture système

# Architecture Matérielle:

(voir la Figure 5)

<https://www.draw.io/?lightbox=1&highlight=0000ff&edit=_blank&layers=1&nav=1#G1TSFSJ0K5nXtpQTIo7lhhtFirrGxJPg1B>

Figure  : Schéma de l’architecture matérielle

# Architecture logicielle :

(voir la Figure 6)

<https://www.draw.io/?lightbox=1&highlight=0000ff&edit=_blank&layers=1&nav=1#G1YRsY8k0mPz8Nq5unQ-XQLZ3r_vWQHjLO>

Figure  : Schéma de l’architecture logicielle