Maquette Flying Arm

Pupitre de commande

Narbey Robin
Elec3-2015

Boitier de commande de la maquette

Le boitier de commande de la maquette contient l'alimentation électrique du système, le module de commande du moteur brushless ainsi que le microcontrôleur. La partie commande est découpler de la partie puissance, c'est-à-dire que les masses entre la puissance et la parties commandes ne sont pas communes. Le bouton d'arrêt d'urgence coupe l'alimentation du contrôleur de moteur mais pas de toute la maquette.

Interface Homme-Machine

1. Introduction

En plus du clavier l'utilisateur dispos d'un potentiomètre libre d'utilisation, de 4 LEDS et d'un bouton simple appui et du clavier matriciel. Ces entrés sorties sont directement câblées sur le microcontrôleur.

2. Câblage

Tout est câblé sur la plus grandes des deux nappes multicolore dans le boitier. L'alimentation de la petite carte HIM ce fait grâce aux 3V disponible sur la carte stm32.

Couleur fil	Pin stm32	Fonction
Jaune	GND	GND
Orange	VDD	3V
Rouge	PC1	LED0
Marron	PC2	Potentiomètre
Noir	PC3	Start/Stop
Blanc	PA0	LED1
Gris	PA1	Clavier bit0
Violet	PA2	Clavier int
Bleu	PA3	Clavier bit2
Vert	PA4	Clavier bit1
Jaune	PA5	LED2
Orange	PA6	Clavier bit3
Rouge	PA7	BUZZER
Marron	PB11	LED3

3. Précisions

Le câblage est fait de telle sorte que tous les signaux soit en logique positive, par exemple un état haut sur PA0 allume la led 1. Au 22/07/2015 le buzzer n'est pas connecter car il nécessite un signal périodique pour fonctionner, il faudra aussi ajouter un transistor pour piloter le buzzer par une sortie du microcontrôleur.

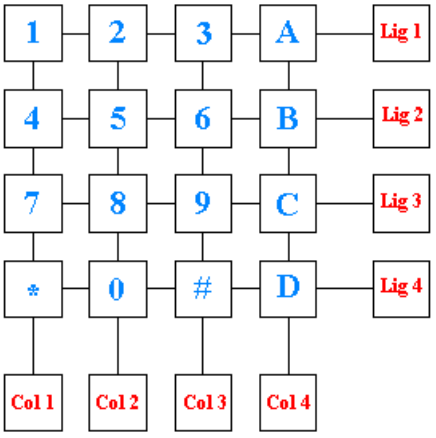
Driver de clavier matriciel 16 touches

1. Introduction :

Les claviers matriciels sont très utilisés encore aujourd'hui car ils sont relativement solides et bon marché, cependant leur contrôle est un peu délicat. Nous avons donc décidé d'utiliser un clavier 16 touches pour réaliser l'interface homme-machine de notre projet ainsi qu'un driver pour faciliter la gestion du clavier par le microcontrôleur. Le clavier est équipé de 16 touches : les chiffre de 0 à 9, les lettres A B C D ainsi que le * et #.

2. Construction des claviers :

Ce sont de simples matrices d'interrupteurs, lorsqu'on appuie sur une touche, l'utilisateur court-circuite la ligne avec la colonne correspondante, on peut donc déterminer la touche avec un montage adapté.



Voici le brochage du clavier que nous avons utilisé : (quand on regarde le clavier de face, avec le 1 en haut à gauche)

De gauche à droite :

Broche	1	2	3	4	5	6	7	8	9	10
Fonction	NC	Col 1	Col 2	Col 3	Col4	Ligne 1	Ligne 2	Ligne 3	Ligne4	NC

3. Construction du driver :

Pour le driver nous avons choisi d'utiliser un petit microcontrôleur PIC de type 16F84A, il dispose d'un port bidirectionnel de 8 bits, parfait pour connecter le clavier et d'un second port de 5 bits qui servira à communiquer avec le microcontrôleur principal. De plus, il supporte une tension d'alimentation entre 3,3V et 5V.

Le driver s'utilise très simplement : le pic surveille l'appui sur les touches et réalise aussi une fonction d'anti-rebond pour éviter l'apparition de données erronées. Quand une nouvelle touche est appuyée, le code correspondant à la touche enfoncée est mis sur les 4 premiers bits du port de communication jusqu'à ce qu'une nouvelle touche soit détectée. De plus, à chaque nouvelle donnée disponible, une impulsion est déclenchée sur le Bit5 du port de communication, ce signal peut être utilisé pour provoquer une interruption sur le microcontrôleur principal.

Voici la table qui donne les correspondances entre la touche active et le code en binaire correspondant placé sur le port de communication :

Touche	Code	Touche	Code	touche	Code	Touche	Code
0	0000	4	0100	8	1000	A	1100
1	0001	5	0101	9	1001	B	1101
2	0010	6	0110	*	1010	C	1110
3	0011	7	0111	#	1011	D	1111

4. Principe de détection :

Concernant le fonctionnement, le PORTB sur lequel est branché le clavier est partagé en deux parties : 4 bits sont configurés en sortie, sur lesquelles on viendra connecter les colonnes, et 4 bits en entrée pour connecter les lignes. A l'exécution du programme, chaque colonne est mise à l'état haut pendant que les 3 autres sont mises à l'état bas tour à tour. Les 4 entrées pour connecter les lignes sont équipées d'un tirage à la masse.

Le PIC génère les signaux sur les colonnes et observe les entrées, si les entrées ne sont pas à 0, on regarde le numéro de la colonne active et le numéro de la ligne active afin de déterminer la touche. Cette méthode ne supporte pas l'appui sur plusieurs touches simultanément.

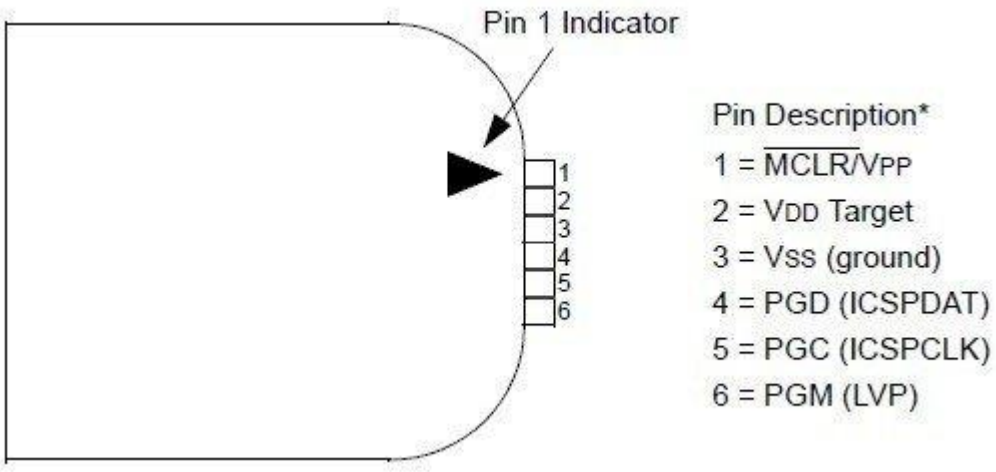
5. Solution de remplacement :

Ce montage avec un PIC16F84A peut facilement être remplacé par le circuit 74C922 qui est un circuit discret spécialisé dans le décodage de clavier matriciel.

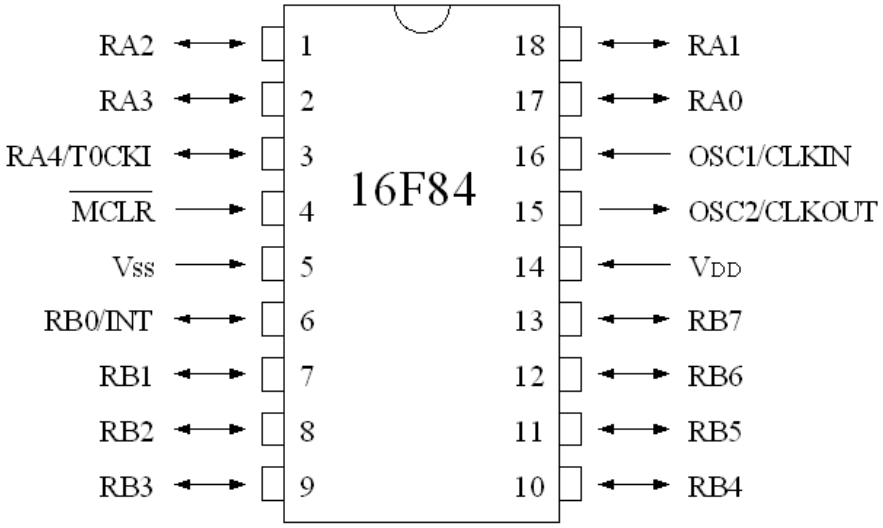
6. Programmation du pic

Pour programmer le pic nous avons utilisé un Pickit3 et nous avons programmé le microcontrôleur avant de le placer sur la carte. Nous avons utilisé l'IDE de Microchip MPLABX avec le compilateur XC8.

Pour programmer avec le pickit il suffit simplement de connecter les broches du pickit aux broches du pic selon la documentation constructeur :



MCLR : MCLR ; VDD : VDD ; VSS : VSS ; PGD : RB7 ; PGC : RB6 ; PGM : Inutile



Note : Quand nous avons programmé le pic nous avions branché un quartz sur les entrées OSC1/OSC2 nous ne savons pas s'il est indispensable. L'alimentation peu être fournie directement par le pickit, il suffit d'activer l'option dans MPLAB.

Afficheur Alphanumérique

1. Introduction

Nous avons utilisé un afficheur alphanumérique pour afficher les informations sur l'état courant de la maquette. Cet afficheur dispose de 2 lignes de 20 caractères chacune. Le driver de l'écran est un chip Hitachi HD44780, il existe donc un bon nombre de bibliothèque déjà faite pour cet écran car ce driver équipe la plus part des écrans alphanumérique. L'écran dispos d'un rétroéclairage pour l'utiliser il faut connecter une résistance (50 ohm sous 5 V environ) en série de la patte « + »

2. Câblage de l'écran

Pinout de l'afficheur Alpha-Numérique

Format : 2*20 caractères

Chip driver : HD44780

Consommation : 10mA (sans retro éclairage)

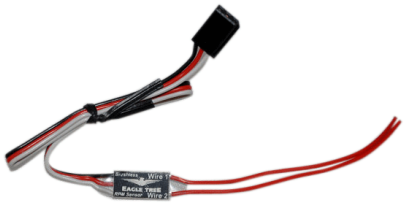
N° Broche Aff	Fil	Description	Pin Stm32F4
1	Jaune	GND	GND
2	Orange	VCC	VCC
3	Rouge	Contraste	GND
4	Vert	Registre Select	PE13
5	Bleu	Read/Write	GND
6	Violet	Enable (clock)	PE11
7	NC	Bit0	NC
8	NC	BIt1	NC
9	NC	Bit2	NC
10	NC	Bit3	NC
11	Gris	Bit4	PE15
12	Blanc	Bit5	PE9
13	Noir	Bit6	PE8
14	Marron	Bit7	PE7

3. Commande

Pour interfacer l'écran avec la board stm32F4 nous avons utilisé une bibliothèque de fonction open source trouvé sur internet.

Lien : http://stm32f4-discovery.com/2014/06/library-16-interfacing-hd44780-lcd-controller-with-stm32f4/

Capteur de vitesse Eagle Tree



1. Introduction

Le capteur de vitesse Eagle Tree RPM Sensor V2 est utilisable sur tous les moteur Brushless de type modélisme, le capteur détecte le passage à zéro du courant dans les bobines du moteur pour obtenir la vitesse de commutation des phases du moteur. Cette information est l'image de la vitesse de rotation du moteur à un facteur près (le nombre de pôles du moteur).

2. Câblage

Concernant le câblage du capteur il faut brancher l'entrée Wire 1 sur une phase quelconque du moteur et Wire 2 sur la phase suivante. En théorie le capteur peut fonctionner en ne connectant que Wire 1 à une seul phase et en laissant Wire2 flottant, cependant cela dépends du moteur, en effet sur les moteurs que nous avons utilisé le capteur ne fonctionnais pas sans connecter les deux fils.

Le connecteur trois fils quant à lui doit être câblé comme suit :

- Blanc : Signal
- Rouge : Masse
- Noir : +5V (VCC)

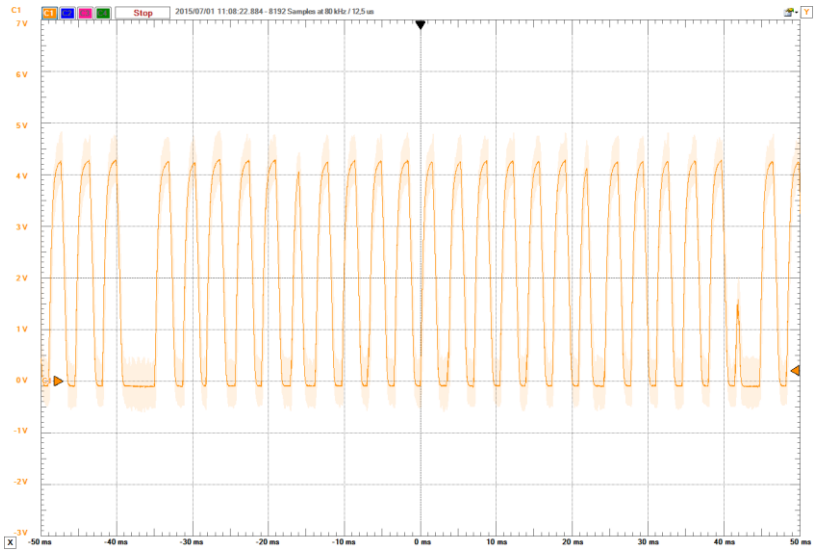
Le fils rouge est bien la masse, ce n'est pas une erreur, le constructeur a certainement fait cela pour éviter que l'on puisse utiliser ce capteur sans acheter le système « datalogger» qui va avec.

3. Informations du capteur

Le Capteur fourni sur la sortie un signal carré de fréquence variable avec vitesse de rotation du moteur. Plus la vitesse de moteur augmente plus la fréquence du signal de sortie augmente. Le signal de sortie n'est pas un signal très propre, il faut

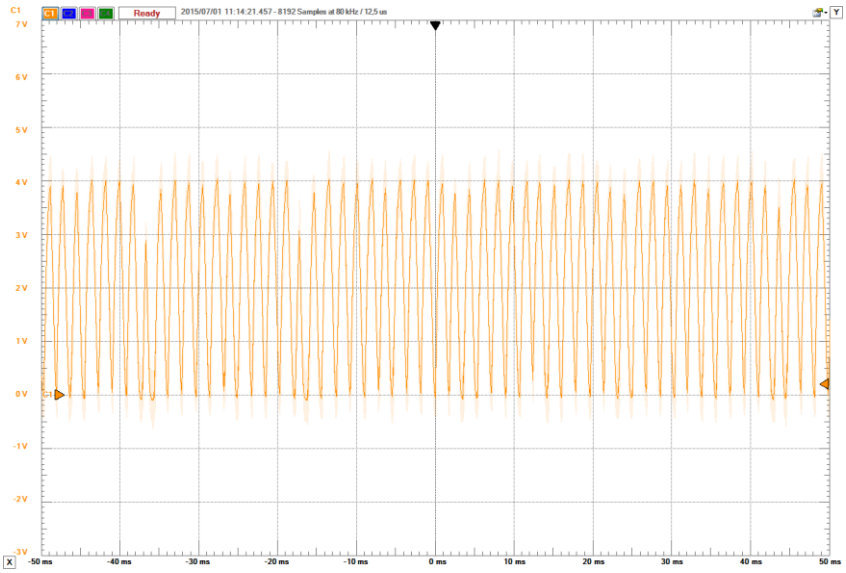
4. Oscillogrammes de la sortie du capteur

Vitesse de rotation faible du moteur :



Ch	Name	Value
C1	Average	1,97 V
C1	Frequency Avg.	267,66 Hz
C1	Pos Duty Avg.	50,61 %
C1	Pos Duty Min.	28,22 %

Vitesse de rotation moyenne du moteur :



Ch	Name	Value
C1	Average	2,06 V
C1	Frequency Avg.	531,98 Hz
C1	Pos Duty Avg.	54,96 %
C1	Pos Duty Min.	26,82 %

5. A faire

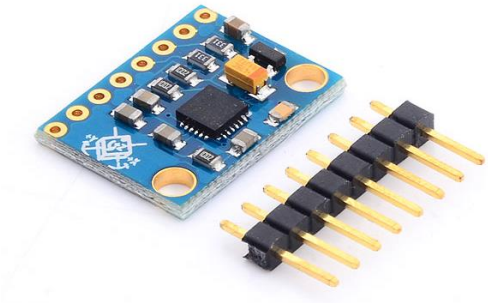
Le 22/07/2015 le capteur n'est pas câblé sur la carte et le code pour traiter ce capteur n'est pas fonctionnel, il faudra utiliser un module CCP de la carte et peut être un trigger de Schmidt avant d'attaquer l'entrée du microcontrôleur pour obtenir un signal plus propre pour le traiter dans de bonnes conditions avec l'uc.

Capteur potentiométrique et IMU

1. Capteur d'angle

L'angle du bras est déterminer par un potentiomètre, il est alimenter par le 3V de la carte stm32. Le point milieu est connecté sur la broche PC0 qui correspond au chanel 10 de l'ADC du stm32. Le signal est ramené au boitier de commande par le câble avec connecteur DB9.

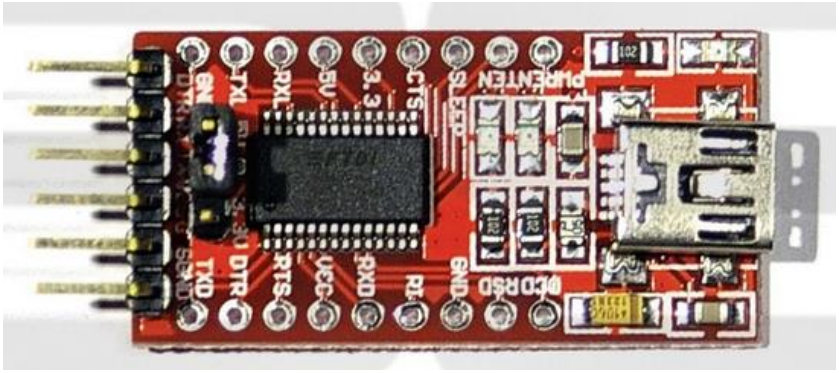
2. IMU



Le bras est équipé d'une centrale inertiel de référence MPU 6050, la communication avec la carte ce fait par protocole I2C, au 22/07/2015 la liaison entre le capteur est la carte n'est pas réalisée. Cependant le capteur s'interface facilement avec la carte grâce à cette bibliothèque open source :

Lien : http://stm32f4-discovery.com/2014/10/library-43-mpu-6050-6-axes-gyro-accelerometer-stm32f4/

Convertisseur USB-Série



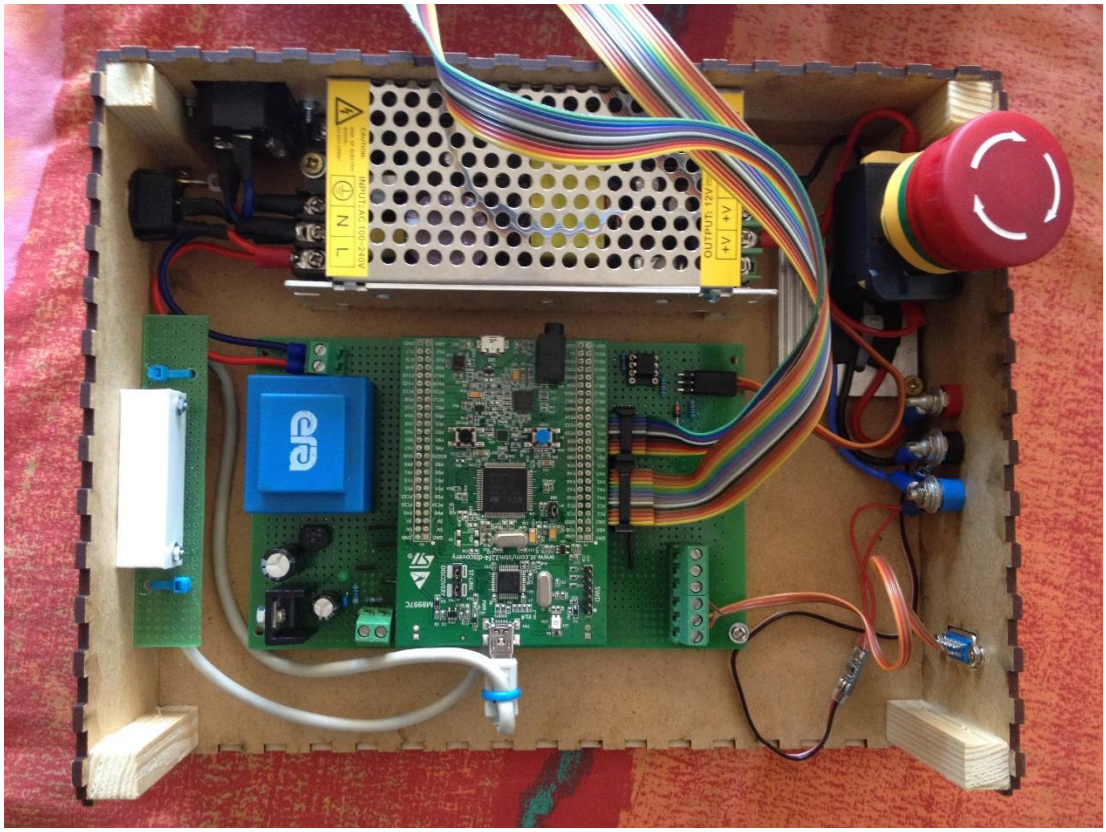
1. Utilisation

Ce module permet de brancher la maquette à un pc par un port USB et de communiquer avec un protocole RS232.

Le câblage est réalisé comme suit : Pins PB6 (TX) et PB7 (RX).

Pour mettre en œuvre ce module il suffit d'activer la liaison série sur les pins PB6 et PB7 du stm32, et du coter du pc il faut ouvrir un port COM virtuel.

Câblage intérieur du boitier



Le premier port USB permet de connecter le stm32 à un pc pour le programmer, le second permet de relier la maquette à un pc par une liaison série pour échanger des informations. Le connecteur DB9 permet de connecter les capteurs de la maquette au boitier de commande. Les trois connecteurs banane permettent de connecter le moteur brushless.

La petite nappe permet de connecter l'écran LCD, la deuxième nappe permet de connecter l'interface homme machine.