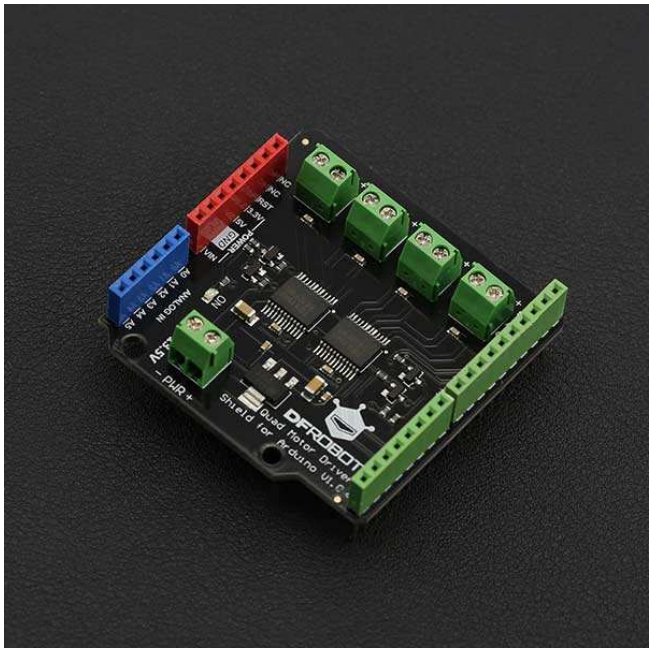
Carte interface moteur

DFROBOT Quad Motor Driver - SKU_DRI0039

https://wiki.dfrobot.com/Quad_Motor_Driver_Shield_for_Arduino_SKU_DRI0039

Introduction

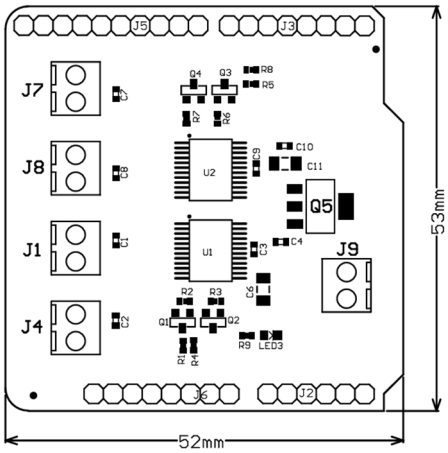
La carte interface de pilote de moteur quad pour Arduino, compatible avec le contrôleur Arduino 5 V/3,3 V, peut contrôler jusqu'à quatre moteurs CC avec 8 broches en même temps. La carte interface prend en charge le contrôle de vitesse PWM et le contrôle de polarité. La carte interface comprend deux circuits de pilotage de moteur TB6612FNG, utilisant la puce L298N traditionnelle, l'efficacité est améliorée et la taille des composants est également considérablement réduite. La puce ne chauffe pas dans la plage nominale avec un courant continu de sortie maximale de 1,2 A. Le module comprend un circuit de détection de tension basse et un circuit de protection thermique, sûrs et fiables.



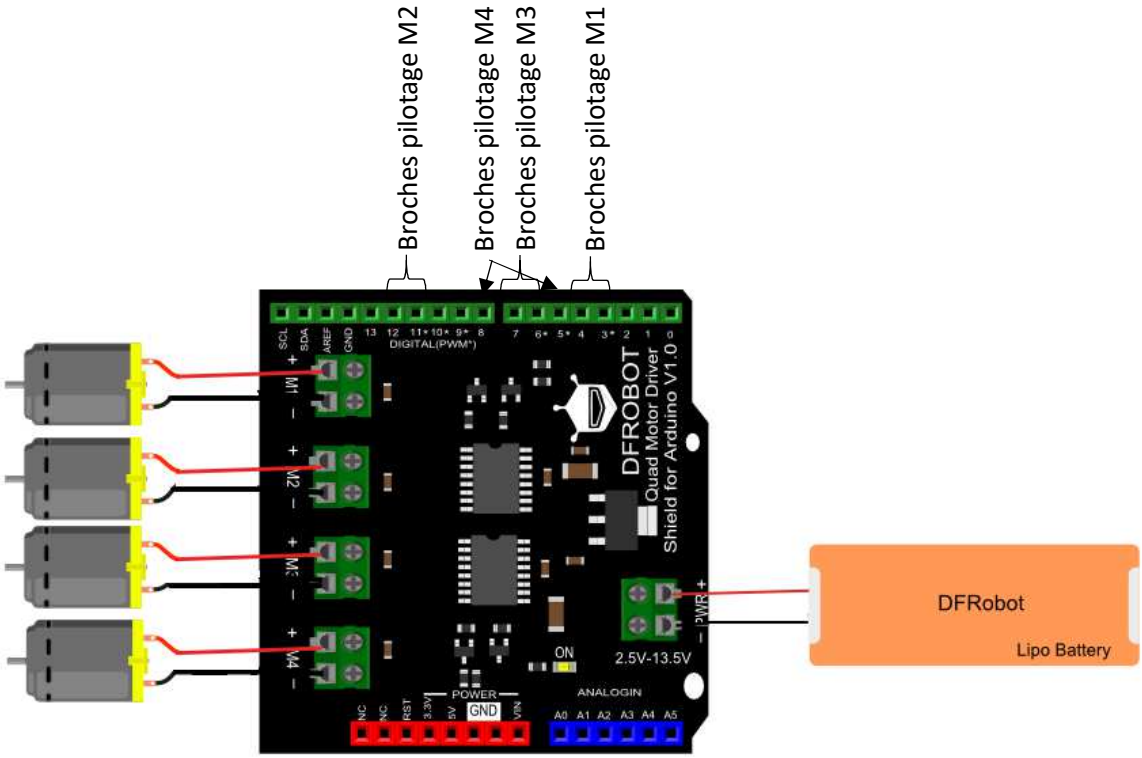
Spécification

- Tension d'alimentation ses moteurs : 2.5 V ~ 13.5 V
- Tension d'alimentation logique de commande VCC : 2.7 ~ 5.5 V
- Intensité de sortie : 1.2A courant continu (par moteur)
- Intensité de pointe: 2A (impulsion continue) / 3.2A (simple impulsion)
- Température de fonctionnement : -20 ~ 85 ℃
- Dimensions : 53mm x 52mm/2.09 x 2.05

Dimension



Brochage de la carte



Numéro des broches sur la carte Arduino UNO

	Pilotage sens (TOR)			Pilotage vitesse (PWM)	
Moteur	**Broche Sens**	**Forward**	**Backward**	**Broche Vitesse**	**Speed range**
M1	4	LOW	HIGH	3	0-255
M2	12	HIGH	LOW	11	0-255
M3	8	LOW	HIGH	5	0-255
M4	7	HIGH	LOW	6	0-255

Numéro des broches sur la carte ESP32 WROOM D1 R32.

	Pilotage sens (TOR)			Pilotage vitesse (PWM)	
Moteur	**Broche Sens**	**Forward**	**Backward**	**Broche Vitesse**	**Speed range**
M1	4	LOW	HIGH	3	0-1023
M2	12	HIGH	LOW	11	0-1023
M3	8	LOW	HIGH	5	0-1023
M4	7	HIGH	LOW	6	0-1023

Exemple de code

```python
# ==============================
#
# Ce programme fait tourner le moteur Câblé sur le connecteur M1 à 2 vitesses différentes dans un sens puis dans l'autre sens
#
# Câblage de la carte Driver sur l'ESP32 :
#
# Motor  Pin Direction     Forward Backward      Pin Speed       Speed range
#   M1    GPIO17 (D4)        LOW    HIGH         GPIO25 (D3)     0-1023
#   M2    GPIO19 (D12)       HIGH   LOW          GPIO23 (D11)    0-1023
#   M3    GPIO12 (D8)        LOW    HIGH         GPIO16 (D5)     0-1023
#   M4    GPIO14 (D7)        HIGH   LOW          GPIO27 (D6)     0-1023
#
# ==============================

from machine import Pin,PWM # importation de la bibliotheque relative au hardware de la carte ESP32

import time # importation de la bibliotheque temps

Sens_Moteur_M1 = Pin(17, Pin.OUT)        # broche 17 de l'ESP32 en sortie (repère D4 shield base 1)
PWM_Moteur_M1=PWM(Pin(25), 10000)        # attache un objet nommé PWM_Moteur de type MLI de fréquence 5000Hz sur broche 25 de l'ESP32 (D3)
Sens_Moteur_M2 = Pin(19, Pin.OUT)        # broche 17 de l'ESP32 en sortie (repère D4 shield base 1)
PWM_Moteur_M2=PWM(Pin(23), 10000)        # attache un objet nommé PWM_Moteur de type MLI de fréquence 5000Hz sur broche 25 de l'ESP32 (D3)
Sens_Moteur_M3 = Pin(12, Pin.OUT)        # broche 17 de l'ESP32 en sortie (repère D4 shield base 1)
PWM_Moteur_M3=PWM(Pin(16), 10000)        # attache un objet nommé PWM_Moteur de type MLI de fréquence 5000Hz sur broche 25 de l'ESP32 (D3)
Sens_Moteur_M4 = Pin(14, Pin.OUT)        # broche 17 de l'ESP32 en sortie (repère D4 shield base 1)
PWM_Moteur_M4=PWM(Pin(27), 10000)        # attache un objet nommé PWM_Moteur de type MLI de fréquence 5000Hz sur broche 25 de l'ESP32 (D3)

sens=0                                   # Variable image du sens de rotation du moteur (0 pour le sens forward, 1 pour le sens backward)
vitesse=0                                # Variable image de le vitesse de rotation du moteur (comprise entre 0 et 1023)

Sens_Moteur_M1.value(sens)               # positionne dans le sens forward la rotation du moteur M2
PWM_Moteur_M1.duty(vitesse)              # pilote le moteur M2 à l'arret

Sens_Moteur_M2.value(sens)               # positionne dans le sens forward la rotation du moteur M2
PWM_Moteur_M2.duty(vitesse)              # pilote le moteur M2 à l'arret

Sens_Moteur_M3.value(sens)               # positionne dans le sens forward la rotation du moteur M3
PWM_Moteur_M3.duty(vitesse)              # pilote le moteur M2 à l'arret

Sens_Moteur_M4.value(sens)               # positionne dans le sens forward la rotation du moteur M4
PWM_Moteur_M4.duty(vitesse)              # pilote le moteur M2 à l'arret


def rotation_M1(v, s):
    Sens_Moteur_M1.value(int(s))         # positionne dans le sens forward la rotation du moteur
    PWM_Moteur_M1.duty(v)                # envoie la commande de vitesse à la carte shield


while True:             #boucle infinie puisque la condition est toujours vraie(True)

    sens=0                               # définit le sens forward
    vitesse=0                            # définit la vitesse 0 (vitesse nulle)
    rotation_M1(vitesse,sens)            # appelle la fonction rotation_M1 en lui donnant les apramètre définits précédemment)
    time.sleep(1)                        # pause de 1s

    vitesse=512                          # définit la vitesse 512 (moitié de 1023 -> Vitesse proche de la moitié de la vitesse max)
    rotation_M1(vitesse,sens)            # appelle la fonction rotation_M1 en lui donnant les apramètre définits précédemment)
    time.sleep(1)                        # pause de 1s

    vitesse=1023                         # définit la vitesse 1023 (Vitesse max)
    rotation_M1(vitesse,sens)            # appelle la fonction rotation_M1 en lui donnant les apramètre définits précédemment)
    time.sleep(1)                        # pause de 1s

    sens=1                               # définit le sens backward
    vitesse=0                            # définit la vitesse 0 (vitesse nulle)
    rotation_M1(vitesse,sens)            # appelle la fonction rotation_M1 en lui donnant les apramètre définits précédemment)
    time.sleep(1)                        # pause de 1s

    vitesse=512                          # définit la vitesse 512 (moitié de 1023 -> Vitesse proche de la moitié de la vitesse max)
    rotation_M1(vitesse,sens)            # appelle la fonction rotation_M1 en lui donnant les apramètre définits précédemment)
    time.sleep(1)                        # pause de 1s

    vitesse=1023                         # définit la vitesse 1023 (Vitesse max)
    rotation_M1(vitesse,sens)            # appelle la fonction rotation_M1 en lui donnant les apramètre définits précédemment)
    time.sleep(1)                        # pause de 1s
```