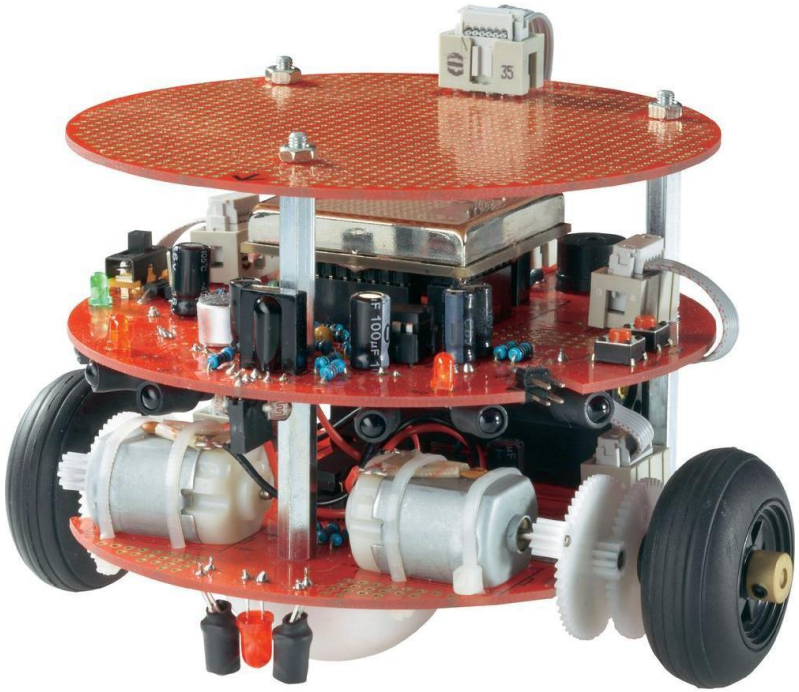
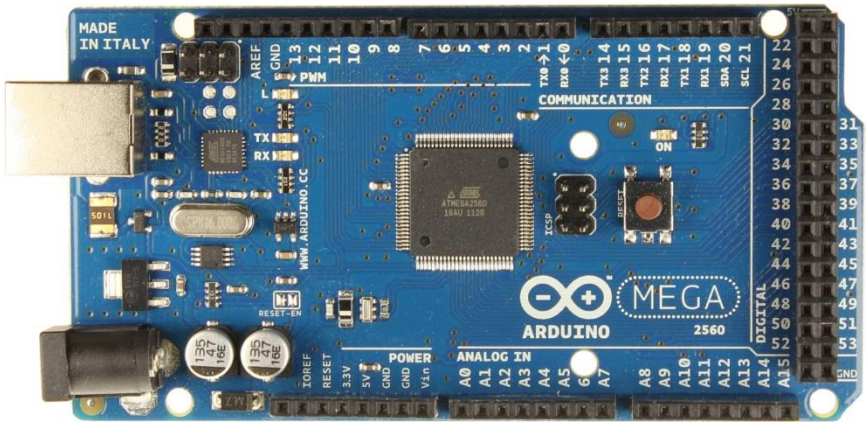




Robot Pro_Bot 128 & Arduino

Pascal MASSON

(pascal.masson@unice.fr)

Edition 2015-2016-V01

POLYTECH NICE-SOPHIA

École Polytechnique Universitaire de Nice Sophia-Antipolis
Cycle Initial Polytechnique
1645 route des Lucioles, 06410 BIOT

Université Nice SOPHIA ANTIPOLIS

Sommaire

1. Introduction

- Le robot Pro-Bot 128 est conçu pour fonctionner avec un microcontrôleur « C-CONTROL Pro 128» basé sur un « AT90CAN » de la société ATMEL
- On se propose ici de remplacer ce microcontrôleur par une carte de développement ARDUINO (Uno ou Méga).
- Cela permettra d'ajouter des fonctions qui ne sont pas déjà intégrées au robot
- Dans ce document nous détaillons le remplacement du microcontrôleur, les différentes fonctions intégrées au robot ainsi que son utilisation.

2.1. Présentation de la carte C-CONTROL Pro 128

Pinout du microcontrôleur AT90CAN

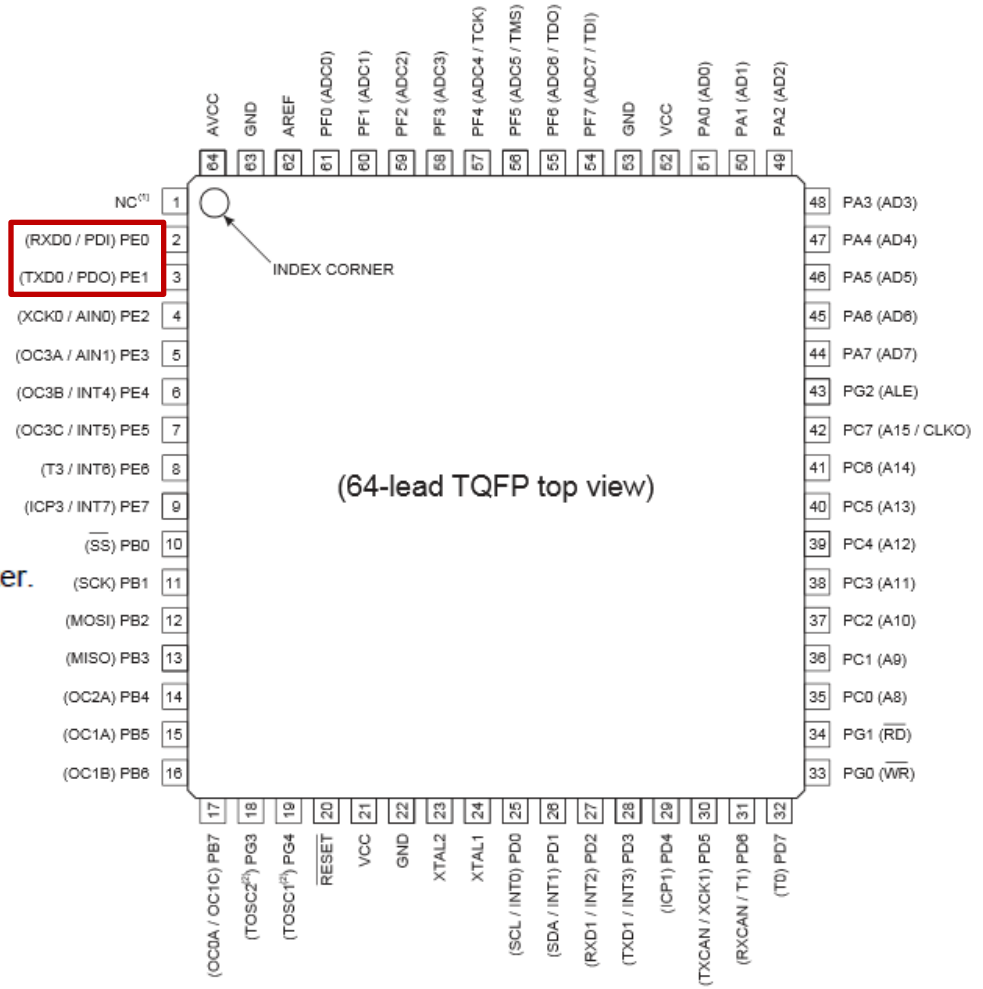
- Les I/O sont :

Port A (PA7..PA0)
Port B (PB7..PB0)
Port C (PC7..PC0)
Port D (PD7..PD0)
Port E (PE7..PE0)

8-bit bi-directional I/O port

Port G (PG4..PG0) 5-bit I/O port

Port F (PF7..PF0) analog inputs to the A/D Converter.



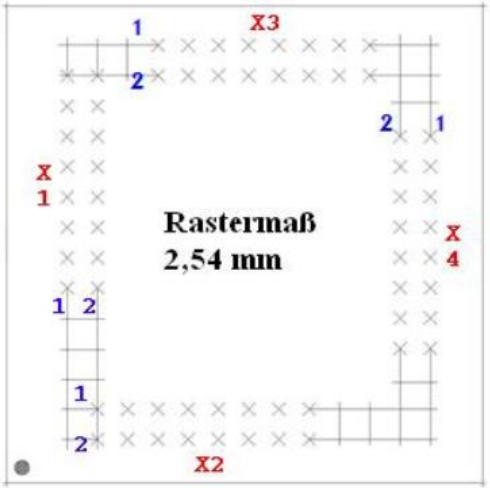
- PE0 et PE1 servent à communiquer avec l'ordinateur

2.1. Présentation de la carte C-CONTROL Pro 128

- **Pinout du boitier C-CONTROL vs microcontrôleur AT90CAN**

Module	M128	Port
X1_16	2	PE0
X1_15	3	PE1
X1_14	4	PE2
X1_13	5	PE3
X1_12	6	PE4
X1_11	7	PE5
X1_10	8	PE6
X1_9	9	PE7
X1_8	10	PB0
X1_7	11	PB1
X1_6	12	PB2
X1_5	13	PB3
X1_4	14	PB4
X1_3	15	PB5
X1_2	16	PB6
X1_1	17	PB7

Module	M128	Port
X2_5	18	PG3
X2_6	19	PG4
X2_3	20	
X2_9	25	PD0
X2_10	26	PD1
X2_11	27	PD2
X2_12	28	PD3
X2_13	29	PD4
X2_14	30	PD5
X2_15	31	PD6
X2_16	32	PD7
X2_7	33	PG0
X2_8	34	PG1
X2_4	43	PG2



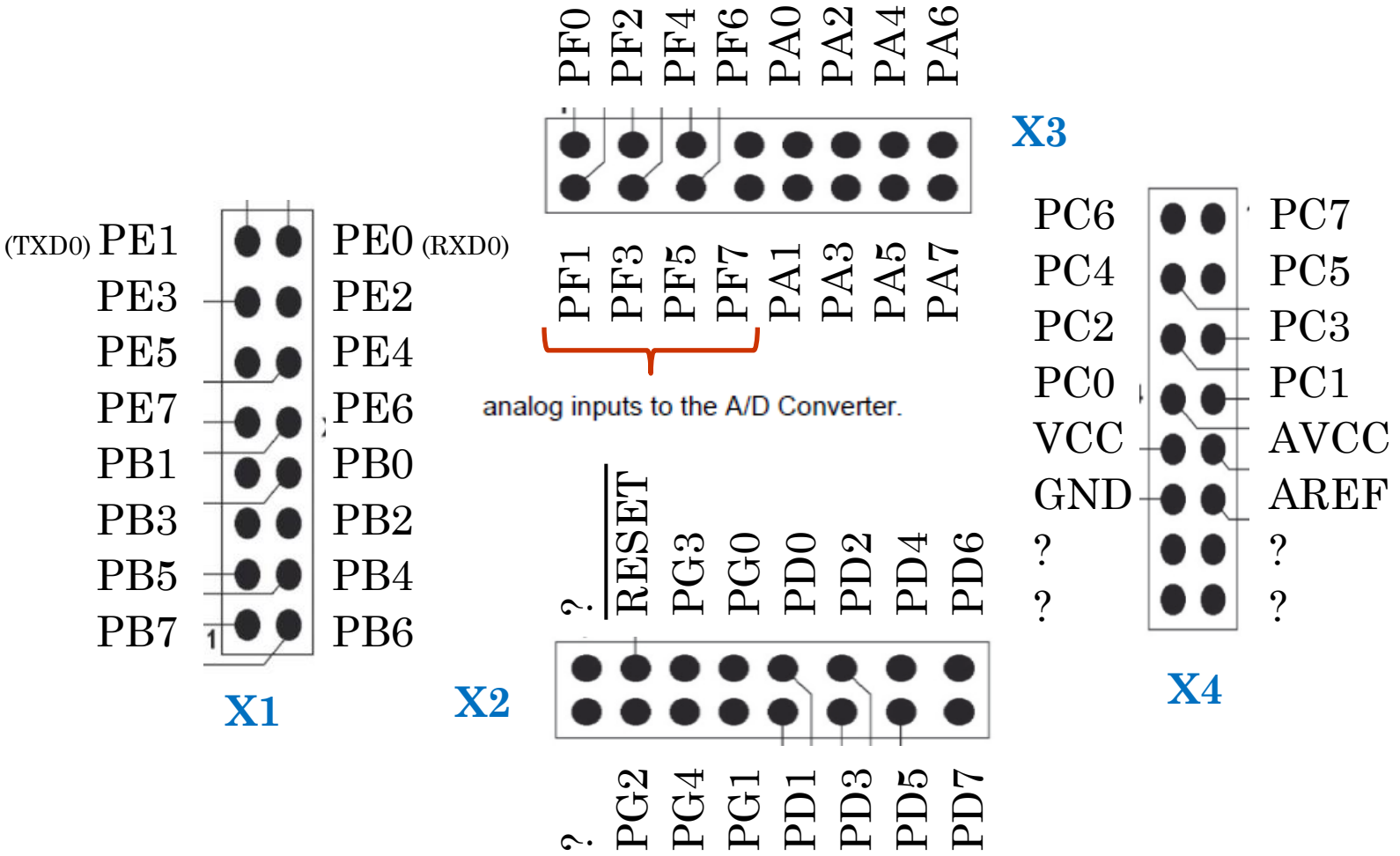
Module	M128	Port
	1	
	23	
	24	

Module	M128	Port
X3_16	44	PA7
X3_15	45	PA6
X3_14	46	PA5
X3_13	47	PA4
X3_12	48	PA3
X3_11	49	PA2
X3_10	50	PA1
X3_9	51	PA0
X3_8	54	PF7
X3_7	55	PF6
X3_6	56	PF5
X3_5	57	PF4
X3_4	58	PF3
X3_3	59	PF2
X3_2	60	PF1
X3_1	61	PF0

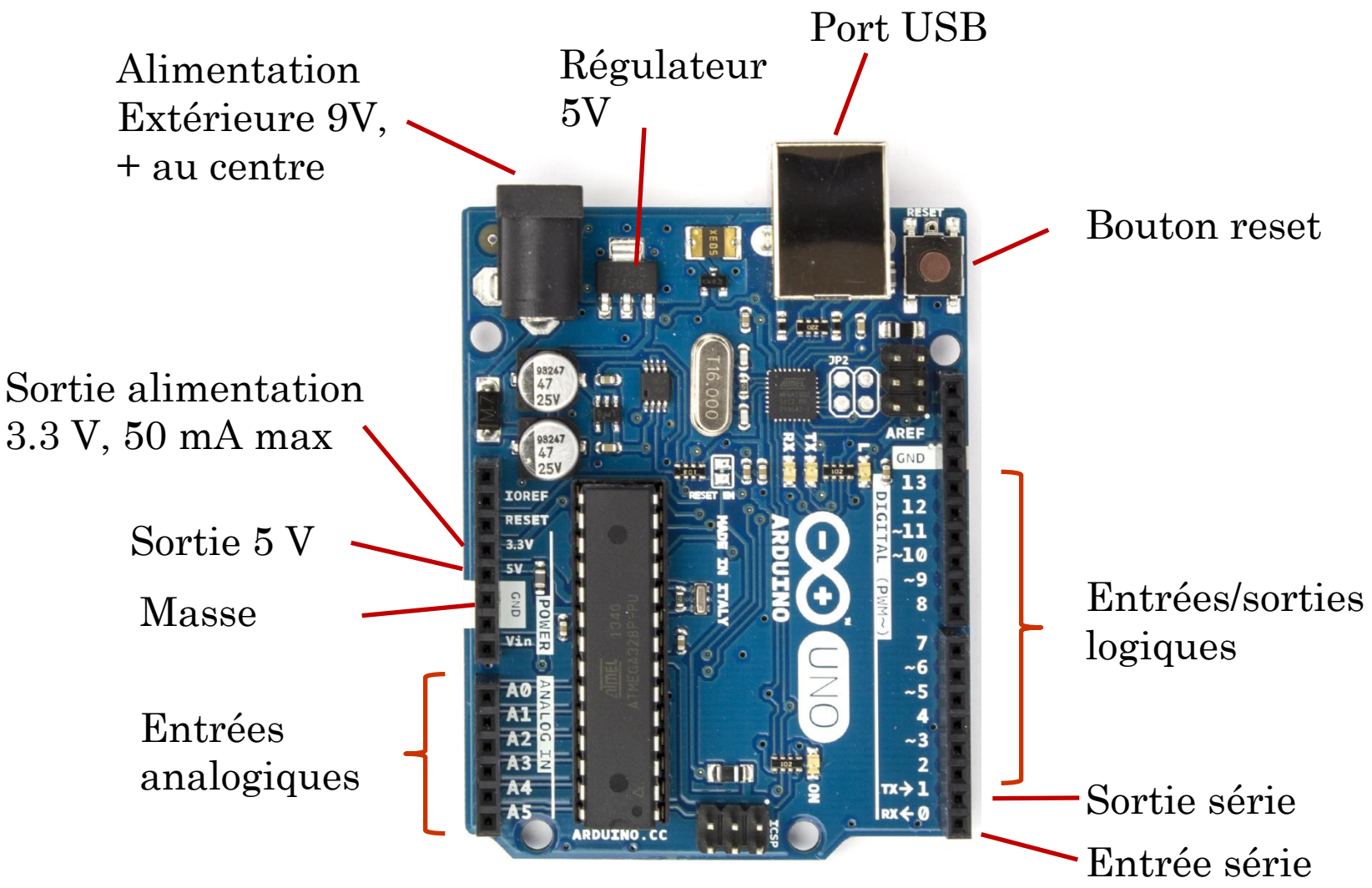
Module	M128	Port
X4_10	21	
X4_12	22	
X4_8	35	PC0
X4_7	36	PC1
X4_6	37	PC2
X4_5	38	PC3
X4_4	39	PC4
X4_3	40	PC5
X4_2	41	PC6
X4_1	42	PC7
X4_10	52	
X4_12 X4_11	53 62	
X4_12	63	
X4_9	64	

2.1. Présentation de la carte C-CONTROL Pro 128

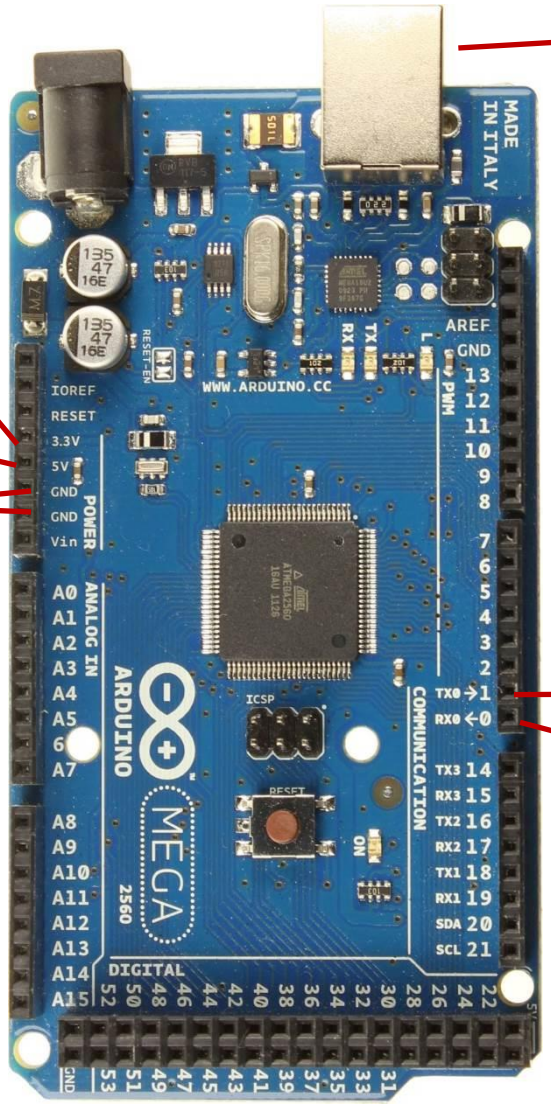
- **Pinout du boitier C-CONTROL vs microcontrôleur AT90CAN**



2.2. Présentation de la carte ARDUINO UNO

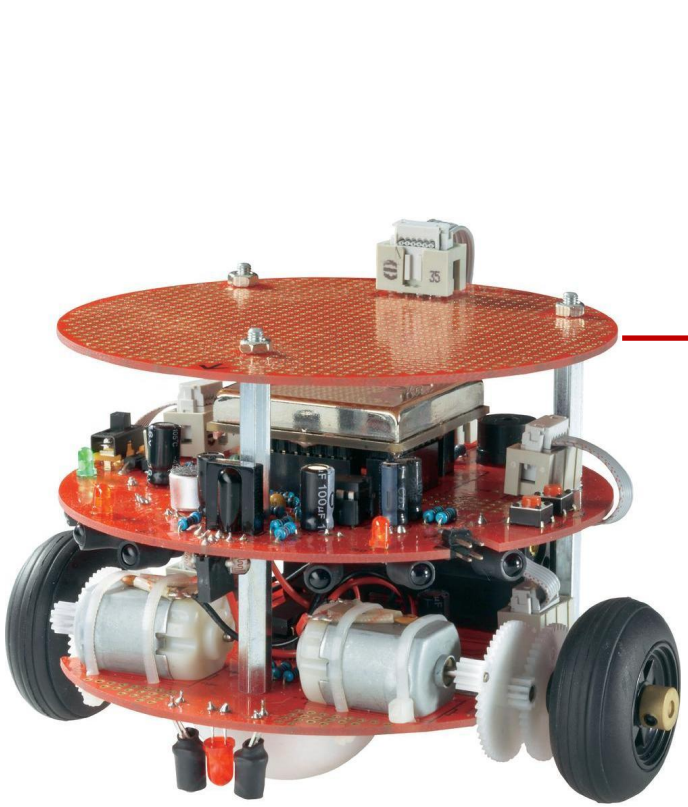


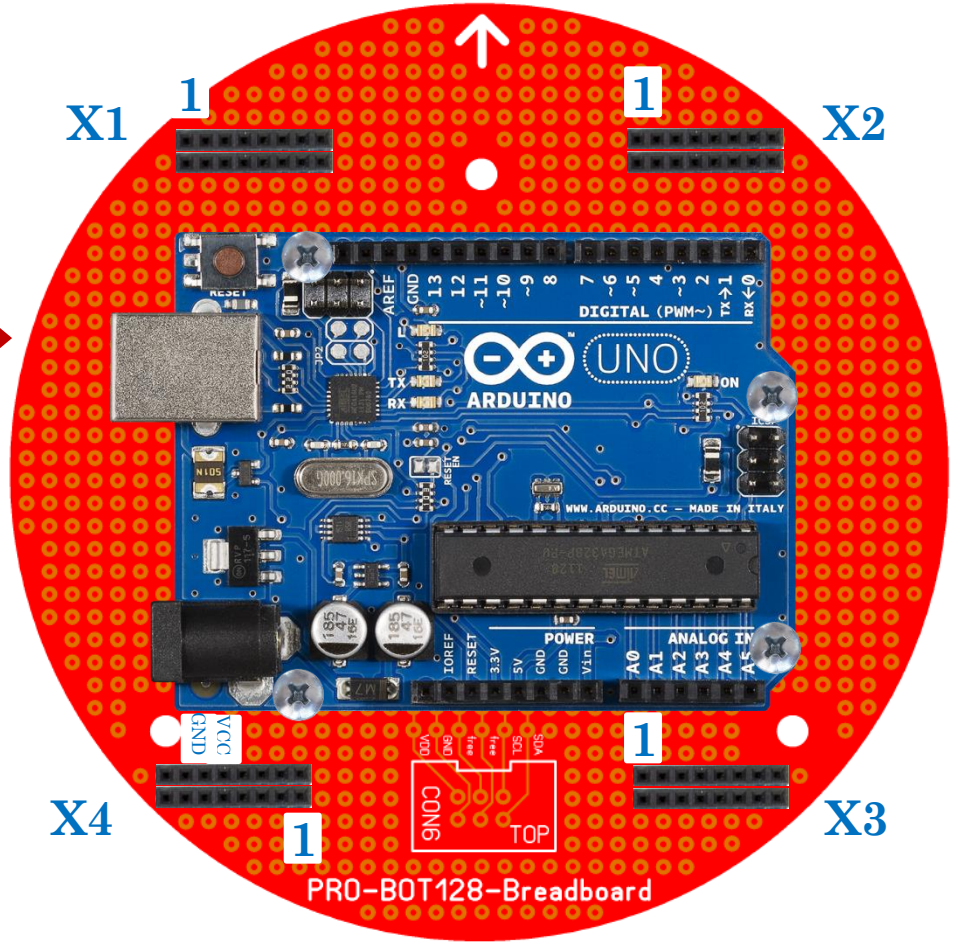
2.3. Présentation de la carte ARDUINO MEGA



2.3. Présentation de la carte ARDUINO MEGA

- On ramène les connecteurs sur la plaque supérieure et on y fixe l'arduino



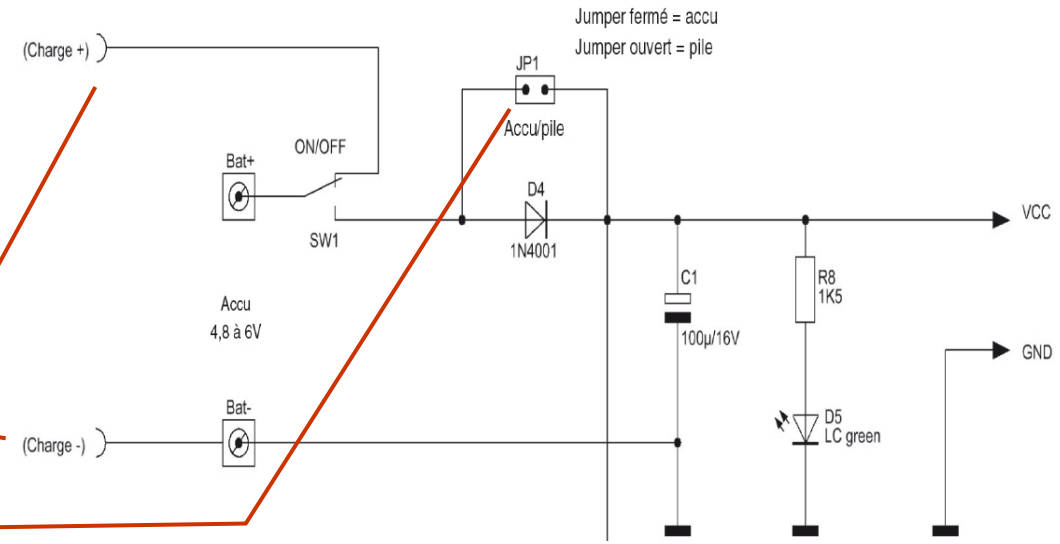


3.1. Alimentation

- Le robot est conçu pour fonctionner avec des accumulateurs (accu) car ils permettent de délivrer plus de courant. Dans ce cas le « jumper » (JP1) doit être en position fermé ce qui court-circuite la diode D4.

- Ces accumulateurs peuvent être rechargés sans être enlevés en basculant le commutateur (SW1) sur OFF (à vérifier)

- Si on utilise des piles alors le « jumper » (JP1) doit être enlevé (ouvert) pour qu'aucun courant ne puisse entrer dans les piles

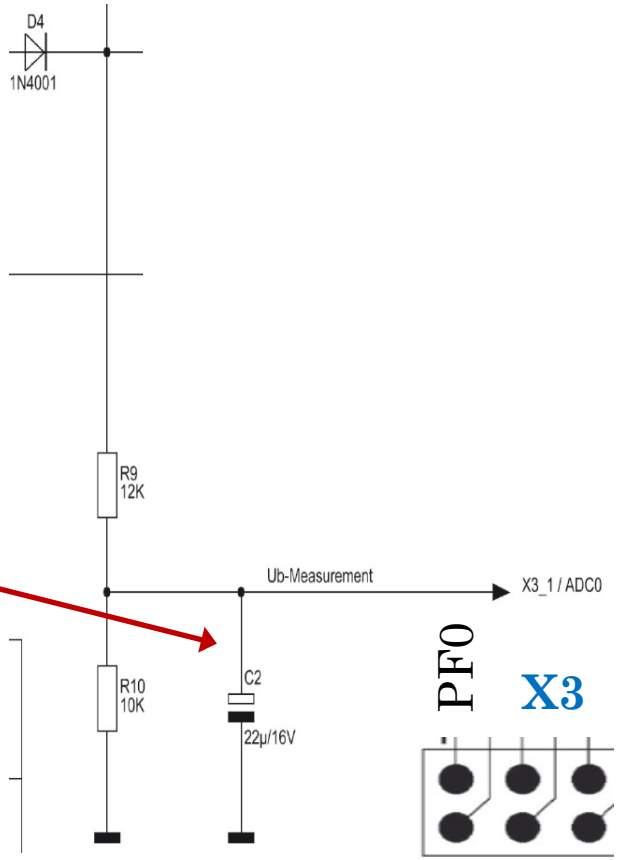


3.1. Alimentation

- Il est possible de mesurer la tension via un pont diviseur (R9 et R10) et une des entrées analogiques de l'Arduino.
- Le pont diviseur est relié à la borne X3_1 c'est-à-dire l'entrée PF0 de l'AT90CAN
- La tension sur la borne X3_1 a pour valeur :

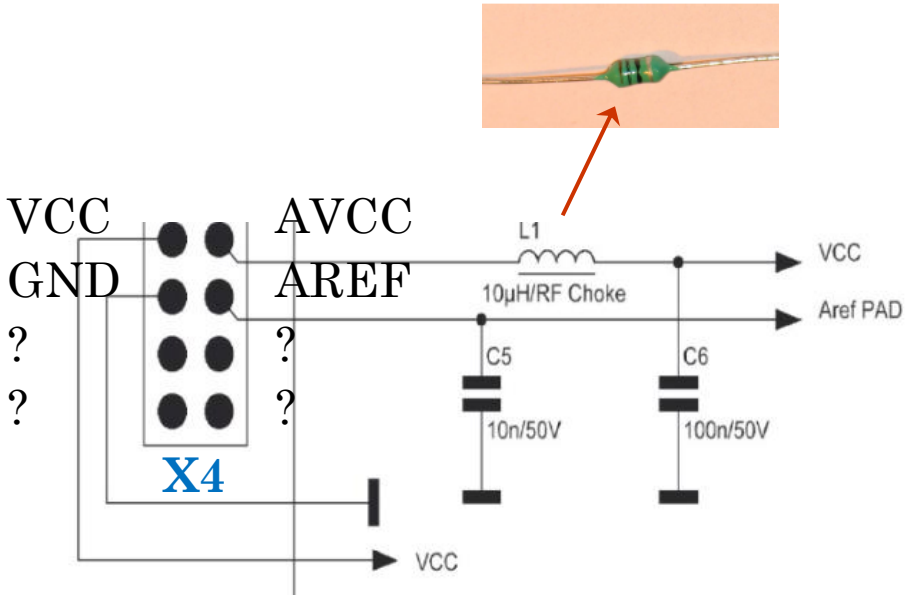
$$V_{X3\_1} = \frac{R_{10}}{R_9 + R_{10}} V_{CC} = 0.4545 V_{CC}$$

- C2 sert à filtrer cette tension



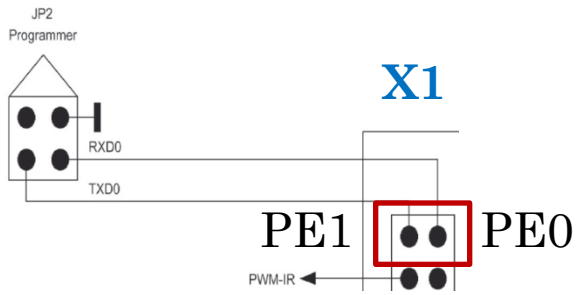
3.1. Alimentation

- Dans la notice de l'AT90CAN, il est spécifié : « AVCC is the supply voltage pin for the A/D Converter on Port F. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter".
- Cela explique la présence de l'inductance L1 (appelée « Choke ») et des capacités C5 et C6
- Le but de ce filtre est d'avoir une tension la plus stable possible pour alimenter les blocs qui font la conversion analogique/numérique



3.2. Connexion vers l'ordinateur

- Ces 2 connecteurs ne seront pas utilisés avec la carte Arduino car ils correspondent à la prise USB

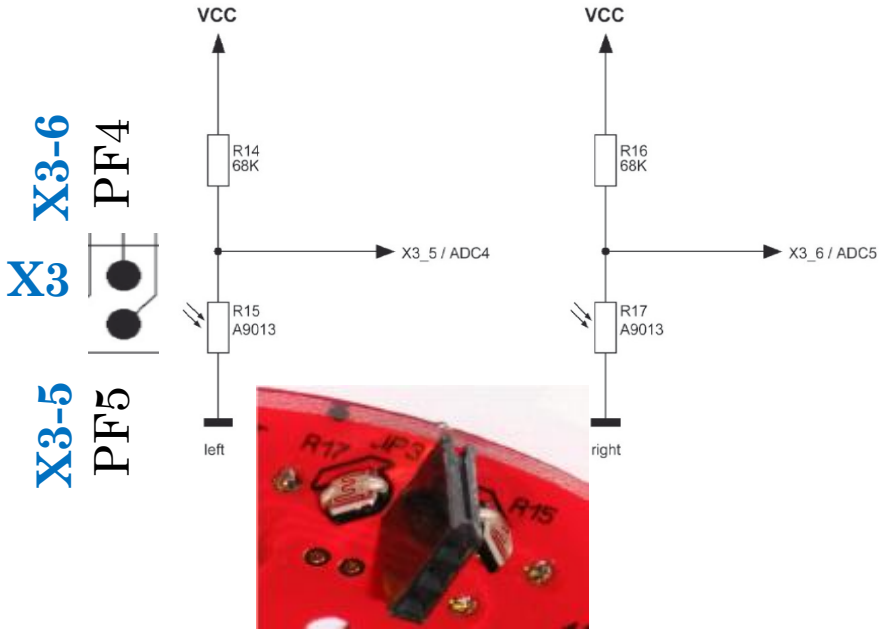


3.3. Résistances dépendantes de la lumière (LDR)

Présentation

- Les résistances dont la valeur change en fonction de l'intensité lumineuse sont ce qu'on appelle des LDR (Light Dependent Resistor)

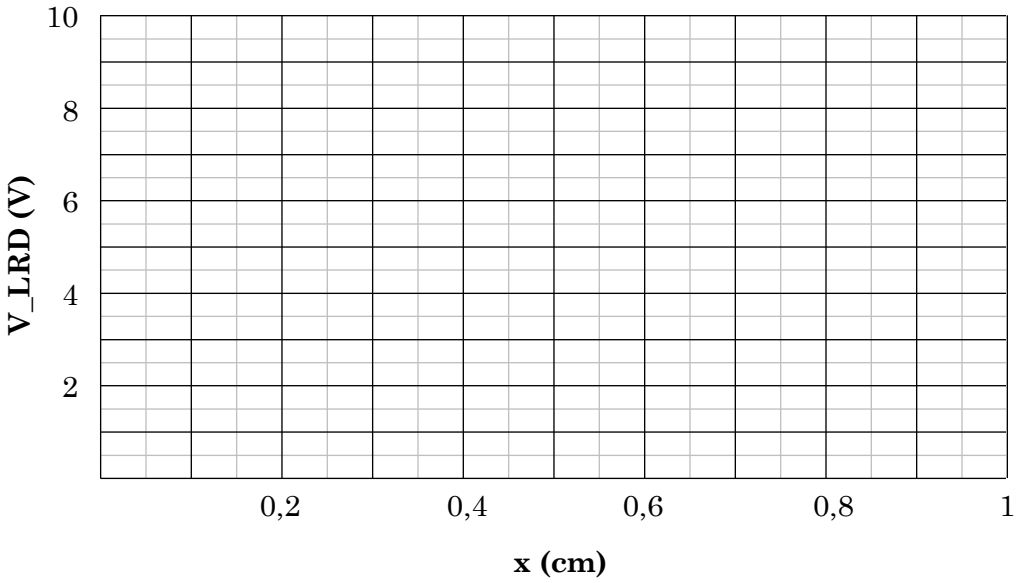
- A connecter sur des entrées analogiques



3.3. Résistances dépendantes de la lumière (LDR)

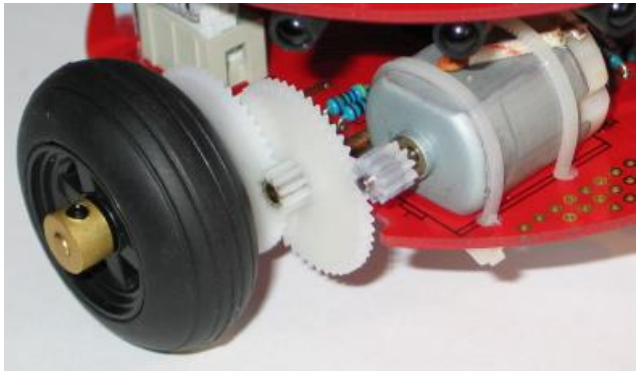
Valeur de la tension

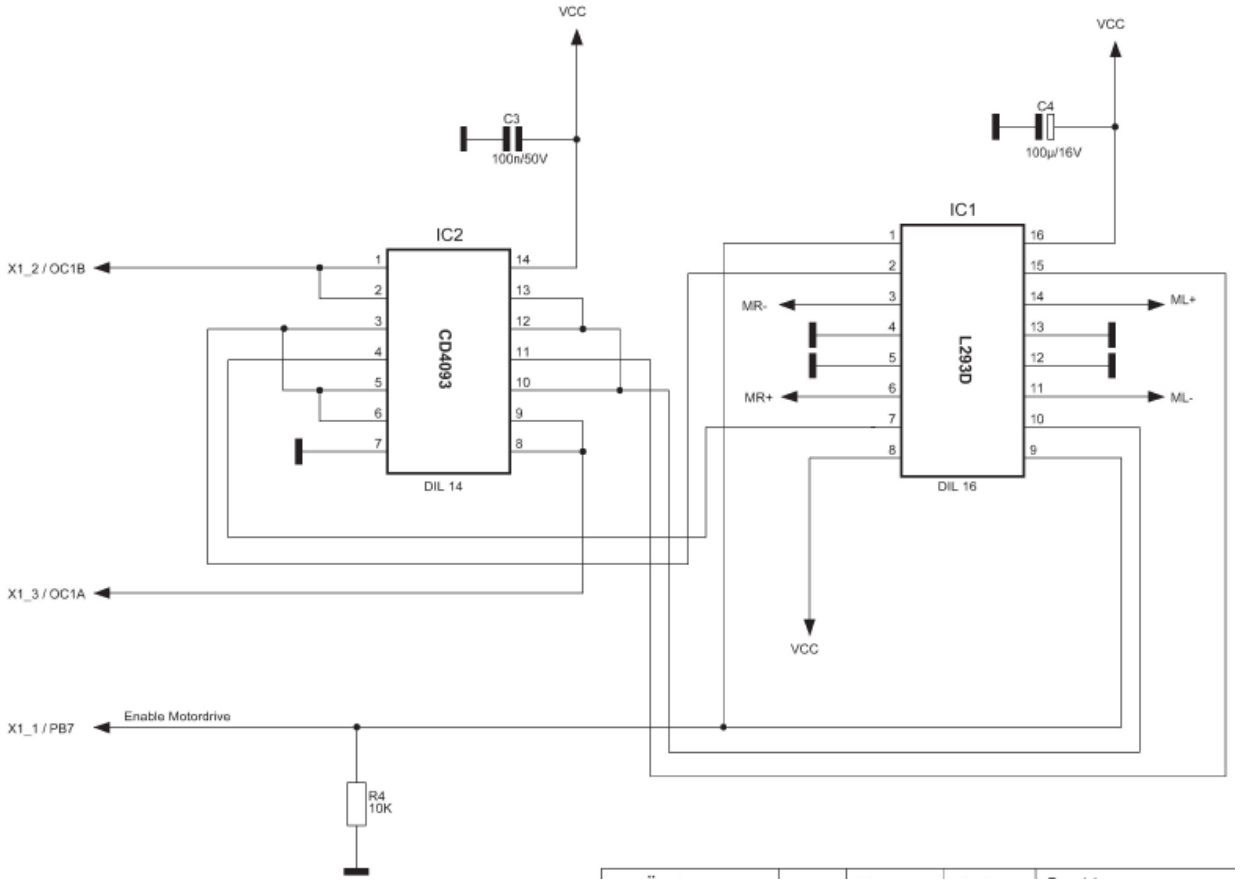
- On relève la tension entre la résistance et le LDR en fonction de la distance de la torche que nous utiliserons pour attirer le robot



3.4. Les moteurs

- Le control des moteurs s'effectue avec 2 CI : le CD4093 (Quad 2-Input NAND Schmitt Trigger) et le L293D (PUSH-PULL FOUR CHANNEL DRIVER WITH DIODES)

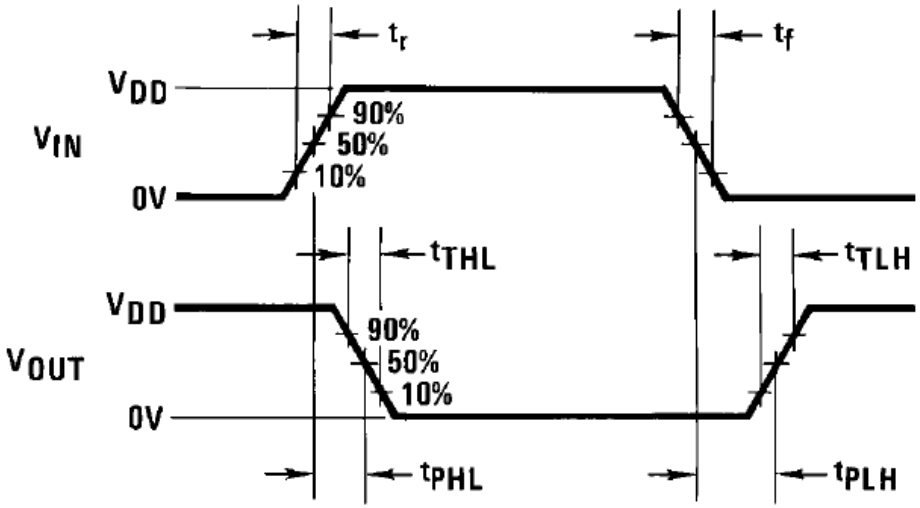


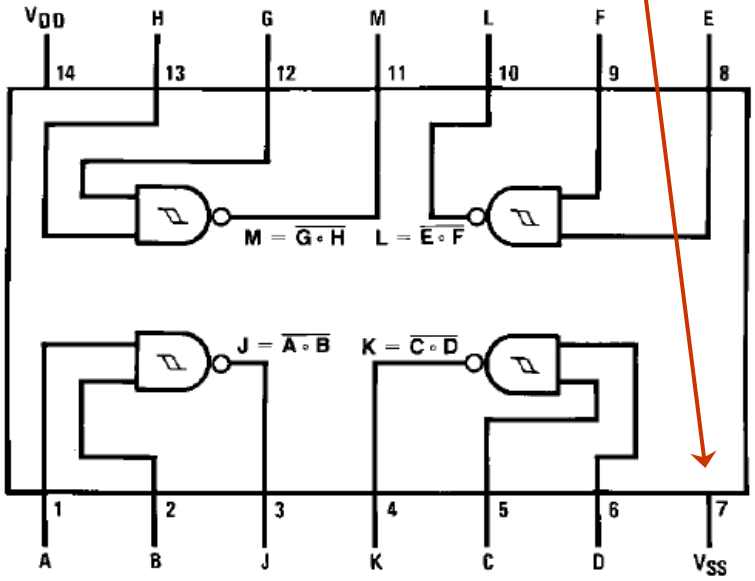


3.4. Les moteurs

Description du CD4093

- Le CI se compose de 4 NAND. V_{DD} correspond à l'alimentation et V_{SS} à la masse.

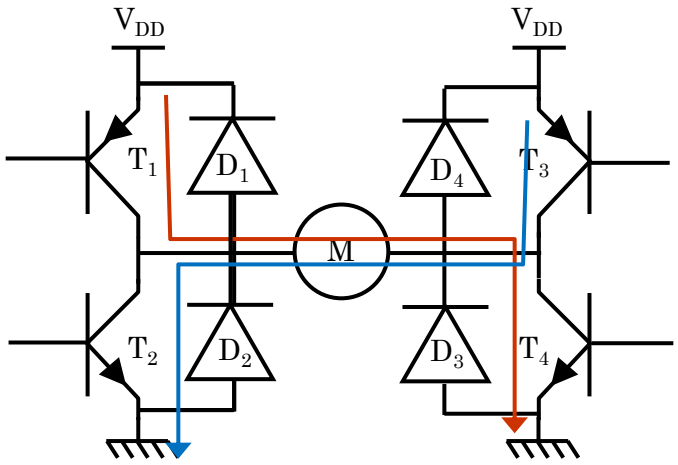




3.4. Les moteurs

Description du L293D

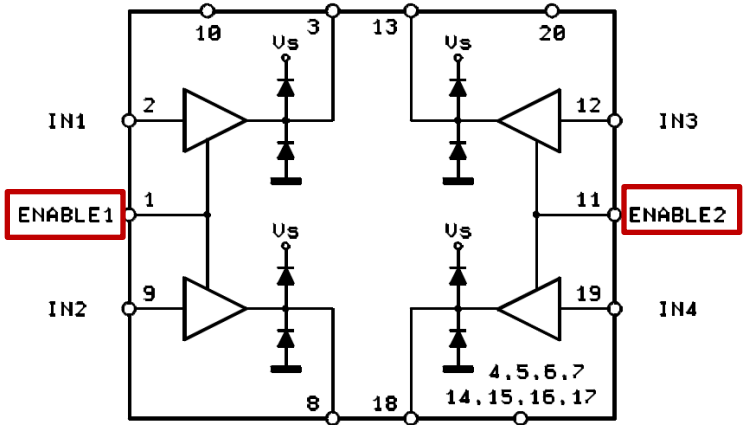
- L'utilisation d'un pont en H permet de faire tourner le moteur dans les 2 sens. Le CI est composé de 2 ponts.
- Les diodes (de roue libre) évacuent l'énergie de la bobine du moteur quand les transistors sont bloqués.

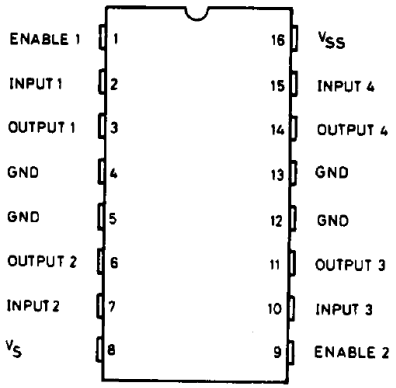


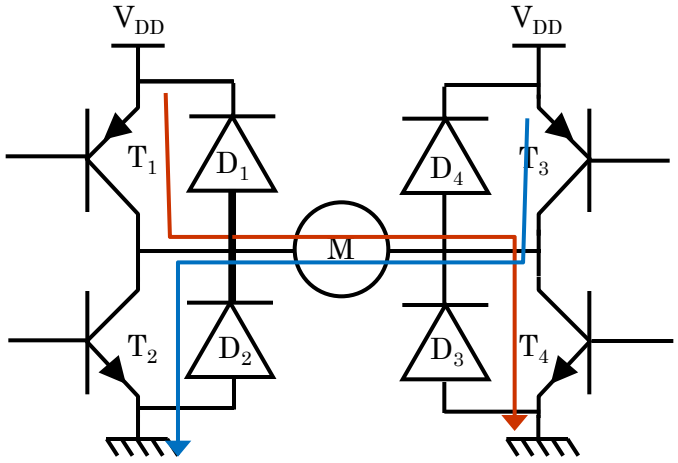
3.4. Les moteurs

Description du L293D

- Des entrées «enable» permettent débloquer les moteurs
- Il y a une alimentation V_{SS} pour la partie logique et les transistors et une alimentation V_S pour le moteur. Cela permet d'avoir un V_S bien plus grand que V_{SS}. Dans le cas de ce robot on a $V_S = V_{SS}$.





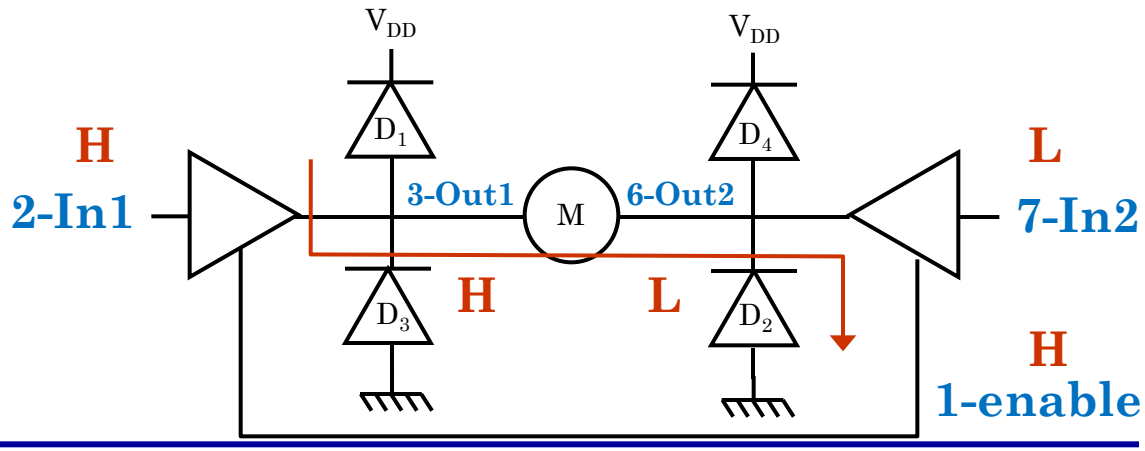


3.4. Les moteurs

Control du moteur droit

- Si on met In1 à l'état H et In2 à l'état L le moteur droit tourne en marche avant

Input	Enable (*)	Output
H	H	H
L	H	L
H	L	Z
L	L	Z

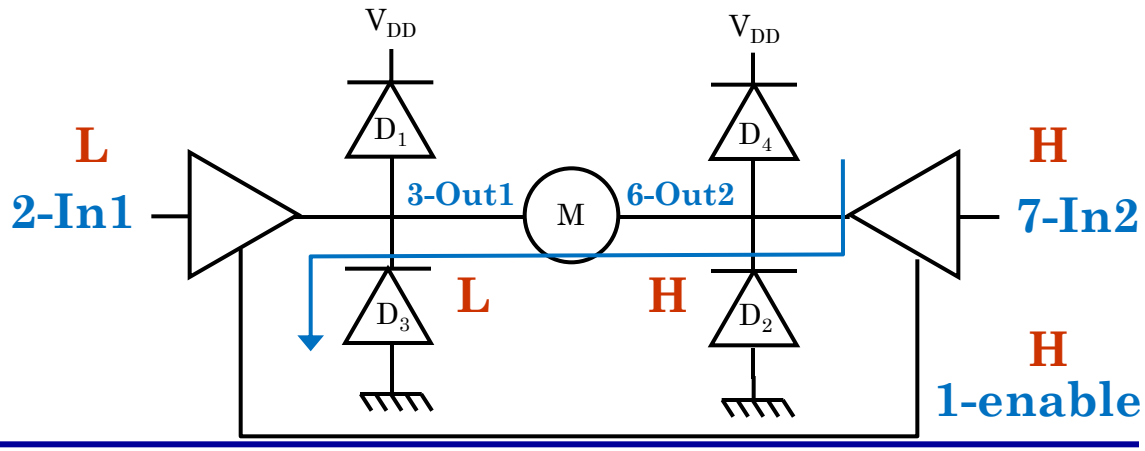


3.4. Les moteurs

Control du moteur droit

- Si on met In1 à l'état H et In2 à l'état L le moteur droit tourne en marche avant
- Si on met In1 à l'état L et In2 à l'état H le moteur droit tourne en marche arrière

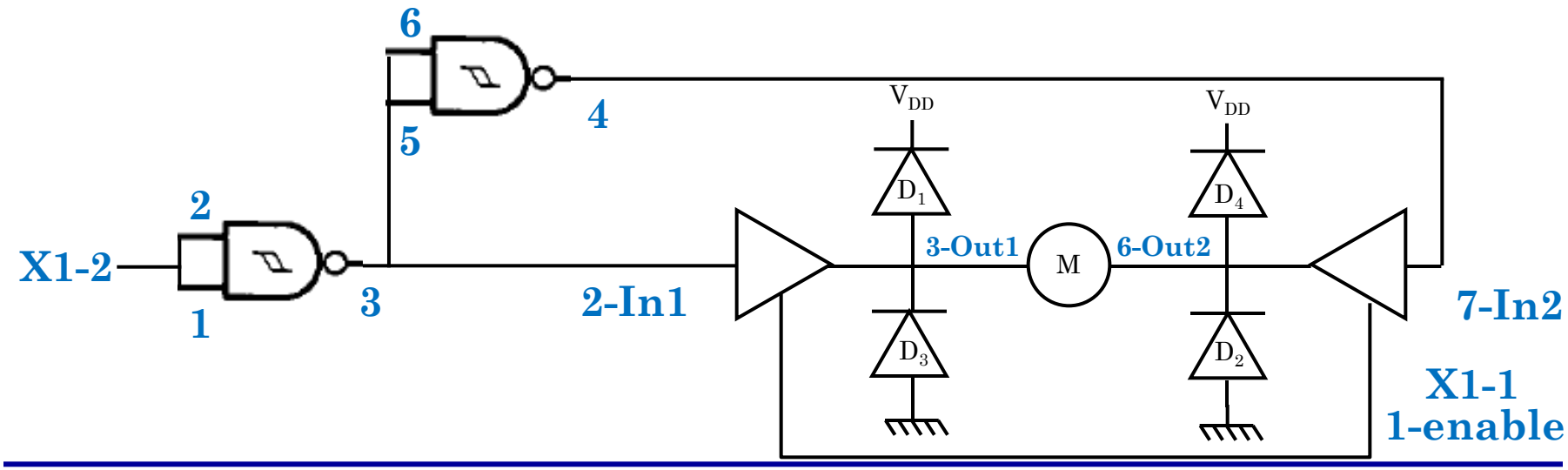
Input	Enable (*)	Output
H	H	H
L	H	L
H	L	Z
L	L	Z



3.4. Les moteurs

Control du moteur droit

- Si on met In1 à l'état H et In2 à l'état L le moteur droit tourne en marche avant
- Si on met In1 à l'état L et In2 à l'état H le moteur droit tourne en marche arrière
- On constate que In2 est le complément de In1 et ce sont les trigger NAND qui s'en chargent. Ils sont utilisés comme des inverseurs.

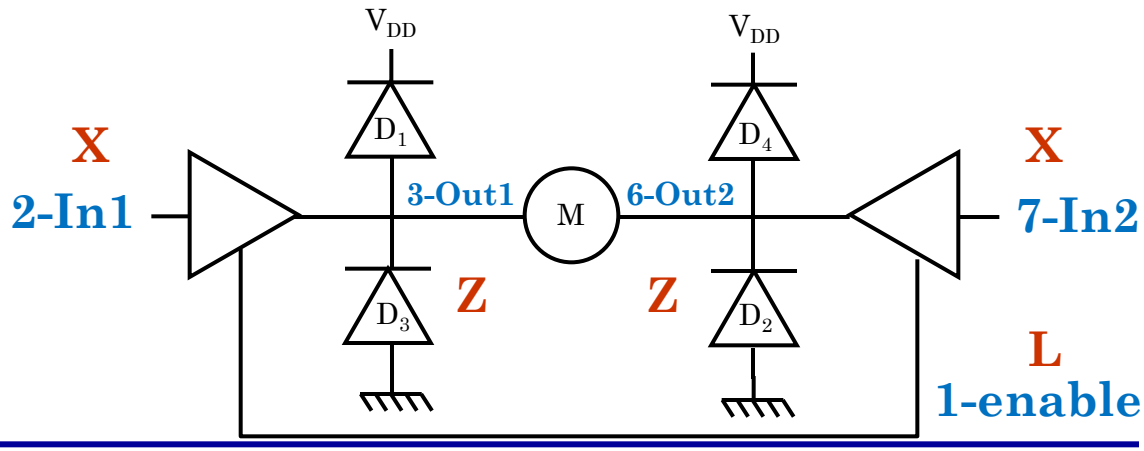


3.4. Les moteurs

Control du moteur droit

- Si enable est à l'état L alors aucun courant ne peut passer par le moteur
- Il est important de noter que la liaison «enable» est commune aux 2 moteurs et qu'il n'est pas possible de bloquer un moteur pendant qu'on fait tourner l'autre.

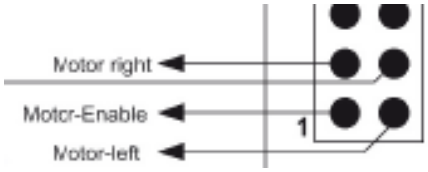
Input	Enable (*)	Output
H	H	H
L	H	L
H	L	Z
L	L	Z



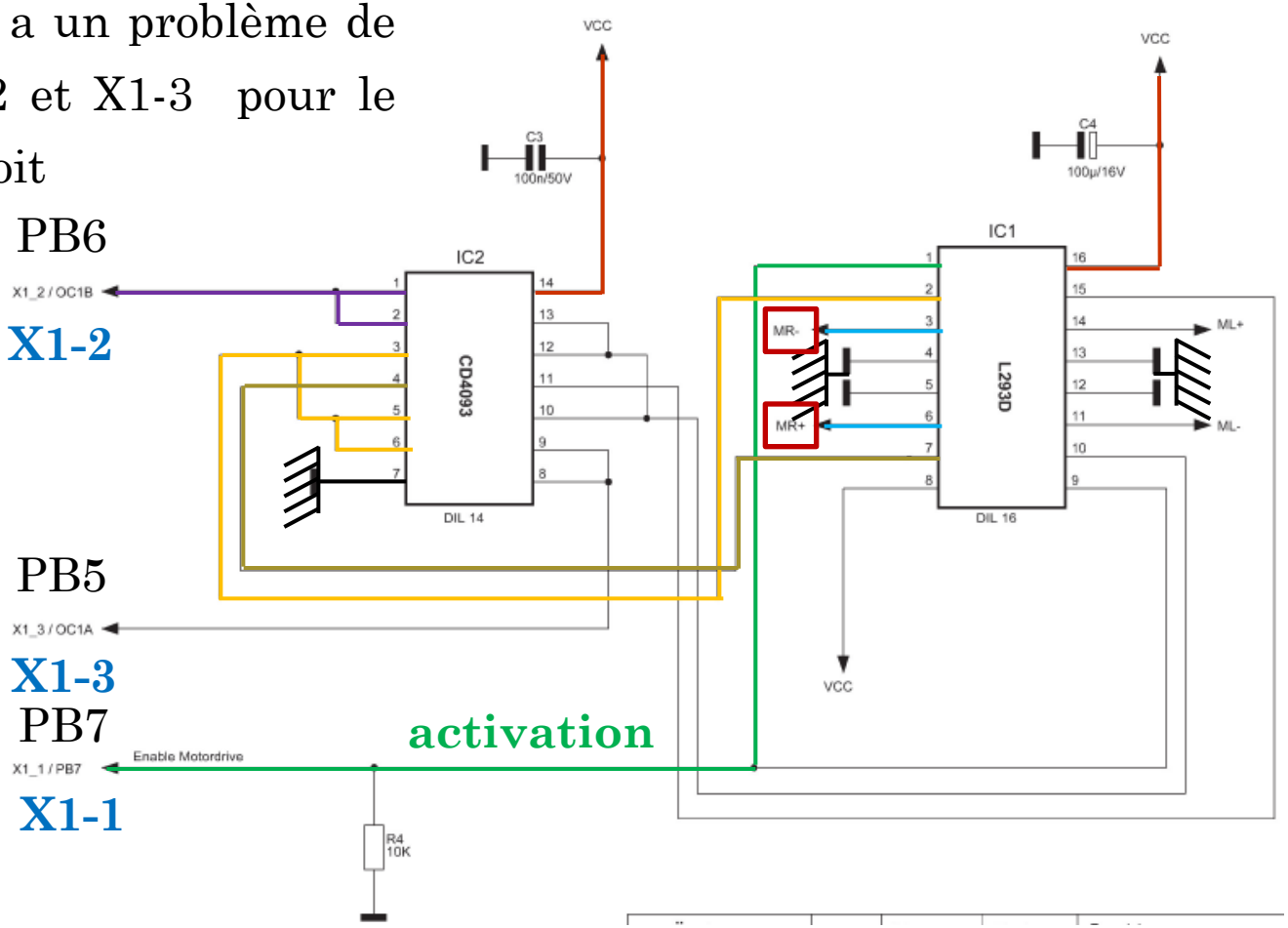
3.4. Les moteurs

Control du moteur droit

- On constate qu'il y a un problème de définition entre X1-2 et X1-3 pour le control du moteur droit



X1



3.4. Les moteurs

Commande des moteurs

- Les états haut et bas font tourner les moteurs dans des sens opposés et il n'y a qu'une liaison enable.
- Pour palier à ce problème d'enable, il faut utiliser la fonction PWM. Pour empêcher un moteur de tourner et il faut appliquer un palier haut identique au palier bas. Dans ce cas, on demande au moteur d'aller en avant et il n'a pas le temps de commencer qu'on lui demande d'aller en arrière etc …
- **analogWrite(X, Y)** : c'est la fonction dédiée au PWM de l'arduino. Il faut spécifier l'I/O X ainsi que la valeur du rapport cyclique Y. Y doit avoir une valeur comprise entre 0 et 255. Y = 0 : la sortie est toujours à 0. Y = 255 : la sortie est toujours à 5 V.
- Les I/O qui permettent d'utiliser les PWM sont précédées du signe « ~ », il s'agit des I/O 3, 5, 6, 9, 10 et 11 pour l'Arduino UNO. La fréquence d'horloge du PWM est de 490 Hz (à vérifier)

3.4. Les moteurs

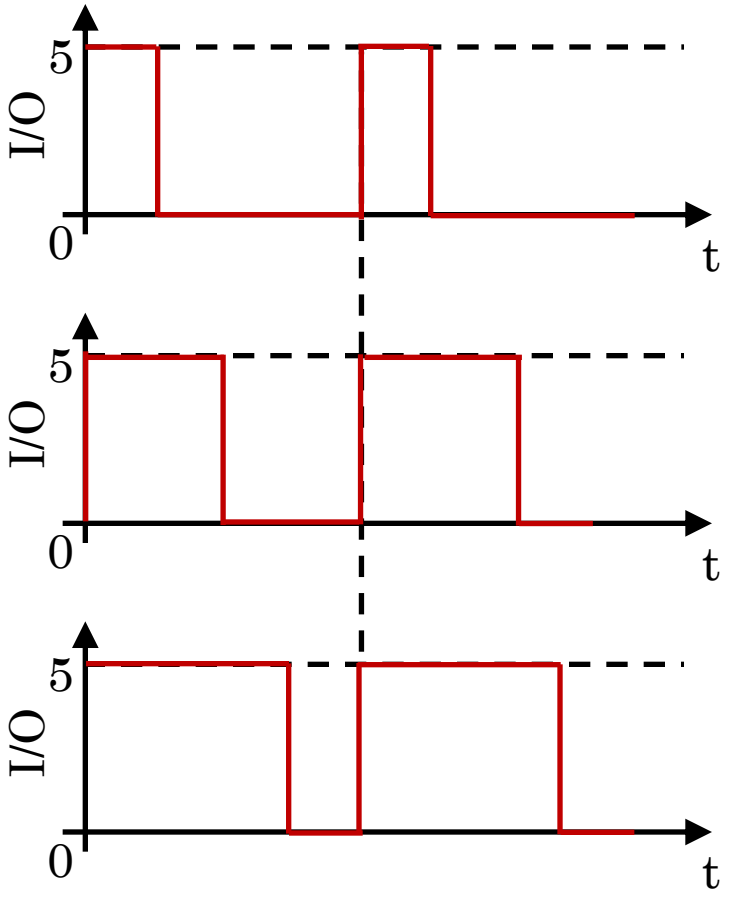
Commande des moteurs

- Le sens de rotation dépend de la valeur donnée à Y.

Y ∈ [0,126]
Moteur tourne en arrière

Y = 127, 128
Moteur à l'arrêt

Y ∈ [129, 255]
Moteur tourne en avant



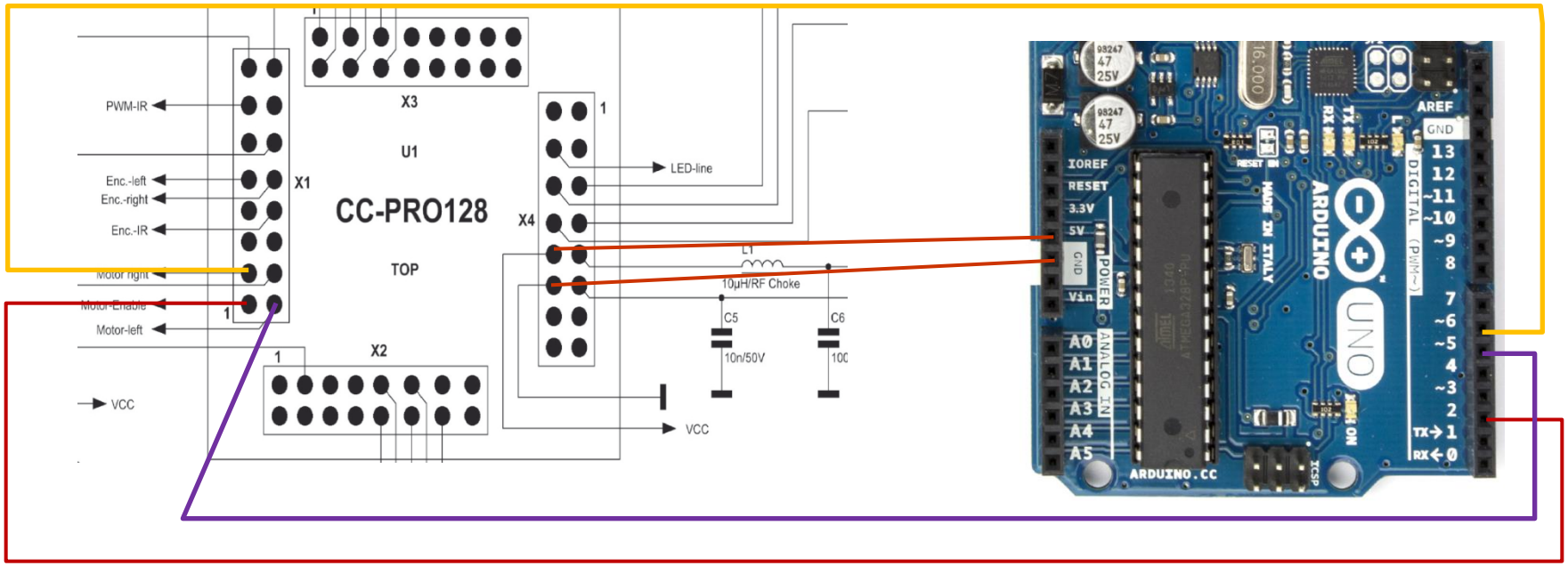
4. Le robot qui tourne en rond

4.1. Objectif

- L'objectif (très simple) est d'apprendre à faire fonctionner les moteurs l'un après l'autre ou en même temps mais en sens contraire ce qui revient à faire tourner le robot en rond.

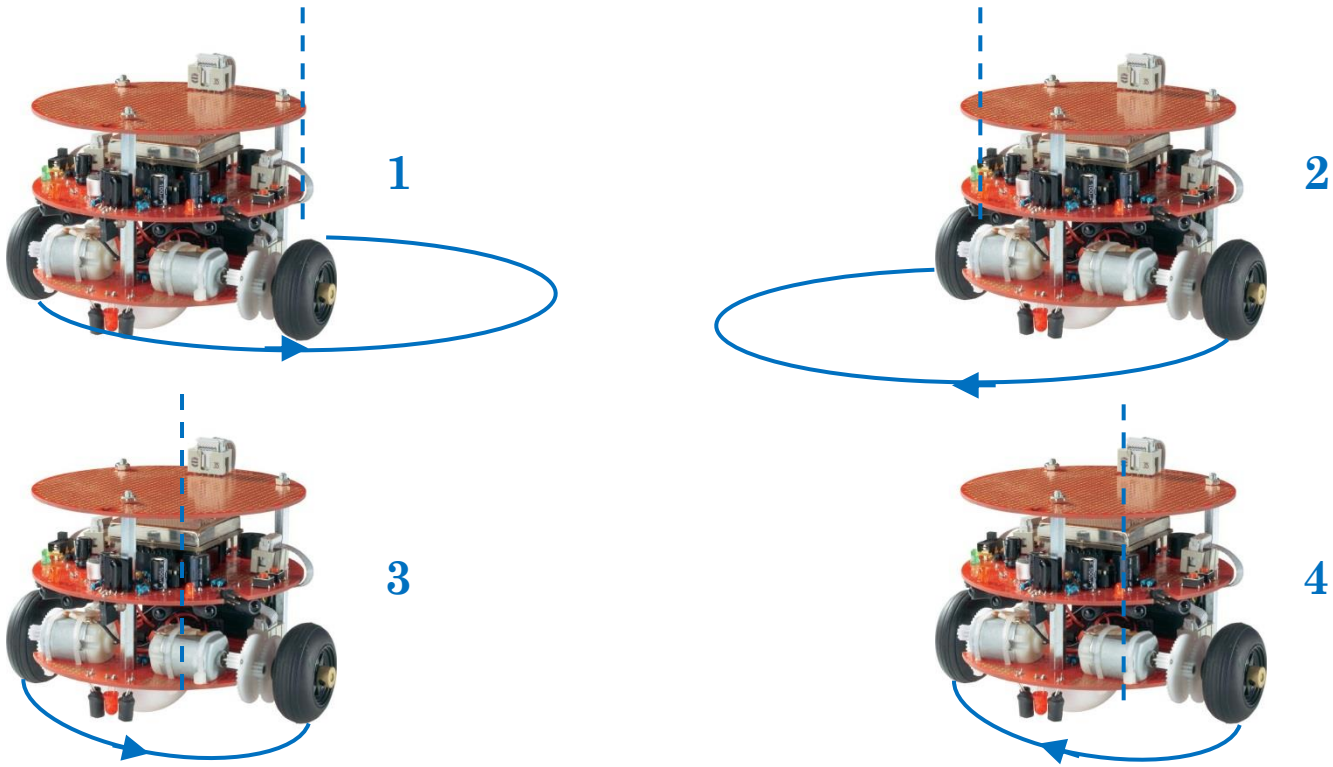
4.2. Montage

- Hors V_{DD} et G_{ND}, le câblage proposé n'est pas unique



4.3. Programme : mouvements du robots

- Le robot va faire 4 mouvements différents tout en restant basé sur des cercles.
- Chaque mouvement dure 5 s et le dernier mouvement est plus lent que les autres



4.3. Programme : code

```
//Pro-Bot128 qui tourne en rond
const int enable=2;
const int MR=6;
const int ML=5;
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
void setup(){ //début du programme
  pinMode(enable,OUTPUT);
  pinMode(MR,OUTPUT);
  pinMode(ML,OUTPUT);
  digitalWrite(enable,HIGH); // Déblocage des moteurs
 } // fin de définition du début de programme avec les variables
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
void loop() { // écriture de ce que fait le programme
  //On bloque ML et on fait tourner MR en avant
  analogWrite(ML,127);
  analogWrite(MR,255);
  delay(5000);
```

1

4.3. Programme 1 : code

```
2   //On bloque MR et on fait tourner ML en avant
    analogWrite(MR,127);
    analogWrite(ML,255);
    delay(5000);

3   //On fait tourner MR en arrière et ML en avant
    analogWrite(MR,0);
    analogWrite(ML,255);
    delay(5000);

4   //On fait tourner ML en arrière et MR en avant mais moins vite qu'avant
    analogWrite(ML,50);
    analogWrite(MR,200);
    delay(5000);

} // indique la fin du programme qui recommencera au début
```

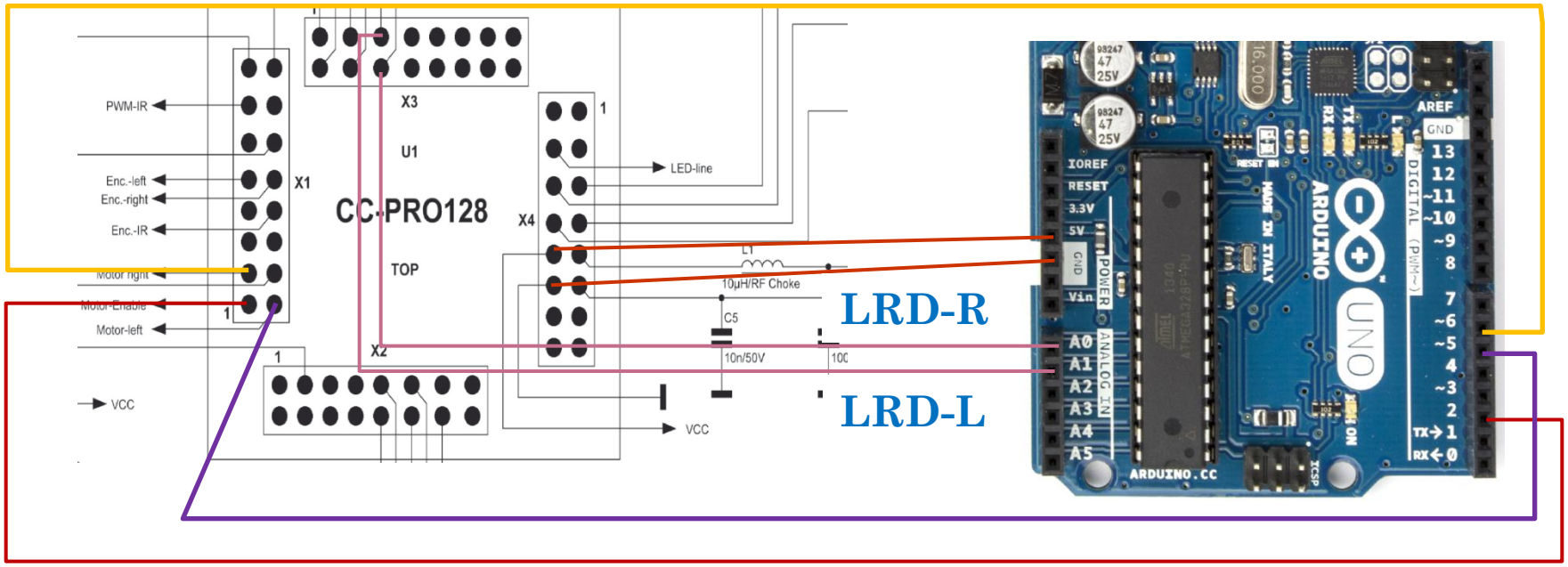
5. Robot attiré par la lumière

5.1. Objectif

- L'objectif est d'attirer le robot avec de la lumière en utilisant les 2 LRD à l'avant du robot. On peut moduler la vitesse du robot en fonction de l'intensité de la lumière reçue.

5.2. Montage

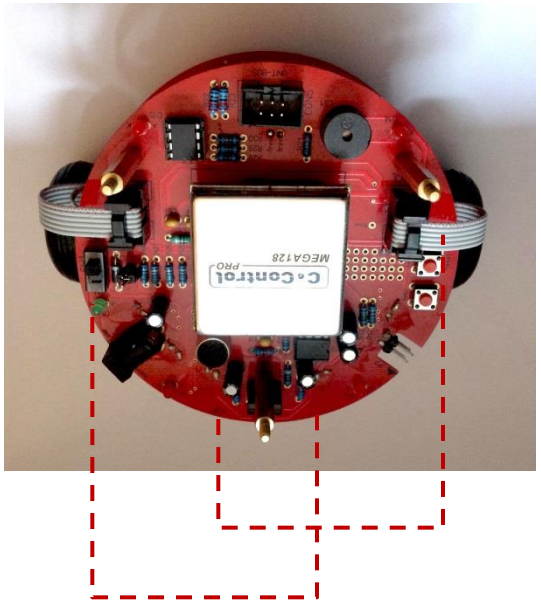
- Hors V_{DD} et G_{ND}, le câblage proposé n'est pas unique



5. Robot attiré par la lumière

5.3. Programme : mouvements du robots

- Les moteurs ne peuvent tourner que vers l'avant
- Le moteur gauche se déclenche avec le capteur droit et inversement



5.3. Programme : mouvements du robots

- **1)** le LRD gauche détecte de la lumière et allume le moteur droit
- **2)** le robot tourne vers sa gauche
- **3)** Les 2 LRD captent de la lumière et allument les 2 moteurs : le robot avance