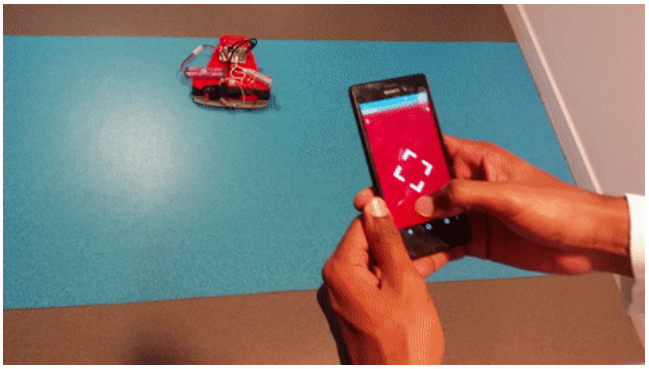
Robot_télécommandé

<- Retour vers la liste des projets

Le robot télécommandé

Voici la documentation de notre projet d'Apprentissage Par Projet d'électronique : création d'un **robot télécommandé**. Le but de ce projet était de concevoir un robot qui serait commandé par WI-FI via une application Androïde que nous devrons développer. Pour cela nous traiterons de plusieurs domaines comme la communication WI-FI sur Arduino, les communications séries, l'asservissement de moteurs ou encore le développement d'une application sur smartphone. Voici un aperçu du rendu final de notre projet :



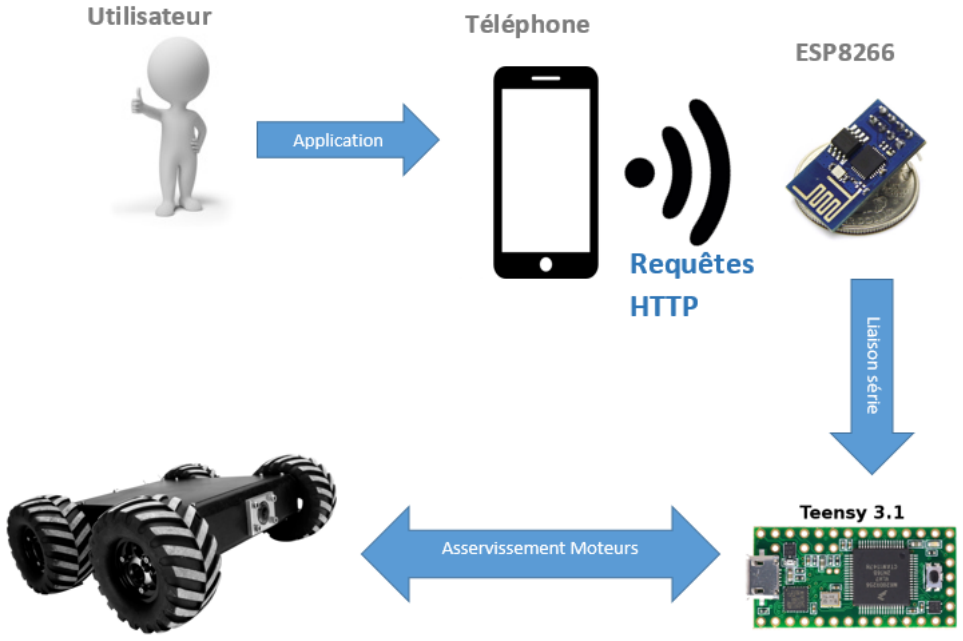
Cette documentation sera découpée en plusieurs parties : premièrement la présentation du projet avec la conception et une vue d'ensemble du projet. Nous verrons ensuite le détail du matériel utilisé; et pour finir les fonctionnement et la collaboration de ces composants.

Fiches de suivi

~~Fiches de suivi~~

Conception du projet

Avant de commencer le projet, nous avons du réfléchir à la manière et aux différents composants que nous utiliserons pour le projet. Dans un premier temps, nous avons défini un schéma bloc résumant le fonctionnement global de notre projet.

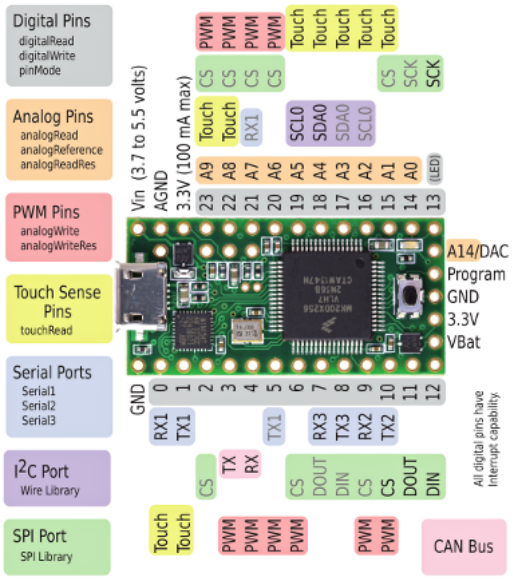


Matériel utilisé

Nous avons été amené à utiliser de nombreux composants électroniques dans ce projet. Nous allons ici présenter les principaux composants de notre projet.

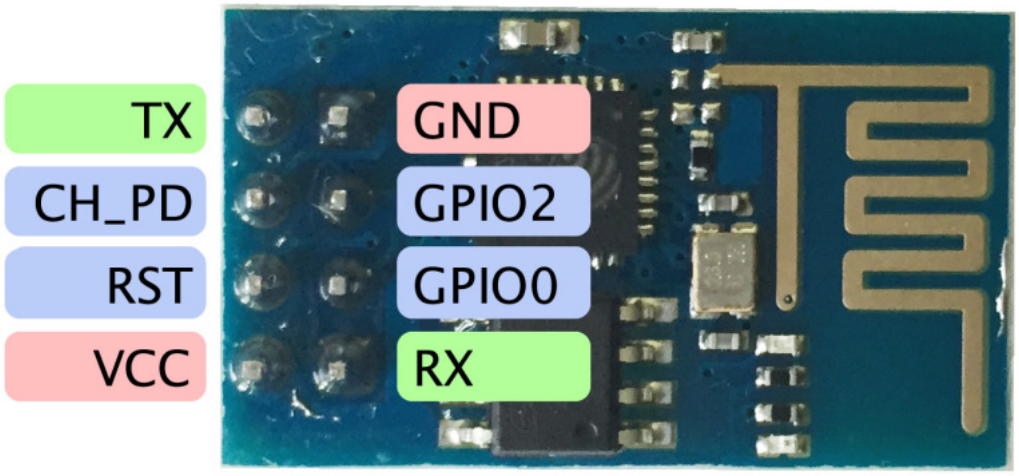
Arduino Teensy 3.1

Premièrement, nous avons utilisé un microcontrôleur embarqué sur le robot. Pour cela nous avons utilisé l'arduino Teensy 3.1. Voici le schéma de cette dernière :



Sur cette carte, nous utilisons tout d'abord les ports de commande pour les moteurs, la led, ou d'autre composants.En suite, nous utilisons les ports 0 et 1 qui servent de communication série : TX et RX. On communique sur ces ports avec le composant effectuant la liaison WI-FI avec le téléphone : **l'ESP8266**

Câblage de l'ESP



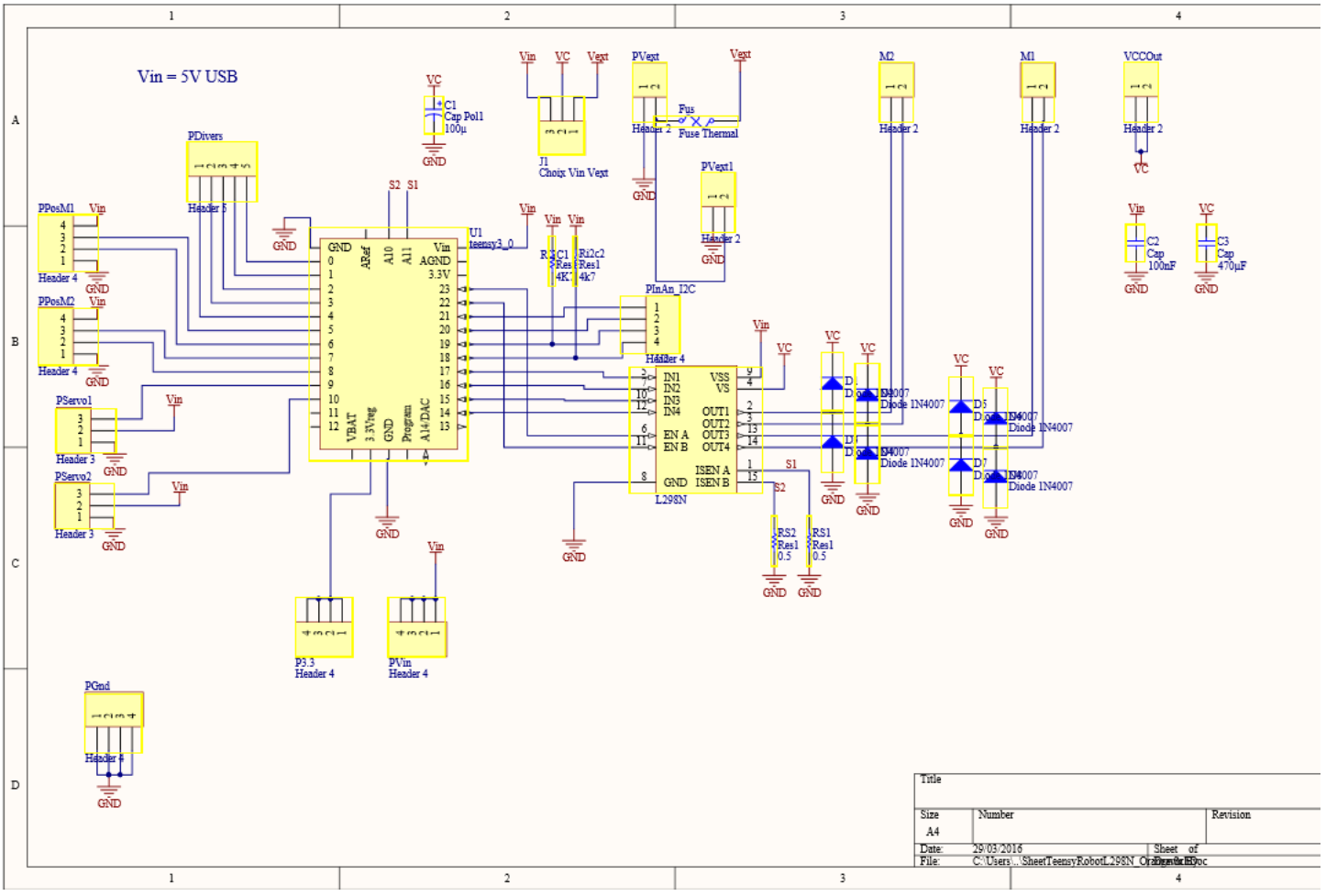
Carte

La carte de type Arduino sera programmée pour recevoir les signaux produits par l'application programmée ci-dessus. Le principal du travail à effectuer sur cette partie sera de traiter les donnée de l'accéléromètre ou de la télécommande..

Robot

Dans cette partie, nous définirons les signaux à émettre pour les moteurs du robot, comment régler leur vitesse, leur orientation, etc.

Câblage du robot :



Réalisation du projet

Création du hotspot avec l'ESP8266

Pour créer l'hotspot à l'aide de l'ESP8266, nous devons mettre en place une communication **Serial** avec l'arduino dans le but de lui envoyer des **commandes AT**. A l'aide de ces commandes on configure les propriétés du hotspot : le mode de fonctionnement, l'adresse IP, le nom du hotspot etc.

Voici les commandes AT que nous avons utilisé :

```
digitalWrite(ledPin, HIGH);  // La LED est niveau pendant l'initialisation

esp8266.begin(115200); // On démare la communication en l'Arduino et l'ESP

delay(2000); // Temps d'attente pour que l'initialisation se déroule correctement

//Initialisation du serveur ESP :
Serial.println( "************************************************************");
Serial.println( "*************** Initialisation AT de l'ESP ******************");
Serial.println( "************************************************************");

sendData("AT+RST\r\n",DEBUG); // reset du module
sendData("AT+CWMODE=3\r\n",DEBUG); // 2 configure comme un point d'accÃ¨s
sendData("AT+CIFSR\r\n",DEBUG); // retourne l'adresse IP
sendData("AT+CIPMUX=1\r\n",DEBUG); // Connexion multiples (Pas obligatoire pour nous)
sendData("AT+CWSAP=\"Hugo\",\"1234test\",5,2\r\n",DEBUG); // DÃ©fini le nom du hotspot et le mdp
sendData("AT+CIPSERVER=1,80\r\n",DEBUG); // mise en route du serveur en port 80
//sendData("AT+CIPSTO=5000\r\n",DEBUG);
//sendData("\r\n",DEBUG);

esp8266.setTimeout(100);
digitalWrite(ledPin, LOW);
```

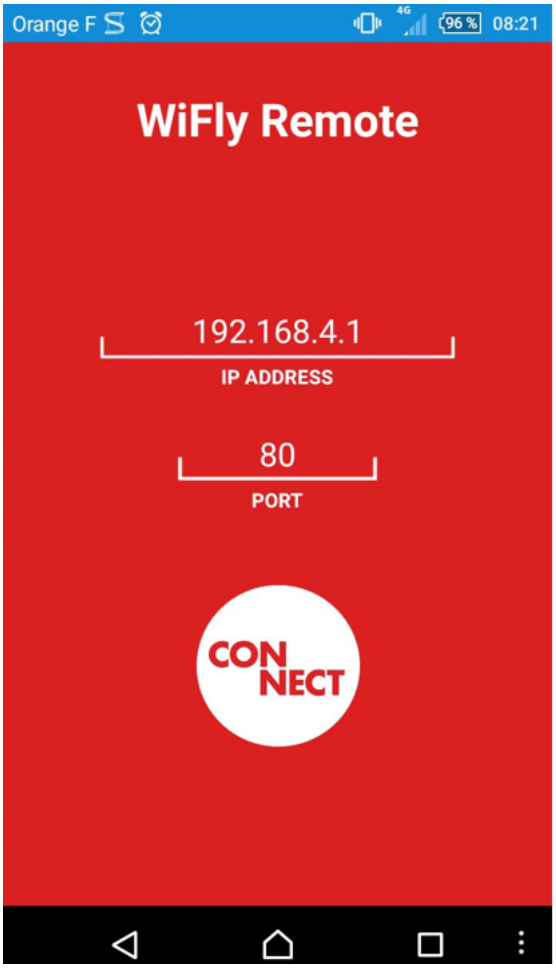
et voici ce que l'ESP renvoie à ces requettes :

```
*************************************************************
*************** Initialisation AT de l'ESP ******************
*************************************************************
rAT+RST

OK

 ets Jan  8 2013,rst cause:4, boot mode:(3,7)

wdt reset
load 0x40100000, len 612, room 16
tail 4
chksum 0x12
load 0x3ffe8000, len 788, room 4
tail 0
chksum 0x50
load 0x3ffe8314, len 264, room 8
tail 0
chksum 0x4a
csum 0x4a

2nd boot version : 1.1
  SPI Speed      : 40MHz
  SPI Mode       : DIO
  SPI Flash Size : 32Mbit
jump to run user1

sAT+CIFSR
+CIFSR:APIP,"192.168.4.1"
+CIFSR:APMAC,"1a:fe:34:9d:fb:03"

OK
AT+CIPMUX=1

OK
AT+CWSAP="Hugo","1234test",5,2

OK
AT+CIPSERVER=1,80
```

Une fois l'initialisation terminée, on apperçoit bien le hotspot et on peut s'y connecter avec le mot de passe 1234test.

Développement de l'application Androïde et connexion

Nous avons ensuite développé une application permettant de se connecter à ce réseau puis d'envoyer des messages correspondants aux commandes au microcontrôleur via l'ESP8266 :

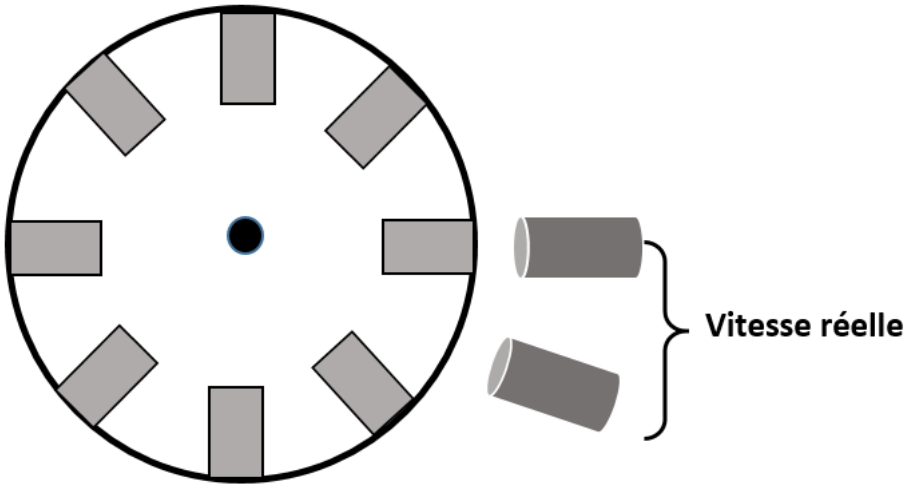


Mise en forme des données :

1. Haut : « F80 »
2. Bas : « B80 »
3. Gauche : « L80 »
4. Droite : « R80 »
5. Stop : « S80 »

Asservissement des moteurs

Nous avons prit en main pour finir l'asservissement des moteurs grâce aux données renvoyées par ces derniers. Voici le fonctionnement de l'axe optique permettant de renvoyer la vitesse réelle des moteurs :



Nous envoyons donc une commande aux moteurs, ils retournent la vitesse réelle, puis le programme calcul une nouvelle commande selon cette erreur.

Code final

Cf. Le fichier E4_Code.pdf dans la liste des fichier, ou alors disponible sur le lien ci-dessous :

https://github.com/TheSirC/Projet-Electronique/blob/Arduino/Arduino/Wifi.ino

<- Retour à la documentation du projet