



République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur

et de la Recherche Scientifique

Université de Tissemsilt



Faculté des Sciences et de la Technologie

Département des Sciences et de la Technologie

Mémoire de fin d'études pour l'obtention du diplôme

de Master académique en :

Filière : Électronique

Spécialité : Instrumentation

Présenté par : **Hechiche Omar et Rahmani Mohamed Elamine**

Thème

Conception et Réalisation d'un Système de Contrôle d'un Véhicule à Distance avec une Carte Arduino

Soutenu le : **29\06\2026**

Devant le Jury :

- | | | | |
|------------------|-----------|-------|---------------|
| • Habel Elhadj | Président | M.C.A | U. Tissemsilt |
| • Marich Mohamed | Encadreur | M.C.A | U. Tissemsilt |
| • Meharrar Aoued | Examineur | M.C.A | U. Tissemsilt |

Année universitaire : 2025-2026

Résumé

Ce mémoire présente la conception et la réalisation d'un véhicule télécommandé basé sur la carte Arduino Mega 2560. L'objectif principal est de développer un système embarqué capable d'assurer le contrôle à distance d'un véhicule à l'aide de la technologie Bluetooth et d'un smartphone Android.

Le système réalisé intègre plusieurs composants électroniques, notamment un module Bluetooth HC-05, un capteur ultrason HC-SR04 pour la détection d'obstacles, un capteur PIR pour la détection de mouvement, un encodeur de vitesse, un capteur de température LM35, un écran LCD I2C ainsi qu'un pilote de moteur L298N. La programmation a été effectuée sous l'environnement Arduino IDE et la simulation du système a été réalisée à l'aide du logiciel Proteus.

Les résultats obtenus montrent que le véhicule peut être commandé à distance de manière fiable tout en assurant des fonctions de surveillance et de sécurité. Ce projet constitue une application pratique des systèmes embarqués et ouvre la voie à des développements futurs dans les domaines de la robotique mobile et de l'Internet des Objets.

Mots-clés : Arduino Mega 2560, véhicule télécommandé, Bluetooth, systèmes embarqués, HC-05, HC-SR04, robot mobile.

Abstract

This dissertation presents the design and implementation of a remote-controlled vehicle based on the Arduino Mega 2560 platform. The main objective is to develop an embedded system capable of controlling a mobile vehicle remotely using Bluetooth technology and an Android smartphone.

The developed system integrates several electronic components, including an HC-05 Bluetooth module, an HC-SR04 ultrasonic sensor for obstacle detection, a PIR motion sensor, a speed encoder, an LM35 temperature sensor, an I2C LCD display, and an L298N motor driver. The software development was carried out using the Arduino IDE environment, while system simulation was performed using Proteus software.

The obtained results demonstrate that the vehicle can be controlled remotely in a reliable manner while providing monitoring and safety functions. This project represents a practical application of embedded systems and offers opportunities for future developments in the fields of mobile robotics and the Internet of Things.

Keywords: Arduino Mega 2560, remote-controlled vehicle, Bluetooth, embedded systems, HC-05, HC-SR04, mobile robot.

الملخص

تتناول هذه المذكرة تصميم وإنجاز مركبة يتم التحكم فيها عن بُعد اعتمادًا على لوحة Arduino Mega 2560 ويهدف هذا العمل إلى تطوير نظام مدمج قادر على التحكم في حركة المركبة عن بعد باستعمال تقنية البلوتوث وهاتف ذكي يعمل بنظام أندرويد.

يعتمد النظام المنجز على مجموعة من المكونات الإلكترونية، من بينها وحدة البلوتوث HC-05 ، ومستشعر الموجات فوق الصوتية HC-SR04 للكشف عن العوائق، ومستشعر الحركة PIR ، ومشفر السرعة، ومستشعر درجة الحرارة LM35 ، وشاشة LCD تعمل ببروتوكول I2C ، بالإضافة إلى دارة قيادة المحركات L298N. وقد تمت برمجة النظام باستخدام بيئة Arduino IDE ، كما تمت محاكاة الدارة باستعمال برنامج Proteus.

أظهرت النتائج المتحصل عليها إمكانية التحكم في المركبة عن بعد بكفاءة مع توفير وظائف المراقبة والسلامة. ويُعد هذا المشروع تطبيقًا عمليًا للأنظمة المدمجة ويمثل قاعدة لتطوير مشاريع مستقبلية في مجال الروبوتات المتنقلة وإنترنت الأشياء.

الكلمات المفتاحية Arduino Mega 2560 ، مركبة موجهة عن بعد، البلوتوث، الأنظمة المدمجة، HC-05 ، HC-SR04 ، روبوت متحرك.

REMERCIEMENT

AVANT TOUT, NOUS REMERCIONS ALLAH, LE TOUT-PUISSANT, QUI NOUS A DONNÉ LA SANTÉ, LA PATIENCE, LA VOLONTÉ ET LE COURAGE NÉCESSAIRES POUR MENER À BIEN CE TRAVAIL.

NOUS TENONS À EXPRIMER NOTRE PROFONDE GRATITUDE À NOTRE ENCADREUR, **M.MARICH MOHAMED** POUR SON ACCOMPAGNEMENT, SES PRÉCIEUX CONSEILS, SA DISPONIBILITÉ ET SON SOUTIEN TOUT AU LONG DE LA RÉALISATION DE CE MÉMOIRE. SES ORIENTATIONS ET SES REMARQUES CONSTRUCTIVES ONT GRANDEMENT CONTRIBUÉ À L'ABOUTISSEMENT DE CE TRAVAIL.

NOUS ADRESSONS ÉGALEMENT NOS SINCÈRES REMERCIEMENTS AUX HONORABLES MEMBRES DU JURY QUI ONT ACCEPTÉ D'ÉVALUER CE MÉMOIRE. NOUS LES REMERCIONS POUR L'INTÉRÊT QU'ILS PORTENT À NOTRE TRAVAIL AINSI QUE POUR LEURS OBSERVATIONS ET SUGGESTIONS QUI CONTRIBUERONT À SON AMÉLIORATION.

NOS REMERCIEMENTS S'ÉTENDENT À TOUS LES ENSEIGNANTS DU DÉPARTEMENT QUI ONT PARTICIPÉ À NOTRE FORMATION ET NOUS ONT TRANSMIS LEURS CONNAISSANCES TOUT AU LONG DE NOTRE PARCOURS UNIVERSITAIRE.

ENFIN, NOUS EXPRIMONS NOTRE GRATITUDE À TOUTES LES PERSONNES QUI, DE PRÈS OU DE LOIN, ONT CONTRIBUÉ À LA RÉALISATION DE CE PROJET.

DÉDICACE 01

JE DÉDIE CE MÉMOIRE À LA MÉMOIRE DE MON CHER PÈRE, QU'ALLAH
LUI FASSE MISÉRICORDE. SON AMOUR, SES SACRIFICES ET LES VALEURS
QU'IL M'A TRANSMISES CONTINUERONT DE M'INSPIRER TOUT AU LONG
DE MA VIE. PUISSE CE TRAVAIL ÊTRE UN TÉMOIGNAGE DE MA
PROFONDE RECONNAISSANCE ET DE MON AFFECTION ÉTERNELLE.

À MA CHÈRE MÈRE, POUR SON AMOUR, SA PATIENCE, SES SACRIFICES ET
SON SOUTIEN INDÉFECTIBLE.

À MA FAMILLE, POUR SES ENCOURAGEMENTS ET SA PRÉSENCE À MES
CÔTÉS.

À MON ENCADREUR, POUR SES PRÉCIEUX CONSEILS, SA DISPONIBILITÉ
ET SON ACCOMPAGNEMENT TOUT AU LONG DE LA RÉALISATION DE CE
MÉMOIRE.

À MES COLLÈGUES ET À TOUS CEUX QUI ONT CONTRIBUÉ, DE PRÈS OU
DE LOIN, À L'ABOUTISSEMENT DE CE TRAVAIL.

DÉDICACE 02

JE DÉDIE CE MÉMOIRE À LA MÉMOIRE DE MA TRÈS CHÈRE MÈRE, QUI
NOUS A QUITTÉS TROP TÔT.

SON AMOUR, SA TENDRESSE, SES SACRIFICES ET SES PRÉCIEUX CONSEILS
ONT ÉTÉ UNE SOURCE INÉPUISABLE DE FORCE ET D'INSPIRATION TOUT
AU LONG DE MA VIE. MÊME SI ELLE N'EST PLUS PARMI NOUS
AUJOURD'HUI, SA PRÉSENCE DEMEURE DANS MON CŒUR ET
M'ACCOMPAGNE À CHAQUE ÉTAPE DE MON PARCOURS.

J'AURAIS TANT SOUHAITÉ QU'ELLE SOIT PRÉSENTE POUR PARTAGER CE
MOMENT SI IMPORTANT. CE TRAVAIL EST LE FRUIT DES VALEURS
QU'ELLE M'A TRANSMISES : LE COURAGE, LA PERSÉVÉRANCE ET LE GOÛT
DE L'EFFORT.

QUE DIEU LUI ACCORDE SA MISÉRICORDE INFINIE, L'ACCUEILLE DANS
SON VASTE PARADIS ET LUI ACCORDE LA PAIX ÉTERNELLE.

À MA MÈRE BIEN-AIMÉE, CE MÉMOIRE EST HUMBLEMENT DÉDIÉ.

Table de matière

Liste des tables

Liste des figures

Abreviations

Chapitre 1 : Généralités sur les systèmes embarqués et les véhicules télécommandés

Introduction	1
1.1 Histoire des systèmes embarqués	1
1.2 Systèmes embarqués	2
1.3 Caractéristiques des systèmes embarqués.....	2
1.4 Classification des systèmes embarqués.....	3
1.4.1 Selon la génération.....	3
1.4.2 Basé sur les exigences fonctionnelles et de performance	4
1.5 Domaines d'application des systèmes embarqués.....	5
1.5.1 Dispositifs médicaux.....	5
1.5.2 Stations de recharge pour véhicules électriques	5
1.5.3 Systèmes automobiles.....	6
1.6 Véhicules télécommandés	7
1.7 Différents types de véhicules télécommandés.....	7
1.7.1 Véhicule thermique télécommandé.....	7
1.7.2 Véhicule électrique télécommandé.....	7
1.8 Principe de fonctionnement	8
1.9 Conclusion.....	8
Chapitre 2 : Étude et conception du système (matériel et logiciel).	9
2.1 Module Arduino	10
2.1.1 Définition du module Arduino2560.....	10
2.1.2 Caractéristiques techniques du module Arduino Mega2560.....	11
2.1.3 Alimentation	12
2.1.4 Mémoires	13
2.1.5 Entrées et sorties numériques	13
2.1.6 Broches analogiques.....	15
2.1.7 Autres broches.....	15
2.1.8 Communications.....	16
2.2 Accessoires de communications avec Arduino Mega2560.....	16
2.2.1 Module Wifi : ESP8266	16
2.2.2 Module Bluetooth HC-05.....	17
2.2.3 Branchement module Bluetooth HC-05	17

2.3	Capteurs.....	17
2.3.1	Caractéristiques des capteurs	18
2.3.2	Classifications des capteurs.....	18
2.3.3	Capteur Ultrason HC-SR04.....	18
2.3.4	Capteur P.I.R.....	21
2.3.5	Encodeur.....	21
2.4	Moteur	22
2.5	Pilote de moteur à double pont en H L298N	22
2.6	Ecran LCD	23
2.6.1	Définition et rôle des écrans LCD	23
2.6.2	I2C LCD Adapter.....	24
2.7	Autres composants utilisés.....	25
2.7.1	LEDs	25
2.7.2	Buzzer	25
2.7.3	Batterie	25
2.8	Logiciel Arduino IDE	25
2.8.1	Structure de programmation d'un sketch Arduino	27
2.8.2	Sélection de la carte Arduino	28
2.9	Choix de la carte.....	29
2.10	PROTEUS (ISIS)	29
2.11	Conclusion.....	30
Chapitre 3 : Réalisation pratique et tests du véhicule télécommandé.....		32
Introduction		33
3.1	Environnement logiciel	33
3.1.1	Environnement de programmation Arduino.....	33
3.1.2	Fonctionnement de l'IDE Arduino	34
3.1.3	Simulation sous Proteus	36
3.2	Matériel utilisé	38
3.2.1	Liste des composants	39
3.2.2	Application de commande sur smartphone	40
3.3	Réalisation et schéma de câblage.....	42
3.3.1	Fonctions principales du système	42
3.3.2	Traitement et sorties	44
3.4	Programmation : code Arduino du récepteur	44
3.4.1	Code source complet.....	44
3.4.2	Architecture matérielle (broches et composants)	51
3.4.3	Cœur du programme : la boucle principale (loop)	52
3.4.4	Mouvements du robot	52

3.4.5	Affichage LCD 20x4	53
3.4.6	Résumé des commandes acceptées.....	53
3.5	Réalisation et tests.....	53
3.5.1	Alimentation électrique.....	53
3.5.2	Montage et câblage.....	54
3.5.3	Tests fonctionnels.....	55
3.6	Conclusion.....	56
Conclusion generale		58

Liste des tableaux

Tableau 1-caractéristiques de la carte Arduino Mega2560.....	12
<i>Tableau 2-Différentes broches entrées/sorties numériques.....</i>	<i>15</i>
<i>Tableau 3-Caractéristique de HC-SR04.....</i>	<i>19</i>
<i>Tableau 4-Le matériel utilisé.....</i>	<i>39-40</i>
<i>Tableau 5-Les fonctions principales de système.....</i>	<i>44</i>

Liste des figures

Fig. 1- Architecture d'un système embarqué.....	2
Fig. 2- Scanners d'échographie.....	5
Fig. 3- Station de recharge pour véhicules électriques.....	6
Fig. 4- Systèmes automobile.....	6
Fig. 5- Véhicule télécommandée thermique.....	7
Fig. 6- Véhicule télécommandée électrique.....	8
Fig. 7 – Module Arduino Mega2560	11
Fig. 8 – Préface de raccordements Bluetooth	17
Fig. 9 – Chaine d'action d'un capteur	18
Fig. 10 – Capteur ultrason HC-SR04.....	19
Fig. 11 – Le fonctionnement de HC-SR04	20
Fig. 12 – Capteur P.I.R	21
Fig. 13 – Encodeur optique.....	22
Fig. 14 – Moteur à courant continu (CC).....	22
Fig. 15 – Pilote de moteur H L298N.....	23
Fig. 16 – Écran LCD	23
Fig. 17 – Rectangles de pixel	23
Fig. 18 – Adaptateur I2C pour écran LCD.....	24
Fig. 19– Diode électroluminescente.....	24
Fig. 20 – Une batterie avec adaptateur.....	25
Fig. 21 – Interface du logiciel IDE.....	25
Fig. 22 – Boutons d'action.....	26
Fig. 23 – Arduino Program Structure.....	27
Fig. 24 – Sélection de la carte Arduino Mega 2560.....	28
Fig. 25 – Icône ISIS.....	29
Fig. 26 – Interface du Proteus.....	30
Fig. 27 – Connexion entre la carte Arduino et le PC.....	33
Fig. 28 – Fenêtre de l'IDE Arduino à l'ouverture (duplicate number).....	34
Fig. 29 – Sélection du type de carte dans l'IDE Arduino.....	35
Fig. 30 – Exemple de code de configuration Bluetooth	35
Fig. 31 – Combinaison PC, Arduino, module Bluetooth et smartphone	35
Fig. 32 – Schéma synoptique du système	36

Fig. 33 – Simulation du véhicule en marche avant.....	37
Fig. 34 – Simulation du véhicule en marche arrière	37
Fig. 35 – Kit de composants Arduino	38
Fig. 36 – Multimètre et fer à souder	38
Fig. 37 – Câble USB pour Arduino Méga.....	40
Fig. 38 – Kit de châssis du véhicule robot (roues et moteurs CC).....	40
Fig. 39 – Combinaison des différents éléments de contrôle à distance (duplicate number)	41
Fig. 40 – Interface de l'application Bluetooth RC Controller	41
Fig. 41 – Utilisation de l'accéléromètre du smartphone pour détecter l'orientation.....	42
Fig. 42 – Prototype du véhicule	42
Fig. 43 – Schéma physique des éléments connectés à l'Arduino Méga 2560.....	43
Fig. 44- Capteur ultrason HC-SR04 monté à l'avant du châssis.....	54
Fig. 45- Câblage de la carte Arduino, du driver moteur et des capteurs.....	55
Fig. 46- Détail du câblage entre l'Arduino, le module L298N et l'alimentation.....	55
Fig. 47- Prototype final du véhicule télécommandé, assemblé et câblé.....	56

Abbreviations

AC : Alternating Current (Courant Alternatif)

ASIC : Application Specific Integrated Circuit

CPU : Central Processing Unit

DC : Direct Current (Courant Continu)

DSP : Digital Signal Processor

EEPROM : Electrically Erasable Programmable Read-Only Memory

ESP8266 : Espressif System 8266

FTDI : Future Technology Devices International

GND : Ground (Masse)

HC-05 : Module Bluetooth HC-05

HC-SR04 : Capteur ultrasonique de mesure de distance

I2C : Inter-Integrated Circuit

ICSP : In-Circuit Serial Programming

IDE : Integrated Development Environment

IoT : Internet of Things (Internet des Objets)

ISIS : Intelligent Schematic Input System

LCD : Liquid Crystal Display

LED : Light Emitting Diode

LM35 : Capteur de température LM35

MIT : Massachusetts Institute of Technology

MISO : Master In Slave Out

MOSI : Master Out Slave In

PDA : Personal Digital Assistant

PIR : Passive Infrared Sensor

PWM : Pulse Width Modulation

RAM (SRAM) : Static Random Access Memory

ROM : Read Only Memory

RX : Réception (Receive)

SCADA : Supervisory Control And Data Acquisition

SCK : Serial Clock

SoC : System on Chip

SPI : Serial Peripheral Interface

SS : Slave Select

TTL : Transistor-Transistor Logic

TX : Transmission (Transmit)

UART : Universal Asynchronous Receiver Transmitter

UHF : Ultra High Frequency

USB : Universal Serial Bus

Vcc : Tension d'alimentation positive

WAN : Wide Area Network

Wi-Fi : Wireless Fidelity

Introduction générale

Le développement rapide des systèmes embarqués a profondément transformé les domaines de l'automatisation et de la robotique. Ces systèmes, basés sur des microcontrôleurs, permettent de contrôler et de superviser divers dispositifs en temps réel, tout en améliorant la précision, la fiabilité et la sécurité des applications.

Parmi les applications les plus répandues figurent les véhicules mobiles télécommandés, qui constituent une plateforme pédagogique idéale dans les domaines de l'électronique, de la programmation et de la mécatronique. Grâce à l'évolution des technologies de communication sans fil, telles que le Bluetooth et le Wi-Fi, il est désormais possible d'assurer un contrôle à distance simple, fiable et efficace.

Dans ce contexte, ce projet, intitulé « Conception et réalisation d'un système de contrôle d'un véhicule à distance basé sur la carte Arduino », porte sur la conception et la mise en œuvre d'un prototype expérimental capable d'exécuter des déplacements en fonction des commandes reçues à distance via un module de communication sans fil HC-05 Bluetooth Module.

Ce travail permet de mettre en œuvre des notions fondamentales telles que la commande des moteurs, la communication série, l'acquisition de données et l'intégration de capteurs. Il ouvre également la voie à des perspectives d'amélioration, notamment l'automatisation partielle du système ou l'évitement d'obstacles.

Ce mémoire est structuré en trois chapitres :

Chapitre 1 : Généralités sur les systèmes embarqués et les véhicules télécommandés.

Chapitre 2 : Étude et conception du système (matériel et logiciel).

Chapitre 3 : Réalisation pratique et tests du véhicule télécommandé.

Chapitre 1 : Généralités sur les systèmes embarqués et les véhicules télécommandés

Introduction

Les systèmes embarqués occupent aujourd'hui une place essentielle dans notre vie quotidienne, en raison de leur intégration dans une grande variété d'équipements électroniques et industriels. Un système embarqué est conçu pour accomplir une tâche spécifique avec des contraintes strictes en termes de performance, de coût et de consommation d'énergie. On les retrouve dans plusieurs domaines tels que l'automobile, l'aéronautique, la médecine et les télécommunications.

Parmi les applications concrètes des systèmes embarqués, les véhicules télécommandés représentent un exemple simple et pédagogique permettant de comprendre leur fonctionnement. Ces systèmes combinent à la fois des composants matériels (microcontrôleurs, capteurs, moteurs) et des techniques de communication sans fil pour assurer le contrôle à distance.

L'étude des systèmes embarqués et des véhicules télécommandés permet ainsi de mieux comprendre les principes fondamentaux de la conception des systèmes électroniques modernes, notamment l'interaction entre le matériel et le logiciel, ainsi que la gestion du temps réel et de la communication.

Dans ce chapitre, nous allons présenter les généralités sur les systèmes embarqués, puis étudier le fonctionnement et les caractéristiques des véhicules télécommandés, avant de mettre en évidence la relation entre ces deux domaines.

1.1 Histoire des systèmes embarqués

- Années 1960 : Charles Stark Draper, au (M.I.T.), a développé l'un des premiers systèmes embarqués utilisé dans l'ordinateur de guidage du programme Apollo program, destiné à la navigation des engins spatiaux.
- 1965 : Autonetics a conçu l'ordinateur D-17B, utilisé dans le système de guidage du missile Minuteman missile. Il s'agit de l'un des premiers ordinateurs embarqués produits en série.
- 1968 : Lancement du premier système embarqué dans le domaine automobile, marquant le début de l'intégration de l'électronique embarquée dans les véhicules.
- 1996 : Microsoft a introduit Windows CE (plus tard appelé Windows Embedded Compact), un système d'exploitation conçu pour les dispositifs embarqués à ressources limitées.
- Fin des années 1990 : Apparition des premiers systèmes embarqués basés sur Linux, offrant des solutions flexibles, économiques et personnalisables pour diverses applications[1].

1.2 Systèmes embarqués

Un système embarqué est un système électronique ou électromécanique qui combine un logiciel (software) et du matériel (hardware) afin d'accomplir un objectif spécifique. Chaque système embarqué est unique, possédant un matériel et un micrologiciel extrêmement spécialisés et adaptés à son usage particulier.

Ces systèmes sont rapidement devenus des composants indispensables pour tout équipement ou produit dans chaque secteur d'activité, incluant les objets domestiques, les dispositifs médicaux, le contrôle industriel, les appareils médicaux et les télécommunications[2].

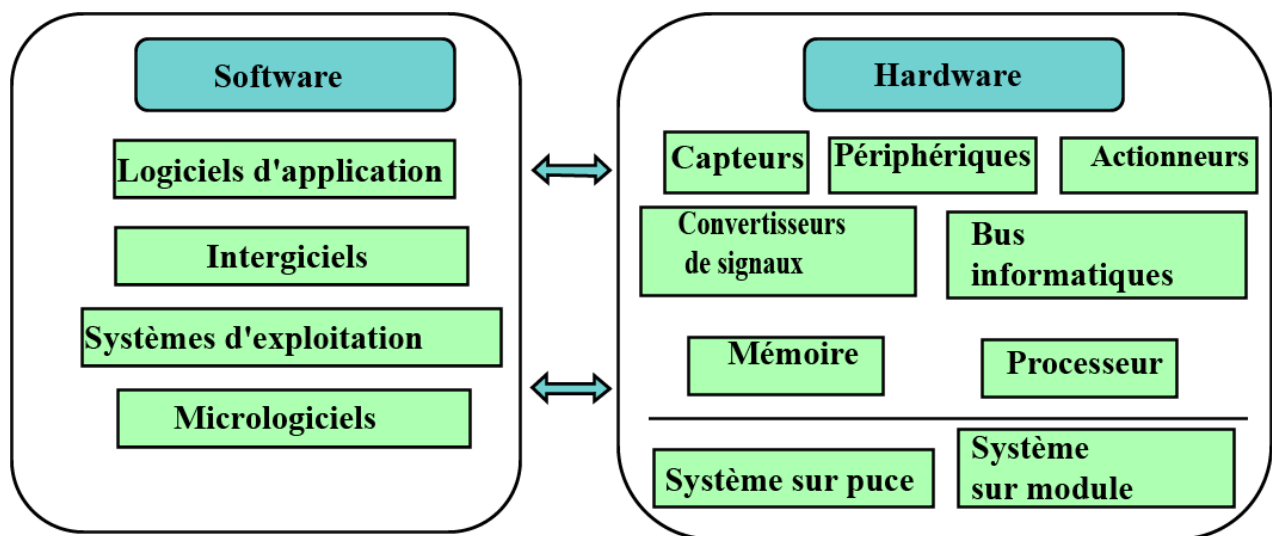


Fig. 1_Architecture d'un système embarqué.

1.3 Caractéristiques des systèmes embarqués

Le terme « embarqué » souligne la connexion intime entre le logiciel et le matériel, par opposition à des composants isolés. Pour bien comprendre ce concept, il faut considérer les caractéristiques suivantes :

- **Processeur et mémoire :** Les systèmes de base nécessitent peu de mémoire, tandis que les systèmes complexes requièrent plus de stockage et peuvent utiliser des processeurs multi-cœurs (CPUs).
- **Efficacité des coûts de fabrication :** Ils présentent des coûts de production réduits et consomment moins d'énergie que les ordinateurs à usage général.
- **Fonctionnalité avancée :** Un système embarqué est conçu pour exécuter des fonctions hautement spécialisées et avancées.

Chapitre 1 Généralités sur les systèmes embarqués et les véhicules télécommandés

- Fonctionnement en temps réel : Ils opèrent selon des structures temporelles prédéterminées ; le logiciel ordonne au matériel d'accomplir des tâches dans un délai strict.
- Contraintes de conception rigoureuses : Les systèmes embarqués sont soumis à des contraintes de conception strictes en termes de coût, de performance, de taille et de consommation d'énergie[3].

1.4 Classification des systèmes embarqués

1.4.1 Selon la génération

Une autre classification des systèmes embarqués repose sur l'ère technologique ou la génération au cours de laquelle ils ont été créés. Elle comprend :

1.4.1.1 Première génération

Depuis les années 1960, ces systèmes reposaient sur des composants individuels tels que les transistors et les résistances. Ils ont été conçus avec des microprocesseurs 8 bits et des microcontrôleurs 4 bits, et leur matériel ainsi que leur logiciel étaient relativement simples. Les systèmes embarqués de première génération incluent les claviers téléphoniques numériques et les dispositifs de contrôle de moteurs pas à pas [3].

1.4.1.2 Deuxième génération

Créés dans les années 1970, ces systèmes utilisaient des circuits intégrés (CI) afin d'améliorer les performances et de réduire la taille. Ils ont remplacé les processeurs 8 bits et les contrôleurs 4 bits par des microprocesseurs 16 bits et des microcontrôleurs 8 bits, ce qui a permis de concevoir des systèmes plus puissants et plus complexes que ceux de la première génération. Les systèmes embarqués de deuxième génération incluent les systèmes d'acquisition et de contrôle de données (SCADA) [3].

1.4.1.3 Troisième génération

Ces systèmes ont été développés dans les années 1980 avec les microprocesseurs comme unité principale de traitement. Durant cette période, des processeurs et contrôleurs spécialisés sont apparus, tels que les processeurs de traitement du signal numérique (DSP) et les circuits intégrés spécifiques à une application (ASIC). Les systèmes d'exploitation temps réel ont également évolué durant cette génération, avec des microprocesseurs 32 bits et des microcontrôleurs 16 bits. Les systèmes embarqués de troisième génération incluent la robotique, le contrôle des processus de fabrication et les réseaux embarqués [3].

1.4.1.4 Quatrième génération

Développée dans les années 1990, cette génération utilise des microprocesseurs et des logiciels améliorés afin d'augmenter les performances et de réduire la consommation d'énergie. Les progrès récents en microprocesseurs et microcontrôleurs, ainsi que les nouvelles technologies telles que les systèmes sur puce (SoC), les processeurs reconfigurables, les processeurs multicœurs et les coprocesseurs, ont considérablement amélioré les performances des systèmes embarqués. Ces systèmes utilisent généralement des systèmes d'exploitation temps réel performants. Les systèmes embarqués de quatrième génération incluent les objets intelligents et les appareils photo numériques [3].

1.4.2 Basé sur les exigences fonctionnelles et de performance

Cette classification des systèmes embarqués est déterminée en fonction de leurs exigences fonctionnelles et de performance. Ce sont des systèmes informatiques hautement spécialisés, programmés pour exécuter des tâches spécifiques. Ils sont classés en quatre catégories:

1.4.2.1 Systèmes embarqués en réseau

Ces systèmes sont connectés à un réseau et permettent aux utilisateurs d'accéder aux ressources à partir d'un emplacement unique. Ce réseau peut être un réseau local (LAN), un réseau étendu (WAN) ou Internet, avec des connexions filaires ou sans fil. Cette catégorie constitue le domaine connaissant la croissance la plus rapide dans les applications des systèmes embarqués. Les systèmes embarqués en réseau comprennent les sites web embarqués, les systèmes de sécurité domestique, les distributeurs automatiques de billets (DAB) ainsi que les dispositifs de lecture de cartes [3].

1.4.2.2 Systèmes embarqués en temps réel

Ces systèmes produisent les résultats requis dans une limite de temps spécifique. Ils respectent des contraintes temporelles strictes pour l'exécution des tâches. Ils peuvent être divisés en systèmes embarqués temps réel souple (soft real-time) et systèmes embarqués temps réel strict (hard real-time). Les systèmes embarqués en temps réel comprennent les systèmes de contrôle des airbags des véhicules ainsi que les systèmes de contrôle de vol [3].

1.4.2.3 Systèmes embarqués mobiles

Ces systèmes sont compacts, faciles à utiliser et économes en ressources. Ils sont souvent intégrés dans des appareils portables tels que les téléphones mobiles, les appareils photo numériques, les lecteurs MP3 et les assistants numériques personnels (PDA). Ces systèmes sont conçus pour être simples d'utilisation et facilement transportables, ce qui les rend idéaux pour les applications mobiles et nomades [3].

1.4.2.4 Systèmes embarqués autonomes

Ces systèmes sont autonomes et ne dépendent pas d'autres systèmes pour fonctionner. Ils reçoivent des données sous forme analogique ou numérique, les traitent, puis fournissent une sortie. Parmi les exemples courants figurent les lecteurs MP3, les appareils photo numériques et les consoles de jeux vidéo [3].

1.5 Domaines d'application des systèmes embarqués

1.5.1 Dispositifs médicaux

Les dispositifs médicaux au sein des établissements de santé intègrent des systèmes embarqués depuis déjà un certain temps. Une nouvelle catégorie de dispositifs médicaux utilise désormais ces systèmes pour assister les patients nécessitant une surveillance fréquente et une attention constante à domicile.

Ces systèmes sont dotés de capteurs permettant de collecter des données relatives à la santé des patients, telles que la fréquence cardiaque, le pouls ou les mesures provenant d'implants. Ces informations sont ensuite transmises vers un Cloud, où un médecin peut les consulter à distance et sans fil sur son propre appareil. Les dispositifs médicaux sont ainsi largement utilisés pour diagnostiquer et traiter les patients de manière efficace ; en voici quelques exemples :

- Le stimulateur cardiaque (Pacemaker)
- Le défibrillateur
- Les scanners d'échographie (Ultrasound scanners)



Fig. 2- Scanners d'échographie

1.5.2 Stations de recharge pour véhicules électriques

Les stations de recharge pour véhicules électriques sont équipées de points ou d'unités de charge qui fournissent l'énergie électrique nécessaire pour alimenter les véhicules connectés. Un

Chapitre 1 Généralités sur les systèmes embarqués et les véhicules télécommandés

système embarqué est intégré à la station de recharge pour assurer la gestion de l'affichage graphique, signaler tout problème technique lié au dispositif et alerter les techniciens lorsqu'une maintenance est nécessaire.



Fig. 3- Station de recharge pour véhicules électriques

1.5.3 Systèmes automobiles

Les systèmes embarqués automobiles sont conçus et installés pour renforcer la sécurité des véhicules. Grâce aux systèmes de sécurité intégrés, le taux de mortalité routière a considérablement chuté ces dernières années. L'industrie automobile redouble d'efforts pour équiper les véhicules de capteurs et de technologies de pointe, ce qui serait impossible sans l'apport des systèmes embarqués.

Parmi les exemples clés de systèmes de sécurité active, on peut citer : le régulateur de vitesse adaptatif, l'alerte de panne, la reconnaissance des piétons, l'assistance au changement de voie et les airbags. Ces fonctionnalités sont conçues pour atténuer les risques d'accidents et stimuler la demande mondiale pour les systèmes embarqués.



Fig. 4- Systèmes automobile

1.6 Véhicules télécommandés

La voiture télécommandée ou bien radiocommandée est un modèle réduit de voiture qui peut être contrôlé à distance à l'aide d'un émetteur ou d'une télécommande (sans fil).

Pour faire cette opération il suffit d'utiliser par exemple un Smartphone qui donne des ordres à la voiture sous forme des ondes radio UHF 2.4GHz au moyen de Bluetooth, ainsi l'utilisateur pourra la contrôler dans les quatre directions, même si l'arrêter et modifier sa vitesse[4].

1.7 Différents types de véhicules télécommandés

On peut distinguer deux types de voitures télécommandées

1.7.1 Véhicule thermique télécommandé

C'est un modèle réduit d'une voiture pour adulte à carburant. Equipée d'une motorisation à combustion, elle est commandée à distance par une manette alimentée par des piles. Ce genre de jouet est normalement réservé aux utilisateurs expérimentés. Il est conseillé d'utiliser la voiture à moteur thermique loin des habitations. En effet, contrairement aux engins fonctionnant uniquement avec des piles, celle-ci fait beaucoup plus de bruit. De plus, une odeur liée à la combustion du carburant émane de la voiture, et peut ne pas être très agréable dans un espace fermé ou confiné[4].



Fig. 5- Véhicule thermique télécommandé

1.7.2 Véhicule électrique télécommandé

Les modèles à alimentation électrique utilisent des unités de contrôle de vitesse, mécaniques ou électroniques, pour réguler la puissance transmise au moteur. Cette puissance est proportionnelle à la commande envoyée par l'émetteur.

Dans les systèmes modernes, la tension est modulée (ou 'pulsée') à l'aide de transistors, ce qui permet d'obtenir une sortie variable avec des transitions fluides et une grande efficacité énergétique.

Chapitre 1 Généralités sur les systèmes embarqués et les véhicules télécommandés

Ces contrôleurs électroniques utilisent des composants à semi-conducteurs pour réguler le cycle de service (duty cycle). De plus, ils permettent d'utiliser le moteur comme un frein magnétique, offrant ainsi une précision de pilotage supérieure aux anciens systèmes mécaniques, qui se contentaient de commuter entre différentes résistances[4].



Fig. 6- Véhicule télécommandé électrique

1.8 Principe de fonctionnement

Les voitures télécommandées sont plus que de simples jouets : ce sont de véritables exemples de l'ingénierie en miniature. Elles fonctionnent grâce à un système sophistiqué qui inclut un émetteur radio, un récepteur, des moteurs, ainsi qu'une batterie. L'émetteur envoie des signaux au récepteur embarqué dans la voiture qui active les moteurs selon les commandes transmises par l'utilisateur. Cette interaction permet de contrôler la vitesse, la direction, et parfois d'autres fonctionnalités spécifiques comme l'accélération ou le freinage[5].

1.9 Conclusion

Ce premier chapitre nous a permis de définir les systèmes embarqués et d'explorer leur vaste champ d'application, allant de la médecine à l'automobile. Nous avons également distingué les différents types de véhicules télécommandés, en mettant l'accent sur les modèles électriques et leurs modes de contrôle. Cette étude théorique pose les bases nécessaires à la compréhension de notre projet. Dans le chapitre suivant, nous détaillerons la conception matérielle de notre véhicule, en présentant les composants choisis, tels que la carte Arduino Uno et le module Bluetooth HC-05.

Chapitre 2 : Étude et conception du système (matériel et logiciel).

Introduction

Dans ce chapitre, nous allons présenter l'étude et la conception du système embarqué dédié au véhicule télécommandé. Cette étape constitue une phase essentielle du projet, car elle permet de définir l'architecture générale du système ainsi que les différents composants matériels et logiciels nécessaires à son fonctionnement.

L'étude matérielle porte principalement sur le choix des composants électroniques utilisés, tels que le microcontrôleur, les modules de communication, les capteurs et les éléments de commande des moteurs. Le choix de ces composants a été effectué selon plusieurs critères, notamment la performance, la consommation énergétique, le coût et la facilité d'intégration.

Par ailleurs, la conception logicielle concerne le développement des programmes assurant le contrôle et la communication entre les différents modules du système. Cette partie inclut également l'élaboration de l'algorithme de fonctionnement du véhicule télécommandé afin d'assurer une exécution fiable et efficace des différentes tâches.

Enfin, ce chapitre présente l'architecture globale du système ainsi que les différentes étapes de conception permettant d'aboutir à une solution fonctionnelle et adaptée aux objectifs du projet.

2.1 Module Arduino

2.1.1 Définition du module Arduino2560

Arduino est son synonyme Genuino, est une marque qui couvre des cartes matériellement libre, une carte électronique open source de prototypage repose sur une plateforme flexible, autant du côté matériels que logiciel. Une plateforme basée sur des Entrées/Sorties, utilisées pour construire des objets interactifs indépendants, ou bien connecté à un ordinateur pour communiquer avec des logiciels, elle peut être connecté aux terminaux mobiles par accessoires de communications, le Wifi, Bluetooth, shield Arduino wifi, etc. pour effectuer des tâches diverses. Utilisé dans les applications industrielles, embarquée et la domotique, mais aussi dans les domaines différents comme l'art contemporain, le pilotage et la commande des appareils mobiles [6].

La carte repose sur un circuit intégré (un mini-ordinateur appelé également microcontrôleur) associée à des entrées sorties qui permettent à l'utilisateur de brancher différents types d'éléments externes.

- **Côté entrées**, des capteurs qui collectent des informations sur leur environnement comme la variation de température via une sonde thermique, le mouvement via un détecteur de présence ou un accéléromètre, le contact via un bouton-poussoir, etc.
- **Côté sorties**, des actionneurs qui agissent sur le monde physique telle une lampe qui

produit de la lumière, un moteur qui actionne un bras articulé, un coulissant etc.

La plateforme Arduino se présente sur plusieurs séries à savoir : Arduino UNO, NANO, Lilypad, DUE et la Méga. Dans cette partie, on s'intéresse à l'Arduino mega 2560 qui sera le cœur de notre système de commande [7].



Fig. 7- Module Arduino Mega2560

L'Arduino Mega2560 dispose:

- Alimentation externe provenir soit d'un adaptateur (AC-DC) ou la batterie (7 à 12V).
- Une connexion USB.
- Un bouton poussoir (RESET), pour réinitialiser le microcontrôleur.
- Un connecteur ICSP (programmation "in-circuit").
- SPI (Interface Série Périphérique) : broches 10(SS), 11(MOSI), 12(MISO) ,13(SCK).Ces broches supportant la communication SPI disponible avec la librairie pour communication.
- 54 Broches numériques Entrées/Sorties (de 0 à 53), en utilisant les fonctions pinMode(), digitalWrite () et digitalRead (). Ils fonctionnent à 5 volts, dont 14 peuvent être utilisées en sortie PWM 8 bits avec la fonction analogWrite ().
- 16 entrées analogiques (0A à 15A).
- 04 UART (ports Hardware).
- Séries (RX) utilisé pour recevoir et (TX) pour transmettre les données sériées de niveau TTL. Ces broches sont également connectées aux broches correspondantes d'ATmega.

2.1.2 Caractéristiques techniques du module Arduino Mega2560

Parmi les caractéristiques comprend ce module, on trouve : [8]

Microcontrôleur	ATMega2560
Tension de fonctionnement	5V
Tension entrée (recommandée)	7 – 12 V
Tension d'entrée (limites)	6 – 20 V
Broches E/S numériques	54 (dont 15 fournir une sortie PWM)
Broches d'entrée analogique	16
Courant continu par E/S pin	40 mA
DC courant pour 3.3 v pin	50 mA
Mémoire flash	256 Ko de dont 8 Kb utilisé par boot loader
SRAM	8 Ko
EEPROM	4 Ko
Fréquence d'horloge	16 MHz
Poids	37 g
Dimension	101 mm x 53.4 mm

Tableau 1-Caractéristiques de la carte Arduino Mega2560

2.1.3 Alimentation

L'Arduino Méga 2560 peut être alimentée via la connexion USB ou avec une alimentation externe. Dans le premier cas, la source d'alimentation est automatiquement sélectionnée.

Les broches d'alimentation sont les suivants : [9]

- **Vin** : représente la tension d'entrée positive lorsque la carte Arduino est alimentée par une source de tension externe, à distinguer du 5 V fourni par la connexion USB ou par une autre source 5 V régulée. Cette broche peut être utilisée pour alimenter la carte directement. Lorsque l'alimentation est fournie via le connecteur d'alimentation (jack), la tension d'entrée peut également être récupérée à partir de cette broche [9].
- **5V** : La tension régulée utilisée pour faire fonctionner le microcontrôleur et les autres composants de la carte Le 5V régulé fourni par cette broche peut donc provenir soit de la tension d'alimentation Vin via le régulateur de la carte, ou bien de la connexion USB ou de tout autre source d'alimentation régulée ;

- **3.3V** : cette broche fournit une tension de 3,3 V générée par le circuit intégré FTDI, chargé de l'adaptation des signaux entre le port USB de l'ordinateur et le port série du microcontrôleur ATmega de la carte Arduino. Cette sortie est utile pour alimenter certains composants externes fonctionnant en 3,3 V au lieu de 5 V. Le courant maximal disponible sur cette broche est de 50 mA.
- **GND** : broche de masse (0 V) de la carte Arduino. Elle sert de référence commune pour les tensions et permet de relier les différents composants du circuit au potentiel de masse.

2.1.4 Mémoires

La carte Arduino Mega 2560 dispose de 256 Ko de mémoire FLASH destinés au stockage du programme, dont 8 Ko sont utilisés par le bootloader. Elle possède également 8 Ko de mémoire SRAM, utilisée pour le stockage temporaire des données pendant l'exécution du programme, ainsi que 4 Ko de mémoire EEPROM, une mémoire non volatile permettant de conserver les données même après la mise hors tension. Cette dernière peut être exploitée à l'aide de la bibliothèque EEPROM.h [9].

2.1.5 Entrées et sorties numériques

Chacune des 54 broches numériques de la carte Arduino Mega 2560 peut être configurée soit comme entrée numérique, soit comme sortie numérique, à l'aide des instructions pinMode(), digitalWrite() et digitalRead() du langage Arduino. Ces broches fonctionnent avec une tension logique de 5 V.

Communication Série	✓ Port Série Serial 0 : 0 (RX) and 1 (TX) ✓ Port Série Serial 1: 19 (RX) and 18 (TX) ; ✓ Port Série Serial 2: 17 (RX) and 16 (TX) ; ✓ Port Série Serial 3: 15 (RX) and 14 (TX) • Utilisées pour recevoir (RX) et transmettre (TX) les données sériel de niveau TTL. • Les broches 0 (RX) and 1 (TX) sont connectées aux broches correspondantes du circuit intégré ATmega8U2 programmé en convertisseur USB-vers-série de la carte, composant qui
---------------------	---

	assure l'interface entre les niveaux TTL et le port USB de l'ordinateur.
Interruptions Externes	✓ Broche 2 (interruption 0), ✓ Broche 3 (interruption 1), ✓ Broche 18 (interruption 5), ✓ Broche 19 (interruption 4), ✓ Broche 20 (interruption 3), ✓ Broche 21 (interruption 2). Ces broches peuvent être configurées pour déclencher une interruption sur une valeur basse, sur un front montant ou descendant, ou sur un changement de valeur. (Voir l'instruction <code>attachInterrupt ()</code>)
Impulsion P.W.M (Largeur d'impulsion modulée)	Broches 0 à 13 fournissent une impulsion PWM 8-bits à l'aide de l'instruction <code>analogWrite ()</code> .
S.P.I (Interface Série Périphérique)	✓ Broche 50 (MISO), ✓ Broche 51 (MOSI), ✓ Broche 52 (SCK), ✓ Broche 53 (SS), Ces broches supportent la communication S.P.I (Interface Série Périphérique) disponible avec la librairie pour communication SPI. Les broches SPI sont également connectées sur le connecteur ICSP qui est mécaniquement compatible avec les cartes Uno, Duemilanove et Diecimila.
I2C	✓ Broche 20 (SDA), ✓ Broche 21 (SCL). • Supportent les communications de protocole I2C (ou interface TWI (Two Wire Interface - Interface "2 fils"),

	disponible en utilisant la librairie Wire/I2C (ou TWI-Two-Wire interface "2fils"). • Noter que ces broches n'ont pas le même emplacement que sur les cartes Uno, Duemilanove ou Diecimila.
LED	Broche 13. Il y a une LED incluse dans la carte connectée à la broche 13. Lorsque la broche est au niveau HAUT, la LED est allumée, lorsque la broche est au niveau BAS, la LED est éteinte.

Tableau 2-Différentes broches entrées/sorties numériques.

2.1.6 Broches analogiques

La carte Arduino Méga 2560 dispose de 16 entrées analogiques, chacune pouvant fournir une mesure d'une résolution de 10 bits (c-à-d sur 1024 niveaux soit de 0 à 1023) à l'aide de la très utile fonction `analogRead ( )` du langage Arduino. Par défaut, ces broches mesurent entre le 0V (valeur 0) et le 5V (valeur 1023), [9] mais il est possible de modifier la référence supérieure de la plage de mesure en utilisant la broche AREF et l'instruction `analogReference ( )` du langage Arduino.

Les broches analogique peuvent être utilisées en tant que broches numériques.

2.1.7 Autres broches

La carte Arduino Méga 2560 dispose également de deux broches supplémentaires importantes:

- **AREF** (Analog Reference)

Cette broche permet de définir une tension de référence externe pour les entrées analogiques, différente de la tension standard de 5V. Elle est utilisée avec l'instruction ``analogReference()`` afin d'améliorer la précision des mesures analogiques dans certaines applications.

- **RESET**

Lorsque cette broche est mise à l'état BAS (LOW), le microcontrôleur est réinitialisé, ce qui provoque le redémarrage du programme exécuté. Cette broche est généralement utilisée pour ajouter un bouton de réinitialisation externe au circuit électronique.

2.1.8 Communications

La carte Arduino Méga 2560 dispose de toute une série de facilités pour communiquer avec un ordinateur, une autre carte Arduino, ou avec d'autres microcontrôleurs. Elle dispose de 4 UARTs (émetteur-récepteur asynchrone universel) pour communication série de niveau TTL (5V) et qui est disponible sur les broches 0 (RX) et 1 (TX).

Un circuit intégré ATmega8U2 sur la carte assure la connexion entre cette communication série de l'un des ports série de l'ATmega 2560 vers le port USB de l'ordinateur qui apparaît comme un port COM virtuel pour les logiciels de l'ordinateur.

Le code utilisé pour programmer l'ATmega8U2 utilise le driver standard USB COM, et aucun autre driver externe n'est nécessaire. Cependant, sous Windows, un fichier.inf est requis.

Le logiciel Arduino inclut une fenêtre terminal série (ou moniteur série) sur l'ordinateur et qui permet d'envoyer des textes simples depuis et vers la carte Arduino.

Les deux LEDs RX et TX sur la carte clignote lorsque les données sont transmises via le circuit intégré ATmega8U2 utilisé en convertisseur USB-vers-série et la connexion USB vers l'ordinateur (mais pas pour les communications série sur les broches 0 et 1).

Le code utilisé pour programmer l'ATmega8U2 utilise le driver standard USB COM, et aucun autre driver externe n'est nécessaire. Cependant, sous Windows, un fichier.inf est requis.

Le logiciel Arduino inclut une fenêtre terminal série (ou moniteur série) sur l'ordinateur et qui permet d'envoyer des textes simples depuis et vers la carte Arduino.

Les deux LEDs RX et TX sur la carte clignote lorsque les données sont transmises via le circuit intégré ATmega8U2 utilisé en convertisseur USB-vers-série et la connexion USB vers l'ordinateur (mais pas pour les communications série sur les broches 0 et 1). L'ATmega2560 supporte également la communication par protocole I2C (ou interface TWI (Two Wire Interface - Interface "2 fils") et SPI.

2.2 Accessoires de communications avec Arduino Mega2560

Le constructeur a suggéré qu'une telle carte doit être dotée de plusieurs ports de communications ; on peut éclaircir actuellement quelques types:

2.2.1 Module Wifi : ESP8266

Est un composant électronique permettant de rajouter la communication Wifi à n'importe quel projet. Il s'inscrit donc directement dans le domaine de l'IDO.

ESP (système sur puce) produit par Expressif Systems. C'est une puce hautement intégrée

conçue pour fournir une connectivité internet complète dans un petit paquet. Il fonctionne a niveau logique de 3.3V, et un courant élevé 300 mA, pour cette raison, il faut prévoir une alimentation autonome[10].

2.2.2 Module Bluetooth HC-05

Le module microcontrôleur arduino Bluetooth est la plateforme populaire Arduino avec une connexion série Bluetooth à la place d'une connexion USB. Très faible consommation d'énergie, très faible portée, faible débit, tres bon marché et peu encombrant[11].

2.2.3 Branchement module Bluetooth HC-05

Module HC-06 permettent simplement de disposer d'une liaison Bluetooth sur projet. Il est vraiment conseillé de les acheter déjà monter sur une plaque comme le montre la figure 8 avec préface de raccordement [11].

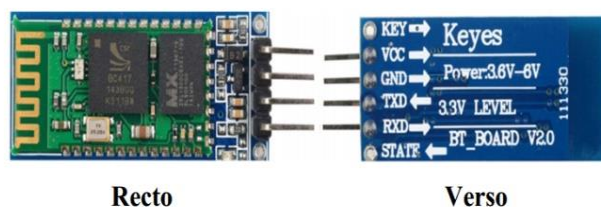


Fig. 8- Raccordements Bluetooth

Le module communique en mode série avec la carte Arduino, Pour cela il faut utiliser la librairie Softserial.

Vcc : Alimentation du module (3.3V a 5V).

GND : GND la masse.

TXD : Interface série, terminal de transmission.

RXD : Interface série, terminal de réception.

KEY, STATE : Sont les deux pins (2-17) entrées/sortie du module HC-05.

2.3 Capteurs

Un capteur est un dispositif convertissant une grandeur physique analogique (pression, température, déplacement, débit...), en une grandeur normée, généralement électrique, qui peut être interprétée par un dispositif de contrôle commande (courant électrique, radiation lumineuse, radiofréquence,...)[10].

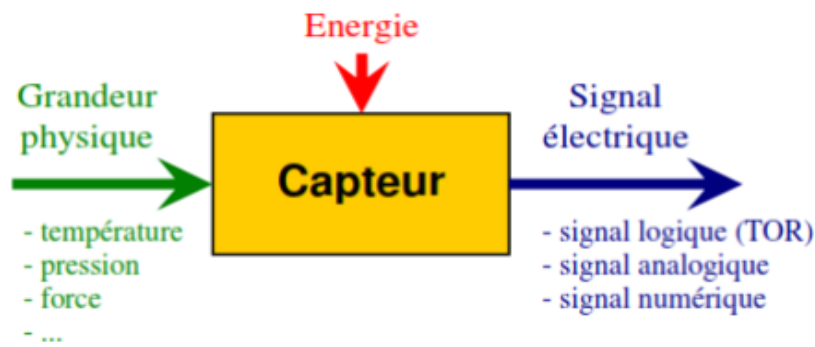


Fig. 9- Chaîne d'action d'un capteur.

2.3.1 Caractéristiques des capteurs

Les capteurs définis par les caractéristiques suivantes[12]

- **Etendue de mesure** : valeurs extrêmes pouvant être mesurée par le capteur.
- **Résolution** : plus petite variation de grandeur mesurable par le capteur.
- **Sensibilité** : variation du signal de sortie par rapport à la variation du signal d'entrée.
- **Précision** : aptitude du capteur à donner une mesure proche de la valeur vraie.
- **Rapidité** : temps de réaction du capteur. La rapidité est liée à la bande passante.
- **Linéarité** : représente l'écart de sensibilité sur l'étendue de mesure.

2.3.2 Classifications des capteurs

- le mesurande qu'il traduit (position, température, pression, etc.).
- le rôle dans le processus industriel (contrôle de produit fini, sécurité, etc.).
- le signal qu'il fournit en sortie qui peut être numérique, analogique, logique ou digital
- leur principe de traduction du mesurande (capteur résistif, piézoélectrique, etc.).
- leur principe de fonctionnement (capteur actif ou passif).

Toutes les classifications permettent d'avoir une vue d'ensemble des capteurs et bien sur aucune des méthodes de classification n'est meilleure que l'autre car toutes présentent des avantages et des inconvénients[10].

2.3.3 Capteur Ultrason HC-SR04

HC-SR04 utilise les ultrasons pour déterminer la distance d'un objet. Il offre une plage de détection sans contact, avec des mesures de haute précision et stables. Son fonctionnement n'est pas influencé par la lumière du soleil ou des matériaux sombres.



Fig. 10- Capteur ultrason HC-SR04.

Il contient quatre pattes de connexion

- **Vcc** : alimentation +5V (DC).
- **Trig** : entrée de déclenchement de la mesure (trigger input).
- **Echo** : sortie de mesure donnée en écho (Echo output).
- **GND** : masse de l'alimentation.

2.3.3.1 Caractéristiques du HC-SR04

Le HC-SR04 présente les caractéristiques évoquées sur le tableau suivant :

Paramètre	Valeur
Nombre de pattes	04
Tension d'alimentation min	4.5 V
Tension d'alimentation max	5.5 V
Courant de fonctionnement min	10 mA
Courant de fonctionnement max	20 mA
Fréquence ultrason	40 KHz
Dimensions	45mm x 20mm x 1 5mm
Plage de mesure	2cm à 400 cm
Résolution de la mesure	0.3 cm
Angle de mesure efficace	15°
Largeur d'impulsion sur l'entrée de déclenchement	10 µs trigger input pulse width

Tableau 3- Caractéristique de HC-SR04.

2.3.3.2 Fonctionnement du HC-SR04

- Pour déclencher une mesure, il faut présenter une impulsion niveau haut (5v), d'au moins 10 μ s sur l'entrée trig.
- Le capteur émet une série de 8 impulsions ultrasoniques à 40 kHz, puis il attend le signal réfléchi.
- Lorsque celle-ci est détectée, il envoie un signal de niveau haut sur la sortie Echo, dont la durée est proportionnelle à la distance mesurée.

La distance parcourue par un son se calcule en multipliant la vitesse du son par le temps de propagation soit:

$$D = V \times T \text{ (Distance = Vitesse x Temps)}$$

$$D = 34000 \text{ cm} / 1\,000\,000 \mu\text{S} \cdot 10 \mu\text{S} \cdot \text{Valeur} / 2$$

Le chronogramme du HC-SR04:

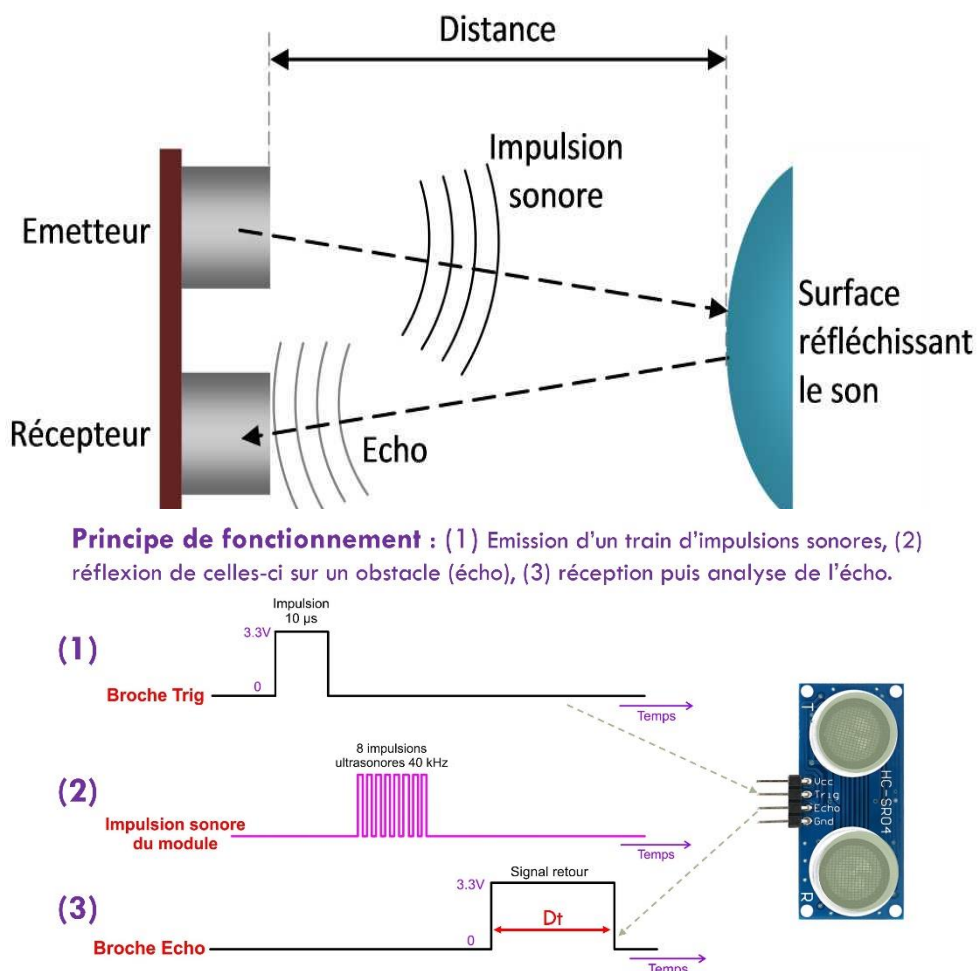


Fig. 11- Fonctionnement de HC-SR04..

2.3.4 Capteur P.I.R

Permet de détecter des formes mobiles qui émettent un rayonnement infrarouge (humains, animaux, robots mobiles ...) Il est utilisé dans divers projets tels que la sécurité, la domotique, ... On en trouve à bas coût, et il est très simple d'utilisation.

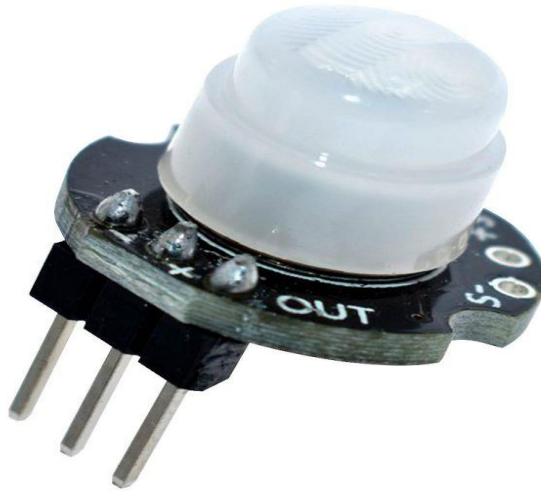


Fig. 12-Capteur P.I.R

Il contient trois pattes de connexion

- **Vcc** alimentation de 3.3V-15V (DC)
- **Out** haut niveau, H = 3.3V, L = 0V.
- **Gnd** masse de l'alimentation.

2.3.5 Encodeur

Un capteur à interruption optique sert à mesurer la vitesse d'un moteur, ainsi que le déplacement mécanique d'un mécanisme motorisé, quel que soit son type (linéaire, rotatif, etc.).

La figure ci-dessous illustre un exemple de montage d'un capteur à codeur optique, où la roue codeuse est reliée à un petit moteur à courant continu pour mesurer sa vitesse.

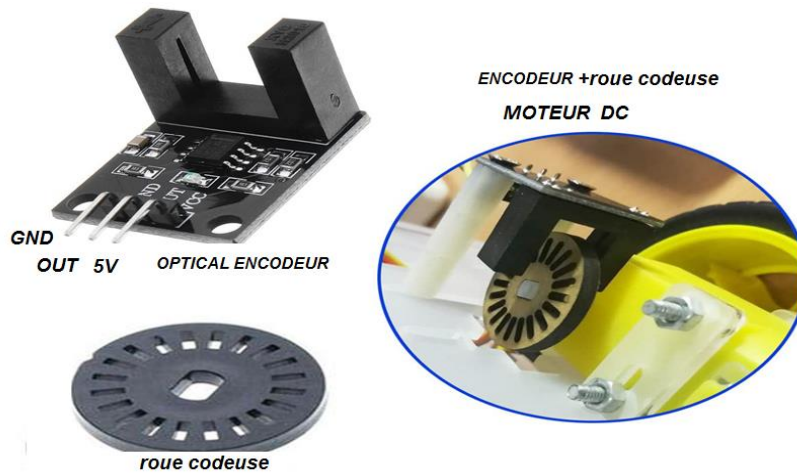


Fig. 13- Encodeur optique

2.4 Moteur

Moteur à courant continu (CC) couplé à une boîte d'engrenages et équipé d'un double axe de diamètre 5 mm. Il est particulièrement adapté à la réalisation de robots mobiles et de projets basés sur Arduino. Ce moteur DC fonctionne sous une tension comprise entre 3 V et 6 V et est généralement associé à une roue de 65 mm de diamètre. Sa vitesse à vide sous 5 V est comprise entre 150 et 200 tr/min.



Fig. 14- Moteur à courant continu (CC)

2.5 Pilote de moteur à double pont en H L298N

Le module L298N est un pilote de moteur basé sur un double pont en H permettant de commander simultanément deux moteurs à courant continu ou un moteur pas à pas. Il sert d'interface entre le microcontrôleur et les moteurs en fournissant le courant nécessaire à leur fonctionnement.

Le L298N permet de contrôler le sens de rotation ainsi que la vitesse des moteurs grâce à la technique de modulation de largeur d'impulsion (PWM). Il accepte une tension d'alimentation des moteurs comprise généralement entre 5 V et 35 V et peut fournir un courant maximal de 2 A par canal (avec un refroidissement approprié).

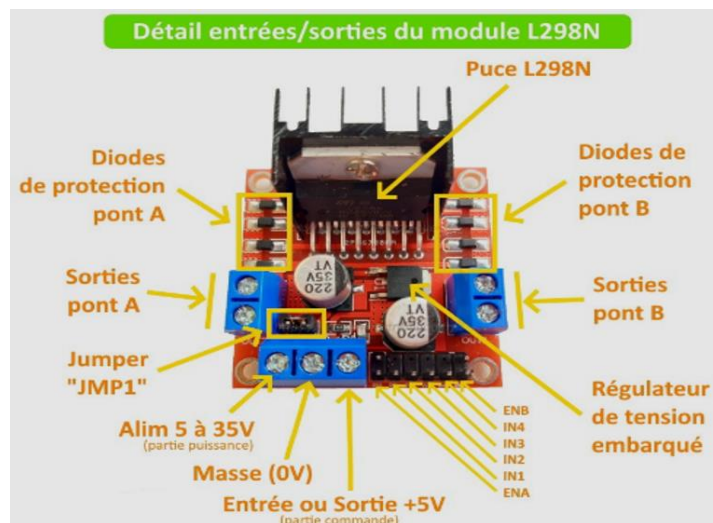


Fig. 15- Pilote de moteur H L298N

2.6 Ecran LCD

2.6.1 Définition et rôle des écrans LCD

LCD signifie « Écran à cristaux liquides ». Ils servent d'interface visuelle entre un système et un utilisateur.

Leur commande est assurée par un microcontrôleur. Ils permettent l'affichage aisé de messages ainsi que la création de caractères personnalisés.

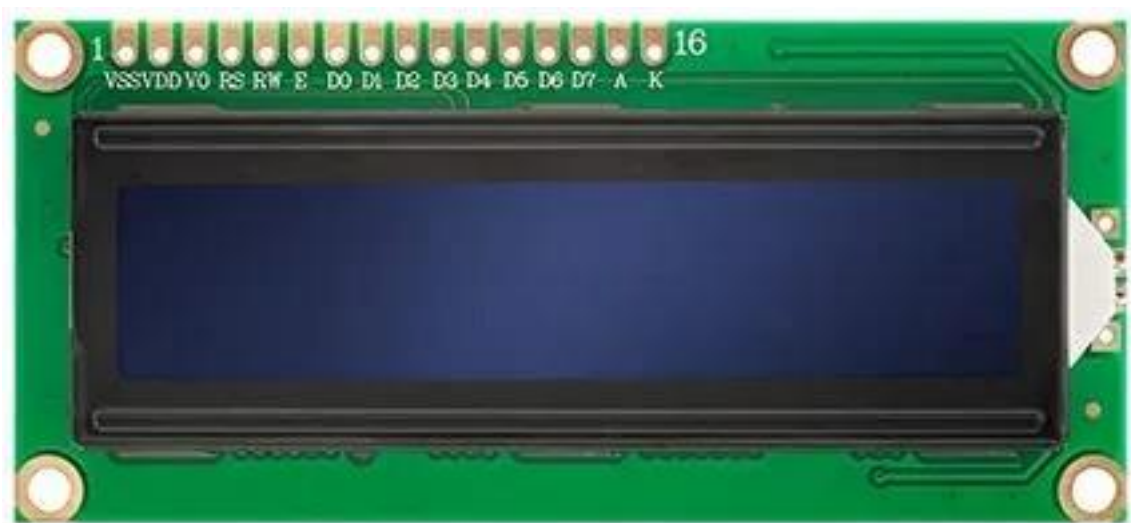


Fig. 16- Ecran LCD

L'afficheur dispose d'un rétro-éclairage LED et peut afficher deux lignes de 16 caractères. Chaque caractère est un rectangle de pixel. Il est possible de contrôler chaque pixel de chaque rectangle pour créer des caractères spécifiques.

L'écran s'appelle 16×2 parce qu'il possède 16 colonnes et 2 lignes. Il y a beaucoup d'autres combinaisons d'écran possible comme 8×1, 10×2, 16×1, mais le 16×2 est le plus connu.

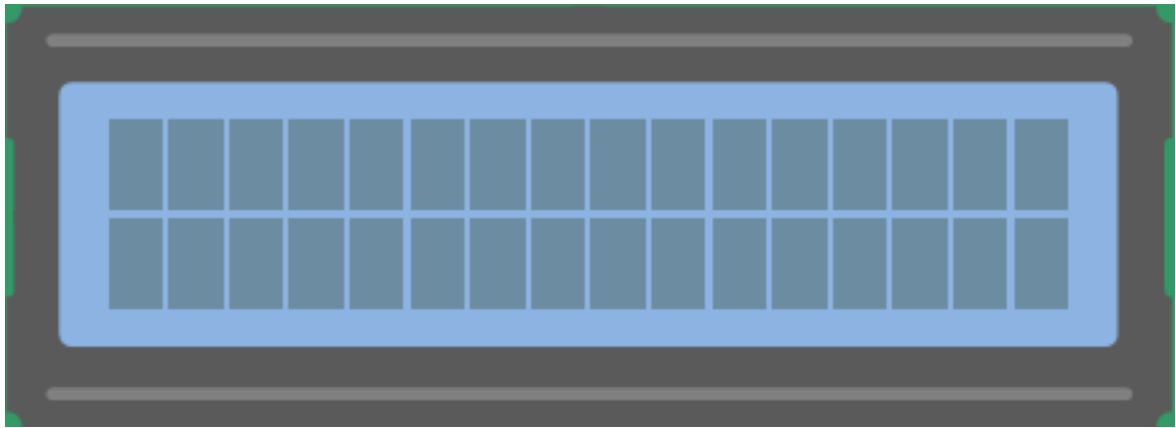


Fig. 17- Rectangles de pixel

2.6.2 I2C LCD Adapter

L'adaptateur I2C pour écran LCD est basé sur le circuit intégré **PCF8574**, qui permet de convertir les données transmises via le protocole **I2C** par la carte Arduino en signaux parallèles nécessaires au fonctionnement de l'écran LCD. Cet adaptateur intègre un **potentiomètre ajustable (trim-pot)** permettant de régler le contraste de l'affichage, ainsi qu'un **cavalier (jumper)** destiné à alimenter le rétroéclairage de l'écran. En retirant ce cavalier et en appliquant une tension externe sur la broche marquée **LED**, il est possible de contrôler et d'ajuster l'intensité lumineuse du rétroéclairage selon les besoins de l'application.

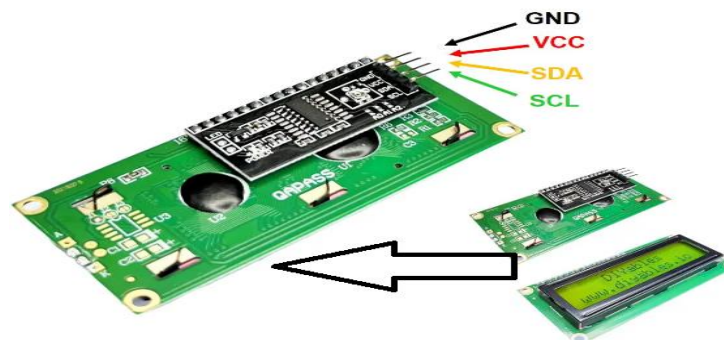


Fig. 18- Adaptateur I2C pour écran LCD.

2.7 Autres composants utilisés

2.7.1 LEDs

La diode électroluminescente (LED) émet de la lumière. Elle est polarisée : la patte '+' s'appelle anode est la plus longue, l'autre patte est la patte '-' cathode. Les broches numériques de l'Arduino, lorsqu'elles sont haut (high), fournissent une tension de 5V et 0V lorsqu'elles sont bas (low).



Fig. 19- La diode électroluminescente.

2.7.2 Buzzer

Le transducteur piézo-électrique est un composant réversible : il peut aussi bien être utilisé en capteur de chocs ou de vibrations qu'en actionneur pouvant émettre des sons stridents parfois modulables.

2.7.3 Batterie

Une batterie est un dispositif qui stocke de l'énergie sous forme chimique pour la restituer sous forme d'électricité. Elle se compose d'accumulateurs rechargeables. Ses usages incluent l'alimentation de véhicules, appareils électroniques et systèmes, La figure ci-dessous illustre un exemple de 9Volt avec adaptateur.



Fig. 20- Batterie avec Adapter

2.8 Logiciel Arduino IDE

Le logiciel Arduino IDE est compatible avec les systèmes d'exploitation Windows, Mac OS et Linux. Il constitue l'environnement de développement utilisé pour créer, compiler, tester et

téléverser les programmes vers la carte Arduino. Son interface, simple et conviviale, est organisée autour de quatre blocs principaux.

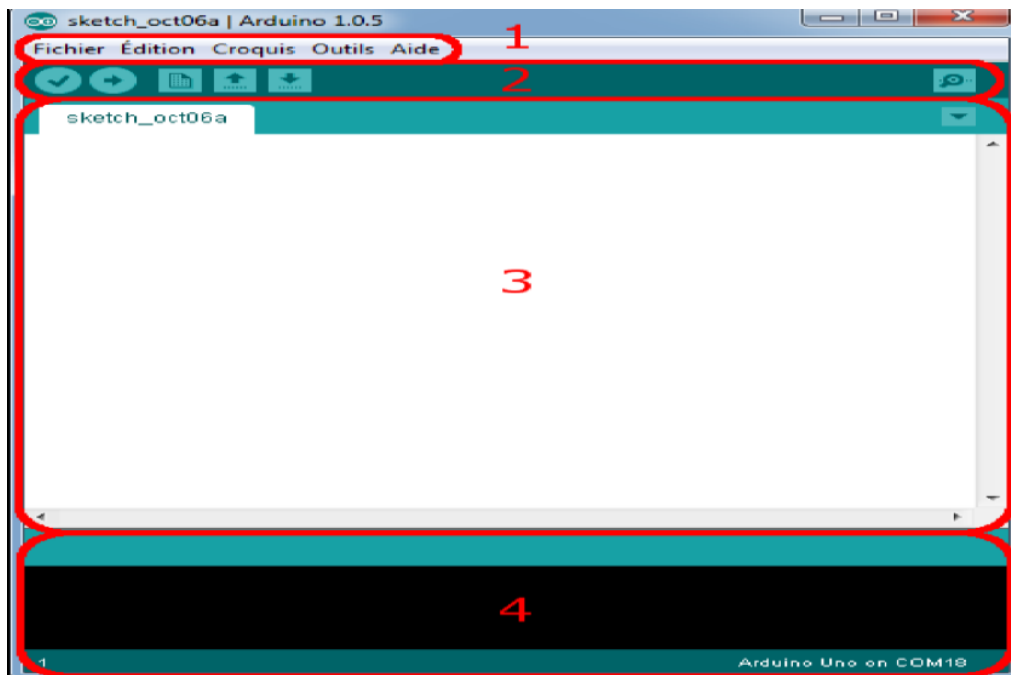


Fig. 21- Interface du logiciel IDE.

1- Menu : Les différents éléments du menu permettent de créer de nouveaux sketches (programmes), de les sauvegarder, et de gérer les préférences du logiciel et les paramètres de communication avec votre carte Arduino. Le menu comprend :

- **Fichier :** pour créer, sauvegarder en spécifiant la destination, et d'appeler un programme.
- **Édition :** Pour couper, copier, coller, supprimer, sélectionner,...etc.
- **Croquis :** regroupe les fichiers réalisés.
- **Outils :** pour spécifier le type de la carte, le port série, formater, recharger et réparer l'encodage, graver la séquence d'initialisation, de la carte branchée sur l'ordinateur.

2- Boutons

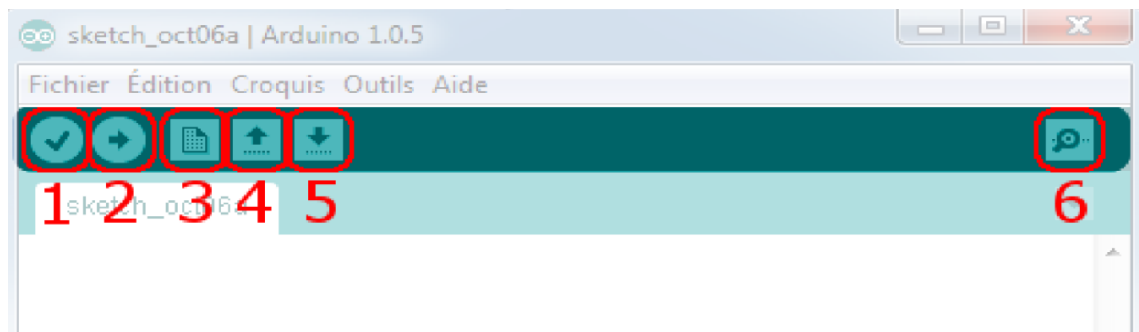


Fig. 22- Boutons d'action

- **Bouton 1** : Ce bouton permet de vérifier le programme, il actionne un module qui cherche les erreurs dans le programme
- **Bouton 2** : Charge (téléverse) le programme dans la carte Arduino
- **Bouton 3** : Cree un nouveau fichier.
- **Bouton 4** : Ouvre un fichier.
- **Bouton 5** : Enregistre le fichier.
- **Bouton 6** : Ouvre le moniteur série.

NOTE : Le moniteur série est un outil de communication entre le pc et la carte Arduino qui sont connecté via le port USB, utilisé généralement pour tester le fonctionnement des programmes.

3- Fenêtre de Programmation

C'est l'éditeur où s'écrit le programme, chaque logiciel obéit à quelques notions pour pouvoir bien structurer le programme à fin de le compiler et éviter les erreurs de syntaxe et autres.

4- Barre des erreurs

La barre des erreurs affiche les erreurs faites au cours du programme, comme l'oubli d'un pointvirgule, le manque d'une accolade ou toute autre erreur dans les instructions.

2.8.1 Structure de programmation d'un sketch Arduino

La structure d'un sketch Arduino se compose de **trois sections principales**, comme expliqué ci-dessous:

2.8.1.1 Section de déclaration

Cette partie du programme est consacrée à la déclaration des variables, des constantes et des broches d'entrée/sortie, ainsi qu'à l'inclusion des bibliothèques requises sous forme de fichiers d'en-tête C++ (*header files*).

2.8.1.2 Section d'initialisation (Setup)

Cette partie du programme contient la fonction **setup()**, chargée d'effectuer les opérations d'initialisation, telles que la configuration des broches d'entrée/sortie, l'initialisation de la communication série et des bibliothèques utilisées. Cette fonction est exécutée une seule fois au démarrage ou après chaque réinitialisation de la carte Arduino.

2.8.1.3 Section de boucle (Loop)

Cette partie renferme le programme principal de l'application. La fonction **loop()** s'exécute en boucle de façon continue et indéfinie après l'exécution de la fonction **setup()**, permettant ainsi la surveillance des entrées, le traitement des données et la commande des sorties en temps réel.

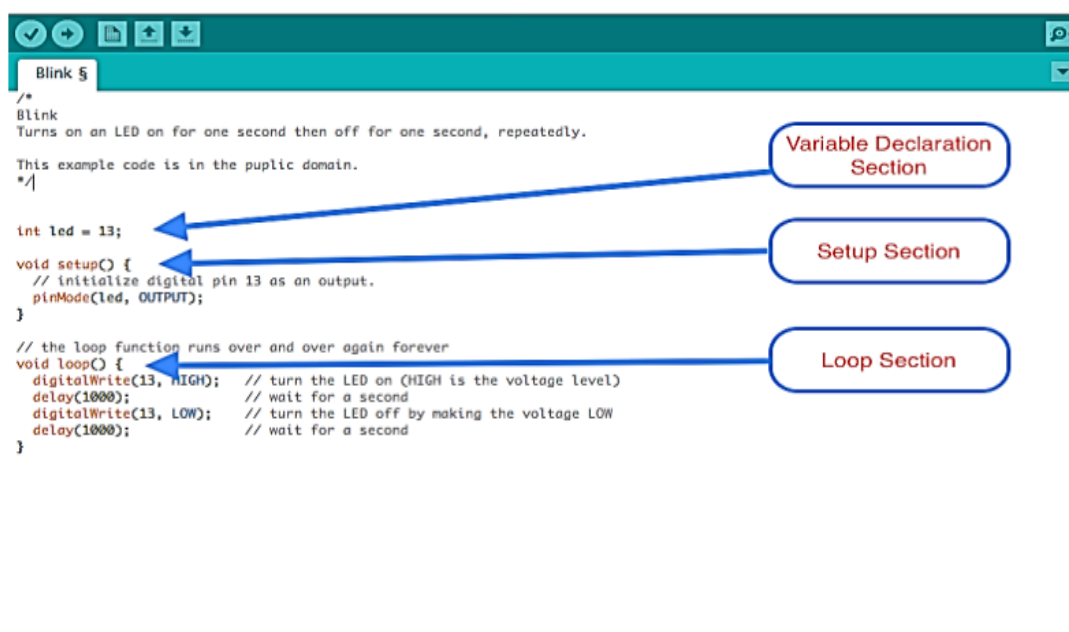


Fig. 23- Arduino Program structure.

2.8.2 Sélection de la carte Arduino

Avant de compiler ou de téléverser un programme, il est nécessaire de sélectionner la carte Arduino appropriée dans l'environnement de développement. Pour ce faire, ouvrez le menu **Tools (Outils)** de l'IDE Arduino, puis choisissez **Board (Carte)**. Une liste des cartes compatibles s'affiche alors. Sélectionnez **Arduino Mega 2560** afin que l'IDE configure correctement les paramètres de compilation et de téléversement en fonction des caractéristiques matérielles de cette carte.

2.9 Choix de la carte

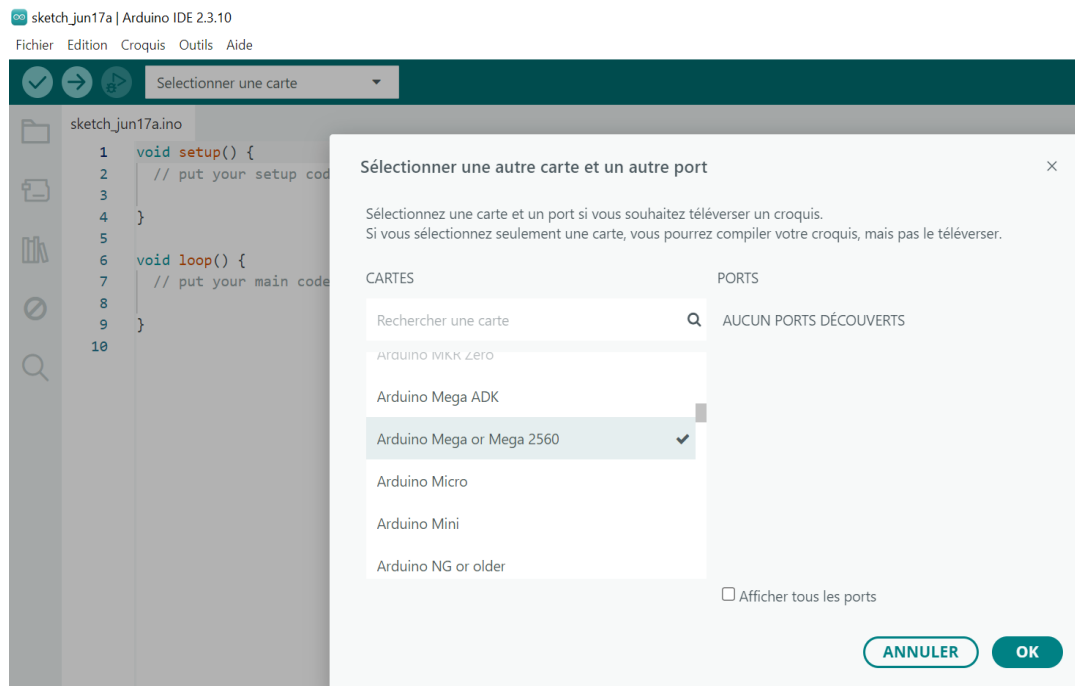


Fig. 24- Sélection de la carte Arduino Mega 2560.

2.10 PROTEUS (ISIS)

Proteus est considéré comme l'un des logiciels de conception de circuits électroniques les plus performants et complets du marché, grâce à ses capacités de simulation et d'analyse, ainsi qu'à sa compatibilité avec une large gamme de microcontrôleurs de différentes familles. Nous commencerons par décrire l'interface ISIS, le module de simulation, et la manière d'y dessiner des schémas. Cette interface est essentielle à la simulation, car elle simplifie et accélère le processus de création des schémas. Elle offre plusieurs options pour une efficacité optimale, comme la division des circuits en plusieurs feuilles de calcul et l'utilisation de sous-circuits. Proteus propose diverses options de simulation, notamment la simulation interactive en temps réel, qui permet de suivre le fonctionnement du circuit directement dans le schéma, grâce à différentes animations, comme s'il s'agissait d'un fonctionnement réel.

Ceci est rendu possible par l'utilisation de composants interactifs: bouton-poussoir, afficheurs, claviers, interrupteurs, LED, capteurs, etc.



Fig. 25- Icône ISIS

En appuyant sur l'icone précédente l'écran décrit dans le guide visuel suivant s'affiche.

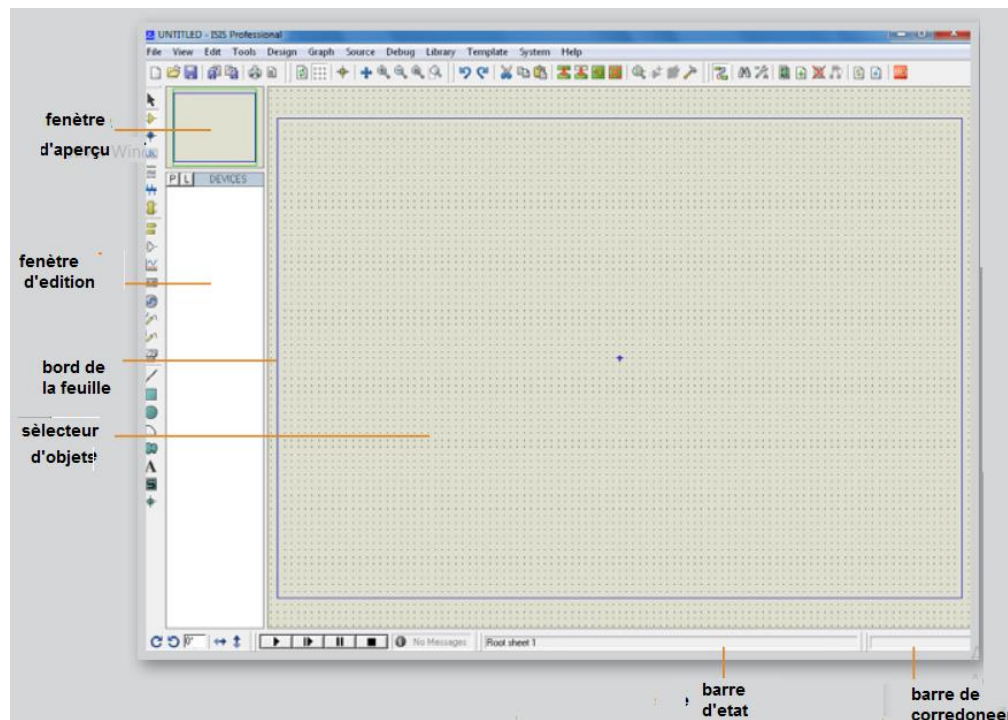


Fig. 26- Interface du Proteus

2.11 Conclusion

En conclusion, ce chapitre a été consacré à l'étude et à la conception du système de commande du véhicule télécommandé. Nous avons présenté les différents composants matériels utilisés, notamment la carte Arduino Mega 2560, le module Bluetooth HC-05, les capteurs, le pilote de moteur L298N, l'écran LCD ainsi que les différents éléments de commande et d'alimentation. Une analyse détaillée de leurs caractéristiques et de leur rôle dans le système a permis de justifier leur choix en fonction des exigences du projet.

Par ailleurs, nous avons étudié l'environnement logiciel utilisé pour le développement et la simulation, à savoir Arduino IDE et Proteus. Ces outils offrent une solution efficace pour la programmation, la simulation et la validation du fonctionnement du système avant sa réalisation pratique.

Cette phase de conception a permis d'établir l'architecture générale du véhicule télécommandé et de définir les interactions entre les différents modules matériels et logiciels. Les éléments présentés dans ce chapitre constituent ainsi la base nécessaire à la réalisation pratique du prototype, qui sera détaillée dans le chapitre suivant.

Chapitre 3 : Réalisation pratique et tests du véhicule télécommandé.

Introduction

La commande à distance, ou téléopération, consiste à piloter un système mobile (le véhicule) depuis une interface distante, sans liaison physique directe. Ce projet s'inscrit dans le domaine de la mécatronique, une discipline qui combine trois champs complémentaires :

- L'électronique : capteurs, contrôleurs, transmission de données.
- L'informatique : algorithmes de traitement et de commande.
- La mécanique : propulsion, direction et châssis.

Ce chapitre présente la mise en œuvre concrète du système décrit dans les chapitres précédents : l'environnement logiciel utilisé pour la programmation et la simulation, le matériel assemblé, le programme de commande embarqué sur la carte Arduino, puis la réalisation physique du prototype et les tests effectués.

3.1 Environnement logiciel

3.1.1 Environnement de programmation Arduino

Le code Arduino est écrit en langage C++, enrichi de fonctions et de méthodes spécifiques propres à la plateforme. C'est un langage de programmation relativement simple à lire et à apprendre. Lorsqu'un fichier « sketch » (nom donné aux fichiers de code Arduino) est créé, il est ensuite traité et compilé en code machine au format .hex.

La dernière version de l'environnement de développement intégré (IDE) peut être téléchargée gratuitement à l'adresse suivante :

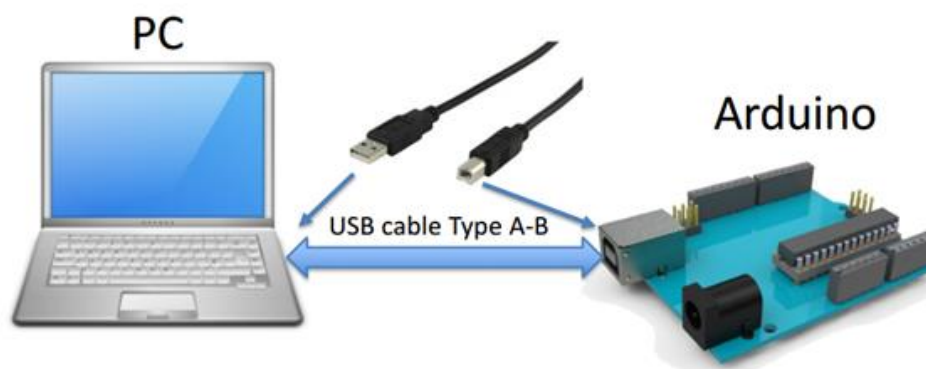


Fig. 27- Connexion entre la carte Arduino et le PC.

3.1.2 Fonctionnement de l'IDE Arduino

L'environnement de développement intégré (IDE) Arduino est le principal éditeur de texte utilisé pour la programmation. C'est là que le code est saisi avant d'être téléversé sur la carte à programmer ; le code Arduino est appelé « croquis » (sketch).

- **Environnement** : présentation de l'IDE Arduino et des spécificités du langage C/C++ pour l'embarqué.
- **Architecture du code** : description des phases d'initialisation, de gestion des interruptions et de décodage des commandes.

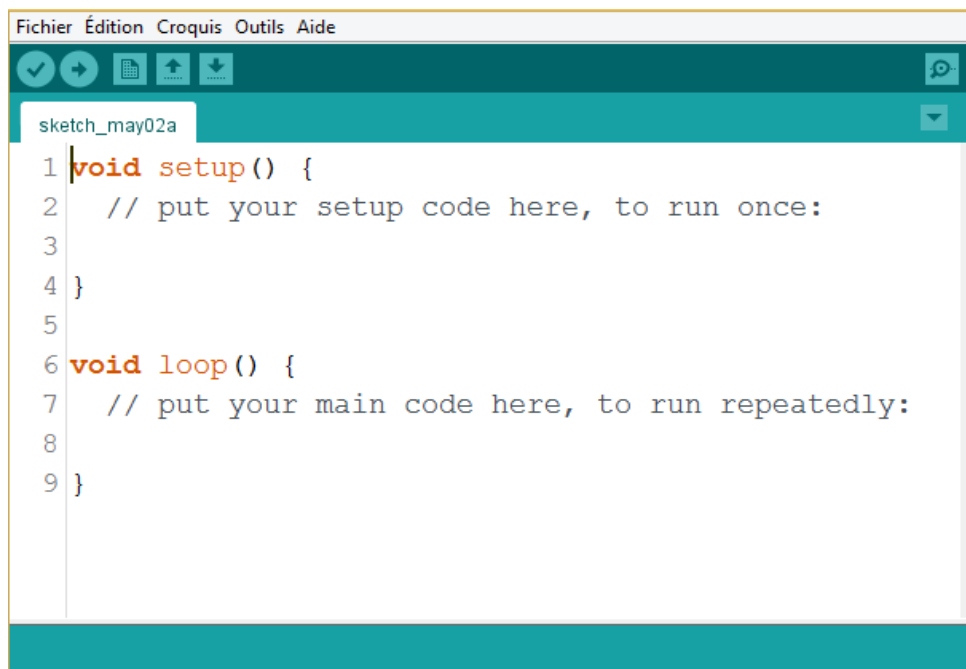


Fig. 28- Fenêtre de l'IDE Arduino à l'ouverture

- Connecter la carte Arduino UNO à l'ordinateur via un câble USB.
- Une fois la carte connectée, vérifier que l'IDE a bien sélectionné le bon type de carte. Dans notre cas, nous utilisons une **Arduino Uno** : menu **Outils** → **Type de carte** → **Arduino/Genuino Mega2560**.
- Sélectionner ensuite le port série auquel la carte est connectée : menu **Outils** → **Port**.

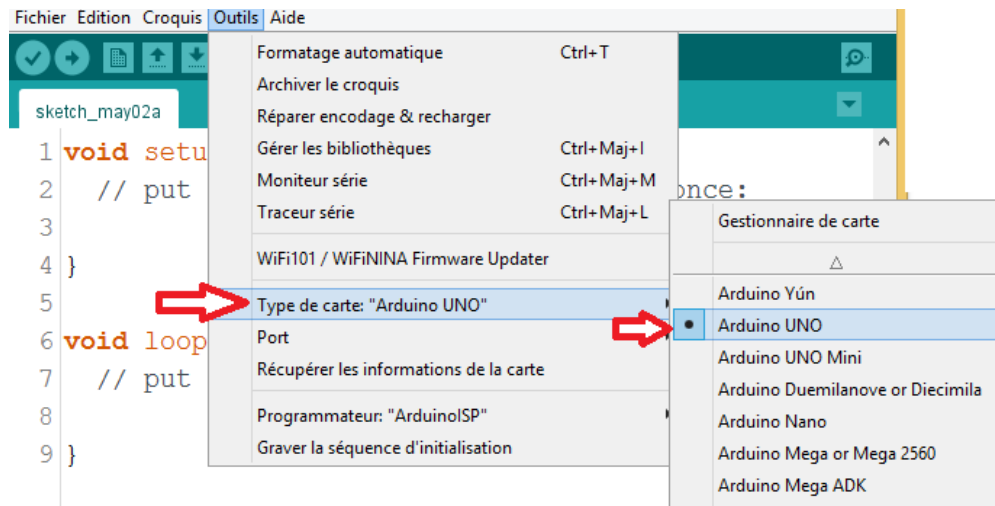


Fig. 29- Sélection du type de carte dans l'IDE Arduino.

3.1.2.1 Télversement d'un programme Arduino

Pour illustrer le téléversement d'un code sur la carte Arduino, un exemple simple de configuration Bluetooth est présenté ci-dessous. Après rédaction du code et vérification par compilation, le programme est injecté dans la carte Arduino, puis testé en activant le Bluetooth d'un smartphone à l'aide d'une application dédiée téléchargeable gratuitement.

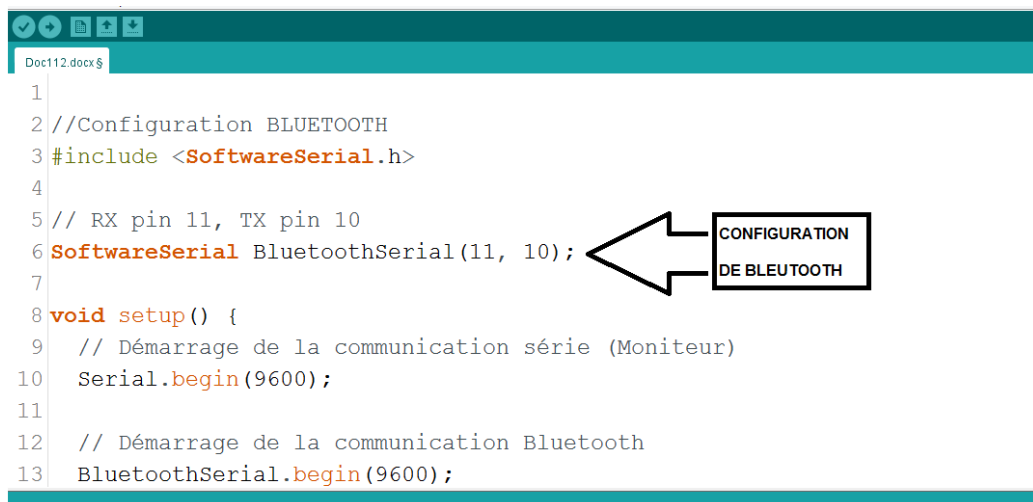


Fig. 30- Exemple de code de configuration Bluetooth.

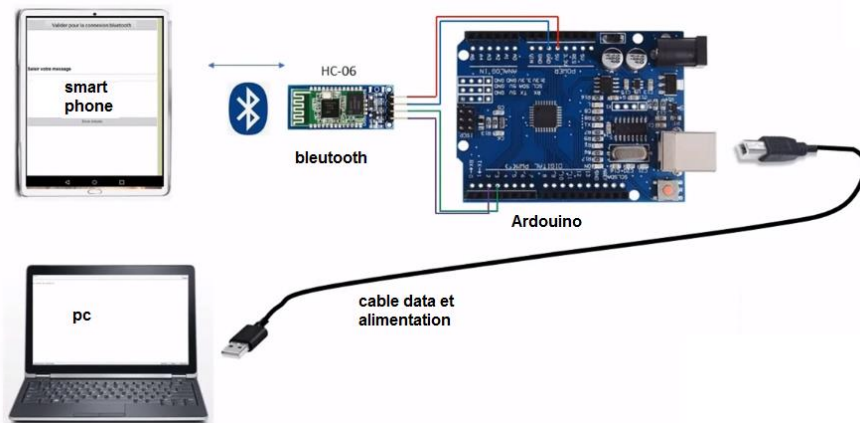


Fig. 31- Combinaison PC, Arduino, module Bluetooth et smartphone.

3.1.3 Simulation sous Proteus

Après rédaction du code Arduino et compilation en langage machine (.hex), le logiciel Proteus permet de simuler, concevoir, tester et planifier rapidement et facilement des cartes de circuits imprimés, de manière professionnelle.

3.1.3.1 Schéma synoptique

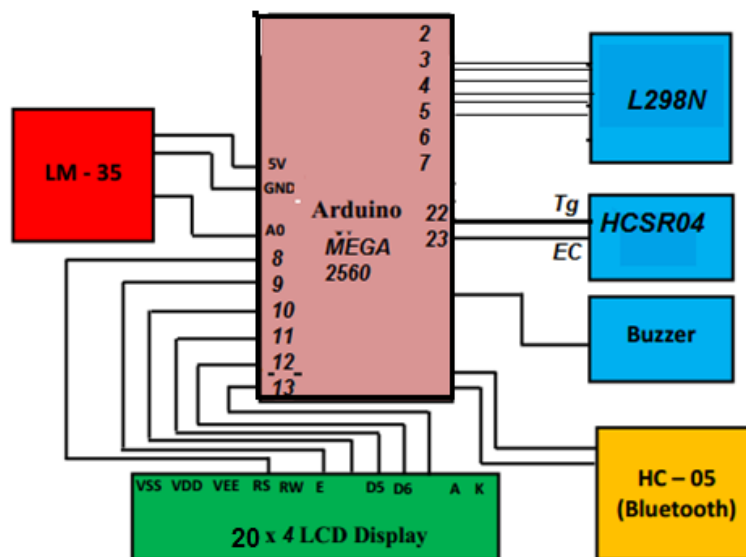


Fig. 32- Schéma synoptique du système

Pour réaliser la simulation du circuit électronique, les composants doivent être sélectionnés conformément à la liste des composants définie au préalable, puis connectés entre eux afin de

reconstituer le circuit électronique complet. La simulation est ensuite effectuée à partir de ce schéma, avec affichage sur l'écran LCD de la température, de la vitesse, de la direction et de l'état du buzzer.

3.1.3.2 Schéma électronique du circuit

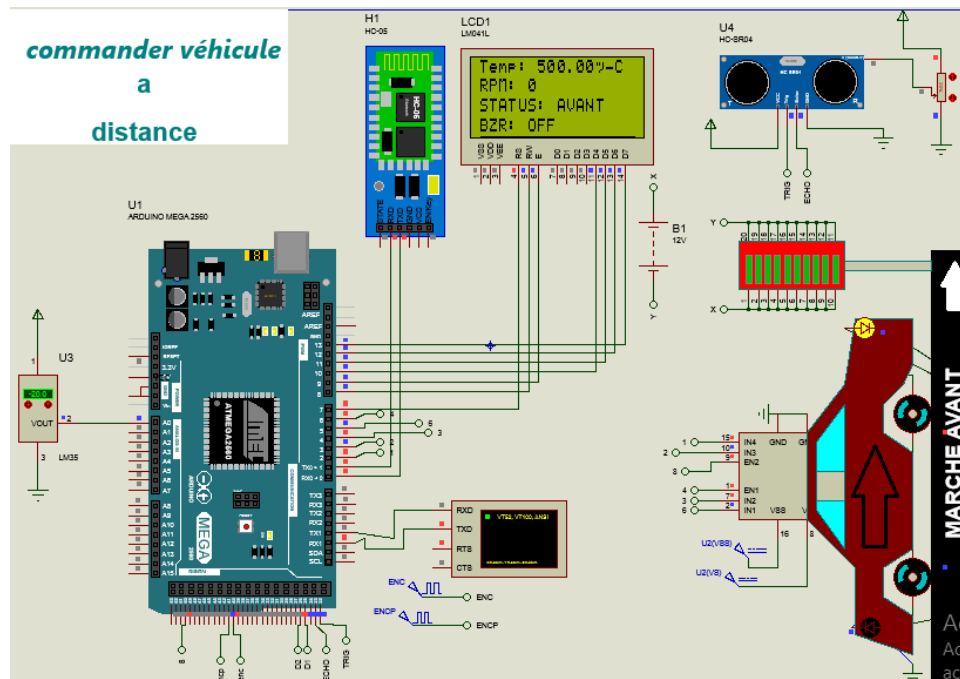


Fig. 33- Simulation du véhicule en marche avant.

Le même principe est appliqué pour simuler la conduite du véhicule en marche arrière

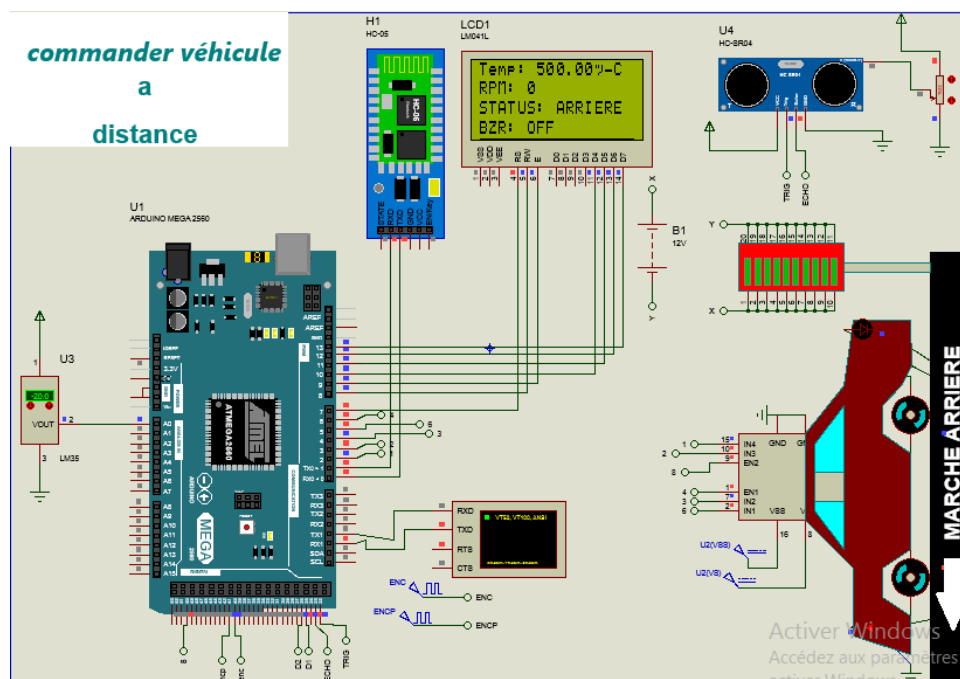


Fig. 34- Simulation du véhicule en marche arrière.

Remarque : il est possible de visualiser la vitesse sous Proteus à l'aide de la bibliothèque de moteur-encodeur, en alimentant le circuit en 12 V ou en créant des impulsions avec un générateur d'impulsions.

3.2 Matériel utilisé

Avant de commencer la partie pratique, les outils suivants sont nécessaires : un multimètre, un fer à souder, ainsi qu'un kit Arduino regroupant l'ensemble des composants nécessaires pour apprendre les bases et réaliser des projets.



Fig. 35- Kit de composants Arduino.

multimetre



fer a souder



Fig. 36- Multimètre et fer à souder.

3.2.1 Liste des composants

Le tableau suivant récapitule l'ensemble des éléments utilisés pour la réalisation du véhicule télécommandé :

#	Composant	Quantité
1	Carte Arduino Méga 2560	1
2	Module Bluetooth HC-05	1
3	Capteur ultrason HC-SR04	1
4	Capteur de mouvement PIR (SR602)	1
5	Capteur de température LM35	1
6	Module driver moteur (pont en H L298N)	1
7	Écran LCD I2C 20x4	1
8	Kit châssis de véhicule robot, 4 moteurs CC et 4 roues	1 kit
9	Encodeur de vitesse	1
10	LEDs de signalisation	2
11	Buzzer	1
12	Piles 1,5 V avec boîtier	4
13	Pile 9 V	1
14	Fils électriques mâle/femelle	Plusieurs
15	Fer à souder léger et fil d'étain	1
16	Câble USB-A / USB-B (Arduino Méga)	1

17	Smartphone sous Android	1
----	-------------------------	---

Tableau 4-Matériel utilisé.**Fig. 37- Câble USB pour Arduino Méga**

3.2.2 Application de commande sur smartphone

Le véhicule est contrôlé à distance depuis un smartphone, soit via Bluetooth (module HC-05/HC-06), soit via Wi-Fi. Que l'on envoie des commandes simples via une application Bluetooth

**Fig. 38- Kit de châssis du véhicule robot (roues et moteurs CC).**

ou via une application sur le réseau local, les deux solutions offrent une configuration simple et abordable : sans interface complexe, elles reposent sur une communication série fiable et un code robuste, qui constituent une excellente base pour un contrôle intelligent et sans fil.

Avant de tester le projet, il faut installer sur le smartphone Android l'application « **Arduino Bluetooth RC Controller** », disponible gratuitement sur le Play Store. Caractéristiques de l'application utilisée :

- Licence : gratuite

- Version : 2.1
- Dernière mise à jour : 29 avril 2024
- Plateforme : Android (à partir d'Android 9.0)
- Langue : anglais
- Taille : 4,36 Mo

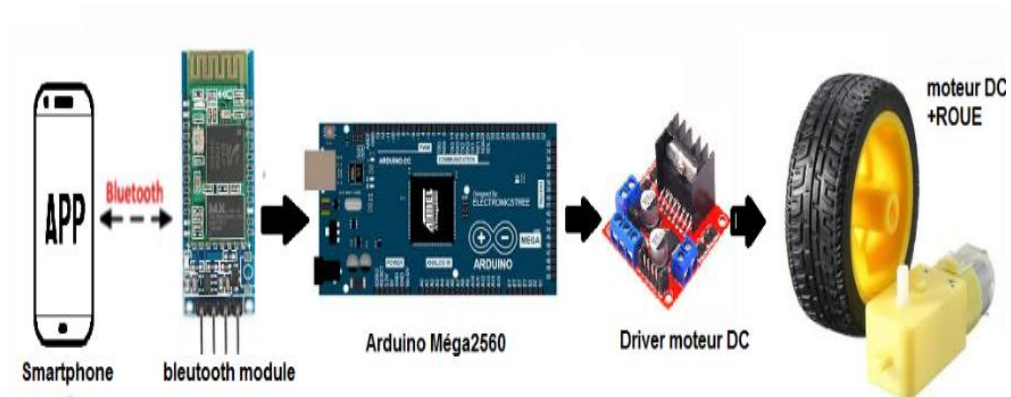


Fig. 39- Combinaison des différents éléments de contrôle à distance.

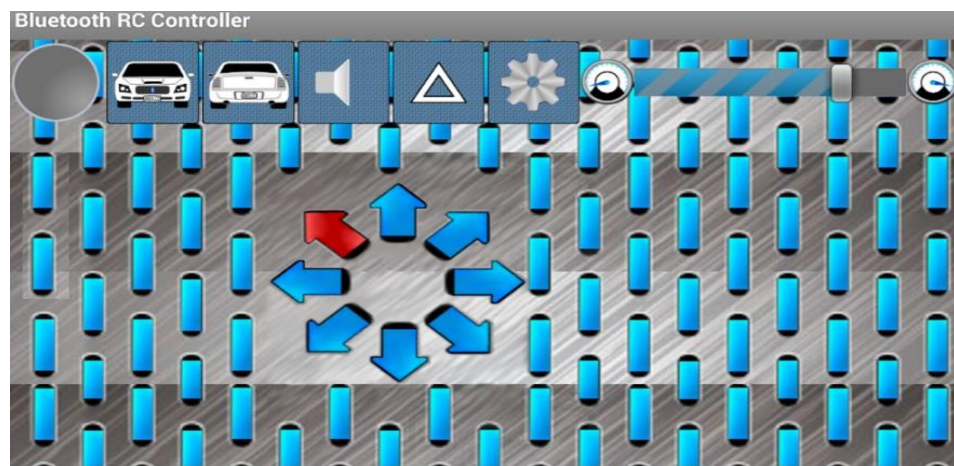


Fig. 40- Interface de l'application Bluetooth RC Controller.



Fig. 41- Utilisation de l'accéléromètre du smartphone pour détecter l'orientation.

Le modèle a été conçu conformément aux spécifications et aux instructions de câblage standard. Il peut se déplacer dans quatre directions (avant, arrière, droite, gauche) et a été conçu pour être contrôlé à distance par l'utilisateur via la technologie Bluetooth.



Fig. 42- Prototype du véhicule.

3.3 Réalisation et schéma de câblage

La réalisation d'un système de commande de véhicule à distance est un projet qui mêle électronique, programmation et réseaux. Le schéma ci-dessous présente la combinaison de l'ensemble des éléments utilisés pour le fonctionnement du véhicule, connectés à la carte Arduino Méga 2560 :

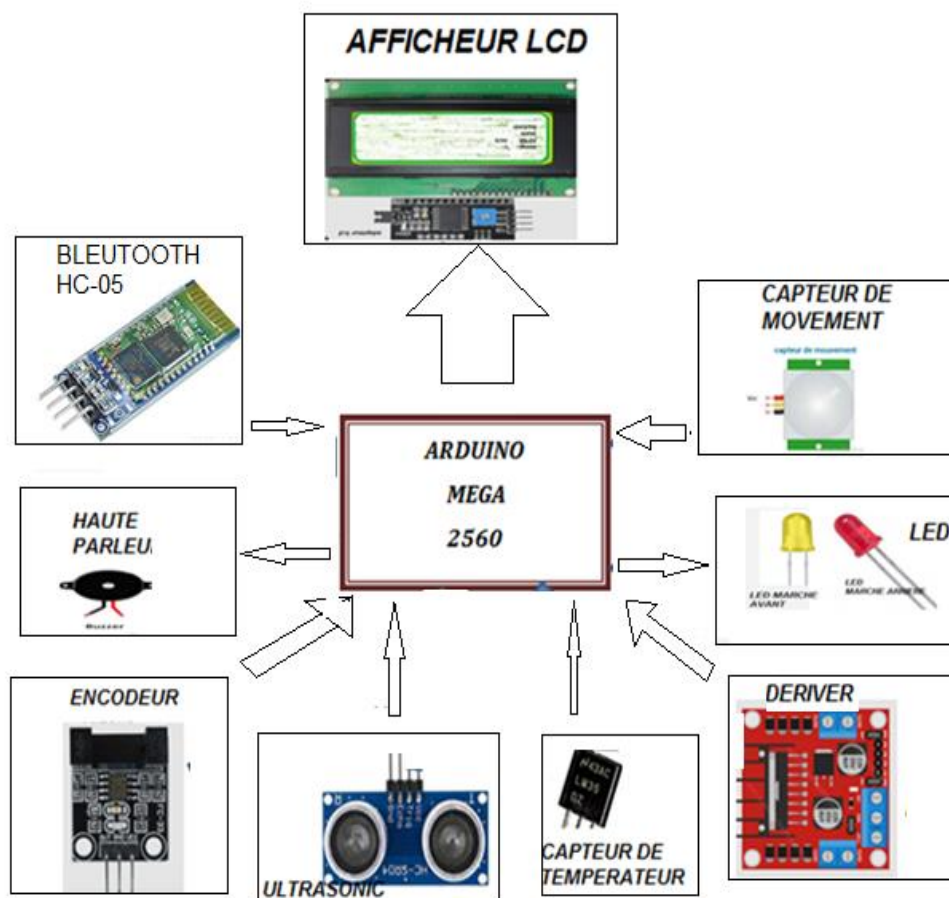


Fig. 43- Schéma physique des éléments connectés à l'Arduino Méga 2560

3.3.1 Fonctions principales du système

Fonction	Description
Avancer / Reculer	Déplacement du véhicule vers l'avant ou l'arrière
Tourner à gauche / à droite	Changement de direction du véhicule
Détection d'obstacle	Mesure de distance via le capteur HC-SR04
Détection de mouvement arrière	Sécurité au recul via le capteur PIR
Mesure de température	Lecture du capteur LM35

Calcul de vitesse	Comptage d'impulsions par encodeur
Alerte sonore	Activation du buzzer en cas d'obstacle
Affichage	Écran LCD 20x4 (I2C)
Communication	Liaison série Bluetooth (HC-05/HC-06)

Tableau 5-Fonctions principales de système.

3.3.2 Traitement et sorties

La carte Arduino Méga 2560 traite les données reçues, prend les décisions de mouvement, calcule la vitesse et gère les obstacles de sécurité. Les sorties pilotées par le programme sont les suivantes :

- Moteurs à courant continu (via le pont en H L298N)
- LEDs de signalisation
- Buzzer d'alerte
- Écran LCD 20x4 (I2C)

3.4 Programmation : code Arduino du récepteur

Cette section présente le programme complet exécuté par la carte Arduino Méga 2560, qui constitue le récepteur du système de téléopération. Ce programme contrôle un robot mobile équipé de quatre moteurs et de plusieurs capteurs de sécurité (ultrason et mouvement), d'un capteur de température, d'un encodeur pour mesurer la vitesse, et d'un écran LCD pour l'affichage en temps réel. Il est piloté à distance via une liaison série, généralement assurée par un module Bluetooth connecté à un smartphone.

3.4.1 Code source complet

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h>

// Configuration LCD (Adresse 0x27 ou 0x3F)

LiquidCrystal_I2C lcd(0x27, 20, 4);
```

// Broches Moteurs (L298N)

```
const int EN_1 = 8;

const int moteur1_A = 3;

const int moteur1_B = 4;

const int EN_2 = 7;

const int moteur2_A = 5;

const int moteur2_B = 6;

const int led1 = 40;

const int led2 = 33;
```

// Broches Capteurs

```
const int trigPin = 9;    // Ultrason

const int echoPin = 10;   // Ultrason

const int encoderPin = 2; // Encodeur (Pin Interruption sur Mega)

const int pirPin = 12;    // Capteur de mouvement SR602

const int tempPin = A0;   // Capteur LM35
```

// Variables

```
volatile int impulsions = 0;

unsigned long tempsPrecedent = 0;

float vitesse = 0;

float temperature = 0;

long duree;

int distance;

const int distanceSecurite = 15;
```

```
char commande;

bool mouvementArriere = false;

void compterImpulsion() {

    impulsions++;

}

void setup() {

    // Configuration Moteurs et LEDs

    pinMode(moteur1_A, OUTPUT);  pinMode(moteur1_B, OUTPUT);  pinMode(led1,
OUTPUT);

    pinMode(moteur2_A, OUTPUT);  pinMode(moteur2_B, OUTPUT);  pinMode(led2,
OUTPUT);

    pinMode(EN_1, OUTPUT); pinMode(EN_2, OUTPUT);

    // Configuration Capteurs

    pinMode(trigPin, OUTPUT); pinMode(echoPin, INPUT);

    pinMode(encoderPin, INPUT_PULLUP);

    pinMode(pirPin, INPUT);

    attachInterrupt(digitalPinToInterrupt(encoderPin), compterImpulsion, RISING);

    Serial.begin(9600);

    lcd.init();

    lcd.backlight();

    afficherStatus("SYSTEME OK", 0, 0, 0);

}
```

```
void loop() {  
  
    // 1. Calcul de la vitesse et température toutes les secondes  
  
    unsigned long tempsActuel = millis();  
  
    if ((tempsActuel - tempsPrecedent) >= 1000) {  
  
        // Calcul Vitesse  
  
        vitesse = (impulsions / 20.0) * 60.0;  
  
        impulsions = 0;  
  
        // Lecture Température LM35  
  
        int lectureAnalogique = analogRead(tempPin);  
  
        float millivolts = (lectureAnalogique / 1024.0) * 5000.0;  
  
        temperature = millivolts / 10.0; // 10 mV = 1°C  
  
        tempsPrecedent = tempsActuel;  
  
        afficherStatus(getDirectionName(commande), distance, vitesse, temperature);  
    }  
  
    // 2. Mesure Ultrason (Obstacle Avant)  
  
    digitalWrite(trigPin, LOW); delayMicroseconds(2);  
  
    digitalWrite(trigPin, HIGH); delayMicroseconds(10);  
  
    digitalWrite(trigPin, LOW);  
  
    duree = pulseIn(echoPin, HIGH);  
  
    distance = duree * 0.034 / 2;  
  
    // 3. Sécurité Mouvement Arrière
```

```
mouvementArriere = digitalRead(pirPin);

if (distance > 0 && distance < distanceSecurite) {

    stopper();

    afficherStatus("!! OBSTACLE !!", distance, 0, temperature);

} else if (commande == 'B' && mouvementArriere == HIGH) {

    stopper();

    afficherStatus("!! PIETON ARR !!", distance, 0, temperature);

} else {

    // Contrôle Smartphone

    if (Serial.available() > 0) {

        commande = Serial.read();

        executerCommande(commande);

    }

}

}
```

```
String getDirectionName(char cmd) {

    switch (cmd) {

        case 'F': return "AVANCER";

        case 'B': return "RECULER";

        case 'L': return "GAUCHE";

        case 'R': return "DROITE";

        case 'S': return "STOP";

        default: return "ATTENTE";

    }

}
```

```
}
```

```
void executerCommande(char cmd) {  
    switch (cmd) {  
        case 'F': avancer(); break;  
        case 'B': reculer(); break;  
        case 'L': tournerGauche(); break;  
        case 'R': tournerDroite(); break;  
        case 'S': stopper(); break;  
    }  
    afficherStatus(getDirectionName(cmd), distance, vitesse, temperature);  
}
```

```
// --- Fonctions de Mouvement ---
```

```
void avancer() {  
    digitalWrite(moteur1_A, HIGH); digitalWrite(moteur1_B, LOW);    digitalWrite(led1,  
LOW);  
    digitalWrite(moteur2_A, HIGH); digitalWrite(moteur2_B, LOW);    digitalWrite(led2,  
HIGH);  
    digitalWrite(EN_1, 150); digitalWrite(EN_2, 150);  
}
```

```
void reculer() {  
    digitalWrite(moteur1_A, LOW);    digitalWrite(moteur1_B, HIGH); digitalWrite(led1,  
HIGH);  
    digitalWrite(moteur2_A, LOW);    digitalWrite(moteur2_B, HIGH); digitalWrite(led2,  
LOW);
```

```
digitalWrite(EN_2, 150); digitalWrite(EN_1, 150);

}

void tournerGauche() {

digitalWrite(moteur1_A, LOW); digitalWrite(moteur1_B, HIGH);

digitalWrite(moteur2_A, LOW); digitalWrite(moteur2_B, LOW);

digitalWrite(EN_2, 100); digitalWrite(EN_1, 100);

}

void tournerDroite() {

digitalWrite(moteur1_A, LOW); digitalWrite(moteur1_B, LOW);

digitalWrite(moteur2_A, LOW); digitalWrite(moteur2_B, HIGH);

digitalWrite(EN_1, 100); digitalWrite(EN_2, 100);

}

void stopper() {

digitalWrite(moteur1_A, LOW); digitalWrite(moteur1_B, LOW); digitalWrite(led1,
LOW);

digitalWrite(moteur2_A, LOW); digitalWrite(moteur2_B, LOW); digitalWrite(led2,
LOW);

digitalWrite(EN_1, 0); digitalWrite(EN_2, 0);

}

// --- Affichage LCD 20x4 ---

void afficherStatus(String msg, int dist, float vit, float temp) {

lcd.setCursor(0, 0);

lcd.print("PROJET FIN D'ETUDE ");
```

```

lcd.setCursor(0, 1);

lcd.print(": "); lcd.print(msg); lcd.print("    ");

lcd.setCursor(0, 2);

lcd.print(" V:"); lcd.print((int)vit); lcd.print("RPM ");

lcd.print("T:"); lcd.print(temp, 1); lcd.print("C ");

lcd.setCursor(0, 3);

lcd.print("Obs: "); lcd.print(dist); lcd.print("cm ");

if (mouvementArriere) lcd.print(" !MOV!"); else lcd.print("    ");

}

```

3.4.2 Architecture matérielle (broches et composants)

Le début du code définit la configuration des composants connectés à la carte Arduino Méga 2560

- **Moteurs (pont en H L298N) :** contrôle de deux moteurs. Les broches moteur1_A/B et moteur2_A/B gèrent le sens de rotation. Les broches EN_1 et EN_2 (reliées à des broches PWM) contrôlent la vitesse des moteurs.
- **Capteur ultrason (HC-SR04) :** utilise trigPin (émission) et echoPin (réception) pour mesurer la distance des obstacles à l'avant.
- **Capteur de mouvement (PIR SR602) :** branché sur la broche pirPin (12) pour détecter une présence ou un piéton à l'arrière du robot.
- **Capteur de température (LM35) :** connecté sur la broche analogique A0.
- **Encodeur de vitesse :** connecté sur la broche 2, qui gère les interruptions sur Arduino, pour compter les rotations de la roue.
- **Écran LCD I2C (20x4) :** permet d'afficher l'état du système sur 4 lignes et 20 colonnes.
- **LEDs :** deux LEDs de signalisation (led1 sur la broche 40, led2 sur la broche 33).

3.4.3 Cœur du programme : la boucle principale (loop)

La fonction `loop()` s'exécute en continu et gère quatre grandes tâches :

3.4.3.1 Calcul de la vitesse et de la température (toutes les secondes)

Le code utilise `millis()` au lieu de `delay()`, ce qui permet au robot de rester réactif pendant qu'il compte le temps. Toutes les 1000 millisecondes :

- **Vitesse** : elle est calculée à partir de la variable `impulsions` (remplie automatiquement par la fonction `compterImpulsion` grâce à l'interruption matérielle dès que la roue tourne). La formule convertit ces impulsions en tours par minute (RPM).
- **Température** : le code lit la valeur analogique du LM35, la convertit en millivolts (sur une base de 5 V / 1024 positions), puis divise par 10, car le LM35 génère 10 mV par °C.

3.4.3.2 Mesure de distance par ultrason

Le robot envoie une impulsion sonore de 10 microsecondes via le `trigPin`, puis mesure le temps de retour de l'écho avec `pulseIn()`. La formule mathématique ($\text{durée} \times 0,034 / 2$) permet de convertir ce temps en centimètres.

3.4.3.3 Sécurité automatique (priorité absolue)

Avant d'écouter les ordres de l'utilisateur, le code vérifie systématiquement la sécurité :

- **Obstacle à l'avant** : si la distance lue est inférieure à `distanceSecurite` (15 cm), le robot s'arrête immédiatement (`stopper()`) et affiche « !! OBSTACLE !! ».
- **Obstacle à l'arrière** : si l'utilisateur demande de reculer (`commande == 'B'`) et que le capteur PIR détecte un mouvement à l'arrière (`mouvementArriere == HIGH`), le robot s'arrête immédiatement et affiche « !! PIETON ARR !! ».

3.4.3.4 Contrôle par smartphone (commande Bluetooth/série)

Si la voie est libre, le programme regarde si des données sont reçues via le port série (`Serial.available() > 0`). Il lit le caractère envoyé (F, B, L, R, S) et appelle la fonction `executerCommande()` pour modifier la trajectoire.

3.4.4 Mouvements du robot

Les fonctions de mouvement appliquent des tensions électriques aux moteurs pour modifier leur sens et leur vitesse, configurée ici à une valeur PWM de 150 sur 255 pour avancer ou reculer, et de 100 pour tourner :

- **avancer()** : les deux moteurs tournent vers l'avant. La LED 2 s'allume.

- **reculer()** : les deux moteurs tournent vers l'arrière. La LED 1 s'allume, faisant office de feu de recul.
- **tournerGauche()** : le moteur 1 tourne en arrière et le moteur 2 est coupé, ce qui fait pivoter le robot vers la gauche.
- **tournerDroite()** : le moteur 2 tourne en arrière et le moteur 1 est coupé, ce qui fait pivoter le robot vers la droite.
- **stopper()** : tous les moteurs et les LEDs sont coupés (vitesse à 0).

3.4.5 Affichage LCD 20x4

La fonction `afficherStatus()` actualise l'écran LCD avec une organisation structurée :

- **Ligne 1** : titre fixe « PROJET FIN D'ETUDE ».
- **Ligne 2** : l'action ou l'alerte en cours (ex. « AVANCER », « !! OBSTACLE !! »).
- **Ligne 3** : la vitesse en RPM et la température en °C (avec un chiffre après la virgule).
- **Ligne 4** : la distance de l'obstacle en cm. Si le capteur arrière détecte quelque chose, le message « !MOV! » s'affiche également en fin de ligne.

3.4.6 Résumé des commandes acceptées

- 'F' : Forward — avancer
- 'B' : Backward — reculer (sous surveillance du capteur de mouvement)
- 'L' : Left — tourner à gauche
- 'R' : Right — tourner à droite
- 'S' : Stop — s'arrêter

3.5 Réalisation et tests

La réalisation d'un système de commande de véhicule à distance est un projet qui mêle électronique, programmation et réseaux, qu'il s'agisse d'un Arduino ou d'un Raspberry Pi. Cette section présente l'assemblage physique du prototype ainsi que les tests effectués sur le véhicule.

3.5.1 Alimentation électrique

Le tableau suivant résume les niveaux de tension et d'intensité maximale disponibles, utilisés pour alimenter les différents éléments du système :

Alimentation	Intensité maximale disponible	Usage
3,3 V	50 mA	Capteurs basse consommation
5 V	500 mA max.	Alimentation principale utilisée pour le module Arduino et les capteurs

Tableau 4-Niveau d'intensité maximale.

3.5.2 Montage et câblage

Le capteur ultrason HC-SR04 a été fixé à l'avant du châssis pour assurer la détection d'obstacles dans l'axe de déplacement du véhicule :

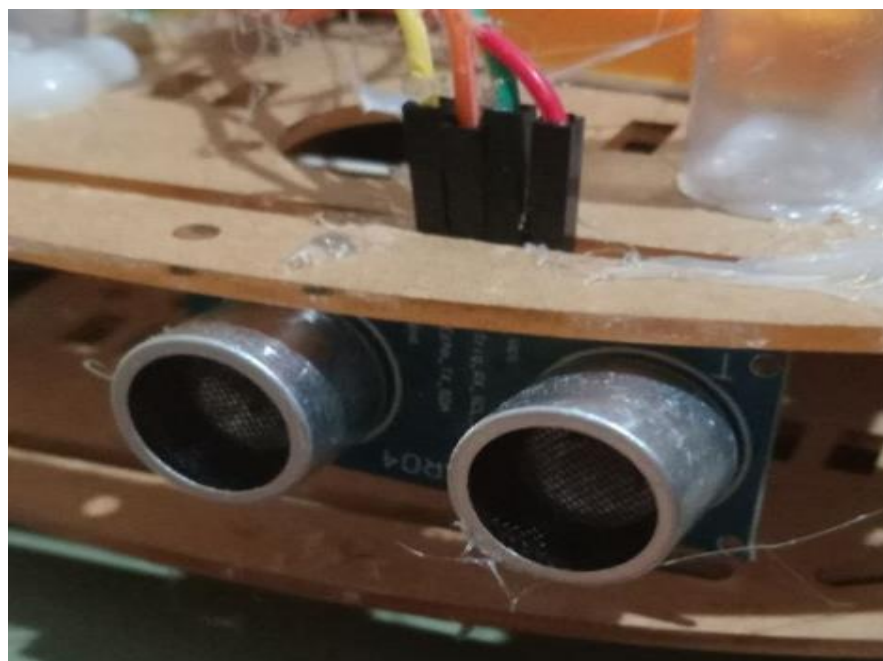


Fig. 44- Capteur ultrason HC-SR04 monté à l'avant du châssis.

La carte Arduino Méga 2560, le module driver moteur (pont en H) et les boîtiers de piles ont été disposés sur le châssis et reliés selon le schéma de câblage présenté précédemment :

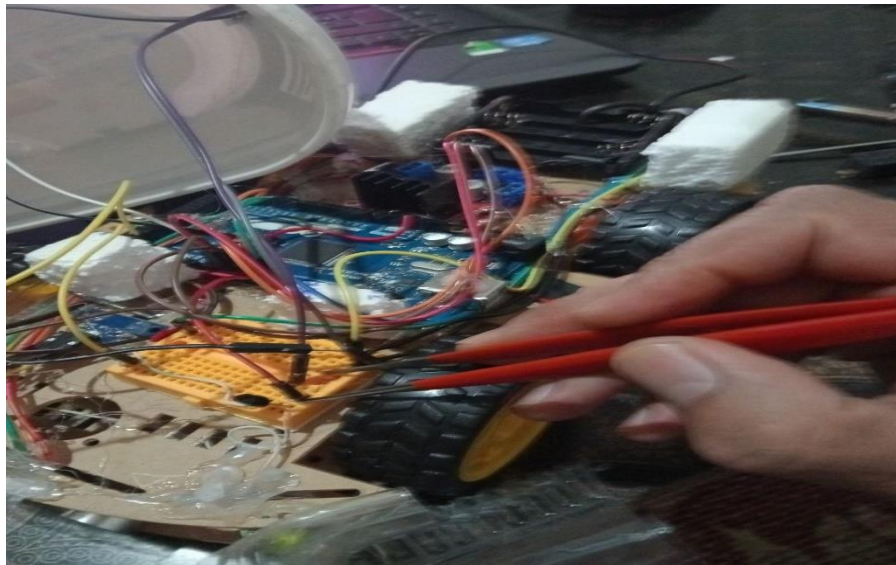


Fig. 45- Câblage de la carte Arduino, du driver moteur et des capteurs.

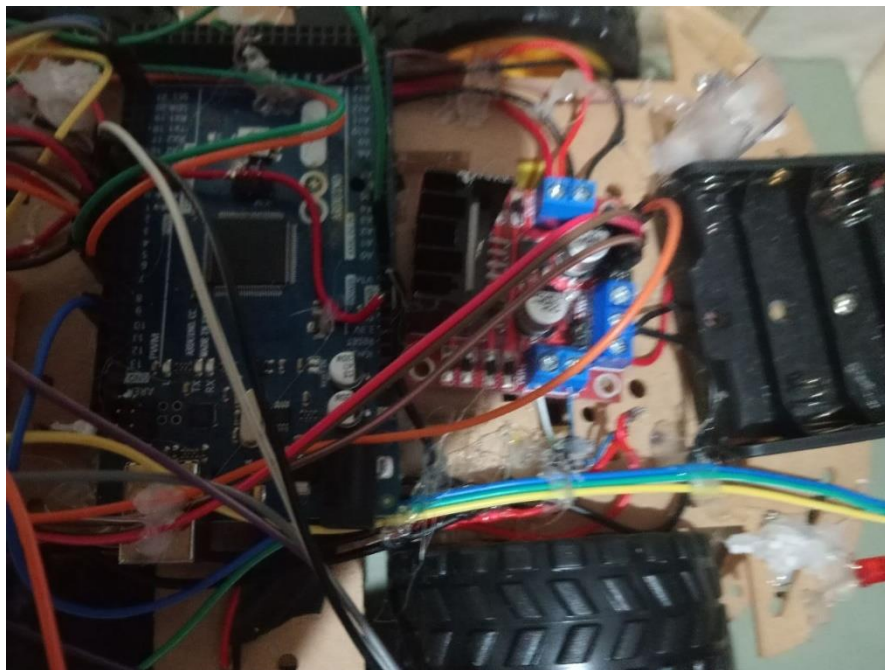


Fig. 46- Détail du câblage entre l'Arduino, le module L298N et l'alimentation

3.5.3 Tests fonctionnels

Les tests effectués sur le prototype assemblé ont permis de vérifier l'ensemble des fonctions décrites dans la section 3.5 : exécution des commandes envoyées depuis le smartphone, détection

et évitement des obstacles, et déplacement du véhicule dans les quatre directions (avant, arrière, droite, gauche). Le véhicule complet, assemblé et câblé, est présenté ci-dessous :

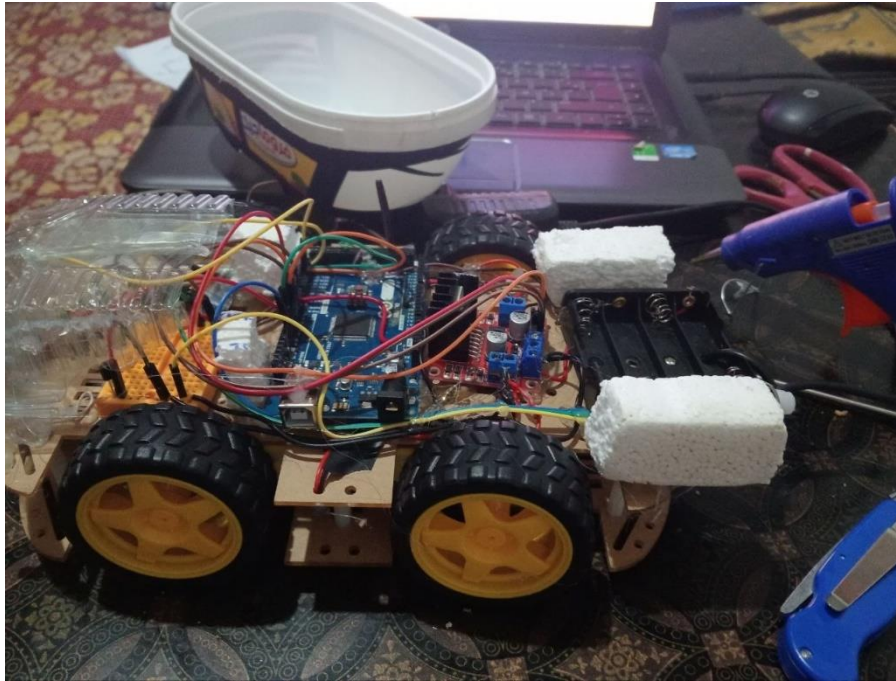


Fig. 47- Prototype final du véhicule télécommandé, assemblé et câblé.

3.6 Conclusion

Ce chapitre a présenté la réalisation pratique du véhicule télécommandé : l'environnement logiciel (IDE Arduino et simulation Proteus), le matériel sélectionné, le schéma de câblage, le programme embarqué sur la carte Arduino Méga 2560, ainsi que l'assemblage et les tests du prototype final.

Le système permet de réaliser concrètement les objectifs fixés : exécuter les commandes envoyées depuis le smartphone, éviter les obstacles grâce au capteur ultrason et au capteur de mouvement, et déplacer le véhicule dans les quatre directions (avant, arrière, droite, gauche), tout en affichant en temps réel sur l'écran LCD la vitesse, la température et la distance aux obstacles.

Conclusion Générale

Conclusion générale

Au terme de ce travail, nous avons réalisé l'étude, la conception et la mise en œuvre d'un système de contrôle d'un véhicule télécommandé basé sur la carte Arduino Méga 2560. Ce projet nous a permis d'approfondir nos connaissances dans les domaines des systèmes embarqués, de l'électronique, de la programmation et des communications sans fil.

Dans un premier temps, nous avons présenté les notions générales relatives aux systèmes embarqués et aux véhicules télécommandés, ainsi que leurs domaines d'application. Ensuite, nous avons étudié les différents composants matériels et logiciels nécessaires à la réalisation du système, en mettant en évidence leurs caractéristiques et leur rôle dans l'architecture globale du projet. Enfin, nous avons procédé à la réalisation pratique du véhicule, à son câblage, à sa programmation et à la validation de son fonctionnement à travers différents tests.

Les résultats obtenus ont montré que le système développé est capable d'assurer le contrôle à distance du véhicule via une liaison Bluetooth, tout en intégrant plusieurs fonctionnalités complémentaires telles que la mesure de distance par capteur ultrasonique, la détection de mouvement, la mesure de température, l'affichage des informations sur écran LCD et la gestion de la sécurité grâce au buzzer et aux LEDs de signalisation.

Ce projet constitue une base intéressante pour le développement de systèmes embarqués plus évolués. Parmi les perspectives d'amélioration, il serait possible d'intégrer une communication Wi-Fi ou Internet des Objets (IoT), d'ajouter un système de navigation autonome avec évitement d'obstacles, ou encore d'utiliser une caméra embarquée permettant la transmission vidéo en temps réel.

Ainsi, ce travail a permis de mettre en pratique les connaissances théoriques acquises durant la formation tout en développant des compétences essentielles dans la conception et la réalisation de systèmes électroniques embarqués.

Bibliographie

- [1] Williams, L. Embedded Systems Tutorial: What is, History & Characteristics. 24, 2024; Disponible sur <https://www.guru99.com/embedded-systems-tutorial.html>.
- [2] Shibu, K., Introduction to embedded systems. 2009: Tata McGraw-Hill Education
- [3] Aggarwal, G. Classification of Embedded Systems. Disponible sur: <https://trainings.internshala.com/blog/classification-of-embedded-systems/>.
- [4] BEN TATA M. et BOUKALLABA A., Conception et développement d'une voiture télécommandée à base d'Arduino, Université Sidi Mohamed Ben Abdellah – Fès, 2021.
- [5] La rédaction, Comprendre les voitures télécommandées et leur fonctionnement. 19 novembre 2024 ; Disponible sur: [Comprendre les voitures télécommandées et leur fonctionnement – Robot et Tech](#)
- [6] Frederic Perrenoud « frederic@perrenoud.com ». Programmation de l'Arduino. De l'idée au concret, comment aboutir dans ses projets. Licence Creative Common CC BYNC-SA. Version: 1.0
- [7] Hamouchi.H, Conception et réalisation d'une centrale embarquée de la domotique, Université Mohamed V – Rabat 2015.
- [8] generationrobots.com spécifications techniques de la carte arduino Mega 2560 <https://www.generationrobots.com/fr/401945-arduino-mega-2560-rev3.htm>
- [9] Selma.H & Moulay. A, Réalisation d'une maison verte dotée des systèmes intelligents de contrôle pour la sécurité, l'environnement et le confort. Université Abou Bakr Belkaid – Tlemcen 2012.
- [10] Guide, quel module ESP8266 choisir, alimentation, programmé avec Arduino IDE <http://sebdelaporte.fr/micro-controleur/esp8266/presentation-de-l-esp8266>
- [11] Module bluetooth HC-06 COMPATIBLE AVEC Arduino-TipTopBoards <http://tiptopboards.com/205-module-bluetooth-hc-06-compatible-avec-arduino>.
- [12] Chapitre 8 Les capteurs – gte.univ-littoral.fr http://gte.univ-littoral.fr/sections/documents-pdagogiques/chapitre-8-mesure/downloadFile/file/Les_capteurs.pdf?nocache=1289041293.82