الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne démocratique et populaire

وزارة التعليم العالي والبحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا
Faculté de Technologie

قسم الإلكترونيك
Département d'Électronique

Mémoire de Master

Filière Électronique
Spécialité Electronique des Systèmes Embarqués

Présentées par

HAIT FATMA_HALLA

&

ZERARKA KHAOULA

Étude et Développement d'un Diagnostic Dédié aux Véhicules Électriques

ESE 15

Encadrées par : Mme TCHOKETCH-KEBIR Selma & Mme CHEGGAGA Nawal

Année Universitaire : 2024-2025

Remerciements

Ce mémoire est la conclusion d'un beau et long parcours universitaire. Avant tout, nous adressons nos louanges et notre profonde gratitude à ***Dieu*** *le* Tout-Puissant, pour nous avoir donné la force afin de mener à bien ce travail. C'est grâce à Sa volonté que nous avons pu surmonter les difficultés et atteindre cette étape importante de finalisation de notre parcours.

Nous remercions tout d'abord notre encadrante, ***Mme TCHOKETCH-KEBIR Selma***, pour sa disponibilité, ses conseils précieux et son accompagnement durant toute la réalisation de ce travail.

Nous adressons également nos sincères remerciements à notre Co-encadrante, ***Mme CHEGGAGA Nawal***, pour son aide, ses conseils avisés et son soutien tout au long de ce mémoire. Grâce à leurs orientations, nous avons pu avancer avec plus de confiance et de motivation.

Nous exprimons également notre gratitude à tous les enseignants et membres du corps administratif du ***département d'Électronique*** pour leur accompagnement durant ces années.

Nous remercions aussi toute l'équipe du ***Centre de recherche en technologies industrielles (CRTI)***, et particulièrement ***Mr. DRAI***, pour leur accueil sympathique, leur soutien et leur patience tout au long de notre stage. Cette expérience nous a beaucoup appris, aussi bien sur le plan professionnel que personnel.

Un grand remerciement à **nos familles** pour leur amour, leur patience et leurs encouragements, même quand les choses semblaient difficiles. Leur présence nous a beaucoup soutenues.

Nous tenons à adresser nos sincères remerciements aux **membres du jury** pour avoir accepté d'évaluer notre travail.
Nous sommes honorées par leur présence et reconnaissantes pour le temps qu'ils nous ont consacré, ainsi que pour leurs remarques constructives et enrichissantes.

Merci aussi à **nos amis** pour leurs présences, leurs encouragements et leurs petits mots qui nous ont aidées à garder le sourire pendant les moments difficiles.

Et enfin, un remerciement très spécial pour nous-mêmes, ***Ilham*** *&* ***Khaoula***. Ce mémoire est le résultat d'un vrai travail d'équipe. Merci l'une à l'autre pour la confiance, l'écoute, la patience et l'entraide tout au long de ce parcours. Merci aussi pour les heures de travail partagées, les moments de doute, les fous rires au milieu de la fatigue et cette belle complicité qui a rendu ce projet possible et agréable.

À toutes et à tous, un remerciement.

Dédicaces

Je dédie ce mémoire avec tout mon amour et ma gratitude :

À **Dieu Tout-Puissant**, pour m'avoir guidée, soutenue et protégée tout au long de ce parcours.

À **mes parents**, pour leur immense amour, leur patience et tous les sacrifices qu'ils ont faits pour moi.

À **mes frères et sœurs**, pour leur soutien, leur présence et leurs encouragements constants.

À **mes grands-parents**, dont les prières et l'amour m'accompagnent toujours.

À **mes tantes**, pour leur bienveillance, leurs conseils et leur tendresse, particulièrement celle qui m'avait accueillis au niveau de *Cheraga* durant le stage au *CRTI*.

À **mes cousines et cousins**, pour leur joie, leur soutien et leurs mots gentils.

À **mes amis**, pour leur présence, leur humour et leurs encouragements, même dans les moments les plus stressants.

À **notre encadrante**, pour sa confiance, sa disponibilité, et son accompagnement bienveillant. Ce travail est aussi le reflet de son soutien.

Et enfin à ***Ilham***, ma binôme, pour cette belle collaboration. Merci pour ta patience, ta confiance, et tous les souvenirs partagés durant ce parcours.

Et une grande dédicace à **moi-même**.

Khaoula

Dédicaces

Je dédie ce mémoire avec reconnaissance et affection :

À **Dieu**, pour sa force, sa sagesse et sa lumière qui m'ont portée jusqu'ici.

À **mes parents**, pour leur amour, leurs encouragements, et tout ce qu'ils ont fait pour moi depuis le début.

À **mes amis**, pour leur bonne humeur, leur écoute et leur aide précieuse.

À **notre encadrante**, pour sa confiance, sa disponibilité, son professionnalisme, son attention et son appui durant toute ce parcours.

Et tout particulièrement à ***Khaoula***, ma binôme, une partenaire exceptionnelle. Merci pour ta gentillesse, ton sérieux, ton esprit d'équipe et tous les moments partagés qui ont fait de ce projet une belle réussite.

Et une grande dédicace à **moi-même**.

ILham

ملخص:

الإلكترونية أنظمتها تعقيد بسبب وذلك والصيانة، التشخيص مجال في جديدة تحديات الكهربائية السيارات تطور يشكل للمركبة الحالية البيانات مراقبة على قادر تشخيص نظام تصميم البحث هذا يتناول .بروتوكولات على واعتمادها وتنبيه الأعطال اكتشاف بهدف الأعطال، وأكواد والجهد، الحرارة، ودرجة البطارية، المحرك، حالة مثل الكهربائية، تحسين هو العمل هذا من الهدف .والتصحيحية الوقائية الصيانة لتسهيل ذكية شاشة عبر الاظهار بواسطة المستعمل .ودقيقة فعالة تشخيص أداة توفير خلال من الكهربائية المركبات وسلامة موثوقية

الكلمات المفتاحية:

السيارات الكهربائية، نظام التشخيص، شاشة ذكية، بروتوكولات.

Résumé

L’essor des véhicules électriques (VE) pose de nouveaux défis en matière de diagnostic et de maintenance, en raison de leur complexité électronique et de l'utilisation de protocoles comme le CAN et l’OBD-II. Ce mémoire porte sur le développement d’un système de diagnostic capable de surveiller les données critiques des VE (Moteur, batterie, température, tension, erreurs, etc.) et d’afficher aux utilisateurs les alarmes via un écran intelligent afin de faciliter la détection des anomalies et faciliter la maintenance. L’objectif est d’améliorer la fiabilité et la sécurité des VE grâce à un outil de diagnostic efficace.

Mots clés :

Véhicules électriques, diagnostic, protocole CAN, affichage intelligent.

Abstract:

The growth development of electric vehicles (EVs) brings new challenges in diagnosis and maintenance due to their complex electronic systems and reliance on protocols like CAN and OBD-II. This project focuses on developing a diagnosis prototype capable of monitoring critical EV’s components such as motor, battery, temperature, voltage, and error codes, to detect faults, display states through intelligent display, and facilitate both preventive and corrective maintenance. The aim of this work is to enhance the reliability and safety of EVs by providing an accurate and efficient diagnosis tool.

Keywords:

Electric vehicles, diagnosis, CAN protocol, intelligent display.

Listes des acronymes et abréviations

ABS: Anti-lock Braking System

ADAS : Advanced Driver Assistance Systems

BMS: Battery Management System

BMU: Battery Management Unit

CAN: Controller-Area-Network

CRC: Cyclic Redundancy Check

DTC: Diagnostics Trouble Codes

ECU : Unités de commande électroniques

ESC: Electronic Stability Control

FDD: Fault Detection & Diagnostics

HS: High-Speed

LIN: Local Interconnected Network

LS : Low-Speed

MCU : Unité de contrôle du moteur

OBD : On-Board-Diagnosis

PDU : L'électronique de puissance via l'unité de distribution d'énergie

PID: Parameter-Identifier Diagnosis

SAE: Society of Automotive Engineers

SE : Systèmes Embarqués

SEE : Systèmes Electroniques Embarqués

VE : Véhicules électriques

US : Ultrason

Table des matières

Chapitre 1 : Généralités des Systèmes Embarqués et du Diagnostic Automobile

Chapitre 2 : Protocoles de Communication & Outils de Diagnostic

Liste des figures

Liste des tableaux

Introduction Générale

Récemment, l'industrie automobile connaît une évolution majeure avec l'essor des véhicules électriques (VEs), qui sont devenus une alternative prometteuse aux véhicules traditionnels (thermiques). Grâce à leur efficacité énergétique accrue et leur faible impact environnemental, les VEs se positionnent comme des solutions incontournables pour la mobilité durable. Cependant, malgré leurs nombreux avantages, ces véhicules introduisent de nouveaux défis techniques, notamment en termes de maintenance et de diagnostic des pannes.

Contrairement aux véhicules traditionnels à moteur thermique, les VEs sont dotés d'une architecture électronique avancée, composée de divers systèmes critiques tels que le moteur électrique, la batterie de traction, l'électronique de puissance et les unités de contrôle embarquées. Un dysfonctionnement dans l'un de ces composants peut entraîner une dégradation des performances, voire une panne complète du véhicule. Le diagnostic du véhicule nécessite une définition complète, qui nécessite l'étude de tous les défauts qui se produisent et la corrélation avec les défauts actuels. Il est donc primordial de mettre en place des outils de diagnostic embarqués capables de surveiller en temps réel l'état des différents sous-systèmes et d'identifier rapidement les anomalies.

Aujourd'hui, les systèmes de diagnostic disponibles reposent majoritairement sur des protocoles de communication standards, notamment le Controller Area Network (CAN) et l'On-Board Diagnostics II (OBD-II). Cependant, les outils d'analyse associés sont souvent coûteux, complexes et réservés aux professionnels (mécaniciens), ce qui limite leur accessibilité aux utilisateurs indépendants et aux ateliers de réparation. Ainsi, il devient nécessaire de développer une solution ouverte, flexible et à moindre coût pour permettre une meilleure compréhension et exploitation des données relatives au diagnostic des véhicules électriques.

Dans ce contexte, ce travail de projet de fin d'étude porte à concevoir et à développer un système de diagnostic embarqué dédié aux VEs (en exploitant des technologies accessibles tel qu'un microcontrôleur Arduino et un module MCP2515 pour l'acquisition des trames CAN). L'objectif principal est d'analyser en temps réel les données du VE, de détecter et d'identifier les anomalies potentielles, ainsi d'afficher les paramètres essentiels (alarmes) via une interface utilisateur intuitive.

Dans ce contexte, nous avons effectué un stage pratique au sein du Centre-de-Recherche en Technologies-Industriels (CRTI), pour réaliser des montages et avoir une bonne maitrise sur le plan pratique.

À travers le développement de ce projet, nous ambitionnons de proposer une solution de diagnostic innovante et accessible, permettant d'améliorer la fiabilité et la maintenance des VEs avec un cout réduit, tout en facilitant leur adoption par un plus grand nombre d'utilisateurs.

Ce mémoire est organisé selon la structure suivante :

Le premier chapitre porte sur une présentation générale des différents systèmes embarques constituants le VEs, leurs défauts et les méthodes de diagnostics relatifs à chaque partie.

Le deuxième chapitre présente les protocoles de communications au sein des VEs, particulièrement le CAN et les outils du diagnostic automobile.

Le troisième chapitre permet de traiter une base de données contenant des données réelles d'un VE (Vitesse de rotation du moteur et vitesse du VE), par l'utilisation d'une méthode intelligente.

Le dernier chapitre, présente le prototype de diagnostic développes, les montages réalises et les résultats obtenus.

Nous avons terminé ce mémoire par une conclusion générale résumant l'essentiel obtenus des résultats et enfin les perspectives envisagées pour la continuité de ce travail.

Chapitre 1 : Généralités des Systèmes Embarqués et du Diagnostic Automobile

1. 1 Introduction

Récemment, l'industrie automobile reconnait une phase de transition majeure, notamment avec l'émergence des véhicules électriques (VE). Cette révolution technologique trouve son origine dans la nécessité de répondre à des enjeux environnementaux de plus en plus pressantes, tel que la réduction des émissions du gaz à effet de serre (GES) et la dépendance aux énergies fossiles. Cependant, ces VE fonctionnant à base de l'énergie stockée dans des batteries rechargeables, se présentent comme une alternative écologique aux véhicules thermiques traditionnels [1]. Toutefois, cette nouvelle génération de véhicules repose sur une architecture complexe, constituée de plusieurs systèmes électroniques et électromécaniques interconnectés par des protocoles de communication développés (Figure 1.1). L'un des aspects les plus importants de cette architecture est l'intégration des systèmes embarqués (SE)**,** qui sont responsables de la gestion et du contrôle de l'ensemble des sous-systèmes du véhicule. Ces systèmes jouent un rôle crucial dans la régulation de la consommation d'énergie, la gestion de la batterie, la régulation de la vitesse du moteur du VE, ainsi que la sécurité des conducteurs et des passagers. À l'ère de la mobilité intelligente, le rôle des SE dans les VE devient donc primordial.

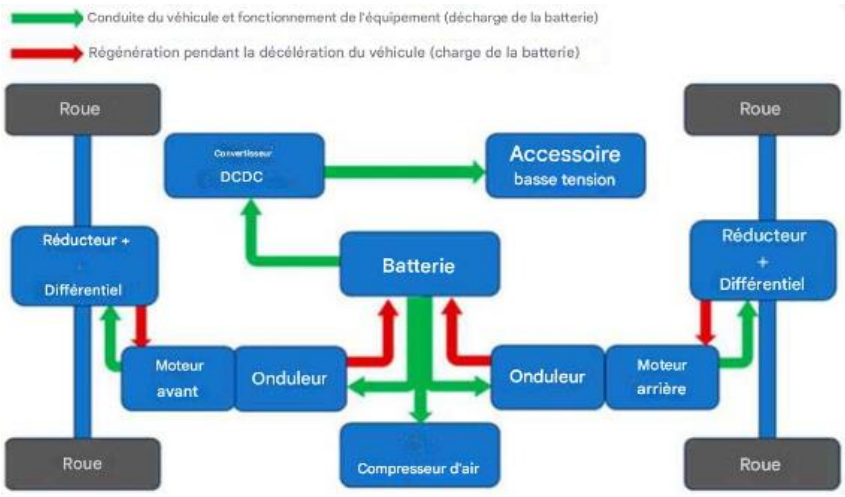


Figure 1.1 Constituants essentiels du VE (Moteur, Batterie, & électronique de puissance). [2]

Le diagnostic embarqué se trouve au cœur de ce système électronique intégré. Il s'agit d'un ensemble de techniques et de technologies permettant de surveiller, d'analyser et de diagnostiquer l'état des différents composants du véhicule en temps réel. Ainsi, ce diagnostic dans un VE est d'autant plus essentiel puisqu'il permet d'assurer le bon fonctionnement du moteur électrique, de la batterie, des composants électroniques de puissance et leurs communications. En effet, une défaillance dans l'un de ces sous-systèmes peut avoir des conséquences sur la performance, la sécurité et la durabilité du véhicule.

Ce chapitre consiste à présenter les concepts fondamentaux des SE dans les VE. Par la suite, les différents défauts qui peuvent survenir dans les VE sont présentés, afin de les détecter, contrôler et diagnostiquer, avec une concentration sur les défis techniques et les opportunités liées à la mise en place d'un système de diagnostic embarqué. En effet, une présentation des méthodes de diagnostic et leurs classifications afin d'assurer un meilleur fonctionnement.

1.2 Systèmes embarqués dans VE

Un système embarqué est un système électronique, informatique et autonome qui peut être intégré comme un dispositif matériel. Il est conçu pour effectuer une tâche bien précise, qui fonctionne souvent en temps réel. Ces systèmes combinent généralement un microprocesseur ou un microcontrôleur, des mémoires, des interfaces de communication et des capteurs/actionneurs (Figure 1.2). Dans le domaine de l'automobile, les SE sont omniprésents. Ils interviennent dans la gestion du moteur, le freinage assisté, la recharge électronique, les systèmes de sécurité (Anti-Blockier-System (ABS), un système de sécurité avancé pour prévenir les accidents (Electronic Stability Program (ESP)), l'affichage au tableau de bord (Human-Machine-Interface (HMI)), la navigation (Global-Positioning-System (GPS)), et bien plus encore. Dans un véhicule moderne, on peut compter plus qu'à 70 unités de commande électroniques (ECU) communicant entre elles via des protocoles comme le Controller-Area-Network (CAN).

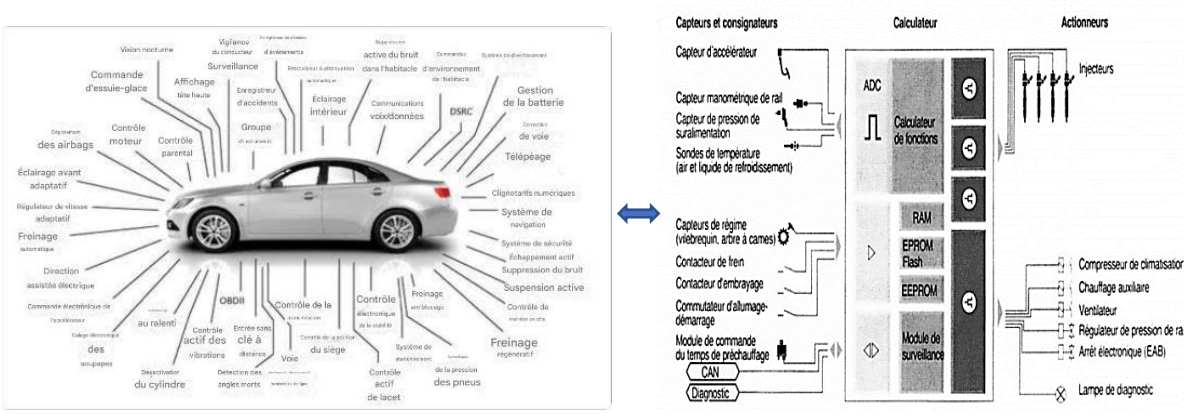


Figure 1.2 SE dans VE. [3]

Le développement des VE a amplifié l'intégration et l'utilisation des SE. Chaque composant principal (moteur, batterie, onduleur, système de recharge) est géré par des ECU spécialisés, nécessitant des mécanismes de contrôle et de diagnostic avancés. Un VE est constitué autour d'une architecture spécifique intégrant plusieurs sous-systèmes électroniques pilotés par des logiciels embarqués. Ces derniers seront présentés dans les sous-sections suivantes.

1.2.1 Unité de contrôle du moteur (MCU)

Le moteur électrique du VE permet de convertir l'énergie électrique en une énergie mécanique. Contrairement à un moteur thermique, il est plus simple mécaniquement mais exige un contrôle électronique plus précis (vitesse, couple, fréquence et température). L'unité de contrôle du moteur (MCU) des VE intègre plusieurs autres composants clés afin de garantir une gestion efficace du moteur et sa sécurité, qui sont illustrés dans la Figure 1.3.

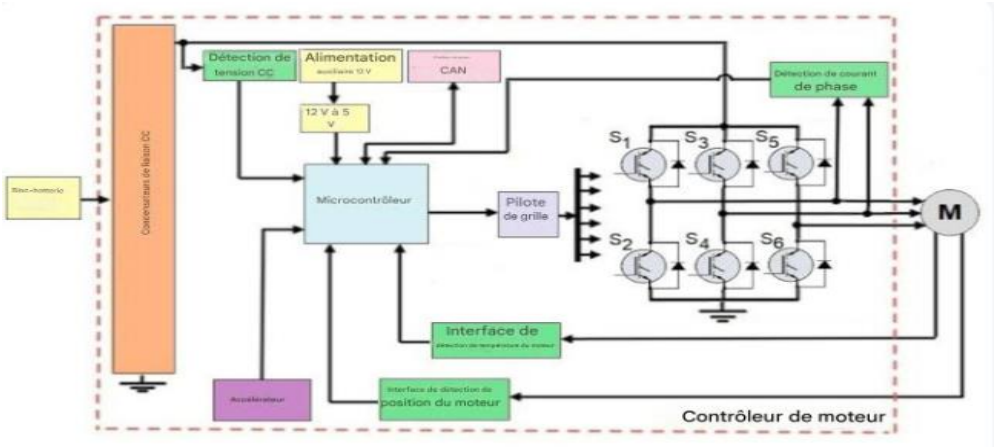


Figure 1.3 Exemple d'un moteur pour VE muni de son contrôle (MCU). [4]

1.2.2 Unité de stockage (BMU)

Le stockage est un élément essentiel pour le VE afin de garantir sa consommation énergétique. Ce stockage est assuré par une unité contenant un ensemble de batteries (Battery Management Unit (BMU)). Elle permet de stocker l'énergie électrique et alimenter le moteur via une électronique de puissance, schématisée dans la Figure 1.4.

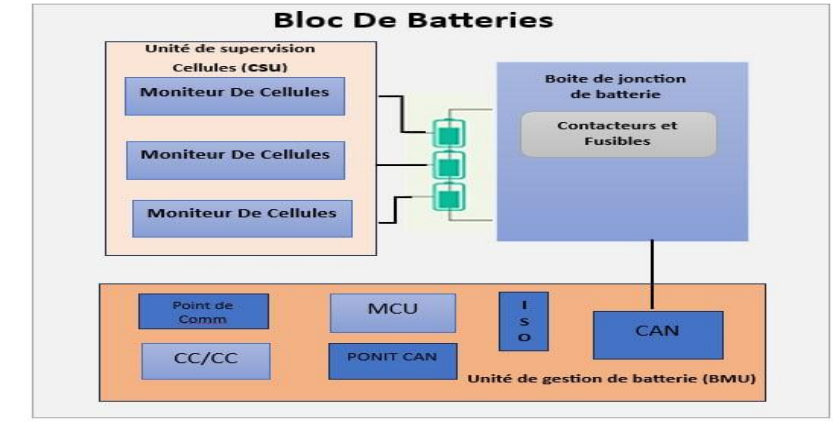


Figure 1.4 Schéma synoptique d'une BMU. [5]

- **Surveillance BMS**

La gestion de la batterie (Battery Management System - BMS) permet de surveiller plusieurs paramètres comme la température, la tension, le courant, le SOC (State Of Charge), et le SOH (State Of Health). Ce BMS est illustré dans la Figure 1.5.

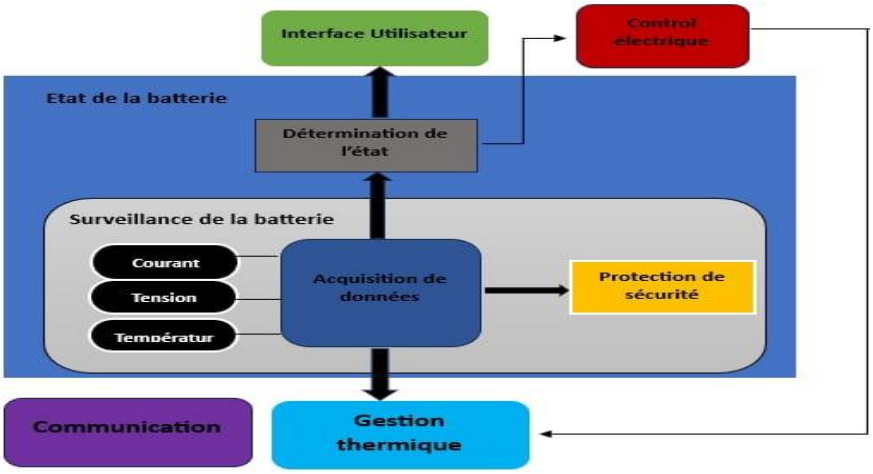


Figure 1.5 Exemple d'un BMS. [6]

1.2.3 Unité d'électronique de puissance (PDU)

Cette électronique de puissance nommée Power-Distribution-Unit (PDU) assure la conversion et la gestion d'énergie entre la batterie, le moteur et d'autres parties (Figure 1.6). Elle comprend l'onduleur, les convertisseurs DC/DC et le chargeur embarqué (On-Board-Charger (OBC)).

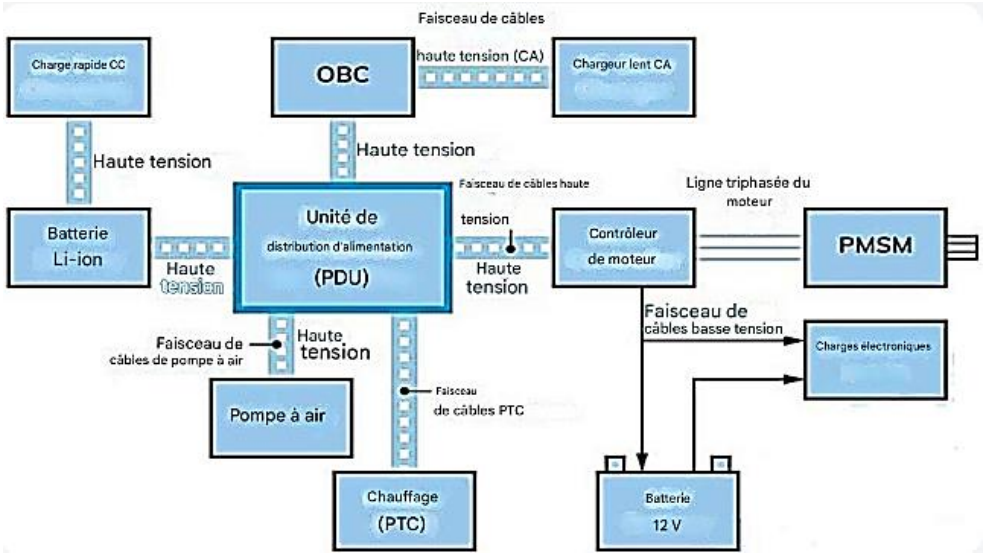


Figure 1.6 Schéma synoptique d'une PDU. [7]

1.2.4 Unités de contrôle électroniques (ECU)

Chaque sous-système du VE est contrôlé par une ou plusieurs ECUs. Celles-ci communiquent via un réseau interne (généralement le protocole CAN) et exécutent des algorithmes de contrôle, de protection, et de diagnostic tel que schématisé dans la Figure 1.7.

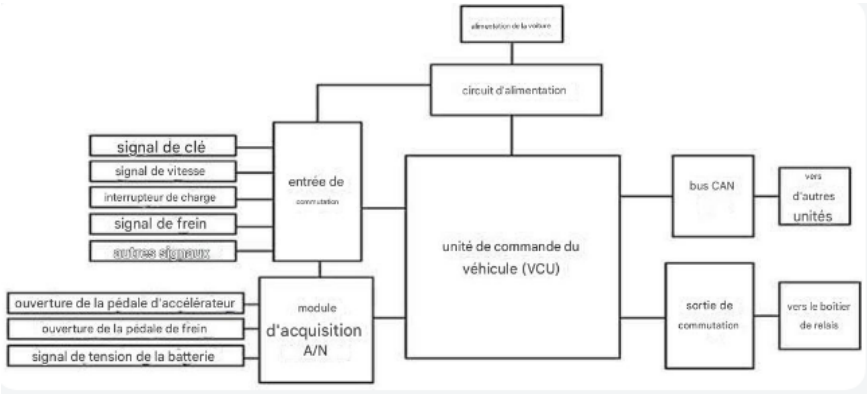


Figure 1.7 Schéma représentatif d'une ECU. [8]

1.2.5 Unité de diagnostic embarqué (OBD)

Le diagnostic embarqué (On-Board-Diagnostics (OBD)) vise à surveiller en temps réel l'état de santé des composants du VE. Il permet de détecter, localiser et interpréter les pannes potentielles ou effectives.

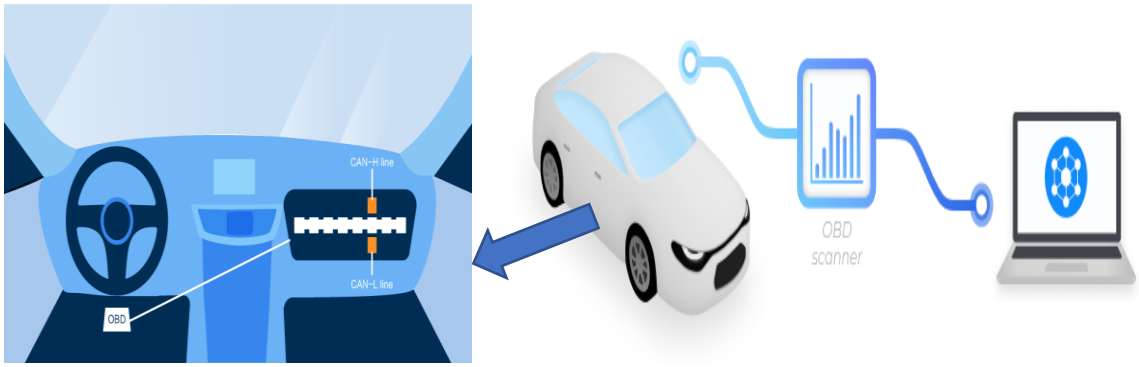


Figure 1.8 Représentation d'un diagnostic embarqué (OBD). [9] [10]

1.3. Diagnostic dans le VE :

Le mot « diagnostic » provient du grec ancien avec une signification (discernement) qui désigne l'identification de la nature et de la cause de quelque chose. Le diagnostic est utilisé dans de nombreuses disciplines différentes, mais toutes font appel à la logique, à l'analyse et à l'expérience pour déterminer les relations de cause à effet. En ingénierie automobile, le diagnostic est généralement utilisé pour identifier les causes des défauts et proposer des solutions aux problèmes rencontrés [11]. Le diagnostic automobile peut être effectué par des méthodes traditionnelles ou récentes. Les méthodes traditionnelles peuvent être basées sur l'observation des symptômes par le réparateur (mécanicien, technicien automobile). Ces méthodes sont souvent lentes, moins précises et nécessitent un niveau élevé d'expertise technique. En effet, les méthodes récentes impliquent un diagnostic embarqué (telles les valises de test), qui s'appuie sur le protocole de communications entre les ECUs, pour identifier de manière automatique les défaillances. Les informations coulent via le protocole CAN entre les ECUs du véhicule et sont disponibles par une interface OBD, qui permet l'interfaçage avec le diagnostic embarqué. Ces méthodes sont plus évoluées et offre plus de sécurité.

Figure 1.9 Le diagnostic dans le VE. [12]

1.3.1 Importance du diagnostic

Avec la transition vers l'électromobilité, les VE deviennent de plus en plus complexes en raison de l'intégration de multiples systèmes électroniques et électromécaniques. Cette complexité rend le **diagnostic embarqué** indispensable pour garantir un bon fonctionnement, une **maintenance efficace** et une **sûreté des usagers**. Contrairement aux véhicules thermiques, les VE reposent fortement sur des composants sensibles comme la batterie de traction, l'onduleur, ou encore les moteurs synchrones, dont les défaillances peuvent entraîner des pannes critiques, voire des risques de sécurité (incendie, perte de puissance, blocage du système de freinage régénératif, etc.). [13]

Le diagnostic embarqué (OBD) permet une **détection précoce des anomalies**, une **surveillance en temps réel des paramètres critiques** (tension, température, courant, etc.), et offre la possibilité d'alerter immédiatement le conducteur ou le technicien. Il assure également la **traçabilité des pannes** via l'enregistrement des codes défauts (Diagnosis Trouble Codes (DTC)), ce qui facilite les interventions de maintenance et réduit considérablement les temps d'immobilisation du véhicule. En parallèle, le diagnostic contribue au **respect des normes environnementales** et techniques imposées par les régulations (comme l'OBD-II), et joue un rôle clé dans l'optimisation de la durée de vie des composants, en particulier celle des batteries. En effet, le diagnostic n'est pas un simple outil de réparation, mais un outil indispensable pour les VEs modernes (la maintenance préventive, l'optimisation des performances, la sécurité des usagers et le respect des normes environnementales).

1.3.2 Méthodes de diagnostic des défauts

Les méthodes de diagnostic des défauts (Faults Diagnosis) sont des approches utilisées pour détecter, localiser et identifier les défaillances dans un système, qu'il soit électrique, électronique, mécanique ou logiciel. [14]

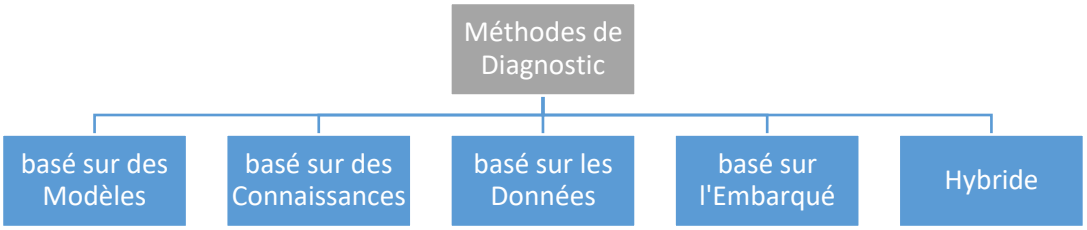


Figure 1.10 Classification des méthodes de diagnostic.

Voici une classification des principales méthodes de diagnostic des défauts, souvent utilisées dans l'industrie automobile, notamment pour les véhicules électriques :

1.3.2.1 Méthodes basées sur des modèles (Model-Based Diagnosis)

Il s'agit de méthodes qui utilisent un modèle mathématique ou physique du système pour comparer le comportement réel au comportement attendu.

a) Méthode par observateur

Cela utilise des observateurs d'état comme *Luenberger* ou *Kalman* pour estimer les états internes. En effet, un défaut est indiqué par un écart entre la sortie mesurée et celle estimée.

b) Analyse des résidus

Le résidu est la différence entre la sortie réelle et celle du modèle. Si le résidu est supérieur à un seuil, cela signifie un défaut.

c) Méthode par réseaux de Petri

Représentation formelle du système avec états et transitions. Ceci, est pratique pour les systèmes complexes ou non-linéaires.

1.3.2.2 Méthodes basées sur la connaissance (Knowledge-Based Diagnosis)

a) Systèmes experts

Utilisent des règles "si-alors" basées sur l'expérience humaine.

Exemple : "Si la tension de batterie est < 10V, alors suspecter une décharge profonde".

b) Logique floue

Utilise des degrés de vérité à la place des booléens simplistes. Ce qui mène cette méthode d'être applicables à des systèmes incertain ou imprécis.

c) Réseaux bayésiens

Procédure basée sur des probabilités prenant en compte le manque de précision lors de la détection de défauts.

1.3.2.3 Méthodes basées sur les données

Avec l'avènement des données volumineux (Data-Driven Diagnosis, big-data), l'intelligence artificielle (AI) et ses dérivés le Machine-Learning (ML) et le Deep-Learning (DL), trouve leurs utilités pour exploiter les données historiques, en temps réel pour prédire et diagnostiquer les défauts.

1.3.2.4 Méthodes de diagnostic embarqué (On-Board Diagnosis)

Intégrées dans le véhicule via le port OBDII (On-Board Diagnostics). Utilise des capteurs internes, des codes d'erreur (DTC), et l'analyse du protocole CAN. Ce qui permet une détection en temps réel sans intervention externe.

1.3.2.5 Méthodes hybrides

Utilisent une combinaison entre plusieurs modèles : modèle + données, expert + réseau de neurones, etc. Ce qui mène à des résultats plus convaincants, notamment dans les systèmes complexes comme les voitures électriques.

1.3.3 Méthodes de détection et de diagnostic dans le VE

La détection et le diagnostic des pannes (Faults Detection & Diagnosis (FDD)), dans les VEs sont essentiels pour garantir la sécurité, la fiabilité et les performances du véhicule. Les systèmes FDD efficaces permettent d'identifier et de diagnostiquer les pannes ou anomalies dans les différents sous-systèmes du véhicule, notamment le moteur, la batterie, le système de traction électrique, l'électronique de puissance et les systèmes de gestion thermique. En détectant les problèmes de manière précoce, les systèmes FDD peuvent prévenir les défaillances potentielles, réduire les coûts de maintenance et améliore le fonctionnement global du véhicule. [15]

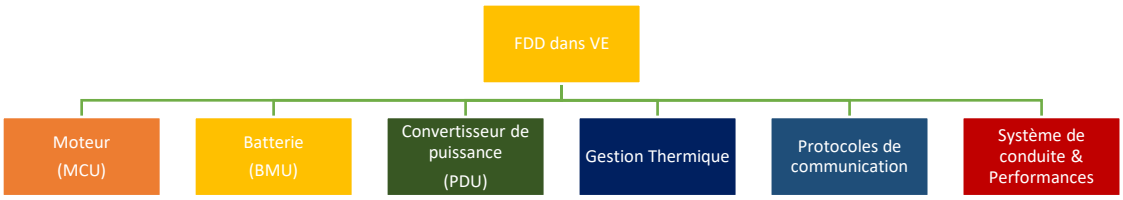


Figure 1.11 Méthodes de détection et du diagnostic dans VE.

1.3.3.1 FDD dans MCU

Le système de traction du moteur électrique (MCU) comprend le moteur électrique, la chaîne de transmission et les systèmes de commande associés. Les pannes dans ce MCU (Figure 1.12) peuvent affecter l'accélération, la maniabilité et les performances globales du véhicule.

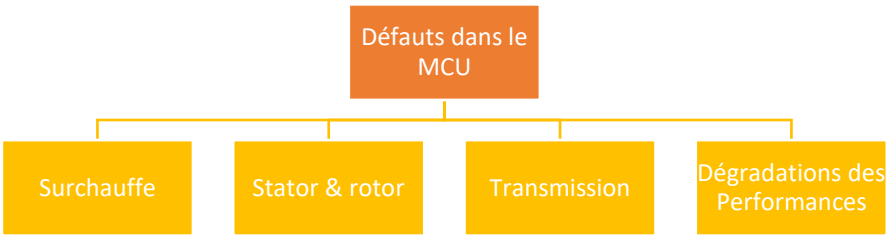


Figure 1.12 Méthodes de détection et du diagnostic dans MCU.

Ci-suivant nous allons présenter les défauts du moteur et les méthodes de son diagnostic.

a) Surchauffe du moteur

- **Détection :** Des capteurs de température surveillent la température du moteur.

- **Diagnostic** : Des températures élevées peuvent indiquer un problème de refroidissement ou une charge excessive.

b) Défauts du stator et du rotor dans le moteur

- **Détection :** Analyse par capteurs de vibrations et mesures électriques.
- **Diagnostic :** Des vibrations anormales ou des signatures électriques inhabituelles peuvent indiquer des défauts dans les enroulements ou les roulements du moteur.

c) Problèmes de transmission

- **Détection :** Surveillés via des capteurs installés dans la chaîne de transmission.
- **Diagnostic :** Des bruits inhabituels ou des problèmes de performance peuvent indiquer des défauts dans les engrenages, roulements ou autres composants de la transmission.

d)Dégradation des performances

- **Détection** : Suivi des indicateurs de performance tels que le couple et le rendement.
- **Diagnostic** : Une baisse de performance peut indiquer des problèmes liés au moteur ou au système de commande.

Il existe ainsi, des techniques avancées pour diagnostiquer MCU, cités ci-suivant :

e) Détection de pannes par analyse des signaux

L'analyse des signaux électriques du moteur permet de détecter des anomalies. Ceci fournit des informations détaillées sur les performances du moteur et les défauts.

f) Maintenance prédictive

L'utilisation des données historiques et des algorithmes pour prévoir les pannes potentielles avant qu'elles ne surviennent. Ce qui permet une maintenance proactive, réduisant ainsi les risques de pannes imprévues.

g) Techniques de redondance

Redondance matérielle, redondance logicielle. Ceci, assurent le fonctionnement continu du système même en cas de défaillance partielle.

h) Conception de systèmes critiques pour la sécurité

Intègre la tolérance aux fautes dans les systèmes essentiels à la sécurité. Ce qui améliore la fiabilité globale du véhicule et la sécurité des passagers.

3.3.2 FDD dans BMU

La détection des pannes est un aspect essentiel de la gestion de la batterie, garantissant la sécurité et la fiabilité du système de batterie. Le système de gestion de batterie (BMS) surveille en continu divers paramètres tels que la tension, le courant et la température afin d'identifier les anomalies pouvant indiquer des pannes potentielles (Figure 1.13). La détection précoce de problèmes tels que la surcharge, la décharge profonde ou l'emballement thermique permet une intervention rapide, évitant ainsi les dommages et renforçant la sécurité globale du véhicule.[15]

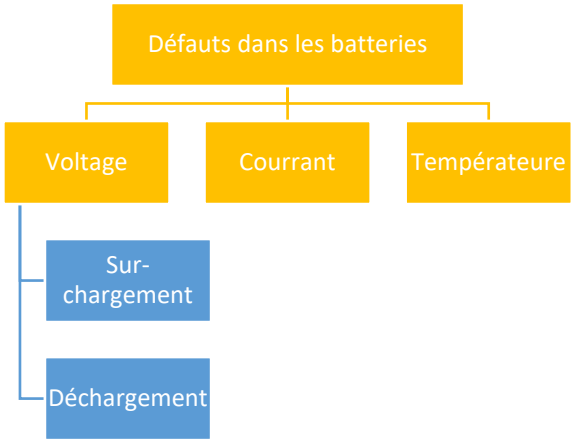


Figure 1.13 Méthodes de détection et du diagnostic dans BMU.

Ci-suivant nous allons présenter les défauts de la batterie et leurs méthodes de diagnostic.

a) Surtension et sous-tension

- **Détection** : Surveillées à l'aide de capteurs de tension sur chaque cellule ou module.
- **Diagnostic** : Des anomalies dans les lectures de tension par rapport aux limites attendues qui peuvent indiquer une surcharge, une décharge profonde ou des cellules défectueuses.

b) Surcharges de courant et courts-circuits

- **Détection** : Surveillés à l'aide de capteurs de courant et de disjoncteurs.

- **Diagnostic** : Des pics ou des chutes anormales de courant peuvent signaler des problèmes tels que des courts-circuits, des défauts internes ou une charge excessive.

c) Anomalies de température

- **Détection** : Des capteurs de température surveillent la température du banc batterie.
- **Diagnostic** : Des relevés de température anormaux peuvent indiquer une surchauffe due à une charge élevée, un mauvais refroidissement ou un emballement thermique dans les cellules.

d) Augmentation de la résistance interne

- **Détection** : Mesurée à l'aide de capteurs d'impédance ou de résistance.
- **Diagnostic** : Une augmentation de la résistance interne peut indiquer une dégradation ou un dysfonctionnement des cellules.

e) **Écarts de l'état de charge (SoC) et de l'état de santé (SoH)**

- **Détection** : Surveillés à l'aide d'algorithmes d'estimation du SoC et du SoH.
- **Diagnostic** : Des écarts importants par rapport aux valeurs attendues de SoC ou du SoH peuvent révéler une dégradation de la batterie ou des problèmes de calibration.

Il existe ainsi, des techniques avancées pour diagnostiquer BMU, cités ci-suivant :

f) Spectroscopie d'impédance électrochimique (EIS) : Mesure l'impédance de la batterie à différentes fréquences pour détecter des défauts internes et la dégradation. Cependant, elle fournit des informations détaillées sur l'état de la batterie et les sources de défaillance.

g) Utilisation du ML : Utilise des bases de données historiques pour entraîner des modèles capables de prédire et détecter des anomalies dans les performances de la batterie. Cependant, elle permet d'identifier des schémas et corrélations subtils que les méthodes traditionnelles peuvent ne pas détecter.

1.3.3.3 FDD dans PDU

L’électronique de puissance (PDU), incluant les convertisseurs DC/DC et les onduleurs, qui ont un rôle essentiel dans le contrôle du flux d’énergie entre la batterie et le moteur électrique. Les pannes de ces convertisseurs (Figure 1.14) peuvent affecter les performances et la sécurité du véhicule.

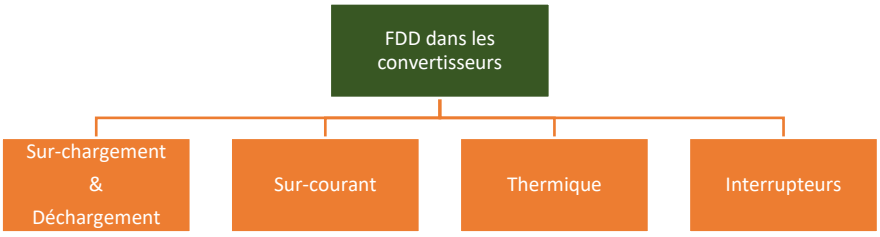


Figure 1.14 Méthodes de détection et du diagnostic dans PDU.

Ci-suivant nous allons présenter les défauts dans PDU et leurs méthodes de diagnostic.

a) Surtension et sous-tension

- **Détection** : Surveillées à l’aide de capteurs de tension dans les circuits d’électronique de puissance.
- **Diagnostic** : Des lectures de tension anormales peuvent indiquer des défaillances de composants ou des problèmes d’alimentation électrique.

b) Sur-courant

- **Détection** : Surveillée à l’aide de capteurs de courant dans les circuits d’électronique de puissance.
- **Diagnostic** : Un courant excessif peut signaler des défaillances de composants ou des courts-circuits.

c) Surcharge thermique

- **Détection** : Des capteurs de température surveillent la température des composants de l’électronique de puissance.
- **Diagnostic** : Des températures élevées peuvent indiquer des problèmes de refroidissement ou une dissipation excessive de puissance.

d) Défauts de commutation

- **Détection** : Surveillés par l'analyse du comportement de commutation des transistors et autres composants.
- **Diagnostic** : Des schémas de commutation irréguliers peuvent révéler des défauts dans l'électronique de puissance.

Il existe ainsi, des techniques avancées pour diagnostiquer PDU, cités ci-suivant :

e) Commande tolérante aux pannes (Fault-Tolerant Control)

- Met en œuvre des systèmes ou algorithmes redondants pour maintenir le fonctionnement malgré des défaillances.
- **Avantages** : Améliore la fiabilité en permettant au véhicule de continuer à fonctionner même si certains composants sont défectueux.

f) Surveillance et diagnostic en temps réel

- Surveille en continu les indicateurs de performance et les compare aux valeurs attendues.
- **Avantages** : Permet une détection et un diagnostic immédiats des pannes.

1.3.3.4 FDD pour la gestion thermique

Le système de gestion thermique (Figure 1.15) régule la température de la batterie, du moteur et d'autres composants critiques. Une gestion thermique efficace est essentielle pour garantir la performance et la sécurité du véhicule.

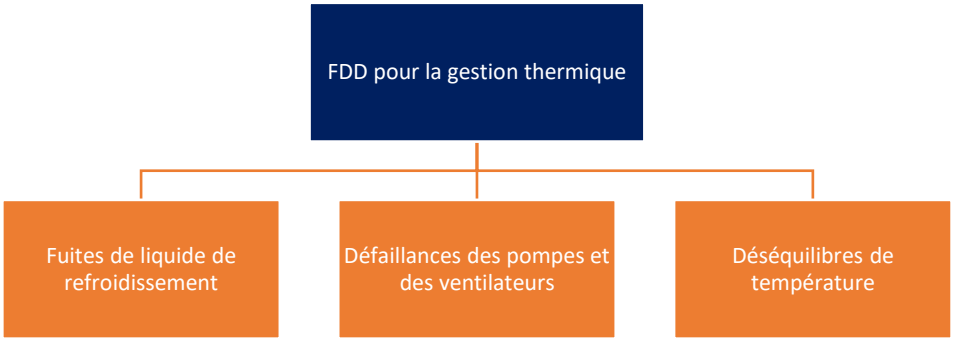


Figure 1.15 Méthodes de détection et du diagnostic pour la gestion thermique.

Ci-suivant nous allons présenter les défauts pour la gestion thermique du VE et leurs méthodes de diagnostic.

a) Fuites de liquide de refroidissement

- **Détection** : Surveillées à l'aide de capteurs et d'inspections visuelles.
- **Diagnostic** : Les fuites peuvent entraîner une défaillance du système de refroidissement et une surchauffe.

b) Défaillances des pompes et des ventilateurs

- **Détection** : Surveillance à l'aide de capteurs opérationnels et de mesures de débit.
- **Diagnostic** : Les défaillances des pompes ou des ventilateurs peuvent provoquer un refroidissement insuffisant.

e) Déséquilibres de température

- **Détection** : Capteurs de température répartis dans le système.
- **Diagnostic** : Des températures inégales peuvent indiquer des problèmes dans les composants de gestion thermique ou dans la circulation du fluide.

Il existe ainsi, des techniques avancées pour le diagnostic thermique, cités ci-suivant :

a) **Imagerie thermique**

Utilisation des caméras infrarouges pour détecter les variations de température et les problèmes potentiels. Ceci fournit une représentation visuelle de la distribution thermique et des anomalies.

b) **Commande prédictive basée sur des modèles**

Utilisation des modèles thermiques pour prévoir et gérer les besoins de refroidissement selon les conditions de fonctionnement. Ceci, améliore l'efficacité et la performance du système thermique.

1.3.3.5 FDD dans les protocoles de communication

Les VE reposent sur des protocoles de communication développés tel que le CAN (Controller Area Network) et le LIN (Local Interconnect Network) pour coordonner les différents sous-systèmes. Les pannes de communication peuvent perturber le fonctionnement du véhicule. Les FDD dans ces protocoles sont classifiés dans la Figure 1.16.

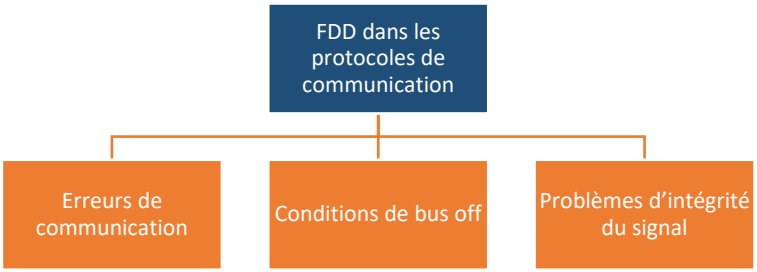


Figure 1.16 Méthodes de détection et du diagnostic dans les protocoles de communication.

Ci-suivant nous allons présenter les défauts pour diagnostiquer les protocoles du VE et leurs méthodes de diagnostic.

a) **Erreurs de communication**

- **Détection** : Surveillées à l'aide de protocoles de vérification d'erreurs et de contrôles d'intégrité des messages.
- **Diagnostic** : Ces erreurs peuvent indiquer des problèmes de réseau ou des composants défectueux.

b) **Conditions de bus off**

- **Détection** : Suivi de l'état du réseau CAN.
- **Diagnostic** : Une condition "bus off" signale des erreurs graves empêchant la communication normale.

c) **Problèmes d'intégrité du signal**

- **Détection** : Analyse par surveillance des signaux et outils de diagnostic.
- **Diagnostic** : Des problèmes d'intégrité peuvent indiquer des défauts de câblage ou de composants.

Il existe ainsi, des techniques avancées pour le diagnostic des protocoles, cités ci-suivant :

d) **Outils de surveillance du réseau**

Surveillent en continu le trafic réseau et les diagnostics. Ce qui permet une vision en temps réel de l'état du réseau de communication.

e) **Protocoles de communication tolérants aux fautes**

Intègrent la redondance et la correction d'erreurs dans les protocoles. Ceci, renforce la fiabilité et la robustesse des communications.

1.3.3.6 FDD dans les performances

Les performances dynamiques regroupent des paramètres essentiels tels que la vitesse du VE liée à la vitesse du moteur, la vitesse des roues, l'angle de braquage et le comportement au freinage. Toute anomalie dans ces paramètres peut impacter la sécurité, la stabilité et la maniabilité du véhicule électrique.

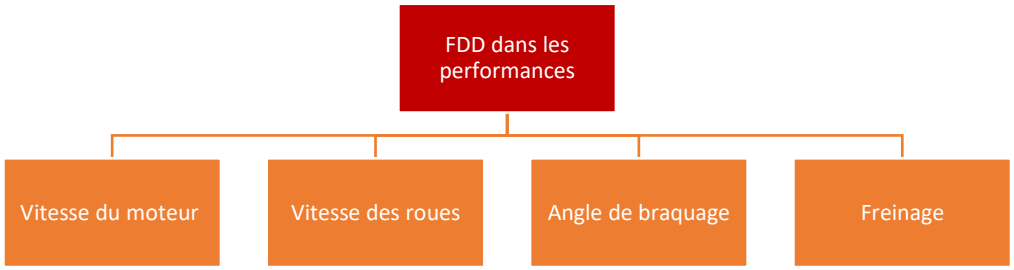


Figure 1.17 Méthodes de détection et du diagnostic dans les performances.

Ci-suivant nous allons présenter les défauts pour diagnostiquer les performances du VE et leurs méthodes de diagnostic.

a) Vitesse du moteur

Détection : Mesurée par des capteurs de rotation (capteurs à effet Hall, codeurs).

Diagnostic : Une incohérence entre la commande et la vitesse réelle peut signaler une panne de l'onduleur, une surcharge ou un dysfonctionnement du moteur.

b) Vitesse des roues

Détection : Captée par des capteurs ABS ou des capteurs de vitesse dédiés.

Diagnostic : Une différence anormale entre les roues peut indiquer un problème de transmission, un glissement ou une défaillance de capteur.

c) Angle de braquage (Steering angle)

Détection : Mesuré via des capteurs d'angle situés sur la colonne de direction.

Diagnostic : Une incohérence entre l'angle de braquage et la trajectoire du véhicule peut révéler une défaillance du système de direction assistée ou un problème électronique.

d) Freinage

Détection : Surveillance de la pression dans le circuit hydraulique et des capteurs de pédale de frein.

Diagnostic : Une réponse de freinage faible ou retardée peut indiquer une fuite, une défaillance du servofrein ou un défaut du système de régénération (frein moteur).

Il existe ainsi, des techniques avancées pour le diagnostic des performances, cités ci-suivant :

a) Fusion de données capteurs (Sensor fusion)

Combine les données des capteurs de vitesse, angle et freinage pour une évaluation cohérente de la dynamique du véhicule. Ce qui permet une détection précise des anomalies comportementales.

b) Surveillance adaptative en temps réel

Comparaison des performances dynamiques mesurées à des modèles de référence selon le contexte de conduite. Ce qui génère des alertes en cas de comportement inattendu, même sans panne franche.

c) Détection de dérive comportementale

Utilisation des algorithmes statistiques ou IA pour détecter des écarts progressifs de performance. Ce qui permet de prévoir les problèmes avant qu'ils ne compromettent la sécurité.

d) Intégration avec les systèmes ADAS (Advanced Driver Assistance Systems)

Les données de performance sont croisées avec les fonctions d'assistance à la conduite (ABS, ESC, freinage d'urgence). Ce qui renforce la précision du diagnostic et la sécurité globale.

3.4 Techniques et outils de détection et de diagnostic des pannes

Des techniques de détection générales sont utilisées pour identifier les problèmes dans différents systèmes du VE, incluant les composants matériels et logiciels. [16]

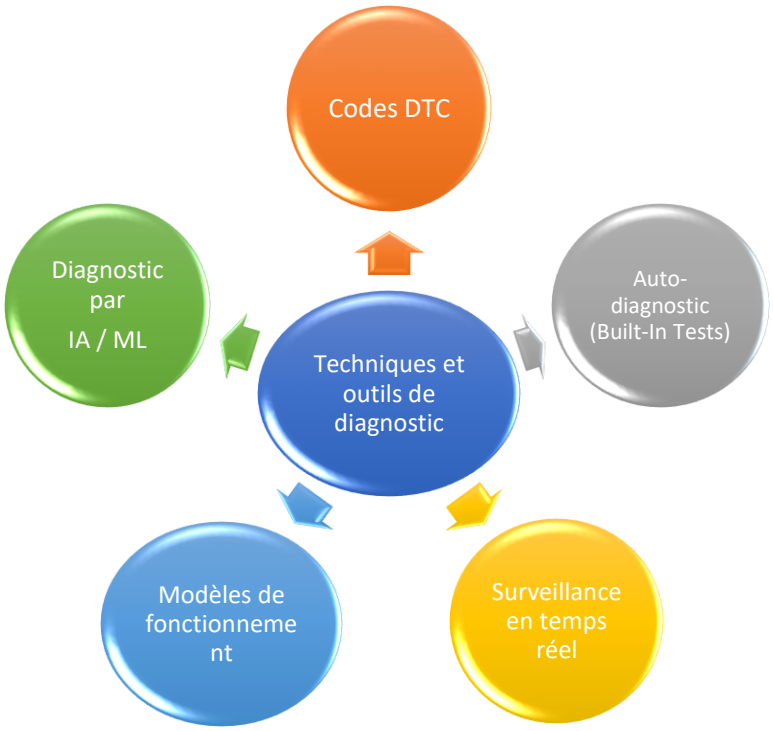


Figure 1.18 Techniques et outils de détection et de diagnostic des pannes

a) **Codes d'anomalie de diagnostic (DTC - Diagnostic Trouble Codes)**

- **Principe** : Utilise des codes normalisés pour signaler les pannes spécifiques détectées par le BMS ou d'autres systèmes.
- **Avantages** : Méthode simple et rapide pour identifier et résoudre les problèmes.

b) **Auto-diagnostic et tests intégrés (Built-In Tests)**

- **Principe** : Effectue périodiquement des tests internes pour vérifier l'état du système et détecter des défauts.
- **Avantages** : Permet de repérer des problèmes potentiels avant qu'ils ne deviennent critiques.

c) Surveillance en temps réel et enregistrement des données

- **Principe** : Surveille et enregistre en continu les données provenant de divers capteurs et composants.
- **Avantages** : Fournit des données historiques détaillées pour l'analyse des pannes.

d) **Diagnostic basé sur l'IA et le ML**

- **Principe** : Applique des algorithmes d'apprentissage automatique pour détecter des motifs et des anomalies dans les données.
- **Avantages** : Améliore la précision et s'adapte à de nouveaux types de pannes à partir des données historiques.

1.3.5. Limites des méthodes de diagnostics actuelles

Les méthodes actuelles de diagnostic embarqué pour les véhicules électriques ont permis des avancées significatives en matière de surveillance et de maintenance. Cependant, elles présentent encore plusieurs limitations qui restreignent leur efficacité et leur applicabilité en conditions réelles. Ci-suivant, les principales contraintes sont identifiées :

- Manque de standardisation et d'interopérabilité.
- Complexité accrue des systèmes embarqués.
- Qualité et disponibilité des données, où les méthodes avancées, celles basées sur l'IA, nécessitent de grands volumes de données labellisées et fiables.
- Temps de réponse élevé et risques de faux diagnostics.
- Maintenance et mise à jour des systèmes de diagnostic.
- Dépendance à des outils spécifiques et formation technique.

En effet, bien que les méthodes actuelles de diagnostic aient permis un progrès en matière de fiabilité et de surveillance des VEs, elles doivent encore surmonter de nombreux défis. Ces limitations constituent un moteur essentiel pour la recherche et le développement de solutions

de diagnostic plus intelligentes, plus rapides, plus flexibles et mieux intégrées aux architectures modernes des véhicules.

Conclusion

Ce chapitre a permis d'introduire les bases des systèmes embarqués impliqués dans les VEs. Nous avons vu l'importance croissante de l'électronique dans l'automobile, les composants clés d'un VE, et le rôle central du diagnostic dans la sûreté et la maintenance de ces véhicules. Les méthodes traditionnelles cèdent la place à des approches intelligentes, automatisées et connectées. Le chapitre suivant se penchera sur les protocoles de communication utilisés dans les systèmes de diagnostic-embarqués, en particulier le protocole CAN et les standards OBD-II.

Chapitre 2 : Protocoles de Communication & Outils de Diagnostic

2.1 Introduction

Les véhicules électriques (VE) modernes dépendent fortement des systèmes électroniques embarqués (SEE) pour un fonctionnement efficace et une sécurité accrue. Pour la transmission des informations entre ces SE, des protocoles de communications sont développés, parmi eux le Controller Area Network (CAN), qui est un réseau de communication robuste, efficace et permettant de réduire le câblage dans le VE. Ce protocole a été développé dans les années 1980, par la société BOSCH qui a développé cette solution de multiplexage des informations, circulant à bord de la voiture. Vu son importance ce chapitre présente le protocole CAN en détail, suivi du diagnostic embarqué l'OBDII.

2.2 Applications du protocole CAN

Plusieurs secteurs de l'électronique embarqué (véhicules, médicales, produits numériques, systèmes électrotechnique…), impliquent l'usage du protocole de communication CAN, comme le montre la Figure 2.1 [16]

Figure 2.1 : Différentes applications du CAN.

Le protocole CAN est utilisé dans de nombreux secteurs comme l'illustre la Figure 2.1, et donc dans chaque secteur il retrouve plusieurs usages et fonctionnalités tel que cités dans les points ci-dessous :

- **Véhicules** : Gestion moteur, ABS, airbags, transmission, communication entre ECU, diagnostic OBD-II.
- **Industriels** : Commande de machines, automates programmables industriels (API), robots, capteurs intelligents.
- **Médicales** : Appareils de diagnostic, radiologie, IRM, surveillance de patients.
- **Domotiques & internet des objets (IoT)** : Automatisation des bâtiments, gestion de la climatisation, chauffage, réseaux intelligents.

Ses domaines d'application s'étendent des réseaux moyens débits aux réseaux de multiplexages faibles coûts, grâce à leurs multiples privilèges tel que :

- Hiérarchisation des messages.
- Garantie des temps de latence.
- Souplesse de configuration.
- Réception de multiples sources avec synchronisation temporelle.
- Fonctionnement multi-maitres.
- Détections et signalisations des erreurs.
- Retransmission automatique des messages altérés dès que le bus est de nouveau au repos.
- Distinction d'erreurs : d'ordre temporaire ou de non-fonctionnalité permanente au niveau d'un nœud.
- Déconnexion des nœuds défectueux.

La mise en œuvre d'un protocole CAN permet aux constructeurs automobiles de réduire considérablement la quantité de câblage dans chaque véhicule (moins de 2Km de câblages). [17]

2.3 Présentation du protocole CAN

Le CAN (Control Area Network) est un protocole de communication série qui supporte des systèmes embarqués temps réel, avec un haut niveau de fiabilité. Il se charge de lier et raccorder plusieurs éléments constituant le VE tel qu'illustré dans la Figure 2.2.

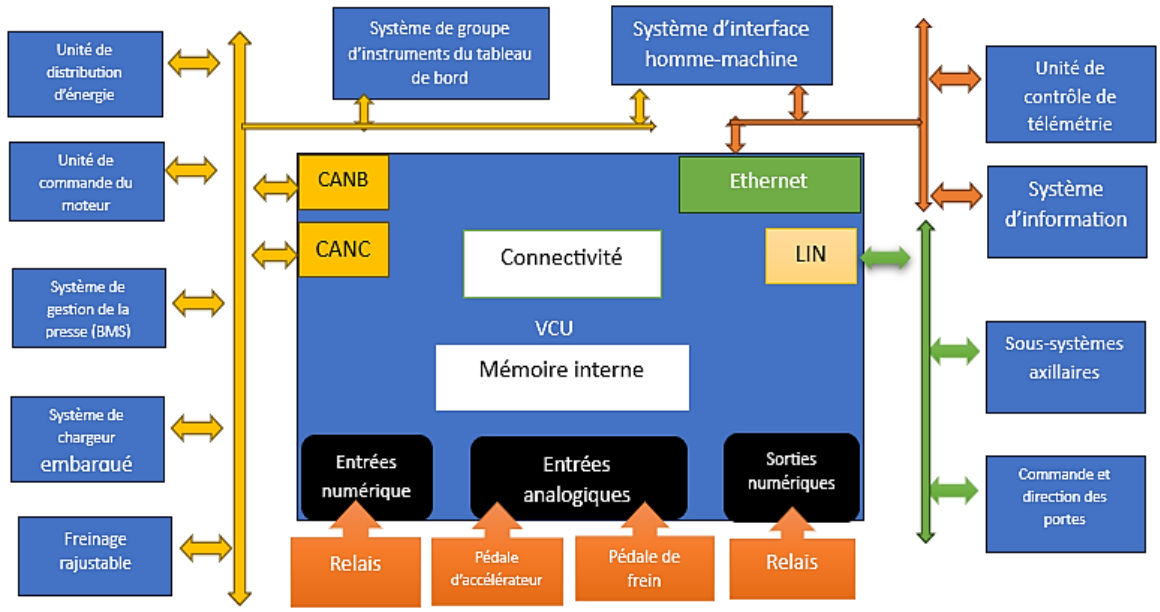


Figure 2.2 : Schéma synoptique de raccordement des éléments du VE avec les protocoles : CAN, LIN, & Ethernet.

En effet, plusieurs protocoles (CAN, LIN (Local Interconnecte Network), Ethernet) peuvent être utilisés dans le même VE, tout en dépendant de la catégorie de fonctionnalités, ayant des besoins en termes de fiabilités, de débits et du coût. Un réseau automobile peut être partagé en plusieurs sous-réseaux correspondant aux différents domaines de fonctionnalités. Ces familles de spécifications ont été classifiées par la SAE (Society for Automotive Engineers) tel que présente le Tableau 2.1, afin de déterminer les réseaux qui peuvent être adopté selon les besoins du système.

Tableau 2.1 : Classification réseaux selon la SAE.

Classe	Débit	Usage	Exemples
A	<10kb/s	Contrôle de l'habitacle	LIN
B	10kb/s → 125kb/s	Transfert de données non-critiques	CAN-B (Low-speed: LS)
C	125kb/s → 1Mb/s	Communications temps-réel critiques	CAN-C (High-speed: HS)
D	>1Mb/s	Multimédia ou X-by-wire	MOST, Flex-Ray

Les CAN sont comparables à Ethernet ou aux LIN qui fournissent une méthode standard pour la communication entre ordinateurs. Le bus LIN est une alternative économique qui utilise moins de faisceaux de câbles. Ce LIN offre des fonctionnalités pour les serrures de portes et la climatisation. L'Ethernet automobile prend en charge des débits de transfert de données plus

élevés qu'un bus CAN. Cette Ethernet prend en charge la bande passante accrue nécessaire pour mettre en œuvre des systèmes tels que les caméras embarquées, les systèmes avancés d'aide à la conduite (ADAS), un système de gestion de flotte et d'autres fonctionnalités nécessitant un transfert de données à haute vitesse. [18]

2.3.1 Historique du CAN

Ce type de communication CAN a été conçu et testé dans les années 1990 sur un modèle Mercedes. Puis, il a pris sa place dans la plupart des véhicules haute de gammes, pour être à ces jours l'un des réseaux les plus utilisés dans l'automobile. Ce type de réseaux a été normalisé en 1991 par la norme ISO 11898 et ISO11519 [REF]. La norme **ISO 11898** est la norme principale du bus CAN qui a été publiée en 1993, mais les travaux ont commencé dès les années 1980, notamment chez Bosch (le créateur du CAN). L'**ISO 11519** ont été publiées en 1994**,** qui est une norme pour une version basse vitesse du CAN, destinée à des applications moins critiques [19].

2.3.2 Caractéristiques techniques du CAN

Il en existe deux types : le bus CAN-B (Low-Speed (LS)) et le bus CAN-C (High-Speed (HS)). Le premier type contient un identifiant de 11 bits qui peut contenir jusqu'à 20 nœuds (système relié au bus) et peut atteindre un débit maximal de 125 kb/s. Le second type contient un identifiant de 29 bits qui peut contenir jusqu'à 30 nœuds et peut atteindre un débit maximal de 1 Mb/s. Nous nous intéresserons au deuxième type (CAN-B), même si les deux ont le même principe de fonctionnement avec seulement quelques différences. En effet, le protocole CAN repose sur une **communication série asynchrone** utilisant un modèle **différentiel** basé sur deux fils (paire filaire différentielle), qui sont le **CAN-High** et le **CAN-Low**, avec une résistance de terminaison de 120 Ohms dans chaque extrémité (Figure 2.3).

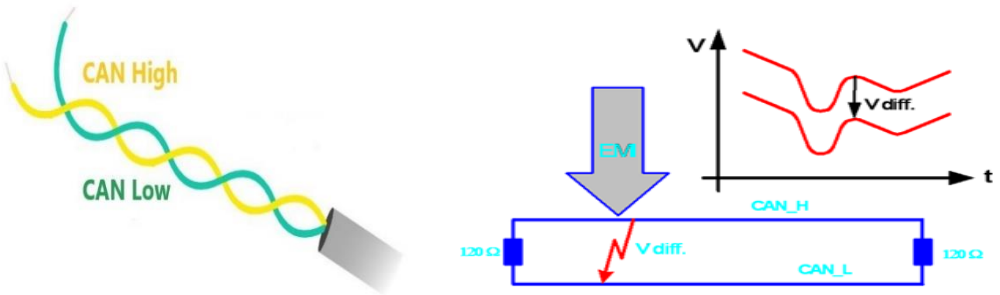


Figure 2.3 : Les deux modes du protocole CAN (HS & LS). [20]

Une ligne principale appelée (backbone) forme la fondation d'un système de communication CAN (Figure 2.4) dans le VE. Cette ligne connecte tous les microcontrôleurs d'un véhicule et fournit des informations à un contrôleur principal centralisé chargé de surveiller tous les systèmes électroniques du VE. Cette configuration simplifie l'identification des pannes potentielles et la résolution des erreurs sans interroger plusieurs sous-systèmes répartis dans un véhicule. Tous les éléments connectés au protocole CAN sont connus sous le nom d'unités de contrôle électronique (ECU) ou de nœuds (Nodes), tel qu'illustrés dans la Figure 2.4.

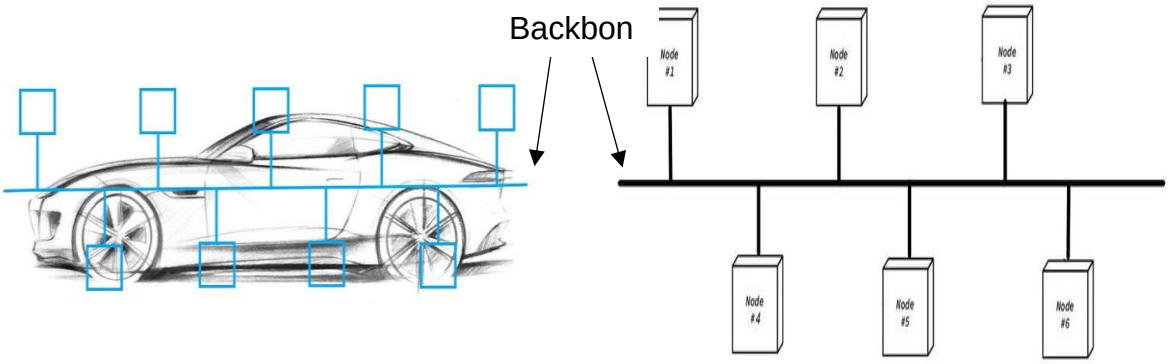


Figure 2.4 : Raccordements des nœuds (ECU) dans le VE via le CAN.[21]

Les ECU peuvent jouer divers rôles dans un système de communication CAN automobile. Les nœuds peuvent servir d'unités de contrôle pour le moteur, le freinage ABS, la boite de vitesse, avec le High-Speed (HS). Ils peuvent ainsi servir pour d'autres systèmes nécessaires au fonctionnement du véhicule, tel que les phares, la climatisation, les airbags et les sièges avec le Low-Speed (LS). Les voitures modernes peuvent avoir jusqu'à 70 ECU qui doivent transmettre et partager des données avec d'autres nœuds (Figure 2.5).

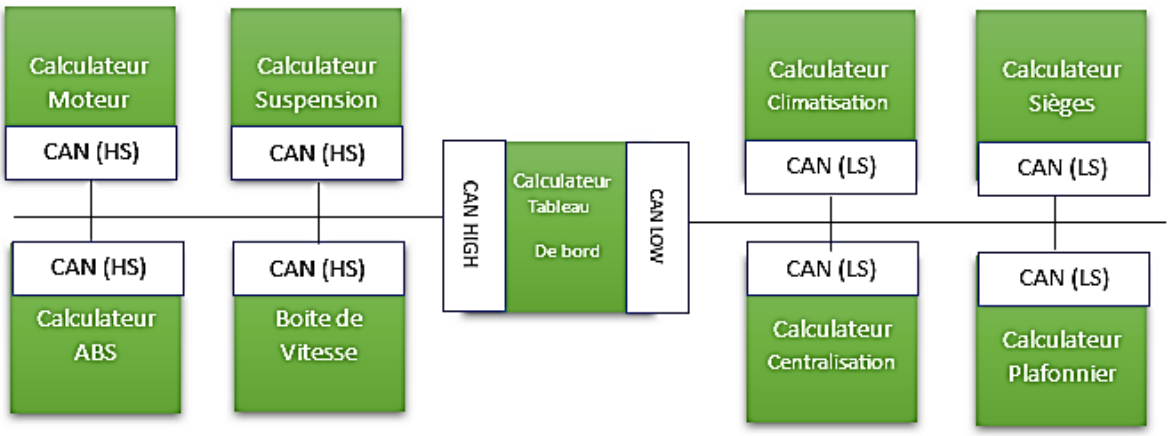


Figure 2.5 : Les principales ECU du VE utilisant le CAN (HS & LS). [22]

Chaque nœud (ECU) comprend au minimum un contrôleur CAN et un microcontrôleur intégré (Figure 2.6). Les données numériques sont converties en messages sur le bus de transfert par le contrôleur CAN. Ce dernier accepte les informations, les traduit et les envoie à un autre contrôleur CAN.

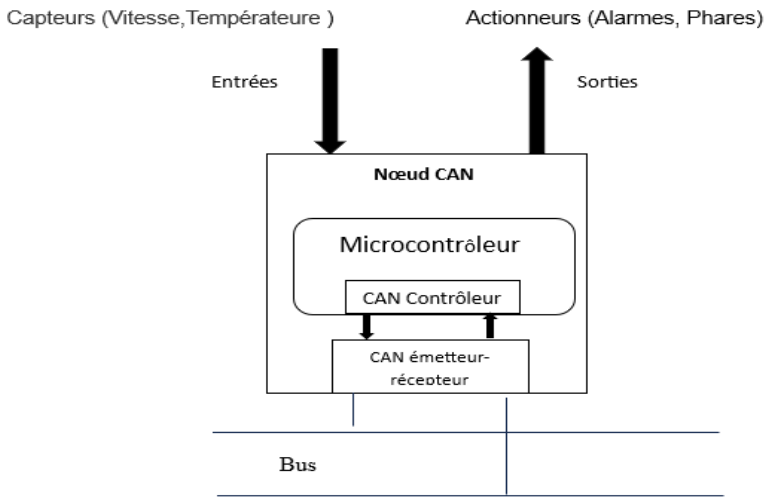


Figure 2.6 : Eléments constituants le nœud (ECU) du bus CAN. [23]

Deux boitiers (ou supercalculateurs) électroniques spécifiques à la communication du bus CAN, qui sont présents dans le bus CAN afin de traiter les informations contenues. Un premier boitier permet simplement d'encoder les informations (détection et émission) émises par les capteurs ou calculateurs selon le protocole de communication et de les envoyer au bus de communication. Un second boitier permet le décodage de ces informations (réception) pour les trier et les transmettre aux autres systèmes, tels que des calculateurs ou des actionneurs. Le microcontrôleur intégré traite les données et effectue des tâches telles que l'allumage d'une lumière dans la voiture ou l'abaissement d'une fenêtre. Les microcontrôleurs peuvent contrôler le flux d'informations vers le tableau de bord en réponse à un message généré par le contrôleur CAN. Un détail dans les sous-sections suivantes est présenté, relatif aux caractéristiques du bus CAN. [24]

a) Topologie du réseau CAN

Lors de l'utilisation du bus CAN, une seule ECU peut transmettre des données à toutes les autres ECU connectées au système. Ces ECU peuvent examiner les informations et choisir de les recevoir ou de les ignorer. Ainsi, la couche de liaison de données du bus CAN est décrite par la norme ISO 11898-1 avec la couche physique décrite par la norme ISO 11898-2. La couche physique d'un bus CAN est constituée de câbles, de l'impédance des câbles, des niveaux de

signal électrique, des exigences des nœuds et d'autres éléments nécessaires au fonctionnement du réseau. ISO 11898-2 décrit les spécifications des éléments de la couche physique tels que la longueur du câble, la terminaison du câble et le débit en bit par seconde. La Figure 2.7 présente quelques exemples de ces spécifications.

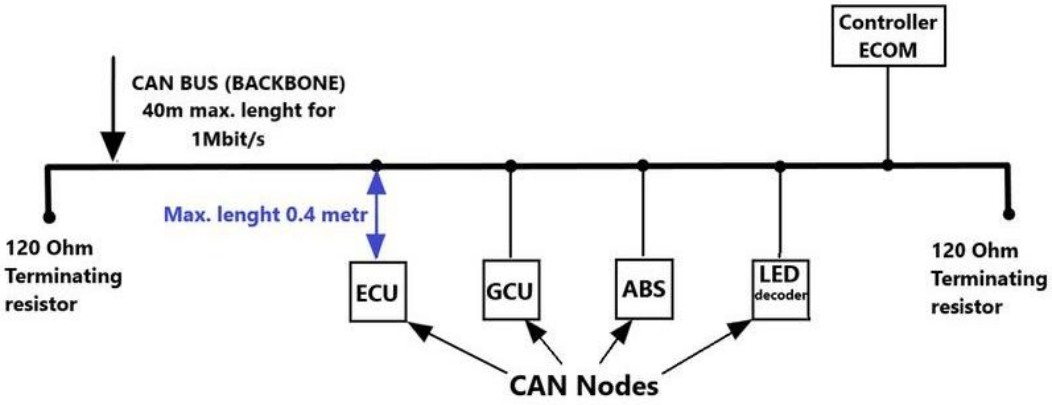


Figure 2.7 : Exemples de nœuds CAN.[21]

Les caractéristiques de cette topologie CAN peut être résumé dans les points ci-dessous :

- Architecture en bus avec des terminaux à chaque extrémité du réseau.
- Résistances de 120Ω aux extrémités du bus CAN pour éviter les réflexions de signal.
- Peut supporter jusqu'à 127 nœuds sur un même réseau.

Les nœuds CAN nécessitent des connexions utilisant deux fils avec des débits en bits par seconde allant jusqu'à 1 Mbit/s (CAN) ou 5 Mbit/s (CAN FD). La longueur du câble pour un bus CAN peut être de 40 mètres pour 1 Mbit/s ou de 500 mètres pour 125 Kbit/s.

b) Modes de transmission

Le protocole CAN supporte plusieurs modes de transmission :

- **Mode normal** : Émission et réception de messages.
- **Mode veille (Sleep-Mode)** : Consommation d'énergie réduite.
- **Mode écoute seule (Listen Only Mode)** : Observation du réseau sans interagir.
- **Mode boucle interne (Loopback Mode)** : Tests internes sans perturbation du réseau.

c) Vitesse de communication

- **125 kbps** (réseau long, ex. : poids lourds, industrie).

- **500 kbps** (véhicules classiques, systèmes embarqués).
- **1 Mbps** (courtes distances, applications critiques).
- **8 Mbps (CAN FD (Flexible Data-Rate))**.

d) Fonctionnement

Les capteurs installés autour du véhicule surveillent en permanence l'état du VE (Figure 2.8) et envoient les données aux unités de contrôle respectives. Par exemple, les données relatives à la pression atmosphérique, à la température du moteur et à la vitesse du moteur recueilli par les capteurs parviennent à l'unité de contrôle du moteur qui, après analyse et traite, envoie des commandes pour contrôler la quantité de carburant à injecter.

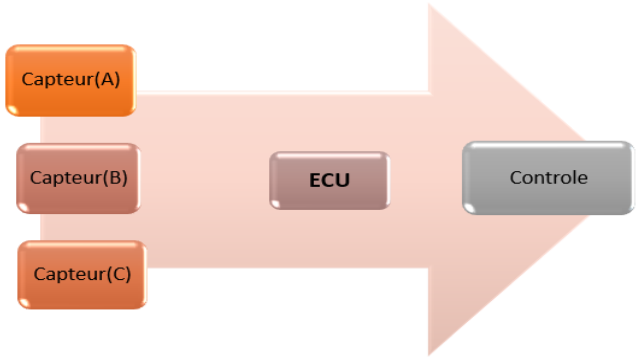


Figure 2.8 : Exemple d'un fonctionnement du protocole CAN.

En général, la communication entre les calculateurs s'effectue de deux manières. Premièrement, chaque message est transmis par des câbles indépendants. Par exemple, si 5 signaux doivent être échangés entre deux unités de contrôle, 5 câbles indépendants sont nécessaires. Plus de messages signifient plus de câbles et plus de broches entre les calculateurs. Deuxièmement, tous les messages entre les calculateurs sont transmis par deux câbles (Figure 2.9). Ainsi, tous les messages, quelle que soit leur taille, peuvent circuler dans les deux câbles. [25]

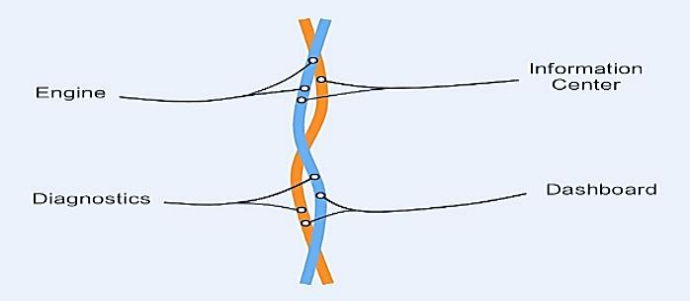


Figure 2.9 : Transfert de messages dans un bus CAN.

2.3.3 Structure d'un message CAN

Le protocole CAN fonctionne avec des trames de données qui contiennent plusieurs champs tels que la montre la Figure 2.10.

Figure 2.10 : Constitution des champs d'une trame de donnée du protocole CAN. [26]

Le Tableau 2.2, montre un détail de la constitution des champs de la trame de données CAN.

Tableau 2.2 : Détails de chaque champ constituant la trame de données CAN.

Champ	**Description**
Start of Frame (SOF)	Indique le début de la trame.
Identifiant (ID)	Définit la priorité du message (11 ou 29 bits).
Remote Transmission Request (RTR)	Indique si c'est une trame de données ou de requête.
Commande	Indique la taille des données (DLC).
Données	Contient les informations (0 à 8 octets en CAN-Standard, jusqu'à 64 en CAN-FD).
Cyclic Redundancy Check (CRC)	Vérifie l'intégrité des données.
Acknowledgment (ACK)	Confirmation de réception.
End of Frame (EOF)	Marque la fin de la trame.

Il existe quatre principaux types de trames pour le protocole CAN :

- **Trame de données (Data Frame)** : Contient les informations utiles.
- **Trame de requête (Remote Frame)** : Demande d'information à un autre nœud.
- **Trame d'erreur (Error Frame)** : Signalisation d'une erreur détectée.
- **Trame de surcharge (Overload Frame)** : Indique un dépassement de capacité.

Le bus CAN fonctionne avec un arbitrage des messages qui permet le passage de priorité, basé sur l'identifiant du message (ID) tel que :

- Plus l'ID est petit, plus la priorité est haute.
- Un arbitrage non destructif : les messages de priorité inférieure attendent leur tour sans avoir perturber les autres transmissions. [27]

2.3.4 Versions du protocole CAN

Il existe trois versions du protocole CAN, qui sont **:**

a) CAN 2.0

- **CAN 2.0A** : 11 bits d'identifiant.
- **CAN 2.0B** : 29 bits d'identifiant (extension du standard).

b) CAN-FD (Flexible Data-Rate)

- Développé pour améliorer la vitesse et la taille des données, qui remplace le CAN-traditionnel dans les prochaines générations.
- Capacité augmentée jusqu'à 64 octets par trame.
- Vitesse jusqu'à 8 Mbps.

c) CAN-XL

- Dernière évolution du CAN.
- Jusqu'à 2048 octets par trame.
- Conçu pour des communications plus complexes, notamment dans les VE. [27]

d) CAN-Open [28]

2.3.5 Mécanismes de détection et correction des erreurs

Le bus CAN est reconnu pour ses forts mécanismes de sécurité, permettant de détecter les erreurs détectées suivants :

- **Erreur de bit** : Si un nœud détecte une incohérence dans les bits transmis.
- **Erreur de trame** : Si un champ de la trame est corrompu.
- **Erreur d'acquittement (ACK)** : Si un message n'est pas reçu.
- **Erreur CRC** : Si la vérification de redondance cyclique échoue.
- **Erreur de format** : Si la structure de la trame est invalide.

2.4 Différence entre CAN-Traditionnel et CAN-FD

Le Tableau 2.3, montre une comparaison entre le CAN-Traditionnel et le CAN-FD.

Tableau 2.3 : Comparaison entre le CAN-Traditionnel et le CAN-FD.

Critère	CAN-Traditional (CAN 2.0A/B)	CAN-FD (Flexible Data-Rate)
Taille des données	**8 octets** max par trame	Jusqu'à **64 octets** par trame
Vitesse de transmission	Max **1 Mbps**	Jusqu'à **8 Mbps**
Fiabilité et Sécurité	CRC **15 bits**	CRC amélioré (**17 ou 21 bits**) pour une meilleure détection d'erreur
Efficacité du bus	**Plus de trames** nécessaires pour transmettre de grandes quantités de données	**Moins de trames**, donc meilleure utilisation du bus
Latence	Plus élevée (plus de trames = plus de temps)	Réduite grâce à une transmission plus rapide
Compatibilité	Standard et largement utilisé	Compatible avec CAN classique, mais nécessite des **contrôleurs CAN-FD**
Consommation d'énergie	Optimisée pour les petites communications	Peut-être plus élevée à haute vitesse
Applications	Automobile, automatisation industrielle, capteurs de base	Véhicules modernes, ADAS, IoT, aviation, robotique avancée

- Le CAN-FD est clairement supérieur en termes de vitesse, efficacité et capacité de données. Il est idéal pour les véhicules modernes, les systèmes avancés et les applications nécessitant un transfert rapide de données.
- Le CAN-Traditionnel reste une option fiable pour les applications moins exigeantes, avec une infrastructure existante et un coût plus faible [29].

2.5 Acquisition, Filtrage & Décodage des Trames CAN

L'acquisition, le filtrage et décodage des trames CAN sont essentiels pour optimiser la communication et extraire les données utiles dans un réseau CAN. Ils sont présentés dans les sous-sections suivantes.

2.5.1 Acquisition des Trames CAN

L'acquisition des trames CAN consiste à capturer les messages circulant dans le bus CAN à l'aide d'outils matériels et logiciels appropriés, qui peuvent être cités ci-après.

- **Interface CAN** : Dispositif permettant de connecter le bus CAN à un ordinateur, souvent via USB**.**
- **Transmetteur CAN** : Convertit les signaux logiques en signaux physiques sur le bus.

- **Câblage CAN :** Le bus CAN utilise généralement une paire torsadée pour les lignes CAN_H et CAN_L.
- Des interfaces telles que celles proposées par *National-Instruments* (par exemple, la série NI-XNET) ou d'autres fabricants comme *Peak* ou *Kvaser* sont couramment utilisées**.**
- **Data-loggueurs CAN** (CANedge, CLX000, CANmod). [30]

Plusieurs logiciels permettent de visualiser et d'analyser les trames CAN :

- **NI LabVIEW** : Avec les modules NI-CAN ou NI-XNET pour une intégration complète [30].
- **CAN Analyzer** : Outil professionnel pour l'analyse des réseaux CAN. [30]
- **Bus Master** : Logiciel open-source pour la surveillance et l'analyse du bus CAN. [30]
- **Wireshark** : Avec des plugins spécifiques pour le CAN. [30]

2.5.2 Décodage des Trames CAN

Après avoir collecté les données CAN tel que discuté dans la section précédente. Un format standardiser pour ces données est reconnu sous le format DBC, contenant les trames CAN (Figure 2.11).

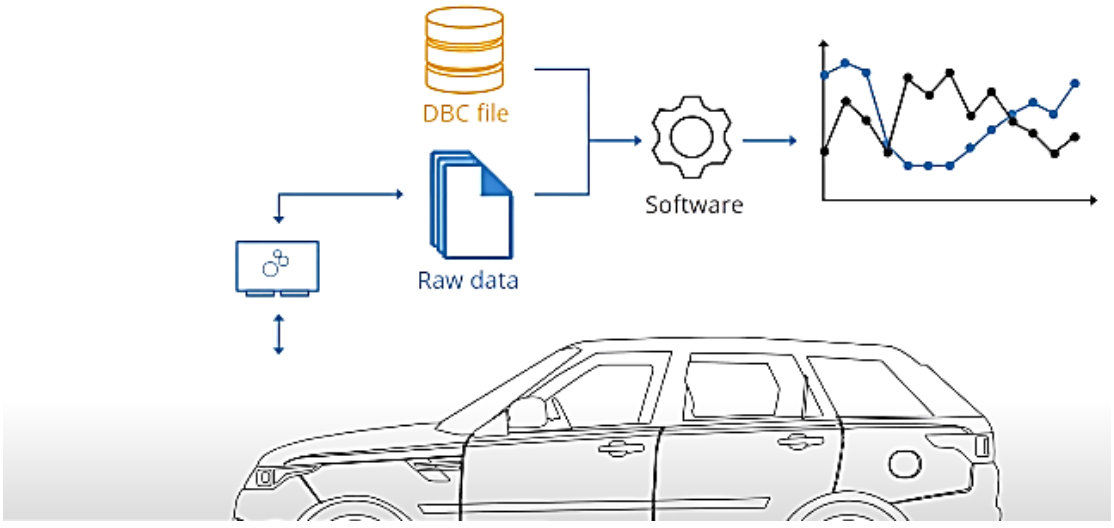


Figure 2.11 : Collecte de données CAN en fichier de format DBC.

Ces fichiers DBC (Figure 2.12), contient des données CAN qui ne peuvent être traités directement par l'usage humain, ce qui nécessite un décodage (scaled) en des valeurs physiques (vitesse du VE est en km/h) [31].

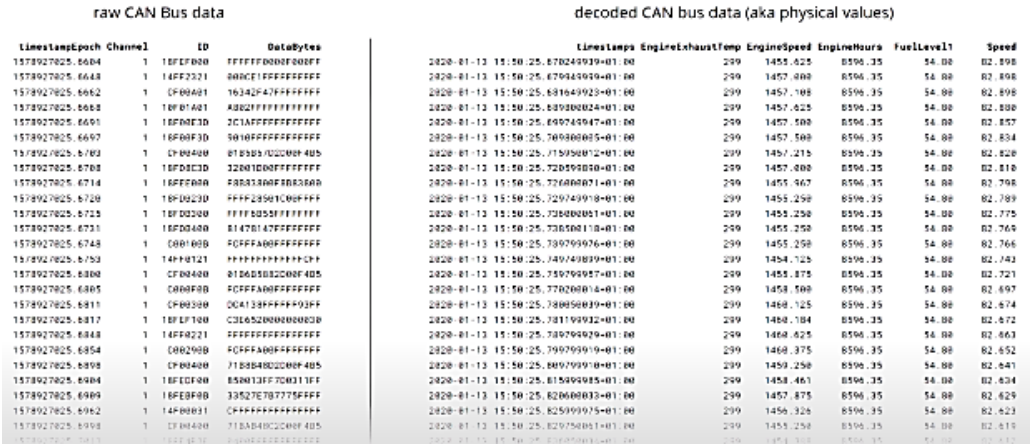


Figure 2.12 : Décodage de données CAN en des valeurs physiques. [32]

Les trames CAN de la Figure.13, contient un nombre particulier de signaux (paramètres) dans les bits data du CAN.

Figure 2.13 : Exemple d'une trame CAN contenant deux signaux avec un seul CAN-ID. [33]

En pratique, l'information recherchée pour extraire les signaux du bus CAN. Une trame CAN contient plusieurs champs qu'il faut analyser pour en extraire les données.

a) Exemple de Trame CAN (hexadécimal)

- Exemple d'une trame reçue en CAN-classique :

0x18DAF110 8 02 10 03 FF FF FF FF FF

Décryptage :

- **ID = 0x18DAF110** → Message provenant de l'ECU moteur.
- **DLC = 8** → 8 octets de données.
- **Données = 02 10 03 FF FF FF FF FF**
  - **02** → Nombre d'octets valides.
  - **10 03** → Message "Mode Diagnostic".

Applications Pratiques :

-Débogage des communications entre ECU.

- Analyse des trames pour le reverse-engineering automobile.

- Surveillance des capteurs dans l'automatisation industrielle.

Ces outils permettent de décoder les trames, d'analyser les erreurs et de surveiller le trafic en temps réel.

2.5.3 Filtrage des Trames CAN

Le filtrage permet de réduire la charge du processeur en ne recevant que les trames pertinentes. En effet, il permet d'éviter la surcharge du microcontrôleur en ignorant les trames non pertinentes. Ainsi, de gérer efficacement un réseau avec de nombreuses ECUs. Améliorer les performances en ne traitant que les données utiles. Il existe deux types principaux pour le filtrage CAN présentés ci-après.

a) Filtrage par identifiant (ID-CAN)

- Chaque trame CAN a un identifiant unique (11 ou 29 bits).
- Un contrôleur CAN peut être configuré pour accepter uniquement certaines plages d'ID.

b) Filtrage par masque et filtre

- Le masque définit quels bits de l'ID doivent être pris en compte.
- Le filtre spécifie les valeurs attendues pour ces bits.

c) Filtrage par contenu des données

- Certains microcontrôleurs avancés permettent de filtrer sur le contenu des données.

Exemple : n'accepter que les messages où le capteur de température dépasse 50°C. [34]

2.6 Outils de diagnostic

Certains protocoles populaires ont été développés pour fournir des fonctionnalités automobiles avancées. Ces protocoles sont cités comme ci-suivant :

- Diagnostic embarqué reconnu par On-Board-Diagnosis (OBD, OBD-II), qui offre aux mécaniciens des fonctionnalités de diagnostic et de rapport pour gagner du temps lors de l'identification des problèmes dans les véhicules.
- J1939 est un protocole standard pour les véhicules lourds comme les camions et les bus.
- CAN-FD étend la couche de liaison de données CAN originale, qui permet une charge utile accrue de 8 à 64 octets (structure de trame CAN dans Figure 2.10). Il peut également fournir des débits plus élevés en fonction du transmetteur CAN utilisé.
- CAN-Open est mis en œuvre dans les applications qui disposent de contrôles embarqués tels que les installations d'automatisation industrielle.

Dans ce qui suit, un détail est concentré au protocole de diagnostic embarqué pour VE, l'OBDII.

2.6.1 Système de diagnostic OBD

Le système de diagnostic embarqué, ou OBD est un ensemble de dispositifs de diagnostic qui est embarqué dans la plupart des véhicules à moteur thermique produits dans les années 2000. L'OBD décrit les moyens à mettre en œuvre pour contrôler l'ensemble des composants du groupe motopropulseur affectant les émissions polluantes du véhicule tout au long de sa vie.

L'OBD-II est un protocole de diagnostic standardisé par l'ISO 15765 [35], utilisé dans les véhicules légers (voitures, utilitaires), relié au protocole CAN (Figure 2.11). Il permet de lire les codes d'erreurs, les paramètres moteurs, et de contrôler l'état de fonctionnement du véhicule, particulièrement électrique.

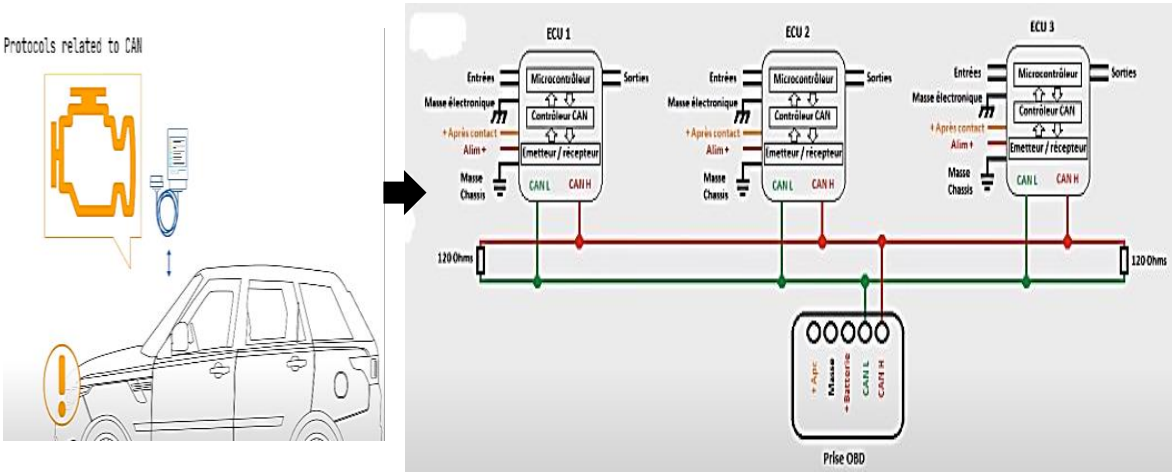


Figure 2.14: Liaison bus CAN avec l'OBD. [36]

En effet, OBDII est le système d'autodiagnostic intégré dans le véhicule. Un voyant de dysfonctionnement permet d'indiquer aux utilisateurs, le traitement OBD sur le tableau de bord des véhicules. Un mécanicien, utilisera un scanner OBDII pour diagnostiquer les problèmes qui peuvent être rencontrés dans un véhicule (Chapitre 1). Pour ce faire, il connectera le lecteur OBDII au connecteur OBDII (Figure 2.12) qui contient 16 broches près du volant (au-dessous). Cela lui permet de lire les codes OBDII, c'est-à-dire les codes de diagnostic reconnu par *Diagnosis-Trouble-Codes* (DTC), pour examiner et résoudre le problème.

Figure 2.15: Connecteur OBDII [37]

Le connecteur OBDII permet d'accéder facilement aux données du véhicule. Standardisé à 16 broches (DLC) et situé sous le tableau de bord côté conducteur. Il utilise le protocole CAN / ISO 15765-4. La norme *Society of Automotive Engineers* (SAE) de format J1962 spécifie deux types de connecteurs OBDII femelles à 16 broches, A et B présentés en Figure 2.13. L'illustration présente un exemple de connecteur à broches OBDII de type A (également appelé connecteur de liaison de données, DLC).[37]

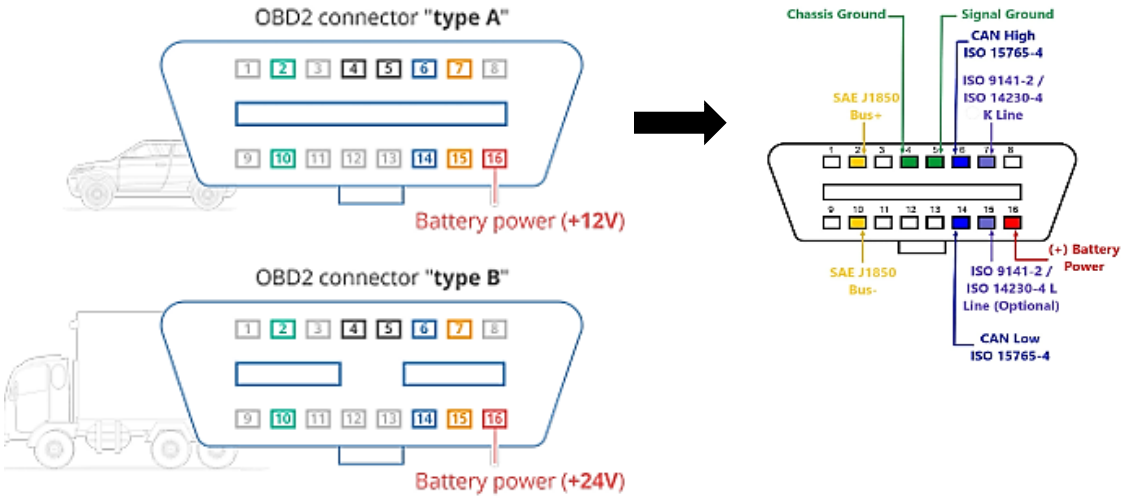


Figure 2.16 Pins du connecteur OBDII : type A & B.

Comme le montre l'illustration de la Figure 2.13, les deux types partagent des brochages OBDII similaires, mais fournissent deux sorties d'alimentation différentes (12 V pour le type A et 24 V pour le type B). Souvent, le taux de transmission diffère également, les voitures utilisant généralement 500K, tandis que la plupart des véhicules lourds utilisent 250K (plus récemment avec la prise en charge de 500K). Pour aider à distinguer physiquement les deux types de prises OBDII, le connecteur OBDII de type B a une rainure interrompue au milieu. Par conséquent, un câble adaptateur OBDII de type B sera compatible avec les types A et B, tandis qu'un câble adaptateur de type A ne rentrera pas dans une prise de type B. [38]

En effet, le protocole CAN sert aujourd'hui de base à la communication OBDII dans la grande majorité des voitures grâce à la norme ISO 15765. Cependant, pour une voiture plus ancienne (avant 2008), il est utile de connaître les quatre autres protocoles qui ont été utilisés comme base pour OBDII. Il est à noter que le format standardisé pour les données OBD est celui sous le format DBC, tel que celui du CAN (présenté en Section 2.4.2).

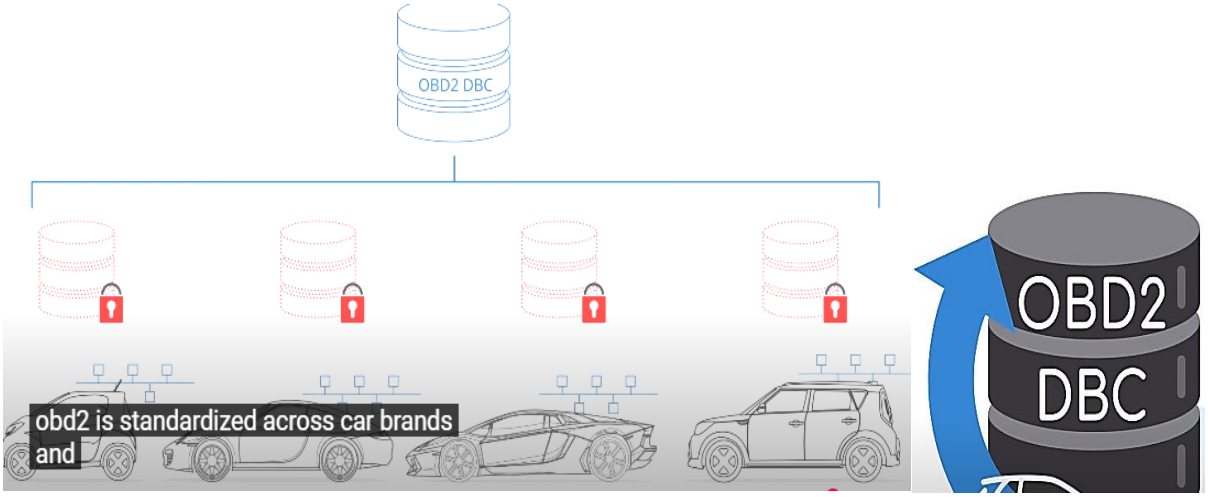


Figure 2.17 Format des données OBDII le DBC. [39]

a) Caractéristiques :

- Introduit dans les années 1990 (obligatoire aux USA dès 1996).
- Utilise le protocole CAN (ou autres protocoles comme ISO 9141, KWP2000, selon les véhicules), à 250 kbps ou 500 kbps.
- Connecteur standardisé à 16 broches (DLC) situé sous le tableau de bord.
- Lecture des codes d'erreurs (Diagnostic Trouble Codes (DTC)).
- Accès à des PID (Paramètre IDs) pour lire : température moteur, régime moteur, vitesse, etc.
- Messages structurés en PGN (Paramètre Group Numbers) pour identifier les données.
- Adresse chaque module via un identifiant logique (ECU Adresse).
- Prise en charge de messages longs grâce au protocole BAM/RTS/CTS.
- Développée par la SAE (Society of Automotive Engainées).
- Diagnostic rapide et standardisé pour tous les véhicules compatibles.

- Utilisé par les scanners OBD, les applications mobiles (Torque, Car Scanner...) et les garagistes. [40]

b) Applications :

- Diagnostic moteur et anti-pollution.
- Lecture des capteurs moteurs : température, régime, oxygène, débit d'air...
- Utilisé par les garagistes, applications smartphone (Torque, OBD-Link…).
- Surveillance en temps réel via dongles, *Bluetooth* ou *WiFi*.
- Contrôle de conformité des émissions. [40]

c) Utilisations :

- Contrôle moteur, transmission, ABS, freins, éclairage...
- Communication entre remorque et tracteur.
- Surveillance des performances, maintenance préventive, télémétrie.
- Communication entre plusieurs calculateurs dans les poids lourds.
- Utilisé dans les bus, camions, machines agricoles, engins de chantier.[40]

2.6.2 Paramètres de Diagnostic (PID – Parameter ID)

Un PID (Parameter-Identifier) est un code standard qui permet d'interroger un véhicule compatible OBD-II pour obtenir des informations en temps réel sur son fonctionnement. Ce dernier, a une liaison avec la température du moteur, le régime moteur, une panne. Un exemple de PID (Mode 01) de l'OBDII est présenté dans la Figure 2.14. [41]

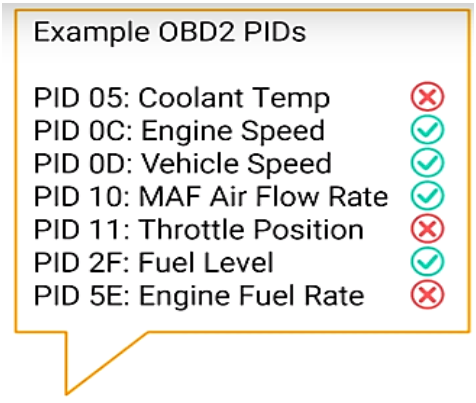


Figure 2.18 Exemple d'un affichage de PID de l'OBDII.

Le Tableau 2.4 Présente une liste des PID (Parameter ID).

Tableau 2.4 : Liste des paramètres PID de l'OBD.

PID	Fonction	Unité / Calcul
0C	Régime moteur	(A×256 + B) / 4 (en tr/min)
0D	Vitesse du véhicule	A (en km/h)
05	Température du liquide de refroid.	A - 40 (en °C)
11	Position du papillon	(A×100) / 255 (%)
10	Débit d'air massique (MAF)	(A×256 + B) / 100 (g/s)

2.6.3 Diagnostic Trouble Code (DTC)

Les DTC sont des codes d'erreurs qui indiquent des anomalies ou de défaillances détectées par l'ECU. Le format d'un DTC est telque:

- Pxxxx : Chaîne de traction (Powertrain)
- Bxxxx : Carrosserie (Body)
- Cxxxx : Châssis (Chassis)
- Uxxxx : Réseau de communication (Network)

Exemples :

- P0300 → Ratés d'allumage aléatoires
- P0420 → Catalyseur inefficace
- U0101 → Perte de communication avec TCU
- P0171 → Mélange pauvre (banque 1)
- U0100 → Perte de communication avec ECU moteur

Ces codes peuvent être lus/supprimés via un lecteur OBD-II. [42]

2.6.4 Communication avec l'ECU via OBD-II

L'ECU (Engine Control Unit) est le cerveau du véhicule. Grâce à l'interface OBD-II, il est possible de :

- Lire les PID en temps réel
- Lire / Effacer les DTC
- Suivre les performances du véhicule
- Utiliser des outils comme ELM327, OBDLink, Torque app. [43]

2.6.5 Normes et protocoles

Les diagnostics sont encadrés par des standards comme :

- **OBD-II** (On Board Diagnostics) : obligatoire sur les véhicules particuliers pour le contrôle des émissions.
- **J1939** : protocole basé sur CAN, utilisé dans les véhicules industriels.
- **ISO 11898** : définit le protocole CAN lui-même.
- **ISO15765** (CAN bus) : Obligatoire dans les voitures américaines depuis 2008 et est aujourd'hui utilisé dans la grande majorité des voitures
- **ISO14230-4** (KWP2000) : Le protocole de mots-clés 2000 était un protocole commun pour les voitures 2003+ par exemple en Asie
- **ISO9141-2** : Utilisé dans les voitures de l'UE, Chrysler et asiatiques en 2000-04
- **SAEJ1850** (VPW) : Utilisé principalement dans les voitures GM plus anciennes
- **SAEJ1850** (PWM) : Utilisé principalement dans les vieilles voitures Ford

2.6.6 Systèmes de Diagnostic Embarqués Existants

Voici quelques outils de diagnostic disponibles sur le marché :

- **Bosch KTS** : valise professionnelle très utilisée dans les garages.

Figure 2.19 : Bosch KTS. [44]

- **Launch X431** : outil multifonction compatible avec plusieurs marques.

Figure 2.20: Launch X431. [45]

- **ELM327** : adaptateur Bluetooth très répandu chez les particuliers.

Figure 2.21 : ELM327. [46]

- **OBDLink MX+** : version améliorée avec support étendu pour les PIDs.

Figure 2.22: OBDLink MX+. [47]

2.6.7 Technologies de collecte et traitement des données

Les véhicules électriques disposent de capteurs intégrés qui mesurent :

- Température de la batterie.
- Tension et courant de charge/décharge.
- Vitesse du moteur.
- Niveau de charge (State Of Charge (SOC)).

Les données sont recueillies via le bus CAN et transmises aux ECUs, qui les interprètent selon des algorithmes embarqués. Les systèmes modernes intègrent également des outils de visualisation (affichage intelligent HMI, écran Nextion, applications mobiles) pour afficher les paramètres en temps réel.

2.6 Conclusion

Ce chapitre a présenté les protocoles de communication, en particulier le CAN (Controller Area Network), ainsi que les outils de diagnostic tels que l'OBD-II. Le CAN est désormais un standard dans les véhicules modernes, permettant une communication robuste entre les unités de contrôle électronique. Il est également utilisé dans d'autres secteurs comme la fabrication et la santé. L'OBD-II est un standard crucial pour la maintenance du véhicule, permettant la lecture des codes d'erreur et la surveillance des paramètres en temps réel. Des équipements de diagnostic tels que le Bosch KTS ou l'ELM327 sont utilisés pour améliorer la durabilité et le bon fonctionnement du véhicule. Ces technologies ne cessent d'évoluer et jouent un rôle central dans la modernisation des systèmes embarqués.

Chapitre 3 : Traitement par IA de la BDD de l'OBDII-CAN

3.1 Introduction

L'Intelligence Artificielle (IA) englobe plusieurs disciplines, tel que l'apprentissage automatique (Machine Learning-ML) et l'apprentissage profond (Deep Learning-DL), illustrés dans la Figure 3.1.

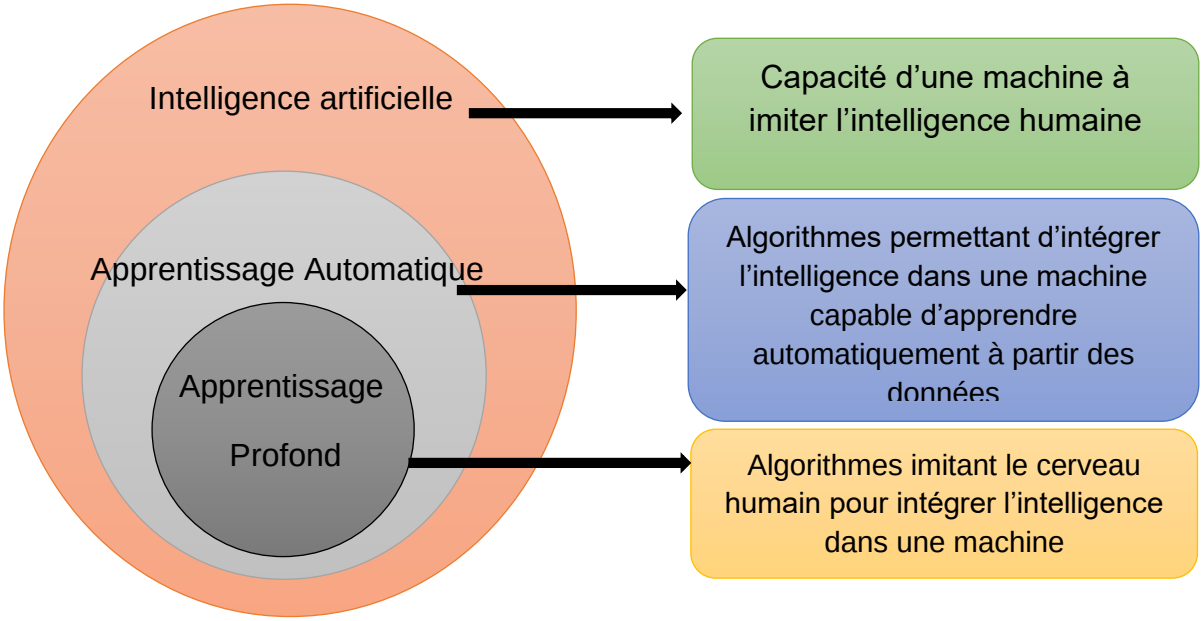


Figure 3.1 : IA, ML & DL.

Les différentes disciplines de la Figure 3.1, sont présentés dans les sous-sections suivantes.

3.1.1 : Intelligence Artificielle (IA)

L'IA (en anglais, Artificial Intelligence (AI)) est un domaine de l'informatique qui cherche à créer des machines capables d'imiter l'intelligence : apprendre, comprendre, raisonner, résoudre des problèmes, prendre des décisions, etc., qui contient le ML et le DL .[48]

3.1.2 Machine Learning (ML)

Le Machine Learning est l'Apprentissage automatique, qui est une branche de l'IA qui permet à une machine d'apprendre à partir de données sans être programmée directement pour

chaque tâche. Plusieurs algorithmes émergent de cette branche, tels que les arbres de décisions aléatoires ou les machines à vecteurs de support (SVM) pour classer les états du système.

3.1.3 Deep Learning (DL)

Le Deep Learning, c'est l'apprentissage profond qui est une sous-branche de l'apprentissage automatique reposant sur l'utilisation des réseaux de neurones artificiels multicouches (voilà de l'appellation "profond").

Le DL imite la cognition cérébrale pour manipuler des informations complexes qui permet de retrouver automatiquement des caractéristiques aux niveaux divers d'abstraction. Capables de détecter des défauts même sans modèle physique. Utilise des réseaux de neurones profonds (CNN, LSTM). [49]

3.2 Traitement par IA/ML

Le traitement par ML passe par des étapes principales, mentionnés ci-après :

a) La récupération d'une base de données (BDD).
b) Choix d'une méthode intelligente (IA/ML/DL) pour le diagnostic.
c) Développement de l'algorithme de la méthode.
d) Evaluation des performances de la méthode.

Dans ce qui suit, nous allons détailler les étapes précédentes du traitement par ML.

3.2.1 BDD de l'OBDII-CAN

Les bases de données (BDD) disponible sur internet sont plusieurs, et où on a choisi d'étudier celle relative à la vitesse de rotation d'un moteur électrique et la vitesse du VE, qui est disponible sur le site de [50]. La collecte d'une BDD est représentée dans le schéma de la Figure 3.2.

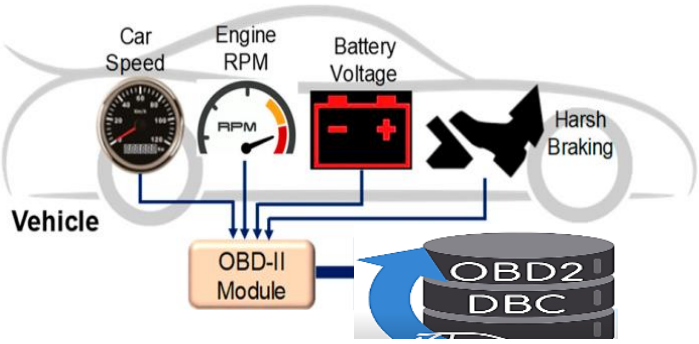


Figure 3.2 : Schéma de récolte de BDD base sur CAN-OBDII.

Cette BDD de l'OBDII-CAN contient des enregistrements pendant 4 jours (22,23,24 & 26) du mois d'Avril 2022. Chaque journée contient des états différents. Dans la figure 3.3 suivante une présentation des allures de cette vitesse du moteur et celle du VE durant une journée, celle du 22 Avril.

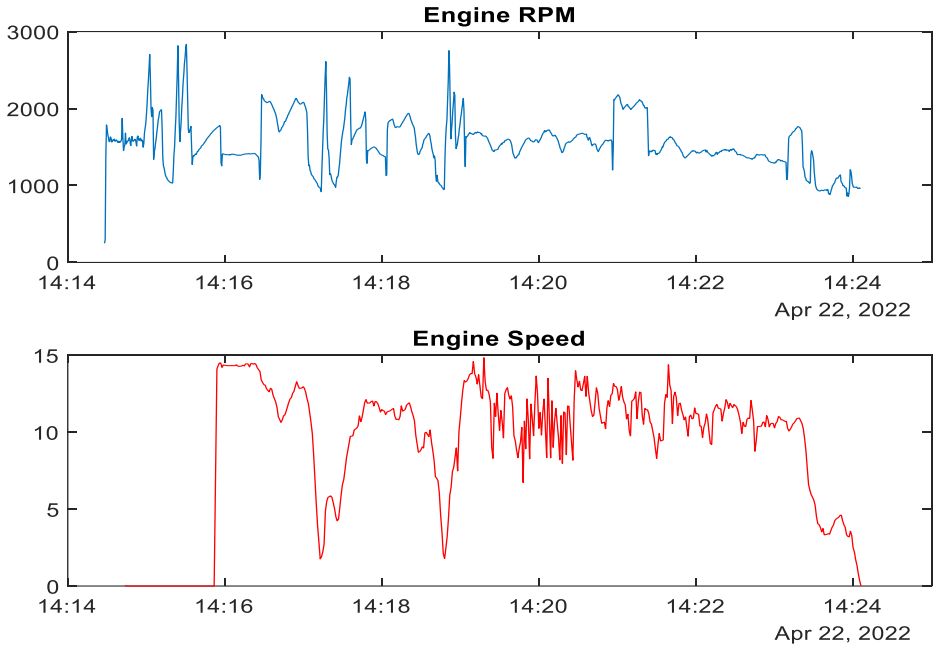


Figure 3.3 : Allures du premier état enregistré dans la journée du 22 Avril.

3.2.2 Choix et développement d'une méthode intelligente pour le Diagnostic :

Pour le traitement et diagnostic de cette BDD (OBDII-CAN), des méthodes ont été développés tel que celle basés sur des statistiques et celle basées sur des lois et seuils (Thresholds), présentés dans les sous-sections suivantes.

a) Basée sur des statistiques :

Des fonctions de la statistique (mean, standard-deviation), ont été impliqués sur les signaux de la vitesse du moteur (RPM), la BDD de l'OBD-CAN. Une comparaison de chaque point de ces données relativement aux fonctions calcules. Les résultats de cette méthode sont montrés dans la Figure 3.3.

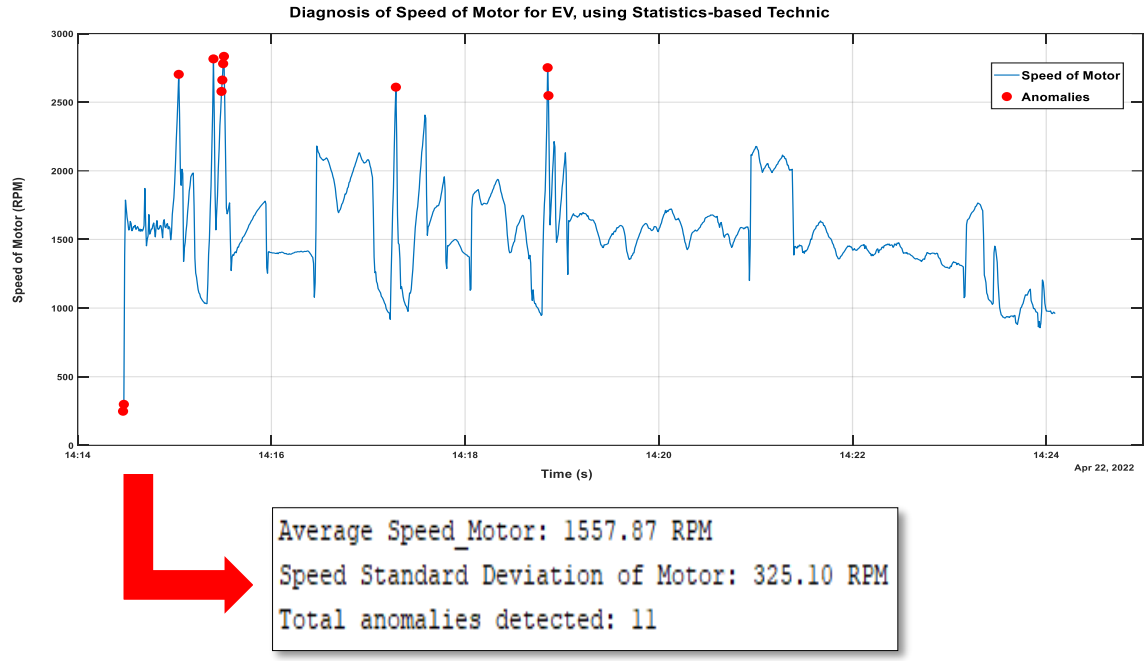


Figure 3.4 : Résultat obtenu du diagnostic basé sur des statistiques.

b) Basée sur des seuils (thresholds & rules-based) :

Pour le moteur des intervalles existent pour ses fonctionnements relatifs au nombre de son rotation par minutes (RPM), schématisés dans la Figure 3.4. Le résultat de l'application de cette méthode est ainsi, montré dans la Figure 3.4.

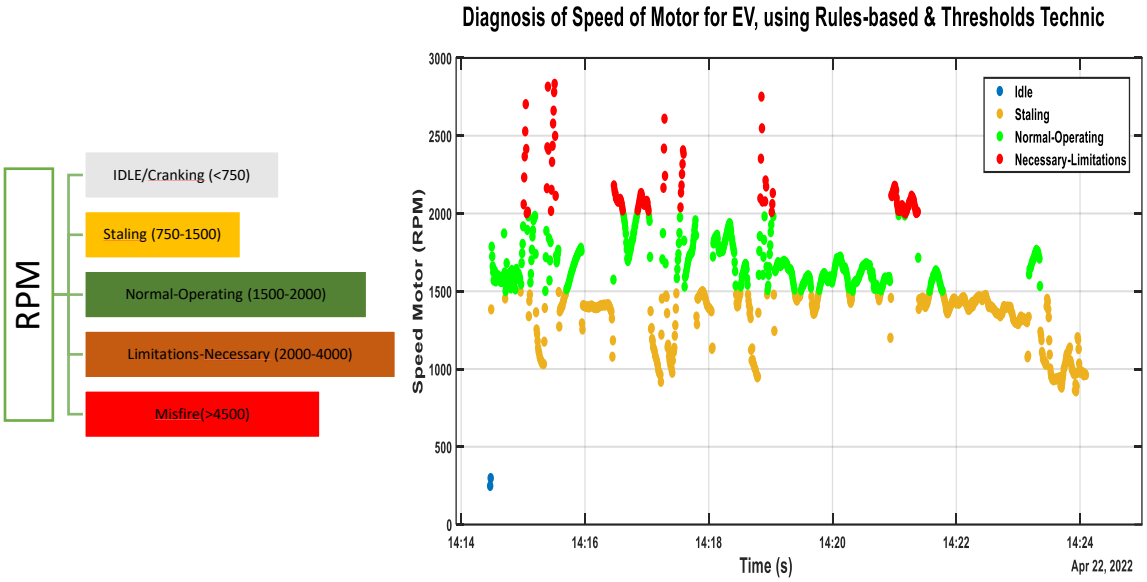


Figure 3.5 : Résultat obtenus pour le diagnostic du moteur, basé sur des seuils.

Pour la vitesse du VE, des intervalles existent pour son fonctionnement qui sont relatifs à l'accélération (dérivée de la vitesse), schématisés dans le schéma de la Figure 3.5. Le résultat de l'application de cette méthode est montré dans la Figure 3.5.

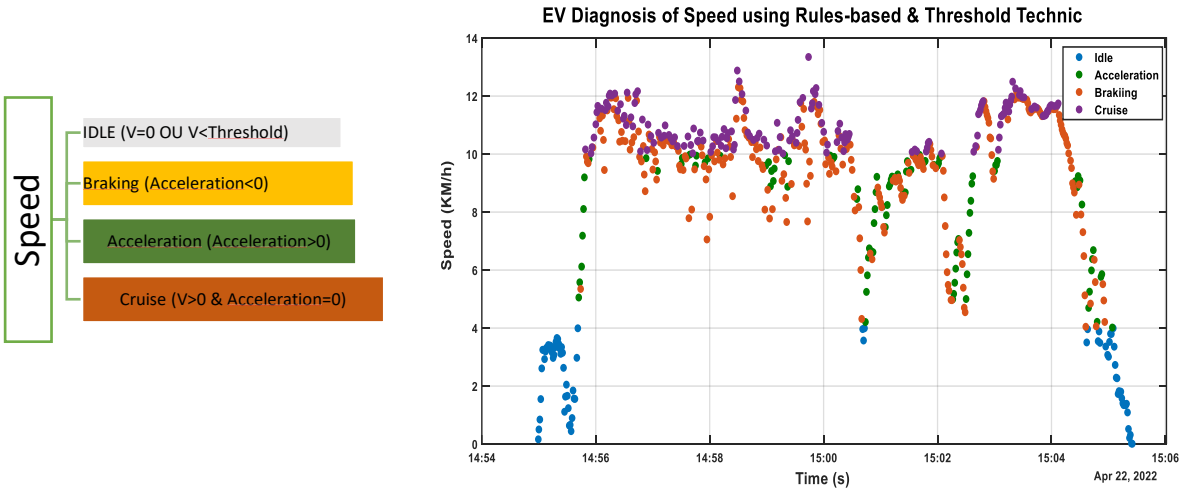


Figure 3.6 : Résultat obtenus pour le diagnostic du moteur, basé sur des seuils.

3.2 Conclusion

Ce chapitre a présenté le traitement et le diagnostic par des méthodes intelligente d'une BDD récolté d'un vrai VE, basé sur un enregistrement CAN-OBDII. Le développement de ces méthodes intelligente (basée sur des statistiques et sur des seuils) a permis le diagnostic de l'élément basique permettant le fonctionnement du VE qui est le moteur, à travers sa vitesse de rotation du moteur RPM. Les résultats obtenus ont montré un bon diagnostic

Chapitre 4 : Conception et Réalisation du Prototype de Diagnostic

4.1 Introduction

Dans les chapitres précédents, une présentation des méthodes de diagnostic automobile a été présenté, particulièrement le diagnostic-embarqué, suivi de l'étude du protocole CAN permettant la communication de SEs dans le VE. Dans ce chapitre, une réalisation d'un prototype de diagnostic-embarqué basé sur le protocole CAN, a été développée. Le schéma synoptique de ce prototype, représentant les différentes parties de ce diagnostic est présentée dans la Figure 4.1.

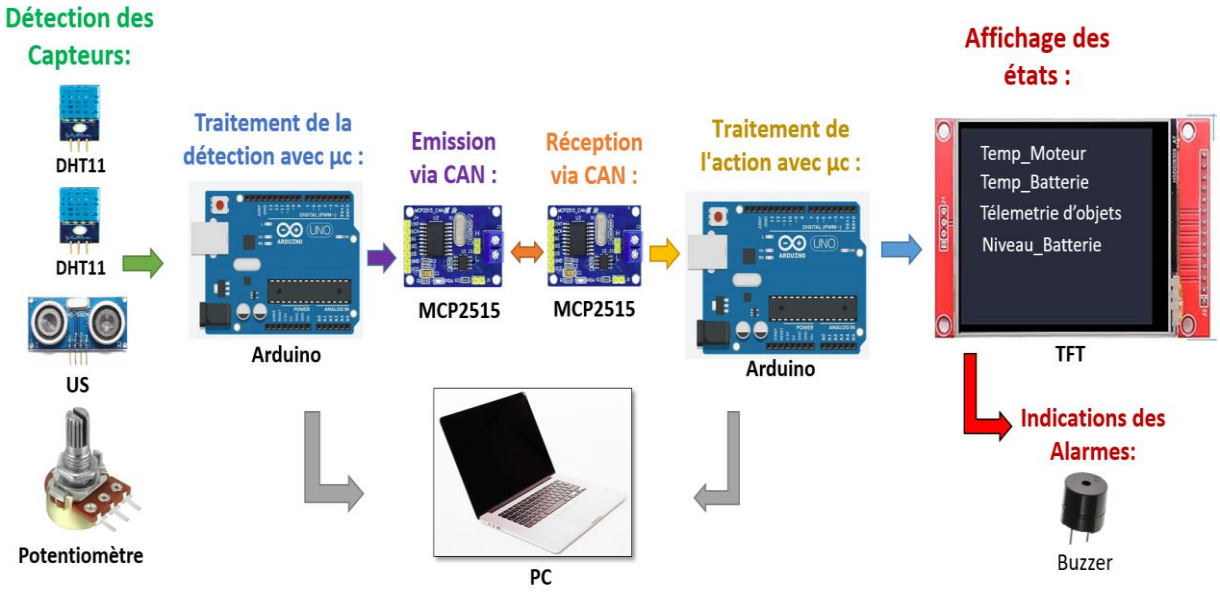


Figure 4.1 : Schéma représentatif du prototype de diagnostic, basé sur CAN.

L'architecture globale du prototype de la Figure 4.1, est constituée de trois parties essentielles :

a) **Détection des capteurs :** celui de température, de distance et de la tension.
b) **Traitement et transmission CAN :** traitement de la détection via microcontrôleur, émission des données détectés via CAN, réception des données transmis via un autre CAN, afin d'être traités dans un autre microcontrôleur.
c) **Affichages et indications des alarmes :** le diagnostic des données selon l'étape précédente est affiché à travers un écran.

Une description de l'environnement logiciel employé est également présenté, les différentes étapes de développement, les schémas techniques ainsi que les montages

Électroniques mis en œuvre pour assurer la communication entre les deux microcontrôleurs, permettant l'acquisition, la transmission et l'affichage des données critiques du véhicule.

4.2 Fonctionnement du prototype

L'architecture du prototype développé se base sur le raccordement de deux microcontrôleurs (Arduino-Uno) via un protocole de communication CAN (2 MCP2515). Ce montage permet la collecte, le traitement et la transmission des données de diagnostic du VE, à partir de plusieurs capteurs embarqués.

Le premier microcontrôleur sert comme émetteur via CAN, chargé de la lecture et diagnostic des données issus des capteurs, tandis que le second joue le rôle d'un récepteur via CAN, chargé de l'affichage des données sur écran.

Ci-suivant une présentation des principaux éléments/composants utilisés et leurs fonctions :

a) Nœud Émetteur (Arduino-Uno) :

- Permet de récolter les données de deux capteurs **DHT11** pour surveiller la température de la batterie et celle du moteur.
- D'un capteur à ultrasons **(HC-SR04)** pour détecter la présence d'obstacles.
- D'un **potentiomètre**, qui permet de simuler la tension de la batterie.
- Un diagnostic des différents grandeurs récoltés des capteurs est effectué à ce nœud.
- Transmet ainsi, les données diagnostiquées de différents capteurs, au récepteur via un module CAN, le **MCP2515**.

b) Communication CAN (via deux modules MCP2515) :

- Une communication fiable est assurée entre les deux Arduino (émetteur et récepteur), en utilisant le protocole CAN.

c) Nœud Récepteur (Arduino-Uno) :

- Reçoit les données du protocole CAN.
- Traite les données reçues.

- Affiche les informations sur un écran TFT 1.77" **(ST7735)**, pour un suivi en temps réel des paramètres du système**.**
- Indications des alarmes sonores via des buzzers.

4.3 Description du matériel utilisé

4.3.1 Arduino-UNO :

L'Arduino-Uno (Figure 4.2) est une carte de développement électronique basée sur le microcontrôleur ATmega328P. Elle est largement utilisée dans les projets d'électronique embarquée grâce à sa simplicité d'utilisation, sa programmation facile via l'IDE Arduino, et sa compatibilité avec de nombreux capteurs et modules.

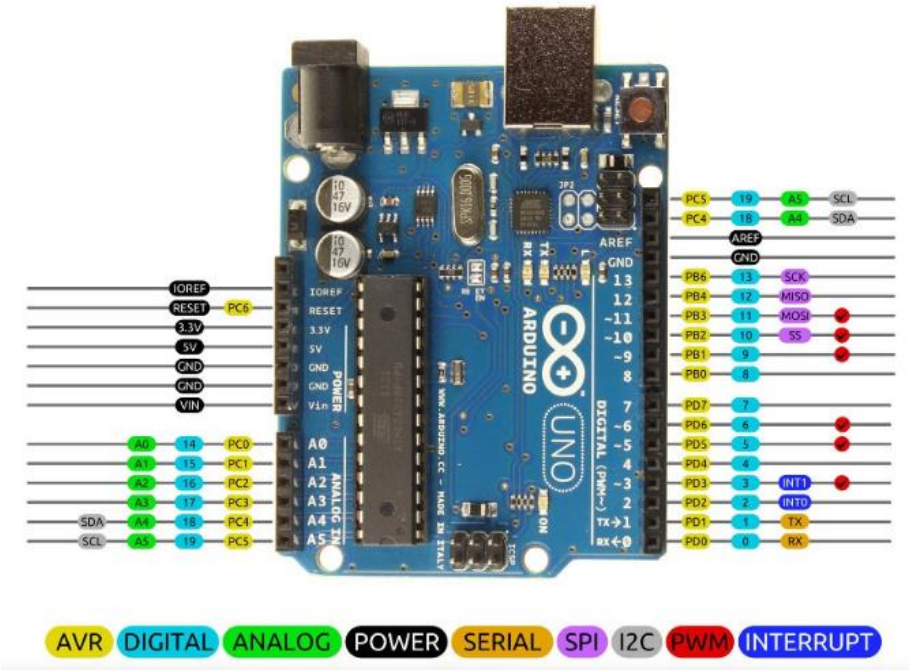


Figure 4.2 : Photo d'une carte Arduino-UNO.

a) Caractéristiques :

- Microcontrôleur : ATmega328P
- Fréquence d'horloge : 16 MHz
- Mémoire flash : 32 Ko (dont 0,5 Ko utilisés par le bootloader)
- SRAM : 2 Ko
- EEPROM : 1 Ko

- Entrées/Sorties numériques : 14 (dont 6 PWM)
- Entrées analogiques : 6
- Tension de fonctionnement : 5V
- Interface USB pour la programmation

b) Rôle

Dans ce prototype développé permettant le diagnostic du VE, l'Arduino-Uno joue un rôle central en tant qu'unité de traitement, selon que :

- **Nœud-Émetteur** : collecte les données de capteurs (2 températures, distance, tension), établir les états (normal, alarmants) et les envoie via le bus CAN.
- **Nœud-Récepteur** : récupère les données via CAN, les traite et les affiche sur un écran TFT. Il assure donc la liaison entre les capteurs et l'affichage, tout en permettant une communication robuste et fiable grâce au protocole **CAN**.

4.3.2 MCP2515 :

Le module de communication **CAN** illustré dans la Figure 4.3, est basé sur le contrôleur MCP2515, qui est utilisé pour permettre la communication entre les microcontrôleurs (Nœuds du CAN), via son CAN-Tranciever (TJA1050).

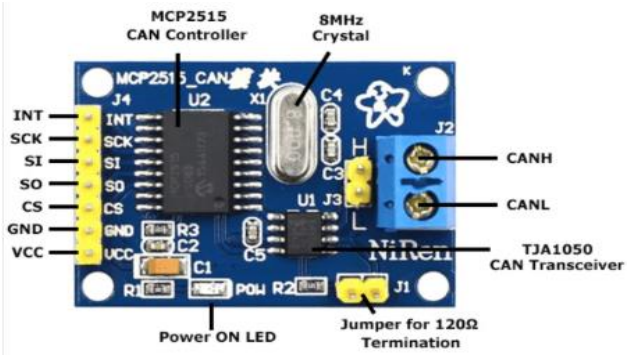


Figure 4.3 : Photo d'un module CAN (MCP2515).

a) Caractéristiques :

- Protocole **CAN 2.0**
- Fonctionne via **SPI**

- Vitesse jusqu'à **1 Mb/s**
- Nécessite un **transceiver** (ex. TJA1050)

b) Rôle :

Assure l'envoi et la réception de messages **CAN** entre les microcontrôleurs pour échanger des données provenant de capteurs ou vers des afficheurs et actionneurs.

4.3.3 DHT11

Le DHT11 (Figure 4.4) est un capteur numérique qui permet de mesurer la température et l'humidité de l'air.

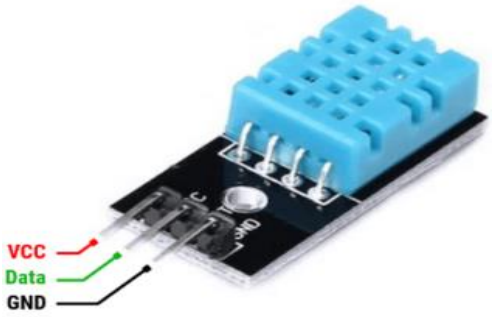


Figure 4.4 : Photo d'un capteur DHT11.

a) Caractéristiques :

- Plage de température : **0 à 50 °C**
- Plage d'humidité : **20 à 90 %**
- Sortie numérique (1 fil)
- Temps de réponse : ~1 s

b) Rôle :

Fournit les valeurs de température et d'humidité pour le diagnostic thermique d'un système.

4.3.4 ULTRASON (HC-SR04)

Le capteur à ultrasons (HC-SR04) présenté en Figure 4.5, permet de mesurer la distance entre un obstacle et ce capteur.

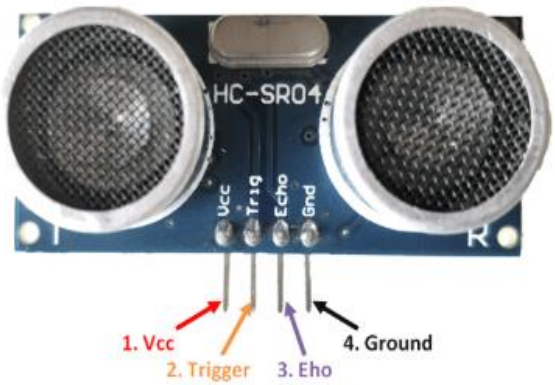


Figure 4.5 : Photo d'un capteur à ultrasons (HC-SR04).

a) Caractéristiques :

- Portée : **2 cm à 400 cm**
- Précision : ± **3 mm**
- Tension : **5 V**
- Interface : **Trig** et **Echo** (2 broches)

b) Rôle :

Détecte la présence d'obstacles ou mesure la distance dans l'environnement du véhicule.

4.3.5 Potentiomètre

Le potentiomètre (Figure 4.6) est une résistance variable, utilisée pour fournir un signal **analogique** en sortie.

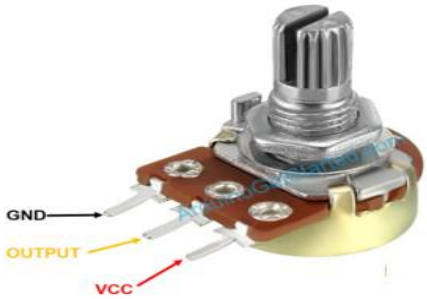


Figure 4.6 : Photo d'un Potentiomètre.

a) Caractéristiques :

- Résistance typique : **10 kΩ**
- Sortie : **0–5 V** proportionnelle à la rotation.

b) Type :

Rotatif ou **linéaire.**

c) Rôle :

Simule un **paramètre variable** (par exemple, tension, courant ou position) dans le système pour tester ou ajuster des seuils.

4.3.6 Plaque d'essai (Hitchman) :

Une plaque d'essai illustrée en Figure 4.7, souvent appelée **breadboard** en anglais, est un outil essentiel en électronique utilisé pour créer des prototypes de circuits sans avoir besoin de souder les composants. Cet outil permettant de tester et valider rapidement des conceptions de circuits électroniques avant de passer à la phase de fabrication sur une carte de circuit imprimé (Printed-Circuit-Board (PCB)).

Figure 4.7 : Photo d'une plaque à essai.

a) Caractéristiques :

- Base en plastique
- Rangées de trous interconnectés : Elles sont généralement organisées en trois zones :
  - Rails d'alimentation verticaux
  - Zone centrale
  - Rails de connexion horizontaux

b) Rôle :

- Prototypage rapide.
- Flexibilité.
- Économie de temps.

4.3.7 Outils logiciels :

La Figure 4.8, présente L'IDE d'Arduino, qui est l'environnement de programmation d'Arduino-UNO.

Figure 4.8 : Fenêtre du logiciel IDE d'Arduino.

4.4 Partie software

L'organigramme de la Figure 4.9 présente les étapes principales du fonctionnement général du prototype :

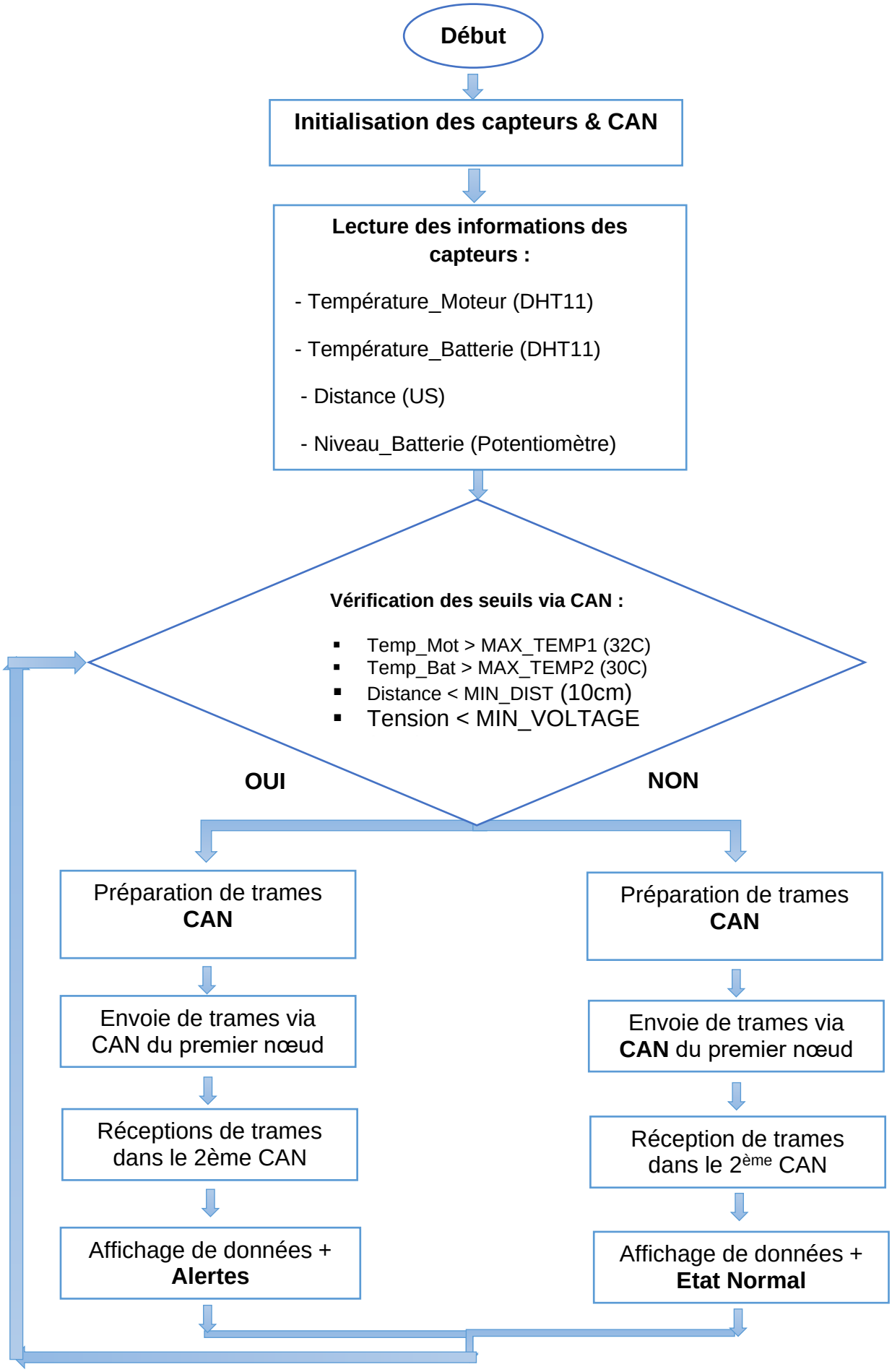

Début
Initialisation des capteurs & CAN
Lecture des informations des capteurs :
- Température_Moteur (DHT11)
- Température_Batterie (DHT11)
- Distance (US)
- Niveau_Batterie (Potentiomètre)
Vérification des seuils via CAN :
Temp_Mot > MAX_TEMP1 (32C)
Temp_Bat > MAX_TEMP2 (30C)
Distance < MIN_DIST (10cm)
Tension < MIN_VOLTAGE
OUI
NON
Préparation de trames CAN
Envoie de trames via CAN du premier nœud
Réceptions de trames dans le 2ème CAN
Affichage de données + Alertes
Préparation de trames CAN
Envoie de trames via CAN du premier nœud
Réception de trames dans le 2ème CAN
Affichage de données + Etat Normal

Figure 4.9 : Organigramme de fonctionnement du prototype.

Les seuils de températures ont été fixés selon des condition environnemental réelles (de 30 à 32). Par contre, en réalité la température du moteur est dans l'intervalle (de 90 à 105), et la température de batterie est dons l'intervalle (de 20 à 40).

4.5 Schémas de Montages

4.5.1 Schémas de raccordement électronique

Le schéma de raccordement électronique comprend les connexions suivantes :

c) Raccordement de 2 MCP2515 avec les 2 Arduino (Figure 4.10).

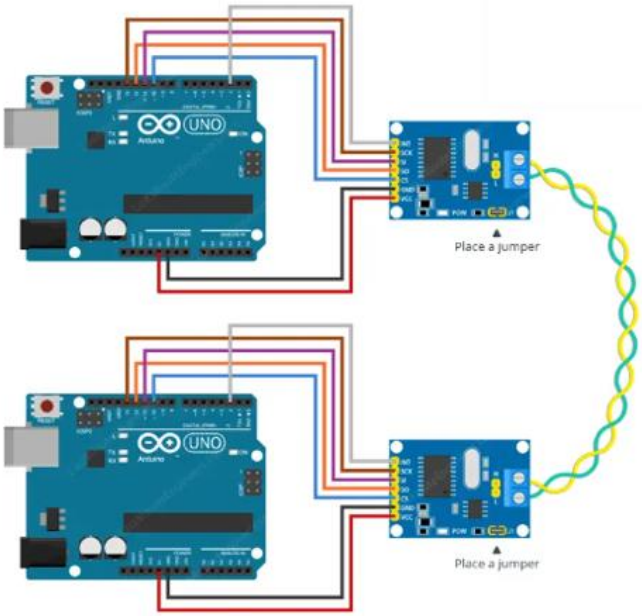


Figure 4.10 : Raccordement de MCP2515+Arduino-UNO.

Ci-dessous sont les PINs de raccordement :

MCP2515 (CAN) vers Arduino-Uno

- VCC → 5V
- GND → GND
- CS → D10
- SCK → D13
- SO (MISO) → D12

- SI (MOSI) → D11
- INT → D2

Il faut relier les deux MCP2515 selon que le CAN_H avec CAN_H et le CAN-L avec le CAN-L.

d) Raccordement du DHT11 avec Arduino (Figure 4.11):

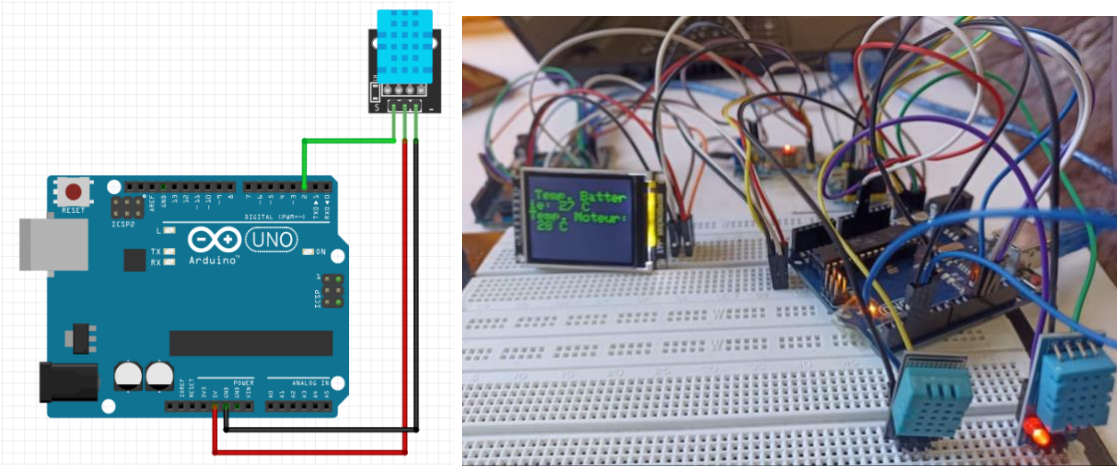


Figure 4.11 : Raccordement de DHT11+Arduino UNO.

Ci-dessous sont les PINs de raccordement :

DHT11 n°1 (Température_Moteur)

- VCC → 5V
- GND → GND
- DATA → D3

DHT11 n°2 (Température_Batterie)

- VCC → 5V
- GND → GND
- DATA → D4

e) Raccordement de l'ultrason (HC-SR04) avec Arduino (Figure 4.12) :

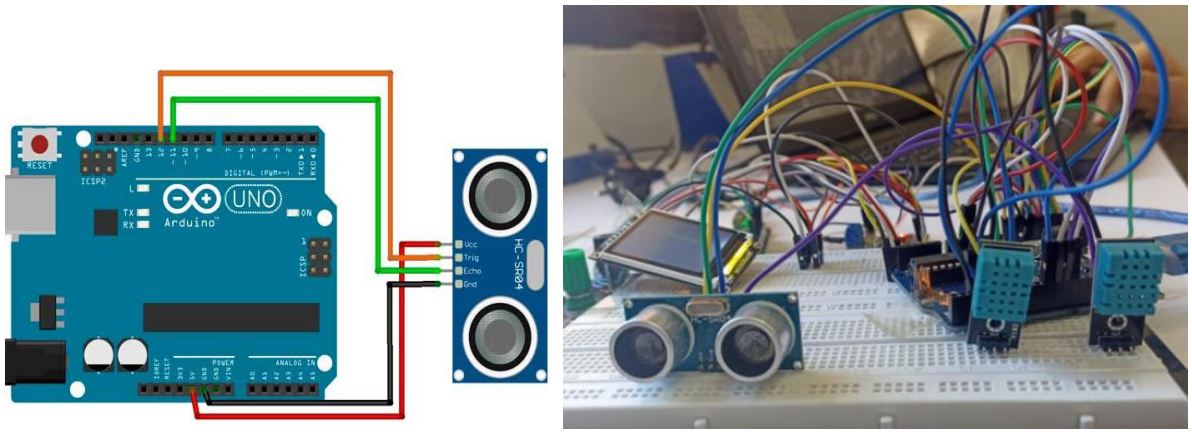


Figure 4.12 : Raccordement de l'ultrason (HC-SR04) + Arduino-UNO

Ci-dessous sont les PINs de raccordement :

Capteur Ultrason (HC-SR04)

- VCC → 5V
- GND → GND
- TRIG → D5
- ECHO → D6

f) **Raccordement du Potentiomètre** avec Arduino (Figure 4.13) :

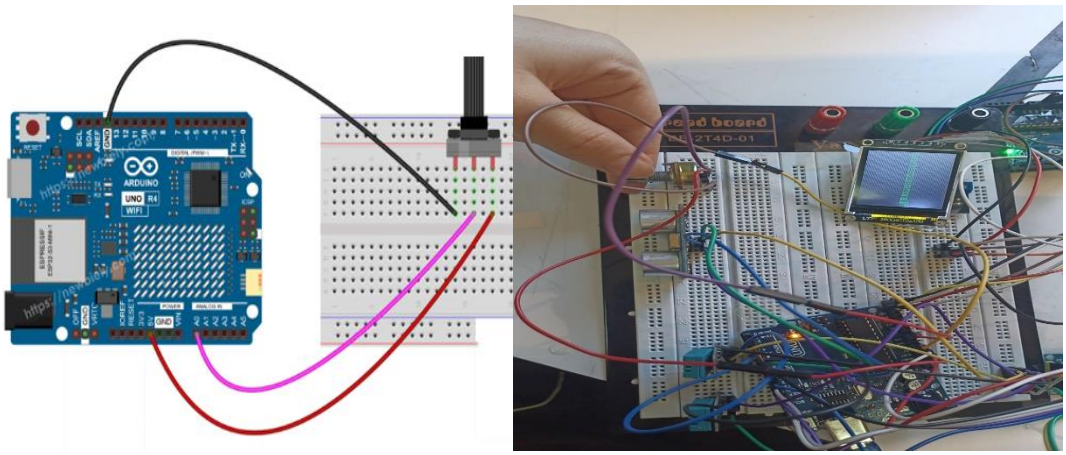


Figure 4.13 : Raccordement du Potentiomètre +Arduino-UNO.

Ci-dessous sont les PINs de raccordement :

Potentiomètre

- VCC → 5V
- GND → GND
- Sortie → A0

g) Raccordement de l'Écran TFT ST7735 avec Arduino :

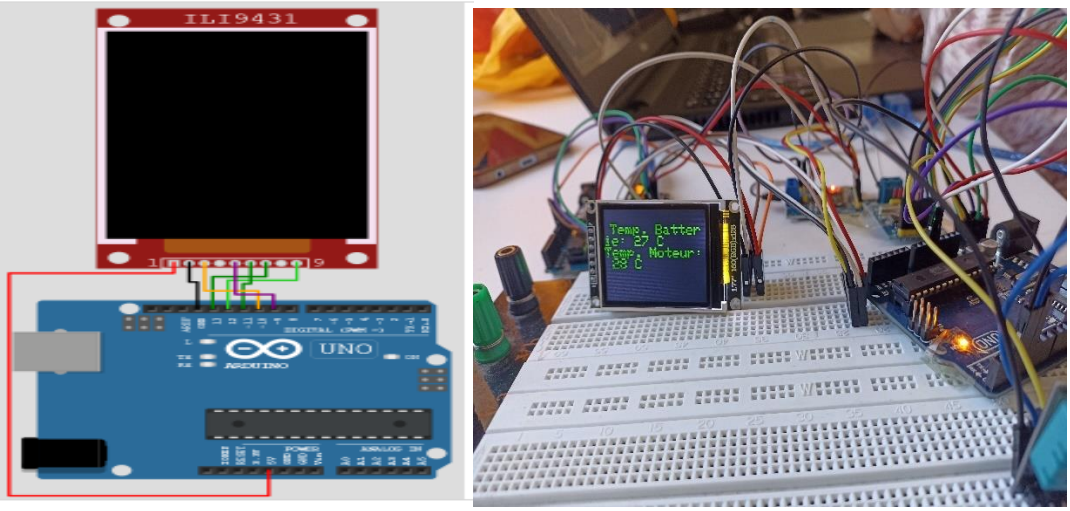


Figure 4.14 : Raccordement de l'écran TFT **(**ST7735) +Arduino-UNO.

Ci-dessous sont les PINs de raccordement :

- VCC → 5V
- GND → GND
- CS → D9
- DC → D8
- RESET → D7
- SCK → D13
- SDA/MOSI → D11

4.6 Schéma Synoptique du prototype

La Figure 4.15 présente le schéma synoptique du prototype de diagnostic développé.

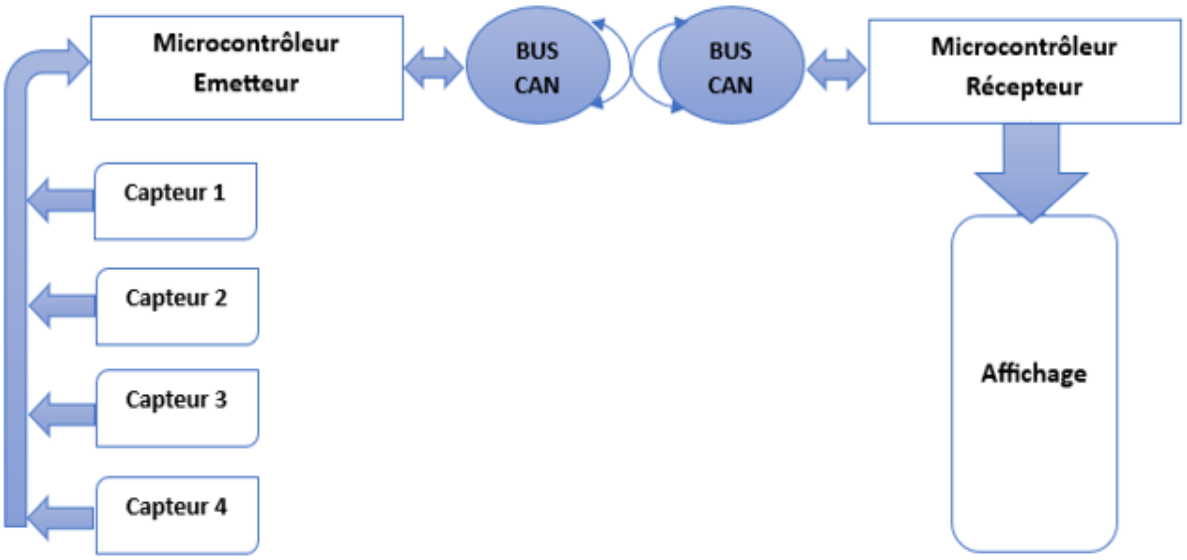


Figure 4.15 : Schéma synoptique du prototype de diagnostic développé

4.7 Montages réalisés

Le montage du prototype (Figure 4.16) a été réalisé sur une plaque d'essai (breadboard) afin de tester le bon fonctionnement de chaque composant et valider l'échange de données via le protocole CAN. Cette étape permet d'expérimenter et d'ajuster les connexions entre les deux microcontrôleurs (Arduino), les capteurs (2 DHT11, US, potentiomètre) et l'affichage TFT. Une fois les tests réussis, le système pourra être réalisé sur un circuit imprimé (PCB) pour une installation plus fiable, robuste et adaptée à une application embarquée dans un véhicule électrique.

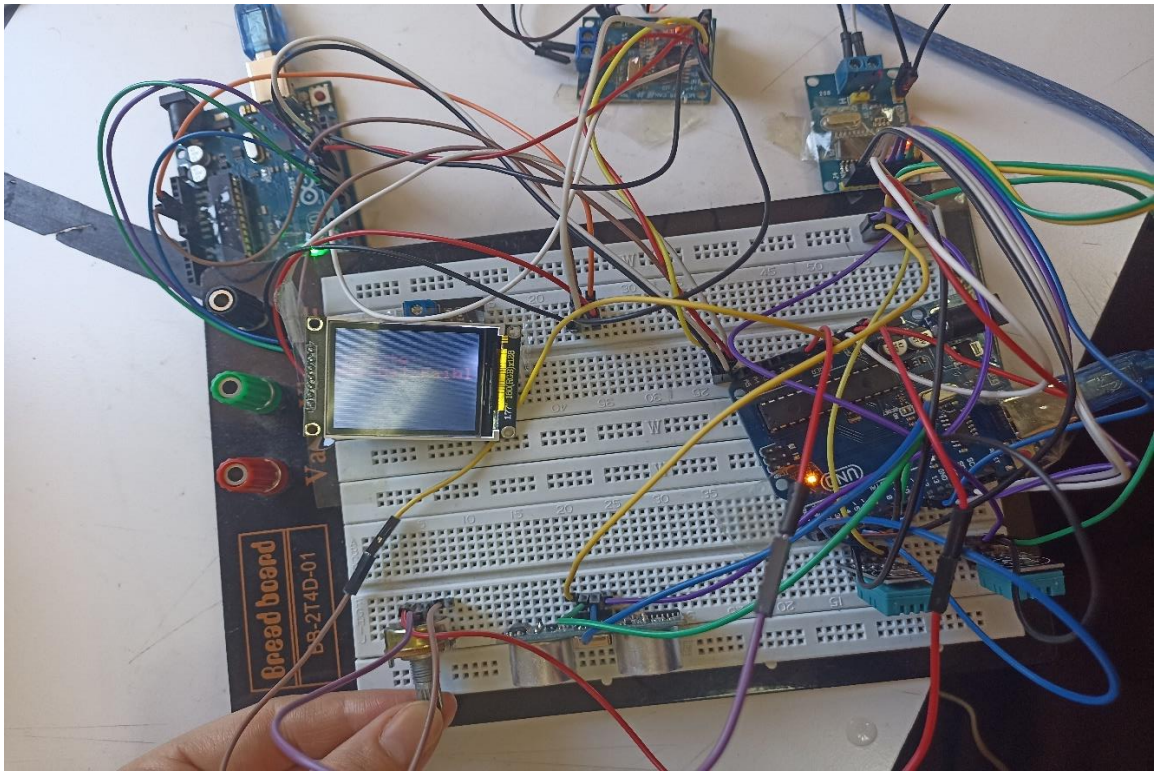


Figure 4.16 : Montage globale du système réalisé.

4.7.1 Résultats obtenus

Le programme du prototype est ci-suivant :

- **Code Emetteur**

```
#include <SPI.h>              // Bibliothèque pour la communication SPI (utilisée par le MCP2515)
#include <mcp2515.h>          // Bibliothèque pour le module CAN MCP2515
#include <DHT.h>              // Bibliothèque pour le capteur de température/humidité DHT11
#define DHTPIN_BATTERY 3      // Pin de connexion du capteur DHT11 pour la batterie
#define DHTPIN_MOTOR 4        // Pin de connexion du capteur DHT11 pour le moteur
#define DHTTYPE DHT11         // Type de capteur DHT utilisé
#define TRIG_PIN 5            // Pin TRIG du capteur à ultrasons HC-SR04
#define ECHO_PIN 6            // Pin ECHO du capteur HC-SR04
#define POTENTIOMETER_PIN A0  // Pin analogique pour le potentiomètre simulant la tension batterie
// Création des objets DHT pour batterie et moteur
DHT dhtBattery(DHTPIN_BATTERY, DHTTYPE);
DHT dhtMotor(DHTPIN_MOTOR, DHTTYPE);
// Création de l'objet MCP2515 en utilisant le pin CS = 10
MCP2515 mcp2515(10);
struct can_frame canMsg;      // Structure pour contenir les données CAN
void setup() {
  Serial.begin(9600);         // Initialisation de la communication série
  SPI.begin();                // Initialisation de la communication SPI
  dhtBattery.begin();         // Démarrage du capteur DHT pour batterie
```

```
  dhtMotor.begin();           // Démarrage du capteur DHT pour moteur
  pinMode(TRIG_PIN, OUTPUT);  // Définir TRIG comme sortie
  pinMode(ECHO_PIN, INPUT);   // Définir ECHO comme entrée
  mcp2515.reset();            // Réinitialisation du module CAN
  mcp2515.setBitrate(CAN_500KBPS, MCP_8MHZ);  // Configuration du débit CAN
  mcp2515.setNormalMode();    // Passage en mode normal (reçoit et envoie des messages)
}
void loop() {
  // Lecture des températures depuis les capteurs DHT11
  float tempBattery = dhtBattery.readTemperature();
  float tempMotor = dhtMotor.readTemperature();
  // Vérifie si les lectures sont valides
  if (isnan(tempBattery) || isnan(tempMotor)) return;
  // Conversion des températures en entiers (1 octet)
  byte temperatureBattery = (byte)tempBattery;
  byte temperatureMotor = (byte)tempMotor;
  // Déclenchement du capteur à ultrasons
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);       // Envoi d'une impulsion
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  long duration = pulseIn(ECHO_PIN, HIGH);     // Lecture de la durée de l'écho
  byte distance = duration * 0.034 / 2;        // Conversion en centimètres
  // Lecture du potentiomètre (simule le niveau de batterie)
  int potValue = analogRead(POTENTIOMETER_PIN);
  byte batteryLevel = map(potValue, 0, 1023, 0, 100);  // Conversion en pourcentage (0-100)
  // Préparation de la trame CAN avec 8 octets
  canMsg.can_id  = 0x200;                // ID du message CAN
  canMsg.can_dlc = 8;                    // Longueur des données
  canMsg.data[0] = temperatureBattery;   // Température batterie
  canMsg.data[1] = temperatureMotor;     // Température moteur
  canMsg.data[2] = (temperatureBattery > 30) ? 1 : 0;  // Alerte température batterie
  canMsg.data[3] = (temperatureMotor > 32) ? 1 : 0;    // Alerte température moteur
  canMsg.data[4] = distance;             // Distance détectée
  canMsg.data[5] = (distance < 10) ? 1 : 0; // Alerte obstacle
  canMsg.data[6] = batteryLevel;         // Niveau batterie
  canMsg.data[7] = (batteryLevel < 20) ? 1 : 0;  // Alerte batterie faible
  mcp2515.sendMessage(&canMsg);          // Envoi du message CAN

  // Affichage des valeurs dans le moniteur série (debug)
  Serial.println("---- Données envoyées ----");
  Serial.print("Temp Batt: "); Serial.println(temperatureBattery);
  if (temperatureBattery < 30) {
    Serial.println("Alerte: Batterie faible !");
  }
  Serial.print("Temp Moteur: "); Serial.println(temperatureMotor);
  if (temperatureMotor < 32) {
    Serial.println(" Alerte: Batterie faible !");
```

```
  }
  Serial.print("Distance: "); Serial.println(distance);
  if (distance < 10) {
    Serial.println(" Alerte: Obstacle!");
  }
  Serial.print("Niveau Batterie: "); Serial.print(batteryLevel); Serial.println(" %");
  if (batteryLevel < 20) {
    Serial.println(" Alerte: Batterie faible !");
  }
  Serial.println("-------------------------");
  delay(2000);  // Pause de 2 secondes avant la prochaine boucle
}
```

- **Code Récepteur**

```
#include <SPI.h>                    // Bibliothèque pour la communication SPI (utilisée par le MCP2515 et l'écran TFT)
#include <mcp2515.h>                // Bibliothèque pour le module CAN MCP2515
#include <Adafruit_GFX.h>           // Bibliothèque graphique de base pour les écrans Adafruit
#include <Adafruit_ST7735.h>        // Bibliothèque pour écran TFT ST7735
#define TFT_CS   9                  // Broche CS (Chip Select) de l'écran TFT
#define TFT_DC   8                  // Broche DC (Data/Command) de l'écran TFT
#define TFT_RST  7                  // Broche RST (Reset) de l'écran TFT
MCP2515 mcp2515(10);                // Création de l'objet MCP2515 avec la broche CS connectée à D10
Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS, TFT_DC, TFT_RST); // Initialisation de l'écran TFT

struct can_frame canMsg;            // Déclaration d'une structure pour recevoir un message CAN
void setup() {
  Serial.begin(9600);               // Initialisation du port série à 9600 bauds
  tft.initR(INITR_BLACKTAB);        // Initialisation de l'écran TFT avec l'option "BLACKTAB"
  tft.setRotation(1);               // Orientation de l'écran (paysage)
  tft.fillScreen(ST77XX_BLACK);     // Remplit l'écran en noir
  tft.setTextSize(2);               // Taille du texte à 2
  tft.setTextColor(ST77XX_WHITE);   // Couleur du texte : blanc
  tft.setCursor(10, 30);            // Position du curseur
  tft.println("En attente CAN...");  // Message d'attente initial sur l'écran

  mcp2515.reset();                  // Réinitialisation du MCP2515
  mcp2515.setBitrate(CAN_500KBPS, MCP_8MHZ); // Configuration du débit CAN à 500 kbps pour un quartz de 8 MHz
  mcp2515.setNormalMode();          // Passage du MCP2515 en mode normal (réception active)
}
void loop() {
  // Vérifie si un message CAN a été reçu
  if (mcp2515.readMessage(&canMsg) == MCP2515::ERROR_OK) {
    // Extraction des données du message CAN
    byte temperatureBattery = canMsg.data[0];     // Température de la batterie
```

```
  byte temperatureMotor   = canMsg.data[1];    // Température du moteur
  byte alertBatteryTemp   = canMsg.data[2];    // Alerte sur température batterie
  byte alertMotorTemp     = canMsg.data[3];    // Alerte sur température moteur
  byte distance           = canMsg.data[4];    // Distance mesurée (ultrasons)
  byte alertObstacle      = canMsg.data[5];    // Alerte obstacle proche
  byte batteryLevel       = canMsg.data[6];    // Niveau de la batterie (en %)
  byte alertLowBattery    = canMsg.data[7];    // Alerte batterie faible
  // Réinitialise l'écran (efface tout)
  tft.fillScreen(ST77XX_BLACK);
  tft.setTextSize(1);                          // Texte plus petit pour les infos
  tft.setTextColor(ST77XX_WHITE);              // Couleur blanche
  // Affiche la température de la batterie
  tft.setCursor(5, 5);
  tft.print("Temp Batt: ");
  tft.print(temperatureBattery);
  tft.println(" C");
  // Affiche la température du moteur
  tft.setCursor(5, 20);
  tft.print("Temp Moteur: ");
  tft.print(temperatureMotor);
  tft.println(" C");
  // Affiche la distance
  tft.setCursor(5, 35);
  tft.print("Distance: ");
  tft.print(distance);
  tft.println(" cm");
  // Affiche le niveau de batterie
  tft.setCursor(5, 50);
  tft.print("Niv Batterie: ");
  tft.print(batteryLevel);
  tft.println(" %");
  // Affichage des alertes si nécessaire
  tft.setTextSize(1);
  if (alertBatteryTemp || alertMotorTemp || alertObstacle || alertLowBattery) {
    tft.setTextColor(ST77XX_RED);              // Couleur rouge pour les alertes
    int y = 70;
    tft.setCursor(5, y);
    tft.println("Alertes:");
    if (alertBatteryTemp) {
      tft.setCursor(10, y + 12); tft.println("Temp Batterie !");
    }
    if (alertMotorTemp) {
      tft.setCursor(10, y + 24); tft.println("Temp Moteur !");
    }
    if (alertObstacle) {
      tft.setCursor(10, y + 36); tft.println("Obstacle proche !");
    }
    if (alertLowBattery) {
      tft.setCursor(10, y + 48); tft.println("Batterie faible !");
```

```
    }
   } else {
    // Aucune alerte → affichage état normal
    tft.setTextColor(ST77XX_GREEN);
    tft.setCursor(5, 70);
    tft.println("Etat: NORMAL");
   }
   // Affichage des données dans le moniteur série
   Serial.println("---- Données reçues ----");
   Serial.print("Temp Batt : "); Serial.println(temperatureBattery);
   Serial.print("Temp Moteur : "); Serial.println(temperatureMotor);
   Serial.print("Distance : "); Serial.println(distance);
   Serial.print("Niveau Batterie : "); Serial.print(batteryLevel); Serial.println(" %");
   // Affichage des alertes dans le moniteur série
   if (alertBatteryTemp) Serial.println("Alerte: Temp Batterie élevée !");
   if (alertMotorTemp)   Serial.println("Alerte: Temp Moteur élevée !");
   if (alertObstacle)    Serial.println("Alerte: Obstacle détecté !");
   if (alertLowBattery)  Serial.println("Alerte: Batterie faible !");
   // Si aucune alerte
   if (!(alertBatteryTemp || alertMotorTemp || alertObstacle || alertLowBattery)) {
    Serial.println(" Etat normal.");
   }
   Serial.println("-----------------------");
   delay(2000); // Pause de 2 secondes avant de traiter un nouveau message
  }
}
```

- **Bibliothèques utilisées :**

- #include <SPI.h>
- #include <mcp2515.h>
- #include <DHT.h>
- #include <SPI.h>
- #include <Adafruit_GFX.h>
- #include <Adafruit_ST7735.h>

4.7.2 Discussions des résultats obtenus :

a) Etat normal :

La Figure 4.17, montre l'affichage des résultats obtenus dans l'état normal.

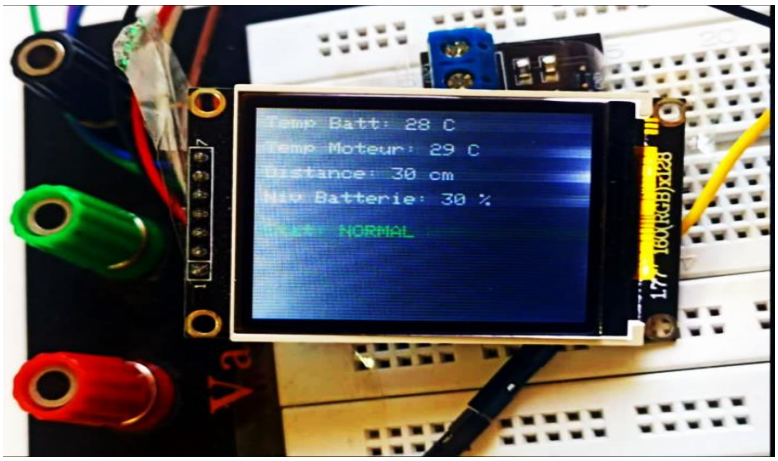
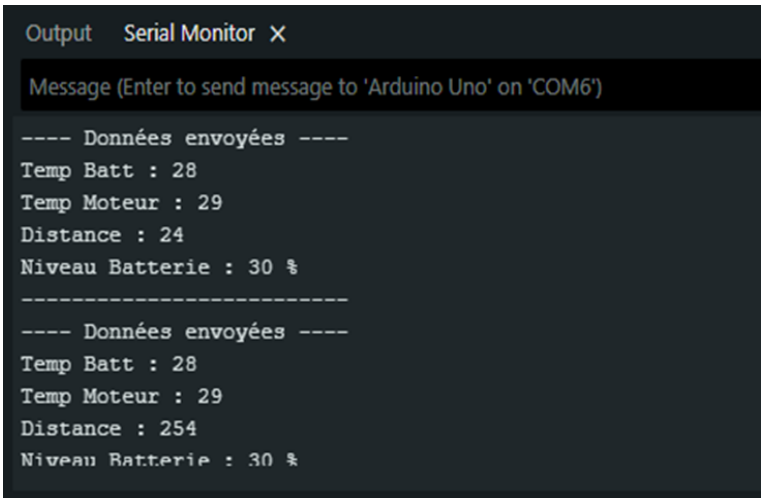
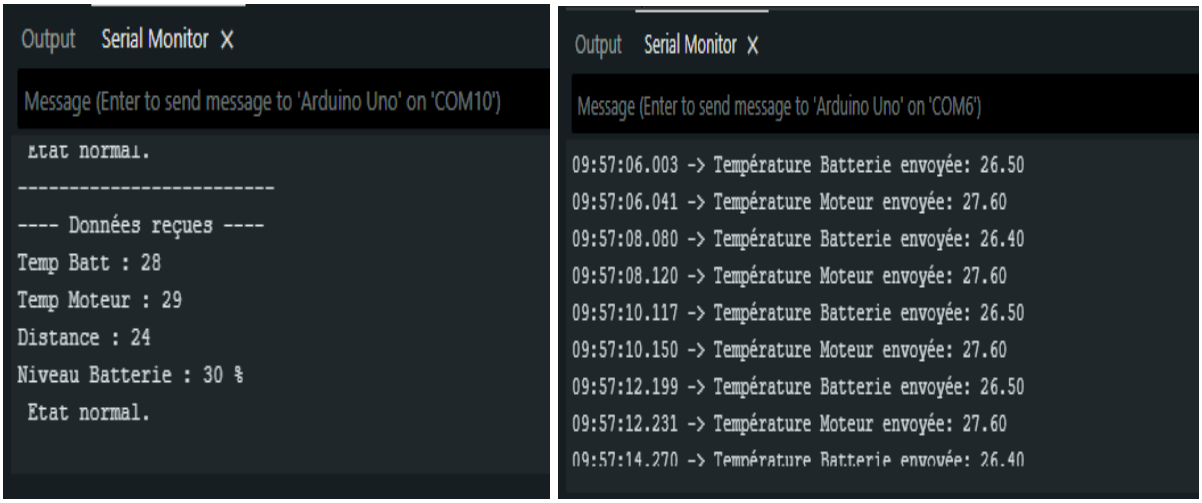


Figure 4.17 : Affichage des résultats obtenus dans l'état normal.

L'affichage de résultats obtenus dans le serial monitor de l'IDE d'Arduino, coté émetteur est ci-suivant :



L'affichage de résultats obtenus dans le serial monitor de l'IDE d'Arduino, coté récepteur est ci-suivant :

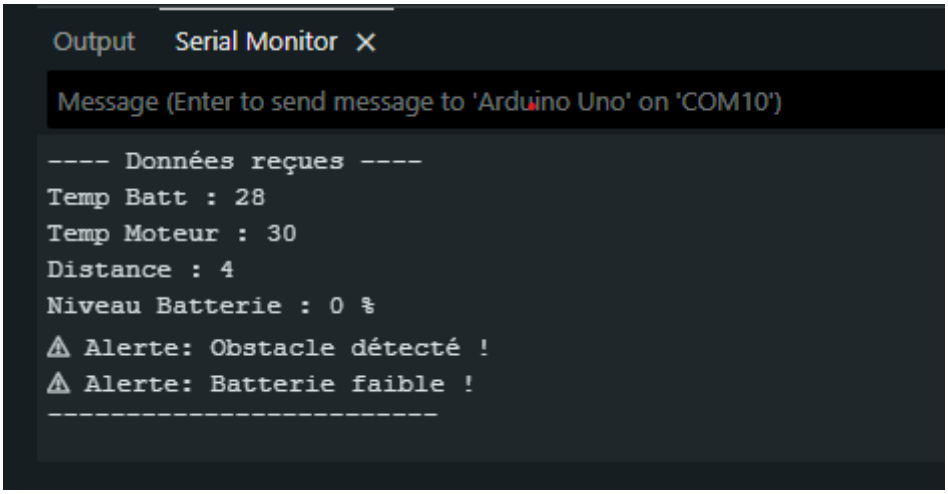


b) Etat d’anomalies :

La Figure 4.18, montre l’affichage des résultats obtenus dans l’état d’anomalies.

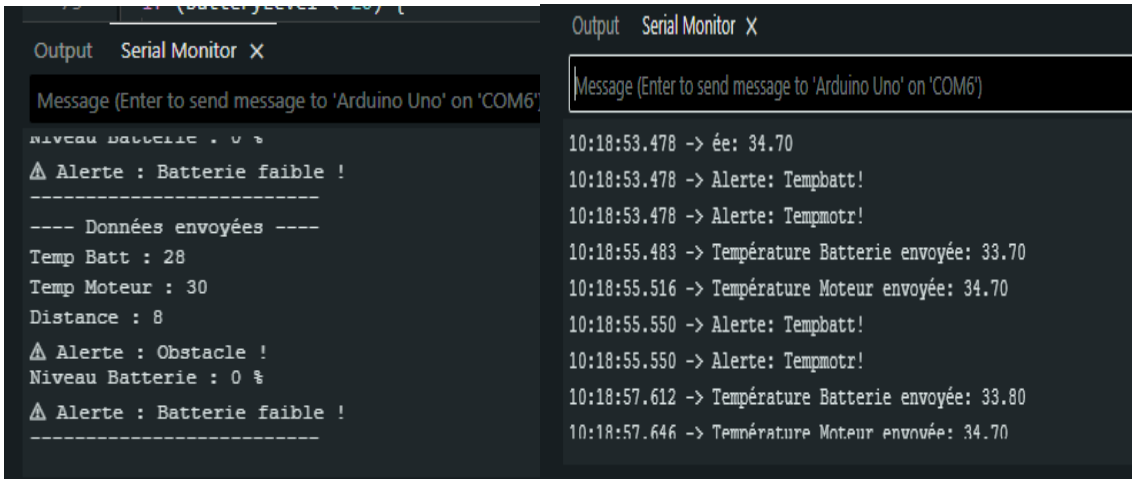
Figure 4.18: Affichage des résulta dans Etat d’anomalies

L’affichage de résultats obtenus dans le serial monitor de l’IDE d’Arduino, coté émetteur est ci-suivant :



L’affichage de résultats obtenus dans le serial monitor de l’IDE d’Arduino, coté récepteur est ci-suivant :

Figure 4.19: Affichage des résulta dans Etat d'anomalies (la température, la distance)



4.8 Conclusion :

Dans ce chapitre, une implémentation pratique a été effectué permettant le développement du prototype de diagnostic dédié au véhicule électrique, via le protocole CAN. Dans ce sens, deux microcontrôleurs (Arduino-Uno) ont été utilisés, et qui ont été interconnectés via CAN (2 modules MCP2515). Nous avons pu assurer la communication entre un module CAN émetteur collectant les données (température Moteur, Température_Batterie, Distance et Niveau_Batterie) et un module CAN récepteur qui se charge de les afficher sur un écran TFT. La collecte des données des différents capteurs utilisés (2DHT11, US et potentiomètre) a rendu le suivi des conditions critiques du système, plus approprié. De ces données une méthode basique de diagnostic a été impliqué, basé sur des seuils (Rules & Thresholds-based)

pour chaque grandeur. Les tests menés sur la plaque d'Hitchman (breadboard) ont validé le bon fonctionnement du prototype de diagnostic développé, ce qui prouve un bon montage du système, qui permet la détection des états alarmants. Cette réalisation pratique pourrait passer vers une phase terminale sur circuits imprimés (Printed Circuit Board (PCB)), tout en ajoutant d'autres capteurs (pour autres anomalies).

Conclusion générale

Dans un contexte général marqué par la tendance vers une mobilité plus durable, les VEs présentent de nouveaux défis en matière de maintenance, principalement en raison de la complexité de leur architecture électronique. La mise en place d'un système de diagnostic efficace, flexible et à moindre coût devient donc une nécessité.

Ce travail a permis de développer et de tester un système de diagnostic spécifique pour les véhicules électriques (VEs), en utilisant une architecture embarquée basée sur des protocoles de communication développés tel que le CAN et l'OBD.

À travers l'utilisation de composants accessibles comme les capteurs, la carte Arduino, le module MCP2515, l'affichage HMI par un écran TFT, nous avons réussi à concevoir une solution capable de lire et d'analyser en temps réel les données critiques du véhicule, tout en offrant une visualisation intuitive des paramètres essentiels tels que la température du moteur et celle de la batterie, l'état de charge de la batterie ou la détection des objets.

Des données récoltées de différents capteurs utilisés une méthode basique de diagnostic a été développé, basé sur des seuils pour chaque grandeur. La simplicité de cette méthode a montré de bons résultats lors du diagnostic Les essais en laboratoire effectués ont permis de valider le bon fonctionnement du système de diagnostic développé.

La capture des trames CAN, leur décodage ainsi que l'affichage des résultats ont été réalisés sans problème. Bien qu'il reste encore à approfondir l'expérimentation sur un véhicule réel via le port OBDII, les résultats obtenus confirment la faisabilité de notre solution.

Bibliographie

[1] Ehsani, M., Gao, Y. et Ebrahimi, K.: « Modern Electric, Hybrid Electric, and Fuel Cell Vehicles », CRC Press, 2018.

[2] An, Y., Yang, B., Park, J., Lee, J., & Park, K. (2023). Analysis of Energy Flow in a Mid-Sized Electric Passenger Vehicle in Urban Driving Conditions. World Electric Vehicle Journal, 14(8), 218.

[3] **Fecih, L.** (2022). Les principales causes des accidents de la circulation routière et les mesures d'atténuation en Algérie (Rapport de licence, Université Badji Mokhtar-Annaba). Université Badji Mokhtar-Annaba.

[4] **北湾南巷**. (2024, 27 juin). **汽**车电机控制器详解 [Technical article]. **极**术社区 (**汽**车电子与软件).

[5] EDN Asia : « BMS design considerations for EV manufacturers », EDN Asia, [en ligne], https://www.ednasia.com/bms-design-considerations-for-ev-manufacturers/, consulté en juin 2025.

[6] Anonyme : « Schéma système embarqué », Google Images, [en ligne], https://images.app.goo.gl/2rppBsys2ZCNyVjC7, consulté en juin 2025

[7] Conquer Electronics : « Electric Vehicle Fuses », Conquer Electronics Co., Ltd., [en ligne], https://www.conquer.com.tw/en-us/media/newsinfo?NId=35, consulté en juin 2025.

[8] ResearchGate. (s. d.). « Diagramme fonctionnel de l'unité de contrôle du véhicule [Image].» Extrait de https://www.researchgate.net/figure/Block-diagram-of-the-Vehicle-Control-Unit-23Main-functions-of-VCU-Vehicle-driving_fig1_342279849

[9] InHand Networks : « Qu'est-ce que le CAN Bus ? », InHand Networks – Blog, [en ligne], https://www.inhand.com/fr/support/blogs/what-is-canbus/, consulté en juin 2025

[10] FlexiHub : « On-Board Vehicle Diagnostics », FlexiHub – Blog technique, [en ligne], https://www.flexihub.com/on-board-vehicle-diagnostics/, consulté en juin 2025.

[11] Denton, T.: « Advanced Automotive Fault Diagnosis (3e édition) », Routledge, 2016.

[12] iCarsoft France. (2021). L'importance du diagnostic moteur pour votre automobile [Article de blog]. iCarsoft France.

[14] Electrical safety and reliability: FDD for EV diagnosis. (2025).

[15] Electrical safety and reliability: FDD for EV diagnosis. (2025).

[16] EBYTE. (2021, juillet 7). What is CAN Bus? https://www.fr-ebyte.com/news/522

[17] CSS Electronics. (s. d.). CAN Bus Software & API Tools. https://www.csselectronics.com/pages/can-bus-software-api-tools#void

[18] Auteur inconnu. (s. d.). Les systèmes embarqués dans l'automobile – Travail de Bachelor [PDF non publié].

[19] Techniques de l'Ingénieur. (2019, 10 mai). Historique du bus CAN. https://www.techniques-ingenieur.fr/.../bus-can-s8140/...

[20] TechnologuePro. (s. d.). Cours systèmes embarqués – Bus CAN. https://www.technologuepro.com/.../cours-systemes-embarques-Bus-CAN.htm

[21 FlexiHub. (2023, 9 mars). Qu'est-ce que le bus CAN ?. https://www.flexihub.com/fr/can-bus/

[22] Microcontrollers Lab. (2020, 6 avril). CAN Communication Protocol. https://microcontrollerslab.com/can-communication-protocol/

[23] Dewesoft. (2023, 12 juillet). What is CAN Bus?. https://dewesoft.com/blog/what-is-can-bus

[24] Last Minute Engineers. (s. d.). MCP2515 CAN Module with Arduino. https://lastminuteengineers.com/mcp2515-can-module-arduino-tutorial/

[21] FlexiHub. (2023, 9 mars). Qu'est-ce que le bus CAN ?. https://www.flexihub.com/fr/can-bus/

[25] Xie, Y. (2019). [Titre non précisé]. Journal of Physics: Conference Series, 1176(1), 052059. https://doi.org/10.1088/1742-6596/1176/1/052059

[26] TechnologuePro. (s. d.). Cours systèmes embarqués – Bus CAN. https://www.technologuepro.com/.../cours-systemes-embarques-Bus-CAN.htm

[27] Wikipédia. (2024, 13 février). Bus de données CAN. https://fr.wikipedia.org/wiki/Bus_de_donn%C3%A9es_CAN

[27] Wikipedia. (2024, 13 février). CAN bus. https://en.wikipedia.org/wiki/CAN_bus

[28] CSS Electronics. (s. d.). CAN Bus Introduction and Tutorial. https://www.csselectronics.com/pages/can-bus-simple-intro-tutorial

[29] DunaSys. (2022, 24 mai). Qu'est-ce que le CAN FD ? Guide pour le comprendre. https://www.dunasys.com/article/quest-ce-que-le-can-fd-guide-pour-le-comprendre

[30] EBYTE. (2021, 22 octobre). CAN Bus Overview. https://www.fr-ebyte.com/news/1702

[31] Yard.onl. (s. d.). Décodage d'une trame CAN. https://yard.onl/sitelycee/cours/busCAN/DecodagedunetrameCAN.html

[32] Yard.onl. (s. d.). Décodage d'une trame CAN. https://yard.onl/sitelycee/cours/busCAN/DecodagedunetrameCAN.html

[33] CSS Electronics. (s. d.). CAN Bus Introduction and Tutorial. https://www.csselectronics.com/pages/can-bus-simple-intro-tutorial

[34] EBYTE. (2021, 25 septembre). CAN Protocol Explanation. https://www.fr-ebyte.com/news/1693

[35] International Organization for Standardization (ISO). (2015). ISO 11898-1:2015 – Véhicules routiers — Réseau CAN. https://www.iso.org/fr/standard/66574.html

[36] CAN Explains. (2020, 20 septembre). CAN Bus Explained [Vidéo]. YouTube. https://www.youtube.com/watch?v=AwDflMtFF00

[37] Geotab. (2023, 9 janvier). OBD-II expliqué. https://www.geotab.com/fr/blog/obd-ii/

[38] FlexiHub. (2023, 5 mai). OBD2 Connector Pinout. https://www.flexihub.com/oobd2-pinout/

[39] Tecnologix. (s. d.). CSS Electronics OBD DBC File – Decode your car data. https://www.tecnologix.it/en/css-electronics-odbc-dbc-file-decode-your-car-data.html

[40] Wikipédia. (2024, 15 janvier). Diagnostic embarqué (automobile). https://fr.wikipedia.org/wiki/Diagnostic_embarqu%C3%A9_(automobile)

[41] Wikipedia. (2024, 1 février). OBD-II PIDs. https://en.wikipedia.org/wiki/OBD-II_PIDs

[42] Geotab. (2023, 7 février). Code DTC (Diagnostic Trouble Codes). https://www.geotab.com/fr/glossaire/code-dtc

[43] CSS Electronics. (s. d.). OBD2 Explained – Simple Intro. https://www.csselectronics.com/pages/obd2-explained-simple-intro

[44] Bosch Aftermarket. (s. d.). KTS 465 – Outil de diagnostic ECU. https://www.boschaftermarket.com/be/fr/equipement/diagnostic-ecu/outils-de-diagnostic-ecu/kts-465/

[45] VXDAS. (s. d.). Launch X431 Pro 5. https://www.vxdas.com/ar/products/launch-x431-pro-5

[46] Amazon. (s. d.). Forscan Scanner ELMconfig FoCCCus Diagnostic. https://www.amazon.com/Forscan-Scanner-ELMconfig-FoCCCus-Diagnostic/dp/B07MQ8GHG3

[47] OBDLink. (s. d.). OBDLink MX+ Product Page. https://www.obdlink.com/products/obdlink-mxp

[48] https://medium.com/@thripuraanuchowdary/introduction-to-ai-ml-dl-8f644baa4103

[49] https://www.oracle.com/fr/database/deep-learning-machine-learning-intelligence-artificielle

[50] CSS Electronics. (n.d.). Projects overview. CSS Electronics. https://www.csselectronics.com/projects.