Augustin LAPRAIS

Yuxin MU

Audrey NICOLLE

Julien PEROL

Corentin PICAHT

CPE Lyon

Semestre 2023

Equipe 3

# Projet Majeur ESE

# Réalisation d'un robot « servante »

# Rapport de pré-étude

Tuteur : Serge NICOLLE

# Table des matières

**Introduction** ..... 3

**I) Présentation du projet** ..... 3

**II) L'équipe** ..... 5

**III) Les différentes tâches à effectuer** ..... 5

**1. Hardware - Réalisation des cartes électroniques** ..... 5

**a. Carte d'alimentation** ..... 5

**b. Carte pour contrôler les moteurs** ..... 6

**c. PCB Interface capteur** ..... 7

**d. Détection des obstacles et suivie de l'utilisateur en utilisant un LIDAR IR.** ..... 7

**2. Software** ..... 8

**a. Mise en mouvement du robot via le séquenceur avec la carte STM32 esclave** ..... 8

**b. Carte maître Stm32** ..... 9

**IV) Planification du projet** ..... 10

**Conclusion** ..... 10

**Annexe :** ..... 11

# Introduction

Avec le développement rapide de la science et de la technologie, les robots de service sont de plus en plus utilisés dans divers domaines. Ce projet vise à développer un robot “servante” de service technique CPE, dont la fonction principale est d'aider les techniciens à gérer les outils et l'équipement sur les chantiers de construction. Non seulement le robot reconnaît et suit le technicien, mais il effectue également des opérations en fonction de ses besoins, ce qui permet d'accroître l'efficacité.

Ce rapport détaillera le processus de conception et de mise en œuvre de ce robot de service, y compris l'architecture matérielle, le développement logiciel, l'intégration des capteurs ainsi que la sélection du protocole de communication. Nous présenterons les rôles des trois différentes cartes de circuits imprimés (PCB) et les traitements à accomplir à l'aide de cartes STM32. En outre, le rapport abordera également la conception de l’interface homme-machine du système.

Grâce à la mise en œuvre de ce projet, nous espérons démontrer le potentiel des robots de service dans des applications réelles et jeter les bases pour des améliorations technologiques et des extensions fonctionnelles futures.

*NB : Ce document est un rapport de pré-étude, les choix de conceptions qui vont y être présentés peuvent être amenés à changer dans le futur en fonction de l'avancée du projet.*

# I) Présentation du projet

L’objectif de ce projet majeur est de réaliser un robot “servante” pour le service technique de CPE. Le robot doit être capable de reconnaître un technicien (électricien ou plombier), de le suivre sur le chantier puis d'adapter ses actions aux besoins du technicien, sans gêner les autres travailleurs. Afin de mener à bien ce projet, nous allons réaliser un prototype du robot “servante”, dans lequel nous allons implémenter les fonctionnalités suivantes :

- Déplacement : Le robot doit être capable de se déplacer dans l’espace en utilisant 2 moteurs BLDC.
- Reconnaissance utilisateur : Le robot doit pouvoir identifier un technicien via un badge à puce RFID.
- Suivi d’utilisateur : Le robot doit pouvoir suivre le technicien en utilisant un système de reconnaissance à lumière infrarouge.
- Évitement d’obstacle : Le robot doit être capable de détecter les obstacles sur sa trajectoire et de réagir en conséquence.
- Manipulation d’outils : Le robot doit pouvoir manipuler différents outils à l’aide d’un bras mécanique.
- Retour automatique : Le robot doit pouvoir revenir à sa base lorsque ces batteries sont trop faibles.
- Interface utilisateur : Il doit être possible d'interagir à distance avec le robot via une interface utilisateur.

Chaque partie présentée ci-dessus sera conçue de manière à pouvoir valider leur fonctionnement par des tests unitaires. Dans le cas où le système entier ne serait pas fonctionnel, ces tests unitaires seront réalisés lors de la recette.

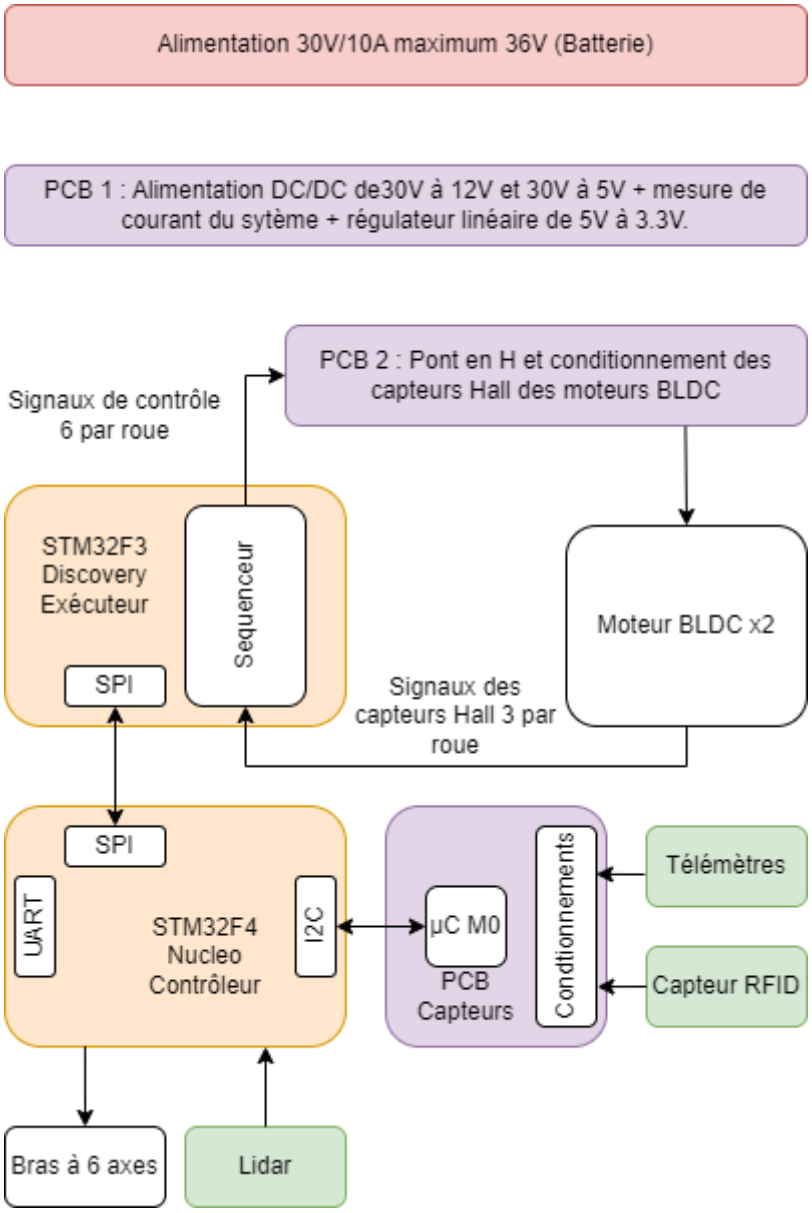



*Figure-1 : Schéma bloc du système à concevoir (robot servante).*

Le schéma bloc ci-dessus est une première représentation physique de notre système. Le système sera alimenté par une alimentation externe de 30V et devra fournir au reste du circuit une alimentation de 5V et 12V. Pour une meilleure lisibilité du schéma, ces lignes d'alimentation ne sont pas représentées. Le robot sera constitué de 3 microcontrôleurs, l'un est dédié au contrôle des moteurs BLDC pour gérer le déplacement du robot en pilotant un pont en H (STM32F3). L'autre sera utilisé pour interagir avec les capteurs (M0). Puis le dernier microcontrôleur sera le superviseur du système sur lequel l'algorithme principal sera codé, il devra aussi recevoir les commandes de l'utilisateur via un module radio et commander le bras mécanique (STM32 NUCLEO).

Les capteurs utilisés sont :

- Le capteur RFID pour la reconnaissance de badge.
- Le lidar pour détecter les obstacles et suivre l'utilisateur à l'aide d'un émetteur infrarouge placé sur celui-ci.
- Les télémètres à ultrasons pour détecter les obstacles ce qui nous permettra d'avoir une redondance et dû à leur facilité d'implémentation de plus facilement commencer l'implémentation de mouvement complexe prenant en compte les obstacles.
- Les capteurs Hall pour alimenter correctement les moteurs sans balais et aussi pour réaliser de l'odométrie.

L'interface homme machine sera implémentée sur un ordinateur et il sera possible d'envoyer des instructions à distance à l'aide d'un module radio ZIGBEE. On peut par exemple imaginer des instructions de démarrage et d'arrêt du robot ou encore des instructions pour activer le suivi de personne. Pour communiquer entre les différents blocs du système, différents protocoles de communication nous ont été imposés (I2C et SPI). Pour le module radio nous avons choisis le protocole UART car il est facile à implémenter et utiliser. Pour ce projet il nous a aussi été demandé de réaliser 3 cartes électroniques : le pont en H pour contrôler les moteurs, le microcontrôleur M0 avec les capteurs et la carte d'alimentation avec la mesure de la tension de la batterie pour déterminer l'état de celle-ci, à l'aide d'un logiciel de conception de circuit imprimé (Kicad).

# II) L'équipe

Notre équipe est composée de cinq personnes :

- Augustin LAPRAIS - Responsable matérielle
- Yuxin MU - Responsable qualité
- Audrey NICOLLE - Chef de projet
- Julien PEROL - Responsable communication
- Corentin PICHAT - Responsable qualité

# III) Les différentes tâches à effectuer

## 1. Hardware- Réalisation des cartes électroniques

### a. Carte d'alimentation

Tout d'abord, nous avons besoin d'une carte pour alimenter le reste de notre système à partir de la batterie 36V de celui-ci. Nous avons besoin d'une tension de 12V et de 5V pour alimenter les différents composants du système avec principalement les cartes STM32, le contrôleur du pont en H et l'alimentation de la carte pour les différents capteurs. Pour cela nous allons réaliser une carte électronique possédant deux abaisseurs de tension réalisés avec le composant TPS40200. Ce composant nous a été imposé pour nous permettre de visualiser les différents éléments à mettre en place qui sont, avec des composants plus récents, de nos jours directement intégrés à celui-ci. En effet, pour réaliser ces deux abaisseurs nous allons mettre en place le schéma électrique suivant, figure 2 :

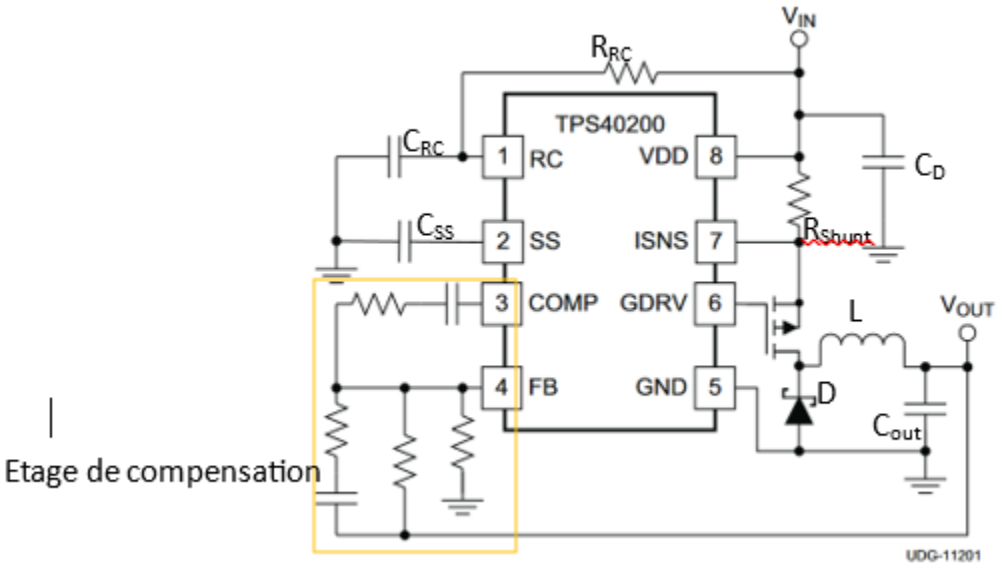



*Figure-2 : Schéma électrique d'un abaisseur de tension réalisé avec le composant TPS40200.*

Nous allons donc dimensionner deux abaisseurs en utilisant ce schéma pour obtenir une tension de 12V et une tension de 5V qui seront fournies au reste du circuit. Pour plus d'informations sur le schéma électrique de ce circuit, voir l'annexe page 11.

Nous allons aussi mettre en place une mesure de tension pour pouvoir déterminer l'état de charge de la batterie. Pour cela, nous allons réaliser un pont diviseur de tension, puis nous fournirons la tension obtenue à un ADC de la STM32 maître pour agir en conséquence si la batterie est déchargée. Nous réaliserons le schéma suivant, figure 3 :

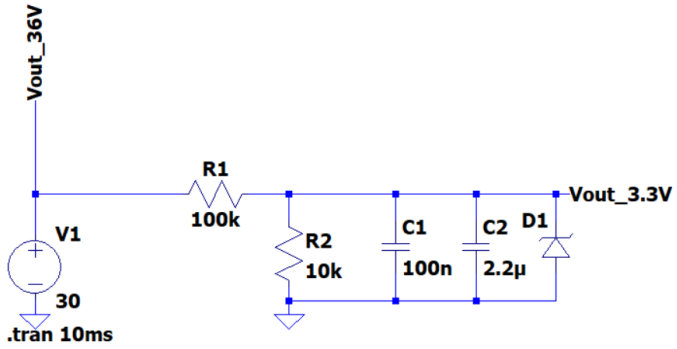



*Figure 3 : Schéma électrique pour mesurer la charge de la batterie.*

Dans ce schéma R1 et R2 forme le pont diviseur de tensions. Le condensateur C1 permet de filtrer les hautes fréquences parasites et le condensateur C2 les basses fréquences parasites qui pourraient endommager l'entrée 3.3V de l'ADC de la STM32 NUCLEO branché sur la sortie Vout_3.3V. La diode Zener est ici pour protéger l'ADC si la tension Vout_3.3V dépasse 3.3V. En effet, à partir de 3.3V, elle deviendra passante en inverse. Correctement dimensionné, ce montage ne devrait pas perturber le reste du circuit. Ainsi, on pourra traiter, à l'aide de la carte STM32 mère, la tension de la charge et agir en conséquence lorsque celle-ci est déchargée.

Cette carte électronique sera réalisée par Audrey et Yuxin.

### b. Carte pour contrôler les moteurs

Pour piloter chacun des moteurs BLDC (Brushless DC motor) on va devoir utiliser un pont en H triphasé car les moteurs BLDC ont trois pôles de commandes. Le pont en H est composé d'un ensemble de transistors formant 3 portes. Pour faire tourner un moteur, chaque transistor doit être activé dans un certain ordre afin de former une séquence. Le changement de comportement des transistors pour passer à la séquence suivante est déclenché par le changement d'état des signaux des capteurs HALL (capteurs magnétiques). Ces capteurs indiquent la position du rotor dans le moteur, ils permettent donc d'indiquer le moment où la polarité des bobines du moteur doit être inversée. Une fois la carte réalisée, la carte STM32F3 fera office de séquenceur pour le moteur. Ce contrôle moteur est schématisé ci-dessous figure 4 :

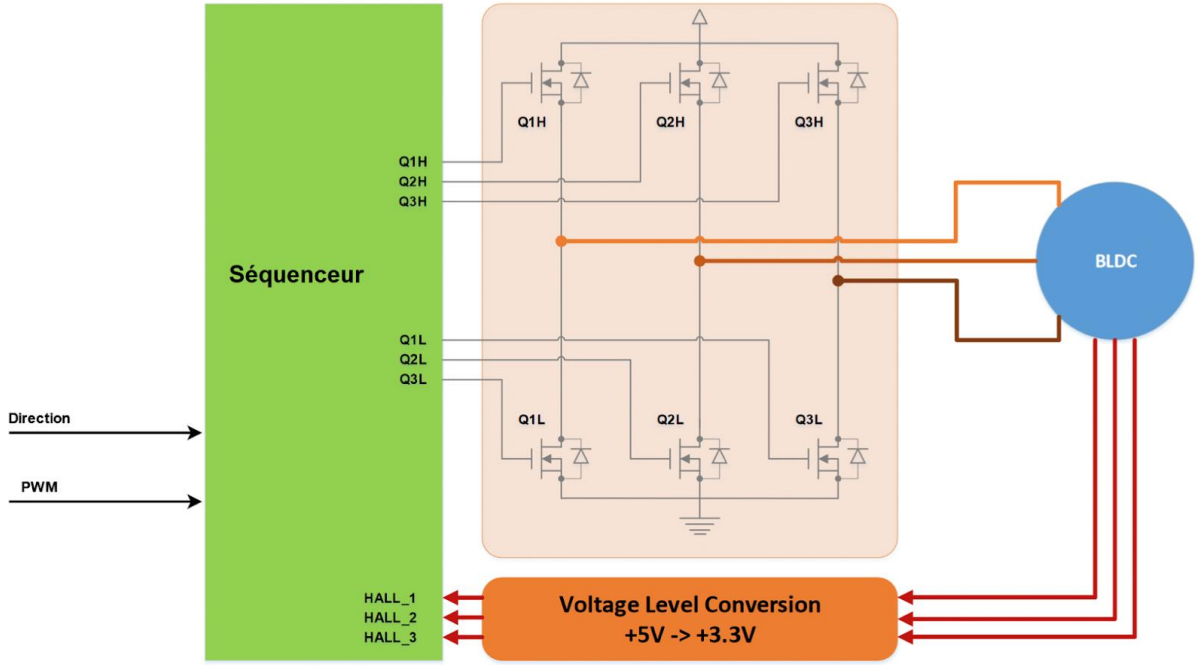



*Figure 4 : Schéma du contrôle des moteur issu de la documentation de CPE.*

Les capteurs HALL intégrés au moteur BLDC nous permettront aussi de réaliser de l'odométrie. En effet, on peut utiliser ces capteurs hall d'une manière semblable à une roue crantée mais qui ne possèderait que 3 pas. L'odométrie nous permettra de connaître la distance parcourue et de diriger le robot vers sa base lorsque l'utilisateur a fini de s'en servir.

Cette carte électronique sera réalisée par Augustin.

### c. PCB Interface capteur

Cette carte sera composée d'un microcontrôleur M0 fourni par l'école et des différents conditionnements des capteurs. Les capteurs que nous allons conditionner sont des télémètres à ultrasons et un capteur RFID. Les conditionnements de ces capteurs n'ont pas encore été réalisés. Après avoir été conditionnés, ces signaux seront envoyés sur la carte STM32 mère via le microcontrôleur M0 pour traiter les informations pour agir en conséquence. Les capteurs télémètres utilisés seront ceux fournis par l'école, soit des capteurs HC-SR04. Ce capteur est alimenté par une tension 5V et peut détecter les obstacles entre 2cm et 4 m sur un angle de 15 degrés. Pour le capteur RFID pour la détection du personnel et l'identification des caisses, nous utiliserons aussi un des capteurs fournis par l'école, le capteur RC522 . Il permet donc la lecture des tag NFC et possède une interface SPI qui pourra donc être directement contrôlée par le périphérique SPI du microcontrôleur M0 sur notre carte.

Ce PCB sera réalisé par Julien et Corentin.

### d. Détection des obstacles et suivie de l'utilisateur en utilisant un LIDAR IR.

Comme dit précédemment nous allons utiliser le Lidar pour détecter les obstacles dans l'environnement mais aussi pour suivre l'utilisateur. En effet, le Lidar RPLIDAR A1 fourni par l'école utilise des rayons infrarouges pour détecter les obstacles. Celui-ci envoie un signal infrarouge et selon le temps qu'il met pour recevoir le signal réfléchi, il en déduit la distance de l'obstacle. En plaçant un émetteur infrarouge sur l'utilisateur, lorsque le lidar fera sa mesure en direction de celui-ci il recevra directement un signal IR en retour ce qui donnera donc un faux positif avec une distance à l'obstacle faible. Pour être sûr que cet obstacle est bien notre utilisateur, l'émetteur IR possède une fréquence d'émission plus faible que la fréquence à laquelle le Lidar effectue ses mesures dans une direction. Ainsi, si l'on détecte des variations de distances significatives à la fréquence de l'émetteur dans une certaine direction, on peut en conclure que c'est notre utilisateur. On ne confondra pas via ce mécanisme notre utilisateur avec un obstacle qui viendrait surgir subitement devant le robot ou bien avec un autre signal IR de même longueur d'onde.

Pour pouvoir effectuer ceci, nous avons besoin que l'émetteur IR émette dans une longueur d'onde proche de celle du Lidar qui est de de 785nm. De plus, sa portée de détection est de 12m, on peut donc chercher un émetteur capable d'émettre jusqu'à 10-15m pour maximiser l'efficacité de notre système. Ensuite, le lidar, selon l'utilisation qu'on en fait, a une fréquence de détection sur un angle d'environ 8kHz. Il nous faut donc produire un signal modulé avec une fréquence inférieure à celle-ci. Cela permettra de filtrer les signaux indésirables qui ne correspondent pas à la modulation prévue et ainsi de reconnaître notre utilisateur. Pour cela, en plus de l'émetteur, on utilisera un oscillateur NE555 pour fournir un signal PWM à celui-ci. Enfin, on ne souhaite pas fausser les autres mesures du Lidar, il est donc nécessaire d'avoir un signal émis dans une direction précise pour limiter les interférences avec les autres signaux émis par le lidar. On utilisera la LED IR ADL-78901TL qui remplit ces conditions, voir les caractéristiques de celle-ci dans le tableau suivant :

| | |
|---|---|
| Longueur d'onde | 785 nm |
| Portée du signal émis | ~10m (dépend de l'environnement) @1kHz |
| Tension Vf | 1.5 à 2V |
| Consommation | 200mW à 100mA pour 2 V avec une PWM de 1Khz et un DC de 50% |

# 2. Software

Afin de garantir un fonctionnement en phase de tous nos composants, de nombreux comportements sont remplis par des solutions softwares. Il s'agit principalement d'utiliser les microprocesseurs de nos cartes matérielles à des fins de supervision des autres pièces, souvent purement analogues. Les 2 cartes STM32 s'imposent donc dans la solution software, avec des potentiels ajouts plus tard dans le projet, si le temps est suffisant. Une carte « esclave » va interagir directement avec les moteurs, ainsi qu'avec les deux ponts en H qui les alimentent. Une carte « maître » servira de cerveau, et prendra les décisions qui conviennent en fonction des retours des capteurs, des commandes utilisateur et autres informations complémentaires de la carte esclave. La répartition des différents traitements à faire entre les membres de l'équipe n'a pas encore été complètement décidée.

## a. Mise en mouvement du robot via le séquenceur avec la carte STM32 esclave

La carte esclave va piloter les moteurs et les ponts en H les alimentant. En premier lieu, les moteurs vont sortir des informations de position sur leur rotor. Il est important de pouvoir les interpréter, ces retours étant essentiels pour garantir une rotation fluide et réactive.

Il y a trois capteurs Halls par moteur, tous actifs à l'état bas. Un capteur est actif lorsque le rotor lui fait physiquement « face » (en terme magnétique). Par le retour des trois capteurs, on peut savoir la position précise du rotor, et ainsi la comparer à la position théorique associée à notre commande. Cette détection d'erreur est simple d'utilisation et rapide à parvenir. En termes de software, il s'agit simplement d'une interprétation et d'une sécurité en code C. Par la suite, il nous faut pouvoir piloter 2 ponts en H triphasés pour alimenter les moteurs. Un pont à trois paires d'entrées (une pour chaque stator), une paire étant formée de l'entrée Haute et de l'entrée Basse. Pour piloter correctement une roue, il faut un ordre précis de déclenchement des stators (d'où le terme *Sequencer*). En termes de software, cela peut être effectué de deux manières : en codant une table de vérité directement ou bien en codant une machine d'état (qui suivrait la même table).

| | Hall sensors | | | Switchs | | | | | | Phases | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Phase | H3 | H2 | H1 | Q1L | Q1H | Q2L | Q2H | Q3L | Q3H | P1 | P2 | P3 |
| I | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | $+V_m$ | Gnd | NC |
| II | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | $+V_m$ | NC | Gnd |
| III | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | NC | $+V_m$ | Gnd |
| IV | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | Gnd | $+V_m$ | NC |
| V | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | Gnd | NC | $+V_m$ |
| VI | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | NC | Gnd | $+V_m$ |

*Figure 5 : Table de vérité à appliquer au Pont en H pour une rotation horaire.*

Pour l'instant, la solution choisie est celle de la table de vérité, en raison de sa simplicité d'implémentation et sa rapidité de déploiement. Cependant, cela est compensé par une sécurité plus faible (au moins dans un premier temps) sur le bon déroulement de l'ordre de déclenchement. Finalement, la carte esclave doit également être capable de recevoir des commandes de déplacement. Il lui revient de convertir ces commandes en comportement adéquat. Cela implique par exemple des commandes de direction et/ou de distance, un sens de parcours, et une vitesse. Il est possible que d'autres commandes soient employées dans le projet plus tard.

Au total, cela correspond donc à 11 broches de la carte juste pour commander les moteurs. Cela est l'une des raisons pour laquelle nous avons besoin d'une carte esclave en premier lieu pour garantir un contrôle et un temps de réaction correct pour le mouvement, et ne pas manquer de broches plus tard.

### b. Carte maître Stm32

La carte maître se charge d'interpréter les données des capteurs et de prendre les décisions. Toute IHM va également passer par elle. Le comportement du software sera idéalement très complexe, comme l'implique la notion de réaction « intelligente ». Cependant, tester de multiples comportements signifie que quasi toutes les parties singulières du robot soient fonctionnelles et intégrées. Puisque nous avons décidé dans un premier temps de nous concentrer sur le fait de délivrer des parties unitaires fonctionnelles, il est important de noter qu'il s'agira d'une partie software réalisée tard dans le projet.

Trois éléments principaux sont en directe interaction avec nos carte maître : les capteurs du robot (tous les capteurs, sauf Halls), la carte esclave, et enfin l'utilisateur (IHM). Les capteurs vont fournir des informations de natures différentes à travers des procédés parfois différents. Cela implique le développement d'un software communication (UART, I2C, SPI…) à multiples facettes. Dans le cadre d'une STM32, il est bien sûr possible de s'appuyer en partie sur les librairies adaptées, mais pas toujours optimisées. L'interprétation de ces données est un élément clé dans la réaction à choisir par la carte. La réaction s'exprime en partie par les commandes que la carte maître va transmettre à la carte esclave (en SPI). Comme mentionné précédemment, il s'agit des commandes de pilotage du robot, soit tout ce qui peut toucher à son déplacement.

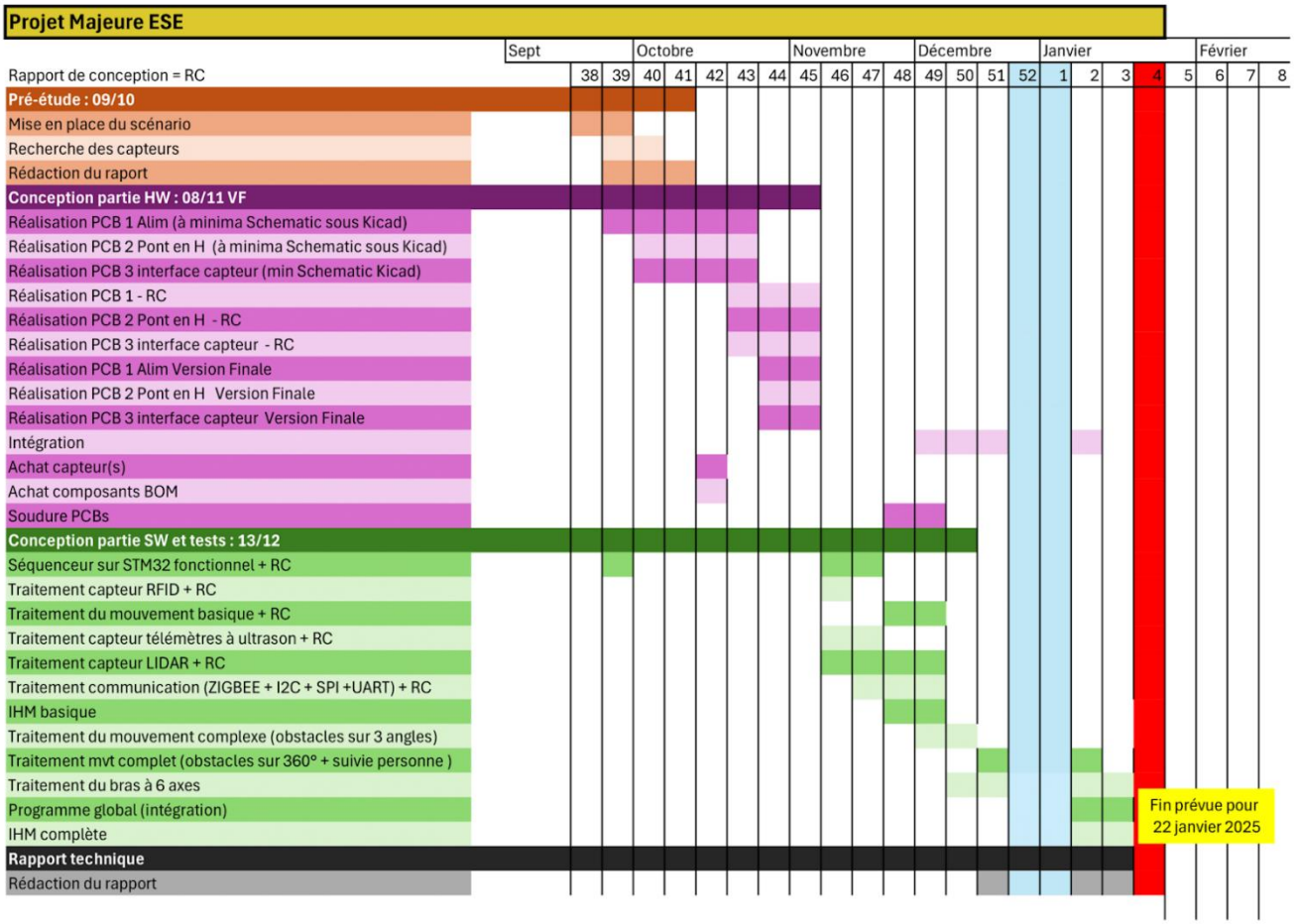
Finalement, le robot doit pouvoir faire preuve d'une certaine interactivité avec l'utilisateur. Cette dernière sera dans un premier temps limité, principalement des commandes manuelles de mouvement. Cela passe par un module radio, ce qui implique sa mise au point software à un moment (UART). D'autres éléments, tels qu'un bras mécanique et une centrale inertielle sont potentiellement ajustables par la suite, mais ce sont des priorités basses pour l'instant, étant donné leur complexité.

En somme, cela occuperait un minimum de 8 broches différentes pour la gestion de tous ces éléments. Cependant, plus la prise de décision est codée en profondeur et plus la complexité augmente, ce qui est surtout au détriment du

temps de réaction du robot. La carte maître sera développée dans un premier temps avec des fonctionnalités très réduites, pour garder notre attention sur les éléments unitaires.

## IV) Planification du projet

Nous avons réalisé un diagramme de Gantt pour nous aider à la planification du projet. Le diagramme présenté ci-dessus est un diagramme global. Certaines activités seront découpées dans des sous diagramme, par exemple la réalisation des PCB qui contient plusieurs étapes. Ce diagramme est amené à évoluer au fur et à mesure que nous appréhendons mieux les sujets de notre projet que nous n'avons pas encore étudié en cours.




## Conclusion

Ce rapport de pré-étude correspond aux premières idées de conception et réponses matérielle que l'on peut appliquer dans le cadre de ce projet. Il est cependant important de retenir qu'il ne s'agit ici que d'un premier jet. Des parties sont voués à changer, qu'il s'agisse d'affinage de fonction ou bien de rajouts d'éléments. Les fonctionnalités déjà décrites sont pour l'instant dans leur état le plus simplifié, et pourront devenir plus complexes dans le futur. Certaines de nos évaluations actuelles peuvent également s'avérer incorrectes, ce qui nous mènerait à changer de composant après-coup. Ce document sera mis à jour progressivement pour aider lors de la création de nos rapports plus complets.

# Annexe :

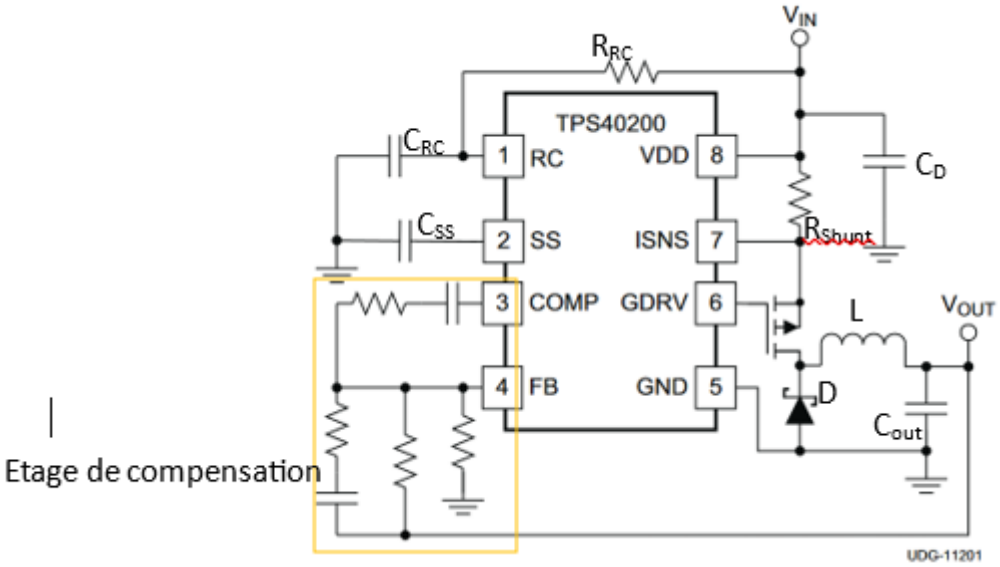



*Figure-2 : Schéma électrique d'un abaisseur de tension réalisé avec le composant TPS40200.*

Pour rappel, l'objectif de ce circuit est de diminuer la tension d'entrée pour obtenir une tension de sortie plus faible stable. Pour cela, à l'aide d'un transistor PMOS, on crée un signal haché qui sera par la suite moyenné par un filtre moyenneur. En fonction de la tension d'entrée fournie, le rapport cyclique du signal haché sera plus ou moins important permettant ainsi de toujours fournir la même tension de sortie quel que soit la valeur de la tension d'entrée. Bien sûr, cette tension d'entrée doit être dans la gamme supportée par le transistor. Par la suite, cette tension hachée est fournie par un filtre moyenneur composé d'une bobine et d'une capacité $C_{out}$. La bobine pendant l'état haut va se charger puis se décharger pendant l'état bas du transistor ce qui nous permet bien d'obtenir un courant de sortie représentant la moyenne du signal haché. Elle est aussi ici pour lisser le courant puisque celle-ci résiste aux variations brusques dues par la commutation du transistor. Elle permet donc réduire les oscillations de courant qui pourrait endommager les composants alimentés par la tension de de sortie de l'abaisseur. Le condensateur réalise le même travail mais pour la tension.

Par ailleurs, pour que le système s'adapte à ces changements, le signal de sortie est fourni au composant TPS40200 qui peut ainsi faire la comparaison entre une tension de référence et Vout. Ainsi à chaque changement de Vin, le système réagit pour faire commuter le transistor de manière à produire le signal haché avec un rapport cyclique adapté pour permettre de toujours fournir la même tension de sortie. Sur cette boucle retour, se trouve aussi un étage dit de compensation. Cet étage sert à ajuster la réponse dynamique et à stabiliser la boucle de rétroaction du convertisseur abaisseur, en optimisant la relation entre stabilité, vitesse de réponse et atténuation des perturbations. Ensuite, nous avons les composants passifs $R_{RC}$ et $C_{RC}$ qui permettent de régler la fréquence du système. Il y aussi la capacité $C_{SS}$ qui permet d'effectuer un "soft-start" du système. En effet, cette capacité force notre système a tout d'abord réaliser la charge de condensateur avant d'entrer dans son régime d'utilisation normale. Cela permet d'éviter tout pics de tension ou de courant qui pourrait endommager les différents composants. Enfin, nous avons la capacité de découplage de la tension d'entrée $V_{in}$. Celle-ci permet d'éviter un endommagement du reste du circuit dû à des variations brusques sur la tension d'alimentation $V_{in}$.