Le cycab de l'INRIA Rhône-Alpes

Gérard Baille, Philippe Garnier, Hervé Mathieu, Roger Pissard-Gibollet

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

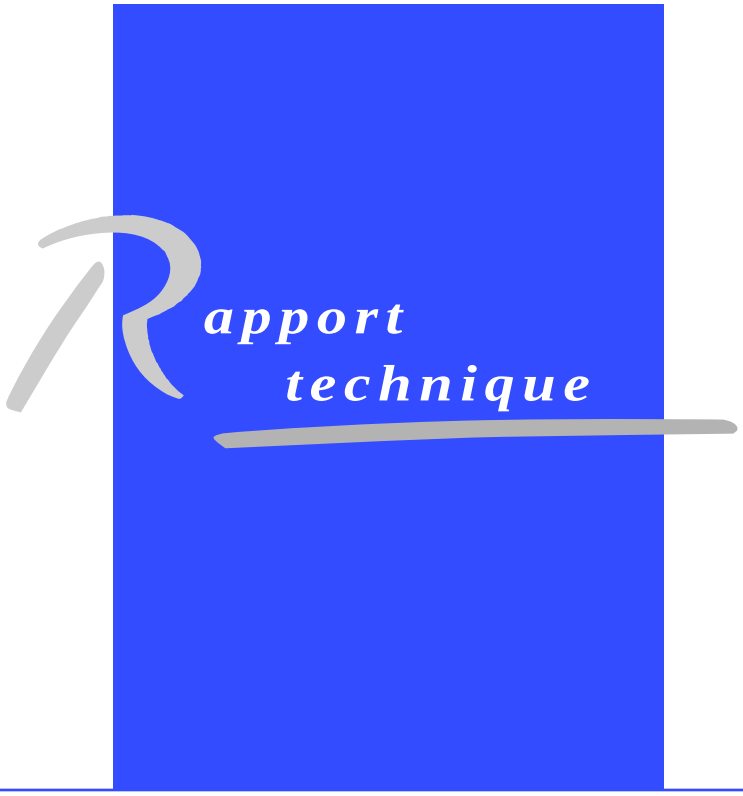
Le Cycab de l'INRIA Rhône-Alpes

Gérard Baille, Philippe Garnier, Hervé Mathieu, Roger Pissard-Gibollet

N° 0229

Avril 1999

THÈME 3



ISSN 0249-0803

Le Cycab de l'INRIA Rhône-Alpes

Gérard Baille, Philippe Garnier, Hervé Mathieu, Roger Pissard-Gibollet

Thème 3 — Interaction homme-machine,
images, données, connaissances
Projet Service Robotique - INRIA

Rapport technique n ° 0229 — Avril 1999 — 87 pages

Résumé :

Dans le cadre de la route automatisée [5], l'INRIA a imaginé un système de transport original de véhicules en libre-service pour la ville de demain - Ce système de transport public est basé sur une flotte de petits véhicules électriques spécifiquement conçus pour les zones où la circulation automobile doit être fortement restreinte. Pour tester et illustrer ce système, deux prototypes, nommés Cycab, ont été réalisé. Le premier est localisé à l'UR de Rocquencourt et le second est localisé à l'UR Rhône-Alpes. Leszek Lisowski, ingénieur-expert SHARP/Praxitéle, a travaillé durant deux ans sur la conception et la réalisation des deux prototypes [3].

Depuis janvier 1998, le Service Robotique [4] est en charge du Cycab de l'INRIA Rhône-Alpes. Les deux Cycab comportent des différences de conception importantes. Ainsi, autant le premier prototype donnait toute satisfaction, autant le second posait d'énormes problèmes de fiabilité. C'est dans ce contexte que l'équipe des Moyens Robotiques a passé l'année 1998 à analyser le véhicule puis le rendre opérationnel. Ce document regroupe les différentes études faites et devrait servir, aussi bien à la maintenance du Cycab de Rocquencourt, qu'à l'industriel [7] choisi pour concevoir le successeur de ces deux Cycabs.

Les deux premiers chapitres présentent une analyse fonctionnelle détaillée du hardware (chapitre 1) et plus précisément du module de commande (chapitre 2). Le chapitre 3 décrit le logiciel embarqué sur les calculateurs du Cycab: les modules MC68332 et la carte MVME162. Finalement, le dernier chapitre décrit les éléments de base pour l'expérimentation comme la modélisation et la calibration du véhicule et la vue du Cycab pour le logiciel de contrôle-commande ORCCAD.

(Abstract: pto)

Unit´e de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN (France)
T´el´ephone : 04 76 61 52 00 - International: +33 4 76 61 52 00
T´el´ecopie : 04 76 61 52 52 - International: +33 4 76 61 52 52

The INRIA Rhône-Alpes Cycab

Abstract: In 1997, a new vehicle called the Cycab, designed specifically for car-free cities, has been presented to the public. This vehicle is entirely under computer control and can be driven manually with a joystick, or can be driven automatically under various modes.

Two Cycabs exist today, one in INRIA Rocquencourt Reseach Unit and one in INRIA Rhône-Alpes Reseach Unit. Leszek Lisowski, INRIA engineer, worked during two years to build these two prototypes.

The Service Robotique at INRIA Rhône-Alpes Reseach Unit has been involved in Rhône-Alpes Cycab operation since 1998. This document resumes all the work done to make this Cycab usable by Reseach Projects.

Hardware is described in chapter 1 and chapter 2, chapter 3 talks about software and chapter 4 explains features useful for experiment approach.

Table des matières

Table des figures

Chapitre 1

Présentation du CyCab Rhône-Alpes

1.1 Introduction

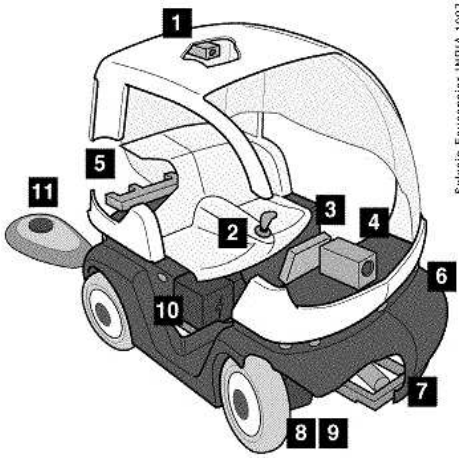
Le CyCab est un prototype de petit véhicule électrique devant servir de plate-forme expérimentale pour les équipes de robotique et vision de l'Unité de Recherche Rhône-Alpes. Il a été construit sur la base d'un chassis tubulaire Andruet raccourci qui supporte une coque conçue par Sylvain Fauconnier et dont les moules appartiennent à l'Inria.

1.2 Caractéristiques générales

Les caractéristiques générales du véhicule sont les suivantes :

- Longueur hors tout : 1,90m
- Largeur hors tout : 1,20 m
- Poids total avec batteries : 300 kg
- motorisation : 4 moteurs électriques de 1 kW
- 4 roues motrices et directrices
- vitesse théorique maximale : 20 km/h
- autonomie : 2 heures d'utilisation continue
- capacité d'accueil : 2 personnes avec bagages
- conduite automatique ou manuelle

– recharge automatique par induction



1. Caméra CCD pour la téléopération
2. Joystick central pour la conduite manuelle sécurisée
3. Écran tactile complétant l'Interface Homme Machine
4. Caméra linéaire pour la localisation et le suivi de cible
5. Balises infrarouges pour l'accrochage immatériel
6. Ceinture de capteurs à ultrasons pour la détection d'obstacles
7. Vérin électrique de direction
8. Un moteur de traction par roue
9. Un frein à commande électrique par roue
10. Quatre batteries de 12 Volts et un gestionnaire automatique de charge
11. Borne de recharge électrique par induction fixée sur la voirie

FIG. 1.1: *Vue éclatée du CyCab*

Pour contrôler et commander ces 9 moteurs (4 de traction, 1 de direction et 4 de frein), une architecture matérielle distribuée a été choisie (figure [1.2]). Elle est constituée de 3 nœuds intelligents pouvant gérer chacun 2 moteurs. Le premier, appelé nœud de direction, gère le joystick et contrôle le vérin motorisé de direction. Deux nœuds de traction, un par train, contrôlent deux moteurs de traction. La commande de frein de parking est unique pour les quatres moteurs. Un PC industriel dont le rôle essentiel est de gérer l'écran tactile de l'IHM, et une carte Motorola MVME162 sous VxWorks complète cette architecture matérielle. Les trois nœuds, le PC et la carte Motorola communiquent entre eux via un bus de terrain Controller Area Network (CAN). Sur ce même bus CAN seront connectés les capteurs ultrasonores ainsi que la caméra linéaire.

1.3 Le chassis

Le chassis du CyCab est dérivé du chassis tubulaire équipant les voiturettes de golf commercialisées par la société ANDRUET SA. Ce chassis supporte les 4 batteries, les 4 blocs roue (roue + moteur + frein mécanique à tambour), les 4 moteurs de frein de parking, les biellettes de direction les reliant mécaniquement entre elles, le vérin motorisé et les boîtiers électroniques.

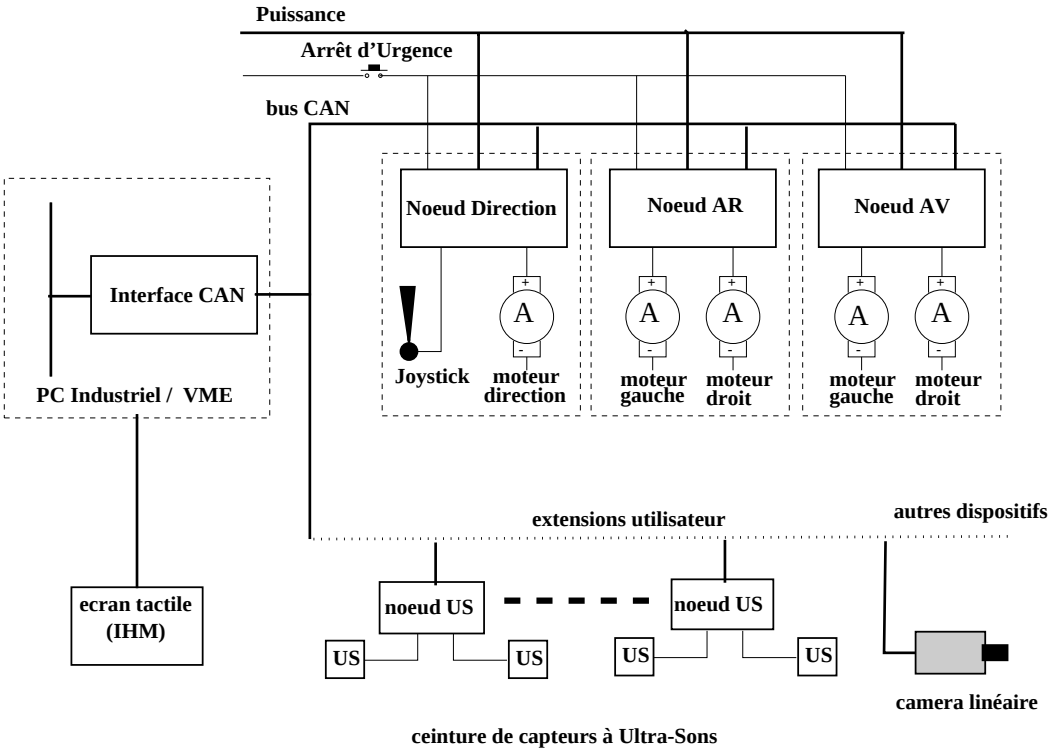


FIG. 1.2: *Synoptique de l'architecture matérielle de Contrôle/Commande*

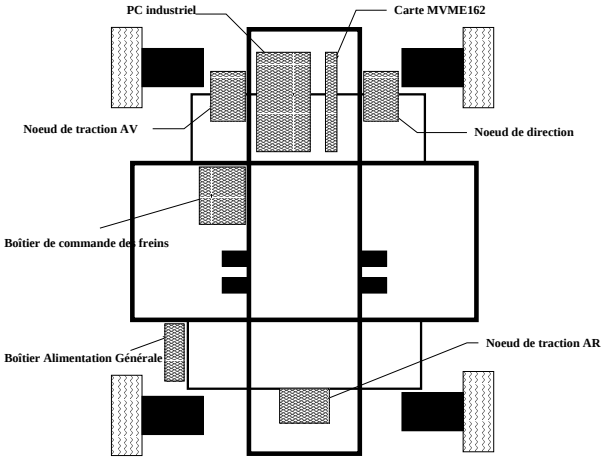


FIG. 1.3: *Architecture matérielle : Implantation sur le chassis*

1.4 Alimentation générale

Toutes les alimentations nécessaires au fonctionnement du CyCab (puissance pour les moteurs, alimentation des nœuds, du PC, de la carte MVME162, de la caméra linéaire, ..) sont fournies par le boîtier d'alimentation générale soit à partir des batteries embarquées, soit à partir de la "prise puissance" tripolaire située sous le boîtier (figure [1.4]).

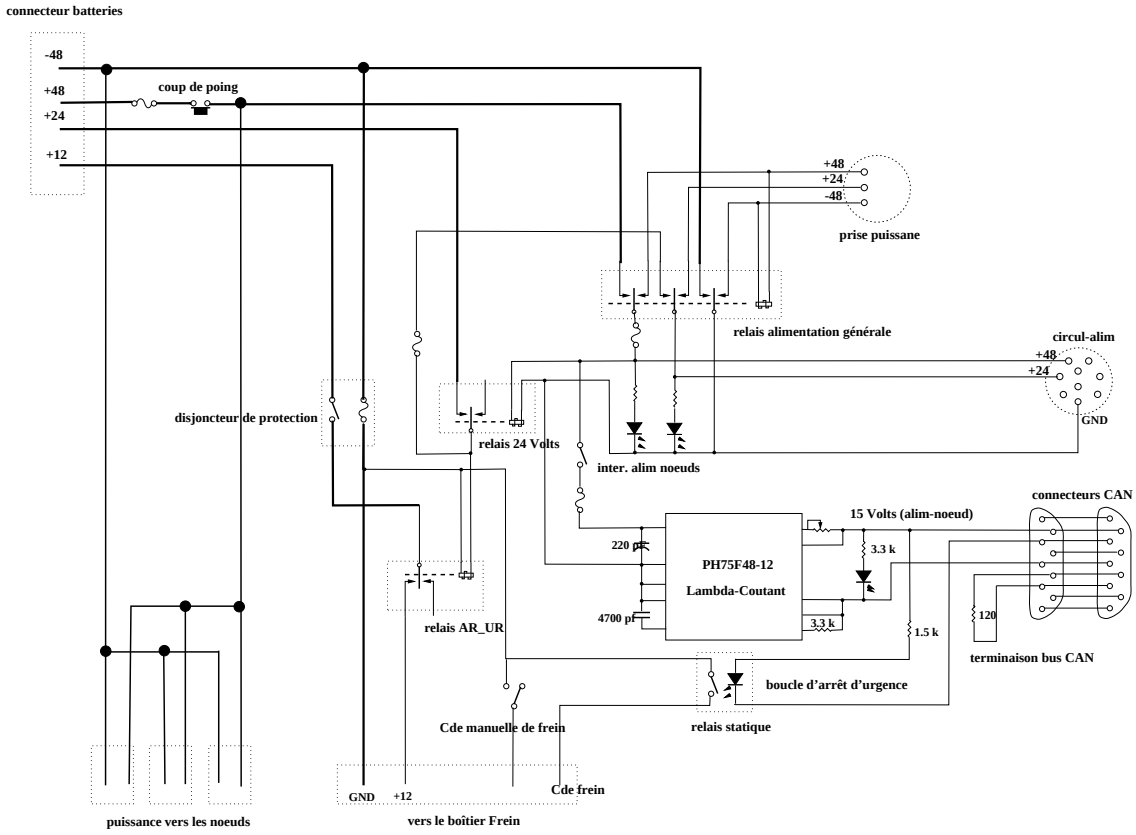


FIG. 1.4: *Boîtier d'alimentation : Schéma fonctionnel*

Le véhicule embarque 4 batteries de 12 Volts qui lui donnent en théorie une autonomie de 2 heures à la vitesse maximale de 20 km/h sur des pentes n'excédant pas 10 %. Les tensions +48, +24, +12 Volts et masse sont disponibles sur le connecteur double PROCONET vers les batteries. L'alimentation des moteurs du CyCab ne se fait que depuis les batteries si l'interrupteur **coup de poing** situé sous le boîtier n'est pas enfoncé. Une **prise Puissance** tripolaire est prévue pour la connection d'une alimentation externe comme par exemple un chargeur délivrant +48 et +24 Volts en sortie. Ceci est utile pour préserver la charge des

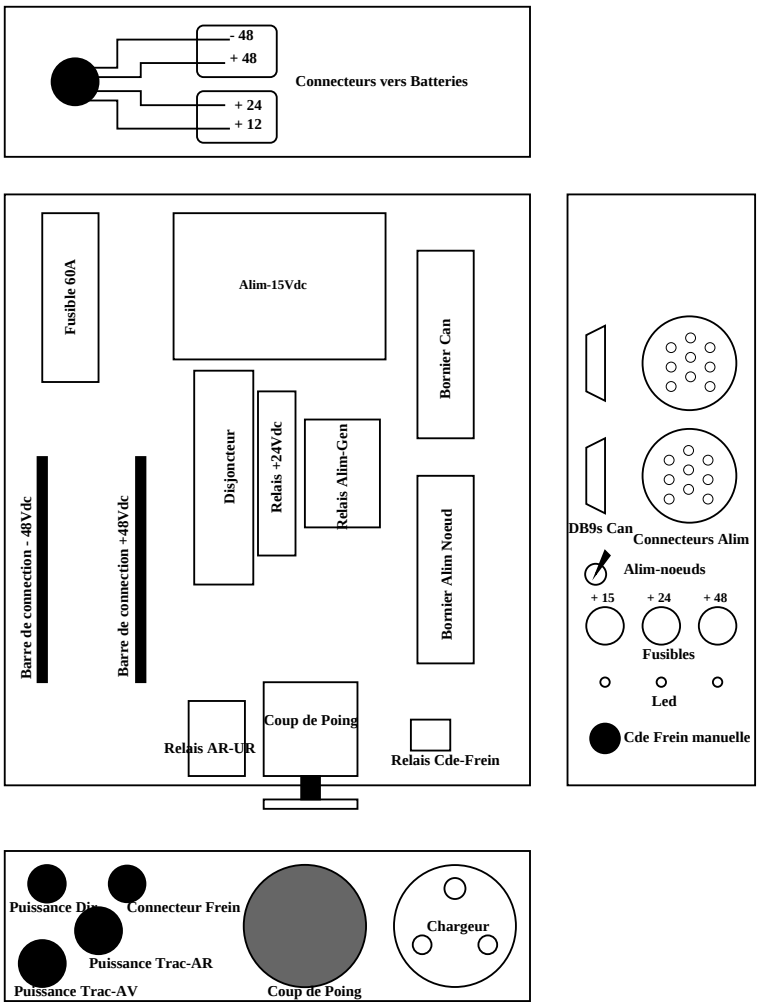


FIG. 1.5: *Boîtier d'alimentation : Implantation*

batteries pendant la phase de mise au point ou le dépannage des nœuds. La présence d'une tension de 48 Volts sur la prise puissance commande le **Relais d'Alimentation Générale** constitué de trois inverseurs assurant le multiplexage des sources d'alimentation pour le **Circuit-Alim** (connecteur circulaire sur le côté du boîtier), le primaire du convertisseur DC DC (**Alim-15Vdc**) fournissant la tension de 15 Volts distribuée aux nœuds à travers le connecteur DB9 CAN, la bobine du **Relais 24 Volts** dont le rôle est d'alimenter (12 Volts) le boîtier de commande des freins par l'intermédiaire d'un relais curieusement baptisé **Relais Arrêt d'Urgence**. L'état de repos (non alimenté) pour les freins est l'état freiné.
Le convertisseur DC DC monté dans le boîtier est un convertiseur Lambda-Coutant du type PH75F48-12 qui à partir d'une tension d'entrée pouvant varier de 36 à 76 Volts, fournit en sortie une tension comprise entre 2 et 28 Volts à $\pm 20\%$, cette tension de sortie étant "programmable" à l'aide d'une résistance et d'un potentiomètre.
Un **Relais Statique** sur la boucle d'arrêt d'urgence (connecteur DB9-CAN) met le véhicule dans l'état freiné s'il n'est pas alimenté (boucle ouverte par l'interrupteur *coup de poing* placé sur la console). En série avec le secondaire de ce relais statique est monté un interrupteur de commande manuelle des freins de parking. Il est monté sur le côté du boîtier sur lequel l'on trouve aussi les fusibles et les indicateurs de présence des alimentations (LED) +48, +24 et +15 Volts. Trois torons de puissance (48 Volts) partent du boîtier d'alimentation générale vers les nœuds (connecteur de puissance).

Remarques :

1. Les moteurs ne peuvent pas être alimentés via la prise puissance.

2. Le chargeur branché sur la prise puissance ne peut pas charger les batteries. Cette recharge ne peut se faire qu'après les avoir déconnectées du boîtier et avec un chargeur ayant un connecteur de sortie PROCONET compatible avec celui des batteries.

3. L'alimentation des moteurs de frein, donc le fait de pouvoir défreiner le véhicule, ne peut se faire qu'à partir des batteries. Ces dernières ne doivent pas être déconnectées même si l'on utilise une alimentation extérieure pour les nœuds.

4. La seule possibilité d'avoir un état défreiné quand les alimentions sont coupées par exemple pour déplacer le véhicule en le poussant, est de couper la puissance (coup de poing) en maintenant enfoncé le bouton de commande manuelle des freins.

1.5 Les moteurs de traction

La chaîne de traction se compose de quatre "moteur-roues" contrôlables séparemment. Les moteurs de traction de type MP 8165/CADDY sont fournis par ANDRUET. Ils sont modifiés par rapport à ceux équipant les voiturettes de golf:

- le rotor a été rebobiné de façon à fonctionner en 48 Volts pour limiter l'intensité du courant dans les MOSFETs (à puissance égale),

- l 'axe du rotor a été prolongé de façon à pouvoir monter un codeur optique incrémental de type HEDS-9140 pour les calculs d'odométrie,
- une thermistance a été rajoutée pour mesurer la température à l'intérieur des moteurs.

1.6 Le vérin motorisé de direction

Pour actionner les biellettes de direction, la crémaillère et le volant ont été remplacés par un vérin électrique. Celui-ci est aussi équipé d'un codeur optique incrémental pour le contrôle du braquage. Il lui est associé un capteur de déplacenment rectiligne de type RECTIP12 de MCB industrie pour pouvoir lire la position initiale du vérin et 2 capteurs inductifs Baumer pour les fin de course.

1.7 Les freins de parking

Chaque bloc roue du CyCab est pourvu d'un frein à tambour mécanique commandable électriquement : un moteur de type essuie-glace de voiture tire ou relâche le câble. Les 4 moteurs de frein sont situés sous le plancher ainsi que le boîtier contenant les relais de commande.
Il était plus ou moins prévu de pouvoir contrôler séparemment chacun des moteurs avec une mesure de la tension exercée sur les câbles avec une jauge de contrainte. Cette fonctionnalité n'a jamais été implémentée et les moteurs ont tous la même commande venant soit d'une sortie Tout ou Rien du nœud de direction, soit d'un bouton poussoir situé sur l'armoire d'alimentation générale, soit de l'arrêt d'urgence et à travers une logique à relais (figure [1.6]).

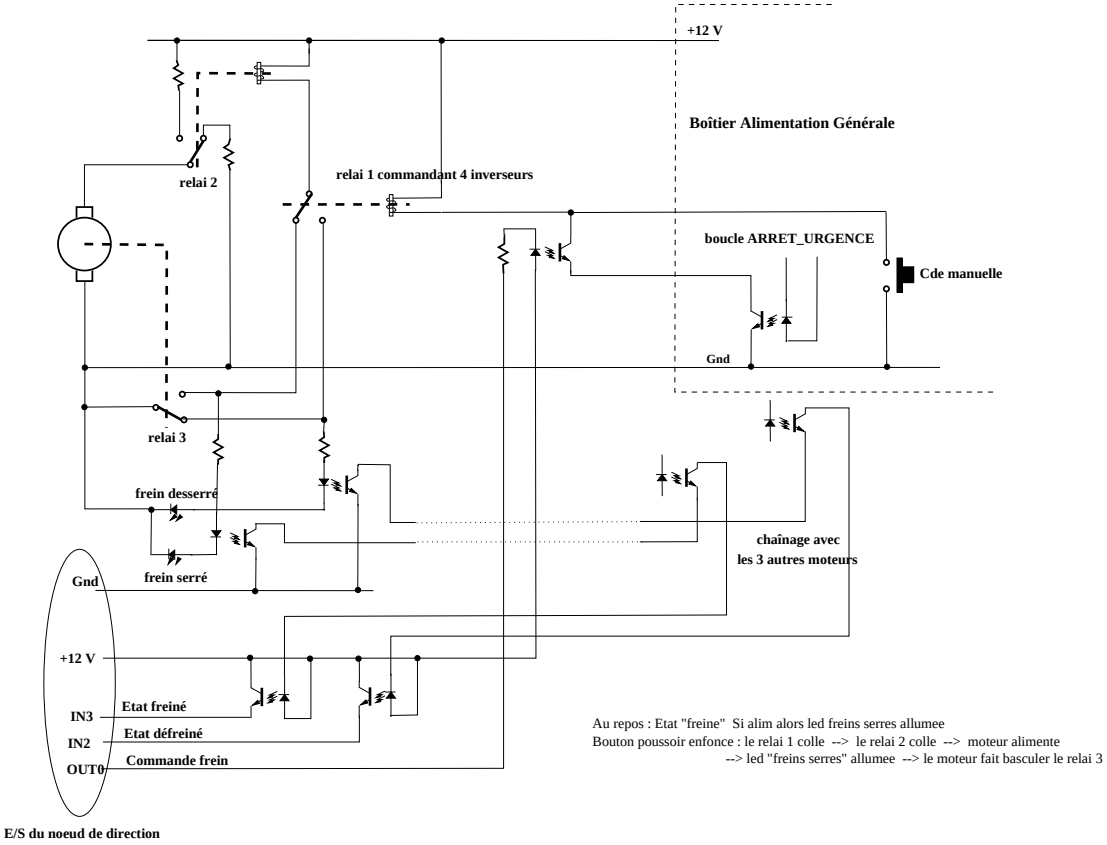


FIG. 1.6: *Freins de parking : schéma de câblage*

1.8 L'Interface Homme Machine

Le conducteur communique avec le véhicule à l'aide d'un joystick de commande centralisée situé entre les deux sièges ce qui permet une prise en main facile à la fois pour les droitiers et les gauchers, et d'un écran tactile intégré dans la console. Sur le sommet de cette console se trouve un bouton *coup de poing* d'arrêt d'urgence et trois diodes électroluminescentes renseignant sur l'état du véhicule :

- la LED de droite de couleur verte indique si le bouton d'arrêt d'urgence est enfoncé (LED éteinte),
- la LED centrale, également verte. indique la présence de l'alimentation générale (15 Volts) vers les nœuds,
- celle de gauche de couleur rouge est allumée s'il y a un dysfonctionnement sur l'un des nœuds. Elle représente le *OU câblé* des états de tous les nœuds.

En fonctionnemnt normal, les 2 LEDs vertes doivent donc être allumées.

1.8.1 Le joystick de commande

En mode de conduite manuelle, le conducteur pilote le CyCab au joystick (réf: RS 162-984) qui est vu par le système matériel de contrôle/commande comme 2 potentiomètres dont la lecture analogique par le nœud de direction simule l'accélérateur et le volant d'un véhicule normal. Il n'y a aucun lien mécanique entre l'organe de commande manuelle (le joystick) et les actionneurs (moteurs de traction et vérin de direction) ("fly by wire")

1.8.2 L'écran tactile

L'écran LCD tactile sert d'écran de contrôle au PC embarqué. Des touches de fonction programmables permettent de charger, démarrer les applications.

1.8.3 La boucle d'arrêt d'urgence

En cas de problème grave ou de danger survenant pendant l'utilisation du Cycab, il est impératif de pouvoir agir rapidement pour bloquer le véhicule en inhibant les commandes des moteurs de traction et de direction et en actionnant les freins de parking. Pour ce faire le Cycab dispose d'un mécanisme d'arrêt d'urgence pouvant être actionné soit par l'interrupteur *coup de poing* situé au dessus de la console, soit par une télécommande radio (série 400) dont le récepteur est monté dans le rack VME et qui actionne un relai normalement fermé. Cette information **ARRÊT d'URGENCE** doit être transmise à tous les nœuds de façon à inhiber les commandes des moteurs. Elle doit donc être chaînée, c'est le rôle de la boucle d'arrêt d'urgence (figure [1.7]) dont l'état *ouvert* est détectée par tous les nœuds.

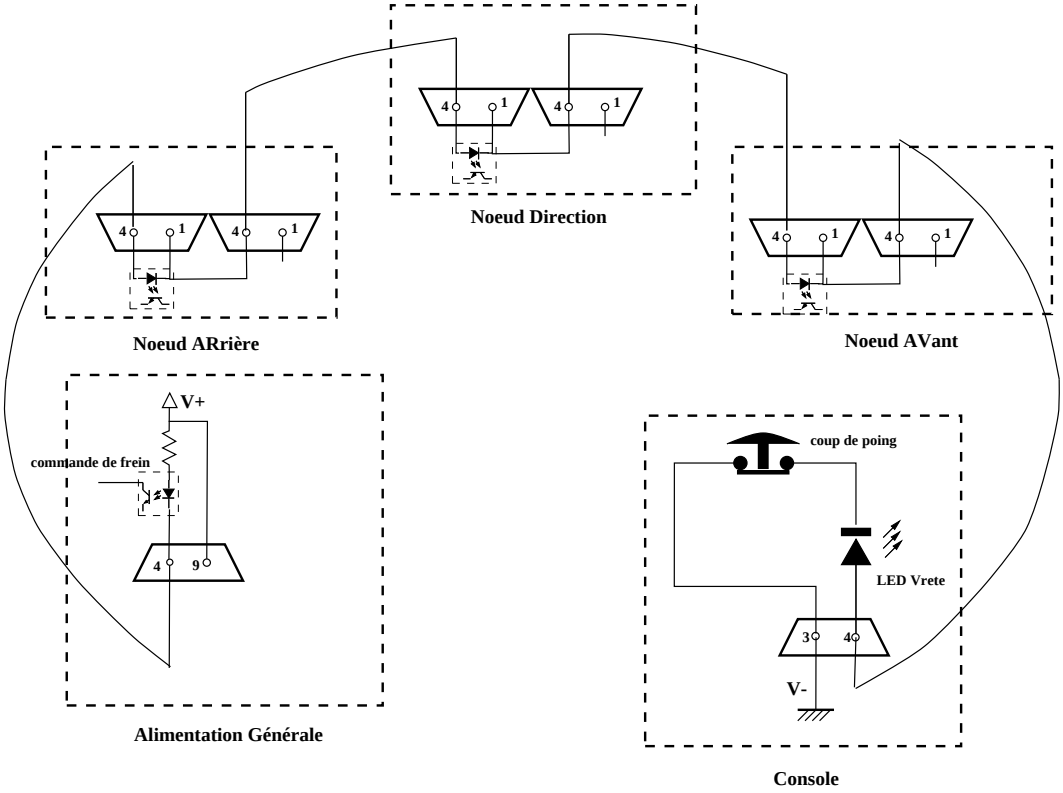


FIG. 1.7: *La boucle d'arrêt d'urgence*

*ATTENTION: Pour le test d'un nœud seul hors du véhicule, sur le banc de test par exemple, il est nécessaire de simuler le fait que le bouton d'arrêt d'urgence ne soit pas enfoncé et donc alimenter l'optocoupleur à travers le connecteur DB9 (CAN) qui sert aussi à l'alimentation du nœud. Il faut donc réaliser 2 ponts d'une part entre les broches 5 (***V-***) et 1 (***cathode***) et d'autre part entre les broches 9 (***V+***) et 4 (***anode***). Sans cette précaution, il ne peut y avoir de commande des transistors de puissance (voir la programmation du circuit EPLD Lattice).*

Chapitre 2

Architecture matérielle de contrôle/commande

2.1 Description fonctionnelle des nœuds

Pour commander et contrôler les différents moteurs du CyCab, nous avons opté pour une architecture répartie autour d'un bus de terrain, le bus CAN (Controller Area Network) très répandu dans le monde de l'automobile. Nous avons développé des nœuds intelligents (Amplificateurs de Puissance Intelligents). Chaque API pilote, pour l'application qui nous intéresse, 2 moteurs à courant continu (par exemple les moteurs de traction des roues avant) mais ils se veulent plus généraux et peuvent commander d'autres types de moteurs (asynchrone ou brushless).

Le rôle des nœuds est d'asservir les moteurs en fonction des consignes de vitesse et de braquage qui transitent sur le bus CAN soit en provenance de l'Interface Homme Machine (la position du joystick), soit calculées par un programme de planification de trajectoires. Il doit donc non seulement être capable de fournir la puissance nécessaire au moteur, mais aussi exécuter les boucles d'asservissement de vitesse ou de position. Pour ce faire il doit prendre en compte un certain nombre d'informations en provenance des capteurs proprioceptifs : état, odométrie, fins de course, mesures de température, de courant, ...
Cette description succinte fait apparaître trois entités composant les nœuds (figure [2.1]) :

- un **module de puissance** avec les transistors MOS-FET de puissance et leur commande de GATE pour piloter 2 moteurs,

- un **module d'interface et de communication** dont le rôle est essentiellement de mettre en forme les signaux Tout ou Rien ou de convertir les signaux analogiques en provenance des capteurs, de l'IHM ou allant vers les indicateurs d'état pour qu'ils

soient exploitables par le microcontrôleur. Ce module gère aussi les communications sur le bus CAN.

- un **module de calcul** qui, à partir des consignes et des données proprioceptives, calcule les courants à envoyer aux moteurs.

Cette décomposition se retrouve dans le matériel et le nœud se présente sous la forme de trois cartes images de ces modules fonctionnels et logées dans un même boîtier. Cette conception modulaire donne de la souplesse et permet d'adapter ce type de nœud à d'autres types de moteurs ou à d'autres gammes de puissance en ne changeant par exemple que l'étage de puissance.

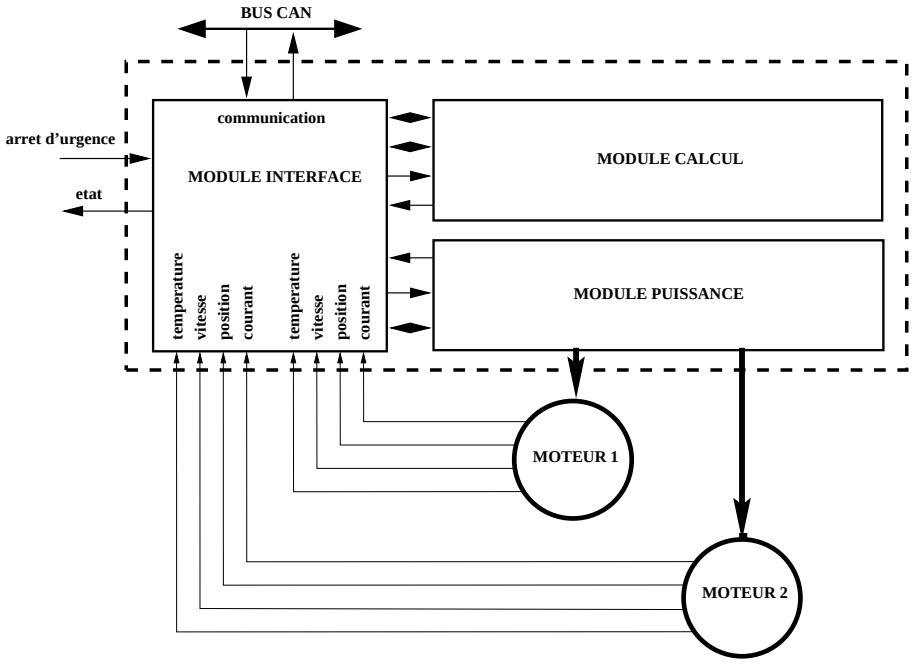


FIG. 2.1: *Schéma fonctionnel d'un nœud*

2.2 L'alimentation des nœuds

Toutes les tensions d'alimentation nécessaires au bon fonctionnement des différentes cartes constituant le nœud sont faites à partir de l'alimentation de 15 Volts générée par le boîtier d'Alimentation Générale et arrivant par le connecteur DB9 CAN entre les broches 9 (ALIM) et 3-4-6 (Gnd). Cette tension est reprise sur la nappe 64 fils de la carte Interface

où l'on trouve un fusible de protection. Cette tension sert à fabriquer la tension principale Vcc à l'aide d'un circuit régulateur de puissance **L4960** fournissant une tension variable de 5.1 Volts à 40 Volts pour un courant de 2.5 Ampère (5.1 dans notre cas par rebouclage de 2 sur la sortie 7). La boucle de régulation est constituée d'un oscillateur en dents de scie, d'un amplificateur d'erreur, d'un comparateur et d'un étage de sortie. Un signal d'erreur est obtenu en comparant la tension de sortie avec une tension de référence interne précise de 5.1 Volts. Ce signal d'erreur est ensuite comparé avec le signal en dents de scie pour générer, à une fréquence déterminée par le réseau RC parallèle connecté sur la broche 5, les impulsions modulées en largeur qui pilotent l'étage de sortie. Les paramètres de la boucle (gain et stabilité) sont ajustés par un réseau RC externe connecté à la broche 3. Le circuit est protégé contre les surintensités au moment des commutations et contre les courts-circuits en sortie : quand le courant de charge dépasse un valeur prédéfinie, l'étage de sortie est désactivé jusqu'à ce que la tension aux bornes de la capacité connectée en 6 atteigne 0.4 Volt. Cette tension est utilisée par toutes les cartes et est disponible sur les broches 3, 4 et 31, 63 des connecteurs BCC, sur la broche 3 (Vcc1) du connecteur vers la carte puissance, et sur les fils 21, 11 et 1 de la nappe 64 fils.
À partir des 15 Volts, une tension ALIM-REG est obtenue en sortie d'un régulateur de tension ajustable **LM317**. Un potentiomètre CMS monté à proximité du circuit permet de l'ajuster finement. Ceci est important pour la carte de puissance.
ATTENTION : *penser à vérifier la valeur de cette tension qui ne doit pas dépasser 12 Volts.*
ALIM-REG est disponible sur les broches 4 et 24 du connecteur de puissance (en fait Vcc2 est soit Vcc soit ALIM-REG, le choix se faisant par un jumper situé à proximité du connecteur), et sur le fil 31 de la nappe.
Sur la carte puissance, cette tension sert principalement à alimenter un ocillateur délivrant des signaux carrés à la fréquence de 250 kHertz qui permettent de fabriquer les alimentations flottantes continues de 9 Volts totalement découplées pour les circuits de commande des Gates des MOS-FET de puissance (4 par carte fille) (figure [2.2]). A partir de ces 9 Volts, on refabrique du 5 Volts pour alimenter les primaires des amplis d'isolement par lesquels transitent les mesures de courant dans les H-Ponts.

2.3 Le module de calcul

Inspirée du module M68332BCC de Motorola, il est bâti autour du microcontrôleur 32 bits MC68332. Il en a les même fonctionnalités mais il est pourvu d'une mémoire Flash ce qui permet de modifier le programme résidant sans être obligé d'ouvrir le nœud pour enlever l'EPROM pour l'effacer avant de la reprogrammer.
ATTENTION : *le design de ces cartes est la propriété de Leszek Lisowski. Cependant il existe l'équivalent chez Newton Research Labs, 14813 NE 13th St. Bellevue, WA 98007*

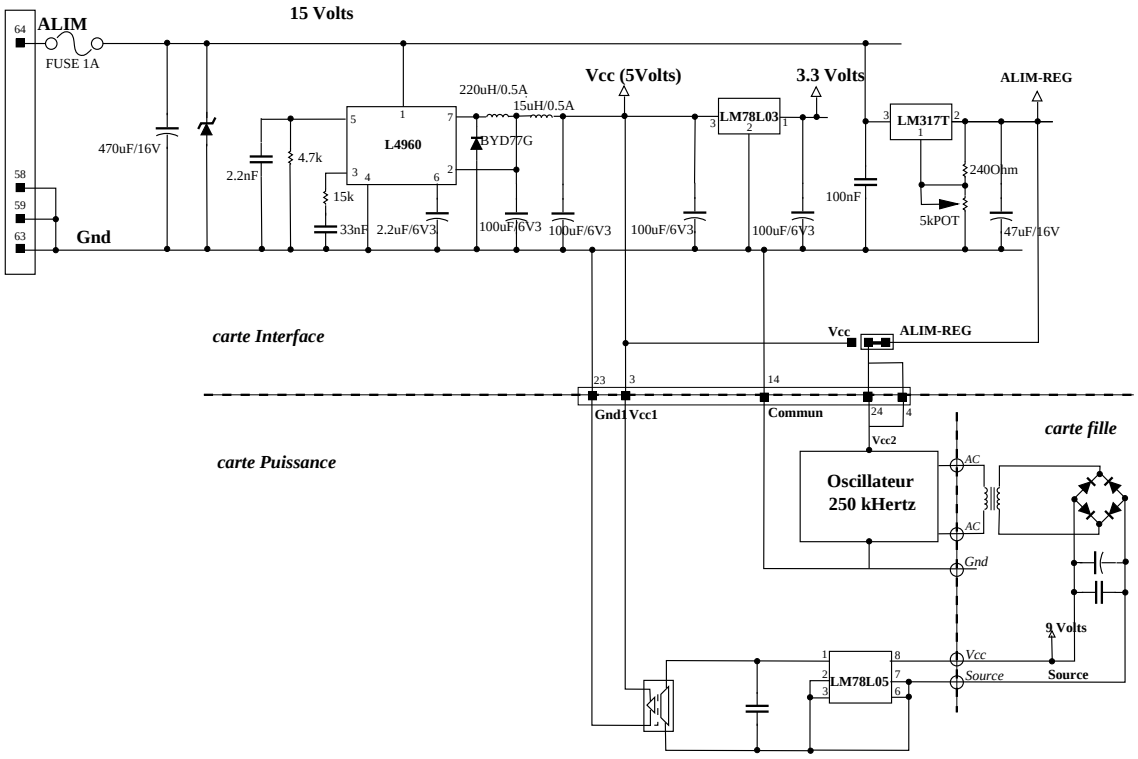


FIG. 2.2: *Les alimentations des nœuds*

2.3.1 Le microcontrôleur MC68332

Le MC68332 est un microcontrôleur CMOS intégré de 32 bits combinant des possibilités avancées de manipulation de donnnées avec des sous systèmes périphériques puissants. Le cœur opératoire du microcontrôleur (block CPU32) s'apparente au MC68020 et est compatible avec les logiciels de la famille MC68000. Le schéma block suivant montre les principaux composants du MC68332. Le code pour le CPU32 est entièrement compatible avec

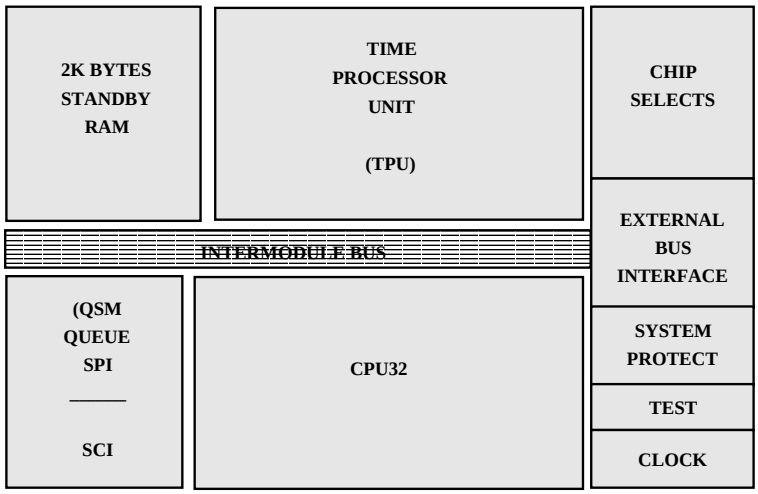


FIG. 2.3: *Schéma block du MC68332*

le code 68010 et la plupart des extensions 68020 (support de mémoire virtuelle, loop mode opération, instructions "pipe-linées", opérations mathématiques sur 32 bits) De nouvelles instructions (table lookup, interpolation, ...) supportent les requêtes spécifiques aux applications de contrôle.
Le MC68332 possède des sous systèmes périphériques autonomes qui augmentent les performances du CPU pour les applications de contrôle/commande de processus détaillés dans les sous-sections suivantes (figure [2.4]).

Time Processor Unit (TPU)

Le TPU contrôle les activités dépendant du temps en réduisant de façon importante l'intervention du CPU avec une unité d'exécution dédiée, un séquenceur à trois niveaux de priorité, une RAM de stockage, et une ROM microcodée. Le TPU contrôle 16 canaux indépendants, chacun d'eux étant associé à une broche E/S. Parmi les algorithmes microcodés disponibles, on trouve :

– Entrées/Sorties discrètes,

- Mesure de période avec détection de transition,
- Génération d'impulsions synchronisées,
- Contrôle de moteurs pas à pas,
- Entrées comptage (capture),
- Correspondance de sortie (output match)
- Sorties périodiques à largeur d'impulsion modulée.

Queued Serial Module (QSM)

Le QSM contient 2 ports série. Le QSPI permet des extensions périphériques aisées ou des communications interprocesseur via un bus full-duplex synchrone (data in, data out et horloge série). Quatre broches programmables de sélection de périphériques permettent d'adresser jusqu'à 16 périphériques. Une queue contenue dans une petite RAM permet au QSPI de gérer 16 transferts série de 8 ou 16 bits chacun ou de transmettre un flot de 256 bits sans l'intervention du CPU. Un mode spécial (wraparound) permet à l'utilisateur de faire de façon continue l'échantillonnage d'un périphérique série en mettant automatiquement à jour la RAM QSPI pour une interface efficace avec des convertisseurs Analogiques Digitaux série par exemple.

System Integration Module (SIM)

Le SIM inclut une interface externe et des fonctions qui réduisent la glu logique autour du boîtier :

- l'interface bus externe (EBI) fournit 24 lignes d'adresses et un bus de données de 16 bits. L'arbitrage externe du bus est facilité par 3 lignes de handshaking.
- Douze chip selects programmables autorisent un accès rapide (en 2 cycles) aux mémoires ou aux périphériques externes. La taille des blocks concernés par ces Chip-Selects est programmable entre 2K et 1M octets. Les accès peuvent être présélectionnés pour des transferts de 8 ou 16 bits. Jusqu'à 13 "wait states" peuvent être insérés durant les transferts. Tous les signaux d'interface au bus sont automatiquement gérés par la logique des chip-selects.
- Les moniteurs de bus, de HALT, d'interruption fatale, les timers de chien de garde et d'interruption périodique sont intégrés au microcontrôleur pour réduire la taille et les coûts induits par des composants externes.
- L'horloge système est générée par un PLL intégré à la puce qui permet une cadence de 16,78 MHz avec un quartz de 32,768 MHz. La vitesse du système peut être changée dynamiquement pour aller vers plus de performances ou une moindre consommation.

Standby Ram Module

Ce module contient 2K bytes de RAM statique rapide alimentée normalement par la broche Vdd. Pendant le "power-down", le contenu de la RAM est maintenu en alimentant la broche Vstby. Cette RAM est très utile pour le CPU comme pile système ou pour le stockage de variable (2 cycles de temps d'accès). Elle est aussi utilisable par le TPU comme RAM d'émulation pour de nouveaux algorithmes de timing. La figure suivante montre le MCU68332 et ses broches d'Entrée/Sortie:

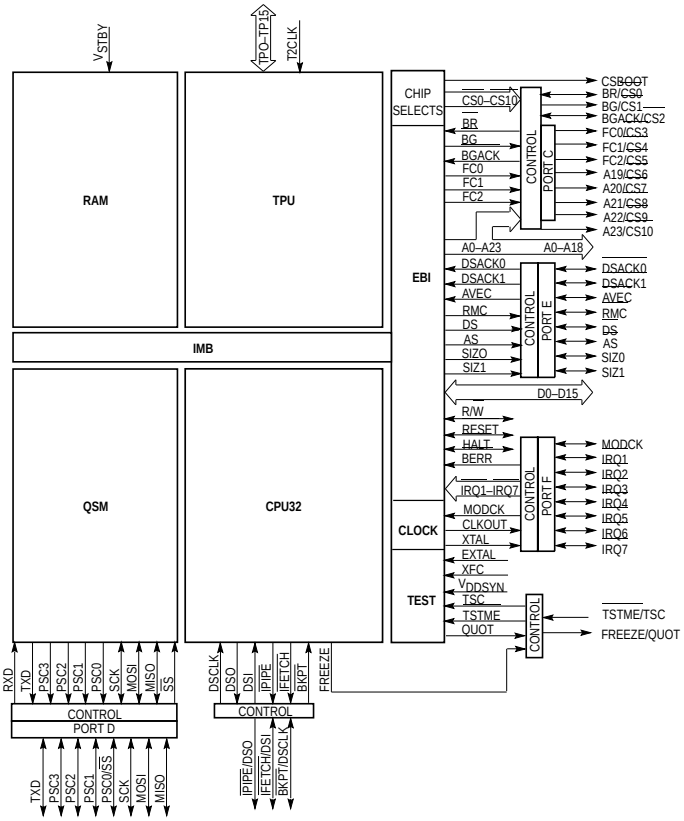


FIG. 2.4: *Le microcontrôleur MC68332*

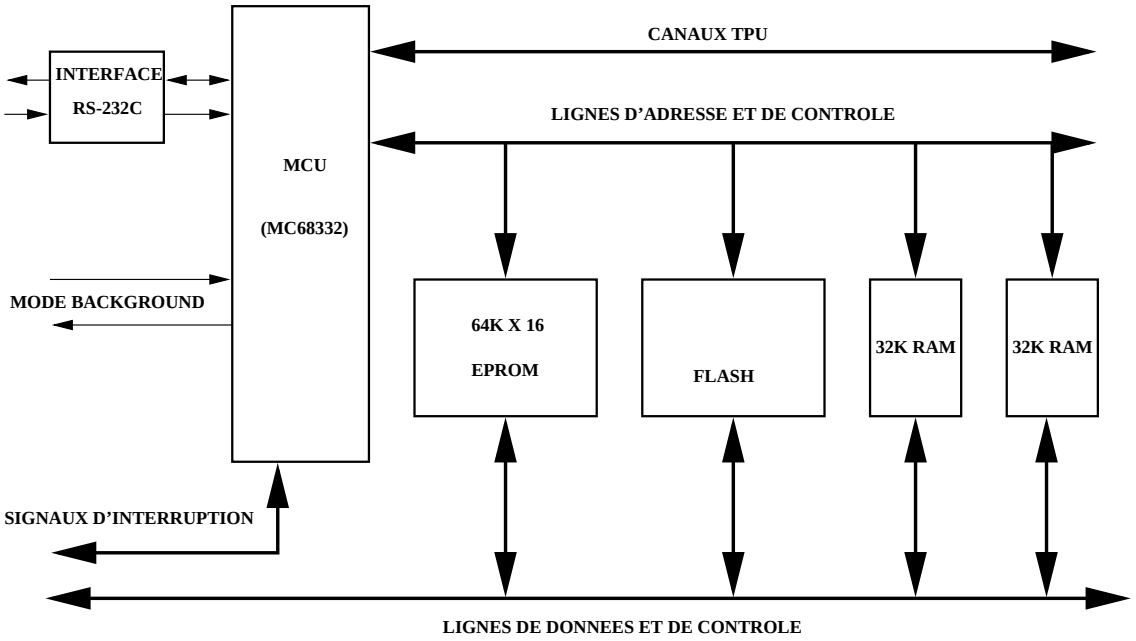


FIG. 2.5: *Schéma synoptique de la carte BCC*

2.3.2 Le module M68332BCC LL-C

Le module BCC est un calculateur monocarte de 5,7cm par 8,9cm qui comprend, outre le microcontrôleur MC68332 avec un cristal de 37.768 kHz :

- Une EPROM de 64k x 16 bits qui contient un logiciel d'évaluation et un outil de déverminage (332Bug) qui peut être utilisé pour développer des systèmes bâtis autour du MCU. Il suffit pour cela de connecter un terminal sur le port RS232. Le 332Bug permet d'afficher et de modifier le contenu de la mémoire et des registres internes au MCU, de contrôler l'exécution (démarrage de l'application, gestion de points d'arrêt et mode trace), d'appeler des fonctions système prédéfinies, de tester le MCU, d'assembler ou de désassembler le programme ligne par ligne.

- Une mémoire RAM de 32 X 16 bits.

- Un port d'Entrée/Sortie compatible RS232 avec convertisseur DC-DC +/- 10Volts

- Une mémoire Boot Block Flash de 2 ou 4 Mbits (128K X 16 ou 256K X 16. Cette mémoire est organisée en 5 (7) blocks effaçables séparemment incluant un boot block de 16K octets vérouillable par "hard", 2 blocks de paramètres de 8K octets chacun, 2 blocks principaux (1 de 96 K et 1 (3) de 128 K). Chacun des blocks peut être effacé séparemment (typiquement en 2,4 secondes pour un block principal ou en 1 seconde pour les blocks de boot ou de paramètres)et ce 100 000 fois. Le boot block est logé en adresses basses et il n'est accessible en effaçage ou en écriture que si sa broche **RP** est alimentée en 12 Volts. L'interface utilisateur de commande (CUI) sert d'interface entre le microcontrôleur et les opérations internes à la Flash. Un automate de programmation et d'effaçage gère de manière interne les algorithmes et le timing nécessaires (incluant la vérification) pour programmer ou effacer la mémoire. L'intérêt de l'utilisation d'une mémoire Flash est de pouvoir faire évoluer les programmes applicatifs en cours de développement sans avoir à retirer la mémoire EPROM pour l'effacer et la reprogrammer, tout en protégeant le minimum pour pouvoir faire "booter" le microcontrôleur. Le fait d'avoir sur la carte à la fois l'EPROM et la mémoire FLASH induit des modifications sur les jumpers de paramétrisation : il faut d'une part pouvoir alimenter la broche **RP** soit en 5 Volts soit en 12 Volts pour modifier le contenu du boot block et d'autre part rediriger le CSBOOT du microcontrôleur soit sur l'EPROM soit sur la FLASH. Il est possible de fonctionner soit avec EPROM + FLASH et boot sur l'EPROM, soit avec la FLASH seule et boot sur la FLASH. Il y a donc 2 mappings mémoire possibles pour la carte BCC.

- Deux connecteurs 64 points reprenent les broches du MC68332 pour l'extension des possibilités de la monocarte par enfichage d'une carte dédiée à l'application envisagée autour du microcontrôleur.

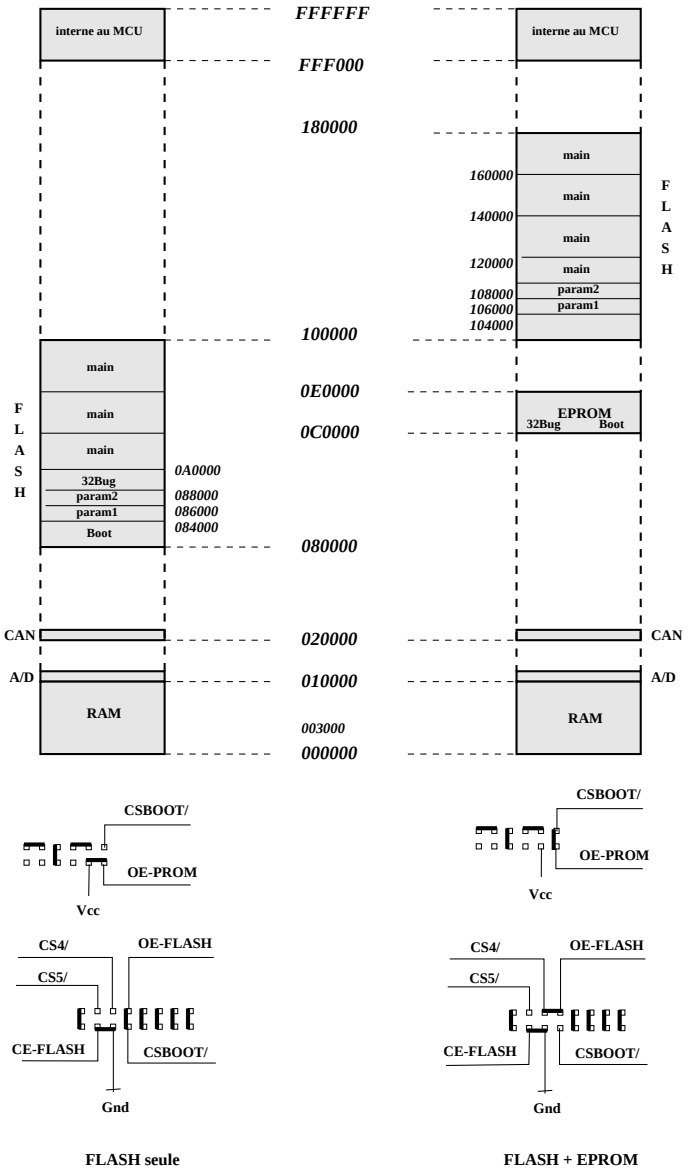


FIG. 2.6: *Les 2 mappings mémoire du BCC*

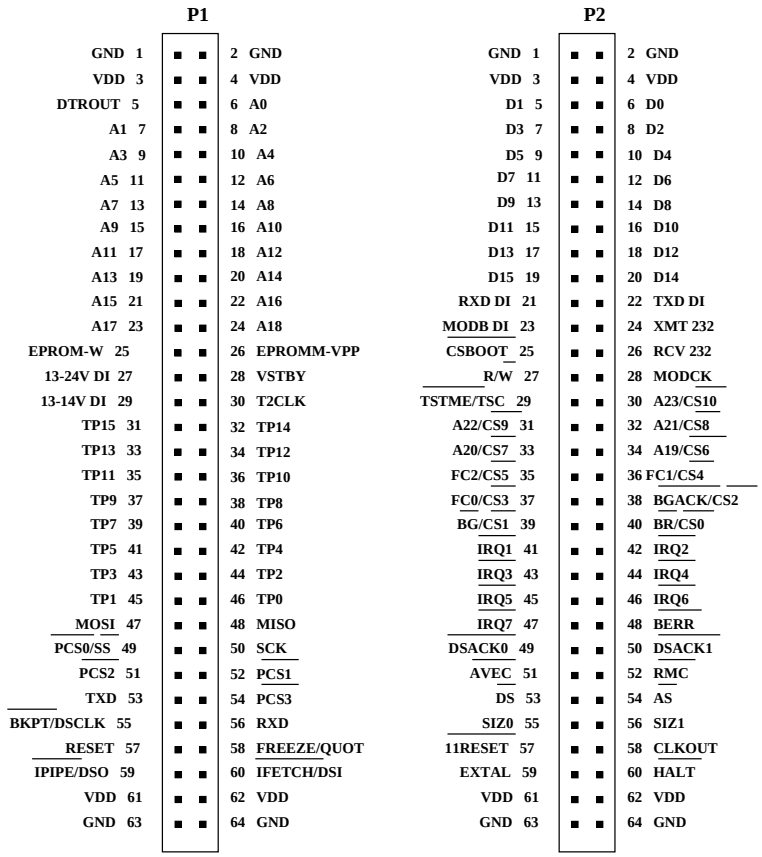


FIG. 2.7: *Connecteurs d'extension de la carte BCC*

2.4 La carte d'interface

Le rôle de cette carte est d'interfacer la carte processeur avec le monde extérieur : liaison série RS232, communication par bus CAN (Controller Area Network), conversion d'entrées analogiques, mise en forme et isolation galvanique des signaux Tout ou Rien aussi bien en entrée qu'en sortie. Le module de calcul vient s'enficher sur la carte Interface et Communication grâce aux connecteurs d'extension de la carte BCC. Elle est relié par une nappe de 64 fils aux connecteurs montés sur le capot du nœud (1 DB25 pour les entrées/sorties, 2 DB9 pour le bus CAN et un DB9 pour la liaison RS232) et à la partie puissance (signaux de commande et de contrôle des drivers de MOS-FET.

2.4.1 Interface bus CAN

Le bus CAN [6] est un bus "série, asynchrone, à 2 fils, symétrique". Il se présente sous la forme normalisée des différentes couches ISO/OSI. Il en est à sa version 2.0 qui comprend 2 parties A et B. La partie A décrit la trame CAN la plus courante qui permet d'adresser des identificateurs sur 11 bits (contre 29 bits pour la version B ou étendue). Le bus CAN possède les principales propriétés suivantes :

- hiérarchisation des messages,
- garantie des temps de latence,
- souplesse de configuration,
- réception de multiples sources avec synchronisation temporelle,
- système multimaîtres, détection et signalisation d'erreurs,
- retransmission automatique des messages altérés dès que le bus est libéré,
- distinction entre des erreurs temporaires et des non-fonctionnalités permanentes au niveau d'un nœud,
- déconnexion automatique des nœuds défectueux.

Le protocole CAN a été subdivisé en différentes couches : la couche Objet, la couche Transfert et la couche Physique.
La couche objet a pour mission principale de trouver quel message doit être transmis, de décider lequel des messages reçus via la couche transfert est en cours d'utilisation, de produire une interface à la couche applicative en relation avec le "hard" relatif au système.
La couche Transfert a pour mission principale de s'occuper du transfert du protocole. c'est à dire de gérer et contrôler la mise en forme de la trame, de réaliser l'arbitrage des conflits de bus, de vérifier l'absence ou la présence d'erreurs et de signaler les différents types d'erreurs s'il y en a ainsi que les fautes de confinement. C'est à l'intérieur de cette couche qu'il est décidé si le bus est libre pour démarrer une nouvelle transmission ou bien si une

réception d'un message incident est juste en train de démarrer.
La couche Physique a pour rôle d'assurer le transfert physique des bits entre les nœuds en accord avec toutes les propriétés électriques (ou électroniques) du système.
L'information véhiculée sur le bus est envoyée dans un format défini de longueur limitée. Lorsque le bus est libre n'importe lequel des participants peut démarrer une transmission. Dans un réseau CAN, un nœud n'a pas à se préoccuper des informations concernant la configuration du système (on peut rajouter ou supprimer des nœuds sans modifications "hard"ou soft"). Le contenu d'un message est repéré par un IDENTIFIER qui n'indique en rien la destination du message mais décrit le contenu des données. Ceci implique que chaque nœud doit être en mesure de décider si le message véhiculé par le bus le concerne ou non. Cette fonction est réalisée par un dispositif de MESSAGE FILTERING. Tous les nœuds peuvent donc recevoir et réagir simultanément à un même message. L'IDENTIFIER défini un message de priorité statique pendant l'accès au bus. Un nœud en envoyant une REMOTE FRAME signale à un autre nœud qu'il désire recevoir des données de sa part sous forme d'une DATA FRAME correspondante. Les 2 frames sont repérées par le même IDENTIFIER. L'unité ayant le message de plus haute priorité gagne l'accès au bus (multi-maître). Les conflits de bus (plusieurs unités démarrent en même temps) sont résolus par un arbitrage bit à bit du contenu de l'IDENTIFIER. Dans la perte de l'arbitrage l'unité considérée doit ne plus envoyer aucun bit. Afin d'assurer la qualité et la sécurité de la transmission, de nombreux dispositifs sont implantés : monitoring du bus (l'émetteur vérifie que le niveau électrique qu'il souhaitait imposer sur le bus est réellement présent), présence d'un CRC, "message frame Check", introduction de "Stuff bit". Une trame CAN est composé

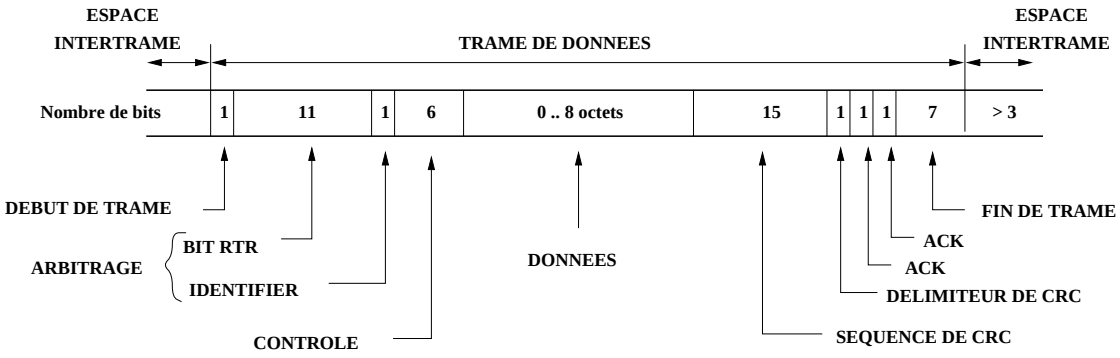


FIG. 2.8: *Protocole CAN : la trame de données*

de 7 champs principaux : (figure [2.8])

- le début de trame constitué par un seul bit "dominant" signalant à tous le début d'un échange et permettant la synchronisation sur le flanc avant de la transmission du bit,
- le champ d'arbitrage constitué par les bits de l'IDENTIFIER ainsi que du bit immédiatement suivant (RTR pour Remote Transmission Request),

- le champ de commande de 6 bits dont 4 indiquent le nombre d'octets contenus dans le champ de données suivis de 2 bits dominants,
- le champ de données de 0 à 8 octets transmis avec le MSB en tête,
- le champ de CRC,
- le champ d'acquittement de 2 bits : ACK SLOT (chaque fois qu'un récepteur a bien reçu un message, il superpose un bit "dominant" au bit "récessif" transmis à ce moment là) et un ACK DELIMITER,
- la fin de trame : la trame se termine par un drapeau formé d'une séquence de 7 bits "récessifs" et d'une huitième zone dite d'espace inter-frame.

Pour gérer ce protocole, la carte Interface et Communication utilise un circuit apte à créer et à déchiffrer le protocole CAN (couche Transfert et Physique du modèle ISO/OSI) le PCA C200 (figure [2.9]). Ce circuit fait le lien entre le microcontrôler (bus d'adresse, de données

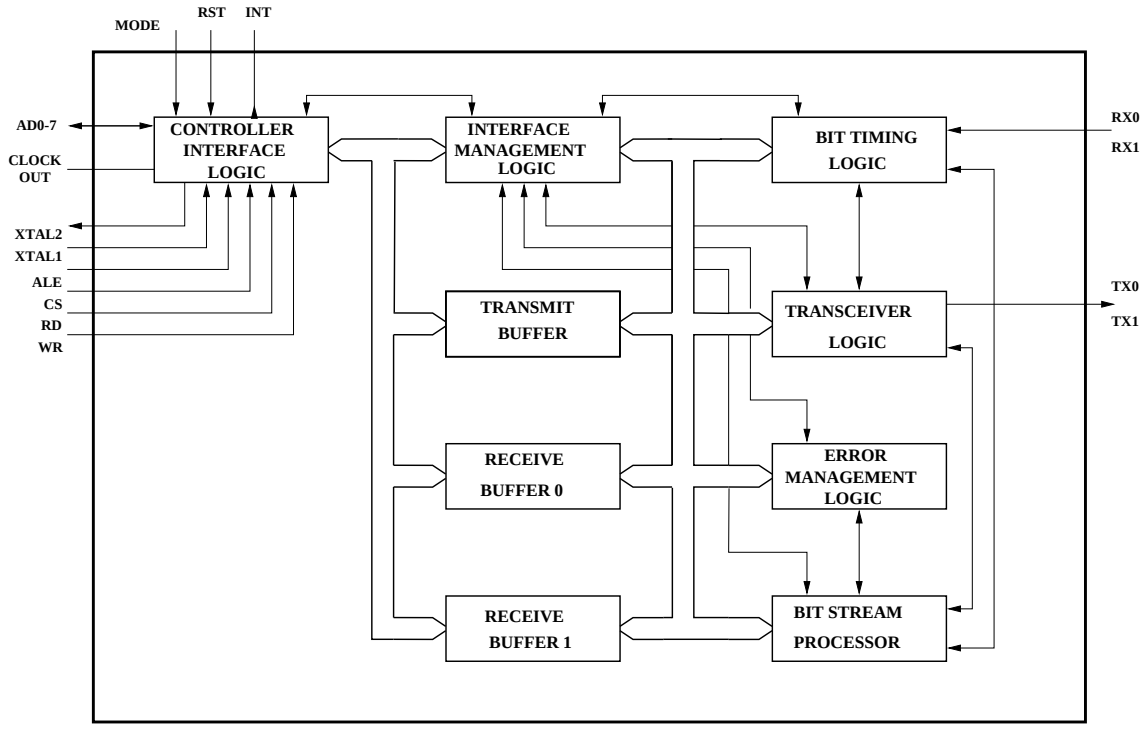


FIG. 2.9: *Schéma synoptique du PCA C200*

et de contrôle) et la ligne bifilaire différentielle de transmission inter-nœuds via un circuit

d'interface de ligne CAN (82C250T) qui réalise la transposition de niveaux asymétriques (0 à 5 Volts) en niveaux différentiels sur une ligne d'impédance caractéristique de 120 Ω. Le schéma synoptique du C200 fait apparaître plusieurs parties bien distinctes :

- Une interface logique assurant la liaison avec le microcontrôleur via un bus d'adresses/données et les signaux de service classiques (ALE, CS, RD, WR, ...). Dans cette partie sont centralisées les différents registres d'état des échanges entre microcontrôleur et bus CAN.
- Deux étages tampon, de 10 octets chacun, peuvent être utilisés en alternance lors de la phase de réception permettant à 2 informations successives de pouvoir arriver sans perturber le traitement déjà en cours.
- Un étage tampon de 10 octets destiné au message devant être prochainement émis.
- Un gestionnaire de protocole et la logique associée au traitement des erreurs,
- De la glu logique nécessaire au bon fonctionnement de l'ensemble.

Le bus adresses/données est connecté aux 8 bits poids faible du bus de données du MC68332. L'initialisation "hard"du circuit se fait à la mise sous tension grâce à un réseau RC. Les signaux de services sont fournis au C200 par un EPLD 16V8 qui les calculent à partir des signaux R/W, AS et CS0 issus du microcontrôleur. La sortie INT est câblée sur l'entrée IRQ6 du μP.

2.4.2 Conversion analogique/digitale

La carte Interface est prévue pour convertir des signaux analogiques en signaux numériques grâce à un ADC 10 bits à 8 canaux d'entrée (AD7778) (figure [2.10]). Le résultat de la conversion est fourni sur le bus de données du microcontrôleur. Le chargement du registre de contrôle, la lecture des registres ADC (résultat des conversions) et le début de la conversion se font sous le contrôle du μP à travers 10 bits de son bus de données et ses lignes de contrôle (à travers le même EPLD qui gère les lignes de contrôle de l'interface CAN). Le choix du canal à échantillonner se fait en programmant le registre de contrôle qui est en lecture seule (bits CR0 à CR2). Il est aussi possible d'échantillonner 2 canaux simultanément (si le bit CR6 est positionné à 1) qui sont adressés par CR0 à CR2 pour le premier et CR3 à CR5 pour le second. Le bit CR7 positionné à 1 permet d'échantilloner la référence REFIN (milieu de la plage de tension) qui est sur l'entrée des multiplexeurs analogiques. Le bit CR8 à 1 met le circuit en mode Power-Down pour lequel toute la circuiterie linéaire est déconnectée. Le bit CR9 est lié au flag $\overline{BUSY}/\overline{INT}$. La conversion commence sur le front montant du signal $\overline{WR}$ et sa fin est donnée par le flag $\overline{BUSY}/\overline{INT}$: si CR9 est positionné à 0, $\overline{BUSY}/\overline{INT}$ passe à l'état bas et est maintenu à l'état bas pendant tout le temps de la conversion. Dans le cas contraire (CR9 = 1) $\overline{BUSY}/\overline{INT}$ passe à l'état bas quand la (ou les) conversion(s) est (sont) terminée(s). Il est donc possible soit de venir scruter l'état du flag pour savoir si une conversion est en cours (mode $\overline{BUSY}$ avec CR9 = 0) soit d'être

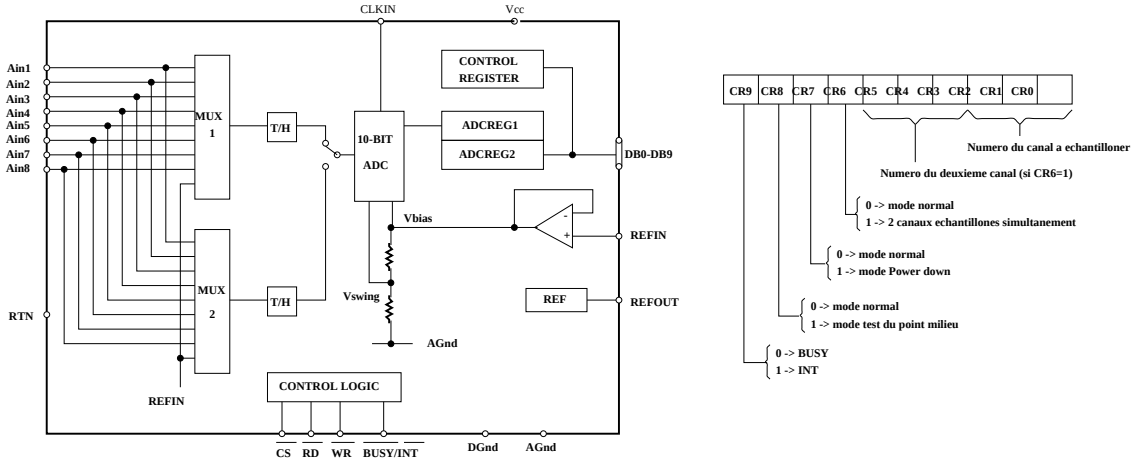


FIG. 2.10: *Schéma Block du Convertisseur Analogique Digital*

averti de la fin de la conversion par changement d'état du flag (mode $\overline{INT}$ avec CR9 = 1). C'est la première solution qui a été choisie : la broche $\overline{BUSY}/\overline{INT}$ est connectée à l'entrée DSACK1 du μP. *Il serait plus judicieux d'utiliser une des entrées d'interruption.*
La plage de variation des signaux d'entrée à échantilloner est de la forme :

$$V_{BIAS} \pm V_{SWING}$$

V_{BIAS} est la tension de référence pour toutes les entrées (pseudo masse). Elle est déterminée par la tension appliquée à la broche REFIN qui peut être soit une source externe soit une référence de 2 Volts fournie sur la broche REFOUT du circuit. L'amplitude du signal d'entrée est alors de : $V_{BIAS}/2$ (ou $REFIN/2$). Ainsi avec REFIN = 2 Volts, l'entrée pourra varier entre 1 et 3 Volts. Il est possible de changer cette plage de fonctionnement entre 0 et V_{REF} en forçant la broche RTN à une valeur supérieure à AGND. Le domaine de variation sera alors de :

$$V_{OFFSET} \text{ à } V_{OFFSET} + REFIN$$

avec :

$$0 \text{ volt} \leq V_{OFFSET} \leq 1 \text{ Volt.}$$

Pour obtenir cette plage de fonctionnement la broche RTN doit être fixée à

$$\text{REFIN - } 2V_{OFFSET}.$$

Par exemple, si elle est reliée à REFOUT, la plage d'entrée devient :

$$0 \text{ Volt à } 2 \text{ Volts}$$

ATTENTION *: La carte a été routée pour un fonctionnement du convertisseur analogique digital avec une plage de variation entre 0 et 2 Volts pour les entrées analogiques (RTN = REFIN = 2 Volts). Elle a ensuite été modifiée pour une plage de variation de 1 à 3 Volts (broche 36 (RTN) dessoudée de la carte, tordue et re-soudée sur la broche 37 (AGND)).*
Les canaux analogiques $A_{IN}1$ à $A_{IN}3$ proviennent de la nappe de connexion vers le monde extérieur à travers des amplificateurs opérationnels LM324 montés en suiveur de tension. Ils servent aux mesures de la température des moteurs et de la tension aux bornes des potentiomètres associés au vérin motorisé et au joystick. Les canaux $A_{IN}4$ à $A_{IN}7$ proviennent du connecteur vers la partie puissance à travers des LM324 montés en amplificateurs différentiels (gain $\sim 1/2$) autour d'une référence de 2 Volts (REFOUT de l'AD7778). Ils servent à la mesure des courants dans les demi-ponts de commande des MOS-FET. La figure [2.11] montre

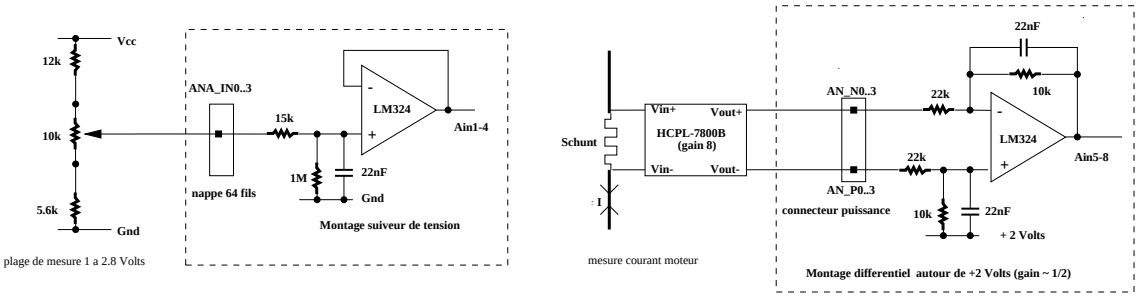


FIG. 2.11: *Les 2 montages pour les canaux analogiques*

un exemple d' utilisation pour chacun des 2 montages des LM324 : le premier pour la mesure de la position du vérin motorisé grâce au potentiomètre linéaire qui lui est associé, le second pour la mesure du courant dans les demi-ponts de commande des moteurs de traction.
ATTENTION *: Les noms des signaux et surtout les indices associés depuis les connecteurs jusqu'au convertisseur sont assez farfelus. Ils doivent correspondre à une logique que j'ai du mal à saisir. Ils sont rappelés dans les 2 tableaux suivants.*
Entrées analogiques externes (venant de la nappe 64 fils) :

nappe 64 fils	sortie LM324	entrée AD7778	numéro de canal
ANA-IN0	ANA1	$A_{IN}2$	1
ANA-IN1	ANA2	$A_{IN}3$	2
ANA-IN2	ANA3	$A_{IN}4$	3
ANA-IN3	ANA0	$A_{IN}1$	0

Entrées analogiques internes (venant du connecteur Puissance) :

connecteur Puissance	entrée LM324	sortie LM324	entrée AD7778	numéro de canal
+V1 -V1	AN_P0 AN_N0	ANA4	$A_{IN}5$	4
+V2 -V2	AN_P1 AN_N1	ANA5	$A_{IN}6$	5
+V3 -V3	AN_P2 AN_N2	ANA6	$A_{IN}7$	6
+V4 -V4	AN_P3 AN_N3	ANA7	$A_{IN}8$	7

2.4.3 EPLD de commande CAN et CAD

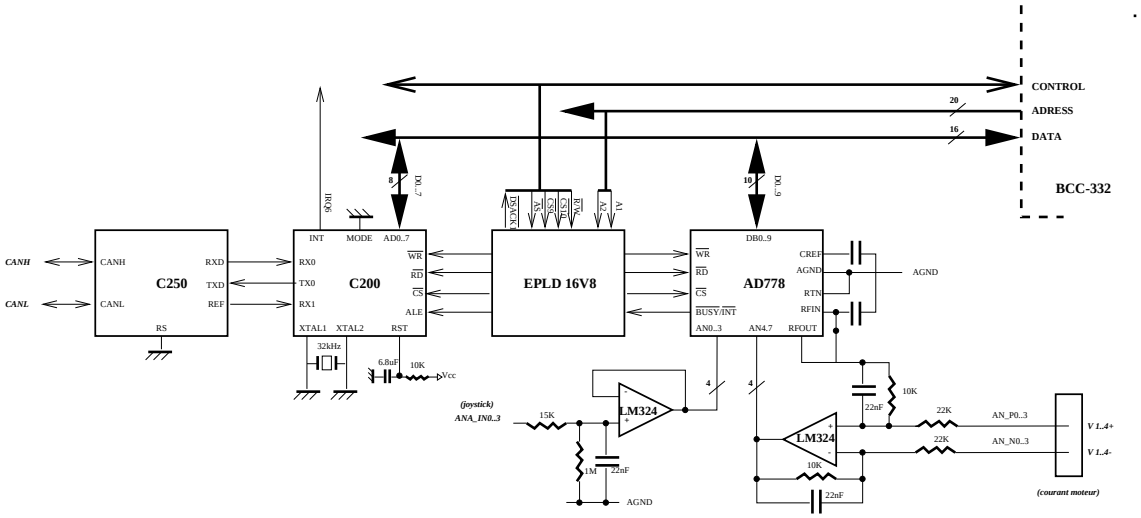


FIG. 2.12: *Interface CAN et conversion analogique/digitale*

Les signaux de service pour interfacer le microcontrôleur avec le circuit contrôleur CAN et le convertisseur analogique digital sont générés par un circuit programmable à partir des bits de service du MC68332 et des 2 bits de poids faible de son bus d'adresse selon les équations logiques suivantes :

– pour le circuit CAN :
$$\begin{cases} \overline{CS} & = & \overline{CS9} + \overline{AS} + \overline{A1} \\ \overline{WR} & = & \overline{\overline{AS} + \overline{CS9} + \overline{R/\overline{W}}} \\ \overline{RD} & = & \overline{\overline{AS} + \overline{CS9} + A1} \\ ALE & = & \overline{\overline{AS} + \overline{CS9} + A1} \end{cases}$$

– pour le circuit ADC: $\begin{cases} \overline{CS} & = & \overline{CS10} \\ \overline{WR} & = & \overline{AS} + \overline{CS10} + R/\overline{W} \\ \overline{RD} & = & \overline{AS} + \overline{CS10} + \overline{R/\overline{W}} \\ \overline{DSACK1} & = & \overline{BUSY}/\overline{INT} \end{cases}$

2.4.4 Liaison RS232

La carte BCC possède une connection RS232 (TxD et RxD) disponible sur les connecteurs d'extension. Ces lignes sont simplement acheminées sur la connecteur 64 points vers la nappe reliée au capot du boîtier.

2.4.5 Arrêt d'urgence et Etat

Chaque nœud doit être prévenu du fait que le bouton d'arrêt d'urgence est enfoncé. Il faut donc récupérer cette information simultanément sur chacun d'eux à l'aide d'un montage en série du bouton avec tous les nœuds. Sur chaque nœud l'information est opto-découplée et dirigée vers le circuit de logique programmable ou elle est combinée à d'autres informations pour générer un état reflétant une anomalie de fonctionnement sur un nœud. Cette information d'état est à la fois envoyée au μP et vers une LED sur la console du véhicule qui doit afficher l'état de l'ensemble des nœuds, donc le OU câblé des états de chacun des nœuds. Cette information ETAT-C générée par le pLSI1016 est amplifiée par un étage Darlington à collecteur ouvert par un MC1013 avant de parvenir au connecteur nappe.

Remarque : *Quand le bouton d'arrêt d'urgence est enfoncé tous les nœuds sont avertis et un défaut est généré automatiquement sur chacun d'eux. Par contre si un nœud est en défaut pour une raison autre qu'un arrêt d'urgence la LED "défaut" sera bien allumée, mais les autres noeuds continueront à fonctionner normalement. Ceci peut s'avérer dangereux. Cet état de dysfonctionnement pourrait être pris en compte sans modification "hard" en reprogrammant les pLSI1016 : sur la sortie ETAT-C on dispose en fait du OU câblé des états de tous les nœuds, cette information pourrait servir à construire l'état à passer au μP.*

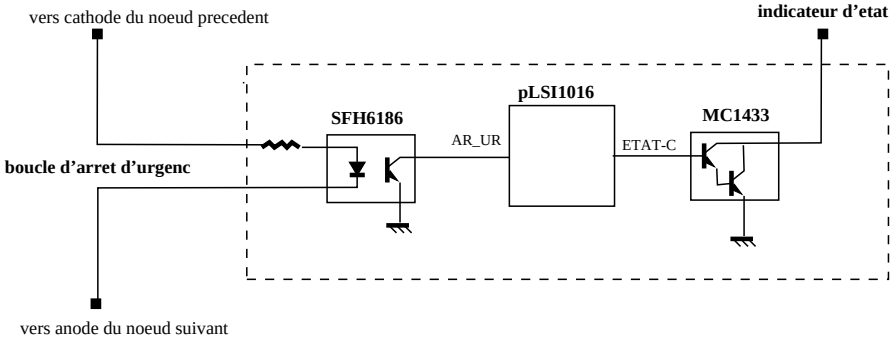


FIG. 2.13: *Arrêt d'Urgence et État*

2.4.6 Entrées/Sorties ToR

Les nœuds sont reliés au monde extérieur par une nappe de 64 fils reliée au boîtier. Sur cette nappe on dispose de 8 signaux Tout ou Rien :

- 2 entrées IN0 et IN1 (de l'extérieur vers le nœud),
- 2 sorties OUT0 et OUT1,
- 4 signaux qui peuvent être au choix configurés en entrée ou en sortie.

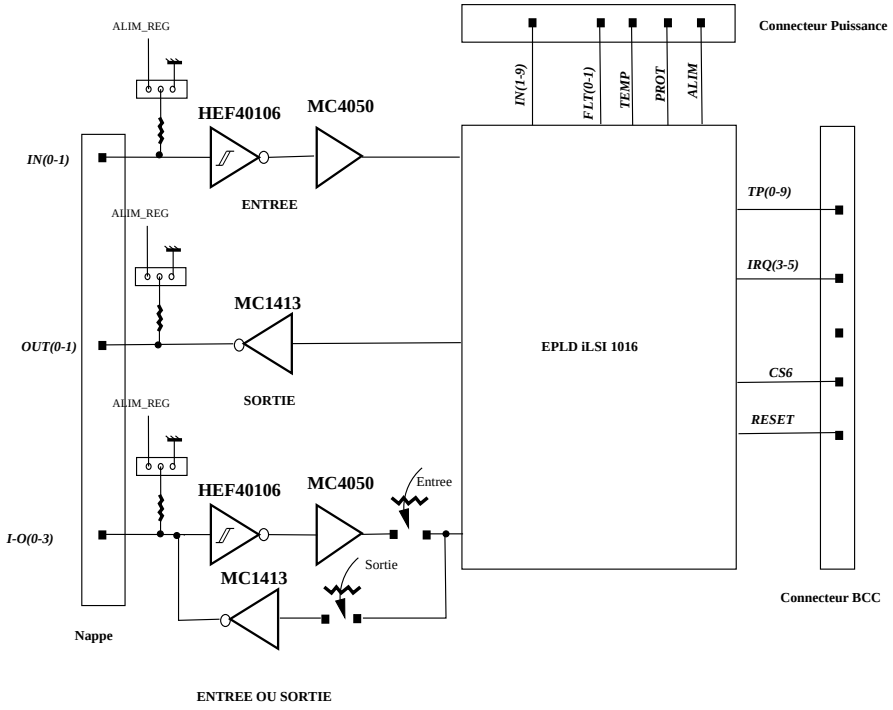


FIG. 2.14: *Interface entrées et sorties Tout ou Rien*

ATTENTION *: ce ne sont pas des signaux bidirectionnels, le sens de l'information véhiculée sur ces fils se fait par intervention "hard" au fer à souder (voir schéma) et par la programmation du circuit de logique programmable.* Ces signaux sont mis en forme et (ou) amplifiés sur la carte. Ils proviennent de (ou vont vers) un circuit de logique programmable (pLSI 1016) qui les combinent entre eux et avec d'autres signaux en provenance du connecteur puissance et du BBC). La programmation de cet EPLD dépend de l'application. En particulier son contenu n'est pas le même pour le nœud Direction et pour les nœuds Traction. Cependant le

câblage du circuit sur la carte reste identique. Les broches TP(0-9), $\overline{IRQ(3-5)}$, $\overline{RESET}$, $\overline{CS6}$ du connecteur d'extension ainsi que les signaux IN(0-8), FLT(0-1), TEMP, PROT, ALIM du connecteur Puissance sont reliés à l'EPLD dont le rôle est en fait de générer les commandes du véhicule en prenant en compte les états de l'application comme les fins de course ou les protections thermiques.
ATTENTION *:*

- *confusion possible entre les signaux IN0 et IN1 venant du connecteur nappe et IN0 et IN1 sortant par le connecteur Puissance !!!*

- *Les nœuds du véhicule semblent identiques quant au cablâge en dur des I-O: I-O0 et I-O1 sont câblés en entrée (R27 et R28 de 0 Ω montées et R23 et R24 absentes) et deviennent IN2 et IN3, I-O2 et I-O3 sont câblés en sortie (R25 et R26 de 0 Ω sont montées et R29 et R30 absentes) et deviennent OUT2 et OUT3.*

Les signaux en Entrée sont tout d'abord mis en forme par un circuit Trigger de Schmitt (HEF 40106) qui accepte en entrées des niveaux jusqu'à 12 Volts (résistance pull-up à V_{REG}) avant d'être convertis en niveaux TTL par un circuit MC4050. Les signaux en sortie sont amplifiés par un étage Darlington (MC1413B).

2.4.7 Interface codeurs incrémentaux

Pour les mesures odmétriques et de vitesse des roues (ou de braquage), les quatre moteurs de traction et le vérin motorisé de direction des CyCabs sont équipés de codeurs optiques incrémentaux HEDS 9140-E12 qui délivrent en sortie 2 ondes carrées en quadrature (500 cycles par tour) plus une sortie Index (top tour). Le constructeur préconise de monter des résistances "pull-up" de $2.7k\Omega$ ($\pm 10\%$) sur les sorties à moins de 1,20 m des codeurs. Dans ces conditions, chacune des sorties CH A, CH B, CH I peut piloter une charge TTL.
Sur la carte d'interface, les signaux venant des codeurs passent à travers un circuit récepteur de ligne différentielle (76LS32) avant d'aller sur des entrées TPU du BCC. Un réseau de résistances "pull-up" de $2.2k\Omega$ avec le point commun pouvant être relié soit à Vcc soit à une autre alimentation Vreg (jumper). En fonctionnement normal, le 76LS32 transforme un signal sous forme différentielle ($\pm$ Vin) en un niveau compatible TTL. Les signaux en provenance des codeurs sont déjà des signaux TTL. Ils sont connectés aux broches + du circuit récepteur et toutes les entrées sont forcées à 2.5 Volts grâce à un pont diviseur 2 fois $2.2k\Omega$ entre Vcc et Gnd logé dans le capot du connecteur DB25. Les emplacements pour des résistances d'adaptation de ligne entre les entrées + et les entrées - des récepteurs de ligne sont prévus sur la carte d'interface.
Remarque : Le circuit UA9637 qui est un double récepteur de ligne différentielle à 8 broches et qui met en forme les signaux CH I des codeurs n'est pas monté sur toutes les cartes : les "tops tour" ne sont pas utilisés par le logiciel.

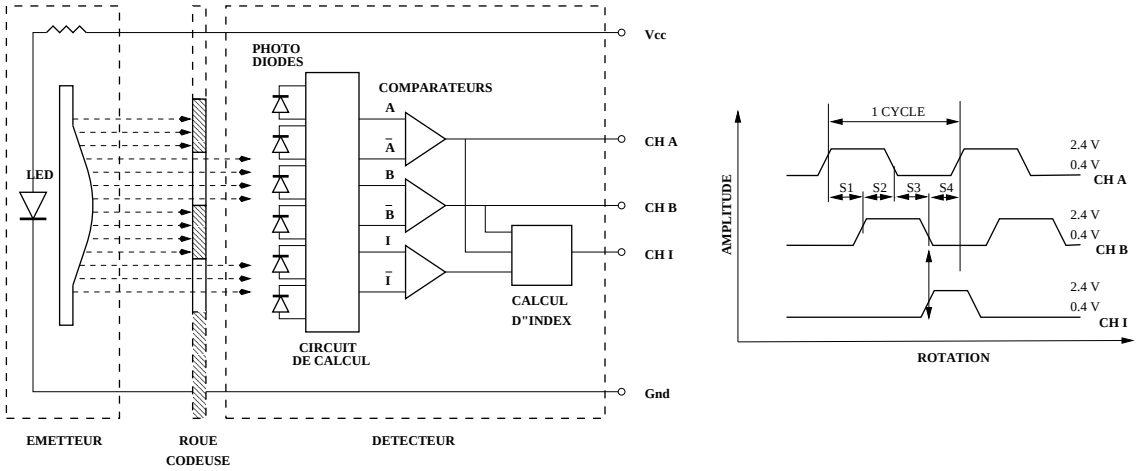


FIG. 2.15: *Schéma block du codeur et forme d'onde en sortie*

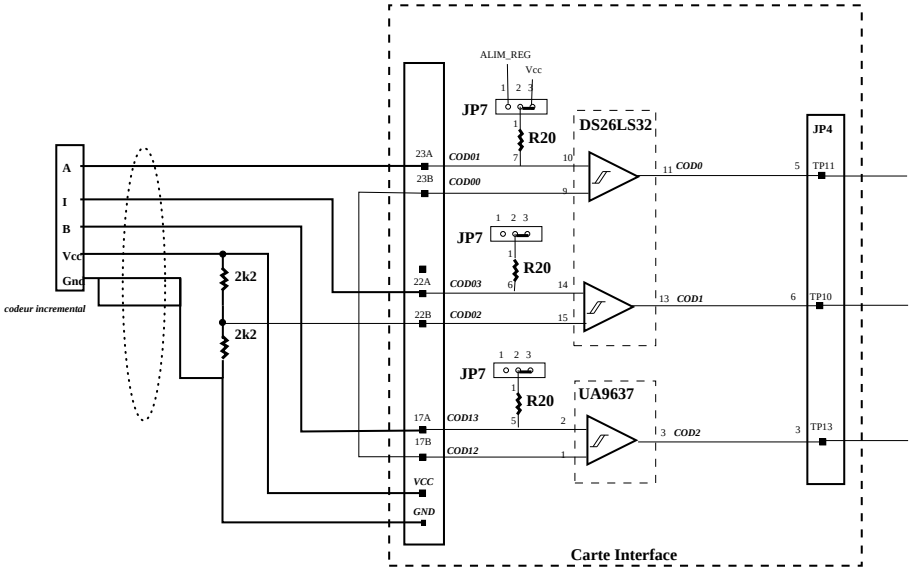


FIG. 2.16: *Mise en forme des signaux codeurs sur la carte Interface*

ATTENTION *: sur les schémas de câblage des cartes, les noms associés aux signaux codeurs changent en traversant les connecteurs :*

- *CH A du codeur devient COD01 (ou COD05) à l'entrée + du récepteur de ligne, l'entrée - étant COD00 (ou COD04). Le signal TTL exploitable correspondant s'appelle COD0 (ou COD2),*
- *CH B du codeur devient COD03 (ou COD11) à l'entrée + du récepteur de ligne, l'entrée - étant COD02 (ou COD10). Le signal TTL exploitable correspondant s'appelle COD1 (ou COD3),*
- *CH I du codeur devient COD13 (ou COD15) à l'entrée + du récepteur de ligne, l'entrée - étant COD12 (ou COD14). Le signal TTL exploitable correspondant s'appelle COD4 (ou COD5),*

2.4.8 Interface resolver et interface SPI

La carte d'Interface et de Communication se veut générique pour des applications de contrôle/commande avec différents types de moteurs. Il est donc prévu sur cette carte les emplacements pour une interface resolver et pour une interface utilisant les possibilités des liaisons SPI du MC68332. Pour l'application CyCab, ces possibilités ne sont pas utilisées et les circuits correspondants ne sont pas montés.

2.4.9 Circuit EPLD Lattice LSI1016

Toute la "glu logique" est réalisée à l'aide d'un circuit programmable qui combine les entrées Tout ou Rien et certains signaux en provenance du microcontrôleur pour réaliser l'interface avec la partie puissance.

Le circuit programmable utilisé est un circuit Lattice (ispLSI 1016) qui intègre :

- 96 registres,
- 32 broches E/S universelles,
- 4 broches dédiées Entrée,
- 3 broches dédiées entrée horloge,
- un système global de routage (Global Routing Pool) permettant une interconnectivité totale entre tous ses éléments.

Les blocks logiques de base (Generic Logic Block) labellisés A0. A1, ... , B7 sont au nombre de 16 pour le circuit choisi. Chaque GLB est constitué de 18 entrées, un taleau programmable AND/OR/XOR, et 4 sorties configurables (directes ou après mémorisation). Les entrées du GLB proviennent du GRP et des entrées dédiées. Toutes ses sorties sont connectées au GRP

et peuvent donc servir d'entrées à n'importe quel autre GLB. Le circuit contient 32 cellules d'E/S directement connectées aux broches. Chacune de ces cellules est séparemment configurable en Entrée directe, inversée ou avec une résistance pull-up, latchée ou mémorisée, en Sortie inversée ou non, totem pole ou avec contrôle trois états, ou enfin en E/S bidirectionnelle. Les sorties de 8 GLBs sont connectées à un ensemble de 16 cellules d'Entrée/Sortie via le Output Routing Pool (ORP). L'ensemble 8 GLBs, 16 I/O cells, 2 Entrées dédiées et 1 ORP constitue 1 Mégablock. Le ispLSI 1016 contient 2 de ces mégablocks. Le GRP combine entre eux tous les signaux provenant de tous les GLBs et de toutes les entrées venant des cellules I/O. Un système de distribution des horloges (Clock Distribution Netork) peut générer 5 signaux d'horloge vers les GLBs et les cellules I/O à partir de 3 broches dédiées du circuit (Y0, Y1 et Y2).
Les fonctions logiques programmées dans l'EPLD dépendent de l'application contrôlée par le nœud. Sur le CyCab, la programmation de l'EPLD du nœud de direction sera différente de celle des nœuds de traction.

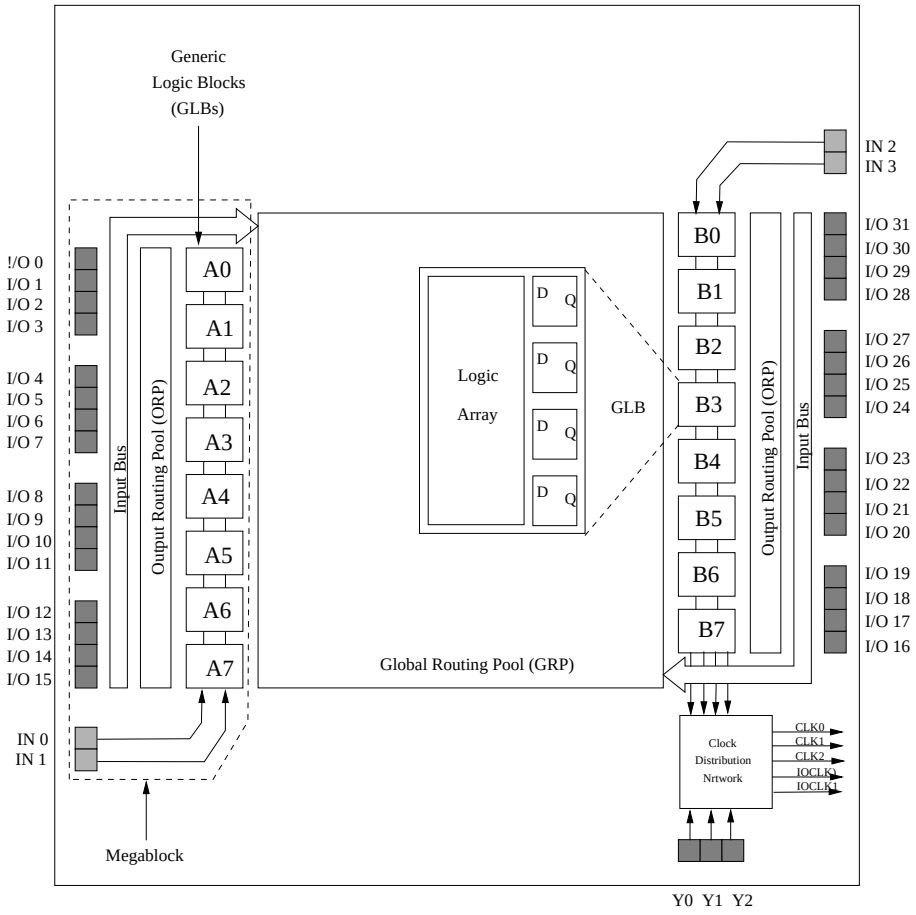


FIG. 2.17: *Diagramme fonctionnel de l'EPLD ispLSI 1016*

2.5 La carte de puissance

Elle communique avec la carte Interface par le connecteur Puissance.

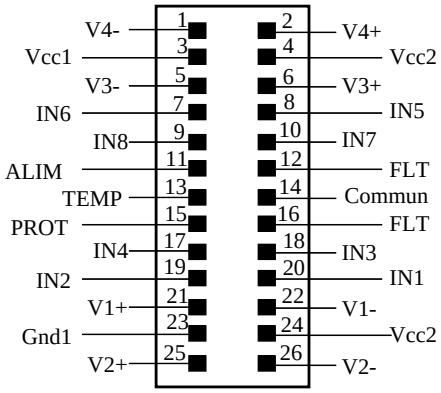


FIG. 2.18: *Connecteur entre la carte Interface et la carte Puissance*

En fait, ce n'est pas une carte unique. Elle est composée d'une carte mère qui supporte les deux H-ponts de transistors de puissance MOS-FET permettant de piloter deux moteurs à courant continu, et de 2 cartes filles qui commandent chacune 4 Gates, soit un H-Pont de MOS-FET.

2.5.1 Fonctionnement du H-Pont

Le moteur est connecté dans la partie horizontale du "pont en H" formé par quatres transistors MOS-FET canal N (figure [2.19]). Si TR7 et TR6 sont "ON" et TR5 et TR8 "OFF", le courant passe dans le moteur de gauche à droite et il tournera dans un sens. Pour inverser le sens de rotation du moteur il faudra inverser le sens du courant et donc rendre TR7 et TR6 "OFF" et TR5 et TR8 "ON". Il faut impérativement empêcher que les transistors situés sur un même côté du pont (TR5 et TR8 ou TR5 et TR6) ne conduisent simultanément ce qui entraînerait un court circuit sur les batteries (smoke test).

Pour faire varier la vitesse d'un moteur à courant continu, on applique une tension continue variable à ses bornes. Si on l'alimente en 48 Volts par exemple à travers TR7 et TR6, il va commencer à accélérer. Le moteur ne répond pas immédiatement et il va mettre un certain temps avant d'atteindre sa vitesse maximale. Si l'on coupe la puissance avant que cette vitesse ne soit atteinte, il va commencer à décélérer. Si on commute la puissance ON et OFF avec une fréquence suffisante, le moteur verra à ces bornes une tension d'alimentation qui sera grossièrement égale au produit de la tension délivrée par les batteries et du rapport cyclique du signal de commande et il tournera à une vitesse comprise entre 0 % et 100 % de la vitesse maximale. C'est exactement ce qui se passe avec un contrôleur PWM (Pulse

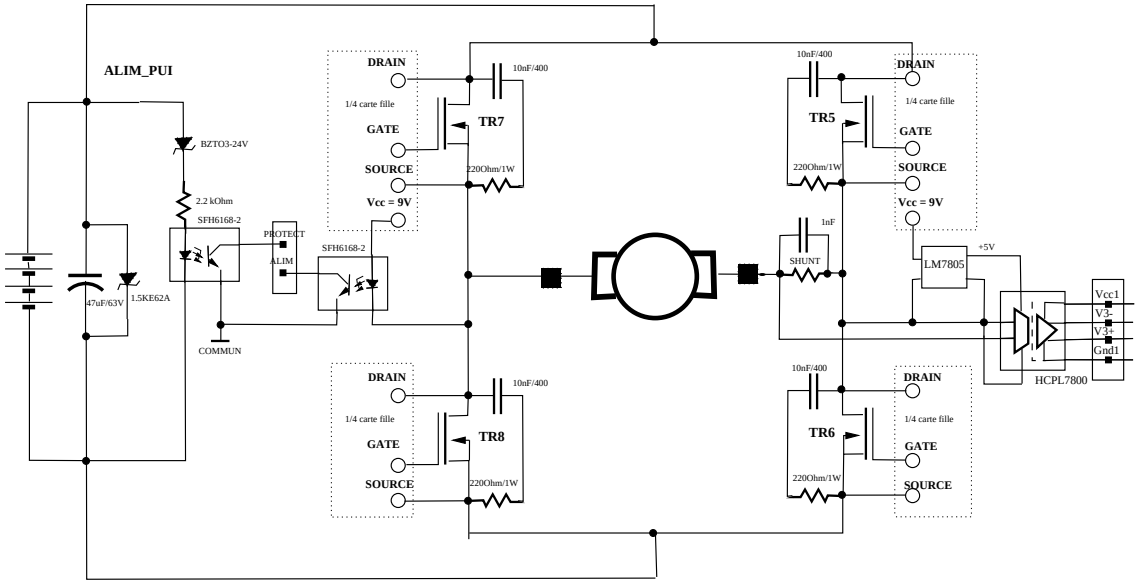


FIG. 2.19: *Carte puissance : le pont H de commande du moteur A*

Width Modulation) qui fait commuter l'alimentation du moteur avec une suite d'impulsions à fréquence fixe et dont on fait varier la largeur. Il existe deux solutions pour contrôler à la fois la vitesse et le sens de rotation du moteur à partir d'un signal PWM :

- la première appelée "Anti Phase Control" permet de commander sens et vitesse avec le seul signal PWM en faisant commuter à chaque cycle les deux diagonales du pont H (dans l'exemple précédent TR6 et TR7 sont commandés par le signal PWM et TR5 et TR8 par son complément) : un rapport cyclique de 100 % produira une vitesse maximale dans le sens direct, un rapport cyclique de 50 % une vitesse nulle et un rapport cyclique de 0 % une vitesse maximale dans le sens opposé. Cette technique PWM est très largement utilisée dans les alimentations à découpage et dans la commande de hâcheurs de puissance.

- L'autre solution demande deux signaux de contrôle, le premier de type Tout ou Rien codant le sens de rotation et le signal PWM dont le rapport cyclique donne la vitesse : 0 % pour une vitesse nulle, 100 % pour la vitesse maximale et 50 % pour une vitesse moitié de la vitesse maximale.

Leszek a utilisé la première solution pour commander les moteurs de traction et de direction et la seconde pour les moteurs de frein de la version Rocquencourt. Une note manuscrite retrouvée par hasard justifie son choix par le fait que seule la première solution permettrait la récupération d'énergie (le chargement des batteries à partir de la fcem induite par les

moteurs de traction en descente par exemple). En effet un MOSFET est un interrupteur qui pemet au courant de passer entre drain et source quand la tension appliquée à la gate est suffisante pour créer le canal. Avec la première méthode, il existe toujours une liaison entre les pôles + et - de la batterie à travers le moteur. Cependant, dans tout MOSFET et en particulier dans ceux spécialement conçus pour leur utilisation dans les hâcheurs de puissance, il existe uen diode parasite intrinsèque inverse entre drain et source qui remplit ce rôle (free wheel diode).
Dans la nouvelle version des nœuds du CyCab Rhône-Alpes la saconde méthode est utilisée. On a pu constater que les pertes par dissipation thermique étaient beaucoup plus faibles ce qui laisserait supposer que la récupération d'énergie est plus efficace.

2.5.2 Mesure du courant dans les branches du H-Pont

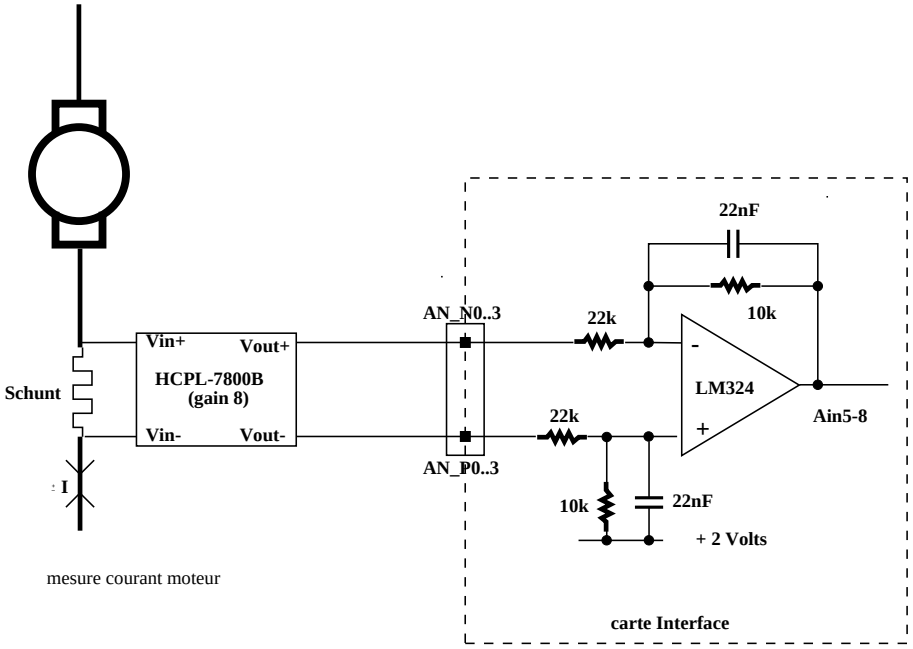


FIG. 2.20: *Mesure du courant moteur*

Pour estimer le courant qui passe dans les moteurs, on fait une mesure analogique de la tension aux bornes d'une résistance de shunt montée en série avec le moteur. Cette tension est d'abord amplifiée par un amplificateur d'isolement HCPL7800 de gain 8, puis par un ampli opérationnel en montage différentiel et avec un gain de 1/2 pour pouvoir être lue par le convertisseur analogique/digital de la carte d'interface. La plage de mesure est de $\pm$ 1

Volt autour de la référence + 2 Volts. La chute de tension aux bornes du shunt doit varier entre :

$$\pm 0.25 Volts = \frac{(\pm 1)\frac{1}{172}}{8}$$

Pour une intensité maximale de 30 Volts dans le moteur, la valeur de la résistance de shunt doit donc être de :

$$R = \frac{0,25}{30} = 0,008\Omega$$

et la puissance dissipée dans ce shunt sera de :

$$\Delta P = I^2 R = 7,2 Watts$$

En guise de shunt, des queues de capa ont été soudées sur la carte puissance (bonjour l'étalonnage !!). Heureusement on ne fait pas d'asservissement en couple est donc la mesure du courant dans les moteurs est inutile.

2.5.3 Commande de Gate des MOS-FET

Sur les cartes filles, on retrouve 4 fois le même motif pour piloter un MOS-FET. Chaque quart de carte fille génère sa propre alimentation de 9 Volts à partir des signaux carrés alternatifs fabriqués sur la carte puissance et disponibles sur les broches ACx1 et ACx2 (x étant le numéro du MOS-FET entre 1 et 8). Cette tension alternative est redressée par un pont de diodes et sert à alimenter les Nand trigger de Schmitt et l'étage amplificateur de sortie de commande de Gate.

Chaque transistor de puissance est commandé à partir d'un signal PWM (Pulse Width Modulation) calculé par le microcontrôleur MC68332 (partie TPU), validé par l'EPLD de la carte Interface et disponible sur le connecteur puissance (INx). Ce signal est optodécouplé (TIL2200), remis en forme (fronts montants retardés par un réseau RC et une porte trigger de Schmitt) puis amplifié pour attaquer la Gate. Un signal d'état FLT traduisant le fait que le transistor ne conduit pas quand il le devrait est calculé et renvoyé après découplage sur l'EPLD de la carte Interface. A partir du signal PWM, une fenêtre d'observation est définie (réseau RC retardant le front montant de 10 microsecondes et trigger). Si pendant l'ouverture de cette fenêtre, le transistor conduit, la tension Drain-Source est voisine de 0 Volt. Cet état logique "0" se retrouve en entrée 5 du Nand 4093, sa sortie 4 est à l'état "1", donc la Led de l'opto-coupleur FLT ne conduit pas. Dans le cas contraire (5 à "1"), on retrouve en 4 l'inverse du signal PWM retardé et le signal de dysfonctionnement FLT est généré.

2.5.4 Protections et bits d'état

Chaque moteur est équipé d'une thermistance qui permet une mesure analogique de la température interne du moteur (voir le chapitre consacré à la carte d'Interface).

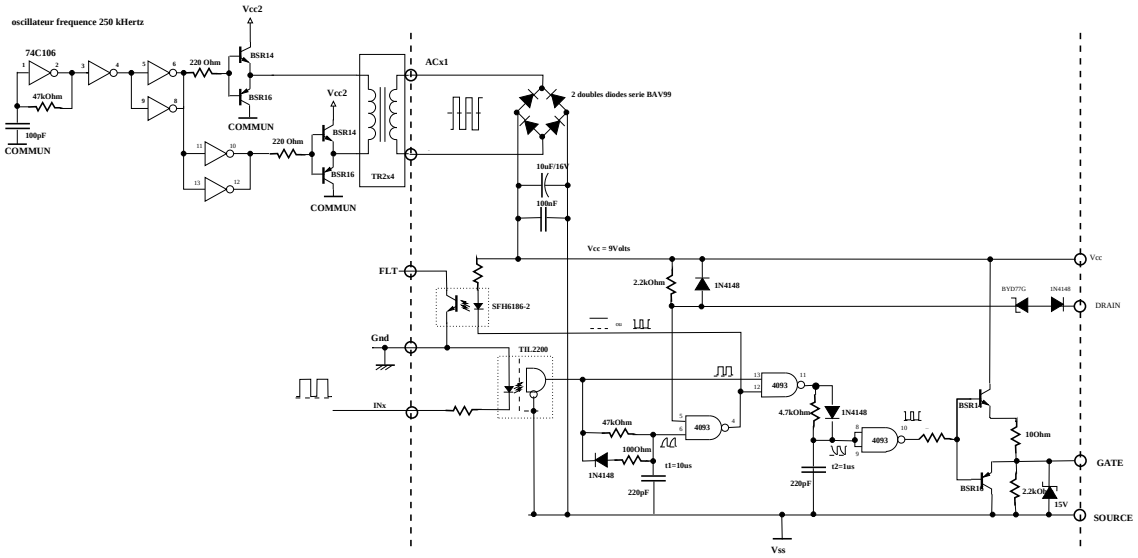


FIG. 2.21: *1/4 de carte fille : driver d'un MOS-FET*

Sur la carte puissance, sont élaborés trois signaux qui participent à la formation par l'EPLD de la carte interface d'un mot d'état au même titre que FLT0 et FLT1 générés sur les cartes filles :

- **TEMP** : Une thermistance, fixée sur le radiateur supportant les MOS-FET, est reliée à la carte Puissance. La tension à ses bornes est optodécouplée puis envoyée à la carte Interface sur laquelle elle sera comparée à un seuil. Le résultat de cette comparaison se retrouve en entrée de l'EPLD pour fournir un des bits du mot d'état.
- **PROTECT** : La diode d'entrée d'un optocoupleur est alimentée en 24 Volts (alim de puissance - tension de Zener de la diode BZT03). Sa sortie traduit la présence du +48 Volts sur le nœud mais peut aussi indiquer que la diode Zener est morte.
- **ALIM** : La présence de la tension continue de 9 Volts (broche Vcc des cartes filles) servant à la génération des commandes de GATE des MOS-FET est vérifiée. Chaque carte fille a une seule broche Vcc. L'une de ces broches sert à alimenter l'ampli d'isolement à travers lequel se fait la mesure du courant dans un des moteurs. L'autre sert à la génération du signal ALIM. La lecture de ce bit d'état à "0" signifie soit que les transistors de l'oscillateur servant à fabriquer les 8 alimentations flottantes (en réalité 6) sont en rideau, soit que le pont redresseur de l'alimentation d'un des transistors a reçu.

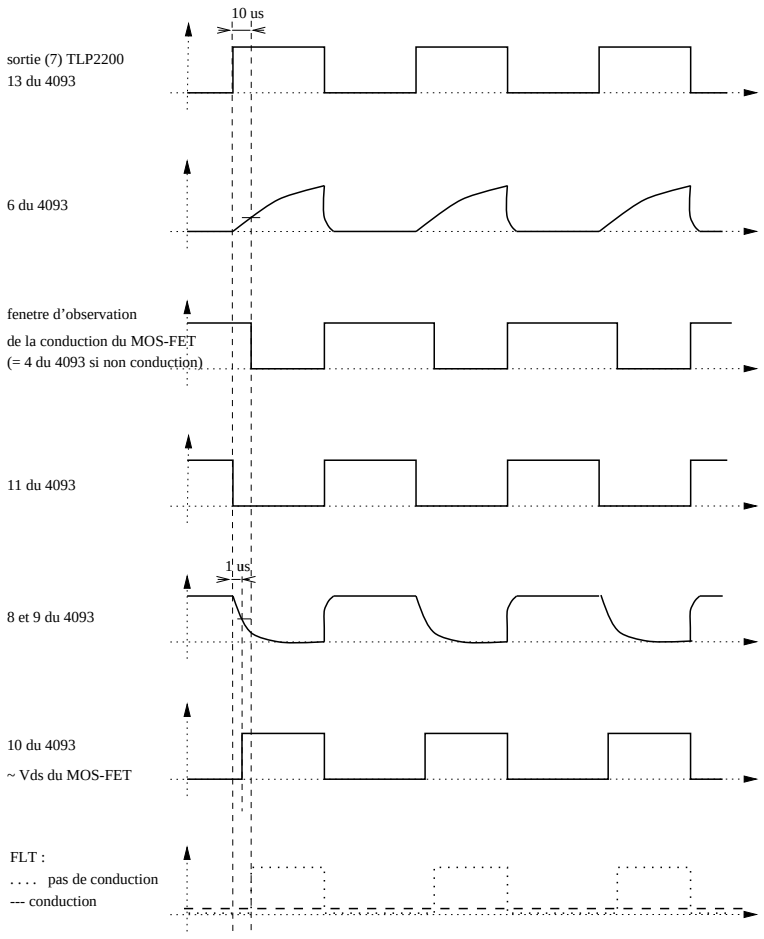


FIG. 2.22: *Chronogramme pour la commande de Gate d'un MOS-FET et génération du signal FLT*

2.6 Remplacement de la partie puissance par des amplis AMC

Cette section concerne le remplacement de la carte puissance des nœuds du CyCab par des SERVO AMPLIFIERS modèle **50A8DD** de Advanced Motion Control.

2.6.1 L'ampli de puissance

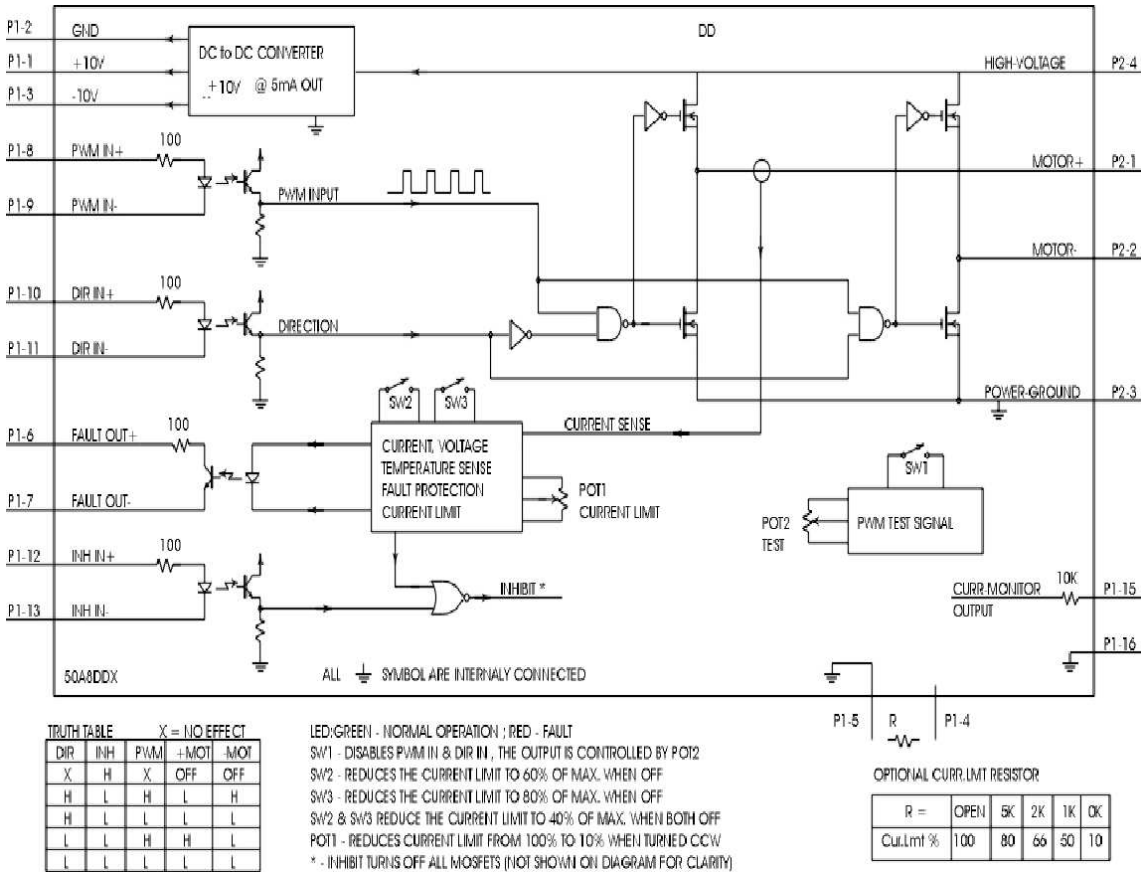


FIG. 2.23: *Servo amplificateur 50A8DD*

Les amplis Advanced Motion Control de la série 50A-DD sont faits pour piloter des moteurs DC à balais à partir d'une entrée PWM. Tous les modèles sont protégés contre les sur-tensions, les sur-courants, les sur-chauffes et les court-circuits sur le moteur ou la

puissance. Pour son fonctionnement l'ampli n'a besoin que d'une alimentation continue non régulée pouvant varier de 20 à 80 Volts. Donc son alimentation directe par les batteries (+48 Volts) est suffisante. Ses amplis sont destinés à être commandés par tout contrôleur qui a une sortie digitale PWM.
L'entrée PWM donne le rapport cyclique de la tension en sortie envoyée au moteur.
L'entrée DIR donne le sens de rotation du moteur, ainsi le moteur peut fonctionner à 100 % de ses possibilités, en marche avant comme en marche arrière alors que dans la solution précédente le signal PWM codait à la fois le rapport cyclique et le sens de rotation (un rapport cyclique de 50 % avait pour effet l'arrêt du moteur).
L'entrée INH permet d'inhiber la génération de tension vers le moteur par soft.
Une sortie FAULT avertit le contrôleur d'un dysfonctionnement (surtension, sur-intensité, sur-chauffe, court-circuit, ...). Il est à noter que ce signal FLT inhibe automatiquement l'amplificateur de puissance, il est impossible de forcer son fonctionnement quand il y a un problème.
La sortie CURR-MONITOR reflète l'intensité du courant qui circule dans le moteur (1 Volt pour 4 Ampères). Elle est callibrée et donc utilisable pour un asservissement en couple.

2.6.2 Connections avec la carte interface

L'idée est dans un premier temps de refaire les nœuds de traction et de direction en utilisant les Servo-Amplifiers AMC à la place de la carte puissance et ce en faisant le moins de modifications possibles au "hard" du nœud. Toutes les connections entre la carte interface et la carte puissance passent par un connecteur femelle 26 points dont seulement 16 connections sont utilisé (8 par moteur : les signaux de commandes PWM, INH, DIR et le retour d'erreur FAULT) plus 4 signaux de masse). Nous avons gardé 6 pistes directement connectées au circuit de logique programmable Lattice, coupé toutes les autres et reliées les 8 connections de masse. On ne fait pas de mesure sur le courant moteur alors que les amplis AMC disposent d'une telle mesure car cela aurait entraîné trop de modification sur la carte. Cette absence de mesure de courant n'est pas génante dans la mesure où elle était faite de manière très approximative pour ne pas dire farfelue.
Dans la version précédente à partir de 2 canaux TPU, l'EPLD Lattice générait les 4 signaux INx permettant de commandre les Gates des MOS-FET du H-Pont associé à un moteur (le sens de rotation et le rapport cyclique étaient codé sur le même canal TPU). Avec les Servo-Amplifier nous avons besoin de 3 canaux TPU par moteur. Il se trouve que les broches TPU4 et TPU5 du BCC sont connectées au Lattice et qu'elles n'étaient pas utilisées dans le design précédent. Elles nous serviront à la génération du sens de rotation respectivement du moteur B et du moteur A.
D'autre part le mot d'état construit par l'EPLD est simplifié : un seul bit pour caractériser le dysfonctionnement d'un moteur. Cette simplification permet de différencier les fin de course gauche et droit pour le nœud de direction.

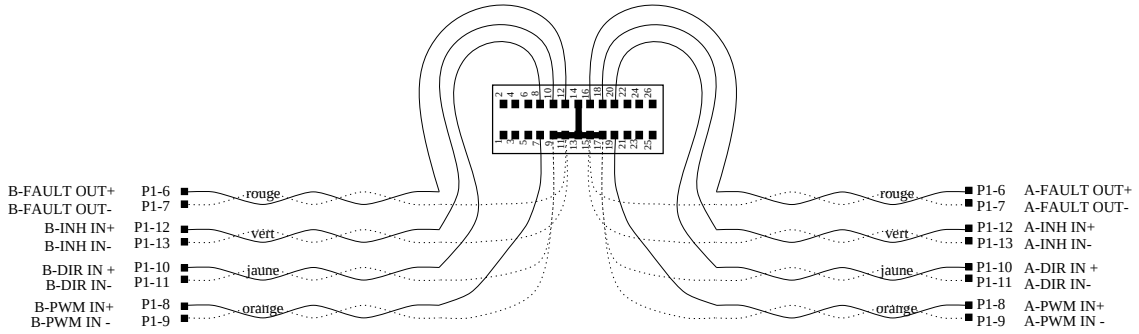


FIG. 2.24: *Connecteur entre la carte Interface et la carte Puissance*

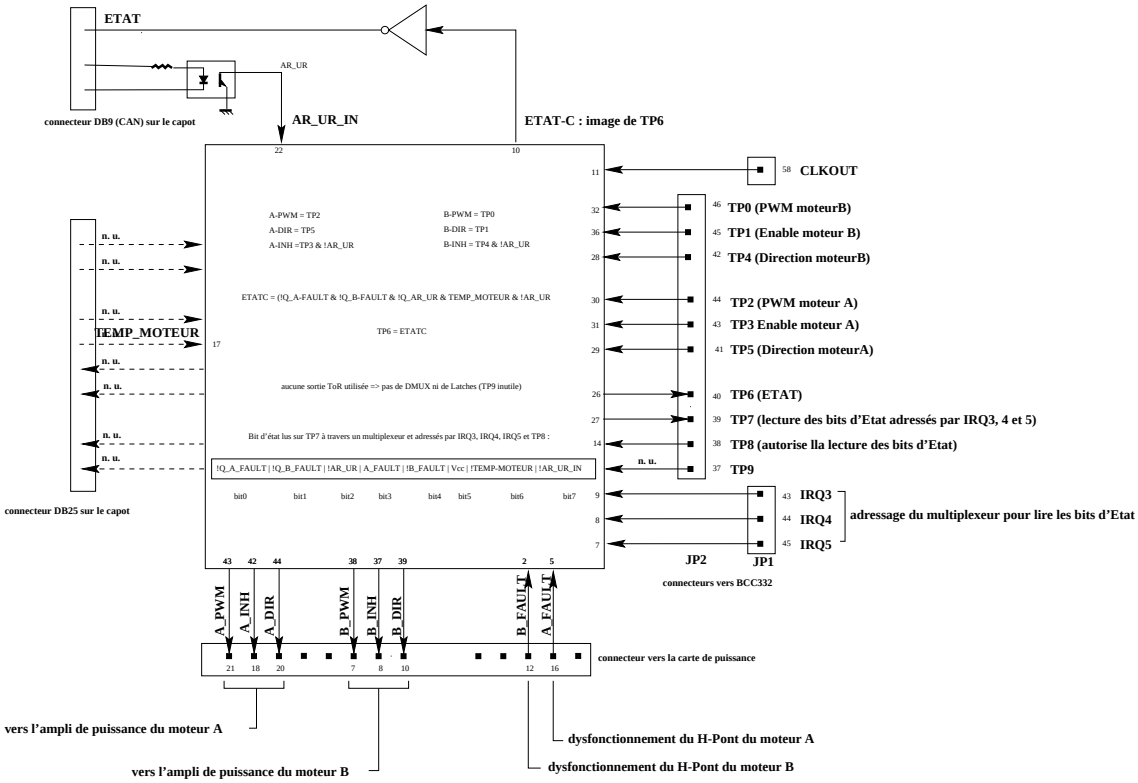


FIG. 2.25: *EPLD de particularisation du nœud de traction*

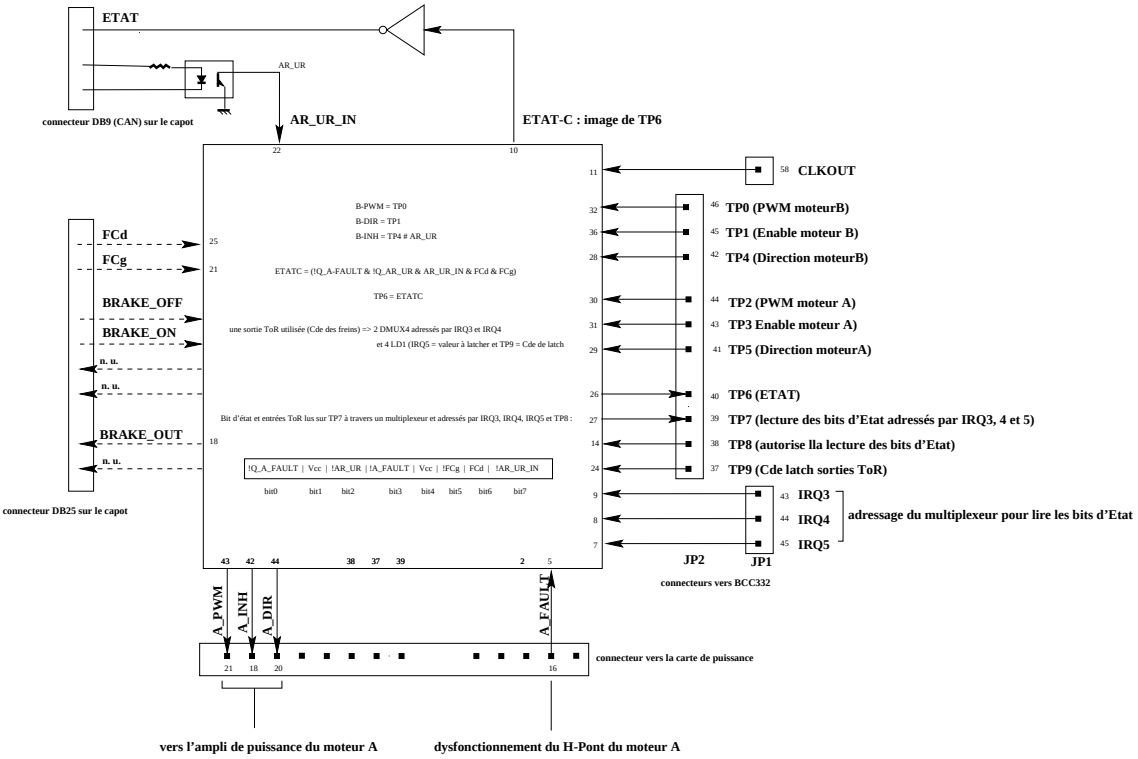


FIG. 2.26: *EPLD de particularisation du nœud de direction*

2.7 Le boîtier et la connectique externe

Les 2 nœuds de traction et celui de direction ont un aspect extérieur identique. Ils sont montés dans le même type de boîtier dont la partie inférieure est constituée par un dissipateur thermique sur lequel sont vissés les 8 transistors de puissance (version LLC). Les capots sont identiques et supportent un voyant indiquant la présence de l'alimentation 15 Volts et les connecteurs : 2 connecteurs DB9 un mâle et un femelle pour la liaison CAN et l'alimentation du nœud (15 Volts), un autre DB9 pour la liaison RS232 (et la liaison vers le joystick pour le nœud de direction) et un connecteur DB25 pour les entrées/sorties. La partie puissance du variateur est disponible sur le connecteur DIN41612 corps H (Alimentation 48 Volts et sortie puissance pour le moteur "A". La sortie puissance pour le moteur "B" se fait par 2 bornes à vis sur le côté du boîtier. Les nœuds de traction et de direction ne diffèrent que

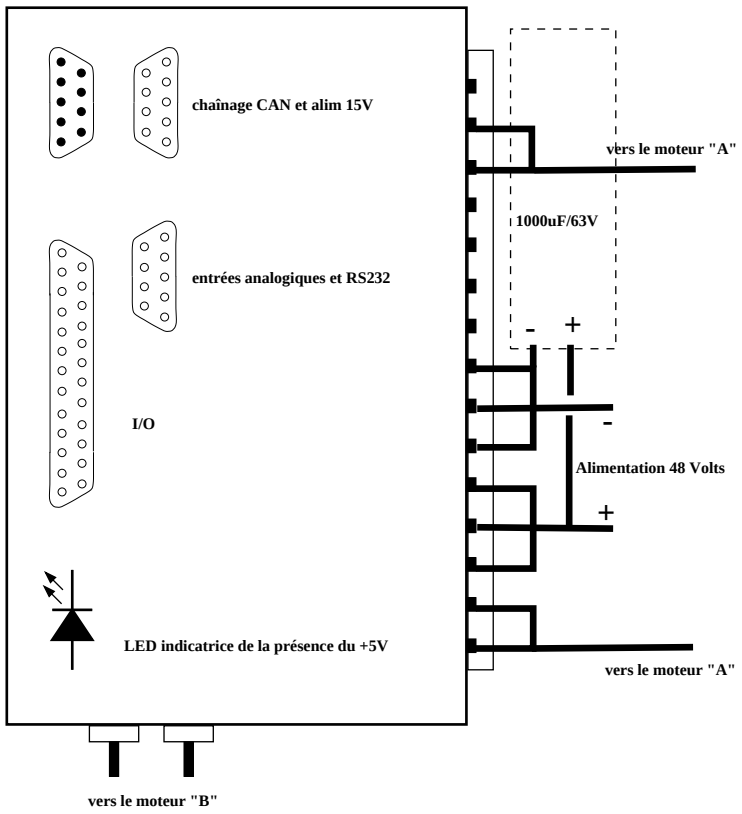


FIG. 2.27: *Le boîtier et ses connecteurs*

par la programmation de l'EPLD qui gère les Entrées/Sorties et donc définit le brochage des connecteurs et plus particulièrement le DB25. Les connecteurs CAN et Alim et la partie

RS 232 du troisième DB9 sont identiques.
TEST des NŒUDS : *pour pouvoir faire marcher les nœuds sur le banc de test, il est nécessaire de "fermer la boucle d'arrêt d'urgence. Pour ce faire il faut alimenter la diode de l'opto-coupleur correspondant. Ceci se fait en reliant la cathode au +15 V de l'alimentation et l'anode au Gnd en réalisant les ponts* **1-5** *et* **9-4**.
D'autre part, le nœud de direction prend en compte les butées du vérin (entrées ToR IN0 et IN1). Pour que le moteur A puisse fonctionner sur le banc de test, il faut simuler le fait que les fins de course ne sont pas atteintes. Ceci se fait en imposant un "1" logique à l'entrée du circuit d'interface HEF40106 (circuit CMOS alimenté en 12 Volts) en réalisant les ponts **6-5** *et* **6-18** *(relier les broches 5 et 18 avec Vcc ne suffit pas : pour un CMOS alimenté en 12 V, 5 V en entrée est vu comme un "0" logique).*

2.8 Le nœud de direction

Son rôle est le contrôle/commande du vérin motorisé, l'interface avec le joystick, et la commande des moteurs de frein de parking.
Le moteur du vérin est alimenté par le connecteur DIN41612. Les fins de course (entrées Tout ou Rien) sont reliés aux broches 18 (IN0) et 5 (IN1) du connecteur DB25. Le codeur incrémental de type HEDS est alimenté à partir des broches 7 (Vcc) et 19 (Gnd) du même connecteur. Les sorties "A", "B" et "I" sont reliées en 8 (COD05), 22 (COD11) et 20 (COD15). Le position absolue du vérin de direction est obtenue par une mesure analogique de la tension aux bornes d'un potentiomètre linéaire de 10 kOhms fixé sur le vérin et monté dans un pont diviseur de tension entre les broches 11 (AVCC) et 23 (AGND). Ce pont diviseur (résistance de 12 kOhms, potentiomètre et résistance de 5,6 kOhms) donne une plage de mesure comprise entre 1 et 2,8 Volts pour les positions extrêmes du potentiomètre. Cette tension est disponible sur la broche 13 (AIN2) du DB25. La température du moteur est mesurée par une thermistance située sur le moteur et connectée sur la broche 12 (AIN3).
La commande du véhicule en mode manuel est donnée par un joystick connecté sur le nœud de direction et alimenté entre AGND et AVCC (respectivement broches 1 et 6 du DB9 RS232). Il délivre deux tensions analogiques : TRA-IN pour la consigne de vitesse et DIR-IN0 pour la consigne de braquage (respectivement broches 4 et 9 du même connecteur).

Le nœud de direction commande aussi les moteurs de frein de parking. Le véhicule prototype dispose d'un frein motorisé par roue, mais les 4 moteurs commandés de la même manière et sans contrôle particulier. Ils sont vus par le nœud comme 2 entrées ToR pour indiquer l'état freiné (IN2 sur la broche 17 du DB25)ou l'état défreiné (IN3 sur la broche 4) et d'une sortie ToR (OUT0 sur la broche 15) pour la commande de changement d'état.

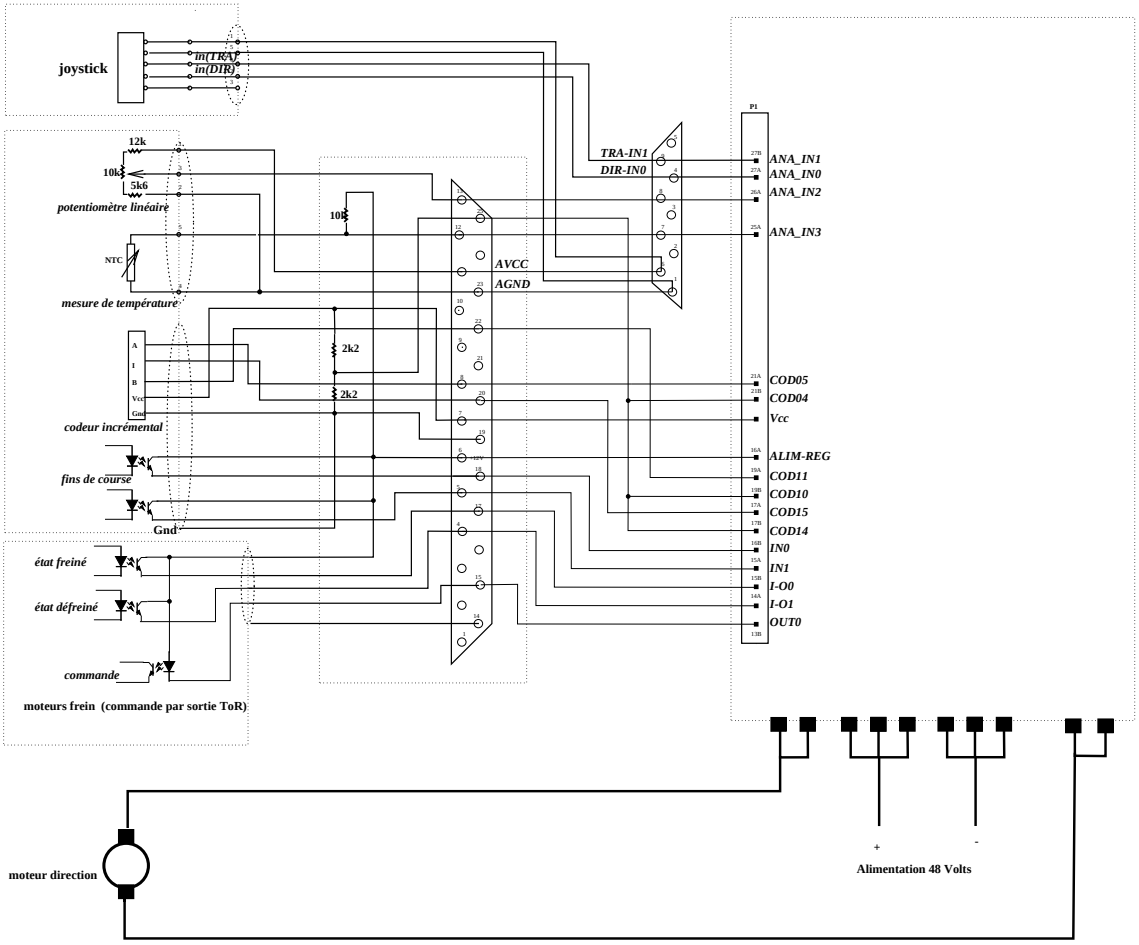


FIG. 2.28: *La connectique du nœud de direction*

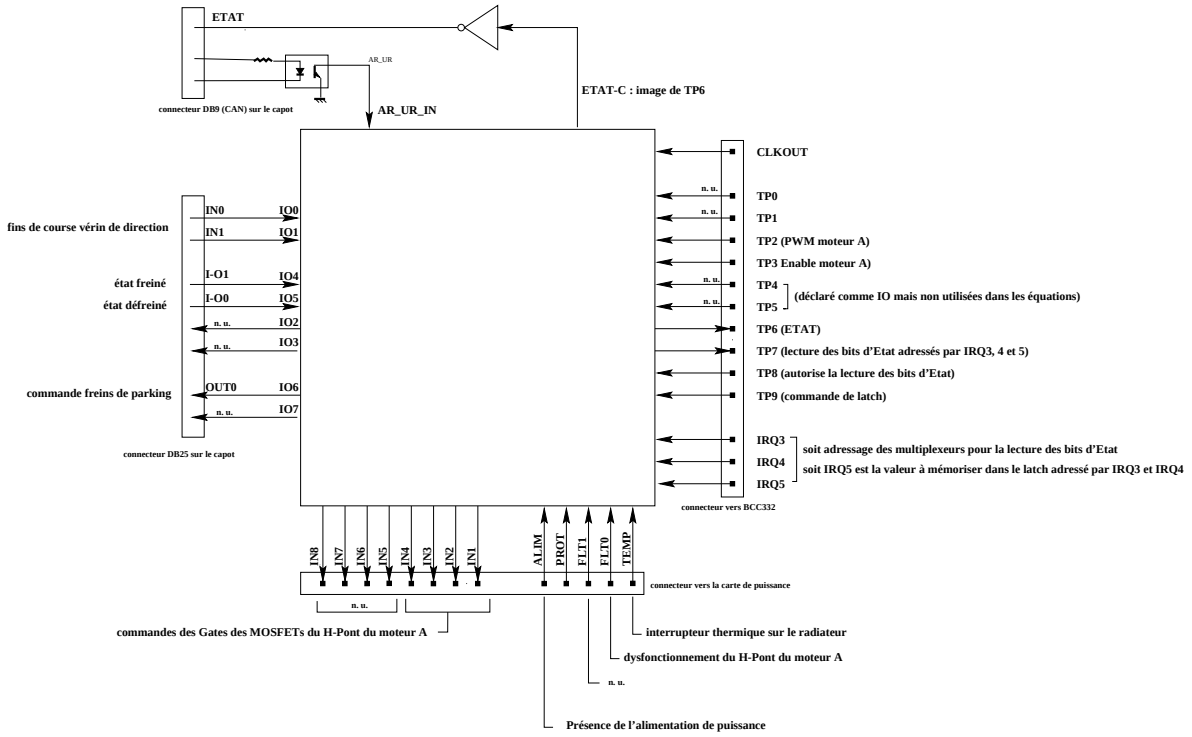


FIG. 2.29: *EPLD de particularisation du nœud de direction*

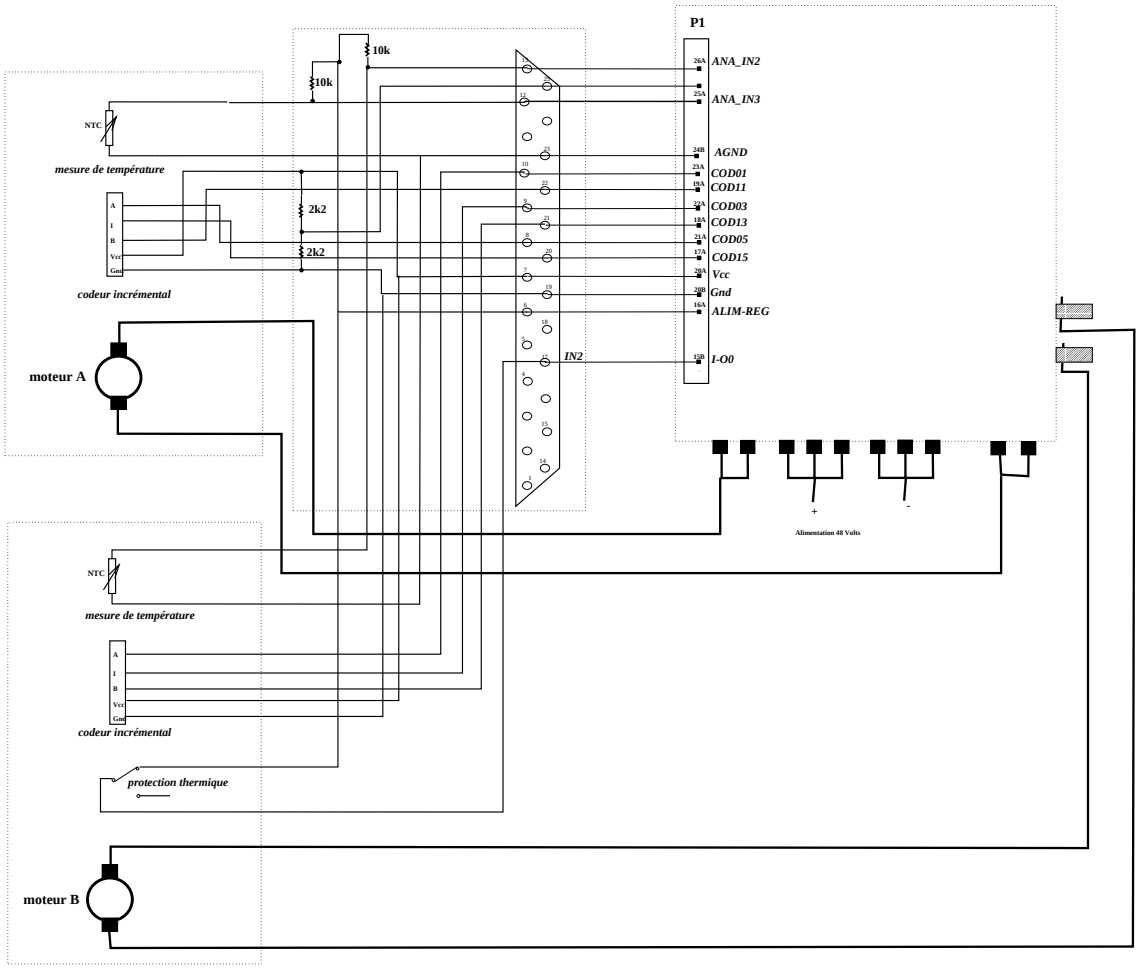


FIG. 2.30: *La connectique du nœud de traction*

2.9 Le nœud de traction

Le nœud de traction commande 2 moteurs séparemment. La puissance vers ces moteurs est disponible sur le connecteur DIN41612 pour l'un et sur les bornes à vis pour l'autre. Chaque moteur est muni d'un codeur incrémental du même type que celui équipant le moteur du vérin motorisé et d'une thermistance pour la mesure analogique de la température du moteur. En outre un des deux moteurs est équipé d'un interrupteur thermique de sécurité utilisant une entrée ToR. Les broches utiles du connecteur DB25 sont donc: les 6 entrées codeur, les canaux analogiques 1 et 2 et l'entrée ToR IN0.

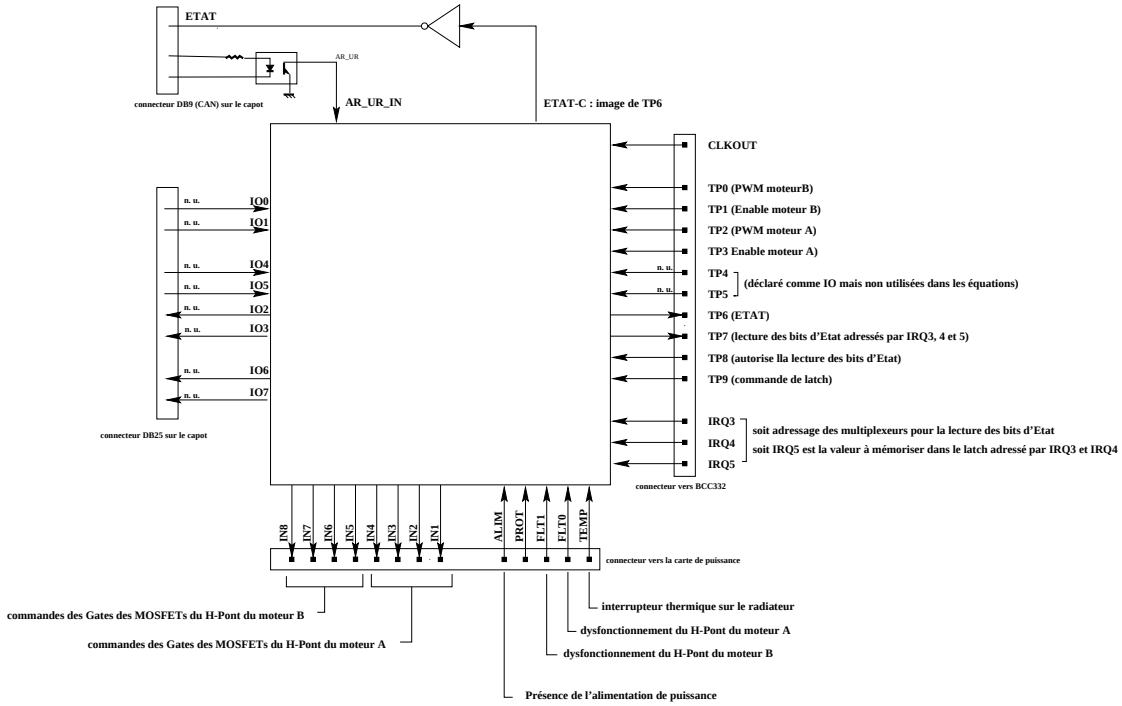


FIG. 2.31: *EPLD de particularisation du nœud de traction*

Chapitre 3

Software du Cycab

3.1 Description de l'implémentation Logicielle sur le CYCAB Rhône-Alpes

Comme pour le matériel, le logiciel du CYCAB Rhône-Alpes est différent de celui de Rocquencourt. A Rocquencourt, le logiciel SynDEx [9] est utilisé pour programmer les applications du véhicule. Pour cela, le noyau a été porté pour les noeuds MC68832 et le PC sous dos qui initialise les applications. La description par flot de données d'une application est ensuite répartie automatiquement par SynDEx sur les différents calculateurs du véhicule. Pour le CYCAB Rhône-Alpes, nous disposions d'algorithmes de contrôle bas-niveau des moteurs placés "manuellement" sur les noeuds et d'un protocole basique de communication entre les noeuds.

Pour le CYCAB Rhône-Alpes, nous n'avons pas pu passer le logiciel sous SynDEx pour plusieurs raisons:

- nous n'avions pas les ressources humaines nécessaires pour mener de front l'analyse des problèmes du véhicule rhônalpin et la refonte du logiciel,

- et nous avions la volonté d'utiliser le logiciel de contrôle-commande ORCCAD, pour l'instant non-compatible avec l'approche SynDEx.

Le fort investissement que nous aurions du faire pour acquérir les compétences SynDEx, et rendre les logiciels ORCCAD et SynDEx compatibles nous a fait renoncer. Sur la base du logiciel existant, nous l'avons simplement épuré et deboggué.

3.1.1 Rappels sur l'informatique embarquée du CYCAB Rhône-Alpes

L'architecture matérielle du CYCAB (confére le schéma synoptique 1.2) est composée de :

- une carte MVME162 [11], avec l'OS VxWorks 5.3 [14], un module IP-CAN (Controller Area Network) [12], et une liaison ethernet HF [13].
- trois Modules (traction avant, traction arrière, direction-joystick) composés chacun d'un microprocesseur 68332, d'une interface CAN, et d'une interface RS232 (liaison série).
- des entités peuvent être rapportées sur le bus CAN (ultra-sons, caméra linéaire)

3.1.2 Choix sur la communication

Quelques règles générales...

- l'interface série des Modules ne sert qu'à la mise au point,
- le bus CAN est le seul moyen de communication entre la carte MVME162 et les Modules,
- la carte MVME162 dialogue avec le réseau (pour l'amorçage par exemple) via le le lien éthernet HF,

Du point de vue du bus CAN, la carte MVME162 est maître, alors que les Modules (ou les autres entités CAN) sont esclaves. C'est donc la carte MVME162 qui prend l'initiative de la communication, le ou les Modules se contentent de rester à l'écoute et de répondre si on le leur demande.

3.1.3 Environnement de Programmation

Pour la carte MVM162 (VxWorks), la chaine de développement croisée Solaris-68k est utilisée. (compilation, loader dynamique,...)

La partie GNU (compilateur croisé solaris-68k) est aussi utilisée pour développer le code des modules. En effet, le coeur du micro-controleur 68332 est celui du 68020. Le code est chargé dans la mémoire flash se trouvant dans le module via la liaison série ou via le bus CAN. A l'amorçage du 68332, le code en Flash est recopié en mémoire, puis exécuté.

Le répertoire de base qui contient toutes les informations sur le CYCAB est *CYCAB* . Les logiciels bas-niveau se trouve dans le sous-répertoire **driver** qui contient 5 packages:

- **load_cpu332** contient les sources et le fichier *load.hex*, un exécutable qui se télécharge via la liaison RS232 sur un module, pour permettre le chargement en flash d'un programme.

- **module/MC68332** contient les sources et les fichiers *moduleX.hex* qui sont chargés en flash de chaque module MC68332. Chaque Module possède un identifiant unique, et donc un fichier généré unique, puisque il a été décidé de ne pas avoir d'allocation dynamique d'identifiant. Il y a, de plus, deux versions de Module, l'une avec une électronique de puissance "maison" (typage module_), l'autre avec une électronique de puissance "sur etagère" (typage moduleB_). Pour les trois Modules du CYCAB, six fichiers sont créés à la compilation. Depuis peu, tous les modules sont en version B, les fichiers utilisés sont donc moduleB_1.hex (train avant), moduleB_2 (train arrière).hex et moduleB_3.hex (direction). Ce logiciel assure les asservissements bas-niveau et la lecture des capteurs proprioceptifs.

- **module/vxworks** qui contient les sources et le fichier *drvModule.vxo* implémente un menu ascii de test des Modules (*moduleMenu()*).

- **lib_com** contient les sources et le fichier *comCycab.vxo* pour l'implémentation de la communication sur le CAN pour la MVM162. L'implémentation de ce protocole CAN_CYCAB est décrit par la suite. Cette librairie utilise les services bas-niveau de la librairie IP-CAN livrée avec le module IP. Elle regroupe les fonctions essentielles (init, send, receive, close).

- **cycab** contient les sources et les fichiers objets pour la carte MVME162 et implémente une couche d'intégration permettant de discuter avec les composants placés sur le CAN. Chaque composant nécessite deux fichiers du type cycabComposant.c et cycabComposant.h. Ces fichiers décrivent les différents appels à utiliser pour piloter le sous-ensemble entité du CYCAB.

  cycabModule permet de s'adresser aux modules MC68332, la vue abstraite de ce module est donné par la suite,

  cycabUs permettra de s'adresser à la ceinture ultra-sons,

  cycabCl permettra de s'adresser à la caméra linéaire.

3.1.4 Modèle d'exécution du logiciel sur les noeuds

A l'amorçage du micro-contrôleur, quatre tâches se déroulent en concurrence:

- une routine d'interruption reçoit les messages venant du CAN et les empile dans une FIFO.

- une routine d'interruption reçoit les messages venant de la liaison série et les empile dans une FIFO.

- le programme principal (main), dépile et traite les messages. Il peut émettre un message CAN ou une sortie sur la liaison série s'il le faut.

– une routine d'interruption déclenchée sur un TIMER à 10 millisecondes, permet de dérouler des fonctions (périodiques). Celles-ci peuvent être autorisées ou non dynamiquement. A titre d'exemple, sur chaque Module de traction se trouve une routine TIMER qui annule la sortie puissance si celle-ci n'est pas mise a jour régulièrement. Sur le Module Direction-Joystick, une routine TIMER permet l'asservissement en position de la direction.

Il est à noter que le protocole de plus haut niveau, décrit par la suite, (un seul maitre, acquittement permanent) permet d'éviter tout blocage sur le 68332.

Le modèle d'exécution est illustré par la figure 3.1.

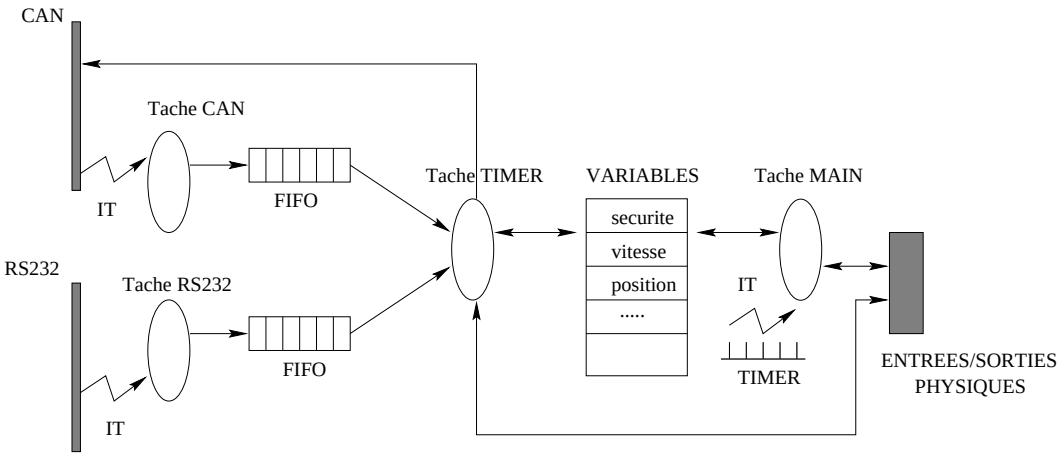


FIG. 3.1: *Modèle d'Exécution*

Le noeud 68332 qui écoute et envoit des messages sur le bus CAN suit le protocole CAN-CYCAB que nous définissons dans la partie suivante.

3.1.5 Le Protocole CAN-CYCAB

Chaque entité sur le bus CAN doit se conformer au protocole suivant.

En résumant, les transferts sur le bus CAN sont basés sur des trames, contituées de 11 bits d'entête et de 8 octets de message.

L'entête est utilisé pour identifier le destinataire de manière unique (8 bits) et le type de message (3 bits).

Les différents types de messages sont:

– envoi d'une requête, avec ou sans donnés et avec ou sans demande d'acquittement,

– réponse à une requête, avec ou sans donnés.

Les 8 octets de messages se décomposent en:

- un octet pour l'identifiant de l'envoyeur,
- un octet pour l'identifiant du message,
- un octet pour le mot de commande à exécuter,
- quatre octets pour les données éventuelles.

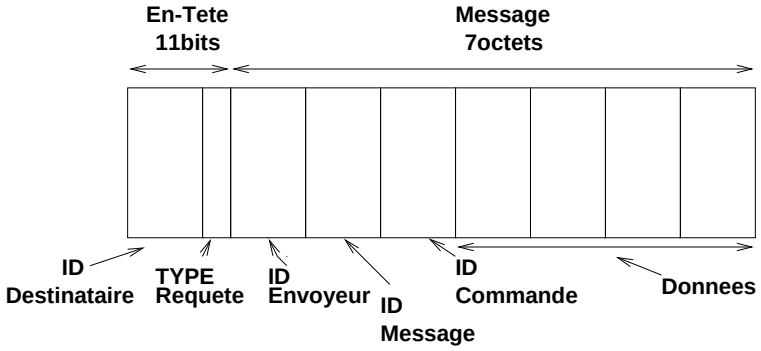


FIG. 3.2: *Trame CAN.*

Dans la version actuelle du CYCAB, seules des requêtes avec demande d'acquittement sont utilisées.

3.1.6 Pilotage du CYCAB via la liaison série

Comme nous l'avons vu précédemment cette façon de procéder n'est utilisée qu'en développement et maintenance.

A la mise sous tension d'un Module, le menu suivant apparait et permet d'utiliser le Module. Chaque Module possède deux sortie puissance (en mode PWM) référencée A et B.

```
BEGIN_MOTOR_A
BEGIN_MOTOR_B        : autorise la partie puissance
ABORT_MOTOR_A
ABORT_MOTOR_B        : neutralise la partie puissance
PUT_PWM_A
PUT_PWM_B            : consigne PWM (entre -100 et +100)
GET_POSITION_MOTOR_A
GET_POSITION_MOTOR_B : lecture des codeurs
GET_ANALOG           : lecture d'une entrée analogique (0,1,2,3)
GET_DIGITAL          : lecture d'une entrée TOR (0,1,2,3)
```

```
PUT_DIGITAL_ON       : mise à zéro d'une sortie TOR (0,1,2,3)
PUT_DIGITAL_OFF      : mise à un d'une sortie TOR (0,1,2,3)
GET_ID               : lecture de l'identifiant du module
GET_STATE,
GET_STATE_BYTE       : lecture du status du module
LOOP_OFF_RING
LOOP_ON_RING         : mise en place ou non d'une routine
                       d'interruption TIMER (0,1,2,...)
FLASH_INIT
FLASH_LOAD
FLASH_QUIT           : utilisation de l'écriture en flash
PUT_DIR              : consigne en position de la direction
       (entre -3500 et +3500)
RETURN_TO_DEBUGER    : retour au debugger du 68332
```

3.1.7 Pilotage du CYCAB via le CAN

Une première méthode consiste à utiliser un menu du même type que celui de la liaison série. Il est lui aussi reservé au développement et à la maintenance. En plus des commandes décrites ci-dessus, apparaissent:

- la possibilité de changer de Module,
- le chargement d'un fichier type hex format en flash,
- un test de vitesse de transmission sur le bus CAN.

L'utilisateur, pour ces démonstrations, utilisera la librairie C dont les fonctions sont décrites ci-après (extrait du fichier d'include C).

```
/* etat = 0, cycab stopped by brakes
   etat = -1, cycab can run */
extern int cycabPutBrake(int etat);
extern int cycabGetBrake(int *etat);

/* etat = 0, enable motor
   etat = -1, disable motor */
extern int cycabPutMotor(int etat);
extern int cycabGetMotor(int *etat);

/* a?? between -100 and +100) */
extern int cycabPutPwm(int avg, int avd, int arg, int ard);
/* pas de cycabGetPwm */
```

```
/* a?? in top coder */
extern int cycabGetPos(int *avg, int *avd, int *arg, int *ard);

/* etat = 0, direction loop TIMER_routine is activated
   etat = -1, direction loop TIMER _routine is disactivated */
extern int cycabPutDir(int etat);
extern int cycabGetDir(int *etat);

/* centieme_degre in 1/100 deg (between -3500 and +3500) */
extern int cycabPutDirPos(int centieme_degre);
extern int cycabGetDirPos(int *centieme_degre);

extern int cycabGetDirCod(int *top);

/* etat_??? bit 0  1  2  3  4  5  6  7
               F0 F1 AU TE AL Fg Fd AR
F0 fault on motor A
F1 fault on motor B
AU emergency stop activated
TE not used anymore
AL not used anymore
Fg left switch limit (on direction only)
Fd right switch limit (on direction only)
AR emergency stop memorised
*/
extern int cycabGetState(int *etat_ava, int *etat_arr,
                         int *etat_dir);

/* j_??? between -100 and +100 */
extern int cycabGetJoystick(int *j_dir, int *j_tra);
```

Chapitre 4

Données en vue d'expérimentations

4.1 Cinématique du Cycab

4.1.1 Notations

Soient $F = (x_F, y_F)$ (resp. $R = (x_R, y_R)$) les coordonnées du centre de l'essieu avant (resp. arrière). De même, v_F (resp. v_R) représente la vitesse instantanée au point F (resp. R). ϕ représente le braquage moyen des roues de l'essieu avant. l_w correspond à l'empattement du véhicule. θ caractérise l'angle que fait l'axe longitudinal du véhicule avec l'axe des abscisses du repère lié à l'environnement. Enfin, ρ_F (resp. ρ_R) est le rayon de giration instantané associé au point F (resp. R).

4.1.2 Véhicule Ligier

Seules les roues avant sont directrices. La vitesse de rotation instantanée du véhicule peut s'écrire :

$$\dot{\theta} = \frac{v_R}{\rho_R} = \frac{v_F}{\rho_F} \tag{4.1}$$

De manière géométrique, on trouve aisément :

$$\rho_R = \frac{l_w}{|tan(\phi)|} = \rho_F * cos(\phi) \tag{4.2}$$

A partir de (4.1) et (4.2), on déduit :

$$v_R = v_F * cos(\phi) \tag{4.3}$$

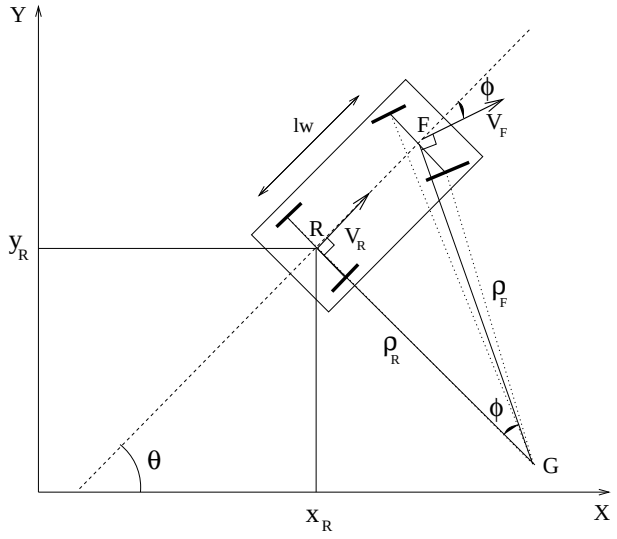


FIG. 4.1: *Modélisation du véhicule Ligier.*

On en déduit les équations du mouvement en R :

$$\left\{\begin{array}{rcl} \dot{x_R} & = & v_R * cos(\theta) \\ \dot{y_R} & = & v_R * sin(\theta) \\ \dot{\theta} & = & v_R * \dfrac{tan(\phi)}{l_w} \end{array}\right. ou \left\{\begin{array}{rcl} \dot{x_R} & = & v_F * cos(\phi) * cos(\theta) \\ \dot{y_R} & = & v_F * cos(\phi) * sin(\theta) \\ \dot{\theta} & = & v_F * \dfrac{sin(\phi)}{l_w} \end{array}\right. \tag{4.4}$$

De même, on en déduit les équations du mouvement en F :

$$\left\{\begin{array}{rcl} \dot{x_F} & = & v_F * cos(\theta + \phi) \\ \dot{y_F} & = & v_F * sin(\theta + \phi) \\ \dot{\theta} & = & v_F * \dfrac{sin(\phi)}{l_w} \end{array}\right. ou \left\{\begin{array}{rcl} \dot{x_F} & = & v_R * \dfrac{cos(\theta + \phi)}{cos(\phi)} \\ \dot{y_F} & = & v_R * \dfrac{sin(\theta + \phi)}{cos(\phi)} \\ \dot{\theta} & = & v_R * \dfrac{tan(\phi)}{l_w} \end{array}\right. \tag{4.5}$$

4.1.3 Véhicule CyCab

Les roues avant et arrière sont directrices avec un coefficient de proportionnalité entre les deux braquages : si le braquage avant est de ϕ, le braquage arrière est de $k\phi$ (avec $k \in [0, 1]$). La valeur de k sera à déterminer par la suite. On introduit le point H défini comme la projection orthogonale du centre de giration instantanée G sur l'axe longitudinal du véhicule.

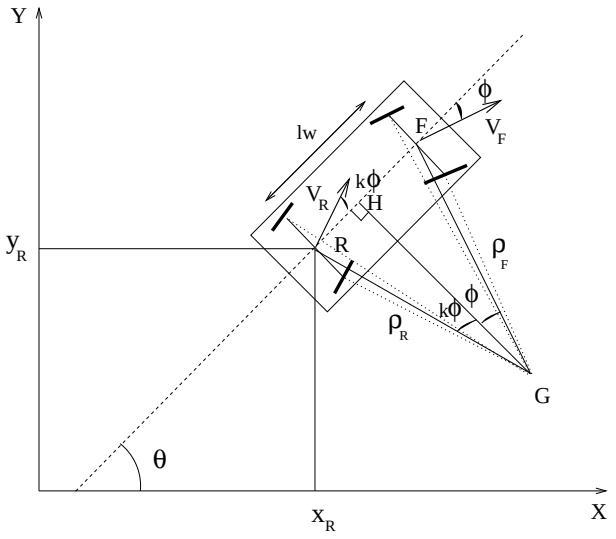


FIG. 4.2: *Modélisation du véhicule CyCab.*

La distance GH peut être exprimée de la manière suivante :

$$GH = \frac{RH}{|tan(k\phi)|} = \frac{HF}{|tan(\phi)|} \tag{4.6}$$

ce qui permet d'obtenir :

$$RH = HF * \frac{tan(k\phi)}{tan(\phi)} \tag{4.7}$$

de plus :

$$RH + HF = l_w \tag{4.8}$$

On en déduit les expressions de RH et HF en fonction de l_w et ϕ :

$$\begin{cases} RH &= \dfrac{l_w}{1 + \frac{tan(\phi)}{tan(k\phi)}} &= l_w \dfrac{cos(\phi) * sin(k\phi)}{sin(\phi + k\phi)} \\ HF &= \dfrac{l_w}{1 + \frac{tan(k\phi)}{tan(\phi)}} &= l_w \dfrac{cos(k\phi) * sin(\phi)}{sin(\phi + k\phi)} \end{cases} \tag{4.9}$$

Les rayons de braquage ρ_R et ρ_F sont donnés par :

$$\begin{cases} \rho_R &= \dfrac{RH}{|sin(k\phi)|} &= l_w * \dfrac{cos(\phi)}{|sin(\phi + k\phi)|} \\ \rho_F &= \dfrac{HF}{|sin(\phi)|} &= l_w * \dfrac{cos(k\phi)}{|sin(\phi + k\phi)|} \end{cases} \tag{4.10}$$

La vitesse de rotation instantanée peut s'écrire:

$$\dot{\theta} = \frac{v_R}{\rho_R} = \frac{v_F}{\rho_F} \tag{4.11}$$

A partir de (4.10) et (4.11), on déduit:

$$v_R = v_F * \frac{\rho_R}{\rho_F} = v_F * \frac{cos(\phi)}{cos(k\phi)} \tag{4.12}$$

Les équations du mouvement en R sont obtenues de manière géométrique:

$$\left\{ \begin{array}{rcl} \dot{x_R} & = & v_R * cos(\theta + k\phi) \\ \dot{y_R} & = & v_R * sin(\theta + k\phi) \\ \dot{\theta} & = & v_R * \frac{sin(\phi + k\phi)}{l_w * cos(\phi)} \end{array} \right. ou \left\{ \begin{array}{rcl} \dot{x_R} & = & v_F * \frac{cos(\phi)}{cos(k\phi)} * cos(\theta + k\phi) \\ \dot{y_R} & = & v_F * \frac{cos(\phi)}{cos(k\phi)} * sin(\theta + k\phi) \\ \dot{\theta} & = & v_F * \frac{sin(\phi + k\phi)}{l_w * cos(k\phi)} \end{array} \right. \tag{4.13}$$

De même, on en déduit les équations du mouvement en F:

$$\left\{ \begin{array}{rcl} \dot{x_F} & = & v_F * cos(\theta + \phi) \\ \dot{y_F} & = & v_F * sin(\theta + \phi) \\ \dot{\theta} & = & v_R * \frac{sin(\phi + k\phi)}{l_w * cos(\phi)} \end{array} \right. ou \left\{ \begin{array}{rcl} \dot{x_F} & = & v_R * \frac{cos(k\phi)}{cos(\phi)} * cos(\theta + \phi) \\ \dot{y_F} & = & v_R * \frac{cos(k\phi)}{cos(\phi)} * sin(\theta + \phi) \\ \dot{\theta} & = & v_F * \frac{sin(\phi + k\phi)}{l_w * cos(k\phi)} \end{array} \right. \tag{4.14}$$

remarque: valeurs particulières de k

- k=0 : on est dans le cas de la Ligier. On retrouve d'ailleurs les équations cinématiques de la section &4.1.2.
- k=1 : les points R et F ont exactement la même cinématique. Cela est également vérifié à partir des équations (4.12) et (4.14).

4.1.4 Détermination des vitesses de rotation des roues

On défini L comme la voie du véhicule (i.e. longueur des essieux) et r comme le rayon d'une roue. Les points RL, RR, FL et FR correspondent aux centres des roues arrière gauche, arrière droit, avant gauche et avant droit. A partir de la figure 4.3, on peut aisément déduire:

$$\left\{ \begin{array}{rcl} x_{RL} & = & x_R - \frac{L}{2} * sin(\theta) \\ y_{RL} & = & y_R + \frac{L}{2} * cos(\theta) \end{array} \right. et \left\{ \begin{array}{rcl} x_{RR} & = & x_R + \frac{L}{2} * sin(\theta) \\ y_{RR} & = & y_R - \frac{L}{2} * cos(\theta) \end{array} \right. \tag{4.15}$$

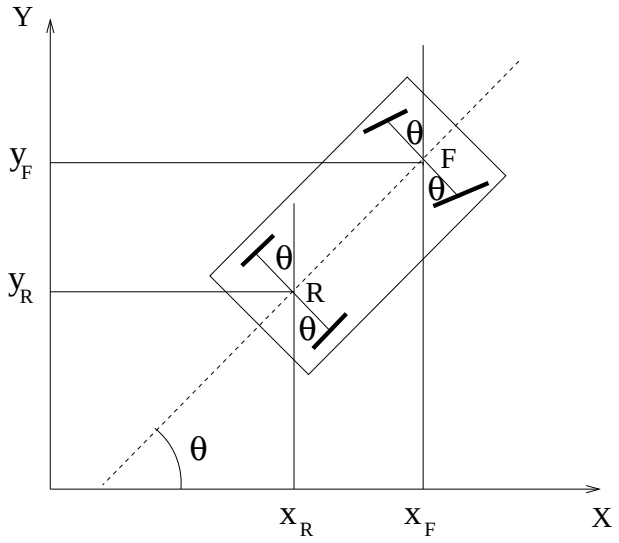


FIG. 4.3: *Mise en évidence de l'orientation θ aux centres des essieux.*

pour les roues arrières.

$$\left\{\begin{array}{lcl} x_{FL} & = & x_F - \frac{L}{2} * sin(\theta) \\ y_{FL} & = & y_F + \frac{L}{2} * cos(\theta) \end{array}\right. et \left\{\begin{array}{lcl} x_{FR} & = & x_F + \frac{L}{2} * sin(\theta) \\ y_{FR} & = & y_F - \frac{L}{2} * cos(\theta) \end{array}\right. \quad (4.16)$$

pour les roues avant.
On en déduit les vitesses en ces points :

$$\left\{\begin{array}{lcl} \dot{x}_{RL} & = & \dot{x}_R - \frac{L}{2} * \dot{\theta} * cos(\theta) \\ \dot{y}_{RL} & = & \dot{y}_R - \frac{L}{2} * \dot{\theta} * sin(\theta) \end{array}\right. et \left\{\begin{array}{lcl} \dot{x}_{RR} & = & \dot{x}_R + \frac{L}{2} * \dot{\theta} * cos(\theta) \\ \dot{y}_{RR} & = & \dot{y}_R + \frac{L}{2} * \dot{\theta} * sin(\theta) \end{array}\right. \quad (4.17)$$

pour les roues arrière.

$$\left\{\begin{array}{lcl} \dot{x}_{FL} & = & \dot{x}_F - \frac{L}{2} * \dot{\theta} * cos(\theta) \\ \dot{y}_{FL} & = & \dot{y}_F - \frac{L}{2} * \dot{\theta} * sin(\theta) \end{array}\right. et \left\{\begin{array}{lcl} \dot{x}_{FR} & = & \dot{x}_F + \frac{L}{2} * \dot{\theta} * cos(\theta) \\ \dot{y}_{FR} & = & \dot{y}_F + \frac{L}{2} * \dot{\theta} * sin(\theta) \end{array}\right. \quad (4.18)$$

pour les roues avant.

Dans tous les cas, les vitesses des roues correspondent à la vitesse du centre de l'essieu concerné (i.e. v_R ou v_F) $\pm\Delta v$ avec :

$$\Delta v = \frac{L}{2} * \dot{\theta} \tag{4.19}$$

On en déduit donc les vitesses linéaires et de rotation des 4 roues :

$$\begin{cases} v_{RL} &= v_R - \frac{L}{2} * \dot{\theta} \\ v_{RR} &= v_R + \frac{L}{2} * \dot{\theta} \\ v_{FL} &= v_F - \frac{L}{2} * \dot{\theta} \\ v_{FR} &= v_F + \frac{L}{2} * \dot{\theta} \end{cases} \tag{4.20}$$

$$\begin{cases} \dot{\psi}_{RL} &= \dot{\psi}_R - \frac{L}{2r} * \dot{\theta} \\ \dot{\psi}_{RR} &= \dot{\psi}_R + \frac{L}{2r} * \dot{\theta} \\ \dot{\psi}_{FL} &= \dot{\psi}_F - \frac{L}{2r} * \dot{\theta} \\ \dot{\psi}_{FR} &= \dot{\psi}_F + \frac{L}{2r} * \dot{\theta} \end{cases} \tag{4.21}$$

avec $\dot{\psi}_R = \frac{v_R}{r}$ et $\dot{\psi}_F = \frac{v_F}{r}$.

Détermination du paramètre k

La détermination du rapport entre les braquages moyens des roues avant et arrière (ou plutôt entre les braquages des roues virtuelles situées respectivement au centre de chacun des essieux) n'est pas chose aisée. Un programme a alors été écrit pour déterminer ce coefficient. La figure 4.5 montre, entre autre, l'évolution de k en fonction de l'angle ϕ correspondant à l'orientation de la manivelle avant de direction par rapport à l'axe longitudinal du véhicule (cf. figure 4.6). Comme on peut le voir, ce coefficient n'est pas constant mais il semble raisonnable de le ramener à une valeur constante soit :

$$k = 0.69 \tag{4.22}$$

sur l'ensemble des valeurs admissibles pour ϕ c'est à dire environ $[-0.4, 0.4]$.
remarque : La notation ϕ caractérise indifféremment le braquage moyen avant (i.e. braquage de la roue virtuelle située au centre de l'essieu avant) et l'angle entre la manivelle avant de direction et l'axe longitudinal du véhicule (cf. figure 4.5).

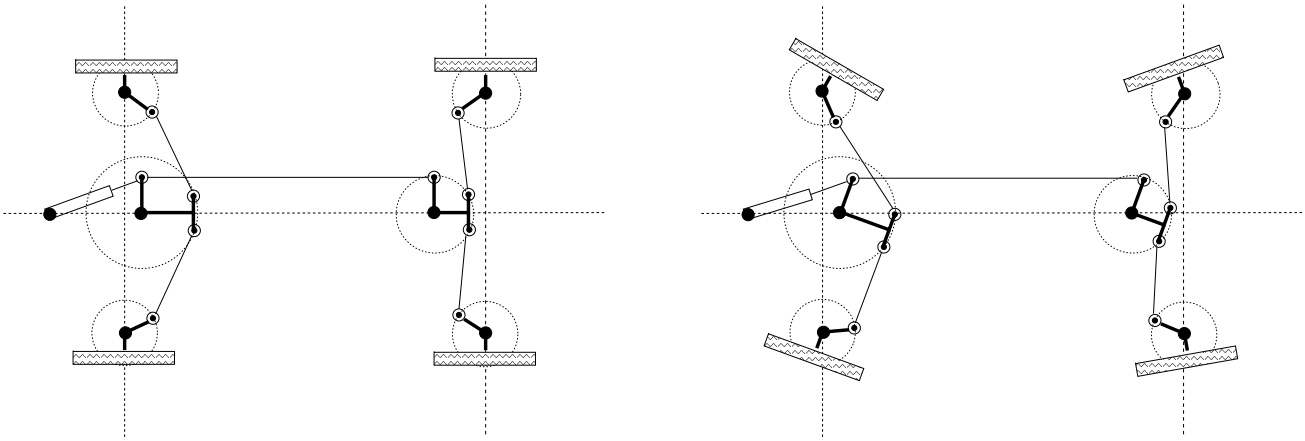


FIG. 4.4: *Mécanisme de direction du CyCab.*

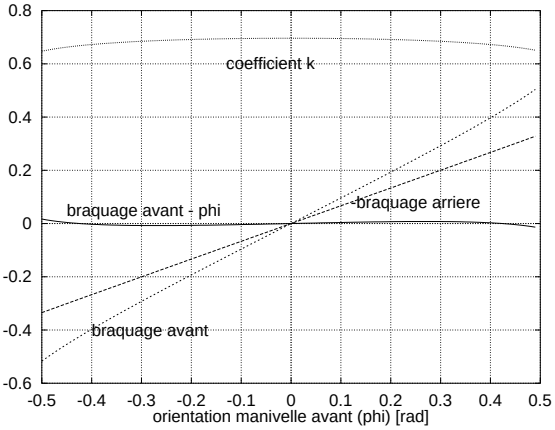


FIG. 4.5: *Evolution des braquages et relation entre ces derniers en fonction de ϕ.*

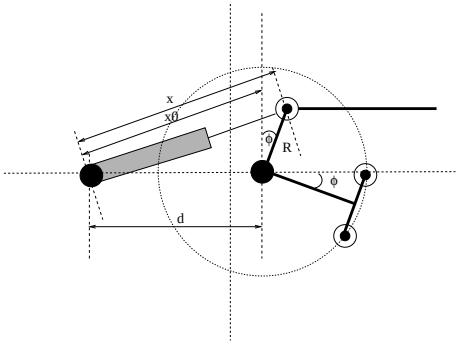


FIG. 4.6: *Détail du mécanisme de direction sur l'essieu avant.*

Relation entre la longueur du vérin hydraulique et l'angle ϕ

D'après les notations de la figure 4.6, on peut écrire:

$$\begin{aligned} x^2 &= (d - R\ sin(\phi))^2 + R^2\ cos^2(\phi) \\ &= d^2 + R^2\ sin^2(\phi) - 2dR\ sin(\phi) + R^2\ cos^2(\phi) \\ &= d^2 + R^2(sin^2(\phi) + cos^2(\phi)) - 2dR\ sin(\phi) \\ &= d^2 + R^2 - 2dR\ sin(\phi) \\ &= x_0^2 - 2dR\ sin(\phi) \end{aligned} \tag{4.23}$$

On en déduit la valeur de ϕ:

$$\phi = \arcsin\left(\frac{x_0^2 - x^2}{2dR}\right) \tag{4.24}$$

4.2 Calibration du CYCAB

Calibration du 3 décembre 1998, avec le CYCAB Rhône-Alpes, muni de sa coque inférieure.

4.2.1 Calibration de la translation

Le but est ici de déterminer le rapport top_codeurs sur avance linéraire des roues. Pour cela, un laser utilisé dans les présentations est fixé sur la coque, pointant vers le sol de manière perpendiculaire.

La calibration consiste en:

- mettre le CYCAB en mode joystick,
- marquer sur le sol, l'impact du laser et lire par logiciel la valeur des quatres codeurs,
- avancer tout droit sur une certaine distance (environ un mètre),

- marquer à nouveau l'impact du laser,
- mesurer la distance entre les deux marques et lire par logiciel la valeur des quatres codeurs.

Le déplacement du CYCAB n'est pas rectiligne, pour des raisons d'imprécisions multiples (donnée joystick, parallélisme des roues,...), il semble cependant raisonnable de penser que les quatres roues aient parcouru sensiblement la même distance. Pour cette calibration, une hypothèse d'une variation de moins de 0.5 % est retenue.

Les données mécaniques sont:

- codeurs, 2048 pas par tours,
- réduction moteur/roue de 20,
- diamètre D_r de la roue d'environ 40 centimètres.

La résolution codeur et le rapport de réduction nous donne le nombre de top par tour de roue, 40960. Le diamètre de la roue n'est qu'indicatif, en effet, ce diamètre dépend de la pression de gonflage, de plus, la valeur qui nous intéresse est la distance entre le centre de la roue (centre de rotation) et le sol, cette valeur dépend de la pression de gonflage et de la charge sur chaque roue.

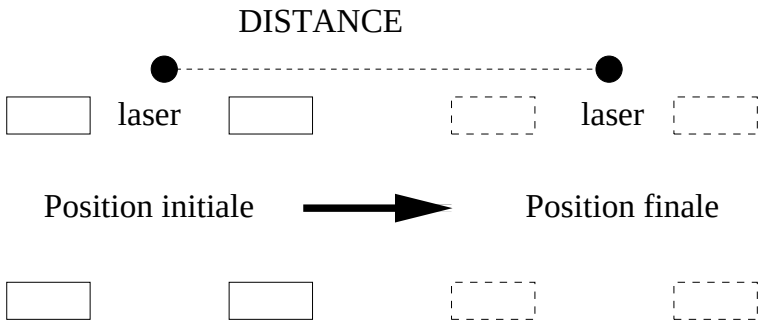


FIG. 4.7: *Schéma de la manip de calibration de la traction.*

DISTANCE MESUREE (mm)	CODEUR ROUE AVG (top)	CODEUR ROUE AVD (top)	CODEUR ROUE ARG (top)	CODEUR ROUE ARD (top)	Moyenne
925	31814	32968	32100	33725	32651.75
1	34.39	35.64	34.70	36.45	35.30

- première méthode avec les données théoriques:
  $top/mm = Top/(\Pi * D_r) = 40960/(3.14 * 400) = 32.59$

– seconde méthode expérimentale:
$top/mm = Moyenne\_Top/Distance = 32651.75/925 = 35.30$

Soit 8.3 % de ((35.30 - 32.59) / 32.59 = 0.083) de différence.

Il est à noter que les différences de lecture codeur sur les roues sont assez importantes. Si cela était dû à une distance parcourue différente, elle serait dans le cas extrême de l'ordre de (33725 - 31814) / 35.3 = 54 millimètres, ce qui n'est pas envisageable.

Ces variations sont plutôt à mettre à l'actif des distances roues sol. Si on prend l'hypothèse que toutes les roues ont roulé sur la même distance, et à partir de la fomule suivante:

$$distance = roue\_sol * 2 * \Pi * Top\_mesure/Top\_par\_tour.$$

On obtient alors pour chaque roue une distance roue_sol (rayon):

roue	AVG	AVD	ARG	ARD	Moyenne
Distance Roue-sol	189.54	182.90	187.85	178.80	184.77

4.2.2 Calibration de la direction

Le but est ici de déterminer le rapport potentiomètre sur angle de rotation de la direction. L'approche est purement expérimentale et ne repose sur aucune donnée géométrique de la mécanique.

La partie précédente sur la cinématique donne une version analytique de l'angle de direction en fonction de l' allongement du vérin, mais il manque la relation entre l'allongement du vérin et la valeur du potentiomètre (qui doit être linéraire), de plus les jeux ne sont pas pris en compte. Une confrontation des deux approches a été faite mais n'est pas présentée ici, néanmoins les résultats obtenus sont proches.

Pour cette calibration, un laser de présentation est fixé sur la roue avant gauche (puis avant droite), pointé vers un mur horizontalement.

La calibration ne sera pas précise car plusieurs paramètres n'ont pas été pris en compte:

– la géométrie de JANTEAU implique un déplacement de la roue (avant ou arrière, ce qui modifie la distance mur-laser,

– les jeux mécaniques de la direction n'ont pas été quantifiés.

La géométrie de JANTEAU, qui implique que la roue intérieure vire plus que la roue extérieure devrait faire apparaitre deux zones sur les données expérimentales.

Une autre conséquence est que la direction recherchée sera celle de la roue virtuelle décrite dans la cinématique (moyenne des deux angles des roues).

Schéma de la manip.

Le scénario de la calibration est:

– mettre l'asservissement de la direction,

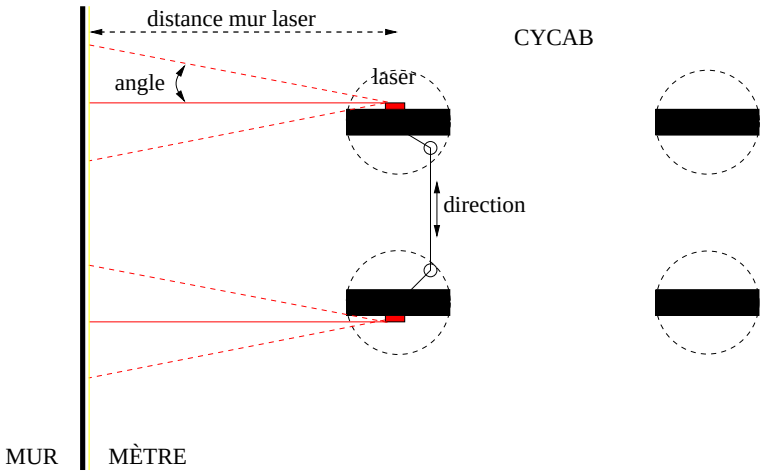


FIG. 4.8: *Schéma de la manip de calibration de la direction.*

- modifier la consigne de direction par logiciel,
- mesurer l'impact du point laser sur le mur à l'aide du mètre et lire le potentiomètre par logiciel.

Le mur se trouve à 233.5 cm du laser. La distance entre la position du laser sur la roue gauche et du laser sur la roue droite est de 122.0 cms (donnée non utilisée pas la suite).

Le tableau 4.9 donne les mesures acquises:

On approxime ces données par deux droites dont le point de coupure est (90.5, -49.0). Soit par un calcul de droite par les moindres carrés,

- une droite de coefficient directeur (0.222, 0.975) i.e. pente de 0.228) et d'erreur (somme des carrés des erreurs) de 83.0.
- une droite de coefficient directeur (0.172, 0.985 i.e. pente de 0.175) et d'erreur (somme des carrés des erreurs) de 31.0.

On approxime ces données par deux droites dont le point de coupure est (205.0, -39.0). Soit par un calcul de droite par les moindres carrés,

- une droite de coefficient directeur (0.183, 0.983 i.e. pente de 0.186) et d'erreur (somme des carrés des erreurs) de 4.75.
- une droite de coefficient directeur (0.229, 0.974 i.e. pente de 0.235) et d'erreur (somme des carrés des erreurs) de 12.0.

Mesure point laser (cm)	Potentiomètre (convertisseur)
10.0	-415.0
20.0	-381.0
33.5	-329.0
50.5	-269.0
62.5	-215.0
73.0	-159.0
86.0	-107.0
90.5	-49.0
97.5	-52.0
101.5	3.0
109.5	7.0
111.5	25.0
118.5	75.0
127.5	124.0

FIG. 4.9: *Tableau des mesures pour la roue gauche*

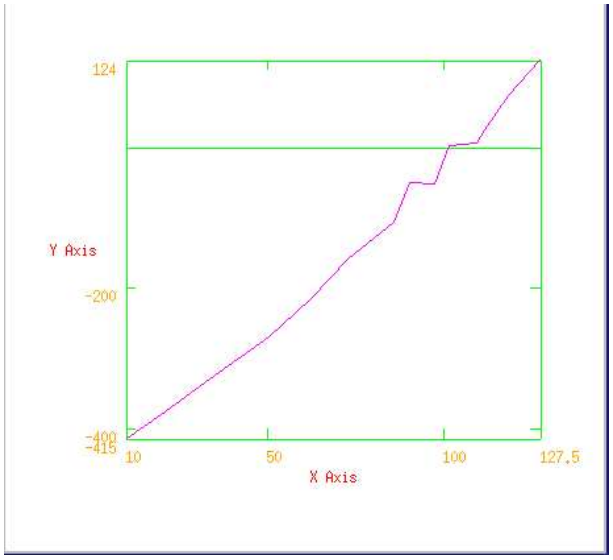


FIG. 4.10: *Plot des données (angle de la roue gauche, valeur du potentiomètre).*

MESURE	POTENTIOMETRE
157.5	-303.0
166.0	-255.0
174.0	-211.0
176.0	-204.0
184.0	-159.0
190.0	-130.0
196.5	-97.0
201.0	-75.0
205.0	-39.0
212.0	-21.0
223.5	34.0
241.5	103.0
244.5	123.0
252.0	155.0
257.0	169.0
261.0	201.0

FIG. 4.11: *Tableau des mesures pour la roue droite*

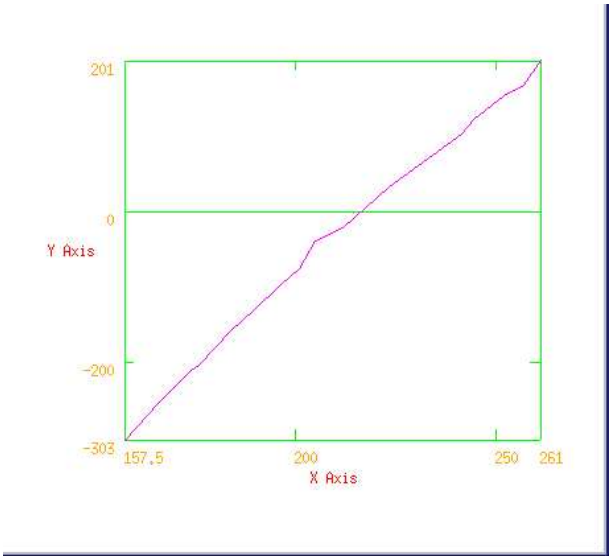


FIG. 4.12: *Plot des données (angle de la roue droite, valeur du potentiomètre).*

En posant:

- M une mesure,
- P la valeur du potentiomètre correspondant à cette mesure,
- M_0 la mesure initiale,
- P_0 la valeur du potentiomètre correspondant à cette mesure initiale,
- α l'angle de direction,
- D_{ml} la distance mur/laser.

On peut en déduire la loi potentiomètre angle de direction α suivante:

$$tan(\alpha) = \frac{M - M_0}{D_{ml}}$$

soit

$$tan(\alpha) = \frac{pente * (P - P_0)}{D_{ml}}$$

soit

$$\alpha = Arctan(\frac{pente * (P - P_0)}{D_{ml}}$$

soit

$$\alpha = Arctan(0.00206 * (P + 40.0)/2.335)$$

en mettant en mètre et en faisant la moyennes des pentes, et en prenant -40 pour l'offset du potentiomètre, on obtient le tableau 4.13

4.3 Orccad

4.3.1 Introduction à ORCCAD

ORCCAD [1] [8] est un environnement logiciel permettant de concevoir et de mettre en œuvre le contrôle et la commande d'un système robotique complexe. Il permet également la spécification et la validation des missions à réaliser par ce système.

ORCCAD est principalement destiné aux applications temps réel critiques en robotique, dans lesquelles les aspects relevant de l'automatique (les asservissements, les commandes) sont amenés à interagir étroitement avec ceux manipulant des événements discrets. De tels systèmes sont souvent qualifiés d'hybrides.

Potentiomètre	angle(deg)
-480	-21.22
-440	-19.44
-400	-17.62
-360	-15.77
-320	-13.88
-280	-11.96
-240	-10.01
-200	-8.03
-160	-6.04
-120	-4.04
-80	-2.02
-40	0.00
0	2.02
40	4.04
80	6.04
120	8.03
160	10.01
200	11.96
240	13.88
280	15.77
320	17.62
360	19.44
400	21.22
440	22.95
480	24.64

FIG. 4.13: *Tableau de correspondance Angle/valeur potentiometre*

Dans cette classe d'applications, ORCCAD s'adresse particulièrement aux systèmes présentant une forte interaction avec l'environnement par le biais de nombreux capteurs et actionneurs. Le contrôle/commande de ces systèmes est souvent embarqué, et le caractère critique de l'application apparaît dans le coût extraordinairement élevé attaché à une défaillance: l'impossibilité ou la difficulté d'intervention sur un sous-marin autonome à longue portée, sur un engin intervenant après un incident technologique majeur ou sur un véhicule planétaire rendent impératif la minimisation du risque de non réalisation de la mission. A cet effet, ORCCAD offre sûreté de programmation et possibilités de validation par simulation extensive ou vérification formelle.

A titre d'exemple, les applications aujourd'hui traitées en laboratoire avec ORCCAD vont du bras manipulateur au véhicule électrique [2] automatique en passant par un robot sous-marin et un robot mobile terrestre avec une caméra.

Une application Orccad est découpée hiérarchiquement en trois entités:

- le Module qui est le grain minimal de l'utilisateur. Il permet de décrire:

  du code de calcul (module de classe algorithmique),

  des interfaces avec les ressources physiques tel que robot ou capteur (module de classe Ressource Physique),

  des événements pour gérer la sécurité de l'entité de niveau supérieur (module de classe automate),

- la Tâche-Robot (TR) qui décrit une action robotique élémentaire qui contient un asservissement et la gestion des événements pour assurer l'intégrité de l'action (erreurs typées, début et fin de l'action). Cette action est spécifiée en interconnectant un certain nombre de Module.

- la Procédure-Robot (PrR) qui décrit en Estérel (ou Maestro) la composition de TRs. Cette composition permet de spécifier l'application que doit réaliser le robot.

4.3.2 La Ressource Physique

La ressource physique assure le passage entre la vue ORCCAD et le Cycab proprement dit, on connecte ses différents ports qui appellent les fonctions pilotant les moteurs ou les capteurs.

C'est le point de départ des applications Cycab développées sous ORCCAD. Une application comportera donc cette ressource physique, les ressources physiques des capteurs utilisés, des modules algorithmiques, et un faisceau de connections.

4.3.3 Exemple d'une Tâche Robot

On connecte via des ports d'entrées-sorties, la ressource physique et des modules algorithmiques, contenant la partie intelligente de l'action. Ici la conduite via le joystick du

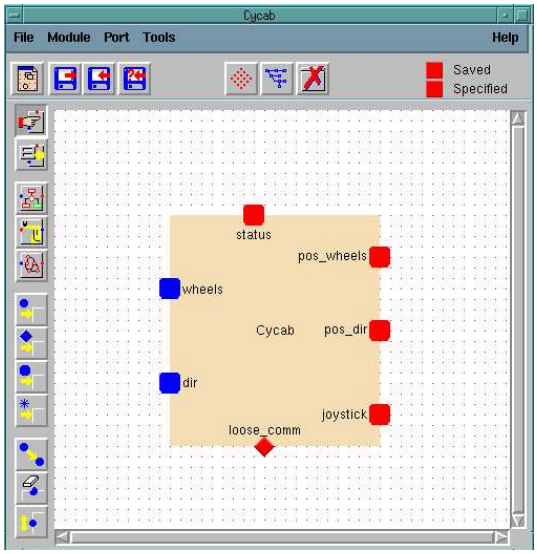


FIG. 4.14: *Représentation de la ressource physique Cycab.*

véhicule. On retrouve la ressource physique du Cycab, le module *SimpleAlgo* qui transforme les données joystick en commandes moteurs, un module *Secure* qui, en fonction du status (mot représentant l'état du véhicule) envoie des messages au module *SimpleAtr* qui représente la machine d'état de la tâche robot.

La Procédure Robot [1] n'est pas décrite ici car élémentaire puisque ne contenant qu'une seule tâche robot.

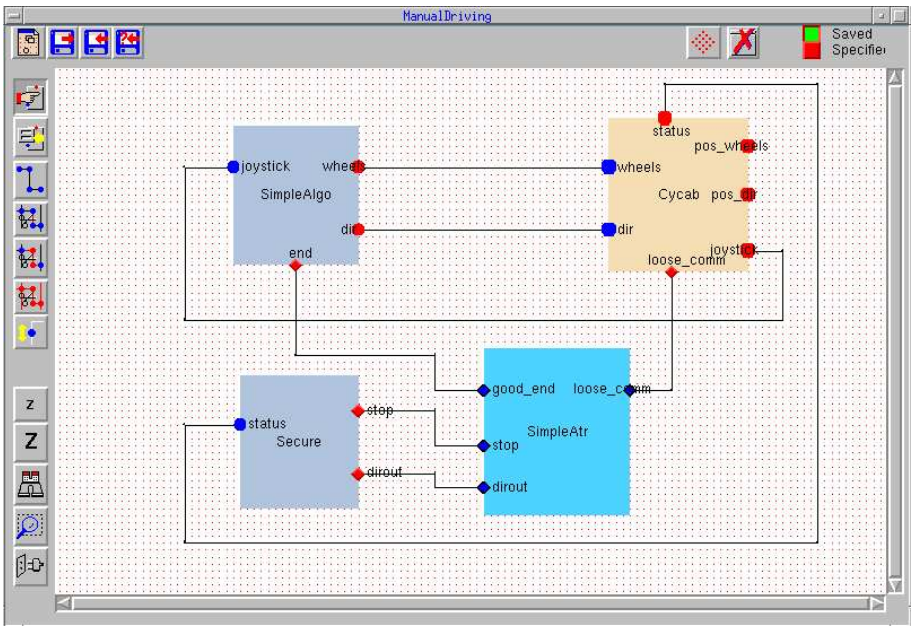


FIG. 4.15: *Représentation de la tâche de conduite par le joystick.*

Chapitre 5

Conclusion

Ce document représente la somme de nos connaissances sur le Cycab à la date de Février 1999. Pour que le véhicule soit opérationnel pour les expérimentations des projets de recherche de l'INRIA Rhône-Alpes, nous devons désormais l'équiper de capteurs extéroceptifs.

En effet, l'automatisation de ces véhicules nécessite des capteurs fournissant des informations sur leur environnement. Les capteurs envisagés ou en cours de montage sont:

- les ultra-sons utilisés pour l'évitement d'obstacles proches et le parking automatisé,
- une ou plusieurs caméras, pour la téléopération ou le recalage du véhicule dans son environnement,
- un GPS (Global Positionning System) couplé aux caméras pour le recalage,
- une caméra linéaire Infra-rouge couplé à une balise infra-rouge pour un suivi automatisé de flotte de véhicules.

Ce véritable travail de "reverse-engineering" pour prendre en main le système Cycab lui même réalisé à l'INRIA peut laisser dubitatif. En tout cas, il ne peut que nous conforter dans l'idée que nous nous faisons de l'utilité d'assurer un support expérimental pérenne et organisé.

Bref du boulot sur la planche, mais aussi une forte motivation pour obtenir un formidable outil de travail, afin de valider toute une série de travaux théoriques menés depuis plusieurs années sur l'automatisation des véhicules.

Bibliographie

[1] D. Simon, B. Espiau, K. Kapellos, R. Pissard-Gibollet: The Orccad Architecture, The Orccad Team, Robotics Research, Special issues on Integrated Architectures for Robot Control and Programming, vol 17, no 4, pp 338-359, April 1998.

[2] S. Abdou, M. Parent and B. Espiau: Spécification et Implémentation Logicielles pour des Véhicules Automatiques , Proc. of Real-Time Systems and Embedded Systems, Paris, 1996. (postscript)

[3] Leszek Lisowski: Documentation Technique, Cycab version 3 noeuds, rapport interne Moyens Robotiques, INRIA Rhône-Alpes, Janvier 1998.

[4] http://www.inrialpes.fr/iramr/: le service robotique de l'INRIA Rhône-Alpes.

[5] http://www-lara.inria.fr/: programme de recherche et développement sur LA Route Automatisée.

[6] http://www.can-cia.de/: le CAN (Controller Area Network).

[7] http://www.robosoft.fr/: l'industriel choisit pour concevoir et réaliser le sucesseur du Cycab décrit dans ce document.

[8] http://www.inrialpes.fr/iramr/Orccad: le logiciel de contrôle-commande ORCCAD.

[9] http://www-rocq.inria.fr/syndex: le logiciel Syndex pour la répartition d'applications temps-réels embarqués.

[10] http://www.inrialpes.fr/bip/people/girault/Projets/Incitative

[11] http://www.mcg.mot.com/WebOS/omf/GSS/MCG/products/overview.html?[prodid=MVME162: carte MVME162 de MOTOROLA

[12] http://www.tews-datentechnik.com/tip810.htm: le module IP-CAN de TEWS.

[13] http://www.breezecom.com/Products/brznprd.htm: le lien éthernet sans fil de BREEZECOM.

[14] http://www.wrs.com/products/html/vxworks.html: Real Time Operating System de WindRiver Systems.

Unit´e de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unit´e de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unit´e de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unit´e de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unit´e de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399