HELMO - SAINT-LAURENT
quai Mativa 38 - 4020 LIEGE
04/349.50.51 - Fax 04/341.13.70
saint-laurent@helmo.be

INTÉGRATION DE CAPTEURS DIGITAUX DANS UNE STATION MÉTÉOROLOGIQUE

Tome 1: Etude du problème

Pierre Muller

Mémoire de fin d'études présenté en vue de l'obtention du diplôme de Bachelier en informatique et systèmes

Année académique 2008-2009

Remerciements.

Je tiens à remercier toutes les personnes qui ont contribué à la réalisation de mon travail de fin d'études et plus particulièrement :

Didier Lodomez, gérant de M&D Electronics, mon maître de stage externe ;

Michele Lodomez;

Jehan Herbrand;

Hocine Daoudi;

Monsieur Delmot, mon parrain à l'Institut Saint-Laurent ;

Ma famille.

Table des matières

1 .Présentation de M&D Electronics.

M&D Electronics SPRL est une PME créée en 1993 à Stavelot (dans l'Est de la Belgique). Il s'agit d'un bureau d'études et de dessins permettant de créer des cartes électroniques pour les entreprises. L'objet de cette entreprise couvre les trois principales catégories d'activités suivantes :

1. l'étude et le développement de prototypes ;
2. la fabrication des unités de gestion et de programmation ;
3. la réorganisation des services afin de mettre à jour certaines applications, en remplaçant les systèmes obsolètes par des nouveaux composants.

Cette société veille à adapter le dessin industriel de cartes électroniques aux besoins de chaque client. Ce dernier sollicite la société M&D pour assouvir un besoin spécifique et bien précis. M&D étudie alors ce besoin et établit une liste de tous les composants nécessaires, les projets de cartes électroniques, pour aboutir à un produit optimal.

Cette analyse réalisée, M&D fait appel à deux sociétés externes. L'une d'entre elle, sur base des dessins, fabriquera les cartes électroniques vierges. A son tour, la seconde entreprise se chargera de placer les composants sur ces cartes.
Concernant l'achat des composants électroniques, M&D utilise autant que faire se peut, des composants basiques et bon marché. Ces composants seront par la suite envoyés à la société qui s'occupe elle-même du montage afin de minimiser les coûts. (Ce type d'externalisation est aussi appelé « contrat de fabrication »).

Une fois la phase de fabrication externe terminée, les microcontrôleurs sont programmés (avec des programmes développés dans la société) et les cartes électroniques sont assemblées mécaniquement. Ensuite, vient une phase de test. En effet, toutes les cartes, sans exception, sont testées une à une afin de garantir au client une qualité optimale. A sa sortie de chez M&D, le fonctionnement de la carte est optimal! Ces dernières sont ensuite livrées directement aux clients.

M&D emploie actuellement 4 personnes :

- Didier Lodomez : fondateur de la société. Il s'occupe principalement de la recherche et du développement ;
- Michèle Lodomez : épouse du fondateur. Elle s'occupe de toutes les tâches administratives comme la gestion de fabrication et de livraison, l'achat du matériel, les contacts avec les clients;
- Jehan Herbrand : s'occupe de la recherche et du développement ;
- Houcine Daoudi : s'occupe du montage mécanique et du test de chaque carte.

Même si ces personnes accomplissent des tâches bien définies, elles apportent à l'entreprise une plus-value résultant de leur polyvalence ainsi que de leur souplesse.

La taille de l'entreprise, sa localisation et son caractère familial font partie de la culture organisationnelle et sont souvent considérés comme un avantage pour les clients. En effet, sa petite taille et son nombre relativement faible de salariés permettent aux clients de connaître toutes les ressources humaines et les relations étroites qui les lient. M&D entretient une excellente relation avec ses clients en favorisant un contact direct et régulier.

Les secteurs d'activités de M&D sont très variés et principalement orientés vers l'industrie :

- carte de contrôle pour trancheuse à pain ;
- carte de contrôle de lave-vaisselle industriels ;
- automatisme de commandes de machines textiles ;
- système de contrôle d'éclairage pour bâtiments industriels, hôtels de luxe,… (application domotique);
- équipement de contrôle de foyers à gaz ;
- interface de commande d'électrovanne pour appareillage médical ;
- système de gestion de compteurs d'eau par carte à puce.

2. Le projet.

2.1 But du projet.

La mission de ce stage au sein de l'entreprise M&D Electronics consiste à développer une station météorologique permettant de réaliser différentes mesures :

- température extérieure ;
- température du sol ;
- taux d'humidité dans l'air ;
- direction du vent ;
- vitesse du vent ;
- Pression atmosphérique ;
- quantité de pluie tombée ;
- taux d'ensoleillement.

La mesure de ces variables est en effet nécessaire aux activités d'un club de parapentistes qui, à l'heure actuelle, doit se contenter de techniques peu développées pour obtenir ces informations (un simple thermomètre, une manche à air,…). Le but du projet est donc de leur faciliter la tâche en améliorant l'efficacité et la précision des mesures.

La station est constituée de trois modules interagissant entre eux :

1. le premier module (MOD_VENT[1]) mesure les caractéristiques du vent et est placé sur un mât à une hauteur de plus ou moins 10m ;
2. le second module (MOD_CAPT[2]) effectue l'ensemble des autres mesures et est placé à 1.5m du sol (sauf le capteur de température du sol qui lui est placé au niveau du sol) ;
3. le troisième et principal module (UBC[3] + UBC_IO[4] qui contrôlent les deux modules précédents) est placé dans un endroit où il est possible de disposer d'une prise électrique (230V).

La station est équipée d'un écran LCD (deux couleurs : noir sur fond vert + rétro-éclairage) permettant d'afficher en temps réel les mesures effectuées (aucune prévision, juste de l'information). De plus, elle est équipée d'un module GSM qui permet d'établir une communication par SMS. Ce principe permet à un membre du club d'envoyer un message (SMS) à la station qui lui répond en transmettant les mesures réalisées. De cette façon, le membre n'est pas obligé de se déplacer (certains viennent d'Anvers) pour finalement se rendre compte que les conditions climatiques ne sont pas adéquates pour prendre son envol.

Ma mission consistait donc à intégrer les différents capteurs à la station, à dessiner les cartes électroniques pouvant accueillir tous les composants (via le logiciel Eagle), à développer les différents programmes (via le logiciel MPLAB IDE) et à monter la station (soudage des composants sur les cartes, intégration mécanique des cartes dans la station).

[1] Abréviation de Module_vent
[2] Abréviation de Module_capteurs
[3] Abréviation de Unit Box Control
[4] Abréviation de Unit Box Control_ Input Output

2.2 Cahier de charge.

A mon arrivée chez M&D, certaines parties de la station étaient déjà existantes. En effet, le module UBC et UBC_IO étaient déjà conçus, ainsi que le module GSM et le capteur de température du sol (CTHI[5]) (au niveau des PCB[6]). Les programmes intégrés à ces cartes étaient opérationnels mais non encore adaptés au projet souhaité. Le module GSM et le module UBC communiquaient entre eux. Cependant, le format du SMS ne convenait pas. Le capteur de température du sol communiquait avec le module UBC sans toutefois traiter la donnée.

La base de la station était donc déjà conçue :

- communication entre le capteur de température et le module UBC;
- affichage des données (affichage basic) ;
- communication GSM avec une structure SMS non définie et erronée.

La première étape de mon travail consistait à me familiariser avec le matériel mis à disposition et à découvrir le logiciel MPLAB IDE (programme utilisé pour développer les programmes). Par la suite, la compréhension du programme du capteur de température du sol s'avérait nécessaire.
Ces différentes étapes m'ont permis de me rendre compte que les échanges d'informations entre les différents modules n'étaient pas appropriés. En effet, les différentes transmissions n'utilisaient pas les mêmes formes de trames (trames de tailles différentes) et ne contenaient pas les mêmes informations (entête de trame différente, contenu différent, …). Il a donc fallu « standardiser » les échanges pour faciliter leur compréhension et leur mise en œuvre. Cela a pris deux semaines pour aboutir à un résultat satisfaisant.

Une fois cette « standardisation » mise en place, je me suis penché sur le capteur de température du sol (CTHI). Celui-ci mesurait et envoyait déjà une valeur mais cette dernière n'était pas correcte. Après modification du programme, le capteur envoyait alors deux valeurs : la partie entière et la partie décimale de la température. Il est à souligner que les nombres à virgules sont exclus des programmes (le microcontrôleur gère difficilement le format « float » (format « virgule flottante »)).
Ensuite, j'ai monté les différents prototypes afin d'observer le fonctionnement des composants (du point de vue électrique) et la façon dont ils interagissaient entre eux. Le point essentiel de cette étape était d'apprendre à souder à l'étain.

J'ai par la suite remanié le programme du capteur de la girouette (MOD_VENT) afin de l'adapter à mon application. Le programme était utilisé pour un codeur 1024 positions. Il a fallu définir des plages de variation de position.
Un problème mécanique s'est alors présenté : l'aimant rotatif servant à indiquer la position était trop éloigné du circuit. Ce problème engendrait un défaut au niveau du capteur qui ne subissait pas correctement le champ magnétique créé par l'aimant. Une fois le disfonctionnement résolu, j'ai monté les autres prototypes pour effectuer des tests complémentaires sur chacun d'eux.

A la suite de ces différentes étapes, j'ai entamé une recherche de matériel pour les autres capteurs (capteurs de lumière, de température/humidité, de pression atmosphérique + convertisseur) (recherche

[5] Le module CTHI est un module disposant d'un PIC et d'un capteur de température. Ce module a été utilisé au début du projet afin de mettre en place les communications CAN. Par la suite, ce capteur a été remplacé par le capteur ION_CAPT et le module MOD_CAPT qui effectue tous les transferts de données. Ces différents modules sont expliqués plus loin.
[6] Le PCB représente la carte électronique au niveau hardware.

des capteurs adéquats, compatibilité des différents composants entre eux,…) afin d'en dégager les principaux avantages pour au final, les sélectionner en fonction de leur prix. Il apparaît que les composants les plus onéreux ne sont pas toujours les plus intéressants.

Cette recherche terminée, j'ai commencé le dessin de la carte électronique qui allait accueillir ces différents composants (capteurs de température, capteur d'humidité, pluviomètre, capteur de lumière (taux d'ensoleillement), capteur de pression atmosphérique ainsi que son convertisseur) ; le module MOD_CAPT.
Il a alors fallu manipuler le second programme: « le logiciel Eagle ».
Avant de lancer la carte à la production, certains capteurs ont été commandés afin d'y réaliser des expériences et de pouvoir, si nécessaire, apporter des modifications. Prenons pour exemple le cas d'un capteur de lumière choisi qui avait une plage de variation beaucoup trop mince. Cet inconvénient a entraîné des recherches supplémentaires pour trouver le capteur adéquat.
De plus, les capteurs analogiques (capteurs de pression atmosphérique et de lumière) demandent une tension d'alimentation très précise, il a fallu intégrer un régulateur de tension de référence pour obtenir des mesures sans erreur.

La carte étant lancée à la production, j'ai pu entamer le développement des différents programmes indispensables à son fonctionnement. Concernant le capteur relatif à la température du sol, j'ai repris le programme déjà développé précédemment (CTHI) pour l'intégrer dans le nouveau software. Du point de vue du capteur de température extérieure et d'humidité, j'ai développé un nouveau programme (les deux mesures sont effectuées par le même composant).
Une fois ce capteur terminé, je me suis penché sur le capteur de luminosité pour finalement m'attarder sur le capteur de pression atmosphérique.
Ce dernier fut assez difficile à mettre en place car quelques difficultés sont apparues quant à la communication entre le PIC et le convertisseur analogique/digital (le convertisseur ne répondait pas aux requêtes faites par le PIC → aucune mesure de pression n'était possible !)

Ce problème une fois résolu, j'ai commencé le développement du programme pour le capteur de vitesse du vent (anémomètre). J'ai dû modifier la carte électronique (MOD_VENT) pour qu'elle puisse accueillir cette mesure car aucune entrée n'était disponible sur le PCB (PCB existant pour le codeur magnétique utilisé dans une autre application).

J'ai par ailleurs dessiné deux autres cartes : la première allant servir au capteur d'humidité et de température extérieure (TH_SENSOR) et la seconde au capteur de lumière (LIGHT_SENSOR).
En effet, le capteur de lumière est un composant assez petit (1.5 x 2 mm) et fragile. Il est donc plus facile de le placer sur un PCB et ainsi le positionner à l'endroit souhaité sans l'abîmer.

Lorsque les prototypes des cartes dessinées sont arrivés (MOD_CAPT + TH_SENSOR + LIGHT_SENSOR), je les ai montés afin de les tester. Ces tests m'ont permis de voir si les capteurs analogiques fonctionnaient correctement, si aucune erreur de dessin n'avait été commise (trace mal placée, perturbations d'un composant sur un autre, …) et si le programme développé fonctionnait. Ces étapes sont importantes car si un capteur utilise un protocole de communication développé dans le programme (un SPI soft par exemple), il se pourrait que celui-ci ne fonctionne pas, alors qu'au point de vue électrique, tout est correct. De même, le module traitant la donnée (le module UBC) pourrait afficher de mauvaises indications si le calcul effectué sur la mesure est erroné.

Les capteurs fonctionnant tous correctement, je me suis penché sur l'affichage des données. L'afficheur LCD indique chaque mesure une à une (bouclage) et de façon dynamique. Il est cependant possible de faire un défilement manuel avec les boutons poussoirs.

J'ai ensuite adapté la communication GSM. Le module GSM fonctionnait déjà pour une autre application, différente de celle de la station météo. Le plus gros du travail était de faire communiquer le GSM : l'envoi des mesures devait être effectué après chaque demande. Or ce n'était pas le cas. Le GSM recevait bien le message mais le supprimait sans même y répondre.

Le développement de chaque programme était à cette étape approximativement terminé. J'ai donc assemblé toutes les cartes (dans leur boitier) afin de les placer dans un environnement réel. En effet, les conditions sont optimales dans un bureau mais la station doit fonctionner à l'extérieur, endroit où les conditions peuvent être bien plus rudes et différentes.

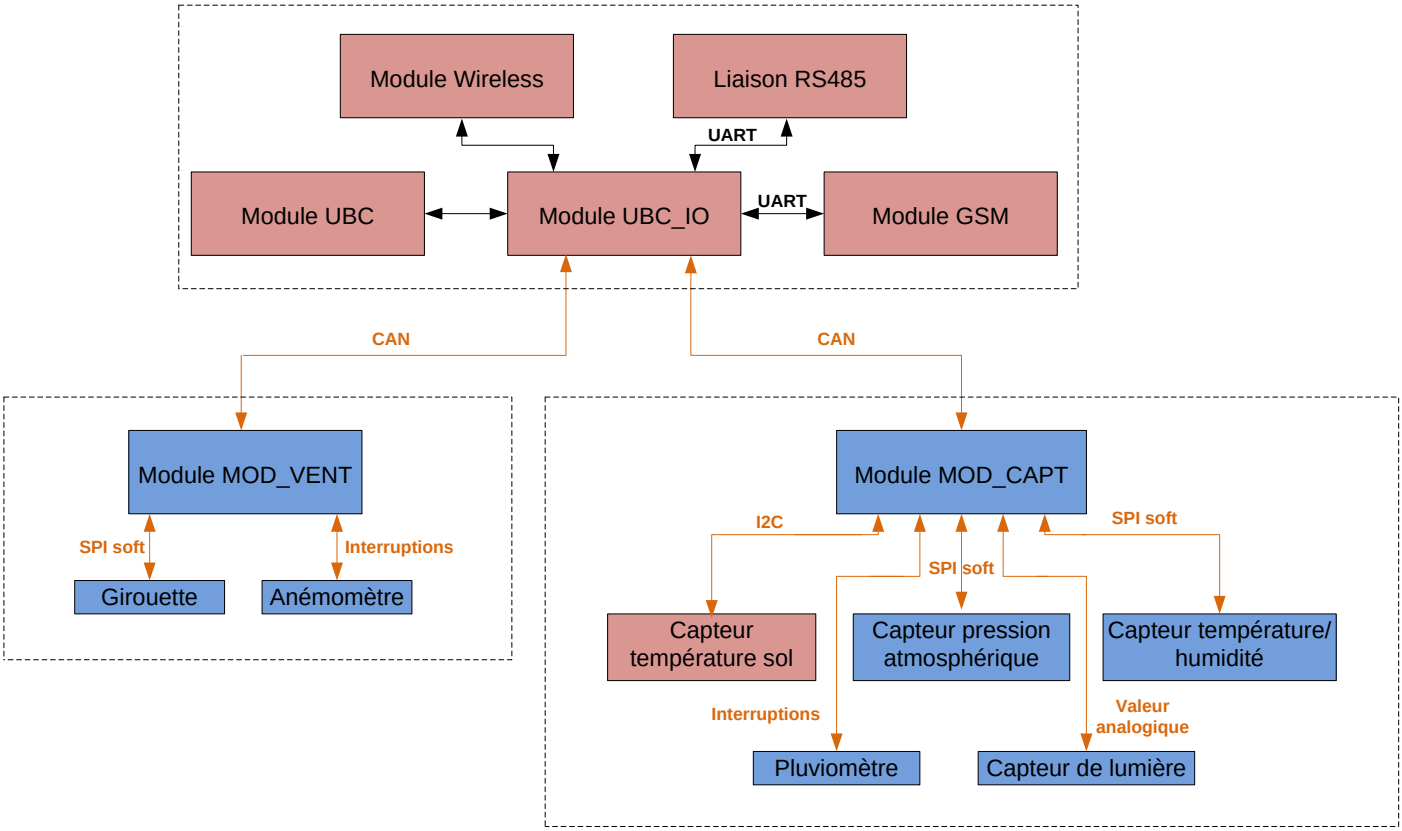
Ci-dessous, vous trouverez un tableau récapitulatif du travail entamé à mon arrivée et son évolution lors de mon stage :

	Niveau PCB	Niveau Programmes
Travail existant :	Dessin de la carte UBC Dessin de la carte UBC_IO Dessin du Module GSM Dessin du capteur température du sol CTHI (non utilisé dans la station finale) Dessin du module ION_CAPT	Programme de base de l'UBC comprenant une communication CAN entre le capteur CTHI et le module UBC
Travail accompli :	Dessin du module MOD_CAPT Dessin du module MOD_VENT Dessin du module TH_SENSOR Dessin du module LIGHT_SENSOR Montage des différents prototypes	Développement du programme MOD_CAPT Développement du programme MOD_VENT Développement du programme TH_SENSOR Développement du programme LIGHT_SENSOR Développement du programme UBC (communication GSM, affichage des données, communication avec tous les autres modules,…)
Problèmes rencontrés :	Mauvais choix du capteur de lumière Problème de proximité de l'aimant pour la girouette (mesure impossible)	Problème de communication entre le capteur de pression atmosphérique et le microcontrôleur

En conclusion, toutes les mesures sont effectuées correctement et les modules communiquent entre eux sans erreur. De plus, la communication GSM fonctionne parfaitement et sans perte d'information. L'affichage des données quant à lui est dynamique et fonctionnel.

En définitive, le projet qui m'a été confié a été correctement réalisé dans un délai imparti.

2.3 Schéma général de la station.



Module déjà dessiné, et programme que j'ai retravaillé!

Module que j'ai dessiné et programme que j'ai totalement développé!

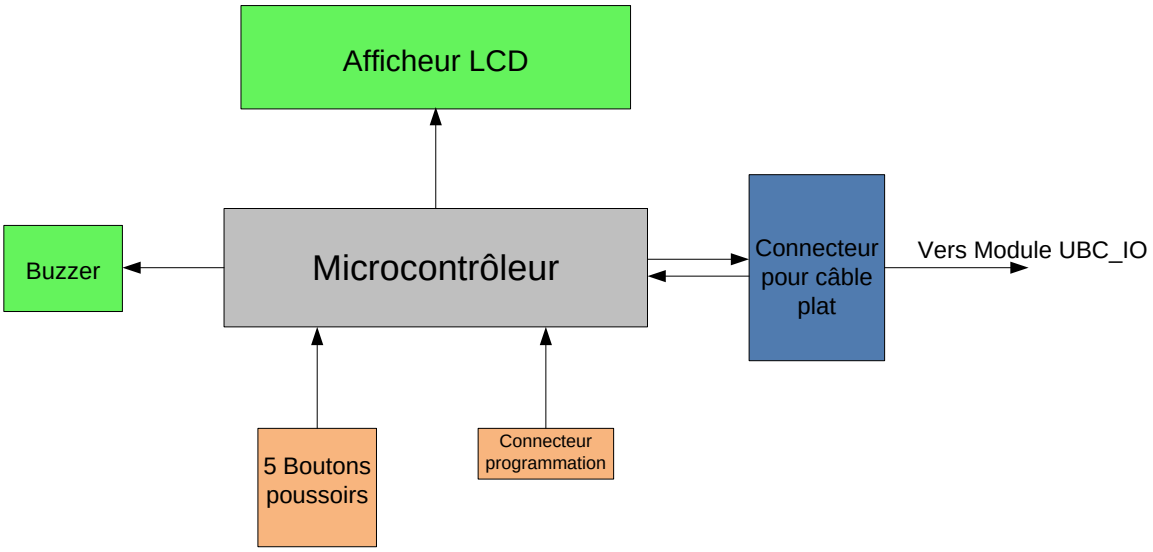
Communication déjà établie

Communication que j'ai développée.

3. Détails des différents modules de la station.

3.1 Le module UBC.

3.1.1 Schéma bloc de la carte.



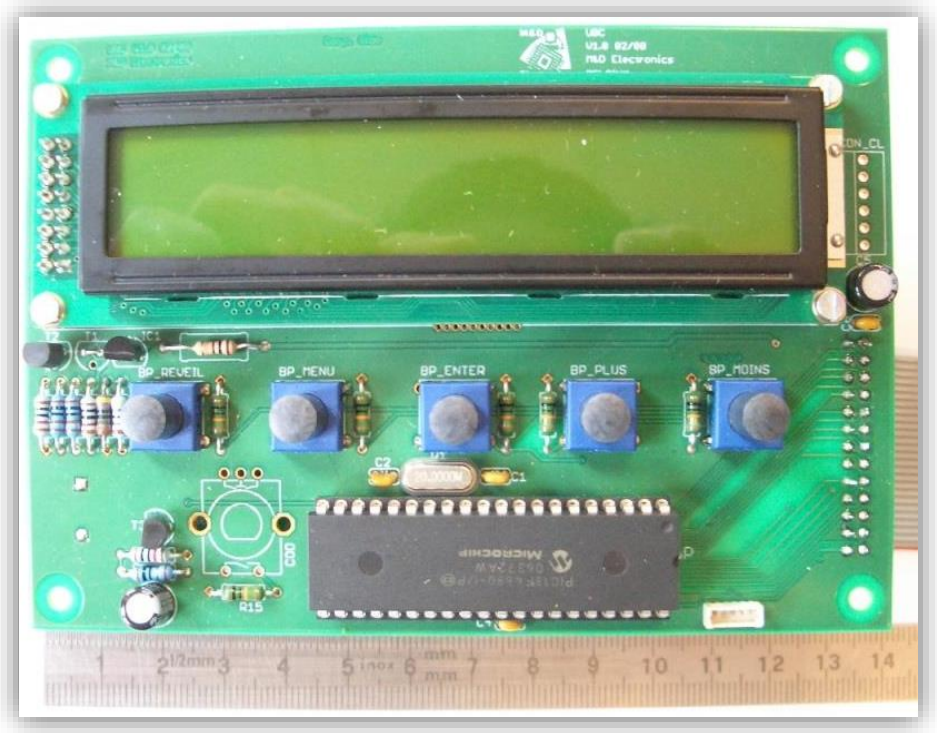


Figure 1 Module UBC hors du boitier

Figure 2 Module UBC dans son boitier

3.1.2 Rôle de la carte.

Différents équipements sont branchés sur la carte. Tout d'abord, nous avons le microcontrôleur, qui représente « l'intelligence » de la carte. Il s'agit du microcontrôleur PIC 18F4680 I/P de chez Microchip *(Voire détails plus loin)*. Nous avons ensuite un écran LCD *24 x 2*, cinq boutons poussoirs et un buzzer (alarme sonore).

Cette carte sert surtout d'interface de visualisation et de contrôle. En effet, les données peuvent être affichées sur l'écran et il est possible de naviguer dans les menus à l'aide des boutons poussoirs. Le Buzzer peut servir d'alarme ou de bip d'avertissement.

C'est cette carte qui envoie toutes les demandes CAN (bus expliqué plus loin) vers les autres modules (demande de mesures). Elle émet des trames CAN vers le module MOD_CAPT pour obtenir les mesures de températures, d'humidité, de pression, de lumière, de pluie. Elle émet également des trames CAN vers le module MOD_VENT pour obtenir les mesures de vitesse et de direction du vent.
Cette carte s'occupe également de la gestion de la communication GSM, RS485. C'est le « Master » de la station météo, un peu comme un « chef d'orchestre ».

Le schéma électrique général de la carte et le dessin de celle-ci se trouvent dans les annexes au page I et II.

3.1.3 Explication des différents composants.

3.1.3.1 Boutons poussoirs.

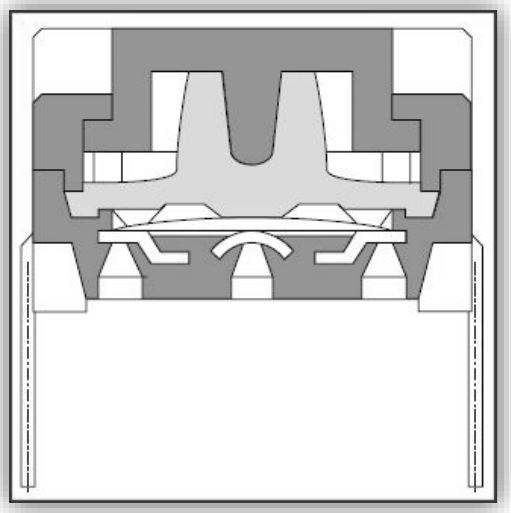


Figure 4 Coupe d'un bouton poussoir

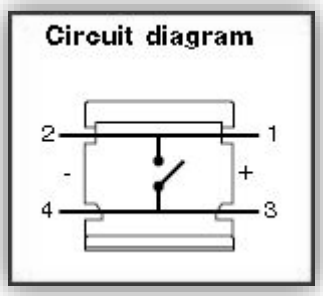


Figure 3 Schéma électrique d'un bouton

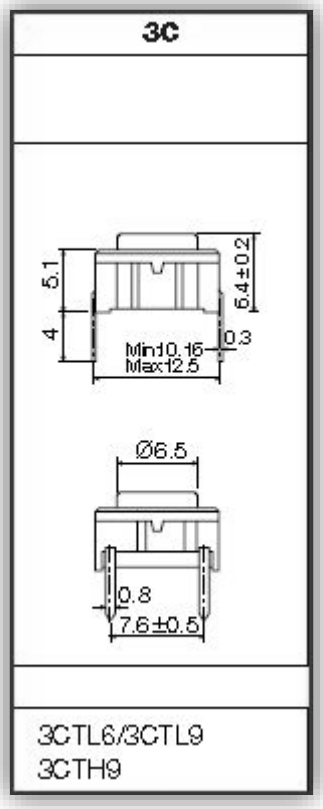


Figure 5 Dimensions d'un bouton

Les boutons poussoirs sont utilisés pour voyager dans les menus et effectuer d'autres fonctions (allumage du rétro-éclairage si on pousse sur l'un d'eux). La carte dispose de 5 boutons poussoirs :

- BP_Reveil : Pour sortir de la mise en veille.
- BP_Menu : Pour entrer/sortir dans le menu.
- BP_Enter : Pour entrer/sortir dans les sous-menus mais aussi pour valider un paramètre.
- BP_Plus et BP-Moins : Pour naviguer dans les sous menus.

3.1.3.2 Afficheur LCD.

L'afficheur LCD est un afficheur 2 lignes/24 colonnes. Il permet d'afficher tous les caractères du code ASCII (norme de codage des caractères en informatique). Ceux-ci sont projetés en noir sur fond vert. Il est possible d'allumer un rétro-éclairage (l'écran est allumé et peut être visible dans l'obscurité). Pour diminuer la consommation électrique de la station, le rétro-éclairage n'est enclenché que par une pression sur un des boutons poussoirs, et ce pendant une durée d'une minute. Les données sont envoyées à l'afficheur grâce à une liaison parallèle. Premièrement, 2 x 4 bits d'adresse sont envoyés. Deuxièmement, 2 x 4 bits de données sont envoyés.

Dans le programme développé, lors de la fonction d'initialisation de l'écran LCD, on signale à celui-ci qu'il fonctionnera en 2x4 bits et non en 1x8 bits car seulement 4 Pins du PIC sont utilisées pour cette communication.

L'afficheur dispose également d'un signal de sélection de registre (RS), un signal Read/Write (pour écrire une donnée ou pour lire une donnée affichée ou autre), un signal permettant d'activer ou non l'écran (Enable) et un signal permettant de régler le contraste.

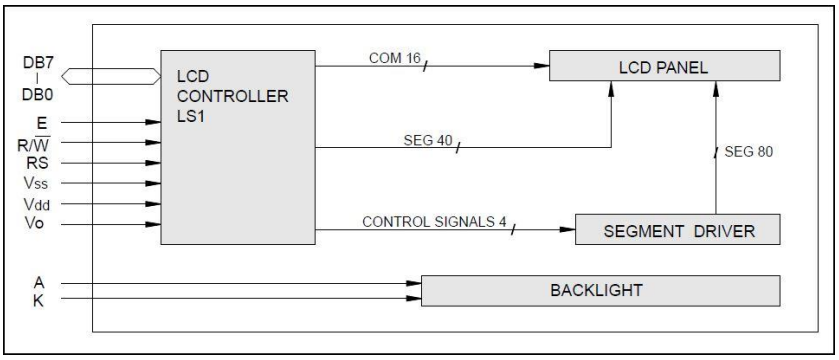


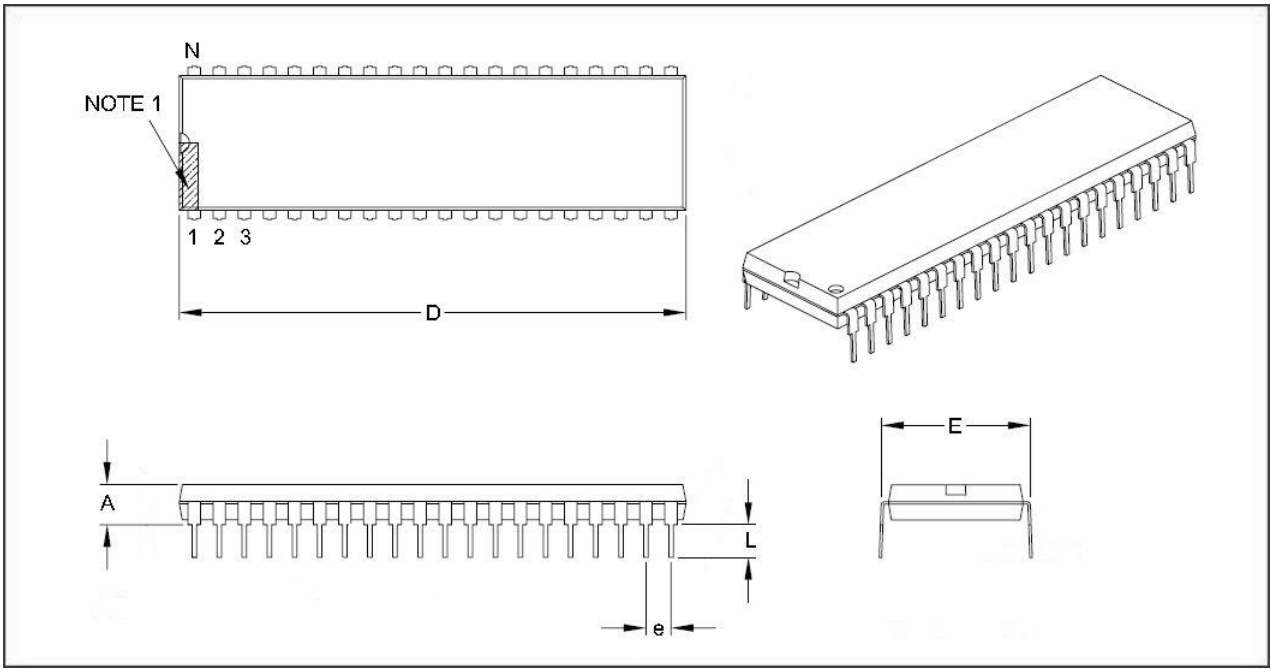
Figure 6 Schéma interne de l'écran

Symbol	Function
Vss	Power supply(GND)
Vdd	Power supply(+)
Vo	Contrast Adjust
RS	Register select signal
R/$\overline{W}$	Data read / write
E	Enable signal
DB0	Data bus line
DB1	Data bus line
DB2	Data bus line
DB3	Data bus line
DB4	Data bus line
DB5	Data bus line
DB6	Data bus line
DB7	Data bus line

Il faut se référer aux annexes pour voir la connexion de l'écran avec le PIC. Sur le schéma ci-dessus, le bus de données est sur 8 fils, alors que dans la station il est en 4 fils.

3.1.3.3 Pic 18F4680 I/P

3.1.3.3.1 Dimensions du composant :

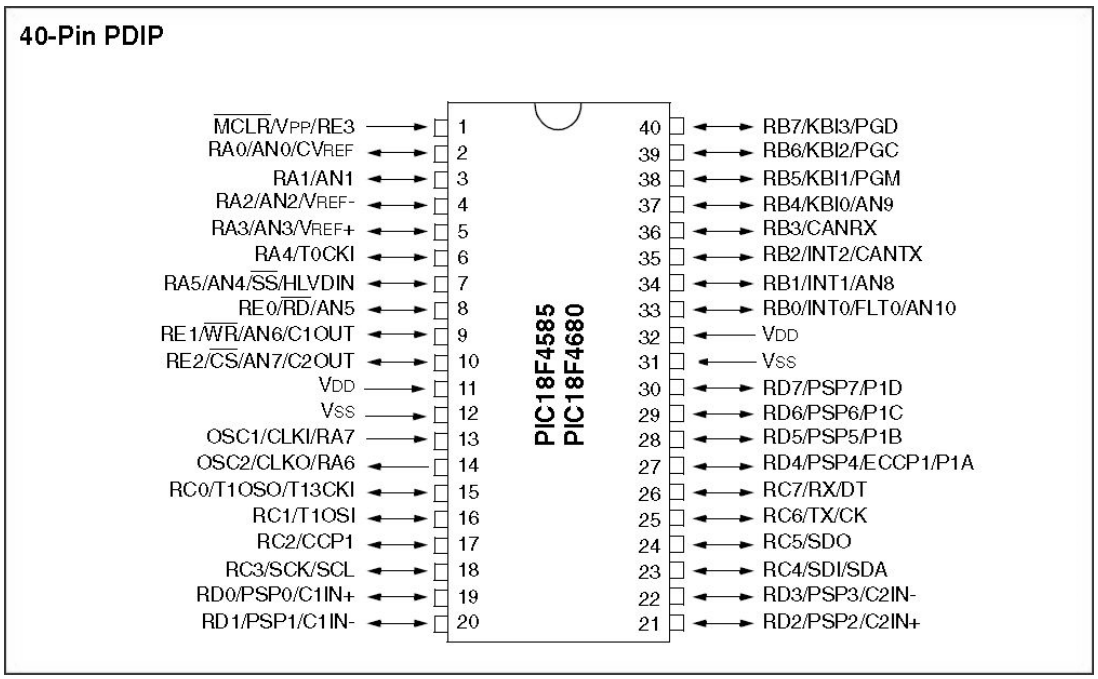


D = Longueur = 52 mm E = Largeur = 15.6 mm

A = Epaisseur du circuit = 3.2mm L = Longueur Pin = 3.6mm

e = Ecartement entre Pin : 2mm

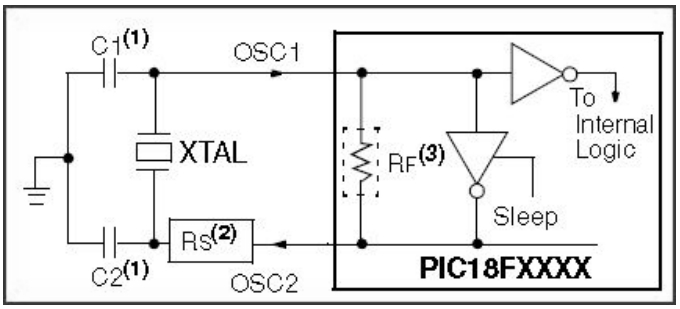
3.1.3.3.2 Brochage du PIC 18F4680 I/P :



3.1.3.3.3 Raccordement de l'oscillateur au PIC :

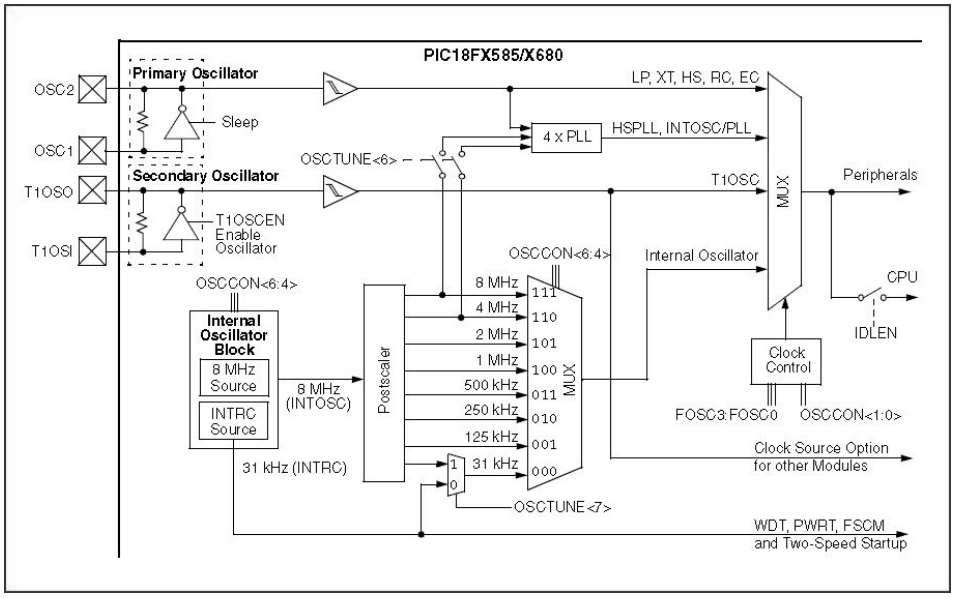
Le PIC est l'organe intelligent de la carte. C'est lui qui effectue toutes les opérations. Le fonctionnement interne d'un microcontrôleur est expliqué au point 4.

On a connecté un oscillateur de type Crystal au PIC, voici le schéma de raccordement :



Le pic peut être programmé pour être utilisé dans 10 modes différents d'oscillations. En le raccordant comme sur le diagramme ci dessus, nous choisissons que celui-ci fonctionne en haute vitesse (HS). Nous plaçons donc un Crystal de 20MHz (XTAL) et deux condensateurs non polarisés de 15pF (C_1 et C_2).
La valeur du Crystal a été calculée pour obtenir un fonctionnement sans faille. En effet, le microcontrôleur a besoin d'être cadencé à une haute fréquence pour permettre une communication CAN sans erreur. Des tests ont été effectués avec des cristaux de fréquences inférieures, et le taux d'erreurs pour la communication CAN était trop élevé. La lecture d'un bit dans une trame ne se faisait pas toujours au même endroit ce qui engendrait un certain décalage. Celui-ci engendrait une erreur d'autant plus grande que la fréquence est basse. Avec un Crystal de 20MHz, il n'y a aucune erreur de lecture.

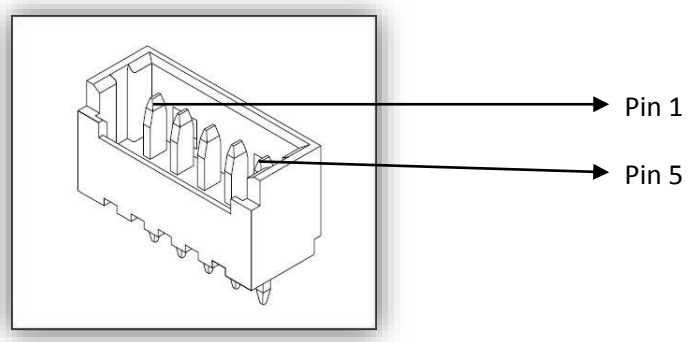
Voici la partie interne « Oscillator » du PIC :



Notre Crystal est connecté aux pins OSC1 et OSC2 du PIC. Ces pins représentent l'oscillateur primaire. Nous pouvons voir sur ce diagramme que le signal venant du Crystal arrive aux bornes d'un multiplexeur (MUX). C'est à cet endroit que l'on va effectuer le choix de l'oscillateur grâce à la partie « Clock Control ». On configure les différents mots de contrôle (FOSC3 : FOSC0 et OSCON<1 :0>) pour faire la sélection et ainsi paramétrer le multiplexeur. On peut constater qu'il est possible d'utiliser un oscillateur interne, déjà intégré au PIC. Il faut alors configurer les mots de contrôle OSCON<6 :4> pour obtenir l'oscillateur adéquat ainsi que sa fréquence de fonctionnement.

3.1.3.4 Les différents connecteurs

3.1.3.4.1 Le connecteur de programmation



On vient brancher sur ce connecteur le câble de programmation venant du programmeur PICKit2. C'est un connecteur 5 pins.
Explication de chaque pin :

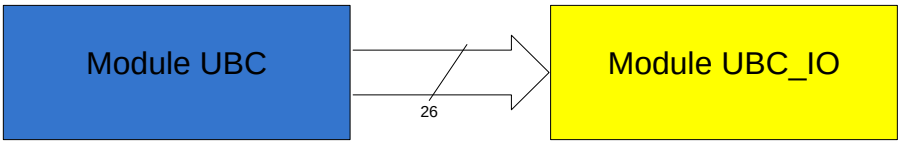
- Pin 1 : Alimentation. On peut alimenter le PIC soit par le programmeur (PICKit2), soit par le circuit lui-même (alimentation venant de la carte). Il faut signaler au PIC si l'alimentation lors de la programmation vient du programmeur ou du circuit (Configuration Bits dans MPLAB).

- Pin 2: MCLR: Master Clear. Pour signaler au PIC qu'il doit arrêter toute activité et qu'il va être programmé.
- Pin 3: PGC: C'est le signal de Clock (horloge) venant du programmeur. Les données envoyées au PIC sont cadencées par ce signal.
- Pin 4: PGD: C'est le signal de données venant du programmeur (synchronisé avec le signal d'horloge).
- Pin 5: Masse (GND).

Pour résumer, le programmeur envoie un signal d'avertissement au PIC (MCLR) signalant qu' il va être reprogrammé. Pendant ce temps, il envoie un signal d'horloge (PGC) et à chaque top d'horloge, un bit de donnée est envoyé (PGD). Les données contiennent le programme que le PIC devra effectuer (programme développé sous MPLAB).

3.1.3.4.2 Le connecteur pour câble plat

Le module UBC est équipé d'un connecteur pour câble plat. Ce câble sert à relier la carte UBC à la carte UBC_IO. Pour des raisons mécaniques, la carte a dû être scindée en deux parties. Une partie « Entrées/sorties » et une partie « Intelligence ». En effet, le boitier allant contenir les deux modules possède des dimensions standards. Il est donc préférable de respecter ces dimensions car un boitier de dimensions non standards coûte excessivement cher.



Par ce câble passe différents signaux :

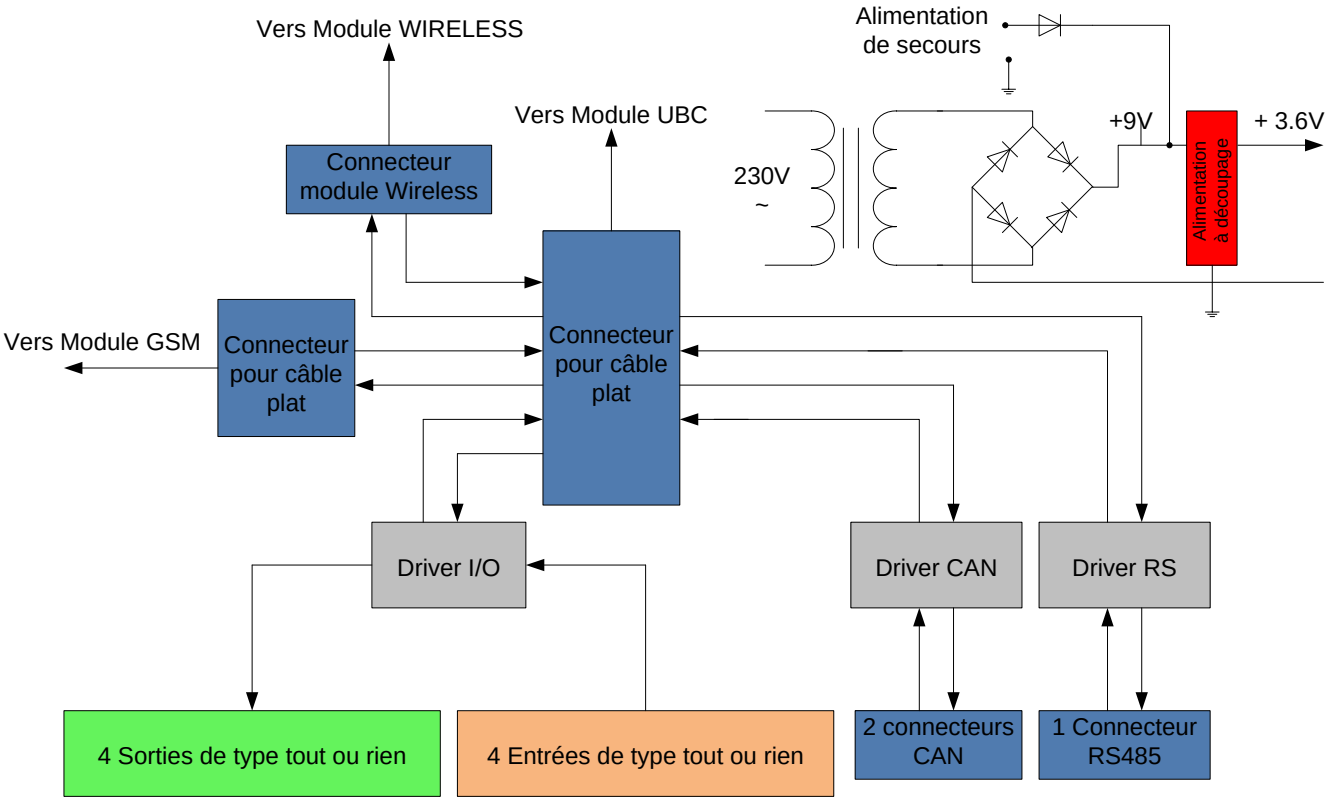
- Signaux pour les communications CAN, RS, Wireless, GSM, I^2C.
- Signaux d'alimentation (+12, GND).
- …

Figure 7 Boitier UBC complet ouvert

Figure 8 Boitier UBC fermé

3.2 Le module UBC_IO

3.2.1 Schéma bloc de la carte.



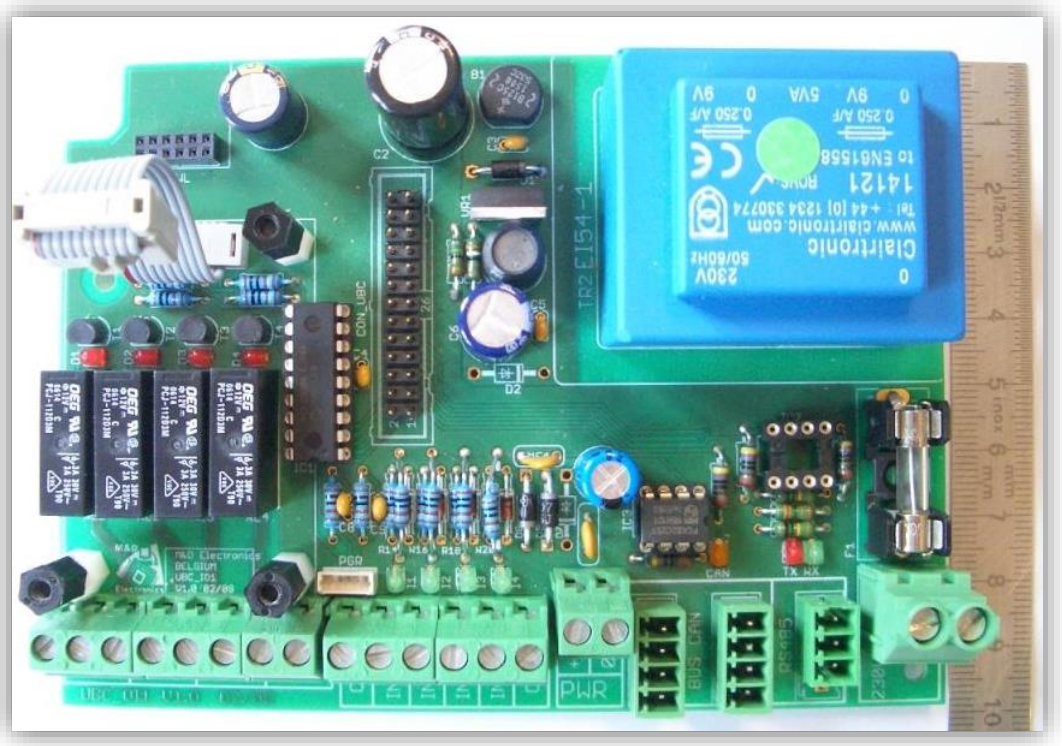


Figure 9 Module UBC_IO sans le module GSM

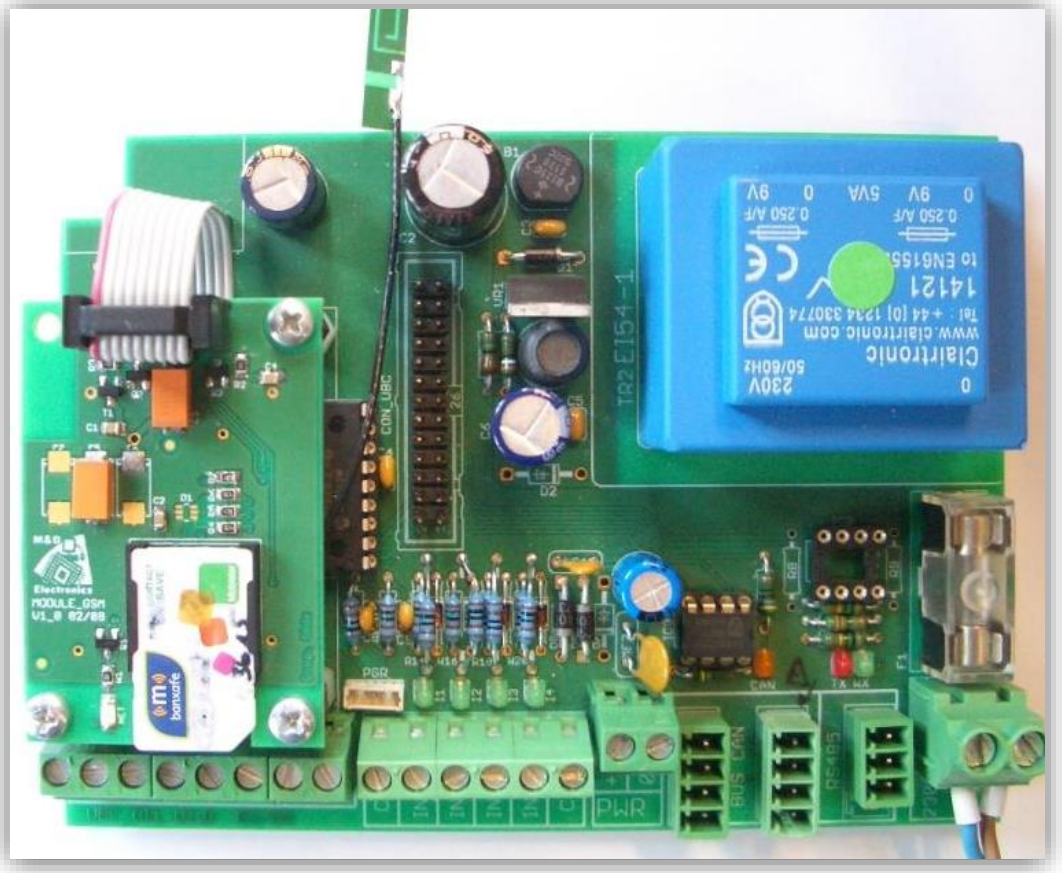


Figure 10 Module UBC_IO avec le module GSM

3.2.2 Rôle de la carte.

Sur cette carte, différents composants sont installés. Tout d'abord, nous avons une parti alimentation. Il est constitué d'un fusible 250V AC 0.5A, un transformateur 230/9V, un pont redresseur, un régulateur de tension par hachage (9V → 3.6V) et tout un ensemble de capacités et de selfs permettant de filtrer et lisser la tension de sortie. Notons qu'il est possible de connecter une alimentation de secours. De ce fait, la station est en fonctionnement malgré une coupure de courant au niveau du réseau.

Cette carte est utilisée pour y connecter toutes les entrées/sorties utilisées par la station. La carte possède 4 sorties de type « tout ou rien » (commandées par des relais) et 4 entrées du même type. Ces entrées/sorties sont contrôlées par le circuit MP23008-E/P de chez Microchip.

Deux bus de données sont raccordés à cette carte :

- L'interface du bus de terrain CAN. La carte possède un driver CAN (PCA82251 de chez Philips) et deux connecteurs. C'est ce bus qui est utilisé pour communiquer avec les modules MOD_CAPT et MOD_VENT
- L'interface du bus série RS485, avec son driver et son connecteur. Ce bus peut être est utilisé pour communiquer avec un PC.

Enfin, cette carte possède 3 connecteurs pour câbles plats (de tailles différentes). A ces connecteurs viennent se brancher le module GSM (pour la communication par sms), le module Wireless (communication radio) et le module UBC (expliqué précédemment).

Le schéma électrique général de la carte et le dessin de celle-ci se trouvent dans les annexes aux pages III et IV.

3.2.3 Explication des différents composants.

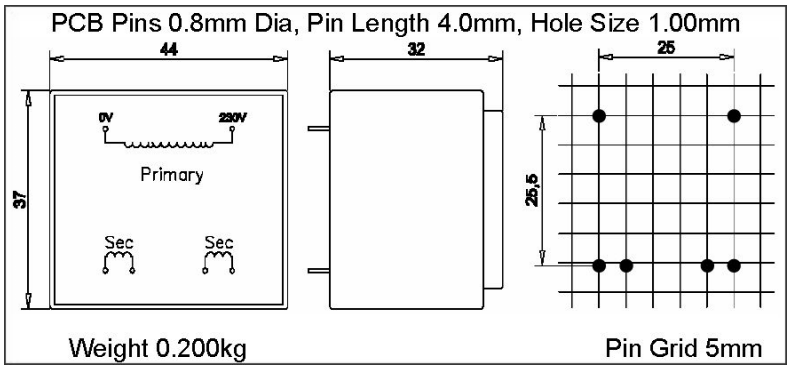
3.2.3.1 La partie alimentation

3.2.3.1.1 Le fusible.

Le fusible est un organe de sécurité. En fonctionnement normal, il doit assurer la conduction du courant électrique. Si un défaut apparaît (court-circuit, erreur de manipulation, etc.), celui-ci va fondre et empêcher le courant de détériorer le reste du circuit. Il est placé à la source de la partie alimentation !

3.2.3.1.2 Le transformateur.

Il a pour but de transformer la tension du réseau (230V, 50Hz) en une tension de 9V. Sa puissance est de 5VA → courant maximum = 5VA / 9V = 555mA. Voici sa fiche technique :



Voici un tableau reprenant les différentes consommations de chaque module avant le transformateur:

Nom du module	Consommation
Module UBC_IO	22,5 mA
Module UBC_IO + module UBC avec écran allumé	24,3 mA
Module UBC_IO + module UBC avec écran éteint + module GSM	22,9 mA
Module UBC_IO + module UBC avec écran allumé + module GSM	25,1 mA
Module UBC complet + module MOD CAPT sans capteurs	25,7 mA
Module UBC complet + module MOD CAPT avec capteurs	25,9 mA
Module UBC complet + module MOD CAPT avec capteurs + module MOD_VENT	27,4 mA

Grâce à ces mesures, il est possible de déduire une puissance consommée :

$$\boldsymbol{P} = U \times I = 230V \times 27.4mA = \mathbf{6.3}\boldsymbol{W}$$

Voici un autre tableau reprenant les différentes consommations de chaque module en alimentant la station via l'alimentation de secours (Batterie 12V 7Ah) :

Nom du module	Consommation
Module UBC_IO	17 mA
Module UBC_IO + module UBC avec écran éteint	22 mA
Module UBC_IO + module UBC avec écran allumé	95 mA
Module UBC_IO + module UBC avec écran éteint + module GSM	30 mA
Module UBC_IO + module UBC avec écran allumé + module GSM	104 mA
Module UBC complet + module MOD CAPT sans capteurs	125 mA
Module UBC complet + module MOD CAPT avec capteurs	136 mA
Module UBC complet + module MOD CAPT avec capteurs + module MOD_VENT	180 mA

Par ce tableau, on se rend compte de la consommation de chaque module :

- le module UBC_IO : 17mA ;
- le module UBC avec l'écran éteint : 5mA ;
- l'écran allumé : 73mA ;
- le module GSM : 8mA ;
- le module MOD_CAPT sans les capteurs : 21mA ;
- les capteurs connectés au module MOD_CAPT consomment : 11mA ;
- le module MOD_VENT : 44mA.

Grâce à ces mesures, il est possible de déduire une puissance consommée :

$$\boldsymbol{P} = U \times I = 12V \times 180mA = \mathbf{2.6}\boldsymbol{W}$$

On se rend compte que le plus gros consommateur de courant est le rétro-éclairage de l'écran LCD. C'est pourquoi je décide de l'allumer uniquement pendant une minute au démarrage et lors d'une pression sur un bouton poussoir. Il n'est pas nécessaire de l'allumer tout le temps.

On observe une différence de puissance consommée entre le raccordement sur batterie ou sur le réseau. En effet, la puissance consommée sur batterie est 2.5 fois moins grande que sur le réseau. Cette différence de puissance vient du transformateur qui n'a pas un rendement optimum. Même à vide, le transformateur consomme quelque chose (22,5mA x 230V = **5,175W** faisant partie des 6.3W !).

Les mesures ont été effectuées sans activer les relais. On se rend compte que le transformateur est surdimensionné pour cette application (550mA alors que 180mA maximum sont utilisés). Mais les cartes UBC et UBC_IO sont utilisées dans d'autres applications où la demande en courant est bien plus importante.

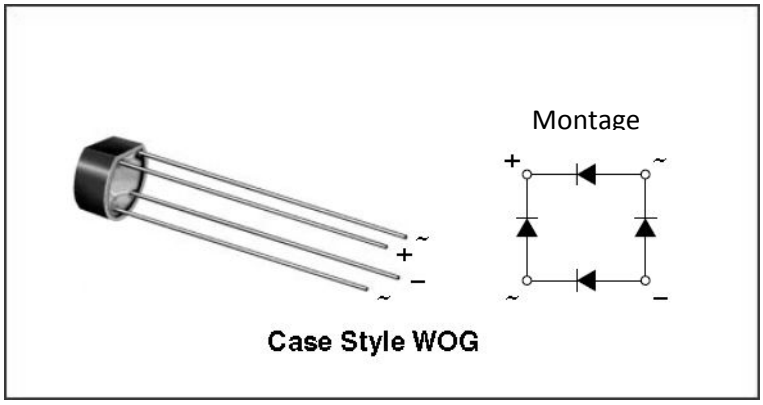
La batterie de secours sera automatiquement montée sur la station pour parer à une éventuelle panne de courant. Un calcul permet de déterminer la durée de vie de la batterie, si celle-ci devait alimenter toute la station :

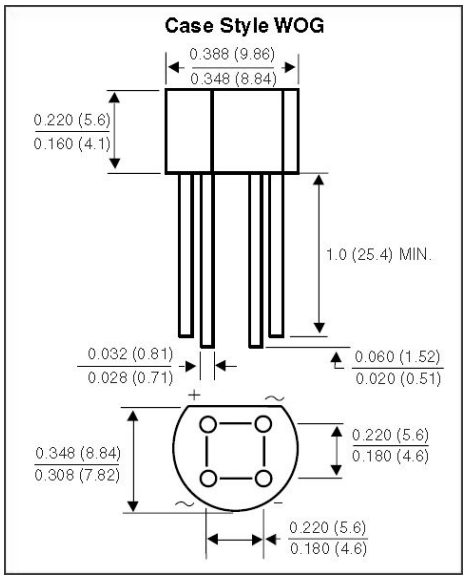
$$\boldsymbol{durée\ de\ vie} = \frac{7Ah}{0.18A} = \mathbf{39}\ \boldsymbol{heures}$$

Cette durée de vie est calculée en admettant que l'écran soit allumé tout le temps, et donc, elle représente la durée de vie minimum.
Si on effectue le même calcul en admettant que le rétro-éclairage ne soit pas allumé, la durée de vie serait de :

$$\boldsymbol{durée\ de\ vie} = \frac{7Ah}{0.107A} = \mathbf{65}\ \boldsymbol{heures}$$

3.2.3.1.3 Le pont redresseur.





Les dimensions sont exprimées en Inc. et en millimètres entre parenthèses.

Figure 11 Dimensions

Le pont redresseur (ou pont de Graetz) est un assemblage de 4 diodes montées en pont (voire schéma plus haut). Il a pour but de redresser un courant alternatif en un courant continu.

Grâce à ce montage, on effectue un redressement double alternance.

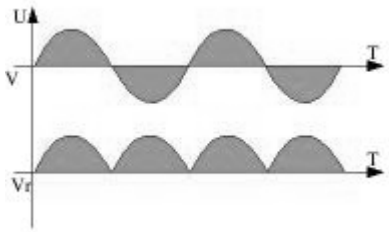


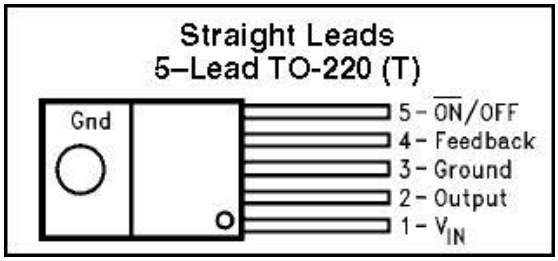
Figure 12 Redressement double alternance

Une fois la tension redressée, une capacité électrolytique va lisser la tension pour obtenir une tension fortement continue.

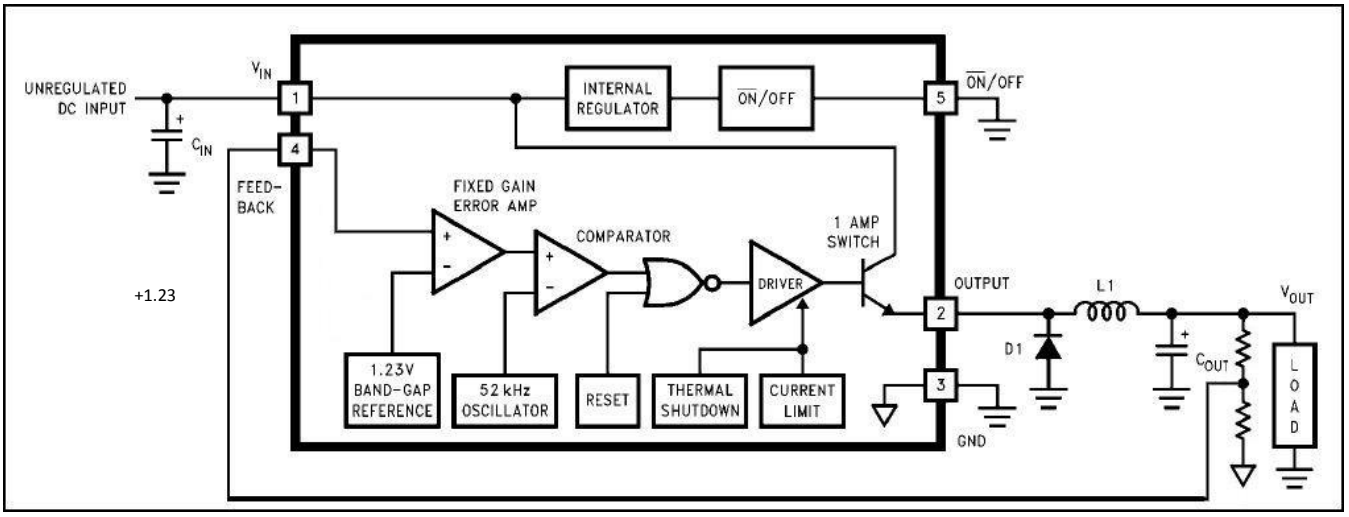
3.2.3.1.4 Le régulateur de tension par hachage.

Le régulateur aura pour but de maintenir la tension de sortie à une valeur constante (3,6volts). Pour que le régulateur fonctionne normalement, la tension à son entrée doit être supérieure de quelques volts vis-à-vis de sa tension de sortie. Il s'agit du circuit LM2575HVT ADJP+ de chez National Semiconductor.

3.2.3.1.4.1 Brochage du composant.



3.2.3.1.4.2 Schéma interne du composant.



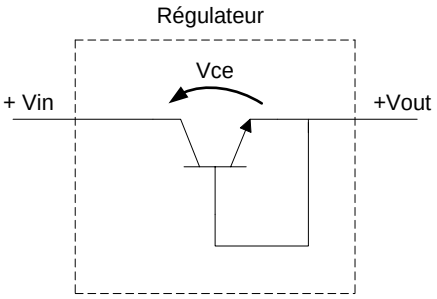
Tout d'abord, via l'entrée ON/OFF (Pin 5), il est possible d'activer ou non le régulateur de tensions. Ensuite, étant donné que c'est un régulateur, il est équipé d'un « Feedback » (Pin 4) → la sortie en tension est reprise et injectée dans la commande de hachage de la tension de sortie afin de régler au mieux celle-ci.

Sans cette boucle, le régulateur ne saurait pas savoir s'il régule correctement la tension ou pas. Cette boucle est donc indispensable quant à l'obtention d'une tension constante.

Sur la Pin 3, nous avons la masse.

La sortie en tension régulée sort sur la Pin 2, c'est la tension utilisée avant le filtrage !

Sur la Pin 1, nous avons la tension d'entrée du régulateur. Celle-ci doit toujours être supérieure à la tension de sortie souhaitée (on parlera d'une marge de 2V pour être certain).

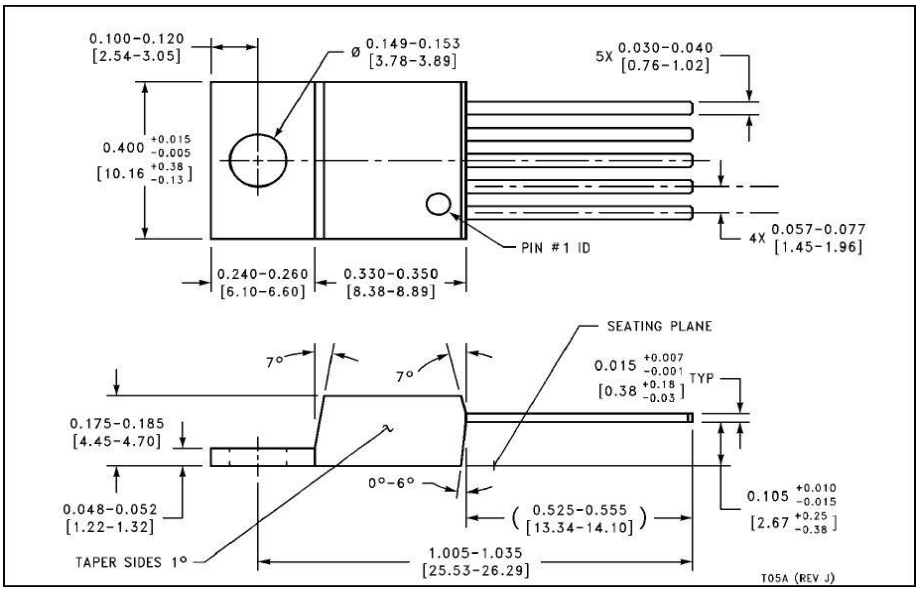


Sur ce schéma, on voit qu'un régulateur linéaire (Internal Regulator) est composé d'un transistor. Quand un transistor conduit, une tension collecteur/émetteur apparaît (V_{CE}). Celle-ci ne peut être négligeable. Admettons que nous désirons une tension de +5V en sortie alors que la tension d'entrée est de +8V. Il sera possible d'obtenir +5V → $V_{IN} – V_{CE} = V_{OUT}$ → 8 – 2 = 6 → OK !
Si la tension d'entrée est de 5V → 5 – 2 = 3 → le régulateur ne sait pas fournir la tension de sortie désirée. Le transistor a beau être passant au maximum, ses pertes font que l'on ne peut négliger la tension V_{CE} !
La valeur V_{CE} (2V) est prise à une valeur suffisante dans les calculs pour éviter les problèmes. Cette valeur varie en fonction de chaque régulateur linéaire.

On constate qu'au point de vue interne du régulateur de type « switching », celui-ci est composé d'autres circuits qui ont eux-mêmes des pertes. Il faut tenir compte de ces pertes pour déterminer le régulateur adéquat.
La différence entre un régulateur linéaire et un régulateur de type « switching », est celle-ci :
Le premier contient un transistor qui laisse passer plus ou moins le courant électrique → on peut obtenir des phénomènes de saturation.
Le second utilise le principe du « tout ou rien », c'est-à-dire qu'il contient un transistor qui laisse passer tout le courant, ou aucun courant → principe du PWM (on fait varier le rapport cyclique).

Fonctionnement du régulateur :
Un pont diviseur placé en sortie du régulateur va nous permettre d'obtenir une tension de +1.23V qui sera injectée dans la commande de hachage. Cette tension arrive dans un comparateur et sera donc comparée à un seuil de +1.23V. Si la tension de « Feedback » venait à descendre en dessous de +1.23V, l'oscillateur va débiter un signal rectangulaire cadencé à 52kHz de plus grand rapport cyclique qui arrivera sur la base du transistor et commandera celui-ci. La sortie du transistor passe par un ensemble self/capacité afin que la tension de sortie soit lissée. Le régulateur va donc jouer sur le rapport cyclique du signal rectangulaire pour faire varier la tension.
Précisons que si la fréquence de l'oscillateur interne augmente, la taille de la self et de la capacité (L_1, C_{OUT}) pourra être beaucoup plus petite, mais le transistor va chauffer beaucoup plus car les pertes de commutation vont augmenter. Certains régulateurs ont des oscillateurs de 1MHz !

3.2.3.1.4.3 Dimensions du composant.

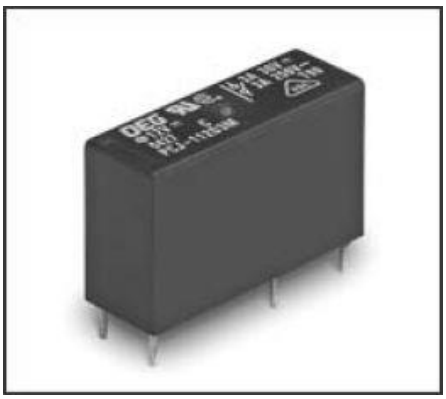


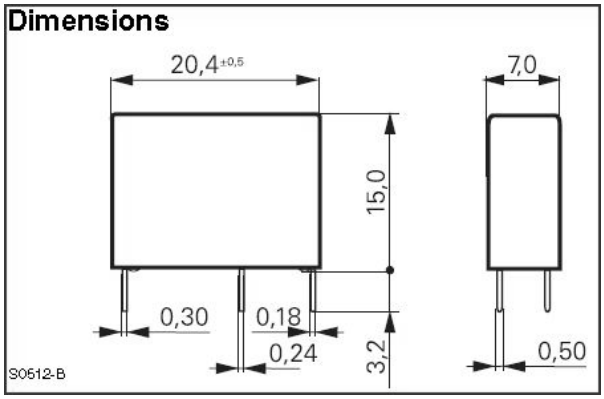
Les dimensions entre crochets sont en millimètres.

3.2.3.1.5 L'alimentation de secours.

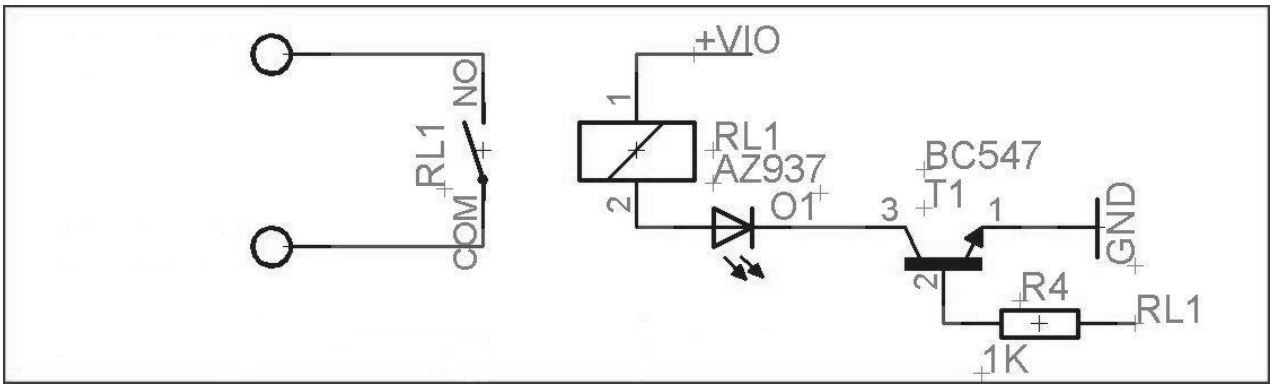
Pour que la station puisse continuer à fonctionner, une alimentation de secours a été installée. Si la station du réseau vient à chuter brusquement, la diode D9 *(voire schéma électrique à l'annexe 3)* est polarisée en direct, et la tension de la batterie peut alimenter la station. En fonctionnement normal, la tension du réseau étant plus grande que la tension de la batterie, la diode est polarisée en inverse et donc aucun courant ne peut circuler.

3.2.3.2 Les sorties « tout ou rien. ».





Chaque relais est raccordé de la façon suivante :



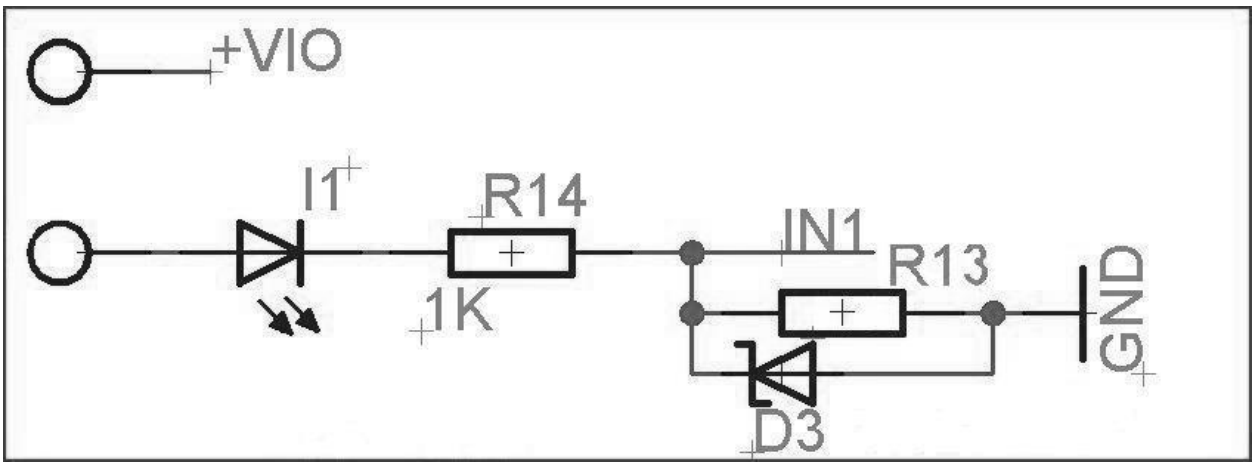
La bobine est raccordée en série avec une led rouge et un transistor NPN. Lorsqu'on applique une tension de 5V à la borne RL1 (sortie du circuit MCP23008-E/P), le transistor T1 devient passant. Ce qui engendre un passage du courant entre la borne $+V_{io}$ et la masse. Le passage du courant au travers de la bobine va enclencher le relais, et le contact RL1 va se fermer. Ce contact peut être placé comme un interrupteur dans n'importe quel circuit. Il suffit de commander le relais pour fermer ou ouvrir le contact.
La led placée en série indique si le relais est ON ou OFF.
Une diode de roue libre pourrait être mise en parallèle avec la bobine du relais. Elle absorberait l'effet selfique de la bobine lorsque on arrête de l'alimenter, mais la résistance de celle-ci étant élevée (2880Ω), le courant du bobinage est donc faible, ce qui donne un faible effet selfique.

$$U_{bobine} = L \times \frac{di}{dt}$$

La tension aux bornes de la bobine reste donc faible lorsqu'on arrête de l'alimenter. Mais pour plus de sécurité, une diode de roue libre sera ajoutée aux futures cartes.

La station possède 4 sorties « tout ou rien ».

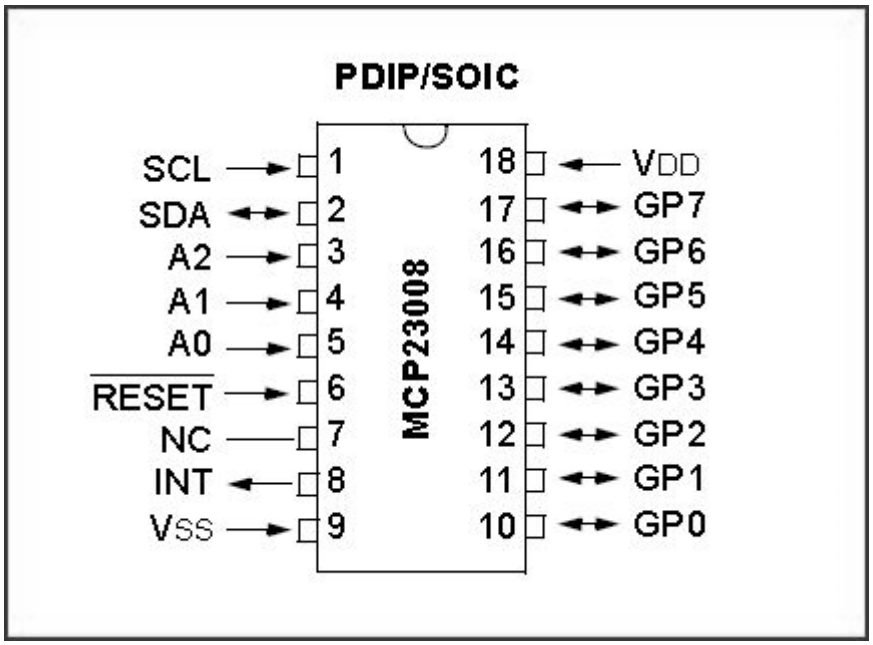
3.2.3.3 Les entrées « tout ou rien ».



Le principe de l'entrée « tout ou rien » est un contact placé entre la borne $+V_{io}$ (+20V) et la deuxième borne. Lorsque le contact est ouvert, la résistance de pull down (R_{13}) impose un potentiel de 0V (GND) à l'entrée IN1 du PIC (→pas de potentiel flottant). Si le contact vient à se fermer, un courant va pouvoir circuler, passant ainsi dans la led qui s'allume. On aurait donc une tension de 20V à la borne IN1, qui est une entrée du circuit MCP23008-E/P. Ceci étant trop élevé, nous plaçons une diode Zenez de 4.7V (D_3) pour amener le potentiel de la borne IN1 à ±5V (potentiel acceptable par le circuit MCP23008-E/P).
La station possède 4 entrées « tout ou rien ».

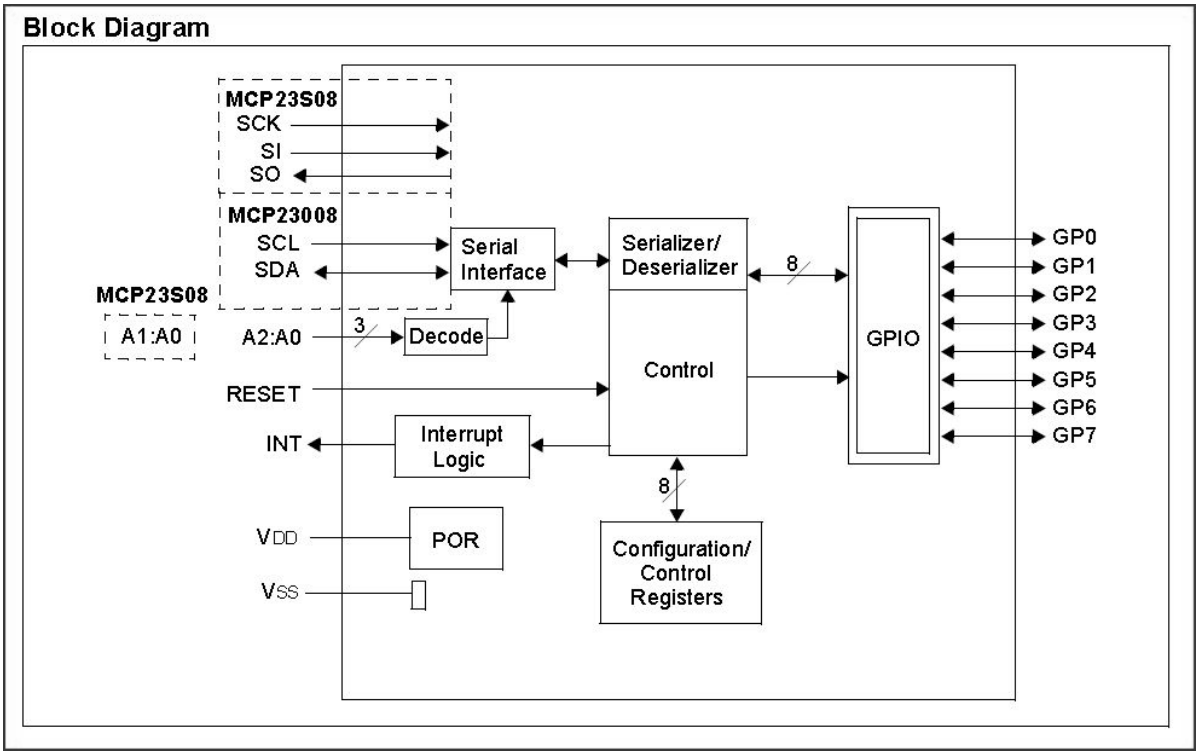
3.2.3.4 Le circuit MCP23008 – E/P.

3.2.3.4.1 Brochage du composant.



- Pin 1 : SCL : Entrée de la Clock du bus I²C reliant le PIC et le circuit.
- Pin 2 : SDA : Canal bidirectionnel. Les données transitent par ce canal et vont du PIC au circuit et inversement.
- Pin 3→5 : Pour encoder une adresse au niveau Hardware (adresse du circuit). En effet, si plusieurs circuits MCP23008 sont utilisés, il faut utiliser des adresses pour accéder à chacun d'eux. On pourra donc dépasser le nombre d'entrées/sorties « tout ou rien » qui est de maximum 8 par circuit.
- Pin 6 : Reset : Pour reseter tout le circuit.
- Pin 7 : NC : Pin non utilisée.
- Pin 8 : INT : sortie interruptible.
- Pin 9 : V_{SS} : Masse du circuit.
- Pin 10→17 : Ce sont des pins bidirectionnels, on les configure soit comme sorties, soit comme entrées, ou même les deux (comme utilisé dans l'application !).
- Pin 18 : V_{DD} : Alimentation du circuit.

3.2.3.4.2 Schéma interne du composant



L'organe central de ce circuit est la partie « Control ». On peut le reseter (via le signal RESET), il contrôle le GPIO, il interagit avec l'interface série et les registres de contrôle. Le GPIO s'occupe de la gestion des entrées/sorties. Selon sa programmation, c'est lui qui fixe les entrées et les sorties (on peut avoir 8 entrées, pas de sorties, ou 3 entrées et 5 sorties, etc.). L'interface série permet de gérer la communication entre le PIC et ce circuit (transfert de données, décodage d'adresse, etc.). On peut communiquer avec le circuit via deux bus. Soit en SPI ou en I²C *(Voire au point 5 pour les explications de ces deux bus).* Mais attention, ce sont deux circuits différents :

- MCP23S08 →SPI
- MCP23008 → I2C

3.2.3.4.3 Explication

Le circuit nous permet donc de contrôler les entrées/sorties « tout ou rien ». Le PIC pourra demander d'activer une sortie, et viendra écrire l'état des sorties dans le circuit. A l'inverse, il peut demander l'état des entrées, et viendra lire l'état de celles-ci dans le circuit. Le circuit MCP23008-E/P permet donc de créer une interface entre le PIC et les entrées/sorties « tout ou rien ».

3.2.3.5 Les différents connecteurs.

3.2.3.5.1 Le connecteur de programmation.

Il s'agit du même connecteur que celui placé sur la carte UBC *(Voire point 3.1.3.4.1 Le connecteur de programmation).* Pour des raisons de facilité, nous avons placé deux connecteurs de programmation. En effet, une fois les cartes placées dans leur boîtier, il est impossible d'accéder au connecteur de programmation placé sur la carte UBC. Le connecteur placé sur la carte UBC_IO permet de programmer le PIC sans devoir démonter tout le boitier mais en enlevant juste le capot en plastique. En enlevant ce capot, on peut accéder à tous les connecteurs de la carte UBC_IO *(voire figure 13).*

Figure 13 Connecteurs UBC_IO (sans le capot en plastique)

3.2.3.5.2 Le connecteur pour le bus CAN.

Le module UBC_IO possède deux connecteurs 4 pins ($+V_{IO}$, CANH, CANL, GND). On peut donc y connecter deux bus CAN.

Embout femelle

Embout mâle

<u>Quelques caractéristiques :</u>

Espacement entre les pins : 3.5mm
I_{MAX} = 8A.
U_{MAX} = 160V AC.

3.2.3.5.3 Le connecteur pour le bus série RS485.

Il s'agit du même connecteur que celui utilisé pour le bus CAN à la différence que le bus RS485 utilise trois fils (RX, TX, GND) → Embout mâle et femelle en 3 pins.

3.2.3.5.4 Les connecteurs pour les entrées/sorties.

Ce sont des connecteurs à visser. Il en existe des différentes tailles.

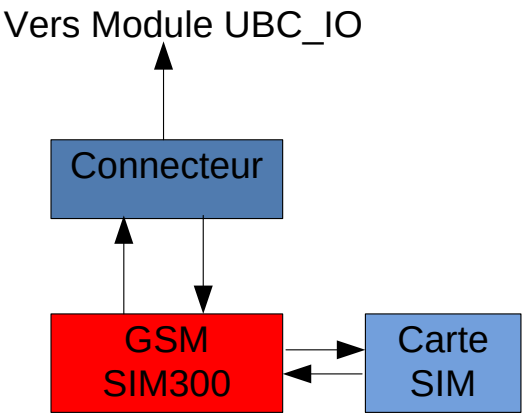
Attention : Le connecteur d'alimentation est bien plus gros que ceux pour les entrées/sorties car un plus gros courant y circule. De plus, le pitch (espacement entre Pin) est plus grand ; un risque d'arc électrique est possible entre les Pins si celles-ci ne sont pas assez écartées car haute tension!

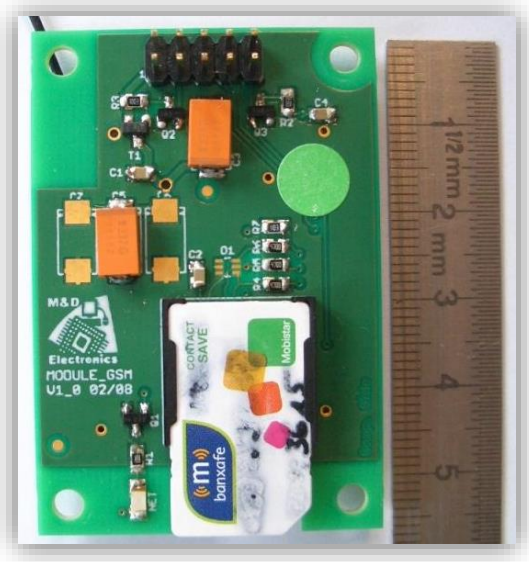
Quelques caractéristiques :

Espacement entre les pins : 6.5mm
I_{MAX} = 32A.
U_{MAX} = 500V AC.

3.3 Le module MODULE_GSM

3.3.1 Schéma bloc de la carte.





Coté carte SIM & PLUG

Coté GSM SIM300

3.3.2 Rôle de la carte.

Le rôle de la carte MODULE_GSM est de servir d'interface de communication entre la station et un GSM externe via une liaison par SMS.

En effet, un utilisateur externe a la possibilité d'envoyer un SMS à la station météo pour en connaître les différentes mesures météorologiques. La station lui répondra le plus rapidement possible.

La carte MODULE_GSM possède :

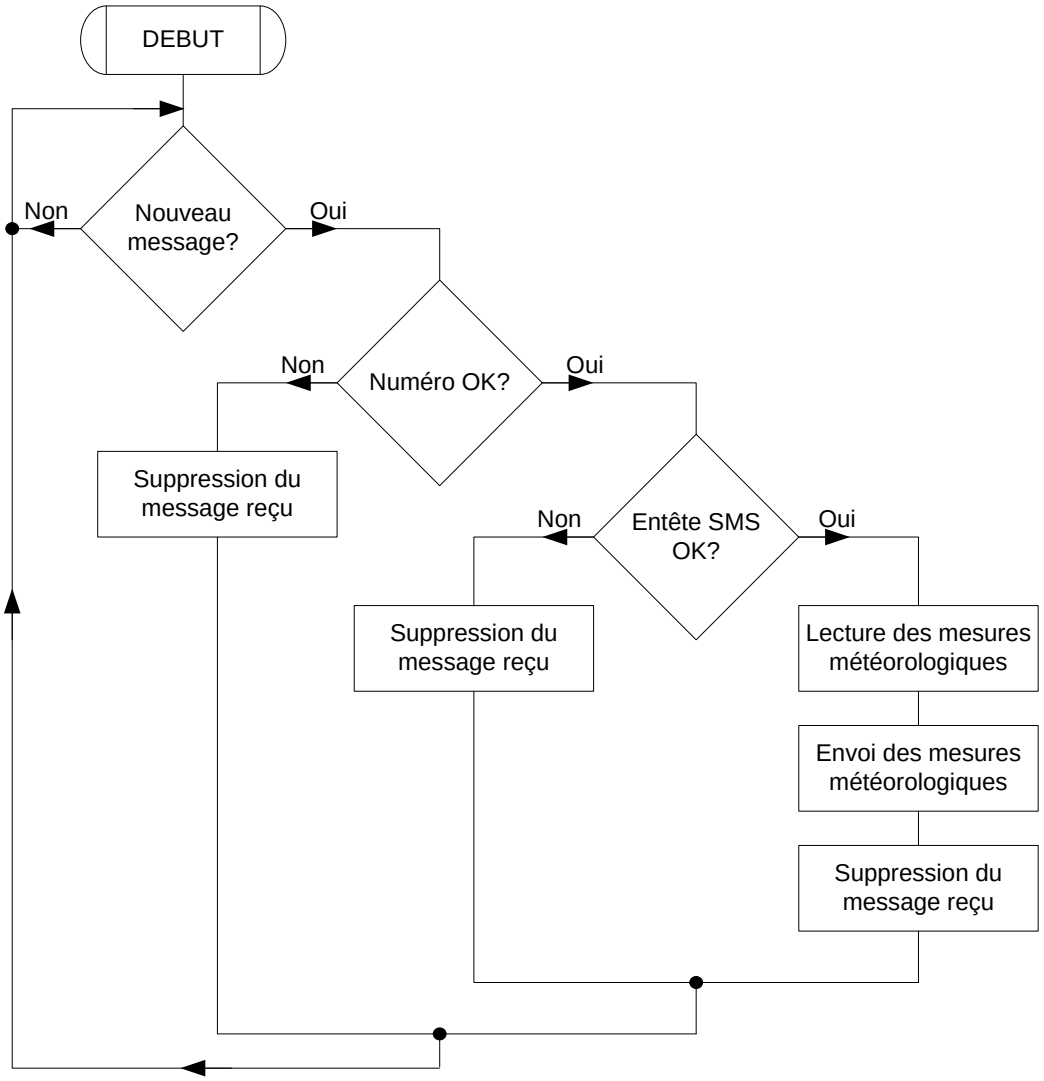
- Un GSM de chez SIM. Il s'agit du type 300DZ.
- Un socle pour carte SIM, nécessaire au fonctionnement normal du GSM (obtention d'un numéro, etc.).
- Une antenne.

C'est le module UBC qui envoie les commandes AT nécessaires (via bus UART) à la communication entre la station et le GSM. Cette carte est placée sur la carte UBC_IO.

Notons que l'on peut placer une autre antenne pour augmenter la portée du GSM et la placer dans un autre endroit.

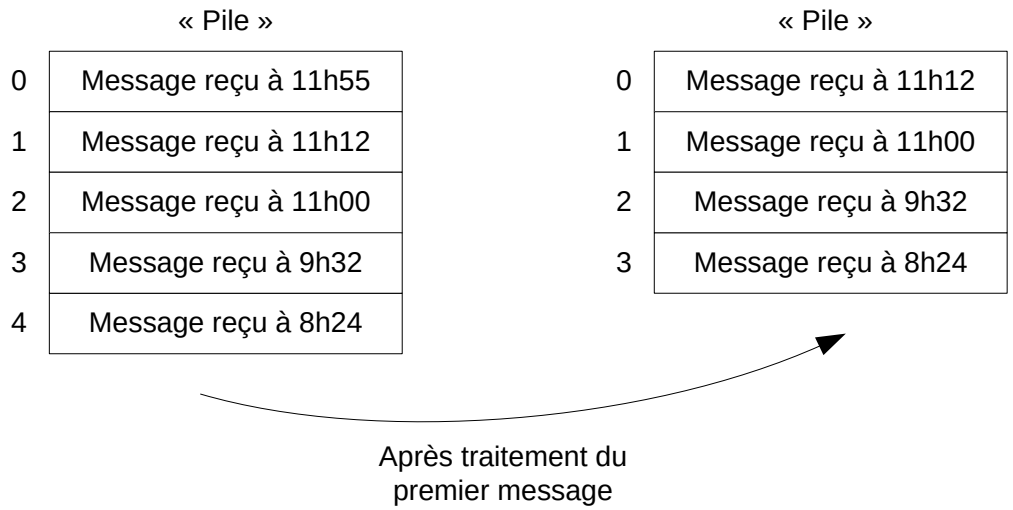
Le schéma électrique général de la carte et le dessin de celle-ci se trouvent dans les annexes aux pages V, VI et VI.

Voici un organigramme du processus d'émission/réception d'un message :



Le module UBC (son PIC) scrute l'arrivée d'un nouveau message toutes les secondes. Lorsqu'un message est arrivé, un flag (drapeau) est levé pour signaler celui-ci. La trame constituant le SMS est alors analysée.
Tout d'abord, on regarde le numéro de GSM et on analyse son format. Le numéro reçu est-il bien un numéro de type +324xxxxxxxx ? Si ce n'est pas le cas, le message reçu est directement supprimé pour ne pas remplir la mémoire de messages inutiles.
Le numéro de GSM reçu étant correct, le contenu du SMS est alors analysé. On regarde l'entête du message. Celle-ci doit être composée du mot « meteo », sans accent ni majuscule !
L'entête ainsi vérifiée et correcte, la station va alors composer le SMS de réponse avec les différentes mesures météorologiques et répondre au numéro « appelant ». Le message reçu sera ensuite supprimé pour les mêmes raisons que précédemment.

Dans le cas où la station reçoit plusieurs messages à la fois (Exemple : on allume le GSM et plusieurs messages ont été envoyés pendant le temps que le module était éteint), celle-ci traite les messages séparément. Les messages sont placés dans une « pile » (sorte de tableau de stockage). On traite le premier message et ensuite, on le supprime. Le deuxième message se retrouve donc en première position, est traité de la même manière et ainsi de suite.



3.3.3 Explication des différents composants.

3.3.3.1 SIM 300DZ.

Ce composant reprend toutes les mêmes fonctions qu'un GSM traditionnel (acheté en magasin). A la différence que celui-ci peut-être configuré selon l'application désirée. Si on veut émettre des appels ou en recevoir, c'est possible. Ici, nous utilisons uniquement l'émission et la réception de SMS. Bien d'autres fonctions pourraient être envisagées mais celles-ci ne seraient pas utiles dans l'application de la station météorologique.

3.3.3.2. Socle pour carte SIM.

Ce composant sert de support pour une carte SIM. Cette carte est nécessaire au fonctionnement de tout GSM ! Il peut accueillir une carte SIM de n'importe quel opérateur téléphonique. Cette carte permet donc d'avoir un numéro de téléphone (à connaître pour joindre la station) et un code PIN (code de sécurité).

3.3.3.3 Les différentes antennes.

Antenne déportée

Antenne standard

Admettons que le module UBC soit placé dans une cabine en métal, les ondes du GSM vont percuter les bords de cette cabine et le signal ne sortira jamais de celle-ci → aucun réseau, aucune communication possible avec l'extérieur.
On intégrera alors l'antenne déportée qui sera placée par exemple sur le toit de la cabine et permettra ainsi d'être dans un environnement « sain ».
Si les conditions sont bonnes (aucunes perturbations), on peut placer l'antenne standard à l'intérieur du boitier contenant le module UBC.

3.3.3.4 Explication des commandes AT.

Les commandes AT constituent un langage de commande développé à l'origine pour la communication avec un modem. Ici, elles sont utilisées pour la communication entre le module GSM et le PIC placé sur le module UBC. Un grand nombre de commandes AT sont disponibles et permettent d'effectuer diverses fonctions mais, dans mon application, les commandes AT sont utilisées pour :

- envoyer le code PIN au GSM ;
- rechercher un réseau ;
- voir si on a reçu un nouveau message ;
- lire le message reçu ;
- envoyer le numéro de GSM sur lequel la station devra répondre ;
- envoyer le contenu du texte du message ;
- supprimer le message reçu.

Je n'ai pas eu besoin des autres commandes AT (émission d'un appel, réception d'un appel,…).

Voici un exemple de syntaxe utilisée dans le programme pour envoyer le code PIN au GSM :

```
case CMD_PIN:
data = 3615;
 sprintf(Trame_TX,"AT+CPIN=\"%d\"\r",data);       // OK
break;
```

On place le code PIN dans une variable « data ». Cette variable sera envoyée au GSM via le signal TX (aussi utilisé pour les communications RS485 → attention aux conflits !)

On envoi donc « AT » suivi de « +CPIN = » et de la variable « data ». La fin de la commande AT se termine par l'envoi de « \r ».

Chaque commande est envoyée sous la forme d'une ligne de texte encodé en ASCII, terminée par le caractère \r seul (code ASCII 13

Un autre exemple de syntaxe utilisée dans le programme pour demander au GSM s'il a reçu un nouveau message :

```
case CMD_NEW_MSG:
strcpy(Trame_TX,"AT+CPMS?\r");
break;
```

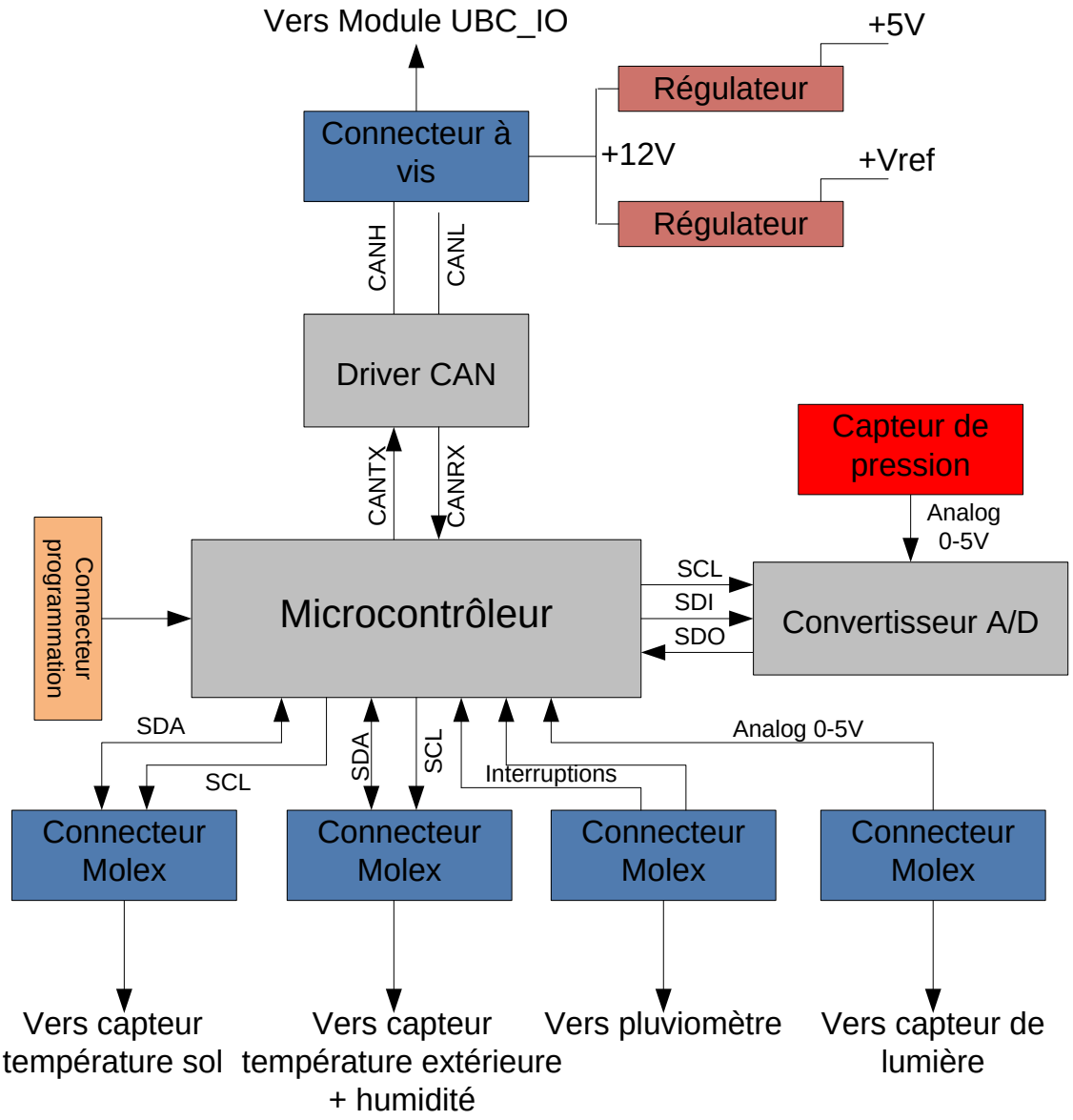
3.3.3.5 Aspect d'un message reçu.

Voici la forme du message envoyé par la station contenant les différentes mesures :

Bonjour : Pierre
Temp ext : +23.1
Temp sol : +19.2
Hum : 51%
Vit : 15m/s
Dir : N-E
Pluie : 0l/h
Lum : 51%
Pres : 984hPa

3.4 Le module MOD_CAPT

3.4.1 Schéma bloc de la carte.



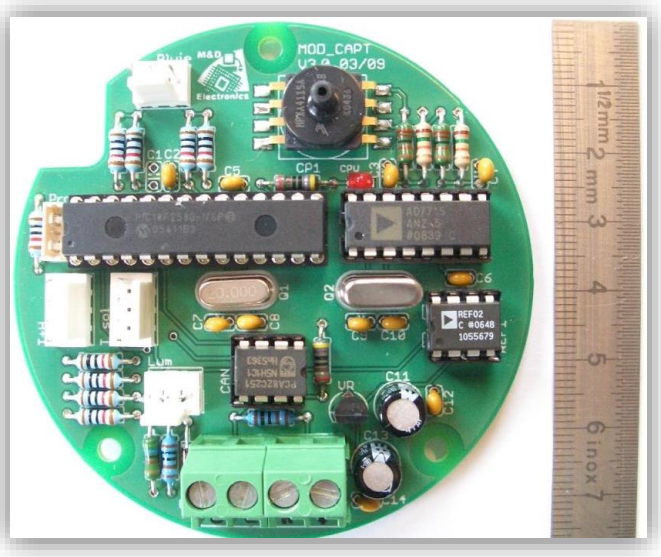


Figure 14 Module MOD_CAPT sans les capteurs connectés

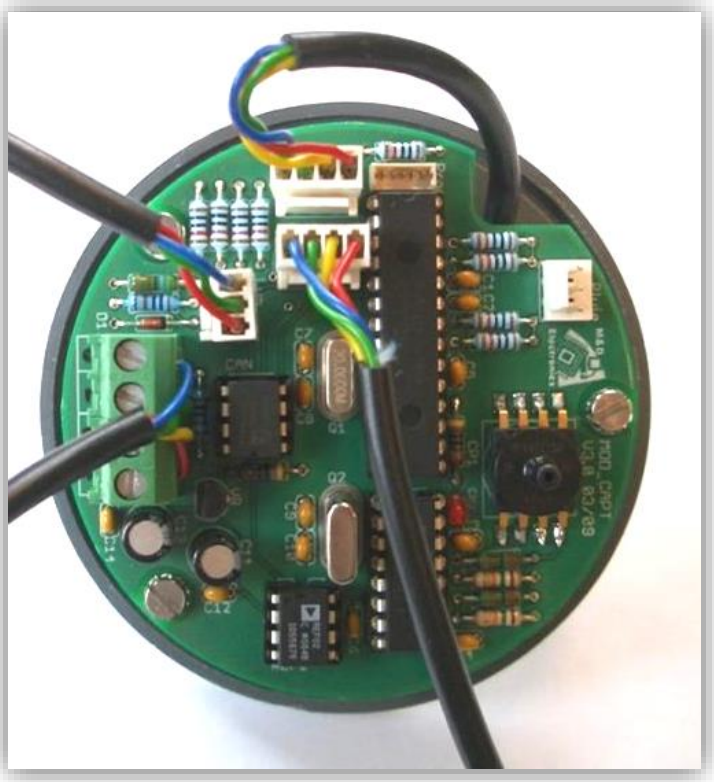


Figure 15 Module MOD_CAPT avec les capteurs connectés

Figure 16 Module MOD_CAPT dans son boîtier

3.4.2 Rôle de la carte.

Cette carte est le premier module effectuant les mesures. Différents capteurs y sont connectés :

- le capteur de pression atmosphérique qui est directement placé sur la carte ;
- le capteur de température extérieure et humidité relative (placé au dessus de la carte) ;
- le capteur de température du sol (placé au niveau du sol) ;
- le pluviomètre ;
- le capteur de lumière.

Les 4 derniers capteurs sont tous déportés. En effet, ils sont placés sur d'autres cartes pour effectuer les mesures aux endroits adéquats.

C'est le module UBC qui envoie les requêtes de mesures via son bus CAN à ce module. Le module MOD_CAPT reçoit ces requêtes, va lire dans ses registres les différentes mesures et envoie celles-ci au module UBC via le même bus CAN. Le module UBC envoie des requêtes toutes les secondes.

La carte possède un microcontrôleur PIC 18F2580-I/SP de chez Microchip, un convertisseur analogique/digital 16 bits, un régulateur de tension de référence pour les mesures analogiques, un driver CAN pour la communication entre le module et l'UBC.

La carte est alimentée en +12V via le bus CAN. Cette tension est ensuite transformée en +5V pour l'alimentation des différents composants.

C'est la première carte que j'ai dû dessiner avec le logiciel Eagle.

Le schéma électrique général de la carte et le dessin de celle-ci se trouvent dans les annexes aux pages VIII et IX.

3.4.3 Explication des différents composants.

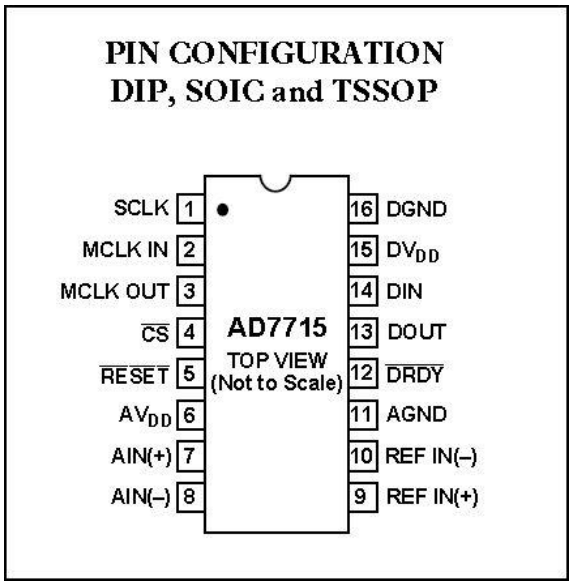
3.4.3.1 Le convertisseur analogique/digital.

Pour des raisons de stabilité de la mesure et surtout de précision, un convertisseur 16 bits a dû être installé entre le capteur de pression (capteur analogique) et le PIC (au lieu d'utiliser le convertisseur 10 bits du PIC) sinon le capteur variait trop brusquement et était fortement perturbé (un simple déplacement d'air avec la main faisait varier la pression atmosphérique→ problème résolu avec le convertisseur).

Ce composant existait déjà auparavant dans une autre application, j'ai donc repris le montage pour éviter de perdre du temps.

Explication d'une piste de solution pour éviter de placer ce convertisseur à la page 55

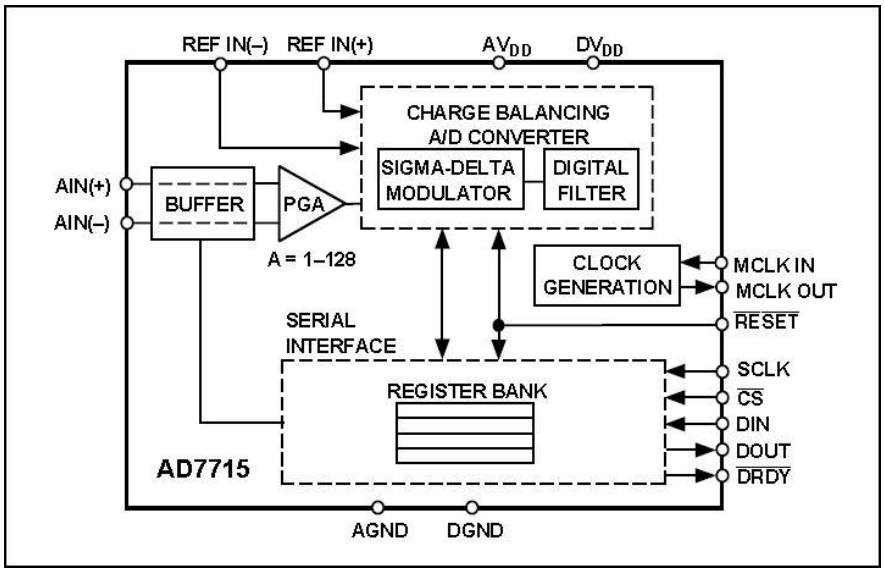
3.4.3.1.1 Description des pins.



- Pin 1: SCLK: C'est une entrée sur laquelle on vient placer une Clock pour accéder aux données du circuit. C'est l'horloge utilisée pour la communication avec le circuit.
- Pin 2: MCLK IN: Dans le cas de cette application, un Crystal de 2.4576MHz est placé entre les pins MCLK IN et MCLK OUT. Ce Crystal rythme le circuit.
- Pin 3: MCLK OUT: Idem Pin 2.
- Pin 4: CS: Lorsque le PIC applique une tension basse (0V), il indique au circuit qu'une commande I²C va être envoyée.
- Pin 5: RESET: Lorsqu'on applique une tension basse (0V), le circuit est reseté.
- Pin 6 : AV_{DD} : Alimentation du convertisseur analogique/digital interne (→$+V_{REF}$).
- Pin 7: AIN(+): Entrée analogique positive. On y raccorde la sortie du capteur de pression.
- Pin 8: AIN (-): Entrée analogique négative. Elle est raccordée à la masse car pas utilisée.
- Pin 9: REF IN (+): Tension de référence positive du circuit. Elle est raccordée au potentiel $+V_{REF.}$
- Pin10: REF IN (-): Tension de référence négative du circuit. Elle est raccordée à la masse.
- Pin 11: AGND: Masse du convertisseur analogique/digital interne.
- Pin 12: DRDY: C'est une sortie permettant d'indiquer qu'une donnée est prête à être lue dans le circuit. Si l'état de cette sortie est à l'état haut, ça nous indique qu'une conversion est en cours.
- Pin 13: DOUT: C'est la Pin de sortie des données (pour une utilisation en communication SPI).
- Pin 14: DIN: C'est la Pin d'entrée des données (pour une utilisation en communication SPI).

- Pin 15: DV_{DD}: Alimentation digitale →Alimentation du circuit (raccordée au potentiel +5V).
- Pin 16: AV_{DD}: C'est le point de référence pour l'alimentation du circuit (raccordée à la masse).

3.4.3.1.2 Diagramme fonctionnel du circuit.



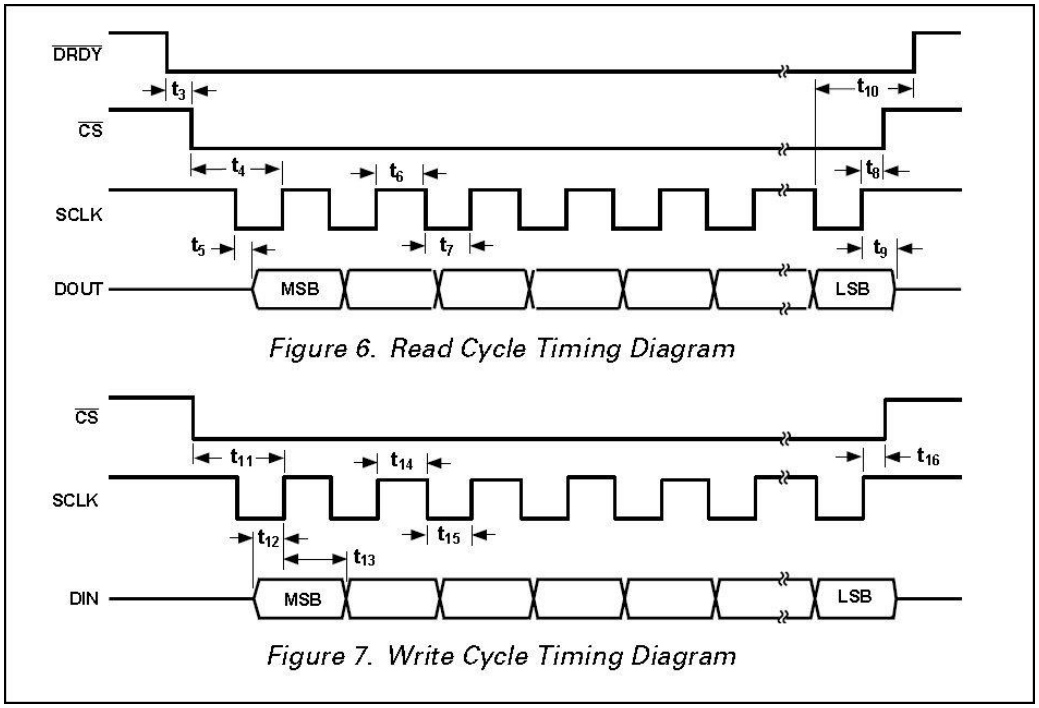
Commençons par expliquer le « Clock Generation ». A ses bornes, on y raccorde le Crystal de 2.4576MHz. Celui-ci va rythmer les calculs et opérations effectués par le circuit.

L'interface série est composé de 4 registres :

- le registre de communication ;
- le registre de réglage ;
- le registre de données ;
- le registre de test.

Avant chaque lecture de données, il faut envoyer le registre de communication au circuit via le bus SPI. Le registre de communication nous informe que dans l'opération qui suit, on effectuera une lecture ou une écriture d'un des registres. Le registre de réglage contient toutes les informations nécessaires au fonctionnement du circuit. Le registre de données contient la valeur de la conversion et le registre de test contient des informations quant à l'état du circuit (si celui-ci fonctionne correctement etc.)
Les entrées analogiques arrivent directement dans un buffer, la valeur analogique peut ensuite être amplifiée (gain de 1 à 128) pour ensuite être traitée et convertie. Une fois la valeur convertie, la donnée est écrite dans le registre de données, la sortie DRDY sera mise à 0 et cette donnée pourra être lue. Pour la conversion, le circuit utilise une tension de référence qui est toujours fixe (intérêt du régulateur de tension de référence). Ainsi, la précision de la conversion est assurée.

3.4.3.1.3 Cycle de lecture et d'écriture dans le circuit.



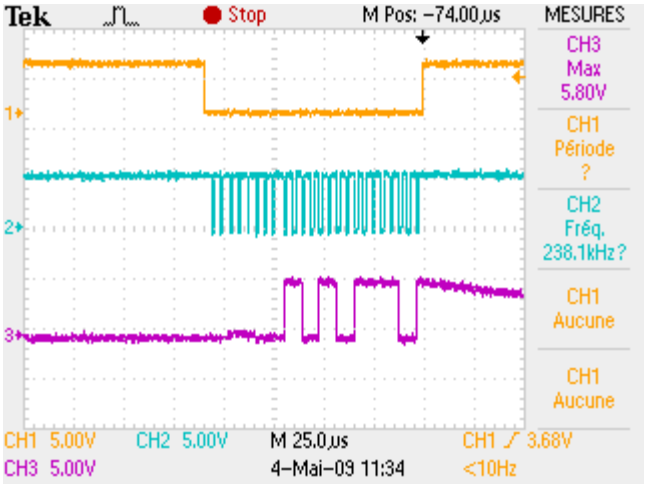


Figure 18 Cycle d'écriture (vue oscilloscope)

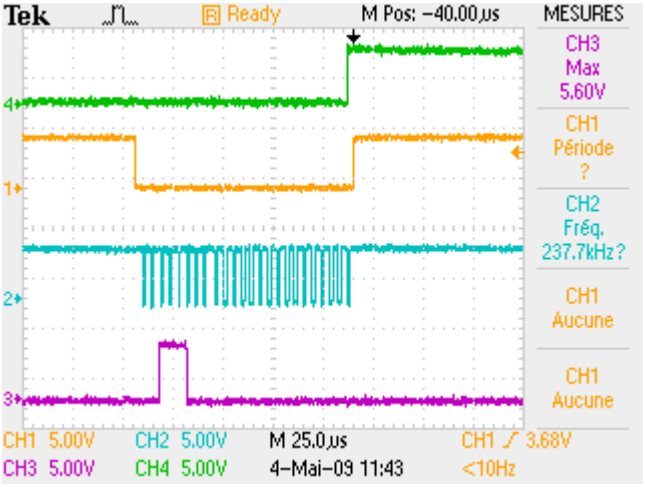


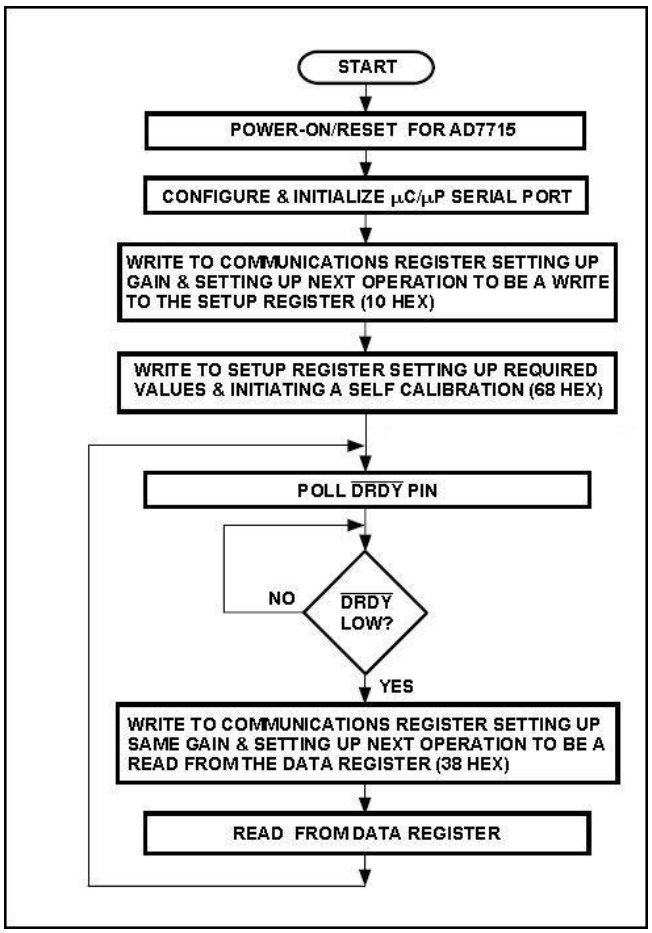
Figure 17 Cycle de lecture (vue oscilloscope)

Pour effectuer une écriture dans l'un des registres, on active le circuit (CS), ensuite on envoie la Clock et enfin la donnée à écrire. Une fois toutes les données envoyées, on arrête la Clock (signal mis à +5V) et on désactive le circuit.

Pour effectuer une lecture, il faut attendre que le circuit soit prêt et que la donnée puisse être lue. Pour cela, il faut que le signal DRDY passe à 0V. Une fois à ce niveau, on peut activer le circuit, envoyer la Clock et récupérer la donnée dans le registre.

La communication entre le microcontrôleur et le convertisseur est assurée par un SPI soft.

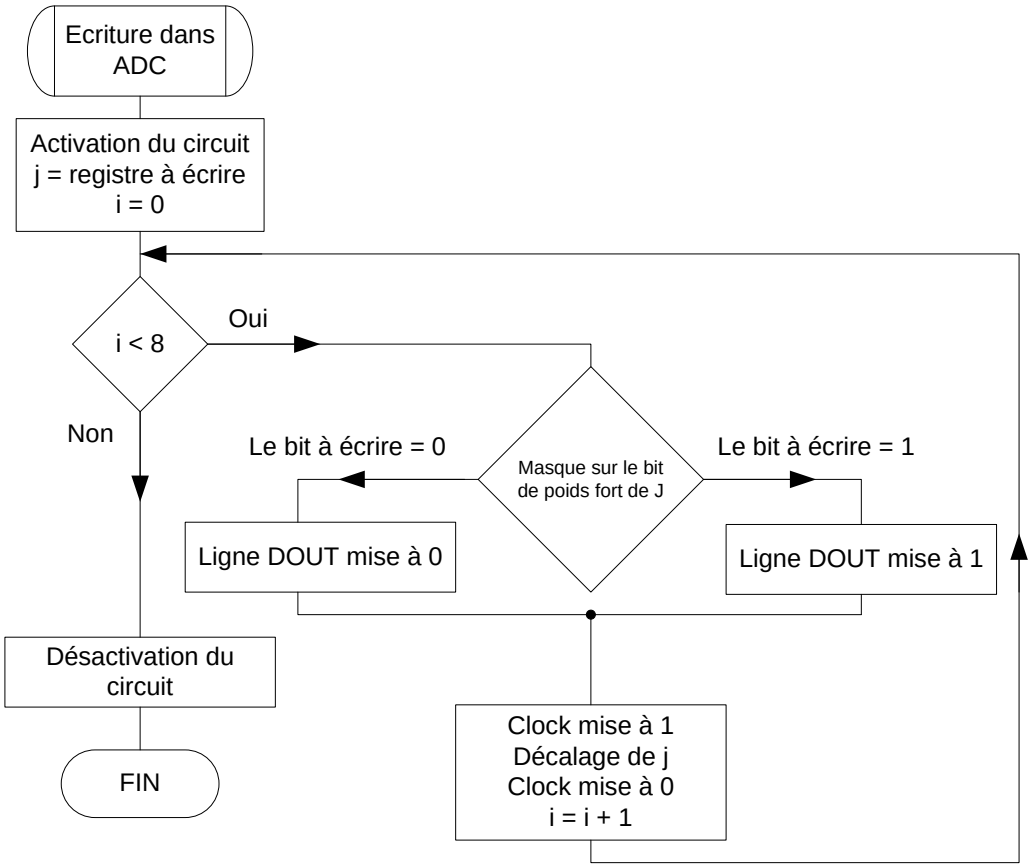
Voici un diagramme à respecter pour effectuer la communication avec le circuit :



Premièrement, le circuit est alimenté et configuré. Deuxièmement, on écrit dans le registre de communication le gain de l'ampli et tous les réglages nécessaires au fonctionnement. On lui indique que l'opération suivante sera une écriture dans le registre de réglages. Troisièmement, on initialise le registre de réglages aux valeurs adéquates.

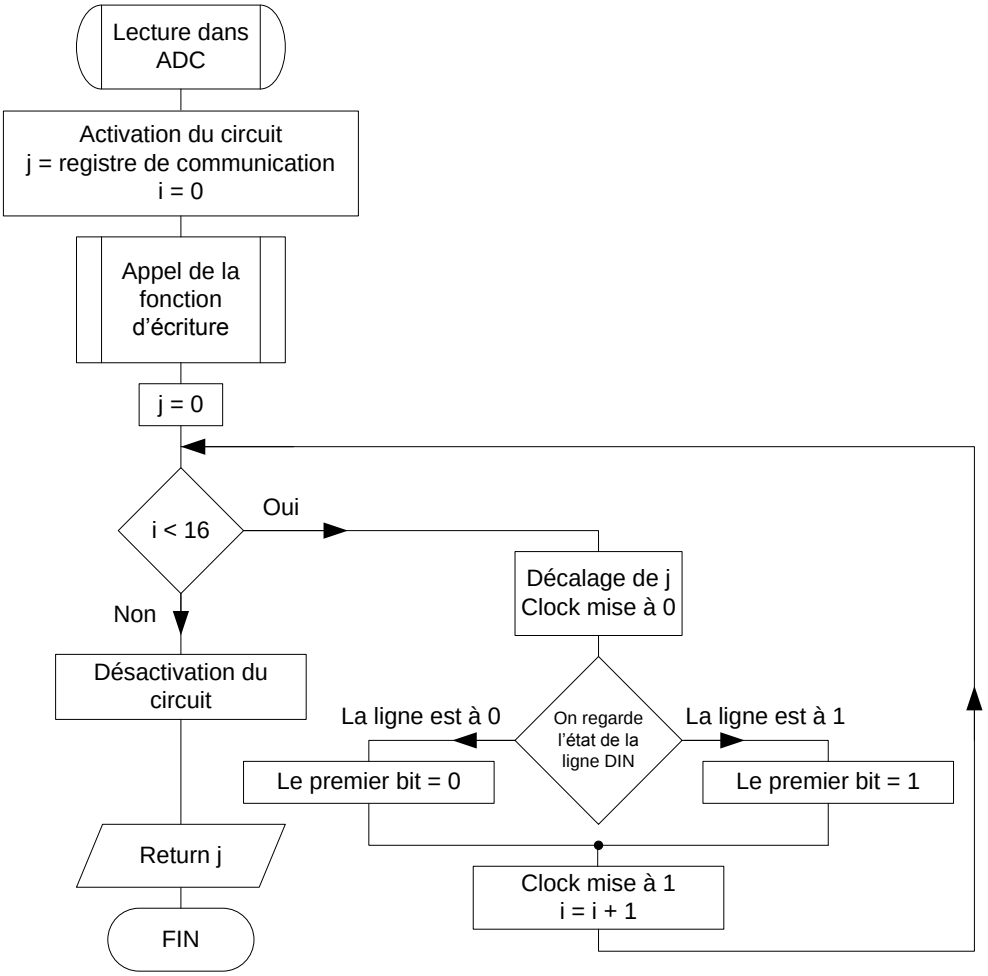
Une fois ces trois opérations effectuées, il y aura un bouclage → On regarde l'état de la Pin DRDY. Si elle est à l'état bas, on passe à l'étape suivante. Dans le cas contraire, on revient en arrière et on regarde l'état de la Pin jusqu'à ce que celle-ci passe à l'état bas. Une fois à l'état bas, on écrit dans le registre de communication que l'on va effectuer une lecture du registre de données (qui contient la donnée convertie étant donné que DRDY est passé à 0). On lit ensuite la donnée qui sera récupérée et envoyée par SPI au microcontrôleur. Après avoir lu la donnée, on scrute à nouveau l'état de la Pin DRDY (pour voir si une valeur a été convertie).

Ordinogrammes de la logique de communication avec le convertisseur :



Explication de la procédure d'écriture :

- On active le circuit.
- On place dans « j » la valeur du registre à devoir écrire. Par exemple, j = 0b11001011. Chaque 1 et 0 correspondent à un bit de données qui sera envoyé.
- On regarde si « i » est plus petit que 8. En effet, chaque registre est composé de 8 bits, sauf le registre de données qui en compte 16. Dans le registre de données, on n'écrit rien, on le lit uniquement !
- Si le résultat du test est « oui », on passe au test suivant, si le résultat est « non », on désactive le circuit → l'écriture est terminée.
- On teste l'état du bit de poids fort du registre à envoyer. Si le résultat est un 1, on place la ligne de données à +5V. Si le résultat est un 0, on place la ligne de données à 0V
- On active le Clock, on décale ensuite « j » de 1 pour ne plus envoyer le même bit au cycle suivant. On fixe la Clock à 0 et on incrémente « i » de 1 pour dire que le bit a été envoyé et que l'on passe au suivant.
- Une fois que « i » atteint 8, les 8 bits de données ont bien été envoyés, l'écriture est ainsi terminée.

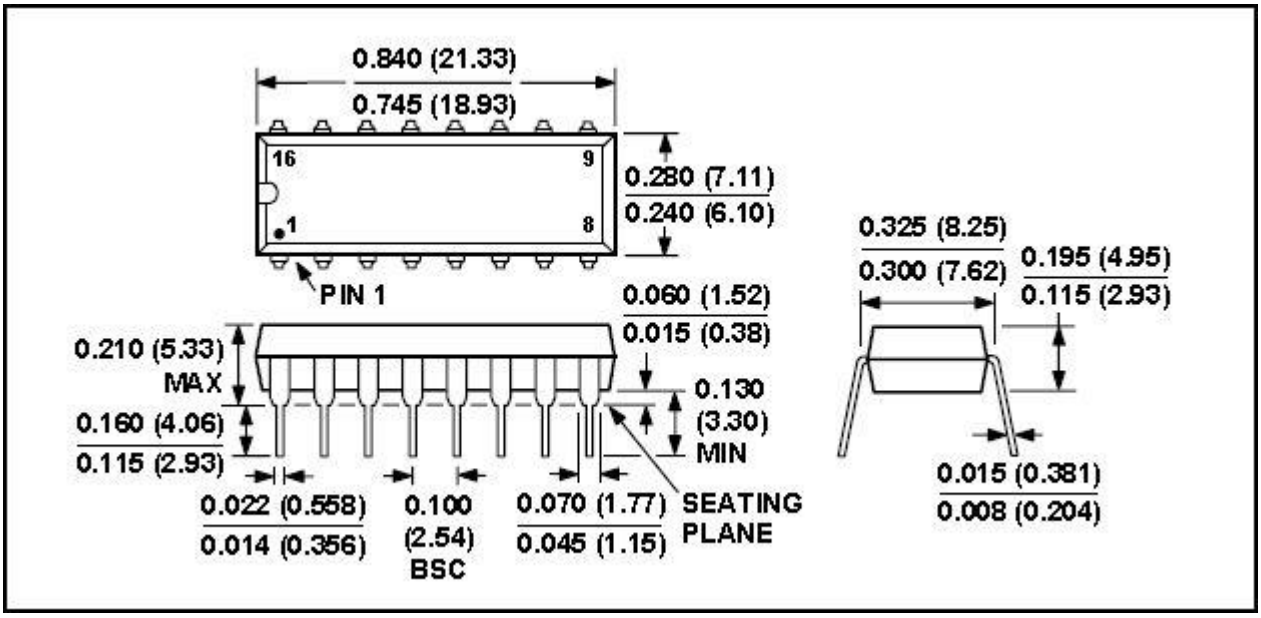


Explication de la procédure de lecture d'une donnée :

- On active le circuit.
- On place dans « j » la valeur du registre à devoir écrire. En effet, avant chaque lecture de données, on doit écrire le registre de communication dans le circuit. On appelle ensuite la fonction d'écriture pour effectuer cette opération !
- On initialise « j » à 0. Il contiendra la valeur de la donnée.
- On regarde si « i » est plus petit que 16. Le registre de données faisant 16 bits, on doit effectuer ce test 16 fois
- Si le résultat est un « oui », on décale « j » de 1 et on met le signal de Clock à 0.
- On regarde ensuite l'état de l'entrée DIN. Si on a un état haut, c'est que le convertisseur nous envoie un « 1 ». Si par contre on a un état bas, alors il nous envoie un « 0 ». On met le signal de Clock à 1 et on incrémente « i » de 1.
- Une fois la boucle effectuée 16 fois, on désactive le circuit et on retourne la valeur lue sur la Pin DIN.

Voici comment j'ai effectué un SPI soft (par programmation) pour ainsi communiquer avec le convertisseur (lecture des données, écriture des registres etc.).

3.4.3.1.4 Dimensions du composant.

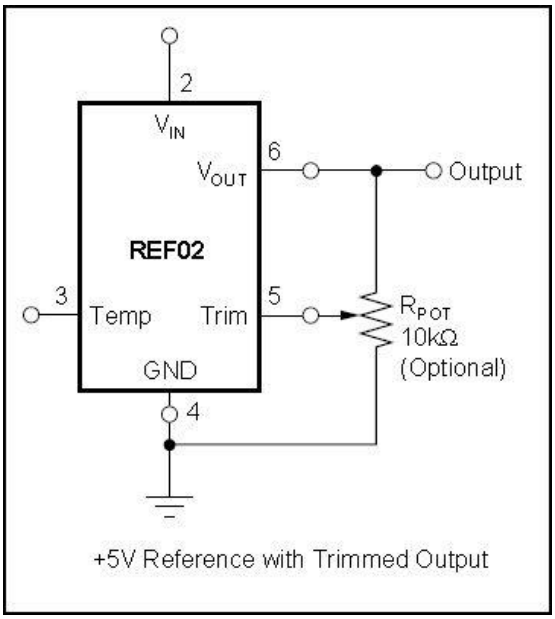


Les dimensions entre parenthèses représentent les dimensions du circuit en millimètres.

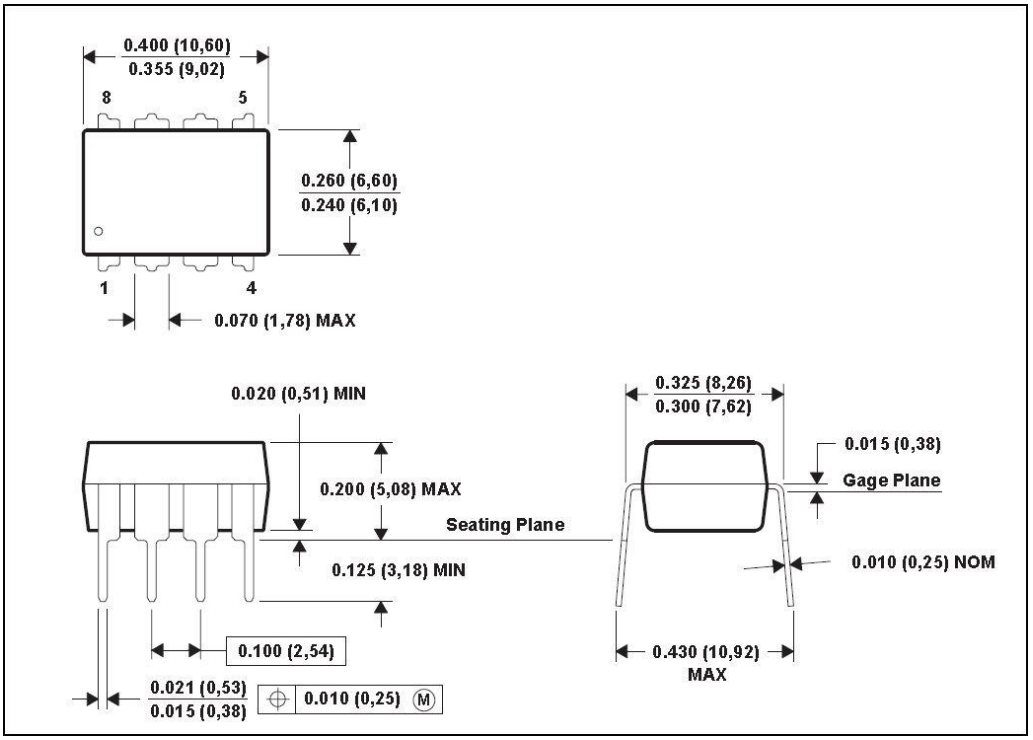
3.4.3.2 Le régulateur de tension de référence.

Pour effectuer des mesures analogiques précises, il a fallu placer un régulateur de tension permettant de garder sa tension de sortie fixe, sans aucune fluctuation. En effet, si on alimente un PIC avec une tension variant de 4.9V à 5.1V, le fonctionnement de celui-ci restera inchangé car le reset du PIC ne se fait qu'avec de forts changements de la tension d'alimentation. Si la tension d'alimentation d'un capteur ou bien la valeur de la tension de comparaison varie, les mesures seront fausses. J'ai donc choisi le circuit REF02 de chez Texas Instrument qui fournit une tension de +5V (± 0.2% Max soit ±0.01V Max) avec précision!

3.4.3.2.1 Raccordement.



3.4.3.2.2 Dimensions.



Les dimensions entre parenthèses sont en millimètres.

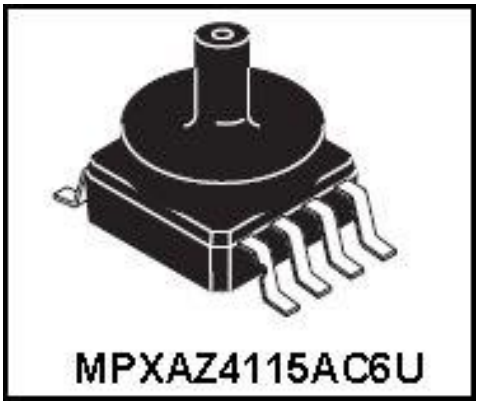
3.4.3.2.3 Utilisations.

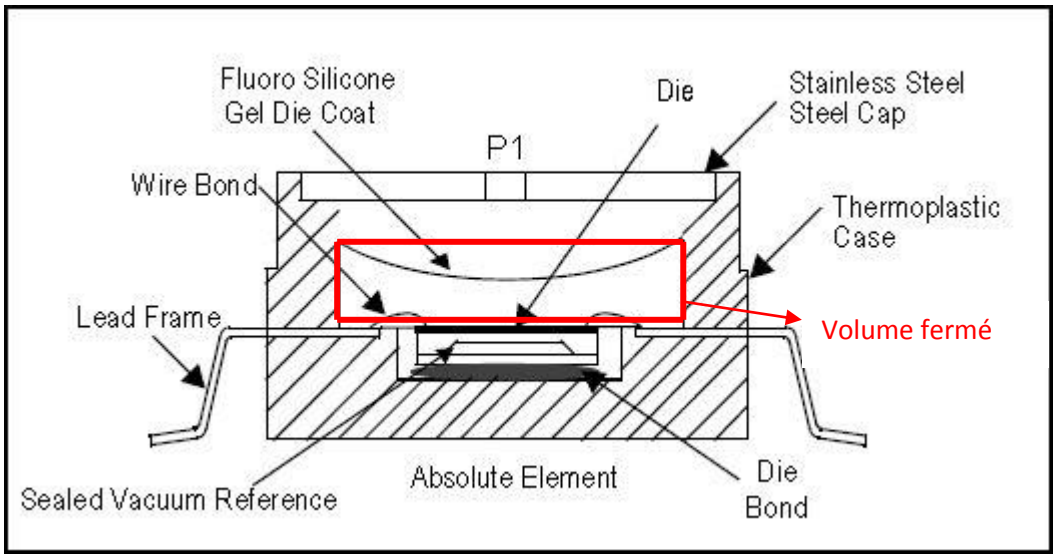
La tension générée par le circuit est utilisée pour :

- l'alimentation du capteur de pression ;
- servir de seuil de référence au convertisseur analogique/digital ;
- l'alimentation du capteur de lumière ;
- servir de seuil de référence au microcontrôleur PIC.

3.4.3.3 Le capteur de pression absolue.

3.4.3.3.1 Détails du composant.

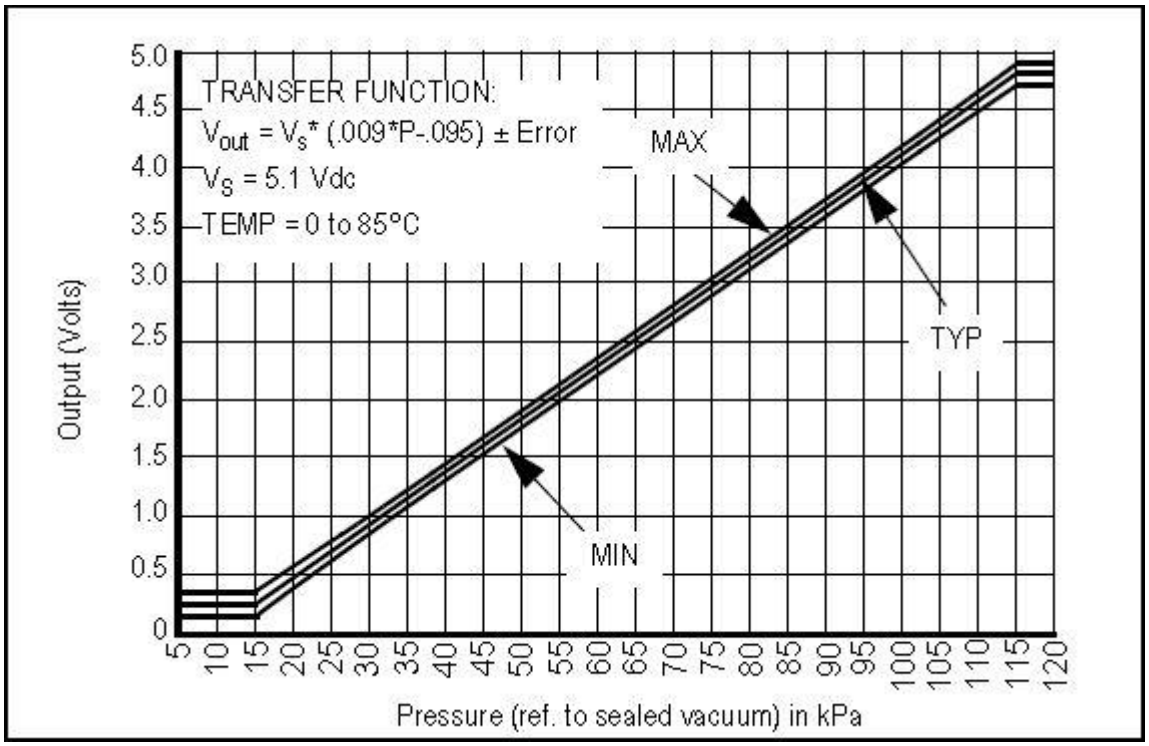




Ce capteur permet de mesurer la pression absolue de l'air. Il peut être utilisé dans des applications telles que les altimètres et baromètres. Il est composé d'un ampli opérationnel et tout un équipement permettant d'avoir une sortie analogique de haut niveau (directement utilisable 0→+5V). Il possède également une compensation de température.

Le principe de fonctionnement de ce capteur est simple. La pression P1 est appliquée sur la surface appelée « Gel Die Coat » qui est en réalité une membrane formant un volume fermé (volume intercalé entre celle-ci et la carcasse du composant). Cette membrane va être compressée et la pression à l'intérieur de ce volume va varier. La pression variant, la lame « Die » va bouger et se rapprocher/s'éloigner du composant semi conducteur placé en dessous. Ce composant semi-conducteur va subir cette variation et nous donnera une image de la pression P1 (l'image étant une sortie analogique qui est convertie).

3.4.3.3.2 Caractéristique de sortie.



On voit que la plage de travail du capteur va de 15kPa à 115kPa.

Notre capteur va donc nous délivrer une tension allant de 0 à ±5V. C'est une valeur analogique ! Cette tension est injectée dans le convertisseur analogique/digital pour des raisons expliquées précédemment.

3.4.3.3.3 Calcul de la pression atmosphérique.

Après avoir expliqué le travail du capteur ainsi que de son convertisseur, on peut alors montrer comment le calcul de la pression est effectué.

Relation entre la tension de sortie et la pression atmosphérique :

$$V_{OUT} = V_{REF} \times ((0.009 \times pression) - 0.095) \pm erreur$$

Dans l'application, le capteur fournit une tension (V_{OUT}), qui est convertie. Pour trouver la valeur de la pression, voici les calculs devant être effectués :

$$V_{OUT} = Valeur\ convertie \times \frac{5}{65535}$$

En effet, la tension de sortie est égale à la valeur convertie par l'A/D (valeur comprise entre 0 et 65535) que l'on adapte à l'échelle. 5 étant la valeur de la tension de référence et 65535 étant égal à 2^{16} (16 car convertisseur 16 bits). Le fait de diviser 5 par 65535 nous donne la résolution du convertisseur, c'est-à-dire le nombre de volts correspondant à un point. Si on prend la valeur d'un point que l'on multiplie par la résolution, on obtient la tension de sortie.

Une fois la valeur de la tension retrouvée, il faut adapter la formule de départ pour en sortir la valeur de la pression :

$$Pression = \frac{\frac{V_{OUT}}{V_{REF}} + 0.095}{0.009} \times 10$$

La pression est multipliée par 10 car la formule initiale travaille avec une pression exprimée en kPa. Or, l'unité conventionnelle dans les applications météorologiques est le hPa.

En conclusion, le capteur est capable de mesurer une pression allant de 150hPa à 1150hPa. Or, dans le monde, la pression atmosphérique la plus basse mesurée est de 807hPa et la plus haute est de 1086hPa. Ce qui fait une sortie en tension au niveau du capteur allant de ±3.2V à ±4.2V. On voit que la plage de variation est assez mince, c'est pourquoi nous avons placé un convertisseur 16 bits (résolution : 5/65535 = 76.29µV/point) au lieu du convertisseur 10 bits intégré dans le PIC (résolution : 5/1024 = 4.88mV/point).

On peut se poser une question quant à l'utilisation d'un convertisseur analogique/digital externe. En effet, celui-ci est un composant assez onéreux (13€/pièce) alors que le PIC possède aussi un convertisseur de ce type. Ce convertisseur externe a fait ses preuves dans une autre application. Selon les consignes qui m'ont été données, je l'ai donc intégré à ma carte.

Une piste de solution pour ne pas utiliser ce convertisseur serait d'utiliser celui du PIC. On sait que le capteur de pression atmosphérique donnera une tension comprise entre 3.2V et 4.2V, ce qui fait une plage de variation de 1V. En appliquant aux bornes $-V_{Ref}$ et $+V_{Ref}$ du PIC les tensions de +3V et +5V, on fixerait une plage de variation maximale de 2V. En sachant que le convertisseur a une résolution de 10 bits (1024 points), on peut en déduire ceci : 2 / 1024 = 1.93mV/point.

On peut effectuer un autre calcul, pour montrer qu'un convertisseur 10 bits est amplement suffisant.

$$Plage\ de\ variation\ de\ pression\ maximale = 1086 - 807 = 279\ hPa$$

$$\frac{279}{1024} = 0.272\ hPa$$

Si on augmente d'un point au niveau du convertisseur, la pression augmentera de 0.272hPa.

➔ Un convertisseur 10bits (celui du PIC) convient pour effectuer cette tâche mais par manque de temps, je n'ai pas su effectuer ces tests complémentaires afin d'apporter des modifications à ce capteur.

Notons également que dans l'autre application où le capteur et son convertisseur sont utilisés, la plage de variation de pression est bien plus grande. En effet, ce n'est pas une mesure de pression atmosphérique mais une mesure de pression dans une canalisation et donc la plage est plus grande !

3.4.3.4 Le microcontrôleur PIC 18F2580-I/SP.

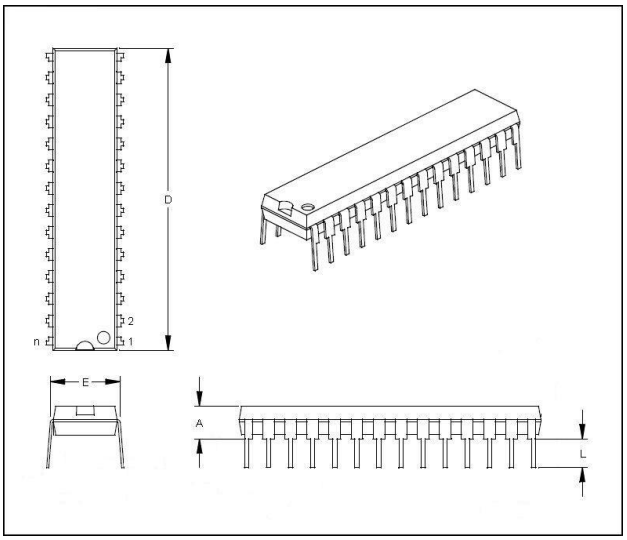
Il a fallut placer un microcontrôleur sur cette carte pour gérer toutes les mesures, les envois CAN et les communications avec les capteurs. Il a pour but de « désengorger » le travail effectué par le PIC placé sur la carte UBC. Le PIC placé sur la carte MOD_CAPT s'occupera du traitement de toutes les mesures, effectuera les calculs nécessaires à la mise en forme de celles-ci et enverra les données traitées à la carte UBC. On obtient ainsi des modules totalement autonomes et ne dépendant pas d'autres modules (si ce n'est que pour communiquer entre eux).

Petite indication concernant le choix d'un microcontrôleur. Voici les critères de sélection :

- le nombre de pins dont on a besoin ;
- la vitesse de travail du microcontrôleur ;
- les bus utilisés (SPI, I2C, CAN,…) ;
- la mémoire dite « programme » (pour sauvegarder le programme à exécuter) ;
- la mémoire RAM (pour les calculs que le PIC doit effectuer) ;
- la mémoire EEPROM (pour la sauvegarde de certaines données si l'alimentation du PIC vient à se couper).

Mon choix s'est tourné vers un PIC 18F2580-I/SP de 28 Pins. Il permet de gérer des communications CAN, SPI et I2C et possède les mémoires suffisantes pour obtenir un fonctionnement satisfaisant.

4.3.5.4.1 Dimensions du composant :

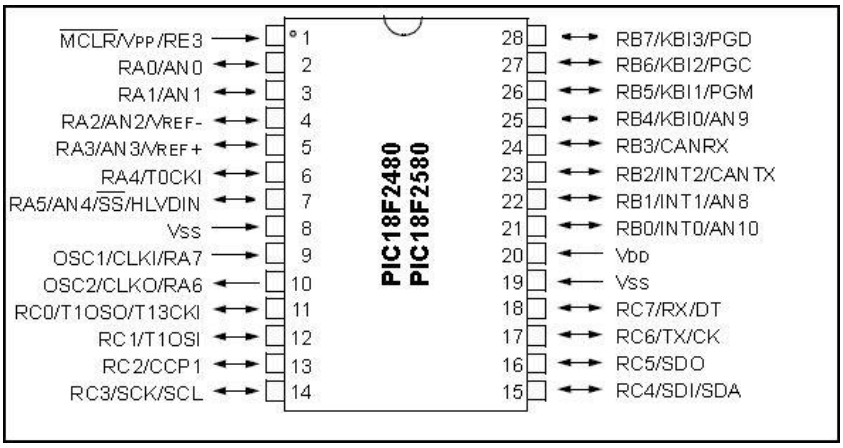


D = Longueur = 34.6 mm E = Largeur = 7.8 mm

A = Epaisseur du circuit = 3.8mm L = Longueur Pin = 3.3mm

e = Ecartement entre Pin : 2.5mm

3.4.3.4.2 Brochage du PIC 18F2580 I/SP.

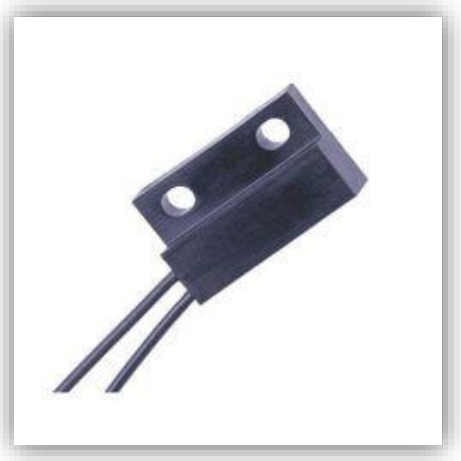


3.4.3.4.3 Raccordement de l'oscillateur au PIC :

L'oscillateur est raccordé de la même façon que pour le PIC 18F4680-I/P. *(Voire point 3.1.3.3.3 Raccordement de l'oscillateur au PIC).*

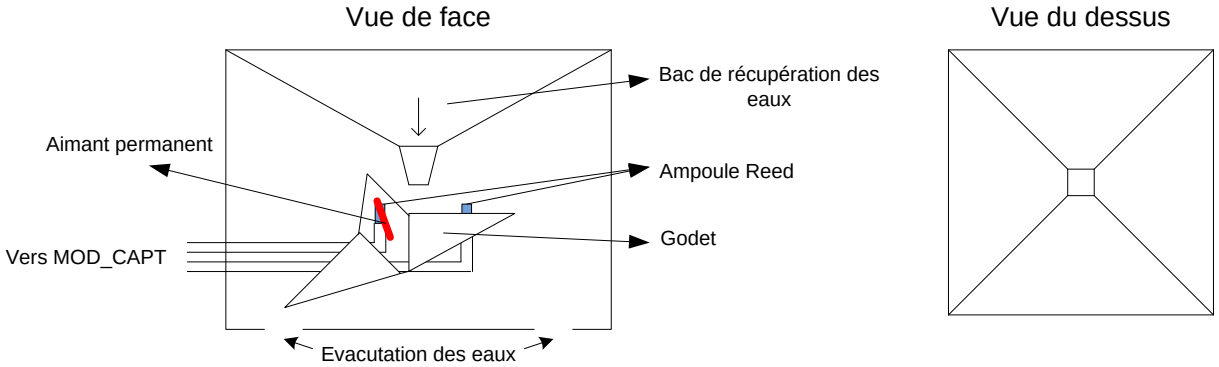
3.4.3.5 Le pluviomètre.

La partie « électronique » du pluviomètre est constituée d'une simple ampoule Reed.



Une ampoule Reed est un interrupteur qui établit ou coupe une connexion électrique en fonction ou non de la présence d'un champ magnétique.
L'ampoule Reed est généralement constituée d'une ampoule de verre protectrice contenant une atmosphère non oxydante et deux contacts souples. Ces contacts son magnétisables et élastiques (à base de fer doux par exemple).
En présence d'un champ magnétique, les contacts s'aimantent par influence et sont attirés l'un par l'autre. Ils se rapprochent et se touchent, établissant ainsi le contact → le courant peut circuler.
Lorsque le champ magnétique cesse, l'aimantation s'arrête aussi et l'élasticité des contacts les écarte, ouvrant ainsi le contact → le courant ne peut plus circuler.

Un aimant permanent est placé sur la partie mobile du pluviomètre. Celui-ci va modifier l'état de l'interrupteur en passant devant, transmettant ainsi une information sur la position de l'aimant.



L'eau de pluie est récoltée par le bac de récupération des eaux. La surface du bac est bien déterminée. Ensuite, elle tombe dans l'un des deux godets. Une fois le godet rempli, grâce à son poids, il va basculer pour se vider tout en entrainant l'aimant permanent avec lui. L'aimant arrivant devant la seconde ampoule Reed, le contact se ferme → un godet a été rempli. L'eau contenue dans le godet s'évacue par le bas lorsque celui-ci bascule.

La mesure de la quantité de pluie tombée est simple car on connaît la surface de récupération et le volume d'un godet rempli. Il suffit de définir un temps pendant lequel on compte le nombre de godets remplis et on obtient le nombre de litres de pluie tombée par m² pendant ce temps.

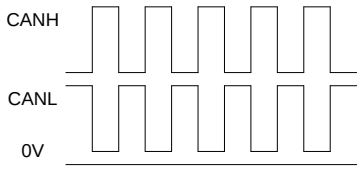
Les deux ampoules Reed sont connectées sur des Pins interruptibles du microcontrôleur. On compte le nombre d'interruptions (nombre de fronts montant) et on obtient le nombre de godets remplis.

3.4.3.6 Les différents connecteurs de la carte MOD_CAP.

3.4.3.6.1 Le connecteur principal.

La carte dispose d'un connecteur principal (connecteur à vis pour des facilités de câblage). Ce connecteur est utilisé par le bus CAN. Pour rappel, le bus CAN est composé de 4 fils :

- Un fil d'alimentation → alimentation de la carte MOD_CAPT.
- Un fil de masse.
- CANH et CANL qui véhiculent un signal de données.



C'est une communication différentielle ou CANH est le « + » et CANL est le «-». De ce fait, un parasite se propageant sur la ligne, se propagera sur les deux signaux → il sera supprimé à l'arrivée car on fait la différence des deux signaux !

3.4.3.6.2 Les connecteurs pour les capteurs déportés.

Tous les capteurs ne sont pas intégrés sur la carte car les mesures ne sont pas effectuées au même endroit. Il a donc fallu placer différents connecteurs pour pourvoir déporter certains capteurs et les relier par câble à la carte MOD_CAPT.

Il s'agit de connecteurs MOLEX :

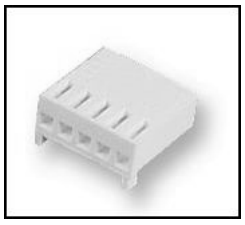


Figure 20 Connecteur femelle

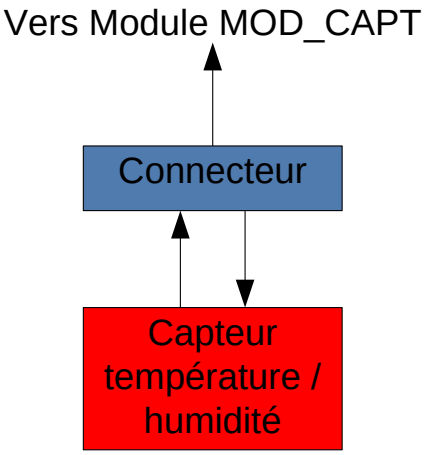
Figure 19 Connecteur mâle

Le connecteur femelle est placé sur le câble et le connecteur mâle sur le PCB. Ces connecteurs existent en différentes configurations (de 2 à 15 pins). Voici les caractéristiques des connecteurs utilisés :

- Ecartement entre pins : 2.54mm (paramètre utilisé pour le dessin de la carte).
- Diamètre des trous sur le PCB : 1.02mm (paramètre utilisé pour le dessin de la carte).
- Courant maximum : 4A.
- Tension maximum : 500V AC.

3.5 Le module TH_SENSOR

3.5.1 Schéma bloc de la carte.



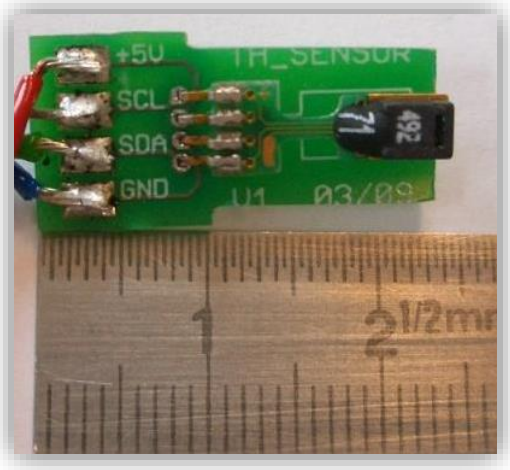


Figure 21 Module TH_SENSOR

3.5.2 Rôle de la carte.

La carte TH_SENSOR permet d'effectuer une mesure de la température extérieure et une mesure du taux d'humidité contenu dans l'air. C'est un des modules déportés de la carte MOD_CAPT, il vient s'y connecter via un des connecteurs Molex. Cette carte a été conçue pour accueillir le capteur SHT71. Ce capteur est assez fragile et pour des raisons de facilité de placement, j'ai dessiné un petit PCB permettant de renforcer le capteur et de faciliter la connexion avec la carte MOD_CAPT.

C'est la carte MOD_CAPT qui émet les demandes de mesures à la carte TH_SENSOR afin de connaître la température et le taux d'humidité. La carte TH_SENSOR répond à ces requêtes en envoyant les mesures instantanées !

Quatre fils partent du module MOD_CAPT vers le module TH_SENSOR :

- L'alimentation pour le capteur (+5V).
- La masse du circuit.
- Le signal d'horloge pour la communication.
- Le signal de données pour le transfert des données dans le capteur et hors du capteur (signal bidirectionnel).

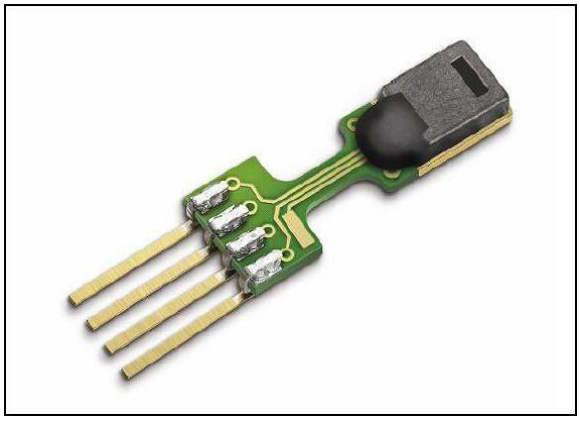
C'est la troisième carte que j'ai dû dessiner avec le logiciel Eagle.

Le schéma électrique général de la carte et le dessin de celle-ci se trouvent dans les annexes aux pages X et XI.

3.5.3 Explication des différents composants.

3.5.3.1 Généralités.

Le composant principal de cette carte est le capteur de température/humidité relative SHT71. Ce capteur est déjà calibré et possède une sortie digitale. Il contient un amplificateur et un convertisseur analogique/digital.



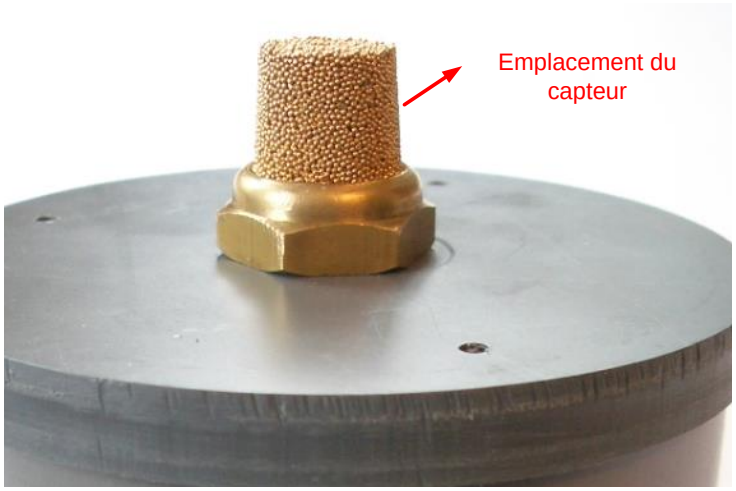
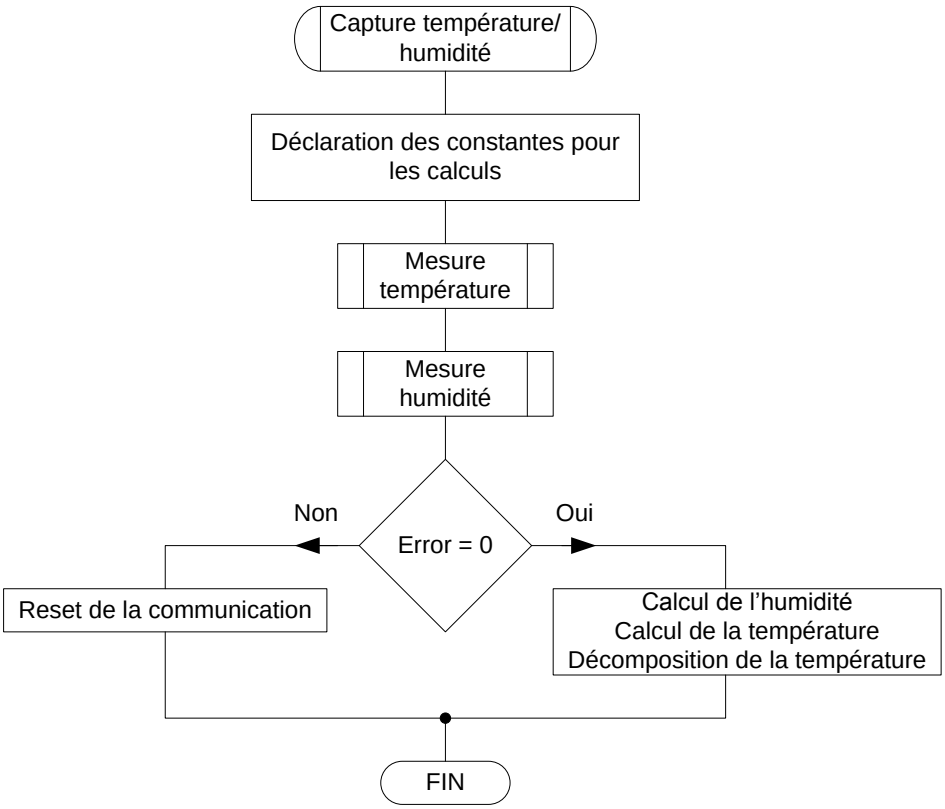


Figure 22 Capteur SHT71

On communique avec le capteur via un I^2C soft.

3.5.3.2 Traitement des données.

Voici l'organigramme du processus de lecture d'une donnée dans le capteur :



Dans la fonction de capture, on fait appel à une fonction de mesure. Cette fonction fait elle-même appel à une fonction qui a pour but d'écrire la donnée mesurée (soit la température, soit l'humidité).

On obtient ainsi deux valeurs de mesures à l'état « brut », il faut effectuer un traitement sur ces données.

Tout d'abord, on calcule la température utilisée dans la compensation de l'humidité :

$$Tempértaure_{Compensation} = d_1 + d_2 \times SO_T$$

La variable « Température $_{Compensation}$ » est utilisée dans le calcul de l'humidité. Les variables d_1 et d_2 sont trouvées dans les tables données avec le capteur. Le capteur est alimenté en +5V et la résolution de celui-ci est de 14 bits (pour une mesure de la température)

→ d_1 = -40.1 et d_2 = 0.01.

SO_T étant la valeur de la température mesurée (valeur à l'état « brut »).

En ayant effectué ce calcul, on peut calculer l'humidité. Le calcul de l'humidité linéaire est d'abord effectué :

$$Humidité\ lin = C_1 + C_2 \times SO_{RH} + C_3 \times {SO_{RH}}^2$$

La variable « humidité linéaire » est utilisée dans le calcul de l'humidité relative. Nous voulons une résolution de 12 bits pour la mesure d'humidité. Selon les tables, nous aurons donc C_1 = -4.0, C_2 = 0.0405, C_3 = -0.0000028.

SO_{RH} étant la valeur de l'humidité mesurée.

On peut ensuite calculer l'humidité relative en tenant compte de la température de l'air et de l'humidité linéaire :

$$Humidité\ relative\ = \left(Température_{\ Compensation} - 25\right) \times (t_1 + t_2 \times SO_{RH}) + humidité\ lin$$

Le calcul de l'humidité étant effectué, on effectue à nouveau le calcul de la température :

$$Température\ = \left(\frac{SO_T}{D_4}\right) - D_3$$

Avec D_3 = 401 et D_4 = 10. *(Transformation de la formule initiale)*.

Après ces différents calculs, une fonction de décomposition de la température est appelée. Cette fonction permet de décomposer la température en deux parties :

- Une partie entière et une partie décimale.

Cette fonction permet de faciliter les échanges de données et de ne pas utiliser de nombres à virgules (car le PIC ne fonctionne pas correctement avec le format de données « float »).

3.5.3.3 Informations complémentaires.

Il est possible de calculer le point de rosée grâce aux deux mesures effectuées.

Physiquement, le point de rosée de l'air représente la température à laquelle l'air devient saturé en vapeur d'eau. C'est le phénomène de condensation, qui survient lorsque le point de rosée est atteint, qui crée les nuages, la brume et la rosée. Lorsque la température augmente, la capacité hygrométrique augmente, et ce inversement. Plus il fait froid, moins l'air sera chargé en humidité.

Lorsque la température tombe sous le point de congélation, l'air peut devenir saturé par rapport à la glace, ce qui va donner la gelée blanche. Dans ce cas, la température de saturation est appelée point de givrage, ce qui mène la vapeur d'eau à se déposer plus généralement sous forme solide que liquide sous le point de condensation (phénomène apparaissant en hiver sur le pare-brise d'une voiture par exemple).

En fonction des mesures effectuées, voici la formule à appliquer pour obtenir le point de rosée :

$$Point\ de\ rosée = \ T_n \times \frac{ln\left(\frac{RH}{100\%}\right) + \frac{m \times T}{T_n + T}}{m - ln\left(\frac{RH}{100\%}\right) - \frac{m \times T}{T_n + T}}$$

Pour des températures positives :

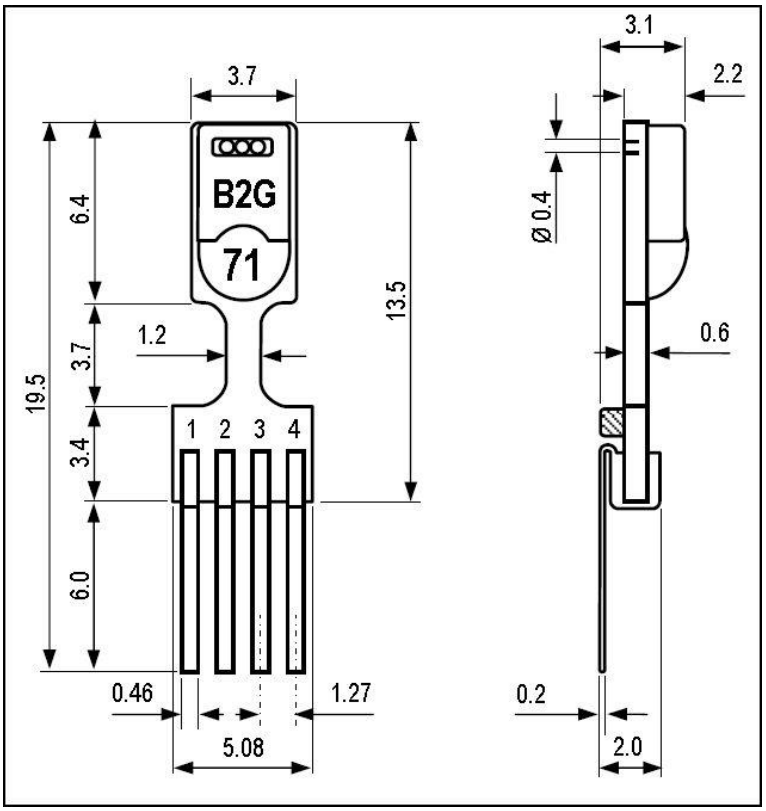
- T_n = 243.12
- m = 17.62

Pour des températures négatives :

- T_n = 272.62
- m = 22.46

Avec T égal à la température de l'air et RH au taux d'humidité.

3.5.3.4 Dimensions du composant.

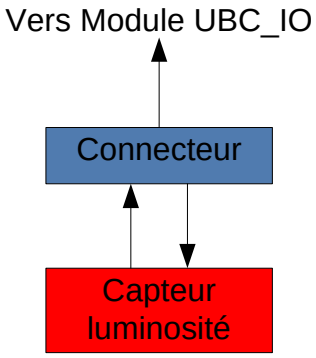


Description des Pins :

- Pin 1 : SCK : C'est l'horloge qui va synchroniser les communications.
- Pin 2 : V_{DD} : Tension d'alimentation du capteur.
- Pin 3 : GND : Masse du circuit.
- Pin 4 : DATA : Signal de données.

3.6 Le module LIGHT_SENSOR

3.6.1 Schéma bloc de la carte.



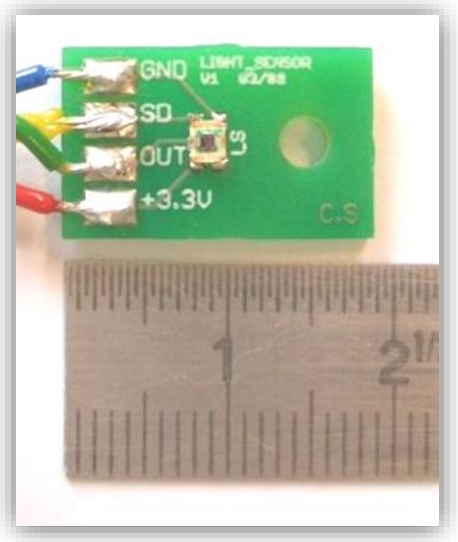


Figure 23 Module LIGHT_SENSOR

3.6.2 Rôle de la carte.

La carte LIGHT_SENSOR permet d'effectuer une mesure du taux d'ensoleillement. C'est un module déporté du module MOD_CAPT, il vient s'y connecter via un des connecteurs Molex. Cette carte a été conçue pour accueillir le capteur APDS-9007 de chez AVAGO. Ce capteur est assez fragile et pour des raisons de facilité de placement (très petite taille), j'ai dessiné un petit PCB permettant de renforcer le capteur et de faciliter la connexion avec le module MOD_CAPT.

C'est la carte MOD_CAPT qui émet les demandes de mesures à la carte LIGHT_SENSOR afin de connaître le taux d'ensoleillement. La carte LIGHT_SENSOR répond à ces requêtes en envoyant les mesures instantanées !

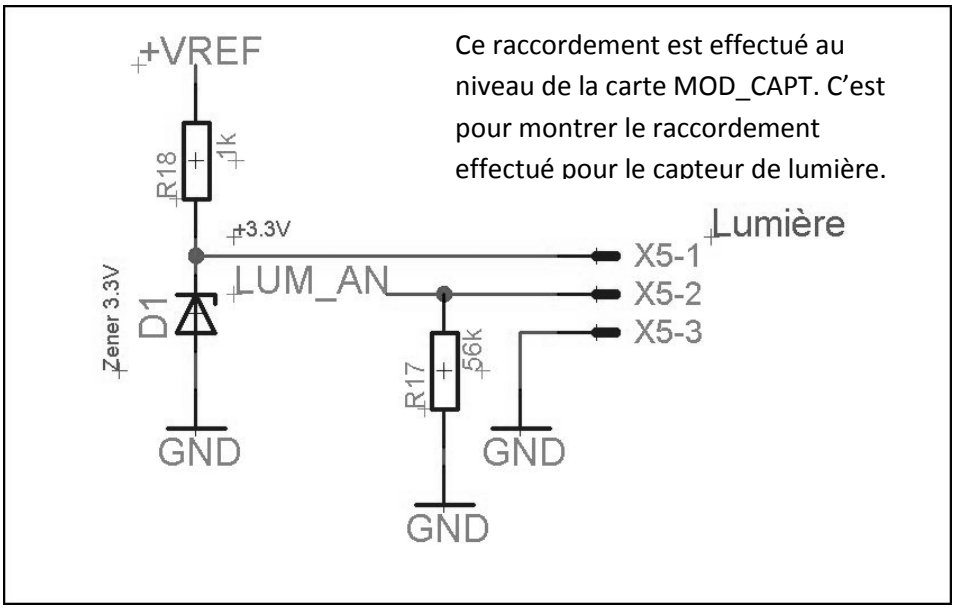
Quatre fils partent du module MOD_CAPT vers le module TH_SENSOR :

- l'alimentation pour le capteur (+3.3V) ;
- la masse du circuit ;
- la sortie analogique du capteur ;
- Un signal « Shutdown ». Le signal SD a été placé au cas où il serait utile dans une autre application mais ce signal n'est pas utilisé ici.

C'est la quatrième carte que j'ai dû dessiner avec le logiciel Eagle.
Le schéma électrique général de la carte et le dessin de celle-ci se trouvent dans les annexes aux pages XII et XIII.

3.6.3 Explication des différents composants.

3.6.3.1 Schéma de raccordement.



Pour éviter de placer un régulateur de 3.3V (nécessaire au fonctionnement du capteur de lumière) sur la carte MOD_CAPT, une diode Zener de 3.3V a été placée. On obtient un montage en forme de « pont diviseur » alimenté par la tension de référence (car capteur analogique → besoin d'une tension stable !).

Le capteur de lumière a besoin d'un courant de 230µA pour être alimenté correctement. En plaçant une résistance de 1kΩ, le courant maximum passant dans la résistance = (5 – 3.3)/1000 = 1.7mA.
→ 1.7 -0.23 = 1.47mA → courant suffisant pour que la diode fonctionne correctement et nous donne une tension de 3.3V.

On voit que la sortie du capteur est chargée par une résistance de 56kΩ. La valeur de la résistance est déterminée par les courbes données par le constructeur :

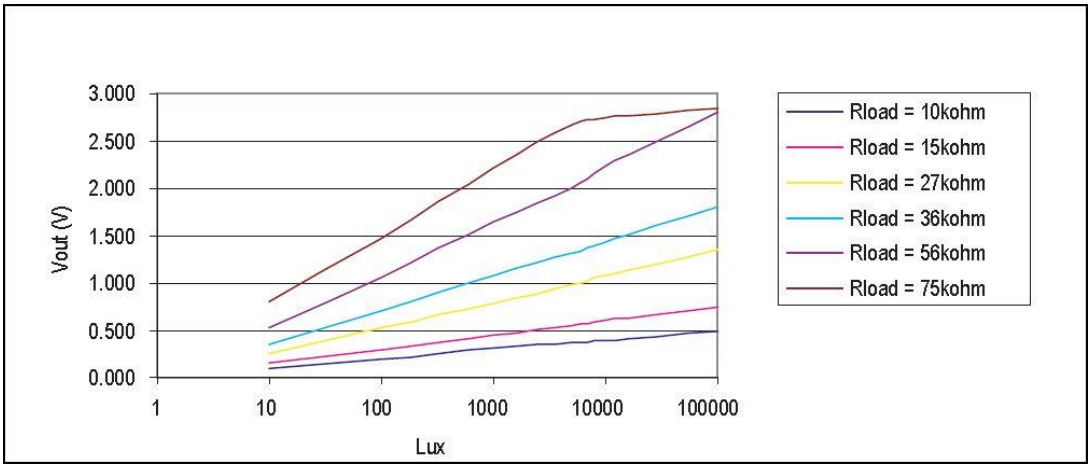


Figure 24 Caractéristique de sortie en fonction de la charge

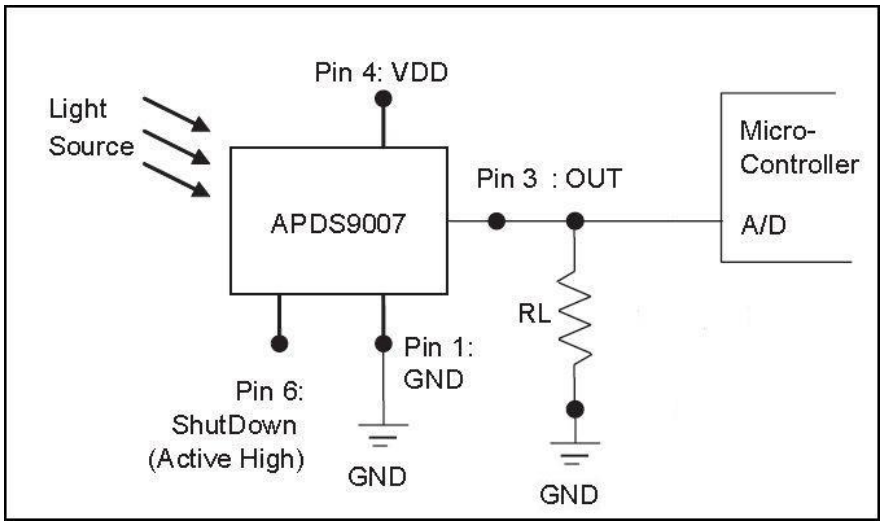
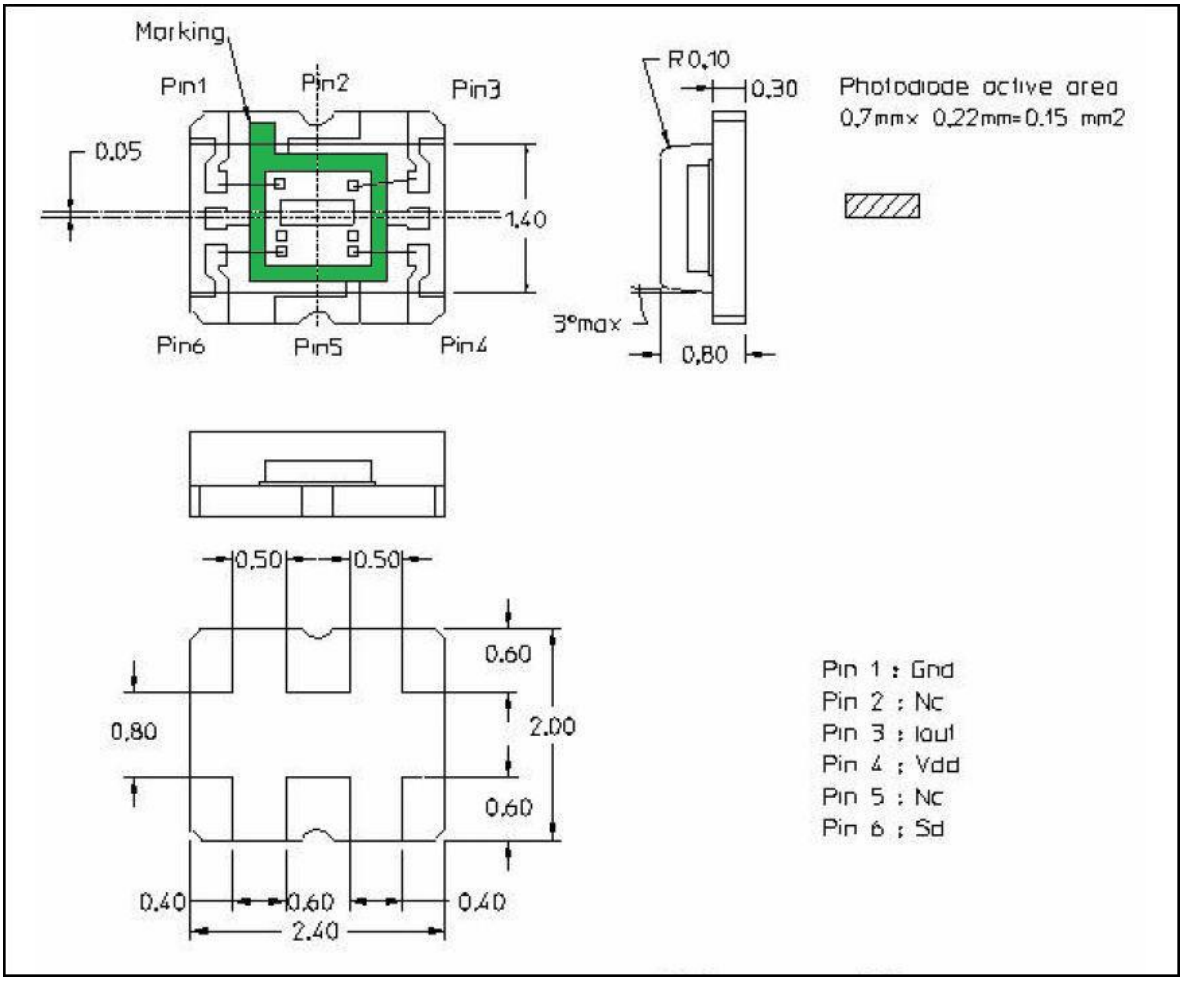


Figure 25Raccordement du capteur

3.6.3.2 Informations sur le capteur.

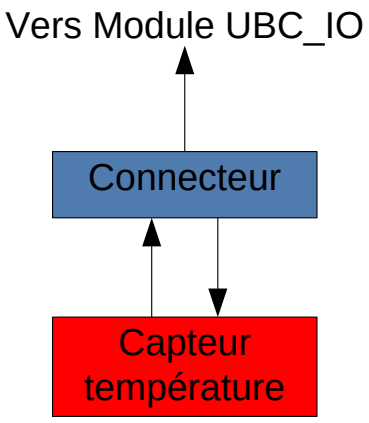
Le capteur APDS-9007 est un capteur de lumière ambiante à sortie analogique en courant d'où la nécessité de le charger par une résistance pour obtenir une tension qui est proportionnelle à la lumière. Il est d'une taille très petite et est utilisé dans des applications de contrôle de backlight (allumage d'un écran de GSM, réglage de l'intensité lumineuse d'une télévision en fonction du soleil,…)

3.6.3.3 Dimensions du composant.



3.7 Le module ION_CAPT

3.7.1 Schéma bloc de la carte.



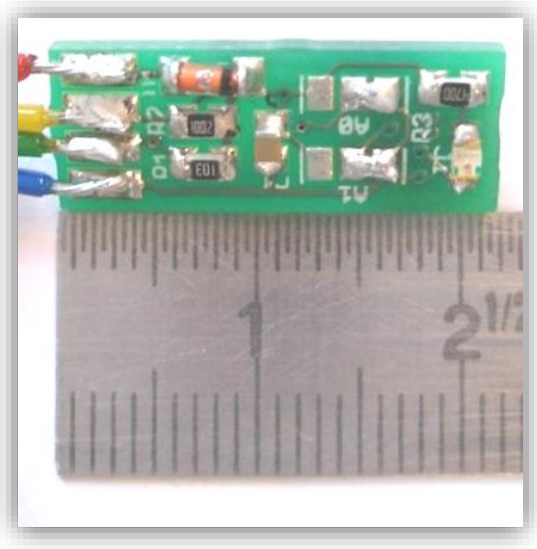


Figure 26 Module ION_CAPT coté alimentation et "jumper"

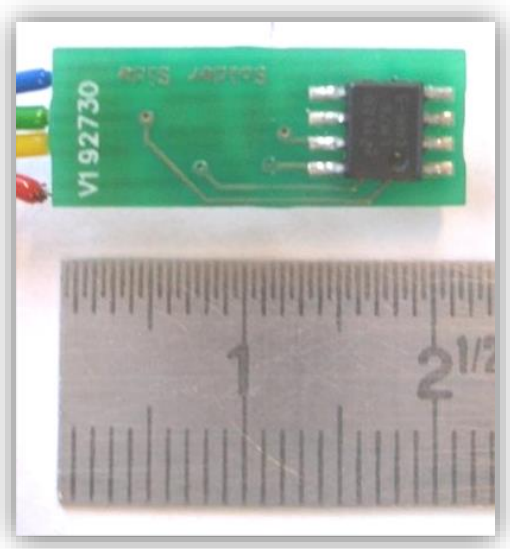


Figure 27 Module ION_CAPT coté LM76

3.7.2 Rôle de la carte.

La carte ION_CAPT permet d'effectuer une mesure de la température. Ici, on mesure la température du sol. C'est un module déporté du module MOD_CAPT, il vient s'y connecter via un des connecteurs Molex. Cette carte a été conçue pour accueillir le capteur LM76 de chez National Semiconductor. Ce capteur étant de petite taille, il a été placé sur un PCB pour améliorer sa solidité et pour faciliter la connexion avec le module MOD_CAPT.

C'est la carte MOD_CAPT qui émet les demandes de mesures à la carte ION_CAPT afin de connaître la température et le taux d'humidité. La carte ION_CAPT répond à ces requêtes en envoyant les mesures instantanées !

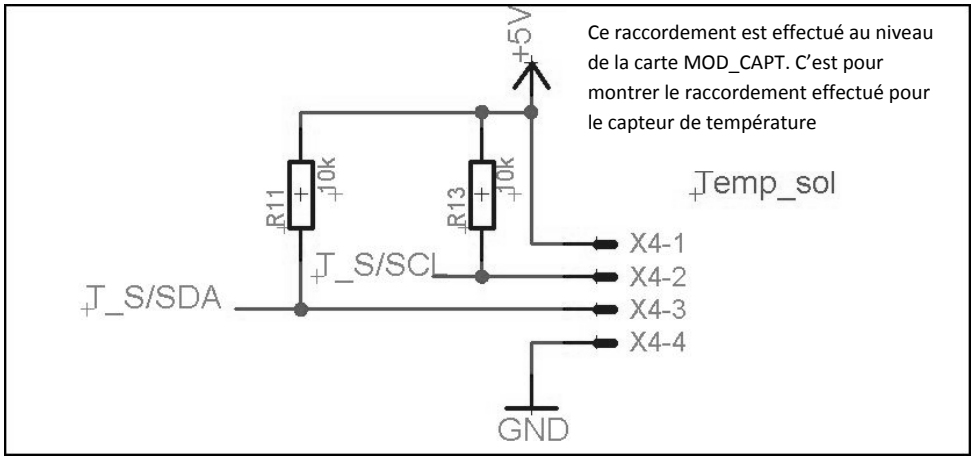
Quatre fils partent du module MOD_CAPT vers le module ION_CAPT :

- l'alimentation pour le capteur (+5V) ;
- la masse du circuit ;
- le signal d'horloge pour la communication ;
- le signal pour le transfert des données.

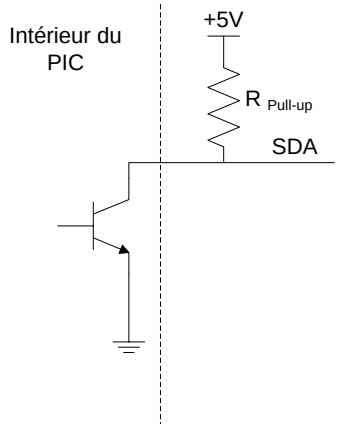
Le schéma électrique général de la carte et le dessin de celle-ci se trouvent dans les annexes aux pages XIV et XV.

3.7.3 Explication des différents composants.

3.7.3.1 Schéma de raccordement.



Deux résistances de « pull-up » sont placées sur le bus « I²C » pour éviter d'avoir des potentiels flottants mais surtout parce que SDA et SCL sont des commandes « pull-down » donc il est indispensable de placer ces résistances.



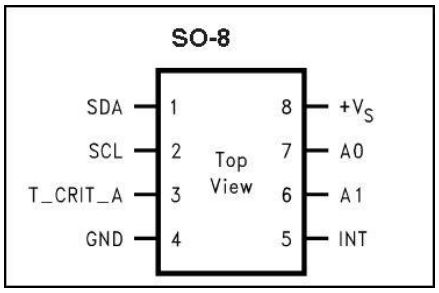
Lorsque le transistor ne conduit pas, la résistance de « pull-up » amène la ligne SDA au potentiel de +5V. Lorsque celui-ci conduit, la ligne est amenée au potentiel de la masse → on commande le transistor pour créer le signal de données !

Sur le PCB, deux jumpers sont disponibles. Ceux-ci servent à programmer une adresse. On peut donc connecter 4 capteurs de température sur le même bus I²C.

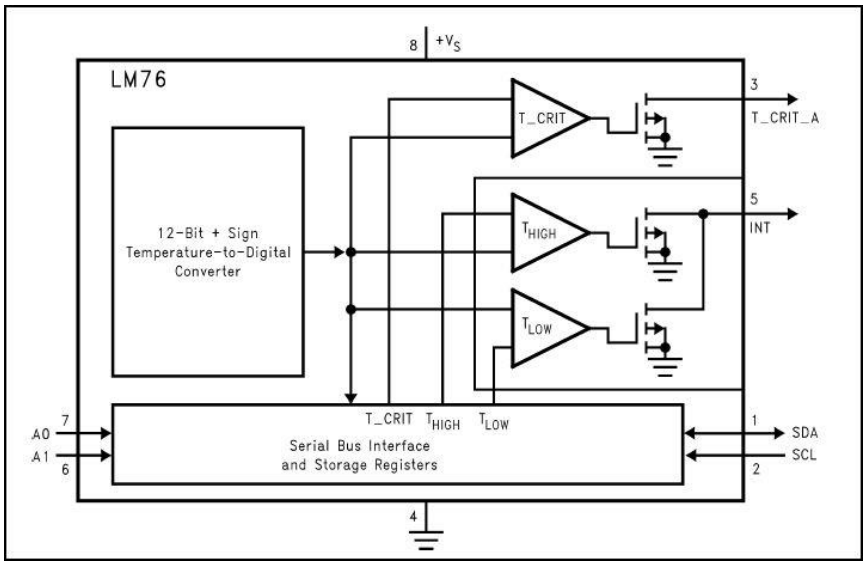
3.7.3.2 Description du capteur.

Le circuit LM76 est un capteur digital de température. Il contient un convertisseur température/valeur digital, 3 comparateurs, une interface série pour le bus I²C et différents registres.

3.7.3.3 Brochage du composant et diagramme simplifié.



- Pin 1 : SDA : Signal de données bidirectionnel pour la communication entre le capteur et le PIC.
- Pin 2 : SCL : Signal d'horloge pour permettre la synchronisation lors des communications.
- Pin 3 : T_CRIT_A : Lorsque la température atteint un seuil préprogrammé, la sortie se met à l'état haut → on peut y placer une alarme.
- Pin 4 : GND : Masse du circuit.
- Pin 5 : INT : Signal d'interruption.
- Pin 6 : A1 : Permet de coder une adresse pour la communication I²C.
- Pin 7 : A0 : Permet de coder une adresse pour la communication I²C.
- Pin 8 : $+V_S$: Alimentation du circuit (+5V).



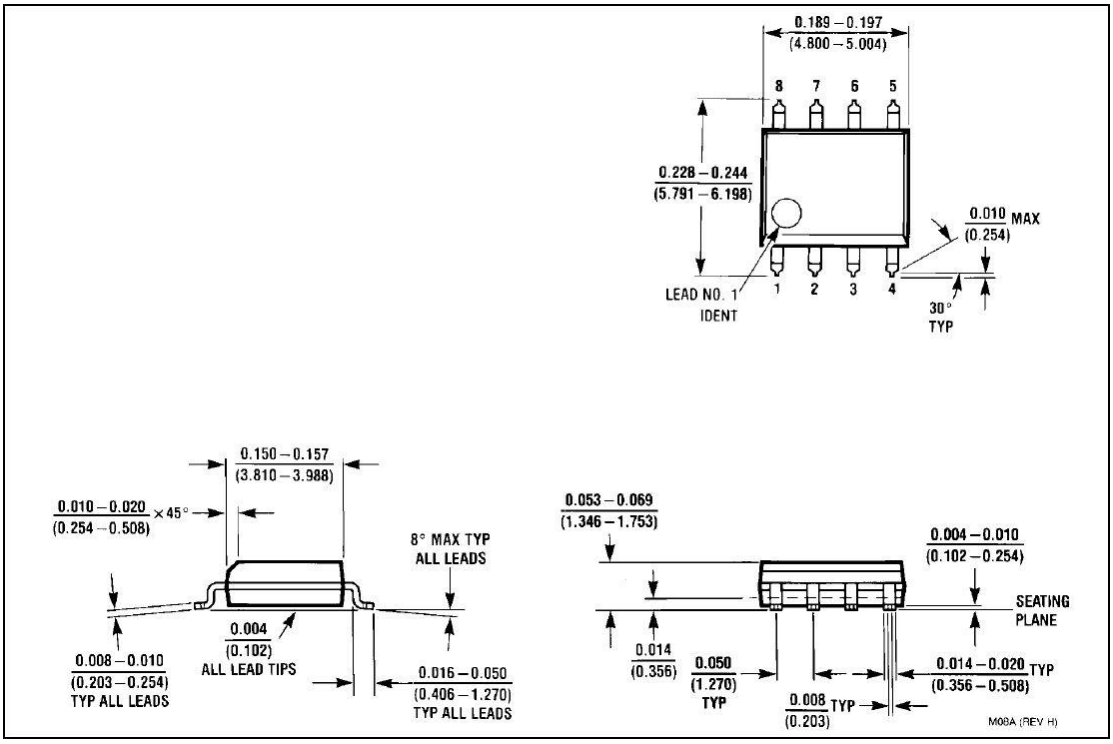
On voit que la sortie du convertisseur rentre dans trois comparateurs ainsi que dans le module gérant l'interface série et contenant les différents registres.

Comme registre, il y a un seuil bas, un seuil haut, un seuil critique de température et d'autres registres de paramétrage du capteur non utilisés.

Si le seuil critique est atteint, la Pin 3 se met à l'état haut. Si la température atteint le seuil bas ou le seuil haut, une interruption sera générée sur la Pin 5.

Les entrées A_0 et A_1 permettent de programmer une adresse du composant sur le bus I²C.

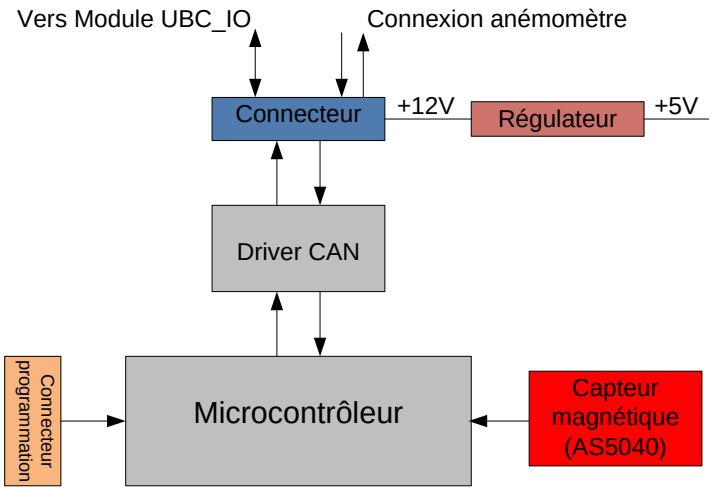
3.7.3.4 Dimensions du composant.



Les valeurs entre parenthèses sont les dimensions du composant en millimètres.

3.8 Le module MOD_VENT

3.8.1 Schéma bloc de la carte.



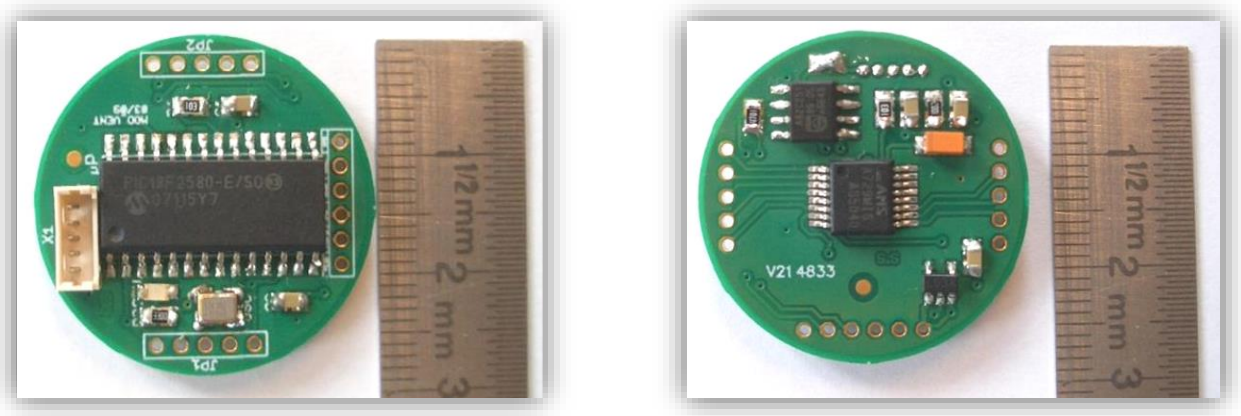


Figure 28 Module MOD_VENT coté PIC à gauche et coté circuit AS5040 à droite

3.8.2 Rôle de la carte.

Cette carte est le deuxième module effectuant les mesures. Différents capteurs y sont connectés :

- La girouette et l'anémomètre.

En effet, le module UBC envoie des requêtes de mesures toutes les secondes via le bus CAN au module MOD_VENT. Le module reçoit donc ces requêtes, lit les valeurs des mesures dans ses registres et envoie à son tour par le bus CAN les informations demandées.

La carte possède un microcontrôleur PIC 18F2580-E/SO de chez Microchip, un driver CAN (A82C251) pour la communication avec le module UBC, un encodeur magnétique (AS5040).

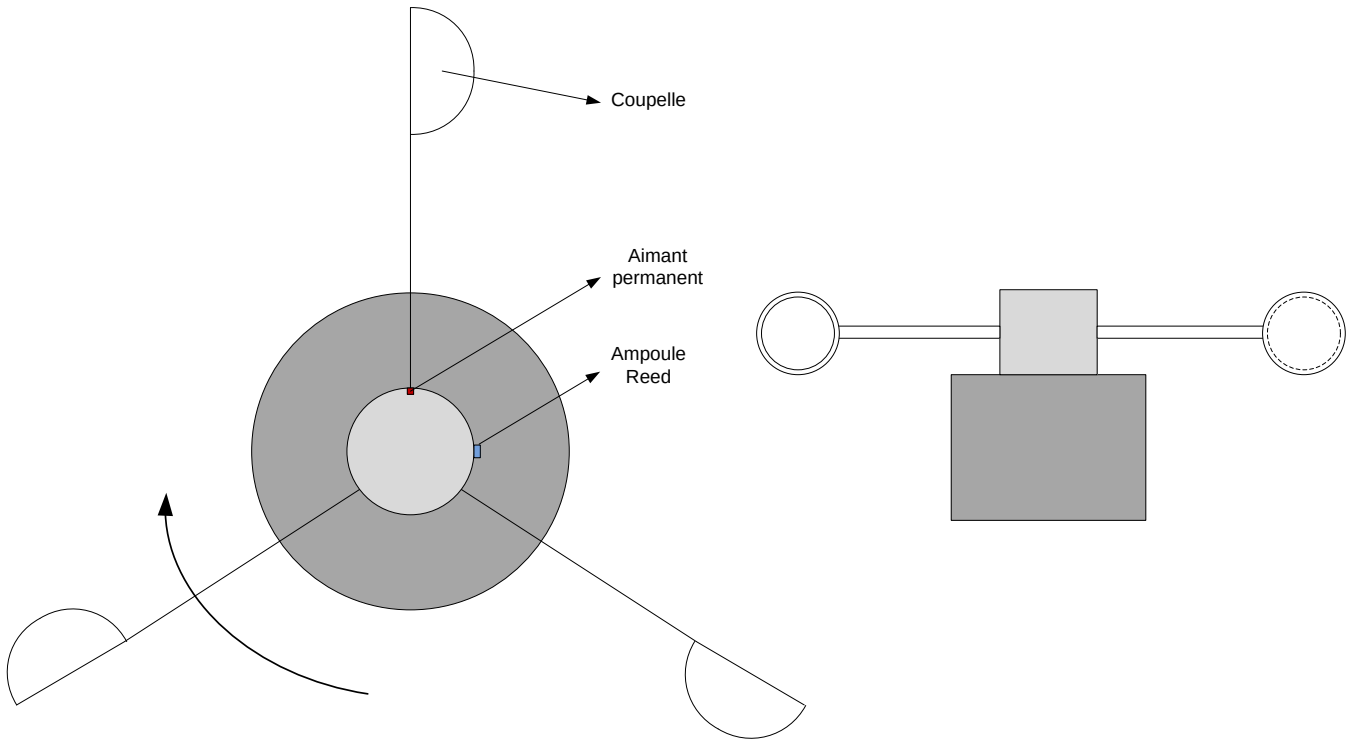
La carte est alimentée en +12V par le même câble que le bus CAN. Ce câble est constitué de 4 fils : +12V, CANH, CANL, GND. Cette tension est ensuite transformée en +5V pour l'alimentation des différents composants.

C'est la deuxième carte que j'ai dû dessiner avec le logiciel Eagle.

Le schéma électrique général de la carte et le dessin de celle-ci se trouvent dans les annexes aux pages XVI et XVII.

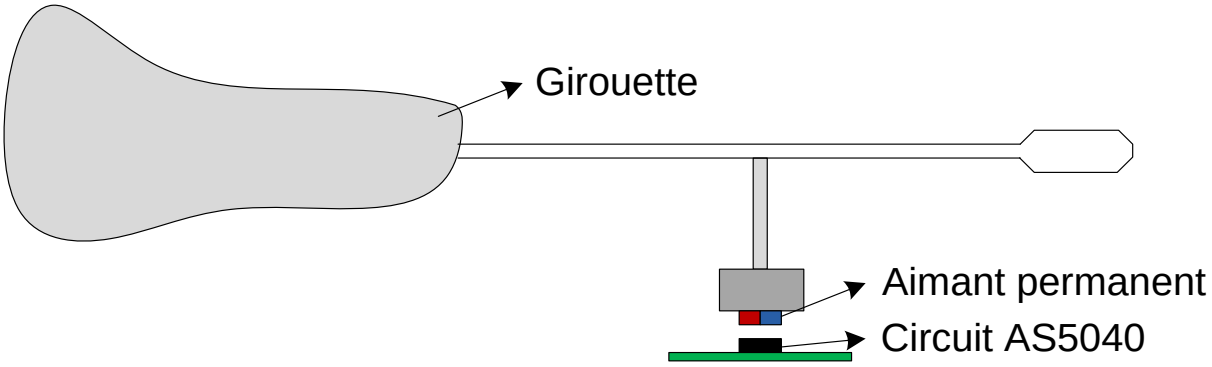
3.8.3 Explication des différents composants.

3.8.3.1 L'anémomètre à coupelles.



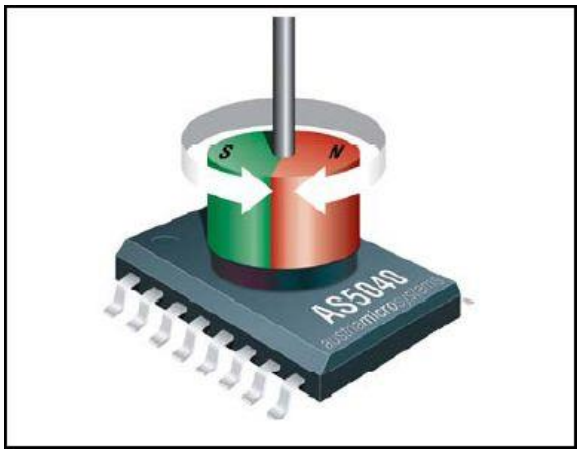
L'anémomètre est basé sur le même principe que le pluviomètre. En effet, à chaque tour de celui-ci, une impulsion est générée par l'ampoule Reed. On compte le nombre de tours effectués (le nombre d'impulsions) par seconde ; connaissant le rayon du cercle formé par l'anémomètre, on en déduit la vitesse de rotation de l'anémomètre et donc du vent en m/s.

3.8.3.2 La girouette.



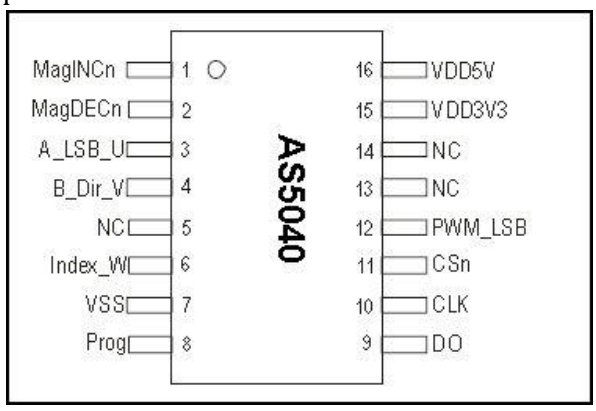
La girouette est placée sur un socle en PVC. Le socle est en forme de cylindre contenant le roulement à billes (pour une rotation sans frottement) et sur lequel vient se poser tout le bras. Le socle contient également l'aimant permanent et la carte électronique.
Le principe de la girouette est simple. Le circuit AS5040 est soumis à un champ magnétique venant de l'aimant permanent. L'aimant est solidaire de la girouette → il tourne donc en fonction de la direction du vent.
Selon l'orientation de l'aimant, le circuit AS5040 (encodeur magnétique) va sortir un angle compris entre 0 et 360°. Cette donnée est convertie par le convertisseur 10 bits intégré dans le microcontrôleur PIC →on obtient une valeur comprise entre 0 et 1024 (2^{10}).

3.8.3.2.1 Explication.



Le circuit AS5040 est un codeur magnétique rotatif sans contact (donc aucune usure !). Il permet d'effectuer des mesures précises sur tout un tour (360°). Le circuit contient différents éléments à effet Hall. Pour mesurer l'angle de rotation, un simple aimant à deux pôles est placé au dessus du circuit (l'aimant tourne avec la girouette). La mesure absolue de l'angle est une indication instantanée de la position angulaire de l'aimant et donc de la direction du vent.
Résolution : 1 point = 0.35° (360°/1024) → grande précision !

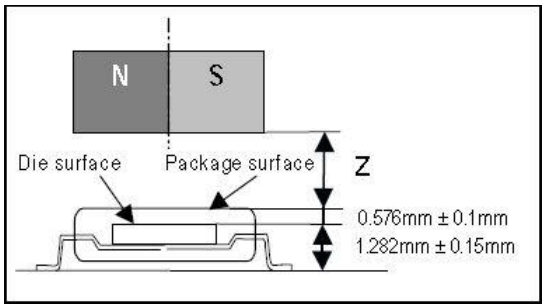
3.8.3.2.2 Brochage du composant.



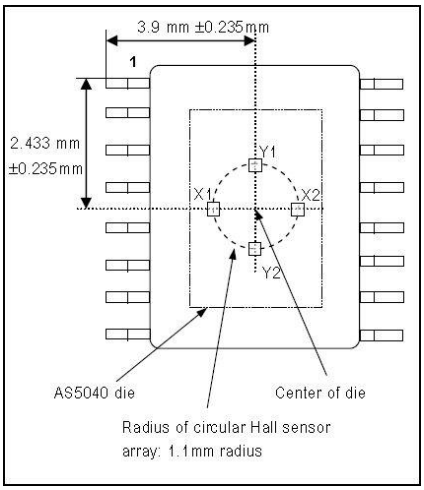
- Pin 1: MagNCn: Un état bas sur cette Pin indique une diminution de la distance entre le circuit et l'aimant.
- Pin 2: MagDECn: Un état bas sur cette Pin indique une augmentation de la distance entre le circuit et l'aimant.
- Pin 3: A_LSB_U: Pin par laquelle sort différentes informations de fonctionnement du circuit.
- Pin4: B_Dir_V: Pin par laquelle sort différentes informations de fonctionnement du circuit.
- Pin5: NC: Non connectée.
- Pin 6: Index_W:
- Pin 7: VSS: Masse du circuit (GND).
- Pin 8: Prog: Entrée de programmation ou entrée de Datas IN. Si on n'utilise pas de programmation, on connecte cette Pin à la masse.
- Pin 9: DO: Sortie de la donnée sur l'interface série synchrone.
- Pin10: CLK: Entrée de la Clock de l'interface série.
- Pin 11: CSn: Activation du circuit. Si on a un état bas sur cette Pin, on active le circuit.
- Pin 12: PWM_LSB: Sortie PWM. La valeur de l'angle est codée sur un signal PWM.
- Pin13: NC: Non connectée.
- Pin 14: NC: Non connectée.
- Pin 15: VDD3V3: Pin pour alimenter le circuit en 3V3 (ne pas connecter la Pin 5V !).
- Pin 16: VDD5V: Pin pour alimenter le circuit en 5V (ne pas connecter la Pin 3V3 !).

3.8.3.2.3 Précautions d'utilisation.

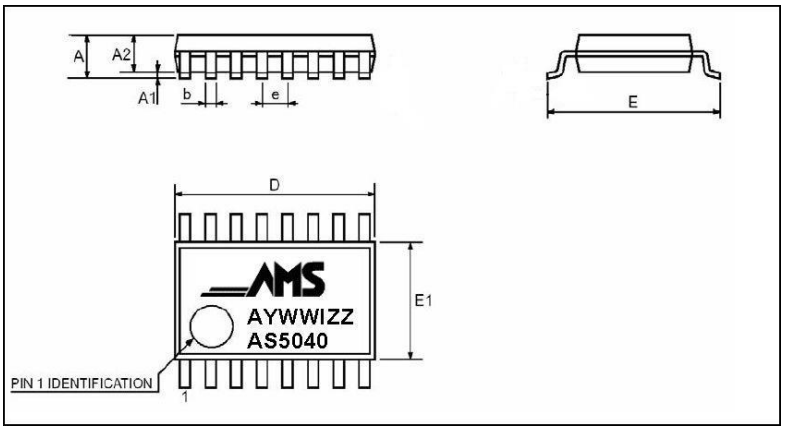
L'aimant permanent doit être placé à une distance très précise du circuit. Il ne peut pas bouger verticalement, mais uniquement tourner. Si l'aimant est placé trop loin du circuit, les mesures seront fausses et le circuit ne fonctionnera pas correctement. Étant donné que le circuit AS5040 est constitué de 4 cellules à effet Hall placées à 90° les unes des autres, l'aimant doit être parfaitement centré !



Avec Z compris entre 0.5mm et 1.8mm. Cette distance doit absolument être respectée pour obtenir un bon fonctionnement du circuit et donc une mesure correcte !



3.8.3.2.4 Dimensions du circuit.



Ce circuit est un composant de type « SMD », voici ses dimensions :

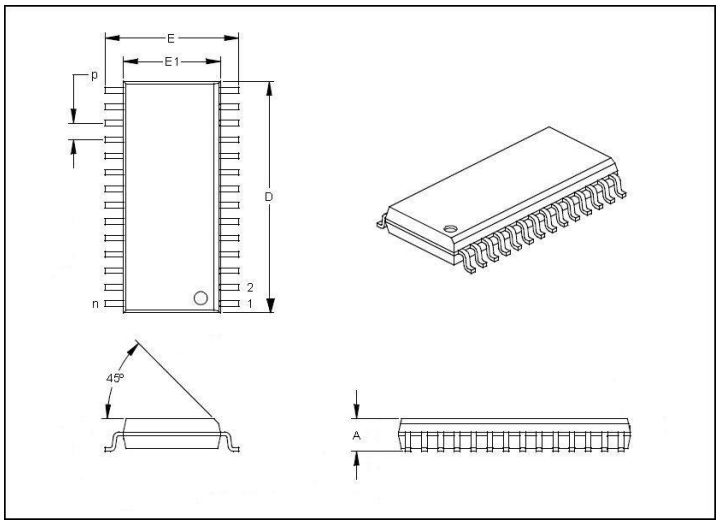
- A: Epaisseur du circuit avec les Pins: 1.86mm
- A1: Epaisseur d'une Pin: 0.13mm
- A2: Epaisseur du circuit: 1.73mm
- E: Largeur du circuit avec Pins : 7.8mm
- E1: Largeur du circuit : 5.3mm
- D: Longueur du circuit : 6.2mm
- b: Largeur d'une Pin : 0.315mm
- e: Distance entre Pins: 0.65mm

3.8.3.3 Le PIC 18F2580-E/SO.

Il a fallu placer un microcontrôleur sur cette carte pour gérer toutes les mesures, les envois CAN et les communications avec les capteurs. Il a pour but de « désengorger » le travail effectué par le PIC placé sur la carte UBC. Le PIC placé sur la carte MOD_VENT s'occupera du traitement de toutes les mesures, effectuera les calculs nécessaires à la mise en forme de celles-ci et enverra les données traitées à la carte UBC. On obtient ainsi des modules totalement autonomes et ne dépendant pas d'autres modules (si ce n'est que pour communiquer entre eux).

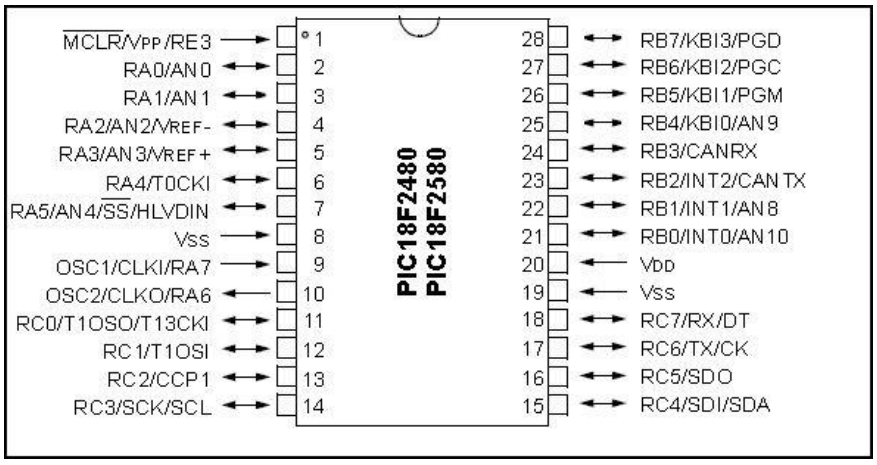
Mon choix s'est tourné pour un PIC 18F2580-E/SO de 28 Pins. Il permet de gérer des communications CAN, SPI et I2C et possède les mémoires suffisantes pour obtenir un fonctionnement satisfaisant.

3.8.3.3.1 Dimensions du composant :



D = Longueur = 17.87 mm
E = Largeur = 10.34 mm
A = Epaisseur du circuit = 2.50 mm
p = pitch = 1.27 mm
E_1 = Largeur sans les pins : 7.49 mm

3.8.3.3.2 Brochage du PIC 18F2580 I/SP :



4. Explication du fonctionnement interne d'un PIC.

Les PIC sont des microcontrôleurs CMOS mis au point par la société Microchip Technologies. Ils constituent un compromis entre simplicité d'emploi et prix de revient.

La structure interne d'un microcontrôleur, le plus simple qui soit, possède les éléments suivants :

- une unité centrale, cœur du système, également appelée CPU (Central Processing Unit). Elle exécute les instructions du programme. Dans cette unité centrale, nous retrouverons plusieurs éléments comme l'unité arithmétique et logique (UAL) ;

- une mémoire contenant le programme à exécuter par le microcontrôleur, généralement appelée mémoire morte ou ROM (Read Only Memory). Cette mémoire à la particularité de sauvegarder en permanence les informations qu'elle contient, même en absence de tension. Ce qui est primordial sinon il faudrait reprogrammer le microcontrôleur à chaque mise sous tension.

- une mémoire vive également appelée RAM (Random Access Memory). Cette mémoire permet de sauvegarder temporairement des informations. En effet, ces dernières sont effacées lorsqu'il n'est plus alimenté. Le microcontrôleur pourra utiliser cette mémoire pour stocker les variables temporaires ou les calculs intermédiaires ;

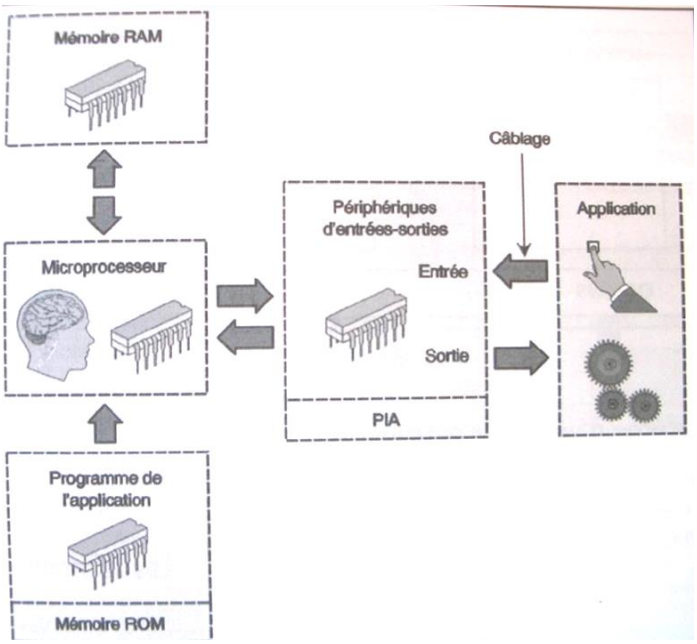
- une mémoire EEPROM (mémoire morte effaçable électriquement et programmable). Cette mémoire permet de sauvegarder des informations qui ne doivent pas être perdues lorsque le composant n'est plus alimenté électriquement. Le microcontrôleur va utiliser cette mémoire afin de stocker les variables fixes. Comme par exemple, les offsets de chaque capteur, les paramètres initiaux,…
Le contenu de la mémoire peut être effacé par un passage d'un courant électrique et elle peut être lue une infinité de fois.

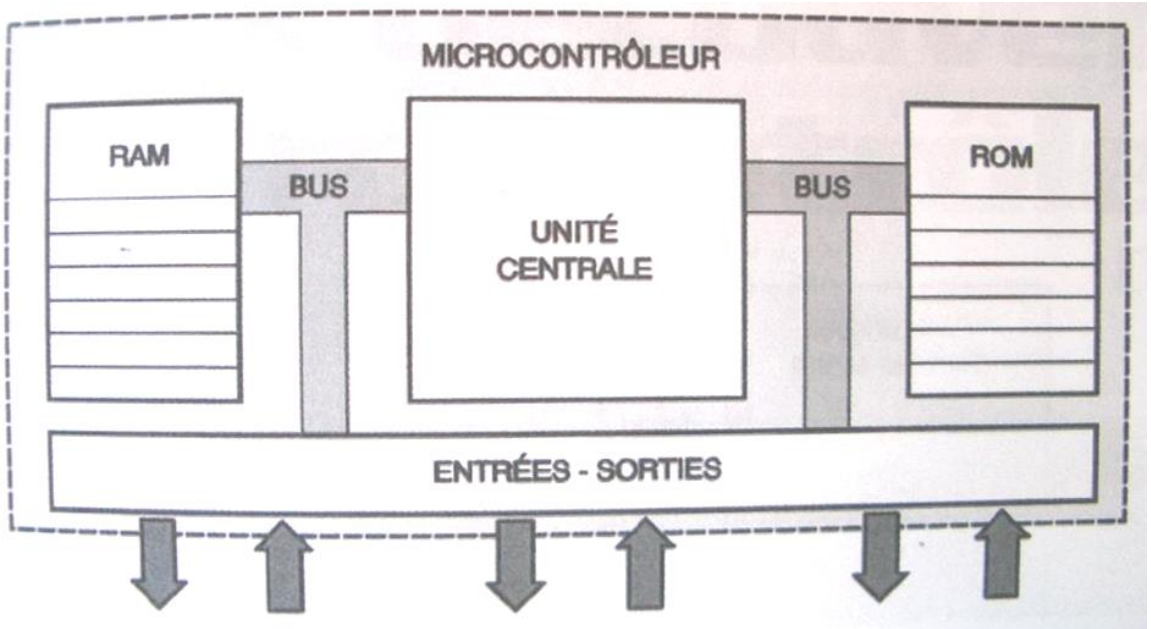
- un port d'entrée-sortie permet au microcontrôleur de dialoguer avec l'extérieur pour prendre par exemple l'état d'un interrupteur, allumer une LED, lire une valeur analogique,…

Les bus internes permettent la communication entre les différents éléments intégrés au microcontrôleur. Il y a trois types de bus :

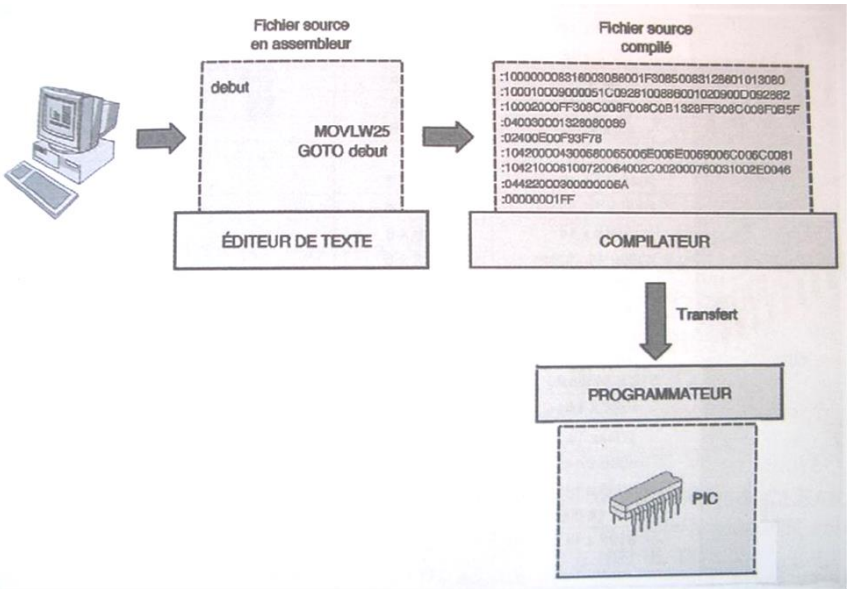
- Le bus Data ;
- le bus d'adresses ;
- le bus de contrôle.

Schéma simple des différents éléments :





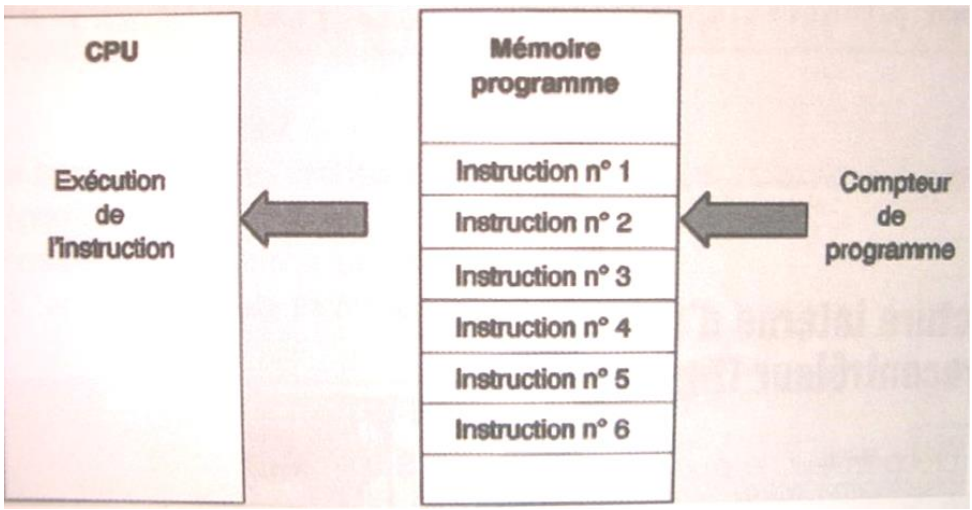
4.1 Programmation d'un microcontrôleur.



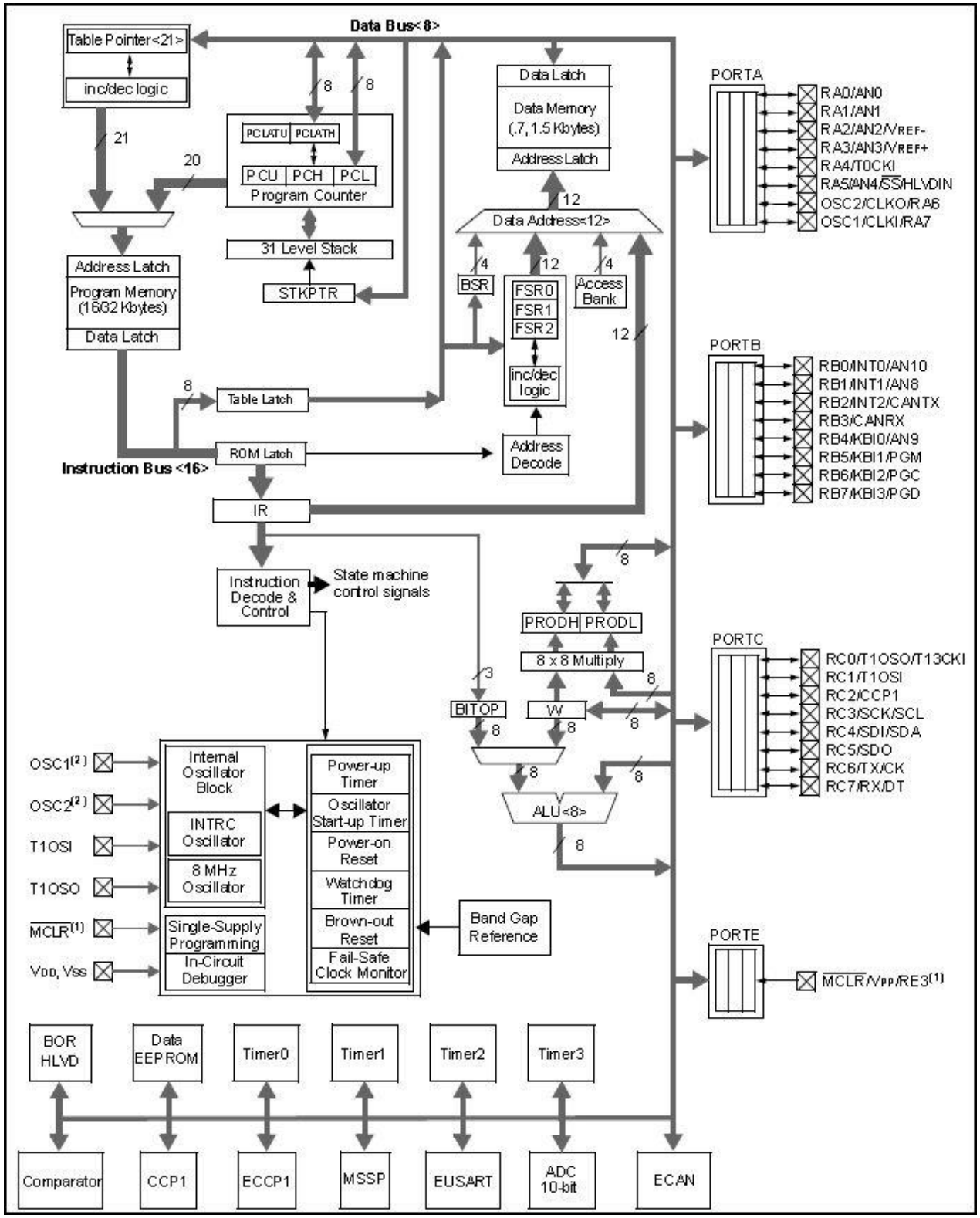
1. Il faut écrire un programme en assembleur avec un éditeur de texte.
2. Il faut compiler ce code source. La compilation consiste à remplacer les ordres mnémoniques du fichier assembleur par des codes binaires compréhensibles par le microcontrôleur. Lorsque la compilation est effectuée, un fichier binaire comportant l'extension .hex est créé par le compilateur.
3. Pour transférer ce fichier binaire du disque dur vers la mémoire programme (flash) du microcontrôleur, il faut utiliser un programmeur qui se connectera sur le socle de programmation.

4.2 Exécution du programme.

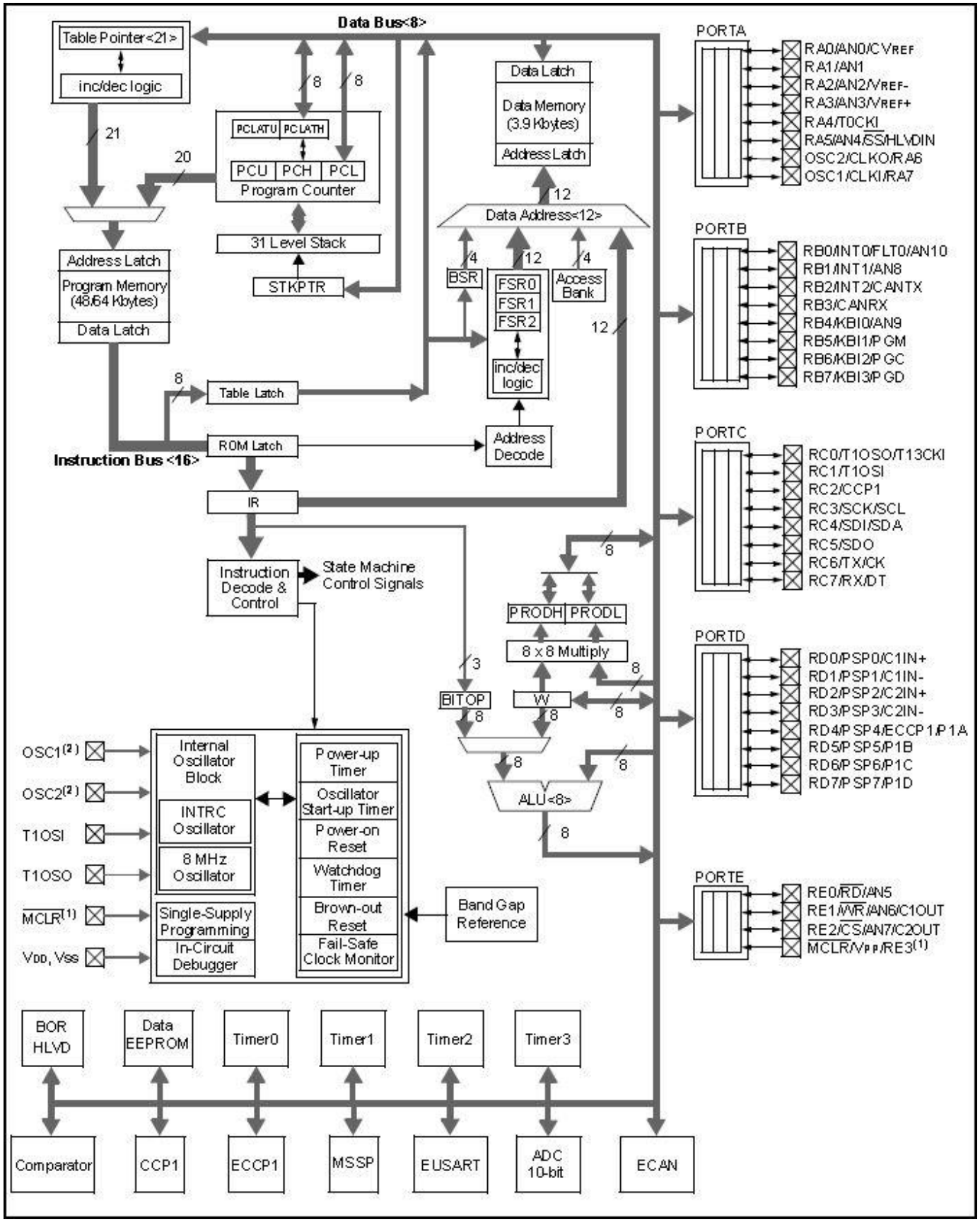
Le microcontrôleur exécute une à une les instructions codées sous forme binaire qu'on a transférées dans la mémoire flash (ROM). Un registre spécifique du microcontrôleur appelé CP (Compteur de programme) est chargé de pointer l'instruction stockée en ROM qui devra être exécutée par la CPU. Il sera incrémenté afin d'exécuter toutes les instructions.



4.3 Structure interne du PIC18F2580.



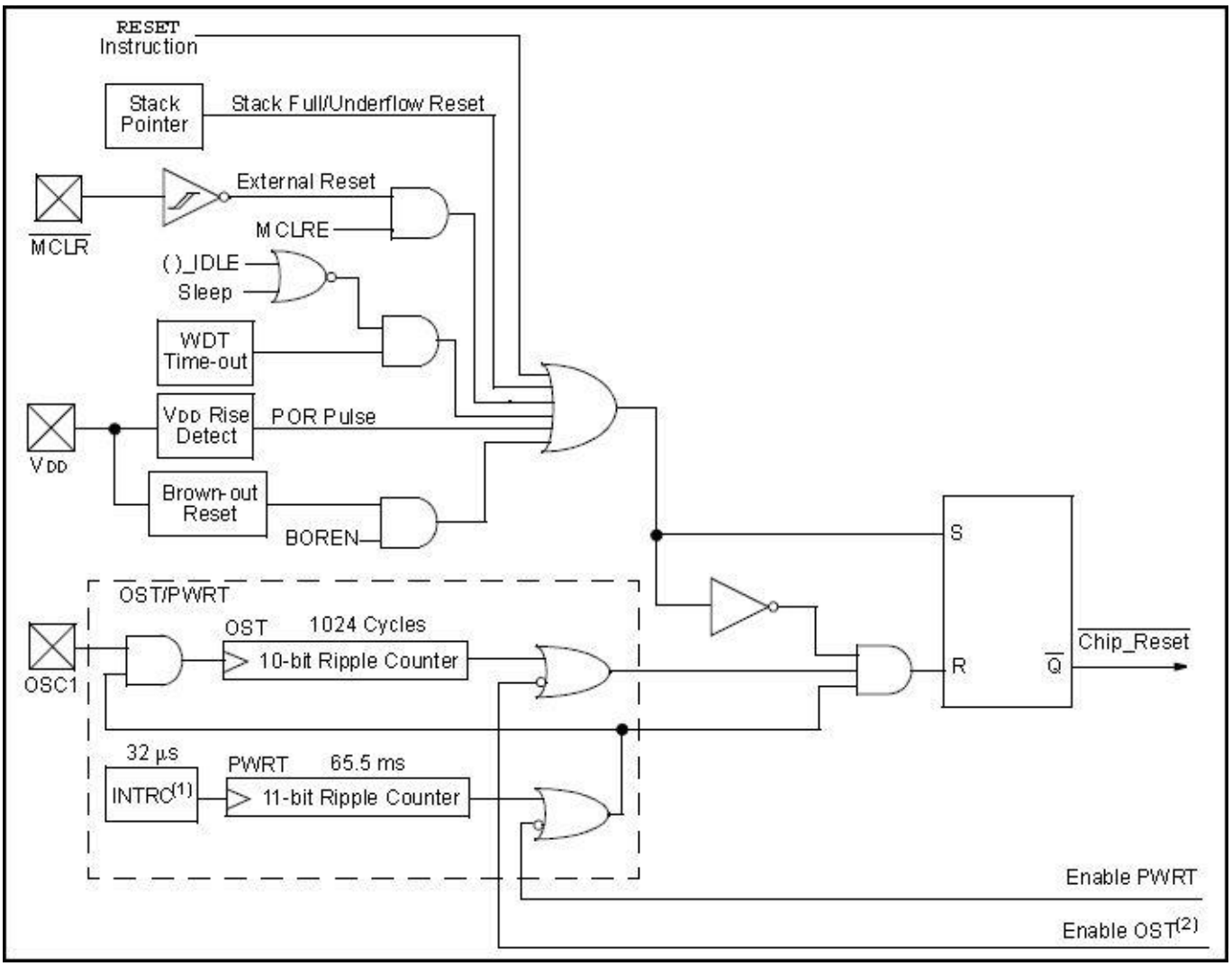
4.4 Structure interne du PIC18F4680.



4.5 Reset du PIC.

Le RESET du composant est possible de plusieurs manières :

- POR (Power-on RESET) : c'est lorsque la tension d'alimentation n'est pas stable ou trop faible ;
- MCLR : c'est lorsque l'on met la masse sur la Pin correspondante ;
- WDT (Watch Dog Timer) : c'est un compteur qui est incrémenté par un oscillateur RC interne au PIC. Si ce compteur passe en OVERFLOW, le PIC est reseté. Dans les programmes, le Watch Dog est remis à zéro avant de passer en OVERFLOW ; si le programme se plante, le Watch Dog n'est plus remis à zéro ; il passe en OVERFLOW et reset automatiquement le PIC.



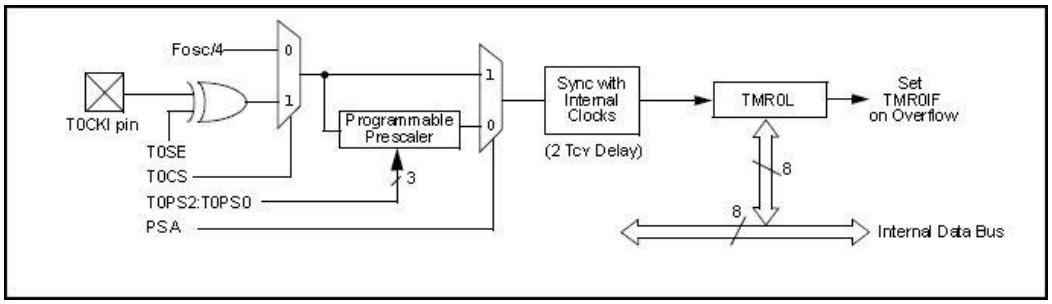
4.6 Les Timer.

Le PIC contient 4 Timer :

- Timer 0 (Timer « principal »).
- Timer 1.
- Timer 2 (non-utilisé dans les projets).
- Timer 3 (non-utilisé dans les projets).

4.6.1 Le Timer 0.

On peut considérer le Timer 0 comme un « générateur d'interruption ». C'est-à-dire, toutes les 10 ms, une interruption est générée. Voici le schéma interne du Timer 0 :



Le Timer 0 est un registre (de 8 bits → 255 max) qui s'incrémente à chaque front montant. Lorsqu'on arrive à l'OVERFLOW (passage de 255 à 0 → bouclage), une interruption est créée.

<u>Voici le réglage du Timer 0 (pour obtenir 10ms) :</u>

Nous voulons donc obtenir une interruption toutes les 10ms tout en utilisant la fréquence de l'oscillateur externe qui est de 20MHz.

En fonction du diagramme, voici les différents réglages à effectuer :

- Oscillateur externe → on règle le premier multiplexeur en mettant TOCS à 0. La fréquence est directement divisée par 4.

$$F_{osc} = \frac{20000000Hz}{4} = 5000000Hz$$

- Le signal doit ensuite passer par le Prescaler pour diminuer la fréquence au maximum. Il faut donc mettre PSA à 0 au niveau du deuxième multiplexeur.
  Le Prescaler est programmable selon l'état de T0PS2, T0PS1, T0PS0 :

T0PS2	T0PS1	T0PS0	Prescaler
0	0	0	2
0	0	1	4
0	1	0	8
0	1	1	16
1	0	0	32
1	0	1	64
1	1	0	128
1	1	1	256

  Pour réduire la fréquence au maximum, j'ai choisi un Prescaler de 256 → T0PS2 = 1 ; T0PS1 = 1 ; T0PS0 = 1.

$$Fréquence = \frac{5000000Hz}{256} = 19531\ Hz$$

  On obtient donc un front montant toutes les **0.051ms**.

- Pour obtenir une interruption toutes les 10 ms, il faut placer une valeur initiale dans le Timer 0 après chaque OVERFLOW. Par exemple, on peut placer la valeur 100, le « compteur » de fronts montant comptera seulement 156 fronts (256 -100).

$$\frac{10ms}{0.051} = 196.07\ soit\ \mathbf{196}$$

→ Valeur initiale du Timer 0 = 256 – 196 = **60**

➔On obtient bien une interruption (TMR0IF) toutes les 10 ms avec ce réglage !

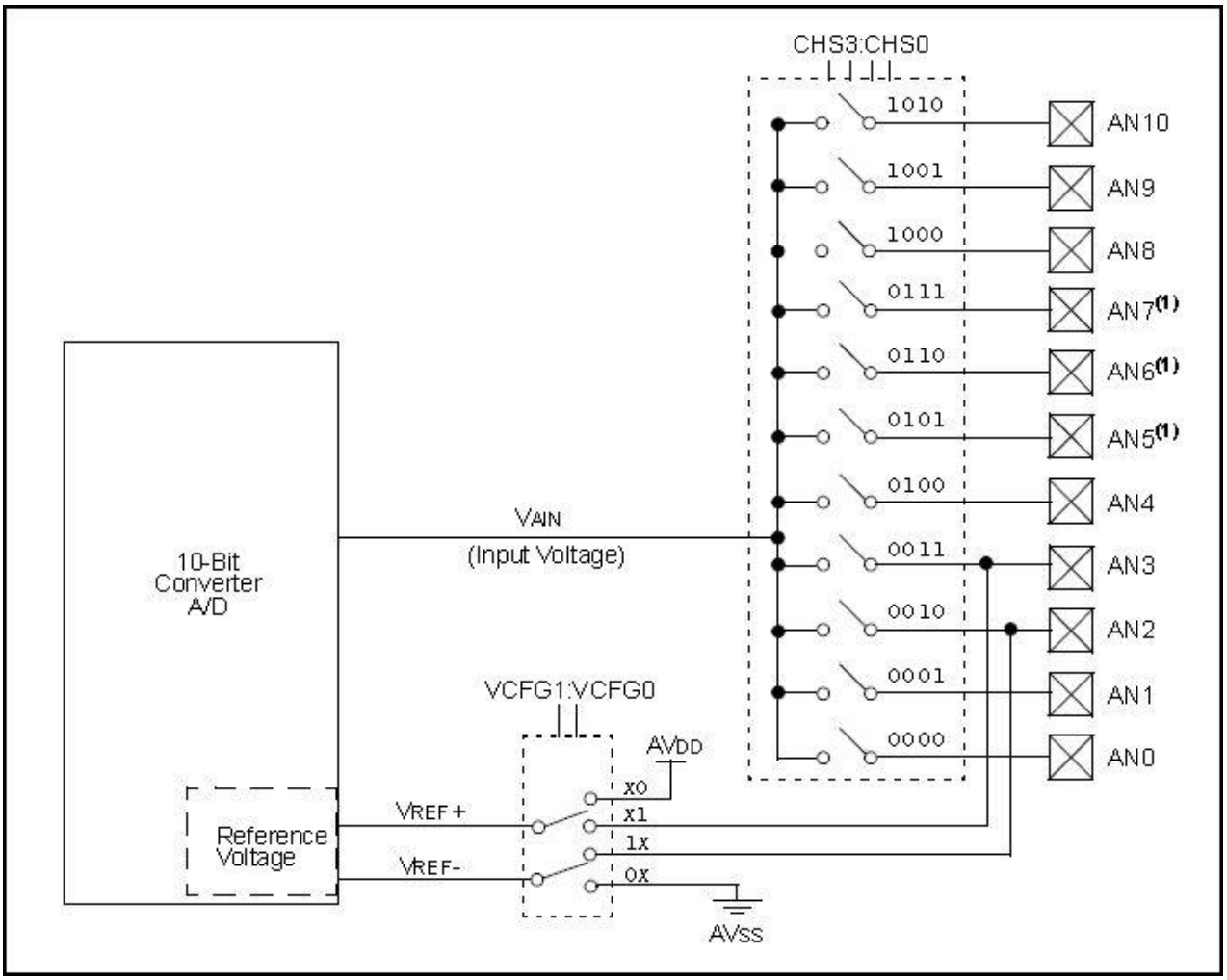
Le principe de « temporisation » est alors facile à mettre en place. Etant donné que l'on dispose d'une interruption toutes les 10ms, il suffit de compter le nombre d'interruptions pour définir un temps → compteur d'interruptions. Ces compteurs sont placés dans l'interruption même du programme. Voici un exemple de programme :

```
if(TMR0IF)                          // Timer 0 = base de temps IT
{
        TMR0 = CNT0_INTR;     // On recharge le timer pour avoir une INTR toute le x ms
        cnt_int_timer_0_UI++;

        // Action sur le Watch Dog
        if(Tempo2 > 0)  Tempo2--;

        // Temps entre chaque mesure
        if(Tempo_CAPTURE > 0)          Tempo_CAPTURE--;

        TMR0IF = 0;
} //if(TMR0IF)
```

Prenons le cas de « Action sur le Watch Dog ». A chaque interruption, TMR0IF passe à 1→ on rentre dans la procédure d'interruption. Si la valeur de la Tempo 2 est plus grande que 0, on décrémente cette valeur de 1 → toutes les 10ms, Tempo 2 sera décrémenté de 1. Si la valeur de Tempo 2 est égale à 100, 10ms x 100 = 1000 ms → 1s → On aura donc une action sur le Watch Dog toutes les secondes.

Pour toutes les temporisations, j'utilise le principe du « compteur d'interruptions de Timer », et j'obtiens ainsi les temps désirés.

4.7 Le convertisseur Analogique/Digital.



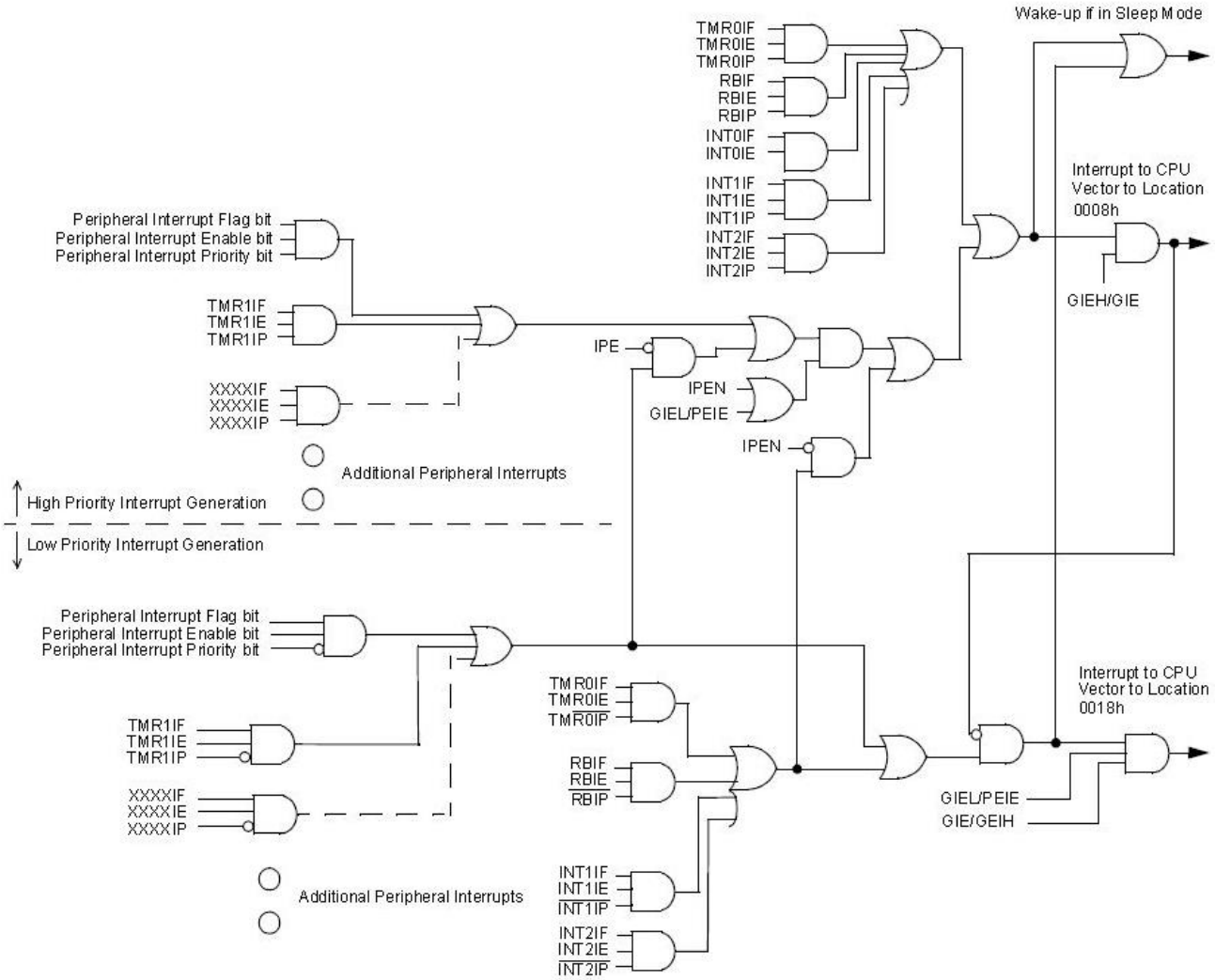
Les Pins du PORT A peuvent être utilisées comme entrées d'un convertisseur analogique/digital sur 10 bits.

Une valeur analogique placée à une des entrées va être convertie en une valeur digitale comprise entre 0 et 1024 (2^{10}).

4.8 Les interruptions (INTR).

Qu'est ce qu'une interruption ?

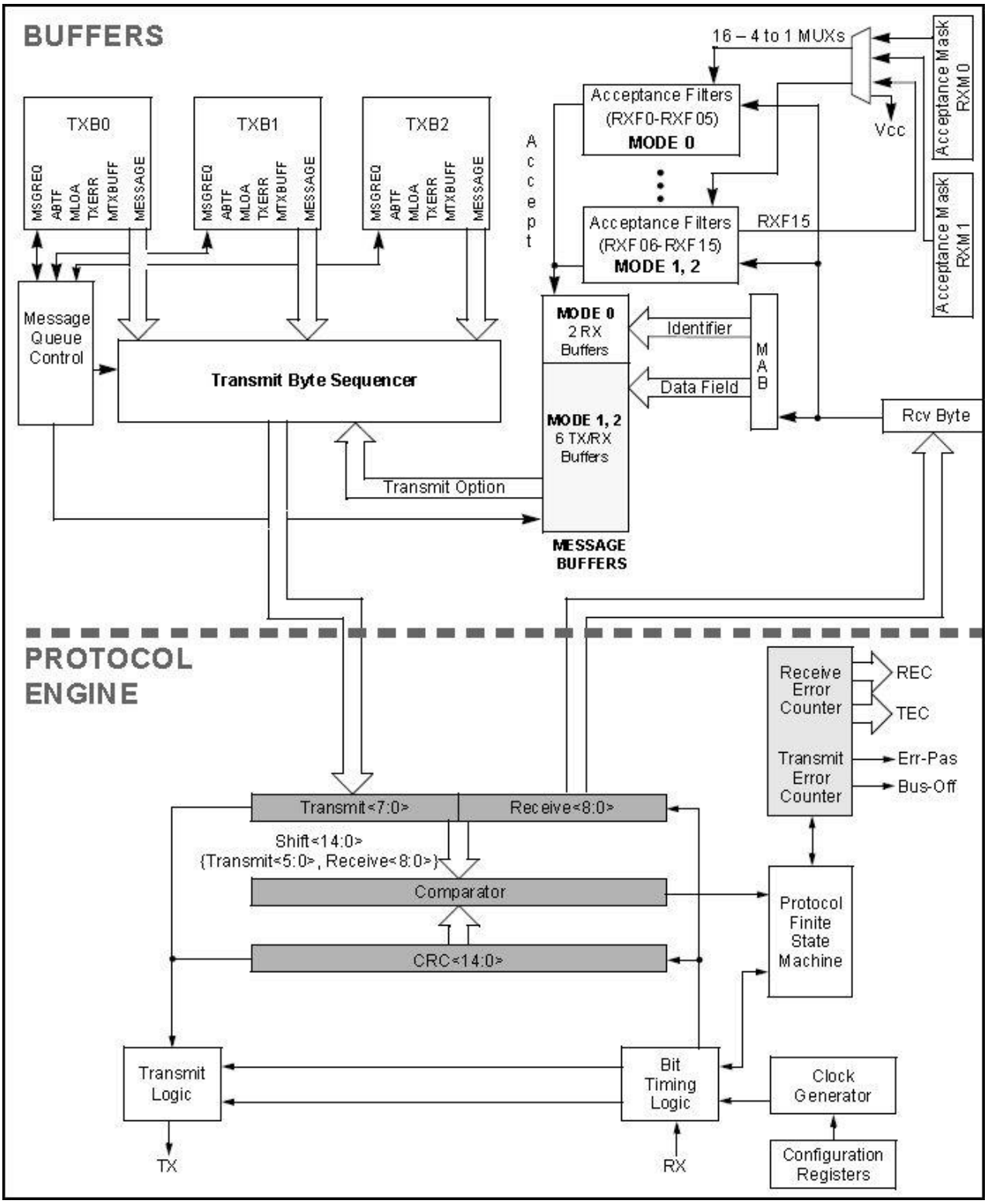
Un programme peut être composé de plusieurs fonctions :
La fonction principale et la fonction d'interruption (plus d'autres fonctions analogues).
La fonction principale est exécutée généralement sans fin, en boucle (boucle « for » infinie). Elle peut être suspendue par l'action d'un périphérique générant une interruption. A ce moment, le CPU termine d'exécuter l'instruction en cours, et JUMP au début de la fonction d'interruption. Une fois le programme de l'interruption terminé, le CPU reprend l'exécution de la fonction principale là où elle avait été laissée.
Une interruption peut être générée par différentes conditions, voici le schéma logique :



Comme sources d'interruptions, nous avons :

- les différents Timer ;
- le convertisseur analogique/digital ;
- les Pins interruptibles du PIC (une interruption est générée sur cette Pin par un périphérique extérieur).
- …

4.9 Diagramme du module CAN contenu dans chaque PIC.



5. Explication des différents bus utilisés.

Différents bus de données sont utilisés dans mon application. Nous disposons d'un bus I²C et d'un bus SPI pour la communication entre les capteurs et le microcontrôleur.
Un Bus CAN est utilisé pour dialoguer entre les différents modules (→ entre les différents microcontrôleurs).
Une liaison RS485 est présente au niveau hardware sur la station mais celle-ci n'est pas utilisée dans cette application (car conflits avec la communication UART pour le GSM→ utilisation des mêmes traces et mêmes Pins du PIC !). Une communication Wireless est également envisageable.
De plus, l'élément principal est qu'il est possible de communiquer avec la station via un GSM (moyen de communication expliqué précédemment).

5.1 I²C (Inter Integrated Circuit Bus).

5.1.1 Explication.

I²C est un bus développé par Philips pour les applications de domotique et d'électronique domestique. Il permet de relier facilement un microprocesseur avec les différents circuits d'une télévision moderne par exemple (ce bus est parfois appelé TWI).

Il existe d'innombrables périphériques exploitant ce bus, il est même implantable par logiciel dans n'importe quel microcontrôleur. Le poids de l'industrie de l'électronique « grand public » a permis des prix très bas aux nombreux composants.

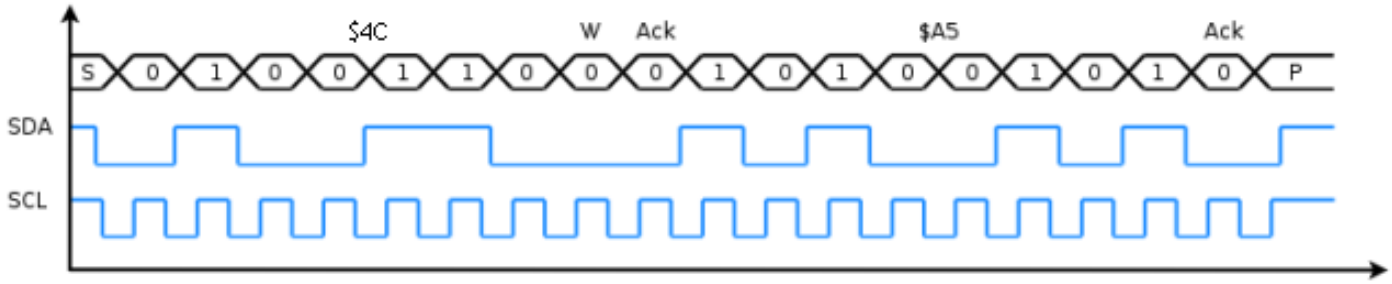
Les données sont transmises en série de manière synchrone ; cela signifie que les informations sont envoyées à la suite sur le même fil (une donnée par coup d'horloge), contrairement à une communication parallèle où plusieurs données sont envoyées en même temps, mais sur différents fils. Par nature, une liaison série est bien plus lente qu'une liaison parallèle, cependant l'I²C convient tout à fait à toutes les applications où la vitesse n'est pas primordiale.

5.1.2 Fonctionnement.

Un bus I²C contient trois fils :

- un signal de données (SDA) ;
- un signal d'horloge (SCL) ;
- un signal de référence (la masse).

Le périphérique qui gère la communication est le maître. C'est lui qui va générer le signal d'horloge et qui envoie les données (sauf acquittement *« Acknowledge »).*
Le signal d'horloge n'est pas une horloge véritable, dans le sens où les t_{ON} et t_{OFF} peuvent varier les uns par rapport aux autres.
L'acquittement est un bit envoyé par le composant esclave pour indiquer qu'il a bien reçu toutes les données ; si c'est le cas, l'esclave impose un niveau 0 sur la ligne de données (SDA), sinon la résistance de « Pull-up » (tirer vers le haut) maintient la ligne à un niveau haut ('1'). On dit alors qu'il n'y a pas d'acquittement.



Au début de la communication, SDA passe de 1 à 0 alors que SCL reste à 1 → c'est le Start Bit.

Après avoir imposé la condition de départ, le maître passe SCL à 0 puis applique sur SDA le bit de poids fort.

Il valide la donnée en appliquant pendant un instant un niveau haut ('1') sur la ligne SCL.

Lorsque SCL revient à '0', il recommence l'opération avec le bit inférieur jusqu'à ce que l'octet complet soit transmis.

Il redéfinit ensuite SDA comme une entrée et scrute son état ; l'esclave doit imposer un niveau bas pour signaler au maître que la transmission s'est effectuée correctement ; c'est l'acquittement. La communication peut donc continuer.
Si l'esclave n'envoie pas d'acquittement, les résistances de « Pull-up » maintiennent les lignes à un état haut. La communication peut alors être arrêtée ou reprendre depuis le début → c'est le Stop Bit.
Le Stop Bit, indiquant la fin de la transmission, est effectué en appliquant un passage de l'état bas à l'état haut de la ligne SDA alors que SCL reste à 1.

Le premier octet envoyé est l'adresse. Il est composé de 7 bits variables selon le composant et du bit de lecture/écriture (0 pour écrire, 1 pour lire).
Le second octet peut être l'octet de contrôle sur certains composants ou peut-être directement la donnée.

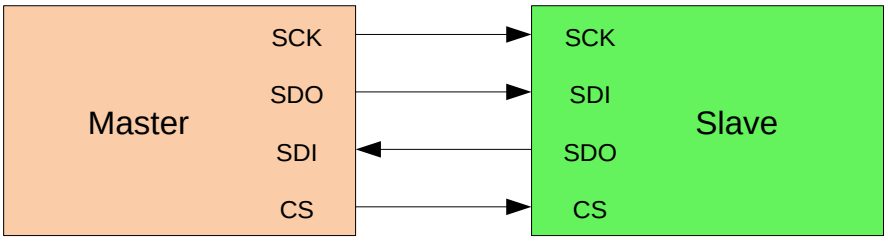
5.1.3 L'adressage.

Plusieurs composants peuvent être branchés sur le même bus I²C. Pour que l'information aille au bon endroit, chaque composant possède sa propre adresse.
Elle est composée d'une partie fixe imposée par le constructeur, d'une partie configurable de façon matérielle par l'utilisateur (par exemple, les Jumpers placés sur le capteur de température du sol « Module ION_CAPT »), et du bit de lecture/écriture qui définit le sens de la transmission.

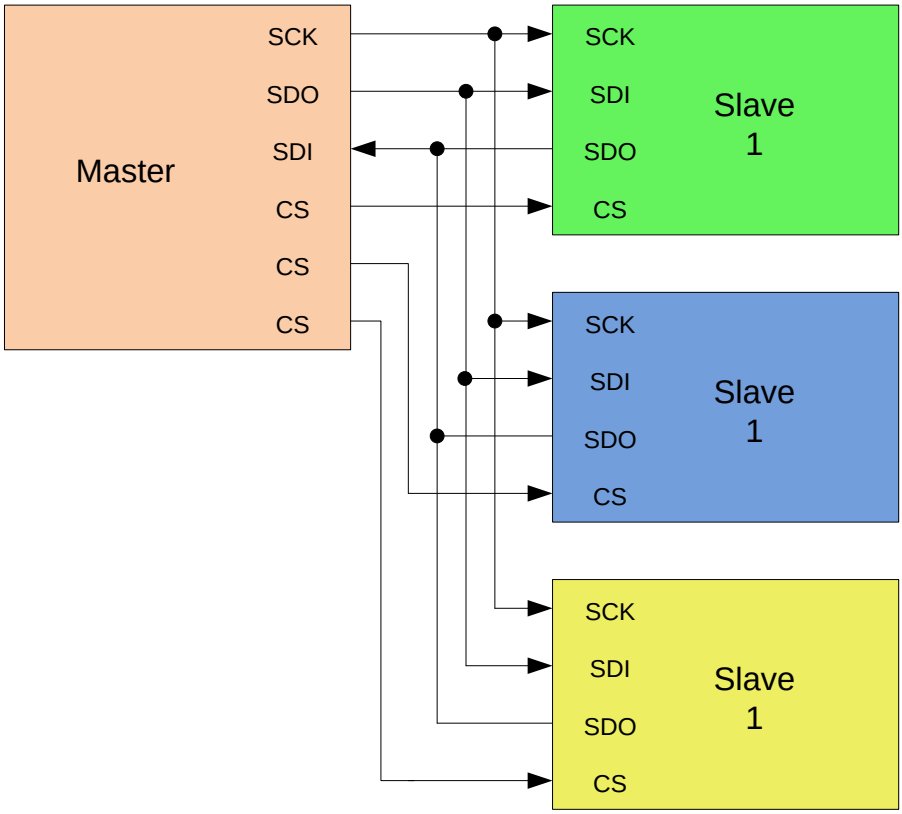
5.2 SPI (Serial Peripheral Interface).

5.2.1 Explication.

Une liaison SPI est un bus de données série synchrone baptisé par Motorola, opérant en mode Full Duplex. Les circuits communiquent selon un schéma maître-esclaves, où le maître s'occupe totalement de la communication. Plusieurs esclaves peuvent coexister sur un bus, la sélection du destinataire se fait par une ligne dédiée entre le maître et l'esclave appelée *Chip Select*.



Communication avec un seul esclave.



Communication avec plusieurs esclaves.

Le bus SPI possède 4 signaux logiques :

- SCK : Signal Clock : signal d'horloge généré par le maître ;
- SDI : Signal Data IN : les données entrantes dans le circuit sont véhiculées par ce signal ;
- SDO : Signal Data OUT : les données sortantes du circuit sont véhiculées par ce signal.
- CS : Chip Select : permet de sélectionner l'esclave avec lequel le maître veut communiquer (Actif à l'état bas).

Le SDO du maître doit être relié au SDI de l'esclave et inversement.

5.2.2 Fonctionnement.

Une transmission SPI est une communication simultanée entre un maître et un esclave. Le maître génère l'horloge et sélectionne l'esclave avec qui il veut communiquer. L'esclave répond aux requêtes envoyées par le maître.

A chaque front actif, le maître et l'esclave s'échangent un bit. Après 8 fronts, le maître a transmis un octet à l'esclave et vice-versa. La vitesse de l'horloge est réglée selon les caractéristiques propres aux périphériques.

5.2.3 Informations complémentaires.

Le bus SPI possède un débit plus important par rapport à un bus I²C. Le bus possède différents avantages :

- aucun arbitre nécessaire car aucune collision possible (deux fils de données) ;
- les esclaves utilisent l'horloge du maître et n'ont donc pas besoin d'oscillateur de précision ;
- communication Full Duplex.

Mais aussi quelques inconvénients :

- le bus monopolise plus de pins qu'un bus I²C ;
- aucun adressage possible, il faut une ligne de sélection par esclave en mode non chaîné ;
- le protocole n'a pas d'acquittement → le maître peut parler dans le vide sans le savoir (problème rencontré dans la communication avec le convertisseur analogique numérique) ;
- Il ne peut y avoir qu'un seul maître sur le bus.

5.3 CAN (Controller Area Network).

Ce bus est très utilisé chez M&D car il permet d'effectuer des échanges de données fiables et rapides. De plus, les sur-protocoles ont été redéfinis afin de mettre en œuvre un bus de communication très performant. Dans cette application, je l'utilise pour les échanges de données entre les différents modules. En effet, le module UBC envoie des trames CAN vers tous les modules chaque seconde. Ces trames peuvent contenir des commandes, des demandes de mesure et les réponses à ces trames sont envoyées par les modules distants.

Ce bus est donc très intéressant dans cette application.

5.3.1 Explications

Le bus CAN est un bus système série. Largement utilisé dans le secteur de l'automobile, il fut présenté par Intel en 1985. L'objectif de ce bus est de réduire la quantité de câbles en faisant communiquer les différents organes de commande sur un bus unique et non plus sur des lignes dédiées.
Sur le bus peut être branché tout appareil respectant les spécifications du bus.
Il existe pour le moment 2 normes couvrant la couche 2 du modèle OSI :

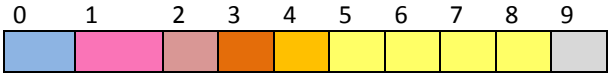
- Le CAN standard ; il possède un identifiant d'objet codé sur 11 bits et le CAN étendu ; il possède un identifiant d'objet codé sur 29 bits.

Ces deux normes sont compatibles. Sur un même réseau peuvent circuler des trames suivant la norme CAN standard et des trames suivant la norme CAN étendu. Mais pour des raisons de standardisation, nous préférons ne pas mélanger les deux normes.

L'accès au bus CAN suit la technique du CSMA/CR. Chaque station écoute avant de parler et chacune prend la parole à son tour → résolution des collisions par priorité.

L'émission d'une trame commence par l'émission de l'identifiant de la cible à atteindre. Les collisions sont résolues par un principe de « bit dominant » : si une station émet un '1' pendant qu'une autre émet un '0', c'est le '0' qui est transmis sur le support. La station qui aura émis le '1' verra qu'elle n'est pas la seule, elle saura qu'elle n'est pas la plus prioritaire et cessera d'émettre.

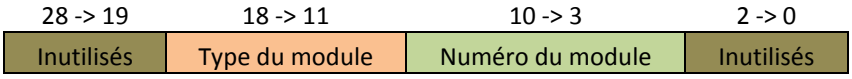
La communication se fait donc en série par le biais de trames. Voici la constitution d'une trame de commande :



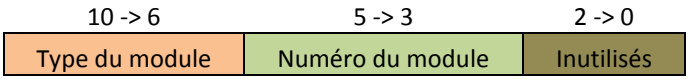
0	ID de la cible à **atteindre**
1	Type du module qui **parle**
2	Numéro du module qui **parle**
3	Fonction demandée
4	N° de registre visé
5 →8	Données
9	Checksum

L'ID de la cible à atteindre correspond à l'identificateur du module à atteindre. Il peut faire 11 ou 29 bits. Voici la décomposition de l'identificateur pour les deux configurations possibles :

En 29 bits :



En 11 bits :



Comme type de module, nous avons des I/O, des capteurs, etc.
Le numéro de module sert à identifier les modules de même type (par exemple : deux capteurs de températures identiques) à atteindre.
Différentes fonctions peuvent être demandées à chaque module ; des fonctions de lecture, d'écriture, de demande de paramètres, etc.
Le numéro de registre visé contiendra différentes valeurs, comme une donnée, un paramètre, une valeur d'initialisation. Lorsqu'un module contient différents capteurs, il faudra disposer de plusieurs registres contenant la mesure de chaque capteur. Chaque registre correspondant à un capteur spécifique.
Nous disposons ensuite de 4 octets de données. Ceux-ci peuvent contenir une mesure d'un capteur, un état logique d'un relais ou même un paramètre d'initialisation.
La trame de commande se termine par un Checksum pour détecter des erreurs éventuelles.

La trame de réponse à une commande possède la même structure et ce, pour des raisons de standardisation des protocoles et pour des raisons de facilités. Celle-ci est constituée de la sorte :

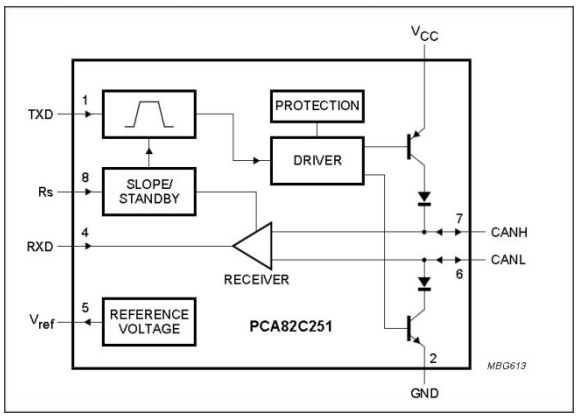
- un identificateur sur 11 ou 29 bits contenant le type et le numéro du module à atteindre ;
- le type du module qui répond ;
- le numéro du module qui répond ;
- la fonction réalisée par ce module ;
- le numéro de registre affecté par la commande ;
- les données et
- un Checksum.

Les données envoyées sur le bus CAN partent de la PIN CANTX du microcontrôleur tandis que les données reçues arrivent sur la PIN CANRX. Entre le microcontrôleur et le bus CAN, nous devons placer un driver CAN. Il s'agit du circuit intégré PCA82C251 de chez Philips.

5.3.2 Le circuit PCA82C251

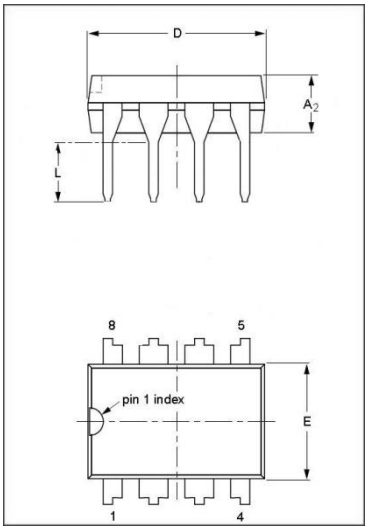
Ce circuit intégré permet de faire l'interface entre le contrôleur de protocole CAN et le bus physique. Il porte le nom de « Driver Can ».

5.3.2.1 Vue interne du composant :

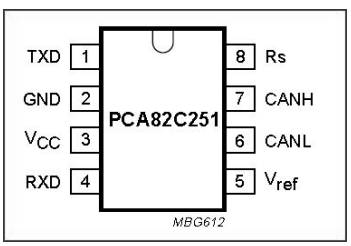


5.3.2.2 Dimensions du composant :

D (Longueur) : 9.8mm
A2 (Epaisseur) : 3.2mm
L (Longueur pin) : 3.6mm
E (Largeur) : 6.48mm



5.3.2.3 Brochage du PCA82C251 :



Pin1 → TXD: Transmit Data Input. Données reçues du µP
Pin2 → GND : Ground. Masse du circuit.
Pin3 → V_{CC}: Supply Voltage. Alimentation du circuit.
Pin4 → RXD: Receive Data Input. Données envoyées au µP
Pin5 → V_{ref}: Reference Voltage Output
Pin6 → CANL: Low level voltage CAN input/output.
Pin7 → CANH: High level voltage CAN input/output.
Pin8→ R_S: Slope resistor input.

5.4 RS485.

5.4.1 Explications générales.

Le bus RS485 est une transmission asynchrone. C'est-à-dire que les horloges de l'émetteur et du récepteur ne sont pas synchrones (car différentes). Il est donc nécessaire de synchroniser la transmission (Start Bit). Il existe deux types de transmission asynchrone :

- la transmission asynchrone en mode caractère : le récepteur et l'émetteur doivent utiliser les mêmes vitesses de transmission et le même formatage pour le caractère (nombre de bits de données par caractère, bit de parité ou non et type de parité, nombre de stop bits). La trame se réduit donc à un caractère ;
- la transmission asynchrone en mode bloc de caractères : le récepteur doit pouvoir reconnaître le début et la fin du bloc et pouvoir se synchroniser. Une manière simple de faire consiste à encapsuler le bloc entre deux caractères spécifiques STX et ETX. Ces caractères spécifiques seront précédés de 2 caractères SYN qui vont permettre la synchronisation des horloges.

Notons que la transmission asynchrone n'est pas très efficace lorsque le débit de transmission croît.

Au niveau des interfaces, la couche physique précise :

- les aspects électriques ;
- les aspects mécaniques ;
- les aspects fonctionnels ;
- les aspects procéduraux des équipements et supports de transmission.

Ces aspects permettent généralement de définir les conventions d'échanges entre le DTE (équipement terminal de traitement des données) et le DCE (équipement terminal de circuit de données).
Le DTE est donc relié au DCE via une interface.
Différentes normes d'interface ont été établies.
Les normes sont généralement divisées en quatre parties, pour les aspects mécanique, électrique, fonctionnel et procédural mais une norme peut préciser un ou plusieurs aspects.

1.) L'aspect mécanique correspond à la jonction physique entre le DTE et le DCE. Il définit le format des connecteurs, la longueur maximale du câble reliant deux connecteurs,…
2.) L'aspect électrique détermine le niveau de tension et les transitions, la méthode d'équilibre de la tension par rapport à la mise à la terre.
3.) L'aspect fonctionnel spécifie les fonctionnalités de chaque broche, les directions de transfert de chaque broche. Les fonctions sont classées en quatre catégories : les données, le contrôle, la synchronisation et la mise à la terre.
4.) L'aspect procédural décrit la séquence des évènements pour la transmission de données en se basant sur les caractéristiques fonctionnelles de l'interface. Le comportement de l'interface est décrit sous forme d'automate à états finis.

La norme RS485 est une extension de la norme RS422A permettant des liaisons multipoints ainsi que point à point : les transmetteurs peuvent être mis à l'état haute impédance. Il peut y avoir 32 émetteurs et 32 récepteurs sur un câble. Il faut prévoir des résistances de terminaison aux deux extrémités de la ligne.

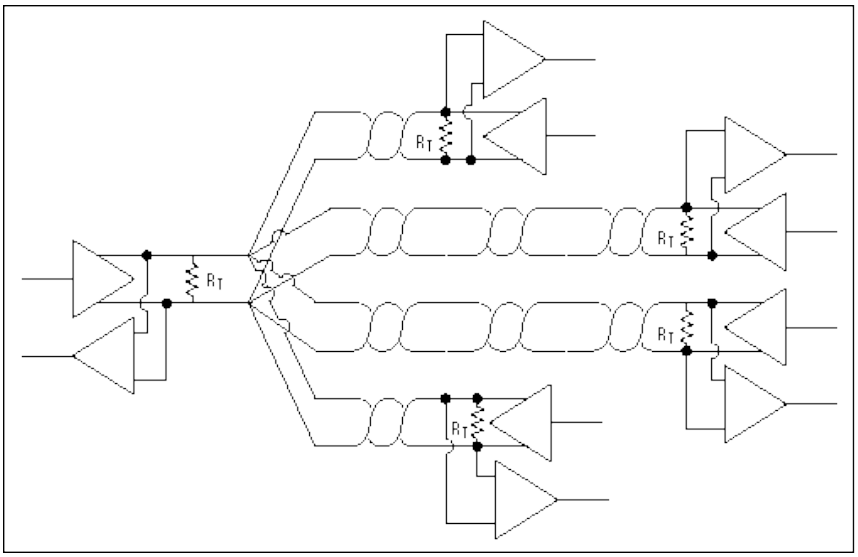


Figure 29 Raccordement multipoints

5.4.2 Rappel sur la norme RS422A.

La norme RS422A définit un mode de transmission différentiel. Il faut donc deux fils pour véhiculer un signal. Ce dernier n'est pas référencé à une masse et est présenté comme un signal différentiel aux sorties du transmetteur et aux entrées du récepteur.

L'emploi d'un dispositif de terminaison est préconisé afin de boucler la ligne sur son impédance caractéristique. Ce montage permet de minimiser le bruit (mode symétrique) et les réflexions assurant ainsi une meilleure qualité de transmission. Généralement le bouclage est effectué à une extrémité de la ligne. Moyennant l'utilisation de paires torsadées et de dispositifs de terminaison de ligne, on peut atteindre des débits de l'ordre de 10 Mbits/s sur 12 mètres et de 100 Kbits/s sur 1200m.

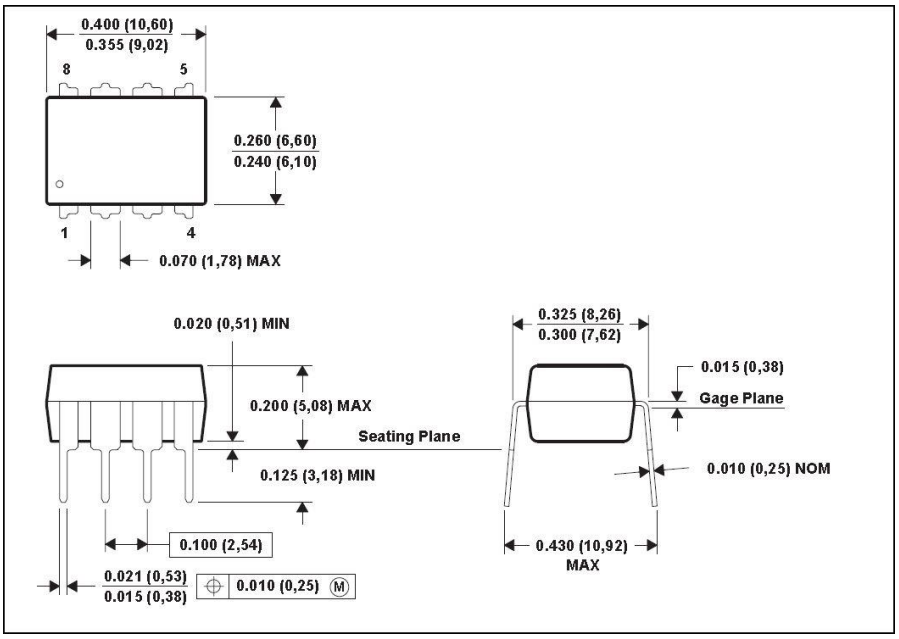
5.4.3 Résumé des caractéristiques de la norme RS485.

- Mode de fonctionnement : différentiel.
- Nombre total de transmetteurs/récepteurs : 32 transmetteurs/32 récepteurs.
- Longueur du câble maximum : 1200m.
- Vitesse de transmission maximum : 10Mbits/s sur 12m et 100Kbits/s sur 1200m.
- Tension maximum en mode commun : de -7V à +12V.
- Tension de sortie minimale en charge : ± 1.5V.
- Tension de sortie maximale en circuit ouvert : ± 6V.
- Impédance transmetteur : 54Ω.
- Plage de tension entrée récepteur : de -7V à +12V.
- Sensibilité entrée récepteur : ± 200mV.
- Impédance entrée récepteur : ≥ 12kΩ.

5.4.4 Le circuit SN75ALS176.

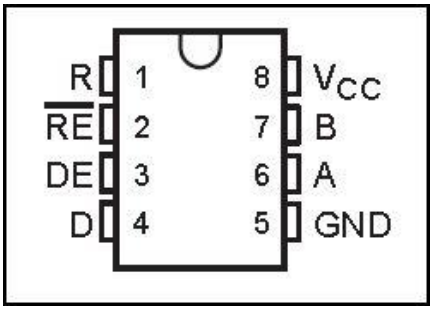
Ce circuit intégré permet de faire l'interface entre le contrôleur de protocole RS485 et le bus physique. Il porte le nom de « Driver RS ».

5.4.4.1 Dimensions du composant.



Les dimensions entre parenthèses sont en millimètres.

5.4.4.2 Brochage du SN75ALS176.



Pin1 → RX: Les données reçues via le bus sortent par cette Pin.
Pin2 → EN_TX : Activation de la transmission.
Pin3 →EN_RX: Activation de la réception.
Pin4 →TX: Les données envoyées via le bus rentrent par cette Pin.
Pin5 → GND: Ground. Masse du circuit.
Pin6 → A: Signal connecté au bus.
Pin7 → B: Signal connecté au bus.
Pin8→ V_{CC}: Alimentation du circuit.

Ce circuit effectue le même « travail » que le circuit PCA82C251 (Driver CAN).

6. Explication des différents logiciels utilisés.

Lors de mon travail, j'ai utilisé deux logiciels mis à ma disposition, voici chacun d'eux :

6.1 Le logiciel MPLAB IDE.

Ce logiciel est un environnement de développement intégré. C'est un éditeur de texte qui permet d'écrire des programmes en C, de compiler ces programmes (code C transformé en code machine) et de les transférer dans les microcontrôleurs (PIC).

6.1.1 Matériel nécessaire.

Figure 30 Connexion avec un PC

Figure 31 Kit MPLAB IDE

- Un ordinateur avec un port USB.
- Le logiciel MPLAB IDE version 8.10.
- Le programmeur PICkit 2.
- Des microcontrôleurs PIC de chez Microchip.

6.1.2 Vue d'ensemble du programme.

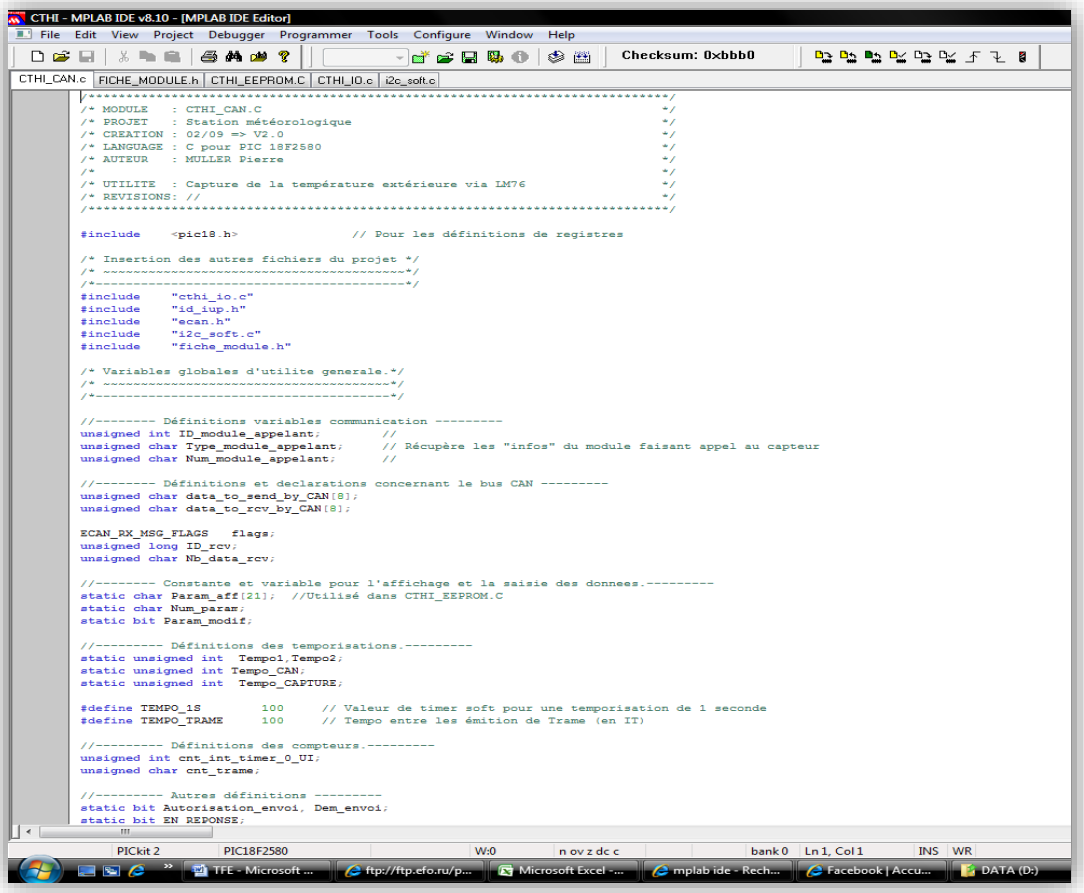


Figure 32 Fenêtre principale

6.1.3 Explication de la barre d'outils.

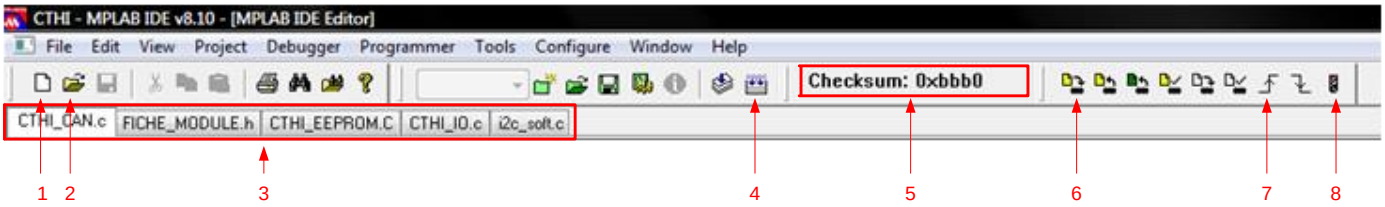


Figure 33 Barre d'outils

1 : Créer un nouveau fichier.
2 : Ouvrir un fichier.
3 : Fichiers ouverts dans le projet.
4 : Compiler
5 : Checksum
6 : Envoyer le programme dans la cible.
7 : Placer le Master Clear de la cible à Vdd
8 : Permet de vérifier si le PIC est bien connecté à la console de programmation, s'il s'agit du PIC adéquat etc.

6.1.4 Fenêtre « Output - Build».

Lorsque l'on clique sur le bouton « Compiler » (4 sur la figure 18), la fenêtre suivante apparaît :

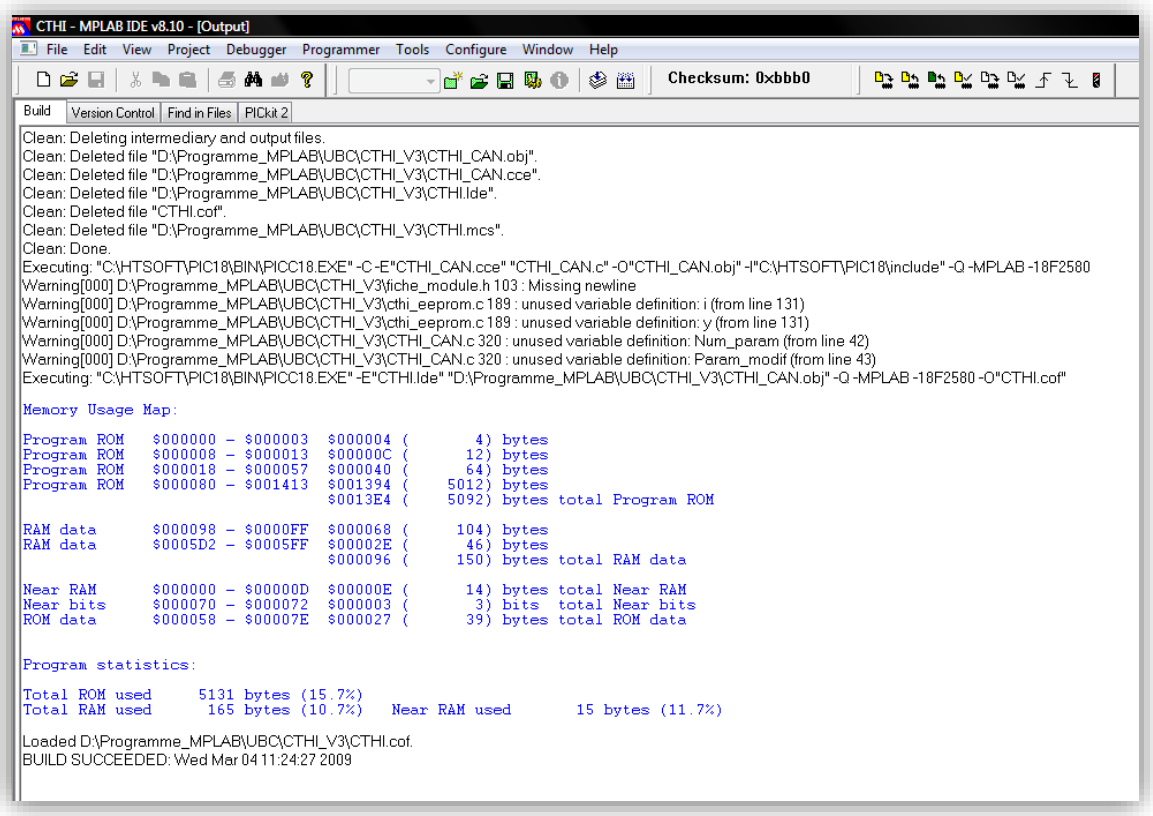


Figure 34 Fenêtre "Build"

Sur cette fenêtre, on peut voir les différents Warnings ainsi que les erreurs faisant partie du programme. Si la ligne « Build succeeded » apparaît, on peut programmer le PIC par la suite, le programme ne contient pas d'erreur.

6.1.5 Fenêtre « Output – PICkit 2 ».

Lorsque l'on clique sur le bouton « Compiler » et ensuite sur le bouton « 8 », la fenêtre suivante apparaît :

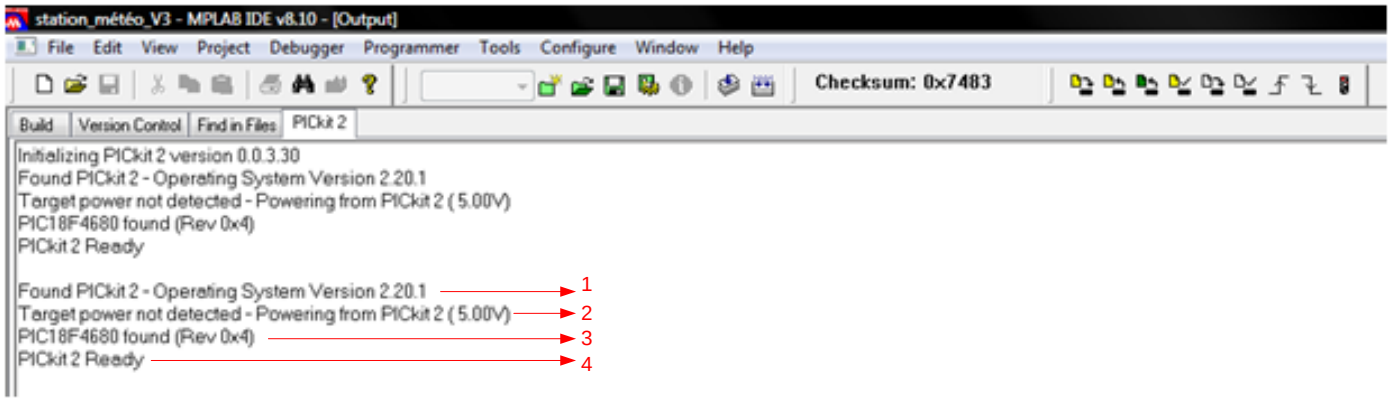


Figure 35 Fenêtre Output

1 : Le PICkit 2 a bien été trouvé.

2 : Aucune puissance sur la cible n'a été détectée.

3 : C'est le bon PIC qui a été connecté.

4 : Prêt à programmer le PIC (pour cela, il faut cliquer sur « Envoyer le programme dans la cible »).

Différentes erreurs peuvent se produire :

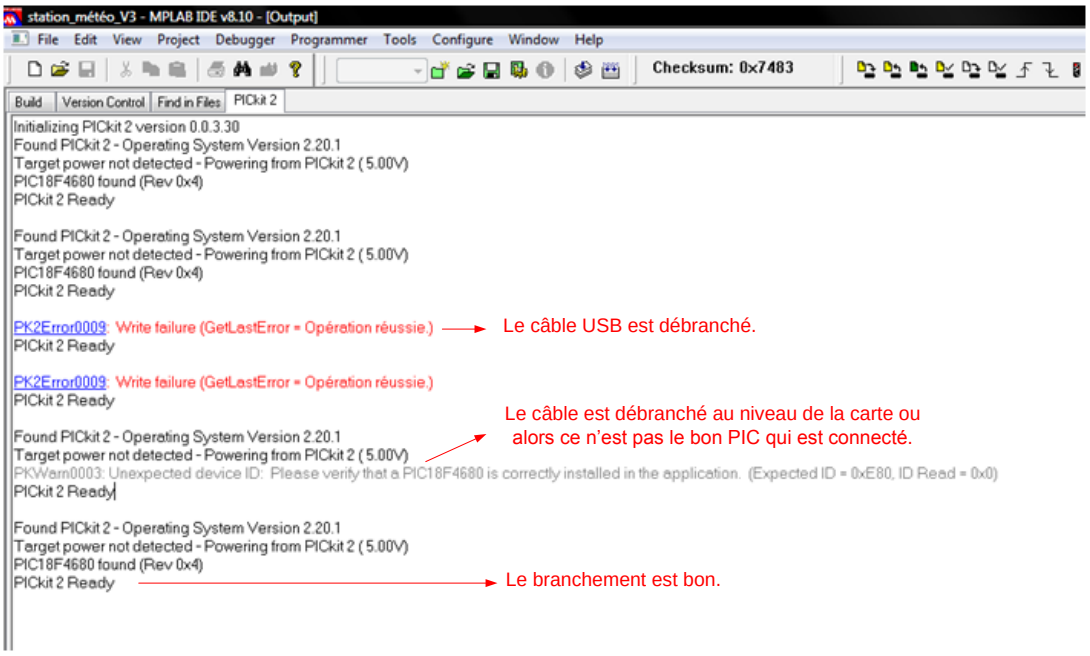


Figure 36 Erreurs affichées dans la fenêtre Output

6.1.6 Programmation du PIC.

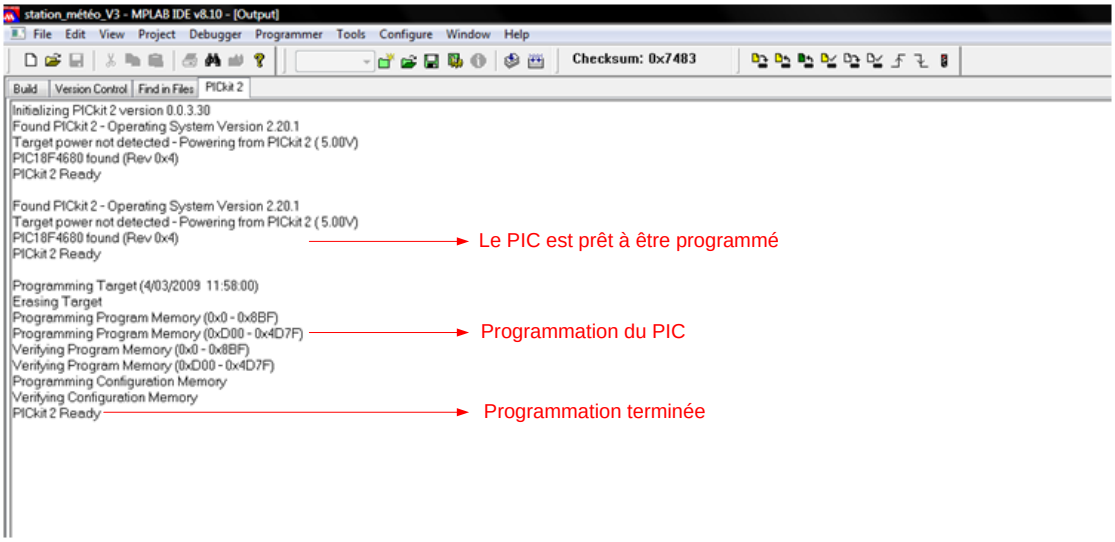


Figure 37 Programmation du PIC

6.1.7 Conclusions.

Grâce à ce logiciel, il m'est donc possible de développer le programme qui sera exécuté par le microcontrôleur et de le programmer dans celui-ci. Le langage utilisé est le langage C et toutes les fonctions utilisées par celui-ci peuvent être utilisées avec MPLAB IDE. C'est le programme avec lequel j'ai le plus travaillé.

6.2 Le logiciel EAGLE.

Ce logiciel permet de dessiner des cartes électroniques. En partant d'un schéma électrique (que l'on dessine), on obtient un dessin de PCB sur lequel les composants peuvent être placés et disposés à notre guise.

6.2.1 Vue d'ensemble du programme.

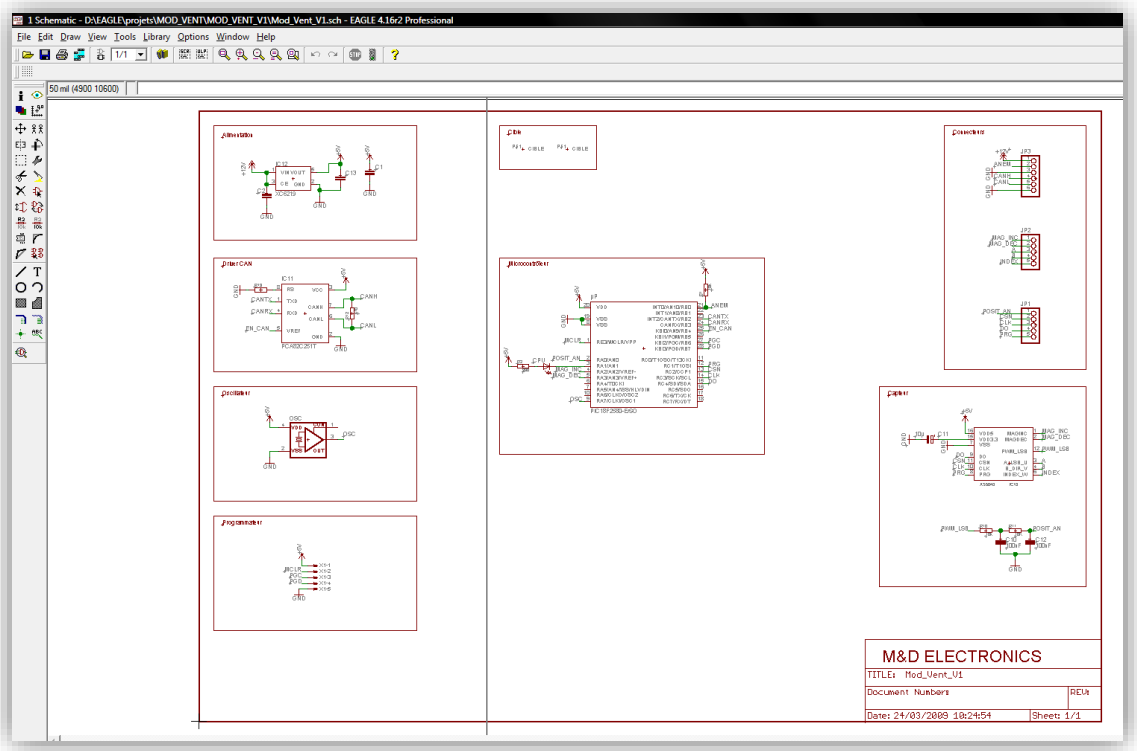


Figure 38 Fenêtre "Schematic"

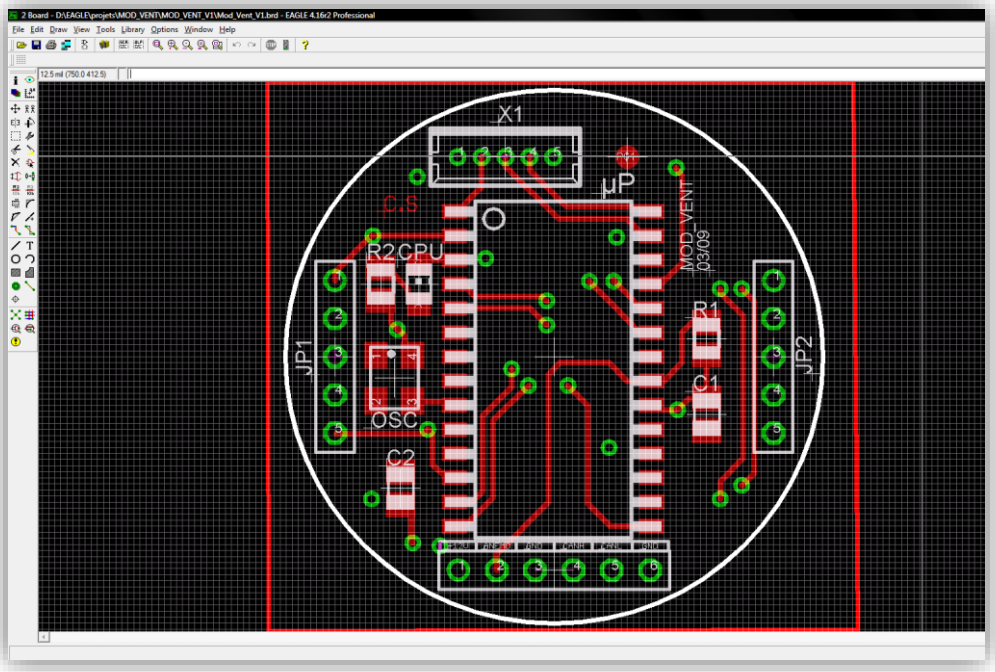


Figure 39 Fenêtre "Board" (dessin du PCB)

6.2.2 Explication de la fenêtre « Schématic ».

6.2.2.1 Barre d'outils.

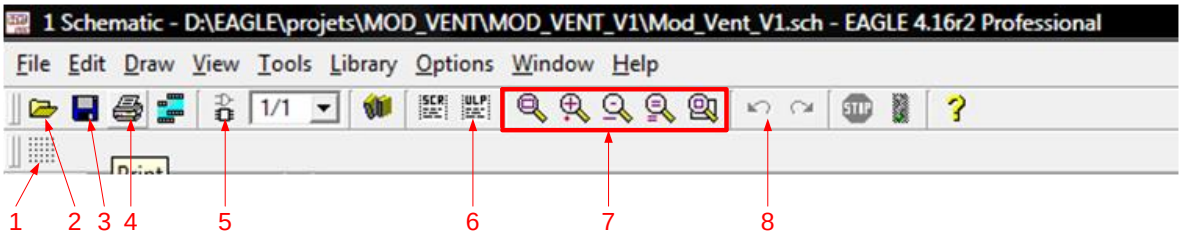


Figure 40 Barre d'outils

1. Appliquer la grille sur le schéma, ou changer les options du curseur.
2. Ouvrir un fichier.
3. Enregistrer.
4. Imprimer le schéma.
5. Passer sur la fenêtre « Board ».
6. Ouvrir des utilitaires (ces utilitaires ont différentes fonctions, comme par exemple zoomer les parties de schéma où les connexions ne sont pas établies, renuméroter les différents composants placés dans le schéma etc).
7. Effectuer un zoom (on peut aussi utiliser le scroll de la souris).
8. Retour en arrière.

6.2.2.2 Barre de dessin.

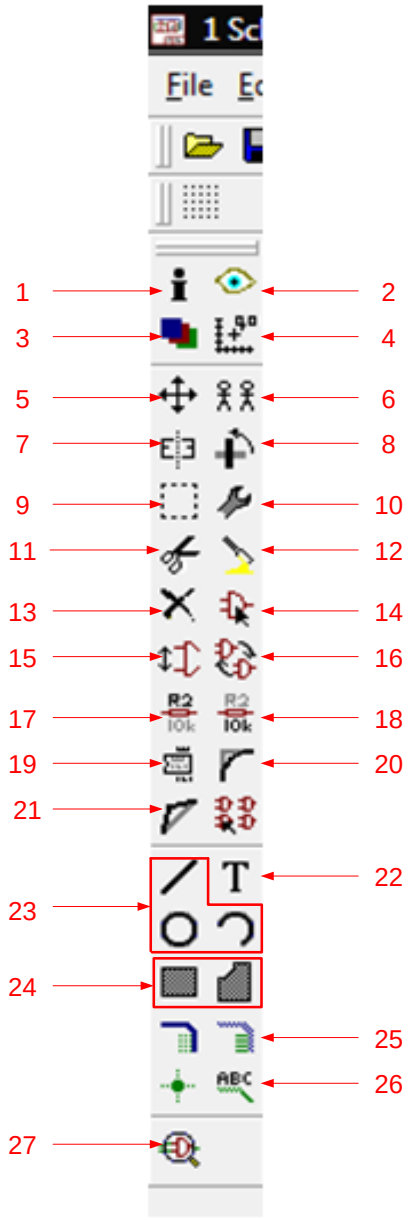
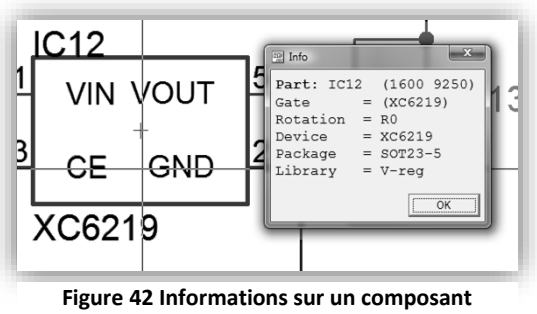


Figure 41 Barre de dessin

1. Pour obtenir des informations sur un composant (sa taille, sa librairie, son package,…) *(Voire la photo plus loin).*
2. Pour voir les différents liens dans le circuit.
3. Pour modifier l'affichage (couches affichées à l'écran etc.).
4. Déplacer l'origine du dessin (axe X, Y).
5. Déplacer un composant ou un groupe de composants.
6. Copier un composant.
7. Effectuer une symétrie axiale du composant (effet miroir).
8. Effectuer une rotation (si composant sélectionné, click droit avec la souris et une rotation est effectuée)
9. Effectuer un groupement de composants (pour les déplacer, les copier, …).
10. Changer une caractéristique (l'épaisseur d'un trait, le texte, la taille du texte,…).

11. Effectuer un couper.
12. Effectuer un coller.
13. Supprimer un composant ou un groupe de composants.
14. Insérer un composant venant de la bibliothèque *(Voire la photo plus loin)*.
15. Changer les pins d'un composant.
16. Inverser deux composants.
17. Ajouter un nom au composant.
18. Ajouter une valeur au composant.
19. Dissocier le nom du composant (pour déplacer le nom par exemple).
20. Effectuer un arc.
21. Eclater un trait en plusieurs segments.
22. Insérer une zone de texte.
23. Dessiner différents types de traits.
24. Insérer un polygone
25. Effectuer un lien (le lien étant la connexion entre les composants).
26. Afficher le nom d'un lien (en cliquant sur le lien).
27. Appliquer l'outil ERC qui va permettre de voir les erreurs dans le schéma (les composants qui ne sont connectés à rien, les erreurs de connexion,…).



Informations obtenues en cliquant sur « Info » (1).

Figure 42 Informations sur un composant

6.2.2.3 Librairie des composants.

Quand on ouvre la librairie, voici la fenêtre que l'on obtient :

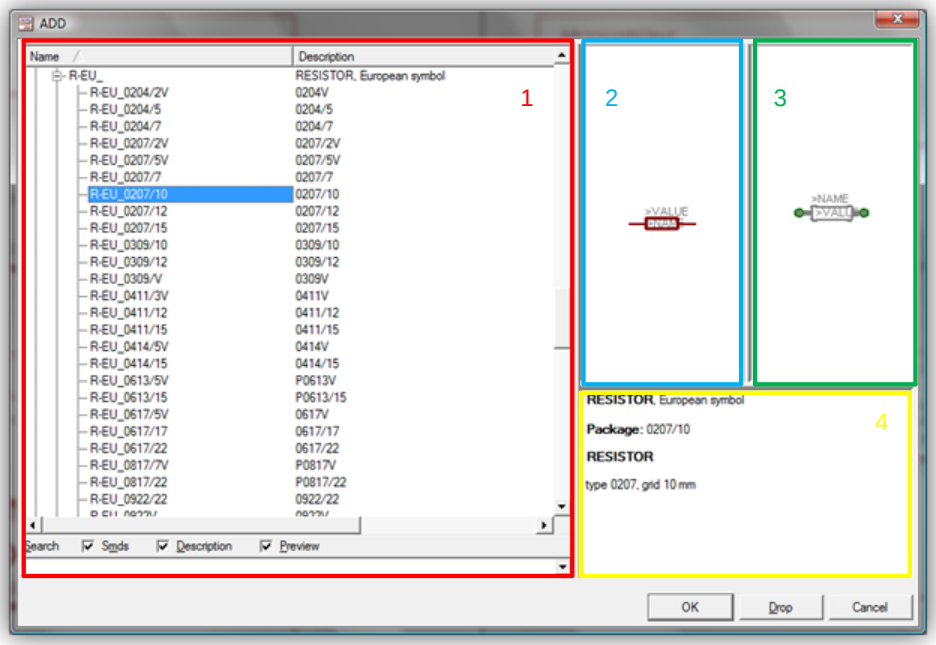


Figure 43 Fenêtre "bibliothèque"

On peut trouver dans la fenêtre 1, la liste de tous les composants à insérer dans le schéma (Il est possible de créer ses propres composants). Dans la fenêtre 2, on trouve le symbole qui figurera dans le schéma électrique. Dans la fenêtre 3, on trouve le package du composant (c'est le package qui figurera dans la fenêtre « Board »). Enfin, dans la fenêtre 4, on y trouve diverses informations.

Grâce à la fenêtre « Schematic », j'ai pu dessiner le schéma électrique de mes différentes cartes électroniques faisant partie du projet. Ce schéma reprend tous les composants et les liens entre eux. Une fois un composant inséré et placé dans le schéma, celui-ci se retrouve dans la fenêtre « Board » où il sera placé sur le PCB à proprement parlé.

6.2.3 Explication de la fenêtre « Board».

Une fois un composant inséré dans le schéma, il se retrouve dans la fenêtre « Board ». Il reste à le placer sur le PCB. Le placement d'un composant doit être judicieux. En effet, il faut veiller à ce que les traces (pistes de cuivre) soient les plus courtes possible. Certains composants peuvent nuire au fonctionnement des autres, il ne faut donc pas les placer côte à côte.

La barre d'outils et la barre de dessin sont similaires à celles de la fenêtre « Schematic ». Notons qu'il est permis d'effectuer en plus des traces[7], de les supprimer, et d'utiliser un outil de correction appelé DRC. Cet outil, sur base d'un fichier à charger, permet de vérifier plusieurs paramètres (espacement toléré entre composants, espacement entre traces, espacement d'un composant et le bord de la carte,…). Cet outil est nécessaire afin de respecter certaines tolérances de fabrication et de fonctionnement de la carte électronique.

[7] Une trace est une piste de cuivre permettant d'effectuer des liaisons entre les composants.

7. Conclusion.

Au terme de mon stage, la station météorologique est fonctionnelle. Diverses modifications devront sans doute être appliquées pour faire face aux défauts qui pourraient apparaître lors de l'utilisation. Néanmoins, nous pouvons observer que la mission initiale a été relevée.

Du point de vue de la station, quelques améliorations pourraient être apportées. Prenons pour exemple la modification de toutes les cartes afin que l'électronique ne dépende que d'un seul panneau solaire. Il serait également possible de modifier la carte UBC_IO afin qu'elle permette de créer une communication GSM et une communication RS485 simultanément. Car pour le moment, on ne peut choisir qu'une seule des deux : Soit le GSM, soit le RS485.

L'approche du monde de l'entreprise s'est avérée très enrichissante : prendre des décisions et faire preuve d'une certaine autonomie représentent un apprentissage instructif.

Ce stage professionnel a considérablement amélioré mes aptitudes techniques et m'a permis de découvrir le métier d'automaticien et plus particulièrement celui de développeur électronicien. De plus, j'ai pu élargir mes connaissances avec l'apprentissage du montage et de la soudure de cartes électroniques relatives à d'autres projets que le mien.

Le caractère enrichissant et instructif de ce stage a contribué à la réussite de ce dernier. De plus, il m'a permis de rapprocher les aspects théoriques et pratiques de l'automation.

Je retiendrai du métier d'automaticien qu'il nécessite une constante mise à jour des connaissances étant donné les exigences actuelles du marché. Cette vision rend ce métier très attrayant.

En définitive, je reste reconnaissant pour la chance qui m'a été donnée de pouvoir mettre mes connaissances à profit mais aussi pour la confiance qui m'a été confiée afin de finaliser au mieux mon projet. Ce stage représente la meilleure transition possible entre les études et la vie active.

8. Listing du programme interne.

Le listing des programmes intégrés dans chaque PIC étant trop important, il est placé dans le tome 2 « Listing des programmes ».

ANNEXES