GEII
Département Génie Électrique
& Informatique Industrielle
IUT Belfort-Montbéliard

RAPPORT DE PROJET

Développement d'une électronique embarquée communicante associée à un PSOC

Abdramane CISSE
Victor BECHTEL
Année universitaire 2016-2017
Rapport de projet | LP VEGA

Enseignant : C. LOMBARD

UFC
UNIVERSITÉ DE FRANCHE-COMTÉ
IUT Belfort-Montbéliard
INSTITUT UNIVERSITAIRE DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD
Tél. +33 (0)3 84 58 77 00 • iut-bm.univ-fcomte.fr

Sommaire

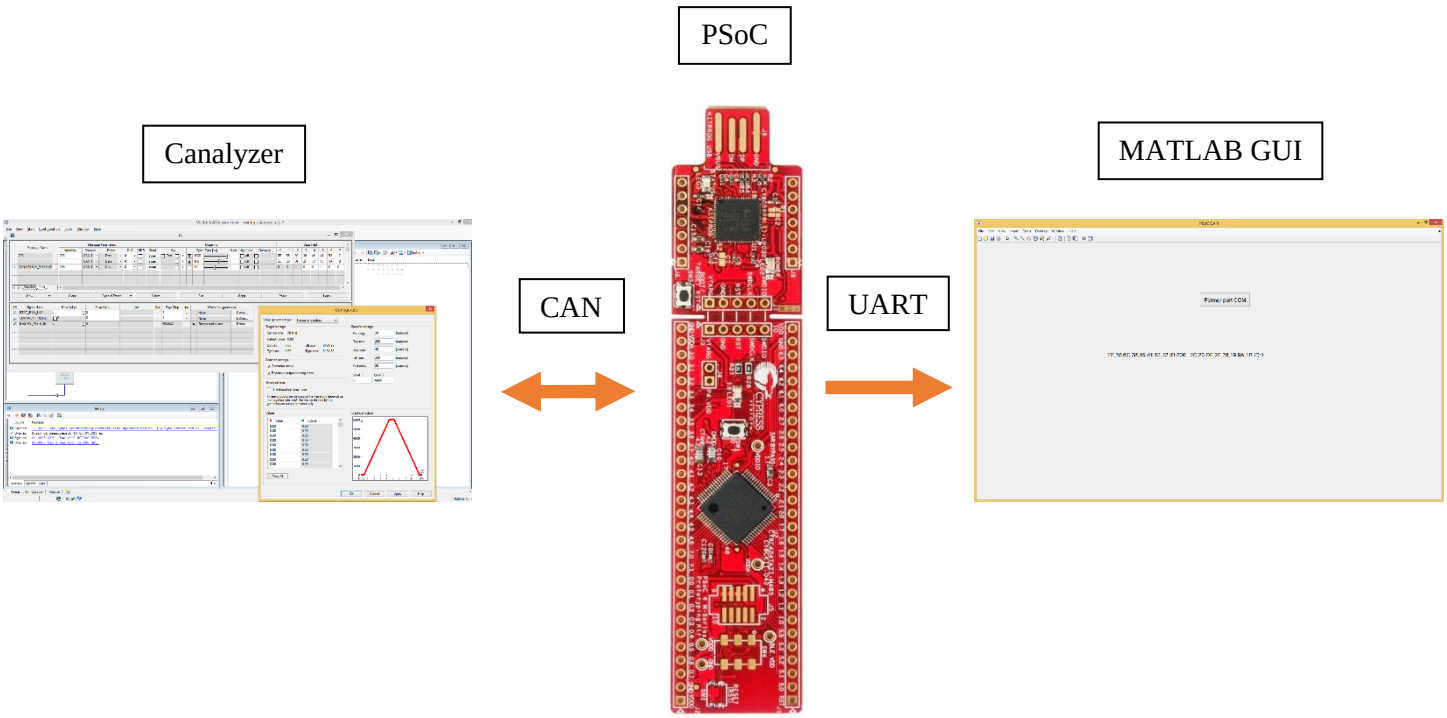
I. Introduction

1. Sujet et objectifs

Sujet : Il est demandé de fournir une organisation matérielle et logicielle détaillée capable de récupérer des trames réseau CAN2A, de les traiter (PSOC) et de communiquer à distance les informations pour un affichage sur une interface graphique à développer (Matlab GUI). Les trames réseaux seront dans un premier temps issues d'un générateur puis ensuite élaborer à partir d'un capteur automobile.

Objectifs : Décoder et visualiser le contenu de trames réseau CAN sur une interface graphique distante (GUI Matlab).

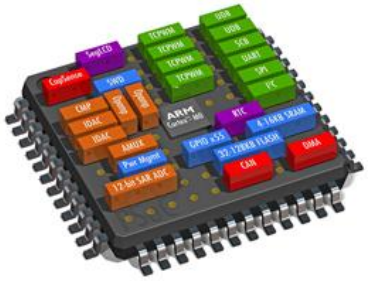
2. Schéma synoptique



II. Description technique

1. PSoC

a. Qu'est-ce que le PSoC ?

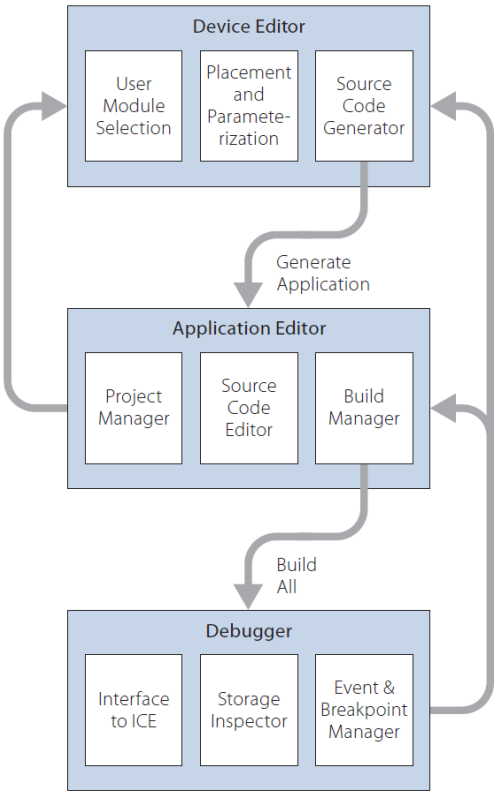


Le PSoC (Programmable System on Chip) est un composant associant un microcontrôleur et des fonctions logiques et analogiques configurables et interconnectables entre eux dont le but est de réduire le nombre de composants.

Blocs logiques et analogiques configurables :

- Blocs logiques configurables en compteurs, timers, UARTs, SPI, générateurs CRC, séquences pseudo aléatoires, etc…
- Blocs analogiques configurables en amplis-op simples, comparateurs, filtres, CNAs, CANs, modem, etc…

Les PSoC, ou processeurs à signaux mixtes (mixed signal arrays) sont des puces qui permettent de gérer des informations aussi bien numériques qu'analogiques. On sait depuis longtemps intégrer des convertisseurs analogiques-numériques au sein de microcontrôleurs, mais l'intégration de fonctions analogiques sur le silicium (gains programmables, filtres à capacités commutées, etc.) est relativement récente.

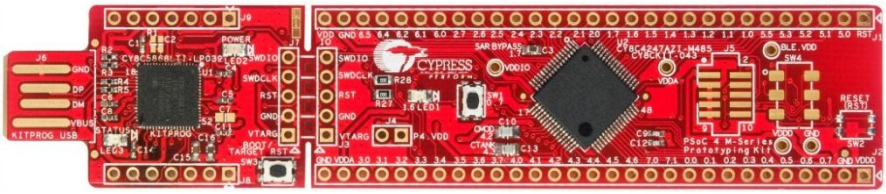


L'approche de la programmation du PSoC est différente de celle des microcontrôleurs que nous utilisons habituellement (Microchip).

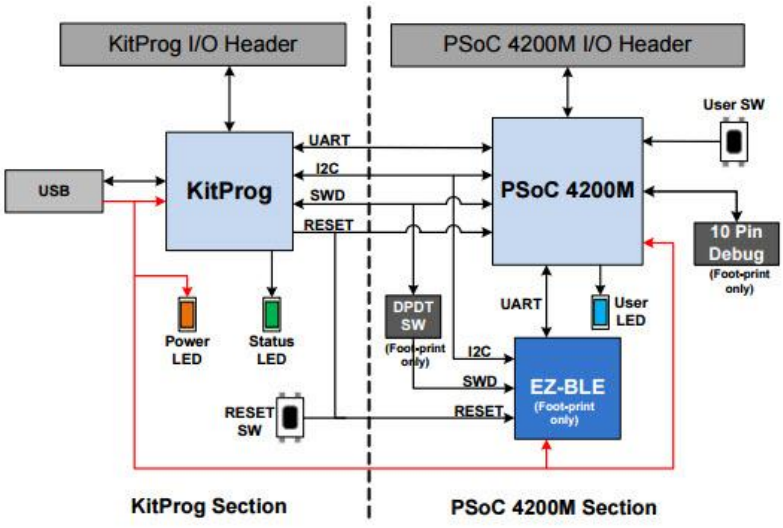
En effet, les PSoC de chez Cypress se programment à l'aide du logiciel PSoC Creator.

La conception d'un nouveau projet nécessite le passage par 3 étapes :

- Placement et connexion des modules
- Configuration des modules et des drivers
- Codage, compilation et débogage



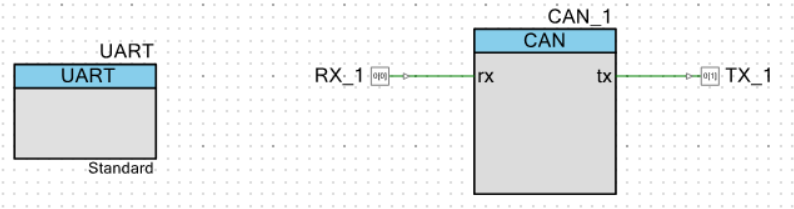
Le PSoC que nous utilisons est un CY8C4247AZI-M485 issu de la famille 4200M de chez Cypress. La carte accueillant le PSoC est une CY8CKIT-043.



La carte est composée de deux parties. Une partie programmateur sécable, une fois que la programmation est terminée. Cette dernière est composée d'un connecteur USB permettant l'alimentation de la carte, et des broches d'UART et d'I2C. L'autre partie de la carte est composée du PSoC. On y retrouve un bouton poussoir et une LED témoin pour faire des essais. On retrouve des broches dédiées aux différents moyens de communications intégrés (CAN, I2C, UART…) et de nombreuses broches d'entrées/sorties.

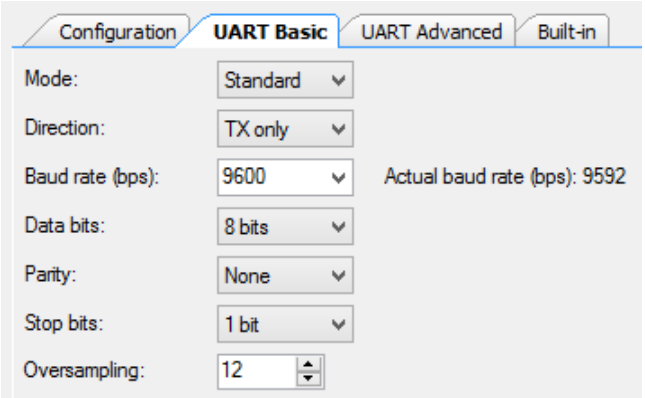
b. Programmation PSoC

Partie Graphique

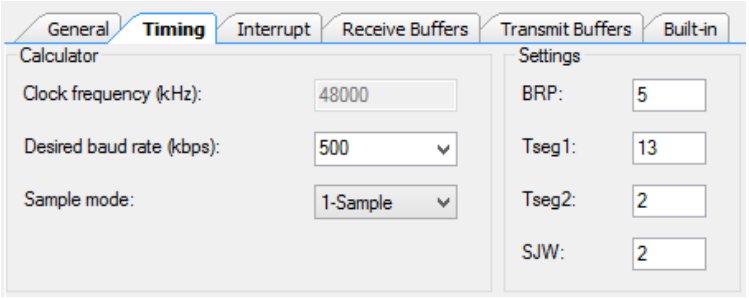


Dans la partie graphique, nous retrouvons simplement un bloc UART et un bloc CAN.

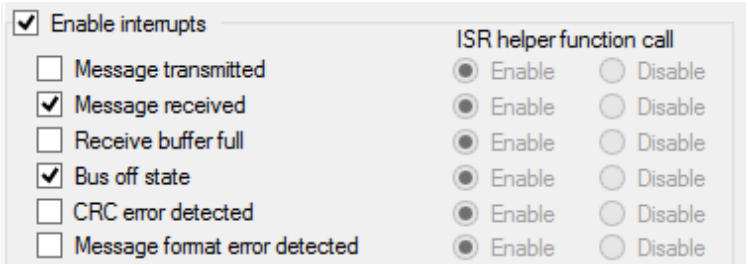
Configuration du bloc UART



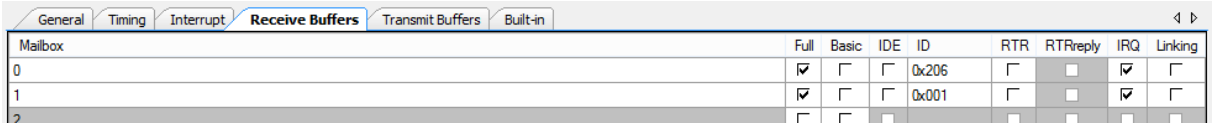
Configuration du bloc CAN



Vitesse réseau CAN : 500 kbps



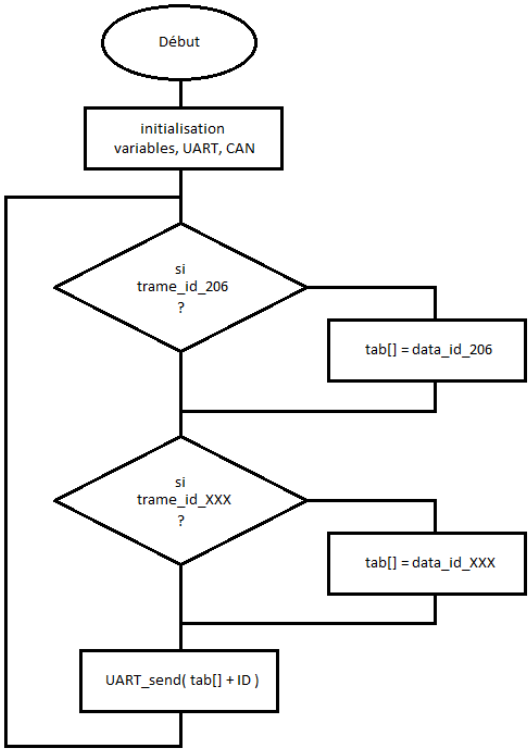
Interruptions à la réception d'un message



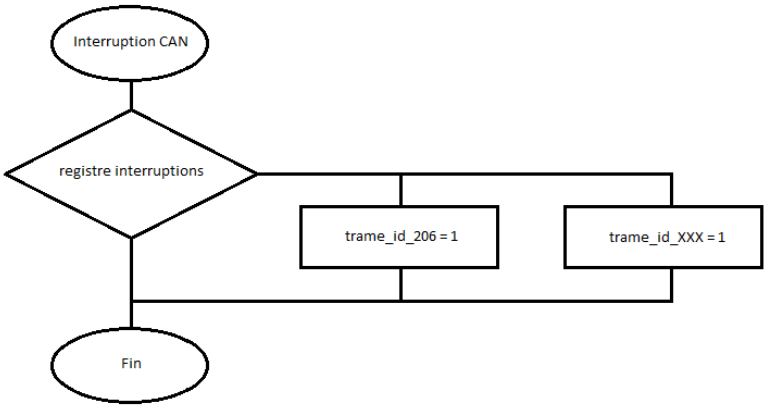
Buffers de réception (ici trames ID 206 et ID 001)

Partie Code

Programme principal



Interruptions



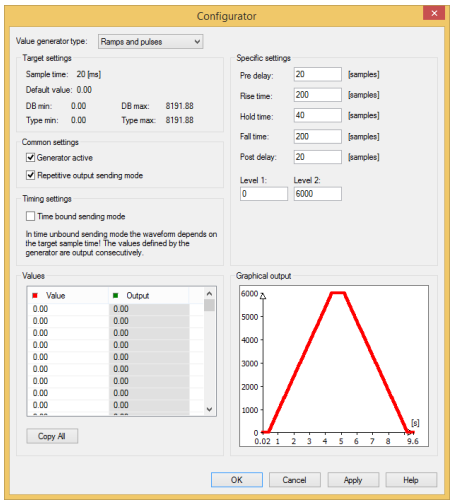
Ci-dessus deux logigrammes expliquant le fonctionnement du programme que nous avons développé.

Au début du programme, on initialise les différentes variables, les modules CAN et UART. Si le PSoC reçoit une trame dont l'ID correspond à sa boîte de réception alors une interruption se déclenche. C'est à ce moment que le programme principal traite les données reçues et les place dans un tableau. Les données des trames ainsi stockées sont transmises sur l'UART.

2. Canalyzer – exemple transmission d'informations

Afin de générer des trames CAN, nous utilisons le logiciel Canalyzer sur un ordinateur.

Côté Canalyzer



Par exemple nous émettons une trame d'ID 208 (trame DYNAMIQUE_MOTEUR).

Les deux premiers octets constituent la donnée REGIME_MOTEUR.

On applique une évolution en rampes sur cette donnée afin de la faire fluctuer.

0.019986 CAN 1 208	DYNAMIQUE_MOTEUR		CAN Frame	Tx	8	8	B8 B4 00 00 00 00 00 00
REGIME_MOTEUR	5910.5000	B8B4					
CONTACT_FREIN2	0	0					
ETAT_RVV_LVV	0	0					

Côté PSoC

Du côté PSoC, la trame est récupérée, chaque donnée est mise dans un tableau. Ensuite, toutes les données de ce tableau sont transmises, séparées chaque fois par une virgule.

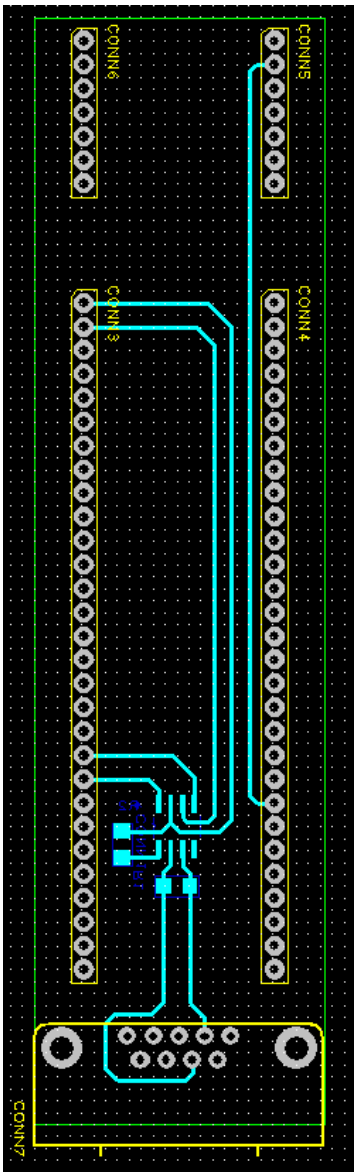
```
sprintf(chaine,"%u,%u,%u,%u,%u,%u,%u,%u\n",val[0],val[1],val[2],val[3],val[4],val[5],val[6],val[7]);
UART_UartPutString(chaine);
```

Côté GUI MATLAB

Du côté GUI MATLAB, on décompose la chaîne reçue afin d'extraire les données qui nous intéressent. La fonction str2num(chaîne) permet de séparer les données de la chaîne. Les virgules servent de séparateurs pour constituer un tableau de valeurs. On fait les opérations nécessaires sur les deux premières valeurs pour restituer la valeur réelle de régime moteur (en trs/min ici) et on l'affiche dans le champ String de la zone de texte « text ».

```
trame = fgetl(s);
trame_val_num = str2num(trame);
text.String = sprintf('%.0f trs/min',(trame_val_num(1)*256 + trame_val_num(2))*0.125);
```

3. PCB



Suite aux tests concluants effectués sur plaquette labdec, nous avons conçu un circuit imprimé avec le logiciel DesignSpark PCB.

Ayant précédemment soudé des broches males sur le dessous du PSoC (pour permettre son utilisation sur plaquette labdec), nous décidons de faire une carte avec broches femelles qui pourra donc s'enficher en dessous du PSoC.

Aussi nous optons pour des composants montés en surface plutôt que des traversants pour des raisons d'encombrements et de modernité tout simplement.

Sur le PCB nous trouvons donc en terme de connexions un connecteur DB9 pour la liaison CAN et des broches femelles pour les différentes liaisons avec le PSoC. Au centre du routage se trouve un MCP2551, un transceiver CAN. Les deux autres composants que l'on retrouve sont deux résistances. La première est de 120 ohms entre les lignes High et Low du bus CAN, et l'autre vaut 10 ohms et se trouve entre la masse et la patte RS du transceiver.

On vient aussi ponter la sortie de l'UART pin 22 vers la ligne de communication côté programmateur USB afin de faire remonter des informations vers un ordinateur.

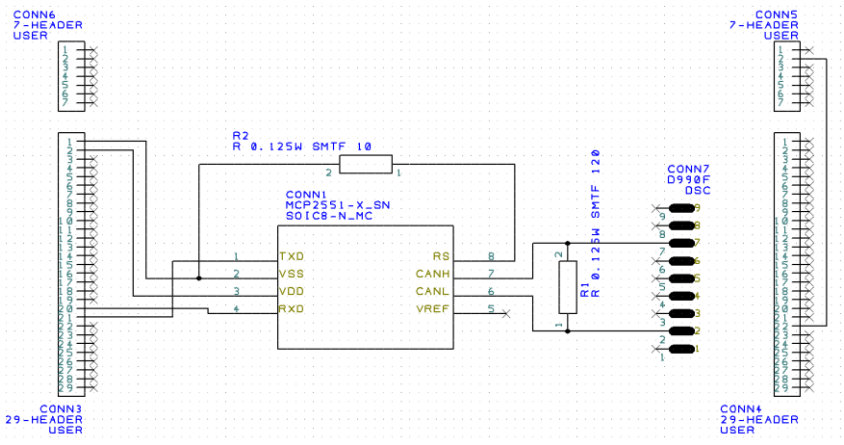


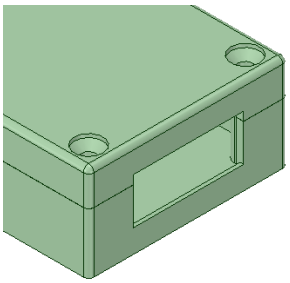
Schéma de câblage du circuit imprimé réalisé

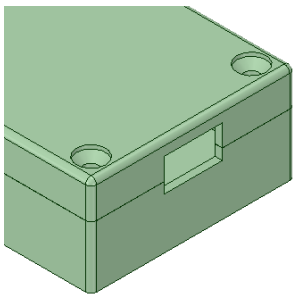
4. Boîtier

La carte PSoC et le circuit imprimé une fois assemblés, nous avons pris les dimensions de l'ensemble et nous avons conçu un boîtier afin de protéger l'ensemble et d'avoir un rendu de produit terminé.

La base du boîtier accueille l'électronique, et le couvercle vient se visser par-dessus.

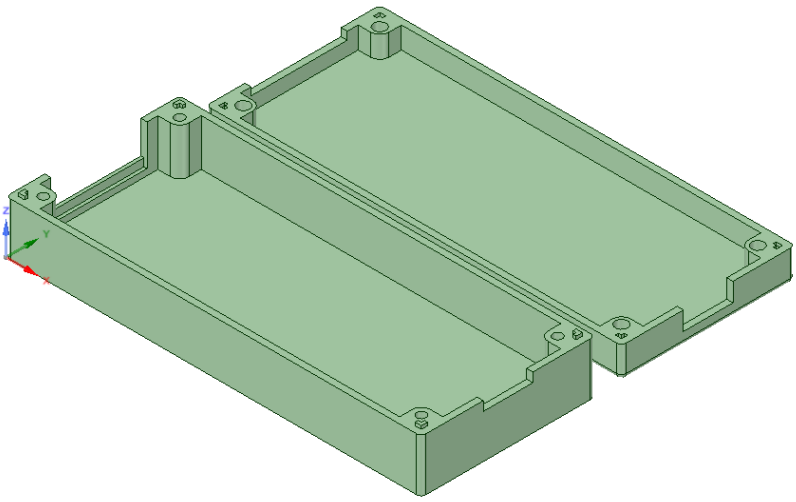
Il y a deux ouvertures pour les connectiques. D'un côté du boîtier un espace aux dimensions d'un connecteur DB9 et de l'autre côté un espace correspondant à une prise USB.



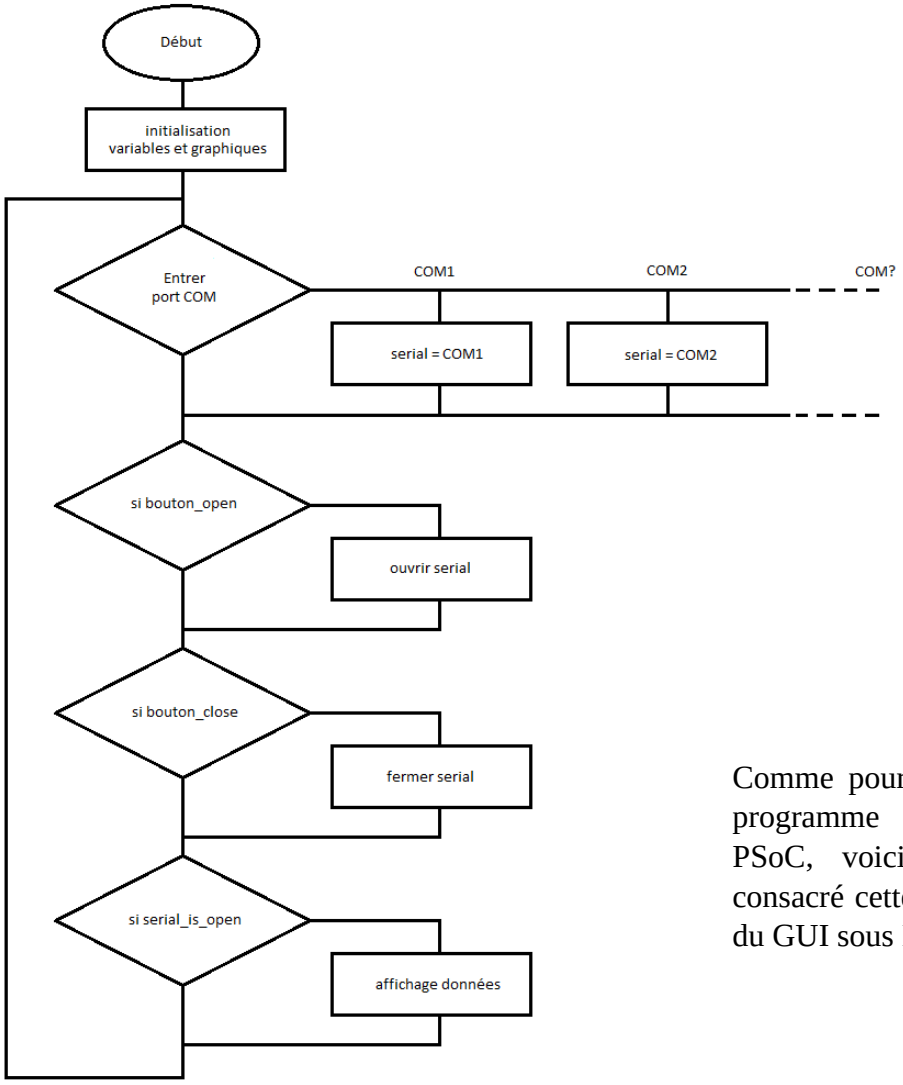


Le boîtier a été conçu avec le logiciel DesignSpark Mechanical. C'est un logiciel gratuit de CAO disponible sur le site de RadioSpares.

Le boîtier a ensuite été imprimé en 3D à l'IUT de Belfort.



5. GUI Matlab

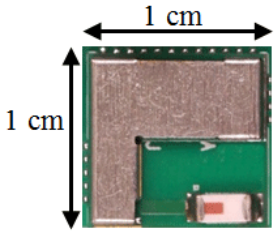


Comme pour la description du programme implanté dans le PSoC, voici un logigramme consacré cette fois-ci au codage du GUI sous MATLAB.

En premier lieu, il est demandé de rentrer le nom du port COM sur lequel on désire récupérer nos informations. Ensuite nous pouvons ouvrir le port en question.

S'affichent alors un bouton pour fermer le port COM ainsi qu'une zone de texte dynamique affichant la chaîne de caractère reçue sur le port.

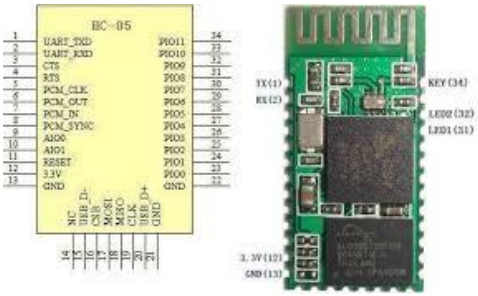
6. Transmission sans-fil



Pour la transmission sans-fil des informations du PSoC vers un ordinateur, le Bluetooth semble convenir à nos besoins. La carte CY8CKIT-043 dispose d'une empreinte pour accueillir un module Bluetooth à monter en surface directement sur la carte.

Cependant, en raison de la complexité de mise en œuvre de ce module (soudures + programmation) nous avons pensé à d'autres solutions.

Après avoir fait quelques recherches sur le Web sur des modules Bluetooth existants, nous trouvons les modules HC-0X qui semblent être assez populaires. Il en existe plusieurs modèles, de HC-03 à HC-09, ayant de légères variantes mais dont le fonctionnement ne change pas dans son ensemble.



En allant voir ce qu'il existait chez Microchip, nous avons trouvé le module RN-42XV.

Les modules HC-0X et RN-42XV utilisent le protocole SPP (Serial Port Profile) qui permet à un appareil connecté en Bluetooth au module de le considérer comme un port série classique (COM X).

III. Conclusion

Malgré les difficultés à mettre en œuvre la liaison CAN aux débuts du projet et la perte conséquente de temps liée à cela, nous avons tout de même pu avancer encore suffisamment pour atteindre les objectifs fixés au départ. Seule la transmissions sans-fil n'est pas présente, mais des solutions ont été évoqué.

D'un point de vue pédagogique, ce projet nous a permis de découvrir les PSoC et leur programmation, ainsi que l'exploitation du logiciel Matlab.