**Ecole supérieure**

POBJ

SLO2

2202 Interface carte ATM pour exercices PROG

**Réalisé par :**

Subahskanth Mohanasarma

**A l’attention de :**

Professeur M.Bovet

**Dates :**

Début du rapport : 12 mai 2022

Fin du rapport : 06 juin 2022

# Table des matières

[1. Table des matières 2](#_Toc105487028)

[2. Cahier de charge 3](#_Toc105487029)

[3. Planning - Suivi de projet 4](#_Toc105487030)

[4. Choix de la trame 4](#_Toc105487031)

[4.1. Header 4](#_Toc105487032)

[4.2. Cmdld1 & Cmdld0 4](#_Toc105487033)

[4.3. DataLen 4](#_Toc105487034)

[4.4. Data 4](#_Toc105487035)

[4.5. CRC 4](#_Toc105487036)

[4.6. Software côté Kit ARM 5](#_Toc105487037)

[4.7. Floatchart 5](#_Toc105487038)

[4.8. Explication du Floatchart : 5](#_Toc105487039)

[4.9. Etat de partie Software côté Kit ARM 6](#_Toc105487040)

[5. Partie Visual Studio application console 7](#_Toc105487041)

[5.1. Communication via port serial 7](#_Toc105487042)

[6. Etat Général du projet 8](#_Toc105487043)

[7. Conclusion 8](#_Toc105487044)

# Cahier de charge

**Description du projet**

L’ETML-ES possède un kit de programmation avec un microcontrôleur de type ARM. Ce kit est utilisé en première année de génie électrique dans le cour d’ELNU pour sensibiliser les premières années à programmer sur un microcontrôleur.

Le but du projet est de crée une liaison entre le cours d’ELNU et POBJ. L’idée est de crée une communication entre une application console réaliser sur Visual Studio avec le kit ARM.

De cette faite, nous pourrions via une application console dialoguer avec tous les périphériques onboard du kit.

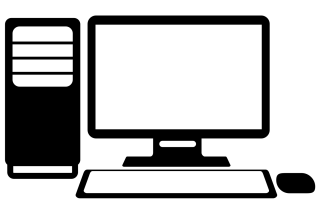
Les périphérique onboard sont les suivantes :

* Bouton
* Leds
* LCD
* Ain
* Aout
* RTCC MCP79411
* Capteur.Temp. LM70
* Carte extension pilotage moteur

**Choix de la communication :**

La communication a utilisé pour la liaison PC -> Kit, c’est la communication serial (UART) via USB.

Le traitement de donné côté PC doit être fait sur Visual Studio dans une application Console en C.



Kit ARM

Liaison USB

(Com.UART)

Application

Console en C

# Planning - Suivi de projet

1. Choix de la trame pour la communication entraine le Kit ARM et le PC.
2. Faire dans un premier temps l’interface sur le Kit ARM pour qu’il reçois et traite la trame.
3. Trouver un moyen de traiter des donner UART via une application console sur PC.
4. Crée une bibliothèque qui pourras être utiliser sur une application console pour communiquer via Serial USB sur le Kit USB.
5. Rédaction d’un rapport.

# Choix de la trame

Voici la trame que j’ai choisie pour ce projet.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Header  Byte0 | Cmdld1  byte1 | Cmdld0  byte2 | DataLen  byte3 | Data  byte 4…*n* | CRC  byte(*n+1*) |
|  |

## Header

Le Header byte sera tout le temps 0xFF pour la communication entre le kit et le PC. Cette byte représente comme une signature qui représente que on communique bien entre le kit et le PC et pas quelque chose d’autre.

## Cmdld1 & Cmdld0

C’est deux bytes sont destinés pour le choix du périphérique, que nous voulons atteindre sur le Kit ARM.

## DataLen

Cette byte représente la taille du Data.

## Data

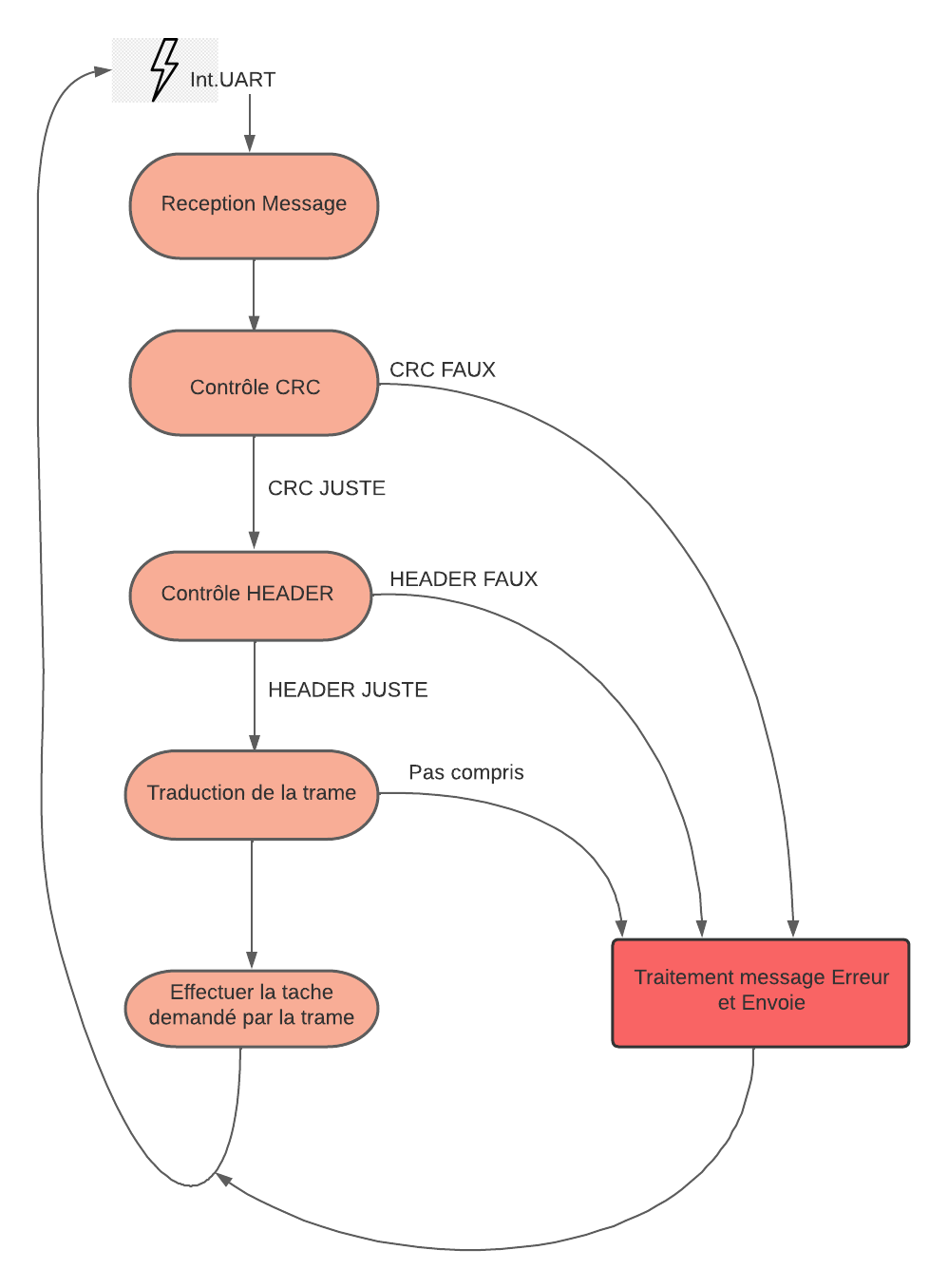
De l’octet 4 à n, ce sont les données.

## CRC

La dernier bytes (n+1) représente le CRC. C’est l’addition de tous les bytes sans compter le Header.

## Software côté Kit ARM

## Floatchart



## Explication du Floatchart :

Ci-dessus, vous pouvez voir le floatchart du programme sur le kit qui est facile à réaliser.

Dans un premier temps, une trame sera envoyé via le PC depuis une application console qui sera reçu par une interruption UART.

Après la réception, nous effectuons en checksum pour vérifier le CRC qui représente la dernière byte de notre message.

Si le CRC est juste, nous effectuons la tache suivante et si elle faute, je réalise une trame avec un code erreur qui sera renvoyer sur la console.

La tache suivante consiste à vérifier si le message est adressé pour le kit en vérifiant le HEADER qui représente la première byte de notre message. Si ce n’est pas le cas, un autre message sera d’erreur sera transmis.

Si le message est bien adressé au kit alors, on devra à ce moment la décodé la trame et vérifier si ce message est bien traitable sur le kit. Sinon un autre message d’erreur sera envoyé.

Si le message est traitable alors, nous effectuerons la tache demandée avant de réactiver l’interruption de l’UART.

Voici les erreurs que nous pouvons avoir en soit :

#define NOT\_SUPPORTED 0xA0 // Message non adressé pour le kit

#define ERROR\_CRC 0xE0 // Erreur de CRC

#define NOT\_UNDERSTOOD 0xC0 // Le kit comprend pas la trame.

Après l’envoie du message d’erreur, on n’oublie pas non plus d’activé l’interruption de l’UART pour recevoir un autre message.

## Etat de partie Software côté Kit ARM

Pour le moment, j’ai réalisé les taches suivantes :

* Réception du message complet-
* Contrôle du CRC
* Contrôle du Header
* Traitement des messages d’erreur et l’envoie
* Traduction de la trame LCD, LED\_ON et LED\_OFF
* Effectuer la tache demander pour le LCD, LED\_ON et LED\_OFF

Le listign du code côté kit ARM sera en Annexe du rapport.

# Partie Visual Studio application console

## Communication via port serial

Pour cette partie, j’ai dû trouver un moyen de communiquer en via port serial avec le kit ARM.

Pour ce faire, j’ai effectué divers recherche sur internet avant de tomber sur ce site : <https://github.com/rrmhearts/Serial-Programming-Win32API-C>

Dans ce site, il explique comment ouvrir une communication serial avec une application console alors j’ai suivi et pris exemple du code pour réaliser la partie suivante :

- Ouverture du port serial

- Paramétrer le port serial

- l’envoie d’un message.

Je n’ai encore pas eu le temps de faire la réception et il y a une petite erreur dans les paramétrages du port serial alors je dois passer par putty pour le configurer et fermer putty avant d’exécuter le code de l’application console.

Après avoir réalisé cette étape, j’ai juste à crée les trames. Pour le moment, j’ai réalisé les trames suivantes :

- Eteindre une LED

- Allumé une LED

- Ecrire sur le LCD

Les fonction cité ci-dessus fonctionne correctement sur le kit ARM et son fonctionnel.

Alors j’ai réalisé une bibliothèque (.h et .c) avec les 3 fonction.

Le listign du code côté kit ARM sera en Annexe du rapport.

# Etat Général du projet

Pour le moment, nous pouvons allumer et éteindre des LEDs et écrire sur le LCD du kit via une application console.

# Conclusion

Ce projet m’a permis d’apprendre et d’approfondir mes connaissances en C. J’ai pu apprendre à ouvrir et utiliser une communication serial via une application console sur visual Studio.

Effectuer divers recherche sur internet pour la réalisation d’ouvrir un port serial sur une application console et aboutir était un exploit pour moi car je ne pensais pas qu’il était possible de le faire.

J’ai pu mettre en pratique mes connaissance pour la réalisation d’une trame pour communiquer entre deux points et de pouvoir le mettre en œuvre. Cette partie fonctionne correctement dans mon système.

Je n’ai pas eu le temps pour réaliser le projet complétement par manque de temps. Mais je peux continuer sur le projet après le rendu car il est très intéressant.

La base du projet est déjà faite par moi, et il est fonctionnel. Il reste juste a faire le traitement des donnés pour rajouter les divers autre périphérique.

Le 06 juin 2022 Mohanasarma Subahskanth