

Rapport projet

**Ecole supérieure
Électronique**

**Système de ticketing pour questions
d'étudiants**

Projet n°1811C

Réalisé par :

Dos Santos Mario

Date :

Début du diplôme : 22 Août 2022
Fin du diplôme : 27 Septembre 2022

Table des matières :

Système de ticketing pour questions d'étudiants	1
Projet n°1811C	1
1 Introduction	5
1.1 Cahier des charges	5
1.2 Analyse de l'existant.....	5
1.3 Amélioration	5
2 Design	6
2.1 Boitier	6
2.2 Carte Master.....	6
2.2.1 Schéma bloc	6
2.2.2 Bouton Reset	7
2.2.3 Alimentation carte SD	7
2.2.4 Pull Up	7
2.2.5 Régulateur 3V3	8
2.2.6 Xbee.....	8
2.2.7 Communication USB.....	8
2.2.8 Ecran TFT	9
2.3 Carte Slaves.....	9
2.3.1 Schéma bloc	9
2.3.2 Erreur de résistance.....	10
2.4 Choix boitier	10
2.4.1 Boitier Master	11
2.4.2 Boitier Slaves	11
3 Routage.....	12
3.1 Taille PCB	13
3.1.1 PCB Master	13
3.1.2 PCB Slave.....	13
3.2 Plan de masse.....	13
3.3 Stitching	14
3.4 Vue 3D	14
3.4.1 PCB Master	14
3.4.2 PCB Slave	15
4 Mise en service	16
4.1 Mesure alimentation	16
4.2 Correction Hadware	16
4.3 Vérification de la version précédente	16
5 Firmware/Software	16
5.1 Master	16
5.1.1 Configuration.....	16
5.1.1.1 System Clock.....	16
5.1.1.2 UART.....	17

5.1.1.3 SPI.....	17
5.1.1.4 Carte SD.....	18
5.1.1.5 Timer	18
5.1.1.5.1 Timer 1.....	18
5.1.1.5.2 Timer 2.....	19
5.1.2 Structogramme/diagramme.....	20
5.1.2.1 Pulling.....	20
5.2 Slave	21
5.2.1 Configuration.....	21
5.2.1.1 System Clock.....	21
5.2.1.2 UART	21
5.2.1.3 Timer	22
5.2.1.3.1 Timer 1.....	22
5.2.1.3.2 Timer 2.....	23
5.2.2 Structogramme/diagramme.....	24
5.3 Application C#	25
5.3.1 Interface	25
5.3.2 Etat.....	26
5.4 Trames	27
5.4.1 Commande.....	27
5.4.2 Broadcast	27
5.4.3 Donnée	27
6 Test et mesure	28
6.1 Ecran TFT	28
6.2 Driver TFT	29
6.3 Xbee	29
6.3.1 Code	29
6.3.2 Temps de communication	30
7 Planification.....	31
8 Etat d'avancement.....	31
9 Amélioration	32
10 Conclusion	32
11 Annexe	33
11.1 Cahier des charges	33
11.2 Planification	33
11.3 Procès-Verbaux.....	33
11.4 Journal de travail	33
11.5 Schéma électrique complet	33
11.6 Liste de pièce	33
11.7 Listing Code	33
11.8 Mode d'emploi	33

1 Introduction

Je dois reprendre un projet déjà existant et le corriger/modifier selon le cahier des charges

1.1 Cahier des charges

Voir Cahier des charges en annexe

1.2 Analyse de l'existant

J'ai essayé de faire fonctionner le projet précédent mais sans succès

Mais il est dans un état avancer. De ce j'ai pu comprendre le projet fonctionnait lorsque le module RF (projet 1623) n'était pas implémenter sur les cartes.

D'après le rapport il n'y a pas d'erreur conséquent mais lorsque j'ai analyser les cartes j'ai pu constater que l'antenne du module RF de la carte master se trouvait en dessous d'un plan de masse cela fera partie des modifications que je devrais faire.

1.3 Amélioration

Mon but sera :

- De corriger les erreurs hardware.
- D'améliorer le software.
- De rajouter une communication USB qui permettra de communiquer avec les ordinateurs.
- De rajouter le software du projet 2126 qui permet de directement aller chercher le nom de la personne connecter à l'ordinateur pour ne plus passer par une carte SB et un carte Slave personnel à chaque élève.
- De créer un programme Windows avec les mêmes fonctionnalités que le LCD.
- De faire une mise en boitier qui ne nécessitera pas l'utilisation d'une imprimante 3D.

2 Design

2.1 Boitier

Dans le cahier des charges il est indiqué que je dois changer le boitier pour un modèle commercial, celui précédemment utilisé avait été créée par un étudiant à l'aide d'une imprimante 3D.

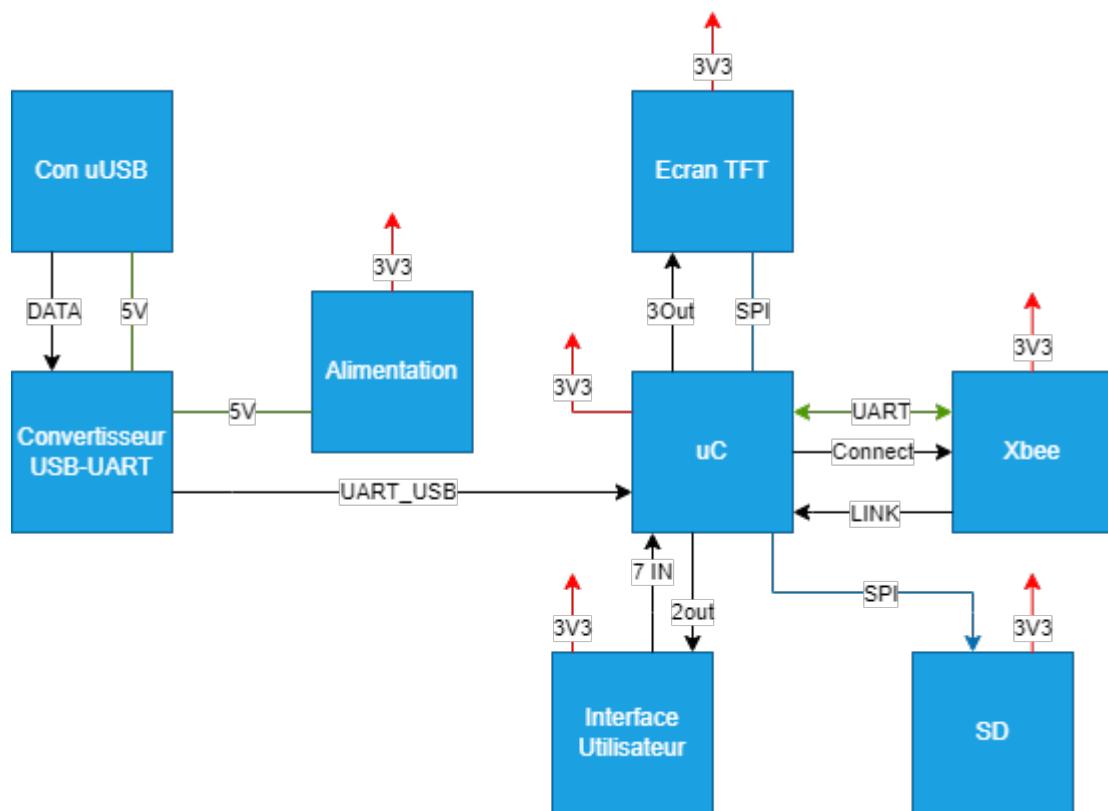
En plus de cela j'ai constaté que l'antenne du Xbee était près des plans de masse ce qui est déconseiller, comme je n'aurais pas la place de le déplacer avec la taille des boitiers existant je devrais prendre un boitier plus grand ce qui me permettrait lorsque j'aurais besoin de changer de composant de ne pas me soucier de leurs tailles.

2.2 Carte Master

La première étape a été de corriger les erreurs de mon prédecesseur :

- Faire fonctionner le bouton reset
- Rajout de l'alimentation pour la carte SD
- Rajout de la pull up du MOSI de la carte SD

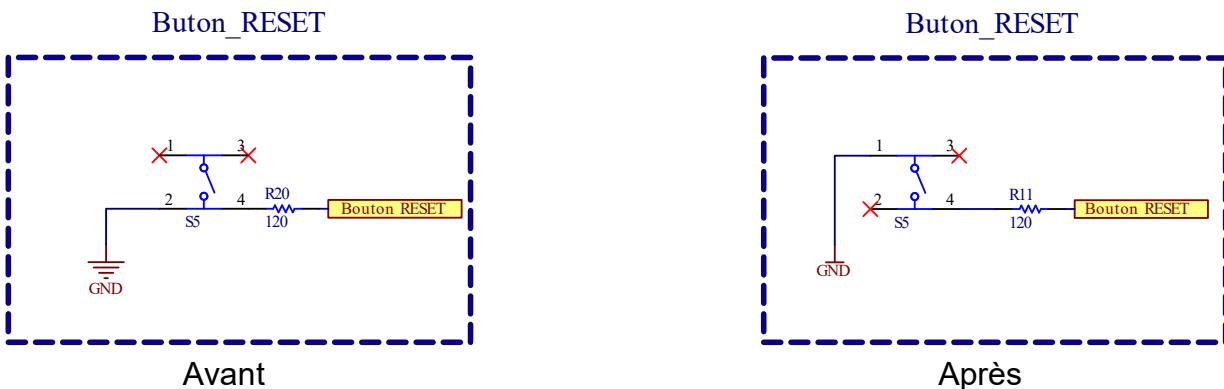
2.2.1 Schéma bloc



2.2.2 Bouton Reset

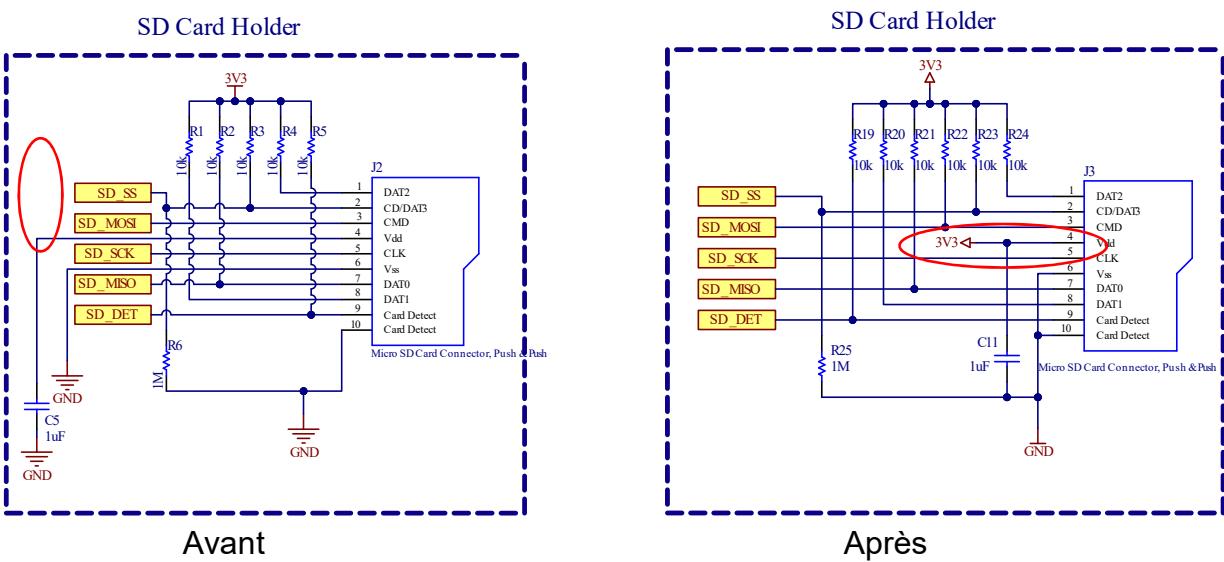
Le reset était toujours connecté au GND sachant que le reset est actif à l'état bas. La conséquence de cette erreur était que la carte ne pouvait pas fonctionner car tous les composants connectés au reset ne pouvaient pas fonctionner.

Pour cette erreur il suffit de changer l'une des 2 pistes et la brancher sur la patte 1 ou 3. J'ai décidé de déplacer la connexion du GND car lors du routage ça ne fera pas de grande différence grâce au plan de masse.



2.2.3 Alimentation carte SD

Il y a eu un oubli de l'alimentation(VDD/3V3) de la carte SD
J'ai simplement mis le symbole du 3V3 et réarranger les pistes.



2.2.4 Pull Up

Lorsque j'ai regardé la carte j'ai constaté qu'il y a eu un rajout d'une résistance. Il était directement branché sur le 3V3 et un via du MOSI.

De ce que j'ai pu comprendre on utilise des pullups seulement si les pattes du master sont en open-collecteur.

Après discussion avec M.moreno on a décidé de laisser les footprint des résistances et dans le cas où on en aurait besoin on les braserai.

2.2.5 Régulateur 3V3

J'ai dû changer le régulateur 3V3 car celui qui y était implémenter était en rupture de stock dans les différents fournisseurs

Dans ce cas j'ai préféré prendre le régulateur MAX1793 qui est fréquemment utilisé à l'etml-es et qui en plus possède du stock (à l'ETML-ES et chez les fournisseurs). Le composant est plus imposant que l'ancien utilisé mais comme je n'ai plus à me soucier de la place je peux me permettre de le prendre

En plus de cela il peut transmettre plus de courant ,1A au lieu de 250mA.

2.2.6 Xbee

Lors de la version A il a été décider d'utiliser le Xbee créer par l'école (projet n° 1623) mais le projet n'a jamais fonctionné avec ce Xbee.

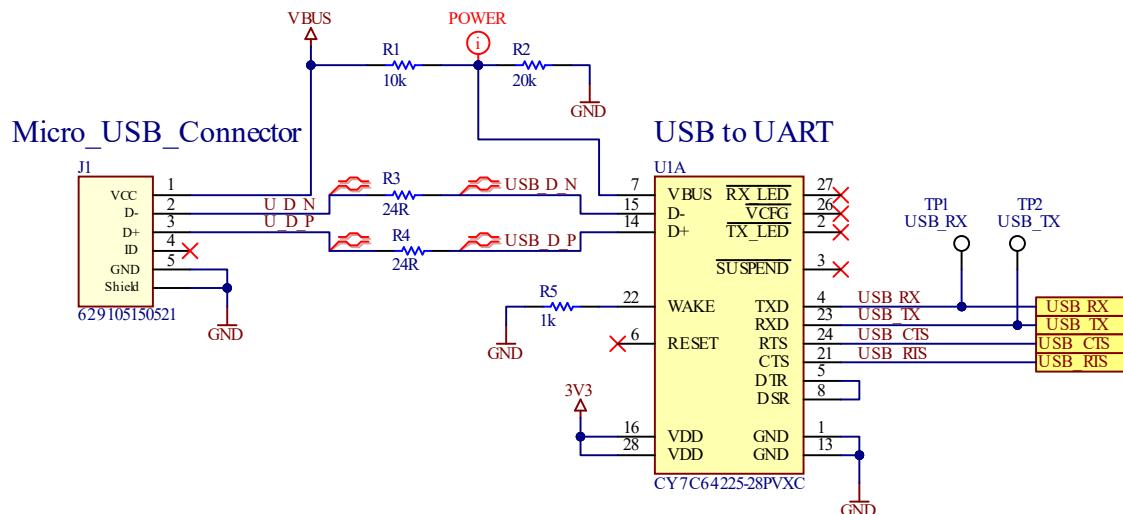
Il a donc été décidé que l'on allait continuer de l'utilisé en espérant que le problème ne me fasse pas perdre trop de temps et si cela ne fonctionne toujours pas nous utiliseront un autre Xbee (XB24CAUIT-001) qui a déjà servi dans plusieurs projets.

Heureusement les pins et la distance entre les pattes sont identique pour les 2 Xbee (alimentation + communication uart).

2.2.7 Communication USB

Pour cette version nous allons ajouter une communication USB qui nous permettra de récupérer le nom de l'élève sans l'utilisation de la carte SD et des adresses des Xbee.

Je vais utiliser le contrôleur de pont USB-UART du projet 2126 (CY7C64225) car il y en a en stock et qu'il a déjà été utiliser.



Dans le datasheet il est précisé que si le microcontrôleur et le contrôleur USB-UART sont alimenter en 3V3 il faut mettre un pont diviseur entre le VBUS du connecteur USB et la patte VBUS de l'IC.

If both CY7C64225 and the microcontroller (MCU) are operating at 3.3 V, connect a divider circuit to provide 3.3 V to VBUS pin of CY7C64225 from VBUS pin of USB port.

$$U_{pont} = \frac{R2}{R2 + R1} * V_{BUS} = \frac{20k}{20k + 10k} * 5 = 3,33V$$

2.2.8 Ecran TFT

Je n'arrive pas à trouver le modèle de l'écran.

J'ai dû chercher dans les écrans d'adafruit les modèles les plus proche de celui sur la carte. Les seules informations que je possède sont :

- Le nombre de pins : 40
- La disposition de ces pins
- La taille de l'écran 2.8"
- Le driver : ILI9341

J'ai pris le modèle 2090 car il était plus récent que le 1770 et que la seule différence que j'ai pu observer était que le premier avait un touchscreen capacitif et le second un résistif.

Comme la seule différence se trouvait dans le touchscreen, que je n'utilise pas,,je pense qu'il n'y aura pas de problème lors de la programmation.

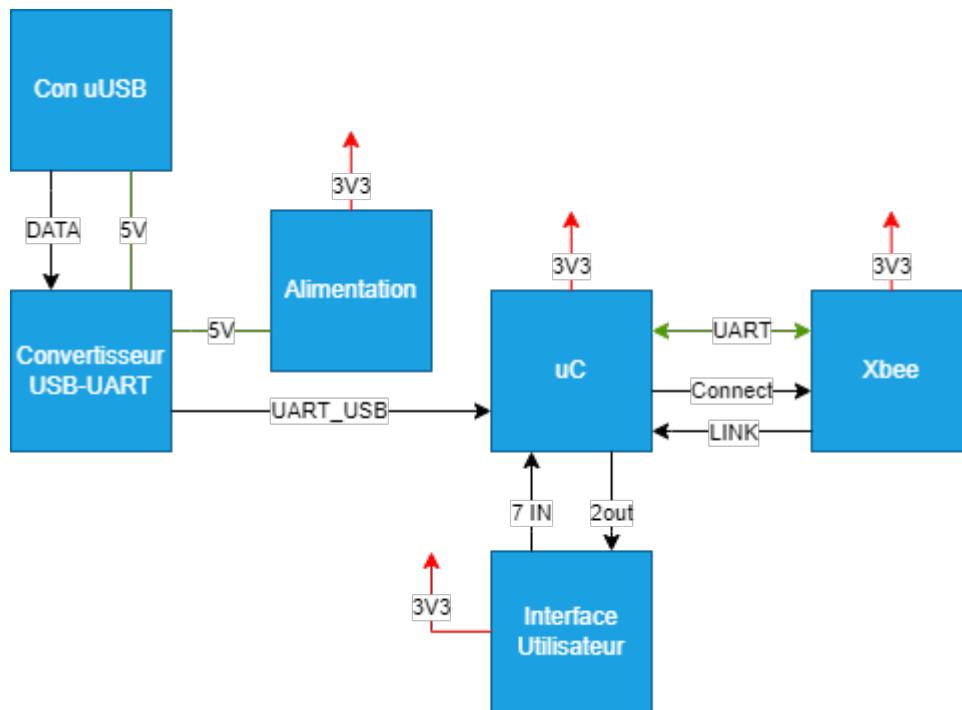
2.3 Carte Slaves

Nous refaisons les mêmes corrections et ajoutons que pour la carte Master sauf pour la partie SD (correction alim et ajout pull up).

Donc :

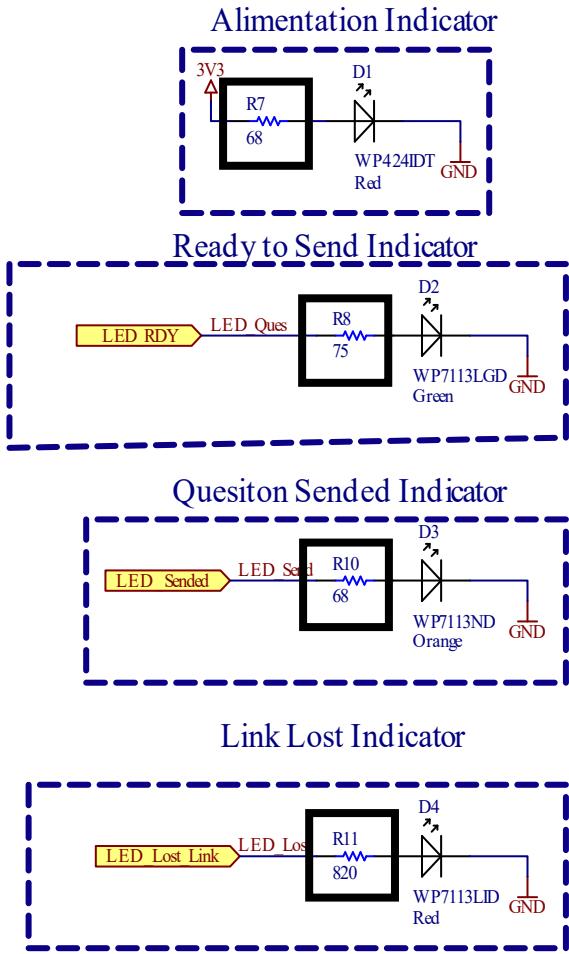
- Correction du bouton reset
- Changement du régulateur 3V3
- Ajout de la communication USB

2.3.1 Schéma bloc



2.3.2 Erreur de résistance

Lors de la review de schéma, M.moreno à remarquer qu'une des résistance utiliser pour les leds étaient 10 fois supérieur aux autres.



Comme j'ai repris un schéma déjà existant et qu'il n'y avait pas d'erreur de signaler sur les leds, je n'y avait pas fait intention.

J'ai pu aussi remarquer que les résistances permettent de soutirer jusqu'à 20mA alors que les I/O du microcontrôleur ne permettent que 15mA

Donc j'ai décidé de tout recalculer.

$$R_{alim} = \frac{VCC - V_{led_red}}{I_{led_red}} = \frac{3.3 - 2}{20m} = 65 \rightarrow 68 E24$$

$$R_{Ques} = \frac{VCC - V_{led_green}}{I_{led_green}} = \frac{3.3 - 1.9}{15m} = 93.3 \rightarrow 100 E24$$

Dans les sites de fournisseur (Digikey et Mouser) il était indiqué que le courant forward était de 2mA alors que dans le datasheet il est indiqué à 25mA.

J'ai donc décidé de suivre les indiquation du datasheet mais avec le courant max des I/O

$$R_{Send} = \frac{VCC - V_{led_orange}}{I_{led_orange}} = \frac{3.3 - 2.05}{15m} = 83.3 \rightarrow 91 E24$$

$$R_{lost_link} = \frac{VCC - V_{led_red}}{I_{led_red}} = \frac{3.3 - 1.7}{15m} = 106.6 \rightarrow 110 E24$$

2.4 Choix boitier

J'ai eu pas mal de peine à trouver un boitier pour la carte commande car je voulais garder un boitier semblable à celui qui avait été fait par le précédent diplômant mais qui est un peu plus large car je dois mettre le Xbee dans un bord sans rien autour (plan de masse, composant, piste).

J'ai été voir chez Hammond manufacturing car ils ont un grand nombre de boitier et qu'ils ont un système de filtrage qui est facilement utilisable.

2.4.1 Boitier Master

Le boitier de la version A faisait ~131 mm x 91 mm x 23 mm

Pour mes recherches j'ai cherché les boîtiers de taille 135 mm x 110 mm x 20 mm (Longueur X Largeur X Hauteur) et qui étaient en plastique.

J'ai paramétré la hauteur à 20 mm car je pensais trouver des boîtiers avec un minimum de 15mm de hauteur en interne du boitier et j'espérais trouver des boîtiers assez proches de cette taille. Mais sur Hammond ils affichent les boîtiers plus grande que ce que le filtre donc sa éliminait quand même les boîtiers trop petits en longueur x largeur.

Si la taille est plus grande je devrais trouver un moyen avec des colonnettes pour avoir un résultat similaire à ce qui a été fait.



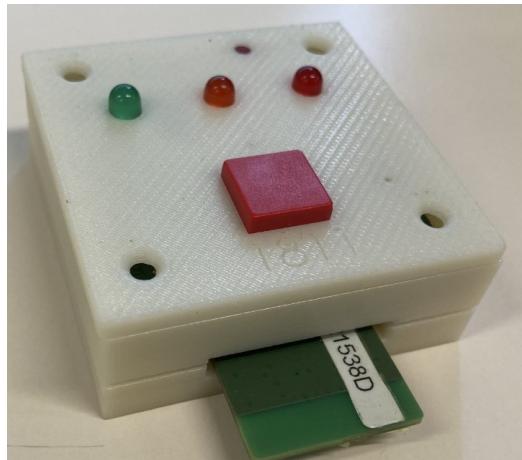
Boitier Master version A

Je vais utiliser le boitier 1555HL2GY qui fait 180 mm x 119 mm x 46 mm. C'est le plus petit boitier en hauteur avec au moins la largeur et longueur égal ou plus grands de ce que j'avais prévu

2.4.2 Boitier Slaves

Le boitier des cartes slaves fut beaucoup plus simple à trouver.

Je cherchais un boitier de ~70 mm x 55 mm x 20mm et j'ai pu trouver rapidement un boitier de 84 mm x 56 mm x 23 mm.



Boitier Slave version A

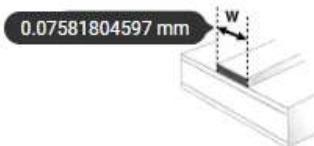
3 Routage

Pour le routage j'utilise la version 21.6.1 d'Altium.

J'ai calculer la largeur des pistes minimum pour être sur qu'il n'y ait pas de problème
Pour ça j'ai utilisé le calculateur de Digikey :

Courant (I)	A	Ambient Temperature	°C
0.5		25	
Épaisseur (t)	μm	Longueur de piste	po
35		Entrez la longueur...	
Augmentation de température (T_{Rise})	°C		
20			

Largeur de piste minimum



Couches externes à l'air libre

Largeur de piste requise (W)

0,07581804597	mm
---------------	----

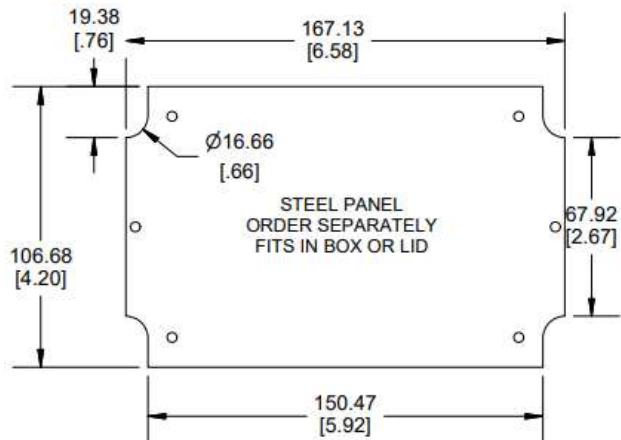
J'ai décidé de faire mon routage avec des pistes de 0.254mm, hors alimentation, et des pistes de 0.5mm pour les piste d'alimentation.

Les contraintes de routage ont été réglée pour correspondre à un PCB de classe 6C sur Eurocircuit.

3.1 Taille PCB

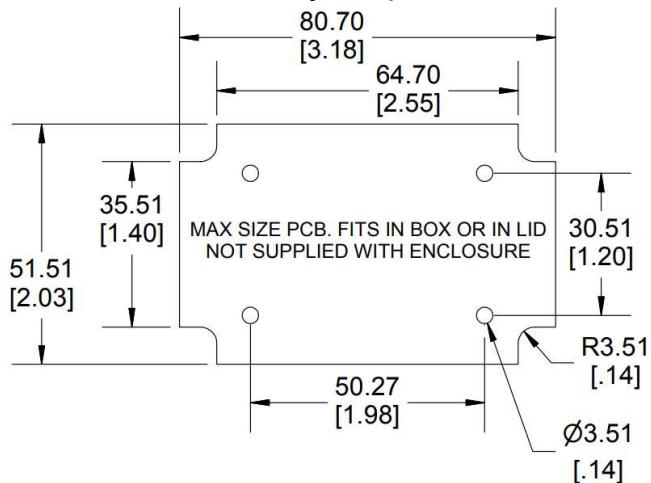
3.1.1 PCB Master

Pour la taille du PCB Master j'ai repris les dimensions données pour une plaque en acier. Il est surtout important de bien placer les trous pour les vis.



3.1.2 PCB Slave

Pour la taille du PCB j'ai repris les dimensions données dans le datasheets du boitier



3.2 Plan de masse

Je mets un plan de masse au top et au bottom pour permettre une meilleure distribution du GND et réduire les perturbations (bruits et interférence).

Dans mon cas je dois faire attention à ne pas mettre de plan de masse sous le module Xbee car il possède une antenne, le plan de masse le perturberait si on le met en dessous.

3.3 Stitching

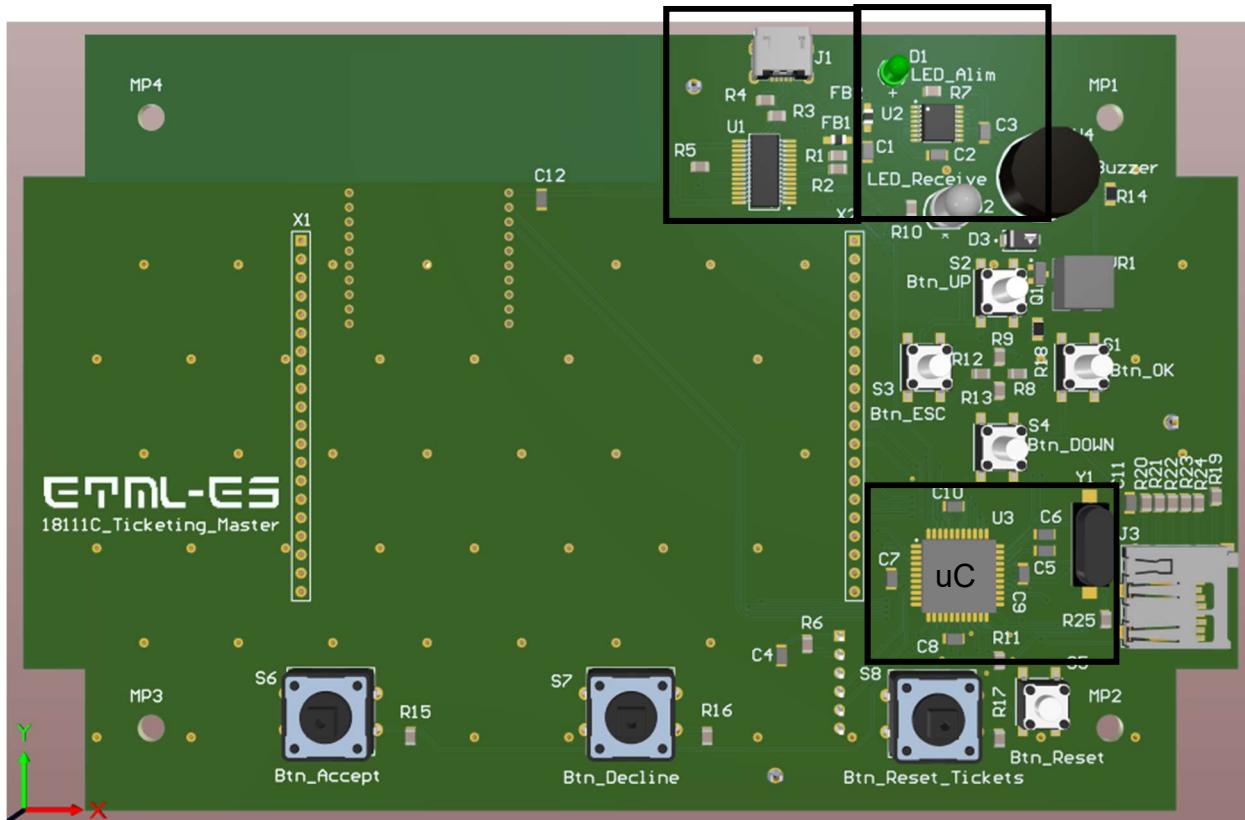
Le stitching permet d'avoir un plan de masse plus homogène sur tout le PCB

3.4 Vue 3D

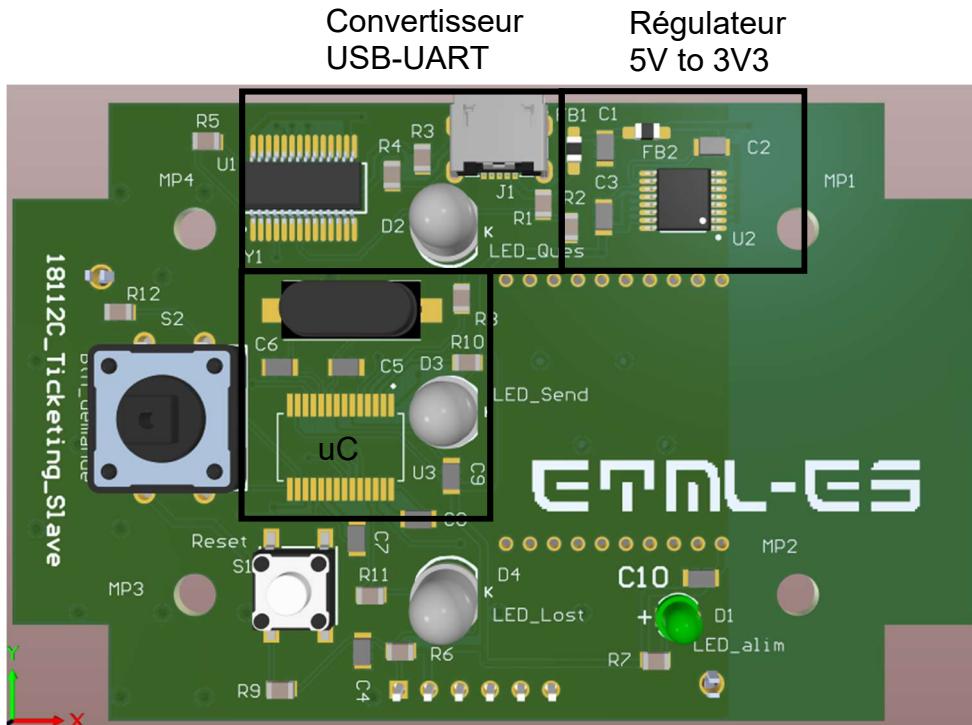
3.4.1 PCB Master

Convertisseur
USB-UART

Régulateur
5V to 3V3



3.4.2 PCB Slave



4 Mise en service

4.1 Mesure alimentation

J'ai mesuré les alimentations 3,3V des 3 cartes (1 Master et 2 slave).
Je me retrouve avec des alimentations d'environ 3,36V stable.

4.2 Correction Hadware

Je n'ai pas eu besoin de faire de correction hardware.

4.3 Vérification de la version précédente

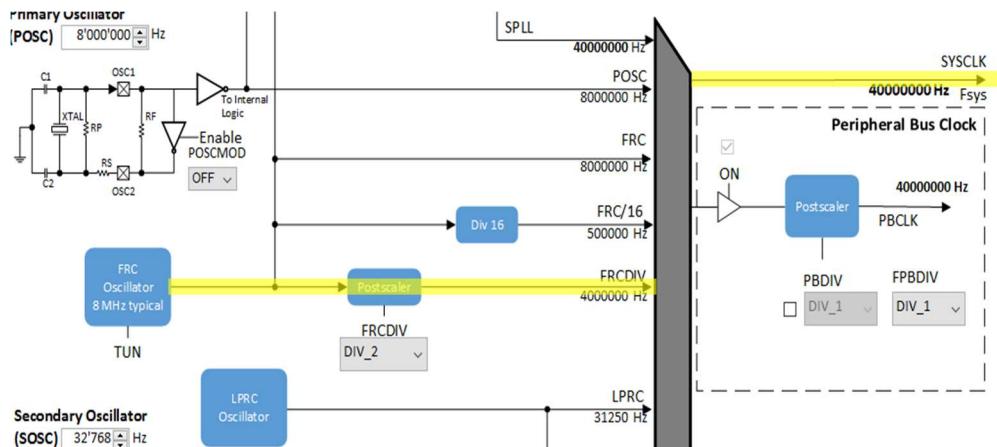
En programmant ma carte avec l'ancien programme j'ai pu observer que je n'arrivais juste à allumer le backlight du LCD de la carte Master.

5 Firmware/Software

5.1 Master

5.1.1 Configuration

5.1.1.1 System Clock



J'ai décidé de laissé le sysclk à 40MHz qui avait été utilisé dans la version A

5.1.1.2 UART

The screenshot shows two side-by-side configurations for the USART driver.

Xbee Configuration:

- USART Driver:** Enabled (checked). Driver Implementation: STATIC. Interrupt Mode: Enabled (checked).
- Byte Model Support:** Enabled (checked).
- Number of USART Driver Instances:** 2. A note states: "**** Each instance can have only one client in STATIC driver mode ****".
- Number of USART Driver Clients:** 2.
- USART Driver Instance 0:**
 - USART Module ID:** USART_ID_1. Baud Rate: 115200.
 - USART Interrupt Priority:** INT_PRIORITY_LEVEL5.
 - USART Interrupt Sub-priority:** INT_SUBPRIORITY_LEVEL0.
 - Operation Mode:** DRV_USART_OPERATION_MODE_NORMAL.
 - Wake On Start:** Unchecked.
 - Auto Baud:** Unchecked.
 - Stop In Idle:** Unchecked.
 - Line Control:** DRV_USART_LINE_CONTROL_8NONE1.
 - Handshake Mode:** DRV_USART_HANDSHAKE_NONE.
- USART Driver Instance 1:** Similar to Instance 0 but with USART Module ID USART_ID_2, Baud Rate 57600, and USART Interrupt Priority INT_PRIORITY_LEVEL1.
- USART Driver:** Enabled (checked).

USB Configuration:

- USART Driver:** Enabled (checked). Driver Implementation: STATIC. Interrupt Mode: Enabled (checked).
- Byte Model Support:** Enabled (checked).
- Number of USART Driver Instances:** 2. A note states: "**** Each instance can have only one client in STATIC driver mode ****".
- Number of USART Driver Clients:** 2.
- USART Driver Instance 0:**
 - USART Module ID:** USART_ID_0. Baud Rate: 115200.
 - USART Interrupt Priority:** INT_PRIORITY_LEVEL1.
 - USART Interrupt Sub-priority:** INT_SUBPRIORITY_LEVEL0.
 - Operation Mode:** DRV_USART_OPERATION_MODE_NORMAL.
 - Wake On Start:** Unchecked.
 - Auto Baud:** Unchecked.
 - Stop In Idle:** Unchecked.
 - Line Control:** DRV_USART_LINE_CONTROL_8NONE1.
 - Handshake Mode:** DRV_USART_HANDSHAKE_NONE.
- USART Driver Instance 1:** Similar to Instance 0 but with USART Module ID USART_ID_1, Baud Rate 57600, and USART Interrupt Priority INT_PRIORITY_LEVEL1.
- USART Driver:** Enabled (checked).

Pour le Xbee j'ai repris les paramètre utilisé dans le projet 1623 (20200214_Module_Xbee_06_SCA) qui a été modifier par M.Castoldi.

5.1.1.3 SPI

The screenshot shows two side-by-side configurations for the SPI driver.

TFT Screen Configuration:

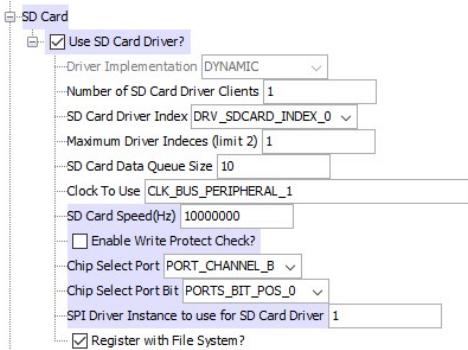
- SPI Driver:** Enabled (checked). Options include: Use Blocking Mode?, Use Interrupt Mode?, Use Polled Mode?, Use Master Mode? (checked), Use Slave Mode?, Use Standard Buffer Mode?, Use Enhanced Buffer (FIFO) Mode? (checked), Use 8-bit Mode? (checked), Use 16-bit Mode?, Use 32-bit Mode?, Use DMA?, Using Client Configure Function In Application? (checked).
- Number of SPI Driver Instances:** 2. A note states: "**** Each instance can have only one client in STATIC driver mode ****".
- Number of SPI Driver Clients:** 2.
- Number of job elements created per instance:** 10.
- SPI Driver Instance 0:**
 - SPI Module ID:** SPI_ID_1.
 - Driver Mode:** SPI Interrupt Priority: INT_PRIORITY_LEVEL1, SPI Interrupt Sub-priority: INT_SUBPRIORITY_LEVEL0.
 - Master\Slave Mode:**
 - Data Width:**
 - Buffer Mode:** Allow Idle Run unchecked.
 - Protocol Type:** DRV_SPI_PROTOCOL_TYPE_STANDARD.
 - Baud Clock Source:** SPI_BAUD_RATE_PBCLK_CLOCK.
 - Clock\Baud Rate - Hz:** 10000000.
 - Clock Mode:** DRV_SPI_CLOCK_MODE_IDLE_LOW_EDGE_FALL.
 - Input Phase:** SPI_INPUT_SAMPLING_PHASE_IN_MIDDLE.
 - Dummy Byte Value:** 0xFF.
 - Max Jobs In Queue:** 10.
 - Minimum Number Of Job Queue Reserved For Instance:** 1.
- SPI Driver Instance 1:** Similar to Instance 0 but with SPI Module ID SPI_ID_2, Driver Mode SPI Interrupt Priority INT_PRIORITY_LEVEL1, SPI Interrupt Sub-priority INT_SUBPRIORITY_LEVEL0, and Master\Slave Mode.
- SPI Driver:** Enabled (checked).

SD Card Configuration:

- SPI Driver:** Enabled (checked). Options include: Use Blocking Mode?, Use Interrupt Mode?, Use Polled Mode?, Use Master Mode?, Use Slave Mode?, Use Standard Buffer Mode?, Use Enhanced Buffer (FIFO) Mode? (checked), Use 8-bit Mode? (checked), Use 16-bit Mode?, Use 32-bit Mode?, Use DMA?, Using Client Configure Function In Application? (checked).
- Number of SPI Driver Instances:** 2. A note states: "**** Each instance can have only one client in STATIC driver mode ****".
- Number of SPI Driver Clients:** 2.
- Number of job elements created per instance:** 10.
- SPI Driver Instance 1:**
 - SPI Module ID:** SPI_ID_2.
 - Driver Mode:** SPI Interrupt Priority: INT_PRIORITY_LEVEL1, SPI Interrupt Sub-priority: INT_SUBPRIORITY_LEVEL0.
 - Master\Slave Mode:**
 - Data Width:**
 - Buffer Mode:** Allow Idle Run unchecked.
 - Protocol Type:** DRV_SPI_PROTOCOL_TYPE_STANDARD.
 - Baud Clock Source:** SPI_BAUD_RATE_PBCLK_CLOCK.
 - Clock\Baud Rate - Hz:** 10000000.
 - Clock Mode:** DRV_SPI_CLOCK_MODE_IDLE_LOW_EDGE_FALL.
 - Input Phase:** SPI_INPUT_SAMPLING_PHASE_IN_MIDDLE.
 - Dummy Byte Value:** 0xFF.
 - Max Jobs In Queue:** 10.
 - Minimum Number Of Job Queue Reserved For Instance:** 1.
- SPI Driver:** Enabled (checked).

J'ai laissé les paramètres déjà utilisé précédemment.
Dans mon projet l'écran et la carte SD ne sont pas essentiel mais je vais quand même essayer de les faire fonctionner.

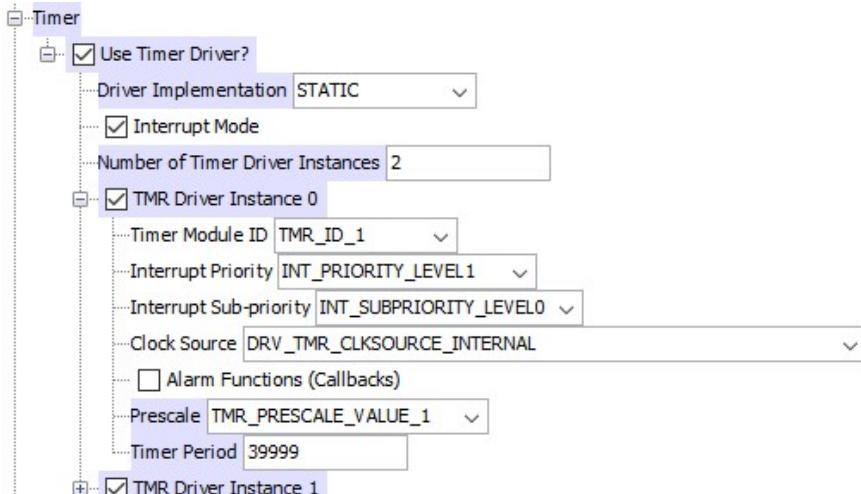
5.1.1.4 Carte SD



Les paramètres de la carte SD avaient déjà été implémentés.

5.1.1.5 Timer

5.1.1.5.1 Timer 1



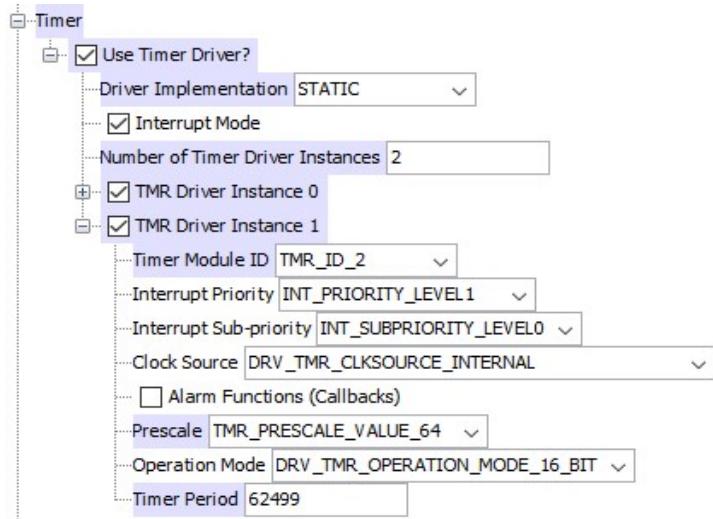
Ce timer est utilisé lors d'une réception de ticket. Il fera vibrer le buzzer à la fréquence donnée.

Le temps du timer est de :

$$\text{Temps} = \frac{(Période[tick] + 1) * prescaler}{SYSCLK} = \frac{(39999 + 1) * 1}{40 * 10^6} - 1 = 1 \text{ ms}$$

Ce timer était déjà implémenté dans la version A.

5.1.1.5.2 Timer 2



Ce timer sera utilisé pour envoyer un broadcast et le pulling toute les 2s

La période cible sera de 100ms. Je ne mets pas le timer à 2s pour ne pas utiliser un timer 32bits.

Le clock du microcontrôleur(SYSCLK) est de 40Mhz

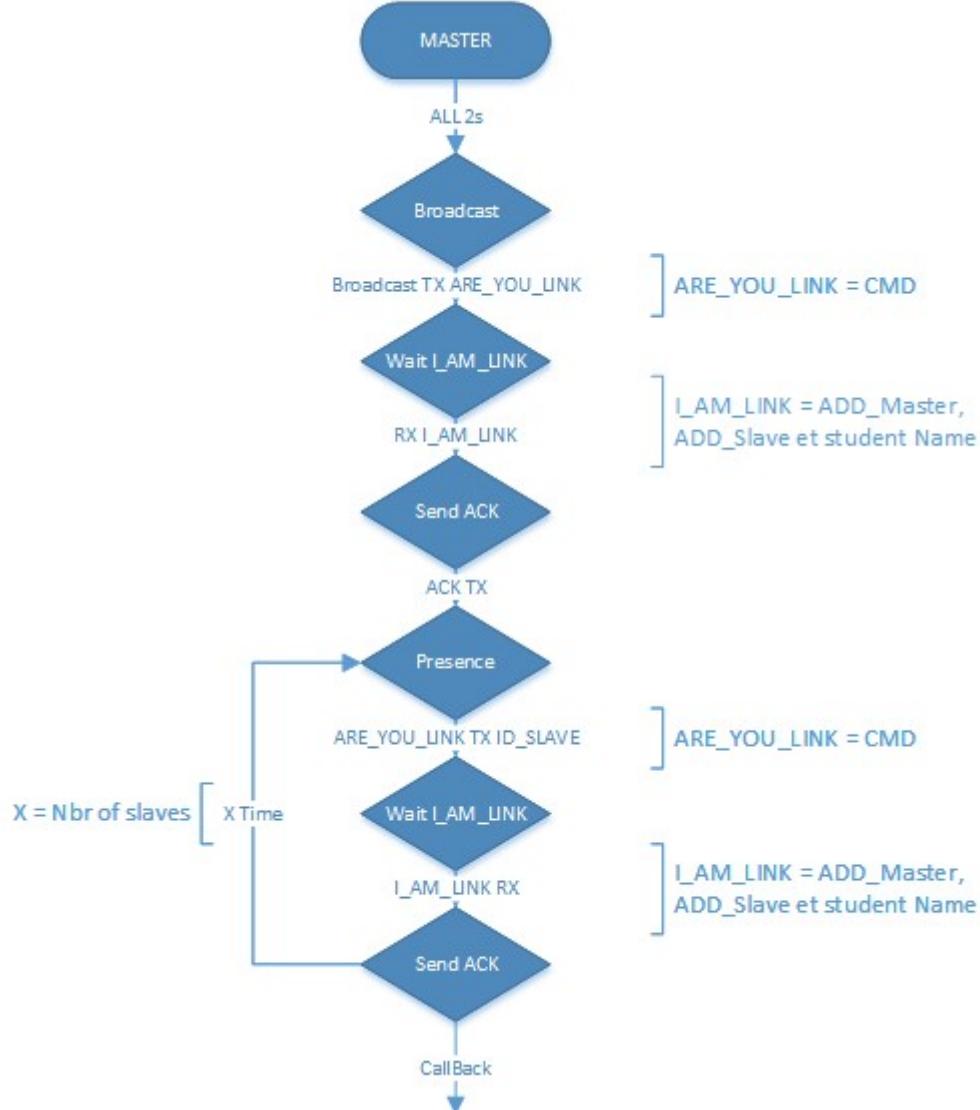
$$\text{Prescaler}_{min} = \frac{T * \text{SYSCLK}}{2^{16}} = \frac{100 * 10^{-3} * 40 * 10^6}{65536} = 61.03 \rightarrow 64$$

La valeur du timer doit être de :

$$\text{Période[tick]} = \frac{T * \text{SYSCLK}}{\text{prescaler}} - 1 = \frac{100 * 10^{-3} * 40 * 10^6}{64} - 1 = 62499$$

5.1.2 Structogramme/diagramme

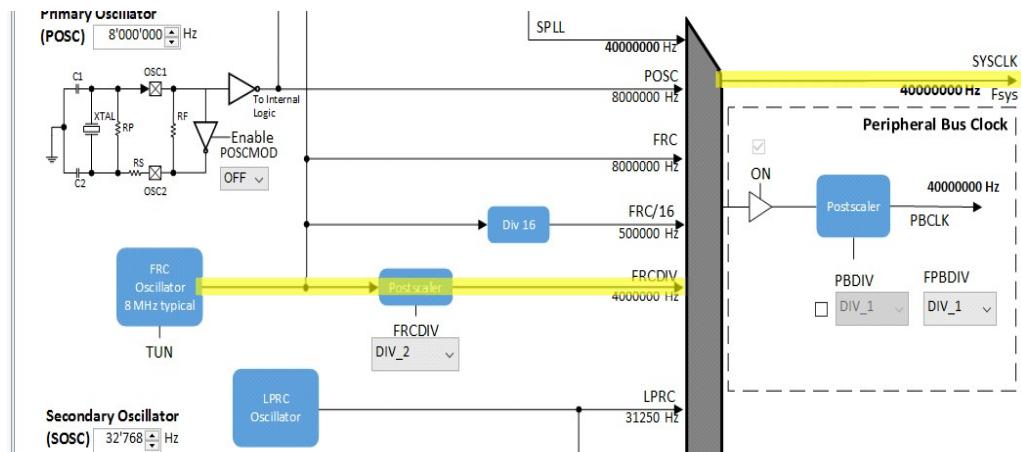
5.1.2.1 Pulling



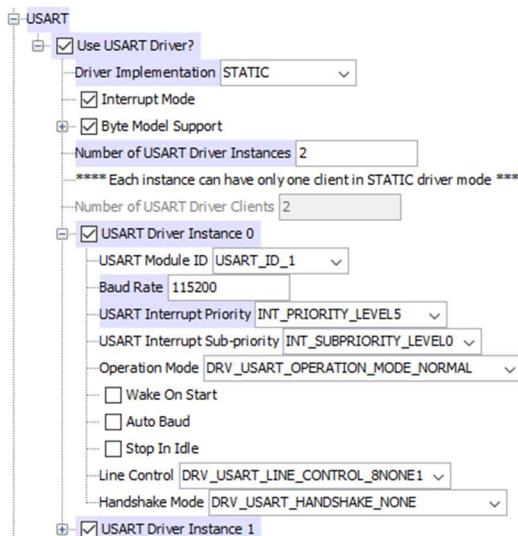
5.2 Slave

5.2.1 Configuration

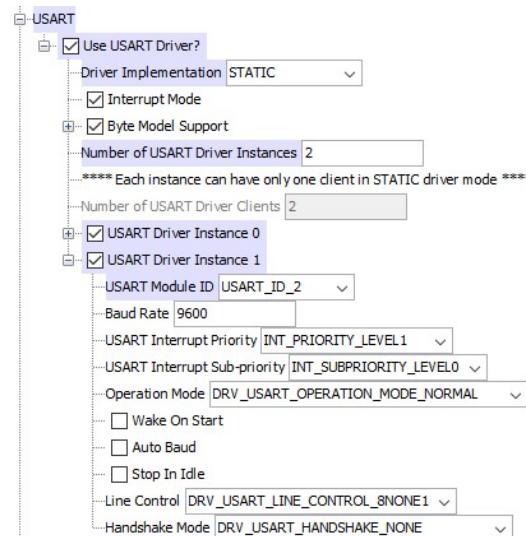
5.2.1.1 System Clock



5.2.1.2 UART



UART Xbee



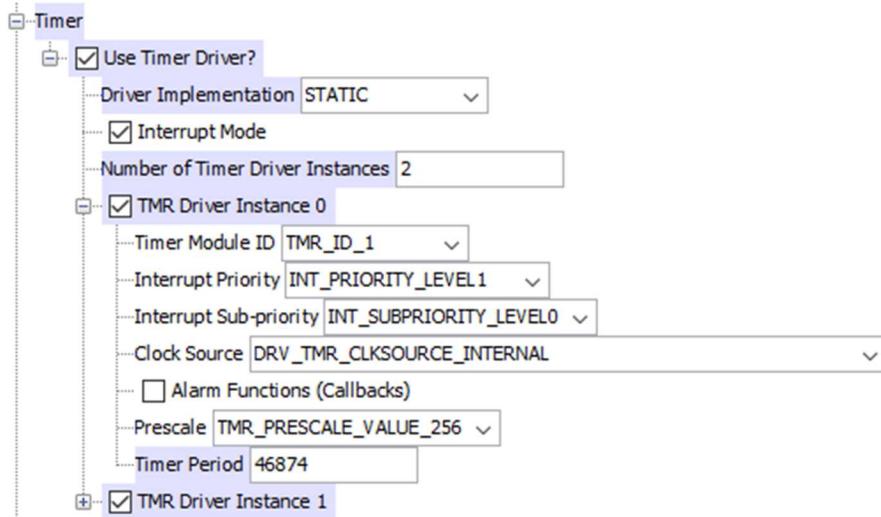
UART USB Slave

J'utilise les même paramètre d'uart du Xbee que le programme maître

Le programme Slave communiquera en USB avec une application windows (2126_AffichageMatriciel) via Uart donc j'ai décidé de reprendre les paramètres de l'UART

5.2.1.3 Timer

5.2.1.3.1 Timer 1



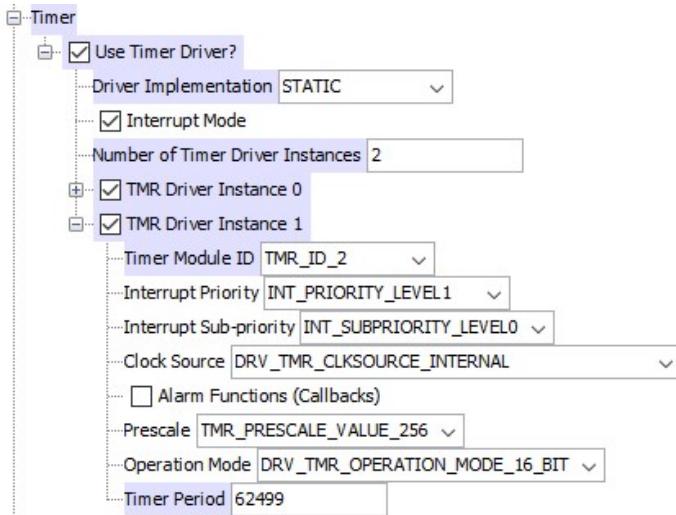
Ce timer est utilisé lorsque la carte Slave n'est pas link au Master. Il fera clignoter la led rouge à la fréquence donnée.

Le temps du timer est de :

$$Temps = \frac{(Période[tick] + 1) * prescaler}{SYSCLK} = \frac{(46874 + 1) * 256}{40 * 10^6} - 1 = 300 \text{ ms}$$

Ce timer était déjà implémenter dans la version A.

5.2.1.3.2 Timer 2



Ce timer sera utilisé pour les 20s de blockage de la carte slave

La période cible sera de 400ms. Je ne mets pas le timer à 20s pour ne pas utiliser un timer 32bits.

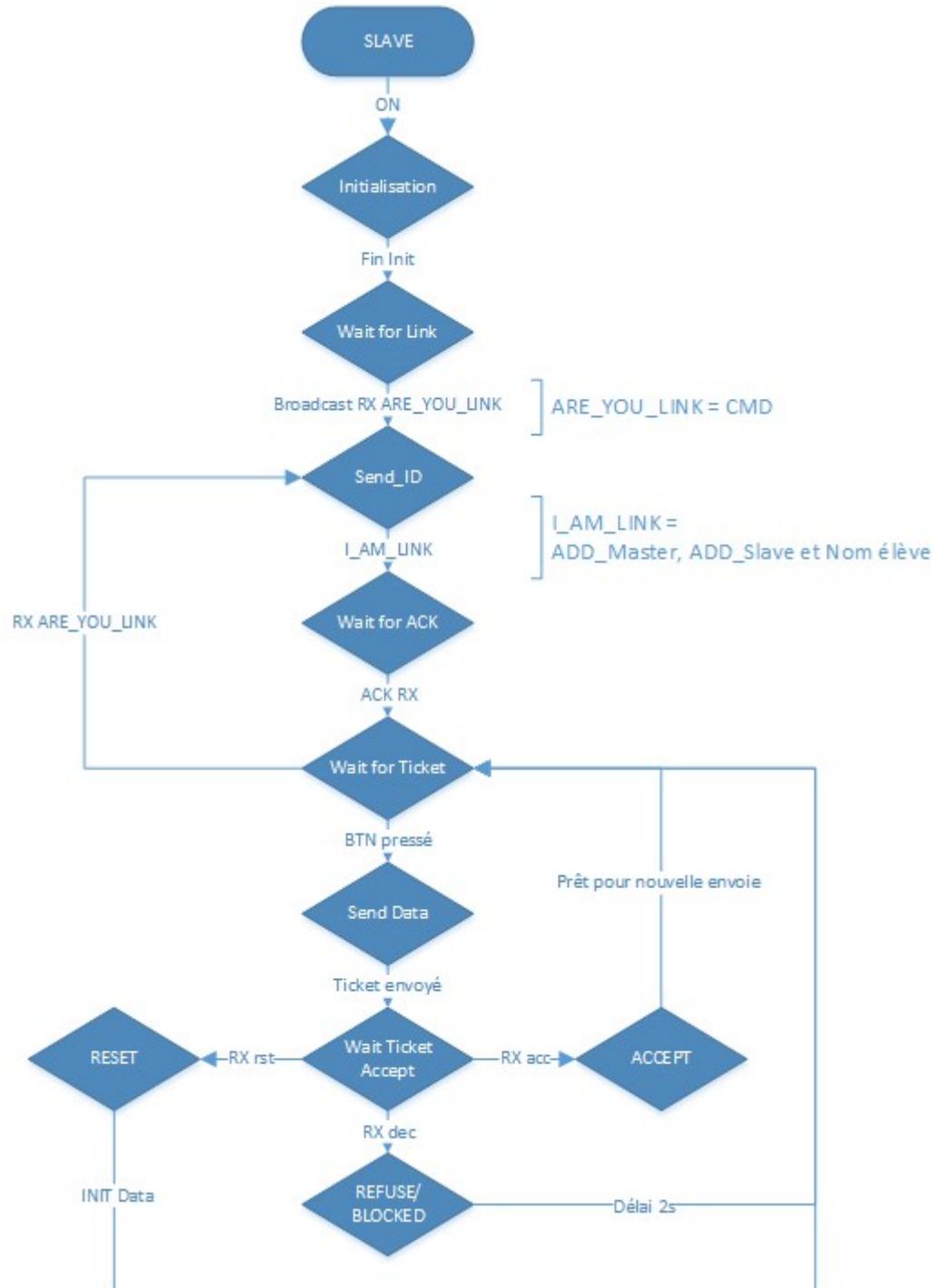
Le clock du microcontrôleur(SYSCLK) est de 40Mhz

$$\text{Prescaler}_{min} = \frac{T * \text{SYSCLK}}{2^{16}} = \frac{400 * 10^{-3} * 40 * 10^6}{65536} = 244.14 \rightarrow 256$$

La valeur du timer doit être de :

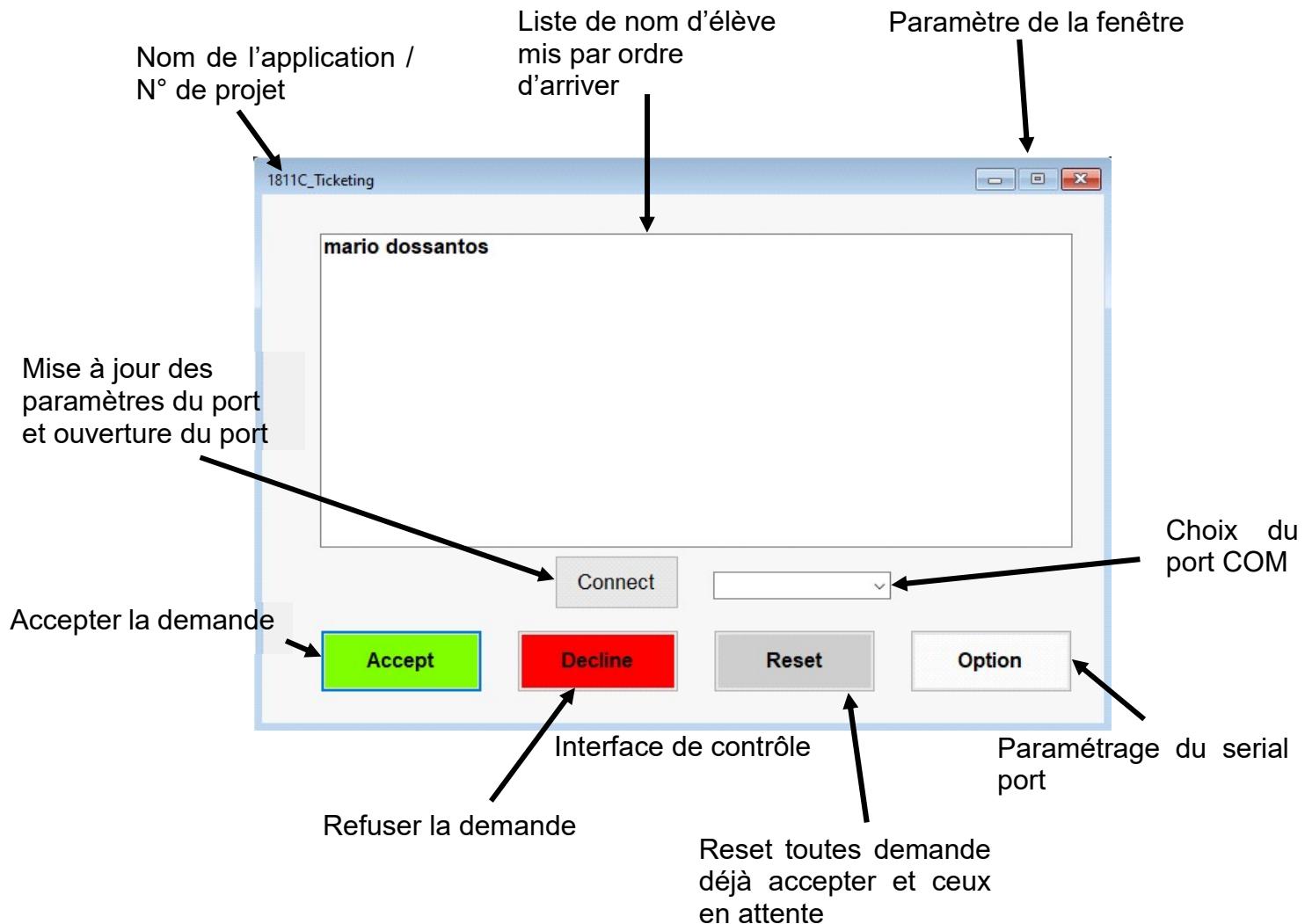
$$\text{Période}[tick] = \frac{T * \text{SYSCLK}}{\text{prescaler}} - 1 = \frac{400 * 10^{-3} * 40 * 10^6}{256} - 1 = 62499$$

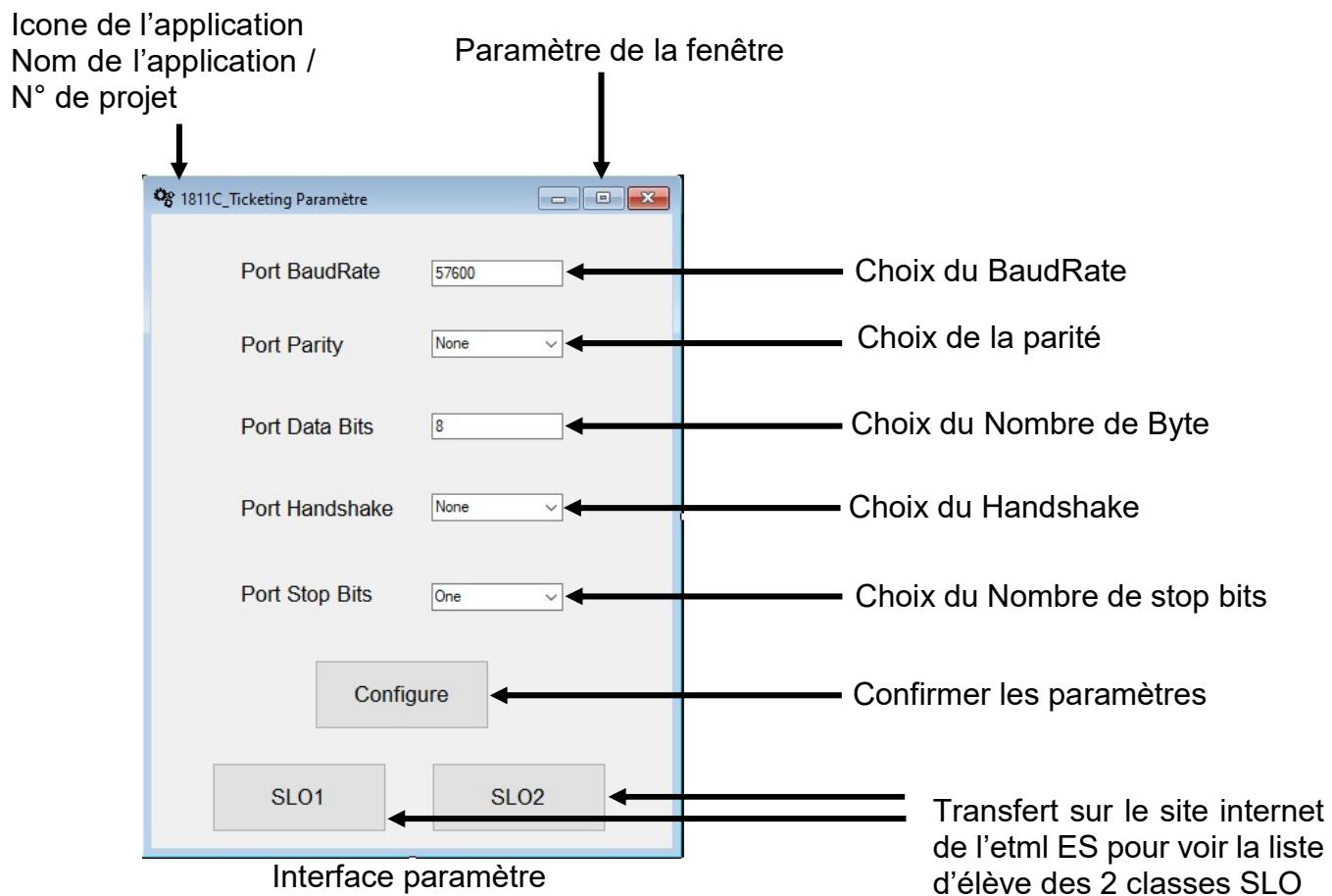
5.2.2 Structogramme/diagramme



5.3 Application C#

5.3.1 Interface





J'affiche les valeurs de paramétrage par default du serial port

Dans la deuxième interface le bouton **Configure** permet de confirmer que les bonnes valeurs ont été inscrit. Et le bouton **Connect** permet de fermer le port COM, de le reprogrammer avec les paramètre inscrit dans l'interface précédent et de rouvrir le port COM.

5.3.2 Etat

Par manque de temps je n'ai pu que faire l'interface de l'application.

J'essayerais de faire fonctionner une partie pour les portes ouvertes et si possible complètement pour la présentation.

5.4 Trames

Chaque élément est en 32bits (8 bytes) sauf à une exception qui le nom de l'élève

5.4.1 Commande

Dans la version A des valeurs ont été donnée pour chaque commande.

Je n'ai aucune idée si ces valeurs ont été définie par hasard ou non. Je les ai laissées pour l'instant car je n'ai pas eu de problème avec mais si par hasard une adresse de xbee correspondrait à une de ces valeurs je ne peux pas garantir le fonctionnement de la carte

```
#define ENVOI_TICKET 0x917283bd
#define TICKET_ANNULER 0xccc18ca5

#define TICKET_ACCEPT 0x00e0d135
#define TICKET_REFUSE 0xc5fb3140
#define TICKET_RESET 0xb0bfe0fc
#define BLOCKED 0xcc9ebb17

#define ARE_U_LINK 0x3103de90
#define I_AM_LINK 0xcd05a45

#define ACK 0x1blac4f6
```

5.4.2 Broadcast

Broadcast	ADD Master	ARE_YOU_LINK
-----------	------------	--------------

Broadcast = 0xFFFFFFFF

ADD Master = récupération de l'adresse du Xbee

ARE_YOU_LINK = adresse prédéfini(0x3103de90)

5.4.3 Donnée

Adresse expéditeur	Adresse destinataire	Data
--------------------	----------------------	------

Adresse expéditeur = adresse de 32 bit qui peut soit être celui du master ou celui du slave

Adresse destinataire = adresse de 32 bit qui peut soit être celui du master ou celui du slave

Data = nom d'un élève ou command

Nom de l'élève → prénom de l'élève + première lettre du nom de famille converti en hexadécimal

Commande → Envoie ticket, accepter, refuser, ...

Les valeurs sont prédéfinies

6 Test et mesure

6.1 Ecran TFT

Lorsque j'ai voulu tester l'écran avec le programme de la version A, j'ai pu rapidement voir que je n'arrivais pas à afficher les informations voulu mais seulement à allumer le backlight.

J'ai pu confirmer que le modèle précédemment utiliser était le adafruit 1770 et non le 2090 car c'était un modèle qui avait déjà été utiliser dans des projets passer.

Comme le drive du TFT est le même sur les 2 modèles je me suis dit qu'il fonctionnerait de la même manière.

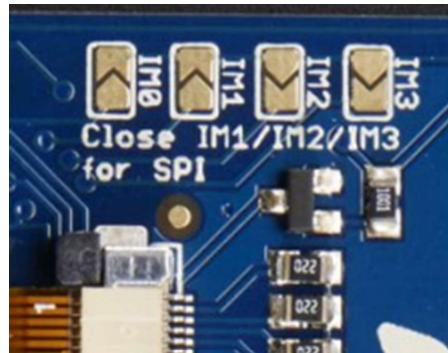
Mais lorsque je l'ai programmé j'ai constaté que seul le backlight fonctionnait. J'ai revérifier que je n'avais pas modifier une partie du code mais je n'ai trouvé aucune différence.

J'ai ensuite essayer de reprendre le code du driver donnée par adafruit sur github mais là aussi dans succès.

Après plusieurs essaye non concluant j'ai demandé à M.Castoldi, qui connaissait déjà un peu l'écran de la version A, s'il avait déjà eu ce genre de problème auparavant. Suite à cette discussion il m'a proposé d'utiliser le même écran de la version A qui n'était pas baser sur un autre projet (1706x_Spectrometre) pour vérifier si le problème se trouvait dans mon code ou mon écran.

Après avoir tester mon code avec l'écran du spectromètre j'ai constaté que l'écran s'allumait. Donc j'ai un problème avec mon écran.

Dans différent forum j'ai pu lire qu'il y avait souvent ce genre de problème (backlight qui s'allume mais pas l'écran) et le problème souvent signaler était le connecteur pcb flexible qui était mal brancher. Après avoir débraser mon écran j'ai constaté que ce n'était pas un problème de connexion. Mais cela m'a permis de voir que je n'avais pas braser les jumpers pour communiquer en SPI.



Après avoir rébraser mon écran et reprogrammer, j'ai constaté qu'il n'y avait pas de changement.

Lorsque je cherchais le problème j'ai pu voir que mon TFT était mal plaquer sur la carte dans j'ai réchauffer les brasures. Après cela j'ai pu constater que l'écran s'allumais.

Au début mon écran ne fonctionnait pas car j'essayais de communiquer en SPI et lui mode 8bits et ensuite je pense que j'avais soit une brasure froide soit un court-circuit que je n'avais pas vu.

6.2 Driver TFT

Il y a plusieurs fonctions qui fonctionne entre les 2 LCD mais aussi certaine qui ne fonctionne pas ou presque pas.

Fonctionnel :

- `tft_setTextColor` : qui permet de changer la couleur du texte
- `tft_setCursor` : permet de modifier l'emplacement du curseur
- `tft_writeString` : permet d'afficher un texte
- `tft_drawFastHLine` : permet de dessiner une ligne

Pas totalement fonctionnel :

- `tft_setTextSize` : permet de changer la taille du texte. On est obligé de mettre le texte en taille 1 sinon il empêchera toute écriture.

Pas fonctionnel :

- `tft_fillRect` : permet de dessiner un rectangle
- `tft_fillScreen` : permet de remplir l'écran d'une couleur

Ces 2 fonction ne font juste rien. N'empêche pas l'écriture de texte.

6.3 Xbee

6.3.1 Code

Il m'était impossible d'envoyer un message par Xbee. En lisant le rapport précédent (Rapport projet 1811) j'ai pu lire la phrase suivante :

Le module RF est beugé sans comprendre pourquoi, car avant son implémentation sur mes cartes les modules communiquaient bien.

Donc cela me confirme que les Xbee arrive bel et bien à communiquer entre elle.

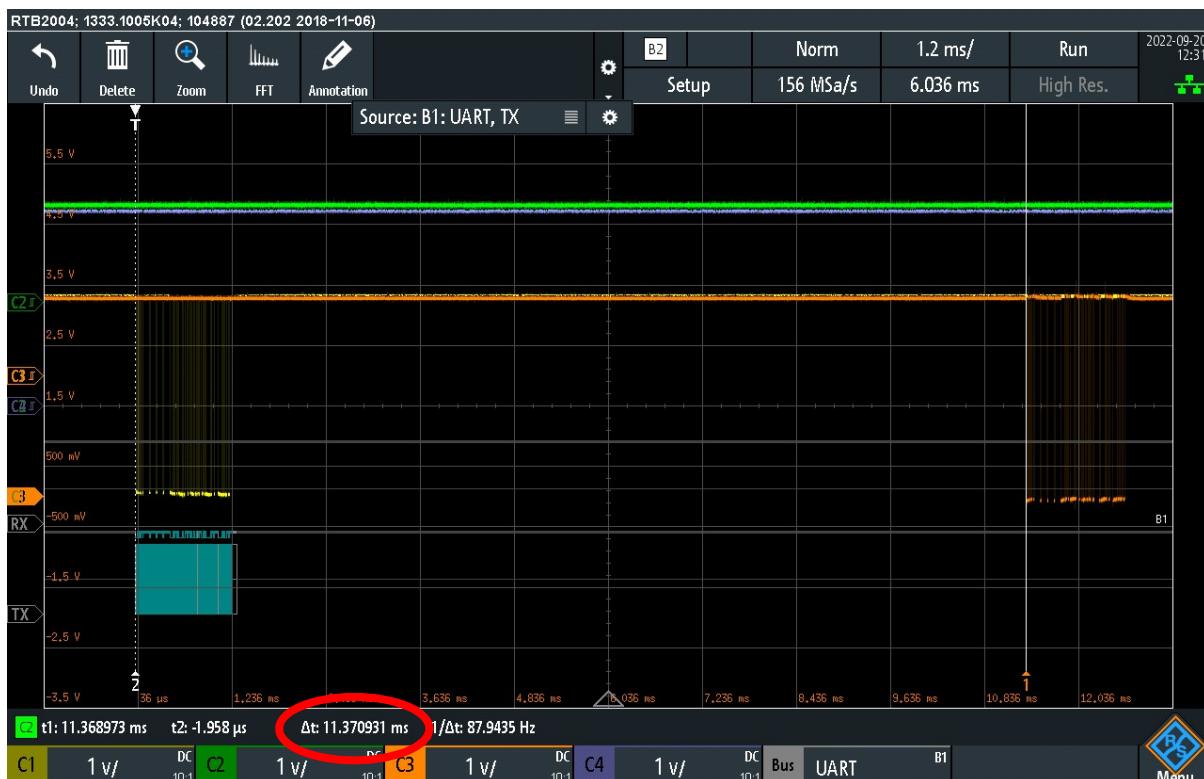
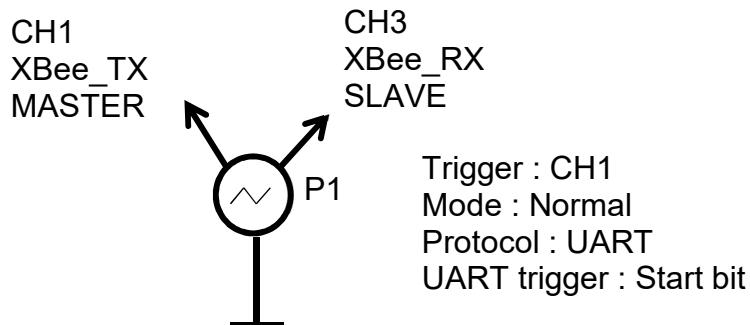
Durant 2 jours j'ai essayé de les faire fonctionner mais sans succès.

J'ai demandé de l'aide à M.Castoldi qui s'en est servi et qui fonctionnait très bien.

Après discussion avec lui nous avons conclu que le problème ne venait pas de ma carte ou de mon code mais du code du Xbee qui s'après M.Castoldi était un peu bancal.

J'ai donc reprogrammer mes carte Xbee avec le firmware de M.Castoldi (20200214_Module_Xbee_06_SCA) du projet 1623

6.3.2 Temps de communication



On peut voir sur l'oscilloscopogramme que le temps que le xbee met à transmettre les données est d'environ 11.37ms.

J'ai fait d'autres mesures et j'ai pu constater que le temps de transmission variait entre 10 et 12 ms.

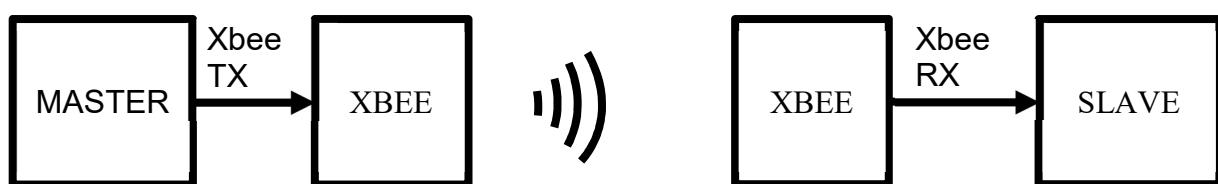


Schéma d'envoi d'une trame

7 Planification

Je savais que je ne pouvais pas passer directement du design de mon PCB au montage mais j'ai oublié de l'afficher dans ma planification.

Le montage de ma carte été faite plus tard que prévu car il y avait les 5 jours de fabrication/livraison au lieu des 3 jours qui était prévu.

J'ai passé plus de temps que prévu sur mon programme car le code de j'ai récupérer de l'ancienne version était truffé d'erreur qui n'était pas préciser dans le rapport et qu'il y avait des parties qui n'était même pas fait. En plus j'ai eu énormément de problème avec le Xbee et l'écran TFT.

Je n'ai pas pu avancer sur la partie mécanique car j'ai pris trop de temps sur la programmation.

8 Etat d'avancement

Affichage	NOK
Réception Nom élève	OK
Communication Xbee	OK
Envoie de ticket	OK
Gestion des tickets	En cours
Gestion de la liste de question	En cours
Application C# gestion des donnée	NOK
Communication C# - Master	NOK
Boitier Master	NOK
Boitier Slave	NOK

Je vais essayer de faire fonctionner les communications entre Slave et Master et de les afficher sur l'application C# pour les portes ouverts du 30.09.

Et je vais essayer de faire fonctionner l'application C# (gestion de la liste) pour ma présentation du 07.10.

L'affichage n'est pas fonctionnel et je ne pense pas m'attarder dessus durant la prochaine semaine.

Le boitier Slave est presque fini d'être dessiner (sur solidworks) il manquera plus qu'à faire le plan de perçage avec les cotations et de le percer le boitier qui a déjà été réceptionner

Le boitier Master est presque fini d'être dessiner (sur solidworks) il manquera de dessiner l'ouverture pour le TFT et la carte SD. Ensuite il faudra faire le plan de perçage avec les cotations et de le percer le boitier qui a déjà été réceptionner

9 Amélioration

Je pense qu'il serait utile d'enlever l'écran TFT et la carte SD si nous ne les utilisons pas. Cela permettra de réduire la taille de la carte, les coûts de fabrication seront réduit et cela nous permettrait de prendre un boitier plus petit qui sont souvent moins cher.

Finaliser le Code. Je vais essayer d'avancer le plus possible sur le code jusqu'à la présentation.

10 Conclusion

Ce projet avait l'air assez simple sur le papier mais certain problème firmware et la reprise d'un projet existant mon rapidement fait penser le contraire.

La partie hardware c'est bien dérouler et j'ai pu commencer la programmation C# et les dessins mécanique en attendant la livraison de mes cartes.

La partie firmware était beaucoup plus compliqué car j'ai dû pas mal modifier le code à cause de la carte SD et de l'écran TFT qui ne fonctionnait pas comme je le souhaitait.

Ce n'est pas la première fois que je reprends un projet pour le continuer/corriger mais à chaque fois je me rends compte se serait plus simple de commencer depuis le début. Dans mon cas la partie hardware était assez facile.

La version A n'avait pas beaucoup de problèmes, même si j'en ai trouvé certain qui n'était indiqué nulle part. La partie firmware était beaucoup plus compliqué car le code n'était pas commenté et que certaine partie du code, qui était écrit dans le rapport comme fait, ne l'était pas.

J'ai beaucoup appris de ce projet je suis un peu déçu du résultat actuel mais je vais continuer à avancer dessus jusqu'aux présentation. J'ai pu constater le résultat que peut avoir un rapport auxquelles ils manquent des informations important surtout au niveau des composants et de même pour un code qui manquent de commentaire.

Lausanne, le 27.09.2022

Dos Santos Mario

11 Annexe

11.1 Cahier des charges

11.2 Planification

11.3 Procès-Verbaux

11.4 Journal de travail

11.5 Schéma électrique complet

18111C_Ticketing_Schéma_électrique_Master

18112C_Ticketing_Schéma_électrique_Slave

11.6 Liste de pièce

11.7 Listing Code

18111C_Ticketing_Listing_Code_Master

18112C_Ticketing_Listing_Code_Slave

1811C_Ticketing_Listing_Code_Application

11.8 Mode d'emploi

11.9 Datasheets

Système de ticketing pour questions d'étudiants

N° projet 1811C

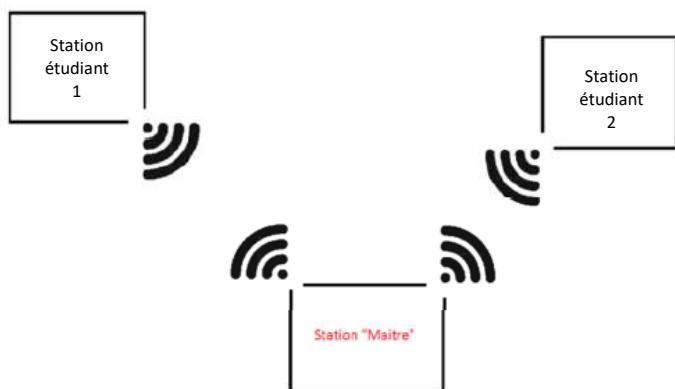
Mandataire

Entreprise/Client:	ETML-ES	Département:	SLO
Demandé par (Prénom, Nom):	Serge Castoldi / J. José Moreno	Date:	08.07.2022

1 Objectif - Cahier des charges

Le but de ce travail de diplôme est de reprendre le projet **1811 Système de ticketing pour questions d'étudiants** pour le modifier/finaliser et le rendre opérationnel. Pour rappel, il fait suite à la demande des étudiants de première année SLO (2017/2018) d'avoir un système de ticket électronique pour que les enseignants puissent gérer au mieux les demandes des étudiants - questionnement lors des cours ou travaux pratiques.

Il consiste en 2 PCBs, l'un du côté de l'enseignant (station maître), l'autre du côté de l'étudiant (station esclave).



1.1 Généralités

La transmission entre module est faite via RF.

Prévoir le cas où plusieurs salles de classe sont équipées : les modules enseignants doivent traiter uniquement les demandes des modules étudiants de la classe.

Partir du projet 1811 (mis à disposition), et modifier selon indications ci-dessous.

1.2 Partie maître

L'étudiant réalisera un PCB pour la partie maître (1 exemplaire) avec les améliorations suivantes

1.2.1 Hardware :

- Ajouter une connexion USB afin de déterminer l'utilisateur
- Contrôler si erreurs à corriger
- Remettre en forme PCB pour mise en boîtier du commerce (acheté)
- Mise en boîtier

1.2.2 Firmware:

- Mise au point – correction des erreurs et finalisation.

1.2.3 Software

- Développement d'un programme Windows de pilotage et affichage de la liste des émetteurs de questions en cours.
L'enseignant doit pouvoir confirmer avoir répondu à une question : la file se « vide » alors d'une personne et un son doit être émis.
- But : projeter la liste au beamer en live. La communication se fait via l'USB.

1.3 Partie étudiant

L'étudiant réalisera un PCB pour la partie étudiant (2 exemplaires à assembler) avec les améliorations suivantes :

1.3.1 Hardware

- Ajouter une connexion USB afin de déterminer l'utilisateur
- Contrôler si des erreurs sont à corriger
- Remettre en forme PCB pour mise en boîtier du commerce (acheté)
- Mise en boîtier

1.3.2 Firmware

- Mettre au point, corriger les erreurs, finaliser
- Implémenter USB + compatibilité avec SW Windows (réutiliser tel quel SW Windows projet 2126_AffichageMatriciel qui permet d'obtenir le nom d'utilisateur via USB)

1.3.3 Software

- Reprendre projet 2126 (voir ci-dessus) pour l'obtention du nom utilisateur

2 A l'issue du projet de diplôme, l'étudiant fournira (liste non exhaustive) :

- Fichiers sources de CAO électronique du PCB réalisé (ALTIUM) + configuration logiciel (version utilisées)
- Tout le nécessaire pour fabriquer un exemplaire hardware : fichiers de fabrication (GERBER) / liste de pièces avec références pour commande (BOM) / implantation (prototype) / modifications, etc
- Fichiers sources de programmation microcontrôleur (.c / .h) + configuration logiciel (version utilisées)
- Fichiers sources de l'application C# (.cs) + configuration logiciel (version utilisées)
- Tout le nécessaire pour programmer le microcontrôleur (logiciel ou fichiers .hex), sous format numérique => utilisation de la structure de projet fourni par l'ES
- Tout le nécessaire à l'installation de programmes sur PC ou autres environnements utilisés durant le travail de diplômes
- Un rapport de diplôme contenant :
 - Les concepts du design Hardware :
 - Études des différents systèmes à implémenter (explications)
 - Choix des composants et dimensionnement de ceux-ci (calculs / simulation / autre)
 - Réalisation schématique / PCB / boitier / montage de la carte
 - Les concepts du design Firmware
 - Structogramme / flowchart / Pseudocode
 - Explication des algorithmes mise en place
 - Démonstration par calculs ou outils de debug des résultats obtenus
 - Validation des concepts mis en place
 - Les concepts du design Software
 - Structogramme / flowchart / Pseudocode
 - Explication des algorithmes mise en place
 - Démonstration par calculs ou outils de debug des résultats obtenus
 - Validation des concepts mis en place
 - Tests & Validation :
 - Méthodologie de tests
 - Mesure(s)
 - Validation des résultats
 - Correction(s) apportée(s) au design (Hardware / Firmware / Software)
 - Estimation des coûts pour le design développé (un prototype)
 - Etat d'avancement & problèmes rencontrés
 - Conclusion
 - Bibliographie / webographie / autre sources
- Les annexes :
 - Calculs détaillés des concepts
 - Listings complets des parties Firmware & Hardware que vous avez implémenté
 - Schématique + plan d'implémentation complète du PCB
 - Dessin / schématique du boitier
 - Mesures
 - Pages utilisée des différents datasheets ou documentations exploités

- Mode d'emploi du système développé pendant le diplôme
- Fichier de modifications pour une reprise ultérieure du projet
- Journal de travail
- PV de séances hebdomadaires
- Un prototype démonstrateur avec toutes les fonctions disponibles. Les parties fonctionnelles décrites en

3 Autres demandes / contraintes / conseils

- **Planifier** dans le détail les travaux demandés.
- Se référer au planning régulièrement, **vérifier son avancement**, rédiger son **journal de projet** quotidiennement.
- Commencer à **rédiger le rapport de diplôme le plus tôt possible**, et régulièrement tout au long du travail de diplôme.
- Prendre du temps, préparer sa réflexion, rechercher des apports théoriques et des exemples pratiques, **envisager plusieurs possibilités** avant de finaliser une solution.
- **Numéroter et dater tous les documents**
- En cas de **problème** (retard, objectif à revoir, difficulté rencontrée, etc.), se référer à l'enseignant et au mandataire au plus vite.
- Toutes les **décisions importantes**, tant au niveau technique qu'organisationnel, doivent être posées **par écrit** dans le PV de séance, le rapport de diplôme et /ou figurer dans le journal de projet, après discussion avec l'enseignant / le mandataire.

4 Documents de références

-

1811B_Ticketing_Planning

Exemple

5,3	Design PCB	<i>3 jours</i>	
		<i>2,5 jours</i>	

R Remise de mémoire

1811B_Ticketing_Planning

Exemple

5,3	Design PCB	<i>3 jours</i> 2,5 jours	R	Remise de mémoire
-----	------------	-----------------------------	---	-------------------

1811B_Ticketing_Planning

Exemple

5,3	Design PCB	<i>3 jours</i> 2,5 jours	R	Remise de mémoire
-----	------------	-----------------------------	---	-------------------

1811B_Ticketing_Planning

Exemple

5,3	Design PCB	<i>3 jours</i> 2,5 jours	R	Remise de mémoire
-----	------------	-----------------------------	---	-------------------

1811B_Ticketing_Planning

Exemple

5,3	Design PCB	<i>3 jours</i> 2,5 jours	R	Remise de mémoire
-----	------------	-----------------------------	---	-------------------

1811B_Ticketing_Planning

Exemple

5,3	Design PCB	<i>3 jours</i> 2,5 jours	R	Remise de mémoire
-----	------------	-----------------------------	---	-------------------

Procès-verbal du 24.08.2022

Présents

- M. Mario Dos Santos
- M. Juan José Moreno

État des lieux

- Correction de la version terminer
- Schémas terminer

Problèmes rencontrés

- Choix du Xbee à utiliser :
- Choix du régulateur de tension :

Solutions proposées

- Prendre le Xbee crée par l'école mais lors de la version A ne fonctionnait pas bien ou prendre un Xbee d'un fabricant et qui a déjà été utiliser dans d'autres projet.
- Prendre le régulateur de tension qui est souvent utilisé par l'école mais je risque de ne plus avoir de place ou prendre un plus petit régulateur mais qui est en fin de vie

Décisions prises

- Le choix du Xbee ne change rien car les pattes des deux composant sont identique ce qui permettrait de les interchanger s'il y a un problème.
-

Suite du projet / objectifs - jusqu'au mercredi 31 août

- Commencer et terminer le PCB de la carte Slave
- Commencer le PCB de la carte Master

Prochaine réunion :

Mercredi 31.08.2022, 13h00 , R110

Destinataires de ce PV

M.Moreno
M.Rossier

Lausanne le 24.08.2022

Dos Santos Mario

Procès-verbal du 31.08.2022

Présents

- M. Mario Dos Santos
- M. Juan José Moreno

État des lieux

- PCB terminer et commander au plus tard le 01.09

Problèmes rencontrés

- Microcontrôleur du PCB Slave en rupture de stock et non disponible à l'etml-es

Solutions proposées

Utilisation du dernier composant disponible mais ne pas respecter le cahier des charges (2 PCB Slave à assembler) ou changer de boitier du microcontrôleur ou bien changer de microcontrôleur.

Décisions prises

Utiliser le même microcontrôleur mais avec un boitier différent (SSOP au lieu de QFN) disponible à l'etml-es (plus de 20)

Suite du projet / objectifs - jusqu'au mercredi 31 août

- Installation des logiciels (Solidworks et Harmony)
- Faire les boîtiers sur Solidworks
- Comprendre les firmware des programmes fais précédemment et commencer la programmation de la Carte Slave

Prochaine réunion :

Mercredi 07.09.2022, 13h00, R110

Destinataires de ce PV

M.Moreno

M.Rossier



Ecole supérieure, école des métiers Lausanne
Rue de Sébeillon 12
CH-1004 Lausanne
Procès verbal 31_08.docx

Version 2.2
du 02.09.2021
www.etml-es.ch
| 1 / 2 |

Lausanne le 31.08.2022

Dos Santos Mario

Procès-verbal du 07.09.2022

Présents

- M. Mario Dos Santos
- M. Juan José Moreno

État des lieux

- Réception du PCB au plus tard le 08.09
- Les logiciels ont été installé (Solidworks et Harmony)
- Les programmes des deux cartes ont été avancer mais manque la partie communication
- J'ai lu et compris les firmwares précédemment fait

Problèmes rencontrés

- Pas de problème majeur rencontrer
- Faible compétence du C#

Solutions proposées

- Demander de l'aide à mes camarades de classe, aux professeurs et chercher des solutions sur internet

Décisions prises

- J'ai décidé de reporter les plans de fabrication des boîtiers pour me concentrer sur l'application C#

Suite du projet / objectifs - jusqu'au mercredi 14 septembre

- Faire les boîtiers sur Solidworks
- Montage des trois cartes
- Tester mes cartes

Prochaine réunion :

Mercredi 14.09.2022, 13h00, R110

Destinataires de ce PV

M.Moreno
M.Rossier

Lausanne le 07.09.2022

Dos Santos Mario

Procès-verbal du 14.09.2022

Présents

- M. Mario Dos Santos
- M. Juan José Moreno

État des lieux

- Le boitier Slave a été fait et le boitier Master est presque terminer (Solidworks)
- Le montage des trois cartes a été fait sans problème de design
- Mes cartes ont été tester. Le microcontrôleur peut être programmer et mes alimentations sont OK

Problèmes rencontrés

- La communication de l'UART fait dans la précédente version bloquait souvent à cause d'une mauvaise gestion des demandes/attentes.

Solutions proposées

- La carte Slave reviendra à son état initial s'il ne reçoit pas le ACK du Master lors de la connexion.
- La carte Master enverra périodiquement une demande de connexion.

Décisions prises

Suite du projet / objectifs - jusqu'au mercredi 21 septembre

- Gérer la communication UART(Xbee) entre la carte master et les slaves.
- Gérer la communication UART(USB) entre la carte maître et l'application C#

Prochaine réunion :

Mercredi 14.09.2022, 13h00, R110

Destinataires de ce PV

M.Moreno
M.Rossier

Lausanne le 14.09.2022

Dos Santos Mario

Procès-verbal du 21.09.2022

Présents

- M. Mario Dos Santos
- M. Juan José Moreno

État des lieux

- Communication Xbee fonctionnelle

Problèmes rencontrés

- Avec le code utiliser il est impossible de savoir si les cartes sont toujours connectées

Solutions proposées

- Modifier la table des états de la communication uart (Xbee)

Décisions prises

- Mettre un pulling

Suite du projet / objectifs - jusqu'au mardi 27 septembre

- Gérer la communication UART(Xbee) entre la carte master et les slaves.
- Gérer la communication UART(USB) entre la carte maître et l'application C#

Prochaine réunion :

Destinataires de ce PV

M.Moreno
M.Rossier

Lausanne le 21.09.2022

Dos Santos Mario

22.08.2022 :

- 10h00 – 10h30 Information Diplôme
- 10h30 – 11h30 Reprise du projet 1811A Ticketing. S'informer sur les problèmes HW et/ou Software de la version A.
- 12h30 – 14h00 Commencer les différents documents et les mettre à jour (Journal de travail, planification, rapport).
Je ne possède pas encore les droits de modification du projet -> je ne peux pas arranger le dossier pour ajouter une version B.
- 14h00 – 14h15 Lecture cahier des charges + discussion avec M.Moreno
Question :
Enlever écran ? -> l'écran ne dérange pas et on ne veut pas trop modifier le projet donc on le laisse
Enlever Carte SD ? on peut laisser le connecteur même si l'on ne connecte pas de carte sd
Quel type d'USB utiliser ? On peut laisser celui qui est déjà implémenter mais il ne faut pas oublier de connecter les pins de données.
- 14h15 - 14h30 Recherche d'information sur le Xbee de l'école (1623).
Possible de communication sur 1 canal et non en broadcast ? -> Si aucune modification n'a été faites. Non, car pas implémenter.
- 14h30 – 16h45 Recherche de solution pour la communication USB (voir possibilité de l'IC pour la communication uart)

23.08.2022 :

- 07h30 – 08h30 Faire les corrections de la version A (version précédente)
- 08h30 – 10h00 Recherche LCD utilisé dans la version précédente
- 10h00 – 12h00 Ajout de la partie communication USB/UART
- 13h00 – 14h40 Corriger les warnings du schéma + ressayer de faire fonctionner la version A
- 14h40 – 16h00 Commencer le Rapport
- 16h00 – 16h30 Recherche des composants critique

24.08.2022 :

- 07h30 – 08h00 Modifier le régulateur de tension 5V à 3V3 car l'ancien modèle n'est pas en stock.
- 08h00 – 09h00 Recherche de boitier
- 09h00 – 11h30 Documentation

12h25 – 12h35	Disjoncteur qui saute
12h35 – 13h10	Documentation
13h10 – 14h10	Meeting avec M.moreno
14h10 – 16h30	faire en sorte que les 2 Xbee soit compatible

25.08.2022 :

07h30 – 08h00	Terminer les schémas électriques
08h00 – 08h30	Rechercher un nouveau boitier pour la carte Slave
08h30 – 09h30	problème de Template de altium -> Recréer un nouveau projet et tous transférer en vérifiant si tout est bien à jour
09h30 – 10h00	Vérifier les footprints
10h00 – 11h40	Router PCB Slaves
12h10 – 12h35	Ajouter à la liste de commande les composants les plus critique de digi-key
12h35 – 14h00	Finir de Router le PCB Slaves
14h00 – 14h50	Cherche de boitier pour la carte Master
14h50 – 16h20	Avancer sur le rapport

26.08.2022 :

07h30 – 08h00	Choix du boitier carte Master (sur les différents modèles précédemment sélectionner)
08h00 – 09h00	Problème de Template de altium -> Recréer un nouveau projet et tous transférer en vérifiant si tout est bien à jour
09h00 – 09h20	Commencer le PCB Master
09h20 – 10h20	Rajout d'élément visuel sur le PCB Slaves
10h20 – 11h30	Faire la liste de pièce du PCB Slaves sur Altium
12h10 – 13h00	Finir la liste de pièce PCB Slaves sur Altium
13h00 – 15h00	Faire la liste de pièce PCB Master sur Altium + corriger les footprints
15h00 – 16h20	Avancer sur le placement des composants du PCb Master

29.08.2022 :

07h30 – 11h40	Router le PCB Master
12h30 – 16h35	Documentation + compléter liste de pièce

30.08.2022 :

07h30 – 11h30	Documentation
---------------	---------------

12h20 – 15h00	Documentation
15h00 – 15h30	Review avec M.moreno des schémas et PCB
15h30 – 16h30	Modification des PCBs (pas de composants bottom + faire deux projet PCB avec des footprint de microcontrôleur un QFN et un SSOP).

31.08.2022 :

07h30 – 10h00	Documentation
10h00 – 11h30	S'informer sur les différents firmware utiliser précédemment (1623, 1811A)
12h20 – 13h00	S'informer sur les différents firmware à utiliser (2126)
13h00 – 13h30	Meeting avec M.moreno
13h30 – 16h00	Corriger/modifier les schémas et PCB selon information meeting
16h00 – 16h50	Faire un panel

01.09.2022 :

07h30 – 10h00	Documentation
10h00 – 11h40	Programmation Partie Slave
12h30 – 16h00	Programmation Partie Slave

02.09.2022 :

07h30 – 11h40	Programmation Partie Maître
12h30 – 16h15	Programmation application C#

05.09.2022 :

07h30 – 11h40	Programmation application C# partie USB/serial port
12h30 – 16h20	Programmation application C# partie Option et gestion nom + adresse reçue (première fois reçue ou pas) et les ajouter à la liste si pas déjà présent.

06.09.2022 :

07h30 – 11h40	Programmation application C#, ajouter les valeurs par default de la communication serial (USB) + régler problème d'affichage des 2 fenêtres (ne plus cacher une fenêtre et en créer une autre mais plutôt de la rendre visible)
12h30 – 15h30	Programmation réglage application C#, problème lié aux port com (port qui restait ouvert)
15h30 – 16h20	Installation SolidWorks + commenter le code C# pour plus de compréhension.

07.09.2022 :

07h30 – 11h30	Rapport Documenter la partie programmation
11h30 – 12h00	Discussion avec expert
12h50 – 13h30	Meeting avec M.moreno
13h30 – 16h10	Rapport Documenter la partie programmation

08.09.2022 :

07h30 – 11h30	Faire boitier Slave et commencer boitier Master
12h20 – 16h30	Commencer montage

09.09.2022 :

07h30 – 11h30	Montage carte Master
12h00 – 16h20	Essayer de faire fonctionner l'écran

12.09.2022 :

07h30 – 11h30	Montage carte Slave
12h10 – 16h20	Faire fonctionner la partie réception de nom (2126) et l'implémenter dans la partie Slave

13.09.2022 :

07h30 –09h30	Faire fonctionner la partie réception de nom (2126) et l'implémenter dans la partie Slave
09h30 – 11h30	Ressayer de faire fonctionner l'écran
12h20 – 16h10	Continuer les essaient

Partie tester :

- Teste de différent programme (adafruit / forum)
- Lecture de registre
- Vérification du connecteur plat de l'écran

Aucun de ces tests n'a été concluent (le connecteur était bien connecté)

14.09.2022 :

07h30 –09h00	Faire fonctionner l'écran TFT
09h00 – 11h30	Modification du menu + test TF et carte SD
12h30- 14h20	faire fonctionner la communication UART
14h20 – 14h40	Meeting avec M.moreno

14h40 – 16h10 Communication UART

15.09.2022 :

07h30 –11h30 Communication UART

12h30 – 15h45 Communication UART

15h45 – 16h35 Discussion avec M.Castoldi au sujet du Xbee etml-es

16.09.2022 :

07h30 –09h30 Faire les modifications sur le Xbee de l'etml-es et rajouter la fonction récupération de l'adresse MAC du Xbee

09h30 – 11h30 Communication UART problème envoie de donnée

12h30 – 16h10 Communication UART problème envoie de donnée

20.09.2022 :

07h30 –12h30 Gestion des données

12h30 –13h10 Vérification de réceptions de l'adresse MAC (sens des valeur reçu)

13h10 – 16h00 perte de la communication entre les deux cartes. Recherche de solution

21.09.2022 :

07h30 –10h00 Problème de réception de donnée coté slave

10h00 –11h30 Discussion avec M.Moreno. Changer la table des états des cartes pour permettre un pulling

12h30 – 15h00 Modification de la partie slave

15h00 – 16h20 Modification de la partie Master

22.09.2022 :

07h30 –11h00 Modification de la partie Master + teste de communication entre les 2 cartes. Problème de réception coté Slave (reçois 8byte au lieu des 14)

11h00 – 12h20 Aide de M.Moreno pour trouver le problème

13h00 – 17h30 Continuer sur la partie communication (envoie coté slave et réception coté Master)

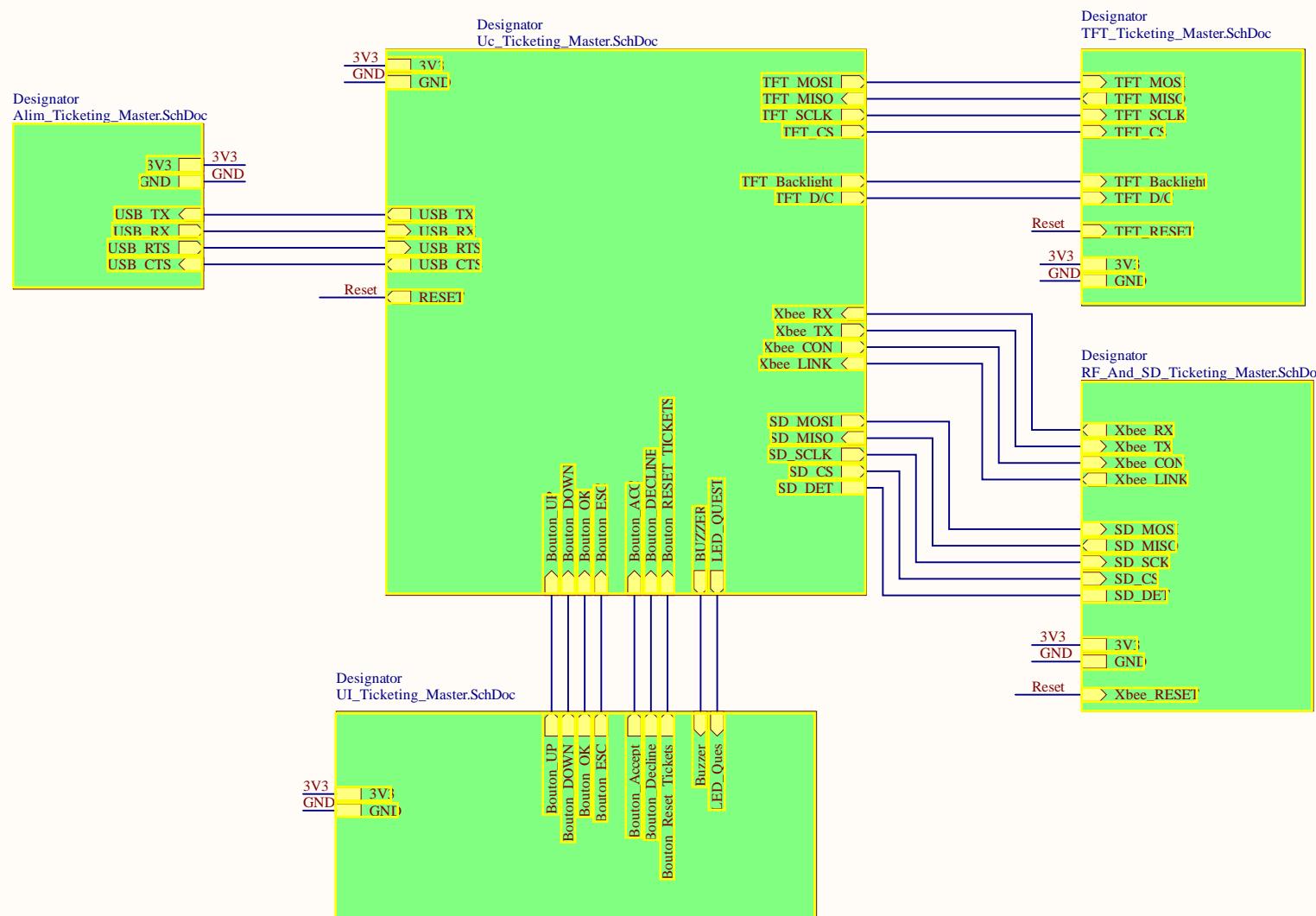
23.09.2022 :

07h30 –11h30 Réception de l'ACK

12h00 – 17h00 envoie du ticket partie Slave et réceptions partie Master

26.09.2022 :

07h30 – 11h30	Rédaction du rapport
12h20 – 16h20	Rédaction du rapport



Fichier : Ticketing_Master.SchDoc

Date : 25.09.2022

Version :

Heure : 10:52:30

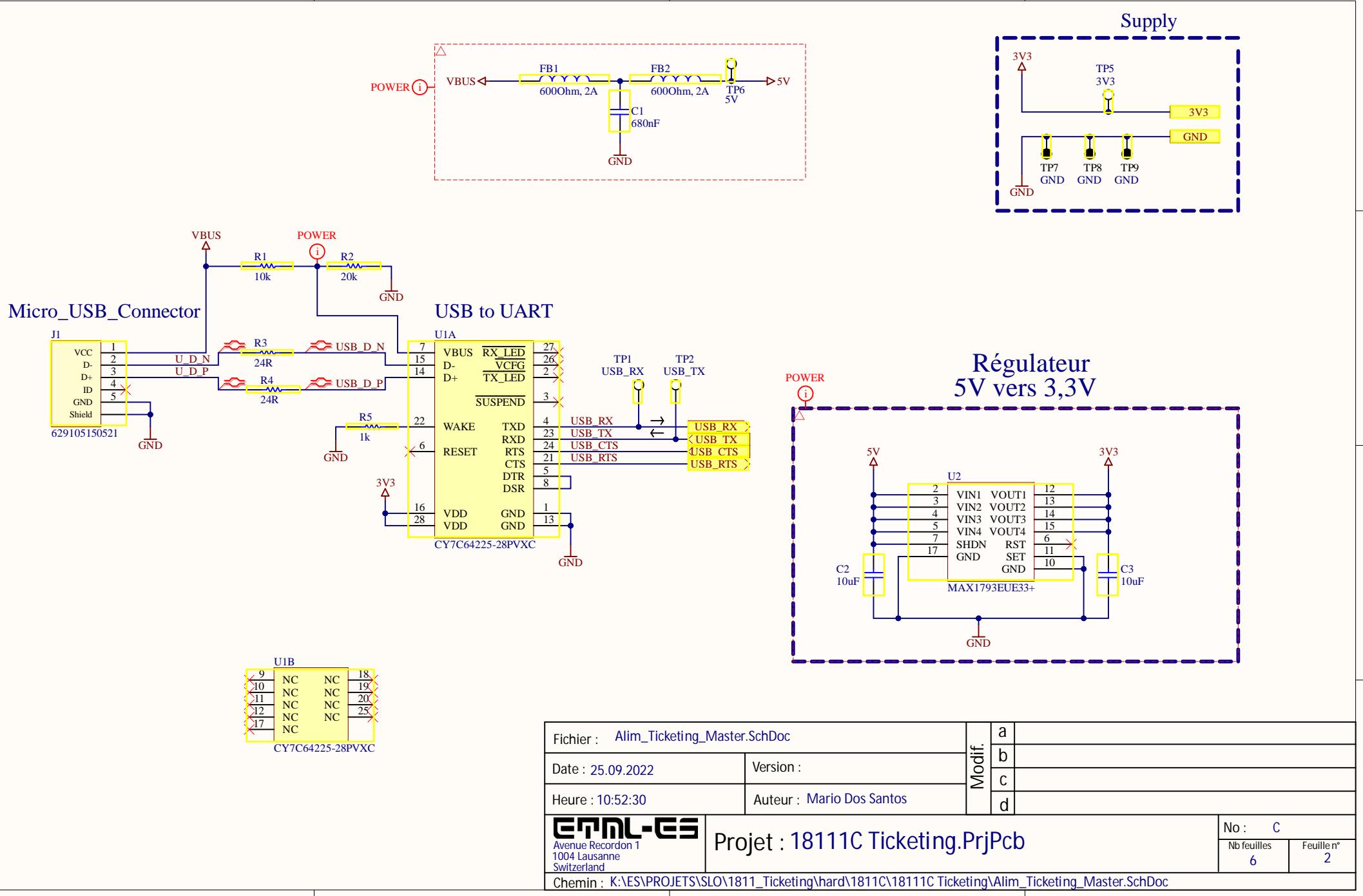
Auteur : Mario Dos Santos

Modif.

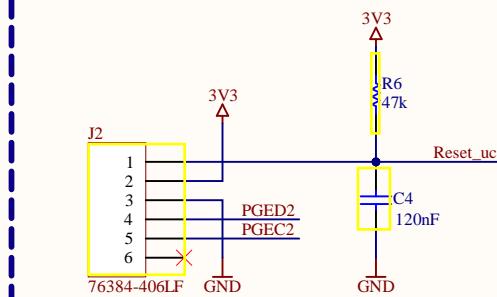
a
b
c
d

No :

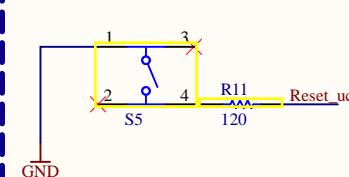
Nb feuillets
6Feuille n°
1



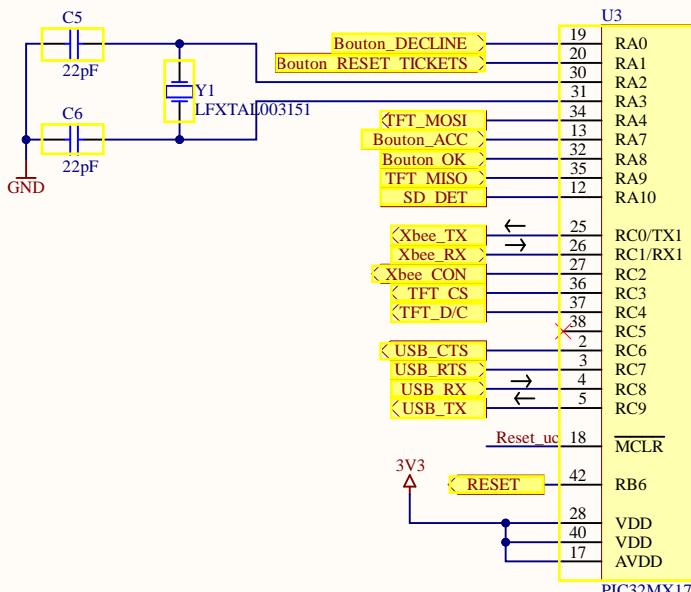
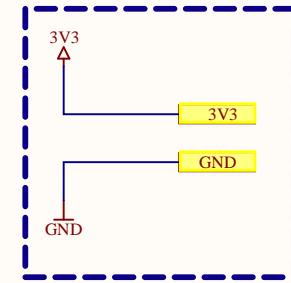
Debug Connector



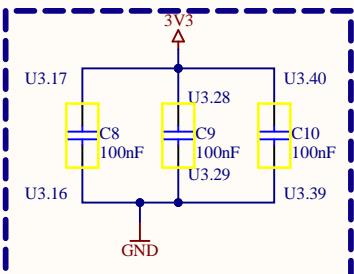
Buton_RESET



Supply



Decoupling Capacitors



Fichier : Uc_Ticketing_Master.SchDoc

Date : 25.09.2022

Heure : 10:52:31

Auteur : Mario Dos Santos

Modif.
a
b
c
d

No : C

Nb feuillets

6

Feuille n°

3

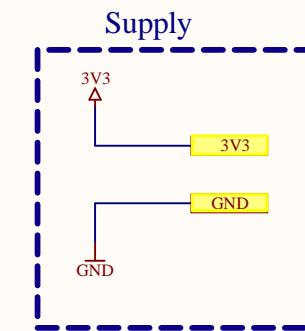
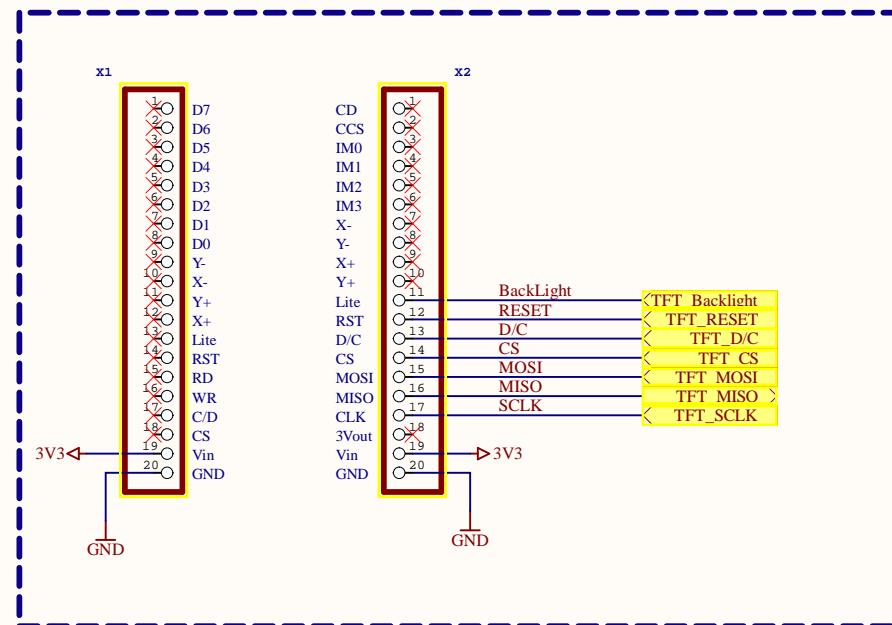
Projet : 18111C Ticketing.PrjPcb

Chemin : K:\ES\PROJETS\SLO\1811_Ticketing\hard\1811C\18111C Ticketing\Uc_Ticketing_Master.SchDoc

A

A

TFT Connector



B

B

C

C

D

D

Fichier : TFT_Ticketing_Master.SchDoc
Date : 25.09.2022

Modif.
a
b
c
d

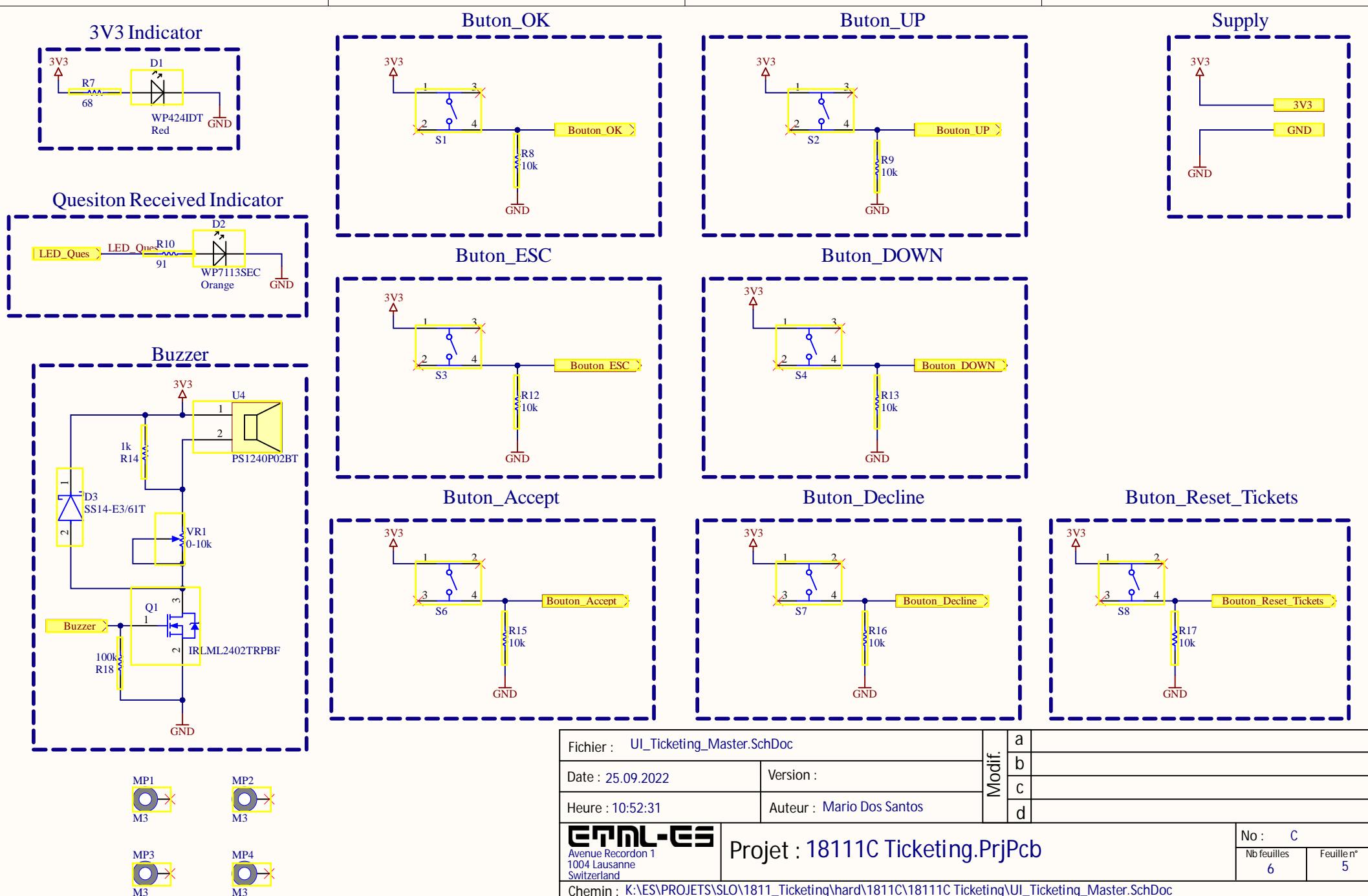
Version :
Heure : 10:52:31 Auteur : Mario Dos Santos

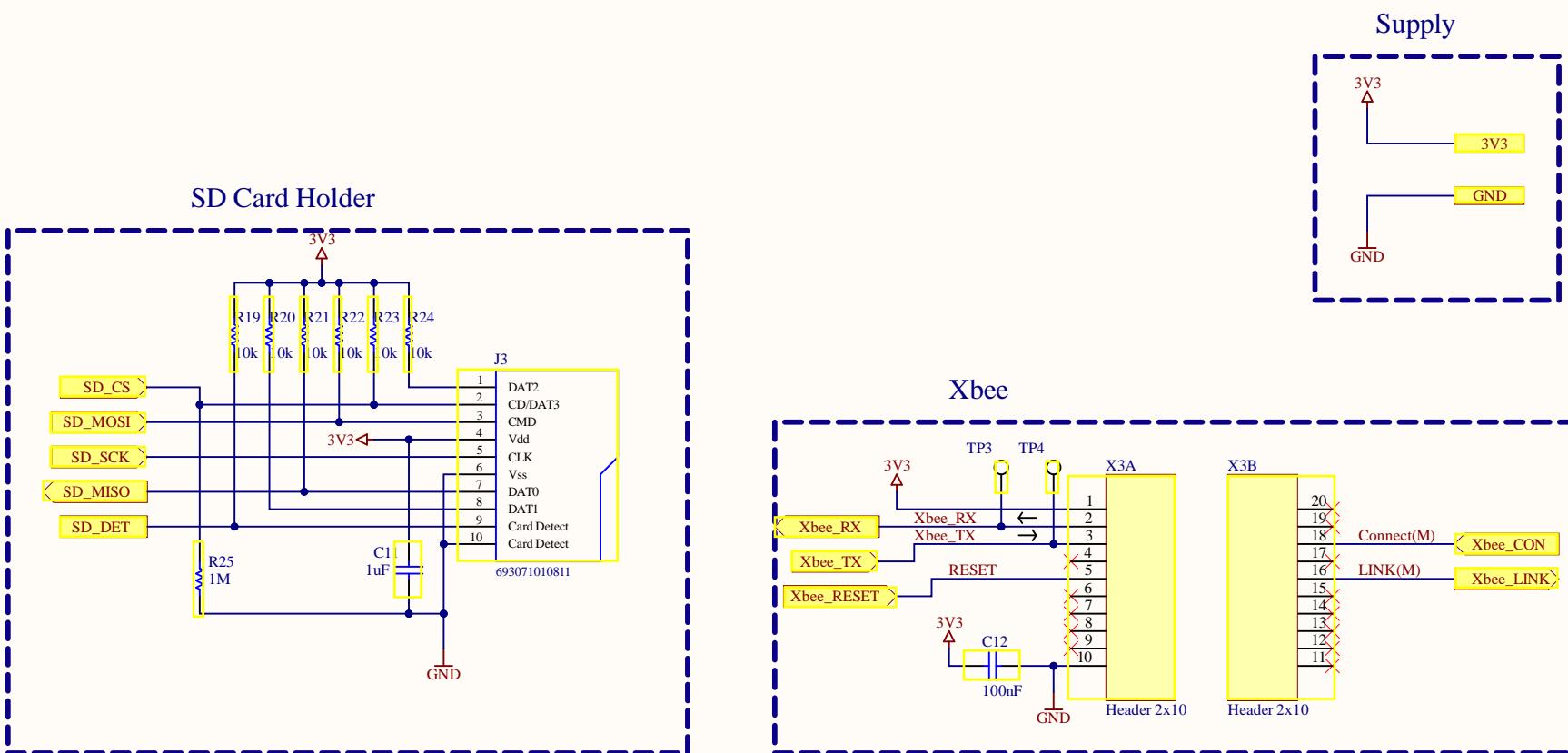
1

2

3

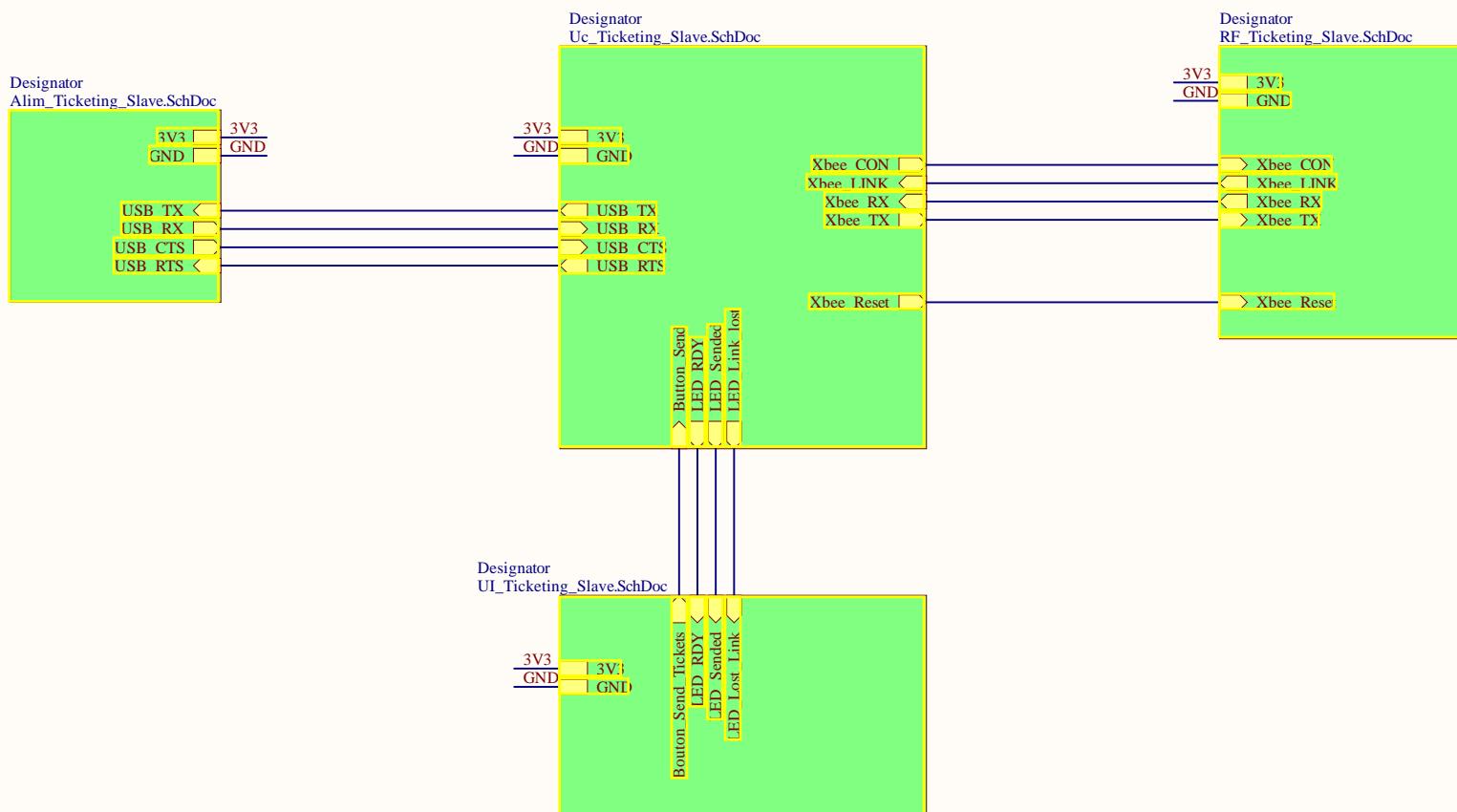
4





Fichier : RF_And_SD_Ticketing_Master.SchDoc		a	
Date : 25.09.2022	Version :	b	
Heure : 10:52:31	Auteur : Mario Dos Santos	c	
		d	
ENL-ES Avenue Recordon 1 1004 Lausanne Switzerland	Projet : 18111C Ticketing.PrjPcb	No : C	
		Nb feilles	Feuille n°
		6	6
Chemin : K:\ES\PROJETS\SLO\1811_Ticketing\hard\1811C\18111C Ticketing\RF_And_SD_Ticketing_Master.SchDoc			

A



B

C

D

A

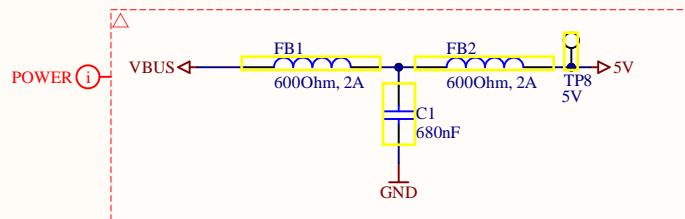
B

C

D

Fichier : Ticketing_Slave.SchDoc	a
Date : 25.09.2022	b
Heure : 10:47:59	c
Auteur : Mario Dos Santos	d
ENL-ES Avenue Recordon 1 1004 Lausanne Switzerland	No : C
Projet : 18112C_Ticketing.PrjPcb	Nb feuillets 5
Chemin : K:\ES\PROJETS\SLO\1811_Ticketing\hard\1811C\18112C_Ticketing_SSOP\Ticketing_Slave.SchDoc	Feuille n° 1

Supply



A

A

B

B

C

C

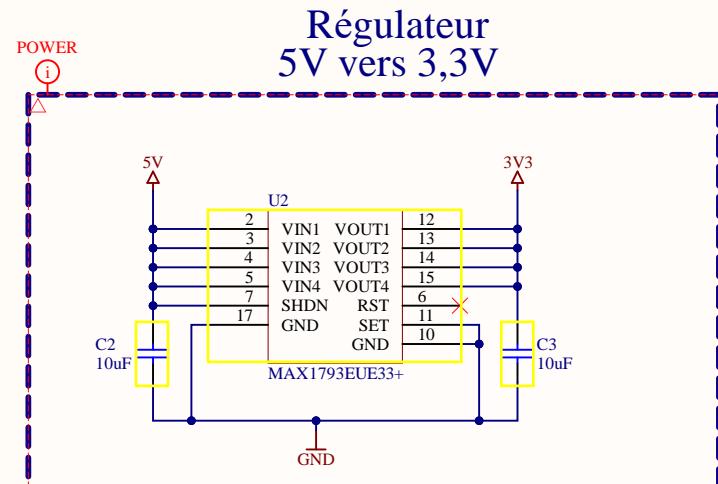
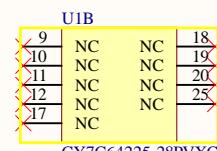
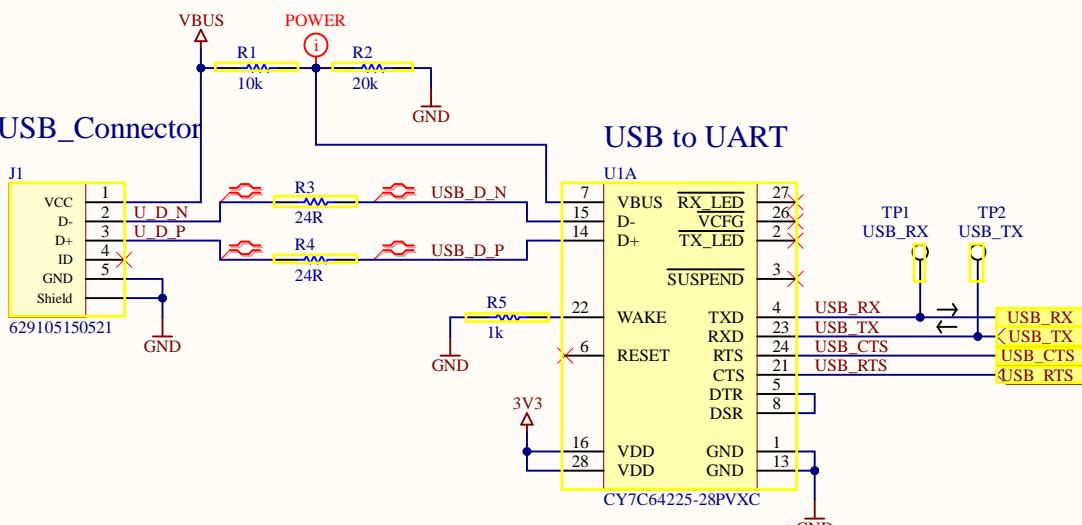
D

D

D

D

Micro_USB_Connector



Fichier : Alim_Ticketing_Slave.SchDoc

Date : 25.09.2022

Heure : 10:47:59

Modif. a

Version :

b

c

d

ETNL-ES
 Avenue Recordon 1
 1004 Lausanne
 Switzerland

Projet : 18112C_Ticketing.PsjPcb

No :

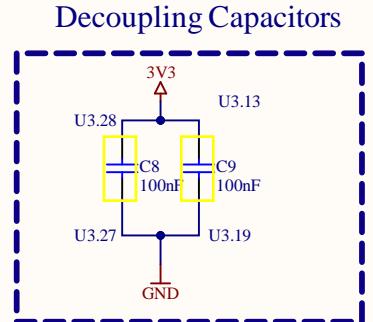
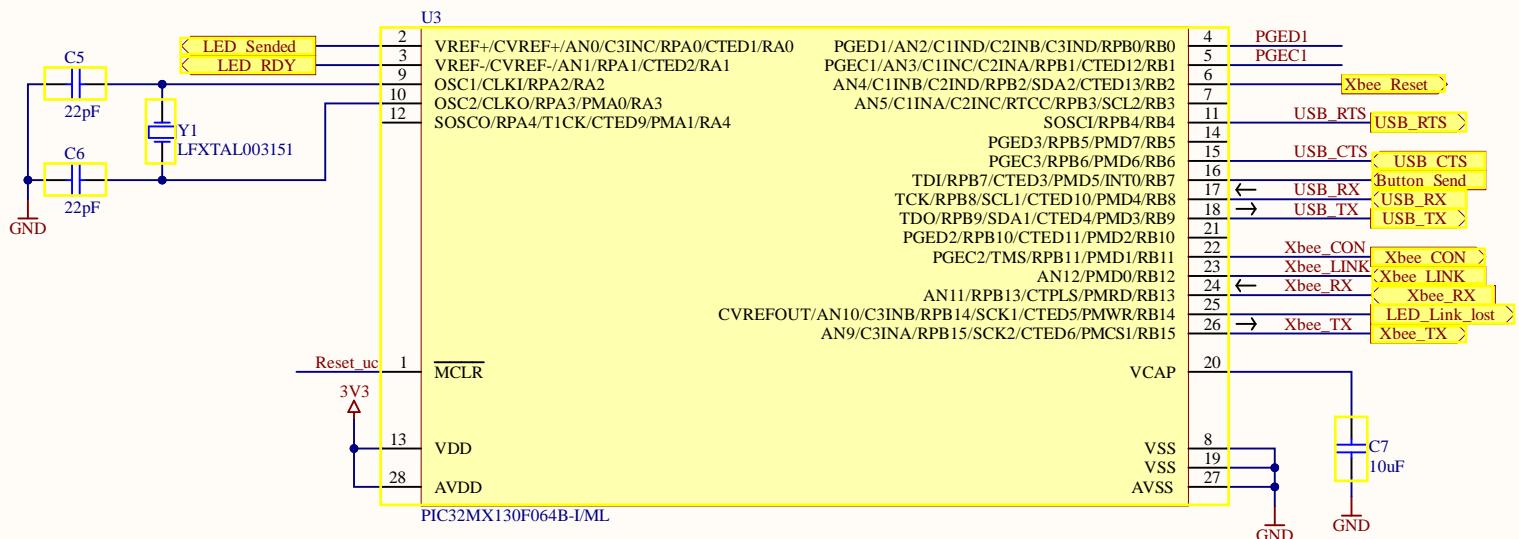
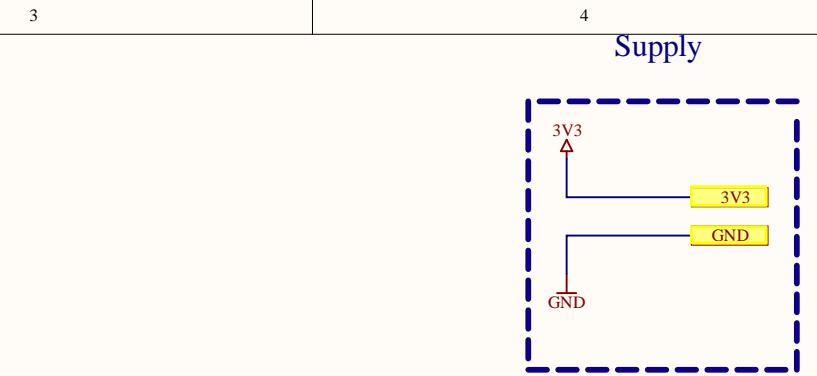
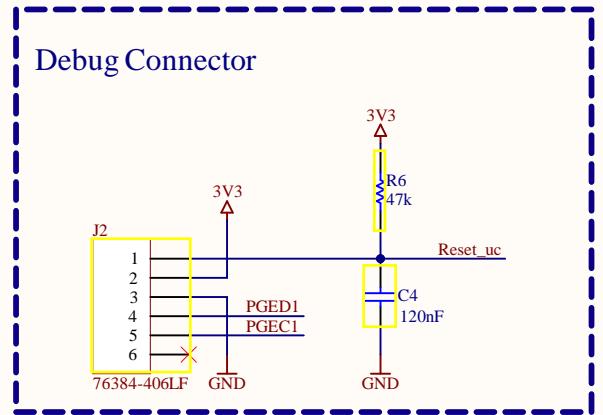
Nb feuillets

5

Feuille n°

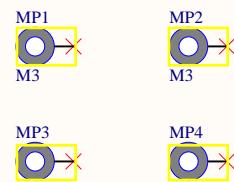
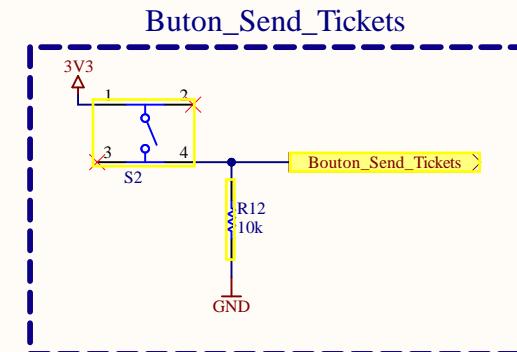
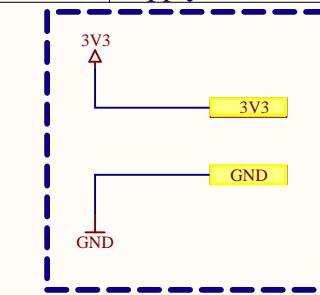
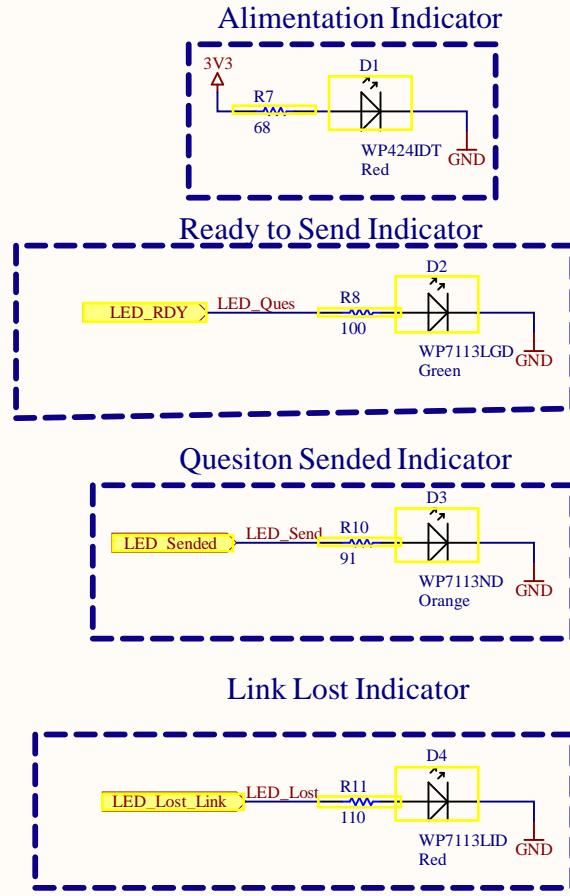
2

Chemin : K:\ES\PROJETS\SLO\1811_Ticketing\hard\1811C\18112C_Ticketing_SSOP\Alim_Ticketing_Slave.SchDoc



Fichier : Uc_Ticketing_Slave.SchDoc	a
Date : 25.09.2022	b
Heure : 10:47:59	c

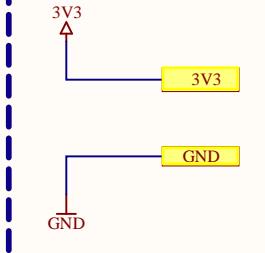
Modif.
d



Fichier : UI_Ticketing_Slave.SchDoc	a
Date : 25.09.2022	b
Heure : 10:47:59	c
Auteur : Mario Dos Santos	d
ENL-ES Avenue Recordon 1 1004 Lausanne Switzerland	No : Nb feuillets 5 Feuille n° 4
Projet : 18112C_Ticketing.PrjPcb	
Chemin : K:\ES\PROJETS\SLO\1811_Ticketing\hard\1811C\18112C_Ticketing_SSOP\UI_Ticketing_Slave.SchDoc	

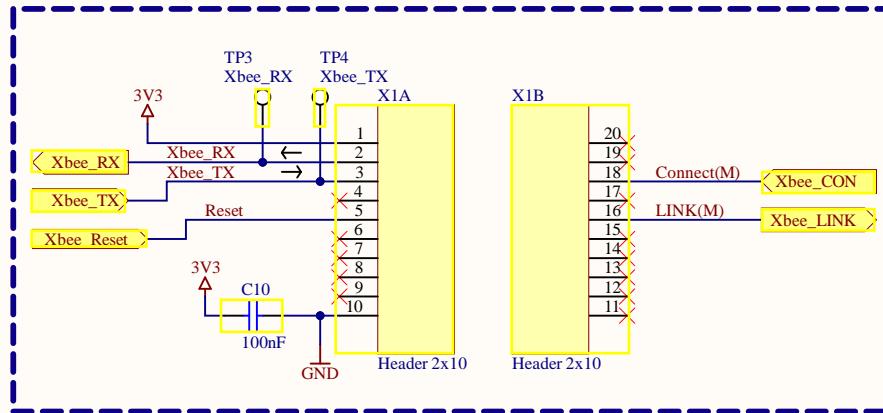
A

Supply



B

Xbee



C

D

Fichier : RF_Ticketing_Slave.SchDoc	Modif.	a
Date : 25.09.2022		b
Heure : 10:47:59		c
Auteur : Mario Dos Santos		d
ENL-ES Avenue Recordon 1 1004 Lausanne Switzerland	Projet : 18112C_Ticketing.PrjPcb	
Chemin : K:\ES\PROJETS\SLO\1811_Ticketing\hard\1811C\18112C_Ticketing_SSOP\RF_Ticketing_Slave.SchDoc		No :
Nb feuillets	5	Feuille n°
	5	5

A

B

C

D

BOM 18111C Ticketing

Designator	Quantity	Comment	Supplier	Supplier Part Number	Supplier unit price	price
C1	1	680nF	Digi-Key	732-7663-1-ND	0,13	0,13
C2, C3, C7	3	10uF	Digi-Key	1276-1298-1-ND	0,1	0,3
C4	1	120nF	Digi-Key	311-4323-1-ND	0,18	0,18
C5, C6	2	22pF	Digi-Key	399-C0805C220J5GAC78 00CT-ND	0,1	0,2
C8, C9, C10, C12	4	100nF	Digi-Key	399-C0805C104M5RAC7 800CT-ND	0,1	0,4
C11	1	1uF	Digi-Key	1276-2931-1-ND	0,1	0,1
D1	1	WP424IDT	Digi-Key	754-1305-ND	0,4	0,4
D2	1	WP7113SEC	Digi-Key	754-1271-ND	0,4	0,4
D3	1	SS14-E3/61T	Digi-Key	SS14-E3/61TGICT-ND	0,48	0,48
FB1, FB2	2	600Ohm, 2A	Digi-Key	732-1620-1-ND	0,22	0,44
J1	1	629105150521	Digi-key	732-5960-1-ND	2,12	2,12
J2	1	76384-406LF	Digi-Key	609-1321-ND	0,66	0,66
J3	1	693071010811	Digi-Key	732-3819-1-ND	2,99	2,99
MP1, MP2, MP3, MP4	4	M3	Digi-Key			
Q1	1	IRLML2402TRPBF	Digi-key	IRLML2402PBFC-ND	0,41	0,41
R1, R8, R9, R12, R13, R15, R16, R17, R19, R20, R21, R22, R23, R24	14	10k	Digi-Key	311-10KLGCT-ND	0,1	1,4
R2	1	20k	Digi-Key	RNCP0805FTD20K0C T-ND	0,1	0,1
R3, R4	2	24R	Digi-Key	311-24.0CRCT-ND	0,1	0,2
R5	1	1k	Digi-Key	RNCP0805FTD1K00C T-ND	0,1	0,1
R6	1	47k	Digi-Key	A130144CT-ND	0,1	0,1
R7	1	68	Digi-Key	RMCF0805JT68R0CT- ND	0,1	0,1
R10	1	91	Digi-Key	RMCF0805JT68R0C T-ND	0,1	0,1
R11	1	120	Digi-Key	A129738CT-ND	0,1	0,1
R14	1	1k	Digi-Key	YAG3703CT-ND	0,1	0,1
R18	1	100k	Digi-Key	YAG3688CT-ND	0,1	0,1
R25	1	1M	Digi-Key	YAG3704CT-ND	0,1	0,1
S1, S2, S3, S4	4	430152095826	Digi-Key	732-6999-1-ND	0,51	2,04
S5	1	430182050816	Digi-Key	732-7005-1-ND	0,51	0,51
S6, S7, S8	3	430466073726	Digi-Key	430466073726-ND	0,56	1,68
TP1, TP2, TP3, TP4, TP5, TP6	6		Digi-Key	36-5019CT-ND	0,33	1,98
TP7, TP8, TP9	3		Digi-Key	36-5011-ND	0,41	1,23

BOM 18111C Ticketing

Designator	Quantity	Comment	Supplier	Supplier Part Number	Supplier unit price	price
U1	1	CY7C64225-28PVXC	Digi-key	428-3158-ND	3,83	3,83
U2	1	MAX1793EUE33+	Digi-Key	MAX1793EUE33+-ND	3,92	3,92
U3	1	PIC32MX170F256D	Digi-Key	PIC32MX170F256D-I/PT-ND	5,43	5,43
U4	1	PS1240P02BT	Digi-Key	445-2525-1-ND	0,6	0,6
VR1	1	0-10k	Digi-Key	3362P-103LF-ND	0,99	0,99
X1, X2	2	HEADER-1X20ROUND	Digi-Key			
X3	1	Header 2x10	Digi-Key			0
Y1	1	LFXTAL003151	Farnell	9713794	0,392	0,392
				total:	34,31	

BOM 18112C_Ticketing

Designator	Quantity	Comment	Supplier	Supplier Part Number	Supplier unit price	price
C1	1	680nF	Digi-Key	732-7663-1-ND	0,13	0,13
C2, C3, C7	3	10uF	Digi-Key	1276-1298-1-ND	0,10	0,3
C4	1	120nF	Digi-Key	311-4323-1-ND	0,18	0,18
C5, C6	2	22pF	Digi-Key	399-C0805C220J5GAC7 800CT-ND	0,10	0,2
C8, C9, C10	3	100nF	Digi-Key	399-C0805C104M5RAC7 800CT-ND	0,10	0,3
D1	1	WP424IDT	Digi-Key	754-1305-ND	0,40	0,4
D2	1	WP7113LGD	Digi-Key	754-1265-ND	0,33	0,33
D3	1	WP7113ND	Digi-Key	754-1896-ND	0,30	0,3
D4	1	WP7113LID	Digi-Key	754-1266-ND	0,33	0,33
FB1, FB2	2	600Ohm, 2A	Digi-Key	732-1620-1-ND	0,22	0,44
J1	1	629105150521	Digi-key	732-5960-1-ND	2,12	2,12
J2	1	76384-406LF	Digi-Key	609-1321-ND	0,66	0,66
MP1, MP2, MP3, MP4	4	M3				
R1, R12	2	10k	Digi-Key	311-10KLGCT-ND	0,10	0,2
R2	1	20k	Digi-Key	RNCPO805FTD20KO CT-ND	0,10	0,1
R3, R4	2	24R	Digi-Key	311-24.0CRCT-ND	0,10	0,2
R5	1	1k	Digi-Key	RNCPO805FTD1K00 CT-ND	0,10	0,1
R6	1	47k	Digi-Key	A130144CT-ND	0,10	0,1
R7	1	68	Digi-Key	RMCF0805JT68ROC T-ND	0,10	0,1
R8	1	100	Digi-Key	RMCF0805FT100RC T-ND	0,10	0,1
R9	1	120	Digi-Key	A129738CT-ND	0,10	0,1
R10	1	91	Digi-Key	311-91ARCT-ND	0,10	0,1
R11	1	110	Digi-Key	A126333CT-ND	0,10	0,1
S1	1	430182050816	Digi-Key	732-7005-1-ND	0,51	0,51
S2	1	430466073726	Digi-Key	430466073726-ND	0,56	0,56
TP1, TP2, TP3, TP4, TP5, TP8	6		Digi-Key	36-5019CT-ND	0,33	1,98
TP6, TP7	2		Digi-key	36-5011-ND	0,41	0,82
U1	1	CY7C64225- 28PVXC	Digi-key	428-3158-ND	3,83	3,83
U2	1	MAX1793EUE33+	Digi-Key	MAX1793EUE33+- ND	3,92	3,92
U3	1	PIC32MX130F064 B-I/ML	Digi-Key	PIC32MX130F064B- I/ML	3,61	3,61
X1	1	Header 2x10				0
Y1	1	LFXTAL003151	Farnell	9713794	0,39	0,392
				total:		22,512

Composants_matériel supplémentaire 1811C Ticketing

Composants/matériel supplémentaire					
Description	Quantity	Supplier	Supplier Part Number	Supplier unit price	Price
Xbee	3	ETML	1623	0	0,00
Xbee	3	Digi-key	602-1892-ND	30,84	92,52
Ecran TFT	1	Digi-key	1528-2848-ND	38,35	38,35
Capuchon vert	1	Digi-key	714308050-ND	0,29	0,29
Capuchon Noir	1	Digi-key	714301050-ND	0,24	0,24
Capuchon rouge	3	Digi-key	714306050-ND	0,24	0,72
Pin socket 1x10	6	Digi-key	732-62001011821-ND	1,17	7,02
pin header 1x5	3	Digi-key	732-5317-ND	0,18	0,54
PCB Master	1	Eurocircuit	18111C	68,08	68,08
PCB Slave*	1	Eurocircuit	18112C	67,89	67,89
			Total:		183,13

* Les 2 PCB sont commander en même temps pour que sa soit moins cher

Choisir l'un des 2

1811C Ticketing

Matériel Mécanique					
Description	Quantity	Supplier	Supplier Part Number	Supplier unit price	Price
Boitier Master	1	Digi-key	164-1555HL2GY-ND	23,39	23,39
Boitier Slave	2	Digi-key	HM3986-ND	5,68	11,36
Total:					34,75

```

1 //18111C_Master
2 //-----
3 // App.c
4 //-----
5 //
6 //
7 // Auteur      :
8 // Date       :
9 // Version     :
10 // Modifications : MDS 26.09.2022
11 // Description :
12 //           Application principal de la carte ticketing Master 1811C
13 //
14 //
15 /*-----*/
16
17
18 #include "app.h"
19 #include "Data_Code.h"
20 #include "Mc32gest_RS232.h"
21 #include "Gest_Menu.h"
22
23
24 APP_DATA appData;
25 APP_DATA appData_Old;
26 APP_DATA appData_callback;
27
28
29 uint8_t Nb_Student = 0; //eleves actuellement en cours de traitement
30 uint8_t Nb_Student_max = 0; //nombre max de slave/eleve actuellement enregistrer
31 bool Message_receive_XBEE; // flag de reception de donnee xbee
32
33 ****
34 Function:
35     void APP_Initialize ( void )
36
37 Remarks:
38     See prototype in app.h.
39 */
40
41 void APP_Initialize ( void )
42 {
43     appData.state = APP_STATE_INIT;
44 }
45 ****
46 Function:
47     void APP_UpdateState ( void )
48
49 Remarks:
50     See prototype in app.h.
51 */
52
53 void APP_UpdateState (APP_STATES NewState)
54 {
55     appData.state = NewState;
56 }
57
58 ****
59 Function:
60     void APP_Tasks ( void )
61
62 Remarks:
63     See prototype in app.h.
64 */
65
66 void APP_Tasks ( void )
67 {
68     static int Count = 0,I,J;
69     int32_t RXSize;
70     char trash;
71     static uint32_t DataCodeToSend = 0;
72     static bool Ticket_Refused = false;
73     U_32 RXData;

```

```

74     U_32 ADD_M;
75     U_32 ADD_S;
76     bool Waiting_Answer = false;
77
78     /* Check the application's current state. */
79     switch ( appData.state )
80     {
81         /* Application's initial state. */
82         case APP_STATE_INIT:
83         {
84
85             DRV_TMR0_Stop(); // stoper le timer 1
86
87             RF_Init(); // initialisation de l Xbee
88
89             TFT_Init(); // initialisation du TFT
90             Init_Menu(); // initialisation du menu
91             InitFifoComm(); // initialisation du fifo
92
93             DRV_TMR1_Start(); //Start le timer 2
94             APP_UpdateState (APP_STATE_GEST_MENU); //changement d'etat
95
96             break;
97         }
98
99         case APP_STATE_GEST_MENU:
100        {
101            //Led_QuestToggle();
102
103            Gest_Menu(); //gestion du menu
104
105            break;
106        }
107        case APP_STATE_LINK_XBEE:
108        {
109            //AddTarget.val32 = Students_Info[0].ID;
110            //AddTarget.val32 = Add_Master;
111            DataCodeToSend = ARE_U_LINK; //preparation de l'envoie
112
113            //sauvegarde de l'etat precedent
114            appData_callback.state = appData.state;
115            //changement de l'etat
116            APP_UpdateState(APP_STATE_SEND_ID);
117            //sauvegarder l'etat
118            appData_Old.state = APP_STATE_LINK_XBEE;
119
120
121            break;
122        }
123
124        case APP_STATE_SEND_ID:
125        {
126            if(appData_Old.state == APP_STATE_LINK_XBEE)
127            {
128                //flag start Link
129                LINK = false;
130                //flag adresse slave connu
131                Add_ON = false;
132                //envoie du message
133                SendMessage(Add_Master,NULL,DataCodeToSend);
134                //flag adresse slave connu
135                Add_ON = true;
136                //Changement d etat
137                APP_UpdateState(APP_STATE_GET_DATA);
138            }
139            else if(appData_Old.state == APP_STATE_PRESENCE)
140            {
141                //envoie du message
142                SendMessage(Add_Master,Add_Slave,DataCodeToSend);
143                //Changement d etat
144                APP_UpdateState(APP_STATE_GET_DATA);
145                //incrementation de la list
146                Nb_In_List++;

```

```

147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
}
else
{
    //envoie du message
    SendMessage (Add_Master,Add_Slave,DataCodeToSend) ;
}

break;
}
case APP_STATE_GET_DATA:
{
    //reception du message et de l expediteur
    GetMessage (&ADD_S,&ADD_M,&RXData);

    //on check que l'expediteur est bien le maître
    if((ADD_M.val32 == Add_Master)|| (Waiting_Answer))
    {
        //on check si l'on recois un ticket ou si on attend une
        //reponse de l'utilisateur
        if ((RXData.val32 == ENVOI_TICKET) || (Waiting_Answer))
        {
            //start du timer 1
            DRV_TMR0_Start();
            //on met un flag d'attente de reponse
            Waiting_Answer = true ;
            //si le bounton Acc a ete appuyer
            if(appButtons.But_ACC)
            {
                //on prepare le data a envoyer
                DataCodeToSend = TICKET_ACCEPT;
                //Changement d'etat
                APP_UpdateState(APP_STATE_SEND_DATA);
                //on enleve le flag
                Waiting_Answer = false ;
                //on arrete le timer 1
                DRV_TMR0_Stop();
            }
            //si le bounton Acc a ete appuyer
            else if(appButtons.But_DECL)
            {
                //on prepare le data a envoyer
                DataCodeToSend= TICKET_REFUSE;
                //Changement d'etat
                APP_UpdateState(APP_STATE_SEND_DATA);
                //on enleve le flag
                Waiting_Answer = false ;
                //on arrete le timer 1
                DRV_TMR0_Stop();
            }
            else if(appButtons.But_ESC)
            {
                DataCodeToSend= TICKET_RESET;
                APP_UpdateState(APP_STATE_SEND_DATA);
                Waiting_Answer = false ;
                DRV_TMR0_Stop();
            }
        }
        // si l'on recois une adresse d'un nouveau slave
        if((New_student)&&(!Waiting_Answer))
        {
            //on enleve le falg de nouveau slave
            New_student = false;
            //on prepare le data a envoyer
            DataCodeToSend= ACK;
            //Changement d'etat
            APP_UpdateState(APP_STATE_SEND_DATA);
        }
    }
    break;
}
case APP_STATE_SEND_DATA:
{

```

```

220
221 //envoie de message au Xbee
222 SendMessage(Add_Master,Add_Slave,DataCodeToSend) ;
223
224 if(appData_Old.state == APP_STATE_LINK_XBEE)
{
    //on retourne au debut de la list
    Nb_In_List = 0;
    //Changement d'etat
    APP_UpdateState(APP_STATE_PRESENCE);
}
else if (appData_Old.state == APP_STATE_PRESENCE)
{
    //Changement d'etat
    APP_UpdateState(APP_STATE_PRESENCE);
    //on incremente la list
    Nb_In_List++;
}
else
{
    //Changement d'etat
    APP_UpdateState(APP_STATE_GET_DATA);
}
break;
}
case APP_STATE_RESET:
{
    //TODO
    break;
}
case APP_STATE_REFUSE:
{
    //Changement d'etat
    APP_UpdateState(APP_STATE_SEND_ID);
    //on prepare le data a envoyer
    DataCodeToSend = TICKET_REFUSE;
    //on prepare l'adress a envoyer
    Add_Slave = Students_Info[Nb_In_List].ID;
    break;
}
case APP_STATE_ACCEPT:
{
    //Changement d'etat
    APP_UpdateState(APP_STATE_SEND_ID);
    //on prepare le data a envoyer
    DataCodeToSend = TICKET_ACCEPT;
    //on prepare l'adress a envoyer
    Add_Slave = Students_Info[Nb_In_List].ID;
    //on reduit le nombre max de question
    Students_Info[Nb_In_List].LimitQues--;
    break;
}
case APP_STATE_PRESENCE:
{
    //on prepare le data a envoyer
    DataCodeToSend = ARE_U_LINK;
    //on prepare l'adress a envoyer
    Add_Slave = Students_Info[Nb_In_List].ID;
    //si on arrive a la fin de la list
    if(Nb_In_List == Nb_Student_max)
    {
        //Changement d'etat
        APP_UpdateState(APP_STATE_SEND_DATA);
        //on retourne dans l etat avant le pulling
        appData_Old.state == appData_callback.state;
    }
    else
    {
        //Changement d'etat
        appData_Old.state == APP_STATE_PRESENCE;
        //incrémentation de la list
        Nb_In_List++;
    }
}

```

```
293         break;
294     }
295     default:
296     {
297         break;
298     }
299 }
300
301
302
303
304 /***** End of File *****/
305
306 */
307
```

```

1 ****
2 MPLAB Harmony Application Header File
3
4 Company:
5 Microchip Technology Inc.
6
7 File Name:
8 app.h
9
10 Summary:
11 This header file provides prototypes and definitions for the application.
12
13 Description:
14 This header file provides function prototypes and data type definitions for
15 the application. Some of these are required by the system (such as the
16 "APP_Initialize" and "APP_Tasks" prototypes) and some of them are only used
17 internally by the application (such as the "APP_STATES" definition). Both
18 are defined here for convenience.
19 ****
20
21 //DOM-IGNORE-BEGIN
22 ****
23 Copyright (c) 2013-2014 released Microchip Technology Inc. All rights reserved.
24
25 Microchip licenses to you the right to use, modify, copy and distribute
26 Software only when embedded on a Microchip microcontroller or digital signal
27 controller that is integrated into your product or third party product
28 (pursuant to the sublicense terms in the accompanying license agreement).
29
30 You should refer to the license agreement accompanying this Software for
31 additional information regarding your rights and obligations.
32
33 SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND,
34 EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF
35 MERCHANTABILITY, TITLE, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE.
36 IN NO EVENT SHALL MICROCHIP OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER
37 CONTRACT, NEGLIGENCE, STRICT LIABILITY, CONTRIBUTION, BREACH OF WARRANTY, OR
38 OTHER LEGAL EQUITABLE THEORY ANY DIRECT OR INDIRECT DAMAGES OR EXPENSES
39 INCLUDING BUT NOT LIMITED TO ANY INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE OR
40 CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT OF
41 SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES
42 (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS.
43 ****
44 //DOM-IGNORE-END
45
46 #ifndef _APP_H
47 #define _APP_H
48
49 // ****
50 // ****
51 // Section: Included Files
52 // ****
53 // ****
54 #include <stdint.h>
55 #include <stdbool.h>
56 #include <stddef.h>
57 #include <stdlib.h>
58 #include <stdio.h>
59 #include "system_config.h"
60 #include "system_definitions.h"
61
62
63 // DOM-IGNORE-BEGIN
64 #ifdef __cplusplus // Provide C++ Compatibility
65
66 extern "C" {
67
68 #endif
69 // DOM-IGNORE-END
70
71 // ****
72 // ****
73 // Section: Type Definitions

```

```

74 // ****
75 // ****
76 // ****
77
78
79
80 /* Application states
81
82 Summary:
83     Application states enumeration
84
85 Description:
86     This enumeration defines the valid application states. These states
87     determine the behavior of the application at various times.
88 */
89
90 typedef enum
91 {
92     /* Application's state machine's initial state. */
93     APP_STATE_INIT=0,
94     APP_STATE_LINK_XBEE,
95     APP_STATE_SEND_ID,
96     APP_STATE_SEND_DATA,
97     APP_STATE_RESET,
98     APP_STATE_REFUSE,
99     APP_STATE_ACCEPT,
100    APP_STATE_PRESENCE,
101    APP_STATE_GEST_MENU,
102    APP_STATE_GET_DATA,
103
104    /* TODO: Define states used by the application state machine. */
105
106 } APP_STATES;
107
108
109 // ****
110 /* Application Data
111
112 Summary:
113     Holds application data
114
115 Description:
116     This structure holds the application's data.
117
118 Remarks:
119     Application strings and buffers are be defined outside this structure.
120 */
121
122 typedef struct
123 {
124     /* The application's current state */
125     APP_STATES state;
126
127     /* TODO: Define any additional data used by the application. */
128
129 } APP_DATA;
130
131
132 extern char buffReadName[]; // Buffer de reception de l'UART
133 extern uint8_t Name_Receive; // Flag de réception
134 extern uint8_t countCar; // Compteur du nombre de characters d'un nom
135 extern uint8_t Nb_Student;
136 extern uint8_t Nb_Student_max;
137 bool flag_Com_Received;
138
139 // ****
140 // ****
141 // Section: Application Callback Routines
142 // ****
143 // ****
144 /* These routines are called by drivers when certain events occur.
145 */
146

```

```

147 // ****
148 // ****
149 // Section: Application Initialization and State Machine Functions
150 // ****
151 // ****
152 // ****
153 // ****
154     Function:
155         void APP_Initialize ( void )
156
157     Summary:
158         MPLAB Harmony application initialization routine.
159
160     Description:
161         This function initializes the Harmony application. It places the
162         application in its initial state and prepares it to run so that its
163         APP_Tasks function can be called.
164
165     Precondition:
166         All other system initialization routines should be called before calling
167         this routine (in "SYS_Initialize").
168
169     Parameters:
170         None.
171
172     Returns:
173         None.
174
175     Example:
176         <code>
177             APP_Initialize();
178         </code>
179
180     Remarks:
181         This routine must be called from the SYS_Initialize function.
182 */
183
184 void APP_Initialize ( void );
185
186
187 // ****
188     Function:
189         void APP_Tasks ( void )
190
191     Summary:
192         MPLAB Harmony Demo application tasks function
193
194     Description:
195         This routine is the Harmony Demo application's tasks function. It
196         defines the application's state machine and core logic.
197
198     Precondition:
199         The system and application initialization ("SYS_Initialize") should be
200         called before calling this.
201
202     Parameters:
203         None.
204
205     Returns:
206         None.
207
208     Example:
209         <code>
210             APP_Tasks();
211         </code>
212
213     Remarks:
214         This routine must be called from SYS_Tasks() routine.
215 */
216
217 void APP_Tasks( void );
218
219 void APP_UpdateState(APP_STATES NewState);

```

```
220 #endif /* _APP_H */
221
222 //DOM-IGNORE-BEGIN
223 #ifdef __cplusplus
224 }
225#endif
226 //DOM-IGNORE-END
227
228 /***** End of File ****/
229 */
230 */
231 */
232 */
```

```

1 //18111C_Master
2 //-----
3 // Gest_TFT.c
4 //-----
5 //
6 //
7 // Auteur      :
8 // Date       :
9 // Version    :
10 // Modifications : MDS 26.09.2022
11 // Description  :
12 //      Gestion du TFT de la carte Ticketing Master 1811C
13 //
14 /*-----*/
15
16 #include "app.h"
17 #include "Gest_TFT.h"
18 #include "DefineLCD.h"
19 #include "Gest_Menu.h"
20
21
22 #include <sys/attribs.h>
23 #include "system_definitions.h"
24 #include <stdio.h>
25 #include <stdlib.h>
26
27 void TFT_Init (void)
28 {
29
30     tft_begin();
31     setRotation(1);
32     tft_fillScreen(ILI9341_BLACK);
33     TFT_Font_debug ();
34
35 }
36
37
38
39
40 void TFT_Menu_Principal (void)
41 {
42     //tft_fillScreen(ILI9341_BLACK);
43     //tft.fillRect(10,10,300,220,0x221F);
44     //tft_setTextSize(2);
45     tft_setTextSize(1);
46     tft_setCursor(15,15);
47     tft_setTextColor(ILI9341_WHITE);
48     tft_writeString("Ticketing Martins");
49     tft_drawFastHLine(10,31,300,ILI9341_WHITE);
50     tft_drawFastHLine(10,32,300,ILI9341_WHITE);
51     //tft_setTextSize(3);
52     tft_setTextSize(1);
53     tft_setCursor(90,50);
54     tft_writeString("Liste");
55     tft_setCursor(90,100);
56     tft_writeString("Config");
57 }
58
59 void TFT_Font_debug (void)
60 {
61
62     /*tft_fillScreen(ILI9341_WHITE);
63     tft_drawFastHLine(23,30,10,ILI9341_BLUE);
64     tft_drawFastHLine(19,31,18,ILI9341_BLUE);
65     tft_drawFastHLine(17,32,22,ILI9341_BLUE);
66     tft_drawFastHLine(15,33,26,ILI9341_BLUE);
67     tft_drawFastHLine(14,34,28,ILI9341_BLUE);
68     tft_drawFastHLine(12,35,32,ILI9341_BLUE);
69     tft_drawFastHLine(11,36,34,ILI9341_BLUE);
70     tft_drawFastHLine(10,37,36,ILI9341_BLUE);
71     tft_drawFastHLine(9,38,38,ILI9341_BLUE);
72     tft_drawFastHLine(8,39,40,ILI9341_BLUE);
73     tft_drawFastHLine(8,40,40,ILI9341_BLUE);
```

```

74 tft_drawFastHLine(7,41,42,ILI9341_BLUE);
75 tft_drawFastHLine(7,42,42,ILI9341_BLUE);
76 tft_drawFastHLine(6,43,44,ILI9341_BLUE);
77 tft_drawFastHLine(6,44,44,ILI9341_BLUE);
78 tft_drawFastHLine(6,45,44,ILI9341_BLUE);
79 tft_drawFastHLine(6,46,44,ILI9341_BLUE);
80 tft_drawFastHLine(5,47,46,ILI9341_BLUE);
81 tft_drawFastHLine(5,48,46,ILI9341_BLUE);
82 tft_drawFastHLine(5,49,46,ILI9341_BLUE);
83 tft_drawFastHLine(5,50,46,ILI9341_BLUE);
84 tft_drawFastHLine(5,51,46,ILI9341_BLUE);
85 tft_drawFastHLine(5,52,46,ILI9341_BLUE);
86 tft_drawFastHLine(5,53,46,ILI9341_BLUE);
87 tft_drawFastHLine(5,54,46,ILI9341_BLUE);
88 tft_drawFastHLine(5,55,46,ILI9341_BLUE);
89 tft_drawFastHLine(5,56,46,ILI9341_BLUE);
90 tft_drawFastHLine(6,57,44,ILI9341_BLUE);
91 tft_drawFastHLine(6,58,44,ILI9341_BLUE);
92 tft_drawFastHLine(6,59,44,ILI9341_BLUE);
93 tft_drawFastHLine(6,60,44,ILI9341_BLUE);
94 tft_drawFastHLine(7,61,42,ILI9341_BLUE);
95 tft_drawFastHLine(7,62,42,ILI9341_BLUE);
96 tft_drawFastHLine(8,63,40,ILI9341_BLUE);
97 tft_drawFastHLine(8,64,40,ILI9341_BLUE);
98 tft_drawFastHLine(9,65,38,ILI9341_BLUE);
99 tft_drawFastHLine(10,66,36,ILI9341_BLUE);
100 tft_drawFastHLine(11,67,34,ILI9341_BLUE);
101 tft_drawFastHLine(12,68,32,ILI9341_BLUE);
102 tft_drawFastHLine(14,69,28,ILI9341_BLUE);
103 tft_drawFastHLine(15,70,26,ILI9341_BLUE);
104 tft_drawFastHLine(17,71,22,ILI9341_BLUE);
105 tft_drawFastHLine(19,72,18,ILI9341_BLUE);
106 tft_drawFastHLine(23,73,10,ILI9341_BLUE);
107 tft_setTextSize(3);
108 tft.setCursor(12,42);
109 tft_setTextColor(ILI9341_WHITE);
110 tft_writeString("Es");
111
112 tft_drawFastHLine(48,25,5,ILI9341_BLUE);
113 tft_drawFastHLine(46,26,9,ILI9341_BLUE);
114 tft_drawFastHLine(45,27,11,ILI9341_BLUE);
115 tft_drawFastHLine(44,28,13,ILI9341_BLUE);
116 tft_drawFastHLine(44,29,13,ILI9341_BLUE);
117 tft_drawFastHLine(43,30,15,ILI9341_BLUE);
118 tft_drawFastHLine(43,31,15,ILI9341_BLUE);
119 tft_drawFastHLine(43,32,15,ILI9341_BLUE);
120 tft_drawFastHLine(43,33,15,ILI9341_BLUE);
121 tft_drawFastHLine(43,34,15,ILI9341_BLUE);
122 tft_drawFastHLine(44,35,13,ILI9341_BLUE);
123 tft_drawFastHLine(44,36,13,ILI9341_BLUE);
124 tft_drawFastHLine(45,37,11,ILI9341_BLUE);
125 tft_drawFastHLine(46,38,9,ILI9341_BLUE);
126 tft_drawFastHLine(48,39,5,ILI9341_BLUE);
127 tft_setTextSize(2);
128 tft.setCursor(46,26);
129 tft_setTextColor(ILI9341_WHITE);
130 tft_writeString("+");
131 tft_setTextSize(1);
132 tft.setCursor(18,75);
133 tft_setTextColor(ILI9341_BLUE);
134 tft_writeString("ETML");
135
136 tft_setTextSize(4);
137 tft.setCursor(20,120);
138 tft_setTextColor(ILI9341_BLACK);
139 tft_writeString("Bienvenue !");*/
140
141
142
143
144 //Si tft_setTextSize n'est pas 1 affiche rien
145 //tft_fillRect ne fonctionne pas
146 //tft_fillScreen ne fonctionne pas;

```

```

147
148
149
150     tft_fillRect(10,10,300,220,ILI9341_WHITE);
151 // tft_setTextSize(2);
152 tft_setTextSize(1);
153 tft_setTextColor(ILI9341_WHITE);
154 tft_setCursor(10,10);
155 tft_writeString("Initializing");
156 tft_drawFastHLine(10,31,300,ILI9341_WHITE);
157 tft_drawFastHLine(10,32,300,ILI9341_WHITE);
158 // tft_setTextSize(1);
159
160
161 // tft_setCursor(10,50);
162 // tft_writeString("SD Is Plugged:");
163 //
164 // tft_setCursor(10,70);
165 // tft_writeString("SD Mount:");
166 //
167 // tft_setCursor(10,90);
168 // tft_writeString("Student file found:");
169 //
170 // tft_setCursor(10,110);
171 // tft_writeString("Getting list of students:");
172 //
173 // tft_setCursor(10,130);
174 // tft_writeString("Students found:");
175 //
176 // tft_setCursor(10,50);
177 // tft_writeString("checking link:");
178 //
179 // tft_setCursor(10,70);
180 // tft_writeString("Students link:");
181
182
183 }
184
185 void TFT_SD_PLUG_OK (void)
186 {
187     tft_setTextSize(1);
188     tft_setTextColor(ILI9341_WHITE);
189     tft_setCursor(95,50);
190     tft_writeString("OK");
191 }
192
193 void TFT_SD_Mount_OK (void)
194 {
195     tft_setTextSize(1);
196     tft_setTextColor(ILI9341_WHITE);
197     tft_setCursor(65,70);
198     tft_writeString("OK");
199 }
200
201 void TFT_Student_File_Detected (void)
202 {
203     tft_setTextSize(1);
204     tft_setTextColor(ILI9341_WHITE);
205     tft_setCursor(125,90);
206     tft_writeString("OK");
207 }
208
209
210 void TFT_List_Of_Students_OK (void)
211 {
212     tft_setTextSize(1);
213     tft_setTextColor(ILI9341_WHITE);
214     tft_setCursor(160,110);
215     tft_writeString("OK");
216 }
217
218 void TFT_Students_Found (char T_Nb_Students [3])
219 {

```

```

220     tft_setTextSize(1);
221     tft_setTextColor(ILI9341_WHITE);
222     tft.setCursor(100,130);
223     tft_writeString(T_Nb_Students);
224 }
225
226 void TFT_Chekking_Link (void)
227 {
228     tft_setTextSize(1);
229     tft_setTextColor(ILI9341_WHITE);
230     tft.setCursor(95,150);
231     tft_writeString("OK");
232 }
233
234
235 void TFT_Students_Link (int Nb_Link)
236 {
237     char T_Nb_Students [3];
238     sprintf(T_Nb_Students,"%d",Nb_Link);
239     tft_setTextSize(1);
240     tft_setTextColor(ILI9341_WHITE);
241     tft.setCursor(95,170);
242     tft_writeString(T_Nb_Students);
243 }
244
245 void TFT_Init_OK (void)
246 {
247     //tft_setTextSize(2);
248     tft_setTextSize(1);
249     tft_setTextColor(ILI9341_WHITE);
250     tft.setCursor(10,190);
251     tft_writeString("OK tap green");
252     tft.setCursor(10,220);
253     tft_writeString("Re_Init tap Red");
254 }
255
256
257 void TFT_Aff_List (int I, char StudentName [15])
258 {
259     //tft_setTextSize(2);
260     tft_setTextSize(1);
261     tft_setTextColor(ILI9341_WHITE);
262     tft.setCursor(30,(35+((I-1)*20)));
263     tft_writeString(StudentName);
264 }
265
266 void TFT_aff_list_more (int I)
267 {
268     char more [9];
269     sprintf(more,"more %d v",I);
270     //tft_setTextSize(2);
271     tft_setTextSize(1);
272     tft_setTextColor(ILI9341_WHITE);
273     tft.setCursor(20,215);
274     tft_writeString(more);
275 }
276
277 void TFT_Cursor_MenuPrincipale (void)
278 {
279     static int LastCursorPos = 0;
280     tft_setTextSize(4);
281     switch (MenuSelected)
282     {
283         case SELECTED_NONE:
284         {
285             if(LastCursorPos != 0)
286             {
287                 //clear L'ancien Curseur
288                 tft_fillRect(40,LastCursorPos,20,28,0x221F);
289             }
290             break;
291         }
292         case SELECTED_LIST:

```

```

293     {
294         if(LastCursorPos != 0)
295         {
296             //clear L'ancien Curseur
297             tft_fillRect(40,LastCursorPos,20,28,0x221F);
298         }
299         tft_setCursor(40,50);
300         tft_writeString(">");
301         LastCursorPos = 50;
302         break;
303     }
304     case SELECTED_Configuration:
305     {
306         if(LastCursorPos != 0)
307         {
308             //clear L'ancien Curseur
309             tft_fillRect(40,LastCursorPos,20,28,0x221F);
310         }
311         tft_setCursor(40,100);
312         tft_writeString(">");
313         LastCursorPos = 100;
314         break;
315     }
316 }
317 }
318
319 void TFT_Font_List(void)
320 {
321     //tft_fillScreen(ILI9341_BLACK);
322     //tft_fillRect(10,10,300,220,0x221F);
323     //tft_setTextSize(2);
324     tft_setTextSize(1);
325     tft_setCursor(15,15);
326     tft_setTextColor(ILI9341_WHITE);
327     tft_writeString("List");
328     tft_drawFastHLine(10,31,300,ILI9341_WHITE);
329     tft_drawFastHLine(10,32,300,ILI9341_WHITE);
330     //tft_setTextSize(2);
331     tft_setTextSize(1);
332     tft_setCursor(15,35);
333     tft_setTextColor(ILI9341_WHITE);
334     tft_writeString(">");
335 }
336
337 void TFT_Font_config(void)
338 {
339     //tft_fillScreen(ILI9341_BLACK);
340     //tft_fillRect(10,10,300,220,0x221F);
341     //tft_setTextSize(2);
342     tft_setTextSize(1);
343     tft_setCursor(15,15);
344     tft_setTextColor(ILI9341_WHITE);
345     tft_writeString("config");
346     tft_drawFastHLine(10,31,300,ILI9341_WHITE);
347     tft_drawFastHLine(10,32,300,ILI9341_WHITE);
348     tft_setTextSize(1);
349     tft_setCursor(15,35);
350     tft_setTextColor(ILI9341_WHITE);
351     tft_writeString(">");
352 }
353
354 void TFT_aff_all_students (void)
355 {
356     int I;
357     tft_setTextSize(1);
358     tft_setCursor(25,35);
359     tft_setTextColor(ILI9341_WHITE);
360     for(I =0; I < I_Nbstudents; I++)
361     {
362         tft_writeString(Students_Info[I].StudentName);
363         tft_setCursor(25,35+((I+1)*10));
364     }
365 }
```

```
366
367 void TFT_Update_Config_cursor (int pos)
368 {
369
370 }
```

```

1 //18111C_Master
2 //-----
3 /* ***** **** * ***** **** * ***** **** * ***** **** * **** */
4 /** Descriptive File Name
5
6     @Company
7         Company Name
8
9     @File Name
10        filename.h
11
12    @Summary
13        Brief description of the file.
14
15    @Description
16        Describe the purpose of this file.
17 */
18 /* ***** **** * ***** **** * ***** **** * ***** **** * **** */
19
20 #ifndef _GEST_TFT_H      /* Guard against multiple inclusion */
21 #define _GEST_TFT_H
22
23
24
25 #ifdef __cplusplus
26 }
27#endif
28
29 void TFT_Init (void);
30 void TFT_Cursor_MenuPrincipale ();
31 void TFT_Font_List();
32 void TFT_Menu_Principal (void);
33 void TFT_Font_debug (void);
34 void TFT_SD_PLUG_OK (void);
35 void TFT_SD_Mount_OK (void);
36 void TFT_List_Of_Students_OK (void);
37 void TFT_Students_Found (char T_Nb_Students [3]);
38 void TFT_Chekking_Link (void);
39 void TFT_Students_Link(int Nb_Link);
40 void TFT_Init_OK(void);
41 void TFT_aff_all_students (void);
42
43     /* Provide C++ Compatibility */
44 #ifdef __cplusplus
45 }
46#endif
47
48#endif /* _EXAMPLE_FILE_NAME_H */
49
50 /* ***** **** * ***** **** * ***** **** * ***** **** * **** */
51 End of File
52 */
53

```



```

73
74         {
75             MenuSelected = SELECTED_LIST;
76         }
77     else
78     {
79         MenuSelected++;
80     }
81     TFT_Cursor_MenuPrincipale ();
82     ClearALLButtons ();
83 }
84 else if (appButtons.But_OK)
85 {
86     if(MenuSelected != SELECTED_NONE)
87     {
88         MenuList = MenuSelected;
89         if(MenuSelected == SELECTED_LIST)
90         {
91             TFT_Font_List ();
92             if(Nb_Students_In_List > 9)
93             {
94                 TFT_aff_list_more (Nb_Students_In_List - 9);
95             }
96         else if(MenuSelected == SELECTED_Configuration)
97         {
98             MenuList = MENU_Config;
99             PosCursorConfig = 0;
100            TFT_Font_config ();
101            TFT_aff_all_students ();
102        }
103    }
104    ClearALLButtons ();
105 }
106 break;
107 }

108 case MENU_List:
109 {
110     if (Nb_Students_In_List > 0)
111     {
112         for(I = 1; I < Nb_Students_In_List; I++)
113         {
114             if(I > 8)
115             {
116                 I = Nb_Students_In_List;
117             }
118             for (J = 0; J < 9; J++)
119             {
120                 if(Students_Info[J].ID == List[I-1])
121                 {
122                     TFT_Aff_List (I, Students_Info[J].StudentName);
123                     J = 10;
124                 }
125             }
126         }
127     }
128     MenuList = Idle;
129     break;
130 }

131 }

132 case Idle:
133 {
134     if(appButtons.But_ESC)
135     {
136         MenuList = MENU_MenuPrincipal;
137         TFT_Cursor_MenuPrincipale ();
138         TFT_Menu_Principal ();
139         ClearALLButtons ();
140     }
141     if(appButtons.But_ACC)
142     {
143

```

```

146     TFT_Font_List();
147     if(Nb_Students_In_List > 9)
148     {
149         TFT_aff_list_more (Nb_Students_In_List - 9);
150     }
151     MenuList = MENU_List;
152     I = I_Nbstudents;
153     ClearALLButtons();
154 }
155 if(appButtons.But DECL)
156 {
157
158     TFT_Font_List();
159     if(Nb_Students_In_List > 9)
160     {
161         TFT_aff_list_more (Nb_Students_In_List - 9);
162     }
163     ClearALLButtons();
164     I = I_Nbstudents;
165     MenuList = MENU_List;
166 }
167 if(appButtons.But_Reset_Tick)
168 {
169
170 // // TFT_Font_List();
171 // if(Nb_Students_In_List > 9)
172 // {
173 //     TFT_aff_list_more (Nb_Students_In_List - 9);
174 // }
175 // ClearALLButtons();
176 // I = I_Nbstudents;
177 // MenuList = MENU_List;
178 //
179 }
180 }
181 break;
182 }
183
184 case MENU_Config:
185 {
186     if(appButtons.But_UP)
187     {
188         if(PosCursorConfig != 0)
189         {
190             PosCursorConfig++;
191             TFT_Update_Config_cursor (PosCursorConfig);
192         }
193         ClearALLButtons();
194     }
195     else if(appButtons.But_Down)
196     {
197         if(PosCursorConfig < I_Nbstudents)
198         {
199             PosCursorConfig--;
200             TFT_Update_Config_cursor (PosCursorConfig);
201         }
202         ClearALLButtons();
203     }
204     else if(appButtons.But_OK)
205     {
206
207         ClearALLButtons();
208     }
209     else if(appButtons.But_ESC)
210     {
211
212         ClearALLButtons();
213     }
214     break;
215 }
216
217 case MENU_Initialisation:
218 {

```

```

219             Menu_loading_init ();
220         }
221     break;
222 }
223 }
224
225
226 void Init_Menu (void)
227 {
228     ClearALLButtons();
229     MenuList = MENU_Initialisation;
230     MenuSelected = SELECTED_NONE;
231 }
232
233 void MenuChange (MENU NewMenu)
234 {
235     MenuList = NewMenu;
236 }
237
238 void ClearALLButtons (void)
239 {
240     appButtons.But_Down = false;
241     appButtons.But_ESC = false;
242     appButtons.But_OK = false;
243     appButtons.But_UP = false;
244     appButtons.But DECL = false;
245     appButtons.But_Reset_Tick = false;
246     appButtons.But_ACC = false;
247 }
248
249 void Menu_loading_init (void)
250 {
251     int I = 0;
252     int J = 0;
253     int Rx_Taille;
254     U_32 Rx_Message;
255     U_32 AddSource;
256     U_32 TX_Message;
257     U_32 AddTarget;
258     static int init_status = 0;
259
260
261
262     switch (init_status)
263     {
264         case 0:
265         {
266             if (SDCARDPLUGED)
267             {
268                 TFT_SD_PLUG_OK ();
269                 init_status++;
270             }
271             else
272             {
273                 init_status = 6;
274             }
275             break;
276         }
277         case 1:
278         {
279             if (SDCARDMOUNTED)
280             {
281                 TFT_SD_Mount_OK ();
282                 init_status++;
283             }
284             break;
285         }
286         case 2:
287         {
288             if (SDCARDFILEFOUND)
289             {
290                 TFT_Student_File_Detected ();
291                 init_status++;
292             }
293         }
294     }
295 }

```

```

292         }
293     break;
294 }
295 case 3:
296 {
297     if(SDCARDFILEREADED)
298     {
299         TFT_List_Of_Students_OK ();
300         TFT_Students_Found(Nbstudents);
301         init_status++;
302     }
303     break;
304 }
305 case 4:
306 {
307     if(!Students_Info[I].IsOk )
308     {
309         TFT_Cheking_Link();
310         init_status++;
311         break;
312     }
313     break;
314 }
315 case 5:
316 {
317     J = 0;
318     for(I = 0; I < I_Nbstudents; I++)
319     {
320         if (Students_Info[I].IsOk == true)
321         {
322             J++;
323         }
324     }
325     TFT_Students_Link (J);
326     init_status++;
327     TFT_Init_OK ();
328     break;
329 }
330 case 6:
331 {
332     if(appButtons.But_ACC)
333     {
334         MenuList = MENU_MenuPrincipal;
335         TFT_Cursor_MenuPrincipale ();
336         TFT_Menu_Principal ();
337         ClearALLButtons();
338         init_status = 0;
339     }
340     else if (appButtons.But DECL)
341     {
342         init_status = 0;
343         TFT_Font_debug();
344         ClearALLButtons();
345     }
346     break;
347 }
348 }
349 }
350 }
351
352
353
354 //interruption lors de changement d'etat sur les boutons
355 void __ISR(_CHANGE_NOTICE_VECTOR, ipl3AUTO) _IntHandlerChangeNotification(void)
356 {
357     APP_UpdateState(APP_STATE_GEST_MENU);
358
359     //Bouton DECLINE
360     if (PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_A, PORTS_BIT_POS_0) && (
361     CNA0IsEnable))
362     {
363         flagDECLPressed = true;
364     }

```

```

364
365 //Bouton RESET TICKET
366 if (PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_A, PORTS_BIT_POS_1) && (
367 CNA1IsEnable))
368 {
369     flagRESETpressed = true;
370 }
371
372 //Bouton ACCEPT
373 if (PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_A, PORTS_BIT_POS_7) && (
374 CNA7IsEnable))
375 {
376     flagACCPressed = true;
377 }
378
379 //Bouton OK
380 if (PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_A, PORTS_BIT_POS_8) && (
381 CNA8IsEnable))
382 {
383     flagOkpressed = true;
384 }
385
386 //Bouton DOWN
387 if (PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_3) && (
388 CNB3IsEnable))
389 {
390     flagDownPressed = true;
391 }
392
393 //Bouton UP
394 if (PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_4) && (
395 CNB4IsEnable))
396 {
397     flagUpPressed = true;
398 }
399
400
401 //Bouton ESC
402 if (PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_13) && (
403 CNB13IsEnable))
404 {
405     flagEscPressed = true;
406 }
407
408 ****
409
410 if (flagDECLPressed)
411 {
412     if(!PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_A, PORTS_BIT_POS_0))
413     {
414         appButtons.But_DECL = true;
415         flagDECLPressed = false;
416     }
417 }
418 if (flagRESETpressed)
419 {
420     if(!PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_A, PORTS_BIT_POS_1))
421     {
422         appButtons.But_Reset_Tick= true;
423         flagRESETpressed = false;
424     }
425 }
426 if (flagACCPressed)
427 {
428     if(!PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_A, PORTS_BIT_POS_7))
429     {
430         appButtons.But_ACC = true;
431         flagACCPressed = false;
432     }
433 }
434 if (flagOkpressed)
435 {
436     if(!PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_A, PORTS_BIT_POS_8))
437     {
438         appButtons.But_OK = true;
439     }
440 }

```

```
431         flagOkpressed = false;
432     }
433 }
434
435 if (flagDownPressed)
436 {
437     if(!PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_3))
438     {
439         appButtons.But_Down = true;
440         flagDownPressed = false;
441     }
442 }
443
444 if (flagUpPressed)
445 {
446     if(!PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_4))
447     {
448         appButtons.But_UP = true;
449         flagUpPressed = false;
450     }
451 }
452
453 if (flagEscPressed)
454 {
455     if(!PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_13))
456     {
457         appButtons.But_ESC = true;
458         flagEscPressed = false;
459     }
460 }
461
462 PLIB_INT_SourceFlagClear(INT_ID_0,INT_SOURCE_CHANGE_NOTICE_A);
463 PLIB_INT_SourceFlagClear(INT_ID_0,INT_SOURCE_CHANGE_NOTICE_B);
464 }
```

```

1  /* **** */
2  /** Descriptive File Name
3
4  @Company
5      Company Name
6
7  @File Name
8      filename.h
9
10 @Summary
11     Brief description of the file.
12
13 @Description
14     Describe the purpose of this file.
15 */
16 /* **** */
17
18 #ifndef _GEST_MENU_H      /* Guard against multiple inclusion */
19 #define _GEST_MENU_H
20
21
22
23 #ifdef __cplusplus
24 }
25 #endif
26
27 typedef enum {
28     SELECTED_NONE = 0,
29     SELECTED_LIST = 1,
30     SELECTED_Configuration = 2,
31
32 }MENUSELECT;
33
34 typedef struct
35 {
36     /* The application's current state */
37     bool But_UP;
38     bool But_Down;
39     bool But_OK;
40     bool But_ESC;
41     bool But_ACC;
42     bool But DECL;
43     bool But_Reset_Tick;
44     /* TODO: Define any additional data used by the application. */
45
46 } APP_BUTTON;
47
48 typedef enum {
49     MENU_MenuPrincipal = 0,
50     MENU_List = 1,
51     MENU_Initialisation,
52     MENU_Config,
53     Idle,
54 }MENU;
55
56 APP_BUTTON appButtons;
57 MENU MenuList;
58 MENUSELECT MenuSelected;
59
60
61 //int IntegrationTimeTmp = 0;
62 //bool Integration_Auto = true;
63
64
65 void Gest_Menu (void);
66 void MenuChange (MENU NewMenu);
67 void TFT_CablrilationInfo(void);
68 void ClearALLButtons (void);
69 void Init_Menu(void);
70 void Menu_loading_init (void);
71
72     /* Provide C++ Compatibility */
73 #ifdef __cplusplus

```

```
74 }
75 #endif
76
77 #endif /* _EXAMPLE_FILE_NAME_H */
78
79 /* **** End of File ****
80
81 */
82
```

```

1 //-----
2 // RF.c
3 //-----
4 // Gestion communication
5 //
6 // Auteur      : SCA
7 // Date       : 5.12.2019
8 // Version    : V1.0
9 // Compilateur : XC32 V2.15
10 // Modifications : MDS 26.09.2022
11 //      code repris du projet 1623 20200214_Module_Xbee_06_SCA
12 //      -
13 /*-----*/
14 #include "RF.h"
15 #include "app.h"
16 #include <xc.h> //pour les définitions des registres
17 #include "Mc32Delays.h"
18 #include "peripheral/usart/plib_usart.h"
19 #include <string.h> //pour memcpy()
20
21 bool Get_Add_Master = false;
22
23
24 void RF_Init(void)
25 {
26     uint32_t dummy, dummy2, dummy3;
27
28     //reset module RF 1 ms
29
30     RstOff();
31     delay_msCt(1);
32     RstOn();
33
34     delay_msCt(10); //attendre fin init module rf
35
36     // puis envoyer trame pour sortie mode config
37     //RF_SendMessage((uint8_t*)"AT+EXIT\r\n", 0);
38
39
40     Get_Add_Master = true;
41
42     //lire sa propre adresse
43     RF_SendMessage((uint8_t*)"AT+GADD", 0);
44     GetMessage(dummy2, dummy, dummy3);
45
46
47     delay_msCt(10); //attendre traitement cmde EXIT
48 }
49
50
51 /* ***** */
52 //envoi trame au module RF via UART 1
53 // soit un nb de bytes défini par nbBytesToSend
54 // sinon si nbBytesToSend==0, envoi jusqu'à trouver une fin de chaine (car. 0)
55 void RF_SendMessage(uint8_t* dataToSend, uint8_t nbBytesToSend)
56 //void Uart_SendMessage(uint8_t* dataToSend, uint8_t nbBytesToSend)
57 {
58     uint8_t i = 0;
59
60     if (nbBytesToSend != 0) //nb de car. à envoyer spécifié
61     {
62         for (i=0 ; i<nbBytesToSend ; i++)
63         {
64             while(PLIB_USART_TransmitterBufferIsFull(USART_ID_1));
65             PLIB_USART_TransmitterByteSend(USART_ID_1, dataToSend[i]);
66         }
67     } else //chaine à envoyer (se termine par car. 0)
68     {
69         while (dataToSend[i] != 0)
70         {
71             while(PLIB_USART_TransmitterBufferIsFull(USART_ID_1));
72             PLIB_USART_TransmitterByteSend(USART_ID_1, dataToSend[i]);
73             i++;
74         }
75     }
76 }

```

74
75
76
77

}
}

```
1 //-----  
2 // RF.h  
3 //-----  
4 // Gestion communication  
5 //  
6 // Auteur : SCA  
7 // Date : 5.12.2019  
8 // Version : V1.0  
9 // Compilateur : XC32 V2.15  
10 // Modifications :  
11 // -  
12 /*-----*/  
13  
14 #ifndef _RF_H /* Guard against multiple inclusion */  
15 #define _RF_H  
16  
17 #include <stdint.h>  
18 #include <stdbool.h>  
19  
20 /* ***** */  
21  
22  
23 /* ***** */  
24 extern bool Get_Add_Master;  
25  
26  
27 /* ***** */  
28  
29 void RF_Init(void);  
30 void RF_SendMessage(uint8_t* dataToSend, uint8_t nbBytesToSend);  
31  
32 #endif /* _RF_H */  
33  
34 /* ***** */  
35 End of File  
36 /* */  
37
```

```

1 //18111C_Master
2 // Mc32Gest_RS232.C
3 // Canevas manipulatio TP2 RS232 SLO2 2016-2017
4 // Fonctions d'émission et de réception des message
5 // CHR 20.12.2016 ajout traitement int error
6 // CHR 22.12.2016 evolution des marquers observation int Usart
7 // MDS 26.09.2022 Modification pour permettre la communication avec le Xbee et la
gestion des donnee reçus
8
9 #include <xc.h>
10 #include <sys/attribs.h>
11 #include <stdint.h>
12 #include "system_definitions.h"
13 // Ajout CHR
14 #include "app.h"
15 #include "Mc32gest_RS232.h"
16 #include <GenericTypeDefs.h>
17 //Ajout MDS
18 #include "GesFifoTh32.h"
19 #include "RF.h"
20 #include "Mc32Delays.h"
21 #include "Data_Code.h"
22
23
24 //definition du byte de fin de trame
25 #define END 0xBB
26
27 //definition du byte de debut de trame
28 #define START 0xAA
29
30 uint8_t countCar = 0;           // Compteur du nombre de characters d'un nom
31 char buffReadName[20] = {' '}; // Buffer de reception de l'UART
32 uint8_t Name_Receive = false; //Flag Nom recu
33
34 typedef union
35 {
36     uint32_t val32;
37
38     struct
39     {
40         uint8_t msb;
41         uint8_t byte1;
42         uint8_t byte2;
43         uint8_t lsb;
44     }
45     sh1;
46 }
47 U_manip32;
48 typedef struct {
49     uint8_t Start;
50     U_32 Add_Master;
51     U_32 Add_Slave;
52     U_32 Broadcast;
53     U_32 data;
54     uint8_t End;
55
56 } StruMess;
57
58 // Struct pour émission des messages
59 StruMess TxMess;
60 // Struct pour réception des messages
61 StruMess RxMess;
62
63 // Declaration des FIFO pour réception et émission
64 #define FIFO_RX_SIZE ( (63 * 7) + 1) // 63 messages
65 #define FIFO_TX_SIZE ( (63 * 7) + 1) // 63 messages
66
67 int8_t fifoRX[FIFO_RX_SIZE];
68 // Declaration du descripteur du FIFO de réception
69 S_fifo descrFifoRX;
70
71 int8_t fifoTX[FIFO_TX_SIZE];

```

```

73 // Declaration du descripteur du FIFO d'émission
74 S_fifo descrFifoTX;
75
76 bool LINK,Add_ON;
77 uint32_t Add_Master = 0;
78 uint32_t Add_Slave = 0;
79 bool ID_In_List;
80 bool New_student;
81 uint8_t Nb_In_List;
82
83 // Initialisation de la communication serial
84 // -----
85
86 void InitFifoComm(void)
87 {
88
89     // Initialisation du fifo de reception
90     InitFifo (&descrFifoRX, FIFO_RX_SIZE, fifoRX, 0 );
91     // Initialisation du fifo d'émission
92     InitFifo (&descrFifoTX, FIFO_TX_SIZE, fifoTX, 0 );
93
94 } // InitComm
95
96
97 bool GetMessage(U_32 *pAdd_S,U_32 *pAdd_M, U_32 *pDataS)
98 {
99     bool startOk = false;
100
101     RxMess.End = END;
102     int8_t CarLu,i,y,x;
103     uint8_t car ;
104     if(Get_Add_Master)
105     {
106
107         uint8_t* dstArray ;
108
109         SYS_INT_SourceDisable(INT_SOURCE_USART_1_RECEIVE); //désactive int uart rx
110
111         while(GetReadSize(&descrFifoRX) > 0)
112         {
113
114             GetCharFromFifo (&descrFifoRX, (int8_t*)&CarLu);
115             dstArray[i] = CarLu;
116             i++;
117         }
118         Get_Add_Master = false;
119         Add_Master = (uint32_t)dstArray;
120
121         SYS_INT_SourceEnable(INT_SOURCE_USART_1_RECEIVE); //réactive int uart rx
122     }
123     else
124     {
125         //vérifie longueur message et présence start
126         // trame 14 byte min:
127         // 1 byte de start
128         // 4 byte d'adresse de l'expéditeur
129         // 4 byte d'adresse de destinataire
130         // 4 byte de donnée
131         // 1 byte de fin
132         while((GetReadSize(&descrFifoRX)) >= 14)
133         {
134             GetCharFromFifo (&descrFifoRX, &CarLu);
135             if (CarLu == (int8_t)START)
136             {
137                 startOk = true;
138
139                 break;
140             }
141         }
142         //Start trouvé, lire message et décoder
143         if (startOk)
144         {
145             //prendre de la fifo les 4 byte de l'expéditeur

```

```

146
147     GetCharFromFifo (&descrFifoRX, &CarLu);
148     pAdd_S->U_32_Bytes.msb = CarLu;
149     GetCharFromFifo (&descrFifoRX, &CarLu);
150     pAdd_S->U_32_Bytes.byte1 = CarLu;
151     GetCharFromFifo (&descrFifoRX, &CarLu);
152     pAdd_S->U_32_Bytes.byte2 = CarLu;
153     GetCharFromFifo (&descrFifoRX, &CarLu);
154     pAdd_S->U_32_Bytes.lsb = CarLu;
155
156     //prendre de la fifo les 4 byte du destinataire
157     GetCharFromFifo (&descrFifoRX, &CarLu);
158     pAdd_M->U_32_Bytes.msb = CarLu;
159     GetCharFromFifo (&descrFifoRX, &CarLu);
160     pAdd_M->U_32_Bytes.byte1 = CarLu;
161     GetCharFromFifo (&descrFifoRX, &CarLu);
162     pAdd_M->U_32_Bytes.byte2 = CarLu;
163     GetCharFromFifo (&descrFifoRX, &CarLu);
164     pAdd_M->U_32_Bytes.lsb = CarLu;
165
166     //lecture byte dans fifo
167     GetCharFromFifo (&descrFifoRX, &CarLu);
168     //tant que l'on a pas le byte de fin on enregistre les donnees
169     do
170     {
171         //copie des donnee dans un tableau
172         buffReadName[countCar] = CarLu;
173         // lire la prochaine valeur du fifo
174         GetCharFromFifo (&descrFifoRX, &CarLu);
175         //incrementation dans le tableau
176         countCar++;
177     }
178     while ( CarLu != '»' );
179
180     //si les data sont <4 alors c'est une commande
181     if(countCar <= 4)
182     {
183         //memset(&pDatas, 0x00, sizeof(pDatas));
184         pDatas->U_32_Bytes.msb = buffReadName[0];
185         pDatas->U_32_Bytes.byte1 = buffReadName[1];
186         pDatas->U_32_Bytes.byte2 = buffReadName[2];
187         pDatas->U_32_Bytes.lsb = buffReadName[3];
188     }
189     //si > alors c'est un nom d'eleve
190     else
191     {
192         //on verifie la liste d'eleve
193         for( i = 0; i < Nb_Student_max ; i++ )
194         {
195             x=0;
196             //on verifie les lettre d'un eleve
197             for( y = 0; y < countCar ; y++ )
198             {
199                 //on compare si ce sont les meme
200                 if(buffReadName[countCar] == Students_Info[i].StudentName[
201                     countCar])
202                 {
203                     x++;
204                     if(x == countCar)
205                     {
206                         //flag pour indiquer que le nom est deja dans la list
207                         ID_In_List = true;
208                         //on indique quel position dans la list
209                         Nb_In_List = i;
210                         // on sort de la boucle
211                         i = Nb_Student_max;
212                     }
213                 }
214                 //flag pour indiquer que le nom n'est pas dans la list
215                 ID_In_List = false;
216                 //on sort de la boucle
217                 y=countCar;

```

```

218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249 // Fonction d'envoi des message du Xbee
250 void SendMessage(uint32_t ADD_M,uint32_t ADD_S, uint32_t pDatas)
251 {
252     uint8_t FreeSize,i,n;
253     TxMess.Start = START;
254     TxMess.Add_Master.val32 = ADD_M;
255     TxMess.Add_Slave.val32 = ADD_S;
256     TxMess.data.val32 = pDatas;
257     TxMess.End = END;
258     TxMess.Broadcast.val32 = ADD_BROADCAST;
259
260     //vérifie longueur disponible dans le fifo
261     // trame 14 byte min:
262     // 1 byte de start
263     // 4 byte d'adresse de l'expéditeur
264     // 4 byte d'adresse de destinataire
265     // 4 byte de donnée
266     // 1 byte de fin
267     if (GetWriteSpace(&descrFifoTX) >= 14)
268     {
269         // on met le byte de début de trame dans le fifo
270         PutCharInFifo (&descrFifoTX, TxMess.Start);
271         //on vérifie si l'on a déjà essayé de se LINK
272         if(!LINK)
273         {
274             // on met l'adresse de broadcast dans le fifo
275             PutCharInFifo (&descrFifoTX, TxMess.Broadcast.U_32_Bytes.msb);
276             PutCharInFifo (&descrFifoTX, TxMess.Broadcast.U_32_Bytes.byte1);
277             PutCharInFifo (&descrFifoTX, TxMess.Broadcast.U_32_Bytes.byte2);
278             PutCharInFifo (&descrFifoTX, TxMess.Broadcast.U_32_Bytes.lsb);
279             LINK = true;
280         }
281
282
283         // on met l'adresse de l'expéditeur dans le fifo
284         PutCharInFifo (&descrFifoTX, TxMess.Add_Master.U_32_Bytes.msb);
285         PutCharInFifo (&descrFifoTX, TxMess.Add_Master.U_32_Bytes.byte1);
286         PutCharInFifo (&descrFifoTX, TxMess.Add_Master.U_32_Bytes.byte2);
287         PutCharInFifo (&descrFifoTX, TxMess.Add_Master.U_32_Bytes.lsb);
288
289         //on vérifie si on a une adresse à envoyer

```

```

290
291     if(Add_ON)
292     {
293         // on met l'adresse du destinataire dans le fifo
294         PutCharInFifo (&descrFifoTX, TxMess.Add_Slave.U_32_Bytes.msb);
295         PutCharInFifo (&descrFifoTX, TxMess.Add_Slave.U_32_Bytes.byte1);
296         PutCharInFifo (&descrFifoTX, TxMess.Add_Slave.U_32_Bytes.byte2);
297         PutCharInFifo (&descrFifoTX, TxMess.Add_Slave.U_32_Bytes.lsb);
298     }
299
300     // on met les datas dans le fifo
301     PutCharInFifo (&descrFifoTX, TxMess.data.U_32_Bytes.msb);
302     PutCharInFifo (&descrFifoTX, TxMess.data.U_32_Bytes.byte1);
303     PutCharInFifo (&descrFifoTX, TxMess.data.U_32_Bytes.byte2);
304     PutCharInFifo (&descrFifoTX, TxMess.data.U_32_Bytes.lsb);
305
306     // on met le byte de fin de trame dans le fifo
307     PutCharInFifo (&descrFifoTX, TxMess.End);
308 }
309 // On met le flag d'envoie de l'uart Xbee
310 PLIB_INT_SourceEnable(INT_ID_0, INT_SOURCE_USART_1_TRANSMIT);
311
312 // !!!!!!!
313 // Attention ne pas oublier de supprimer la réponse générée dans system_interrupt
314 // !!!!!!!
315 void __ISR(_UART_1_VECTOR, ipl5AUTO) _IntHandlerDrvUsartInstance0(void)
316 {
317     USART_ERROR UsartStatus;
318     int8_t Carac, TXsize, TxBuffFull;
319     // Is this an Error interrupt ?
320     if ( PLIB_INT_SourceFlagGet(INT_ID_0, INT_SOURCE_USART_1_ERROR) &&
321         PLIB_INT_SourceIsEnabled(INT_ID_0, INT_SOURCE_USART_1_ERROR) ) {
322         /* Clear pending interrupt */
323         PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_USART_1_ERROR);
324         // Traitement de l'erreur à la réception.
325     }
326
327
328     // Is this an RX interrupt ?
329     if ( PLIB_INT_SourceFlagGet(INT_ID_0, INT_SOURCE_USART_1_RECEIVE) &&
330         PLIB_INT_SourceIsEnabled(INT_ID_0, INT_SOURCE_USART_1_RECEIVE) ) {
331
332         // Oui Test si erreur parité ou overrun
333         UsartStatus = PLIB_USART_ErrorsGet(USART_ID_1);
334
335         if ( (UsartStatus & (USART_ERROR_PARITY |
336             USART_ERROR_FRAMING | USART_ERROR_RECEIVER_OVERRUN)) == 0)
337         {
338             while(PLIB_USART_ReceiverDataIsAvailable(USART_ID_1))
339             {
340                 //Led_QuestToggle();
341                 Carac = PLIB_USART_ReceiverByteReceive(USART_ID_1);
342                 PutCharInFifo(&descrFifoRX, Carac);
343
344             }
345             APP_UpdateState(APP_STATE_GET_DATA);
346             // buffer is empty, clear interrupt flag
347             PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_USART_1_RECEIVE);
348             //Message_receive_XBEE = true;
349         } else {
350             // Suppression des erreurs
351             // La lecture des erreurs les efface sauf pour overrun
352             if ( (UsartStatus & USART_ERROR_RECEIVER_OVERRUN) ==
353                 USART_ERROR_RECEIVER_OVERRUN) {
354                 PLIB_USART_ReceiverOverrunErrorClear(USART_ID_1);
355             }
356         }
357     } // end if RX
358
359
360     // Is this an TX interrupt ?
361

```

```

362     if ( PLIB_INT_SourceFlagGet(INT_ID_0, INT_SOURCE_USART_1_TRANSMIT) &&
363         PLIB_INT_SourceIsEnabled(INT_ID_0, INT_SOURCE_USART_1_TRANSMIT) ) {
364
365     TXsize = GetReadSize(&descrFifoTX);
366     TxBuffFull = PLIB_USART_TransmitterBufferIsFull(USART_ID_1);
367     if ((TXsize > 0)&& (TxBuffFull == false))
368     {
369         do//Faire la boucle tant que le message n'est pas envoyé
370         {
371             //Led_QuestToggle();
372             //On va chercher les valeurs a envoyer
373             GetCharFromFifo(&descrFifoTX, &Carac);
374             PLIB_USART_TransmitterByteSend(USART_ID_1, Carac);
375             TXsize = GetReadSize(&descrFifoTX);
376             TxBuffFull = PLIB_USART_TransmitterBufferIsFull(USART_ID_1);
377
378         }while((TXsize > 0)&& (TxBuffFull == false) && (Carac!= '»'));
379
380         // Clear the TX interrupt Flag (Seulement apres TX)
381         PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_USART_1_TRANSMIT);
382         TXsize = GetReadSize(&descrFifoTX);
383         if (TXsize == 0)
384         {
385             //Pour eviter les interruption inutile
386             PLIB_INT_SourceDisable(INT_ID_0, INT_SOURCE_USART_1_TRANSMIT);
387         }
388     }
389     else
390     {
391         // disable TX interrupt (pour éviter une interrupt. inutile si plus
392         // rien à transmettre)
393         PLIB_INT_SourceDisable(INT_ID_0, INT_SOURCE_USART_1_TRANSMIT);
394     }
395 }
396
397 // void __ISR(_UART_2_VECTOR, ipl5AUTO) _IntHandlerDrvUsartInstance1(void)
398 //{
399 //     USART_ERROR UsartStatus;
400 //     int8_t Carac, TXsize, TxBuffFull;
401 //     // Is this an Error interrupt ?
402 //     if ( PLIB_INT_SourceFlagGet(INT_ID_0, INT_SOURCE_USART_2_ERROR) &&
403 //         PLIB_INT_SourceIsEnabled(INT_ID_0, INT_SOURCE_USART_2_ERROR) ) {
404 //         /* Clear pending interrupt */
405 //         PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_USART_2_ERROR);
406 //         // Traitement de l'erreur à la réception.
407 //     }
408 //
409 //
410 //     // Is this an RX interrupt ?
411 //     if ( PLIB_INT_SourceFlagGet(INT_ID_0, INT_SOURCE_USART_2_RECEIVE) &&
412 //         PLIB_INT_SourceIsEnabled(INT_ID_0, INT_SOURCE_USART_2_RECEIVE) ) {
413 //
414 //         // Oui Test si erreur parité ou overrun
415 //         UsartStatus = PLIB_USART_ErrorsGet(USART_ID_2);
416 //
417 //         if ( (UsartStatus & (USART_ERROR_PARITY |
418 //             USART_ERROR_FRAMING | USART_ERROR_RECEIVER_OVERRUN)) == 0 )
419 //         {
420 //             while(PLIB_USART_ReceiverDataIsAvailable(USART_ID_2))
421 //             {
422 //                 Carac = PLIB_USART_ReceiverByteReceive(USART_ID_2);
423 //                 PutCharInFifo(&descrFifoRX, Carac);
424 //             }
425 //             // buffer is empty, clear interrupt flag
426 //             PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_USART_2_RECEIVE);
427 //             Message_receive_USB = true;
428 //         } else {
429 //             // Suppression des erreurs
430 //             // La lecture des erreurs les efface sauf pour overrun
431 //             if ( (UsartStatus & USART_ERROR_RECEIVER_OVERRUN) ==
432 //                 USART_ERROR_RECEIVER_OVERRUN) {
433 //                 PLIB_USART_ReceiverOverrunErrorClear(USART_ID_2);

```

```

433 // }
434 // }
435 //
436 // } // end if RX
437 //
438 //
439 //
440 // Is this an TX interrupt ?
441 // if ( PLIB_INT_SourceFlagGet(INT_ID_0, INT_SOURCE_USART_2_TRANSMIT) &&
442 //      PLIB_INT_SourceIsEnabled(INT_ID_0, INT_SOURCE_USART_2_TRANSMIT) ) {
443 //
444 //     TXsize = GetReadSize(&descrFifoTX);
445 //     TxBuffFull = PLIB_USART_TransmitterBufferIsFull(USART_ID_2);
446 //     if ((TXsize > 0)&& (TxBuffFull == false))
447 //     {
448 //         do//Faire la boucle tant que le message n'est pas envoyé
449 //         {
450 //             //On va chercher les valeurs a envoyer
451 //             GetCharFromFifo(&descrFifoTX, &Carac);
452 //             PLIB_USART_TransmitterByteSend(USART_ID_2, Carac);
453 //             TXsize = GetReadSize(&descrFifoTX);
454 //             TxBuffFull = PLIB_USART_TransmitterBufferIsFull(USART_ID_2);
455 //
456 //         }while((TXsize > 0)&& (TxBuffFull == false));
457 //
458 //         // Clear the TX interrupt Flag (Seulement apres TX)
459 //         PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_USART_2_TRANSMIT);
460 //         TXsize = GetReadSize(&descrFifoTX);
461 //         if (TXsize == 0)
462 //         {
463 //             //Pour eviter les interruption inutile
464 //             PLIB_INT_SourceDisable(INT_ID_0, INT_SOURCE_USART_2_TRANSMIT);
465 //         }
466 //     }
467 //     else
468 //     {
469 //         // disable TX interrupt (pour éviter une interrupt. inutile si plus
470 //         // rien à transmettre)
471 //         PLIB_INT_SourceDisable(INT_ID_0, INT_SOURCE_USART_2_TRANSMIT);
472 //     }
473 // }

```

```

1 #ifndef Mc32Gest_RS232_H
2 #define Mc32Gest_RS232_H
3 /*-----
4 // Mc32Gest_RS232.h
5 -----*/
6 // Description : emission et reception spécialisée
7 // pour TP2 2016-2017
8 //
9 // Auteur : C. HUBER
10 //
11 // Version : V1.3
12 // Compilateur : XC32 V1.42 + Harmony 1.08
13 //
14 /*-----*/
15
16 #include "GesFifoTh32.h"
17
18
19
20
21 typedef union
22 {
23     uint32_t val32;
24
25     struct
26     {
27         uint8_t msb;
28         uint8_t byte1;
29         uint8_t byte2;
30         uint8_t lsb;
31     }
32     U_32_BytEs;
33 }
34 U_32;
35
36 /*-----
37 // Définition des fonctions prototypes
38 -----*/
39
40 // prototypes des fonctions
41 void InitFifoComm(void);
42
43
44 // Descripteur des fifos
45 extern S_fifo descrFifoRX;
46 extern S_fifo descrFifoTX;
47
48 extern bool Message_receive_USB;
49 extern bool Message_receive_XBEE;
50 extern bool LINK,Add_ON;
51
52 extern uint32_t Add_Master;
53 extern uint32_t Add_Slave;
54
55 extern uint8_t Nb_Student;
56 extern uint8_t Nb_In_List;
57 extern bool New_student;
58
59 bool GetMessage(U_32 *Add_S,U_32 *Add_M, U_32 *pDatas);
60 void SendMessage(uint32_t Add_M,uint32_t Add_S, uint32_t pDatas);
61
62 #endif
63

```

```

1 // ETML Ecole Technique
2 // Fichier GesFifoTh.c
3
4 // Exemple gestion d'un fifo de caractère, utilisation de pointeur et
5 // d'un descripteur de fifo
6
7 // Auteur      : C. Huber
8 //
9 // Version     : V1.6
10 // Compilateur : XC32 V1.42 + Harmony 1.08
11 //
12 // Modifications :
13 //      CHR 19.12.2014 remplacement typedef32 par stdint
14 //      CHR 20.12.2016  fifosize en int32_t pour permettre des
15 //                      fifo de grande taille
16 //
17 /*-----*/
18 #include <stdint.h>
19 #include "app.h"
20 #include "GesFifoTh32.h"
21
22 /*-----*/
23 /* InitFifo */
24 /*=====*/
25
26 // Init avec possibilité de fournir une valeur de remplissage
27 // Initialisation du descripteur de FIFO
28
29 void InitFifo ( S_fifo *pDescrFifo, int32_t FifoSize, int8_t *pDebFifo, int8_t InitVal
    )
30 {
31     int32_t i;
32     int8_t *pFif;
33     pDescrFifo->fifoSize = FifoSize;
34     pDescrFifo->pDebFifo = pDebFifo; // début du fifo
35     // fin du fifo
36     pDescrFifo->pFinFifo = pDebFifo + (FifoSize - 1);
37     pDescrFifo->pWrite   = pDebFifo; // début du fifo
38     pDescrFifo->pRead    = pDebFifo; // début du fifo
39     pFif = pDebFifo;
40     for (i=0; i < FifoSize; i++) {
41         *pFif = InitVal;
42         pFif++;
43     }
44 } /* InitFifo */
45
46
47 /*-----*/
48 /* GetWriteSpace */
49 /*=====*/
50
51 // Retourne la place disponible en écriture
52
53 int32_t GetWriteSpace ( S_fifo *pDescrFifo)
54 {
55     int32_t writeSize;
56
57     // Détermine le nb de car. que l'on peut déposer
58     writeSize = pDescrFifo->pRead - pDescrFifo->pWrite -1;
59     if (writeSize < 0) {
60         writeSize = writeSize + pDescrFifo->fifoSize;
61     }
62     return (writeSize);
63 } /* GetWriteSpace */
64
65
66 /*-----*/
67 /* GetReadSize */
68 /*=====*/
69
70 // Retourne le nombre de caractères à lire
71
72 int32_t GetReadSize ( S_fifo *pDescrFifo)

```

```

73
74     int32_t readSize;
75
76     readSize = pDescrFifo->pWrite - pDescrFifo->pRead;
77     if (readSize < 0) {
78         readSize = readSize + pDescrFifo->fifoSize;
79     }
80
81     return (readSize);
82 } /* GetReadSize */
83
84 /*-----*/
85 /* PutCharInFifo */
86 /*-----*/
87
88 // Dépose un caractère dans le FIFO
89 // Retourne 0 si OK, 1 si FIFO full
90
91 uint8_t PutCharInFifo ( S_fifo *pDescrFifo, int8_t charToPut )
92 {
93     uint8_t writeStatus;
94
95     // test si fifo est FULL
96     if (GetWriteSpace(pDescrFifo) == 0) {
97         writeStatus = 1; // fifo FULL
98     }
99     else {
100         // écrit le caractère dans le FIFO
101         *(pDescrFifo->pWrite) = charToPut;
102
103         // incrément le pointeur d'écriture
104         pDescrFifo->pWrite++;
105         // gestion du reboulement
106         if (pDescrFifo->pWrite > pDescrFifo->pFinFifo) {
107             pDescrFifo->pWrite = pDescrFifo->pDebFifo;
108         }
109
110         writeStatus = 0; // OK
111     }
112     return (writeStatus);
113 } // PutCharInFifo
114
115
116 /*-----*/
117 /* GetCharFromFifo */
118 /*-----*/
119
120 // Obtient (lecture) un caractère du fifo
121 // retourne 0 si OK, 1 si empty
122 // le caractère lu est retourné par référence
123
124 uint8_t GetCharFromFifo ( S_fifo *pDescrFifo, int8_t *carLu )
125 {
126     int8_t readSize;
127     uint8_t readStatus;
128
129     // détermine le nb de car. que l'on peut lire
130     readSize = GetReadSize(pDescrFifo);
131
132     // test si fifo est vide
133     if (readSize == 0) {
134         readStatus = 1; // fifo EMPTY
135         *carLu = 0; // carLu = NULL
136     }
137     else {
138         // lis le caractère dans le FIFO
139         *carLu = *(pDescrFifo->pRead);
140
141         // incrément du pointeur de lecture
142         pDescrFifo->pRead++;
143         // gestion du reboulement
144         if (pDescrFifo->pRead > pDescrFifo->pFinFifo) {
145             pDescrFifo->pRead = pDescrFifo->pDebFifo;

```

```
146      }
147      readStatus = 0; // OK
148  }
149  return (readStatus);
150 } // GetCharFromFifo
151
152
153
```

```

1 #ifndef GesFifoTh32_H
2 #define GesFifoTh32_H
3 /*-----*/
4 //  GesFifoTh32.h
5 /*-----*/
6 //  Description :  Création entête pour utilisation
7 //                  de GesFifoTh32
8 //
9 //  Auteur       :  C. Huber
10 //
11 //  Version      :  V1.6
12 //  Compilateur :  XC32 V1.42 + Harmony 1.08
13 //
14 //  Modifications :
15 //      CHR 19.12.2014  remplacement typedef32 par stdint
16 //      CHR 20.12.2016  fifosize en int32_t pour permettre des
17 //                      fifo de grande taille
18 //
19 /*-----*/
20
21
22
23
24 // structure décrivant un FIFO
25 typedef struct fifo {
26     int32_t fifoSize;    // taille du fifo
27     int8_t *pDebFifo;   // pointeur sur début du fifo
28     int8_t *pFinFifo;   // pointeur sur fin du fifo
29     int8_t *pWrite;     // pointeur d'écriture
30     int8_t *pRead;      // pointeur de lecture
31 } S_fifo;
32
33 /*-----*/
34 /* Définition des fonctions de gestion du fifo */
35 /*-----*/
36
37 /*-----*/
38 /* InitFifo */
39 /*=====*/
40
41 // Initialisation du descripteur de FIFO
42 void InitFifo ( S_fifo *pDescrFifo, int32_t FifoSize, int8_t *pDebFifo, int8_t InitVal
43 );
44
45 /*-----*/
46 /* GetWriteSpace */
47 /*=====*/
48
49 // Retourne la place disponible en écriture
50 int32_t GetWriteSpace ( S_fifo *pDescrFifo);
51
52 /*-----*/
53 /* GetReadSize */
54 /*=====*/
55
56 // Retourne le nombre de caractères à lire
57
58 int32_t GetReadSize ( S_fifo *pDescrFifo);
59
60 /*-----*/
61 /* PutCharInFifo */
62 /*=====*/
63
64 // Dépose un caractère dans le FIFO
65 // Retourne 0 si OK, 1 si FIFO full
66
67 uint8_t PutCharInFifo ( S_fifo *pDescrFifo, int8_t charToPut );
68
69 /*-----*/
70 /* GetCharFromFifo */
71 /*=====*/
72

```

```
73 // Obtient (lecture) un caractère du fifo
74 // retourne 0 si OK, 1 si empty
75 // le caractère lu est retourné par référence
76
77 uint8_t GetCharFromFifo ( S_fifo *pDescrFifo, int8_t *carLu );
78
79 #endif
80
```

```
1 ****
2 System Interrupts File
3
4 File Name:
5 system_interrupt.c
6
7 Summary:
8 Raw ISR definitions.
9
10 Description:
11 This file contains a definitions of the raw ISRs required to support the
12 interrupt sub-system.
13
14 Summary:
15 This file contains source code for the interrupt vector functions in the
16 system.
17
18 Description:
19 This file contains source code for the interrupt vector functions in the
20 system. It implements the system and part specific vector "stub" functions
21 from which the individual "Tasks" functions are called for any modules
22 executing interrupt-driven in the MPLAB Harmony system.
23
24 Remarks:
25 This file requires access to the systemObjects global data structure that
26 contains the object handles to all MPLAB Harmony module objects executing
27 interrupt-driven in the system. These handles are passed into the individual
28 module "Tasks" functions to identify the instance of the module to maintain.
29 ****
30
31 // DOM-IGNORE-BEGIN
32 ****
33 Copyright (c) 2011-2014 released Microchip Technology Inc. All rights reserved.
34
35 Microchip licenses to you the right to use, modify, copy and distribute
36 Software only when embedded on a Microchip microcontroller or digital signal
37 controller that is integrated into your product or third party product
38 (pursuant to the sublicense terms in the accompanying license agreement).
39
40 You should refer to the license agreement accompanying this Software for
41 additional information regarding your rights and obligations.
42
43 SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND,
44 EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF
45 MERCHANTABILITY, TITLE, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE.
46 IN NO EVENT SHALL MICROCHIP OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER
47 CONTRACT, NEGLIGENCE, STRICT LIABILITY, CONTRIBUTION, BREACH OF WARRANTY, OR
48 OTHER LEGAL EQUITABLE THEORY ANY DIRECT OR INDIRECT DAMAGES OR EXPENSES
49 INCLUDING BUT NOT LIMITED TO ANY INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE OR
50 CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT OF
51 SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES
52 (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS.
53 ****
54 // DOM-IGNORE-END
55
56 // ****
57 // ****
58 // Section: Included Files
59 // ****
60 // ****
61
62 #include "system/common/sys_common.h"
63 #include "app.h"
64 #include "app_sdcard.h"
65 #include "system_definitions.h"
66
67 // ****
68 // ****
69 // Section: System Interrupt Vector Functions
70 // ****
71 // ****
```

```

74
75
76 static uint8_t compt_Link = 0;
77
78
79
80
81
82
83
84
85 /*
86
87 void __ISR(_CHANGE_NOTICE_VECTOR, ipl1AUTO) _IntHandlerChangeNotification(void)
88 {
89     // TODO: Add code to process interrupt here
90     PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_CHANGE_NOTICE_A);
91 } */
92
93
94
95 void __ISR(_TIMER_1_VECTOR, ipl1AUTO) IntHandlerDrvTmrInstance0(void)
96 {
97
98
99     PLIB_PORTS_PinToggle(PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_8);
100    PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_TIMER_1);
101 }
102
103 //timer 100ms
104 void __ISR(_TIMER_2_VECTOR, ipl1AUTO) IntHandlerDrvTmrInstance1(void)
105 {
106
107     if(compt_Link < 19)
108     {
109         compt_Link++;
110     }
111     else
112     {
113         APP_UpdateState(APP_STATE_LINK_XBEE);
114         compt_Link = 0;
115     }
116     PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_TIMER_2);
117 }
118
119 void __ISR(_SPI_1_VECTOR, ipl1AUTO) _IntHandlerSPIInstance0(void)
120 {
121     DRV_SPI_Tasks(sysObj.spiObjectIdx0);
122 }
123 void __ISR(_SPI_2_VECTOR, ipl1AUTO) _IntHandlerSPIInstance1(void)
124 {
125     DRV_SPI_Tasks(sysObj.spiObjectIdx1);
126 }
127 ****
128 End of File
129 */
130

```

```

1 //18111C_Master
2 /*-----*/
3 // DriverLCD.c
4 /*-----*/
5 // Description : Utilitaire qui gère l'initialization de
6 // l'écran ainsi que d'autres fonctions
7 //
8 // Auteur      : Paulo Gomes
9 // Version     : V1.0
10 //
11 /*-----*/
12 // Section: Included Files
13 /*-----*/
14 /*-----*/
15
16 #include "DriverLcd.h"
17 #include "Mc32Delays.h"
18 #include "DefineLCD.h"
19 #include "glcdfont.c"
20 #include "app.h"
21 #include <stdlib.h>
22 #include <stdio.h>
23 #include <string.h>
24
25 #define D_C_low PLIB_PORTS_PinWrite(PORTS_ID_0, PORT_CHANNEL_C, PORTS_BIT_POS_4, false)
26 #define D_C_High PLIB_PORTS_PinWrite(PORTS_ID_0, PORT_CHANNEL_C, PORTS_BIT_POS_4, true)
27
28 #define TFT_CS_Low
PLIB_PORTS_PinWrite(PORTS_ID_0, PORT_CHANNEL_C, PORTS_BIT_POS_3, false)
29 #define TFT_CS_High
PLIB_PORTS_PinWrite(PORTS_ID_0, PORT_CHANNEL_C, PORTS_BIT_POS_3, true)
30
31 #define BCKL_ON PLIB_PORTS_PinWrite(PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_5, true)
32 #define BCKL_OFF PLIB_PORTS_PinWrite(PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_5, false)
33
34 #define RST_ON PLIB_PORTS_PinWrite(PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_6, false)
35 #define RST_OFF PLIB_PORTS_PinWrite(PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_6, true)
36 /*-----*/
37 // Section: User Functions
38 /*-----*/
39
40 // Fonction qui permet d'envoyer des commandes
41 void writecommand(char c)
42 {
43     // Datacommand low
44     D_C_low;
45     // Chip Select low
46     TFT_CS_Low;
47
48     // Envoi des données avec le SPI
49
50     PLIB_SPI_BufferWrite(SPI_ID_1, c);
51     do {
52
53         } while (PLIB_SPI_IsBusy(SPI_ID_1));
54
55         // Chip select high
56         TFT_CS_High;
57     }
58
59 // Fonction qui permet d'envoyer des datas
60 void writedata(char c)
61 {
62     // Datacommand high
63     D_C_High;
64
65     // Chip Select low
66     TFT_CS_Low;
67
68     // Envoi des données avec le SPI
69     PLIB_SPI_BufferWrite(SPI_ID_1, c);
70     do {
71

```

```

72     } while (PLIB_SPI_IsBusy(SPI_ID_1));
73
74     // Chip select high
75     TFT_CS_High;
76 }
77
78
79
80 // Initialisation de l'écran
81 void tft_begin(void) {
82
83     //RST_ON;
84     RST_OFF;
85     BCKL_OFF;
86     BCKL_ON;
87
88     delay_ms(500); // delay 500 ms
89     BCKL_OFF;
90     //RST_OFF;
91     TFT_CS_High;// Chip Select low
92     D_C_High; // Datacommand high
93
94     writecommand(0xEF);
95     writedata(0x03);
96     writedata(0x80);
97     writedata(0x02);
98
99     writecommand(0xCF);
100    writedata(0x00);
101    writedata(0XC1);
102    writedata(0X30);
103
104    writecommand(0xED);
105    writedata(0x64);
106    writedata(0x03);
107    writedata(0X12);
108    writedata(0X81);
109
110    writecommand(0xE8);
111    writedata(0x85);
112    writedata(0x00);
113    writedata(0x78);
114
115    writecommand(0xCB);
116    writedata(0x39);
117    writedata(0x2C);
118    writedata(0x00);
119    writedata(0x34);
120    writedata(0x02);
121
122    writecommand(0xF7);
123    writedata(0x20);
124
125    writecommand(0xEA);
126    writedata(0x00);
127    writedata(0x00);
128
129    writecommand(ILI9341_PWCTR1); //Power control
130    writedata(0x23); //VRH[5:0]
131
132    writecommand(ILI9341_PWCTR2); //Power control
133    writedata(0x10); //SAP[2:0];BT[3:0]
134
135    writecommand(ILI9341_VMCTR1); //VCM control
136    writedata(0x3e);
137    writedata(0x28);
138
139    writecommand(ILI9341_VMCTR2); //VCM control2
140    writedata(0x86);
141
142    writecommand(ILI9341_MADCTL); // Memory Access Control
143    writedata(0x48);
144

```

```

145     writecommand(ILI9341_PIXFMT);
146     writedata(0x55);
147
148     writecommand(ILI9341_FRMCTR1);
149     writedata(0x00);
150     writedata(0x18);
151
152     writecommand(ILI9341_DFUNCTR); // Display Function Control
153     writedata(0x08);
154     writedata(0x82);
155     writedata(0x27);
156
157     writecommand(0xF2); // 3Gamma Function Disable
158     writedata(0x00);
159
160     writecommand(ILI9341_GAMMASET); //Gamma curve selected
161     writedata(0x01);
162
163     writecommand(ILI9341_GMCTRP1); //Set Gamma
164     writedata(0x0F);
165     writedata(0x31);
166     writedata(0x2B);
167     writedata(0x0C);
168     writedata(0x0E);
169     writedata(0x08);
170     writedata(0x4E);
171     writedata(0xF1);
172     writedata(0x37);
173     writedata(0x07);
174     writedata(0x10);
175     writedata(0x03);
176     writedata(0x0E);
177     writedata(0x09);
178     writedata(0x00);
179
180     writecommand(ILI9341_GMCTRN1); //Set Gamma
181     writedata(0x00);
182     writedata(0x0E);
183     writedata(0x14);
184     writedata(0x03);
185     writedata(0x11);
186     writedata(0x07);
187     writedata(0x31);
188     writedata(0xC1);
189     writedata(0x48);
190     writedata(0x08);
191     writedata(0x0F);
192     writedata(0x0C);
193     writedata(0x31);
194     writedata(0x36);
195     writedata(0x0F);
196     writecommand(ILI9341_SLOUT); //Exit Sleep
197     delay_ms(120);
198     writecommand(ILI9341_DISPON); //Display on
199
200     _width = ILI9341_TFTWIDTH;
201     _height = ILI9341_TFTHEIGHT;
202
203
204     // Backlight ON
205     BCKL_ON;
206 }
207
208
209
210
211 // Remplissage de l'écran avec une couleur
212 void tft_fillScreen(unsigned short color) {
213 /* Rempli l'écran en entier avec une certaine couleur
214 * Parameters: color: 16-bit color value
215 * Returns: Nothing
216 */
217     tft_fillRect(0, 0, _width, _height, color);

```

```

218 }
219
220 // Dessine et remplit un rectangle avec une couleur
221 void tft_fillRect(short x, short y, short w, short h,
222                     unsigned short color) {
223 /* Dessine un écran rempli avec une certaine couleur.
224 * Commence top-left (en haut à gauche) et on peut définir
225 * la taille et la hauteur
226 * Parameters:
227 *     x: x-coordinate of top-left vertex; top left of screen is x=0
228 *         and x increases to the right
229 *     y: y-coordinate of top-left vertex; top left of screen is y=0
230 *         and y increases to the bottom
231 *     w: width of rectangle
232 *     h: height of rectangle
233 *     color: 16-bit color value
234 * Returns:    Nothing
235 */
236
237 // rudimentary clipping (drawChar w/big text requires this)
238 if((x >= _width) || (y >= _height)) return;
239 if((x + w - 1) >= _width) w = _width - x;
240 if((y + h - 1) >= _height) h = _height - y;
241
242 // Défini l'endroit où écrire
243 tft_setAddrWindow(x, y, x+w-1, y+h-1);
244
245 // Masquage pour envoyer en 2x8bits
246 uint8_t hi = color >> 8;
247 uint8_t lo = color;
248
249 //D_C_High;// Datacommand high
250 D_C_low;
251 TFT_CS_Low; // Chip Select low
252
253 for(y=h; y>0; y--) {
254     for(x=w; x>0; x--) {
255         // MSB
256         PLIB_SPI_BufferWrite(SPI_ID_1, hi);
257         do {
258
259             } while (PLIB_SPI_IsBusy(SPI_ID_1));
260
261         // LSB
262         PLIB_SPI_BufferWrite(SPI_ID_1, lo);
263         do {
264
265             } while (PLIB_SPI_IsBusy(SPI_ID_1));
266
267         }
268     }
269     TFT_CS_High; // Chip Select high
270 }
271
272 // Fonction qui choisit l'endroit où on veut écrire ou dessiner
273 void tft_setAddrWindow(unsigned short x0, unsigned short y0, unsigned short x1,
274                         unsigned short y1) {
275
276     // x
277     writecommand(ILI9341_CASET); // Column addr set
278     writedata(x0 >> 8);
279     writedata(x0 & 0xFF); // XSTART
280     writedata(x1 >> 8);
281     writedata(x1 & 0xFF); // XEND
282
283     // Y
284     writecommand(ILI9341_PASET); // Row addr set
285     writedata(y0>>8);
286     writedata(y0); // YSTART
287     writedata(y1>>8);
288     writedata(y1); // YEND
289
290     // Écrit dans la mémoire

```

```

291     writecommand(ILI9341_RAMWR); // write to RAM
292 }
293
294 // Fonction qui permet de dessiner des Pixels
295 void tft_drawPixel(short x, short y, unsigned short color) {
296 /* Dessine un pixel dans la localisation voulu (x,y) avec une certaine couleur
297 * Parameters:
298 *      x: x-coordinate of pixel to draw; top left of screen is x=0
299 *          and x increases to the right
300 *      y: y-coordinate of pixel to draw; top left of screen is y=0
301 *          and y increases to the bottom
302 *      color: 16-bit color value
303 * Returns: Nothing
304 */
305
306     if((x < 0) || (x >= _width) || (y < 0) || (y >= _height)) return;
307
308     // Défini l'endroit où écrire
309     tft_setAddrWindow(x,y,x+1,y+1);
310
311     // Masquage pour envoyer en 2x8bits
312     uint8_t hi = color >> 8;
313     uint8_t lo = color;
314
315     D_C_High;// Datacommand high
316     TFT_CS_Low; // Chip Select low
317
318     // MSB
319     PLIB_SPI_BufferWrite(SPI_ID_1, hi);
320     do {
321
322         } while (PLIB_SPI_IsBusy(SPI_ID_1));
323
324     // LSB
325     PLIB_SPI_BufferWrite(SPI_ID_1, lo);
326     do {
327
328         } while (PLIB_SPI_IsBusy(SPI_ID_1));
329
330     TFT_CS_High; // Chip Select high
331 }
332
333
334
335 /* DrawLine(Xa,Ya,Xb,Yb,color);
336  * dessine une ligne entre 2 points A et B
337  * http://www.brackeen.com/vga/shapes.html
338  */
339 void DrawLine(short Xa, short Ya, short Xb, short Yb, unsigned short color)
340 {
341     int dx,dy,sdx,sdy,px,py,dxabs,dyabs,i;
342     float slope;
343
344     dx=Xb-Xa;      /* the horizontal distance of the line */
345     dy=Yb-Ya;      /* the vertical distance of the line */
346     dxabs=abs(dx);
347     dyabs=abs(dy);
348     if(dx==0 ) {tft_drawFastVLine(Xa,Ya,dy,color);return;}
349     if(dy==0 ) {tft_drawFastHLine(Xa,Ya,dx,color);return;}
350     sdx=(dx>0)? 1:-1;
351     sdy=(dy>0)? 1:-1;
352     if (dxabs>dyabs) /* the line is more horizontal than vertical */
353     {
354         slope=(float)dy / (float)dx;
355         for(i=0;i!=dx;i+=sdx)
356         {
357             px=i+Xa;
358             py=slope*i+Ya;
359             tft_drawPixel(px,py,color);
360         }
361     }
362     else /* the line is more vertical than horizontal */
363     {

```

```

364     slope=(float)dx / (float)dy;
365     for(i=0;i!=dy;i+=sdy)
366     {
367         px=slope*i+Xa;
368         py=i+Ya;
369         tft_drawPixel(px,py,color);
370     }
371 }
372
373 }
374
375
376 // Fonction qui dessine des lignes horizontales
377 void tft_drawFastHLine(short x, short y, short w, unsigned short color)
378 {
379     // Rudimentary clipping
380     if((x >= _width) || (y >= _height)) return;
381     if((x+w-1) >= _width) w = _width-x;
382
383     // Défini l'endroit où écrire
384     tft_setAddrWindow(x,y,x+w-1,y);
385
386     D_C_High;// Datacommand high
387     TFT_CS_Low; // Chip Select low
388
389     // Masquage pour envoyer en 2x8bits
390     uint8_t hi = color >> 8;
391     uint8_t lo = color;
392
393     while (w--) {
394         writedata(hi); // MSB
395         writedata(lo); // LSB
396     }
397
398     TFT_CS_High;// Chip Select high
399 }
400
401 // Fonction qui dessine des lignes verticales
402 void tft_drawFastVLine(short x, short y, short h, unsigned short color) {
403
404     // Rudimentary clipping
405     if((x >= _width) || (y >= _height)) return;
406     if((y+h-1) >= _height)
407         h = _height-y;
408
409     // Défini l'endroit où écrire
410     tft_setAddrWindow(x, y, x, y+h-1);
411
412     D_C_High;// Datacommand high
413     TFT_CS_Low;// Chip Select low
414
415     // Masquage pour envoyer en 2x8bits
416     uint8_t hi = color >> 8;
417     uint8_t lo = color;
418
419     while (h--) {
420         writedata(hi); // MSB
421         writedata(lo); // LSB
422     }
423
424     TFT_CS_High;// Chip Select high
425 }
426
427 // Fonction qui permet de dessiner un caractère
428 void drawChar(short x, short y, unsigned char c,
429                 unsigned short fgcolor, unsigned short bgcolor, unsigned short size)
430 {
431     if((x >= _width)           || // Clip right
432        (y >= _height)          || // Clip bottom
433        ((x + 6 * size - 1) < 0) || // Clip left TODO: is this correct?
434        ((y + 8 * size - 1) < 0)) // Clip top TODO: is this correct?
435         return;
436

```

```

437 if (fgcolor == bgcolor) {
438     // This transparent approach is only about 20% faster
439     if (size == 1) {
440         uint8_t mask = 0x01;
441         int16_t xoff, yoff;
442         for (yoff=0; yoff < 8; yoff++) {
443             uint8_t line = 0;
444             for (xoff=0; xoff < 5; xoff++) {
445                 if (font[c * 5 + xoff] & mask) line |= 1;
446                 line <<= 1;
447             }
448             line >>= 1;
449             xoff = 0;
450             while (line) {
451                 if (line == 0x1F) {
452                     tft_drawFastHLine(x + xoff, y + yoff, 5, fgcolor);
453                     break;
454                 } else if (line == 0x1E) {
455                     tft_drawFastHLine(x + xoff, y + yoff, 4, fgcolor);
456                     break;
457                 } else if ((line & 0x1C) == 0x1C) {
458                     tft_drawFastHLine(x + xoff, y + yoff, 3, fgcolor);
459                     line <<= 4;
460                     xoff += 4;
461                 } else if ((line & 0x18) == 0x18) {
462                     tft_drawFastHLine(x + xoff, y + yoff, 2, fgcolor);
463                     line <<= 3;
464                     xoff += 3;
465                 } else if ((line & 0x10) == 0x10) {
466                     tft_drawPixel(x + xoff, y + yoff, fgcolor);
467                     line <<= 2;
468                     xoff += 2;
469                 } else {
470                     line <<= 1;
471                     xoff += 1;
472                 }
473             }
474             mask = mask << 1;
475         }
476     } else {
477         uint8_t mask = 0x01;
478         int16_t xoff, yoff;
479         for (yoff=0; yoff < 8; yoff++) {
480             uint8_t line = 0;
481             for (xoff=0; xoff < 5; xoff++) {
482                 if (font[c * 5 + xoff] & mask) line |= 1;
483                 line <<= 1;
484             }
485             line >>= 1;
486             xoff = 0;
487             while (line) {
488                 if (line == 0x1F) {
489                     tft_fillRect(x + xoff * size, y + yoff * size,
490                         5 * size, size, fgcolor);
491                     break;
492                 } else if (line == 0x1E) {
493                     tft_fillRect(x + xoff * size, y + yoff * size,
494                         4 * size, size, fgcolor);
495                     break;
496                 } else if ((line & 0x1C) == 0x1C) {
497                     tft_fillRect(x + xoff * size, y + yoff * size,
498                         3 * size, size, fgcolor);
499                     line <<= 4;
500                     xoff += 4;
501                 } else if ((line & 0x18) == 0x18) {
502                     tft_fillRect(x + xoff * size, y + yoff * size,
503                         2 * size, size, fgcolor);
504                     line <<= 3;
505                     xoff += 3;
506                 } else if ((line & 0x10) == 0x10) {
507                     tft_fillRect(x + xoff * size, y + yoff * size,
508                         size, size, fgcolor);
509                     line <<= 2;
510                 }
511             }
512         }
513     }
514 }

```

```

510                     xoff += 2;
511             } else {
512                 line <= 1;
513                 xoff += 1;
514             }
515         }
516         mask = mask << 1;
517     }
518 }
519 } else {
520     // This solid background approach is about 5 time faster
521     tft_setAddrWindow(x, y, x + 6 * size - 1, y + 8 * size - 1);
522     uint8_t xr, yr;
523     uint8_t mask = 0x01;
524     uint16_t color;
525     uint8_t hi ;
526     uint8_t lo ;
527     uint8_t bhi = bgcolor >> 8;
528     uint8_t blo = bgcolor;
529     for (y=0; y < 8; y++) {
530         for (yr=0; yr < size; yr++) {
531             for (x=0; x < 5; x++) {
532                 if (font[c * 5 + x] & mask) {
533                     color = fgcolor;
534                 } else {
535                     color = bgcolor;
536                 }
537                 hi = color >> 8;
538                 lo = color;
539                 for (xr=0; xr < size; xr++) {
540
541                     writedata(hi);
542                     writedata(lo);
543                 }
544             }
545             for (xr=0; xr < size; xr++) {
546                 writedata(bhi);
547                 writedata(blo);
548             }
549         }
550         mask = mask << 1;
551     }
552 }
553 }
554
555 // Fonction qui permet de faire une rotation de 90° de l'écran
556 void setRotation(short m) {
557
558     writecommand(ILI9341_MADCTL);
559     short rotation = m % 4; // can't be higher than 3
560     switch (rotation) {
561         case 0: // Écran initial
562             writedata(MADCTL_MX | MADCTL_BGR);
563             _width = ILI9341_TFTWIDTH;
564             _height = ILI9341_TFTHEIGHT;
565         break;
566
567         case 1: // Rotation de l'écran 90°
568             writedata(MADCTL_MV | MADCTL_BGR);
569             _width = ILI9341_TFTHEIGHT;
570             _height = ILI9341_TFTWIDTH;
571         break;
572
573         case 2: // Rotation de l'écran 180°
574             writedata(MADCTL_MY | MADCTL_BGR);
575             _width = ILI9341_TFTWIDTH;
576             _height = ILI9341_TFTHEIGHT;
577         break;
578
579         case 3: // Rotation de l'écran 270°
580             writedata(MADCTL_MX | MADCTL_MY | MADCTL_MV | MADCTL_BGR);
581             _width = ILI9341_TFTHEIGHT;
582             _height = ILI9341_TFTWIDTH;

```

```

583         break;
584     }
585 }
586
587 // Defini l'emplacement du curseur
588 void tft_setCursor(short x, short y) {
589 /* Set cursor for text to be printed
590 * Parameters:
591 *      x = x-coordinate of top-left of text starting
592 *      y = y-coordinate of top-left of text starting
593 * Returns: Nothing
594 */
595
596     cursor_x = x; // en haut à gauche
597     cursor_y = y; // en haut à gauche
598 }
599
600 // Modifie la couleur de la police
601 void tft_setTextColor(unsigned short c) {
602     // For 'transparent' background, we'll set the bg
603     // to the same as fg instead of using a flag
604     textcolor = textbgcolor = c;
605 }
606 // Modifie la couleur ET le background de la police
607 void tft_setTextColor_F(unsigned short fore, unsigned short Back)
608 {
609
610     textcolor = fore;
611     textbgcolor = Back;
612 }
613
614 // Fonction qui permet d'écrire une chaine de caractères
615 void tft_writeString(char* str){
616 /* Call tft_setCursor(), tft_setTextColor(), tft_setTextSize()
617 * as necessary before printing
618 */
619     while (*str){
620         tft_write(*str++);
621     }
622 }
623
624 // Fonction qui permet d'écrire
625 void tft_write(unsigned char c){
626     if (c == '\n') {
627         cursor_y += textszie*8;
628         cursor_x = 0;
629     } else if (c == '\r') {
630         // skip em
631     } else if (c == '\t'){
632         int new_x = cursor_x + 4;
633         if (new_x < _width){
634             cursor_x = new_x;
635         }
636     } else {
637         drawChar(cursor_x, cursor_y, c, textcolor, textbgcolor, textszie);
638         cursor_x += textszie*6;
639         if (wrap && (cursor_x > (_width - textszie*6))) {
640             cursor_y += textszie*8;
641             cursor_x = 0;
642         }
643     }
644 }
645
646 // Modifie la taille du text à afficher
647 void tft_setTextSize(unsigned char s) {
648 /*Set size of text to be displayed
649 * Parameters:
650 *      s = text size (1 being smallest)
651 * Returns: nothing
652 */
653     textszie = (s > 0) ? s : 1;
654 }
655

```

```

1 #ifndef XC_HEADER_TEMPLATE_H
2 #define XC_HEADER_TEMPLATE_H
3 /*-----*/
4 // DriverLCD.h
5 /*-----*/
6 // Description : Utilitaire qui contient les prototypes
7 //                 de fonctions des différentes variables pour
8 //                 l'écran
9 //
10 // Auteur       : Paulo Gomes
11 // Version      : V1.0
12 //
13 /*-----*/
14
15
16
17
18
19
20
21
22 // Variables pour certains calculs
23 unsigned short _width, _height, cursor_y, cursor_x,
24             textcolor, textbgcolor, textsize, wrap;
25
26
27 // Initialisation de l'écran
28 void tft_begin(void);
29
30 // Écriture d'un caractère
31 void tft_write(unsigned char c);
32
33 // Remplissage de l'écran avec une couleur
34 void tft_fillScreen(unsigned short color);
35
36 // Dessine et rempli un rectangle avec une couleur
37 void tft_fillRect(short x, short y, short w, short h,
38                   unsigned short color);
39
40 // Choisi l'endroit où on veut écrire ou dessiner
41 void tft_setAddrWindow(unsigned short x0, unsigned short y0,
42                        unsigned short x1, unsigned short y1);
43
44 // Dessine un pixel
45 void tft_drawPixel(short x, short y, unsigned short color);
46
47 /* DrawLine(Xa,Ya,Xb,Yb,color);
48 * dessine une ligne entre 2 points A et B
49 * http://www.brackeen.com/vga/shapes.html
50 */
51 void DrawLine(short Xa, short Ya, short Xb, short Yb, unsigned short color);
52
53 // Dessine une ligne horizontale
54 void tft_drawFastHLine( short x, short y, short w, unsigned short color);
55
56 // Dessine une ligne verticale
57 void tft_drawFastVLine(short x, short y, short h, unsigned short color);
58
59 // Dessine un caractère
60 void drawChar(short x, short y, unsigned char c,
61               unsigned short fgcolor, unsigned short bgcolor, unsigned short size);
62
63 // Fait une rotation de l'écran de 90° vers la droite (max 4 fois)
64 void setRotation(short m);
65
66 // Définit l'emplacement du curseur
67 void tft_setCursor(short x, short y);
68
69 // Modifie la couleur de la police
70 void tft_setTextColor(unsigned short c);
71
72 void tft_setTextColor_F(unsigned short fore, unsigned short Back);
73 // Modifie la taille du texte à afficher

```

```
74 void tft_setTextSize(unsigned char s);  
75  
76 // Fonction qui permet d'écrire  
77 void tft_write(unsigned char c);  
78  
79 // Fonction qui permet d'écrire une chaîne de caractères  
80 void tft_writeString(char* str);  
81  
82 #endif /* XC_HEADER_TEMPLATE_H */  
83  
84
```

```

1 //18111C_Master
2 /*-----*/
3 // DefineLCD.h
4 /*-----*/
5 // Description : Utilitaire gestion des constantes du LCD
6 //
7 // Auteur      : Paulo Gomes
8 // Version     : V1.0
9 //
10 /*-----*/
11 #ifndef DefineLCD_H
12 #define DefineLCD_H
13 #include "DriverLCD.h"
14
15 // Definition Pins Hardware !
16 /*
17 #define D_C LATDbits.LATD3
18 #define SS LATDbits.LATD1
19 #define BCKL LATDbits.LATD4
20 #define LCDRESET LATDbits.LATD2
21
22 #define LCDRESET_SENS TRISDbits.TRISD2
23 #define D_C_SENS TRISDbits.TRISD3
24 #define SS_SENS TRISDbits.TRISD1
25 #define BCKL_SENS TRISDbits.TRISD4
26 */
27
28 // Defines qui contient les dimensions en pixels de l'écran
29 #define ILI9341_TFTWIDTH 240
30 #define ILI9341_TFTHEIGHT 320
31
32 // Defines pour l'initialisation de l'écran
33 #define ILI9341_NOP 0x00
34 #define ILI9341_SWRESET 0x01
35 #define ILI9341_RDDID 0x04
36 #define ILI9341_RDDST 0x09
37
38 #define ILI9341_SLPIN 0x10
39 #define ILI9341_SLOUT 0x11
40 #define ILI9341_PTLON 0x12
41 #define ILI9341_NORON 0x13
42
43 #define ILI9341_RDMODE 0x0A
44 #define ILI9341_RDMADCTL 0x0B
45 #define ILI9341_RDPIXFMT 0x0C
46 #define ILI9341_RDIMGFMT 0x0D
47 #define ILI9341_RDSELFDIAG 0x0F
48
49 #define ILI9341_INVOFF 0x20
50 #define ILI9341_INVON 0x21
51 #define ILI9341_GAMMASET 0x26
52 #define ILI9341_DISPOFF 0x28
53 #define ILI9341_DISPON 0x29
54
55 #define ILI9341_CASET 0x2A
56 #define ILI9341_PASET 0x2B
57 #define ILI9341_RAMWR 0x2C
58 #define ILI9341_RAMRD 0x2E
59
60 #define ILI9341_PTLAR 0x30
61 #define ILI9341_MADCTL 0x36
62 #define ILI9341_PIXFMT 0x3A
63
64 #define ILI9341_FRMCTR1 0xB1
65 #define ILI9341_FRMCTR2 0xB2
66 #define ILI9341_FRMCTR3 0xB3
67 #define ILI9341_INVCTR 0xB4
68 #define ILI9341_DEFUNCTR 0xB6
69
70 #define ILI9341_PWCTR1 0xC0
71 #define ILI9341_PWCTR2 0xC1
72 #define ILI9341_PWCTR3 0xC2
73 #define ILI9341_PWCTR4 0xC3

```

```

74 #define ILI9341_PWCTR5 0xC4
75 #define ILI9341_VMCTR1 0xC5
76 #define ILI9341_VMCTR2 0xC7
77
78 #define ILI9341_RDID1 0xDA
79 #define ILI9341_RDID2 0xDB
80 #define ILI9341_RDID3 0xDC
81 #define ILI9341_RDID4 0xDD
82
83 #define ILI9341_GMCTR1 0xE0
84 #define ILI9341_GMCTR1N 0xE1
85
86 /*
87 #define ILI9341_PWCTR6 0xFC
88 */
89
90 #define MADCTL_MY 0x80
91 #define MADCTL_MX 0x40
92 #define MADCTL_MV 0x20
93 #define MADCTL_ML 0x10
94 #define MADCTL_RGB 0x00
95 #define MADCTL_BGR 0x08
96 #define MADCTL_MH 0x04
97
98 // Defines des couleurs
99 #define ILI9341_BLACK 0x0000 /* 0, 0, 0 */
100 #define ILI9341_NAVY 0x000F /* 0, 0, 128 */
101 #define ILI9341_DARKGREEN 0x03E0 /* 0, 128, 0 */
102 #define ILI9341_DARKCYAN 0x03EF /* 0, 128, 128 */
103 #define ILI9341_MAROON 0x7800 /* 128, 0, 0 */
104 #define ILI9341_PURPLE 0x780F /* 128, 0, 128 */
105 #define ILI9341_OLIVE 0x7BE0 /* 128, 128, 0 */
106 #define ILI9341_LIGHTGREY 0xC618 /* 192, 192, 192 */
107 #define ILI9341_DARKGREY 0x7BEF /* 128, 128, 128 */
108 #define ILI9341_BLUE 0x001F /* 0, 0, 255 */
109 #define ILI9341_GREEN 0x07E0 /* 0, 255, 0 */
110 #define ILI9341_CYAN 0x07FF /* 0, 255, 255 */
111 #define ILI9341_RED 0xF800 /* 255, 0, 0 */
112 #define ILI9341_MAGENTA 0xF81F /* 255, 0, 255 */
113 #define ILI9341_YELLOW 0xFFE0 /* 255, 255, 0 */
114 #define ILI9341_WHITE 0xFFFF /* 255, 255, 255 */
115 #define ILI9341_ORANGE 0xFD20 /* 255, 165, 0 */
116 #define ILI9341_GREENYELLOW 0xAFE5 /* 173, 255, 47 */
117 #define ILI9341_PINK 0xF81F
118
119
120
121 #endif
122

```

```
1 //-----  
2 // Data_Code.h  
3 //-----  
4 //  
5 //  
6 // Auteur :  
7 // Date :  
8 // Version :  
9 // Modifications : MDS 26.09.2022  
10 // Description :  
11 // Definition des differente commande  
12 //  
13 //  
14 /*-----*/  
15  
16 #ifndef _DATA_CODE_H      /* Guard against multiple inclusion */  
17 #define _DATA_CODE_H  
18  
19 #define ENVOI_TICKET 0x917283bd  
20 #define TICKET_ANNULER 0xccc18ca5  
21  
22 #define TICKET_ACCEPT 0x00e0d135  
23 #define TICKET_REFUSE 0xc5fb3140  
24 #define TICKET_RESET 0xb0bfe0fc  
25 #define BLOCKED 0xcc9ebbb17  
26  
27 #define ARE_U_LINK 0x3103de90  
28 #define I_AM_LINK 0xcd05a45  
29  
30 #define ACK 0x1b1ac4f6  
31  
32  
33  
34 #define ADD_BROADCAST 0xFFFFFFFF  
35  
36  
37     /* Provide C++ Compatibility */  
38 #ifdef __cplusplus  
39 }  
40 #endif  
41  
42 #endif /* _EXAMPLE_FILE_NAME_H */  
43  
44 /* *****  
45 End of File  
46 */  
47
```

```

1 //18112C_Slave
2 //-----
3 // App.c
4 //-----
5 //
6 //
7 // Auteur      :
8 // Date       :
9 // Version     :
10 // Modifications : MDS 26.09.2022
11 // Description :
12 //           Application principal de la carte ticketing Master 1811C
13 //
14 //
15 /*-----*/
16 #include "app.h"
17 #include "GesFifoTh32.h"
18 #include "Mc32gest_RS232.h"
19 #include "Retrieve_name.h"
20 #include "Data_Code.h"
21 #include "Mc32Delays.h"
22
23
24
25
26
27 // ****
28 /* Application Data
29
30 Summary:
31   Holds application data
32
33 Description:
34   This structure holds the application's data.
35
36 Remarks:
37   This structure should be initialized by the APP_Initialize function.
38
39   Application strings and buffers are be defined outside this structure.
40 */
41
42 APP_DATA appData;
43 APP_DATA appData_Old;
44
45
46 bool Btn_tickets = true,Btn_tickets_ON;          //Bouton tickets
47 bool flagTickPressed = false;
48 //uint32_t Name_Student = 0x4D6172696F2044;
49
50 // ****
51 // ****
52 // Section: Application Local Functions
53 // ****
54 // ****
55
56
57 /* TODO: Add any necessary local functions.
58 */
59
60
61 // ****
62 // ****
63 // Section: Application Initialization and State Machine Functions
64 // ****
65 // ****
66
67 // ****
68 Function:
69   void APP_Initialize ( void )
70
71 Remarks:
72   See prototype in app.h.
73 */

```

```

74
75 void APP_Initialize ( void )
76 {
77     /* Place the App state machine in its initial state. */
78     appData.state = APP_STATE_INIT;
79
80
81     /* TODO: Initialize your application's state machine and other
82      * parameters.
83      */
84 }
85 void APP_UpdateState (APP_STATES NewState)
86 {
87     appData.state = NewState;
88 }
89
90 /***** Function: *****
91     void APP_Tasks ( void )
92
93 Remarks:
94     See prototype in app.h.
95 */
96
97
98 void APP_Tasks ( void )
99 {
100     static int Count = 0;
101     int32_t RXSize;
102     char trash;
103     static uint32_t DataCodeToSend = 0;
104     static bool Ticket_Refused = false;
105     U_32 RXData;
106     U_32 ADD_M;
107     U_32 ADD_S;
108
109
110     /* Check the application's current state. */
111     switch ( appData.state )
112     {
113         /* Application's initial state. */
114         case APP_STATE_INIT:
115         {
116
117             RF_Init();
118             InitFifoComm();
119             //start du timer
120             DRV_TMR0_Start();
121             ALL_LED_OFF();
122             //ALL_LED_ON;
123
124
125             //APP_UpdateState(APP_RETRIEVE_NAME);
126             //appData.state = APP_SEND;
127             appData.state = APP_WAIT_FOR_LINK;
128             //appData.state = APP_READY_TO_SEND;
129
130             break;
131         }
132         case APP_RETRIEVE_NAME:
133         {
134
135             ALL_LED_ON();
136             Retrive_Name();
137             if(Name_Receive == true)
138                 APP_UpdateState(APP_WAIT_FOR_LINK);
139                 //Name_Student = atoi( buffReadName);
140                 ALL_LED_OFF();
141
142             break;
143         }
144         case APP_WAIT_FOR_LINK:
145         {
146

```

```

147 //on récupère le message
148 GetMessage(&ADD_M,&ADD_S,&RXData);
149
150 //on vérifie que ca soit bien le maitre qui nous parle
151 if(Message_Broadcast)
152 {
153     Add_Master = ADD_M.val32;
154     Message_Broadcast = false;
155     //On vérifie que le message recu est bien un message de link
156     if(RXData.val32 == ARE_U_LINK)
157     {
158         APP_UpdateState(APP_SEND_ID);
159     }
160     else
161     {
162         APP_UpdateState(APP_ERROR);
163     }
164 }
165 else
166 {
167     APP_UpdateState(APP_ERROR);
168 }
169
170     break;
171 }
172 case APP_SEND_ID:
173 {
174     //on prépare le message de réponse
175     //DataCodeToSend = Name_Student;
176     //stop le timer de clignotement des LEDs
177     DRV_TMR0_Stop();
178     ALL_LED_OFF();
179
180     //envoi du message et de l'adresse du module maître par UART
181     SendMessage(Add_Slave, Add_Master, DataCodeToSend);
182
183     APP_UpdateState(APP_WAIT_FOR_ACK);
184
185
186     appData_Old.state = APP_SEND_ID;
187
188     break;
189 }
190
191 case APP_WAIT_FOR_ACK:
192 {
193
194     //réception du message et de la source
195     GetMessage(&ADD_M,&ADD_S,&RXData);
196
197     //on check que la source est bien le maître
198     if(ADD_M.val32 == Add_Master)
199     {
200         if(ADD_S.val32 == Add_Slave)
201         {
202             if(RXData.val32 == ACK)
203             {
204                 //comme il sagissait d'un envoi de donné
205                 //on regarde quel etat l'as provoqué pour
206                 //ensuite le rediriger au bon état suivant
207                 switch(appData_Old.state)
208                 {
209                     ALL_LED_ON();
210                     case APP_SEND_ID:
211                     {
212                         ALL_LED_OFF();
213                         DRV_TMR0_Stop();
214                         APP_UpdateState(APP_WAIT_FOR_TICKET);
215                         appData_Old.state = APP_WAIT_FOR_ACK;
216                         break;
217                     }
218                     case APP_READY_TO_SEND:
219                     {

```

```

220 //                         LED_WAIT_OFF;
221 //                         appData_Old.state = APP_WAIT_FOR_ACK;
222 //                         appData.state = APP_WAIT_FOR_TICKET_ACCEPT;
223 //                         break;
224 //
225 //                         case APP_WAIT_FOR_TICKET_ACCEPT:
226 //                         {
227 //                             LED_WAIT_OFF;
228 //                             appData.state = APP_READY_TO_SEND;
229 //                             break;
230 //                         }
231 //                     }
232 //                 }
233 //             }
234 //         else
235 //         {
236 //             APP_UpdateState(APP_ERROR);
237 //         }
238 //     }
239 //
240     break;
241 }
242
243 case APP_WAIT_FOR_TICKET:
244 {
245 //On allume la LED verte
246 Led_ReadyOn();
247 if(appButtons.Btn_Tickets)
248 {
249 //on prépare l'envoi du ticket
250 DataCodeToSend = ENVOI_TICKET;
251 //LED verte éteinte
252 ALL_LED_OFF();
253 //LED orange allumée
254 Led_SendedOn();
255 APP_UpdateState(APP_SEND_DATA);
256 appData_Old.state = APP_WAIT_FOR_TICKET;
257 }
258
259
260 RXSize = GetReadSize(&descrFifoRX);
261 if(RXSize >= 8)
262 {
263 //réception du message et de la source
264 GetMessage(&ADD_M,&ADD_S,&RXData);
265 //On attend que l'utilisateur appuie sur le bouton
266 if(ADD_M.val32 == Add_Master)
267 {
268     if(ADD_S.val32 == Add_Slave)
269     {
270         if(RXData.val32 == ARE_U_LINK)
271         {
272             APP_UpdateState(APP_SEND_ID);
273             appData_Old.state = APP_WAIT_FOR_TICKET;
274         }
275     }
276 }
277 }
278
279 }
280 break;
281
282
283 }
284
285
286
287
288
289
290
291
292

```

```

293
294
295     case APP_SEND_DATA:
296     {
297         //envoi du message et de l'adresse du module maitre par UART
298         SendMessage(Add_Slave, Add_Master, DataCodeToSend);
299         APP_UpdateState(APP_WAIT_FOR_TICKET_ACCEPT);
300         appData_Old.state = APP_SEND_DATA;
301
302         break;
303     }
304
305
306     case APP_WAIT_FOR_TICKET_ACCEPT:
307     {
308         //Led d'attente
309         Led_SendedOn();
310         //on regarde si on recois un message via l'UART
311
312         //reception du message
313         GetMessage(&ADD_M,&ADD_S,&RXData);
314         //check si la source est bien le maitre
315         if(ADD_M.val32 == Add_Master)
316         {
317             if(ADD_S.val32 == Add_Slave)
318             {
319                 //ici on regarde l'info qui nous a été retourné
320                 //et selon la réponse retournée et redirige sur les
321                 //différents états
322                 if(RXData.val32 == TICKET_ACCEPT)
323                 {
324                     APP_UpdateState(APP_ACCEPT);
325                 }
326                 else if(RXData.val32 == TICKET_REFUSE)
327                 {
328                     Ticket_Refused = true;
329                     APP_UpdateState(APP_REFUSED);
330                 }
331                 else if(RXData.val32 == BLOCKED)
332                 {
333                     APP_UpdateState(APP_BLOCKED);
334                 }
335                 else if(RXData.val32 == TICKET_RESET)
336                 {
337                     APP_UpdateState(APP_RESET);
338                 }
339                 else
340                 {
341                     APP_UpdateState(APP_ERROR);
342                 }
343             }
344         }
345     }
346     //si l'utilisateur appuie longtemps sur le bouton
347     //il générera une annulation du ticket
348     if(PLIB_PORTS_PinGet(PORTS_ID_0,PORT_CHANNEL_B,PORTS_BIT_POS_7))
349     {
350         Count++;
351         if(Count >= 5000)
352         {
353             DataCodeToSend = TICKET_ANNULLER;
354             APP_UpdateState(APP_SEND_DATA);
355             appData_Old.state = APP_WAIT_FOR_TICKET_ACCEPT;
356         }
357     }
358     else
359     {
360         Count = 0;
361     }
362     break;
363 }
364
365

```

```

366
367
368
369
370
371
372
373
374     case APP_ACCEPT:
375     {
376         //si le ticket a accepté
377         //on fait clignoté la led verte
378         //et on retourne dans le ready to send
379         Led_SendedOff();
380         Blink_LED_ACC();
381         APP_UpdateState(APP_WAIT_FOR_TICKET);
382         break;
383     }
384
385     case APP_REFUSED:
386     {
387         //si le ticket a été refusé
388         //on allume la led rouge
389         //et on bloque l'envoi de ticket pendant un moment
390         //débloque via le temps ou le reset de ticket
391         Led_Link_LostOn();
392
393         DRV_TMR1_Start();
394         if(Ticket_Refused == false) //débloqué par le timer
395         {
396             DRV_TMR1_Stop();
397             Led_Link_LostOff();
398             APP_UpdateState(APP_WAIT_FOR_TICKET);
399         }
400         //Reception du message de reset
401         RXSize = GetReadSize(&descrFifoRX);
402         if(RXSize >= 8)
403         {
404             GetMessage(&ADD_M,&ADD_S,&RXData);
405             if(ADD_M.val32 == Add_Master)
406             {
407                 if(RXData.val32 == TICKET_RESET)
408                 {
409                     Led_Link_LostOff();
410                     APP_UpdateState(APP_WAIT_FOR_TICKET);
411                 }
412             }
413         }
414         break;
415     }
416
417
418     case APP_ERROR:
419     {
420         //vide le FIFO
421         RXSize = GetReadSize(&descrFifoRX);
422         {
423             while (RXSize > 0)
424             {
425                 GetCharFromFifo(&descrFifoRX, &trash);
426                 RXSize--;
427             }
428             APP_UpdateState(APP_WAIT_FOR_LINK);
429         }
430         break;
431     }
432
433     case APP_RESET:
434     {
435         //Vide le FIFO
436         RXSize = GetReadSize(&descrFifoRX);
437         {
438             while (RXSize > 0)

```

```

439         {
440             GetCharFromFifo(&descrFifoRX, &trash);
441             RXSize--;
442         }
443         APP_UpdateState(APP_WAIT_FOR_TICKET);
444     }
445     break;
446 }
447
448 /* TODO: implement your application state machine.*/
449
450
451     /* The default state should never be executed. */
452     default:
453     {
454         /* TODO: Handle error in application's state machine. */
455         break;
456     }
457 }
458
459
460
461 void __ISR(_CHANGE_NOTICE_VECTOR, ipl3AUTO) _IntHandlerChangeNotification(void)
462 {
463
464     //Bouton DECLINE
465     if (PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_7))
466     {
467         flagTickPressed = true;
468     }
469     if (flagTickPressed)
470     {
471         if (!PLIB_PORTS_PinGet (PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_7))
472         {
473             appButtons.Btn_Tickets = true;
474             flagTickPressed = false;
475         }
476     }
477
478     PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_CHANGE_NOTICE_B);
479 }
480
481 void Blink_LED_ACC (void)
482 {
483     int I;
484     ALL_LED_OFF;
485     for(I = 0; I < 5000; I++)
486     {
487
488     }
489     ALL_LED_ON;
490     for(I = 0; I < 5000; I++)
491     {
492
493     }
494     ALL_LED_OFF;
495 }
496
497 void ALL_LED_ON ()
498 {
499     Led_ReadyOn();
500     Led_SendedOn();
501     Led_Link_LostOn();
502 }
503 void ALL_LED_OFF ()
504 {
505     Led_ReadyOff();
506     Led_SendedOff();
507     Led_Link_LostOff();
508 }
509
510 ****
511 End of File

```



```

1 ****
2 MPLAB Harmony Application Header File
3
4 Company:
5 Microchip Technology Inc.
6
7 File Name:
8 app.h
9
10 Summary:
11 This header file provides prototypes and definitions for the application.
12
13 Description:
14 This header file provides function prototypes and data type definitions for
15 the application. Some of these are required by the system (such as the
16 "APP_Initialize" and "APP_Tasks" prototypes) and some of them are only used
17 internally by the application (such as the "APP_STATES" definition). Both
18 are defined here for convenience.
19 ****
20
21 //DOM-IGNORE-BEGIN
22 ****
23 Copyright (c) 2013-2014 released Microchip Technology Inc. All rights reserved.
24
25 Microchip licenses to you the right to use, modify, copy and distribute
26 Software only when embedded on a Microchip microcontroller or digital signal
27 controller that is integrated into your product or third party product
28 (pursuant to the sublicense terms in the accompanying license agreement).
29
30 You should refer to the license agreement accompanying this Software for
31 additional information regarding your rights and obligations.
32
33 SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND,
34 EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF
35 MERCHANTABILITY, TITLE, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE.
36 IN NO EVENT SHALL MICROCHIP OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER
37 CONTRACT, NEGLIGENCE, STRICT LIABILITY, CONTRIBUTION, BREACH OF WARRANTY, OR
38 OTHER LEGAL EQUITABLE THEORY ANY DIRECT OR INDIRECT DAMAGES OR EXPENSES
39 INCLUDING BUT NOT LIMITED TO ANY INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE OR
40 CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT OF
41 SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES
42 (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS.
43 ****
44 //DOM-IGNORE-END
45
46 #ifndef _APP_H
47 #define _APP_H
48
49 // ****
50 // ****
51 // Section: Included Files
52 // ****
53 // ****
54
55 #include <stdint.h>
56 #include <stdbool.h>
57 #include <stddef.h>
58 #include <stdlib.h>
59 #include "system_config.h"
60 #include "system_definitions.h"
61
62 #include "Data_Code.h"
63
64
65 // DOM-IGNORE-BEGIN
66 #ifdef __cplusplus // Provide C++ Compatibility
67
68 extern "C" {
69
70 #endif
71 // DOM-IGNORE-END
72 //todo for SSOP
73
```

```

74
75 // ****
76 // ****
77 // Section: Type Definitions
78 // ****
79 // ****
80 extern char buffReadName[];// Buffer de reception de l'UART
81 extern uint8_t Name_Receive ;// Flag de réception
82 extern uint8_t countCar;// Compteur du nombre de characters d'un nom
83 extern bool Ticket_Refused;
84 /* Application states
85
86 Summary:
87 Application states enumeration
88
89 Description:
90 This enumeration defines the valid application states. These states
91 determine the behavior of the application at various times.
92 */
93
94 typedef enum
95 {
96     /* Application's state machine's initial state. */
97     APP_STATE_INIT=0,
98     APP_RETRIEVE_NAME,
99     APP_WAIT_FOR_LINK,
100    APP_SEND_DATA,
101    APP_SEND_ID,
102    APP_ERROR,
103    APP_WAIT_FOR_ACK,
104    APP_WAIT_FOR_TICKET_ACCEPT,
105    APP_ACCEPT,
106    APP_REFUSED,
107    APP_BLOCKED,
108    APP_RESET,
109    APP_WAIT_FOR_TICKET,
110
111    /* TODO: Define states used by the application state machine. */
112
113 } APP_STATES;
114 typedef struct
115 {
116     /* The application's current state */
117     bool Btn_Tickets;
118     /* TODO: Define any additional data used by the application. */
119
120 } APP_BUTTON;
121 APP_BUTTON appButtons;
122
123 // ****
124 /* Application Data
125
126 Summary:
127 Holds application data
128
129 Description:
130 This structure holds the application's data.
131
132 Remarks:
133 Application strings and buffers are be defined outside this structure.
134 */
135
136 typedef struct
137 {
138     /* The application's current state */
139     APP_STATES state;
140     /* TODO: Define any additional data used by the application. */
141
142 } APP_DATA;
143
144
145 // ****
146 // ****

```

```

147 // Section: Application Callback Routines
148 // ****
149 // ****
150 /* These routines are called by drivers when certain events occur.
151 */
152
153 // ****
154 // ****
155 // Section: Application Initialization and State Machine Functions
156 // ****
157 // ****
158
159 /*****
160     Function:
161     void APP_Initialize ( void )
162
163     Summary:
164         MPLAB Harmony application initialization routine.
165
166     Description:
167         This function initializes the Harmony application. It places the
168         application in its initial state and prepares it to run so that its
169         APP_Tasks function can be called.
170
171     Precondition:
172         All other system initialization routines should be called before calling
173         this routine (in "SYS_Initialize").
174
175     Parameters:
176         None.
177
178     Returns:
179         None.
180
181     Example:
182         <code>
183             APP_Initialize();
184         </code>
185
186     Remarks:
187         This routine must be called from the SYS_Initialize function.
188 */
189
190 void APP_Initialize ( void );
191 void Blink_LED_ACC (void);
192 void APP_UpdateState (APP_STATES NewState);
193 void ALL_LED_ON (void);
194 void ALL_LED_OFF (void);
195
196 /*****
197     Function:
198     void APP_Tasks ( void )
199
200     Summary:
201         MPLAB Harmony Demo application tasks function
202
203     Description:
204         This routine is the Harmony Demo application's tasks function. It
205         defines the application's state machine and core logic.
206
207     Precondition:
208         The system and application initialization ("SYS_Initialize") should be
209         called before calling this.
210
211     Parameters:
212         None.
213
214     Returns:
215         None.
216
217     Example:
218         <code>
219             APP_Tasks();

```

```
220     </code>
221
222     Remarks:
223     This routine must be called from SYS_Tasks() routine.
224 */
225
226 void APP_Tasks( void );
227
228
229 #endif /* _APP_H */
230
231 //DOM-IGNORE-BEGIN
232 #ifdef __cplusplus
233 }
234#endif
235 //DOM-IGNORE-END
236
237 /***** End of File ****/
238 */
239
240
241
```

```

1 //18112C_Slave
2 //
3 //Mc32Gest_RS232.C
4 // Canevas manipulatio TP2 RS232 SLO2 2016-2017
5 // Fonctions d'émission et de réception des message
6 // CHR 20.12.2016 ajout traitement int error
7 // CHR 22.12.2016 evolution des marquers observation int Usart
8 // MDS 26.09.2022 Modification pour permettre la communication avec le Xbee et la
9 //gestion des donnee reçus
10
11 #include <xc.h>
12 #include <sys/attribs.h>
13 #include <stdint.h>
14 #include "system_definitions.h"
15 // Ajout CHR
16 #include "app.h"
17 #include "Mc32gest_RS232.h"
18 #include <GenericTypeDefs.h>
19 //Ajout MDS
20 #include "Retrieve_name.h"
21 #include "RF.h"
22 #include "Mc32Delays.h"
23 #include "Data_Code.h"
24 #include "GesFifoTh32.h"
25
26 //definition du byte de fin de trame
27 #define END 0xBB
28
29 //definition du byte de debut de trame
30 #define START 0xAA
31
32
33 // Structure décrivant le message (version 2016)
34
35
36 typedef union
37 {
38     uint32_t val32;
39
40     struct
41     {
42         uint8_t msb;
43         uint8_t byte1;
44         uint8_t byte2;
45         uint8_t lsb;
46     }
47     sh1;
48 }
49 U_manip32;
50
51 typedef struct {
52     uint8_t Start_First;
53     uint8_t Start;
54     U_32 Add_Master;
55     U_32 Add_Slave;
56     U_32 Data;
57     uint8_t End;
58     char Name [20];
59 }
60 StruMess;
61
62 // Struct pour émission des messages
63 StruMess TxMess;
64 // Struct pour réception des messages
65 StruMess RxMess;
66
67 // Declaration des FIFO pour réception et émission
68 #define FIFO_RX_SIZE ( (63*7) + 1) // 63 messages
69 #define FIFO_TX_SIZE ( (63*7) + 1) // 63 messages
70
71 int8_t fifoRX[FIFO_RX_SIZE];
72 // Déclaration du descripteur du FIFO de réception

```

```

73 S_fifo descrFifoRX;
74
75
76 int8_t fifoTX[FIFO_TX_SIZE];
77 // Declaration du descripteur du FIFO d'émission
78 S_fifo descrFifoTX;
79
80 uint32_t Add_Slave = 0;
81 uint32_t Add_Master = 0;
82
83 bool Message_receive;
84 bool Message_Broadcast = false;
85 // Initialisation de la communication serial
86 // -----
87
88 void InitFifoComm(void)
89 {
90
91     // Initialisation du fifo de reception
92     InitFifo (&descrFifoRX, FIFO_RX_SIZE, fifoRX, 0 );
93     // Initialisation du fifo d'émission
94     InitFifo (&descrFifoTX, FIFO_TX_SIZE, fifoTX, 0 );
95
96 } // InitComm
97
98
99 bool GetMessage(U_32 *pAdd_M,U_32 *pAdd_S, U_32 *pDataS)
100 {
101
102     bool startOk = false;
103     static int CommStatus = 0;
104
105     int8_t CarLu,i=0,Car_Start_Trame = 0;
106
107     RxMess.End = END;
108
109     // Traitement de reception à introduire ICI
110     if(Get_Add_Slave)
111     {
112
113         uint8_t* dstArray ;
114
115         SYS_INT_SourceDisable(INT_SOURCE_USART_1_RECEIVE); //désactive int uart rx
116
117         while(GetReadSize(&descrFifoRX) > 0)
118         {
119
120             GetCharFromFifo (&descrFifoRX, (int8_t*)&CarLu);
121             dstArray[i] = CarLu;
122             i++;
123         }
124         Get_Add_Slave = false;
125         Add_Slave = (uint32_t)dstArray;
126
127         SYS_INT_SourceEnable(INT_SOURCE_USART_1_RECEIVE); //réactive int uart rx
128     }
129     else
130     {
131         //vérifie longueur message et présence start
132         // trame 14 byte min:
133         // 1 byte de start
134         // 4 byte d'adresse de l'expediteur
135         // 4 byte d'adresse de destinataire
136         // 4 byte de donnee
137         // 1 byte de fin
138         while((GetReadSize(&descrFifoRX)) >= 14)
139         {
140             GetCharFromFifo (&descrFifoRX, &CarLu);
141             if (CarLu == (int8_t)0xAA)
142             {
143                 startOk = true;
144
145                 break;

```

```

146
147 }
148 //Start trouvé, lire message et décoder
149 if (startOk)
{
150
151     //prendre de la fifo les 4 byte de l'expéditeur
152     GetCharFromFifo (&descrFifoRX, &CarLu);
153     pAdd_M->U_32_Bytes.msb = CarLu;
154     GetCharFromFifo (&descrFifoRX, &CarLu);
155     pAdd_M->U_32_Bytes.byte1 = CarLu;
156     GetCharFromFifo (&descrFifoRX, &CarLu);
157     pAdd_M->U_32_Bytes.byte2 = CarLu;
158     GetCharFromFifo (&descrFifoRX, &CarLu);
159     pAdd_M->U_32_Bytes.lsb = CarLu;
160
161
162     //On vérifie si l'adresse de l'expéditeur est un broadcast
163     if(pAdd_M->val32 == ADD_BROADCAST)
{
164         //prendre de la fifo les 4 byte de l'expéditeur en écrasant le
165         //broadcast
166         Message_Broadcast = true;
167         GetCharFromFifo (&descrFifoRX, &CarLu);
168         pAdd_M->U_32_Bytes.msb = CarLu;
169         GetCharFromFifo (&descrFifoRX, &CarLu);
170         pAdd_M->U_32_Bytes.byte1 = CarLu;
171         GetCharFromFifo (&descrFifoRX, &CarLu);
172         pAdd_M->U_32_Bytes.byte2 = CarLu;
173         GetCharFromFifo (&descrFifoRX, &CarLu);
174         pAdd_M->U_32_Bytes.lsb = CarLu;
175
176     }
177     else
178     {
179         //prendre de la fifo les 4 byte du destinataire
180         GetCharFromFifo (&descrFifoRX, &CarLu);
181         pAdd_S->U_32_Bytes.msb = CarLu;
182         GetCharFromFifo (&descrFifoRX, &CarLu);
183         pAdd_S->U_32_Bytes.byte1 = CarLu;
184         GetCharFromFifo (&descrFifoRX, &CarLu);
185         pAdd_S->U_32_Bytes.byte2 = CarLu;
186         GetCharFromFifo (&descrFifoRX, &CarLu);
187         pAdd_S->U_32_Bytes.lsb = CarLu;
188     }
189
190
191     //prendre de la fifo les 4 byte de datas
192     GetCharFromFifo (&descrFifoRX, &CarLu);
193     pDatas->U_32_Bytes.msb = CarLu;
194     GetCharFromFifo (&descrFifoRX, &CarLu);
195     pDatas->U_32_Bytes.byte1 = CarLu;
196     GetCharFromFifo (&descrFifoRX, &CarLu);
197     pDatas->U_32_Bytes.byte2 = CarLu;
198     GetCharFromFifo (&descrFifoRX, &CarLu);
199     pDatas->U_32_Bytes.lsb = CarLu;
200
201
202     //prendre de la fifo le byte de fin
203     GetCharFromFifo (&descrFifoRX, &CarLu);
204
205     //on vérifie si on a bien eu le byte de fin
206     if (CarLu == RxMess.End)
{
207         startOk = true;
208     }
209     else
{
210         startOk = false;
211     }
212
213     //si les data était un are_you_link
214     if(pDatas->val32 == ARE_U_LINK)
{
215         //Changement d'état
216         APP_UpdateState(APP_WAIT_FOR_LINK);
217

```

```

218         }
219     }
220 }
221 return startOk;
222 } // GetMessageFromMessage
223
224
225
226 // Fonction d'envoi des message
227 void SendMessage(uint32_t Add_S,uint32_t Add_M, uint32_t Datas)
228 {
229     static uint8_t First_Transmit = true;
230     int8_t FreeSize,i;
231     // Gestion du control de flux
232     TxMess.Start = 0xAA;
233     TxMess.End = 0xBB;
234     TxMess.Add_Master.val32 = Add_M;
235     TxMess.Add_Slave.val32 = Add_S;
236     TxMess.Data.val32 = Datas;
237     //TxMess.Name = buffReadName;
238
239     //vérifie longueur disponible dans le fifo
240     // trame 14 byte min:
241     // 1 byte de start
242     // 4 byte d'adresse de l'expéditeur
243     // 4 byte d'adresse de destinataire
244     // 4-20 byte de donnée (depend du nom de l'utilisateur)
245     // 1 byte de fin
246     if (GetWriteSpace(&descrFifoTX) >= (10 + countCar))
247     {
248         // on met le byte de début de trame dans le fifo
249         PutCharInFifo (&descrFifoTX, TxMess.Start);
250         // on met l'adresse de l'expéditeur dans le fifo
251         PutCharInFifo (&descrFifoTX, TxMess.Add_Slave.U_32_Bytes.msb);
252         PutCharInFifo (&descrFifoTX, TxMess.Add_Slave.U_32_Bytes.byte1);
253         PutCharInFifo (&descrFifoTX, TxMess.Add_Slave.U_32_Bytes.byte2);
254         PutCharInFifo (&descrFifoTX, TxMess.Add_Slave.U_32_Bytes.lsb);
255
256         // on met l'adresse du destinataire dans le fifo
257         PutCharInFifo (&descrFifoTX, TxMess.Add_Master.U_32_Bytes.msb);
258         PutCharInFifo (&descrFifoTX, TxMess.Add_Master.U_32_Bytes.byte1);
259         PutCharInFifo (&descrFifoTX, TxMess.Add_Master.U_32_Bytes.byte2);
260         PutCharInFifo (&descrFifoTX, TxMess.Add_Master.U_32_Bytes.lsb);
261
262         //on vérifie si c'est le premier envoie de donnée
263         if(First_Transmit)
264         {
265             for(i = 0 ; i <= countCar - 1 ; i++)
266             {
267                 //on met les different byte dans le fifo
268                 PutCharInFifo (&descrFifoTX, buffReadName[i]);
269
270             }
271             //on enleve le flag de premiere envoie
272             First_Transmit = false;
273         }
274         else
275         {
276             // on met les datas dans le fifo
277             PutCharInFifo (&descrFifoTX, TxMess.Data.U_32_Bytes.msb);
278             PutCharInFifo (&descrFifoTX, TxMess.Data.U_32_Bytes.byte1);
279             PutCharInFifo (&descrFifoTX, TxMess.Data.U_32_Bytes.byte2);
280             PutCharInFifo (&descrFifoTX, TxMess.Data.U_32_Bytes.lsb);
281         }
282         PutCharInFifo (&descrFifoTX, TxMess.End);
283     }
284
285     PLIB_INT_SourceEnable(INT_ID_0, INT_SOURCE_USART_1_TRANSMIT);
286 }
287
288 // !!!!!!!
289 // Attention ne pas oublier de supprimer la réponse générée dans system_interrupt
290 // !!!!!!!

```

```

291
292 void __ISR(_UART_1_VECTOR, ipl5AUTO) _IntHandlerDrvUsartInstance0(void)
293 {
294     USART_ERROR UsartStatus;
295     int8_t Carac, TXsize, TxBuffFull;
296     // Is this an Error interrupt ?
297     if ( PLIB_INT_SourceFlagGet(INT_ID_0, INT_SOURCE_USART_1_ERROR) &&
298         PLIB_INT_SourceIsEnabled(INT_ID_0, INT_SOURCE_USART_1_ERROR) ) {
299         /* Clear pending interrupt */
300         PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_USART_1_ERROR);
301         // Traitement de l'erreur à la réception.
302     }
303
304
305     // Is this an RX interrupt ?
306     if ( PLIB_INT_SourceFlagGet(INT_ID_0, INT_SOURCE_USART_1_RECEIVE) &&
307         PLIB_INT_SourceIsEnabled(INT_ID_0, INT_SOURCE_USART_1_RECEIVE) ) {
308
309         // Oui Test si erreur parité ou overrun
310         UsartStatus = PLIB_USART_ErrorsGet(USART_ID_1);
311         if ( (UsartStatus & (USART_ERROR_PARITY |
312             USART_ERROR_FRAMING | USART_ERROR_RECEIVER_OVERRUN)) == 0 )
313     {
314
315         while(PLIB_USART_ReceiverDataIsAvailable(USART_ID_1))
316     {
317         //Led_ReadyToggle();
318         Carac = PLIB_USART_ReceiverByteReceive(USART_ID_1);
319         PutCharInFifo(&descrFifoRX, Carac);
320     }
321         //Message_receive = true;
322         // buffer is empty, clear interrupt flag
323         PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_USART_1_RECEIVE);
324     } else {
325         // Suppression des erreurs
326         // La lecture des erreurs les efface sauf pour overrun
327         if ( (UsartStatus & USART_ERROR_RECEIVER_OVERRUN) ==
328             USART_ERROR_RECEIVER_OVERRUN) {
329             PLIB_USART_ReceiverOverrunErrorClear(USART_ID_1);
330         }
331         //Led_ReadyOff();
332     } // end if RX
333
334
335
336     // Is this an TX interrupt ?
337     if ( PLIB_INT_SourceFlagGet(INT_ID_0, INT_SOURCE_USART_1_TRANSMIT) &&
338         PLIB_INT_SourceIsEnabled(INT_ID_0, INT_SOURCE_USART_1_TRANSMIT) ) {
339
340         TXsize = GetReadSize(&descrFifoTX);
341         TxBuffFull = PLIB_USART_TransmitterBufferIsFull(USART_ID_1);
342         if ((TXsize > 0)&& (TxBuffFull == false))
343     {
344         do//Faire la boucle tant que le message n'est pas envoyé
345     {
346         //Led_SendedToggle();
347         //On va chercher les valeurs a envoyer
348         GetCharFromFifo(&descrFifoTX, &Carac);
349         PLIB_USART_TransmitterByteSend(USART_ID_1, Carac);
350         TXsize = GetReadSize(&descrFifoTX);
351         TxBuffFull = PLIB_USART_TransmitterBufferIsFull(USART_ID_1);
352
353     }while((TXsize > 0)&& (TxBuffFull == false));
354
355     // Clear the TX interrupt Flag (Seulement apres TX)
356     PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_USART_1_TRANSMIT);
357     TXsize = GetReadSize(&descrFifoTX);
358     if (TXsize == 0)
359     {
360         //Pour eviter les interruption inutile
361         PLIB_INT_SourceDisable(INT_ID_0, INT_SOURCE_USART_1_TRANSMIT);
362     }

```

```
363     }
364     else
365     {
366         // disable TX interrupt (pour éviter une interrupt. inutile si plus
367         // rien à transmettre)
368         PLIB_INT_SourceDisable(INT_ID_0, INT_SOURCE_USART_1_TRANSMIT);
369     }
370 }
371
372
373
374 }
```

```

1 #ifndef Mc32Gest_RS232_H
2 #define Mc32Gest_RS232_H
3 /*-----
4 // Mc32Gest_RS232.h
5 -----*/
6 // Description : emission et reception spécialisée
7 // pour TP2 2016-2017
8 //
9 // Auteur : C. HUBER
10 //
11 // Version : V1.3
12 // Compilateur : XC32 V1.42 + Harmony 1.08
13 //
14 /*-----*/
15
16 #include <stdint.h>
17 #include <stdbool.h>
18 #include "GesFifoTh32.h"
19
20 typedef union
21 {
22     uint32_t val32;
23
24     struct
25     {
26         uint8_t msb;
27         uint8_t byte1;
28         uint8_t byte2;
29         uint8_t lsb;
30     }
31     U_32_BytEs;
32 }
33 U_32;
34
35 /*-----
36 // Définition des fonctions prototypes
37 -----*/
38
39 // prototypes des fonctions
40 void InitFifoComm(void);
41
42
43 // Descripteur des fifos
44 extern S_fifo descrFifoRX;
45 extern S_fifo descrFifoTX;
46 extern bool Message_receive;
47 extern bool Message_Broadcast;
48
49 bool GetMessage(U_32 *pAdd_M, U_32 *pAdd_S, U_32 *pDatas);
50 void SendMessage(uint32_t ADD_M, uint32_t Add_S, uint32_t Datas);
51
52 #endif
53

```

```

1  ****
2 // Project      : 2126 Affichage Matriciel Nom Etudiant
3 // Author       : Ricardo Crespo
4 // Date         : 12.06.2022
5 // Description   : Fichier du code principal
6 //
7 // Modification: Mario Dos Santos
8 // Date        : 31.08.2022
9 // Description : Modifier le fichier pour n'avoir que la partie lecture de nom.
10 //                  pour pouvoir l'utiliser dans le projet 1811C Ticketing
11 //                  (Supprimer la partie Matrix, I2C et animation)
12 //
13 ****
14 #include "Retrieve_name.h"
15 #include "app.h"
16
17
18 // ****
19 // ****
20 // Section: Global Data Definitions
21 // ****
22 // ****
23
24
25 uint8_t countCar = 0;           // Compteur du nombre de characters d'un nom
26 char buffReadName[20] = {'M','a','r','i','o',' ','D'};// Buffer de reception de l'UART
27
28 bool Name_Receive = false;    //Flag Nom recu
29
30 ****
31 Function:
32     void Retrive_Name ( void )
33
34 Remarks:
35     See prototype in Retrive_Name.h.
36 */
37 void Retrive_Name ( void )
38 {
39
40     static uint8_t numberChar = 0;           // Compteur de la taille de la clé d'envoi
41
42     char key[] = {'C', '\0'};                // Clé de confirmaiton pour le software
43     char keyCom = 'x';                     // Clé de communication de la part du
44     software
45     char keyEndName[] = {'X', 'D', 'R'};    // Clé de fin de nom complet
46     char character;                      // Buffeur du character actuellement lu
47     char reciveCharacter = ' ' ;          // Buffeur de réception des caracteres de
48     l'UART
49
50
51
52
53
54 // Tant que l'on reçois des datas dans le RX buffeur
55 while(PLIB_USART_ReceiverDataIsAvailable(USART_ID_2))
56 {
57     // Récupération du caractere depuis le RX buffeur
58
59     character = PLIB_USART_ReceiverByteReceive(USART_ID_2);
60
61
62
63 // Si on a pas reçu la clé de communication et qu'on a pas encore lu le nom
64 if((character != keyCom) && (!startReadName))
65 {
66     // Reset du compteur du nombre de caracteres du nom
67     countCar = 0;
68     // Peut démarrer la lecture du nom complete
69     startReadName = true;
70 }
71 // Si non on stock la clé de communication avec le software

```

```

72
73
74     // Sauvegarde de la clé de communication avec le software
75     receiveCharacter = character;
76 }
77
78 // Si on peut faire la lecture du nom complète
79 if(startReadName)
80 {
81     // Stockage du caractère actuel dans le buffer de stockage du nom
82     buffReadName[countCar] = character;
83     // Incrémentation du nombre de caractères du nom
84     countCar++;
85 }
86
87
88 // Si on a reçu la clé du Software ET que l'on lit pas le nom
89 if((receiveCharacter == keyCom) && !startReadName)
90 {
91     // Tant qu'on a pas fini la chaîne ET que l'on a pas plus de 8 caractères
92     while ((key[numberChar] != 0) && (numberChar < SIZE_KEY))
93     {
94         // Attente que le TX buffer soit disponible
95         while(PLIB_USART_TransmitterBufferIsFull(USART_ID_2));
96         // Envoi de la clé de confirmation au software
97         PLIB_USART_TransmitterByteSend(USART_ID_2, key[numberChar]);
98         // Incrémentation du compteur de nombre de caractères
99         numberChar++;
100    }
101    // Reset du buffer de réception des caractères
102    receiveCharacter = ' ';
103    // Reset du compteur de nombre de caractères
104    numberChar = 0;
105 }
106
107 // Si on a reçu un nom d'élève
108 if((buffReadName[0] != 0x20)
109 && (buffReadName[0] != NULL)
110 && (buffReadName[0] != keyCom)
111 && (buffReadName[countCar - 3] == keyEndName[0])
112 && (buffReadName[countCar - 2] == keyEndName[1])
113 && (buffReadName[countCar - 1] == keyEndName[2]))
114 {
115     // Enlève les positions de la clé de réception du nom
116     countCar -= (END_NAME_KEY_SIZE - 1);
117     // Insère un '.' après la lettre majuscule du nom de l'élève
118     Name_Receive = true;
119 }
120
121 }
122
123 ****
124 End of File
125 */
126

```

```

1  ****
2 // Project      : 2126 Affichage Matriciel Nom Etudiant
3 // Author       : Ricardo Crespo
4 // Date        : 12.06.2022
5 // Description  : Header du code principal
6 //
7 // Modification: Mario Dos Santos
8 // Date        : 31.08.2022
9 // Description : Modifier le fichier pour n'avoir que la partie lecture de nom.
10 //              pour pouvoir l'utiliser dans le projet 1811C Ticketing
11 //              (Supprimer la partie Matrix et I2C)
12 //
13 ****
14
15 ****
16 MPLAB Harmony Application Header File
17
18 Company:
19     Microchip Technology Inc.
20
21 File Name:
22     app.h
23
24 Summary:
25     This header file provides prototypes and definitions for the application.
26
27 Description:
28     This header file provides function prototypes and data type definitions for
29     the application. Some of these are required by the system (such as the
30     "APP_Initialize" and "APP_Tasks" prototypes) and some of them are only used
31     internally by the application (such as the "APP_STATES" definition). Both
32     are defined here for convenience.
33 ****
34
35 //DOM-IGNORE-BEGIN
36 ****
37 Copyright (c) 2013-2014 released Microchip Technology Inc. All rights reserved.
38
39 Microchip licenses to you the right to use, modify, copy and distribute
40 Software only when embedded on a Microchip microcontroller or digital signal
41 controller that is integrated into your product or third party product
42 (pursuant to the sublicense terms in the accompanying license agreement).
43
44 You should refer to the license agreement accompanying this Software for
45 additional information regarding your rights and obligations.
46
47 SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND,
48 EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF
49 MERCHANTABILITY, TITLE, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE.
50 IN NO EVENT SHALL MICROCHIP OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER
51 CONTRACT, NEGLIGENCE, STRICT LIABILITY, CONTRIBUTION, BREACH OF WARRANTY, OR
52 OTHER LEGAL EQUITABLE THEORY ANY DIRECT OR INDIRECT DAMAGES OR EXPENSES
53 INCLUDING BUT NOT LIMITED TO ANY INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE OR
54 CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT OF
55 SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES
56 (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS.
57 ****
58 //DOM-IGNORE-END
59
60 #ifndef _APP_H
61 #define _APP_H
62
63 // ****
64 // ****
65 // Section: Included Files
66 // ****
67 // ****
68
69 #include <stdint.h>           // Library standard de types
70 #include <stdbool.h>          // Library standard pour le type bool
71 #include <stddef.h>            // Library standard de types implicites
72 #include <stdlib.h>             // Library standard d'accès à la mémoire
73 #include "system_config.h"      // Library des configurations system

```

```
74 #include "system_definitions.h"           // Library des definitions system
75
76
77
78
79
80
81
82
83 #define END_NAME_KEY_SIZE      3          // Taille de la clé de fin de nom
84
85 #define SIZE_KEY                2          // Taille de la clé pour le Software
86
87
88
89
90
91
92
93
94
95 void Retrieve_Name( void );
96
97 #endif /* _APP_H */
98
99 ****
100 End of File
101 */
102
```

```

1 ****
2 System Interrupts File
3
4 File Name:
5 system_interrupt.c
6
7 Summary:
8 Raw ISR definitions.
9
10 Description:
11 This file contains a definitions of the raw ISRs required to support the
12 interrupt sub-system.
13
14 Summary:
15 This file contains source code for the interrupt vector functions in the
16 system.
17
18 Description:
19 This file contains source code for the interrupt vector functions in the
20 system. It implements the system and part specific vector "stub" functions
21 from which the individual "Tasks" functions are called for any modules
22 executing interrupt-driven in the MPLAB Harmony system.
23
24 Remarks:
25 This file requires access to the systemObjects global data structure that
26 contains the object handles to all MPLAB Harmony module objects executing
27 interrupt-driven in the system. These handles are passed into the individual
28 module "Tasks" functions to identify the instance of the module to maintain.
29 ****
30
31 // DOM-IGNORE-BEGIN
32 ****
33 Copyright (c) 2011-2014 released Microchip Technology Inc. All rights reserved.
34
35 Microchip licenses to you the right to use, modify, copy and distribute
36 Software only when embedded on a Microchip microcontroller or digital signal
37 controller that is integrated into your product or third party product
38 (pursuant to the sublicense terms in the accompanying license agreement).
39
40 You should refer to the license agreement accompanying this Software for
41 additional information regarding your rights and obligations.
42
43 SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND,
44 EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF
45 MERCHANTABILITY, TITLE, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE.
46 IN NO EVENT SHALL MICROCHIP OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER
47 CONTRACT, NEGLIGENCE, STRICT LIABILITY, CONTRIBUTION, BREACH OF WARRANTY, OR
48 OTHER LEGAL EQUITABLE THEORY ANY DIRECT OR INDIRECT DAMAGES OR EXPENSES
49 INCLUDING BUT NOT LIMITED TO ANY INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE OR
50 CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT OF
51 SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES
52 (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS.
53 ****
54 // DOM-IGNORE-END
55
56 // ****
57 // ****
58 // Section: Included Files
59 // ****
60 // ****
61
62 #include "system/common/sys_common.h"
63 #include "app.h"
64 #include "system_definitions.h"
65
66 // ****
67 // ****
68 // Section: System Interrupt Vector Functions
69 // ****
70 // ****
71
72 static uint8_t compt_Block = 0;
73 bool Ticket_Refused;

```

```
74 //void __ISR(_UART_2_VECTOR, ipl0AUTO) _IntHandlerDrvUsartInstance1(void)
75 //{
76 //    DRV_USART_TasksTransmit(sysObj.drvUsart1);
77 //    DRV_USART_TasksError(sysObj.drvUsart1);
78 //    DRV_USART_TasksReceive(sysObj.drvUsart1);
79 //}
80
81
82
83 void __ISR(_TIMER_1_VECTOR, ipl1AUTO) IntHandlerDrvTmrInstance0(void)
84 {
85     PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_TIMER_1);
86     Led_Link_LostToggle();
87 }
88
89 void __ISR(_TIMER_2_VECTOR, ipl1AUTO) IntHandlerDrvTmrInstance1(void)
90 {
91     if(compt_Block < 49)
92     {
93         compt_Block++;
94     }
95     else
96     {
97         Ticket_Refused = false;
98         compt_Block = 0;
99     }
100    PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_TIMER_2);
101 }
102 ****
103 End of File
104 */
105
```

```
1 //-----  
2 // Data_Code.h  
3 //-----  
4 //  
5 //  
6 // Auteur      :  
7 // Date       :  
8 // Version     :  
9 // Modifications : MDS 26.09.2022  
10 // Description :  
11 //           Definition des differente commande  
12 //  
13 //  
14 /*-----*/  
15  
16 #ifndef _DATA_CODE_H      /* Guard against multiple inclusion */  
17 #define _DATA_CODE_H  
18  
19 #define ENVOI_TICKET 0x917283bd  
20 #define TICKET_ANNULER 0xccc18ca5  
21  
22 #define TICKET_ACCEPT 0x00e0d135  
23 #define TICKET_REFUSE 0xc5fb3140  
24 #define TICKET_RESET 0xb0bfe0fc  
25 #define BLOCKED 0xcc9ebbb17  
26  
27 #define ARE_U_LINK 0x3103de90  
28 #define I_AM_LINK 0xcd05a45  
29  
30 #define ACK 0x1b1ac4f6  
31  
32  
33  
34 #define ADD_BROADCAST 0xFFFFFFFF  
35  
36  
37     /* Provide C++ Compatibility */  
38 #ifdef __cplusplus  
39 }  
40 #endif  
41  
42 #endif /* _EXAMPLE_FILE_NAME_H */  
43  
44 /* *****  
45 End of File  
46 */  
47
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Threading.Tasks;
5  using System.IO.Ports;
6  using System.Threading;
7  using System.Windows.Forms;
8
9
10 namespace _1811C_Ticketing
11 {
12     public static class Global
13     {
14         public static string All_Messages;//reception du message complet
15         public static bool Receive_Message = false;//vérification si un
16             message a été reçu
17         public static string[] List_Name_student = new string[25];//nom des
18             élèves enregistrer
19         public static string[] List_Adress = new string[25];//adresse reçue
20
21         public static char[] Name_student;//nom des élèves enregistrer
22         public static char[] Adress = new char[20];//adresse reçue
23
24         public static int Nbr_Student_Save = 0;//Nombre de nom d élève
25             différent
26         public static int Nbr_Adress_Save = 0;//Nombre d adresse différent
27
28         public static string Actual_Adress;//dernière adresse reçue
29         public static string Actual_Student;
30
31         public static bool Student_OK = false;//vérification si un message a
32             été reçu
33
34         public static int Last_Adress;
35
36     static class Constants
37     {
38         public const string Start = "AA", End = "BB";
39         public const int Size_Indicator = 2;
40         public const int Size_Adress = 6; // 4 pour le Xbee XB24CAPIT-001 ou
41             16 selon mode utiliser
42         public const int Size_Adress_Broadcast = 8;// 4 pour le Xbee
43             XB24CAPIT-001 et 8 pour le broadcast du xbee de 1 es
44         public const string Adress_Broadcast_ES = "00000000";//adresse de
45             broadcast du xbee de 1 es
46         public const string Adress_Broadcast_Xbee4 = "FFFF";//adresse de
47             broadcast du xbee XB24CAPIT-001
48         public const string Adress_Broadcast_Xbee16 = "000000000000FFFF";//
49             adresse de broadcast du xbee XB24CAPIT-001
50         public const string Accept = "ACCEPT", Decline = "DECLINE", Reset =
51             "RESET";// constant des réponses à envoyer
52
53
54         public const int Default_BaudRate = 57600;// valeur de default du
55             BaudRate de la communication serial
56         public const System.IO.Ports.Parity Default_Parity =
```

```
        System.IO.Ports.Parity.None;// valeur de default de la Partiy de la ↵
        communication serial
46    public const int Default_DataBits = 8;// valeur de default du nombre ↵
        de bytes de la communication serial
47    public const System.IO.Ports.Handshake Default_Handshake = ↵
        System.IO.Ports.Handshake.None;// valeur de default du handshake de ↵
        la communication serial
48    public const System.IO.Ports.StopBits Default_Stop_Bits = ↵
        System.IO.Ports.StopBits.One;// valeur de default de la taille du ↵
        stop bits de la communication serial
49
50    }
51    static class Program
52    {
53        /// <summary>
54        /// Point d'entrée principal de l'application.
55        /// </summary>
56
57
58        [STAThread]
59        static void Main()
60        {
61            Application.EnableVisualStyles();
62            Application.SetCompatibleTextRenderingDefault(false);
63            Application.Run(new Form1());
64
65            if (Global.Receive_Message == true)// on verifie que l'on a bien ↵
                receptionner un message
66
67            {
68                char[] Message = new char[Global.All_Messages.Length];
69                int i = 0, compt = 0;
70
71                for (i = 0; i < Global.All_Messages.Length; i++)
72                {
73                    Message[i] = Global.All_Messages[i]; //copier le string ↵
                    dans un tableau de char
74                }
75                if (Message[0] == 'A' && Message[1] == 'A')// on verifie si le ↵
                    debut de la trame recu correspond à l'indicateur ↵
                    Start_First = AA
76            {
77                do
78                {
79                    Global.Adress[compt] = Message[compt + ↵
                    Constants.Size_Indicator]; //copier l'adresse dans un ↵
                    tableau de char (en ignorant les 2 premier caracteres qui ↵
                    sont l'indicateur d envoie)
80                    compt++; //indicateur de la position du curseur
81
82                } while (compt != Constants.Size_Adress + ↵
                    Constants.Size_Indicator); // tant que l'on atteint pas 8 ↵
                    continuer la boucle (2 caracteres pour indicateur + 6 de l ↵
                    adresse)
83                compt = 0;// on reinitialise la position du curseur
84                string Adress = new string(Global.Adress); // on copie le ↵
```

```
    tableau de caratere dans une variable string
85
86
87        for (i = 0; i < Global.List_Adress.Length; i++)// on fait ↵
88        une boucle de la taille du tableau liste d adresse
89        {
90            if (Global.List_Adress[i] != Adress)//On verifie s'il ↵
91            existe
92                Global.List_Adress[Global.Nbr_Adress_Save] = ↵
93                Adress; // on copie l adresse dans le tableau de string ↵
94                liste d adresse
95        }
96        Global.Actual_Adress = Adress;
97
98
99
100       for (i = Adress.Length + Constants.Size_Indicator; i < ↵
101       Global.All_Messages.Length - Adress.Length - 2 * ↵
102       Constants.Size_Indicator; i++)// faire la boucle (taille ↵
103       max du message total moins la taille de l adress et des ↵
104       indicateur de début et de fin)
105       {
106           Global.Name_student[i] = Message[i + Adress.Length + ↵
107           Constants.Size_Indicator];//copie du nom de l eleve du ↵
108           message totale dans un tableau de char
109       }
110
111       string Name_Student = new string(Global.Name_student);//on ↵
112       copie le tableau de caratere dans une variable string
113       Global.Nbr_Student_Save++;
114       Global.List_Name_student[Global.Nbr_Student_Save] = ↵
115       Name_Student;// on copie le nom de l eleve dans le tableau ↵
116       de string liste de nom d eleve
117       Global.Actual_Student = Name_Student;
118   }
119   /*else if (Message[0] == 'B' && Message[1] == 'B')// on ↵
120   verifie si le debut de la trame recu correspond à ↵
121   l'indicateur Start = BB
122   {
123       do
124       {
125           Global.Adress[compt] = Message[compt + ↵
126           Constants.Size_Indicator]; //copier l'adresse dans un ↵
127           tableau de char (en ignorant les 2 premier caracteres qui ↵
128           sont l'indicateur d envoie)
129           compt++; //indicateur de la position du curseur
130
131       } while (compt != Constants.Size_Adress + ↵
132           Constants.Size_Indicator); // tant que l'on atteint pas 8 ↵
133           continuer la boucle (2 caracteres pour indicateur + 6 de l ↵
134           adresse)
135           compt = 0;// on reinitialise la position du curseur
136           string Adress = new string(Global.Adress);// on copie le ↵
137           tableau de caratere dans une variable string
138           Global.Actual_Adress = Adress;
139           while(Global.Student_OK == true)
140           {
```

...ntos\Desktop\1811C_Ticketing\1811C_Ticketing\Program.cs 4

```
118                     if(string.Equals(Global.List_Adress[compt] , Adress))
119                     {
120                         Global.Student_OK = true;
121                         Global.Actual_Student = Global.List_Name_student  ↵
122                             [compt];
123                         }
124                         else
125                         {
126                             compt++;
127                         }
128                         }
129                         compt = 0;
130
131                     }*/
132                 }
133             }
134         }
135     }
136 }
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.IO.Ports;
10 using System.IO;
11 using System.Windows.Forms;
12
13 namespace _1811C_Ticketing
14 {
15
16     public partial class Form1 : Form
17     {
18
19         public static int BaudRate, Data_bits;
20         public static string Parity, Handshake, Stop_Bits, CB_Port_Past;
21
22         public static bool New_Port = true, Unconnect = false, Start = true,    ↴
23             Confirme_Change, Close_Windows;
24
25         public static Form1 frm1 = new Form1();
26
27         public Form1()
28     {
29         InitializeComponent();
30
31         //transfert des variables de la form2 à la form1
32         BaudRate = Form2.BaudRate_Change;
33         Parity = Form2.Parity_Change;
34         Data_bits = Form2.Data_Bits_Change;
35         Handshake = Form2.Handshake_Change;
36         Stop_Bits = Form2.Stop_Bits_Change;
37         Confirme_Change = Form2.Change;
38         Close_Windows = Form2.Windows_Close;
39     }
40
41     private void Form1_Load(object sender, EventArgs e)
42     {
43     }
44
45     private void Btn_Acc_Click(object sender, EventArgs e)
46     {
47         if (Global.Receive_Message == true)
48     {
49             // preparation du message
50             // indicateur de start = AA
51             // dernière adresse reçue
52             // Message, ici c'est "ACCEPT"
53             // indicateur de fin = BB
54             string envoie = Constants.Start + Global.Actual_Adress +    ↴
55                 Constants.Accept + Constants.End;
```

```
55             if (AnyClass.port.IsOpen)//Verifier si le port com est ouvert
56             {
57                 //envoie du message
58                 SerialPort.WriteLine(envoie);
59             }
60             else
61             {
62                 //Si le port n'est pas ouvert ouverture de fenetre pour     ↵
63                 //indiquer d'en selectionner un
64                 MessageBox.Show("Please select a port first and connect     ↵
65                 it");
66             }
67             Global.Receive_Message = false;
68         }
69     }
70
71     private void Btn_Dec_Click(object sender, EventArgs e)
72     {
73         if (Global.Receive_Message == true)
74         {
75             // preparation du message
76             // indicateur de start = AA
77             // derniere adresse recu
78             // Message, ici c'est "DECLINE"
79             //indicateur de fin = CC
80             string envoie = Constants.Start + Global.List_Adress           ↵
81             [Global.Last_Adress] + Constants.Decline + Constants.End;
82             if (AnyClass.port.IsOpen)//Verifier si le port com est ouvert
83             {
84                 //envoie du message
85                 SerialPort.WriteLine(envoie);
86             }
87             else
88             {
89                 //Si le port n'est pas ouvert ouverture de fenetre pour     ↵
90                 //indiquer d'en selectionner un
91                 MessageBox.Show("Please select a port first and connect     ↵
92                 it");
93             }
94         }
95
96         Global.Receive_Message = false;
97     }
98
99     private void Btn_Rst_Click(object sender, EventArgs e)
100    {
101        if (Global.Receive_Message == true)
102        {
103            // preparation du message
104            // indicateur de start = AA
105            // Adresse de broadcast
106            // Message, ici c'est "RESET"
107            //indicateur de fin = CC
108            string envoie = Constants.Start +           ↵
```

```
        Constants.Adress_Broadcast_ES + Constants.Reset +
        Constants.End;
106    if (AnyClass.port.IsOpen)//Verifier si le port com est ouvert
107    {
108        //envoie du message
109        SerialPort.Write(envoie);
110    }
111    else
112    {
113        //Si le port n'est pas ouvert ouverture de fenetre pour
114        //indiquer d'en selectionner un
115        MessageBox.Show("Please select a port first and connect
116        it");
117    }
118}
119
120 private void Btn_Opt_Click(object sender, EventArgs e)
121{
122    //si on n'a aucun message à traiter
123    if (Global.Receive_Message == false)
124    {
125
126        try
127        {
128            //on rend visible l'interface 2
129            Form2 frm2.Visible = true;
130            //on rend invisible la premiere interface
131            Hide();
132        }
133        catch(Exception)
134        {
135            Form2 frm2 = new Form2();
136            frm2.Show();
137            Hide();
138        }
139
140    }
141}
142
143
144 private void CB_Port_Click(object sender, EventArgs e)
145{
146
147    //on recois une list de nom de port
148    var ports = SerialPort.GetPortNames();
149    //on affiche les nom dans la combo box
150    CB_Port.DataSource = ports;
151}
152
153
154 private void CB_Port_SelectedIndexChanged(object sender, EventArgs e)
155{
156    //on convertie le nom selectionner en string et on l'affiche
157    CB_Port.Text = CB_Port.SelectedItem.ToString();
158    //on verifie si le Texte afficher à été modifier
```

```
158         if (CB_Port_Past != CB_Port.Text)
159         {
160             //on met un flag pour deconnecter le port precedent
161             Unconnect = true;
162
163             //vérifie si c'est la premiere fois que nous passons
164             if(Start != true)
165                 //si ce n'est pas la premiere fois en entre dans la      ↵
166                 méthode connect
167                 Connect();
168
169             //on copie la valeur de Texte
170             CB_Port_Past = CB_Port.Text;
171
172             //on reset le flag de deconnection
173             Unconnect = false;
174             Start = false;
175         }
176         //on verifie si c'est on va utiliser un nouveau port
177         if (New_Port == true)
178         {
179             //Si on entre dans cette déclaration cela veut dire que le      ↵
180             //port est déjà fermer
181
182             //en cree une nouvelle instance
183             AnyClass.port = new SerialPort(CB_Port.SelectedItem.ToString      ↵
184             ());
185
186
187
188     }
189
190     private void Btn_USB_Click(object sender, EventArgs e)
191     {
192         //on verifie si un port a été selectionner
193         if (CB_Port.SelectedIndex > -1)
194         {
195             //on indique dans une nouvelle fenetre que le port à été      ↵
196             //electionner
197             MessageBox.Show(string.Format("You selected port '{0}'",      ↵
198             CB_Port.SelectedItem));
199             Connect();
200         }
201         else
202         {
203             MessageBox.Show("Please select a port first");
204         }
205
206         //class global
207         abstract class AnyClass
208         {
209             //public static SerialPort port = new SerialPort();
```

```
209         public static SerialPort port;
210
211     }
212     private void Connect()
213     {
214
215         //SerialPort port = new SerialPort(CB_Port.SelectedItem.ToString()    ↵
216         //());
216         //AnyClass.port = new SerialPort(CB_Port.SelectedItem.ToString());
217
218         if (Unconnect == true)
219     {
220             AnyClass.port.Close();
221             New_Port = true;
222         }
223         else
224     {
225             //on vérifie si le port est ouvert
226             if(AnyClass.port.IsOpen)
227                 //s'il est ouvert on le ferme par précaution
228                 AnyClass.port.Close();
229
230             if(Confirme_Change == true)
231     {
232                 //modification des paramètres de l'UART selon valeur      ↵
233                 inscrit par l'utilisateur
234                 AnyClass.port.BaudRate = BaudRate;
234                 AnyClass.port.Parity = (Parity)Enum.Parse(typeof(Parity),    ↵
235                 Parity);
235                 AnyClass.port.DataBits = Data_bits;
236                 AnyClass.port.Handshake = (Handshake)Enum.Parse(typeof(Handshake),    ↵
236                 Handshake);
237                 AnyClass.port.StopBits = (StopBits)Enum.Parse(typeof(StopBits),    ↵
237                 Stop_Bits);
238         }
239         else
240     {
241                 //modification des paramètres de l'UART selon valeur par    ↵
242                 default
242                 AnyClass.port.BaudRate = Constants.Default_BaudRate;
243                 AnyClass.port.Parity = Constants.Default_Parity;
244                 AnyClass.port.DataBits = Constants.Default_DataBits; ;
245                 AnyClass.port.Handshake = Constants.Default_Handshake;
246                 AnyClass.port.StopBits = Constants.Default_Stop_Bits;
247         }
248     }
249     //on ouvre le port
250     AnyClass.port.Open();
251 }
252
253     private void SerialPort_DataReceived(object sender,
253         SerialDataReceivedEventArgs e)
254     {
255
256         try
257     {
```

```
258             //on copie le message recu
259             Global.All_Messages = SerialPort.ReadLine();
260         }
261         catch (TimeoutException) { }
262         //on met un flag de reception
263         bool Receive_Message = true;
264
265     }
266 }
267 }
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 using System.Collections;
11 using System.Collections.Generic;
12 using System.IO.Ports;
13
14 namespace _1811C_Ticketing
15 {
16     public partial class Form2 : Form
17     {
18         public static int BaudRate_Receive, BaudRate_Change, Data_Bits_Receive, ↪
19             Data_Bits_Change;
20         public static string Parity_Receive, Parity_Change, Handshake_Receive, ↪
21             Handshake_Change, Stop_Bits_Change, Stop_Bits_Receive;
22         public static bool Change = false, Windows_Close;
23         public static Form2 frm2 = new Form2();
24
25         public Form2()
26         {
27             InitializeComponent();
28         }
29
30         private void Form2_Load(object sender, EventArgs e)
31         {
32
33         private void Form2_FormClosing(object sender, FormClosingEventArgs e)
34         {
35             Windows_Close = true;
36             Hide();
37             Form1 frm1.Visible = true;
38         }
39
40         private void BaudRate_TextChanged(object sender, EventArgs e)
41         {
42
43             BaudRate_Receive= int.Parse(BaudRate.Text);
44             Change = true;
45         }
46
47         private void Parity_SelectedIndexChanged(object sender, EventArgs e)
48         {
49             Parity_Receive = Parity.Text;
50             Change = true;
51         }
52
53         private void Data_Bits_TextChanged(object sender, EventArgs e)
54         {
```

```
55         Data_Bits_Receive = int.Parse(Data_Bits.Text);
56         Change = true;
57     }
58
59     private void Handshake_SelectedIndexChanged(object sender, EventArgs e)
60     {
61         Handshake_Receive = Handshake.Text;
62         Change = true;
63     }
64
65     private void Stop_Bits_SelectedIndexChanged(object sender, EventArgs e)
66     {
67         Stop_Bits_Receive = Stop_Bits.Text;
68         Change = true;
69     }
70
71     private void Btn_Configure_Click(object sender, EventArgs e)
72     {
73         BaudRate_Change = BaudRate_Receive;
74         Parity_Change = Parity_Receive;
75         Data_Bits_Change = Data_Bits_Receive;
76         Handshake_Change = Handshake_Receive;
77         Stop_Bits_Change = Stop_Bits_Receive;
78         Hide();
79         Form1 frm1 = new Form1();
80         frm1.Visible = true;
81         frm2.Visible = false;
82     }
83
84     private void Btn_SL01_Click(object sender, EventArgs e)
85     {
86         System.Diagnostics.Process.Start("https://gesteleves.etmlnet.local/ ↵
87             class/SL01");
88     }
89
90     private void Btn_SL02_Click(object sender, EventArgs e)
91     {
92         System.Diagnostics.Process.Start("https://gesteleves.etmlnet.local/ ↵
93             class/SL02");
94     }
95 }
```

Mode d'emploi

Ecole supérieure
Électronique

Système de ticketing pour questions
d'étudiants

Projet n°1811C

Réalisé par :

Dos Santos Mario

Table des matières :

Système de ticketing pour questions d'étudiants	1
Projet n°1811C.....	1
1 Manuelle d'utilisation	4
1.1 Alimentation.....	4
1.2 Présentation Carte Master	4
1.3 Présentation Carte Slave	5
1.4 Présentation application Ticketing.....	7
1.4.1 Fenêtre principal.....	7
1.4.2 Fenêtre paramétrage.....	8

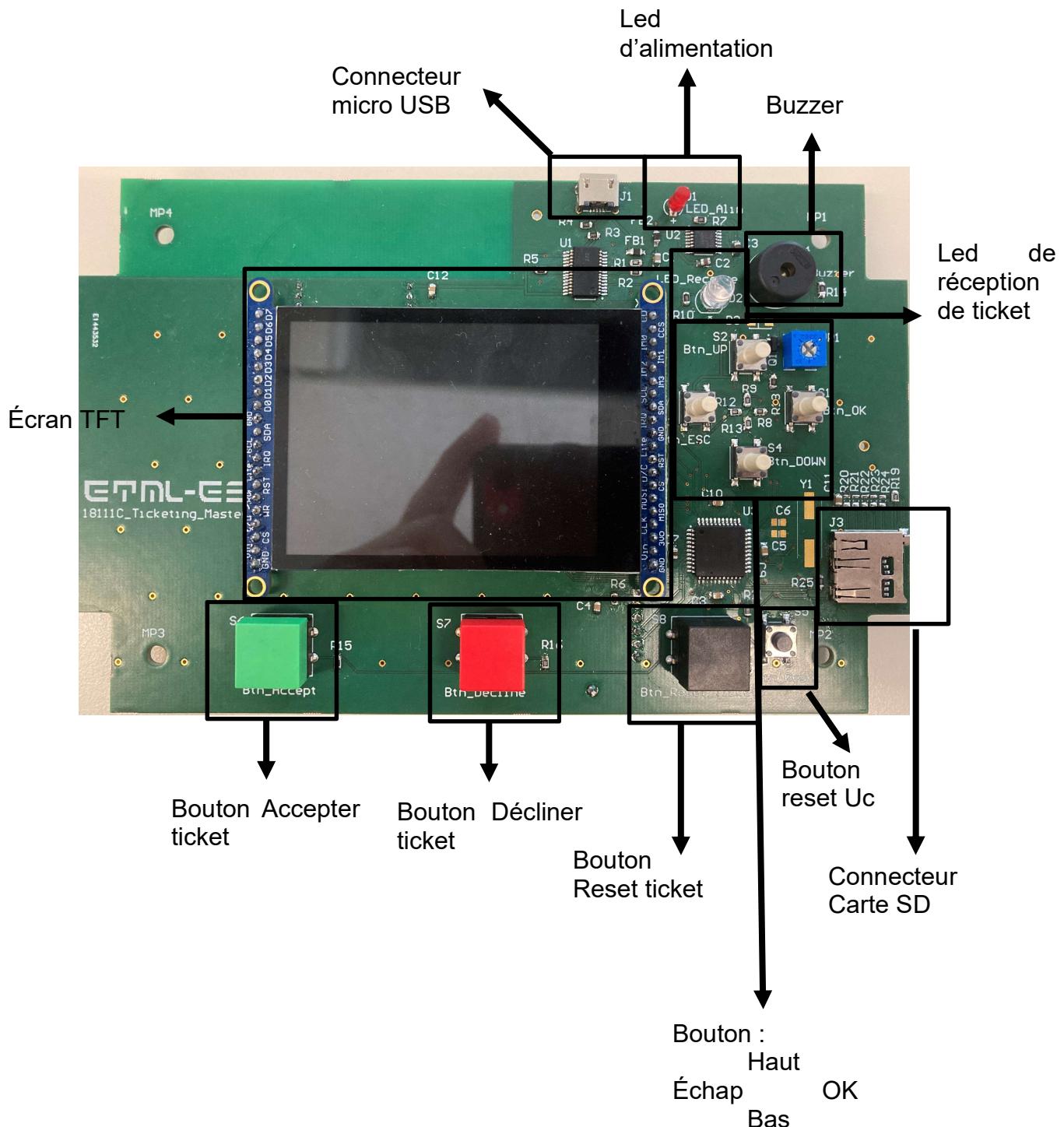
1 Manuelle d'utilisation

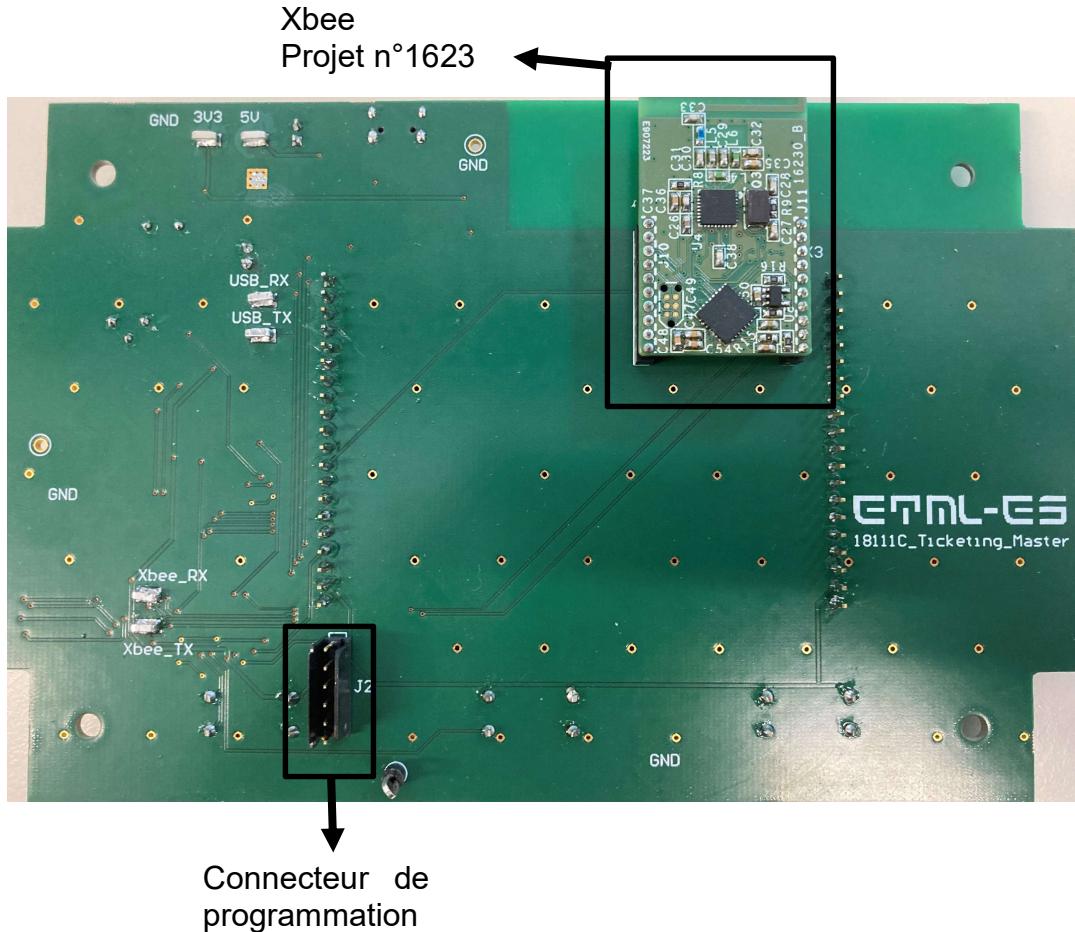
1.1 Alimentation

Les cartes seront alimentées en 5V en microUSB

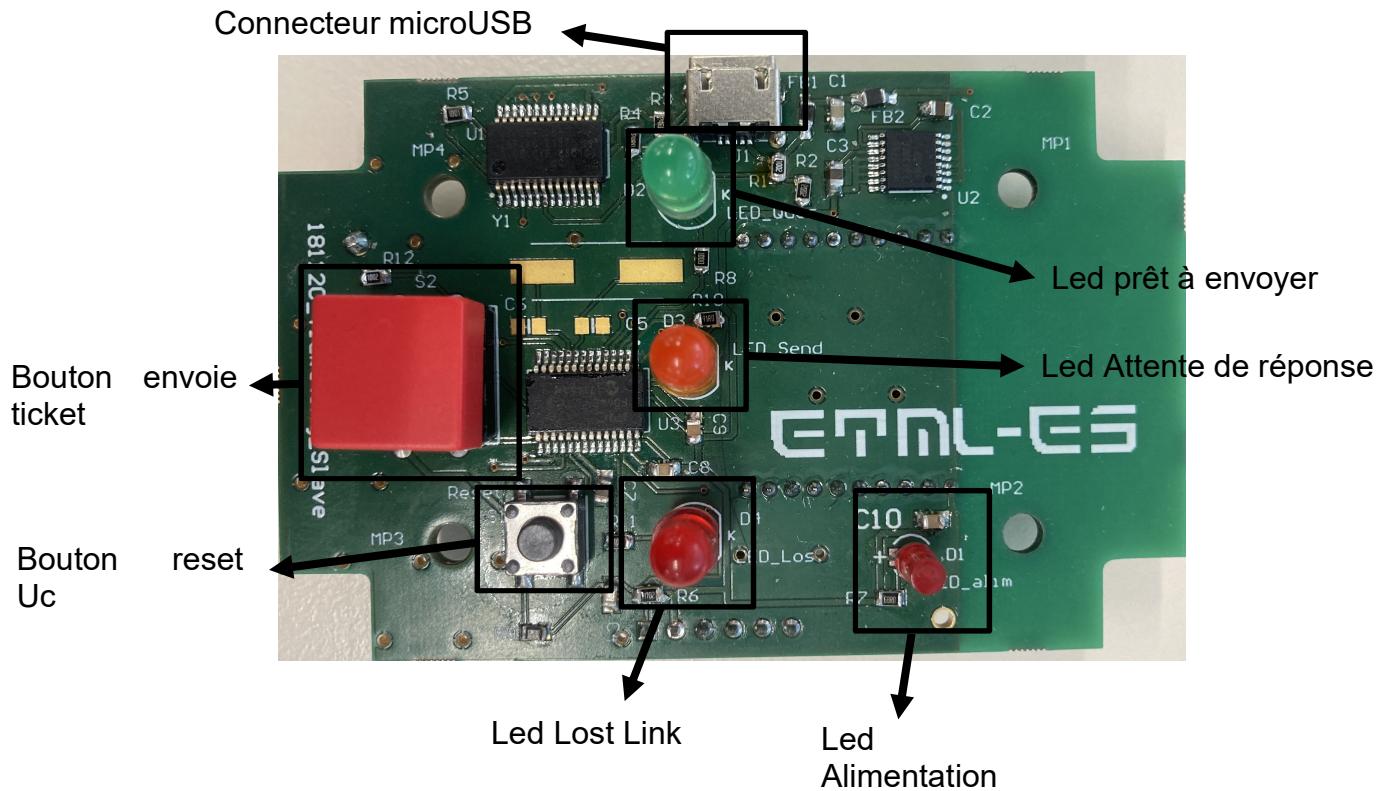
Pour un bon fonctionnement de la carte l'alimentation sera fourni par un port d'un ordinateur de l'ES.

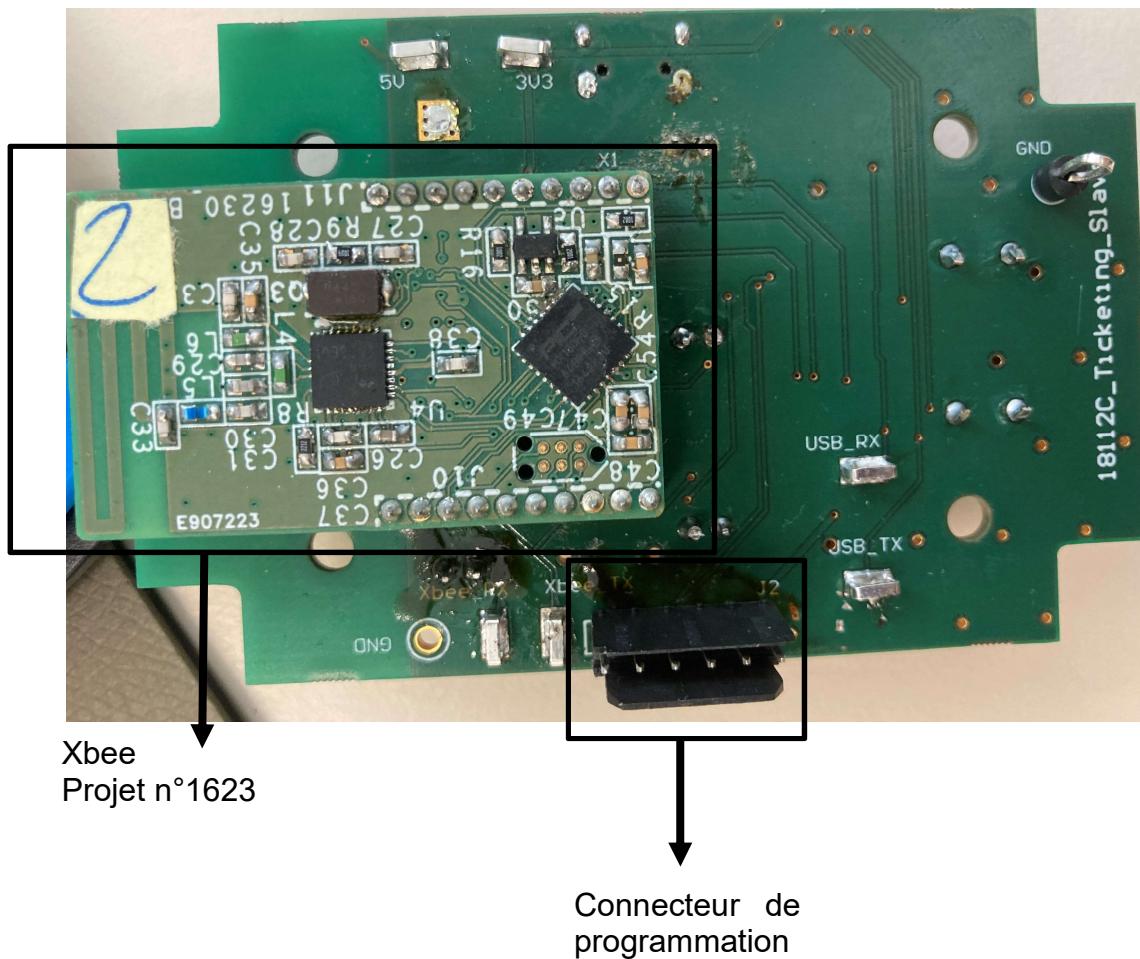
1.2 Présentation Carte Master





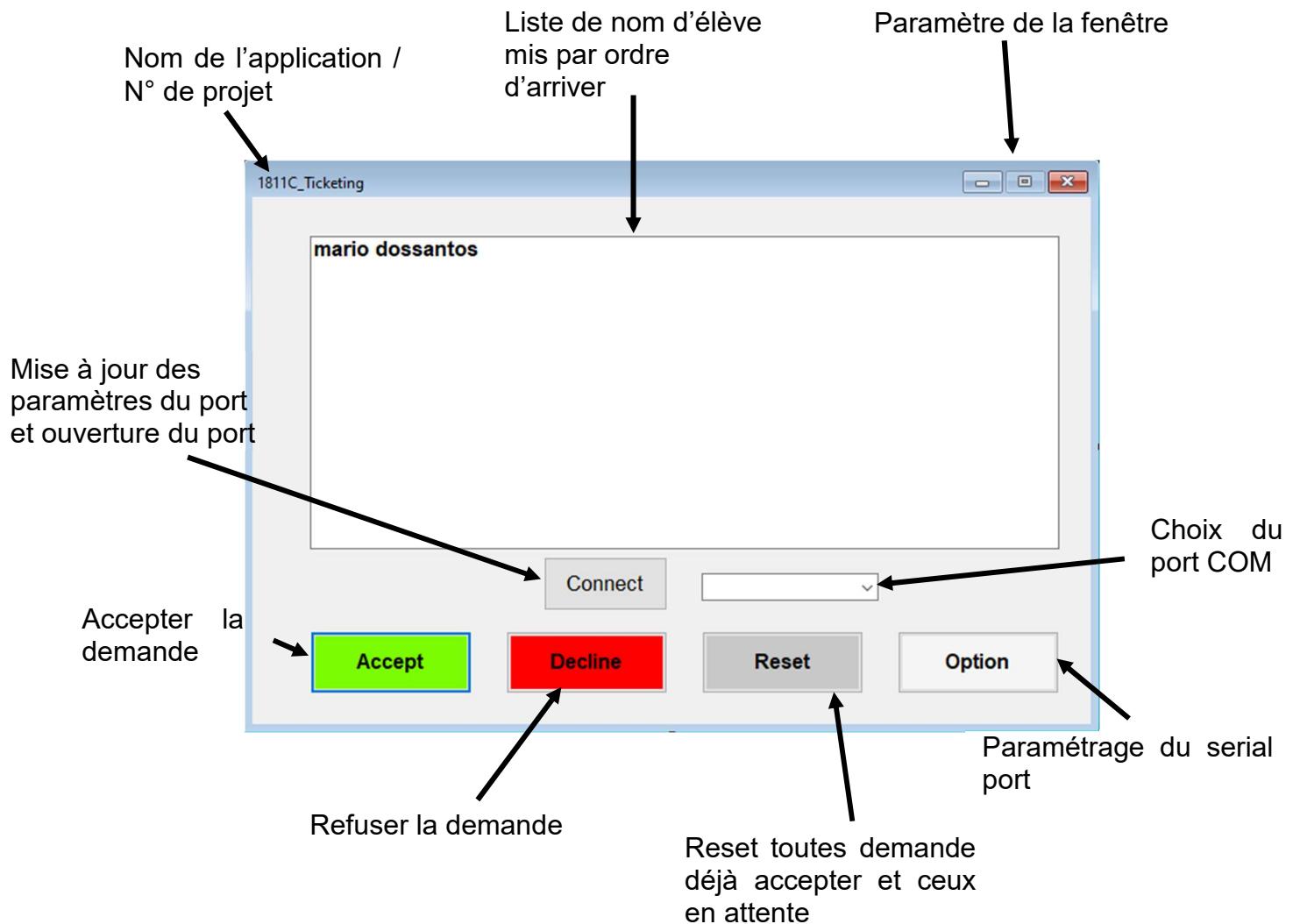
1.3 Présentation Carte Slave





1.4 Présentation application Ticketing

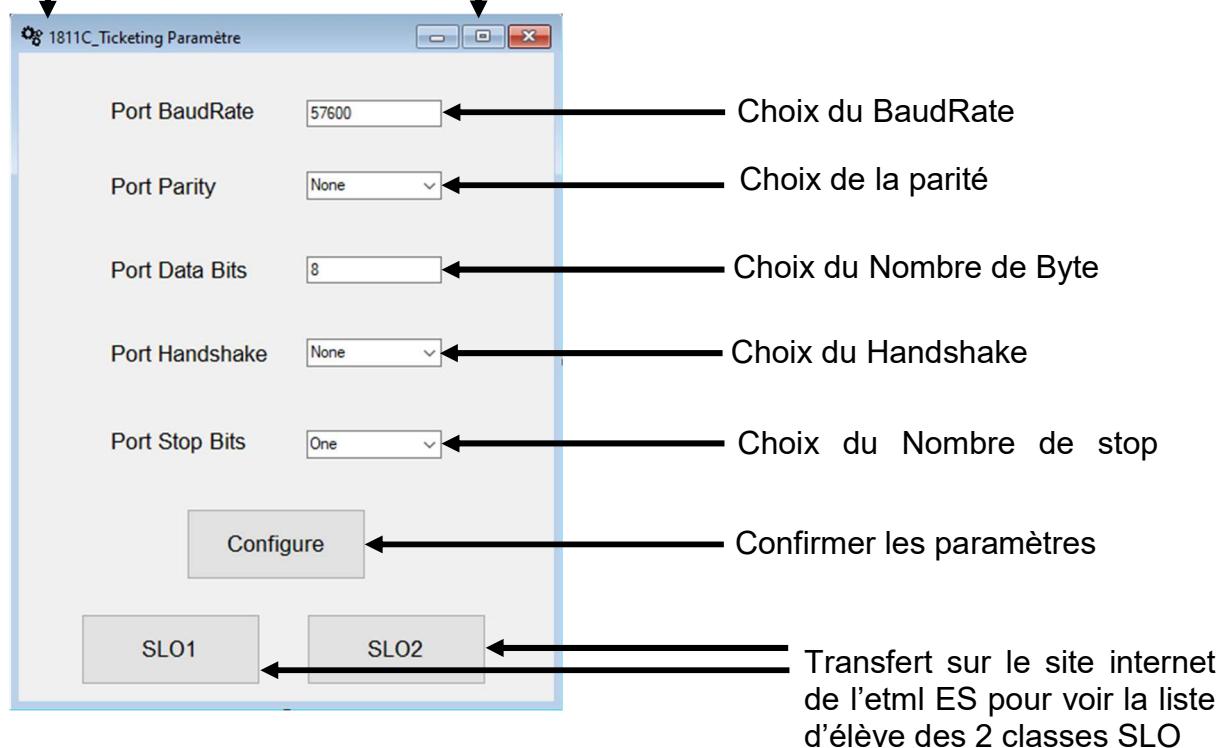
1.4.1 Fenêtre principal



1.4.2 Fenêtre paramétrage

Icône de l'application
Nom de l'application /
N° de projet

Paramètre de la fenêtre



1.5 Mise en service

Connecter un câble microUSB de la carte Master à un ordinateur. Pareillement pour la carte Slave

L'écran de la carte Master s'allume est attente

Les leds de la carte slave s'allume

Ouvrir l'application du projet 2126 et attendre (env. 5s)

Les leds de la carte slave s'éteigne et la led rouge se met à clignoter

Lorsque la carte slave arrive à se link avec la carte master la led rouge arrête de clignoter et la led verte s'allume

Lorsque l'on appuie sur le bouton de la carte slave la led verte s'éteint et la led orange s'allume, un ticket vient d'être envoyé

Lorsque la carte Master reçoit le ticket, la led orange s'allume et le buzzer vibre tant qu'il n'y a pas de réponse

Si l'on appuie sur le bouton vert de la carte master le ticket est accepté et le nom de l'élève qui a envoyé le message se retrouve dans la liste d'attente.

Si l'on appuie sur le bouton rouge le ticket est refusé.

Sur la carte Slave si le ticket est accepté alors la led orange s'éteint et la led verte se rallume. Par contre s'il a été refuser la led orange s'éteint et la led rouge s'allume pendant 20s, durant ce temps il est impossible de renvoie un ticket. Passer se délai la led rouge s'éteint et la led verte s'allume.

A tout moment il est possible de reset la liste d'attente et les demande de ticket grâce au bouton reset_ticket sur la carte master.

L'application C# reprend l'affichage de l'écran TFT et permet les mêmes fonctionnalités et plus encore.

**MICROCHIP****PIC32MX1XX/2XX 28/36/44-PIN**

32-bit Microcontrollers (up to 256 KB Flash and 64 KB SRAM) with Audio and Graphics Interfaces, USB, and Advanced Analog

Operating Conditions

- 2.3V to 3.6V, -40°C to +105°C, DC to 40 MHz
- 2.3V to 3.6V, -40°C to +85°C, DC to 50 MHz

Core: 50 MHz/83 DMIPS MIPS32® M4K®

- MIPS16e® mode for up to 40% smaller code size
- Code-efficient (C and Assembly) architecture
- Single-cycle (MAC) 32x16 and two-cycle 32x32 multiply

Clock Management

- 0.9% internal oscillator
- Programmable PLLs and oscillator clock sources
- Fail-Safe Clock Monitor (FSCM)
- Independent Watchdog Timer
- Fast wake-up and start-up

Power Management

- Low-power management modes (Sleep and Idle)
- Integrated Power-on Reset and Brown-out Reset
- 0.5 mA/MHz dynamic current (typical)
- 44 µA IPD current (typical)

Audio Interface Features

- Data communication: I²S, LJ, RJ, and DSP modes
- Control interface: SPI and I²C
- Master clock:
 - Generation of fractional clock frequencies
 - Can be synchronized with USB clock
 - Can be tuned in run-time

Advanced Analog Features

- ADC Module:
 - 10-bit 1.1 Msps rate with one S&H
 - Up to 10 analog inputs on 28-pin devices and 13 analog inputs on 44-pin devices
- Flexible and independent ADC trigger sources
- Charge Time Measurement Unit (CTMU):
 - Supports mTouch™ capacitive touch sensing
 - Provides high-resolution time measurement (1 ns)
 - On-chip temperature measurement capability
- Comparators:
 - Up to three Analog Comparator modules

Packages

Type	SOIC	SSOP	SPDIP	QFN		VTLA		TQFP
Pin Count	28	28	28	28	44	36	44	44
I/O Pins (up to)	21	21	21	21	34	25	34	34
Contact/Lead Pitch	1.27	0.65	0.100"	0.65	0.65	0.50	0.50	0.80
Dimensions	17.90x7.50x2.65	10.2x5.3x2	1.365"x.285"x.135"	6x6x0.9	8x8x0.9	5x5x0.9	6x6x0.9	10x10x1

Note: All dimensions are in millimeters (mm) unless specified.

- Programmable references with 32 voltage points

Timers/Output Compare/Input Capture

- Five General Purpose Timers:
 - Five 16-bit and up to two 32-bit Timers/Counters
- Five Output Compare (OC) modules
- Five Input Capture (IC) modules
- Peripheral Pin Select (PPS) to allow function remap
- Real-Time Clock and Calendar (RTCC) module

Communication Interfaces

- USB 2.0-compliant Full-speed OTG controller
- Two UART modules (12.5 Mbps):
 - Supports LIN 2.0 protocols and IrDA® support
- Two 4-wire SPI modules (25 Mbps)
- Two I²C modules (up to 1 Mbaud) with SMBus support
- PPS to allow function remap
- Parallel Master Port (PMP)

Direct Memory Access (DMA)

- Four channels of hardware DMA with automatic data size detection
- Two additional channels dedicated for USB
- Programmable Cyclic Redundancy Check (CRC)

Input/Output

- 10 mA source/sink on all I/O pins and up to 14 mA on non-standard VOH
- 5V-tolerant pins
- Selectable open drain, pull-ups, and pull-downs
- External interrupts on all I/O pins

Class B Support

- Class B Safety Library, IEC 60730

Debugger Development Support

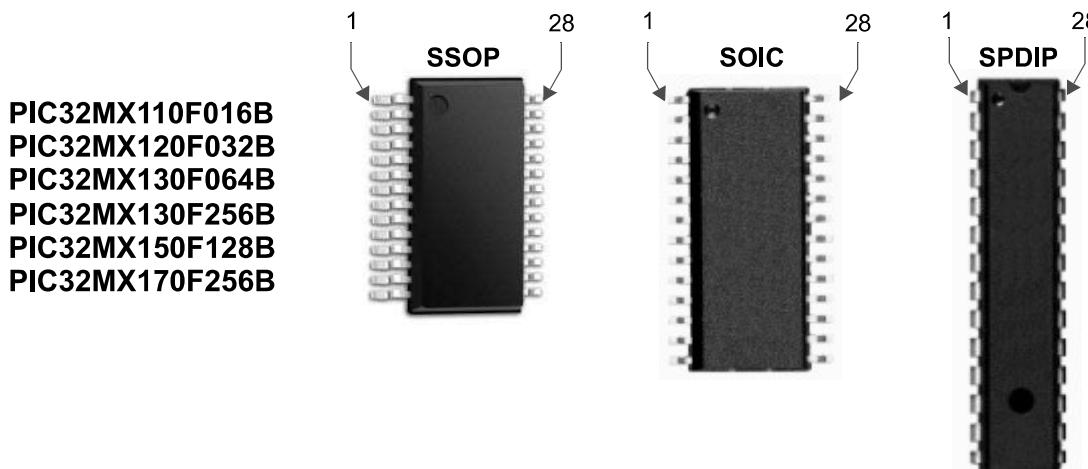
- In-circuit and in-application programming
- 4-wire MIPS® Enhanced JTAG interface
- Unlimited program and six complex data breakpoints
- IEEE 1149.2-compatible (JTAG) boundary scan

PIC32MX1XX/2XX 28/36/44-PIN FAMILY

Pin Diagrams

TABLE 3: PIN NAMES FOR 28-PIN GENERAL PURPOSE DEVICES

28-PIN SOIC, SPDIP, SSOP (TOP VIEW)^(1,2,3)



Pin #	Full Pin Name	Pin #	Full Pin Name
1	MCLR	15	PGEC3/RPB6/PMD6/RB6
2	VREF+/CVREF+/AN0/C3INC/RPA0/CTED1/RA0	16	TDI/RPB7/CTED3/PMD5/INT0/RB7
3	VREF-/CVREF-/AN1/RPA1/CTED2/RA1	17	TCK/RPB8/SCL1/CTED10/PMD4/RB8
4	PGED1/AN2/C1IND/C2INB/C3IND/RPB0/RB0	18	TDO/RPB9/SDA1/CTED4/PMD3/RB9
5	PGEC1/AN3/C1INC/C2INA/RPB1/CTED12/RB1	19	Vss
6	AN4/C1INB/C2IND/RPB2/SDA2/CTED13/RB2	20	VCAP
7	AN5/C1INA/C2INC/RTCC/RPB3/SCL2/RB3	21	PGED2/RPB10/CTED11/PMD2/RB10
8	Vss	22	PGEC2/TMS/RPB11/PMD1/RB11
9	OSC1/CLKI/RPA2/RA2	23	AN12/PMD0/RB12
10	OSC2/CLKO/RPA3/PMA0/RA3	24	AN11/RPB13/CTPLS/PMRD/RB13
11	SOSCI/RPB4/RB4	25	CVREFOUT/AN10/C3INB/RPB14/SCK1/CTED5/PMWR/RB14
12	SOSCO/RPA4/T1CK/CTED9/PMA1/RA4	26	AN9/C3INA/RPB15/SCK2/CTED6/PMCS1/RB15
13	VDD	27	AVss
14	PGED3/RPB5/PMD7/RB5	28	AVDD

- Note**
- 1: The RPn pins can be used by remappable peripherals. See Table 1 for the available peripherals and Section 11.3 “Peripheral Pin Select” for restrictions.
 - 2: Every I/O port pin (RAx-RCx) can be used as a change notification pin (CNAx-CNCx). See Section 11.0 “I/O Ports” for more information.
 - 3: Shaded pins are 5V tolerant.

PIC32MX1XX/2XX 28/36/44-PIN FAMILY

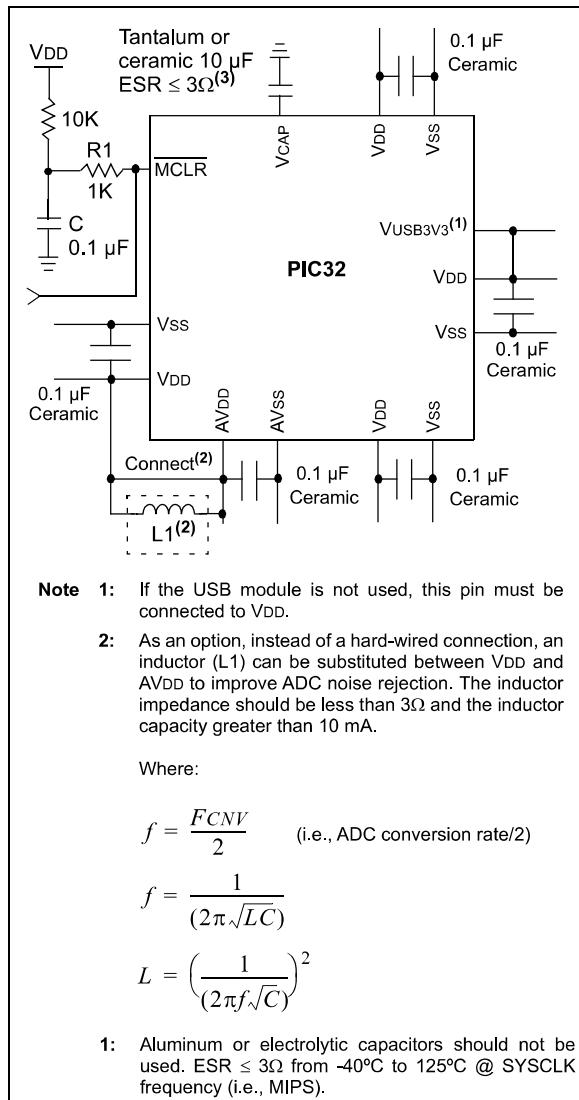
TABLE 11: PIN NAMES FOR 44-PIN GENERAL PURPOSE DEVICES

44-PIN TQFP (TOP VIEW) ^(1,2,3,5)	
Pin #	Full Pin Name
1	RPB9/SDA1/CTED4/PMD3/RB9
2	RPC6/PMA1/RC6
3	RPC7/PMA0/RC7
4	RPC8/PMA5/RC8
5	RPC9/CTED7/PMA6/RC9
6	Vss
7	Vcap
8	PGED2/RPB10/CTED11/PMD2/RB10
9	PGE2/RPB11/PMD1/RB11
10	AN12/PMD0/RB12
11	AN11/RPB13/CTPLS/PMRD/RB13
12	PGED4 ⁽⁴⁾ /TMS/PMA10/RA10
13	PGE4 ⁽⁴⁾ /TCK/CTED8/PMA7/RA7
14	CVREFOUT/AN10/C3INB/RPB14/SCK1/CTED5/PMWR/RB14
15	AN9/C3INA/RPB15/SCK2/CTED6/PMCS1/RB15
16	AVss
17	AVdd
18	MCLR
19	VREF+/CVREF+/AN0/C3INC/RPA0/CTED1/RA0
20	VREF-/CVREF-/AN1/RPA1/CTED2/RA1
21	PGED1/AN2/C1IND/C2INB/C3IND/RPB0/RB0
22	PGE1/AN3/C1INC/C2INA/RPB1/CTED12/RB1
Pin #	Full Pin Name
23	AN4/C1INB/C2IND/RPB2/SDA2/CTED13/RB2
24	AN5/C1INA/C2INC/RTCC/RPB3/SCL2/RB3
25	AN6/RPC0/RC0
26	AN7/RPC1/RC1
27	AN8/RPC2/PMA2/RC2
28	Vdd
29	Vss
30	OSC1/CLKI/RPA2/RA2
31	OSC2/CLKO/RPA3/RA3
32	TDO/RPA8/PMA8/RA8
33	SOSCI/RPB4/RB4
34	SOSCO/RPA4/T1CK/CTED9/RA4
35	TDI/RPA9/PMA9/RA9
36	RPC3/RC3
37	RPC4/PMA4/RC4
38	RPC5/PMA3/RC5
39	Vss
40	Vdd
41	PGED3/RPB5/PMD7/RB5
42	PGE3/RPB6/PMD6/RB6
43	RPB7/CTED3/PMD5/INT0/RB7
44	RPB8/SCL1/CTED10/PMD4/RB8

- Note**
- 1: The RPn pins can be used by remappable peripherals. See Table 1 for the available peripherals and Section 11.3 "Peripheral Pin Select" for restrictions.
 - 2: Every I/O port pin (RAx-RCx) can be used as a change notification pin (CNAx-CNCx). See Section 11.0 "I/O Ports" for more information.
 - 3: The metal plane at the bottom of the device is not connected to any pins and is recommended to be connected to Vss externally.
 - 4: This pin function is not available on PIC32MX110F016D and PIC32MX120F032D devices.
 - 5: Shaded pins are 5V tolerant.

PIC32MX1XX/2XX 28/36/44-PIN FAMILY

FIGURE 2-1: RECOMMENDED MINIMUM CONNECTION



2.2.1 BULK CAPACITORS

The use of a bulk capacitor is recommended to improve power supply stability. Typical values range from 4.7 μ F to 47 μ F. This capacitor should be located as close to the device as possible.

2.3 Capacitor on Internal Voltage Regulator (V_{CAP})

2.3.1 INTERNAL REGULATOR MODE

A low-ESR (3 ohm) capacitor is required on the VCAP pin, which is used to stabilize the internal voltage regulator output. The VCAP pin must not be connected to VDD, and must have a CEFC capacitor, with at least a 6V rating, connected to ground. The type can be ceramic or tantalum. Refer to **30.0 “Electrical Characteristics”** for additional information on CEFC specifications.

2.4 Master Clear (MCLR) Pin

The MCLR pin provides two specific device functions:

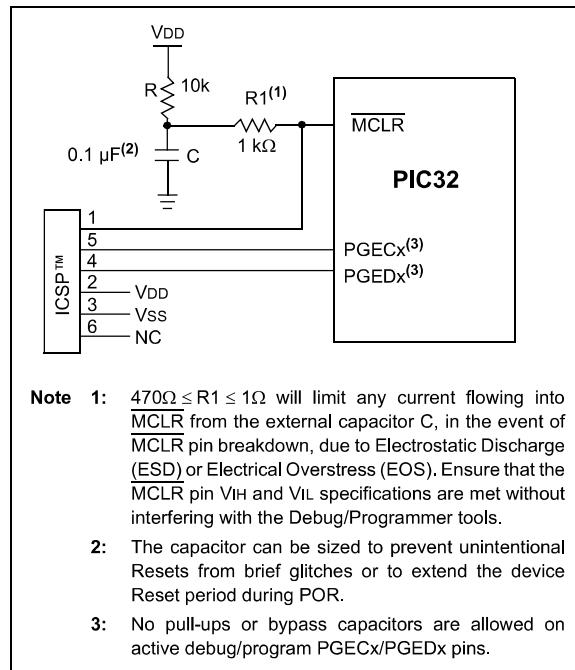
- Device Reset
 - Device programming and debugging

Pulling The MCLR pin low generates a device Reset. Figure 2-2 illustrates a typical MCLR circuit. During device programming and debugging, the resistance and capacitance that can be added to the pin must be considered. Device programmers and debuggers drive the MCLR pin. Consequently, specific voltage levels (V_{IH} and V_{IL}) and fast signal transitions must not be adversely affected. Therefore, specific values of R and C will need to be adjusted based on the application and PCB requirements.

For example, as illustrated in Figure 2-2, it is recommended that the capacitor C, be isolated from the MCLR pin during programming and debugging operations.

Place the components illustrated in Figure 2-2 within one-quarter inch (6 mm) from the MCLR pin.

FIGURE 2-2: EXAMPLE OF MCLR PIN CONNECTIONS



2.5 ICSP Pins

The PGECx and PGEDx pins are used for ICSP and debugging purposes. It is recommended to keep the trace length between the ICSP connector and the ICSP pins on the device as short as possible. If the ICSP connector is expected to experience an ESD event, a series resistor is recommended, with the value in the range of a few tens of Ohms, not to exceed 100 Ohms.

PIC32MX1XX/2XX 28/36/44-PIN FAMILY

TABLE 11-1: INPUT PIN SELECTION

Peripheral Pin	[pin name]R SFR	[pin name]R bits	[pin name]R Value to RPn Pin Selection
INT4	INT4R	INT4R<3:0>	0000 = RPA0 0001 = RPB3 0010 = RPB4 0011 = RPB15 0100 = RPB7 0101 = RPC7 ⁽²⁾ 0110 = RPC0 ⁽¹⁾ 0111 = RPC5 ⁽²⁾ 1000 = Reserved . . .
T2CK	T2CKR	T2CKR<3:0>	
IC4	IC4R	IC4R<3:0>	
<u>SS1</u>	SS1R	SS1R<3:0>	
REFCLKI	REFCLKIR	REFCLKIR<3:0>	1111 = Reserved
INT3	INT3R	INT3R<3:0>	0000 = RPA1 0001 = RPB5 0010 = RPB1 0011 = RPB11 0100 = RPB8 0101 = RPA8 ⁽²⁾ 0110 = RPC8 ⁽²⁾ 0111 = RPA9 ⁽²⁾ 1000 = Reserved . . .
T3CK	T3CKR	T3CKR<3:0>	
IC3	IC3R	IC3R<3:0>	
<u>U1CTS</u>	U1CTSR	U1CTSR<3:0>	
U2RX	U2RXR	U2RXR<3:0>	
SDI1	SDI1R	SDI1R<3:0>	1111 = Reserved
INT2	INT2R	INT2R<3:0>	0000 = RPA2 0001 = RPB6 0010 = RPA4 0011 = RPB13 0100 = RPB2 0101 = RPC6 ⁽²⁾ 0110 = RPC1 ⁽¹⁾ 0111 = RPC3 ⁽¹⁾ 1000 = Reserved . . .
T4CK	T4CKR	T4CKR<3:0>	
IC1	IC1R	IC1R<3:0>	
IC5	IC5R	IC5R<3:0>	
U1RX	U1RXR	U1RXR<3:0>	
<u>U2CTS</u>	U2CTSR	U2CTSR<3:0>	1000 = Reserved
SDI2	SDI2R	SDI2R<3:0>	
OCFB	OCFBR	OCFBR<3:0>	1111 = Reserved
INT1	INT1R	INT1R<3:0>	0000 = RPA3 0001 = RPB14 0010 = RPB0 0011 = RPB10 0100 = RPB9 0101 = RPC9 ⁽¹⁾ 0110 = RPC2 ⁽²⁾ 0111 = RPC4 ⁽²⁾ 1000 = Reserved . . .
T5CK	T5CKR	T5CKR<3:0>	
IC2	IC2R	IC2R<3:0>	
<u>SS2</u>	SS2R	SS2R<3:0>	
OCFA	OCFAR	OCFAR<3:0>	1111 = Reserved

Note 1: This pin is not available on 28-pin devices.

2: This pin is only available on 44-pin devices.

PIC32MX1XX/2XX 28/36/44-PIN FAMILY

TABLE 11-2: OUTPUT PIN SELECTION

RPn Port Pin	RPnR SFR	RPnR bits	RPnR Value to Peripheral Selection
RPA0	RPA0R	RPA0R<3:0>	0000 = No Connect 0001 = <u>U1TX</u> 0010 = <u>U2RTS</u> 0011 = SS1 0100 = Reserved 0101 = OC1 0110 = Reserved 0111 = C2OUT 1000 = Reserved
RPB3	RPB3R	RPB3R<3:0>	
RPB4	RPB4R	RPB4R<3:0>	
RPB15	RPB15R	RPB15R<3:0>	
RPB7	RPB7R	RPB7R<3:0>	
RPC7	RPC7R	RPC7R<3:0>	
RPC0	RPC0R	RPC0R<3:0>	
RPC5	RPC5R	RPC5R<3:0>	
RPA1	RPA1R	RPA1R<3:0>	0000 = No Connect 0001 = Reserved 0010 = Reserved 0011 = SDO1 0100 = SDO2 0101 = OC2 0110 = Reserved 0111 = C3OUT
RPB5	RPB5R	RPB5R<3:0>	
RPB1	RPB1R	RPB1R<3:0>	
RPB11	RPB11R	RPB11R<3:0>	
RPB8	RPB8R	RPB8R<3:0>	
RPA8	RPA8R	RPA8R<3:0>	
RPC8	RPC8R	RPC8R<3:0>	
RPA9	RPA9R	RPA9R<3:0>	1111 = Reserved
RPA2	RPA2R	RPA2R<3:0>	0000 = No Connect 0001 = Reserved 0010 = Reserved 0011 = SDO1 0100 = SDO2 0101 = OC4 0110 = OC5 0111 = REFCLKO 1000 = Reserved
RPB6	RPB6R	RPB6R<3:0>	
RPA4	RPA4R	RPA4R<3:0>	
RPB13	RPB13R	RPB13R<3:0>	
RPB2	RPB2R	RPB2R<3:0>	
RPC6	RPC6R	RPC6R<3:0>	
RPC1	RPC1R	RPC1R<3:0>	
RPC3	RPC3R	RPC3R<3:0>	1111 = Reserved
RPA3	RPA3R	RPA3R<3:0>	0000 = <u>No Connect</u> 0001 = <u>U1RTS</u> 0010 = U2TX 0011 = Reserved 0100 = SS2 0101 = OC3 0110 = Reserved 0111 = C1OUT 1000 = Reserved
RPB14	RPB14R	RPB14R<3:0>	
RPB0	RPB0R	RPB0R<3:0>	
RPB10	RPB10R	RPB10R<3:0>	
RPB9	RPB9R	RPB9R<3:0>	
RPC9	RPC9R	RPC9R<3:0>	
RPC2	RPC2R	RPC2R<3:0>	
RPC4	RPC4R	RPC4R<3:0>	1111 = Reserved

17.0 SERIAL PERIPHERAL INTERFACE (SPI)

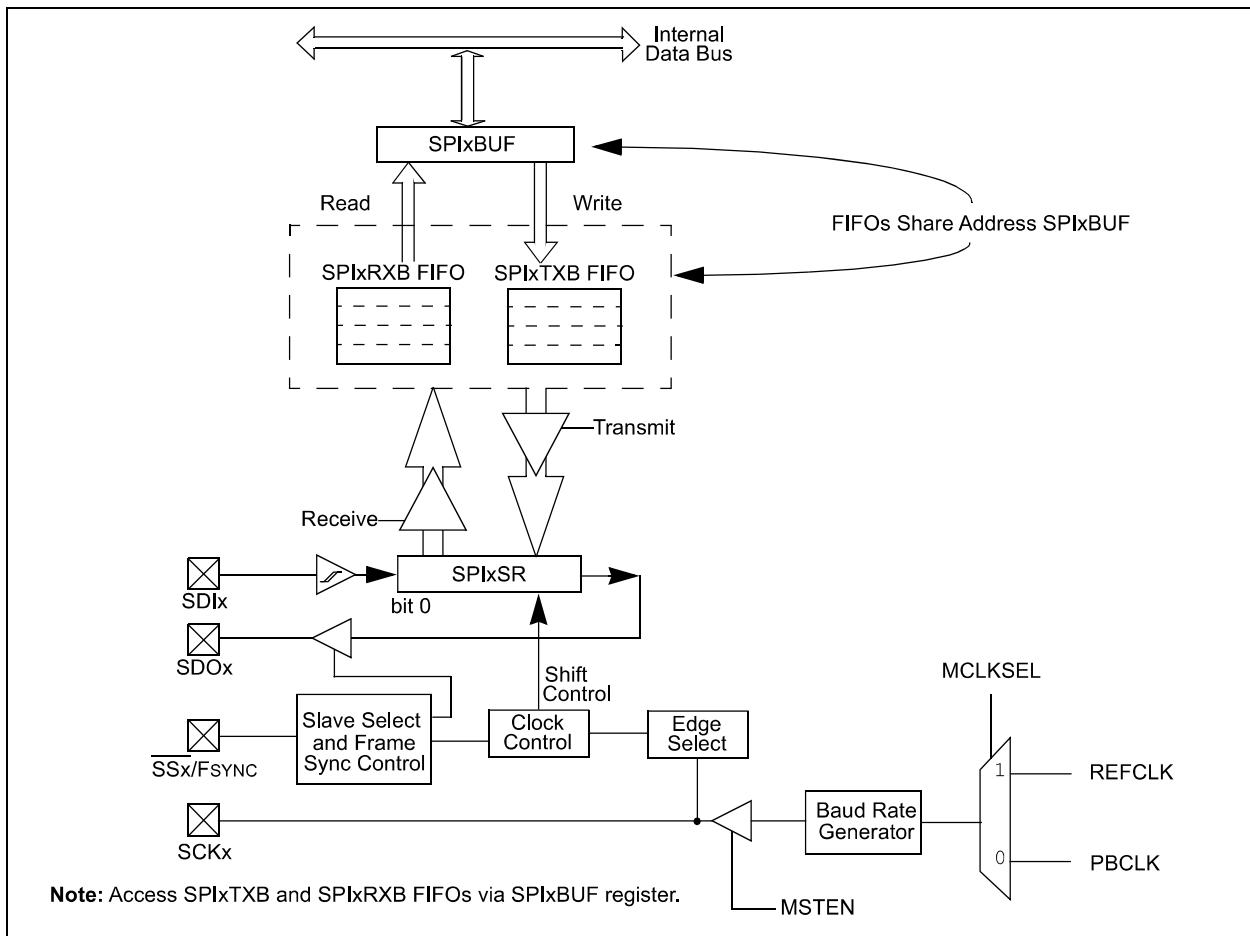
Note: This data sheet summarizes the features of the PIC32MX1XX/2XX 28/36/44-pin Family of devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to **Section 23. "Serial Peripheral Interface (SPI)"** (DS60001106), which is available from the *Documentation > Reference Manual* section of the Microchip PIC32 web site (www.microchip.com/pic32).

The SPI module is a synchronous serial interface that is useful for communicating with external peripherals and other microcontrollers. These peripheral devices may be Serial EEPROMs, Shift registers, display drivers, Analog-to-Digital Converters (ADC), etc. The PIC32 SPI module is compatible with Motorola® SPI and SIOP interfaces.

Some of the key features of the SPI module are:

- Master mode and Slave mode support
- Four clock formats
- Enhanced Framed SPI protocol support
- User-configurable 8-bit, 16-bit and 32-bit data width
- Separate SPI FIFO buffers for receive and transmit
 - FIFO buffers act as 4/8/16-level deep FIFOs based on 32/16/8-bit data width
- Programmable interrupt event on every 8-bit, 16-bit and 32-bit data transfer
- Operation during Sleep and Idle modes
- Audio Codec Support:
 - I²S protocol
 - Left-justified
 - Right-justified
 - PCM

FIGURE 17-1: SPI MODULE BLOCK DIAGRAM



PIC32MX1XX/2XX 28/36/44-PIN FAMILY

19.0 UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER (UART)

Note: This data sheet summarizes the features of the PIC32MX1XX/2XX 28/36/44-pin Family of devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to **Section 21. “Universal Asynchronous Receiver Transmitter (UART)”** (DS60001107), which is available from the *Documentation > Reference Manual* section of the Microchip PIC32 web site (www.microchip.com/pic32).

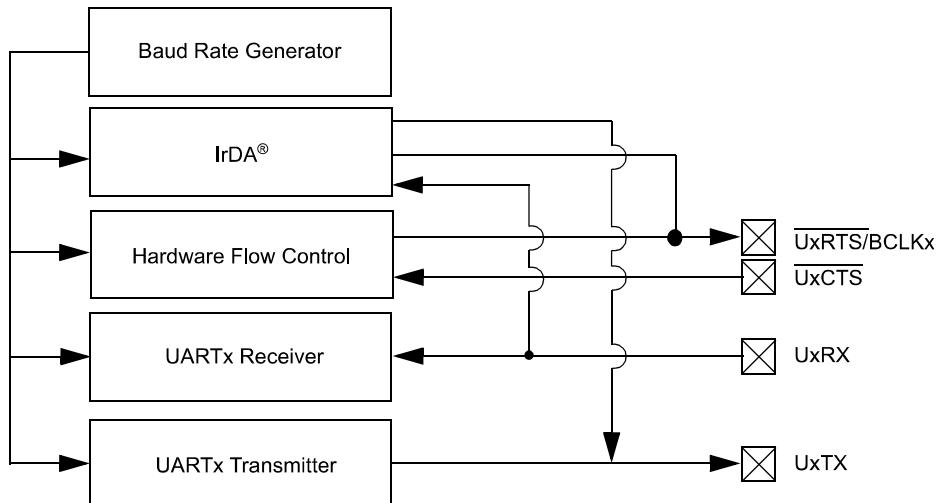
The UART module is one of the serial I/O modules available in PIC32MX1XX/2XX 28/36/44-pin Family devices. The UART is a full-duplex, asynchronous communication channel that communicates with peripheral devices and personal computers through protocols, such as RS-232, RS-485, LIN, and IrDA®. The UART module also supports the hardware flow control option, with UxCTS and UxRTS pins, and also includes an IrDA encoder and decoder.

Key features of the UART module include:

- Full-duplex, 8-bit or 9-bit data transmission
- Even, Odd or No Parity options (for 8-bit data)
- One or two Stop bits
- Hardware auto-baud feature
- Hardware flow control option
- Fully integrated Baud Rate Generator (BRG) with 16-bit prescaler
- Baud rates ranging from 38 bps to 12.5 Mbps at 50 MHz
- 8-level deep First In First Out (FIFO) transmit data buffer
- 8-level deep FIFO receive data buffer
- Parity, framing and buffer overrun error detection
- Support for interrupt-only on address detect (9th bit = 1)
- Separate transmit and receive interrupts
- Loopback mode for diagnostic support
- LIN protocol support
- IrDA encoder and decoder with 16x baud clock output for external IrDA encoder/decoder support

Figure 19-1 illustrates a simplified block diagram of the UART module.

FIGURE 19-1: UART SIMPLIFIED BLOCK DIAGRAM



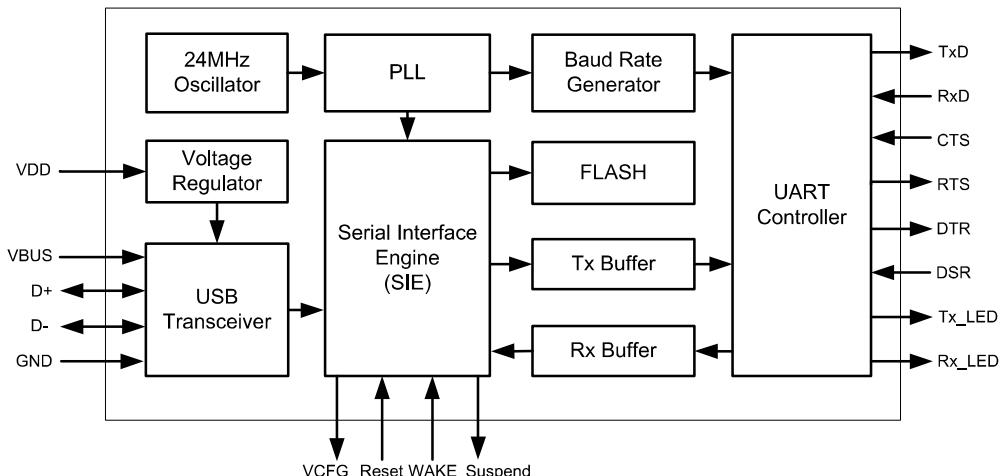
Note: Not all pins are available for all UART modules. Refer to the device-specific pin diagram for more information.

Features

- Universal Serial Bus (USB) Integration
 - Full-Speed USB peripheral compliant with USB2.0 specification
 - USB-IF certified with TID 40001425
 - Support for bus-powered and self-powered configurations
 - 3 endpoints (1 Interrupt IN, 1 Bulk OUT and 1 Bulk IN)
 - Integrated USB transceiver, 1.5 kΩ pull-up resistor on D+ line
- Universal Asynchronous Receiver Transmitter (UART)
 - Baud rate generation (300 to 230400)
 - Data format:
 - 8 data bits
 - 1 stop bit
 - No parity, even parity or odd parity
 - Support for Parity, Overrun and Framing errors
 - Supports flow control using CTS,RTS,DTR, DSR
 - LED signals to indicate activity on TxD and RxD lines

- Full device operation from a single voltage supply of 3.3 V or 5 V
- Low power consumption in suspend mode
 - 225 µA at 5 V operating voltage
 - 207 µA at 3.3 V operating voltage
- Integrated 24 MHz oscillator
- Integrated 3.3 V regulator
- Integrated flash to store device configuration
- Software support for ease of development
 - Configuration utility to program device parameters such as VID, PID and string descriptors.
 - Certified Cypress VCP driver for Windows (8 / 7 / Vista / XP)
 - Support for device drivers for Android, Mac, Linux, Window CE 4.2, 5.0, 6.0
- 28-pin SSOP 10 mm × 7.5 mm, RoHS compliant package
- Temperature grade
 - Commercial operating temperature range of 0 °C to +70 °C

Figure 1. CY7C64225 Block Diagram



Pin Configuration

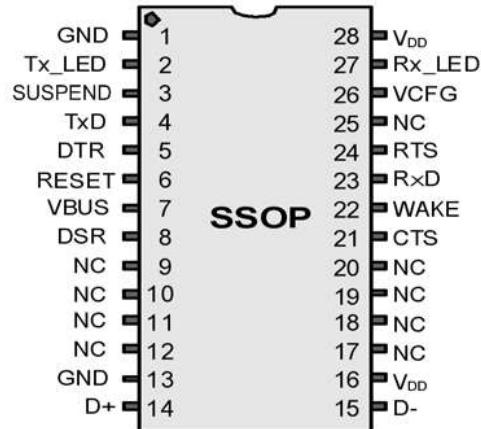
28-pin part pinout Description

The CY7C64225 USB-to-UART Bridge device is available in a 28-pin package as shown in Figure 2. The pin description is listed in Table 2.

Table 2. 28-pin part pinout (SSOP)

Pin No.	Name	I/O	Description
1	GND	Power	Ground
2	Tx_LED	Output	Active low, UART Tx_LED, max current -20 mA
3	SUSPEND	Output	Active low indicates USB is suspended
4	TxD	Output	UART Data Transmit, Output
5	DTR	Output	Data Terminal Ready (DTR) Pin
6	RESET	Input	No Connect (NC)
7	VBUS	Input	Used for VBUS monitoring. This pin requires a series resistor when connected to VBUS. The recommended values are in the range of 1 kΩ–10 kΩ.
8	DSR	Input	Data Set Ready (DSR) pin
13	GND	Power	USB Ground
14	D+	USB	USB D+ Line
15	D-	USB	USB D- Line
16	V _{DD}	Power	Supply Voltage (3.3 V or 5 V)
21	CTS	Input	Clear to Send (CTS) input, handshake signal
22	WAKE	Input	Active high on this pin, generates Remote Wake-Up signal on the Bus
23	RxD	Input	UART Data Receive, Input
24	RTS	Output	Request to Sent (RTS) output, handshake signal
26	VCFG	Output	Active low indicates VBUS is detected and device is configured
27	Rx_LED	Output	Active low, UART Rx_LED, max current -20 mA
28	V _{DD}	Power	Supply Voltage. 3.3 V or 5 V
9	NC	NC	No Connect
10	NC	NC	No Connect
11	NC	NC	No Connect
12	NC	NC	No Connect
17	NC	NC	No Connect
18	NC	NC	No Connect
19	NC	NC	No Connect
20	NC	NC	No Connect
25	NC	NC	No Connect

Figure 2. CY7C64225 USB-UART Bridge Device



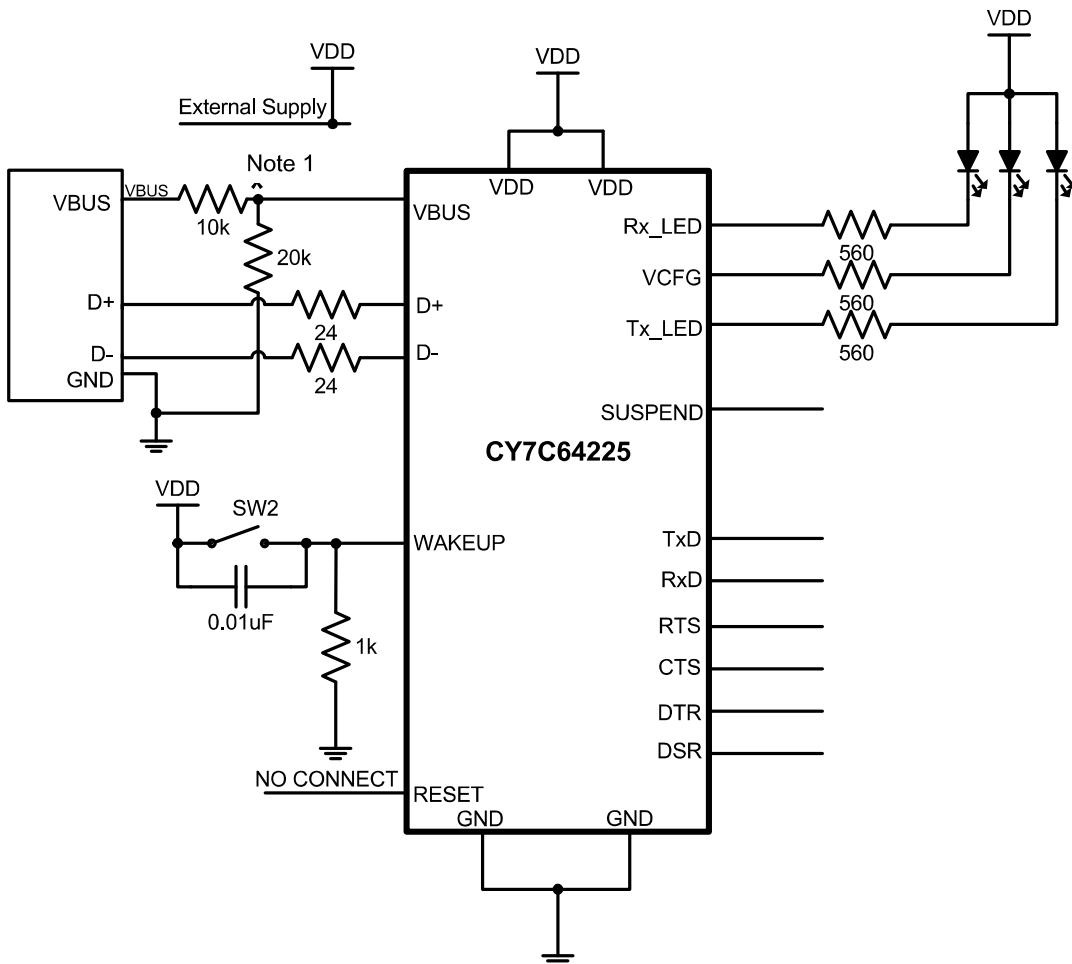
Self Powered Design

Figure 5 illustrates the use of CY7C64225 in self powered mode operating at 3.3 V. VDD is obtained from an external power supply. As shown in Figure 5, a voltage divider circuit is used to

provide 3.3 V from VBUS of USB port to VBUS pin of CY7C64225.

A self powered device can draw more current for its operation from external supply during USB active mode as well as suspend mode as this will not affect the operation of the USB.

Figure 5. Self Powered Design (VDD = 3.3 V)



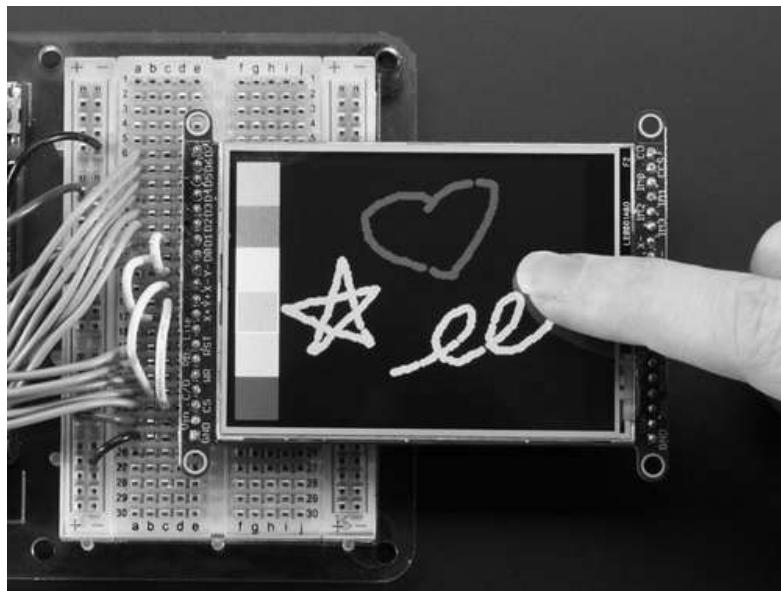
Note

1. Replace the voltage divider circuit (10K and 20K resistors) with 1K series resistor as shown in Figure 3, if 5 V is applied at VDD in this design.



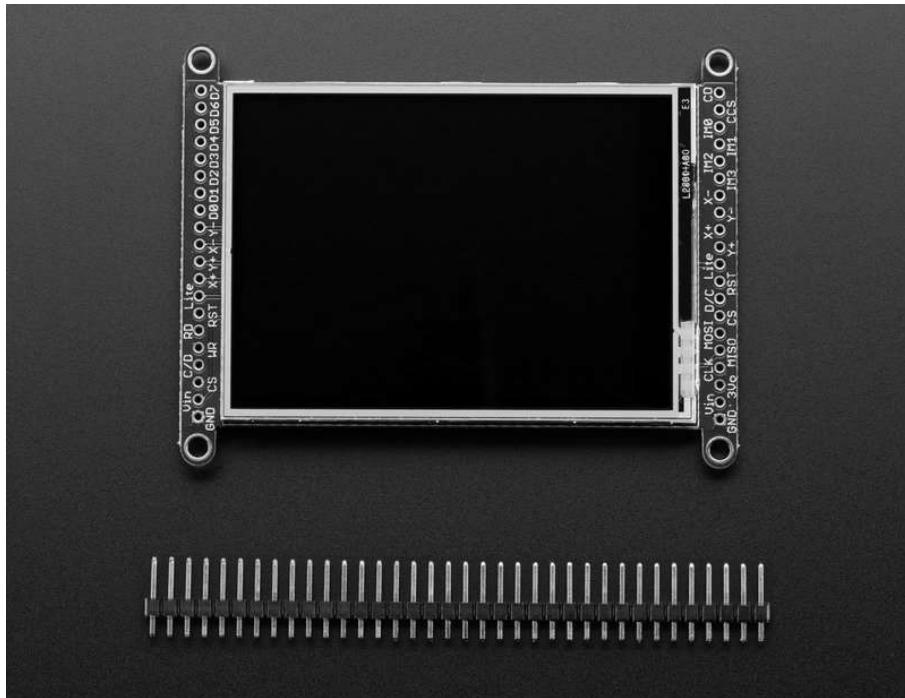
Adafruit 2.8" and 3.2" Color TFT Touchscreen Breakout v2

Created by lady ada



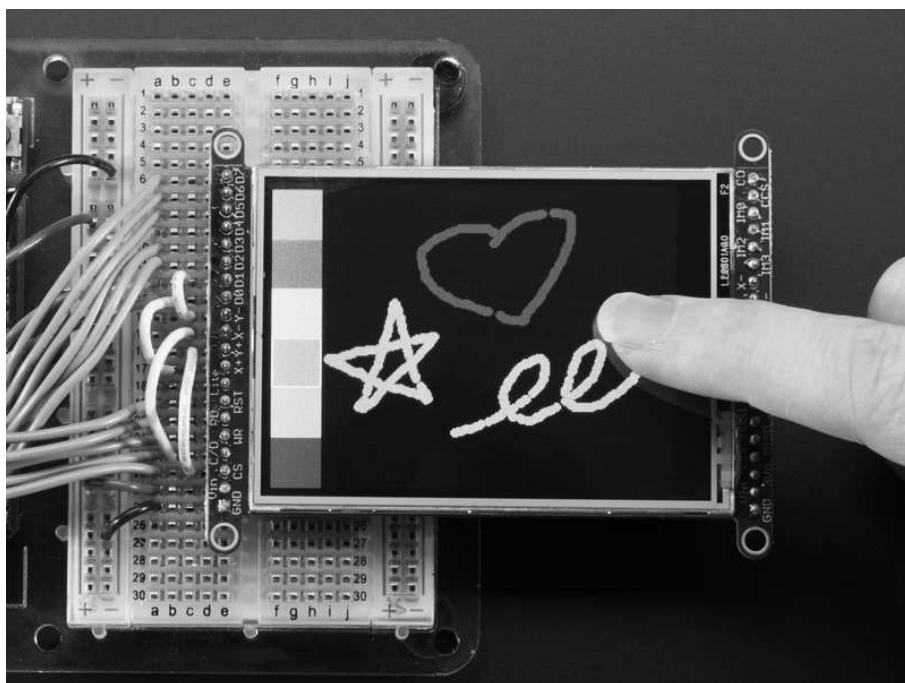
Last updated on 2020-06-19 03:59:40 PM EDT

Overview



Add some jazz & pizazz to your project with a color touchscreen LCD. These TFT displays are big (2.8" or 3.2" diagonal) bright (4 or 6 white-LED backlight) and colorful! 240x320 pixels with individual RGB pixel control, this has way more resolution than a black and white 128x64 display.

As a bonus, this display has either a resistive or capacitive touchscreen attached to it already, so you can detect finger presses anywhere on the screen.



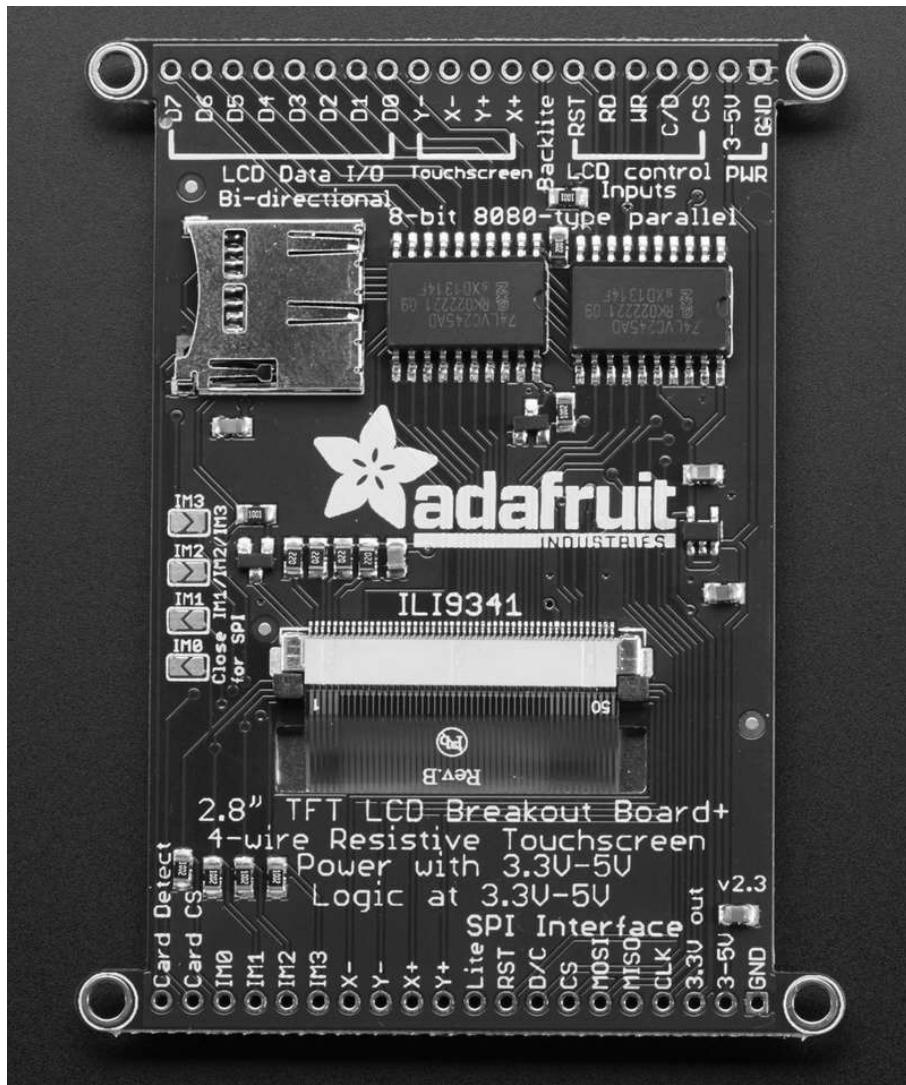
Pinouts

The 2.8" and 3.2" TFT display on this breakout supports many different modes - so many that the display itself has 50 pins. However, we think most people really only use 2 different modes, either "SPI" mode or 8-bit mode (which includes both 6800 and 8080). Each 'side' of the display has all the pins required for that mode. You can switch between modes, by rewiring the display, but it cannot be used in two modes at the same time!

All logic pins, both 8-bit and SPI sides, are 3-5V logic level compatible, the 74LVX245 chips on the back perform fast level shifting so you can use either kind of logic levels. If there's data output, the levels are at 3.3V



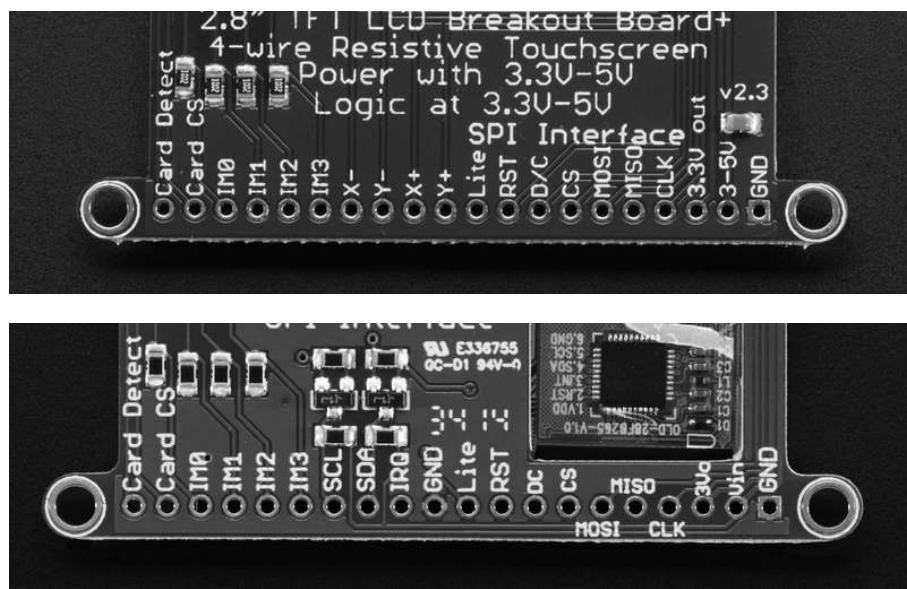
We show the 2.8" version of this breakout in the photos below but the 3.2" TFT is identical, just a lil bit bigger



SPI Mode

This is what we think will be a popular mode when speed is not of the utmost importance. It doesn't use as many pins (only 4 to draw on the TFT if you skip the MISO pin), is fairly flexible, and easy to port to various microcontrollers. It also allows using a microSD card socket on the same SPI bus. However, its slower than parallel 8-bit mode because you

have to send each bit at a time instead of 8-bits at a time. Tradeoffs!



- **GND** - this is the power and signal ground pin
- **3-5V / Vin** - this is the power pin, connect to 3-5VDC - it has reverse polarity protection but try to wire it right!
- **3.3Vout** - this is the 3.3V output from the onboard regulator
- **CLK** - this is the SPI clock input pin
- **MISO** - this is the SPI Microcontroller In Serial Out pin, its used for the SD card mostly, and for debugging the TFT display. It isn't necessary for using the TFT display which is write-only
- **MOSI** - this is the SPI Microcontroller Out Serial In pin, it is used to send data from the microcontroller to the SD card and/or TFT
- **CS** - this is the TFT SPI chip select pin
- **D/C** - this is the TFT SPI data or command selector pin
- **RST** - this is the TFT reset pin. There's auto-reset circuitry on the breakout so this pin is not required but it can be helpful sometimes to reset the TFT if your setup is not always resetting cleanly. Connect to ground to reset the TFT
- **Lite** - this is the PWM input for the backlight control. It is by default pulled high (backlight on) you can PWM at any frequency or pull down to turn the backlight off
- **IM3 IM2 IM1 IM0** - these are interface control set pins. In general these breakouts aren't used, and instead the onboard jumpers are used to fix the interface to SPI or 8-bit. However, we break these out for advanced use and also for our test procedures
- **Card CS / CCS** - this is the SD card chip select, used if you want to read from the SD card.
- **Card Detect / CD** - this is the SD card detect pin, it floats when a card is inserted, and tied to ground when the card is not inserted. We don't use this in our code but you can use this as a switch to detect if an SD card is in place without trying to electrically query it. Don't forget to use a pullup on this pin if so!

Resistive touch pins

- **Y+ X+ Y- X-** these are the 4 resistive touch screen pads, which can be read with analog pins to determine touch points. They are completely separated from the TFT electrically (the overlay is glued on top)

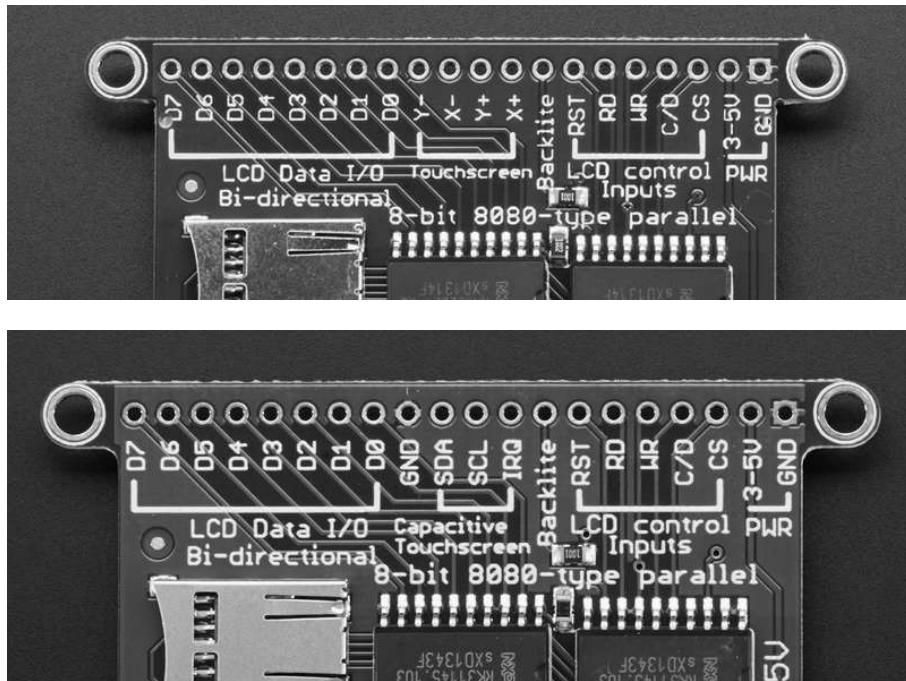
Capacitive touch pins

- **SDA** - this is the I2C data pin for the captouch chip, there's level shifting on this pin so you can use 3-5V logic.

- There's also a 10K pullup
- **SCL** - this is the I²C clock pin for the captouch chip, there's level shifting on this pin so you can use 3-5V logic.
There's also a 10K pullup
 - **IRQ** - this is the captouch interrupt pin. When a touch is detected, this pin goes low.

8-Bit Mode

This mode is for when you have lots of pins and want more speed. In this mode we send 8 bits at a time, so it needs way more pins, 12 or so (8 bits plus 4 control)! This isn't recommended because most microcontrollers don't have a ton of pins and also we optimize our libraries for SPI!



- **GND** - this is the power and signal ground pin
- **3-5V (Vin)** - this is the power pin, connect to 3-5VDC - it has reverse polarity protection but try to wire it right!
- **CS** - this is the TFT 8-bit chip select pin (it is also tied to the SPI mode **CS** pin)
- **C/D** - this is the TFT 8-bit data or command selector pin. It is **not the same as the SPI D/C pin!** Instead, it's the same as the SPI CLK pin.
- **WR** - this is the TFT 8-bit write strobe pin. It is also connected to the SPI D/C pin
- **RD** - this is the TFT 8-bit read strobe pin. You may not need this pin if you don't want to read data from the display
- **RST** - this is the TFT reset pin. There's auto-reset circuitry on the breakout so this pin is not required but it can be helpful sometimes to reset the TFT if your setup is not always resetting cleanly. Connect to ground to reset the TFT
- **Backkite** - this is the PWM input for the backlight control. It is by default pulled high (backlight on) you can PWM at any frequency or pull down to turn the backlight off
- **D0 thru D7** - these are the 8 bits of parallel data sent to the TFT in 8-bit mode. **D0** is the least-significant-bit and **D7** is the MSB



WP7113LGD

T-1 3/4 (5mm) Solid State Lamp

DESCRIPTION

- The Green source color devices are made with Gallium Phosphide Green Light Emitting Diode

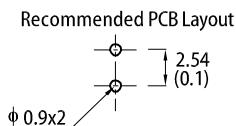
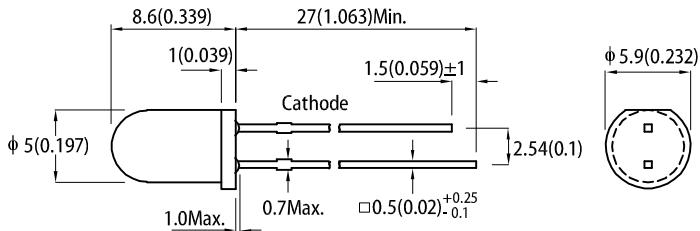
FEATURES

- Low power consumption
- Popular T-1 3/4 diameter package
- General purpose leads
- Reliable and rugged
- Long life - solid state reliability
- Available on tape and reel
- RoHS compliant

APPLICATIONS

- Status indicator
- Illuminator
- Signage applications
- Decorative and entertainment lighting
- Commercial and residential architectural lighting

PACKAGE DIMENSIONS



Notes:

- All dimensions are in millimeters (inches).
- Tolerance is ±0.25(0.01") unless otherwise noted.
- Lead spacing is measured where the leads emerge from the package.
- The specifications, characteristics and technical data described in the datasheet are subject to change without prior notice.

SELECTION GUIDE

Part Number	Emitting Color (Material)	Lens Type	I _v (mcd) @ 2mA ^[2]		Viewing Angle ^[1] 2θ/2
			Min.	Typ.	
WP7113LGD	Green (GaP)	Green Diffused	1.2	3	30°

Notes:

- θ/2 is the angle from optical centerline where the luminous intensity is 1/2 of the optical peak value.
- Luminous intensity / luminous flux: +/-15%.
- Luminous intensity value is traceable to CIE127-2007 standards.

ELECTRICAL / OPTICAL CHARACTERISTICS at $T_A=25^\circ\text{C}$

Parameter	Symbol	Emitting Color	Value		Unit
			Typ.	Max.	
Wavelength at Peak Emission $I_F = 2\text{mA}$	λ_{peak}	Green	565	-	nm
Dominant Wavelength $I_F = 2\text{mA}$	$\lambda_{\text{dom}}^{[1]}$	Green	568	-	nm
Spectral Bandwidth at 50% Φ REL MAX $I_F = 2\text{mA}$	$\Delta\lambda$	Green	30	-	nm
Capacitance	C	Green	15	-	pF
Forward Voltage $I_F = 2\text{mA}$	$V_F^{[2]}$	Green	1.9	2.25	V
Reverse Current ($V_R = 5\text{V}$)	I_R	Green	-	10	μA
Temperature Coefficient of λ_{peak} $I_F = 2\text{mA}$, $-10^\circ\text{C} \leq T \leq 85^\circ\text{C}$	$\text{TC}_{\lambda_{\text{peak}}}$	Green	0.1	-	$\text{nm}/^\circ\text{C}$
Temperature Coefficient of λ_{dom} $I_F = 2\text{mA}$, $-10^\circ\text{C} \leq T \leq 85^\circ\text{C}$	$\text{TC}_{\lambda_{\text{dom}}}$	Green	0.06	-	$\text{nm}/^\circ\text{C}$
Temperature Coefficient of V_F $I_F = 2\text{mA}$, $-10^\circ\text{C} \leq T \leq 85^\circ\text{C}$	TC_V	Green	-2	-	$\text{mV}/^\circ\text{C}$

Notes:

1. The dominant wavelength (λ_d) above is the setup value of the sorting machine. (Tolerance $\lambda_d : \pm 1\text{nm}$.)
2. Forward voltage: $\pm 0.1\text{V}$.
3. Wavelength value is traceable to CIE127-2007 standards.
4. Excess driving current and / or operating temperature higher than recommended conditions may result in severe light degradation or premature failure.

ABSOLUTE MAXIMUM RATINGS at $T_A=25^\circ\text{C}$

Parameter	Symbol	Value	Unit
Power Dissipation	P_D	62.5	mW
Reverse Voltage	V_R	5	V
Junction Temperature	T_j	110	$^\circ\text{C}$
Operating Temperature	T_{op}	-40 to +85	$^\circ\text{C}$
Storage Temperature	T_{stg}	-40 to +85	$^\circ\text{C}$
DC Forward Current	I_F	25	mA
Peak Forward Current	$I_{\text{FM}}^{[1]}$	140	mA
Electrostatic Discharge Threshold (HBM)	-	8000	V
Thermal Resistance (Junction / Ambient)	$R_{\text{th JA}}^{[2]}$	590	$^\circ\text{C/W}$
Thermal Resistance (Junction / Solder point)	$R_{\text{th JS}}^{[2]}$	460	$^\circ\text{C/W}$
Lead Solder Temperature ^[3]		260 $^\circ\text{C}$ For 3 Seconds	
Lead Solder Temperature ^[4]		260 $^\circ\text{C}$ For 5 Seconds	

Notes:

1. 1/10 Duty Cycle, 0.1ms Pulse Width.
2. $R_{\text{th JA}}, R_{\text{th JS}}$ Results from mounting on PC board FR4 (pad size $\geq 16\text{ mm}^2$ per pad).
3. 2mm below package base.
4. 5mm below package base.
5. Relative humidity levels maintained between 40% and 60% in production area are recommended to avoid the build-up of static electricity – Ref JEDEC/JESD625-A and JEDEC/J-STD-033.

WP7113LID T-1 3/4 (5 mm) Solid State Lamp



DESCRIPTION

- The High Efficiency Red source color devices are made with Gallium Arsenide Phosphide on Gallium Phosphide Orange Light Emitting Diode

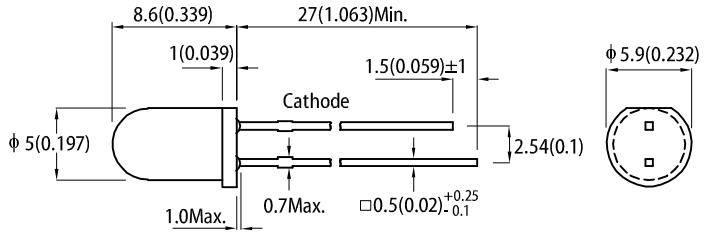
FEATURES

- Low power consumption
- Popular T-1 3/4 diameter package
- General purpose leads
- Reliable and rugged
- Long life - solid state reliability
- Available on tape and reel
- RoHS compliant

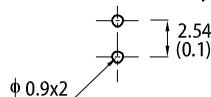
APPLICATIONS

- Status indicator
- Illuminator
- Signage applications
- Decorative and entertainment lighting
- Commercial and residential architectural lighting

PACKAGE DIMENSIONS



Recommended PCB Layout



Notes:

- All dimensions are in millimeters (inches).
- Tolerance is ±0.25(0.01") unless otherwise noted.
- Lead spacing is measured where the leads emerge from the package.
- The specifications, characteristics and technical data described in the datasheet are subject to change without prior notice.

SELECTION GUIDE

Part Number	Emitting Color (Material)	Lens Type	Iv (mcd) @ 2mA ^[2]		Viewing Angle ^[1] 2θ/2
			Min.	Typ.	
WP7113LID	■ High Efficiency Red (GaAsP/GaP)	Red Diffused	1.2	4	30°
			*0.7	*2	

Notes:

- θ/2 is the angle from optical centerline where the luminous intensity is 1/2 of the optical peak value.
- Luminous intensity / luminous flux: +/-15%.
- * Luminous intensity value is traceable to CIE127-2007 standards.

ELECTRICAL / OPTICAL CHARACTERISTICS at $T_A=25^\circ C$

Parameter	Symbol	Emitting Color	Value		Unit
			Typ.	Max.	
Wavelength at Peak Emission $I_F = 2\text{mA}$	λ_{peak}	High Efficiency Red	627	-	nm
Dominant Wavelength $I_F = 2\text{mA}$	$\lambda_{\text{dom}}^{[1]}$	High Efficiency Red	617	-	nm
Spectral Bandwidth at 50% Φ REL MAX $I_F = 2\text{mA}$	$\Delta\lambda$	High Efficiency Red	45	-	nm
Capacitance	C	High Efficiency Red	15	-	pF
Forward Voltage $I_F = 2\text{mA}$	$V_F^{[2]}$	High Efficiency Red	1.7	2.1	V
Reverse Current ($V_R = 5\text{V}$)	I_R	High Efficiency Red	-	10	μA
Temperature Coefficient of λ_{peak} $I_F = 2\text{mA}$, $-10^\circ C \leq T \leq 85^\circ C$	$TC_{\lambda_{\text{peak}}}$	High Efficiency Red	0.12	-	$\text{nm}/^\circ C$
Temperature Coefficient of λ_{dom} $I_F = 2\text{mA}$, $-10^\circ C \leq T \leq 85^\circ C$	$TC_{\lambda_{\text{dom}}}$	High Efficiency Red	0.06	-	$\text{nm}/^\circ C$
Temperature Coefficient of V_F $I_F = 2\text{mA}$, $-10^\circ C \leq T \leq 85^\circ C$	TC_V	High Efficiency Red	-1.9	-	$\text{mV}/^\circ C$

Notes:

1. The dominant wavelength (λ_d) above is the setup value of the sorting machine. (Tolerance $\lambda_d : \pm 1\text{nm}$.)
2. Forward voltage: $\pm 0.1\text{V}$.
3. Wavelength value is traceable to CIE127-2007 standards.
4. Excess driving current and / or operating temperature higher than recommended conditions may result in severe light degradation or premature failure.

ABSOLUTE MAXIMUM RATINGS at $T_A=25^\circ C$

Parameter	Symbol	Value	Unit
Power Dissipation	P_D	75	mW
Reverse Voltage	V_R	5	V
Junction Temperature	T_j	125	$^\circ C$
Operating Temperature	T_{op}	-40 to +85	$^\circ C$
Storage Temperature	T_{stg}	-40 to +85	$^\circ C$
DC Forward Current	I_F	30	mA
Peak Forward Current	$I_{FM}^{[1]}$	160	mA
Electrostatic Discharge Threshold (HBM)	-	8000	V
Thermal Resistance (Junction / Ambient)	$R_{th JA}^{[2]}$	560	$^\circ C/W$
Thermal Resistance (Junction / Solder point)	$R_{th JS}^{[2]}$	390	$^\circ C/W$
Lead Solder Temperature ^[3]		260°C For 3 Seconds	
Lead Solder Temperature ^[4]		260°C For 5 Seconds	

Notes:

1. 1/10 Duty Cycle, 0.1ms Pulse Width.
2. $R_{th JA}, R_{th JS}$ Results from mounting on PC board FR4 (pad size $\geq 16\text{ mm}^2$ per pad).
3. 2mm below package base.
4. 5mm below package base.
5. Relative humidity levels maintained between 40% and 60% in production area are recommended to avoid the build-up of static electricity – Ref JEDEC/JESD625-A and JEDEC/J-STD-033.



WP7113ND T-1 3/4 (5mm) Solid State Lamp

DESCRIPTION

- The Pure Orange source color devices are made with Gallium Arsenide Phosphide on Gallium Phosphide Pure Orange Light Emitting Diode

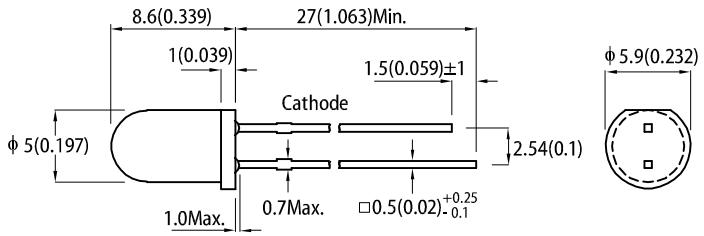
FEATURES

- Low power consumption
- Popular T-1 3/4 diameter package
- General purpose leads
- Reliable and rugged
- Long life - solid state reliability
- Available on tape and reel
- RoHS compliant

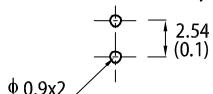
APPLICATIONS

- Status indicator
- Illuminator
- Signage applications
- Decorative and entertainment lighting
- Commercial and residential architectural lighting

PACKAGE DIMENSIONS



Recommended PCB Layout



Notes:

- All dimensions are in millimeters (inches).
- Tolerance is ±0.25(0.01") unless otherwise noted.
- Lead spacing is measured where the leads emerge from the package.
- The specifications, characteristics and technical data described in the datasheet are subject to change without prior notice.

SELECTION GUIDE

Part Number	Emitting Color (Material)	Lens Type	I _v (mcd) @ 10mA ^[2]		Viewing Angle ^[1] θ _{1/2}
			Min.	Typ.	
WP7113ND	■ Pure Orange (GaAsP/GaP)	Orange Diffused	20	45	30°
			*12	*30	

Notes:

- θ_{1/2} is the angle from optical centerline where the luminous intensity is 1/2 of the optical peak value.
- Luminous intensity / luminous flux: +/-15%.
- * Luminous intensity value is traceable to CIE127-2007 standards.

ELECTRICAL / OPTICAL CHARACTERISTICS at $T_A=25^\circ C$

Parameter	Symbol	Emitting Color	Value		Unit
			Typ.	Max.	
Wavelength at Peak Emission $I_F = 10mA$	λ_{peak}	Pure Orange	607	-	nm
Dominant Wavelength $I_F = 10mA$	λ_{dom} [1]	Pure Orange	602	-	nm
Spectral Bandwidth at 50% Φ REL MAX $I_F = 10mA$	$\Delta\lambda$	Pure Orange	35	-	nm
Capacitance	C	Pure Orange	15	-	pF
Forward Voltage $I_F = 10mA$	V_F [2]	Pure Orange	1.95	2.3	V
Reverse Current ($V_R = 5V$)	I_R	Pure Orange	-	10	μA
Temperature Coefficient of λ_{peak} $I_F = 10mA$, $-10^\circ C \leq T \leq 85^\circ C$	TC _{λ_{peak}}	Pure Orange	0.13	-	nm/ $^\circ C$
Temperature Coefficient of λ_{dom} $I_F = 10mA$, $-10^\circ C \leq T \leq 85^\circ C$	TC _{λ_{dom}}	Pure Orange	0.06	-	nm/ $^\circ C$
Temperature Coefficient of V_F $I_F = 10mA$, $-10^\circ C \leq T \leq 85^\circ C$	TC _V	Pure Orange	-1.9	-	mV/ $^\circ C$

Notes:

1. The dominant wavelength (λ_d) above is the setup value of the sorting machine. (Tolerance $\lambda_d : \pm 1nm$.)
2. Forward voltage: $\pm 0.1V$.
3. Wavelength value is traceable to CIE127-2007 standards.
4. Excess driving current and / or operating temperature higher than recommended conditions may result in severe light degradation or premature failure.

ABSOLUTE MAXIMUM RATINGS at $T_A=25^\circ C$

Parameter	Symbol	Value	Unit
Power Dissipation	P_D	62.5	mW
Reverse Voltage	V_R	5	V
Junction Temperature	T_j	125	$^\circ C$
Operating Temperature	T_{op}	-40 to +85	$^\circ C$
Storage Temperature	T_{stg}	-40 to +85	$^\circ C$
DC Forward Current	I_F	25	mA
Peak Forward Current	I_{FM} [1]	145	mA
Electrostatic Discharge Threshold (HBM)	-	8000	V
Thermal Resistance (Junction / Ambient)	$R_{th JA}$ [2]	560	$^\circ C/W$
Thermal Resistance (Junction / Solder point)	$R_{th JS}$ [2]	350	$^\circ C/W$
Lead Solder Temperature [3]		260°C For 3 Seconds	
Lead Solder Temperature [4]		260°C For 5 Seconds	

Notes:

1. 1/10 Duty Cycle, 0.1ms Pulse Width.
2. $R_{th JA}$, $R_{th JS}$ Results from mounting on PC board FR4 (pad size $\geq 16 mm^2$ per pad).
3. 2mm below package base.
4. 5mm below package base.
5. Relative humidity levels maintained between 40% and 60% in production area are recommended to avoid the build-up of static electricity – Ref JEDEC/JESD625-A and JEDEC/J-STD-033.

WP424IDT T-1 (3mm) Cylindrical LED Lamp



DESCRIPTION

- The High Efficiency Red source color devices are made with Gallium Arsenide Phosphide on Gallium Phosphide Orange Light Emitting Diode

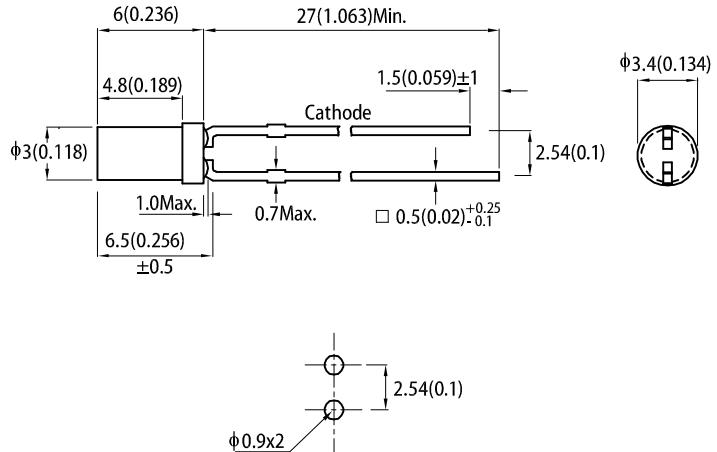
FEATURES

- Cylindrical type
- Low power consumption
- Reliable and rugged
- Long life - solid state reliability
- Available on tape and reel
- Halogen Free
- RoHS compliant

APPLICATIONS

- Status indicator
- Illuminator
- Signage applications
- Decorative and entertainment lighting
- Commercial and residential architectural lighting

PACKAGE DIMENSIONS



Notes:

- All dimensions are in millimeters (inches).
- Tolerance is ±0.25(0.01") unless otherwise noted.
- Lead spacing is measured where the leads emerge from the package.
- The specifications, characteristics and technical data described in the datasheet are subject to change without prior notice.

SELECTION GUIDE

Part Number	Emitting Color (Material)	Lens Type	Iv (mcd) @ 10mA [2]		Viewing Angle [1]
			Min.	Typ.	
WP424IDT	■ High Efficiency Red (GaAsP/GaP)	Red Diffused	2	6	140°
			*1.2	*4	

Notes:

- θ1/2 is the angle from optical centerline where the luminous intensity is 1/2 of the optical peak value.
- Luminous intensity /luminous flux: +/-15%.

* Luminous intensity value is traceable to CIE127-2007 standards.

ELECTRICAL / OPTICAL CHARACTERISTICS at $T_A=25^\circ C$

Parameter	Symbol	Emitting Color	Value		Unit
			Typ.	Max.	
Wavelength at Peak Emission $I_F = 10mA$	λ_{peak}	High Efficiency Red	627	-	nm
Dominant Wavelength $I_F = 10mA$	λ_{dom} [1]	High Efficiency Red	617	-	nm
Spectral Bandwidth at 50% Φ REL MAX $I_F = 10mA$	$\Delta\lambda$	High Efficiency Red	45	-	nm
Capacitance	C	High Efficiency Red	15	-	pF
Forward Voltage $I_F = 10mA$	V_F [2]	High Efficiency Red	1.9	2.3	V
Reverse Current ($V_R = 5V$)	I_R	High Efficiency Red	-	10	μA
Temperature Coefficient of λ_{peak} $I_F = 10mA$, $-10^\circ C \leq T \leq 85^\circ C$	$TC_{\lambda_{peak}}$	High Efficiency Red	0.13	-	nm/ $^\circ C$
Temperature Coefficient of λ_{dom} $I_F = 10mA$, $-10^\circ C \leq T \leq 85^\circ C$	$TC_{\lambda_{dom}}$	High Efficiency Red	0.06	-	nm/ $^\circ C$
Temperature Coefficient of V_F $I_F = 10mA$, $-10^\circ C \leq T \leq 85^\circ C$	TC_V	High Efficiency Red	-1.9	-	mV/ $^\circ C$

Notes:

1. The dominant wavelength (λ_d) above is the setup value of the sorting machine. (Tolerance $\lambda_d : \pm 1nm$.)
2. Forward voltage: $\pm 0.1V$.
3. Wavelength value is traceable to CIE127-2007 standards.
4. Excess driving current and / or operating temperature higher than recommended conditions may result in severe light degradation or premature failure.

ABSOLUTE MAXIMUM RATINGS at $T_A=25^\circ C$

Parameter	Symbol	Value	Unit
Power Dissipation	P_D	75	mW
Reverse Voltage	V_R	5	V
Junction Temperature	T_j	125	$^\circ C$
Operating Temperature	T_{op}	-40 to +85	$^\circ C$
Storage Temperature	T_{stg}	-40 to +85	$^\circ C$
DC Forward Current	I_F	30	mA
Peak Forward Current	I_{FM} [1]	160	mA
Electrostatic Discharge Threshold (HBM)	-	8000	V
Thermal Resistance (Junction / Ambient)	$R_{th JA}$ [2]	660	$^\circ C/W$
Thermal Resistance (Junction / Solder point)	$R_{th JS}$ [2]	420	$^\circ C/W$
Lead Solder Temperature [3]		260°C For 3 Seconds	
Lead Solder Temperature [4]		260°C For 5 Seconds	

- Notes:
1. 1/10 Duty Cycle, 0.1ms Pulse Width.
 2. $R_{th JA}$, $R_{th JS}$ Results from mounting on PC board FR4 (pad size $\geq 16 mm^2$ per pad).
 3. 2mm below package base.
 4. 5mm below package base.
 5. Relative humidity levels maintained between 40% and 60% in production area are recommended to avoid the build-up of static electricity – Ref JEDEC/JESD625-A and JEDEC/J-STD-033.