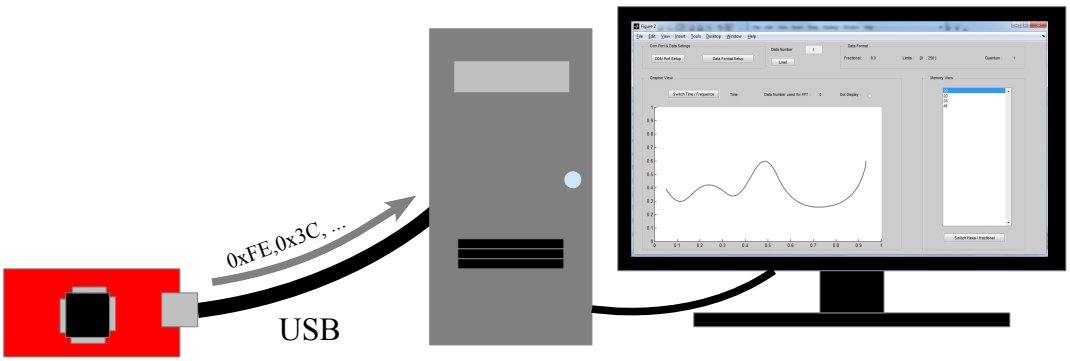
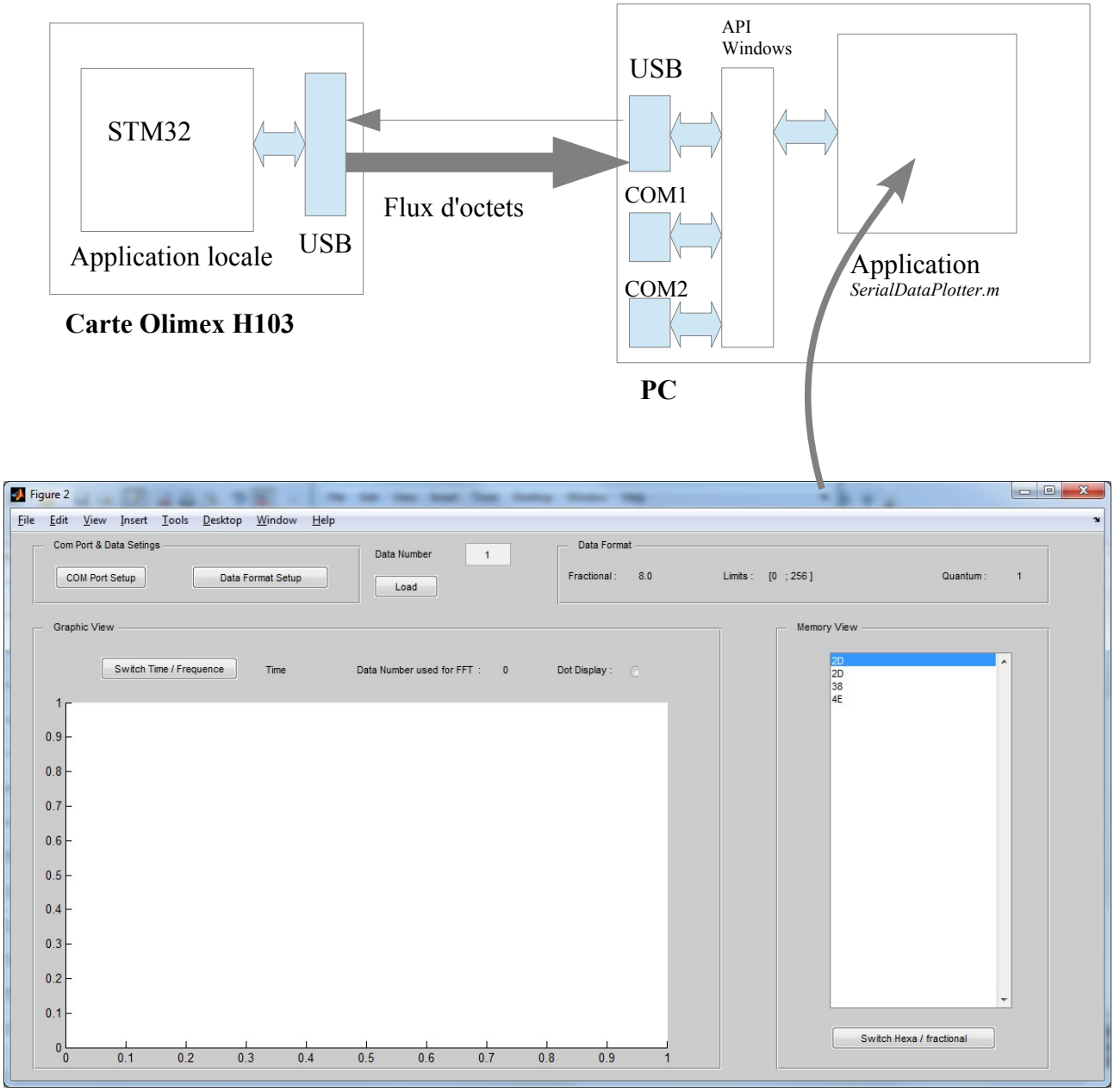
Guide d'utilisation de l'application
SerialDataPlotter.m (Matlab)
avec un STM32 (OlimexH103)



T.Rocacher,
S.Dimercurio

1. Aperçu

L'application *SerialDataPlotter.m,* développée sous *MATLAB,* permet de récupérer des données en provenance d'un port série (COM1 à COM20) et de les afficher sous forme de courbe (oscilloscope). Elle permet aussi d'afficher la FFT du bloc de données.



L'application permet de :

- Configurer le port COM. Il s'agit de choisir le port série à utiliser et de le configurer (Baud rate, bits de stop, Time out)

  NB : La liaison USB mise en œuvre est vue pour MATLAB comme un port série. Cependant, il est inutile de paramétrer la vitesse. Le transfert se fait au maximum de débit possible par le port USB de la carte Olimex.

- Télécharger des données
- Reformater les données en 8/16/32/64 bits, signé ou non, fractionnaire avec choix du format (sauf pour 64 bits).
- Afficher les données formatées sous forme de courbe (points ou lignes)
- Afficher la FFT normalisée des données formatées (points ou lignes) sur un nombre de valeur s'écrivant en 2^N. Si le nombre de données ne satisfait pas cette condition, les données en plus sont ignorées.

2. Détail de l'application *SerialDataPlotter.m*

- Bouton ***COM Port Setup*** : Donne accès au paramétrage de la liaison série.
- Bouton ***Data Format Setup*** : Permet de choisir l'interpétation qui doit être faite des données séries (de base 8 bits non signé). Le choix peut se faire avant ou après le chargement des données.

  NB : Le champ *Data Number* est le nombre de données à charger au **format spécifié**. Donc, si par exemple le format choisi est *u16*, et que le *Data Number* = 10 alors l'application va charger 20 octets.

- Bouton ***Switch Time/Frequence*** : affiche alternativement les données brutes ou la FFT. Le champ *Data Number used for FFT* indique le nombre de points effectivement pris pour calculer la FFT, de manière à ce qu'il soit le plus grand possible tout en étant une puissance de 2.
- Case à cocher ***Dot Display*** : affiche la prochaine courbe sous forme de points.
- ***Memory view*** : Contenu de la mémoire. Sous forme *Hexadécimale*, elle correspond au bloc d'octets chargé. Utile pour vérifier si le transfert de données s'est bien passé. Dans le format *fractionnaire*, la mémoire indique la suite de valeurs converties sous forme décimale. Ce sont ces valeurs qui sont ensuite affichées.

3. Programmation du microcontrôleur

L'utilisation de l'application *SerialDataPlotter.m* avec un STM32F103 (carte OlimexH103) via USB , nécessite dans KEIL :

- L'intégration de deux fichiers dans le projet : *libusb.lib* ainsi que *stm32f10x_it.o*
- L'instrumentation du code, c'est à dire l'ajout des instructions et autres déclarations permettant d'opérer le transfert USB. L'accès aux fonctions de l'USB se fait via l'API *libusb.h*.

Rem : si compilateur GCC, aller dans l'onglet *linker*,

- ajouter le *path* qui pointe dans *KEIL_USB\Gcc*,
- donner le nom de la lib dans *Include Librairies* : *usb* (en fait, à partir du nom de la librairie, *libusb.a*, il faut enlever le préfixe *lib*, et l'extension *.a*, ce qui donne *usb* dans ce cas)

3.1. Initialisation de l'USB

La fonction qui permet l'initialisation du périphérique USB est :

void Init_USB_Olimex_103(void).

Elle doit être lancée dans le programme principal. Elle est bloquante tant que le PC auquel il est relié n'a pas établi la connexion USB.

3.2. Emission des données série en USB

La fonction permettant l'envoi d'un tableau d'octets *uint8_t* est :

void Send_Byte_On_Received_Byte (*uint8_t* *Data_Tab, *uint32_t* Data_Nb);

La fonction nécessite donc l'adresse du tableau et le nombre de données à émettre. La fonction attend la réception d'un octet (quelconque) en provenance de l'application, puis envoie en série les octets. La fonction est donc bloquante.

NB : A la fin du bloc de donnée, le string « EOF » est émis.

3.3. Déclaration des variables

La fonction d'émission USB nécessite l'adresse d'un tableau contenant des variables de type *uint8_t*. (Il s'agit d'un type similaire à char).

On pourra donc déclarer un tableau ou un pointeur :

Pointeur :	Tableau :
*uint8_t * Ptr_Tab_USB;*	*uint8_t Tab_USB[64];*

L'émission se fera alors comme suit :

Send_Byte_On_Received_Byte (*Tab_USB*, 64); pour le tableau *uint8_t Tab_USB[64]*

Send_Byte_On_Received_Byte (*Ptr_Tab_USB*, 64); pour le pointeur *uint8_t * Ptr_Tab_USB*

Transmission d'un tableau de données de type différent de *uint8_t*

Il se peut que dans l'application en cours de développement, on veuille analyser une table de données dans un autre format que le type *uint8_t* . Par exemple, prenons une table *Ma_Table*, déclarée comme vs16 *Ma_Table[8]*.

Le plus simple est alors de passer par la déclaration d'un pointeur pour l'émission USB :

*uint8_t * Ptr_Tab_USB;*

Avant l'appel de la fonction d'émission, on opérera le casting suivant :

*Ptr_Tab_USB = (uint8_t *) Ma_Table;*

Cette ligne permet d'affecter au pointeur une adresse qui pointe initialement sur un autre type que *uint8_t* .

!!! Pour l'instant, la pile USB utilisée est limitée en nombre d'octets transmissibles.

La limite testée et validée est **1024 octets** émis.

Cette limitation conduit donc à :

1024 données *8bits*

512 données *16 bits*

256 données *32 nits*

128 données *64bits*

4. Problèmes connus, et résolution

Côté STM32, plantage de l'USB

La pile USB utilise une interruption régulièrement déclenchée de priorité niveau 8. La fonction *Send_Byte_On_Received_Byte* doit être exploitée dans le main (tâche de fond while(1)) et non dans une interruption, sinon l'USB plante.

Plantage apparent du PC qui détecte le port com mais qui refuse de l'ouvrir : le nombre d'octets à transmettre est trop important (voir limitations page précédente). C'est donc un problème côté STM.

Configuration du port COM

Le bouton *Update Available Port* est pour le moins capricieux. Lorsqu'on branche la carte *Olimex*, Windows doit détecter le nouveau périphérique USB. Il est vu comme un port série COMx. Le bouton en question devrait mettre à jour la liste des ports COM en ajoutant le nouveau.

Si le bouton est sans effet, il faut :

- Fermer la fenêtre de l'application
- Brancher la carte *Olimex*
- Relancer l'application *SerialDataPlotter.m*.. En principe la liste des ports est mis à jour, il faut alors choisir le dernier de la liste qui doit être le dernier détecté, donc celui de la carte Olimex.

NB: L'application ne peut détecter que les ports COM1 à COM20. Au delà, l'application ne prend plus de COM série en charge.

Si la procédure ne donne encore pas satisfaction :

- Quitter Matlab
- Vérifier par une autre application (X-CTU, hyperterminal, ...) que l'olimex est bien vue, et identifier son numéro de port COM
- Fermer toute application qui requiert ce port COM et débrancher l'USB
- Lancer Matlab à nouveau
- Rebrancher l'USB et lancer l'application *SerialDataPlotter.m*