USART

1 Configuration

Pour pouvoir relier une liaison série à un ordinateur hôte, il faut soit une liaison série disponible sur l'hôte (ce qui est de plus en plus rare) soit utiliser un port USB à l'aide un composant permettant de transformer les trames séries en paquets USB (convertisseur USB/série). L'adaptateur de débogage présent sur la carte Nucleo, est capable de jouer, en plus du débogage, le rôle d'un convertisseur USB/série. Cet adaptateur est relié sur la carte aux broches PA2 et PA3 du STM32. Ces deux broches peuvent être contrôlées par le périphérique USART2.
Une fois configuré, il sera possible de communiquer avec l'hôte via cette liaison série.

La liaison série via PA2 (TX) et PA3 (RX) nécessite d'une part de configurer le GPIO pour ces 2 pins en mode 'alternate' en choisissant la fonction alternative N°7 (AF07) et d'autre part de configurer le mode de transmission de l'USART. La transmission choisie est une transmission asynchrone, sans parité, un bit de stop et sans contrôle de flux. On souhaite travailler à 115200 bauds. La principale difficulté est de calculer en fonction de l'horloge de périphérique les facteurs de division à appliquer pour obtenir le baud rate correct. Afin que vous ne perdiez pas trop de temps dans la configuration, vous pouvez récupérer le fichier **usart.c** qui contient une fonction d'initialisation de l'USART. Il ne vous reste qu'à configurer le GPIO pour les pins PA2 et PA3.
Pour pouvoir utiliser la fonction de configuration de l'USART fournie, il est également nécessaire d'inclure dans votre projet le fichier **System_stm32f4xx.c**, issu de la librairie STM32, qui initialiser la FPU et défini la variable globale SystemCoreClock qui contient la valeur de la fréquence d'horloge du ceur.

2 Utilisation

2.1 Emission

2.1.1 Envoie d'une donnée 8bits

Dans cette partie vous allez écrire une fonction permettant d'envoyer un caractère vers l'hôte via l'USART2. Le prototype de la fonction est semblable à la fonction de la bibliothèque standard putchar() ;

```
int __io_putchar(int ch);
```

cette fonction prend en paramètre un caractère et renvoie le même caractère (sous format int).
La procédure pour envoyer un caractère est la suivante :

- vérifier que le drapeau USART_SR_TXE du registre s'état (SR) est à 1 (attendre le qu'il passe à 1 si celui-ci est à 0)
- copier la donnée dans le registre de donnée (DR)

Note : lorsqu'une donnée est copiée dans le registre DR, celle-ci est copiée dans le registre à décalage d'émission. La transmission s'effectue bit à bit à l'aide de ce registre à décalage. Il est possible de déterminer si tous les bits

ont été transmis en testant le bit TC du registre d'état.

Afin de vérifier que les données sont correctement transmises au terminal hôte, ouvrir un terminal (putty ou hyperterminal ou autre) sur la liaison série avec les paramètres 115200, 8, N, 1.
Sous linux le périphérique de la liaison série de l'adaptateur est /dev/ttyACM0. Sous windows déterminez le numéro de COM à l'aide du gestionnaire de périphérique. Il est noter que sous putty, il faut parfois utiliser la syntaxe \\.\COMxx sur les numéros élevés.

2.1.2 Envoi d'une trame

Ecrire une fonction permettant d'envoyer une chaîne de caractère via la liaison série :

```
void USART2_Transmit(uint8_t * data, uint32_t len)
```

où data est l'adresse de la chaîne et len le nombre de données à transmettre.

2.2 Réception

Le principe utilisé en réception est proche de celui utilisé en émission. En réception, le logiciel attend qu'une donnée soit reçue en scrutant le drapeau RXNE du registre d'état (registre de donnée réception Not Empty). Lorsque ce drapeau est à 1, une donnée non lue est présente dans DR et le logiciel peut la récupérer. Il est à noter que la lecture de registre de donnée remet le bit RXNE à 0.

Note : lorsque plusieurs données arrivent les unes à la suite des autres, il est nécessaire que le logiciel récupère la donnée avant qu'une autre arrive. Dans le cas contraire il y aura perte de donnée (écrasement de l'ancienne ou rejet de la nouvelle). Un drapeau du registre d'état (OVE : Overrun error) est activé dans ce cas. La durée de réception d'une donnée dépend du baud rate, par exemple à 115200,8,1,N, 10 symboles sont reçus, soit 10 cycles à 115200 Hz ce qui donne une durée d'emission/reception de 10/115200 = 86us à comparer avec la fréquence de travail du processeur. Par exemple à 16MHz, le processeur génère une instructions tous les 60ns, cela donne un rapport proche de 1000 (1000 instructions peuvent être éxécutées entre la réception de deux données).

2.2.1 Réception d'une donnée unique

Prototype de la fonction de réception d'une donnée :

```
int __io_getchar(void)
```

La fonction renverra le caractère reçu sur l'octet de poids faible de l'entier. Dans cette fonction vous ne tiendrez pas compte des erreurs possible en réception (OVE, FE, NE).
Vous pourrez tester cette fonction à l'aide du terminal.

2.2.2 Réception d'une trame

Le prototype de la fonction est cette fois

```
int32_t USART2_Receive(uint8_t * data, uint32_t len)
```

avec

- data : adresse du buffer de l'application où seront stockées des données
- len : le nombre de données à transférer

la fonction renvoie le nombre de données effectivement reçues.

Vous mettrez en place un time-out après la réception de la première donnée afin de détecter les fins de trame. Ce time-out devra avoir une durée supérieure à la durée de réception d'un caractère.

Pour envoyer une trame continue depuis votre station de travail, voir l'annexe 3.2.

3 Annexes

3.1 Redirection de la libC

Il est possible de rediriger les entrées/sorties standard de la libc (stdio) vers la liaison série. Les fonctions 'redirigeable' ou à implémenter se trouve dans les fichiers syscall.c et symem.c autogénérés par l'IDE.
Les principale pour rediriger les entrées/sortie stanrdard sont :

- _sbrk : cette fonction est appelée par la stlib s'il ne reste pas de mémoire disponible pour la fonction malloc(). Elle permet d'allouer un nouveau bloc mémoire qui passera dans le pool de malloc. Au premier appel de malloc cette fonction est automatiquement appelée. La fonction renvoie l'adresse du nouveau bloc mémoire.
- _write : fonction appelée par toutes les fonctions utilisant la sortie standard (e.g. printf, puts, ...)
- _read : fonction appelée par les fonctions qui utilise l'entrée standard (e.g. scanf, getchar, …)

Note :

- Les fonctions printf et scanf sont des fonctions bufferisée qui allouent par défaut sur newlib 1ko chacune.
- Les fonctions standard de la librarie C augmente l'emprunte mémoire (par exemple printf utilise de l'ordre de 5ko) pour les sytèmes qui ont peu de mémoire, il ne vaut mieux pas utiliser ces fonctions.

La fonction _sbrk fournie par STMicroelectornics devrait fonctionner pour une simple application sans exécutif multitâche.

3.2 Envoie de donnée depuis la station hôte

Pour tester la réception de données par trames continue, l'utilisation du terminal ne fonctionne pas. Il est nécessaire, soit d'écrire un programme, soit d'utiliser la console.

- Programme : déterminer comment configurer le pilote série (baud rate, parité, …) enduite utiliser les fonctions traditionnelle de pilote de périphériques (open, write, close). Un exemple de programme pour linux est donné et dispo sous moodle.
- Console : configurer la paramètre de la liaison série puis rediriger une fichier vers le périphérique série relié à la carte nucleo. Exemple :

  Linux :
  stty -F /dev/ttyACM0 115200 (-f sur macOS/bsd)
  echo -ne "Several characters string without a line feed" > /dev/ttyACM0
  (/dev/tty.usbmodemxx sur mac)
  Pour visualiser ce qui est reçus sur la liaison série de l'hôte :
  cat /dev/ttyACM0
  Window, doit être quelque chose comme :
  mode COMxx BAUD=115200 PARITY=n DATA=8
  avec xx l numéro de port virtuel COM (obtenu quand la carte est branchée)
  echo hello > \\.\COMxx