TP 2 FOX G20
Application utilisant le port série

Objectif :
Développer une application d'acquisition de donnée utilisant le port série.
Lancer des applications automatiquement au démarrage de la carte.

Ports séries sous Linux

Le port série est souvent le seul moyen de communication disponible sur un système embarqué (surtout en phase de configuration). Il important de maitriser son utilisation car il est très utiliser pour communiquer facilement avec différents systèmes.

Références :

- http://tldp.org/HOWTO/Serial-Programming-HOWTO/x56.html#AEN88
- http://tldp.org/HOWTO/Serial-HOWTO.html
- http://www.nullmodem.com
- http://www.easysw.com/~mike/serial/serial.html
- http://www.acmesystems.it/?id=foxg20_serial_rs232

Considérations matérielles :
DTE signifie Data Terminal Equipment et correspond en général au PC, alors que DCE signifie Data Communication Equipement et correspond au périphérique.
Pour relier deux éléments de même nature il faut toujours un câble croisé.
Pour relier un DCE à un DTE un câble droit suffit.
Si le contrôle de flux matériel n'est pas utilisé, il suffit de câbler Txd,Rxd et la masse.
Sur un système embarqué on peut avoir soit un port à la norme RS232 soit un port fonctionnant directement avec les tensions du microprocesseur (3.3 ou 5V). Dans ce dernier cas un adaptateur vers RS232 est nécessaire (circuit type MAX232).
Si votre pc ne comporte pas de port série un adaptateur USB/RS232 est nécessaire.

Sur la carte FOX on dispose de 6 ports série maximum. Un port série nommé DPI "Debug Port Interface" est utilisé comme console par défaut. Un convertisseur vers USB est disponible sur la carte de TP.

Identifiez vos ports série

Les ports série apparaissent comme des fichiers spéciaux dans /dev.
Généralement /dev/ttyS0, /dev/ttyS1,... et /dev/ttyUSB0,/dev/ttyUSB1,... si vous utilisez un adaptateur USB.

Listez les ports série de votre machine et de la fox (ls -l) et déterminez ceux qui sont réellement présents (par exemple avec stty). Afficher leurs configurations.

TP 2 FOX G20
Application utilisant le port série

Lorsque l'on a plusieurs adaptateurs série, il est commode d'attribuer un nom unique, constant et évocateur à chaque adaptateur grâce à l'interface udev.
Pour cela nous allons créer une règle udev dans /etc/udev/rules.d/ (voir "créer une règle udev").
Par la commande **`lssub`** relevez le "product ID" et le "vendor ID" de votre adaptateur.
Comme on a souvent des circuits identiques (circuit FTDI par exemple), il faut en plus avoir le numéro de série pour l'identifier de manière unique. On peut relever le numéro de série par la commande :

```
udevadm info -a -n /dev/ttyUSBx | grep '{serial}' | head -n1
```

Ensuite créer un fichier nommé par exemple `99-usb-serial.rules` dans `/etc/udev/rules.d/` contenant pour chaque adaptateur une ligne de la forme :

```
SUBSYSTEM=="tty", ATTRS{idVendor}=="0403",
ATTRS{idProduct}=="6001", ATTRS{serial}=="FTC8HZZH",
SYMLINK+="fox01"
```

L'attribut SYMLINK est le lien symbolique souhaité. Mettre fox<numéro binôme>

Débranchez et rebranchez votre adaptateur et vérifiez qu'il est bien accessible par /dev/ttyUSBx et par le nom constant /dev/fox01 (ls -l /dev/fox01).

Vous pouvez rajouter d'autres lignes dans ce même fichier si vous possédez d'autres adaptateurs ou périphériques usb/série.

Sur la foxg20 le port série /dev/ttyS0 utilisé pour le DPI (Debug Port Interface) est associé au programme getty. Il suffit d'enlever la ligne correspondante dans inittab pour pouvoir utiliser ce port comme un port série classique.
Expliquez ce qu'est getty et à quoi sert inittab.
Cependant les messages du noyau restent actif sur /dev/tyyS0 car c'est ce port qui a été définit comme console par la ligne de commande du noyau. Vérifier en connectant une clé usb par exemple, les messages sur /dev/ttyS0. Si on veut développer une application utilisant ce port, ces messages vont interférer avec les messages de l'application. Ces messages sont émis par la fonction printk (équivalent de printf pou le noyau). Son comportement est paramétrable via le pseudo système de fichier /proc par /proc/sys/kernel/printk.
Voir google " /proc/sys/kernel/printk" par exemple.
Vérifier que :

```
echo "0 4 1 7" >/proc/sys/kernel/printk
```

supprime les messages

```
echo "7 4 1 7" >/proc/sys/kernel/printk
```

rétablie les messages

Vérifier que les messages restent accessible dans /var/log/messages

TP 2 FOX G20
Application utilisant le port série

Utilisation en ligne de commande.

Configurez votre port /dev/fox01 pour communiquer avec la foxg20 avec la commande stty.
Loggez vous sur la fox en ssh et configurez le port série pour communiquer avec le pc.
Communiquez entre la fox et le pc grâce aux commandes echo et cat.
Sauvez dans un fichier les données reçues, envoyer le contenu d'un fichier texte.
Installez le programme gtkterm (ou minicom ou cutecom) et utilisez le pour communiquer.
Avec ce programme vérifiez que vous savez sauvegarder une configuration, envoyer un fichier texte, sauvegarder dans un fichier les données reçues.

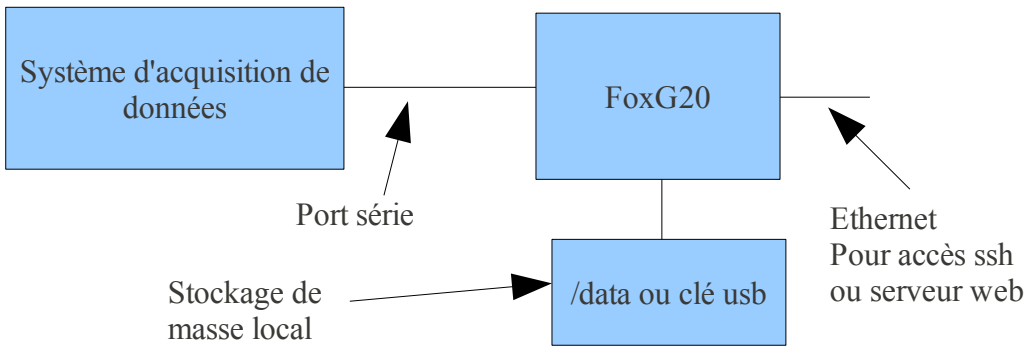
Utilisation en C.

Lire le **serial programming HOWTO** et expliquer les différences entre les modes canonique et non-canonique ainsi qu'entre le fonctionnement synchrone et asynchrone.
En vous inspirant de l'exemple écrivez un programme (en mode canonique synchrone) qui transmet une chaîne de caractères toutes les cinq secondes et un programme qui lit et affiche les chaînes reçues. Vos programmes prendront le périphérique et la vitesse de transmission en argument sur la ligne de commande : readtty -d /dev/ttyUSB0 -b 2400 (sans option on utilise /dev/ttyS0 à 9600bd).
(utilisez getopt() voir le manuel de la libc à http://www.gnu.org/software/libc/manual/)

Compiler une version pour pc et une version pour la fox de chaque programme.

Développement d'une application utilisant le port série.

On va maintenant développer une application très simple de stockage et d'horodatage de données reçues sur un port série.
Le problème se pose souvent en ces termes : un système d'acquisition de données est accessible par des commandes sur un port série. On doit archiver et dater les données reçues et les rendre accessibles sur un réseau local (par stfp ou par un serveur web par exemple).



TP 2 FOX G20
Application utilisant le port série

Simulateur de système d'acquisition

Le système d'acquisition de données est simulé par un programme tournant sur le pc.
Ce programme écoute le port série et répond à des commandes définies dans un fichier texte cmd.conf. Les paramètres du port sont configurables par des options sur le ligne de commande. Pour simuler des pannes du systèmes d'acquisition, une probabilité de panne est aussi paramétrable par une option sur la ligne de commande.
Exemple de fichier cmd.conf :

```
# commentaire
#cmd   format min max unité
!T1     %3d   -20     100     C
!T2     %3d   0       200     C
!N1     %3x   0       1023    bits
!V1     %2d   -12     +12     Volts
```

Options sur la ligne de commande :
-d /dev/ttyS0 : port série
-s 9600 : vitesse
-c cmd.conf : fichier de configuration
-f 20 : probabilité de panne (0-100)
-d 4 : délais en s entre la réception d'une commande et son exécution
Prévoir des options par défauts cohérentes.

Lorsque le système est en panne il renvoie une chaine incohérente ou bien ne répond plus aux commandes pendant une durée aléatoire (2 ou 10 fois le délais normal par exemple)

Exemple de dialogue > foxg20 <simulateur

```
< #sim v1.0
<#rdy
>!T1
<#T1 30 xx  // xx check sum
>!N1
<#N1 2F1 xx
>!V1
                    //panne délais
>!V1
#Nz34'rrr  // panne données incohérentes
>!V1
!V1 10 xx  // retour fonctionnement normal.
```

Décrire l'organisation générale de votre programme, codez le et proposez une série de test de recette. Le langage n'est pas imposé (c, python, c++/Qt?...) mais doit être libre.

TP 2 FOX G20
Application utilisant le port série

Acquisition horodatage et archivage sur la carte foxg20

Écrire un programme qui dialogue avec le système d'acquisition simulé pour acquérir l'ensemble des données disponibles à intervalles réguliers, les stocker dans un fichier en les horodatant. Le langage est libre, de préférence différent du simulateur.
Votre programme pourra utiliser le même fichier de configuration que le simulateur et sera paramétrable par le ligne de commande.
Le fichier d'archive aura le format suivant :

```
#date          time          T1      T2      N1      V1
2011-01-23     10:03:05       23     123     3F1     11
2011-01-23     10:04:06       24     123     3F0     12
2011-01-23     10:05:05      ---     ----    ----    ---   //  --- absence de données
2011-01-23     10:06:05      !!      124     !!      09 //    !!  données incohérentes
```

Le nom du fichier d'archive devrait inclure la date de début des mesures. On peut changer de fichier d'archive toutes les heures ou toutes les N mesures. Veillez à ce qu'en aucun cas votre programme ne puissent détruire des mesures précédemment archivées.

Créer un script pour lancer votre programme dès la mise sous tension de la carte.
(voir /etc/init.d update update-rc.d)