Raspberry Pi 2 Liaison Série

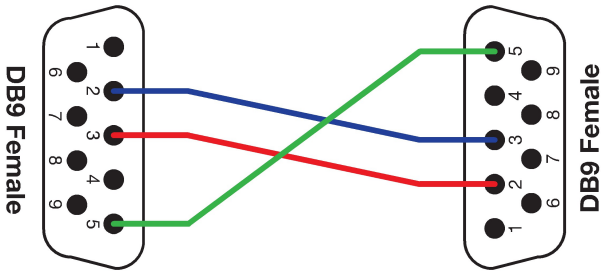
1 Présentation

UART (**Universal Asynchronous Receiver Transmitter**) assurent l'émission et la réception asynchrone.
Sur le Raspberry Pi, les niveaux sont
0v, 3.3v
TXD : Port GPIO14 (broche 8)
RXD : Port GPIO15 (broche 10)

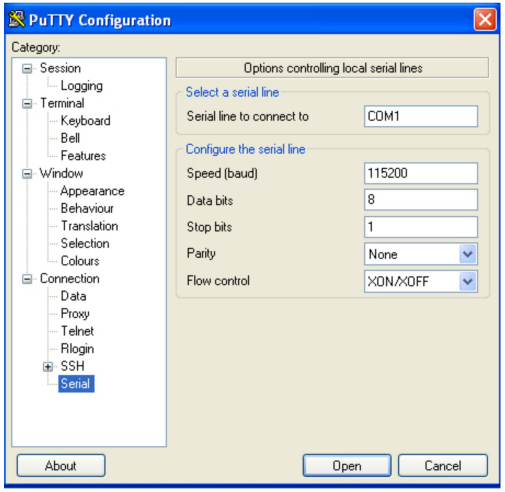
Une carte hat qui comprend les circuits nécessaires à la conversion de niveaux, doit être ajoutée pour obtenir une liaison série standard RS232.

https://www.abelectronics.co.uk/p/51/Serial-Pi-Plus

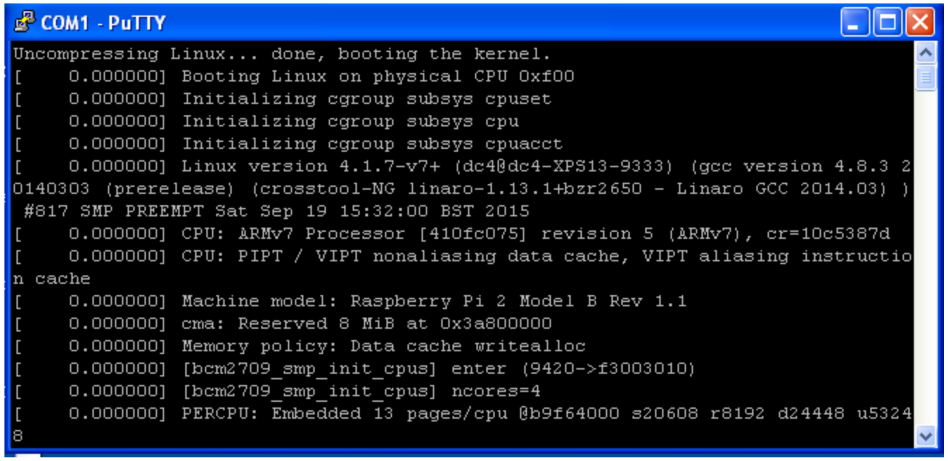
Connecter la carte sur le port com du PC avec un câble DB9 femelle/femelle. La broche 2 correspond à RX (data receive). La broche 3 à TX (data transmit). La broche 5 est la masse.



ouvrir PuTTY et configuer la liaison de la façon suivante:



Mettre sous tension le Rapberry Pi



Sous Rasperry PI2, l'UART est connu sous le nom de ttyAMA0. On le trouve donc ici : **/dev/ttyAMA0**.
Par défaut l'UART du Raspberry Pi sert de port de débug pour Linux.

2 Libérer l'UART du mode debug

Empêcher l'émission de messages du Kernel et l'activation du mode debugging sur l'UART

```
pi@raspberrypi /dev $ sudo nano /boot/cmdline.txt

dwc_otg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait
```

devient

```
dwc_otg.lpm_enable=0 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait
```

Pour désactiver le terminal sur l'UART, il faut éditer le fichier /etc/inittab et commenter la ligne faisant référence à l'UART :

```
#Spawn a getty on Raspberry Pi serial line
#T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

Après le redémarrage, on peut vérifier qu'aucun processus n'utilise UART /dev/ttyAMA0.

```
pi@raspberrypi ~ $ ps aux | grep ttyAMA0
```

3 Configuration

Stty permet de connaître la configuration actuelle de la liaison

```
pi@myraspberry ~/C $ stty -F /dev/ttyAMA0
speed 9600 baud; line = 0;
-brkint -imaxbel
```


sur l'écran ci-dessus on peut voir que la vitesse de transmission est 9600 bauds

4 Utilisation en ligne de commande

Sous Linux, "tout est fichier". L'envoi ou la réception de données sur le RPi se fait en lisant ou en écrivant dans /dev/ttyAMA0.

Par exemple pour envoyer le contenu du fichier achat.c sur la sortie série, la commande pourrait être la suivante:

```
pi@myraspberry ~/C $ cat achat.c > /dev/ttyAMA0
```

Autre exemple : avec la commande echo

```
pi@myraspberry ~/C $ echo 'bonjour le monde' > /dev/ttyAMA0
```

5 Programmation en C

En C, nous utiliserons par exemple **open read write close**
pour en savoir plus : man 2 open man 2 read etc etc

```
#include <stdio.h>   /* Standard input/output definitions */
#include <string.h>  /* String function definitions */
#include <fcntl.h>   /* File control definitions */
#include <errno.h>

int main(int argc, char **argv) {
    int retour,fd;
    char recu[256];
    const char *message1="Bienvenue sur Raspberry Pi\n";
    const char message2[]="Message envoyé : ";

    memset(recu, 0, 256);      // efface le buffer
    fd = open("/dev/ttyAMA0", O_RDWR | O_NOCTTY);
```

```
    write(fd, message1, 27);  // envoi du message 1
    read(fd, recu, 255);      // reception d'un message
    write(fd, message2, 18);  // envoi du message 2
    write(fd, recu, 255);     // envoi du message reçu

    printf("Message recu:\n%s\n", recu);
    retour=close(fd);
    if ( retour == -1 ){
        printf("pb fermeture: %s\n",strerror(errno));
        return(errno);
    }
    return 0;
}
```


Fiche 16 Raspberry pi 2 Liaison série