

# RPI : Uart Liaison serie



	3.3V	1	2	5V
	GPIO2 (SDA1)	3	4	5V
	GPIO3 (SCL1)	5	6	GND
	GPIO4 (GPIO_GCLK)	7	8	GPIO14 (UART_TXD0)
	GND	9	10	GPIO15 (UART_RXD0)
	GPIO17 (GPIO_GEN0)	11	12	GPIO18 (GPIO_GEN1)
	GPIO27 (GPIO_GEN2)	13	14	GND
	GPIO22 (GPIO_GEN3)	15	16	GPIO23 (GPIO_GEN4)
	3.3V	17	18	GPIO24 (GPIO_GEN5)
	GPIO10 (SPI0_MOSI)	19	20	GND
	GPIO9 (SPI0_MISO)	21	22	GPIO25 (GPIO_GEN6)
	GPIO11 (SPI0_CLK)	23	24	GPIO8 (SPI_CE0_N)
	GND	25	26	GPIO7 (SPI_CE1_N)
	ID_SD (I2C EEPROM)	27	28	ID_SC (I2C EEPROM)
	GPIO5	29	30	GND
	GPIO6	31	32	GPIO12
	GPIO13	33	34	GND
	GPIO19	35	36	GPIO16
	GPIO26	37	38	GPIO20
	GND	39	40	GPIO21

[Liaison-UART-du-Raspberry-Pi](#)

## Liaison UART du Raspberry Pi 29 janvier 2014

Avoir un PC embarqué c'est bien, pouvoir communiquer facilement avec les microcontrôleurs de l'électronique, c'est mieux !

Pour cela, le Raspberry Pi propose plusieurs protocoles de communication :

1. I2C
2. SPI
3. UART

Nous pourrions reprocher à l'I2C et au SPI de ne pas permettre de communications asynchrones. C'est en partie pour cela que nous nous sommes intéressés à l'UART.

Si ces trois protocoles sont des protocoles de communication de type série, l'UART est souvent considéré comme LE protocole série. C'est l'un des protocoles de type série les plus simples.

Sur le Raspberry Pi sous Raspbian, l'UART est connu sous le nom de ttyAMA0. On le trouve donc ici : /dev/ttyAMA0. Cependant, pour se servir de la liaison UART d'un Raspberry Pi sous Raspbian, il y a quelques manipulations préalables à réaliser. Confiscation par le noyau

Tout d'abord, la liaison est accaparée par le noyau pour offrir un terminal par la liaison série (UART). Il faut dire au noyau de :

1. ne pas se servir de la liaison série au démarrage ;
2. ne pas créer un terminal sur cette liaison.

Pour supprimer l'utilisation de l'UART du démarrage, il faut éditer le fichier /boot/cmdline.txt, supprimer la partie en rouge :

/boot/cmdline.txt

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

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

L'édition du fichier /boot/cmdline.txt demande les droits d'administrateur si vous travaillez directement sur le Raspberry Pi.

Pour désactiver le terminal sur l'UART, il faut éditer le fichier /etc/inittab et supprimer ou commenter (en ajoutant un # en début de ligne) les lignes faisant référence à notre UART :

```
/etc/inittab
#T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

Après un redémarrage, vous devriez pouvoir accéder à l'UART. Pour vérifier que tout s'est bien passé, vous pouvez vérifier qu'aucun processus n'utilise votre précieux /dev/ttyAMA0.

Terminal

```
ps aux | grep ttyAMA0
```

Et que le noyau vous a bien rendu la main :

Terminal

```
cat /proc/cmdline
```

## Configuration de la liaison UART

Mais ce n'est pas fini ! En effet, la liaison `/dev/ttyAMA0` est configurée pour vous renvoyer un écho de ce qu'elle reçoit. La liaison se configure avec `stty`. Pour connaître votre configuration actuelle utilisez :

Terminal

```
stty -F /dev/ttyAMA0 -a
```

Pour comprendre précisément la configuration, la page man de `stty` vous sera bien utile. Nous utilisons la commande suivante pour avoir une configuration fonctionnelle.

Terminal

```
sudo stty -F /dev/ttyAMA0 115200 cs8 -cstopb -onlcr -echo -echoe -echok -opost
```

Vous pouvez changer 115200 par la vitesse souhaitée de votre liaison série. Bug du Raspberry Pi

Enfin, vous êtes confronté à un bug du Raspberry Pi, son rapport est ici. Pour éviter d'avoir des octets aléatoires en début de transmission, il est recommandé de laisser le port ouvert :

Terminal

```
exec 9> /dev/ttyAMA0
```

Et voilà ! Vous êtes prêt à utiliser l'UART. Si vous voulez modifier la vitesse de la liaison série, n'hésitez pas à utiliser `stty` ! Utilisation

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`. En C, nous utiliserons par exemple `fopen`, `fprintf` et `fgetc`, tandis qu'en bash, nous utiliserons les commandes suivantes. Pour la réception des données du port série :

Terminal

```
cat /dev/ttyAMA0
```

Pour l'émission de données sur le port série :

Terminal `echo -e -n "Bonjour\x00" > /dev/ttyAMA0`

Pour la commande `echo`, nous utilisons les options suivantes :

1. e Demande à la fonction echo d'interpréter les caractères spéciaux, ce qui transformera notre '\x00' en caractère de fin de chaîne.
2. n Demande à la fonction echo de ne pas générer automatiquement de retour à la ligne.

## Liens web

[utiliser-luart-port-serie-du-raspberry-pi-](#)

[RPI Serial Connection - eLinux.org \(HTML - 37.2 kio\)](#) Guide pour récupérer la main sur la liaison série (en anglais)

[Uart RPI FR](#)

[UART RPI EN](#)

[Comment utiliser l'UART sur RPI EN](#)

From:

<https://chanterie37.fr/fablab37110/> - **Castel'Lab le Fablab MJC de Château-Renault**

Permanent link:

<https://chanterie37.fr/fablab37110/doku.php?id=start:raspberry:uart&rev=1740291029>

Last update: **2025/02/23 07:10**

