# Stratégies d’abaissement de la consommation d’une carte Raspberry

A l’état initial, c’est-à-dire un Raspberry avec Linux fraîchement installé et sans avoir effectué de réglage dessus, la consommation théorique est de 3,5 watt, en partant du fait que le courant conseillé est de 0,7 ampères et un voltage de 5 volts.

Afin de faires baisser la consommation, plusieurs stratégies seront appliquées:

## Alléger le système d’exploitation

* En quoi cela consiste?

Les systèmes d’exploitation des ordinateurs doivent répondre aux besoin du plus grand nombre. C’est pour cela qu’on retrouve nombre de programmes comme des jeux, un bloc note, un navigateurs, divers utilitaires, etc… Ces programmes prennent de la place sur la carte mémoire, et certains d’entre eux sont susceptibles de se lancer automatiquement, ce qui va consommer beaucoup de ressources (processeur et mémoire vive)

Or, nous n’avons pas besoin d’interface graphique et ce qui nous intéresse c’est de pourvoir lancer un programme.

Nous n’avons donc besoin que du noyaux Linux, ses utilitaires usuels (pour utiliser les commandes notamment), et de quoi lancer le programme.

* Comment ça se passe?

Plutôt que de partir sur un système d’exploitation classique tel que Raspbian qui serait ensuite dépouillé, nous allons installer “Arch Linux”. C’est une distribution dont les efforts sont portés sur la légèreté, la simplicité, et fonctionne sans outil graphique. En résumé, le principe est le suivant: on installe seulement le minimum vital (le coeur du système), et on ajoute juste ce dont nous avons besoin.

Procéder ainsi nous assure non seulement un certain gain de temps comparé au dépouillage de Raspbian, et est beaucoup moins risqué du point de vue de la stabilité du système.

Nous avons une carte ne faisant fonctionner que l’essentiel, évitant les charges inutiles et les montées en température du CPU.

* Quel sont les gains?

Ils sont difficiles à mesurer dans la mesure où une installation par défaut d’un système de type Raspbian ne consomme pas plus qu’un Arch Linux. L’objectif ici est de s’assurer que le maximum de ressources matérielles sont disponibles pour faire fonctionner les programmes dont nous aurons réellement besoin.

## 2) L’underclocking

* En quoi cela consiste?

Par défaut, un système d’exploitation utilise toutes les ressources (CPU et GPU notamment). C’est parfaitement normal. L’underclocking consiste à faire baisser la fréquence de fonctionnement (la fréquence d’horloge) pour la faire marcher plus lentement. Autrement dit, on va lui demander d’utiliser moins de ressources que disponible

* Comment ça se passe?

Ce changement se fait en éditant le fichier **config.txt**, dans ce fichier il est possible de configurer la fréquence du processeur, et mémoire vidéo allouée. La mémoire vidéo est fixée à 16Mb (on dessous, le raspberry peut avoir des difficultés à fonctionner correctement). Dans un souci d’avoir une bonne performance de calcul tout en consommant le moins possible, nous avons baissé la fréquence du processeur à 200 MHz (au lieu de 700 MHz). Nous avons choisis 200 MHz à la suite de plusieurs tests où nous avons mesuré le temps de calcul et la consommation.

* Quel sont les gains?

Le CPU dégagera moins de chaleur et consommera quelques mA en moins (entre 5 et 7) car il sera moins sollicité.

## 

## 

## 3) L’utilisation d’une liaison série

* En quoi cela consiste?

Bien que moins présent aujourd’hui dans les produits grands publics, beaucoup d’appareils disposent d’une liaison dite “série”. Cette liaison, passant par l’interface du même nom, permet d’établir une liaison directe entre le dit appareil et un ordinateur.

Ainsi, pas besoin de passer par le réseau et d’utiliser l’utilitaire ssh comme on le fait d’habitude pour obtenir le même résultat.

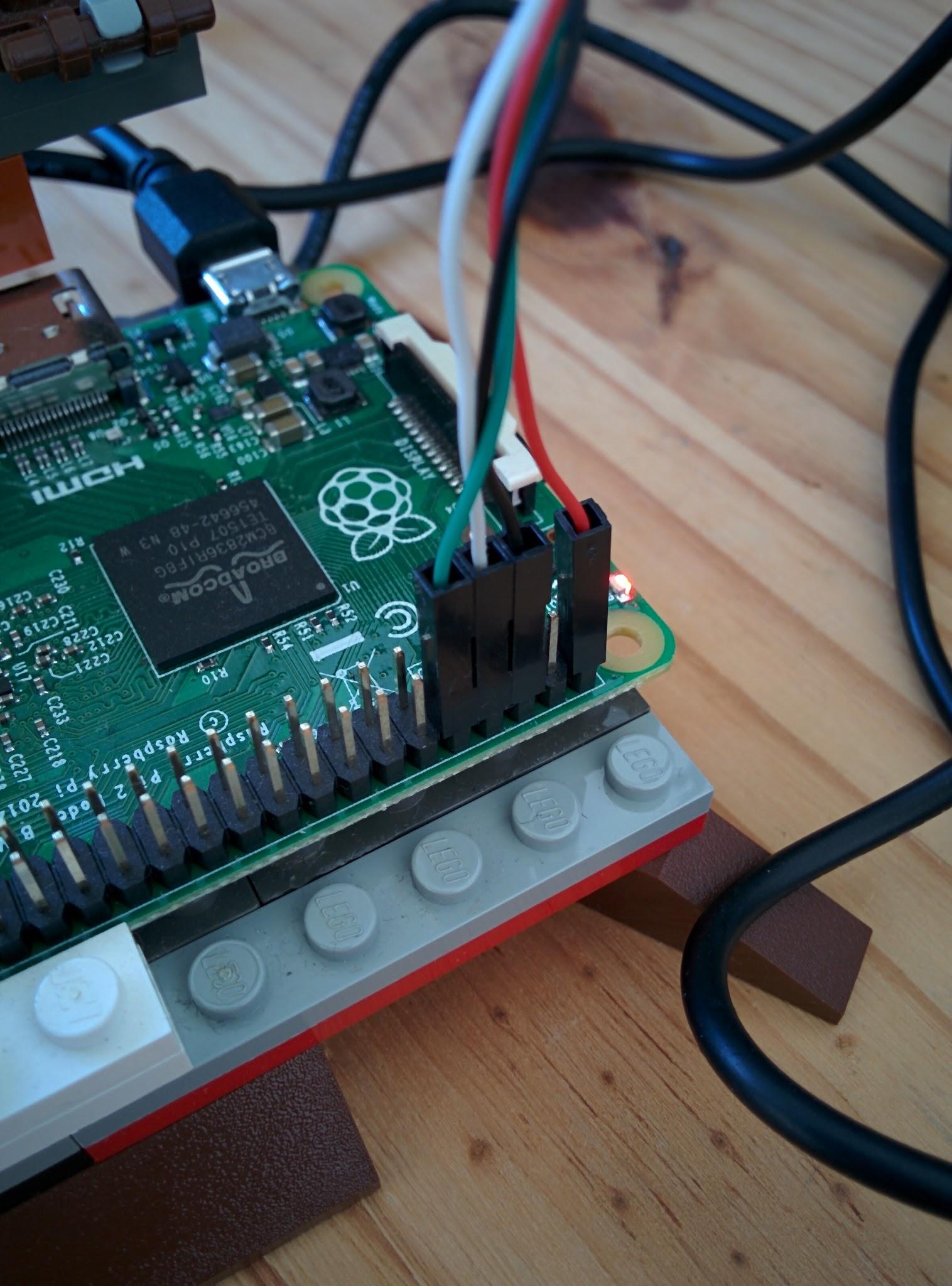
* Comment ça se passe?

Pour faire notre fameuse liaison série, nous allons utiliser un câble dit “RS232”.



Câble RS232 utilisé

D’un côté nous avons une prise usb à brancher sur un ordinateur, et de l’autre des fils à brancher sur les pins 8 et 10 du Rapsberry (dédiés respectivement au transfert de données TX et à la réception de données RX):

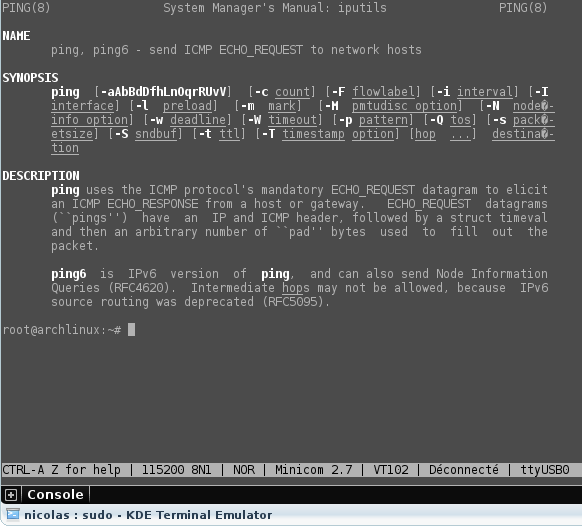


Maintenant côté ordinateur, il faut installer “minicom”, qui va assurer la communication entre Linux (utilisé ici) et le raspberry.

Ensuite, nous allons brancher le câble RS232 en usb, puis l’alimentation du raspberry, puis lancer la commande suivante sur l’ordinateur:

**sudo minicom -b 115200 -D /dev/ttyUSB0**

Nous obtenons ainsi un accès direct au terminal du commande du raspberry:



Exemple du résultat d’une commande “man”

* Quels les gains?

Nous avons un accès intégral au système sans passer par le réseau. Nous avons donc la possibilité de désactiver des composants matériels inutiles comme ici le port RJ45, très consommateur. Rien qu’avec cela, nous pouvons économiser environ 17 mA. Cela se fait en saisissant la commande **ip link set eth0 down** dans minicom.

## Combien toute cette installation coûte?

Il faut compter 45€ le raspberry pi 2 (modèle B) chez LDLC (prix de livraison inconnu):

<http://www.ldlc.com/fiche/PB00182827.html>

5,49€ pour une carte mémoire SD de 4Go (prix de livraison inconnu):

<http://www.amazon.fr/Transcend-TS4GSDHC10E-Emballage-D%C3%A9baller-s%C3%A9nerver/dp/B003VNKNE6/ref=cm_cr_pr_product_top?ie=UTF8>

3,90€ + 1,39€ de livraison pour un câble RS232:

<http://www.ebay.fr/itm/USB-TO-SERIAL-ADAPTER-MODULE-TTL-RS232-ARDUINO-CABLES-4-PINS-FTDI-FT232RL-/262230922613>?

Cela fait un total de 55,73€.