

[openelec](#), [os](#), [linux](#), [kodi](#), [xbmc](#), [media-center](#), [archive](#)



Raison de l'abandon

Suite à des divergences d'opinion entre le fondateur et la majorité des développeurs impliqués dans le projet, ceux-ci sont partis pour créer le [fork LibreELEC](#). OpenELEC se retrouve du coup actuellement dans un stade plutôt végétatif avec surtout deux versions de Kodi de retard et il aurait été dommage de se contenter de la version 15 Isengard au lieu de la 17 Krypton au moment de passer à un Raspberry Pi 3.

Késako?

La signification littérale est “**Open Embedded Linux Entertainment Center**”.

Mais encore?

C'est un système d'exploitation construit à partir du noyau Linux pour supporter au mieux le logiciel multimédia ~~XBMC~~ [Kodi](#).

Qu'est-ce qui le différencie de ses concurrents?

OpenELEC n'est pas un OS pré-existant adapté et optimisé dans un but précis, il a été développé à partir de rien dans le seul but de supporter un logiciel.

Il en résulte qu'**OpenELEC est réellement sensiblement plus stable que ses concurrents** mais par contre il ne fait rien d'autre: impossible d'installer le moindre paquet non prévu par l'équipe de développement ni même d'installer un paquet existant dans une autre version que celle dans laquelle il est fourni.

Premier contact

Petit test avec la version stable en cours (4.2.1): c'est réactif, l'installation est simple et permet par exemple dès le départ de paramétrer manuellement le réseau, pas besoin de SSH pour passer à un clavier azerty et la stabilité semble excellente: histoire de mettre le système à rude épreuve et de voir un peu ceux que je ne connaissais pas, j'ai installé une bonne douzaine de thèmes plus ou moins lourds (Xbian n'avait pas supporté plus de 3 ou 4) et il n'y a pas eu le moindre plantage (par contre, même constat qu'avec Xbian ou RaspBMC: il faut éviter les thèmes trop lourds, pas de surprise de ce côté là).

OpenELEC est souvent présenté comme bien plus simple et limité que ses concurrents et c'est pour cela que je l'avais jusque là laissé de côté, mais, contrairement à ce que je pensais, on retrouve toutes les fonctions de ~~XBMC~~ Kodi, seul l'OS est limité (mais tous les services annexes sont là: SSH, Samba, etc.).



Informations de connexion SSH par défaut:

- login: root



• mot de passe: openelec

Un risque beta

La version 5.0 beta d'OpenELEC (curieusement numérotée 4.95.1 0-8) qui inclut le nouveau ~~XBMC~~ Kodi¹⁾ encore en *beta* est disponible, et, d'après ce que l'on peut lire sur le forum OpenELEC, le tout est plutôt fonctionnel (sauf avec quelques outils du genre PVR et quelques cartes graphiques ou audio Intel) donc je tente...

L'installation

J'aurais voulu pouvoir directement faire une installation sur un clef USB mais la procédure trouvée n'est pas valable et j'ai donc laissé cela de côté [pour plus tard](#).

La procédure d'installation d'OpenELEC est par ailleurs standard dans le monde Raspberry, rapide et très simple: décompression d'une image disque sur la carte SD et 5 minutes plus tard, tout est en route! <3

Fonctionnement de base



Kodi étant encore en version beta, il n'y a que 6 thèmes utilisables à ce jour (11/11/2014)

Aucun problème à signaler



Les développeurs de Kodi comme ceux d'OpenELEC ont fait un sacré boulot et ce duo de versions beta fonctionne très bien sur Raspberry Pi (modèle B avec 512Mo de RAM).

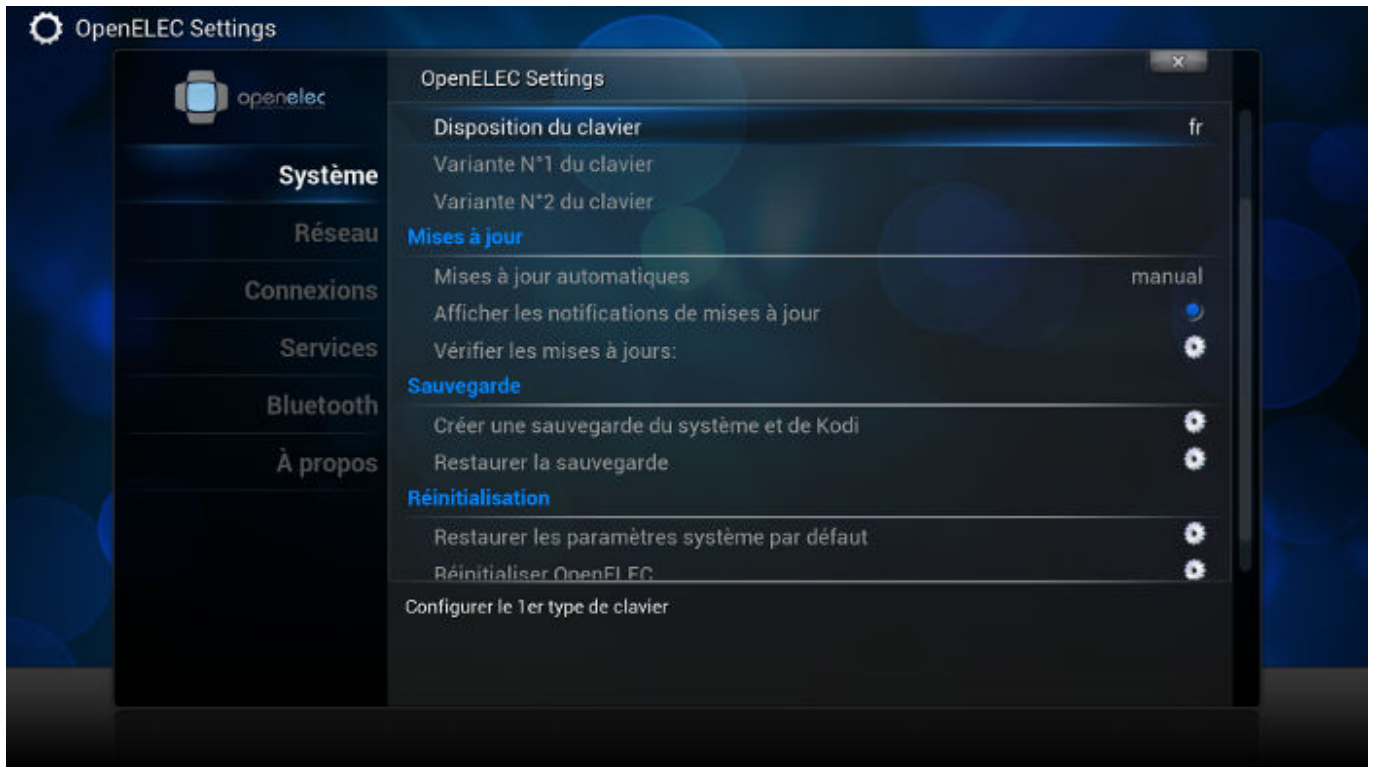
Mise à jour d'une version beta

Avec les versions beta d'OpenELEC, les mises à jour automatiques sont désactivées par défaut et il vaut mieux ne pas activer cette option (l'option se trouve dans l'extension OpenELEC) et procéder manuellement en téléchargeant tout simplement [la version dédiée aux Raspberry Pi voulue](#) (pas les images disque) dans le partage *Update* du bidule.

Configuration

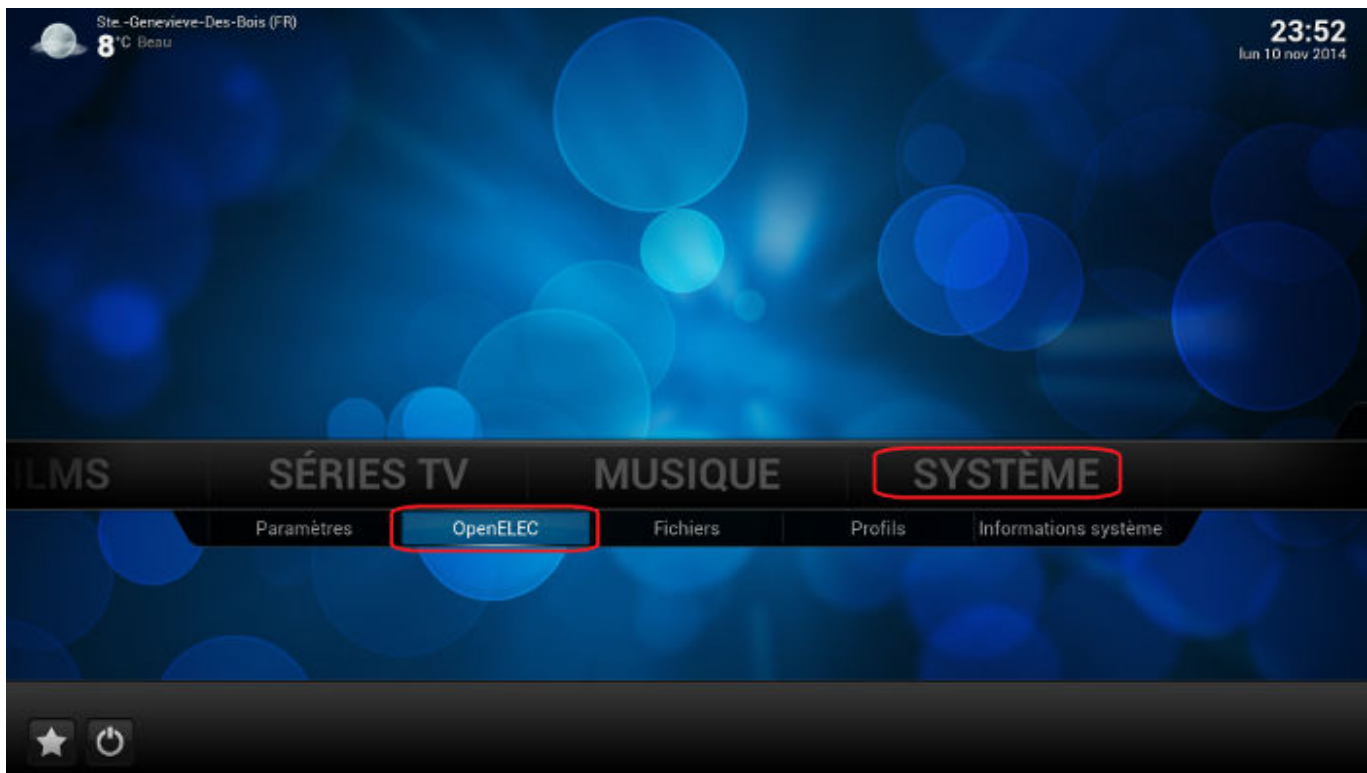
OpenELEC

La configuration avancée dont je parlais se fait via cette fenêtre:




Notez la présence de la rubrique **[Sauvegarde]** et n'oubliez pas de jeter un œil à la rubrique **[Services]** pour vous assurer d'activer *SSH* et *Samba*.

Avec le thème *Confluence*, cet écran est accessible par la rubrique **[Système]**:



Avec les thèmes qui n'ont pas prévu d'accès direct, on peut y accéder par les extensions de type **[Programmes]**.

 Il est bien utile (en particulier pour les sauvegardes) d'ajouter au thème utilisé un raccourcis vers cette extension.

Kodi

Voir [l'article dédié](#).


Thème

Attention aux thèmes trop gourmands en ressources. Typiquement, si lors de l'installation d'un thème, une flopée d'extensions sont installées avant qu'il soit lui-même activé (il y aura toujours une notification pour chaque extension installées), on peut presque toujours partir du principe que le thème en question sera à priori plutôt lourd.

Si l'on désinstalle le thème concerné, les extensions restent, à moins de les désinstaller

manuellement 

Logiquement les autres thèmes ne les utilisent pas donc l'impact en les laissant devrait être assez limité mais on ne sait jamais...

 Notez qu'il n'y a aucun risque de provoquer une catastrophe en désinstallant par erreur une extension encore utilisée



puisque'un message indiquer que c'est tout simplement impossible lors de la tentative de désinstallation.

Sauvegarde et restauration

Donc l'outil OpenELEC (plus ou moins accessible selon le skin), inclut un système de sauvegarde et de restauration qui concerne aussi bien OpenELEC lui-même que Kodi. Génial ! 😎 <3

Enfin au moins sur le papier... 😞

Dans la pratique, il peut y avoir des effets de bord étranges comme se retrouver avec tous les films dédoublés. Je me suis ainsi retrouvé avec deux éléments portant le même nom et pointant sur le même fichier et pourtant bien distincts avec chacun son statut "vu" ou pas et il a fallu restauré une sauvegarde de la base de données pour m'en sortir (car la fonction de [Nettoyage de la médiathèque]

ne se préoccupe que des éléments disparus, pas des doublons 😐).

En conclusion: le système de sauvegarde et de restauration est bien pensé (**très bien** même, je persiste) mais pêche légèrement par la pratique et ne dispense pas de faire par ailleurs des sauvegardes régulières de la base de données (qu'il s'agisse d'une sauvegarde MySQL automatique ou d'un export au format "1 fichier" de Kodi).

Configuration avancée, optimisations et bidouilles

Un peu de nettoyage

Par défaut, OpenELEC crée sur la carte SD des dossiers destinés à recevoir les images, les films, etc. Autant dire qu'ils ne servent pas à grand chose et qu'on peut sans problème les supprimer pour faire un peu de ménage.

- En ce qui concerne les sources, il suffit d'aller dans la rubriques [Fichiers] de chacun des types de médias, de sélectionner la source en question, d'invoquer le menu contextuel (touche [C]) et de choisir [Supprimer].
- Pour les partages, c'est un peu plus compliqué car la configuration est assez particulière: on trouve bien un fichier `/etc/samba/smb.conf` contenant toutes la configuration *Samba* mais

même pour l'utilisateur *root* il est en lecture seule 😞. Voir le paragraphe ci-dessous.

Beaucoup de nettoyage



Kodi n'est pas bon en ménage

Même après désinstallation, un *skin* laisse derrière lui des plugins et des lignes inutiles dans le fichier *userdata/guisettings.xml* (environ 6000 lignes quand même ! bon d'accord, j'ai chargé la mule en terme du nombre de *skins* testés: à peu près tous soit presque 40). Je pars du principe qu'avec une machine faible comme le Raspberry, tout élément inutile chargé en mémoire et tout processus lancé inutilement ralentissent la machine pour rien.

Pour les plugins en trop, certains sont encore chargés en mémoire et lancent un script de temps en temps même si le *skin* qui l'a installé est désinstallé. Pour le nettoyage, [c'est par là que ça se passe](#). Pour le fichier *guisettings.xml*, sous OpenELEC, il se trouve dans */storage/.kodi/userdata* et ne peut être modifié tant que Kodi tourne (toute modification sera écrasée au reboot). Voici donc la méthode la plus simple que j'ai trouvé (via *SSH*):

```
# systemctl stop xbmc
```

Cette commande va fermer Kodi mais pas OpenELEC et à partir de là, on peut modifier sans problème le fichier */storage/.kodi/userdata* (via *SSH* ou un programme comme *WinSCP* associé à un éditeur comme *Geany* ou *Notepad++*).

Dans la même idée (mais avec le seul intérêt de réduire le temps des sauvegardes OpenELEC), les packages téléchargés lors de l'installation des *plugins* restent ad vitam eternam dans */storage/.kodi/addons/packages* mais le nettoyage est bien plus simple: il suffit de supprimer les fichiers.

Personnaliser Samba

Le fichier */etc/samba/smb.conf* est donc protégé en écriture et totalement géré par OpenELEC mais il est heureusement possible de modifier la configuration par défaut à partir du fichier */storage/.config/samba.conf.sample*.

- commencer par créer une copie sans l'extension finale:

```
cp /storage/.config/samba.conf.sample /storage/.config/samba.conf
```

- éditer le fichier de configuration créé ci-dessous pour y supprimer les partages en trop

```
nano /storage/.config/samba.conf
```



Il est plus simple d'utiliser *nano* car le comportement de *vi* est assez surprenant sous OpenELEC et je ne suis pas parvenu à supprimer la moindre ligne en mode insertion donc il faut utiliser la commande *d\$* qui supprime la ligne en cours (je suis loin d'être un pro de *vi*). On peut aussi se connecter en tant que root avec un logiciel comme *WinSCP* pour éditer les fichiers.

Voici une petite liste des partages que l'on peut supprimer sans le moindre scrupule: *Emulators, Music, Pictures, Recordings, TV Shows* et *Videos*.

Utiliser une clef USB pour les données

Le principal défaut des cartes SD est la corruption potentielle des données au fil du temps à cause du nombre de cycles d'écritures. Dans ce cas il n'y a plus qu'à la formater et tout recommencer 😞 La solution consiste à utiliser une clef USB pour la partition Storage ce qui va accessoirement accélérer l'accès aux *artworks* en cache et aux menus 😊

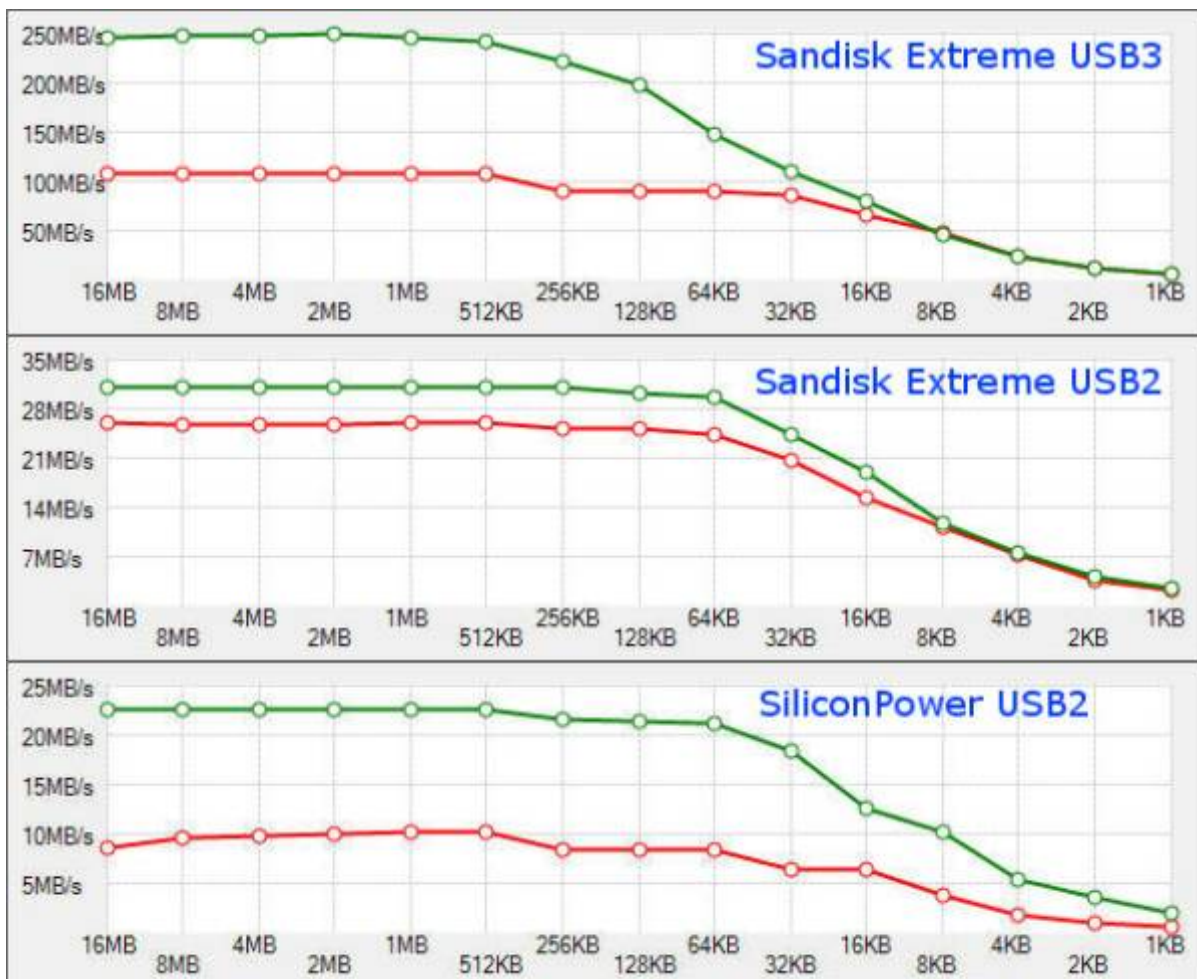
Le hardware

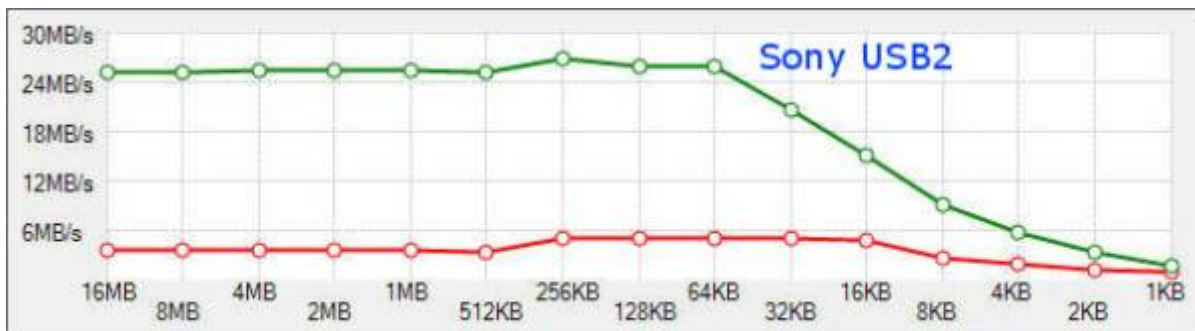
Pour en arriver là, il faut une **bonne** clef USB (pour des fichiers plutôt petits, les *artworks* en tous genres se situant entre 30 et 300Ko).

Mon choix s'est porté sur une SanDisk Extreme USB 3.0 de 32Go.

Alors oui, cette clé est bien trop grosse et surtout c'est du gâchis d'utiliser une clé USB 3.0 alors que le Raspberry Pi n'est équipé qu'en USB 2.0. Et bien pas vraiment.

Voici les résultats de tests réalisés avec [FlashBench^{2\)}](#), respectivement pour la clef SanDisk Extreme en USB 3.0, la même sur un port USB 2.0, une clef Silicon Power de 8Go (USB 2.0) et une clef Sony de 16Go (USB 2.0 aussi):





Si les courbes d'écriture sont comparables et ne compenseraient pas la différence de prix, les courbes de lecture placent la SanDisk Extreme en USB 2.0 nettement devant les deux autres.

La procédure

- Commencer par une sauvegarde du système avec l'extension OpenELEC (on est **jamais** trop prudent)
- Éteindre le bidule (quel que soit le système installé, l'insertion d'un dispositif USB peut provoquer un redémarrage avec une éventuelle perte des données)
- Formater la clef avec une partition primaire en Ext4 avec par exemple l'étiquette StorageUSB(sous Windows, il faudra un outil comme [MiniTool Partition Wizard Home Edition](#))
- Insérer la clef USB
- Démarrer le bidule si l'insertion de la clef ne l'a pas fait
- Se connecter en SSH
- Trouver le point de montage de la clef USB:

```
rootfs on / type rootfs (rw)
devtmpfs on /dev type devtmpfs (rw,relatime,size=185412k,nr_inodes=46353,mode=755)
proc on /proc type proc (rw,relatime)
sysfs on /sys type sysfs (rw,relatime)
/dev/mmcblk0p1 on /flash type vfat (ro,noatime,fmask=0022,dmask=0022,codepage=437,iocharset=ascii,shortname=mixed,utf8,errors=remount-ro)
/dev/mmcblk0p2 on /storage type ext4 (rw,noatime,data=ordered)
/dev/loop0 on / type squashfs (ro,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620)
tmpfs on /run type tmpfs (rw,nosuid,nodev,mode=755)
tmpfs on /sys/fs/cgroup type tmpfs (ro,nosuid,nodev,noexec,mode=755)
cgroup on /sys/fs/cgroup/systemd type cgroup (rw,nosuid,nodev,noexec,relatime,xattr,release_agent=/usr/lib/systemd/systemd-cgroups-agent,name=systemd)
mqueue on /dev/mqueue type mqueue (rw,relatime)
tmpfs on /tmp type tmpfs (rw)
debugfs on /sys/kernel/debug type debugfs (rw,relatime)
tmpfs on /var type tmpfs (rw,relatime)
/dev/sda1 on /var/media/StorageUSB type ext4 (rw,nosuid,nodev,noexec,noatime,data=ordered)
```

Habituellement, ce sera quelque chose comme /var/media/ suivi de l'étiquette choisie lors du formatage, ici /var/media/StorageUSB.

- Copier le contenu visible **et caché** du dossier /storage vers la clef USB

```
cp -r /storage/. /var/media/point_de_montage
```




Si `/.` est ommis, le contenu caché ne sera pas copié et, par ailleurs, le point de montage est évidemment à adapter en fonction du retour de la commande `mount` ci-dessus.



La copie peut être longue donc c'est le bon moment pour une pause café

- Trouver l'identifiant unique ou *UUID* de de la clef USB

```
blkid
```

```
/dev/mmcblk0: PTUUID="9c96996e" PTTYPE="dos"  
/dev/mmcblk0p1: SEC_TYPE="msdos" UUID="2638-B68D" TYPE="vfat" PARTUUID="9c96996e-01"  
/dev/mmcblk0p2: UUID="32f503c2-0f95-4e18-871b-94d8d08195e2" TYPE="ext4" PARTUUID="9c96996e-02"  
/dev/loop0: TYPE="squashfs"  
/dev/sda1: LABEL="StorageUSB" UUID="740fa9af-c200-d001-1007-a9afc200d001" TYPE="ext4"
```

Noter le code hexadimal (le fameux identifiant) correspondant à la clef USB (ici: 740fa9af-c200-d001-1007-a9afc200d001).

- Remonter la partition `/flash` pour la rendre accessible en écriture

```
mount /flash -o remount,rw
```

- Sauvegarder le fichier `cmdline.txt`:

```
cp /flash/cmdline.txt /flash/cmdline.sd
```

L'extension `sd` permettra de se souvenir si nécessaire qu'il s'agit du fichier qui était à l'origine sur la carte SD.

- Éditer le fichier pour remplacer la portion `disk=/dev/mmcblk0p2` afin d'indiquer que l'on souhaite utiliser la clef USB.

```
nano /flash/cmdline.txt
```

Et voici le résultat dans mon cas:

```
GNU nano 2.3.5 File: /flash/cmdline.txt  
boot=/dev/mmcblk0p1 disk=UUID=740fa9af-c200-d001-1007-a9afc200d001 quiet
```

- Il est temps de redémarrer le bidule:

```
reboot
```

Pour vérifier, il suffit de se connecter à nouveau via SSH et d'utiliser la commande `mount` qui devrait renvoyer une ligne indiquant que la partition `/storage` pointe bien sur la clef USB:

```
rootfs on / type rootfs (rw)
devtmpfs on /dev type devtmpfs (rw,relatime,size=185412k,nr_inodes=46353,mode=755)
proc on /proc type proc (rw,relatime)
sysfs on /sys type sysfs (rw,relatime)
/dev/mmcblk0p1 on /flash type vfat (ro,noatime,fmask=0022,dmask=0022,codepage=437,iocharset=ascii,shortname=mixed,utf8,errors=remount-ro)
/dev/sda1 on /storage type ext4 (rw,noatime,data=ordered)
/dev/loop0 on / type squashfs (ro,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620)
tmpfs on /run type tmpfs (rw,nosuid,nodev,mode=755)
tmpfs on /sys/fs/cgroup type tmpfs (ro,nosuid,nodev,noexec,mode=755)
cgroup on /sys/fs/cgroup/systemd type cgroup (rw,nosuid,nodev,noexec,relatime,xattr,release_agent=/usr/lib/systemd/systemd-cgroups-agent,name=systemd)
mqueue on /dev/mqueue type mqueue (rw,relatime)
tmpfs on /tmp type tmpfs (rw)
debugfs on /sys/kernel/debug type debugfs (rw,relatime)
tmpfs on /var type tmpfs (rw,relatime)
/dev/mmcblk0p2 on /var/media/mmcblk0p2-mmc-SDU1_0xe1500363- type ext4 (rw,nosuid,nodev,noexec,noatime,data=ordered)
```

Pour aller au bout de l'idée, on peut éteindre le bidule le temps de protéger la carte SD contre l'écriture avec le loquet prévu à cet effet mais il faudra la déverrouiller pour pouvoir mettre à jour le système.



Pensez à une petite sauvegarde



Cette procédure a été établie grâce à [cette page](#).

Overclocker le bidule



Attention: d'une manière générale, l'overclocking est déconseillé par les fabricants de matériel informatique car il peut provoquer des dégâts matériels, réduire sa durée de vie ou simplement le rendre instable. Il est aussi connu, sur le Raspberry Pi, pour être potentiellement source de corruption des données, et par ailleurs, la résistance à l'overclocking n'est pas la même d'un Raspberry Pi à l'autre.

Par défaut, le processeur est bridé à 700MHz mais certains le font tourner jusqu'à 1GHz



Avant de commencer

Avant d'overclocker le bidule, il est important de jeter un simple coup d'œil aux températures normales que l'on trouve dans les [Informations système] accessibles par le menu [Paramètres]:

| | |
|----------------------------------|-----------------|
| Au repos | 48 à 50° |
| Après 60mins de visionnage vidéo | 53 à 56° |

Mise en place

On peut jouer sur 4 paramètres:

- la fréquence du processeur (**arm_freq**)
- la fréquence du GPU (**gpu_freq**)
- la fréquence de la RAM (**sdram_freq**)
- le survoltage (**over_voltage**)



Le survoltage est réputé particulièrement hasardeux sur Raspberry Pi (d'ailleurs il paraît que sans survoltage, on ne met pas la garantie en danger mais cela reste à vérifier).

Et voici une présentation des réglages "types" (les valeurs intermédiaires sont tout à fait possibles):

| Overclocking | arm_freq | core_freq | sdram_freq | over_voltage |
|----------------------------|-----------------|------------------|-------------------|---------------------|
| Aucun (valeurs par défaut) | 700 | 250 | 400 | 0 |
| Leger | 800 | 300 | 400 | 0 |
| Moyen | 900 | 333 | 450 | 2 |
| Élevé | 950 | 450 | 450 | 6 |
| Extrême | 1000 | 500 | 500 | 6 |

* Il faut se connecter en SSH et modifier le fichier `/flash/config.txt` mais pour cela, il faut avant tout passer la mémoire flash en mode écriture:

```
mount /flash -o remount,rw
```

- Par principe, il vaut mieux sauvegarder le fichier original:

```
cp /flash/config.txt /flash/config.txt.original
```

- Puis enfin éditer le fichier:

```
nano /flash/config.txt
```

- Décommenter et éditer les lignes ci-dessous:

```
arm_freq=800
core_freq=300
sdram_freq=400
```

Vous êtes évidemment libre de choisir les valeurs qui vous conviennent, à vos risques et périls



- Il ne reste plus qu'à rebooter:

reboot

NFS ou SMB?

La bataille fait rage 😄
Il y a tout de même vraiment beaucoup d'adeptes du NFS qui prouvent, chiffres à l'appui, que leur protocole favori est plus rapide que SMB et moins gourmand au niveau du processeur. Au final, tout

dépend du hardware, du réseau, et peut-être du taux d'humidité de l'air 😊
Pour ma part, SAMBA est en place sur le serveur Debian sur lequel mes fichiers sont stockés et c'est une nécessité pour les partager avec des PCs Windows et par ailleurs, même à travers le réseau CPL de la Freebox, il n'y a pas le moindre problème de lecture, même sur les films HD 1080, donc pas de nécessité de mettre en place NFS.

1)

suite au changement de nom à cause d'une histoire de droits sur l'utilisation du nom complet original « Xbox Media Center »

2)

il vaut mieux décocher la case *[Send Report]* pour éviter que le logiciel ne plante à la fin

From:
<https://wiki.geekitude.fr/> - **Geekitude**

Permanent link:
<https://wiki.geekitude.fr/archives/info/os/openelec/accueil>

Last update: **2022/09/14 00:12**

