

Paul KLAK

François MONTEIL

Compte rendu – Linux pour l'embarqué sur clé USB



Table des matières

Introduction	3
Objectifs	3
I – Préparation de la clé USB	4
Configuration de la clé.....	4
Création du script :	4
II – Installation de grub	5
III - Installation du noyau linux	5
Noyau linux statique :	5
Noyau linux dynamique :	6
IV – Installation de Busybox	
(http://pficheux.free.fr/articles/lmf/hs24/busybox/bb_nutshell.pdf)	7
Busybox statique :	7
Busybox dynamique :	7
Fichier rcS	8
V – Clavier FR (azerty)	10
VI – Réseau	11
Adresse IP fixe :	11
Adresse IP dynamique :	11
VII – Utilisateurs.....	12
VIII – Ncurses	13

Introduction

Ce compte rendu a pour objectif d'expliquer ce qui a été fait lors de ces quatre premières séances de linux pour l'embarqué.

La configuration et utilisation de cette clé linux a été effectué d'abord sur un ordinateur des salles de cours de l'IG2I, puis reproduit sur un HP 6305 afin de mieux comprendre tout le processus d'installation d'un linux pour l'embarqué.

Objectifs

Les différents objectifs de ces séances sont les suivants :

Basique	Clavier FR
	Busybox dynamique
	Réseau
	Utilisateurs
	Login sur première console
ncurses	Recompilation
	Installation sur PC dév.
	Makefile exemples
	Test exemples avec menu
	Installation sur clé
	Test clé
ncurses : allez plus loin	Automatisation terminfo
	Menu de lancement
	Menu s'affiche au démarrage
framebuffer	Noyau pinguins
	Fichiers de périphériques
	Vga = ask puis automatique
	Librairies images installés
	Fbv
	Résolution problème gif
framebuffer : allez plus loin	Librairies images recompilés
	Fbvshot
	Fbvdump
	Fim
SDL	Installation lib
	Compilation exemples
	Compilation jeu
	Copie libs. Sur clé
	Test sur clé
	Souris avec SDL
	Souris en console
SDL : allez plus loin	Recompilation SDL
Réseau	Serveur WEB
	Serveur SSH
	Empreinte mémoire finale

I – Préparation de la clé USB

Configuration de la clé

Dans un premier temps il faut paramétrer la clé de manière qu'elle puisse accueillir notre installation.

Pour ceci nous avons décidé de créer une partition de 200 Mo au format ext4.

Création du script :

Afin de ne pas avoir à répéter les mêmes manipulations pour une prochaine installation, nous œuvrons à la mise en place d'un script Bash qui ferait entièrement l'installation de la clé.

La première étape de ce script sera de vérifier qu'il est bien exécuté en tant que root car ce statu est nécessaire aux différentes actions qui vont suivre.

Ensuite on demande à l'utilisateur de spécifier le chemin de la clé

Puis on procède à son partitionnement et son formatage.

II – Installation de grub

Après avoir configuré la clé USB, il est nécessaire d'installer grub qui se chargera de nous fournir une interface nous permettant de booter ensuite sur un noyau installé sur la clé.

On crée alors un dossier boot à la racine de la clé, et on y installe le grub à l'aide d'une ligne de commande :

```
grub-install --force --removable --boot-directory=/mnt/cle/boot /dev/sdb (sdb à vérifier à l'aide la commande df -h)
```

La configuration de celui-ci se fera en fonction du mode d'installation de linux.

III - Installation du noyau linux

Noyau linux statique :

Tout d'abord on laisse le choix à l'utilisateur s'il veut installer linux en statique sur sa clé.

Pour installer linux en statique sur la clé, il faut copier le noyau et les bibliothèques du linux de l'ordinateur, soit on copie les fichiers vmlinuz et initrd présent dans le dossier /boot de l'ordinateur vers le dossier boot de la clé USB.

Ensuite il faut créer un fichier de config pour que le grub sache quoi utiliser :

```
menuentry 'LPE' {
    set root='hd0,msdos1'
    linux16 /boot/vmlinuz root=/dev/sdb1 rootdelay=5
    linux16 /boot/initrd
}
```

Une fois ces manipulations faites, on peut redémarrer la machine et utiliser des fonctionnalités de linux en ligne de commande.

Noyau linux recompilé :

Cette fois l'installation ne se fait pas à l'aide des éléments déjà présent sur le poste de travail. Il faut télécharger un noyau linux sur internet, pour ce TP je l'ai téléchargé sur <https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux.git> qui propose plusieurs version de noyau linux stable, nous pouvons alors télécharger le dernier en date.

Une fois téléchargé, il faut le décompresser, puis prendre connaissance des pilotes qui seront nécessaire au fonctionnement du noyau linux à l'aide de commandes linux :

- dmesg qui permet de voir les périphériques connectés.
- Lspci -vv qui permet de voir les périphériques connectés au bus PCI.
- lsmod qui liste les pilotes chargés dans le noyau actuel.

Une fois une liste des pilotes utiles déterminé on peut configurer le noyau à partir d'un fichier de config vierge en effectuant la commande make allnoconfig.

Pour éditer la configuration il faut faire un make menuconfig (Peut nécessiter l'installation de librairies supplémentaire sur le poste tel que bison, flex ou encore libelf-dev)

Pour une configuration minimal fonctionnel il faut cocher 64 bits, network support, et certain élément de Devices drivers à l'aide de ce que l'on a pu déterminer plus tôt.

Une fois la configuration terminée, on peut exécuter la commande make -j9 qui va produire le nouveau noyau que l'on utilisera sur la clé.

On copie ce noyau (bzImage) dans le dossier boot de la clé USB.

Et pour finir on configure le grub de manière à ce qu'il démarre sur ce noyau :

```
menuentry 'LPE' {  
    set root='hd0,msdos1'  
    linux16 /boot/bzImage root=/dev/sda1 rootdelay=5  
}
```

On remarque que cette fois on spécifie sda1 et non sdb1, la raison est que dans le cas du noyau linux statique le disque dur est détecté et identifié en tant que sda1 et donc la clé en tant que sdb1.

Pour le linux dynamique, le disque n'est pas détecté, la clé est donc identifiée en tant que sda1.

Le script copie directement le bzImage déjà généré.

IV – Installation de Busybox

(http://pficheux.free.fr/articles/lmf/hs24/busybox/bb_nutshell.pdf)

Tout d'abord il faut télécharger Busybox sur <https://busybox.net/>, nous téléchargeons alors la dernière version stable en date, dans notre cas il s'agit de la version 1.29.3.

Après avoir décompressé Busybox, on doit le configurer, et pour ça il faut effectuer la commande `make menuconfig` (Nécessite au préalable l'installation de `libncurses5` à l'aide de la commande `apt install libncurses5-dev`).

Ensuite en prévision de la suite des événements, nous créons les dossiers nécessaires sur la clé :

- dev
- proc
- sys
- lib
- lib64

Busybox statique :

Une fois dans le menu de configuration, nous n'avons qu'à cocher la case « Build static binary (no shared libs) » qui se trouve dans le menu « Settings ».

Une fois la configuration terminée, on doit effectuer la commande `make -j9` qui va nous permettre de récupérer un exécutable Busybox.

Et pour finir il faut installer Busybox sur la clé à l'aide de la commande :

```
Make CONFIG_PREFIX=/CHEMIN_DE_LA_CLE install
```

Busybox dynamique :

Cette fois nous décochons la case « Build static binary (no shared libs) » qui se trouve dans le menu « Settings ».

De la même manière que pour le Busybox statique nous devons exécuter les commandes `make -j9` puis `Make CONFIG_PREFIX=/CHEMIN_DE_LA_CLE install`.

La différence étant que les librairies ne sont pas installées avec comme le laisse deviner « no shared libs ».

Il faut donc déterminer quelles sont-elles puis les mettre à disposition de Busybox.

Pour cela il y a une commande qui permet de déterminer les librairies que requiert un exécutable : `Ldd /CHEMIN_DE_LA_CLE/bin/busybox` qui nous retourne une liste de librairie présente sur le poste de travail dans les dossiers `lib` ou `lib64`.

Il nous suffit alors de copier ces librairies dans les dossiers `lib` ou `lib64` créés sur la clé précédemment.

Il nous faut ensuite remplir le dossier dev de la clé à l'aide de la commande MAKEDEV.

L'exécutable MAKEDEV est utilisé pour créer les fichiers de périphérique dans /dev.

Les fichiers de périphérique sont des fichiers spéciaux par lesquels les applications peuvent interagir avec le matériel.

(Information trouvé sur <https://packages.debian.org/fr/jessie/makedev>)

Il peut être nécessaire d'installer makedev sur le poste à l'aide de la commande `apt install makedev`.

Une fois l'installation faite il faut se placer dans le dossier /dev à la racine de la clé, puis effectuer la commande : `/sbin/MAKEDEV generic console`.

Fichier rcS

Pour un meilleur fonctionnement de Busybox il est important de monter certains systèmes de fichiers tel que /proc, pour ce faire nous pouvons instancier ces montages dans le fichier rcS que nous allons placer dans /etc/init.d et que l'on rendra exécutable (`chmod +x rcS`) et qui se lancera au démarrage.

On y ajoute :

```
#!/bin/sh
mount -t proc proc /proc
mount -o remount,rw /
mount -a
```

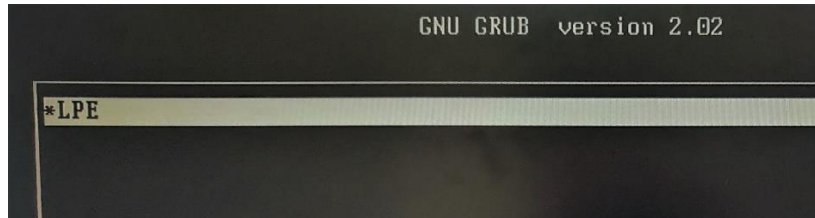
La dernière ligne monte tous les systèmes de fichiers décrits dans /etc/fstab. Il faut donc créer ce fichier et le remplir de cette manière :

/dev/hda4	/	ext3	defaults	1	1
None	/dev/pts	devpts	mode=622	0	0
None	/proc	proc	defaults	0	0

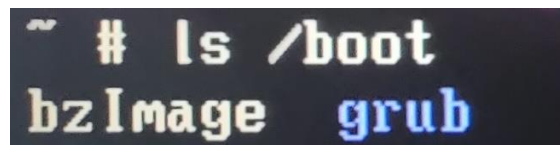
Test de fonctionnement :

Nous allons maintenant procéder à des tests nous permettant de déterminer le fonctionnement des éléments installés et donc de valider le point : **Busybox dynamique**.

Grub :



Noyau linux dynamique :

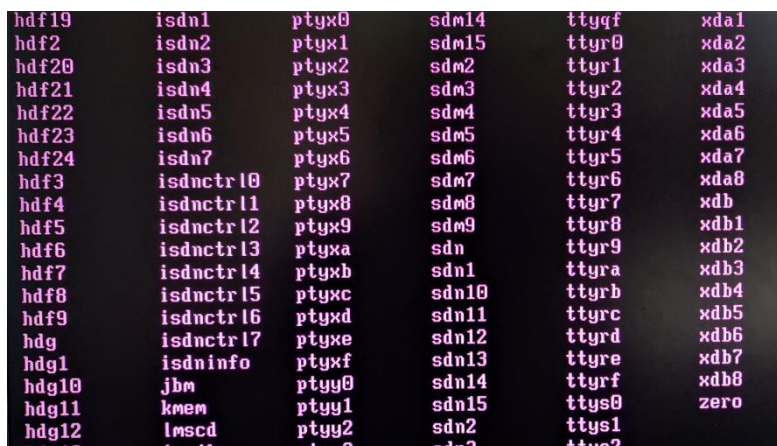


Installation Busybox dynamique :



On peut voir que les librairies ont bien été importées et que Busybox est bien installé.

Makedev :



/dev a bien été rempli grâce à la commande MAKEDEV

V – Clavier FR (azerty)

Il faut au préalable faire attention que les paramètres `dumpkmap` et `loadkmap` sont activés dans la configuration de Busybox (Console Utilities)

Ensuite on récupère un fichier `.kmap` qui est en réalité un fichier de configuration du clavier, on se sert pour cela de la commande `./bin/dumpkmap > etc/azerty.kmap` à la racine de la clé.

Le fichier est alors dans le dossier `etc` de la clé, nous pouvons ajouter une ligne dans le fichier `profile` : `loadkmap < /etc/azerty.kmap`, cette commande va configurer le clavier en azerty au démarrage du linux.

VI – Réseau

Pour le réseau nous avons le choix entre utiliser une adresse IP fixe, ou dynamique. Nous laissons alors le choix à l'utilisateur lorsqu'il utilise le script d'installation de la clé.

Adresse IP fixe :

Dans le cas où l'utilisateur choisit de fixer son adresse IP, nous lui demandons d'abord l'adresse qu'il veut utiliser, celle-ci sera renseignée dans le rcS précédé de quelques lignes :

```
#Configuration du réseau
Ifconfig lo 127.0.0.1
Ifconfig eth0 [ADRESSE IP]
```

Nous devons aussi copier des fichiers de la machine de préparation dans un dossier /lib/modules que l'on aura créé au préalable.

```
cp -a -r -f * /mnt/cle/lib/modules
Il y a des modifications par rapport à busybox in a nutshell, du aux différences de version.
cp /etc/modprobe.conf /mnt/cle/etc
```

Adresse IP dynamique :

Dans le cas d'une adresse IP dynamique, busybox met à notre disposition un script qui va nous permettre de requêter le serveur DHCP afin d'obtenir une adresse IP dynamique.

Pour cela il faut créer un dossier udhcpc et y mettre le script :

```
mkdir -p /mnt/cle/usr/share/udhcpc
cp [LEMB]/build/busybox/examples/udhcp /mnt/cle/usr/share/udhcpc/default.script
chmod +x /mnt/cle/usr/share/udhcpc/default.script
```

Ensuite on ajoute dans le rcS la commande à lancer au démarrage de la machine :

```
#Configuration du réseau
udhcpc
```

VII – Utilisateurs

Pour pouvoir utiliser plusieurs utilisateurs, busybox met à notre disposition un fichier `inittab` que l'on doit copier dans le dossier `etc` de la clé.

Ensuite il faut créer deux fichiers `passwd` et `group` tout deux dans `etc` et écrire :

`passwd :`

`root :0 :Super User:/:/bin/sh`

`group :`

`root :x :0 :`

Il suffira ensuite de mettre un mot de passe à `root` avec la commande `passwd root` et de créer un utilisateur avec la commande `adduser` .

VIII – Ncurses

Installation sur ordinateur de développement

Tout d'abord sur l'ordinateur de préparation, nous devons télécharger ncurses, pour cela nous pouvons suivre les instructions du site ftdp.org/HOWTO/NCURSES-Programming-HOWTO/intro.html.

```
wget -c ftp://ftp.gnu.org/pub/gnu/ncurses/ncurses-6.1.tar.gz.
```

Une fois le dossier récupéré et décompresser, nous devons utiliser la commande

```
./configure --prefix=[DOSSIER DE DESTINATION]
```

```
make -j9
```

```
make install
```

Ce dossier de destination sera notre dossier de travail pour des programmes utilisant les bibliothèques ncurses.

On récupère le dossier ncurses_programs qui va nous permettre d'utiliser des programmes déjà écrit tel que hello_world.c et menu_simple.c.

Il nous faudra juste les compiler et pour ça nous allons faire un Makefile.

Rédaction du Makefile

```
all: hello_world menu
```

```
PATH_LIB=/home/eleve/Bureau/lemb/target_ncurses/lib
```

```
hello_world: hello_world.o
```

```
    gcc hello_world.o -L$(PATH_LIB) -o hello_world -lncurses
```

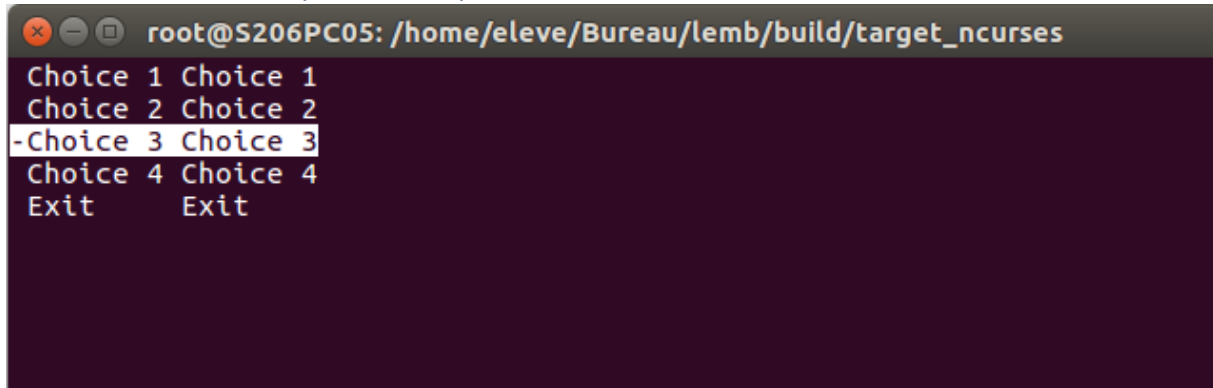
```
hello_world.o : hello_world.c
```

```
    gcc -c hello_world.c -o hello_world.o
```

```
menu_simple: menu_simple.c
```

```
    gcc -o menu_simple -L$(PATH_LIB) menu_simple.c -lmenu -lncurses
```

Test du menu_simple sur le pc de dev



```
root@S206PC05: /home/eleve/Bureau/lemb/build/target_ncurses
Choice 1 Choice 1
Choice 2 Choice 2
-Choice 3 Choice 3
Choice 4 Choice 4
Exit      Exit
```

Installation sur clé

Pour l'installation sur clé, nous copions les exécutables ainsi que les dossiers bin, includes et share.

Automatisation du terminfo

Pour automatiser le terminfo, nous rentrons une ligne supplémentaire dans le fichier /etc/profile :

```
# !bin/sh
```

```
export TERMINFO=/share/terminfo
```

Le DHCP (Dynamic Host Configuration Protocol)

Le DHCP (Dynamic Host Configuration Protocol) est un protocole réseau permettant d'assurer la configuration automatique d'une adresse IP. En lui adressant par ce biais :

- Une adresse IP automatique
- Un masque sous réseau

Dans le cadre d'un projet de linux embarqué, ce protocole peut s'avérer extrêmement utile pour les objets connectés. En effet la complexité d'un adressage non automatique rend le paramétrage d'adresse ip complexe et rend difficile l'accès au grand public et donc à une commercialisation.

Pour ces raisons l'Internet Software Consortium a développé la solution DHCP.

La machine n'ayant pas automatiquement d'adresse IP prédéfinie à son démarrage émet un paquet sur le réseau local à destination d'un serveur DHCP programmé et possédant une adresse IP fixe. Il y passe en paramètre les informations relatives au type de requête, port ...)

Les différents types de paquets :

Number	Client	Serveur
1	DHCPDISCOVER (localiser le serveur)	
2		DHCPOFFER (réponse du serveur)
3	DHCPREQUEST (requête du client au server. ex: prolonger son bail)	
4		DHCPACK (réponse du serveur qui contient des paramètres et l'ip du client)
5		DHCPNAK (annonce la fin du bail du client ou une mauvaise configuration réseau)
6	DHCPDECLINE (annonce au serveur que l'adresse est déjà utilisée)	
7	DHCPRELEASE (libère son IP)	
8	DHCPINFORM (demande des paramètres locaux, il a déjà son adresse IP)	

Le premier paquet émis est le DHCPDISCOVER afin de localiser le serveur.

Le serveur répond avec DHCPOFFER pour soumettre une adresse IP au client.

Une fois la connexion établie par le client, il fait un DHCPREQUEST pour valider son adresse IP.

Le serveur répond par DHCPACK avec l'adresse IP pour confirmer l'attribution.

Pour des raisons d'optimisation, une date de validité appelé "bail" définit la durée d'allocation de l'adresse IP par soucis d'optimisation et de performance. Le client peut à tout moment prolonger l'échéance avec DHCPREQUEST.