



UNIVERSITÉ DE TECHNOLOGIE DE
COMPIÈGNE

MI11

MI11 - Rapport du TP 1 Linux embarqué

Clément BLANQUET et Rafik CHENNOUF

Juin 2017

Sommaire

1	Rapport TP 1 - Linux embarqué	3
	Exercice 1 : Prise en main de la carte DevKit8600	3
	1.1 Question 1.1	3
	1.2 Question 1.2	3
	1.3 Question 1.3	3
	1.4 Question 1.4	3
	1.5 Question 1.5	4
	Exercice 2	4
	2.1 Question 2.1	4
	2.2 Question 2.2	4
	2.3 Question 2.3	5
	Exercice 3	6
	3.1 Question 3.1	6
	3.2 Question 3.2	6
	3.3 Question 3.3	6
	3.4 Question 3.4	6
	3.5 Question 3.5	8
	Exercice 4 : Ajout de paquets	10
	4.1 Question 4.1	10
	4.2 Question 4.2	12
	4.3 Question 4.3	12
	4.4 Question 4.4	12
	Exercice 5 : Compilation manuelle du noyau	13
	5.1 Question 5.1	13
	5.2 Question 5.2	13
	5.3 Question 5.3	14
	5.4 Question 5.4	14
	5.5 Question 5.5	16
	5.6 Question 5.6	17
	5.7 Question 5.7	18
A	Messages de sortie du terminal entre l'allumage de la cible et le prompt de login	19

Table des figures

1.1	Dossier image	5
1.2	Mise en évidence du port série dans /proc/devices	7
1.3	Mise en évidence du port série dans /dev	8
1.4	Dossiers /bin et /sbin	8
1.5	Dossier /usr/bin	9
1.6	Dossier /usr/sbin	9
1.7	Dossier <i>ipk</i>	10
1.8	Dossier <i>ipk/devkit800</i>	11
1.9	Dossier <i>ipk/armv7a-vfp-neon</i>	12
1.10	Dossier <i>/usr/lib</i>	13
1.11	Configurations par défaut de <i>devkit8600</i>	14
1.12	Activation du driver pour les LEDs	15
1.13	Logs de démarrage	17
1.14	Logs du kernel	18

Rapport TP 1 - Linux embarqué

Exercice 1 : Prise en main de la carte DevKit8600

1.1 Question 1.1

La DevKit8600 est basée sur les processeurs AM3359 de Texas Instrument. Elle possède un ARM Cortex-A8 cadencé à 720 MHz avec un boot ROM On-Chip de 176KB, une mémoire NAND Flash de 512MB et une mémoire RAM (DDR3SDRAM) de 512MB. Elle possède un certain nombre d'interfaces comme un port LAN, une entrée/sortie audio, des USB, une interface JTAG... etc.

1.2 Question 1.2

On peut stocker le noyau via TFTP (Trivial File Transfer Protocol) et le système de fichiers via NFS (Network File System).

1.3 Question 1.3

Nous avons utilisé une liaison série pour nous connecter avec l'application cutecom.

Voici les paramètres que l'on a utilisés :

1. Bits per second : 115200
2. Data bits : 8
3. Parity : None
4. Stop bits : 1
5. Flow Control : None

1.4 Question 1.4

On constate qu'il y a une erreur au démarrage.

```
1 TFTP error: 'File not found' (1)
3 ERROR: can't get kernel image!
```

La cible ne peut donc pas démarrer. Le fichier manquant est l'image du kernel, appelé **uImage**. Il faudrait la placer dans le dossier `/tftpboot` sur notre PC, puisque la cible nous indique au démarrage qu'elle va chercher l'image à cet endroit.

1.5 Question 1.5

Même si l'image était présente, la cible ne pourrait pas démarrer car il manque un rootfs (système de fichier racine).

Exercice 2

2.1 Question 2.1

Le dossier `/home/mi11/poky/build/conf` contient les fichiers de configurations de poky qui permettent donc de configurer l'image selon nos besoins :

- `bblayers.conf`
- `local.conf`
- `sanity_info`
- `templateconf.cfg`

Le dossier `/home/mi11/devkit8600/meta-devkit8600` contient les fichiers spécifiques à notre cible qui vont permettre de construire une image qui lui est adaptée.

2.2 Question 2.2

Nous avons ajouté une ligne dans le fichier `bblayers.conf` :

```
1 BBLAYERS ?= " \
  /home/mi11/poky-dizzy-12.0.3/meta \
3 /home/mi11/poky-dizzy-12.0.3/meta-yocto \
  /home/mi11/poky-dizzy-12.0.3/meta-yocto-bsp \
5 /home/mi11/devkit8600/meta-devkit8600 \ // CELLE LA
  "
```

De plus, à la ligne 36 du fichier `local.conf`, nous avons inscrit :

```
MACHINE ??= "devkit8600"
```

Le nom `"devkit8600"` est en fait le nom du fichier de configuration du même nom situé ici : `/home/mi11/devkit8600/meta-devkit8600/conf/machine`, ce qui fait donc le lien entre la génération de l'image et les paramètres de la cible.

```

mi11@mi11-VirtualBox ~/poky/build/tmp/deploy/images $ ll devkit8600/
total 79736
drwxr-xr-x 2 mi11 mi11 4096 avril 14 15:13 ./
drwxr-xr-x 3 mi11 mi11 4096 avril 14 15:07 ../
-rw-r--r-- 1 mi11 mi11 5442 avril 14 15:12 core-image-base-devkit8600-20170414130644.rootfs.manifest
-rw-r--r-- 1 mi11 mi11 18255233 avril 14 15:13 core-image-base-devkit8600-20170414130644.rootfs.tar.bz2
-rw-r--r-- 1 mi11 mi11 30670848 avril 14 15:13 core-image-base-devkit8600-20170414130644.rootfs.ubi
-rw-r--r-- 1 mi11 mi11 29458432 avril 14 15:13 core-image-base-devkit8600-20170414130644.rootfs.ubifs
lrwxrwxrwx 1 mi11 mi11 57 avril 14 15:13 core-image-base-devkit8600.manifest -> core-image-base-devkit8600-20170414130644.rootfs.manifest
lrwxrwxrwx 1 mi11 mi11 56 avril 14 15:13 core-image-base-devkit8600.tar.bz2 -> core-image-base-devkit8600-20170414130644.rootfs.tar.bz2
lrwxrwxrwx 1 mi11 mi11 52 avril 14 15:13 core-image-base-devkit8600.ubi -> core-image-base-devkit8600-20170414130644.rootfs.ubi
lrwxrwxrwx 1 mi11 mi11 54 avril 14 15:13 core-image-base-devkit8600.ubifs -> core-image-base-devkit8600-20170414130644.rootfs.ubifs
-rw-r--r-- 1 mi11 mi11 21617 avril 14 15:07 modules--3.1.0-r0-devkit8600-20170410160636.tgz
lrwxrwxrwx 1 mi11 mi11 47 avril 14 15:07 modules-devkit8600.tgz -> modules--3.1.0-r0-devkit8600-20170410160636.tgz
-rw-r--r-- 1 mi11 mi11 294 avril 14 15:11 README - DO NOT DELETE FILES IN THIS DIRECTORY.txt
-rw-r--r-- 1 mi11 mi11 192 avril 14 15:13 ubinize.cfg
lrwxrwxrwx 1 mi11 mi11 46 avril 14 15:07 uImage -> uImage--3.1.0-r0-devkit8600-20170410160636.bin
-rw-r--r-- 1 mi11 mi11 3215152 avril 14 15:07 uImage--3.1.0-r0-devkit8600-20170410160636.bin
lrwxrwxrwx 1 mi11 mi11 46 avril 14 15:07 uImage-devkit8600.bin -> uImage--3.1.0-r0-devkit8600-20170410160636.bin
mi11@mi11-VirtualBox ~/poky/build/tmp/deploy/images $

```

FIGURE 1.1 – Dossier image

2.3 Question 2.3

Voici ce qu'on obtient dans le dossier `/home/mi11/poky/build/tmp/deploy/images` après la compilation :

On obtient notre image kernel *uImage* qui est en fait un lien symbolique vers le fichier *uImage-3.1.0-r0-devkit8600-20170410160636.bin*.

On voit également le système de fichier nommé *rootfs* qui est compressé dans une archive bzip2.

Si on compare la taille des fichiers qui viennent d'être générés avec ceux de notre VM sous Linux Mint :

- CIBLE :
 - Taille image : 3,2 MB
 - Taille rootfs : 1,8 MB
- HOTE :
 - Taille image : 6,6 MB (on le voit dans `/boot`)
 - Taille système de fichiers : environ 7 GB

On constate une bonne différence entre les deux images car l'image de la cible n'est faite que pour cette cible là, elle gère moins de choses et propose moins de fonctionnalités que celle de notre VM. Pour le système de fichiers, on constate une énorme différence car beaucoup plus de programmes sont installés sur notre VM par rapport à la cible sur laquelle rien ou presque n'est installé. La cible dispose en fait de la configuration minimale.

Exercice 3

3.1 Question 3.1

Comme nous l'avions indiqué dans le premier exercice, nous avons copié uImage dans le dossier `/tftpboot`, comme c'est indiqué dans le fichier de configuration `/etc/xinetd.d/tftp`. Le rootfs, lui, est copié dans le dossier `/tftpboot/rootfs`

3.2 Question 3.2

Messages de sortie du terminal entre l'allumage de la cible et le prompt de login : voir annexe A.1.

Analyse :

Du début jusqu'à "Starting kernel ...", la cible charge l'image via tftp et la lance. Une fois cela fait, Linux se lance et donne tout un tas d'informations sur le système (le nom de la machine, le type de CPU, la mémoire...). Viennent ensuite plusieurs opérations de calibrages et de tests, puis d'initialisations (comme le Bluetooth ou les interfaces réseaux). Enfin, le système se lance.

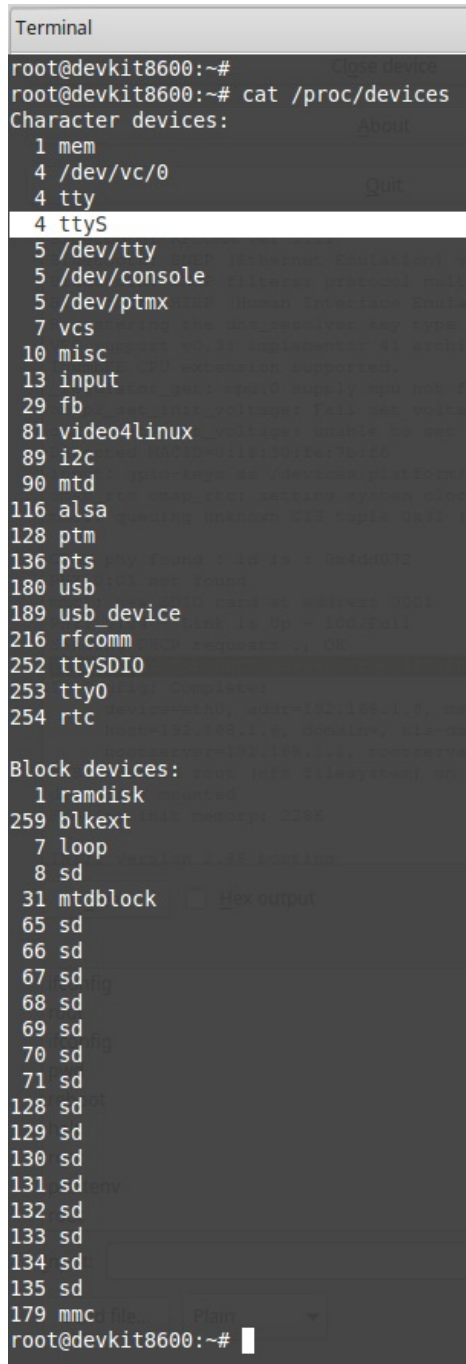
3.3 Question 3.3

L'IP de la cible est 192.168.1.6. On peut voir cette adresse sur les messages de sortie du boot : *"IP-Config : Got DHCP answer from 192.168.1.1, my address is 192.168.1.6 (sortie du boot)"*

3.4 Question 3.4

`proc/devices` contient les périphériques (dans la première section) ainsi que les stockages (dans la deuxième section).

Le numéro en début de ligne dans le fichier `/proc/devices` correspond au numéro mineur, qui indique si les périphériques sont gérés par le même driver.



```
Terminal
root@devkit8600:~#
root@devkit8600:~# cat /proc/devices
Character devices:
1 mem
4 /dev/vc/0
4 tty
4 ttys
5 /dev/tty
5 /dev/console
5 /dev/ptmx
7 vcs
10 misc
13 input
29 fb
81 video4linux
89 i2c
90 mtd
116 alsa
128 ptm
136 pts
180 usb
189 usb_device
216 rfcomm
252 ttySIO
253 tty0
254 rtc
Block devices:
1 ramdisk
259 blkext
7 loop
8 sd
31 mtdblock
65 sd
66 sd
67 sd
68 sd
69 sd
70 sd
71 sd
128 sd
129 sd
130 sd
131 sd
132 sd
133 sd
134 sd
135 sd
179 mmcblk0
root@devkit8600:~#
```

FIGURE 1.2 – Mise en évidence du port série dans `/proc/devices`


```

root@devkit8600:~# ls /dev/ttyS
ttyS0 ttyS1 ttyS2 ttyS3
root@devkit8600:~# ls /dev/ttyS

```

FIGURE 1.3 – Mise en évidence du port série dans /dev

3.5 Question 3.5

Voici une rapide description des dossiers :

```

root@devkit8600:~# ls /bin/
ash          df           ln           netstat      stty
bash         dmesg       login        pidof        su
busybox      dnsdomainname login.shadow pidof.sysvinit su.shadow
busybox.nosuid dumpkmap    ls           ping         sync
busybox.suid echo         lsmod        ping6        tar
cat          egrep       lsmod.kmod   ps           touch
chattr       false       mkdir        pwd          true
chgrp        fgrep       mknod        rm           umount
chmod        fgrep       mktemp       rmdir        uname
chown        gunzip      more         run-parts    usleep
cp           gzip        mount        sed          vi
cpio         hostname    mountpoint   sh           watch
date         kill        mountpoint.sysvinit sleep         zcat
dd           kmod        mv           stat

root@devkit8600:~# ls /sbin/
bootlogd    ifconfig    iwpriv       modinfo.kmod route        sysctl
depmod      ifdown     iwspy        modprobe     runlevel    syslogd
depmod.kmod ifup       killall5     modprobe.kmod runlevel.sysvinit telinit
fdisk       init        klogd        nologin      setconsole  udhcpd
fsck        init.sysvinit ldconfig     pivot_root   shutdown    vigr
fstab-decode insmod     loadkmap     poweroff     shutdown.sysvinit vigr.shadow
fstrim      insmod.kmod logread      poweroff.sysvinit start-stop-daemon vipw
getty       ip          losetup      reboot        sulogin     vipw.shadow
halt        iwconfig   lsmod        reboot.sysvinit swaponoff   swapoff
halt.sysvinit iwgetid    mkswap       rmmod        swapon      switch_root
hwclock     iwlist     modinfo      rmmmod.kmod

```

FIGURE 1.4 – Dossiers /bin et /sbin

- /bin : contient les binaires utilisables dès le boot, avant que la partition /usr soit montée. On y voit les commandes de base de type ls ou cat.
- /sbin : même chose que /bin, mais commandes nécessitant des droits particuliers (superuser d'où le s)

```

root@devkit8600:~# ls /usr/bin/
[[
ar
awk
basename
bashbug
bunzip2
bzip2
chage
chfn
chfn.shadow
chsh
chsh.shadow
chvt
ciptool
clear
cmp
cut
dbusclient
dbus-cleanup-sockets
dbus-daemon
dbus-launch
dbus-monitor
dbus-run-session
dbus-send
dbus-uuidgen
dc
deallocvt
dfutool
diff
dirname
dlist_test
du
dumpleases
env
evtest
expiry
expr
faillog
find
flock
free
fuser
gatttool
get device
get driver
get module
gpasswd
groups
groups.shadow
hcitool
head
hexdump
id
killall
l2ping
l2test
last
last.sysvinit
lastb
lastlog
less
logger
logname
lsusb
lsusb.py
md5sum
msg
msg.sysvinit
microcom
mkfifo
mpicalc
nc
nettle-hash
nettle-lfib-stream
newgidmap
newgrp
newgrp.shadow
newuidmap
nfctool
nohup
nslookup
od
openssl
opentv
opkg
opkg-cl
opkg-key
passwd
passwd.shadow
patch
pkcs11-conv
printf
psplash
psplash-default
psplash-write
python
python-config
python2
python2-config
python2.7
python2.7-config
rctest
readlink
realpath
renice
reset
rfcomm
scp
sdptool
seq
sexp-conv
sg
sha3sum
sort
ssh
stress
strings
systool
tail
tee
telnet
test
tftp
time
top
tr
traceroute
tty
udevadm
uniq
unzip
update-alternatives
uptime
usb-devices
usbhid-dump
users
utmpdump
utmpdump.sysvinit
vlock
wall
wall.sysvinit
wc
wget
which
who
whoami
wpa_passphrase
xargs
yes

```

FIGURE 1.5 – Dossier /usr/bin

- /usr/bin : contient les binaires généraux du système (comme les applications que l'utilisateur installe)

```

root@devkit8600:~# ls /usr/sbin/
addgroup
adduser
avahi-daemon
bccmd
bluetoothd
chpasswd
chpasswd.shadow
chroot
delgroup
deluser
dropbear
dropbearconvert
dropbearkey
dropbearmulti
fbset
genl-ctrl-list
groupadd
groupdel
groupmems
groupmod
grpck
grpconv
grpunconv
hciattach
hciconfig
hciemu
loadfont
logout
newusers
nl-class-add
nl-class-delete
nl-class-list
nl-classid-lookup
nl-cls-add
nl-cls-delete
nl-cls-list
nl-link-list
nl-pktloc-lookup
nl-qdisc-add
nl-qdisc-delete
nl-qdisc-list
ofonod
pwck
pwconv
pwunconv
rdate
rfkill
rpcbind
rpcinfo
run-postinsts
udhcpd
update-rc.d
update-usbids.sh
useradd
userdel
usermod
wpa_cli
wpa_supplicant

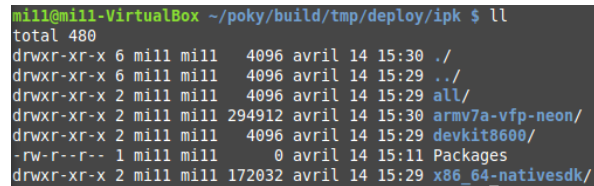
```

FIGURE 1.6 – Dossier /usr/sbin

- /usr/sbin : même chose que /usr/bin, mais nécessitant des droits particuliers (superuser d'où le s)

Exercice 4 : Ajout de paquets

4.1 Question 4.1



```
mi11@mi11-VirtualBox ~/poky/build/tmp/deploy/ipk $ ll
total 480
drwxr-xr-x 6 mi11 mi11  4096 avril 14 15:30 ./
drwxr-xr-x 6 mi11 mi11  4096 avril 14 15:29 ../
drwxr-xr-x 2 mi11 mi11  4096 avril 14 15:29 all/
drwxr-xr-x 2 mi11 mi11 294912 avril 14 15:30 armv7a-vfp-neon/
drwxr-xr-x 2 mi11 mi11  4096 avril 14 15:29 devkit8600/
-rw-r--r-- 1 mi11 mi11    0 avril 14 15:11 Packages
drwxr-xr-x 2 mi11 mi11 172032 avril 14 15:29 x86_64-native-sdk/
```

FIGURE 1.7 – Dossier *ipk*

Le dossier `/home/mi11/poky/build/tmp/deploy/ipk` est organisé par architectures (x86/x64, devkit8600, armv7a, etc.).

```

mill@mill-VirtualBox ~/poky/build/tmp/deploy/ipk/devkit8600 $ ll
total 36764
drwxr-xr-x 2 mill mill 4096 avril 14 15:29 ./
drwxr-xr-x 6 mill mill 4096 avril 14 15:30 ../
-rw-r--r-- 1 mill mill 4046 avril 14 15:08 base-files_3.0.14-r89_devkit8600.ipk
-rw-r--r-- 1 mill mill 840 avril 14 15:08 base-files-dbg_3.0.14-r89_devkit8600.ipk
-rw-r--r-- 1 mill mill 872 avril 14 15:08 base-files-dev_3.0.14-r89_devkit8600.ipk
-rw-r--r-- 1 mill mill 922 avril 14 15:08 base-files-doc_3.0.14-r89_devkit8600.ipk
-rw-r--r-- 1 mill mill 1072 avril 14 15:07 depmodwrapper-cross_1.0-r0_devkit8600.ipk
-rw-r--r-- 1 mill mill 704 avril 14 15:07 depmodwrapper-cross-dbg_1.0-r0_devkit8600.ipk
-rw-r--r-- 1 mill mill 736 avril 14 15:07 depmodwrapper-cross-dev_1.0-r0_devkit8600.ipk
-rw-r--r-- 1 mill mill 2722 avril 14 15:07 kernel-3.1.0_3.1.0-r0_devkit8600.ipk
-rw-r--r-- 1 mill mill 678 avril 14 15:07 kernel_3.1.0-r0_devkit8600.ipk
-rw-r--r-- 1 mill mill 30265112 avril 14 15:07 kernel-dev_3.1.0-r0_devkit8600.ipk
-rw-r--r-- 1 mill mill 3195142 avril 14 15:07 kernel-image-3.1.0_3.1.0-r0_devkit8600.ipk
-rw-r--r-- 1 mill mill 702 avril 14 15:07 kernel-modules_3.1.0-r0_devkit8600.ipk
-rw-r--r-- 1 mill mill 1922 avril 14 15:07 kernel-module-scsi-wait-scan_3.1.0-r0_devkit8600.ipk
-rw-r--r-- 1 mill mill 19494 avril 14 15:07 kernel-module-usbserial_3.1.0-r0_devkit8600.ipk
-rw-r--r-- 1 mill mill 3939110 avril 14 15:07 kernel-vmlinux_3.1.0-r0_devkit8600.ipk
-rw-r--r-- 1 mill mill 826 avril 14 15:08 opkg-config-base_1.0-r1_devkit8600.ipk
-rw-r--r-- 1 mill mill 682 avril 14 15:08 opkg-config-base-dbg_1.0-r1_devkit8600.ipk
-rw-r--r-- 1 mill mill 714 avril 14 15:08 opkg-config-base-dev_1.0-r1_devkit8600.ipk
-rw-r--r-- 1 mill mill 776 avril 14 15:07 packagegroup-base_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 692 avril 14 15:07 packagegroup-base-3g_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 726 avril 14 15:07 packagegroup-base-bluetooth_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 790 avril 14 15:07 packagegroup-base-dbg_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 822 avril 14 15:07 packagegroup-base-dev_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 660 avril 14 15:07 packagegroup-base-extended_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 668 avril 14 15:07 packagegroup-base-ipv6_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 688 avril 14 15:07 packagegroup-base-nfc_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 690 avril 14 15:07 packagegroup-base-nfs_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 712 avril 14 15:07 packagegroup-base-usb-gadget_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 742 avril 14 15:07 packagegroup-base-usb-host_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 712 avril 14 15:07 packagegroup-base-vfat_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 772 avril 14 15:07 packagegroup-base-wifi_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 680 avril 14 15:07 packagegroup-base-zeroconf_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 734 avril 14 15:07 packagegroup-core-boot_1.0-r17_devkit8600.ipk
-rw-r--r-- 1 mill mill 804 avril 14 15:07 packagegroup-core-boot-dbg_1.0-r17_devkit8600.ipk
-rw-r--r-- 1 mill mill 832 avril 14 15:07 packagegroup-core-boot-dev_1.0-r17_devkit8600.ipk
-rw-r--r-- 1 mill mill 684 avril 14 15:07 packagegroup-distro-base_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 722 avril 14 15:07 packagegroup-machine-base_1.0-r83_devkit8600.ipk
-rw-r--r-- 1 mill mill 26783 avril 14 15:29 Packages
-rw-r--r-- 1 mill mill 4259 avril 14 15:29 Packages.gz
-rw-r--r-- 1 mill mill 2558 avril 14 15:29 Packages.stamps
-rw-r--r-- 1 mill mill 868 avril 14 15:07 poky-feed-config-opkg_1.0-r2_devkit8600.ipk
-rw-r--r-- 1 mill mill 684 avril 14 15:07 poky-feed-config-opkg-dbg_1.0-r2_devkit8600.ipk
-rw-r--r-- 1 mill mill 714 avril 14 15:07 poky-feed-config-opkg-dev_1.0-r2_devkit8600.ipk
-rw-r--r-- 1 mill mill 1450 avril 14 15:09 shadow-securetty_4.2.1-r3_devkit8600.ipk
-rw-r--r-- 1 mill mill 690 avril 14 15:09 shadow-securetty-dbg_4.2.1-r3_devkit8600.ipk

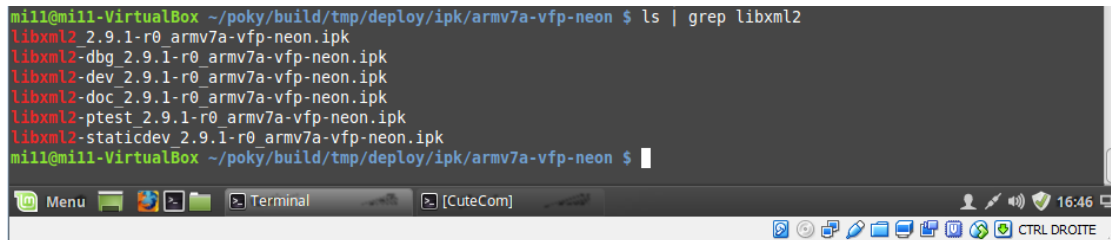
```

FIGURE 1.8 – Dossier *ipk/devkit800*

Les sous-dossiers contiennent des paquets d'extension *.ipk*. Le noyau se trouve dans le sous-dossier *devkit8600* comme nous pouvons le voir sur la copie d'écran ci-dessus. Quant au paquet *libxml2*, il se trouve dans le sous-dossier *armv7a-vfp-neon*.

4.2 Question 4.2

Les différents paquets relatifs à *libxml2* sont visibles sur la copie d'écran suivante.



```
mill@mill-VirtualBox ~/poky/build/tmp/deploy/ipk/armv7a-vfp-neon $ ls | grep libxml2
libxml2_2.9.1-r0_armv7a-vfp-neon.ipk
libxml2-dbg_2.9.1-r0_armv7a-vfp-neon.ipk
libxml2-dev_2.9.1-r0_armv7a-vfp-neon.ipk
libxml2-doc_2.9.1-r0_armv7a-vfp-neon.ipk
libxml2-ptest_2.9.1-r0_armv7a-vfp-neon.ipk
libxml2-staticdev_2.9.1-r0_armv7a-vfp-neon.ipk
mill@mill-VirtualBox ~/poky/build/tmp/deploy/ipk/armv7a-vfp-neon $
```

FIGURE 1.9 – Dossier *ipk/armv7a-vfp-neon*

Nous avons ensuite copié le fichier

```
libxml2_2.9.1-r0_armv7a-vfp-neon.ipk
```

dans le système de fichiers de la cible et nous avons installé le paquet avec la commande suivante :

```
opkg install libxml2_2.9.1-r0_armv7a-vfp-neon.ipk
```

4.3 Question 4.3

La première méthode pour copier le fichier dans le système de fichiers de la cible consiste à utiliser directement la commande *cp* comme suit :

```
sudo cp libxml2_2.9.1-r0_armv7a-vfp-neon.ipk /tftpboot/rootfs/home/root/
```

La deuxième méthode consiste à copier le fichier à distance via la commande *scp* qui se base sur *ssh* comme suit :

```
scp libxml2_2.9.1-r0_armv7a-vfp-neon.ipk root@192.168.1.6:/home/root/
```

4.4 Question 4.4

Les fichiers de *libxml2* installés par le gestionnaire de paquets se trouvent dans le dossier */usr/lib* de notre cible.

Ce dossier contient des fichiers *shared object* (.so) qui sont des bibliothèques/bibliothèques partagées dynamiques. Les .so sont des bibliothèques qui se chargent en mémoire au moment de l'exécution d'un programme qui les utilisent. Si plusieurs programmes les utilisant sont lancés en même temps, une seule instance de la bibliothèque dynamique réside en mémoire.

```

root@devkit8600:~# ls /usr/lib/
dbus/                                libgio-2.0.so.0.4000.0             libnl-genl-3.so.200
gio/                                libglib-2.0.so.0                   libnl-genl-3.so.200.20.0
libX11.so.6                         libglib-2.0.so.0.4000.0           libnl-nf-3.so.200
libX11.so.6.3.0                     libgmodule-2.0.so.0               libnl-nf-3.so.200.20.0
libXau.so.6                         libgmodule-2.0.so.0.4000.0        libnl-route-3.so.200
libXau.so.6.0.0                     libgmp.so.10                       libnl-route-3.so.200.20.0
libXdmpc.so.6                       libgmp.so.10.2.0                  libopkg.so.1
libXdmpc.so.6.0.0                   libgnutls.so.28                   libopkg.so.1.0.0
libavahi-common.so.3                libgnutls.so.28.38.0              libpyglib-2.0-python.so.0
libavahi-common.so.3.5.3            libgobject-2.0.so.0               libpyglib-2.0-python.so.0.0.0
libavahi-core.so.7                  libgobject-2.0.so.0.4000.0        libpython2.7.so.1.0
libavahi-core.so.7.0.2              libgpg-error.so.0                 libreadline.so.6
libbluetooth.so.3                   libgpg-error.so.0.10.0            libreadline.so.6.3
libbluetooth.so.3.13.0              libgthread-2.0.so.0               libssl.so.1.0.0
libdaemon.so.0                      libgthread-2.0.so.0.4000.0        libtirpc.so.1
libdaemon.so.0.5.0                  libhistory.so.6                    libtirpc.so.1.0.10
libdbus-1.so.3                       libhistory.so.6.3                  libxcb.so.1
libdbus-1.so.3.8.4                  libhogweed.so.2                    libxcb.so.1.1.0
libdbus-glib-1.so.2                  libhogweed.so.2.5                  libxml2.so.2
libdbus-glib-1.so.2.2.2              libkmod.so.2                       libxml2.so.2.9.1
libexpat.so.1                       libkmod.so.2.2.8                   locale/
libexpat.so.1.6.0                   libnettle.so.4                     near/
libffi.so.6                         libnettle.so.4.7                   opkg/
libffi.so.6.0.2                     libnl-3.so.200                     python2.7/
libgcrypt.so.20                     libnl-3.so.200.20.0                ssl/
libgcrypt.so.20.0.1                 libnl-cli-3.so.200
libgio-2.0.so.0                     libnl-cli-3.so.200.20.0

```

FIGURE 1.10 – Dossier `/usr/lib`

Exercice 5 : Compilation manuelle du noyau

5.1 Question 5.1

La page 72 de la documentation nous dit qu'il est possible d'allumer et d'éteindre les LEDs via les commandes suivantes :

```

root@DevKit8600:~# echo 1 > /sys/class/leds/user_led/brightness
root@DevKit8600:~# echo 0 > /sys/class/leds/user_led/brightness

```

Comme le dossier

`user_led`

n'existe pas ce n'est pas opérationnel.

5.2 Question 5.2

Nous avons ensuite compiler manuellement le noyau via la chaîne de compilation croisée en exécutant le script suivant :

```

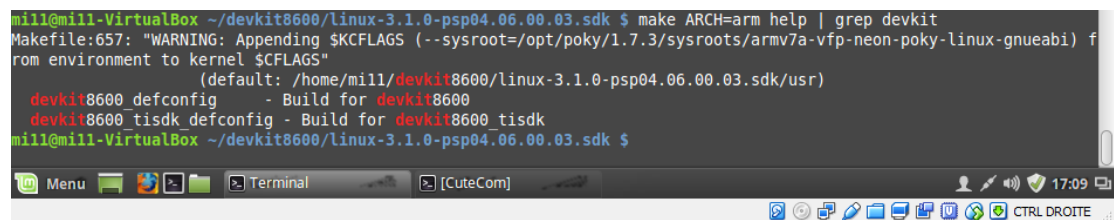
source /opt/poky/1.7.3/environment-setup-armv7a-vfp-neon-poky-linux-gnueabi
unset LDFLAGS

```

Le fichier ci-dessus sert à mettre en place l'environnement de compilation de manière à avoir les bons préfixes pour une compilation croisée. Le préfixe de la chaîne de compilation croisée est *arm-poky-linux-gnueabi-*.

5.3 Question 5.3

Pour obtenir la liste des configurations par défaut du noyau pour une architecture ARM, il faut utiliser la commande `make ARCH=arm help`. Les configurations par défaut possibles pour la carte *devkit8600* sont visibles sur la copie d'écran ci-dessous :



```
mill@mill-VirtualBox ~/devkit8600/linux-3.1.0-psp04.06.00.03.sdk $ make ARCH=arm help | grep devkit
Makefile:657: "WARNING: Appending $KFLAGS (--sysroot=/opt/poky/1.7.3/sysroots/armv7a-vfp-neon-poky-linux-gnueabi) f
rom environment to kernel $CFLAGS"
      (default: /home/mill/devkit8600/linux-3.1.0-psp04.06.00.03.sdk/usr)
devkit8600_defconfig - Build for devkit8600
devkit8600_tisdsk_defconfig - Build for devkit8600_tisdsk
mill@mill-VirtualBox ~/devkit8600/linux-3.1.0-psp04.06.00.03.sdk $
```

FIGURE 1.11 – Configurations par défaut de *devkit8600*

Nous avons retenu uniquement la première configuration et nous l'avons mise en place via la commande suivante :

```
make devkit8600_defconfig
```

5.4 Question 5.4

Ensuite, nous avons lancé la personnalisation du noyau via la commande suivante : `make ARCH=arm menuconfig`. Le but étant d'ajouter un driver pour les LEDs connectées par GPIO. Pour cela, nous avons activé l'option *LED Support for GPIO connected LEDs* comme on peut le voir sur la copie d'écran suivante.

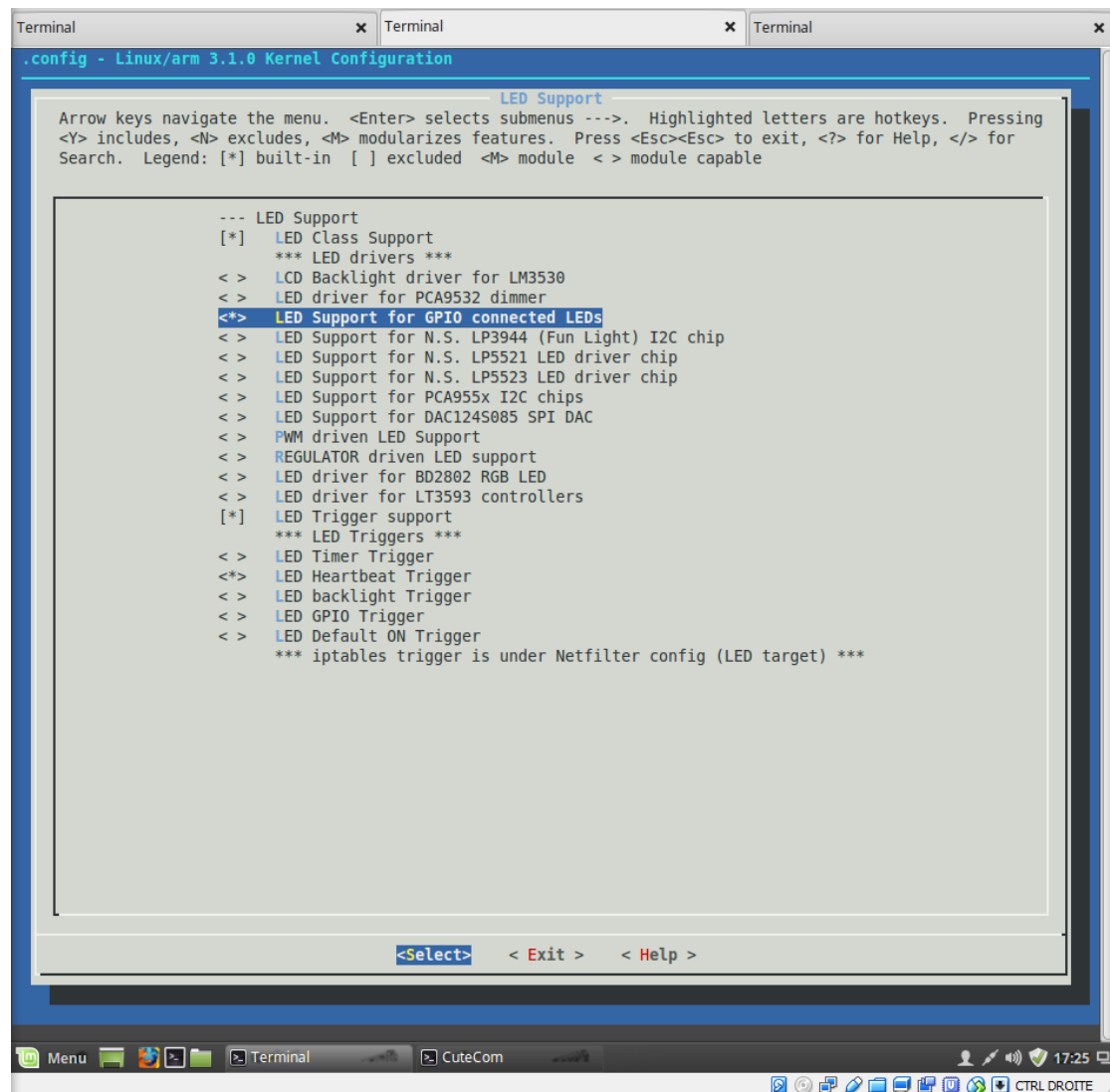


FIGURE 1.12 – Activation du driver pour les LEDs

Ce driver peut être activé via deux modes, *modularizes features* ou *built-in*. Le premier mode est utilisé si l'on souhaite ajouter un driver en tant que module qui sera chargé en mémoire uniquement en cas de besoin et déchargé lorsque le kernel n'en a pas plus besoin. Ceci est utile lorsque l'on souhaite avoir un kernel pas très lourd. Quant au deuxième mode, le driver sera directement intégré au kernel et il sera disponible tout le temps. Le kernel sera donc plus gros, plus lent et utilisera plus de mémoire. Dans notre cas, nous avons utilisé le mode *built-in*.

Nous avons ensuite compiler notre noyau avec la commande suivante : *make ARCH=arm uImage -j2*.

5.5 Question 5.5

Le résultat de la compilation se trouve dans *arch/arm/boot/uImage*.

Nous avons ensuite copier le fichier *uImage* dans */tftpboot* pour qu'il soit utilisé par la cible puis nous l'avons démarré.

5.6 Question 5.6

Pour vérifier dans les logs de démarrage que le noyau utilisé est bien celui qui vient d'être compilé, il suffit de lire la date de compilation :

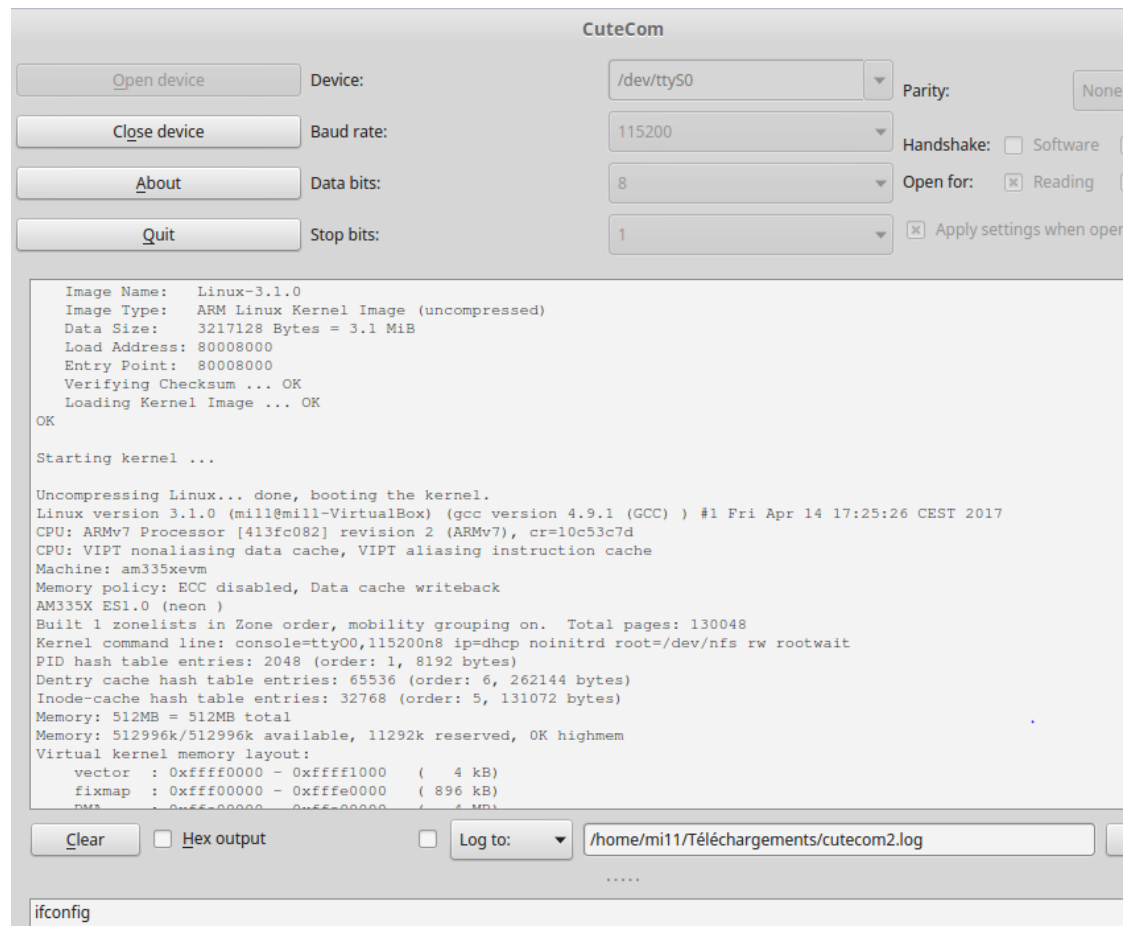


FIGURE 1.13 – Logs de démarrage

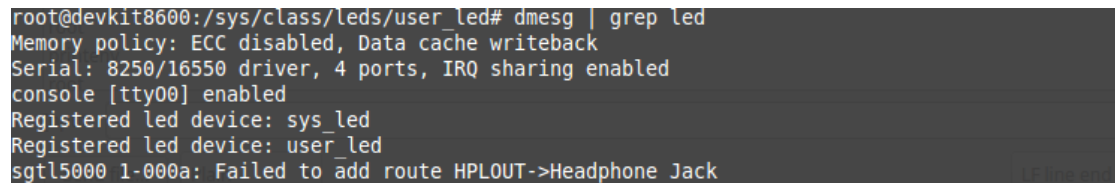
Nous pouvons lire sur la copie d'écran précédente la ligne suivante :

```
Linux version 3.1.0 (mi11@mi11-VirtualBox) (gcc version 4.9.1 (GCC) )
#1 Fri Apr 14 17:25:26 CEST 2017
```

La date de compilation du noyau correspond bien à celui qui vient d'être compilé.

5.7 Question 5.7

Pour vérifier dans les logs de démarrage que la fonctionnalité ajoutée est bien présente, on peut utiliser la commande *dmesg* pour afficher les messages du kernel :



```
root@devkit8600:/sys/class/leds/user_led# dmesg | grep led
Memory policy: ECC disabled, Data cache writeback
Serial: 8250/16550 driver, 4 ports, IRQ sharing enabled
console [tty00] enabled
Registered led device: sys_led
Registered led device: user_led
sgtl5000 1-000a: Failed to add route HPL0UT->Headphone Jack
```

FIGURE 1.14 – Logs du kernel

Le driver a bien été ajouté et le dossier

`used_led`

a été créé.

Messages de sortie du terminal entre l'allumage de la cible et le prompt de login

```

1 CCCCCCCC
  U-Boot SPL 2011.09-svn (May 22 2012 - 11:19:00)
3 Texas Instruments Revision detection unimplemented
  Booting from NAND...
5
7 U-Boot 2011.09-svn (May 22 2012 - 11:19:00)
9 I2C:   ready
  DRAM:  512 MiB
11 WARNING: Caches not enabled
  Did not find a recognized configuration , assuming General purpose EVM
    in Profile 0 with Daughter board
13 NAND:  HW ECC Hamming Code selected
    512 MiB
15 MMC:   OMAP SD/MMC: 0
  Net:    cpsw
17 Hit any key to stop autoboot:  3 \0x08\0x08\0x08 2 \0x08\0x08\0x08 1
    \0x08\0x08\0x08 0
  Card did not respond to voltage select!
19 Booting from network ...
  miiphy read id fail
21 link up on port 0, speed 100, full duplex
  BOOTP broadcast 1
23 DHCP client bound to address 192.168.1.6
  Using cpsw device
25 TFTP from server 192.168.1.1; our IP address is 192.168.1.6
  Filename 'uImage'.
27 Load address: 0x82000000
  Loading: *\0x08
    #####
29 \0x09
    #####
    \0x09
    #####
31 \0x09
    #####

```

```

\0x09
#####
33 \0x09
#####
\0x09
#####
35 \0x09
#####
\0x09
#####
37 \0x09 #####
done
39 Bytes transferred = 3215152 (310f30 hex)
## Booting kernel from Legacy Image at 82000000 ...
41 Image Name: Linux-3.1.0
Image Type: ARM Linux Kernel Image (uncompressed)
43 Data Size: 3215088 Bytes = 3.1 MiB
Load Address: 80008000
45 Entry Point: 80008000
Verifying Checksum ... OK
47 Loading Kernel Image ... OK
OK
49
Starting kernel ...
51
Uncompressing Linux... done, booting the kernel.
53 Linux version 3.1.0 (mi11@mi11-VirtualBox) (gcc version 4.9.1 (GCC) )
#1 Mon Apr 10 18:15:11 CEST 2017
CPU: ARMv7 Processor [413fc082] revision 2 (ARMv7), cr=10c53c7d
55 CPU: VIPT nonaliasing data cache, VIPT aliasing instruction cache
Machine: am335xevm
57 Memory policy: ECC disabled, Data cache writeback
AM335X ES1.0 (neon )
59 Built 1 zonelists in Zone order, mobility grouping on. Total pages:
130048
Kernel command line: console=ttyO0,115200n8 ip=dhcp noinitrd root=/
dev/nfs rw rootwait
61 PID hash table entries: 2048 (order: 1, 8192 bytes)
Dentry cache hash table entries: 65536 (order: 6, 262144 bytes)
63 Inode-cache hash table entries: 32768 (order: 5, 131072 bytes)
Memory: 512MB = 512MB total
65 Memory: 512996k/512996k available, 11292k reserved, 0K highmem
Virtual kernel memory layout:
67 vector : 0xffff0000 - 0xffff1000 ( 4 kB)
fixmap : 0xffff0000 - 0xffffe000 ( 896 kB)
69 DMA : 0xffa00000 - 0xffe00000 ( 4 MB)
vmalloc : 0xe0800000 - 0xf8000000 ( 376 MB)
71 lowmem : 0xc0000000 - 0xe0000000 ( 512 MB)
modules : 0xbf000000 - 0xc0000000 ( 16 MB)

```

```

73      .text : 0xc0008000 - 0xc05c6000    (5880 kB)
      .init : 0xc05c6000 - 0xc05ff000    ( 228 kB)
75      .data : 0xc0600000 - 0xc065e618    ( 378 kB)
      .bss : 0xc065e63c - 0xc0699694    ( 237 kB)
77 NR_IRQS:396
IRQ: Found an INTC at 0xfa200000 (revision 5.0) with 128 interrupts
79 Total of 128 interrupts on 1 active controller
OMAP clockevent source: GPTIMER1 at 25000000 Hz
81 OMAP clocksource: GPTIMER2 at 25000000 Hz
sched_clock: 32 bits at 25MHz, resolution 40ns, wraps every 171798ms
83 Console: colour dummy device 80x30
Calibrating delay loop... 718.02 BogoMIPS (lpj=3590144)
85 pid_max: default: 32768 minimum: 301
Security Framework initialized
87 Mount-cache hash table entries: 512
CPU: Testing write buffer coherency: ok
89 devtmpfs: initialized
print_constraints: dummy:
91 NET: Registered protocol family 16
GPMC revision 6.0
93 OMAP GPIO hardware version 0.1
omap_l3_smx omap_l3_smx.0: couldn't find resource
95 omap_mux_init: Add partition: #1: core, flags: 0
omap_i2c.1: alias fck already exists
97 The board is general purpose EVM in profile 0
omap_hsmmc.0: alias fck already exists
99 omap_hsmmc.2: alias fck already exists
Configure Bluetooth Enable pin...
101 error setting wl12xx data
omap2_mcspi.1: alias fck already exists
103 omap2_mcspi.2: alias fck already exists
bio: create slab <bio-0> at 0
105 SCSI subsystem initialized
usbcore: registered new interface driver usbfs
107 usbcore: registered new interface driver hub
usbcore: registered new device driver usb
109 registerd cppi-dma Intr @ IRQ 17
Cppi41 Init Done Qmgr-base(e083a000) dma-base(e0838000)
111 Cppi41 Init Done
omap_i2c omap_i2c.1: bus 1 rev4.0 at 100 kHz
113 Advanced Linux Sound Architecture Driver Version 1.0.24.
Bluetooth: Core ver 2.16
115 NET: Registered protocol family 31
Bluetooth: HCI device and connection manager initialized
117 Bluetooth: HCI socket layer initialized
Bluetooth: L2CAP socket layer initialized
119 Bluetooth: SCO socket layer initialized
Switching to clocksource gp timer
121 Switched to NOHz mode on CPU #0

```

```

123 musb-hdrc: version 6.0, ?dma?, otg (peripheral+host)
musb-hdrc musb-hdrc.0: dma type: dma-cppi41
musb-hdrc musb-hdrc.0: USB OTG mode controller at e080a000 using DMA,
    IRQ 18
125 musb-hdrc musb-hdrc.1: dma type: dma-cppi41
musb-hdrc musb-hdrc.1: USB OTG mode controller at e080c800 using DMA,
    IRQ 19
127 NET: Registered protocol family 2
IP route cache hash table entries: 4096 (order: 2, 16384 bytes)
129 TCP established hash table entries: 16384 (order: 5, 131072 bytes)
TCP bind hash table entries: 16384 (order: 4, 65536 bytes)
131 TCP: Hash tables configured (established 16384 bind 16384)
TCP reno registered
133 UDP hash table entries: 256 (order: 0, 4096 bytes)
UDP-Lite hash table entries: 256 (order: 0, 4096 bytes)
135 NET: Registered protocol family 1
RPC: Registered named UNIX socket transport module.
137 RPC: Registered udp transport module.
RPC: Registered tcp transport module.
139 RPC: Registered tcp NFSv4.1 backchannel transport module.
NetWinder Floating Point Emulator V0.97 (double precision)
141 VFS: Disk quotas dquot_6.5.2
Dquot-cache hash table entries: 1024 (order 0, 4096 bytes)
143 JFFS2 version 2.2. (NAND) (SUMMARY) \0xc2\0xa9 2001-2006 Red Hat,
    Inc.
msgmni has been set to 1001
145 io scheduler noop registered
io scheduler deadline registered
147 io scheduler cfq registered (default)
Could not set LED4 to fully on
149 da8xx_lcd dc da8xx_lcd.0: GLCD: Found AT043TN24 panel
Console: switching to colour frame buffer device 60x34
151 Serial: 8250/16550 driver, 4 ports, IRQ sharing enabled
omap_uart.0: ttyO0 at MMIO 0x44e09000 (irq = 72) is a OMAP UART0
153 console [ttyO0] enabled
omap_uart.1: ttyO1 at MMIO 0x48022000 (irq = 73) is a OMAP UART1
155 omap_uart.2: ttyO2 at MMIO 0x48024000 (irq = 74) is a OMAP UART2
omap_uart.3: ttyO3 at MMIO 0x481a6000 (irq = 44) is a OMAP UART3
157 omap_uart.4: ttyO4 at MMIO 0x481a8000 (irq = 45) is a OMAP UART4
omap_uart.5: ttyO5 at MMIO 0x481aa000 (irq = 46) is a OMAP UART5
159 brd: module loaded
loop: module loaded
161 i2c-core: driver [tsl2550] using legacy suspend method
i2c-core: driver [tsl2550] using legacy resume method
163 mtdoops: mtd device (mtddev=name/number) must be supplied
omap2-nand driver initializing
165 ONFI flash detected
ONFI param page 0 valid

```

```

167 NAND device: Manufacturer ID: 0xad, Chip ID: 0xdc (Hynix H27U4G8F2DTR
    -BC)
    Creating 8 MID partitions on "omap2-nand.0":
169 0x000000000000-0x000000020000 : "SPL"
    0x000000020000-0x000000040000 : "SPL.backup1"
171 0x000000040000-0x000000060000 : "SPL.backup2"
    0x000000060000-0x000000080000 : "SPL.backup3"
173 0x000000080000-0x0000000260000 : "U-Boot"
    0x0000000260000-0x0000000280000 : "U-Boot Env"
175 0x0000000280000-0x0000000780000 : "Kernel"
    0x0000000780000-0x00000020000000 : "File System"
177 OneNAND driver initializing
    davinci_mdio davinci_mdio.0: davinci mdio revision 1.6
179 davinci_mdio davinci_mdio.0: detected phy mask fffffffe
    davinci_mdio.0: probed
181 davinci_mdio davinci_mdio.0: phy[4]: device 0:04, driver unknown
    CAN device driver interface
183 CAN bus driver for Bosch D_CAN controller 1.0
    d_can d_can: d_can device registered (irq=55, irq_obj=56)
185 usbcore: registered new interface driver cdc_ether
    usbcore: registered new interface driver cdc_subset
187 Initializing USB Mass Storage driver...
    usbcore: registered new interface driver usb-storage
189 USB Mass Storage support registered.
    gadget: using random self ethernet address
191 gadget: using random host ethernet address
    usb0: MAC 06:50:55:8c:0c:93
193 usb0: HOST MAC 62:9a:93:a2:1e:53
    gadget: Ethernet Gadget, version: Memorial Day 2008
195 gadget: g_ether ready
    musb-hdrc musb-hdrc.0: MUSB HDRC host driver
197 musb-hdrc musb-hdrc.0: new USB bus registered, assigned bus number 1
    usb usb1: New USB device found, idVendor=1d6b, idProduct=0002
199 usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
    usb usb1: Product: MUSB HDRC host driver
201 usb usb1: Manufacturer: Linux 3.1.0 musb-hcd
    usb usb1: SerialNumber: musb-hdrc.0
203 hub 1-0:1.0: USB hub found
    hub 1-0:1.0: 1 port detected
205 musb-hdrc musb-hdrc.1: MUSB HDRC host driver
    musb-hdrc musb-hdrc.1: new USB bus registered, assigned bus number 2
207 usb usb2: New USB device found, idVendor=1d6b, idProduct=0002
    usb usb2: New USB device strings: Mfr=3, Product=2, SerialNumber=1
209 usb usb2: Product: MUSB HDRC host driver
    usb usb2: Manufacturer: Linux 3.1.0 musb-hcd
211 usb usb2: SerialNumber: musb-hdrc.1
    hub 2-0:1.0: USB hub found
213 hub 2-0:1.0: 1 port detected
    mousedev: PS/2 mouse device common for all mice

```



```
215 input: ti-tsc-adcc as /devices/platform/tsc/input/input0
omap_rtc omap_rtc: rtc core: registered omap_rtc as rtc0
217 i2c /dev entries driver
Linux video capture interface: v2.00
219 usbcore: registered new interface driver uvcvideo
USB Video Class driver (1.1.1)
221 OMAP Watchdog Timer Rev 0x01: initial timeout 60 sec
Bluetooth: HCI UART driver ver 2.2
223 Bluetooth: HCI H4 protocol initialized
Bluetooth: HCI BCSP protocol initialized
225 Bluetooth: HCILL protocol initialized
Bluetooth: HCIATH3K protocol initialized
227 cpuidle: using governor ladder
cpuidle: using governor menu
229 usbcore: registered new interface driver usbhid
usbhid: USB HID core driver
231 usbcore: registered new interface driver snd-usb-audio
_regulator_get: 1-000a supply VDDA not found, using dummy regulator
233 _regulator_get: 1-000a supply VDDIO not found, using dummy regulator
_regulator_get: 1-000a supply VDDD not found, using dummy regulator
235 sgtl5000 1-000a: sgtl5000 revision 17
print_constraints: 1-000a: 850 <=> 1600 mV at 1200 mV normal
237 _regulator_get: 1-000a supply VDDA not found, using dummy regulator
_regulator_get: 1-000a supply VDDIO not found, using dummy regulator
239 sgtl5000 1-000a: Using internal LDO instead of VDDD
mmcl: card claims to support voltages below the defined range. These
will be ignored.
241 sgtl5000 1-000a: Failed to add route HPLOUT->Headphone Jack
sgtl5000 1-000a: dapm: unknown pin MONO_LOUT
243 sgtl5000 1-000a: dapm: unknown pin HPLCOM
sgtl5000 1-000a: dapm: unknown pin HPRCOM
245 asoc: sgtl5000 <-> davinci-mcasp.0 mapping ok
ALSA device list:
247 #0: AM335X EVM
oprofile: hardware counters not available
249 oprofile: using timer interrupt.
nf_contrack version 0.5.0 (8015 buckets, 32060 max)
251 ip_tables: (C) 2000-2006 Netfilter Core Team
TCP cubic registered
253 NET: Registered protocol family 17
can: controller area network core (rev 20090105 abi 8)
255 NET: Registered protocol family 29
can: raw protocol (rev 20090105)
257 can: broadcast manager protocol (rev 20090105 t)
Bluetooth: RFCOMM TTY layer initialized
259 Bluetooth: RFCOMM socket layer initialized
Bluetooth: RFCOMM ver 1.11
261 Bluetooth: BNEP (Ethernet Emulation) ver 1.3
Bluetooth: BNEP filters: protocol multicast
```

```
263 Bluetooth: HIDP (Human Interface Emulation) ver 1.2
    Registering the dns_resolver key type
265 VFP support v0.3: implementor 41 architecture 3 part 30 variant c rev
    3
    ThumbEE CPU extension supported.
267 _regulator_get: mpu.0 supply mpu not found, using dummy regulator
omap2_set_init_voltage: Fail set voltage-dpll_mpu_ck(f=720000000 v
    =1260000)on vddmpu
269 omap2_set_init_voltage: unable to set vdd_mpu
    Detected MACID=0:18:30:fe:7b:f6
271 input: gpio-keys as /devices/platform/gpio-keys/input/input1
omap_rtc omap_rtc: setting system clock to 2000-01-01 00:00:00 UTC
    (946684800)
273 mmc1: queuing unknown CIS tuple 0x91 (3 bytes)

275 CPSW phy found : id is : 0x4dd072
    PHY 0:01 not found
277 mmc1: new SDIO card at address 0001
    PHY: 0:04 - Link is Up - 100/Full
279 Sending DHCP requests ., OK
    IP-Config: Got DHCP answer from 192.168.1.1, my address is
    192.168.1.6
281 IP-Config: Complete:
    device=eth0, addr=192.168.1.6, mask=255.255.255.0, gw
    =255.255.255.255,
283     host=192.168.1.6, domain=, nis-domain=(none),
    bootserver=192.168.1.1, rootserver=192.168.1.1, rootpath=/
    tftpboot/rootfs
285 VFS: Mounted root (nfs filesystem) on device 0:15.
    devtmpfs: mounted
287 Freeing init memory: 228K

289 INIT: version 2.88 booting

291 Starting udev
    udevd[718]: starting version 182
293 bootlogd: cannot allocate pseudo tty: No such file or directory
    Populating dev cache
295 Fri Apr 14 13:12:58 UTC 2017

297 INIT: Entering runlevel: 5

299 Configuring network interfaces... ifup skipped for nfsroot interface
    eth0
    run-parts: /etc/network/if-pre-up.d/nfsroot exited with code 1
301 Starting system message bus: dbus.
    Starting Dropbear SSH server: Generating key, this may take a while
    ...
303 Public key portion is:
```

```

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDHQW67zEKPRleKx6VZxLER0R/2HThSN
/
SWD3fMk23eXy9j3PudyVMJfjGBF258qnZNicoMsK0Mx5JrpV124XKCzvKTAYsMjLc67WdqcX73zSzt1Cpl
+Dq16Nld2ZuhfGDidLPvrSqOfRaUYRH6048XV/
E7penoQ8oP2tJV0kiGnXRQMoqSyLyZFWW/0xNzatB/Wr6o+
I7Iboc2KWDyOu8caJxP4fxrV/4zEjyTvSQCBCzwuv2RSFPJV7lleMD2XkKN+
QOGCgG1Eq8TgrNDU0Jps+RelENrtkj+lgVJF2odabmPMtmsB+8
lhtgdgk8wth0dbjQzedRFt root@devkit8600
305 Fingerprint: md5 a7:0c:a2:b1:07:9a:77:98:01:7e:31:13:10:02:42:2c
dropbear.
307 Starting rpcbind daemon...rpcbind: cannot create socket for udp6

309 rpcbind: cannot create socket for tcp6

311 done.
Starting syslogd/klogd: done
313 * Starting Avahi mDNS/DNS-SD Daemon: avahi-daemon
...done.
315 Starting Telephony daemon
Starting Linux NFC daemon
317 /etc/rc5.d/S64neard: line 26: /usr/lib/neard/nfc/neard: No such file
or directory

319
321 Poky (Yocto Project Reference Distro) 1.7.3 devkit8600 /dev/ttyO0

323
325 devkit8600 login: root
root@devkit8600:~#

```

Listing A.1 – Messages de sortie du terminal entre l’allumage de la cible et le prompt de login