TP3 – Création d'un device Android

1) Détermination des fichiers sources et des headers pour la compilation d'une libusb sur Android

Les différents fichiers sources permettant la compilation d'une libusb sur Android sont accessibles dans le dossier dont le chemin est le suivant :

TP3/libusb-1.0.3/libusb

On y retrouve entres autres deux fichiers headers : libusb.h et libusbi.h. Les fichiers sources sont quant à eux au nombre de quatre : sync.c, descriptor.c, core.c et io.c. Il existe également un autre dossier, nommé os, contenant des fichiers sources et headers. Ces fichiers sont les suivants : darwin_usb.c, darwin_usb.h, linux_usbfs.c et linux_usbfs.h.

2) Ecriture du fichier Android.mk

Lorsque nous éditons le fichier Android.mk présent dans le dossier libusb-1.0.3, nous découvrons les lignes suivantes :

```
LOCAL_PATH := $(call my-dir)
ubdirs := $(addprefix $(LOCAL_PATH)/,$(addsuffix /Android.mk, \
                libusb \
))
include $(subdirs)
```

La première ligne, "LOCAL_PATH := $(call my-dir)" nous permet de définir la variable LOCAL_PATH et de localiser les fichiers sources, en l'occurrence présent dans le répertoire actuel (d'où le "my-dir"). Il s'agit des fichiers sources cités dans la partie précédente.

La deuxième partie (ubdirs) correspond aux fichiers que nous souhaitons inclure dans notre makefile. "addprefix" et "addsuffix" permettent respectivement d'ajouter des préfixes et des suffixes aux fichiers. Ici, nous allons donc chercher tous les fichiers Android.mk, présents dans le répertoire local mais aussi dans le dossier libusb.

Nous allons désormais écrire le fichier Android.mk présent dans le dossier libusb.

```
# Copyright 2005 The Android Open Source Project
#
# Android.mk for libusb
#

commonSources:= \
        core.c \
        descriptor.c \
        io.c \
        sync.c \
        os/darwin_usb.c \
        os/linux_usbfs.c

# For the host
LOCAL_PATH := $(call my-dir)
include $(CLEAR_VARS)

LOCAL_SRC_FILES := \
        $(commonSources)

LOCAL_C_INCLUDES += $(LOCAL_PATH) $(LOCAL_PATH)/os
LOCAL_MODULES := libusb

include $(BUILD_HOST_STATIC_LIBRARY)
```

```
#For the device
include $(CLEAR_VARS)

LOCAL_SRC_FILES := \
        $(commonSources)

LOCAL_MODULE := libusb
LOCAL_MODULE_TAGS := optional

include $(BUILD_SHARED_LIBRARY)
```

A l'aide de ce fichier, nous allons produire deux librairies : une librairie statique et une librairie partagée. Nous avons ici ajouté à la variable **commonSources** les différents fichiers .c mentionnés dans la première partie. Ils pourront donc être utilisés en tant que fichiers sources locaux par la librairie statique et la librairie partagée à l'aide du tag **LOCAL_SRC_FILES**.

Lors de la compilation, on peut effectivement constater qu'une erreur se produit sur la macro TIMESPEC_TO_TIMEVAL : celle-ci n'est pas définie. Après recherche, on peut constater que la macro TIMESPEC_TO_TIMEVAL est bien utilisée dans le fichier io.c mais n'est pas présente dans le header correspondant (libusbi.h). Pour corriger cette erreur, il nous faudra donc compléter le fichier libusbi.h. On peut définir la macro de la façon suivante :

```
# define TIMESPEC_TO_TIMEVAL(tv, ts)
        do  {
                (tv)->tv_sec = (ts)->tv_sec;
                (tv)->tv_usec = (ts)->tv_nsec / 1000;
        } while (0)
```

La deuxième erreur que nous obtenons est la suivante : **build/tools/apriori/prelinkmap.c(137) : library "libusb.so" not in prelink map.** Pour corriger cette erreur, il est nécessaire d'ajouter deux lignes dans le fichier ./build/core/prelink-linux-arm.map. Ces deux lignes sont les suivantes :

libqcamera.so	**0xA9400000**
libusb.so	**0xA8000000**

3) Implémentation d'un nouveau produit Android

Nous partons dans un premier temps du dossier *device* qui nous est fourni. Considérons que le nom du vendeur est UTBM et le nom de notre produit lo52_Lepy_Sidawy. Nous créons donc l'arborescence suivante:

/device/UTBM/lo52_Lepy_Sidawy

Dans ce dossier, plusieurs fichiers sont nécessaires à la définition d'un nouveau produit :

- Un fichier Android.mk pour définir des instructions de compilation si de nouvelles sources sont présentes.
- Un fichier AndroidProducts.mk pour définir le mk pour la production
- Un fichier CleanSpec.mk pour donner les directives de make clean
- Un fichier BoardConfig.mk pour définir le produit, sans être lié au hardware
- Un fichier vendorsetup.sh
- Un fichier LO52_Lepy_Sidawy.mk pour personnaliser certaines propriétés
- Un dossier overlay qui contiendra les fichiers à surcharger pour ce produit

Dans le dossier overlay nous ajoutons l'image sym_keyboard_delete.png que nous voulons surcharger.

Après une recherche, nous trouvons qu'elle existe à deux endroits, en deux dimensions différentes : aux chemins /samples/SoftKeyboard/res/drawable-[mdpi/hdpi].

Nous recréons donc une arborescence identique à l'intérieur du dossier overlay et nous y plaçons l'image qui surchargera l'image originale.

La rédaction du fichier LO52_Lepy_Sidawy.mk se déroule en plusieurs étapes. Dans un premier temps, nous allons hériter le produit Hikey de Linaro, à l'aide de la directive :

$(call inherit-product, <product>)

Certaines variables telles que PRODUCT_NAME sont définies pour les informations principales du produit. Pour déterminer les propriétés à surcharger, la variable PRODUCT_PROPERTY_OVERRIDES doit être utilisée.

Enfin, on définit également la variable PRODUCT_PACKAGE_OVERLAYS pour indiquer le dossier où nous déposerons des ressources qui remplaceront celles du produit hérité.

Pour ajouter la libusb aux packages du produit, on ajoute à la variable PRODUCT_PACKAGES la libusb.

Le contenu intégral du fichier LO52_Lepy_Sidawy.mk est donc le suivant :

```
$(call inherit-product, device/Linaro/hikey/device-hikey.mk)

PRODUCT_NAME := lo52_Lepy_Sidawy
PRODUCT_DEVICE := lo52
PRODUCT_PROPERTY_OVERRIDES := ro.hw = lo52 net.dns1 = 8.8.8.8 net.dns2 = 4.4.4.4
PRODUCT_PACKAGE_OVERLAYS := device/UTBM/LO52_Lepy_Sidawy/overlay
$(PRODUCT_PACKAGE_OVERLAYS)

PRODUCT_PACKAGES += libusb
```

Ce makefile ainsi que les fichiers relatifs au TP sont disponibles dans les sources du TP sur git.