TP : projet Yocto A3S10

Douillet Thibault, Rousselet Léo, Terrien Marie

1 - Préparation des layers

On cherche à créer deux layers pour le projet avec les priorités fixées, sans les activer, puis les valider avec bitbake-layers. On commence par créer le dépôt Git avec les commandes indiquées.

On charge l'environnement Yocto grâce à la commande source, puis on crée les layers avec la commande create-layer de bitbake-layers. On supprime les dossiers inutiles et on modifie les priorités dans les fichiers conf/layer.conf :

```
jourdes@jourdes-VirtualBox:~$ source w/src/poky/oe-init-build-env w/build-scarthgap/
This is the default build configuration for the Poky reference distribution.

### Shell environment set up for builds. ###

You can now run 'bitbake <target>'

Common targets are:
    core-image-minimal
    core-image-full-cmdline
    core-image-sato
    core-image-weston
    meta-toolchain
    meta-ide-support

You can also run generated qemu images with a command like 'runqemu qemux86-64'.

Other commonly useful commands are:
 - 'devtool' and 'recipetool' handle common recipe tasks
 - 'bitbake-layers' handles common layer tasks
 - 'oe-pkgdata-util' handles common target package tasks
jourdes@jourdes-VirtualBox:~/w/build-scarthgap$ bitbake-layers create-layer ../src/esme-a3s10/meta-esme-images
NOTE: Starting bitbake server...
Add your new layer with 'bitbake-layers add-layer ../src/esme-a3s10/meta-esme-images'
jourdes@jourdes-VirtualBox:~/w/build-scarthgap$ bitbake-layers create-layer ../src/esme-a3s10/meta-esme-custom
NOTE: Starting bitbake server...
Add your new layer with 'bitbake-layers add-layer ../src/esme-a3s10/meta-esme-custom'
```

```
  GNU nano 6.2                                                  conf/layer.conf *
# We have a conf and classes directory, add to BBPATH
BBPATH .= ":${LAYERDIR}"

# We have recipes-* directories, add to BBFILES
BBFILES += "${LAYERDIR}/recipes-*/*/*.bb \
            ${LAYERDIR}/recipes-*/*/*.bbappend"

BBFILE_COLLECTIONS += "meta-esme-images"
BBFILE_PATTERN_meta-esme-images = "^${LAYERDIR}/"
#Priorité changée de 6 à 20
BBFILE_PRIORITY_meta-esme-images = "20"

LAYERDEPENDS_meta-esme-images = "core"
LAYERSERIES_COMPAT_meta-esme-images = "scarthgap"
```

```
  GNU nano 6.2                                  ../meta-esme-custom/conf/layer.conf *
# We have a conf and classes directory, add to BBPATH
BBPATH .= ":${LAYERDIR}"

# We have recipes-* directories, add to BBFILES
BBFILES += "${LAYERDIR}/recipes-*/*/*.bb \
            ${LAYERDIR}/recipes-*/*/*.bbappend"

BBFILE_COLLECTIONS += "meta-esme-custom"
BBFILE_PATTERN_meta-esme-custom = "^${LAYERDIR}/"
#Priorité changée de 6 à 30
BBFILE_PRIORITY_meta-esme-custom = "30"

LAYERDEPENDS_meta-esme-custom = "core"
LAYERSERIES_COMPAT_meta-esme-custom = "scarthgap"
```

On vérifie que les layers ne sont pas activées avec show-layers.

```
jourdes@jourdes-VirtualBox:~/w/src/esme-a3s10/meta-esme-images$ bitbake-layers show-layers
NOTE: Starting bitbake server...
layer                 path                                                                                priority
==========================================================================================================================
core                  /home/jourdes/w/src/poky/meta                                                       5
yocto                 /home/jourdes/w/src/poky/meta-poky                                                  5
yoctobsp              /home/jourdes/w/src/poky/meta-yocto-bsp                                             5
openembedded-layer    /home/jourdes/w/src/meta-openembedded/meta-oe                                       5
meta-python           /home/jourdes/w/src/meta-openembedded/meta-python                                   5
networking-layer      /home/jourdes/w/src/meta-openembedded/meta-networking                               5
filesystems-layer     /home/jourdes/w/src/meta-openembedded/meta-filesystems                              5
raspberrypi           /home/jourdes/w/src/meta-raspberrypi                                                9
meta-esme-a3s         /home/jourdes/w/src/meta-esme-a3s                                                   6
```

L’arborescence est la suivante :

```
jourdes@jourdes-VirtualBox:~/w/src$ tree ~/w/src/esme-a3s10/
/home/jourdes/w/src/esme-a3s10/
├── meta-esme-custom
│   ├── conf
│   │   └── layer.conf
│   ├── COPYING.MIT
│   └── README
├── meta-esme-images
│   ├── conf
│   │   └── layer.conf
│   ├── COPYING.MIT
│   └── README
└── README

4 directories, 7 files
```

2 - Préparation des layers

On cherche à configurer le layer meta-esme-images pour qu'il dépende du layer meta-raspberrypi et à désactiver temporairement meta-raspberrypi et tester si meta-esme-images peut être activé.

On ajoute la dépendance au fichier layer.conf.

```
#Ajout de la dépendance raspberrypi
LAYERDEPENDS_meta-esme-images = "raspberrypi"
```

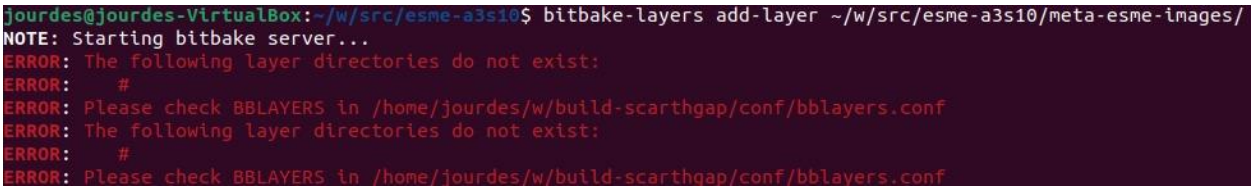
On désactive le layer meta-raspberrypi :

```
  GNU nano 6.2                          /home/jourdes/w/build-scarthgap/conf/bblayers.conf *
# POKY_BBLAYERS_CONF_VERSION is increased each time build/conf/bblayers.conf
# changes incompatibly
POKY_BBLAYERS_CONF_VERSION = "2"

BBPATH = "${TOPDIR}"
BBFILES ?= ""

BBLAYERS ?= " \
  /home/jourdes/w/src/poky/meta \
  /home/jourdes/w/src/poky/meta-poky \
  /home/jourdes/w/src/poky/meta-yocto-bsp \
  /home/jourdes/w/src/meta-openembedded/meta-oe \
  /home/jourdes/w/src/meta-openembedded/meta-python \
  /home/jourdes/w/src/meta-openembedded/meta-networking \
  /home/jourdes/w/src/meta-openembedded/meta-filesystems \
#  /home/jourdes/w/src/meta-raspberrypi \
  /home/jourdes/w/src/meta-esme-a3s \
  "
```

On teste l’activation de meta-esme-images sans meta-raspberrypi, ce qui produit l’erreur suivante :



La dépendance fonctionne donc correctement.

3 – Activation des layers

On réactive les layers avec la commande add-layer :

```
jourdes@jourdes-VirtualBox:~/w/src/esme-a3s10$ bitbake-layers add-layer meta-esme-images/
NOTE: Starting bitbake server...
jourdes@jourdes-VirtualBox:~/w/src/esme-a3s10$ bitbake-layers add-layer meta-esme-custom/
NOTE: Starting bitbake server...
```

On vérifie qu'elles sont bien activées en ouvrant bblayers.conf.

```
BBLAYERS ?= " \
  /home/jourdes/w/src/poky/meta \
  /home/jourdes/w/src/poky/meta-poky \
  /home/jourdes/w/src/poky/meta-yocto-bsp \
  /home/jourdes/w/src/meta-openembedded/meta-oe \
  /home/jourdes/w/src/meta-openembedded/meta-python \
  /home/jourdes/w/src/meta-openembedded/meta-networking \
  /home/jourdes/w/src/meta-openembedded/meta-filesystems \
  /home/jourdes/w/src/meta-raspberrypi \
  /home/jourdes/w/src/meta-esme-a3s \
  /home/jourdes/w/src/esme-a3s10/meta-esme-images \
  /home/jourdes/w/src/esme-a3s10/meta-esme-custom \
  "
```

4 - Liste des layers pour la configuration de build

On liste le contenu de /build/conf :

```
jourdes@jourdes-VirtualBox:~/w/build-scarthgap/conf$ ls
bblayers.conf   conf-summary.txt  local.conf.save  task-depends.dot
conf-notes.txt  local.conf        pn-buildlist     templateconf.cfg
```

On aura besoin de local.conf et de bblayers.conf.

On les copie dans le dossier qui sera rendu, dans le dossier extra_files.

5 – Surcharge de la recipe “rpi-config” (et question 6)

On crée un fichier bbappend pour la recette rpi-config avec la commande recipetool newappend :

```
jourdes@jourdes-VirtualBox:~/w/src/esme-a3s10$ recipetool newappend rpi-config
NOTE: Starting bitbake server...
recipetool newappend: error: argument destlayer: 'rpi-config' must be a path to a valid layer
usage: recipetool newappend [-h] [-e] [-w] destlayer target

arguments:
  destlayer             Base directory of the destination layer to write the bbappend to
  target                Target recipe/provide to append

options:
  -h, --help            show this help message and exit
  -e, --edit            Edit the new append. This obeys $VISUAL if set, otherwise $EDITOR, otherwise vi.
  -w, --wildcard-version
                        Use wildcard to make the bbappend apply to any recipe version
```

```
## Enable UART

RaspberryPi 0, 1, 2 and CM will have UART console enabled by default.

RaspberryPi 0 WiFi and 3 does not have the UART enabled by default because this
needs a fixed core frequency and enable_uart will set it to the minimum. Certain
operations - 60fps h264 decode, high quality deinterlace - which aren't
performed on the ARM may be affected, and we wouldn't want to do that to users
who don't want to use the serial port. Users who want serial console support on
RaspberryPi 0 Wifi or 3 will have to explicitly set in local.conf:

    ENABLE_UART = "1"
```

```
jourdes@jourdes-VirtualBox:~/w/src/esme-a3s10$ recipetool newappend meta-esme-custom rpi-config
NOTE: Starting bitbake server...
Loading cache: 100% |                                                                                | ETA:  --:--:--
Loaded 0 entries from dependency cache.
Parsing recipes: 100% |##########################################################################################| Time: 0:02:19
Parsing of 2778 .bb files complete (0 cached, 2778 parsed). 4686 targets, 438 skipped, 0 masked, 0 errors.
WARNING: No bb files in default matched BBFILE_PATTERN_meta-esme-images '^/home/jourdes/w/src/esme-a3s10/meta-esme-images/'
WARNING: No bb files in default matched BBFILE_PATTERN_meta-esme-custom '^/home/jourdes/w/src/esme-a3s10/meta-esme-custom/'

Summary: There were 2 WARNING messages.
/home/jourdes/w/src/esme-a3s10/meta-esme-custom/recipes-bsp/bootfiles/rpi-config_git.bbappend
```

6) Vérification de la mise à jour de la recipe rpi-config

On peut vérifier que les mises à jours se soient bien éffectuées en vérifiant l’environnement de build avec la commande “bitbake -e”. On peut rediriger la sortie dans un grep afin de trouver les changements spécifiques que l’on souhaite vérifier.

```
  GNU nano 6.2                                                rpi-config_git.bbappend
ENABLE_UART = "1"
```

```
jourdes@jourdes-VirtualBox:~/w/src/esme-a3s10/meta-esme-custom/recipes-bsp/bootfiles$ bitbake -e rpi-config | grep ^ENABLE_UART=
ENABLE_UART="1"
```

7 - Recompiler l’image, la flasher et la démarrer

On peut ensuite recompiler l’image et la flasher sur la carte SD pour tester son fonctionnement sur la carte.

```
jourdes@jourdes-VirtualBox:~/w/build-scarthgap/tmp/deploy/images/raspberrypi0-wifi$ sudo bmaptool copy --bmap rpi-test-image-raspber
rypi0-wifi.rootfs.wic.bmap rpi-test-image-raspberrypi0-wifi.rootfs.wic.bz2 /dev/sdb
[sudo] Mot de passe de jourdes :
bmaptool: info: block map format version 2.0
bmaptool: info: 113357 blocks of size 4096 (442.8 MiB), mapped 52083 blocks (203.4 MiB or 45.9%)
bmaptool: info: copying image 'rpi-test-image-raspberrypi0-wifi.rootfs.wic.bz2' to block device '/dev/sdb' using bmap file 'rpi-test
-image-raspberrypi0-wifi.rootfs.wic.bmap'
bmaptool: info: 100% copied
bmaptool: info: synchronizing '/dev/sdb'
```

Avec picocom, on peut venir tester le fonctionnement de l’image en branchant la carte gràace à un TTL série.

```
jourdes@jourdes-VirtualBox:~$ sudo picocom -b 115200 /dev/ttyUSB0
```

```
Starting bluetooth: bluetoothd.
Starting syslogd/klogd: done
 * Starting Avahi mDNS/DNS-SD Daemon: avahi-daemon
[   17.610173] Bluetooth: BNEP (Ethernet Emulation) ver 1.3
[   17.615708] Bluetooth: BNEP filters: protocol multicast
[   17.638561] Bluetooth: BNEP socket layer initialized
[   17.671826] Bluetooth: MGMT ver 1.22
[   17.707807] NET: Registered PF_ALG protocol family
   ...done.
Starting Telephony daemon
Starting Linux NFC daemon
[   18.203146] nfc: nfc_init: NFC Core ver 0.1
[   18.212520] NET: Registered PF_NFC protocol family
Start
/etc/init.d/rc: /etc/rc5.d/S99esme-led.sh: line 23: can't create : nonexistent d
irectory
LED started with PID
GPIO17 set to 1
[   18.381565] Bluetooth: RFCOMM TTY layer initialized
[   18.386607] Bluetooth: RFCOMM socket layer initialized
[   18.404877] Bluetooth: RFCOMM ver 1.11

Poky (Yocto Project Reference Distro) 5.0.4 raspberrypi0-wifi /dev/ttyS0

raspberrypi0-wifi login: GPIO17 set to 0
```

On peut bien remarquer le démarrage de la Raspberry ci-dessus (la capture a été faite après l’intégration du script GPIO au démarrage d’où la présence de ”GPIO17 set to 0”).

8) Création d'un programme

Pour la création du programme GPIO, on utilise la librairie gpiod.c dont le fonctionnement est très similaire au TP de l'an dernier.

```c
#include <unistd.h>

#define GPIO_CHIP "/dev/gpiochip0"
#define GPIO_LINE 17

int main (){

        struct gpiod_chip *chip;
        struct gpiod_line *line;
        int value = 0;

        chip = gpiod_chip_open(GPIO_CHIP);
        if (!chip) {
                perror("gpiod_chip_open, error");
                return 1;
        }

        line = gpiod_chip_get_line(chip, GPIO_LINE);
        if (!line){
                perror("gpiod_line_chip_open, error");
                gpiod_chip_close(chip);
                return 1;
        }

        if (gpiod_line_request_output(line,"gpio-toggle",0) < 0){
                perror("Output request failed!");
                gpiod_chip_close(chip);
                return 1;
        }

        while (1){
                value = !value;
                gpiod_line_set_value(line,value);
                printf("GPIO%d set to %d\n", GPIO_LINE, value);
                sleep(1);
        }
         gpiod_line_release(line);
         gpiod_chip_close(chip);

         return 0;

}
```

9) Création du Makefile pour compiler le programme

Le makefile suit le modèle de Makefile que nous avions vu en cours.

```
EXE := gpiod
OBJS := gpiod.o
INSTALL_DIR ?= ./.install
SCRIPT := esme-led.sh

CFLAGS += $(shell pkg-config --cflags libgpiod)
LDLIBS += $(shell pkg-config --libs libgpiod)

all: $(EXE)

$(EXE): $(OBJS)

install: $(EXE)
        install -D $(INSTALL_DIR)/usr/bin
        install $(SCRIPT) $(INSTALL_DIR)/etc/init.d
        install -Dm755 $(EXE) $(INSTALL_DIR)/usr/bin/$(SCRIPT)

clean:
        -$(RM) $(OBJS) $(EXE)
```

10) Création du script de démarrage

Le script fonctionne grâce à un case vérifiant le paramètre mis en entrée et réagit en fonction. Avant chaque action, le script vérifie le PID du processus sur lequel il tourne. S'il s'agit de celui du script de la LED, il effectue alors l'action rentrée en paramètre (exécuter, arrêter, ou donner le statut du programme)

```
#!/bin/sh

### BEGIN INIT INFO
#Provides: esme-led
#Required-Start: $remote_fs $time
#Required-Stop: $remote_fs $time
#Default-Start: 3 4 5
#Default-Stop 0 1 2 6
#Short-Description: ESME LED GPIO#17 toggle service
### END INIT INFO

PROGRAM_PATH="/usr/bin/gpio"
PID_PATH="/tmp/esme-led.pid"

case "$1" in
        start)
                echo "Start"
                if [ -f "$PID_PATH" ]
                then
                        echo "LED running"
                else
                        $PROGRAM_PATH &
                        echo $? > "$PID_FILE"
                        echo "LED started with PID"
                fi
        ;;
        stop)
                echo "Stop"
                if [ -f "$PID_PATH" ]
                then
                        kill "$(cat $PID_PATH)"
                        rm "$PID_PATH"
                        echo "LED stopped"
```

```
                else
                        echo "Led not running"
                fi
        ;;
        restart)
                echo "Restart"
                $0 stop
                $0 start
        ;;
        status)
                if [ -f "$PID_PATH" ]
                then
                        echo "LED is running"
                else
                        echo "LED is not running"
                fi
        ;;
        *)
                echo "Usage :"
                echo "start : start script"
                echo "stop : stop script"
                echo "restart : restart script"
                echo "status : get script status"
        ;;
esac

exit 0
```

12) Création d'un workspace

On crée ensuite un workspace devtool pour pouvoir ajouter la recipe "esme-gpio" qui permettra l'installation des programmes gpio-toggle compilés sur l'ilmage de la carte.

```
jourdes@jourdes-VirtualBox:~/w/src/esme-a3s10$ devtool create-workspace workspace
NOTE: Starting bitbake server...
NOTE: Reconnecting to bitbake server...
NOTE: Retrying server connection (#1)... (12:19:58.968419)
NOTE: Reconnecting to bitbake server...
NOTE: Reconnecting to bitbake server...
NOTE: Retrying server connection (#1)... (12:19:58.968419)
NOTE: Retrying server connection (#1)... (12:19:58.968419)
NOTE: Starting bitbake server...
INFO: Enabling workspace layer in bblayers.conf
jourdes@jourdes-VirtualBox:~/w/src/esme-a3s10$ ls
extra_files  meta-esme-custom  meta-esme-images  pn-buildlist  README  task-depends.dot  workspace
jourdes@jourdes-VirtualBox:~/w/src/esme-a3s10$ cd workspace/
jourdes@jourdes-VirtualBox:~/w/src/esme-a3s10/workspace$ ls
conf  README
```

13) DEVTOOL : création d'une recipe

Pour créer la recipe à partir du dossier "gpio-toggle", on exécute la commande suivante :

$devtool add esme-gpio ~/w/src/gpio-toggle

On l'édite ensuite avec "$recipetool edit esme-gpio" et on rajoute "inherit pkgconfig" pour que la recipe hérite des fonctionnalités de cette classe.

```
/home/jourdes/w/src/esme-a3s10/workspace/recipes/esme-gpio/esme-gpio.bb *
# (Feel free to remove these comments when editing.)

# Unable to find any files that looked like license statements. Check the accom>
# documentation and source headers and set LICENSE and LIC_FILES_CHKSUM accordi>
#
# NOTE: LICENSE is being set to "CLOSED" to allow you to at least start buildin>
# this is not accurate with respect to the licensing of the software being buil>
# will not be in most cases) you must specify the correct value before using th>
# recipe for anything other than initial testing/development!
LICENSE = "CLOSED"
LIC_FILES_CHKSUM = ""

inherit pkgconfig

DEPENDS = "libgpiod (<2.0)"
```

```
Sstate summary: Wanted 10 Local 0 Mirrors 0 Missed 10 Current 292 (0% match, 96% complete)#################          | ETA:  0:00:00
Initialising tasks: 100% |#########################################################################################| Time: 0:00:01
NOTE: Executing Tasks
NOTE: esme-gpio: compiling from external source tree /home/jourdes/w/src/gpio-toggle
NOTE: Tasks Summary: Attempted 892 tasks of which 876 didn't need to be rerun and all succeeded.
```

14) Mise à jour de la version préférée de ligpiod. On vérifie la version avec apt-cache show libgpiod-dev

On peut d'abord vérifier la version de la librairie utilisée par notre machine :

```
jourdes@jourdes-VirtualBox:~/w/src/gpio-toggle$ apt-cache show libgpiod-dev
Package: libgpiod-dev
Architecture: amd64
Version: 1.6.3-1build1
```

La version native est bien en 1.6.3, mais la version build était en 3.x, on doit donc la modifier.

Pour cela, on utilise la variable PREFERRED_VERSION_libgpiod comme suit :

```
CONF_VERSION = "2"

BB_NUMBER_THREADS = "2"
PARALLEL_MAKE = "-j 2"
LICENSE_FLAGS_ACCEPTED = "synaptics-killswitch"

DISTRO_FEATURES:remove = "ptest"

PREFERRED_VERSION_libgpiod = "1.6.4"
```

15) Mise à jour de la task do_install de la recipe

On modifie d'abord le Makefile afin d'installer le script dans le dossier /etc/init.d du dossier d'installation, sans oublier d'accorder les permissions correspondantes.

```
EXE := gpiod
OBJS := gpiod.o
INSTALL_DIR ?= ./.install
SCRIPT := esme-led.sh

CFLAGS += $(shell pkg-config --cflags libgpiod)
LDLIBS += $(shell pkg-config --libs libgpiod)

all: $(EXE)

$(EXE): $(OBJS)

install: $(EXE)
	install -d $(INSTALL_DIR)/usr/bin
	install -d $(INSTALL_DIR)/etc/init.d
	install -m 755 $(SCRIPT) $(INSTALL_DIR)/etc/init.d
	install -m 755 $(EXE) $(INSTALL_DIR)/usr/bin/$(EXE)

clean:
	-$(RM) $(OBJS) $(EXE)
```

On modifie ensuite la fonction do_install afin de spécifier le dossier d'installation à rentrer en paramètre lors de la compilation:

```
do_install () {
        # This is a guess; additional arguments may be required
        oe_runmake install INSTALL_DIR=${D}
}
```

On peut vérifier que le fichier s'est installé au bon endroit lors de la compilation :

```
jourdes@jourdes-VirtualBox:~/w/src/gplo-toggle/oe-workdir/image/etc/init.d$ ls
esme-led.sh
```

16) DEVTOOL : déclaration du script à démarrer

```
LICENSE = "CLOSED"
LIC_FILES_CHKSUM = ""

inherit pkgconfig
inherit update-rc.d

DEPENDS += "libgpiod (< 2.0)"

PREFERRED_VERSION_libgpiod = "1.6.4"
```

```
INITSCRIPT_PACKAGES = "${PN}"

INITSCRIPT_NAME = "esme-led.sh"

INITSCRIPT_PARAMS = "start 99 5 2 . stop 20 0 1 6 ."
```

On modifie la recette en ajoutant la classe update-rc.d ainsi que des variables pour associer le script esme-led au script d'initialisation. La variable INITSCRIPT_PARAMS permet de spécifier des niveaux d'initialisation pour le script à exécuter.

17) DEVTOOL : construction de l'image avec la recipe

Après avoir fait devtool build-image rpi-test-image :

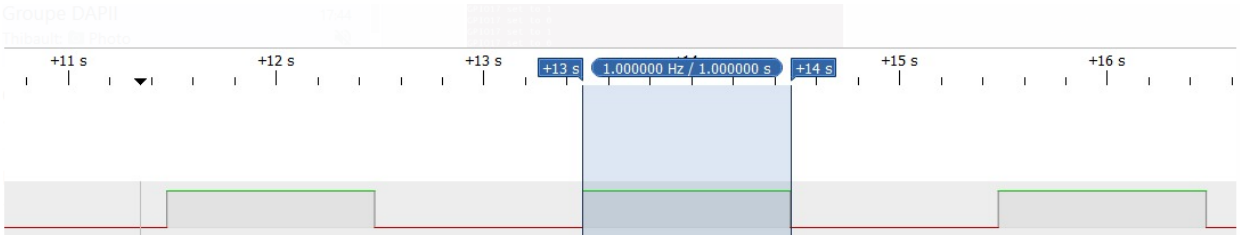
```
INFO: Successfully built rpi-test-image. You can find output files in /home/jourdes/w/build-scarthgap/tmp/deploy/images/raspberrypi0
-wifi
```

Pour flasher la carte SD, on utilise les commandes suivantes :

```
jourdes@jourdes-VirtualBox:~/w/build-scarthgap/tmp/deploy/images/raspberrypi0-wifi$ umount /media/jourdes/root
^[[Ajourdes@jourdes-VirtualBox:~/w/build-scarthgap/tmp/deploy/images/raspberrypi0-wifi$ umount /media/jourdes/boot
jourdes@jourdes-VirtualBox:~/w/build-scarthgap/tmp/deploy/images/raspberrypi0-wifi$ sudo bmaptool copy --bmap rpi-test-image-raspber
rypi0-wifi.rootfs.wic.bmap rpi-test-image-raspberrypi0-wifi.rootfs.wic.bz2 /dev/sdb
```

On peut ensuite tester si la LED clignote bien grâce au débug sur la console ;

```
root@raspberrypi0-wifi:/etc/init.d# ./esme-led.sh start
Start
./esme-led.sh: line 23: can't create : nonexistent directory
LED started with PID
root@raspberrypi0-wifi:/etc/init.d# GPIO17 set to 1
GPIO17 set to 0
GPIO17 set to 1
GPIO17 set to 0
GPIO17 set to 1
GPIO17 set to 0
GPIO17 set to 1
GPIO17 set to 0
GPIO17 set to 1
GPIO17 set to 0
GPIO17 set to 1
GPIO17 set to 0
GPIO17 set to 1
GPIO17 set to 0
GPIO17 set to 1
```



On observe à l'analyseur logique que la GPIO 17 s'allume et s'éteint correctement toute les secondes, comme prévu. On observe cependant de très légère variation dans les périodes d'allumage, qui ne sont jamais exactement d'une seconde. C'est sûrement dû au scheduleur du noyau linux qui ne permet pas d'avoir une fonction sleep constante.

18) DEVTOOL : export de la recipe dans un dépôt distant et mise à jour de l'URI de la recipe

Afin de rendre la recette disponible à tout le monde, on peut la placer sur un dépôt distant public comme sur Github.

```
jourdes@jourdes-VirtualBox:~/w/src/gpio-toggle$ git remote add origin https://github.com/Tdouillet/GPIO-TOGGLE
error: la distante origin existe déjà.
jourdes@jourdes-VirtualBox:~/w/src/gpio-toggle$ git -M branch main
option inconnue : -M
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--super-prefix=<path>] [--config-env=<name>=<envvar>]
           <command> [<args>]
jourdes@jourdes-VirtualBox:~/w/src/gpio-toggle$ git branch -M main
jourdes@jourdes-VirtualBox:~/w/src/gpio-toggle$ git push -u origin main
```

On modifie l'URI de la recipe afin que la recette compile directement à partir du dépot distant au lieu d'un dépot local lors de la prochaine compilation.

```
# No information for SRC_URI yet (only an external source tree was specified)
SRC_URI = "git://github.com/Tdouillet/GPIO-TOGGLE.git;protocol=https;branch=scarthgap"
```

19) DEVTOOL : finalisation de la recipe

Pour publier la recipe dans la layer "meta-esme-custom, il faut utiliser la commande suivante :

```
jourdes@jourdes-VirtualBox:~/w/src/esme-a3s10/workspace$ devtool finish esme-gpi
o ../meta-esme-custom
```