Anexos

A. Guías de Instalación

A.1 Guía de Instalación Debian Embebido

A.2 Guía de Instalación Linaro 15.01

A.3 Guía de instalación Access Point

B. Guías prueba caso de estudio

B.1 Guía con IDE de Arduino

B.2 Guía con terminal de Linux

C. Encuesta

Anexo A

Guías de Instalación

En este anexo se facilitan los procesos llevados a cabo para la instalación de sistemas operativos como de instalación de programas de configuración para la PandaBoard

A.1 Guía de Instalación de Debian Embebido

Si la versión de Linux del pc es de 64 bits, necesita instalar la librería de 32bits para su distribución.

Debian based (deb)	**extra**	**pkgs: (sudo apt-get update ; sudo apt-get install xyz)**
Ubuntu 14.04 -> 15.10		libc6:i386 libstdc++6:i386 libncurses5:i386 zlib1g:i386
Debian 7, 8 & 9 (Stretch)	sudo dpkg --add-architecture i386	libc6:i386 libstdc++6:i386 libncurses5:i386 zlib1g:i386

Descargar/Extraer:

~/

```
wget -c https://releases.linaro.org/14.09/components/toolchain/binaries/gcc-linaro-arm-linux-gnueabihf-4.9-2014.09_linux.tar.xz
tar xf gcc-linaro-arm-linux-gnueabihf-4.9-2014.09_linux.tar.xz
export CC=`pwd`/gcc-linaro-arm-linux-gnueabihf-4.9-2014.09_linux/bin/arm-linux-gnueabihf
```

Test:

Si la prueba falla, verificar que la librería de 32bits esté instalada en tu sistema de desarrollo

~/

```
${CC}gcc --version
arm-linux-gnueabihf-gcc (crosstool-NG linaro-1.13.1-4.9-2014.09 - Linaro GCC 4.9-2014.09) 4.9.2 20140904 (prerelease)
Copyright (C) 2014 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR
```

```
PURPOSE.
```

Bootloader: U-Boot

Descargar:

~/

```
git clone git://git.denx.de/u-boot.git
cd u-boot/
git checkout v2015.10 -b tmp
```

Patches:

~/u-boot

```
wget -c https://rcn-ee.com/repos/git/u-boot-patches/v2015.10/0001-omap4_common-uEnv.txt-bootz-n-fixes.patch

patch -p1 < 0001-omap4_common-uEnv.txt-bootz-n-fixes.patch
```

Configurar y Construir:

~/u-boot

```
make ARCH=arm CROSS_COMPILE=${CC} distclean
make ARCH=arm CROSS_COMPILE=${CC} omap4_panda_defconfig
make ARCH=arm CROSS_COMPILE=${CC}
```

Linux Kernel

Este script construirá el kernel y los módulos y los copiará en el directorio de desarrollo.

Descarga:

~/

```
git clone https://github.com/RobertCNelson/armv7-multiplatform
cd armv7-multiplatform/
```

Para v4.1.x (Longterm):

~/armv7-multiplatform

```
git checkout origin/v4.1.x -b tmp
```

For v4.2.x (Stable):

~/armv7-multiplatform

```
git checkout origin/v4.2.x -b tmp
```

For v4.3.x (Prepatch):

~/armv7-multiplatform

```
git checkout origin/v4.3.x -b tmp
```

Build:

~/armv7-multiplatform

```
./build_kernel.sh
```

Root File System

Debian 8

User	Password
debian	temppwd
root	root

Descargar:

~/

```
wget -c https://rcn-ee.com/rootfs/eewiki/minfs/debian-8.2-minimal-armhf-2015-09-07.tar.xz
```

Verificar:

~/

```
md5sum debian-8.2-minimal-armhf-2015-09-07.tar.xz
406cd5193f4ba6c2694e053961103d1a debian-8.2-minimal-armhf-2015-09-07.tar.xz
```

Extraer:

~/

```
tar xf debian-8.2-minimal-armhf-2015-09-07.tar.xz
```

Setup microSD/SD card

Para esta instrucción, asumimos: DISK=/dev/sdb1 mmcblk0, "lsblk" es muy usado para determinar el id del dispositivo.

```
export DISK=/dev/sdb
```

Borrar microSD/SD card:

```
sudo dd if=/dev/zero of=${DISK} bs=1M count=10
```

Instalar Bootloader:

~/

```
sudo dd if=./u-boot/MLO of=${DISK} count=1 seek=1 bs=128k
sudo dd if=./u-boot/u-boot.img of=${DISK} count=2 seek=1 bs=384k
```

Crear Distribución de la Partición:

Con util-linux v2.26, sfdisk fue reescrito y ahora está basado enlibfdisk.

```
sudo sfdisk --version
```

sfdisk >= 2.26.x

```
sudo sfdisk ${DISK} <<-__EOF__
1M,,L,*
__EOF__
```

sfdisk <= 2.25.x

```
sudo sfdisk --in-order --Linux --unit M ${DISK} <<-__EOF__
1,,L,*
__EOF__
```

Formatear Particiones:

```
 for: DISK=/dev/sdb
sudo mkfs.ext4 ${DISK}1 -L rootfs
```

Montar Particiones:

En algunos sistemas, estas particiones pueden ser automontadas

```
sudo mkdir -p /media/rootfs/

for: DISK=/dev/sdb
```

```
sudo mount ${DISK}1 /media/rootfs/
```

Instalar Kernel y Root File System

Script Complete
eewiki.net: [user@localhost:~$ export kernel_version=4.X.Y-Z]

Copiar y pegar el "export kernel_version=4.X.Y-Z" exactamente como se muestra en el entorno de construcción propio build/desktop y con enter creamos la variable de entorno para ser usado más tarde.

```
export kernel_version=4.X.Y-Z
```

Copiar Root File System

~/
```
sudo tar xfvp ./*-*-*-armhf-*/armhf-rootfs-*.tar -C /media/rootfs/
```

Crear /boot/uEnv.txt

~/
```
sudo sh -c "echo 'uname_r=${kernel_version}' >> /media/rootfs/boot/uEnv.txt"
```

Device Tree Binary

```
PandaBoard ES
sudo sh -c "echo 'dtb=omap4-panda-es.dtb' >> /media/rootfs/boot/uEnv.txt"
```

Copiar archivos de Kernel

Imagen del Kernel:

~/
```
sudo cp -v ./armv7-multiplatform/deploy/${kernel_version}.zImage /media/rootfs/boot/vmlinuz-${kernel_version}
```

Copiar Kernel Device Tree Binaries:

~/
```
sudo mkdir -p /media/rootfs/boot/dtbs/${kernel_version}/
sudo tar xfv ./armv7-multiplatform/deploy/${kernel_version}-dtbs.tar.gz -C /media/rootfs/boot/dtbs/${kernel_version}/
```

Copiar Modules del Kernel:

~/

```
sudo tar xfv ./armv7-multiplatform/deploy/${kernel_version}-modules.tar.gz -C /media/rootfs/
```

File Systems Table (/etc/fstab)

/etc/fstab

```
sudo sh -c "echo '/dev/mmcblk0p1 / auto errors=remount-ro 0 1' >> /media/rootfs/etc/fstab"
```

Networking

Editar: /etc/network/interfaces

```
sudo nano /media/rootfs/etc/network/interfaces
```

Adicionar:

/etc/network/interfaces

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp
```

WiFI

Instalar Firmware:

~/

```
git clone git://git.kernel.org/pub/scm/linux/kernel/git/firmware/linux-firmware.git --depth=1

sudo mkdir -p /media/rootfs/lib/firmware/ti-connectivity
sudo cp -v ./linux-firmware/ti-connectivity/* /media/rootfs/lib/firmware/ti-connectivity
```

Setup WiFi

```
sudo nano /media/rootfs/etc/network/interfaces
```

/etc/network/interfaces

```
auto wlan0
iface wlan0 inet dhcp
    wpa-ssid "essid"
    wpa-psk  "password"
```

Remover microSD/SD card:

```
Sync
sudo umount /media/rootfs
```

A.2 Guía de instalación Linaro 15.01

Se definen los siguientes parámetros en la consola con la SD insertada en el computador. Previamente se corre el comando 'lsblk' con el fin de conocer la identificación de la tarjeta de memoria en el sistema (Para este caso sdb)

- SDCARD=/dev/sdb
- URL=http://releases.linaro.org/15.01/ubuntu/panda/panda-utopic_developer_20150117-707.img.gz
- curl $URL | gunzip | sudo dd bs=64k of=$SDCARD

Según la capacidad de la red y el uso de los servidores de linux, esta tarea puede tardar de 5 a 70 minutos. Cabe mencionar que al finalizar el proceso, esta instalación no hace uso de toda la capacidad de almacenamiento de la tarjeta de memoria, por lo que para evitar posteriores problemas de almacenamiento se procede a ampliar el tamaño de la partición con Gparted.

A.3 Guía de Instalación Access Point

Guía para convertir la pandaboard en un punto de acceso inalámbrico (Access Point)

Los comandos usados de ejemplo son para Ubuntu 14.04. En nuestro caso, en la Pandaboard está instalado Debian 7.7. Se deberá tener acceso a internet para descargar hostapd y dnsmasq (dnsmasq es un pequeño servidor DNS/DHCP que se utilizar en la configuración). Para iniciar se debe ejecutar en consola el siguiente comando:

```
sudo apt-get install hostapd dnsmasq
```

Después necesitas crear y editar el archivo de configuración

```
zcat /usr/share/doc/hostapd/examples/hostapd.conf.gz | sudo tee -a /etc/hostapd/hostapd.conf
```

Se procede con la configuración del archivo ***/etc/hostapd/hostapd.conf*** Se utilizan como ejemplo el SSID '**Example-WLAN**' y la contraseña ***'PASS'***, tal y como aparecen en el ejemplo siguiente:

```
interface=wlan0
ssid=Example-WLAN
hw_mode=g
wpa=2
wpa_passphrase=PASS
wpa_key_mgmt=WPA-PSK WPA-EAP WPA-PSK-SHA256 WPA-EAP-SHA256
```

Además se debe editar el siguiente archivo para configurar HostAP. Agregar la línea DAEMON_CONF="/etc/hostapd/hostapd.conf":

```
# /etc/default/hostapd
# Defaults for hostapd initscript
#
# See /usr/share/doc/hostapd/README.Debian for information about alternative
# methods of managing hostapd.
#
# Uncomment and set DAEMON_CONF to the absolute path of a hostapd configuration
# file and hostapd will be started during system boot. An example configuration
# file can be found at /usr/share/doc/hostapd/examples/hostapd.conf.gz
#
#DAEMON_CONF=""

DAEMON_CONF="/etc/hostapd/hostapd.conf"

# Additional daemon options to be appended to hostapd command:-
#       -d   show more debug messages (-dd for even more)
#       -K   include key data in debug messages
#       -t   include timestamps in some debug messages
#
# Note that -B (daemon mode) and -P (pidfile) options are automatically
# configured by the init.d script and must not be added to DAEMON_OPTS.
#
```

#DAEMON_OPTS=""

Luego, se debe configurar el archivo ***/etc/network/interfaces***, el archivo debe quedar similar al que se muestra en el ejemplo siguiente:

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback


auto wlan0
iface wlan0 inet static
hostapd /etc/hostapd/hostapd.conf
address 192.168.8.1
netmask 255.255.255.0
```

Ahora se debe configurar el DNS y el servidor DHCP, esto se hace dentro del archivo ***/etc/dnsmasq.conf*** que debe mostrarse o quedar de la siguiente manera:

```
interface=lo,wlan0
no-dhcp-interface=lo
dhcp-range=192.168.8.20,192.168.8.254,255.255.255.0,12h
```

Lo siguiente que se necesita para asegurarse que el kernel de Linux reenvía el tráfico de nuestra red inalámbrica a las redes de destino, es configurar el archivo ***/etc/sysctl.conf*** y asegurarse que contenga la siguiente línea:

```
net.ipv4.ip_forward=1
```

Se hace necesario activar NAT, que es un traductor de direcciones de red, esto se hace en el archivo ***/etc/rc.local*** en el cual se debe activar la línea que aparece a continuación:

```
iptables -t nat -A POSTROUTING -s 192.168.8.0/24 ! -d 192.168.8.0/24  -j MASQU
ERADE
```

Es necesario en algunos hardware la activación del interruptor on/off virtual, si se tiene instalado ***rfkill*** se debe correr con el siguiente comando ***rfkill unblock 0,*** sino es necesario hacer la descarga correspondiente y luego se ejecuta.

También es necesario ejecutar el administrador de red, y para ello se hace necesario tener el archivo ***/etc/NetworkManager/NetworkManager.conf,*** este debe quedar o verse de la siguiente manera:

```
[main]
plugins=ifupdown,keyfile,ofono
dns=dnsmasq

[ifupdown]
managed=false
```

Y al final, para que todos los cambios surtan efecto, se debe reiniciar el equipo.

Si todo funciona bien, la nueva red WLAN debe ser detectada, y el servidor WLAN verá una salida similar a partir del uso se los siguientes comandos:

```
$ iw wlan0 info
Interface wlan0
        ifindex 3
        type AP
        wiphy 0

$ iwconfig
wlan0     IEEE 802.11bgn  Mode:Master  Tx-Power=20 dBm
          Retry  long limit:7   RTS thr:off   Fragment thr:off
          Power Management:off

$ ifconfig
wlan0     Link encap:Ethernet  HWaddr f4:ec:38:de:c8:d2
          inet addr:192.168.8.1  Bcast:192.168.8.255  Mask:255.255.255.0
          inet6 addr: fe80::f6ec:38ff:fede:c8d2/64 Scope:Link
```

```
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:5463040 errors:0 dropped:0 overruns:0 frame:0
TX packets:8166528 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:861148382 (861.1 MB)  TX bytes:9489973056 (9.4 GB)
```

Anexo B

Guías prueba caso de estudio

En este anexo se encuentran las guías para la prueba definida para el caso de estudio. Se debe tener en cuenta que la tarjeta debe estar conectada al computador con el cable usb-serial, pero sin la alimentación a la fuente hasta que se indique.

B.1 Guía con IDE de Arduino

1. Abrir un terminal de linux y acceder como super usuario (password:guia1234)

2. Ir a la carpeta dev con el comando cd /dev

3. Una vez en la carpeta dev se corre el comando ls para verificar que se ha reconocido la tarjeta vía puerto serial. Debe aparecer el nombre ttyUSB0 en la lista.

4. Se ejecuta el comando chmod 777 ttyUSB0 para conceder permisos de escritura, lectura y ejecución.

5. Se escribe el comando cu -l ttyUSB0 -s 115200 para acceder a la tarjeta a través de puerto serial. Se conecta la tarjeta a la alimentación, se espera hasta que el sistema inicie y se introduce el login (debian) y el password (temppwd).

6. Abrir IDE de Arduino e ir a Tools->Board y seleccionar Pandaboard ES

7. Abrir el ejemplo para servidor UDP y modificarlo para que la tarjeta reciba datos desde una red de sensores vía WiFi y posteriormente los envíe hacia otro servidor vía Ethernet.

8. A continuación presionar el botón de verificar/compilar o ctrl+r para revisar si el código presenta errores y así resolverlos.

9. Una vez esté listo se procede a presionar el botón de upload, el cual realiza la acción de enviar el ejecutable a la tarjeta.

B.2 Guía con terminal de Linux

1. Abrir un terminal de linux y acceder como super usuario (password:guia1234)

2. Ir a la carpeta dev con el comando cd /dev

3. Una vez en la carpeta dev se corre el comando ls para verificar que se ha reconocido la tarjeta vía puerto serial. Debe aparecer el nombre ttyUSB0 en la lista.

4. Se ejecuta el comando chmod 777 ttyUSB0 para conceder permisos de escritura, lectura y ejecución.

5. Se escribe el comando cu -l ttyUSB0 -s 115200 para acceder a la tarjeta a través de puerto serial. Se conecta la tarjeta a la alimentación, se espera hasta que el sistema inicie y se introduce el login (debian) y el password (temppwd).

6. En el pc se abre el editor de texto Sublime y se desarrolla un código que permita configurar la PandaBoard como servidor UDP para que la tarjeta reciba datos desde una red de sensores vía WiFi y posteriormente los envíe hacia otro servidor vía Ethernet.

7. En otro terminal se va hasta la ruta donde se encuentre el archivo que contiene el código desarrollado en el paso anterior con el comando cd ruta/del/archivo.

8. Se ejecuta el archivo makefile con el comando make. Este archivo debe estar en el mismo directorio donde esté guardado el código con un nombre genérico (test.elf) que ya se encuentra definido dentro del makefile. Si salen errores se revisa el código y se ejecuta nuevamente el makefile hasta que todos los problemas sean resueltos y se haya generado el ejecutable.

9. Se corre el comando ls para verificar la creación del ejecutable (test.elf).

10. Se envía el código a la tarjeta con el comando scp -P 22 test.elf debian@dirección-ip-de-la-tarjeta:~

Anexo C

Encuesta

El siguiente Anexo es una encuesta desarrollada para hacer una prueba cualitativa del caso de estudio

C.1 Encuesta prueba dos para caso de Estudio

En la siguiente encuesta se definirá como primer método el que hace uso del IDE de Arduino con soporte para PandaBoard y como segundo método el que hace uso de la serie de herramientas disponibles para programar y compilar archivos para dicha tarjeta.

1. ¿Le pareció intuitivo programar con el primer método en comparación con el segundo? SI_ NO_

2. ¿Cuál método considera que es más rápido en cuanto a tiempos de desarrollo? Metodo 1_ Metodo 2_

3. ¿El uso de una interfaz gráfica le hace más cómoda la tarea de programar? SI_ NO_

4. ¿Considera que el segundo método hace uso de demasiadas herramientas software para el proceso de desarrollo? SI_ NO_

5. ¿Considera que el primer método podría mejorar o complementar la experiencia de desarrollo en la PandaBoard? SI_ NO_

6. ¿Considera que el primer método motivaría a las personas a utilizar la PandaBoard en sus proyectos? SI_ NO_