

Para quemar una imagen oficial personalizada de OpenWrt en una MicroSD y que esta imagen inicie correctamente en una BPI-R3 tenemos que seguir los siguientes pasos:

1. Obtención de imagen para quemar en la MicroSD.
2. Grabado de la imagen en la tarjeta.
3. Configuración de los jumpers para iniciar desde la MicroSD.
4. Inicio de la MicroSD.

OBTENCIÓN DE LA IMAGEN PERSONALIZADA DE OPENWRT

La BananaPi BPI-R3 tiene un procesador MediaTek Filogic MT7986a. OpenWrt ofrece una herramienta web para personalizar la creación de la imagen de instalación. Con esta herramienta web podemos compilar una imagen personalizada, con los paquetes que nos interesa. Lo bueno de esto es que, elijamos los paquetes que elijamos, la propia herramienta de compilación seleccionará automáticamente todas las dependencias.

Los enlaces a la herramienta web para las diferentes versiones son:

Snapshot
24.10.4

En mi caso, personalizo los paquetes desplegando la flecha hacia abajo, borro todos los que aparecen y los reemplazo con esta lista personalizada:

```
# U-Boot

uboot-envtools

# Hardware

# Chipset

mt7986-wo-firmware
kmod-mt7986-firmware

# Ventilador

kmod-hwmon-pwmfan

# Wifi

kmod-mt7915-firmware
kmod-mt7915e
kmod-mac80211
kmod-mt7615e
# SFP

kmod-sfp

# USB 2

kmod-usb2
kmod-usb-core
kmod-usb-ehci
kmod-usb-ohci
usbutils
usbids

# USB 3

kmod-usb3
kmod-usb-xhci-hcd
```

```
kmod-usb-xhci-mtk

# GPIO

kmod-gpio-button-hotplug
kmod-i2c-gpio
kmod-leds-gpio

# MTD

mtd

# NVMe

kmod-nvme

# PCI

pciutils
pciids

# Software

# Herramientas para terminal (mandatorias para el funcionamiento del sistema)

base-files
busybox
dnsmasq
dropbear
wpad-basic-mbedtls (mejor hostapd-openssl en x86 porque es más completo)
libc
libgcc
libstream-mbedtls (mejor libstream-openssl en x86 porque es más completo)
logd
netifd
nftables
openssh-sftp-server
opkg
procd
procd-ujail
procd-seccomp
uci

# Herramientas para terminal (extra)

mc
nano
curl
git
hwclock
ethtool

# Acceso a volúmenes

e2fsprogs
f2fsck
fstools
mkf2fs
blkid
block-mount
blockd
blockdev
dosfstools
fdisk
kmod-fs-vfat
kmod-usb-storage
parted
nand-utils
lsblk
```

```
# Web

luci
luci-i18n-base-es

# Adblock

adblock
luci-app-adblock
luci-i18n-adblock-es
tcpdump
msmtp

# DDNS

ddns-scripts
ddns-scripts-services
luci-app-ddns
luci-i18n-ddns-es
bind-host

# Cortafuegos

firewall4
luci-app-firewall
luci-i18n-firewall-es

# OPKG

opkg
luci-app-package-manager
luci-i18n-package-manager-es

# Programación Wifi

wifischedule
luci-app-wifischedule
luci-i18n-wifischedule-es

# Samba

luci-app-samba4
luci-i18n-samba4-es

# Terminal en LUCI

luci-app-ttyd
luci-i18n-ttyd-es

# uPnP

luci-app-upnp
luci-i18n-upnp-es

# Wake on LAN

luci-app-wol (Instala la dependencia etherwake)
luci-i18n-wol-es

# Watchcat

luci-app-watchcat
luci-i18n-watchcat-es

# Ejecutar scripts o comandos desde LUCI (Agrega Sistema >> Comandos personalizados)

luci-app-commands
luci-i18n-commands-es
```

```
# VPN

kmod-wireguard
wireguard-tools
luci-proto-wireguard
qrencode

# Módem

luci-proto-modemmanager

# MÓDEM EC25 (Modo QMI)

.
kmod-usb-serial
kmod-usb-serial-wwan # Kernel support for USB GSM and CDMA modems
kmod-usb-serial-option
.
kmod-mii
kmod-usb-net
kmod-usb-wdm
kmod-usb-net-qmi-wwan # QMI WWAN driver for Qualcomm MSM based 3G and LTE modems
wwan
uqmi
.
usb-modeswitch
minicom
```

Aquí abajo, todos los paquetes en una única línea:

```
uboot-envtools mt7986-wo-firmware kmod-mt7986-firmware kmod-hwmon-pwmfan kmod-mt7915-firmware kmod-mt7915e kmod-
mac80211 kmod-mt7615e kmod-sfp kmod-usb2 kmod-usb-core kmod-usb-ehci kmod-usb-ohci usbutils usbids kmod-usb3 kmod-
usb-xhci kmod-usb-xhci-mtk kmod-gpio-button-hotplug kmod-i2c-gpio kmod-leds-gpio mtd kmod-nvme pciutils pciids
base-files busybox dnsmasq dropbear wpad-basic-mbedtls libc libgcc libstream-mbedtls logd netifd nftables openssh-
sftp-server opkg procd procd-ujail procd-seccomp uci mc nano curl git hwclock ethtool e2fsprogs f2fsck fstools
mkf2fs blkid block-mount blockd blockdev dosfstools fdisk kmod-fs-vfat kmod-usb-storage parted nand-utils lsblk
luci luci-i18n-base-es adblock luci-app-adblock luci-i18n-adblock-es tcpdump msmtp ddns-scripts ddns-scripts-
services luci-app-ddns luci-i18n-ddns-es bind-host firewall4 luci-app-firewall luci-i18n-firewall-es opkg luci-app-
package-manager luci-i18n-package-manager-es wifischedule luci-app-wifischedule luci-i18n-wifischedule-es luci-app-
samba4 luci-i18n-samba4-es luci-app-ttyd luci-i18n-ttyd-es luci-app-upnp luci-i18n-upnp-es luci-app-wol luci-i18n-
wol-es luci-app-watchcat luci-i18n-watchcat-es luci-app-commands luci-i18n-commands-es kmod-wireguard wireguard-
tools luci-proto-wireguard qrencode luci-proto-modemmanager kmod-usb-serial kmod-usb-serial-wwan kmod-usb-serial-
option kmod-mii kmod-usb-net kmod-usb-wdm kmod-usb-net-qmi-wwan wwan uqmi usb-modeswitch minicom
```

Todos los paquetes de arriba garantizan que la BPI-R3 funcione correctamente, con todas sus funcionalidades activadas. La propia herramienta selecciona todas las dependencias para esa compilación y se asegura que no falte ninguno que tenga relación con los paquetes que le pegamos. Esto nos asegura una instalación funcional de OpenWrt en la BPI-R3 en la que podremos utilizar todos sus componentes y hacer todas las funciones básicas de un router (y alguna más).

De los paquetes oficiales que vienen cuando uno instala la compilación por defecto, he dejado fuera los siguientes:

```
ca-bundle # Conjunto de certificados raíz para que el sistema confíe en conexiones HTTPS/TLS. Sin
esto, HTTPS falla.
fitblk # Soporte para imágenes FIT con particiones en bloque. Se usa en firmwares modernos para
manejar el arranque y actualizaciones desde imágenes FIT almacenadas en flash o eMMC.
kmod-crypto-hw-safexcel # Descarga operaciones crypto al hardware mejorando el rendimiento de VPNs, IPsec, TLS,
etc. si el SoC lo soporta.
kmod-nft-offload #
Offload de NAT/conntrack al hardware usando nftables. Reduce carga de CPU y aumenta throughput en routing/NAT. Útil
en routers con soporte de flow offload.
kmod-phy-aquantia # Driver para PHYs Aquantia (10G/5G/2.5G/1G). Necesario si el hardware usa chips Aquantia
para Ethernet multigigabit.
odhcp6c #
Cliente DHCPv6. Se usa para obtener IPv6 desde el ISP (prefijo, DNS, etc.) en la interfaz WAN.
odhcpd-ipv6only # Asigna direcciones IPv6 (sin IPv4) y anuncios de router a la LAN. Normalmente va junto
```

```

con odhcp6c.
ppp                # Soporte base de PPP (Point-to-point Protocol). Capa común para conexiones PPP (PPP over
serial, PPPoE, etc.).
ppp-mod-pppoe      # Módulo PPPoE. Permite conectar a ISPs que usan PPPoE (muy común en fibra y ADSL).
procd-ujail        # Soporte de ujaíl para procd. Permite ejecutar servicios en jaulas ligeras (sandboxing)
con namespaces y restricciones. Mejora seguridad del sistema.
uclient-fetch      # Cliente HTTP/HTTPS ligero. Lo usa OpenWrt para descargar archivos (actualizaciones,
scripts, feeds) sin depender de wget/curl completos.
urandom-seed       # Guarda y restaura una semilla de entropía entre reinicios. Evita que el sistema arranque
con entropía baja, lo cual es crítico para crypto.
urngd              # Daemon que alimenta /dev/random usando /dev/random. Mantiene un nivel saludable de
entropía, especialmente en sistemas sin buen RNG hardware.

```

Al quemar una imagen personalizada, debemos tener en cuenta que, además de preparar la imagen para la MicroSD, si vamos a querer tener instalado OpenWrt en la SPI NAND de la BPI-R3, deberemos tener cuidado con que la imagen generada no ocupe más de 128MB, pues es el tamaño máximo que disponemos para quemar imágenes en ella. Si, por otro lado, queremos tener OpenWrt instalado en la EMMC, o si vamos a correrlo desde la propia MicroSD, no nos hace falta preocuparnos por el espacio que ocupe la imagen generada; la EMMC de la BPI-R3 tiene 8GB de espacio y una MicroSD, pues lo que estemos dispuestos a gastar.

Después de ingresar estos paquetes en el personalizador, y antes de compilar, podemos crear un script que se ejecutará la primera vez que la imagen de OpenWrt se inicie en la BPI-R3. Esto lo hacemos en el campo de texto que está por debajo del campo de selección de paquetes (uci-defaults). Un script básico podría ser:

```

.
```

Después de darle a compilar, y de que la compilación haya resultado exitosa, podremos descargar 8 tipos de archivos individuales:

openwrt-xxx-mediatek-filogic-bananapi_bpi-r3-sdcard.img: Imagen para quemar directamente en la MicroSD.

openwrt-xxx-mediatek-filogic-bananapi_bpi-r3-nor-preloader.bin: Esta va en la mtd bl2?

openwrt-xxx-mediatek-filogic-bananapi_bpi-r3-nor-bl31-uboot.fip: Es la imagen de cargador de arranque. Va en la mtd fip?

openwrt-xxx-mediatek-filogic-bananapi_bpi-r3-snand-preloader.bin: Esta va en la mtd bl2?

openwrt-xxx-mediatek-filogic-bananapi_bpi-r3-snand-bl31-uboot.fip: Es la imagen de cargador de arranque. Va en la mtd fip?

openwrt-xxx-mediatek-filogic-bananapi_bpi-r3-emmc-preloader.bin: Es la correspondiente a bl2_emmc.

openwrt-xxx-mediatek-filogic-bananapi_bpi-r3-emmc-bl31-uboot.fip: Imagen de cargador de arranque. Software de bajo nivel que carga el sistema operativo en la primera fase del arranque.

openwrt-xxx-mediatek-filogic-bananapi_bpi-r3-squashfs-sysupgrade.itb: Para actualizar un sistema que ya ejecuta OpenWrt.

El archivo que nos importa ahora mismo es el de la SD. Descomprimos el archivo y obtendremos otro archivo con extensión .img.

QUEMADO DE LA IMAGEN EN LA TARJETA

Podemos quemar la imagen en la MicroSD de esta forma:

```
sudo dd if=/home/usuario/openwrt-xx.xx.x-mediatek-filogic-bananapi_bpi-r3-sdcard.img of=/dev/mmcblk0 bs=10M
status=progress oflag=direct
```

Eso sí, al acabar de quemar la imagen de la MMC en la tarjeta, hay que prestar atención a que la última partición (que es donde se montará el overlay) tiene de tamaño, como mucho, el total de espacio que suman los paquetes elegidos. Y eso es muy poco. Por eso, antes de sacarla del ordenador GNU/Linux donde la hemos «quemado» y antes también de meterla en la BPI-R3, hay que hacer que esa última partición (que es la que se llama «production») sea tan grande como todo el espacio libre disponible al final de la tarjeta de memoria. Esto lo hacemos abriendo en la distro donde la quemamos, una consola con permisos de administrador y ejecutando dentro:

```
sudo cfdisk /dev/mmcblk0
```

...donde /dev/mmcblk0 es el dispositivo de la tarjeta de memoria.

Nos desplazamos con flecha abajo hasta la última partición y luego, con flecha lateral nos desplazamos hasta la opción de resize.

Damos Enter en resize y aceptamos que use todo el espacio libre disponible.

Nos desplazamos hasta write y damos Enter, aceptando la escritura.

Salimos de cfdisk presionando Enter sobre «Quit».

El binario **cfdisk**, en caso de que no lo tengamos instalado, forma parte del paquete util-linux.

CONFIGURACIÓN DE JUMPERS PARA INICIAR DESDE LA MICROSD

Expulsamos la MicroSD, la metemos en el router BPI-R3, ponemos los jumpers 1, 2 y 4 hacia arriba. La posición del jumper 4 determinará si el OpenWrt booteado desde la MicroSD podrá acceder a la SPI NOR o a la SPI NAND. Para la NAND, arriba. Para la NOR, abajo.

INICIO DE LA MICROSD

Conectamos el cable de corriente y el OpenWrt debería iniciar sin problemas.

