

AirPlus DWL-520: Instalación del driver acx100 para GNU/Linux

Jorge G. Arenas <syvic@sindormir.net>
<http://sindormir.net>
14/04/2004 Rev 0.1

Este texto está publicado bajo la Licencia de Documentación Libre de GNU (<http://es.tldp.org/Licencias/fdles/fdl-es.html>)

0.- Antes de empezar

Antes de comenzar con los procesos de instalación, suponemos que tienes la tarjeta pinchada en tu ordenador, ya sea por medio de la pci o bien con la mini-pci (Este documento no trata las tarjetas dlink USB)

Para comprobar que está correctamente detectada por el sistema (aunque no nos funcione). Ejecutamos el siguiente comando, a ver qué nos dice:

```
midnight ~# lspci | grep ACX
0000:02:03.0 Network controller: Texas Instruments ACX 100 22Mbps Wireless Interface
```

Si aparece algo como ésto, ¡Enhorabuena! Has pinchado correctamente la tarjeta a su bus. Vamos a ver un poco más de información de la tarjeta, por curiosidad. (Substituye 02:03.0 por el número que corresponda en tu PC):

```
midnight ~# lspci -vvvs 02:03.0
0000:02:03.0 Network controller: Texas Instruments ACX 100 22Mbps Wireless Interface
Subsystem: D-Link System Inc: Unknown device 3b01
Control: I/O+ Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR-
Status: Cap+ 66Mhz- UDF- FastB2B- ParErr- DEVSEL=medium >TAbort-
Latency: 32, cache line size 08
Interrupt: pin A routed to IRQ 11
Region 0: I/O ports at ec60 [size=32]
Region 1: Memory at f8ffe000 (32-bit, non-prefetchable) [size=4K]
Region 2: Memory at f8fe0000 (32-bit, non-prefetchable) [size=64K]
Capabilities: [40] Power Management version 2
Flags: PMEClk- DSI- D1+ D2+ AuxCurrent=0mA PME(D0+,D1+,D2+,D3hot+,D3cold-)
Status: D3 PME-Enable- DSel=0 DScale=0 PME-
```

1.- Pasos previos: el kernel

La configuración del kernel para la serie 2.4 requiere que ciertas opciones estén habilitadas para el correcto funcionamiento de este driver. Si no están de esta forma en nuestro kernel será necesario recompilar. El núcleo que generemos debe tener activadas las opciones **CONFIG_CARDBUS**, **CONFIG_NET_RADIO** y **CONFIG_NET_WIRELESS**; y DESACTIVADA la opción de multiprocesador simétrico: **CONFIG_SMP**.

Las rutas en el menuconfig de 2.4 para ellas son:

```
CONFIG_CARDBUS:      General setup ->
                    PCMCIA/CardBus support ->
                    <*> PCMCIA/CardBus support
                    [*] CardBus support

CONFIG_NET_RADIO y   Network device support ->
CONFIG_NET_WIRELESS: Wireless LAN (non-hamradio) ->
                    [*] Wireless LAN (non-hamradio)
```

Es muy importante que el fichero `/usr/src/linux/.config` corresponda con el kernel que estamos ejecutando, en caso contrario podríamos tener problemas extraños con el funcionamiento de esta tarjeta.

Resaltar que este driver está preparado para funcionar en los -no tan-nuevos kernels de la serie 2.6

2.- Paquetes necesarios

Para poder gestionar la tarjeta correctamente necesitamos un software genérico para todas las tarjetas wireless que deberemos instalar:

- Paquete `pcmcia-cs`, aunque esta tarjeta no utiliza el bus `pcmcia`, sino el `cardbus`, requiere de ciertas utilidades que proporcionan las `pcmcia-cs`.
- Paquete `wireless-tools`, que proporciona las herramientas a nivel usuario para gestionar los parámetros propios de 802.11n (Siendo n: a,b ó g).
- Utilidad `wavemon`, muy útil para la monitorización de la calidad del enlace y para obtener estadísticas. Opcional.
- Cliente `dhcp`, muy útil también en las redes ciudadanas para obtener los datos de red automáticamente del servidor. Opcional.

3.- Hardware soportado por este driver

No todas las tarjetas que llevan este chip pueden ser controladas por este driver. Los desarrolladores han puesto a disposición del público una tabla donde se recogen todas aquellas tarjetas que funcionan o que dan problemas con el chip de Texas Instrument:

<http://acx100.sourceforge.net/matrix.html>

Es muy recomendable mirar esta página antes de adquirir una tarjeta con este chip de TI, ya que al parecer no es muy fiable, e incluso recomiendan no comprar tarjetas con él. Por ejemplo, en mi **Gentoo**, este es el aviso que mete en el `syslog` cuando intenta cargar el módulo:

```
[knl] acx100: It looks like you were coaxed into buying a wireless network card
[knl] acx100: that uses the mysterious ACX100 chip from Texas Instruments.
[knl] acx100: You should better have bought e.g. a PRISM(R) chipset based card,
[knl] acx100: since that would mean REAL vendor Linux support.
[knl] acx100: Given this info, it's evident that this driver is quite EXPERIMENTAL,
[knl] acx100: thus your mileage may vary Visit http://acx100.sf.net for support
```

Lo pone bien clarito, ¿verdad?

4.- Let's go: Descargando el driver

Bajamos la última versión del CVS, ya que se nota considerablemente el rendimiento y la calidad de los drivers de desarrollo con respecto a las versiones "paquetizadas":

```
cvs -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/acx100 login
```

Pulsar intro simplemente donde pide "CVS password:"

```
cvs -z3 -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/acx100 co acx100
```

(La versión sobre la que se realiza este HOWTO es concretamente la 2004.04.13)

Una vez haya terminado el proceso, veremos que sobre el directorio actual se nos ha creado el subdirectorio `acx100`.

5.- Compilando, que es gerundio

Una vez hemos descargado la última versión del driver, sólo nos queda compilar siguiendo los siguientes sencillos pasos:

1. Entramos al directorio `acx100/`
2. `make fetch_firmware`

Esto descargará los drivers oficiales para windows (tm) de dlink. La versión disponible a día de hoy es la 307. Si quieres saber más acerca de este extraño procedimiento, tienes documentación aquí: http://acx100.sf.net/ndis_cludge.html

3. `make config.mk`
4. `make driver`
5. `make install`

Esto copiará el módulo del kernel a `/lib/modules/`uname -r`/net/acx_pci.o`)

6.- Probar el driver

Si el proceso anterior finalizó con éxito, es hora de hacer la primera prueba con nuestra tarjeta. Para ello, ejecutamos el siguiente comando (sin movernos del directorio):

```
insmod acx_pci firmware_dir=`pwd`/firmware
```

Esto debería cargar provisionalmente el módulo capaz de controlar la tarjeta. El dispositivo que creará el sistema se denominará **wlan0**. Para comprobar que efectivamente el sistema ha reconocido la placa realizamos el siguiente comando:

```
midnight acx100 # iwconfig wlan0
wlan0 IEEE 802.11b+ ESSID:"STAAB11B2" Nickname:"acx100 v0.2.0pre8"
Mode:Auto Channel:1 Access Point: 00:00:00:00:00:00
Bit Rate=11Mb/s Tx-Power:20 dBm Sensitivity=187/255
Retry min limit:5 RTS thr:off
Encryption key:off
Power Management:off
Link Quality:0 Signal level:0 Noise level:0
Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0
Tx excessive retries:0 Invalid misc:0 Missed beacon:0
```

Si nos aparece una información similar, entonces podemos proceder con la instalación definitiva y la configuración necesaria para su autoarranque con el sistema.

7.- Instalando definitivamente el driver

El *firmware* que nos ha descargado durante la compilación necesita estar en una localización fija en el sistema, ya que es necesario en cada inicialización del módulo, esto es, siempre. Aunque dentro de la estructura de directorios linux, no hay ningún lugar en el que sea lógico situar estos drivers de *windows*, sugiero que se copien al directorio `/lib/firmware`, aunque eres libre de ponerlo en cualquier otra ruta siempre que cambies el script de inicialización de la tarjeta.

El propio paquete de `acx100` provee un script de inicio de servicio y otro de parada, pero no están adaptados al *SystemV* de Linux y deberemos cambiarlo para adaptarlo a nuestra distribución. De todas formas, el script está un poco embarullado, así que en el **Anexo I** tenéis otro script más apto para colocar en `/etc/init.d/` y posteriormente enlazar desde los runlevels deseados.

8.- ¡Quiero ser un juanker!

¡Ah amigo! ¿Así que no tienes un AP propio, pero tu vecino si? ¿Que cómo haces para meterte en su red? Primero necesitarás saber qué **essid** (identificador de red wireless) utiliza y rezar para que no use **WEP** (Wired Equivalent Privacy) o que tenga mucho tráfico wireless si así es.

El chip que lleva esta tarjeta permite un modo especial de operación llamado modo monitor, que hace que la tarjeta se ponga en una especie de modo promiscuo pero a nivel físico. Es decir, nuestra tarjeta se convierte en un escaner de radiofrecuencias a 2.4Ghz.

El modo monitor, se configura mediante las llamadas privadas al sistema (*private ioctl*). Podemos ver si nuestra tarjeta lo soporta mediante el comando **iwpriv**:

```
midnight acx100 # iwpriv wlan0
wlan0 Available private ioctl :
      set_debug      (8BE0) : set   1 int   & get   0
      list_reg_domain (8BE1) : set   0       & get   0
      set_reg_domain  (8BE2) : set   1 byte & get   0
      get_reg_domain  (8BE3) : set   0       & get  1 byte
      set_s_preamble  (8BE4) : set   1 byte & get   0
      get_s_preamble  (8BE5) : set   0       & get  1 byte
      set_antenna     (8BE6) : set   1 byte & get   0
      get_antenna     (8BE7) : set   0       & get   0
      set_rx_ant      (8BE8) : set   1 byte & get   0
      set_tx_ant      (8BE9) : set   1 byte & get   0
      set_ed          (8BEA) : set   1 int   & get   0
      set_cca         (8BEB) : set   1 byte & get   0
      set_led_power   (8BEC) : set   1 byte & get   0
      monitor        (8BED) : set   2 int   & get   0
      test           (8BEE) : set   0       & get   0
```

Como vemos, el penúltimo modo permite establecer la modalidad de monitor para la tarjeta. Sabremos si una tarjeta está en modo monitor porque al hacer un **ifconfig** nos muestra una dirección física de red (MAC) un tanto extraña:

```
midnight acx100 # ifconfig wlan0 | grep HW
wlan0 Link encap:UNSPEC HWaddr 00-80-C8-AB-11-B2-00-00-00-00-00-00-00-00-00-00
```

En lugar de la salida que ofrece cuando está operando en modo normal:

```
midnight acx100 # ifconfig wlan0 | grep HW
wlan0 Link encap:Ethernet HWaddr 00:80:C8:AB:11:B2
```

¿Y cómo ponemos la tarjeta en modo monitor? Pues muy sencillo, sólo tenemos que teclear el siguiente comando:

```
iwpriv wlan0 monitor 2 4
```

Ahora sólo nos falta lanzar alguna herramienta juanker como **kismet**, para que nos empiece a escanear en todos los canales buscando redes.

La línea de **kismet** que necesitas para que funcione es:

```
source=acx100,wlan0,generic
```

Tienes otras herramientas muy útiles para este fin, como **airsnort**, **airtraf**, **Wellenreiter**, **apradar** o **wlandetect** (Estas dos últimas no las conozco, son las recomendadas en la web del driver)

9.- Quiero saber más

Si has llegado hasta este punto, significa que **NO** te has leído el maravilloso README que viene con el paquete. Está en inglés, pero merece mucho la pena leerlo, ya que aparte de enseñarte cómo se instala el driver, tiene unas anotaciones muy interesantes. Una perla:

"This BRAINDEAD STUPIDITY in device naming easily entitles D-Link for the "Most Braindead Hardware Vendor 2003" award." ;)

Por otro lado, en la propia página del proyecto tienes mucha más documentación, entre la que me permito reseñar los siguientes links:

Especificaciones del protocolo 802.11b: <http://standards.ieee.org/getieee802/>

Otro howto de instalación: http://www.houseofcraig.net/acx100_howto.php

Lista de drivers para tarjetas wireless en Linux y sus características: http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Linux.Wireless.drivers.html

Información y documentación general sobre wireless: <http://madridwireless.net>

ANEXO I - Script inicio/parada servicio acx100

```
#!/bin/bash
# Copiar a /etc/init.d/wireless (p.e) y enlazarlo desde los runlevels
# correspondientes.
DEV=wlan0
ESSID="madridwireless"
RATE=11M
CHAN=6
TXPOWER=18
MODE=Managed
#DEBUG=0xb
#KEY="B401CD21B44CCD21DEADBEEF11" # WEP128
#ALG=open # open == Open System, restricted == Shared Key
USE_DHCP=0 # set to 1 for auto configuration instead of fixed IP setting
IP=192.168.0.10
NETMASK=255.255.255.0
GATEWAY=192.168.0.254
FIRMWARE_DIR="/lib/firmware"

#No tocar a partir de aquí
case $1 in
start)
    sync
    sleep 1
    insmod acx_pci firmware_dir=$FIRMWARE_DIR
    iwconfig $DEV rate $RATE
    iwconfig $DEV channel $CHAN
    iwconfig $DEV txpower $TXPOWER
    iwconfig $DEV essid "$ESSID"
    iwconfig $DEV mode $MODE
    #iwconfig $DEV key $ALG "$KEY"

    if test $USE_DHCP -eq 1; then
        dhcpcd $DEV
    else
        ifconfig $DEV $IP netmask $NETMASK
        route add default gw $GATEWAY
    fi
    ifconfig $DEV mtu 576
    sync;;

stop)
    ifconfig $DEV down
    sleep 1
    sync
    rmmmod acx_pci;;

status)
    iwconfig $DEV;;

esac
#Fin script
```