WLAN on DILNetPC DNP9200
External SWAP Device on DNP9200

Picture 1: DNP9200 + eval board SK23, external mini USB2.0 Hub with a 11Mbps WLAN USB Adapter and 1GB high speed(192x) USB SWAP device.

Attention: This HOWTO works also with a Netgear MA111

So I will explain now all the steps you need to get the WLAN USB Adapter D-Link DWL-122 and external SWAP device work. First you need the DIL/NetPC DNP/9200 Linux Upgrade Kernel 2.6.16 CD. Ver. 1.0 07/2006 © SSV from www.dilnetpc.com. # is the root user, $ is the local user.

1.)copy the whole CD to a local directory

```
$ cd  ~
$ mkdir DNP
$ cd DNP
$ cp -r /media/cdrom/* .   (from the mounted CD)
# chmod -R 777 *
# chown -R user:user *      (e.g. user=bob  or what else, your username)
$ ls
img-SSV20060616  mHT9200-24.pdf  modules  other  rimage-0.62.26  src
toolchain zimage
```

2.)install ARM cross compiler

```
# cd /usr/local/
# tar -xzf (path_to_tar)/arm-????????.tgz
```

3.)extract kernel sources

```
$ cd ~/DNP/src
$ tar -xzf linux-2.6.16.20-at91-ssv1.tar.gz      (takes a bit..)
```

4.)make a new kernel

```
$ . kbuildenv.sh    (maybe edit this file first, module directories..)
$ cd linux-2.6.16.20-at91-ssv1
$ cp ../../zimage/CONFIG-DNP9200-2.6.16.20-SSV20060616 dnp.config
$ make menuconfig   (packet libncurses5-dev are required)
```

then you say load external config.. -> dnp.config
and you have to activate following kernel options:

```
General Setup
        [*] Support for paging of anonymous memory (swap)
Device Driver
        Network Device Support
                Wireless LAN (non-hamradio)
                        [*] Wireless LAN drivers (non-hamradio) & Wireless Extensions
```

(if you would have support for a webcam)

```
        Muldimedia Devices
                <*> Video For Linux
        USB Support
                <*> USB OV511 Camera support     (for e.g. Logitech WebCam III)
```

Go back, save+exit

```
$ make Image
$ make modules
$ make modules_install
```

5.)build the WLAN driver

Download the Version 0.2.5 from ftp://ftp.linux-wlan.org/pub/linux-wlan-ng/
to ~/DNP/

```
$ tar -xzf linux-wlan-ng-0.2.5.tar.gz
$ cd DNP/src
$ . kbuildenv.sh
$ cd ..
$ cd linux-wlan-ng-0.2.5
# cd /usr/bin       (maybe you don't need this, try to enter cc)
# ln -s gcc cc      (also don't need if cc is known on the system)
$ ./Configure
-------------- Linux WLAN Configuration Script -------------

The default responses are correct for most users.
Build Prism2.x PCMCIA Card Services (_cs) driver? (y/n) [n]:
```

Build Prism2 PLX9052 based PCI (_plx) adapter driver? (y/n) [n]:
Build Prism2.5 native PCI (_pci) driver? (y/n) [n]:
Build Prism2.5 USB (_usb) driver? (y/n) [y]:
Linux source directory [/home/mich/DNP/src/linux-2.6.16.20-at91-ssv1]:
The kernel source tree is version 2.6.16.20-at91-ssv1.
WARNING: the current running kernel is actually version 2.6.17-2-686.
The current kernel build date is Fri Aug 11 20:25:51 2006.
Alternate target install root directory on host []:
  Module install directory [/lib/modules/2.6.16.20-at91-ssv1]:
It looks like you have a System V init file setup.
Prefix for build host compiler? (rarely needed) []: arm-ssv1-linux-
Build for debugging (see doc/config.debug) (y/n) [n]: y
Configuration successful. Now type 'make' and pray.

-> answers to the ./Configure script are found up here
-> don't ignore the cc error, if you have this make a symlink to gcc
-> now type:

$ make
(.... /bin/sh: ./mkmetadef: cannot execute binary file   ....)
And I'm be sure you get this error, the host machine tries to execute an arm-executable. Type make again.

$ make
(.... /bin/sh: ./mkmetastruct: cannot execute binary file   ....)
And you also get this error, you have to execute these 2 files on the DILNetPC! Type again make:

$ make
(....   ../include/wlan/p80211metastruct.h:47:1: unterminated #ifndef   ....)
You get sth like this -> in this header file is sth missing, now get this pice of missing code!

Copy these files (~/DNP/linux-wlan-ng-0.2.5/src/mkmeta/mkmetadef and mkmetastruct) with ftp or what else to the DNP9200 root directory and execute them, redirect the output to a file and copy the results back.

$ telnet [ip from DNP9200]
[root@emblinux /root]$./mkmetadef > def.h
[root@emblinux /root]$./mkmetastruct > struct.h

So copy the def.h and struct.h back to your local machine and goto the directory ~/DNP/linux-wlan-ng-0.2.5/src/include/wlan (you can copy def/struct.h also here)
Here you can find 2 header files named: p80211metadef.h and p80211metastruct.h . Open the p80211metadef.h add the content of def.h and add a #endif at the end of the file, do this again with p80211metastruct.h and struct.h Do this in your favourite text editor with copy/paste ;)
And do this for compile the rest (please set these links back after successful compile).
cd /usr/bin
rm gcc (is a symlink, look forward where does this link point)
ln -s /usr/local/arm-ssv1-linux/bin/arm-ssv1-linux-gcc gcc

# mv ld ld_orig (move this back to ld later)
# ln -s /usr/local/arm-ssv1-linux/bin/arm-ssv1-linux-ld ld
$ make (and pray..)

So all is done.
The modules you needes are:
~/DNP/linux-wlan-ng-0.2.5/src/p80211/p80211.ko
~/DNP/linux-wlan-ng-0.2.5/src/prism2/driver/prism2_usb.ko

6.)build the new Image

$ cd DNP/img-SSV20060616
$ mv Image Image_orig
$ cp ../src/linux-2.6.16.20-at91-ssv1/arch/arm/boot/Image .
$ gzip -d rimage.gz
# mount -o loop -t minix rimage /mnt
# cd /mnt/flash
# cp /home/bob/DNP/linux-wlan-ng-0.2.5/src/p80211/p80211.ko .
# cp /home/bob/DNP/linux-wlan-ng-0.2.5/src/prism2/driver/prism2_usb.ko .
# cp /home/bob/DNP/linux-wlan-ng-0.2.5/src/wlanctl/wlanctl .
# create a file „autostart.sh“ with following content

```
#!/bin/sh
insmod p80211.ko
insmod prism2_usb.ko
./wlanctl wlan0 lnxreq_ifstate ifstate=enable
./wlanctl wlan0 lnxreq_autojoin ssid=yourESSID authtype=opensystem
dhcpcd wlan0
#ifconfig eth0 down  (disable the cable ethernet)
#mkswap /dev/sda1
#swapon /dev/sda1
#ifconfig eth0 down       (you don't need an ethernet cable any more)
```

# chmod +x autostart.sh
# sync
# cd /
# umount /mnt
$ cd DNP/img-SSV20060616
# fsck.minix rimage
$ gzip -9 rimage
$ ./mkimage.sh

finnished. Flash the img-dnp9200 to the DNP9200 and reboot it. You must not write such an autostart script. You can execute these commands by hand for see what they are doing.

For demonstration:

```
mich@debian700:~$ telnet 192.168.1.105
Trying 192.168.1.105...
Connected to 192.168.1.105.
Escape character is '^]'.
```

```
- SSV Embedded Linux - Version 0.62.26

emblinux login: root
Password:
[root@emblinux /root]$cat /proc/meminfo
MemTotal:        30656 kB
MemFree:         13296 kB
Buffers:          8084 kB
Cached:           3464 kB
SwapCached:          0 kB
Active:           4992 kB
Inactive:         7996 kB
HighTotal:           0 kB
HighFree:            0 kB
LowTotal:        30656 kB
LowFree:         13296 kB
SwapTotal:     1007592 kB
SwapFree:      1007592 kB
Dirty:              12 kB
Writeback:           0 kB
Mapped:           2892 kB
Slab:             2124 kB
CommitLimit:   1022920 kB
Committed_AS:     3892 kB
PageTables:        132 kB
VmallocTotal:   989184 kB
VmallocUsed:     34660 kB
VmallocChunk:   954364 kB
[root@emblinux /root]$ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0

wlan0     Link encap:Ethernet  HWaddr 00:09:5B:B4:0C:EE
          inet addr:192.168.1.105  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST NOTRAILERS RUNNING  MTU:1500  Metric:1
          RX packets:161 errors:0 dropped:0 overruns:0 frame:0
          TX packets:111 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000

[root@emblinux /root]$ping www.google.at
PING www.l.google.com (72.14.221.99): 56 data bytes
64 bytes from 72.14.221.99: icmp_seq=0 ttl=245 time=31.1 ms
64 bytes from 72.14.221.99: icmp_seq=1 ttl=245 time=30.5 ms

--- www.l.google.com ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 30.5/30.8/31.1 ms
[root@emblinux /root]$
```

Happy WLAN!