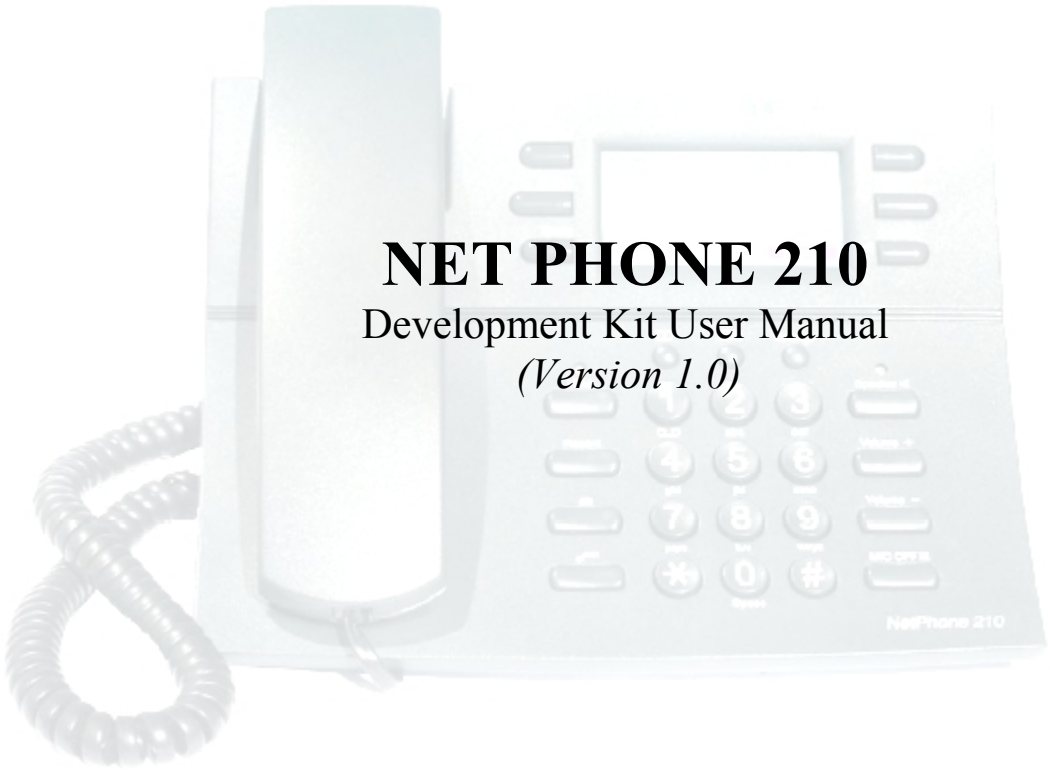
voipac



NET PHONE 210

Development Kit User Manual

(Version 1.0)

Table of Contents

1 Overview

1.1 Hardware specification

- XScale PXA 255/200MHz
- FLASH 32MB
- SDRAM 64MB
- 2 x 10Mb Ethernet
- Full graphic LCD display
- Numeric keypad
- Headset connector
- PS2 interface for external keyboard

Possible configurations

- CPU XScale PXA255 200 - 400 MHz
- FLASH 8 - 32MB
- SDRAM 32 - 64MB

1.2 Software specification

- OS Linux 2.4.19

2 Getting started

This section describes how to connect your development kit and how to begin with the work.

Connecting the cables:

1.Connect RS232 serial cable to NP210 phone on one side and to the RS232 connector in PC on other side.

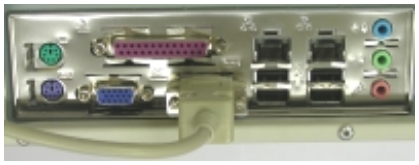


Figure 2-1 PC RS232 connection

Figure 2-2 NP210 RS232 connection

2.Connect **Ethernet 1** (closer to power connector) cable to the phone and to the switch/hub (if you want to connect directly to PC, you must use cross cable).

Figure 2-3 Ethernet connection

3.On your PC run Hyperterminal (Windows: Start -> Programs -> Accessories -> Communication -> Hyperterminal) or similar program with following parameters:

Baud rate – 38400
Data bits – 8
Parity – None
Stop bits – 1
Flow control – None
Serial port number – COM1 or COM2, depends, where you connect serial cable.

4.Connect the provided DC adapter into the power jack. If you use other DC source, than supplied, ensure, that input voltage is in interval 9 - 28V DC and minimum current is 600mA. (outside pole of power jack is negative).

Figure 2-4 Power connection

5.When you switch the power on, the phone will start (you can see messages on LCD display), and some messages will be coming in terminal via the RS232 interface.

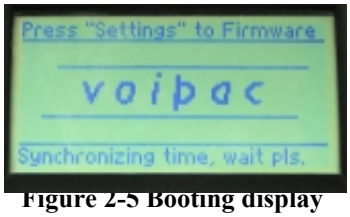


Figure 2-5 Booting display

Terminal booting messages:

```
ARMboot 1.3.0 by Voipac <www.voipac.com> (Aug 17 2005 - 20:04:09)

ARMboot code: a1000000 -> a1016ba4
CPU: Intel XScale-PXA255 (ARM 5TE) revision A0
Clock: Mem=99.53MHz (*27), Run=199.07MHz (*2), Turbo=199.07MHz (*1.0,inactive)
DRAM Configuration:
Bank #0: a0000000 64 MB
Bank #1: a4000000 0 KB
Bank #2: a8000000 0 KB
Bank #3: ac000000 0 KB
Flash: 32 MB
Hit any key to stop autoboot:  0

Starting Linux kernel image at 0x40000
        Architecture type: 89
            Command line: mem=64M root=/dev/mtdblock2 rw console=ttyS0,38400
mac0=00
0d15020bb8 mac1=000d15020bb9

Uncompressing Linux..................................... done, booting
the
kernel.
Linux version 2.4.19-rmk7-pxa2-iphone (root@sarge) (gcc version 2.95.3
20010315
(release)) #4 Thu Aug 11 13:01:41 CEST 2005
Kernel_ID: N210-23329.1
CPU: XScale-PXA255 revision 6
Machine: Voipac PXA250 Developement board
Memory clock: 99.53MHz (*27)
Run Mode clock: 199.07MHz (*2)
Turbo Mode clock: 199.07MHz (*1.0, inactive)
On node 0 totalpages: 16384
zone(0): 16384 pages.
zone(1): 0 pages.
zone(2): 0 pages.
Kernel command line: mem=64M root=/dev/mtdblock2 rw console=ttyS0,38400
mac0=000
d15020bb8 mac1=000d15020bb9
Calibrating delay loop... 198.65 BogoMIPS
Memory: 64MB = 64MB total
Memory: 63408KB available (1111K code, 213K data, 56K init)
Dentry cache hash table entries: 8192 (order: 4, 65536 bytes)
Inode cache hash table entries: 4096 (order: 3, 32768 bytes)
Mount-cache hash table entries: 1024 (order: 1, 8192 bytes)
Buffer-cache hash table entries: 4096 (order: 2, 16384 bytes)
Page-cache hash table entries: 16384 (order: 4, 65536 bytes)
POSIX conformance testing by UNIFIX
Linux NET4.0 for Linux 2.4
Based upon Swansea University Computer Society NET3.039
Initializing RT netlink socket
Starting kswapd
JFFS2 version 2.1. (C) 2001 Red Hat, Inc., designed by Axis Communications AB.
pty: 256 Unix98 ptys configured
Voipac LCD driver (c) 2003
Serial driver version 5.05c (2001-07-08) with no serial options enabled
ttyS00 at 0x0000 (irq = 15) is a PXA UART
ttyS01 at 0x0000 (irq = 14) is a PXA UART
```

```
ttyS02 at 0x0000 (irq = 13) is a PXA UART
SA1100/PXA Watchdog Timer: timer margin 60 sec
eth0: cs8900 rev J found at 0xf0000300
cs89x0 media RJ-45, IRQ 37, programmed I/O, MAC 00:0d:15:02:0b:b8
eth1: cs8900 rev J found at 0xf1000300
cs89x0 media RJ-45, IRQ 42, programmed I/O, MAC 00:0d:15:02:0b:b9
ac97_codec: AC97 Audio codec, id: 0x4144:0x5360 (Analog Devices AD1885)
Probing Lubbock flash at physical address 0x00000000 (32-bit buswidth)
Using buffer write method
Using static partition definition
Creating 3 MTD partitions on "Lubbock flash":
0x00000000-0x00040000 : "Bootloader"
0x00040000-0x00100000 : "Kernel"
0x00100000-0x02000000 : "Filesystem"
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP
IP: routing cache hash table of 512 buckets, 4Kbytes
TCP: Hash tables configured (established 4096 bind 4096)
ip_conntrack (512 buckets, 4096 max)
ip_tables: (C) 2000-2002 Netfilter core team
NET4: Unix domain sockets 1.0/SMP for Linux NET4.0.
NET4: Ethernet Bridge 008 for NET4.0
NetWinder Floating Point Emulator V0.95 (c) 1998-1999 Rebel.com
VFS: Mounted root (jffs2 filesystem).
Freeing init memory: 56K
INIT: version 2.84 booting
Mounting local filesystems...
proc on /proc type proc (rw)
none on /var type tmpfs (rw)
devpts on /dev/pts type devpts (rw)
Creating /var subdirectories...
INIT: Entering runlevel: 2
Decompressing iphone...
Process ID:20
Press "Settings" to Firmware
Setting host name to `NP210'
Setting up IP spoofing protection: rp_filter.
eth1: 10Base-T (RJ-45) has no cable
eth1: using half-duplex 10Base-T (RJ-45)
eth0: using half-duplex 10Base-T (RJ-45)
done.
Fri Jan  1 00:00:00 GMT 1999
23  Aug  08:15:18  ntpdate[69]:  step  time  server  195.113.144.201  offset
209636117.7
62490 sec
Starting OpenBSD Secure Shell server: sshd.
Got signal 15.
Waiting for SIGQUIT signal with 1changed to 0
SUCCESS->END
Starting IPhone ...
At time: 0
Daemon started with pid 100
IPhone running.
Runlevel 2 READY

NP210
NP210 login:

No jabber server specified!
Cannot login
```

```
HTB init, kernel part version 3.6
```

You can debug phone software now. Password and login is set default to “root”.

3 Hardware

3.1 NP210 Block diagram

Device NP210 is builded from several parts:

- PCBs boards
  - NP210 Main Board
  - Display
  - Keyboard
  - Hook
- Housing
  - Plastic parts
  - Screws
  - Rubber buttons
  - ...

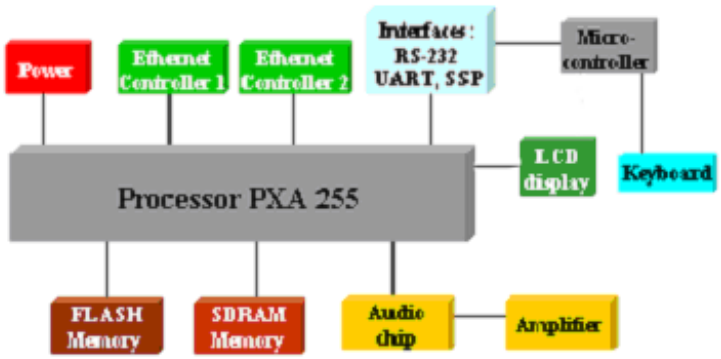


Figure 3-6 Block diagram of NP210

3.2 NP210 Power supply

A standard 2.1mm DC jack is used to provide power to the board. The center of jack is positive. It is recommended to power the board by stabilized source 9V-28V. Power supply provided by VOIPAC is 12V/600mA.

3.3 NP210 Main Board

3.3.1 Block Diagram

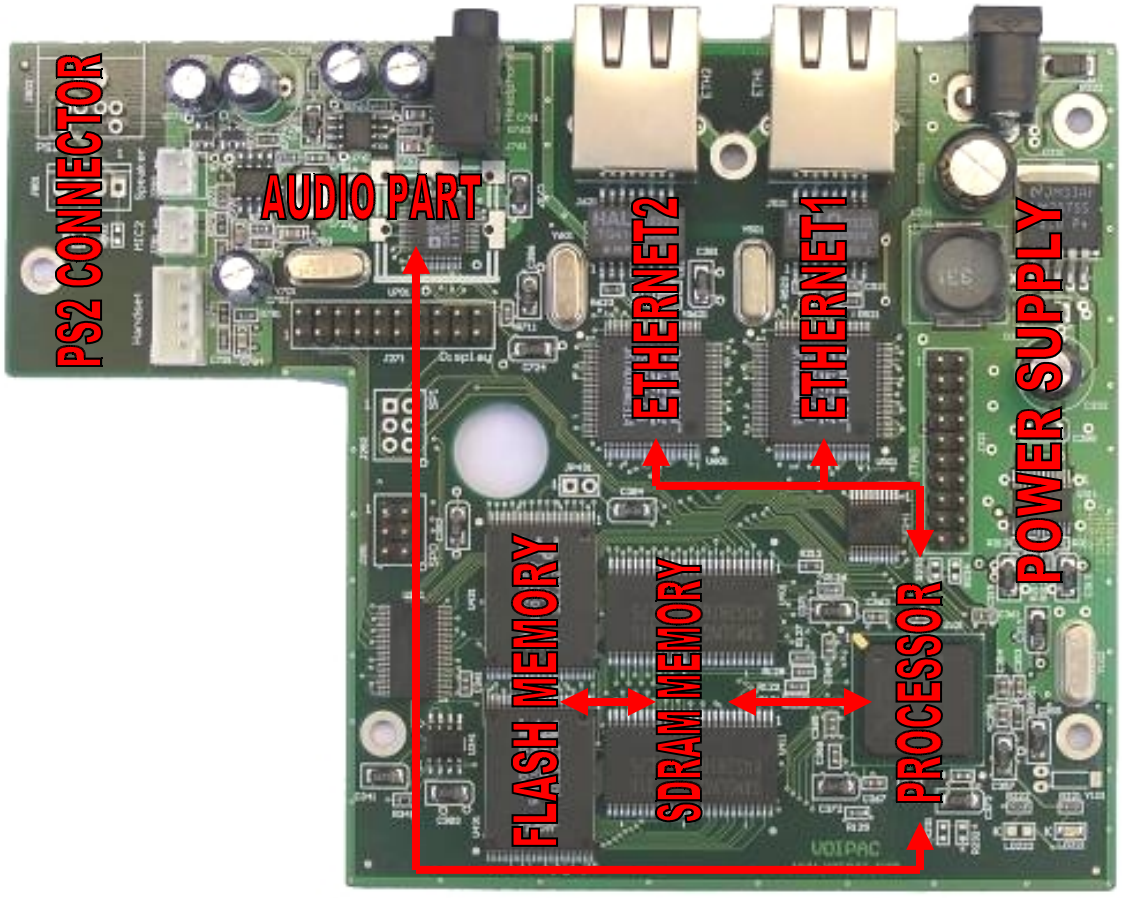


Figure 3-7 NP210 Main board picture with block description

3.3.2 Board Layout

Components are located on top and bottom side of board.

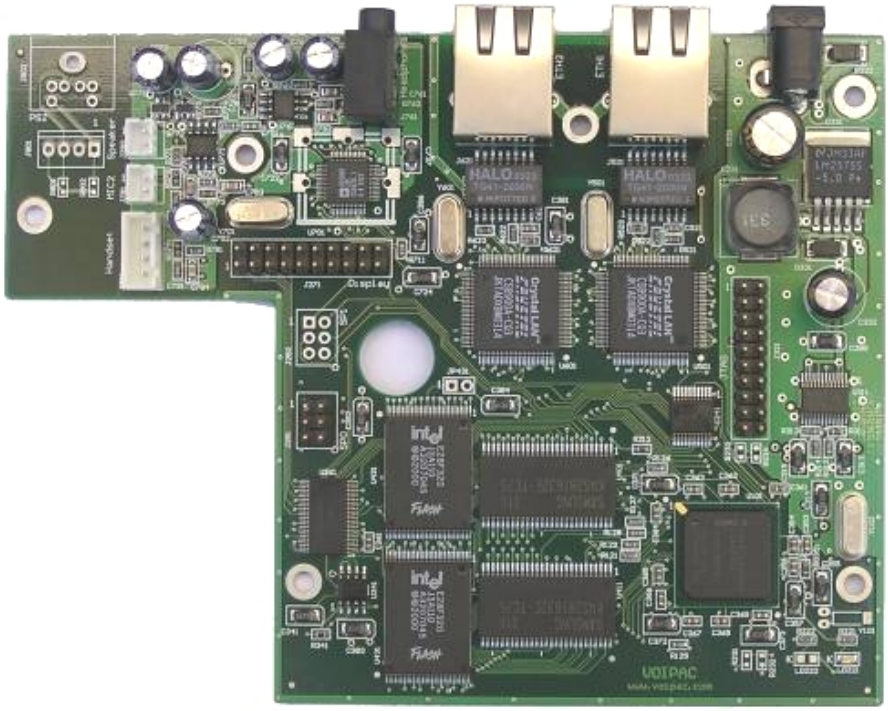


Figure 3-8 Main Board TOP VIEW

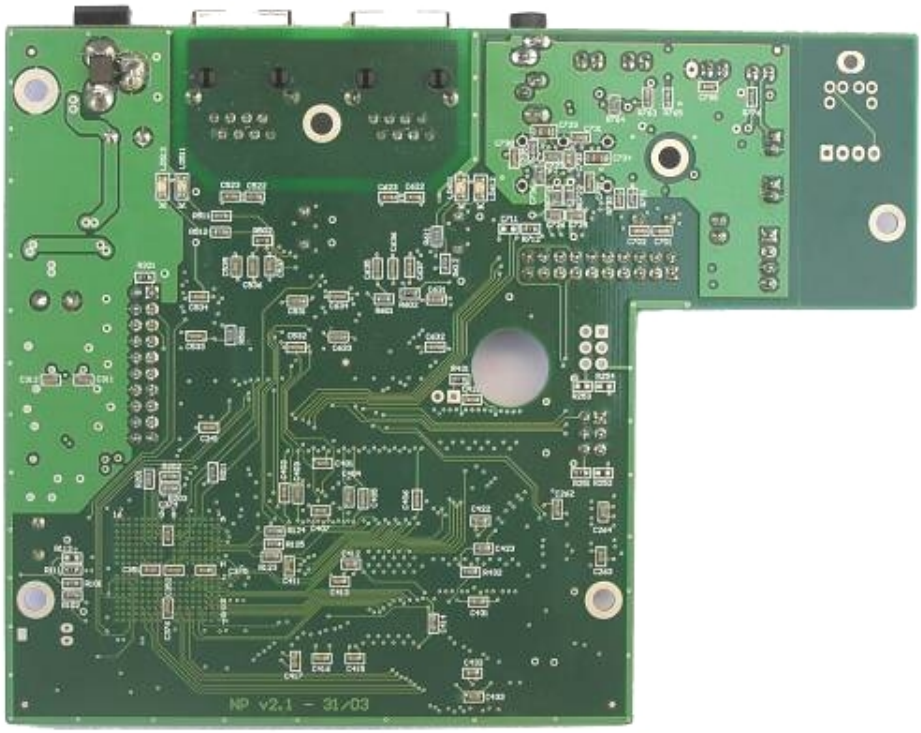


Figure 3-9 Main Board BOTTOM VIEW

3.3.2.1 Production information

PCB

- NP210 PCB is 8-layer PCB with Top and Bottom Overlay description.
- Solder mask is from both sides – Top and Bottom.
- Isolated pad is used under crystals.

3.3.3 Memory map

Pin	Description
CS0	(0-0x03FF.FFFF) flash memory
CS1	(0x0400.0000-0x07FF.FFFF) unused, used as GPIO pin
CS2	(0x0800.0000-0x0BFF.FFFF) Ethernet chip 1 (offset 0x400)
CS3	(0x0C00.0000-0x0FFF.FFFF) Ethernet chip 2 (offset 0x400) and PCMCIA status buffer (U502 – offset 0x0)
CS4	(0x1000.0000-0x13FF.FFFF) unused, used as GPIO pin
CS5	(0x1400.0000-0x17FF.FFFF) unused, used as GPIO pin

PCE1 is used for Display mapping

3.3.4 Description of basic chips on main board

CPU

Intel PXA255 is used. The PXA255 processor is an integrated system-on-a-chip design based on the Intel® XScale™ microarchitecture core with many peripherals to let one design products for the handheld market.
Possible frequency is 200, 300 or 400MHz. The PXA255 processor is a 32-bit device.

SDRAM

NP210 uses two 128 or 256 Mbit SDRAM (2x16 / 2x32 MByte) devices organized as one 32-Bit Bank. They support 100MHz operation. For the cheaper version, one 16bit chip only is sufficient to be used – but than you need to re-set SDRAM registers.

FLASH memory

Two 4, 8 or 16MB Intel Strata Flash memory chips are used.

Used chips	Total capacity
E28F320-J3 (4MByte)	2x4 = 8MByte
E28F640-J3 (8MByte)	2x8 = 16MByte
E28F128-J3 (16MByte)	2x16 = 32MByte

Power Supplies

There are two power supplies used:

- LM2575 – simple switcher 1A step-down voltage regulator
  Input voltage 9-30V -> transformed to -> 5V/1A

- TPS70302 – dual-output low-dropout voltage regulator
  Voltage 5V -> transformed to -> 3.3V (IO and peripherals voltage) and 1.33V (Processor Core Voltage)

Power input is protected with 30V bi-directional transient voltage suppressor diode – fixed in version 2.2.

Reset
For system reset is used TLC7733 micropower supply voltage regulator.

Buffers
Display part is powered with 5V voltage. For bus level compatibility is used 74LVC16245 bus transceiver with 5V pins tolerance.
Serial ports are separated with buffer too. It is 74LVC244 octal buffer driver with 5V input tolerance.

Power Amplifiers
Dual operational amplifiers TS922 are used for Speaker, Microphone and Headphones signals amplifying.

3.3.5 Peripherals

3.3.5.1 JTAG

This interface is necessary for the first program saving into the FLASH memory. Usually, short software called bootlader (Armboot) is used to save this way. After this process is done, it is possible to communicate with board via serial interface and Ethernet network.
JTAG is used together with JFLASHMM program on PC.

Interface	Connector/Header
JTAG	J321 – JTAG

3.3.5.2 PXA255 serial ports

The PXA255 has three asynchronous serial ports (FFUART-SP0, BTUART-SERIAL1 and STUART-IR (unconnected in NP210 design)) and one synchronous serial port (SSPC) (unconnected in NP210 design).
The FFUART supports full handshaking. The maximum tested baud rate on this UART is 230.4 kbps. The BTUART supports RTS/CTS only and supports baud rates up to 921.6 kbps. STUART is tested at maximum baud rate 230.4kbps, but it does not support modem control capability.

Interface	Connector/Header
FFUART-SP0	J251 – SP0
BTUART-SP1	J252 – SP1
STURAT/IR	Not Connected
SSPC	Not connected

3.3.5.3 Ethernet controllers

There are two Crystal LAN 10Mbit Ethernet chip CS8900A. Each chip is equipped with two status LED's (active status, link status) on the bottom side. **Only Ethernet 1 is default initialized in Armboot!**

Interface	Connector/Header	LEDs
ETHERNET1	J521 – ETH1	LD511 (Link), LD512 (Activity)
ETHERNET2	J621 – ETH2	LD611 (Link), LD612 (Activity)

3.3.5.4 AC-97 stereo audio codec

One from PXA255 interfaces is AC'97. AC'97 Analog Devices audio codec (AD1885) allows transmitting and receiving analog audio data. The AD1885 is configured on AC'97 input 0.
Dual operational amplifiers TS922 are used for Speaker, Microphone and Headphones signals amplifying.

NOTE
After our experience, we recommend to change the AC97 codec. AD1885 is very ESD sensitive component. See this caution from AD1885 datasheet file:

CAUTION
ESD (electrostatic discharge) sensitive device. Electrostatic charges as high as 4000 V readily accumulate on the human body and test equipment and can discharge without detection. Although the AD1885 features proprietary ESD protection circuitry, permanent damage may occur on devices subjected to high-energy electrostatic discharges. Therefore, proper ESD precautions are recommended to avoid performance degradation or loss of functionality.

You can use it in NP210 design and it will be work perfectly, but we have received several complaints about this chip damage. We recommend changing it for example for UCB1400. This AC97 codec has good support in Linux and we have drivers and kernels with them.

Interface	Connector/Header
Output for speaker	J782 – Speaker
Microphone input	J781 – MIC2
Handset connector	J791 - Handset
Headphones output	J761 - Headphones

3.3.5.5 Display connection

3.3V processor bus is separated from 3.3V display interface with bus transceiver. Display is mapped into PCMCIA memory space and using PCE1 chip select for device selection.

Interface	Connector/Header
Display	J271 – Display

3.3.5.6 PS2 Keyboard connector

PS2 Keyboard connector is placed on NP210 board, but PS2 signals are connected with Display board via J801 connector and there are processed. When PS2 is used, don't forget to place also R802 and R801 components.

Interface	Connector/Header
PS2	J802 – PS2
PS2 to Display	J801

3.3.6 Connectors

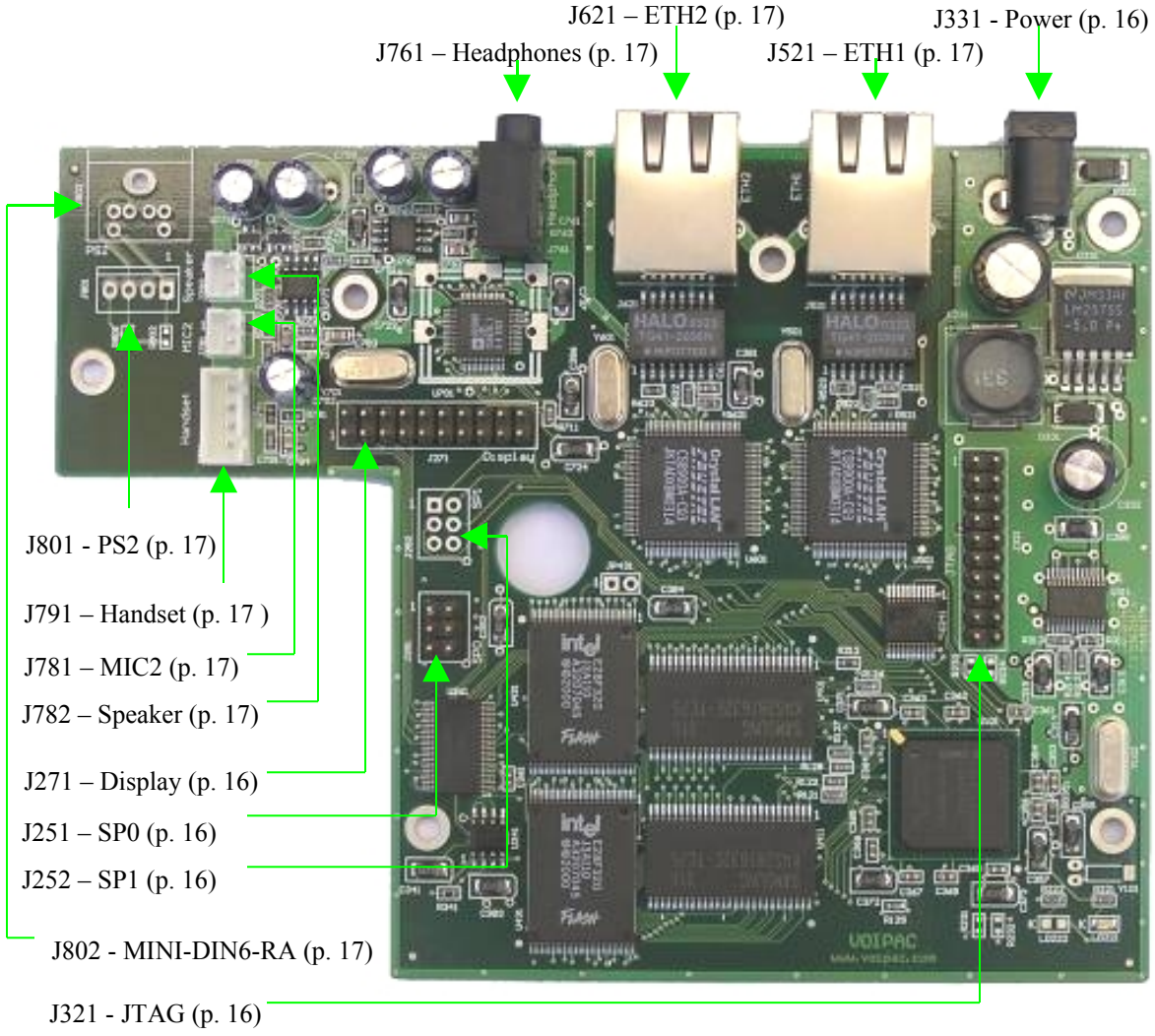


Figure 3-10 Main Board connector description

Notes:
VCC3 = 3.3V
VCC5 = 5V
NC = Not Connected

J271 Display – this connector is connected with display PCB

Pin	Description	Pin	Description
1	D_D0	2	D_D1
3	D_D2	4	D_D3
5	D_D4	6	D_D5
7	D_D6	8	D_D7
9	D_-RD	10	D_DATA/-CMD
11	D_-WR	12	-D_PCE1
13	D_RESET	14	-D_RESET
15	SP1_RXD	16	SP1_TXD
17	GND	18	GND
19	VCC3	20	VCC5

J251 SERIAL0 – asynchronous serial port (FULLUART), Input pins with 5V tolerance

Pin	Description	Pin	Description
1	SP0_RXD	2	SP0_TXD
3	SP0_CTS	4	SP0_RTS
5	GND	6	3.3V or 5.5V

J252 SERIAL1 – asynchronous serial port (BTUART), Input pins with 5V tolerance

Pin	Description	Pin	Description
1	SP1_RXD	2	SP1_TXD
3	SP1_CTS	4	SP1_RTS
5	GND	6	3.3V or 5.5V

J321 JTAG – JTAG/Debug port

Pin	Description	Pin	Description
1	VCC3	2	+3.3V
3	*P_TRST	4	GND
5	P_TDI	6	GND
7	P_TMS	8	GND
9	P_TCK	10	GND
11	NC	12	GND
13	TDO	14	GND
15	-RESET	16	GND
17	NC	18	GND
19	NC	20	GND

J331 Power – power supply connector (connector: 5.5x2.1mm, center positive)

Input voltage is 9-30V DC.

J521 RJ45 – Ethernet 1 connector

Pin	Description	Pin	Description
1	TXD-	2	TXD+
3	RXD+	4	NC
5	NC	6	RXD-
7	NC	8	NC

J621 RJ45 – Ethernet 2 connector

Pin	Description	Pin	Description
1	TXD-	2	TXD+
3	RXD+	4	NC
5	NC	6	RXD-
7	NC	8	NC

J761 Stereo Jack 3.5mm – Stereo Line Out
Stereo jack for external headphones

J781 MIC2 – External Microphone Input (Input jack for external microphone)

Pin	Description	Pin	Description
1	MIC2_IN	2	AGND

J782 Speaker – Speaker Output (Output to internal speaker)

Pin	Description	Pin	Description
1	Speaker_OUT	2	AGND

J791 Handset – Stereo Line Out (Stereo jack for external headphones)

Pin	Description	Pin	Description
1	MIC_1IN	2	HP_OUT
3	AGND	3	AGND

J801 PS2 – PS2 to Display board connector

Pin	Description	Pin	Description
1	CLK_PS2	2	GND_PS2
3	DATA_PS2	4	VCC5_PS2

J802 MINI-DIN6-RA – PS2 input connector

Pin	Description	Pin	Description
1	DATA_PS2	2	NC
3	GND_PS2	4	CLK_PS2
5	CLK_PS2	6	NC

3.3.7 LEDs and Jumper

Top side
LD221 – Power LED – lighting when board is powered

LD222 – User LED (for example is used for software upgrade status)

Bottom side
LD511 – Link status of Ethernet 1 device
LD512 – Activity status of Ethernet 1 device

LD611 – Link status of Ethernet 1 device
LD612 – Activity status of Ethernet 1 device

Jumper
JP431 – not used. Can be used, when R431 is not placed. Then CLOSED = FLASH can be erased, OPEN = FLASH content is protected and can't be erased.

3.4 Display

3.4.1 Block Diagram

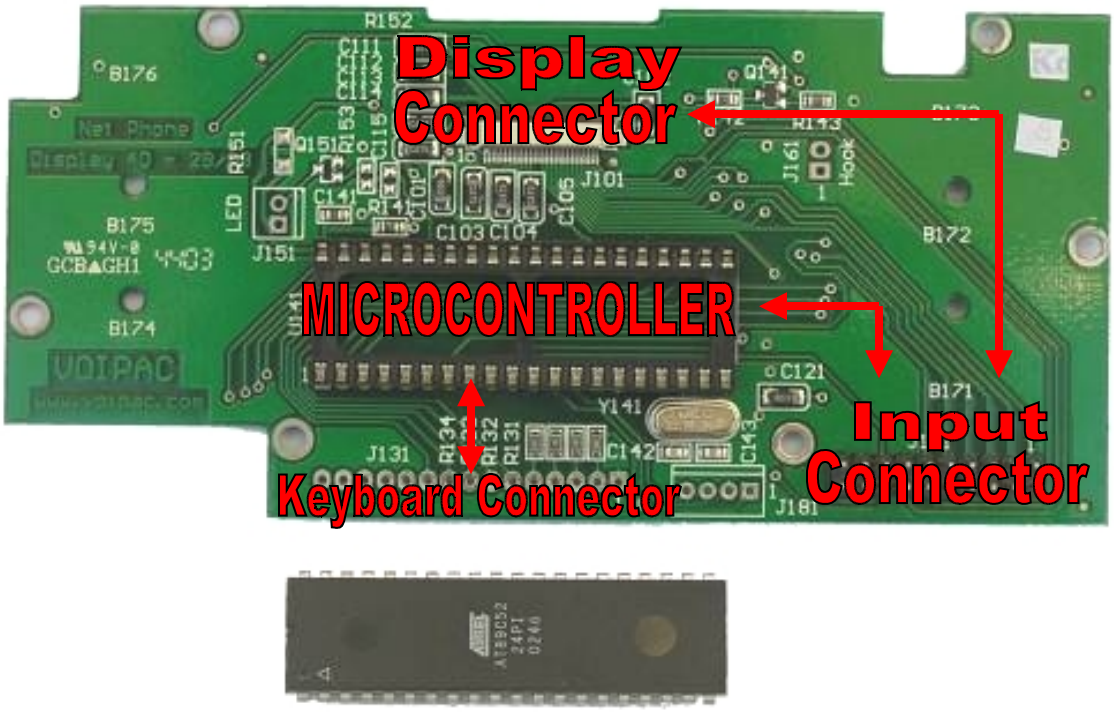


Figure 3-11 Display board picture with block description

3.4.2 Board Layout

All components are located on the top side of board.

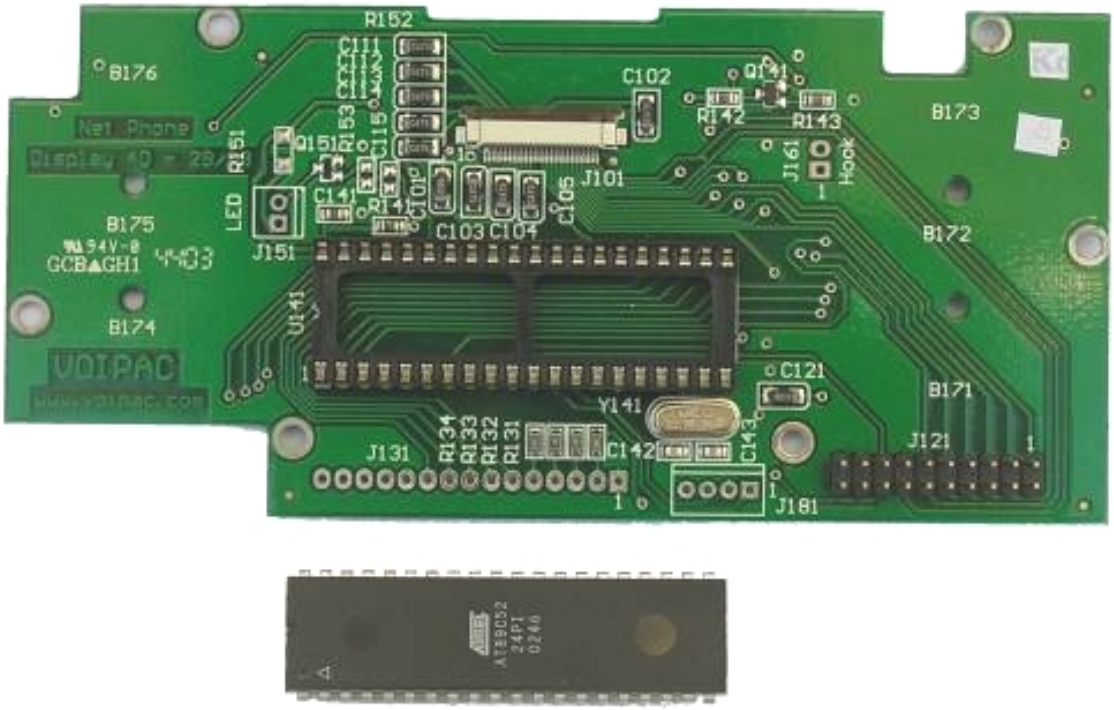


Figure 3-12 Display Board TOP VIEW

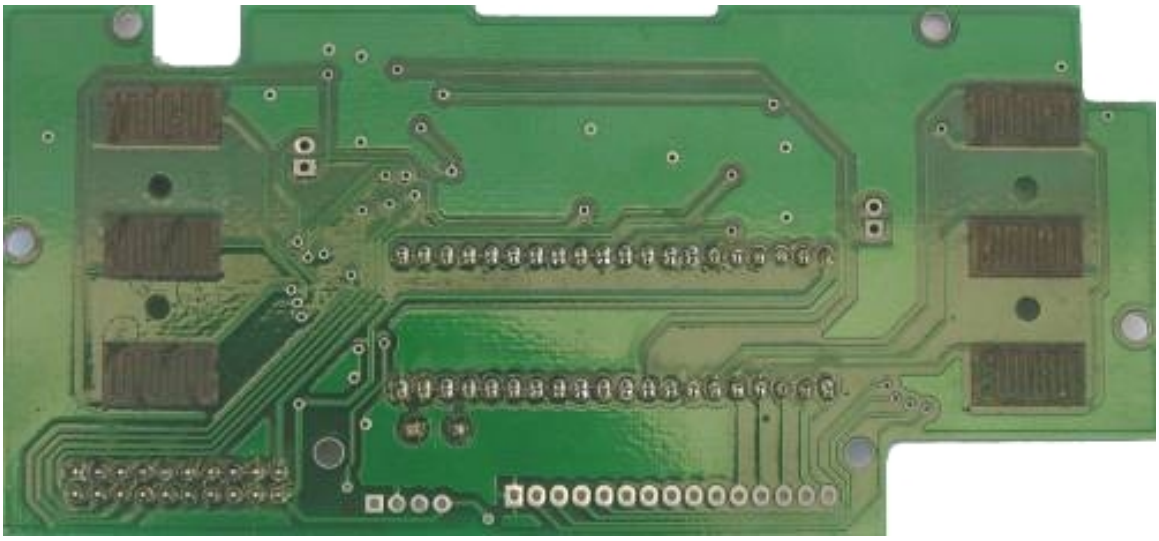


Figure 3-13 Display Board BOTTOM VIEW

3.4.2.1 Production information

PCB

- Display buttons surface is CARBONED!
- Display PCB is standard TWO layer PCB with Top Overlay description.
- Solder mask is from both sides – top and bottom.
- ATMEL microcontroller is put in socket.
- Isolated pad is used under crystal.

3.4.3 Description of basic chips on main board

ATMEL Microcontroller

The AT89C52 is a low-power, high-performance CMOS 8-bit microcomputer with 8K bytes of Flash programmable and erasable read only memory (PEROM).
For later firmware upgrade possibility is processor placed into socket.

3.4.4 Peripherals

3.4.4.1 ATMEL AT89C52 serial port

This serial port is used for communication with NP210 Main board

3.4.4.2 Display

Display signals are connected from NP210 Main board through J121 and J101 connectors to display. Between connectors are placed necessary components for properly display functionality.

Interface	Connector/Header
Display interface from NP210 Main board	J121 – DISPLAY_CONTROL_CON
Display connector	J101 – DISPLAY_CON

3.4.4.3 Backlight connector

Is not connected in basic version!
Display board supporting LED backlight connection. Then there are additional components needed: R151, R152 (optional), R153, Q151. Component values depend on used backlight.

Interface	Connector/Header
Display LED Backlight	J151 – LED_BL_CON

3.4.4.4 PS2 Keyboard connector

PS2 Keyboard connector is placed on NP210 board, but PS2 signals are connected with Display board via J181 connector. PS2 signals are processed with ATMEL microcontroller and data are sent by UART serial interface to NP210 Main board.

Interface	Connector/Header
PS2	J181 – KB_PS2_CON

3.4.4.5 Keyboard interface

Keyboard interface has support for maximum 8x7 buttons connected into matrix. Matrix 5x5 is connected with Keyboard PCB (connector J131) and 1x6 buttons are used on Display PCB. One matrix combination is used for HOOK switch connection.

Interface	Connector/Header
Keyboard	J131 – KEYBOARD_CAB

3.4.4.6 HOOK connector

PCB with Hook switch is mapped to matrix keyboard through J161 connector.

Interface	Connector/Header
HOOK connector	J161 – HOOK_HEADER_CAB

3.4.5 Others

Power Supply

Board is powered from NP210 Main board through J121 connector with 5V voltage.

Reset

For display and microcontroller reset, TLC7733 micropower supply voltage regulator placed on NP210 Main board is used.

3.4.6 Connectors

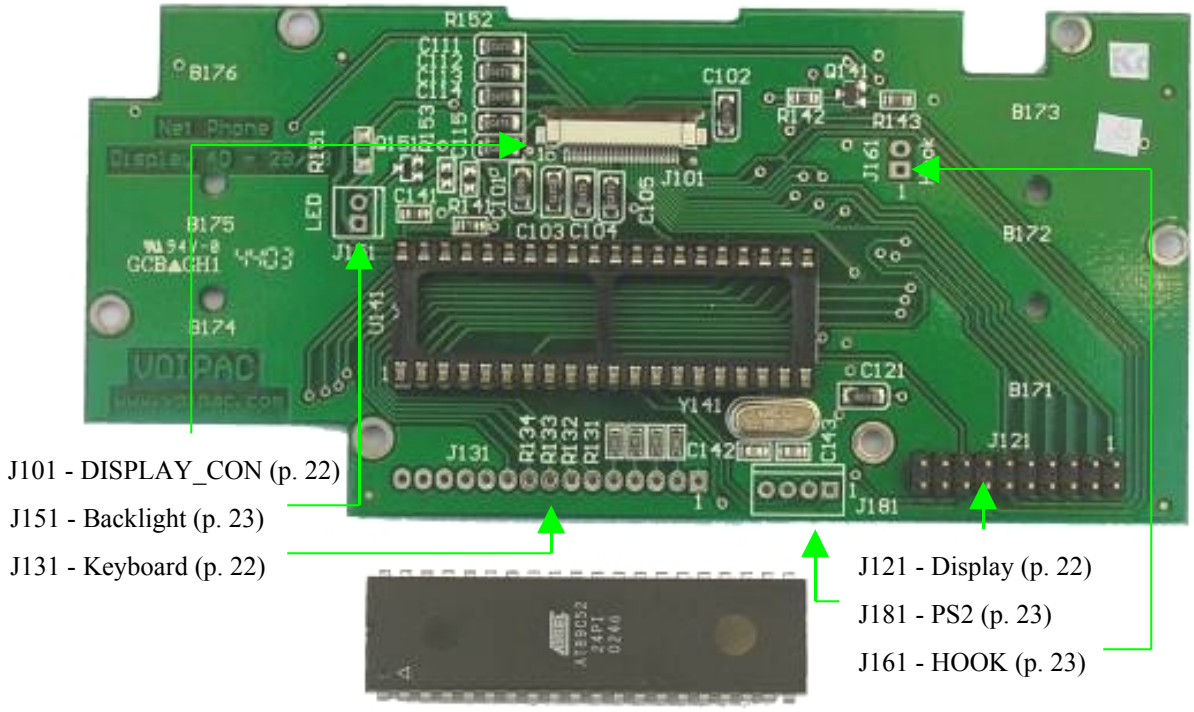


Figure 3-14 Display Board connector description

Notes:
VCC3 = 3.3V
VCC5 = 5V
NC = Not Connected

J101 DISPLAY_CON – ZIF connector for LCD display connection

Pin	Description	Pin	Description
1	CS86 (GND)	2	GND
3	V5	4	V4
5	V3	6	V2
7	V1	8	CAP2+
9	CAP2-	10	CAP1-
11	CAP1+	12	CAP3-
13	VOUT	14	GND
15	+3.3V	16	D7
17	D6	18	D5
19	D4	20	D3
21	D2	22	D1
23	D0	24	-RD
25	-WR	26	DATA/-CMD
27	-RST	28	-CS

J121 Display – this connector is used for connection with display signals from NP210 Main board

Pin	Description	Pin	Description
1	D0	2	D1
3	D2	4	D3
5	D4	6	D5
7	D6	8	D7
9	-RD	10	DATA/-CMD
11	-WR	12	-CS
13	NC	14	-RST
15	TXD	16	RXD
17	GND	18	GND
19	+3.3V	20	+5V

J131 Keyboard – this connector is used for connection with keyboard PCB via cable

Pin	Description	Pin	Description
1	GND	2	LED0
3	LED1	4	LED2
5	LED3	6	Ouput_Y0
7	Ouput_Y1	8	Ouput_Y2
9	Ouput_Y3	10	Ouput_Y4
11	Input_X4	12	Input_X3
13	Input_X2	14	Input_X1
15	Input_X0		

J151 Backlight – LED Backlight connector

Pin	Description	Pin	Description
1	A_LED+	2	K_LED-

J161 HOOK – Hook connector for cables from HOOK PCB

Pin	Description	Pin	Description
1	Ouput_Y6	2	Input_X6

J181 PS2 – PS2 connector from NP210 main board

Pin	Description	Pin	Description
1	CLK	2	GND
3	DATA	4	VCC5

3.5 Keyboard

3.5.1 Board Layout

Board has two components only – LED diodes.

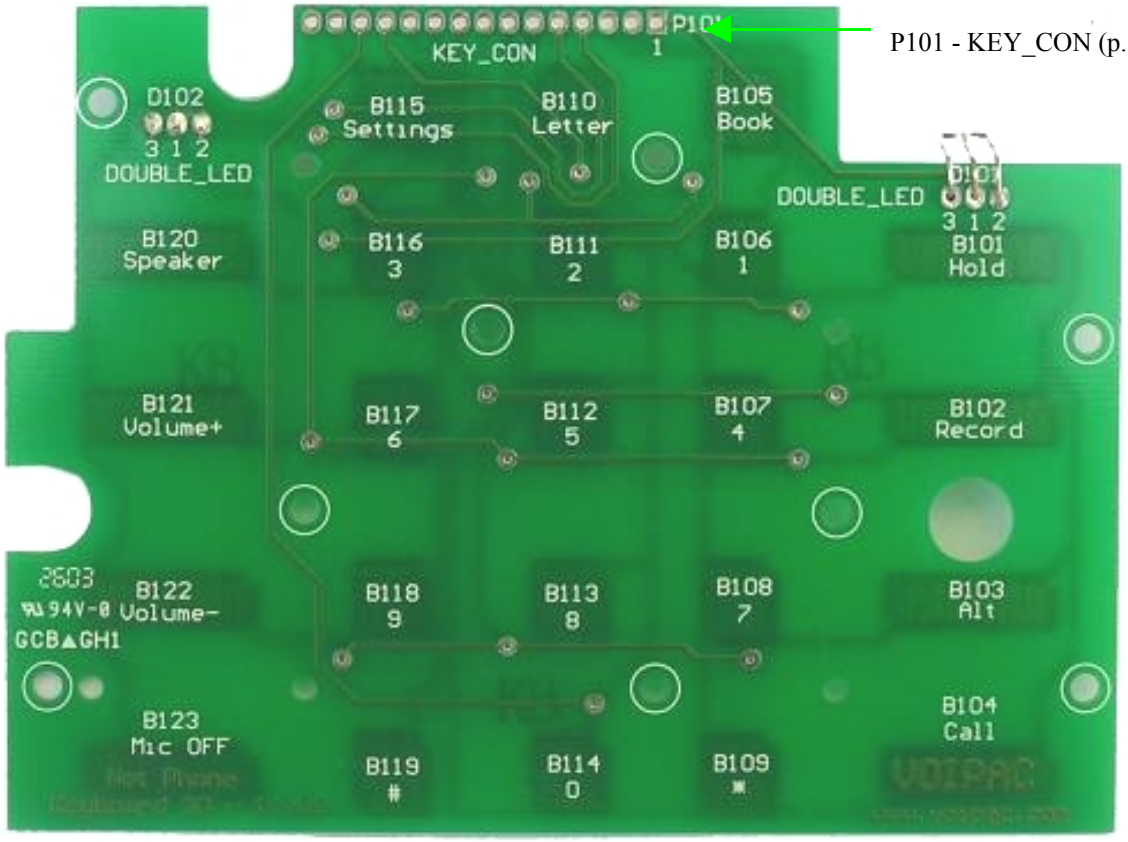


Figure 3-15 Keyboard TOP VIEW

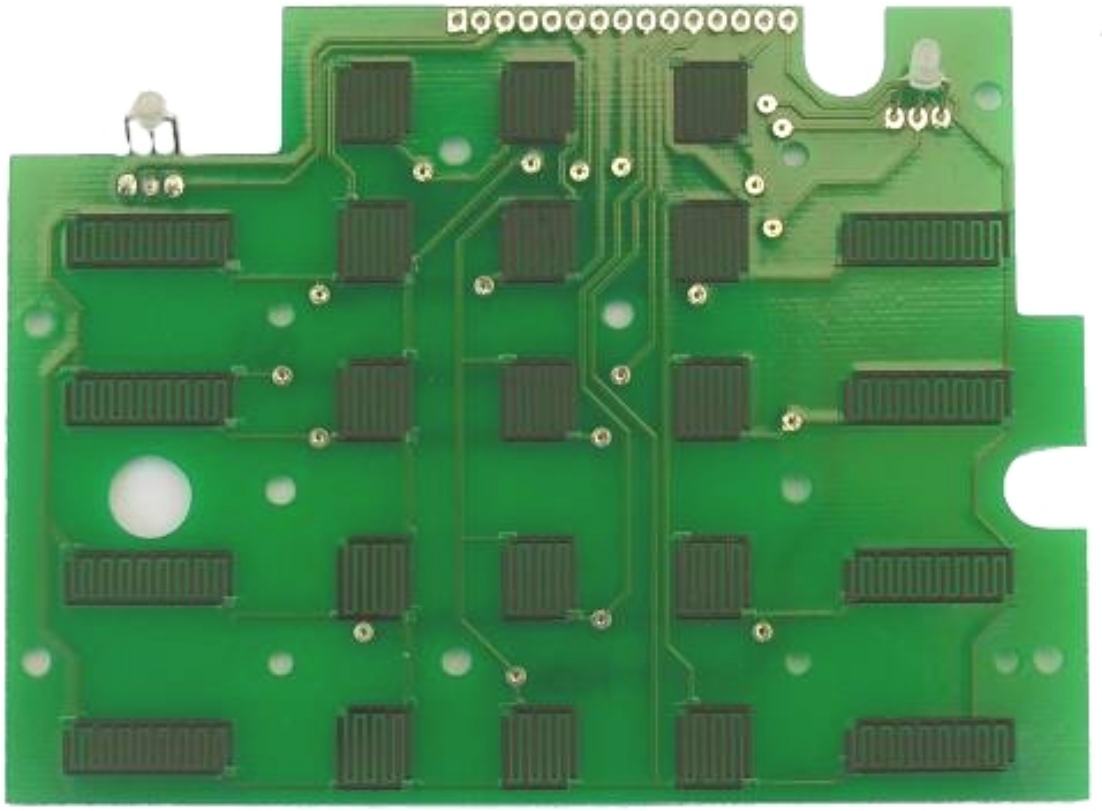


Figure 3-16 Keyboard BOTTOM VIEW

3.5.1.1 Production information

PCB

- Buttons surface is CARBONED
- Keyboard PCB is standard with TWO layers PCB and Top Overlay description.
- Solder mask is from both sides – top and bottom.
- For more about LEDs connection – see the picture.

3.5.2 Connectors

P101 KEY_CON – this connector is used for connection with keyboard PCB via cable

Pin	Description	Pin	Description
1	GND	2	LED0
3	LED1	4	LED2
5	LED3	6	Ouput_Y0
7	Ouput_Y1	8	Ouput_Y2
9	Ouput_Y3	10	Ouput_Y4
11	Input_X4	12	Input_X3
13	Input_X2	14	Input_X1
15	Input_X0		

3.5.3 LEDs

On the board, there are two red + green color LEDs for phone status signalization.

3.6 Hook

3.6.1 Board Layout

Hook is very simple PCB.

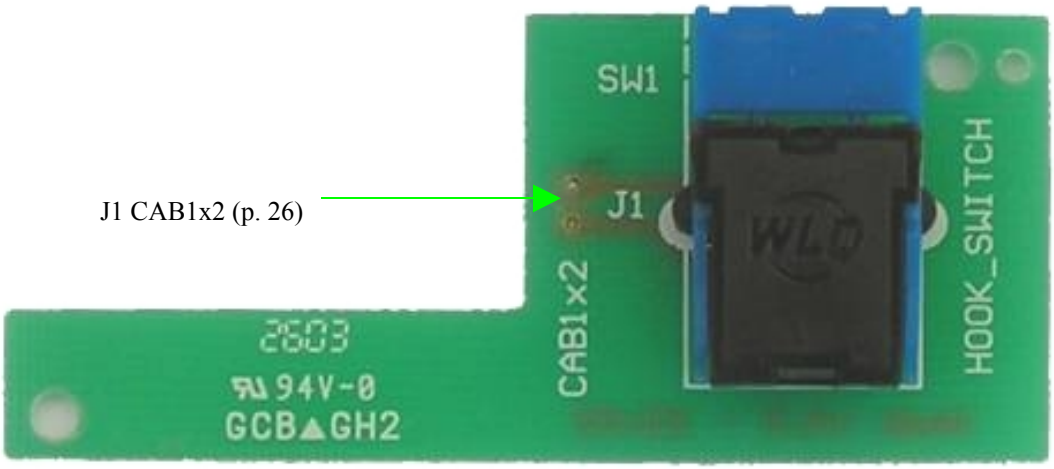


Figure 3-17 Hook TOP VIEW

Figure 3-18 Hook BOTTOM VIEW

3.6.1.1 Production information

PCB

- Keyboard PCB is standard ONE layer PCB with Top Overlay description.
- Solder mask is on bottom side.

J1 CAB1x2 – Hook connector for cables from HOOK PCB

Pin	Description	Pin	Description
1	Ouput_Y6	2	Input_X6

3.7 Other electronic components and PCBs cabling

Other electronic components used in NP210 design are:

- LCD display
- Speaker
- Microphone

PCBs cabling:

3.8 How some things works briefly

- PS2 is processed by ATMEL microcontroller
- Display is mapped directly into PXA255 memory space
- UART interface is used for communication between PXA255 and ATMEL microcontroller
- All buttons + hook switch are connected into matrix
- LCD LED Backlight can be controlled by PWM from ATMEL microcontroller
- PXA255 Serial port 0 is used for console output
- PXA255 Serial port 1 is used for communication with ATMEL microcontroller

4 Software & Development Tools

System is supplied with following software configuration:

Bootloader: Armboot (Ethernet system loading support)
Linux OS: Debian Linux ver. 2.4.19

4.1 Bootloader

4.1.1 JTAG cable

This cable is necessary for the first time loading of the program into the FLASH memory. It is used together with JflashMM program.

4.1.2 Bootloader Burning

NP210 Development kit is delivered with bootloader. If you need to change or update this software, use JTAG or Ethernet. Also, for first board flashing use JTAG cable.

The process of the programming under OS W2000:

1. Connect the JTAG cable into the computer LPT.
2. Load the parallel port driver by the command
   `instdrv giveio c:\winnt\system32\drivers\giveio.sys`
   Note:
   giveio.sys copy before first use to c:\winnt\system32\drivers\ directory
3. Connect the JTAG cable to the connector on the board (JTAG connector J321) Note: If you need to update Armboot in NP210 by JTAG, then first remove upper housing part of the phone.
4. Connect the serial cable to the NP210 RS232 output.
5. Start up the terminal (with the settings of 38400, 8, N, 1, N).
6. Connect DC adapter to the phone.
7. Program the binary file of ArmBoot program
   `jflashmm.exe pxa255 armboot200-c2i.bin P 0 INS NOD`
   and follow the listing:

```
JFLASH Version 5.01.003
COPYRIGHT (C) 2000 - 2003 Intel Corporation

PLATFORM SELECTION:
 Processor=             Cotulla
 Development System=    Lubbock
 Data Version=          1.0

Cotulla revision B2
Found flash type: 28F320J3A

Erasing block at address 0
Starting programming
Using BUFFER programming mode...
Writing flash at hex address     2500, 11.53% done
```

Wait until the software is programmed and verified.

8. Then Power OFF and Power ON the phone (Disconnect and connect DC adapter).
9. Follow the listing on the terminal.

Listing after programming and resetting of the board correctly:

```
ARMboot 1.3.0 by Voipac <www.voipac.com> (Aug 17 2005 - 20:04:09)

ARMboot code: a1000000 -> a1016ba4
CPU: Intel XScale-PXA255 (ARM 5TE) revision A0
Clock: Mem=99.53MHz (*27), Run=199.07MHz (*2), Turbo=199.07MHz (*1.0,inactive)
DRAM Configuration:
Bank #0: a0000000 64 MB
Bank #1: a4000000 0 KB
Bank #2: a8000000 0 KB
Bank #3: ac000000 0 KB
Flash: 32 MB
```

```
Hit any key to stop autoboot:  0
ArmBoot>
```

Note: The console output can be a little different – it depends on software version.

4.1.3 Armboot

Program ArmBoot is often used for software debugging. It also enables fast burning of the kernel and of the file system into FLASH memory. It is also possible to test the peripheries with its help, while it enables a direct access to the physical address of the memory.

Starting

1. Connect the serial cable to the phone.
2. Start up the terminal with settings of 38400, 8, N, 1, N. You can use for example Start - Programs - Accessories - Communication - Hyperterminal in the operational system Windows.
3. Connect the power to the phone.
4. You will see the listing of the program running.

```
ARMboot 1.3.0 by Voipac <www.voipac.com> (Aug 17 2005 - 20:04:09)

ARMboot code: a1000000 -> a1016ba4
CPU: Intel XScale-PXA255 (ARM 5TE) revision A0
Clock: Mem=99.53MHz (*27), Run=199.07MHz (*2), Turbo=199.07MHz (*1.0,inactive)
DRAM Configuration:
Bank #0: a0000000 64 MB
Bank #1: a4000000 0 KB
Bank #2: a8000000 0 KB
Bank #3: ac000000 0 KB
Flash: 32 MB
Hit any key to stop autoboot:  0

Unknown command 'FIXME' - try 'help'
ArmBoot>
```

ArmBoot - a simple description of selected instructions

env

- an extract of adjusted variables. It is necessary to control their content before you start to work with the Armboot program. Otherwise, for example, the file downloading from the Network does not have to work.

save

- new values of variables are saved in env

Instruction for saving and running of the program kernel from SDRAM memory:

tftp "zImage"

- the instruction saves the file through the network by the command tftp into the board's memory

bootz

- run the program that has been saved using the command tftp

Running of the program kernel from any part of the memory:

go 40000

- this command starts up the program from the address 0x40000

In case, you want to save e.g. file system, use the following commands:

erase 1:4-127

- this command erases the memory in sector one (usually, there is only one sector, use the value 1) from the block 4 to the block 127

tftp a2000000 file-system. bin

- it saves the file file-system.bin placed on tftp server from the address 0xa2000000 into SDRAM memory.

An example of a file transport:

```
ArmBoot> tftp a2000000 zImage
ARP broadcast 1
eth addr: 00:e0:18:c3:16:13
TFTP from server 192.168.1.76; our IP address is 192.168.1.89
Filename 'zImage'.
Load address: 0xa2000000
Loading:
#################################################################
#############################################################
done
Bytes transferred = 660364 (a138c hex)
ArmBoot>
```

Program the file into FLASH memory by the using the following way:

cp.b a2000000 100000 a138c

- this command will copy the length 0xa138c (this one is taken from the listing after data transport) by bytes from the address 0xa2000000 to the address 0x100000

You can work with the memory using the following basic commands:

md 41000000

- gives a listing of the memory content from the address 0x41000000

mw 41000000 88f

- changes the memory content in the address 0x41000000 to the value 0x88f

mtest

- this command serves as a simple checkout of a memory part

Other commands you get by a word entry:

help

- gives a list of all commands in ArmBoot program

In case you need a more detailed help for a concrete command, use for example:

help cp

- where you can entry a different command instead of cp, the one that you are interested in

Please use also the following commands by the process of the program upgrading for removement and switch on the protection of the block against deletion:

protect off

- switches off the protection of the block against deletion

protect on

- switches on the protection of the block against deletion

4.1.4 Network Flashing (kernel and file system)

Firstly we set IP address board and tftp server. Default address is 192.168.1.160 for board and 192.168.1.76 for tftp server. We can change it, but it depends on your network environment.

```
ArmBoot> setenv ipaddr 192.168.1.100
ArmBoot> setenv serverip 192.168.1.111
ArmBoot>saveenv
Un-Protected 1 sector
Saving Environment to Flash… done
Protected 1 sector
ArmBoot>
```

Use this commands for kernel and file system flashing (16MB FLASH configuration):

1. *erase 1:1-63*
2. *tftp "zImage"*
3. *cp.b a0300000 40000 xxx*
   xxx – lenght of zImage file
4. *tftp "jffs2-root.bin"*
5. *cp.b a0300000 100000 xxx*
   xxx – lenght of file sytem
6. *go 40000*
   or bootz 40000

Here you can see some screens form this process

Erasing

```
ArmBoot> erase 1:1-3
Erase Flash Sectors 1-3 in Bank # 1:
Erasing sector 1 ... ok
Erasing sector 2 ... ok
Erasing sector 3 ... ok
Done
ArmBoot>
```

Downloading

```
ArmBoot> tftp "zImage"
ARP broadcast 1
eth addr: 00:50:04:e0:31:f3
TFTP from server 192.168.1.76; our IP address is 192.168.1.160
Filename 'zImage'.
Load address: 0xa0300000
Loading: ################################
Done
Bytes transferred = 654788 (9fdc4 hex)
ArmBoot>
```

Copy into Flash

```
ArmBoot> cp.b a0300000 40000 9fdc4
Copy to Flash... 100% done.
ArmBoot>
```

4.1.5 Helpful

Very helpful is to put several commands into one line. For example:

VOIPAC development kit software upgrading with one line command:

erase 1:1-127;tftp a2000000 dk/stn/kernel_image;cp.b a2000000 40000 c0000;tftp "dk/stn/filesystem_image";cp.b a2000000 100000 1ecc000;set bootdelay 8;set bootcmd bootz 40000;save

When you need to put this line with more commands into CMD variable then the separator is:

\;

For example:
set cmd md 48000000\;go 40000

Also it is possible to control user led on board. Very helpful is TURN ON led before long commands executing and TURN OFF the led after finish. You don't need then to connect the serial console.

For example optimal setting for production:
Compile Armboot with cmd variable in this command order:
<TURN USER LED> .... COMANDS LINE (FLASH KERNEL AND FILESYSTEM) ... <TURN OFF LED>

Then:
FLASH Armboot by JTAG to FLASH, TURN OFF and TURN ON power supply, USER LEDS turn ON and after that will be kernel and filesystem automatically saved into FLASH memory. When process is successful user LED will turn OFF.

4.2 OS Linux preparation

4.2.1 TFTP server

You will need TFTP server for the file transfer from the PC to your board. Install tftp server on the Debian Linux system (use the package tftpd). The client occurs in Arm Boot program. In case you also have a client on another PC, you can test the function of tftp. Create the file tftp_test.bin in /bootdirectory. Then you can try to download:

```
if=/dev/zero of=/boot/tftp_test.bin bs=1024 count=1
# tftp localhost
tftp> get test.bin
Received 1024 bytes in 0.4 seconds
tftp> quit
```

4.2.2 Cross Compiler Installation

To be able to create binary files from source codes, that can be run on xScale processors, you need to compile them. You can use a prebuilt application from

www.arm.linux.co.org or from our server to install Cross Compiler. To install prebuilt tools use the following commands:

```
mkdir /usr/local/arm
cp cross-X.X.tar.bz2 /usr/local/arm
cd /usr/local/arm
tar -xjf cross-X.X.tar.bz2
```

Then you can set PATH to binary executable in the shell enviroment PATH=$PATH:/usr/local/arm/X.X/bin and test the compilator arm-linux-gcc hello_world.c -o hello_world

4.2.3 Preparation of OS Linux kernel source codes

It is necessary to prepare the source codes, you are going to use, before starting the work:

1. Copy the source codes of the kernel you want to use.
2. Apply relevant rmk. You can find it on: http://www.arm.linux.org.uk/developer/v2.4/ or here.
3. Apply PATCH to PXA processor.
4. Apply PATCH to Voipac hardware. You can start with patch-kernel-vpac or if you need Ethernet (chip CS8900) try patch-kernel-vpac+eth.
5. Apply PATCH to concrete hardware (if you need to use periferial on board).

This following sequence can be an example:

```
tar -xjf linux-2.4.19.tar.bz2
cp patch-2.4.19-rmk4.gz linux-2.4.19
cp diff-2.4.19-rmk4-pxa2.gz linux-2.4.19
cp diff-2.4.19-rmk4-pxa2-vpac.gz linux-2.4.19
mv linux-2.4.19 linux-2.4.19-rmk4-pxa2-vpac
cd linux-2.4.19-rmk4-pxa2-vpac/
```

Then we patch kernel.

```
gunzip patch-2.4.19-rmk4.gz
patch -p1 <patch-2.4.19-rmk4
gunzip diff-2.4.19-rmk4-pxa2.gz
patch -p1 <diff-2.4.19-rmk4-pxa2
gunzip diff-2.4.19-rmk4-pxa2-vpac.gz
patch -p1 <diff-2.4.19-rmk4-pxa2-vpac
```

Another possibility is to use simple kernel source codes from our server.

4.2.4 OS Linux kernel preparation

We compile source codes of prepared OS by following commands:

```
make menuconfig
make clean
make dep
make zImage
```

4.2.5 File system preparation

Create a root file system for arm in your own directory in OS Linux used by you. Create an image of the file system from this file system by the command mkfs.jffs2 -d adresar -e 262144 -o rootfs.jffs2.

4.2.6 Phone Starting

One of the possibilities of how OS Linux can be started on this platform: Assume that the kernel and the file system image are saved in FLASH memory. The kernel starts at the address 0x40000.

go 40000

- we can start up OS Linux from ArmBoot program by this command

After booting, we sign in by login and password (login and password supplied as a standard is root).

An example of boot log:

```
ARMboot 1.3.0 by Voipac <www.voipac.com> (Aug 17 2005 - 20:04:09)

ARMboot code: a1000000 -> a1016ba4
CPU: Intel XScale-PXA255 (ARM 5TE) revision A0
Clock: Mem=99.53MHz (*27), Run=199.07MHz (*2), Turbo=199.07MHz (*1.0,inactive)
DRAM Configuration:
Bank #0: a0000000 64 MB
Bank #1: a4000000 0 KB
Bank #2: a8000000 0 KB
Bank #3: ac000000 0 KB
Flash: 32 MB
Hit any key to stop autoboot:  0

Starting Linux kernel image at 0x40000
        Architecture type: 89
            Command line: mem=64M root=/dev/mtdblock2 rw console=ttyS0,38400
mac0=00
0d15020bb8 mac1=000d15020bb9

Uncompressing Linux..................................... done, booting
the
kernel.
Linux version 2.4.19-rmk7-pxa2-iphone (root@sarge) (gcc version 2.95.3
20010315
(release)) #4 Thu Aug 11 13:01:41 CEST 2005
Kernel_ID: N210-23329.1
CPU: XScale-PXA255 revision 6
```

```
Machine: Voipac PXA250 Developement board
Memory clock: 99.53MHz (*27)
Run Mode clock: 199.07MHz (*2)
Turbo Mode clock: 199.07MHz (*1.0, inactive)
On node 0 totalpages: 16384
zone(0): 16384 pages.
zone(1): 0 pages.
zone(2): 0 pages.
Kernel command line: mem=64M root=/dev/mtdblock2 rw console=ttyS0,38400
mac0=000
d15020bb8 mac1=000d15020bb9
Calibrating delay loop... 198.65 BogoMIPS
Memory: 64MB = 64MB total
Memory: 63408KB available (1111K code, 213K data, 56K init)
Dentry cache hash table entries: 8192 (order: 4, 65536 bytes)
Inode cache hash table entries: 4096 (order: 3, 32768 bytes)
Mount-cache hash table entries: 1024 (order: 1, 8192 bytes)
Buffer-cache hash table entries: 4096 (order: 2, 16384 bytes)
Page-cache hash table entries: 16384 (order: 4, 65536 bytes)
POSIX conformance testing by UNIFIX
Linux NET4.0 for Linux 2.4
Based upon Swansea University Computer Society NET3.039
Initializing RT netlink socket
Starting kswapd
JFFS2 version 2.1. (C) 2001 Red Hat, Inc., designed by Axis Communications AB.
pty: 256 Unix98 ptys configured
Voipac LCD driver (c) 2003
Serial driver version 5.05c (2001-07-08) with no serial options enabled
ttyS00 at 0x0000 (irq = 15) is a PXA UART
ttyS01 at 0x0000 (irq = 14) is a PXA UART
ttyS02 at 0x0000 (irq = 13) is a PXA UART
SA1100/PXA Watchdog Timer: timer margin 60 sec
eth0: cs8900 rev J found at 0xf0000300
cs89x0 media RJ-45, IRQ 37, programmed I/O, MAC 00:0d:15:02:0b:b8
eth1: cs8900 rev J found at 0xf1000300
cs89x0 media RJ-45, IRQ 42, programmed I/O, MAC 00:0d:15:02:0b:b9
ac97_codec: AC97 Audio codec, id: 0x4144:0x5360 (Analog Devices AD1885)
Probing Lubbock flash at physical address 0x00000000 (32-bit buswidth)
Using buffer write method
Using static partition definition
Creating 3 MTD partitions on "Lubbock flash":
0x00000000-0x00040000 : "Bootloader"
0x00040000-0x00100000 : "Kernel"
0x00100000-0x02000000 : "Filesystem"
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP
IP: routing cache hash table of 512 buckets, 4Kbytes
TCP: Hash tables configured (established 4096 bind 4096)
ip_conntrack (512 buckets, 4096 max)
ip_tables: (C) 2000-2002 Netfilter core team
NET4: Unix domain sockets 1.0/SMP for Linux NET4.0.
NET4: Ethernet Bridge 008 for NET4.0
NetWinder Floating Point Emulator V0.95 (c) 1998-1999 Rebel.com
VFS: Mounted root (jffs2 filesystem).
Freeing init memory: 56K
INIT: version 2.84 booting
Mounting local filesystems...
proc on /proc type proc (rw)
none on /var type tmpfs (rw)
devpts on /dev/pts type devpts (rw)
Creating /var subdirectories...
```

```
INIT: Entering runlevel: 2
Decompressing iphone...
Process ID:20
Press "Settings" to Firmware
Setting host name to `NP210'
Setting up IP spoofing protection: rp_filter.
eth1: 10Base-T (RJ-45) has no cable
eth1: using half-duplex 10Base-T (RJ-45)
eth0: using half-duplex 10Base-T (RJ-45)
done.
Fri Jan  1 00:00:00 GMT 1999
23  Aug  08:15:18  ntpdate[69]:  step  time  server  195.113.144.201  offset
209636117.7
62490 sec
Starting OpenBSD Secure Shell server: sshd.
Got signal 15.
Waiting for SIGQUIT signal with 1changed to 0
SUCCESS->END
Starting IPhone ...
At time: 0
Daemon started with pid 100
IPhone running.
Runlevel 2 READY

NP210
NP210 login: No jabber server specified!
                                        Cannot login
                                                    HTB init, kernel part
versio
n 3.6
```

Contact:

Voipac, s.r.o.
Janka Kráľa 3
911 01, Trenčín
Slovakia
Tel. +421 32 6538513, 31
Fax: +421 32 6538533
E-mail: hw@voipac.com