KwikByte KB9202 User’s Guide

Revision 1.04
October 3, 2006

Revision history

Date	Revision	Change
07/05/2005	0.9	Initial creation
08/03/2005	0.91	Minor clarifications and fixes (contributor: Mark Limbach)
08/16/2005	0.92	Added nfs mount and USB keydisk information
06/08/2006	1.01	Added preliminary information regarding “B” part suffix
06/21/2006	1.02	Corrected date and removed “Proprietary” from footer
08/18/2006	1.03	Updated crash recovery section to reference jumper table.
10/03/2006	1.04	Updated accordingly for KB9202B additions and shift to u-boot operation.

All trademarks are the property of their respective owners.

KwikByte, Inc. (referred to as “KwikByte”) reserves the right to make changes, at any time, to this document or products produced by KwikByte. KwikByte is not responsible for future availability or procurement, or obligated to produce products described or referenced, of previous revisions of documents or hardware beyond any existing agreements. KwikByte does not assume any liability arising out of the application of any product described or shown herein; nor does it convey any license under its copyrights or any rights of others. All materials provided by KwikByte are provided “as is”. The user is responsible to obtain rights and licenses as required for the particular application. The representation or examples shown in source, design, diagram, picture, or other information does not grant any freedom from claims of infringement; the information provided herein is for reference only. KwikByte expressly disclaims any warranty whatsoever pertaining to the adequacy of any such implementation, including but not limited to any implied warranties of merchantability or fitness for a particular purpose. KwikByte does not represent that devices shown or products described herein are free from patent infringement or from any other third party right. KwikByte assumes no obligation to correct any errors contained herein or to advise any user of this text of any correction if such be made. KwikByte will not assume any liability for the accuracy or correctness of any engineering or software support or assistance provided to a user. KwikByte products are not intended for use in life support appliances, devices, or systems.

TABLE OF CONTENTS

Introduction

The KB9202 is intended for use in evaluating the ARM9 core and peripherals and developing prototypes using a proven reference design platform. The standard kit includes the development board, serial cable, and distribution CD. The serial cable allows simple program download and debug support from a standard serial terminal – which is independent of the host operating system. The distribution CD includes sample source programs, board schematics, and scripts to build a cross-compiler toolchain and reference Linux kernel.

The KB9202 features:

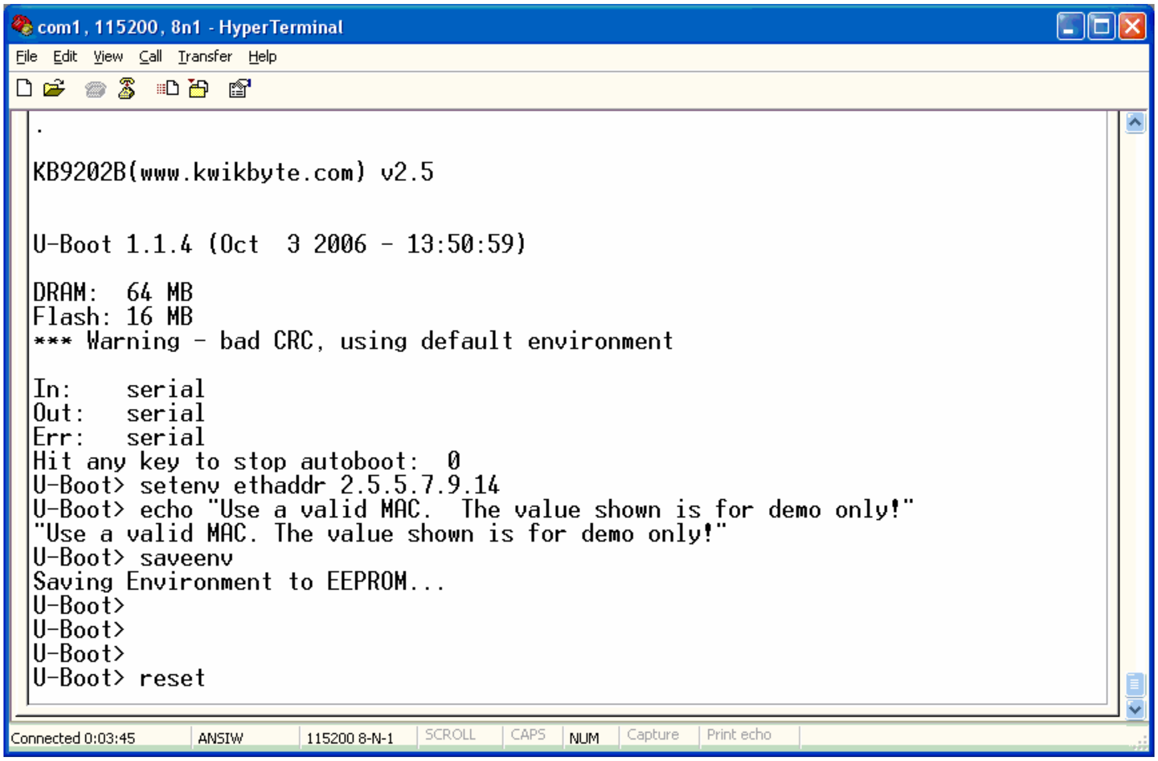
- 180 MHz, 200 MIPS ARM9 highly-integrated core (AT91RM9200)
- Independent 16KB instruction and 16KB data caches
- 32 MB SDRAM (64MB for –B)
- 16 MB parallel FLASH
- 32 MB NAND FLASH (-B only)
- Two, empty SPI serial FLASH sites (8 cason, 8-SOIC) for custom use
- SD/MMC card support
- 10/100 Ethernet
- USB 2.0 full speed host port*
- USB 2.0 full speed device port
- IrDA transceiver (optional)
- Parallel LCD interface supporting Optrex PN F-51320GNB-LW-AB or F-51320GNY-LY-AA (optional)
- RS-232 serial, dual transceiver, single DB9 connector
- RS-485 transceiver on USART 3 (-B only)
- 16KB boot EEPROM - with bootloader
- 16KB internal SRAM
- 128KB internal ROM
- 3 user LEDs
- On-board digital temperature sensor
- External memory interface
- Many signals available at 0.100" x 0.100" headers (e.g. I^{2}S, timers, counters, synchronous serial, asynchronous serial, programmable clocks, digital I/O, interrupt lines, SPI, TWI)
- Core clock up to 180MHz
- Serial boot loader for simple development
- On-chip Real-time clock (RTC) and DMA
- Switching power supply provides 2.5A and accepts wide AC/DC voltage range from standard 2.1mm jack
- JTAG compatible debug port
- 100mm x 160mm Eurocard format

Quick start

Connect the KB9202 to your host computer using the included serial cable. Open a terminal session (using Windows® Hyperterminal or Linux minicom) with the following port parameters:

Baud = 115200
Data bits = 8
Parity = none
Stop bits = 1
Flow control = none

Connect the power supply (not included) to the KB9202. The terminal window shows the initial boot process (see Figure 1). The full boot screen can be seen *here*.

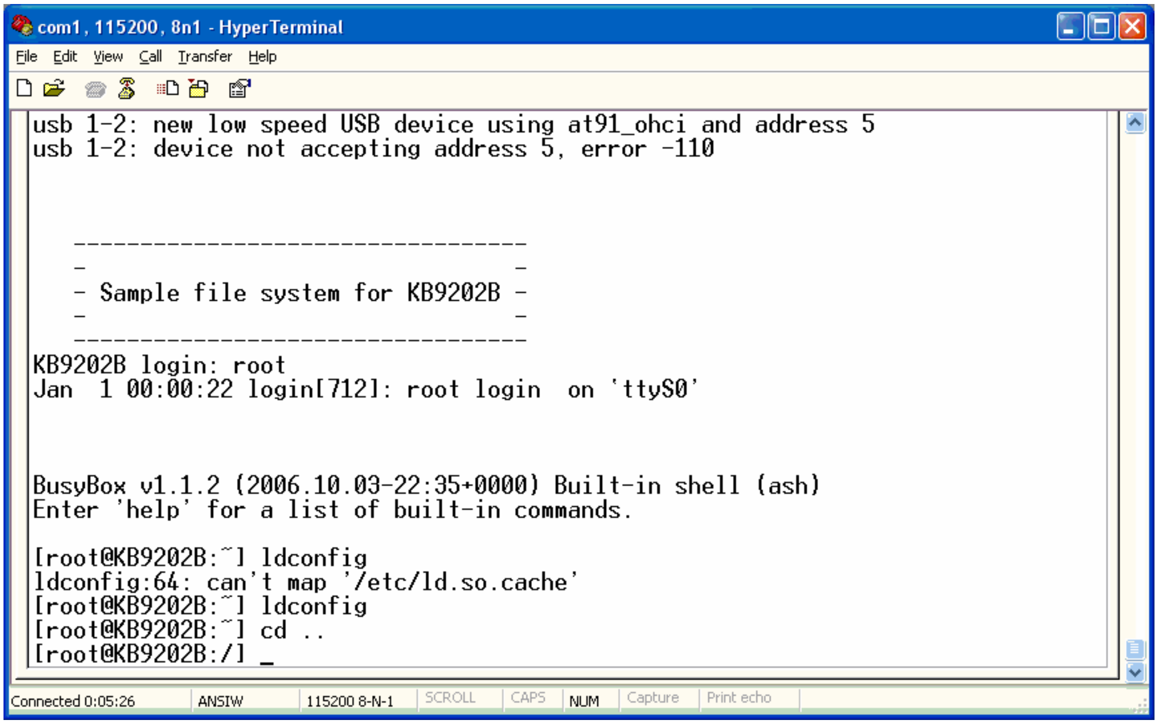

com1, 115200, 8n1 - HyperTerminal
File Edit View Call Transfer Help
Connected 0:03:45
ANSIW
115200 8-N-1
SCROLL
CAPS
NUM
Capture
Print echo

```
.

KB9202B(www.kwikbyte.com) v2.5


U-Boot 1.1.4 (Oct  3 2006 - 13:50:59)

DRAM:  64 MB
Flash: 16 MB
*** Warning - bad CRC, using default environment

In:    serial
Out:   serial
Err:   serial
Hit any key to stop autoboot:  0
U-Boot> setenv ethaddr 2.5.5.7.9.14
U-Boot> echo "Use a valid MAC.  The value shown is for demo only!"
"Use a valid MAC. The value shown is for demo only!"
U-Boot> saveenv
Saving Environment to EEPROM...
U-Boot>
U-Boot>
U-Boot>
U-Boot> reset
```

Figure 1: Initial bootscreen from terminal

Press the reset switch (S1). Then, press a key to abort the auto boot sequence (before the auto bootsequence begins). Set the MAC address for the board using your custom value. The MAC is a unique address assigned to each device operating on the network. The values shown are used as an example only. You must obtain a MAC for your KB9202 board. For development, most customers use a MAC from an old netcard no longer in use.

After the reset operation completes, login as ‘root’ and execute the ‘ldconfig’ command to resolve shared library mappings (see Figure 2). This operation is only required once; although repetition will not cause any damage. The ‘ldconfig’ command is also used when user-built shared libraries are utilized.



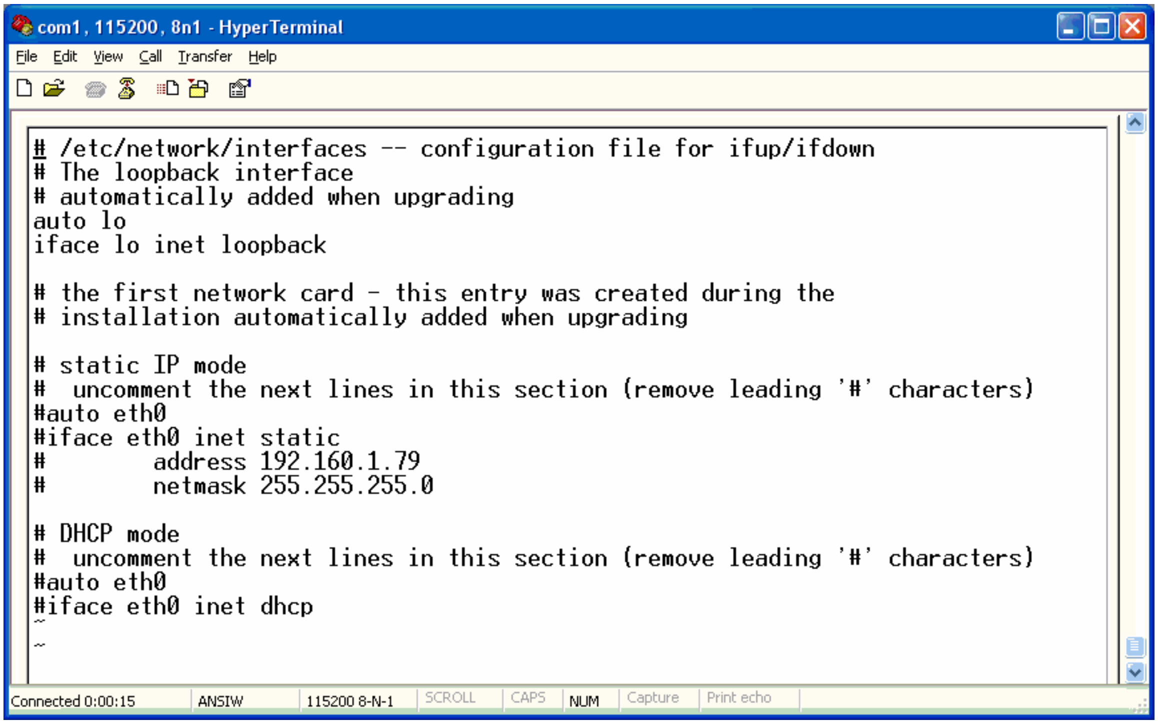
```
usb 1-2: new low speed USB device using at91_ohci and address 5
usb 1-2: device not accepting address 5, error -110



   ----------------------------------
   -                                -
   - Sample file system for KB9202B -
   -                                -
   ----------------------------------
KB9202B login: root
Jan  1 00:00:22 login[712]: root login  on 'ttyS0'



BusyBox v1.1.2 (2006.10.03-22:35+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

[root@KB9202B:~] ldconfig
ldconfig:64: can't map '/etc/ld.so.cache'
[root@KB9202B:~] ldconfig
[root@KB9202B:~] cd ..
[root@KB9202B:/] _
```

Figure 2: Resolve shared library mapping

Next, set the IP address according to your network. This will typically be one of two values:

1) DHCP
2) Fixed IP address

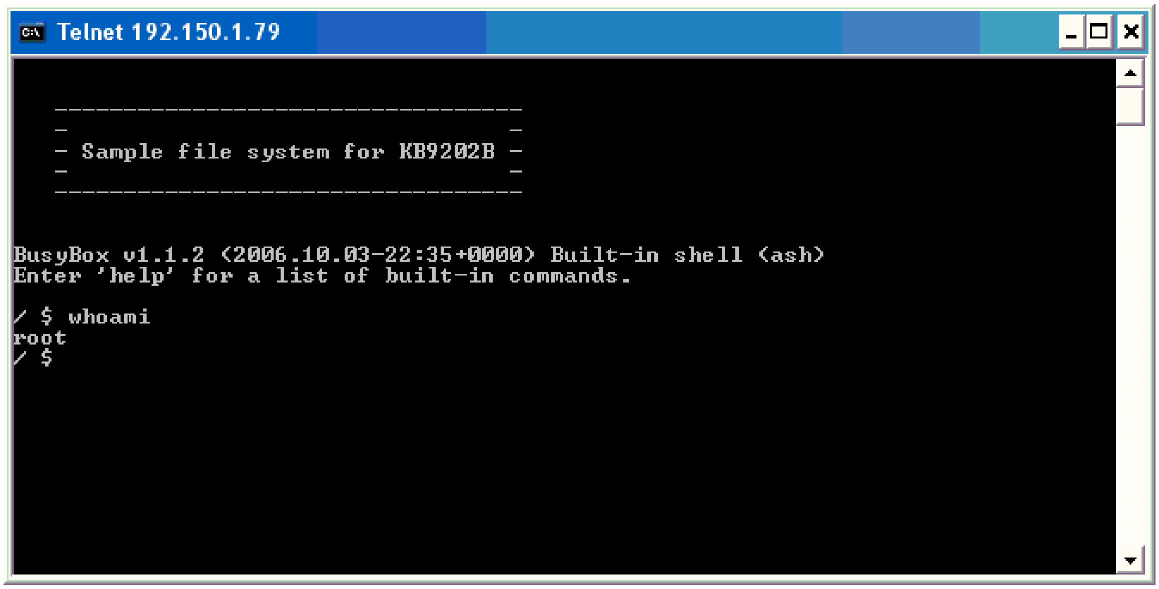
The IP is set in the file `/etc/network/interfaces`. Display the contents of the file with ‘cat /etc/network/interfaces’ to check the current setting. Edit the file using ‘vi /etc/network/interfaces’ (see Figure 3). If you are not familiar with the vi editor, use the arrow keys to move, ‘a’ key to enter edit mode, Backspace key to remove characters. The ESC key to leaves edit mode (returns to command mode). In command mode, save the changes with ‘:wq’ or exit without saving changes with ‘:q!’ followed by the Enter key.


com1, 115200, 8n1 - HyperTerminal
File Edit View Call Transfer Help
Connected 0:00:15
ANSIW
115200 8-N-1
SCROLL
CAPS
NUM
Capture
Print echo

```
# /etc/network/interfaces -- configuration file for ifup/ifdown
# The loopback interface
# automatically added when upgrading
auto lo
iface lo inet loopback

# the first network card - this entry was created during the
# installation automatically added when upgrading

# static IP mode
#  uncomment the next lines in this section (remove leading '#' characters)
#auto eth0
#iface eth0 inet static
#        address 192.160.1.79
#        netmask 255.255.255.0

# DHCP mode
#  uncomment the next lines in this section (remove leading '#' characters)
#auto eth0
#iface eth0 inet dhcp
~
~
```

Figure 3: Editing the IP address using vi

Press the reset switch and allow the auto boot sequence to complete. After the boot is complete, login as 'root'. At this point, you can telnet to the target (see Figure 4). Notice, in this screenshot, the target has been set differently than the other examples (192.150.1.79). It is recommended to follow the example installation and set the target with 192.160.1.79.



```
   -----------------------------------
   -                                 -
   - Sample file system for KB9202B -
   -                                 -
   -----------------------------------


BusyBox v1.1.2 (2006.10.03-22:35+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

/ $ whoami
root
/ $
```

Figure 4: Telnet to target

Test the operation using command Linux commands: e.g., "ps –ef", "mount", "df –h", etc. Now, start developing your applications! See *Practical Software Development for KB9202B* (freely available for download) for information on restoring the factory configuration, constructing a Linux-hosted development system, building sample applications, mounting removable media, and other useful information.

Notice, this same command sequence can be executed in other operating system environments (i.e., Linux) using typical terminal programs such as minicom, etc.

If using minicom, it is recommended that you disable modem commands and adjust/save the default configuration according to your system.

Jumpers and header information

See Table 1 for a description of the jumpers on the KB9202.

KB9202 Jumper	KB9202B Jumper	Description
JP1/JP2	JP4/JP6	Shorting pins 1-2 enable debug serial signals on the DB9 connector. Shorting pins 2-3 enable serial channel 0 (Rx0/Tx0) on the DB9 connector.
JP3	JP2	Shorting pins 1-2 sets BMS for parallel FLASH operation. This mode is not supported.
JP4	JP3	Shorting pins 1-2 alters the EEPROM address, effectively disabling the EEPROM from the boot sequence.
JP5	JP10	This jumper is used for convenient access to the serial channel 1 (Rx1/Tx1) signals and power/ground signals.
JP6	JP1	Shorting pins 1-2 has no effect. Shorting pins 2-3 enables the +5V supply to the USB host port.
JP7	JP8	This jumper is used to steer the SPI CS0 signal between the two SPI FLASH sites. These sites are not populated, by default.
JP8	JP9	Shorting pins 1-2 disable the IrDA transceiver.
JP9	JP5	This jumper provides convenient access to the debug serial signals.
	JP7	RS-485 differential signals wired to transceiver.

Table 1: Jumper definition

The following tables describe the signals available at the header sites:

PIN	SIGNAL	FUNCTION	NOTE
1	GND		
2	JTAGSEL		pull-down
3	+3.3V		
4	TDI		pull-up
5	TMS		pull-up
6	TDO		pull-up
7	TRSTn		pull-up
8	TCLK		pull-up

9	GND		
10	RSTn		

Table 2: Header SV1 description

PIN	SIGNAL	FUNCTION	NOTE
1	D27	databus/PC27	
2	PC10	NCS4/CFCS	
3	D26	databus/PC27	
4	PC11	NCS5/CFCE1	
5	D25	databus/PC25	
6	PC12	NCS6/CFCE2	
7	D24	databus/PC24	
8	PC13	NCS7	

Table 3: Header SV2 description

PIN	SIGNAL	FUNCTION	NOTE
1	PA24	SCK2/PCK1	
2	PA23	TXD2/IRQ3	
3	PA22	RXD2/TIOB2	
4	PA21	RTS0/TIOA2	
5	PA20	CTS0/TIOB1	
6	PA19	SCK0/TIOA1	

Table 4: Header SV4 description

PIN	SIGNAL	FUNCTION	NOTE
1	PA26	TWCK/IRQ1	TWI EEPROM/temp sensor
2	PA25	TWD/IRQ2	TWI EEPROM/temp sensor
3	GND		
4	+3.3V		
5	PB11	RF1/TIOB5	
6	PB10	RK1/TIOA5	
7	PB9	RD1/TIOB4	
8	PB8	TD1/TIOA4	
9	PB7	TK1/TIOB3	
10	PB6	TF1/TIOA3	

Table 5: Header SV5 description

PIN	SIGNAL	FUNCTION	NOTE
1	PA18	RXD0/TIOB0	
2	PA17	TXD0/TIOA0	
3	PA6	NPCS3/RXD3	
4	PA5	NPCS2/TXD3	
5	PA4	NPCS1/PCK1	
6	PA3	NPCS0/IRQ5	
7	PA2	SPCK/IRQ4	
8	PA1	MOSI/PCK0	
9	PA0	MISO/PCK3	
10	+3.3V		
11	GND		

12	GND		

Table 6: Header SV6 description

PIN	SIGNAL	FUNCTION	NOTE
1	BD0		BD = buffered data
2	BD1		
3	BD2		
4	BD3		
5	BD4		
6	BD5		
7	BD6		
8	BD7		
9	GND		
10	GND		
11	BD8		
12	BD9		
13	BD10		
14	BD11		
15	BD12		
16	BD13		
17	BD14		
18	BD15		
19	+3.3V		
20	+3.3V		

Table 7: Header SV7 description

PIN	SIGNAL	FUNCTION	NOTE
1	BA0		BA = buffered address
2	BA1		
3	BA2		
4	BA3		
5	BA4		
6	BA5		
7	BA6		
8	BA7		
9	GND		
10	GND		
11	BA8		
12	BA9		
13	BA10		
14	BA11		
15	BA12		
16	BA13		
17	BA14		
18	BA15		

Table 8: Header SV8 description

PIN	SIGNAL	FUNCTION	NOTE
1	BA16		BA = buffered address
2	BA17		

PIN	SIGNAL	FUNCTION	NOTE
3	BA18		
4	BA19		
5	BA20		
6	BA21		
7	BA22		
8	BA23		
9	GND		
10	GND		
11	BCS0n		
12	BCS2n		
13	BCS3n		
14	BOEn		
15	BWEn		
16	BBS1n		
17	BBS3n		
18	BRSTn		

Table 9: Header SV9 description

KB9202 Header Reference	KB9202B Header Reference
SV1	SV8 (standard 20-pin JTAG)
SV2	SV1
SV4	SV5
SV5	SV6
SV6	SV3
SV7	SV2
SV8	SV4
SV9	SV7

Table 10: KB9202B Header Naming Convention

The following signals are also used on-board:

IrDA transceiver	PB20/PB21 (TXD1/RXD1)
USB device port Rpu enable switch	PB22 (SCK1)
USB device port +Vbus detection	PB24 (CTS1)
First-half of dual serial port	PA30/PA30 (DRXD-CTS2/DTXD-RTS2)
Second-half of dual serial port	PA17/PA18 (TXD0-TIOA0/RXD0-TIOB2)
User LEDs	D18/D19/D20 (PC18/PC19/PC20)
LCD Reset	D22 (PC22)
LCD Backlight	D23 (PC23)
RS-485 transceiver (KB9202B)	PA5/PA6 (TXD3/RXD3)
RS-485 Tx/Rx enable	PB0/PB1

Detailed peripheral information

The following sections provide important information for selected peripherals.

SDRAM

The 32 MB SDRAM is allocated at address 0x20000000. The device(s) are configured during the auto-boot operation (64 MB SDRAM on KB9202B).

Parallel FLASH

The 16 MB parallel FLASH is allocated at address 0x10000000. The device(s) can be erased and programmed using the ramMonitor utility (see ramMonitor).

NAND FLASH

The 32 MB NAND FLASH (KB9202B) is accessed via the processor’s NAND flash controller and is supported in the Linux kernel.

USB full speed host port

This device is controlled by the internal registers of the AT91RM9200. The power available (+5VDC) for USB host port does not provide over-current detection/prevention as specified in the USB spec. The supply is internally protected against over-current conditions and can optionally be disabled via jumper (see Jumpers and header information).

LCD interface (optional)

The LCD control and data registers are allocated at addresses 0x30000000 and 0x30010000, respectively. The backlight can be enabled/disabled in software on pin PC23. The LCD is available externally using Optrex part number F-51320GNB-LW-AB or F-51320GNY-LY-AA.

Temperature sensor

The on-board temperature sensor is accessed on the TWI (two-wire interface) at device address 0x49. The device provides a ±2º C accuracy.

Power supply

The on-board switching power supply provides +3.3VDC at a minimum of 2.5A for installed devices and expansion devices. The circuit includes a rectifier to eliminate the concern on the polarity of external plug. Many standard (2.1mm x 5.5mm female jack) AC or DC supplies can be used to power the board.

Parameter	**AC**	**DC**
Voltage, min	9 VAC	11 VDC
Voltage, max	18 VAC	22 VDC
Current, min	300 mA	300 mA

Table 11: Power requirements

Notice, the values provided are not related; i.e., the DC requirements are not based on the AC values. These values are calculated for on-board devices only and do not include power-supply budget for user-added devices.

It is also possible to power the device from battery or a lower-voltage source by skipping the rectifier. The switching supply requires 5 V minimum, but will operate at reduced efficiency at this lower voltage.

Suitable power supplies are readily available. For example, power supplies have been tested from an original Nintendo® game system, laptop, other lab boards, and home phone set. If a suitable supply is not readily available, one can be purchased from various sources. The test system uses 12VAC, 60Hz, 1000mA supplies from Jameco (PN: 10081(CC) selling for $5.20 at the time of this writing).

Boot sequence

By default, the KB9202 copies the bootloader from EEPROM to internal SRAM and transfers execution to the image in SRAM. The bootoloader performs the following:

1) Configure basic hardware such as debug serial port, SDRAM, and parallel FLASH.
2) Execute the u-boot bootloader in FLASH.

The u-boot bootloader is currently supported by SourceForge.net. This open, standard loader performs network boot and transfers as well as serial transfers.

ramMonitor

NOTE:

The u-boot program has been adapted for KB9202B. As a result, ramMonitor is not longer necessary. Instead, it is only useful for testing, crash-recovery, or to reset the factory default installation.

A utility is provided with the KB9202 for use in debug, development, and test. This utility is called "ramMonitor". The program provides a simple command-line interface to download programs/images, update FLASH, and verify peripherals.

The ramMonitor can be stored in FLASH and executed there, copied to SDRAM and executed there, or downloaded (TFTP or serial) to SDRAM. In the first case, the program copies itself to SDRAM and transfers execution there.

Notice, the ramMonitor support SPI read/write operations which have been deprecated in the KB9202. As a result, the following power-up message will be shown because the SPI devices are not installed:

```
Entry: RAM Monitor v2.5.0
 Unexpected SPI FLASH status: 0xFF
```

Download

Currently, two methods are available to download programs/images from within ramMonitor: serial port and Ethernet.

Serial download

The serial port can be used to download programs/images in bootloader and ramMonitor modes. In both cases, the "x" command is used to transfer binary data via serial port to the board using X-modem protocol. This transfer mode is supported in Windows® terminal and Linux "download" programs. The Linux "download" is a custom, stand-alone program provided with the distribution CD. Because data is transferred at 115200 bps, large images require significant time to download. As an approximation, a 1 MB compressed Linux kernel downloads over serial in about 1.5 minutes.

To perform a serial download perform the following steps:

1) x <destination address>
2) Exit the connection to the board
3) ./download <file>
4) Open the connection

The status of the previous transfer can be viewed using the "x" command without any additional parameters (ramMonitor mode only). The following output shows a failed transfer or no transfer attempted condition.

```
>x
```

```
 Local buffer available of size: 4194304 bytes (4MB) at address: 0x2000B494

 Last x-modem transfer: FAIL (or not initiated)
```

Ethernet download (TFTP)

The Ethernet port can be used to download programs/images in bootloader and ramMonitor modes. In both cases, the “TFTP” command is used to transfer binary data via Ethernet port using the TFTP protocol. A TFTP server must be operational on the network to provide the request data. Configuration of the TFTP server is beyond the scope of this document. In order to complete a TFTP download, the following actions must be taken (in order):

1) Set the MAC address of the KB9202 (“m # # # # # #”).
2) Set the IP address of the KB9202 (“ip # # # #”).
3) Set the IP address of the TFTP server (“server_ip # # # #”).
4) Execute the transfer (“TFTP <destination address> <image name>”).

The status of the previous transfer can be viewed using the “TFTP” command without any additional parameters. The following output shows a failed transfer or no transfer attempted condition.

```
>tftp

 -- Last TFTP transfer info --
    address: 0x00000000
    size: 0x00000000
```

ramMonitor commands

Each of the following commands is case-sensitive and expected to be used in lower-case. Unless otherwise stated, all address values are interpreted as hexadecimal and the leading “0x” characters are optional. Many of the commands used in bootloader operation are also available in ramMonitor mode. The description of those commands is not repeated here.

? | h | help

These commands provide a limited help menu displaying supported commands and basic usage information.

bus

The bus command is used to display the processor memory bus speed.

Format: `bus`

c (Copy)

The copy command is used to copy data from one location in memory to another location. Typically, this command is used to transfer an image from FLASH to SDRAM.

Format: `c <destination address> <source address> <size in bytes>`

compare

The compare operation is used to byte-wise compare two sections of memory. This command is particularly useful in verifying FLASH update operations.

Format: `compare <start address 1> <start address 2> <size in bytes>`

e (Execute at address)

The execute command is used to begin execution of an image. The image can be loaded in RAM, SDRAM, or FLASH. Typically, the execute command is used to start the Linux kernel in FLASH.

Format: `e <execution address>`

eeread

The eeread command is used to copy the contents of EEPROM to SDRAM. The SDRAM can then be read to determine the contents of the EEPROM.

Format: `eeread <EEPROM address> <destination address> <size in bytes>`

eewrite

The eewrite command is used to modify the contents of EEPROM using data from SDRAM.

Format: `eewrite <EEPROM address> <source data address> <size in bytes>`

f (FLASH operations)

The "f" commands are used to perform FLASH operations.

ERASE

Format: `f e`
Operation: Unlocks protect bits and erases entire FLASH device.

Format: `f e <start address> <end address>`
Operation: Erases the sectors containing the addresses specified.

PROGRAM

Format: `f p <FLASH start address> <source data start address> <size in bytes>`
Operation: Program FLASH with the contents of memory specified.

ip (Set local IP address)

The ip command sets the IP address of the KB9202. The format is in decimal and does not include any "." demarcation characters.

Format: `ip # # # #`
Example: `ip 192 168 1 20`

irda

The irda command is used to verify basic operation of the IrDA transceiver. In order to test duplex operation, another receiving end is required; most applications will report a failure from this operation. However, typical remote control devices can be used to test the receive operation of the KB9202.

Format: `irda`

lcd

The lcd command is used to cycle test patterns on the LCD display as well as control the backlight.

Format: `lcd`
Operation: Cycles through preset display patterns on the LCD.

Format: lcd <on | off>
Operation: Enables or disables the LCD backlight.

led

The led command is used to enable/disable the user LEDs.

Format: led <0 – 7>
Operation: Displays the binary pattern specified on the LEDs.

m (Set local MAC address)

The MAC command is used to set the Ethernet MAC value of the KB9202. The format is in hexadecimal and does not include any “.” demarcation characters.

Format: m # # # # # #
Example: m 2 4 6 8 a 12

You must supply a valid, unique MAC address for the Ethernet to function properly. The value shown is for example purpose only and should not be used.

proc

The proc command is used to display the processor clock frequency.

Format: proc

read

The read command is used to display the contents of memory.

Format: read <access size> <start address> <size in bytes>
Example: read 4 0x10000000 0x100

server_ip (Set server IP address)

The server_ip command sets the IP address of the TFTP server on the network. The format is in decimal and does not include any “.” demarcation characters.

Format: server_ip # # # #

Example: server_ip 192 168 1 24

temp

The temp command is used to test the on-board temperature sensor.

Format: `temp`

tftp (Perform TFTP download)

The TFTP command is used to execute a TFTP download operation.

Format: `tftp <destination address> <image name>`

uhp

The uhp command is used to verify USB host-device loopback operation. A loopback cable must be attached between the host and device ports for this operation to complete.

Format: `uhp`

write

The write command is used to modify memory.

Format: `write <access size> <address> <value>`

x (Perform X-modem download)

The x-modem command is used to execute a serial, X-modem download operation.

Format: `x <destination address>`

Crash recovery

See the section Restoring the Factory Installation in *Practical Software Development for KB9202B*.

Schematics

See *KB9202 schematics* (freely available for download).