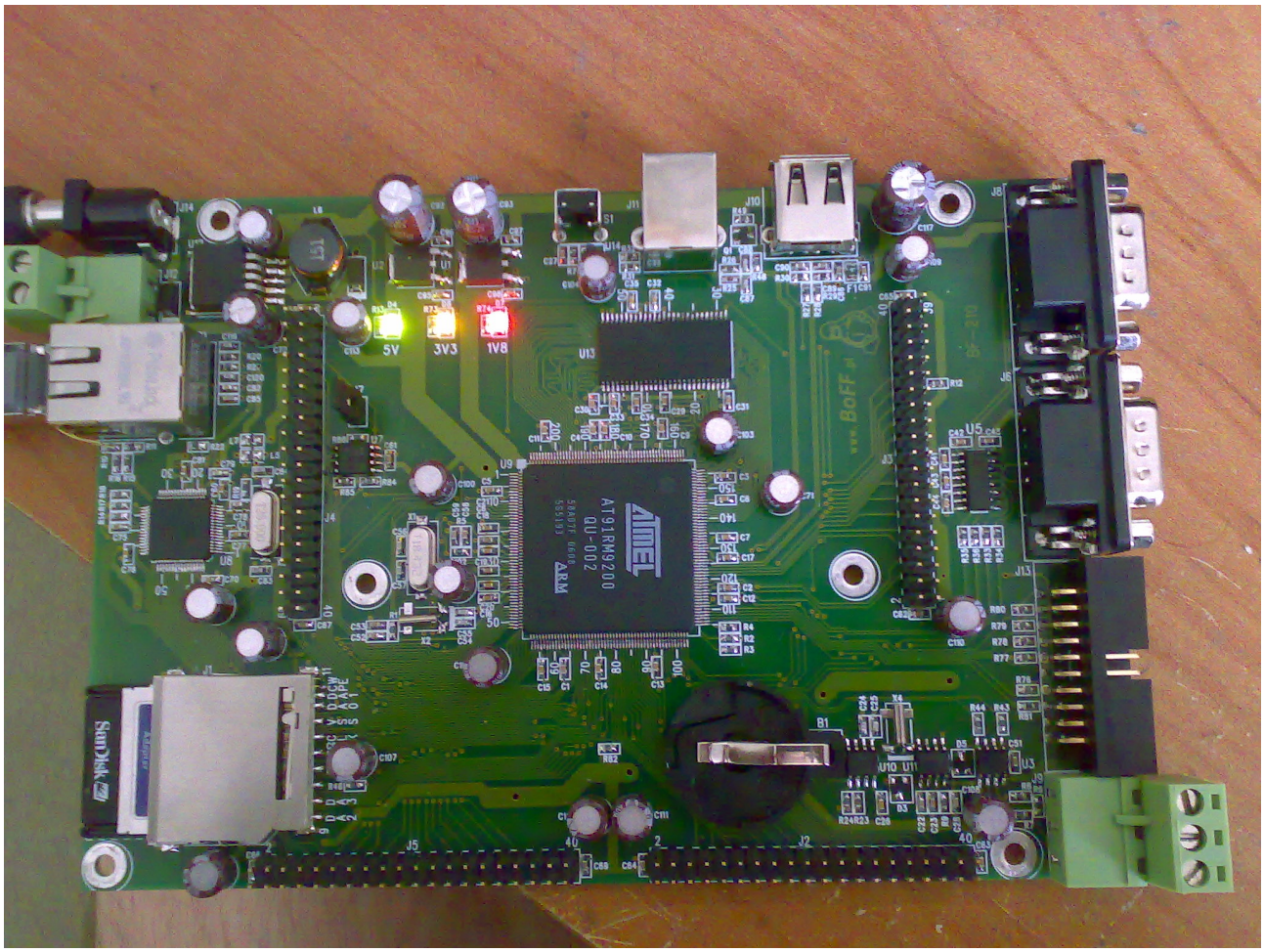
BF210 – Linuksowy ARMputer z procesorem AT91RM9200

Instrukcja obsługi



Spis treści

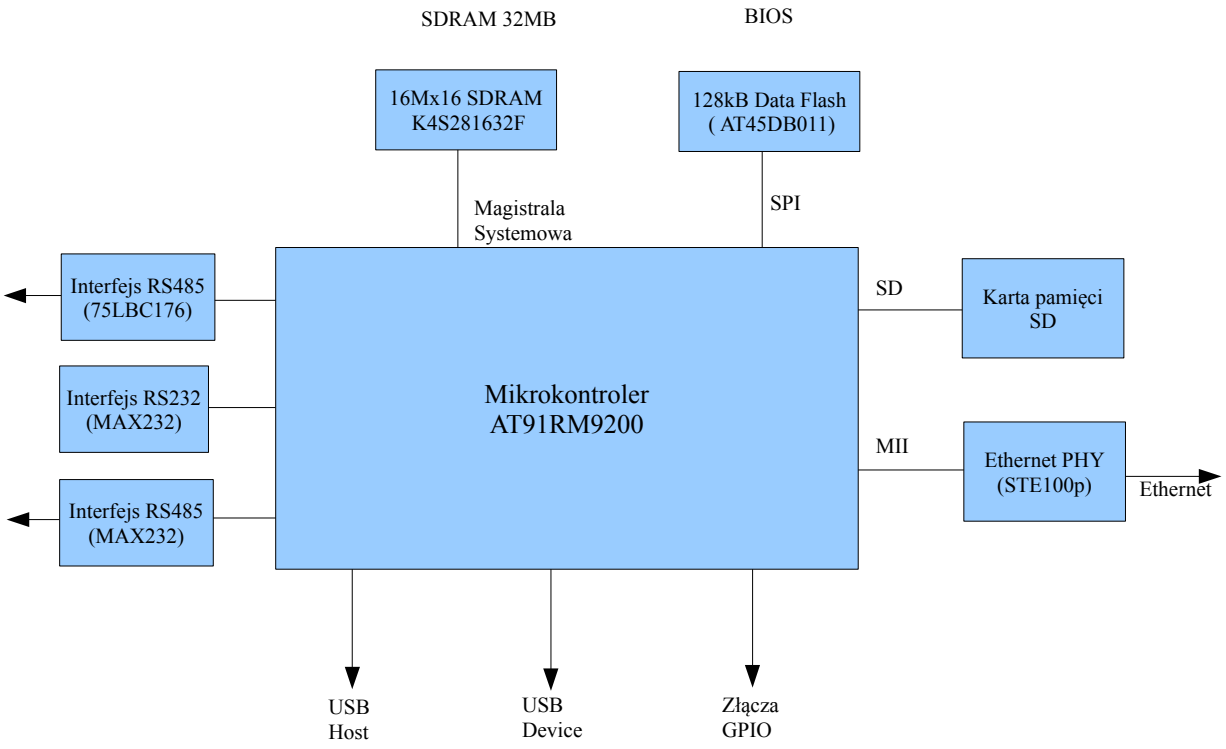
1. Opis urządzenia

Linuksowy ARMputer **BF-210** jest następcą ARMputera **BF-100** zaprezentowanego na łamach czasopisma Elektronika Praktyczna (EP3/2008). Dzięki wydajnemu 32-bitowemu układowi z rdzeniem **ARM920T**, dużej 32MB pamięci **SDRAM**, uzyskujemy wysoką wydajność niemożliwą do osiągnięcia za pomocą klasycznych mikrokontrolerów z rdzeniem **ARM7TDMI-S**. Wykorzystanie systemu operacyjnego linux pozwala na pisanie zaawansowanych aplikacji w krótkim czasie, dzięki wielkiej bazie otwartego oprogramowania „*Open Source*". Wszelkie skomplikowane zadania takie jak obsługa sieci **TCP/IP** obsługa interfejsów **USB**, których realizacja z wykorzystaniem mikrokontrolera jednoukładowego jest skomplikowana i czasochłonna, staje się banalnie prosta dzięki zastosowaniu ARMputera **BF-210**. Będący częścią układu blok zarządzania pamięcią **MMU** (*Memory Management Unit*) zapewnia niezależną pamięć wirtualną dla każdego procesu, przez co urządzenie wykorzystujące ARMputer **BF-210** jest dużo bardziej niezawodne, gdyż błędne działanie jednej aplikacji nie jest w stanie zawiesić działania całego systemu. Do urządzenia dostarczana jest płyta DVD z oprogramowaniem, która zawiera cały toolchain (kompilatory), umożliwiająca pisanie własnego oprogramowania, oraz podstawowa dystrybucja linuksa, stanowiąca bazę dla własnych projektów.

Armputer **BF-210** charakteryzuje się następującymi parametrami technicznymi:

Parametry techniczne ARMputera **BF-210**
- Procesor: ARM920T 180MHz z MMU (AT91RM9200)
- Pamieć operacyjna: 32MB SDRAM
- Pamieć BIOS: 128KB Serial Data Flash (bootloader)
- Dysk: Dowolna karta SD/MMC/SDHC 256MB-32GB
- Kontroler hosta USB 2.0 Full Speed
- Kontroler urzadzenia USB 2.0 Device – Full speed
- Kontroler sieci Ethernet 100MBit
- Zegar czasu rzeczywistego RTC
- Interfejs I2C
- 3 Interfejsy SSP/I2S
- Interfejs SPI
- 61 linii GPIO (tolerujace 5V)
- 2 porty RS232
- port RS485 z automatyczną zmianą kierunku
- złącze JTAG
- Napiecie zasilające 7V-35V/max .200mA (max. 700mA, przy maksymalnym obciażeniu USB Host)
- Oprogramowanie: dedykowany Boff Linux (Kernel 2.6.27.8)

Na rysunku (n) przedstawiono schemat blokowy ARMputera BF-210



Sercem układu jest mikrokontroler **AT91RM9200** zawierający rdzeń **ARM9** (**ARM920T**) wraz z układem **MMU**, kontrolerem pamięci **SDRAM**, układami czasowo-licznikowymi, zegarem RTC, oraz szeregiem różnych interfejsów komunikacyjnych takich jak: kontroler hosta USB, kontroler urządzenia USB, SPI, I2C itp. Mikrokontroler posiada wbudowane 16kB wewnętrznej pamięci **RAM** przez co konieczne stało się dołączenie zewnętrznej pamięci **SDRAM** o wielkości 32MB. Na płytce umieszczono również pamięć Data Flash o pojemności 128KB dołączonej do magistrali SPI0, która pełni rolę odpowiednika **BIOS-u** znanego z komputerów PC. W pamięci tej przechowywane są programy ładujące (bootloadery) odpowiedzialne za podstawowe testy, konfigurację pamięci **SDRAM** oraz proces uruchamiania linux-a z karty **SD**, lub poprzez sieć z serwera **NFS**. Karta **MMC/SD/SDHC** została dołączona do mikrokontrolera za pomocą dedykowanej magistrali **MMC/SD**, która ma szerokość 4 bitów. Możemy tutaj użyć dowolnej karty powszechnie dostępnej na rynku o pojemności od 64MB, aż do kilkunastu GB w zależności od konkretnych wymagań. **AT91RM9200** posiada wbudowany kontroler Ethernet MAC, który musi zostać wyposażony w zewnętrzny układ warstwy fizycznej tak zwany (PHY). W naszym rozwiązaniu wykorzystano układ **STE100p**. Z myślą o zastosowaniach przemysłowych do jednego z portów szeregowych mikrokontrolera dołączono typowy układ konwertera zapewniający wyjście w standardzie RS485, z automatycznym przełączaniem kierunku. Do wyjścia dwóch UARTów dołączono układy konwerterów zapewniający translację napięć do standardu RS232. Jednym z układów dołączonych do konwertera **RS232** jest **UART-DBG**, który stanowi konsolę dla linuxa i bootloadera, więc dołączając się do tego portu za pomocą terminala możemy sprawdzić jak przebiega proces uruchamiania, lub zalogować się do konsoli bootloadera czy linuxa. Na płytce umieszczono także złącze Hosta USB, co daje możliwość dołączania urządzeń przeznaczonych dla

komputerów PC takich jak: Pamięci Pendrive, Kamery USB, drukarki itp. **BF-210** zawiera także złącze **USB-DEVICE**, zatem możemy go również dołączyć do komputera PC tak żeby się zgłaszał jako urządzenie USB, na przykład modem, port szeregowy, pamięć USB itp. Wszystkie pozostałe linie mikrokontrolera GPIO oraz pozostałe magistrale **SPI, I2C** są wyprowadzone na zewnątrz płytki za pomocą trzech 40-pinowych złącz typu gold-pin w typowym rastrze 2,54mm. Do złącz tych możemy dołączać inne układy w zależności od okoliczności. Do zasilania modułu należy użyć dowolnego zasilacza zasilacza niestabilizowanego 7-35V. **BF-210** pobiera około 200mA jednak ze złącza hosta USB zgodnie ze specyfikacją może być pobierany prąd o natężeniu **500mA**, dlatego potrzebny jest zasilacz o stosunkowo dużej wydajności prądowej.

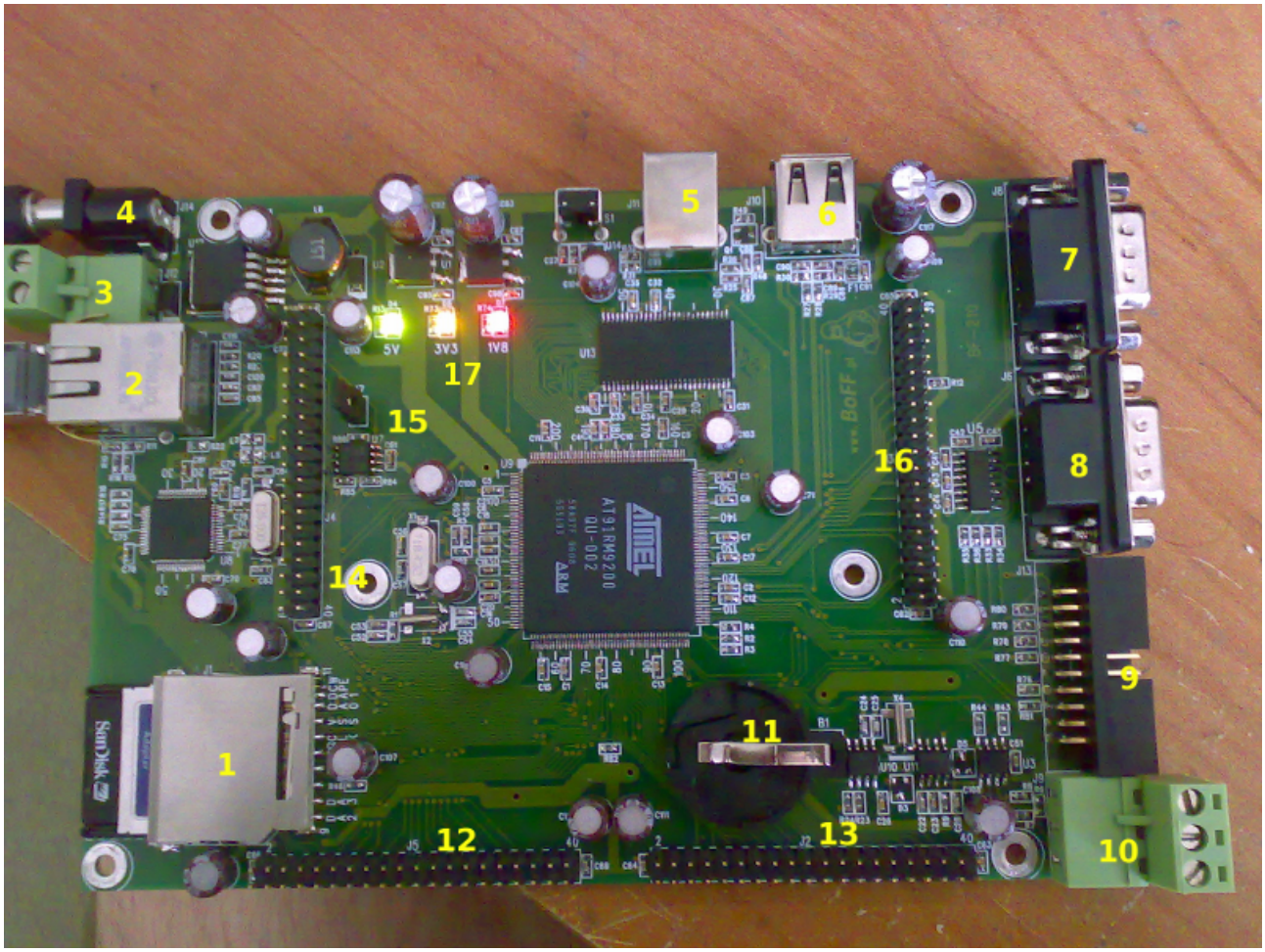
2. Oprogramowanie.

Oprogramowanie ARMputera **BF-210** jest stosunkowo skomplikowane i składa się z dwóch części. Stanowią je pamięć BIOS (Serial Dataflash) o pojemności 128KB, w której zawarte są dwa bootloadery inicjalizujący, oraz bootloader główny ***u-boot*** . Natomiast drugą część zawartą na karcie **MMC/SD/SDHC** stanowi system linux. Mikrokontroler po włączeniu napięcia zasilania rozpoczyna wykonywanie programu zawartego w wewnętrznej pamięci **ROM**, który wczytuje do wewnętrznej pamięci **RAM** o rozmiarze 16kB inicjalizujący bootloader zawarty w pamięci Serial DataFlash. Zadaniem tego bootloadera jest inicjalizacja podstawowych układów peryferyjnych mikrokontrolera pętli **PLL**, pamięci **SDRAM**, testowanie pamięci **SDRAM**, a następnie skopiowanie bootloadera głównego **u-boot** do pamięci SDRAM oraz jego uruchomienie. Bootloader ten posiada także podstawową funkcjonalność która pozwala na przeprowadzenie podstawowych testów pamięci, oraz umożliwia przeprogramowanie pamięci **BIOS** z wykorzystaniem protokołu X-Modem. Bootloader **u-boot** jest odpowiedzialny za odczytanie z karty pamięci **SD** jądra linuxa, a następnie jego uruchomienie. Jest on dużo bardziej rozbudowany od wspomnianego wcześniej bootloadera inicjalizującego i zawiera rozbudowany wiersz poleceń, obsługę sieci, obsługę systemu plików **EXT2**. Umożliwia on również uruchomienie ARMputera z serwera sieciowego **TFTP, NFS**.

Urządzenie **BF-210** jest dostarczane z zaprogramowaną pamięcią **DATA-FLASH** zawierającą oba bootloadery. W komplecie dostarczana jest także karta **SD-1GB** zawierająca, podstawową dystrybucję linuksa, która może być wykorzystana jako podstawa własnych projektów.

3. Opis złącz zestawu BF-210

Na rysunku 2 przedstawiono opis zewnętrzny złącz ARMputera **BF-210**



1 – Złącze kart pamięci SD/MMC/SDHC

2 – Złącze LAN

3 , 4 – Złącza zasilające (Napięcie 7-35V max. 700mA)

5 – Gniazdo USB-Device

6 – Gniazdo USB-Host

7 – Gniazdo portu szeregowego UART1

8 – Gniazdo portu szeregowego DBG (konsola linuxa)

9 – Złącze JTAG WIGLER

10 – Złącze interfejsu RS485

11 – Złącze baterii dodatkowego zegara RTC

12,13,14,16 – Złącza portów GPIO

15 – zwora odłączenia pamięci BIOS

17 – sygnalizacja napięć zasilających.

4. Uruchomienie oraz podłączenie ARMputera

- Do gniazda karty pamięci **MMC/SD** [1] należy włożyć kartę pamięci z systemem linux.
- Do złącza LAN [2] podłączyć kabel sieci lokalnej
- Do złącza konsoli szeregowej **DBGU** [8] dołączyć port szeregowy komputera RS232 z uruchomionym dowolnym programem terminalowym (np. Hyperterminal, minicom) skonfigurowanym z następującymi parametrami: 115200,n,8,1
- Do złącza zasilającego (3) lub (4) należy dołączyć dowolny zasilacz o napięciu 7-35V i wydajności prądowej minimum 200mA.

Uwaga! Do połączenia konsoli szeregowej z komputerem PC należy użyć kabla crossowego tzw. NULL MODEM RS232.
Na rysunku (3) przedstawiono przykładowy kabel crossowy RS232 typu DB9-DB9
[3] ---------- [2]
[2] ---------- [3]
[5] ---------- [5]

Dystrybucja linuksa zawarta na karcie **SD** domyślnie skonfigurowana jest do pobierania adresu z serwera **DHCP**, zatem w sieci lokalnej powinien być aktywny serwer **DHCP**. Po włączeniu napięcia zasilania powinniśmy zaobserwować proces uruchamiania się systemu linux oraz powinna zostać wyświetlona informacja o aktualnie przydzielonym adresie IP. Przykładowe komunikaty wypisywane podczas uruchamiania systemu na konsoli szeregowej przedstawiono na rysunku 4.

```
BoFF loader - Thanks to the darrel-loader  project
Version 1.1. Build Nov 15 2008 20:22:24
Based on Darrel loader project  Lucjan Bryndza <lucjan.bryndza@ep.com.pl>.
License GPL v2/3
DRAM:32MB


1: Upload Darrell's loader to Dataflash
2: Upload u-boot to Dataflash
4: Start u-boot
5: Erase dataflash
6: Memory test
DataFlash:AT45DB161
```

```
Dataflash read successful: Starting U-boot



U-Boot 1.3.3 (Jun 21 2008 - 12:06:23)

DRAM:  32 MB
Atmel: Flash:  0 kB
DataFlash:AT45DB161
Nb pages:   4096
Page Size:    528
Size= 2162688 bytes
Logical address: 0xC0000000
Area 0: C0000000 to C0002C57 (RO) Darrell loader
Area 1: C0002C58 to C0020DEF (RO) U-boot
Area 2: C0020DF0 to C0020FFF      Environment
In:    serial
Out:   serial
Err:   serial
Hit any key to stop autoboot:  1 ### 0
Manufacturer ID:       1C
OEM/Application ID:    5356
Product name:          SDC
Product Revision:      1.0
Product Serial Number: 95684228
Manufacturing Date:    08/04
SDv2/SDHC Card detected (RCA 45928) SDHC flag 1
CSD structure version:   1.1
MMC System Spec version: 0
Card command classes:    5f5
Read block length:       512
Does not support partial reads
Write block length:      512
Does not support partial writes
Does not support group WP
Card capacity:        3814 MB
File format:            0/0
Write protection:
mmc: Using 53248 cycles data timeout (DTOR=0x5d)

1199980 bytes read
## Booting kernel from Legacy Image at 21000000 ...
   Image Name:   Linux-2.6.27.8
   Image Type:   ARM Linux Kernel Image (uncompressed)
```

```
   Data Size:    1199916 Bytes =  1.1 MB
   Load Address: 20008000
   Entry Point:  20008000
   Verifying Checksum ... OK
   Loading Kernel Image ... OK
OK


Starting kernel ...


Uncompressing Linux.....................................................................
done, booting the kernel.
[    0.000000] Linux version 2.6.27.8 (lucck@vbox) (gcc version 4.3.1 (GCC) ) #1 PREEMPT Mon Dec 15
20:25:15 CET 2008
[    0.000000] CPU: ARM920T [41129200] revision 0 (ARMv4T), cr=c0007177
[    0.000000] Machine: Boff AT91RM9200 Board
[    0.000000] Memory policy: ECC disabled, Data cache writeback
[    0.000000] Clocks: CPU 179 MHz, master 59 MHz, main 18.432 MHz
[    0.000000] CPU0: D VIVT write-back cache
[    0.000000] CPU0: I cache: 16384 bytes, associativity 64, 32 byte lines, 8 sets
[    0.000000] CPU0: D cache: 16384 bytes, associativity 64, 32 byte lines, 8 sets
[    0.000000] Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 8128
[    0.000000] Kernel command line: mem=32M root=/dev/mmcblk0p1 console=ttyS0,115200n8 rootdelay=1
init=/sbin/init
[    0.000000] AT91: 96 gpio irqs in 3 banks
[    0.000000] PID hash table entries: 128 (order: 7, 512 bytes)
[17179569.184000] Console: colour dummy device 80x30
[17179569.184000] console [ttyS0] enabled
[17179569.272000] Dentry cache hash table entries: 4096 (order: 2, 16384 bytes)
[17179569.280000] Inode-cache hash table entries: 2048 (order: 1, 8192 bytes)
[17179569.292000] Memory: 32MB = 32MB total
[17179569.296000] Memory: 29980KB available (2188K code, 183K data, 88K init)
[17179569.308000] Calibrating delay loop... 89.34 BogoMIPS (lpj=178688)
[17179569.388000] Mount-cache hash table entries: 512
[17179569.396000] CPU: Testing write buffer coherency: ok
[17179569.416000] net_namespace: 288 bytes
[17179569.424000] NET: Registered protocol family 16
[17179569.484000] SCSI subsystem initialized
[17179569.496000] usbcore: registered new interface driver usbfs
[17179569.504000] usbcore: registered new interface driver hub
[17179569.512000] usbcore: registered new device driver usb
[17179569.560000] NET: Registered protocol family 2
[17179569.604000] IP route cache hash table entries: 1024 (order: 0, 4096 bytes)
[17179569.612000] TCP established hash table entries: 1024 (order: 1, 8192 bytes)
[17179569.620000] TCP bind hash table entries: 1024 (order: 0, 4096 bytes)
[17179569.628000] TCP: Hash tables configured (established 1024 bind 1024)
```

```
[17179569.632000] TCP reno registered
[17179569.648000] NET: Registered protocol family 1
[17179569.656000] NetWinder Floating Point Emulator V0.97 (double precision)
[17179569.676000] msgmni has been set to 58
[17179569.680000] io scheduler noop registered
[17179569.684000] io scheduler anticipatory registered (default)
[17179571.332000] atmel_usart.0: ttyS0 at MMIO 0xfefff200 (irq = 1) is a ATMEL_SERIAL
[17179571.344000] atmel_usart.1: ttyS1 at MMIO 0xfffc4000 (irq = 7) is a ATMEL_SERIAL
[17179571.356000] atmel_usart.2: ttyS2 at MMIO 0xfffc8000 (irq = 8) is a ATMEL_SERIAL
[17179571.376000] eth0: Link now 100-FullDuplex
[17179571.380000] eth0: AT91 ethernet at 0xfefbc000 int=24 100-FullDuplex
[17179571.388000] eth0: STE100P PHY
[17179571.396000] Driver 'sd' needs updating - please use bus_type methods
[17179571.408000] at91_ohci at91_ohci: AT91 OHCI
[17179571.416000] at91_ohci at91_ohci: new USB bus registered, assigned bus number 1
[17179571.424000] at91_ohci at91_ohci: irq 23, io mem 0x00300000
[17179571.492000] usb usb1: configuration #1 chosen from 1 choice
[17179571.500000] hub 1-0:1.0: USB hub found
[17179571.504000] hub 1-0:1.0: 1 port detected
[17179571.620000] usb usb1: New USB device found, idVendor=1d6b, idProduct=0001
[17179571.628000] usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
[17179571.636000] usb usb1: Product: AT91 OHCI
[17179571.640000] usb usb1: Manufacturer: Linux 2.6.27.8 ohci_hcd
[17179571.648000] usb usb1: SerialNumber: at91
[17179571.652000] Initializing USB Mass Storage driver...
[17179571.660000] usbcore: registered new interface driver usb-storage
[17179571.664000] USB Mass Storage support registered.
[17179571.672000] udc: at91_udc version 3 May 2006
[17179571.680000] mice: PS/2 mouse device common for all mice
[17179571.692000] at91_rtc at91_rtc: rtc core: registered at91_rtc as rtc0
[17179571.696000] AT91 Real Time Clock driver.
[17179571.708000] AT91 Watchdog Timer enabled (5 seconds, nowayout)
[17179571.716000] at91_mci at91_mci: 4 wire bus mode not supported - using 1 wire
[17179571.736000] TCP cubic registered
[17179571.740000] NET: Registered protocol family 17
[17179571.756000] at91_rtc at91_rtc: setting system clock to 1998-01-01 00:00:14 UTC (883612814)
[17179571.768000] Waiting 1sec before mounting root device...
[17179571.780000] mmc0: host does not support reading read-only switch. assuming write-enable.
[17179571.788000] mmc0: new SDHC card at address b368
[17179571.796000] mmcblk0: mmc0:b368 SDC   3906560KiB
[17179571.800000]  mmcblk0: p1
[17179572.784000] kjournald starting.  Commit interval 5 seconds
[17179572.796000] EXT3 FS on mmcblk0p1, internal journal
```

```
[17179572.800000] EXT3-fs: mounted filesystem with ordered data mode.
[17179572.808000] VFS: Mounted root (ext3 filesystem).
[17179572.812000] Freeing init memory: 88K
Populating /dev using udev: done
Standard directories...
Initializing random number generator... done.
Starting network...
[17179580.244000] eth0: Link now 100-FullDuplex
udhcpc (v1.13.1) started
Sending discover...
Sending select for 192.168.16.105...
Lease of 192.168.16.105 obtained, lease time 140773632
deleting routers
route: SIOCDELRT: No such process
adding dns 192.168.16.1
Setup time from NTP...
16 Dec 15:51:56 ntpdate[766]: step time server 207.46.197.32 offset 345829891.803440 sec
Starting dropbear sshd: OK


Welcome to BoFF BF-210 board
boff-bf210 login:
```

Po uruchomieniu ARMputera możemy zalogować się do niego z komputera **PC** używając konsoli szeregowej i terminala , lub protokołu **SSH** za pomocą polecenia *ssh root@ip_armputera*. W dystrybucji domyślnie uruchomiony jest lekki serwer **SSH** (*dropbear*).

Uwaga! Domyślnie w BF-210 hasło do konta **root** ustawione jest na **w1sl0k**

5. Płyta DVD – toolchain BF-210

5.1 Zawartość płyty

Wraz z ARMputerem **BF-210** dostarczana jest płyta DVD zawierająca toolchain bazujący na buildroocie umożliwiający na komputerze **PC** z zainstalowanym linuksem tworzenie własnego oprogramowania dla ARMputera. W skład płyty DVD wchodzą:

- Toolchain dla komputerów PC z 32 bitowym procesorem.
- Toolchain dla komputerół PC z 64 bitowym procesorem
- Schemat ideowy ARMputera BF-210
- Przykładowa aplikacja (Hello World)
- Skompilowane bootloadery oraz linux zawarte w pamięci EEPROM oraz na karcie SD BF-

210

5.2 Minimalne wymagania komputera PC

Minimalne wymagania systemowe jakie musi spełniać komputer PC, do prawidłowej pracy toolchaina umożliwiającego tworzenie własnych aplikacji przedstawiono w tabeli 2.

Minimalne wymagania systemowe komputera PC niezbędnego do uruchomienia toolchaina
• Procesor minimum 32 bitowy i586 • 512MB pamięci RAM • Zainstalowany system Linux (Preferowany ubuntu 8.10 lub nowszy)

5.3 Instalacja toolchaina, oraz przykładowej aplikacji

Przed instalacją należy upewnić się czy używamy wersji 32 lub 64 bitowej linuksa. W przypadku wersji 32-bitowej należy użyć pliku **buildroot-toolchain-i386.tgz** natomiast w przypadku wersji 64-bitowej **buildroot-toolchain-amd64.tgz** Następnie należy utworzyć dowolny katalog w którym zainstalowany będzie toolchain (np. za pomocą polecenia *mkdir ~/bf210*) oraz rozpakować go do w tym katalogu (*tar xvfz buildroot-toolchain-i386.tgz*). Do katalogu należy również z płyty DVD skopiować pliki *acompile.env kernel.env* oraz przykładową aplikację *helloapp.tgz*, a następnie rozpakować ją.

5.4 Kompilacja przykładowej aplikacji oraz uruchomienie

Przed kompilacją programu należy ustawić zmienne środowiskowe toolchaina, w tym celu będąc w katalogu gdzie wcześniej rozpakowaliśmy toolchain należy wydać polecenie *source ./acompile.env*. Następnie należy wejść do katalogu helloapp i wydać polecenie make. Po tej czynności powinien powstać plik wynikowy hello na platformę ARM, który należy przegrać do armputera np. za pomocą polecenia scp (*scp ./hello* root@ip_*armputera:~*), następnie należy zalogować się do konsoli BF-210 np. za pomocą polecenia *ssh* i uruchomić przykładową aplikację wpisując ./hello. Na konsoli powinniśmy wówczas zobaczyć napis: *Hello World*

6. Skład zestawu BF-210

W skład zestawu **BF-210** wchodzą następujące elementy:

- Płytka urządzenia **BF-210**
- Płyta **DVD** zawierająca toolchain umożliwiający tworzenie oprogramowania dla **BF-210**