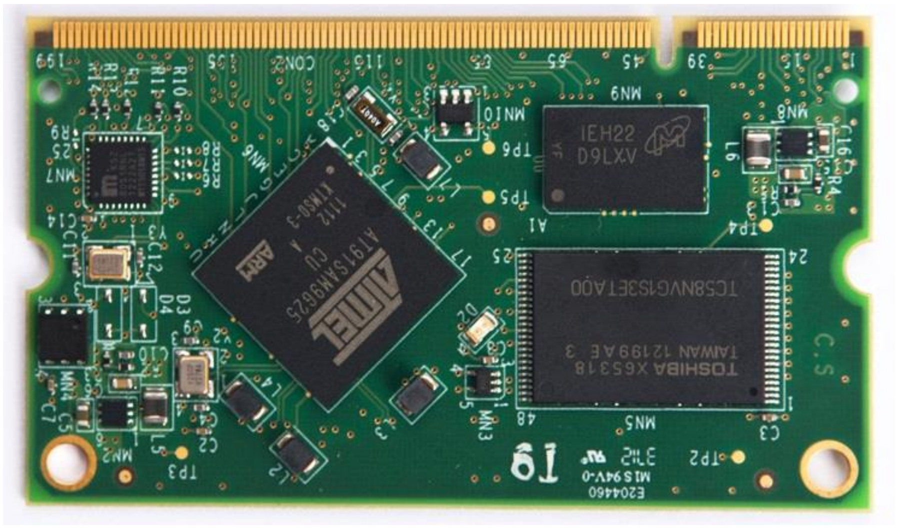
AT-901

ARM9 System on Module



Software user manual

Revision 1.0

Contents

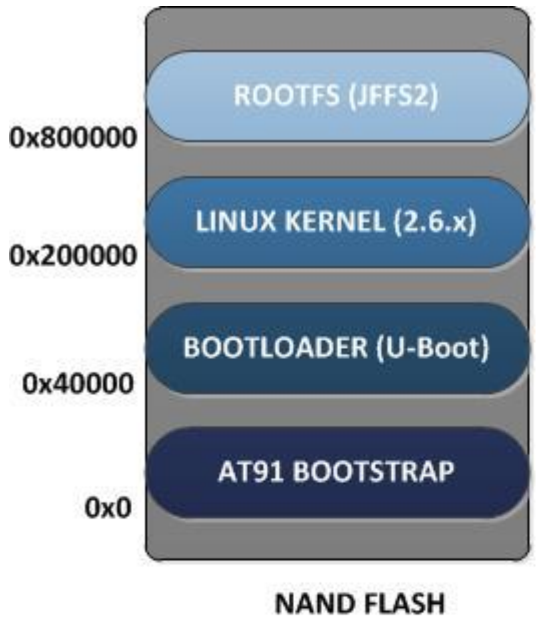
1. Overview

The Software user manual describes all the required procedure for using the ShiraTech SoM including the procedures for downloading the SW parts to the Shiratech AT-901 board, SW packages installation cross compilation and more. The description is assuming using Shiratech's Starter kit (See more details on the starter kit in appendix A)

2. Linux Firmware

Linux firmware consists of following modules:

1. Boot Program (located in the internal ROM of the processor) - Check if a valid application is present in FLASH and if it is the case downloads it into internal SRAM.
2. AT91Bootstrap - In charge of hardware configuration, download U-Boot binary from FLASH to SDRAM, start the boot loader
3. U-Boot - The boot loader, in charge of download kernel binaries from FLASH, network, USB key, etc. Start the kernel.
4. Linux kernel - The operating system kernel.
5. Root File-system - Contains applications which are executed on the target, using the OS kernel services.



3. Deliveries

We provide two types of deliveries:

- A “clean” version including only SSH (dropbear) and a light weight *ethtool* in that case we provide several other packages like *Snmpd*, SQLite, *Java*, *Mongoose*, *dhclient* which the user can install, additional packages can be provided (connects support@shiratech@com).

- A full version in which all the packages are already integrated.

For both options we provide Bootstrap, U-boot, Kernel, Debian-file system for running either from NAND flash or SD card.

4. SW Installation steps

The following steps describes the how to install boot, U-boot and Linux Kernel and Debian file system. The boot and U-boot will be supplied with the SoM but the following process can be done in case of memory corruption, or in case of an upgrade. For these cases we offer to use the SAM-BA tools supplied by Atmel.

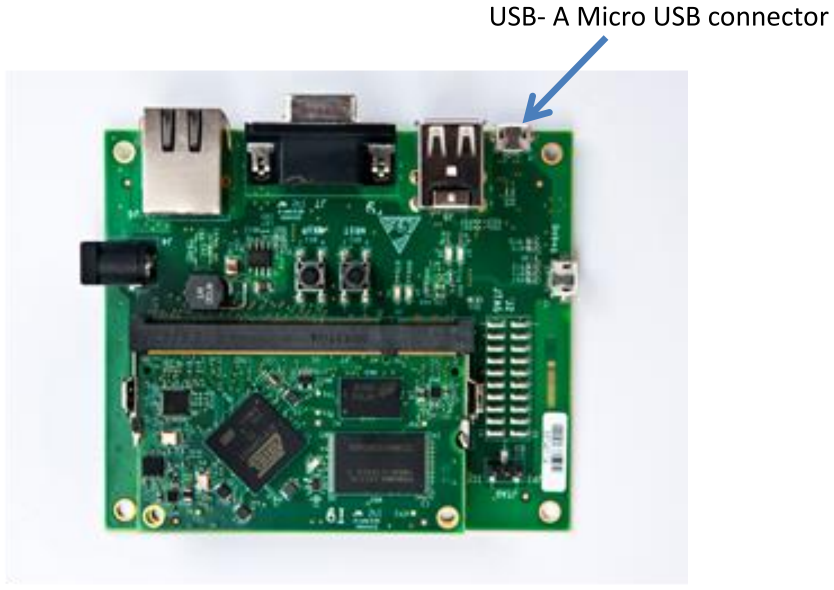
Kernel and Debian can be installed after download via TFTP from a host PC.

4.1. SAM-BA Introduction

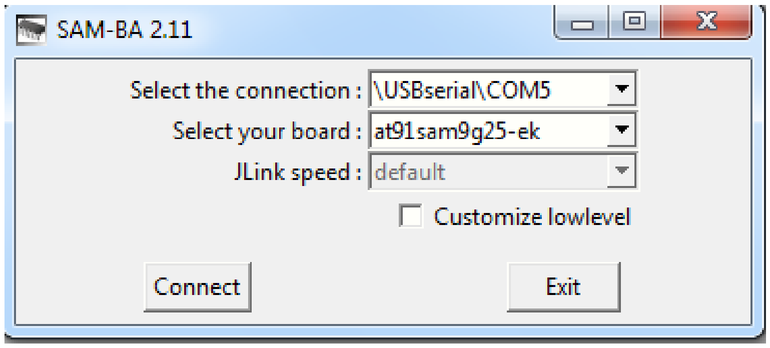
ATMEL provides a software tool called SAM-BA to burn the boot loader in Flash memory. Download the Software from ATMEL website and install it. After installation it will create a virtual COM port on Windows. It can be downloaded from the following link:

http://www.atmel.com/tools/ATMELSAM-BAIN-SYSTEMPROGRAMMER.aspx

Connect the USB cable to the Micro USB (USB A) port of the board. See port location in the picture below.



Double click on SAM-BA and it will open the following window:


SAM-BA 2.11

Select the connection : \USBserial\COM5

Select your board : at91sam9g25-ek

JLink speed : default

☐ Customize lowlevel

The virtual COM port number (COM5) may vary from PC to PC. Click Connect button and it will open a window.

Note: The SAM-BA tool will be connected only if there is no valid Boot on the NAND flash.

4.2. AT91Bootstrap and U-boot installation steps

For ATMEL AT91SAM9G25 board the boot loader and Linux kernel are installed using software SAM-BA. The Boot loader consists of two components.

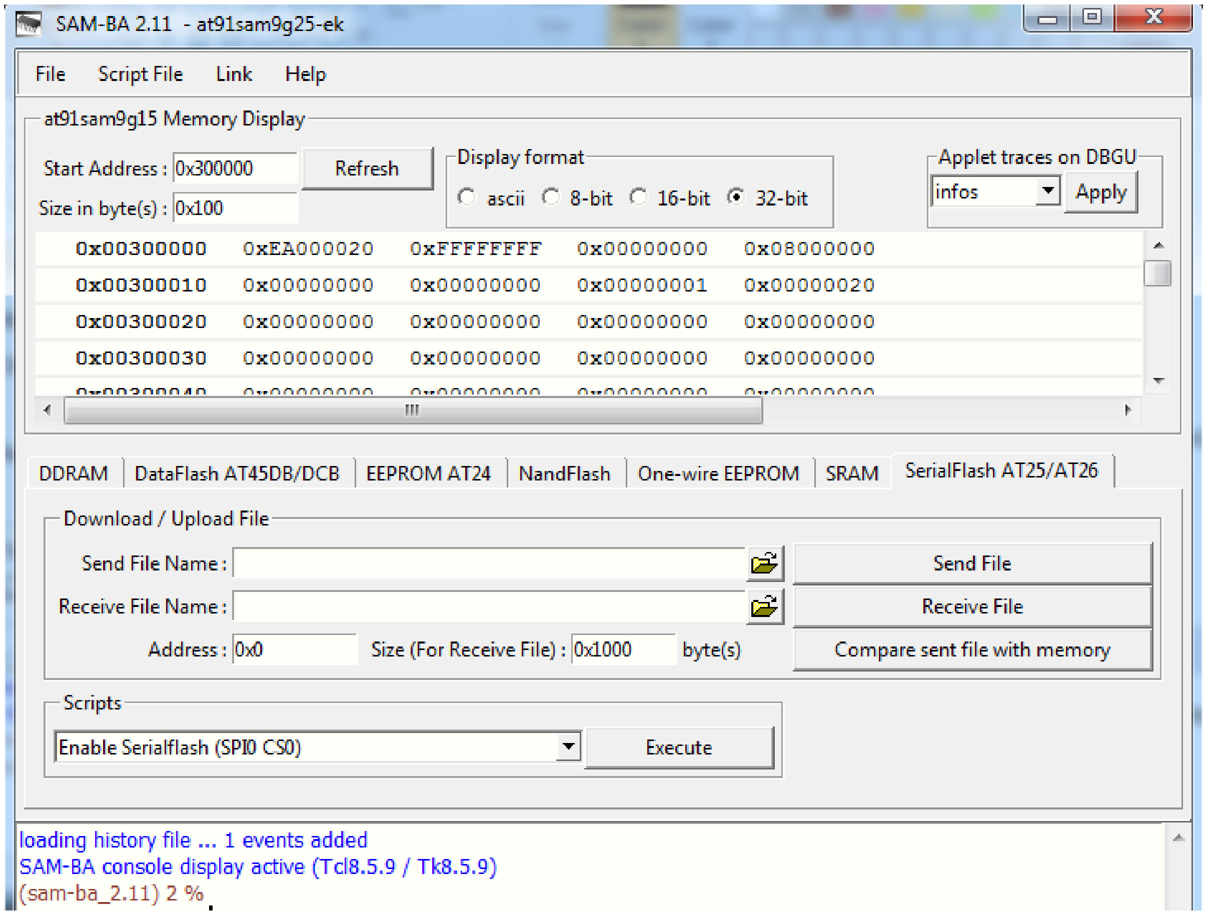
- AT91Bootstrap
- U-boot

4.2.1. AT91Bootstrap download procedure:

1. Choose the NandFlash media tab in the SAM-BA GUI.

2. Initialize the media choosing the Enable NandFlash action in the Scripts rolling menu and press Execute.

3. Select Send Boot File in the Scripts rolling menu and press Execute. Then select the bootstrap file and press Open; the media is written down.



Start Address : 0x300000
Size in byte(s) : 0x100
Display format: ascii [✓ 32-bit] 8-bit 16-bit
Applet traces on DBGU: infos

```
0x00300000   0xEA000020   0xFFFFFFFF   0x00000000   0x08000000
0x00300010   0x00000000   0x00000000   0x00000001   0x00000020
0x00300020   0x00000000   0x00000000   0x00000000   0x00000000
0x00300030   0x00000000   0x00000000   0x00000000   0x00000000
```

Send File Name :
Receive File Name :
Address : 0x0   Size (For Receive File) : 0x1000 byte(s)

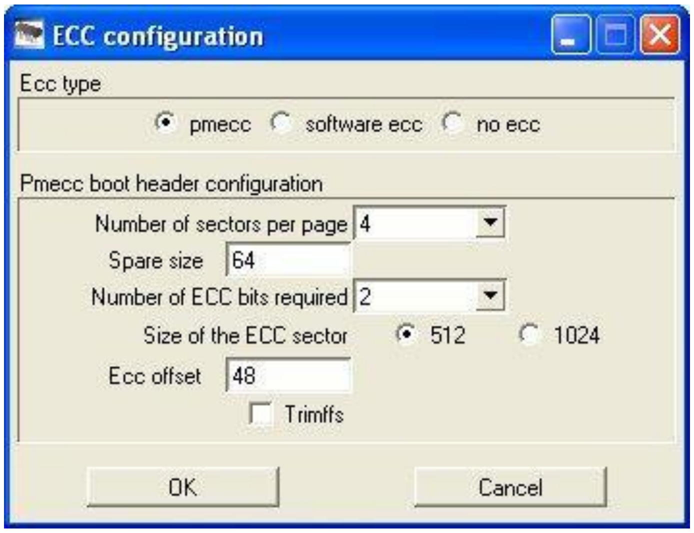
Scripts: Enable Serialflash (SPI0 CS0)

```
loading history file ... 1 events added
SAM-BA console display active (Tcl8.5.9 / Tk8.5.9)
(sam-ba_2.11) 2 %
```

4.2.2. U-boot download procedure:

To download the U-boot to the NAND flash the following steps should be done:

- Choose the NandFlash media tab in the SAM-BA GUI.
- Initialize the media choosing the Enable Nand Flash action in the Scripts rolling menu and press Execute.
- Select Enable OS PMECC parameters in the Scripts rolling menu and press Execute. The default ECC Configuration should be ok so you should have this pop-up appearing:



Ecc type: [✓ pmecc] software ecc no ecc

Pmecc boot header configuration

Number of sectors per page: 4

Spare size: 64

Number of ECC bits required: 2

Size of the ECC sector: [✓ 512] 1024

Ecc offset: 48

Trimffs: ☐

Press the OK button.

- To erase only the U-Boot part into the NAND FLASH, type this command after the SAM-BA prompt:

  NANDFLASH::EraseBlocks 0x40000 0xBFFFF

  Then press the Enter Key.

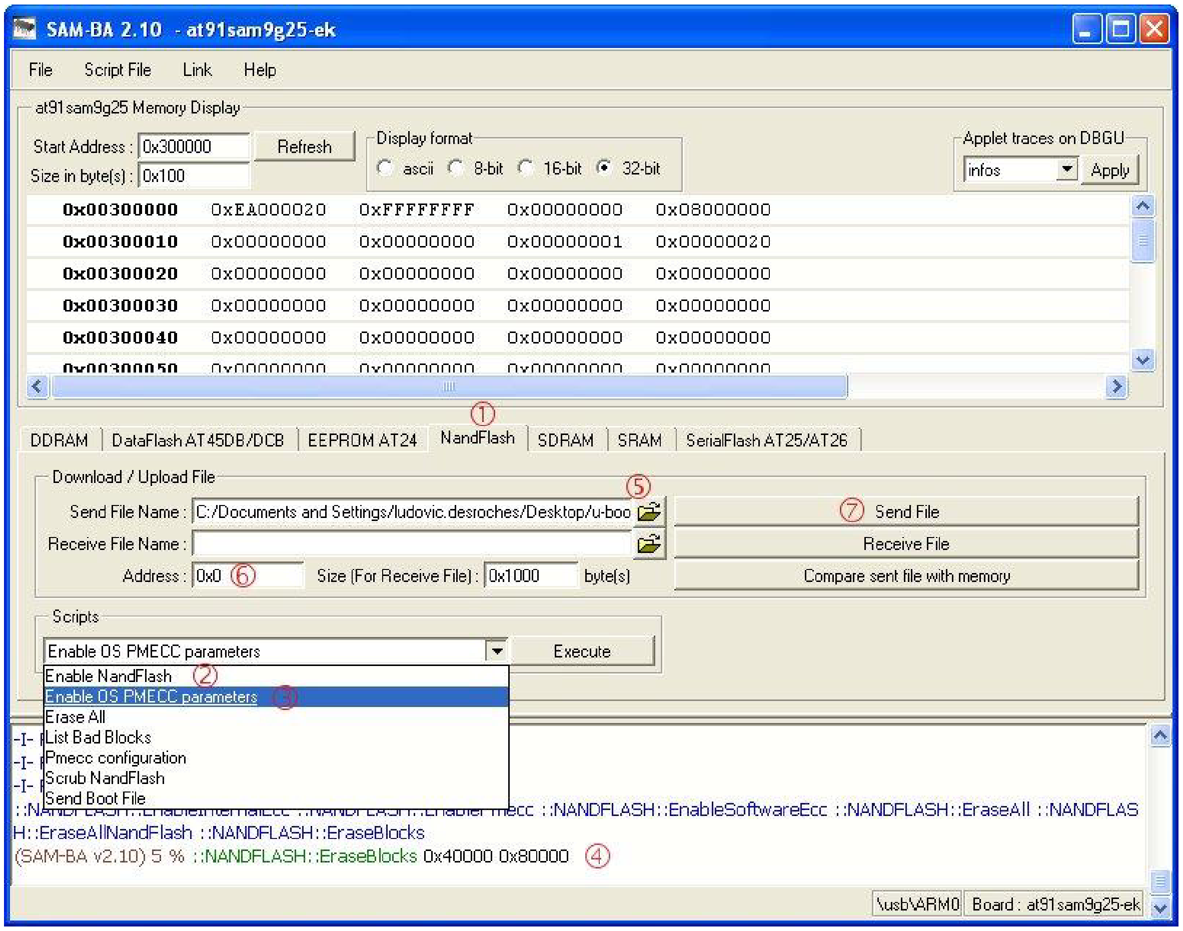
  Note: The SAM-BA EraseBlocks command take two parameters: a start and an end address of FLASH that you want to erase.

- Press Send File Name Browse button and choose your U-Boot binary file.
- Enter the proper address on media in the Address text field. Its value should be 0x40000.
- Press Send File button.
- Close SAM-BA, remove the USB cable.

See also the picture below for the exact steps required

.

Once the Bootloader is installed power up the board. You will get the Bootloader prompt (U-boot>).



Start Address : 0x300000
Size in byte(s) : 0x100
Display format: ascii ( ) 8-bit ( ) 16-bit ( ) 32-bit (●)
Applet traces on DBGU: infos

```
0x00300000   0xEA000020   0xFFFFFFFF   0x00000000   0x08000000
0x00300010   0x00000000   0x00000000   0x00000001   0x00000020
0x00300020   0x00000000   0x00000000   0x00000000   0x00000000
0x00300030   0x00000000   0x00000000   0x00000000   0x00000000
0x00300040   0x00000000   0x00000000   0x00000000   0x00000000
0x00300050   0x00000000   0x00000000   0x00000000   0x00000000
```

Send File Name : C:/Documents and Settings/ludovic.desroches/Desktop/u-boo
Receive File Name :
Address : 0x0
Size (For Receive File) : 0x1000 byte(s)
Scripts: Enable OS PMECC parameters

```
-I- 
-I- 
-I- 
::NANDFLASH::EnableInternalEcc ::NANDFLASH::EnablePmecc ::NANDFLASH::EnableSoftwareEcc ::NANDFLASH::EraseAll ::NANDFLAS
H::EraseAllNandFlash ::NANDFLASH::EraseBlocks
(SAM-BA v2.10) 5 % ::NANDFLASH::EraseBlocks 0x40000 0x80000
```

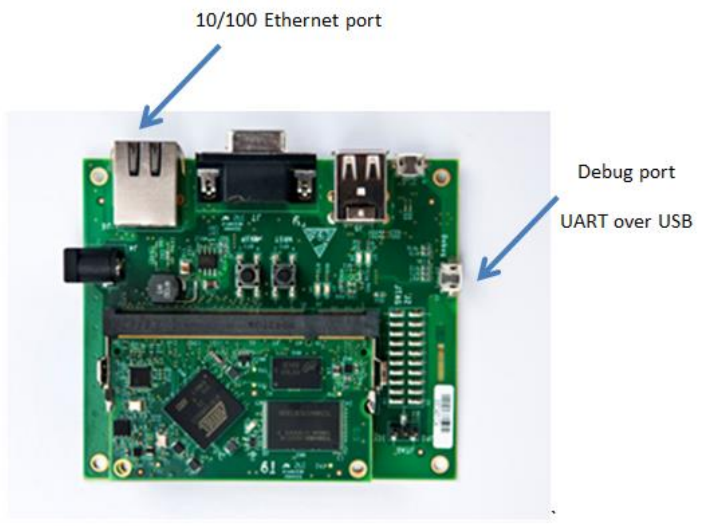
\usb\ARM0 Board : at91sam9g25-ek

4.3. LINUX KERNEL Installation

Once the boot loader is installed on board, the Linux kernel is installed by the u-boot prompt using TFTP commands to install the Linux Kernel.

To install the Kernel do the following steps:

- Open HyperTerminal in PC. Connect a USB cable to the debug port(see picture below for port location), select the virtual com detected by the PC. Set the rate to 115,200 Bps for maximum performance.
- Power up the board
- Enable a TFTP server the host PC.
- Connect the Ethernet port of the board to the network



- Configure the IP address of host, IP address of board and Ethernet address of board by using u-boot commands as follows (The IP addresses can varied according to the setup used)

  u-boot > setenv serverip 10.10.10.10

  " 10.10.10.10" is IP address of host PC

  u-boot>setenv ipaddr 10.10.10.5

  "10.10.10.5" will be IP address of board.

  u-boot>setenv ethaddr 12:34:56:78:9a:bc
  u-boot>saveenv

- Put the file "uImage-2.6.39-at91sam9x5ek.bin" on Windows host PC in which TFTP is configured.

  e.g. In Windows create tftpboot folder and provide read/write permission for the folder. Put the kernel file "uImage-2.6.39-at91sam9x5ek.bin" inside tftpboot folder. Disable firewall in host pc.

- Run the following commands from u-boot prompt

  u-boot> tftp 0x22000000 uImage-2.6.39-at91sam9x5ek.bin
  (should be a ***valid name*** of the Kernel file)
  u-boot> nand erase 0x200000 0x600000
  u-boot> nand write 0x22000000 0x200000 0x250000

- Configure the booting arguments.

```
u-boot>setenv bootargs 'mem=128M console=ttyS0,115200
mtdparts=atmel_nand:8M(bootstrap/uboot/kernel)ro,-(rootfs) root=/dev/mtdblock1
rw rootfstype=jffs2 :rootfs'
u-boot>saveenv
```

- Boot the Linux kernel using following command.

```
u-boot> boot
```

  After boot it will show error proper file system couldn't be mounted, since the file system was not installed

4.4. DEBIAN FILE SYSTEM Installation

To install the Debian file system follow the following steps:

- Put the compressed file system image "*debian_baseV1.jffs2* " in tftpboot folder of host PC(the file name should be the valid file used).

The following steps are used to install Debian file system from boot loader prompt.

```
u-boot> tftp 0x22000000 debian_baseV1.jffs2
u-boot> nand erase 0x800000 0xf800000
u-boot> nand write.trimffs 0x22000000 0x800000 0x30A32C0
```

Then reset the board. After reboot you will get a login prompt.

Username: root
Password: root

5. Installing SW packages

Installing package can be done using TFTP for downloading the packages from a remote PC or via a USB dongle plugged in to the USB port. The packages to install on the board are provided as tar.gz files.

5.1. Installation using TFTP

To install packages in the file-system you have to copy the each package files in to your Debian file system. For copying tar.gz files follow the below mentioned steps.

- Configure and run a TFTP server with the package_name.tar.gz file.

- After booting the Debian in ShiraTech board use the following command to copy the package-name.tar.gz to the file-system.

tftp –g –r package-name.tar.gz tftp_server_ip

e.g: tftp –g –r web-app.tar.gz 128.88.159.144

- Extract the "package-name.tar.gz" using,

  tar –xvf package-name.tar.gz
  e.g: "tar –xvf web-app.tar.gz"

- The result of the above commands will become a "package-name" folder
  e.g: "web-app"
- Please note the path of the "package-name" folder. Now you can proceed to install the package.

5.1.1. Install SQLite

To install SQLite database use the following command.

install-package SQLITE "path_to_sqlite_package
e.g: "install-package SQLITE /home/packages/sqlite"

5.1.2. Install SNMPD

To install SNMPD use the following command.

install-package SNMPD "path_to_snmpd_package
e.g: "install-package SNMPD /home/packages/snmpd"

5.1.3. Install SSH Server - Dropbear

To install SSH Server "Dropbear", use the following command.

```
cd /etc
mkdir dropbear
cd dropbear
dropbearkey -t rsa -f dropbear_rsa_host_key
dropbearkey -t dss -f dropbear_dss_host_key
dropbear start
```

For remote access to the shiratech board, run the below commands in linux machine

ssh root@ipaddress_of_Shiratech_board

Then enter the password of Shiratech login.

5.2. Install using SD CARD or USB dongle

Packages to install on the board can be installed using an SD card or a USB dongle. To install packages in the file system you have to copy the packages required to the SD card or USB dongle and mount the Sd card/USB dongle. Mounting the SD card/USB should be done as follows:

dmesg → It will display the device path on which the sdcard/USB is mounted to.

e.g : /dev/sda1

mount /dev/path /mnt

e.g.: mount /dev/sda1 /mnt

5.2.1. Install SQLite

To install ***SQLite*** database use the following command:

install-package SQLITE "path of sqlite package"

e.g : install-package SQLITE /mnt/ sqlite

5.2.2. Install SNMPD

To install ***snmpd*** package use the following command.

install-package SNMPD "path to install SNMPD"

e.g.: install-package SNMPD /mnt/ snmpd

6. Cross Compilation

The following paragraphs describes how build the environment for cross compiling along with instruction on how to cross compile each part of the SW

6.1. Setting Cross Compilation Environment

The following steps will lead you how to set a cross compilation environment:

- Download the ARM tool chain *"arm-2008q1-126-arm-none-linux-gnueabi-i686-pc-linux-gnu.tar.bz2"* to present directory (see link below). http://en.sourceforge.jp/projects/emlinux/downloads/31638/arm-2008q1-126-arm-none-linux-gnueabi-i686-pc-linux-gnu.tar.bz2/

- mkdir /usr/local/arm
- cp arm-2008q1-126-arm-none-linux-gnueabi-i686-pc-linux-gnu.tar.bz2 /usr/local/arm/
- cd /usr/local/arm
- tar -jxf arm-2008q1-126-arm-none-linux-gnueabi-i686-pc-linux-gnu.tar.bz2
- Set the following variables in your shell environment:

  “export PATH=/usr/local/arm/arm-2008q1/bin:${PATH}”

  “export CROSS_COMPILE=arm-none-linux-gnueabi-“

  “export ARCH=arm”

Now the cross compilation environment is ready.

6.2. Cross Compiling Bootstrap

AT91Bootstrap does some minimal initialization of SDRAM and clock. Then it will check the presence of boot loader (u-boot). Boot loader will initialize UART (serial) and Ethernet. Then u-boot will check the presence of Linux kernel image. If it is present u-boot will load it from flash to DDR and starts execution. If Linux kernel image is not present, u-boot prompt will be shown.

e.g: u-boot>

The U-boot provides a rich set of commands to get hardware information, downloading images to memory, writing images to flash etc.

The Linux kernel will start execution and it will load all device drivers. Then Linux kernel will search for a root file system in compressed format on flash. The file system used here is based on Debian.
The following steps will lead you how to cross compilation a Bootstarp:

- Set the cross compilation environment as per the steps in section 6.1.
- Download the bootstrap source code.

  tar -xvzf   AT91Bootstrap-5series_1.2.tgz

- Before building Bootstrap, you have to configure it for 5series boards and to indicate where you want to store the environment.

  cd AT91Bootstrap-5series_1.2

  make at91sam9x5nf_dfconfig   →NAND FLASH

  make at91sam9x5sduboot_dfconfig →SDcard

- Then a configure file will be generated. You can customize AT91Bootstrap with

“make menuconfig”.

make

- The result of this operation is a new AT91Bootstrap binary located in the binaries directory and called at91sam9x5ek-xxxboot-3.1.bin
  - at91sam9x5ek-nandflashboot-3.1.bin is the binary file able to boot the application located in NAND FLASH.
  - at91sam9x5ek-sdcardboot-3.1.bin is the binary file able to boot the application called u-boot.bin located in the FAT formatted SD card. Rename the binary file as boot. In for the use in SDcard.

6.3. Cross Compiling u-boot

The following steps will lead you how to cross compilation a U-boot:

- Set the arm **cross compilation environment** as per the steps in section 6.1**.**
- Download the u-boot source.
- Extract the source with “tar xvjf u-boot-2010.06.tar.bz2”.
- cd u-boot-2010.06
- Apply the patch provided by **ATMEL** using the command *“cat u-boot-patch | patch -p1”*
- Before building U-Boot, you have to configure it for 5series boards and to indicate where you want to store the environment.

make at91sam9x5ek_sdcard_config → SDCARD

make at91sam9x5ek_nandflash_config → NAND FLASH

make

- The result of this operation is a fresh u-boot binary called u-boot.bin

6.4. Cross Compiling Linux Kernel 2.6.39

The following steps will lead you how to cross compilation a Linux Kernel:

- Set the arm **cross compilation environment** as per the steps in section 6.1**.**
- Download the Linux kernel source.
- Extract the source with “tar xvjf linux-2.6.39.tar.bz2”.
- cd linux-2.6.39
- Download the kernel patch.
- Extract the kernel patch with “tar xvzf 2.6.39-at91-exp.tar.gz”.

- Apply the patch using *"for p in 2.6.39-at91-exp/* ; do patch -p1 < $p ; done"*
- Firstly, use the 5series default kernel configuration

  make ARCH=arm at91sam9x5ek_defconfig
- Then you can customize the kernel configuration with the following steps:
- "make ARCH=arm menuconfig"   and make the below changes.
  - Enable ext2,ext3 and ext4 in File System.
  - Select the required features if any needed.
- vi .config and enable EXT2, EXT3 and EXT4 if not enabled.

  eg: CONFIG_EXT4_FS=y
- Run the following commands:

```
make ARCH=arm
mkimage -A arm -O linux -C none -T kernel -a 20008000 -e 20008000 -n linux-2.6 -d
arch/arm/boot/zImage image.bin
```

6.5. Building Debian file-system using Multistrap

The following steps will lead you how to cross compilation a Debian file-system:

- Install Debian/Ubuntu latest version in your PC
- Run apt-get update
- Install Qemu for arm ("sudo apt-get install qemu-user-static")
- Install multistrap("sudo apt-get install multistrap")
- create a multistrap configuration file as follows:

multistrap.conf

```
[General]
arch=armel
directory=multistrap/
unpack=true
noauth=true
debootstrap=Grip Networking Debian
aptsources=Debian

[Grip]
packages=nano
source=http://www.emdebian.org/grip
```

```
keyring=emdebian-archive-keyring
suite=squeeze

[Debian]
packages=
source=http://ftp.debian.org/debian
keyring=debian-archive-keyring
suite=squeeze

[Networking]
packages=net-tools  dhcp3-client ethtool
source=http://www.emdebian.org/grip
keyring=emdebian-archive-keyring
suite=squeeze
```

- Run the command " multistrap -f path_to_multistrap.conf"
- Run "cp /usr/bin/qemu-arm-static multistrap/usr/bin/"
- chroot multistrap/
- dpkg –configure -a
- Create a "multistrap/etc/hostname" file and just add "debian" as its content, then save and close that file.
- Create/edit "multistrap/etc/passwd" file and change/ paste the content as follows:

/etc/passwd

```
root::0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
```

```
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
telnetd:x:101:102::/nonexistent:/bin/false
```

Create/edit “multistrap/etc/inittab” and change/paste the content as follows:

/etc/inittab

```
# /etc/inittab: init(8) configuration.
# $Id: inittab,v 1.91 2002/01/25 13:35:21 miquels Exp $

# The default runlevel.
id:2:initdefault:

# Boot-time system configuration/initialization script.
# This is run first except when booting in emergency (-b) mode.
si::sysinit:/etc/init.d/rcS

# What to do in single-user mode.
~~:S:wait:/sbin/sulogin

# /etc/init.d executes the S and K scripts upon change
# of runlevel.
#
# Runlevel 0 is halt.
# Runlevel 1 is single-user.
# Runlevels 2-5 are multi-user.
# Runlevel 6 is reboot.

l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6
# Normally not reached, but fallthrough in case of emergency.
z6:6:respawn:/sbin/sulogin

# What to do when CTRL-ALT-DEL is pressed.
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now

# Action on special keypress (ALT-UpArrow).
#kb::kbrequest:/bin/echo “Keyboard Request–edit /etc/inittab to let this
work.”
```

```
# What to do when the power fails/returns.
pf::powerwait:/etc/init.d/powerfail start
pn::powerfailnow:/etc/init.d/powerfail now
po::powerokwait:/etc/init.d/powerfail stop

# /sbin/getty invocations for the runlevels.
#
# The “id” field MUST be the same as the last
# characters of the device (after “tty”).
#
# Format:
# <id>:<runlevels>:<action>:<process>
#
# Note that on most Debian systems tty7 is used by the X Window System,
# so if you want to add more getty’s go ahead but skip tty7 if you run X.
#
1:2345:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty2
3:23:respawn:/sbin/getty 38400 tty3
4:23:respawn:/sbin/getty 38400 tty4
5:23:respawn:/sbin/getty 38400 tty5
6:23:respawn:/sbin/getty 38400 tty6

# Example how to put a getty on a serial line (for a terminal)
#
T0:23:respawn:/sbin/getty -L ttyS0 115200 vt100
#T1:23:respawn:/sbin/getty -L ttyS1 9600 vt100

# Example how to put a getty on a modem line.
#
#T3:23:respawn:/sbin/mgetty -x0 -s 57600 ttyS3
```

After booting, it will be a Read-only File system. To make it read/write either of the following two steps can be done.

- Run “mount -o remount,rw /” to make it usable.
- Edit multistrap/etc/fstab and paste the following into it.

/etc/fstab

```
proc /proc proc defaults 0 0
/dev/mmcblk0p2  /         auto   errors=remount-ro   0   1
/dev/mmcblk0p1  /boot/uboot auto   defaults           0   0
```

6.5.1. File-system tuning steps (only for booting from NAND FLASH)

The following steps shows how to trim and tune the file system

```
apt-get install openjdk-6-jre
apt-get install sqlite3
apt-get install busybox
apt-get install openssl
apt-get clean
aptitude remove apt
rm -rf  /var/lib/apt/
rm -rf /var/lib/aptitude/
rm -rf  /var/cache/apt/
rm -rf /usr/share/
```

6.5.2. Modifying Post login message

Information to be printed just after login is found in /etc/motd.
The file /var/run/motd is a text file which contains a message or system identification to be printed before the login prompt

```
Distributor ID :  Debian
Description    :  Debian GNU/Linux 6.0.4 (squeeze)
Release        :  6.0.4
Codename       :  squeeze
kernel Version :  2.6.39
```

6.5.3. Modifying pre-login message

Information to be printed before login /etc/issue

```
Debian GNU/Linux 6.0 \n \l

       ____ ____
      |    | |     |
      |____| |___  |
      |    | |     |
      |    | |     |_____

EMDEBIAN CONTROLLER
```

The file /etc/issue is a text file which contains a message or system identification to be printed before the login prompt. It may contain various @char and \char sequences, if supported by getty

/etc/issue - escape code

The issue-file (/etc/issue or the file set with the -f option) may contain certain escape codes to display the system name, date and time etc. All escape codes consist of a backslash (\) immediately followed by one of the letters explained below.

- \b : Insert the baud rate of the current line.
- \d : Insert the current date.
- \s : Insert the system name, the name of the operating system.
- \l : Insert the name of the current tty line.
- \m : Insert the architecture identifier of the machine, eg. i486
- \n : Insert the node name of the machine, also known as the hostname.
- \o : Insert the domain name of the machine.
- \r : Insert the release number of the OS, eg. 1.1.9.
- \t : Insert the current time.
- \u : Insert the number of current users logged in.
- \U : Insert the string "1 user" or " users" where is the number of current users logged in.
- \v : Insert the version of the OS, eg. the build-date etc

6.5.4. Creating users and passwords

Enable user guest
The following command enabled a user guest:

```
useradd  -m  -r  -s /bin/bash  guest
```

- The flag **-m** is used to create the guest user’s home directory if it is not existing.
- The flag **-r** is used to create a system account. That is, a user with a UID lower than the value of UID_MIN defined in */etc/login.defs* and whose password does not expire.
- The flag **-s** is used to specify the name of the user's login shell. The default is to leave this field blank, which causes the system to select the default login shell.

*Note that **useradd** will not create a home directory for such an user, regardless of the default setting in /etc/login.defs. You have to specify **-m** option if you want a home directory for a system account to be created.*

Enable password for guest user:
The following command enabled password for user guest:

```
passwd guest
```

Then provide a password for the guest user. Verify the file /etc/passwd to see the following entry:

```
guest:x:999:999::/home/guest:/bin/bash
```

Enable password for root user:
To enable a root password use the command *passwd:*

```
passwd root
```

Then provide a password for the root user, verify the file /etc/passwd to see the following entry:

```
root:x:0:0:root:/root:/bin/bash
```

6.5.5. bashrc file – root

```
#configure the network
#ifconfig eth0 down
#ifconfig eth0 172.16.1.113
#ifconfig eth0 up
#route add default gw 172.16.0.1

#set the java path
export PATH=$PATH:/usr/lib/jamvm/java-1.6.0-openjdk/bin:/usr/lib/jamvm
export JAVA_HOME=usr/lib/jvm/java-1.6.0-openjdk/

#start the basic services
/usr/bin/ServiceManage
/usr/bin/dbinit
```

```
#Manage Basic service
/usr/bin/ServiceManage
```

6.5.6. Nandflashinfo

```
#!/bin/bash

df -h
```

6.5.7. Cpuinfo

```
#! /bin/bash
#Command will give the details of the CPU of the board

echo "CPU            : ARM Core"
echo "Architecture    : ARMv4"
echo "Family          : ARM9"
cat /proc/cpuinfo | grep Hardware
cat /proc/cpuinfo | grep Proce
echo "Core            : ARM 9263EJ-S"
echo "Max Speed       : 400 MHz"
echo "DataBus Width   : 32"
```

6.5.8. Boardinfo

```
#! /bin/bash

#Command for displaying the information of the board

echo "CPU Core         :  ARM926EJ-S"
echo "Chipset          :  AT91SAM9"
echo "Serial Interfaces :  SPI , I2c"
echo "Other Interfaces  :  ETH , USB , UART"
echo "RAM size         :  128 MB"
echo "Flash Size       :  256 MB"
echo "DataFlash        :  Yes"
echo "NandFlash        :  Yes"
echo "FlashBooting      :  Yes"
echo "SDCard Booting    :  Yes"
```

6.6. Running User Applications on ShiraTech Board

The following paragraph explains how a user can compile a C or JAVA coded application.

6.6.1. Cross compiling C code

Follow the below mentioned steps to cross compile C source code application

- Download the ARM tool chain *“arm-2008q1-126-arm-none-linux-gnueabi-i686-pc-linux-gnu.tar.bz2”* (from the link below) to present directory.
  http://en.sourceforge.jp/projects/emlinux/downloads/31638/arm-2008q1-126-arm-none-linux-gnueabi-i686-pc-linux-gnu.tar.bz2/
- Run the following commands:

```
mkdir /usr/local/arm

cp  arm-2008q1-126-arm-none-linux-gnueabi-i686-pc-linux-gnu.tar.bz2  /usr/local/arm/

cd /usr/local/arm

tar -jxf arm-2008q1-126-arm-none-linux-gnueabi-i686-pc-linux-gnu.tar.bz2
```

- Set the following PATH in your shell environment:
  “export PATH=/usr/local/arm/arm-2008q1/bin:${PATH}”
- Create a sample C program. For e.g. source.c
- Compile the program using:
  “arm-linux-gcc source.c -o source_output -static”
- Then the result will be a binary named “source_output”.
- Transfer the binary using one of the following options:
  - Using scp, transfer the source_output binary to the file-system:
    “scp –r source_output root@<shiratech_board_IPaddress>:/root/”
  - Using tftp, transfer the source_output binary to the file-system using,
    “tftp -g -r source_output   tftp_server_ip”
- Give execution permission to “source_output” using:
  “chmod +x source_output”
- run the binary using:
  “./source_output

6.6.2. Cross compiling Java code

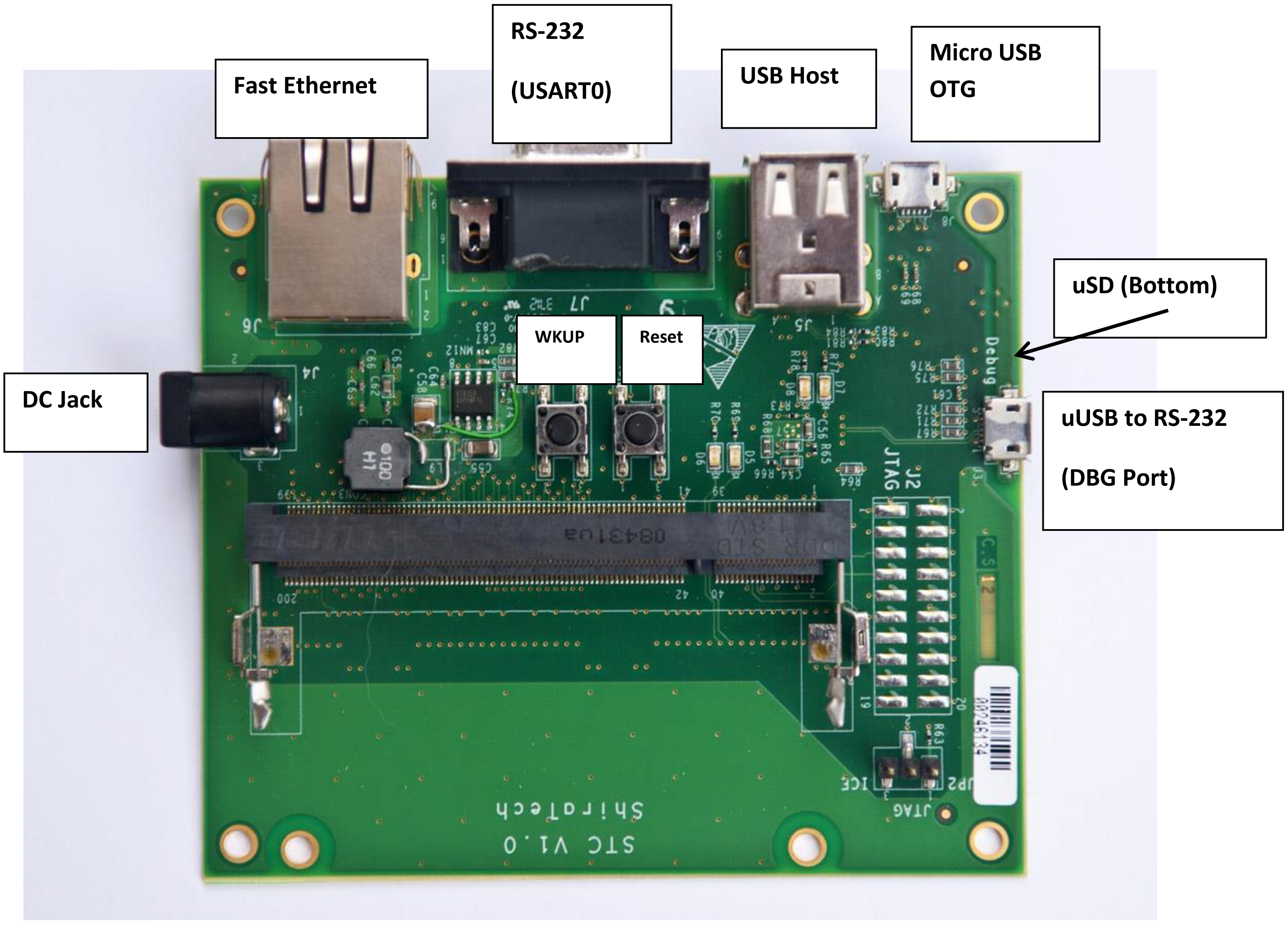
Follow the below mentioned steps to cross compile Java source code application

- For compiling java applications, we need java jdk version 1.6.0_24 in Ubuntu Linux machine
- Install java jdk version 1.6.0_24, using below command.
  “sudo apt-get install openjdk-6-jdk”

- Use “java –version” command to get the current jdk version.
- Compile the java application using,
  “javac Source_name.java”
- The result of the above command will get like
  “Source_name.class”
- There are two options to transfer the application
    - Using “scp –r source_output root@<shiratech_board_IPaddress>:/root/”
    - Using “tftp –g –r Source_name.class tftp_server_ip”
- run java application using,
  “java Source_name”

Appendix A – CB10 Starter Kit description

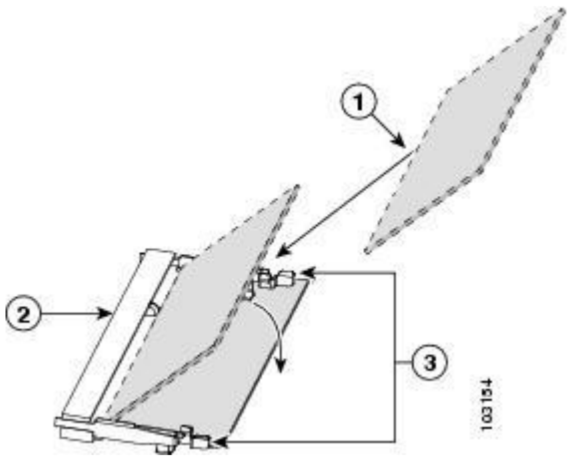
The CB-10 offers a small, low cost carrier board for ShiraTech SoMs. The board is used as a development platform offering all the relevant interfaces required for software development such as Ethernet, USB, RS-232 and debug port.



1. Setup process

Connecting the STC-200 to power:

1. Insert the SOM to the SO-DIMM 200 connector. First insert the SOM in a ~25 degree angle and then rotate it down. See the following picture.
2. Connect the AC adaptor to the DC jack.



Connecting to the SAM-BA tool:

1. Make sure there is no bootable SW in the NAND or in the SD card.
2. Connect a standard USB type A to Micro USB type A/B cable from the PC to the OTG Micro USB connector on the EVB (See picture above)
3. Start the SAM-BA SW.

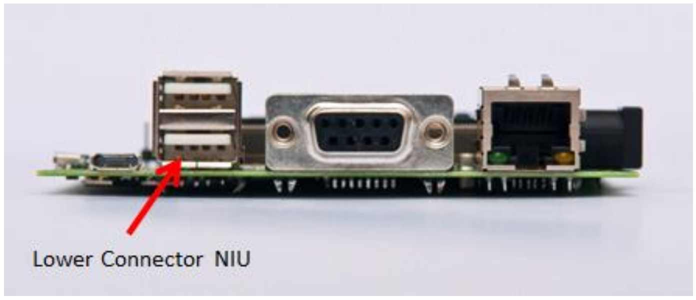
Connecting the debug port:

The EVB integrates a USB to RS-232 converter connected to the processor debug UART. Follow the following steps to connect a terminal to the EVB:

1. On the PC, make sure that you have a connection the Internet for driver download.
2. Connect a standard USB type A to Micro USB type A/B cable from the PC to the DEBUG Micro USB connector on the EVB (See picture above)
3. Wait until the chip driver is installed.
4. Open the Hyper-terminal application, select the right com and configure the port baud-rate to Baud rate – 115200, Data – 8 bits, Parity – None, Stop – 1 bit, Flow control – None .

Connecting a USB dongle:

For USB host applications only the upper USB type A port can be used(the lower port) see picture below.



1.1. Running from a Micro-SD card

The EVB boot priorities are:

1. Micro-SD card.
2. NAND
3. SAM-BA

It means that if a bootable SD is inserted it will be get priority over any other boot source thus the SD card can be used for development and for recovery in case the NAND flash get corrupted.

1. Insert a boot-able Micro-SD card to the Micro-SD card connector at the bottom of the EVB.
2. Reset the EVB.

1.2. Running from a NAND FLASH

To run from the NAND flash do the following steps:

1. Make sure there is no SD card in the Micro-SD card connector.
2. Load the Boot and U-Boot files to the Flash using the SAM-BA or other.
3. Reset the EVB.

Note – When the NAND Flash is programmed with a valid code, there is no way to boot from the SAM-BA only from the SD card.