OVERVIEW

The TS-Linux is a compact Linux distribution that comes installed by default in the on-board flash. It demonstrates the easy utilization of the TS-72XX ARM SBCs. This manual provides some basic instructions required to get the TS-Linux running the first time.

> **Warning**
> This document is applicable to the **TS-7200**, **TS-7250** and **TS-7260** models only. If you are using another board, please refer to the specific product's manual. The **Linux for ARM** section of this document can be used as a general guide though.

STARTUP

Console and Power Up

The TS-72XX SBCs have no video controller or keyboard interface. This was done to keep the board size small and the cost low. COM1 is typically used as a console port to interface the TS-72XX to a standard terminal emulation program on a Host PC.

An ANSI terminal or a PC running a terminal emulator is required to communicate with your Embedded PC. Simply connect an ANSI terminal (or emulator) to COM1 (DB9 female connector) using a null modem cable (this is included in the TS-ARM Development Kit), using serial parameters of 115,200 baud, 8 data bits, no parity, no flow control, 1 stop bit (8N1), and make sure jumper **JP2** is installed. If you are running Linux, the minicom program works well, Windows users can run the Hyperterm application. Technologic Systems offers a null modem cable with both 25 pin and 9 pin connectors at each end as part number CB7-05. Some systems also require the 10-pin header to 9-pin Sub-D adapter which is P/N: RC-DB9.

The console can be changed to COM2 by installing **JP4** (with JP2 also installed). If your application does not require a console or both COM ports are required, then removing the jumper **JP2** easily disables all console output.

Connect a regulated 5VDC, (1A minimum) power source using the included 2 screw terminal strip/connector. Please note the polarity printed on the board. The boot messages, by default, are all displayed on COM1 at 115200 baud.

Boot Sequence

The boot sequence has four distinct stages: TS-BOOTROM messages, RedBoot ROM monitor messages, Linux Kernel messages and Login prompts.

After applying 5VDC power, the board mounted LED will blink, followed immediately by the display of TS-BOOTROM messages and RedBoot messages. RedBoot, if not interrupted by the user within one second, loads the Linux kernel from flash and boots to the on-board flash chip.

> **Warning**
> Ensure that JP5 is removed when booting from the default flash file-system. Special scripts look for it and, when found, run special test programs that will use up system resources.

eCos/RedBoot

RedBoot is a feature rich boot-ROM monitor, that allows manipulation of the on-board flash, JFFS and YAFFS images, loading and execution of a kernel or executable from either tftp (trivial ftp), http or flash, and gdb debugging stubs. From RedBoot, one can load and execute any standalone binary. Most commonly, a Linux kernel binary is used. One can also write applications within the eCos environment and load them with RedBoot.

Using Redboot

By default, a pre-existing RedBoot script is executed on initialization time, if not interrupted by the user within one second. The default script instructs RedBoot to load the Linux kernel from the flash, and instructs the Linux kernel to use the JFFS/YAFFS image on the flash chip for its root file system. One can view the RedBoot defaults for the board, as well as the default script, by entering “fconfig -l” at the RedBoot command prompt (Ctrl+C within one second after power up).

The defaults can be changed by simply entering “fconfig” at the RedBoot prompt and answering the prompts. A final chance to write or discard the changes to the board will be given by RedBoot. Also, the main RedBoot commands can be viewed by entering “help” at the prompt, and further information about a single command can be viewed by typing “help <command name>”.

Loading and execute kernel from RedBoot

RedBoot can load a kernel or executable via the serial console, a tftp server, http server, or directly from flash.

Important
The Linux kernel must be loaded into memory address 0x00218000.

FLASH

Loading the kernel from flash is done automatically by RedBoot in the default script with the following command:

```
RedBoot> fis load zimage -b 0x00218000
```

HTTP

Prior to loading a new kernel into RebBoot using HTTP, make sure you have configured RedBoot with a network configuration that can reach the network. You may use the RedBoot "fconfig" command to set network parameters.

Get to the RedBoot prompt by hitting Ctrl+C key immediately after power on and type the following command:

```
load -v -r -b 0x00218000 -m http -h <http sever IP> <kernel name>
```

For example:

load -v -r -b 0x00218000 -m http -h 67.40.67.44 /ftp/ts7kv/vmlinuxts7200ts9.bin

TFTP

To load a kernel from a simple TFTP server, the following command is needed:

```
load -r -b 0x00218000 -h <tftp server IP> <kernel name>
```

For example:

```
load -b 0x00218000 -h 192.168.0.1 vmlinux
```

Executing the Kernel and Booting the Root File System

Now that a kernel has been loaded into memory, it can be executed. This is accomplished with the following command:

```
exec -c "<kernel parameters>”
```

For example:

```
exec -c “console=ttyAM0,115200 ip=dhcp root=/dev/mtdblock1”
```

The “exec” command executes the loaded kernel image, passing to the kernel the arguments specified via the “-c” switch. In the previous example, kernel messages are sent out on the first serial port (note that ttyAM0 is used instead of the familiar ttyS0) at 115200 baud and the root file-system is on the first mtdblock of the flash chip.

On the TS-7200 model, to load the root file system from the Compact Flash card, the following command should be used instead:

```
exec -c “console=ttyAM0,115200 ip=dhcp root=/dev/hda1”
```

To load a NFS Root file system from a NFS server, use the following command:

```
exec -c “console=ttyAM0,115200 ip=dhcp nfsroot=<IP of NFS server>:</path/to/NFSROOT>
```

LINUX FOR ARM OS SUPPORT: TS-LINUX DISTRIBUTION

The ARM processor (the EP9302) comes from Cirrus and the platform is very similar to the Cirrus EDB9302 evaluation board. Cirrus has strongly promoted running Linux on this chip and has done most of the legwork in creating a patch set to the Linux 2.4 kernels, but we have also had to modify the Linux Kernel (**TS-Kernel**) so it can support the 8MB on-board Flash chip (via mtd drivers), the compact flash IDE driver, and the A/D converter. If you want to use Linux and aren't tied to the x86 architecture, the TS-72XX boards can be very cost-effective.

The TS-72XX SBCs are shipped standard with the compact **TS-Linux** embedded operating system installed in the on-board Flash memory. The full-featured **Debian Linux** can also be used with an NFS root file system or larger Flash drives, such as Compact Flash cards, SD cards and USB flash drives. The **TS-Kernel** used is based upon the version 2.4.26, patched and compiled for the Cirrus EP9301 ARM920T processor, and is real-time capable through RTAI.

The root file system used by the Linux OS can be any of the following:

- ✔ JFFS/YAFFS file system image in the on-board Flash (RedBoot should include the option root=/dev/mtdblock1 to instruct the kernel to boot here)
- ✔ EXT2 file system image in the Compact Flash card (RedBoot should include the option root=/dev/hda)
- ✔ NFS root, via Ethernet port (RedBoot should include the option root=/dev/nfs nfsroot=<IP>:<DIRECTORY> ip=dhcp)

Logging In and Basic Commands

After the desired Linux Kernel is loaded and executed through RedBoot, the file system loads and networking, logging, Apache web server, etc. are all started. When the login prompt is displayed, type “**root**” to login, with no password. A Bash login prompt will then appear. At this point, you are ready to enjoy your TS-72XX SBC running Linux. Some very basic commands for one beginner user to start using Linux are:

- ✔ pwd: informs the current directory
- ✔ ls: lists current directory contents
- ✔ cd: changes directory
- ✔ man: accesses the system's manual pages of a given command
- ✔ cat: displays the entire content of a given file
- ✔ vi: Linux most common file editor (reading further documentation is recommended)

The most common file handling commands are “cp”, “mv”, “rm”, “mkdir”. Help information is provided by supplying “--help” to any given command, for example “cp –help”

Shutdown

Use the "shutdown –h now" command to halt the Linux system when running from Compact Flash, SD or USB memory card to avoid a potentially lengthy file system check on the next boot, since the file system running is EXT2 formatted.

On the other hand, the JFFS/YAFFS file systems are highly tolerant of power cycles while the file systems are mounted. Therefore, the "shutdown" command is not required when the root file system is JFFS/YAFFS, but is still recommended.

Initialization Scripts

After the "exec" command on RedBoot, the Kernel boots and drivers are loaded. Then, the initialization process reads the file "/etc/inittab". The inittab file will call "/etc/rc.d/rcS.sysinit" as part of the system initialization. The run level then defaults to 3, which will run the "/etc/rcS" script and call all the scripts linked in the "/etc/rc3.d/" directory in numerical order. For example, the following are the initialization scripts for run level 3 found at TS-Linux:

```
/etc/rc.d/rc3.d# ls
S10Network  S11portmap  S20inetd  S30telnetd  S40apache
```

Changing the run level or re-invoking the initialization scripts is possible through the "init" command. A "halt" or "reboot" command will change the run level to 0 or 6 and execute the "/etc/rc0.d" scripts or "etc/rc6.d" scripts respectively.

Network Setup

The main utilities for network configuration under Linux are:

- ifconfig: prints network settings and configures ethernet interfaces
- ifup: turns given network interface up
- ifdown: turns given network interface down

Entering "ifconfig" shows the current ethernet settings. These utilities require a network device as parameter. On Linux, the ethernet devices are generally named eth0, eth1, etc. Therefore, the command "ifup eth0" or "ifconfig eth0 up" brings up the on-board ethernet interface on TS-72XX SBCs.

To configure the network, you need to manage the proper configuration files. On TS-Linux systems, these files are located in the "/etc/sysconfig/" directory. By default, Linux systems on TS-72XX boards are configured to assign the IP 192.168.0.50 to the on-board ethernet interface.

Setting Up the networking with TS-Linux

To configure the network when booting to the TS-Linux image on the flash chip, the files in "/etc/sysconfig/" must be edited. Network interfaces are configured on a file per interface basis. The first Ethernet device, eth0, is controlled by the file "/etc/sysconfig/ifcfg-eth0". An example of "ifcfg-eth0" is shown below:

```
DEVICE=eth0 #Name of ethernet interface
IPADDR=192.168.0.50 #IP address of this ethernet interface
NETMASK=255.255.255.0 #Used with NETWORK to determine local IPs
NETWORK=192.168.0.0 #Used with NETMASK to determine local IPs
BROADCAST=192.168.0.255 #Broadcast IP for system wide messages
BOOTPROTO=static #Static IP (change "static" to "DHCP")
ENABLE=yes #Load device on boot
```

The TCP/IP network settings are configured in the file '/etc/sysconfig/network_cfg', here is a listing:

```
NETWORKING=yes #Enable networking on startup
GATEWAY="192.168.0.1" #Gateway for internet access
GW_DEV=eth0 #Default gateway
HOSTNAME=ts7200 #Host name of this computer
BOOTPROTO=no
FORWARD_IPV4=no
DEFRAG_IPV4=no
```

The TCP/IP name resolution server is configured in '/etc/resolv.conf'. Here is a listing:

```
Nameserver 192.168.0.1 #Name server for domain name lookups
```

Those lines starting with a # symbol are comments. As the above example shows, eth0 is given the static address of 192.168.0.50. If one wishes eth0 to obtain its IP from a DHCP server, then change the line BOOTPROTO=static to BOOTPROTO=dhcp

Note
In order to test the default network settings with TS-Linux, open a web browser and use the embedded Apache web server by entering the default IP 192.168.0.50, or simple "ping" or "telnet" to 192.168.0.50.

Network Services

TS-Linux includes solutions for the main network services, including Telnet, HTTP, FTP, SSH, NFS and Mail. Some of these services can be started, restarted or stopped by management scripts located at the "/etc/init.d" directory. For example, the following command will restart the apache server:

```
/etc/init.d/apache restart
```

Also, the "/etc/inet.conf" file is used to configure the initialization and parameters of other services.

On-board Flash File Systems

The on-board flash contains a custom-made JFFS or YAFFS formatted file system image. JFFS2 is a compressed, Journaling Flash File System, for the NOR Strata Flash memory on TS-7200. The YAFFS file system is used for the on-board flash on the TS-7250 and TS-7260, which utilize NAND flash technology.

The NOR and the NAND flash devices are divided in three partitions. The first is a 16KB sized partition that contains the TS-BOOTROM. The last partition, which occupies 2MB-3MB, contains the eCos/RedBoot system and the Linux Kernel. The remaining space in the middle partition is either for the JFSS or the YAFFS image used as the root file system. The following shows how a 8MB NOR chip is recognized by the Linux Kernel MTD system on a TS-7200:

```
TS-7200 flash: Found 1 x16 devices at 0x0 in 16-bit bank
 Intel/Sharp Extended Query Table at 0x0031
Creating 3 MTD partitions on "TS-7200 flash":
0x00000000-0x00020000 : "TS-BOOTROM"
0x00020000-0x00620000 : "Linux"
0x00620000-0x00800000 : "RedBoot"
```

In both NOR and NAND cases, the Linux block device entry corresponding to the on-board flash is the "/dev/mtdblock". Therefore, the following command can be used to mount the flash file system if you are not booted to the on-board flash:

```
mount /dev/mtdblock/1 /mnt
```

Network File System - NFS

NFS uses the Ethernet connection to mount the root file system for the SBC as an exported directory on a Linux host PC. This bypasses the need to download any files from the host to the SBC, because any file needed on the SBC is just copied to the exported directory on the host system and is instantly available to the embedded PC. Mounting the root file system via NFS allows the developer to use the editors, compilers, etc. from the single board computer.

Mounting NFS roots requires that the ”portmap” daemon is running before executing the mount command. The following example demonstrates mounting an NFS file system hosted on a server at 192.168.0.1

```
portmap &
mount -t nfs 192.168.0.1:/path/to/nfsroot /mnt
```

Setting Up an NFS File System

To mount the root file system via NFS, Prepare the exported root directory on the server:

1) Download the pre-made Debian tarfile from our website or the TS-ARM Linux CD to your Linux host machine.

2) Untar NFS root package to a directory on the host machine.

```
tar -C /path/to/nfsroot -xvjf /path/to/debian-file-system.tar.bz2
```

3) Export the directory by adding the following line to the file “/etc/exports”

```
/path/to/nfsroot 192.168.0.0/255.255.255.0
(rw,no_root_squash,insecure)
```

Note: The IP mask in the above example will only allow NFS connections from computers having an IP starting with 192.168.0. Substitute the appropriate parameters for your local network.

4) Restart NFS so the directory is available for export. Typically, this would be:

```
/etc/init.d/nfs-server restart
```

5) Modify the /path/to/nfsroot/etc/fstab file on your host system for your local network settings.

```
Add: "192.168.0.1:/path/to/nfsroot / nfs exec,dev,suid 1 1"
Comment out: "/dev/hda1 / ext2 defaults 1 1"
```

6) Load the kernel from Redboot with the following command line options:

```
fis load vmlinux
exec -c “console=ttyAM1,115200 ip=dhcp
nfsroot=192.168.0.1:/path/to/nfsroot”
```

CONTACT TECHNOLOGIC SYSTEMS

16610 East Laser Drive #10
Fountain Hills, AZ 85268
TEL 1.480.837.5200
FAX 1.480.837.5300

www.embeddedARM.com
support@embeddedARM.com

DOCUMENT HISTORY

05.19.2006 – CREATED
07.05.2007 – Rename series to TS-72XX, only applicable to these boards.