An Autonomous Underwater Vehicle
CPU Programming How To Guide

Roger Cortesi

Document Version 0.1: May 5, 2006

Contents

1 Introduction

This document serves as a guide to get students familar with processor for the Autonomous Underwater Vehicle (AUV). This document assumes minimal understanding of Linux. The students should have an understanding of C programming. Additionally References [?],[?], and [?] should be available as they provide the information critical to the understanding of this document and this processor

The processor must control the following equipment on the vehicle:

- Depth Cell (analog output)
- 2 to 4 thrusters (analog input/output)
- Doppler Velocimeter Log (DVL) (serial input/output)
- Compass (serial input/output)
- 1 or 2 Cameras (serial input/output)
- Control Surfaces (serial input/output)
- Payload Release Device (binary input)

The vehicle processor is a TS-7200 Single Board Computer (SBC) manufactured by Technologic Systems. The TS-7200 is an ARM9 processor on a PC104 board running Linux with the following features:

INSERT PHOTO OF TS-7200 HERE

- 200 MHz ARM9 Processor
- 32 Megs of RAM
- 16 megs of onboard flash memory for a file system
- Compact Flash Card for for a file system
- 1 10 Base Ethernet Port
- 2 USB (1.1) Ports
- 2 Serial Ports
- Power Consumption 1.8 Watts at 5 VDC
- 8 Channels of 12 bit A/D Conversion
- PC104 ISA connector and mounting holes to allow expansion
- Appoximmately $200 each (without the CF Card)

The TS-7200's low power consumption and cost make it a good choise for a vehicle computer. The computer's power consumption can be further reduced to less than one watt if the onboard ethernet circuits are powered down.

Users connect and use the TS-7200 via a text console via a serial connection or via the network using a secure shell (ssh or PuTTY).

The TS-7200 can be configured to used either the onboard flash memory or the CF card as its root file system.

CF as Root File System Advantages	Onboard Flash as Root File System Advantages
"Unlimited" File System Size	Faster Boot Time
Code can be compiled directly on the vehicle's SBC.	Robust Handling of Loss of Power

For vehicle development we have elected to use the CF card as the root file system. This means that all the development can be done on the vehicle computer itself. If the vehicle was going to be put in a more production-like environment, then it might be useful to move the root file system over the onboard flash to prevent corruption on a sudden loss of power.

Using the CF card as the root file system allows a complete Debian Linux distribution to be used. This means that almost the full range of applications can be used for development and running the vehicle. Applications that we have found to be useful for vehicle development include:

vim is a very capable text editor. It definately has a learning curve. However, it supports syntax highlighting, editing multiple files at one, advanced search and replace, running shell commands from within the editor, and many other features which make it useful for code development. See Section **??** for more information and examples using VIM.

minicom is a termial program like Hyper-Terminal, but much better. This is really useful for learning and trouble-shooting hardware that is connected to the SBC via the serial ports. See Section **??** for more information and exaples of minicom useage.

screen is a 'terminal multiplexing' program. It allows you to have multiple windows in a single text console and two swap back and forth between them. This is very useful for running multiple programs. It also allows you to save a session, log out, and return to it later.

rsync performs incremental file transfers across the network. Only the difference between the two files are transfered. This is very useful for performing a backup of vehicle's complete file system since only the changes since the last backup are sent.

scp is a program for securely copying files across the network. It is easier to use than rsync but it always copies whole files.

gcc is the GNU Compiler Collection is a series of tools for compiling, linking, debugging, and profiling C and C++ code.

2 Configuring the TS-7200

2.1 Configuring the RedBoot boot loader

When the TS-7200 is powered up, the RedBoot boot loader loads the linux kernel into RAM, passes configuration parameters to it and executes the kernel, passing control of the SBC from RedBoot to Linux. Users interface with RedBoot over the first serial port at 115200 Baud[1]. Pressing Control-C as RedBoot starts halts the boot process and returns a RedBoot prompt.

The default configuration shipped with the TS-7200 is to use the onboard flash as the root file system. To change the the RedBoot boot script must be changed. At the RedBoot prompt, the command **fconfig** is used to reconfigure RedBoot's boot script. Below is the boot script for booting with the CF card as the root file system.

```
fis load vmlinux.bin
exec -c "console=ttyAM0,115200 root=/dev/hda1"
```

Listing 1: Redboot boot script for using CF card as the root file system.

The first line loads the kernel vmlinux.bin into ram. The second line executes the kernel with the console on ttyAM0 at 115200 baud and the root file system at "*/dev/hda1*" (the CF Card). Listing 2.1 show the RedBoot boot script for mounting the onboard flash as the root file system.

```
fis load vmlinux.bin
exec -c "console=ttyAM0,115200 root=/dev/mtdblock1"
```

Listing 2: Redboot boot script for using onboard flash memory as the root file system.

See Reference [**?**] for more information on configuring and using RedBoot with the TS-7200.

2.2 Building a new CF Card

The Debian file system for the CF card is distributed from Technologic System's website. Uncompressed it takes up approximately 200 megs of space. So at least a 512 meg CF card should be use to allow adequate space for applications and data.

The summary of the steps to build new file system on a compact flash are:

1. Partition the CF card.
2. Format the CF card with the ext2 file system.
3. Mount the CF Card.
4. Download the current file system tarball from Technologic Systems website.
5. As root, unpack the tarball to the CF card.

[1]The first serial port is has the DB-9 connector on the board labeled COM1. It is referred to as */dev/ttyAM0* by Linux.

6. Make any configuration changes to the system files now on the CF card.
7. Install the card in the TS-7200, boot the SBC and install any additional packages which are required (requires a network connection).

Use a desktop or laptop computer to prepare the compact flash card. Currently I'm using a laptop running the same version of Debian (Woody) as the SBC. The downside to this is that CF card support is not as seamless as it could be. Here is an article on getting a CF card reader running under linux, http://www.cs.sfu.ca/∼ggbaker/personal/cf-linux . More modern Linux distributions should have better CF card support.

My CF card reader is recognized as the 2nd SCSI device /dev/sdb and gets mounted to "/media/cf".

The page 10 of Reference **??** also explains how to build the file system on a CF card.

fdisk is a utility for paritioning disks. You need to run fdisk on the raw unmounted device (i.e. /dev/sda or /dev/sdb).

```
$fdisk /dev/sdb
```

Here is the squences of fdisk commands to partition the CF card. Keystrokes are in paraenthisis:

1. Delete the current partition (d)
2. Create a new primary partition (n,p,1)
3. Use the default start and stop cyclinders
4. view the partition table (p)
5. If there is NOT an * in the boot column press (a,1)
6. Write the changes (w) and quit

Once the CF card is partitioned, format it as ext2 with the **mkft.ext2** command

```
# mkfs.ext2 /dev/sdb1
```

Note the '1' appended on the end of the device file you partitioned with fdisk. This refers to the first partition. Once the card has been formatted it can be mounted.

Make sure the file */etc/fstab* as the line

```
/dev/sdb1 /media/cf auto rw,user,noauto 0 0
```

This will allow the mounting of the CF card to the directory */media/cd*. If this directory does not exist, create it. Once these steps have been completed mount the CF card to the */media/cf* directory with the command:

```
$ mount /media/cf
```

Next unpack the tarball on the CF card. The tar command should be run as root and the -p flag should be used. This creates the files on the card with the proper root ownership and permissions.

```
# tar -C /media/cf -pxvf debian256-10-28-2004.tar
```

When this is finished unmount the card with the command "umount /media/cf". You should wait until the prompt returns after the tar command **and** the light on the CF card reader is off, indicating the write to the card is complete.

2.3 Configuring Consoles

2.4 Network Setup

During development the vehicle's computers have static IP addresses assocated with local domain names so we can easily connect to and transfer files between them.

To configure a static IP address set the eht0 section of */etc/network/interfaces* to

```
iface eth0 inet static
address xxx.xxx.xx.x
netmask 255.255.252.0
gateway 131.122.80.254
```

Where xxx.xxx.xx.x is the IP address to the particlar domain name assigned by IT services. If you are not on our network, then your netmask and gateway addresses will also be different.

2.5 Securing the CPU

By default the TS-7200 has **telnet** and **ftp** enabled. These are both insecure services. To disable them, comment out the appropriate lines of */etc/inetd.conf* with a # at the start of the line.

Connect to the TS-7200 over the network using **ssh** or **PuTTY** (not telnet). To transfer files use **scp** or **rsync**, not ftp.

2.6 Adding and Switching Users

The default configuration of the TS-7200 has no users (except root), and the root account has no password. **You must set the root password!!!**

When the SBC is first booted with a new file system login as root and set a new password with **passwd** command. You will be prompted twice for the new password.

As root you can also create additional user accounts using the **adduser** command. **adduser** is an easier interface to the functions provided by **useradd**. Chapter ? of Reference [**?**] as much more information on user account configuration.

2.7 Allowing Users Root Access

The command **su** allows a user to switch to a different user. **su** with no arguments will switch to root (after prompting for root's password[2]). Users may also switch to another user by specifying a username after **su**.

Root access is required to perform most system configuration and administrative functions. Currently root access is also required to access raw hardware memory directly. Many of the TS-7200's features are controlled by reading and writing to raw memory addresses. These features include the onboard A/D converter, the watchdog timer, the LEDs, powering up and down the ethernet circuit, and using the D/A extension card.

2.8 Serial Port Permissions

The file permissions for the serial port device files are often set such that normal users cannot access the ports. This can be fixed by login in as root and changing the file permissions with the **chmod** command. However, these changes will **not** be persistant across reboots.

The default file permissions for the ports are specified in the configuration file */etc/devfs/perms*. The file consists of lines of regular expressions that match the device files followed by their owner, group and permissions. The serial ports typically have their permissions set to 0660, thereby allowing only the port's owners and groups access. Setting the ports permissions to 0666 allows all users to access the port. See Chapter 4 of *Running Linux* [**?**] for more information on file permissions.

The following lines in */etc/devfs/perms* have had their permissions changed to 0666.

```
REGISTER \^ttyAM[1-9]+\$ PERMISSIONS root.tty 0666
...
REGISTER \^(tts|cua)/[\^/]*\$ PERMISSIONS root.dialout 0666
```

Note that the device file ttyAM0 has **not** had its permissions changed. This device is a serial console interface to the TS-7200, so userland programs should be able to use that port. To make sure that only ttyAM0 is configured as a serial console and that the SBC's second serial port (ttyAM1) is not. The default configuration is to have ttyAM0 and ttyAM1 cofigured as consoles. This can be edited in the file */etc/inittab*. Below are the relavant lines of */etc/inittab* which configure ttyAM0 as a console and ttyAM1 as a regular serial port

```
T1:23:respawn:/sbin/getty -L ttyAM0 115200 vt100
#T1:23:respawn:/sbin/getty -L ttyAM1 115200 vt100
```

The line for ttyAM1 has been commented out with a # sign and the ttyAM0 is set to communicate at 115200 baud as a vt100 terminal.

2.9 Troubleshooting Serial Ports

minicom is an execellent terminal program available on the TS-7200. Only root can specify which port minicom should use. Once the default configuration is saved, then regular users can use minicom

[2] If a user is a member of the group "wheel" as specified in */etc/group*, then no password is required to switch to the root user.

over that port in any configuration. Additionally alternate configurations can be saved for different ports, that regular users can access multiple ports with **minicom**. See the Section **??** for more information.

If user programs (minicom or custom code) can use the second onboard serial port (ttyAM1), check to make sure that ttyAM1 is not configured as a serial console in */etc/inittab*. See Section 2.8 for more information.

The following sub-sections discuss troubleshooting various PC104 multi port serial cards we have used.

2.9.1 Technologic Systems TS-SER4 Card

Check to make sure the card is detected on boot. RedBoot will print a message on the serial console.

```
Found COM3, COM4, COM5, COM6''
```

Login in to the TS-7200 and verify the file */dev/ttyS0* through */dev/ttyS3* exist. They will exist if the kernel has successfully loaded the driver for the card. These files are actually just soft links to the files 0, 1, 2, and 3 in the directory */dev/tts*. If these files are **not** found, then it is likely that the jumpers are misconfigured on the TS-SER4. **Jumpers should be installed on COM1, IRQ2, IRQ4, COM C: RS-232, and COM D: RS-232**

If data comes across the ports slowly and sporadically, then it is likely that the IRQ's are setup incorrectly. This is particularly evident by connecting a device that supplies a steady stream of data (gps, compass, or dvl) to one of the ports and reading the data with **minicom**. If the data does not being read smoothly and it has a lot of dropped characters, then investigate the IRQ settings. The command **setserial** is particularly useful for this. The -a option displays all the information about the port specified port.

```
$ setserial -a /dev/ttyS0
/dev/ttyS0, Line 0, UART: 16550A, Port: 0x89c003f8, IRQ: 33
        Baud_base: 115200, close_delay: 50, divisor: 0
        closing_wait: 3000
        Flags: spd_normal skip_test
```

Listing 3: A setserial listing for the configuration of /dev/ttyS0.

The listed IRQ should match the IRQ specified by the jumpers on the serial expansio card… sort of. Since Linux is primarily an x86 based operating system, it uses x86 IRQ numbers regardless of the underlying hardware. The IRQ numbers for an ARM processor (like the TS-7200) are different. This requires one to do some translation of IRQ numbers.

x86 IRQ	ARM IRQ
33	6
40	7

On the TS-SER4 the IRQ numbers are listed as the ARM IRQ numbers and the IRQ value is the sum of the jumpers. Jumpering IRQ pins 2 and 4 sets ARM IRQ 6. These ports should then be configured with IRQ 33 in Linux using **setserial**. For example " setserial /dev/ttyS0 irq 33".

2.9.2 Connect Tech Xtreme/104 card

The Connect Tech Xtreme/104 card is an 8 port PC104 serial expansion card. The default Linux kernel shipped with the TS-7200 is only configured to use 4 additional serial ports. To use more serial ports the kernel must be recompiled. See Section 2.11 on building custom kernels for the TS-7200.

To allow more than 4 additional serial ports to be used with the TS-7200, the kernel source files *serial.h* and *regmap.h* must be editted. These files are located in the directory *include/asm-arm/arch-ep93xx* in the kernel source tree.

In *regmap.h* I replaced the lines

```
#define TS7XXX_TTYS1 (TS7XXX_IO8_BASE + 0x01C003f8) etc....
```

with

```
#define TS7XXX_TTYS0 (TS7XXX_IO8_BASE + 0x01C00300)
#define TS7XXX_TTYS1 (TS7XXX_IO8_BASE + 0x01C00308)
#define TS7XXX_TTYS2 (TS7XXX_IO8_BASE + 0x01C00310)
#define TS7XXX_TTYS3 (TS7XXX_IO8_BASE + 0x01C00318)
#define TS7XXX_TTYS4 (TS7XXX_IO8_BASE + 0x01C00320)
#define TS7XXX_TTYS5 (TS7XXX_IO8_BASE + 0x01C00328)
#define TS7XXX_TTYS6 (TS7XXX_IO8_BASE + 0x01C00330)
#define TS7XXX_TTYS7 (TS7XXX_IO8_BASE + 0x01C00338)
```

The last three digits of each address is found in the documentation for the Xtreme/104 card, and they are jumper dependant. The jumper configuration is no jumpers on J1 and all pins jumpered on J2 (setting all ports to RS-232).

The following changes were made in *serial.h*:

Set the value of "RS_TABLE_SIZE" to 8. This determines the maximum number of additional serial ports the kernel will create. Set the default IRQs for each port.

```
/* UART CLK PORT IRQ FLAGS */ #define STD_SERIAL_PORT_DEFNS \
 { 0, BASE_BAUD, TS7XXX_TTYS0, 40, STD_COM_FLAGS }, /* ttyS0 */ \
 { 0, BASE_BAUD, TS7XXX_TTYS1, 40, STD_COM_FLAGS }, /* ttyS1 */ \
 { 0, BASE_BAUD, TS7XXX_TTYS2, 40, STD_COM_FLAGS }, /* ttyS2 */ \
 { 0, BASE_BAUD, TS7XXX_TTYS3, 40, STD_COM_FLAGS }, /* ttyS3 */ \
 { 0, BASE_BAUD, TS7XXX_TTYS4, 40, STD_COM_FLAGS }, /* ttyS4 */ \
 { 0, BASE_BAUD, TS7XXX_TTYS5, 40, STD_COM_FLAGS }, /* ttyS5 */ \
 { 0, BASE_BAUD, TS7XXX_TTYS6, 40, STD_COM_FLAGS }, /* ttyS6 */ \
 { 0, BASE_BAUD, TS7XXX_TTYS7, 40, STD_COM_FLAGS }, /* ttyS7 */ \
```

Where 40 replaces _SER_IRQ1 and _SER_IRQ0 in the original file. This this example the additional ports have been set to x86 IRQ 40 (ARM IRQ 7).

2.10 Startup Scripts

2.11 Building A Custom Kernel

2.12 Installing a New Kernel

The RedBoot boot manager can use used to temporarily or permenantly load a new kernel for uses by the TS-7200.

2.12.1 Temporarily Loading the Kernel

Once the new kernel (called vmlinux.bin this this example) has been compiled copy it to a location on the CF card. The root of the CF

2.13 User and Application Specific Configuration

Index