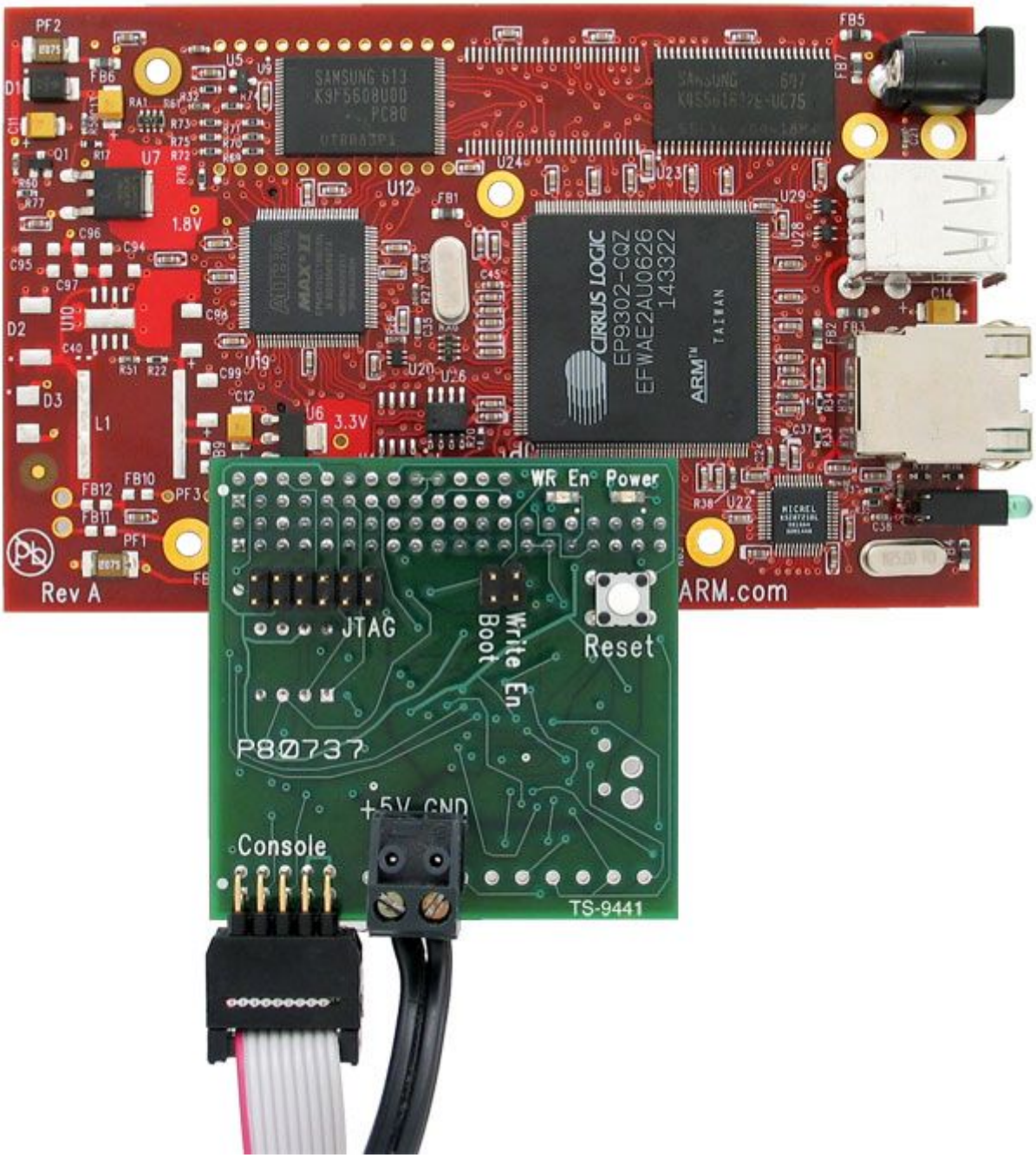
www.embeddedARM.com

TS-7400/TS-9441 Manual
Hardware & Software



Technologic SYSTEMS

1.4

Feedback and Update to this Manual

To help our customers make the most of our products, we are continually making additional and updated resources available on the **Technologic Systems website** (www.embeddedARM.com).

These include manuals, application notes, programming examples, and updated software and firmware. Check in periodically to see what's new!

When we are prioritizing work on these updated resources, feedback from customers (and prospective customers) is the number one influence. If you have questions, comments, or concerns about your Embedded Computer, please let us know at **support@embeddedARM.com**.

Limited Warranty

Technologic Systems warrants this product to be free of defects in material and workmanship for a period of one year from date of purchase.

During this warranty period Technologic Systems will repair or replace the defective unit in accordance with the following process:

A copy of the original invoice must be included when returning the defective unit to Technologic Systems, Inc.

This limited warranty does not cover damages resulting from lightning or other power surges, misuse, abuse, abnormal conditions of operation, or attempts to alter or modify the function of the product.

This warranty is limited to the repair or replacement of the defective unit. In no event shall Technologic Systems be liable or responsible for any loss or damages, including but not limited to any lost profits, incidental or consequential damages, loss of business, or anticipatory profits arising from the use or inability to use this product.

Repairs made after the expiration of the warranty period are subject to a repair charge and the cost of return shipping. Please, contact Technologic Systems to arrange for any repair service and to obtain repair charge information.

FCC Advisory Statement

This equipment generates, uses, and can radiate radio frequency energy and if not installed and used properly (that is, in strict accordance with the manufacturer's instructions), may cause interference to radio and television reception. It has been type tested and found to comply with the limits for a Class A computing device in accordance with the specifications in Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference, in which case the owner will be required to correct the interference at his own expense.

If this equipment does cause interference, which can be determined by turning the unit on and off, the user is encouraged to try the following measures to correct the interference:

- Reorient the receiving antenna.
- Relocate the unit with respect to the receiver.
- Plug the unit into a different outlet so that the unit and receiver are on different branch circuits.
- Ensure that mounting screws and connector attachment screws are tightly secured.
- Ensure that good quality, shielded, and grounded cables are used for all data communications.

If necessary, the user should consult the dealer or an experienced radio/television technician for additional suggestions. The following booklets prepared by the Federal Communications Commission (FCC) may also prove helpful:

- How to Identify and Resolve Radio-TV Interference Problems (Stock No. 004-000-000345-4)
- Interface Handbook (Stock No. 004-000-004505-7)

These booklets may be purchased from the Superintendent of Documents, U.S. Government Printing Office, Washington, DC 20402.

TABLE OF CONTENTS

1 INTRODUCTION

1.1 About this Manual

This manual is intended to provide the user with an overview of the board and benefits, complete features specifications, and set up procedures. It contains important safety information as well.

1.2 Product Overview

The **TS-7400** is a small (2.9" x 4.7") embedded computer module (System on Module) that is designed to provide extreme performance for applications which demand high reliability, fast bootup/startup and connectivity at low cost and low power, such as point-of-sales (PoS), vending machines, data acquisition units, data recorder modules, etc.

The **TS-7400** System on Module runs on a 200 MHz ARM9 processor with power under 2 Watts. Low board complexity, low component count, and low power/heat makes for an extremely reliable embedded engine. The **TS-7400** is available in thousands of configurations, many of which are Commercial off the Shelf (COTS) and available to ship today.

The EP9302 processor from Cirrus is the highly integrated 200Mhz ARM9 processor that the **TS-7400** is built around and includes an on-chip 10/100 ethernet, USB, serial, and Flash/SDRAM controller. A supplemental PLD provides glue logic, and watchdog timer. Integer CPU performance is about 20% faster than our 133 Mhz x86 offerings.

Even with the standard power consumption of 2 Watts, the **TS-7400** runs without fans or heat sinks in the temperature range of -40° to +70°C. Extended Temperature -40° to +85°C is also standard, but CPU clock must be decreased to 166MHz for higher temperatures.

Unlike other TS-7000 ARM products, the **TS-7400** is not a stand-alone Single Board Computer. The TS-9441 peripheral board is needed during development in order to configure and set up the TS-7400 flash system. The TS-9441 has 2KB EEPROM, and 2MB serial Flash, so it can be used to bring-up/recover "dead" TS-7400 boards. Using a console on the TS-7400 is only possible through the TS-9441 RS232 serial port, named Console Header. Also, the TS-9441 enables regulated 5VDC power in. Once the TS-7400 is configured and loaded, it can safely run stand-alone.

1.3 Benefits

Out-of-the-Box Productivity

Technologic Systems Linux products get you to your application quickly. Our Single Board Computers boot directly to Linux as shipped. There is no complicated CMOS setup or configuring of a Linux derivative Operating System to source, define, and load. Technologic Systems has pre-configured each SBC in flash memory.

The **TS-7400**'s user can power up the board and immediately (using the TS-9441) begin application development. Of course, should you wish to configure your own version of Linux or use a different operating system, this is easy too. Technologic Systems provides the solution to fast application development without tedious OS configuration.

Impressive Performance

The ARM920T's 32-bit architecture, with a five-stage pipeline, delivers very impressive performance at very low power. The EP9302 CPU has a 16 KB instruction cache and a 16

KB data cache to provide zero-cycle latency to the current program and data, or they can be locked to guarantee no-latency access to critical sections of instructions and data. For applications with instruction-memory size restrictions, the ARM920T's compressed Thumb instruction set can be used to provide higher code density and lower Flash storage requirements.

As a benchmark, the **TS-7400**'s CPU integer performance, at a supplied 200 MHz, is about twice as fast as the Technologic Systems 133MHz 586-based products.

1.4 Features

The **TS-7400** comes **standard** with these features:

- ✔ **TS-Linux** Embedded Operating System Installed
- ✔ 200 MHz ARM9 CPU with MMU
- ✔ 32 MB integrated high-speed flash with hardware ECC (Boots to TS-Linux)
- ✔ 32 MB RAM (64 MB or 128 MB optional)
- ✔ 2 USB 2.0 Compatible OHCI ports (12 Mbit/s Max)
- ✔ 3 TTL serial UARTs
- ✔ SD Flash card connector
- ✔ 10/100 Megabit Ethernet port
- ✔ 20 general purpose I/O pins with Schmitt trigger
- ✔ Power requirements are 5V DC @ 350mA
- ✔ Flexible 40 pin expansion connector
- ✔ Small size 2.9 x 4.7 inches (7.4 x 11.9 cm)
- ✔ Optional battery-backed real time clock
- ✔ Operating Temperature Range: Fanless from -40° to +70°C
- ✔ Extended Temperature -40° to +85°C standard at lower CPU clock speeds
- ✔ RoHS Compliant

1.5 Configurability

The **TS-7400** can be configured for your application using the following available on-board options and external accessories:

On-board Options

- ✔ **TS-7400-yyy-32F**: TS-7400 with up to 128 MB of on-board SDRAM. For example, **TS-7400-64-32F** selects model **TS-7400** with 64 MB of SDRAM.
- ✔ **OP-BBRTC**: on-board sealed battery backed RTC
- ✔ **OP-PS30V** : 8-30VDC switching power input through screw terminal
- ✔ **OP-TMPSENSE**: High-precision temperature sensor

Note
The **TS-7400** SBC is RoHS compliant by default (Restriction of Hazardous Substances) Directive. Contact Technologic Systems for RoHS support.

External Accessories

- ✔ **SD-512**: 512 MB SD Flash Card with full ARM tool chain installed and Debian (included in KIT-7400)

- **USB-FLASH-256**: 256 MB flash drive with full ARM tool chain installed and Debian
- **WIFI-G-USB:** Linux-supported USB 802.11g WiFi transceiver for wireless networking
- **TS-9441** : Boot/Console Peripheral Board for TS-7400 (required during development) (included in the KIT-7400)
- **TS-DC420-ENC**: Peripheral board provides 8-30VDC power input, Xbee radio socket, DIO lines, 3 A/D channels, 3 COM ports, RS-485/RS-422 with full-duplex and aluminum enclosure
- **PS-18VDC-REG**: Regulated 18VDC wall mounted power supply (compatible with OP-PS30V)
- **PS-5VDC-1AMP**: 5VDC 1AMP Power Supply (100-120V) (included in the KIT-7400)
- **PS-5VDC-2_5REG**: 5VDC 2.5AMP Regulated Power Supply

Note
Check our website at **www.embeddedARM.com** for an updated list of options and external accessories

1.6 TS-WIFIBOX Application Kit

The **TS-7400** powers the TS-WIFIBOX application kit, a small wifi-enabled computer box. For more information see **http://www.embeddedarm.com/products/board-detail.php?product=TS-WIFIBOX**

1.7 TS-ARM Development Kit

The KIT-7400 TS-ARM Development Kit for the **TS-7400** Single Board Computer includes all equipment necessary to boot into the operating system of choice and start working. The development kit is highly recommended for a quick start on application development.

The KIT-7400 Development Kit contains a 256 or 512 MB Flash drive (SD Card) which includes:

- ✔ A self-hosting ARM installation of the **Debian Linux** 3.0 distribution compiled for ARM
- ✔ gcc 2.95.4 and gcc 3.0 compiler with full tool-chain
- ✔ Hardware test routines source code and other example source code
- ✔ Debian package system: apt-get, tasksel, dselect

The development kit additionally includes:

- ✔ SD Card reader
- ✔ 5 VDC regulated power supply (international versions available)
- ✔ NULL modem cable
- ✔ Adapter cable from 10-pin header to DB9
- ✔ Various cables for connection DIO, LCD, Keypad, etc.
- ✔ Development CD with complete TS-Kernel source, manuals, example code, etc.
- ✔ Printed supporting documentation for TS-7400 Hardware, Linux for ARM and Development Kit.

Note
Single Board Computer is not included on the Development Kit (sold separately).

1.8 Software and Support

Software features include:

- ✔ Boots Linux out-of-the-box in 1.10 seconds (to shell prompt).
- ✔ Flexible booting options (SD card, NAND flash or offboard SPI flash)
- ✔ SD card pre-installed with standard Debian Linux distribution.
- ✔ Firmware has ability to verify boot medium CRC before allowing bootup.
- ✔ Ability to boot password-locked SD cards.
- ✔ Startup Linux miniroot scripts allows flexible root and backup filesystem selection (SD, flash, NFS, USB flash) as well as software field upgrade support.
- ✔ Linux "bootload" program allows booting of Linux kernels and other OS' from within Linux itself.

Technologic Systems provides:

- ✔ Free system software and documentation updates available on our web site
- ✔ Long-term availability and short lead-times (20 year history)
- ✔ Free technical support from engineers by phone, fax, or email
- ✔ Production tests and burn-in prior to shipment on every board.
- ✔ Board customizations available with no minimum order.
- ✔ Factory loading of customer supplied software available.
- ✔ Readily available hardware and software professional services for hire.
- ✔ 30-day, money back guarantee on evaluation units
- ✔ One-year, full warranty

Linux OS Support

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 NAND and NOR Flash chips (via mtd drivers), a compact flash IDE driver, A/D converters, SD Card through the TS-SDCORE, additional ethernet ports and more. If you want to use Linux and aren't tied to the x86 architecture, the TS-7400 can be very cost-effective.

Debian can also be used with an NFS root file system or USB flash drives. The TS-Kernel used is based upon the version 2.4.26, patched and compiled for the Cirrus EP9302 ARM920T processor, and is real-time capable through RTAI.

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

- ✔ EXT2 file system image in the SD card
- ✔ JFFS/YAFFS file system image in the on-board Flash
- ✔ NFS root, via Ethernet port (after fast bootup, mount a NFS root and chroot to it)

Note
The TS-Kernel supports the **Real-Time** Application Interface (**RTAI** project), making the embedded operating system capable of handling applications with hard real-time restrictions.

Other OS Support

The **TS-7400** can be loaded with other operating systems such as Windows CE, NetBSD, etc. Technologic Systems will provide support for these, and possibly other operating systems, in the future. Currently, only **Linux** and **NetBSD** are supported on the **TS-7400**.

2 GETTING STARTED

2.1 Installation Procedure

Before performing any set up or placement procedures, take the precautions outlined in this section.

Handling the Board Safely

Be sure to take appropriate Electrostatic Discharge (ESD) precautions. Disconnect the power source before moving, cabling, or performing any set up procedures.

> **Warning**
> Inappropriate handling may cause damage to the board.

Setup and Installation Instructions

Follow these guidelines for safety and maximum product performance:

- ✔ Observe local health and safety requirements and guidelines for manual material handling

Setup Tools

Depending on placement and cabling, you may need the following tools:

- ✔ Small flat-blade screwdriver
- ✔ Small Phillips screwdriver

Setup Procedure

After locating, setting up, grounding, and cabling the **TS-7400**:

- ✔ Apply power
- ✔ Monitor the **TS-7400** using a terminal emulator to verify that the board is operating properly

Disconnecting AC Power

- ✔ Unplug from the power source.
- ✔ Disconnect other cables as required.

2.2 Console and Power Up

An ANSI terminal or a PC running a terminal emulator and a TS-9441 peripheral are required to communicate with your **TS-7400** computer. Simply connect an ANSI terminal (or emulator) to the console header on the TS-9441 using a null modem cable and the DB9 adapter for the header (these are 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). 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. The TS-9441 also requires the 10-pin header to 9-pin Sub-D adapter which is P/N: RC-DB9.

> **Note**
> TS-7400 only has TTL UARTs and to connect to the serial console with a PC's serial port you need to use the TS-9441 or external RS232 level converters.

Connect a regulated 5VDC, (1A minimum) power source on the power input connector. Please note the polarity printed on the board. The boot messages, by default, are all displayed on COM1 at 115200 baud. The board will also answer telnet connections to IP address 192.168.0.50.

The TS-7400 board has Linux installed by default on onboard flash. Upon bootup, The board will boot within 1.1 seconds to a Linux prompt on UART #0 (/dev/ttyAM0).

The default fastboot shell has available several standard Linux commands through the "busybox" program. Technologic Systems has made several modifications to the busybox source code to keep bootup as fast and simple as possible. The modified source code is available to Technologic Systems customers.

Upon bootup, you should see out of your serial port:

```
>> TS-FLASHBOOT - built Sep 27 2006
>> Copyright (c) 2006, Technologic Systems
.
.
.
Finished booting in 1.10 seconds
Type 'tshelp' for help
$
```

At this point, if you type 'exit' from the serial shell, the TS-7400 will then attempt a full Debian Linux bootup from the SD card on partition #3. If the SD card is not present or the EXT2 filesystem does not pass a basic sanity test or the special file "/notrootfs" exists, a demonstration version of a miniature Linux distribution contained on the onboard flash itself is instead booted. This version of Linux, named TS-Linux, contains Apache, SSH, PPP, and FTP server and many other common utilities and libraries and is identical to the distribution installed by default on the TS-7200 line of single board computers. Other community-supported embedded Linux distributions are available. For instance, the "Buildroot" project at http://buildroot.uclibc.org/ allows one to easily build custom filesystems and cross-toolchains.

Should you wish to automatically bypass the fastboot and proceed directly into starting the SD card version of Linux, you can do so with the following command issued to the fastboot shell:

```
ln -sf /linuxrc-sdroot /linuxrc; save
```

To automatically boot from onboard flash the command is:

```
ln -sf /linuxrc-mtdroot /linuxrc; save
```

To automatically boot from USB flash dongle or USB hard drive:

```
ln -sf /linuxrc-usbroot /linuxrc; save
```

To get back to the fastboot shell, you can do so by placing the file "/fastboot" in the root directory of the filesystem.

The '/linuxrc' file is a shell script that is the very first thing run by the kernel on startup. Several sample startup scripts are included and can either be used directly ("ln -sf /linuxrc-XXX /linuxrc" command) or modified to include custom bootup logic. These shell scripts were designed to be as fast and simple as possible (approximately 45 lines of code) for easy customer modifications. It is anticipated that this shell script be modified from the default to implement things in the customer's product such as backup configurations, software field updates, conditional booting/verification of SD cards, etc. Technologic Systems professional services are available should you need help in implementing a specific feature.

TS-7400 Recovery

Although it is easy to get your board into an unbootable state during development if you botch a modification, it is equally easy to use the TS-9441 to recover the default startup. To do so, place the "Boot" jumper on the TS-9441 and reset the board. In approximately 3 seconds, the board will have fastbooted to the TS-9441 flash and presented a shell prompt. To recover the default initrd and linuxrc, enter the command "save", remove the Boot jumper, and reboot back to a restored to factory default TS-7400.

2.3 Bootup Process

The processor first runs code from the dedicated EEPROM chip containing the TS-FLASHBOOT bootup program. TS-FLASHBOOT sets up bus timing, SDRAM, and other low-level hardware initialization. It also loads sector 0 from the NAND flash chip at 0x1000 physical address. Once its loaded, it jumps (in ARM Thumb mode) to the first instruction, with register r0 containing a function pointer to a routine to read sectors from the boot media (NAND flash) and in r1 a routine to a function to print a character string to the serial port. The routines have the following C signature:

```
void read(unsigned int sector, unsigned char *buf, unsigned int
nsectors)
void puts(unsigned char *buf)
```

By default, whats contained in sector 0 of the NAND flash is a DOS-style partition table at offset 446 and a 446 byte Linux-specific kernel bootloader at offset 0. This mini-bootloader contains code to parse the partition table looking for 2 partitions marked with partition ID '0xda'. It uses the start sector and size parameters of the partition table to load the first partition (Linux kernel) at offset 0x2180000 and the second partition (if it exists) at 0x1000000 (Linux initrd). If a bad sector or a sector failing an ECC check is detected, it skips to the next 16kbyte boundary and continues-- in this way, NAND chips with bad sectors remain bootable. Next, it composes up the necessary pre-boot ATAG data structures required by Linux kernels and jumps to address 0x2180000 at which point the Linux kernel takes over.

One of the first things the Linux kernel does is disable the hardware watchdog. The watchdog is armed for 8 second expiry before automatic reboot and will be constantly fed as long as the firmware or MBR Linux bootloader is reading from flash. The board will appear to constantly reboot every 8 seconds should something go wrong. The TS-7400 uses an external watchdog implemented in the CPLD instead of the CPU internal watchdog to guarantee proper reset operation in the case of a severely malfunctioning CPU. The CPU internal watchdog is also available, but Technologic Systems does not recommend using it.

2.4 Loading or Transferring Files

Three methods are available for transferring files between a desktop PC and your **TS-7400**: Ethernet downloads, flash memory devices, and Zmodem downloads. Full descriptions of each are detailed below. Other programs that use serial ports to transfer should work as well.

Transferring Files via the Ethernet Port

The default JFFS Linux root file system includes a small FTP server that can be used for uploading/downloading of files across an Ethernet network. Simply point your preferred FTP client to your **TS-7400** IP address (default is 192.168.0.50). You can login as root or any valid user previously created from the useradd utility. By default, the JFFS image will not accept anonymous FTP.

Transferring Files via Flash Memory Device

An SD card or an USB flash memory card can be used to easily move files from a host system. We suggest using a low-cost SanDisk USB Compact Flash card or SD card interface for your host system. USB memory devices need no extra accessory to connect to the host PC. The flash memory devices can then be hot swapped (inserted or removed without rebooting the host PC).

Zmodem Downloads

Using the Zmodem protocol to send files to and from the **TS-7400** SBC is simple and straightforward. The only requirement is a terminal emulation program that supports Zmodem, and virtually all do. If you are using Windows 95 or later for your development work, the HyperTerminal accessory works well.

To download a file to the **TS-7400** from your host PC, execute lrz at the Linux command line on the **TS-7400** (while using console-redirection from within your terminal emulator) and begin the transfer with your terminal emulator. In HyperTerminal, this is 'Send File...' from the 'Transfer' menu.

To upload a file from the **TS-7400** to your host PC, execute lsz <FILENAME> at the Linux command line on the **TS-7400** and start the transfer in your terminal emulator. Many emulators, HyperTerminal among them, will automatically begin the transfer themselves.

Occasionally there may be errors in transmission due to background operations. This is not a problem -- Zmodem uses very accurate CRC checks to detect errors and simply resends bad data. Once the file transfer is complete the file is completely error free. For best results when using HyperTerminal, the hardware handshaking must be enabled in HyperTerminal.

2.5 TS-7400 production lifetime

The TS-7400, as shipped by default from Technologic Systems will never be altered significantly from its originally introduced state. The board has an intended infinite production lifetime, meaning Technologic Systems will build and sell TS-7400's in as low as single piece quantities as long as its constituent parts are readily available. This typically means a production lifetime of 10-15 years. A stable hardware platform and default install keeps unexpected surprises from cropping up down the road when products must be retested or redesigned because of manufacturer introduced hardware/software design changes or obsolescence.

The 802.11g card for the TS-7400 is not manufactured by Technologic Systems. 802.11 wireless ethernet is tightly coupled to the consumer electronics and desktop computer industries and as such products tend to be more transient (though also less expensive). Often times manufacturers change chipsets or discontinue products with little or no notice. Although usually there are many alternatives and Technologic Systems can seamlessly transition to a new supplier, TS-7400 integrators should be aware of the possibility of different 802.11 type devices being used on the default product in the future.

3 SOFTWARE

3.1 Software-update facility in factory configuration

Although Technologic Systems can load the flash with customer supplied software from the factory, it is often more convenient to have a local production process for installing or updating the files and programs that make your product unique. The default software installation enables this by providing a hook to allow customer code to "hijack" the normal fast-boot bootup process. The default linuxrc (linuxrc-fastboot) will run a program "/bin/check-usb-update" in the background after bootup has completed. This program looks for a script "/tsinit" on the USB mass storage device (USB thumb-drive, or USB hard drive) connected to the bottom USB slot. If this script exists, it is then run automatically as the Linux "root" user-ID. A sample /tsinit script that copies a program "myprogram" to onboard flash, and then changes the default bootup to the SD card follows:

```
#!/bin/sh

ln -sf /linuxrc-sdroot /linuxrc
mount /dev/mtdblock/3 /onboardflash
cp /mnt/root/myprogram /onboardflash/bin
umount /onboardflash
save
reboot
```

While the "tsinit" script is run, the red LED will be on. After it is complete, the red LED will go off. A customer could mass-update many hundred TS-7400's easily by using a USB flash dongle with a custom "tsinit" and data files and applying power with the USB dongle on, wait for the LED to go off, then proceed to the next board. The "tsinit" script will begin to execute approximately 3 seconds after power-on.

3.2 Updating/Recovering the bootloader kernel

Although usually not necessary, customers wanting to build and install their own bootloader kernels can do so from the fastboot environment by using NFS and an internal busybox utility "mtdcp". To mount an NFS filesystem from the fastboot shell, type:

```
mount 192.168.0.1:/tsarm-nfsroot /mnt/root
```

You may have to change the server IP and server mount point to whats appropriate on your network. The default TS-7400 IP address is 192.168.0.50, but this could be changed with the command:

```
ifconfig eth0 <NEW_IP_ADDRESS>
```

Then, assuming /mnt/root/zImage is your new kernel. Issue the command:

```
insmod /ts7xxx_nand.o; mtdcp /mnt/root/zImage 1 0
```

This copies the zImage kernel binary to /dev/mtdblock/1, with appropriate handling of any potentially bad flash sectors. /dev/mtdblock/1 is the first MBR partition of the TS-7400 NAND flash chip. Do not use the 'dd' command to copy as it does not do the right thing when presented with bad NAND flash sectors.

Note that changing the default bootloader kernel should be limited to only when absolutely necessary. Instead, you should keep the known-good, factory default Linux kernel as the bootloader kernel and then use the facilities of the "bootload" Linux command to reboot into your custom kernel as described in the following section.

3.3 Booting custom kernels and OS images from within Linux

Technologic Systems has developed a Linux application "bootload" that allows arbitrary booting of Linux and other OS kernels within Linux itself. The power-on bootloader contained in the MBR of the flash chip is not extremely flexible as it was instead designed to be very fast (1.1 second Linux bootup) and small (fits in the 443 bytes of empty space in the MBR). The "bootload" program allows one to use the full facilities of Linux to retrieve kernel files. Doing so also allows the use of standard shell scripts for the programmatic selection of appropriate kernels, Linux initrd's, and kernel command line arguments for maximum flexibility.

By default, the "linuxrc-mtdroot" sample startup script will look for a file in the YAFFS2 NAND flash filesystem named "/vmlinux.bin" and attempt to load and boot it. You can modify the kernel command line parameters or image file by changing the respective line in the startup script. By using a file in the YAFFS2 filesystem in this way for the kernel, it is possible to atomically and safely field update the kernel from, e.g., ftp with a command such as:

```
wget ftp://mycompany.com/newvmlnx.bin && mv newvmlnx.bin
/vmlinux.bin
```

Command usage information follows:

```
$ bootload --help
Usage: bootload [OPTION] FILE
Linux to Linux bootloader - (re)boots a TS-7xxx board to another
kernel, OS image, or raw executable by replacing the running Linux
kernel.

General options:
    -c, --cmdline=CMD Use CMD as the Linux kernel boot args
    -r, --initrd=FILE Use FILE for Linux initial ramdisk
    -s, --initrdsz=SZ Only read SZ bytes from the initrd file
    -b, --base=ADDR   Load the image at ADDR instead of 0x218000
    --version         Print version and copyright information
    -h, --help        This help
```

When FILE is -, reads from standard input.

Some "one-line" examples of usage:

```
# Boot a compressed kernel image:
bunzip -c vmlinux.bin.bz2 | bootload -

# Reboot a kernel, but pass the 1MB running ramdisk to the new
kernel:
mount -o remount,ro /dev/rd/0 /
bootload -c "console=ttyAM0,115200 root=/dev/ram0" -r /dev/rd/0
\  -r 0x100000

# Boot one of 2 kernels based on the state of DIO line #7:
if dio_data_get 7; then bootload vmlinux.backup.bin; else \
bootload vmlinux.bin; fi
```

The bootload application requires a kernel module "bootloader.o" to be installed using "insmod" prior to invocation. This module, as well the the "bootload" application itself are installed on the default onboard YAFFS2 NAND flash Linux filesystem (/dev/mtdblock/3).

3.4 Accessing internal TS-7400 registers from Linux userspace

Linux applications run in a protected and separate environment where they can do no damage to either the kernel or other applications running simultaneously. This protected environment does not allow arbitrary manipulation of hardware registers by default. Applications may be allowed temporary access through memory space windows granted by the mmap() system call applied to the /dev/mem device node. For instance, to set up access to the GPIO registers at 0x12c00000, the following snippet of C code is provided as an example:

```
#include <sys/mman.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

{
int fd = open("/dev/mem", O_RDWR|O_SYNC);
char *gpioregs;

gpioregs = (char *)mmap(0, 4096, PROT_READ|PROT_WRITE,
  MAP_SHARED, fd, 0x12c00000);

gpioregs[0] = 0xff; //data direction register set all outputs */
gpioregs[2] = 0x12; //output high to DIO_01 & DIO_4, all else low
}
```

Some notes about the preceding code:

- ✔ Make sure to open using O_SYNC, otherwise you may get a cachable MMU mapping which unless you know what you're doing, probably is not what you want when dealing with hardware registers.
- ✔ mmap() must be called only on pagesize (4096 byte) boundaries and size must at least have pagesize granularity.
- ✔ mmap() of /dev/mem is only allowed for processes with UID 0 (root)
- ✔ More information on mmap() and open() system calls can be had by running "man mmap" or "man open" from most any Linux shell prompt.
- ✔ When working with char * types to registers, make sure to compile with the "-mcpu=arm9" option otherwise the wrong ARM opcodes will be used and your byte reads/writes may be turned into 32-bit reads and writes.

3.5 Changing/Updating the TS-BOOTROM

Normally, the TS-7400 boot firmware is loaded with the TS-FLASHBOOT bootup program. This program bootstraps the CPU by loading the first 512 bytes from the NAND flash and jumping into it. This program then loads the kernel and initrd from the NAND flash. Once the kernel has booted and mounted the initrd, it can "pivot_root" and give the illusion it actually booted directly from SD, NAND, NFS, etc file systems. Should you wish to actually load the kernel and fastboot initrd from an SD card, the TS-FLASHBOOT bootup program must be replaced with TS-SDBOOT. This can be done with the "tsbootrom-update" program.

Usage and command line help for this command follows:

```
$ tsbootrom-update --help
Usage: tsbootrom-update [OPTION] ...
Updates TS-BOOTROM bootup program stored on EEPROM

General options:
  -n             Do not actually write EEPROM
  -s, --sdboot       Write TS-SDBOOT bootup program
  -f, --flashboot    Write TS-FLASHBOOT bootup program
  -u, --burninboot   Write TS-BURNINBOOT bootup program
  -p, --spiflashboot Write TS-SPIFLASHBOOT bootup program
  -b, --blastboard   Write to blast board EEPROM instead of SBC
  -h, --help         This help

EEPROM security block options:
  -m, --mac=X        Write X as ethernet MAC address
  -l, --verifylen=N  Checksum includes first N 512 byte sectors
  -d, --device=FILE  Use FILE to re-compute checksum value
  -V, --verifydat=N  Use N as pre-computed checksum value
  -L, --lockdat=X    Use X for the SD unlock data token from
                 previous "sdlock --set" command
  -k, --verifylock   Do not boot to an unlocked SD card
  -c, --noconsole    Disable serial console bootup messages

  TS-production specific options:
  -a, --alloc-mac    Get MAC address from /var/ts-production/mac
```

TS-SDBOOT contains several features for high security. One feature is the ability to store a checksum of the SD card on the board to verify before bootup. If the checksum fails, the bootup firmware will refuse to boot the inserted SD card. Another feature is the ability to boot a password protected SD card. With this, it is possible to make an SD unreadable to any device except the TS-7400 to which it is assigned. Although not directly a function of TS-SDBOOT, an SD card can also be made permanently write-protected through a software command. The combination of these features allows product designers several options on the security of their software and of their deployed TS-7400 based devices.

The TS bootup programs will by default print a banner message on bootup to the serial port displaying its build date, etc. If you wish to use all serial ports for your application and do not wish to have a serial console, the tsbootrom-update program can be used to silence early bootstrap banner messages so as to not confuse any potential external device UART #0 may be connected to.

Other bootstrap programs are available for-pay from Technologic Systems should you need them and custom ones may be designed for volume customers. For instance, the TS-ETHBOOT bootup program may be used to completely boot Linux from the network without NAND flash or SD card installed.

3.6 SD flash card security features

Technologic Systems provides a "sdlock" Linux command which can be used to manipulate SD card hardware-enforced password locks and set the card's permanent write-protect feature. Using a password protected SD card is a great way to ensure software security and/or to make sure your TS-7400 based product cannot be used in an unintended matter once deployed.

```
$ sdlock
Usage: sdlock [OPTION] ...
Controls SD card lock and permanent write-protect features.

General options:
  -p, --password=PASS Use PASS as password
  -c, --clear         Remove password lock
  -s, --set           Set password lock
  -u, --unlock        Unlock temporarily
  -e, --erase         Erase entire device (clears password)
  -w, --wprot         Enable permanent write protect
  -h, --help          This help
```

When the TS-7400 is configured with the TS-SDBOOT bootup firmware, the SD unlock password can be stored in onboard EEPROM for automatic unlocking and booting of password protected SD cards. By default, TS-SDBOOT will still boot unlocked cards, but this behavior can be changed with the "--verifylock" option to the "tsbootrom-update" command described above-- with the "--verifylock" option the TS-7400 will only boot locked SD cards.

TS-SDBOOT can also verify an arbitrary number of sectors of the SD flash card before allowing bootup. If the stored CRC does not match the actual CRC, the board will refuse to boot and blink the red LED continuously.

The various SD commands that manipulate the password lock are marked as "optional" in the SD card specification. This means that not all SD card vendors may implement them in their devices. If they are not implemented, you will not be able to set the SD lock with the "sdlock" command.

For further information, contact a Technologic Systems' engineer.

3.7 TS-7400 specific Linux devices

Although working with the TS-7400 Linux is identical in most ways to working with a PC version Linux, one does need to be aware of some driver differences.

- ✔ The serial port device nodes are /dev/ttyAM0, /dev/ttyAM1, and /dev/ttyTS0, respectively. The default PC uses /dev/ttyS* as device names. The software API to these devices is the same as on the PC.
- ✔ The onboard flash is broken up into partitions and accessed through the Linux driver framework known as "MTD", or (M)emory (T)echnology (D)evice. The partitioning is dynamic and depends on the DOS-style MBR found at sector 0 of the flash. This MBR can be changed by using the "fdisk" command on the /dev/mtdblock/0 device, but doing so is not recommended.

  /dev/mtdblock/0 - Whole disk block device driver.

  /dev/mtdblock/1 - First MBR partition (bootloader kernel binary)

  /dev/mtdblock/2 - Second MBR partition (bootloader initrd)

  /dev/mtdblock/3 - Third MBR partition (Linux YAFFS2 filesystem)

/dev/mtdblock/4 - 4th MBR partition (unused in default load)

Note that the MBR installed by default on the TS-7400 contains a 443 byte bootloader program that loads the initial power-on kernel and initrd from the first and second partitions. Replacing it with a MBR found on a PC would not work as a PC MBR contains an x86 code bootup program. Doing so would cause the device to constantly reset itself every 8 seconds as the hardware watchdog expires.

- Linux uses a NAND flash filesystem called YAFFS2 for general purpose file storage. This filesystem is a log-structured filesystem which is safe against corruption caused by system crashes and power loss without the need for consistency checking on next boot. A normal PC cannot use this filesystem as it is specifically designed for NAND flash which a PC does not have.
- The TS version of Linux uses a special device driver at /dev/misc/bootloader to accommodate the hooks needed by the "bootload" program to allow Linux to act as a bootloader and boot other Linux kernels and operating systems.

3.8 Debian Linux OS

The typical way of doing Linux development on the TS-7400 is actually on the board itself. Since the TS-7400 CPU is a PC-class processor in everything but power consumption and performance, it has no problem running real PC-class operating systems such as Linux. By running the full version of Linux (and not scaled-down microcontroller project OS's such as ucLinux), the TS-7400 can run the entire suite of applications contained in the Debian Linux distribution including the compilers. Since almost every open source program available for Linux is contained within the Debian Linux binary distribution, one rarely has to compile the large code-bases that would otherwise have forced integrators to a complicated cross-compilation environment due to the limited RAM/Mhz of the embedded computer. All too often, open-source projects do not anticipate the possibility of cross-compilation in their build systems, leaving issues for the system integrator to resolve.

The default SD card contains compilers and everything needed for developing applications in C, C++, PERL, PHP, and SH. Java, BASIC, TCL, Python and others are available for Debian, but not installed by default.

One can still use cross-compilers hosted on just about any platform if there is a specific need. Technologic systems includes binary versions of the popular Linux "crosstool" project at http://www.kegel.com/crosstool/ to allow cross-compiling on Windows/cygwin or a Linux/i386 PC on our website website.

apt-get

When using the Debian Linux file system, adding new packages and removing undesired ones is done all through Debian's package management. “apt“, “dpkg“, all behave as expected. With Debian, one can easily install and remove software packages. For a quick demonstration of how easy it is to remove and install programs with Debian, try the following commands:

```
apt-get install hexedit
hexedit /etc/passwd
^C (hit CTRL+C to safely exit)
apt-get remove hexedit
```

apt-get install installs a package name, while apt-get remove removes the named package. Visit the Debian home-page for further information, since a full in-depth discussion on Debian is outside the scope of this document.

- http://www.debian.org

3.9 Getting Started with Linux

Logging In and Basic Commands

After the desired Linux Kernel is loaded and executed, 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-7400 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

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-7400 SBCs.

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.

Setting Up the networking with Debian

To configure the network interfaces when booting into Debian Linux, edit the file "/etc/network/interfaces". A typical interfaces file would contain the following:

```
auto lo eth0
# The loopback interface
iface lo inet loopback
# The first network card
auto eth0
#iface eth0 inet dhcp
iface eth0 inet static
address 192.168.0.50
netmask 255.255.255.0
gateway 192.168.0.1
```

Those lines starting with a # symbol are comments. The line “auto lo eth0” means both the loopback interface and the first ethernet interface will be started automatically by the Debian networking scripts. The above example shows that eth0 would be assigned the static address of 192.168.0.50, using 192.168.0.1 as the default gateway. If one was to comment out those lines, and then uncomment the line iface eth0 inet dhcp, then eth0 would use a dhcp client to obtain it's IP and other relevant network information.

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
```

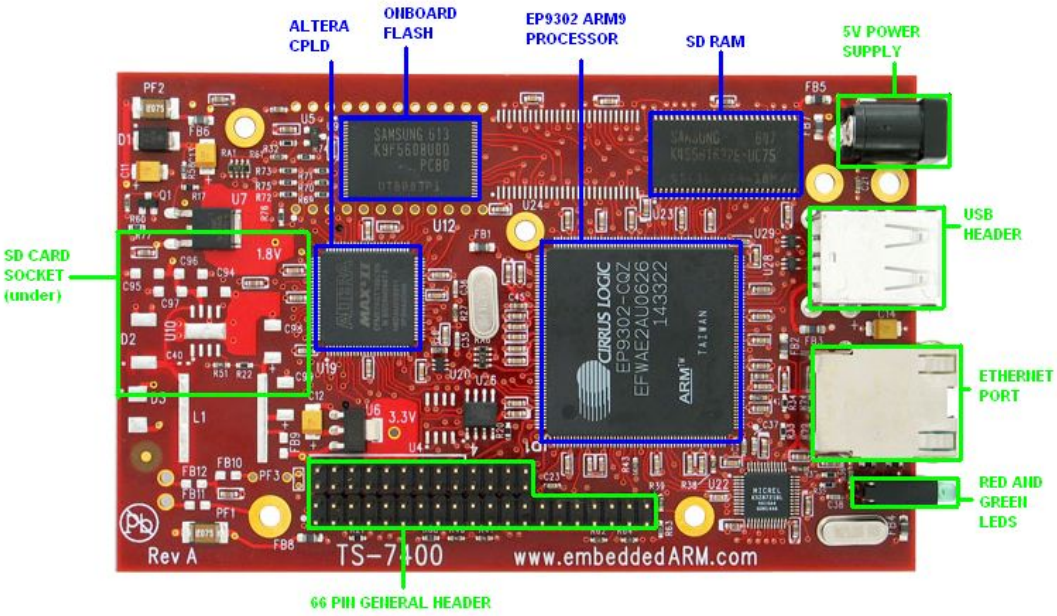
Also, the “/etc/inet.conf” file is used to configure the initialization and parameters of other services.

Note
For further information regarding the software solutions available for the **TS-7400** and instructions about Debian Linux, please refer to the **Linux for TS-ARM User's Guide**. This can be found for download at the Technologic Systems website.

4 HARDWARE COMPONENTS

The following picture shows where the main headers, connectors and most important hardware components are located on the **TS-7400**. Understanding this picture will help you to follow the header-connector oriented organization of this manual. The blue marked objects on the picture are the on-board chips and components, while the green ones are the various on-board headers and connectors for peripherals.

Picture: TS-7400 Hardware Components



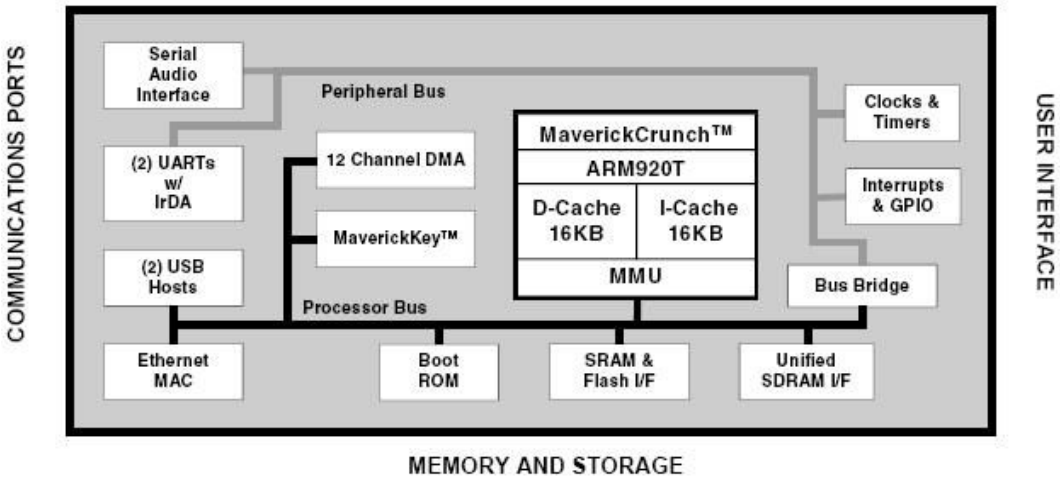
4.1 Processor

Cirrus EP9302

The EP9302 features an advanced 200 MHz ARM920T processor design with a memory management unit (MMU) that allows support for high-level operating systems such as Linux, Windows CE, and other embedded operating systems. The ARM core operates from a 1.8 V supply, while the I/O operates at 3.3 V with power usage between 100 mW and 750 mW (dependent on speed). As a general-purpose processor, it provides a standard set of peripherals on board and a full set of Technologic Systems add-on peripherals via the standard PC/104 Bus.

The ARM920T's 32-bit architecture, with a five-stage pipeline, consisting of fetch, decode, execute, memory, and write stages, delivers very impressive performance at very low power. The EP9302 CPU has a 16 KB instruction cache and a 16 KB data cache to provide zero-cycle latency to the current program and data, or they can be locked to guarantee no-latency access to critical sections of instructions and data. For applications with instruction-memory size restrictions, the ARM920T's compressed Thumb instruction set can be used to provide higher code density and lower Flash storage requirements.

Picture: Cirrus EP9302 Block Diagram



EP9302 key features include:

- ✔ ARM (32-bit) and Thumb (16-bit compressed) instruction sets
- ✔ 32-bit Advanced Micro-Controller Bus Architecture (AMBA)
- ✔ 16 kbyte Instruction Cache with lockdown
- ✔ 16 kbyte Data Cache (programmable write-through or write-back) with lockdown
- ✔ MMU for Linux®, Microsoft® Windows® CE and other operating systems
- ✔ Translation Look Aside Buffers with 64 Data and 64 Instruction Entries
- ✔ Programmable Page Sizes of 1 Mbyte, 64 kbyte, 4 kbyte, and 1 kbyte
- ✔ Independent lockdown of TLB Entries

For further information about the EP9302 features, refer to the **EP9301 User's Guide**.

Note
The EP9302 is identical silicon to the EP9301 except it is rated to run at 200 Mhz, instead of 166 Mhz. The available **EP9301 User's Guide** can still be used as the main reference manual.

MMU

The EP9031 features a Memory Management Unit, enabling high level operating systems such as Embedded Linux and Windows CE to run on the **TS-7400**. In the same way, the Linux TS-Kernel takes advantage of the MMU functionality.

The MMU is controlled by page tables stored in system memory and is responsible for virtual address to physical address translation, memory protection through access permissions and domains, MMU cache and write buffer access. In doing so, software applications can access larger "virtual" memory space than the available physical memory size, allowing multiple programs to run and use the system memory simultaneously.

For further information about the MMU functionalities, refer to the EP9301 User's Guide.

Interrupts

The EP9302 interrupt controller allows up to 54 interrupts to generate an Interrupt Request (IRQ) or Fast Interrupt Request (FIQ) signal to the processor core. Thirty-two hardware priority assignments are provided for assisting IRQ vectoring, and two levels are provided for FIQ vectoring. This allows time critical interrupts to be processed in the shortest time possible.

Internal interrupts may be programmed as active high or active low level sensitive inputs. GPIO pins programmed as interrupts may be programmed as active high level sensitive, active low level sensitive, rising edge triggered, falling edge triggered, or combined rising/falling edge triggered.

The EP9302 interrupt controller also includes the following features:

- ✔ Supports 54 interrupts from a variety of sources (such as UARTs, GPIO and ADC)
- ✔ Routes interrupt sources to either the ARM920T's IRQ or FIQ (Fast IRQ) inputs
- ✔ Three dedicated off-chip interrupt lines operate as active high level sensitive interrupts
- ✔ Any of the 19 GPIO lines maybe configured to generate interrupts
- ✔ Software supported priority mask for all FIQs and IRQs

Note
For peripheral driver development purpose, notice that the external IRQ lines 5,6 and 7, which are ISA/X86 architecture based, are mapped to EP9302 external interrupt lines 22, 33 and 40, respectively. For further information about interrupts, including the EP9302 interrupt controller and map, refer to the **EP9301 User's Guide**, chapter 5.

4.2 Memory

TS-7400 uses three types of memory. The SDRAM is the fast access volatile memory used to run applications by the processor and the on-board flash is the non-volatile memory used for storage purpose. Flash memory may also be added using USB memory drivers.

On-Board SDRAM

The **TS-7400** uses 32 MB SDRAM technology to provide 32, 64, or 128 MB of high-speed volatile memory. The memory is soldered directly to the board, making the **TS-7400** more reliable in high-vibration environments.

The **TS-7400's** RAM is not contiguous in the physical memory map of the EP9302. But the MMU is programmed to remap the blocks of RAM to appear as a contiguous block of memory at the very beginning of the virtual memory map. In the case of a 256 Megabit SDRAM chip (32 MB), it is located at 0 through 32 MB in the virtual memory map.

Refer to the MMU section of this manual to understand how the physical memory is mapped and the virtual memory is translated.

Note
It is possible to use larger sizes of the SDRAM chip than the standard 32 MB one. The **TS-7400** is designed to accommodate both 32 MB and 64 MB chips, providing up to 128 MB of RAM memory. Contact Technologic Systems for larger SDRAM sizes.

On-Board NAND Flash

The **TS-7400** uses a NAND Flash chip for its on-board Flash resource. The physical address of the Flash chip is **0x6000_0000**. The on-board flash is broken up into partitions and accessed through the Linux driver framework known as "MTD", or Memory Technology Device. The partitioning is dynamic and depends on the DOS-style MBR found at sector 0 of the flash. This MBR can be changed by using the "fdisk" command on the /dev/mtdblock/0 device, but doing so is not recommended.

- ✔ /dev/mtdblock/0 - Whole disk block device driver
- ✔ /dev/mtdblock/1 - First MBR partition (bootloader kernel binary)

- /dev/mtdblock/2 - Second MBR partition (bootloader initrd)
- /dev/mtdblock/3 - Third MBR partition (Linux YAFFS2 filesystem)
- /dev/mtdblock/4 - 4th MBR partition (unused in default load)

The Linux YAFFS2 file system is a journaling file system that is aware of the wear-out mechanism of the NAND flash and incorporates ECC algorithms at the file system level to maximize Flash lifetime. It is also extremely tolerant of power failures during file write sequences.

Note
It is possible to use larger sizes of the NAND Flash than the standard 32 MB chip. The **TS-7400** is designed to accommodate both 32 MB or 128 MB chips. Contact Technologic Systems for larger Flash sizes.

USB Flash Drive or Compact Flash Card

Additional non-volatile storage may be added with a USB flash drive or a Compact Flash card. These devices supply additional non-volatile storage either for data or for a complete operation system distribution, such as Debian. A tar-file of Debian is available on the Technologic Systems website. Alternatively, the developer's kit includes a USB flash thumb-drive or Compact Flash card pre-loaded with Debian.

Flash memory provided by these devices behaves much as a hard drive does with sizes ranging from 32MB to 1GB. These products are inherently more rugged than a hard drive since they are completely solid-state with no moving parts. However, they have the added advantage of being removable media

Use of a Compact Flash card with TS-7400 SBC requires a USB Compact flash adapter, which will also be included in the TS-ARM Development Kit if requested. The USB flash drive can be hot swapped.

Note
Drivers are available in the TS-Kernel to support USB flash drives. One can load Debian OS with two scripts provided by the on-board flash TS-Linux file system or available for download at our website. First, invoke /usr/bin/loadUSBModules.sh, then run the script /usr/bin/loadUSB.sh to chroot into the Debian OS.

SD Memory Card

Technologic Systems has a full license for using the additional SD features which are reserved for members of the SD Card Association. This has allowed us to design both the hardware logic core and software specifically tuned to the capabilities of the **TS-7400** CPU using the official SD specification documents. Since both a Linux driver module and an ARM9 object file containing OS-independent access routines are provided to customers purchasing the board hardware, customers do not have to seek SD licensing themselves.

SD Memory Card technology provides large capacity and fast access combined with a compact and slim profile, making it very appealing for a wide range of next generation products and applications. In addition, SD Cards feature content protection, planned capacity growth, high-speed data transfer, and a write protect switch. These devices supply additional non-volatile storage either for data or for a complete operation system distribution, such as Debian, to be used with the **TS-7400** SBC**.**

The Technologic System SD Card core is a very small implementation and can be integrated on the **TS-7400** CPLD. Four 8-bit registers are available for the software layer to control the SD Card hardware:

Table: SD Card core registers

I/O Addr	Name	Description
BASE + 0	SDCMD	SD Command register
BASE + 2	SDDAT	SD Data register
BASE + 4	SDSTATE	SD State register
BASE + 6	SDCTRL	SD Control register

4.3 Glue Logic CPLD

The **TS-7400** ARM SBC's include an Altera MAXII CPLD which is responsible for taking control of the internal components communication through glue logic implementation. For instance, the CPLD is used to control the NAND flash through internal register configuration.

The CPLD has a watchdog timer, interfaces to the real-time clock and controls the EEPROM chip select. It also implements peripheral features that, together with EP9302 modules, makes available an advanced set of communication ports, DIO pins, ADC converters, and others. The inclusion of a CPLD on the SBC allows customized programming for customers with special needs, without having to do a more expensive board redesign.

The CPLD can be programmed using the TS-9441 JTAG header and special software/hardware supporting tools. Contact Technologic Systems for support on CPLD programming software and tools.

4.4 Real-Time Clock

The **TS-7400** optionally supports a Non-volatile Battery-backed real-time clock (RTC) which is soldered onto the board. This option uses an ST Micro M48T86PC1 module for the real-time clock function. This module contains the lithium battery, 32.768 kHz crystal, and a RTC chip with 114 bytes of battery-backed CMOS RAM. It will maintain clock operation for a minimum of 10 years in the absence of power.

The 114 bytes of non-volatile RAM, physically located in the RTC chip, are available to the user. Contact Technologic Systems for driver support.

The RTC is accessed differently on the **TS-7400** when compared to other TS products. DIO on the EP9302 are wired to pins on the RTC and must be accessed by twiddling the DIO in the correct manner. At the time of writing, there are no user-space tools available for reading/writing to the RTC, the modified kernel sources for accessing the RTC on the **TS-7400** can be found here: **ftp://oz.embeddedarm.com/ts-arm-sbc/ts-7400-linux/sources/rtc.c** These are byte-wide registers with the Index Register property of write only. The Data Register has a read/write property. Valid Index Register values are between 0 and 127, decimal. The first 14 index locations are used for accessing the RTC Time and Date registers. The next 114 locations are non-volatile RAM locations.

4.5 Watchdog Timer

The **TS-7400** incorporates a Watchdog Timer (WDT) unit. The WDT can be used to prevent a system “hanging” due to a software failure. The WDT causes a full system reset when the WDT times out, allowing a guaranteed recovery time from a software error. To prevent a WDT timeout, the application must periodically “feed” the WDT by writing a specific value to a specific memory location.

Table: Watchdog Control Registers

Register	Address	Access
WDT Control register	0x2380_0000	Read/Write
WDT Feed register	0x23C0_0000	Write Only

The WDT Control register must be initialized with the timeout period desired. This may be as short as 250 mS or may be as long as 8 seconds. After the WDT has been enabled,

the WDT counter begins. The application software can reset this counter at any time by "feeding" the WDT. If the WDT counter reaches the timeout period, then a full system reset occurs.

Table: Watchdog Timeout Register

Value	MSB	MID	LSB	Timeout Period
0x00	0	0	0	Watchdog Disabled
0x01	0	0	1	250 mS
0x02	0	1	0	500 mS
0x03	0	1	1	1 second
0x04	1	0	0	-- Reserved
0x05	1	0	1	2 seconds
0x06	1	1	0	4 seconds
0x07	1	1	1	8 seconds

In order to load the WDT Control register, the WDT must first be "fed", and then within 30 uS, the WDT control register must be written. Writes to this register without first doing a "WDT feed", have no affect. In order to clear the WDT counter (feeding the watchdog), a value of Hex 05 must be written to the WDT Feed register.

By default, a user process does not have the physical address space (access) of the watchdog registers mapped. When using the Linux OS, the watchdog can be reached from user C code by using the mmap() system call on the /dev/mem special file to map the areas of physical address space into process user address space. See section 3.4.

Warning
Use only the Watchdog Timer implemented by Technologic Systems in the CPLD. The Watchdog Timer included in the EP9302 has serious problems.

5 COMMON INTERFACES GENERAL INFORMATION

The purpose of this section is to provide general information about the common interfaces, such as Serial Ports and Digital Input/Output, which appear in more than one header or connector of the **TS-7400**. For further information on these features, refer to the Connectors and Headers section of this manual.

5.1 Serial Ports

The **TS-7400** has 3 TTL-level serial ports. Use the TS-9441 peripheral, RC-DB9 Connector, and null modem cable to connect to a console program on a PC.

5.2 Digital I/O

There are 20 individually configurable GPIO pins available on the TS-7400. Each of these pins can be configured as a floating input (high impedance, no internal pull-up/pull-down) or as an output driving a 0 or 1 logic signal. When configured as an input, the voltage thresholds are nominally at standard 3.3V LVCMOS levels, but with 400 mV of hysteresis (Schmitt triggers) for better noise immunity with slow slew signals. As an output, GPIO pins can drive 8mA while still presenting valid LVCMOS thresholds as measured at the output pin. More current can be sourced/sunk from output pins (2-3x more is typically safe) but the resulting output voltage will no longer be within LVCMOS thresholds-- e.g. attempting to source 30 mA out of a pin set to logic '1' will result in voltage of 1.0V instead of the nominal 3.3V.

The 20 GPIO pins are NOT 5V tolerant. This means if you drive any GPIO pin configured as input to above 3.3V, you may cause permanent damage to the device.

A common workaround for 5V tolerance is to use external Schottky diode clamps with series current limiting resistors. A better workaround is to use LVC245 or equivalent buffer chips. With diode clamps or external buffers guaranting 5V tolerance on input pins, one can safely interface to 5V TTL logic devices since in every other way 3.3V LVCMOS is compatible with 5V TTL logic. Note that 5V CMOS is NOT compatible with 3.3V LVCMOS as the logic '1' threshold for 5V CMOS is 3.5V. Since this is 0.2V above the 3.3V the TS-7400 is capable of, reliable operation cannot be guaranteed although the interfacing may actually seem to work some of the time.

Some GPIO pins have dual functionality. This includes the pins used for the GPBUS and the serial UART. GPBUS is a simple 8-bit multiplexed address/data bus and is described below. The serial UART on pins DIO_15, DIO_16, and DIO_17 are available for GPIO manipulation when the UART baud rate is set to '0' (disabled). If the UART is configured for operation with a non-zero baud rate, the GPIO pins are taken over by the UART core.

Using the GPIO pins in Linux involves getting access to the GPIO data direction register and data register (listed above). In general userspace applications, this involves opening the "/dev/mem" driver and using the mmap() system call to acquire a memory window where direct pointer manipulation of the hardware registers can occur. One may also write a kernel driver and interface to user applications through a defined API. Because it is difficult to anticipate the needs for a kernel driver API and the performance implications involved in even the simplest of userspace to kernelspace API's, Technologic Systems does not provide such a driver. Moving GPIO logic to the kernel may still be preferable if the bulk of GPIO client code is in the kernel or if there are strict real-time requirements to be met.

5.3 A/D Converters

The EP9302 A/D converter is standard on all TS-7000 series boards**.** The Cirrus EP9302 features a 5 channel, 12-bit Analog to Digital Converter with an analog multiplexer, having an input range of 0 to 3.3 V. The Cirrus A/D converter can do a maximum of 925 samples per second, and requires a settling time of 2 milliseconds between channel switches.

To maintain 12-bit accuracy, the analog signal being measured must have a low source impedance (less than 10 ohms). Otherwise, an operational amplifier may need to be added to buffer the A/D input. For detailed information, please see the Cirrus **EP9301 User's Guide**, page 518.

The EP9302 A/D converter can do a maximum of 925 samples per second, and requires a settling time of 2 milliseconds between channel switches. To maintain 12-bit accuracy, the analog signal being measured must have a low source impedance (less than 10 ohms). Otherwise, an operational amplifier may need to be added to buffer the A/D input. The EP9302 A/D converter should not be driven by a source impedance greater than 10 ohms to ensure accurate results (the EP9302 A/D converter has an input impedance that is not completely linear and may be as low as 10K ohms). For detailed information, please see the Cirrus **EP9301 User's Guide**, page 518.

Table: ADC Switch Values (EP9302)

Input to Measure	*ADC Switch Value*
ADC0	0x0000_0608
ADC1	0x0000_0680
ADC2	0x0000_0640
ADC3	0x0000_0620

The ADC0-3 pins can be found on the lower-header, pins 27-30 respectively, see section 6.4 General Header for more information on their location.

The following steps outline the software execution to use the Cirrus A/D converter:

1. Unlock the software lock before setting the TSEN bit in the ADCClk register by writing 0xAA to the ADCSWLock register (0x8090_00C0). “OR” in the TSEN bit (bit 31) to the ADCClkDiv register (0x8093_0090)
2. Unlock the software lock (again) before OR'ing in the ADCEN (ADC clock enable, bit 31) to 0x8093_0080
3. Clear bit 2, the ADCPD (ADC Power Down) bit, at 0x8093_0080. This bit MUST be set to 0 (see page 91 of the EP9301 User's Guide)
4. After unlocking the software lock, write the channel's magic value (see Cirrus EP9301 User's Guide, table 20-2) to the ADCSwitch register (0x8090_0018) to select that channel for the next data acquisition
5. Poll the ADCResult register (0x8090_0008) until bit 31 is not set
6. Using a 32 bit read operation, read the result from 0x8090_0008, masking off the upper 16 bits

Interpreting Cirrus A/D Converter

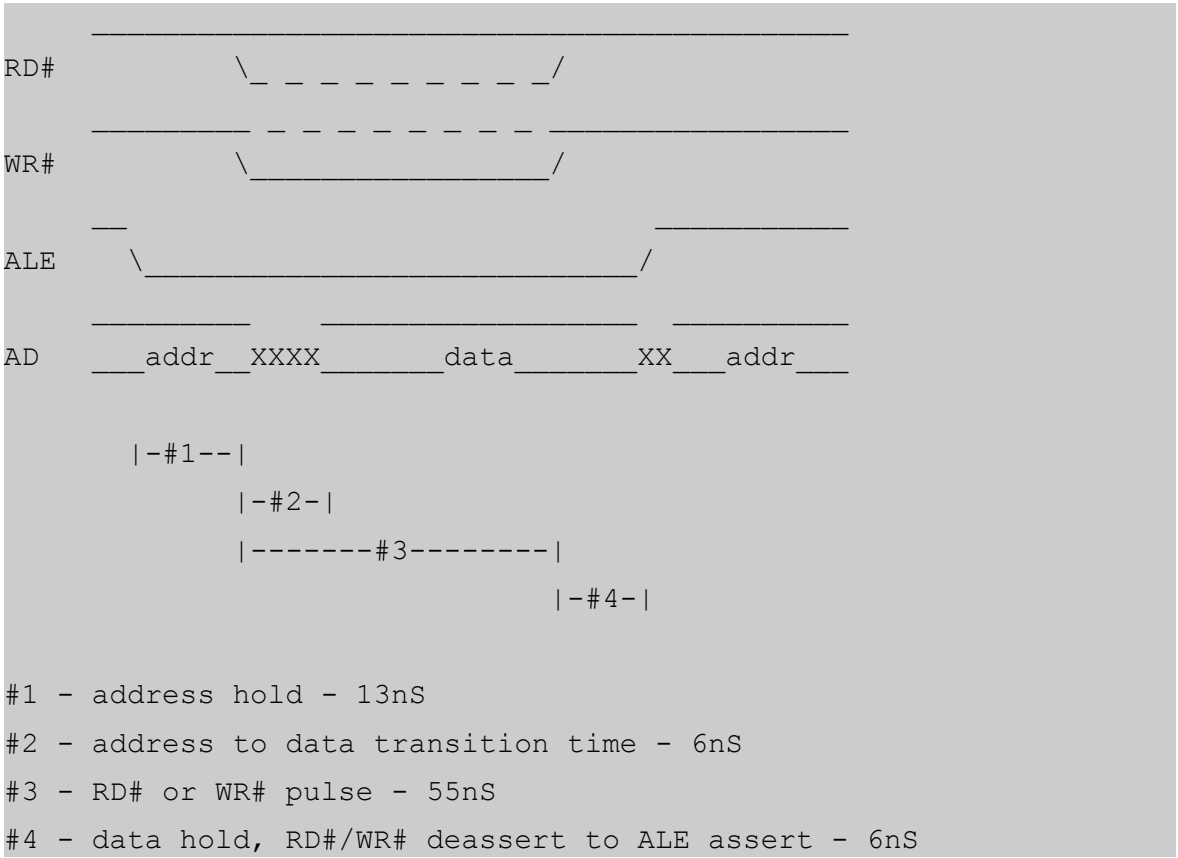
The Cirrus on-chip A/D converter is a successive approximation A/D converter. Each A/D channel is calibrated on the **TS-7400** and these 16-bit values are stored in a binary file located at /etc/ADC-calibration.dat. These calibration values minimize the offset errors and gain errors in the EP9302 A/D. It is important for the user program to use these values as per our sample code, which can be found either on our website or in the CD included in the Developer's Kit. Two reference points, 2.5 and .833 Volts, with the corresponding reference values stored in the calibration file. The file is structured to have the 16-bit reference value for 2.5v of each channel, 0-3; followed by the 16-bit reference value for .833v of each channel, 0-3.

The reference points are stored as a 16 bit value, and should be used to correlate the values returned by the Cirrus A/D converter to voltage.

5.4 General Purpose 8-bit Multiplexed Bus (GPBUS):

To use the general purpose bus, GPIO pins 7-0 first MUST be programmed as general purpose outputs by writing an 0xff to **0x12c0_0000**. Also, GPIO pins 8-10 must also be programmed as outputs if all 3 strobes will be used (ALE, RD, and WR). A strobe pin set up as a GPIO input will not be toggled at the appropriate times during a GPBUS bus cycle-- this can save GPIO pins in the case of a read-only or write-only bus, but is probably not what you want. GPIO #8 is the ALE, #9 is RD, and #10 is WR. The typical usage of ALE is an active high signal representing the period of time during which the address/data bus is driving the address. The read and write strobes are typically active low and signified as RD# and WR#. The polarity of each strobe is programmed by the contents of the GPIO data register for pins 8, 9, and 10. Knowing the data register bit locations of ALE, RD, and WR, you should then program them to default output ALE asserted with RD and WR deasserted. In the case of ALE active high, WR/RD active low, the value written to bits 2-0 at **0x12c0_0003** should be 7 (binary 111). For the purposes of the timing diagrams below, we will assume this configuration.

The default timing of the GPBUS cycle uses a 55 nS read/write pulse when reading/writing the data register at **0x60c0_0000**. This is with the EP9302 bus cycles programmed for 60nS operation. (EP93xx SMCBCR6 register value 0x34c2). Although you may not change this value since the NAND flash chip is on the same EP93xx chip select, you may change the SMCBCR7 register for slower or faster operation and use the data register alias at 0x70c0_0000. The GPBUS waveform for 0x60c0_0000 accesses looks like this:



The above timing should be compatible with a standard 74HC373 octal latch chip should you wish to de-multiplex the address/data bus into something more similar to ISA/PC104 externally. If you have questions about whether a particular design will work or not, please feel free to contact Technologic Systems with your schematic for review.

For fast back-to-back accesses to consecutive addresses, you should use the the data register at 0x60c0_0002. This is very useful for reading contiguous memory quickly without having to interleave updates to the address register. Although there is only an 8-bit address register, it is trivial to use other free GPIO lines as high address lines should you need more than 256 bytes of address space for the GPBUS.

Should you require it, a 14.7456Mhz clock can be output from the TS-7400 on DIO_13. To enable this, set bit 3 of the 8-bit register at address 0x12000000 and set the data direction for DIO_13 as output. External IRQ is also available on DIO_11 by similarly setting bit 1. One can also enable DMA with the EP9302 CPU M2M DMA controller by allowing DRQ on DIO_12 (bit 2).

If you choose to use DMA or IRQ support for the GPBUS, you will likely want to write a kernel driver. Simple userspace applications can not directly set up interrupt handler functions or translate the virtual addresses of process data structures to the physical addresses needed by the DMA controller. When writing a kernel driver in C, one should be aware of certain details:

- To set a handler on the GPBUS IRQ, you must request IRQ #33 and allow IRQ sharing (the UART and SD drivers also generate the same IRQ). This means calling the request_irq() kernel function with SA_SHIRQ in the 'flags' parameter.
- To use DMA, you must manipulate the M2M1 EP9302 DMA channel. Details on its programming can be found in the EP9302 CPU User Manual from Cirrus Logic at http://www.cirrus.com
- To access various registers in arbitrary physical address space, use the __ioremap() function and not the ioremap() kernel function. ioremap() does not expect physical addresses and will apply an offset to your requested address on the EP9302 ARM Linux kernel.
- If you are unable to write a driver yourself, you may consider contracting Technologic Systems to write one for you. Often times, TS engineers can accomodate your requirements for less cost than it takes to accomplish internally.

6 CONNECTORS AND HEADERS

6.1 10/100 Base-T Ethernet Connector

The EP9302 Ethernet LAN controller incorporates all the logic needed to interface directly to any MII compatible Ethernet PHY chip. A low-power Micrel KS8721 chip is used to implement the Ethernet PHY function and an integrated RJ-45 connector with built-in 10/100 transformer and LED indicators completes the Ethernet sub-system.

The **TS-7400** has both a LINK/ACTIVITY LED and a 10/100 speed LED built into each RJ-45 connector that indicates the current Ethernet status. The LINK LED (left side of connector, green) is active when a valid Ethernet link is detected. This LED should be ON whenever the **TS-7400** is powered and properly connected to a 10/100BaseT Ethernet network. The LINK/ACTIVITY LED will blink to indicate network activity for either inbound or outbound data. The SPEED LED (right side of connector, amber) will be on when a 100Mb network is detected and off for a 10Mb network. Both of these LEDs are controlled by the KS8721 and do not require any overhead by the processor.

The Ethernet PHY chip can be powered down, under software control, to save approximately 90 mA of current consumption. This is controlled by the EP9302 Digital output on Port H, bit 2. A logic zero will power down the KS8721 PHY interface.

Note
TS-Kernel provides all the software support to use the EP9302 10/100 Ethernet core. For more details, find the TCP/IP configuration instructions on the Linux documentation.

6.2 USB Connector

The USB Connector on the **TS-7400** provide two USB interfaces for the user. These are directly connected to the EP9302 processor, which integrates an USB dual-port Open Host Controller Interface (OHCI), providing full-speed serial communications ports at a baud rate of 12 Mbits/sec. Up to 127 USB devices (printer, mouse, camera, keyboard, etc.) and USB hubs can be connected to the USB host in the USB “tiered-star” topology. This includes the following features:

- ✔ USB 2.0 compatible
- ✔ OHCI Rev 1.0 compliant
- ✔ USB device connections support at both low-speed (1.5 Mbps) and full-speed (12 Mbps)
- ✔ Root HUB integrated with 2 downstream USB ports
- ✔ Transceiver buffers integrated, over-current protection on ports
- ✔ Supports power management
- ✔ Operates as a master on the bus

Note
TS-Kernel implements all the necessary driver support to enable the USB OHCI. Also, a wide variety of USB drivers for devices such as mouse, keyboard and flash memory are available. Refer to the Linux for TS-ARM User's Guide or contact us for further information on how to integrate an USB device and an USB Linux driver with your **TS-7400**.

6.3 SD Card Connector – CPLD

The SD Card socket (ALPS connector) at the back side of the **TS-7400** enables SD Cards to be plugged to the SBC. The hardware core implemented by Technologic Systems is integrated inside the on-board CPLD. Technologic Systems has written a binary Linux driver module and a set of generic, OS-independent read/write routines for accessing the SD flash inside of an ARM object (.o) file. The format of the SD card must be in EXT2 format for proper operation with Linux as a root file system.

6.4 General Header

Besides the 10/100 ethernet jack, SD card socket, and USB host ports, the TS-7400 also includes a .1" pin spacing external header for board-to-board interfacing. The TS-7400 external interface uses a total of 66 pins which are broken up between a 40 pin lower header (closest to board edge) and a 26 pin upper header. Pin numbering is arranged as follows:

Upper header

2	4	6	8	10	12	14	16	18	20	22	24	26
1	3	5	7	9	11	13	15	17	19	21	23	25

Lower header

2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38	40
1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39

The combination of these two headers provides the following functionality:

- ✔ High speed dedicated (Up to 14.7Mhz) SPI bus
- ✔ USB 2.0 full-speed (12 Mb/s) port
- ✔ 3 TTL-level serial UART ports (1 with tx-enable output for RS485)
- ✔ Simple 8-bit multiplexed general purpose parallel bus with
- ✔ 14.7Mhz clock, IRQ and DMA support and capable of 10Mbyte/sec operation.
- ✔ 20 bidirectional Schmitt-trigger GPIO pins
- ✔ 1.8V, 3.3V, and 5V power supply pins (5V power can optionally be input to the TS-7400 through this connector)
- ✔ 4 12-bit 0-3.3V analog input (ADC) pins
- ✔ I2S/AC97 audio codec interface
- ✔ External reset input
- ✔ Low-level bootup hijack facility to recover dead boards. (Used in TS-9441 production blast/test/recovery boards)

Upper header Pin-Out

Pin #	Name	Function
1	TDO	TS-production reserved
2	TMS	TS-production reserved
3	GND	Ground
4	TDI	TS-production reserved
5	BLAST_BOOT#	input, low will hijack CPU SPI bootstrap
6	TCK	TS-production reserved
7	UART0_TXD	ep9302 UART #0 output (/dev/ttyAM0 in Linux)
8	UART0_RXD	ep9302 UART #0 input (/dev/ttyAM0 in Linux)
9	SPI_MISO	SPI master-in, slave-out input
10	3.3V	3.3V TS-7400 regulator output (or input, if U6 regulator is not populated)
11	BLAST_EE_CS#	chip-select for SPI boot hijack EEPROM
12	SPI_MOSI	SPI master-out, slave in output
13	FLASH_CS#	chip-select used for 2 megabyte SPI flash on TS-9441 (used by TS-SPIBOOT boot program)
14	SPI_CLK	SPI clock output
15	5V	5V regulated power input/output
16	EXT_RESET#	External reset input, low triggers board reset
17	BLAST_PRESENT#	Boot hijacker present input
18	GND	Ground
19	PORTB_4	CPU connected GPIO pin, 5V tolerant with external series resistor.
20	GND	Ground
21	EN_5V	Switching power supply enable input, open-drain, pull low to disable 5V switcher
22	EP_USB+	EP93xx CPU USB port data signal
23	FIL_VIN	Reserved
24	EP_USB-	EP93xx CPU USB port data signal
25	PORTB_7	CPU connected GPIO pin, 6.49k pull-down and 1k series resistors. 5V tolerant.
26	USB_5V	USB 5V power

Lower header pin-out

Pin #	Name	Function
1	DIO_00	GPIO #0 or GPBUS multiplexed address/data #0
2	3.3V	3.3V TS-7400 regulator output (or input, if U6 regulator is not populated)
3	DIO_01	GPIO #1 or GPBUS multiplexed address/data #1
4	DIO_02	GPIO #2 or GPBUS multiplexed address/data #2
5	DIO_03	GPIO #3 or GPBUS multiplexed address/data #3
6	DIO_04	GPIO #4 or GPBUS multiplexed address/data #4
7	DIO_05	GPIO #5 or GPBUS multiplexed address/data #5
8	DIO_06	GPIO #6 or GPBUS multiplexed address/data #6
9	DIO_07	GPIO #7 or GPBUS multiplexed address/data #7
10	DIO_08	GPIO #8 or GPBUS ALE (address latch enable)
11	DIO_09	GPIO #9 or GPBUS RD (read strobe)
12	GND	Ground
13	DIO_10	GPIO #10 or GPBUS WR (write strobe)
14	DIO_11	GPIO #11 or GPBUS IRQ (active high, level sensitive)
15	DIO_12	GPIO #12 or GPBUS DRQ (uses EP93xx M2M1 dma channel)
16	DIO_13	GPIO #13 or GPBUS 14.7456Mhz clock
17	DIO_14	GPIO #14
18	5V	5V regulated power input/output
19	DIO_15	GPIO #15 or serial UART #2 transmit-enable (/dev/ttyTS0 device node in Linux)
20	DIO_16	GPIO #16 or serial UART #2 receive data input (/dev/ttyTS0 device node in Linux)
21	DIO_17	GPIO #17 or serial UART #2 transmit output (/dev/ttyTS0 device node in Linux)
22	DIO_18	GPIO #18 or serial UART #0 transmit output (/dev/ttyAM0 device node in Linux)
23	DIO_19	GPIO #19 or serial UART #0 receive data input (/dev/ttyAM0 device node in Linux)
24	UART1_RXD	serial UART #1 receive data input (/dev/ttyAM1 device node in Linux)
25	UART1_TXD	serial UART #1 transmit output (/dev/ttyAM1 device node in Linux)
26	1.8V	1.8V regulator output (or input if U7 regulator not populated)
27	ADC0	EP93xx CPU analog to digital channel #0 input
28	ADC1	EP93xx CPU analog to digital channel #1 input
29	ADC2	EP93xx CPU analog to digital channel #2 input
30	ADC3	EP93xx CPU analog to digital channel #3 input
31	GND	Ground
32	ABIT_CLK	EP93xx CPU AC97 audio codec signal
33	ASDO	EP93xx CPU AC97 audio codec signal
34	ASYNCH	EP93xx CPU AC97 audio codec signal
35	ARST#	EP93xx CPU AC97 audio codec signal
36	ASDI	EP93xx CPU AC97 audio codec signal
37	SSP_TX	EP93xx SPI/SSP/I2S signal
38	SSP_RX	EP93xx SPI/SSP/I2S signal
39	SSP_FRM	EP93xx SPI/SSP/I2S signal
40	SSP_CLK	EP93xx SPI/SSP/CLK signal

6.5 TS-9441 Console Header

Note
The information on this section applies to the TS-9441 recovery/console peripheral board for the TS-7400.

The console header brings out a serial port. Use the RC-DB9 connector to connect the TS-7400/TS-9441 to a DB9 cable and to your PC running a terminal emulator.

6.6 TS-9441 JTAG Header

Note
The information on this section applies to the TS-9441 recovery/console peripheral board for the TS-7400.

The JTAG header can be used to program the on-board CPLD using special software and hardware support tools. However, it is not available for application debug purposes since it has no connection to the EP9302 JTAG interface. The following table shows which pins of the JTAG header are used for the JTAG interface signals:

Table: JTAG signals at JTAG Header

PIN	Signal	Description
9	TCK	Test Clock
11	TDI	Test Data In
13	TMS	Test Mode Select
14	TDO	Test Data Out
1, 3, 5	3.3 VCC	Power Supply
10, 12	GND	Ground

The JTAG header is also utilized for jumper configuration. See the Jumpers section of this manual for more details.

Technologic Systems has made the design choice to save on board real-estate and not bring out the JTAG header. If you need access to the JTAG pins, a skilled technician can solder wires to the pins.

6.7 Power Supply Connector

The **TS-7400** requires regulated 5VDC at 450 mA @ 200 MHz(maximum). It is possible to lower this power significantly by lowering the CPU clock rate or by powering-down the Ethernet PHY chip. For example, by shutting down the Ethernet PHY chip and scaling down the CPU clock rate to 20 MHz, one can obtain power consumptions less than 150 mA. If you really need a low-power board, consider the **TS-7260**, which is optimized for low power applications.

A quick release screw-down terminal block for the 5V power and power GND connections is provided on the TS-9441 for easy connection to an external power supply.

Warning
Supply voltages over 6 VDC may damage the **TS-7400**.

Be sure to use a regulated 5 VDC power supply, preferably with current limiting to 1 to 3 Amps. A current limited supply is very forgiving of common errors during development. A PC power supply that may be capable of supplying 20 Amps or more is not recommended. It is possible to do irreversible damage to the **TS-7400** if the polarity on the power leads is reversed.

7 LEDS, JUMPERS AND BUTTONS

7.1 Status LEDs

The **TS-7400** has two LEDs (one Red and one Green) available for user software. These LEDs may be used for diagnostics, status messages, and simple output. When power is first supplied to the **TS-7400**, both LEDs are immediately turned on under hardware control. Once the processor begins execution, the LEDs are turned off, and then flashed on and off again briefly. After booting is complete, these LEDs can be used for user applications.

The RED and Green LEDs can be controlled at physical address location **0x8084_0020**. Bit 1 is the Red LED and bit 0 is the Green LED. A Logic “1” turns the LED on.

7.2 Buttons

The Reset button on the TS-9441 can be used to cycle power on the **TS-7400**.

7.3 Jumpers

There are 2 jumpers on the TS-9441, labeled Write En and Boot. The Boot jumper is used to recover a bricked TS-7400 by booting from the TS-9441. When the Write Enable jumper is installed, it is possible to write to the TS-9441.

8 SPECIFICATIONS

To ensure optimum product operation you must maintain the operational environmental specifications listed in the table below.

Table: Environmental Specification for TS-7400

Environmental Specification	Standard Temp	Extended Temp
Ambient Temperature	-20° to +70° C The internal temperature must not exceed +70° C.	-40° to +85° C Extended temperature range is also standard in our TS-7300 product **Note:** Extended Temp requires lower CPU speed (<=166Mhz) at higher temperatures **Note:** Refer to your product manual, or contact Technologic Systems if the environmental temperature of the location is in doubt.
Relative Humidity	0 to 90% relative humidity. Not to exceed 90% non-condensing.	Not to exceed 90% non-condensing.

9 FURTHER REFERENCES

- **Linux for TS-ARM User's Guide**
  (http://www.embeddedarm.com/documentation/software/arm-tslinux-ts72xx.pdf)
- **TS-7400 Data Sheet**

  (http://www.embeddedarm.com/documentation/ts-7400-datasheet.pdf)
- **EP9301 User's Guide**
  (http://www.embeddedarm.com/documentation/third-party/ts-7000_ep9301-ug.pdf)
- **EP9301 Data sheet**
  (http://www.embeddedarm.com/documentation/third-party/ts-7000_ep9302-ds.pdf)
- **TS-7000 Yahoo Users' Group** (http://groups.yahoo.com/group/TS-7000/)
- **TS-7400 schematic**
  (http://www.embeddedarm.com/documentation/ts-7400-schematic.pdf)
- **TS-7400 mechanical drawing**

  (http://www.embeddedarm.com/documentation/ts-7400-mechanical.pdf)
- **TS-7400's download section**

  (ftp://ftp.embeddedarm.com/ts-arm-sbc/ts-7400-linux/)
- **TS-9441 schematic**

  (http://www.embeddedarm.com/documentation/ts-9441-schematic.pdf)

APPENDIX A: DOCUMENT HISTORY

Date of Issue/Revision	Revision Number	Comments
10/06/2006	0.1	Initial release
10/23/2006	0.2	Added sections on SD locking, security features, and GPIO
10/24/2006	0.4	Added mention of "/fastboot" file autoboot abort
01/11/2007	0.6	Errata for Rev B CPLD
05/18/2007	0.8	Modified UART naming convention for consistency
05/22/2008	1.0	Full manual released
07/08/2008	1.1	Fixed broken web links
06/01/2008	1.2	Updated mailing address
04/01/2010	1.3	Fixed section on using EP9302 ADC
05/26/2010	1.4	Fixed RTC section

APPENDIX B: MEMORY AND REGISTER MAP

Address Region	Function
0xF000_0000 - 0xFFFF_FFFF	nCS0 (not used)
0xD000_0000 - 0xDFFF_FFFF	SDRAM (not used)
0xC000_0000 - 0xCFFF_FFFF	SDRAM (not used)
0x8084_0000 - 0x8084_00C8	GPIO control registers
0x8000_0000 - 0x800F_FFFF	AHB mapped registers
0x7000_0000 - 0x7FFF_FFFF	CS7 (bit bus cycles)
0x6000_0000 - 0x6FFF_FFFF	CS6 (Flash)
0x3000_0000 - 0x3FFF_FFFF	CS3 (not used)
0x2000_0000 - 0x2FFF_FFFF	CS2 (16-bit bus cycles)
0x1000_0000 - 0x1FFF_FFFF	CS1 (8-bit bus cycles)
0x0001_0000 - 0x0000_FFFF	SDRAM region

Register Address	Function
0x8090_0020	Cirrus A/D lock register
0x8090_0018	Cirrus A/D channel select register
0x8090_0008	Cirrus A/D result register (RO)
0x808D_0000 - 0x808D_FFFF	UART2 control registers
0x808C_0000 - 0x808C_FFFF	UART1 control registers
0x808A_0000 - 0x808A_FFFF	SPI control registers
0x8084_0044	LCD_EN, LCD_RS, LCD_WR direction reg.(bits3-5)
0x8084_0040	LCD_EN, LCD_RS, LCD_WR data reg.(bits3-5)
0x8084_0034	DIO_8 direction register (bit 1)
0x8084_0030	DIO_8 data register (bit 1)
0x8084_0020	On-board LEDs register (bits 0, 1)
0x8084_0018	Port C direction register
0x8084_0008	Port C data register
0x8081_0000 - 0x8081_FFFF	Timer Control registers
0x8080_0000 - 0x8FFF_FFFF	APB mapped registers
0x800B_0000 - 0x800B_FFFF	VIC 0 registers
0x8006_0000 - 0x8006_FFFF	SDRAM control registers
0x8002_0000 - 0x8002_FFFF	USB registers
0x8001_0000 - 0x8001_FFFF	Ethernet MAC registers
0x60c0_0002	GPBUS data register with address auto-increment (read/write of this register initiates GPBUS bus cycle with automatic increment of 8-bit address register at 0x12c0_0002)
0x60c0_0000	GPBUS data register (read/write of this register initiates GPBUS bus cycle)
0x62c0_0002	GPBUS address register (write only)
0x6040_0003	ECC bits 21-16, read/write of this register resets ECC bits bit 5:0 - ECC bits 21-16 bit 7:6 - reserved * ECC is fed by read or write data to the flash data register at 0x6000_0000
0x6040_0002	ECC bits 15-8
0x6040_0001	ECC bits 7-0
0x6040_0000	NAND flash control register bit 0 - ALE signal bit 1 - CLE signal bit 2 - CS signal

Register Address	Function
	bit 4:3 - reserved bit 5 - flash busy signal bit 7:6 - reserved
0x6000_0000	NAND flash data register
0x23C0_0000	WDT Feed register (bits 0-2)
0x2380_0000	WDT Control register (bits 0-2)
0x2340_0000	TS-CPLD revision register bit 2:0 - CPLD revision number bit 7:3 - reserved
0x2300_0000	SPI EEPROM chipselect control bit 0 - reserved bit 1 - onboard SPI EEPROM chip-select enable bit 2 - offboard (boot hijacker) SPI EEPROM chip-select bit 7:3 - reserved
0x2200_0000	Model Number (bits 0-2)
0x1300_0000-0x1300_0003	SD card controller (/dev/sdcard0/disc0 in Linux)
0x1240_0001	UART #2 (/dev/ttyTS0 in Linux) DAT register
0x1240_0000	UART #2 (/dev/ttyTS0 in Linux) STAT register bit 0 - TBRE, Transmit buffer empty (RO) bit 1 - DR, Receive data ready (RO) bit 2 - OERR, Overflow error (RO) bit 4:3 - reserved bit 7:5 - MODE, baud rate (RW) 0 - 115200 8N1 1 - 57600 8N1 2 - 38400 8N1 3 - 19200 8N1 4 - 9600 8N1 5 - 4800 8N1 6 - 2400 8N1 7 - UART off, IRQ disabled
0x12c0_0003	GPIO data register for DIO_8 to DIO_15
0x12c0_0002	GPIO data register for DIO_0 to DIO_7/GPBUS address register
0x12c0_0001	GPIO direction for DIO_8 to DIO_15 ('1' means output)
0x12c0_0000	GPIO direction for DIO_0 to DIO_7 ('1' means output)
0x1200_0001	GPIO direction/data for DIO_16 to DIO_19 bit 3:0 - data register bit 7:4 - data direction register ('1' means 'output')
0x1200_0000	TS-7400 control register bit 0 - if set, enables UART #0 on DIO_18 and DIO_19 pins bit 1 - if set, enables IRQ on DIO_11 pin bit 2 - if set, enables DRQ on DIO_12 pin bit 3 - if set, enables 14.7456Mhz clock on DIO_13 pin bit 7:4 - reserved

APPENDIX C: TS-ARM SBC FEATURE MATRIX

Product	TS-7200	TS-7250	TS-7260	TS-7300	TS-7400
CPU	200 Mhz AMR920T	200 Mhz AMR920T	200 Mhz AMR920T	200 Mhz AMR920T	200 Mhz AMR920T
PC/104 connector	Yes	Yes	Yes	Yes	No
On-board FPGA	No	No	No	Yes	No
RAM	32 MB	32 MB	32 MB	32 MB	32 MB
Optional RAM	64 MB	64 MB 128 MB	64 MB 128 MB	64 MB 128 MB	64 MB 128 MB
On-board Flash	8 MB	32 MB	32 MB	No	32 MB
Optional on-board Flash	16 MB	64 MB 128 MB 256 MB	64 MB 128 MB 256 MB	Possible Contact us	64 MB 128 MB 256 MB
Standard A/D	Yes - 2 ch	Yes	Yes - 2 ch	Yes - 1 ch	Yes - 4 ch
Optional A/D	8 ch	8 ch	No	No	No
Ethernet	1x 10/100	1x 10/100	1x 10/100	2x 10/100	1x 10/100
USB1.1 ports	Yes - 2x	Yes - 2x	Yes - 2x	Yes - 2x	Yes - 2x
VGA Video Out	No	No	No	Yes	Yes
IDE Compact Flash	Yes	No	No	No	No
SD Card Interface	No	No	Yes - 1x on RevB	Yes - 2x	Yes - 1x
Digital I/O	20	20	30	55	20
TS-XDIO	No	No	Yes - 1	Yes - 2	No
RS-485	Opt. full/half	Opt. full/half	Opt. full/half	Opt. full/half	Opt. full/half
Standard COM Ports	2	2	3	10	3 TTL
Optional COM Ports	No	No	2	Yes	No
RS-232 Console	Yes	Yes	Yes	Yes	No
RTC	Opt.	Opt.	Opt	Opt.	Opt.
LCD Interface	Yes	Yes	Yes	Yes	No
Keypad Interface	Yes	Yes	Yes	Yes	No
Linux2.4	Yes	Yes	Yes	Yes	Yes
Bootloader	Redboot	Redboot	Redboot	Linux	Linux
Real-Time RTAI	Yes	Yes	Yes	Yes	Yes
Java	Yes	Yes	Yes	Yes	Yes
USB WiFi Support	Opt.	Opt.	Opt.	Opt.	Opt.
USB Flash Support	Opt.	Opt.	Opt.	Opt.	Opt.
Extended Temperature	Yes	Yes	Yes	Yes	Yes
Switching-Mode Power Supply	No	No	Yes	No	Opt.
Quantity 1 Pricing	$149.00	$149.00	$179.00	$219.00	$129.00
Quantity 100 Pricing	$119.00	$119.00	$149.00	$189.00	$99.00

APPENDIX D: CONTACT TECHNOLOGIC SYSTEMS

16525 East Laser Drive
Fountain Hills, AZ 85268
TEL 1.480.837.5200
FAX 1.480.837.5300

www.embeddedARM.com
support@embeddedARM.com

Call us Monday-Friday, **from 9 am to 5 pm**, Arizona-USA time;
or email us at any time.

Our engineers answer tech support calls and are more than happy to talk to you about your needs and help you find the best solution for your project.