10.0 Peripheral Equipment

There are a number of peripheral equipment that interface to the system through one of the serial ports on the surface and the subsea ends. Following is a description for each of these interfaces.

10.1 Touch Screen Interfaces

The touch screens have been installed on Rocket Port Com 0 and Rocket Port Com 1. Each is set at 9600Baud. This setup is not user configurable, and, as well needs to be the same as the driver software for the touch screens. There's not much to be setup here. In the event the touch screen needs to be reconfigured see the manual regarding the Linux driver that came with the touch screens.

10.2 Tether Management System Interface:

This serial string is defined by MBARI. Baud rate 9600. To reconfigure this string the source code needs to be recompiled. This file is telemetryCom.C

10.3 Teleos Interface:

The Teleos interface is at 38Kbaud and is configured in the C file telemetryCom.C. To edit this string the user needs to edit and recompile the C code. The Transmission to the Teleos consists of the information that is required for the Teleos control system to maintain control of the vehicle. This list of datum was provided by MBARI. Basic datum are Heading, Depth, Accelerations, Vehicle attitude etc. The information from the Teleos is for the thrusters.

10.4 Logging Interface:

The logging interface is also defined by MBARI and can be configured in the C files. Any configuration change will ed to be re-compiled to be effected. Logging Data can be any of the telemetry datum either from the surface or the sub or from any of the peripheral devices. This string is defined in telemetryCom.C Logging is sent at 9600, None, and 1 Stop Bit.

10.5 FOG Gyro Interface:

The interface to the FOG gyro is set up to handle the default sting from the FOG. Other string parsing can be done by editing the C code for the vehicle. Any changes require re-compiling of the code for the change to be effected. There are two strings to/from the FOG, one is for the data from the FOG and one is for the control of the FOG. The control to the FOG is not necessary for the operation of the Gyro if the Gyro has been set up by the software that runs on Windows from Octans. The Communications from/to the FOG are at 38Kbaud.

10.6 Paroscientific Pressure Transducer Interface:

The interface to the Pressure Transducer interface board is at 9600 Baud, no parity, and 1 stop bit.

10.7 KVH Compass Interface:

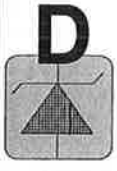
The interface to the backup compass is set to 9600 Baud. This interface is user selectable to either setup the KVH C100, defining what kind of data it outputs , or to receive this data. For best results the backup compass need to be set to output the heading at the highest speed possible without any delay. Even at the highest speed possible the backup compass will not be able to give the same response as the FOG for the auto-pilot functions. When using the backup compass for auto pilot functions the settings for autopilot Proportional, Integral, and Derivative control will have to be reduced to stop the vehicle from oscillating when the auto pilots are engaged.

10.8 DVL Interface:

The DVL interface is different than any of the other interfaces in that this telemetry control system is "camping" on the transmitted data that typically goes directly to the Fiber Optics can to be transmitted directly to the surface. Normally this data from the DVL is only used by the Teleos control system, however the telemetry control system will also use the speed information to maintain the vehicle speed that was attained when cruise control was engaged . To adjust the cruise speed the pilot must disengage the cruise control, then attain the desired speed manually, then re-engage the cruise control.

10.9 Altimeter Interface:

The altimeter interface is for the reception of the altitude from altimeter #2.



11.0 Ventana Software Documentation

11.1 General Description:

This description is separated into two parts: the Surface and Subsea systems. The Surface system is event driven by a high level scripting language Tcl which is glued together with the Commands created by the low level C/C++ objects. This system uses POSIX threads extensively to enable concurrent programming rather than a sequential program and thus can use the processor time more efficiently.

11.1.1 Login and Users:

When the system is turned on the login name is ***ventana*** and the password is ***venta1na***. The system has been set up so that this user (***ventana***) will have startx automatically installed and started. After startx is finished loading an ICON for the telemetry is at the bottom of the screen. It is the picture of a goat and can be clicked to start the Ventana software.

For remote access over the LAN another user has been set up. This user name is ***rventana*** and the password is ***venta1na.*** The remote user can become a super user by the command ***su*** with the password ***ventana.***

11.2 High Level Language (GUI Creation)

The GUI is created with Tcl/Tk and C/C++. Following are descriptions for each of these tools and the classes and objects formed within them.

11.2.1 Tcl

Tcl was created at the University of California at Berkeley and maintained by John K. Ousterhout at Sun Microsystems Laboratories. It is a high level scripting language that can create the GUI without the need for C/C++ or any other language. However, TK is used as a tool to combine the C/C++ program. One benefit of using Tcl is that it is possible to create sophisticated GUIs entirely as Tcl scripts, without writing any C code at all.

11.2.2 Tk

TK is a widget creation tool. It allows you to create the graphical user interfaces for the X Window System by writing Tcl Scripts. It is implemented as a C library package that can be included in C application.

We have two copies of most of the Tcl files for the multiple display purposes. Where in each copy of the file, it has to specify the correct display environment for each screen.

The TCL files that create graphics for the system are :

but_small_popup.tcl, but_small_popup2.tcl

This file will contain the code to create button which can be displayed anywhere in the screen. There are two types of button : alternate and momentary. Alternate button will stay pressed (On) if the user click it once, and Off when the user click on it again. Momentary button will stay On as long as the user is pressing it, and goes Off soon after the user release it.

The user can select the connection, type, and color for each button. To remove the button, user has to click fill in the connection name for the button to be removed.

slidebar_popup.tcl, slidebar_popup2.tcl

This file contain the source code to create slidebar. The value of slidebar goes from 0 to 100 (percent). When the user set the value, the percentage value is then mapped into –5 to 5 volts. User can also select connection, color, and position for each slidebar. If the user needs to remove the slidebar, one should enter the correct connection field of the slidebar and press "Remove".

compass.tcl, compass2.tcl

This file create the heading, auto heading graphic, and also heading histories (70 values). It also have the information of the number of turns the vehicle has made, the validity of Gyro value (red if invalid, and green if valid)

depthbar.tcl, depthbar2.tcl
Contains the current depth, auto depth, vertical speed, and the depth histories (50 values) graphic.

vehicle.tcl, vehicle2.tcl
The graphic will show the position of the vehicle, 2 cameras and 2 horizontal thrusters, and 2 vertical thrusters in 3D graphical representation.

lights.tcl, lights2.tcl
The light graphic will show each position of 10 lights in the vehicle. The user can also change the state of the light (on/off) by clicking any of the camera. On state will change the color to white, and Off to be blue.

bar.tcl, bar2.tcl
Bar graph will represent the voltage for some of the analog channel that the user is going to have. The user is able to set the maximum, minimum, average value for the analog value. In addition, the user can also set the yellow range and red range of the graph. If the voltage goes inside red range values, then an alarm will be generated.

alarm.tcl, alarm2.tcl
The code will be dedicated to generate alarm graphic. The alarm description and name for the channel has already been predefined.

11.3 Sub Menus:

There is also a directory called submenu that is dedicated for the submenu files under "Maintenance" button in the GUI. It is inside this submenu that the user can call the graphics. Under Maintenance, there exists 2 submenus which are Surface and Vehicle. The structure for the two submenus are:

11.3.1 Surface submenu

- Button
- Slidebar,
- Digital element.
- Analog element.

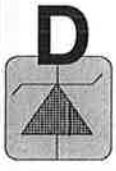
11.3.2 Vehicle submenu

- Compass
- Light
- Vehicle
- Digital element
- Analog element :
    * Depth graphic
    * Bar graph

11.4 Togl:

Togl is a Tk widget for OpenGL rendering. Togl is a special program created by ???????? that allows the programmer to create and manage a special Tk/OpenGL widget with Tcl and render into it with a C program. That is, a typical Togl program will have Tcl code for managing the user interface and a C program for computations and OpenGL rendering. The vehicle graphic is created using this program. We are currently using Togl version 1.5 written by Ben Bederson and copyrighted by Brian Paul.

11.4 MESA / OpenGL:



Mesa is a 3-D graphics library with an API which is very similar to that of OpenGL. It is a viable alternative to OpenGL. Mesa utilizes the OpenGL Command syntax or state machine. It enables us to develop interactive and highly sophisticated 2D and 3D graphic application. Most of the OpenGL code will be able to run in Mesa.

11.5 Ventana Vehicle control C/C++ Objects

11.5.1 Objects and Classes

Below is a short description of classes/interfaces contained in the Ventana vehicle control system: (refer to Fig. 1.1)

Classes:

Alarm class

This class contains the database for alarm that occur during run time. If the user choose to clear the alarm list from the GUI, then all the alarm history will be deleted from the Alarm database.

Configuration class

This class will contain the member functions that will read / write the configuration file, parse the file and put each element into its class / database

Depth class

This class will contain the information for depth, heading, pitch and roll and history values needed by GUI.

Graphic class

This class will contain the information database for the compass, lights, and vehicle graphic (eg. position on both screen)

IO_element class

Each of these objects is an abstract representation of the physical io_element. Meaning that each of them will map an input to an output either from Ventana vehicle control system to submarine.

IO_element_control class

This class creates IO_element objects dynamically as needed. It uses the STL vector to enable growing/shrinking of the database.

IO_poll class

IO_poll is responsible for the polling of the vehicle control system digital/analog channels. It first starts the analog conversion, polls the digital channels and then sends the analog channels. Thus guarantees that the digital channels have a very fast response.

Telemetry class

This class, basically, will do most of the Communication with the bottom control system, sending and receiving values such as heading, depth, pitch, roll, etc.

Telemetry_string class

The class is dedicated to send and receive string, such as Logging, Vision, TMS, and HDTV string.interfaces

Analog_interface

This directory contains the interface (c file) for the rest of the system to read in voltage from the AIO card from the Ventana control vehicle. There also contains some test programs in which one can compile and output voltage to the AIO card in order to test to see if the card is working properly. The readVoltage() function will depend on first calling the startConversion() function to start the analog to digital conversion on all 16 channels. Please refer to the PCAIO16 documentation provided for more details.

Digital_interface

InitializePort() is originally called when the ioElementControl object is created. This will set all the digital channels to be inputs. The readDigitalChannel() will then be called from the system to read individual channels based on their group, port and channel number.

Note that there are a possible of 3 groups of 24 channels for a total of 72 digital channels. On each group the ports are divided into A, B, and C. Please refer to the PCDIO72 documentation provided for details.

VideoOverlay

The class will set up the overlay display using UNIX ncurses. It is basically a standalone program that has its own internal database and receives the information through a FIFO (named piped). The system first sets up the database through the GUI by sending a predetermined IPC_Message. This message is then parsed and is either of two types of functionality, the ones that setup the database are as follows:

make_O_item(), delete_o_item(),annotate(), anddeannotate()
The make_O_item and delete_O_item will either add or delete an videoOverlay output item in its database respectively.

```
typedef struct outputItem {
        struct   outputItem *next;
        int      type;
        int      channel;
        int      line;
        int      col;
        int      color;
        float    value;
} oNode;
```

11.5.2 Messages

The other type of messages are the analog channels that come from the sub. Every message that is received by the Ventana vehicle control system is also written to the FIFO which is received by the Video Overlay program, which will either discard the message if it is not one that already exist in its database (user has not chose to display it from the GUI) or it will be displayed on the appropriate line/col. Note that the line/col will be different depending on the monitor size + resolution and text size being used. Thus a little playing around is required. When the Video Overlay is first started, it will output four points (1, 2, 3, 4) in places where it thinks the four boundary display corners are, if these cannot being seen, it is necessary to change either the font size/resolution, or by changing the following macro text replacement at the top of the file.

```
#define LEFTBOUND    4
#define RIGHTBOUND 50
#define TOPBOUND     0
#define BOTBOUND     23
```

Then rebuild the program by typing

```
> gcc videoOverlay.c -o videoOverlay –lcurses
```

in the Xterminal.

11.6 Submarine Software subsystem:

11.6.1 RTLinux

RTLinux is a Real Time extension of Linux. This patented idea is to run Linux as a lowest priority process while introducing a Real Time Hardware abstraction layer which handles the interrupts while providing hard or soft real time constraints. Programmers must use the POSIX threads extensively along with the real time constraints for a concurrent real time programming system. Note that Linux threads are actually treated as processes while sharing memory using System V shared memory mechanism. The reasoning behind this is that on SMP systems, each thread will have its own context of execution and can take advantage of multiple processor. In our system, threads are separated into two types, real time threads and normal linux threads. While receiving the strings from the top and digital/analog IO are done in real time, the long calculation of arms and autopilots is done as normal linux threads. This allows the more lengthy tasks to be done in the background while servicing the shorter tasks more quickly.

11.6.2 Flash ROM

The PICO FLASH ARRAY installation is provided by GESPAC which is provided in the next page. The ROM driver is provided in DOS and allows the user to format the ROM as just another C:/ drive. This is the most simple way and is the way we have chose due to its simplicity. The file system and the kernel can thus be directly copied on to the ROM via a floppy or another hard drive. The kernel can be automated simply by the means of syslinux boot loader, and scripted in the autoexec.bat file. The kernel will decompress the file system named fs.gz which is of type cramfs. Note that for faster execution, shadowing is enabled and thus is copied into RAM before executing.

11.6.3 File system

NOTE: FOLLOWING PARAGRAPH IS FOR INFORMATIVE PURPOSES, IT CAN BE SKIPPED.
The creation of the file system can be quite a long process, and is not trivial as trial and error is needed to figure out which essential files are needed. Busybox is used to provide the essentials of the UNIX utilities which can be useful in maintenance and diagonistic purposes (i.e. be able to execute the subIO standalone driver without running the whole telemetry system). Linux requires that the essential directories are there. These include:

/bin	/tmp	/sbin
/dev	/var	/mnt
/opt	/usr	/lib
/root	/proc	/etc

Some of these directories are simply empty directories, while others contain essential maintenance programs and programs that Linux expects to find. Perhaps the most important piece is the glibc library that is required both to start the init process and to execute our application. Our user application is located in the /sbin directory. Note that we have removed the need to login and thus different user and password protection is not possible. For simplicity, we have provided a file system with the essential programs. Note that one must be careful about removing directories or programs since Linux may not boot anymore if an essential program is removed. As a handy heuristic, you probably should not remove anything from the /etc and /lib directories unless you are sure of what you are doing. The /bin, /usr and /sbin essentially contains useful utilities and user applications. If you are sure you do not need such a tool, it can be removed. The /lib directory contains the two essentially library and should not be touched. The /mnt contains the information in order to mount certain devices such as the floppy. The /dev directory contains the essential devices, of course you can add more device files as needed to this directory. Please review and refer to the BUILDING LINUX FROM SCRATCH HOWTO in the cdrom labeled VENTANA DOCUMENTATION before making any changes.

11.6.4 The Custom Root Filesystem For Ventana Sub

Instead of building the filesystem from scratch which is a very time consuming task. We will use the root filesystem supplied by DTEC. This filesystem is a very customized and stripped down version. The goal was to build it using the smallest libraries and only the minimal/essential tools.
CAUTION!: Because the supplied root filesystem is customized strictly for the Ventana software, if you were to place normally compiled programs on to this filesystem, it may not work. This filesystem is meant for only the Ventana software and thus only contains the libraries needed for it. Furthermore, we should NOT be placing other software into the boot image.

Insert the disk labeled "Ventana Subsea source code".

```
mount /mnt/floppy                    // don't worry about modprobe error, we probably
cd /usr/rtlinux/rtlinux-3.1/ventana                    // didn't unable the ability in rtlinux (we shouldn't)
cp /mnt/floppy/vent_root_fs.tgz .
tar -zxvf vent_root_fs.tgz
cd vent_root_fs
```

You should now see the following directory

ls

/bin	/mnt	/tmp
/dev	/proc	/usr
/etc	/root	/var
/lib	/sbin	

NOTE: For a full description on how each directory and its functionality please consult one of the Linux System administration books. My favorite is "Linux: installation, configuration, use" by Michael Kofler. There is a 2nd edition out, however, an old edition will suffice if we are strictly in terminal mode and do not need the graphical capabilities.

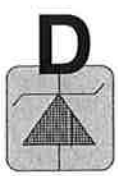
Copy our kernel modules into this directory. We will assume that the kernel modules for Ventana have already been compiled.

If it has not been compiled:

```
cd /usr/src/rtlinux/rtlinux-3.1/ventana
make
cd /home/ventana/vent_root_fs
```

Copy our programs. The following lines are in the script file burn. To run burn instead of typing all this in manually type ***./burn***

```
cd lib/modules
objcopy --strip-debug /usr/src/rtlinux/rtlinux-3.1/ventana/telemetryCom.o telemetryCom.o
```



Create a zero device.

cd /usr/src/rtlinux/rtlinux-3.1/ventana/vent_root_fs
dd if=/dev/zero of=./fs bs=1k count=1220
mke2fs -m 0 -N 1220 ./fs // zero == 0
type y // for answer to ./fs is not a block serial device. Burn shows this as well.

Note: the following is in the script file burn2 to run burn2 instead of typing all this in manually type ***./burn2***

mount -o loop -t ext2 ./fs /mnt // o == letter o, spaces; ext2 (space) ./fs (space) /mnt
ls /mnt // should see lost+found
rm -r -f /mnt/lost+found // removes lost and found
cp -dpR ./bin /mnt // use up arrow and just change dir
cp -dpR ./dev /mnt
cp -dpR ./etc /mnt
cp -dpR ./lib /mnt
cp -dpR ./mnt /mnt
cp -dpR ./proc /mnt
cp -dpR ./root /mnt
cp -dpR ./sbin /mnt
cp -dpR ./tmp /mnt
cp -dpR ./usr /mnt
cp -dpR ./var /mnt
ls /mnt // we should see all the directories
df // shows the amount of space used
umount /mnt
gzip -9 ./fs //answer y to overwrite the existing file (if there is one)

fs.gz is our root filesystem with our program on it. rtzImage + fs.gz is essentially our whole linux image. The rtzImage should not change once it is on the EEPROM. Our changes in the source code is reflected by the fs.gz, and this file can be transferred down using the download software Command.

Copy fs.gz onto a blank floppy and transfer it down either using the download new software or hooking up a monitor and keyboard and go through a manual install.

mount /mnt/floppy // use a blank disk
cp fs.gz /mnt/floppy //copy the new image to the floppy
cp /boot/rtzImage /mnt/floppy/bzimage // the bottom
umount /mnt/floppy

NOTE: to format the floppy disk use format /dev/fd0

Manual Download from the hard drive:

mount –t msdos /dev/hda6 /mnt/DOS //3 errors are OK
cp fs.gz /mnt/DOS //overwrite yes
umount /mnt/DOS
shutdown –r now
CNTR-ALT-ESC to get into the G96 BIOS
Use the G96 Bios to change the mode to the EPROM not the hard drive
F10 to Save //DOS should boot from the EEPROM
When DOS starts Cnt-C to terminate the batch file
Copy FS.GZ to C: from either A: (if floppy) or D: (if Hard Drive) //DOS command
Restart the computer (CNTR-ALT-DEL)

11.6.5 Downloading New Software

We will assume that you have a new fs.gz file available. This file can either come from DTEC or a new compiled and compressed file.

The fs.gz file is the only software that can be downloaded through the Ventana download option. This is done so that our bootstrapping + downloading software can reside safely if there is a failure. The fs.gz can just be retransmitted.

Place the fs.gz into the download_software directory

mount /mnt/floppy
cd /home/ventana/source/download_software
ls // we should see fs.gz and fs.gz.bak and other images

NOTE: fs.gz.bak is our read-only file that is the working version of sub-software

mv fs.gz fs.gz.MMDDYYYY // where MMDDYYYY is month/day/year eg. fs.gz.05222002

NOTE: you can use any extension you'd like to backup the fs.gz. We find this convention very useful to figure out changes made and good/bad versions.

cp /mnt/floppy/fs.gz /home/ventana/source/download_software
exit // exits the current xterm

Start the Ventana software (goat or any other way).
Power up the sub.
Wait for the telemetry link to establish (telemetry status light turns green).
Select File --> Download New Software. Minicom will start, the control software will close and downloading will start
Close minicom by clicking x on the xterm.
Restart Ventana
Wait for the telemetry link to go green.

WARNING!!! If the fs.gz image is not good or there is a failure to start the sub. Follow these instructions.

cd /home/ventana/source/download_software
cp fs.gz.bak fs.gz // we will replace fs.gzmake sure sub power is down

- make sure minicom and ventana is closed
- **start ventana software**
- **select file--> download new software**
- wait until it says uploading fs.gz
- power up the sub

If a failure is due to downloading a bad image, then we need only to replace the image. However, if there is a EEPROM failure or we need to program a new EEPROM from scratch, then please see the "Programming sub EEPROM" from scratch.

11.7 Surface and Subsea Communication:

Serial port Communication (ASCII string) The Communication between the sub and the control system is at a Baudrate of 115kbps. The strings are in ASCII and thus eliminate the worries of byte ordering and word size. The throughput is quite good and we have elected to send the digitals on a change and gets priority over the analog values. The vehicle control uses standard Red Hat Linux 7.0 with kernel 2.2.16. The standard linux distribution uses the termcap to handle the serial Communications. Termcap is also used to Communicate with tty and printers. The basic setup is to fill in the termios structure, which tells Linux how to process the characters, the speed, and other terminal options, then set up the serial device with the termcap. The serial driver then handles the Communication appropriately.

11.8 References:

Stevens, W.R. 1993. Chapter 11 – Terminal IO, Advanced Programming in the UNIX Environment. Prentice-Hall, Englewood Cliffs, N.J.

Linux serial HOWTO (provided in the Linux HOWTO directory of the CD)
Real Time Linux website http://www.rtlinux.org/
Termios man page
Linux BootDisk HOWTO (provided in the Linux HOWTO directory of the CD)
http://togl.sourceforge.net
http://www.opengl.org
http://www.mesa3d.org

Appendix A Board Jumper Selections

Telemetry Board Switch Configurations:

A '1' designates ON and a '0' designates OFF.

Surface Computer I/O Configuration

Board	IRQ	DMA	Base Address	Switch Settings	Settings
Com 1	4	none	0x3F8	none	
Com 2	3	None	0x2F8	none	
Parallel 1	7	none	0x378	none	
Rocket port			Plug'n'Play	none	
PIO 96	none	none	Plug'n'Play	none	
AIO-16P-A	none	none	0x300	111100	Single ended, gain 2, no counter, DMA na
AIO-16P-B	none	none	0x310	011100	Single ended, gain 2, no counter, DMA na

Subsea I/O Configuration

Board	IRQ	DMA	Base Address	Address A2-A9	Settings
GESDAC – 1	none	none	0xF840	XX00 1111	J1(all+15),J4-J11(1+2)(3+4)
GESDAC – 2	none	none	0xF860	XX10 1111	J1(all+15),J4-J11(1+2)(3+4)
GESADC – 1	none	none	0xF8A8	0101 0111	J1(2+3)(4+5)(6+7)(8+9), J2(1+2), J3(1+2), J4(3-4), J5(4+3)(1-2), J6-J10(none), J11(1+4)(2+3), J12-13(none)
GESADC – 2	none	none	0xF8B0	1001 0111	J1(2+3)(4+5)(6+7)(8+9), J2(1+2), J3(1+2), J4(1-2), J5(4+3)(1-2), J6-J10(none), J11(1+4)(2+3), J12-13(none)
GESADC – 3	none	none	0xF8B8	0001 0111	J1(2+3)(4+5)(6+7)(8+9), J2(1+2), J3(1+2), J4(3-4), J5(4+3)(1-2), J6-J10(none), J11(1+4)(2+3), J12-13(none)
GESADC – 4	none	none	0xF8C0	1110 0111	J1(2+3)(4+5)(6+7)(8+9), J2(1+2), J3(1+2), J4(1-2), J5(4+3)(1-2), J6-J10(none), J11(1+4)(2+3), J12-13(none)
GESOUT – 1	none	none	0xF800	1111 1111	J1(1-2)(3+4), J2(1+2)(3+4), J3(1+2)(3+4)
GESOUT – 2	none	none	0xF808	0111 1111	J1(1-2)(3+4), J2(1+2)(3+4), J3(1+2)(3+4)
GESINP – 1	IRQ4	none	0xF820	1101 1111	J1(none) G96 IRQ4 is SBS47 IRQ5
GESMPL – 1	none	none	0xF8D0	0101 1000	
GESMPL – 2	none	none	0xF8D8	1101 1000	
GESMPL – 2	none	none	0xF8D8	1101 1000	
GESSIO – 1	IRQ5	none	0xF100	1101 1xxx	J9(all on), J11(11011), J10(on), IRQ5(G96) is IRQ15 (SBS-47)

The following coding is for the jumpers on the boards. J1(1-2)(3+4) is for J1 and has not got a jumper between pin 1 and pin 2 but has a jumper between 3 and 4. Note that the Jumpers for the address selection for the GESPAC boards are inverted. The shorting of an address line makes that address line 0 or off. The address selection for the MPL board is not inverted. The shorting of an address line makes that address line 1 or on.
The address selection is all 8 bit asynchronus. All the I/O boards use F8xx addresses except the GESSIO14 which uses F0xx. Actually these addresses are the same with the G96 bus architecture. That's the reason the GESSIO is on F100. This is the next available address after the last MPL board and is the same as F900.

Appendix B Plug and Play Address

The PCI bus board in the surface computer are of two varieties. There is an analog I/O board that is ISA bus architecture. This board has it's address selected by the jumper and switch selections on the board. It's base address is 0x300 and must match the software which is expecting it to be at address 0x300. This is the easier of the two IO board to configure. The second I/O board (digital) is a PCI bus architecture board and has it's address assigned to it by the BIOS Plug and Play software. This address can in effect be anything that the BIOS decides that it should be. With Ventana's present hardware configuration the Plug and Play software has assigned 0xE000 as the address for the Digital I/O board. However if there are PCI bus hardware additions or removals the BIOS will probably change the address it assigns to the PIO-96 digital I/O board. This includes the removal of the Ethernet, and VGA boards. If hardware changes are made it will be necessary to probe the PCI bus to get its new assignment. There is a program that runs under DOS or Windows called PCI Find from International Computer Source that reports the addresses of ICS PCI bus I/O boards. Unfortunately this program is not Linux compatible and could not be used to probe the ports upon Linux boot.

PCI find is on a DOS bootable disk supplied By DTEC and if installed into the floppy drive on power up will allow the user to find the PCI address of the PCI slot boards. If the address is not 0xD400 for the PIO96 the software will have to be changed and recompiled to reflect the change.

Appendix C Supplied Software

The following software is shipped with the system:

On CD ROM:

The following CDs are supplied with the system.

Redhat Linux Version 7.1

Redhat Linux Version 7.1

Redhat Linux CD:

Redhat Linux Version 7.0.

TK/TCL installation CD:

Contains all of the programs used by the TK/Tcl environment.

DTEC Ltd. Manual and Drawings CD:

This document in Word for Windows format.
DXF format Drawing of the wiring of the surface system.
DXF format Drawing of the wiring of the subsea system.
DXF format Drawings of all of the Schematics of Termination boards used in the sub.
DXF format Drawings of all of the General Arrangements of the Termination boards.
TCL source code, Surface C++ source code, and Subsea C source code.

On Floppy:

The following Floppies are supplied with the system.

Matrox Graphics Driver:

This floppy contains the files required to install the 4 head graphics board. See the Appendix on the Matrox multi head graphics board installation. The file mga-1_2_0beta.tgz contains all the files need to install the multihead display adaptor.

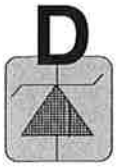
Rocket Port :

The drivers for installing the rocket port serial port board into Linux. The file 1800024D.tgz contains the driver for the rocket port installation.

Xtouch touch screen:

The files for the installing of the touch screen are as follows:
LICENCE.tsi
elo.config
xtouch
There are 2 directories one for each of the displays. These directories have the same name files but the files are different. In other words each directory has files to support xtouch but the files have been modified for the second screen in the second directory. Xtouch is run twice once from the first directory and the second time from the second directory.

Surface Source Code:



This is a duplicate floppy(s) that contain the surface and subsea source code.

Subsea Source Code:

These are a pair of floppies that contain the bootable code for the sub.

TKFrame:

This file is needed for TK/Tcl to support 2 displays.

edited vi tkInt8.0.h
line 36 commented out
UNDO to get back
to LAST Chance Config.

This is a duplicate floppy(s) that contain the surface and subsea source code.

Subsea Source Code:

These are a pair of floppies that contain the bootable code for the sub.

TKFrame:

This file is needed for TK/Tcl to support 2 displays.

604-469-3392 Jon 339

LATEST

c

edited vi tkInt8.0.h

line 36 commented out

UNDO to get Back

To LAST Change Config.

Appendix D Surface Software Installation

Redhat Linux 7.0 Installation

When installing a new unformatted hard drive the CMOS setup has to be configured to boot from the CDROM drive. When the computer first is turned on press the DEL key to enter the CMOS setup and change the boot sequence to CDROM first. Install the Redhat CD 7.0 Disk #1 into the CDROM and restart the computer. The Redhat disk will take you through the installation process. We have found that certain hard drives require that a partition is needed before the computer can be booted from the CDROM. Any DOS system disk can be used to place a DOS partition onto the hard drive. After this partition is installed the system is able to boot from the CDROM and Linux will format and partition the drive as required.
Choice Selections:

- Custom Install
- US English
- 2 Button PS2 Mouse
- Generic 101 Keyboard
- Partition with Disk Druid
  - Change mount point of hda2 to /
  - Format /dev/hda2 /
- Uncheck or check backup disk, all others default.
- No Network
- Select Time Zone
- Root Password: ventana
- Account Name: ventana
- Account Password: window
- Full Name: Palmer
- Add
- All Default for Authentication Protocol

/boot hda1 16 M
/ hda5 12089 M
<swap> hda 6 64 M.

Notes: If using fdisk instead of disk druid to partition the disk. Swap partition is usually twice the RAM or 128Meg. Remove all existing partitions and create the swap partition and another big one that uses the rest of the hard drive.
Give the partitions a HEX ID. The swap one needs to be 82 and the large one needs to be 83 which is the default.
Write the information to disk and exit. "W".

☐ Check "Select individual packages"

Required Software Modules

The following modules are required by the Ventana software. These modules are included in the Redhat 7.0 CD and are described when installing. The installation program will check for all the modules dependencies again at the end of installation process. Go through all the directories to make sure that the following are checked.

☐ Applications, Look under To See following files.

Application Modules

Archiving:

- Dump
- RMT
- Sharutils
- Unzip
- Zip

Communication Modules:

- Dip
- Minicom
- Pilot-link
- lrzsz: Upload and Download software.

Editor Modules:

- emacs
- emacs –X11
- emacs – nox : with or without Xwin support
- gedit
- gnotepad
- vim –X11
- vim - enhanced

Engineering:
- gnuplot

Internet Modules:
- finger
- ftp
- gftp
- kde network
- ncftp -> FTP client
- netscape Common
- netscape Communicator
- netscape navigator
- openssh
- openssh askpass
- openssh gnome
- openssh clients
- pine
- plugger
- rsh
- rsync
- talk
- telnet
- stunnel
- traceroute
- whois

Multimedia
- desktop backgrounds
- dia : like Visio, draw class diagram, etc.
- ee : electric eyes
- gnome audio
- kde graphics
- sndconfig
- xsane : scanner, digital camera

Productivity
- ical

Publishing
- Ghostscript
- Ghostscript fonts
- Gv
- Xpdf

System
- Arpwatch : network monitoring tool
- Anaconda : RH linux installer
- Anaconda runtime
- Bind – utils
- Control panel
- Dialogs : create tty dialog boxes
- Dosftools
- Gnome – utils
- Gnorpm
- Gtop : show processes, memory use, etc
- Iproute
- Kernelcfg
- Modemtool
- Mtool
- Netcfg
- Rp3
- rdate
- Screen
- Statserial : debug serial port / modem prob
- Sudo
- Timetool : to set time / date in user interface
- Usermod :

Text
- Indent

Development
Debugger
- Gdb
- Memprof
- Xxgdb : debugger with user interface

Languages
- Cpp
- Dev86
- Gcc
- Gcc – c++
- Gcc – objc
- Gnome – objc
- Kgcc

Libraries
- Mesa-devel
- XFree86devel
- Compat – glibc
- Gtk+ - devel
- Kudzu – devel
- Libgtop - devel
- Libstdc++ - devel
- Linuxconf – devel
- Ncurses – devel
- Openssl – devel
- PCI utils – devel
- Tcllib

System
- Kernel header
- Kernel-source

Tool
- Binutils
- Bison
- CVS
- Diffstat
- Flex
- Libtool
- Make
- Njamd : trace memory leaks

- Patch
- Pmake
- Rcs
- Rpm build

Documentation

- Man pages
- Gnome users guide
- Kernal-doc

System Environment

Base

- chkfontpath
- Gnome – print
- IPCchain : set up firewall / ip masquerading
- IPtables
- Yptool :

Daemon

- LPRng
- XFree86xfs
- dhcp
- IPutils : ping
- Nfsutils
- Openssh server
- Pidentd
- ppp
- Portmap : prevent NIS(YP) theft
- RP – pppoe : over Ethernet
- Rsh – server
- Rusers : check other user on various machines
- Rwho
- TCPwrappers : Samba
- telnet – server
- wvdial
- ypbind

Kernel

- kernel boot

Libraries

- Mesa
- SDL
- Xfree86 libs
- Freetype
- Compat – libstdc++

- Gnomelibs
- Gtk+
- Imlib
- Imlib cfgeditors
- Kdelibs
- Libgtop
- Libjpeg
- Libpng
- Libungif
- Libtool – libs
- Libtiff
- Ncurses4

Shell

- Mc : midnight Comander (ftp, view tar, etc)
- Tcsh

User Interface

- ~~Desktop~~ DESKTOP
- Control center
- Gnome – core
- Kdebase
- Switchdesk
- Sawfish
- Switchdesk – gnome
- Switchdesk – kde
- Wmconfig

X ~~[illegible]~~

- XFree86
- XFree~~75 dpi fonts~~ xfree86-75dpi-fonts
- XFree86 – tool : xcalc
- XFree86 – xf86cfg
- Urw – fonts
- Xinitrc

X Support

- XFree86 – mach64
- XConfigurator

NEXT and Then . . .
Check
Install packages to satisfy dependencies.
Use unprobed monitor
Select Matrox Millenium G200
Installation will proceed. The install takes about 1 hour. You will need disk #2 of Redhat 7.0.

Rocket Port Installation Procedure

Read the instructions found in the HW-INSTALL file or the Hardware Reference Card shipped with the board to install the RocketPort/RocketModem into your computer.

After the board is installed:

Copy the 1800024D.tgz file to /usr/src. Put the DTEC floppy labeled Rocket Port Device drivers into the floppy.

mount /mnt/floppy
cd /mnt/floppy
cp /mnt/floppy/1800024D.tgz /usr/src

Tar the file using GNU tar:

① ***cd /usr/src***
② ***tar xzvf 1800024D.tgz***
~~***tar xzvf comtrol-1.23.tar.gz***~~
~~cd comtrol~~
③ ***umount /mnt/floppy***
THIS MUST BE WRONG

Remove the floppy form the drive

3. A subdirectory of /usr/src now exists called comtrol. This contains the RocketPort/RocketModem driver and associated files.

4. Recompile the RocketPort/RocketModem driver. (This is not always necessary, depending on the version of your kernel, but it is safe to do):

cd comtrol
make clean
make

5. Install the RocketPort/RocketModem driver as root:

make install

6. If you are using more than one ISA RocketPort/RocketModem board, or have installed your ISA RocketPort/RocketModem card at a non-standard address, you will need to set up the file /etc/rocketport.conf. /usr/src/comtrol contains an example rocketport.conf file. Copy it to the /etc directory and edit it according to the Comments contained in that file. This step is NOT necessary if you are using only PCI RocketPorts.

7. Optionally, edit your /etc/rc.d/rc.S file (or other appropriate boot-up script) so that the "rc.rocket" script is run automatically each time your system boots. This is done for you automatically if you are using a system with System V init files, such as is used in the RedHat or Debian releases.

8. Your RocketPort/RocketModem driver has now been installed in your system. Either reboot to load the driver into the currently running system, or manually load the driver running the rc.rocket script with the argument "start": For Ventana this is done for you in the script files so you don't need to do it.

./rc.rocket start

9. Configure your applications and/or your getty scripts as appropriate for your application. The RocketModem looks to Linux like a RocketPort card with modems attached; you may refer to the RocketModem manual for the AT Commands.

Matrox Graphics Setup

BETA DRIVER FOR XFree86
Version "1.2.0"

Following is a procedure from Matrox on how to configure Linux for the Multi Head VGA Board. The easiest thing to do is to use the copy of the xf86config that is supplied by DTEC. This file should be copied into the directory /etc/X11/. It is a good idea to make a backup of the existing xf86config file first if one exists from a previous installation. DTEC generated the supplied xf86config file with this procedure.

Another option would be to generate your own xf86config file using the following procedure and then editing it to be similar to the one supplied by DTEC. The procedure re-compiles the entire VGA driver and takes about 2 hours to complete.

Description Of This Release

This XFree86 4.0.2/4.0.3 beta driver introduces open source DualHead support for the G450 graphics adapter. The G400 and G200 series are also supported through use of the Matrox HAL library, which can be optionally installed to provide extra features that are not included with the standard XFree86 mga driver. These features include DualHead, digital flat panel, and TVout support for the G400, and multiple display support for the G200 Multi-Monitor series. Please take note that there have been no changes made to the HAL library since the previous driver release, therefore G400 and G200 users will not see any performance improvements in this release. This driver is also programmed to set the internal clock speeds to production levels, increasing 2D and 3D performance under XFree86

The driver can either be installed by copying the binary file(s) into a working installation of XFree86 4.0.2 or 4.0.3, or by compiling the modules from source. Ventana has Version 4.0.2.

The driver is for x86 based architectures only. Support for other architectures is planned for a future release.

Please take note that this is a beta driver and is not supported by Matrox.

Notes

The installation instructions in this document are based on Red Hat Linux 7.0. The directory locations and procedures listed may differ slightly when working with other distributions.

This driver is based on the open source development being done within the XFree86 Project. The driver remains unmodified, with the exception of the HAL module, which is a closed source library used to enable features such as DualHead, TV out, and support for digital flat panel monitors.

Due to certain legal liabilities and for the protection of intellectual property, Matrox reserves licensing rights to the library and prohibits reverse engineering but allows free distribution under any operating system. Matrox encourages members of the open source Community to freely distribute and assist in the further development of this driver.

Installation

A working installation of XFree86 4.0.2 or 4.0.3 is required before the binaries can be installed. If your Linux distribution uses an older version of XFree86 you will either need to upgrade your X server, or your Linux distribution.

XFree86 4.0.2 and 4.0.3 source files and installation instructions are available from: http://www.xfree86.org/, or from one of their mirror sites: http://www.xfree86.org/#mirrors. .

For Ventana we have RedHat 7.1 CD's to use. Install Redhat 7.1 Source CD and change directory to SRPMS

mount /mnt/cdrom
cd /mnt/cdrom/SRPMS
rpm –i Xfree86-4.0.3-5.src.rpm ← need to use Capital "F" in XFree86
F

Tar the file it created in /usr/src/redhat.

cd /usr/src/redhat/SOURCES
cp Xfree86-4.0.3.tar.bz2 /
cd / ← F ← F
tar –Ixvf Xfree86-4.0.3.tar.bz2
Where I is capital i.

Once the new version of XFree86 is installed and working properly, the G450 open source driver can be installed by copying mga_drv.o into /usr/X11R6/lib/modules/drivers (or wherever your X11R6 directory is located.) This file is located on the "Matrox Software Disk".

G400 and G200 MMS users will need to copy both files (mga_drv.o and mga_hal_drv.o) into the drivers directory in order to enable Dual Head or multiple monitor support. The HAL module can also be used with the G450. If the mga_hal_drv.o file is present in the drivers directory the HAL module will be used instead of the open source driver.

It is recommended to back up the existing mga_drv.o file before installing the new one. The mv Command can be used to back up the mga_drv.o file as shown below.

cd /usr/X11R6/lib/modules/drivers
mv mga_drv.o mga_drv.o_old

Installation From Source

To install the driver from source you will need the XFree86 4.0.2 or 4.0.3 source files, which are available for download from the sites listed above. Once the XFree86 source files have been extracted, the existing mga Directory should be backed up and renamed. This will be replaced by the new mga directory, which will be extracted from the driver source file. Use the following Command to make a backup of the mga directory:

cd /xc/programs/Xserver/hw/xfree86/drivers
mv mga mga_old

Now move or copy the mga driver source file into the drivers directory and extract the files by using the following Command: , INSERT THE MATROX FLOPPY TO THE floppy drive.

mount /mnt/floppy
cp mga-1_2_0beta.tgz /xc/programs/Xserver/hw/xfree86/drivers
cd /xc/programs/Xserver/hw/xfree86/drivers
tar xvzf mga_filename.tgz ← tar xvzf mga-1_2_0beta.tgz
umount /mnt/floppy

Use the Hal library instead of the G450 open source driver. You will need to create and edit a host.def file from within the XFree86 source tree:

cd /xc/config/cf
cp xf86site.def host.def

Using a text editor such as pico, vi, or emacs, uncomment the HaveMatroxHal line in the host.def file by removing the "*/" Comment from the line after "HaveMatroxHal" and placing it at the end of the previous line. Here is an example of how this section should appear once it has been edited:

```
/*
    * If you have the HALlib.a binary installed in
    * xfree86/drivers/mga/HALlib,
    * uncomment this:
*/
#define HaveMatroxHal        YES
```

Note: The option #define UseMatroxHal is also required, but should already be enabled by default in the host.def file. IT WAS NOT, so I uncommented it and changed NO to YES.
If you have a G450 and have chosen to compile the Hal module, it can always be disabled later by removing the ga_hal_drv.o file from the usr/X11R6/lib/modules/drivers directory. This will automatically enable use of the G450 open source driver.

To compile and install XFree86 4.0.2 go back to the /xc directory and enter the following Commands:

make World (this will take awhile) this has taken about 1.5 hours to do on Ventana's Computers.
make install This takes about 5 minutes on Ventana's Computers.

Now that the source has been compiled and installed, the new XFree86 binary may need to be linked to X:

ls -l X This should already be OK on Ventana.

This will show which binary X is linked to. If it lists anything other than XFree86, you can link it to 4.0.x by using the following Command:

ln -sf /usr/X11R6/bin/XFree86 ./X

If you are upgrading from XFree86 3.3.6 or older, a new XF86Config file needs to be created. One has been provided on disk by DTEC. Copy the XF86Config file from disk. Insert the Matrox disk into the floppy.

mount /mnt/floppy
cp XF86Config-4 /usr/X11R6/bin
umount /mnt/floppy

Optionally you can generate a new config file using xf86Config.

cd /usr/X11R6/bin
./xf86config

The xf86config Command will start a text based configuration program. The values you specify here will determine the parameters for your mouse, keyboard, monitor, and graphics card. This will create an XF86Config file in /etc/X11, which will need to be edited later. If newer boards such as the G450 are not listed when choosing a graphics device, it is safe to choose another adapter that uses the mga driver, such as the G200. Once the values you specified have been saved, you will need to edit the XF86Config file to include references to the newly supported features.

The answers to some of the questions for the monitors and so on are as follows:

PS2 Mouse.
101 Keyboard.
US English.

Horiz Sync Rate 31.5-48.5
Vert Sync Refresh 50-90
ID = "Optiquest"

Note: Some Linux distributions install more than one version of XFree86. If XFree86 4.0. or higher is installed on your system, you may already have a config file specific to that version. For example, if your Linux distribution has installed both XFree86 3.3.6 and 4.0.1, there is likely to be a separate config file for each in /etc/X11; XF86Config for 3.3.6, and XF86Config-4 for 4.0.1.

Direct Rendering Infrastructure

No idea where to find this?

The Direct Rendering Infrastructure, also known as DRI, enables the use of 3D hardware acceleration under Linux. DRI acceleration is not enabled by default in XFree86. However, most recent Linux distributions automatically enable DRI support through the kernel during setup. Direct rendering requires that your kernel version has support for agpgart. These newer distributions should also configure agpgart automatically, making it unnecessary to upgrade the kernel as suggested below.

You can find out if direct rendering is enabled by looking at the output created when starting X. You can create an output file by typing:

startx >& Xoutput.log | where does this file get saved?

or you can view the output in /var/log/XFRee86.0.log, which is a log file that gets created each time the X server is started. Look for the "direct rendering" line, which will tell you if DRI is enabled or disabled. If direct rendering is disabled, it may be possible that the module exists but is not loading. From a console or terminal, type:

from which directory?

insmod mga
lsmod | would not run

where?

The insmod Command will load the mga.o module if it exists and is located in the proper directory. The lsmod Command lists all of the modules that are currently loaded. This does not include modules that have been built into the kernel. If lsmod shows that the mga and agpgart modules are loaded, then the X output log file should state that direct rendering is enabled.

If you are using an older kernel that does not include agpgart support, it will be necessary to upgrade to a later version, such as 2.4.x. When configuring the kernel, be sure to enable agpgart (as a module) and select the proper chipset. Once the new kernel is installed and working properly, DRI can be enabled from within the following directory by compiling and installing the mga.o module:

cd /xc/programs/Xserver/hw/xfree86/os-support/linux/drm/kernel
make -f Makefile.linux mga.o
insmod agpgart
insmod mga.o
lsmod (to make sure the modules are loaded)

You may want to copy the mga.o module into the proper kernel modules directory; usually lib/module/your_kernel_version/misc/. This should allow you to load the module from anywhere by using the lsmod Command. Otherwise, you will have to return to the kernel directory each time you wish to reload the module. If there is already an older mga.o module in the misc directory it would be a good idea to make a backup before installing the new version.

The XF86Config file must now be configured to load DRI. These lines may already be present in the "Module" section of your XF86Config file.

```
Section "Module"
Load "glx"
Load "dri"
EndSection
```

Added / SAVED / ReBooted

Dave Said I didn't need to do

After starting X there should be a "direct rendering enabled" message in the X server startup log. Please note that dri will not be enabled if you are using a configuration which uses Xinerama. You may want to keep two XF86Config files on hand; one for dri and the other for DualHead.

Dualhead And Multimonitor

(Xf86 config) Not Need if Copy over

To add DualHead support for the G450/G400 and multiple display support for the G200 MMS, the following sections of the XF86Config file need to be modified:

Device section
Screen Sections
ServerLayout sections

Use the sample XF86Config files provided at the end of this document as an example of what needs to be added to your config file. Take note of the "BusID" lines. You will need to edit these to match the BusID output shown in your Xfree86 log file. You can create a log file by redirecting the X output to a text file:

startx >& Xoutput.log

The section of the output file which lists the BusID values will look similar to the following example. Search the output file for "scanpci" to find the PCI BusID's more quickly.

```
II) Loading /usr/X11R6/lib/modules/libscanpci.a
(II) Module scanpci: vendor="The XFree86 Project"
compiled for 4.0, module version = 0.1.0
(II) Unloading /usr/X11R6/lib/modules/libscanpci.a
(--) PCI:*(2:0:0) Matrox MGA G200 AGP rev 3, Mem @ 0xf0000000/24,
0xd2000000/14, 0xd0000000/23
```

Use the following Command to start the Xserver in multi display mode: startx -- +xinerama

Note: The Xinerama extension can be set up to load automatically by adding the following line to the ServerLayout section of the XF86Config file: Option "Xinerama"

Other Features

Matrox is one of the first graphics chip companies to offer TV out capabilities as well as support for digital flat panel monitors under XFree86. This driver provides preliminary support for these features, while improved performance and added features are planned for future releases.
Note: The TV out option is not supported by the G450. This feature is currently only supported by the G400.
To add digital flat panel and TV out support the following lines must be added to the device section of your XF86Config file (usually located in /etc/X11/)

```
Opt1on "TV" "yes"
Option "DigitalScreen" "yes"
```

The default TV standard and cable types are NTSC and composite. Support for PAL and other SCART cable types can be enabled by adding the following options to the device section:

```
Option "TVStandard" "PAL"
Option "CableType" "SCART_RGB"
Option "CableType" "SCART_COMPOSITE"
Option "CableType" "SCART_TYPE2"
```

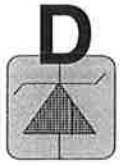
Both the TV out and digital flat panel features can be used either in clone mode or virtual mode by using the xinerama extension. In order for the TV out feature to work the display resolution must be set to 640x480 with a refresh rate of 60Hz for NTSC, or 50Hz for PAL.

The DigitalScreen option can be enabled if you have a G400 digital flat panel module. Digital flat panels generally require a vertical refresh of 50-60Hz. The "Tv" and/or "DigitalScreen" option must be added to the second device section of the XF86Config file. The example below is configured for clone mode, in which the desktop is cloned on the second screen. If you wish to span the desktop across two screens, the ServerLayout section must appear as it does in Sample 1. Do not use the "Tv" and the "Digital Screen" options together. You can disable one or the other by placing the # symbol in front of the option you do not wish to use, or by simply excluding that option from the config file.

The following examples are from Matrox. DTEC examples are supplied on floppy disk.

Example: Clone Mode

```
Section "Device"
   Identifier  "G400_1"
   Driver      "mga"
   BusID       "PCI:1:0:0"
   Screen      0
EndSection
Section "Device"
   Identifier  "G400_2"
   Driver      "mga"
   BusID       "PCI:1:0:0"
   Screen      1
   Option "Tv" "yes"
  #Option  "DigitalScreen "yes"
EndSection
Section "ServerLayout"
    Identifier    "Layout1"
    Screen        "Screen0"
    Screen        "Screen1"
    Option        "Xinerama"
    InputDevice   "Mouse0" "CorePointer"
    InputDevice   "Keyboard0" "CoreKeyboard"
EndSection
```



Sample XF86Config FILES

Sample 1: DualHead

```
# **********************************************************************
# Graphics device section
# **********************************************************************
Section "Device"
   Identifier  "G400_1"
   Driver      "mga"
   BusID       "PCI:1:0:0"
   Screen      0
EndSection
Section "Device"
   Identifier  "G400_2"
   Driver      "mga"
   BusID       "PCI:1:0:0"
   Screen      1
EndSection
   **********************************************************************
# Screen sections
# **********************************************************************
# Any number of screen sections may be present.  Each describes the
configuration of a single screen.  A single specific screen section may be
specified from the X server Command line with the "-screen" option.
Section "Screen"
   Identifier  "Screen 0"
   Device      "G400_1"
   Monitor     "Nokia"
   DefaultDepth 16
   Subsection "Display"
      Depth       8
      Modes  "640x480" "800x600" "1024x768" "1280x1024"
      ViewPort    0 0
   EndSubsection
   Subsection "Display"
      Depth       16
      Modes  "1024x768" "800x600" "640x480"
      ViewPort    0 0
   EndSubsection
   Subsection "Display"
      Depth       24
      Modes  "640x480" "800x600" "1024x768" "1280x1024"
      ViewPort    0 0
   EndSubsection
EndSection
Section "Screen"
   Identifier  "Screen 1"
   Device      "G400_2"
   Monitor     "Nokia"
   DefaultDepth 16
   Subsection "Display"
      Depth       8
      Modes  "640x480" "800x600" "1024x768" "1280x1024"
      ViewPort    0 0
   EndSubsection
   Subsection "Display"
      Depth       16
      Modes  "1024x768" "800x600" "640x480"
      ViewPort    0 0
   EndSubsection
   Subsection "Display"
      Depth       24
      Modes  "640x480" "800x600" "1024x768" "1280x1024"
      ViewPort    0 0
   EndSubsection

EndSection


# **********************************************************************
# ServerLayout sections.
# **********************************************************************
# Any number of ServerLayout sections may be present.  Each describes
the way multiple screens are organized.  A specific ServerLayout section
may be specified from the X server Command line with the "-layout"
option.  In the absence of this, the first section is used. When now
ServerLayout section is present, the first Screen section is used alone.
Section "ServerLayout"

# The Identifier line must be present
   Identifier  "Simple Layout"

# Each Screen line specifies a Screen section name, and optionally the
relative position of other screens.  The four names after primary screen
name are the screens to the top, bottom, left and right of the primary
screen.  In this example, screen 2 is located to the right of screen 1.

   Screen "Screen 0" LeftOf "Screen 1"
   Screen "Screen 1"

# Each InputDevice line specifies an InputDevice section name and
optionally some options to specify the way the device is to be used.
Those options include "CorePointer", "CoreKeyboard" and
"SendCoreEvents".

   InputDevice "Mouse1" "CorePointer"
   InputDevice "Keyboard1" "CoreKeyboard"

EndSection
```

Sample 2: G200 MultiMonitor

```
# **********************************************************************
# Graphics device section
# **********************************************************************
Section "Device"
   Identifier  "G200_1"
   Driver      "mga"
   BusID       "PCI:2:0:0"
   Option      "hw cursor" "off"
EndSection
```

```
Section "Device"
   Identifier  "G200_2"
   Driver      "mga"
   BusID       "PCI:2:4:0"
   Option      "hw cursor" "off"
EndSection
Section "Device"
   Identifier  "G200_3"
   Driver      "mga"
   BusID       "PCI:2:8:0"
   Option      "hw cursor" "off"
EndSection
Section "Device"
   Identifier  "G200_4"
   Driver      "mga"
   BusID       "PCI:2:12:0"
   Option      "hw cursor" "off"
EndSection

# **********************************************************
# Screen sections
# **********************************************************
# Any number of screen sections may be present.  Each describes the
configuration of a single screen.  A single specific screen section may be
specified from the X server Comand line with the "-screen" option.
Section "Screen"
   Identifier  "Screen 1"
   Device      "G200_1"
   Monitor     "NOKIA"
   DefaultDepth 16
   Subsection "Display"
      Depth       8
      Modes  "640x480" "800x600" "1024x768" "1280x1024"
      ViewPort    0 0
   EndSubsection
   Subsection "Display"
      Depth       16
      Modes       "1024x768"
      ViewPort    0 0
   EndSubsection
   Subsection "Display"
      Depth       24
      Modes  "640x480" "800x600" "1024x768" "1280x1024"
      ViewPort    0 0
   EndSubsection
EndSection
Section "Screen"
   Identifier  "Screen 2"
   Device      "G200_2"
   Monitor     "NOKIA"
   DefaultDepth 16
   Subsection "Display"
      Depth       8
      Modes   "640x480" "800x600" "1024x768" "1280x1024"
      ViewPort    0 0
   EndSubsection
   Subsection "Display"
      Depth       16
      Modes       "1024x768"
      ViewPort    0 0
   EndSubsection
   Subsection "Display"
      Depth       24
      Modes   "640x480" "800x600" "1024x768" "1280x1024"
      ViewPort    0 0
   EndSubsection
EndSection
Section "Screen"
   Identifier  "Screen 3"
   Device      "G200_3"
   Monitor     "NOKIA"
   DefaultDepth 16
   Subsection "Display"
      Depth       8
      Modes   "640x480" "800x600" "1024x768" "1280x1024"
      ViewPort    0 0
   EndSubsection
   Subsection "Display"
      Depth       16
      Modes       "1024x768"
      ViewPort    0 0
   EndSubsection
   Subsection "Display"
      Depth       24
      Modes   "640x480" "800x600" "1024x768" "1280x1024"
      ViewPort    0 0
   EndSubsection
EndSection
Section "Screen"
   Identifier  "Screen 4"
   Device      "G200_4"
   Monitor     "NOKIA"
   DefaultDepth 16
   Subsection "Display"
      Depth       8
      Modes  "640x480" "800x600" "1024x768" "1280x1024"
      ViewPort    0 0
   EndSubsection
   Subsection "Display"
      Depth       16
      Modes       "1024x768"
      ViewPort    0 0
   EndSubsection
   Subsection "Display"
      Depth       24
      Modes   "640x480" "800x600" "1024x768" "1280x1024"
      ViewPort    0 0
   EndSubsection
EndSection

# **********************************************************
# ServerLayout sections.
# **********************************************************
# Any number of ServerLayout sections may be present.  Each describes
the way multiple screens are organized.  A specific ServerLayout section
```

may be specified from the X server command line with the "-layout" option. In the absence of this, the first section is used. When now ServerLayout section is present, the first Screen section is used alone.

ction "ServerLayout"
The Identifier line must be present Identifier "Simple Layout". Each Screen line specifies a Screen section name, and optionally the relative position of other screens. The four names after primary screen name are the screens to the top, bottom, left and right of the primary screen. In this example, screen 2 is located to the right of screen 1.

```
Screen "Screen 1" LeftOf "Screen 2"
Screen "Screen 2" LeftOf "Screen 3"
Screen "Screen 3" LeftOf "Screen 4"
Screen "Screen 4"
```

Each InputDevice line specifies an InputDevice section name and optionally some options to specify the way the device is to be used. Those options include "CorePointer", "CoreKeyboard" and "SendCoreEvents".

```
InputDevice "Mouse1" "CorePointer"
InputDevice "Keyboard1" "CoreKeyboard"

EndSection
```

Tcl/Tk 8.0 Installation :

1. change /usr/include/GL/glut.h line 202 from "extern void exit(int);" to "extern void exit(int) throw();"
2. install tcl8.0 from cdrom. Put the tcl8.0 CDROM into the drive

   mount /mnt/cdrom
   cp tcl80~1.gz ~~to-disk~~ / cp tcl80t~1.gz /
   umount /mnt/cdrom
   gunzip tcl80t~1.gz
   tar –xvf tcl80t~1
   cd tcl8.0/unix
   ./configure –enable-gcc
3. Error fix

   cd ../generic/
   Edit file tclPosixStr.c using vi, pico, or emacs
   line 340 Comment out
   // #ifdef EOPNOTSUPP
   // case EOPNOTSUPP: return "EOPNOTSUPP";
   // #endif
   line 787 Comment out
   // #ifdef EOPNOTSUPP
   // case EOPNOTSUPP: return "operation not supported on socket";
   // #endif
   cd /tcl8.0/unix
4. Make the files

   make
5. Install the files

   make install
6. install tk8.0. Put the CD labeled tk8.0 into the CDROM drive

   mount /mnt/cdrom
   cp tk80ta~1.gz ~~to-disk~~ /
   umount /mnt/cdrom
   gunzip tk80ta~1.gz
   tar –xvf tk80ta~1
   cd tk8.0/unix

tk8.0 has to be edited to accept the dual display required by Ventana. There is a file on a floppy labeled TKFrame.c. Copy this file into the current directory

   mount /mnt/floppy
   cp /mnt/floppy/TKFrame /tk8.0/generic/ TKframe.C
   umount /mnt/floppy
   ./configure --enable-gcc — run this from tk8.0/unix
   make
   make install

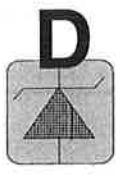
7. Create AIO16 under /dev directory

   cd /usr/bin
   ls –l cc
   if cc is not pointing to kgcc remove cc and make it again pointing to kgcc:

      rm cc
      ln –s kgcc cc

   cd /home/ventana
   mkdir source
   cd source

@ from bin type. • /rmmod AIO16 • /rmmod AIO16B Should Take a run.



Install floppy #1 of the DTEC source.
mount /mnt/floppy
cp –r /mnt/floppy/* .
umount /mnt/floppy
Change the floppy to #2 of the DTEC Source.
mount /mnt/floppy
cp –r /mnt/floppy/* .
umount /mnt/floppy
cd c++
cd analog_interface
make, ~~the modules~~
mknod /dev/AIO16 c 125 0
mknod /dev/AIO16b c 126 0
insmod –f AIO16.o
cd /usr/bin
rm cc (answer yes)
ln –s g++ cc
ls –l cc to check to make sure it points to g++.

8. Copy the DTEC profile to /etc directory. Insert the floppy disk labeled Profile.
mount /mnt/floppy
cd /etc
mv profile profile_bak
cp /mnt/floppy/profile .
umount /mnt/floppy

touch Installation

Install the Xtouch software. Put the floppy marked Xtouch #1 into the floppy dirve.

mount /mnt/floppy
mkdir /usr/lib/tsi
cp /mnt/floppy/* /usr/lib/tsi
umount /mnt/floppy
Put the floppy marked Xtouch #2 intot he floppy drive.
mount /mnt/floppy
cp /mnt/floppy/* /usr/lib/tsi
umount/mnt/floppy

Make a second directory for the second touch screen.

mkdir /usr/lib/tsi2
cp /usr/lib/tsi/* /usr/lib/tsi2
cd /usr/lib/tsi2
mv setup setup2
mv xtouch xtouch2
cd /

Source Code Installation

Install the source code into the ventana directory.

Put the floppy labeled Source Code Disk #1 into the floppy drive.

mount /mnt/floppy
cp –r /mnt/floppy/* /home/ventana/source
umount /mnt/floppy

Put the second floppy labeled Source Code Disk #2 into the floppy drive.

mount /mnt/floppy
cp –r /mnt/floppy/* /home/ventana/source
umount /mnt/floppy

Create the /var/log/ventana Directory

Linux will need this directory to know when it is OK to start the second GNOME Session.
Change directories to the /var/log directory.

cd /var/log

Create the Ventana directory inside the /var/log directory.

mkdir ventana

Change back to the directory you will run Ventana from.

cd /home/ventana

Run the Ventana Software

./start will run the Ventana software normally from the /home/ventana/source directory.

Appendix E Subsea Software Installation

WARNING!!! If the subsea software is running correctly and the software needs only to be updated, you should use the download new software function from the Ventana Control software. Follow instructions there. This document is meant for a total reprogram of the EEPROM and all the information will be erased. You should only do this under these circumstances:

1) This is a new EEPROM. or
2) the subsea software no longer boots correctly.

This Document contains step by step instructions to load the Ventana subsea software into the EEPROM. We need a floppy drive to be hooked up to the can. Alternatively, an IDE Hard drive with a DOS partition or Linux/DOS partitions combination will do as long as the files match the files as described on the floppies disk 1 and disk 2 are copied into the partition.

1) At boot up, press Delete key to get into the BIOS. You may also use the combination Ctrl+Alt+Esc to reboot into the BIOS if you didn't press Delete fast enough.

2) Use the right arrow key to select Boot-Seq menu.
Use the down arrow key to select Boot Sequence.
Use the Space Bar or +/- to cycle until "Menu" is chosen.

If using a Hard drive rather than a floppy drive, you need to also use the down arrow key to select "Drive C Assignment" and cycle until "IDE" is selected. Press Esc to get out of the Boot-Seq menu. Use the right arrow key to select the Shadow menu. Use down arrow key to select D400 ADAPTER and cycle until RW-Shadow. Do the same for the D000 ADAPTER. Press Esc to get out the menu and F10 to quit and save.

Insert disk 1 and boot from the floppy (press 'A'). Enter to clear time and date.

4) type: ***TFORMAT C: /S:GESTFF.BIN*** ('C:' if using the floppy, D: the Hard Drive) enter for OK "for all data to be lost"

5) type: ***SYS C:*** (or D) to install the MSDOS system files to the Flash disk.

6) Remove the Floppy disk, then restart the system.

If you used a Hard Drive, you will now need to go into the BIOS and change the D400 and D000 ADAPTER back to "ROM #2" or "vacant" and the "Drive C Assignment to SCSI to ensure that it does not boot off the IDE Hard Drive (see step 2).

Press 'C' key to boot into the Flash drive. Enter to clear time and date.

7) Insert Disk 2 and copy all the files into the FLASH drive. ***copy *.* C:***
There are a total of Five files, including the following:

Autoexec.bat	Loadlin.exe	DWLoad.exe
Bzimage	Fs.gz	

Remove the Disk from the Floppy Drive.

8) Reboot the system and select 'C:' from the Boot-Seq, and make sure that D000 and D400 ADAPTERS are on "ROM#2" or "vacant". The system should now be able to boot itself, uncompress linux, and load the modules. If you can see the penguin on the monitor, or hear the cycling. Then the software is loaded.

Boot sequence; C:1st Permit Boot A: No Permit Boot C: Yes
Drive C Assignment: SCSI

Appendix F Real Time Linux Installation

Setting up RTLinux for Ventana development

Installing RedHat Linux distribution, a workstation installation is sufficient for us.
DO NOT choose graphical login. In the case that you do, go to /etc/inittab and Comment out the last line.
#x:5:respawn:/etc/X11/prefdm –nodaemon

Before you begin, make sure you have gcc 2.7.2.3 or egcs-1.1.2 or egcs-2.91 installed. You can verify that by typing

gcc –v

If you do not have either of those two compilers installed. You can and should use the kernel compiler that comes with Red Hat Linux. Note: if you have installed Red Hat Linux using a straight workstation install, it will be necessary to install the kernel compiler. Put in RedHat 7.0 binary CD.

mount /mnt/cdrom
cd /mnt/cdrom/RedHat/RPMS
umount /mnt/cdrom
rpm -i kgcc-1.1.2-40.i386.rpm
whereis gcc

go into the directory.

cd /usr/bin
mv gcc gcc.bak // this will backup the normal gcc compiler
ln –s kgcc gcc // create a symbolic link to kernel compiler
ls -l gcc

This will allow tell RTLinux to use the kernel compiler rather than gcc. Note that we will most likely be using this version of gcc in the future since most of our ventana programs will be compiled as kernel modules and thus should be using the kernel compiler.

Alternatively, if we are going to be using the gcc compiler to compile normal programs. I.e. Non-kernel modules, we could change the makefile in the line.
CC = $(CROSS_COMPILE)gcc
in the Linux kernel Makefile to
CC = kgcc

Please note that the versions we have used for this installation is Rtlinux v. 3.1 and fresh linux kernel 2.4.4. If we wish to dual-boot, please ensure that there are two kernels and be very careful about the linux link in the /usr/src directory.

mount /mnt/cdrom (rtlinux installation cd)
cd /mnt/cdrom
ls
cp linux-2.4.4.tar.gz /usr/src/
umount /mnt/cdrom

Place a fresh copy of the Linux kernel anywhere you like in your machine. (From here on, assume that we have placed our fresh Linux kernel copy in /usr/src/linux and that we have renamed our old source tree so that we do not overwrite it):

cd /usr/src
tar vxzf linux-2.4.4.tar.gz

Place a fresh copy of the RTLinux kernel anywhere you like in your machine, and create a symbolic link to your Linux distribution source tree (From here on, assume that we have placed our fresh RTLinux copy in /usr/src/rtlinux):

rm -rf /usr/src/rtlinux // don't necessarily need this unless we change ver.
mkdir /usr/src/rtlinux
cd /usr/src/rtlinux
mount /mnt/cdrom
cp /mnt/cdrom/rtlinux-3.1.tar.gz .
umount /mnt/cdrom
tar vxzf rtlinux-3.1.tar.gz

Patch the kernel with the RTLinux patch:

cd /usr/src/linux
patch -p1 </usr/src/rtlinux/rtlinux-3.1/kernel_patch-2.4.4

At this point we have exploded linux and the rtlinux source code. The last Command patch has modified the necessary changes in the linux 2.4.4 kernel so that it is now the rtlinux kernel.

Compiling and setting up the RTLinux kernel.

Configure the Linux kernel:

cd /usr/src/linux
make mrproper
make clean

Either ***make config,*** *or* **make *menuconfig***, *or* ***make xconfig***, depending on whether you want to use the text, curses, or X configuration tool, respectively. (usually we will choose make menuconfig).
Configure your kernel by answering the questions that appear. Three particular things to notice. [ ] means unchecked. [*] checked. Only options need to be changed will be discussed below. Other options could depend on the new features needed or wish. The ones not mentioned should stay as defaulted.

Code maturity level options -->
    [ ] Prompt for development and/or incomplete code/drivers
Loadable module support →
[ ] set version information on all module symbols
    Processor type and features →
        [*] 486 Processor family
        [ ] Symmetric multi-processing support
    General Setup →
        [ ] Advanced Power Management BIOS support
    Memory Technology Devices (MTD) -->
        [ ] Memory Technology Devices (MTD) support
    Parallel port support -->
        [ ] Parallel port support
    Plug and Play configuration -->
        [ ] Plug and Play support

Block devices →
[*] Loopback device support
[*] RAMdisk support
[*] Initial RAMdisk (initrd) support
Multi-devices support (RAID and LVM) -->
[ ] Multiple devices driver support (RAID and LVM)
Telephony Support -->
[ ] Linux telephony support
SCSCI support -->
[ ] SCSI support
I20 device support -->
[ ] I20 support
Armature Radio support -->
[ ] Amateur radio support
IrDA (infrared support -->
[ ] IrDA subsystem support
ISDN subsystem -->
[ ] ISDN support
Old CD-ROM drivers (not SCSI, not IDE) -->
[ ] Support non-SCSI/IDE/ATAPI CDROM drives
Input core support -->
[ ] Input core support
Character devices --> LEAVE DEFAULTS ON
Multimedia devices -->
[ ] Video For Linux
File systems →
[*] DOS FAT fs support
[*] MSDOS fs support
Console drivers -->
[*] VGA text console
Sound -->
[ ] sound for Linux
USB support -->
[ ] support for USB
Kernel hacking -->
[ ] Magic SysRq key

non-essential configurations, but should be unchecked.

Note: the rest of the options are safe to remove or add depending on your need. Be aware however that our kernel is dedicated embedded kernel and thus it is advised to remove all the unnecessary features such as Sound, USB support, etc. The size of your kernel image will be greatly dependent on what features you have chosen.
Note: Please make sure to specify the correct CPU type for the target machine. (486)
Note: For machines with one CPU, it is advisable to disable SMP support.
Note: Enabling APM support is not recommended. APM BIOS calls may have unpredictable effect on real-time performance.

CAUTION: PLEASE MAKE SURE YOU COMPILE THE LINUX KERNEL ON A 586 OR HIGHER MACHINE. RECOMPILING ON A 386 WILL TAK 12 OR MORE HOURS TO BUILD A 2.4 KERNEL.