EARTH STATION CONTROL COMPUTER – OPERATING SYSTEM SELECTION

D. Varney and L. D'Addario

June 11, 1991

Abstract

This report describes a survey of real-time operating systems that may form a suitable platform for construction of the control software for the Green Bank OVLBI Earth Station. After considering the performance requirements, desire for standardization, hardware and software costs, and several other factors, we have selected Venix, by VenturCom, Inc. This includes a fully-preemptable real-time kernel that is compliant with UNIX interface standards, including SVID, Posix 1003.1 and 1003.4, and ABI/386. It runs on 80386 processors in the IBM-AT architecture.

1 Introduction

The software requirements for the OVLBI Earth Station control computer were outlined in OVLBI-ES Memo No. 8. In the present memo, we discuss the implications for the choice of operating system (OS), evaluate several candidate OS's, and report the final selection. This will have a strong effect on the choice of hardware platform, which will be discussed in a subsequent memo.

The main requirements on the OS are (1) that it provide good real-time performance for time-critical tasks, including predictable interrupt response and task switching times regardless of system loading; and (2) that it provide good support for networking, both local- and wide-area, including the protocols commonly used at the NRAO for remote access and file transfer, namely TCP/IP. To this we add a desire for good debugging facilities to ease development, especially for real-time tasks.

UNIX has become the most standard and portable operating system available. It provides the following benefits:

- a standard operating system;
- efficient software development tools;
- multi-tasking, multi-user capabilities;
- portability across different platforms; and
- support for advanced technologies, including networking.

UNIX has also become the *de facto* standard OS for general purpose computing within the NRAO. For these reasons, we prefer an OS that is as close to standard UNIX [1] as possible.

[1] "Standard UNIX" is a bit ambiguous, since it has developed in two somewhat different branches: the AT&T branch, currently System V; and the Berkeley branch, currently 4.3 BSD. But recently each branch has been incorporating features from the other, so that the differences are rapidly becoming unimportant. In addition, there is a set of IEEE standards being developed to which many vendors promise compliance.

Because of their history, the standard UNIX implementations lack real-time capabilities. In our application, there are several tasks that must be performed at accurately predictable times, including antenna motion control and two-way time transfer (Doppler compensation). Other users have similar needs, and operating system vendors have recently been responding to this. We can thus find on the market many operating systems that include a kernel which claims to provide good real-time performance, including deterministic interrupt response and task switching times. These have various degrees of compatibility with standard UNIX and various levels of support for networking and software development.

We have done a survey of some of these real-time operating systems in an attempt to select one well suited to our project. Included in the survey were:

- vxWorks, by Wind River Systems;
- VRTXvelocity, by Ready Systems;
- VENIX, by VenturCom;
- iRMX, by Intel;
- USX, by US Software;
- AMX, by Kadak Products Ltd.;
- OS-9/68000, by Microware; and
- LynxOS, by Lynx Real-Time Systems;

We obtained copies of the first three for testing, and evaluated the others through study of manufacturers' literature and review articles. We selected vxWorks because it is already in use at the NRAO on several other projects; VRTX because it is a direct competitor to vxWorks in the high-end real-time control market; VENIX because it has been selected for a related project at Interferometrics, Inc.; and the others because of favorable comments in the literature. Only the first three were studied in detail; most of the others were eliminated because they lack UNIX compatibility. (Although LynxOS appears to be UNIX compatible, it came to our attention too late for detailed study.)

2 VenturCom – VENIX/386

2.1 Description

VENIX/386, by VenturCom, is a real-time kernel for the 80x86 microprocessors in IBM PC-AT architecture machines. Supplied with it is a C library of routines for accessing the real-time features. The kernel is based on AT&T System V Release 3.2 UNIX but it is fully preemptive and includes various features designed to provide good real-time performance. It runs UNIX shells and applications provided by third parties. In particular, VenturCom re-sells the 386/ix operating system made by Interactive Systems, along with various other packages by Interactive, including a Software Development System (C compiler, linker, debugger, libraries, utilities), TCP/IP package (sockets, RFS, ftp, telnet), and graphical user interface (X Windows).

2.2 UNIX Compliance

VENIX/386 is said to comply fully with AT&T's System V Interface Definition (SVID), making it source-code compatible with any System V application. And it is said to comply also with the 80386 Application Binary Interface standard (ABI/386), making it compatible with binary code written for many other 80386-UNIX systems, including XENIX and SCO UNIX.

In addition, VENIX/386 version 3.2 is said to be 95+% compliant with POSIX (Portable Operating System Interface) 1003.1. (AT&T System V Release 4 and the future VENIX/386 version 4 are fully compliant with 1003.1). VenturCom also intends to comply with the real-time subgroup, 1003.4, while maintaining backward compatibility with current VENIX/386 applications. The current VENIX real-time functionality is a superset of the anticipated 1003.4 feature set.

2.3 Network Communications

Ethernet communication using TCP/IP is provided by an Interactive Systems package, which includes the Berkeley socket interface and extensive user-level applications. Included are:

- FTP, which allows transferring files to and from other hosts, including non-UNIX systems;
- Telnet, which provides the capability to remotely log into UNIX and non-UNIX hosts;
- BSD networking commands and applications, which provide access to resources on remote hosts, including rcp, rlogin, rsh, ruptime and rwho.

2.4 Debugging Support

Debugging under VENIX is possible at several levels: source, assembly and kernel. Source level and assembly level debugging are provided by separate programs (`sdb` and `adb`) having similar features. When working on a memory image from an aborted program, the debugger reports which line in the source program caused the error and allows all variables to be accessed symbolically. During an execution, breakpoints may be placed at selected statements, or the program may be single-stepped. Procedures can also be called directly from the debugger.

Kernel debugging is provided by a RAM-resident routine that uses RPN style input to inspect arguments and executions. The debugger has the capability of referring to objects in the kernel by a symbolic name. I/O ports, registers, and process status can be inspected.

Additional debugging support is provided by a trace function that can be attached to any process. `Trace` allows the UNIX kernel or a device driver to report debugging information without the use of the `printf` function. The basic mechanism allows calls to the trace driver to store short bursts of data in system character buffers for later retrieval. The trace function has an average execution time of 300μsec.

2.5 File System and I/O

VENIX includes a file system in which each file consists of physically contiguous extents of arbitrary size. The contiguous file system also guarantees a single disk access to any portion of a file.

Access to MS-DOS files is possible through VP/ix, an optional utility that allows DOS to be run as a UNIX process.

A process can overlap I/O transfers with processing via the UNIX buffer cache, but unbuffered I/O is also supported; this provides controlled transfers, making possible a much higher throughput for large blocks.

2.6 Documentation

Documentation provided with VENIX is comprehensive and easy to follow. VenturCom supplies one bound volume on the kernel and real-time library, and most of the other documentation originates with AT&T or Interactive. We received the following volumes with our evaluation system:

- o Programmers Guide, 2 volumes (AT&T);
- o Programmers Reference Manual (AT&T);
- o Utilities Release Notes (AT&T);
- o UNIX System V/386 Users Guide (AT&T);
- o Systems Administrator's Reference Manual (AT&T);
- o Systems Administrator's Guide (AT&T);
- o NFS and TCP/IP in 2 binders (Interactive);
- o 386/ix Operating System Guide (Interactive);
- o 386/ix Software Development System Guide (Interactive);
- o UNIX System V.3.2/386 Integrated Software Development Guide (Interactive);
- o VENIX Real-Time Programmers Guide (VenturCom).

2.7 Other Features

Resources are managed with a mechanism known as a lock. Only one process may hold a lock at any given instant. When a lower priority process holds a lock to a resource, a higher priority process is placed in a request queue. When the lock is released, the highest priority process awaiting its release will be granted the lock. The lock mechanism solves the semaphore flag problem, as a flag operation is a non-atomic function that is open to preemption and subsequent modification.

Virtual memory is supported while maintaining determinism by locking real-time processes in RAM, thus avoiding virtual paging faults and subsequent disk accesses.

VENIX uses device drive daemon processes to shift excessive interrupt processing from interrupt level to process level. This allows full preemption and schedulability. Daemons are system processes that start at initiation. A driver daemon is similar to an interrupt service routine; they are available in VENIX for such services as kernel mode swapping, logins, memory management and buffer caching. Daemons allow context switching, where an interrupt handler usually does not.

3 Wind River Systems – vxWorks

3.1 Description

This product is a real-time kernel, library, and shell for execution on 680x0-series microprocessors. Various single-board computers are supported, including many of the VME boards made by Motorola. Wind River markets vxWorks primarily for embedded processor applications, where an OEM incorporates his application code with vxWorks library routines and kernel into a processor that forms a fixed part of some larger product, with the product being produced and sold in volume. Accordingly, the development environment is kept completely separate from the execution (or "target") environment; on the latter, only a simple one-user shell and a few utilities are provided. VxWorks relies on a cross-development system running on a UNIX workstation, typically a Sun 3 or 4, and linked to the target via ethernet. Editors, compilers, debuggers and other development tools are supplied by third parties to run on the workstation. The vxWorks OS running on the target is then capable of loading, executing, and debugging user code over the network.

3.2 UNIX Compliance

The target system, while somewhat UNIX-like, is not compatible with UNIX standards and cannot run third-party software. UNIX compatibility is dependent on the workstation's OS.

3.3 Network Communications

The vxWorks kernel and libraries provide extensive network support for TCP/IP communication over ethernet. Included are sockets, remote procedure calls, remote login, telnet and ftp.

3.4 Debugging Support

Debugging tools are available on the target as part of the vxWorks shell, allowing low level monitor-like functions such as inspection of memory, display of task status, loading and deleting program modules, disassembling of code, starting and stopping tasks, and calling any loaded function. There is a global symbol table, so the shell can access symbolically all entry points and global data from the application code.

Remote debugging over the network is also possible using third-party software running on the workstation. Currently, Wind River bundles the GNU debugger (and also the GNU compiler and linker) with vxWorks. This, together with a remote procedure call support task running in the target, allows source-level debugging, including source code breakpoints, single-stepping, and symbolic subroutine traces.

3.5 File System

The primary file system in vxWorks uses a non-hierarchical RT-11 compatible format. However, two additional file system options are available (in version 5.0): one compatible with MS-DOS 4.0 files, and hence hierarchical; and a "raw disk" system where an entire disk is treated as one large file.

The current file systems are significantly different from the one (RT-11) used in the previous version of vxWorks (version 4.0.2), which until recently was the version used by most groups at NRAO. Considerable rewriting of the MVME320 disk controller code at Socorro was required because of this.

3.6 Documentation

Documentation for vxWorks consists of a binder for the reference manual and a spiral bound programmers guide. The guide had some useful examples, but the material generally lacked depth and detail. Little tutorial material is included.

4 Ready Systems – VRTX

4.1 Description

Ready Systems has developed a real-time kernel called VRTX32 in versions for many 16- and 32-bit microprocessors, including the Motorola 68000/08/10/20/30; Intel 8086, 80186/188/286/386; AMD 29000; SPARC; and others. The kernel provides a deterministic response time independent of the number of tasks, and includes the usual task management, intertask communication, and interrupt servicing features. The functionality of the kernel and its high level language interfaces are independent of the target processor. The design emphasizes optimum real time performance, with low interrupt latency and fast context switching. Available to run on top of the kernel are a set of file system management routines (IFX) and a set of network support routines for TCP/IP (TNX).

Software development can be done on a workstation separate from the target system, as in vxWorks, by relying on ethernet communication. Ready Systems has available a Sun-hosted cross-development package, VRTXvelocity, to support this. The concept of this product and its intended market are very similar to those of vxWorks; indeed, the two are direct competitors and together are the leading contenders for this market.

The development system and target may also be the same. In particular, a PC-AT with an 80386 can be used by running Ready System's VRTX-PC/386. This product provides interfaces and drivers for installing VRTX32 on a PC while making partial use of the PC BIOS. It also provides a shell and development environment that allows some debugging tools to be run in the real time system. However, compilers, editors, and other development tools are run under DOS; a mechanism is provided for switching between the DOS and VRTX OS's. Both can access the same files.

4.2 UNIX Compliance

No compliance with UNIX standards is claimed for VRTX32, so third party software will not run on the target system. The cross-development system relies on SunOS; the PC system relies on MS-DOS.

4.3 Network Communication

The networking package (TNX) includes TCP/IP support and uses the UNIX 4.3 BSD sockets programming interface. An ethernet driver is provided only for the Motorola MVME147 board; users of other boards must provide their own. Utilities including telnet and ftp are available as part of the development packages VRTXvelocity and VRTX-PC.

4.4 Debugging Support

With the cross-development package VRTXvelocity, debugging aids are provided at the system level and source level. At the system level, a utility called RTscope allows objects can be examined while the code is executing. Breakpoints can be set within tasks or interrupt service routines (ISR's). The source-level debugger, RTsource, can inspect multiple tasks simultaneously. The user can start/stop task execution, set breakpoints, and modify and set watch variables.

RTscope is also included with VRTX-PC, but RTsource is not.

4.5 File System

The file system (IFX) is MS-DOS compatible, and supports features common to MS-DOS and UNIX like working directories, default devices, pipes, and standard input and output. There are separate sets of routines for disk I/O and stream I/O. A disk can also be treated as a single large file, although it is then not DOS compatible.

4.6 Documentation

The documentation for VRTX32, both the PC and Sun-hosted versions, is comprehensive and understandable, with many helpful examples and numerous tutorials. We received seven binders for the 680x0 version and four for the PC version.

Requirement	VRTX	VENIX	vxWorks
target hardware	680x0/80x86[6]	80z86[6]	680z0[6]
documentation	7/4 binders	12 books	2 binders
supplied examples	excellent	very good	poor
customer support	excellent	excellent	
support within NRAO	none[1]	none[2]	excellent
portability	excellent	very good	good[3]
backward compatibility	excellent	very good	poor[4]
user comments	excellent	very good[2]	good[5]

Notes

[1]not available at NRAO

[2]via Interferometrics Inc.

[3]design issue, could be improved

[4]porting from 4.0.2 to 5.0 has proven problematic

[5]NRAO users

[6]$x \in \{0..4\}$, $z \in \{2..4\}$

Table 1: Operating system evaluation results

5 Discussion and Conclusions

A summary of the characteristics of the three primary systems surveyed is given in Table 1, and costs are summarized in Table 2. The costs of all components of a given OS that we are likely to purchase for our project are included, and all discounts available to the NRAO have been applied.

We see very little difference between vxWorks and VRTXvelocity (neglecting VRTX-PC for now) in functionality or performance. A major factor in favor of vxWorks is that it has already been adopted for other projects in the NRAO, so that there might be some sharing of expertise. On the other hand, some ancillary information is negative: the documentation is poor, some users think that the company's support people are not very knowledgeable, and their recent release of version 5.0 fails to be backward compatible in major ways, forcing users to re-write application code. In addition, the development system price has increased 80% in two years, as has the annual maintenance charge. VRTXvelocity's list price is the same as that of vxWorks, so both are very expensive; but whereas NRAO already owns some copies of vxWorks, Wind River considers additional purchases of the development system as "additional server licenses" for which they charge 50% of the price of an original purchase, making it more competitive. (The prices in Table 2 are an attempt to give the actual costs to our project, so this consideration is included.)

VRTX-PC is somewhat attractive, but its lack of UNIX compatibility and lack of a source-level debugger make it less desirable than the other candidates.

The product that we see as best for our application is VENIX, marketed by VenturCom. They provided very helpful technical support during our evaluation and the documentation includes abundant examples. Although the Observatory has not yet purchased this product for other projects, Interferometrics Inc. is porting the Mark III "Field System" to VENIX, and they promise to incorporate VLBA recorder support into that package. There is thus one other nearby group using this OS in a related application. NRAO has a software transfer agreement with Interferometrics.

It is our view that the major advantage of vxWorks, namely that it has already been adopted by several projects within the NRAO, is outweighed by the following advantages of VENIX:

- compliance with UNIX standards;

Component	VRTX32/680x0	VRTX32/PC	VENIX	vxWorks
development software	16,000	10,240	3,060[1]	9,750
target software	500[10]	300[10]	1,027[11]	600[10]
annual maintenance	2,400	1,320	3,000[2]	3000
training classes, 1 person	1,495	995	1,125	1,950[6]
compiler	included	716[3]	included	included[5]
assembler	"	1,036[4]	"	"
debugger	"	156[4]	"	"
SW cost w/3yr maintenance	25,195	17,403	10,202	21,300
target hardware	9,420[7]	4,500[9]	4,500[9]	9,420[7]
development hardware	4,800[8]	4,500[9]	4,500[9]	4,800[8]
total	39,415	26,403	19,202	35,520

Notes

[1]includes networking and DOS⇌UNIX
[2]first year, $995 thereafter
[3]Metaware
[4]Phar Lap
[5]Gnu, Free Software Foundation products
[6]published price; free classes for 2 persons negotiated for GBT
[7]VLBA station computer configuration
[8]SPARCstation IPC
[9]80386-33, FPU, VGA, 8MB RAM, 100MB HD
[10]embedded system target
[11]only runtime system and TCP/IP on target PC

Table 2: Operating system costs, including all available discounts

- o ability to do development on a system identical to the target;
- o much lower cost for the OS itself; and
- o much lower hardware costs and maintenance costs.

We believe that using a UNIX-compliant OS will make our code easier to maintain because of the ability to incorporate third-party hardware and software without relying on support from the OS vendor. It will also make our code more portable. In the case of our small project with only one target computer, the ability to have an inexpensive but directly compatible development machine is non-trivial. If we used a cross-development system like vxWorks, we would be likely to need two target machines as well as the development workstation in order to allow the "real" target to be used for hardware tests while the other is used for software development. (Even the VLBA control system, which will have 10 targets in the field, needs two additional targets in the lab to support hardware and software testing.) When the very high initial and continuing costs of vxWorks and VRTX are added to these considerations, it seems clear that neither of these is appropriate to the OVLBI earth station project.

Therefore, VENIX has been selected as the software platform on which the earth station control system will be built. It is likely that there are other OS's that would also be acceptable; but in view of our schedule for completion of the project, we do not propose to do additional evaluations.