LS-147(ANL)
M. Knott, M. Kraimer, and F. Lenkszus
May 1990

APS CONTROL SYSTEM OPERATING SYSTEM CHOICE

Contents:

Introduction
What is the operating system?
APS control system architecture - IOC, OPI, Network
RISC / CISC
What is UNIX?
UNIX and RISC - Future trends
Other projects
The choice

Introduction:

The purpose of this document is to set down the reasons and decisions regarding what is an important choice for the APS Control System design staff, namely the choice of an operating system for its principle computer resources. Since the choice also may affect cost estimates and the design handbook, there is a further need to document the process.

The descriptions and explanations which follow are intended for reading by other APS technical area managers and will contain a minimum of buzz-words, and where buzz-words are used, they will be explained. I hope that it will help in understanding the current trends and developments in the volatile and fast-developing computer field.

What is the Operating System?:

The Program's View:

The operating system can be thought of as the software which lies between the actual computer hardware and what we think of as the program. There are actually several additional layers between the operating system and the program. In these layers are routines to handle storage devices and keyboards, libraries of mathematical functions, and many others.

The Programmer's View:

To the human interacting with the computer, the operating system appears as the interpreter of instructions to manipulate files, edit and compile programs, search for information within files and directories, invoke various system utilities such as mail, and to set various personal features such as terminal type and custom translation of commands.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

The User's View:

To the user of a well-written application program there is usually a minimum of interaction with the operating system, but even that can be unpleasant. Operating systems can be cryptic in pointing out errors, for example. In short, the very attributes which make the operating system easy and efficient for the programmer can make it unfriendly to the casual user.

APS Control System Architecture:

The APS control system is based on a distributed set of Input-Output Controllers (IOCs), each having a microprocessor and appropriate interface modules for connection to the equipment being monitored and controlled. The IOCs have processing capability which is used to convert raw binary data to engineering units, check for alarm limits, and perform stand-alone control loop and action-reaction tasks. The IOCs will use a high-performance real-time operating system, VxWorks, which provides many development features such as a run-time symbolic debugger.

At the operator interface level, a high resolution graphic workstation called an Operator Interface (OPI) is provided. This is where so-called application programs are run - programs which emulate control panels of the APS equipment, provide graphic tools to monitor and study machine parameters, provide both simple and complex control of machine equipment, log and archive collected data for later analysis, and provide links to other computation facilities. Since the OPI can be called upon to perform not only complex computational tasks, but also update display windows, respond to operator input from the keyboard or mouse, and respond to and generate network traffic, its implementation should utilize the most capable computation platform we can afford.

Connecting all IOCs and OPIs together is a communication network which will be initially implemented with Ethernet, a 10 M-bit/second multiple-access (or party-line) protocol. If, due to functional or numeric expansion of the network, Ethernet proves to have inadequate bandwidth, the Fiber-optic Distributed Data Interface (FDDI) with its 100 M-bit/second capability will be installed.

RISC / CISC:

A recent innovation in computer architecture is the Reduced Instruction Set Computer (RISC) architecture (actually a re-innovation, since this same architecture was utilized in some early computers). The study of computer instructions and their relative frequency of usage has revealed that most of a computer's time is spent in the execution of a small subset of its repertoire of instructions. RISC architecture takes advantage of this fact by streamlining the execution of this subset and by implementing the less used and more complex instructions with combinations of the (now fast) small set of instructions. Since there is now a small set of simple instructions, parallel "pipelining" can be used to increase execution speed. In this method more than one instruction can be executed simultaneously by staggering in time the various sub-operations. Some processors can even average more than one instruction per clock cycle.

The converse to RISC architecture is Complex Instruction Set Computer (CISC) architecture. (Although it predated RISC, the CISC acronym was not coined until RISC was developed.) Most computer architectures developed prior to 1980 are of the the CISC type, a typical example being the VAX. Although it is possible to pipeline CISC instructions, such a design will be very complex and expensive.

The other computer architecture improvements developed in recent years, such as memory caches where future instructions and/or data are read from memory on the chance they will be needed in order to avoid the delay associated with the main memory, can be added to both RISC and CISC but the entire cache process is always more complex and expensive with CISC.

The reason for including a discussion of RISC vs CISC is because it is apparent that for any given semiconductor implementation, the RISC architecture will always be able to out-perform CISC. Even today there is a five-to-one advantage in raw MIPS (millions of instructions per second) for RISC devices. This should be discounted to some degree since RISC requires more instructions to perform some types of operations, but an advantage of even two-to-one is obtainable on reasonable benchmarks.

What is UNIX?:

UNIX was developed for AT&T by their development arm, Bell Labs. It was developed as a word processing tool, but soon expanded to support software development and finally grew into a full-featured operating system. Most operating systems are written in some compiler language, for example the VMS operating system used by the VAX is written in BLISS, a proprietary language designed to take advantage of all features of the VAX architecture. VMS itself is proprietary of course, and VMS and the VAX were designed at the same time and are intimately related. UNIX was written in the "C" language, also developed at Bell Labs. Since AT&T was at that time part of a regulated monopoly, it was prevented from marketing UNIX or C, and both moved effectively into the public domain. C has since become a standardized language like FORTRAN, and is offered for nearly all computers as a well-developed and very popular language. UNIX was written by programmers for use by programmers and as such, it has many peculiar aspects, not the least of which is its set of cryptic commands.

The keys to UNIX's success however, are that it is extensible, is in the public domain (in that several vendors market compatible, licensed copies), and is written in a popular "public" language. These facts combine to allow anyone the ability to make enhancements, remove features, and tailor it to specific needs. At first it was a curiosity in the computer science departments of universities, but it soon became apparent that if an application program or utility program could run on one computer under the UNIX operating system, it could run on any computer under UNIX. One indication of this universality is the fact that the UNIX operating system is available for microprocessors as well as supercomputers. Thus, if a startup company chose UNIX as its operating system and made the minor changes necessary to support its chosen computer architecture, any existing software that ran under UNIX could run on their computer. In this way, a major breakthrough was made, similar to that prompted by the IBM-PC and MS-DOS, in that a new computer could be introduced with a complement of software already available.

When the statement is made that a program can "run under UNIX," this means that it makes no special demands on a particular operating system or hardware, and if simply recompiled with the UNIX-based computer's compiler, it will perform exactly as expected. All interactions with the operator will be unchanged. Sometimes changes in the file structure or communication system will have to be adjusted, but these are examples of the dependencies referred to above.

UNIX and RISC - Future trends:

Because of the two developments described above, namely RISC and UNIX, nearly all companies in the computer business are developing and marketing RISC-based computers and workstations which run under the UNIX operating operating system. Competition is forcing higher performance while keeping prices low and this trend is likely to continue. There will always be a market for CISC architecture computers and operating systems such as VAX/VMS, but a principle reason is the installed base of application software and the steady improvements made to the hardware by companies like DEC and IBM. In fact, as computer performance increases, vendors can afford more user-friendly interfaces and it will matter little whether the underlying operating system is UNIX or some other since the user will not have to deal with it.

Since UNIX is extensible and easily modified and improved, vendors have used this fact to help their market position and provide their own "Enhanced UNIX." This prevents true transparency between vendors' products. Various standards efforts are trying to address this and vendors will follow standards as they are developed or risk being left behind their competition.

The U.S. federal government is moving in the direction of requiring UNIX for computer acquisitions in many departments. While this will not likely affect us directly, it will definitely pressure the computer market in this direction.

Other Projects:

It is sometimes useful to monitor the choices made by other accelerator-related projects and I will list a few. ESRF has chosen the HP line of computers running under UNIX. CEBAF has made the same two choices. Nearly all new activities at CERN employ the UNIX operating system, although this can be tempered by the fact that UNIX has enjoyed a greater and longer popularity in European laboratories. CERN has even established a policy of requiring UNIX in all future applications. ALS is developing a control and monitoring system based on an in-house architecture and may not employ an operating system as such. However, this choice is considered by many as an aberration. They will be including workstations as upper-level consoles and although some will be IBM PCs (or equivalent), others will no doubt be of higher capability and will likely be RISC/UNIX based.

The Choice:

It would seem to be obvious on the face of it to choose UNIX as the operating system for any system to be delivered in the mid 1990s. There are, however, some mitigating factors. Most important is that UNIX has several deficiencies related to real-time processing such as guaranteed priorities, preemptive kernel, etc. Some vendors address this problem by providing "real-time extensions." HP is one such vendor. Standardization efforts such as POSIX are addressing this area and other vendors will add such features. It should be noted that the operating system to be used at the microprocessor, or IOC level, is to be a high performance real-time system known as VXWorks and will not be subject to this problem. UNIX will be used only on the host computer and console workstations. The VXWorks software is developed under UNIX (using SUN computers in our case), but this does not in itself demand that UNIX be the operating system for the remainder of the control system.

Another edge to the UNIX sword is the possibility that with many vendors marketing RISC/UNIX products in this very competitive field, the vendor chosen solely for price/performance reasons may be driven out of the market in the future. So the choice comes down to getting the best price/performance ratio, the highest performance for the available funds (you never have enough performance), and being careful to choose a firm with staying power.

For the reasons documented above, we feel that UNIX would be a prudent choice for the operating system of the host and console workstations. The possibility of mixing products from two or more vendors who subscribe to the same standards will not only enable us to take advantage of market competition and obtain the highest performance/cost ratio, but also make us somewhat immune to the failure of the initial supplier of equipment. When coupled with RISC architecture processors, a future of steadily increasing performance availability at reasonable cost is assured.

The following items summarize the reasons for choosing UNIX:

* Interoperability of various software and hardware components from several vendors. This will provide the option of using equipment from several vendors and provide a measure of protection from business failures.

* Availability of software developed at other accelerator laboratories which have adopted UNIX (a growing list).

* Higher system performance (both initial and future) employing RISC architecture.

* Indirect benefits gained by adopting an operating system which is under the influence of standardization efforts.

It should be added that the adoption of UNIX does NOT mean that everyone must learn to deal with UNIX. Only those who develop software for use with the control system will have to be concerned with this development and in those cases they may welcome what is reputed to be a superior software development environment. As for VMS systems and experience, we intend to provide a convenient means for VMS (and other) computers to communicate with the control system and to have access to all accelerator data.

We will now proceed to evaluate workstations and computer servers which provide UNIX as at least one operating system choice. RISC architecture will no doubt be a feature of the leading candidates.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.