Digital's Direction in Real-Time

The Alpha chip's penetration of the embedded systems market is not occurring in real-time, exactly, but by leveraging its historical position as a leader in real-time operating systems, Digital could start gaining speed.

The embedded systems development market — putting processors in other hardware such as dashboards, data acquisition devices and rocket ships — is about $1 billion strong annually and is growing by nearly 10 percent per year. Hardware sales account for about 80 percent of the market, and Digital wants the Alpha chip to get its share. The market is dominated by the Motorola 68000 family of microprocessors and the 960 line from Intel. For high-performance applications, Sun Microsystems' Sparc chip has grabbed an enviable amount of marketshare.

The remaining 20 percent of the market goes to software, and it is in this area that Digital will need to make strides to get the Alpha accepted as a standard real-time target processor. The company certainly is trying hard.

From RT-11 to Mach

Digital is so anxious to move forward in the real-time market that it is working closely with IBM to produce a new real-time operating system, code-named Libra and based on IBM's microkernel technology. IBM's technology is base on the Mach operating system developed at Carnegie-Mellon University in the mid-1980s. The original version of Mach ran on a VAX-11/784 but now runs on virtually all architectures, since the Open System Foundation (OSF) standard is based on Mach.

Mach is considered a distributed operating system, and thus Digital's move toward a distributed operating system both for real-time applications and for the operating system that may someday replace OpenVMS implies that the real-time and general-purpose operating system markets are merging. Other distributed operating systems are also finding implementation in real-time environments.

The strongest factor fueling the drive to distributed operating systems for real-time applications is the fact that most real-time applications now require process scheduling and a variety of other common operating system services as well as the need to communicate with other real-time applications running on machines located elsewhere on a network.

Unfortunately, most real-time operating systems are like Digital's VAXELN, which dictates that the user compile the application on the development host and then download the binary to the target system over a serial or Ethernet connection, where it is then tested and debugged. The operating system is downloaded at the same time, unless it has already been downloaded or resides in firmware on the target.

A distributed real-time operating system, on the other hand, allows the user to both compile and debug from the development host across the network

Bradford T. Harrison

directly on the target system. A download is no longer required, and concurrent development on multiple targets that may themselves be communicating is facilitated.

Ironically, Digital's RT-11, the company's first dedicated real-time operating system, also ran directly on the "target," a PDP-11, and allowed the user to compile and debug by directly using its facilities. The terms "host" and "target" didn't apply, however, because the application was developed on the same

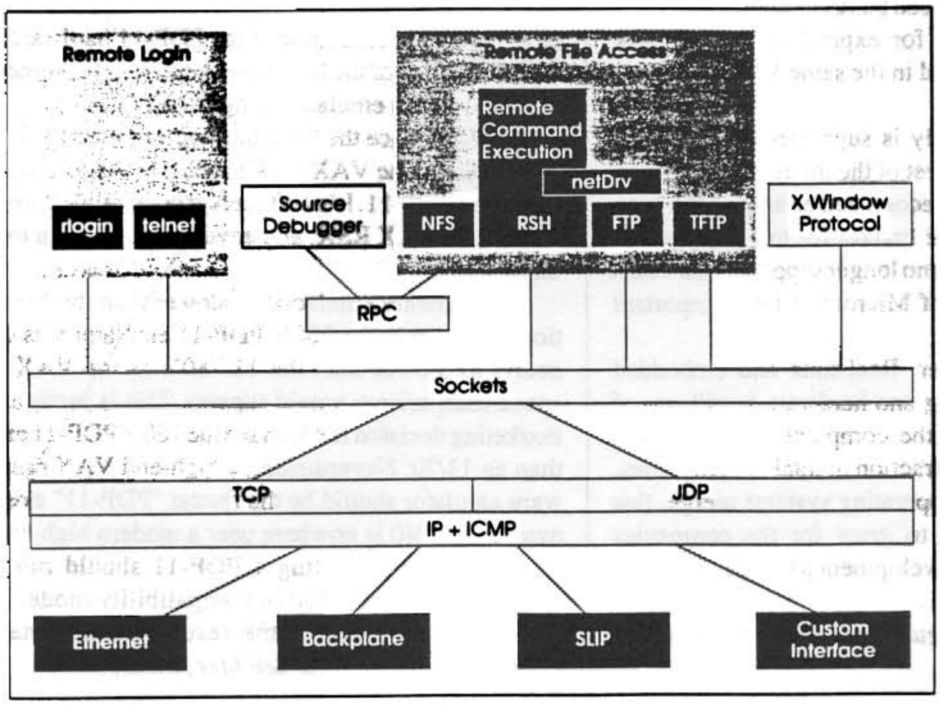

Figure. VxWorks network components.

machine that was used to deploy the application.

Interestingly, RT-11 was one of Digital's biggest successes. Not only was it purchased widely for manufacturing and other real-time applications, but it was also used as a general-purpose operating system. Digital sold it as a single-user system, but several third-party companies such as S&H Computer Systems (Nashville, Tenn.) developed packages layered on RT-11 to convert it into a multiuser system. Digital sold the COS-300 product, which was layered on top of RT-11 and allowed multiple users to run DIBOL programs. With the multiuser programs, PDPs running RT-11 were deployed for many common business applications as well as real-time applications.

Then Digital went VAX/VMS-crazy and refused, even under a deluge of complaints from DECUS members, to move RT-11 to the VAX. The potential product, referred to as RT-11/32, never appeared, and the exact reasons Digital didn't respond will never be known. Most of the RT-11 product development group at Digital left in disgust, and the real-time community had to make the choice between VAXELN and migration to a different platform. Most chose third-party development tools running on UNIX platforms — especially SunOS. Digital lost its core real-time customer base and has never regained it. Today, with the real-time possibilities for Alpha, that hurts more than ever before.

VxWorks

RT-11 has all but faded away, and Digital continues to se VAXELN for real-time development on OpenVMS systems, b the company's current thrust in real-time operating systems — least until Libra arrives — is VxWorks. Digital is buying th product from Wind River Systems (Alameda, Calif.) and enhan ing it. Like VAXELN, VxWorks is implemented in the hos target environment described above and is available only on DEC OSF/1 Alpha host. Digital is not su porting it on OpenVMS systems.

VxWorks applications are downloaded alo with the operating system kernel to either Alpha or a Motorola 68000 VMEbus-bas target system, where they are debugged usi the VxGDB source-level debugger, also fro Wind River. However, Digital supports thr APIs in its version of VxWorks: VxWor native, POSIX.1 (real-time POSIX, former POSIX.4), and a VAXELN subset for tho porting VAXELN applications from OpenV to DEC OSF/1.

The Figure shows the network componer of VxWorks. These components give VxWor many powerful distributed computing faci ties. The RPC facility allows VxWorks app cations to communicate with the debugg running on the development host as well with other VxWorks applications running other targets. After initializing a target sy tem, the network is transparent to applicatio using the VxWorks native API. File access and procedure calls that occur over the network using NFS a RPCs are transparent to applications and look exactly like loc file accesses and procedure calls.

COHESIONworX and Board Support Packages

Digital is supporting VxWorks and DEC OSF/1 real-time dev opment work in the COHESIONworX software engineeri environment. This environment is based on Digital's COH SION CASE product that it has been selling for years, but use COHESION in the real-time environment is especially import now, since highly functional development environments becoming the critical component in the offerings of leadi embedded systems development tool companies such Micro Research and ISI (both of Santa Clara, Calif.).

Digital is also offering board support packages (BSP) for AXPvme 64 and AXPvme 160 single-board computers (SB BSPs facilitate the development of real-time applications providing the software interface to processor-specific features SBCs such as registers and interrupt structures, making it eas to develop applications. Real-time application developers sa time using prebuilt BSPs.

With COHESIONworX, VxWorks, AXPvme 160 SBCs i VMEbus chassis, and the accompanying BSP, Digital is offeri one of the most sophisticated embedded real-time programming environments for the industry's most powerful chip. This is a 64-bit chip running at 160 MHz, configurable with up to 128 MB in RAM. The package is most suitable to applications like image processing, high-speed telecommunications, process control and real-time simulations.

Alpha boards also implement a peripheral component interconnect (PCI) technology for their Ethernet, SCSI-2 and VMEbus interfaces. The PCI bus operates at 32 MHz for a peak theoretical bandwidth of 128 MBps. A high-speed bus that supports standard hardware interfaces is important for expanding the range of possible cards that can be packaged in the same VME card cage with the SBC.

Unfortunately, the Alpha family is supported for real-time applications only by Digital. The rest of the industry seems to be looking away. The technology is recognized as among the best, but Digital neglected the real-time market for too many years. Companies like Microtec Research no longer support OpenVMS, although at one time it was one of Microtec's most important development platforms.

But the game is far from over. Real-time and embedded systems applications programming and hardware is still one of the fastest growing segments of the computer industry, even though it accounts for just a small fraction of total industry sales. If real-time and general-purpose operating systems merge, this percentage of the pie is certain to grow for the companies positioned with comprehensive development packages that support leading hardware, like Digital.

—Bradford T. Harrison writes regularly for DEC Professional *and* Internetwork *magazines.*

Q **We are considering upgrading our VAX-11/75 ning VMS V4.1 and the RSX emulation packa new system. My application is a cross-assembler compiling** code for a Motorola 6809 embedded system. **The assembler uses** the RSX mode of the VAX **As far as I know, the VAX-**11/750 emulates the RSX instructions in hardware, and the new system uses software emulation for a MicroVAX 3100 Model 40 **Is such an** upgrade **viable, or is** the RSX mode dead?

A RSX is an operating system for PDP-11s. **What the 11/**750 is emulating is the PDP-11 hardware architecture Much of the RSX code runs directly. Some of it traps out **to VMS,** which emulates things like I/O.

No VAX since the 8650 has implemented PDP-11 **emulatior in hardware. The** VAX RSX product contains a software emula tor for the PDP-11. It's 100 percent compatible with the hardware emulation. VAX RSX, and anything you run on top of **it, won' know the difference.**

The software emulation is slower than the hardware **emulation. As I recall, the 750's PDP-11 emulator was quite fast, not nearly as slower than the 11/780's as the VAX native-mode speed comparisons would suggest. This is partly an artifact of a marketing decision not to make the 780's PDP-11 emulator faster than an 11/70. Nevertheless, a high-end VAX running the software emulator should be the fastest "PDP-11" ever built. However, a 3100/40 is nowhere near a modern high-end syst**

A 3100/40 emulating a PDP-11 should run around one third the speed of a 750 in compatibility mode. You'll really have to try it to see if the results are acceptable for your application. — ***Jerrold Leichter, Ph.D.***

Getting to Know NetBIOS

continued from page 10

For example, if you have users claim their usernames during login, you can type "NAME username" and see what PC the user is using (perhaps to then use the **PC name in ADMIN/PCSA BROADCAST,** which is limited by a bug **to six characters for the node name), or** "NAME pcname" **to see who is logged into a given PC.**

You can use CLAIMASY to speed booting of the PC. If you **claim (asynchronously or otherwise, but only** the asynchronous **claim makes for the net time saving) the PC's name** (Byte16 == 0) before setting the name with the new PATHWORKS **version 5.1** command USE/SETNAME, the already-claimed **name will be used, saving a broadcast timeout wait. Similarly, if you claim the PC's name with Byte16 == 3 before running RCV,** some smart code in RCVMORE will allow RCV **to use the already-claimed name.**

IF YOU HAVE UNDERSTOOD this so far, **I hope you can see some of the reliability implications of NetBIOS:**

■ **If a node already owns a name but is temporarily disconnected from the extended LAN for any reason, another node can legitimately or otherwise claim the same names as the first node leading to confusion or worse. This can also happen if the** network is very busy, perhaps because of a lot of PCs shouting a one another in NetBIOS, so that claims **don't get seen by nodes that might want to reject them.**

■ **Because of the** trade-off between **the number of retries re**quired to make sure a multicast gets through and the time taken **it is possible for a message simply not to arrive. For example there** is no guarantee that a **pop-up message sent by ADMIN/** PCSA BROADCAST **(or BCAST, from a PC) will arrive at its** destination.

■ There **is nothing to stop a PC from claiming a whole lot o** currently unclaimed names that it doesn't need. For example, try claiming the name of your neighbor's PC **with Byte16 == 0 while it is switched off,** and then **watch his or her face as [USE** SETNAME] fails when he or she boots. There could be hours o fun for the dedicated troublemaker.

—Nick Brown is Network Manager for a European intergovernmer organization based in Strasbourg, France. He has been w ***with Digital systems since 1982 and with PATHWORKS since 1989. Contact him on the Internet at nick.brown@dct.coe.fr or on CompuServe at 100113,2107.***