

Nucleus MNT and Nucleus VNET Windows NT and 95 Based Version of Nucleus PLUS

More and more companies would like to develop their software at the same time as their hardware, so that their product can be released much faster.

With the introduction of Nucleus MNT and Nucleus VNET it is possible to prototype the software without having all the hardware equipment that is needed for the embedded system.

INTRODUCTION

Developers of embedded systems are increasingly under pressure to get their products to market faster. Since these products generally require software and hardware, software efforts are often stalled while waiting for target hardware. Preparing the software generally requires more time than hardware. Therefore, the time developers are delayed in starting a project is wasted.

Entering into this equation is the need for sophistication in embedded systems brought on by more powerful microprocessors and increasingly complex applications. Most embedded developers are turning to off-the-shelf real-time kernels or operating systems to assist them in managing this complexity. Therefore, it is incumbent on providers of kernels and operating systems to ensure that their customers can productively use the time while target hardware is being developed.

Since the introduction of our first product, Nucleus RTX (a real-time kernel), Accelerated Technology has provided our customers with a version of the software that permitted them to develop applications on the PC. We ported the three target specific modules of the kernel to run natively on a PC host. This way, developers can use the Borland or Microsoft development tools and build their applications using Nucleus to run as MS-DOS executables (.EXE files). We followed this same strategy when we introduced Nucleus PLUS (a more advanced real-time kernel), Nucleus FILE (an MS-DOS compatible file system), Nucleus NET (our proprietary TCP/IP stack), and Nucleus EPILOGUE (a combination of our Nucleus PLUS kernel and Epilogue, Corp.'s TCP/IP stack). That is, all of these products can be used for development in a native PC environment before being executed on the target hardware.

The application-programming interface to the various embedded software products we offer is identical on the PC and the target versions of the software. After preliminary development on the PC, the user simply recompiles for the target environment. The same technology employed to make our software products available on MS-DOS has been duplicated so that

Nucleus PLUS (and eventually all of our embedded products) can run native Windows NT or Windows 95 processes. We call this product Nucleus MNT.

To provide TCP/IP services within Windows NT, Nucleus NET was ported to the Nucleus MNT environment. To facilitate communication between Nucleus MNT processes and other TCP/IP enabled devices (both those resident on Windows NT using WinSock and those external to the Windows NT environment on

an actual network) a virtual networking facility was developed. We call this product Nucleus VNET.

This paper describes the use of Nucleus MNT and Nucleus VNET and some of the design considerations that went into developing it.

Hardware peripherals can be simulated with standard Windows NT device drivers. This method was used when Nucleus VNET was developed.

BASIC DESIGN PRINCIPLES

Nucleus PLUS was originally designed to be portable to dissimilar CPU architectures. Isolating three modules of the system and dividing them into target dependent and independent portions facilitated portability. These three modules perform initialization, scheduling, and timer management functions.

For Nucleus MNT, these modules were ported to the Windows NT threads environment. The initialization module sets up some interrupt vectors for the timer and the terminal interface. The scheduling module employs the Windows NT thread model to manage the switching of tasks. The timer module processes a timer tick to facilitate the Nucleus PLUS task sleep, time slicing, time-out, and timer thread capabilities.

The most interesting aspect of Nucleus MNT is the scheduling mechanism. Nucleus MNT runs as a Windows NT process. All Nucleus tasks are created as Windows NT threads. They are all assigned equal Windows NT priorities. Nucleus MNT is responsible for the preemptive or time based scheduling of the tasks. The Windows NT thread facilities are simply used to perform the necessary context saving and restoring of tasks. Hardware peripherals can be simulated with standard Windows NT device drivers. This method was used when Nucleus VNET was developed.

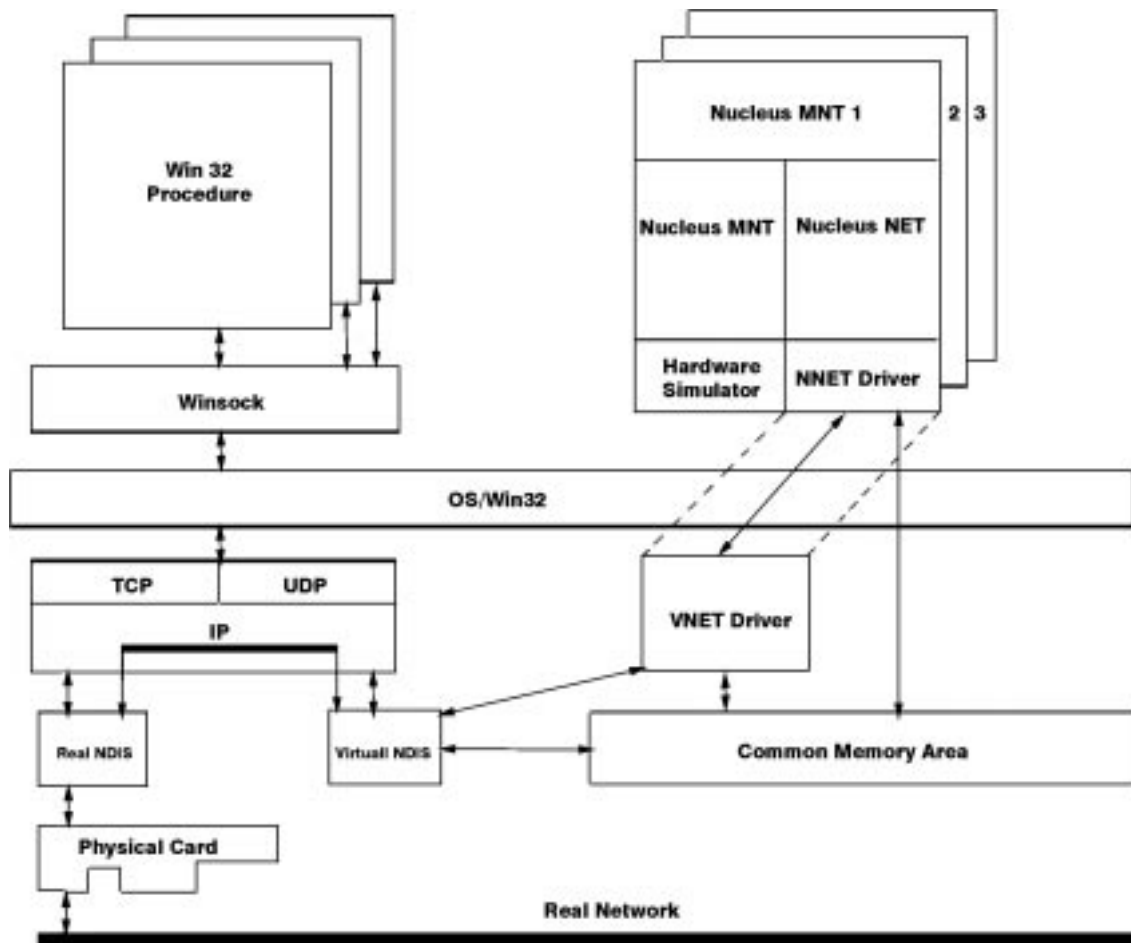


Figure 1.

NUCLEUS VNET

Nucleus VNET allows Nucleus MNT processes to communicate with each other via a shared memory area. That is, multiple versions of Nucleus MNT can be executed on an NT machine, each with its own IP address. These MNT processes can communicate with each other via TCP/IP as if they were independent embedded devices. Developers who use this technology have the ability to simulate a network on an NT machine. The same shared memory area is used to communicate with Windows NT applications via a virtual NDIS driver. If IP forwarding is enabled on the NT machine, then the virtual Nucleus devices can communicate, not only between themselves and the local NT workstation, but also with any remote device executing a TCP/IP protocol stack.

Perhaps the best way to describe how Nucleus VNET works is to describe each of the components that make up the complete system. The following discussion will make frequent reference to figure 1.

There will be at least one Nucleus MNT process executing. This MNT process includes not only the Nucleus MNT operating system, but also the Nucleus VNET code. Also included is the developer's application. The hardware simulator is that portion of the Nucleus PLUS operating system that is target depen-

dent. The small box labeled "NNET Driver" is the Nucleus NET device driver. In a typical port of Nucleus NET the NNET Driver would talk directly to hardware such as an Ethernet controller. In this port of Nucleus NET, the NNET Driver talks to the VNET Driver, an NT device driver.

The VNET driver acts as the interface between the Nucleus MNT process and the common memory area, and between the MNT process and the Virtual NDIS driver. The VNET driver is invoked from the application level via the NT DeviceIoControl service.

The Common Memory Area among other things, contains a list of buffers and many buffer rings. At initialization, all buffers are on the free ring, that is they are available for use. As buffers are used, they are placed on the buffer ring of the receiving MNT process. After the buffer is read it will be placed back on the free ring. The virtual NDIS driver also has a ring of buffers in the common memory area so that it can receive packets from the virtual network.

The Virtual NDIS driver also has access to the common memory area. Thus it can send packets to and receive packets from the virtual network devices. The virtual NDIS driver like any other NDIS driver is installed on the NT workstation. It has its own IP

Nucleus MNT was developed and designed to work with the Microsoft Visual C++ tool set.

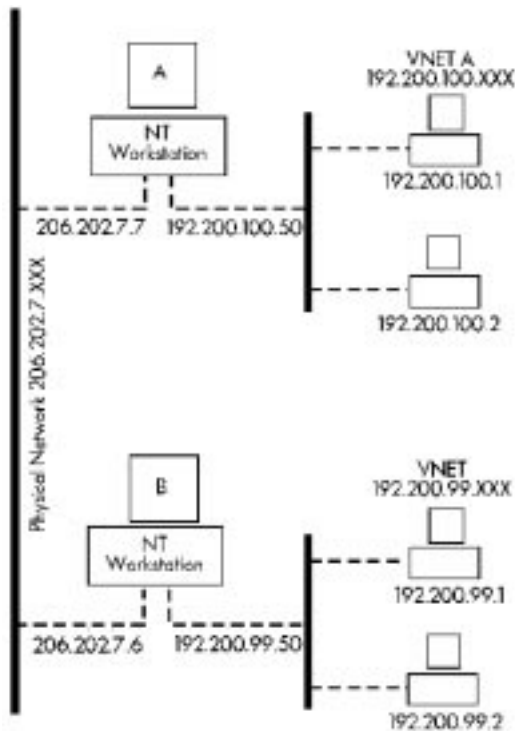


Figure 2.

address and is configurable through the NT's network settings just like an NDIS driver for a physical ethernet card.

Initialization

When the NT workstation is booted the Virtual NDIS driver is started. This driver is responsible for allocating the global memory area. The starting address is written to the system registry where it can be found by the other drivers and processes. The VNET Driver is then started manually by the user. The VNET driver must be started after the NDIS driver because it depends on the NDIS to allocate and initialize the common memory. After both drivers have been started, one or more Nucleus MNT applications can be executed.

Communication

Another way to explain Nucleus VNET is to describe the path a packet takes from source to destination. Assuming a packet has been passed down through the protocol stack, it will eventually be passed to the NNET driver for transmission. The NNET Driver communicates with the VNET Driver via NT's DeviceControl call.

To place a packet onto the network medium requires four steps. First, the NNET driver issues a DeviceControl call to the VNET driver requesting a buffer from the free ring. An offset into the common memory is returned. The start address of the common memory area was mapped into the process' address space during initialization. The offset is added to that address to get the starting address of the buffer. Secondly, the packet to be sent is copied into the buffer. Next, another DeviceControl call is made to place the

buffer back into the common memory area. This time the buffer is inserted into the ring of the receiving process. Finally, the receiving process must be notified that a packet is ready to be processed. Yet another DeviceControl call is made to the VNET driver. The VNET driver responds to this final DeviceControl command by completing an IO request that the receiving process is pending on.

The following steps are performed by the receiving MNT process. During initialization an NT thread is created that will receive packets. This thread acts as an ISR. When this thread executes, it issues a DeviceControl command to check the processes ring buffer for buffers. If there are no received buffers, then the thread suspends pending the completion of the IO request by the VNET driver. When a packet is transmitted to this MNT process, then the IO request will be completed. Once the IO request is completed, or if there was a buffer already present, then the ISR thread will issue another DeviceControl command to retrieve the buffer. The packet is copied from the receive buffer into local memory. Finally, a DeviceControl command is issued to the VNET driver to return the buffer to the free ring within the common memory area. The packet is passed up to the TCP/IP protocol stack.

Because the Nucleus VNET product requires two Windows NT device drivers it will execute on a Windows NT workstation only. Nucleus MNT will execute on either a Windows NT or Windows 95 workstation.

DEVELOPMENT

Nucleus MNT was developed and designed to work with the Microsoft Visual C++ tool set. Therefore, the Visual C++ integrated development environment including the editor; make/project capabilities, compiler, librarian, assembler, linker, and debugger are all available.

By using the project file supplied with Nucleus MNT, the developer is up-and-running almost immediately. The release files shipped with Nucleus MNT are loaded into a directory, the user adds the project to their Visual C++ environment, and the "Build" menu selection is invoked to produce a Windows NT

console application. The executable contains a demonstration program that exercises almost all of the Nucleus PLUS capabilities. The user can modify this program or replace it with tasks that they create to begin development of their project.

DEBUGGING

Since Nucleus MNT and the programs developed using it are true Windows NT applications, they can be debugged using the standard Visual C++ debugger. Other debugging aids supplied with Visual C++ (e.g., Spy) can also be employed to assist in the debugging effort.

Some of the peripheral capabilities of the target system can be simulated or substituted with PC capable equivalents.

SUMMARY

Anyone who has developed an embedded system is well aware of the difficulties associated with cross development. Download times are generally long, problems frequently exist with the target hardware or monitoring software, and the clutter of additional equipment compounds this problem. By removing the necessity of early hardware and the sometimes complicated cross development environments, Nucleus MNT can be used to prototype most of the C code that will be employed in the target system. Furthermore, with a little imagination and effort, some of the peripheral capabilities of the target system can be simulated or substituted with PC capable equivalents. This means that you, the developer, can begin your software efforts early on in the project thus getting your product to market faster.

Nucleus PLUS was designed to be highly portable. It is this portability and the capabilities afforded by the Windows NT thread environment that made Nucleus MNT a reality. Since it is a native Windows NT application, Nucleus MNT programs can be developed using the standard Visual C++ tools set. As an added benefit, the powerful debugging facilities supplied with Visual C++ can be used to get your application in working order quickly.

Accelerated Technology has one of the most capable and complete sets of real-time embedded software products available. If you would like to place an order or have any questions about Nucleus MNT or any of these other products, please contact us by phone, fax, or e-mail. ■

Neil F. Henderson began his professional career in the United States Navy in 1975 where he was responsible for operating computerized radar systems. While serving as an instructor, he complet-

ed his BBA at National University in San Diego, California.

In 1983, Mr. Henderson accepted employment with Science Applications International Cooperation (SAIC), Comsystems Division as a Software Quality Assurance Engineer. By learning the CMS-2 computer programming language, he was prepared for his next position at Digital Wizards, Inc. in 1984.

At DWI, Mr. Henderson developed computer programs for U.S. Navy shipboard database and communication systems. During his tenure at DWI, he was transferred to the Mississippi gulf coast to participate in a contract as an employee of DWI to Litton Data Systems Division. In 1986, Mr. Henderson completed his MS in Computer and Information Sciences at the University of South Alabama. He also participated in various work groups responsible for selecting an embedded operating systems standard for the U.S. Navy. As a result, Mr. Henderson recognized the possibility of bidding on a U.S. Navy project to develop a standard embedded operating system product.

In August of 1990, Mr. Henderson co-founded Accelerated Technology, Inc. to develop, market, and sell an embedded operating system. In 1995, Mr. Henderson consolidated all operations in the company in Mobile, Alabama, and accepted the title of President of ATI. He currently maintains all operations in Mobile.

Recently, ATI opened new sales offices in California, the United Kingdom, France and Germany. The distribution network includes offices in Japan, Russia, Italy, Korea, Israel, Australia, and Taiwan. Accelerated Technology maintains an effort toward reaching customers in new regions and finding the best way to serve local embedded systems communities.

ADVERTISEMENT INDEX

CETIA	73	LYNX RTS	79
COMPUTER PRODUCTS	45	MEZZANINES	42
DIGITAL EQUIPMENT CORP.	69-91	MICROTEC RESEARCH	11
DSP 97	87	QNX SOFTWARE	27
ECHTZEIT 97	35-49	RTS 98	83
ENEA DATA	15	RTUSI	103
EONIC SYSTEMS	86	SYSTRAN CORP.	31
FORCE COMPUTERS	116	US SOFTWARE	99
GREEN HILLS SOFTWARE	23	VENTURCOM	39
GREENSPRING COMPUTERS	19	WIND RIVER SYSTEMS	2
INTEGRATED SYSTEMS INC.	115		

To reserve your advertisement, contact Alexander Teetaert at:

Tel: 32-2-520.55.77 or Fax: 32-2-520.83.09 or Email: info@realtime-info.be