

Hard Real-Time Control: Moving PLC Technology to the PC

Many manufacturers are turning to alternate solutions to replace their current proprietary PLC technology used on the factory floor. PC-based control is increasingly the method of choice. The benefits to using a PC-based control solution that incorporates flow chart programming include faster design cycles, lower down-time on the factory floor, ease of use through Windows-based solutions, and improved economies of scale.

For many manufacturers, these benefits alone are all the reasons they need to switch from their 20 year old PLCs to PC-based control. For manufacturers with discrete applications, however, their need for deterministic scan times as low as 1 millisecond requires a solution that can achieve hard real-time in the Windows environment. This white paper summarizes the need for hard real-time in discrete applications and how it can be achieved with a PC-based control solution.

The overwhelming benefits of personal computers and the Windows® operating environment in factory control is a trend that cannot be ignored. The economies of scale of the \$100 billion dollar PC industries far outweigh the comparatively tiny PLC industry. The R&D alone spent on accelerating PC technology is greater than the entire industrial controls market. And now, as in most other industries, the industrial control market is moving from expensive proprietary hardware to open PC hardware and software solutions causing most experts in the industry to agree: PCs are the control platform of the future.

The commercial PC industry, however, is not focused on real-time requirements for discrete manufacturing. One of the biggest fears control engineers have about replacing their PLC with a PC is giving up deterministic hard real-time control. Hard real-time control provides the ability to get fast, deterministic, repeatable scan times from the control engine. PLCs are, by their nature, hard real-time controllers that give predictable, repeatable results every scan. Understandably, PLC users demand the same level of hard real-time control that they have now as they move to PC-based control.

How does the PLC accomplish hard real-time control? A PLC is really nothing more than a microprocessor with a real-time operating system (RTOS) in proprietary form. The RTOS is the kernel of code that controls all operations and tasks run on the microprocessor. PLCs perform control by building the control engine around the RTOS, which gives the PLC its fast, deterministic response. It is the RTOS that controls the PLC I/O scan and logic control functions inside the PLC.

This same level of deterministic control can be attained on a PC by using the same type of hard real-time operating system used in a PLC—an RTOS with

capabilities beyond Windows NT®. Microsoft® defines a hard real-time operating system as one which "must, without fail, provide a response to some kind of event within a specified time window. The response must be predictable and independent of other activities undertaken by the operating system."¹ Microsoft goes on to explain that "under this definition, Microsoft Windows NT Workstation is not a hard real-time operating system."

Certainly, some slower process applications can withstand control engine hiccups of 30+ milliseconds while Windows services higher priority disk accesses or network communications. In these cases, soft-logic control with Windows-based engines may be acceptable (see Figure 1). In discrete applications, however, where scan times range from 1 to 50 milliseconds, the impact of the non-deterministic nature of Windows 3.1 and Windows NT is unacceptable. Stopping the control engine to run system tasks such as servicing the hard disk or Ethernet card could cause lost machine cycles, damaged equipment, or even injury.

Still, Windows offers too many advantages to be ignored. The ability to run Windows-based programming, operator interface, program monitoring and access to other Windows applications on the same PC that runs control is one of the key benefits of moving to PC-based control. The question remains: How does one gain all of the benefits of Windows running on the PC yet achieve the reliability required of a PLC? The answer lies in moving PLC RTOS technology to the PC.

Hard real-time PC-based controllers, like the Visual Logic Controller® (VLC®) by Steeplechase Software, marry the core of the PLC (a hard real-time operating system) with Windows. By running the PLC control engine as high priority tasks in a RTOS and Windows as a background task, control engineers get the best of both worlds: real-time control and the Windows operating environment running on the same PC.

With a hard real-time operating system at the core, all of the memory and CPU bandwidth required for the PLC function is set aside at boot-up. Windows loads after the control engine as low priority background tasks within the RTOS. The control engine is completely protected from Windows, which cannot access

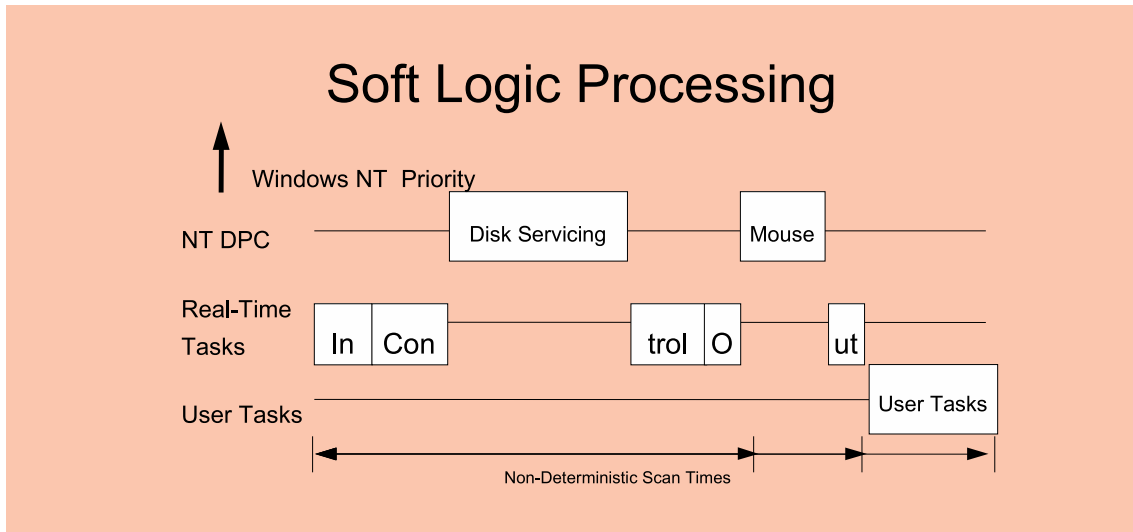


Figure 1. Soft Logic Control with a Windows-based Control Engine

the control engine memory or pre-empt the control engine. Windows can lock-up-or even crash-with no impact to the real-time control operation.

By using a hard real-time operating system for control, the PLC function can be separated and run as a higher priority task that is completely independent of Windows (see Figure 2). Windows can run applications best suited for non-real-time: data management, net-

work communications and graphical user interfaces. Control logic programming, operator interface, data acquisition and client/server networking can run on the same PC as real-time control logic without impacting control engine performance. Other Windows applications running on the same PC, while independent of the RTOS, can access its real-time control data. For example, shifts reports or batch recipes can be managed with Excel, Visual Basic can be used for custom

Ad Computer Products

PC BASED CONTROL

applications and SPC packages can be implemented to monitor production data for quality problems at the control station.

As a practical note, because the control engine always takes precedence, heavy control programs may impact the performance of Windows applications. Like every control application, sizing the controller (in this case the PC) to the application is important. One of the benefits of PC-based control is the range of PCs available to meet your application need. Also important to note: PCs can already significantly out-perform most PLCs. For example, Steeplechase's VLC running on a low-end 60 MHz Pentium processor performs Boolean logic 50 times faster than the leading PLC-integer manipulation 216 times faster. And many PC-based control applications consume only 5-15% of the CPU bandwidth for logic control, leaving 85-95% of the PC left over to run Windows.

With a PC-based control system that can provide all of the hard real-time control characteristics of a PLC, control engineers can look to PCs to meet their most demanding discrete control applications that may require deterministic, repeatable scan times as fast as 1 millisecond. By moving the capabilities of the PLC to the PC, hard real-time PC-based control can offer the best of both worlds: deterministic control and Windows functionality. The logic control engine can be combined with easy-to-use Windows programming tools and operator interfaces. The challenge is not trying to make Windows function as a PLC. The challenge is to move the PLC into the PC.

With most soft-logic control systems, the control engine is run as a high priority "real-time" task under Windows NT. "Real-time" tasks can be interrupted by deferred process calls (DPCs) which are used to service NT system functions such as disk accesses, network communications and mouse interrupts. DPCs can be triggered by lower priority user tasks such as opening files or starting applications. This phenomenon is known as "priority inversion"--when a user task can cause a higher priority task to wait.

Stated simply, user tasks or system functions can force the control engine to wait for a significant amount of time while the DPC is being serviced. In many slower process applications, these delays are insignificant to

the speed at which the process must run. For discrete applications, however, these delays can be unacceptable.

In a hard real-time system, the core of a PLC--a hard real-time operating system (RTOS)--is loaded first. The control engine runs as the highest priority task in the RTOS. All Windows functions run as the lowest priority task within the RTOS. The logic control engine always has priority and is completely protected from Windows, which cannot pre-empt real-time control.

For many critical discrete applications, deterministic control is required. User and system functions cannot pre-empt control. Hard real-time PC-control provides deterministic scan times as fast as 1 millisecond.

CONCLUSION

Discrete manufacturing applications can view PC-based control as a viable alternative to outmoded PLCs when that solution provides for the necessary deterministic scan times the applications demand. PC-based control solutions that provide hard real-time control guarantee that the control function runs as a higher priority task independent of Windows.

This benefits the control engineer by providing the hard real-time they need with an easy to use Windows environment running on a standard PC. The manufacturer benefits from the PCs economies of scale, its ease of use, its low maintenance and upkeep, and its ability to scale as the technology matures. A PC-based control solution that guarantees hard real-time enables a manufacturer to take advantage of the undeniable business benefits that results from using standard technology.

Michael Klein, President and CEO, founded Steeplechase Software in early 1993. Klein has 15 years of experience in manufacturing control and computer industries. He held various management, engineering, and sales positions at Allen-Bradley and Motorola Semiconductor. Klein is an honors graduate in Computer Engineering from Rochester Institute of Technology. He holds a Masters degree in Electrical Engineering from Arizona State University.

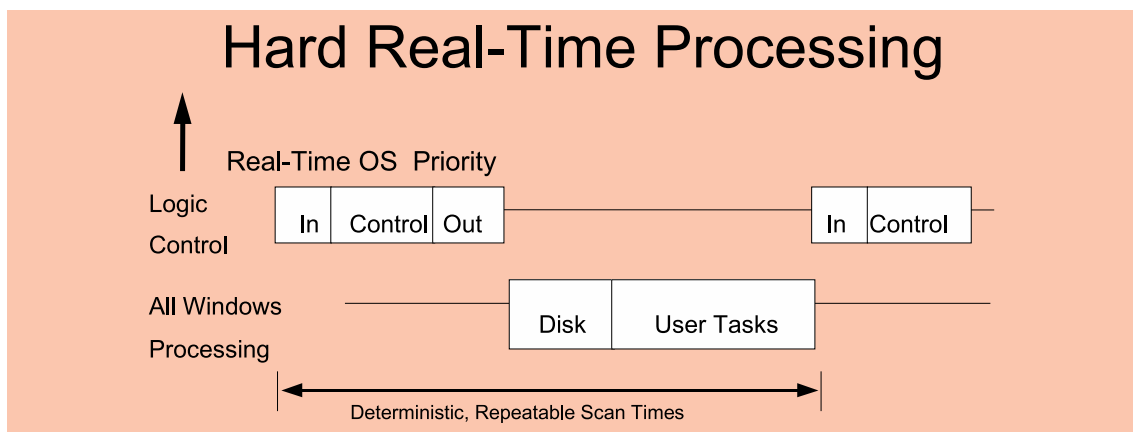


Figure 2. Hard Real-Time Control