

Beckhoff real-time kernels for DOS, Windows, Embedded OS and multi-core CPUs

Real-time processing – the basis for PC Control

Beckhoff employs Microsoft operating systems for its PC-based control technology. When used in industrial automation, the operating system must be extended at the lowest level for deterministic real-time behavior. The real-time kernels for MS-DOS and Windows developed by Beckhoff form the basis for this. As a Microsoft technology partner, Beckhoff obtains early access to new Windows technologies and is thus able to guarantee the operability of the new operating systems and their Embedded variants, as well as multi-core support for real-time applications.



Ramon Barth, director of system, HMI and real-time software development, Beckhoff Automation

The great advantage of PC-based control lies in the possibility to process general IT tasks and functions in parallel, since they must be executed in real-time. Real-time-capable execution means that the temporal start and end of a computing operation or a sequence of computing operations are repeatedly deterministic. In order to enable the coexistence of these different worlds, the standard PC for PC control is extended by real-time.

Generally available operating systems with significant market distribution are only conditionally real-time-capable, since this is not demanded in the higher software layers or application levels of a PC. However, the processing of hardware signals and interrupts also has to take place quickly and deterministically even in office and home PCs, so that events generated in quick succession by the hardware are not lost. For that reason, particular attention is paid to deterministic processing and short response times even at the lowest level in general PC operating systems. This is where a real-time extension for a general operating system comes in.

The central starting point for the temporal allocation of PC computing capacity and the control of the real-time functions of PC control is the timer interrupt. The source of this time-controlled interrupt depends on the development history of the PC architecture, wherein the timer that triggered the real-time interrupt under MS-DOS is also still available on current PC hardware. The use of other interrupt sources is more efficient nowadays, however.

As the name implies, the timer interrupt interrupts the CPU's current instruction processing after a configurable length of time and branches via the interrupt vector table to the specific code for the interrupt concerned. This ensures that real-time code is cyclically processed. The precisely maintained length of time between two interrupts is the basis for the quality of the real-time functionality of the PC control system. With each interrupt it is optionally possible to accomplish a context change between two different processing tasks via a scheduler.

PC Control – Real-time under MS-DOS

MS-DOS or PC-DOS (simply referred to as DOS from now on) is a "low level" operating system from today's point of view. With the help of the PC-BIOS, DOS (Disk Operating System) mainly manages the PC's hardware resources, such as mass storage devices and interfaces to connected devices. As a "single task" operating system, DOS can only start one program at a time. If another program is to be executed, the one currently being processed must be terminated first. In order to make background functions possible despite that, TSR (Terminate Stay Resident) provides a "back door" to leave

small auxiliary programs in memory after terminating the program. These TSR programs can be brought to "quasi life" using interrupts. Examples of this are PC remote control programs, which allow access to a DOS-based computer by modem.

Beckhoff developed its first DOS-based software PLC solution in 1988 under the designation S1000. The further development for software-based PLC/NC/CNC with a user interface in one program followed in 1993 with the S2000 software.

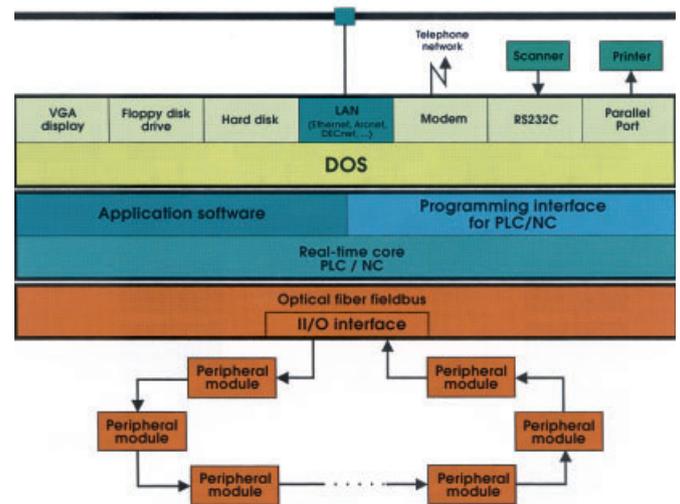


Fig. 1: PC Control – Real-time under MS-DOS

The S2000 software distinguished between three task levels: the NC task with the highest priority, followed by the PLC task and the foreground task and the user interface, which was executed in the remaining available computing time of the PC. The changeover between the task levels was accomplished via a scheduler, which was called by the timer interrupt in adjustable integral multiples of one millisecond. The NC function had priority over the freely programmable PLC function. The PLC consisted of one task, which could be programmed by the user in an S5-IL dialect. The sensors and actuators of the controlled machine were read in or output via Beckhoff's Lightbus network.

The S2000 foreground task consisted of cooperative multitasking, which made user inputs in parallel to background functions possible, thus removing

the single-task limitation of the DOS operating system. The visual display of the user interface was implemented by textual graphics with 80 x 25 characters and an ASCII character set. DOS was used for the administration of the PC operating equipment and essentially for data storage in mass storage devices.

PC Control on the basis of Microsoft Windows

With the further development of general PC technology with higher processing power and full graphic displays, extensive operating system extensions became necessary in order to use the resulting new solutions. A graphical user interface for DOS was initially developed on the PC in the form of Windows, which enjoyed increasing popularity from version 3.1 onwards. Windows 95 used 32-bit processors and introduced pre-emptive multitasking as well as a modern user surface.

In Windows NT, Microsoft developed a fundamentally new operating system for server and workstation systems, initially for the professional field. Due to the higher stability of Windows NT in comparison with Windows 95 (which still relied internally on MS-DOS) NT enjoyed greater acceptance than Windows 95 in the industrial environment. Windows NT then also formed the basis for TwinCAT, the first Windows-based PC control solution from Beckhoff. In TwinCAT (The Windows Control and Automation Technology), a completely new automation system was developed on basis of the PC Control concept that replaces conventional PLC and NC/CNC controllers as well as operating devices. The hardware components familiar from conventional automation technology (PLC, NC, axis cards, etc.) are implemented in TwinCAT as so-called software devices. TwinCAT can be modularly extended depending on needs without having to change existing software structures.

The TwinCAT software devices can be distributed to different components, depending on requirements: The TwinCAT PLC programs can run both on Beckhoff PCs and on Bus Terminal Controllers. A so-called message router manages and distributes all messages in the system via TCP/IP connections (PC systems) or via serial interfaces and fieldbuses (Bus Terminal Controllers).

Real-time functionality under Windows

The integration of a real-time extension into a complex operating system such as Windows NT (Windows 2000, Windows XP, Windows Vista and Windows 7 are very closely related in this regard) is considerably more difficult than was the case with MS-DOS. Windows is inherently capable of multitasking, so that many processes can be executed in parallel. Therefore, hardware access can take place only in the protected kernel mode. Drivers and the operating system expect a defined temporal behavior of the PC hardware. In order to enable the most transparent possible integration of real-time functionality in Windows without at the same time changing the operating system or other standard software components, a timer implementation was developed for TwinCAT and patented by Beckhoff: the "double-tick."

"Double-tick" is the name given to the system characteristic of triggering an interrupt each time when switching from non-real-time mode to real-time mode and back again. When switching to real-time mode, the interrupt is used at the same time to activate the TwinCAT scheduler. The active, deterministic switching back to non-real-time mode after an adjustable length of time guarantees not only that Windows is given sufficient computing time by the respective CPU, but also that the necessary response times for certain hardware functions, such as modem, network or USB, are adhered to.

The TwinCAT real-time mode is interrupted in normal mode only by the double-tick interrupt, which activates the scheduler of the real-time kernel and switches back to Windows if necessary. Exceptions to this, however, are the NMI (Non Maskable Interrupt), which is triggered by intolerable hardware errors, and the SMI (System Management Interrupt). In exceptional cases, however, these events can also be prevented by the configuration of TwinCAT. All regular interrupts necessary for the operation of the PC system are allowed in non-real-time mode and are also processed there. Under the conditions described, TwinCAT can ensure the necessary real-time functionality without disturbing the operation of the PC system.

In contrast to the S2000 software, TwinCAT has a modular structure and offers greater automation functionality. In order to support this modularity and finer granularity in the timing and prioritization, a real-time core was developed for TwinCAT that allows 64 task priorities to be assigned, which are then executed in the real-time mode. Beyond that the real-time core offers functions for the synchronization of tasks and CPUs and for intertask communication. The pre-emptive scheduler ensures that the task with the highest priority is started at the desired time and runs to its end. The tasks with a lower priority share the remaining computing time according to their rank.

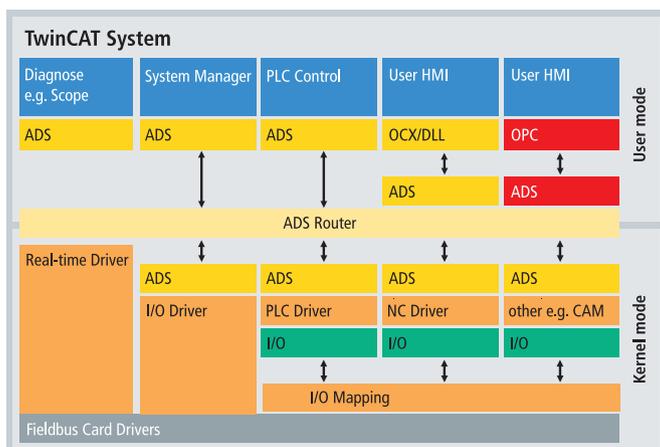


Fig. 2: The hardware components familiar from conventional automation technology (PLC, NC, axis cards etc.) are implemented in TwinCAT as so-called software devices. TwinCAT can thus be modularly extended depending on needs without having to change existing software structures.


Windows CE


Fig. 3: The CX9000 Embedded PC series integrates PC-based control technology in a compact bus terminal housing with Windows CE operating system.

Windows CE – PC Control scales itself for Embedded systems

Increasingly, powerful PC systems with ever more powerful operating systems and convenient user interfaces are frequently oversized for simple automation applications. In Windows CE, Microsoft has developed an operating system which is compatible to desktop Windows and can be finely scaled in terms of size and function. Originally planned for use mainly in mobile devices, Windows CE has been real-time-capable since version 3.0 and is used increasingly in the industrial environment.

Due to the relationship of Windows CE with the “large” versions of Windows, it was possible to adapt “TwinCAT CE” to “TwinCAT XP” with compatible source code. This means that all current and future TwinCAT functions are also available under Windows CE. The only restrictions are those due to the available hardware platforms.

TwinCAT CE uses the native real-time capability of Windows CE, extended by a fine-granular timer (resolution < 1 ms) on Beckhoff Embedded PCs. Since TwinCAT has mapped all real-time functions on Windows CE, TwinCAT real-time applications such as software PLC and Motion Control can coexist with real-time applications of a different origin.

Since version 3.0 of Windows CE, released in 2000, the operating system has been continuously developed, but always remains at least one step behind the desktop Windows. The current version of Windows CE is Windows Embedded Compact 7. It now supports SMP (symmetric multi-processing) for the first time on multiprocessor systems. TwinCAT CE 3 will also use and support this function of the operating system.

Multi-core – PC Control processing power multiplies itself

In addition to dual core CPUs, quad or octuple core units are now also available at a reasonable cost. This development benefits software-based automation solutions because they are able to distribute tasks depending on the number of available CPU cores. In other words: functional units such as HMI, PLC control, PLC runtime and NC can be distributed to dedicated cores with less effort than has been the case until now.

Beckhoff facilitates the utilization of multi-core systems through corresponding configuration and diagnostics tools. For example, the “TwinCAT System Manager” enables monitoring of real-time task runtimes and manual configuration of priorities or task sequences. Tasks can be allocated statically to a particular core via configurable core affinities. Ready-made profiles can be used to mimic conventional classification into PLC and NC runtime systems. In the development of real-time or PLC applications in the TwinCAT system environment, the switch from single to dual-core systems is taking place seamlessly since TwinCAT uses only one core here. However, the transition from the single or dual-core to the multi-core system will also be a fluid one, because TwinCAT can release several cores so that the available computing power can be used. The real-time runtime environment continues to use only one CPU so that existing PLC projects can be used directly without losing any of their benefits. A special characteristic of the software generation, TwinCAT 3, is that each used core sets the optimum system clock depending on the cycle time of its task. This saves computing power and reduces the energy demand.

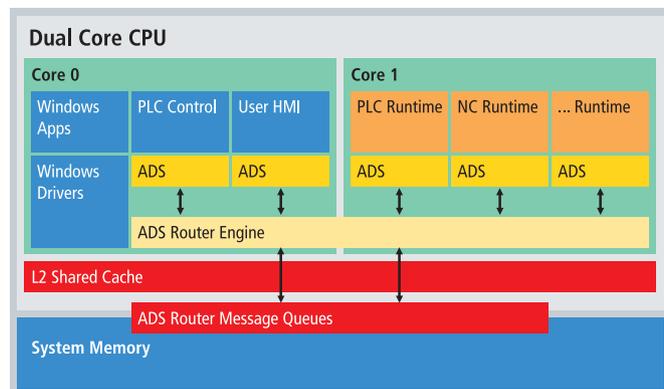


Fig. 4: Dual-core CPU. Due to the use of multi-core systems, functional units (e.g. PLC and NC runtimes, HMI) are distributed to individual computer cores.

As TwinCAT makes unused CPU time available for Windows applications, the Windows operating system sees two CPUs, of which one is running at part capacity. Windows applications built from several program threads can benefit from this. The Windows operating system distributes the application threads to the available CPUs, which physically run in parallel, so that the CPU hardware is optimally used. However, synchronization gaps are more likely to occur in the application of physical parallel processing than in the virtual parallel processing of threads. Here the CPU changes very quickly back and forth between various programs so that, although it looks like parallel processing, it actually takes place sequentially. In order to optimally use multi-core systems in the future, all applications must be divided in a modular way into threads or tasks as far as possible. Windows and TwinCAT can thus distribute the processing of program components optimally to the available

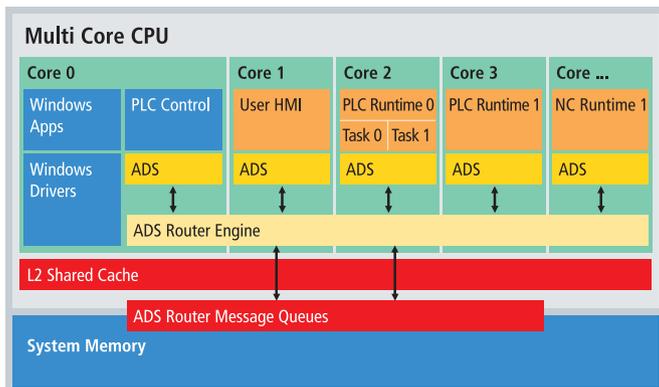


Fig. 5: Example of use for the load of a multi-core CPU

CPU cores. Monolithic programs will still continue to work, but they will only be able to use the existing computing capacity to an ever-decreasing degree. Due to the falling costs of dynamic RAM and the increasing memory requirements of resource-hungry applications such as HD video processing, new PCs are increasingly being offered with memory capacities of more than 4 GB. Since this is the theoretical limit of the address space for a 32-bit CPU (in reality approx. 3.5 GB is actually available), advanced PCs are increasingly based on 64-bit operating systems. 64-bit Windows utilizes a mode of modern x86 CPUs that enables 32-bit applications to be used unmodified in parallel with 64-bit applications. This operating mode developed by AMD therefore enables a smooth transition from 32-bit to 64-bit environments. AMD calls this processor mode 64-bit longmode, while Intel offers CPUs with these features under the name IA-32e mode. Both technologies are compatible with one another and offer 32-bit and 64-bit compatibility as sub-modes. In reference to the x86 history of the respective processors, Microsoft refers to this mode as x64 in current development tools.

Due to the processor characteristics mentioned, Windows can execute 32 or 64-bit applications in the 64-bit version, wherein 32-bit applications must accept slight speed losses, since all accesses to the 64-bit operating system must be converted from 32 to 64 bits. Like the operating system for x64, device drivers that run in kernel mode must be compiled and also signed. Signing is intended to enable unique determination of the origin of a device driver, thereby preventing harmful code from being loaded into the address space of the operating system.

Programs compiled for x64 work in a 64-bit address space and can access an additional eight "general purpose registers" with a 64-bit width. The registers that are familiar from x86 are also 64 bits wide. Segment registers are no longer required or have no purpose with the exception of the GS register for addressing of operating system structures. In 64-bit mode the features described above result in advantages or disadvantages for the performance of the overall system, depending on the application. On the one hand, more

memory and more registers are available; on the other hand, the compiled code is significantly larger, requiring more memory and cache. A general gain in performance can be achieved only if all features are consistently used, or if an application urgently needs a larger address space.

For automation applications the address space of a 32-bit system is usually adequate. An extended address space naturally results in a larger potential for applications such as Condition Monitoring or digital image analysis, which the PC can execute in addition to the tasks that it already has. Beckhoff refers to these technologies as Scientific Automation.

Optimum utilization of modern processor architectures

In addition to continuously increasing computing power, compatibility between CPU generations is a key feature of the x86 architecture. x86 processors of the latest generation are still able to execute programs that were compiled for the Intel® 8086 CPU dating back to 1978. For compatibility reasons many 32-bit applications still only use the instruction set and the features of an 8086 CPU. While even these programs benefit from increased speed with each new processor generation, the capacity of new processor architectures is utilized to a lesser and lesser extent. An additional complication is that current processors emulate certain compatibility functions with the aid of micro code, which may even result in slight performance losses. In these situations hyperthreading may offer a solution. It offers improved resource utilization through parallelization, provided the respective software supports it. However, in the medium term optimum utilization of the available computing power can only be achieved with new instruction sets. TwinCAT 3 was fundamentally revised with this in mind and can optionally be optimized for current CPU types. Among other things this applies to floating point calculations and utilization of the CPU cache.

After multi-core comes general parallel processing

The success of PC Control is based on the convergence of the technical achievements of the IT world with automation technology. Automation technology benefits here from the continuous development of hardware and software at reasonable costs. Technical progress in the development of new processors is most striking, but general software development is also producing new and interesting possibilities for use in automation technology. The current catchword here is AMP (Accelerated Massive Parallelism), an extension of the C++ programming language intended to enable better utilization of available computing capacity. With the aid of AMP, developers can in the future address every kind of resource in the PC system and distribute instructions to CPUs, GPUs and discrete graphic processors. In the future, the distribution of functions to other computers and virtual machines will also be supported.

However, this new technology must be provided with the necessary real-time capability in order to be suitable for use in automation technology. PC Control is on the right road here as well, with TwinCAT 3 and the integration of C/C++.