Many-core control for greater intelligence in the Smart Factory

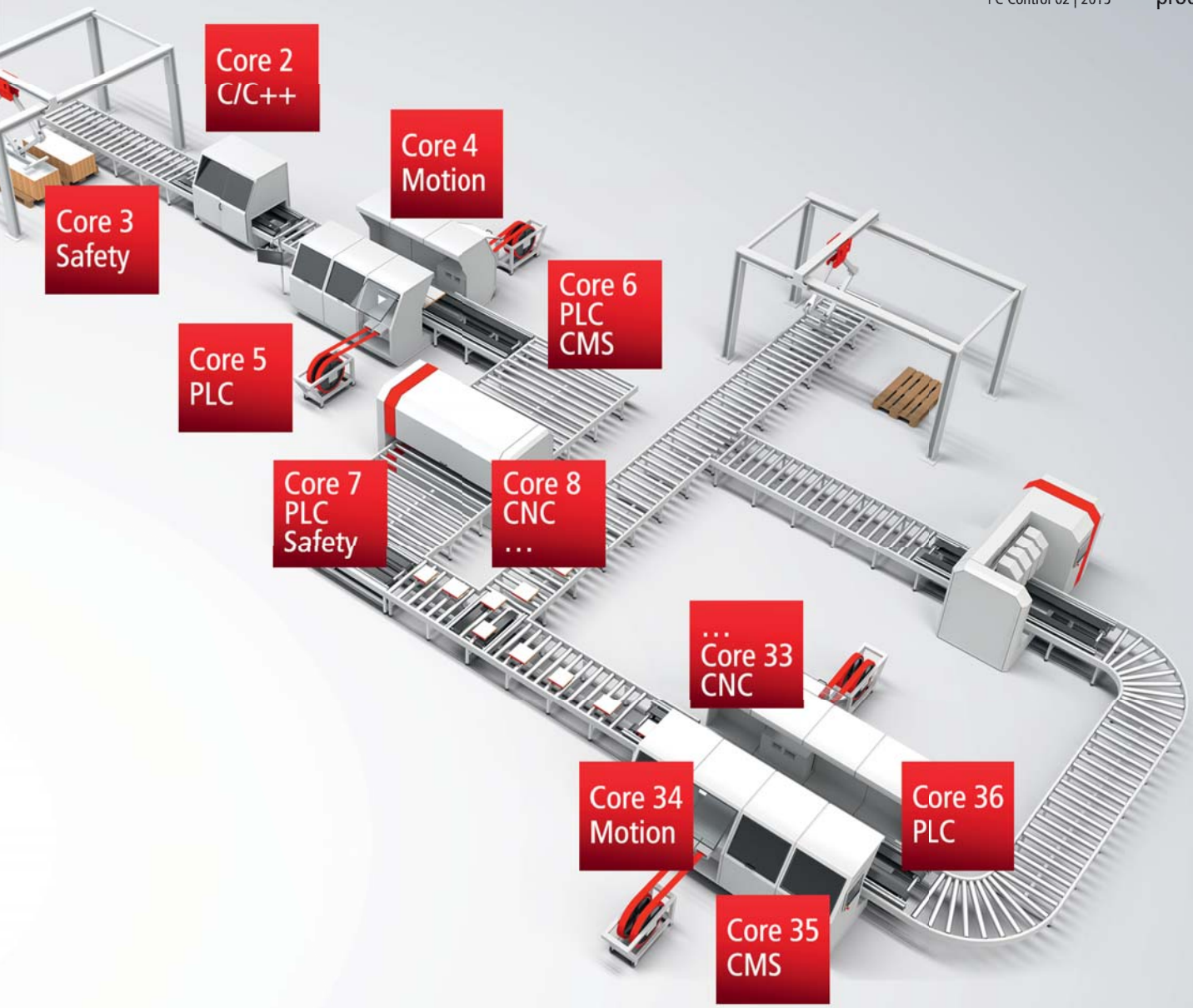
TwinCAT – how are functions assigned to the cores?

The Beckhoff philosophy centers on PC-based control technology. With ever more powerful PCs, it is possible to realize a central machine control system in which all PLC, motion, robotics, and CNC applications run on a single Industrial PC. Beckhoff uses the term "Scientific Automation" to describe the combination of conventional automation tasks with solutions from engineering science that go beyond the limits of conventional control. For example, it is now possible to integrate demanding applications such as image processing, measurement technology, and condition monitoring into standard control software. The goal is to gather data not only on the quality of manufactured products, but also to continuously monitor the current machine and equipment status. This is a prerequisite for fail-safe, cost-effective production.

Computing power fully leveraged with TwinCAT 3

The demand for computing power obviously increases as the complexity of an individual machine or a plant rises. Beckhoff offers a scalable range of CPUs – from ARM or Intel® Atom™-based processors for entry-level controllers to modern "Core i" series processors, to many-core server systems for high-end control applications. For example, the C6670 industrial server with 12, 24, or 36 physical cores offers abundant computing power for demanding control tasks in large production facilities. This many-core machine control system includes two Intel® Xeon® processors, each of which combine a number of cores in a single package. Each package has its own internal cache and memory. These systems therefore have two separate physical main memories, resulting in significantly increased access speed. For users, and therefore also for real-time applications, these two main memories appear as a single large memory. Due to their memory architecture, such systems are sometimes referred to as "Non-Uniform Memory Access" (NUMA) systems.

The current TwinCAT software version 3.1 can use up to 256 cores in a targeted manner. As a result, users have the complete range of latest generation processors available for automation applications. The number of cores and the corresponding computing power can be configured as required for running real-time applications. Such applications can specify cores for running Windows, as well as cores that are not used by Windows – so-called isolated cores. When using cores for Windows, the processor time is divided into real-time and Windows time. The proportion of real-time is limited by the "CpuLimit" parameter and can be set between 10 and 90 percent. Switching between real-time and Windows takes place cyclically with a freely selectable base time;



task cycle times are derived as multiples of the base time. Isolated cores do not have to switch between real-time and Windows, so that the full power of the processor is available for real-time applications. The use of isolated cores is recommended for fast tasks with cycle times of 100 µs or less. When using NUMA systems with many real-time cores, it makes sense to isolate a complete processor, so that the cache of the isolated processor is exclusively available for real-time operations.

From TwinCAT modules to the cores

In TwinCAT, individual automation tasks are realized as modules. Modules may be for motion control, PLC or C++ applications, for example. These modules are assigned to individual tasks of the TwinCAT system and executed cyclically based on a user-defined sampling rate, i.e. the cycle time. The tasks are then distributed to the available real-time cores, and typically several tasks are performed on one core. Therefore, the tasks are assigned priorities to define the execution sequence; priorities control the execution sequence of tasks. The higher the priority, the more accurately a task is executed. Processing of tasks with lower priorities can be interrupted by tasks with higher priorities. As a general rule: "The shorter the cycle time, the higher the priority."

As an example, Figure 1 shows the execution sequence of the tasks for a typical motion control application with PLC and C++ software components. The real-time proportion is limited to 90 percent of the base period (here 200 µs), so that Windows (OS) is always allocated at least 10 percent of the computing capacity. This ensures that the Windows operating system is always guaranteed to be active for a minimum time within a base time. Motion Control NC PTP is divided into an SAF task (German: "Satz-Ausführungs-Task", English: "block execution task") with a cycle time of 200 µs and computing time of 30 µs and an SVB task (German: "Satz-Vorbereitungs-Task", English: "block preparation task") with a cycle time of 400 µs and a computing time of 100 µs. The C++ task and the PLC task both run with a cycle time of 200 µs and a computing time of 40 µs and 60 µs respectively. To comply with the cycle time, the computing time obviously has to be shorter than the required cycle time, which is the case in this example. The tasks are executed according to the priorities 1, 2, 3 and 4 in the sequence SAF, C++, SPS and SVB, as indicated. All tasks are activated at time 0 µs, and the TwinCAT real-time scheduler processes them sequentially, based on the specified priorities. The tasks SAF, SPS, and C++ have a cycle time of 200 µs and are therefore reactivated at 200 µs. At this point in time, the SVB task has not yet been completely processed. The tasks

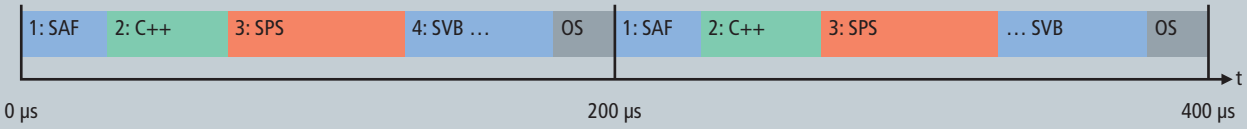


Figure 1: Single-core with base time 200 µs and 90 percent real-time limit.

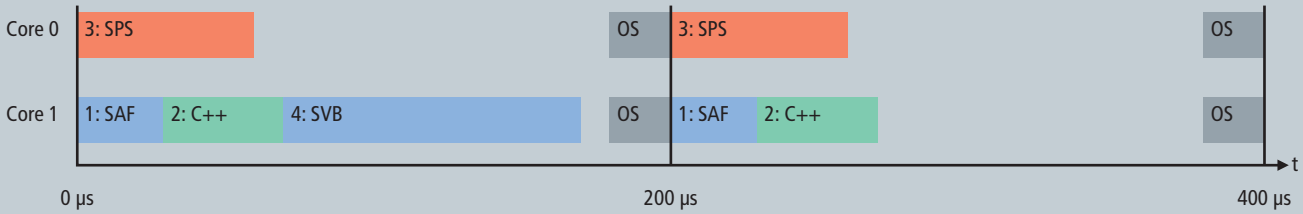


Figure 2: Multi-core with base time 200 µs and 90 percent real-time limit.

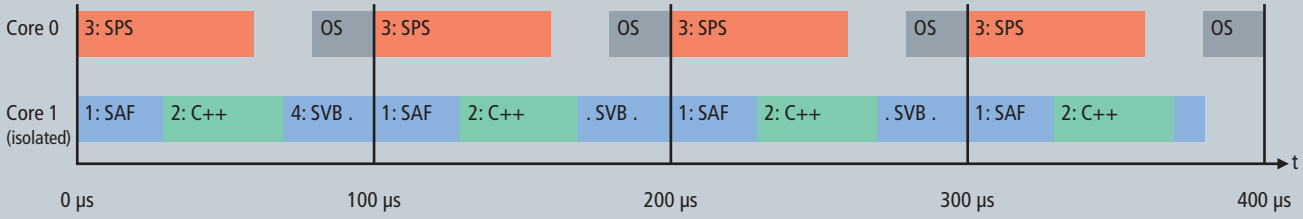


Figure 3: Multi-core with base time 100 µs, 80 percent real-time limit, and core isolation.

with shorter cycle times, which were assigned priorities 1 to 3, are prioritized over the SVB task, which has a priority of 4. This ensures that they comply with the cycle time, as in the previous cycle, and are not "held up" by the SVB task. Processing of the SVB task then continues. If a task repeatedly misses its activation, a cycle timeout error (Exceed) is triggered. However, the task reporting a timeout may not be responsible for the timeout. It is therefore always advisable to examine the task runtimes of the higher-priority tasks on the core.

In this example, the computing power of the Industrial PC is fully utilized. To extend the application, it is possible to distribute it to two cores. Figure 2 shows a possible distribution. In this configuration, all tasks except the PLC task are assigned to a separate core. Note that in the single-core configuration, the PLC task is executed after SAF and C++. Since each core calculates the execution sequence locally for the tasks assigned to it, the PLC starts with the SAF task on the second core in parallel. Thanks to the additional computing power, the SVB task is calculated within the first cycle, making more computing time available for additional tasks on both cores. This can be used either for an extension of the existing application or for other modules.

Alternatively, the additional computing power can be used to increase the sampling rate for the existing application. In this case, the cycle time of one or more tasks should be reduced. Such an example is shown in Figure 3. On both cores, the base time is halved to 100 µs; in addition, the second core is being isolated. The latter is indicated by the absence of the "OS" proportion in the execution sequence. On the first core, the length of a single Windows time remains unchanged at 20 µs, i.e. the real-time limit is 80 percent in this case. Therefore, 20 percent of the computing power of the first core is available for Windows. The cycle times of the SAF, C++, and PLC tasks are reduced to 100 µs. As a result, the sampling rate of these tasks doubles. Although the SVB task is now interrupted more frequently, the calculations for all tasks are completed before their next activation. In such an approach, the available bandwidth on the connected fieldbus must be adequately dimensioned, because the number of fieldbus telegrams per unit of time doubles, resulting in increased overall fieldbus load.

A distribution to more than one core makes sense, for example, in cases with many computationally intensive instances of a module, which can be calculated independently. One such application example would be condition monitoring.

In principle, each module does not need to be assigned a separate task, as in the example above. Depending on the computational requirements of a single module, several modules can be assigned to a task. The resulting task runtime must not exceed the required cycle time of a module. Otherwise, further modules have to be assigned to an additional task and executed on a separate core. Naturally, the behavior greatly depends on the respective application. In any case, it is advisable to commission the system step-by-step. If modules with different cycle times must be processed, they should always be assigned separate tasks with suitable configuration.

Summary

These days, any gain in computing power is increasingly achieved by increasing the number of cores per processor instead of a significant increase in processor clock speed. TwinCAT 3.1 supports this trend and enables the use of single-core systems, multi-core systems, and indeed many-core or NUMA systems from the server segment. The increased computing power can be used to migrate existing systems with several Industrial PCs to a single PC, or to expand and increase the control quality of an individual Industrial PC. This article describes the reduction of task cycle times through distribution across a multi-core system, based on a typical motion control application, as an example. Another example is Scientific Automation, which can complement existing systems with sophisticated measuring or image processing applications. This enables enhanced system monitoring or optimization during runtime. Beckhoff continuously develops this technology further and in this way enables customers to use cutting-edge Industrial PC systems for automation to increase performance and to ensure higher availability while retaining the benefits of centralized control systems.

Author: Dr. Henning Zabel, Real-Time Software Development, Beckhoff

Further information:
www.beckhoff.com/TwinCAT3
www.beckhoff.com/many-core-control