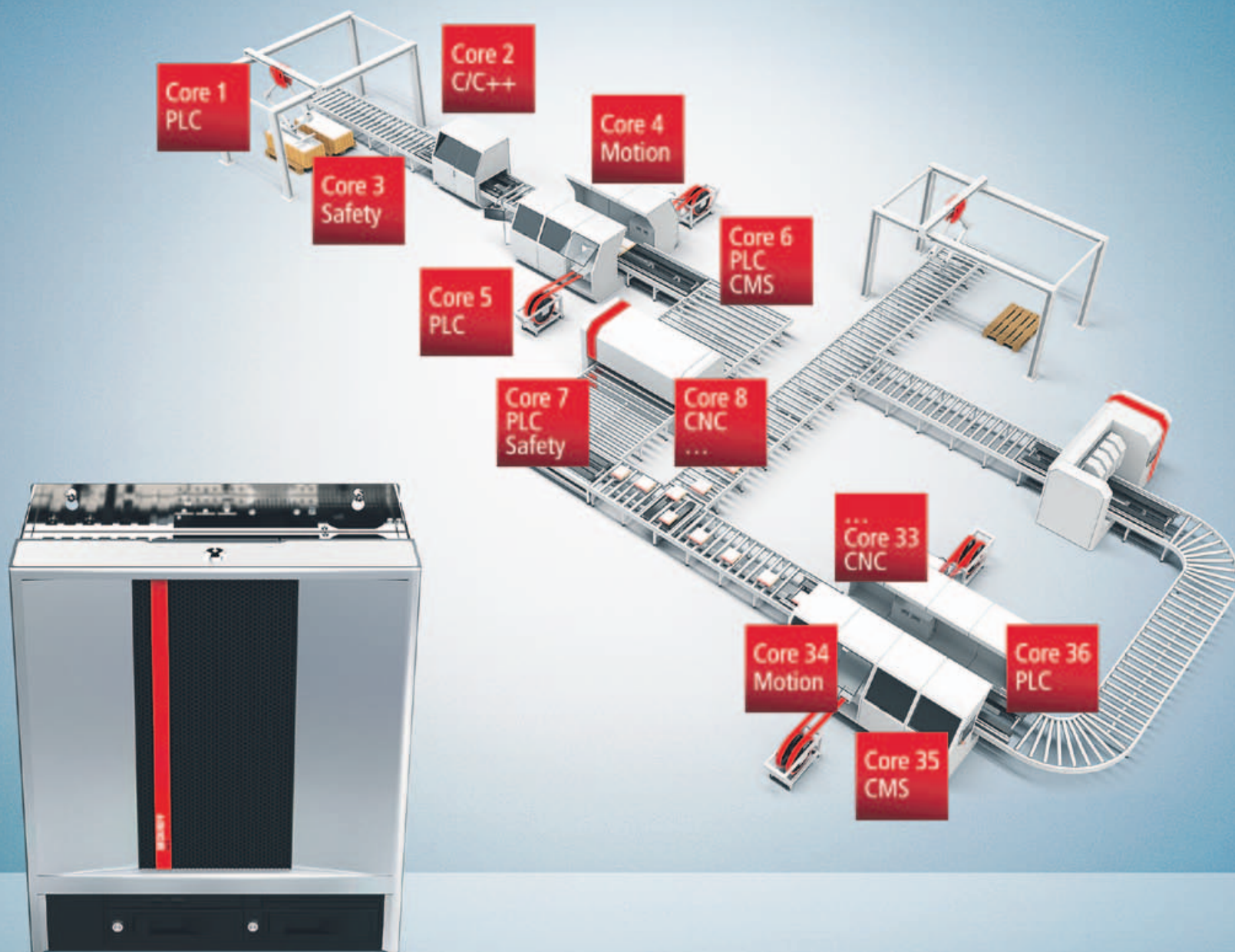


A *Computer &* AUTOMATION

Fachmedium der Automatisierungstechnik



TITEL: **DIE KERNFRAGE**

39 Fernwartung
**Sicherheit auf dem
Prüfstand**

46 Robotik
Schutzzaun adé

56 Sensorik
**Alternative zur
Lichtschranke**

Dr. Henning Zabel

Die Kernfrage

Der Zugewinn an Rechenleistung im PC-Bereich wird heute verstärkt erreicht durch eine höhere Anzahl an Kernen pro Prozessor anstelle einer signifikanten Steigerung des Prozessortaktes. Damit stellt sich zwangsläufig die Frage: Wie lassen sich die Automatisierungsfunktionen auf die unterstützten Cores – im Fall der aktuellen Twincat-Software von Beckhoff sind das immerhin 256 – sinnvoll verteilen?

Mit immer leistungsfähigerer Rechnertechnologie ist es heute möglich, eine zentrale Maschinensteuerung zu realisieren, bei der alle SPS-, Motion-, Robotik- und CNC-Applikationen auf einem Industrie-PC ausgeführt werden. Dabei lassen sich auch komplexe Anwendungen wie Messtechnik, Condition Monitoring und Bildverarbeitung in klassische Steuerungssoftware einbinden. Für diese Integration ingenieurwissenschaftlicher Erkenntnisse in die Automatisierungssoftware, wie sie an Universitäten gelehrt werden und die über den Rahmen der klassischen Steuerung hinausgehen, hat

Beckhoff den Begriff ‚Scientific Automation‘ definiert. Letztendliches Ziel dabei ist es, neben der Qualitätsüberwachung der Produkte den aktuellen Zustand der Maschine laufend zu erfassen und zu überprüfen. Dies ist Voraussetzung für eine ausfallsichere und kosteneffiziente Produktion.

Für die Umsetzung derartiger Konzepte steht dem Anwender heute ein breites beziehungsweise skalierbares Spektrum an CPUs zur Verfügung: von ARM- oder Intel-Atom-basierten Prozessoren für Kleinststeuerungen über moderne Core-i-Prozessoren bis hin zu Many-Core-Systemen aus dem

Serverbereich wie dem Industrie-Server C6670, der 12, 24 oder 36 physikalischen Kerne zur Verfügung stellt. In dieser Maschinensteuerung sind zwei Intel-Xeon-Prozessoren verbaut, die jeweils eine gewisse Anzahl Kerne auf einem Package vereinen. Jedes Package besitzt einen eigenen internen Cache und eigenen Speicher. Physikalisch verfügen diese Systeme daher über zwei getrennte Hauptspeicher, was die Zugriffsgeschwindigkeit deutlich erhöht. Für den Anwender und damit auch die Echtzeit-Anwendungen stellen sich diese zwei Hauptspeicher jedoch wie ein gemeinsamer großer Speicher dar. Aufgrund der Speicherarchitektur bezeichnet man solche Systeme auch als ‚Non-Uniform Memory Access‘ oder kurz als NUMA-Systeme.

Mit der Hardware allein beziehungsweise mit einer Vielzahl an Rechnerkernen ist es allerdings nicht getan. Entscheidend ist, dass und vor allem wie die Automatisierungssoftware derartige Konzepte unterstützt. Im Beispiel der aktuellen Twincat-Version 3.1 wird die erforderliche Anzahl an Kernen je nach Bedarf an Rechenleistung für die Aus-



(Bilder: Beckhoff)

führung von Echtzeit-Anwendungen konfiguriert. Dabei lassen sich sowohl Windows-Kerne als auch von Windows nicht genutzte Kerne – die sogenannten isolierten Kerne – für Echtzeit-Anwendungen nutzen. Bei der Verwendung von Windows-Kernen erfolgt eine Unterteilung der Prozessorzeit in Echtzeit und Windows-Zeit. Der Anteil der Echtzeit wird durch das ‚CpuLimit‘ begrenzt.

Der Wechsel zwischen Echtzeit und Windows erfolgt zyklisch in einer frei wählbaren Basiszeit. Von letzterer leiten sich die Zykluszeiten der Tasks als ein Vielfaches ab. Bei isolierten Kernen ist kein Wechsel zwischen Echtzeit und Windows notwendig, so dass hier die volle Rechenleistung des Prozessors für die Echtzeit-Anwendungen zur Verfügung steht. Für schnelle Tasks mit Zykluszeiten von 100 µs und weniger empfiehlt sich die Verwendung von isolierten Kernen. Bei der Verwendung von NUMA-Systemen mit vielen Echtzeit-Kernen ist zudem die Isolierung eines kompletten Prozessors sinnvoll, da dadurch der Cache des isolierten Prozessors exklusiv für die Echtzeit zur Verfügung steht.

Vom Software-Modul auf die Kerne

Wie bereits erwähnt, werden in modernen Software-Umgebungen einzelne Automatisierungsaufgaben in Modulen realisiert – also zum Beispiel in Motion-Control-, SPS- oder auch C++-Applikationen. Den Modulen gilt es nun einzelne Tasks des Systems zuzuweisen, die diese zyklisch mit einer vom Anwender definierten Abtakte – der Zykluszeit – ausführen. Anschließend werden die Tasks auf die vorhandenen Echtzeit-Kerne verteilt, wobei ein Kern typischerweise mehrere Tasks ausführt. Um die Ablaufreihenfolge zu definieren, sind den Tasks Prioritäten zuzuordnen. Je höher die Priorität, desto exakter kommt eine Task zur Ausführung. Tasks mit niedrigen Prioritäten können zudem in ihrer Abarbeitung von Tasks mit höheren Prioritäten unterbrochen werden. In der Regel gilt: Je kürzer die Zykluszeit, desto höher die Priorität.

Bild 1 zeigt beispielhaft die Ablaufreihenfolge der Tasks einer typischen Motion-Control-Anwendung mit SPS und C++-Anteil. Der Echtzeit-Anteil

ist auf 90 % der Basiszeit – hier 200 µs – begrenzt, so dass Windows (OS) immer mindestens 10 % der Rechenkapazität zugeordnet wird. Damit ist das Windows-Betriebssystem innerhalb einer Basiszeit stets für eine minimale garantierte Zeit aktiv. Motion Control (Twincat NC PTP) ist aufgeteilt in SAF-Task (Satz-Ausführungs-Task) mit einer Zykluszeit von 200 µs und einer Rechenzeit von 30 µs und die SVB-Task (Satz-Vorbereitungs-Task) mit einer Zykluszeit von 400 µs und einer Rechenzeit von 100 µs. C++- und SPS-Task werden jeweils mit 200 µs ausgeführt und benötigen eine Rechenzeit von 40 beziehungsweise 60 µs.

Zum Einhalten der Zykluszeit muss die notwendige Rechenzeit kleiner sein als die geforderte Zykluszeit, was in diesem Beispiel der Fall ist. Die Tasks werden hier entsprechend der Prioritäten 1, 2, 3 und 4 in der gezeigten Reihenfolge SAF, C++, SPS und SVB ausgeführt. Alle Tasks werden zum Zeitpunkt 0 µs aktiviert und der Twincat-Echtzeit-Scheduler arbeitet diese anhand der Prioritäten der Reihenfolge nach ab. Die Tasks SAF, SPS und C++ haben eine Zykluszeit von 200 µs und werden daher zum Zeitpunkt 400 µs erneut aktiviert. Zu diesem Zeitpunkt ist die SVB-Task noch nicht vollständig abgearbeitet. Die Tasks mit den kürzeren Zykluszeiten erhalten hier Vorrang, indem diesen die Prioritäten 1 bis 3 zugeordnet wurden und somit gegenüber der Priorität 4 der SVB-Task den Vorrang erhalten. Nur so ist sichergestellt, dass sie wie im vorherigen Zyklus ihre Zykluszeit einhalten und nicht durch die SVB ‚aufgehalten‘ werden. Die Abarbeitung der SVB-Task wird anschließend fortgesetzt. Sollte eine Task wiederholt ihre Aktivierung verpassen, kommt es zur Zykluszeit-Überschreitung (Exceed). Eine Task, die eine Überschreitung meldet, muss jedoch nicht zwingenderweise dafür verantwortlich sein. In diesem Fall sollte immer ein Blick auf die Task-Laufzeiten der höher priorisierten Tasks auf dem Kern geworfen werden.

Bei diesem Beispiel ist die Rechenkapazität des Industrie-PC vollständig ausgenutzt. Um die Anwendung zu erweitern, bietet sich eine Verteilung auf zwei Kerne an. *Bild 2* zeigt eine mögli-

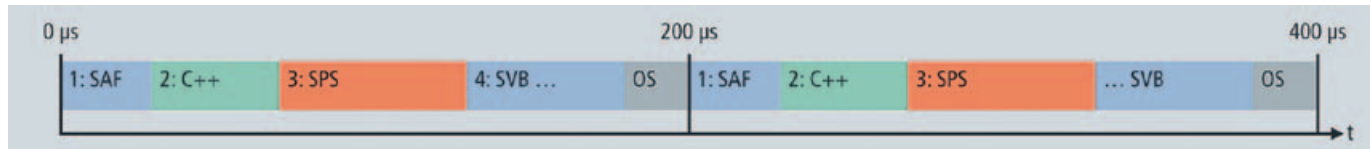


Bild 1. Single-Core mit Basiszeit 200 µs und Echtzeit-Limit 90 %.

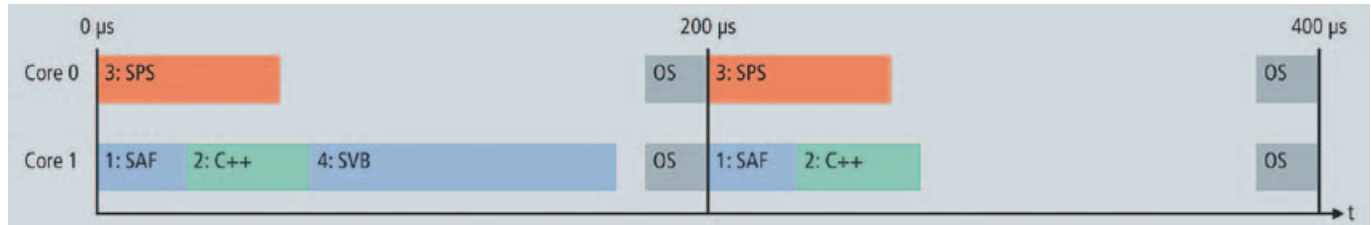


Bild 2. Multi-Core mit Basiszeit 200 µs und Echtzeit-Limit 90 %.

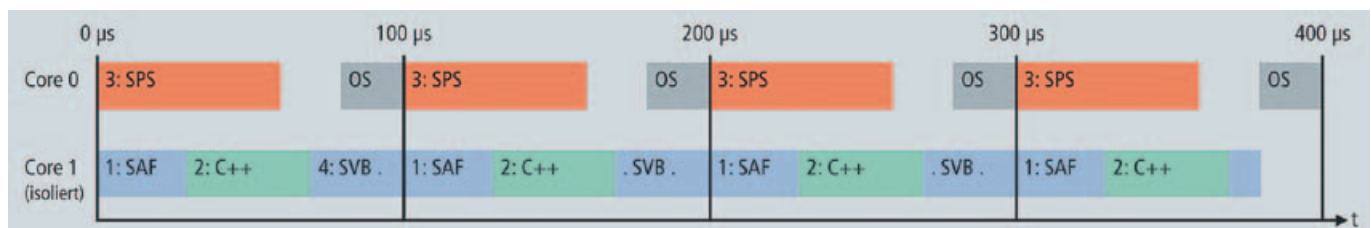


Bild 3. Multi-Core mit Basiszeit 100 µs, Echtzeit-Limit 80 % und Core-Isolation.

che Aufteilung. Bei dieser Konfiguration werden alle Tasks mit Ausnahme der SPS-Task einem separaten Kern zugeordnet. Man beachte, dass in der Single-Core-Konfiguration die SPS-Task nach der SAF und C++ zur Ausführung kommt. Da jeder Kern die Ablaufreihenfolge lokal für die ihm zugeordneten Tasks berechnet, startet die SPS hier parallel mit der SAF-Task auf dem zweiten Kern. Durch die zusätzliche Rechenleistung wird die SVB-Task innerhalb des ersten Zyklus berechnet, und es ist weitere Rechenzeit für zusätzliche Aufgaben auf beiden Kernen vorhanden. Diese lässt sich entweder für eine Erweiterung der bisherigen Anwendung nutzen oder für weitere Module.

Die Rechenleistung kann alternativ zur Erhöhung der Abtastrate der vorhandenen Anwendung dienen. Dazu ist die Zykluszeit einer oder mehrerer Tasks abzusenken. Ein solches Beispiel ist in Bild 3 dargestellt. Hier wird auf beiden Kernen die Basiszeit auf 100 µs halbiert und der zweite Kern wird zusätzlich isoliert. Letzteres ist am Fehlen des 'OS'-Anteils in der Ablaufreihenfolge zu erkennen. Auf dem ersten Kern bleibt die Länge einer einzelnen Windows-Zeit absolut unverändert bei 20 µs, das heißt das Echtzeit-Limit liegt hier bei 80 %. Dadurch steht Windows

20 % der Rechenleistung des ersten Kerns zur Verfügung. Die Zykluszeiten der SAF-, C++- und SPS-Task werden auf 100 µs gesenkt. Dadurch verdoppelt sich deren Abtastrate. Zwar kommt es hier zu einer häufigeren Unterbrechung der SVB-Task; dennoch werden alle Tasks vor ihrer nächsten Aktivierung durchgerechnet. Bei einem solchen Vorgehen muss die verfügbare Bandbreite auf dem angeschlossenen Feldbus allerdings ausreichend dimensioniert sein, denn es verdoppelt sich die Anzahl der Feldbus-Telegramme pro Zeiteinheit und damit erhöht sich die Gesamtauslastung des Feldbusses.

Eine Aufteilung auf mehrere Kerne ist zum Beispiel dann sinnvoll, wenn es viele rechenintensive Instanzen eines Modules gibt, die sich unabhängig voneinander berechnen lassen. Eine solche Anwendung wäre das Condition Monitoring. Prinzipiell muss nicht – wie im obigen Beispiel – jedem Modul eine eigene Task zugewiesen werden. Abhängig vom Rechenbedarf eines einzelnen Moduls lassen sich mehrere Module auch einer Task zuweisen. Dabei darf die entstehende Task-Laufzeit nicht die geforderte Zykluszeit eines Moduls überschreiten. Anderenfalls sind weitere Module einer zusätzlichen Task zuzuweisen und auf einem separaten Kern

auszuführen. Das Verhalten ist stark von der jeweiligen Anwendung abhängig. Insofern empfiehlt sich hier eine schrittweise Inbetriebnahme. Sind Module mit unterschiedlichen Zykluszeiten abzuarbeiten, sollten sie immer getrennten Tasks mit passender Konfiguration zugewiesen werden.

Zusammenfassend lässt sich festhalten: Der Zugewinn an Rechenleistung durch eine höhere Anzahl an Kernen pro Prozessor anstelle einer signifikanten Steigerung des Prozessortaktes kann zur Migration bestehender Anlagen mit mehreren Industrie-PCs auf einen einzelnen genutzt werden, aber ebenso zur Erweiterung und Erhöhung der Regelgüte eines einzelnen Industrie-PC selbst. Nicht zuletzt lassen sich auf diese Weise bestehende Anlagen mit komplexen messtechnischen Anwendungen oder Bildverarbeitung ergänzen und damit letztlich besser überwachen oder auch während der Laufzeit optimieren. *gh*



Dr. Henning Zabel

arbeitet im Bereich
Software-Entwicklung
Echtzeit bei Beckhoff.