

## BI140 - Neben- und Parallelläufigkeit mit C++

## BI140 - Concurrency and Parallelism using C++

---

Allgemeine Informationen	
<b>Modulkürzel oder Nummer</b>	BI140
<b>Eindeutige Bezeichnung</b>	NebenPLCPP-01-BA-M
<b>Modulverantwortlich(e)</b>	Prof. Dr. Manzke, Robert (robert.manzke@haw-kiel.de) Greve, Thomas (thomas.greve@haw-kiel.de)
<b>Lehrperson(en)</b>	Greve, Thomas (thomas.greve@haw-kiel.de)
<b>Wird angeboten zum</b>	Wintersemester 2025/26
<b>Moduldauer</b>	1 Fachsemester
<b>Angebotsfrequenz</b>	Regelmäßig
<b>Angebotsturnus</b>	In der Regel im Wintersemester
<b>Lehrsprache</b>	Deutsch
<b>Empfohlen für internationale Studierende</b>	Nein
<b>Ist als Wahlmodul auch für andere Studiengänge freigegeben (ggf. Interdisziplinäres Modulangebot - IDL)</b>	Ja

Studiengänge und Art des Moduls (gemäß Prüfungsordnung)
Studiengang: B.Eng. - E - Elektrotechnik (PO 2017, V3) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Eng. - E - Elektrotechnik (PO 2023, V4) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Eng. - Me (PO 2024) - Mechatronik (PO 2024, V5) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Eng. - Wing - Wirtschaftsingenieurwesen - Elektrotechnik (PO 2025, V2) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Eng. - Wing - Wirtschaftsingenieurwesen - Elektrotechnik (PO 2017, V1) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Sc. - INF - Informatik (PO 2021,V1) Modulart: Wahlmodul Fachsemester: 5

Kompetenzen / Lernergebnisse
<i>Kompetenzbereiche: Wissen und Verstehen; Einsatz, Anwendung und Erzeugung von Wissen; Kommunikation und Kooperation; Wissenschaftliches Selbstverständnis/Professionalität.</i>

<p>Kennenlernen der allgemeinen Konzepte:</p> <ul style="list-style-type: none"> <li>- Asynchronie in Form von Neben- und Parallelläufigkeit</li> <li>- Multitasking und Multithreading</li> </ul> <p>Vermittlung</p> <ul style="list-style-type: none"> <li>- der C++-Sprachkonstrukte, mit denen diese Konzepte realisiert werden können</li> <li>- von Bibliotheken, für solche Konstrukte, die (noch) nicht im C++-Standard enthalten sind.</li> </ul> <p>Die Teilnehmer setzen diese Konstrukte im Rahmen der Programmierübung anhand von Aufgaben ein.</p>
<p>Teilnehmer der Veranstaltung können:</p> <ul style="list-style-type: none"> <li>- einschätzen, bei welchen Aufgabenstellungen Neben- und Parallelläufigkeit sinnvoll eingesetzt werden kann (und bei welchen nicht)</li> <li>- entscheiden, welche der unterschiedlichen Sprachkonstrukte, die C++ für die Umsetzung bietet, den meisten Nutzen bieten</li> <li>- Neben- und Parallelläufigkeit einschliesslich ggf. erforderlicher Synchronisationsmechanismen in C++ programmieren</li> </ul>
<p>Durch die Projektarbeit im Team (2. Teil der Veranstaltung) können die Teilnehmer neben der Umsetzung des Gelernten ihre Fähigkeit trainieren:</p> <ul style="list-style-type: none"> <li>- nicht triviale softwaretechnische Sachverhalte zu diskutieren und so zu einem gemeinsamen Lösungsansatz für eine gestellte Aufgabe zu kommen</li> <li>- einen effizienten Weg für die Realisierung des Lösungsansatzes zu finden (Aufgabenteilung, Wiederverwendung)</li> </ul>
<p>Die Teilnehmer können Aufgabenstellungen, deren Realisierung Neben- oder Parallelläufigkeit voraussetzen, selbstständig identifizieren und lösen</p>

<b>Angaben zum Inhalt</b>	
<b>Lehrinhalte</b>	<p>'The free lunch is over' (Herb Sutter) und die Konsequenzen daraus: Effiziente Nutzung von Multicore-Systemen</p> <ul style="list-style-type: none"> <li>- Asynchronie in Form von Neben- und Parallelläufigkeit</li> <li>- Prozesse, Threads und Fibers/Coroutinen</li> <li>- Hardware-Threads vs OS-Threads vs Threads of Execution</li> <li>- Synchronisationsmechanismen und deren potentielle Probleme</li> <li>- Tasks vs Threads</li> </ul> <p>Umsetzung in C++:</p> <ul style="list-style-type: none"> <li>- Coroutinen</li> <li>- Möglichkeiten der Darstellung von Parallelläufigkeit: Überladungen von Funktionstemplates aus der Algorithm-Bibliothek des Standards vs <code>std::async</code> vs <code>std::thread</code></li> <li>- Ergebnisübertragung mit <code>std::promise</code> und <code>std::future</code></li> <li>- <code>std::packaged_task</code></li> <li>- Synchronisation durch Semaphoren, Mutexes, Locks, Barriers und Latches</li> <li>- Signalisierte Datenübertragung durch <code>std::condition_variable</code></li> <li>- <code>atomics</code></li> </ul> <p>Nutzung der boost-Bibliotheken:</p> <ul style="list-style-type: none"> <li>- <code>boost::process</code> für das Handling von Prozessen und die Interprozesskommunikation</li> <li>- <code>boost::asio::thread_pool</code> für eben diese</li> </ul> <p>Ausblick auf Konstrukte die erst mittelfristig im Standard enthalten sein werden:</p> <ul style="list-style-type: none"> <li>- Executors</li> <li>- Continuation</li> </ul>

<b>Literatur</b>	<p>--- Allgemeine Aspekte ---</p> <p>The Art of Concurrency          Clay Breshears          O'Reilly Media, Inc., 2009          ISBN: 978-0-596-52153-0</p> <p>Multicore-Software          Urs Gleim und Tobias Schuele          dpunkt.verlag, 2012          ISBN: 978-3-89864-758-8</p> <p>--- C++ - Spezifika ---</p> <p>The C++ Programming Language, 4th ed.          (Chapters 41+42, pp. 1191... 1251)          Bjarne Stroustrup          Addison-Wesley, 2013          ISBN: 978-0-321-56384-2</p> <p>C++ Concurrency in Action, 2nd ed.          Anthony Williams          Manning          ISBN: 978-1-617-29469-3</p> <p>C++ High Performance, 2nd ed.          Bjoern Andrist, Viktor Sehr          Packt          ISBN: 978-1-83921-654-1</p>
------------------	--

<b>Lehrformen der Lehrveranstaltungen</b>	
<b>Lehrform</b>	<b>SWS</b>
Lehrvortrag + Übung	2
Projekt	2

<b>Arbeitsaufwand</b>	
<b>Anzahl der SWS</b>	4 SWS
<b>Leistungspunkte</b>	5,00 Leistungspunkte
<b>Präsenzzeit</b>	48 Stunden
<b>Selbststudium</b>	102 Stunden

<b>Modulprüfungsleistung</b>	
<b>Voraussetzung für die Teilnahme an der Prüfung gemäß PO</b>	Die Termine mit Anwesenheitspflicht wurden wahrgenommen
<b>BI140 - Portfolioprfung</b>	Prüfungsform: Portfolioprfung Gewichtung: 100% wird angerechnet gem. § 11 Absatz 2 PVO: Nein Benotet: Ja Anmerkung: Bestehend aus Praesentation zum Semesterprojekt und abschliessendem Test. Details in der Vorlesung.

<b>Sonstiges</b>	
<b>Empfohlene Voraussetzungen</b>	Bestandenene Modulleistung: Programmieren in C++ (PIC)