

BI140 - Neben- und Parallelläufigkeit mit C++

BI140 - Concurrency and Parallelism using C++

Allgemeine Informationen	
Modulkürzel oder Nummer	BI140
Eindeutige Bezeichnung	NebenPLCPP-01-BA-M
Modulverantwortlich(e)	Prof. Dr. Manzke, Robert (robert.manzke@haw-kiel.de) Greve, Thomas (thomas.greve@haw-kiel.de)
Lehrperson(en)	Greve, Thomas (thomas.greve@haw-kiel.de)
Wird angeboten zum	Wintersemester 2026/27
Moduldauer	1 Fachsemester
Angebotsfrequenz	Regelmäßig
Angebotsturnus	In der Regel im Wintersemester
Lehrsprache	Deutsch
Empfohlen für internationale Studierende	Nein
Ist als Wahlmodul auch für andere Studiengänge freigegeben (ggf. Interdisziplinäres Modulangebot - IDL)	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"> - Asynchronie in Form von Neben- und Parallelität - Multitasking und Multithreading <p>Vermittlung</p> <ul style="list-style-type: none"> - der C++-Sprachkonstrukte, mit denen diese Konzepte realisiert werden können - von Bibliotheken, für solche Konstrukte, die (noch) nicht im C++-Standard enthalten sind. <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"> - einschätzen, bei welchen Aufgabenstellungen Neben- und Parallelität sinnvoll eingesetzt werden kann (und bei welchen nicht) - entscheiden, welche der unterschiedlichen Sprachkonstrukte, die C++ für die Umsetzung bietet, den meisten Nutzen bieten - Neben- und Parallelität einschliesslich ggf. erforderlicher Synchronisationsmechanismen in C++ programmieren
<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"> - nicht triviale softwaretechnische Sachverhalte zu diskutieren und so zu einem gemeinsamen Lösungsansatz für eine gestellte Aufgabe zu kommen - einen effizienten Weg für die Realisierung des Lösungsansatzes zu finden (Aufgabenteilung, Wiederverwendung)
<p>Die Teilnehmer können Aufgabenstellungen, deren Realisierung Neben- oder Parallelität voraussetzen, selbstständig identifizieren und lösen</p>

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

Literatur	<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>
------------------	--

Lehrformen der Lehrveranstaltungen	
Lehrform	SWS
Projekt	2
Lehrvortrag + Übung	2

Arbeitsaufwand	
Anzahl der SWS	4 SWS
Leistungspunkte	5,00 Leistungspunkte
Präsenzzeit	48 Stunden
Selbststudium	102 Stunden

Modulprüfungsleistung	
Voraussetzung für die Teilnahme an der Prüfung gemäß PO	Die Termine mit Anwesenheitspflicht wurden wahrgenommen
BI140 - Portfolioprüfung	Prüfungsform: Portfolioprüfung 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.

Sonstiges	
Empfohlene Voraussetzungen	Bestandenene Modulleistung: Programmieren in C++ (PIC)