

<b>KARTA OPISU MODUŁU KSZTAŁCENIA</b>		
Nazwa modułu/przedmiotu <b>Zaawansowane zastosowania kart graficznych</b>		Kod <b>1010512331010519522</b>
Kierunek studiów <b>Informatyka</b>	Profil kształcenia (ogólnoakademicki, praktyczny) <b>ogólnoakademicki</b>	Rok / Semestr <b>2 / 3</b>
Ścieżka obieralności/specjalność <b>Systemy rozproszone</b>	Przedmiot oferowany w języku: <b>polski</b>	Kurs (obligatoryjny/obieralny) <b>obieralny</b>
Stopień studiów: <b>II stopień</b>	Forma studiów (stacjonarna/niestacjonarna) <b>stacjonarna</b>	
Godziny Wykłady: <b>30</b> Ćwiczenia: - Laboratoria: <b>30</b> Projekty/seminaria: -		Liczba punktów <b>4</b>
Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny) (ogólnouczelniany, z innego kierunku) <b>kierunkowy z danego kierunku</b>		
Obszar(y) kształcenia i dziedzina(y) nauki i sztuki <b>nauki techniczne</b>		Podział ECTS (liczba i %) <b>4 100%</b>
<b>Odpowiedzialny za przedmiot / wykładowca:</b>  dr inż. Witold Andrzejewski email: Witold.Andrzejewski@cs.put.poznan.pl tel. 61 6652965 Instytut Informatyki ul. Piotrowo 2, 60-965 Poznań		
<b>Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:</b>		
1	<b>Wiedza:</b>	Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę o architekturze komputerów, o programowaniu obiektowym i proceduralnym oraz o metodach oceny złożoności algorytmów. Dodatkowo, powinien posiadać podstawową wiedzę dotyczącą geometrii obliczeniowej.
2	<b>Umiejętności:</b>	Powinien posiadać umiejętność rozwiązywania podstawowych problemów algorytmicznych, programowania w języku C/C++, oraz umiejętność pozyskiwania informacji ze wskazanych źródeł.
3	<b>Kompetencje społeczne</b>	Powinien również rozumieć konieczność poszerzania swoich kompetencji i mieć gotowość do podjęcia współpracy w ramach zespołu. Ponadto w zakresie kompetencji społecznych student musi prezentować takie postawy jak uczciwość, odpowiedzialność, wytrwałość, ciekawość poznawcza, kreatywność, kultura osobista, szacunek dla innych ludzi.
<b>Cel przedmiotu:</b> 1. Przekazanie studentom podstawowej wiedzy z programowania procesorów kart graficznych (GPU), w zakresie: a. Wybranych aspektów architektury sprzętowej popularnych kart graficznych. b. Logicznego modelu współbieżności kart graficznych w kontekście API CUDA i OpenCL. c. Optymalizacji wykonania programów na kartach graficznych. d. Wybranych podstawowych algorytmów równoległych, w tym: map, reduce, compact, sort, scan, search, generowania ntych wyrazów ciągów. e. Wybranych struktur danych, w tym: macierzy, tablic hashowych i drzew CSS i algorytmów ich przetwarzania. f. Oceny złożoności algorytmów równoległych (w tym model PRAM) i ich związku z faktyczną złożonością programów wykorzystujących karty graficzne. g. Zastosowania kart graficznych do rozwiązywania praktycznych problemów związanych z przetwarzaniem i wizualizacją danych. 2. Rozwijanie u studentów umiejętności: a. Rozwiązywania problemów algorytmicznych z ukierunkowaniem na zrównoleglenie operacji przetwarzania danych. b. Optymalizacji programów wykorzystujących procesory kart graficznych.		
<b>Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia</b>		
<b>Wiedza:</b>		

<ol style="list-style-type: none"><li>1. ma zaawansowaną i pogłębioną wiedzę z zakresu narzędzi i środowisk programistycznych użytecznych przy tworzeniu programów na procesory kart graficznych - [K2st_W1]</li><li>2. ma zaawansowaną wiedzę szczegółową dotyczącą tworzenia algorytmów równoległych na procesory kart graficznych - [K2st_W3]</li><li>3. ma zaawansowaną i szczegółową wiedzę o działaniu i architekturze procesorów kart graficznych - [K2st_W5]</li><li>4. zna zaawansowane metody, techniki i narzędzia stosowane przy rozwiązywaniu złożonych zadań inżynierskich związanych z projektowaniem i optymalizacją niskopoziomową programów wykorzystujących procesory kart graficznych - [K2st_W6]</li></ol>
<b>Umiejętności:</b>
<ol style="list-style-type: none"><li>1. potrafi posługiwać się technikami informacyjno-komunikacyjnymi wykorzystywanymi przy realizacji przedsięwzięć informatycznych - [K2st_U2]</li><li>2. potrafi - przy formułowaniu i rozwiązywaniu zadań inżynierskich - integrować wiedzę z różnych obszarów informatyki oraz zastosować podejście systemowe, uwzględniające także aspekty pozatechniczne - [K2st_U5]</li><li>3. potrafi ocenić przydatność i możliwość wykorzystania nowych osiągnięć i produktów z zakresu kart graficznych - [K2st_U6]</li><li>4. potrafi ocenić przydatność metod i narzędzi służących do rozwiązania zadania inżynierskiego, polegającego na budowie lub ocenie systemu informatycznego, lub jego składowych, wykorzystujących akcelerację GPU, w tym dostrzec ograniczenia tych metod i narzędzi - [K2st_U9]</li><li>5. potrafi ? zgodnie z zadaną specyfikacją, uwzględniającą aspekty pozatechniczne ? zaprojektować i zaimplementować algorytm wykorzystujący akcelerację GPU potrafi - zgodnie z zadaną specyfikacją, uwzględniającą aspekty pozatechniczne - zaprojektować i zaimplementować algorytm wykorzystujący akcelerację GPU - [K2st_U11]</li></ol>
<b>Kompetencje społeczne:</b>
<ol style="list-style-type: none"><li>1. rozumie, że w informatyce wiedza i umiejętności z zakresu kart graficznych bardzo szybko stają się przestarzałe - [K2st_K1]</li><li>2. rozumie znaczenie wykorzystywania najnowszej wiedzy z zakresu kart graficznych w rozwiązywaniu problemów badawczych i praktycznych - [K2st_K2]</li></ol>

<b>Sposoby sprawdzenia efektów kształcenia</b>
<p>Ocena formująca</p> <p>a) w zakresie wykładów weryfikowanie założonych efektów kształcenia realizowane jest przez odpowiedzi na pytania dotyczące materiału omówionego na poprzednich wykładach</p> <p>b) w zakresie laboratoriów / ćwiczeń weryfikowanie założonych efektów kształcenia realizowane jest przez: okresową ocenę przygotowania studenta do poszczególnych sesji zajęć laboratoryjnych (sprawdzian wejściowy) oraz ocenę umiejętności związanych z realizacją ćwiczeń laboratoryjnych</p> <p>Ocena podsumowująca:</p> <p>a) w zakresie wykładów weryfikowanie założonych efektów kształcenia realizowane jest przez ocenę wiedzy i umiejętności wykazanych na zaliczeniu pisemnym o formie testu wielokrotnego wyboru składającego się z ok. 30 pytań, łączna liczba punktów za prawidłowe odpowiedzi: 30, minimalna liczba punktów umożliwiających zaliczenie: 16</p> <p>b) w zakresie laboratoriów / ćwiczeń weryfikowanie założonych efektów kształcenia realizowane jest przez ocenę i obronę przez studenta sprawozdania z realizacji projektu.</p> <p>Uzyskiwanie punktów dodatkowych za aktywność podczas zajęć, a szczególnie za:</p> <ul style="list-style-type: none"><li>- efektywność zastosowania zdobytej wiedzy podczas rozwiązywania zadanego problemu,</li><li>- uwagi związane z udoskonaleniem materiałów dydaktycznych,</li><li>- wskazywanie trudności percepcyjnych studentów umożliwiające bieżące doskonalenia procesu dydaktycznego.</li></ul>
<b>Treści programowe</b>
<p>Program wykładu obejmuje następujące zagadnienia:</p> <ol style="list-style-type: none"><li>1. Motywacja stojąca za wykorzystaniem kart graficznych do obliczeń. Dodatkowo: przedstawienie różnych rozwiązań technologicznych umożliwiających przetwarzanie równoległe, wprowadzenie podstawowych pojęć stosowanych w kolejnych wykładach. Omówienie podstaw modelu programistycznego wykorzystywanego w programach wykorzystujących procesory kart graficznych.</li><li>2. Architektury sprzętowa kart graficznych. Omówienie związków pomiędzy architekturą sprzętową a modelem programistycznym. Przedstawienie trików bitowych.</li><li>3. Omówienie hierarchii pamięci i metod wydajnego dostępu dla różnych architektur sprzętowych. Przedstawienie przykładowych metod optymalizacji wykorzystujących różne poziomy pamięci operacyjnej. Omówienie rozwiązań pozwalających na zapewnienie równoległego przesyłu danych pomiędzy komputerem a kartą graficzną i wykonywania obliczeń.</li><li>4. Omówienie mechanizmów komunikacji i synchronizacji wątków</li><li>5. Wprowadzenie podstaw teoretycznych oceny złożoności algorytmów równoległych. Omówienie obliczeń w modelu maszyny PRAM. Podstawowe algorytmy wykorzystywane podczas konstrukcji bardziej złożonych rozwiązań, w tym: map, gather, scatter, reduce, compact, search, scan. Szczegółowe omówienie algorytmu scan. Ocena złożoności wprowadzonych algorytmów w ich wersji sekwencyjnej i równoległej.</li><li>6. Szczegółowe omówienie algorytmów : compact i reduce oraz algorytmów przeszukiwania i sortowania danych. Ocena złożoności przedstawionych algorytmów.</li></ol>

7. Algorytmy generowania elementów w sekwencji kombinacji, permutacji bądź w wyniku iloczynu kartezyjskiego wraz z ich przykładowymi zastosowaniami.
8. Algorytmy połączeń pionowych i poziomych w oparciu o algorytm merge-path
9. Problemy związane z wykorzystywaniem standardowych struktur danych. Omówienie wydajnych metod przetwarzania list (drzewo CSS), macierzy (w tym macierzy rzadkich), tablic haszowych oraz grafów. Oszacowanie złożoności algorytmów przetwarzania tych struktur.
10. Praktyczne zastosowania GPU do przetwarzania, eksploracji i wizualizacji danych.

Program laboratorium obejmuje następujące zagadnienia:

1. Zapoznanie się z CUDA API.
2. Ćwiczenia związane z prawidłową konstrukcją siatki obliczeń.
3. Implementacja prostych algorytmów równoległych.
4. Omówienie sposobów debugowania programów wykorzystujących karty graficzne.
5. Realizacja ćwiczeń w których należy uwzględnić komunikację i synchronizację wątków.
6. Testowanie wydajności dostępu do różnych dostępnych rodzajów pamięci operacyjnej na karcie graficznej.
7. Zapoznanie się z biblioteką thrust. Omówienie podstawowych koncepcji stojących za API tej biblioteki. Przedstawienie podstawowych algorytmów w niej zaimplementowanych.
8. Omówienie biblioteki CURAND. Ćwiczenia wykorzystujące bibliotekę thrust. Budowa złożonych algorytmów z podstawowych algorytmów takich jak: map, reduce, gather, scatter, search, scan.
9. Omówienie API OpenCL oraz jego podobieństw i różnic w stosunku do NVIDIA CUDA
10. Omówienie NVIDIA Optix API i jego zastosowań przy wizualizacji danych.

Metody dydaktyczne:

1. wykład: prezentacja multimedialna, prezentacja ilustrowana przykładami podawanymi na tablicy, rozwiązywanie zadań.
2. ćwiczenia laboratoryjne: rozwiązywanie zadań, ćwiczenia praktyczne, pokaz multimedialny, demonstracja.

#### Literatura podstawowa:

1. Cuda w przykładach : wprowadzenie do ogólnego programowania procesorów GPU / Jason Sanders, Edward Kandrot
2. OpenCL : akceleracja GPU w praktyce / Marek Sawerwain.
3. NVIDIA: CUDA C Programming Guide: <http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html>
4. NVIDIA: CUDA C Best Practices Guide: <http://docs.nvidia.com/cuda/cuda-c-best-practices-guide/index.html>
5. NVIDIA: OpenCL Jumpstart Guide: [http://www.cs.cmu.edu/afs/cs/academic/class/15668-s11/www/cuda-doc/OpenCL\\_Jumpstart\\_Guide.pdf](http://www.cs.cmu.edu/afs/cs/academic/class/15668-s11/www/cuda-doc/OpenCL_Jumpstart_Guide.pdf)
6. NVIDIA: OpenCL Best Practices Guide: [http://www.nvidia.com/content/cudazone/CUDABrowser/downloads/papers/NVIDIA\\_OpenCL\\_BestPracticesGuide.pdf](http://www.nvidia.com/content/cudazone/CUDABrowser/downloads/papers/NVIDIA_OpenCL_BestPracticesGuide.pdf)
7. NVIDIA Optix API documentation <http://raytracing-docs.nvidia.com/optix/index.html>

#### Literatura uzupełniająca:

1. Jianlong Zhong\* and Bingsheng He. Medusa: Simplified Graph Processing on GPUs. Accepted by TPDS 2013: IEEE Transactions on Parallel and Distributed System
2. Bingsheng He and Jeffrey Xu Yu. High-Throughput Transaction Executions on Graphics Processors. Proceedings of Very Large Data Bases (VLDB) 2011
3. Andrzejewski, Witold; Boinski, Pawel Efficient spatial co-location pattern mining on multiple GPUs Journal Article Expert Systems with Applications, 93 (Supplement C), pp. 465-483, 2018, ISBN: 0957-4174.
4. Andrzejewski, Witold; Boinski, Pawel Parallel GPU-based Plane-sweep Algorithm for Construction of iCPI-trees Journal Article Journal of Database Management, 26 (3), pp. 1-20, 2015, ISSN: 1063-8016.
5. Andrzejewski, Witold; Boinski, Pawel Parallel approach to incremental co-location pattern mining, Information Sciences, accepted for publication

#### Bilans nakładu pracy przeciętnego studenta

Czynność	Czas (godz.)
----------	--------------

1. udział w zajęciach laboratoryjnych	30	
2. przygotowanie do ćwiczeń laboratoryjnych	15	
3. udział w konsultacjach związanych z realizacją procesu kształcenia, w szczególności ćwiczeń laboratoryjnych / projektu	2 8	
4. przygotowanie do sprawdzianów / kolokwium	30	
5. udział w wykładach	10	
6. zapoznanie się ze wskazaną literaturą / materiałami dydaktycznymi	5	
7. przygotowanie do zaliczenia wykładów		
<b>Obciążenie pracą studenta</b>		
<b>forma aktywności</b>	<b>godzin</b>	<b>ECTS</b>
Łączny nakład pracy	100	4
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	62	2
Zajęcia o charakterze praktycznym	45	2