KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu
Architektura systemów brzegowych [S2Inf1-PB>ASB]

Przedmiot

Kierunek studiów
Informatyka

Rok/Semestr
1/1

Studia w zakresie (specjalność)
Przetwarzanie brzegowe

Profil studiów
ogólnoakademicki

Poziom studiów
drugiego stopnia

Język oferowanego przedmiotu
polski

Forma studiów
stacjonarne

Wymagalność
obligatoryjny

Liczba godzin

Wykład
30

Laboratorium
30

Inne
0

Ćwiczenia
0

Projekty/seminaria
0

Liczba punktów ECTS

5,00

Koordynatorzy

dr inż. Adam Turkot
adam.turkot@put.poznan.pl

Wykładowcy

Wymagania wstępne

Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę z zakresu protokołów sieciowych HTML, TCP, UDP, SSH, ICMP, programowania w języku C i C++, w interpreterze poleceń Bash oraz w języku Python i JavaScript. Powinien również rozumieć konieczność poszerzania swoich kompetencji / 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.

Cel przedmiotu

Zaznajomienie studentów z metodologią projektowania architektury systemów brzegowych. Przekazanie studentom poszerzonej wiedzy z zakresu architektur systemów brzegowych oraz stosowanych w nich rozwiązaniach sprzętowych i programowych. Wykształcenie umiejętności techniki programowania zapewniającej: efektywne wykorzystanie zasobów sprzętowych systemów brzegowych. Optymalne, dla danego zadania realizacje aplikacji z użyciem odpowiedniej platformy sprzętowej, z obsługą dedykowanych modułów peryferyjnych i przy uwzględnieniu wymogów związanych z oszczędnością energii i wydajnością obliczeniową. Opanowanie technik komunikacji pomiędzy kontrolerem, a cyfrowymi i analogowymi elementami systemów brzegowych. Zaznajomienie studentów z możliwościami i ograniczeniami budowania systemów brzegowych w oparciu o kontrolery oraz komputery jednopłytkowe. Kształtowanie u studentów umiejętności pracy zespołowej poprzez realizacje elementów projektu i połączenie ich w całość.

Przedmiotowe efekty uczenia się

Wiedza:
1. ma zaawansowaną i pogłębioną wiedzę z zakresu szeroko rozumianych systemów informatycznych, podstaw teoretycznych ich budowania oraz metod, narzędzi i środowisk.
2. ma uporządkowaną i podbudowaną teoretycznie wiedzę ogólną związaną z kluczowymi zagadnieniami z zakresu informatyki.
3. zna zaawansowane metody, techniki i narzędzia stosowane przy rozwiązywaniu złożonych zadań inżynierskich i prowadzeniu prac badawczych w wybranym obszarze informatyki.

Umiejętności:
1. potrafi pozyskiwać informacje z literatury, baz danych oraz innych źródeł (w języku polskim i angielskim), integrować je, dokonywać ich interpretacji i krytycznej oceny, wyciągać wnioski oraz formułować i wyczerpująco uzasadniać opinie.
2. potrafi wykorzystać do formułowania i rozwiązywania zadań inżynierskich i prostych problemów badawczych metody analityczne, symulacyjne oraz eksperymentalne
3. potrafi — przy formułowaniu i rozwiązywaniu zadań inżynierskich — integrować wiedzę z różnych obszarów informatyki (a w razie potrzeby także wiedzę z innych dyscyplin naukowych) oraz zastosować podejście systemowe, uwzględniające także aspekty pozatechniczne.
4. potrafi dokonać krytycznej analizy istniejących rozwiązań technicznych oraz zaproponować ich ulepszenia (usprawnienia).
5. potrafi - stosując m.in. koncepcyjnie nowe metody - rozwiązywać złożone zadania informatyczne, w tym zadania nietypowe oraz zadania zawierające komponent badawczy.
6. potrafi — zgodnie z zadaną specyfikacją, uwzględniającą aspekty pozatechniczne — zaprojektować złożone urządzenie, system informatyczny lub proces oraz zrealizować ten projekt — co najmniej w części — używając właściwych metod, technik i narzędzi, w tym przystosowując do tego celu istniejące lub opracowując nowe narzędzia.

Kompetencje społeczne:
1. rozumie, że w informatyce wiedza i umiejętności bardzo szybko stają się przestarzałe.
2. rozumie znaczenie wykorzystywania najnowszej wiedzy z zakresu informatyki w rozwiązywaniu problemów badawczych i praktycznych.

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Ocena formująca:
a) w zakresie wykładów: na podstawie odpowiedzi na pytania dotyczące materiału omówionego na poprzednich wykładach
b) w zakresie laboratoriów / ćwiczeń: na podstawie oceny bieżącego postępu realizacji zadań,
Ocena podsumowująca:
a) w zakresie wykładów weryfikowanie założonych efektów kształcenia realizowane jest przez: egzamin ustny połączony z obroną projektu, w przypadku wątpliwości część pisemna (test w postaci elektronicznej na platformie Moodle);
b) w zakresie laboratoriów weryfikowanie założonych efektów kształcenia realizowane jest przez: ocenę przygotowania studenta do poszczególnych sesji zajęć laboratoryjnych poprzez sprawdzenie przygotowania zadanych projektów/ćwiczeń oraz ocenę umiejętności związanych z realizacją ćwiczeń laboratoryjnych, ocenianie ciągłe, na każdych zajęciach (odpowiedzi ustne) premiowanie przyrostu

umiejętności posługiwania się poznanymi zasadami i metodami, ocenę dokumentacji tworzonej systematycznie wraz z postępami prac projektowych; dokumentacja przygotowywana częściowo w trakcie zajęć, a częściowo po ich zakończeniu; ocena ta obejmuje także umiejętność pracy w zespole, ocenę i obronę przez studenta sprawozdania z realizacji projektu,

Treści programowe

Podstawy architektury systemów brzegowych. Techniki efektywnego wykorzystania zasobów sprzętowych. Ocena możliwości sprzętowych. Środowisko programistyczne. Techniki programowania. Metody optymalizacji kodu. Interfejs użytkownika. Rozwiązania sprzętowe i programistyczne umożliwiające zarządzanie poborem mocy. Technik zabezpieczeń oprogramowania (integralność programu, odporność prze nieautoryzowanym kopiowaniem) Architektury kontrolerów. Zasoby lokalne i współdzielone, konsekwencje współdzielenia zasobów. Magistrale w systemach rozproszonych. Techniki sprzętowe i programowe dla zwiększenia niezawodności łącza komunikacyjnego. Techniki i protokoły komunikacji wykorzystywane w komunikacji z peryferiami oraz pomiędzy kontrolerami i w chmurze. Zwiększania niezawodności systemów bezobsługowych, techniki zapewniające gospodarkę energetyczną systemów autonomicznych.
Zajęcia laboratoryjne prowadzone są w formie piętnastu 2-godzinnych ćwiczeń, odbywających się w laboratorium, poprzedzonych 2-godzinną sesją instruktażową na początku semestru. Ćwiczenia realizowane są przez 2-osobowe zespoły studentów.
Laboratoria obejmują: Programowanie architektur komputerów jednopłytkowych. Programowania architektur systemów mikrokontrolerów. Protokoły komunikacyjne w szczególności SPI, I2C, UART. Protokołu komunikacyjne IoT. Programowanie architektury modułowych na przykładzie Colibri iMX7.

Tematyka zajęć

Podstawy architektury systemów brzegowych. Techniki efektywnego wykorzystania zasobów sprzętowych. Ocena możliwości sprzętowych. Środowisko programistyczne. Techniki programowania. Metody optymalizacji kodu. Interfejs użytkownika. Rozwiązania sprzętowe i programistyczne umożliwiające zarządzanie poborem mocy. Technik zabezpieczeń oprogramowania (integralność programu, odporność prze nieautoryzowanym kopiowaniem) Architektury kontrolerów. Zasoby lokalne i współdzielone, konsekwencje współdzielenia zasobów. Magistrale w systemach rozproszonych. Techniki sprzętowe i programowe dla zwiększenia niezawodności łącza komunikacyjnego. Techniki i protokoły komunikacji wykorzystywane w komunikacji z peryferiami oraz pomiędzy kontrolerami i w chmurze. Zwiększania niezawodności systemów bezobsługowych, techniki zapewniające gospodarkę energetyczną systemów autonomicznych.
Zajęcia laboratoryjne prowadzone są w formie piętnastu 2-godzinnych ćwiczeń, odbywających się w laboratorium, poprzedzonych 2-godzinną sesją instruktażową na początku semestru. Ćwiczenia realizowane są przez 2-osobowe zespoły studentów.
Laboratoria obejmują: Programowanie architektur komputerów jednopłytkowych. Programowania architektur systemów mikrokontrolerów. Protokoły komunikacyjne w szczególności SPI, I2C, UART. Protokołu komunikacyjne IoT. Programowanie architektury modułowych na przykładzie Colibri iMX7.

Metody dydaktyczne

Metody dydaktyczne:
wykład: prezentacja multimedialna, prezentacja ilustrowana przykładami podawanymi na tablicy, prezentacje wybranych rozwiązań studenckich.
ćwiczenia laboratoryjne: ćwiczenia praktyczne, wykonywanie eksperymentów, dyskusja, praca w zespole.

Literatura

Podstawowa
Linux w systemach embedded Bis, Marcin. Wydawnictwo btc, 2011
Building embedded Linux systems Yaghmour, Karim. O"Reilly, cop. 2003.
Uzupełniająca
Wbudowane systemy mikroprocesorowe Timofiejew, Aleksander., Akademia Podlaska (Siedlce). Wydawnictwo.Wydawnictwo Akademii Podlaskiej, 2010.

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	125	5,00
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	60	2,50
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych/ćwiczeń, przygotowanie do kolokwiów/egzaminu, wykonanie projektu)	65	2,50