KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu
Sterowniki dla systemu Linux [S2Inf1-PB>LINUX]

Przedmiot

Kierunek studiów
Informatyka

Rok/Semestr
1/2

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
15

Laboratorium
15

Inne
0

Ćwiczenia
0

Projekty/seminaria
0

Liczba punktów ECTS

3,00

Koordynatorzy

dr inż. Mariusz Naumowicz
mariusz.naumowicz@put.poznan.pl

Wykładowcy

Wymagania wstępne

Student rozpoczynający przedmiot powinien posiadać podstawową wiedzę z zakresu systemów operacyjnych i elektroniki. Powinien również rozumieć konieczność poszerzania swoich kompetencji oraz mieć gotowość do podjęcia współpracy w ramach zespołu.

Cel przedmiotu

- Przekazanie studentom wiedzy związanej z nowoczesnymi systemami wbudowanymi oraz systemem operacyjnym Linux i jego zastosowaniem w systemach IoT. - Zapoznanie studentów z nowoczesnymi metodami projektowania, testowania i prototypowania sterowników w systemie Linux dla systemów wbudowanych. - Rozwijanie u studentów umiejętności rozwiązywania złożonych problemów projektowych w zakresie systemów wbudowanych i systemów operacyjnych. - Kształtowanie u studentów umiejętności pracy zespołowej.

Przedmiotowe efekty uczenia się

Wiedza:
1. ma zaawansowaną i pogłębioną wiedzę z zakresu szeroko rozumianych systemów informatycznych oraz metod i narzędzi wykorzystywanych do ich implementacji, szczególnie dotyczących budowania

warstwy sprzętowej systemów reprogramowalnych - [k2st_w1]
2. ma zaawansowaną wiedzę szczegółową dotyczącą wybranych zagadnień z zakresu informatyki, szczególnie dotyczącą konstruowania systemów wbudowanych - [k2st_w3]
3. ma zaawansowaną i szczegółową wiedzę o procesach zachodzących w cyklu życia systemów informatycznych, szczególnie warstwy sprzętowej systemów - [k2st_w5]
4. zna zaawansowane metody, techniki i narzędzia stosowane przy rozwiązywaniu złożonych zadań inżynierskich i prowadzeniu prac badawczych w wybranym obszarze informatyki - [k2st_w6]

Umiejętności:
1. 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 - [k2st_u5]
2. potrafi ocenić przydatność i możliwość wykorzystania nowych osiągnięć (metod i narzędzi) oraz nowych produktów informatycznych - [k2st_u6]
3. potrafi - stosując m.in. koncepcyjnie nowe metody - rozwiązywać złożone zadania informatyczne, w tym zadania nietypowe oraz zadania zawierające komponent badawczy - [k2st_u10]
4. potrafi zgodnie z zadaną specyfikacją, zaprojektować złożone urządzenie, system informatyczny lub proces oraz zrealizować ten projekt używając właściwych metod, technik i narzędzi, w tym przystosowując do tego celu istniejące lub opracowując nowe narzędzia - [k2st_u11]
5. potrafi określić kierunki dalszego uczenia się i zrealizować proces samokształcenia, w tym innych osób - [k2st_u16]

Kompetencje społeczne:
1. rozumie, że w informatyce wiedza i umiejętności bardzo szybko stają się przestarzałe - [k2st_k1]
2. rozumie znaczenie wykorzystywania najnowszej wiedzy z zakresu informatyki w rozwiązywaniu problemów badawczych i praktycznych - [k2st_k2]

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Ocena formująca:
a) na podstawie przygotowanej prezentacji w ramach wykładu,,
b) w zakresie laboratoriów: na podstawie oceny bieżącego postępu realizacji zadań,

Ocena podsumowująca:
a) w zakresie wykładów weryfikowanie założonych efektów kształcenia realizowane jest przez zaliczenie ustne połączone 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 sprawdzian projektowy i ocenę zadań realizowanych w ramach każdego spotkania laboratoryjnego;
Uzyskiwanie punktów dodatkowych za aktywność podczas zajęć, a szczególnie za:
- omówienia dodatkowych aspektów zagadnienia,
- efektywność zastosowania zdobytej wiedzy podczas rozwiązywania zadanego problemu,
- umiejętność współpracy w ramach zespołu praktycznie realizującego zadanie szczegółowe w laboratorium.

Treści programowe

Treści programowe obejmują zagadnienia związane z tworzeniem sterowników dla systemu operacyjnego Linux.

Tematyka zajęć

Program wykładu obejmuje następujące zagadnienia:
- wprowadzenie do Linuxa, budowanie kernela, budowanie modułów (w drzewie i poza), konfiguracja, devicetree, bootargs, uruchamianie Linuxa. Wprowadzenie do narzędzi wspierających projektowanie i testowanie sterowników,
- struktura sterownika, zagadnienia związane z obsługa sterownika w systemie operacyjnym. Przybliżenie podstawowych zagadnień projektowania sterowników na przykładzie urządzenia znakowego. Interakcja sterownika ze sprzętem (mapowanie pamięci, dostęp do rejestrów, ciągła pamięć dla DMA),

- implementacja sterownika konfigurującego peripheral. Obsługa przerwań jądro systemu Linuxa, implementacja handlera przerwań w sterownikach (rejestracja/derejestracja, definicja w devicetree, wielowątkowość). Podsystemy w Linuxie. Budowa sterowników wykorzystywanych do komunikacji w systemach wbudowanych.

Zajęcia laboratoryjne prowadzone są w formie 2-godzinnych spotkań, odbywających się w laboratorium, poprzedzonych sesją instruktażową na początku semestru. Ćwiczenia realizowane są przez 2-osobowe zespoły studentów.

Program laboratorium obejmuje następujące zagadnienia:
- przygotowanie środowiska programistycznego niezbędnego do programowania sterowników dla dedykowanego systemu wbudowanego, konfigurowanie, modyfikowanie i budowanie jądra systemu Linux,
- programowanie sterownika znakowego, rejestrowanie sterownika, usuwanie sterownika, obsługa drzewa urządzeń,
- kopiowanie danych pomiędzy przestrzenią użytkownika a przestrzenią jądra, programowanie aplikacji służącej do interakcji ze sterownikiem z poziomu przestrzeni użytkownika,
- interakcja ze sprzętem (mapowanie pamięci, dostęp do rejestrów, ciągła pamięć dla DMA),
- obsługa przerwań, wielowątkowość,
- podsystemy w Linuxie.

Część wymienionych wyżej treści programowych realizowana jest w ramach pracy własnej studenta.

Metody dydaktyczne

1. wykład: prezentacja multimedialna uzupełniona przykładami podawanymi na tablicy.
2. ćwiczenia laboratoryjne: ćwiczenia praktyczne, dyskusja, praca w zespole, zawody projektowe.

Literatura

Podstawowa:
1. Jonathan Corbet, Alessandro Rubini, and Greg Kroah-Hartman., Linux Device Drivers, 3rd Edition. O"Reilly Media, Inc. 2005. ISBN: 0596005903.
2. Alberto Liberal de los Ríos, Linux Driver Development for Embedded Processors - Second Edition: Learn to develop Linux embedded drivers with kernel 4.9 LTS, Independently Published, 2018. ISBN: 1729321828.

Uzupełniająca:
1. Daniel Bovet and Marco Cesati, Understanding the Linux Kernel, Third Edition, O’Reilly & Associates Inc., 2005. ISBN: 0596005652.

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	75	3,00
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	30	1,00
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych/ćwiczeń, przygotowanie do kolokwiów/egzaminu, wykonanie projektu)	45	2,00