KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu
Zarządzanie zasobami sprzętowymi w systemach wbudowanych

Przedmiot

Kierunek studiów
Informatyka
Studia w zakresie (specjalność)
Przetwarzanie brzegowe
Poziom studiów
drugiego stopnia
Forma studiów
stacjonarne

Rok/semestr
1/2
Profil studiów
ogólnoakademicki
Język oferowanego przedmiotu
polski
Wymagalność
obieralny

Liczba godzin

Wykład
15
Ćwiczenia

Laboratoria
15
Projekty/seminaria

Inne (np. online)

Liczba punktów ECTS

3

Wykładowcy

Odpowiedzialny za przedmiot/wykładowca:
dr inż. Mariusz Naumowicz
email: mariusz.naumowicz@put.poznan.pl
tel. +48 61 665-2364
Wydział Informatyki i Telekomunikacji
ul. Piotrowo 3 60-965 Poznań

Odpowiedzialny za przedmiot/wykładowca:

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) w zakresie wykładów: na podstawie odpowiedzi na pytania dotyczące materiału omówionego na poprzednich wykładach,
b) w zakresie laboratoriów: 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 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

Program wykładu obejmuje następujące zagadnienia:

Budowa jądra systemu Linux, budowa modułów, konfiguracja i budowanie jądra Linuxa. Struktura sterowników Linuxa na bazie sterownika znakowego. Obsługa przerwań przez jądro systemu Linuxa, wykorzystanie przerwań w sterownikach urządzeń. Sterowniki wspierające wymianę danych między urządzeniami, standardy komunikacji w systemach wbudowanych. Opóźnienia w systemie Linux i dostęp do sekcji krytycznej.

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:

Wprowadzenie do Linuxa, budowanie kernela, budowanie modułów (w drzewie i poza), konfiguracja, devicetree, bootargs, bootowanie Linuxa. Renode - uruchomienie platformy. Struktura drivera, budowanie jako moduł i jako część kernela, ładowanie i rejestracja modułów, alokacja pamięci w modułach kernela. Character device driver. Dostęp do zasobów kernela z user space. Komunikacja aplikacji user space z driverami - IOCTLs. Immplementacja prostej aplikacji user space do interakcji z driverem. Interakcja z hardware (mapowanie pamięci, dostęp do rejestrów, ciągła pamięć dla DMA). Implementacja drivera który konfiguruje peripheral. Przerwania (jak Linux obsługuje przerwania),

implementacja handlera przerwań w driverach . Latency obsługi przerwań, dostęp do sekcji krytycznej. Podsystemy w Linuxie. Implementacja drivera I2C. Sysfs, implementacja podsystemu industrial I/O w driverze.

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. Noam Nisan, Shimon Schocken, The Elements of Computing Systems: Building a Modern Computer from First Principles, The MIT Press, 2005. ISBN: 0262640686.

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	75	3,0
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	30	1,5
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych, przygotowanie do sprawdzianu/egzaminu, wykonanie projektu)[1]	45	1,5

1 niepotrzebne skreślić lub dopisać inne czynności