Politechnika Białostocka									
Kierunek studiów	**Informatyka**							**Poziom i forma studiów**	**pierwszego stopnia inżynierskie niestacjonarne**
Specjalność / Ścieżka dyplomowania	**---**							**Profil kształcenia**	ogólnoakademicki
Nazwa przedmiotu	**Programowanie hybrydowe**							**Kod przedmiotu**	**INZ1PHY**
								Rodzaj przedmiotu	obieralny
Forma zajęć i liczba godzin	W	Ć	L	P	Ps	T	S	**Semestr**	5,6
	20				10			**Punkty ECTS**	**5**
Przedmioty wprowadzające									
Cele przedmiotu	Celem przedmiotu jest przygotowanie studentów do programowania systemów komputerowych na styku hardware-software z wykorzystaniem technik programowania hybrydowego (niskopoziomowego oraz wysokopoziomowego). Studenci poznają metody przygotowania oprogramowania układowego w językach niskopoziomowych (assembler) oraz łączenie go z językami wyższego poziomu (C, C++, C#, Java, Python). Uczestnictwo w zajęciach pozwoli także studentom na zrozumienie konieczności stosowania technik niskopoziomowych w celu zapewnienia lepszej optymalizacji oprogramowania.								
Treści programowe	Wykład 1. Podstawowe informacje o programowaniu niskopoziomowym i hybrydowym. Języki maszynowe. Asembler. Zastosowania programowania niskopoziomowego i hybrydowego. 2. Mikrokontrolery i ich możliwości. Komponenty SoC: GPIO, Timery, PWM. Interfejsy: UART, I2C, SPI, QSPI, I2S, CAN. 3. Architektura ARM. Zestaw instrukcji ARM. Tryby adresowania. Cechy wyróżniające architekturę ARM. 4. Proces kompilacji. Podstawowe struktury programistyczne: przekazywanie parametrów, zwracanie wartości. 5. Metody i techniki łączenia kodu assemblera i języka C/C++. 6. Metody i techniki łączenia kodu assemblera/C i Javy (JNI). 7. Linux w systemach wbudowanych. Device Tree i jego wykorzystanie do personalizacji jądra Pracownia specjalistyczna: 1. Narzędzia do tworzenia oprogramowania z użyciem asemblera. 2. Procesy kompilacji i konsolidacji. Tworzenie programów wielomodułowych i bibliotek. 3. Tworzenie aplikacji hybrydowych. Wywoływanie kodu w asemblerze z aplikacji C/C++. 4. Tworzenie aplikacji hybrydowych. Wywoływanie kodu w asemblerze i C z aplikacji C#/Java/Python. Standard JNI (Java Native Interface).								
Metody dydaktyczne	programowanie z użyciem komputera, metoda projektów, wykład informacyjny,								
Forma zaliczenia	Wykład: Pisemny sprawdzian wiadomości. Warunkiem dopuszczenia jest uprzednie zaliczenie formy towarzyszącej. Pracownia specjalistyczna: Ocena wykonania zadań projektowych. Ocena uwzględnia poprawność, samodzielność i terminowość wykonywania zadań.								

Symbol efektu uczenia się	**Zakładane efekty uczenia się**	**Odniesienie do kierunkowych efektów uczenia się**
EU1	Posiada podstawową wiedzę o programowaniu z użyciem asemblera.	K_W03 K_W04
EU2	Zna zasady i techniki programowania hybrydowego oraz trendy rozwojowe.	K_W03 K_W06 K_K01
EU3	Potrafi projektować proste aplikacje z użyciem asemblera.	K_U06 K_U09
EU4	Potrafi wykorzystać programowanie hybrydowe do optymalizowania projektów tworzonych w językach wysokiego poziomu.	K_U04 K_U05

Symbol efektu uczenia się	**Sposób weryfikacji efektu uczenia się**	**Forma zajęć na której zachodzi weryfikacja**
EU1	Pisemny sprawdzian wiadomości.	W
EU2	Pisemny sprawdzian wiadomości. Ocena projektu.	W, Ps
EU3	Ocena projektu. Dyskusja.	Ps
EU4	Ocena projektu. Dyskusja.	PS

Bilans nakładu pracy studenta (w godzinach)	**Liczba godz.**
Wyliczenie	
1 - Udział w wykładach. - 10x2h	20
2 - Udział w pracowni specjalistycznej. - 10x1h	10
3 - Praca nad projektami w domu -	80
4 - Udział w konsultacjach. -	2
5 - Przygotowanie do egzaminu. -	15
6 - Obecność na zaliczeniach i egzaminie. -	2
RAZEM:	**129**

Wskaźniki ilościowe	**GODZINY**	**ECTS**
Nakład pracy studenta związany z zajęciami wymagającymi bezpośredniego udziału nauczyciela	34 (1)+(2)+(4)+(6)	1,3
Nakład pracy studenta związany z zajęciami o charakterze praktycznym	90 (2)+(3)	3,5

Literatura podstawowa	1. W. Hohl, Asembler dla procesorów ARM: Podręcznik programisty, Helion, 2014 2. K. Guntheroth, C++: optymalizacja kodu, APN Promise, 2016. 3. J. Hennessy, D. Patterson, Computer architecture: a quantitative approach, Morgan Kaufmann Publishers, 2007. 4. Dokumentacje na stronach ARM (https://developer.arm.com/docs).
Literatura uzupełniająca	1. A. Kowalczyk, Assembler, Wydawnictwo CROMA, 2000. 2. A. Aho, R. Sethi, J. Ullman, Compilers: principles, techniques, and tools, Addison-Wesley Publishers, 1993 3. W. Stallings, Organizacja i architektura systemu komputerowego: projektowanie systemu a jego wydajność, Wydawnictwa Naukowo-Techniczne, 2004.

Jednostka realizująca	Katedra Systemów Informacyjnych i Sieci Komputerowych	**Data opracowania programu**
Program opracował(a)	dr inż. Tomasz Grześ,mgr inż. Maciej Szymkowski	5 kwietnia 2019