(faculty stamp)

COURSE DESCRIPTION

1. Course title: EMBEDDED SYSTEMS PROGRAMMING	**2. Course code:** ESP
3. Validity of course description: 2019/2020	
4. Level of studies: 2nd cycle of higher education	
5. Mode of studies: intramural studies	
6. Field of study: informatics	
7. Profile of studies: general academic	
8. Specialty: Industrial Informatics Systems	
9. Semester: 2	
10. Faculty teaching the course: Institute of Informatics	
11. Course instructor: Dariusz Caban, PhD, Michał Maćkowski, PhD	
12. Course classification: specialization courses	
13. Course status: obligatory	
14. Language of instruction: English	
15. Pre-requisite qualifications: Computer Programming, Microprocessor and Embedded Systems, Computer Construction	
16. Course objectives: The aim of the course is to introduce students into topics of designing of embedded systems software, both bare-metal and executed under the supervision of the real-time system kernel. The issues related to the optimization of programs: memory and energy are also presented.	
17. Description of learning outcomes:	

No.	Learning outcomes description	Method of assessment	Teaching methods	Learning outcomes reference code
1.	The student has knowledge of the C language extensions for embedded systems programming.	Discussion	Lecture	K2A_W11
2.	The student learns the methods of memory and energy optimization of C programs for embedded systems.	Discussion	Lecture	K2A_W11
3.	The student knows the services provided by the real-time kernel.	Discussion	Lecture	K2A_W11
4.	The student knows how to write software, which is resilient to errors induced by electromagnetic fields.	Discussion	Lecture	K2A_W11

5.	The student is able to write software for embedded systems, both bare-metal and executed under the supervision of the real-time kernel.	Report	Laboratory	K2A_U13
6.	The student is able to optimize the program in terms of memory usage and energy consumption	Report	Laboratory	K2A_U07 K2A_U12

18. Teaching modes and hours

Lecture: 15 h, **Laboratory:** 30 h

19. Syllabus description:

Lecture

The embedded system definition. The features of embedded system software. The C language extensions for embedded systems programming. The memory optimization of C programs. The cross linker. The formats of files with executable code. The embedded system software architectures. Using of interrupts, the data sharing problem. The impact of interrupt blocking on the system response time. Alternatives to interrupt blocking.

The introduction to real-time kernel. The tasks and their states. The services of the kernel: semaphores, message queues, mailboxes, time functions, dynamic memory allocation. The interrupt service routines in programs executed under supervision of the kernel. Basics of designing programs executed under supervision of the kernel. The time and memory overheads being introduced by the kernel. Energy optimization of embedded system software.

As part of the lecture, the legal regulations applicable in the European Union will be presented related to the introduction of embedded systems into the market. Lecture issues: electromagnetic disturbances, sources of disturbances, types of couplings, effects of disturbances, grounding and mass, immunity to radiated and conducted disturbances, improving software performance against susceptibility to EMI, software techniques for comprehensive EMC testing of embedded systems, EMC-aware programming.

Laboratory

Memory optimization of C programs for AVR microcontrollers

The FreeRTOS kernel for PIC18 microcontrollers

The software for embedded system with ZigBee interface

Energy optimization of embedded system software

Immunity of communication standards to electromagnetic disturbances.

Radiated emission of IT devices.

Influence of IT equipment on voltage fluctuations in the power grid.

Testing of devices immunity to SURGE and ESD.

Emission and immunity - aware programming

20. Examination: no

21. Primary sources:

1. Lewis D. W.: Fundamentals of embedded software: where C and assembly meet. Prentice Hall, 2002
2. Simon D. E.: An embedded software primer. Addison-Wesley, 1999.

22. Secondary sources:

1. Labrosse J. J.: µC/OS-II. The Real-Time Kernel. User’s Manual. Micriµm Press, 2015.
2. Labrosse J. J.: µC/OS-III. The Real-Time Kernel. User’s Manual. Micriµm Press, 2010.

23. Total workload required to achieve learning outcomes

No.	Teaching mode:	Contact hours / Student workload hours
1.	Lectures	15 /15
2.	Classes	
3.	Laboratory	30/ 15
4.	Project	
5.	BA/MA Seminar	
6.	Other (exam)	

Total number of hours:	45 / 30

24. Total hours: 75
25. Number of ECTS credits: 3
26. Number of ECTS credit allocated for contact hours: 2
27. Number of ECTS credit allocated for in-practice hours (laboratory, classes, project): 1
28. Comments:

Approved:

………………………….…… ……………………………………………………....

(date, instructor's signature) (*date, the Director of the Faculty Unit signature)*