IOT AND PHYSICAL COMPUTING

Code: 27205

Main Scientific Area: Intelligent Systems and Control

Lecturer: Tiago Rafael dos Santos Martins Pereira Rodrigues

Language of Instruction: Portuguese

Regime: S2

Contact Hours: 30h Total Workload: 138h

ECTS: 5,0

Objectives

Provide an overview of physical computing and the Internet of Things (IoT).Promote knowledge of electronic concepts, electronic components and microcontrollers, with a focus on game development.Develop knowledge of microcontroller programming.Promote knowledge of how to develop a game using physical computing and IoT.

Learning Outcomes

Acquire basic knowledge of different electronic components, sensors and actuators.Acquire basic knowledge of building electronic circuits.Acquire knowledge of programming microcontrollers.Acquire the skills to use IoT in physical computing projects.Acquire the ability to design and develop a game using IoT and physical computing.

Course Contents

1. introduction to physical computing and IoT

1.2 Basics of electronics

1.2 Electronic components

1.3 Sensors and actuators

1.4 Microcontrollers and microprocessors: comparing the architecture of prototyping platforms: Arduino and RaspberryPi.

1.5 Internet of Things (IoT)

2. Programming microcontrollers

2.1 Introduction to the Arduino IDE

2.2 Introduction to Arduino programming (data types, loops, conditional structures, functions and procedures, etc.)

2.2 Assembly of simple circuits using a protoboard

3. Digital and analog I/O on Arduino

3.1 Configuring digital inputs and outputs in Arduino

3.2 Pulse length modulation (PWM)

3.3 Operating principle of an RGB LED and control of the color emitted from a PWM control signal

3.4 Operating principle of a DC motor and control of its rotational speed from a PWM control signal

3.5 Components of a servomotor, operating principle and motion control from a PWM control signal

3.6 Principle of operation of a piezo tweeter (Buzzer) and generation of sound signals with different frequencies and duration

4. The generic analog-to-digital conversion process

4.1 Analog inputs on the Arduino

4.2 Digitizing an analogue signal

4.3 Binary coding (3-bit example)

4.4 Characteristics of the A/D conversion module

4.5 Examples of how analog sensors work

4.6 A/D conversion in Arduino

4.7. Using the potentiometer as a position sensor

4.8 Example application with the development of a device driver for measuring angular rotation using a potentiometer

4.9 Example application with the brightness sensor based on an LDR

4.10 Example application with the temperature sensor

5. Event-driven programming in Arduino

5.1 Pooling vs. interrupts

5.2 Advantages of interrupts and their applications

5.3 Interrupt service process

5.4 Interrupt service routine (ISR)

5.5 Interrupts in Arduino

5.6 Digital external interrupts (falling edge, rising edge, etc)

5.7 Periodic interrupts using the "Timer.h" library

6. Object-oriented programming in Arduino

6.1 Thinking "Objects"

6.1 Introduction to OOP programming in Arduino - abstraction, encapsulation, inheritance and polymorphism

7. Practical project
7.1 Accompanying the practical project

Recommended Bibliography

McKerrow, P., Introduction to robotics, Addison-Wesley,1991

Craig ,J. J., Introdution to Robotics Mechanics and Control, Addison-Wesley Publishing Company, 1989

Jeremy, B., Exploring Arduino: Tools and Techniques for Engineering Wizardry, Wiley, 2013

Pfister, C., Getting started with the Internet of things, O'Reilly Media, Inc, 2011

Learning and Teaching Methods

Provide an overview of physical computing and the Internet of Things (IoT).

Electronics basics
Electronic components
Sensors and actuators
Microcontrollers and microprocessors: comparison of the architecture of prototyping platforms: Arduino and RaspberryPi.
Internet of Things

Promote knowledge of electronic concepts, electronic components and microcontrollers, with a focus on game development.

Electronics basics
Electronic components
Sensors and actuators

Develop knowledge of microcontroller programming.

2. Programming microcontrollers

2.1 Introduction to the Arduino IDE

2.2 Introduction to Arduino programming (data types, loops, conditional structures, functions and procedures, etc.)

2.2 Assembly of simple circuits using a protoboard

3. Digital and analog I/O on Arduino

3.1 Configuring digital inputs and outputs in Arduino

3.2 Pulse length modulation (PWM)

3.3 Operating principle of an RGB LED and control of the color emitted from a PWM control signal

3.4 Operating principle of a DC motor and control of its rotational speed from a PWM control signal

3.5 Components of a servomotor, operating principle and motion control from a PWM control signal

3.6 Principle of operation of a piezo tweeter (Buzzer) and generation of sound signals with different frequencies and duration

4. The generic analog-to-digital conversion process

4.1 Analog inputs on the Arduino

4.2 Digitizing an analogue signal

4.3 Binary coding (3-bit example)

4.4 Characteristics of the A/D conversion module

4.5 Examples of how analog sensors work

4.6 A/D conversion in Arduino

4.7. Using the potentiometer as a position sensor

4.8 Example application with the development of a device driver for measuring angular rotation using a potentiometer

4.9 Example application with the brightness sensor based on an LDR

4.10 Example application with the temperature sensor

5. Event-driven programming in Arduino

5.1 Pooling vs. interrupts

5.2 Advantages of interrupts and their applications

5.3 Interrupt service process

5.4 Interrupt service routine (ISR)

5.5 Interrupts in Arduino

5.6 Digital external interrupts (falling edge, rising edge, etc)

5.7 Periodic interrupts using the "Timer.h" library

6. Object-oriented programming in Arduino

6.1 Thinking "Objects"

6.1 Introduction to OOP programming in Arduino - abstraction, encapsulation, inheritance and polymorphism

Promote knowledge for the development of a game using physical computing and IoT.

7. Practical project
7.1 Accompanying the practical project

Assessment Methods

Distributed evaluation without final exam.

Final grades will be calculated taking into account the following criteria:
10% - Participation and Attendance
35% - Pre-Project
55% - Final Project

- The non-delivery of the pre-project or the final project will be classified with 0 (zero). - Works delivered after the deadline will have a 20% penalty with a 24 hour delivery limit. Beyond that time it is considered undeliverable.