Miami Dade College

Course Description

CEN2211 | C/C++ Programming for Embedded Devices | 4.00 credits

This course teaches the principles of programming in the C/C++ languages for embedded devices. The student will learn how to create programs to control open-source hardware for building digital devices that can sense and control the physical world around them.

Course Competencies

Competency 1: The student will demonstrate an understanding of electronic circuits foundations by:

1. Defining the fundamental concepts of electricity, including current, voltage and resistance
2. Explaining Ohm's Law and how it relates voltage, current, and resistance in an electrical circuit
3. Identifying common units of measurement used in electronics (amps, volts and ohms) and describing how they are measured
4. Describing the role of basic components in circuits, including resistors, capacitors, diodes transistors, and inductors
5. Understanding the difference between Direct Current (DC) and Alternating Current (AC) and their applications in electronics

Competency 2: The student will demonstrate an understanding of basic programming concepts using C++ by:

1. Describing the basic structure of a C++ program, including functions, variables, and data types
2. Understanding different data types such as integers, floating-point numbers, and characters, and when to use them
3. Demonstrating the use of arithmetic operators (e.g., +, -, *, /) and relational operators (e.g., ==, !=, >, <)
4. Writing simple programs that include user input, process data, and display output to the console
5. Applying conditional statements (if, else if, else) to create decision-making logic in programs
6. Writing reusable functions and using them to simplify code by modularizing tasks

Competency 3: The student will demonstrate an understanding of control structures and data handling in C++ by:

1. Implementing loops (for, while, and do-while) to repeat actions in a C++
2. Using arrays to store and manipulate collections of data within a program
3. Debugging C++ programs to identify logical and syntactical errors using systematic testing methods
4. Applying common libraries and functions in C++ to extend functionality (e.g., cmath for mathematical functions)

Competency 4: The student will demonstrate an understanding of programming embedded development boards using C++ by:

1. Explaining what an embedded development board is and how it functions in embedded systems (e.g., Arduino)
2. Writing C++ programs to control the inputs and outputs on a development board, such as turning LEDs on/off or reading sensor data
3. Using functions like digitalWrite(), digitalRead(), analogWrite(), and analogRead() to interact with hardware components on the development board
4. Uploading programs to a development board and using real-time feedback (e.g., serial monitor) to troubleshoot and debug hardware interactions
5. Developing programs that respond to environmental changes, such as controlling an LED based on sensor data (e.g., turning it on when it is dark)

Competency 5: The student will demonstrate an understanding of basic electronics and circuit design by:

1. Identifying and explaining the function of basic electronic components such as resistors, capacitors, LEDs, transistors, and integrated circuits

2. Designing and assembling simple circuits using breadboards, connecting components in series and parallel configurations
3. Using a multimeter to measure and verify circuit properties such as voltage, current, and resistance
4. Applying Ohm's Law to calculate the necessary values for components in simple circuits

Competency 6: The student will demonstrate an understanding of interfacing sensors and actuators with embedded systems by:

1. Defining sensors and actuators and explaining their roles in embedded systems
2. Writing C++ programs to collect data from common sensors (e.g., temperature, light, proximity) and process that data to create automated responses
3. Controlling actuators such as motors, LEDs, and speakers using Pulse Width Modulation (PWM) techniques to create varying levels of output
4. Designing and building simple projects that integrate sensors and actuators, creating interactive systems that respond to real-world inputs
5. Modifying programs to adjust sensor thresholds and actuator behaviors based on input data, fine-tuning the system's response to environmental changes

Competency 7: The student will demonstrate an understanding of basic circuit assembly and soldering techniques for prototype building by:

1. Setting up a basic electronics workspace, ensuring safety and organization for circuit assembly
2. Identifying the tools required for circuit assembly, including breadboards, wires, multimeters, and soldering irons
3. Performing basic soldering techniques to connect components to circuit boards safely and accurately.
4. Building circuits using breadboards to prototype and test designs without soldering
5. Using a multimeter to diagnose issues in circuits, verifying correct component connections and functionality
6. Assembling working circuits that incorporate both sensors and actuators for functional embedded prototypes

Learning Outcomes:

- Use quantitative analytical skills to evaluate and process numerical data
- Solve problems using critical and creative thinking and scientific reasoning
- Use computer and emerging technologies effectively