Integrating IoT in Mechatronics Lab for Mechanical Engineering Technology Curriculum: Embracing Industry 4.0

Jiayue Shen, Daniel Jones, Kazi Imran, SUNY Polytechnic Institute; Xiangyu Wang, Purdue University Fort Wayne; Weiru Chen, Slippery Rock University; Lanju Mee, University of Maryland Eastern Shore

Abstract

In the context of Industry 4.0, mechatronics labs are increasingly incorporating Internet of Things (IoT) technologies to enhance the teaching of system control and monitoring concepts. This paper presents the development of nine lab modules to integrate IoT technologies into the mechatronics lab for the first mechatronics course in Mechanical Engineering Technology (MET). The lab modules provided students with practical experience in using IoT technologies such as MQTT, ThingSpeak, and Simulink to design and control mechatronic systems. The modules covered a range of topics, including motor control, feedback control, and system modeling and simulation. The course provided students with a strong foundation in the theoretical concepts of mechatronics, which were then reinforced through the hands-on lab modules. The success of the course is reflected in the positive feedback from students, who appreciated the practical skills gained through the lab modules. Moving forward, the course will evolve to meet the changing needs of students and industry, ensuring that graduates are well-prepared to enter the workforce and contribute to developing IoT-enabled mechatronic systems. Overall, the integration of IoT in the mechatronics lab prepares students for Industry 4.0, empowering them to design and optimize IoT-driven mechatronic systems and contribute to the evolving field of modern engineering.

1. Introduction

As a multidisciplinary field, mechatronics has emerged as an essential component of modern engineering systems. It involves the integration of mechanical, electrical, and computer engineering to design and develop intelligent systems that perform complex tasks efficiently. In recent years, mechatronics has gained significant attention in MET programs as it offers a unique approach to teaching fundamental concepts of control systems and automation. Several studies have shown the benefits of incorporating mechatronics into the MET curriculum. [1-3] . However, due to the complexity of mechatronic systems, students can struggle to grasp the concepts through theoretical lectures and traditional laboratory experiments.

The Internet of Things (IoT) has revolutionized the field of mechatronics by enabling connectivity between various mechanical and electronic devices. The integration of IoT in mechatronics systems has led to enhanced automation and control, improved monitoring and maintenance, and increased efficiency and productivity. The application of IoT in mechatronics has become an active research area in recent years, with numerous studies exploring its potential and practical implementations. For instance, both Sunday A. Afolalu et al. and David Bradley et al. [4], [5] investigated the integration of IoT in mechatronics systems for enhanced automation and real-time monitoring. Similarly, Mutaz Ryalat et al. [6] proposed an IoT-enabled mechatronics system for smart manufacturing. The promising results of these studies demonstrate the potential of IoT in mechatronics, and highlight the importance of incorporating IoT technologies in mechatronics education [7].

The objective and purpose of integrating IoT in a mechatronics lab for the first MET mechatronics course is to enhance students' understanding and application of mechatronics principles by providing them with hands-on experience with IoT devices and technologies. This approach aligns with the current trend in engineering education that emphasizes the integration of emerging technologies into the curriculum to prepare students for the demands of the modern engineering industry [7]. The integration of IoT in a mechatronics lab can provide opportunities for students to develop skills in sensor data acquisition, data analysis, and control system design, which are crucial for success in mechatronics-related careers [8]. Section 2 gives an overview of the course, while Section 3 presents detailed information about the lab modules. Section 4 explores student feedback and suggestions for potential improvements to the course. Finally, Section 5 concludes the paper.

2. Course Overview

2.1 Course Description

A new course on mechatronics has been developed and offered as a junior-level elective course for MET students. This course provides an introduction to the principles and applications of mechatronics, a multidisciplinary field that combines mechanical, electrical, and computer engineering to design and develop intelligent systems. The course covers topics such as sensors and actuators, control systems, programming languages, and microcontroller-based design. Students will have the opportunity to work on a variety of hands-on projects, including the design and implementation of a simple mechatronic system.

2.2 Course Textbook

Textbook: *Introduction to Mechatronics and Measurement Systems,* D. Alciatore & M. B. Histand, McGraw Hill, 5th Edition, 2019.

Reference Book: 1. *Introduction to Mechatronic Design*, J. Edward Carryer, R. Matthew Ohline, Thomas W. Kenny, Prentice-Hall, 2011; 2. Control Systems Engineering, Nise, Norman S, John Wiley, 8th edition, 2019.

2.3 Objectives

The overall objectives of the course include, but are not limited to,

- Understand the fundamental principles and concepts of mechatronics
- Identify and select appropriate sensors and actuators for a given system
- Develop control algorithms for mechatronic systems
- Use microcontrollers for mechatronic system design
- Apply programming skills in mechatronic applications

The laboratory activities have been developed to partially or fully fulfill certain student learning outcomes for the MET program outlined by ABET, which include

- Apply knowledge, techniques, skills, and modern tools of mathematics, science, engineering, and technology to solve broadly-defined engineering problems that are relevant to the discipline.
- Design systems, components, or processes that meet specified needs for broadly-defined engineering problems that are appropriate to the discipline.
- Utilize written, oral, and graphical communication effectively in both technical and non-technical environments and identify and employ relevant technical literature as needed.
- Conduct standard tests, measurements, and experiments, and analyze and interpret the results to improve processes.
- Demonstrate the ability to function as an effective member and leader of technical teams.

In addition to theoretical lessons on mechatronics, this course incorporates IoT technologies to provide students with practical, hands-on experience. The laboratory activities encompass

- Provide students with training on Arduino IDE.
- Review the laboratory manual and watch related tutorials.
- Conduct the experiments.
- Gather data and compare results.
- Collaborate with team members to write laboratory reports and submit them for assessment.

2.4 Course Outlines

Table 1 provides an overview of the topics covered on a weekly basis in both the theory and laboratory classes. With a total of four credit hours, students will spend three hours each week in lecture and one hour in laboratory activities.

2.5 Assessment

Students will be evaluated based on their performance in various activities throughout the course. These activities include homework assignments, mid-term and final exams, laboratory reports, and a final project. Each of these activities is weighted equally at 20% of the final grade. This assessment structure is designed to ensure that students' understanding of the course material is tested in multiple ways, including theoretical knowledge and practical application in the laboratory. The laboratory reports and final project provide opportunities for students to apply the concepts learned in class to real-world situations and demonstrate their understanding of the subject matter. By distributing the weightage equally across all the assessment activities, students are motivated to perform well in each area, thereby achieving a comprehensive understanding of the course material.

Table 1. Contents of theory and laboratory classes.

Week	Theory Class	Laboratory Activities
1	Introduction to Mechatronics and IoT	Setting Up the Arduino Board, Arduino IDE, and Adding Libraries
2	Introduction to Sensors	Analog sensor and Digital sensor measurements
3	Introduction to Actuators	Controlling Servo motor and Two TT DC Motors
4	Communication Protocols	Implementing Wi-Fi communication between the Arduino and a computer using ESP8266 module
5	Data Acquisition and Analysis	Data Acquisition and Analysis using DHT11 Sensor
6	Mid-term Review and Exam	N/A
7 8	Cloud Platforms for IoT	Connecting the Arduino to the cloud using the MQTT protocol and ThingSpeak
9	Introduction to Control Systems	Controlling the position of a servo motor using feedback control
10 11	Feedback Control	Implementing PID control for a DC motor
12	System Modeling and Simulation	Modeling and Simulation of a Mechatronic System Using Simulink
13	Final Project: e.g. Autonomous Obstacle Avoidance Vehicle, candy color sorters, robotic arms, etc.	
14	Final Review and Exam	N/A

3. Lab Modules

The mechatronics laboratory is built upon an affordable IoT starter kit, the SunFounder Ultimate Starter Kit Compatible with Arduino, and utilizes MATLAB/Simulink. This comprehensive kit contains all the necessary components for conducting various aspects of the mechatronics lab, including a control architecture (1x controller board compatible with Arduino UNO and 1x 74HC595), ten different types of sensors (such as ultrasound, infrared obstacle avoidance, and photoresistor), and nine types of electric actuators (including DC TT motors, servo SG90 motor, and a passive buzzer). Additionally, the starter kit includes basic electronic parts, such as an ESP8266 Wi-Fi module, breadboard, and wires, to support various lab activities.

The lab activities are organized weekly and cover a range of topics such as motor control, sensor calibration, and data acquisition. Additionally, the lab activities involve programming with Arduino and the use of Simulink for system modeling and simulation. Students are assessed based on their performance in the lab activities, laboratory reports, and a final project. The use of the IoT starter kit allows for a cost-effective and practical implementation of mechatronics concepts, preparing students for a successful career in the field. Below are the details of each lab module:

3.1 MODULE 1: Setting Up the Arduino Board, Arduino IDE, and Adding Libraries

Objectives: The objective of this lab exercise is to introduce students to the Arduino development environment, including setting up the Arduino board, installing the Arduino integrated development environment (IDE), and adding libraries for programming. The basic knowledge of the Arduino development environment is essential for future mechatronics laboratory activities involving Arduino-based systems.

Equipment: The equipment required for this lab exercise includes an Arduino board, a USB cable, a computer with internet access, the Arduino IDE downloaded from the official Arduino website, and the required libraries also downloaded from the official Arduino website. These components are essential for setting up the Arduino board and programming it using the Arduino IDE.

Approaches: The lab exercise involves several approaches to achieve the objective of introducing students to the Arduino development environment. First, students will set up the

Arduino board by connecting it to a computer using a USB cable and installing the necessary drivers if required. Next, they will install the Arduino IDE by downloading it from the official Arduino website and following the installation instructions for their operating system. Then, students will add libraries for programming by downloading the required libraries from the official Arduino website and extracting them to the "libraries" folder in the Arduino IDE installation directory. They will then select the appropriate library in the Arduino IDE under "Sketch" > "Include Library." After setting up the environment, students will write a simple program, such as blinking an LED, in the Arduino IDE. They will then verify the program by clicking on the "Verify" button and upload it to the Arduino board by clicking on the "Upload" button. Finally, students will observe the behavior of the Arduino board, such as the LED blinking. By following these approaches, students will gain a basic understanding of the Arduino development environment and be better prepared for future mechatronics laboratory activities involving Arduino-based systems.

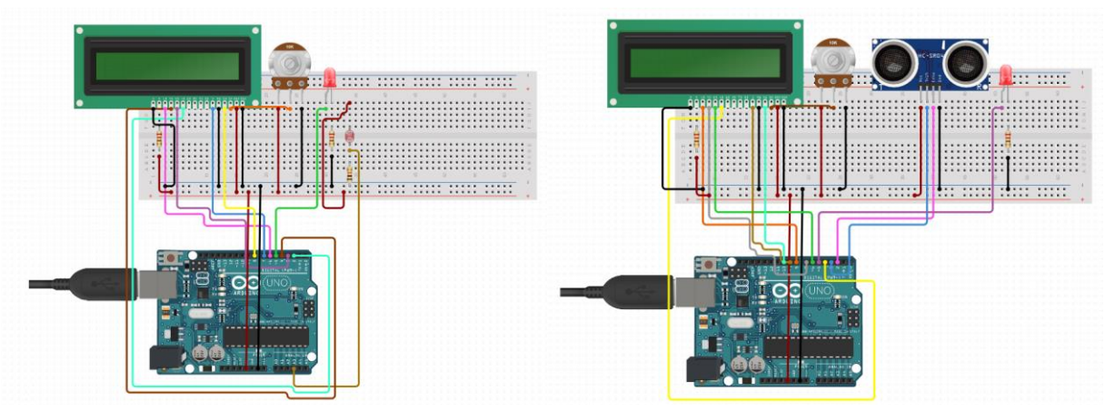
3.2 MODULE 2: Analog Sensor and Digital Sensor Measurements

Objectives: The objectives are to provide students with hands-on experience in working with different types of sensors and to familiarize them with the Arduino IDE for data acquisition and processing. The lab activities aim to enhance students' understanding of the basic principles of analog and digital sensors and their applications in mechatronics. Additionally, the lab experiments encourage students to apply their knowledge and skills in programming, circuit design, and data analysis to solve real-world problems related to mechatronics systems.

Equipment: The analog sensor experiment used a photoresistor to measure light levels, and the data were displayed on an LCD screen. This experiment required an Arduino UNO board, breadboard, photoresistor, 10KΩ resistor, I2C LCD 1602 module, and jumper wires. On the other hand, the digital sensor experiment used an ultrasonic sensor to measure distance, and an LED was activated when the distance was less than a certain threshold. The digital sensor experiment required an Arduino UNO board, breadboard, ultrasonic sensor, LED, 220Ω resistor, and jumper wires. Both experiments involved connecting components on the breadboard and programming the Arduino using the Arduino IDE. The code for each experiment was provided, and the results were observed through the LCD screen and LED, respectively.

Approaches: In Fig.1(a), the analog sensor experiment involved connecting a photoresistor and a 10KΩ resistor to a breadboard, which were then connected to an analog pin of the Arduino UNO board. The I2C LCD 1602 module was also connected to the board using a separate set of jumper wires. The Arduino code provided involved reading the analog value of the photoresistor, displaying it on the LCD screen, and repeating the process with a delay of 500 milliseconds. The potentiometer on the I2C LCD 1602 module was adjusted to adjust the contrast of the display.

In Fig.1(b), the digital sensor experiment involved connecting an ultrasonic sensor and an LED to a breadboard. The ultrasonic sensor was connected to digital pins of the Arduino UNO board, and the LED was connected to a digital output pin with a 220Ω resistor. The provided Arduino code involved measuring the distance using the ultrasonic sensor, turning on the LED when the distance was less than a certain threshold, and displaying the distance on the serial monitor. The potentiometer on the I2C LCD 1602 module was not used in this experiment.



(a) (b)

Fig.1. Wiring schematic of (a) analog sensor measurement and (b) digital sensor measurement.

In both experiments, the Arduino IDE was used to program the Arduino UNO board with the provided code. The equipment used for each experiment was specific to the type of sensor being measured. The analog sensor experiment used a photoresistor and a 10KΩ resistor, while the digital sensor experiment used an ultrasonic sensor and an LED. The results were observed through the I2C LCD 1602 module in the analog sensor experiment and the LED and Serial Monitor in the digital sensor experiment.

3.3 MODULE 3: Controlling Servo Motor and Two TT DC Motors

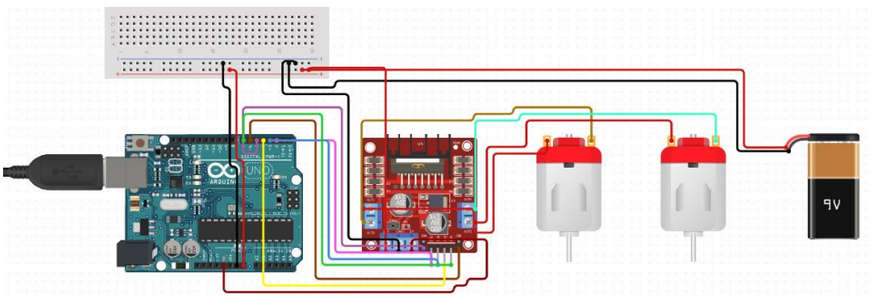
Objectives: The objectives are to familiarize the user with controlling two TT DC motors using an L298N motor driver module with an Arduino UNO board, and to provide a basic understanding of servo motors and how to control their position using an Arduino microcontroller. The lab aims to demonstrate the use of Arduino *Equipment:* in controlling different types of motors, which is a fundamental concept in mechatronics. By completing these experiments, the user will gain experience in connecting and programming the L298N motor driver module and servo motor, and learn how to adjust their speed and position using code.

In this module, two experiments are conducted. The first experiment involves controlling TT DC motors using an Arduino UNO board, L298N motor driver module, two TT DC motors, a 9V battery, jumper wires, and a breadboard. The second experiment involves controlling a servo motor using an Arduino UNO board, a servo motor, jumper wires, and a breadboard. Both experiments require the use of the Arduino IDE for programming the Arduino board. The L298N motor driver module is a critical component for controlling the TT DC motors, while the Arduino's servo library is used to control the servo motor. The jumper wires and breadboard are used to connect the components in the circuit.

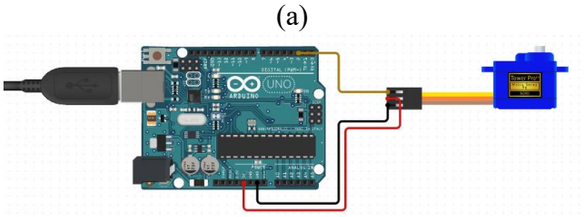
Approaches: This module involves two experiments that focus on controlling different types of motors using an Arduino microcontroller. The first experiment is for controlling two TT DC motors using an L298N motor driver module as shown in Fig. 2(a). To conduct this experiment, an Arduino UNO board, an L298N motor driver module, two TT DC motors, a 9V battery, jumper wires, and a breadboard are required. The L298N motor driver module provides bidirectional control for the motors and can handle high currents and voltages. The motor driver module is connected to the breadboard and the Arduino, and the TT DC motors are connected to the motor driver module. The control pins of the L298N motor driver module are connected to the digital pins of the Arduino. A code is written in the Arduino IDE to control the speed and direction of the motors, which is then uploaded to the Arduino board. By running the code, the TT DC motors can be observed moving in different directions at varying speeds.

The second experiment is for controlling a servo motor using an Arduino Fig. 2(b). To conduct this experiment, an Arduino UNO board, a servo motor, jumper wires, and a breadboard are required. The servo motor is connected to the breadboard and the Arduino, with its ground and power wires connected to the breadboard's ground and power rails, respectively, and its signal wire connected to digital pin, e.g., pin 2, of the Arduino. The servo library is included in the Arduino IDE, and a servo object is declared in the code. The servo is attached to digital pin in the setup() function, and the servo's position is controlled using the write() function in the loop() function. By varying the angle values in the write() function and uploading the code to the Arduino board, the servo motor can be observed moving to different angles.

Overall, both experiments demonstrate the use of Arduino in controlling different types of motors, which is a fundamental concept in mechatronics. The L298N motor driver module is crucial for controlling the TT DC motors, while the built-in servo library is used for controlling the servo motor.



(a)



(b)

Fig. 2. Wiring schematic of module 3: (a) controlling TT DC motors; (b) controlling servo motor.

3.4 MODULE 4: Implementing Wi-Fi Communication between the Arduino and a Computer Using ESP8266 Module

Objectives: The objective of the experiment is to learn how to establish Wi-Fi communication between an Arduino board and a computer using an ESP8266 Wi-Fi module. The experiment involves connecting the ESP8266 module to the Arduino, writing a code in Arduino IDE to establish the Wi-Fi connection, and testing the connection by sending data back and forth between the two devices. The goal is to gain a better understanding of how Wi-Fi communication works in the context of the Arduino and to acquire a useful skill for IoT applications that require wireless communication.

Equipment: The equipment required for the experiment includes an Arduino board, an ESP8266Wi-Fi module with a logic level converter (bi-directional), a breadboard, jumper wires, a USB cable, and a computer with the Arduino IDE installed. The Arduino board is the main controller for the experiment, while the ESP8266 Wi-Fi module provides wireless connectivity. The breadboard and jumper wires are used to connect the module to the Arduino board. The USB cable is used to power the Arduino board and to upload the code to the board from the computer. The computer with the Arduino IDE installed is used to write the code for the experiment and to monitor the communication between the Arduino board and the computer via the Wi-Fi network.

Approaches: The experiment focused on implementing Wi-Fi communication between an Arduino board and a computer using an ESP8266 Wi-Fi module. The first approach was to set up the hardware by connecting the ESP8266 module to the Arduino board according to the pin diagram. This was accomplished by using a breadboard and jumper wires to connect the power, ground, and data pins of the module to the corresponding pins on the Arduino board as shown in Fig.3.

Once the hardware was set up, the second approach was to write the code to establish a Wi-Fi connection between the Arduino and the computer. This was done using the Arduino IDE, where the necessary libraries for ESP8266 Wi-Fi communication were included in the code. The code used the ESP8266WiFi library to connect to a Wi-Fi network, and the network's SSID and password were provided in the code. The Arduino board tried to establish a connection with the network using the WiFi.begin() function, and the connection status was monitored using the while loop and printed to the serial monitor.

The third approach was to send and receive data between the Arduino and the computer. The code sent a string "Hello, computer!" to the computer using the Serial.println() function, and data was received from the computer using the Serial.available() and Serial.readString() functions. This data were then printed to the serial monitor using the Serial.println() function.

Finally, the experiment was tested by uploading the code to the Arduino board using the USB cable and monitoring the Wi-Fi connection status on the serial monitor. Data were then sent from the Arduino to the computer, and data were received from the computer by the Arduino. The results and observations were recorded for further analysis.

Overall, the experiment provided a useful introduction to implementing Wi-Fi communication between an Arduino board and a computer using an ESP8266 Wi-Fi module. The approaches used in the experiment included setting up the hardware, writing the code to establish the Wi-Fi connection, sending and receiving data between the two devices, and testing the connection. This experiment provided a useful foundation for further exploration of Wi-Fi communication and IoT applications that require wireless communication.

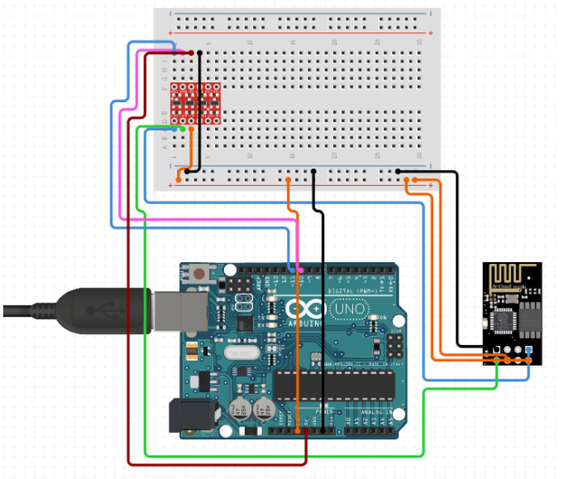


Fig.3. Wiring schematic of module 4.

3.5 MODULE 5: Data Acquisition and Analysis Using DHT11 Sensor

Objectives: The objective of this experiment is to demonstrate how to collect and analyze sensor data using an Arduino board and the DHT11 temperature and humidity sensor. The experiment involves connecting the sensor to the Arduino board, writing the code to read data from the sensor, displaying the data on the serial monitor, and analyzing the collected data using statistical methods such as mean, median, and standard deviation. This experiment provides a foundation for further exploration of data acquisition and analysis using sensors and the Arduino platform.

Equipment: The equipment needed for the experiment includes an Arduino UNO, a DHT11 temperature and humidity sensor, a breadboard, and jumper wires. These components are used to collect and analyze sensor data using the Arduino platform and statistical methods such as mean, median, and standard deviation. The DHT11 sensor is connected to the Arduino board using the breadboard and jumper wires, while the Arduino IDE is used to write the code for reading and displaying the collected data on the serial monitor.

Approaches: The experiment involved connecting the DHT11 temperature and humidity sensor to an Arduino board using a breadboard and jumper wires as shown in Fig. 4. The DHT11 library was uploaded to the Arduino IDE to provide functions for reading temperature and humidity data from the sensor. The code was then written to collect data from the sensor and display it on the serial monitor using the Serial.print() and Serial.println() functions.

Once the code was saved and uploaded to the Arduino board, the experiment involved opening the serial monitor to view the collected data. The temperature and humidity readings could be observed and analyzed, and a table was filled with the readings over time to better visualize the data. A graph was plotted to provide a visual representation of the data.

The collected data were then analyzed using statistical methods such as mean, median, and standard deviation to better understand the overall trends of the data. This provided a useful way to summarize and analyze the data collected by the sensor.

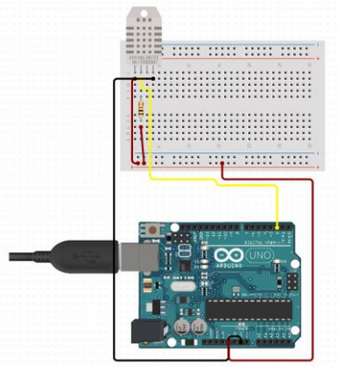


Fig. 4. Wiring schematic of module 5.

In conclusion, the experiment provided a foundation for understanding data acquisition and analysis using an Arduino board and the DHT11 sensor. The approaches used in the experiment included connecting the sensor to the Arduino board, writing the code to collect and display the data, visualizing the data using tables and graphs, and analyzing the data using statistical methods. These techniques can be applied to a wide range of sensor data collection and analysis applications in the future.

3.6 MODULE 6: Connecting the Arduino to the Cloud Using the MQTT Protocol and ThingSpeak

Objectives: The objective of this module is to teach how to connect an Arduino board to the internet using Wi-Fi or Ethernet shields, set up a ThingSpeak account, and use MQTT protocol to send sensor data to ThingSpeak. Additionally, this module aims to provide hands-on experience in visualizing and analyzing data on ThingSpeak, a cloud platform for IoT.

Equipment: The equipment required for this experiment includes an Arduino board, an ESP8266Wi-Fi module with a logic level converter (bi-directional), sensors such as DHT11, a breadboard, and jumper wires. Additionally, a computer with the Arduino IDE and an internet connection is necessary to set up and access the ThingSpeak account. The required libraries for MQTT and ThingSpeak need to be installed in the Arduino IDE.

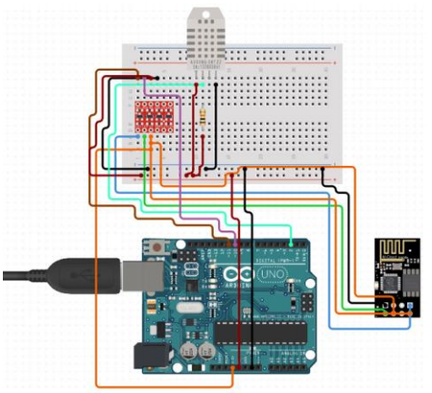


Fig. 5. Wiring Schematic of module 6.

Approaches: In this experiment, the Arduino board was connected to the cloud using the MQTT protocol and ThingSpeak to remotely monitor and control IoT devices. First, the ESP8266 Wi-Fi module was connected to the Arduino board and the Wi-Fi credentials were set up. Next, the required libraries for MQTT and ThingSpeak were installed in the Arduino IDE. The DHT11 sensor was connected to the Arduino board and the code was written to read the sensor data and publish them to the cloud using the MQTT protocol. The code was modified to include the ThingSpeak API key, which is a unique identifier that allows the Arduino to send data to the correct ThingSpeak channel. The wiring schematic is shown in Fig. 5.

After setting up a ThingSpeak account by following the online tutorial [9] and creating a channel to receive the sensor data, the code was uploaded to the Arduino board. The sensor data were successfully collected and published to the cloud using the MQTT protocol, and was displayed in real-time on ThingSpeak. The data were easily observed and analyzed, allowing for remote monitoring and control of the IoT device.

The experiment demonstrates the importance and usefulness of cloud platforms for IoT and provides hands-on experience in connecting an Arduino board to the cloud using the MQTT protocol and ThingSpeak. This technology has a wide range of applications in various fields, such as agriculture, healthcare, and industrial automation.

3.7 MODULE 7: Controlling the Position of a Servo Motor Using Feedback Control

Objectives: The experiment aims to teach students about feedback control and how to control the position of a servo motor using an Arduino board and an ESP8266 Wi-Fi module for wireless control. Students will learn practical skills such as programming the Arduino board, connecting and configuring the servo motor, and establishing a wireless connection using the ESP8266 Wi-Fi module. The objective is to provide students with a deeper understanding of mechatronic systems and the principles behind them.

Equipment: The equipment required for this experiment includes an Arduino board, a servo motor, a potentiometer, a breadboard, and jumper wires. The Arduino board serves as the control unit for the system, while the servo motor is the actuator that moves according to the input signal. The potentiometer is used as a feedback sensor to measure the output of the system and provide feedback for the control algorithm. The breadboard and jumper wires are used to connect and power the components. Additionally, an ESP8266 Wi-Fi module with a logic level converter (bi-directional) is used to establish a wireless connection for controlling the servo motor remotely. The equipment enables students to learn about mechatronic systems and develop practical skills such as programming, circuit design, and control system implementation.

Approaches: The first step in the experiment is to connect the potentiometer to the breadboard and then connect it to the Arduino board. Next, the servo motor is connected to the Arduino board and powered up. The ESP8266 Wi-Fi module is then connected to the Arduino board and configured to establish a wireless connection for controlling the servo motor remotely.

Once the hardware setup is complete, the students will develop the control algorithm for the servo motor using feedback control. The potentiometer is used as the feedback sensor to measure the output of the system and provide feedback to the control algorithm. The control algorithm adjusts the input signal to the servo motor based on the feedback received from the potentiometer. The ESP8266 Wi-Fi module is used to establish a wireless connection to control the servo motor remotely. The detailed wiring schematic is shown in Fig. 6.

After the control algorithm is developed, the students will upload it to the Arduino board and test the system. The students will then observe and analyze the behavior of the servo motor under different input conditions and adjust the control algorithm as needed to improve system performance.

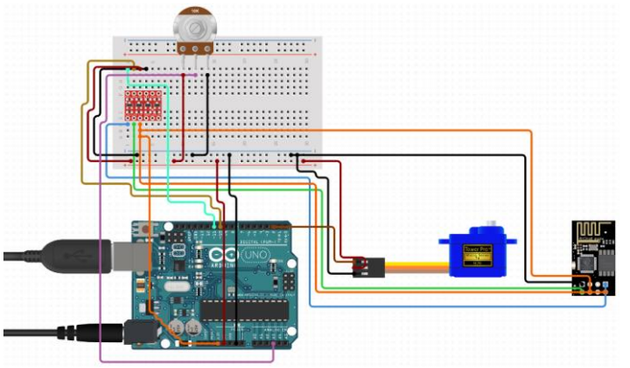


Fig. 6. Wiring Schematic of module 7.

Through this experiment, students will gain hands-on experience in programming the Arduino board, connecting and configuring the servo motor, and establishing a wireless connection using the ESP8266 Wi-Fi module. The experiment aims to provide students with a deeper understanding of mechatronic systems and the practical skills needed to design and implement such systems.

3.8 MODULE 8: Implementing PID Control for a DC Motor

Objectives: The objective of the experiment is to design and implement a PID control system using an Arduino and motor driver to regulate the speed of a DC motor. The experiment aims to demonstrate how a PID controller can maintain a stable and accurate motor speed with minimal overshoot and settling time by adjusting the PID gains and analyzing the system's response. The experiment involves setting up the circuit, defining the variables, initializing the PID controller, reading the potentiometer input, running the PID controller, and continuously adjusting the motor output to maintain the desired position. The performance of the system is evaluated by analyzing the overshoot, settling time, and steady-state error.

Equipment: The equipment required for the experiment includes an Arduino UNO board, an L298N motor driver module, a DC motor, a potentiometer, a breadboard, and jumper wires. The Arduino UNO board is used to control the motor, while the L298N motor driver module provides power to the motor and allows the Arduino to control its speed and direction. The potentiometer is used to provide input to the Arduino, and the breadboard and jumper wires are used to connect all the components together. Overall, the equipment allows for the implementation of a PID control system to regulate the speed of a DC motor using an Arduino and motor driver.

Approaches: The experiment involves designing and implementing a PID control system to regulate the speed of a DC motor using an Arduino and motor driver. To begin, the circuit is set up by connecting the DC motor to the Arduino using the L298N motor driver module, and a potentiometer is used to provide input to the analog pin A3 on the Arduino. The motor driver module is connected to a power source, and the pins connected to the motor driver, potentiometer, and motor are defined in the Arduino IDE, along with the variables for the PID controller, including the PID gains. The detailed wiring schematic is shown in Fig. 7.

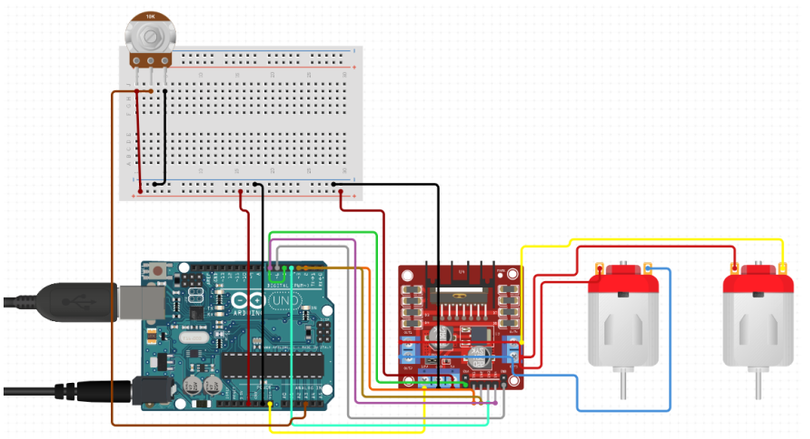


Fig. 7. Wiring schematic of module 8.

The PID controller is initialized with the PID gains and input/output limits. The input is read from the potentiometer and mapped to a value between 0 and 180 degrees to represent the desired motor position. The PID controller then calculates the output value needed to reach the desired position, and the output value is written to the motor driver, which adjusts the motor's speed and direction. This process is repeated continuously, with the PID controller adjusting the motor output to maintain the desired position.

The system’s performance is evaluated by analyzing the overshoot, settling time, and steady-state error. The PID gains can be adjusted to improve the system's performance, and the experiment can be repeated to verify the improvements. The response of the system is analyzed by plotting the input and output values and observing the system's behavior. Increasing the K_p gain reduces settling time but increases overshoot, while increasing the K_i gain reduces steady-state error but can lead to instability if set too high. Increasing the K_d gain reduces overshoot and oscillations but can slow down the system. The system's response can be optimized by adjusting the PID gains to achieve the desired performance. Overall, the experiment demonstrates the effectiveness of a PID control system in regulating the speed of a DC motor.

3.9 MODULE 9: Modeling and Simulation of a Mechatronic System Using Simulink

Objectives: The objective of the experiment is to model and simulate a mechatronic system using Simulink, a tool within MATLAB software, and analyze the system’s response to different input

conditions. The experiment aims to demonstrate the importance of modeling and simulation in the design and analysis of mechatronic systems, and to show how Simulink can be used to model the behavior of a system, adjust its parameters, and analyze its performance. The experiment involves creating a Simulink model, setting up an Arduino board, connecting the potentiometer and servo motor, and running the simulation to observe the system's response to changes in the input signal. The results of the experiment provide insights into the system's dynamics and performance, and suggest future experiments to optimize the system's behavior.

Equipment: The equipment used in the experiment includes MATLAB software with the Simulink module, an Arduino board, a breadboard, jumper wires, a potentiometer, and a servo motor. The MATLAB software with the Simulink module is used to model and simulate the mechatronic system, while the Arduino board provides input to the system. The potentiometer is used to adjust the input signal, and the servo motor is used as the output of the system. The breadboard and jumper wires are used to connect all the components together. Overall, the equipment allows for the simulation and analysis of the behavior of a mechatronic system under different input conditions.

Approaches: The experiment involves modeling and simulating a mechatronic system using Simulink and analyzing its response to changes in the input signal. To begin, a new Simulink model is created, and the "Simulink Support Package for Arduino Hardware" block is selected from the Simulink Library Browser. The "Arduino IO Setup" block is dragged and dropped into the model, and its port and board type are selected. The "Analog Input" block is also added to the model, and its pin number is selected. The "Scope" block is used to display the output of the system and is connected to the "Analog Input" block. The Simulink model is then saved.

In the next step, the "StandardFirmata" sketch is uploaded to the Arduino board using the Arduino IDE, and the board is connected to the computer. The Simulink model is then run, and the "Scope" block displays the analog input values from the Arduino board. By experimenting with different analog input pins and sample times, changes in the output can be observed. The system's response can be analyzed by plotting the input and output values and observing the system's behavior. The results show that the mechatronic system responds linearly to changes in the potentiometer input, and the response of the servo motor is proportional to the input signal. However, the system also exhibits overshoot and oscillations due to the inherent dynamics of the servo motor. The wiring schematic is very similar to the Fig. 6. The only difference is the ESP8266 module is removed, as shown in Fig. 8.

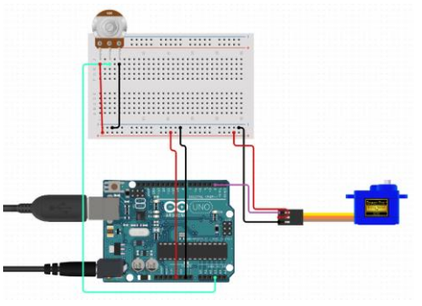


Fig. 8. Wiring schematic of module 9.

The experiment demonstrates the importance of modeling and simulation in the design and analysis of mechatronic systems. Using Simulink, the behavior of the mechatronic system can be modeled, its parameters can be adjusted, and its performance can be analyzed. The results of the experiment provide insights into the system's dynamics and performance, and future experiments can be conducted to optimize the system's behavior. Overall, the experiment highlights the effectiveness of Simulink as a tool for modeling and simulating mechatronic systems, providing a valuable tool for engineers to design and analyze complex systems.

Typically, the modules consisted of interactive sessions that combined both theory and practical applications. While there were instances where the intentional destruction of components was allowed as a means of teaching, the instructor closely monitored student activities and provided guidance to prevent any unintended damage to the equipment. Moreover, the instructor offered theoretical explanations of most observations as necessary during the session. However, there were certain theoretical concepts that could not be directly observed using these modules alone.

In such cases, the instructor would wrap up the session by discussing those unobserved theoretical concepts.

At the end of the program, the students were assigned to design an IoT-based mechatronic system capable of performing a specific task for their term project. The available project options involved creating a control system for a robotic arm, a line-tracking vehicle, a vending machine dispenser system, an autonomous obstacle avoidance vehicle, and other projects with comparable aims. The specifics of the projects and evaluation criteria will be reported in the future.

4. Student Response and Potential Improvements

The course is rated 4.6 out of 5. The feedback for the course has been generally positive, with students finding Arduino programming to be a suitable language for beginners and appreciating its integration into the course's tight schedule. However, the grading system has garnered mixed reviews from students, with some fast learners liking it and slow learners finding it discouraging. While there are no plans to significantly change the grading approach, the speed component of grading will be reduced to bridge the gap between fast and slow learners. To further enhance the course, the Arduino programming module will be simplified and given additional coverage time, allowing students to better understand the necessary skills. Students will also be encouraged to save their programs for future use to avoid the frustration of having to rewrite programs repeatedly. To alleviate the challenges that arise from more complex programs, sample codes will be provided to save program time during classes. By implementing these measures, the course will be better optimized for the learning needs of all students and foster a more supportive learning environment.

5. Conclusion

The paper describes the development of nine lab modules that integrate IoT technologies in the mechatronics lab for the first MET mechatronics course. These lab modules provided practical experience for students to design and control mechatronic systems using IoT technologies like MQTT, ThingSpeak, and Simulink. The lab modules covered topics such as motor control, feedback control, and system modeling and simulation.

The course began with theoretical concepts of mechatronics, which were then reinforced through the lab modules. Although the paper did not discuss the theoretical content, it provided an essential foundation for the practical application of IoT technologies in mechatronics.

The course was successful in equipping students with a comprehensive understanding of mechatronics and IoT technologies. The students appreciated the hands-on nature of the lab modules and the practical skills they gained. Moving forward, the course will continue to evolve to meet the changing needs of students and the industry, preparing graduates to contribute to the development of IoT-enabled mechatronic systems.

References

[1] G. Marzano, A. Martinovs, and S. Ušča, "Mechatronics education: needs and challenges," in *ENVIRONMENT. TECHNOLOGIES. RESOURCES. Proceedings of the International Scientific and Practical Conference*, 2019, vol. 2, pp. 214-217.

[2] I. M. R. Ogli and T. A. I. Ogli, "A Role of Mechanical Engineering in Mechatronics," *JournalNX*, pp. 824-828, 2021.

[3] U. S. Dixit, M. Hazarika, and J. P. Davim, *A brief history of mechanical engineering*. Springer, 2017.

[4] S. A. Afolalu, O. M. Ikumapayi, A. Abdulkareem, S. B. Soetan, M. E. Emetere, and S. O. Ongbali, "Enviable roles of manufacturing processes in sustainable fourth industrial revolution–A case study of mechatronics," *Materials Today: Proceedings*, vol. 44, pp. 2895-2901, 2021.

[5] D. Bradley, D. Russell, I. Ferguson, J. Isaacs, A. MacLeod, and R. White, "The Internet of Things–The future or the end of mechatronics," *Mechatronics*, vol. 27, pp. 57-74, 2015.

[6] M. Ryalat, H. ElMoaqet, and M. AlFaouri, "Design of a Smart Factory Based on Cyber-Physical Systems and Internet of Things towards Industry 4.0," *Applied Sciences*, vol. 13, no. 4, p. 2156, 2023.

[7] M. Bhuvanesh Kumar, N. Balaji, S. Senthil, and P. Sathiya, "Role of IoT in Product Development," *Integration of Mechanical and Manufacturing Engineering with IoT: A Digital Transformation*, pp. 215-234, 2023.

[8] T. Bunk, S. Sheppard, and H. Chen, "Exploring the impact of project-based mechatronics course design on alumni's entrepreneurial career pathways," in *2022 ASEE Annual Conference & Exposition*, 2022.

[9] Robotic App. ThingSpeak Account Creation, https://www.roboticapp.com/thingspeakaccountcreation.html, last accessed 2023/09/07.

Biographies

JIAYUE SHEN, PhD**,** is an assistant professor within the College of Engineering at SUNY Polytechnic Institute. Her research pursuits center on wireless sensing technology, flexible electronics, sensor development, and their multifaceted applications, encompassing biomedical diagnostics, robotics and structural health monitoring. Dr. Shen has assumed pivotal roles as principal investigator and senior personnel for over 10 research and teaching projects sponsored by diverse external agencies. Her prolific scholarly output encompasses 25+ publications spanning journals and conference proceedings. Beyond her research, she actively engages in conference committees and extends her expertise as a diligent reviewer for esteemed journals and conferences in her specialized field.

DANIEL K. JONES, PhD, PE, is an associate professor of Mechanical Engineering Technology SUNY Poly in Utica, NY. He teaches a variety of courses including mechanical components, advanced machine design, mechanical measurements, vibrations analysis, and capstone experience. He has established a state-of-the-art EEG laboratory and is collaborating with multi-disciplinary researchers. His research interests also include the development of assistive technology for people with physical disabilities.

KAZI IMRAN, Ph.D., earned his PhD in Mechanical Engineering from North Carolina A&T State University. He is currently Assistant Professor at the State University of New York Polytechnic Institute (SUNY Poly). Prior to joining the State University of New York Polytechnic Institute team, Dr. Imran worked as Assistant Professor at the New York City College of Technology, CUNY. He also has more than 5 years of industrial experience in thermal power plant and HVAC system. Dr. Imran's research focused on design, analysis, manufacturing, characterization of advanced multifunctional composites and nanocomposite materials. His teaching expertise includes in the area of Solid Mechanics, Machine Design, Advanced Computer Aided Manufacturing (CAD/CAM/CNC), Finite Element Analysis (FEA), Advanced Solid Modeling, Thermodynamics, Material Science and Failure Analysis.

XIANGYU WANG, PhD, is an assistant professor of Mechanical Engineering Technology in the School of Polytechnic at Purdue University Fort Wayne. He received his PhD in Mechanical Engineering at Old Dominion University. His research interest focuses on several areas within robotics, including Collaborative Robotics, Bio-inspired Robotics, Characterization of robotics applications, robotics rehabilitation, and robotics education. His research endeavors aim to advance the understanding of human-robot collaboration and enhance collaborative strategies in complex contexts, particularly in tasks characterized by non-repetitiveness and low structural mechanical impedance.

WEIRU CHEN, PhD, is an associate professor of Information Systems and Technology Management at Slippery Rock University of Pennsylvania. He received a PhD degree in Information Technology (AACSB-accredited) and an M.S. degree in Electrical and Computer Engineering (ABET-accredited) from Old Dominion University. His current research focuses on blockchain, cybersecurity, artificial intelligence, and big data. With over a decade of professional experience spanning both industry and academia, Weiru has served as a principal investigator or senior personnel on numerous projects funded by external funding agencies. He boasts a portfolio of more than 20 technical articles published in peer-reviewed journals and conference proceedings.

LANJU MEI, PhD, received her BS degree in Mechanical Engineering from Nanjing University of Aeronautics and Astronautics. She received her PhD degree from the Department of Mechanical and Aerospace Engineering, Old Dominion University, in 2019. She is currently an assistant professor at University of Maryland Eastern Shore. Her research interests focus computational fluid dynamics, micro/nanofluidics, electrokinetics, non-Newtonian fluid. She is the author of 20+ peer-reviewed journal papers.