**PROCTECH 4MH3**

**Course Project:**

**Vibration Monitoring System**

**Group 15**

**Group members:**

**Gao-Jyun Liang 400263076**

**Steven Steven 400271959**

**Abdulraman Hajjaj 400265450**

**Introduction (How It Works)**

The main goal of this project is to design a vibration monitoring system that integrates hardware and software components for data collection and analysis. The project mainly includes the ADXL335 accelerometer for measuring vibrations, the Arduino Uno microcontroller for data processing, and the Bluetooth module for wireless transmission. With these components, we successfully collected vibration data, analyzed it, and generated a condition report.

Python scripts were used for data recording, visualization, and report generation. Through Fast Fourier Transform (FFT) and time-domain metrics, the system achieves real-time monitoring and detailed analysis. This project has the potential to detect machine health conditions and provide health reports to help monitoring personnel easily understand the current status of the equipment.

**Operation Process**

A diagram of a process

Description automatically generated

Figure 1 Operation Flowchart

Our vibration monitoring system combines hardware and software to achieve real-time data collection, vibration data transmission, and data analysis. The system starts with the ADXL335 accelerometer, which measures vibrations in the XYZ directions and outputs analog signals to represent real-time data. The sensor's high sensitivity allows it to capture small and stable vibrations, making it more accurate in identifying abnormalities.

The analog signal is sent to the Arduino Uno microcontroller, which acts as the core of data processing. The Arduino Uno converts the analog signal into digital data and formats it for transmission. It communicates with the MH-10 Bluetooth module via UART, and the processed vibration data is wirelessly transmitted to a computer for further processing. Wireless communication increases the system's flexibility and makes it easier to set up.

On the software side, Python scripts handle the received data. The scripts log the data into CSV files with timestamps and acceleration values. Then, the data in these CSV files is analyzed using methods like frequency-domain and time-domain analysis. FFT extracts frequency components to detect abnormalities, while RMS and P2P metrics provide insights into vibration characteristics to assess the condition of the equipment.

Finally, the processed data is summarized into condition reports. These reports include information about the equipment's health, making it easier for monitoring personnel to identify the equipment's status.

**Block Diagram Overview:**

一張含有 圖表, 文字, 螢幕擷取畫面, 行 的圖片

自動產生的描述

Figure 2 Initial Design

Our project uses a wooden board as a stable base. The front part of the board has the Arduino Uno fixed with screws, while the sensor is placed in front of the Arduino Uno. A magnet is attached to the bottom of the sensor to ensure it can stay firmly on the test bed during testing and measure vibration data directly. The back part of the board is used to secure the battery and the Bluetooth module. The Bluetooth module is positioned within the signal range and is responsible for wirelessly transmitting the data to the computer.

**System Design**

**Pin Connection:**

一張含有 文字, 螢幕擷取畫面, 數字, 字型 的圖片

自動產生的描述

Figure 3 PIN designation

**Physical**

**Arduino Uno**

In this project, we chose to use the Arduino Uno as the microcontroller because of its simplicity and flexibility, making it easier for us to use. As the core of data processing, the Arduino Uno collects vibration data from the ADXL335 accelerometer and sends it to the Bluetooth module for wireless communication. Using the Arduino IDE for programming allows us to easily adapt it to different project needs. Additionally, since the system is wireless, a 12V battery is used to provide power and ensure stability.

一張含有 電子產品, 電子元件, 電路元件, 電子工程 的圖片

自動產生的描述

Figure 4 Arduino UNO

**Sensor (ADXL335)**

We chose the ADXL335 accelerometer to measure vibration data in the XYZ directions. This sensor outputs analog signals, allowing us to effectively collect real-time vibration data. Its ±2g sensitivity range is ideal for detecting small and stable vibrations, which is important for analyzing machine conditions. The sensor is small and lightweight, making it easier to combine with other components and reduce the overall size of the project to ensure our project is within the limited size range.

一張含有 電路元件, 電子產品, 電子元件, 電子工程 的圖片

自動產生的描述

Figure 5 Accelerometer ADXL335

**Bluetooth Module (MH-10)**

In this project, we decided to use the MH-10 Bluetooth module for wireless data transmission. The module connects to the Arduino Uno via UART and sends the processed vibration data to a computer for analysis. The MH-10 supports Bluetooth Low Energy (BLE), ensuring low power consumption and stable communication over short distances. We chose this module because it is easy to integrate, and our original plan was to transmit data wirelessly. Its small size also makes it easy to fit into the system design.

一張含有 電子產品, 電子元件, 電路元件, 被動元件 的圖片

自動產生的描述

Figure 6 BLE4.0 HM-10

**Battery**

In this project, we used a 12V battery to power the Arduino Uno and other components, providing a stable and portable power source. This solved the problem of relying on a fixed power supply, making testing more flexible and helping us achieve the goal of wireless transmission.

**Chassis**

The chassis is designed to be affordable, noise-proof and customizable. We separated it into two main parts:

*The main body:*

In the main body part, to fit into the size constraint of the project, we decided to have the Arduino stand up sideways. The controller is screwed on to a wood piece, which has the battery, BLE module, and the upper part of the magnets strapped on with rubber-bands for the convenience of prototyping and customizability.

*sensor body:*

As for the sensor body, we attached the accelerometer to a wooden piece with minimum thickness that we can find, with plastic in between for ESD protection, and attached directly to the magnet. This allows the vibration to transmit through the magnet and the wood piece with minimum damp, straight into the accelerometer.

The goal of this separation is to decrease potential dampening from the big main body, and to fit into the size constraints of the project.

**Project Showcase**

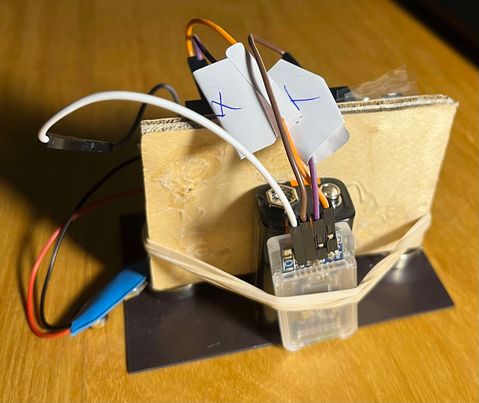
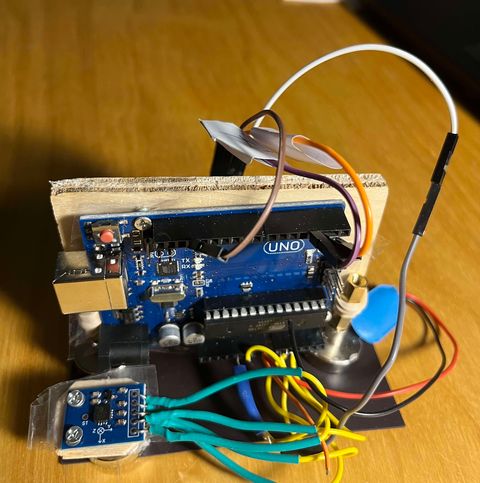


Figure 7 Front View of Vibration Monitoring Device Figure 8 Side View of Vibration Monitoring Device

**Software**

**Slave Side (Vibration Monitoring Device)**

The Arduino Uno on the slave side is programmed using C in the Arduino IDE. The main function of this software is to monitor analog signals from the ADXL335 accelerometer and transmit them, in digital form, through a serial connection to the Bluetooth module (MH-10). This ensures accurate and consistent data transmission for further processing.

**Master Side (Data Logging and Processing Device)**

We used a computer to receive, process and store the signals, from our Vibration Monitoring Device, with Python. Reason being that it is of most convenience for prototyping. 3 Scripts are written for this project that is to be run separately.

1. **BT\_Receiver.py**  
   This module establishes a connection with the Bluetooth device, logs the raw data into CSV format, and re-uploads the data via TCP for further processing. Additionally, it can be modified to send commands back to the vibration monitoring unit, allowing synchronization of data collection with machine operation to reduce redundancy.
2. **Visualizer.py**  
   This script functions as a TCP server, listening to the data sent by BT\_Receiver.py. Its primary task is to plot the data in real-time for visualization. Future functionality may include expanding its capabilities to handle real-time data processing or triggering emergency stops based on critical conditions.
3. **Report\_Generator.py**  
   Once the data collection process is complete, this script reads the CSV files generated by BT\_Receiver.py and performs detailed analysis. It uses Fast Fourier Transform (FFT) to analyze the frequency-domain attributes of the data and calculates key metrics, such as RMS and Peak-to-Peak values, to assess the system’s vibration characteristics. This module helps generate condition reports that summarize the equipment's health.

The combination of these scripts provides a seamless workflow for data logging, visualization, and analysis, ensuring that the vibration monitoring system is both functional and efficient.

**Data Analysis**

A graph of data collected

Description automatically generated

Figure 9 Data Collected (before conversion from analog signal)

The data analysis is done on our Report\_Generator.py. We first convert the analog signal into usable data with the conversion formula provided in the ADXL335 datasheet.

* + In Time domain Analysis

With the raw data, we can analyze each run for characteristics. Below lists common ones with formulas:

* + - RMS:



* + - Peak2Peak:

A black and white math equation

Description automatically generated

* + - StD:

A number and symbols on a white background

Description automatically generated

* + In Frequency-Domain-Analysis:

Next step would be to perform Fast-Fourier-Transform of these signals to get their attributes in the frequency domain, with formulas listed below:

* + - FFT: A close up of a quote

      Description automatically generated with medium confidence
    - Magnitude



* + - Frequency:

A math equation with a number and a question mark

Description automatically generated with medium confidence

A screenshot of a computer screen

Description automatically generated

Figure 10 FFT Plot

The next step would be to analyze and compare the data with baseline data, to determine the condition of the machine, which will be future work that will be discussed in the next part.

**Future work/Improvement**

Our project (vibration monitoring system) is just a basic initial design, but there is still plenty of room for improvement to enhance its functionality and even add new features.

**Enhancing Data Analysis Capabilities**  
Currently, the system uses FFT and various time-domain metrics such as RMS and P2P for basic vibration analysis. While these metrics provide some insights, we can use more advanced algorithms in the future, such as machine learning models, to make more accurate failure predictions or provide more detailed condition reports. Additionally, due to the lack of baseline data, this project only focused on data collection and processing, without detailed analysis.

**Improving Communication Protocols**  
At the moment, our system relies on a Bluetooth module for wireless communication. While Bluetooth is simple and convenient for this small project, it comes with limitations such as short communication range, susceptibility to interference, and limited scalability (a master device can connect to only up to 7 slaves). In the future, we can adopt more stable communication protocols like RS485, which offers better stability, longer range, and larger network capacity. Other options, such as Wi-Fi or LoRa, can also be considered for longer distances and better compatibility.

**Optimizing Hardware Design**  
The chassis is fully exposed to meet the size limitation for the project, and to fit the Arduino uno controller in. Ideally, it should be durable with fix points for wires, and reliable against vibration as a vibration monitoring device. In addition, the size Arduino controller is the biggest factor to consider when designing the device. It is multifunctional and flexible for prototy33ping, but for a finished product, a specialized SoCs would be more ideal for this case, such as ESP32.

Figure 11 ESP32

**Automating the System**  
Although the system can collect vibration data and perform analysis, integrating it with automated machinery could make the system more complete. For example, if the data analysis detects or predicts dangerous vibration levels, the system could automatically trigger an emergency stop and notify monitoring personnel to prevent potential damage or accidents.

**Power Efficiency**  
The system currently relies on a 12V battery for power. We can explore more feasible alternatives to optimize energy consumption and extend system runtime. For instance, using rechargeable batteries paired with solar panels could be a sustainable option, though it may be limited by environmental factors.

**HMI (Human-Machine Interface)**  
The current project does not include an HMI panel. Adding an HMI would make it easier for monitoring personnel to read the equipment’s health status and identify emergency situations, allowing them to respond quickly and avoid problems.