A Project Stage-II Report on

# Health Monitoring System of Universal Vibration Test Rig using AI Approach

By

| | |
|---|---|
| *Mr*. Patil Harshwardhan Pradip | Exam. Seat No : B150361068 |
| *Mr*. Patil Kshitij Shivaji. | Exam. Seat No : B150361073 |
| *Mr*.Patil Punit Sanjay | Exam. Seat No : B150361078 |
| *Mr*.Pawar Vishal Rajendra | Exam. Seat No : B150361091 |

**Guide**

Prof. S. P. Shinde

Department of Mechanical Engineering

Sinhgad Technical Education Society's

Smt. Kashibai Navale College of Engineering [2021-22]

**Sinhgad Technical Education Society's**

**Smt. Kashibai Navale College of Engineering**

# CERTIFICATE

This is to certify that

| | |
|---|---|
| *Mr*. Patil Harshwardhan Pradip | Exam. Seat No………………… |
| *Mr*. Patil Kshitij Shivaji. | Exam. Seat No………………… |
| *Mr*. Patil Punit Sanjay | Exam. Seat No………………… |
| *Mr*. Pawar Vishal Rajendra | Exam. Seat No………………… |

have successfully completed the Project Stage – I entitled "**Health Monitoring System of Universal Vibration Test Rig using AI Approach**" under my supervision, in the partial fulfilment of *Bachelor of Engineering-Mechanical Engineering* of Savitribai Phule Pune University.

Date: -

Place: - Pune

***Prof. S. P. Shinde***
***Guide***

***Prof…………***
***External Examiner***

***Prof. T. S. Sargar***
**Head of Department**

***Dr. A.V. Deshpande***
**Principal**

## ACKNOWLEDGEMENT

I take this opportunity to thank all those who have contributed in successful completion of this Project Stage -1 work. I would like to express my sincere thanks to my guide **Prof. S. P. Shinde** who have encouraged me to work on this topic and provided valuable guidance wherever required. I also extend my gratitude to **Prof. T. S. Sargar (H.O.D Mechanical Department)** who has provided facilities to explore the subject with more enthusiasm.

I express my immense pleasure and thankfulness to all the teachers and staff of the **Department of Mechanical Engineering** of **Smt. Kashibai Navale College of Engineering,** for their co-operation and support.

| | |
|---|---|
| *Mr*. Patil Harshwardhan Pradip | Sign………………. |
| *Mr*. Patil Kshitij Shivaji. | Sign………………. |
| *Mr*. Patil Punit Sanjay | Sign………………. |
| *Mr*. Pawar Vishal Rajendra | Sign………………. |

# CONTENT

# List of Figures

## Abstract

Machines have been an inevitable part of our life in today's era. It has become paramount to look after various machines and keep them in safe as well as efficient condition. To check the health of Universal Vibration Test Rig machine, Accelerometer sensor have been used which measures vibration parameter of machine. Any defect in the machine is indicated by unusual behavior in the vibration parameter. Time domain graph are used to measure vibrations in the machine. In health monitoring FFT analysers used, however the cost of FFT analyzers is very high and it may not be affordable to small-scale industries. Also, they rarely have a provision to measure the speed, temperature, and power consumed by the machine. The present research work is an effort to provide a low-cost solution to the existing health monitoring systems along with facility to measure vibration parameter in the machine set-up. A low-cost controller Arduino Uno R3 is used along with Accelerometer ADXL335 sensor and integrated to Arduino IDE & Python Software to store and display the acquired data. By analyzing the time domain graph of vibration parameter and acquiring the vibration data health monitoring system identifies the unusual behaviour of the Universal Vibration Test Rig by a bit of coding in Arduino IDE & Python software. The developed device was compared with existing systems and found to have a good agreement. This device can be used as a monitoring tool in small-scale industries where high-cost FFT analyzers become obsolete due to costing issue.

**Keywords:-** FFT ,Machine health , Health monitoring, Vibration, Accelerometer ADXL335, Arduino Uno R3, Arduino IDE, Python, Time domain.

# 1. Introduction

In this project there's going to be a Health Monitoring System for traditional machines in advanced way to be more aware of their working condition with help of Artificial Intelligence and Machine Learning. Artificial Intelligence uses Machine Learning to mimic human intelligence. The computer learns how to respond to certain actions, so it uses algorithms and historical data to create something called a propensity model. Propensity models will then start making predictions (like scoring leads or something). Here with help of same there will be judging machine's work health on basis of some specific parameters such as velocity, vibration and distance. The machine's readings which were used to be taken manually now will be digitally transferred using Sensors (to detect) and Data Acquisition System (DAQ) to AI and ML programs and then their working health data in real-time would be then presented in form of graphs or charts. (Artificial intelligence is a constellation of many different technologies working together to enable machines to sense, comprehend, act, and learn with humanlike levels of intelligence. While Machine learning is branch of A.I. and a method of data analysis that automates analytical model building).

Inertial microelectromechanical systems (MEMS) sensors assume a huge part in the huge extension of today's personal electronic gadgets. Their little size, low power, simplicity of combination, abnormal state of usefulness, and wonderful execution energize, and empower development in contraptions, for example, cell phones, gaming controllers, motion trackers, and advanced picture outlines. Likewise, inertial MEMS sensors have generously enhanced unwavering quality and lessened expense in car wellbeing frameworks, permitting them to be conveyed in many cars [1]. The object of frequency analysis is to break down a complex signal into its components at various frequencies, and in order to do this, the practical engineer needs to understand the frequency analysis parameters and how to interpret the results of spectrum measurements [2]. The continuous advancement in practical integration and performance has conjointly helped MEMS accelerometers notice their approach into various industrial systems. Some of these applications offer lower-cost alternatives to current product and services, whereas others are segregating inertial transducer action in a new and unique way. New adaptations in vibration sensing area are concluding that fast distribution and affordable price of possession are the reasons to judge the integrated MEMS devices. Vibration monitoring is coming up as associate application that has each variety of users. Conventional instruments that observe machine health for maintenance and safety usually use piezoelectric technology. High-speed automation instrumentation setup monitors vibration to trigger feedback management of lubrication, speed, or belt tension or to switch off instrument for fast attention from the operating staff [3–5]. MEMS accelerometers provide quick, efficient integration, and cost-effective solution to a rising cluster of latest users. Additionally, their advanced practical integration permits devices like the ADIS16229 digital MEMS vibration sensing element with embedded RF transmitter and receiver to supply an entire resolution inclusive of communications and signal processing [6, 7]. This kind of device will wake itself up repeatedly, record time domain vibration information data, perform fast Fourier transform (FFT) on the information recorded, apply user-configurable spectral analysis on the FFT result, provide easy pass/fail results over economical wireless transmission, offer access to that information and results and then return to hibernate or sleep mode [8]. To enhance manual measurements, embedded MEMS-based sensors give a more cost-effective method for instrumentality that needs real-time vibration information.

## 1.1 Problem statement:

Mechanical industry is in the midst of a significant transformation regarding the way it produce products thanks to the digitization of manufacturing. This transition is so compelling that it is being called Industry 4.0 to represent the fourth revolution that has occurred in manufacturing. The fourth industrial revolution has taken on what was started in the third with the adoption of computers and automation and enhance it with smart and autonomous systems fueled by data and machine learning. Mechanical Machines are a bit late with this upcoming revolution, leading to its urgent need to make its way onto this ongoing future.

## 1.2 Objectives:

Is to create a system that give us indication on machine's health. By means of the graph that we get from data acquisition system with help of Artificial Intelligence and Machine Learning. This can help in better information collection with accurate work. The change in regular pattern of graph will indicate some ongoing failure in machine and can y

## 1.3 Scope of the Project:

This system will identify and quantify any damage or deterioration state that might occur the service life of the machine. In future this system will reduce the downtime of the machine and it will be easy to detect the defects in machine and correct them. By using this system it will avoid the financial as well as life loss in the industry.

## 1.4 Methodology:

In this project there's going to be a Health Monitoring System for traditional machines in advanced way to be more aware of their working condition with help of Artificial Intelligence and Machine Learning. Artificial Intelligence uses Machine Learning to mimic human intelligence. The computer learns how to respond to certain actions, so it uses algorithms and historical data to create something called a propensity model. Propensity models will then start making predictions (like scoring leads or something). Here with help of same there will be judging machine's work health on basis of some specific parameters such as velocity, vibration and distance. The machine's readings which were used to be taken manually now will be digitally transferred using Sensors (to detect) and Data Acquisition System (DAQ) to AI and ML programs and then their working health data in real-time would be then presented in form of graphs or charts. (Artificial intelligence is a constellation of many different technologies working together to enable machines to sense, comprehend, act, and learn with humanlike levels of intelligence. While Machine learning is branch of A.I. and a method of data analysis that automates analytical model building).

**Process Flow:**

1. Connecting Arduino Sensors to Arduino Uno R3

2. Connecting Arduino Uno Board to Arduino IDE software for code loading.

3. Serial Monitoring in Arduino IDE Software.

4. Python3 software used for data acquisition.

5. Importing Python Library for live graph plotting.

6. Detecting the machine operation or fault through graph reading.

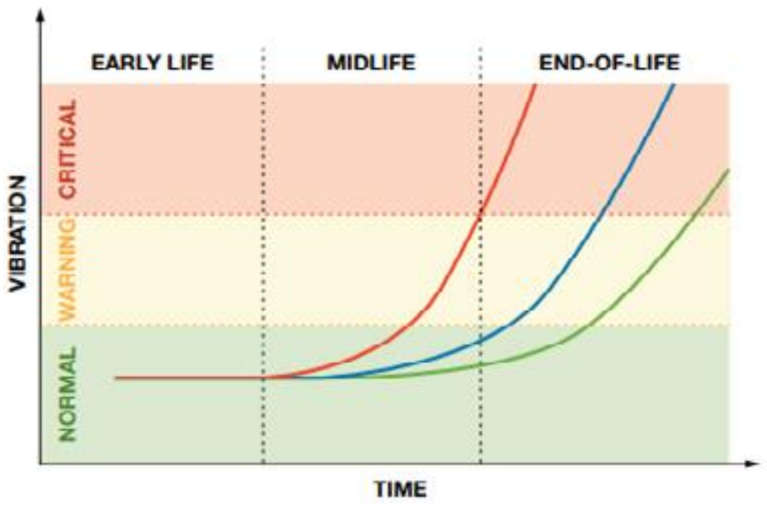
7. Feeding data in machine learning algorithm and analyse results.

## 2. Literature Review:-

In the industries health of the machine is most prior thing which cannot be ignored, machines have been an inevitable part of our life in today's era. It has become paramount to look after various machines and keep them in safe as well as efficient condition. Like the health of the human body, health of the machine must be checked frequently to avoid financial as well as the living loss in the industry. It is not possible or advantageous to check machine frequently as if machine stops industry may suffer from loss because industry has some daily targets to achieve in constrained time period, so it became must to create a system that can give us indications that some part of machine is suffering from damage and if not repaired can create major failure. If Human body gets sick conditions, it gives some indications, likewise if created such system to get indications for machines, we can avoid its future collapse.
A review of the IoT, ML, cloud computing and conditional monitoring, which have an effective place in industry 4.0, has been reviewed, especially on predictive maintenance. There are many predictive maintenance techniques, these techniques are particle analysis, thermography, oil analysis, vibration monitoring, acoustic emission, corrosion monitoring and performance monitoring. To check the health of such machines, by measuring vibration, temperature, noise level, and power consumption. Any defect in the machines is indicated by unusual behavior in the above parameters.

1. **An Introduction to MEMS Vibration Monitoring – by Mark Looney. –**
   Inertial MEMS sensors play a significant role in the massive expansion of today's personal electronic devices. Their small size, low power, ease of integration, high level of functionality, and superb performance encourage and enable innovation in gadgets such as smartphones, gaming controllers, activity trackers, and digital picture frames. In addition, inertial MEMS sensors have substantially improved reliability and reduced cost in automotive safety systems, allowing them to be deployed in most automobiles.
   When using vibration to observe machine health, the objective is to correlate observable vibration with typical wear-out mechanisms, such as bearings, gears,

chains, belts, brushes, shafts, coils, and valves. In a typical machine, at least one these mechanisms requires regular maintenance. Figure 1 shows three examples of the vibration vs. time relationship for a normal wear-out mechanism. Although it takes time and experience to develop this type of relationship, a well-correlated vibration signature can be a cost-saving alternative to regular maintenance that follows short cycle times. Using actual observations, such as vibration, provides an opportunity to take quick action when warning conditions are detected (red curve), while avoiding premature maintenance on machines that have more life remaining (blue and green curves).



(Fig 1: Vibration vs. Time graph)

2. **Towards devising a vibration based machinery health monitoring system – by Md. Harunur Rashid Bhuiyan a, Iftekhar Morshed Arafat a, Masfiqur Rahaman b, Tarik Reza Toha a, Shaikh Md. Mominul Alam -**
   Many parameters are considered while monitoring a machine’s health condition such as sound, mechanical wear, robustness, etc. However, vibration signals carry a great deal of information about a mechanical system’s health. Vibration analysis is a very technique among condition monitoring systems. Machine health monitoring helps in scheduling maintenance, taking measures in stopping the consequence of machine failure. Any change in the machine’s health condition will result in changing the

output of the vibration signal. Excess vibration for a long period can cause deterioration in rotary machinery which can result in reduced production efficiency . It can also decrease the performance and the expected lifetime of a machine . Failure of a machine due to vibration can result in machine shutdown. Finding a way of continuously monitoring the vibration of a running machine is important. Existing Wi-Fi and Ethernet-based solutions can monitor vibration data of machinery. However, they are expensive and improper to use in a factory due to several reasons. Although the Wi-fi and Ethernet itself are not that costly, installing them inside a factory would need additional infrastructure. This can make the vibration monitoring system expensive. Other solutions are not wireless. Applying them inside an industry would require maintenance personnel to continuously stand in front of the machine. As a solution to those problems, in this paper, we present a lowcost wireless vibration monitoring system. Here, we sense the vibration data of machinery through an SW-420 vibration sensor on Arduino UNO R3. The sender Arduino sends the data through an HC-05 Bluetooth module to the receiver Arduino. Based on our work, we make the following contributions: We use a low-cost SW-420 vibration sensor for collecting vibration data. We present a system where data will be sent wirelessly through Bluetooth module which is highly applicable inside the factory compared to other existing systems. We deploy the device in textile factories and collect real-life data of machine vibration. By deploying our device in textile factories to collect real data, we have obtained important findings on the relation between machine age and machine vibration. Analyzing the vibration data of different machinery, we can better un- derstand the maintenance requirements of those machineries.

3. **Condition monitoring based on IoT for predictive maintenance of CNC machines – by Yahya Mohmmad Al-Naggar, Norlida Jamil, Mohd firdaus Hassan, Ahmed Razlan Yousuof.-**

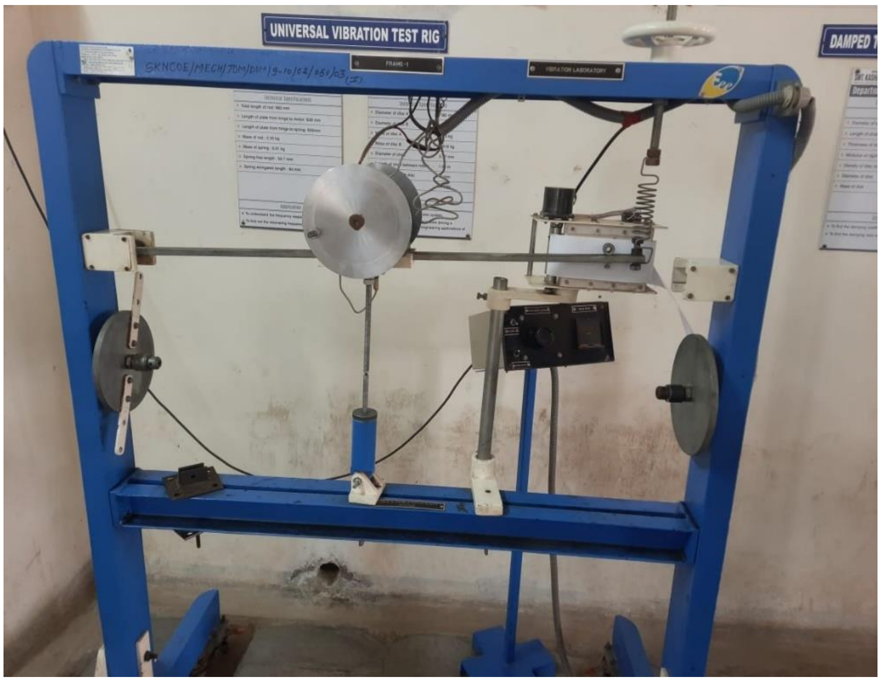
   Early detection of the machine condition can help in predictive maintenance. Vibration analysis isgenerally used in the predictive maintenance procedure and as a support for decisions in machinery maintenance. This type of analysis can be performed to avoid the occurrence of failure. The increased level of vibration in machines typically indicates the imminent occurrence of breakdown or failure. The nature and severity of machine defect can be determined and machine failure can be predicted via monitoring and analysis of vibration.

Monitoring the condition of the machine via vibration analysis provides many benefits, including increased revenue, improved efficiency of production, reduced downtime, reduced maintenance costs, increased availability of machinery and reduced stocks of spare parts.

4. **The Experimental Application of Popular Machine Learning Algorithms on Predictive Maintenance and the Design of IIoT Based Condition Monitoring System – by Mustafa Cakir, Mehmet Ali Guvenc, Selcuk Mistikoglu**. –
Condition monitoring techniques fall into two categories: offline and online monitoring (Malla & Panigrahi, 2019). Offline condition monitoring is performed within a certain time interval of data from sensors on the machine and is also referred to as periodic condition monitoring. The analysis of the sensor data after reading is carried out in a laboratory environment, away from the machine. Another method is online condition monitoring, which is called continuous condition monitoring. Data is continuously collected by the sensors in the running machine and compared to an acceptable value, or in other words, a threshold value. Recently online CMS is very popular due to the development of Internet of Things (IoT) solutions. The IoT concept provides some convenience in industrial environments. IoT solutions in industrial environments, which is called IioT, can lead to the development of innovative and highly reliable systems aiming to increase operational efficiency in new generation smart factories today (Khuntia, Rueda, & van der Meijden, 2019). Unlike periodic maintenance, IioT systems that perform continuous control operations can provide great advantages to the company with a warning that a serious failure will occur.

## 3. Design of proposed work :

### 3.1Universal Vibration Test Rig



(Fig 2: Universal Vibration Test Rig)

- Type: Vibration
- Usage/Application: College Laboratory
- Test Component: Experiments to illustrate & verify the principles & relationships involved in the study of vibration
- Material: Frame - Aluminum Extrusion / MS part - plating / Powder Coating

We are engaged in offering Universal Vibration Test Rig ( 10 Experiments ) apt for conducting various experiments such as single pendulum, compound pendulum, bifilar suspension for determination of M.I., spring mass system with damped vibrations and others. Advanced machines and equipment are used to manufacture this apparatus so that it results in better functionality and provides precise outputs.

**Universal Vibration Test Rig machine is used to perform various experiments as follows:**

- Simple Pendulum
- Compound Pendulum
- Bifilar Suspension for determination of M.I
- Spring mass System with damped vibrations
- Spring mass System with undamped vibrations
- Equivalent Spring Mass System
- Torsional Vibrations Single Rotor
- Torsional Vibrations Two Rotor
- Forced Vibrations Lateral
- Single rotor Viscous damping
- Dunkerly's System

**Technical Specifications:**

- Total length of rod: 960 mm
- Length of plate from hinge to motor: 639 mm
- Length of plate from hinge to spring: 935 mm
- Mass of rod: 2.35 kg
- Mass of spring: 6.01 kg
- Spring free length: 50.7 mm
- Spring elongated length: 64 mm

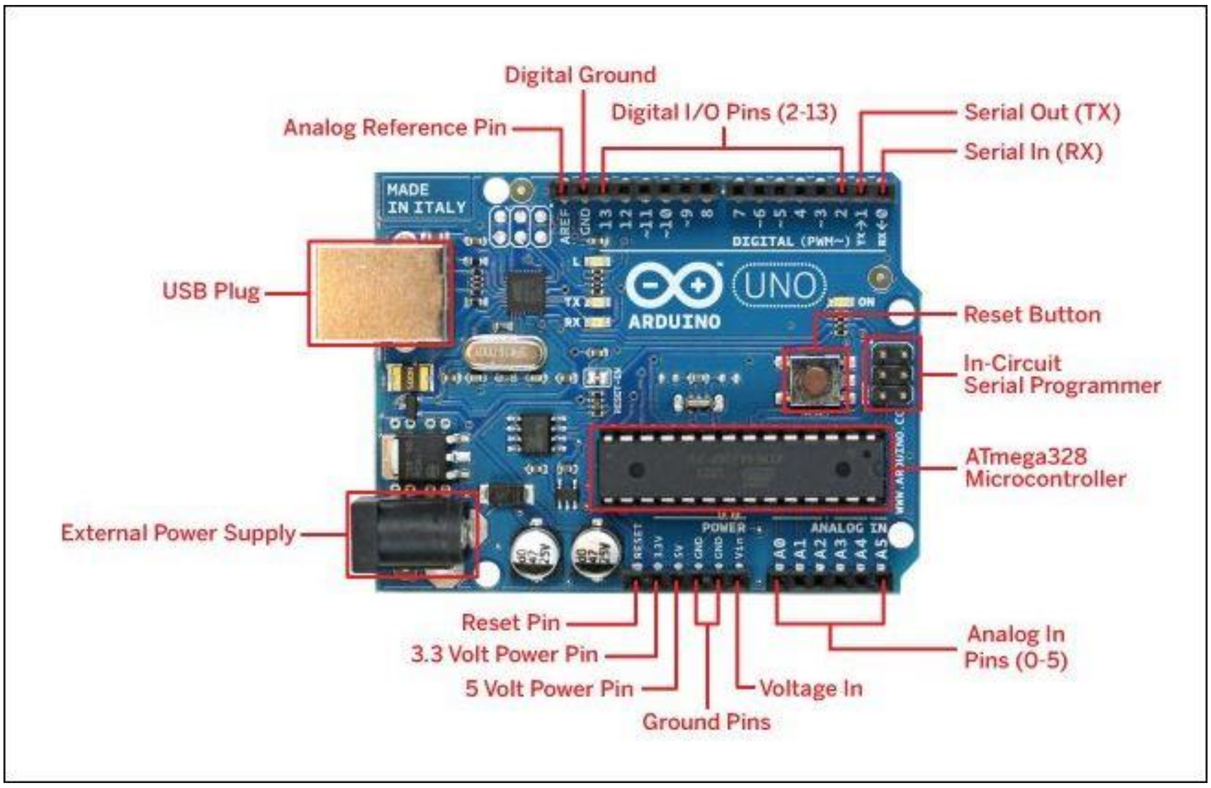
**Two Rotor System Specifications:**

- Diameter of disc A: 190 mm
- Diameter of disc B: 225 mm
- Mass of disc A: 2.75 kg
- Mass of disc  B: 3.41 kg
- Diameter of shaft: 3.17 mm

- Length of shaft between rotors: 1.08 m
- Thickness of both disc: 11 mm
- Modulus of rigidity: 84 Gpa

## 3.2 Arduino Uno R3

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output pins that may be interfaced to various expansion boards and other circuits.



(Fig 3: Arduino Uno R3)

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc.[2][3] The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits.[1] The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable.[4] It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts.

The word "uno" means "one" in Italian and was chosen to mark the initial release of Arduino Software.[1] The Uno board is the first in a series of USB-based Arduino boards;[3] it and version 1.0 of the Arduino IDE were the reference versions of Arduino, which have now evolved to newer releases.[4] The ATmega328 on the board comes preprogrammed with a bootloader that allows uploading new code to it without the use of an external hardware programmer.

ATmega328P
8-bit AVR Microcontroller with 32K Bytes In-System
Programmable Flash

**Features :**

- High performance, low power AVR® 8-bit microcontroller
- Advanced RISC architecture
- 131 powerful instructions – most single clock cycle execution
- 32 □ 8 general purpose working registers
- Fully static operation
- Up to 16MIPS throughput at 16MHz
- On-chip 2-cycle multiplier
- High endurance non-volatile memory segments
- 32K bytes of in-system self-programmable flash program memory
- 1Kbytes EEPROM
- 2Kbytes internal SRAM
- Write/erase cycles: 10,000 flash/100,000 EEPROM
- Optional boot code section with independent lock bits
- In-system programming by on-chip boot program

- True read-while-write operation
- Programming lock for software security
- Peripheral features
- Two 8-bit Timer/Counters with separate prescaler and compare mode
- One 16-bit Timer/Counter with separate prescaler, compare mode, and capture mode
- Real time counter with separate oscillator
- Six PWM channels
- 8-channel 10-bit ADC in TQFP and QFN/MLF package
- Temperature measurement
- Programmable serial USART
- Master/slave SPI serial interface
- Byte-oriented 2-wire serial interface (Phillips I2
  C compatible)
- Programmable watchdog timer with separate on-chip oscillator
- On-chip analog comparator
- Interrupt and wake-up on pin change
- Special microcontroller features
- Power-on reset and programmable brown-out detection
- Internal calibrated oscillator
- External and internal interrupt sources
- Six sleep modes: Idle, ADC noise reduction, power-save, power-down, standby,
- and extended standby
- I/O and packages
- 23 programmable I/O lines
- 32-lead TQFP, and 32-pad QFN/MLF
- Operating voltage:
- 2.7V to 5.5V for ATmega328P
- Temperature range:
- Automotive temperature range: –40°C to +125°C

- Speed grade:
- 0 to 8MHz at 2.7 to 5.5V (automotive temperature range: –40°C to +125°C)
- 0 to 16MHz at 4.5 to 5.5V (automotive temperature range: –40°C to +125°C)
- Low power consumption
- Active mode: 1.5mA at 3V - 4MHz
- Power-down mode: 1µA at 3V

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode.

**Revision 3 of the board has the following new features:**

1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
Stronger RESET circuit.
Atmega 16U2 replace the 8U2.
"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is

the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

**Summary:**
Microcontroller ATmega328 Operating Voltage 5V Input Voltage (recommended) 7-12V Input Voltage (limits) 6-20V Digital I/O Pins 14 (of which 6 provide PWM output) Analog Input Pins 6 DC Current per I/O Pin 40 mA DC Current for 3.3V Pin 50 mA Flash Memory 32 KB (ATmega328) of which 0.5 KB used by bootloader SRAM 2 KB (ATmega328) EEPROM 1 KB (ATmega328) Clock Speed 16 MHz

Schematic & Reference Design
ArduinoUnoSmd.jpg

EAGLE files: arduino-uno-Rev3-reference-design.zip

Schematic: arduino-uno-Rev3-schematic.pdf

Note: The Arduino reference design can use an Atmega8, 168, or 328, Current models use an ATmega328, but an Atmega8 is shown in the schematic for reference. The pin configuration is identical on all three processors.

Power
The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

**The power pins are as follows:**

VIN. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
5V. The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
GND. Ground pins.
Memory
The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output
Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library. There are a couple of other pins on the board:

AREF. Reference voltage for the analog inputs. Used with analogReference().
Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.
See also the mapping between Arduino pins and ATmega328 ports. The mapping for the Atmega8, 168, and 328 is identical.
Communication
ArduinoUno r2 front

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

Programming

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno from the Tools > Board menu (according to the microcontroller on your board). For details, see the reference and tutorials.

The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.
On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.
You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment.

This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics

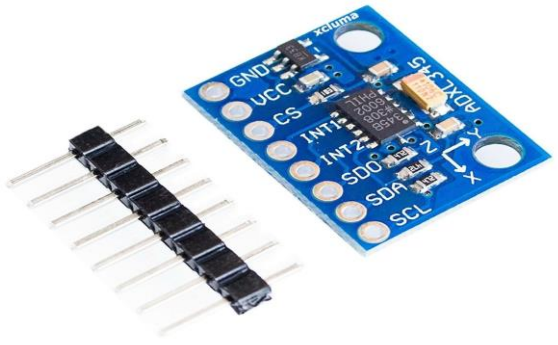
The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Four screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

## 3.3 Accelerometer

An accelerometer is a tool that measures proper acceleration. Proper acceleration is the acceleration of a body in its own instantaneous rest frame; this is different from coordinate acceleration, which is acceleration in a fixed coordinate system. The ADXL335 is a small, thin, low power, complete 3-axis accelerometer with signal conditioned voltage outputs. The product measures acceleration with a minimum full-scale range of ±3 g

**ADXL335 Pinout Configuration :**

| Pin Name | Description |
|---|---|
| VCC | The Vcc pin powers the module, typically with +5V |
| GND | Power Supply Ground |
| X | X-axis Analog Output Pin |
| Y | Y-axis Analog Output Pin |
| Z | Z- axis Analog Output Pin |
| ST | Self Test pin. The pin controls the Self-Test feature |



(Fig 4: Accelerometer adxl335)

**Accelerometer Module Features & Specifications:**

- Operating Voltage: 3V to 6V DC
- Operating Current: 350µA
- Sensing Range: ±3g
- 3-axis sensing
- High Sensitivity for small movements

- Needs no external components
- Easy to use with Microcontrollers or even with normal Digital/Analog IC
- Small, cheap and easily available

Alternate Sensor Modules: IR Sensor Module, LDR Sensor Module, TP4056ALi-ion Battery Charging/Discharging Module, DS3231 RTC Module, TMC2209 Stepper Motor Driver Module, DRV8825 Stepper Motor Driver Module, A4988 Stepper Motor Driver Module, NEO-6MV2 GPS Module, Joystick Module, EM18 - RFID Reader Module, GLCD 128x64, ADXL335 Accelerometer Module, HMC5883L Magnetometer Module, Soil Moisture Sensor

Related Components: ADXL335, Resistors, Voltage Regulator IC

Brief about Accelerometer Module
This ADXL335 Accelerometer module consists of an ADXL335 Accelerometer IC, Voltage Regulator IC, resistors, and capacitors in an integrated circuit. Different manufacturers use a different voltage regulator IC. Most of the modules use XC6206P332MR (662K) IC.
ADXL335 IC from Analog Devices is the brain of this module. The ADXL335 is a small, thin, low power, complete 3-axis accelerometer with signal conditioned voltage outputs. The product measures acceleration with a minimum full-scale range of ±3 g.

Apart from ADXL335 IC this module also consists of a 3.3V voltage regulator IC.

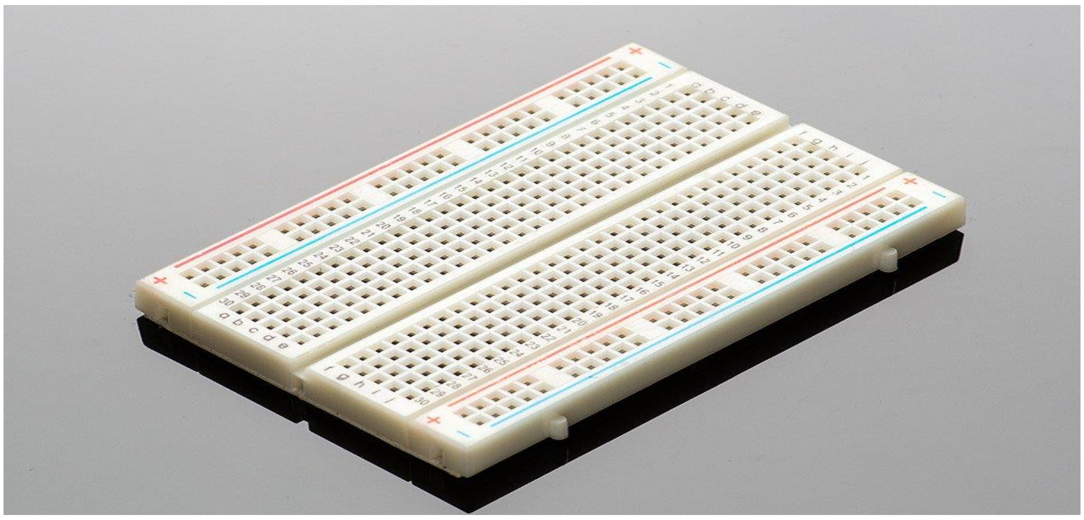
How to Use the ADXL335 Accelerometer Module
ADXL335 Accelerometer module consists of six pins i.e. VCC, GND, X, Y, Z, and ST. Using the Accelerometer module with a microcontroller is very easy. Connect VCC and GND pins to 5V and GND pins of Microcontroller. Also connect X, Y, and Z pins to the Analog pins of Arduino. The basic structure of the accelerometer consists of fixed plates and moving plates. When the acceleration is applied on an axis capacitance between fixed plates and moving plates is changed. This results in a sensor output voltage amplitude which is proportional to the acceleration

**Applications of ADXL335 Accelerometer :**

- Cost-sensitive, low power, motion- and tilt-sensing applications
- Mobile devices
- Gaming systems
- Disk drive protection
- Image stabilization
- Sports and health devices

## 3.4 Breadboard

A breadboard is a rectangular plastic board with a bunch of tiny holes in it. These holes let you easily insert electronic components to prototype (meaning to build and test an early version of) an electronic circuit, like this one with a battery, switch, resistor, and an LED (lightemitting diode).



(Fig 5: Breadboard)

## 3.5 Jumper Wires

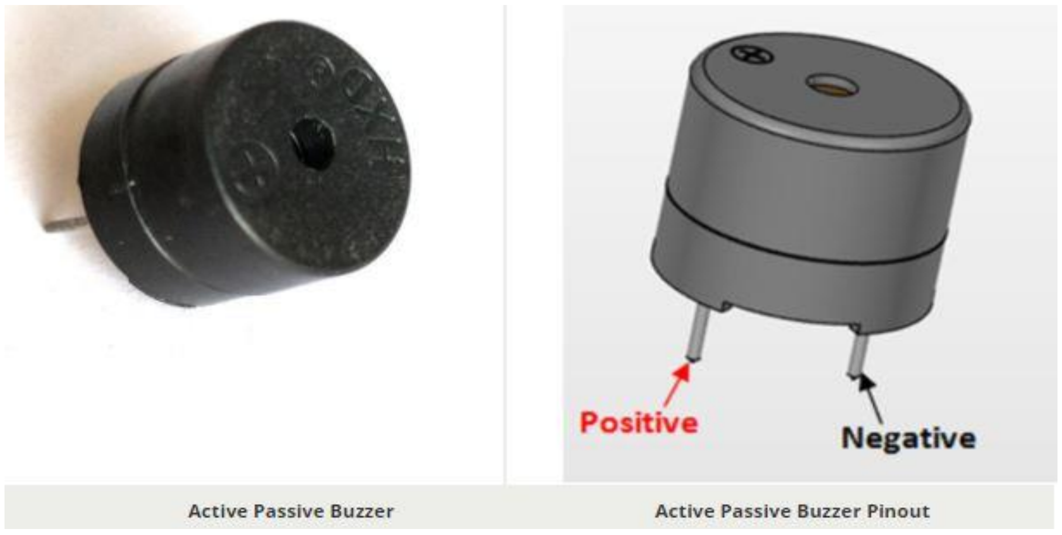
Jumper wires are used to connect two points in a circuit. All Electronics stocks jumper wire in a variety of lengths and assortments. Frequently used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed.

(Fig 6: Jumper Wires)

## 3.6 Piezo buzzer:

A "piezo buzzer" is basically a tiny speaker that you can connect directly to an Arduino. "Piezoelectricity" is an effect where certain crystals will change shape when you apply electricity to them. By applying an electric signal at the right frequency, the crystal can make sound.

Piezo buzzers are constructed by placing electrical contacts on the two faces of a disk of piezoelectric material and then supporting the disk at the edges in an enclosure. When a voltage is applied across the two electrodes, the piezoelectric material mechanically deforms due to the applied voltage.

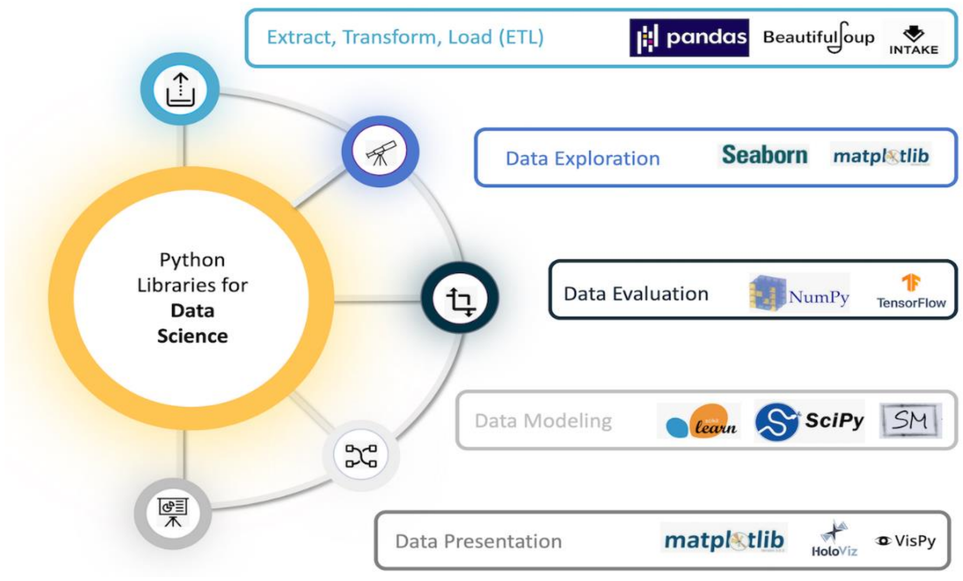


(Fig 7. Buzzer)

## 4. Software:

### 1. Python:

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.



(Fig 8: Python of Data Science)

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

**Data analysis and machine learning :**

Python has become a staple in data science, allowing data analysts and other professionals to use the language to conduct complex statistical calculations, create data visualizations, build machine learning algorithms, manipulate and analyze data, and complete other data-related tasks.

Python can build a wide range of different data visualizations, like line and bar graphs, pie charts, histograms, and 3D plots. Python also has a number of libraries that enable coders to write programs for data analysis and machine learning more quickly and efficiently, like TensorFlow and Keras.

**Modules used in Python:**

1. **Pyserial Module:**

This module encapsulates the access for the serial port. It provides backends for Python running on Windows, OSX, Linux, BSD (possibly any POSIX compliant system) and IronPython. The module named “serial” automatically selects the appropriate backend.

**Features:**

- Same class based interface on all supported platforms.
- Access to the port settings through Python properties.
- Support for different byte sizes, stop bits, parity and flow control with RTS/CTS and/or Xon/Xoff.
- Working with or without receive timeout.

- File like API with “read” and “write” (“readline” etc. also supported).
- The files in this package are 100% pure Python.
- The port is set up for binary transmission. No NULL byte stripping, CR-LF translation etc. (which are many times enabled for POSIX.) This makes this module universally useful.
- Compatible with io library
- RFC 2217 client (experimental), server provided in the examples.

### 2. Matplotlib Module:

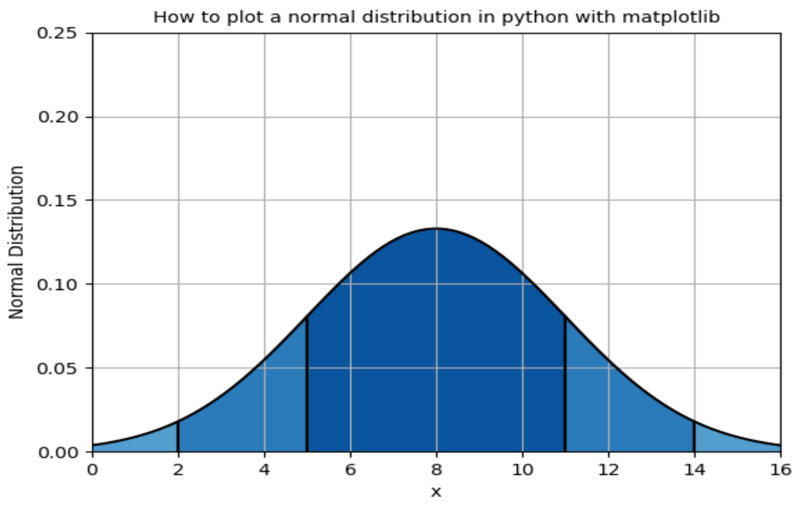
Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged.[3] SciPy makes use of Matplotlib.

**Toolkits**

Several toolkits are available which extend Matplotlib functionality. Some are separate downloads, others ship with the Matplotlib source code but have external dependencies.

- Basemap: map plotting with various map projections, coastlines, and political boundaries
- Cartopy: a mapping library featuring object-oriented map projection definitions, and arbitrary point, line, polygon and image transformation capabilities.(Matplotlib v1.2 and above)
- Excel tools: utilities for exchanging data with Microsoft Excel
- GTK tools: interface to the GTK library
- Qt interface
- Mplot3d: 3-D plots
- Natgrid: interface to the natgrid library for gridding irregularly spaced data.
- tikzplotlib: export to Pgfplots for smooth integration into LaTeX documents (formerly known as matplotlib2tikz)
- Seaborn: provides an API on top of Matplotlib that offers sane choices for plot style

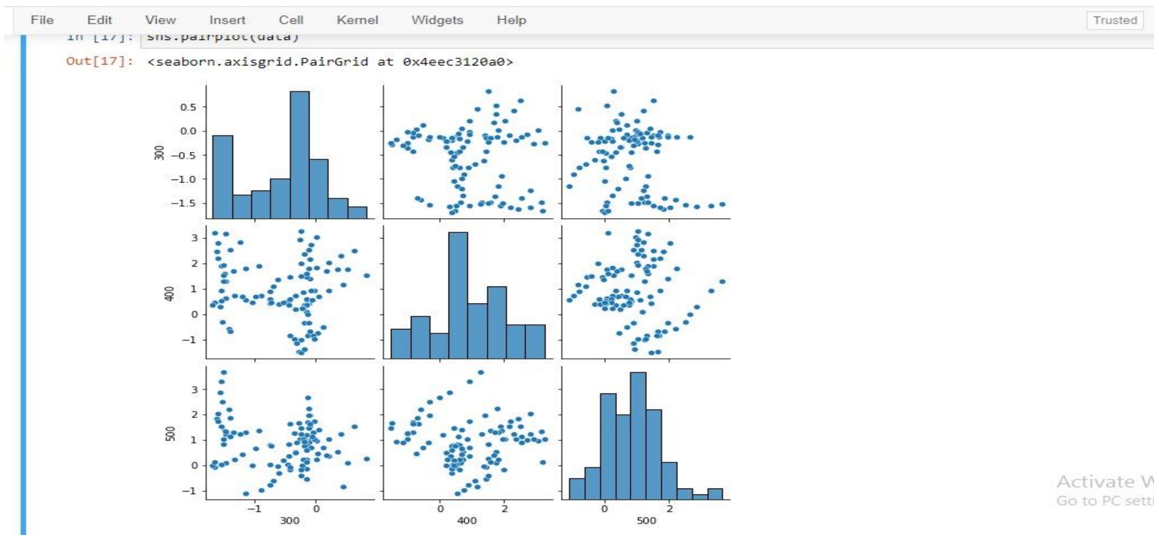
and color defaults, defines simple high-level functions for common statistical plot types, and integrates with the functionality provided by Pandas



(Fig 9: Matplotlib module)

### 3. Seaborn Module:

Seaborn is an open-source Python library built on top of matplotlib. It is used for data visualization and exploratory data analysis. Seaborn works easily with dataframes and the Pandas library. The graphs created can also be customized easily.

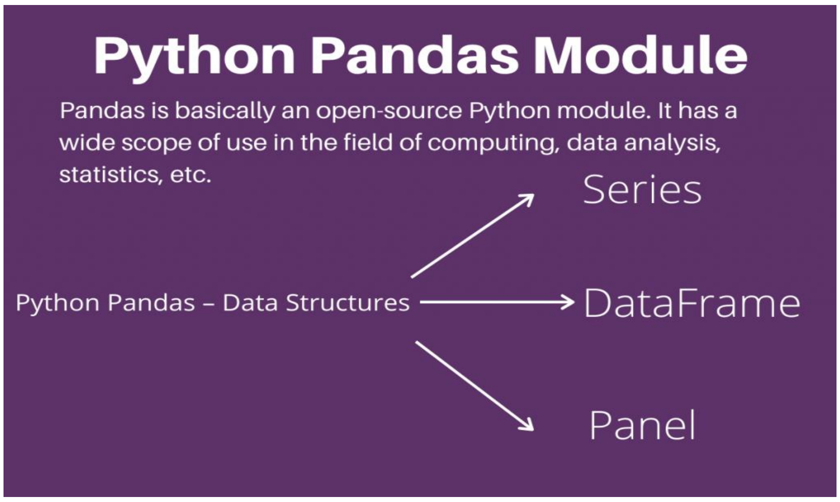


```
In [17]: sns.pairplot(data)
Out[17]: <seaborn.axisgrid.PairGrid at 0x4eec3120a0>
```

(Fig 10: Seaborn module)

### 4. Pandas Module:

Pandas is an open-source library that is made mainly for working with relational or labeled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. This library is built on top of the NumPy library. Pandas is fast and it has high performance & productivity for users.


Python Pandas Module
Pandas is basically an open-source Python module. It has a wide scope of use in the field of computing, data analysis, statistics, etc.
Python Pandas – Data Structures
Series
DataFrame
Panel

**(**Fig 11: Pandas module)

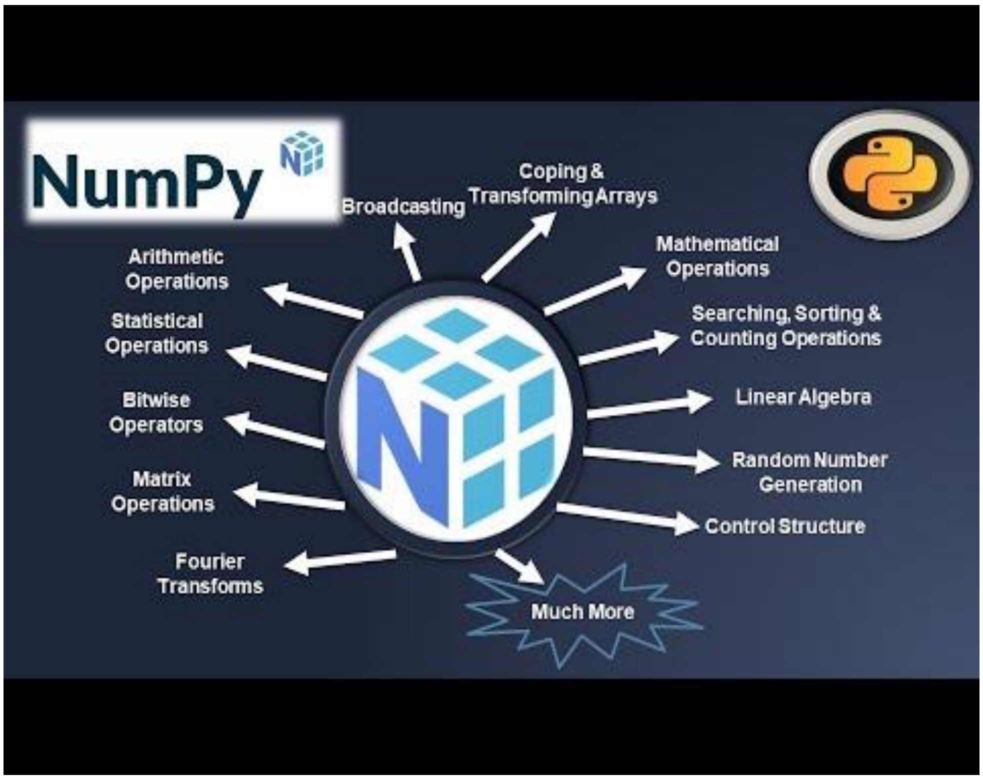
### 5. Numpy Module:

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It is open-source software. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities
- Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using

Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

- It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers.
- In NumPy dimensions are called axes. The number of axes is rank.
- NumPy's array class is called ndarray. It is also known by the alias array



(Fig 12: Numpy module)

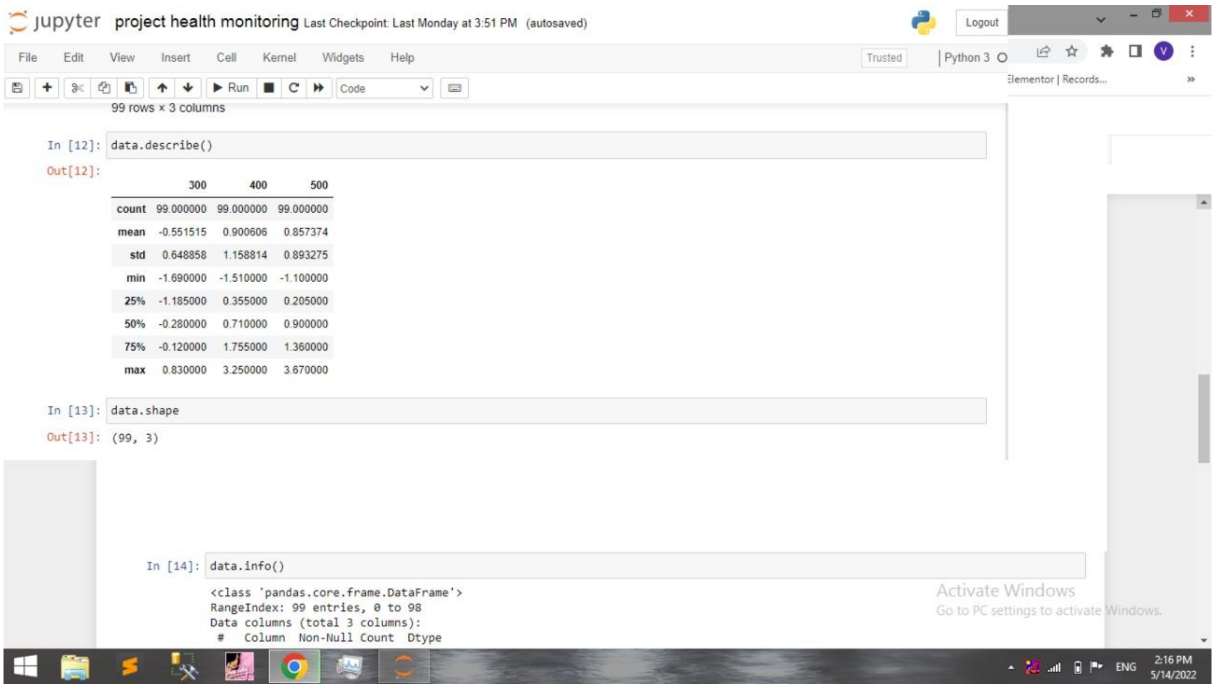
## 6. Jupyter Notebook:

Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating notebook documents.

A Jupyter Notebook document is a browser-based REPL containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media. Underneath the interface, a notebook is a JSON document, following a versioned schema, usually ending with the ".ipynb" extension.

Jupyter notebooks are built upon a number of popular open-source libraries:

- Ipython
- ZeroMQ
- Tornado
- jQuery
- Bootstrap (front-end framework)
- MathJax



```
99 rows × 3 columns

In [12]: data.describe()
Out[12]:
```

| | 300 | 400 | 500 |
|---|---|---|---|
| count | 99.000000 | 99.000000 | 99.000000 |
| mean | -0.551515 | 0.900606 | 0.857374 |
| std | 0.648858 | 1.158814 | 0.893275 |
| min | -1.690000 | -1.510000 | -1.100000 |
| 25% | -1.185000 | 0.355000 | 0.205000 |
| 50% | -0.280000 | 0.710000 | 0.900000 |
| 75% | -0.120000 | 1.755000 | 1.360000 |
| max | 0.830000 | 3.250000 | 3.670000 |

```
In [13]: data.shape
Out[13]: (99, 3)

In [14]: data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99 entries, 0 to 98
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
```

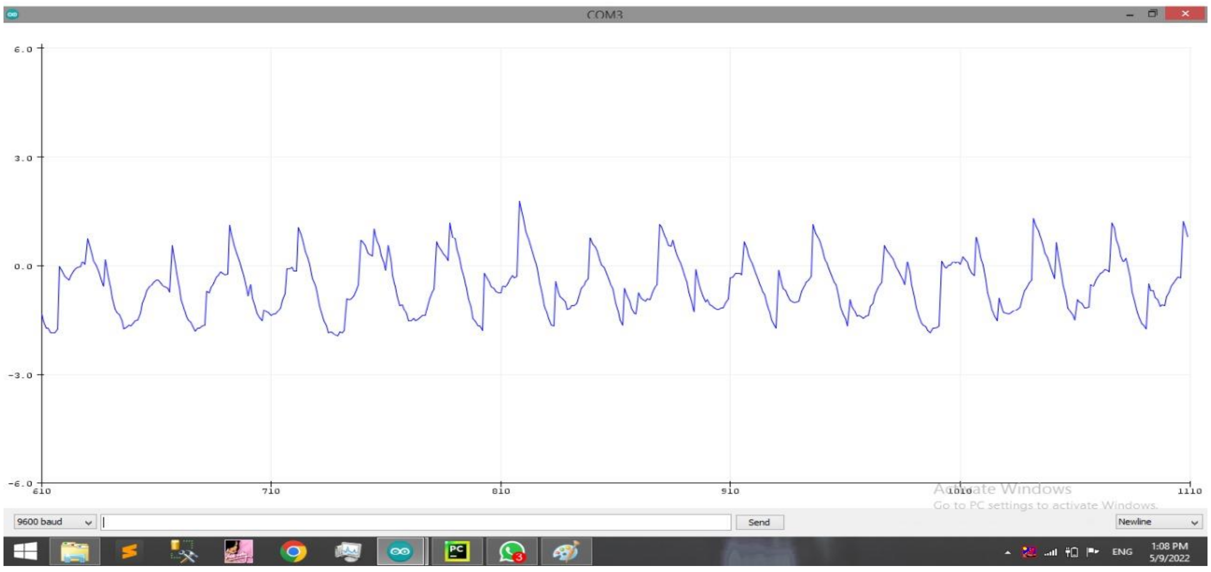
(Fig 13: Jupyter notebook)

## 5. Data Acquisition:

### Aurdino IDE:

The Arduino integrated development environment (IDE) is a cross-platform application (for Microsoft Windows, macOS, and Linux) that is written in the Java programming language. It originated from the IDE for the languages Processing and Wiring. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple one-click mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus. The source code for the IDE is released under the GNU General Public License, version 2.

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.



(Fig 14: Time Domain graph in Arduino IDE)

**Code to collect data from sensors in Arduino:**

```
 int xpin = A3;
 int ypin = A2;
 int zpin = A1;
 int xvalue;
 int yvalue;
 int zvalue;

 void setup()
 {
   Serial.begin(9600);          // initialize the serial communications:
 }
 void loop()
 {
  //xvalue = analogRead(xpin);                                    //reads values from x-pin & measures
acceleration in X direction
  //int x = map(xvalue, 267, 400, -100, 100);              //maps the extreme ends analog values
from -100 to 100 for our understanding
//; you need to replace the 267 & 400 value with your values from calibration
 // float xg = (float)x/(-100.00);                          //converts the mapped value into acceleration
in terms of "g"
  // Serial.print(xg);                                   //prints value of acceleration in X direction
  //Serial.print("g   ");                                //prints "g
 //yvalue = analogRead(ypin);
 //int y = map(yvalue, 272, 406, -100, 100);
 // float yg = (float)y/(-100.00);
  //Serial.print("\n");
  //Serial.print(yg);
  //Serial.print("g   ");
```

```
zvalue = analogRead(zpin);
int z = map(zvalue, 277, 410, -100, 100);
float zg = (float)z/(100.00);
Serial.print("\n");
Serial.print(zg);
//Serial.println("g   ");
delay(100);
```

Data acquisition is the process of sampling signals that measure real world physical conditions and converting the resulting samples into digital numeric values that can be manipulated by a computer. Data acquisition systems, abbreviated by the initialisms DAS, DAQ, or DAU, typically convert analog waveforms into digital values for processing.

The components of data acquisition systems include:

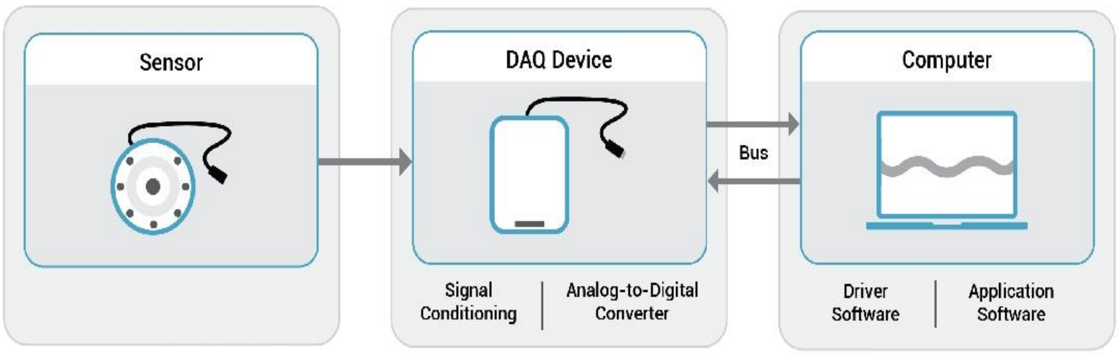
- Sensors, to convert physical parameters to electrical signals.
- Signal conditioning circuitry, to convert sensor signals into a form that can be converted to digital values.
- Analog-to-digital converters, to convert conditioned sensor signals to digital values.

Data Acquisition is generally accepted to be distinct from earlier forms of recording to tape recorders or paper charts. Unlike those methods, the signals are converted from the analog domain to the digital domain and then recorded to a digital medium such as ROM, flash media, or hard disk drives.

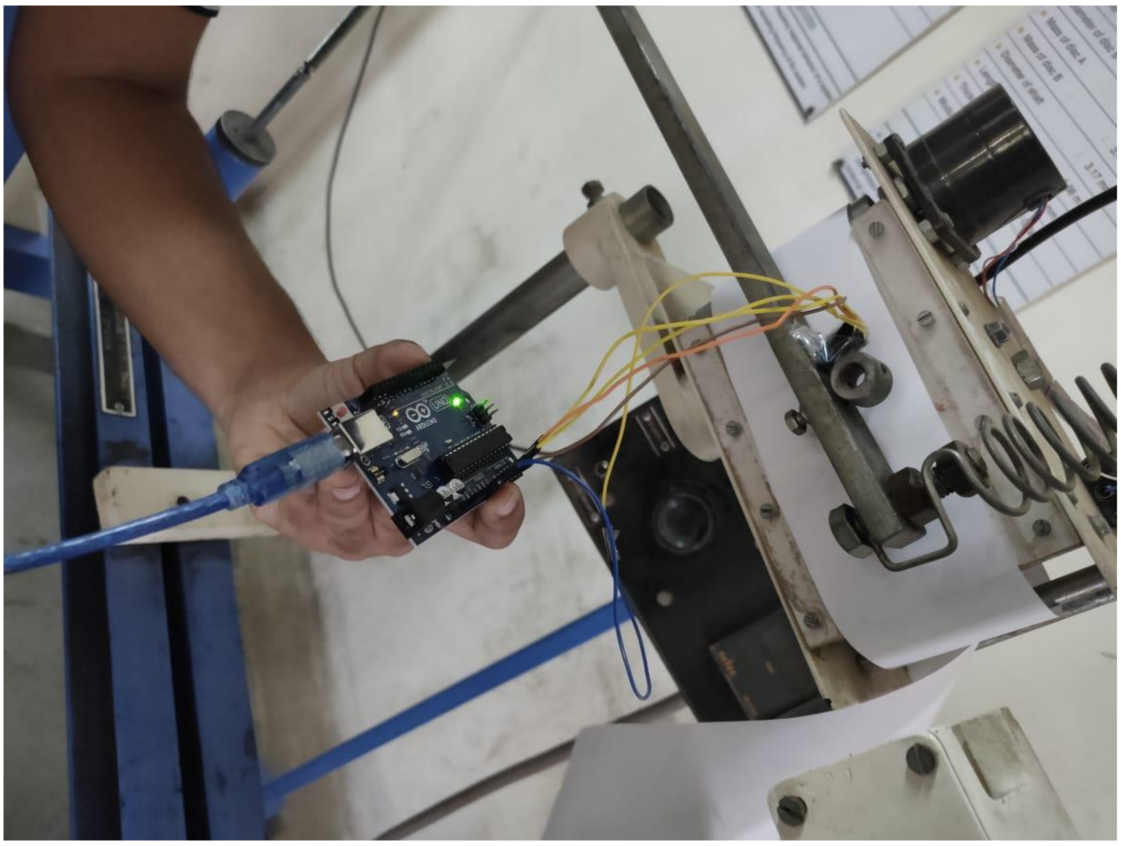
Modern digital data acquisition systems consist of four essential components that form the entire measurement chain of physics phenomena:

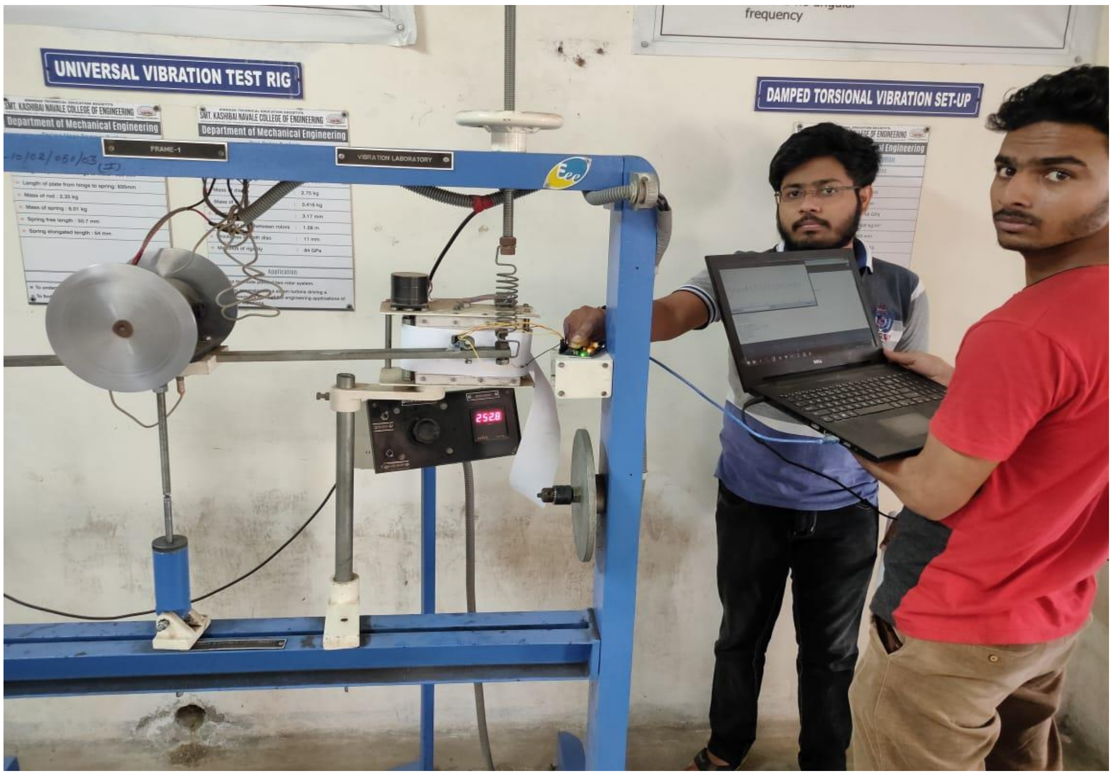
- Sensors
- Signal Conditioning

- Analog-to-Digital Converter
- Computer with Python software for signal logging and analysis

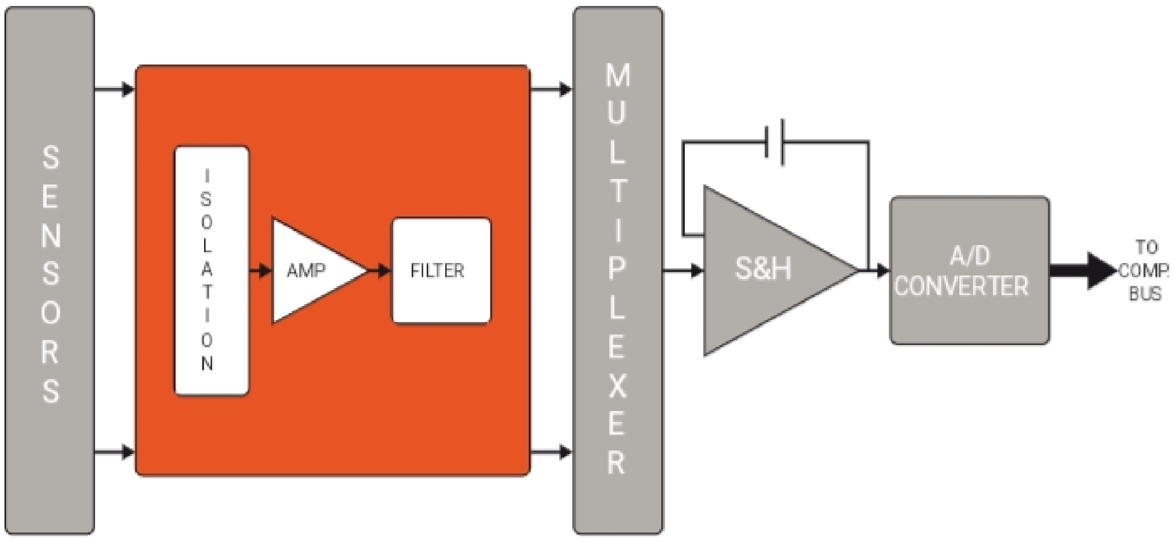


(Fig 15: DAQ Process)




UNIVERSAL VIBRATION TEST RIG
DAMPED TORSIONAL VIBRATION SET-UP
Department of Mechanical Engineering
VIBRATION LABORATORY
FRAME-1
2528

(Fig 16: Experimental Pictures)



(Fig 17: A/D Converter)

Data acquisition systems are principally in the business of measuring physical phenomena such as:

- Temperature
- Voltage
- Current
- Strain and Pressure
- Shock and Vibration
- Distance and Displacement
- RPM, Angle, and Discrete Events
- Weight

The primary purpose of a data acquisition system is to acquire and store the data. But they are also intended to provide real-time and post-recording visualization and analysis of the data. Furthermore, most data acquisition systems have some analytical and report generation capability built-in.

A recent innovation is the combination of data acquisition and control, where a high-quality DAQ system is connected tightly and synchronized with a real-time control system. You can read more about this topic in the related article: "Merging Data Acquisition with a Real-Time Control System".

Engineers in different applications have various requirements, of course, but these key capabilities are present in varying proportion:

- Data recording
- Data storing
- Real-time data visualization
- Post-recording data review

- Data analysis using various mathematical and statistical calculations
- Report generation

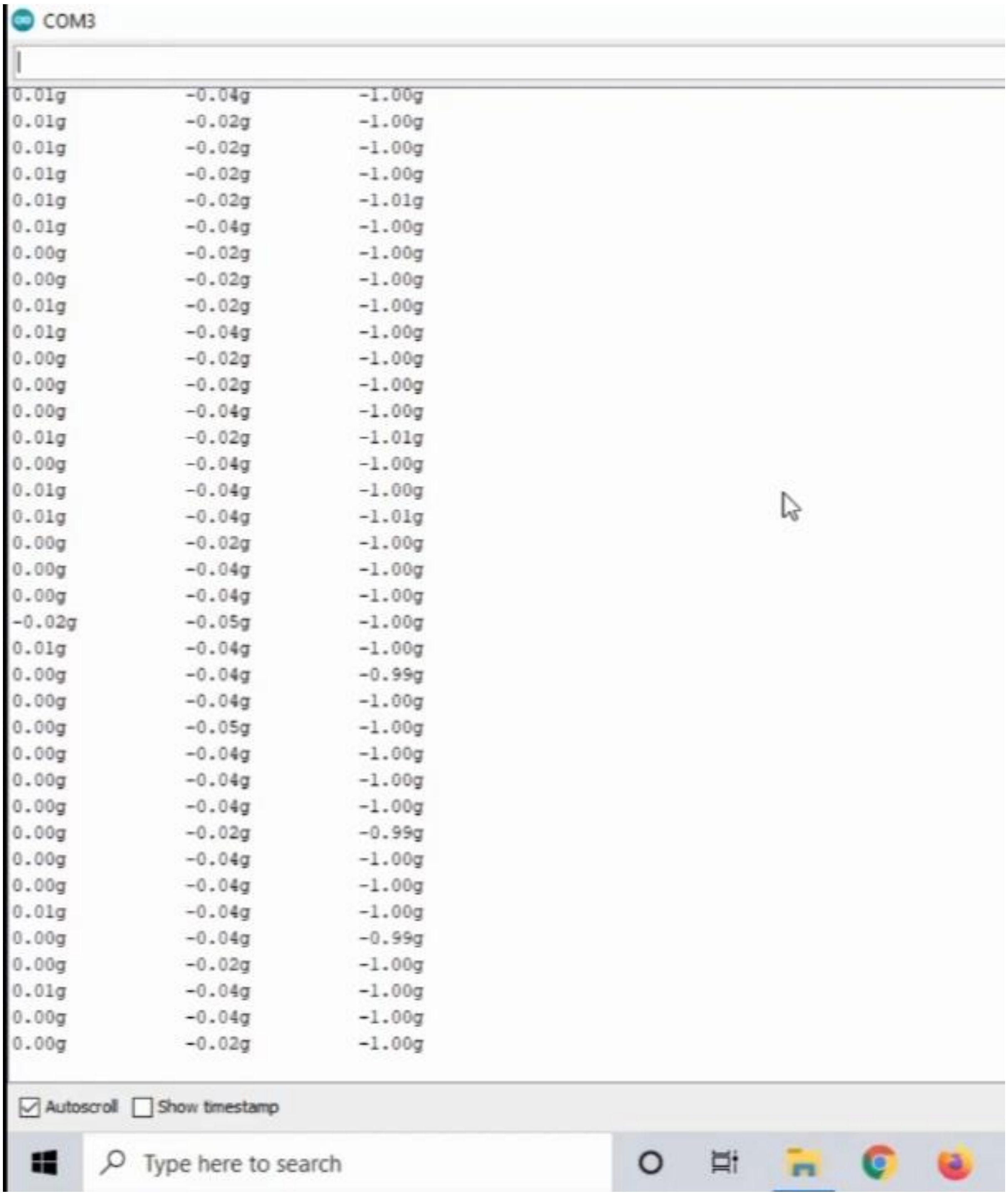
## History:

In 1963, IBM produced computers which specialized in data acquisition. These include the IBM 7700 Data Acquisition System, and its successor, the IBM 1800 Data Acquisition and Control System. These expensive specialized systems were surpassed in 1974 by general purpose S-100 computers and data acquisitions cards produced by Tecmar/Scientific Solutions Inc. In 1981 IBM introduced the IBM Personal Computer and Scientific Solutions introduced the first PC data acquisition products.

## Methodology:

Data acquisition begins with the physical phenomenon or physical property to be measured. Examples of this include temperature, vibration, light intensity, gas pressure, fluid flow, and force. Regardless of the type of physical property to be measured, the physical state that is to be measured must first be transformed into a unified form that can be sampled by a data acquisition system. The task of performing such transformations falls on devices called sensors. A data acquisition system is a collection of software and hardware that allows one to measure or control physical characteristics of something in the real world. A complete data acquisition system consists of DAQ hardware, sensors and actuators, signal conditioning hardware, and a computer running DAQ software. If timing is necessary (such as for event mode DAQ systems), a separate compensated distributed timing system is required.

A sensor, which is a type of transducer, is a device that converts a physical property into a corresponding electrical signal (e.g., strain gauge, thermistor). An acquisition system to measure different properties depends on the sensors that are suited to detect those properties. Signal conditioning may be necessary if the signal from the transducer is not suitable for the DAQ hardware being used. The signal may need to be filtered, shaped or amplified in most cases. Various other examples of signal conditioning might be bridge completion, providing current or voltage excitation to the sensor, isolation, linearization. For transmission purposes, single ended analog signals, which are more susceptible to noise can be converted to differential signals. Once digitized, the signal can be encoded to reduce and correct transmission errors.



```
0.01g          -0.04g          -1.00g
0.01g          -0.02g          -1.00g
0.01g          -0.02g          -1.00g
0.01g          -0.02g          -1.00g
0.01g          -0.02g          -1.01g
0.01g          -0.04g          -1.00g
0.00g          -0.02g          -1.00g
0.00g          -0.02g          -1.00g
0.01g          -0.02g          -1.00g
0.01g          -0.04g          -1.00g
0.00g          -0.02g          -1.00g
0.00g          -0.02g          -1.00g
0.00g          -0.04g          -1.00g
0.01g          -0.02g          -1.01g
0.00g          -0.04g          -1.00g
0.01g          -0.04g          -1.00g
0.01g          -0.04g          -1.01g
0.00g          -0.02g          -1.00g
0.00g          -0.04g          -1.00g
0.00g          -0.04g          -1.00g
-0.02g         -0.05g          -1.00g
0.01g          -0.04g          -1.00g
0.00g          -0.04g          -0.99g
0.00g          -0.04g          -1.00g
0.00g          -0.05g          -1.00g
0.00g          -0.04g          -1.00g
0.00g          -0.04g          -1.00g
0.00g          -0.04g          -1.00g
0.00g          -0.02g          -0.99g
0.00g          -0.04g          -1.00g
0.00g          -0.04g          -1.00g
0.01g          -0.04g          -1.00g
0.00g          -0.04g          -0.99g
0.00g          -0.02g          -1.00g
0.01g          -0.04g          -1.00g
0.00g          -0.04g          -1.00g
0.00g          -0.02g          -1.00g
```

(Fig 18: Serial Monitoring)

## Data Acquisition code in Python:

```
import serial

import pandas as pd

ser = serial.Serial('COM3',9600)

ser.close()

ser.open()

i = 100

z = []

while i > 0:

    data = ser.readline()

    h = data.decode()

    b = h.strip()

    z.append(b)

    i -= 1

dict = {'Z':z}

df = pd.DataFrame(dict)

df.to_csv('data1 500.csv')
```

**Why DAQ should be used ?**

- Improves the efficiency and reliability of processes or machinery. Steel mills, utilities, or a research lab have some kind of data acquisition device that silently monitors some parameter. These collected data can be used to improve efficiency, ensure reliability or ensure that machinery operates safely.

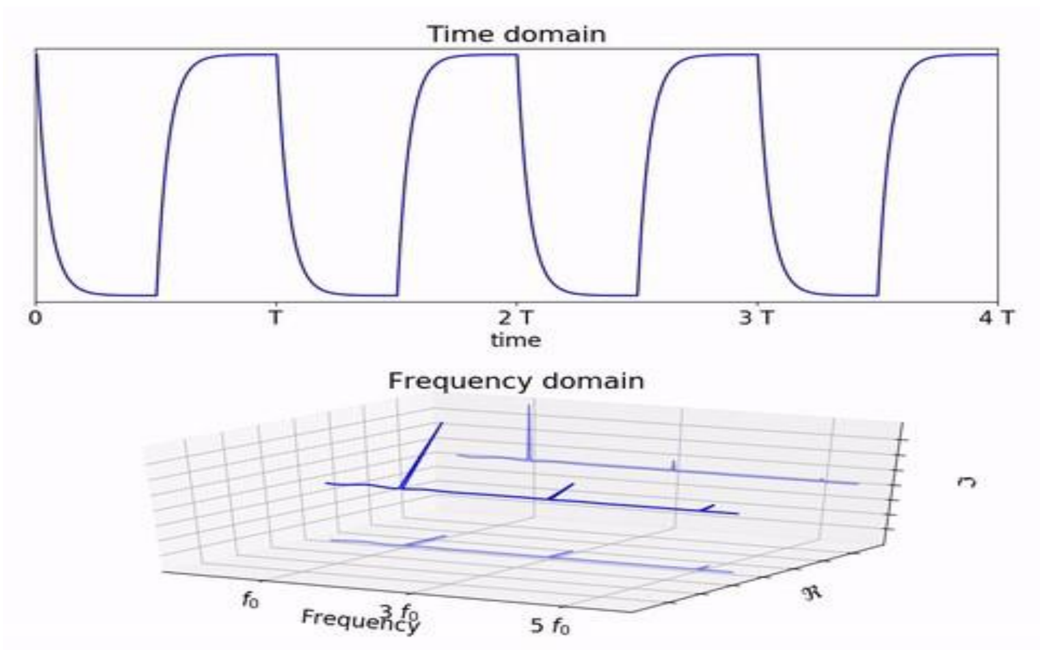
- Problems are analyzed and solved faster. With the use of real-time data acquisition systems, measurements are generated and displayed without delay. Thanks to this system, a technician can intervene faster in any problem and make the machine reach optimum performance in less time.
- Data redundancy is reduced. With the application of a system of this type, companies forget to have duplicate data and adopt a technology that facilitates the analysis of the information obtained, as it allows them to work without any noise that hinders the analysis.
- Decrease update errors. This type of system automates data entry processes that were previously done by hand. Automation reduces errors by eliminating human error and misplacement.
- Increased data integration and reliance on other programs. The fewer programs that intervene in a more agile process, the more agile it will be. Thanks to a DAQ system, it ensures that the information is complete and correct without having to rely on other types of applications.
- Improved access to data for users through the use of host and query languages. With these systems it is easier to access the database and retrieve information for processing and analysis.
- Improves data security. By automating the process of capturing data from reality, the human factor is no longer involved and the security risks associated with this procedure are reduced.
- Data entry, storage and retrieval costs are reduced. These three processes are cheaper because data is entered faster, takes up less space, and can be retrieved in less time.
- Quality control. A system of this type can confirm that a system is meeting the design specifications and that a product meets the user’s needs. In addition, you can test whether a product has the quality required for marketing and detect those that are defective.
- Supervision of processes without human interaction. With such a system, the

company’s various procedures are tracked and monitored to identify and resolve faults faster.

## Time Domain:

Time domain refers to the analysis of mathematical functions, physical signals or time series of economic or environmental data, with respect to time. In the time domain, the signal or function's value is known for all real numbers, for the case of continuous time, or at various separate instants in the case of discrete time. An oscilloscope is a tool commonly used to visualize real-world signals in the time domain. A time-domain graph shows how a signal changes with time, whereas a frequency-domain graph shows how much of the signal lies within each given frequency band over a range of frequencies.

Though most precisely referring to time in physics, the term time domain may occasionally informally refer to position in space when dealing with spatial frequencies, as a substitute for the more precise term spatial domain.



(Fig 19: Time & Frequency Domain)

## Frequency Domain:

In physics, electronics, control systems engineering, and statistics, the frequency domain refers to the analysis of mathematical functions or signals with respect to frequency, rather than time.[1] Put simply, a time-domain graph shows how a signal changes over time, whereas a frequency-domain graph shows how much of the signal lies within each given frequency band over a range of frequencies. A frequency-domain representation can also include information on the phase shift that must be applied to each sinusoid in order to be able to recombine the frequency components to recover the original time signal.

A given function or signal can be converted between the time and frequency domains with a pair of mathematical operators called transforms. An example is the Fourier transform, which converts a time function into a complex valued sum or integral of sine waves of different frequencies, with amplitudes and phases, each of which represents a frequency component. The "spectrum" of frequency components is the frequency-domain representation of the signal. The inverse Fourier transform converts the frequency-domain function back to the time-domain function. A spectrum analyzer is a tool commonly used to visualize electronic signals in the frequency domain.

Some specialized signal processing techniques use transforms that result in a joint time–frequency domain, with the instantaneous frequency being a key link between the time domain and the frequency domain.

## Advantages:

- One of the main reasons for using a frequency-domain representation of a problem is to simplify the mathematical analysis. For mathematical systems governed by linear differential equations, a very important class of systems with many real-world applications, converting the description of the system from the time domain to a frequency domain converts the differential equations to algebraic equations, which are much easier to solve.

- In addition, looking at a system from the point of view of frequency can often give an intuitive understanding of the qualitative behavior of the system, and a revealing scientific nomenclature has grown up to describe it, characterizing the behavior of physical systems to time varying inputs using terms such as bandwidth, frequency response, gain, phase shift, resonant frequencies, time constant, resonance width, damping factor, Q factor, harmonics, spectrum, power spectral density, eigenvalues, poles, and zeros.

- An example of a field in which frequency-domain analysis gives a better understanding than time domain is music; the theory of operation of musical instruments and the musical notation used to record and discuss pieces of music is implicitly based on the breaking down of complex sounds into their separate component frequencies (musical notes).

## Magnitude and phase:

In using the Laplace, Z-, or Fourier transforms, a signal is described by a complex function of frequency: the component of the signal at any given frequency is given by a complex number. The modulus of the number is the amplitude of that component, and the argument is the relative phase of the wave. For example, using the Fourier transform, a sound wave, such as human speech, can be broken down into its component tones of different frequencies, each represented by a sine wave of a different amplitude and phase. The response of a system, as a function of frequency, can also be described by a complex function. In many applications, phase information is not important. By discarding the phase information, it is possible to simplify the information in a frequency-domain representation to generate a frequency spectrum or spectral density. A spectrum analyzer is a device that displays the spectrum, while the time-domain signal can be seen on an oscilloscope.

## Types

Although "the" frequency domain is spoken of in the singular, there are a number of different mathematical transforms which are used to analyze time-domain functions and are referred to as "frequency domain" methods. These are the most common transforms, and the fields in which they are used:

- Fourier series – periodic signals, oscillating systems.
- Fourier transform – aperiodic signals, transients.
- Laplace transform – electronic circuits and control systems.
- Z transform – discrete-time signals, digital signal processing.
- Wavelet transform — image analysis, data compression.

## Discrete frequency domain:

The Fourier transform of a periodic signal has energy only at a base frequency and its harmonics. Another way of saying this is that a periodic signal can be analyzed using a discrete frequency domain. Dually, a discrete-time signal gives rise to a periodic frequency spectrum. Combining these two, if we start with a time signal which is both discrete and periodic, we get a frequency spectrum which is also both discrete and periodic. This is the usual context for a discrete Fourier transform.

## 6. Conclusion

The relationship between the vibration of the machine and failure is obtained by measuring the vibration of the machine and analysing the signal of vibration to evaluate the machine condition. Early deduction of the machine state will help schedule activities and conveniently reduce downtime and losses for predictive maintenance. The acceleration of vibration signals was plotted on monitor display with the help of Arduino IDE and Python software. In this project we monitor Universal Vibration Test Rig by using A.I. approach.

## 7. References

1. Novel Machine Health Monitoring System by M. S. Shewale et al (2019)

2. Towards devising a vibration based machinery health monitoring system by Md. Harunur Rashid Bhuiyan, Iftekhar Arafat, M. Rahaman et al (2021)

3. Condition monitoring based on IoT for predictive maintenance of CNC machines by Yahya Mohammed Al-Naggar et al./ Procedia CIRP 102 (2021) 314-318

4. The Experimental Application of Popular Machine Learning Algorithms on Predictive Maintenance and the Design of IoT Based Condition Monitoring System by Mustafa Cakir, Mehmet Ali Guvenc, Selcuk Mistikoglu (2020)

5. An introduction to MEMS vibration monitoring by Mark Looney, Analog Dialogue 48-06 (2014)