RAE

1. **TIPO DE DOCUMENTO:** Trabajo de grado para optar por el título de INGENIERO MECATRÓNICO.
2. **TÍTULO:** Banco de pruebas para el análisis de fallas en sistemas rotacionales.
3. **AUTORES:** Alejandro Rojas López, Carlos A. Castellanos López, Diego A. Delgadillo Murillo.
4. **LUGAR:** Bogotá, D.C.
5. **FECHA:** Julio de 2020.
6. **PALABRAS CLAVE:** Detección de fallas en motores de inducción, Banco de pruebas, Vibración, Acústica, Corriente, Velocidad, Termografía, DAQ, FFT, HMI, Rodamiento, Jaula de ardilla, Desbalance, Rotor, Estator, Matlab, AppDesigner.
7. **DESCRIPCIÓN DEL TRABAJO:** Estudios anteriores han abordado la detección de fallas y el aislamiento de los motores de inducción y se ha propuesto una amplia variedad de métodos de detección. Sin embargo, la mayoría de estos estudios utilizan una sola señal para realizar el proceso de detección de fallas. En este trabajo se describe el desarrollo de un banco de pruebas que analiza fallos como el desequilibrio de la carga, la degradación de los cojinetes y las barras rotas (rotor) en los motores de inducción. El banco de pruebas también está instrumentado con sensores de vibración, velocidad y sonido que permiten una mejor comprensión de estos fallos. Los datos obtenidos pueden utilizarse en futuras investigaciones para desarrollar sistemas de control tolerantes a las fallas y estrategias de detección de fallas. Este trabajo describe el diseño conceptual y las etapas de diseño detallado, integración y prueba de este banco de pruebas.
8. **LÍNEAS DE INVESTIGACION:** Procesamiento de señales, Adquisición de datos, Mantenimiento.
9. **METODOLOGÍA:** Para el desarrollo de este proyecto se empleó el uso del lineamiento VDI 2206.
10. **CONCLUSIONES:**

    -El análisis de las investigaciones anteriores nos permite concluir que los principales fallos de los motores de inducción de los sistemas rotativos pueden dividirse en fallos del estator y del rotor, desequilibrio mecánico y desgaste de los cojinetes. Esta caracterización estableció los requisitos del banco de pruebas.

    -El mecanismo de reproducción de las fallas fue desarrollado bajo la directriz VDI 2206. A partir de este punto, el banco de pruebas se descompuso en subsistemas funcionales y se diseñaron los componentes críticos. En este aparte, el diseño del HMI facilita la adquisición de datos que captura datos sincronizados en tiempo real para tratar más allá la señal implementando métodos como la FFT para mostrar información gráfica organizada al usuario. Por otro lado, el desarrollo de piezas mecánicas intercambiables para el banco de pruebas facilita la reproducción de las fallas.

    -Los experimentos se llevaron a cabo mediante un proceso didáctico. Esto permite verificar el correcto funcionamiento del sistema de adquisición de datos y del HMI. El análisis espectral de los datos de aceleración permite identificar la frecuencia fundamental de la oscilación. Además, la captura de sonido también muestra diferencias para cargas equilibradas y desequilibradas. Por último, en estas condiciones los datos de velocidad no dieron lugar a datos concluyentes.

Test Bench For Fault Analysis In Rotational Systems

UNIVERSIDAD DE SAN BUENAVENTURA BOGOTÁ

Alejandro Rojas López
Carlos A. Castellanos López
Diego A. Delgadillo Murillo

Major:
Mechatronics Engineering

Universidad de San Buenaventura Bogotá
Faculty of Engineering
Bogotá D.C., Colombia
2020

Test Bench For Fault Analysis In Rotational Systems

Alejandro Rojas López
Carlos A. Castellanos López
Diego A. Delgadillo Murillo

Thesis Director:
Edwin Villarreal López

Area of Research:
Industrial Maintenance
Signal Processing

Universidad de San Buenaventura Bogotá
Faculty of Engineering
Bogotá D.C., Colombia
2020

Abstract

Many previous studies have tackle fault detection and isolation on induction motors and a wide variety of detection methods have been proposed. Nevertheless, most of these studies use only one signal to perform the fault detection process. In this work, the development of a test bench that analyzes faults such as load imbalance, bearings degradation and broken bars in induction motors is depicted. The test bench is also instrumented with vibration, velocity and sound sensors allowing a better understanding on these faults. The data generated can be used in future research to develop fault tolerant control systems and fault detection strategies. This work describes the conceptual design and detail design, integration and testing stages of this test bench.

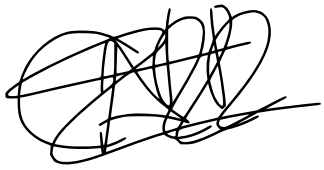
Keywords: Bearing, Squirrel cage, Imbalance, Fault detection and isolation on induction motors, Acoustic signal, Vibration signal, Test Bench, DAQ, FFT, HMI

Statement

Let me affirm that I have done this thesis in an autonomous way and with only the help of the allowed means and not different from those mentioned in the thesis itself. All passages that have been taken textually or figuratively from texts published and unpublished, I have recognized there job here. No part of this work has been used in any Another type of thesis.

Bogotá, D.C., 08.07.2020

Alejandro Rojas López



Carlos Castellanos López

Diego Alexander Delgadillo Murillo

Index

List of Figures

List of Tables

1. Introduction

Industrial advances and the automation of different systems have created a fourth industrial revolution, called industry 4.0, which consists of combining advanced production techniques with intelligent technologies that are integrated into organizations, people, and assets. This revolution is marked by the integration of technologies such as robotics, data analytics, artificial intelligence, cognitive technologies, nanotechnology, and the Internet of Things (IoT), among others. This new revolution aims to recover the information of the physical world and create a digital record of it; Also, share and interpret this data using advanced analytics and scenario analysis.

In industrial sectors, induction motors are present in most of the activities, and this is due to their easy maintenance and efficiency in specific tasks. These characteristics are exploited in a wide variety of applications including production systems, home appliances and transport mechanisms

Due of this boom of applications, and aligned with the 4.0 industry paradigm, A new field of research is focusing on monitoring and improving the conditions of each motor, as well as in its useful life and maintenance, increasing productivity while reducing costs and down-times.

A detailed review made it possible to determine the importance of fault detection techniques in mechanical systems with induction motors. The development of this research will focus on a simultaneous analysis of various signals that allow us to advance in the understanding of the nature of different failures present in this type of systems.

Based on the fact that most of the sources consulted focus on a specific type of signal, for example, audio, vibration, speed, among others, a combined management from various sources can improve the quality and robustness of a detection system. The use of this condition monitoring technology can improve the performance of rotational systems with induction motors, since there will be more ways to detect faults, and thus give possible solutions and provided preventive maintenance before any definitive damage.

From the literature review, a knowledge gap can be evidenced in terms of the detection

of failures in rotation systems using multiple sensor information, increasing sensibility and reducing detection times.

Therefore, the purpose of this project is to create a test bench, that allows the development and testing of failures identification techniques based on multiple sensor information.

The main difference between this approach and existing test beds is the use different sources of signals acquired simultaneously for the analysis of the same system failure. This test bench will give a broader and precise range of information of the type of failure or cause of failure in the motor, in order to use this information in real-time control systems and preventive maintenance strategies.

This document is organized as follows, first a detailed review on previous research on fault detection and isolation in induction motors is depicted. Then, the conceptual and detailed design of the test bench subsystems is described. Later, some initial experiments are detailed to verify the test bench performance under different fault scenarios. Finally, finally, some conclusions about the process are given.

1.1. Objective

1.1.1. General Objective

Implement a test bench for fault analysis in an induction motor driven rotational system.

1.1.2. Specific Objectives

- Characterize a set of failures of significant incidence in rotational systems with induction motors.
- Determine the mechanisms of reproduction of the selected failures in a test bench and the means of data acquisition.
- Obtain experimental operating data of the rotational system under normal and faulty operating conditions.

1.2. Scope and limitations

- Carry out structural calculations of critical components.
- The acquisition system captures signals of: Vibration, Audio, current and velocity, The resolution and the sampling frequency will be defined from the literature review.
- Failures are characterized.
  Mechanics: Load imbalance, bearing failure
  Electrical: Squirrel cage rotor bar breakage, alteration in armature resistance.
- Interchangeable mechanical parts are implemented to reproduce different faults.

2. State of the art

Due to the high production requirements of modern manufacturing systems, unscheduled stops caused by failures, breakdowns or machine damages generate significant losses to companies. This fact has increased the attention to the development of new technologies that avoid the occurrence of these events by detecting incipient faults

Due to their robustness, induction motors are widely used on industrial processes. Nevertheless, there are machines that, over time, due to poor maintenance and natural wear, present different failures or damages that, if not treated in time, cause financial losses and loss in production time.

For the development of this thesis, a rigorous investigation was carried out of various academic and scientific databases, searching for investigations focused on the analysis of failures on induction motors and the different fault detection techniques used, thus coming to find extensive information in this field of research. It was possible to obtain a clearer vision of the studies already carried out and how they can contribute to the development of this project, being able to fill gaps in this line of research involving the different techniques studied for the implementation of the test bench.

2.1. Research Matrix

From the research carried out on this topic, the most relevant scientific articles were consolidated and classified by the different failures presented in the induction motors and also by the techniques used to detect and analyze them. This structure contributes in their analysis and allow the authors to draw conclusions related to the type of faults that must be included into the test bench.

	Failure			
Reference	Bearing	Imbalance	Stator	Rotor
Acoustic				
[1]				x
[2]	x	x		x
[3]		x		
[4]			x	x
[5]	x			
[6]				x
[7]	x	x		x
[8]	x			
[9]			x	x
[10]	x			x
Current				
[11]				x
[12]			x	
[13]	x			
[14]				x
[15]	x			
[16]	x			
[17]			x	
[18]			x	
[19]				x
[20]		x		
[21]				x
[22]			x	
[23]		x		
[24]				x

Continued on the next page.

Reference	Bearing	Imbalance	Stator	Rotor
[25]			x	
[26]			x	
[27]	x			
[28]			x	x
[29]			x	
Thermal				
[30]	x			
[31]				x
[32]	x	x		x
[33]			x	
[34]				x
[35]			x	
[36]	x			
[37]	x	x		
[38]	x	x		
[39]				x
Vibration				
[40]	x	x		x
[41]	x			x
[42]				x
[43]	x			
[44]				x
[45]	x	x		x
[46]	x	x		x
[47]	x	x		x
[48]		x		
[49]		x		
Velocity				
[50]				x

Table 2-1.: Analysis of common failures for preliminary design of the project from the state of the art

2.2. Failures in induction motors

2.2.1. Fault by Imbalance

The initial study focused on diagnosing the main failures or damages in rotating machines, taking into account literature, scientific research, articles, academic journals, and other studies, in which different causes were found. The most common of these is the imbalance in the rotor [51]

Additionally, some types of specific imbalances are mentioned, such as the torque imbalance, which is characterized by rotating the rotor around its vertical axis, and the two imbalances forces generate a torque on the rotor. Static imbalance occurs when the central longitudinal axis of inertia of the rotor is displaced parallel to the axis of rotation. A dynamic imbalance, the central longitudinal axis intersects the axis of rotation of the rotor. [52]

Imbalance damages are the cause of vibrations, which depending on the type of imbalance, will be obtained in a greater magnitude of frequency. Having a vibration in the machine will reduce motor performance. By having the motor in operation, the probabilities of a failure due to breakage or wear will be guaranteed.

2.2.2. Eccentricity

Going deeper into the investigation, it is established that Eccentricity is another of the most common electromechanical errors or problems in induction motors. Which is more or less, the center-line of the shaft is not the same as the center-line of the rotor. It is due to a higher weight on one side of the center of rotation than on the other.

The eccentricity fault occurs in the rotor as in the stator. According to the study by Kenji Yamamoto, Cesar da Costa, and Joao da Silva Sousa in their academic journal, Eccentricity in the rotor Occurs when Eccentricity occurs in the rotor. It is not concentric with its axis of rotation. This also causes a non-uniform distribution of the spacing between the rotor and the stator. In this case, the non-uniformity is not stationary but moves together with the rotor rotation. In other words, the eccentric rotor rotates in the center of the stator, but this does not coincide with the geometric center of its center of rotation, acts as a cam, which causes different air gaps between the rotor and the stator [52].

It is important to note that an induction motor with an eccentric rotor will increase the vibration levels throughout the operating time and increases the temperature levels, since,

in some parts, there will be friction.

Also, the magnetic force generated in the area where the rotor and the stator are closest will be generators of highly directional vibration. Such a defect may have its origin not only in the deficiencies that the motor winding may have had during any repair but also in distortions that the motor's internal structure may have suffered due to blows due to fall.

2.2.3. Electrical Failures

Induction motors not only have mechanical faults, but also electrical failures are common in these machines, as these are also exposed to loads, contaminants, construction defects. The vast majority of electrical failures in induction motors are caused by construction defects, especially after winding repairs. One of the most common electrical fault is the asymmetry of the stator windings. This causes weak magnetic fields rotating in the opposite direction to the main stator magnetic field, causing a more considerable effort directly proportional to the square of the current.

Asymmetry failure on the rotor windings (cage) appears due cracks, broken or loose bars damage the stator windings when deformations occur. In addition to this fault, the fault in the package of stator coils is attached. That consists of wear or deterioration in the coils insulation, which is the cause of heating, overvoltage, movement of the coils, and high currents.

A no less important problem causing vibration is the stator winding loose. That is, the windings are slightly loose, increasing the vibration frequency twice or more. That causes a fault in the conductors insulation leading to a short circuit between the windings. [29]

Finally, the blocked rotor failure is caused by a thermal deterioration of the insulation of the three phases of the motor, caused by excessively high currents due to the rotor blockage. This goes hand in hand with a bearing failure that appears due to excessive loads, misalignment, improper or incorrect lubrication, and deformation under load. High vibration frequencies manifest these failures. However, as the damage becomes more significant, the frequencies start decreasing. [15]

2.3. Analysis techniques

2.3.1. Acoustic analysis

The acoustic analyses of faults in induction motors is a non-invasive technique. Like all other techniques for analyzing faults in induction motors, it comes with pros and cons. Its pros are that it comes at a low cost, and the measurements are instantaneous. The cons of this method are that the acoustic signal can be affected by other types of signals. Another con to this method of measurement is that the location of the fault cannot be determined. To uses this method to determine faults, the use of a microphone and a computer are needed. Low frequencies are required for condition monitoring. This method of analysis uses previous recordings of healthy motors and motors with specific faults to create a database, that is later used to compare the recording of the motor being analyzed. This is done by splitting the recording into smaller files and then running these files through filters. And then comparing them to the previous test to determine what fault the motor has. the size and quality of the database used determines the quality of the acoustic analyses.[1] To process the acoustic signals the Fast Fourier Transform and Wavelet Transform are commonly used.[3]

In [1] the authors present a new analysis technique for single-phase motors, as it does in previous articles, implements a microphone that is the signal receiver, It begins the process by dividing the signal into smaller soundtracks. It introduces them into an amplitude normalizer, then an FFT (Fast Fourier Transform), then the Shortened Method of Frequencies Selection Multiexpanded (SMOFS-22-MULTIEXPANDED), that uses the Nearest Neighbor classifier. The data files were then processed by the FFT method and the Hamming window. The FFT calculated 16,384 frequency components. The frequency components were processed by the SMOFS-22-MultieExpanded witch calculated 1-22 vectors of common characteristics.
With each of the signals taken, the author makes some audio files to make a prediction phase. This is done by finding patterns of acoustic signals from faulty motors. This process facilitates the detection of failures because when comparing signals from an motor under inspection, with one already studied, the fault of the motor under test can be easily identified.

2.3.2. Current analysis

Current fault analysis is commonly used for a specific fault, and it is the failure by bearings since these are connected to the rotor. When there is a fault, it produces vibrations that affect the air gap eccentricity between the stator and the rotor and induces a stator current fault frequency. In [26] the stator current is taken from the induction motor under healthy and defective bearing conditions and processed by spectral subtraction using different wavelet

decomposition techniques. The stator current after spectral subtraction is used for the fault indexing parameter. The experimental setup contains a 3 phase, 4 pole induction motor with a charging speed of 1435 rpm. A three-phase automatic transformer is used to power the motor. The current signature is extracted in the computer through the data acquisition system and normalized. The sampling frequency is 10 kHz, and the number of samples is 10,240 to avoid errors during reconstruction. Deep groove ball bearings, single row, are used at both the motor and drive ends. Different wavelet transformation techniques are compared to discover bearing failures in the induction machine using current frequency spectral subtraction.
Another author implements a technique based on analyzing the motor current signature (MCSA), although it is clear that this type of analysis has some limitations when it comes to the function of induction motors. The proposed technique is based on three main steps: first, the induction motor current is measured; second, the square of the current is calculated, and finally, a frequency analysis of the square current is performed. This technique allows obtaining more information on a motor with rotor failure than the classic MCSA. [13]

2.3.3. Thermographic analysis

The thermographic analyses of induction motors to detect faults are a non-invasive method [30]. This method consists of using a thermographic camera to take images of the induction motor in use to allow the visualization of the surface temperature of the motor. A thermographic camera can take two types of measurement direct or indirect. Direct measurements are the ones that have no thermal isolation between the power source and the thermographic camera, and the indirect measurement is when there is isolation between the power source and the sensor. For the analyses of induction motor, the measurement is considered indirect. For analyses in systems that have an indirect measurement, a higher understanding of how the system works is required. One of The drawbacks of thermographic analyses for fault detection is that the cost of the required devices.

A straightforward way to carry out this analysis is by utilizing thermal images when the motor is in operation. Overheating can be detected; this can be an Indicator of a specific failure. This analysis is carried out by Adam Glowacz, in their academic article "Diagnostics of stator faults of the single-phase induction motor using thermal images, MoASoS (Method of Areas Selection of Image Differences) and selected classifiers." [31] Where the techniques of diagnosis of single-phase induction motor failures were described. The techniques presented were based on thermal image analysis.

For the analysis of the failures, they measure and analyze 3 states of the single-phase induction motor. In this paper, an original method of extracting features from thermal images called the Area Selection Method of States and an image histogram to form feature vectors. The classification of the vectors obtained was carried out by Nearest neighbor classifier, that is, comparing the said result with a good motor and Gaussian mixture models.

Failure analysis through thermal images gives excellent support to complement the other types of analysis since the thermal image can give more specific information on where the fault is located, by means of the colors intensity.

2.3.4. Vibration analysis

In [46] a type of vibration analysis was performed, which consists on taking the vibration signal by a tri-axial accelerometer based on micro-electromechanical system (MEMS) (model LIS3L02AS4) from SMT electronics. Of the three acceleration axes (X, Y, and Z) of the vibration signal, from which the best results are obtained with the signal corresponding to the x-axis (AX).

The 4-channel digital data converter (ADC) with serial output (ADS7841) is used to acquire vibration data. The instrumentation system uses 1500 Hz sampling frequency to obtain 4096 samples of vibration signals during induction motors in a steady-state. Additionally, a motor is taken to which two 7.9mm diameter holes are made without damaging the shaft, to analyze the vibration failure.

These elements implemented to determine the condition of the motor, each of the acquired vibration signals, pass to an analysis phase. The decomposition is carried out through complementary ensemble empirical mode decomposition (CEEMD) and, subsequently, the obtained IMF is transformed into the frequency domain. The transformation to the frequency domain is performed using the calculation of the marginal frequency of the Gabor representation time-frequency (TFDG). Finally, once the intrinsic mode functions (IMF) spectra is obtained, the motorization condition is evaluated by frequency analysis related to the failures.

These responses are compared with the frequencies of a good condition motor to validate the method used and have a guide signal that determines if there is a failure due to rotor breakage.

Continuing with the investigation, another group of researchers used a different method

for vibration analysis. In [53] the authors present a method for detecting and diagnosing electrical failures in the stator through vibration analysis, which consisted of identifying characteristic frequencies of those failures.

The faults were inserted in a three-phase motor, WEG (FH 88747), squirrel cage rotor, 5 hp, 1730 rpm, 220 V, 60 Hz, category N, 44 bars, 36 slots, bearing SKF 6205-2Z, ID-1, 100L frame. A DC generator that feeds a resistor bank is used as a charging system. The generator is connected to the electric motor through flexible couplings and a torque meter that could guarantee the same operation and full load conditions in all tests.

The experiment continues to collect spectra of magnetic flux and vibration (total of 120), in a series of 10 tests for each reaction (without failure, short circuits of 2, 4, 6 and 10 turns, imbalance phase) , and repeated randomly under them loading conditions The NI-6251 board manufactured by National Instruments was used for data acquisition. This board has 16 analog input channels that can display up to 200 kHz and 2 digital counters of 24 bits each.

The analog inputs have a resolution of 16 bits. Magnetic flow and vibration signals were sent to an anti-lap filter with a cutoff frequency of 2.5 kHz. Matlab software was used to implement the data acquisition algorithm and fault diagnosis. This analyzed data to find a characteristic frequency for this type of error. In [54] the authors describe the implementation of a method for the analysis of failures, Which consists of analyzing the vibrations of the base and the bearings of an induction motor, with control systems in place and without them. From this analysis, they noticed that with the vibration control system, the motor could use the speed range for all operations. However, without it, steady-state operation is not possible; the objective of this article was to show the use of vibration to control the system. This was done by using actuators between the motor feet and the base. The way the data of the fault is taken is essential because the way that it is adjusted can minimize the external noises and obtain more precise data.

After consulting different articles, it is evident that to analyze failures by vibration, the implementation of a sensor (accelerometer) to acquire data becomes necessary. Later this data will be processed in a computer program that will generate a graph. These graphs will show different peaks that will be identified as the characteristic frequencies for the type of problem presented in the motor. Additionally, it should be taken into account that when using this type of analysis, the sensor must be mounted very close to the motor so that the acquisition of the data is as accurate as possible. The problem arises in being very close to it. The Signal can be distorted by different noises associated with the motor, not including mechanical failure. But a signal due to a mismatch in the bank or some element close to the

machine.[54]

2.4. Characterized Faults

After carrying out a conceptualization of the investigation, the most recurrent failures in the induction motors are evident, for this reason, this chapter can be concluded by characterizing the failures to proceed with the development of the implementation.

The most common faults, as can be seen in the Table **2-1**, occur in most of the motor components such as bearing, rotor (squirrel cage), stator. Additional to this, faults can occur after the axis of rotation of the motor, such as a load imbalance or axis misalignment between the motor and the axis of the driven machine, from this one has a more general idea of what the requirements and characteristics are to meet the main project objective .

The stator failure can be generated by a variation in the armature resistance which generates a voltage imbalance and a high current circulation, and this leads to the stator windings overheating, reducing the useful life of the thermal insulation until it breaks the winding, another common failure is caused by wear from other types of failures such as load imbalance, since having unbalanced the rotor can generate excessive contact with the stator.

The failure in the rotor (squirrel cage) can occur due to a break in the bar caused by lack of preventive maintenance since this piece is in constant motion and is made of copper or aluminum, another of the most common failures is the break of the clamping rings thus producing a malfunction which can cause another type of failure such as imbalance.

The imbalance in the rotary systems occurs when there is an uneven distribution of the mass along the axis, which causes the center of mass to be different from the center of rotation and a centrifugal force is produced which reproduces vibrations throughout the coupled system.

Failure due to bearing in poor condition is commonly caused by lack of lubrication or the end of the bearings useful life, this type of failure generates a high effort of rotation to the motor, causing other types of failures such as excessive consumption of electrical energy, which can generate high temperatures and wear on the other components.

From the characterization of the failures previously analyzed, it is taken as a starting point for the implementation of the test bench, since from these the mechanisms of reproduction of failures, requirements and characteristics are determined, to fulfill the main objective of the project.

3. Implementation

The VDI 2206 design guideline in mechatronics was taken as a starting point to start with the structuring of the project implementation. This guideline helps the correct development of the project based mainly on state of the art, analyzing and discussing the main requirements, scope, and limitations.

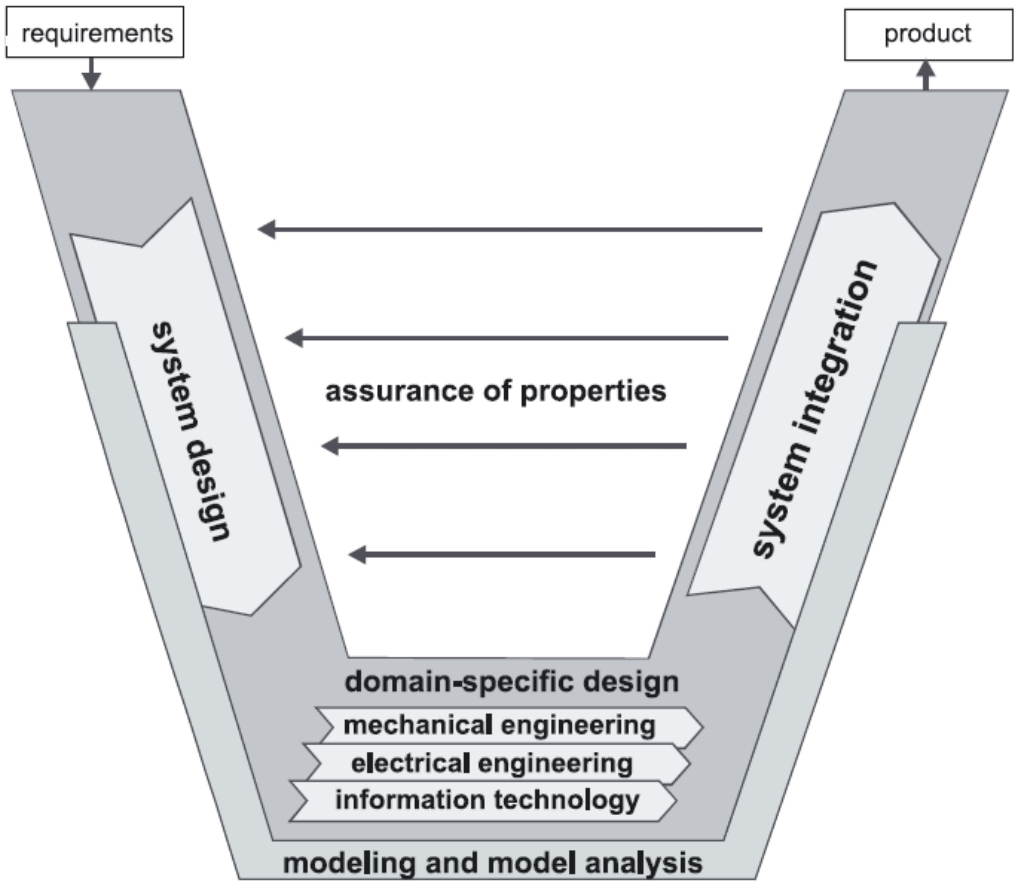


Figure 3-1.: The V-Model according to guideline VDI 2206 (VDI, 2004; Gausemeier and Moehringer, 2003)

3.1. Requirements

Starting from the conclusion from state of the art, the characteristics of the project are defined in general terms to determine how the complete system will be conceptualized under specific criteria determined from the research. Such as the characterization of failures that will be implemented in the test bench. With this characterization, decisions are made on how these failures will be reproduced and what type of data acquisition will be used, this to define the requirements and characteristics of the test bench.

For the fault reproduction part, several elements are required to simulate the different faults covered, such as the exchange of easily accessible mechanical parts; These components correspond to:

- Bearing in good and bad condition
- Rotor (Squirrel cage) In good and bad condition
- Adjustable inertial imbalance disc
- Resistance bank for an alteration of armature resistance
- Variable frequency drive for speed control

It is necessary to use some load to simulate the operation under the real conditions of a system powered by an induction motor. For this reason, it was decided to use a generator coupled to the motor utilizing a shaft, altering its field current to emulate a load.

For the data acquisition, it was necessary to use different sensors that provide particular measurements to be analyzed. Based on the investigation and the conclusion of this, it was decided to implement 3 types of data acquisition, which are mentioned below.

Vibration An accelerometer that is capable of taking acceleration measurements on all 3 axes is required with a quantity of at least 6 samples per revolution, taking into account the nominal values of motor operation.

Acoustic A good quality microphone is required for audio capture. In addition to this, it is convenient to implement acoustic isolation to the test bench to reduce unnecessary information such as environmental noise.

Velocity A high-resolution incremental encoder is required to obtain the motor angular position and be processed to determine its speed.
For data acquisition, it is necessary to use an embedded system capable of capturing the signals delivered by each sensor and performing the respective treatment to obtain data in real-time, taking into account the processing speed of the embedded system and the communication interface with the computer.
It is required to make a human-machine interface (HMI), which allows us to interact with the acquisition of data in real-time to be stored and processed later.

3.2. Conceptual Design

Starting from the global requirements established for the development of the project, a conceptualization of the system is made, dividing it into 3 subsystems, as shown in Figure **3-2**. The first subsystem refers to the mechanism of reproduction of failures, which is made up of different interchangeable components that allow reproducing the faults. The second subsystem refers to the data acquisition system, which consists of the respective measurement sensors and the embedded signal acquisition and treatment system. The third subsystem refers to the human-machine interface (HMI) consisting of a PC and a communication interface between the embedded system and the PC.

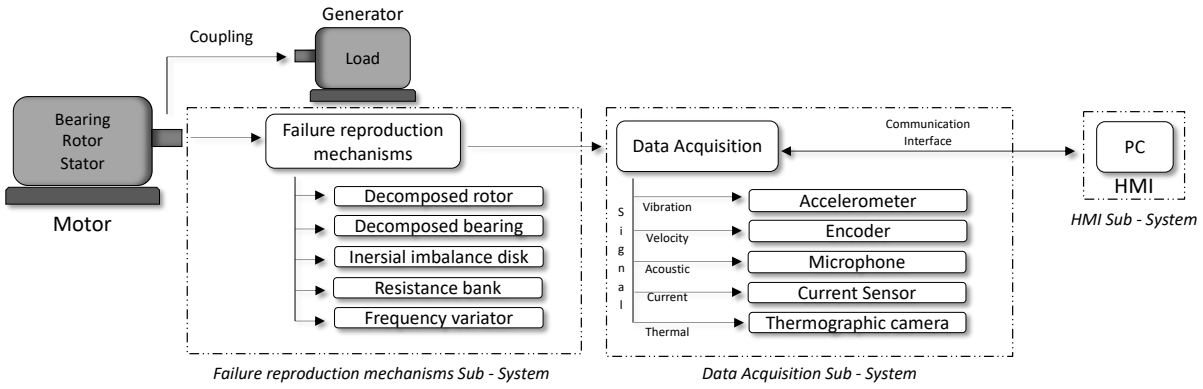


Figure 3-2.: Conceptual Design

3.3. System Design

From the conceptual design proposed, each subsystem is detailed, taking into account the project requirements, specifying in detail the operation and the integration process in the complete system, always looking for the best options that satisfy the general requirements. The following appendix, A.1 has all the blueprints for the following sections.

3.3.1. Failure reproduction mechanisms Sub-System

Imbalance

It is necessary to implement some mechanism that generates an imbalance through the motor shaft to reproduce the unbalance failure. For this reason, it was proposed to make an inertial imbalance disk, as shown in the Figure **3-3**, which allows through grooves to adjust the magnitude of the imbalance by employing a sliding mass. To implement the disc it is necessary to position it on an axis which will be coupled to the motor shaft, for this reason, it is necessary to design the shaft in detail as this will be a critical element of the entire system; As a result of unbalanced load, normal bending stress is produced through the shaft that can cause deformation.

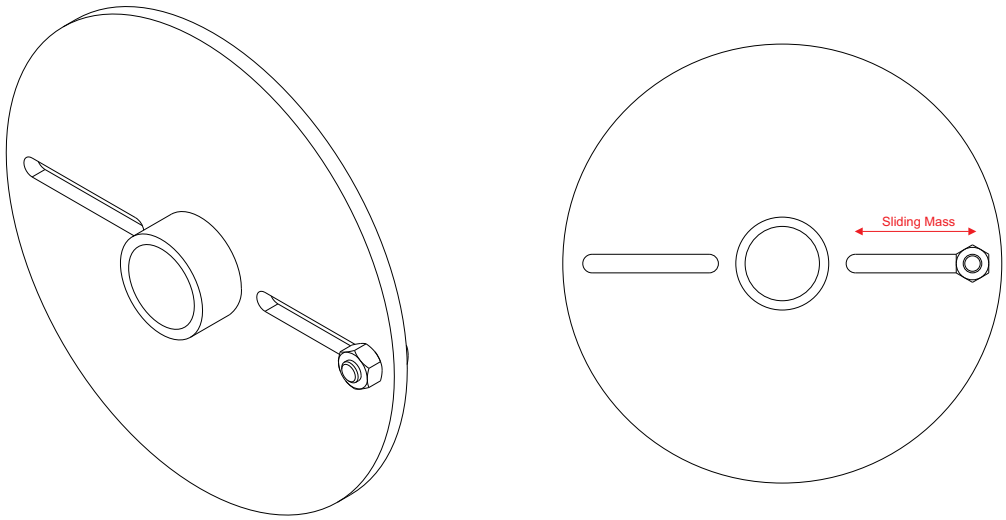


Figure 3-3.: Inertial imbalance disk

Axis Design A detailed calculation is made for the shaft since this is a critical component due to the forces generated on it from the operation of the test bench. The critical force that is exerted on the shaft is generated by the imbalance caused by the adjustable mass of the inertial imbalance disc, from this the shaft diagram is proposed.

The analysis that will be carried out to determine the safety and deformation factors of the axis implemented with the dimensions established so that the axis is symmetrical to the motor axis is carried out by making a stress analysis due to the off-center load which generates a centrifugal force. Being in constant rotation from the movement of the motor; This analysis is done statically since the generated centrifugal force is located perpendicular to the axis so that the flexion of the shaft will be presented in parallel to the force exerted by the off-center mass when the system reaches dynamic equilibrium, it is for this reason that a static analysis is performed.

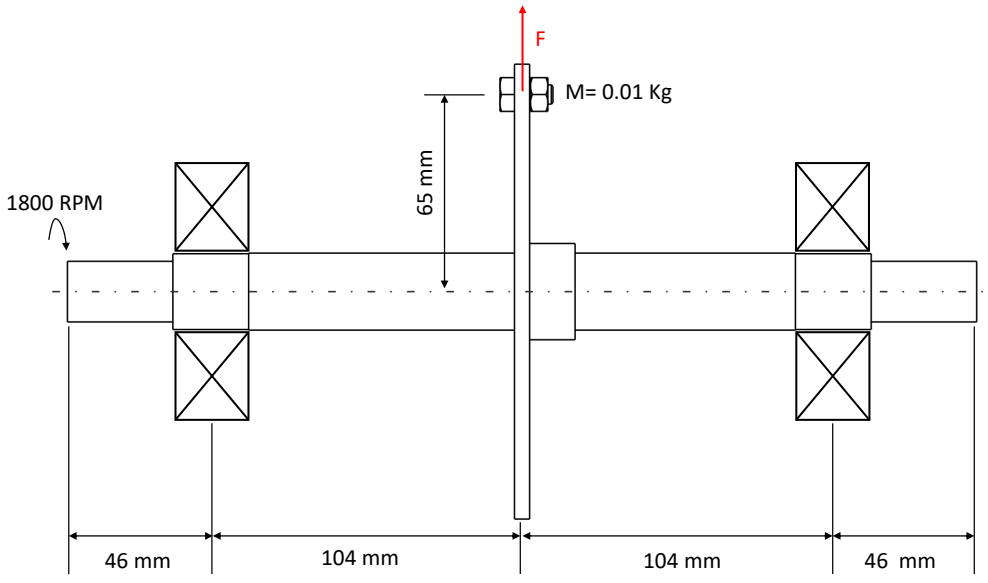


Figure 3-4.: Axis Diagram

using the table A-20 from [55] mechanical properties of steels

$$Sy = 210MPa \tag{3-1}$$

$$Sut = 380MPa \tag{3-2}$$

to calculate the centrifugal force the following equations were used.

$$\omega = 1800_{RPM} \cdot \frac{2 \cdot \pi}{60 \cdot RPM \cdot seg} = 188{,}496 \frac{rad}{seg} \tag{3-3}$$

$$F = m \cdot \omega^2 \cdot r = 23{,}095N \tag{3-4}$$

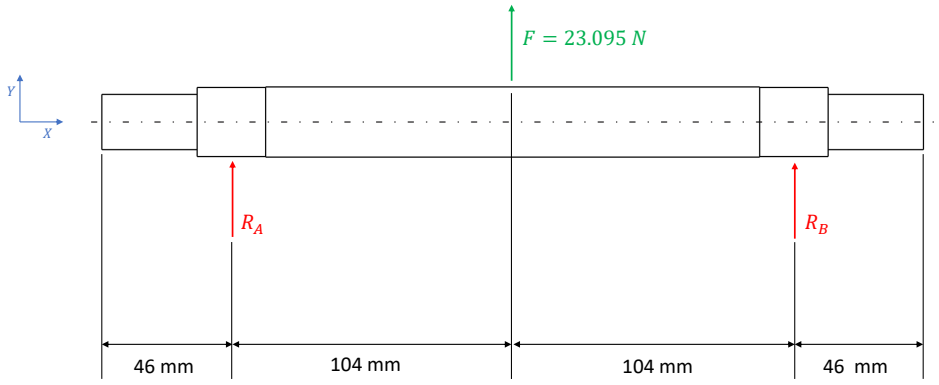


Figure 3-5.: Free-Body diagram

The reaction in support R_A is cleared, with a moment diagram in R_B:

$$\Sigma F_y = 0 = F - R_A - R_B \tag{3-5}$$

$$F = R_A + R_B \tag{3-6}$$

$$\Sigma M_{Ra} = 0 = R_B \cdot (X_{R_{A-B}}) - F \cdot (X_{R_A - F}) \tag{3-7}$$

When $X_{R_{A-B}}$ is the distance between R_A and R_B
When X_{Ra-F} is the distance between R_A and F

$$23{,}095N \cdot (0{,}104m) = R_B \cdot (0{,}208m) \tag{3-8}$$

$$\frac{568{,}492N.m}{0{,}208m} = R_B \tag{3-9}$$

$$R_B = 11{,}5475N \tag{3-10}$$

$$R_A = F - 11{,}5475N \tag{3-11}$$

$$R_A = 11{,}5475N \tag{3-12}$$

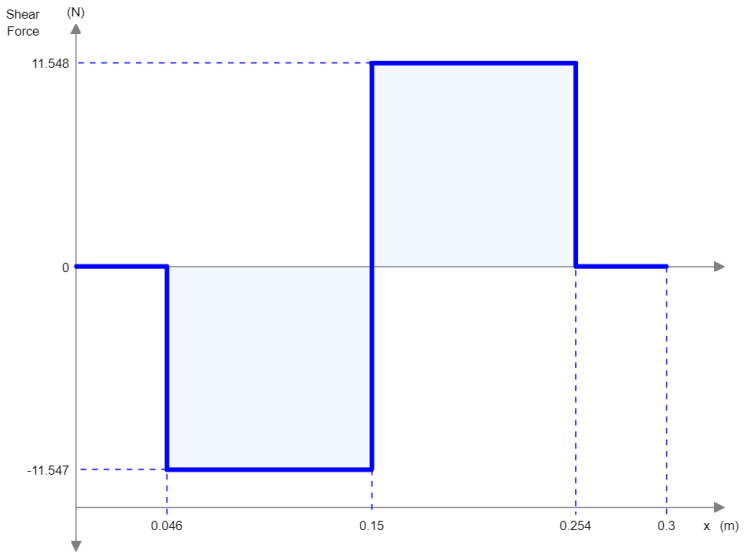


Figure 3-6.: Shear Force Diagram

Having the shear force diagram the next step is to calculate the bending moment:

$$-R_A \cdot (X_F - X_{R_A}) + M_{load} = 0 \tag{3-13}$$

$$M_{load} = 11{,}5475 \cdot (0{,}15 - 0{,}046) = 1{,}2Nm \tag{3-14}$$

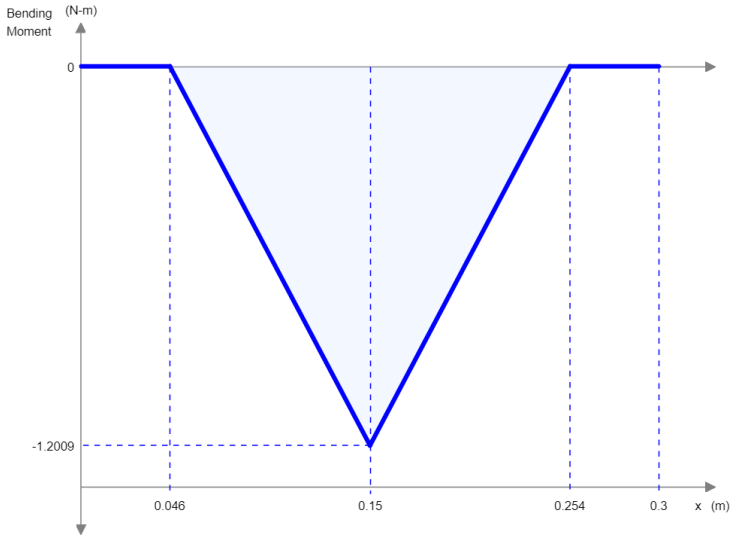


Figure 3-7.: Bending Moment Diagram

The next step would be to calculate the maximum stress, Radius of curvature, and the unitary deformation.

$$\sigma_{max} = \frac{M \cdot \frac{d}{2}}{\frac{\pi}{64} \cdot d^4} = 1{,}148Mpa \tag{3-15}$$

$$\frac{1}{\rho} = \frac{M}{E \cdot \frac{\pi}{64} \cdot d^4} = 0{,}509mm \tag{3-16}$$

$$\varepsilon = \frac{M \cdot \frac{d}{2}}{E \cdot \frac{\pi}{64} \cdot d^4} = 0{,}006mm \tag{3-17}$$

were ρ is the Radius of curvature and ε is the unitary deformation.
To obtain the security factor for the axes the following equation is used:

$$(\frac{S_y}{S_f})^2 = (\frac{M \cdot \frac{d}{2}}{\frac{\pi}{64} \cdot d^4})^2 \tag{3-18}$$

S_f is the security factor.

$$S_f = S_y \cdot (\frac{\frac{\pi}{64} \cdot d^4}{M \cdot \frac{d}{2}}) \tag{3-19}$$

Using the small diameter of the axes of 22mm, the security factor is equal to 183, which is extremely good. After making the calculations, it can be determined that it will not fail in extreme cases, making it adequate for academic use.

Squirrel Cage Breaking

To perform the reproduction of squirrel cage rupture failure, it is essential to implement an alternative rotor of the used motor to which it must be modified by making a hole in one of the bars, as shown in Figure **3-8**, this emulates the common fault that occurs in induction motors.

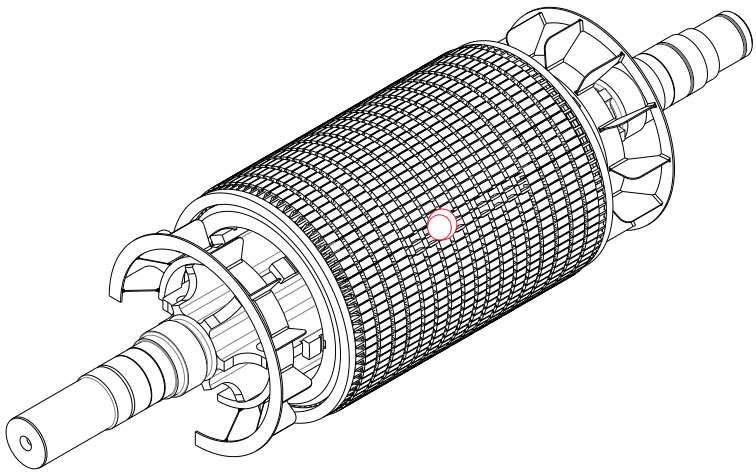


Figure 3-8.: Modified squirrel cage

Bearing

To reproduce the bearing wear fault, it is necessary to implement a faulty bearing to emulate the fault, to do this it is decided to use bearings that fulfill the function of positioning the shaft for the inertial imbalance disk, this to avoid the modification of the motors own bearing, for this a bearing is used which is modified as observed in the Figure **3-9**, altering one of the balls. It is essential to highlight how the system is designed so that it is easy to exchange the bearings positioned in their respective bearing. This can be seen in the Figure **3-10**

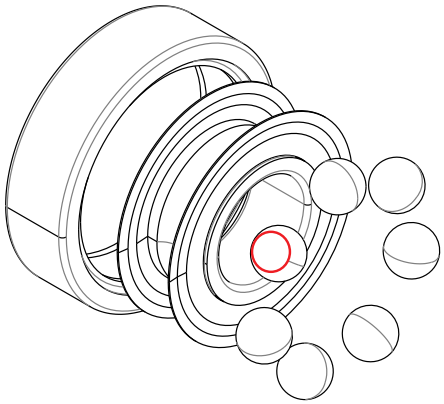


Figure 3-9.: Bearing

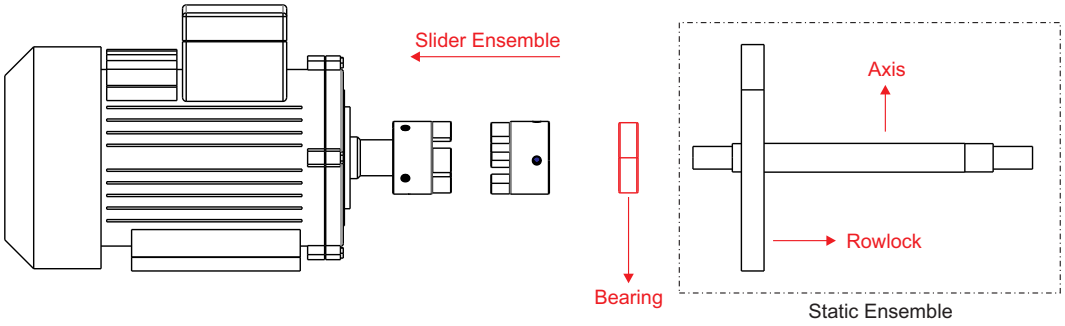


Figure 3-10.: Dismount Bearing

3.3.2. Data acquisition Sub-System

Vibrations

To perform the acquisition of vibration data it is necessary to use a sensor that measures its acceleration, that is why an accelerometer must be used which is capable of delivering high-speed measurements taking into account the nominal speed of the motor, to select the suitable sensor, the calculation presented in the Ecuación 3-20 is performed, which is based on the fact that at least 6 samples must be obtained per revolution of the motor so that the acquired data is truly functional for the test bench.

$$\frac{Hz}{Rev} = \frac{1800rpm}{60rpm} = 30Hz \tag{3-20}$$

$$180Hz = 6 \times 30Hz \tag{3-21}$$

From the previous calculation, it is determined that the sensor must be 200 Hz; for this reason, the BWT901Cl accelerometer from Wit Motion is selected, which meets the minimum requirements.

The BWT901Cl accelerometer has internal processes to output a digital signal with all the information. The signal that is output from the accelerometer is a series of hexadecimal. The application decodes this information in the following way. Every 10 hexadecimal is one sample of data, and each sample starts with the number 0x55. Another number that has a distinct meaning is the number 0x52, which indicates that the type of information is acceleration. What the program does is when it sees the number 0x55 it knows that it is a new sample and it then goes on to look for the number 0x52 that indicates that the information is acceleration and it takes the 3 following hexadecimal which are 16 bits each. These in turn, are the acceleration on the x, y, and z-axis. The program then organizes this data into vectors and plots it. This can be viewed in the appendix B.2 under the comment loop Data capture and processing.

Velocity

To perform the acquisition of velocity data, it is necessary to use a sensor that measures the position of the motor shaft in real-time. This is why an incremental encoder must be used, capable of delivering very accurate speed measurements, For this reason, a high-resolution encoder is selected, which performs measurements of 1000 pulses per revolution based on the position and the sampling time. The micro-controller can calculate the speed in real-time. The selected sensor is ES38-6G1000 encoder from CALT.

The encoder ES38-6G1000 delivers a digital signal for each pulse generated from the rotation of the axis, this signal is received by the microcontroller as shown in Figure **A-6** which performs the digital signal treatment to be able to deliver data regarding the angular velocity of the motor in $\frac{rad}{s}$, to To do this, the derivative of the angular position with respect to time is performed, to obtain a vector of speed data in $\frac{rad}{s}$, this signal treatment can be reviewed in appendix B.1 under the comment Signal Processing.

Current

It is necessary to use a sensor that measures the phase current of the motor for acquiring the data of the motors current, which is why a non-invasive sensor must be used that is capable of measuring the nominal current of the motor, the selected sensor is YHDC SCT013, which delivers an analog voltage signal relative to the current passing through the coil.656.

Acoustic

For the process of procuring the acoustic data, the use of a microphone is required. Any microphone with a USB is optimal to facilitate the interaction with the pc. The TONOR TN120072BL Conference USB Microphone was selected for its quality and price. the data is captured directly by the pc with the Matlab tool for audio.

3.3.3. HMI Sub-System

To perform the Human-Machine interface, develop applications using the Matlab software App designer tool to capture data from the respective test bench measurements in a centralized and synchronized way, for user ease in the generation of reports and analysis of the respective measurements.
A conceptual design of the logic of the data acquisition system was made using the next flowchart.

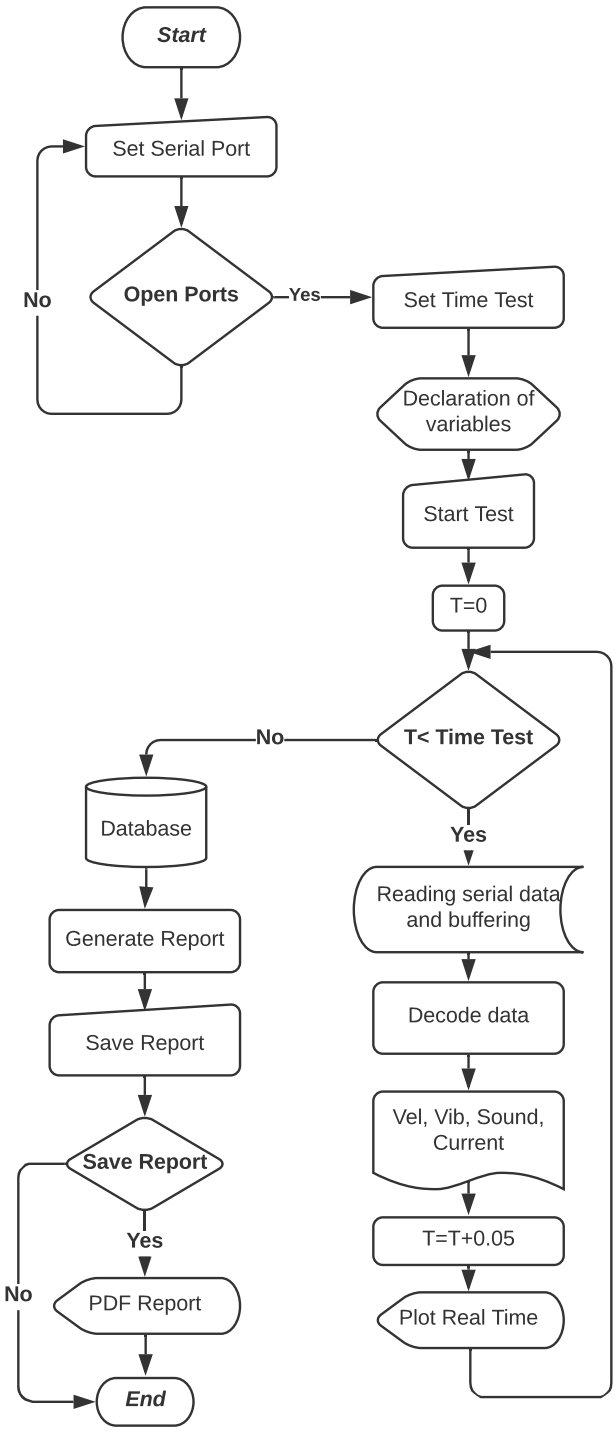


Figure 3-11.: Flowchart HMI

The logic designed in the flowchart of the Figure **3-11** is implemented with the App designer tool as shown appendix B.3, and the development of the graphical interface is carried out, taking into account the connections via the serial port. The acquisition of data in real-time, the decoding of the acquired signals and the treatment of the signals using mathematical tools such as the fast Fourier transform (FFT), for their subsequent interpretation and sample of signals from the different sensors; finally the data acquisition software developed is visualized as follows.

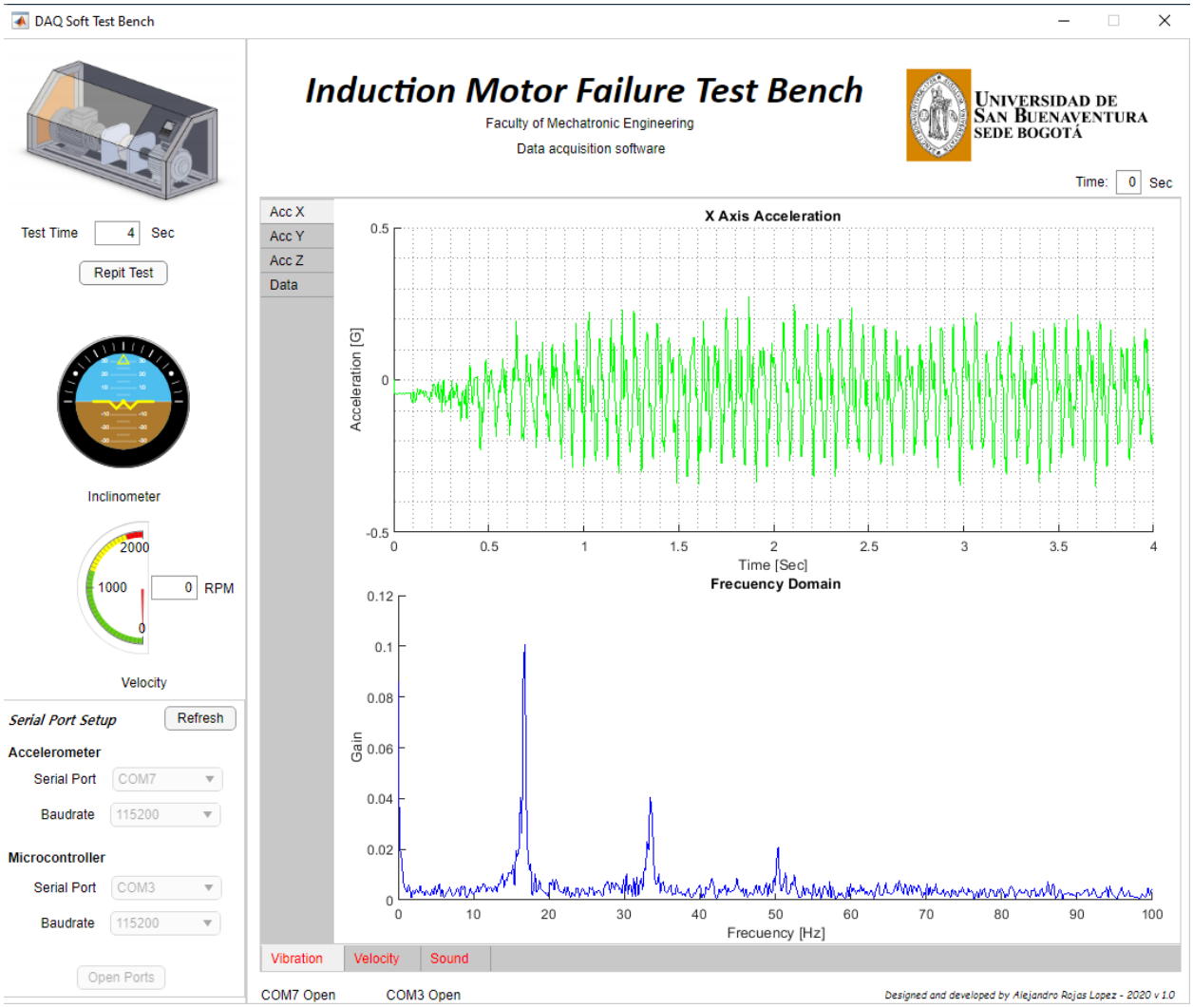


Figure 3-12.: DAQSoft Test Bench (HMI)

3.4. Implementation

To conclude this section, the entire system's implementation is carried out based on the conceptualization and development of the different sub-systems.

The test bench is manufactured from the sketch made Figure **3-13**, taking into account what is described in the Subsección 3.3.1 using the corresponding dimensions, locations, and materials.

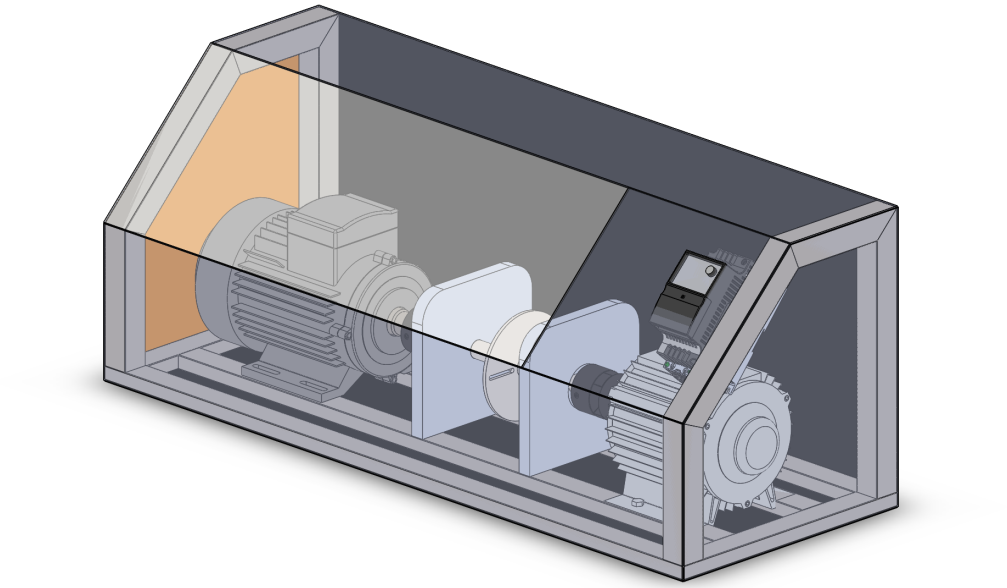


Figure 3-13.: Sketch Test bench implementation

After the manufacture of the parts and the respective assembly, the test bench looks as follows.

Figure 3-14.: Test bench implementation

The Figure **3-15** conceptually shows the implementation of the sub-systems that comprise the complete system.

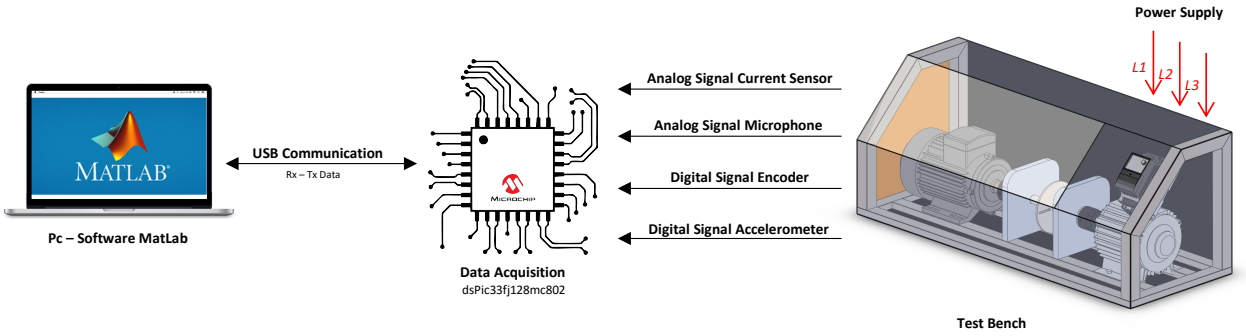


Figure 3-15.: Conceptual implementation

4. Experimental Results and Analysis

Experimental tests of data acquisition and signal treatment are carried out using a didactic prototype in which a DC motor is tested with no load and an unbalanced load, this in order to simulate the actual operation of the test bench and can check the correct operation of the data acquisition system and treatment of the different sensor signals.

For this, the developed software (DAQ Soft Test Bench) is used, and a 5-second test is performed, obtaining the following results.

4.1. Vibration

Accelerometer data acquisition is performed by storing the linear acceleration in the 3 axes (X, Y, Z), taking into account that data collection is performed with a frequency of 200 Hz. From the captured signals, the software can carry out the corresponding treatment of the signals using the mathematical tool of the fast Fourier transform (FFT), where the spectrum of the signal in the frequency domain can be observed.

4.1.1. Motor With No Load

For the realization of this experiment, the developed application was implemented, carrying out a test that lasted 5 seconds. The following linear acceleration signals were captured in each axis, and are depicted in Figure **4-1**, Figure **4-2**, and Figure **4-3** reflecting the motor vibrations.

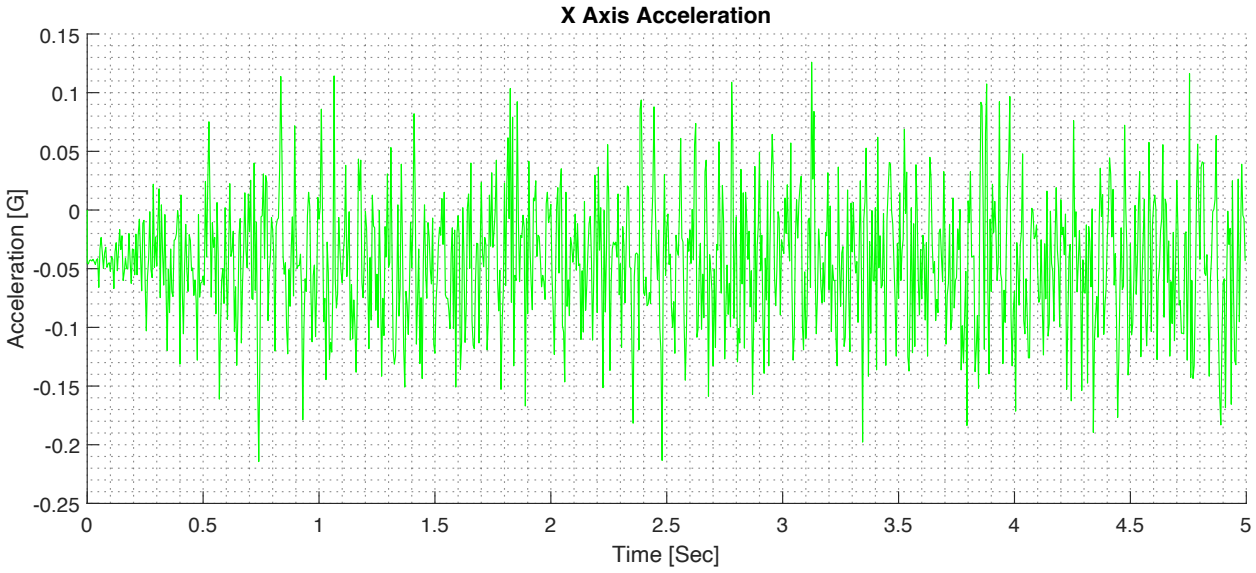


Figure 4-1.: Acceleration Axis X - Motor With No Load

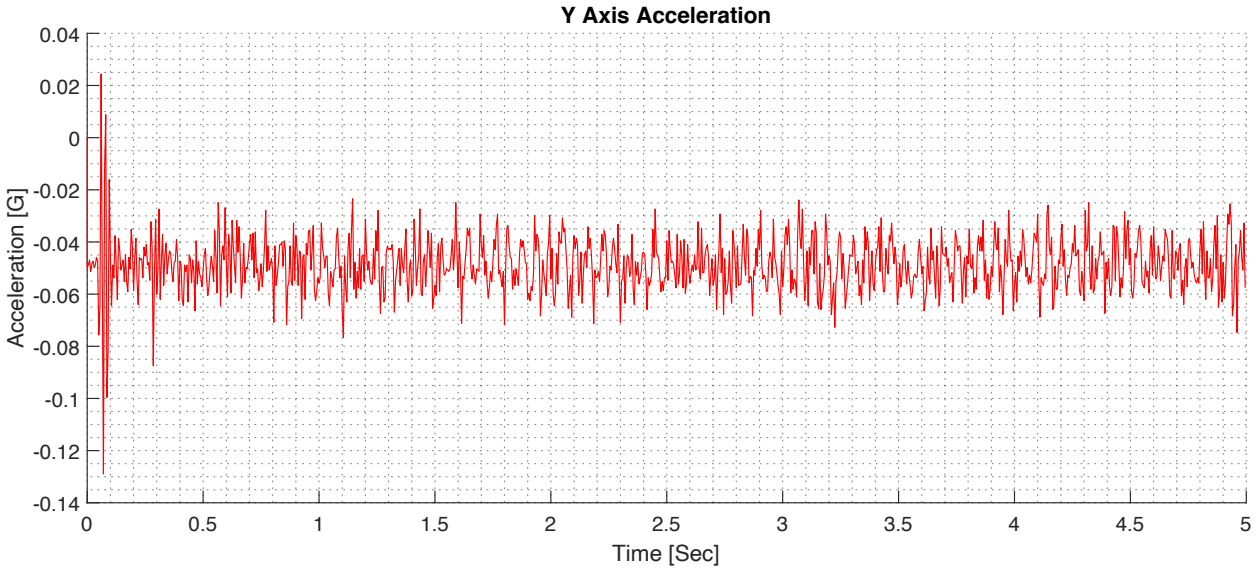


Figure 4-2.: Acceleration Axis Y - Motor With No Load

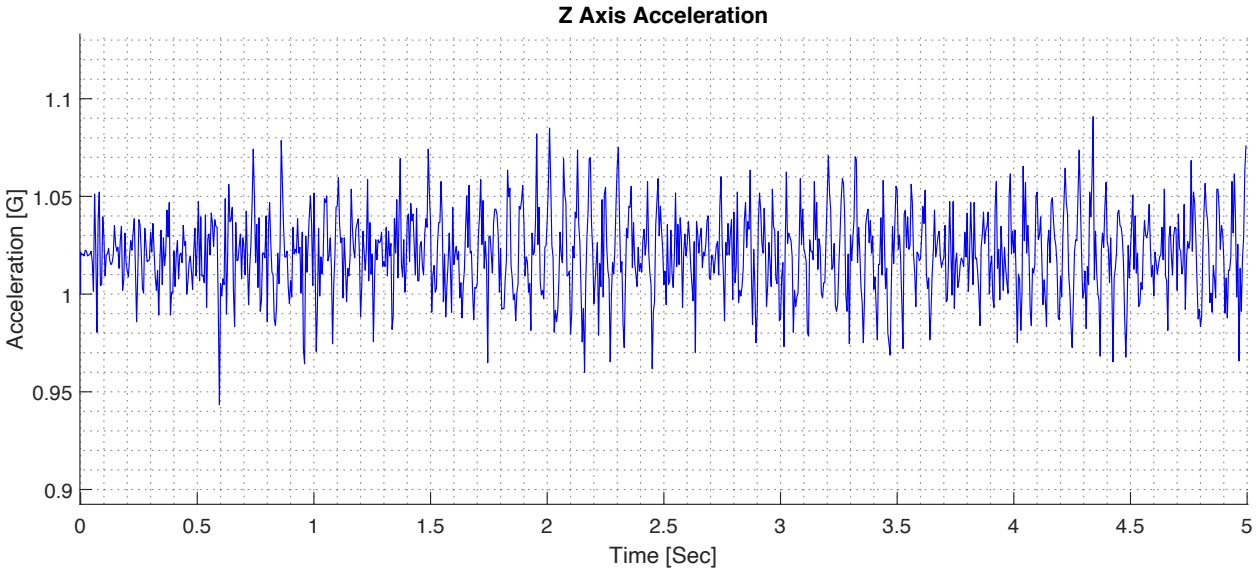


Figure 4-3.: Acceleration Axis Z - Motor With No Load

4.1.2. Motor With an Imbalance Load

For the realization of this experiment, DAQ soft was used, carrying out a test that lasted 5 seconds in which the motor has an unbalanced load coupled, this is done to reproduce a fault, the following Figure **4-4**, Figure **4-5**, and Figure **4-6** linear acceleration signals were captured on each axis which reflects vibrations in the motor.

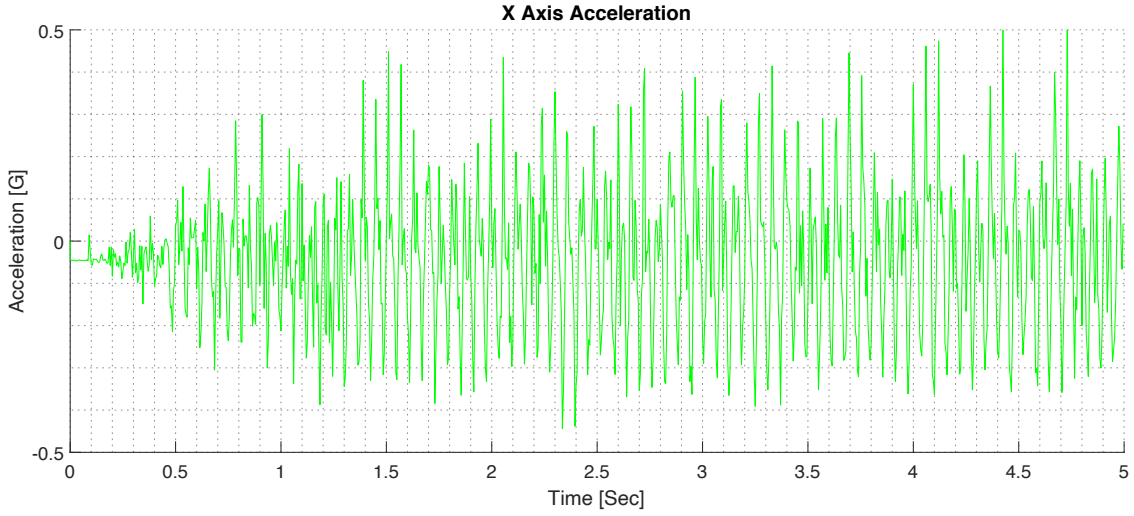


Figure 4-4.: Acceleration Axis X - Motor With an Imbalance Load

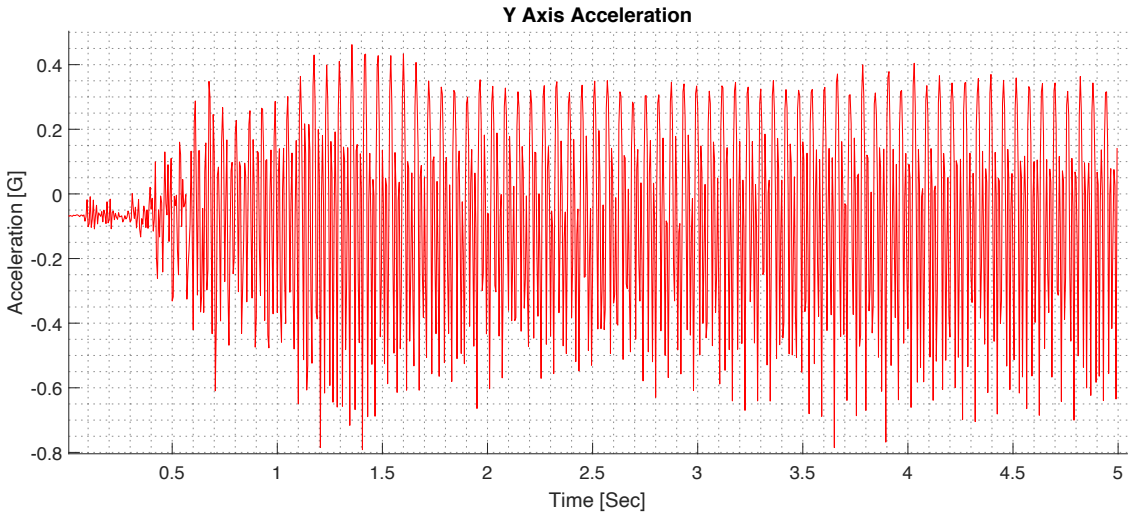


Figure 4-5.: Acceleration Axis Y - Motor With an Imbalance Load

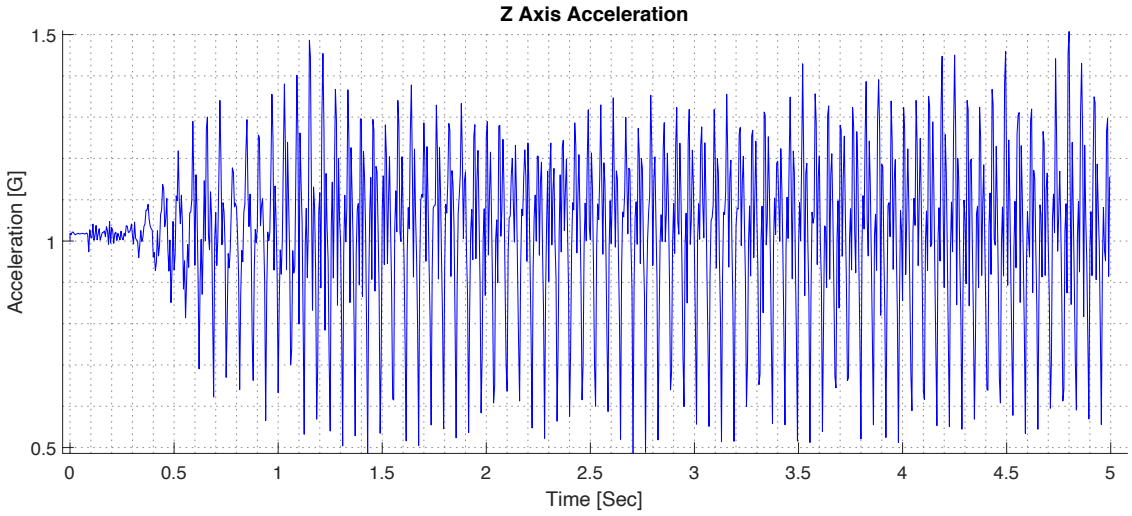


Figure 4-6.: Acceleration Axis Z - Motor With an Imbalance Load

4.1.3. Comparative Frequency Spectrum

After performing both experiments one with load and one without load, the DAQ soft (HMI) performs the respective treatment of the signals to obtain the frequency spectrum of each one, as shown in the following graphs Figure **4-7**, Figure **4-8**, and Figure **4-9**.

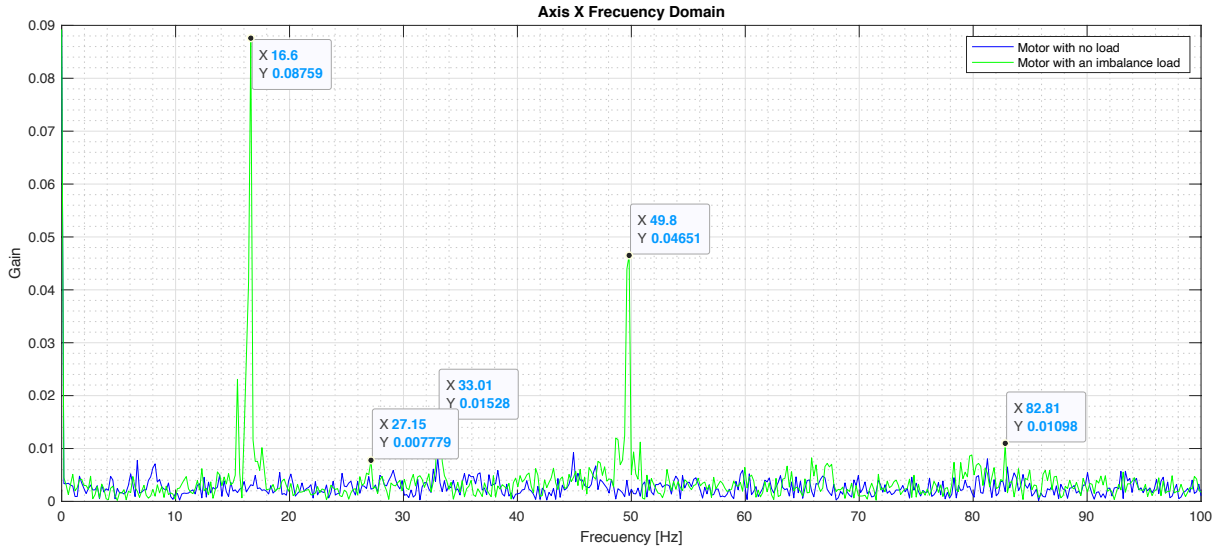


Figure 4-7.: Spectrum Axis X

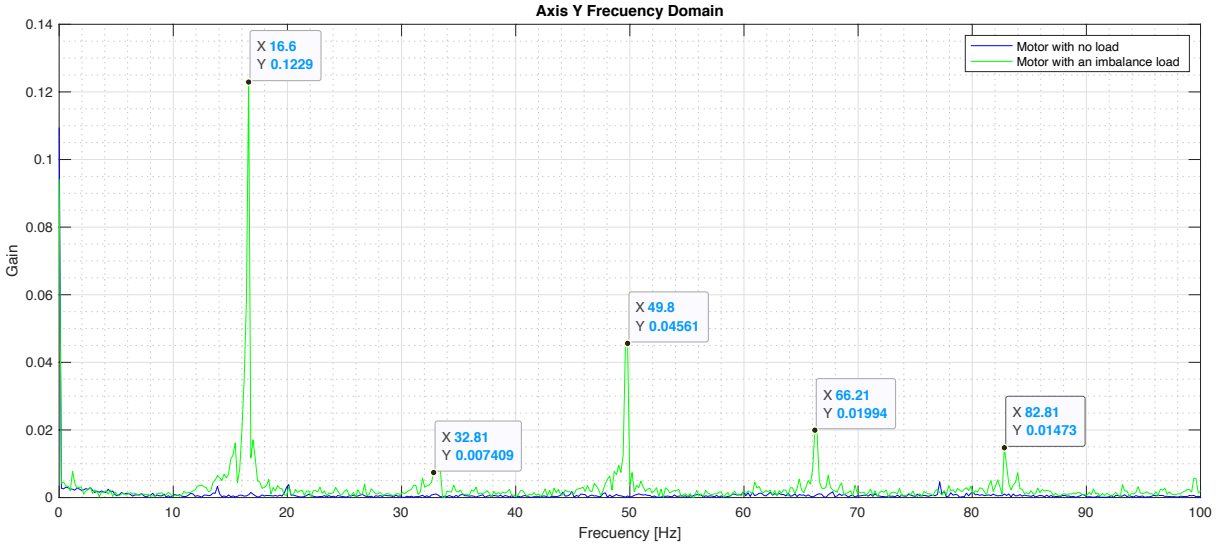


Figure 4-8.: Spectrum Axis Y

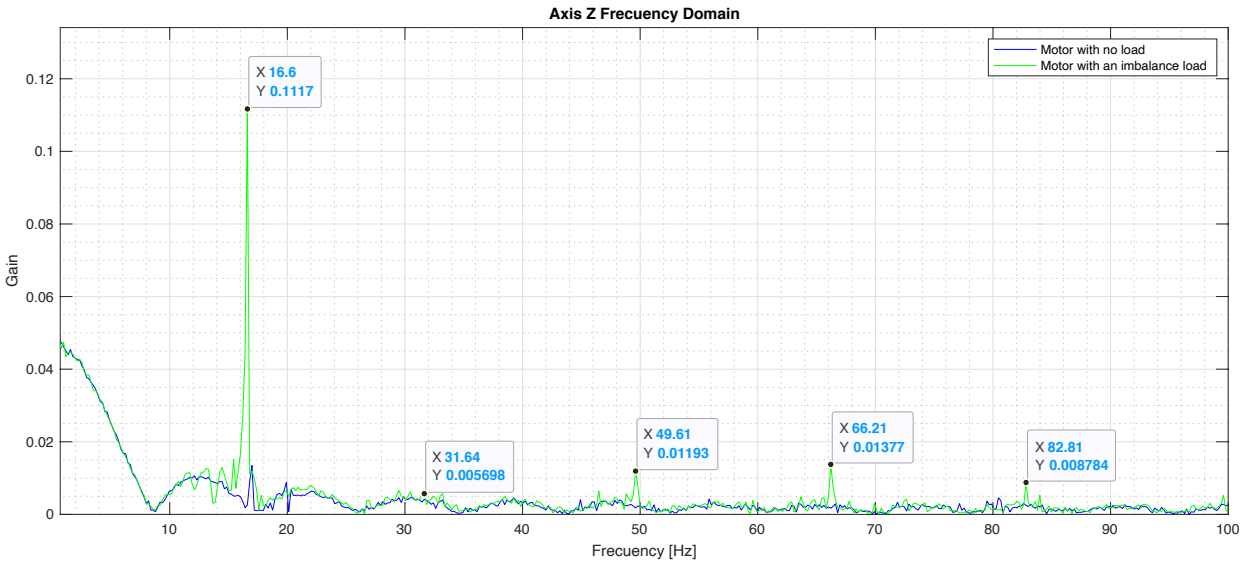


Figure 4-9.: Spectrum Axis Z

From the Figure **4-7**, Figure **4-8**, and Figure **4-9**, the frequency peaks presented by the system can be evidenced with the fault, which is the unbalanced load. When making a comparison between experiments, the difference in vibration behavior can be noticed. The frequency of the motor speed is reflected in 17 Hz and their respective harmonics. In addition to this, we see frequency peaks that can be highlighted as frequencies generated due to the unbalanced load.

4.2. Acoustic

The acquisition of audio data is performed by storing the samples captured by a microphone, taking into account that the data collection is carried out with a 44100 Hz frequency. From the captured signal, DAQ Soft is capable of performing the respective treatment of the signal using the FFT mathematical tool. The spectrum of the signal in the frequency domain can be observed.

4.2.1. Motor With No Load

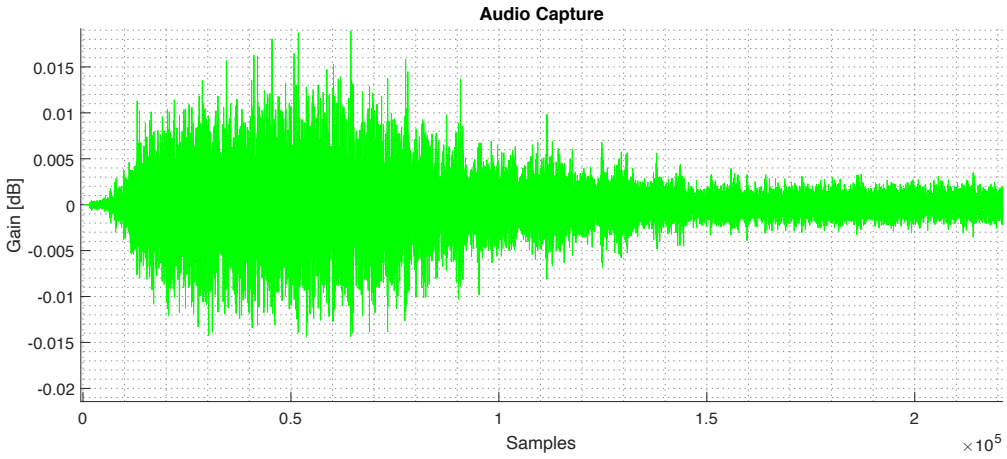


Figure 4-10.: Audio Capture - Motor With No Load

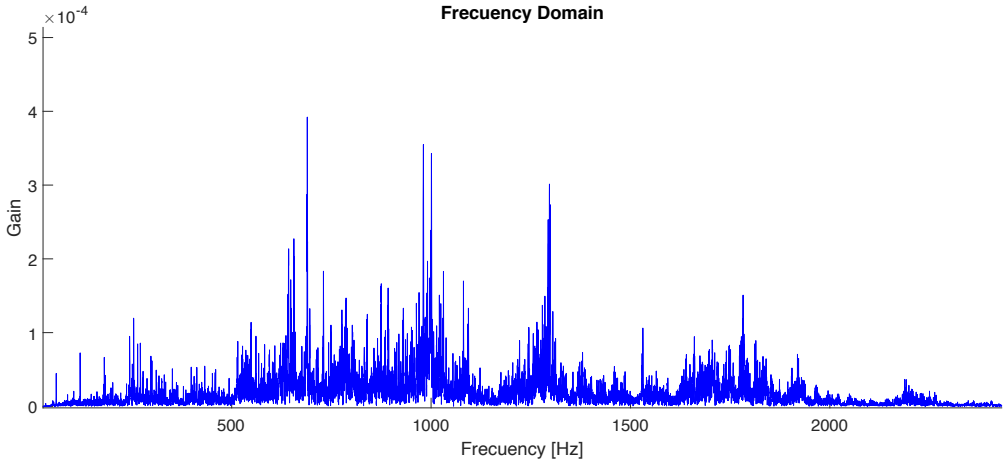


Figure 4-11.: Spectrum Audio - Motor With No Load

4.2.2. Motor With an Imbalance Load

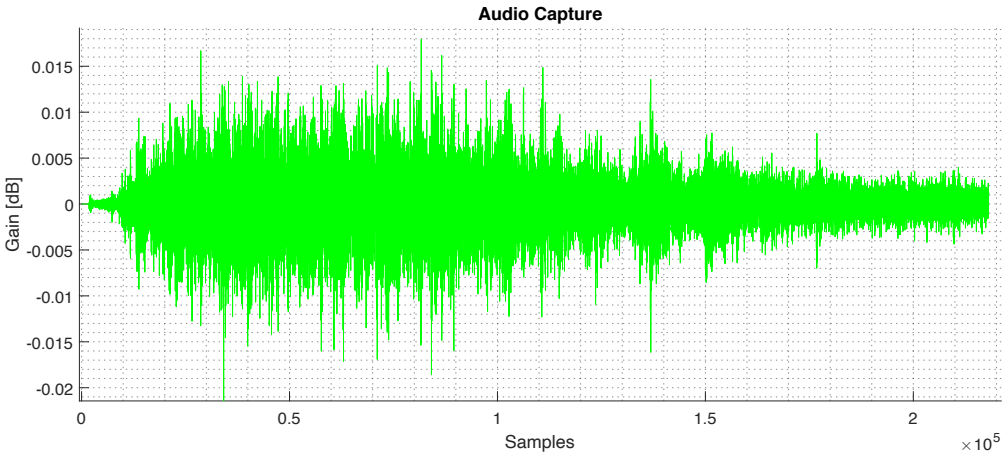


Figure 4-12.: Audio Capture - Motor With an Imbalance Load

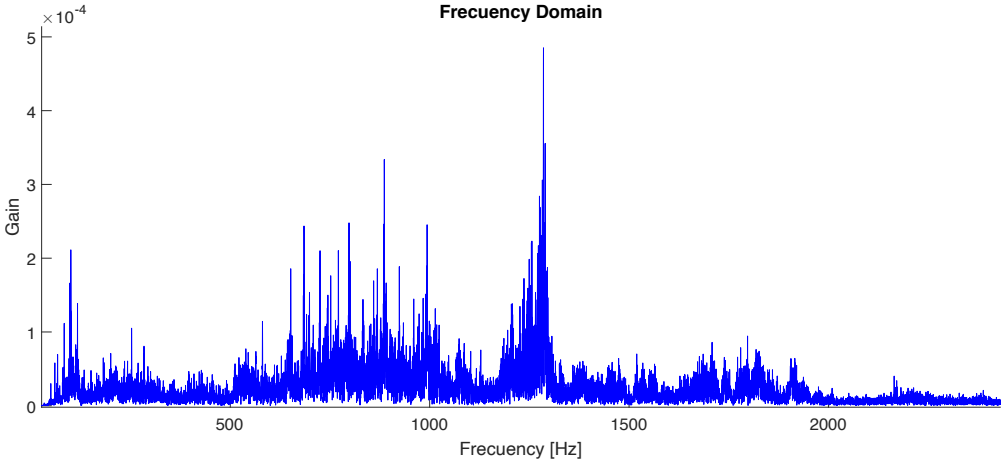


Figure 4-13.: Spectrum Audio - Motor With an Imbalance Load

Comparing Figure **4-11** and Figure **4-13**, it is determined that using audio analyses as a method for detecting faults is valid because of different frequency pecks being observed in both graphs.

4.3. Velocity

The data acquisition of the incremental encoder is carried out, storing it in real-time, taking into account that the data collection is carried out with a 200 Hz frequency. From the captured signal, DAQ Soft is capable of performing the corresponding treatment of the signal, to determine the speed from the position making the respective derivative.

4.3.1. Motor With No Load

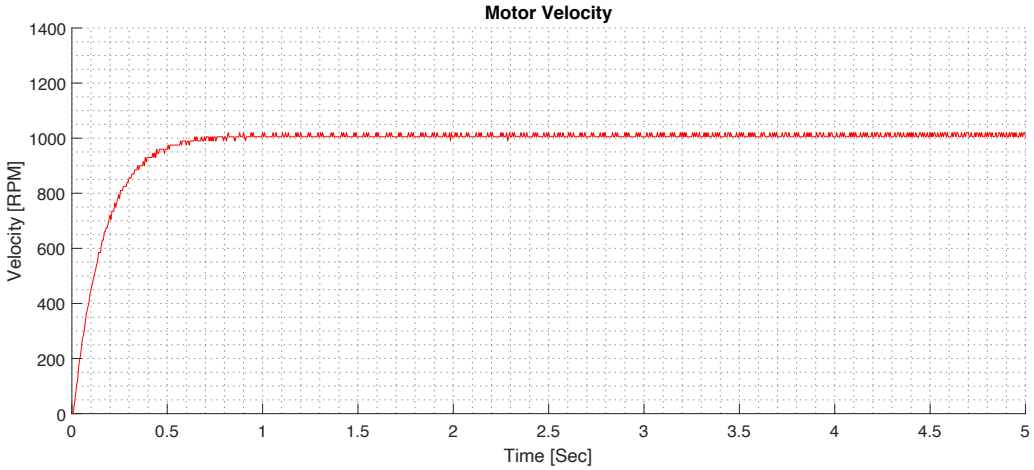


Figure 4-14.: Angular Velocity - Motor With No Load

4.3.2. Motor With an Imbalance Load

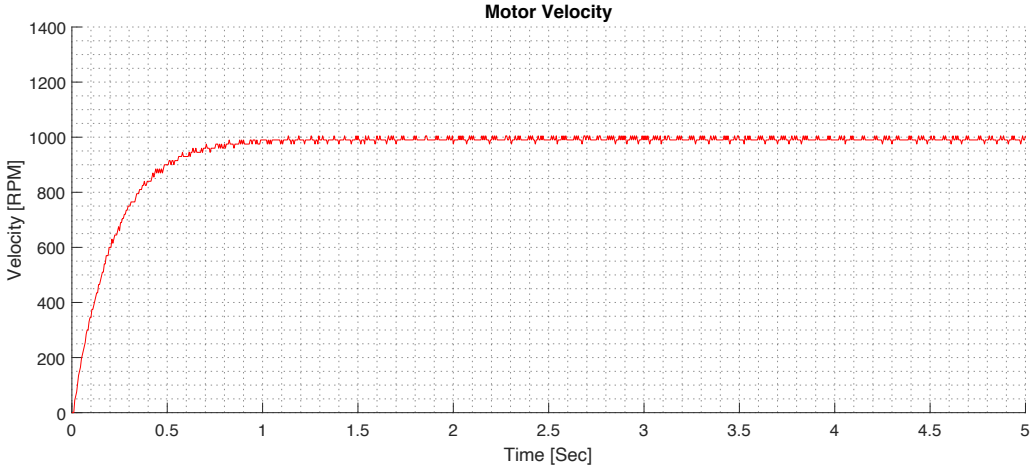


Figure 4-15.: Angular Velocity - Motor With an Imbalance Load

5. Future Work

This project was developed to contribute on future research, as an academic learning tool for future generations of the Universidad de San Buenaventura, leaving the test bench for failure analysis in rotational systems completely structured, as shown through this document. They can develop preventive maintenance programs for induction motors, develop control systems capable of mitigating failures, and develop a cloud computing system in which to store the data in real-time of the different failure analysis techniques to generate solutions on artificial intelligence. This project also provides ease for the treatment of analog and digital signals and their corresponding processing to be able to be analyzed and determine in a general way the characterization of the faults. In addition, a better development can be made in terms of the graphical interface (HMI), being able to provide the user with a much friendlier environment where they can obtain information on different provisional solutions to the failure presented, or give indications on how to mitigate the failures. At the same time, The respective repair is carried out. This project also contributes to develop programming skills under the MatLab, understanding the operation of reading data in real-time, and how this data can be used. For technological developments focused on the industrial revolution 4.0 and the IoT.

Up to this point, it can be highlighted how this project encompasses several branches of research in which the students of the Universidad de San Buenaventura can take advantage of, extending their knowledge in the different aspects previously mentioned, generating new developments and high interest in the central theme of this thesis.

6. Conclusions

- The analysis of previous research allow us to conclude that the main faults of rotational systems induction motors can be divided into stator and rotor failures, mechanical imbalance, and bearing wear. This characterization established the requirements of the test bench.
- The faults reproduction mechanism of each faults were developed under the VDI 2206 guideline. From this point the test bench was decomposed into functional subsystems and critical components were then designed. In this apart, the design the HMI facilitates the data acquisition witch captures synchronized data in real-time to treat further the signal implementing methods like FFT to display organized graphic information to the user. On the other hand, developing interchangeable mechanical parts for the test bench facilitates the reproduction of faults.
- Experiments were carried out using a didactic process. This allows to verify the correct operation of the data acquisition system and the HMI. Spectral analysis of acceleration data allows the identification of the oscillation fundamental frequency. Additionally, sound capture also shows difference for balance and unbalanced loads. Finally, under this conditions velocity monitoring did not yield to conclusive data.

Final Considerations

Comment with Approval from the Director

Director's Signature

Vo.Bo. Edwin Villarreal López
Thesis Director

Student's Signature

Alejandro Rojas López

Carlos Castellanos López

Diego Alexander Delgadillo Murillo

July 8 of 2020

A. Blueprints

A.1. Mechanical Blueprints

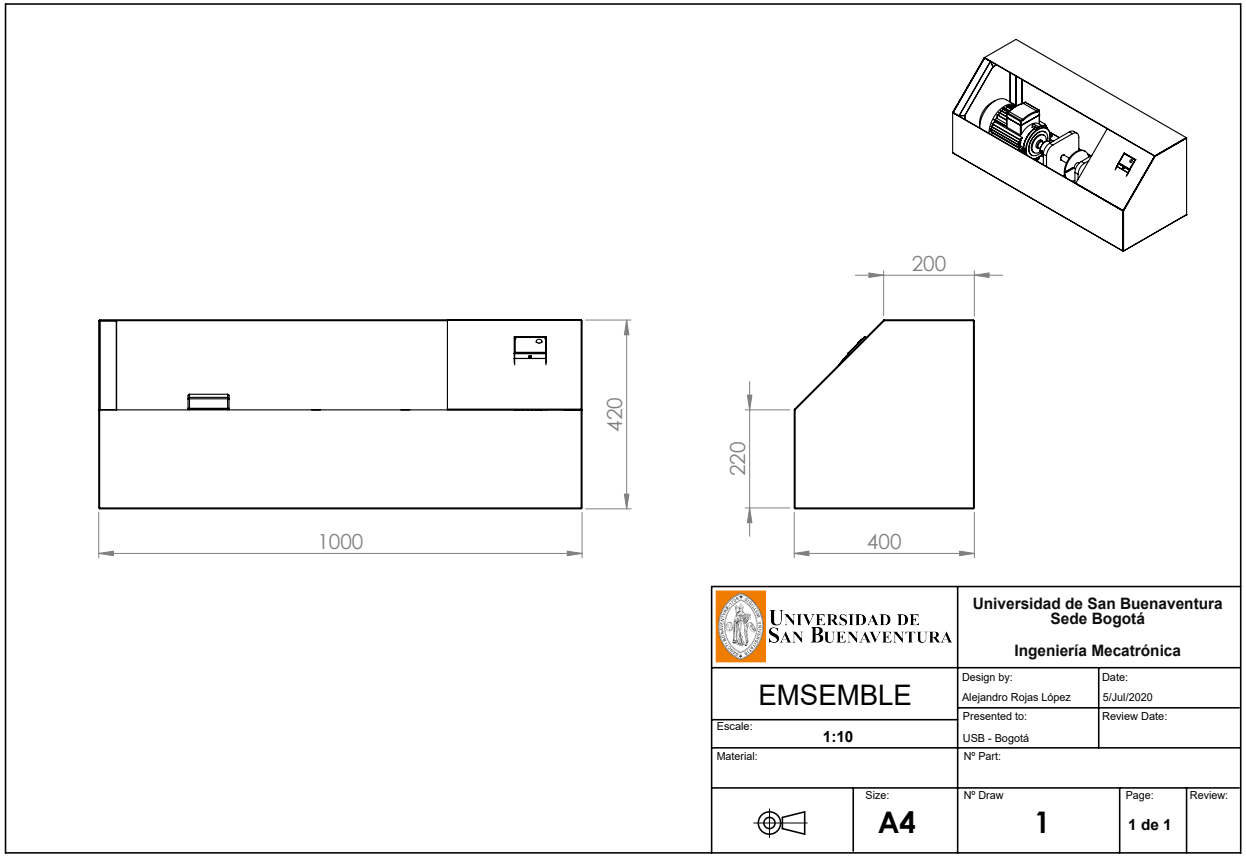
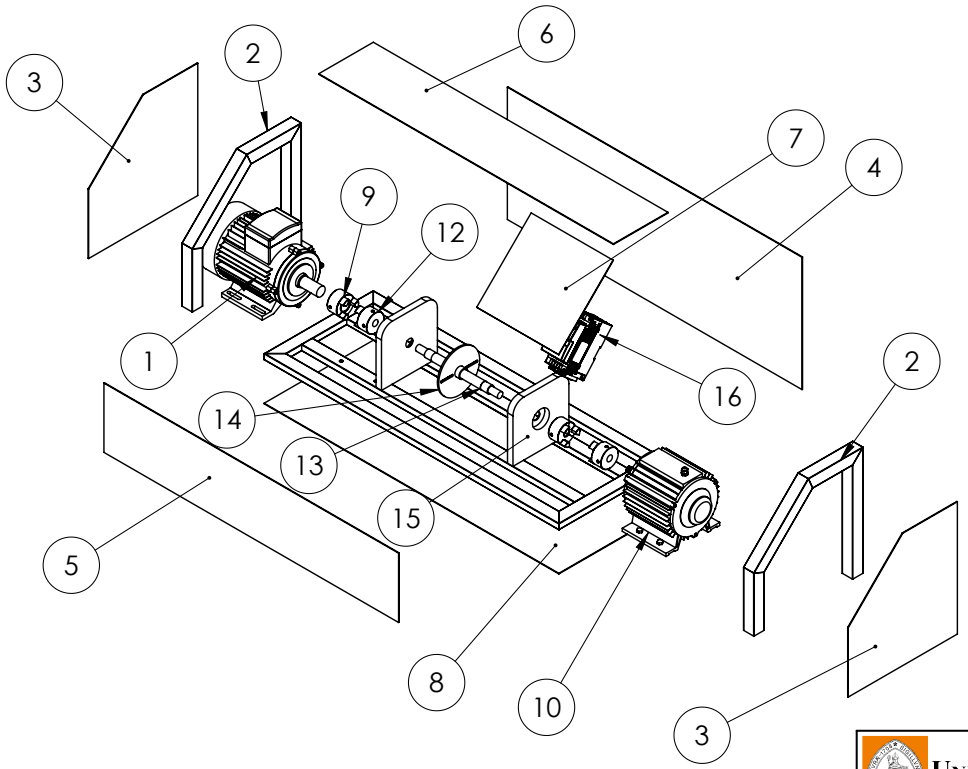


Figure A-1.: Emsemble



N.° Part	Part	Quantity
1	Induction Motor	1
2	Structural Sides	2
3	Side Case	2
4	Rear Case	1
5	Front Case	1
6	Top Case	1
7	Front Panel	1
8	Bottom Case	1
9	Motor Coupling	1
10	Generator	1
11	Generator Coupling	1
12	Axis Coupling	2
13	Axis	1
14	Inertial ImbalanceDisk	1
15	Rowlock	2
16	Frecuency Variator	1

UNIVERSIDAD DE SAN BUENAVENTURA

Universidad de San Buenaventura Sede Bogotá

Ingeniería Mecatrónica

EXPLODED VIEW

Design by: Alejandro Rojas López		Date: 5/Jul/2020	
Presented to: USB - Bogotá		Review Date:	
Escale: **1:15**			
Material:		Nº Part:	
	Size: **A4**	Nº Draw **2**	Page: **1 de 1** / Review:

Figure A-2.: Exploded View

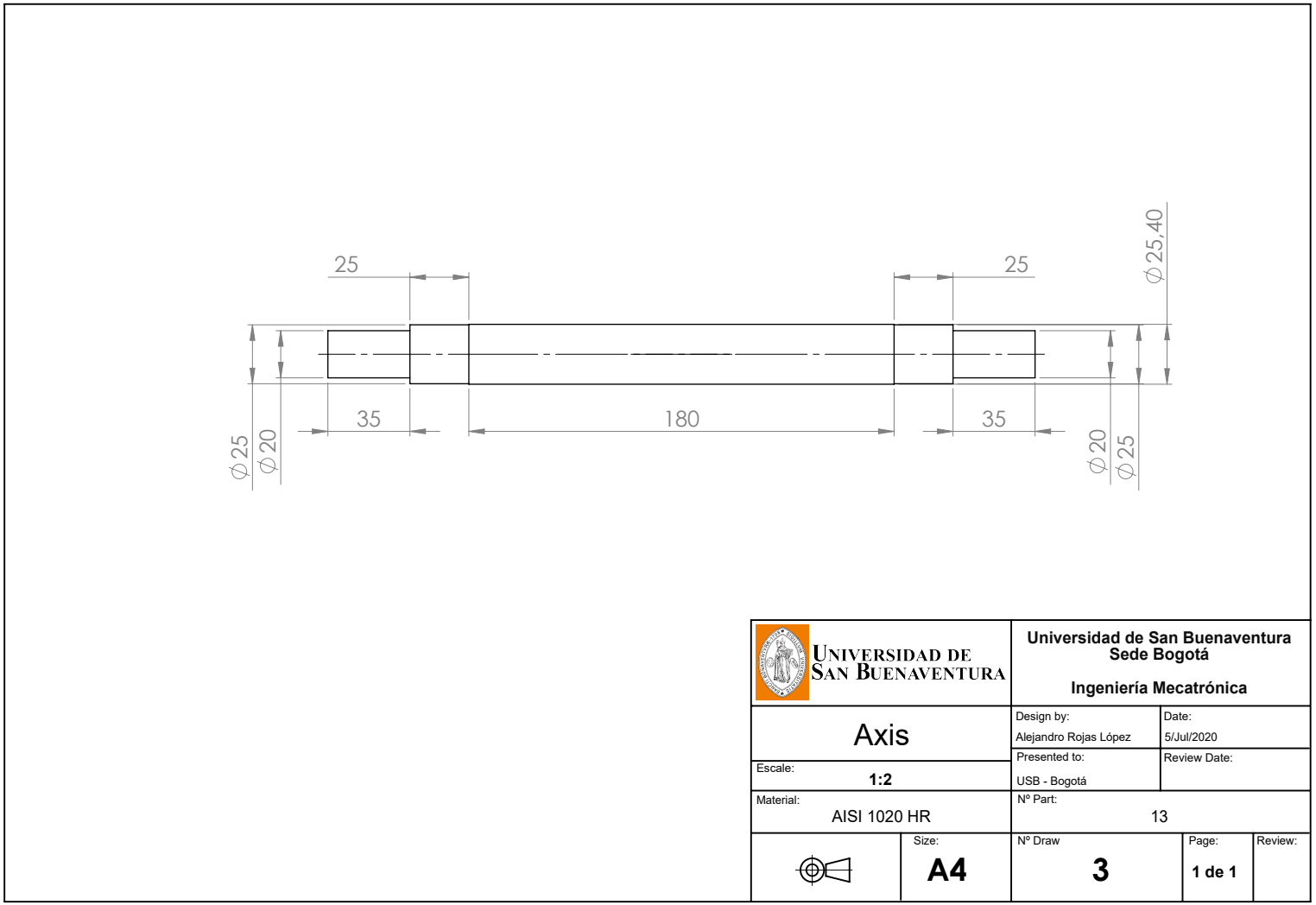


Figure A-3.: Axis

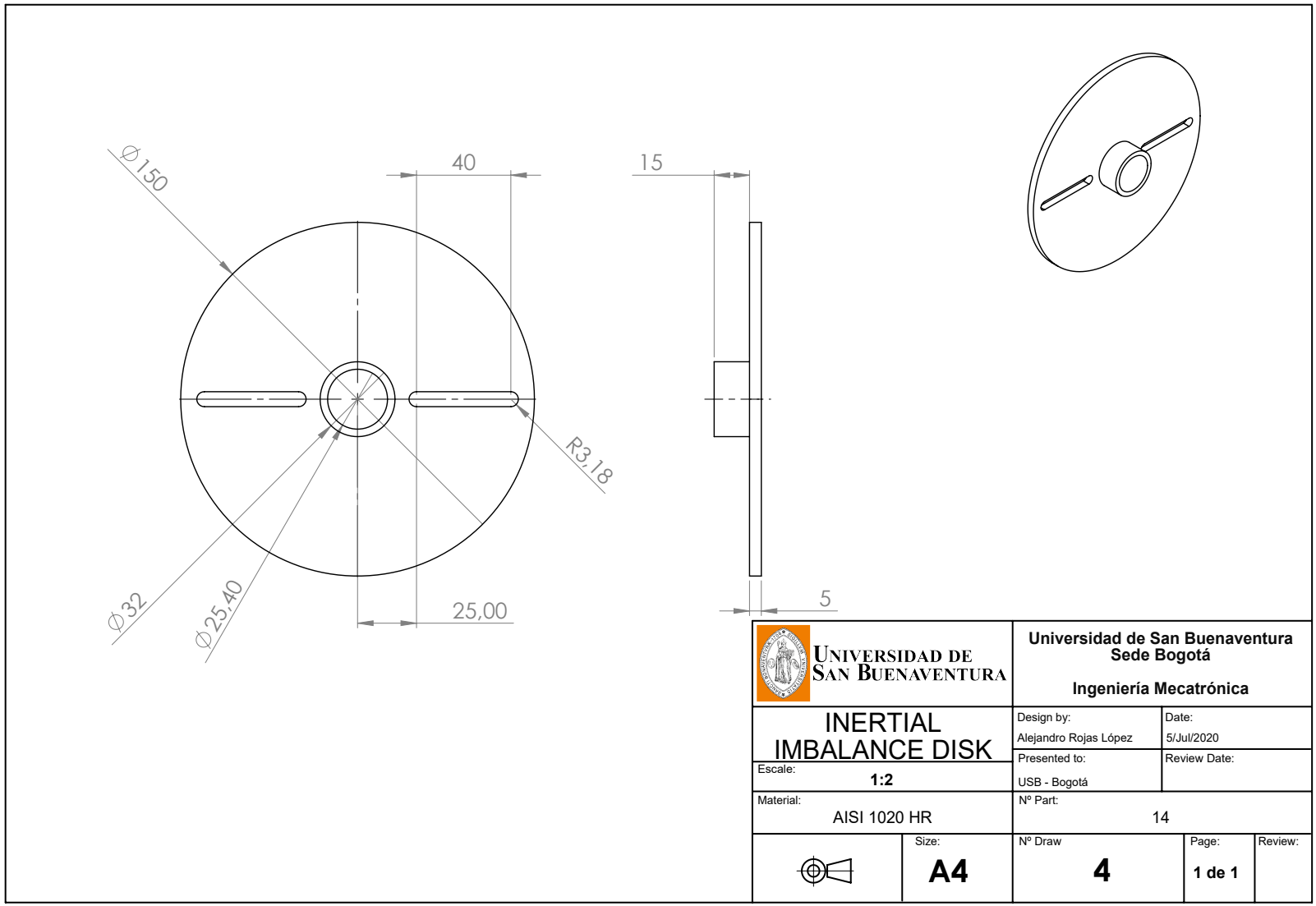


Figure A-4.: Inertial Imbalance Disk

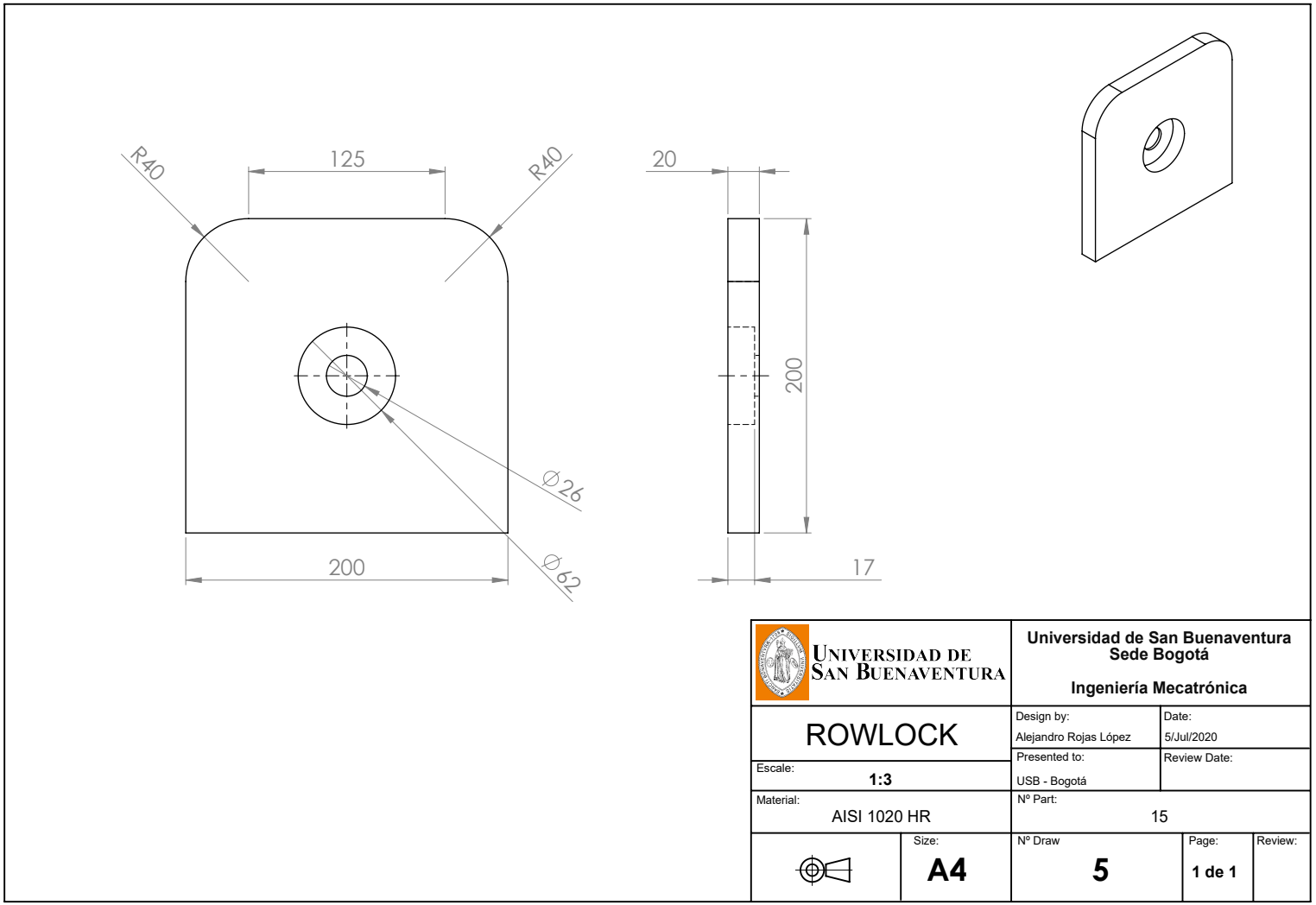


Figure A-5.: Rowlock

A.2. Embedded System Diagram

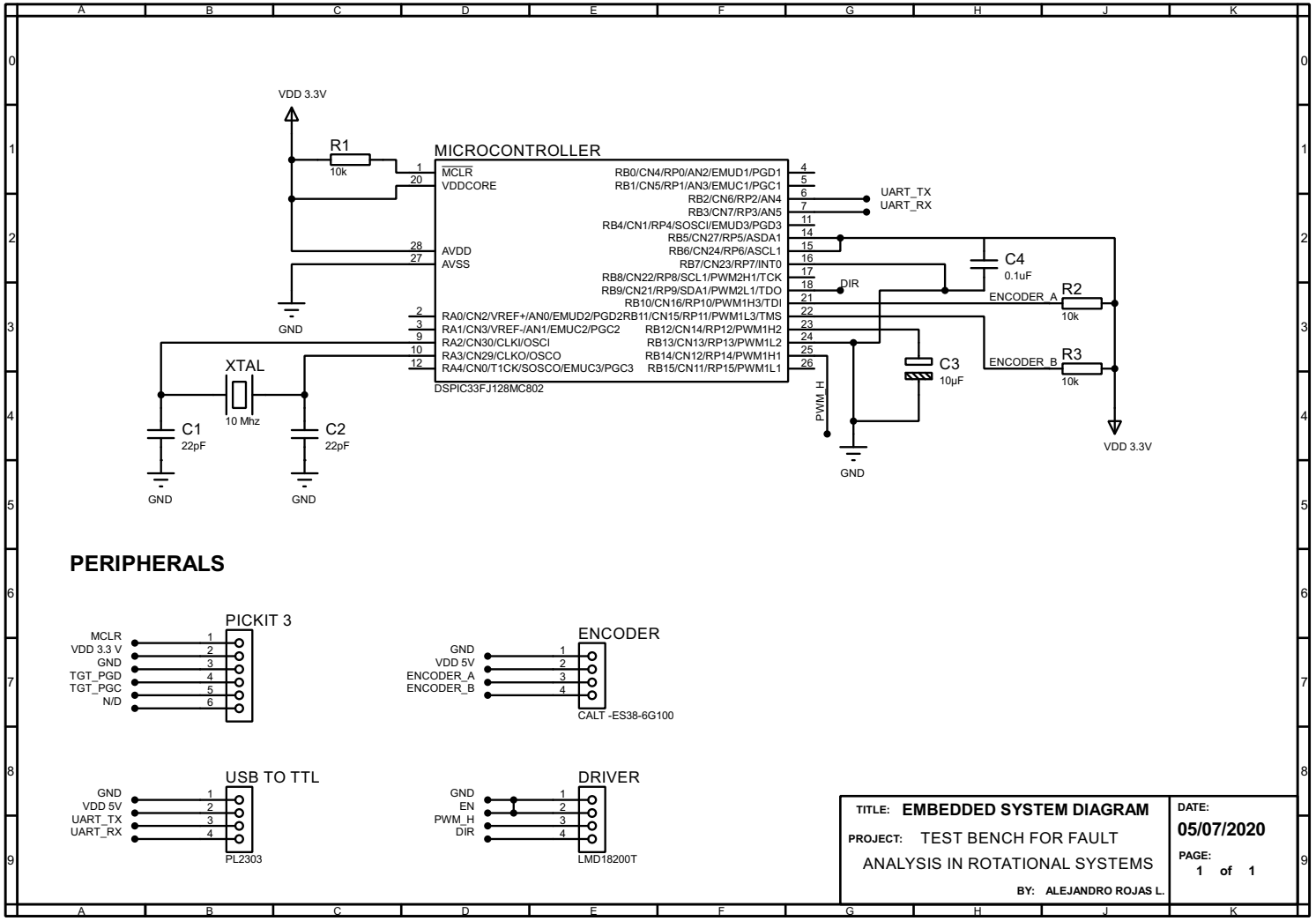


Figure A-6.: Embedded System Diagram

B. Scripts

B.1. Firmware Microcontroller (XC16)

```
#include <libpic30.h>
#include <p33FJ128MC802.h>
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <float.h>

_FOSCSEL(FNOSC_FRC & IESO_OFF); // Internal FRC start-up without PLL,
// no Two Speed Start-up
_FOSC(FCKSM_CSECMD & OSCIOFNC_OFF & POSCMD_XT); // Clock switch enabled,
// Primarly Oscillator XT
_FWDT(FWDTEN_OFF); // Watchdog Timer disabled
_FPOR(FPWRT_PWR128); // Power-up Timer enabled 128 ms
_FICD(JTAGEN_OFF); // Disable JTAG

// Function prototypes
void configure_pins();
void __attribute__((__interrupt__, no_auto_psv)) _T1Interrupt(void);

int  tempRX, pwm, i=0, posm;
unsigned int timePeriod;
double Tm=0.005,r,time=0.0,refs,radsm,radsma,posma,angm,radm,velm;

void InitClock() {
// Configure PLL prescaler, PLL postscaler, PLL divisor: cristal externo...
    10MHz
PLLFBD=30; // M = 32
CLKDIVbits.PLLPOST=0; // N2 = 2
CLKDIVbits.PLLPRE=0; // N1 = 2
// Initiate Clock Switch to Primary Oscillator with PLL (NOSC = 0b011)
__builtin_write_OSCCONH(0x03);
__builtin_write_OSCCONL(OSCCON | 0x01);
```

```
32 // Wait for Clock switch to occur
33 while (OSCCONbits.COSC != 0b011);
34 // Wait for PLL to lock
35 while(OSCCONbits.LOCK!=1) {};
36 }
37
38 // Capture Interrupt Service Routine
39 void __attribute__((__interrupt__,auto_psv)) _IC1Interrupt(void)
40 {
41 unsigned int t1,t2;
42 t1=IC1BUF;
43 t2=IC1BUF;
44 IFS0bits.IC1IF=0;
45 }
46
47                       // INPUT DATA - UART
48
49 void __attribute__((__interrupt__,auto_psv)) _U2RXInterrupt(void)
50 {
51     _U2RXIF = 0;
52     tempRX = U2RXREG;
53
54     switch (tempRX) {
55
56         case 52 : r=3000;
57            break;
58         default : r=0;
59            break;
60      }
61 }
62
63 int main(void)
64 {
65   InitClock();
66   configure_pins();
67   __C30_UART=2;
68
69   while (1)
70   {
71       __delay32(200000);
72   }
73     return 0;
74 }
75
76 // Timer 1 interrupt service routine
77 void __attribute__((__interrupt__, no_auto_psv)) _T1Interrupt(void)
```

```
78 {
79     IFS0bits.T1IF = 0;       // Clear Timer 1 interrupt flag
80     LATBbits.LATB6 = ¬LATBbits.LATB6; //Monitor LED
81
82
83    //Signal Processing
84
85     posm=POS1CNT; //Read Encoder (Pulses)
86     angm=(posm*360.0)/800.0; //Angular Position (Grades)
87     radm=angm*(3.1416/180.0);
88     velm=((posm-posma)/Tm)/800.0; //Derivative of the position (vel=R/s)
89     radsm=velm*((2.0*3.1416)); //Velocity in Rad/s
90
91
92                     // useful cycle allocation
93         if (r≥0) {
94         pwm=r;
95         P1DC1 = pwm;
96         LATBbits.LATB9 = 1;
97         }
98         if (r<0) {
99         pwm=r;
100        P1DC1 = -pwm;
101        LATBbits.LATB9 = 0;
102        }
103
104                // Variable overflow limiter Radsm
105        if (radsm<-1000.0){radsm=radsma;}
106
107                // variable update
108    posma=posm;
109    radsma=radsm;
110
111                // Time and Printf initialization
112        if (r>0) {
113        time=time+Tm;
114        printf("%2.4f,%2.4f\r\n",time,radsm);
115        }
116
117 }
118
119 void configure_pins()
120 {
121     //Configure Pins as Analog or Digital
122     AD1PCFGL = 48;
123
```

```
124     // Configure Digitals I/O directions
125     TRISB = 0xFFFF;
126
127     // Configure Remappables Pins
128     __builtin_write_OSCCONL(OSCCON & ¬(1<<6));
129     RPINR14 = 0x0B0A; // ENCODER MOTOR
130     RPINR16 = 0x0706; // ENCODER PENDULO
131     RPINR19 = 0x02;
132     RPOR1 = 0x0500;
133     RPINR7bits.IC1R=5;
134     __builtin_write_OSCCONL(OSCCON | (1<<6));
135
136     // Setup Encoder
137     QEI1CONbits.QEIM = 0; // Disable QEI Module
138     QEI1CONbits.CNTERR = 0; // Clear any count errors
139     QEI1CONbits.QEISIDL = 0; // Continue operation during sleep
140     QEI1CONbits.SWPAB = 0; // QEA and QEB not swapped
141     QEI1CONbits.PCDOUT = 0; // Normal I/O pin operation
142     DFLT1CONbits.CEID = 1; // Count error interrupts disabled
143     DFLT1CONbits.QEOUT = 0; // Digital filters output disabled for QEn ...
        pins
144     POS1CNT = 0; // Reset position counter
145     QEI1CONbits.QEIM = 7; // (x4 mode) with position counter reset by ...
        match (MAXxCNT)
146     MAX1CNT=0xFFFF;
147
148     // Setup UART
149     U2MODEbits.STSEL = 0;
150     U2MODEbits.PDSEL = 0;
151     U2MODEbits.ABAUD = 0;
152     U2MODEbits.BRGH = 1;
153     U2BRG = 85;              // 38461.53846153846 baud, error=0.16%
154     _U2RXIE = 1;               // Enable Interrupt
155     U2STA = 0x2400;
156     U2MODEbits.UARTEN=1;
157     U2STAbits.UTXEN=1;
158
159     // Configure Timer 1: Interrupci?n cada 20 mseg
160     T1CON = 0;               // Timer reset
161     IFS0bits.T1IF = 0;       // Reset Timer1 interrupt flag
162     IPC0bits.T1IP = 1;       // Timer1 Interrupt priority level=4
163     IEC0bits.T1IE = 1;       // Enable Timer1 interrupt
164     TMR1 = 0;                // Reset Timer 1 counter
165     PR1 = 3125;             // Set the Timer 1 period (max 65535)
166                              // PR1=(Periodo en seg)*Fcy/Prescaler
167     T1CONbits.TCKPS = 2;   // Prescaler (0=1:1, 1=1:8, 2=1:64, 3=1:256)
```

```
168     T1CONbits.TON = 1;      // Turn on Timer 1
169
170     // Configure Timer 2: Para InputCapture1
171     T2CON = 0;              // Timer reset
172     IFS0bits.T2IF = 0;      // Reset Timer2 interrupt flag
173     TMR2 = 0;               // Reset Timer 2 counter
174     PR2 = 64000;            // periodo m?ximo en ticks: (max 65535)
175
176     T2CONbits.TCKPS = 1;    // Prescaler (0=1:1, 1=1:8, 2=1:64, 3=1:256)
177
178     // Configure PWM
179     P1TCONbits.PTMOD = 0b00;    //for free running mode
180     P1TCONbits.PTCKPS = 0b00;   //prescale=1:1
181     P1TCONbits.PTOPS = 0b00;   // PWM time base input clock period is ...
        T_CY
182     P1TPER = 2047;      // 19531.25 Hz y 12bits para 100% pwm en P1DC1
183     PWM1CON1bits.PMOD1 = 0;
184     PWM1CON1bits.PMOD2 = 0;
185     PWM1CON1bits.PMOD3 = 0;
186     PWM1CON1bits.PEN1H = 1;
187     PWM1CON1bits.PEN2H = 0;
188     PWM1CON1bits.PEN3H = 0;
189     PWM1CON1bits.PEN1L = 1;
190     PWM1CON1bits.PEN2L = 0;
191     PWM1CON1bits.PEN3L = 0;
192     PWM1CON2bits.IUE = 1;
193     P1OVDCONbits.POVD3H = 0;
194     P1OVDCONbits.POVD2H = 0;
195     P1OVDCONbits.POVD1H = 1;
196     P1OVDCONbits.POVD3L = 0;
197     P1OVDCONbits.POVD2L = 0;
198     P1OVDCONbits.POVD1L = 1;
199   P1DC1 = 0;          // 0% duty cycle: 12bits -> 100% pwm (max is 4095)
200   P1TMR = 0;          // Clear 15-bit PWM timer counter
201   P1TCONbits.PTEN = 1;        // Enable PWM time base
202
203     // Initialize the Input Capture Module Encoder
204     IC1CONbits.ICM = 0b00; // Disable Input Capture 1 module
205     IC1CONbits.ICTMR = 1; // Select Timer2 as the IC1 Time base
206     IC1CONbits.ICI = 0b01; // Interrupt on every second capture event
207     IC1CONbits.ICM = 0b011; // Generate capture event on every Rising ...
        edge
208
209     // Enable Capture Interrupt And Timer2 Encoder
210     IPC0bits.IC1IP = 2; // Setup IC1 interrupt priority level
211     IFS0bits.IC1IF = 0; // Clear IC1 Interrupt Status Flag
```

```
212     IEC0bits.IC1IE = 1; // Enable IC1 interrupt
213     T2CONbits.TON = 1;     // Turn on Timer 2
214
215
216 }
```

B.2. Data acquisition and signal processing (MatLab)

```
1  clear all;  %Delete Variables
2  close all;  %Close Windows and Open Processes
3  instrreset; %Reset Open peripherals
4  delete(instrfindall);    % Reset Com Port
5  delete(timerfindall);    % Delete Timers
6
7  % Serial Accelerometer Configuration
8      sa=serial('COM7');
9      sa.Baudrate=115200;
10     sa.StopBits=1;
11     sa.Parity='none';
12     sa.FlowControl='none';
13     sa.InputBufferSize=999999;
14     fopen(sa);
15
16 % Serial PIC Configuration
17     sp=serial('COM3');
18     sp.Baudrate=115200;
19     sp.StopBits=1;
20     sp.Parity='none';
21     sp.FlowControl='none';
22     sp.InputBufferSize=999999;
23     fopen(sp);
24
25 %Declaration of variables
26     Var1=[0 0 0 0 0 0 0 0 0 0 0];
27     VelD=[0 0];
28     aa=[0 0 1];
29     t=0;
30     time=0;
31     tm=0.005;  %Sampling time
32
33 %Execution Audio capture
34
```

```
35      record(audio); %Start taking audio
36
37  fwrite(sp,52,'char') % Start motor turn
38  tic % Initial mark Execution time
39
40  % Loop Data capture and processing
41  while (t<10)
42
43     Head = fread(sa,2,'uint8'); %Reading data header
44      if (Head(1)≠uint8(85)) %Header location condition 0x55
45          continue;
46      end
47
48       switch(Head(2)) %Selection of data vector subscript
49
50           case 81 %subscript 0x52 "Acceleration" [G]
51
52          a = fread(sa,3,'int16')/32768*16 ; %Data storage and decoding
53          a = a'; %Transpose vector
54              aa = [aa;a(:,1) a(:,2) a(:,3)]; %Data vector creation
55              %t = t+tm %Time variable increment
56
57
58               [t Vel] = strread(fscanf(sp,'%s'), '%f%f', 1, 'delimiter', ...
                     ',');
59               Vel=Vel*(30/pi); %Conversion of Rad/s to RPM
60               VelD = [VelD;[t Vel]];
61               t=t
62               time = [time;t']; %Creation of data vector "Time"
63          end
64
65      Cap = [time aa]; %Creation full data vector Vibration
66      End = fread(sa,3,'uint8'); %Reading and storing additional data in ...
           the chain
67     toc
68
69  end
70
71  stop(audio); %Stops audio sampling
72  au=getaudiodata(audio); %audio data creation vector
73
74
75  fwrite(sp,48,'char') % Start motor turn
76  fclose(sa) % Close Accelerometer Port
77  fclose(sp) % Close PIC Port
78
```

```
79
80 % Vibration Plot
81 figure (1)
82
83     %X Acceleration Plot
84     subplot(3,2,1)
85         plot(time,aa(:,1))
86         title('Aceleracion X')
87
88     %Y Acceleration Plot
89     subplot(3,2,3)
90         plot(time,aa(:,2))
91         title('Aceleracion Y')
92
93
94     %Z Acceleration Plot
95     subplot(3,2,5)
96         plot(time,aa(:,3))
97         title('Aceleracion Z')
98
99     % Plot and Fourier Transform Calculation
100
101    %Plot y FFT X
102    subplot(3,2,2)
103        ftx=fft(aa(:,1));    %Transformada Rapida de Fourier Aceleracion X
104        ftx_mag=abs(ftx);    %Creacion vector de magnitud
105        plot(ftx_mag)
106        title('Espectro X')
107
108    %Plot y FFT Y
109    subplot(3,2,4)
110        fty=fft(aa(:,2));    %Transformada Rapida de Fourier Aceleracion Y
111        fty_mag=abs(fty);    %Creacion vector de magnitud
112        plot(fty_mag)
113        title('Espectro Y')
114
115    %Plot y FFT Z
116    subplot(3,2,6)
117        ftz=fft(aa(:,3));    %Transformada Rapida de Fourier Aceleracion Z
118        ftz_mag=abs(ftz);    %Creacion vector de magnitud
119        plot(ftz_mag)
120        title('Espectro Z')
121
122 %Plot Velocity
123 figure (2)
124
```

```
125     plot(VelD(:,1),VelD(:,2))
126
127 %Plot Audio
128 figure (3)
129
130     %Plot Samples
131     subplot(2,1,1);
132         plot(au,'g','linewidth',1)
133         xlim([0 length(au)]);
134
135         grid on
136
137     %Frequency Domain Plot
138     subplot(2,1,2);
139         xlim([0 55000]);
140         plot((1:55000),auf(1:55000),'r','linewidth',1.5); %Plot auf
141
142         grid on
```

B.3. DAQSoft - HMI implementation (App Designer)

```
classdef DAQSoft < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        DAQSoftTestBenchUIFigure       matlab.ui.Figure
        GridLayout                     matlab.ui.container.GridLayout
        LeftPanel                      matlab.ui.container.Panel
        Panel                          matlab.ui.container.Panel
        AccelerometerLabel             matlab.ui.control.Label
        SerialPortSetupLabel           matlab.ui.control.Label
        SerialPortDropDownLabel        matlab.ui.control.Label
        PortAcc                        matlab.ui.control.DropDown
        BaudrateDropDownLabel          matlab.ui.control.Label
        BaudrateAcc                    matlab.ui.control.DropDown
        SerialPortDropDown_2Label      matlab.ui.control.Label
        PortMicro                      matlab.ui.control.DropDown
        BaudrateDropDown_2Label        matlab.ui.control.Label
        BaudrateMicro                  matlab.ui.control.DropDown
        MicrocontrollerLabel           matlab.ui.control.Label
        RefreshPort                    matlab.ui.control.Button
        OpenPorts                      matlab.ui.control.Button
        VelocityEditField              matlab.ui.control.NumericEditField
        VelocityLabel                  matlab.ui.control.Label
        VelIndicator                   matlab.ui.control.SemicircularGauge
        Image2                         matlab.ui.control.Image
        Test                           matlab.ui.control.Button
        SecLabel_2                     matlab.ui.control.Label
        TestTimeEditFieldLabel         matlab.ui.control.Label
        TestTimeEditField              matlab.ui.control.NumericEditField
        RPMLabel                       matlab.ui.control.Label
        InclinometerArtificialHorizonLabel  matlab.ui.control.Label
        InclinometerArtificialHorizon  Aero.ui.control.ArtificialHorizon
        RightPanel                     matlab.ui.container.Panel
        TabGroup                       matlab.ui.container.TabGroup
        VibrationTab                   matlab.ui.container.Tab
        TabGroup3                      matlab.ui.container.TabGroup
        AccXTab                        matlab.ui.container.Tab
        AccxAxes                       matlab.ui.control.UIAxes
        EspxAxes                       matlab.ui.control.UIAxes
        AccYTab                        matlab.ui.container.Tab
        AccyAxes                       matlab.ui.control.UIAxes
```

```
42          EspyAxes                        matlab.ui.control.UIAxes
43          AccZTab                         matlab.ui.container.Tab
44          AcczAxes                        matlab.ui.control.UIAxes
45          EspzAxes                        matlab.ui.control.UIAxes
46          DataTab_2                       matlab.ui.container.Tab
47          AccData                         matlab.ui.control.Table
48          VelocityTab                     matlab.ui.container.Tab
49          TabGroup2                       matlab.ui.container.TabGroup
50          GraphTab                        matlab.ui.container.Tab
51          VelGraph                        matlab.ui.control.UIAxes
52          DataTab                         matlab.ui.container.Tab
53          VelData                         matlab.ui.control.Table
54          SoundTab                        matlab.ui.container.Tab
55          AudioPlot                       matlab.ui.control.UIAxes
56          EspAudio                        matlab.ui.control.UIAxes
57          DesignedanddevelopedbyAlejandroRojasLopez2020v10Label  ...
               matlab.ui.control.Label
58          ComStatusAcc                    matlab.ui.control.Label
59          ComStatusMicro                  matlab.ui.control.Label
60          Image                           matlab.ui.control.Image
61          TestBenchInductionMotorLabel    matlab.ui.control.Label
62          FacultyofMechatronicEngineeringLabel  matlab.ui.control.Label
63          DataacquisitionsoftwareLabel    matlab.ui.control.Label
64          SecLabel_3                      matlab.ui.control.Label
65          TimeLabel                       matlab.ui.control.Label
66          TimeEditField                   matlab.ui.control.NumericEditField
67      end
68
69      % Properties that correspond to apps with auto-reflow
70      properties (Access = private)
71          onePanelWidth = 576;
72      end
73
74
75      properties (Access = private)
76
77          %Serial Ports
78          s1  % Serial Port Micro
79          s2  % Serial Port ACC
80          Pitch  % Pitch Accelerometer
81          Roll  % Roll Accelerometer
82          Accx  %
83          Accy  %
84          Accz  %
85          Posx
86          Posy
```

```
87          Posz
88          Audio%
89
90          %Global Variable
91
92          t  % Time
93          Vel  % Vel
94          Flag1
95
96      end
97
98      methods (Access = private)
99
100         function  timeplot(app)
101
102               app.TimeEditField.Value=app.t;
103
104         end
105
106
107     end
108
109
110
111     % Callbacks that handle component events
112     methods (Access = private)
113
114         % Code that executes after component creation
115         function startupFcn(app)
116              app.PortAcc.Items=serialportlist;
117              app.PortMicro.Items=serialportlist;
118
119             instrreset
120             delete(instrfindall)
121             delete(timerfindall)
122         end
123
124         % Changes arrangement of the app based on UIFigure width
125         function updateAppLayout(app, event)
126             currentFigureWidth = app.DAQSoftTestBenchUIFigure.Position(3);
127             if(currentFigureWidth ≤ app.onePanelWidth)
128                 % Change to a 2x1 grid
129                 app.GridLayout.RowHeight = {872, 872};
130                 app.GridLayout.ColumnWidth = {'1x'};
131                 app.RightPanel.Layout.Row = 2;
132                 app.RightPanel.Layout.Column = 1;
```

```
133             else
134                 % Change to a 1x2 grid
135                 app.GridLayout.RowHeight = {'1x'};
136                 app.GridLayout.ColumnWidth = {220, '1x'};
137                 app.RightPanel.Layout.Row = 1;
138                 app.RightPanel.Layout.Column = 2;
139             end
140         end
141 
142         % Value changed function: PortAcc
143         function PortAccValueChanged(app, event)
144             PortACC = app.PortAcc.Value;
145 
146 
147         end
148 
149         % Button pushed function: RefreshPort
150         function RefreshPortButtonPushed(app, event)
151             app.PortAcc.Items=serialportlist;
152             app.PortMicro.Items=serialportlist;
153         end
154 
155         % Close request function: DAQSoftTestBenchUIFigure
156         function DAQSoftTestBenchUIFigureCloseRequest(app, event)
157 
158             delete(app)
159 
160 
161         end
162 
163         % Button pushed function: Test
164         function TestButtonPushed(app, event)
165 
166 
167 
168             % Inicializacion Variables Micro
169             app.Flag1=1;
170             app.Test.Enable=false;
171 
172             Veld=[0 0];
173               t=0;
174               Vel = 0;
175 
176               TestTime=app.TestTimeEditField.Value
177               cla(app.VelGraph)
178               app.VelGraph.XLim=[0 TestTime]
```

```
179             app.VelGraph.YLim=[0 1400]
180             p=animatedline(app.VelGraph,t,Vel,'Color','r');
181
182              %Inicializacion Variables ACC
183            Accx=0;
184            Accy=0;
185            Accz=1;
186            Accv=[0 0 0 0];
187
188
189
190            cla(app.AccxAxes)
191            cla(app.AccyAxes)
192            cla(app.AcczAxes)
193
194            cla(app.EspxAxes)
195            cla(app.EspyAxes)
196            cla(app.EspzAxes)
197            cla(app.AudioPlot)
198            cla(app.EspAudio)
199
200            app.AccxAxes.XLim=[0 TestTime]
201            app.AccxAxes.YLim=[-0.5 0.5]
202
203            app.AccyAxes.XLim=[0 TestTime]
204            app.AccyAxes.YLim=[-0.5 0.5]
205
206            app.AcczAxes.XLim=[0 TestTime]
207            app.AcczAxes.YLim=[0.5 1.5]
208
209
210            ax=animatedline(app.AccxAxes,t,Accx,'Color','g');
211            ay=animatedline(app.AccyAxes,t,Accy,'Color','r');
212            az=animatedline(app.AcczAxes,t,Accz,'Color','b');
213            flush(app.s1)
214           flush(app.s2)
215
216            %Audio capture initialization
217
218            audio=audiorecorder(44100,24,1);
219            record (audio);
220
221            write(app.s1,52,"char");
222
223
224                while (t<TestTime)
```

```
225
226                     % Read ACC
227                 Head=read(app.s2,2,'uint8');
228
229                   if (Head(1)≠85)
230                       continue;
231                   end
232
233                   switch(Head(2))
234
235                       case 81
236
237                           Acc=read(app.s2,3,'int16')/32768*16;
238                           Accx=Acc(:,1);
239                           Accy=Acc(:,2);
240                           Accz=Acc(:,3);
241
242
243                       Accv=[Accv;[t Acc(:,1) Acc(:,2) Acc(:,3)]];
244
245
246                   end
247
248 %
249
250
251                   addpoints(ax,t,Accx);
252                   addpoints(ay,t,Accy);
253                   addpoints(az,t,Accz);
254
255
256                   %Read Micro
257
258
259                   End=read(app.s2,3,'uint8');
260
261                   data=readline(app.s1);
262                   [t Vel] = strread(data,'%f%f',1,'delimiter',',');
263
264                   Vel=Vel*(30/pi);
265
266                   app.t=t;
267                   app.Vel=Vel;
268
269
270                   app.VelocityEditField.Value=Vel;
```

```
271             app.VelIndicator.Value=Vel;
272
273             Veld = [Veld;[t Vel]];
274             timeplot(app);
275             addpoints(p,t,Vel);
276
277             drawnow limitrate nocallbacks;
278
279
280             end
281
282
283         stop(audio); %Stop Audio Capture
284          au=getaudiodata(audio); %Audio data creation vector
285
286          %Audio Spectrum
287          Fsa=44100;
288         La=length(au);
289         Nffta=2^nextpow2(La);
290         a=fft(au,Nffta)/La;
291         fa=Fsa/2*linspace(0,1,Nffta/2+1);
292
293
294         plot(app.AudioPlot,au,'color','g');
295         plot(app.EspAudio,fa,2*abs(a(1:Nffta/2+1)),'color','b');
296         app.VelData.Data=Veld;
297         app.AccData.Data=Accv;
298
299
300         Fsv=200;
301
302         %Axis X Spectrum
303         Lx=length(Accv(:,2))
304         Nfftx=2^nextpow2(Lx);
305         x=fft(Accv(:,2),Nfftx)/Lx;
306         fx=Fsv/2*linspace(0,1,Nfftx/2+1);
307         plot(app.EspxAxes,fx,2*abs(x(1:Nfftx/2+1)),'color','b');
308
309         %Axis Y Spectrum
310         Ly=length(Accv(:,3))
311         Nffty=2^nextpow2(Ly);
312         y=fft(Accv(:,3),Nffty)/Ly;
313         fy=Fsv/2*linspace(0,1,Nffty/2+1);
314         plot(app.EspyAxes,fy,2*abs(y(1:Nffty/2+1)),'color','b');
315
316         %Axis Z Spectrum
```

```
317             Lz=length(Accv(:,4))
318             Nfftz=2^nextpow2(Lz);
319             z=fft(Accv(:,4),Nfftz)/Lz;
320             fz=Fsv/2*linspace(0,1,Nfftz/2+1);
321             plot(app.EspzAxes,fz,2*abs(z(1:Nfftz/2+1)),'color','b');
322
323
324
325             write(app.s1,48,"char");
326             app.t=0;
327             Vel=0;
328             app.VelocityEditField.Value=Vel;
329             app.VelIndicator.Value=Vel;
330             timeplot(app);
331             app.Test.Enable=true;
332             app.Test.Text='Repit Test'
333             app.Flag1=0
334
335         end
336
337         % Button pushed function: OpenPorts
338         function OpenPortsButtonPushed(app, event)
339
340             app.Flag1=0;
341             app.ComStatusAcc.Visible = true;
342             app.ComStatusMicro.Visible = true;
343             app.ComStatusAcc.Text= sprintf('%s Open',app.PortAcc.Value);
344             app.ComStatusMicro.Text= sprintf('%s ...
                    Open',app.PortMicro.Value);
345
346
347             BaudMic=str2double(app.BaudrateMicro.Value);
348             BaudAcc=str2double(app.BaudrateAcc.Value);
349
350             %Port Selection
351             app.s1=serialport(app.PortMicro.Value,BaudMic);
352             app.s2=serialport(app.PortAcc.Value,BaudAcc);
353
354             app.OpenPorts.Enable=false;
355             app.PortAcc.Enable=false;
356             app.PortMicro.Enable=false;
357             app.BaudrateAcc.Enable=false;
358             app.BaudrateMicro.Enable=false;
359
360             %Activation Test
361
```

```
362             app.TabGroup.Visible=true;
363             app.Test.Enable=true;
364             app.TestTimeEditField.Enable=true;
365             app.TestTimeEditFieldLabel.Enable=true;
366             app.SecLabel_2.Enable=true;
367
368
369
370         end
371     end
372
373     % Component initialization
374     methods (Access = private)
375
376         % Create UIFigure and components
377         function createComponents(app)
378
379             % Create DAQSoftTestBenchUIFigure and hide until all ...
                    components are created
380             app.DAQSoftTestBenchUIFigure = uifigure('Visible', 'off');
381             app.DAQSoftTestBenchUIFigure.AutoResizeChildren = 'off';
382             app.DAQSoftTestBenchUIFigure.Position = [100 100 1072 872];
383             app.DAQSoftTestBenchUIFigure.Name = 'DAQ Soft Test Bench';
384             app.DAQSoftTestBenchUIFigure.Resize = 'off';
385             app.DAQSoftTestBenchUIFigure.CloseRequestFcn = ...
                    createCallbackFcn(app, ...
                    @DAQSoftTestBenchUIFigureCloseRequest, true);
386             app.DAQSoftTestBenchUIFigure.SizeChangedFcn = ...
                    createCallbackFcn(app, @updateAppLayout, true);
387
388             % Create GridLayout
389             app.GridLayout = uigridlayout(app.DAQSoftTestBenchUIFigure);
390             app.GridLayout.ColumnWidth = {220, '1x'};
391             app.GridLayout.RowHeight = {'1x'};
392             app.GridLayout.ColumnSpacing = 0;
393             app.GridLayout.RowSpacing = 0;
394             app.GridLayout.Padding = [0 0 0 0];
395             app.GridLayout.Scrollable = 'on';
396
397             % Create LeftPanel
398             app.LeftPanel = uipanel(app.GridLayout);
399             app.LeftPanel.BackgroundColor = [1 1 1];
400             app.LeftPanel.Layout.Row = 1;
401             app.LeftPanel.Layout.Column = 1;
402
403             % Create Panel
```

```
404             app.Panel = uipanel(app.LeftPanel);
405             app.Panel.BackgroundColor = [1 1 1];
406             app.Panel.Position = [0 6 219 269];
407
408             % Create AccelerometerLabel
409             app.AccelerometerLabel = uilabel(app.Panel);
410             app.AccelerometerLabel.FontWeight = 'bold';
411             app.AccelerometerLabel.Position = [5 210 89 22];
412             app.AccelerometerLabel.Text = 'Accelerometer';
413
414             % Create SerialPortSetupLabel
415             app.SerialPortSetupLabel = uilabel(app.Panel);
416             app.SerialPortSetupLabel.FontName = 'Malgun Gothic';
417             app.SerialPortSetupLabel.FontWeight = 'bold';
418             app.SerialPortSetupLabel.FontAngle = 'italic';
419             app.SerialPortSetupLabel.Position = [5 241 102 22];
420             app.SerialPortSetupLabel.Text = 'Serial Port Setup';
421
422             % Create SerialPortDropDownLabel
423             app.SerialPortDropDownLabel = uilabel(app.Panel);
424             app.SerialPortDropDownLabel.HorizontalAlignment = 'right';
425             app.SerialPortDropDownLabel.Position = [22 186 62 22];
426             app.SerialPortDropDownLabel.Text = 'Serial Port';
427
428             % Create PortAcc
429             app.PortAcc = uidropdown(app.Panel);
430             app.PortAcc.Items = {'COM1'};
431             app.PortAcc.ValueChangedFcn = createCallbackFcn(app, ...
                    @PortAccValueChanged, true);
432             app.PortAcc.Position = [99 186 100 22];
433             app.PortAcc.Value = 'COM1';
434
435             % Create BaudrateDropDownLabel
436             app.BaudrateDropDownLabel = uilabel(app.Panel);
437             app.BaudrateDropDownLabel.HorizontalAlignment = 'right';
438             app.BaudrateDropDownLabel.Position = [29 154 54 22];
439             app.BaudrateDropDownLabel.Text = 'Baudrate';
440
441             % Create BaudrateAcc
442             app.BaudrateAcc = uidropdown(app.Panel);
443             app.BaudrateAcc.Items = {'9600', '14400', '19200', '38400', ...
                    '57600', '115200', '128000', '256000'};
444             app.BaudrateAcc.Position = [97 154 100 22];
445             app.BaudrateAcc.Value = '9600';
446
447             % Create SerialPortDropDown_2Label
```

```
448             app.SerialPortDropDown_2Label = uilabel(app.Panel);
449             app.SerialPortDropDown_2Label.HorizontalAlignment = 'right';
450             app.SerialPortDropDown_2Label.Position = [22 88 62 22];
451             app.SerialPortDropDown_2Label.Text = 'Serial Port';
452
453             % Create PortMicro
454             app.PortMicro = uidropdown(app.Panel);
455             app.PortMicro.Items = {};
456             app.PortMicro.Position = [98 88 100 22];
457             app.PortMicro.Value = {};
458
459             % Create BaudrateDropDown_2Label
460             app.BaudrateDropDown_2Label = uilabel(app.Panel);
461             app.BaudrateDropDown_2Label.HorizontalAlignment = 'right';
462             app.BaudrateDropDown_2Label.Position = [29 56 54 22];
463             app.BaudrateDropDown_2Label.Text = 'Baudrate';
464
465             % Create BaudrateMicro
466             app.BaudrateMicro = uidropdown(app.Panel);
467             app.BaudrateMicro.Items = {'9600', '14400', '19200', ...
                    '38400', '57600', '115200', '128000', '256000'};
468             app.BaudrateMicro.Position = [97 56 100 22];
469             app.BaudrateMicro.Value = '9600';
470
471             % Create MicrocontrollerLabel
472             app.MicrocontrollerLabel = uilabel(app.Panel);
473             app.MicrocontrollerLabel.FontWeight = 'bold';
474             app.MicrocontrollerLabel.Position = [5 115 93 22];
475             app.MicrocontrollerLabel.Text = 'Microcontroller';
476
477             % Create RefreshPort
478             app.RefreshPort = uibutton(app.Panel, 'push');
479             app.RefreshPort.ButtonPushedFcn = createCallbackFcn(app, ...
                    @RefreshPortButtonPushed, true);
480             app.RefreshPort.Position = [146 241 65 22];
481             app.RefreshPort.Text = 'Refresh';
482
483             % Create OpenPorts
484             app.OpenPorts = uibutton(app.Panel, 'push');
485             app.OpenPorts.ButtonPushedFcn = createCallbackFcn(app, ...
                    @OpenPortsButtonPushed, true);
486             app.OpenPorts.Position = [67 7 80 22];
487             app.OpenPorts.Text = 'Open Ports';
488
489             % Create VelocityEditField
490             app.VelocityEditField = uieditfield(app.LeftPanel, 'numeric');
```

```
491             app.VelocityEditField.Position = [134 365 42 22];
492
493             % Create VelocityLabel
494             app.VelocityLabel = uilabel(app.LeftPanel);
495             app.VelocityLabel.HorizontalAlignment = 'center';
496             app.VelocityLabel.Position = [104 279 47 22];
497             app.VelocityLabel.Text = 'Velocity';
498
499             % Create VelIndicator
500             app.VelIndicator = uigauge(app.LeftPanel, 'semicircular');
501             app.VelIndicator.Limits = [0 2000];
502             app.VelIndicator.Orientation = 'west';
503             app.VelIndicator.ScaleColors = [1 0 0;1 1 0;0.3922 0.8314 ...
                    0.0745];
504             app.VelIndicator.ScaleColorLimits = [1800 2000;1200 1800;0 ...
                    1200];
505             app.VelIndicator.Position = [67 316 65 120];
506
507             % Create Image2
508             app.Image2 = uiimage(app.LeftPanel);
509             app.Image2.Position = [5 714 209 152];
510             app.Image2.ImageSource = 'TestBench.png';
511
512             % Create Test
513             app.Test = uibutton(app.LeftPanel, 'push');
514             app.Test.ButtonPushedFcn = createCallbackFcn(app, ...
                    @TestButtonPushed, true);
515             app.Test.Enable = 'off';
516             app.Test.Position = [69 649 80 22];
517             app.Test.Text = 'Test';
518
519             % Create SecLabel_2
520             app.SecLabel_2 = uilabel(app.LeftPanel);
521             app.SecLabel_2.Enable = 'off';
522             app.SecLabel_2.Position = [134 685 26 22];
523             app.SecLabel_2.Text = 'Sec';
524
525             % Create TestTimeEditFieldLabel
526             app.TestTimeEditFieldLabel = uilabel(app.LeftPanel);
527             app.TestTimeEditFieldLabel.HorizontalAlignment = 'right';
528             app.TestTimeEditFieldLabel.Position = [11 685 57 22];
529             app.TestTimeEditFieldLabel.Text = 'Test Time';
530
531             % Create TestTimeEditField
532             app.TestTimeEditField = uieditfield(app.LeftPanel, 'numeric');
533             app.TestTimeEditField.Position = [83 685 41 22];
```

```
534             app.TestTimeEditField.Value = 1;
535
536             % Create RPMLabel
537             app.RPMLabel = uilabel(app.LeftPanel);
538             app.RPMLabel.Position = [182 364 32 22];
539             app.RPMLabel.Text = 'RPM';
540
541             % Create InclinometerArtificialHorizonLabel
542             app.InclinometerArtificialHorizonLabel = ...
                    uilabel(app.LeftPanel);
543             app.InclinometerArtificialHorizonLabel.HorizontalAlignment ...
                    = 'center';
544             app.InclinometerArtificialHorizonLabel.Position = [74 447 ...
                    71 22];
545             app.InclinometerArtificialHorizonLabel.Text = 'Inclinometer';
546
547             % Create InclinometerArtificialHorizon
548             app.InclinometerArtificialHorizon = ...
                    uiaerohorizon(app.LeftPanel);
549             app.InclinometerArtificialHorizon.Position = [49 484 120 120];
550
551             % Create RightPanel
552             app.RightPanel = uipanel(app.GridLayout);
553             app.RightPanel.BackgroundColor = [1 1 1];
554             app.RightPanel.Layout.Row = 1;
555             app.RightPanel.Layout.Column = 2;
556
557             % Create TabGroup
558             app.TabGroup = uitabgroup(app.RightPanel);
559             app.TabGroup.TabLocation = 'bottom';
560             app.TabGroup.Position = [12 29 832 700];
561
562             % Create VibrationTab
563             app.VibrationTab = uitab(app.TabGroup);
564             app.VibrationTab.Title = 'Vibration';
565             app.VibrationTab.BackgroundColor = [1 1 1];
566             app.VibrationTab.ForegroundColor = [1 0 0];
567
568             % Create TabGroup3
569             app.TabGroup3 = uitabgroup(app.VibrationTab);
570             app.TabGroup3.TabLocation = 'left';
571             app.TabGroup3.Position = [0 1 832 674];
572
573             % Create AccXTab
574             app.AccXTab = uitab(app.TabGroup3);
575             app.AccXTab.Title = 'Acc X';
```

```
576             app.AccXTab.BackgroundColor = [1 1 1];
577
578             % Create AccxAxes
579             app.AccxAxes = uiaxes(app.AccXTab);
580             title(app.AccxAxes, 'X Axis Acceleration')
581             xlabel(app.AccxAxes, 'Time [Sec]')
582             ylabel(app.AccxAxes, 'Acceleration [G]')
583             app.AccxAxes.GridAlpha = 1;
584             app.AccxAxes.MinorGridAlpha = 0.5;
585             app.AccxAxes.XMinorGrid = 'on';
586             app.AccxAxes.YMinorGrid = 'on';
587             app.AccxAxes.BackgroundColor = [1 1 1];
588             app.AccxAxes.Position = [11 335 738 333];
589
590             % Create EspxAxes
591             app.EspxAxes = uiaxes(app.AccXTab);
592             title(app.EspxAxes, 'Frecuency Domain')
593             xlabel(app.EspxAxes, 'Frecuency [Hz]')
594             ylabel(app.EspxAxes, 'Gain')
595             app.EspxAxes.BackgroundColor = [1 1 1];
596             app.EspxAxes.Position = [12 3 738 333];
597
598             % Create AccYTab
599             app.AccYTab = uitab(app.TabGroup3);
600             app.AccYTab.Title = 'Acc Y';
601             app.AccYTab.BackgroundColor = [1 1 1];
602
603             % Create AccyAxes
604             app.AccyAxes = uiaxes(app.AccYTab);
605             title(app.AccyAxes, 'Y Axis Acceleration')
606             xlabel(app.AccyAxes, 'Time [Sec]')
607             ylabel(app.AccyAxes, 'Acceleration [G]')
608             app.AccyAxes.GridAlpha = 1;
609             app.AccyAxes.MinorGridAlpha = 0.5;
610             app.AccyAxes.XMinorGrid = 'on';
611             app.AccyAxes.YMinorGrid = 'on';
612             app.AccyAxes.BackgroundColor = [1 1 1];
613             app.AccyAxes.Position = [11 335 738 333];
614
615             % Create EspyAxes
616             app.EspyAxes = uiaxes(app.AccYTab);
617             title(app.EspyAxes, 'Frecuency Domain')
618             xlabel(app.EspyAxes, 'Frecuency [Hz]')
619             ylabel(app.EspyAxes, 'Gain')
620             app.EspyAxes.BackgroundColor = [1 1 1];
621             app.EspyAxes.Position = [12 3 738 333];
```

```
622
623             % Create AccZTab
624             app.AccZTab = uitab(app.TabGroup3);
625             app.AccZTab.Title = 'Acc Z';
626             app.AccZTab.BackgroundColor = [1 1 1];
627
628             % Create AcczAxes
629             app.AcczAxes = uiaxes(app.AccZTab);
630             title(app.AcczAxes, 'Z Axis Acceleration')
631             xlabel(app.AcczAxes, 'Time [Sec]')
632             ylabel(app.AcczAxes, 'Acceleration [G]')
633             app.AcczAxes.GridAlpha = 1;
634             app.AcczAxes.MinorGridAlpha = 0.5;
635             app.AcczAxes.XMinorGrid = 'on';
636             app.AcczAxes.YMinorGrid = 'on';
637             app.AcczAxes.BackgroundColor = [1 1 1];
638             app.AcczAxes.Position = [11 335 738 333];
639
640             % Create EspzAxes
641             app.EspzAxes = uiaxes(app.AccZTab);
642             title(app.EspzAxes, 'Frecuency Domain')
643             xlabel(app.EspzAxes, 'Frecuency [Hz]')
644             ylabel(app.EspzAxes, 'Gain')
645             app.EspzAxes.BackgroundColor = [1 1 1];
646             app.EspzAxes.Position = [12 3 738 333];
647
648             % Create DataTab_2
649             app.DataTab_2 = uitab(app.TabGroup3);
650             app.DataTab_2.Title = 'Data';
651
652             % Create AccData
653             app.AccData = uitable(app.DataTab_2);
654             app.AccData.ColumnName = {'Time [Seg]'; 'Acc X [G]'; 'Acc Y ...
                    [G]'; 'Acc Z [G]'};
655             app.AccData.RowName = {};
656             app.AccData.Position = [4 4 760 665];
657
658             % Create VelocityTab
659             app.VelocityTab = uitab(app.TabGroup);
660             app.VelocityTab.Title = 'Velocity';
661             app.VelocityTab.ForegroundColor = [1 0 0];
662
663             % Create TabGroup2
664             app.TabGroup2 = uitabgroup(app.VelocityTab);
665             app.TabGroup2.TabLocation = 'left';
666             app.TabGroup2.Position = [0 1 831 674];
```

```
667
668             % Create GraphTab
669             app.GraphTab = uitab(app.TabGroup2);
670             app.GraphTab.Title = 'Graph';
671             app.GraphTab.BackgroundColor = [1 1 1];
672
673             % Create VelGraph
674             app.VelGraph = uiaxes(app.GraphTab);
675             title(app.VelGraph, 'Motor Velocity')
676             xlabel(app.VelGraph, 'Time [Sec]')
677             ylabel(app.VelGraph, 'Velocity [RPM]')
678             app.VelGraph.GridAlpha = 1;
679             app.VelGraph.MinorGridAlpha = 0.5;
680             app.VelGraph.XMinorGrid = 'on';
681             app.VelGraph.YMinorGrid = 'on';
682             app.VelGraph.BackgroundColor = [1 1 1];
683             app.VelGraph.Position = [13 183 738 333];
684
685             % Create DataTab
686             app.DataTab = uitab(app.TabGroup2);
687             app.DataTab.Title = 'Data';
688
689             % Create VelData
690             app.VelData = uitable(app.DataTab);
691             app.VelData.ColumnName = {'Time [Seg]'; 'Vel [Rpm]'};
692             app.VelData.RowName = {};
693             app.VelData.Position = [4 7 759 661];
694
695             % Create SoundTab
696             app.SoundTab = uitab(app.TabGroup);
697             app.SoundTab.Title = 'Sound';
698             app.SoundTab.BackgroundColor = [1 1 1];
699             app.SoundTab.ForegroundColor = [1 0 0];
700
701             % Create AudioPlot
702             app.AudioPlot = uiaxes(app.SoundTab);
703             title(app.AudioPlot, 'Audio Capture')
704             xlabel(app.AudioPlot, 'Samples')
705             ylabel(app.AudioPlot, 'Gain [dB]')
706             app.AudioPlot.GridAlpha = 1;
707             app.AudioPlot.MinorGridAlpha = 0.5;
708             app.AudioPlot.XMinorGrid = 'on';
709             app.AudioPlot.YMinorGrid = 'on';
710             app.AudioPlot.BackgroundColor = [1 1 1];
711             app.AudioPlot.Position = [44 334 738 333];
712
```

```
713             % Create EspAudio
714             app.EspAudio = uiaxes(app.SoundTab);
715             title(app.EspAudio, 'Frecuency Domain')
716             xlabel(app.EspAudio, 'Frecuency [Hz]')
717             ylabel(app.EspAudio, 'Gain')
718             app.EspAudio.BackgroundColor = [1 1 1];
719             app.EspAudio.Position = [45 2 738 333];
720
721             % Create DesignedanddevelopedbyAlejandroRojasLopez2020v10Label
722             app.DesignedanddevelopedbyAlejandroRojasLopez2020v10Label = ...
                    uilabel(app.RightPanel);
723             app.DesignedanddevelopedbyAlejandroRojasLopez2020v10Label.FontName ...
                    = 'Comic Sans MS';
724             app.DesignedanddevelopedbyAlejandroRojasLopez2020v10Label.FontSize ...
                    = 9;
725             app.DesignedanddevelopedbyAlejandroRojasLopez2020v10Label.FontAngle ...
                    = 'italic';
726             app.DesignedanddevelopedbyAlejandroRojasLopez2020v10Label.Position ...
                    = [576 1 270 16];
727             app.DesignedanddevelopedbyAlejandroRojasLopez2020v10Label.Text ...
                    = 'Designed and developed by Alejandro Rojas Lopez - ...
                    2020 v 1.0';
728
729             % Create ComStatusAcc
730             app.ComStatusAcc = uilabel(app.RightPanel);
731             app.ComStatusAcc.Position = [13 2 70 14];
732             app.ComStatusAcc.Text = 'Acc Closed!';
733
734             % Create ComStatusMicro
735             app.ComStatusMicro = uilabel(app.RightPanel);
736             app.ComStatusMicro.Position = [126 1 79 15];
737             app.ComStatusMicro.Text = 'Micro Closed!';
738
739             % Create Image
740             app.Image = uiimage(app.RightPanel);
741             app.Image.Position = [564 754 282 97];
742             app.Image.ImageSource = 'san_buenaventura.png';
743
744             % Create TestBenchInductionMotorLabel
745             app.TestBenchInductionMotorLabel = uilabel(app.RightPanel);
746             app.TestBenchInductionMotorLabel.FontName = 'Calibri';
747             app.TestBenchInductionMotorLabel.FontSize = 35;
748             app.TestBenchInductionMotorLabel.FontWeight = 'bold';
749             app.TestBenchInductionMotorLabel.FontAngle = 'italic';
750             app.TestBenchInductionMotorLabel.Position = [53 805 520 46];
```

```
751             app.TestBenchInductionMotorLabel.Text = 'Induction Motor ...
                    Failure Test Bench';
752
753             % Create FacultyofMechatronicEngineeringLabel
754             app.FacultyofMechatronicEngineeringLabel = ...
                    uilabel(app.RightPanel);
755             app.FacultyofMechatronicEngineeringLabel.Position = [216 ...
                    784 195 22];
756             app.FacultyofMechatronicEngineeringLabel.Text = 'Faculty of ...
                    Mechatronic Engineering';
757
758             % Create DataacquisitionsoftwareLabel
759             app.DataacquisitionsoftwareLabel = uilabel(app.RightPanel);
760             app.DataacquisitionsoftwareLabel.Position = [244 761 140 22];
761             app.DataacquisitionsoftwareLabel.Text = 'Data acquisition ...
                    software';
762
763             % Create SecLabel_3
764             app.SecLabel_3 = uilabel(app.RightPanel);
765             app.SecLabel_3.Position = [815 730 26 22];
766             app.SecLabel_3.Text = 'Sec';
767
768             % Create TimeLabel
769             app.TimeLabel = uilabel(app.RightPanel);
770             app.TimeLabel.HorizontalAlignment = 'right';
771             app.TimeLabel.Position = [743 731 35 22];
772             app.TimeLabel.Text = 'Time:';
773
774             % Create TimeEditField
775             app.TimeEditField = uieditfield(app.RightPanel, 'numeric');
776             app.TimeEditField.RoundFractionalValues = 'on';
777             app.TimeEditField.ValueDisplayFormat = '%11g';
778             app.TimeEditField.Position = [785 731 23 22];
779
780             % Show the figure after all components are created
781             app.DAQSoftTestBenchUIFigure.Visible = 'on';
782         end
783     end
784
785     % App creation and deletion
786     methods (Access = public)
787
788         % Construct app
789         function app = Prueba1
790
791             % Create UIFigure and components
```

```
792             createComponents(app)
793
794             % Register the app with App Designer
795             registerApp(app, app.DAQSoftTestBenchUIFigure)
796
797             % Execute the startup function
798             runStartupFcn(app, @startupFcn)
799
800             if nargout == 0
801                 clear app
802             end
803         end
804
805         % Code that executes before app deletion
806         function delete(app)
807
808             % Delete UIFigure when app is deleted
809             delete(app.DAQSoftTestBenchUIFigure)
810         end
811     end
812 end
```

Bibliografía

[1] A. Glowacz, "Acoustic based fault diagnosis of three-phase induction motor," vol. 137, pp. 82–89. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0003682X18300951

[2] A. Garcia-Perez, R. J. Romero-Troncoso, E. Cabal-Yepez, R. A. Osornio-Rios, and J. A. Lucio-Martinez, "Application of high-resolution spectral analysis for identifying faults in induction motors by means of sound," vol. 18, no. 11, pp. 1585–1594. [Online]. Available: http://journals.sagepub.com/doi/10.1177/1077546311422925

[3] S. Prainetr, S. Wangnippanto, and S. Tunyasirut, "Detection mechanical fault of induction motor using harmonic current and sound acoustic," in *2017 International Electrical Engineering Congress (iEECON)*. IEEE, pp. 1–4. [Online]. Available: http://ieeexplore.ieee.org/document/8075725/

[4] A. Glowacz and Z. Glowacz, "Diagnosis of stator faults of the single-phase induction motor using acoustic signals," vol. 117, pp. 20–27. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0003682X16303401

[5] A. Glowacz, W. Glowacz, Z. Glowacz, and J. Kozik, "Early fault diagnosis of bearing and stator faults of the single-phase induction motor using acoustic signals," vol. 113, pp. 1–9. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0263224117305432

[6] H. Akcay and E. Germen, "Identification of acoustic spectra for fault detection in induction motors," in *2013 Africon*. IEEE, pp. 1–5. [Online]. Available: http://ieeexplore.ieee.org/document/6757650/

[7] P. A. Delgado-Arredondo, D. Morinigo-Sotelo, R. A. Osornio-Rios, J. G. Avina-Cervantes, H. Rostro-Gonzalez, and R. d. J. Romero-Troncoso, "Methodology for fault detection in induction motors via sound and vibration signals," vol. 83, pp. 568–589. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0888327016302151

[8] L. Haas, "Reprints available directly from the publisher photocopying permitted by license only," vol. 17, no. 1, pp. 1–6. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/1071441950170102

[9] E. Germen, M. Başaran, and M. Fidan, "Sound based induction motor fault diagnosis using kohonen self-organizing map," vol. 46, no. 1, pp. 45–58. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S088832701300647X

[10] H. Akcay and E. Germen, "Subspace-based identification of acoustic noise spectra in induction motors," vol. 30, no. 1, pp. 32–40. [Online]. Available: http://ieeexplore.ieee.org/document/6860319/

[11] A. Bracale, G. Carpinelli, L. Piegari, and P. Tricoli, "A high resolution method for on line diagnosis of induction motors faults," in *2007 IEEE Lausanne Power Tech.* IEEE, pp. 994–998. [Online]. Available: http://ieeexplore.ieee.org/document/4538451/

[12] S. Sarkar, P. K. Hembram, P. Purkait, and S. Das, "Acquisition and pre-processing of three phase induction motor stator current signal for fault diagnosis using FPGA, NI compact-RIO real time controller," in *2016 IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics Engineering (UPCON).* IEEE, pp. 110–114. [Online]. Available: http://ieeexplore.ieee.org/document/7894635/

[13] C. Morales-Perez, J. Grande-Barreto, J. Rangel-Magdaleno, and H. Peregrina-Barreto, "Bearing fault detection in induction motors using MCSA and statistical analysis," in *2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC).* IEEE, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/document/8409780/

[14] W. Thomson and M. Fenger, "Case histories of current signature analysis to detect faults in induction motor drives," in *IEEE International Electric Machines and Drives Conference, 2003. IEMDC'03.*, vol. 3. IEEE, pp. 1459–1465. [Online]. Available: http://ieeexplore.ieee.org/document/1210644/

[15] X. Song, Z. Wang, and J. Hu, "Detection of bearing outer race fault in induction motors using motor current signature analysis," in *2019 22nd International Conference on Electrical Machines and Systems (ICEMS).* IEEE, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/document/8922036/

[16] V. C. M. N. Leite, J. G. Borges da Silva, G. F. C. Veloso, L. E. Borges da Silva, G. Lambert-Torres, E. L. Bonaldi, and L. E. de Lacerda de Oliveira, "Detection of localized bearing faults in induction machines by spectral kurtosis and envelope analysis of stator current," vol. 62, no. 3, pp. 1855–1865. [Online]. Available: https://ieeexplore.ieee.org/document/6871315/

[17] V. F. Pires, D. Foito, J. F. Martins, and A. J. Pires, "Detection of stator winding fault in induction motors using a motor square current signature analysis

(MSCSA)," in *2015 IEEE 5th International Conference on Power Engineering, Energy and Electrical Drives (POWERENG)*. IEEE, pp. 507–512. [Online]. Available: http://ieeexplore.ieee.org/document/7266369/

[18] R. Sharifi and M. Ebrahimi, "Detection of stator winding faults in induction motors using three-phase current monitoring," vol. 50, no. 1, pp. 14–20. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0019057810000959

[19] M. Fenger, B. Lloyd, and W. Thomson, "Development of a tool to detect faults in induction motors via current signature analysis," in *Cement Industry Technical Conference, 2003. Conference Record. IEEE-IAS/PCA 2003.* IEEE, pp. 37–46. [Online]. Available: http://ieeexplore.ieee.org/document/1204707/

[20] M. Blodt, J. Regnier, and J. Faucher, "Distinguishing load torque oscillations and eccentricity faults in induction motors using stator current wigner distributions," vol. 45, no. 6, pp. 1991–2000. [Online]. Available: http://ieeexplore.ieee.org/document/5238531/

[21] Y.-H. Kim, Y.-W. Youn, D.-H. Hwang, J.-H. Sun, and D.-S. Kang, "High-resolution parameter estimation method to identify broken rotor bar faults in induction motors," vol. 60, no. 9, pp. 4103–4117. [Online]. Available: http://ieeexplore.ieee.org/document/6355663/

[22] E. Geetha and C. Nagarajan, "Induction motor fault detection and classification using current signature analysis technique," in *2018 Conference on Emerging Devices and Smart Systems (ICEDSS)*. IEEE, pp. 48–52. [Online]. Available: https://ieeexplore.ieee.org/document/8544272/

[23] M. Blodt, M. Chabert, J. Regnier, and J. Faucher, "Mechanical load fault detection in induction motors by stator current time-frequency analysis," vol. 42, no. 6, pp. 1454–1463. [Online]. Available: http://ieeexplore.ieee.org/document/4012278/

[24] O. A. Mohammed, N. Y. Abed, and S. C. Ganu, "Modeling and characterization of induction motor internal faults using finite element and discrete wavelet transform," in *INTERMAG 2006 - IEEE International Magnetics Conference*. IEEE, pp. 769–769. [Online]. Available: http://ieeexplore.ieee.org/document/4262202/

[25] R. Chawla, B. Akhil Vinayak, and G. Jagadanand, "Modelling and detection of stator incipient open circuit fault in three-phase induction motor," in *2018 IEEE International Conference on Power Electronics, Drives and Energy Systems (PEDES)*. IEEE, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/8707681/

[26] L. Pereira, D. da Silva Gazzana, and L. Pereira, "Motor current signature analysis and fuzzy logic applied to the diagnosis of short-circuit faults in induction motors," in *31st Annual Conference of IEEE Industrial Electronics Society, 2005. IECON 2005.* IEEE, p. 6 pp. [Online]. Available: http://ieeexplore.ieee.org/document/1568916/

[27] L. Frosini and E. Bassi, "Stator current and motor efficiency as indicators for different types of bearing faults in induction motors," vol. 57, no. 1, pp. 244–251. [Online]. Available: http://ieeexplore.ieee.org/document/5166499/

[28] M. Drif and A. J. M. Cardoso, "Stator fault diagnostics in squirrel cage three-phase induction motor drives using the instantaneous active and reactive power signature analyses," vol. 10, no. 2, pp. 1348–1360. [Online]. Available: http://ieeexplore.ieee.org/document/6750029/

[29] V. F. Pires, T. G. Amaral, and J. F. Martins, "Stator winding fault diagnosis in induction motors using the dq current trajectory mass center," in *2009 35th Annual Conference of IEEE Industrial Electronics.* IEEE, pp. 1322–1326. [Online]. Available: http://ieeexplore.ieee.org/document/5414714/

[30] D. Lopez-Perez and J. Antonino-Daviu, "Application of infrared thermography to failure detection in industrial induction motors: Case stories," vol. 53, no. 3, pp. 1901–1908. [Online]. Available: http://ieeexplore.ieee.org/document/7822935/

[31] A. Glowacz and Z. Glowacz, "Diagnosis of the three-phase induction motor using thermal imaging," vol. 81, pp. 7–16. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1350449516306259

[32] A. G. Garcia-Ramirez, L. A. Morales-Hernandez, R. A. Osornio-Rios, J. P. Benitez-Rangel, A. Garcia-Perez, and R. d. J. Romero-Troncoso, "Fault detection in induction motors and the impact on the kinematic chain through thermographic analysis," vol. 114, pp. 1–9. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0378779614001266

[33] G. Singh, T. Anil Kumar, and V. Naikan, "Induction motor inter turn fault detection using infrared thermographic analysis," vol. 77, pp. 277–282. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1350449515301547

[34] M. Riera-Guasp, M. Cabanas, J. Antonino-Daviu, M. Pineda-Sanchez, and C. Garcia, "Influence of nonconsecutive bar breakages in motor current signature analysis for the diagnosis of rotor faults in induction motors," vol. 25, no. 1, pp. 80–89. [Online]. Available: http://ieeexplore.ieee.org/document/5350691/

[35] G. Singh and V. Naikan, "Infrared thermography based diagnosis of inter-turn fault and cooling system failure in three phase induction motor," vol. 87, pp. 134–138. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1350449517301196

[36] J. A. R. Nunez, L. M. Velazquez, L. A. M. Hernandez, R. J. R. Troncoso, R. A. Osornio-Rios, and H.-C. Telematica, "Low-cost thermographic analysis for bearing fault detection on induction motors," p. 4.

[37] F. Jeffali, B. E. Kihel, A. Nougaoui, and F. Delaunois, "Monitoring and diagnostic misalignment of asynchronous machines by infrared thermography," p. 8.

[38] A. G. Garcia-Ramirez, L. A. Morales-Hernandez, R. A. Osornio-Rios, A. Garcia-Perez, and R. J. Romero-Troncoso, "Thermographic technique as a complement for MCSA in induction motor fault detection," in *2014 International Conference on Electrical Machines (ICEM)*. IEEE, pp. 1940–1945. [Online]. Available: http://ieeexplore.ieee.org/document/6960449/

[39] M. J. Picazo-Rodenas, R. Royo, J. Antonino-Daviu, and J. Roger-Folch, "Use of infrared thermography for computation of heating curves and preliminary failure detection in induction motors," in *2012 XXth International Conference on Electrical Machines*. IEEE, pp. 525–531. [Online]. Available: http://ieeexplore.ieee.org/document/6349920/

[40] P. A. Delgado-Arredondo, A. Garcia-Perez, D. Morinigo-Sotelo, R. A. Osornio-Rios, J. G. Avina-Cervantes, H. Rostro-Gonzalez, and R. d. J. Romero-Troncoso, "Comparative study of time-frequency decomposition techniques for fault detection in induction motors using vibration analysis during startup transient," vol. 2015, pp. 1–14. [Online]. Available: http://www.hindawi.com/journals/sv/2015/708034/

[41] W. Li and C. K. Mechefske, "Detection of induction motor faults: A comparison of stator current, vibration and acoustic methods," vol. 12, no. 2, pp. 165–188. [Online]. Available: http://journals.sagepub.com/doi/10.1177/1077546306062097

[42] N. P. Sakhalkar and P. Korde, "Fault detection in induction motors based on motor current signature analysis and accelerometer," in *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*. IEEE, pp. 363–367. [Online]. Available: https://ieeexplore.ieee.org/document/8390117/

[43] G. Sun, Q. Hu, Q. Zhang, A. Qin, and L. Shao, "Fault diagnosis for rotating machinery based on artificial immune algorithm and evidence theory," in *The 27th Chinese Control and Decision Conference (2015 CCDC)*. IEEE, pp. 2975–2979. [Online]. Available: http://ieeexplore.ieee.org/document/7162380/

[44] Z. Kanovic, D. Matic, Z. Jelicic, M. Rapaic, B. Jakovljevic, and M. Kapetina, "Induction motor broken rotor bar detection using vibration analysis — a case study," in *2013 9th IEEE International Symposium on Diagnostics for Electric Machines, Power Electronics and Drives (SDEMPED)*. IEEE, pp. 64–68. [Online]. Available: http://ieeexplore.ieee.org/document/6645698/

[45] M. Moghadasian, S. M. Shakouhi, and S. S. Moosavi, "Induction motor fault diagnosis using ANFIS based on vibration signal spectrum analysis," in *2017 3rd International Conference on Frontiers of Signal Processing (ICFSP)*. IEEE, pp. 105–108. [Online]. Available: http://ieeexplore.ieee.org/document/8097151/

[46] P. A. Delgado-Arredondo, D. Morinigo-Sotelo, R. A. Osornio-Rios, J. G. Avina-Cervantes, H. Rostro-Gonzalez, and R. d. J. Romero-Troncoso, "Methodology for fault detection in induction motors via sound and vibration signals," vol. 83, pp. 568–589. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0888327016302151

[47] M. Hernandez-Vargas, E. Cabal-Yepez, and A. Garcia-Perez, "Real-time SVD-based detection of multiple combined faults in induction motors," vol. 40, no. 7, pp. 2193–2203. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0045790614000068

[48] H. Su, K. T. Chong, and R. Ravi Kumar, "Vibration signal analysis for electrical fault detection of induction machine using neural networks," vol. 20, no. 2, pp. 183–194. [Online]. Available: http://link.springer.com/10.1007/s00521-010-0512-3

[49] S. Korkua, H. Jain, W.-J. Lee, and C. Kwan, "Wireless health monitoring system for vibration detection of induction motors," in *2010 IEEE Industrial and Commercial Power Systems Technical Conference - Conference Record*. IEEE, pp. 1–6. [Online]. Available: http://ieeexplore.ieee.org/document/5489899/

[50] D. Karthik and T. R. Chelliah, "Analysis of scalar and vector control based efficiency-optimized induction motors subjected to inverter and sensor faults," p. 5.

[51] E. Marín, *Elementos de medición y análisis de vibraciones en máquinas rotatorias*. Editorial Félix Varela. [Online]. Available: https://books.google.com.co/books?id=W8MCtAEACAAJ

[52] G. K. Yamamoto, C. da Costa, and J. S. da Silva Sousa, "A smart experimental setup for vibration measurement and imbalance fault detection in rotating machinery," vol. 4, pp. 8–18. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2351988616300185

[53] P. L. Filho, R. Pederiva, and J. Brito, "Detection of stator winding faults in induction machines using an internal flux sensor," in *2007 IEEE International Symposium on Diagnostics for Electric Machines, Power Electronics and Drives*. IEEE, pp. 432–437. [Online]. Available: https://ieeexplore.ieee.org/document/4393133/

[54] U. Werner, "Vibration control of large induction motors using actuators between motor feet and steel frame foundation," vol. 112, pp. 319–342. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0888327018302358

[55] R. G. Budynas, *Diseño en ingeniería mecánica de Shigley*. McGrawhill, no. 9786071507716.