University of
HUDDERSFIELD

Wireless Sensor Network (WSN) Platform for Railway Condition Monitoring

By
Kalyan C Gundampati

WIRELESS SENSOR NETWORK (WSN) PLATFORM FOR RAILWAY CONDITION MONITORING

KALYAN CHAKRAVARTHY GUNDAMPATI

A thesis submitted to the University of Huddersfield in partial fulfillment of the requirements for the degree Master of Research in Engineering

THE UNIVERSITY OF HUDDERSFIELD

January 2019

Table of Contents

Abstract

In recent years, the sensors are cheaper and are widely available, and also many sensing technologies are rapidly emerging. Sensors are available in a compact size and hence can be installed at inaccessible situations with less effort. They are deployed on a large scale in various industries because of their low cost and discrete varieties. Due to their broad range, there is a rapid expansion in the condition monitoring of the rail systems, for instance, bridges, rail tracks, signaling, etc. Condition monitoring reduces the risks and time involved during human inspections by automated condition monitoring, and it also reduces the maintenance by identifying faults before they get worse and improves safety and reliability. With the rapid improvements in wireless communications, wireless sensor networks (WSNs) are used to monitor these infrastructures effectively using MEMS sensors. WSN condition monitoring reduces the risks, increases the reliability and provide secure communication using the IEEE 802.15.4 protocol. The paper summarizes a background and a literature survey on the WSNs in remote condition monitoring. This paper presents the application of remote condition monitoring by using the ADXL 335 MEMS accelerometer sensor. The sensor records the acceleration of the train and the signal is pre-processed before sending it to the WSN. The output is received at the base station wirelessly via WSN.

Keywords:

Wireless Sensor Networks, WSN nodes, Remote Condition Monitoring, Railway Condition Monitoring, WSN in railways, Accelerometer sensor in railways

LIST OF FIGURES

LIST OF TABLES

Acknowledgments

Firstly, I would like to thank YHWH, my GOD, and my heavenly FATHER, who created the heavens and the earth and the things inside them, for giving me an opportunity to pursue a Masters at the University of Huddersfield. I have been extremely grateful for his love towards me and also for helping me to come this far in my career. I want to thank HIM for giving me the strength, knowledge, financial assistance and ability to complete the thesis satisfactorily. Without his blessings, this achievement is not possible.

Secondly, I would like to thank my supervisor, Dr. Farouk Balouchi, Research Fellow, Department of Engineering and Technology at the University of Huddersfield for his tremendous support during the entire thesis period. The first meeting with him on a coffee made me feel comfortable, and I was confident that I was in right hands. He has given me the freedom to do the things in my way but always kept me cautious, so I do not deviate from the core of my research. He pushed me beyond my capabilities with his honest opinions and reviews on my research. Without his guidance, I would not have completed the research, and I am very grateful for his assistance.

I want to thank Mick Krembling, the author of the quotation, 'You didn't come this far to only come this far'. The quote kept me motivating when I was lost during my research work.

My acknowledgment will be incomplete if I do not thank my parents. I want to thank Mrs. Naga Lakshamma Gundampati, my mother for believing in me and sending me to this country. Though she does not understand what I study, she always hoped that I should be happy in what I do.

I want to dedicate this research work to my late father Mr. Bala Sundaram Gundampati, who always dreamt of giving me a good education. He sacrificed his life for the sake of our family and tried his best to give us a good life. Thank you, Father.

Chapter 1

1.1 Introduction

Railways are one of the proficient means of traveling from one place to another safely and securely. The latest forecast reveals that UK Network Rail(NR) is spending a total of £6.5 billion in 2017-18, which represents 0.8 percent of total public spending [1] However, due to the improper conditioning of rail and its infrastructure, the safety of the passengers and the goods is compromised. Moreover, it is risky and complicated to monitor the condition of rail infrastructure in inaccessible places and harsh weather conditions. Besides, the status of the infrastructure can be identified or predicted only when the infrastructure is monitored continuously. The effective rail management system plays a key role in upgrading, developing and expansion of the railway industry at low budget cost and is mainly dependent on the intelligent and smart infrastructure.

A simple condition monitoring uses a sensor to monitor the condition of railway infrastructure [2]. The sensors gather all the information from the railway machinery, and when the data reaches or crosses the predetermined level or abnormal conditions, an alarm is activated. However, this simple condition monitoring may lead to false alarms and may even miss the failures. However, in Automated Condition monitoring, granular information of the infrastructure is obtained by building a closed network with sensors and the data is gathered collectively and shared with the adjacent nodes to create an intelligent architecture. The collected data is shared and is accessible from remote places wirelessly. Automated Condition Monitoring could play a vital role in the effective management of rail infrastructure and with the latest developments in wireless technologies, MEMS sensors an effective and efficient Wireless Sensor Network architecture can be built with smart intelligence algorithms.

The proposed architecture in the research work consists of Wireless Sensor Network (WSN) built with National Instruments hardware namely, NI – 3226, NI – 9791 on LabVIEW Development system, pre-processing circuit and a base station to monitor the status of the signal. In Chapter 1, the project is introduced, and a WSN is proposed by studying the background and by performing a literature survey. The complete research work is completed in Chapters 3, 4 and 5. In Chapter 3, the frequency analysis of the given input sinusoidal signal is performed by conducting some trial experiments using the hardware. Chapter 4 and Chapter 5 constitutes the crucial phase of the research work that is carried out. In Chapter 4, the accelerometer signal obtained from the sensor is analyzed by developing and evaluating the MATLAB script. Also, a flowchart is provided to demonstrate the flow of the process. In Chapter 5, the communication is established between the MATLAB and Arduino Uno and the pre-processing circuit is built by using R2R Digital-to-Analog-Converter (DAC). At the end, the acceleration voltage signal is received at the base station and compared with the input to find the real errors. All the results are displayed in the form of tables and figures for easy understanding.

1.2 Background

The condition of the UK's infrastructure has gained expanded attention, principally because of the catastrophic events for instance, the collapse of the third busiest I-35 bridge in Minneapolis in 1st August 2007 and the destruction of the Morandi motorway bridge in Genoa, Italy on August 14th, 2018 resulted in number of deaths and the causalities. However, the degrading and the deteriorating infrastructure has burdened the transportation authorities and other stakeholders for many years. In the long run, bridges continue to age, and the funds provided for the replacement and repairing of the infrastructure are insufficient at present budget levels. According to the Royal Automobile Club Foundation for Motoring (RAC) foundation survey and Stagecoach group, there are almost 74,000 bridges in the UK local road network and nearly 40,000 bridges and tunnels in the UK rail network respectively [4]. Table 1, shows that most of the accidents in the railway system are occurring due to the direct or indirect involvement of the human error. Some facts relating to the accidents occurred due to human error is shown in Table 1. From the table, it is clear that most of the accidents are caused by human error. Below are some of the significant areas where the occurrence of human error in the railway operations is noticed.

- signal passing
- train speed
- signaling or dispatching

Table 1 Data on accidents due to human error across the world [3]

Time Period	**Location**	**Deaths**	**Injuries**	**% of the accidents human error involvement**
2000-2004	United States	4623	51597	-
1900-1997	United Kingdom	-	-	70% of 141 accidents
1970-1998	Norway	-	-	62% of 13 accidents
Annually	India	Over 400		66% of accidents (per year)
1988	Clapham Junction, UK	30 due	69	100% (wiring problem)
1999	Canada	-	-	100% (using outdated equipment)
2005	Bangkok, Thailand	-	200 injured	-
2005	Pakistan	133	Three train collision	-
1999	UK	31	227	-
1989	Purley, UK	5	88	100%

In Condition Monitoring process the specific parameter of condition in the machinery is monitored, for instance, vibration, temperature, acceleration, etc., to identify the critical change which is demonstrative of creating a fault. In contrast to the conditional assessment, the conditional monitoring takes into account of the broader set of granular data that incorporates sensor information from the asset, past inspections, different segments of similar type, location and condition of the plant and authentic trends [5]. The analysis on granular data gives the present status or condition of the component, and it predicts the future issues and the time they are likely to happen. Once, the issues are identified the precautions are taken, for instance, replacing the component when it is worn out. The primary benefits of conditional monitoring include reduced maintenance costs, maximized production, an extended lifetime of machinery, optimized inventory for spare parts, accurate and relevant data for driving product development [5]. In railways, the conditional monitoring helps to improve the performance and availability of the railway assets. Each manufacturing company has its unique pattern or method of performing condition monitoring. However, there are some most regularly used Condition Monitoring (CM) techniques and are:

1. Vibration Analysis and Diagnostics
2. Lubricant Analysis
3. Infrared Thermography
4. Acoustic Emission (Airborne Ultrasound)
5. Ultrasound testing (flaw testing)
6. Motor Current Signature Analysis (MCSA)
7. Model-Based Voltage and Current systems (MBVI)

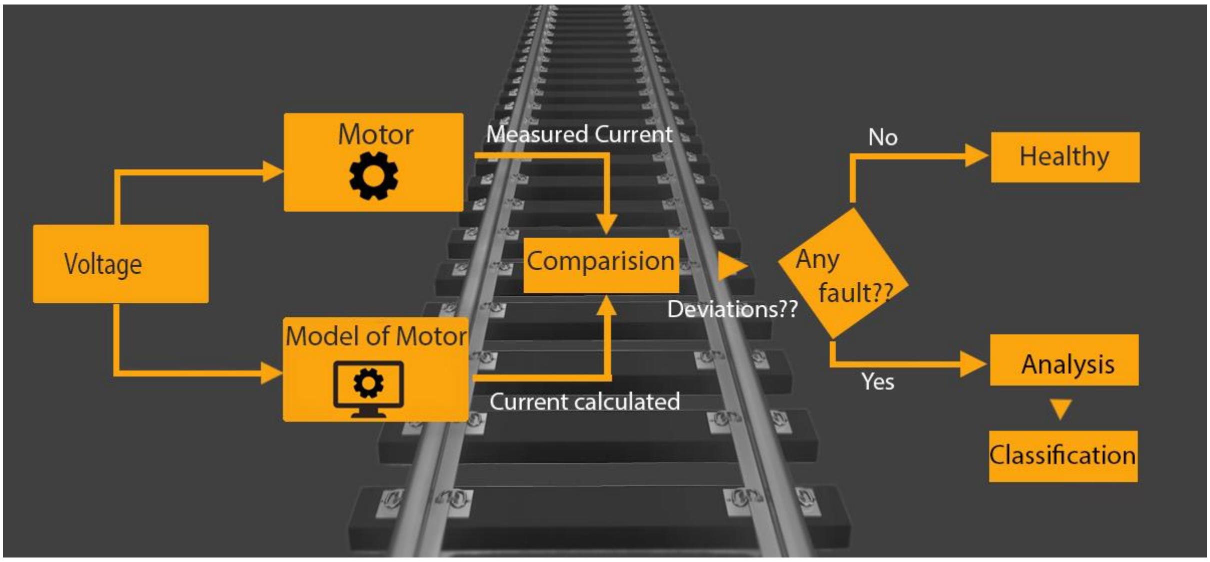


Figure 1 Condition Monitoring using MBVI technique adopted from [5]

In the flow diagram of Figure 1, a specific voltage is sent across the motor; the current is measured and compared to that of the mathematical model, which is fed with accurate real-time data from the same motor. The two readings are checked with the comparator, and the deviations are recorded. Based on the deviation conditions, for instance, if right which means there are some discrepancies between the mathematical model and the physical motor, and then the flow diagram meets the analysis stage where the cause of the problem is found. After the problem is identified, it is then classified, and the solution is applied.

In the above demonstration, the idea of condition monitoring is observed. It makes sense to monitor the motor and record its status, instead of performing only a temporary diagnosis check. Thus, the historical trends are continuously or periodically observed and captured, demonstrating how the mechanical and; electrical problems and their parameters change over time.

Condition Monitoring at the advanced level is classified into two types namely: Wireless-Route based condition monitoring and Continuous Remote condition monitoring. Wireless, route-based condition monitoring employs different sensors on the assets that one must monitor. The technicians within the range of sensors wirelessly retrieve the information. In contrast to the traditional monitoring solution with no cables and wires, the technicians monitor and record each measurement from the sensor simultaneously. The technician moves onto the next asset and records the measurements. Thus, accurate measurements in less time are recorded and can be exported using wireless diagnose machines. It is helpful for the companies that require a short-term and immediate diagnostic solution in less time. However, this process is not recommendable for long-term monitoring and does not provide a broader view of their assets. In Figure 2, the technicians' route themselves towards their assets to monitor the condition. The application-oriented software on the tablet or the device that is connected wirelessly with the sensor retrieves the information from the asset. Wireless Route based CM is very helpful for predictive maintenance on a periodic basis. However, this method cannot provide the granular details on a long-term and continuous basis.

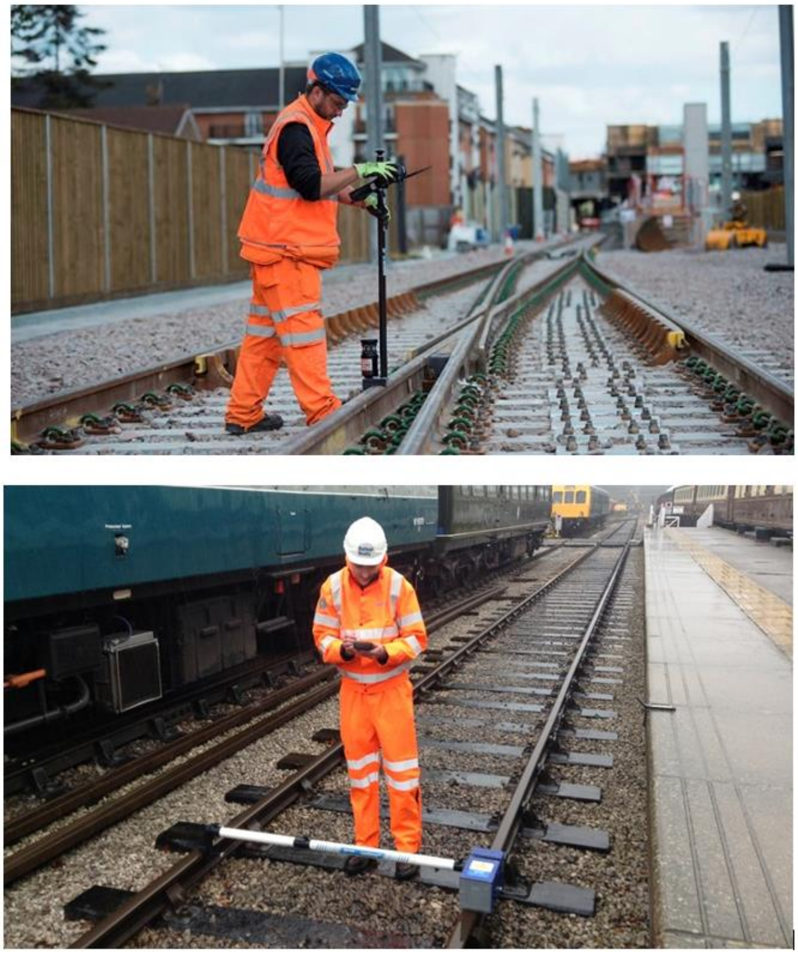


Figure 2 Wireless Route-based condition monitoring [6] and [7]

The continuous Remote monitoring process is fabricated for long-term monitoring of their assets and give immediate access to the condition or health status of the asset without disturbing its function. Unlike wireless route-based monitoring, where a technician is necessary to record and within the range of a sensor, the users with the internet access can monitor and take the measurements of the assets with remote or cloud-based condition monitoring from anywhere in the world. In Figure 3, the sensors are deployed on different assets for each application of condition monitoring, for instance, obstacle detection sensor in level crossing, IR sensor for track fail detection, etc., These sensors are connected to the wireless sensor nodes A, B, C, D and E, and the nodes are connected to the WSN gateway wirelessly using IEEE 802.15.4 protocol. Besides, all the nodes are connected in a single WSN using the Mesh network. The gateway is connected with the base controller via Ethernet cable (blue). Using the host controller that is connected with a cloud database, the user can access the data or monitor the condition remotely. Also, using the same host controller, the user can alter any changes and can be imposed on the sensor node without disturbing its operations.

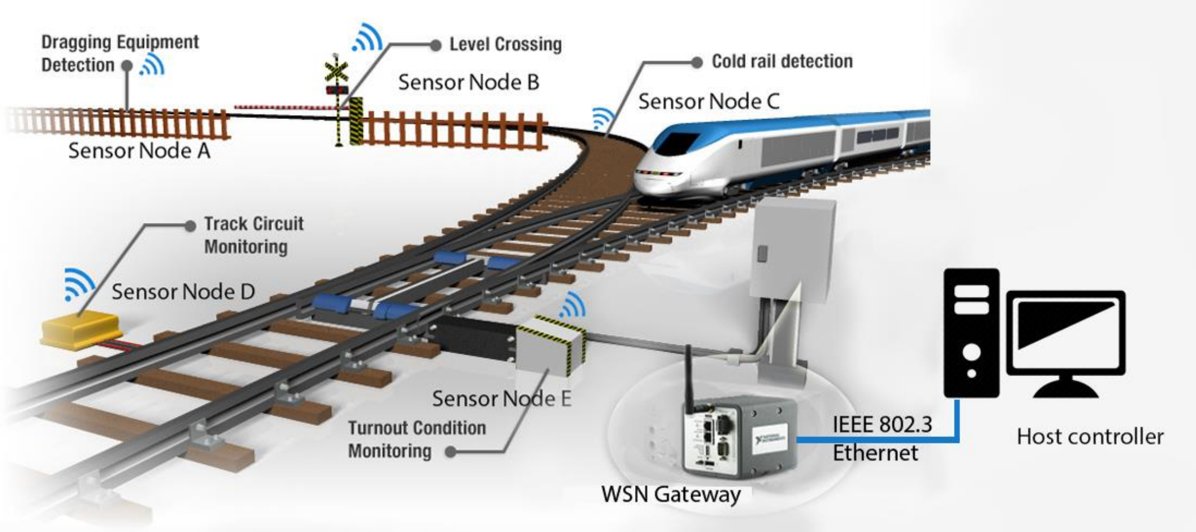


Figure 3 Remote condition monitoring of the rail system adopted from [8]

1.3 Literature Review

1.3.1 Condition Monitoring

Condition Monitoring (CM) is widely accepted and used in the rail industry to prevent failures in railway operations. One of the reasons for failures in the railway operations is the improper management and unanticipated damage to infrastructure. CM is the concept used in various industries to prevent asset damages. CM can be an effective method to determine the status or condition of the asset while it is still in operation. By knowing its state, one can intercept the incipient fault and can implant specific solutions onto the asset to prevent failures.

1.3.2 Remote Condition Monitoring

Wireless Sensor Network (WSN) can be defined as the network of devices that can communicate and transfer the information collected from a monitoring area through the wireless connections. The technology of WSNs is used to overcome the limitations of

Wireless- Route based CM limitations like continuous monitoring, remote monitoring collecting granular information in less time. Remote condition monitoring (RCM) is a method or a practice of using sensors and software to monitor the condition and record the status of the asset. The advances in sensor technology that are integrated with cloud-based data delivery are making the pre-established route-based CM practice to more cost-efficient, accessible and useful. RCM can provide real-time and continuous data acquisition results; effective data management; increases the monitoring frequency; improves data integrity and accessibility; integration of combined data from sensors at any point of time; enhances the durability of data; initiates 'predict and prevent' policy for intelligent analysis of the data using intelligent algorithms; ability to filter data and give specified information [2]. The typical WSN comprises of a distributed network of sensor nodes that are wirelessly connected with the gateway. With rapid improvements in CMOS technologies, camera and microphone have become an integral part of WSN [9]. Hence, the integration of image sensing and image processing into WSN by CMOS technology improves the overall performance of the system. Moreover, with the improvements in CMOS fabrication processes and packaging technologies, helps to plant many transistors on a single wafer and thus reducing the overall size of the node, making it cost effective and power efficient.

In Figure 4, the dotted circle is the wireless sensor network formed by the spatially distributed sensor nodes. The data retrieved from sensors is sent to the gateway using wireless communication and using the wireless protocol ZigBee, IEEE 802.15.4.

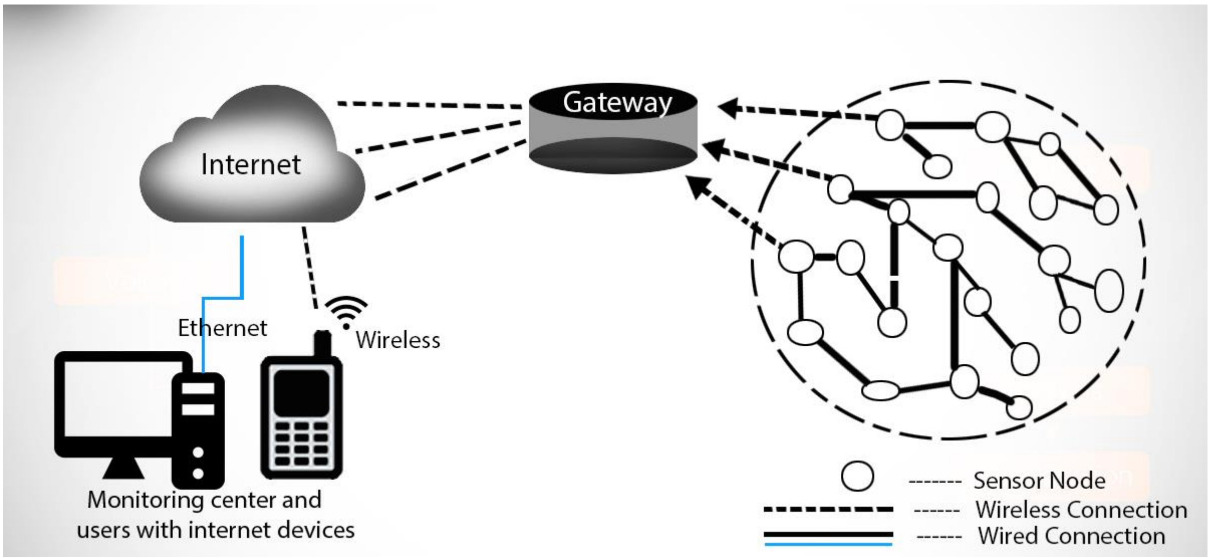

Internet
Gateway
Ethernet
Wireless
Monitoring center and users with internet devices
------- Sensor Node
------ Wireless Connection
------ Wired Connection

Figure 4 Typical WSN architecture for Remote Condition Monitoring

In the first instance, IEEE 802.15.4 is aimed at low-power, low-rate and short-distance personal wireless area networks (WPANs). However, this is now an encouraging contestant for correlations between wireless sensor nodes. IEEE 802.15.4 makes use of altered Carrier Sense Multiple Access with collision avoidance (CSMA/CA) protocol to gain channel access for data transmission; therefore, different nodes can access the same channel at a different time without any interference [10]. The gateway is connected with the internet and allowing the information to be stored on the cloud. The user with the internet access either by wired Ethernet or by wireless mobile data can access the data and can monitor the infrastructure as long as the user is connected to the internet.

There are many technologies and many primary research tools available in the market to monitor the condition of infrastructure remotely however they tend to be costly. To access the health of infrastructure continuously and comprehensively, one must look for the more cost-effective continuous approach. The MEMS technology has gained much reputation in the field of remote condition monitoring due to its small compact size, low size and a wide variety of application-oriented sensors. MEMS sensors based on MEMS fabrication technology are the standard sensors that are made-up of micro-mechanical and electrical components embedded on a single chip. Figure 5, is the network architecture of a typical sensor node in a WSN. MEMS sensors are connected to the AI/O pins of the WSN node. The analog signals from the sensors are converted into digital using an Analog-to-Digital converter; the ADC is connected to micro-controller for further processing and flow of data is bilateral at micro-controller. The node can be used in two different ways: end node and the router. When it acts as an end node, the transceiver sends the information to the end route, and when it acts as a router, it sends and receives the information. The power for the sensor and the signal processing is supplied by the power source and is provided externally (5-30) V or internally by batteries (1.5V).

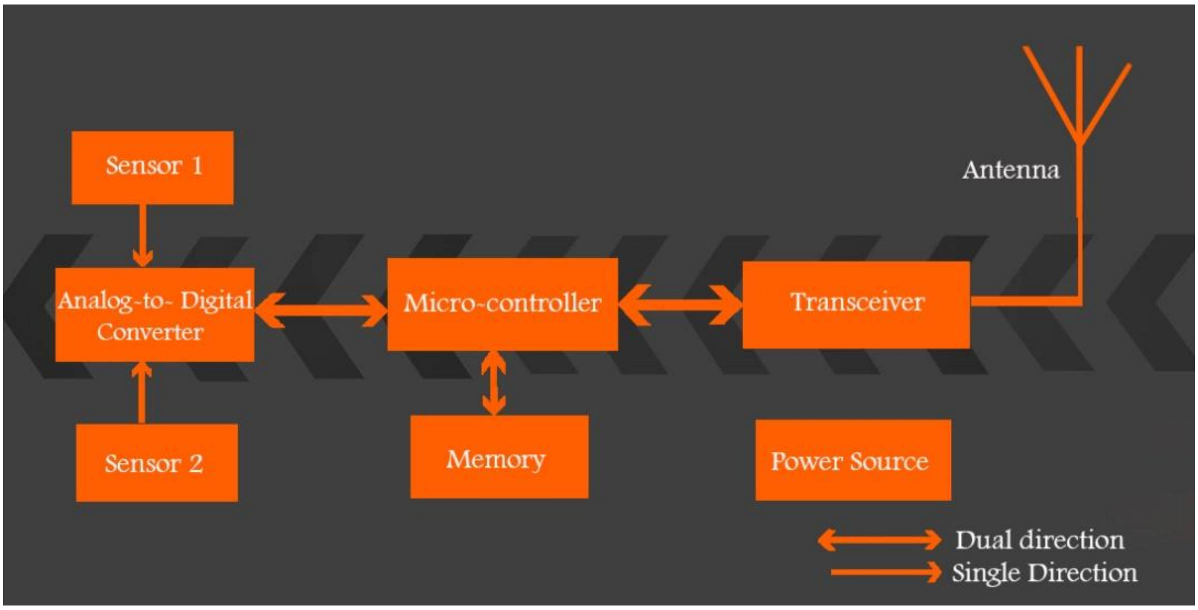


Figure 5 Sensor Node architecture adopted from [2]

Destruction in the critical railway structures, for instance, railway bridges cause massive damage both to the railways and to the humankind. Many authors of various papers have demonstrated and deployed WSNs to monitor the condition of railway bridges. In [11], the commercial Tmote Sky WSN platform is implemented to monitor the railway bridge in Sweden. Eight WSN nodes are used in the WSN, and the root node is connected to the base station. A sending policy is created to avoid packet collision between the nodes where each node transmits the data one particular time slot given to it. Tmote Sky platform collects the information on the status of the bridge, i.e., deformations in the infrastructure when the train passes through the bridge. The authors have used Bosch Sensortec SMB380, a low-cost tri-axil sensor to detect the event of the arrival of the train and to record the acceleration of the train.

Few train accidents are being noticed at the level crossings are notably observed in under-developed and developing countries. Until the date, most of the signaling operations at level crossings are monitored and operated by humans, and a slight error would lead to too many deaths. A WSN is proposed in [12], to control the railway signaling. Different sensors are connected hierarchically to achieve maximum efficiency and reliability. The IEEE 802.15.4 communication protocol is used among the sensors to gain both the flexibility of implementing multi-hop and redundant paths. From the gateway or the access point, the data is sent to the base station or control center using technologies like GPRS, Wi-MAX.

In [13], a prototype on automated monitoring and control system is developed using ZigBee wireless communication. Two infrared sensors are used to detect the arrival and departure of the train. To ensure the reliable transmission of data from source to destination, the authors have proposed an optimal recovery path routing algorithm based on ZigBee protocol in Application Programming Interface (API) mode. The authors are successful in implementing the ZigBee protocol in designing automated, monitored system, but they could not take complete advantage of WSN to perform Remote Condition Monitoring.

The concept of WSN technology is deployed to monitor the integrity of freight trains in [14]. Sometimes, the wagons become isolated from the locomotive for the wrong reasons. This event is monitored using different sensors. Accelerometer and vibration sensors are used to monitor the condition, and when an unexpected change is observed, a false alarm is raised. Thus, the WSN technology replaced the existing trackside systems to check the trains' integrity.

High-speed rail (HSR) network trains in China, TGV Lyria of Europe, the Swedish SJ trains and many more trains in the world travel with the speeds of 200 – 350 km/h [15]. These high-speed trains may lead to massive deaths if they face any severe safety challenges for instance earthquakes. Even small-scale earthquake can be extensive damage for railways. To overcome this, in [15], a WSN is proposed by a group of authors to detect the prior occurrence of earthquakes and security control in railways. The authors stated that the WSN of Early Earthquake warning system (EEW) and WSN of railway health monitoring system work together.

1.3.3 Network Model

Open Systems Interconnection (OSI) model is the most common network model used in wireless sensor networks of railway systems and its applications. The typical WSN protocol stack is shown in Figure 6.

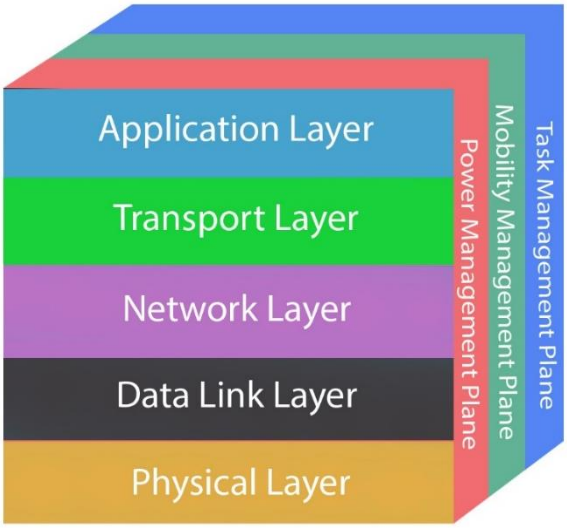


Figure 6 Sensor Network Protocol Stack

The stack has five layers in total and with three planes to transmit the data from the sensor nodes to the base station. On the application layer, different types of application-oriented software are built and are used by the end-user. Transport layer regulates the back and forth flow of data from the sensor node. The network layer routes the data from the transport layer in the form of packets. The data link layer has Medium Access Control (MAC) protocol and

Logical Link Control (LLC) as its sublayers. In the case of a noisy environment and sensor node mobility, there is a probability of collision of broadcasting packets. The MAC protocol avoids packet collisions and controls data access. The LLC protocol controls the error checking and provides data synchronization. The data link layer with its two sub-layers plays a crucial role in railway operations. For example, when the trains that are acting like data mules, when the trains are moving in the tunnels or thick forests, the WSN railway monitoring addresses problems like asynchronous data communication, packets collision, etc... In [16], Cañete et al., have proposed a monitoring system that measures the vibrations and displacements in the track using well-built communication architecture to overcome the problem of data collision and asynchronous data communication. The sensors transmit the data to the trains passing by. The train was acted like a data mule uploading the information to the remote control center and finally drops the data at a database of the base station. An application at the base station investigates the data to detect and predict any future failures. The physical layer determines the path of flow of data from an address to another address. Three planes manage the coordination of sensing tasks of different sensor nodes and lowers their overall power consumption. Power management plane leads the power of sensor node, for instance, time to sleep, time to awake, to send a message to the router when its power is low, to sleep after receiving information from its neighboring node to prevent duplication of information, etc. The mobility management plane records the addresses of the nodes when there is a change in their location. In addition, each node is aware of their neighboring nodes. In the situation like, when the adjacent node is too far away from a node, the node goes to sleep and stops sending the information. Task Management plane is responsible for balancing and scheduling the sensing tasks in a specific region. Depending on the power levels and its location in the particular area, the task management plane assigns a task to each node. For example, when sensor A and B are in a specific region, but the power level of A is higher than that of B, the plane chooses sensor A as the priority and assigns multiple tasks.

13.4 MEMS Sensors in RCM

The sensors used in the Remote Condition Monitoring of railways falls under the category of MEMS. A large scale of MEMS sensors is available in the market to accomplish the needs of various industries, for instance, sensors like accelerometer, gyroscope, temperature, humidity, air pressure, airflow direction and velocity, GPS and many more. These small and compact integrated devices are built with electrical and mechanical components. Moreover, these are cheap, efficient and occupy less space in the deployment [17]. Hence, MEMS sensors can be deployed in large scale even at demanding environments in less time with greater confidence. Sensor nodes employed with sensors are pre-built and embedded with the data processing algorithms for autonomous execution. In RCM, the sampling interval determines the life of the power source, for instance, while monitoring the growth of a crack over a certain period, the power requirement will be low, and hence, the life of power source is extended beyond that of the residual life.

Milne et al., in [18], performed laboratory trails and direct comparison of trackside measurements using ±16 g MEMS accelerometer and geophone. With the comparison of results, they have provided sufficient evidence to demonstrate the effectiveness, ease of deployment and the adoption of MEMS devices in the railway applications. Hence, the lower cost and greater robustness of MEMS enables the deployment for continuous trackside monitoring.

In the Structural Health monitoring of the Ben Franklin Bridge on the Delaware River, a star network of ten strain sensors were deployed to operate in the low-power sampling mode for train detection on the bridge with a low sampling frequency of approximately 6 Hz. The collected strain waveform data is stored onto the flash memory on the wireless sensor nodes [19]. The author noted that the power required for event detection was less than one mA while it was 30 mA for continuous operation. From the above work, it is quite clear that specific relevant information of the signal can be stored on the flash memory of the sensor node of the corresponding WSN for further processing in the future.

In [20], the authors used piezoelectric strain sensors using accumulators, and the photovoltaic modules permanently charge these accumulators. Solar energy can be used as an alternative source of power, but the drawback lies in frequent cleaning for the maximum performance. Moreover, the sunlight is limited to a few months, and at some places, it is hard to generate due to adverse weather conditions. Hence, the authors of [21] and [22] used solar power as just a backup. The sensors are put into sleep mode and are wake-up once an event is detected and this reduces the energy usage. This approach is termed as wake-on-event two-layer event detection. One sensor detects the event is happening while the other sensor wakes up the first sensor using interrupts [2].

1.4 Conclusion

The authors in the above all papers and all-conference proceedings have exploited the technologies like MEMS and RCM on different structures and infrastructures to perform various sensing operations. The following conclusions are noted down from the literature review.

1. The authors have used different varieties of MEMS sensors to build a WSN

2. The authors have proposed, developed and implemented different WSN architectures

3. Few energy-harvesting techniques are introduced and designed to provide sufficient energy to the sensor node

From the survey, it is evident that all the authors have proposed and developed similar WSN architecture and is shown in Figure 4; however, they have implemented with different MEMS sensors on their corresponding sensor nodes to accomplish their tasks. Although various WSNs are proposed using IEEE 802.15.4 ZigBee protocol, they are not very successful in implementing it.

In the current project, the WSN architecture and the sensor node architecture shown in Figure 4 and Figure 5 respectively are used. Besides, in contrast to the proposed harvesting energy techniques, the signal is pre-processed by a microprocessor or by any pic-controller before sending it to the WSN architecture. The pre-processing circuit reduces the size of the raw signal data packets to a minimal size. The energy consumed by the minimal data packets is less when compared to the energy consumed by the entire raw signal. The aims and objectives of this paper are achieved with the knowledge and usage of the above techniques and technologies.

Chapter 2

2.1 Aims and Objectives

Human negligence, human errors and delay in human inspection at the sites are the primary reasons for most of the accidents across the world. With the recent developments in the area of wireless communications and sensing technologies, many disastrous accidents can be prevented. The sensor nodes are placed at inaccessible locations for humans and are supplied with minimum power supply using batteries. Hence, to make use of the batteries for longer time, the load on the nodes should be kept minimum or an alternative power source should be provided as a backup.

The main objectives of the project are:

- To perform a literature survey on WSN technologies and their corresponding network models
- To identify the perfect WSN configuration by performing trial experiments on the WSN hardware
- Signal Conditioning of received signal from the MEMS sensor
- Establishing communication between the MATLAB workspace and Arduino Hardware
- Identify the suitable pre-processing circuit required for transmitting information on MATLAB to the base station

The primary focus of the project is

- To build the righteous configuration of WSN system
- To understand the limitations of the National Instruments (NI) Hardware
- To retrieve and investigate the relevant information from the MEMS sensor and sending it to the base station via WSN system
- Receiving and logging the information at the base station
- Comparing the information received at the base station with that of the input signal to verify the efficiency of the WSN configuration

In Chapter 3, some trail experiments are conducted to verify the configuration of WSN’s node architecture, by changing the sampling intervals (in seconds) on the WSN node. Similarly, in Chapter 4, the signal conditioning is performed on the retrieved raw acceleration signal and the relevant information is recorded. The information is then processed into digital format in the pre-processing circuit. In Chapter 5, the digital information obtained inside the pre-processing circuit will be further converted to analog format using the R-2R ladder circuit and later sent to WSN. In section 5.3.3 of Chapter 5, the signal received at the base station is compared with the input signal and the percentage of error is tabulated and discussed.

Chapter 3

3.1 Introduction

The hardware and software play an essential role in the manufacturing of Embedded and Computer systems. The WSN architecture is entirely dependent on the configuration between software and hardware. In this experiment, devices from National Instruments like NI WSN-3226 node and NI-WSN 9791 gateway and software like NI-LabVIEW and NI-MAX (Measurement and Automation Explorer) are used to build the WSN system. Before installing the hardware in the external environment, it is initially advisable to use the hardware internally so that the difficulties and limitations are understood. Moreover, to use the hardware in different modes and to extend its capabilities a careful examination of its strength and limitations is performed.

In this Chapter, the system is tested and verified by using the above-mentioned hardware and software systems. The objectives of this chapter include:

- To build a righteous configuration of WSN
- To verify the WSN system by performing some trial experiments

3.2 Description

Figure 7, is the block diagram of Chapter 3. As seen, the input signal is generated from the signal generator with a peak-peak voltage of 4 – 5 volts. The frequency and amplitude of the signal are monitored and recorded using USB – 6008 on the LabVIEW Development system that is pre-installed on the local computer and is termed as a reference input signal. A wireless sensor network is configured using WSN – 3226 and WSN – 9791 on the LabVIEW Development system at the base station (local computer). In addition, Figure 8 shows the hardware manufactured from National Instruments that is used for creating WSN.

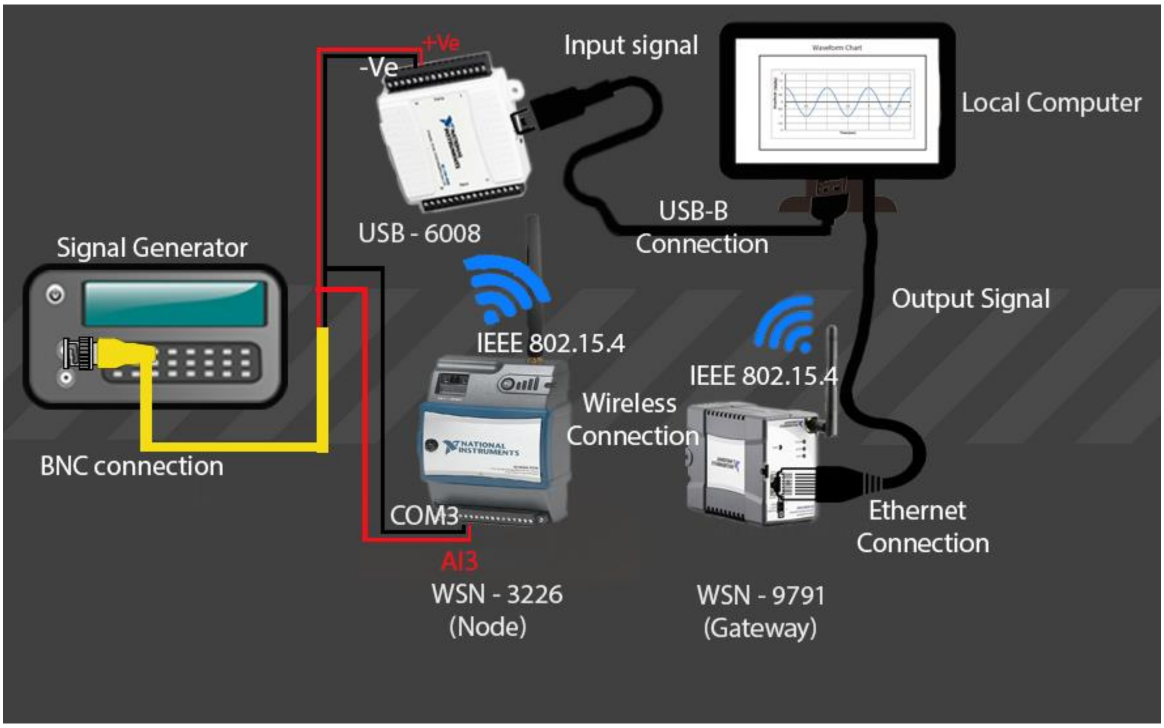
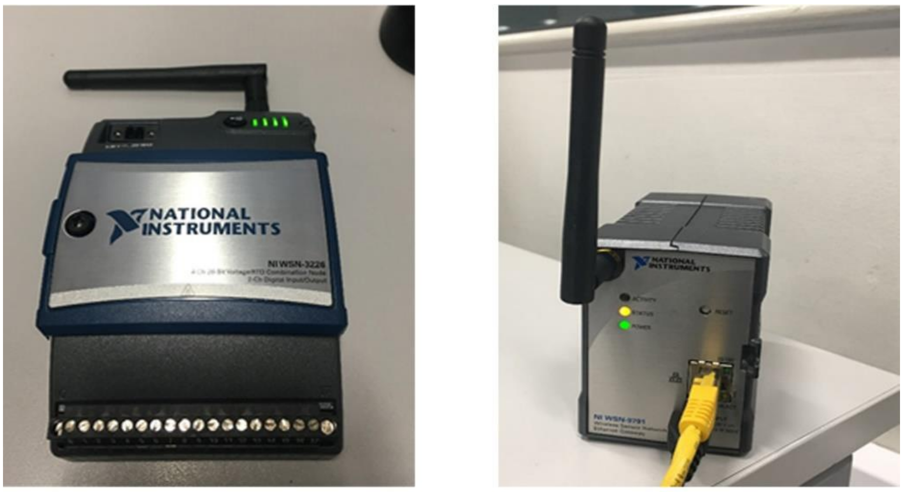


Figure 7 Block Diagram of Phase 1



NI - 3226

NI - 9791

Figure 8 National Instruments hardware

The output signal is acquired and monitored on the LabVIEW environment (local machine) using the WSN network. Later, the output signal is then verified with that of the input signal by plotting against each other using MATLAB. Two successive peaks of the input and output are marked to calculate the frequency of the input and output signals. If both the input signal and the output resemble similar characteristics like the same amplitude and frequencies, then it can be understood that a secured wsn wit correct configuration is built.

3.2.1 WSN Architecture

In Figure 9, the output signal from the analog input terminal (AI3) of WSN-3226 is received and monitored on the waveform chart 2, and the samples are recorded in the Excel Technical Data Management Streaming (TDMS) file using Write To Measurement File block. The input signal is sampled at the sampling interval of one sample per second. Also, when the sampling interval is set to one second (1Hz), the radio on the node is also powered up at the same sampling interval. Similarly, the samples of the corresponding input signal are also acquired using USB – 6008 DAQ Assistant of the LabVIEW system and are stored in another Excel TDMS file. The timing for N samples is the time (in seconds) taken to receive N samples at the base station.

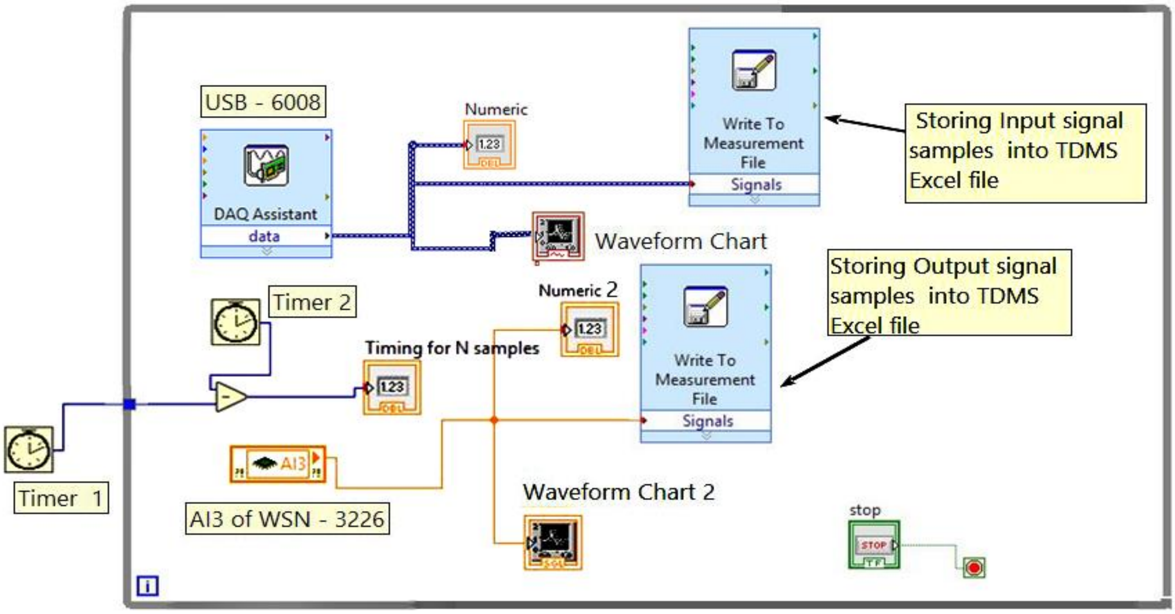

USB - 6008
DAQ Assistant
data
Numeric
Write To Measurement File
Signals
Storing Input signal samples into TDMS Excel file
Waveform Chart
Timer 2
Numeric 2
Storing Output signal samples into TDMS Excel file
Timing for N samples
Write To Measurement File
Signals
AI3
Timer 1
AI3 of WSN - 3226
Waveform Chart 2
stop
STOP

Figure 9 WSN architecture on LabVIEW environment

3.3 Frequency Analysis of a sinusoidal signal

3.3.1 Case 1:

An input signal of 1.05 Hz is generated from the analog signal generator and is passed through the WSN architecture via WSN – 3226 node with the sampling period/sampling interval of 0.025 seconds (sampling frequency of 40Hz). The output signal received at the base station is monitored and recorded into a measurement file. The input signal is compared with the output signal, and the changes in the output signal are noted down.

3.3.1.1 Results

In Figure 10, the input signal (blue) is plotted against the output signal (Red) on MATLAB. Two successive peak voltage values of both the signals are marked with their respective coordinates to calculate the time period and frequency of the signal.

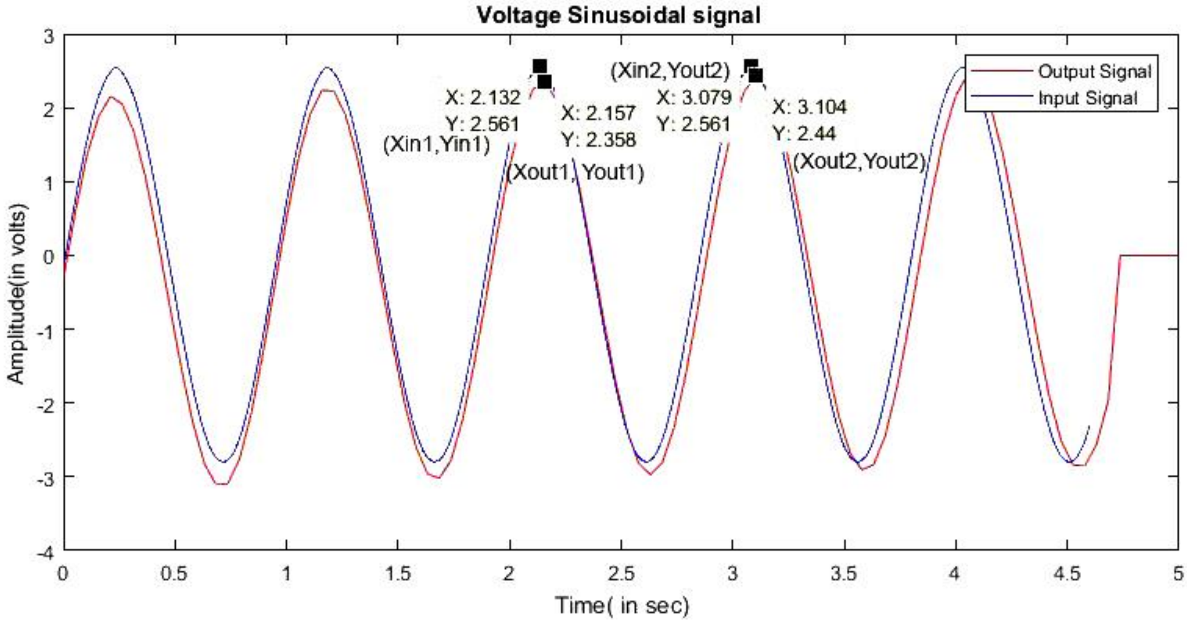


Figure 10 1Hz Input signal Vs. Output signal at the sampling period of 0.025 seconds

Calculations:

Input signal

Table 2 Frequency of the input signal

Xin1 (volts)	Xin2 (volts)	Time Period, T1 (seconds) Xin2-Xin1	Frequency, f1 (Hz) F1 = 1/T1
2.132	3.079	0.95	1.05

Output signal:

Table 3 Frequency of the output signal received for the sampling period of 0.025 sec

Xout1 (volts)	Xout2 (volts)	Time Period, T2 (seconds) Xout2-Xout1	Frequency, f2 (Hz) F2 = 1/T1
2.157	3.104	0.947 ~ 0.95	1.05

Table 2 and Table 3 displays the input and output signal calculations that are obtained from Figure 10. From the calculations, it is clear that the time period and frequency of the output signal is like that of the input signal. Moreover, without changing its characteristics, the input signal is received at the base station via the WSN network. Hence, it can be concluded that the WSN architecture is correctly built. Moreover, the experiment is repeated for various intervals to verify the efficiency of the WSN system and to observe the behaviour of WSN system.

3.4 Frequency Analysis of the signal for various sampling intervals

In this section, the sampling interval (in seconds) is changed at various intervals for the same WSN configuration. is performed by changing the sampling intervals but without changing the configuration of WSN. The sampling interval is altered either by using the I/O node properties on the WSN node or by programming the node using ‘Config Node’ in the LabVIEW WSN target.

3.4.1 Description

The input signal generated from the signal generator, is fed to WSN – 3226, and is received at the base station via WSN. However, this section is divided into two cases:

Case 2: 1.05 Hz input signal for the sampling intervals of 0.1 and 0.25 seconds on NI – 3226

Case 3: 2 Hz input signal for the sampling intervals of 0.0125 and 0.035 seconds on NI – 3226

The received output signals are plotted and compared against the input signal.

3.4.2 Case 2:

1.05 Hz frequency signal with an amplitude of 2.5 volts is generated from the signal generator and is fed to the WSN - 3226 node and USB – 6008. The signal received from USB – 6008 is considered as the input and the signal through WSN – 3226 is considered as an output signal. The input and output signals are monitored and recorded at the base station to perform signal analysis. The sampling interval on the WSN node is changed to 0.1 seconds using the I/O node properties.

3.4.2.1 Results

In Figure 11, the signal in red colour is the 1.05 Hz sinusoidal signal, and the signal in blue colour is the output signal received at the base station via WSN. As seen in Figure 11, the signals are plotted against each other to verify their amplitudes and frequencies respectively. The frequency of both the signals are similar, i.e., 1.05 Hz however, the amplitude (in volts) at the output has slightly differed. Besides, due to the quantization by Analog-to-Digital converter (ADC) on the WSN node the shape of the output signal is also differed somewhat.

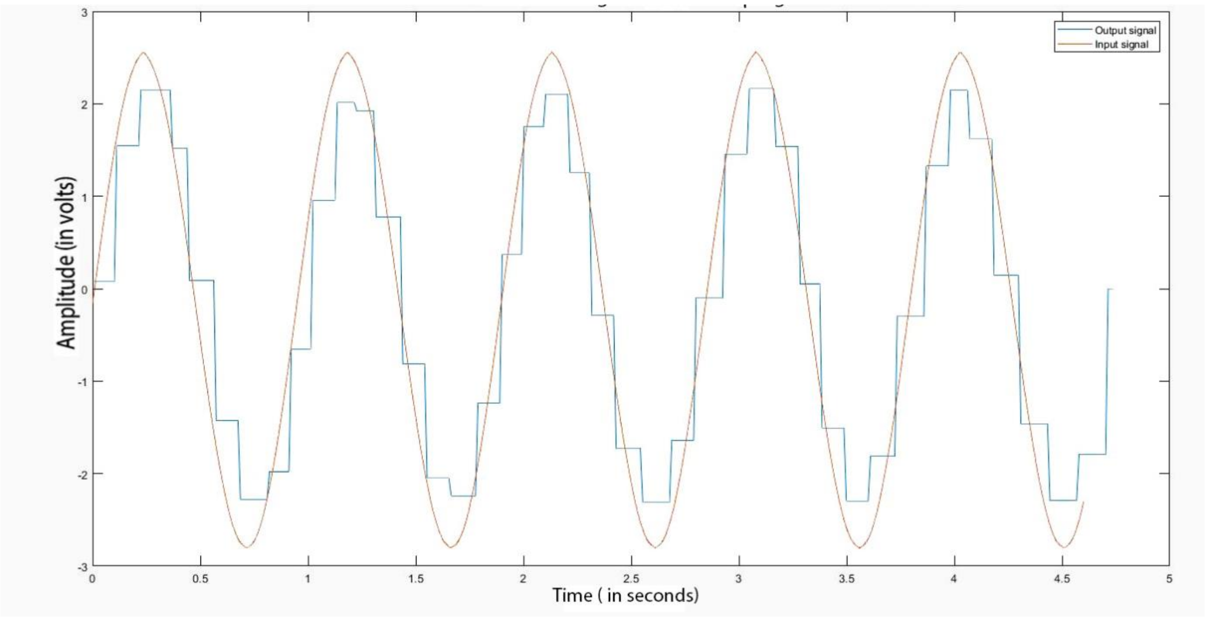


Figure 11 1Hz input signal Vs. Output signal for the sampling period of 0.1 seconds

Similarly, for the sampling period of 0.25 seconds on the WSN node, an output signal is received. Likewise, the signal is plotted against the input signal on MATLAB to compare its characteristics. The MATLAB plot is shown in Figure 12. From Figure 12, the frequency and the time period of both the signals are measured by performing calculations.

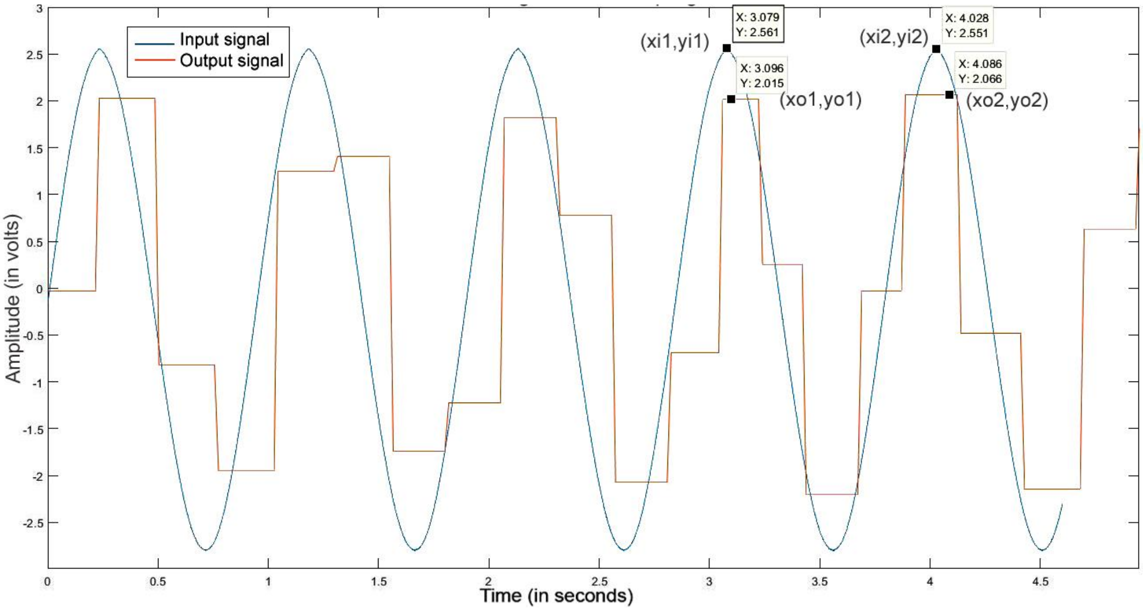


Figure 12 1Hz sinusoidal input signal Vs. the output signal for the sampling period of 0.25 seconds on NI - 3226

Xin1 and xin2 are the input coordinates while xout1 and xout2 are the output coordinates of their two successive peak amplitude values. The frequencies of the two signals are calculated as below:

Input Signal:

Table 4 Frequency of the input signal

Xin1 (volts)	Xin2 (volts)	Time Period, T1 (seconds) Xin2-Xin1	Frequency, f1 (Hz) f1 = 1/T1
3.079	4.086	0.95	1.05

Output signal:

Table 5 Frequency of the output signal at 0.25 seconds sampling period

Xout1 (volts)	Xout2 (volts)	Time Period, T2 (seconds) Xout2 - Xout1	Frequency, f2 (Hz) f2 = 1/T2
3.096	0.6975	1.01	0.99

Table 4 and Table 5 displays the input and output signal calculations that are obtained from Figure 12. By comparison, of f1 and f2 frequencies, it is understood that the frequency of the output signal that is received at the base station is lesser than that of the input signal sent via WSN. Besides, in Figure 12, it is observed that the output signal lost its similarities concerning shape and amplitude when compared to the input signal.

3.4.3 Case 3:

Similarly, a 2Hz is generated from the signal generator and is fed to the WSN - 3226 node and USB – 6008 simultaneously. Besides, the input and output signals are plotted against each other using MATLAB. In this case, the sampling period is changed to 0.0125 on the WSN node.

3.4.3.1 Results

In Figure 13, the signal in blue colour is the input signal, and the signal in red is the output-sampled signal. The frequency of the signal at the output is calculated and verified with that of the input signal. The output signal (orange colour) received at the base station is like that of the input signal (blue colour) concerning its shape and amplitude (2 V)

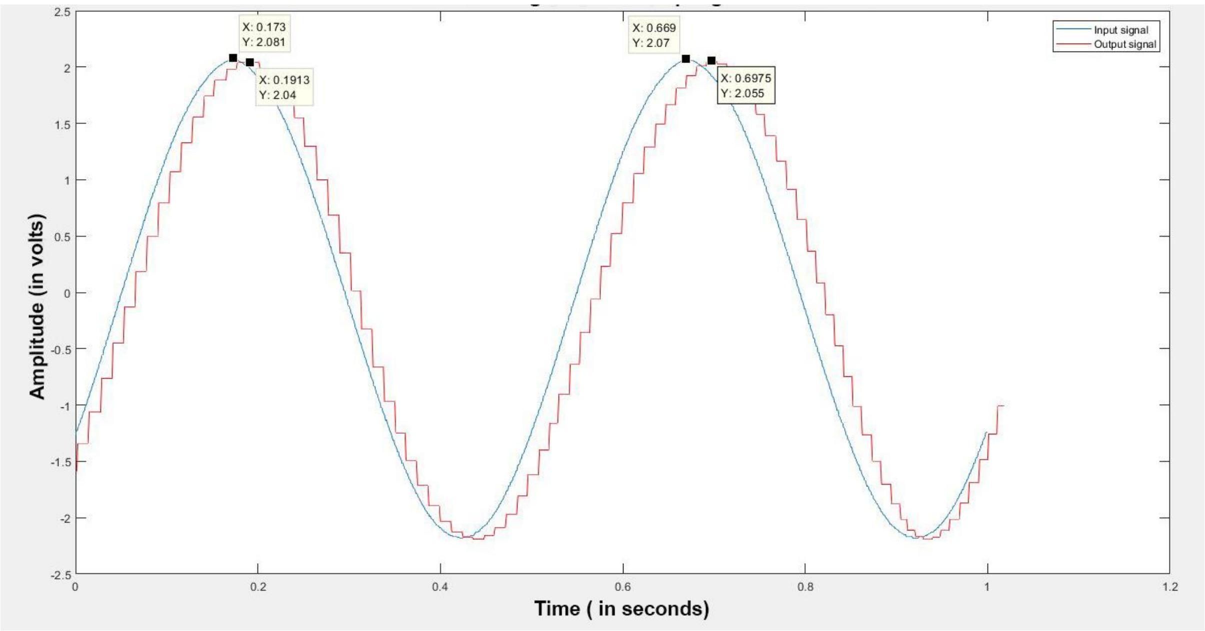


Figure 13 2Hz input signal Vs. Output signal at sampling interval of 0.0125 seconds on NI - 3226

Calculations:

Input signal:

Table 6 2Hz sinusoidal input signal

Xin1 (volts)	Xin2 (volts)	Time Period, T1 (seconds) Xin2-Xin1	Frequency, f1 (Hz) f1 = 1/T1
0.173	0.669	0.5	2

Output signal:

Table 7 Frequency of the output signal at a sampling period of 0.0125 secs

Xout1 (volts)	Xout2 (volts)	Time Period, T2 (seconds) Xout2 – Xout1	Frequency, f2 (Hz) f2 = 1/T2
0.1913	0.6975	0.506	2

From Table 6 and Table 7, it is evident that the input signal is received at the base station via WSN without any change in its frequency and the time period. Likewise, the sampling period is changed to 0.035 seconds for the same 2 Hz input signal and the output signal is received at the base station. The 2Hz input signal is plotted against the output signal as shown in Figure 14.

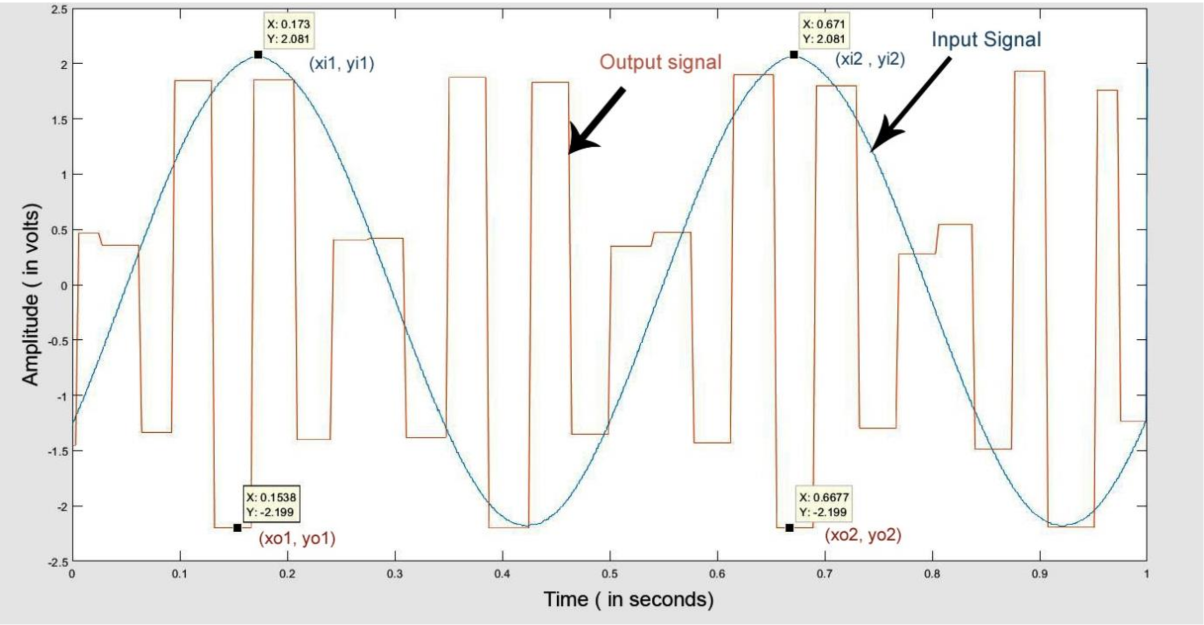


Figure 14 2Hz input signal Vs. Output signal at sampling period of 0.035 sec on NI – 3226

Calculations to find their corresponding frequencies:

Input signal

Table 8 2Hz sinusoidal input signal

Xin1 (volts)	Xin2 (volts)	Time Period, T1 (seconds) Xin2-Xin1	Frequency, f1 (Hz) F1 = 1/T1
0.173	0.669	0.5	2

Output signal:

Table 9 Frequency of the output signal at a sampling period of 0.035 secs

Xout1 (volts)	Xout2 (volts)	Time Period, T2 (seconds) Xout2-Xout1	Frequency, f2 (Hz) f2 = 1/T2
0.154	0.667	0.51	1.961

Table 8 and Table 9 displays the input and output signal calculations that are obtained from Figure 14. In Figure 14, the output signal received at the base station has lost its characteristics like shape and the frequency of the signal. The appearance of the input signal is changed, and the frequency of the output does not match that of the input signal.

3.5 Conclusion

The sampling period was varied on NI – 3226 using its I/O properties. It is observed that the sampling period can be changed from one-millisecond to one million seconds on the WSN node. In Table 10, the results of Case 1, Case 2 and Case 3 are displayed for various sampling periods for comparison purpose.

Table 10 Comparison between input and output frequencies

N^0	Case	The frequency of the input signal (Hz)	Sampling Period (seconds)	Sampling Frequency (Hz)	The frequency of the output signal (Hz)
1	Case 1	1.05	0.025	40	1.05
2	Case 2	1.05	0.1	10	0.99
			0.25	4	1.05
3	Case 3	2	0.0125	80	1.961
			0.035	28.57	2

The following conclusions are drawn from Table 10.

1. Case 1(orange colour) shows the comparison between the input (1.05 Hz) and the output signal frequencies at the sampling period of 0.025 seconds on the node. It is also observed that the frequency of the output signal received is like that of the input signal. Moreover, in Figure 10, the output signal received is overlapped with that of the input signal, and thus a well-built WSN architecture is established.
2. In Case 2 (green colour), the output was received at the base station at sampling periods of 0.1 and 0.25 seconds on the WSN node, for the given input signal of 1.05 Hz. From Table 5, it can be seen that the frequency of the output signal (0.99 Hz) slightly differs with that of the input signal at the sampling interval of 0.1 sec. However, at 0.25 seconds of sampling interval, the output frequency (1.05 Hz) matches with that of the input signal (1.05 Hz). Furthermore, in Figure 11, at 0.1 sampling interval, the amplitude of the signal was decreased, and the shape of the signal was differed.
3. In Case 3 (yellow colour), for the 2Hz input signal, the frequencies of the output signal received for two sampling periods namely, 0.0125 and 0.035 seconds are 1.961 and 2Hz respectively.

In all the cases, it is observed that the output signal was received at the base station successfully and hence, it proves that the built WSN architecture is ideal for the proposed

WSN system. It is also observed that when the sampling frequency/sampling rate is larger than the twice of the input maximum frequency, the output signal received at the base station is an exact replica of the input signal. This can be observed for the sampling frequencies of 80 Hz and 40 Hz in Table 10.

Chapter 4

4.1 Introduction

In Chapter 4, the proposed project is completed by performing various tasks and these are broken down into different sections. Figure 15, is the flow chart describing the steps in Chapter 4. A script is developed on MATLAB for each section, and the results are displayed in their corresponding sections.

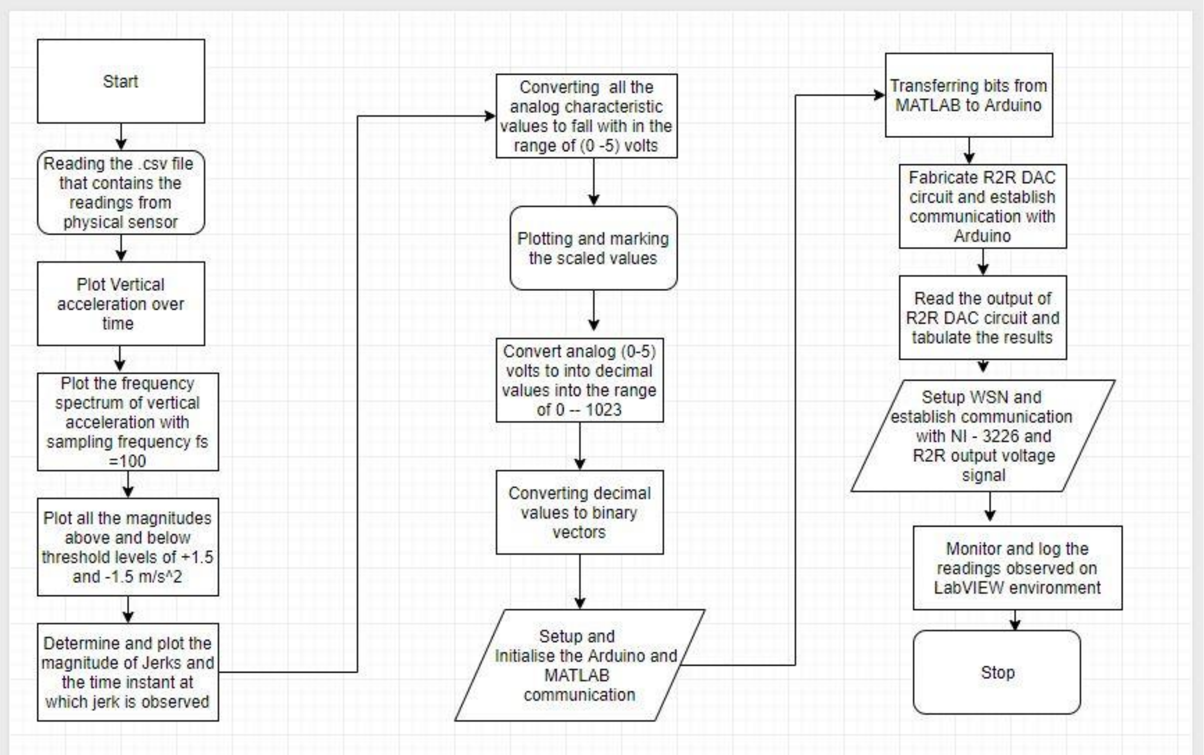

Start
Reading the .csv file that contains the readings from physical sensor
Plot Vertical acceleration over time
Plot the frequency spectrum of vertical acceleration with sampling frequency fs =100
Plot all the magnitudes above and below threshold levels of +1.5 and -1.5 m/s^2
Determine and plot the magnitude of Jerks and the time instant at which jerk is observed
Converting all the analog characteristic values to fall with in the range of (0 -5) volts
Plotting and marking the scaled values
Convert analog (0-5) volts to into decimal values into the range of 0 -- 1023
Converting decimal values to binary vectors
Setup and Initialise the Arduino and MATLAB communication
Transferring bits from MATLAB to Arduino
Fabricate R2R DAC circuit and establish communication with Arduino
Read the output of R2R DAC circuit and tabulate the results
Setup WSN and establish communication with NI - 3226 and R2R output voltage signal
Monitor and log the readings observed on LabVIEW environment
Stop

Figure 15 Flowchart of Phase II

Figure 16, is the proposed WSN architecture for WSN system. In the figure, the pre-processing circuit is described with ADXL335 accelerometer sensor and the Arduino Uno hardware. ADXL335 is a tri-axial accelerometer and is connected to the Arduino Uno board, and the sensor readings (raw data) are obtained and stored on a .csv file. R-2R is the Digital-to-Analog-converter circuit built to convert the binary bits to analog values. The WSN is formed by the NI - 3226 (sensor node), NI – 9791 (Real-time gateway) and a base station to monitor the status of the signal. The IEEE 802.15.4 protocol connects the sensor node and the gateway wirelessly. The raw sensor data has three axial readings namely, x, y, and z respectively and the sensitivity (S) of the sensor is noted as 390mg/step. The linear motions along the axes are named as longitudinal, lateral and vertical translations respectively. Also, the rotatory motions around these axes are termed as roll, pitch, and yaw, respectively.

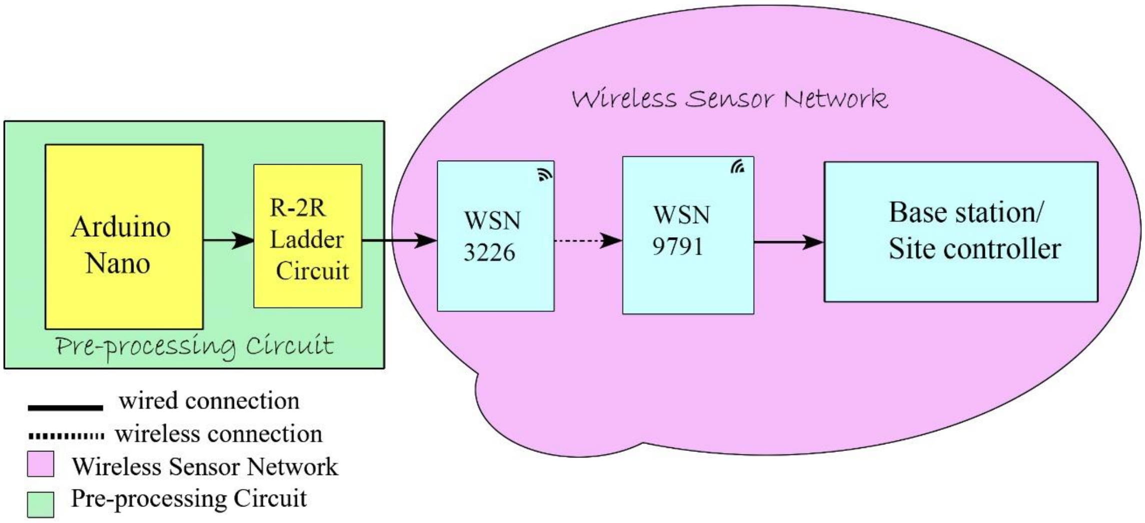


Figure 16 Proposed WSN architecture

Figure 17, illustrates the axes naming convention of a typical accelerometer on the train. The sensor data obtained from the sensor is converted into the units of acceleration (m/s$^{2)}$ as follows:

$\text{Sensor Resolution } (S) = 0.0039\text{g/step}$

$\text{Acceleration Gravity } (g) = 9.807 \text{ m/s}^2$

$\text{Lateral Acceleration } (Y) = l * S * g$

$\text{Vertical Acceleration } (Z) = v * S * g$

$\text{Longitudinal Acceleration } (X) = x * S * g$

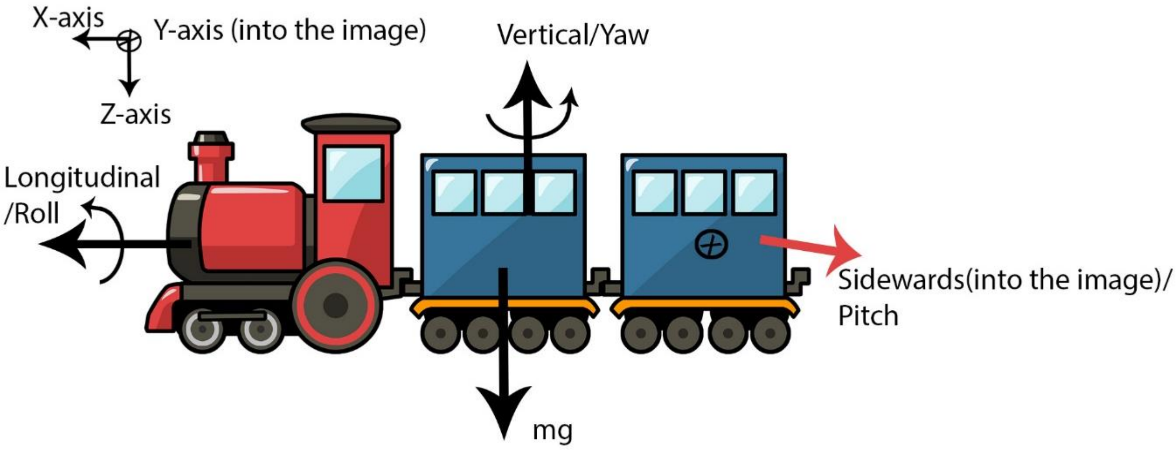


Figure 17 Naming convention of accelerometer axes

The sensor data is stored on the .csv file, and the readings are imported onto MATLAB. The imported readings are converted to the units of acceleration (m/s^2) and plotted against each other.

4.2 Tri-axial Acceleration Magnitude signal

A script is developed on MATLAB as shown in Appendix 1.1: Reading the .csv file. The readings are imported from a .csv file, and the readings are plotted as shown in Figure 18. In Figure 18, it is observed that the vertical acceleration (red colour) magnitude of the acceleration signal shown higher deflections when compared with the other axes namely, longitudinal (black colour) and lateral (blue colour). The deviations in the vertical acceleration signal indicate that the displacement of the train is upwards and downwards. Signal analysis is performed on the vertical acceleration signal to identify the characteristics like,

1. Maximum acceleration (on both positive and negative axes)
2. Maximum frequency
3. Magnitude values above and below of threshold levels
4. Time at which jerks observed

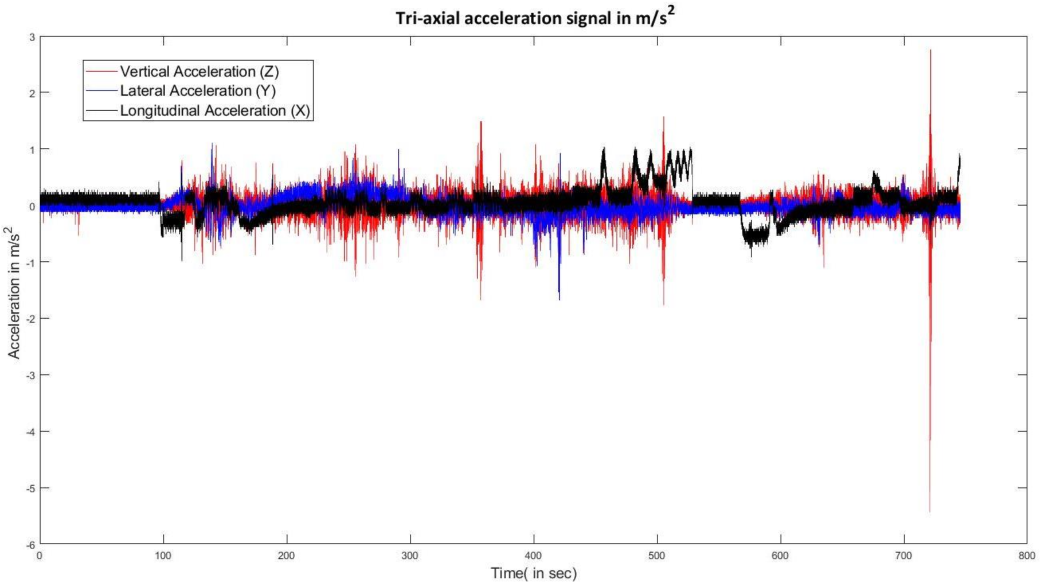


Figure 18 Tri-axial Acceleration signal

4.3 Vertical Acceleration signal and Fast-Fourier Transform

As seen in Figure 18, high deflections are observed on the vertical acceleration signal and are observed along the Z-axis. Besides, the deviations in their magnitude on +Z and –Z axes are marked on the vertical acceleration signal using MATLAB script, and a plot can be visualized in Figure 19(a). The peak acceleration along the +Z -axis is termed as maximum acceleration and the peak acceleration along the –Z-axis is termed as the minimum acceleration. The MATLAB script for plotting the vertical acceleration signal is shown in Appendix 1.2.a: Plot the vertical acceleration signal.

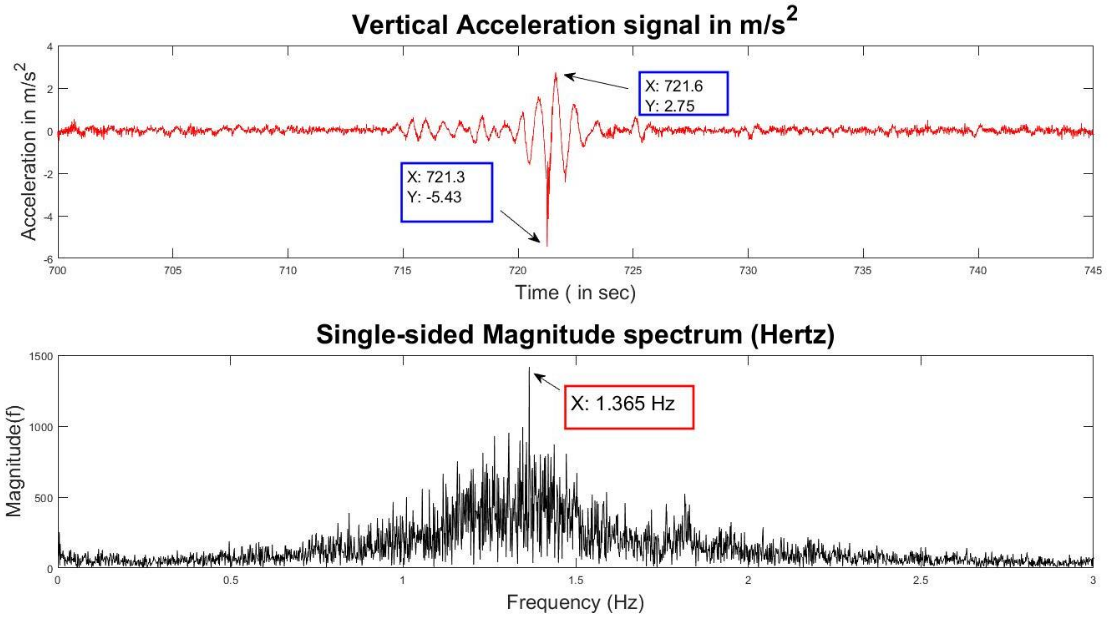


Figure 19 (a) Vertical Acceleration signal in m/s^2 (b) Single-sided Magnitude spectrum

The Fast-Fourier Transform (FFT) is performed to identify the frequency components in the signal. FFT is a powerful mathematical tool that converts the signal in time-domain to frequency- domain, and hence it becomes easy to analyze the signal. Besides, FFTs produce the average frequency content of the signal over the entire time. The initial calculations for analyzing the signal include converting from a two-sided power spectrum to the single-sided power spectrum. The power spectrum produces an array that contains the two-sided power spectrum of a time-domain signal. Two-sided power spectrum shows negative and positive frequency components at an amplitude height of $\frac{A_k^2}{4}$, where A_k is the peak amplitude of the sinusoidal component at frequency k. However, the negative frequency components are redundant since the spectrum of real word signal is symmetrical around DC. Therefore, a plot of single-sided power spectrum produces only positive frequencies. This is achieved by disregarding the second half of array and multiplying every point except for DC by factor two.

$$G_{AA}(i) = S_{AA}(i), i = 0 \text{ (DC)}$$

$$G_{AA}(i) = 2 * S_{AA}(i), i = 1 \text{ to} \frac{N}{2} - 1$$

Where, $S_{AA}(i)$ is the double-sided spectrum, $G_{AA}(i)$ is the single-sided spectrum, N is length othe f double-sided spectrum. A plot of the single-sided spectrum is shown in Figure 19(b) and the, corresponding MATLAB script is shown in Appendix 1.2.b: Plot the frequency spectrum of the vertical acceleration signal. The highest frequency component of the FFT signal is termed as the maximum frequency of the vertical acceleration signal.

4.4 Magnitudes above and below Threshold levels

Sometimes, it may be required to identify the acceleration values above and below the threshold levels. Not all the magnitudes below the threshold levels may be of use since, their magnitudes are very small, and they do not give us any information about the signal. In this case, the threshold on the positive Z-axis is chosen as 1.5 m/s^2 and the negative Z-axis as -1.5 m/s^2 respectively. The magnitudes above 1.5 m/s^2 and below 1.5 m/s^2 are marked on a plot. Figure 20, is the plot of peaks or magnitudes above and below the threshold levels. '*' are the threshold values above 1.5 m/s^2 on the positive Z-axis and '◦' are the threshold values that are below 1.5 m/s^2 on the negative Z-axis. The MATLAB script to find the threshold values is shown in Appendix 1.3 a: Finding the threshold values. Of all the values observed in Figure 20, the greatest values on both the +Z and -Z-axes are marked as maximum acceleration and minimum acceleration magnitudes with the units of m/s^2.

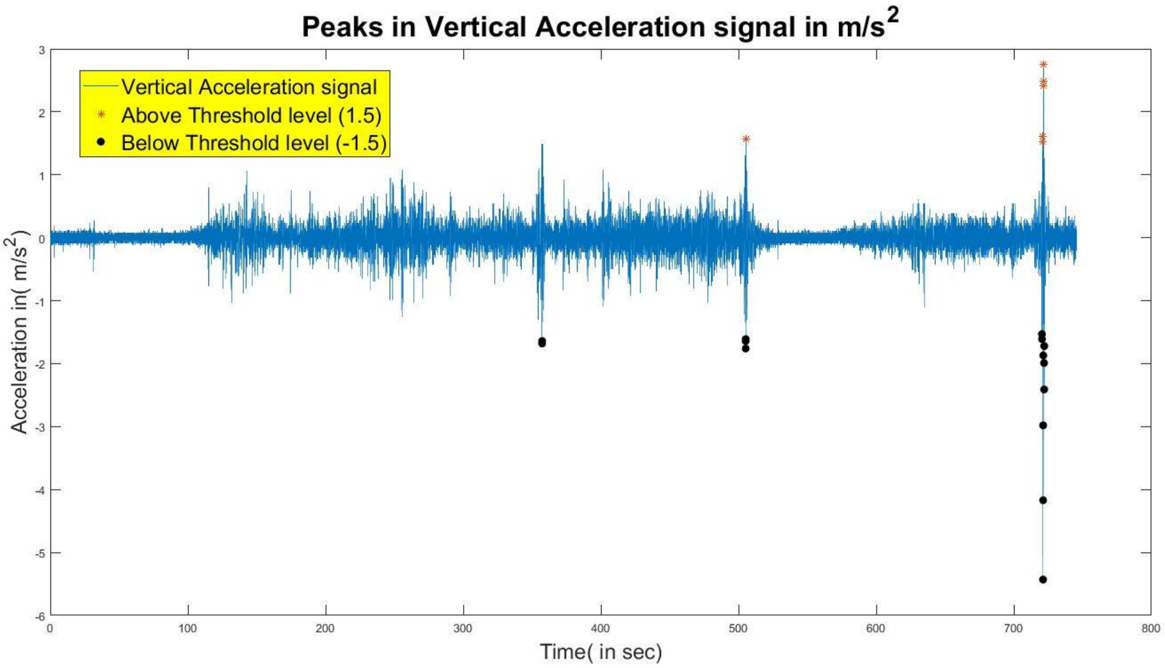


Figure 20 Peaks observed above and below the threshold values

A jerk can be defined as the rate of change of acceleration of the train. The jerk is observed at the maximum and minimum acceleration magnitudes namely, 2.75 m/s^2 and -5.43 m/s^2. Also, it is also possible to identify the time instant at which the jerks are noticed. By evaluating MATLAB script of Appendix 1.3 b: Finding the jerks in the acceleration signal, the jerks are identified and plotted in Figure 21. In addition, 721.3 and 721.6 seconds are the respective time instants at which jerks are observed in the acceleration signal.

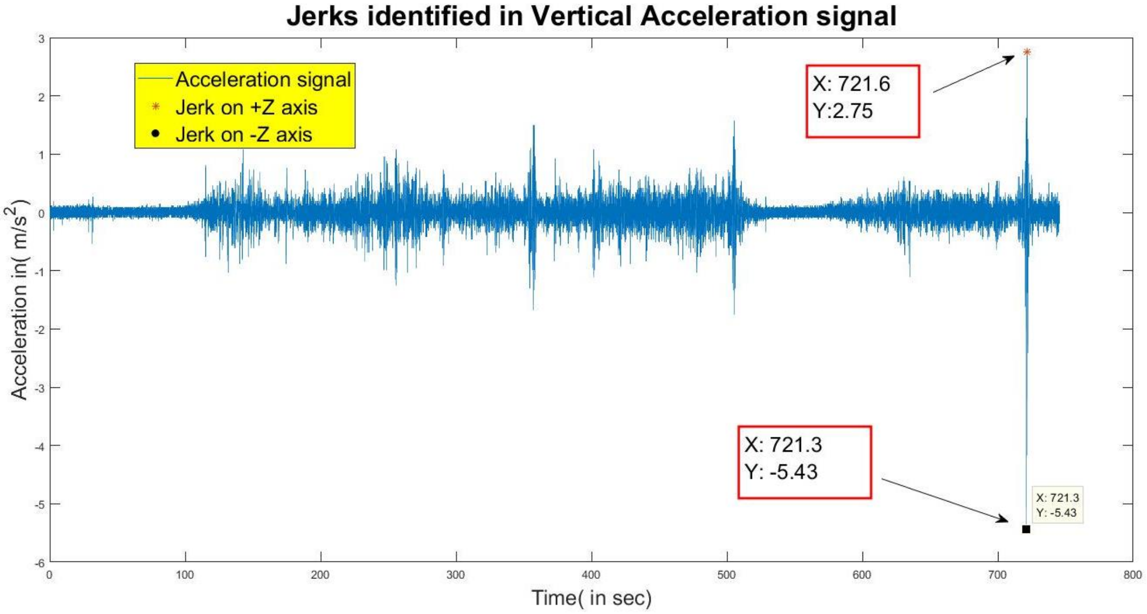


Figure 21 Jerks identified in the vertical acceleration spectrum

4.4.1 Results & Discussion

After the script is evaluated on MATLAB, the characteristics namely maximum and minimum acceleration, above and below threshold levels and the jerks are marked as shown in Figure 19, Figure 20 and Figure 21 respectively. The marked readings on the plots are shown in Table 11 and Table 12 respectively. As seen in Table 11, the threshold values are greater than 1.5 m/s^2 and less than -1.5 m/s^2. The values that are below the threshold levels can be ignored, as they do not provide any significant information. Also, the essential characteristics of the signal are maximum acceleration, minimum acceleration, and maximum frequency and are shown in Table 12.

Table 11 Magnitudes above and below threshold levels

N^0	Threshold values above 1.5 m/s^2		Threshold values below -1.5 m/s^2	
	Acceleration (m/s^2)	Time (in secs)	Acceleration (m/s^2)	Time (in secs)
1	1.570	505.5900	-1.680	357.3100
2	1.610	720.8700	-1.640	357.3700
3	1.530	720.9500	-1.640	505.1900
4	2.410	721.5800	-1.760	505.2200
5	2.750	721.6300	-1.610	505.2600
6	2.490	721.6700	-1.530	720.4600
7	-	-	-1.610	720.5000
8	-	-	-5.430	721.2600
9	-	-	-4.170	721.3000
10	-	-	-2.980	721.3500
11	-	-	-1.870	721.4000
12	-	-	-1.990	722.0100
13	-	-	-2.410	722.0500
14	-	-	-1.720	722.0900
15	-	-	-1.720	722.1100

Table 12 Characteristics of the Vertical acceleration

N^0	Characteristics	Readings
1	Maximum Acceleration	2.75 $\mathrm{m/s^2}$
2	Minimum Acceleration	-5.43 $\mathrm{m/s^2}$
3	Maximum Frequency	1.365 Hz

4.5 Analog to Analog reference Scaling

As seen in Table 12, the magnitude of minimum vertical acceleration is in negative m/s^2. Since it is not possible to convert negative magnitude values to digital values directly, the voltages are scaled to lie within the range of Arduino Uno input voltage (0-5V). An equation is established using the point-slope form, to scale the negative and positive voltage values to fall within 0-5 V as shown in Figure 22.

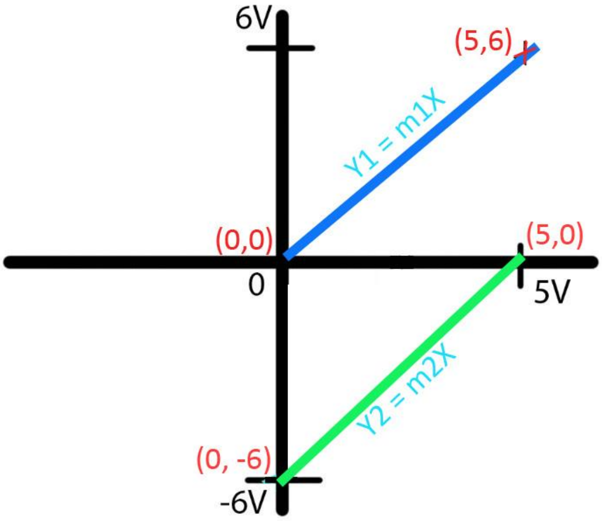


Figure 22 Point-slope form to calculate analog to analog reference values

Let (x1, y1) = (0, -6), (x2, y2) = (5,0), (x3, y3) = (0,0) and (x4, y4) = (5,6)

Using, the point-slope form, $(\mathrm{y} - \mathrm{y1}) = \mathrm{m}(\mathrm{x} - \mathrm{x1})$

$$\text{Where, } \mathrm{slope, m} = \frac{\mathrm{y2-y1}}{\mathrm{x2-x1}}$$

$$\mathrm{m_1} = \frac{0-(-6)}{5-0} = \frac{6}{5} \text{ And } \mathrm{m_2} = \frac{0-(6)}{0-5} = \frac{6}{5}$$

Hence, $(\mathrm{y} + 6) = \frac{6}{5}(\mathrm{x} - 0)$

$$\text{Line } \mathrm{L_1} \text{ is } \mathrm{x} = \mathrm{round}\left(\frac{5}{6}(\mathrm{y} + 6)\right)$$

Therefore, $\mathrm{Analog\ reference} = \mathrm{round}(\frac{5}{12}(\mathrm{analog} + 6))$ --------------- 1

Also, Line L_2 is given by, $\mathrm{x} = \mathrm{round}\left(\frac{5}{6}(\mathrm{y})\right)$ ------------------------------ 2

Using, the equation 1 and 2, the MATLAB script is developed on MATLAB and is shown in Appendix 1.4: Scaling the values to 0 – 5 V.

4.5.1 Results & Discussion

Two different equations of the line are derived using the point-slope form and are namely,

1. L1, X = m1Y where m1 is 5/6
2. L2, X = m2Y where m2 is 5/6

Using the line equations, L1 and L2 all the analog characteristic magnitudes are scaled to Analog_reference magnitudes of range (0 – 5) volts.

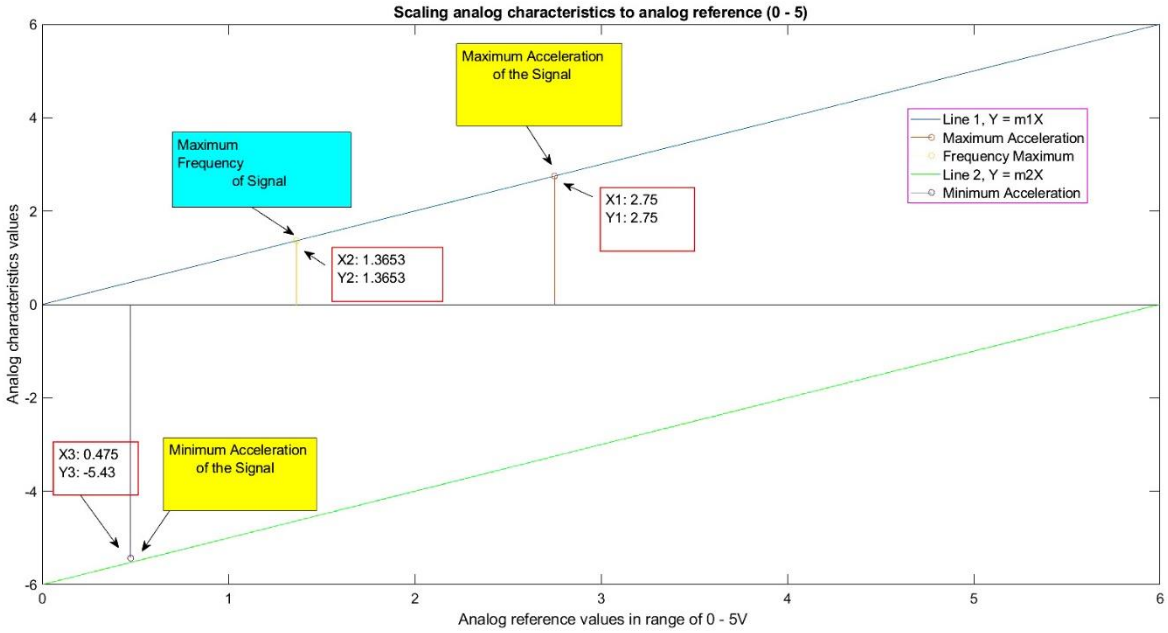


Figure 23 Analog characteristics to Analog reference (0-5) scaling

Figure 23, displays the scaling of analog characteristics to fall in the range of (0-5) volts and Table 13, shows the magnitudes of scaled readings. Also, the magnitudes are marked and indicated with the arrows as shown in Figure 23.

Table 13 Scaling Analog values to Analog reference

N^0	Characteristics	Analog Reference Readings
1	Maximum Acceleration	2.75 $\mathrm{m/s^2}$
2	Minimum Acceleration	0.475 $\mathrm{m/s^2}$
3	Maximum Frequency	1.3653 Hz

4.6 Analog reference to Digital reference scaling

The analog reference voltage values have a range from 0 – 5V. However, the Arduino Uno micro-controller does not support to produce the outputs in analog format. The micro-controller has digital output pins ranging from D0 – D13. Therefore, the analog values are converted into digital format initially and later to binary digits. The command writeDigitalPin (a, pin, value), writes each binary bit to the Arduino digital pins D0-D13. In the digital conversion, the quality of the analog signal is defined by its resolution value. The resolution is the number of bits that are used to represent the entire signal. For instance, if the sampler is n –bit resolution, there are total 2^{n} possible digital values that each sample to be converted. In general, the higher the bits of resolution, the greater the quality of the signal. When chosen a 10-bit resolution, the minimum digital value is given by,

$$\mathrm{min} = \frac{\mathrm{max\_analogvoltage}}{2^{\mathrm{n}}} = \frac{5}{2^{10}} = 0.0048\mathrm{volts}$$

The point-slope form line scales the analog values to the digital values, and the generalized equation of the line is calculated as shown in Figure 24.

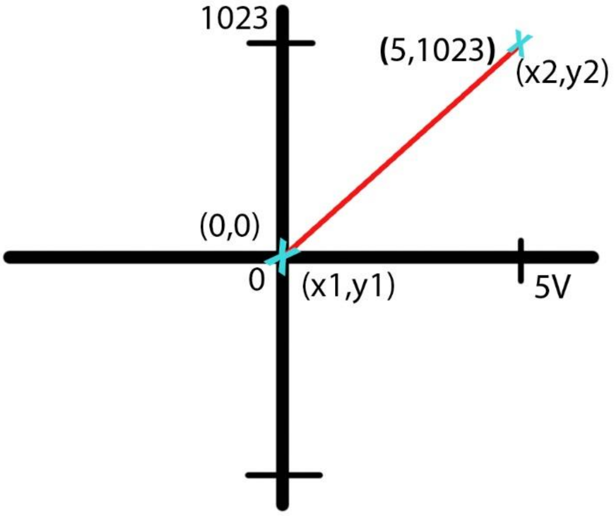


Figure 24 Point-slope form to calculate digital reference from analog reference values

Let (x1, y1) = (0, 0) and (x2, y2) = (5, 1023)

$$\mathrm{m2} = \frac{1023 - 0}{5 - 0} = \frac{1023}{5}$$

Hence, $(\mathrm{y} - 0) = \frac{1023}{5}(\mathrm{x} - 0)$

$$\mathrm{y} = \mathrm{round}\left(\frac{\left(\frac{1023}{5}(\mathrm{x})\right) * 100}{100}\right)$$

Therefore, $\mathrm{Digital} = \mathrm{round}\left(\frac{\frac{1023}{5}(\mathrm{analog\ reference}) * 100}{100}\right)$ -------------------- 3

Similarly, using equation 3, the MATLAB script is developed as shown in Appendix 1.5: Scaling the values to 0 – 1023, to calculate the digital values of corresponding analog values and stored in the variable Digital_reference. Also, the analog reference values are plotted against their corresponding digital reference values as shown in Figure 25.

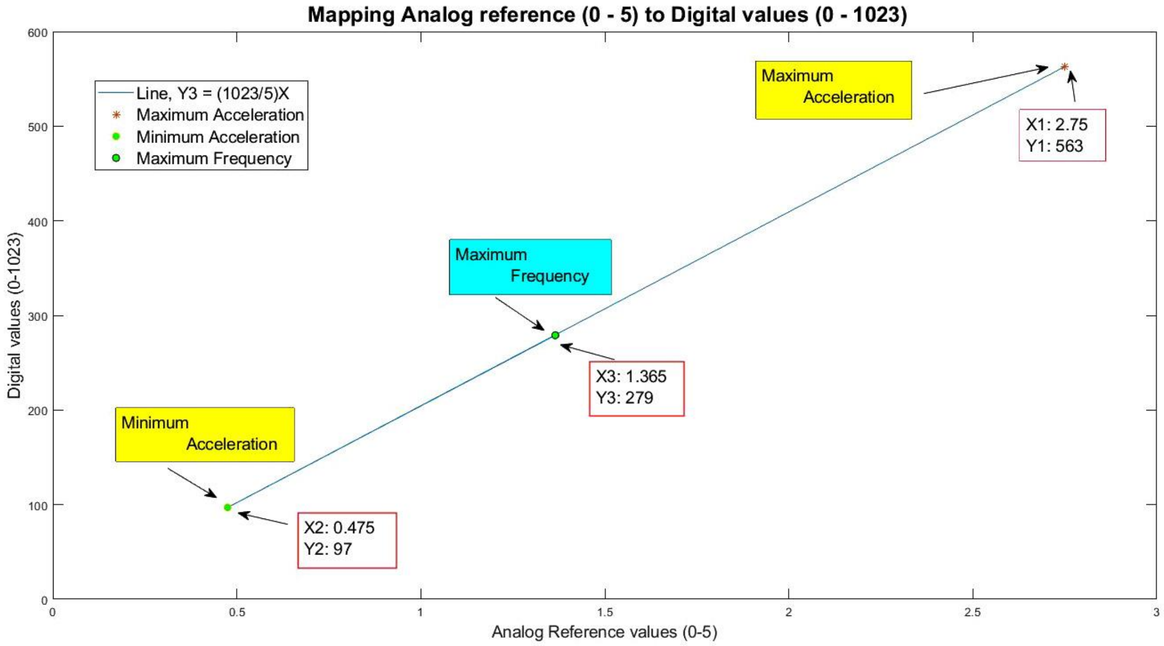


Figure 25 Plot of analog reference against a digital reference

4.6.1 Results & Discussion

The Analog_reference readings of range (0-5) volts are mapped to the decimal values ranging from 0 – 1023 with a step division of 0.00488 volts. The decimal values are non-negative and therefore it is possible to convert them into binary vectors. The scaled values are displayed in Table 14.

Table 14 Analog Reference to Digital Reference scaling

N^0	Characteristics	Digital Reference Readings (Decimal format)
1	Maximum Acceleration	563
2	Minimum Acceleration	97
3	Maximum Frequency	279

4.7 Digital (decimal) values to binary conversion

The command, writeDigitalPin (a, pin, value), writes the data the digital pins. However, the value in the command must be logical, i.e., 0 and 1. MATLAB Script is developed on the MATLAB to convert the digital voltage values to binary format, and it is shown in Appendix 1.6: Binary conversion of decimal values.

4.7.1 Results & Discussion

The decimal values of range (0 – 1023) is converted to 10-bit binary number using the function decimalToBinaryVector on MATLAB. Each binary number has ten digits, and each binary bit is later transferred to ten digital pins of Arduino. The binary vectors of ten digits length are displayed in Table 15.

Table 15 Analog Characteristics in the form of binary vectors

N^0	Characteristics	Digital Readings (Binary form)
1	Maximum Acceleration	10 0011 0011
2	Minimum Acceleration	00 0110 0001
3	Maximum Frequency	01 0001 0111

Chapter 5

5.1 Introduction

Chapter 5 is the continuation of Chapter 4. In Chapter 4, the retrieved important information from the tri-axial acceleration signal is converted to the binary format of 10 bits each. These binary numbers are stored in different variables on MATLAB workspace. It is observed that that the WSN – 3226 node has 4 analog channels (AI0 – AI3) and 2 digital channels (DIO0 and DIO1). The digital channels can accept either digital 1 or 0 and hence a total of only 2 bits can be given to those two channels. All the bits of the 10-bit binary numbers cannot be sent to the WSN node at one instant. Therefore, a pre-processing circuit is the solution, obtained to transmit the information stored in a 10-bit binary number. The circuit converts the 10-bit binary information to a single analog value and this value can be easily given to any of the analog channels (AI0 – AI3) of NI – 3226.

5.1.1 Aims and Objectives:

The primary focus of this chapter is to transmit the 10-bit binary information stored in the variables on MATLAB workspace to base station on the LabVIEW environment. The objectives of Chapter 5 include:

- Building a pre-processing circuit
- Establishing communication between pre-processing circuit and WSN system
- Receiving and logging the information at the base station on LabVIEW environment

5.2 Building a pre-processing circuit

The pre-processing circuit is built with the combination of Arduino Uno R3 and R2R DAC (Digital-to-Analog converter) hardware. Arduino Uno R3 is a microcontroller board based on the Microchip ATmega328P AVR microcontroller and the board has 14 digital I/O pins of which 6 of them provide PWM output. The digital pins D2 - D11(10 terminals) are chosen to fulfil the purpose of receiving 10 bits binary number as the input. R2R DAC circuit is a simple Digital-to-Analog converter and is built by the parallel combination of resistances R and 2R. Firstly, a communication is established between the MATLAB workspace and the Arduino Uno. Secondly, the output received at the digital output pins is sent to R2R DAC circuit, which produces a single analog voltage value at its output terminal.

5.2.1 Communication between MATLAB and Arduino

The MATLAB support package for Arduino Hardware helps to communicate with the Arduino board. The command writeDigitalPin (a, pin, value) writes the binary digits obtained in section 4.7 Digital (decimal) values to binary conversion, to digital pins (D11-D2) of the Arduino board. The flow of bits from the MATLAB to the Arduino digital pins is shown in Figure 26. Also, the 10- bits obtained from the MATLAB script is displayed on the left-hand side with Most Significant Bit on the top and Least Significant Bit at the bottom. Also, the PORTD (D11-D2) are displayed on the right-hand side. The different colours in the Port_D and PORT D indicate that all the bits are different and have a relationship concerning the other bits. The MATLAB script to describe the flow of bits is displayed in Appendix 2.1: Mapping bits to Digital pins of Arduino board.

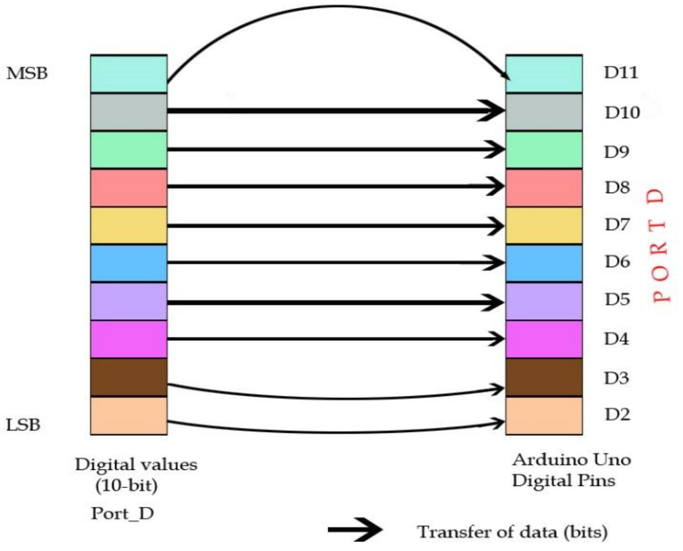


Figure 26 Flow of bits from MATLAB to Arduino Digital pins

The bits received on the digital pins D11-D2 are either 0 or 1's. These bits contribute together to form the required analog voltage. However, this process requires a special Digital-Analog-

Circuit (DAC). Of all the DACs, R-2R DAC circuit is easy to fabricate with just two resistor values namely, nK and 2nK, where n is the natural number.

5.2.2 R-2R ladder circuit

R-2R ladder circuit directly converts a parallel symbol into an analog voltage. The weighted contribution of each digital input (b1, b2, etc.) is summed to get the analog voltage. The R-2R ladder network has some important properties when compared to the other DAC circuits, and some of them are provided below:

1. The network can be easily extended by adding any number of bits to increase to the resolution
2. Requires only two values of resistors and hence easy for fabrication and integration
3. The output impedance is always R, despite the number of bits

The generalized equation to calculate the analog value is given by,

$$\mathrm{Vout} = \frac{\mathrm{Vb0}}{2^{\mathrm{n}}} + \frac{\mathrm{Vb1}}{2^{\mathrm{n}-1}} \ldots\ldots\ldots\ldots + \frac{\mathrm{Vb(n-1)}}{2^{1}} + \frac{\mathrm{Vbn}}{2^{0}} \qquad \text{---------------------- } 4$$

Where n = number of bits

Vb = voltage contributed by the corresponding bit

Vout = Analog value

From Figure 27, the resistors R and 2R are 10K and 20K respectively. Hence, equation 4, can be written as,

$$\mathrm{Vout} = \frac{\mathrm{Vb0}}{2^{10}} + \frac{\mathrm{Vb1}}{2^{9}} + \frac{\mathrm{Vb2}}{2^{8}} + \frac{\mathrm{Vb3}}{2^{7}} + \frac{\mathrm{Vb4}}{2^{6}} + \frac{\mathrm{Vb5}}{2^{5}} + \frac{\mathrm{Vb6}}{2^{4}} + \frac{\mathrm{Vb7}}{2^{3}} + \frac{\mathrm{Vb8}}{2^{2}} + \frac{\mathrm{Vb9}}{2^{1}} + \frac{\mathrm{Vb10}}{2^{0}}$$

Besides, Figure 28 is the R-2R ladder network developed on the stripboard. The terminals (black colour) are provided on the top of the circuit to connect the ground, bits (1-10) and Output voltage (Vout) respectively.

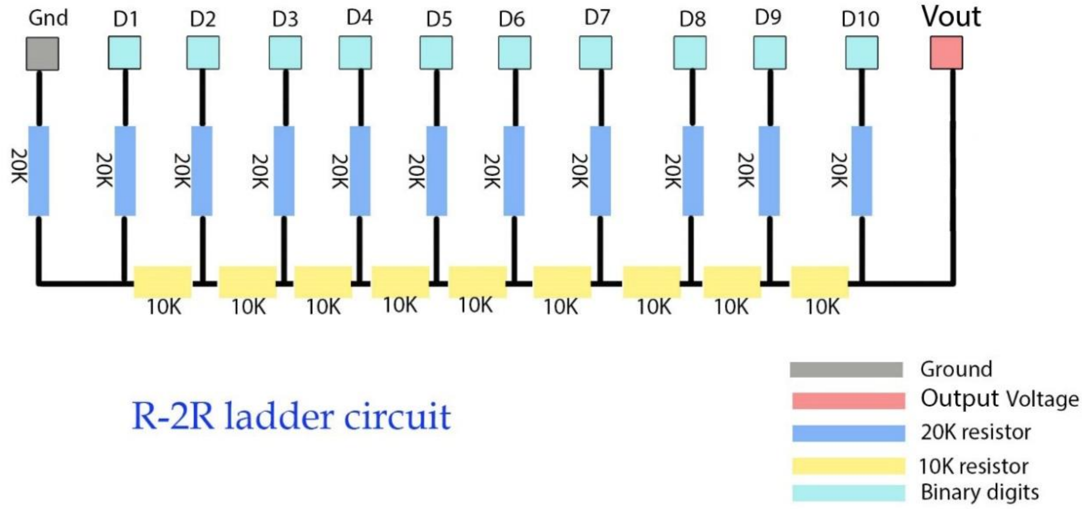


Figure 27 R-2R Ladder network circuit

The circuit is fabricated using the 10K and 20K resistors with a terminal on the top for secure connectivity. The DAC circuit is built as shown in Figure 28 on the stripboard.

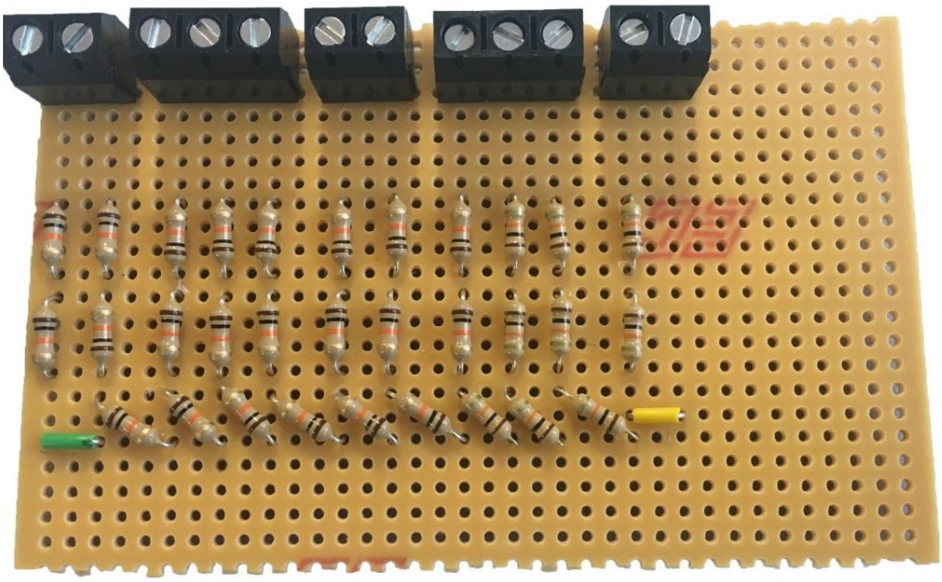


Figure 28 R-2R ladder network circuit on stripboard

Now that the pre-processing circuit is ready, a connection is made between the Arduino Uno R3 and R–2R DAC circuit. The 10 digital output pins of Arduino namely, D2 – D11 are connected to digital input terminals (D21 – D10) of R-2R DAC circuit.

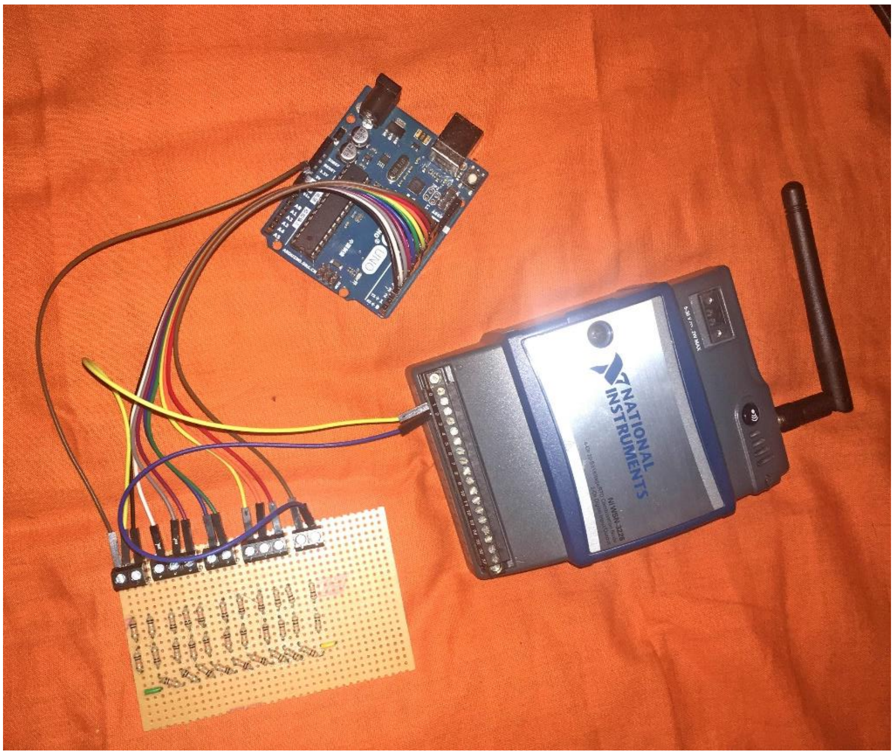


Figure 29 Connection of Arduino board and R2R DAC circuit

Figure 29 illustrates the connection between Arduino digital pins (D2 – D11) and DAC circuit terminals. The output (analog value) is received at the pin Vout as shown in Figure 27. To check the efficiency of the pre-processing circuit, the characteristics of acceleration magnitude signal received at the output terminal (Vout) are compared with that of the magnitudes that were saved on MATLAB workspace. Therefore, the corresponding output of all the characteristics of vertical acceleration signal are received on the LabVIEW environment by via USB-6008 Data Acquisition (DAC) device before feeding into the WSN system.

5.2.3 Results and Discussion

On the LabVIEW environment, all the readings received are logged in a text file with the help of Data Acquisition (DAQ) assistant using USB - 6008 and the readings are displayed in Table 16. From Table 16, it is observed that the DAC circuit produced the characteristic values at the output terminal with an error of almost 1.5%. The error can be minimised by allowing the output to pass through the voltage buffer. Also, thanks to the NI – 3226 as it has inbuilt voltage buffer embedded onto the circuit board. Therefore, from Table 16, it can be said that the pre-processing circuit is efficient with minimal error.

Table 16 Output readings of R2R DAC circuit

N^0	Case	Characteristics	Analog Output Readings	% of Error
1	Case A	Maximum Acceleration	2.7149 $\mathrm{m/s^2}$	1.2763%
2	Case B	Minimum Acceleration	-5.34 $\mathrm{m/s^2}$	1.6474%
3	Case C	Maximum Frequency	1.3858 Hz	1.5015%

5.3 Measuring the voltage signal using WSN system

In WSN condition monitoring, the voltage signal obtained at the output terminal (Vout) of pre-processing circuit is received at the base station via WSN system. Hence, a WSN system is built using the hardware, NI – 3226 (sensor node) and NI – 9791 (Ethernet gateway) respectively. The hardware is programmed on LabVIEW, a development platform designed by the company itself for a visual programming language.

5.3.1 Communication between pre-processing circuit and WSN system

As mentioned, the architecture of WSN system is built using National Instruments devices and is already shown in Figure 16. In this figure, the output signal of pre-processing circuit is a voltage signal and it should be given to the input terminal of NI – 3226. In voltage mode, NI – 3226 is operated only with single ended DC measurements. Each channel has one input terminal and one ground terminal. The connection between pre-processing circuit and the WSN node is shown in Figure 29 . From Figure 27, the output pin, Vout and the Ground, Gnd are considered as the positive and negative terminals respectively. The positive terminal (Vout) of R2R DAC circuit is connected to AI0, an analog input terminal and the negative terminal (Gnd) to COM0, the ground terminal of WSN – 3226 to measure the voltage of the signal. The connections of the output voltage signal to the NI– 3226 is shown in Figure 30.

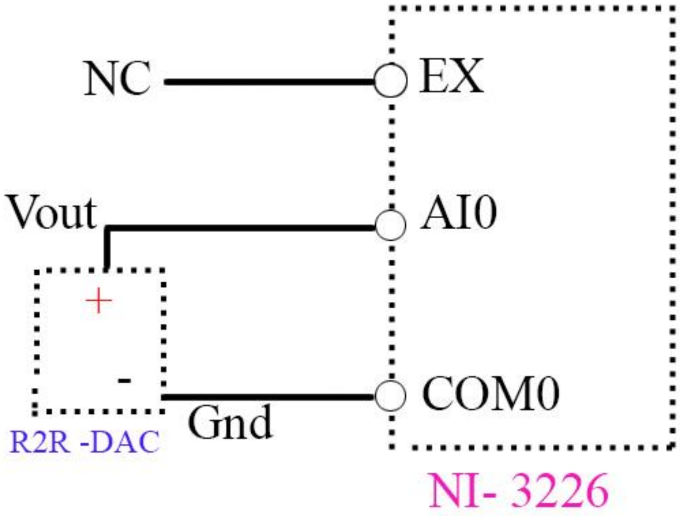


Figure 30 Voltage source connected to Analog input terminal

Each WSN node has a unique serial number and it is used to connect with the WSN gateway wirelessly using the graphical user interface NI – MAX (Measurement and Automation Explorer). The gateway is connected to the base station via ethernet cable. In this project, a single node is connected to the gateway and a maximum of eight nodes can be connected to the gateway at any given time. The flow of information between pre-processing circuit and the base station is as follows:

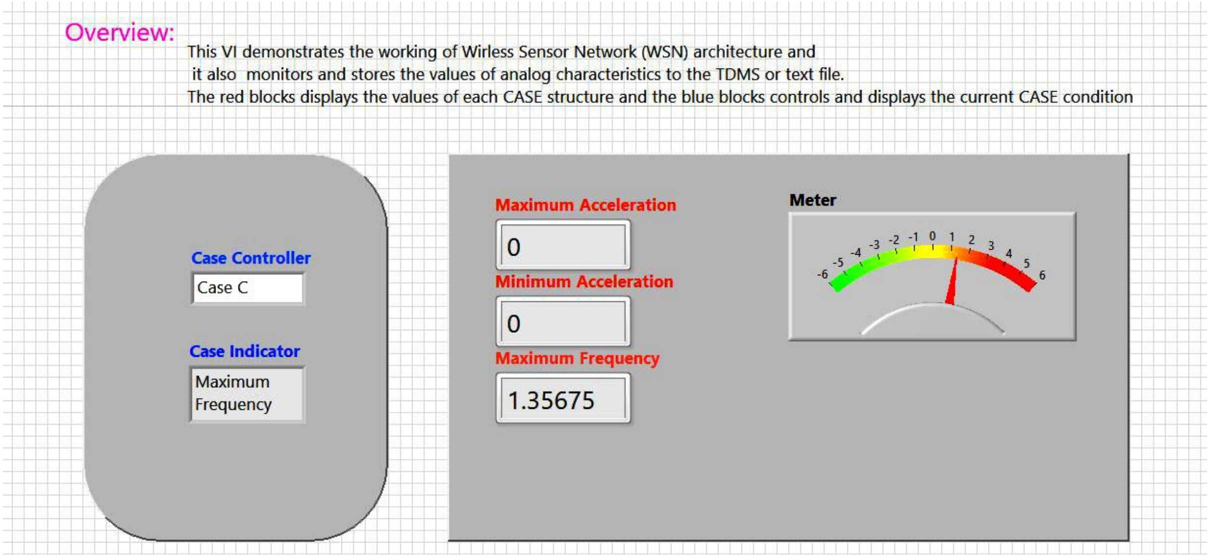
PRE-PROCESSING CIRCUIT → NI – 3226 → NI – 9791 → BASE STATION

5.3.2 Receiving and logging information at the base station

On the base station (local computer), a Virtual Instrumentation (VI) is created on the LabVIEW environment system using NI - 3226 node properties. A WSN project is created using the WSN node, NI – 3226 that is connected to the gateway, NI – 9791. The project holds all variables namely, input and output channels, link quality, battery voltage, external power etc. The VI demonstrates the working of WSN. The VI has two windows namely, front panel and the block diagram. Initially, the front panel is created with specific indicators and controllers. Figure 31, is the front panel formed in the project.

The operations of the front panel are as follows:

1. Controls the case conditions
2. Displays the current case condition
3. Display the analog characteristic magnitude for the current CASE condition
4. Graphical meter reading to show the magnitude



Overview: This VI demonstrates the working of Wirless Sensor Network (WSN) architecture and it also monitors and stores the values of analog characteristics to the TDMS or text file.
The red blocks displays the values of each CASE structure and the blue blocks controls and displays the current CASE condition

Case Controller: Case C

Case Indicator: Maximum Frequency

Maximum Acceleration: 0

Minimum Acceleration: 0

Maximum Frequency: 1.35675

Figure 31 Front panel of the WSN project

The functioning of the front panel is solely dependent on the graphical programming of a block diagram. Figure 32, is the block diagram of the WSN system. In Figure 32, AI0, an analog input channel variable of NI – 3226 is dropped from the list of NI – 3226 nodal variables from the WSN project. A Case structure is used in the graphical programming of block diagram. The Case structure has three cases namely, Case A, Case B and Case C for maximum acceleration, minimum acceleration and maximum frequency of the signal respectively.

The case controller selects the CASE among three different cases namely,

- CASE A for maximum acceleration of the signal
- CASE B for minimum acceleration of the signal
- CASE C for a maximum frequency of the signal

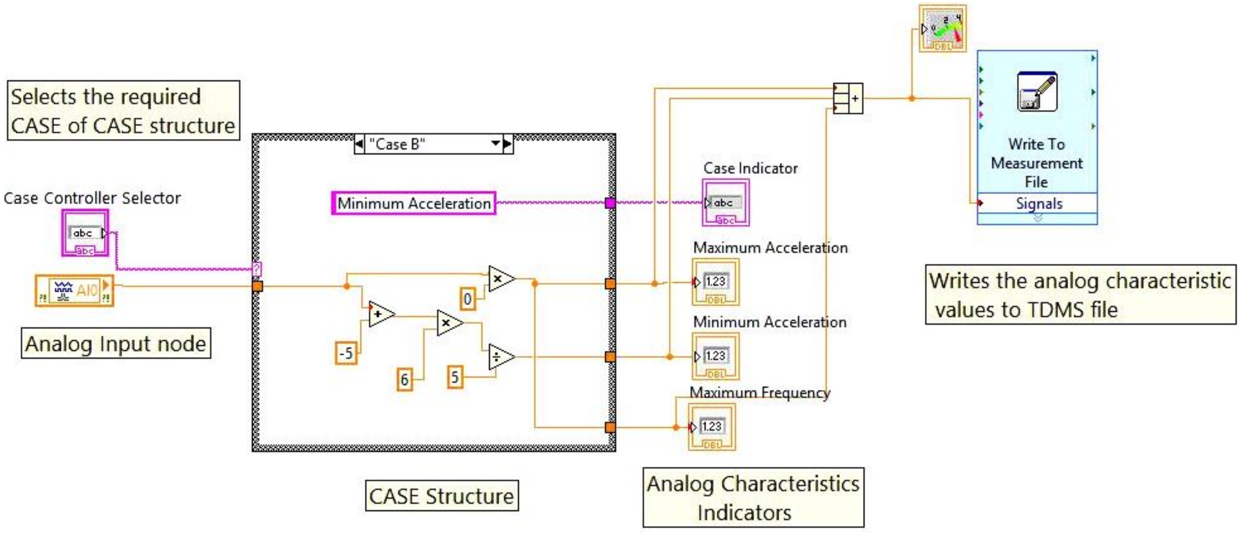


Figure 32 Block Diagram of the WSN project

Based on the case selection, the voltage signal from the AI0 terminal of NI - 3226 is sent to the corresponding case structure and some arithmetical operations are performed inside the structure. For instance, if Case A, is selected, then case structure Case A will be opened and after performing calculations, the case indicator and its value will be displayed on Case Indicator and Maximum Acceleration indicators respectively. Also, the other indicators display a value of ‘0’. All the displayed values on indicators are stored in a TDMS file in their respective columns. The magnitude is fed to the indicators and to the Measurement file for storing the data. Figure 32, displays the CASE condition (CASE ‘B’), which monitors and logs the minimum acceleration of the signal.

5.3.3 Results & Discussion

In section 5.3.2 Receiving and logging information at the base station, Case C is selected on the case controller of the front panel. The corresponding case structure, Case C is executed on the block diagram and the output produced is displayed on the indicators. Since Case C is selected, maximum frequency indicator displays the measured value and rest of the indicators displaying ‘0’. A random case structure, Case B is chosen to explain the operations inside the case structure. As shown in Figure 32, for Case B condition, the voltage signal received on the analog input terminal AI0 is rescaled to its original value to fall with the range of 0 m/s^2 to – 6 m/s^2. Also, the magnitude of the voltage signal is multiplied with value zero to display other quantities as zero and the quantities are stored on the TDMS file. The logged data on the Measurement file of the block diagram is presented in Table 17

Table 17 Analog characteristics recorded via WSN architecture

N^0	Case	Characteristics	Analog Output Readings	% of Error
1	Case A	Maximum Acceleration	2.73194 $\mathrm{m/s^2}$	0.6567%
2	Case B	Minimum Acceleration	-5.4293 $\mathrm{m/s^2}$	0.3798%
3	Case C	Maximum Frequency	1.35637 Hz	0.6540%

Conclusion

This research work has researched a variety of WSNs used to perform condition monitoring in the railway industry. The significant importance is on the intelligent WSN architecture to receive the data securely at a remote place.

In the recent past, the condition monitoring of railways is performed by human inspection and these methods are applied sporadically and especially on the infrastructures [6], [7]. Despite the hard work of humans, detailed information on the complete infrastructure is not obtained. Sensors can provide the complete information of the infrastructure both internally and externally. Thus, with the overall health condition of the infrastructure, it is easy to evaluate its endurance and lifetime. Since there are wide range of sensors and WSN technologies, it is a challenge to choose the best WSN architecture that can provide effective, secure and stable connection. From [11] and [20], it is evident that the architecture in Figure 4 provides secure and effective communication.

An application of MEMS sensors in WSN for remote condition monitoring is also introduced in the research work. In this research work, an ADXL 335 MEMS accelerometer is used to take advantage of its low price, compact size, and less energy consumption. The WSN smart architecture is built as shown in Figure 16. The figure illustrates the fabrication of the pre-processing circuit and its connection with the WSN. NI – 3226 sensor node, NI – 9791 real-time gateway and a base station to monitor and store the received data, forms the WSN. ATmega328P microcontroller of pre-processing circuit extracted extensive information from the vertical acceleration signal. Initially, the architecture is tested by allowing a sample sinusoidal signal to verify its robustness. The data is delivered at the base station via WSN architecture with a zero percent error and thus proved its robustness.

The effectiveness of WSN architecture is observed acutely in Chapter 3 of this paper. In Table 3, displays the output signal attributes received at the base station via WSN architecture. By comparison, of the attributes in Table 2 and Table 3, it is proved that the chosen WSN architecture is stable and effective with zero error.

In Chapter 4, the raw acceleration signal is imported on MATLAB, and signal analysis is performed to identify the characteristics like maximum and minimum magnitudes, peaks above and below threshold levels. Thus, a continuous signal analysis provides significant information of the infrastructure can be known, and hence it becomes easy to identify the abnormalities in the infrastructure.

Most of the sensors that are available in the market provide a digital output. However, when building the circuits, not all the manufactured devices have multiple digital pins. This situation is observed on NI – 3226 that has just four digital ports. Since the digital values are a 10-bit resolution in this research; an R-2R DAC circuit is built as shown in 5.2.2 R-2R ladder circuit to convert the digital values to analog and thus giving it to one analog port (AI0) of NI – 3226. The circuit can be used in all the applications that require Digital to Analog conversion of readings.

In 5.3 Measuring the voltage signal using WSN system, the pre-processed accelerometer signal is sent to the established WSN, and the output is received on the LabVIEW environment at the base station. The results obtained with an error of 0.65% as shown in Table 17.

The following things are achieved at the end of the research work:

- A detailed study on RCM and on WSN technologies by performing literature survey
- Building a WSN network and verifying its efficiency by conducting some trail experiments
- Signal Conditioning of the retrieved acceleration signal using MATLAB software
- Building the pre-processing circuit and establishing a communication between MATLAB and WSN system
- Receiving and logging the information received at the base station using NI manufactured WSN devices on the LabVIEW software

Moreover, the efficiency of the pre-processing circuit can be improved by adding some complex algorithms and the robustness of the WSN architecture can be improved by implementing various protocols. Also, by installing a greater number of nodes with various MEMS sensors in the WSN system, more RCM processes can be done.

Future Research:

The size of the vertical acceleration signal is 72000 samples, and after processing, the size of the signal is just three samples. The three samples are received at the base station with an error of 0.65%, and this may be ignored. The power consumed for transmitting the three samples can be calculated as follows:

Let, the power consumed for transmitting the complete vertical acceleration signal (72000) be P mW. Therefore,

100% = 72000 samples

3 samples = (3*100)/72000 = 0.004167 %

Therefore, the power consumed for transmitting three samples of a signal is 0.004167P mW. Thus it can be said that by decreasing the sample size of the signal before sending it to the WSN, a large proportion of battery power can be saved.

Also, the sampling interval and transmission interval on NI – 3226 sensor node play a vital role in saving the energy usage. It is realized that the minimum sampling interval of NI – 3226 WSN node should not be less than one second for the reception of signal without losing its characteristics. An experiment is conducted with the setup as shown in Figure 7 and by keeping the sampling and transmission intervals at 1 second. In this experiment, a sinusoidal is passed to NI – 3226 and is received continuously at the base station via WSN. By default, the sampling interval of NI - 3226 node is one second, and at this interval, each sample is transmitted to the gateway via radio message. The battery voltage of NI - 3226 is monitored using its Config Node properties. As seen in Table *18*, the battery life is significant, and thus the power saving can be increased by increasing the sampling and transmission intervals.

Table 18 Battery life for the change in transmission and sample intervals

N^0	Sample Interval	Transmission Interval	Battery life (months)
1	1 second	1 second	1.31
2	1 Second	5 second	2.46

User Memory (WSN):

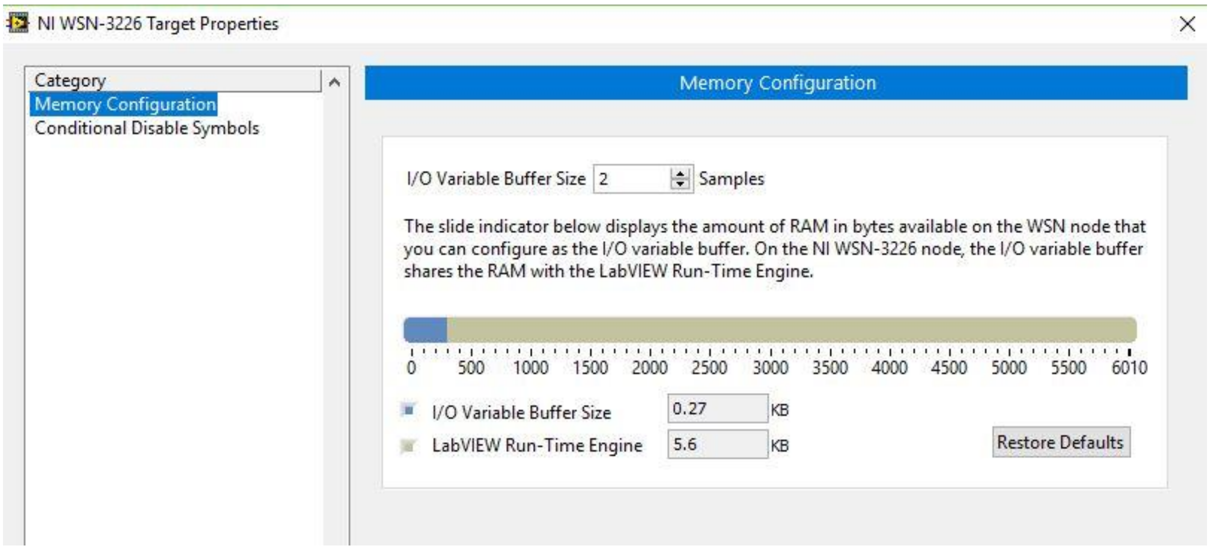
A portion of the flash memory can be accessed on the WSN NI 3226 node and is referred as the flash memory. The user memory can be used to store configuration data or any other important information that must persist across the power cycles and firmware constants.

Memory Configuration:

LabVIEW WSN Target is created for the particular application on the Project Explorer in the project. The memory configuration page contains the following parameters:

- I/O Variable Buffer size – This specifies the number of samples I/O the buffer can store. LabVIEW itself allocates the best value for the I/O variable buffer. The default buffer depth is two samples. Figure 33, displays the I/O Variable Buffer size
- Restore Defaults: This resets the value to its default value if any changes are made to the buffer size.

NI WSN- 3226 shares RAM with the LabVIEW Run-Time Engine.



NI WSN-3226 Target Properties

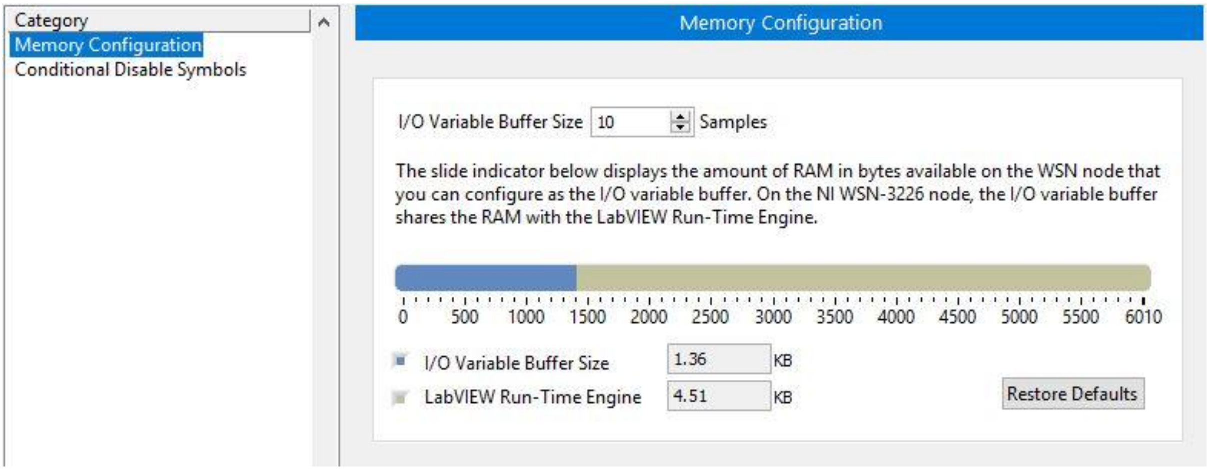
Memory Configuration

I/O Variable Buffer Size 2 Samples

The slide indicator below displays the amount of RAM in bytes available on the WSN node that you can configure as the I/O variable buffer. On the NI WSN-3226 node, the I/O variable buffer shares the RAM with the LabVIEW Run-Time Engine.

■ I/O Variable Buffer Size	0.27	KB
■ LabVIEW Run-Time Engine	5.6	KB

Figure 33 I/O Variable Buffer size of 2 samples



I/O Variable Buffer Size 10 Samples

The slide indicator below displays the amount of RAM in bytes available on the WSN node that you can configure as the I/O variable buffer. On the NI WSN-3226 node, the I/O variable buffer shares the RAM with the LabVIEW Run-Time Engine.

I/O Variable Buffer Size 1.36 KB

LabVIEW Run-Time Engine 4.51 KB

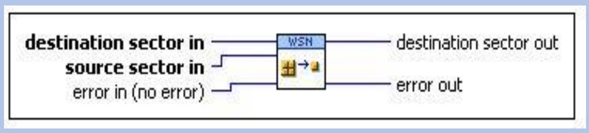
Figure 34 I/O Variable Buffer size of 10 samples

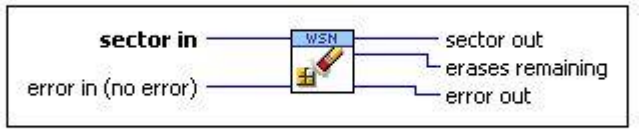
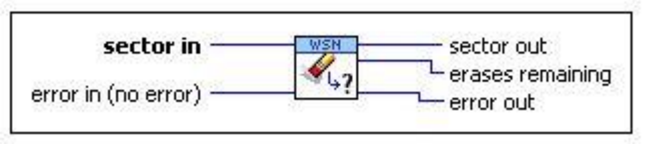
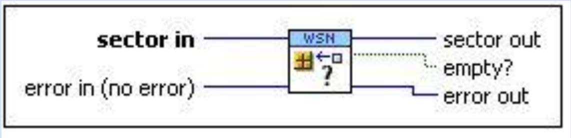
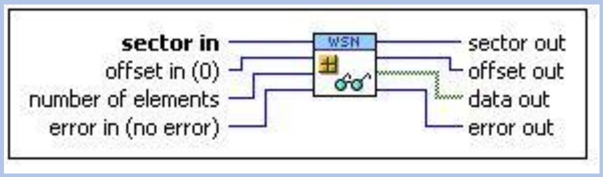
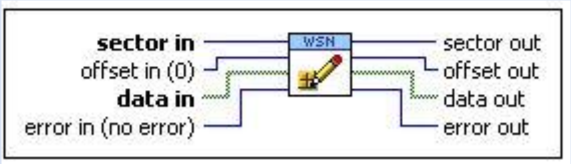
The I/O variable buffer is having an inverse relationship with the LabVIEW Run-Time Engine RAM. Figure 34, shows when the buffer size is changed from 2 samples to 10 samples. From the above figures, when the sample size is increased for I/O variable buffer, the LabVIEW Run-Time Engine RAM is decreased.

User Memory sector:

Using user memory VIs, the data in the user memory sector is accessed. National Instruments recommends avoiding the data logging to user memory because of the flash memory has a limited number of read/write cycles. NI WSN-3226 has 100K flash erase cycles per user memory sector. If the user continuously performs data logging onto the user memory, it is likely to exceed the life of the flash memory before the normal operational life of the WSN 3226 node. The User Memory VIs available in the LabVIEW is shown in Table 19.

Table 19 User Memory VIs

N^0	Palette Object	Symbol	Description
1	Copy Sector	 destination sector in source sector in error in (no error) WSN destination sector out error out	Copies all the data from the source sector to that of the destination in user memory

2	Erase Sector	 sector in error in (no error) WSN sector out erases remaining error out	Erases the specific user memory sector. The VI erases one 4KB user memory sector
3	Get Number of Erase Cycles Remaining	 sector in error in (no error) WSN sector out erases remaining error out	Returns the number of erase operations remaining for a user memory sector
4	Is Sector Empty	 sector in error in (no error) WSN sector out empty? error out	Checks if the sector is empty and returns TRUE if it is empty
5	Read from Sector	 sector in offset in (0) number of elements error in (no error) WSN sector out offset out data out error out	Reads data from specific user memory starting at byte offset. The offset is advanced by the length of the data
6	Write to Sector	 sector in offset in (0) data in error in (no error) WSN sector out offset out data out error out	Writes data into user memory sector. Later, the data is read to verify that the write is succeeded.

The parameter, Flash.SectorSize of the node on WSN module determines the size of the user memory sector. For NI WSN-3226, the user flash size is 188 Kbytes. The parameter, Flash.NumberofSectors of the node on WSN module returns the number of user memory sectors available. Thus the 188 Kbytes flash memory of NI – 3226 can be used to store a significant information and can be transmitted whenever it is required. The information can be stored, and the node can be put in sleep node without transmitting, thus the transmission energy required for this information can be saved.

References

1. "Network Rail - Office for Budget Responsibility", Office for Budget Responsibility, 2017. [Online]. Available: http://budgetresponsibility.org.uk/forecasts-in-depth/tax-by-tax-spend-by-spend/network-rail/. [Accessed: 29- Oct- 2017].
2. Hodge, V., O'Keefe, S., Weeks, M., & Moulds, A. (2015). Wireless Sensor Networks for Condition Monitoring in the Railway Industry: A Survey. IEEE Transactions On Intelligent Transportation Systems, 16(3), 1088-1106. doi: 10.1109/tits.2014.2366512
3. Dhillon, B. (2013). Safety and human error in engineering systems. Boca Raton, FL: CRC Press.
4. Ames, C. (2018). Backlog on local British bridge repairs jumps by £1bn - The Transport Network. Retrieved from https://www.transport-network.co.uk/Backlog-on-local-British-bridge-repairs-jumps-by-1bn/14756
5. Why Condition Monitoring is the Basis of Industrial IoT. (2018). Retrieved from https://www.seebo.com/condition-monitoring/
6. Points condition monitoring. Retrieved from https://www.balfourbeatty.com/expertise/rail-engineering/digital-technology/condition-monitoring/points-condition-monitoring/
7. Measurement. Retrieved from http://bby.the-group.net/expertise/rail-engineering/digital-technology/measurement/
8. Wayside Conditioning Monitoring | Smart Rail Solutions | Moxa. Retrieved from https://www.moxa.com/Solutions/Railway/Solution/Wayside%20Conditioning%20Monitoring.htm
9. Chefi, A., Soudani, A., & Sicard, G. (2013). Contribution to the design of a CMOS image sensor with low-complexity video compression for wireless sensor networks. Journal Of Systems Architecture, 59(10), 818-825. doi: 10.1016/j.sysarc.2013.07.010

10. Tseng, H., Sheu, S., & Shih, Y. (2011). Rotational Listening Strategy for IEEE 802.15.4 Wireless Body Networks. IEEE Sensors Journal, 11(9), 1841-1855. doi: 10.1109/jsen.2010.2096208

11. Bischoff, R., Meyer, J., Enochsson, O., Feltrin, G., & Elfgren, L. (2009). Event-based strain monitoring on a railway bridge with a wireless sensor network. In 4th International Conference on Structural Health Monitoring of Intelligent Infrastructure (SHMII-4) 2009. Zurich

12. Pascale, A., Varanese, N., Maier, G., & Spagnolini, U. (2012). A wireless sensor network architecture for railway signaling. In Proceedings of 9th Italian Networking Workshop. Italy.

13. Mansattha, M., & Benjapolakul, W. (2017). Automated Railway Crossing System Using ZigBee/IEEE 802.15.4 Standard. KMUTNB International Journal Of Applied Science And Technology. doi: 10.14416/j.ijast.2017.12.005

14. Scholten, H., Westenberg, R., & Schoemaker, M. (2009). Sensing Train Integrity. 2009 IEEE Sensors, (pp. 669-674).

15. Ni, Y., & Ye, X. (2014). Proceedings of the 1st International Workshop on High-Speed and Intercity Railways (pp. 415-430). Berlin: Springer Berlin.

16. Cañete, E., Chen, J., Díaz, M., Llopis, L., Reyna, A., & Rubio, B. (2015). Using Wireless Sensor Networks and Trains as Data Mules to Monitor Slab Track Infrastructures. Sensors, 15(7), 15101-15126. doi: 10.3390/s150715101

17. Giurgiutiu, V., Tomizuka, M., & Yun, C. (2006). Smart structures and materials 2006(pp. 406-418). Bellingham, Wash.: SPIE.

18. Milne, D., Le Pen, L., Watson, G., Thompson, D., Powrie, W., Hayward, M., & Morley, S. (2016). Providing MEMS Technologies for Smarter Railway Infrastructure. In The 3rd International Conference on Transportation Geotechnics (pp. 1077-1084). Guimarães, Portugal.: Procedia Engineering.

19. Wilson, J. (2011). Sensor technology handbook. Burlington: Elsevier (pp. 586-588). Oxford, UK.

20. Sekuła, K., & Kołakowski, P. (2010). Piezo-based weigh-in-motion system for the railway transport. Structural Control And Health Monitoring, 19(2), 199-215. doi: 10.1002/stc.416

21. Waki, H. (2010). Monitoring of Contact Line Equipment by Trains in Operation. JR EAST Technical Review, (17), 11-14. Retrieved from http://www.jreast.co.jp/e/development/tech/pdf_17/Tec-17-11-14eng.pdf

22. Wolfs, P., Bleakley, S., Senini, S., & Thomas, P. (2006). An Autonomous, Low Cost, Distributed Method for Observing Vehicle Track Interactions. In ASME Joint Rail Conference (pp. 279-286). Atlanta, GA: IEEE.

APPENDIX

Appendix 1

Appendix 1.1: Reading the .csv file

```
%% Reading data from the csv file
Signal = dlmread('accel_data.csv');
lateral_accel = Signal(:, 1);
long_accel = Signal(:, 3);
Vertical_accel = Signal(:, 2); %% Reads the second column of csv file
Sen_resolution = 0.0039; %% Obtained from the data sheet of the sensor
Acc_gravity = 9.807; %% Newton's constant
sen_vert_accel = Vertical_accel * Sen_resolution * Acc_gravity; %% Converts
value of g into m/s^2
sen_lateral_accel = lateral_accel * Sen_resolution * Acc_gravity;
sen_longitude_accel = long_accel * Sen_resolution * Acc_gravity;
sen_vertical_accel = round(sen_vert_accel * 100)/100; % two decimals remain
sen_later_accel = round(sen_lateral_accel * 100)/100; % two decimals remain
sen_long_accel = round(sen_longitude_accel * 100)/100; % two decimals
remain
N= length(Vertical_accel); %%length of sampled data
Sampling_time = 0.01;
Time_dummy = ([ 0.01: 0.01 : 745.62]);
Time = round(Time_dummy * 100)/100; % two decimals remain
figure
plot(Time, sen_vertical_accel, '-', 'Color', 'r');
hold on
plot(Time, sen_later_accel, '-', 'Color', 'b')
hold on
plot(Time, sen_long_accel, '-', 'Color', 'k');
title(' Tri-axial acceleration signal in m/s^2');
xlabel('Time( in sec)')
ylabel('Acceleration in m/s^2');
```

Appendix 1.2

Appendix 1.2.a: Plot the vertical acceleration signal

```
%% Plot the Vertical Acceleration signal in m/s^2
figure
subplot(2,1,1)
plot(Time, sen_vertical_accel, '-', 'Color', 'r');
xlim([700 745])
title('Vertical Acceleration signal in m/s^2');
xlabel('Time( in sec)')
ylabel('Acceleration in m/s^2');
```

Appendix 1.2.b: Plot the frequency spectrum of the vertical acceleration signal

```
%% Plot the frequency spectrum of the Single-Sided Vertical Acceleration
Magnitude spectrum
fax_bins = [0 : N-1];
Acc_magnitude = abs(fft(sen_vertical_accel));
fs = 1/Sampling_time;
```

```
fax_Hz = fax_bins*fs/N;
N_2 = ceil(N/2);
subplot(2,1,2)
plot(fax_Hz(1:N_2), Acc_magnitude(1:N_2), '-', 'Color', 'k');
xlabel('Frequency (Hz)')
ylabel('Magnitude(f)');
title('Single-sided Magnitude spectrum (Hertz)');
axis tight
xlim([0 3])
ylim([ 0 1500])

%% The frequency at which the jerk is identified
Mag_Fre = max( Acc_magnitude(1:N_2));
[mag_Fre, max_idx_mag_Fre] = max(Acc_magnitude(1:N_2))
max_val_Fre = fax_Hz((max_idx_mag_Fre))
```

Appendix 1.3

Appendix 1.3 a: Finding the threshold values

```
%% Finding the maximum  and minimum peaks above and below of threshold
levels (1.5 V and -1.5V) respectively
[ppk,plk] = findpeaks(sen_vertical_accel,Time,'MINPEAKHEIGHT', 1.5);
[npk,nlk] = findpeaks(-sen_vertical_accel,Time,'MINPEAKHEIGHT', 1.5);
figure
plot(Time,sen_vertical_accel,plk,ppk,'*')
hold on;
plot( nlk,-npk,'ko','MarkerFaceColor','k')
title('Peaks in Vertical Acceleration signal in m/s^2');
xlabel('Time( in sec)')
ylabel('Acceleration in( m/s^2)');
```

Appendix 1.3 b: Finding the jerks in the acceleration signal

```
%% Determine the sample at which the jerk is noticed in the vertical
acceleration
Vertical_max= max( sen_vertical_accel) %reads the peak value at which jerk
is noticed
Vertical_min= min( sen_vertical_accel);
[max_val_mag, max_idx] =max(sen_vertical_accel)
[min_val_mag, min_idx] =min(sen_vertical_accel);
max_val_time = Time(max_idx); % read the time at the max of amplitude
min_val_time = Time(min_idx); % read the time at the min of amplitude
Pos_Jerk = [max_val_time Vertical_max];%join the sample time and
corresponding max_peak amplitude on one array nx2
Neg_Jerk = [min_val_time Vertical_min];%join the sample time and
corresponding min_peak amplitude on one array nx2
figure
plot(Time,sen_vertical_accel,  max_val_time, Vertical_max, '*',
min_val_time , Vertical_min, 'ko','MarkerFaceColor','k');
title('Jerks identified in Vertical Acceleration signal');
xlabel('Time( in sec)')
ylabel('Acceleration ( m/s^2)');
```

Appendix 1.4: Scaling the values to 0 – 5 V

```
%% Converting Analog characteristic values to fall within the range of (0 -
5)
to_ref_max_accel = Vertical_max;        % positive acceleration values (0-
5)m/s^2 to range of (0-5)V
Max_Accel_reference = ((to_ref_max_accel *(5./5.)) *100/100);
to_ref_min_accel = Vertical_min;  % negative acceleration values (0 -  -
5)m/s^2 to range of (0-5)V
Min_Accel_reference = ((to_ref_min_accel *(5./6.)+5) *100/100)
to_ref_fre = max_val_Fre; % Frequency range of (0-5)Hz to range of (0-5)V
Fre_reference = ((to_ref_fre *(5./5.)) *100/100)
figure
ref_x = 0:0.01:6;
Accel_max_y = 0:0.01:6;
Accel_min_y = -6:0.01:0;
plot( ref_x,Accel_max_y)
hold on
stem(to_ref_max_accel,Max_Accel_reference)
hold on
stem(to_ref_fre, Fre_reference)
hold on
plot(ref_x, Accel_min_y, 'g')
hold on
stem(Min_Accel_reference, to_ref_min_accel)
title('Scaling analog characteristics to analog reference (0 - 5)');
xlabel(' Analog reference values in range of 0 - 5V');
ylabel('Analog characteristics values');
```

Appendix 1.5: Scaling the values to 0 – 1023

```
%% Converting important characteristic analog values to digital values
(decimal) in the range of 0 to 1023
e = [Max_Accel_reference Min_Accel_reference   Fre_reference]
Digital_zero  = zeros(length(e),1)
for ii = 1:length(e)
   Digital = round(e(ii) *(1023./5.) *100 /100)
   Digital_read = Digital
   Digital_reference(ii)   = Digital_read
end
figure
plot(e ,Digital_reference, e(1), Digital_reference(1), '*', e(2),
Digital_reference(2), 'o',e(3),
Digital_reference(3),'ko','MarkerFaceColor','g')
xlabel('Analog Reference values (0-5)')
ylabel('Digital values (0-1023)')
title(' Mapping Analog reference to Digital values')
```

Appendix 1.6: Binary conversion of decimal values

```
%% Digital reference (decimal) values to binary conversion
Digital_port = dec2bin(Digital_reference)
Vertical_max_bin = decimalToBinaryVector(Digital_reference(1), 10,
'MSBFirst')
Vertical_min_bin = decimalToBinaryVector(Digital_reference(2), 10,
'MSBFirst')
Max_val_fre_bin = decimalToBinaryVector(Digital_reference(3), 10,
'MSBFirst')
VMax = length(Vertical_max_bin);
VMin = length(Vertical_min_bin);
```

```
Fmax = length(Max_val_fre_bin);
Digital_default = '0000000000';
display(Digital_reference)
Digital1 = Vertical_max_bin
Digital2 = Vertical_min_bin
Digital3 = Max_val_fre_bin
```

Appendix 2

Appendix 2.1: Mapping bits to Digital pins of Arduino board

```
%% Assigning binary values to the respective digital pins of Arduino Uno R3
a= arduino();  %% creating arduino variable
display(a);
k = 11:-1:2     %% D11 -D2, the range of digital pins on arduino uno R3 for
10 bits
Port_D = transpose(zeros((length(k)),1))
Pin_number = k;
digital_order = input('Enter the case:  ');     %% Case statement for each
information
     switch(digital_order)
 case 'A'
    for j = 1:10 , m =  k
      D = Digital1(j);               %% Case A for Maximum Vertical
Acceleration
      D_D = D;
      Port_D(j)   = D_D
      D_pin = m               %% Digital pin numbers from 11 to 2
    end
 case 'B'
    for j = 1:10 , m =  k
      D = Digital2(j);
      D_D = D;                   %% Case B for Minimum Vertical
Acceleration
      Port_D(j)   = D_D
      D_pin = m               %% Digital pin numbers from 11 to 2
    end
   case 'C'
    for j = 1:10 , m =  k
      D = Digital3(j);
      D_D = D;                   %% Case B for Maximum Frequency
      Port_D(j)   = D_D
      D_pin = m               %% Digital pin numbers from 11 to 2
    end
 otherwise
     fprintf('It is an Illegal case');
       end

d11 = Port_D(1);
d10 = Port_D(2);
d9 = Port_D(3);
d8 = Port_D(4);
d7 = Port_D(5);
d6 = Port_D(6);
d5 = Port_D(7);
d4 = Port_D(8);
d3 = Port_D(9);
d2 = Port_D(10);
writeDigitalPin(a, 'D11', d11);
writeDigitalPin(a, 'D10', d10);
writeDigitalPin(a, 'D09', d9);  %% writing binary bits to arduino digital
pins D11 - D2
writeDigitalPin(a, 'D08', d8);
writeDigitalPin(a, 'D07', d7);
writeDigitalPin(a, 'D06', d6);
writeDigitalPin(a, 'D05', d5);
writeDigitalPin(a, 'D04', d4);
writeDigitalPin(a, 'D03', d3);
writeDigitalPin(a, 'D02', d2);
```