**Department of Electronics and Communication Engineering**

**CERTIFICATE**

This is to certify that the mini project report work entitled **“Rescue Bot”** carried out by **B. Kethan, S. Ranjith Kumar, S. Sai Vikas** Roll Number **15881A04J7, 15881A04P2**, **15881A04P5** submitted to the department of Electronics and Communication Engineering, in partial fulfillment of the requirements for the award of degree of **Bachelor of Technology** in

**Electronics and Communication Engineering** during the year 2018-2019.

**Name & Signature of the HOD**

**Prof. Dr. J V R Ravindra,**

**Head, ECE**

**Name & Signature of the Supervisor**

### Mr. N. Nagaraju,

### Associate Professor

A

Mini Project Report on

**Rescue Bot**

Submitted in the Partial Fulfillment of the

Requirements

for the Award of the Degree of

**Bachelor of Technology**

**in**

**Electronics and Communication Engineering**

**Submitted**

**By**

**B Kethan - 15881A04J7**

**S Ranjith Kumar - 15881A04P2**

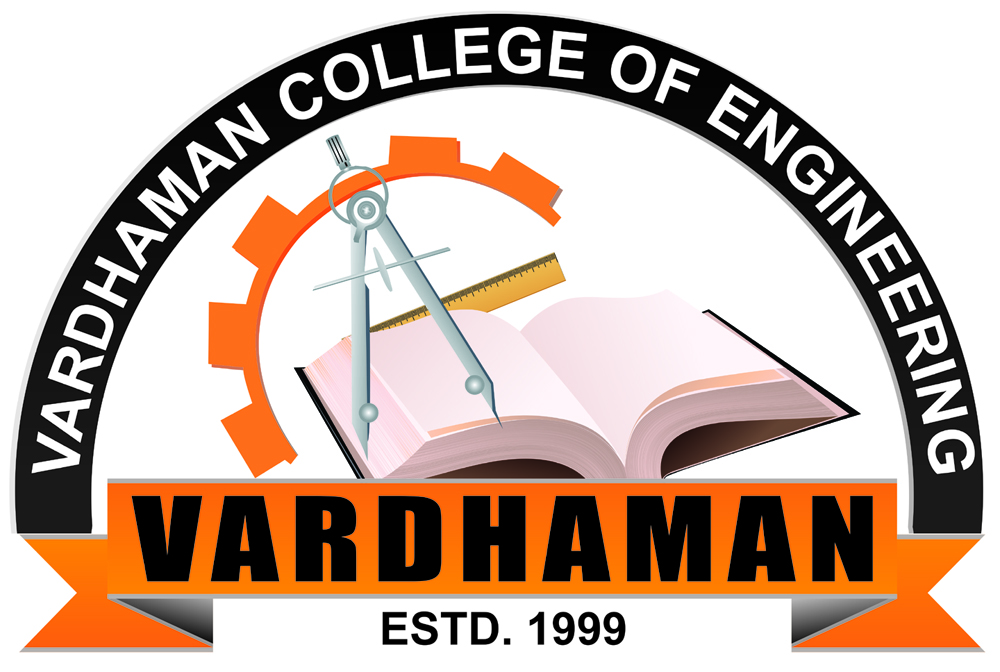
**S Sai Vikas - 15881A04P5**

**Under the esteemed guidance of**

**Mr. N. Nagaraju**

**Associate Professor**

**Department of ECE**



**Department of Electronics and Communication Engineering**

**Vardhaman College of Engineering**

**(AUTONOMOUS)**

**(Approved by AICTE, Affiliated to JNTUH & Accredited by NBA)**

**2018- 19**

**ACKNOWLEDGEMENT**

The satisfaction that accompanies the successful completion of the task would be put incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success.

I wish to express my deep sense of gratitude to **N. Nagaraju**, Associate Professor & Mini Project Supervisor, Department of Electronics & Communication Engineering, Vardhaman College of Engineering, for his able guidance and useful suggestions, which helped me in completing the mini project in time.

I am particularly thankful to **Prof. Dr. J V R Ravindra**, Head, Department of Electronics and Communication Engineering for his guidance, intense support and encouragement, which helped us to mould my project into a successful one.

I show gratitude to my honorable Principal **Dr. S. Sai Satyanarayana Reddy**, for having provided all the facilities and support.

I also thank all the staff members of Electronics and Communication Engineering department for their valuable support and generous advice. Finally, thanks to all my friends and family members for their continuous support and enthusiastic help.

B Kethan

(15881A04J7)

S Ranjith Kumar

(15881A04P2)

S Sai Vikas

(15881A04P5)

(ii)

**ABSTRACT**

In present day scenario natural calamities may happen at any time. We can’t guess what is going to happen in near future. When such things happen, we may come across situations where humans can’t do the work. A system which can go to every place and monitor the situation there is needed. This is the main idea behind this project. As the name suggests this bot is used in disaster management. During the crisis there are many places in which the humans can’t actually go and see what the situation over there is? Are there any persons alive or else to monitor the situation over there? In such cases this will be of a very great use. The way in which this works is quite unique. This will be controlled from the base station by the person through Wi-Fi and then it will have a camera over on its head and with the help of that we can get a real picture of scenario happening over there. This will work even in the worse condition like any type of terrain. This will move in all the eight directions with the help of the motors fixed to it. With the help of the camera we can get a real time monitoring of the situation.

(iii)

**CONTENTS**

|  |  |  |
| --- | --- | --- |
|  | Acknowledgements | (ii) |
|  | Abstract | (iii) |
| **1** | **INTRODUCTION** | **1** |
|  | * 1. Project Description   2. Need for Rescue Bot   3. System Design   4. System Module   5. Block diagram |  |
| **2** | **COMPONENTS REQUIRED** | **5** |
|  | 2.1 MQ6 Sensor  2.1.1 Features  2.1.2 Specifications  2.1.3 Pin Definition | 4 |
|  | 2.2 Camera  2.3 LED Light | 4 |
|  | 2.4 myRIO Hardware  2.4.1 Specifications  2.4.2 Launch The Getting Started Wizard  2.4.3 Hardware Overview  2.4.4 Connector pinouts | 5 |
|  | 2.5 LabVIEW Software  2.5.1 Hardware Support  2.5.2 LabVIEW Project Explorer  2.5.3 Building a LabVIEW VI | 6 |
| **3** | **WORKING AND RESULTS**  3.1 Working | **7** |

**4 CONCLUSIONS**

4.1 Future Scope

**5 REFERENCES**

**CHAPTER 1**

**INTRODUCTION**

LabVIEW is a programming environment and system design tool for a visual programming language, developed by American company National Instruments. If you’re electronics, electrical or instrumentation engineer, or a budding one, LabVIEW should definitely be on your radar, if it already isn’t, that is. This powerful platform is used worldwide in hundreds of fields, and the amount of scope for a LabVIEW programmer is nothing short of enormous.

National Instruments was founded in the early 1970’s by James Truchard, Jeff Kodosky and Bill Nowlin, who worked at the University of Texas, Austin. They were working on a U.S. navy project and realized that the data acquisition and collection methods they were using were highly inefficient, increasing time required to complete the project. Sometime later, they decided to create a product to ease their workload and this National Instruments was born. It was 1976, and testament to the fact that every great thing starts out small, one of the biggest names in the electronics and automation industry started out from garage.

Natural Calamities are a great cause to the destruction of land, life and property. There are many ways to help the people who are victims of these calamities. However, needed help is not getting to people who need it. Due to these calamities many lives are lost. We can’t get back what’s done, but we can control the extent of damage. When these disasters happen, sometime people get stuck at places where help can’t be reached. There must be a system to monitor their situation in those places. The situation should be monitored. There can also be dangerous gases when people are stuck at any deeper parts. The system should monitor both the life of people and search for harmful gases. It indicates whether harmful gases are present using LED.

**1.1 Project Description**

Natural Calamities are a great cause to the destruction of land, life and property. There are many ways to help the people who are victims of these calamities. However, needed help is not getting to people who need it. Due to these calamities many lives are lost. We can’t get back what’s done, but we can control the extent of damage. When these disasters happen, sometime people get stuck at places where help can’t be reached. There must be a system to monitor their situation in those places. The situation should be monitored. There can also be dangerous gases when people are stuck at any deeper parts. The system should monitor both the life of people and search for harmful gases. It indicates whether harmful gases are present using LED.

The primary objects of the present project to provide a novel means for the officials to monitor the situation of the people stuck at any place. There is a camera present in the system. It gives live feed of the people to the base station. The transmission can be over Wi-Fi or any communication protocol like TCP/IP or UDP. It can be accessed over Wi-Fi or remotely by providing static IP to the hardware present in the system. This can be done by creating a hotspot and connecting the hardware to the hotspot making it online. It then can be accessed by anyone who is having internet and LabVIEW software.

Another object of the present invention is to provide information about the harmful gases (if any) present at the places. This can alert the people about the dangerous situation. The sensor is used to detect the presence of a dangerous gases which are harmful and inflammable. This unit can be easily incorporated into a LED which indicates the presence of harmful gas. The sensor has excellent sensitivity combined with a quick response time. The sensor can also sense iso-butane, propane, LNG and cigarette smoke.

**1.2 Need for Rescue Bot**

Many lives are lost in disasters caused be it due to nature or self. So here we propose a system to monitor the situation of people who are stuck at places where help can’t be reached. It also detects harmful gases present at the situation and provides security alert to the people present over there. We propose to build the system using Camera and MQ2 gas detection sensor and interface it with myRIO using LabVIEW.

**1.3 System Design**

Lives of people are lost as there is no proper system to monitor their situation who are at places where help can’t be reached. This can be due to natural disaster or a self-cause. Their situation can be monitored by the system which is being developed here. It consists of a camera to get live feed of the situation and a MQ2 sensor which is used to detect the presence of any harmful gases. Camera and MQ2 sensor is connected to myRIO. MQ2 is connected to analog input of myRIO and its values are taken. When the values are greater than the threshold limit then the LED connected to myRIO glows and indicates the presence of harmful gas or any other gas. According to the value obtained by the sensor we can tell the gas is harmful or not. The camera connected to myRIO captures and provides input to myRIO as frames and these frames are analyzed by myRIO and sends feed to the base station. This can be done in two methods. When base is near we can use Wi-Fi provided to myRIO by connecting to myRIO. In second method we can provide a remote IP to myRIO and access it using static IP provided to it. This can be done from anywhere in the world. All you need is an internet connection and LabVIEW software to capture the live feed. Thus, it only transfers feed not only to base station, but also to higher authorities. In this way we can rescue the people who are stuck at any point.

**1.4 System Module**

The block diagram for the Rescue Bot is shown below. The system comprises of a Camera, MQ2 sensor, LED, PC installed with LabVIEW software, myRIO, Monitor, Harmful gases (if any). In the first step the Camera captures the frames and sends it to the LabVIEW software from myRIO. It also sends to values of gas sensor which are taken from the place to LabVIEW software where the values can be analyzed and stored. These values are compared to threshold values and if they reach threshold values then the LED is glows.

**1.5 Block Diagram**

POWER

SUPPLY

HARMFUL GASES

(if any)

MQ2

SENSOR

myRIO

LabVIEW

SOFTWARE

CAMERA

LED

**Figure 1.5 Block diagram for Rescue Bot**

**CHAPTER 2**

**COMPONENTS REQUIRED**

**2.1 MQ2 Sensor**

It is used for sensing LPG (composed of mostly propane and butane) and is often referred to as liquefied petroleum (LPG) sensor. The MQ2 sensor can detect gas concentrations anywhere within the range of 200-1000 ppm. It has high sensitivity and fast response time. It is highly sensitive to LPG iso-butane, propane and less sensitive to alcohol, cooking fume and cigarette smoke. Its resistance changes as the concentration of gas changes. It requires a power supply pf 5V (AC or DC). It consists of a Tin dioxide (SnO2) sensitive layer on the top, measuring electrode, and heater made of plastics and stainless steel. It has a Load resistance (RL) of about 20 K. The sensor has six pins out of which 4 pins are used for fetching signals and remaining 2 pins are used for providing heating current.

Sensitive material of MQ2 gas sensor is SnO2, which with lower conductivity in clean air. When the target flammable gas exists, the sensor’s conductivity gets higher along with the gas concentration rising. Users can convert the change of conductivity to correspond output signal of gas concentration through a simple circuit.MQ2 gas sensor can detect kinds of flammable gases, especially has high sensitivity to LPG (propane). It is a kind of low-cost sensor for many applications.



**Figure 2.1 MQ2 Gas Sensor**

**2.1.1 Specifications**

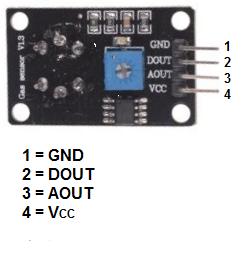
* Power supply needs: 5V
* Interface type: Analog
* Pin Definition: Pin DO - Digital Output, Pin AO - Analog Output, Pin 3 – VCC, Pin 4 - GND
* High Sensitivity to LPG, iso butane, propane
* Small sensitivity to alcohol, smoke
* Fast response
* Stable and long life
* Simple drive circuit

**2.1.2 Applications**

* Gas leak detection system
* Fire/Safety detection system
* Gas leak alarm
* Gas detector

**2.1.3 Pin Definition**

* Pin 1 - GND
* Pin 2 - DOUT
* Pin 3 - AOUT
* Pin 4 – VCC



**Figure 2.1.3 Pin description of MQ2 Sensor**

**2.2 Camera**

A camera is an optical instrument which is used for recording and capturing images, which may be stored locally, transmitted to another location, or both. The images may be individual still photographs or sequences of images constituting videos or movies. The camera is a remote sensing device as it senses subjects without any contact. The word camera comes from camera obscurra, which means “dark chamber” and is the Latin name of the original device for projecting an image of external reality onto a flat surface. The modern photographic camera evolved from the camera obscurra. The functioning of the camera is very similar to the functioning of the human eye.



**Figure 2.2 USB Camera**

**2.2 LED Light**

A **light-emitting diode** (**LED**) is a two-lead semiconductor light source. It is a pn junction diode that emits light when activated. When a suitable voltage is applied to the leads, electrons are able to recombine with electron holes within the device, releasing energy in the form of photons. This effect is called electroluminescent, and the color of the light (corresponding to the energy of the photon) is determined by the energy band gap of the semiconductor. LEDs are typically small (less than 1 mm2) and integrated optical components may be used to shape the radiation pattern. Infrared LEDs are still frequently used as transmitting elements in remote-control circuits, such as those in remote controls for a wide variety of consumer electronics. The first visible-light LEDs were also of low intensity and limited to red. Modern LEDs are available across the visible, ultraviolet, and infrared wavelengths, with very high brightness. LEDs have led to new displays and sensors, while their high switching rates are useful in advanced communications technology. LEDs have many advantages over incandescent light sources, including lower energy consumption, longer lifetime, improved physical robustness, smaller size, and faster switching. Light-emitting diodes are used in applications as diverse as aviation lighting, automotive light lamps, advertising, general lighting, traffic signals, camera flashes, and lighted wallpaper. They are also significantly more energy efficient and, arguably, have fewer environmental concerns linked to their disposal. Unlike a laser, the color of light emitted from an LED is neither coherent nor monochromatic, but the spectrum is narrow with respect to human vision, and for most purposes the light from a simple diode element can be regarded as functionally monochromatic.



**Figure 2.2 LED light**

**2.3 NI myRIO**

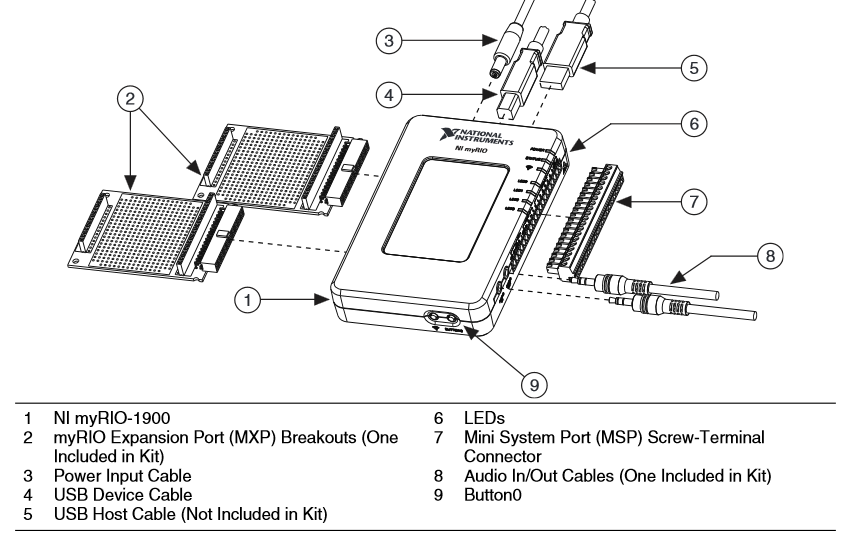
myRIO is a real-time embedded evaluation board made by National Instruments. It is used to develop applications that utilize its onboard FPGA and microprocessor. It requires LabVIEW.

|  |  |
| --- | --- |
| |  | | --- | | 1152107_BB_00_FB.EPS_1000.jpg | |

**Figure 2.3 NI myRIO**

**2.3.1 Specifications**

* Xilinx Z-7010 processor 667 MHz (ARM Cortex A9 x2 cores 28 nm process NEON SIMD, VFPv3 Vector Flash)
* Memory: NV: 256 MB, DDR3 512MB, 533 MHz, 16 bits
* FPGA type same as processor
* Wireless: IEEE 802.11 b, g, n ISM 2.4 GHz 20 MHz’s
* USB 2.0 Hi-Speed
* Breakout Board support
* 2 ports of 16 Digital I/O lines
* 3 axis accelerometers
* Max power consumption: 14 W
* Typical idle: 2.6 W
* LED's



Rather than spending copious amounts of time debugging code syntax or developing user interfaces, students can use the LabVIEW graphical programming paradigm to focus on constructing their systems and solving their design problems without the added pressure of a burdensome tool. LabVIEW differs from most other development platforms in the regard that it depends on ‘visual programming’ more than actual coding. In simple words, you use either predefined or custom components to complete programming tasks. For example, to create a reiterating series, all you need to do is drag and drop the corresponding loop functionality onto the block diagram. Add your conditions, outputs, connections, and you’re good to go!

The language used in LabVIEW is also called ‘G’, which is completely different from the numeric control programming language G-code or G programming language. G is a dataflow programming language, which means that it models it programs on directed paths. Or simply, it is a dynamic path-based language that changes outputs depending on variables in the program flow, which can be changed to reflect output changes. It is a highly versatile programming language that reflects changes in real time, suitable for real world applications.

**2.3.2 Launch the Getting Started WIZARD**

With the Getting Started Wizard, you can quickly observe the functional status of the NI myRIO unit. This wizard checks for connected NI myRIO devices, connects to the selected device, ensures that the software is up to date, suggests an update if the software is out of date, offers the option of renaming the device, and then shows a screen similar to a front panel that you can use to observe the accelerometer function, turn on and off onboard LEDs, and test the user-defined button. Using LabVIEW is deceptively easy in the beginning since it just involves connections and components, but once you dig deep, it gets much more complex. First of all, LabVIEW programs are called ‘virtual instruments’ or Vis, even the file extension is .vi.

Once you open it up and start a new project, you get two windows, namely, the front panel and block diagram. Both of these windows are used simultaneously to create your functions/programs. The front panel has components called ‘controls’ and ‘indicators’. The former refers to inputs, while the latter refers to outputs. There’s multiple forms and types, so depending on application, you can use any one, or multiple, for that matter

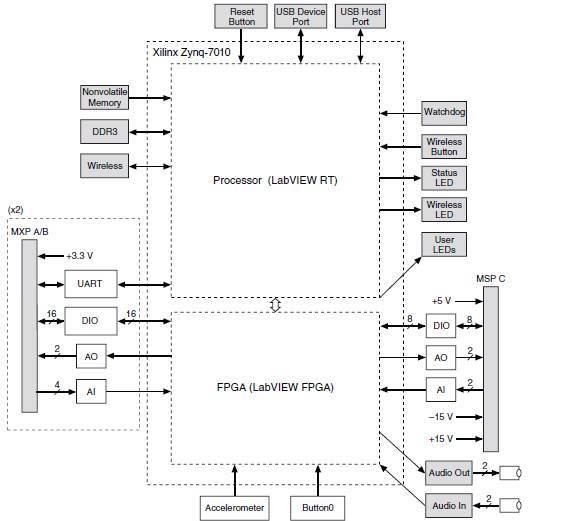
****

**Figure 2.3.2 Launch the Getting Started Wizard**

**2.3.3 Hardware Overview**

The NI myRIO-1900 provides analog input (AI), analog output (AO), digital input and output (DIO), audio, and power output in a compact embedded device. The NI myRIO-1900 connects to a host computer over USB and wireless 802.11 b, g, n.

The following figure shows the arrangement and functions of NI myRIO-1900 components

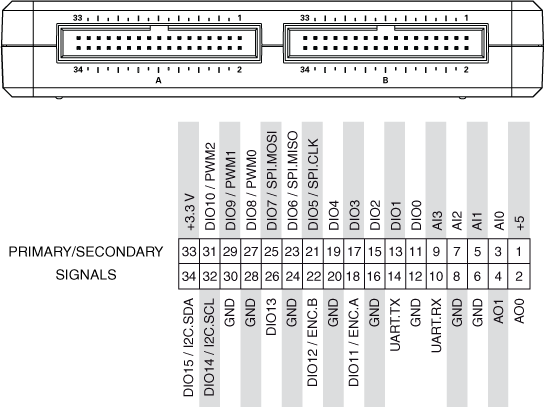
****

**Figure 2.3.3 NI-myRIO Hardware Block Diagram**

**2.3.4 Connector Pinouts**

NI myRIO-1900 Expansion Port (MXP) connectors A and B carry identical sets of signals. The signals are distinguished in software by the connector name, as in Connector A/DIO1 and Connector B/DIO1. Refer to the software documentation for information about configuring and using signals. The following figure and table show the signals on MXP connectors A and B.

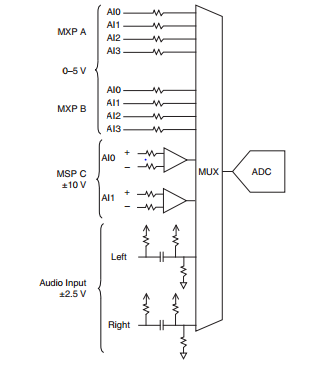
Note that some pins carry secondary functions as well as primary functions.



**Figure 2.3.4 Primary/Secondary Signals on MXP Connectors A and B**

**2.3.5 Analog Input (AI)**

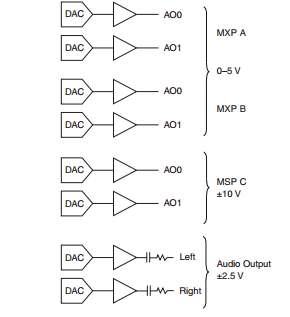
The NI myRIO-1900 has analog input channels on myRIO Expansion Port (MXP) connectors A and B, Mini System Port (MSP) connector C, and a stereo audio input connector. The analog inputs are multiplexed to a single analog-to-digital converter (ADC) that samples all channels. MXP connectors A and B have four single-ended analog input channels per connector, AI0-AI3, which you can use to measure 0-5 V signals. MSP connector C has two high impedances, differential analog input channels, AI0 and AI1, which you can use to measure signals up to ±10 V. The audio inputs are left and right stereo line-level inputs with a ±2.5 V full-scale range.



**Figure 2.3.5 Analog Input Channels**

**3.2.6 Analog Output (AO)**

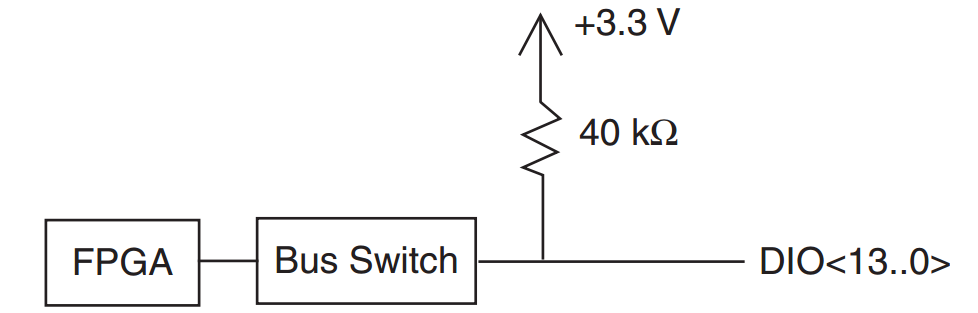
The NI myRIO-1900 has analog output channels on MyRIO Expansion Port (MXP) connectors A and B, Mini System Port (MSP) connector C, and a stereo audio output connector. Each analog output channel has a dedicated digital-to-analog converter (DAC), so they can all update simultaneously. The DACs for the analog output channels are controlled by two serial communication buses from the FPGA. MXP connectors A and B share one bus, and MSP connector C and the audio outputs share a second bus. MXP connectors A and B have two analog output channels per connector, AO0 and AO1, which you can use to generate 0-5 V signals. MSP connector C has two analog output channels, AO0 and AO1, which you can use to generate signals up to ±10 V. The audio outputs are left and right stereo line-level outputs capable of driving headphones.



**Figure 2.3.6 Analog Output Channels**

**3.2.7 Digital Input/output (DIO)**

The NI myRIO-1900 has 3.3 V general-purpose DIO lines on the MXP and MSP connectors. MXP connectors A and B have 16 DIO lines per connector. On the MXP connectors, each DIO line from 0 to 13 has a 40 kΩ pull up resistor to 3.3 V, and DIO lines 14 and 15 have 2.2 kΩ pull up resistors to 3.3 V. MSP connector C has eight DIO lines. Each MSP DIO line has a 40 kΩ pull down resistor to ground. DGND is the reference for all the DIO lines. You can program all the lines individually as inputs or outputs. Secondary digital functions include Serial Peripheral Interface Bus (SPI), I2C, pulse-width modulation (PWM), and quadrature encoder input. Refer to the NI myRIO software documentation for information about configuring the DIO lines.



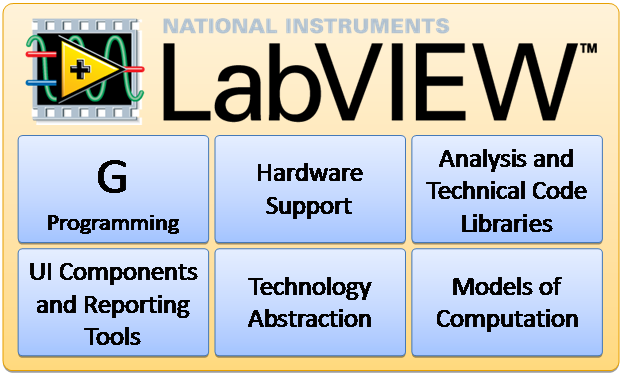
**Figure 2.3.7 DIO lines**

**Power Requirements**

* NI myRIO-1900 requires a power supply connected to the power connector
* Power supply voltage range..............................6-16 VDC
* Maximum power consumption .........................14 W
* Typical idle power consumption.......................2.6 W

**2.4 LabVIEW Software**

Laboratory Virtual Instrument Engineering Workbench (LabVIEW) is a system-design platform and development environment for a visual programming language from National Instruments. Originally released for the Apple Macintosh in 1986, LabVIEW is commonly used for data acquisition, instrument control, and industrial automation on a variety of operating systems (OSs), including Microsoft Windows and OS X. The latest version of LabVIEW is 2016, released in August 2016.



**Figure 2.4 LabVIEW Supporters**

**2.4.1 Hardware Support**

* Support for thousands of hardware devices, including:
  + Scientific instruments
  + Data acquisition devices
  + Sensors
  + Cameras
  + Motors and actuators
* Familiar programming model for all hardware devices
* Portable code that supports several deployment targets

**2.4.2 LabVIEW Project Explorer**

The LabVIEW Project Explorer helps you control resources that are all related to an application. These resources might include multiple VIs, controls, images, text documents, configuration information, build information, and deployment information. The project structure allows for quick and easy resource control, and you can use the LabVIEW Project Explorer to allocate resources to multiple devices (typically called targets). For those of you accustomed to other integrated development environments (IDEs), the LabVIEW Project Explorer will seem familiar and its usage will be mostly the same. LabVIEW is a cross-platform language with comprehensive support for its add-ons and libraries. You can write most VIs while targeting them to the development machine and easily retarget them to an NI device such as NI myRIO because these targets also run an OS that works with LabVIEW applications and add-on libraries. The LabVIEW Project Explorer hierarchy for an individual project does not reflect the organization of the source code files on the disk. Typically, a project is created in a directory and all of its source code is placed in the same directory with no folder system, even if the project view is highly organized. Moving files in the project view does not affect the actual disk copy of it—just the view of that file in the LabVIEW Project Explorer.

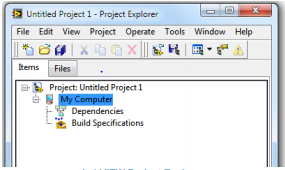
Complete the following steps to create a new project:

* Select the Create Project button on the LabVIEW Getting Started window.
* The Create Project window opens and offers several options. Explore all of the options by either creating a project or using the More Information link associated with any project template you are interested in.
* For now, select the Blank Project option as the starting point for the project.
* Then select the Finish button.

This creates an empty project and opens the LabVIEW Project Explorer. To save a new project:

* From the LabVIEW Project Explorer window, select File» Save.
* A save dialogue box opens and requests a destination directory and project name. Choose an appropriate location to save the project and give it a meaningful name.

Now the LabVIEW Project Explorer window is open and a brand new project is ready to be populated with resources and source code. The LabVIEW Project Explorer features a standard menu of familiar options (File, Edit, View, and so on) along with some options exclusive to LabVIEW. The LabVIEW Project Explorer itself has two panes—items, and files. The items page shows the items in a project organized in a hierarchy, while the files page shows all of the items in the project with associated files on the disk.

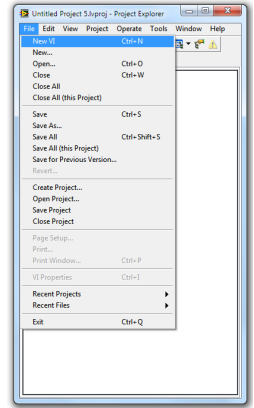


**Figure 2.4.2 LabVIEW Project Explorer Window**

In the items page of the LabVIEW Project Explorer, you can see the organization of the project. The project root, the first item on the list, shows which project you are working in. Under the project root, the next level of the tree contains all of the targets that the project is pointing to. A blank project defaults to the local target or “My Computer.” Under the My Computer target, the build specifications for the target are shown. Build specifications include the build configurations for source distributions and other builds available in LabVIEW toolkits and modules. You use these when deploying applications to embedded systems. While this workshop focuses on simple VIs, you need to understand the LabVIEW Project Explorer once you are developing more involved code. For now, however, the project is pretty void and doesn’t accomplish much. Build a simple VI in this project to dive into programming in LabVIEW.

**2.4.3 Building a LabVIEW VI**

To enter the programming portion of LabVIEW, you must create a new VI by selecting File» New VI in the LabVIEW Project Explorer.



**Figure 2.4 Building a LabVIEW VI**

LabVIEW provides an easy-to use tool for scientist and engineers who want to design custom data collection and automated tools for research or industrial applications. While there are plenty of graphical languages available, LabVIEW has the advantage of two decades' worth of development and wide-spread adoption, meaning code is already available for most devices and nearly any function you may want to run. As a programmer, all you have to do is determine how you want to interact with the device, and what you want to do with any collected data.

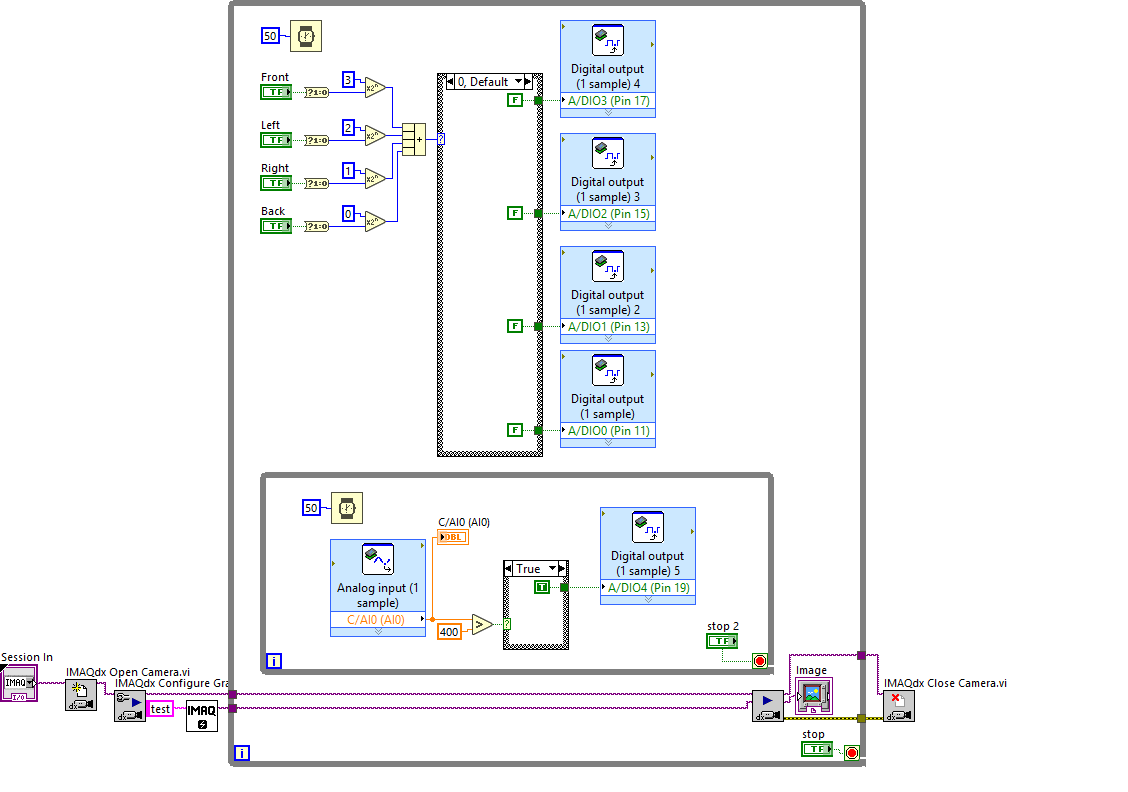
LabVIEW remains a proprietary tool, so in order to use it you will need to obtain a license. Many add-ons, as well, are only available through participation with National Instruments' app network, potentially adding to the cost of your LabVIEW projects. However, proprietary solutions also include better support than most open-source alternatives, which will come in handy for all those non-programmers using it.

**CHAPTER 3**

**WORKING AND RESULTS**

**3.1 Working**

**Block Diagram**



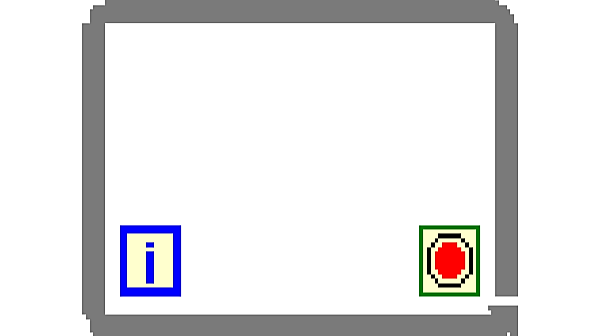
**Figure 3.1 Block Diagram for Rescue Bot**

**3.2 While loop**

While loop is a control flow statement that allows code to be executed continuously based on a given Boolean condition. It executes the sub diagram until the conditional terminal receives a particular Boolean value. It will continue to execute the condition until it is true. It is found in the structure palette under function palette of LabVIEW. The while loop will execute a condition at least once even when the condition is false.

**Building a While loop**

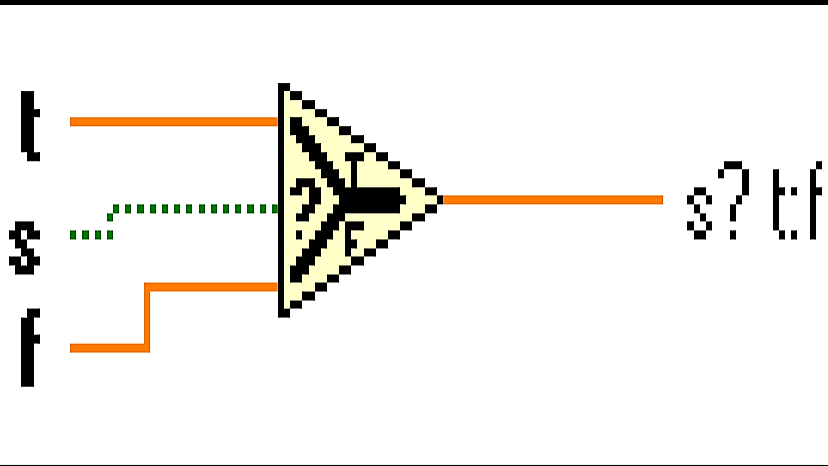
* Open a new VI. You can open a blank VI by selecting File » New VI
* If the Functions palette is not visible, right-click any blank space on the block diagram to display a temporary version of the palette. Click the thumbtack in the upper left corner of the Functions palette to pin the palette so it is no longer temporary.
* Select the while loop from the Structures palette under the Functions palette



**Figure 3.2 While loop**

**3.3 Selector**

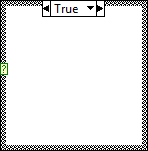
It returns the value wired to T or F input depending on the value of S. The function returns the value wired to T if S is TRUE. If S is false it returns the value wired to F, if S is FALSE. The selector function is selected by clicking the select function from the comparison palette.



**Figure 3.3 Selector Terminal**

**3.4 Case Selector**

It is similar to switch case in other programming languages. It has different cases where a single case is selected according to the given input. It can take data types like Boolean, Integer, String, Enum and Cluster. Every input should consist of two or more cases where different cases can be selected. Cases are based on the type of data. For example, if the data is Boolean there are only two cases TRUE and FALSE. It changes according to the input given to the case selector.



**Figure 3.4 Comparator**

**3.5 Indicator**

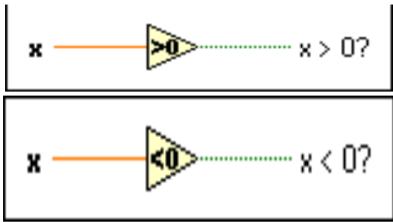
Here we use LED and toggle indicators. Whenever there is a gas leakage, it will be indicated as a numeric in the front panel the LED will glow correspondingly to the leak value. If it crosses the limit, the toggle switch turns on.



**Figure 3.5 Indicator**

**3.6 Comparator**

Greater than and less than are the two comparison tools used to compare the input value with the set value.



**Figure 3.6 Comparator**

**3.7 Timer**

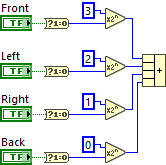
Timer is used for waiting for the process to complete. Timer specifies how much time to wait. It has a constant control which specifies the time to wait. It takes control in the form of milliseconds. It is important when CPU is assigned to another process. It makes the CPU free from the process with out having load on other processes.



**Figure 3.7 Timer**

**3.8 Boolean Controllers**

These controllers are used to control the bot in all the eight directions. The Boolean controls are used to on or off the controller. When the Boolean is ON, the output of the conditional terminal will be 1 and when the Boolean is OFF, the output is 0. It is converted into decimal by addition block where all the outputs of the terminal are given. Its output depends on which Boolean is ON and which is OFF. These are given to case structure. Where four different cases are designed and output of the respective block is executed. This can also be implemented by Enum, but using Boolean data type makes the memory less. Enum takes integer datatype whereas Boolean takes bit datatype. Enum occupies 16 bytes of data whereas Boolean occupies 4 bits of size.



**Figure 3.8 Control Selector**

**3.9 NI myRIO Digital Output**

As mentioned in Chapter – 2 NI myRIO has Digital Input/output. Digital Output is the one which is used to get the output from myRIO. When case selectors are selected their respective cases are executed. Bot is moved with the help of the case selector. All the four directions are embedded into the cases. Desired cases can be selected with the help of Boolean selectors. Four desired controls and a default control are present in the case selector from which the output is given to the myRIO digital output pins.



**Figure 3.9 Digital Output**

**3.10 Analog Output**

Analog output gives the analog value of the sensor attached to it. The output of the sensor can be obtained by just attaching the analog pin of the sensor to myRIO analog output. The output of the analog output is given to the indicator. The value of the indicator is kept as a reference value. We can fix a value as threshold value which is minimum of the gas value present at environment. By comparing the threshold value with the output of the gas sensor we can tell the gas is harmful or not. The sensitivity of the input gases can be adjusted with the help of potentiometer. Thus, analog output can be used to obtain an analog value from a sensor.



**Figure 3.10 Analog Output**

**3.11 IMAQdx Open Camera VI**

Opens a camera, queries. the camera for its capabilities, loads a camera configuration file, and creates a unique reference to the camera. Use the IMAQdx Close Camera VI when you are finished with the reference



**Figure 3.11 IMAQdx Open Camera VI**

**3.12 IMAQdx Configure Grab VI**

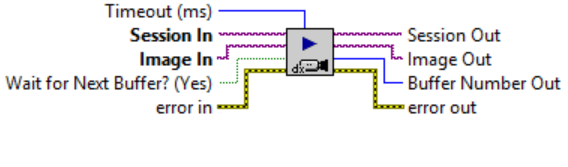
Configures and starts a grab acquisition. A grab performs an acquisition that loops continuously on a ring of buffers. Use the grab VI for high-speed image acquisition. Use the IMAQdx Grab VI to copy an image out of the buffer. If you call this VI before calling the IMAQdx Open Camera VI, the IMAQdx Configure Grab VI uses cam0 by default. Use the IMAQdx unconfigure Acquisition VI to unconfigure the acquisition

****

**Figure 3.12 IMAQdx Configure Grab VI**

**3.13 IMAQdx Grab2 VI**

Acquires the most frame into image out. Call this VI only after calling IMQdx configure Grab VI. If the image type dose not match the video format of the camera, the VI changes the type to a suitable format.



**Figure 3.13 IMAQdx Grab2 VI**

**3.14 IMAQdx Close Camera VI**

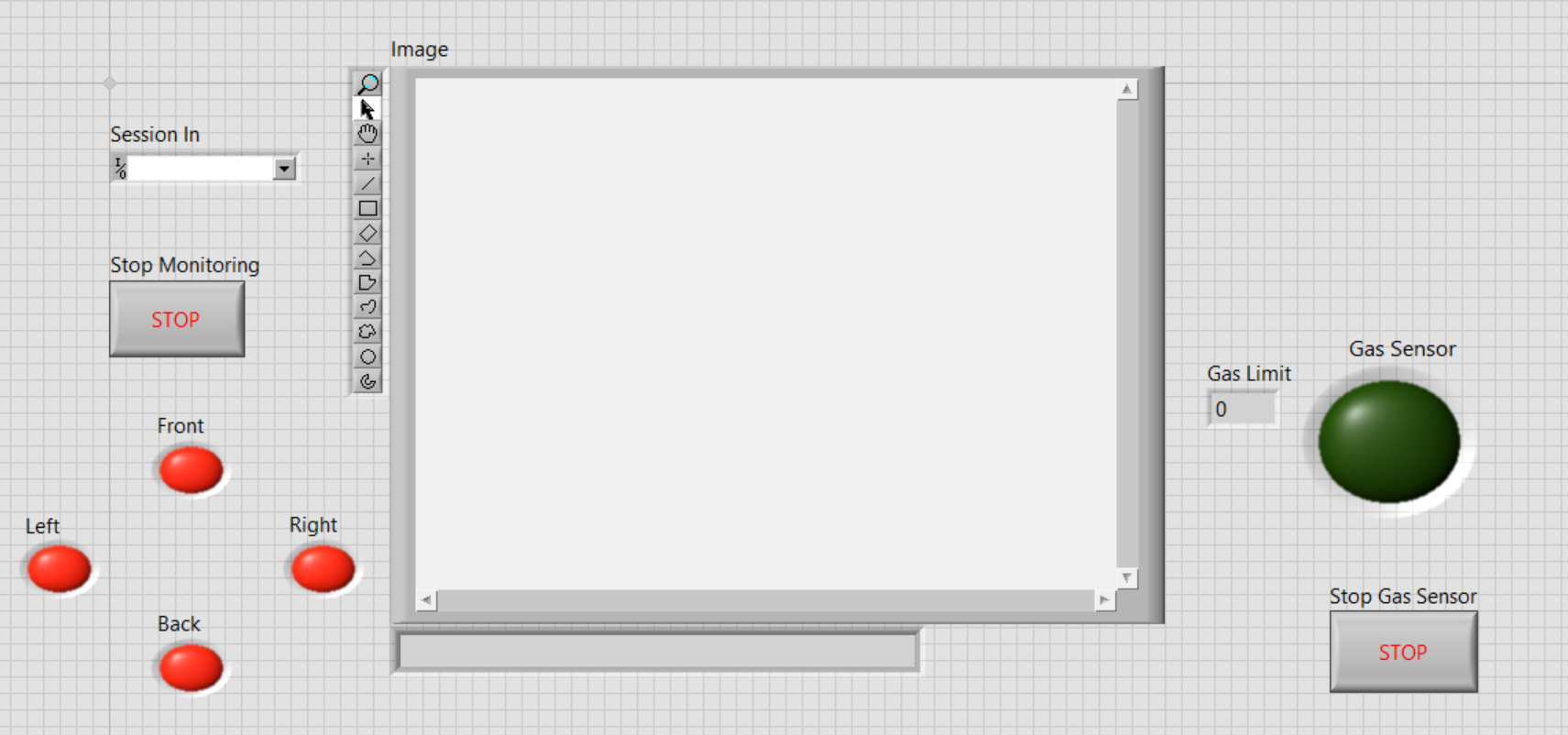
Stops an acquisition in progress, releases resources associated with an acquisition, and closes the specified Camera Session.

****

**Figure 3.14 IMAQdx Close Camera VI**

**3.15 Front Panel**

Hence our system is developed to monitor the situation of the places where disasters happen. Bot is used to travel the places where human can’t reach. With the help of camera fixed to the bot we can ponder the situation over there. Gas sensor attached to the bot detect for combustible gases. If present it gives signal to both the people who are present there as well as to the people who are monitoring. All this is done with the help of myRIO connected to the host computer with the help of Wi-Fi. All the data communication is done with the help of Wi-Fi.

****

**Figure 3.15 Front Panel for Rescue Bot**

**CHAPTER 4**

**CONCLUSIONS**

Overall system was designed and tested by introducing the small amount of LPG gas sensor module. The system detects the level of gas in the air if it exceeds the safety level it activates the LED Light which includes the Alert. The user at home in abnormal condition and to take the necessary condition.

**4.1 Future Scope**

In future, we would like to implement it in various domains. We are also planning to implement it with the help of TCP/IP protocol where data is transmitted in this protocol. Transferring data in this protocol not only makes the transfer fast, but also makes it secure and robust. When data is transmitted in this way it can be accessed from anywhere in the world. An Android application can also get data via TCP/IP protocol thus implementing it to largest source in world.

**REFERENCES**

[1] Modelling and simulation of complex control systems using LabVIEW, P.srinivas, P. DurgaPrasadaRao, K .Vijaya Lakshmi

[2] Hanwei Electronics Co. Ltd, MQ 2 sensor Datasheet.

[3] Introduction to Real-time Control using LabVIEW, Ch.Salzmann, D.Gillert, P.Huguenin

[4] Design and Implementation of DC Motors, Ashish Shrivastava, Raghul veerma, Rajesh kumar, Ratnesh Prabhaker.

[5] Current Teaching and Research Projects using LabviewControl Design and Simulation, Kim Lankford, National Instruments, VIP 2005.

[6] N.NithyaRani “Implementation of OPC-Based Communication Between Temperature Process and DCS on LabVIEW Platform” in the “BEST: International Journal of Management, Information Technology and Engineering (BEST: IJMITE)” Vol. 1, Issue 1, Oct 2013.

[7] N.NithyaRani and S.Ranganathan “Advances in Control Techniques and Process Analysis with LabVIEW and DCS” in the “International Journal of Electronics, Communication & Instrumentation Engineering Research and Development (IJECIERD)” June 2013; Volume : 3; Issue : 2; June 2013.

[8] N.NithyaRani “Advanced Process Analysis on LabVIEW” in the “International Journal of Advanced Research in Electrical and Electronics Engineering (IJAREEE)” Vol.1, No.1 November 2013.