ISSN:2348-2079

Volume-7 Issue-1

International Journal of Intellectual Advancements and Research in Engineering Computations

# Automatic Weigh Bridge Monitoring Using Image Processing Technique in LABVIEW

**K.S. Hariprasath[1], G.Swaathe[1], G. Praveen santhoshkumar[2]**

[1]UG Scholars, Department of Electronics and Instrumentation Engineering,
Nandha Engineering College, Autonomous, Erode.

[2]Associate Professor,Department of Electronics and Instrumentation Engineering,
Nandha Engineering College, Autonomous, Erode.

## ABSTRACT

This Project is designed to reduce human errors and increase the accuracy of the process. This project replaces the RFID method to the modern Image Processing method using LabView. Earlier RFID tags are used to identify the vehicle and feed data to the system about the load carried by the vehicle, from/to details, etc. The problem arises when entering the data to an incorrect database or when the RFID is missing. To overcome this issue the new system replaces the use of RFID tags to Number Plate recognition using Image Processing in LabView.

## INTRODUCTION

Project is made using Arduino Microcontroller interfaced to LabView system. This project uses load cell as a weigh bridge prototype to measure weight. A high quality camera is used to recognize the number plate of the vehicle and to fetch its data from the database like its origin, destination, type of good, etc. after measuring the weight the load details are logged to the corresponding database of the vehicle, hence there is no chance for mismatch entries done in the earlier RFID system. This project can be implemented in weigh bridges, industries, factories, etc [1-3]

## IMAGE PROCESSING

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Nowadays, image processing is among rapidly growing technologies. It forms core research area within engineering and computer science disciplines too.

Image processing basically includes the following three steps:

- Importing the image via image acquisition tools;
- Analysing and manipulating the image;
- Output in which result can be altered image or report that is based on image analysis.

There are two types of methods used for image processing namely, analogue and digital image processing. Analogue image processing can be used for the hard copies like printouts and photographs. Image analysts use various fundamentals of interpretation while using these visual techniques. Digital image processing techniques help in manipulation of the digital images by using computers. The three general phases that all types of data have to undergo while using digital technique are pre-processing, enhancement, and display, information extraction. [4-7].

**Author for correspondence:**
Department of Electronics and Instrumentation Engineering, Nandha Engineering College, Autonomous, Erode.

# EQUIPMENT

## Hardware Equipment

**Liquid crystal display (LCD)**

**Arduino Uno**

The Microcontroller or the processing module is an interfacing and controlling module, that interfaces the various peripherals and other modules used in the circuit. It integrates the function of various modules such as the Zero Crossing Detector (ZCD), X-OR gate, Relay driver (ULN2003A) etc. The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

**Fig. 1. Arduino Uno is used to interface software and hardware through parallel communication**

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector. The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.[8-10].

# AUTOMATIC (SOFTWARE) RESET

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of theATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 □farad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload. This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data. The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details

## LIQUID CRYSTAL DISPLAY (LCD)

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom characters (unlike in seven segments), animations and so on. A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data. The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD.


**Fig. 2. Liquid Crystal display is to display the load measured in weigh bridge using load cell**

## SIMULATION AND HARDWARE RESULTS

### Labview

LabVIEW(short for Laboratory Virtual Instrument Engineering Workbench) is a system-design platform and development environment for a visual programming language from National Instruments.LabVIEW is a graphical programming environment used by thousands of engineers and scientists to develop sophisticated control systems using graphical icons and wires that resemble a flowchart. It offers integration with thousands of hardware devices and provides hundreds of built-in libraries for advanced control, analysis, and data visualization all for creating user-defined systems more quickly. The LabVIEW platform is scalable across multiple targets and OSs, and, in the case of Compact RIO, LabVIEW can be used to access and integrate all of the components of the LabVIEW reconfigurable I/O (RIO) architecture.

### Graphical programming

LabVIEW ties the creation of user interfaces (called front panels) into the development cycle. LabVIEW programs/subroutines are called virtual instruments (VIs). Each VI has three components: a block diagram, a front panel and a connector panel. The last is used to represent the VI in the block diagrams of other, calling VIs. The front panel is built using controls and indicators. Controls are inputs – they allow a user to supply information to the VI. Indicators are outputs – they indicate, or display, the results based on the inputs given to the VI. The back panel, which is a block diagram, contains the graphical source code. All of the objects placed on the front panel will appear on the back panel as terminals. The back panel also contains structures and functions which perform operations on controls and supply data to indicators. The structures and functions are found on the Functions palette and can be placed on the back panel. Collectively controls, indicators, structures and functions will be referred to as nodes. Nodes are connected to one another using wires – e.g. two controls and an indicator can be wired to the addition function so that the indicator displays the sum of the two controls. Thus a virtual instrument can either be run as a program, with the front panel serving as a user interface, or, when dropped as a node onto the block diagram, the front panel defines the inputs and outputs for the

given node through the connector panel. This implies each VI can be easily tested before being embedded as a subroutine into a larger program.

The graphical approach also allows non-programmers to build programs by dragging and dropping virtual representations of lab equipment with which they are already familiar. The LabVIEW programming environment, with the included examples and documentation, makes it simple to create small applications. This is a benefit on one side, but there is also a certain danger of underestimating the expertise needed for high-quality G programming. For complex algorithms or large-scale code, it is important that the programmer possesses an extensive knowledge of the special LabVIEW syntax and the topology of its memory management. The most advanced LabVIEW development systems offer the possibility of building stand-alone applications. Furthermore, it is possible to create distributed applications, which communicate by a client/server scheme, and are therefore easier to implement due to the inherently parallel nature of G.

The image above is an illustration of a simple LabVIEW program showing the dataflow source code in the form of the block diagram in the lower left frame and the input and output variables as graphical objects in the upper right frame. The two are the essential components of a LabVIEW program referred to as a Virtual Instrument.

## Interfacing

A key of LabVIEW over other development environments is the extensive support for accessing instrumentation hardware. Drivers and abstraction layers for many different types of instruments and buses are included or are available for inclusion. The abstraction layers offer standard software interfaces to communicate with hardware devices. The provided driver interfaces save program development time. The sales pitch of National Instruments is, therefore, that even people with limited coding experience can write programs and deploy test solutions in a reduced time frame when compared to more conventional or competing systems. A new hardware driver topology (DAQ mxBase), which consists mainly of G-coded components with only a few register calls through NI Measurement Hardware DDK (Driver Development Kit) functions, provides platform independent hardware access to numerous data acquisition and instrumentation devices.

## Front panel

When created a new VI or selected an existing VI, the Front Panel and the Block Diagram for that specific VI will appear.

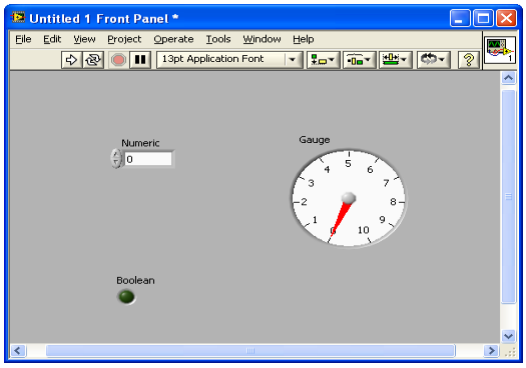


**Fig. 3. Front panel of the labview software is to display the parameters used and measured in the circuit**

In LabVIEW, build a user interface, or front panel, with controls and indicators. Controls are knobs, push buttons, dials, and other input devices. Indicators are graphs, LEDs, and other displays. To build the front panel with controls and indicators, which are the interactive input and output terminals of the VI respectively..Controls simulate instrument input devices and supply data to the block diagram of the VI. Indicators simulate instrument output devices and display data the

block diagram which was shown in Fig.5.7 acquires or generates.

## BLOCK DIAGRAM

After build the user interface, the code can be built using VIs and structures to control the front panel objects. The block diagram contains this code. In some ways, the block diagram which was shown in Fig.3.8 resembles a flowchart. After the front panel was built, the code can be added using graphical representations of functions to control the front panel objects.

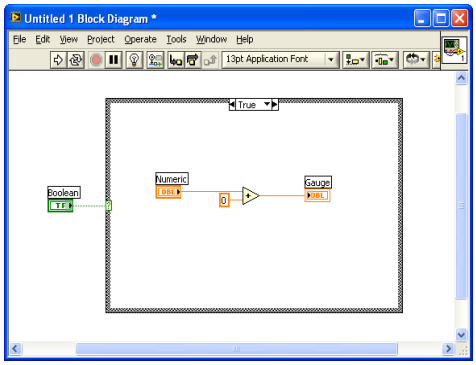


**Fig. 4. Block diagram of the circuits are shown in this figure**

The block diagram contains this graphical source code. Front panel objects appear as terminals, on the block diagram. Block diagram objects include terminals, subVIs, functions, constants, structures, and wires, which transfer data among other block diagram objects.

## LABVIEW COMPONENTS USED

### Control palette

The Controls and Functions palettes contain sub palettes of objects that can use to create a VI. When click a sub palette icon, the entire palette changes to the sub palette was selected. To use an object on the palettes, click the object and place it on the front panel or block diagram. The Controls palette is available only on the front panel. contains the controls and indicators use to build the front panel. The most used Sub Palettes are the Numeric Sub Palette, the Boolean Sub Palette and the String & Path Sub Palette.

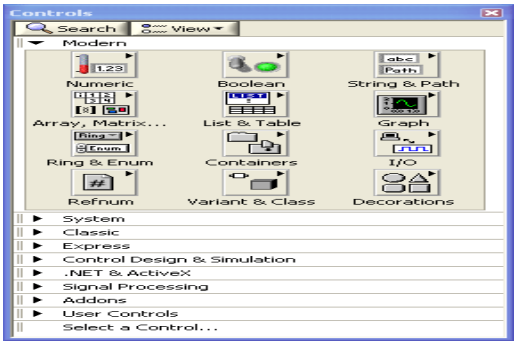


**Fig. 5. Control panel helps in picking the components from the list. It consists of all the control elements in the software.**

Pin the palette, so it is always visible, just click the little pin button in the upper left corner of the palette. To change the content and appearance of the palette, click the "View" button.

### Function palette

The Functions palette is available only on the block diagram. The Functions palette which was shown in Fig.3.14 contains the VIs and functions use to build the block diagram.

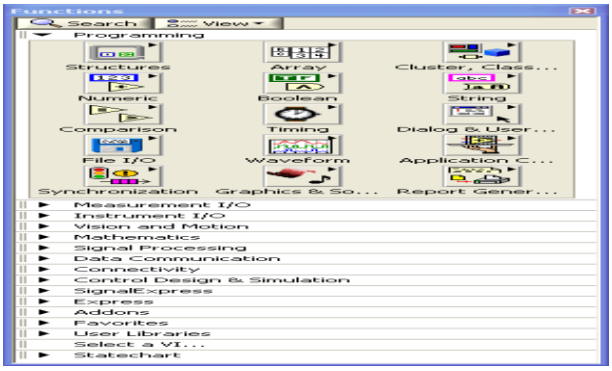


**Fig. 6. Functions palette**

## WAVEFORM GRAPHS

The waveform graph displays one or more plots of evenly sampled measurements. The waveform graph plots only single-valued functions, as in $y = f(x)$, with points evenly distributed along the x-axis, such as acquired time-varying waveforms. The following front panel shows an example of a waveform graph. The waveform graph can display plots containing any number of points. The graph also accepts several data types, which minimizes the extent to which you must manipulate data before you display it. The waveform graph accepts several data types for single-plot waveform graphs. The graph accepts a single array of values, interprets the data as points on the graph, and increments the $x$ index by one starting at $x = 0$.

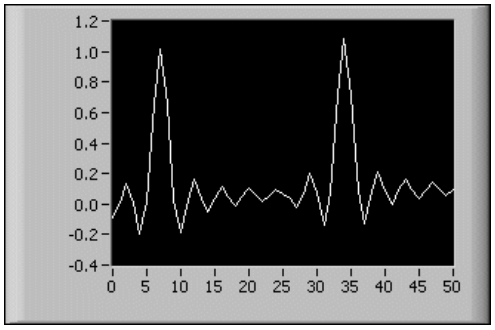


**Fig. 7. Sample waveform that will be obtained in software**.

The graph accepts a cluster of an initial $x$ value, a delta $x$, and an array of $y$ data. The graph also accepts the waveform data type, which carries the data, start time, and delta $t$ of a waveform. The waveform graph also accepts the dynamic data type, which is for use with Express VIs. In addition to the data associated with a signal, the dynamic data type includes attributes that provide information about the signal, such as the name of the signal or the date and time the data was acquired. Attributes specify how the signal appears on the waveform graph. When the dynamic data type includes a single numeric value, the graph plots the single value and automatically formats

the plot legend and x-scale time stamp. When the dynamic data type includes a single channel, the graph plots the whole waveform and automatically formats the plot legend and x-scale time stamp.

## CONTROLS AND INDICATORS

Use the system controls and indicators located on the **System** palette in dialog boxes you create. The system controls and indicators are designed specifically for use in dialog boxes and include ring and spin controls, numeric slides, progress bars, scroll bars, list boxes, tables, string and path controls, tab controls, tree controls, buttons, checkboxes, radio buttons, subpanels, and an opaque label that automatically matches the background colour of its parent. These controls differ from those that appear on the front panel only in terms of appearance. These controls appear in the colours you have set up for the system. The system controls change appearance depending on which platform you run the VI. When you run the VI on a different platform, the system controls adapt their color and appearance to match the standard dialog box controls for that platform.

### Filepath

When files are located beneath certain special folders, callers that link to these files will link to them using a symbolic path, rather than an absolute or relative path. The symbolic path will be relative to the special folder.

## WORKING

In this project the weight of the vehicle is measured through load cell and displayed by the LCD. Here, the camera captures the image of vehicle's number and process the data related to database and check whether the image processed and and the number in the database are same. If that matches, then the data are stored in database for future use. RFID card is also used to store database but due to some human errors image processing technique is used.

### Simulation Model Oflabview

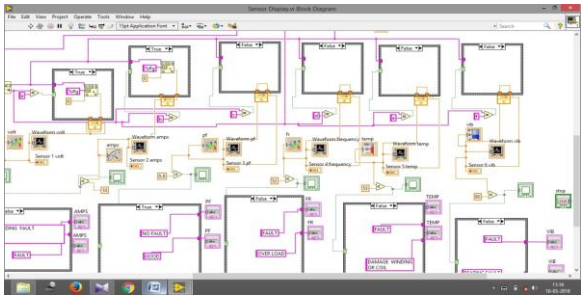

**Fig. 8. Circuit diagram designed in Labview.**

## RESULT AND CONCLUSION

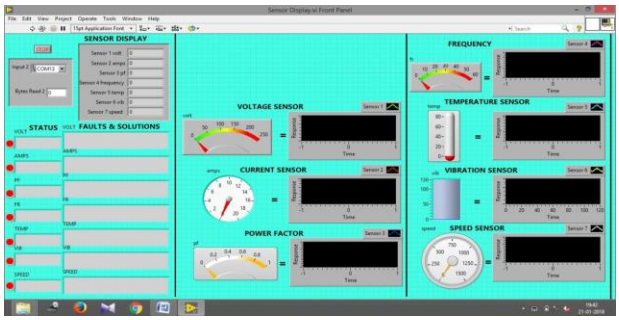


**Fig. 9. Final output obtained on simulating the circuit**.

The automatic monitoring of weigh bridge is designedand validated successfully for monitoring the weigh bridge and the materials are stored in database by image processing techniqueusingArduino Uno. The system was validated successfully by test conditions. Experimental trials confirm the accuracy and validity of the prototype. Behaviour and performance of the automatic monitoring system was found to be as per expectations with minimal deviations.

## REFERENCES

[1]. L. Eikvil, "Otical Character Recognition," December 1983.
[2]. A. Turing, "Mind: Computing Machinery and Intelligence," 433-460, 1950.
[3]. B. &. F. Auditors, "Nzoia Sugar Factory Audit Report, Physical Year 2011," Nairobi, 2011.
[4]. A. A. Aburas and A. S. Rehief, "JPEG for Arabic Handwritten Character Recognition. Add a dimension of application,"Internatioanl Islamic University, 2009.
[5]. A. Rani, K. Ramakrish and M. Kantikiran, "A novel approach for Indian Licence Plate Recognition," IEEE Explore, 2010.
[6]. L. D. Jackel, M. Y. Battista and C. E. Stenard, "Neural Net Applications in Character Recognition and Document analysis," AT&T Laboratories , Holmdel NJO7733 USA.
[7]. S. A. Angadi and M. M. Kodabagi, "Text Region Extraction from Low Resolution Scene Images using Texture features," in Advanced Computing Conference (IACC), 2010.
[8]. Z. Zhang and Yuanli, "Research on the processing method of Automatic Reading Water Meter System.," in Conference on Artificial Intelligence and Computational Intelligence , China, 2009.
[9]. P. Anishiya, "Number Plate Recognition for Indian Cars using Morphological Dilation and Erosion with aid of OCRs," in Conference of Information and Network Technonolgy, Singapore, 2011.
[10]. R. Schultheis and M. Simmer, "Process modelling for Information Systems," Research and Development, pp. 153-165, 1971.