COMPARISON BETWEEN A LOW COST ARDUINO TRAFFIC LIGHT AND ONE DONE WITH NI USB 6008 DAQ DEVICE

Mihai Bogdan
Computer Science and Electrical Engineering Department,
Lucian Blaga University of Sibiu, 550025, Romania, E-Mail: mihai.bogdan@ulbsibiu.ro

ABSTRACT – The purpose of this paper is to design and implement two low cost project intended in terms of hardware and software, to make a street traffic light. The first project will be realized using the Arduino Uno development platform, and the second using the NI USB-6008 data acquisition board. The program used for both projects is LabVIEW. As a result, the hardware resources that will be used in the projects are: Arduino Uno, NI USB 6008, a red, yellow and green LED, a breadboard, 3 x suitable resistors for the LEDs you have (probably 220 Ohms is fine) and connecting wires.
The results will be displayed through the serial interface on the computer in the LabVIEW program, and also on the three LEDs on the breadboard.

1. INTRODUCTION

Traffic problems now-a-days are increasing because of the growing number of vehicles and the limited resources provided by current infrastructures. Traffic light is an optical signaling device that indicates different signals related to traffic, rail, naval and pedestrian. The traffic light aims, a traffic safety [3].

The data acquisition hardware used in this paper is an Arduino Uno board and an NI USB-6008 multifunction I/O device, which interfaces to the PC through a USB connector [2].

For implement this project we will need the following materials: Arduino UNO, NI USB-6008, breadboard, 3 LEDs (Red, Yellow and Green), 3x220 ohm resistors and jumpers wires to connect.

Arduino find application in almost any project where automation is needed. Various models of Arduino are available. Each of them is "imprisoned" for various tasks. Some boards are fundamentally different from the one shown in the figure below. But most of them have the following identical components.

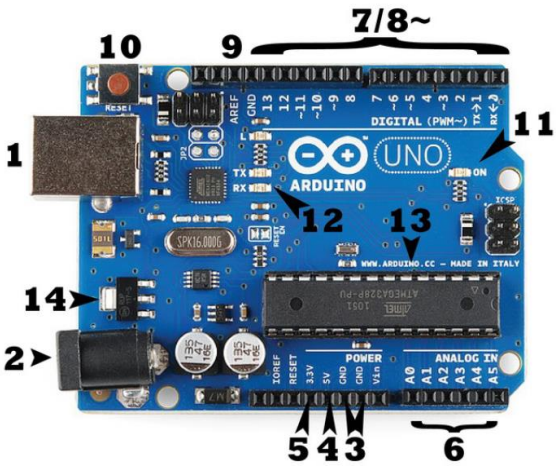

Figure 1. The Arduino Uno board

Arduino Uno can be powered from a USB cable from your personal computer or from a separate adapter that connects to the connector on the board. In the figure 1, the connection via USB is marked (**1**), and the connector for the external power source is (**2**).
The pins on your Arduino board are the provided connectors to which you will connect wires from peripheral devices. These pins are the following:

-GND pins (**3**): short for 'Ground'. There are several GND pins on the boards, each of which can be used to ground your electrical circuit.

-5V (**4**) and 3.3V (**5**) pins: provide power of 5 volts and 3.3 volts respectively. Many components connected to Arduino need to be powered of 5 volts and others of 3.3 volts.

-Analog pins (**6**): Analog inputs are located in the area called 'Analog In' (from A0 to A5 to Arduino Uno). These pins allow you to read signals from analog sensors (for example, a temperature sensor, light sensor) and convert them into digital values, which we further operate.

-Digital pins (**7**): opposite the analog pins there are digital pins (from 0 to 13 on Arduino Uno). These pins are used for digital input signals (for example, pressing a button) and for generating digital output signals (for example, powering the LED).

-PWM pins (**8**): you probably noticed the sign (~) next to some digital pins (3, 5, 6, 9, 10, and 11 on UNO). These pins work in both the conventional digital mode and PWM mode. If to explain briefly - these pins can simulate the analog output signal (for example, for gradual powering of the LED).

-AREF (analog reference) pin (**9**): It is the reference voltage against which all other analog voltages (analog inputs) are measured against.

Reset Button

As with the original Nintendo, Arduino has a reset button (**10**). When you click on it, the reset contact closes to ground and the code loaded on the Arduino starts working again.

Power LED

A little bit on the right and below the inscription "UNO" is an LED signed "on" (**11**). This LED should light up when you connected the Arduino to a power source. If the LED does not light up - a bad sign;).

TX and RX LEDs

TX-short for transmit, RX for receive. These conventions are often found in electronics to denote contacts that are responsible for serial data exchange. On Arduino Uno, these contacts occur twice on digital pins 0 and 1 and as TX and RX LEDs (**12**). These LEDs allow you to visually track, transmit or receive Arduino data (for example, when the program is loaded onto the board).

Main integrated circuit (IC)

A black part with metal connectors on both sides is an integrated microcircuit, a microprocessor (IC or Integrated Circuit) (13). You can safely assume that this is the "brains" of our Arduino.

Voltage regulator

The voltage regulator (**14**) performing the function indicated in the name - it controls the voltage that is supplied to the Arduino board. You can imagine it as a security guard, which does not let too much voltage on the board to avoid damage to it. Of course, the regulator has its own limit. So Arduino can not be powered by a voltage more than 20 volts.

Figure 2.The NI USB-6008 multifunction I/O device

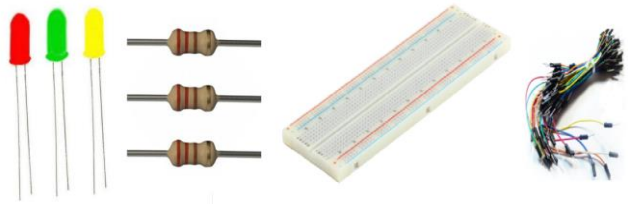

Figure 3. The components necessary for the elaboration of the technical projects

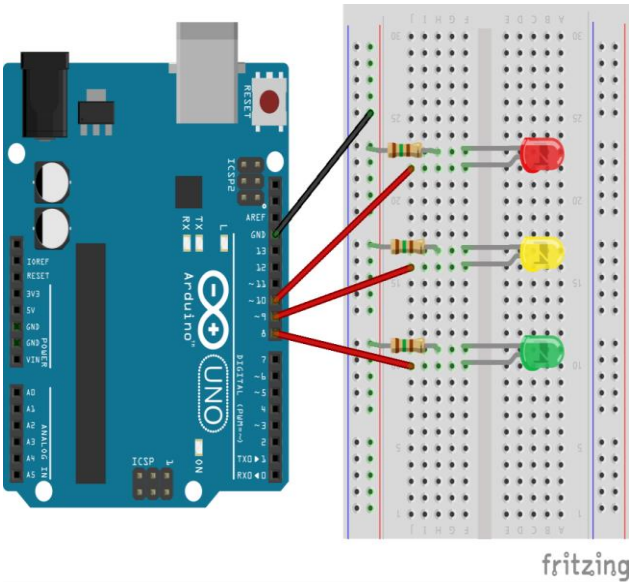

Figure 4. The hardware of the traffic light with Arduino

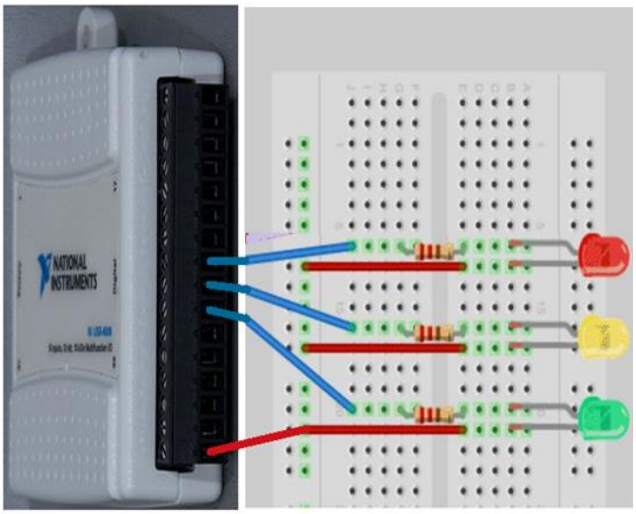

Figure 4. The hardware of the traffic light with NI USB 6008

2. THE VIRTUAL INSTRUMENT FOR THE ARDUINO TRAFFIC LIGHT

Programs developed in LabVIEW are called virtual instruments or Vis, and have the extension .vi. These programs have the role of receiving data from the user or from the computer interfaces with the process, processing them and then displaying, storing or transmitting them remotely.

A VI contains the following three components:

- Front Panel—Serves as the user interface.

- Block Diagram—Contains the graphical source code that defines the functionality of the VI.

- Icon and Connector Pane—Identifies the interface to the VI so that you can use the VI in another VI. A VI within another VI is called a subVI. A subVI corresponds to a subroutine in text-based programming languages [8].

Represented in Figure 5 is the Front Panel of the Arduino traffic light. It contains the following controls and indicators:
- A control for setting the serial port to which Arduino is connected;
- A boolean Stop control to turn off the virtual instrument;
- 3xLEDs indicators, belonging to the traffic lights.

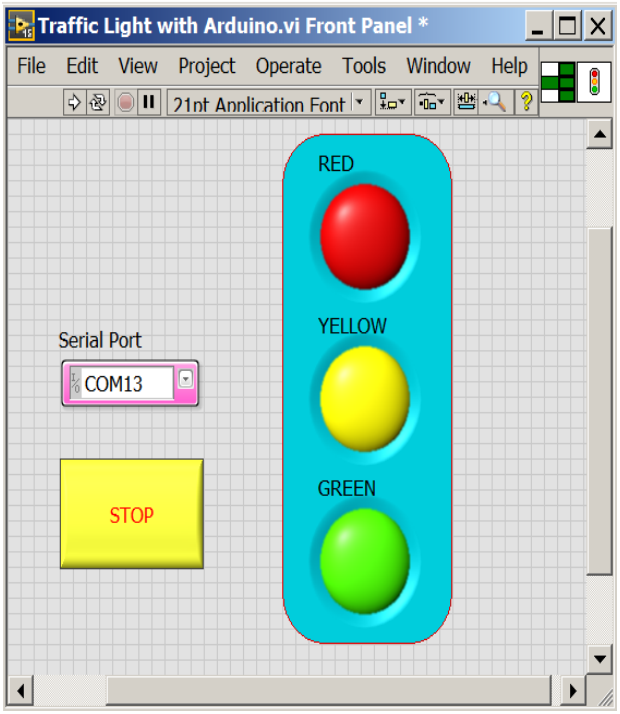

Figure 5. The Front Panel of the Arduino traffic light

To make the Block Diagram of the VI (Figure 6), I used the LINX software, that is the the new way of programming Arduino. LINX provides easy to use LabVIEW VIs for interacting with common embedded platforms, like Arduino. The LINX functions operate in a similar manner like the LIFA function, but they have different names.

The Block Diagram of VI contains the controls and indicators terminals of the Front Panel, the various nodes, constants and the wires. The nodes in LabVIEW are different functions, subVIs and programming structures.

LINX is an open source project by VI Package Manager and is designed to make it easy to develop embedded applications using LabVIEW. LINX includes VIs for over 30 of the most common embedded sensors as well as hardware agnostic APIs for accessing peripherals like digital I/O, analog I/O, PWM, I2C, SPI, and UART [9].

The Bock Diagram of the VI, also contains a While loop and the three sub-diagrams of the Case structure. A While Loop, executes the code it contains until a condition occurs. A Case structure has two or more sub-diagrams, or cases. The Case selector is Enum type, with three items: RED, YELLOW and GREEN. At the border of the While loop I created a shift register.

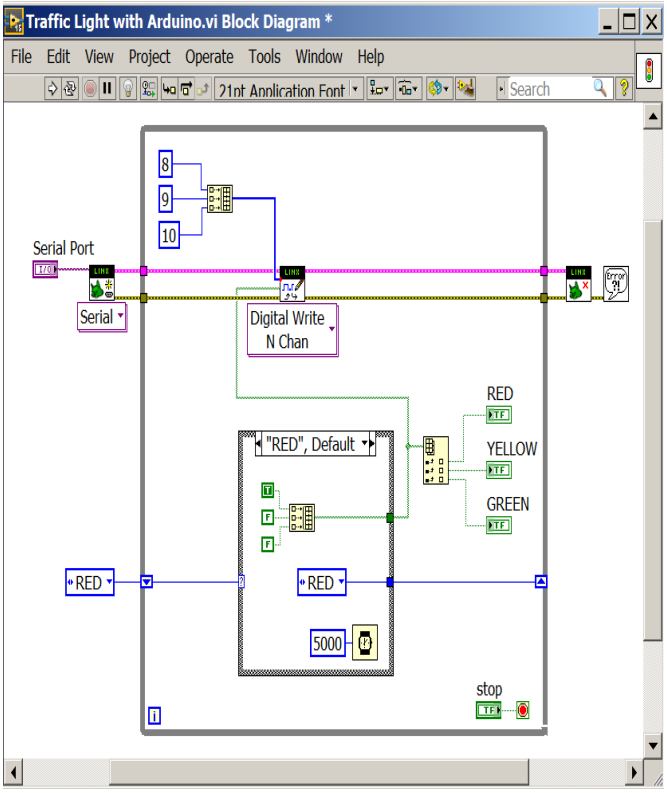

Figure 6. The Block Diagram of the Arduino traffic light

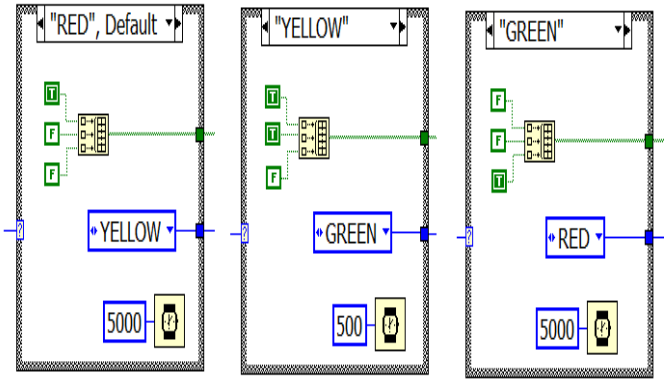

Figure 7. The three sub-diagrams of the Case structure

Use shift registers when you want to pass values from previous iterations through the loop to the next iteration. A shift register appears as a pair of terminals, shown as follows, directly opposite each other on the vertical sides of the loop border [10].

3. THE VIRTUAL INSTRUMENT FOR THE NI USB 6008 TRAFFIC LIGHT

Represented in Figure 8 is the Front Panel of the measuring system. It contains the following indicators and controls:

- 3xLEDs indicators, belonging to the traffic lights;
- a boolean Stop control to turn off the virtual instrument.

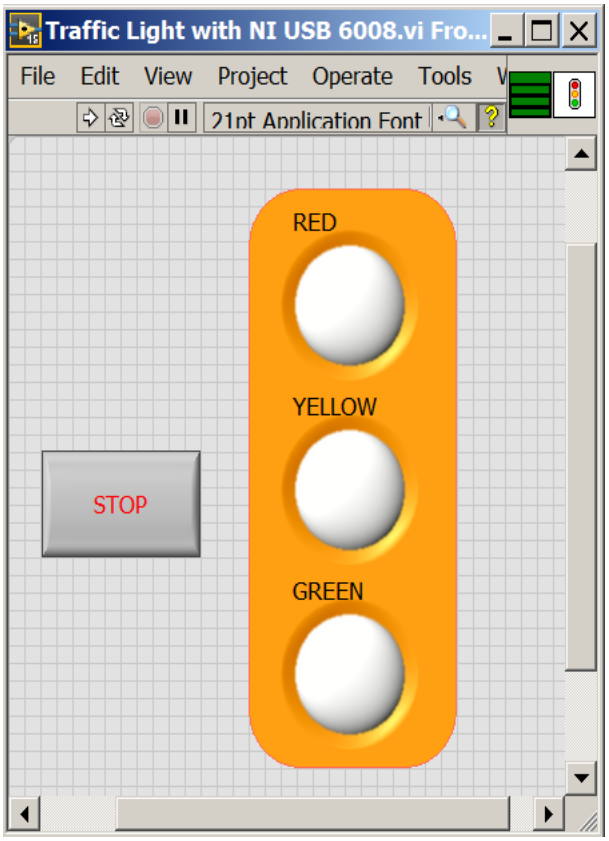

Figure 8. The Front Panel of the NI USB 6008 traffic light

After making the front panel of the VI, must implement the functionality of the program; the block diagram, which represents the source code of the instrument, is built.

The Block Diagram (Figure 9) consists of a While loop and a Case structure. At the borders of the While loop I created a shift register, which is initialized with an Enum constant with three Items (RED, YELLOW, and GREEN).

The Case structure has three sub-diagrams (RED, YELLOW, and GREEN). In each sub-diagram, we created three Local Variables for each of the three LED terminals in the Front Panel. At the three variables I wired the three Boolean constants. Using the Build Array function, the signals from the output of Boolean constants will also be transmitted to the digital output (DO) of the NI USB-6008, via the DAQ Assistant.

The data acquisition driver developed by National Instruments, NI DAQmx, is the software for easy hardware communication. NI DAQmx forms an intermediate level between the actual application and the hardware, allowing for avoiding low-level programming at the registers level. An example of using the driver is the DAQ Assistant application that uses NI Express technology. DAQ Assistant is an Express VI that provides an easy interface for configuring, testing and programming data acquisition.

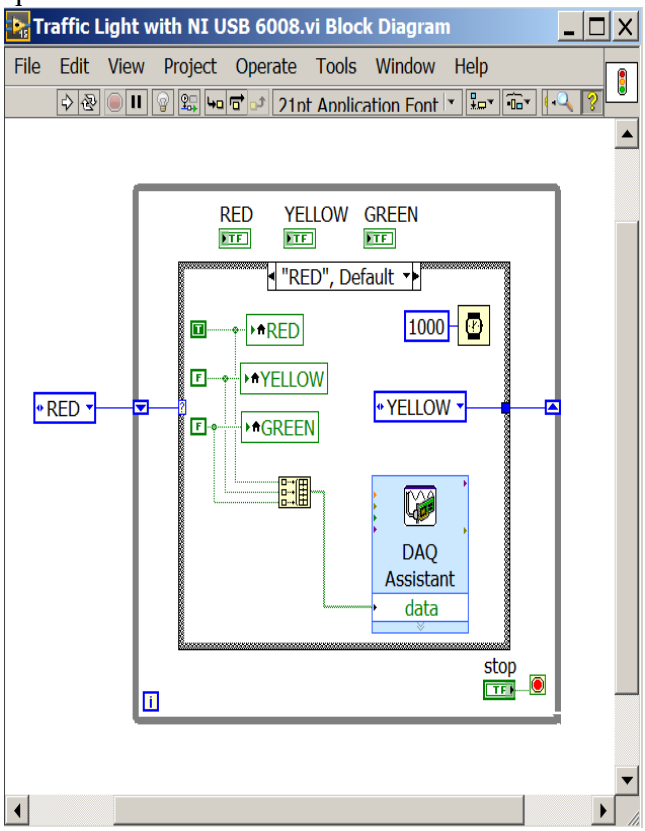

Figure 9. The Block Diagram of the NI USB 6008 traffic light

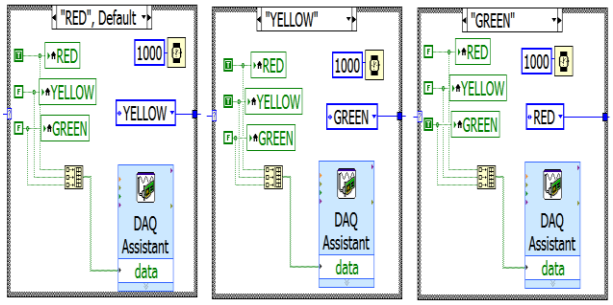

Figure 10. The three sub-diagrams of the Case structure

4. CONCLUSIONS

Research methods use mathematical models of systems, including sensors and actuators, which they simulate under different conditions and constraints. Different numerical methods are used, with the objective of validating the performance of the model and optimizing it [11].

Since research on physical models presents some important disadvantages, such as the long duration of research with high consumption of intellectual work

and the impossibility of encompassing economic factors, the current trend in technological process management is the widespread use of simulated models [12] [13].

The designing and implementing a LabVIEW traffic light was very useful and successful. The Arduino platform and the DAQ device along with LabVIEW are used to create the virtual instrument for designing a real time low cost traffic light.

NI USB 6008 DAQ devices are supported by NI-DAQmx driver, which is supported by Windows operating system.

Arduino platform are traditionally programmed with text-based code. But now, we can easily interface with them via LabVIEW allowing you to access many of the device features from within LabVIEW. For the interface of LabVIEW with Arduino, we used it LINX software. LINX provides easy to use VIs for interacting with common embedded platforms like Arduino.

The main advantages of virtual instrumentation:
- does not require physical storage space;
- can be with distributed items (can measure in multiple places at once);
- the data can be transmitted over the internet (the measuring lab can be located in a certain place and the analysis of the results can be done in a completely different way);
- measurements can be made in dangerous places for humans, and it is not necessary to have it in the immediate vicinity of the measuring system;
- tool configuration flexibility (virtual tools can be easily transformed by programming);
- significantly reduced costs (a single multifunctional acquisition card with the associated software can replace a lot of other dedicated physical tools)

5. REFERENCES

1. Bogdan M., *Traffic Light Using Arduino Uno and LabVIEW*, Proceedings of the 12th International Conference on Virtual Learning ICVL 2017, ISSN: 1844-8933 - ISI Proceedings, p. 286-290, October 28, 2017
2. Bogdan, M., *Using the NI USB-6008 DAQ Device, to Make a Traffic Light*, Proceedings of the 12th International Conference on Virtual Learning ICVL 2017, ISSN:1844-8933 - ISI Proceedings, p. 281-285, October 28, 2017
3. Dinesh K., N., Bharagava Sai, G., *Trafic Control System Using LabVIEW*, Global Journal of Advanced Engineering Technologies, ISSN: 2277-6370, Vol2-Issue2-2013
4. Bogdan, M., *Virtual Signal Generator Using the NI-USB 6008 Data Acquisition*, Nonconventional Technologies Review, Volume XVII, Nr. 2/2013, ISSN 1454-3087, p.5-8, June, 2013
5. Bogdan, M., *Virtual Instrument for the Study of the Signals Sampling*, Proceedings of the 10th International Conference on Virtual Learning, ISSN: 1844-8933-ISI Proceedings, p.354-358, October 31, 2015
6. Bogdan, M., *Measurement experiment, using NI USB-6008 data acquisition*, Journal of Electrical and Electronics Engineering, Vol.2, Nr.1, ISSN 1844-6035, 2009.
7. https://www.arduino.cc
8. http://www.ni.com/getting-started/labview-basics
9. http://zone.ni.com/reference/en-X/help/371361H-01/lvconcepts/intro_to_vis/
10. http://www.ni.com/getting-started/labview-basics/shift-registers
11. Țîțu, M., Oprean C. Management of intangible assets in the context of knowledge based economy, Editura LAP Lambert, ISBN-13 978-3-659-79332-5, ISBN-10 3659793329, 280 pages, Germany
12. Țîțu, M., Oprean, C., Boroiu, Al. Cercetarea experimentală aplicată în creșterea calității produselor și serviciilor, Colecţia Prelucrarea Datelor Experimentale, Editura AGIR, ISBN 978-973-720-362-5, 684 pagini, București, 2011
13. Ţîţu, M. Fiabilitate şi mentenanţă, Editura AGIR, ISBN 978-973-720-169-0, 370 pagini, București, 2008
14. Bogdan, M., Analiza şi Prelucrarea Numerică a Semnalelor din Sistemele Electroenergetice, ISBN 973-9358-38-1, Editura Mediamira, Cluj-Napoca, 2000