Imperial Journal of Interdisciplinary Research (IJIR)
Vol-2, Issue-5, 2016
ISSN: 2454-1362, http://www.onlinejournal.in

Real Time DC Motor Speed Control Using PID in LabVIEW with Arduino

Vishnu V S[1], Aneesa K A[2], Arun Lal[3], Absal Nabi[4]
[1, 2, 3] B-Tech Student, [4]Assistant Professor
[1,2,3,4] Department of Electrical & Electronics Engineering, ICET, Muvattupuzha.

Abstract*: DC Motor plays a crucial role in research and laboratory experiments because of their simplicity and low cost. The proportional integral derivative (PID) controller is the most common method used in closed loop control system. It can be used for various industrial applications. The main objective of our paper is to show how a dc motor can be controlled using labVIEW. The Arduino Uno board acts as the low coast data acquisition device. The Arduino also feedbacks the speed of the DC motor the labVIEW. The speed sensed by means of an IR sensor. The feed backed speed is then compared with the set speed in the PID pallet of the labVIEW. Then control signal for the error's are produced*

Keywords: *Proportional integral derivative (PID), Direct current (DC), Pulse width modulation (PWM), Laboratory virtual instrumentation engineering workbench (labVIEW), labVIEW interface for Arduino (LIFA).*

1. Introduction

The aim of our paper is to show how DC motor can be controlled by using a PID controller in LabVIEW. DC Motor will be interfaced with Lab VIEW using an Arduino Uno board. Arduino Uno board plays the role of data acquisition. The speed of the DC motor will be set by creating a VI for PID Controller in Lab VIEW. LabVIEW will in turn pass this speed to the DC motor using a PWM pins on the Arduino Uno board. DC motor will move with the speed set by the user in LabVIEW. The speed of the dc motor will be sensed by using the IR sensor. The output is sent back to the PID Controller in Lab VIEW via Arduino board. PID Controller compares the actual speed of the DC motor with the set speed. If its speed is not same, PID Controller will try to minimize the error and bring the motor to the set point value.

2. PID controller

A proportional–integral–derivative controller is a control loop mechanism widely used in industrial systems. A PID is the most commonly used feedback controller. A PID controller calculates an "error" value as the difference between a measured process variable and a desired set point [1]. The controller tries to minimize the error by adjusting the process control inputs. A PID controller is the best controller for any closed loop application. However, for best performance, the PID parameters used in the calculation must be tuned according to the nature of the system

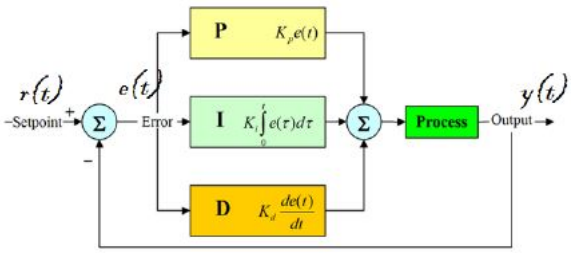

Figure 1. General block diagram of PID system

When the p controller is acted along, there may be a chance of steady state error. The difference between the stabilized output and the reference is called the Steady State Error. It responds immediately to the current tracking error but it cannot achieve the desired set-point accuracy without an unacceptably large gain. Integral control causes an output signal to change as a function of the integral of the error signal over time duration. Integral term yields zero steady-state error in tracking a constant set-point. It also rejects constant disturbances. Derivative action reduces transient errors and causes an output signal to change as a function of the rate of change of the error signal. The contributions of the three terms will yield the control output, or the control variable. So here we use PID controller for better and accurate response [3].

Control Variable = Pout + Iout + Dout

3. Arduino Uno board

An Arduino board consists of an 8-bit Atmel AVR microcontroller with components to facilitate programming and incorporation into other circuits.

An important aspect of the Arduino is the standard way that connectors are exposed, allowing the CPU board to be connected to a variety of interchangeable modules known as shields [5]. Some shields communicate with the Arduino board directly over various pins. Official Arduino have used the mega AVR series of chips, specifically the ATmega8, ATmega168, ATmega328, ATmega1280, and ATmega2560. Most boards include a 5 volt linear regulator and a 16 MHz crystal oscillator, although some designs such as the Lily Pad run at 8 MHz and dispense with the onboard voltage regulator due to specific form- factor restrictions. An Arduino microcontroller is also pre-programmed with a boot loader that simplifies uploading of programs to the on-chip flash memory, compared with other devices that typically need an external programmer. The main advantage of Arduino boards compared with the other microcontroller is that the user can program and upload it directly to the controller through Arduino IDE's. The language for the programming is more user friendly.

4. Using PWM' On Arduino

PWM is a method for supplying the electrical power to a load having slow response. The supply is a train of pulses. The width of the each pulses control the voltage level to the load. The Arduino already contains 6 PWM pins on it. The PWM output on the Arduino can be varied by varying the duty cycle from "0" to "255". Thus the speed can be varied. The PWM pin will have either V_s volt during the pulse or zero volts otherwise. The voltage signal comprises of pulses of duration T_0 that repeats every T_c units of time. If this signal is supplied as input to a device that has a response time much larger the T_c, the device will experience the signal as an DC input with an effective voltage

$$V_{eff} = V_s (T_0/T_c)$$

The ratio $T_0 = T_c$ is called the duty cycle of the DC pulses. The DC voltage applied to the load is controlled by varying this duty cycle.

5. LabVIEW interface for Arduino (LIFA)

LabVIEW interface for Arduino is a way for connecting the Arduino with the labVIEW. It is a sketch for the Arduino for the serial communication with the VI's of the labVIEW. This helps to move data's from Arduino to labVIEW without adjusting the communication. By using open, read, write, close conventions in labVIEW we can access the digital, analog and PWM signals of Arduino[2]. For this the Arduino should be connected to the system either through USB, serial or Bluetooth.

6. Results and discussion

7.1 Open loop system:

Block diagram:

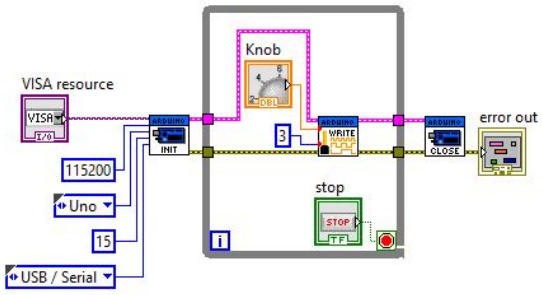

Figure 2.block diagram of open loop motor speed control

The VI is made inside a while loop for providing a stopping facility for the VI. For making a communication to the VI from the Arduino we use the INT and CLOSE pallets. They are initialized with there values. The main part of the VI lies inside the while loop. That is a PWM write pallet and knob. The PWM write pallet is initialized with a constant 3 to select the 3rd PWM pin on the Arduino. The knob is used for varying the duty cycle of PWM and there by controlling the speed of the motor[4].

Front panel:

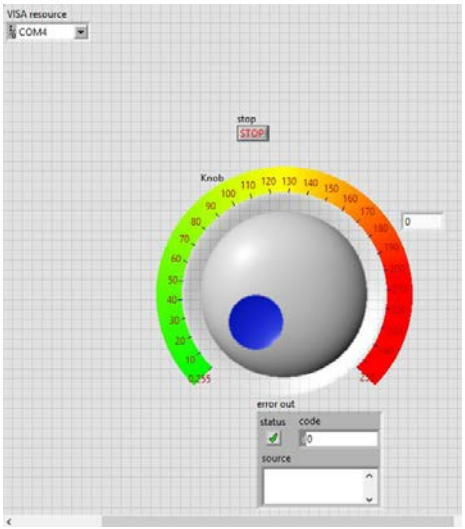

Figure 3.front panel of open loop motor speed control

The front panel for the open loop DC motor speed control is as shown above. It consist mainly a knob. The knob has a dial ranges from 0 to 255. The knob is varied from 0 to 255 for controlling the speed of

the motor. It actually changes the duty cycle of the PWM output of the Arduino. There by the speed is changed. It also consists of a stop button and a com box. The stop button is used to stop the running VI. The com box is used for selecting the common port for the communication with the Arduino board.

7.2 Closed loop system:

Block diagram:

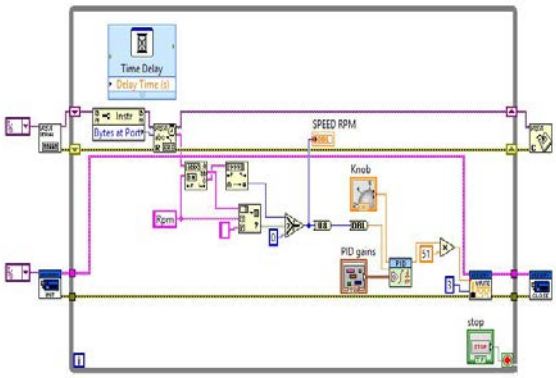

Figure 4.block diagram of closed loop motor speed control

The connecting panel for the closed loop DC motor speed control is as shown above. The VI is made inside a while loop for providing a stopping facility for the VI. For making a communication to the VI from the Arduino we use the INT and CLOSE pallets. They are initialized with there values. After that we set the required Arduino pins for output or input. It also consists of a knob for setting the required speed. The dial ranges from 0 to 255.The main part of the closed loop speed control is the PID VI. It consists of many inputs and outputs in it. We set the required speed at the set point port. There is a process variable port where we give the feed back signal that is the speed of the motor at that time. There are also ports for setting the PID values for the motor. During the operation, the PID will compare the set point with the process variable and will produce a signal for eliminating the error in output if any. The output of the PID given to the PWM write pallet. Where the PWM pin 3 is enabled and there by the speed of the motor is controlled. The speed of the motor sensed through Arduino by serial communication. Because through LIFA the signal sampling is high,. So Arduino has difficult to read the signals. That's why we used the serial communication.

Front panel:

The front panel for the open loop DC motor speed control is as shown above. It is also similar to the open loop speed control front panel, except that an additional element of PID is included. It is used for setting the required PID values. The front panel also consists of a knob. The knob has a dial ranges from 0 to 255. The knob is varied from 0 to 255 for controlling the speed of the motor [1]. There by the speed is changed. It also consists of a stop button. The stop button is used to stop the running VI.

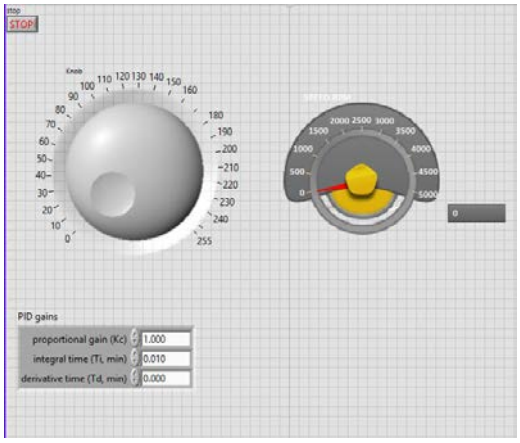

Figure 5.front panel of closed loop motor speed control

7. Conclusion

The method adopted in our project is low cost technique for controlling the speed of a DC motor. Arduino Uno board plays the role of Data Acquisition System. The DC motor is interfaced with PID Controller in LabVIEW via Arduino Uno board. Speed of the motor is sensed by the tachometer which is sent back to PID Controller as feedback for calculating and compensating the error produced if any. The method implemented can be used for various industrial applications. This technique helps in maintaining the stability of the system

8. References

1. M. Saranya, D. Pamela, "A Real Time IMC Tuned PID Controller for DC Motor", IJRTE ISSN: 2277-3878, Volume 1, Issue-1, April 2012
2. Naveen Kumar and Dr. Prasad Krishna, "Low cost data acquition and control using Arduino Prototyping platform and LabVIEW",IJSR,Volume 2, Issue-2, February 2013
3. Purna Chandra Rao et. al., "Robust Internal Model Control Strategy based PID Controller for BLDCM" ,International Journal of EngineeringScience and Technology, Volume 2(11), 2010, 6801-6811
4. Olden. P, "Open Loop motor speed control with LabVIEW", SoutheastCon, Proceedings IEEE, pp. 259-262, 2001, (Conference Publications)
5. Website: https://www.arduino.cc/ A complete guide to Beginners.