MANTECH Publications

Journal of Current Trends in Electrical Engineering
Volume 7, Issue 3, September-December, 2022

MATLAB-Based Speed Control of a Permanent Magnet DC Motor Using an Arduino Interface

Pankaj Bansal[1], Girish Sharma[2]

Associate Professor[1], Student[2]

Department of Electrical Engineering

Datta Meghe College of Engineering

Corresponding Author's Email: -*grishsharma225@gmail.com[2]*

Abstract

This article describes a closed loop speed control system for a permanent magnet direct current (PMDC) motor. In MATLAB Simulink, a Proportional plus Integral (PI) controller is created. The computer's control signal is sent into an Arduino microcontroller board, which provides the appropriate pulse width modulated (PWM) signal, which drives the PMDC motor through a driver IC. The speed of the motor shaft is measured by a tacho- generator and transmitted back to the controller block in MATLAB. The controller has been developed in such a manner that the motor speed may be kept constant even when the load changes abruptly. The results of hardware experiments are presented to illustrate the performance of the designed approach.

Keywords: *PI controller, L293D chip, PMDC motor, Arduino UNO Microcontroller*

INTRODUCTION

PMDC motors are utilised in applications that demand high beginning torque at low speeds. In comparison to conventional DC motors, PMDC motors have a few advantages, including no need for a field excitation arrangement, a smaller motor size due to the absence of a field coil, improved efficiency due to the absence of field excitation, and being less expensive and economical for fractional kW rated applications. Because the magnetic flux in the air gap is fixed and cannot be adjusted externally, armature control is the only way to vary the speed of a PMDC motor [1, 2].

Several speed control systems for PMDC motors have been presented by various academics throughout the years. Different controllers, such as fuzzy logic controllers [3, 6], PI controllers to achieve the controlling signal with different types of microcontrollers [3-7], can regulate the speed of PMDC motors or other types of DC motors. If a fuzzy controller is used to regulate the speed of a DC motor, substantially more human adjustment by trial and error will be necessary to achieve the best performance level since the fuzzy controller utilises an asymmetric membership function. Manual adjustment, on the other hand, may be avoided by employing a PI controller. Although the PI controller produces greater performance [7], it is not cost effective if hardware components are utilised to send the controller signal to the DC motor. To make the PI controller economically possible, the present work employs an Arduino microcontroller to create the needed PWM signal, allowing the same level of performance to be obtained at a reduced hardware cost and complexity.

The traditional armature speed control approach is employed in this work, but with a unique manner. The need for a hardware control circuit is avoided since the control circuit is created in MATLAB and operates the PMDC motor with Arduino, a programmable microcontroller. It is mostly used in robotics to regulate the forward, reverse, or other types of motions of a DC motor [8]. The speed control circuit in MATLAB must be developed with the right feedback mechanism in order to maintain the motor speed constant under varying mechanical load. Designing the control circuit in MATLAB avoids power loss in the control circuit and allows you to create a wide range of various control circuit configurations without making further expenditures in different control circuit hardware models.

This article describes the hardware implementation of a PMDC drive system that seeks to maintain a constant speed in response to fluctuating loads. MATLAB Simulink is used to create a closed loop (Proportional-Integral) PI control circuit, and control signals from the PC are utilised to operate a PMDC motor using Arduino[9] interface with L293D (motor driver chip) [10].

MODELING OF PMDC MOTOR FOR SPEED CONTROL

Understanding the mathematical model of the motor and the proper regulating mechanism are required to work with the speed control scheme of a PMDC motor.

PMDC Motor Model

This section describes the method used to identify the final transfer function of a PMDC motor from its equivalent circuit parameters [11], as shown in Figure. 1.

From the equivalent circuit as shown in Figure. 1, the armature supply voltage (Va) can be related to the back emf (Eg) as:

$$V_a = E_g + I_a R_a + L_a \frac{dI_a}{dt} \quad (1)$$

Where, Ia is armature current (A), Ra and La are armature resistance (Ω) and inductance (H) respectively.

Torque equation of the PMDC motor is given by

$$T_d = J\frac{d\omega}{dt} + B\omega + T_L \quad (2)$$

Here, Td is torque developed (Nm), TL is load torque (Nm), J is moment of inertia (Kg/m2), B is friction coefficient and ω is angular velocity (rad/s) of the motor.

Assuming the value of rotor friction of the motor to be negligible, consider B=0. Therefore, the torque equation (2) can be simplified as:

$$T_d = J\frac{d\omega}{dt} + T_L \quad (3)$$

As the motor is of PMDC type, therefore the flux φ is constant. Then the back emf of the motor can be expressed as

$$E_g = K\omega \quad (4)$$

And torque is

$$T_d = KI_a \quad (5)$$

Where, K is a constant.

Since the aim is to keep the speed of the motor constant independent of the applied mechanical load, therefore the output of the drive system isspeed (ω) and input is reference armature voltage (Va). Figure.2 shows the block diagram of the PMDC motor, using (1) – (5).

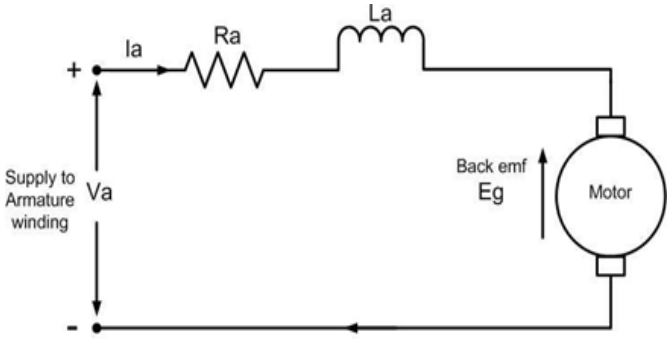

Fig. 1. PMDC motor equivalent circuit

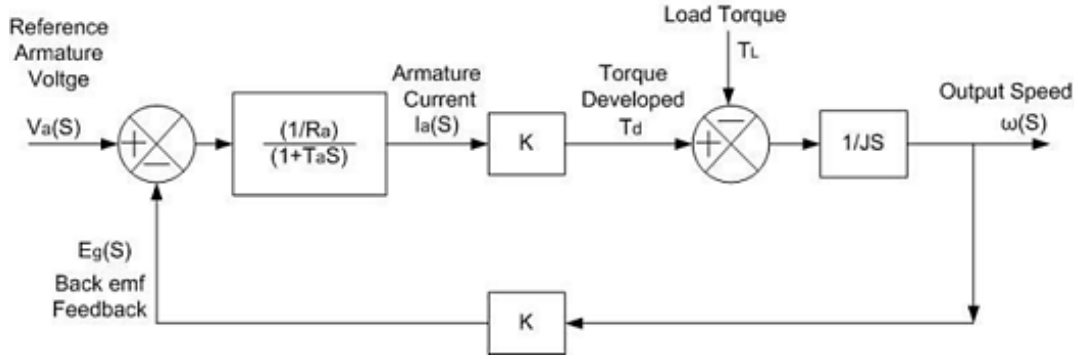

Fig. 2. PMDC motor block diagram [12]

Now combining (1) – (5), transfer function of the PMDC motor can be derived as:

$$\frac{\omega(S)}{Va(S)} = \frac{(\frac{K}{Ra})/\{JS(1+TaS)\}}{1+(\frac{K^2}{Ra})/JS(1+TaS)} \quad (6)$$

Where, armature time constant, $T_a = \frac{L_a}{R_a}$

Electromechanical time constant of the motor is given by:

$$T_m \quad (7)$$

At the time of starting of the motor, assuming load torque TL = 0 Nm and also considering negligible armature inductance La, the derived transfer function is

$$\frac{\omega(S)}{Va(S)} = \frac{(1/K)}{(1+STm+S^2TaTm)} \quad (8)$$

As Ta<< Tm [7], therefore $(1 + ST_m + S^2T_aT_m)$ Can be written as .

Hence,

$$\frac{\omega(S)}{Va(S)} = \frac{(1/K)}{(1+STm)(1+STa)} \quad (9)$$

The final transfer function given by (9) will be used to model the PMDC motor in MATLAB.

Controller Circuit Modeling

Because the motor's speed must be kept constant at the reference value even under fluctuating mechanical load, the controller circuit must be properly designed. Control circuits that are commonly used include proportional (P), proportional-integral (PI), and proportional-integral-derivative (PID).

Among them, the PI controller is utilised because the P controller can respond quickly and the I component can reduce the steady-state speed error to zero [11, 13].

Figure 3 depicts a block schematic of a basic PI controller. As indicated in Figure 4, the controller is put before the PMDC motor model.

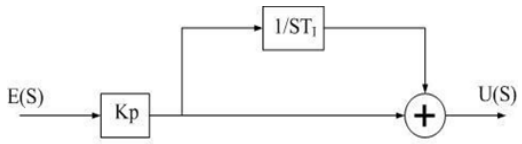

Fig. 3. PI controller

From Figure. 3, the relation between E(S) and U(S) is as

$$\frac{U(S)}{E(S)} = K_P + \frac{K_I}{S} \quad (10)$$

Where, KP and KI are proportional and integral gains respectively and also. Here, TIis integral time.

MATLAB Simulation of PMDC Motor with Pi Controller

Before implementation of the speed control circuit of motor in hardware, MATLAB simulation is done for cross verification of thesystem. To drive the motor from a PWM signal [14], a chopper module has been inserted between the PI controller block and the PMDC motor block as shown in Figure. 4.

Parameters of the PMDC motor used for simulations are given below

Voltage rating (Va): 12volt

Armature resistance (Ra): 0.5Ω

Armature inductance (La): 0.01H

Rated armature current (Ia): 10A

Rated speed

(Nr): 3000rpm = 314.1593rad/s

Back emf constant (K): 0.0954volt-s/rad

Moment of inertia (J): 0.4Kg-m2

Armature time constant

(Ta = La/Ra): 0.02s

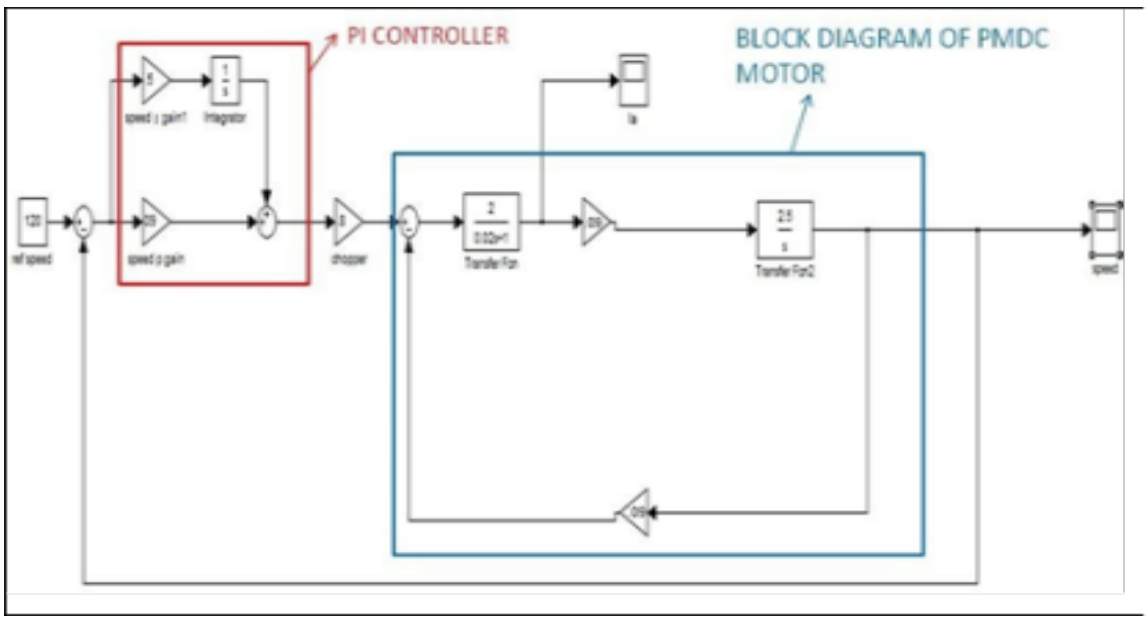

Fig. 4 PMDC motor block diagram with PI controller under no load condition

Simulated speed-time curve of the PMDC motor with PI controller is shown in Figure. 5.The reference speed of the motor is set to 120 rad/s in the MATLAB simulink model shown inFigure.4shows that as anticipated, speed of the motor is found to remain constant at the set value of 120 rad/s under no-load.

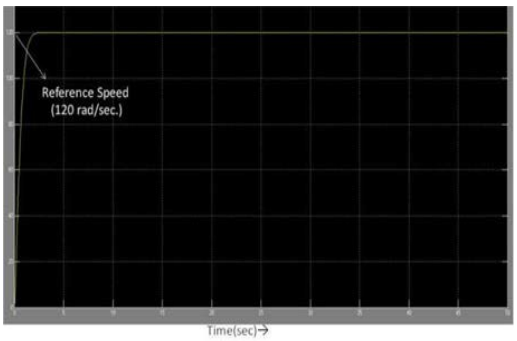

Fig. 5. Speed-time curve of PMDC motor with PI controller

Hardware Implementation

After verification of the simulation model, the challenge is to run a PMDC motor at a set value of speed under variable load conditions in hardware setup. The controller circuit remains in MATLAB Simulink. Interfacing between theHardware and software model is done by Arduino microcontroller with L293D, a motor driver chip.

PMDC motor

Two identical PMDC machines are used in the study as shown in Figure. 6.

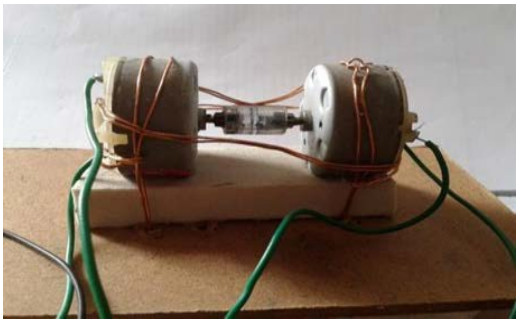

Fig. 6. PMDC motor and generator at coupledcondition

One is used as a motor for which the speed is to be maintained at a fixed value at different mechanical loads. The second one is used as a tacho generator to sense the speed of motor and produce a voltage signal at its output terminal that is proportional to the speed. The shafts of the motor and generator being directly coupled, tacho generator output voltage can directly be used as a feedback signal proportional to the motor speed.

Arduino Board

The Arduino microcontroller board as shown in Figure. 7 is used as a bridge between the PMDC motor and MATLAB Simulink model of the drive setup. The Arduino board is used to transmit the controller output signal to the motor via motor driver chip L293D.

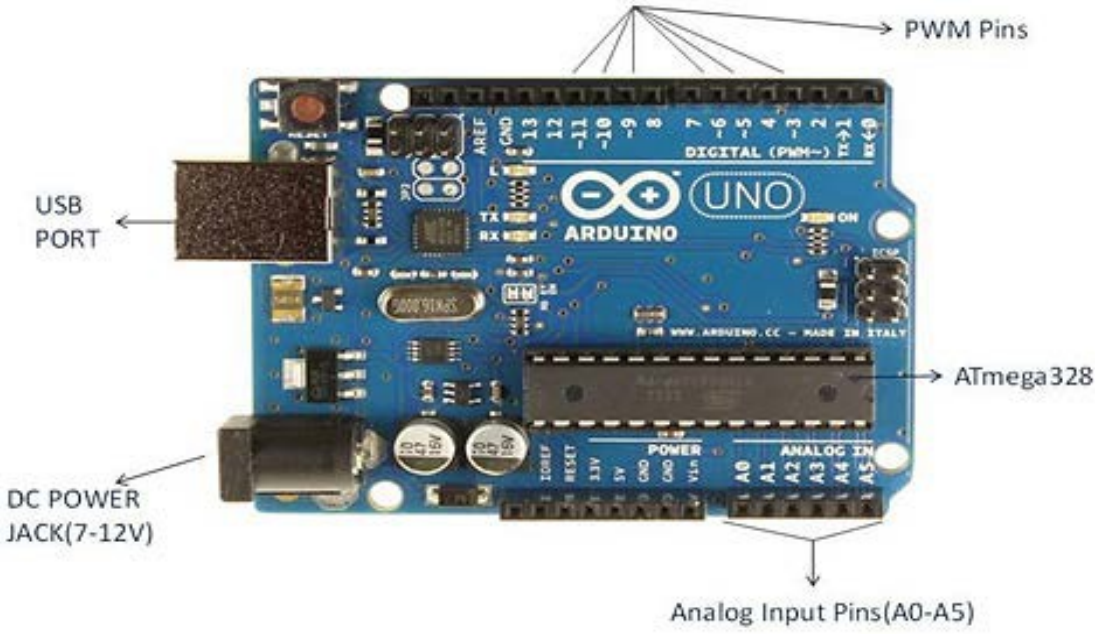

Fig. 7Arduino UNO board [15]

A detailed specification of the Arduino board is given in Table I.

Table I Detail Specifications of Arduino Uno [16]

Components & Parameters	**Types &Values**
Microcontroller	ATmega328
Operating	5V
Input Voltage	7-12V
Input Voltage	6-20V
Digital I/O Pins	14(of which 6 provide
Analog Input	6
DC Current per	40mA
DC Current for	50mA
Flash Memory	32KB(ATmega328) of
SRAM	2KB(ATmega328)
EEPROM	1KB(ATmega328)
Clock Speed	16MHz

L293D – A Motor Driver Chip [17,18]

The motor driver chip (L293D) is connected between Arduino board and the PMDC motor. As the Arduino output signal strength is less than the motor voltage and current ratings, therefore Arduino output cannot be used to directly drive the motor.

The driver chip amplify the controller signal received from Arduino to match the required voltage, current rating for the motor.The L293D driver chip can be used to drive two motors simultaneously, though it has been used to drive only one motor is the present work.

In Figure.8, the terminals of L293D necessary to drive a single motor are marked.

Hardware Connection of Motor-Generator with Arduino and L293D

Schematic diagram of the entire hardware set up with the Arduino board, motor driver chip, motor, and generator is shown in Figure.8.The PWM signal is transmitted from Arduino pin 11 to L293D pin 2. From the chip pin 3 the PWM signal goes to the motor. All the 5V terminals are shorted and 0V terminals are shorted as shown. To drive the motor through L293D chip, a 12V supply is given to the chip pin 8.From the output of the generator the voltage is dropped by using a pot, because the Arduino cannot deal with a voltage having high value. Then Pot transmits the generator output voltage to arduino analog pin 0. Via the USB port, the arduino connected to MATLAB Simulink model at a computer and an AC to DCadapter is connected at DC power jack of arduino.

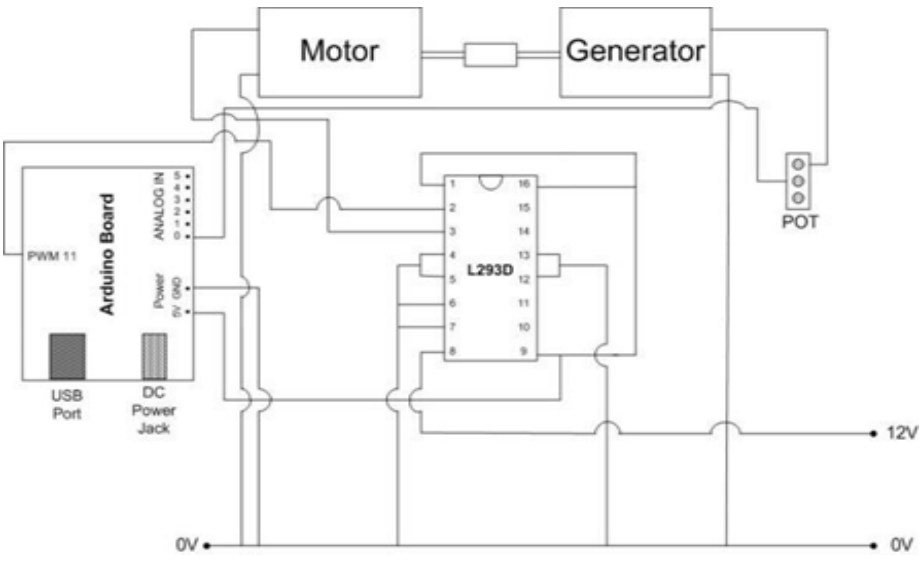

Fig. 8 Layout of Hardware Connection

Photograph of the actual set up is shown in Figure. 9.

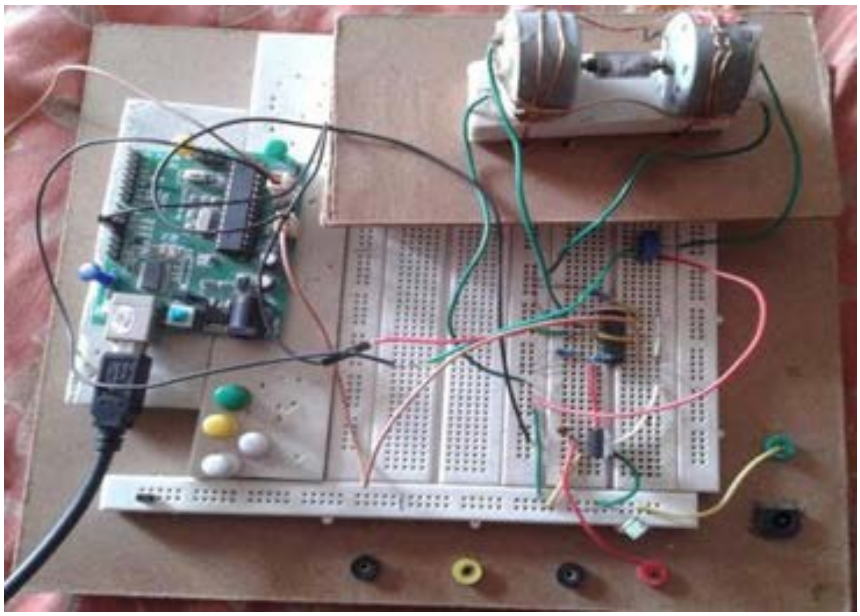

Fig. 9. Hardware connection

MATLAB Simulink Model of PI Controller and Feedback

Figure 10 depicts the MATLAB simulation of the driving system. The reference speed is retained at 150rad/s in this case. The speed signal is then sent to the PI controller block with the appropriate gain. The PI controller's output then enters the rounding function block, where the fractional data is transformed to integer data with round up, which the Arduino accepts. The next block is Arduino 1 Analog Write Pin 11, which converts the analogue signal to a PWM signal at Arduino pin 11. The signal is sent from Arduino pin 11 to L293D input pin 2. In the feedback channel, Arduino 1 Analog Read Pin 0 receives the output of the tacho generator, and the signal is sent to the input to be compared to the reference speed and, if necessary, to create an erroneous speed signal. A gain block is utilised after the analogue read block. The increase is equal to 0.249.

Arduino can read up to 1024 bytes and write up to 255 bytes. As a result, (255/1024) = 0.249. Now, this block translates the value read by the Arduino in the 0 - 255 range and sends it along the feedback channel to the comparator, which compares it to the reference value to create an error signal, which is then sent to the PI Controller [19]. The Arduino IO setup block displays the kind of Arduino board as well as the port number (COM 8) to which the board is attached.

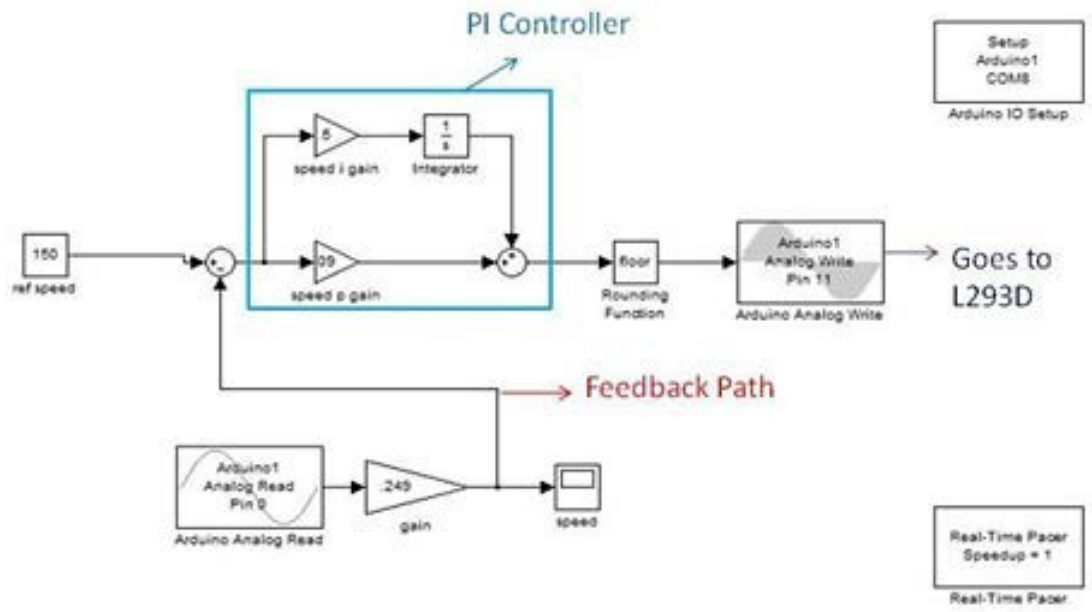

Figure:-10 Simulink model of controller

RESULT AND ANALYSIS

Shows the Speed – Time Curve of The Motor.

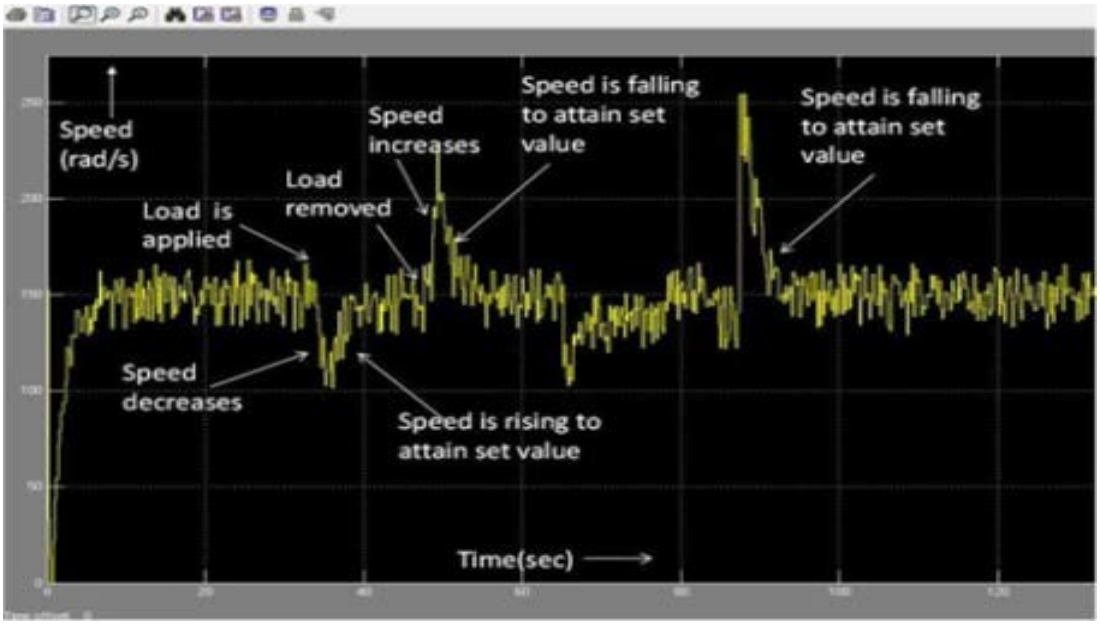

Figure:- 11 Speed-time curve of PMDC motor with PI controller under variable load with Arduino interface

As seen in the picture, at roughly 33s, mechanical force is immediately applied to the motor's shaft, causing an initial drop in speed, but the motor quickly recovers to its reference speed of 150rad/s. The external load is released at 47s, resulting in an

immediate rise in speed before returning to its reference speed. As demonstrated in Figure 11, this procedure of reacting to unexpected load changes is designed to happen multiple times during the run.

The speed-time curve in Figure 11 is not smooth like the simulated figure in Figure 5, which is entirely developed in MATLAB Simulink. This is because, in software, all parameters are perfect, however in hardware, certain unavoidable factors and disturbances are introduced into the system.

CONCLUSION AND FUTURE SCOPE

This paper describes a novel closed loop speed control system for PMDC motors under variable load situations. The technique is unique in that it creates a MATLAB Simulink model of the controller, avoiding otherwise complex hardware circuitry. An Arduino UNO microcontroller and an L293D motor driver IC connect the motor to the software-based PI controller.

The hardware testing findings reported in the article illustrate the controller's ability to keep the motor speed at a constant value regardless of the mechanical stresses applied.

Higher capacity motors may be handled by utilising higher versions of Arduino in conjunction with MOSFET or other FET-based motor drivers rather than the Arduino UNO board. Other types of motors can also be controlled utilising the main notions of the suggested approach.

REFERENCES

1. J. Larminie and J. Lowry, Electric Vehicle Technology Explained, 2nded.Hoboken, NJ, USA: Wiley, 2012.
2. C. Mi, A. Masrur, and D. Gao, Hybrid Electric Vehicles: Principles andApplications with Practical Perspectives. Hoboken, NJ, USA: Wiley,2011.
3. H. R. Jayetileke, W. R. de Mei, H. U. W. Ratnayake, “Real-time fuzzy logic speed tracking controller for a DC motor using Arduino Due”, ISSN: 2151-1802
4. Julio Noel Hernández-Pérez, Jesús Ebert Giral-Salas, Roberto Morales-Caporal, Rafael Ordoñez-Flores, Miguel Ángel Morales-Flores, “Speed and current control of a permanent-magnet DC servo motor using a real-time microcontroller”, ISBN:978-1-4799-1460-9

5. Y. S. E. Ali, S. B. M. Noor, S. M. Bashi, M. K. Hassan, “Microcontroller performance for DC motor speed control system”, ISBN: 0-7803-8208-0
6. AhmetIsik, Oktay Karakaya, P.AlperÖner, Mehmet KubilayEker, “PMDC motor speed control with fuzzy logic algorithm using PIC16F877 micro controller and plotting data on monitor”, ISBN: 978- 1-4244-3429-9
7. R. Sankar, S. Ramareddy, “Closed loop controlled PMDC motor drive” ISSN : 0537-9989
8. National Instruments- Robotics Fundamentals Series: Motors, Publish Date: Mar 16, 2016
9. https://www.arduino.cc
10. http://www.arduino.cc/documents/datasheets/L293D.pdf
11. Ned Mohan,Tore M. Undeland, William P. Robbins, Power Electronics: Converters, Applications, and Design, 3rded, India, Wiley, 2009, pp. 377 – 391.
12. Gopakumar, K., Power Electronics and Electrical Drives, Video Lectures 1-25, Centre for Electronics and Technology, Indian Institute of Science, Bangalore.
13. Choudhury, d. Roy, Modern Control Engineering, India, PHI, 2012, pp. 323 – 346..
14. Cengelci, S. U. Sulistijo, B. O. Woo, R. Teodrescu, and F.Blaabjerg, “A new medium-voltage PWM inverter topology for adjustable-speed drives,” IEEE Trans. Ind. Appl. , vol. 35, no. 3, pp. 628–637, May/Jun. 1999.
15. https://www.arduino.cc/en/Guide/Windows
16. https://www.arduino.cc/en/main/arduinoBoardUno
17. https://www.arduino.cc/documents/datasheets/H-bridge_motor_driver.PDF
18. http://www.ti.com/lit/ds/symlink/l293.pdf
19. https://www.arduino.cc/en/Tutorial/SecretsOfArduinoPWM