Implementation Of PID Controller Using Arduino Microcontroller

Sumit Mhatre[1], Rahul Nair[2], Omakar Thakare[3], A.M Gund[4]

[1, 2, 3, 4] Dept of Instrumentation

[1, 2, 3, 4] Bharati Vidyapeeth College Of Engineering

Abstract- *This paper is associated with the design and implementation of microcontroller based PID controller. The controller was implemented on Arduino Uno Board. In this paper position algorithm has been used for the implementation. To study the issues in implementing a digital PID Controller in Arduino microcontroller is the main objective behind this study. After completing all the signal conversions, the PID parameters for a particular controller are adjusted manually. Tuning of PID is done with the help of trial and error method.*

I. INTRODUCTION

PID controller is most widely used in industrial control system to get desired output by feedback mechanism. One of the main key difference between digital and analog controller is actual value is not measured continuously; rather it is periodically sampled at a fixed sampling interval. [1][2] As Arduino is low voltage operating robust microcontroller it is very good platform to implement PID on it. It's features like inbuilt ADC and PWM mode are very useful and reduces circuitry and complexity of the design. [3]

In order to demonstrate our work, we have chosen level as our controlling parameter .Level of the tank is measured with the help of level sensor or ultrasonic sensor. The measured analog values are given to analog pin of Arduino Uno which will convert into its digital equivalent. Now this digital value is compared with the desired set point value and generates an error signal. The error generated can be minimized and then these analog values can be passed on to the DAC module and then to I/P convertor which will convert the current signal to pneumatic pressure and then this pressure can control the control valve accordingly such that desired level will remain maintained in the tank . This level which we keep in our limit is very important considering the safety of the tank and the whole plant.

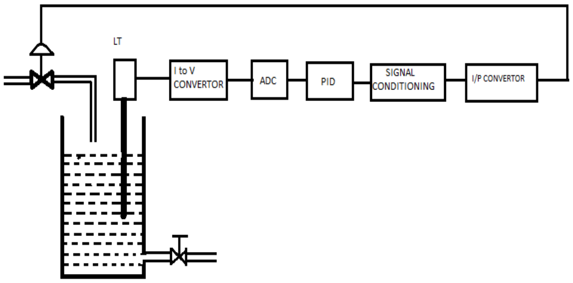


Fig 1: Block diagram

In this paper we have split our project in sections. Section 2 tells about the basic PID controller. Section 3 gives the specification of the arduino Controller, Section 4 talks about the designing of PID controller, the next section, Section 5 depicts the signal conversions used in our project and section 6 represents the flowchart of our program at last it represents the future scope of this project.

II. PID CONTROLLER

PID controller is the combination of proportional, integral and derivative controller modes. PID controller works in a closed-loop system using the schematic shown below. The variable represents the tracking error, the difference between the desired input value and the actual output. This error signal will be sent to the PID controller, and the controller computes both the derivative and the integral of this error signal. The control signal to the plant is equal to the proportional gain times the magnitude of the error plus the integral gain times the integral of the error plus the derivative gain times the derivative of the error.

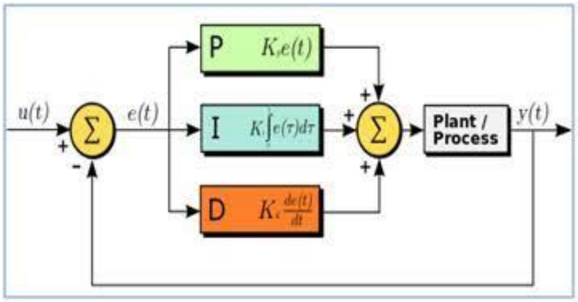


Fig 2: PID Block diagram

PID controller output equation

$$u(t)=K_p e_p + K_pK_i \int e_p dt + K_pK_d \frac{de_p}{dt} + u(0)$$

III. ARDUINO MICROCONTROLLER

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 an AC-to-DC adapter or battery to get started. The specifications of Arduino Uno are as follows

Arduino Uno has 14 digital input/output (I/O) pins. Conventional, i.e., not PWM, operation of the digital I/O pins is controlled with the pinMode, digitalRead and digitalWrite functions. The pinMode function is used to configure a pin as an input or output. When a digital I/O pin is configured as an input, digital Read reads the state of the pin, which will be either HIGH or LOW. When a digital I/O pin is configured for output, digital Write is used to set the pin voltage to HIGH or LOW. On an Arduino Uno, PWM output is possible on digital I/O pins 3, 5, 6, 9, 10 and 11. The specifications of Arduino Uno are as follows:

Microcontroller ATmega328
Operating Voltage 5V
Input Voltage (recommended) 7-12V
Input Voltage (limits) 6-20V
Digital I/O Pins 14 (of which 6 provide PWM output)
Analog Input Pins 6
DC Current per I/O Pin 40 mA
DC Current for 3.3V Pin 50 mA
Flash Memory 32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM 2 KB (ATmega328)
Clock Speed 16 MHz

IV. DESIGN OF PID CONTROLLER

PID control is a basic control loop feedback mechanism. The controller minimizes the difference between the measured and the desired value of a chosen system variable by adjusting the system control inputs.

During the programming part, first of all we need to initialize the value of setpoint,Ki, Kp, K_d value. There are two different modes available for tuning of controller that is either manual or in auto mode. Our program is based on trial and error method of selection of values of k_p,k_i and k_d. After several trials we were able to conclude the values of these parameters where we were able to get stable output.

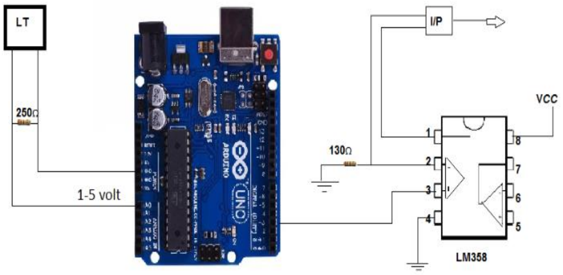
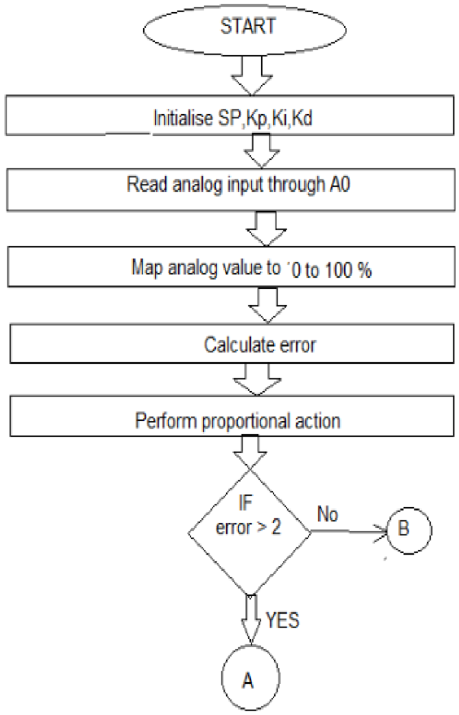


Fig 3 :- Circuit Diagram

The next step was to read the input from level transmitter. The output of any transmitter is usually 4-20mA which is widely used in industrial application. Then signal conversion is required to convert this 4-20mA signal to 0-5 V dc output. This is given as an input to the Arduino Uno. The output of the Arduino is taken from pin 3 which provides PWM output.

The output which is 0-5 V dc out is again converted to standard 4-20 mA signal which can actuate the final control element. The final control element in our case is control valve. I/P convertor circuit are used to complete this task.

V. FLOW CHART


START
Initialise SP,Kp,Ki,Kd
Read analog input through A0
Map analog value to '0 to 100 %
Calculate error
Perform proportional action
IF
error > 2
No
B
YES
A

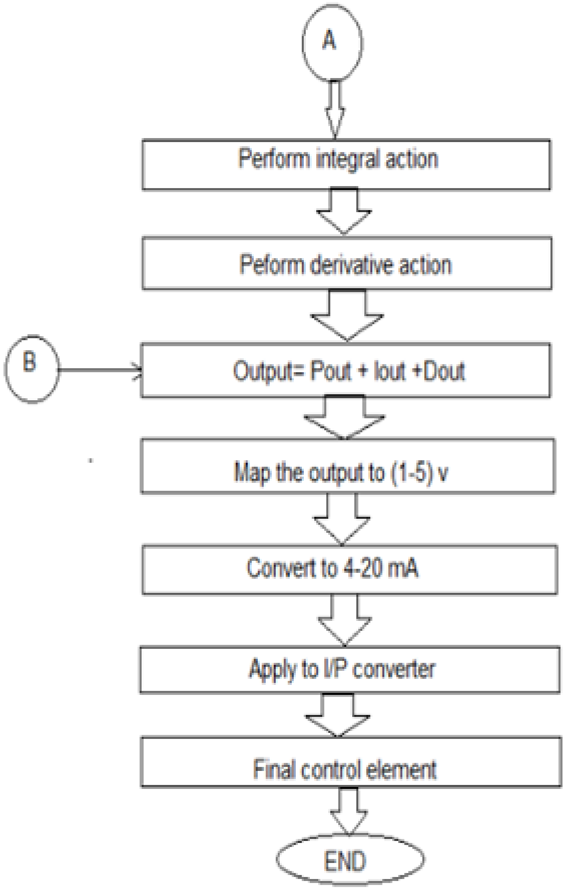


Fig 4:- Flowchart of PID algorithm

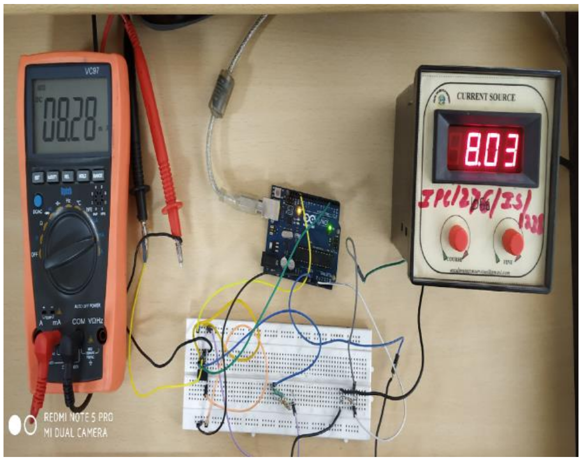


Fig:-5 Hardware Setup

VI. SIGNAL CONVERSIONS

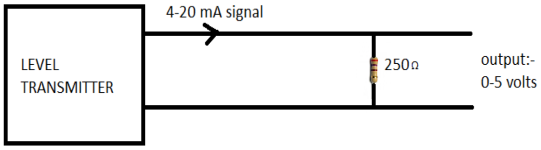


Fig 6:- I to V Convertor

Figure 6 represents the signal conversion required for converting the 4-20mA signal from the transmitter to the desired (0-5) V Output.

For testing we had used a current source which provides 4-20mA output and used a 250ohm resistor in between positive and negative supply. The output is measured across the resistor terminals. This output is send directly to the Arduino as Arduino has high input impedance. We can use any resistor in the range of 250ohm and then map the output accordingly in the Arduino program

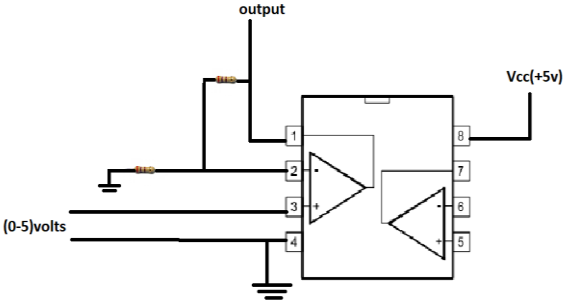


Fig:-7 V to I convertor

Figure 7 is a V to I convertor circuit. We have used a LM358 opamp IC which has dual opamp powered by common power supply. The main advantage of this IC is that it works on +5V supply. The (0-5) Volts is given to pin number 3 and 4 and it is grounded. The output 4-20 mA signal is obtained from the pin number 1 and 2. This current signal output is then given to the I/P converter which converts the current signal to equivalent pressure signal and control the actuation of the control valve and hence we can control the level of the tank.

VII. CONCLUSION

A detailed study of proportional plus integral plus derivative controller is done. A good understanding of Arduino Uno hardware and software and its programming platform with libraries and function is acquired. Different control algorithms are studied and interactive controller algorithm is chosen. We have successfully done the signal conversions part and thus maintained the level of the tank

VIII. FUTURE SCOPE

This implementation can replace the conventional PID controller. Even the cost of Arduino Uno is comparatively affordable and it helps in saving energy as it consumes less power.

REFERENCES

[1] Katsuhiko Ogata, “modern Control Engineering”, 5th Edition

[2] Norman S. Nise, “Control System Engineering “, 6th Edition

[3] https://www.arduino.cc/

[4] https://en.wikipedia.org/wiki/Arduino_Uno

[5] umar V., Nakra B. C. and Mittal A. P.,*Areview on classical and fuzzy PID controllers*, Int.J. Inte. Cont. and Syst., vol. 16, no. 3, 2011

[6] Ang K. H., Chong G., and Li Y., *PID controlsystem analysis, design and technology* , IEEETrans. Cont. Syst. Tech., Vol. 13, No. 4, Jul. 2005

[7] Arduino in Action Martin Evans, Joshua Noble, and Jordan Hochenbaum