1st International Conference on Science and Technology Innovation (ICoSTEC)
February, 26 2022. Yogyakarta, Indonesia
ISBN: 978-623-331-338-4

Simulation on the effects of the Arduino PID controller parameters using the WOKWI online simulator

Djoko Untoro Suwarno
Electrical Engineering Department
Faculty of Science and Technology
Sanata Dharma University
Email: joko_unt@usd.ac.id

***Abstract* - PID controllers are known in the industrial world as reliable controllers and are studied in universities. The explanation of the ideal PID mathematically often makes it difficult for students to understand the computational process and the implementation that occurs. Often there is difficulty in selecting the correct controller parameters, for example, the Kp, Ki, and Kd parameters.**
In this study, PID parameters were selected in the form of Kp, Ti, and Td and observed controller output. The Ti parameter as the time integral is easier to understand than the Ki parameter as the gain integral. The Arduino simulator used is wokwi which is an online Arduino simulator. The PID library used is the Arduino Brett Beauregard PID.
The results obtained are the effect of changes in the parameters of Kp, Ti, and Td on the controller output. For larger Kp, the controller output is proportional to the amount of input. The larger the Ti, the slower the system output, while the effect of Td is used when the input changes frequently

Keywords: PID simulator, Arduino PID, PID tuning, wokwi

I. INTRODUCTION

PID controller is a controller that is widely used in industry and PID is described ideally [1]-[3]. The description of PID for the industry by Setiawan [4], Implementation of the PID controller to control the plant is carried out by Rosada [5]. Ali [6] implements PID using Matlab. The implementation of PID using Arduino was carried out by [7][8]. Students in making projects with PID controllers often have difficulty understanding the processes that occur on the PID controllers. Arduino WOKWI Simulator is an online simulator for Arduino and other microcontroller modules released in 2021 by CodeMagic LTD.

The project uses Arduino and common modules such as LED, potentiometer, LCD, and some of the modules provided in the WOKWI simulator. Through the WOKWI simulator, users can try out Arduino projects virtually so that they can speed up creating projects using Arduino. In this paper, an Arduino-based PID controller simulation is performed using WOKWI.

In this study, visualization of each stage in the PID controller starting from the SP and PV signals was carried out as well as the calculation process on the PID controller and adjustment of the output signal.

Problems to be solved

1. Step visualization of operation on PID controllerusing WOKWI
2. Adjustment of the controller output signal to anactuator

II. METHOD

The block diagram of the PID controller simulation is shown in Fig. 1.

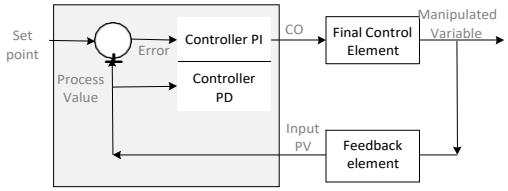

Fig 1. a PID controller diagram block

The controller section consists of the setpoint (reference point), the PV (Process Value) as input section, the error signal, the Proportional controller section, the Integral controller section, and the derivative controller section. Based on the error signal, the control mode is divided into direct-acting and reverse acting. Direct-acting is calculated from error = SP-PV, while reverse acting is calculated from error = PV-SP.

SP is used as a reference point. The SP value is obtained from the voltage on the potentiometer which is connected to the Arduino analog input A0

SP signal is obtained by:

1. manual mode through the voltage of the potentiometer
2. Up and down mode, triangle signal
3. pulse mode, with the pulse width determined by the length of time pressing the button
4. doublet mode

Manual mode is obtained from the potential voltage which is connected as a voltage divider

Manual mode is obtained from the potential voltage which is connected as a voltage divider and connected to the analog input A0

The up-and-down mode is generated programmatically with a lower bound Vb and an upper bound Va, a linear increment slop.

Pulse mode is triggered by pressing a button. If the button is not pressed, the CO (Controller Output) is assigned a lower limit value of Vb, when the button is pressed, the CO output is assigned an upper limit value of Va.

Doublet mode, triggered by the keypress. When the button is not pressed, the output is the middle-value Vn, whereas when the button is pressed, the output will be the upper limit. At the initial Vn = 500, Va = 900 and Vb = 100.

III. RESULTS AND DISCUSSION

Potentiometer Characteristics: The potentiometer used is linear. From the curve fig 2 obtained a partial linear potential (piecewise linear), that is at a distance of 20 mm to 50 mm (33% up to 77%) from a range of 65mm

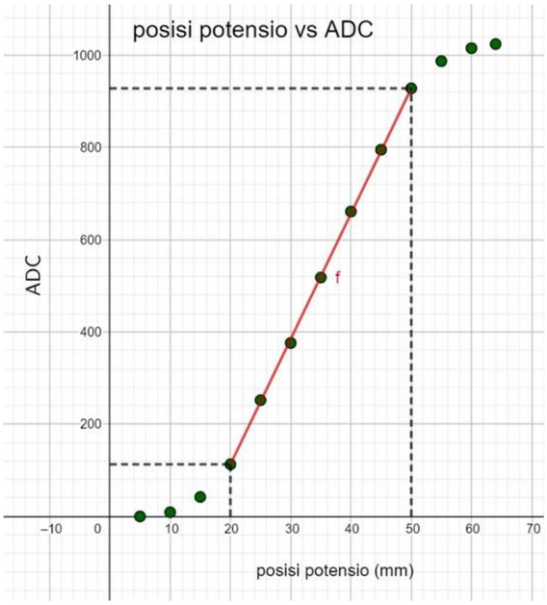

Fig 2. Potensio position vs resistance, ADC

A. *SP signal mode*

1. Manual mode via a potentiometer as a voltage divider and connected to the Analog in A0 Arduino. The output value for manual mode is 100 to 900. The graphic form that occurs is shown in Fig 3.

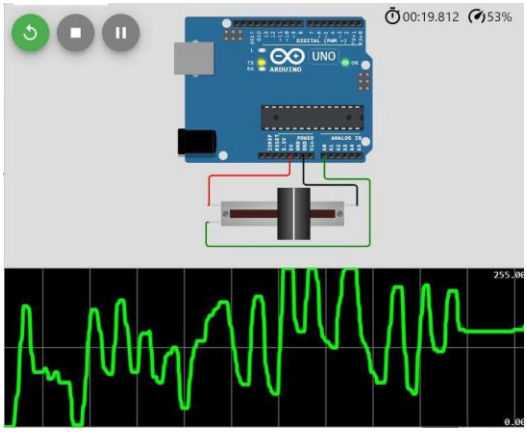

Fig 3. example of manual SP signal from potentiometer

2. Up and down mode, the output value starts from 100 up to 900 and down to 100 and repeats. The output form is a triangular wave. The period of the triangular waveform is determined by the delay between output points. For a delay of 1ms and a total point of 2x(900-100) = 1600 samples produce 1.6 seconds. The resulting triangular waveform is shown in Fig 4. If the SP is an input for the position control system, then the change in the SP value is the speed. An example of a position control system with a range of 80cm (ADC value = 1000) is shown in Fig 5. The movement speed can be calculated as follows: position delta/time delta = 48cm/0.8 seconds=60cm/s

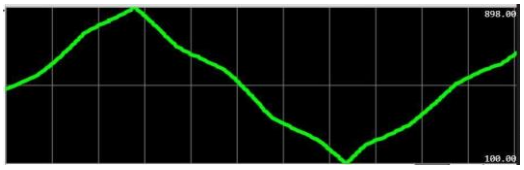

Fig 4. The resulting triangular waveform

3. Pulse code with a trigger from a button. When normal, the output value is 500, and when the button is pressed the output value is 900. When the button is released, the output value returns to 500. An example of a pulse wave is shown in Fig 6

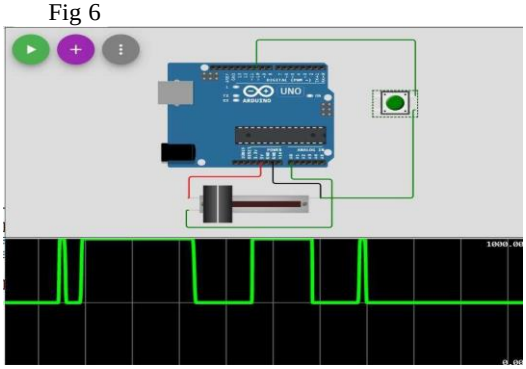

Fig 5. An example of a pulse wave

B. SP input unit

SP value as input and compared with PV to produce an error, it is necessary to pay attention to the unit used. For example, for a position control system, the SP unit is in the form of a distance unit in cm. The position sensor uses a potentiometer, so the unit is voltage. Fig. 6 shows the SP units that can be used: cm, Vpot, ADC, and %. The unit used in the calculation is better in the form of a percent with a normal number of 1 to 100%. The SP calculation uses a fractional data format (real, float).

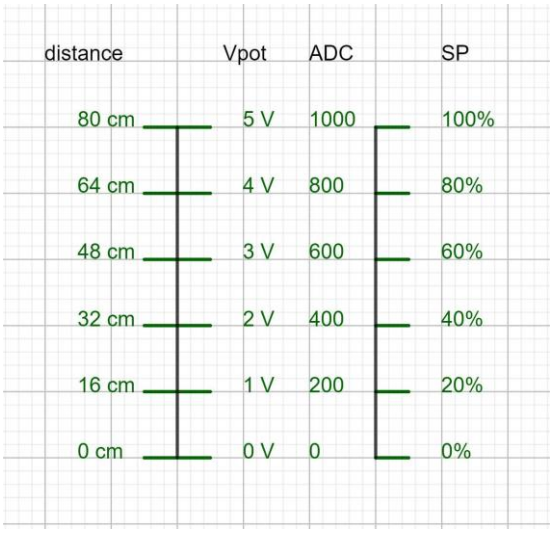

Fig 6. units SP in cm, Vout, ADC, %

C. Controller output unit.

There are several standards for controller output used in industry, namely the output voltage of 1V to 5V or in the form of a current of 4mA to 20mA. The other output is an 8bit PWM signal or connected to a servo motor, timing trigger Thyristor, or SCR

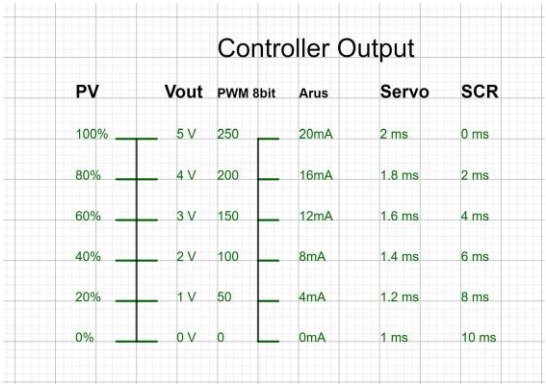

Fig 7. The controller output signal

D. Error Signal

The error signal is calculated from the difference between SP and PV

For direct-acting, the error calculation is as follows

$$Error = SP - PV \quad (1)$$

For reverse acting

$$Error = -(SP - PV) \quad (2)$$

The SP and PV values from the external are connected to the analog input from the microcontroller. Microcontroller analog input in the form of ADC. For 10-bit ADC, the ADC output starts from 0 to 1023. Error calculation is in percentage units.

E. Proportional Gain

The proportional gain is calculated based on the error signal and is shown in Eq 3.

$$CO = Kp \times Error \quad (3)$$

The results of the Controller Output calculation can produce positive values and can produce negative values. For the error signal = 0 (meaning PV = SP) then the controller output is set to zero and the actuator does not change.

An example of the output is a servo motor with a normal position at an angle of 0 degrees, a maximum angle of 90 degrees, and a minimum angle of -90 degrees. Then the normal position is in the 50% position. controller output needs to be offset by offset. An example of the output of Eq 3 becomes Eq 4.

$$CO = Kp \times Error + offset \quad (4)$$

The proportional gain value Kp is a function of the magnitude of the change in error. With a gain of 5x and a 10% error change, the controller output change is 50%. The gain is calculated from the change in output divided by the change in input as shown in Eq 5. A large proportional gain causes the controller output to quickly become maximal, which can cause the controller to go into ON/OFF mode.

$$Gain = \frac{\Delta ouput}{\Delta input} \quad (5)$$

Maximum gain is obtained from maximum output/minimum input. From table 1 it can be seen that the value of sp, PV, error, and CO on gain proportional = 2.

Table 1. SP, PV, Error, CO values

SP	PV	Err	CO
43.75,	39.84,	3.91	7.81
43.75,	39.84,	3.91	7.81
43.75,	39.84,	3.91	7.81
43.75,	39.84,	3.91	7.81
43.75,	39.84,	3.91	7.81

F. *Proporsional Gain and Proporsional Band*

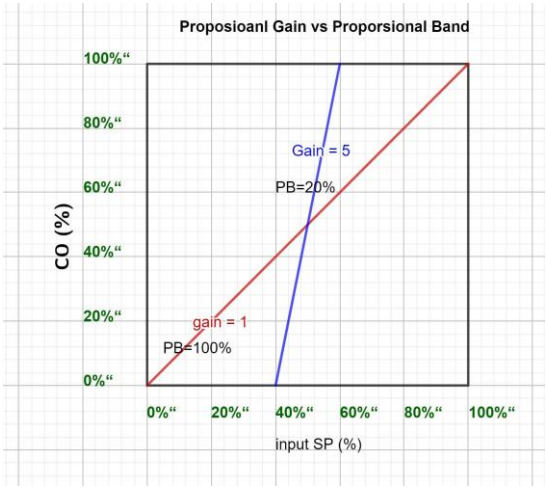

Fig 8. Proporsional Gain vs Proporsional Band

A proportional band is an indication of how wide or narrow the controller parameter settings are. The definition of the proportional band is the opposite of proportional gain as shown in Eq. 6.

$$PB = \frac{1}{Kp} \tag{6}$$

The larger the gain, the narrower the proportional band as shown in Fig. 8

H. *integral gain*

The integral controller part Ki is the reciprocal of the integral time as shown in Eq. 7.

$$Ki = Ti \tag{7}$$

With a large integral time, the system is slow. The integral for the input error which remains in the form of a ramp function is the larger it is. For a smaller integral time causes a larger integral gain, the system output quickly increases as shown in Fig 9. One of the effects of an integral part causes the output of the controller to go to a saturated (unchanged) output.

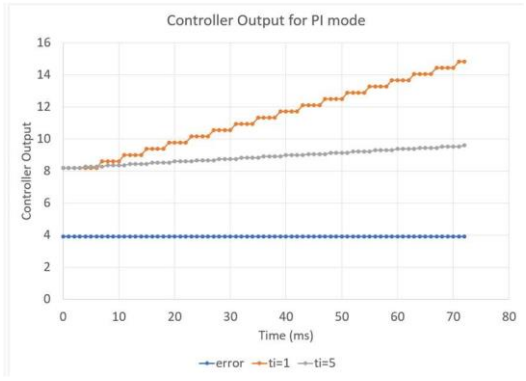

Fig 9. Controller PI mode

G. *Derivative gain*

The effect of the derivative part on the PID controller is that if the SP input changes rapidly it will cause a large error change. If the system does not change much, then the effect of the derivative part on the error is small. To overcome the occurrence of a derivative kick, the error change is calculated from the change in PV, not from the change in SP as explained in Eq 8-10.

$$error = SP - PV \tag{8}$$

So that

$$\frac{d}{dt}\, error = \frac{d}{dt}(SP - PV) \tag{9}$$

assuming that the SP is a constant then the differential SP = 0 so that

$$\frac{d}{dt} error = -\frac{d}{dt} PV \tag{10}$$

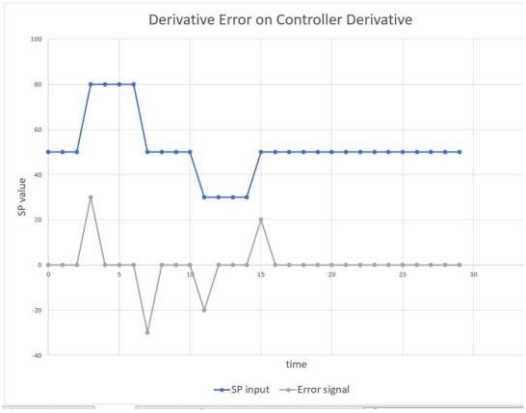

Fig 10. Derivative on Error

IV. CONCLUSIONS

From the research results can be concluded as follows:

1. Wokwi is used as an Arduino simulator, and you can add additional input-output modules visually
2. as a PID controller on Arduino requires a PID library
3. Changes in PID parameters can be seen on the serial monitor and the selected output device
4. The controller output needs to be adjusted to the input and output devices, it is recommended to use percentage units

ACKNOWLEDGMENT

The authors would like to thank the Center for the Study of Intelligent Technology (PKTC), the Faculty of Science and Technology, and LPPM USD for supporting this research.

REFERENCES

[1] Ogata, Katsuhiko. Modern control engineering. Prentice hall, 2010.Seborg, Dale E., et al. Process dynamics and control. John Wiley & Sons, 2017.

[2] Nise, N. S. Control System Engineering, 6th edition. New Jersey: John Wiley and Son. 2010

[3] Pedro Albertos, I. M. Feedback, and Control for Everyone. Berlin: Springer, 2010

[4] Setiawan, Iwan. Kontrol PID untuk proses industri. Elex Media Komputindo, 2013.

[5] Rosada, Kevin. Sistem Kontrol Pompa Air Menggunakan Kontroler PID Berbasis Raspberry Pi. Diss. Institut Teknologi Sepuluh Nopember, 2017.

[6] Ali, Muhamad. "Pembelajaran Perancangan sistem kontrol PID dengan software MATLAB." Jurnal Edukasi@ Elektro 1.1 2004.

[7] Visioli, Antonio. Practical PID control. Springer Science & Business Media, 2006.

[8] Bista, Dinesh. "Understanding and Design of an Arduino-based PID Controller." 2016