GRAPHIC USER INTERFACE (GUI) OF A PID CONTROLLER SIMULATOR USING VISUAL BASIC.

Azlinda bte. Aznam
Faculty of Electrical Engineering,
University Teknologi MARA,
40450, Shah Alam, Selangor,
Malaysia.
E-mail: dot_santa@yahoo.com

Abstract – In this project, it is aimed to develop and design a graphic user interface (GUI) for water level measurement system using Visual Basic. The design is based on the hardware available in the laboratory at the Faculty of Electrical Engineering, UiTM, Shah Alam. The system was design in such a way that users are able to enter the PID parameters and display the output response of the system. The design also used the animation technique to show the flow of water. The system performance is compared to the actual experiment in term of output response generated for appropriate PID setting. This is a hope that the success of this project will be an alternative for students to conduct an experiment.

Keywords: PID controller, visual basic and 3-tank experiment.

1.0 INTRODUCTION

The PID algorithm has been successfully used in the process industries since the 1940s and remains the most often used algorithm today. Major manufacturing and chemical process industries have been using PID controllers in the automatic control system since that year. Since then, it has evolved from a pneumatic mechanical to a digital electronic device. Often, people use PID to control processes that include heating and cooling systems, fluid level monitoring, flow control, and pressure control. Continuous feedback control offers the potential for improved plant operation by maintaining selected variables close to their desired values. [1]

The performance of the entire feedback system depends on the structure of the algorithm and the parameter used in the algorithm. All other elements are process equipment and instrumentation, which are costly and time-consuming to alter, so a key area of flexibility in the loop is the control calculation. [1]

PID controllers are capable stabilizing processes at any set-point by utilizing a mathematical function in the form of the control algorithm. It may seem surprising that one algorithm can be successful in many applications - petroleum processing, steam generation, polymer processing and many more. [2]

The PID algorithm is a simple, single equation, but it can provide good control performance for many different processes. This flexibility is achieved through several adjustable parameters, whose values can be selected to modify the behaviour of the feedback system. [2]

The procedure for selecting the values is termed tuning, and the adjustable parameters are termed tuning constants. Each element of the algorithm is termed a mode and uses the time-dependent behaviour of the feedback information in a different manner, as indicated by the name proportional-integral-derivative. [3]

1.1 The Theory of the PID Controllers.

PID controllers are undeniably the most commonly used control algorithm for industrial processes. In spite of their simple structure, they often perform well and can meet the specifications provided that their parameters are properly chosen.

Currently, there is several equation of the PID control algorithm. The PID controller compares the setpoint (SP) to the process variable (PV) to obtain the error (*e*) as refer in equation (1):

$$e = SP - PV \quad \text{..........} \quad (1)$$

Then the PID controller calculates the controller action, *u* (*t*), where *Kc* is controller gain as refer in equation (2):

$$u(t) = K_c\left(e + \frac{1}{T_i}\int_0^t e dt + T_d \frac{de}{dt}\right) \quad \text{..........} \quad (2)$$

From the equation (2), K_c is the proportional gain, or simply gain. Conventionally the gain is applied across the other two modes, as well, as indicated in the definition. It seems logical for the first mode to make the control action (i.e, the adjustment to the manipulated variable) proportional to the error signal, because as the error increases, the adjustment to the manipulated variable should increase. The proportional mode calculation, as refer in equation (3):

$$U_p(t) = K_c e \quad \text{..........} \quad (3)$$

T_i is the integral time. Its reciprocal is often called the reset rate. This reset rate has dimensions of "repeats/time". This is because at constant error input ΔE, the controller output will increase by $K_c\Delta E$ in each time increment T_i. Deactivate integral mode by setting T_i to infinity. Short T_i, or high reset rate, represents more aggressive control response. To eliminate residual system error or (steady-state error) e_{ss} the controller's response must be changed. The integral controller has an output whose rate of change is proportional to the error. In most practical system this time-variant output alone, it is too slow. It usually coupled with *P* controller. The following formula represents the integral action as refer in equation (4):

$$U_I(t) = \frac{K_c}{T_i}\int_0^t e dt \quad \text{..........} \quad (4)$$

T_d is the derivative time. It is also called preact. Deactivate derivative mode by setting T_d to zero. Large T_d represents more aggressive control response. The derivative mode is applied only to the controlled variable and not the error term as shown in the equation. By this, set point changes would not lead to sudden manipulated variable changes, driven by the derivative mode. The *P+I* controller will remove all steady-state error and in many process the *P* part will provide adequately fast response especially to steps error. However, process with large inertia need some form of 'Kick' in response to a step in error. These overcome the inertia, providing with faster response to error steps than even a *P* controller can given. The following formula represents the derivative action, as refer in equation (5):

$$U_D(t) = K_c T_d \frac{de}{dt} \quad \text{..........} \quad (5)$$

Controller output is the summation of the proportional, integral, and derivative action, as shown in the following formula (6):

$$U(k) = U_P(k) + U_I(k) + U_D(k) \quad \text{..........} \quad (6)$$

To obtain the equation that a computer can implement the continuous differential equation must be converted to a discrete differential equation as refer to the equation (7):

$$V_{MV} = K_p V_e + K_I \int V_e dt + K_d \frac{dV_e}{dt} + V_o \quad \text{......} \quad (7)$$

In the computer-based system, at the real world there is only each cycle. The cycle time *T*, set this time interval $dt = T$. After a few steps then we finally get the equation that we should put it in the programming of visual basic which refer to the equation (8):

$$V_o = V_{on-1} + K_p(e_n - e_{n-1}) + K_I e_n T + \frac{K_D}{T}(e_n - 2e_{n-1} + e_{n-2}) \quad \text{......} \quad (8)$$

The units of *I* and *D* may differ from one instrument manufacturer to another instrument manufacturer. For instance, the *I* constant, one manufacturer may use min per repeat (integral time) while other manufacturer may use repeat per min (integral gain). Integral time and integral gain is inversely related to each other. Combining the proportional, integral and derivative controllers produces the three mode controller. It offer rapid proportional response to error while having an automatic reset from *I* part to eliminate residual (steady-state error). The derivative section stabilizes the controller and allows it to respond rapidly to changes in error. [4]

1.2 The GUI by Using Visual Basic

Visual basic is a high level programming language evolved from the earlier DOS version called BASIC. It is a fairly easy programming language to learn. Visual Basic is a visual and events driven Programming Language. In BASIC, programming is done in a text-only environment and the program is executed sequentially. The older version of VB was derived heavily from BASIC and enables rapid application development (RAD) of graphical user interface (GUI) applications, access to databases using DAO, RDO, or ADO, and creation of ActiveX controls and objects [5]. In Visual Basic, programming is done in a graphical environment. Because users may click on a certain object randomly, so each object has to be programmed independently to be able to response to those actions (events). Therefore, a Visual Basic Program is made up of many subprograms, each has its own program codes, and each can be executed independently and at the same time each can be linked together in one way or another. [6]

As we know that the tuning for the PID algorithm by using the traditional way is quite difficult as the good control performance can be achieved with a proper choice of tuning constant values, but poor performance and even stability can result from a poor choice of values. So as to improved this system we can apply it into the visual basic program so that we could find the accurate result for the PID optimum algorithm. This paper introduces a new approach in modelling the PID tuning by using the visual basic software where we managed tuned the PID algorithm by automatic way which the adjustable variable is from computer panel. By taken from the experiment of PID controller tank as a reference of PID algorithm we managed to apply in the software by drawing the tank animation, beside build the hardware which is cost very expensive for its components. It is a cheap usage, easy and accurate to get the optimum value for the fine-tuning.

2.0 METHODOLOGY

In this project, GUI by using Visual Basic is used to design a water level measurement system that available in the laboratory at Faculty of Electrical Engineering, UiTM Shah Alam. The design includes the system performance for first order system, second order system and when the disturbance is applied to the system. The animation techniques also include in the design so that the flow of water can be visualised. The overall aspects of the project are as follows:

2.1 System Design

The Fig. 1 shows the block diagram for the whole process. In PID control, we must specify a process variable and a setpoint. The process variable is the system parameter that we want to control, such as temperature, pressure, or flow rate, and the setpoint is the desired value for the parameter you are controlling. A PID controller determines a controller output value, such as the heater power or valve position. The controller applies the controller output value to the system, which in turn drives the process variable toward the setpoint value.

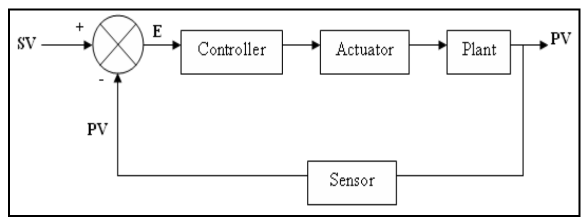

Fig. 1 Block diagram of the overall process control.

The complete PID equation, which is the sum of the three modes as shown in Fig. 2, is then reviewed and a few example control response are presented. There is no consistency in commercial control equipment regarding the sign of the subtraction when forming the error.

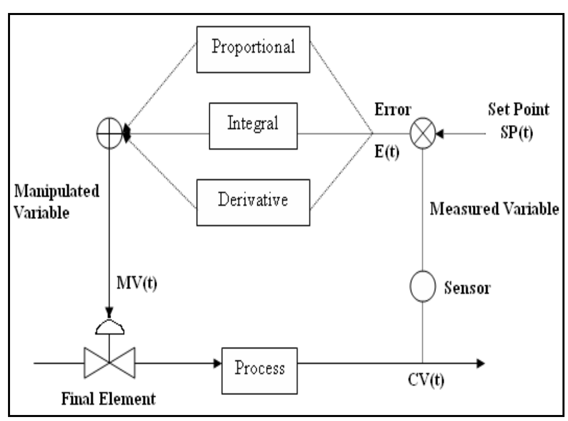

Fig. 2 Overview schematic of a PID control loop.

2.2 Flow Chart

The Fig. 3 shows the flow chart of the process by using the visual basic. By referring from the 3-tank experiment and applying it in the visual basic software where, I used the flow level system as to show how this PID system works.

- Literature review: Study on the existing research related to the PID controller.
- Implement visual basic system to get the accurate PID measurement for this flow level response.
- Design and develop accurate system using as follows:
    1. Ensure that when the water that pumps from the inlet valve and the outlet valve is reaching the setpoint although in any position of percentage.
    2. Ensure that when we enter the PID variable it must develop the stabilize system as the water come in and out.
- Design and develop the accurate element to the system so that the required output as mentioned will be achieved.

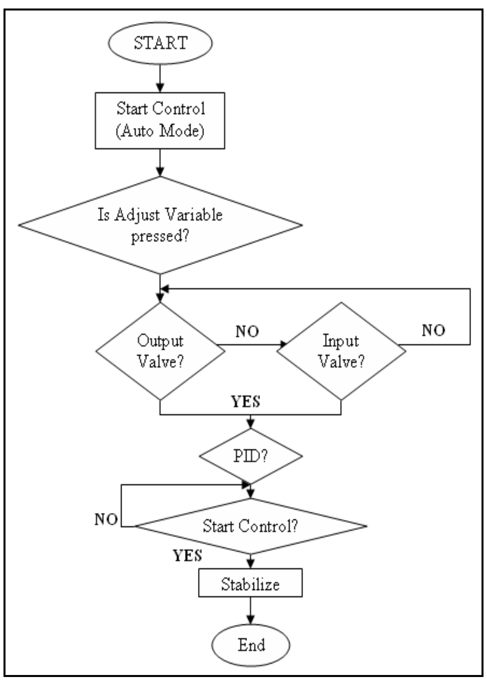

Fig. 3 The flow chart of the process by using the Visual Basic.

2.3 Software Development.

From the Fig. 4 shows the layout that taken from the visual basic for this PID simulator. By referring from the 3-tank experiment it has been apply in the visual basic to see how the flow level response based on the characteristic. Here the proportional is represent as gain, integral as reset and the derivative as rate. The gain value is range between 0% - 100%, while the reset and rate value is range between 0 seconds - 120 seconds. The inlet valve and the outlet valve are adjustable from the 0% - 100% position. There is a 'Start Control' button which functions to stabilize the water level within in any value and condition. The 'Adjust Variable' button is proposed to adjust any variable such as the inlet and outlet valves, proportional, integral and derivative controller. Beside that there is the 'Disturbance' button that simulates a sudden change in the flow of water be it into or out of the process. When we already adjust the variable on what we desire we must press the 'Start Control' again to ensure that the water is stable. There are two graphs which represent the 'Process Variable (PV)' and the 'Output Valve Position'. The output valve position is range between 0% - 100%.

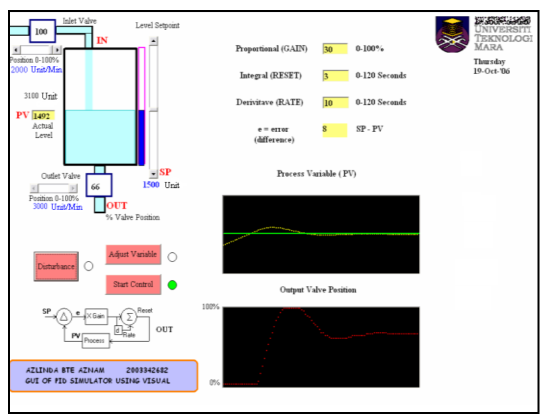

Fig. 4 The layout of the interface for the PID simulator.

3.0 RESULT AND DISCUSSION

In this session there are results which are the system performance in stable mode, in unstable mode and the system when the disturbance is applied.

3.1 The System Performance in Stable Mode.

From the Fig. 5 shows the layout of the whole process using the visual basic. Here, this system is set at 1500 unit as the setpoint. When we press the ‘Start Control’ button, this system is stable with this initial value that shown in Fig.7, as the water flow in to reach the level setpoint.

As in Fig. 6 shows more clearly about the tank animation on how the water flows in and out in reaching the setpoint. To reach the setpoint, the inlet valve is in position of 100%, where the water has been pumped for 2000 unit/min. While for the outlet valve it is in position of 66%. For this moment, the process variable did not reach the exact setpoint, because of the certain calculation on it. The amount of process variable or the actual variable is 1492 unit.

Furthermore for this circumstance the amount of proportional (gain) is 30%, integral (reset) is 3 seconds, and the derivative (rate) is 10 seconds. Here there is a few amount of error that is 8 when we compared between the setpoint (SP) and the process variable (PV) as shown in Fig. 7.

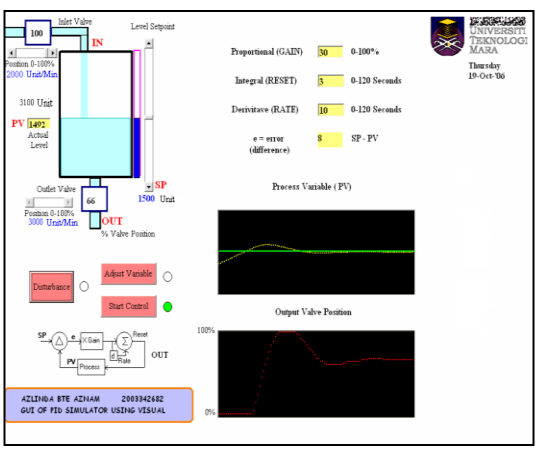

Fig. 5 The layout of the interface.

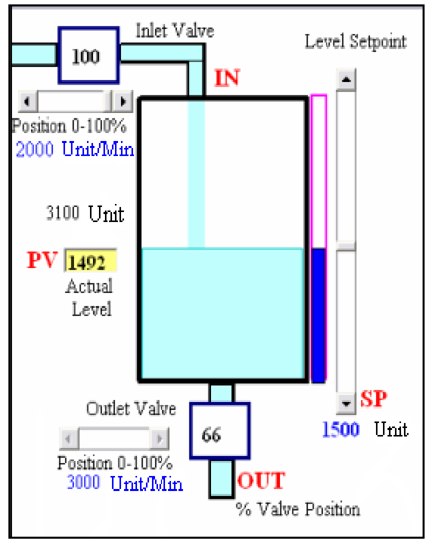

Fig. 6 The water tank animation.

Proportional (GAIN)	30	0-100%
Integral (RESET)	3	0-120 Seconds
Derivitave (RATE)	10	0-120 Seconds
e = error (difference)	8	SP - PV

Fig. 7 The gain, reset and rate value when it is in stable.

This Fig. 8 shows the graph that represent the process variable (PV) and the output valve position. For the process variable graph the green line shows the setpoint. When we pressed the ‘Start Control’ button the water will increase smoothly then it will stable for certain time as shown in yellow line. While the output valve position which represent the red line, it will increase suddenly and it will stable again which is shows at position of 66%. From Fig. 9 shows the graph that been taken from the 3-tank experiment. Here it looks quiet similar with the simulation in the visual basic, by using the same value of PID controller.

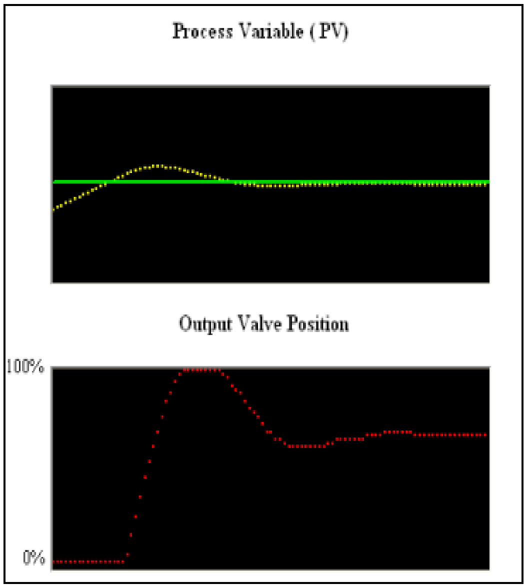

Fig. 8 The graph process variable (PV) and the output valve position.

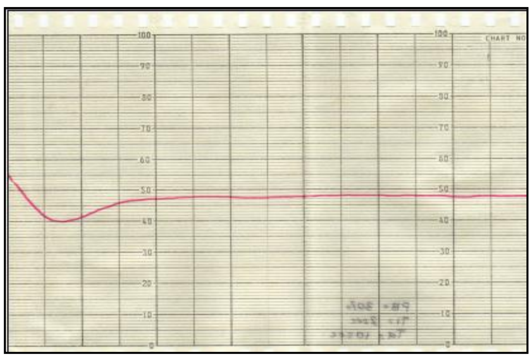
Fig. 9 Output response from the graph.

3.2 The System Performance when it has been Adjustable.

For the next example, to see more on how this system works, now gain, reset and rate value has been adjustable as shown in Fig. 10 where the gain is 68.5%, the reset is 24.2 seconds and the rate is 6.05 seconds. For this moment the error that has been produced is 16. This also can be seen in Fig. 11 as shown in layout of the tank animation. The process variable (PV) at this moment is 1484 unit.

To see more clearly how the output valve position at this moment it can be seen in the graph which is shown in Fig. 12. From here we can see that it will be in 100% position for a few times and it suddenly decreases to 0% position for a short time. This shows that the water is not quite stable to reach the setpoint.

From the Fig. 13 shows the original graph that has been taken from the 3-tank experiment with using the same value of PID controller.

Proportional (GAIN)	68.5	0-100%
Integral (RESET)	24.2	0-120 Seconds
Derivitave (RATE)	6.05	0-120 Seconds
e = error (difference)	16	SP - PV

Fig. 10 The value of gain, reset and rate after it has been adjustable.

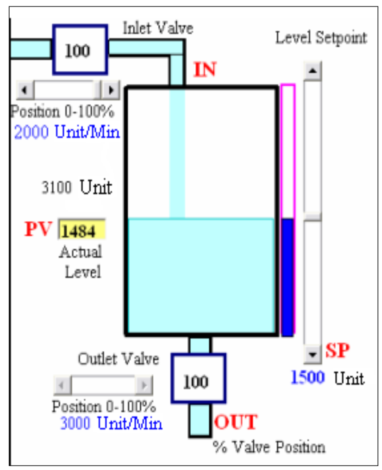

Fig. 11 The tank animation.

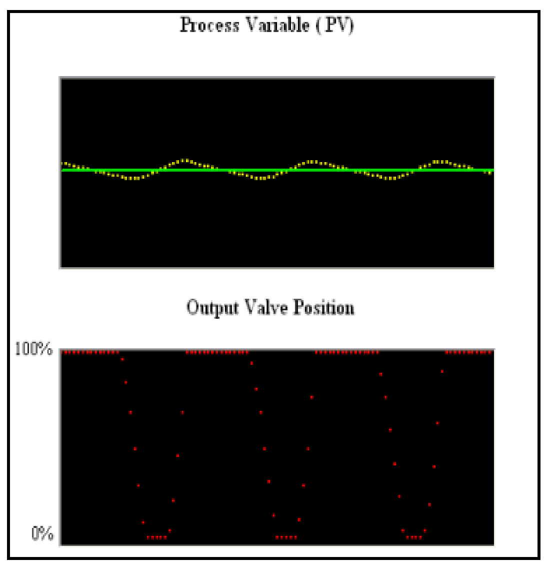

Fig. 12 The process variable (PV) graph and the output valve position graph.

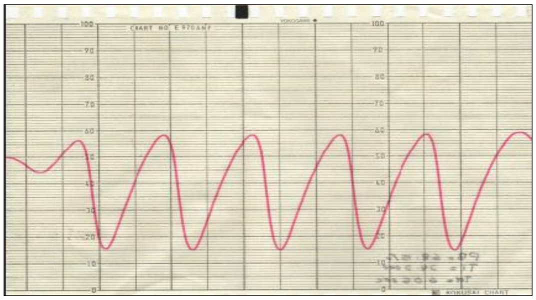

Fig. 13 Output response from the hardware.

3.3 The System Performance when the Disturbance is Applied.

In Fig. 14 shows that we did a next testing which is on disturbance by referring the same value of proportional, rate, and reset during the water level is in stable condition. Here we can see that, after punching the button of 'Start Control' in condition of disturbance we can see that the value of error is 5 after comparing between the setpoint and the process variable.

As refer to the Fig. 15, the layout of tank animation shows that the outlet valve is in position of 65%, while the inlet valve is in position of 100%, where it need 1946 unit/min to pump water in the tank. During that time the process variable (PV) which is the actual level is at 1495 unit.

To see more clearly between the process variable (PV) and the output valve position graph by referring the Fig. 16, it shows the graph when we introduce the disturbance on it. From the graph we can see that when the disturbance has been applied the water that flow in is trying to stable but it takes for a while to stable the level of water response to the PID controller. Fig. 17 shows the original graph that has been taken from the 3-tank experiment with the same value of PID controller.

Proportional (GAIN)	30	0-100%
Integral (RESET)	3	0-120 Seconds
Derivitave (RATE)	10	0-120 Seconds
e = error (difference)	5	SP - PV

Fig. 14 The value of gain, reset, rate and error during the disturbance condition.

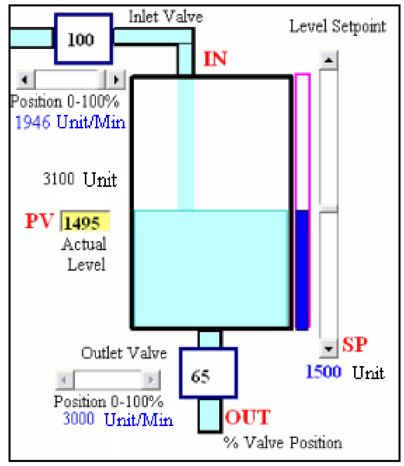

Fig. 15 The tank animation.

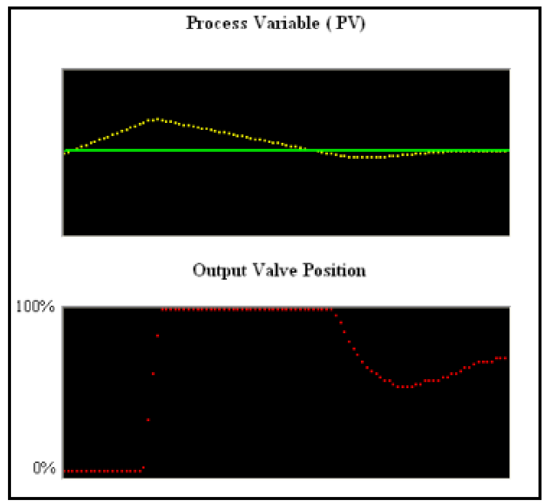

Fig. 16 The process variable (PV) graph and the output valve position graph during the disturbance.

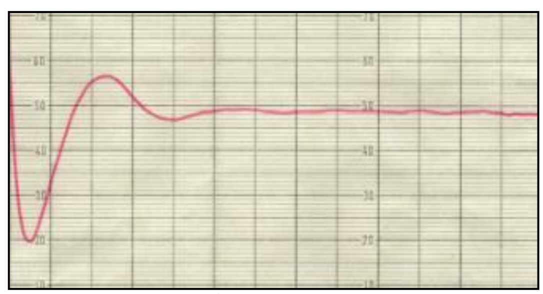

Fig. 17 Output Response from the hardware.

4.0 CONCLUSION AND FUTURE DEVELOPMENT

The network architecture that created was found suitable for the predicting and analyze to get a good measurement system in tuning the process of PID simulator. This paper also shows the benefit of the *P*, *I* and *D* controller, where it depends on each other to get a good performance of output in certain processes. PID is a commonsense approach to control based on the nature of error. It is defined in abstract terms and so can be applied to wide varieties of systems. It was developed long before computers, so it has been executed in a variety of hardware; it is adapted to microprocessors today. It is not 'optimal control' but can be successfully applied over a range of conditions in real processes.

Moreover this paper also gives a good opportunity as a tool for the industrial control and students because it is easy and more accurate in getting the fine-tuning measurement, where it can be controlled from the computer panel by just pressing a single button.

This application in future can be improved into interface between the hardware and software where it gives more benefit, faster and easier for them in learning more about the PID controller in effective way.

ACKNOWLEDGEMENT

The author would like to express sincere appreciation to Pn. Zuriati bte Janin as a supervisor, En. Zain Azlan, Dr. Anuar and Pn. Halimatun, for their valuable comments.

REFERENCES

[1] Process Control "Designing Processes and Control System for Dynamic Performance, Thomas E. Marlin, Copyright 1995 by McGraw-Hill, 2nd Edition.

[2] Process Control System "Application, Design, and Tuning", F.G.Shinskey, Copyright 1979 by McGraw-Hill, Inc, 3rd Edition.

[3] "Reformulation of the Tangent Method for PID Controller Tuning", Abdul Aziz Ishak Muhammed Azlan Hussain, Department of Chemical Engineering Faculty of Engineering, Universiti Malaya, 50603 Kuala Lumpur, Malaysia, TENCON2000, IEEE Region 10, Kuala Lumpur, 24 –27 September 2000, aabi63@hotmail.com **http://aabi.tripod.com**

[4] "Overview of The PID Control Toolkit, 1996 – 2006 National Instrument Corporation, **www.ni.com/info**.

[5] "Complete Idiot's Guide to Visual Basic 6", Clayton Walnum, Macmillian Computer Publishing, 15/12/98, www.itknowledge.com/faq/faq.html.

[6] "Teach Yourself Visual Basic In 24 Hours", Arthur Luhigo, http://www.geocities.com/mimituu