TOTAL ENGINEERING SERVICES TEAM, INC.

RTU / SCADA SYSTEMS

PID CONTROL OPTION

Document No. 1160-01

P.O. Drawer 1760
671 Whitney Ave.
Gretna, La. 70054
(504) 368-6792

Contact Arthur Zatarain, PE
via www.artzat.com
For information on this document

TOTAL ENGINEERING SERVICES TEAM, INC.
RTU / SCADA SYSTEMS
PID CONTROL OPTION

INTRODUCTION

This document describes the Proportional-Integral-Derivative (PID) control option in the TEST RTU/SCADA system. The control algorithm is a classic PID loop with several enhancements suitable for industrial and oilfield applications. Both open and closed loop methods are supported. This document is not an in-depth discussion of PID loops. It provides information on how the PID control option is implemented in the system and offers tips and suggestions for its use.

The PID loop is provided as a new channel type derived from the Value channel. It has all of the normal Value channel attributes such as alarm mode, units, and decimal palaces. The PID type adds in the features needed for the control algorithm, and these are covered below.

PID BASICS

PID control is a classical but complex method of maintaining control of a dynamic system. The PID control system monitors a process signal (measured variable or MV) and compares its value to a setpoint (SP). The output of the PID will control another variable that will in some way be related to the measured variable. The typical example used to explain PID systems is a hot water heater. The measured variable will be the water temperature. The setpoint will be the desired temperature. And the output will control the amount of heat added to the water tank.

Note that the input and setpoint variables are in the same units, such as Deg-F. The output is also in the same units, although actual practice avoids thinking of the output in terms of normal values. Instead, the output is normally thought of in terms of percent full scale for the output. The PID calculates an output value that would be "added" to the input value to make it equal the desired setpoint. In our example, we cannot add degrees. We can add heat, so there must be a relationship between added degrees and heat input. This is where the complexity comes in.

Most real-life systems such as a flow valve or a water heater have many interrelated variables involved. If our example is simple, we can adjust the heat input to match the heat output. A given heat input will cause the temperature to remain stable *as long as nothing changes.* This is not normally the case. Something as simple as a water heater can have many dynamic conditions such as:

1. How much hot water is being used at the moment?
2. How fast is the water usage changing?
3. How much water is in the tank?
4. How good is the gas being used to heat the water?
5. How much heat is being lost through the tank wall?
6. What is the current desired temperature?

These sorts of variables are common in any control system. The water tank is used here because its easy to understand. The point of the PID algorithm is to provide optimum control for a dynamic system having changes such as those listed above. This is done by monitoring the input signal according to a complex formula, which we will not cover in detail here. The end result is what is important, and this can be understood without ever looking at a complicated math equation. But for those who are really interested, the TEST PID calculation uses a common digital 'position method' based on the following equation:

$$m(i) = [Kc * e(i)] + [T * Ki * \sum_{k=0}^{i} e(k)] + [(Kd/T) * (e(i) - e(i-1))]$$

Where:

T	Sampling interval
e(i)	error at ith sampling intercal = S(t) -X(t)
e(i-1)	error at previous sampling interval
m(i)	controller output deviation (PID output)
Kc	Proportional Gain
Ki	Integral action time
Kd	Derivative action time

The P, I, and D in the PID term do not mean "Piping and Instrument Drawings." The letters stand for Proportional, Integral, and Derivative. These terms refer to the three components of the complex equation that calculates the output of the PID loop. Understanding how each one relates to the real world is the key to understanding the PID system. In quick terms, each factor relates to the following:

Proportional	- How far is the input from the setpoint?
Integral	- How long have we been off the setpoint?
Derivative	- How quickly are we approaching or departing from the setpoint?

Note that alternate terms are often used for Integral and Derivative. The Integral term is also called RESET because it attempts to restore the input to its setpoint. Derivative is often called RATE because it relates to the rate of change of the input. This document will stick to the PID terms because they are the classic names found in most control material.

P - PROPORTIONAL TERM

Understanding proportional control is the key to understanding the entire PID system. The Proportional output is directly related to the difference between the input and the setpoint. If the input is 32, and the setpoint is 40, then the proportional output is based on the value 8 (40-32). The *difference* is what is important, not the absolute value of the variables. The proportional output is given a factor called GAIN that is simply a multiple of the calculated difference. If the gain is one, then our difference of 8 will produce an output of 8. However, if the

gain is only 0.5, the difference of 8 will produce an output of 4 (one half of 8).

TEST's system always uses gain as the method of expressing the proportional term. Some systems use the inverse of gain times 100, called Proportional band, but this is confusing to most people. This document will always refer to the proportional term as a number that represents the multiplier for the difference error. A higher gain will produce a higher output for a given difference from the setpoint. Many systems get by fine with a gain between 0.1 and 1.0, although any gain number is possible. It all depends on how fast the system can react to changes in control.

The proportional output forms the basis of the other two output controls. Everything is based on the current difference between the input and the setpoint. If there is no error, then there is no output. This is a problem for many real world systems because a zero output will not normally produce the desired input variable. Consider the cruise control in a car. If we are at the desired speed, then the control cannot tell the accelerator to turn off all the gas. There is no error, but we still need some gas. That is the basic problem with proportional control. In order to produce an output, the input must be incorrect.

Simple control systems often live with this error and get by just fine. Your home air conditioner is a good example. You set your thermostat to produce a comfortable temperature. When it is reached, the unit shuts off. The temperature changes in a matter of minutes. When enough error is sensed by the thermostat, it kicks the unit back on. The temperature adjusts, and the cycle completes. But note that no temperature control occurs unless there is an error.

If you dial in 74 degrees you would expect to get a room that was maintained at exactly 74 degrees. This is not usually the case because the temperature is sensed at only one point in the room. A temperature of 74 at the thermostat may mean that its 81 degrees were you are sitting. If you are too warm, you will adjust the thermostat downward so that the *resulting temperature* at the desired point is 74 degrees. This may require a thermostat setting of 68, and that is where the manually entered error comes in. This setup will work fine once the magic setting is found and nothing changes. But different weather or more people in the room will change the dynamic conditions such that the magic number no longer works.

Room temperature is not that critical, and we don't mind adjusting the thermostat every now and then. Industrial control systems are usually more demanding, and we need a way to have the system automatically adjust to load changes. The other two terms of the PID loop take care of this by further manipulating the normal proportional output to maintain the desired setpoint under changing conditions.

I - INTEGRAL or RATE TERM

The second term, Integral, monitors the time that an input is off the setpoint. It then adds or subtracts from the proportional error to cause additional (or less) corrective action. The Integral term does not act alone. It only alters the effects of the proportional term. Proportional was specified in gain, which has no units. A gain of 1.2 is simply 1.2 times the difference. Integral is specified in units of minutes. A value of one means that departure from the setpoint for one minute will result in one unit of proportional correction being *added* to the current output.

The value given to the integral term tells the PID loop how many equivalent proportional corrections to make for every minute that the input is off the setpoint. This is a simplification, but is close enough for now. So, a steady state system that is off the setpoint will have its PID output change every minute by an additional value equal to the proportional output for the same setpoint-input difference. A larger value will cause a higher change to the output for a given amount of time away from the setpoint. Note that the integral term is additive, and produces an ever increasing (or decreasing) error correction as long as the input remains off of the setpoint.

Integral is helpful in situations where the load on a system changes from time to time. In our water heater example, the desired temperature will be maintained by proportional alone as long as the amount of water flow does not change. We could easily balance the heat-in with the heat-out by adjusting the desired temperature until we found an error that did the trick. But we would always be off the setpoint, and would have to get the desired temperature by artificially raising the setpoint to generate the desired error.

The integral term will keep an eye on the input value and make adjustments to the output based on time as well as error. Eventually, the output will be sufficiently adjusted to cause the desired effect on the system. The setpoint will be reached, but the output will not be zero. The output will be the value necessary to compensate for the load on the system *at that moment*. Changes in the load will be detected over a period of time, and the system will balance.

D - DERIVATIVE or RATE TERM

The third term in PID is derivative, or rate. It monitors changes in the input to see how fast or slow it is changing relative to the setpoint. Slow moving changes do not require much compensation in order to keep the system stable. But a fast moving input will require anticipation on the part of the PID loop in order to keep things from getting too far off too quickly. Derivative is the least often used term because most systems have a fairly consistent rate characteristic. But small values for the Integral term can often be used to cause an improved response in systems that can get sluggish if the input strays too far from the setpoint.

The Derivative term is also specified in minutes. It tells the PID calculation how many minutes it will take to reach a certain level of output in advance of the time in which the proportional output would have done the same thing. In effect, the derivative term *anticipates* the required output and jumps ahead of the proportional value.

NORMAL AND REVERSE ACTING CONTROLLERS

The PID calculation uses a positive error equal to the Setpoint Minus Input. An input that is lower than the setpoint will generate a positive error. A higher input will generate a negative error. The normal thinking process tells us that if an input is lower than the setpoint we will increase an output to make the correction. If the temperature is to low we want to add more heat to the tank. Too slow, then add speed. It makes sense and is easily understood. This is referred to as 'Normal Action' and is intuitive.

Some real systems, however, require the opposite action to occur. For example, a pressure that is too low may be increased by closing a valve. This means that a higher error will require a lower output, or a 'reverse action' from the

control system. This may be done by the final control element (such as the control valve) or it may have to be built directly into the PID control system. TEST's system can easily provide reverse action if required by using a negative value for the Proportional Term. This effectively reverses all of the PID calculations causing a higher error to generate a lower output. The I and D terms are specified in minutes, however, and are always positive numbers. Their action will follow that of the P term to provide the reverse action as well.

PID TERM CHANNELS

The starting point for all PID control is the setpoint value. This can be any type of SCADA channel type such as Analog input or Value channel. The PID calculator will simply get the current value of the specified channel on each pass through the calculation and use that value as the desired setpoint. Where the number comes from is not important. The number enters the PID calculation in real units such as PSI or DEG-F.

The next thing normally needed for the PID loop is the input variable. Like the setpoint, this can be any type of SCADA channel. The PID calculator gets the input channel's value and compares it with the current setpoint. The input also has real units and they should be the same as the setpoint channel.

The calculated output of the PID loop is also in the same real units. The output is the *adjustment to the input* that will move it towards the setpoint based on the recent history of the inputs movement. However, real life systems cannot normally add to the process in terms of the input and setpoint variable. If we are monitoring pressure, we cannot simply add PSI. We can open a valve to increase pressure, which results in increased pressure. This is an important point. The input is the measured variable, and the output manipulates the controlled variable. They are not the same thing, but are related by some physical system.

Most mechanical or electronic PID systems operate strictly in terms of 0-100% of some arbitrary scale. For example, a pressure system that ranges from 100 to 500 pounds will have a zero point of 100 and a full scale of 500. The full range spans 400 pounds. But it is convenient to think of the input and setpoint in real terms, not in percent of full scale. The TEST PID system takes care of the conversions for you so that you can always think of your input and setpoint in the units in which they are measured by the SCADA system.

It does this by requiring that all PID loops be scaled with a zero and span value. The PID calculator will take care of any 0-100% type calculations required for various parts of the calculation.

A default option exists for each PID loop that allows some parameters to be specified in terms of percent full scale. When this is selected, the output will be generated as a percent number using the zero and span for the channel. This is the traditional PID controller method where the output signal is thought of as a percent of full scale (4-20ma, 3-15psi). The output value was internally calculated in terms of the input and setpoint units, but converted automatically into percent for convenience. This percent technique also works for some of the other output related parameters to be discussed shortly.

It is not necessary to use all three terms in the PID system be used in each installation. The P term is always needed because it forms the basis of the other

two. A SCADA channel that determines the Proportional factor is always required. In many practical control loops the I and D factors can often be skipped. With this in mind, TEST's PID setup will allow the I and D terms to be ignored by simply blanking out the I and D channel names during the PID channel setup.

Some loops do not even require an input channel. This effectively bypasses the PID calculation but provides a convenient method of providing a scaled analog output from the SCADA system. If no input channel is provided in the PID setup, the PID loop will simply set the output equal to the setpoint. A change in the output can be easily done by changing the value of the setpoint channel. This type of system is covered in a separate section a little later in this document.

The final channel specification is for a steady state offset value that can be used to improve the response of the PID output. This forms a starting point from which the PID logic will start its correction. It is provided to the PID calculation from any channel just like the input and setpoint channels. The steady state channel is optional, and the value defaults to zero if no channel is provided.

The steady state value would be the anticipated value of the output for the specified setpoint. For example, it may be known that a 45% output is needed to maintain the setpoint under normal conditions. By providing the steady state number to the PID logic, it will start at 45% and begin correcting from there. This prevents startup swings that occur if the output starts at 0. The PID loop will correct itself if the steady state value is not provided or if it is within reason. But using the variable will improve the overall action of the PID by providing a built-in output offset that gets the output near the correct value as quickly as possible.

If the PID channel is using the percent method, then the steady state variable is expressed in percent full scale. If not in percent mode, then the steady state value is in the same units as the input and setpoint channels.

INPUT FILTERING

Noise in electronic systems refers to unwanted variations in a signal due to outside influences. Noisy inputs are a problem in any digital control system that takes samples at specific intervals. The noise may come from a number of sources such as interference, process variations, and electronic sampling error. The PID channel has a noise elimination factor that tends to smooth out periodic noise using a standard factoring solution. The noise factor, which is always a real number between 0.0 and 1.0, affects the sampled input according to the following formula:

Filtered Input = [(1-factor)* Input] + [factor * (previous filtered value)]

A filter value of 0.0 effectively eliminates any input filtering. A value of 1.0 will cause the highest amount of filtering.

The proper filter factor is completely dependent on the many variables in each installation. It is suggested that a value of 0.0 be used unless special considerations require otherwise. Keep in mind that the SCADA Function channels can provide input averaging which may be a more suitable means of eliminating periodic variations in an input signal.

PID OUTPUT CONTROLS

There are several internal settings for each PID loop that control the range of the output value. These can be expressed in either percent full scale or in the units of the input and setpoint. Unlike the parameters discussed above, they are not specified as separate SCADA channels. They are setup values entered directly into the PID channel configuration. These values determine the limits on the output signal to prevent huge plus or minus errors from occurring over a period of time. For example, a system that is off line may have an input value that is very far from the desired setpoint. If left unchecked, the integral term will accumulate huge output errors in an attempt to bring the input back in line. When the system is restarted, the input may correct itself quickly but will overshoot because of the accumulated output error. This is called "reset windup", and can be limited by two parameters in the PID channel setup.

Each channel has an output high and low clamp which prevents the output from going above or below the specified value. If the PID calculation determines that the new output would be beyond the specified range, it simply keeps the output value at the limit. For example, it may be reasonable to set a valve position to operate between 20 and 80 percent of full scale. The PID calculation will stop at a low of 20% or a high of 80% regardless of the results of the PID algorithm.

A rate clamp is also provided to prevent drastic changes due to step changes in the input or setpoint variables. The rate clamp is expressed in either percent full scale or in terms of the input variable and represents the largest change that will be accepted, measured in units per minute.

OUTPUT OFFSET

A setup parameter in each PID channel determines if the resulting output requires a 20% offset. Many industrial control systems use physical signals that range from 20% to 100% of a specified scale. Examples are 1-5VDC, 4-20Ma, and 3-15PSI. The intent is to avoid operation in the lower end of the scale where the mechanical or electronic equipment is less accurate.

The PID channel calculates a 'raw' binary number that will eventually be sent to an analog output board by a hardware driver in the SCADA program. The binary value will be converted into an electrical signal that corresponds to a point within an overall output range, such as 1-5VDC. How the binary number is generated is determined by the scale of the channel and the need for an output offset.

If no offset is required, than the resulting raw binary number is a reflection of the PID output in terms of 0-100% of the scale of the channel. A 35% output will result in a raw number that is 35% of the way between 0 and 32,767. The 35% value would go to a hardware board that is expected to produce an electrical signal that is 35% full scale. If the full scale is 1-5VDC, then the desired output would be 35% of the way between 1 and 5 volts, or 2.40 volts.

Some analog output hardware requires that the driving software generate the necessary 20% output offset. On those devices, 0-100% full scale corresponds to 0-5VDC, not 1-5VDC. The initial one volt must be provided by the software. Note that the board may generate 0-20ma or 4-20ma in a similar manner. The important point is whether the hardware provides the offset or not.

The PID channel allows the program to provide the correction needed to work with standard 20% offset channels. This is done by selecting 'Y' on the configuration screen at the "OFFSET OUTPUT?" field. When this option is selected, the program will automatically rescale the output to range between 20 and 100% of the full 16 bit range of the raw output value.

The correct setting for this option is dependent on the actual hardware used in the analog output system. An incorrect configuration will result in either no offset being made or a double offset being made.

SIMPLIFIED ANALOG OUTPUTS

It is not necessary to use the complete PID system in order to get an adjustable analog output. Three alternate methods are available that will allow direct manipulation of the analog output without the use of a normal PID input channel. These methods are:

1. Use an analog input channel to drive the Analog output.

2. Use the PID channel output as its own input.

3. Use no input channel at all.

The first method will bypass the PID system completely. A PID channel is not even used. The analog output is configured to simply copy whatever is coming in on a specified analog input point. The raw value of the analog input is copied to the raw analog output without any conversion or manipulation. Thus may be sufficient in cases where the analog input and hardware scale match the analog output exactly. This method requires that the particular analog output board be configured to map specific analog inputs into the desired analog outputs.

If there is no analog input channel available, a simplified PID can be set up that does not use a normal PID input to drive the output. The most common method will be to use the PID channel as its own input. In effect, the output of the PID is looped back into itself. This provides the functions of an intelligent analog output that will continuously adjust itself to match the value of the setpoint channel. The normal PID constraints will apply such that a change in the setpoint will cause a PID type response to the output. The output will always eventually match the setpoint, but it will move gradually according to the values in the PID terms. This prevents step changes in the setpoint from causing radical changes in the output.

When using the PID channel in this way, it is necessary to set up the scaling and steady state channels to cause the desired result. It is also necessary to assign the same channel to both the setpoint and the steady state output. With this setup, the PID output will constantly seek to reduce the error between itself and the setpoint. It will eventually attain the steady state value specified by the setpoint when the error is zero. The best way to do this is to use a scale of 0-100% for the PID range. Other arrangements can be done, but will require proper adjustment of the various parameters to insure that the PID loop will cancel itself out.

An even easier method of obtaining a simple analog output is to assign no input channel at all. This special configuration is recognized by the PID driver. It

bypasses the normal PID calculation and simply outputs the necessary value to match the setpoint. No PID smoothing is done. The only manipulations are related to output scaling. This is helpful in passing an analog input to an output when the input and output are not in the same scale range. In effect, the PID channel simply rescales the input into the output without any of the normal PID considerations.

PID CALCULATION TIMING

The PID calculation is a processor intensive process similar to the AGA3 meter calculation. The effects on the computer's performance will depend greatly on the existing load on the computer and is difficult to predict. The PID channel has a setting which determines how often the PID output is updated. It is specified in seconds, and is designed to allow staggering of PID calculations in low power systems having multiple PID loops. Faster systems (286 and 386) can normally use the default value of 1 so that the PID is updated each second. Heavily loaded systems can use other numbers such that each channel will be eligible for calculation at a different time. For example one channel may be set to 3 and another to 4. In this case, they would both require calculation at the same time every 12 seconds. The other 11 seconds will see either none or a single channel calculation requirement. This spreads the load at the expense of a slower response on the PID output.

The PID speed of response is not a problem in almost all applications, so use the update setting to reduce wasted processor time. Considering that many large DCS and process control systems calculate PID loops every minute, the three or four second wait in the TEST PC based system is hardly a problem.

PID LOOP TUNING

'Tuning the loop' refers to making the necessary adjustments in the PID terms to provide optimal response in the control system. The technical details of this process are impossible to determine in a general way because each installation has its own set of physical constraints. Entire volumes have been written on the tuning techniques for particular types of equipment, so a few tips are all that can be given here. A simple method suitable for most industrial system is as follows:

1. Set the I and D terms to 0 for the initial testing. Set the P term to a value between 0.1 and 1.0 to test the response of the system.

2. Change the setpoint by 20% and monitor the reaction of the system. If it over reacts, lower the P term. If it is to slow, increase the P term.

3. Duplicate step 2 for a low, medium, and high setpoint. Record the final output value for each setpoint. Determine how much variation in the P term is needed to get a suitable response at each setpoint.

4. Begin adding some I term, starting at a value of 0.05. Repeat steps 2 and make adjustments to the I term this time.

5. If step changes in the load are anticipated, experiment with very small D terms to allow anticipation by the control system. Normally, very small values are all that is needed in industrial systems. Values from 0.05 to 0.25 are common.

6. Test the final configuration at various setpoints. Keep in mind that it may be desirable to manipulate the P,I, and D values for different setpoints or operating conditions. Because these numbers are contained in standard SCADA channels, they can be easily modified locally or remotely just like any other channel.

MANUAL OUTPUT CONTROL

Normally the PID output is calculated once every so many seconds as specified in the setup for each channel. Any attempt to set the PID output value directly (by a CALC or DATA statement) will not work. This is because the value stuffed into the channel will be overwritten as soon as the PID is calculated again. In order to directly control the output, the PID channel calculation must be temporarily disabled. This is done with the TSP HOLD command. A channel on hold retains all of its normal capabilities except that it no longer does any internal conversions. In the case of the PID, the calculation is suspended until the channel is reactivated with an UNHOLD command.

This is analogous to 'manual mode' on a stand-alone PID controller. When the PID loop is on Hold, any normal channel set method can be used to manipulate the PID output. For example, a simple `CALC P1=40` line could be used to set PID channel number 1 to 40. The value of 40 would mean 40% if the channel is in percent mode. Otherwise, it would mean 40 in terms of the units defined for that channel.

A simple program can be prepared to allow manual manipulation of any PID channel. The example below allows local keyboard control of any PID output with the keypad 0 and 9 keys. A direct setting can also be done by hitting the plus key and then providing the desired output value.

```
; Program SETPID.  Manually adjusts PID output directly with
; local keyboard control.
cls
local x,y,z
msg MANUAL PID OUTPUT CONTROL
cursor 1,5
input, Enter PID (as in P1) ,%1
calc y = 5  ; Assume 5% change per bump
input, Enter value for adjustment,y
cursor 1,24
msg Use 1 for increase and 0 for decrease.  ESC to quit
hold $%1
cursor off

:LOOP
cursor 70,1,$T
cursor 1,10 ; put up the PID text line
calc $%1=

x = @key(0)
if x =0
  goto loop
endif
```

```
; A key was pressed
cursor 1,14
if x =57    ; ASCII 9    Raise output
  calc $%1 = $%1 + Y
  goto loop
endif

if x =48    ; ASCII 0     Lower output
  calc $%1 = $%1 - Y
  goto loop
endif

if x =43    ; ASCII +     Manual Setting
  cursor on
  gotoxy 1,22
  input,Enter PID Output, z
  calc $%1 = z
  cursor 1,22
  cursor on
  goto loop
endif

if x =27    ; ASCII ESC
  cls
  cursor 1,10
  msg PID manual control of $%1 is over.
  msg Return the channel operation with the UNHOLD $%1 command
  cursor 1,24
  cursor on
  return  ; get out of here
endif

goto loop
; --------------  End of Program
```

This TSP procedure is an example of how the output could be manually controlled. Note that the channel is placed on hold by the procedure prior to the manual manipulation. In this example, the channel remains on hold at the end of the program. These types of features are completely up to the user.

When a frozen PID channel is returned to normal with the UNHOLD command, a *"Bumpless Transfer"* will occur when the PID calculation resumes. The output will not jump to a new value because the actual PID output had been manipulated by the manual control. Some electronic controllers cannot do this because the manual output is separate from the automatic one. The TEST PID system provided the bumpless transfer automatically when the UNHOLD command returns the PID channel to its normal state.

> *WARNING: The PID control system contained in the TEST SCADA system relies on digital computer hardware and techniques. Like any other computer based system, the potential exists for hardware and software failures beyond the control of the program. The PID control system and output hardware are not intended for use as a primary safety system. Other means of protecting life and equipment must be*

provided externally to the computer system in order to prevent injury to personnel or damage to equipment affected by this system. If necessary, external analog backup or bypass systems should be used to provide uninterrupted analog outputs to critical equipment that will be adversely affected by the loss of reliable output from the SCADA system

References:

Anderson, Norman A. *Instrumentation for Process Measurement and Control*, Chilton Books, 1972.

Deshpande, Pradeep B., and Ash, Raymond H., *Elements of Computer Process Control*, Instrument Society of America, 1981

Mellichamp, D.A., *Real-Time Computing with Applications to Data Acquisition and Control*, Van Nostrand, 1983