What is I in PID? ... a (too) simple answer: it is the “Integral” component of Proportional-Integral-Derivative control ...

foreword by Ron Beaufort: I have previously posted much of the material contained in this paper on the “Live Question and Answer” forum at www.PLCs.net ... this was just one in a series of related threads which discussed the basic characteristics of the three components of PID control ... anyone who is interested in following the full context of the questions and answers of the original threads should be able to find them at the following links:

What Is P in PID? http://www.plctalk.net/qanda/showthread.php?t=13459
What Is I in PID? http://www.plctalk.net/qanda/showthread.php?t=11242
What Is D in PID? http://www.plctalk.net/qanda/showthread.php?t=11969
Proportional-only Control – Why Won’t It Work?
http://www.plctalk.net/qanda/showthread.php?t=14497

since my original postings, many readers have requested that I make the text and illustrations easier to access by providing a single continuous PDF file of the material ... this document is the result of those requests ... I’ve reformatted some parts of the text and added more detail in certain places to make the ideas easier to follow ... most of the forum’s original conversational tone has been left in place ...

while I believe that the information included here is correct, it is offered free of charge with no warranty of any kind ... it may be copied and distributed as long as there is no commercial interest involved and as long as the material is copied and transferred in its entirety - including this notice and all notices of copyright ...

I hope that this material proves helpful ... corrections, comments, and suggestions from readers are always welcome ...

well, you’ve got to start somewhere ... so this paper assumes (rightly or wrongly) that the reader is already familiar with the very basic idea that PID (Proportional-Integral-Derivative) is a mathematical function (an equation) which may be used to automatically control systems such as temperatures, flow rates, speeds, pressures, levels, etc. ... the subject of this paper is the Integral component of PID control as used in many different types of Allen-Bradley PLCs (Programmable Logic Controllers) ... please keep in mind while reading this that there are exceptions to every rule ... this material is intended to be a beginner’s level introduction to a very complex subject ... in order to keep things as straightforward as possible, it has been necessary to leave out quite a few topics which I wish I had time and space to cover ... but this is at least a starting point at answering the basic question:

What is I in PID?

for many students the biggest problem associated with learning how the PID’s Integral term functions is the difficulty of understanding the “units” which are used to measure its action ... ... I’ve had some pretty good success using the following explanation in the hands-on “technician level” PID classes that I teach ... most of the students in those classes need to work around PID-controlled systems, but they don’t really have very strong math skills ... anyway ... let’s see if this approach helps you too ...

first things first ... there are MANY different ways of setting up PID control ... but so that we’ll have a specific brand and model of controller for our examples, I’ll be using the Allen-Bradley PID for this discussion ... specifically, I’ll be using the “Dependent Gains” equation (or as it’s also known) the “ISA” equation of a PLC-5 processor ...

note: in addition to the most-common “Dependent Gains/ISA” equation, the PLC-5 and ControlLogix platforms also support the “Independent Gains” or (as it’s also known) the “AB” equation ... we won’t go there for this initial discussion ... if you’re using the SLC or MicroLogix platforms, then the “Dependent Gains/ISA” is the only option available ... where PID control is concerned, the biggest differences between the various Allen-Bradley platforms are in the ranges of the analog input and output signals used by each controller ... where possible, I’ll use “percentages” for these units rather than the raw data values ... students using the SLC-500 and MicroLogix controller platforms should be able to follow the basic ideas of this text by mathematically converting between the different data ranges involved ...

first let’s start with a simple temperature control system ... now the natural thing to do would be to use “degrees” as the units for our process signal ... but to keep things as simple as possible, let’s use “PERCENT of FULL SCALE” instead ...

let’s say that the controller’s SP (Setpoint – or “target”) has been set for something like 10% of full scale for a very long time ...

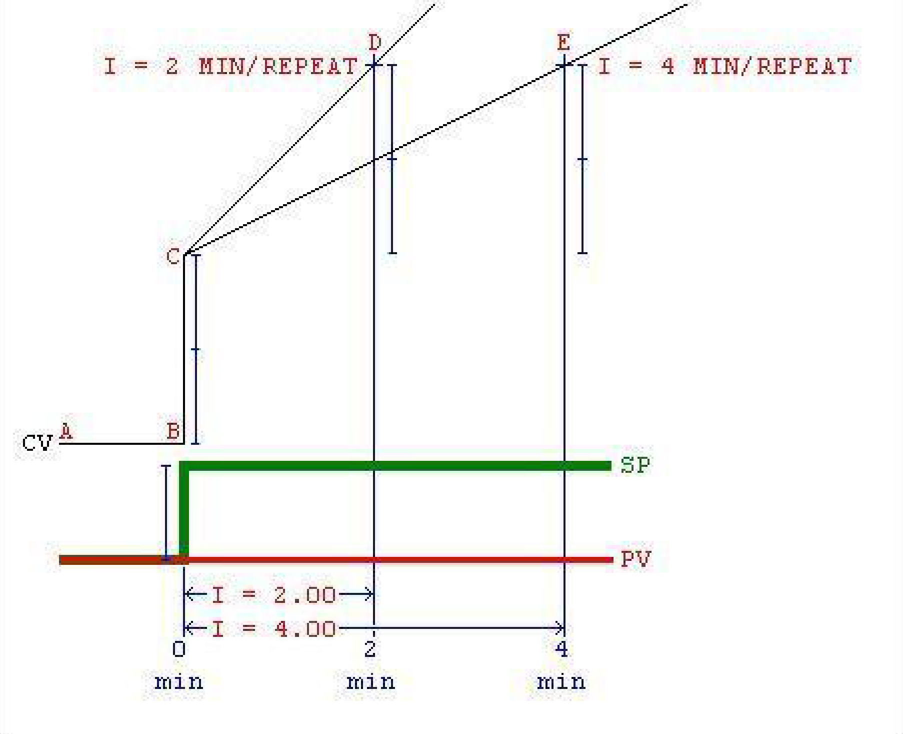
let’s say that the controller has been regulating the CV (Control Variable – or controller “output”) at 25% for a very long time ...

let’s say that the system’s PV (Process Variable – or “temperature”) has been accurately controlled at the desired target of 10% for a very long time ...

you should already understand that every time the PID controller is executed, it calculates a value for E (Error – or “how far are we away from the target?”) ... in our example, the formula for this error calculation is “E=SP-PV” ... since E=10-10 then E=0% of full scale ... this “0” error value is the PID controller’s way of knowing that WHATEVER the value going out to the CV, it’s PERFECT ... in other words, we’re presently right on target ... the PV is tracking the SP ... so let’s keep everything right where we’ve got it ... “steady as she goes” ...

and those are the conditions that we see at the left side of Figure 1 as shown below ... so just to nail things down, we have a well-controlled system simmering along right on target ... the sun is shining ... the birds are singing ... life is lovely ...

FIGURE 1 - INTEGRAL UNITS - MINUTES PER REPEAT



now ... let’s do something really strange ... we’re going to do this just for EDUCATIONAL purposes only ... we’re going to physically JAM the system’s output valve (the real-world field device) in its present (25%) position ... now at first glance you might think that this will really “mess things up” ... but it won’t ... in fact, as long as everything else (room temperature, gas pressure, etc.) stays the same , the system will just keep on simmering along ... right on target ... think about it this way: we’ve already said that the CV has been running along at 25% for a very long time ... and even though we’ve physically jammed the valve at the 25% position, the system should stay right on target – just as long as nothing else changes ... and now the stage is set ...

first we’ll begin with the “P” or “Proportional” setting ... we’ll get to the “I” or “Integral” setting in due course ...

the CV starts out at position “A” on the graph ... that’s the 25% setting ... as soon as we reach position “B” on the graph, we suddenly increase the SP (the “target”) from its original setting of 10% to a new setting of 20% ... the PID controller sees this change ... and it calculates a new Error based on this equation: “E=SP-PV” ... since E=20-10 then E=10% of full scale ...

now let’s say that our controller’s “P” or “Proportional” setting is currently a nice round value of 2.00 ... the controller takes the E (10%) and multiplies that by the P setting (2.00) and then adds the result (20%) to the previous CV (25%) ... the result of this calculation (45%) becomes the new value for the CV ... and that’s what we see happening at position

“C” on the graph ... specifically, the controller’s proportional action has instantly stepped the CV from 25% to 45% in an effort to drive the PV up to the new target value ...

now notice the little blue “yardsticks” on the graph ... notice that one of these “yardsticks” is used to measure the sudden increase in the SP ... this “yardstick” is 10% of full scale ...

now notice that it takes two “yardsticks” to measure the 20% jump in the CV between position “B” and position “C” ... just for discussion, if our controller’s P setting happened to be 1.00 instead of 2.00, then the increase in the CV from point “B” would only have been half as much ... specifically, it would have gone up one 10% “yardstick” rather than two ... since the PID’s proportional action is discussed in another paper, we won’t go into any more detail about it here ... and now we’re ready to tackle the “I” or “Integral” setting ...

now remember that the system’s control valve has been physically jammed in the 25% position ... it cannot possibly move ... so even though the PID controller really and truly sends out a new value of 45% to the output valve, the valve (the field device) stays in the 25% position ... that means that the PV is going to keep right on tracking along at its 10% value ... you can see that happening on the graph ... specifically, the PV line stays completely horizontal ... even though the CV (the value in the PLC’s memory) does indeed go up ...

so now the Proportional action has done as much as it can to drive the PV toward the target ... but ... since we’ve jammed the valve, the PV doesn’t change ... so the PID starts using the Integral action to try to increase the PV ... and that’s the “ramping” action that you see taking place in the CV after its initial jump up to position “C” ...

so now let’s look at the CV as it takes off from position “C” ... for discussion purposes, I’ve given you two possible paths for the CV ... let’s take the first one which leads up to position “D” ... this is the path that the CV would take if the controller’s “I” setting was 2.00 ... that setting would literally mean “2.00 minutes per repeat” ... in other words, it would take 2.00 minutes for the Integral action to REPEAT the Proportional action ... and that is what the two “yardsticks” are showing at position “D” ...

now let’s look at the CV again as it takes off from position “C” ... but this time we’ll take the second path ... the one which leads up to position “E” ... this is the path that the CV would take if the controller’s “I” setting was 4.00 ... that setting would literally mean “4.00 minutes per repeat” ... in other words, with this setting it would take 4.00 minutes for the Integral action to REPEAT the Proportional action ... and that is what the two “yardsticks” are showing at position “E” ... specifically, the Integral action caused the CV to increase the same 20% (two “yardsticks”) ... BUT ... notice that it took twice as long (4 minutes rather than 2 minutes) to make the SAME amount of increase ...

so ... one thing that we’ve just learned is that for LESS Integral ACTION, we need to use a HIGHER Integral SETTING ... specifically, a BIGGER NUMBER gives us LESS ACTION ...

conversely ...

for MORE Integral ACTION, we need to use a LOWER Integral SETTING ... specifically, a SMALLER NUMBER gives us MORE ACTION ...

believe me this is NOT a trivial matter ... I've seen inexperienced technicians fight a PID tuning problem for hours while changing the Integral setting in the WRONG direction ... just to nail it down: I refuse to let my students say "increase the Integral" or "decrease the Integral" ... because those statements are ambiguous ... do they mean "increase the Integral ACTION" or "increase the Integral SETTING"? ... taken the wrong way, the effect of their statement would be exactly the REVERSE of what they actually meant to do ...

important! ... please remember that we're basing this discussion on an Allen-Bradley controller using the "Dependent Gains/ISA" equation ... the "bigger number means less action" and "smaller number means more action" rule that we just nailed down is BACKWARDS if you use the "Independent Gains/AB" type of equation ... further, the Integral units in that equation are NOT based on the same "repeats per minute" idea that we're covering in our example ...

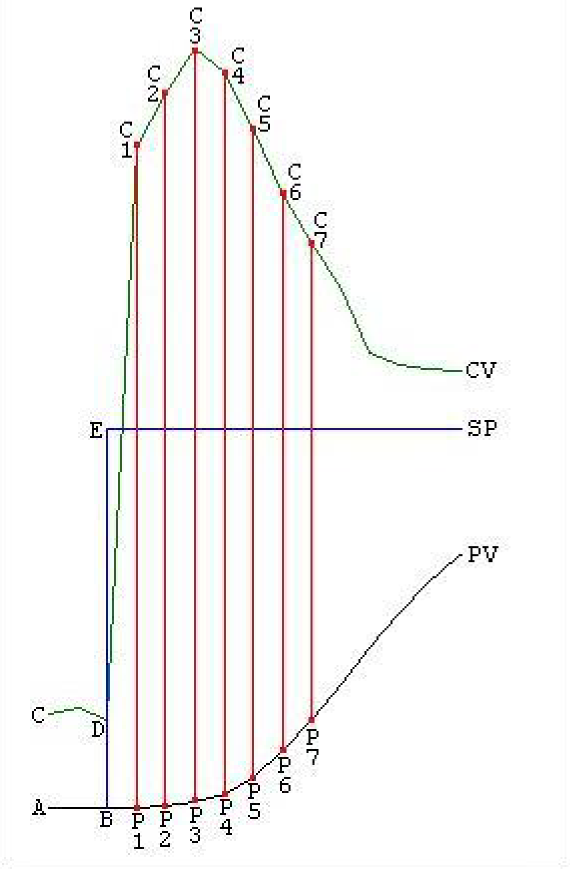
important exception to the "smaller-NUMBER-means-more-ACTION" rule: entering a value of "0.00" actually turns the Integral action OFF ...

summing up ... in an Allen-Bradley system using the common "Dependent Gains/ISA" equation ... the values of the Integral term are "minutes per repeat" ... in other words, the length of time that it takes for the Integral action to REPEAT the Proportional action ...

next we'll have a quick review of how the PID is executed in a "step-by-step" manner based on time ... in most cases the "time" issue is taken care of by programming a timer's Done bit which is then used to "trigger" the PID into executing ... for the purposes of discussion, I usually call each execution of the PID a "tick" ... the red lines in the figure below are used to show how the PID's action can be broken down into a series of these regularly-spaced step-by-step "ticks" or intervals ...

Figure 2 shows a common type of "trend" (a graph) of a process being controlled by a PID ... the three signals that we're most interested in right now are the black trace (the PV - Process Variable – or "input" signal) ... and the blue trace (the SP – Setpoint – or "target" value) ... and the green trace (the CV – Control Variable – or "output" signal) ... each of these are labeled at the right end of their traces ... the red lines on the trend are there just to help us discuss how the operation of the PID takes place in a "step-by-step" fashion just like the "tick-tick-tick" of a clock ... each "tick" is used to "trigger" the PID into executing at an evenly-spaced interval of time ... another perfectly valid way to think about this "step-by-step" execution is in terms of the PID's "update" time ... but that particular terminology can sometimes lead to miscommunication ... let's just stay with "ticks" for this discussion ...

FIGURE 2 - STEP BY STEP



let’s begin discussing Figure 2 by looking at what is happening between point A and point B ... between point A and point B the PV (black) is tracking along right on top of the SP (blue) ... in other words, here the PID is giving perfect control of the process ... and if we take a look at the CV (green) during this particular period, we’ll see that it’s happily simmering right along from point C to point D ... the sun is shining ... the birds are singing ... life is lovely ...

and then, just as the PV is reaching point B, the operator decides to crank the SP up a notch ... you can see the SP move instantly from point B right straight up to point E ... now let’s take a careful step-by-step look at what happens next ...

time is marching on ... and eventually we reach the time of the PID’s next “tick” ... and now when the PID is triggered into execution, it sees the PV (its input) just as it reaches point P1 ... notice that the PV is still simmering along from point B to point P1 without ANY change in its value ...

when triggered into operation at point P1, the PID faithfully does its math calculations and finds out that the PV is now suddenly quite a bit “off target” from the SP ... and based on the present value of the SP and the present value of the PV, the PID calculates a new value for the CV ... and based on that new value, the CV is increased from point D right up to point C1 ...

and time is still marching on ... eventually we reach the time of the PID’s next “tick” just as the PV reaches point P2 ... and if we look very closely, this time we might notice just a very slight increase in the value of the PV ... this rise in the PV is due to the recent increase in the CV ... that increase in the CV (output) is forcing the PV (input) to move toward the desired SP (target) ...

and now when the PID is triggered into operation at point P2, it does its math calculations again and comes up with another new value for the CV ... and based on that new value, the CV is increased again ... this time from point C1 up to point C2 ...

and time is still marching on ... eventually the PID “ticks” again just as the PV reaches point P3 ... and this time we notice a little bit more increase in the value of the PV ... the increased CV is still forcing the PV to keep moving toward the SP ...

and when the PID is triggered into operation at point P3, it comes up with another new value for the CV ... and the CV is increased again ... this time from point C2 up to point C3 ...

and just once more for good measure ... time is still marching on ... eventually the PID “ticks” again just as the PV reaches point P4 ... and this time we notice even more of an increase in the value of the PV ... the increased CV is still forcing the PV to keep moving toward the SP and by now the PV has finally started making a substantial amount of change from one tick to the next ...

and as before, when the PID is triggered into operation at point P4, it comes up with another new value for the CV ... BUT ... this time the CV is not increased ... instead it is decreased in value from point C3 down to point C4 ...

even though this discussion focuses on the Integral action, for completeness I’ll just mention that the sudden jump in the CV from point D to point C1 was caused almost totally by the PID’s Proportional action ... the Proportional action (as discussed in another paper) reacts only to the “amount of error” (how far the PV happens to be from the SP) ... this “amount of error” suddenly increased when the operator kicked the SP straight up from point B to point E ... and once the Proportional action had “done its thing”, then the Integral action was responsible for the slow but steady increase of the CV from point C1 to point C2 and on to point C3 ... the Integral action (as we’re about to discuss) reacts to the “amount of error over time” ... notice that after its original jump, the “error” (the distance of the PV from the SP) did not increase again during the period of time between point B and point P3 (actually the “error” decreased during this period as the PV slightly approached the SP) ... but (as we’ll soon see) the Integral action monitors not only the “amount-of-error” but also the “amount-of-time-that-the-error-has-existed” ... and that’s how the Integral action contributed the slow steady increase of the CV from point C1 - to point C2 - to point C3 ...

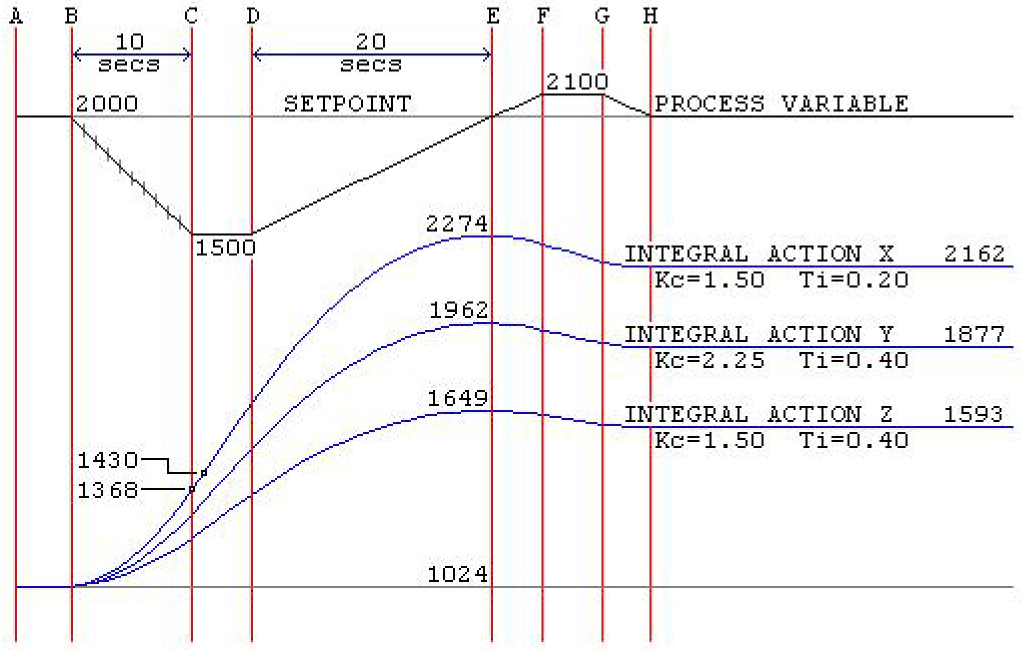
possibly the most important thing involved in the little “review” that we’ve just had is the basic idea of how the PID executes ONLY at these little “tick – tick – tick” points in time ... specifically, when we (as humans) look at a trend graph, we invariably focus on the shapes and patterns of the little squiggly lines ... the PID doesn’t look at the process in the same way ... the PID bases everything that it knows, and everything that it does, on these quick little “snapshots” of the data that it sees whenever it is “triggered” into executing ... once the technician comes to grips with this reality, it usually becomes a lot easier to understand “what-the-PID-is-doing” and “why-the-PID-is-doing-it” and “how-to-fix-it” when the PID doesn’t work to suit us ...

next we’ll focus on the basics of how the Integral action responds to various types of inputs ... as we work through this please remember that these examples are based on a certain specific brand of PLC (Allen-Bradley) ... and a certain specific platform (PLC-5) ... and a certain specific equation (ISA – Dependent Gains) ... and a certain specific signal for our Integral action (the PV) ... etc. ... the main point is that different systems use different methods of handling the Integral action ... what follows is just one (but a very common one) of those many different methods ...

the top trace (black) in Figure 3 shows an artificial PV (input) signal that we’ll use for discussion purposes ... the basic idea is that if we feed this exact same input signal into our PID over and over again, then we can use slightly different adjustments for our PID tuning values each time ... we can then compare “apples-to-apples” by examining any differences in the PID’s output from one experiment to the next ... just keep in mind that the “profile” of the PV signal is NOT responding to the

PID’s output ... specifically, this input is an artificially generated “test” signal that always “does the same thing” regardless of how the PID tries to control it ...

FIGURE 3 – INTEGRAL'S RESPONSE TO ERROR



the test starts out at point A with the PV perfectly tracking along the SP line ... specifically, between point A and point B, the PID is giving us perfect control of the process ... the SP is 2000 ... the PV is 2000 ... the sun is shining ... the birds are singing ... life is lovely ...

note: the PID system in our example uses numbers which range in value from 0 to 4095 to represent the signals we’re discussing ... to nail this down, the value 2000 in this example represents a signal which is about 48.8% of full scale ... (2000 divided by 4095 equals 0.488) ... if (for one example) our process happened to be using a temperature transmitter scaled from 0 to 500 degrees F. then our 2000 count signal would correspond to a temperature of about 244 degrees (500 times 0.488 equals 244) ... for the time being, don’t worry too much about the numeric values shown on the figure ... we’ll get into the math later ... let’s just look at the patterns of the squiggly lines for now ...

back to our story ... now at point B something happens to the system which sends the PV signal downward away from the SP ... maybe someone shoved another load of cookies into the oven ... but regardless of whatever caused the disturbance, the next time the PID “ticks” (is “triggered” into execution) it will see an input signal which is no longer “right on target” ... in other words, the PLC will see the deviation of the PV from the SP as an “error” ... and that “error” will cause the PID to make a change in the CV ... now you should already realize that any change in the CV will be the result of a change in the Proportional action – and/or in the Integral action – and/or in the Derivative action ... but the only one of these three actions that we’re interested in for this discussion will be the Integral ...

and now let’s talk about those three examples of Integral action shown in Figure 3 ... let’s start out with the first one (the one labeled “Integral Action X”) ... notice that as we moved along from point A to point B, the Integral action was steady at 1024 ... specifically, the PID’s Integral action was contributing 25% to the CV signal ... and it was this 25% CV signal which was keeping the PV “on target” (SP=2000 and PV=2000) from point A to point B ...

how about the Proportional’s action? ... well the Proportional term only makes a contribution to the CV whenever we have an Error (a deviation from the target) ... from point A to point B we were right on target and so there was no Error ... so the Proportional action had nothing to do with the original 25% CV output ... well how about the Derivative’s action? ... well the Derivative term only makes a contribution to the CV whenever we have a change in the PV ... from point A to point B there was no change in the PV ... it started at 2000 and it stayed at 2000 ... so the Derivative action had nothing to do with the original 25% CV output ... nope ... it was the Integral and the Integral alone who was giving us that initial 25% CV to keep us on target from point A to point B ...

but now as soon as we reached point B, things changed ... specifically the PV changed ... let’s see what the Integral action does in response to that change ...

simply put: the Integral reacts to “error over time” ... it looks not only at how LARGE the error is, but also at how LONG the error has existed ... and it will do whatever it can do in order to make the error go away ...

important note: keep in mind that the three signals shown at the bottom of this figure are NOT the CV (the total output of the PID) ... instead they represent ONLY the Integral action which I have mathematically extracted from the CV for the purposes of this discussion ...

so let’s take a detailed look at what happens just as the PV leaves point B ... the Integral notices the error ... but it also notices that the error is very small ... and it also notices that the error has only existed for a very short time ... based on all of this, the Integral calculates a slight increase to the CV ... and you’ll notice that the curve of Integral Action X makes a small (just a little – just a little) change upward from its original 1024 value as it leaves point B ...

and so in very basic terms, what we’ve seen so far is that as long as the PV was tracking along “on target” then the Integral maintained its present 25% output to the CV ... BUT ... just as soon as the PV started going downward in value, the Integral noticed the error and began making a greater contribution to the CV ... this greater contribution serves to increase the CV in an effort to overcome the PV’s downward path ...

but ... notice that the Integral doesn’t get “carried away” when it calculates this increase for the CV ... it takes more of a “well, let’s crank it up just a little bit and then keep an eye on it for awhile” approach ... and that’s why the Integral term gives us that nice smooth sweeping curve effect ...

now let’s follow the PV signal downward from point B to point C ... notice that this is a straight-line downward slope ... but the response of Integral Action X does NOT follow a straight line during this part of the graph ... the Integral is still keeping track of not only how large an error we have, but also how long the error has existed ... that’s what causes the smooth sweeping (but constantly increasing) response from the

Integral term ... by the time we reach point C the Integral has started to “get serious” about correcting the error ...

from point C to point D the PV signal levels off ... so does that satisfy the Integral and cause it to decrease its contribution? ... no, not at all ... the Integral is still not happy ... notice that even though the error is not getting any larger between point C and point D, the Integral is still increasing its contribution ... it’s still trying to turn the wayward PV around and push it back toward the target ...

it’s important that you notice this: from point C to point D, the error is not getting any larger ... BUT ... the Integral is still constantly INCREASING its contribution to the CV ... it’s as though each time the PID is executed, the Integral thinks to itself: “it looks like that error is not getting any better so I’d better try an even bigger change to fix it” ...

and now let’s see what happens next ... from point D to point E the PV starts rising ... finally ... we’re heading back toward the setpoint ... this should make the Integral drop off again – right? ... nope ... in fact the Integral is STILL INCREASING its contribution ... why? ... because we’ve STILL got an error hanging around ... and as long as we have an error, the Integral will keep changing its contribution to the CV in an effort to make the error go away ... and as we finally get to point E, the PV reaches the setpoint and the error has finally “gone away” ... and look at what happens to the Integral’s contribution ... it STILL doesn’t start to decrease ... instead it levels off and holds its current value (2274 in the case of our Integral Action X example) ...

from point E to point F the PV keeps on rising ... and so we’ve got a little bit of “overshoot” to worry about ... notice that the Integral finally starts to “back off” a little bit on its output contribution ... that’s because now the error is on the other side of the setpoint ... the Integral responds to this “oops! too high” condition by decreasing its output ... it’s trying to bring the PV back toward the target ...

from point F to point G the PV levels out ... but even though the error isn’t getting any larger, we’re still “off target” and so the Integral keeps on making adjustments to its output contribution ...
from point G to point H the error is finally becoming smaller and smaller ... the Integral notices that pattern and starts to “level off” on its output contribution ... at point H the error has finally “gone away” and so the Integral decides to “hold what we’ve got” and keeps a “steady as she goes” output for the rest of the graph ...

here in a nutshell is the Integral action’s narrow-minded little view of the world:

(1) do we have an error? ... I don’t like that ... I’ll change my contribution to the CV in order to make the error go away ...

(2) do we STILL have an error? ... I don’t like that ... I’ll make an even bigger change to my contribution in order to make the error go away ...

(3) is the error finally gone? ... excellent ... now I’ll just hold my current setting and keep it “steady as she goes” ...

so far we’ve seen step-by-step how the Integral reacts to “no error” ... and to an error which is increasing in size ... and to an error which is decreasing in size ... and to an error which is staying the same ...

now let’s take a look at the three “Integral Action” traces X, Y and Z in Figure 3 to see how changes to the PID tuning parameters affect the operation of the Integral term ... notice that the basic shapes of all three traces are the same ... and they all start out at the same 1024 (25%) value ... the only differences between them are the heights reached by each one as the graph runs along ...

first let’s compare trace X to trace Z ... notice that the Proportional settings for each of these two traces are identical ... specifically “Kc” (controller gain) is 1.50 for both trace X and for trace Z ... but the Integral settings are different ... specifically, “Ti”(time integral) for trace X is 0.20 ... “Ti” for trace Z is 0.40 ... we learned at the beginning of this paper that the units for the Integral term are expressed in “minutes per repeat” ... we also learned that a SMALLER value for “Ti” will result in a LARGER output response from the Integral term ... and that’s why even though both traces start out at the same initial value (1024), trace X is able to reach a higher maximum value (2274) compared to the maximum value (1649) which was reached by trace Z ... also the “smaller-equals-stronger” Integral setting (0.20 minutes per repeat) of trace X resulted in a higher final value (2162) than the final value of trace Z (1593) with its “larger-equals-weaker” Integral setting ...

next we’ll compare trace Y to trace Z ... notice that the Integral settings for each of these two traces are identical ... specifically, “Ti”(time integral) for trace Y is 0.40 ... “Ti” for trace Z is also 0.40 ... since both Integral settings are the same, we might naturally expect identical results from each trace ... but obviously that is not the case ... the answer lies in the difference between the “Kc” settings ... the fact is that the PID’s Proportional setting (Kc) affects the final results of the Integral term ... specifically, once the Integral term has calculated its contribution for the CV, the PID multiplies the results of that calculation by the Proportional setting ... the outcome of that multiplication gives the final results of the Integral action ...

secret handshake: one of the names for the type of PID equation that we’re using in this discussion is “Dependent Gains” ... that name literally refers to the fact that the final result of the Integral contribution mathematically “depends” on the value of the Proportional setting ... from a technician’s “nuts-and-bolts” viewpoint this means that as he cranks the PID’s “Kc” setting up and down to adjust the Proportional response, he is also affecting the Integral response at the same time ... along exactly the same lines, the Derivative response works in precisely the same way ... specifically, increasing the Proportional setting increases not only the Proportional action but also increases the Integral action and the Derivative action too ... conversely, decreasing the Proportional setting decreases not only the Proportional action but also decreases the Integral action and the Derivative action at the same time ...

now one more exercise for the previous figure and then we’ll move on ... I’ve included just one example of the math functions involved in the Integral’s calculations ... I’ve shown two values on trace X near point C ... the first value (1368) was the result of the PID’s Integral calculation at point C ... the next value (1430) was the result of the calculation on the very execution of the PID which occurred one second later ...

although you do NOT need to do any of these calculations yourself in order to work with PID control, some students find that working through one or two problems like this helps them to understand the concepts involved ... you may skip the math if you want to – but most people are amazed that working through it requires nothing more complicated than the same common “add, subtract, multiply, and divide” type math that we use in our everyday lives ...

behold the formula that the PID uses to calculate its Integral contribution:

I.cont.now = (Kc * (((1/Ti) * E.now) * LUT.min)) + I.cont.last

I.cont.now = Integral contribution now (being calculated on this execution of the PID) ...
Kc = PID Proportional setting ...
Ti = PID Integral setting ...
E.now = Error now (on this execution of the PID) ...
LUT.min = PID Loop Update Time setting (expressed in minutes) ...
I.cont.last = Integral contribution calculated on last execution of the PID ...

plugging in the numbers for our example we have:
Kc = 1.50 ...
Ti = 0.20 ...
E.now = 500 (E=SP-PV and so E=2000-1500) ...
LUT.min = be careful with this one ... the PID Loop Update Time setting is entered on the PID setup screen expressed in SECONDS ... our example uses a setting of 1 second ... to convert this to minutes: LUT.min=LUT.sec/60 ... so LUT.min=1/60 or about 0.0167 ...
I.cont.last = 1368 for the step in this example ...

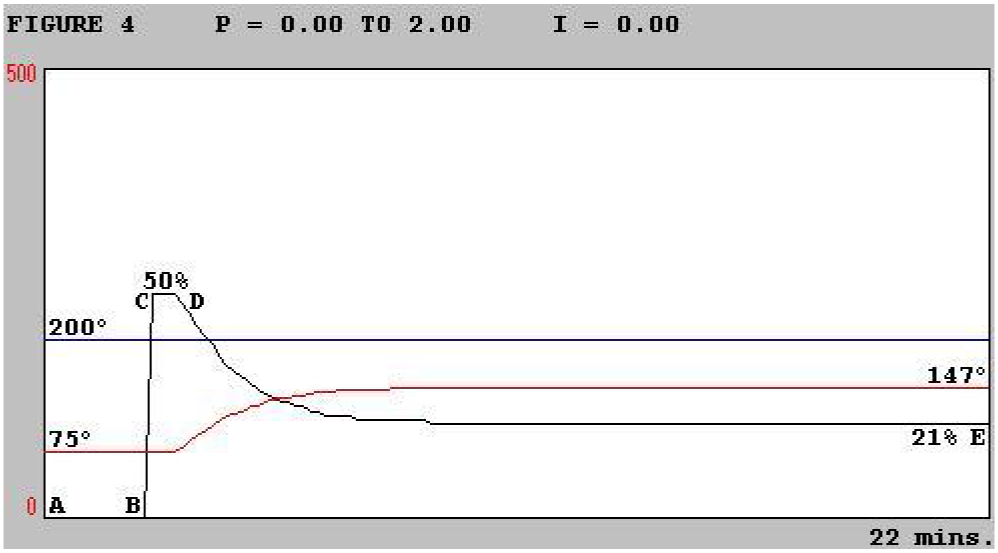
and substituting in the formula gives:
I.cont.now = (1.50*(((1/0.20)*500)*0.0167))+1368 = 1430.625 ...

which when allowing for “round off” comes out within one point of our PLC’s calculated value of 1430 for its PID Integral contribution ...

the most important thing to take away from this particular exercise is the fact that the PLC does NOT use any “elegant math” functions (such as those used in true calculus) when calculating the PID’s Integral contribution ... specifically, any fourth-grade student who’s “pretty good” at math should be able to work through any and all of the PID’s Integral calculations once the formulas are all laid out ... and incidentally, the same thing can be said for the Proportional and the Derivative actions too ... so why do the user manuals and textbooks use all of those squiggly “integration” signs and “derivative” symbols when discussing the PID formulas? ... your guess is as good as mine ... now on to other things ...

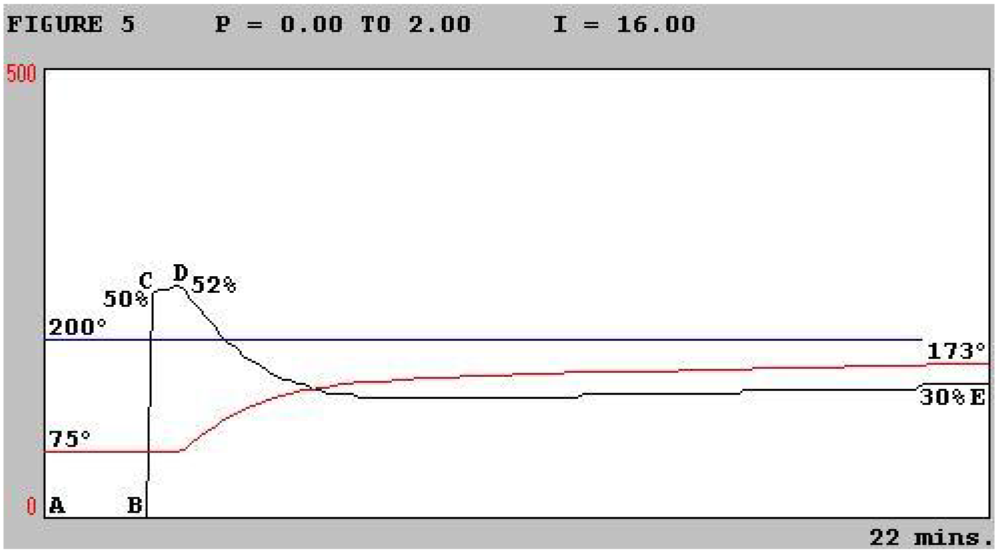
so far we’ve looked at how the PID’s Integral response is measured ... and how it responds to an artificially generated input signal ... now we’re almost ready to see how it can help us control a system in the real world ... well, almost real ... the system we’ll be using for our next examples will be the “Ramrod” ... this is a software simulation of the Hotrod trainer which I use for student lab exercises ... basically it’s a small heat control application ... the temperatures that we’ll be using will range from 0 to 500 degrees F ... each of the graphs below will have a time scale (X axis) of 22 minutes ... our setpoint will be kept at a constant 200 degrees for each test ...

next in Figure 4 we’ll look at a system which does not use Integral action at all ... as shown by the legend at the top, we start the test with the PID’s Proportional setting at 0.00 ... this basically means that the PID is turned off ... since we have no heating action yet, the PV is currently running along at 75 degrees which is our room temperature ...



from point A to point B the Proportional setting remains at 0.00 ... then at point B we suddenly change the Proportional setting to 2.00 ... as discussed in another paper, the PID calculates a CV output as follows ... E=SP-PV ... so E=200-75 ... E=125 ... converting to a percentage, 125/500=0.25 so our Error = 25% of full scale ... the Proportional setting is 2.00 ... so the Proportional contribution to the CV is 2.00*25% or 50% ... and that’s the 50% CV output that you find at point C ... now even though the CV is presently driving the RAMROD’s heater at 50%, the actual temperature cannot change instantly ... the PV temperature remains at its original 75 degrees reading for almost a full minute ... personally I usually refer to this period as the system’s “deadtime” ... other people call it the system’s “lag” time ... and notice that the CV during this period remained at a constant 50% drive as shown by the flatline response from point C to point D ... then once the heat reaches the temperature sensor, the PV finally begins to rise ... the rising PV means a decrease in the Error and so the CV drops accordingly ... the system eventually comes to a steady-state temperature of 147 degrees with a heater drive of 21% ... the remaining Error of 53 degrees is often referred to as the “droop” resulting from the use of a Proportional-only controller ... this is a textbook example of a system which needs Integral action to reach and maintain the desired setpoint ...

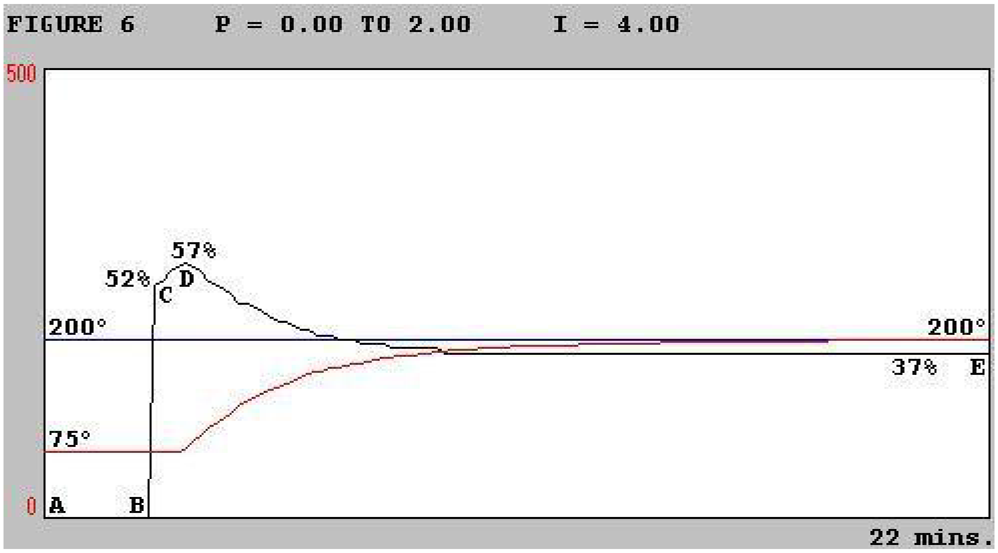
now in Figure 5 we’ll add in just a little bit of Integral action and see if we can note any improvement to our controller’s operation ...



again we’ll start with a “room temperature” PV and with our PID controller turned off by a Proportional setting of 0.00 ... we’ll go ahead and dial in an Integral setting of 16.00 ... this will NOT affect the CV, however, as long as the Proportional setting remains in its “turned off” condition of 0.00 ... (remember that the Integral response is multiplied by the Proportional setting ... and zero times anything is still zero) ... and so we’re running along from point A to point B with a PV temperature of 75 degrees ... at point B we turn on the PID by increasing the Proportional setting to 2.00 ... as in the last figure, the CV jumps up to 50% due to the Proportional action ... the Integral setting of 16.00 means that in another 16 minutes the CV will have “repeated” the Proportional’s 50% response by driving the output up to 100% on ... and you can actually see the Integral action “ramping” the CV up from 50% to 52% between point C and point D ... but before the Integral can work through that extra 16 minutes of ramping the CV, the PV begins to rise ... the Proportional action responds to the decrease in Error by decreasing the CV ... but unlike the previous figure, this time the small amount of Integral action continually adds to the CV in an effort to eliminate that residual error

... and so at the end of the 22 minutes of our graph, the PV has risen to 173 degrees (up from the 147 degrees of the previous test) ... and note (very important) that the CV is STILL continuing to increase ... the Integral hasn’t given up yet ... in fact, if we let the system continue to run through another 22-minute graphing period or two, then the PV would indeed reach the target and then begin to track the setpoint ... this is the Integral action at work ... the only problem is that we didn’t dial in enough Integral action to get the job done in a timely fashion ... so let’s try a smaller Integral setting ... remember that where the Integral is concerned, a smaller setting gives a larger response ...

here in Figure 6 we’ve increased our Integral action by a factor of four over our previous test ...



and this time when we turn our PID on at point B, the Integral has enough horsepower to kick the CV up to 52% at point C ... this is already a step in the right direction ... but the biggest advantage of the increased Integral action is still to come ... notice how much more rapidly the CV is being ramped up from point C to point D ... now we’re up to 57% before the Proportional action begins to taper off on the CV output ... and here’s the important part ... the Integral continues to respond to the remaining error all the way through the 22 minutes shown on the graph ... until just about at the end of the test, we’ve finally reached our setpoint of 200 degrees ... the increase in the CV from the inadequate 21% value of Figure 4 to the proper value of 37% here in Figure 6 is due to one thing and to one thing only ... the Integral action ...

well now we might ask ourselves this: if an Integral setting of 16.00 didn’t give us enough Integral action ... and an Integral setting of 4.00 was an improvement ... then maybe even MORE Integral action would be even better for our system ... maybe we can crank in enough Integral action to reach the setpoint in much less time and make the boss even happier ...

so here in Figure 7 we’ve dialed in an Integral setting of 1.00 ... this will increase our Integral action by a factor of four over our last test ...

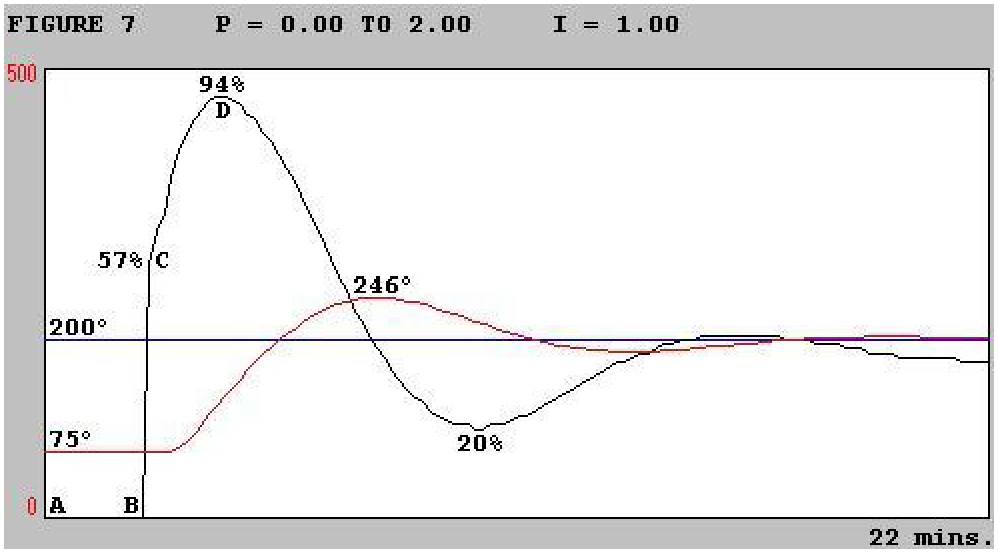


FIGURE 7 P = 0.00 TO 2.00 I = 1.00

but oops! this time we’ve gone too far ... now the Integral has enough horsepower to drive the CV all the way up to 94% ... the PV responds to this higher CV by overshooting the setpoint ... we get all the way up to 246 degrees before the PV starts heading back down again ... so this points out the fact that too much of a good thing can turn out to be a bad thing ... if we use too much Integral action then the system will begin to oscillate on us ...

but what the heck ... we’re here to learn ... so while we’re playing around let’s just crank the Integral up by another factor of four ... let’s see what this puppy will really do ... and in Figure 8 we have our answer ...

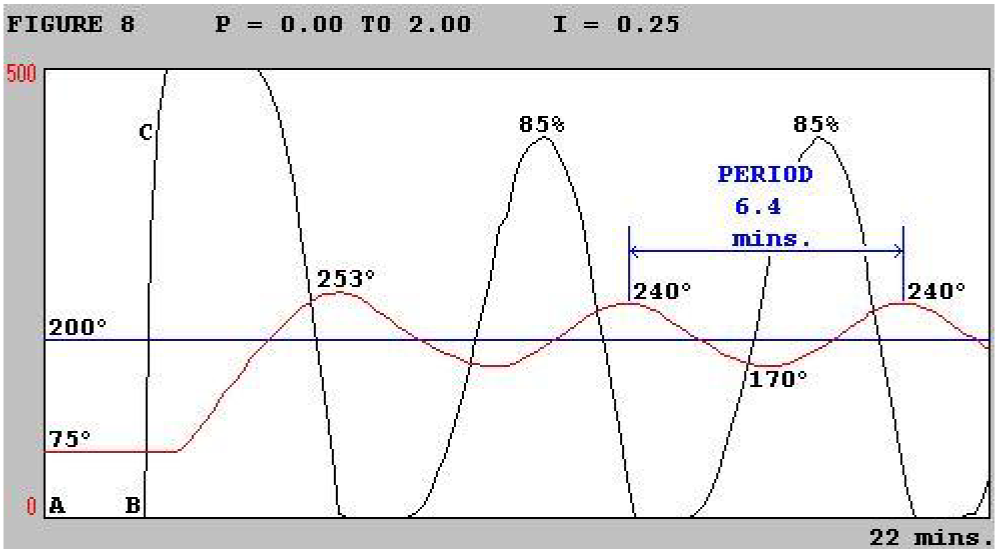


FIGURE 8 P = 0.00 TO 2.00 I = 0.25

after an initial surge up to 253 degrees, the PV goes into a series of continuous oscillations ... these take us from a low of 170 degrees up to a high of 240 degrees ... over and over and over ... but while we’re here let’s go ahead and measure the system’s period of oscillation ... a little bit of work with a stopwatch will show us that our RAMROD system is oscillating with a period of 6.4 minutes ... personally I prefer to print out the graph and measure the period with a ruler ... setting up a simple mathematical ratio between the peak-to-peak distance and the distance of the entire graphing period gives me much better results than the stopwatch method ... but that's just my personal way of doing it ...

ok ... one more experiment and we should be just about finished with the basics of the PID's Integral action ... for this next one we'll turn our Integral action completely off (by using a setting of 0.00) and go back to Proportional-only control again ...

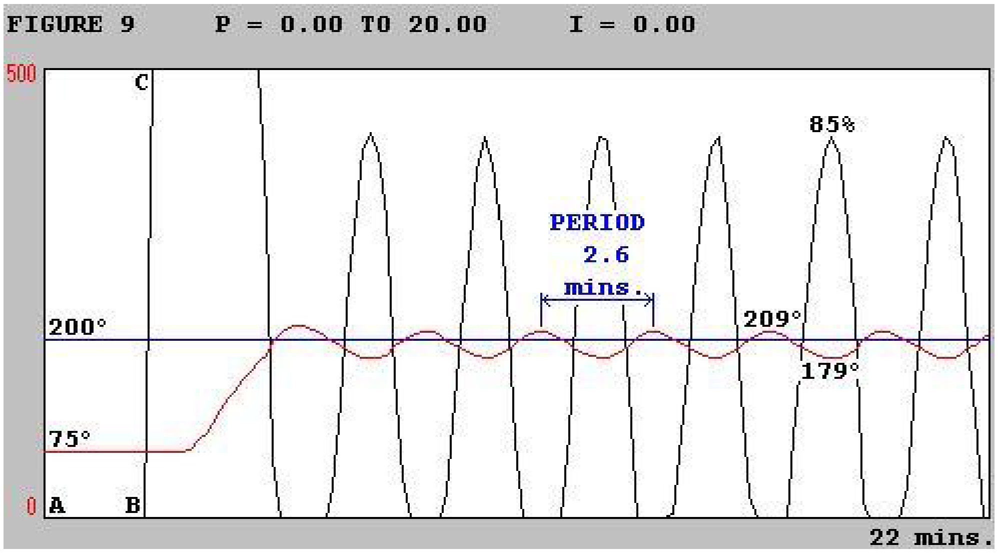


FIGURE 9 P = 0.00 TO 20.00 I = 0.00

in Figure 9 we've cranked our Proportional action up by a factor of ten over our previous experiments ... and that's enough Proportional action to cause the RAMROD system to continuously oscillate ... but look at the difference in the period of oscillation compared to what we saw in Figure 8 ... the fact is that either too much Proportional action or too much Integral action can cause a system to oscillate ... but in most systems, the period of oscillation will be much shorter (faster) when Proportional action is causing the mischief ... in contrast, too much Integral action will usually cause oscillations with a much longer (slower) period ... knowing this little fact can often help you decide which of the two components to adjust to eliminate oscillations while tuning a system by the trial-and-error method ...

summary: so now we’ve seen how to measure the Integral action in units of “minutes-per-repeat” ... we’ve seen how the Integral action responds to a “no error” condition ... we’ve seen how the Integral action responds to an error which is increasing ... and to an error which is decreasing ... and to an error which is staying the same ... we’ve seen how different adjustments to the Integral setting and to the Proportional setting can each affect the Integral’s response ... we’ve seen how Integral action can help us eliminate the “droop” below the setpoint which results from using a Proportional-only controller ... we’ve seen how too much Integral action can cause the system to oscillate ... and we’ve seen that the oscillations caused by too much Integral action are usually much slower than the oscillations caused by too much Proportional action ...

and that’s all that I have to say about Integral action for today ...

best regards,
Ron