12 OVERRIDE (SELECTOR) CONTROL

❖ OVERRIDE CONTROL

Override control, also called selector control, exists when one process variable is the controlling variable in normal operation. During abnormal operation, however, another process variable assumes control to prevent some safety, process, or equipment limit from being exceeded.

A key element of an override control strategy is a selector switch, implemented either as a hardware device or a software function block. Depending on how it's configured, this selector switch passes the higher or lower of several input signals to its output. There are several ways of using selector switches in a control strategy. One is to select the higher or lower of several measurement signals to be passed on as the process variable to a feedback controller. For example, the highest of several process temperatures may be selected automatically to become the controlling temperature. As process conditions change, the location of the highest temperature may change also. The selector switch assures that, regardless of process conditions, the controlling point is the highest of the measured temperatures.

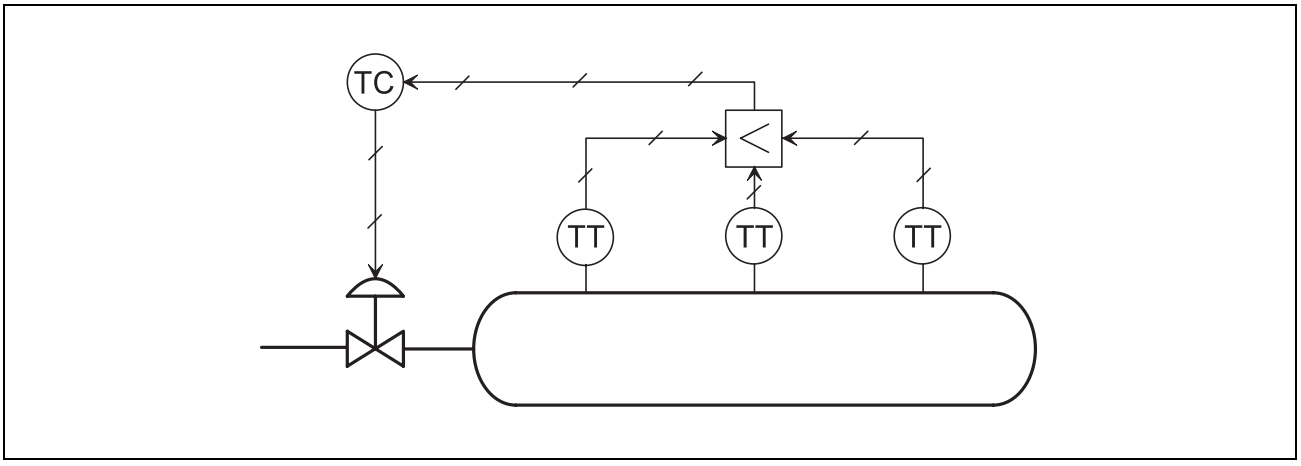

Figure 12-1. Controlling from the Highest Point of a Temperature Profile

Placing a selector switch in the measurement side of a controller, though perhaps important from the vantage point of a particular process application, poses very little technical challenge for the control engineer. If each of the process sensors responds in a similar way to changes in

the controller output, then the transition from one sensor to another will be virtually imperceptible.

There are also selectors which select the middle of three inputs. These are used primarily in high-criticality applications, where the failure of a signal, either by going "high" or "low", could not be tolerated. By taking the middle-of-three inputs, then as long as two of the signals remain viable, the application can continue.

From a control engineers, viewpoint, however, a more interesting application than these occurs when the selector device selects between the higher or lower of several controller outputs.

For example, suppose that the outlet temperature from a process heater is normally controlled by manipulating a fuel-valve position. In normal operation, the outlet temperature should be maintained at its set point. Suppose, however, that the temperature of the heater tubes has an operational limit. If the heater has been properly designed, the tube temperature will remain below the limit during normal operation. On the other hand, abnormal conditions, such as coking in the heater tubes or overloading of the heater, may cause the tube temperature to rise.

If a representative tube temperature is measured, a limiting controller may be used to prevent encroachment on the operational limit. Here, the two controller outputs would be connected to a low signal selector; the controller demanding the lower fuel valve position will override the other. In normal operation, this controller will be the heater outlet temperature controller.

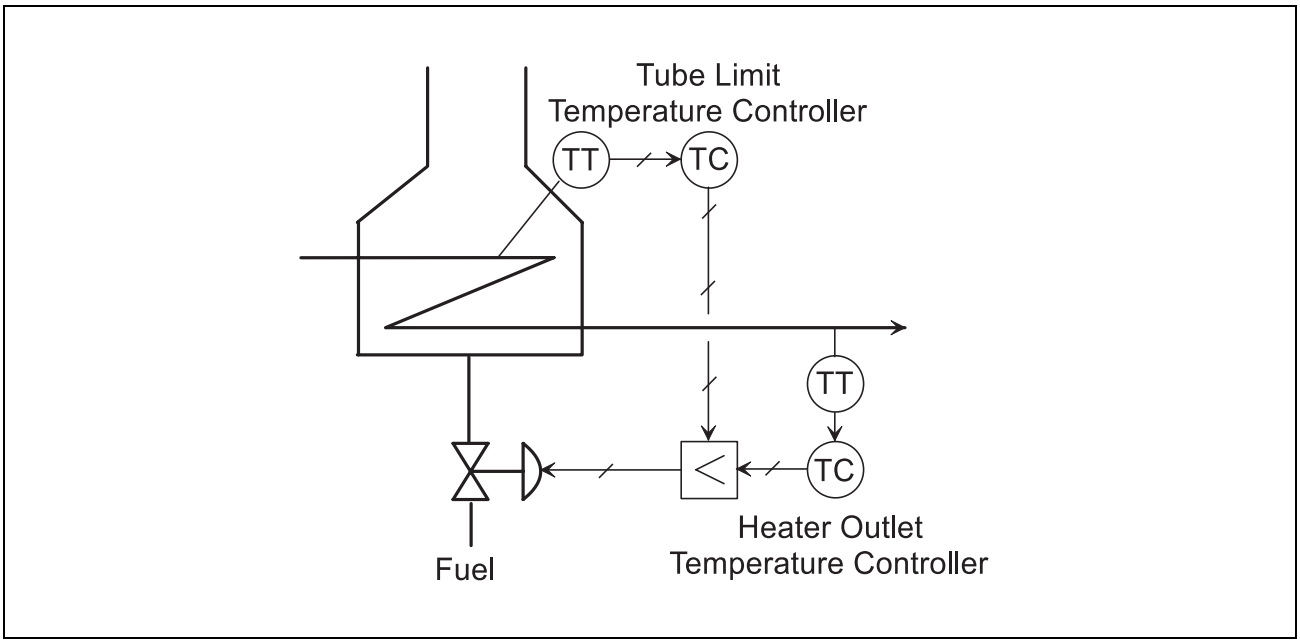

Figure 12-2. An Example of Override [Selector] Control

Note that we present this example merely as an easy-to-visualize application of override control. We will not attempt to explore in depth here the question of how to obtain a representative measurement of tube temperature. One possibility would be to measure several tube tempera-

tures, as in Figure 12-1, and select the higher of the temperatures as the process variable for the limiting controller. We also assume that in an actual installation of this type, one or more alarm points would be set just below the set point of the limiting controller to warn the operator of an impending override. In addition, there would likely be an independent temperature sensor that has a safety shutdown point set just higher than the set point of the limiting controller, in case the override control failed to prevent the tube temperature from rising further. These additional items, however necessary, lie outside the scope of this book.

Let us first suppose that we utilize ordinary PI controllers for this application, as shown in Figure 12-3.[1] Since the fuel valve is undoubtedly a fail-closed type, then both controllers will be set for reverse action. In normal operation, the heater outlet temperature will be near its set point and the tube temperature will be below its set point (limit value). We can assume that the output of the heater outlet temperature controller is between its extreme limits, that is, it is someplace mid scale. If the heater has been in this condition for some time, with the tube temperature at less than its set point, then the integral action in the tube temperature controller will have increased its output to the maximum value. In other words, the heater tube temperature controller will be in a windup condition.

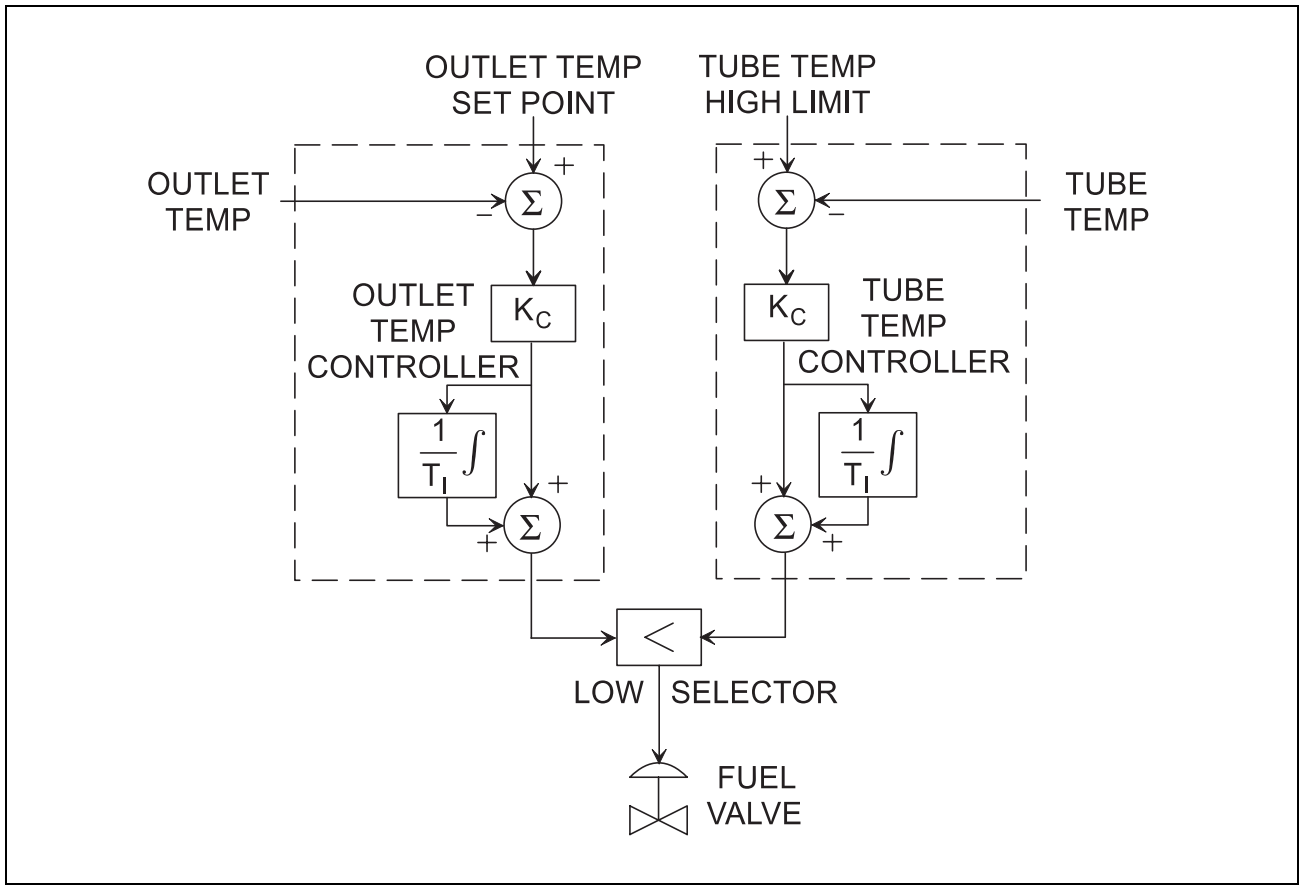

Figure 12-3. Override Control Using Ordinary PI Controllers (Not Recommended)

1. We refer to PI rather than PID because the derivative mode is inconsequential to the technology discussed here.

Since the output of the heater outlet temperature controller is less than that of the tube temperature controller, the outlet temperature controller's output will be selected and will be passed to the fuel valve. As long as the tube temperature stays below its limit, the heater will remain under the control of the outlet temperature controller, just as if the other controller were not present.

Now, suppose that, because of some abnormal process condition, the heater tube temperature rises toward the limit value. For this application, several process conditions could cause the tube temperature to rise. For instance:

- An increase in the heater feed rate would call for additional fuel to be released, consequently a hotter combustion zone and ultimately a higher tube temperature.

- Coking inside the heater tubes, which some steam reformer furnaces experience, will reduce the heat transfer. This will require a higher temperature in the combustion zone (i.e., additional fuel) to transfer sufficient heat to the process fluid. This also results in a higher tube temperature.

- A decrease in heater efficiency will also result in increased fuel firing, and consequently a higher tube temperature.

For the purpose of the discussion here, what caused the tube temperature to rise is unimportant. We only need to know that it is a situation that could occur. If the tube temperature rises slowly toward the limit without exceeding it, the output of the limiting controller will remain at its maximum (wound-up) value and will have no effect on the heater firing rate.

Once the tube temperature crosses the limit, however, the error reverses in sign and the controller output starts decreasing. It must decrease all the way below the output of the heater outlet temperature controller before it will have any effect on the position of the fuel valve. The tube temperature could remain over limit for a significant period before the limit controller overrides and begins reducing the fuel flow. The length of time depends on the rate at which the limit controller output decreases (which in turn depends on the controller tuning as well as the amount by which the limit value is exceeded) and the value of the heater outlet temperature controller output.

To summarize our understanding to this point, our application objective—installing a high-limit controller that can override the normal outlet temperature controller if the temperature of the tube becomes high—is a valid goal. Yet, our application is somewhat flawed because of windup experienced by the nonselected controller.

We will now digress and return to a PID controller modification that we first presented in chapter 5. Recall that Figure 5-12 and the discussion related to it presented the concept of external reset feedback (ERF). A PI controller constructed with ERF has an extra input port (the reset feedback port), which, if it is connected directly to the controller output, causes the controller to behave as an ordinary PI controller. The signal into the reset feedback port can

originate from some place other than the controller output, however. We will make use of this feature in the application under discussion.

Suppose that we replace the two ordinary PI controllers with controllers that have ERF. The controller outputs are connected to a low-signal selector as before, but the reset feedback for both controllers originates from the output of the low-signal selector, as shown in Figure 12-4.

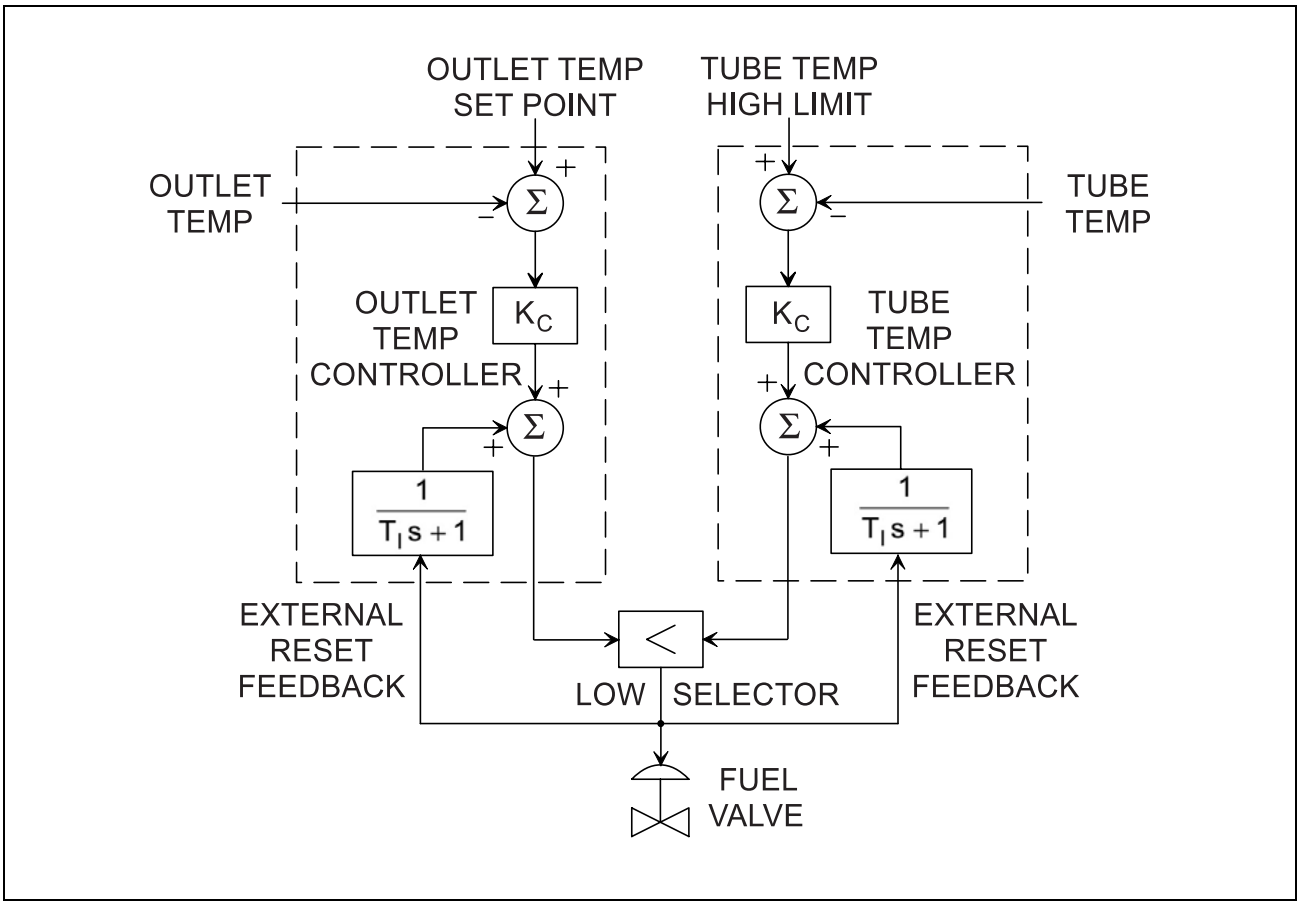

Figure 12-4. Application of External Reset Feedback for the Heater Tube Temperature Example.

We will demonstrate that with this arrangement the nonselected controller will not wind up, even though its set point and measurement may differ. To provide an intuitive insight into this idea, we will use the numerical values shown in Tables 12-1 through 12-5. Transfer each set of these numbers to a diagram like that shown in Figure 12-4 in order to follow a "moving" scenario.

Suppose that the outlet temperature set point is 40 percent of its scale, and the tube temperature limit set point is 70 percent of its scale. Suppose that in normal operation the valve is positioned at 50 percent open, the actual heater outlet temperature is maintained at set point, and the actual tube temperature runs below its set point, say, at 65 percent of its scale. Suppose further that both controllers are tuned with a gain of 1 (PB of 100%).

Table 12-1. Normal Operation

Signal	Heater Outlet Controller	Selector Output	Tube Limit Controller
Set Point	40%		70%
PV	40%		65%
Error	0%		5%
$K_C \times$ Error	0%		5%
Reset Feedback	50%		50%
Output of First-order Lag	50%		50%
Controller Output	50%		55%
Valve Position		50%	

Table 12-1 summarizes the various signal values during normal operation. Note that the system could remain in the normal operation state indefinitely without the nonselected controller (tube limit) winding up. The reason for this is that with the ERF configuration there is no free integrator with a nonzero input that would cause windup. Instead, there is a first-order lag whose input and output signals are equal in the steady state.

Now, suppose that the tube temperature rises slightly, say, to 69 percent of scale. Table 12-2 shows the new scenario. Since the tube temperature has not reached the set point, there is no effect on furnace firing.

Table 12-2. Tube Temperature Rise But Not To Limit

Signal	Heater Outlet Controller	Selector Output	Tube Limit Controller
Set Point	40%		70%
PV	40%		69%
Error	0%		1%
$K_C \times$ Error	0%		1%
Reset Feedback	50%		50%
Output of First-order Lag	50%		50%
Controller Output	50%		51%
Valve Position		50%	

Now suppose that the temperature rises an additional 1 percent, just to the limit set point. Table 12-3 depicts this situation. Now, we are in a “don’t care” situation in which both controller outputs are the same, so the selector switch continues to pass the same values to the fuel valve as well as to the reset feedback ports.

Table 12-3. Tube Temperature Rise To Limit

Signal	Heater Outlet Controller	Selector Output	Tube Limit Controller
Set Point	40%		70%
PV	40%		70%
Error	0%		0%
$K_C \times$ Error	0%		0%
Reset Feedback	50%		50%
Output of First-order Lag	50%		50%
Controller Output	50%		50%
Valve Position		50%	

Again, suppose the temperature rises 1 percent above the limit. The three previous tables have shown a static situation, one that could remain indefinitely, but Table 12-4 is transitory. It depicts the situation right after the tube temperature rise. This is a transitory table because the first-order lags are unbalanced. For the tube limit controller, the output of the first-order lag will fall to match its input. This will force down the controller output, the signal to the valve, and the reset feedback to both controllers. This situation will be repeated as long as the tube temperature is above the limit. In other words, the tube limit controller is now behaving like a normal PI controller.

Table 12-4. Tube Temperature Rise Above Limit

Signal	Heater Outlet Controller	Selector Output	Tube Limit Controller
Set Point	40%		70%
PV	40%		71%
Error	0%		-1%
$K_C \times$ Error	0%		-1%
Reset Feedback	49%		49%
Output of First-order Lag	50%		50%
Controller Output	50%		49%
Valve Position		49%	

Consider the heater outlet temperature controller right after the condition depicted by Table 12-4. Two situations are occurring simultaneously:

- The output of the first-order lag will go down to match its input; this will tend to cause this controller output to go down.

- If the heater outlet temperature has been maintained at set point with a 50 percent valve position, it is logical to assume that with a lower fuel valve position the heater outlet temperature will go down. Thus, the error will go to a positive number; this will tend to cause this controller's output to go up.

Let us assume that with the abnormal condition, the tube limit controller stabilizes ($PV - SP$) when the signal to the valve has decreased to 45 percent. Furthermore, suppose that with that new valve position, the heater outlet temperature drops to 37 percent. (Recall that a valve position of 50 percent was required to maintain heater outlet temperature at its set point of 40 percent.) Table 12-5 depicts the new equilibrium condition.

Table 12-5. Tube Limit Temperature Controller In Control

Signal	Heater Outlet Controller	Selector Output	Tube Limit Controller
Set Point	40%		70%
PV	37%		70%
Error	3%		0%
$K_C \times$ Error	3%		0%
Reset Feedback	45%		45%
Output of First-order Lag	45%		45%
Controller Output	48%		45%
Signal to Flow Cont		45%	

Table 12-5 is again a static table. The control system could remain in this condition indefinitely, as long as the abnormal condition exists. The point is that the situation has reversed from that shown in Table 12-1, in which the outlet temperature was controlling but the tube limit controller was not winding up. The tube limit condition is now controlling. The heater outlet temperature is below set point, but the outlet temperature controller is not winding up. When the abnormal situation is cleared up, the steps above will be reversed: the control will revert to the heater outlet temperature controller.

Following this example carefully reveals the difference in behavior between this control scheme and the control scheme that used ordinary PI controllers (Figure 12-3). In particular, observe Table 12-3. The tube temperature has just reached the limit, and the tube limit controller is on the verge of overriding the other controller and assuming control of the valve. With ordinary PI controllers, the tube limit temperature controller would merely be on the verge of starting to unwind from its maximum output value. It would then have to unwind all the way below the other controller's output before it had any effect on furnace firing.

The foregoing discussion is but one example of override (selector) control. The opportunities for applying this technique are manifold. The characteristics of all such applications are as follows: in normal operation one controller is in control, but in abnormal operation another con-

troller (or perhaps one of several other controllers) overrides. Often the goal is to protect process equipment, even at the sacrifice of normal process control.

Another type of application for override control is to automatically modify the control strategy between process startup and normal operation. In the pulp and paper industry, steam to a batch digester is initially controlled by a flow controller until the pressure rises to the operating point, then a pressure controller overrides and assumes control of the valve (see Figure 12-5).

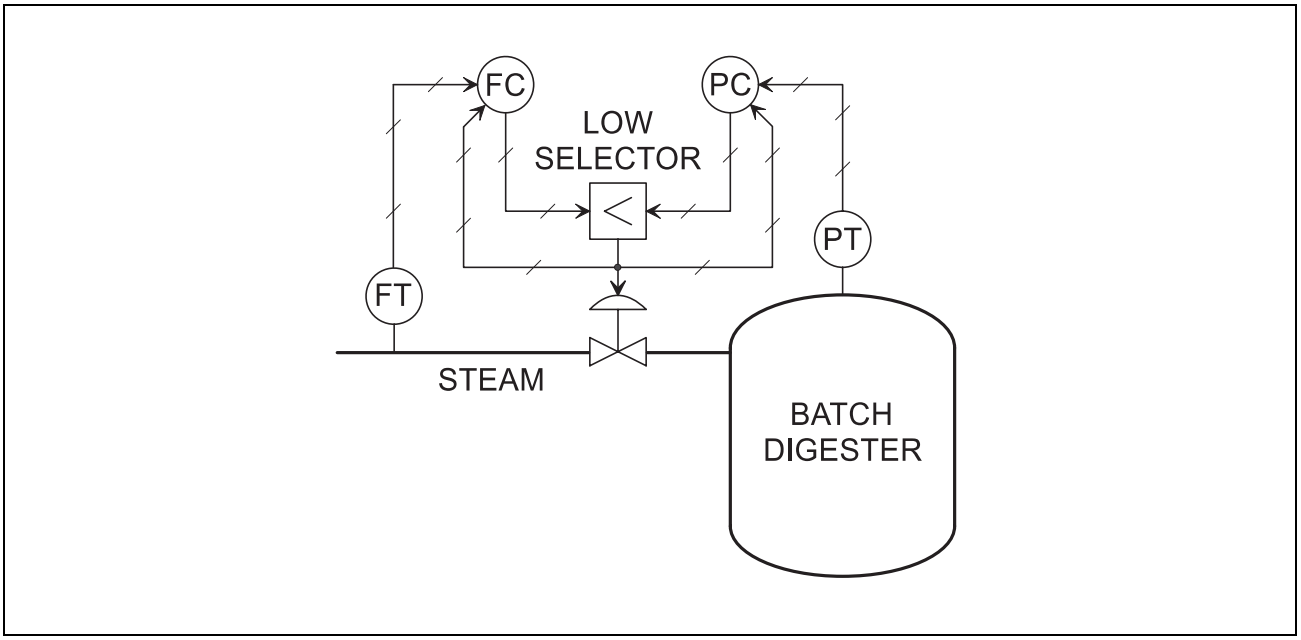

Figure 12-5. An Example of the Use of Override Control for Process Startup

For a generic process, suppose that the feed rate to a process unit is maintained constant by a flow controller. Further, suppose that the process unit uses some utility, for example, steam, as a part of the process. If the steam supply is limited, then the process controller may move the steam valve wide open. Rather than produce off-spec product, the normal procedure would be to reduce feed rate. This procedure can be automated by installing a "valve position controller" (see chapter 16) for the steam valve. Its set point would be near the maximum allowable valve position, say 95 percent. If the process controller drives the valve to that limit, the valve position controller will override the feed-flow controller to reduce feed rate (see Figure 12-6).

For a distillation column, an adverse condition known as "tower flooding" can occur under certain abnormal conditions. This is normally caused by excessive vapor flow within the column. It can often be detected by measuring the differential pressure across a section of the column. If a composition controller (or inferred composition controller, such as temperature) normally controls the steam to the reboiler, then a differential-pressure controller should override and reduce the steam flow. In this case, the selected signal is the set point for a secondary steam-flow controller, not the valve signal itself. The external reset feedback signal could be the flow controller set point. A preferred signal, however, would be the actual flow measurement. This would provide the correct ERF signal regardless of the manual/automatic status of the flow controller.

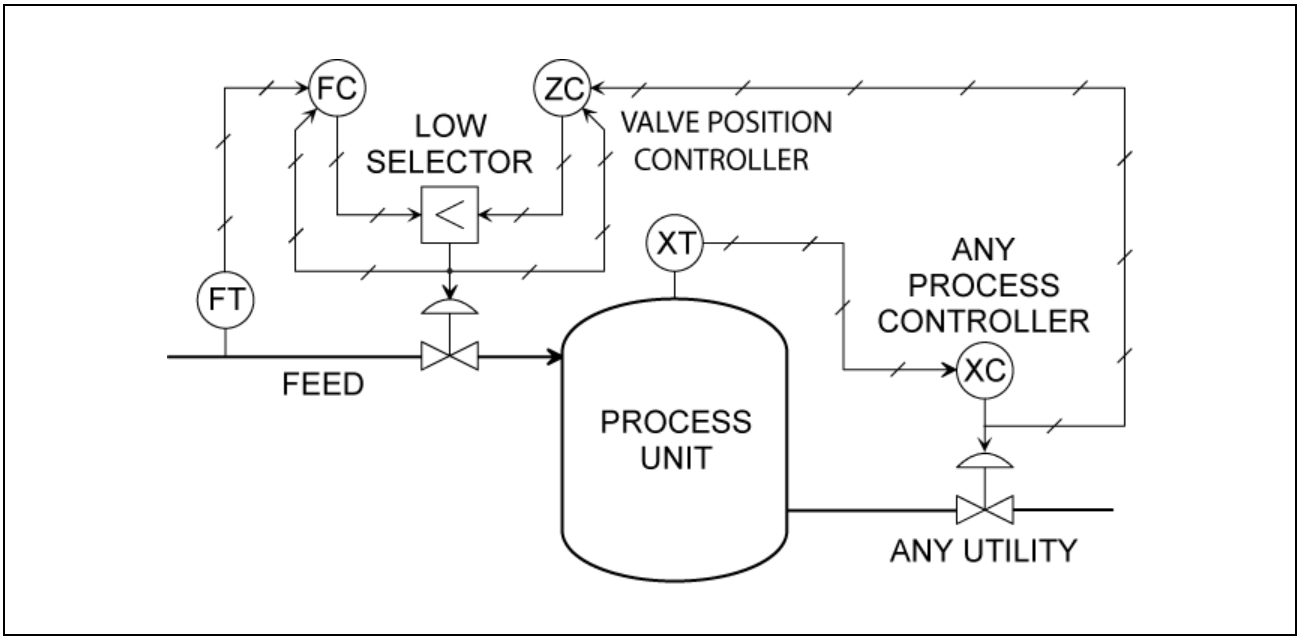

Figure 12-6. An Example of the Use of Override Control for Operating Near Process Limits

In a further level of sophistication, suppose that the temperature controller does not directly set the steam flow, but sets a required steam-to-column feed ratio. For a PI controller with ERF to function as a normal PI controller, its ERF signal must represent the same quantity as its output. (In Figure 12-4, each controller output represents a required valve position; the ERF also represents valve position.) Thus, for the distillation tower it would be incorrect to use a steam-flow signal, whether flow controller set point or actual steam flow, as the ERF to the temperature controller, since its output represents a required steam-to-feed ratio. The correct configuration would be to use the measured steam and feed rates to calculate the actual steam-to-feed ratio, and use this as the ERF for the temperature controller, as shown in Figure 12-7 (Ref. 12-1).

At a compressor station in the pipeline industry, both the suction and discharge pressures are monitored. Whichever pressure demands the lower compressor speed becomes the overriding controller, as shown in Figure 12-8.

For a multi-engine-compressor station, there may be limiting controllers that apply to the station as a whole as well as limiting controllers that apply only to individual engines. Engine speeds are the ultimate manipulated variables; these respond to the lowest of several controller outputs. Figure 12-9 shows controls for two engine-compressor sets. A typical station may have four or more compressors.

Overall station controls are suction pressure, discharge pressure, and station flow. The lowest of these controller outputs sets an upper limit for the speed of all the engines. Individual engine controls, such as torque and engine temperature, may further reduce the speed for any particular engine. In addition, the operator may set a maximum speed setting that overrides all of the other controls.

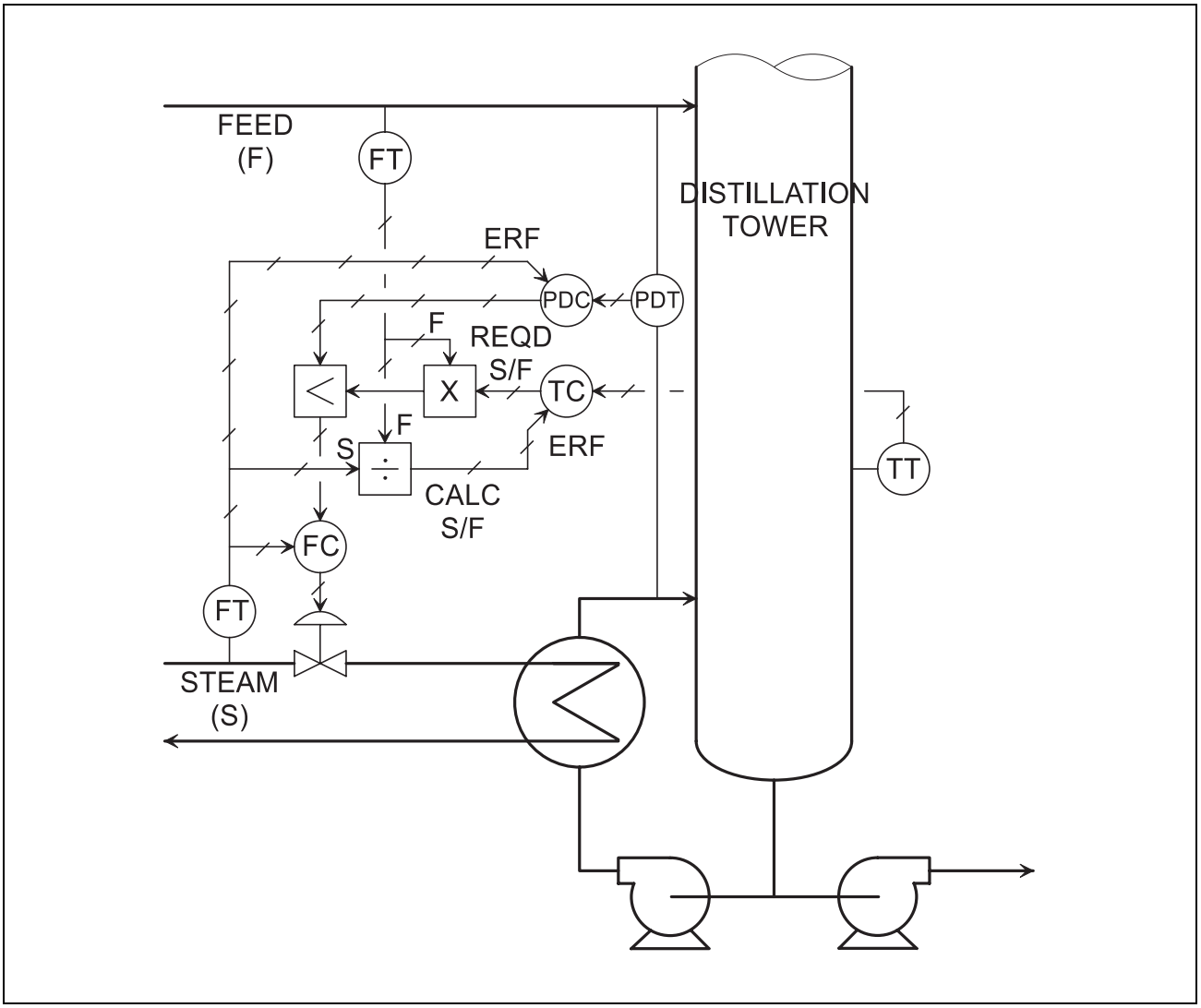

Figure 12-7. An Example of Combining Override, Ratio, and Feedforward Control

To prevent windup of the individual engine controls, conventional external reset feedback is applied to the torque and engine temperature controls. The operator speed setting, being simply a constant entry, cannot wind up; hence it does not require external reset feedback.

Determining an appropriate signal for external reset feedback for the station controllers presents an interesting situation. At any one time, the overriding station controller may be in control of one or more engine speeds. Suppose that the suction pressure controller output in Figure 12-9 is lower than any of the other station controls. Further, suppose that it is also lower than any of the engine controllers, including the operator speed setting, for engine #1. Then the suction pressure controller will be setting the speed for engine #1. Suppose at the same time that the output from one of the controllers from engine #2, say the torque controller, is lower than the suction pressure controller output. The torque controller will be in control of the speed of engine #2. The external reset feedback for the station controllers should be the *higher* of the individual engine speeds. In this case, it would be the signal to the engine #1 speed controller. With this configuration, none of the controllers will wind up regardless of whether an engine controller or a station controller is in control of any particular engine speed.

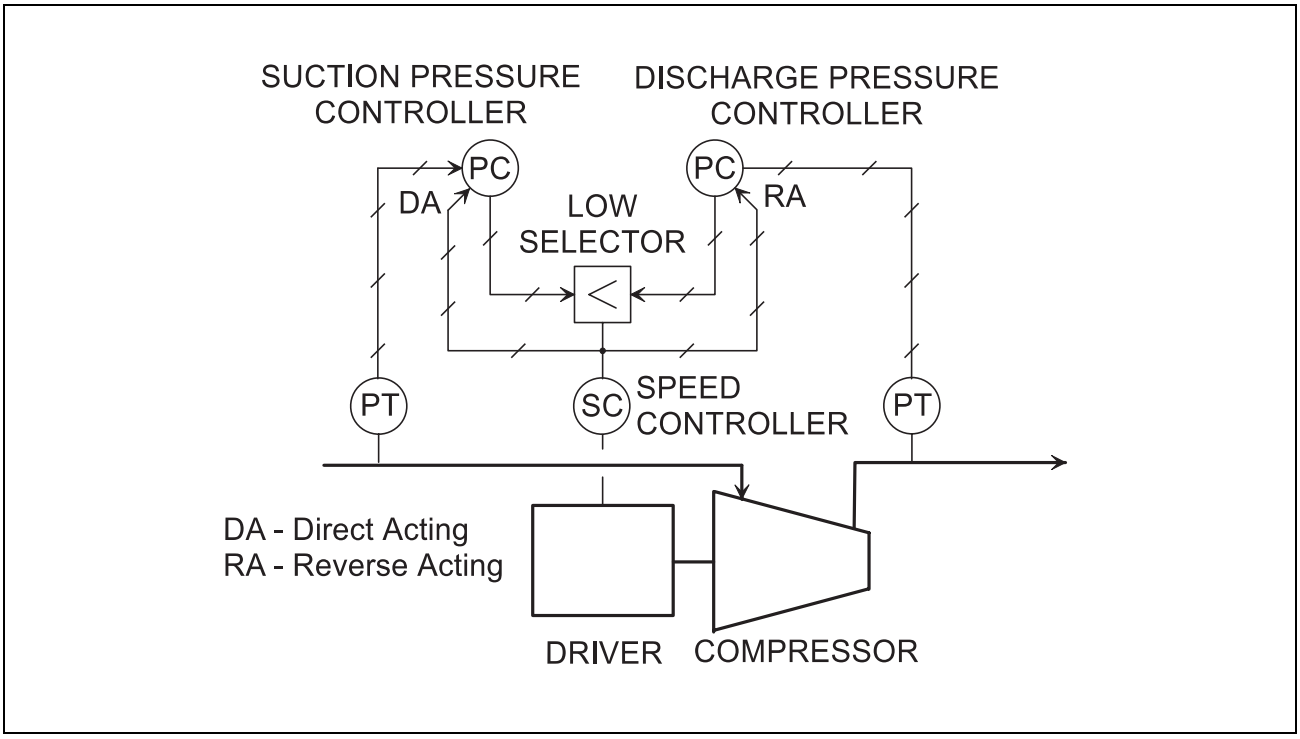

Figure 12-8. An Example of Override Control in the Pipeline Industries

The configuration described here has been recommended for a particular four-engine-compressor station. As far as the author knows, however, no existing implementation embodies this exact configuration.

(As a separate topic, consideration should be given to adjusting the gain of the station controllers, depending upon the number of engine speeds which can be manipulated by a station controller at any one time.)

These examples have presented a broad spectrum of applications for override control, ranging from very simple applications (Figures 12-2, 12-5, and 12-8) to complex configurations (Figures 12-7 and 12-9). From these examples, the reader should be able to draw inspiration in applying override control to other situations.

❖ OTHER METHODS OF IMPLEMENTATION

The discussion in the last section focused on the traditional configuration of override control and began with its development during the age of pneumatic controllers. Traditional override control persists in digital form to the present day with some commercial systems. Other systems, however, use some different form of implementation. The following paragraphs present these other methods of implementation, along with a critique of each.

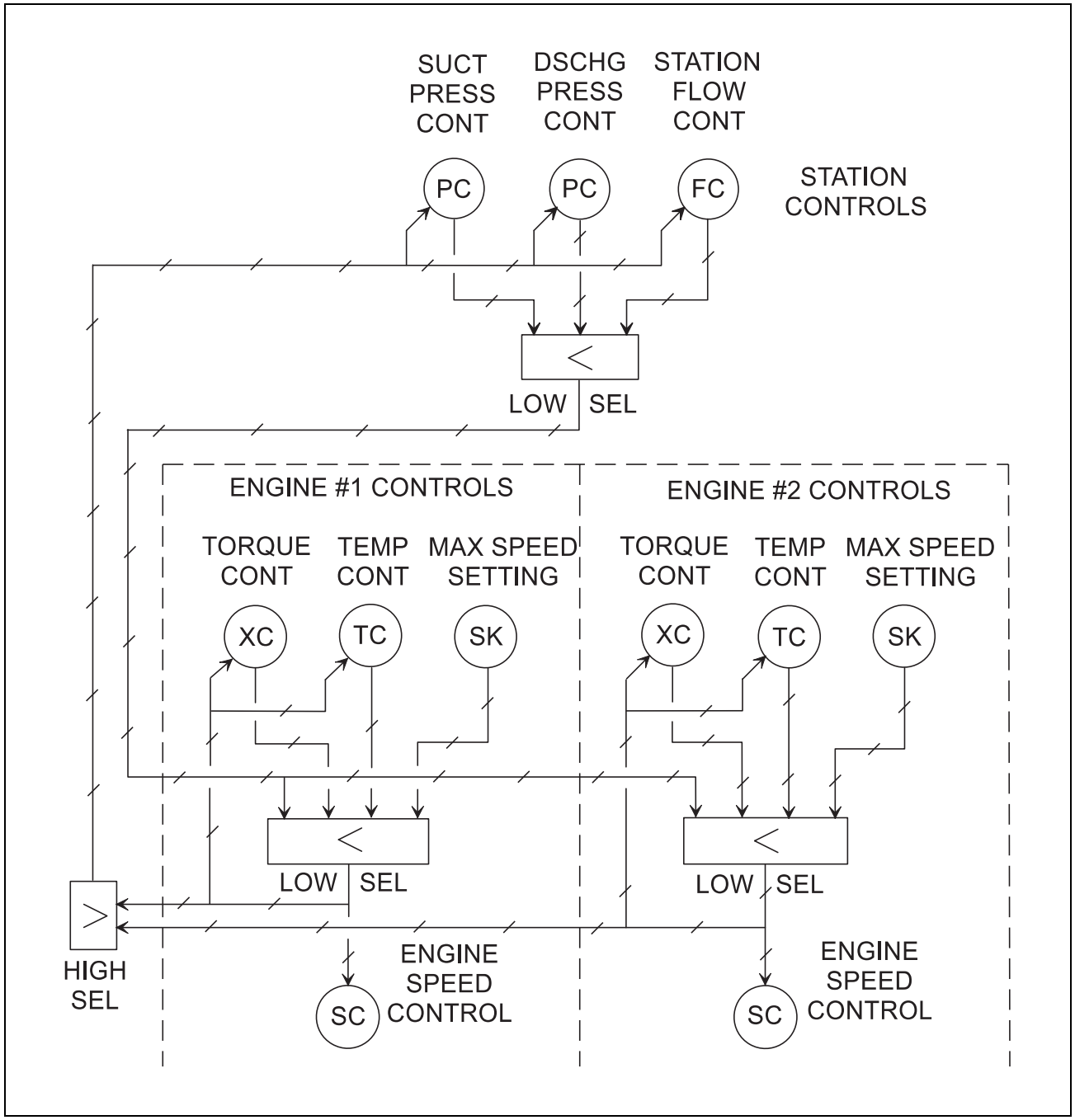

Figure 12-9. A Complex Application of Override Control for a Multiple-Compressor Pipeline Station

♦ "Pass-through" Method

In the traditional configuration of override control, the nonselected controller tracks the output of the other controller through a first-order lag whose time constant is the integral time of the controller. Thus, the tracking speed is governed by the required integral time. Lipták (Ref. 12-2, Section 1.17) states that this can be a problem when two or more controllers have significantly different integral times. After extensive simulation study, the author did not find this to

be true, and in fact found the traditional configuration performed better than any proposed solutions when the controller integral times differed significantly.

One of the proposed solutions has been to avoid the tracking speed problem by having external logic detect when a controller is a nonselected controller. Then, the output of the selected controller is immediately "passed through" (or bypasses) the reset lag, as shown in Figure 12-10. The output of the nonselected controller, then, will always be the output of the selected controller, plus gain multiplied by error of the nonselected loop.

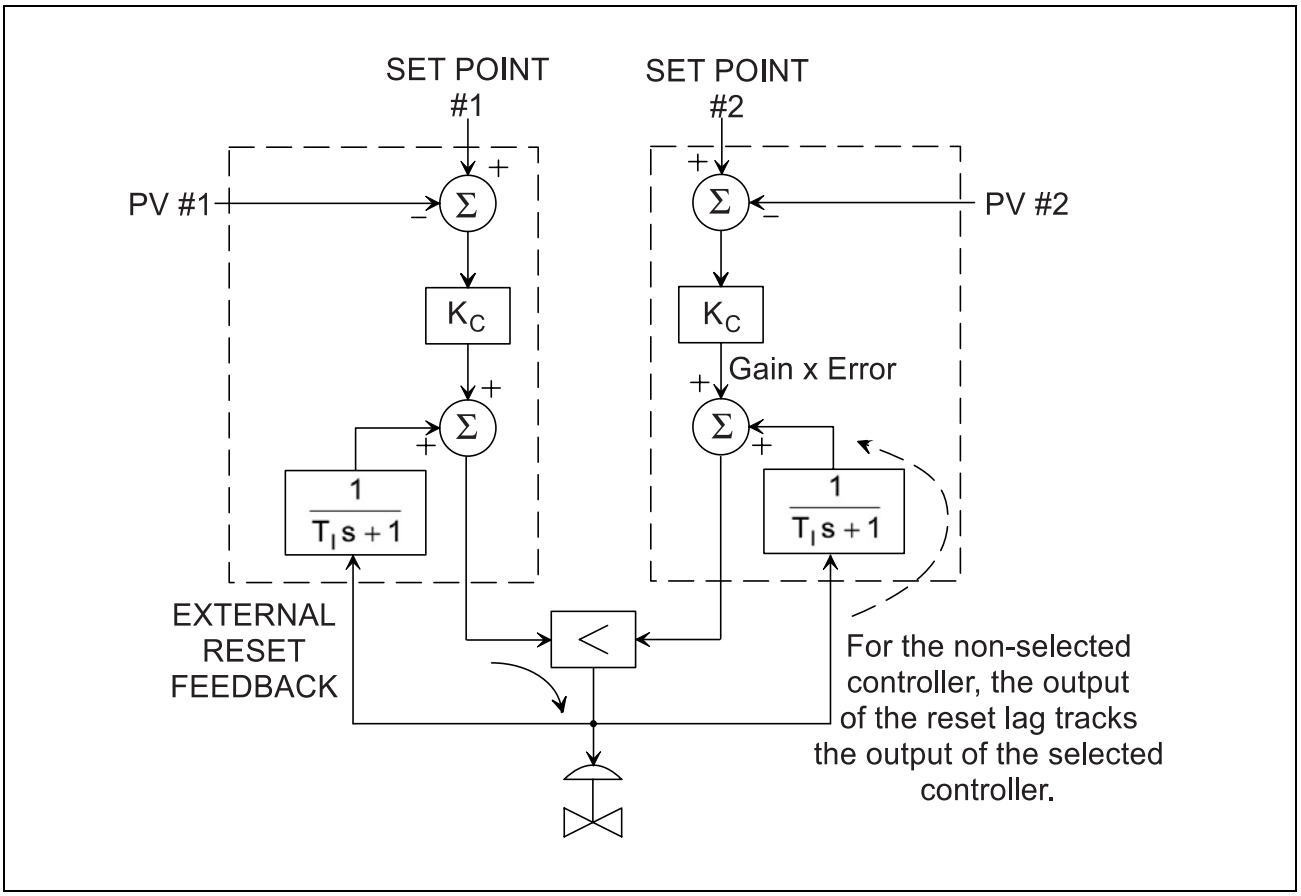

Figure 12-10. Block Diagram of the "Pass-through" Implementation Method for Override Control

Note that with the traditional method of implementation, there is never a jump in either controller output due to switchover. With the "pass-through" method, however, at the moment when one controller switches over to another, the output of the nonselected controller may jump, depending on the magnitude of its error. If the jump is away from the other controller output, then this jump is of no consequence. However, if the jump is in the other direction, it can again override the other controller and move the valve to an unwarranted position, as the Figure 12-11 and the detailed analysis that follows illustrate.

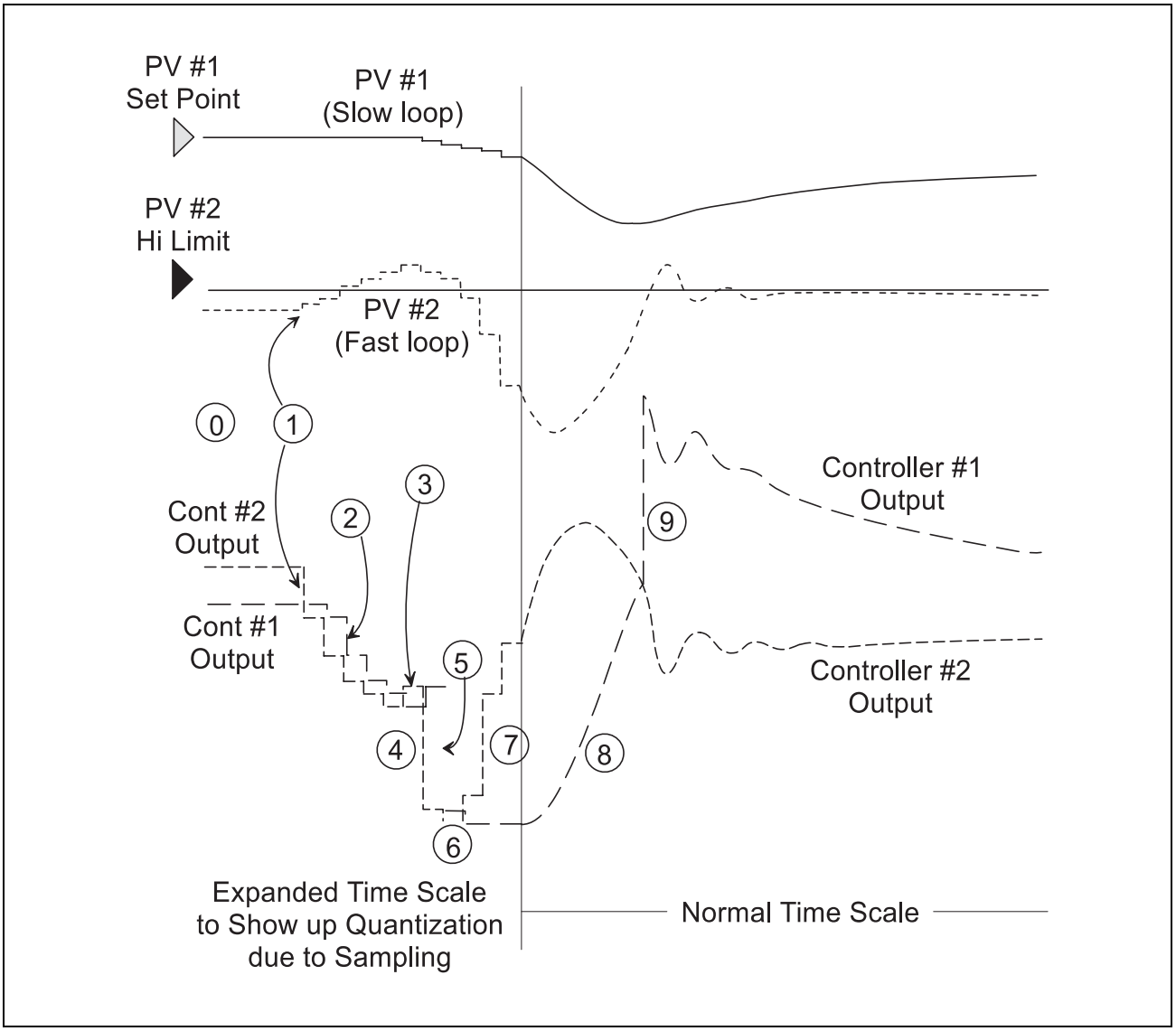

Figure 12-11. Possible Behavior of "Pass-through" Implementation Method

Detailed Analysis of Figure 12-11

(0) Initial Conditions: PV #1 (slow loop) is at set point. PV #2 (fast loop) is below its limit value. Controller #2 output is above Controller #1 output, thus Controller #1 is the selected controller. Both controllers are reverse-acting.

(1) A sudden disturbance causes PV #2 to rise and the output of its controller to fall below the output of Controller #1. Thus, Controller #2 is now the selected controller.

(2) Controller #2 output continues to fall to correct the deviation of PV #2. The output of Controller #1 tracks Controller #2's output, but with one sample time delay. (Actually, Controller #1's output is Controller #2's output plus Gain × Error of loop #1. However, since loop #1 is a slow loop and was previously on set point, then for the example its error is presumed to still be zero.)

(3) At some point, the corrective action to PV #2 will cause its controller output to reverse, then become greater than Controller #1's output. Controller #1 is now the selected controller (for one scan cycle).

(4) Since Controller #2 is now the nonselected controller, its output will be forced to Controller #1 output plus Gain × Error of loop #2. In the example, PV #2 is still above its limit, so the error will be negative. Thus, the output of Controller #2 will make a significant decrease, again overriding Controller #1 and becoming the selected controller.

(5) Controller #1 output tracks (delayed by one sample period) the output of Controller #2, plus Gain × Error of loop #1. In the example, the error of loop #1 is still small, so Gain × Error is essentially zero.

(6) The effect of a significant decrease in Controller #2 output causes PV #2 to change fairly rapidly. This in turn causes an increase in the output of Controller #2, so it becomes greater than Controller #1. Controller #1 is now the selected controller.

(7) Controller #2's output tracks Controller #1's output, plus Gain × Error in loop #2. If by now PV #2 is below its limit, then the error is positive, so there will be a jump upward in Controller output #2, away from Controller output #1.

(8) The valve is now significantly depressed from the condition required by PV #1 or by PV #2. It causes a major upset to PV #1 that is corrected slowly; hence, the output for Controller #1 (the selected controller) increases gradually.

(9) When the output for Controller #1 crosses over the value for the output for Controller #2, the output for Controller #2 is selected. Controller #1's output now makes a jump upward, since PV #1 is below its set point, hence there is a positive error.

This nine-part analysis was observed in a simulation, and shows what *can* happen under adverse circumstances. Under milder circumstances, such as a slow load disturbance on the faster loop, these conditions would probably not be observed. Even under milder conditions, however, noisy systems can adversely affect the logic that determines the selected and nonselected controllers, thus causing unpredictable results.

As we noted earlier, with the traditional method of implementation, there is never a jump in either controller output. Thus, in summary, we can say that the "pass-through" method may either perform comparably or not perform as well as the traditional method. It will never perform better.

♦ Forced Manual for the Nonselected Controller

Some systems force the nonselected controller to the equivalent of the manual mode, and depend on the bumpless transfer feature to prevent windup. If one could guarantee that a switchover would always be done with essentially zero error in the loops, then the perfor-

mance of this method would be comparable to the traditional method. If there is an error in the nonselected loop, then this method may demonstrate behavior problems similar to the "pass-through" method.

♦ Velocity (Incremental) Mode Control Algorithms

The traditional implementation of override control presented earlier utilizes position-mode PID algorithms. Hence, the final signal to the valve is used for external reset feedback. If the PID control is implemented with a velocity-mode rather than position-mode control algorithm (see chapter 5), then other considerations must be made.

In one of the early computer systems, a velocity-mode PID algorithm was used. In this system, both controllers (assuming that only two controllers participated in the override scheme) calculated a "Δm", or the amount each controller wanted to move the valve. The implementation philosophy was that the controller that should win was the one that wanted to move the valve the greater extent toward closed. The selector switch chose the minimum (algebraic) Δm and applied it to the present position of the valve. For example, if one controller wanted to close the valve by 2% (Δm = -2) and the other controller was presently on set point and did not want the valve to move (Δm = 0), then the Δm of -2 was selected, and the valve decreased in position by 2% (see Figure 12-12).

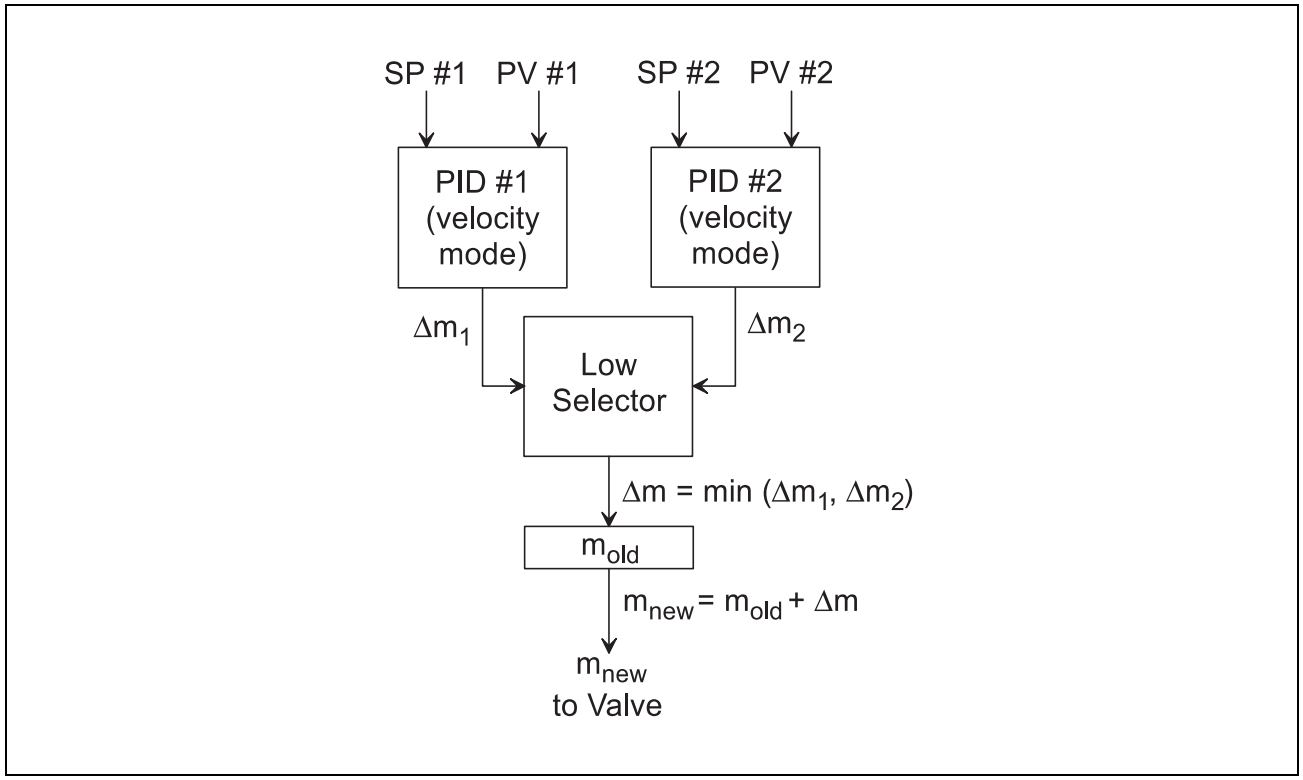

Figure 12-12. Implementation of Override Control with Velocity-mode Algorithms (Not Recommended)

One particular problem observed with this system occurred with a bottoms-stream composition controller on a distillation tower and a tower differential-pressure (inferential of tower flooding) controller, both manipulating a reboiler steam valve. The composition controller, a slow loop, was normally in control while the ΔP controller, a fast loop, normally ran below its limit. Suppose a sudden disturbance, such as a slug of light component in the feed, caused the ΔP to increase. The ΔP controller overrode and decreased the position of the steam valve. Then, if the disturbance rapidly disappeared, the ΔP controller's output became positive (Δm's > 0), but if the composition controller had not yet been upset, its Δm's were zero. Thus, there was no way to recover the former valve position until there was an upset in the column composition. Then, the composition controller recovered the valve position.

With the traditional method of implementation, if a sudden disturbance causes the faster loop to override, then the disturbance disappears and the valve will very quickly recover to the original position, without significant upset to the slower loop.

♦ Pseudo-Velocity Method

One commercial system uses a velocity-mode PID algorithm, then adds the incremental change (Δm) of each controller to a register that represents the previously desired output of that controller. The selector is, in essence, operating between desired positions, not incremental changes. Logic within the selector then adjusts the output of the nonselected controller so it is equal to the output of the selected controller, plus the Gain × Error of the nonselected loop (see Figure 12-13). In essence, then, this system should have the behavior of the "pass-through" method we described earlier.

♦ Selection Based on Error

Other implementers have suggested that the selection criterion be based on the loop errors rather than the loop outputs. The lowest (algebraic) or highest of the error signals, depending on the application, would then be passed to a common PID algorithm. The advantage of this is that there is never a bump in the controller output, as a result of the switchover from one error signal to another. Its disadvantage, however, is that a single set of tuning parameters applies to all of the affected process variables. If there is a slow loop and a fast loop, then a single set of parameters will have to suffice for both.

The author is familiar with a custom implementation that circumvented this problem. In addition to error selection, separate logic determined which error signal was selected and obtained a unique set of tuning parameters from a table for that particular process variable. In essence, this was scheduled tuning. This system reportedly worked quite satisfactorily since the process variables were all relatively noise-free. Furthermore, the disturbances were relatively slow, so all the error signals were near to zero at the time of switchover. Had the "bumpless tuning" provisions described in chapter 4 been utilized, the system would have been immune to problems of switchover even with significant error in the loops.

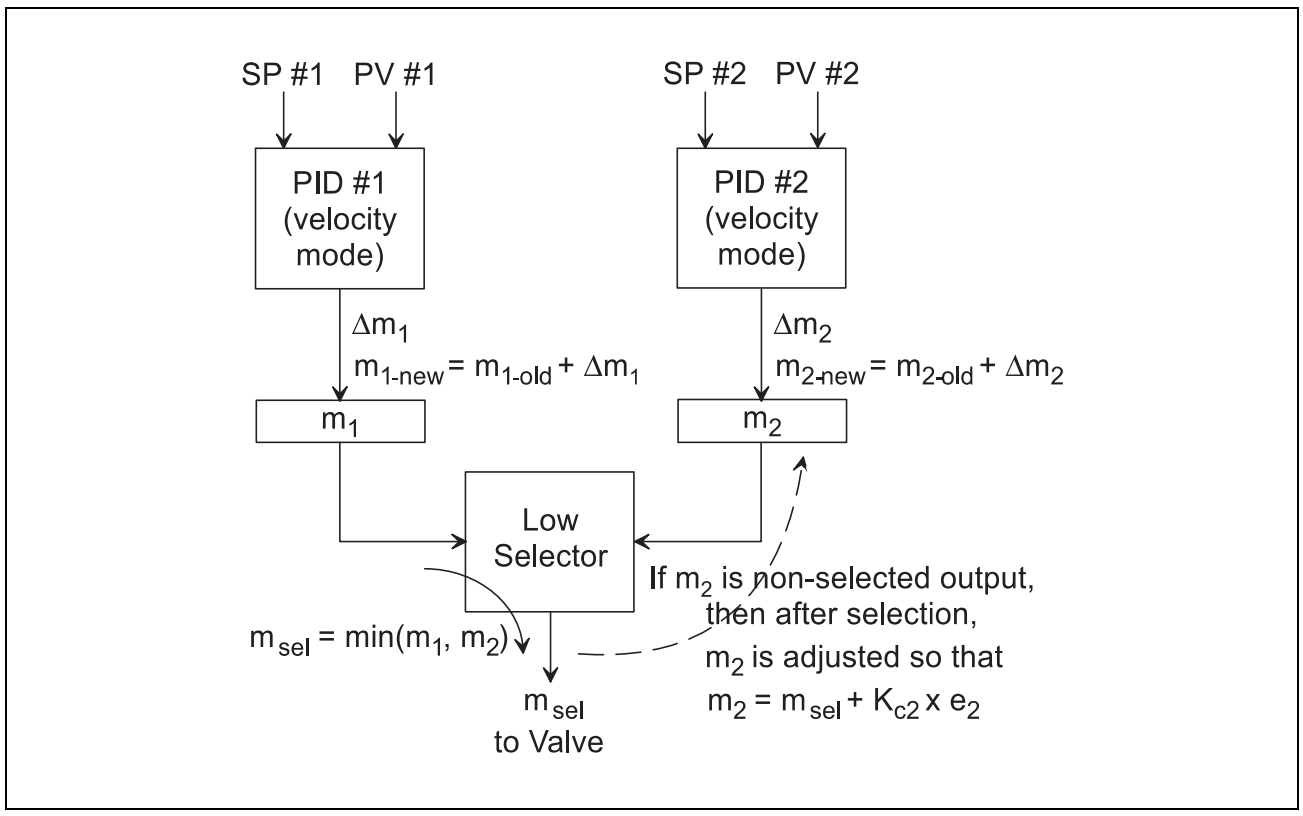

Figure 12-13. Another Implementation of Override Control with Velocity-mode Algorithms

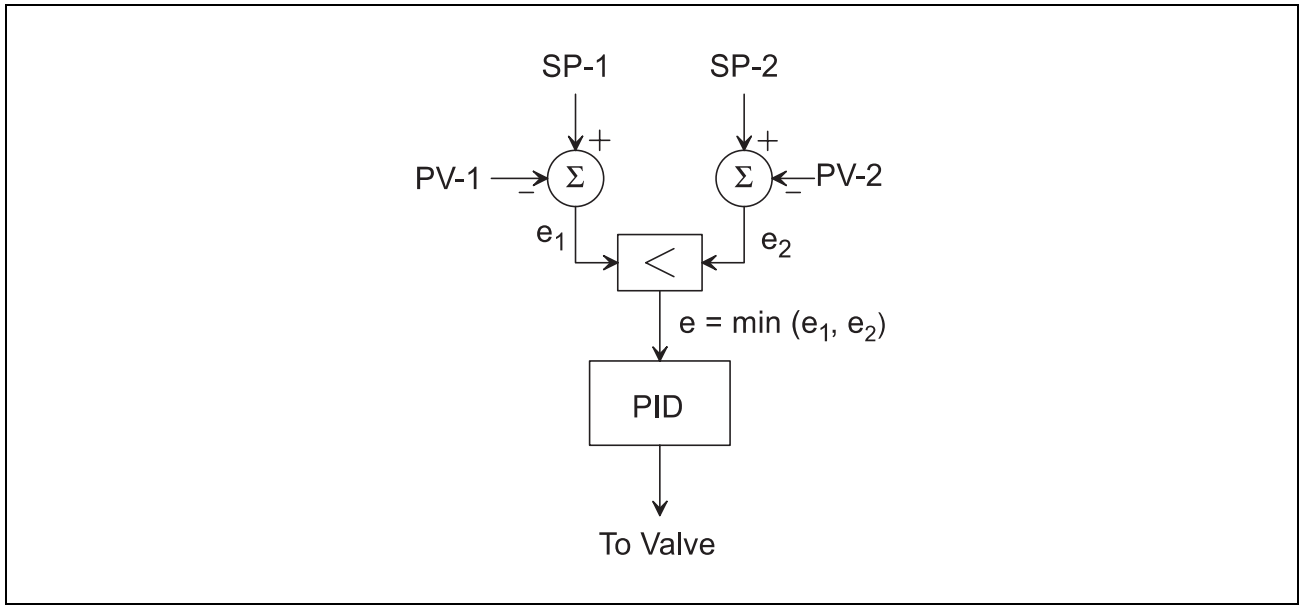

Figure 12-14. Override Control Based on Selection of Errors Rather Than Controller Outputs

❖ OVERRIDE CONTROL USING FOUNDATION™ FIELDBUS

The Fieldbus Foundation standard (Ref. 12-3) defines a control selector (CS) function block. This block is in the control class; hence it supports the back-calculation procedure and is intended for use in override control strategies. The CS block provides for up to three inputs

from other control-class blocks, such as PID. (Nonconfigured inputs are ignored.) The block also provides three BKCAL_OUT signals, one for each of the three inputs. The block can be configured as either a high select or a low select. In the automatic mode, the lower (or higher) of the selected input signals is passed to the output. The status returned to the *nonselected* PID via the back-calculation link is "not selected," and the value is the CS output value, which is the same as the selected controller output. The nonselected controller's output is then made equal to this value. In other words, it is equated to the selected controller's output.

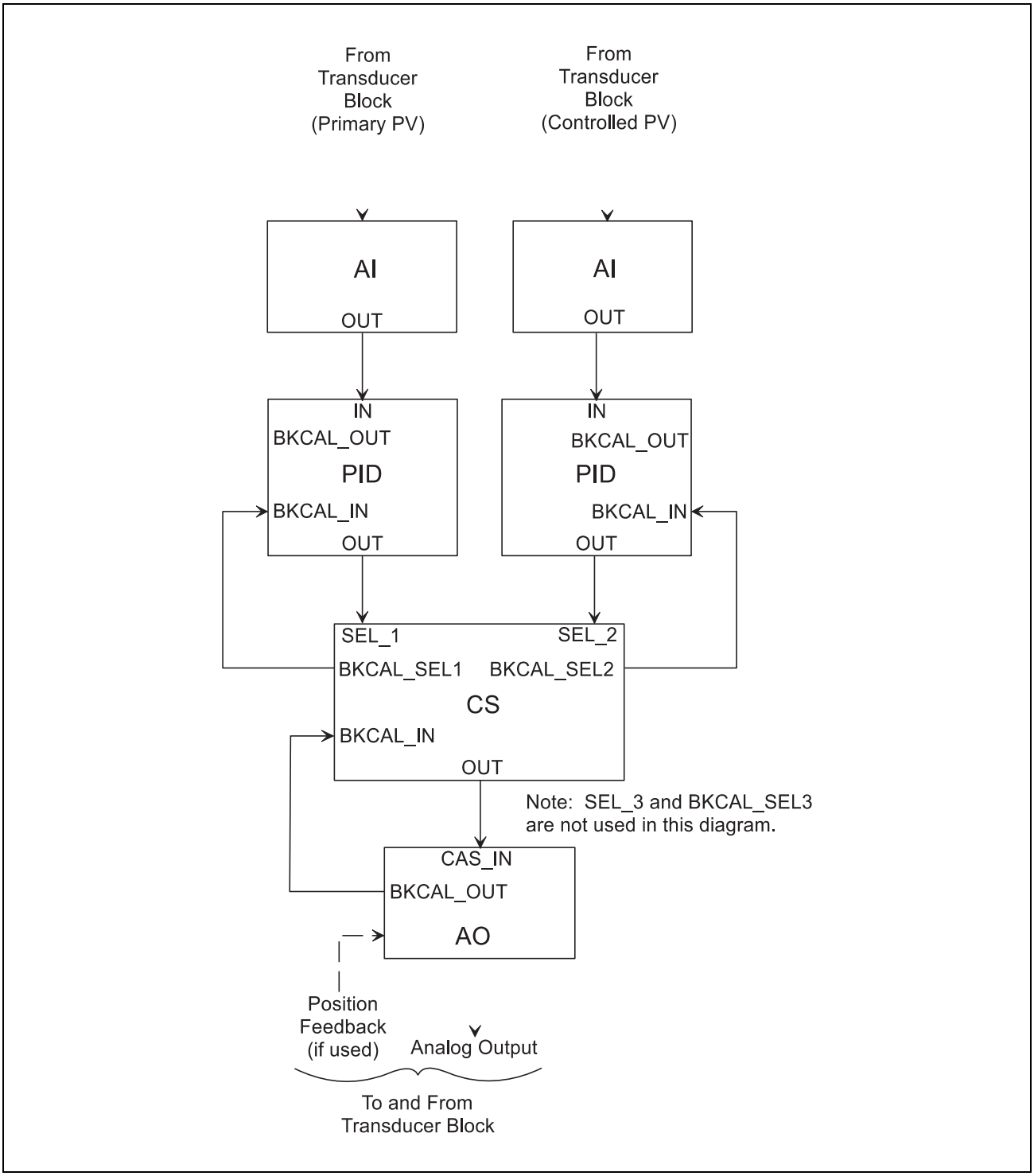

Figure 12-15. Override Control Using FOUNDATION™ Fieldbus Function Blocks

To analyze the behavior of this control scheme, suppose that the CS block is configured for low selection and that the selected controller's PV is at SP. The deviation of the nonselected controller must be such that it is calling for a higher valve position; otherwise, it would have been selected. If process conditions do not change appreciably before the next calculation cycle for each of these blocks, the selected controller will calculate the same valve position as before, but the nonselected controller will again request a higher valve position. The previously selected controller's output will again be selected, and the scenario will repeat.

Suppose, however, that process conditions change, so that each of the controllers requests a new valve position. The CS block will then select the lower of the two new valve positions. Since the controller outputs had previously been forced to be equal, the lower valve position is determined by whichever controller most wants to increment the valve toward closed (or least wants to increment the valve toward open).

In essence, the behavior of this system appears to be similar to the velocity (incremental) mode system we described previously. It may therefore be subject to the same potential problems if the dynamics between the controller output and the two process variables differ greatly, that is, if one PV responds quite rapidly and the other quite slowly.

Some manufacturers may not support the CS block.

❖ REFERENCES

12-1. F. G. Shinskey. *Distillation Control for Productivity and Energy Conservation*, 2d ed., McGraw Hill, Inc., 1984, p. 217.

12-2. Béla G. Lipták, ed. *Instrument Engineers' Handbook: Process Control,* 3d ed., Chilton Book Co., 1995, p. 117.

12-3. Fieldbus Foundation, *Foundation Specification: Function Block Application Process, Document FF-891, Part 2.*