

# **Application Module Control Functions**

**AM09-602**



**Implementation**  
**Application Module - 1**

# ***Application Module*** ***Control Functions***

**AM09-602**  
**Release 620**  
**12/00**

# Notices and Trademarks

**Copyright 2000 by Honeywell International Inc.  
Revision 5 December 1, 2000**

While this information is presented in good faith and believed to be accurate, Honeywell disclaims the implied warranties of merchantability and fitness for a particular purpose and makes no express warranties except as may be stated in its written agreement with and for its customers.

In no event is Honeywell liable to anyone for any indirect, special or consequential damages. The information and specifications in this document are subject to change without notice.

Honeywell and **TotalPlant** are registered trademarks of Honeywell International Inc.

Other brand or product names are trademarks of their respective owners.

Honeywell  
Industrial Automation and Control  
16404 North Black Canyon Highway  
Phoenix, AZ 85053  
**1-800 343-0228**

# About This Publication

This publication supports **TotalPlant** Solution (TPS) system network Release 620. TPS is the evolution of TDC 3000<sup>X</sup>.

This is a reference manual for process engineers, control-system engineers, and application engineers who design and implement data-acquisition and control strategies to be accomplished through a TPS System with a Local Control Network. This publication defines the control functions in the Application Modules (AMs).

This publication is part of a set of publications that define control-system functions. The other members of the set are

- *System Control Functions in the Implementation/Startup & Reconfiguration - 2 binder*
- *Process Manager Control Functions and Algorithms in the Implementation/Process Manager - 1 binder*
- *Hiway Gateway Control Functions in the Implementation/Hiway Gateway - 1 binder.*
- *Logic Manager Control Functions in the Implementation/Logic Manager binder.*
- *Programmable Logic Controller Gateway Control Functions in the Implementation/PLC Gateway binder.*

**Change bars are used to indicate paragraphs, tables, or illustrations containing changes that have been made to this manual effective with release 530.** Pages revised only to correct minor typographical errors contain no change bars.



# Table of Contents

<b>1</b>	<b>REFERENCES</b>
1.1	References
<b>2</b>	<b>AM CONTROL FUNCTIONS</b>
2.1	AM Error Handling
2.1.1	LCN Communication Errors
2.1.2	"Bad PV" Errors
2.1.3	Point Interconnection Errors
2.1.4	AM Detection of Failed Nodes
2.1.5	Communication Errors on General Inputs
2.2	AM Data-Point Processing
2.2.1	Scheduling AM Data Points
2.2.2	Associated Display
2.3	Process Alarms
2.3.1	Process Alarm Detection
2.3.2	Alarm Trip Points
2.3.3	PV Alarm Deadbands in AMs
2.3.4	Advisory Deviation Alarms
2.3.5	Bad-Control Alarm
2.3.6	Bad-Configuration Alarm
2.3.7	Bad-PV Alarm
2.3.8	CL Error Alarms
2.3.9	CL Fail Alarms
2.3.10	Pre-Preset and Pre-Pre-Preset Alarms
2.3.11	Custom Process Alarms
2.3.12	Unit Alarm Operations
2.3.13	Contact Cutout
2.3.14	Changing the Alarm Priorities
2.3.15	Primary Module Points
2.3.16	AM Multiple Primmods Alarming Option
2.3.17	Auxiliary Unit
2.4	Messages
2.4.1	Message Generation
2.4.2	Message Distribution
2.5	External Mode Switching in AMs
2.5.1	Mode Switching
2.5.2	Mode Permissives
2.6	AM System Events
2.7	Restarting AMs
2.7.1	Checkpointing the AM Database
2.7.2	Cold Restart
2.7.3	Warm Restart
2.7.4	Hot Restart
2.7.5	No Point-Processing Restart
<b>3</b>	<b>DATA POINT DESCRIPTIONS</b>
3.1	AM Regulatory Data Points
3.1.1	Functional Structure
3.1.2	AM Regulatory Point Execution States
3.1.3	Processing Order
3.1.4	AM Regulatory Data-Point Interconnections
3.1.5	PV Processing
3.1.6	Setpoint Handling

# Table of Contents

3.1.7	SP/OP Tolerance Check
3.1.8	AM Regulatory Data Point Alarms
3.1.9	Limits in AM Regulatory Data Points
3.1.10	Initialization in AM Regulatory Points
3.1.11	Windup Protection in the AM
3.1.12	Override Control
3.1.13	Control Output Processing in AMs
3.1.14	Functional Summary Chart for AM, and for CB, MC, and EC
3.2	AM Numeric Data Points
3.2.1	AM Numeric Point Functional Description
3.2.2	Limiting PV Range in AM Numeric Points
3.2.3	AM Numeric Point Processing
3.2.4	AM Numeric Point Restarts
3.2.5	AM Numeric Point Configuration
3.2.6	AM Numeric Point Error and Warning Messages
3.3	AM Counter Data Points
3.3.1	AM Counter Point Functional Description
3.3.2	Scheduling of AM Counter Points
3.3.3	Counter Input and PVAUTO Calculation
3.3.4	PV Alarms in AM Counter Points
3.3.5	AM Counter Point Deviation Alarming
3.3.6	Accumulation Processing in AM Counter Points
3.3.7	AM Counter Point Accumulation Alarming
3.3.8	Accumulator Control in AM Counter Points
3.3.9	Other AM Counter Functions
3.3.10	Display- and History-Related Parameters in AM Counters
3.3.11	AM Counter Point Restarts
3.3.12	AM Counter Point Configuration
3.3.13	AM Counter Error and Warning Messages
3.4	AM Flag Data Point
3.4.1	Scheduling of AM Flag Points
3.4.2	AM Flag Point Display- and History-Related Parameters
3.4.3	Off-Normal Alarms in AM Flag Points
3.4.4	AM Flag Point Restarts
3.4.5	AM Flag Point Configuration
3.4.6	AM Flag Point Error Messages
3.5	AM Timer Data Points
3.5.1	AM Timer Point Functional Description
3.5.2	AM Timer Point Time-out Alarm
3.5.3	AM Timer Point Scheduling
3.5.4	AM Timer Point Restarts
3.5.5	AM Timer Point Configuration
3.5.6	AM Timer Point Error and Warning Messages
3.6	AM Custom Data Points
3.6.1	Custom Data Point Identification Data
3.6.2	Custom Data Point Schedule and General Operation
3.6.3	Custom Data Point Alarms
3.6.4	Custom Data Points and CL Segment Data
3.6.5	CL Block Insertion Points in Custom Data Points
3.7	AM Switch Data Points
3.7.1	General Parameters
3.7.2	Switch Parameters
3.7.3	Switch Alarming Functions
3.7.4	CL Block Insertion Points in Switch Data Points



# Table of Contents

## 4 CONTROL-LANGUAGE SUPPORT

- 4.1 CL Execution in the AM
  - 4.1.1 CL's Place in Point Processing
  - 4.1.2 CL's Connection to Data Points
  - 4.1.3 Indirect Reference through Point Identifiers
  - 4.1.4 Data Access and CL Runtime Errors
  - 4.1.5 CL Segments
  - 4.1.6 Background CL Operation
  - 4.1.7 CL Runtime Extensions
- 4.2 CL Switches on AM Regulatory Data Points
  - 4.2.1 CL Switch Data Point Parameters
  - 4.2.2 Changing State Representations in CL Switch Points

# Table of Contents

## INTRODUCTION

### Section 1

*This section provides references to other publications that are useful or necessary in implementing control system functions.*

#### 1.1 REFERENCES

This manual describes the data-acquisition and control functions that reside in Application Modules (AMs). The following are companion publications:

- *System Control Functions* in the *Implementation/Startup & Reconfiguration* - 2 binder.
- *Process Manager Control Functions and Algorithms* in the *Implementation/Process Manager* - 1 binder.
- *Application Module Implementation Guidelines* in the *Implementation/Application Module* - 2 binder.
- *Hiway Gateway Control Functions* in the *Implementation/Hiway Gateway* - 1 binder.
- *Logic Manager Control Functions* in the *Implementation/Logic Manager* binder.
- *Programmable Logic Controller Gateway Control Functions* in the *Implementation/PLC Gateway* binder.

You should be familiar with the content of *System Control Functions* before using the other control functions publications.

Other topics related to data-acquisition and control functions on Data Hiways are provided in these publications:

- Algorithms in Application Modules—*Application Module Algorithm Engineering Data* in the *Implementation/Application Module* - 1 binder.
- Algorithms in Basic Controllers—*CB Algorithm Engineering Data* in the *Product Manual* binder in the BASIC System bookset.
- Algorithms in Multifunction Controllers—*MC Algorithm Engineering Data* in the *Product Manual* binder in the BASIC System bookset.
- Algorithms in Extended Controllers—*EC Algorithm Engineering Data* in the *Product Manual* binder in the BASIC System bookset.

**Parameter Reference Dictionaries**—For details on all of the data-point parameters, including value types, default values, and access levels (keys), refer to the following publications:

*Application Module Parameter Reference Dictionary* in the *Implementation/Application Module* - 2 binder

*PM Family Parameter Reference Dictionary* in the *Implementation/Advanced Process Manager* - 2 binder

*Hiway Gateway Parameter Reference Dictionary* in the *Implementation/Hiway Gateway* - 1 binder

*Computer Gateway Parameter Reference Dictionary* in the *Implementation/CM50S-1*, and *Implementation/Computer Gateway* binders.

*Programmable Logic Controller Gateway Parameter Reference Dictionary* in the *Implementation/PLC Gateway* binder.

*Logic Manager Parameter Reference Dictionary* in the *Implementation/Logic Manager* binder.

The following sections of the *Engineer's Reference Manual* in the *Implementation/Startup & Reconfiguration* - 2 binder have information related to AM control functions:

Section 21, Checkpointing

Section 22, Processor Status Data Point

Section 23, Configuring for Performance

## AM CONTROL FUNCTIONS

### Section 2

*This section defines the data-acquisition and control functions, other than data-point functions, that are accomplished in AMs. The data-point functions are defined in Section 3.*

#### 2.1 AM ERROR HANDLING

This subsection defines how the AM handles these conditions:

- **LCN Communication Errors** (2.1.1)
- **"Bad PV" Errors** (2.1.2)
- **Point Interconnection Errors** (2.1.3)
- **Detection of Failed Nodes** (2.1.4)
- **Communication Error on General Inputs** (2.1.5)

##### 2.1.1 LCN Communication Errors

These are errors detected in communications on the Local Control Network. They are most likely associated with the failure of a module or gateway. When a communication error is detected, processing of all data points is delayed for one cycle (0.5 sec.). If the error persists for the next cycle, the value status of the data points that are affected changes to bad, and point processing, even with the bad data, continues. In this situation, the values in the parameters with bad data can't be changed and a system alarm is generated.

##### 2.1.2 "Bad PV" Errors

Figure 2-1 is a flow chart of the handling of seven conditions that can cause bad PVs. The following is a summary of each of these situations:

- **CVB Overflow in the AM**—This error occurs if the current-value buffer (CVB) in the AM is full and can accommodate no more prefetched or poststored data (see 2.3 in *Application Module Implementation Guidelines*).
- **LCN Communication Error**—The PV value status for a point has changed to Bad because of an LCN communication error, as described under 2.1.1.

- **UAC Switchover**—The PV value status for a point has changed to Bad because an Uninterruptible Automatic Control (UAC) switchover on a Data Hiway has occurred, and the AM was fetching the PV during the switchover (a UAC switchover is the switching of control from a CB, MC, or EC to its reserve controller).
- **Node Failure**—The PV value status for a point has changed to Bad because the node the value was fetched from has failed.
- **Intermittent Communication Error**—A communication error has occurred that appears to be because a temporary box-level communication error occurred. This type of error doesn't cause an alarm, because of its "immediate" recovery; however, an AM point that receives the value during the intermittent condition receives a bad input.
- The **value** in parameter **PVAUTO** is **out of range** and is not clamped.
- The **value** for a required **PV-Algorithm coefficient** is **bad**.
- The **value** for a required **PV input** is **NaN**.

Note that the flow chart shows that if the PVSOURCE parameter for the point with the bad PV contains Auto, bad-control alarms for this point are not distributed. If PVSOURCE contains Man (manual) or Sub (substituted PV), the bad-control PV alarm is distributed.

### 2.1.3 Point Interconnection Errors

The point interconnection-error categories are as follows:

- Software error
- Communication error
- Configuration error
- Value storage failed
- Value stored with an error
- Range error
- Key-level error

When such an error is detected an appropriate message is displayed at the Universal Station(s).

#### 2.1.3.1 Software Errors

Software errors should be very rare, but if they occur, you will need assistance from Honeywell. When a software error is detected, the module in which it was detected, is automatically shut down.

### 2.1.3.2 Configuration Errors

These are caused by a missing data point or a missing parameter, or caused by an incorrect parameter or some other incompatibility (including off-LCN data access security violations). When a configuration error occurs, an attempt to read a missing parameter results in a bad-value status (except for general input connections, see 3.1.4.4) and an attempt to write in a missing parameter is inhibited. In both cases, a configuration alarm is generated.

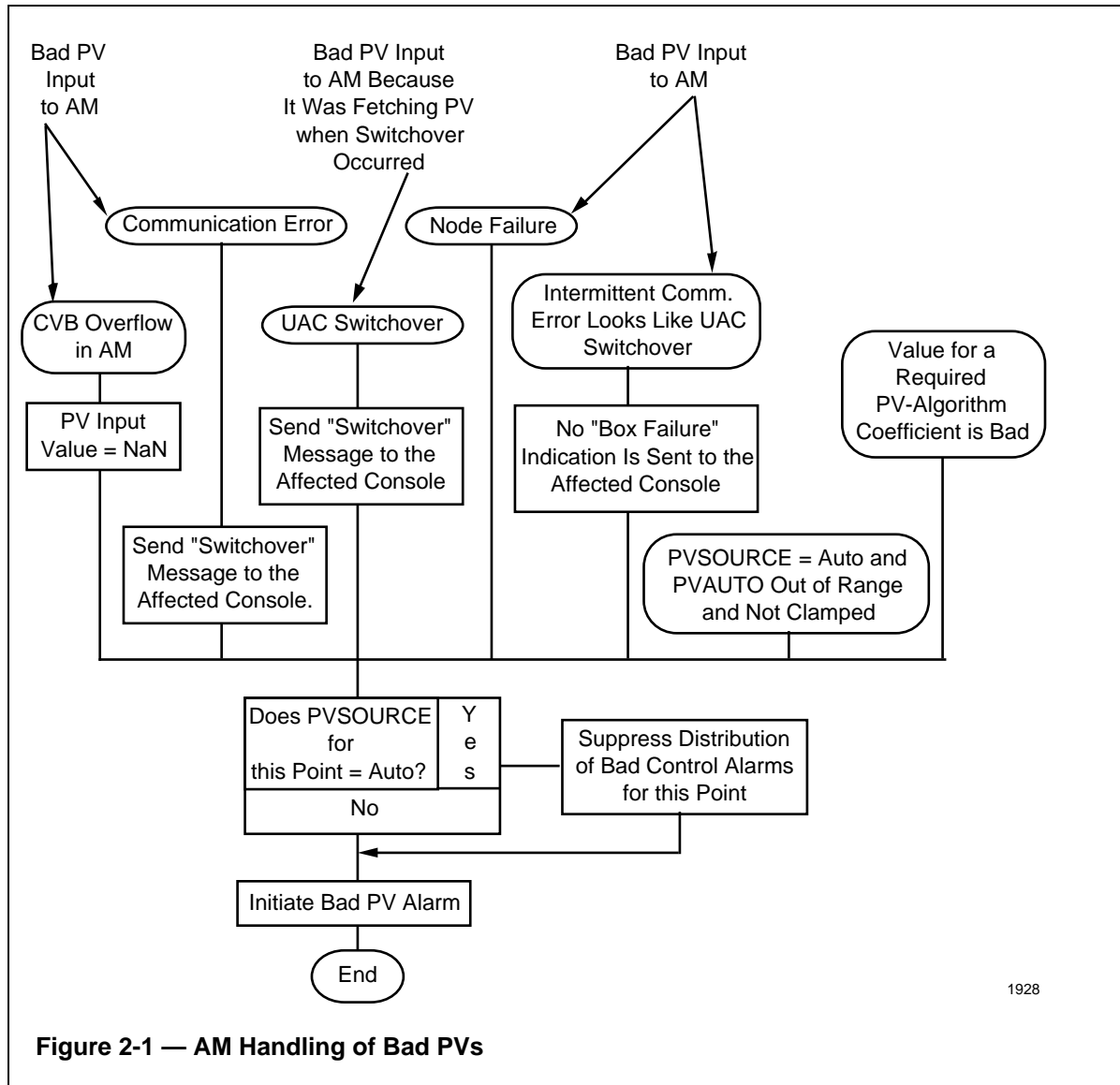


Figure 2-1 — AM Handling of Bad PVs

### **2.1.3.3 Value Storage Failure**

When an attempt to write in a parameter fails because of control interlocks, an error indication is generated. Examples of such interlocks are that the point is in the wrong mode, has an incorrect PVSOURCE, or is being initialized. In general, standard control functions don't act on such an error. The error indication is accessible to CL programs, and corrective action could be taken by a CL program.

### **2.1.3.4 Value Storage With an Error**

When an attempt to write in a parameter is successful but the data was changed in some way, such as by clamping action, an error indication is generated but, in general, standard control functions don't act on such an error. The error indication is accessible to CL programs, and corrective action could be taken by a user-written CL program.

### **2.1.3.5 Range Error**

A range error is caused by an attempt to store outside the system-specified value or subscript range.

### **2.1.3.6 Key-Level Error**

A key-level error is caused by an attempt to store into a parameter that cannot be written to with the access key presented.

### **2.1.3.7 Bad Value Handling at Point Interconnections**

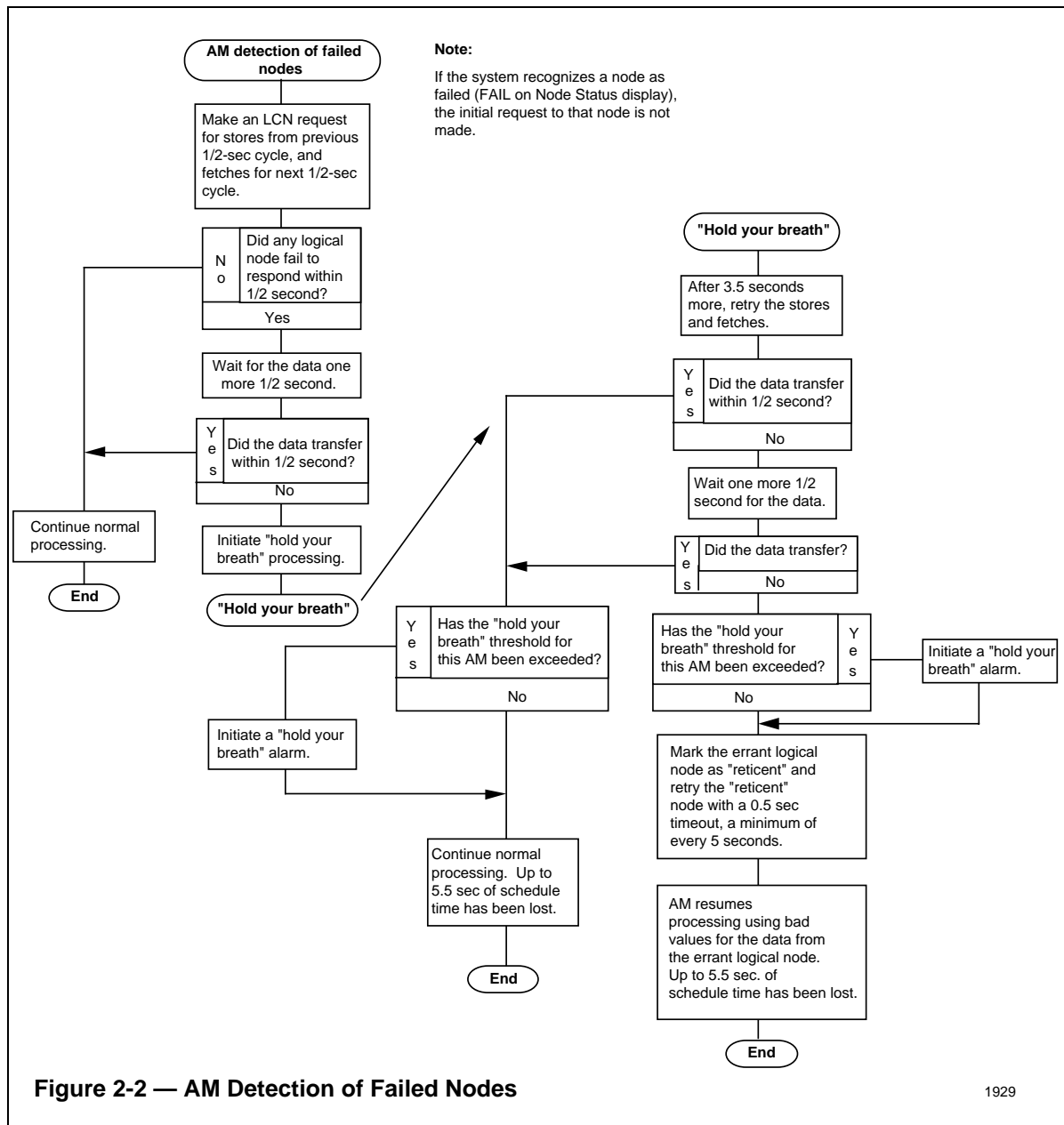
When a value in a parameter is NaN, the value status transferred over input and output connections is "bad" (except for general input connections, see 3.1.4.4). Also,

- Input connections do transfer bad values (except for general input connections, see 3.1.4.4).
- Output connections don't transfer bad values.

## **2.1.4 AM Detection of Failed Nodes**

The flow chart on Figure 2-2 shows how the AM detects a failure of another node (module or gateway) that the AM communicates with as it is processing data points. If such a node fails to provide or to accept data for a full second, the AM assumes a node failover is occurring, and stops processing long enough for failover to occur (4.5 seconds) before retrying the request. This suspension of point processing can be referred to as the AM "hold your breath" philosophy, and results in accepting a permanent 5 to 5.5-second slip in the point processing schedules. No attempt is made by the point processors to make up for this lost time.





**Figure 2-2 — AM Detection of Failed Nodes**

1929

The "hold your breath" path in the flow chart shows that if this path has been taken more than a system-specified number of times, a "hold your breath" alarm is distributed. On each pass, the AM attempts for a total of 4.5 seconds to recover the data transfers and if the data doesn't successfully transfer in that period, the AM resumes processing with bad-value status for any data needed from the failed node.

### 2.1.5 Communication Errors on General Inputs

If a communication error occurs when a general input to an AM point is fetched, the general input isn't stored in the point, the status for this general input goes to Commerr, but no alarm is distributed for this condition.

## 2.2 AM DATA-POINT PROCESSING

AM data points are processed at configured intervals from once each second to once each day, thus tailoring the processing rate to the dynamics of the process and allowing a reasonably even distribution of the processing load.

AM points can also be processed as demanded.

### 2.2.1 Scheduling AM Data Points

To schedule a AM regulatory data point, do the following as you configure the point:

1. **Assign a scheduling period** from the periods that appear on the point-builder display, **or configure event-driven processing**. Events can include an operator request, a request from a user-written program, or a request from another data point. Unless you have specified that the point is to be processed by the internetwork (IPP) processor, the scheduling period that you select will determine to which processor the point is assigned. Selection of 1 second through 2 minutes selects the fast-point processor (FPP). Selection of 1 "slow" minute (1S Min) through 24 hours selects the slow-point processor (SPP).
2. You can specify that the point be processed **before or after** a point that has already been scheduled. Note that before/after processing between points assigned to different processors is not allowed.
3. You can assign the point to a **processing cycle**, or you can allow the system to assign the cycle on which the point executes. Points that have not been assigned to be processed before or after another point, and that have no user-configured cycle assignment, are automatically assigned to the least-loaded processing cycle.

The processing-cycle assignment and before/after assignment are mutually exclusive: either one or the other can be configured for a point, but not both.

CL programs can schedule points configured for **event-driven processing** to be processed at a relative time, such as "five seconds from now."

### 2.2.1.1 Data Point Processors

Application Modules have three data-point processors: the fast processor (FPP), the slow processor (SPP) and the internetwork processor (IPP). The fast processor has a higher priority than the other two processors, and these processors can consume up to 70% of each processing cycle. The remaining time is available for Background CL programs and other AM processing functions.

The internetwork processor can be used for access of data on remote LCNs through the Network Gateway. (Note that background CL programs can access remote data regardless of the point's processor assignment.) Points can only be assigned to the internetwork processor if IPP has been enabled during configuration of that AM. IPP points do not have overlapped prefetch/poststore processing. The FPP/SPP processors process cycle  $n$  while fetching for cycle  $n+1$  and storing for cycle  $n-1$ . The IPP processes cycle  $n$ , and then waits for both the cycle  $n$  stores and the cycle  $n+1$  fetches to complete before it processes cycle  $n+1$ .

Because of its higher priority, points assigned to the fast processor are processed more nearly on schedule than those assigned to the other two processors. Points assigned to the fast processor are scheduled to execute within a given one-half second. Points assigned to the slow processor are scheduled to execute on a particular one-half second within a given minute. A 5-second leeway is allowed for actual execution of the point from the assigned one-half second.

IPP processing and data accessing is independent from the FPP/SPP and is considered less time critical. Note that timer and counter data points on the IPP cannot be expected to count time accurately. Use FPP/SPP instead.

Point-processor loading is effectively balanced across the minute. Thus, if there are 120 one-minute points assigned to the slow processor, each one-half second one of these is executed, so that by the end of the one-minute period, all 120 points have been executed. The 8-, 12-, and 24-hour points are scheduled to execute within the first minute of the appropriate hourly period, as specified by the cycle assignment.

When the fast processor has work to do, it interrupts the SPP or IPP after processing for the current point is complete (background CL programs are interrupted immediately). Once one of the processors begins processing a point, it continues to completion. Thus, if a point with a large foreground CL block is assigned to the slow processor, it could cause the fast processor to be delayed. If a processor is delayed, it tries to catch up.

Parameters are available in the AM processor-status point that can be configured to cause the AM to generate an alarm (FPP/SPP/IPP) or to fail (FPP/SPP only) if a processor is excessively delayed (see Section 22 in the *Engineer's Reference Manual*).

### 2.2.1.2 Period Assignments and Special Processing

Data points processed by the data point processors are assigned to processing periods. The processing periods are as follows:

- **Fast Processor** – 1 Sec, 2 Sec, 5 Sec, 10 Sec, 15 Sec, 30 Sec, 1 Min, and 2 Min.
- **Slow Processor** – 1S Min, 2S Min, 5 Min, 10 Min, 15 Min, 30 Min, 1 Hr, 8 Hr, 12 Hr, and 24 Hr.
- **Internetwork Processor** – 5 sec, 10 Sec, 15 Sec, 30 Sec, 1 Min, 2 Min, 5 Min, 10 Min, 15 Min, 30 Min, 1 Hr, 8 Hr, 12 Hr, and 24 Hr.

The periods on the fast and slow processors overlap each other (1 Min is on the fast processor and 1S Min is one minute on the slow processor), thus, providing some flexibility in period assignments. The accuracy of the actual processing period depends on the processor to which the point is assigned and the time when the preceding point was processed.

Points are processed within timing tolerances that assure the accuracy of dynamic control calculations by algorithms such as PIDs and Lead-Lag algorithms.

Special processing can be requested by US operators, user-written programs, and requests from other points. When an operator requests special processing from the Detail Display for a point, the request is accomplished by placing the value On in parameter PPS. A user-written program can request special processing by placing Normal in parameter PPSREQ, or it can request special processing at some later time by placing a delay time in seconds in parameter PPSCYCLE (if a CL block requests special processing for the point it is attached to, such processing takes place after the next normal processing cycle). Parameter PPSTYPE indicates the type of any special processing request currently in effect for a point. Refer to the *Application Module Parameter Reference Dictionary* for more detailed information about these parameters.

There are two categories of special-processing requests:

- **Full processing**—initiated by requests to parameters PPS, PPSREQ, or PPSCYCLE.
- **Partial processing** that occurs for these specific functions (PPSTYPE indicates which)

Process-Special Initialization—Execute the set of point functions involved in changing cascade connection or mode.

Process-Special Override—Execute the set of functions involved in override processing.

Process-Special Manual Mode—This Process Special is invoked when a operator

changes OP while in MANual mode.

These are internal functions and they occur as a result of some other activity, such as an operator changing the OP value while in MANual mode.

### 2.2.1.3 Automatic Cycle Loading

After selecting the processing cycle, you can let the system determine the least-loaded cycle on which to schedule the point for processing. This option is selected when you build an AM point, by entering "No" in parameter BEFAFT, or by entering "Cycle" in BEFAFT and entering -1 in NORMCYCL.

The system maintains loading information in a loading array that consists of 240 "load intervals" for points with fast periods, 60 load intervals for points with slow-minute periods (1S Min through 1 hr), and 24 load intervals for points with slow-hourly periods (8 hr, 12 hr, and 24 hr). Each load interval contains the number of points scheduled to process for all periods whose cycles are associated with that load interval. For example, slow-hourly load interval number 20 contains the number of points scheduled with an 8-hr period on cycle 4, a 12-hr period on cycle 8, and a 24-hr period on cycle 20. Refer to the following chart:

Load Interval Points Loaded on the Interval Per Cycle			
0	8 hr cycle 0	12 hr cycle 0	24 hr cycle 0
1	8 hr cycle 1	12 hr cycle 1	24 hr cycle 1
2	8 hr cycle 2	12 hr cycle 2	24 hr cycle 2
3	8 hr cycle 3	12 hr cycle 3	24 hr cycle 3
4	8 hr cycle 4	12 hr cycle 4	24 hr cycle 4
5	8 hr cycle 5	12 hr cycle 5	24 hr cycle 5
6	8 hr cycle 6	12 hr cycle 6	24 hr cycle 6
7	8 hr cycle 7	12 hr cycle 7	24 hr cycle 7
8	8 hr cycle 0	12 hr cycle 8	24 hr cycle 8
9	8 hr cycle 1	12 hr cycle 9	24 hr cycle 9
10	8 hr cycle 2	12 hr cycle 10	24 hr cycle 10
11	8 hr cycle 3	12 hr cycle 11	24 hr cycle 11
12	8 hr cycle 4	12 hr cycle 0	24 hr cycle 12
13	8 hr cycle 5	12 hr cycle 1	24 hr cycle 13
14	8 hr cycle 6	12 hr cycle 2	24 hr cycle 14
15	8 hr cycle 7	12 hr cycle 3	24 hr cycle 15
16	8 hr cycle 0	12 hr cycle 4	24 hr cycle 16
17	8 hr cycle 1	12 hr cycle 5	24 hr cycle 17
18	8 hr cycle 2	12 hr cycle 6	24 hr cycle 18
19	8 hr cycle 3	12 hr cycle 7	24 hr cycle 19
20	8 hr cycle 4	12 hr cycle 8	24 hr cycle 20
21	8 hr cycle 5	12 hr cycle 9	24 hr cycle 21
22	8 hr cycle 6	12 hr cycle 10	24 hr cycle 22
23	8 hr cycle 7	12 hr cycle 11	24 hr cycle 23

To determine the least-loaded cycle to which the point should be assigned, the system compares the load intervals for the point's unit and period group (fast periods, slow-minute periods, or slow-hourly periods). To select the cycle within the period (for example, cycle 0, 1, 2, or 3 of a 2-sec. period), a trial calculation is made to determine the maximum interval load associated with that particular cycle, by comparing the load intervals in which the point would be scheduled. The cycle that corresponds to the minimum of these trial loads is selected as the least-loaded. For example, a point with a 1-sec. period can be assigned to cycle 0 or to cycle 1. The maximum loading of the 120 even-numbered load intervals is compared to the maximum loading of the 120 odd-numbered load intervals, and the cycle that corresponds to the minimum of these two numbers is selected.

In assigning a point to a cycle, the algorithm that determines the least-loaded cycle considers the following variables:

- This point's unit, only (loads for other units on the same processing cycle are ignored).
- The number of points in a period group (fast periods, slow-minute periods, or slow-hour periods) (point processing times and loads in other period groups are ignored).

#### **2.2.1.4 Before and After Assignments**

Data points that require a certain relative processing order can be assigned a before/after relationship. You can configure either a before or an after relationship to establish the order of point processing. Such order is important when a point needs up-to-the-moment data from another point.

Points that have before/after assignments are processed in the specified order in the same one-half second if on the fast process or in the same 5-second period if on the slow processor. If point A is before point B, and point B is before point C, then points A, B, and C are processed in A, B, C order.

Such schedules are used in cascade control strategies to assure proper propagation of information. Two or more points with before/after relationships to each other are defined as a "before/after set."

Here are some restrictions on such schedules:

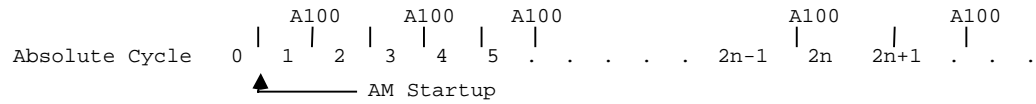
1. All points within a before/after set must be assigned to the same processor (FPP, SPP, or IPP).
2. The processing period of every point in a before/after set must be some multiple of the period of the fastest point in the set.
3. Each before/after set must be in the same process unit.
4. Where point A is configured to name point B to be processed before or after point A, point B must be loaded in the AM before point A is loaded. Points in before/after sets are loaded by the Data Entity Builder's Load Multiple command. This command and establishing the loading order with a selection list are defined under 7.2.4 in the *Data*

5. All points in a previously loaded before/after set must be rebuilt if any of the set is rescheduled.

### 2.2.1.5 Cycle Assignment by Users

In some cases you may want to load processing cycles yourself. To do this you need to understand what cycles are available for each processing period.

On the fast processor, each cycle effectively occurs at each half-second, thus, for the 1-sec. period there are two cycles: 0 and 1. For the 2-min. period, there are 240 cycles (0 through 239). The other processing periods follow this pattern. If you assign a point to cycle 1 of a 1-sec. period, that point will be executed every other cycle, starting with the second half-second after the AM starts up. For example, if point A100 is assigned to cycle 1 of a 1-sec. period, it is processed as indicated by this diagram:



4996

Fast processor cycle assignments are as follows:

Period	Cycles
1 Sec	0, 1
2 Sec	0, 1, 2, 3
5 Sec	0, 1, . . . , 9
10 Sec	0, 1, . . . , 19
15 Sec	0, 1, . . . , 29
30 Sec	0, 1, . . . , 59
1 Min	0, 1, . . . , 119
2 Min	0, 1, . . . , 239

Slow processor cycle assignments are as follows:

Period	Cycles
1S Min	0
2S Min	0, 1
5 Min	0, 1, 2, 3, 4
10 Min	0, 1, . . . , 9
15 Min	0, 1, . . . , 14
30 Min	0, 1, . . . , 19
1 Hr	0, 1, . . . , 59
8 Hr	0, 1, . . . , 7
12 Hr	0, 1, . . . , 11
24 Hr	0, 1, . . . , 23

Internetwork processor cycle assignments are as follows:

Period	Cycles
5 Sec	0
10 Sec	0, 1
15 Sec	0, 1, 2
30 Sec	0, 1, . . . , 5
1 Min	0, 1, . . . , 11
2 Min	0, 1, . . . , 23
5 Min	0, 1, . . . , 4
10 Min	0, 1, . . . , 9
15 Min	0, 1, . . . , 14
30 Min	0, 1, . . . , 29
1 Hr	0, 1, . . . , 59
8 Hr	0, 1, . . . , 7
12 Hr	0, 1, . . . , 11
24 Hr	0, 1, . . . , 23

#### 2.2.1.6 Handling of Processing Overloads

If a point processor can't complete the processing of all of the points on a processing cycle, processing of all the points in the overloaded cycle is completed and the AM tries to catch up in succeeding cycles.

If the number of late cycles (indicated by processor status data-point parameter **OVERRUNS**) exceeds the threshold established by the user in processor status data-point parameter **AMOVRTHR**, a processor status-point alarm is generated. When normal processing resumes, a return-to-normal message is generated. (The processor-status data point is discussed Section 21 in the *Engineer's Reference Manual*.)

If processing is late by more than the number of cycles specified by the user in processor status data-point parameter **AMOVRABT**, the AM fails.

You can select an alternative overrun-handling method. To do so, configure a CL block on a custom data point to monitor the overrun parameters, **OVERRUNS**, **OVRFASTC**, and **OVRSLOWC**, and an associated CL block to deactivate selected points to reduce the overload.

When behind, the IPP uses a technique for catching up that processes some "next cycle" points on the current cycle. This can result in the skipping of duplicate executions of higher frequency points. The difference between the number of scheduled cycles and the number of actual "fetch" cycles represents the count of skipped executions during catchup periods. The point's PSDP parameters **IPPRCYCC** and **IPPRCYCP** give this information.



### **2.2.1.7 AM Schedule Dumper**

You can use the AM schedule dumper to print or display a report that provides cycle-loading and schedule information of the points in an AM unit. To do so, refer to subsection 5.1 in *Application Module Implementation Guidelines*.

### **2.2.2 Associated Display**

With Release 510 and later software, an Associated Display can be configured for each AM point by entering the name of a custom built schematic in the ASSOCDSP parameter. The name of the custom schematic can be up to 8 characters long. At operating time, by pressing the ASSOC button, the associated custom schematic can be called up from a Point Detail Display, Group Display with a point selected, or any Summary Display with a point selected.

The ASSOCDSP parameter can be defined in the DEB at build time, or it can be changed from the DEB's Alter\_ Parameter function, the Detail Display, Schematics, or CL/AM. The Access Lock for the ASSOCDSP parameters is Engineering.

## **2.3 PROCESS ALARMS**

See *System Control Functions* for general discussions of process alarms. In addition to the alarm types described below, the AM also detects the alarm types defined in *System Control Functions*.

### **2.3.1 Process Alarm Detection**

Process Alarm detection occurs as individual data points are processed in normal point processing.

Each appropriate, process-related parameter is checked to determine if it is within user-configured alarm trip points, and the corresponding alarm flag is set or reset as the result of the test. Additionally, in AM Regulatory points, a 3-state alarm-transition parameter value is derived for most analog variables that are subject to alarm checking. The three states are Alarm, Rtn, and NoChange. The values of the transition parameters are meaningful only during the point's execution and it is specifically intended for the use of CL blocks on the same point.

### 2.3.2 Alarm Trip Points

Alarm trip points are user-configured values. When a monitored variable exceeds the trip point, an alarm is generated.

In AMs, you can configure the trip point as NaN—not a number. When an alarm trip point is defined as NaN, the trip-point test is not performed, the corresponding alarm flag is changed to false, and the alarm-transition flag is changed to "no transition."

### 2.3.3 PV Alarm Deadbands in AMs

For Regulatory data points and Counter data points in AMs, a deadband is configured for each data point. The deadband is specified as a percentage of the PV range or as an engineering-unit value. The deadbands reduce nuisance alarms that can occur when a PV varies around the trip point.

Parameter PVALDB is configured with one of the following values: Half, One, Two, Three, Four, Five, or Eu. The values Half through Five specify the deadband in percent of the engineering-units range. The value Eu indicates that parameter PVALDBEU contains the deadband value in engineering units.

The deadband effects the return-to-normal. Alarms are generated when the PV increases through the trip point and the return-to-normal indication occurs when the PV returns through the trip point and the deadband. Deadband values default to 1.0%.

### 2.3.4 Advisory Deviation Alarms

An advisory-deviation alarm is generated when the absolute value of the difference between the PV and the advisory setpoint exceeds the advisory-deviation trip point. The advisory-deviation alarm return-to-normal is generated when the absolute value of the difference becomes less than the trip point plus the deadband value. The alarm priority parameter for the advisory-deviation alarm is ADVDEVPR. See 3.1.6.3 and 3.1.7.

### 2.3.5 Bad-Control Alarm

A bad-control alarm is generated when control processing results in  $CV = NaN$  (CV is the calculated value from control processing). This occurs if any control input or PV is NaN. When the NaN input returns to normal, the point is initialized and a "bad-control return-to-normal" indication is generated. The alarm priority parameter for this alarm is BADCTLPR, and it is available on Counter and Regulatory points.

### 2.3.6 Bad-Configuration Alarm

In AMs, a single bad-configuration alarm is generated for all reported configuration, software, key-level, and range errors for process inputs, general inputs, control inputs, and control outputs. The alarm priority parameter for this alarm is CNFERRPR, and it is available on Custom, Regulatory, Switch, Timer, and Counter points.

### **2.3.7 Bad-PV Alarm**

When the PV input to a data point is NaN, a bad-PV alarm is generated and other PV-related alarms are not detected. Until the PV returns to normal, the last state is maintained and the alarm transition flags are in the no transition state.

When a valid PV is again available, a bad-PV return-to-normal indication is generated and normal alarm computation is resumed, using the prealarm alarm status as a reference in generating return-to-normal reports and alarm transition flags. The alarm priority parameter for the bad-PV alarm is BADPVPR, and it is available on Counter and Regulatory points.

Other aspects of the AM's handling of bad PVs is provided under 2.1.2.

### **2.3.8 CL Error Alarms**

A CL error alarm is generated when a Foreground CL program encounters a configuration error that probably will require an "engineer" fix, using the CL compiler or DEB. See heading 4.1.4.7. The Background CL error alarm indicates the same condition on a Background CL program. The Foreground CL and Background CL error alarm priority parameters are CLEALMPR and BCLEALPR respectively. They are available on Custom, Regulatory, and Switch points.

### **2.3.9 CL Fail Alarms**

A CL fail alarm is generated when a Foreground CL program cannot proceed because of a communications error or a bad value. See heading 4.1.4.7. The Background CL fail alarm indicates the same condition on a Background CL program. The Foreground CL and Background CL fail alarm priority parameters are CLFALMPR and BCLFALPR respectively. They are available on Custom, Regulatory, and Switch points.

### **2.3.10 Pre-Preset and Pre-Pre-Preset Alarms**

These alarms apply to only AM counter data points. A pre-preset alarm is generated when the accumulation value is equal to the preset (accumulation target) value, minus a user-defined accumulation deviation-one target value.

A pre-pre-preset alarm is generated when the accumulation value is equal to the accumulation-target value minus a user-defined accumulation deviation-two target value.

The alarm priority parameter for the pre-preset alarm is PRPRSTPR, and the alarm priority parameter for the pre-pre-preset alarm is PPPRSTPR. See 3.3.7 for additional information.

### 2.3.11 Custom Process Alarms

AM switch data-points provide parameters that can be used with CL/AM programs to create custom process alarms. These parameters are SALMFL1, SALMFL2, and SALMFL3, and the corresponding alarm priority parameters are SWALM1PR, SWALM2PR, AND SWALM3PR. See 3.7.3 for additional information.

### 2.3.12 Unit Alarm Operations

All alarms for all data points in a process unit can be enabled, disabled, or inhibited at once, by a Universal Station operator. These unit-based alarm-state changes override the enable/disable/inhibit states for the individual data points.

### 2.3.13 Contact Cutout

Contact cutout is a state in an alarmable point in which alarms are “cut out” (not reported). The main purpose of the contact cutout function is to prevent a proliferation of alarms from being reported to the operator, which could occur when a critical failure causes a number of related alarms in other points. In the AM, the contact cutout state is determined by the CONTCUT parameter, which is available in all alarmable points (all points except numeric). This parameter is a logical type with a values of On or Off. This parameter can be written to from CL to turn the contact cutout state on or off. The CONTCUT parameter cannot be accessed from the Data Entity Builder (DEB) or from schematics.

The AM regulatory, counter, and timer points also have a configured contact cutout input connection that can control CONTCUT and thereby turn the contact cutout state on or off. This involves two parameters, CCINPT and CCSRC. CCINPT (Contact Cutout Input required) is a Yes/No enumeration which, if set to Yes, causes a data entry box to appear on the Parameter Entry Display of the DEB, requiring the entry of the point.parameter CCSRC (Conatact Cutout Source). The source point.parameter entered here must be of type logical (On/Off). As an example, the Point-in-Alarm paramater (PTINAL) of a point (primary) could be entered as the Contact Cutout Source (CCSRC) in one or more other points (secondaries).

#### 2.3.13.1 Universal Personality Function

An NCF option (new in R500) allows the user to select how existing unacknowledged alarms are handled on the US Alarm Summary display when a point goes into the contact cutout state. A choice of two options is available.

- **CLEAR IMMEDIATELY**—Clears all of the points unacknowledged alarms from the Alarm Summary.
- **CLEAR WHEN ACKED**—Leaves unacknowledged alarms on the Alarm Summary, but backlights the time stamp. Clears the alarms when they are acknowledged by the operator. This is the default conditon.

When a point goes into the contact cutout state, the following actions take place:

- All of the point's acknowledged alarms are cleared from the Alarm Summary display.
- Depending on the NCF alarm summary option selected, one of the following happens:
  - All of the point's unacknowledged alarms are cleared, or
  - The time stamps on all of the point's unacknowledged alarms are backlighted, and the alarms are cleared when they are acknowledged by the operator.
- Any new alarms detected are not displayed on the Alarm Summary display. Cutout alarms are not journaled, but continue to be reported to the AM for event initiated processing (EIP). A Contact Cutout True message is journaled for points in alarm.

When contact cutout is removed and the point is in alarm, the AM will redistribute the alarm with the following results:

- If the alarm is displayed on the Alarm Summary display, the backlighting is removed from the time stamp of the alarm, and the time stamp is changed to the time that contact cutout was removed. If the alarm is not displayed on the Alarm Summary, a new alarm message is displayed.
- A Contact Cutout False event is journaled.

When a point returns to normal from an alarm while in the cutout state, and then goes out of the cutout state, the following actions take place:

- Backlighting is removed from the time stamps of the point's unacknowledged alarms on the Alarm Summary, and the priority indicator is backlighted indicating that the point returned to normal. The alarms will be cleared when acknowledged by the operator. (The priority indicator is a single character (E, H, or L), or optionally a symbol, that indicates whether the alarm was Emergency, High, or Low. It is located in the column just to the right of the time stamp.)
- A Contact Cutout False event is journaled.

### **2.3.14 Changing the Alarm Priorities**

Prior to R620, alarm priorities on scheduled AM point types such as Regulatory, Custom, Timer, Counter, and Switch could not be changed when the point was active. With R620, the alarm priority of these AM point types can be changed while the points are active. This is also applicable to points whose PERIOD is set to NO PERIOD.

After an alarm priority has been changed on an alarm limit, and if that alarm limit is currently in alarm, then the Alarm Summary Display at US/GUS nodes are updated with the new alarm status.

For example,

If the priority on PVHIPR changed from High to No Action and the point is currently in violation of PVHITP, then the alarm is removed from the Alarm Summary Display.

If the priority changed from Journal / Printer / No Action/ or Jnlprint to Low, then the alarm is added to the Alarm Summary Display.

### 2.3.15 Primary Module Points

Primary Module (PRIMMOD) Points provide a mechanism for grouping related points; it is used to collect alarms and events from points that are related for some purpose. The PRIMMOD parameter contains the name of a single point that is linked to other related points. All points with matching PRIMMOD parameters are considered to be in the same group.

PRIMMOD parameters can be used in the following ways:

- Create alarm groups—the composite alarm status is maintained for a PRIMMOD group and can be used to support more flexible and customized alarm annunciation through button LEDs, custom schematics, and the Area Annunciator Display.
- Retrieve events—from the Event History Retrieval Display, you can enter the primary module point name to retrieve events related to points with the same PRIMMOD parameter.
- Identify a set of points—as belonging to a batch equipment unit or batch ID.
- Identify the Process Module Point—that is responsible for manipulating a group of points.

Any type of LCN point can be used as a Primary Module Point. All LCN points have a PRIMMOD parameter except for UCN component and CG points. There is no limit to the number of points that can be grouped under a primary module point.

Use the following guidelines when assigning points to a Primary Module:

- The PRIMMOD default is null (underscores “\_” or dashes “-”).
- The PRIMMOD value of an AM point can be any valid local LCN point.
- The PRIMMOD value of a UCN point must be a point in the local NIM.
- The PRIMMOD value of a Hiway point must be a point in the local HG.

### 2.3.16 AM Multiple Primmods Alarming Option

With R520, a new NCF option allows a single alarm on an AM point to be reflected in four new Primmod alarm groups in addition to the standard Primmod parameter group which is used for alarm displays and event history. This accommodates the sharing of a common piece of equipment by up to five processes; e.g., batches or oil movements, simultaneously.

These new Primmod alarm groups are identified by four 16-character string-type parameters: \$MPROD1, \$MPROD2, \$MPROD3, and \$MPROD4. These strings will take the place of the PRIMMOD parameter strings in LCN alarm events. The PRIMMOD and \$MPROD groups can be referenced from displays and button LEDs, and used for event history retrieval.

The NCF Configurator has an option parameter that controls whether the Multiple PRIMMOD functionality is enabled or not. This option has three modes:

- Multiple PRIMMOD Option Disabled—Alarms are reflected only in PRIMMOD parameter alarm groups (not \$MPROD).
- Multiple PRIMMOD Option Enabled Exclusive—Alarming on PRIMMOD and \$MPROD is mutually exclusive (e.g., if a point is in an alarm state and at least one \$MPROD parameter on that point has a value, that alarm will be reflected in the alarm groups for those \$MPROD parameters which have values and will not be in the alarm group for the PRIMMOD parameter. If none of the \$MPRODs have values, the alarm will be reflected in the PRIMMOD group).
- Multiple PRIMMOD Option Enabled Inclusive—An alarm on a point will be reflected in the alarm groups for all the point's \$MPROD parameters which have values and also on the alarm group for the PRIMMOD parameter.

All alarmable AM points have these new \$MPROD parameters. This functionality does not apply to the HG, NIM, or CG.

### 2.3.17 Auxiliary Unit

In R520, all alarmable AM point types have a new parameter called Auxiliary Unit (\$AUXUNIT). This new parameter is in the Data Entity Builder (DEB) and on the Configuration Page of the point's Detail Display. It allows all process alarms on a point to be dynamically redirected from the point's primary unit to an alternate unit for alarm displays.

Process alarm events, that are generated from points whose \$AUXUNIT has been configured, will be accepted only by Universal Stations having an Area Database in which both the Primary Unit and the Auxiliary Unit are configured. If the \$AUXUNIT parameter is set to null (--), alarms and messages on that point go to the primary unit. If a valid Unit ID is specified, alarms and messages from that point go to the Auxiliary Unit.

A point's \$AUXUNIT parameter can be modified by Schematic, Detail display, DEB, Alter Parameters, Computing Modules, or CL as determined by the key-access level that is configured in the NCF. As long as there is no key level violation, the user can modify the \$AUXUNIT parameter at any time even if the point is active.

An Auxiliary Unit on the Alarm Summary and on the Alarm Annunciator displays will appear in the unit column in reverse video, half-intensity green. If an Auxiliary unit does not exist, the unit column will contain the primary unit number and will not be back-lit.

If you do not want to use this feature, leave the new parameter \$AUXUNIT at its default value, "--".

## **2.4 MESSAGES**

### **2.4.1 Message Generation**

Messages generated by Application Modules are 1-way messages that may or may not be acknowledged by the message receiver. Messages in the AM are originated by CL Blocks.

### **2.4.2 Message Distribution**

Messages are distributed to Universal Stations at the end of each point-processing cycle in the same way as alarms. The messages are sent to the Universal Station(s) that the unit containing the point is assigned to. Each message also contains a time indication, the point tag-name, and the unit ID.

## **2.5 EXTERNAL MODE SWITCHING IN AMS**

External mode switching (EMS) is typically used to establish mode interlocks or, under certain process conditions, to restrict the use of a mode that invokes a higher level of control.

Each regulatory data point has these three logical parameters that are used in EMS: ESWMAN, ESWAUTO, and ESWCAS. They correspond to MAN, AUTO, and CAS. General input connections or general output connections can be used to change these parameters from other data points.

The following mode-switching options can be configured in EXTSWOPT for each regulatory point:

- Ems—External Mode Switching
- Emp—External Mode Permissives
- None

When one of the first two options is configured, the function can be enabled or disabled by the operator or by a user-written program (with ESWENBST). Mode changes made this way have no effect on the mode attribute of the data point or on the normal mode and normal mode attribute of the point. When a mode change is made, the last values of the normal mode and normal mode attribute parameters stay in effect.



## 2.5.1 Mode Switching

When mode switching is enabled, the operator and the system functions are prevented from changing the mode of the data point and the mode is determined by the ESWMAN, ESWAUTO, and ESWCAS parameters, as indicated on Table 2-1.

## 2.5.2 Mode Permissives

The principle use of mode permissives is to restrict mode changes, under certain process conditions, that lead to a higher-level of control (e.g., AUTO to CAS), or to cause a shed to a lower-level mode (e.g., CAS to MAN). When MODPERM contains Enabled, the status of EMS parameters ESWMAN, ESWAUTO, and ESWCAS affects the modes as shown in Table 2-2.

**Table 2-1 — Mode-Switching Operation**

Table 2-1: Mode Changing Operation			
EMS Enabled?	EMS Parameters (Note 1) M   A   C	Effective New Mode	Can Mode Be Changed By Operator or Program?
No	X   X   X	Last Mode Requested	Yes
Yes	F   F   F	Previous Mode (Note 2)	No
	T   X   X	Man	
	F   T   X	Auto (Note 3)	
	F   F   T	Cas (Note 3)	
<b>Notes for Table 2-1:</b> Note 1 — M is ESWMAN, A is ESWAUTO, and C is ESWCAS, T = True, F = False, and X = don't care. Note 2 — Previous Mode is the mode when the point was last processed. Note 3 — If the effective new mode is not legal for the control algorithm, the mode doesn't change.			

**Table 2-2 — Mode-Permissive Operation**

Permissives Enabled?	Status of the EMS Parameters (1)			Last Mode (2)	Effective New Mode	Can the Mode Be Changed By An Operator or a Program?
	M	A	C			
NO	X	X	X	any	same as last mode	Yes
YES	F	F	F	any	same as last mode	Yes
YES	T	X	X	AUTO,CAS,	MAN	The mode cannot be changed to CAS
YES	F	T	X	MAN or AUTO	same as last mode	
				CAS	AUTO (3)	
YES	F	F	T	any	same as last mode	Yes

**Notes for Table 2-2:**

Note 1 — M is ESWMAN, A is ESWAUTO, and C is ESWCAS, T = TRUE, F = FALSE, and X = don't care.

Note 2 — Last Mode is the mode last entered by the operator or a program, or the mode when the point was last processed.

Note 3 — If AUTO mode is applicable, else MAN.

## 2.6 AM SYSTEM EVENTS

The following system events can be reported on the LCN for each AM:

- **AM Failure**—Indicates that the AM is no longer available for data acquisition, control, or event-reporting.
- **AM Startup**—The AM is being restarted; it may not yet be available for data acquisition, control, or event reporting, but once requalified and running, it will be.
- **Point-Processing Schedule Overrun**—Point-processing schedules are not being met within the tolerances specified under 2.2.1.1. See 2.2.1.6 for additional information about schedule overruns.

## 2.7 RESTARTING AMs

Restarting an AM consists of loading the on-process personality (.PI files), the network-configuration files (NCFs), and the database (checkpoint files). The restart is requested by an operator at a Universal Station, through the AM Status display (see the *Process Operations Manual*). The files to be loaded can be on floppy disks, or they can be in a History Module.

These are the four types of AM restarts:

- **Cold Restart**, subsection 2.7.2
- **Hot Restart**, subsection 2.7.4
- **Warm Restart**, subsection 2.7.3
- **No-Point-Processing Restart**, subsection 2.7.5

### NOTE

You must determine the type of restart to use based on the situation that exists at the time. The factors that will influence your decision are the length of time since the last checkpoint, the requirements and characteristics of your process (especially the volatility), and the characteristics of each type of restart. Subsections 2.7.2 through 2.7.5 give the characteristics of the four restart types.

### 2.7.1 Checkpointing the AM Database

Checkpointing consists of the transfer of a copy of the AM database to a checkpoint file in a History Module by automatic checkpointing (see Section 21 of the *Engineer's Reference Manual*), or when a Universal Station operator requests it. The operator can enable or disable automatic checkpointing.

The database is organized by process units. When checkpointing for a unit starts, all activity for that unit stops. This assures that the checkpointed data is a "snapshot" of the data at the time the unit activity stopped. The checkpoint file for each unit includes a record of that time. You can determine the time of the last AM checkpoint at the AM Status display, by selecting an AM node, selecting `AUTOLOAD NET`, selecting `MANUAL LOAD`, and then selecting `CHKPNT TIME`.

#### NOTE

If you use the `AUTOLOAD NET` or the `AUTOLOAD LOCAL` targets to initiate an AM reload and restart, the load type targets (hot, warm, cold, etc.) do not appear automatically. The load type will be the default that is configured in the NCF. If you use `AUTOLOAD NET`, an `OVERRIDE DEFAULT` target appears. Selecting the `OVERRIDE DEFAULT` target brings up the load type targets (hot, warm, cold, and no point process,) allowing you to override the NCF default load type.

#### NOTE

The following descriptions of cold, warm, hot, and no-point processing restarts assume that the files mentioned under 2.7 have been reloaded into the AM before the restart occurs, or that they were retained in the AM during the time it was "off-process."

### 2.7.2 Cold Restart

The characteristics of a cold restart are:

- Primary data points in the AM that have a secondary in a PM, LM or a box on a Data Hiway go into initialization (SPC or DDC control of the secondaries doesn't automatically resume). During the first processing pass for each such AM point, the AM turns off the "remote cascade" request in its process-connected secondary points.

A Universal Station operator must then put each process-connected secondary in CAS mode to reestablish control from the AM.

- All alarms are cleared from the AM database and from the Universal Station displays. As point processing resumes, any alarms detected are reported to the Universal Station(s) as new alarms.
- The value status for each AM PV is changed to Bad. During the first processing pass, the PV is recalculated. This may result in momentary Bad PV alarms for points whose

PV inputs are received from other points that have not yet been processed.

- On the first processing pass, all points that use control processing go through initialization, and then resume normal processing on subsequent passes.

### **2.7.3 Warm Restart**

The characteristics of a warm restart are:

- On the first processing pass, AM points that have secondary points in a PM or a box on a Data Hiway are initialized and change their secondaries' mode to CAS if the secondary has an outstanding "remote cascade" request. SPC or DDC control of the secondary points in process-connected boxes resumes on the second processing pass.
- All existing alarms are retained in the AM's database and on Universal Station displays. As point processing resumes, these alarms are not reported, and if they have returned to normal they are taken off the displays.
- All PVs remain at their checkpointed values. During the first point-processing pass, PV initialization takes place, and normal PV processing resumes on subsequent passes.
- On the first processing pass, all points that use control processing go through initialization and resume normal processing on subsequent passes.

### **2.7.4 Hot Restart**

The characteristics of a hot restart are:

- On the first processing pass, SPC or DDC control of secondary points in PMs or boxes on a Data Hiway is resumed without initialization. If such a secondary point has an outstanding "remote cascade" request, its primary in the AM changes the secondary's mode to CAS. The CV of the primary in the AM is adjusted to the initialization value received from the secondary if the control algorithm in the AM has a floating output (PIDs do). On subsequent processing passes, the AM points receive normal processing.
- All existing alarms are retained in the AM's database and on Universal Station displays. As point processing resumes, these alarms are not reported and if they have returned to normal they are taken off the displays.
- All PVs remain at their checkpointed values. Normal PV processing takes place on the first pass and subsequent passes.

### **2.7.5 No Point Processing Restart**

The AM restarts, but no point-processing cycles are executed.



## DATA POINT DESCRIPTIONS

### Section 3

### 3.1 AM REGULATORY DATA POINTS

Regulatory data points are used to control analog process variables by maintaining the variable at a setpoint value, with as little deviation as possible. Regulatory data points, by themselves, or in combination with other regulatory points, act as controllers that manipulate a process element, such as a valve, to maintain the process variable at the desired value.

#### 3 1.1 Functional Structure

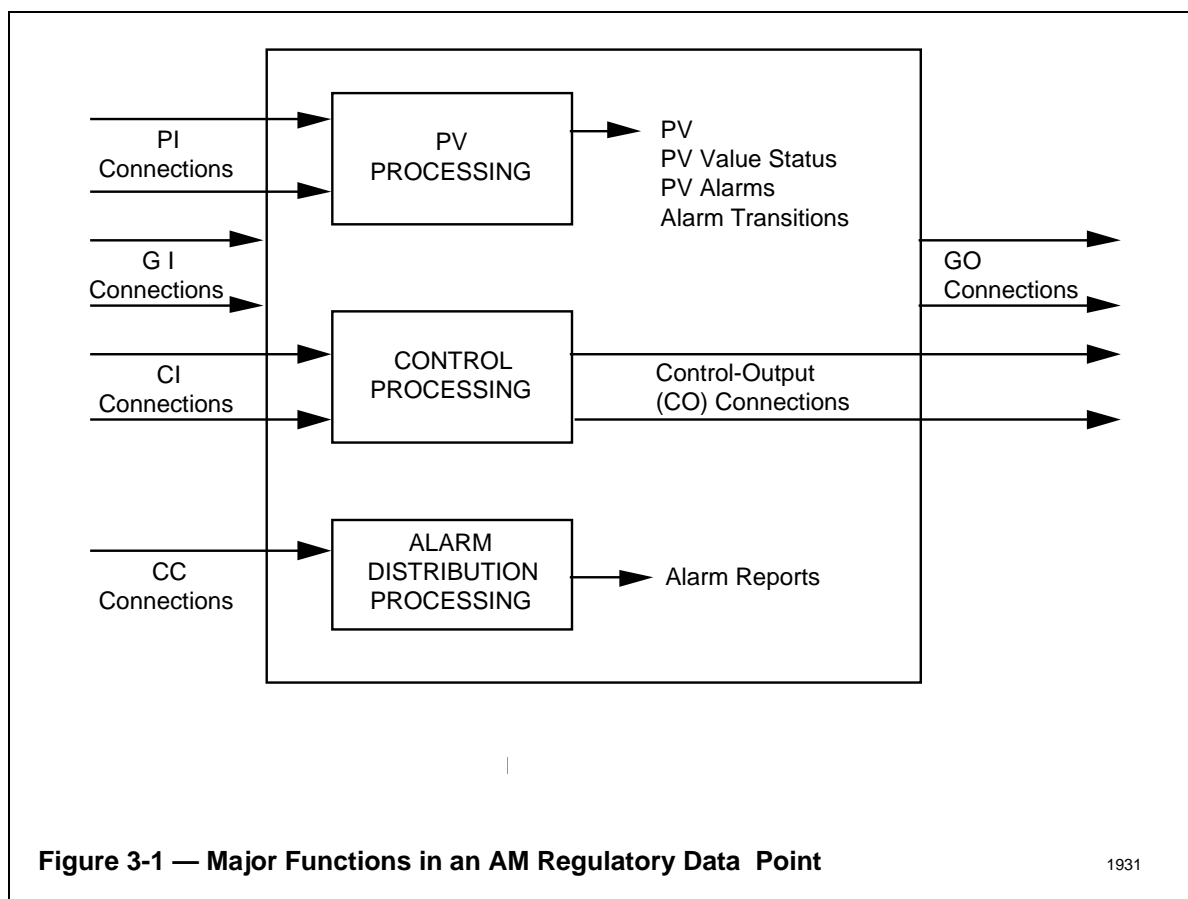
The major functions of an AM regulatory point are presented in Figure 3-1. On that figure, PV processing and control processing are the most significant. AM regulatory points can be configured to use either standard PV and control processing, or you can attach user-written programs to these points, in the form of CL Blocks, to customize the points' functions. General input (GI) and general output (GO) connections are also available to help customize the operation of these points.

The main purpose of PV processing is to calculate a process variable (PV) and to check the results against configured alarm limits. An example of PV calculation is mass compensation of a flow value read from the process to yield a PV for more precise flow control. Input value(s) from the process or from other data points are obtained through PV-input (PI) connections.

The main purpose of control processing is to calculate an output value (OP) that drives a valve or serves as an input to another (secondary) data point. Inputs from the process or other data points are obtained through control input (CI) connections. The output value is sent to one or more secondary points through a control output (CO) connection or connections. If the output drives a valve, it does so through an HG regulatory or analog output point, so in this sense, AM CO connections always go to a secondary data point.

A contact-cutout feature stops the reporting of alarms to the Universal Station under certain circumstances (alarm-history collection continues). An AM-regulatory data point can accept one contact cutout-input (CC) connection from a point that dictates whether alarms are cutout or not.

General input (GI) connections and general output (GO) connections are used to exchange information between data points. CL blocks can be attached to AM regulatory data points and executed at any of several insertion points in the processing sequence. They can be used to take special actions, such as alarm actions, and they can perform special calculations or logical operations. Custom PV algorithms and control algorithms can be developed through the use of the CL PV algorithm and the CL Control algorithm.



### 3.1.2 AM Regulatory Point Execution States

AM regulatory point-parameter PTEXECST indicates the execution state of the point as follows:

- **NotConfig** The point is being loaded into the AM by the Data Entity Builder. This is a temporary state. When loading is complete, the state automatically changes to Inactive.
- **Inactive** The point is not being processed. All of the parameters needed to define the structure of the point are present, but some other parameter values can yet be specified. Some configuration parameters can be changed only in this state.

When the point's state changes from Active to Inactive, the value status for PVCALC, PVAUTO, and CV are changed to Bad, all alarms are cleared, and, if one or more control-output connections is configured, the control initialization state indicated by INITMAN changes to On. If PVSOURCE contains Subs or Auto, the PV value status is changed to Bad.



- **Active**        The point is fully configured and is being processed as scheduled. This is the normal operating state. When the state changes from Inactive to Active, the point is subjected to one initialization pass before resuming normal processing.

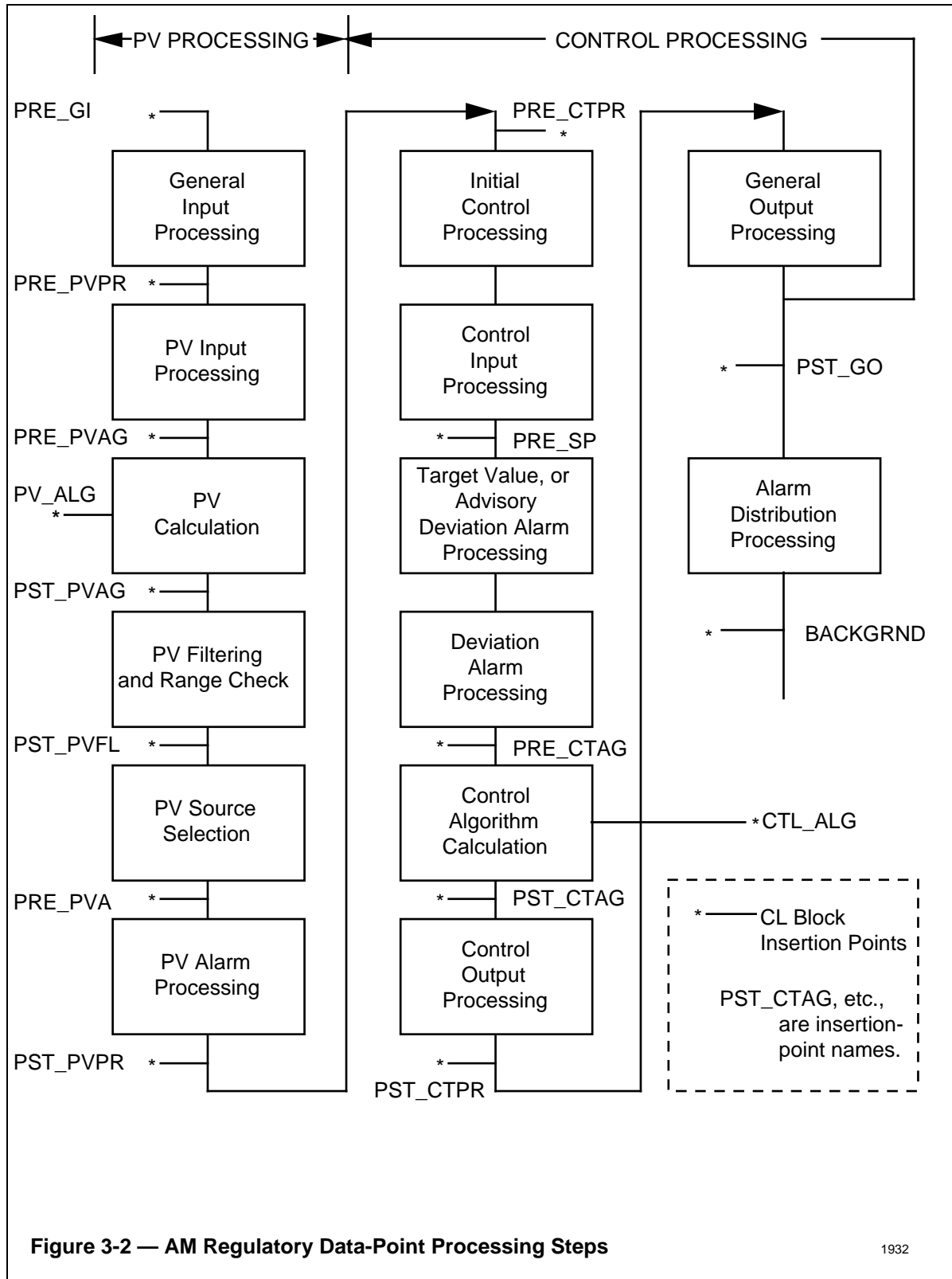
Parameter PTEXECST can be changed at a Universal Station by someone with a Supervisor's or an Engineer's key, or by a user-written (CL) program.

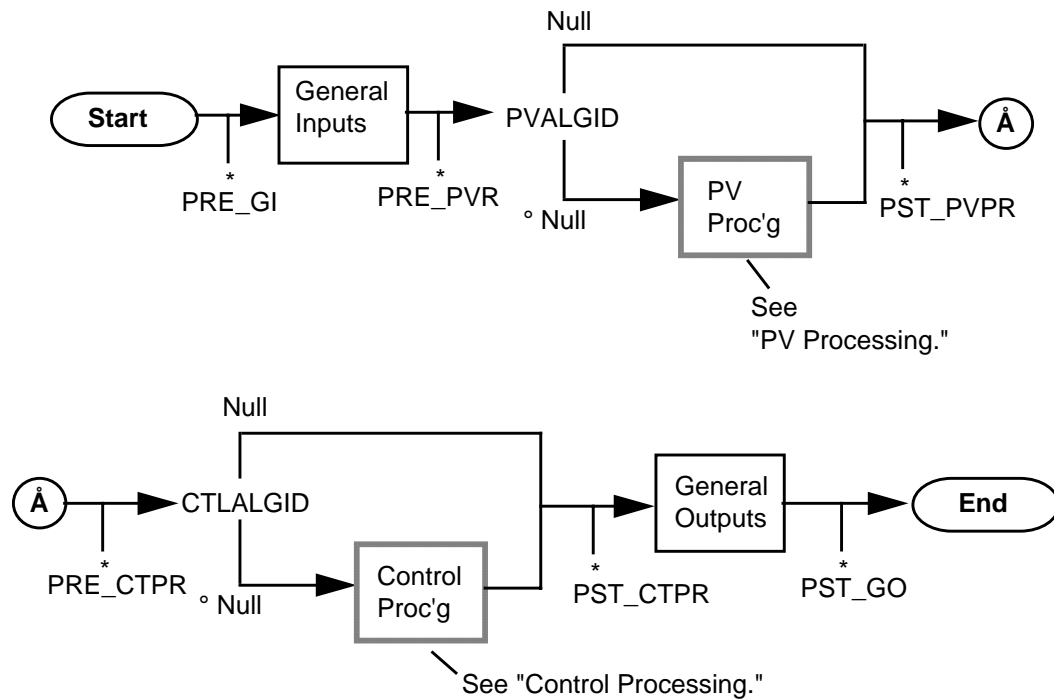
### 3.1.3 Processing Order

The major steps in processing an AM regulatory data point are shown in Figure 3-2. All of these steps can be taken, if the point is so configured; however, if a standard or user-written PV algorithm isn't configured, PV processing won't take place, and if a standard or user-written control algorithm isn't configured, control processing won't take place.

Between most of the processing blocks in Figure 3-2, "\*----" is shown to indicate points where CL blocks or CL switches can be inserted. The PV calculation can be the use of a standard algorithm or it can be the use of a user-written algorithm—an algorithm written in CL. Likewise, the control calculation can be the use of a standard or a user-written (CL) algorithm.

Figure 3-2a shows the processing sequence for AM regulatory points. It can be used to determine the relative order in which decisions and calculations are made.





Processing depends on the value in PPSTYPE, as follows:

- PPSTYPE = None    Normal, scheduled pass; process everything.
- = Normal    Process everything that is real-time dependent, such as PID calculations, SP, target-value ramping, and dead time.
- = Init,    Only parts of Control Processing are Executed. See
- = Or,    "Control Processing."
- or
- = Man

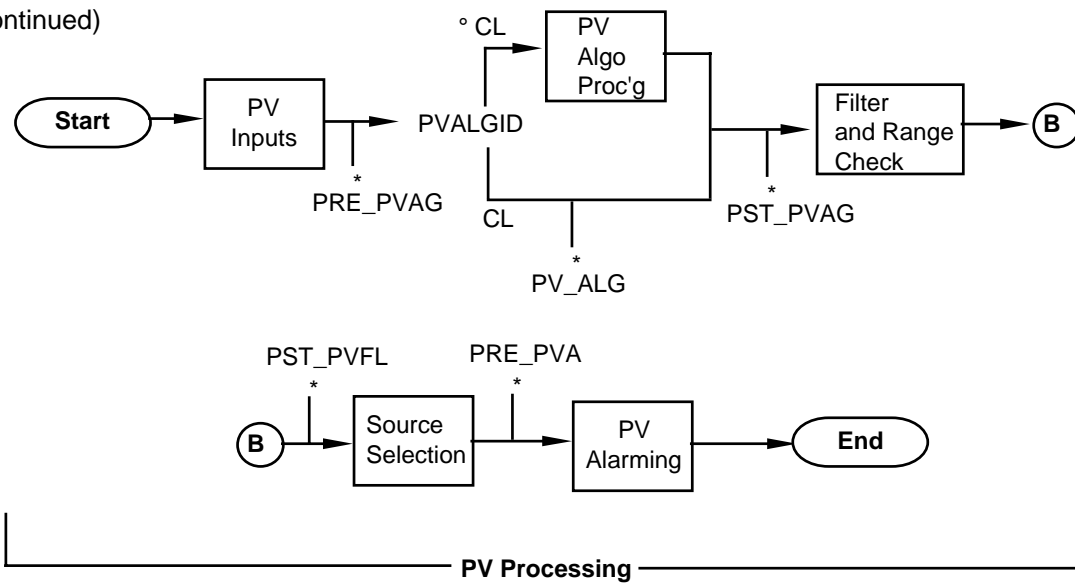
Processing Overview

**Figure 3-2a — AM Regulatory Data-Point Processing Sequence**

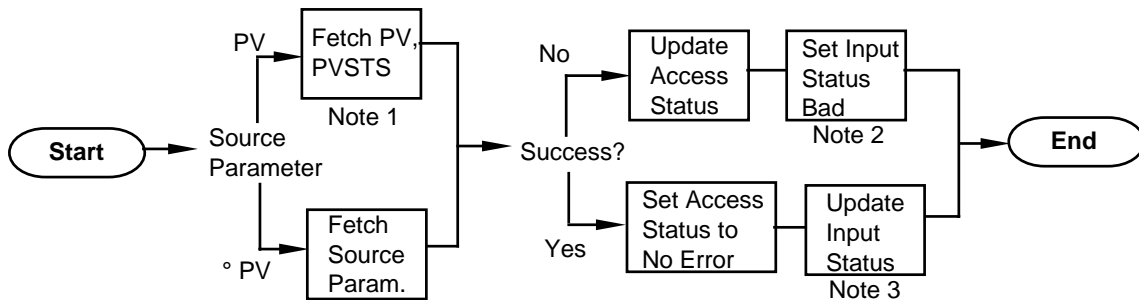
1933

(Continued)

(Continued)



Repeat the following for each active connection:



- Notes:
1. PVSTS is fetched only if the source point has PVSTSREC.
  2. This step is bypassed for a control-input connection that stores in SP.
  3. Control-input connections that store in SP obey all restrictions applicable to the continuous-control access key. The access status is updated if an error is detected as the value is stored.

**PV-Input and Control-Input Connections**

**Figure 3-2a — AM Regulatory Data-Point Processing Sequence**

1933

(Continued)

(Continued)

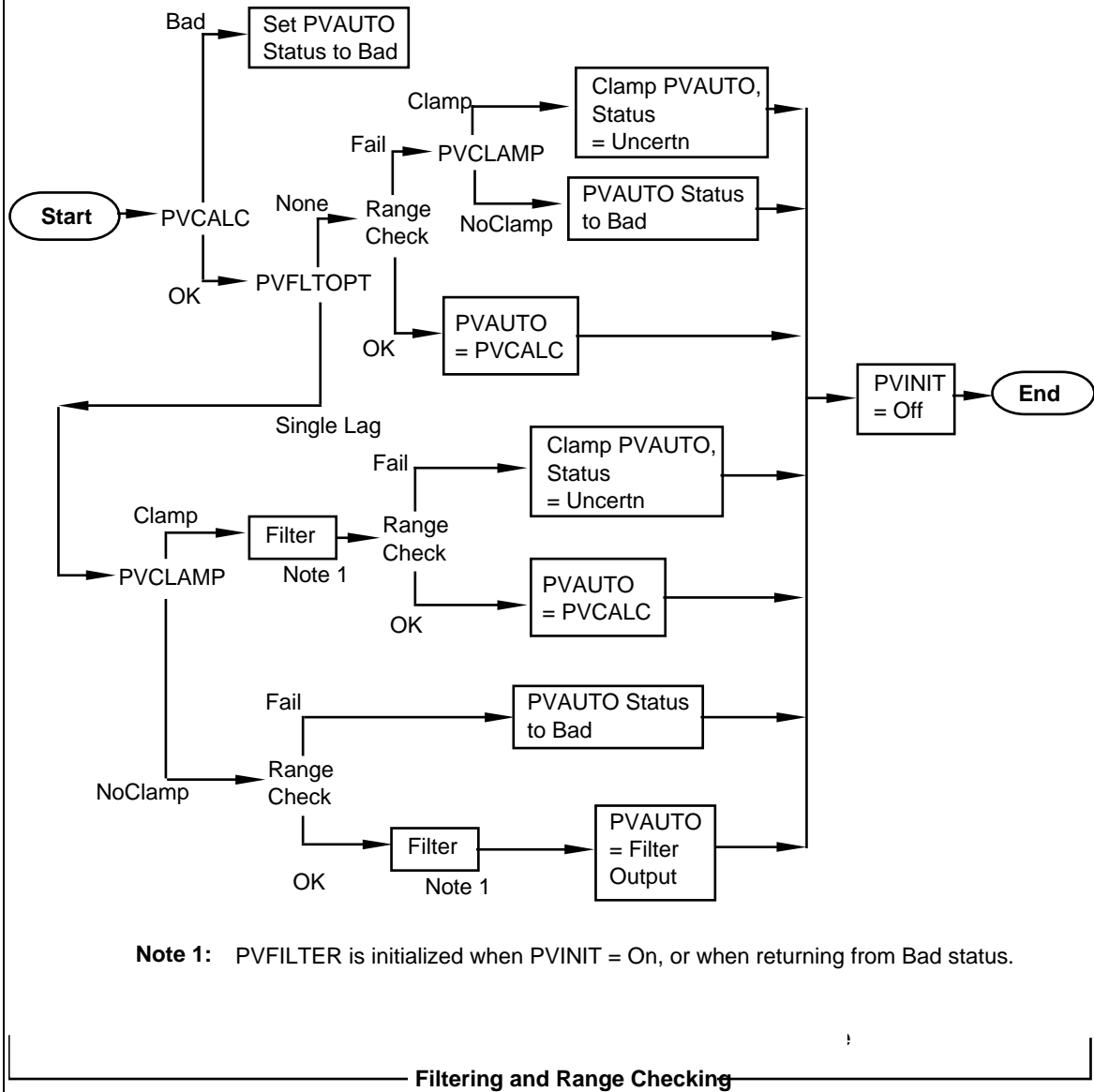
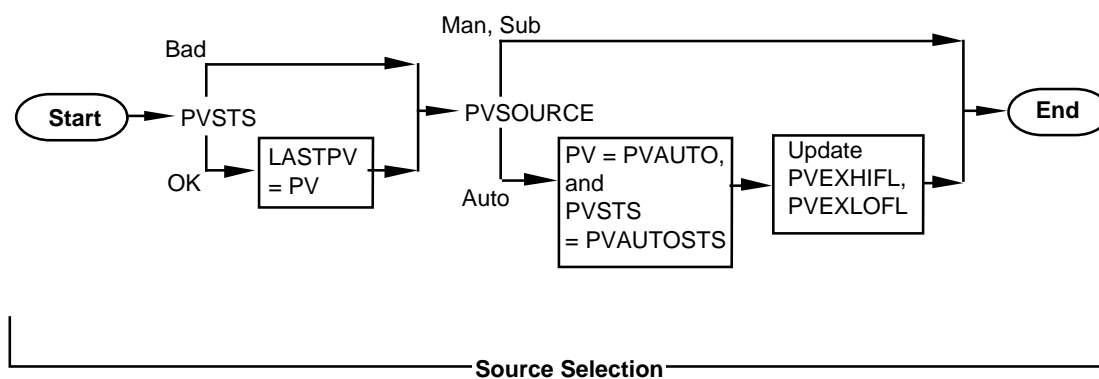


Figure 3-2a — AM Regulatory Data-Point Processing Sequence

1933

(Continued)

(Continued)

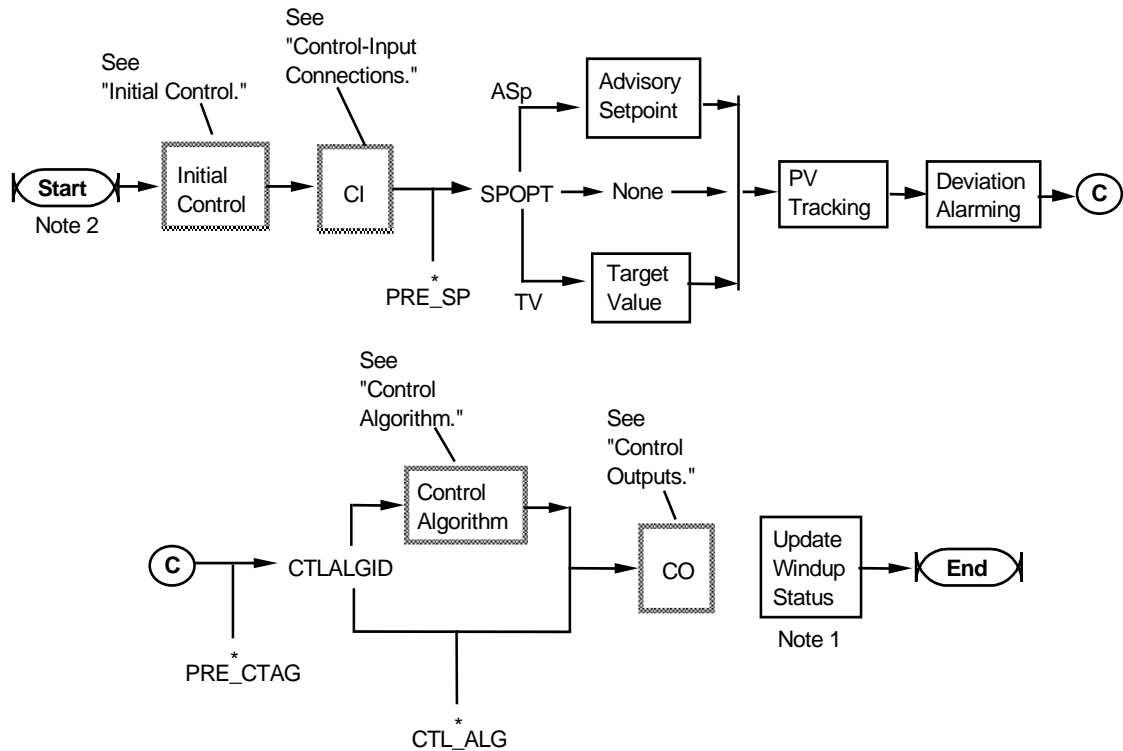


**Figure 3-2a — AM Regulatory Data-Point Processing Sequence**

1933

(Continued)

(Continued)



**Notes:** 1. This block is always executed.

2. If PPSTYPE = Init — Only initial control, control algorithm (including CL), PRE\_CTAGE and PST\_CTAGE insertion points, and control outputs are executed.

If PPSTYPE = Or — Only initial control and control algorithm (including CL), and PRE\_CTAGE insertion points are executed.

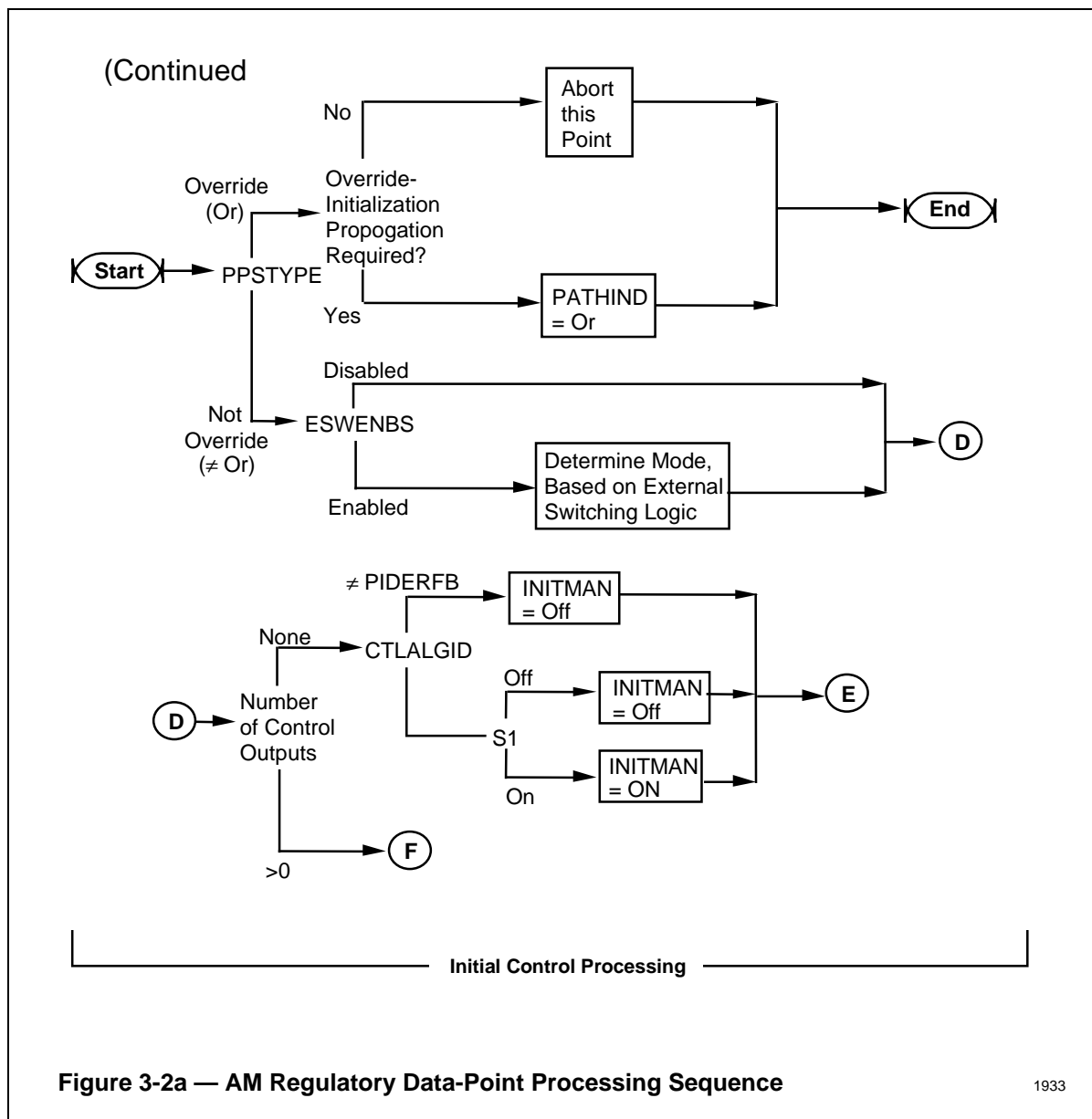
If PPSTYPE = Man — Only initial control and control outputs are executed.

**Control Processing**

**Figure 3-2a — AM Regulatory Data-Point Processing Sequence**

1933

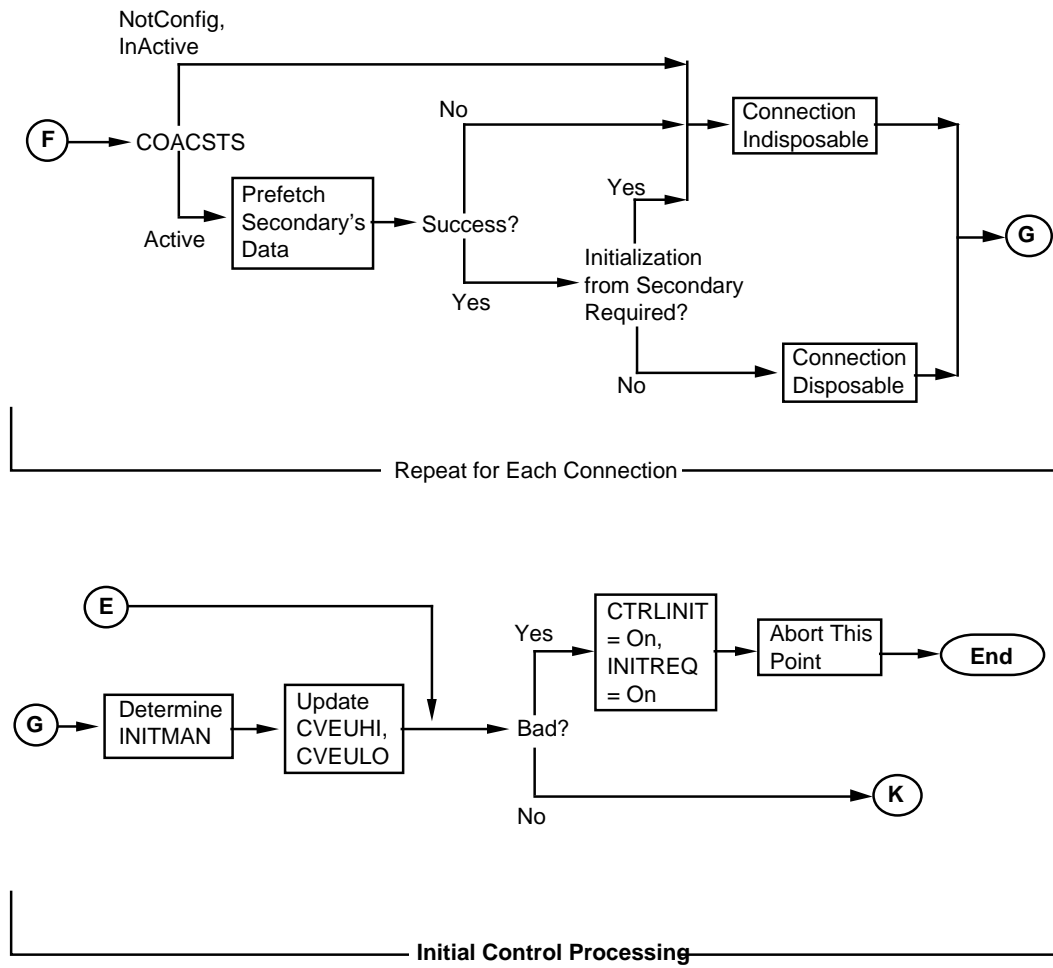
(Continued)



(Continued)



(Continued)



**Figure 3-2a — AM Regulatory Data-Point Processing Sequence**

1933

(Continued)

(Continued)

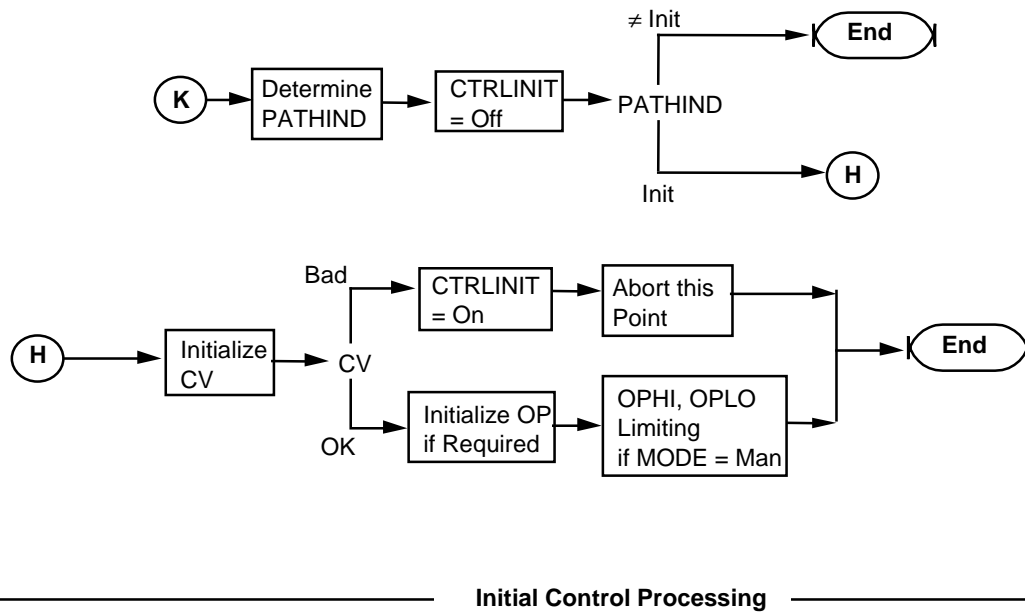
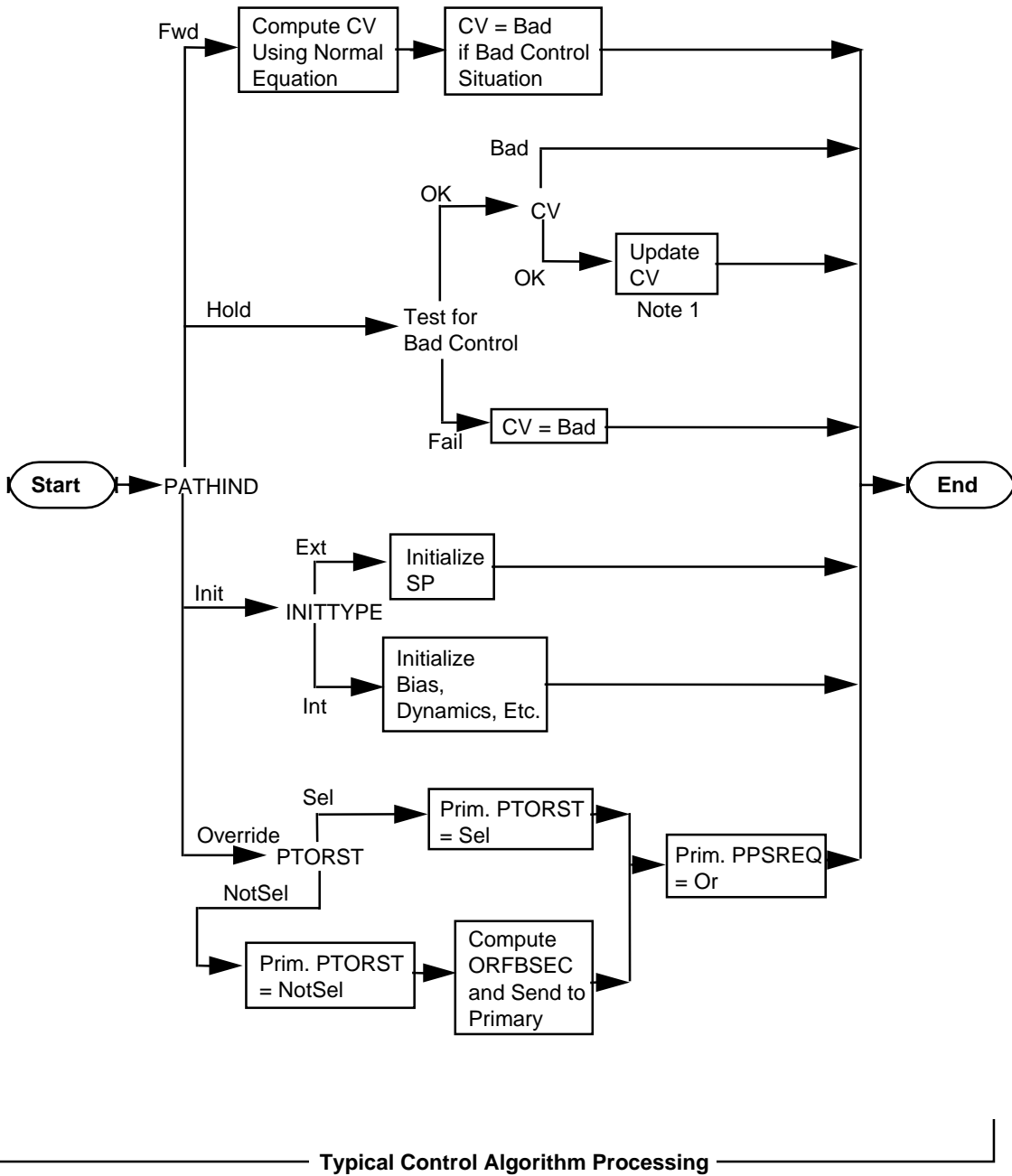


Figure 3-2a — AM Regulatory Data-Point Processing Sequence

1933

(Continued)

(Continue



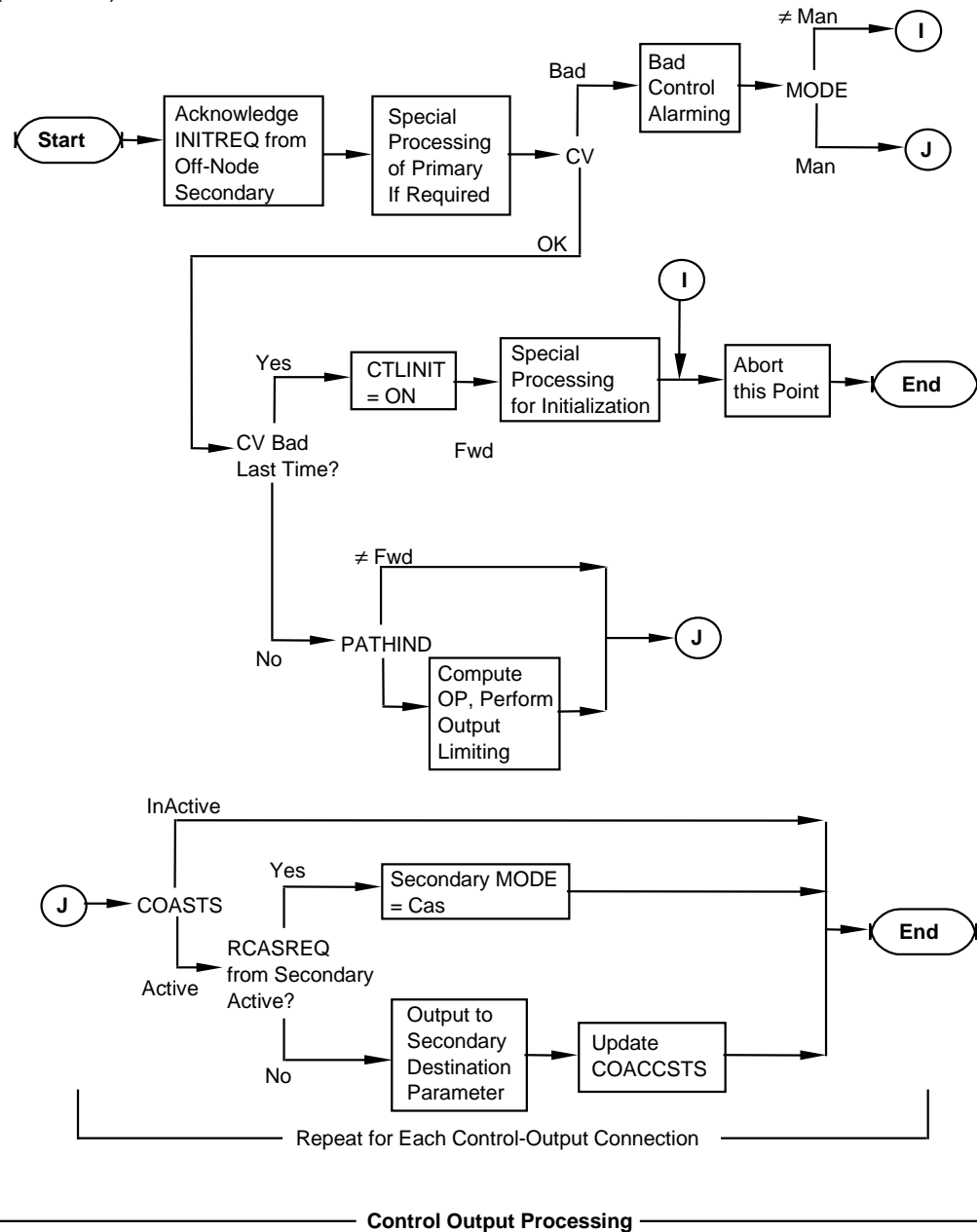
**Note 1:** Either compute new CV or initialize CV from output.

**Figure 3-2a — AM Regulatory Data-Point Processing Sequence**

1933

(Continued)

(Continued)



**Figure 3-2a — AM Regulatory Data-Point Processing Sequence**

1933

### 3.1.4 AM REGULATORY DATA-POINT INTERCONNECTIONS

Strategies that involve more than one data point are created through input and output connection between data points. Input and output connections read data from and write data to the parameters of other data points in the system. The following types of input and output connections are available:

- PV Input Connections
- Control Input Connections
- Contact-Cutout Input Connection
- General Input Connections
- Control Output Connections
- General Output Connections

All connections are optional. You configure the number of connections and the source or destination for each connection, as you build each data point, through the Data Entity Builder. The connections to be configured can be documented before point building, on the AM Regulatory Data Point Configuration Forms, in the *Application Module Forms* package in the *Implementation/Configuration Forms* binder.

#### 3.1.4.1 PV Input Connections

PV input connections are used to specify the source(s) for the inputs to PV algorithms. A maximum of eight PV input connections can be configured. For each such connection, a source point and a source parameter in that point must be configured. The source parameter must contain a real number. The destination parameters for these sources default to one for each input to the configured PV algorithm.

For some PV algorithms, one or more inputs default to a bad value (see AM Algorithm Engineering Data). At least one input connection must be configured for each of these inputs, otherwise the point would attempt to run with a bad value. You can configure initial values for other inputs, if you need to. If no input connection is configured for an input parameter, it retains the default value or the configured initial value.

#### 3.1.4.2 Control Input Connections

Control input connections are typically used to acquire "noninitializable" inputs to the configured control algorithm. They can be used to acquire initializable inputs, but if initialization and windup protection are required, a control output connection from the primary point must be used.

A maximum of eight control input connections can be configured. For each such connection, the source point and a parameter in that point must be configured. The value in the source parameter must be a real number. The destination parameters for these sources default to one for each noninitializable input to the configured control algorithm.

For some control algorithms, one or more inputs default to a bad value (see *Application Module Algorithm Engineering Data*). At least one input connection must be configured for these inputs, otherwise the point would attempt to run with a bad value. You can configure initial values for other noninitializable inputs, if you need to. If no input connection is configured for an input parameter, it retains the configured initial value or the default value.

When the CL Control Algorithm is used, control inputs cannot be configured in the Parameter Entry Display (PED). Inputs are acquired by direct references in CL or through general inputs to a CDS that is included in the data point.

Any control input with a status other than normal or Uncertain, is a bad status, and causes the value in CV to be NaN.

#### **3.1.4.3 Contact-Cutout Input Connection**

In regulatory, counter, and timer points, a contact-cutout input connection through the parameter CCSRC provides a dedicated input to the point's contact cutout parameter (CONTCUT, see 4.3.1.7 in *System Control Functions*). The source parameter must contain a logical value. Only one such connection can be configured. See heading 2.3.13 for additional information on contact cutout.

#### **3.1.4.4 General Input Connections**

General input connections are used to change parameters in the data point. These are usually static parameters such as the SP high limit, the PV low-alarm trip point, the gain for a PID, etc. A maximum of eight general input connections can be configured. For each such connection, you must configure a source data point and a parameter in that point, plus a destination parameter in this data point. The source and destination parameters must have the same data type.

These connections store values in the destination parameter in a manner similar to that of CL blocks, and they follow all of the constraints that apply to CL blocks. A bad value is never stored in a destination parameter whose data type is real number.

#### **3.1.4.5 Control Output Connections**

Control output connections are used to form initializable connections between Regulatory data points or between Regulatory data points and Analog Output points. After appropriate limit checks and conversions, the output of the control algorithm is transferred to the secondary data point.

A maximum of eight control output connections can be configured. For each such connection, the destination point and the destination parameter in that point must be configured. The following are the valid destination parameters when designated as an initializable input for the destination point algorithm:

- The destination (secondary) point is in the same unit and in the same AM; SP, X1, X2, X3, and X4.
- The destination (secondary) point is in a different unit in the same AM, or in a different AM; SP and X1.
- The destination (secondary) point is in a NIM or HG; SP and OP.

#### **3.1.4.6 General Output Connections**

General output connections are used to change parameters in other data points. Usually these destination parameters are static parameters.

A maximum of eight general output connections can be configured. For each such connection, the source parameter (in this point), the destination point, and the destination parameter must be configured. The source and destination parameters must have the same data type.

These connections store values in the destination parameter in a manner similar to that of CL blocks, and they follow all of the constraints that apply to CL blocks. A bad value is never stored in a destination parameter whose data type is real number.

### WARNING

Do not connect the General Output of a point to the PPS input of a second point. Each time the point with the General Output is processed, it will set or reset the PPS of the second point—even if it is already in the desired state. If the operator requests special processing by setting the PPS to On, a conflict results that can cause the second point to “hang” (stop processing). To initiate special processing safely, provide a CL block that detects the desired condition and sets the PPS of the second point, or changes the state of its PPSREQ to Normal. (A timer point would require configuring a separate point to which the CL block would be attached).

#### 3.1.4.7 Activity Status

For each input and output connection, there is an activity-status parameter that indicates the state of the connection, as follows:

- NotConfig—The connection is not configured. This status can't be changed unless the connection is fully configured. If the connection is fully configured, this cannot be its state.
- InActive—The connection is fully configured, but data is not being transferred over it.
- Active—The connection is fully configured and transfers are active.

The parameters are as follows:

- PIACTSTS(n)—PV-input activity status
- CIACTSTS(n)—Control-input activity status
- CCACTSTS—Contact-Cutout activity status
- GIACTSTS(n)—General-input activity status
- COACTSTS(n)—Control-output activity status
- GOACTSTS(n)—General-output activity status

More than one PV, general, or control input connection can be configured for the same destination parameter, but only one can be active at any time. This allows selection of alternate sources for a destination parameter. When an input connection is made active, all other connections to the same parameter are automatically made inactive.

#### 3.1.4.8 Access Status

An access status parameter is provided for each input and output connection to indicate the result of the last value transfer. The values that can be held in these parameters are defined in the *Application Module Parameter Reference Dictionary*. The parameter names are like those under 3.1.4.7 except that "ACT" is replaced by "ACC," hence PIACCSTS(n), COACCSTS(n), GOACCSTS(n), etc.

### 3.1.5 PV Processing

The following are the principal PV processing functions:

- PV input-connection processing
- PV algorithm processing
- PV filtering
- PV source selection
- PV range checking
- PV alarm detection

If parameter PVALGID contains Nul, PV processing does not take place.

#### 3.1.5.1 PV Input Connection Processing

Here, the values are transferred into this data point over the PV input connections. See 3.1.4.1.

#### 3.1.5.2 PV Algorithm Processing

The configured PV algorithm accepts the values received over the PV input connections and produces the calculated PV value, PVCALC, plus its value status, PVAUTOST. If the CL PV Algorithm is configured, a user-written CL Block is used in place of a standard PV algorithm. The AM algorithms are described in the *Application Module Algorithm Engineering Data* manual.

#### 3.1.5.3 PV Filtering

If PV filtering is configured, a single-lag filter is applied to the PVCALC value to remove noise. The output of the filter is placed in PVAUTO. If filtering is not configured, PVAUTO contains the same value as PVCALC.

The PV filtering function is configured through the Data Entity Builder by placing SinglLag in PVFLTOPT. The lag-time constant is specified in minutes in TF. Legal values for TF are 0.0001 to 1440.0; a value less than 0.0001 is assumed to be zero.

Regardless of the PVFLTOPT option, the filtering function also makes range checks on PVCALC. If the PV source is AUTO and if PV clamping is not configured, it sets the value status (PVAUTOST) bad when PVCALC goes outside the extended PV range. This prevents any delay in detecting an out-of-range PV caused by the filtering action.



#### 3.1.5.4 PV Source Selection

The source of the PV can be PV processing, a Universal Station, or a user-written program. It is specified by the value in PVSOURCE, which can be changed by an operator, a supervisor, an engineer, or a user-written program. PVSOURCE has one of these values:

- **Auto**—The PV is received from PV processing through the PV filtering function. The value is in PVAUTO and its status is contained in PVAUTOST.
- **Man**—The PV is entered by an operator, supervisor, or engineer at a Universal Station.
- **Sub**—The PV is entered by a user-written program.

During normal operation the PV source is Auto, and the PV and its value status (PVSTS) become equal to PVAUTO and PVAUTOST, respectively, before PV range checks are made (3.1.5.5).

When the PV source is changed from Auto to Man or Sub, the PV remains at the last value until it is changed by the operator (Man) or a program (Sub), so it doesn't "bump." In Man or Sub, the status in PVSTS is Uncertn.

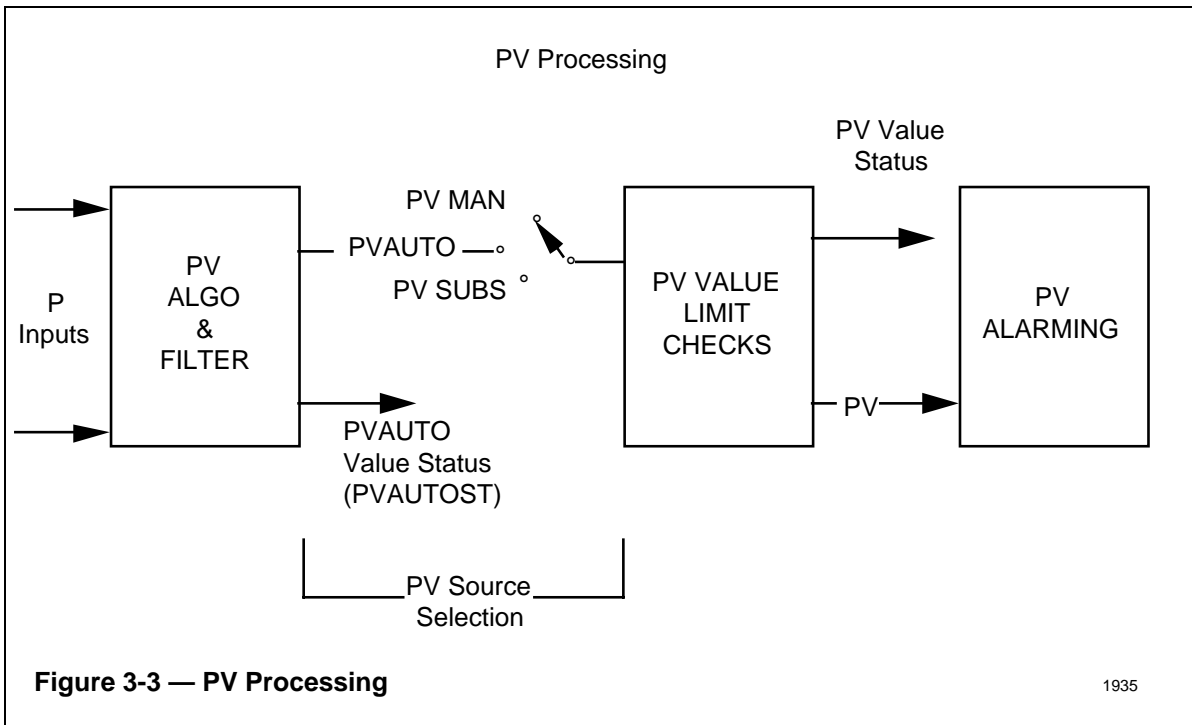
When the PV source is changed from Man or Sub to Auto, the PV immediately goes to the PVAUTO value. This might cause a bump in the value unless it is changed gradually to the value in PVAUTO before changing the source.

When the source is Man, only someone at a Universal Station can change the PV.

When the source is Sub, only a user-written program can change the PV. A program can store a bad value in PV, and if it does, PVSTS goes Bad.

You can prevent PV source changes by configuring OnlyAuto in PVSRCOPT. This fixes the source as AUTO and the parameter PVSOURCE is removed from the point. Configuring All in PVSRCOPT allows normal PV source selection.

If PVSRCOPT equals All and the control algorithm is PID, PIDFF, or PIDERFB, to change PV source requires the mode to be MAN and External Mode Switching must be disabled.



### 3.1.5.5 PV Value Limit Checks

The PV range is configured in PVEULO and PVEUHI in a range from 0% to 100.0% of the engineering-units range. This is usually the normal operating range for the PV but it can extend into a configurable extended range, defined by PVEXEULO and PVEXEUHI. The extended range is forced to be equal to or greater than the range defined by PVEULO and PVEUHI. The PV is generally constrained within PVEXEULO and PVEXEUHI, but the checks and constraints depend on the PV source, as follows:

- If **PVSOURCE** contains **Auto**—(1) If PVAUTOST contains Bad, PVSTS is Bad and no PV-range checks are made. (2) If PVAUTO is outside the extended range, and PV clamping (PVCLAMP) is not configured, the PV becomes a bad value, PVSTS is Bad; however, if PV clamping is configured, the PV becomes equal to the value of the violated range, and PVSTS becomes Uncertain. In either case, the appropriate PV-range violation flag is set (PVEXLOFL or PVEXHIFL).
- If **PVSOURCE** contains **Man**—Any value from the Universal Station that is outside the extended range is not accepted. PVSTS is already uncertain because the source is Man.
- If **PVSOURCE** contains **Sub**—If a program stores a PV value outside the extended range, the appropriate PVEXLOFL or PVEXHIFL flag is set. PVSTS is already uncertain because the source is Sub.

Data-point parameter LASTPV always holds the last good value of the PV.

### 3.1.5.6 PV Value Status

The value in the PV value-status parameter, PVSTS, is determined as follows:

- Normal** PVSOURCE = Auto, PVAUTOST = Normal, and the PV value is within the range defined by PVEULO and PVEUHI.
- Uncertn**
- 1) PVSOURCE = Man or Sub and the PV value = NaN (is a valid, real number), or,
  - 2) PVSOURCE = Auto, and PVAUTOST = Uncertn. Note that PVAUTOST contains Uncertn if at least one of the required algorithm inputs is Uncertn and none of the required algorithm inputs is Bad, or,
  - 3) PVSOURCE = Auto and the value in PVAUTO is outside the engineering-units range and is clamped.
- Bad** The PV value is NaN. This results from one of the following;
- 1) PVSOURCE = Auto and PVAUTO = NaN.
  - 2) PVSOURCE = Auto, the value in PVAUTO is out-of-range, and has not been clamped.
  - 3) PVSOURCE = Sub or Man and the PV is stored as NaN.

### 3.1.5.7 PV Alarm Checks

The following types of alarms are detected during PV alarm processing:

- PV High/Low
- PV High-High/Low-Low
- PV Rate-of-Change Positive/Negative
- PV Significant Change

PV source selection has no effect on alarm processing. For more detailed alarm information, see 4.3 in *System Control Functions*.

### 3.1.6 Setpoint Handling

Setpoint handling takes place only if the configured control algorithm requires a setpoint. Setpoint-handling functions are as follows:

- Access restrictions
- SP Target-Value processing
- Advisory Deviation-alarm processing
- Ratio and bias options

### 3.1.6.1 SP Access restrictions

The activities that can store a value in the SP are defined by Table 3-1.

**Table 3-1 — Setpoint Storage Access**

Mode	INACTIVE or INITMAN	PID Control Algorithm				Non-PID Control Algorithm			
		PVTRACK = Y		PVTRACK = N		EXTINTI = Y		EXTINIT = N	
		Attrb. =Oper	Attrb. =Prog	Attrb. =Oper	Attrb. =Prog	Attrb. =Oper	Attrb. =Prog	Attrb. =Oper	Attrb. =Prog
Man	No	I	I	Op	P	I	I	Cc	Cc
Auto	No	Op	P	Op	P	P	P	Op	P
Cas	No	Cc	Cc	Cc	Cc	Cc	Cc	Cc	Cc
Man	Yes	I	I	Op	P	I	I	Cc	Cc
Auto	Yes	I	I	Op	P	I	I	Op	P
Cas	Yes	I	I	Cc*	Cc*	I	I	Cc	Cc

Key: I = SP is being initialized.  
 Op = Operator at a Universal Station.  
 Cc = Input/Output connection or CL Block continuous-control access.  
 P = Access level = program.

\*INITTYPE must not contain Ext.

### 3.1.6.2 SP Target-Value Processing

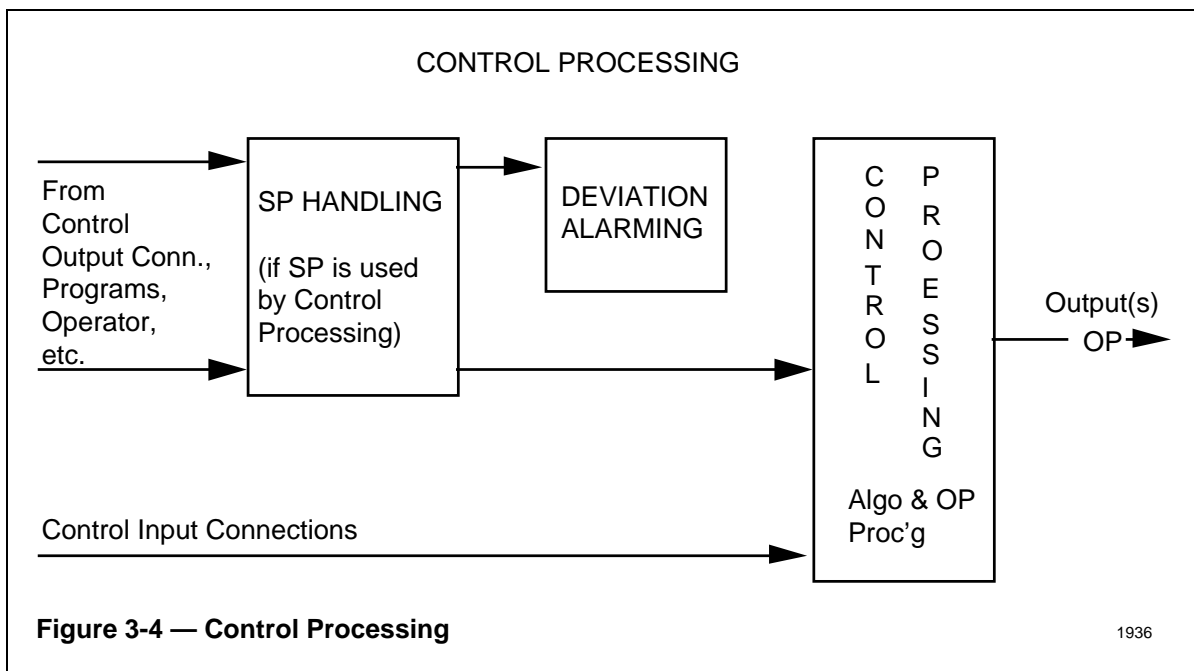
This option lets a Universal Station operator or a CL program "ramp" the setpoint from the current value to a new value over a period of time. The option is configured through the Data Entity Builder by entering TV in parameter SPOPT. If an operator is to ramp the SP, the mode attribute in MODATTR must be Oper, and if a CL program is to ramp the SP, the mode attribute must be Prog.

The operator does the following to use this option:

- Place Preset in the TVPROC parameter (point must be in Auto mode and INITMAN must contain Off).
- Enter the desired new SP value in SPTV.
- Enter the ramp time, in minutes, in RAMPTIME.
- Enter Run in TVPROC.
- The SP begins moving linearly toward the new value and the value in RAMPTIME decreases with time. When RAMPTIME = 0, SP reaches the new value, and the status in TVPROC changes to Off.

TVPROC can be changed to the Run state only from the Preset state. While TVPROC contains either Preset or Run; SP-high and SP-low limits, and the SP high and low engineering-unit ranges can't be changed. The following applies to TVPROC if it is in the Run state:

1. If MODE is changed to Man, state goes to Preset.
2. If MODE is changed to CAS, state goes to OFF.
3. If INITMAN is true, state goes to Preset.
4. A store to SP forces the state to OFF.
5. If  $SPTV > SP$  and ARWNET indicates that SP is wound HI or HILO, the SP stops changing. When ARWNET indicates that SP is no longer wound HI or HILO, SP ramping continues from the stop position. Note that when SP is ramping, ARWNET is not shown on the Group or Detail Displays. SP can normally be inferred from the output windup status. ARWNET can be accessed from a custom display.
6. If  $SPTV < SP$  and the output becomes wound LO or HILO, the SP stops changing.
7. If none of the above is true, the SP is ramped to SPTV.



### 3.1.6.3 Advisory Deviation Alarming

This option is used to allow an operator to manually change the setpoint (SP) to a predetermined value. The predetermined value is usually calculated by a user-written program that stores the value in the advisory setpoint parameter, ADVSP, rather than storing it directly in SP. Advisory-deviation alarming is selected by configuring Asp in the setpoint option parameter, SPOPT.

When this option is selected, an alarm is generated if the difference between the PV and the value in ADVSP is greater than the trip-point value in ADVDEVTP. This alarm returns to normal when the difference between the PV and ADVSP is less than the value in ADVDEVTP minus a deadband equal to 10% of the trip-point value.

The following conditions must be true if advisory-deviation alarming is to function:

- SPOPT = Asp
- ADVDEVTP • NaN
- The PV • NaN
- The alarm-enable status in ALENBST • Inhibit

If the advisory-deviation alarm is present and the value of one of the configuration parameters above is changed, the advisory-deviation alarm is cleared.

If parameter ASPPROC = Disable, the value in ADVSP equals the value in SP.

The alarm priority parameter for the advisory deviation alarm is ADVDEVPR.

#### 3.1.6.4 Ratio and Bias Options

The Ratio and Bias options apply to only PID control algorithms. For a more functional description of these options, see 18.4.11 in *Application Module Algorithm Engineering Data* manual.

These options are configured by entering one of the following values in parameter RBOPT:

- **FixRatBi**—Fixed Ratio and Bias
- **AutoRat**—Auto Ratio and Bias
- **AutoBi**—Fixed Ratio and Auto Bias

If one of these options is configured, the SP is modified before being used by the PID algorithm as follows:

$$\text{SP\_Store\_Value} * \text{RATIO} + \text{BIAS}$$

Where SP\_Store\_Value is the setpoint before the modification.

You can configure limits for both the RATIO and the BIAS values in the following parameters:

- RTHILM—Ratio high limit
- RTLOLM—Ratio low limit
- BIHILM—Bias high limit
- BILOLM—Bias low limit

In normal operation (CAS mode and INITMAN containing Off), all three options work alike. RATIO and BIAS can be changed by a Universal Station operator or by user-written programs (depending on whether MODATTR contains Oper or Prog). During initialization

of this point, however, RATIO and BIAS can be changed only if they are not being initialized, based on the value in RBOPT, as follows:

- **FixRatBi**—The initialization value calculated for this point's primary is

$$\text{INITVAL} = (\text{SP} - \text{BIAS})/\text{RATIO}$$

- **AutoRat**—RATIO is initialized as follows:

$$\text{RATIO} = (\text{SP} - \text{BIAS})/\text{Store Value}$$

If RATIO attempts to go outside one of its limits, it is clamped at the limit, and INITVAL is calculated as follows:

$$\text{INITVAL} = (\text{SP} - \text{BIAS})/\text{RATIO}$$

- **AutoBi**—BIAS is initialized as follows:

$$\text{BIAS} = (\text{SP} - \text{Store\_Value}) * \text{RATIO}$$

If BIAS attempts to go outside one of its limits, it is clamped at the limit, and INITVAL is calculated as follows:

$$\text{INITVAL} = (\text{SP} - \text{BIAS})/\text{RATIO}$$

### 3.1.7 SP/OP Tolerance Check

Release 530 provides a new function called “SP/OP Tolerance Check”; this functionality has two new parameters, called \$SPTOL and \$OPTOL, that allow a SP (setpoint) and OP (output) tolerance value to be configured by the engineer.

Manually entered SP and OP values for the AM, HG, and NIM Regulatory Control points, OP values for HG and NIM Analog Output points, and OP values for HG Analog Composite points are checked against this new specified tolerance. If the tolerance is violated in either a plus or a minus direction from the current set value, the operator is alerted with a beep from the keyboard and a warning message. Operator confirmation is required before the value is stored.

The tolerance check is made from the Detail Display, the Group Display, and in schematic actors RS\_SYS, CHG\_ZONE, and USER\_CZ.

### 3.1.8 AM Regulatory Data Point Alarms

Regulatory data points in AMs can generate the following types of alarms:

• General Alarms	Alarm Priority Parameter
Configuration error	CNFERRPR
Bad control	BADCTLPR
Bad PV	BADPVPR
CL error	CLEALMPR
Background CL error	BCLEALPR
CL fail	CLFALMPR
Background CL fail	BCLFALPR
• Control Alarms	Alarm Priority Parameter
Advisory deviation	ADVDEVPR
Deviation hi/lo	DEVHIPR, DEVLOPR
• PV Alarms	Alarm Priority Parameter
PV hi/lo	PVHIPR, PVLOPR
PV hihi/lolo	PVHHPR, PVLLPR
PV rate-of-change pos/neg	PVROCPPR, PVROCNPR
PV significant change	PVSCHPR

See 4.3 in *System Control Functions* for additional alarm information.

### 3.1.9 Limits in AM Regulatory Data Points

Limiting prevents certain variables from exceeding user-configured limit values. For limiting functions that apply in CL programs, see 4.4.1.2.

#### 3.1.9.1 Setpoint Limits

Setpoint limits prevent setpoint values from exceeding user-configured high and low limit values. These limits are configured in the same engineering units as the SP and must be within the SP range plus extensions. No crossover of setpoint limits is allowed. The configured setpoint limits also apply to the advisory-target value.

Configured setpoint limits are entered through the point builder and can be modified through a Universal Station by someone who has a Supervisor key. The limits default to SPEUHI +6.9% and SPEULO -6.9%. If NaN (not a number) is entered in one of these parameters, the value of its extended limit (SPEXEUHI or SPEXEULO) replaces the limit value.

Setpoint limits for regulatory points in an AM are observed in initialization calculations. If the limits are violated by the SP value, anti-reset windup-status propagation is invoked.

#### 3.1.9.2 Ratio Limits

You can configure high and low limits for the RATIO value that can be applied to the setpoint in PID points in an AM. A Universal Station user with a Supervisor key can



change these limits. An operator is not allowed to enter a value that exceeds these limits. If a user-written program attempts to store a value outside the limits, it is clamped to the limit. Crossover of these limits is prohibited.

### **3.1.9.3 Bias Limits**

Two user-setable high and low limits on the SP-related bias parameter are provided. Whenever operator or CL entries are outside the limits the entries are clamped to the closest limit. Crossover of limits is inhibited.

You can configure high and low limits for the BIAS value, that can be applied to the setpoint in PID points in an AM. A Universal Station user with a Supervisor key can change these limits. An operator is prohibited from entering a value exceeding these limits. A user-written program is clamped to the exceeded limit. Crossover of these limits is prohibited.

### **3.1.9.4 Output Limits**

You can configure high- and low-limit values for regulatory data points that are configured for control processing. These limits are expressed as a percentage of the output range and the limit values can range from -6.9% to 106.9%. A Universal Station user with a Supervisor key can change these limits. Crossover of these limits is not permitted.

These output limits apply to regulatory data points in Application Modules. For AM points whose output is in engineering units, the output limits are still expressed as a percentage of full range.

When an output limit is reached or exceeded, windup status is propagated up through cascade strategies.

### **3.1.9.5 Output Rate-of-Change Limits**

You can configure a maximum rate of change in percent-per-minute for output values of regulatory data points in an AM. The effect of this limit is to reduce excessive rates of change in the output, to the limit. The smallest limit can't be less than one percent per point-processing interval. The default value for the limit is NaN, which eliminates the limit check.

Typically, the output rate-of-change limit is configured for AM points doing DDC control through a slot in a process-connected box, and is used to match the slew rate of the final control element to the control dynamics.

### **3.1.9.6 Minimum Output-Change Limit**

You can configure a minimum output-change value for regulatory points in an AM. This value is a percentage of the output-value range. If the absolute difference between the output value at one processing pass and the next doesn't equal or exceed the minimum change, the earlier value is maintained. A Universal Station user with a Supervisor key can change the minimum output-change value.

The default minimum output-change value is NaN, which eliminates the minimum change check.

This feature is typically configured for AM points that are doing DDC control through a

slot in a process-connected box. It is used to minimize "wear and tear" on the final control device.

#### **3.1.9.7 Limiting in CAscade Mode**

SP and OP limits are observed when transporting data between points within the same box and between AMs. When transporting data from AM points to points in process-connected boxes, SP and OP limits are not observed unless the box slot contains a PIDDDC-type algorithm and the store is to SP. The stores to SP in this case obey operator rules; that is, they get clamped.

In process-connected boxes, these limiting functions remain as they were before introduction of the Local Control Network, and they are described in the appropriate Configuration Form Instructions.

#### **3.1.9.8 Limiting in Manual Mode**

No output limits, output rate-of-change or output minimum-change limits are observed while in manual mode. An indication is given to the operator if a manually entered or program-entered output-value exceeds the output limits.

#### **3.1.10 Initialization in AM Regulatory Points**

Initialization provides meaningful initial values in the data point parameters before processing is started or restarted. Separate mechanisms are provided to initialize PV-related parameters and control-related parameters.

The operator needs to know the value a primary point will initialize with when the secondary point is put back into control; they can see the initialize value of the primary point's output while in the initialize mode, not just at the time the initialization goes away.

For AM regulatory points with one output connection, continuous tracking will be provided during the initialization state from the secondary point. Points with more than one output connection will initialize to the first disposable secondary. The AM initialize logic matches the behavior of the NIM/PM in similar situations.

#### **NOTE**

The primary OP will not update when initialization exists and more than one control output exists. The initialization value is not known until the first disable secondary is available.

#### **3.1.10.1 PV Initialization**

The principal purpose of PV initialization is to set up starting values the first time the point is processed or the first time it is processed after recovering from a Bad PV value status. PV initialization is useful only for functions involved in history collection or for dynamically varying values. Stated another way, PV initialization is required where the new value depends on the previous value.

The following events cause PV initialization:

- The point's execution state is changed to Active.

- The AM undergoes a warm or cold restart.
- PVAUTOST recovers from Bad value status.
- A PV initialization request is made by a program (PVINIT contains On). CL Blocks can change PVINIT to force PV initialization.

PV initialization consists of the following functions:

- PVCALC is calculated from the P inputs, using the steady-state portion of the equation. For calculations that don't involve time, the normal equation is used.
- PVAUTO is made equal to PVCALC. If PV filtering is configured, the filter dynamics are initialized to steady state.
- No other PV processing is affected by PV initialization.

### 3.1.10.2 Control Initialization

Control initialization allows normal control strategies to be reestablished, after they have been interrupted, without "bumps" in the output to the process and without the need for manual balancing of values to avoid such bumps.

For control initialization, the control-processing blocks use their initialization procedures to compensate for changes that may have occurred since the normal control strategy was last operating. For example, a Universal Station operator might have taken over control of the output to the process, so that it now has a value that is different than normal processing would calculate. The initialization procedures automatically readjust either the bias value in the data point(s) or an input to the data point(s) so that when normal control is reestablished, the output to the process does not move or "bump." For most control algorithms, you can configure one of these three initialization choices:

- **None**—No initialization, so a bump can occur when normal processing resumes.
- **External**—A new value is back-calculated for an input that absorbs any output change. This value and an initialization request are sent to the primary data point that provides the input; thus, the primary absorbs the change and it must take similar action with its own primary, if it has one, so that the whole strategy can absorb the change.
- **Internal**—Each data point absorbs output changes. Usually, this is done by calculating a bias value that holds the output unchanged when the data point is put back in the control strategy.

By configuring a control output connection from one point (primary) to an initializable input to another point (secondary), an initialization path is created. It is along this initialization path that a value is transferred for use by a primary to absorb external process-upsets that may have occurred at the secondary.

Two or more active paths from a single primary to multiple secondaries are referred to as "fanout" connections. Where there are two or more control output connections from a

primary to two or more secondaries and all of these outputs are indisposable, the primary goes into the initialization state.

The value that is to be protected from a bump (the value to back-calculate from) is obtained at the point's output, at a secondary's initializable input, or at a secondary's output, depending on the type of control output connection (SPC or DDC) and according to where the upset occurred.

**General Mechanism**—At each point-processing pass, the following information is retrieved from all secondaries to which the data point has control output connections. The data that is retrieved to support initialization is:

- Initialization Request
- Initialization Value

An initialization request from the secondary causes the control output connection on the primary to go to an "output indisposable" state, a condition where a newly generated output to the secondary has no effect on the secondary. A control output connection also has "output indisposable" status if it is inactive or if a communication error has been detected when the initialization request and initialization value should have been received.

When all connections from a primary are in the "output-indisposable" state, the primary is forced to the initialized state. This causes the primary to set its initialization request, if it is configured for external initialization. In turn, this may cause the control output connection of a preceding primary to assume the "output indisposable" state and may force that primary to be initialized as well. In this manner, the initialization state is propagated upstream to all interconnected primaries.

During the initialization state, each point remains in that state until a disposable output connection is found. At the first processing pass when a disposable output connection is available, the primary sets its output value to the initialization value received from its secondary.

When initialization is correctly configured for a control strategy, each data point's output is readjusted by back-calculating an initializable input or by an internal (bias) adjustment. When each point resumes normal calculations, the new output and the input value at its secondary are balanced and no bump occurs.

Indisposable outputs are not the only factor for triggering initialization. It is possible to force a back-calculation by writing to a point's output while it is in MANual mode, or a CL program can set the control initialization flag in a point. For these cases, the point does not go to the initialization state but its primary may, because the point sent an initialization request to the primary.

**Conditions that Cause Control Initialization**—The need to initialize a data point in an AM is indicated by external upsets that directly affect the point or it is indicated by an initialization request from a secondary data point.

Control initialization is caused by any of the following:

- The point is in Man mode and goes to Cas or Auto mode.
- A CL program has requested initialization (see "Initialization Forced by a Program," below).
- The point is active for the first time (an inactive to active transition).
- The point is executing the first time after a warm AM restart.
- All control output connections were indisposable and now one or more output connections is disposable.

A control output connection is indisposable when

- A secondary has made an initialization request, or
- The control output connection is inactive, or
- A communication error has been detected in retrieving an initialization request and initialization value from a secondary.

The following are the reasons why a secondary data point sends an initialization request to its primary data point:

- The secondary isn't in CAS mode, or
- The secondary is inactive, has failed, or is doing only basic control (process-connected boxes), or
- The initializable input to the secondary (the destination of the primary's control output connection) is not selected for a secondary that uses the switch algorithm or the override selector algorithm, or
- The secondary is in the initialization state and is configured for external initialization.

**Initialization of a Point in the Same AM and the Same Process Unit**—For a primary and a secondary that are both in the same AM and the same process unit, an enhancement is provided. This enhancement presents the operator with an immediate indication of initialization when a cascade strategy is opened or closed. The following events cause initialization to be quickly propagated up through the data points in the strategy.

- The secondary is given special processing caused by the transition from CAS mode to some other mode.
- The secondary is given special processing caused by transition of the secondary from another mode to CAS mode.
- The primary is given special processing when the secondary is processed because of either of the two preceding events. This causes the primary to pick up the initialization state.

**Initialization Forced by a Program**—A user-written program can cause a data point to initialize by setting the data point's control initialization-request flag (CTRLINIT). The next time the point is processed, it initializes. CTRLINIT is cleared when the processing pass is complete. If the point is configured for external initialization, an initialization request is sent to its primary, thus propagating initialization up through the control strategy. Initialization requests are not propagated across unit boundaries.

**Limit Checking During Control Initialization**—Limit checks apply during initialization except for output rate-of-change limits.

**How Initialization is Indicated at Universal Stations**—When a data point that is initializing is displayed, the display indicates that it is initializing. For example, on a group display or a detail display, INIT appears in the point-status field that is just below the mode indicator.

### 3.1.11 Windup Protection in the AM

PID algorithms in the AM are protected from windup caused by reset action. Windup status parameters are maintained that pass the status "upstream" to the primary points along the initialization path. Each PID algorithm checks its output windup status and takes appropriate action to prevent reset windup. These functions are standard and require no configuration by the user.

#### 3.1.11.1 Windup Status

The following parameters contain windup-status information:

- ARWOP—Output (OP) windup status
- ARWNET—Windup status for SP or another initializable input.
- SECARW(n)—Windup status, as seen from this data point, of the secondary indicated by n, where n can range from 1 through the integer in NOCOPTS.

When this point's ARWOP contains something other than Normal, integral control in the windup direction stops. Integral action in the other direction and P and D action continue.

For the remainder of the control subsystem, the windup status serves only as a warning, and not as a constraint. For example if the status in ARWNET is Lo, lowering SP won't have an immediate effect on the output of the final secondary; however, SP can be lowered if the SP-low limit has not yet been reached.

The values in the windup-status parameters indicate whether raising or lowering the associated parameter value will affect the output of the final secondary, as it should. The values are as follows:

- Normal—Free to move in either direction
- Hi—Free to move only in the lower direction
- Lo—Free to move only in the upper direction
- HiLo—Not free to move in any direction

#### 3.1.11.2 Status Propagation

Windup status is propagated from SECARW(n) to ARWOP and ARWNET of the same point, and then from ARWNET of the secondary point to SECARW(1) of the primary point, and so on.

Propagation from secondary to primary is instantaneous if both are in the same process unit and the same AM. Otherwise, it takes place on the next processing pass for each point.

#### 3.1.12 Override Control

Override control strategies are primarily used for multivariable constraint control. In such a strategy, control of a single process variable is based on one of two or more PVs. These multiple PVs are referred to as "constraints." A selector algorithm selects the PV to be used for control. When an different PV is selected, the former PV has been overridden—the new PV has constrained the others.

Consider the following examples:

- In typical **boiler-control strategy**, the constraint on the fuel flow (SP of the fuel-flow PID controller) can be that it cannot exceed the actual air flow, multiplied by a ratio (which may be computed by the O<sub>2</sub> controller). In this case, the fuel flow is the primary PV to be controlled and the actual air-flow PV is the "constraint".
- In some **heating applications** it may be desirable to control the temperature of the feed (the primary PV) as well as possible, without ever letting the temperature of the hottest part of the pot (or the heat exchanger) exceed a safe limit. In this case, the safe limit on the pot temperature is the "constraint."
- In an **oven temperature-control** application, the temperatures can be measured at several places in the oven, and it may be desirable to control all these PVs by controlling one valve. One PID controller can be used for each PV, and the PID controller representing the PV that is farthest from its SP can be allowed to control the valve. In this case, there is no primary PV, since all the PVs have the same importance. Each PID controller represents a constraint on the other PID controllers.

Most often the objective is to achieve the best possible control of a PV without violating any of the constraints. The manipulated variable is driven by the output of an Override Selector algorithm that selects the highest or the lowest of up-to-four inputs. A PID that is in a cascade strategy, but is not selected, is prevented from winding up with the help of override initialization. In the rest of this section, the term "O/R selector" is used to mean an

Override Selector control algorithm that is configured for external initialization. See Section 23 in the *Application Module Algorithm Engineering Data*, for more about this algorithm.

### 3.1.12.1 O/R Status and Feedback

Regulatory data points contain the following parameters to support O/R strategies:

PTORST—This parameter contains the O/R status of the point as follows:

NotCon	The point is not connected to an O/R selector. Strictly, it means that this point is not on an initializable path to an O/R Selector or it is now being initialized. PTORST defaults to this value.
Sel	The point is a part of an O/R control strategy and is now selected.
NotSel	The point is part of an O/R control strategy and is not selected by the O/R selector.

When the point is returned from inactive to active status, when it undergoes a cold start, or when it is initialized, the status in PTORST becomes NotCon. This point resides on the primary.

ORFBSEC—This parameter contains the override feedback value that is sent to nonselected primaries. This is the initial output value that the primary is to use when it is next selected. This parameter resides on the primary and derives its value from the secondary.

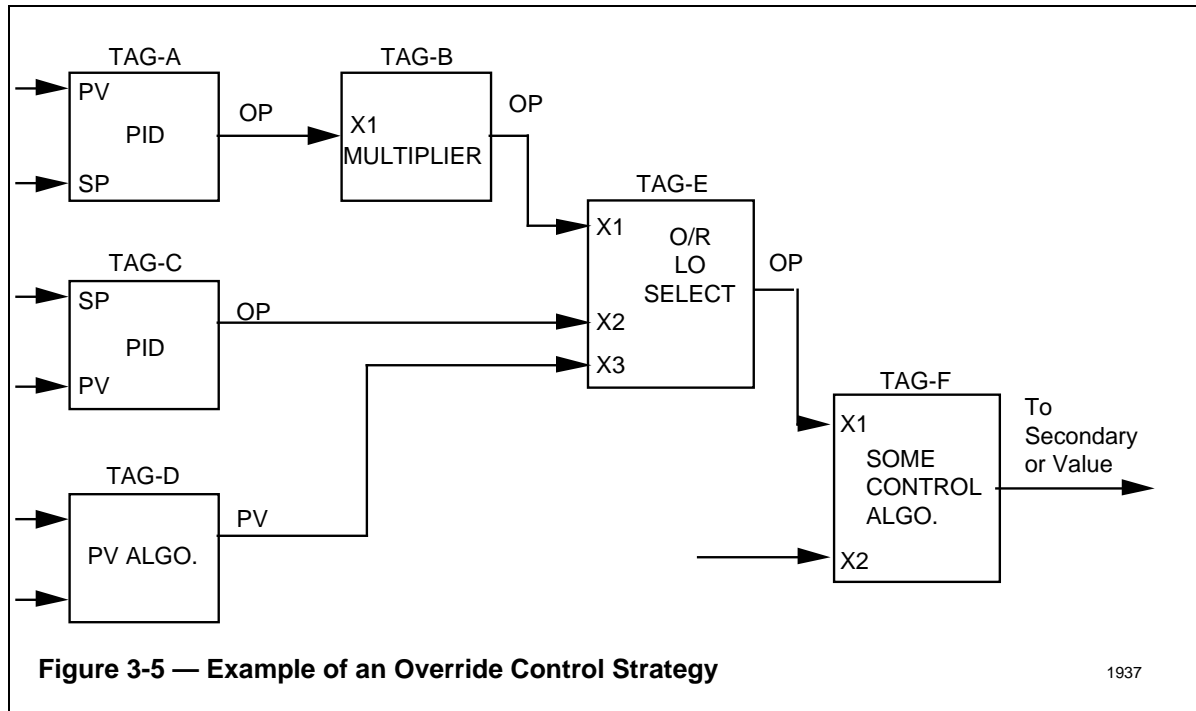
### 3.1.12.2 Processing in an Override Control Strategy

Figure 3-5 is an example of an O/R strategy. The override portion of the strategy includes the Override Selector point and all points "upstream" from it. Here is how O/R processing works:

- There must be at least one PID controller in the O/R strategy. In Figure 3-5 the points named TAG-A, TAG-B, TAG-C, TAG-D, and TAG-E constitute the O/R strategy.
- All points upstream of the O/R selector are processed on normal schedules. In the example, the points may be processed in TAG-A, TAG-B, TAG-C, TAG-D order. Their PV and control algorithms are executed normally.
- The next point to be processed is the O/R selector (TAG-E in the example). It selects one input. Assume input X2 is selected.
- The O/R selector then propagates appropriate O/R status to each one of its own initializing primaries. It also propagates the O/R-feedback value to the nonselected, initializing primaries. In the example, TAG-E propagates O/R status of "SEL" to TAG-C because input X2 is selected, and O/R-status NOTSEL to TAG-B. Further, TAG-E propagates the O/R-feedback value to the nonselected, initializing primary, TAG-B. TAG-D does not receive the O/R status nor the feedback value because it is not an



initializing primary when it is configured as a PV algorithm.



- Each primary (provided it is configured for external initialization and is in CAS mode), in turn, propagates O/R status to its own primaries (if any). It also propagates O/R feedback upstream, if it is not selected. In the example, TAG-B would propagate NOTSEL and an O/R feedback value to TAG-A.
- The propagation upstream continues until there are no more primaries. The output of any PID in a cascade chain, connected to a nonselected input of the O/R selector is initialized to override-feedback value, plus gain-times deviation (PV-SP). Since TAG-A contains a PID algorithm and it is not selected, it undergoes O/R initialization.
- The whole cycle is repeated.
- All points downstream of the O/R selector are processed at their specified interval and according to the configured before/after relationships, with respect to the O/R selector. In the example, TAG-F is processed after TAG-E.

### If there are multiple O/R selectors in a strategy

O/R-feedback propagation is initiated only by the most downstream O/R selector. In the example, if TAG-F were also an O/R selector, the O/R strategy would consist of points TAG-A, TAG-B, TAG-C, TAG-D, TAG-E, and TAG-F. O/R propagation would be initiated by TAG-F and not TAG-E, as before.

## Initialization in an Override Strategy

When a cascade is broken in an O/R strategy, initialization propagation supersedes O/R propagation. In the example, if point TAG-B is placed in MAN mode, it doesn't propagate O/R status or an O/R-feedback value to TAG-A. TAG-A's O/R status then is NOTCON.

### 3.1.12.3 Guidelines for Using Override Control

You should follow these guidelines to configure properly functioning O/R strategies:

#### Configuring for Initialization

The Override Selector point must be configured for external initialization. All points between the O/R selector and the final PID must be configured for external initialization. In the example of Figure 3-5, TAG-E must be configured for external initialization. TAG-B, must be configured for external initialization.

#### Proportional and Derivative Action on PIDs

While PIDs in an O/R scheme can be configured with proportional and derivative action on SP, use of these actions should be carefully considered because undesired results may occur, such as momentary oscillation caused by "kicks" in the error.

#### Scheduling

The O/R selector and all the points upstream of it, along initializable paths, must be scheduled for the same processing interval. Also, proper before/after relationships (i.e., primary before the secondary) must be specified for these points. TAG-A, TAG-B, TAG-C, TAG-D, and TAG-E, all must have the same processing interval and

- TAG-A must be before TAG-B
- TAG-B must be before TAG-E
- TAG-C must be before TAG-E
- TAG-D must be before TAG-E

Further, it takes one second to propagate O/R feedback from a secondary to the primary, and up-to-one second for the completion of the primary's processing. In order to assure that propagation of O/R feedback is completed following a normal processing pass and before the start of the next pass, all the points must have a sample time at least equal to  $(N+1)$  seconds, where  $N$  is the number of points in the longest cascade chain connected to the O/R selector (the most downstream O/R selector when there are more than one). In the example, the longest cascade chain consists of TAG-A, and TAG-B. It would take three seconds; two seconds for propagation of O/R status and feedback to TAG-A, and up to one second for processing of TAG-A, after TAG-E (the O/R selector) is processed. Each point in the O/R strategy in the example must be scheduled to run at intervals greater than 3 seconds.

## Unit and Module Boundaries

O/R feedback can't be propagated beyond a process unit or outside an Application Module. The entire O/R strategy must be within one process unit and within one AM.

### Fanout:

No fanout control output connections are allowed in an O/R strategy. All primaries in upstream from the O/R-selector point can have only one control output connection.

### 3.1.13 Control Output Processing in AMs

The primary task of control output processing in an AM is to make the control algorithm calculation available to the rest of the system (displays, printers, CL programs, other data points, etc.) in percent or EUs, as needed. This section describes control output processing as it applies to regulatory points in AMs.

Control algorithms produce outputs in percent or in engineering units. A different type of output processing is required to support each case. For standard control algorithms, the appropriate control output processing is automatically selected by the system; however, for user-written algorithms (written in CL), you must specify the type of control output processing to be used. This is specified in parameter CVTYPE.

Control Output Processing does the following:

- Provides the output value in percent and EUs for displays, printing, CL programs, and interpoint communications.
- Processes control output connections that send the output to the secondary data point after conversion to EUs and percent.
- Constrains the output to the configured limits.
- Generates and propagates windup status if there is a violation of an output limit.

The following parameters contain significant output-processing information. All of them are accessible for displays and printing, and all can be accessed by programs written in CL, Pascal, and FORTRAN.

- CV = The result of calculation of the control algorithm, can be in percent or EU depending upon the control algorithm.
- OP = The final control output, expressed as % of EU Range of the output.
- OPEU = Final control output in Engineering Units.
- CVEUHI = Output EU Range corresponding to 100% value of OP.
- CVEULO = Output EU Range corresponding to 0% value of OP.
- CVTYPE = Type of CV (PERCENT or EU)

Note that for standard control algorithms, CVTYPE is determined by the system and can't be specified by the user; however, for CL control algorithms, you must specify CVTYPE.

### **3.1.13.1 Initial Control Processing**

During initial control processing, initialization data is fetched from the secondary points indicated by each control-output connection. If a communication or configuration error is encountered, the output-connection status is updated.

The output engineering-unit range in CVEULO and CVEUHI is determined based on the EU range of the secondary pointed to by the first active control-output connection. If this connection has a communication or configuration error, the output range is set to bad and the point is aborted. If this happens, the CI connections, CO connections, and any CL blocks scheduled for execution after initial control processing are not processed and their status is not updated.

As an example of such a situation, consider a CI connection that is acquiring a PV from an NIM or HG with a CO connection through the same NIM or HG. If that NIM or HG fails, initial control processing encounters a communication error when it fetches initialization data, so CV range is set to NaN, parameter COACCSTS is set to Config, and processing of the point is aborted. Parameter CIACCSTS retains a value of NoError, from the previous processing pass.

### **3.1.13.2 Control Output Processing for CV In Percent**

For CV in EUs, see 3.1.13.3.

## Normal Computation

1. CV is calculated by the control algorithm.
2. OP is set equal to CV after checking output minimum-change, output rate-of-change, output high-limit, and output low-limit. If any limits are exceeded, the OP value is adjusted or clamped, as appropriate. The calculated CV value is not clamped or adjusted.
3. The windup-status is set (see 7.5.10).
4. Inactive control output connections are not processed. Each active control output connection is processed as follows:

If the output is disposable,

The corresponding secondary value is made equal to OP, if the destination parameter is OP.

For all other destinations, the value in OP is converted to the destination-parameter's EU range and stored, subject to the applicable limits of the destination parameter (e.g., SP limits).

## In MANual mode

1. OP is set (in percent) by the operator or by a CL program, as determined by the data point's attribute.
2. Inactive control output connections are not processed.

Remaining process is as previously defined under "Normal Computation."

When initializing because of a request from a secondary data-point:

1. If all of the control output connections are indisposable, CV remains at its last value. When the first disposable control output connection becomes available, the CV is back-calculated (an EU to % conversion if the destination parameter is an input) from the destination parameter of the secondary.
2. After output high and low limit checks are made, OP is set equal to CV.

### 3.1.13.3 Control Output Processing for CV in EUs

For CV in %, see 3.1.13.2.

#### Normal computation

1. CV is calculated by the control algorithm.
2. OP is calculated from CV, with the units based on the range defined by CVEULO and CVEUHI. Note that the values in CVEULO and CVEUHI are determined by the engineering-units range of the secondary point to which the first active connection is made.
3. OP is checked for minimum output change, output rate-of-change, output high-limit, and output low-limit. If any of the limits is exceeded, the OP value is adjusted or clamped as applicable.
4. The windup status is set.
5. Inactive control output connections are not processed. Each active control output connection is processed as follows:

If the control output connection is disposable,

the corresponding value in the secondary point is made equal to OP, if the destination parameter is OP, otherwise it is made equal to OPEU.

the OP or OPEU value is transferred to the secondary's destination parameter, subject to applicable limits on the destination parameter (e.g., SP limits).

#### In MANual Mode

1. OP is set (in percent) by an operator or by a CL program, depending on the attribute.
2. Inactive control output connections are not processed. Each active control output connection receives normal processing, as described above. Remaining process is as defined under "Normal Computation."

#### When initializing because of a request from a secondary data-point

1. If all of the control output connections are indisposable, CV is initialized to its last value. When the first disposable control output connection becomes available, the CV is back-calculated (% to EU conversion if the destination parameter is OP) from the destination parameter of the secondary.
2. OP is calculated from CV, based on CVEULO and CVEUHI.

### 3.1.14 Functional Summary Chart for AM, and for CB, MC, and EC

Regulatory data points in UCN nodes and in CBs, MCs, and ECs on the Data Hiway have many of the same or similar features to those of regulatory points in an AM. The following chart compares these functions.

<b>Regulatory Data-Point Function</b>	<b>CB, MC, EC</b>	<b>AM</b>	<b>PM/APM</b>
• Point Interconnections	x		
• <b>Input Connections</b>		x	x
- PV Inputs		x	
- Control Inputs		x	
- General Inputs		x	

**Notes:**                    x    =    Function provided  
                                  Blank    =    Function not available

(Continued)

## Regulatory Point Functions Chart (Continued)

Regulatory Data-Point Function	CB, MC, EC	AM	PM/APM
<b>Output Connections</b>			x
- Control Outputs		x	
- General Outputs		x	
• Connections Active/Inactive		x	
• <b>Alarm generation</b>			
- Advisory Deviation		x	
- Bad Control		x	x
- Bad PV	x	x	x
- Dev High	x	x	x
- Dev High, High	EC		
- Dev High, High, High	EC		
- Dev Low	x	x	x
- Dev Low, Low	EC		
- Dev Low, Low, Low	EC		
- PV High	x	x	x
- PV High, High	EC	x	x
- PV High, High, High	EC		
- PV Low	x	x	x
- PV Low, Low	EC	x	x
- PV Low, Low, Low	EC		
- PV Significant Change		x	x
- PV Rate of Change	EC,BC		
- PV Rate of Change Negative		x	x
- PV Rate of Change Positive		x	x
- Logic Out as an alarm (Status 1,3,4)	EC		
- Ramp/Soak Mark1,2	EC		
- Ramp/Soak Offset1,2,3	EC		
• Alarm Cutout	HG	x	x
• Alarm Disable	HG	x	x
• Alarm Inhibit	HG	x	x
• Unit Alarm Disable		x	
• Unit Alarm Inhibit		x	
• Alarm Level (per type)	HG	x	x
• Alarm Priority (per parameter)	HG	x	x
• Control Processing	CB,EC,BC	x	x
• Point Execution States		x	x
• Point Active/Inactive		x	x
• Mode Switching	EC(limited)		
• Mode Permissive		x	x
• CL Switches		x	
• <b>Limiting</b>		x	x
- PV Range Limit		x	x
- PV Range Extension		x	x
- PV Range Clamping		x	x
- SP High Limit		x	x
- SP Low Limit		x	x

**Notes:** x = Function provided  
Blank = Function not available  
CB, MC, EC = Function provided in that box type

(Continued)



## Regulatory Point Functions Chart (Continued)

Regulatory Data-Point Function	CB, MC, EC	AM	PM/APM
- Output High Limit	x	x	x
- Output Low Limit	x	x	x
- Output Rate-of-Change Limit		x	x
- Minimum Output Change		x	x
- Integral Limit High		x	
- Integral Limit Low		x	
• <b>Modes</b>			
- MAN	x	x	x
- AUTO	x	x	x
- CASC	x	x	x
- BCASC	x		
• Attributes: Prog, Oper	MC, HG	x	x
• Normal Mode	x	x	x
• Cascade Request	EC, HG	x	x
• Time-out Gate	x		
• PV processing	EC, HG	x	x
• PV Source selection	x	xx	x
• PV Tracking	x	x	x
• PV Value Status	x	x	x
• Red Tag		x	x
• Reverse/Direct Control (for PID only)	x	x	x
• PV Target Value	x	x	x
• SP Target Value		x	x
• <b>Point Scheduling</b>		x	
- Processing period		x	
- Optional Before/After		x	
- Optional Processing Cycle		x	
- By Scan Rate			x

**Notes:**            x    =    Function provided  
                          Blank   =    Function not available  
                          CB, MC, EC   =    Function provided in that box type

## 3.2 AM NUMERIC DATA POINTS

A Numeric data point in an Application Module stores a real number that can be changed by an operator at a Universal Station or by a user-written program. Numeric data points are used to store data, such as a value that is needed later or recipe data for a batch operation.

### 3.2.1 AM Numeric Point Functional Description

Data-point parameter PV stores the real-number value. PV stores a single-precision, 32-bit, real number in accordance with the IEEE standard for floating-point values. When not limited, PV can store any representable value in a range from +infinity to -infinity, or NaN (not a number).

Though the values stored in PV can be within this very large range, there are limits to the number of significant digits that can be shown on Universal Station displays and on printed logs and reports. Further, parameter PVFORMAT is configured to specify the number of decimal digits to display, in a range from 0 to 6.

### 3.2.2 Limiting PV Range in AM Numeric Points

Parameters RANGEHI and RANGELO can be configured to limit the range of PV values, as indicated in the chart, below. These parameters can also be changed by someone at a Universal Station with an engineer's key, or by a user-written program. If NaN (not a number) is placed in RANGEHI or RANGELO, the corresponding range limit is not in effect.

<b>RANGEHI, RANGELO Values</b>	<b>PV Range</b>
RANGEHI = NaN RANGELO = NaN	Unlimited, as described above
RANGEHI ° NaN RANGELO = NaN	From the largest negative value to the value in RANGEHI
RANGEHI = NaN RANGELO ° NaN	From the value in RANGELO to the largest positive value
RANGEHI ° NaN RANGELO ° NaN	From the value in RANGELO to the value in RANGEHI

Any attempt to store a new value from a Universal Station that exceeds a configured limit is rejected and an "out-of-range" message is displayed. An attempt by a user-written program to store a value beyond a configured limit results in the value being clamped at the limit value, and a "value-clamped" indication being returned to the program.

#### **NOTE**

Refer to the *Application Module Parameter Reference Dictionary* for further information on the parameters mentioned in this section.

### 3.2.3 AM Numeric Point Processing

AM Numeric data points are not scheduled and are not processed at regular intervals. The only activity of one of these points is to accept new PV values from a Universal Station or a user-written program, subject to the limiting or clamping, as defined under 3.2.2.

### 3.2.4 AM Numeric Point Restarts

The database for a Numeric data point is restored from its checkpoint file in a History Module, during cold, fast, and hot restarts.

### 3.2.5 AM Numeric Point Configuration

The configuration forms and Parameter Entry Displays for AM Numeric data points show several default values, which are entered when the point is built through the Data Entity Builder unless the default value is changed. Any parameters that don't have default values must be entered or the point can't be loaded into the AM.

The following are the configurable parameters for an AM Numeric data point (see the *Application Module Parameter Reference Dictionary* for further parameter details):

NAME—The Tag Name for the Numeric data point (no default value)

UNIT—The ID of the process unit that contains this point (no default value)

PTDESC—The description of this data point (default value = all blanks)

KEYWORD—The keyword for this data point (default value = all blanks)

PRIMMOD—The process module associated with this point (the default value represents a "null" tag name)

EUDESC—The engineering-units descriptor for the PV (default value = all blanks)

RANGELO—The PV-low limit (see 3.2.2) (default value = NaN)

RANGEHI—The PV-high limit (see 3.2.2) (default value = NaN)

PVFORMAT—The number of decimal places in displayed and printed PV values (default = D0, or nnnnnn.)

PV—The initial value for this point's PV (default = 0.0)

### 3.2.6 AM Numeric Point Error and Warning Messages

The following error or warning messages may be returned when attempts are made to store new values in AM Numeric data point parameters:

- **Error Messages**

Limit or Range Exceeded—The newly stored value exceeded a range limit.

Limit or Range Crossover—The newly stored range-limit value crossed over the opposite range-limit value.

Read-Only Parameter—An attempt was made to store a value in a parameter that can only be read.

- **Warning**

Value-Clamped Error—An attempt was made by a user-written program to store a value beyond one of the PV range limits, so the PV was clamped at the limit value.

### 3.3 AM COUNTER DATA POINTS

Counter data points are used for flow or speed measurement, accumulation of total flow for volumetric control, and event counting. An AM Counter data point receives a count from a Control Counter data point in an HG, by means of an input connection to the AM counter point (the count source for the HG's Control Counter is a 32-bit counter in an HLPIU). The AM Counter point computes a rate value for its PV parameter and an optional accumulation value for its AV parameter (displayed as OP).

#### 3.3.1 AM Counter Point Functional Description

While parameter PTEXECST indicates that the AM Counter point is active, the following data is acquired (prefetched) through the input connection from the Control Counter point in the HG:

- **Delta Counts**—The difference in the count value between two sample periods, computed by the Control Counter.
- **Delta Time**—The difference in time, in milliseconds, between the two sample periods of the Control Counter, calculated by using the HLPIU's sequence-of-events clock.
- **Total Counts**—The latest sampled value of the Control Counter's count, which is held in a 32-bit binary counter.

<b>NOTE</b>
-------------

Refer to the *Application Module Parameter Reference Dictionary*, for further information on the parameters mentioned in this section.

- **Rollover Threshold**—The 32-bit HLPIU counter's rollover threshold value, from the PIU's database.
- **Point State**—The worst state of the HLPIU counter since the last time the AM read from the Control Counter. This detects any reset of the HLPIU, which resets the HLPIU's counters.
- **Delta State**, which indicates the validity of Delta Counts and Delta Time. Delta Count and Delta Time are valid only when the HG has had two, successive, good readings of data from the HLPIU.

The HG updates this information in the Control Counter data point every ten seconds. When the AM reads from the Control Counter point, the Point State value in the Control Counter is set to OK.

Each time the AM Counter data point is processed the following functions occur, in the order they are listed:

- **General Input Processing**, if a general input connection is configured. This connection is similar to general input connections for Regulatory data points (see 3.1.4.4)
- **Contact Cutout Processing**, if a contact cutout connection is configured. This function is similar to the contact cutout function for AM Regulatory data points (see 3.1.4.3 in this publication and 4.3.1.7 in *System Control Functions*).
- **Counter Input and PVAUTO Calculation** (see 3.3.3)
- **PV Source Selection and Range Checks**. These are the same as for Regulatory data points (see 3.1.5.4 and 3.1.5.5).
- **PV Alarm Processing** (see 3.3.4)
- **Deviation Alarm Processing** (see 3.3.5)
- **Accumulation Processing** (see 3.3.6)
- **General Output Processing**, if a general output connection is configured. This function is similar to general output processing for AM Regulatory data points (see 3.1.4.6).
- **Alarm Distribution Processing**

### 3.3.2 Scheduling of AM Counter Points

AM Counter data points are scheduled as described under 2.2.

### 3.3.3 Counter Input and PVAUTO Calculation

A value for AM Counter parameter PVAUTO is calculated only if all three of the following are true:

1. The counter input connection-activity status in CIACTSTS is Active. If it is Inactive, the HG's Control Counter data is not processed.
2. The counter input-access status in CIACCSTS is NoError. This status indicates that no error was detected during the last acquisition of data from the Control Counter.
3. Delta State received from the Control Counter is OK.

If these conditions are true, the PVAUTO value and PVAUTO-value status are calculated as follows:

$$PVAUTO = \frac{\text{Delta Counts}}{\text{Delta Time}} * PVCONV$$

and

$$PVAUTOST = \text{Normal}$$

The PVCONV factor converts the rate expressed in counts per millisecond to an engineering-units representation. The Delta Counts and Delta Time values are available in parameters PVCOUNTS and TIMELPSD.

If any of the three conditions above are not true, the PVAUTO value is not calculated and PVAUTO contains NaN. PVAUTOST then contains Bad.

If the status in CIACCSTS is Config, a configuration error has been detected, the configuration-error flag in CNFERRFL goes to On, and a configuration alarm is generated.

When Delta Status is not OK, the HG has not processed two successive reads from the HG's Control Counter. This condition should occur only, very briefly, following an HG startup or an HLPIU startup.

If the PVSOURCE is AUTO, the value Last PV (LASTPV) represents the last "good" calculated PV; i.e., if PVAUTO is not equal to NaN, LASTPV is set equal to PV before determining the new PV.

The PV value status (PVSTS) is determined exactly as it is for regulatory points. See 3.1.5.6.

Once a PV and a PVSTS are computed, the PV is checked against the configured PV Extended Ranges (PVEXEUHI and PVEXEULO). PV and PVSTS can be altered according to the PV clamping option (PVCLAMP) if the PV is outside the extended ranges. Parameters PV-EU extended-high (PVEXHIFL) and extended-low (PVEXLOFL) flags signify if the PV has violated a given extended-range limit.

The PV is accessible in percent-of-range (PVEUHI minus PVEULO) through the parameter PV percent (PVP).

### 3.3.4 PV Alarms in AM Counter Points

Parameter PV receives its value from the PV Source Selection and PV Range Checks, which are the same as for AM Regulatory data points (see 3.1.5.4). If the PV Source is PVAUTO, the value in PVAUTO is placed in PV.

The PV value can be checked for the following types of alarms:

Alarm Type	Alarm Priority Parameter
PV High	PVHIPR
PV High High	PVHHPR
PV Low	PVLOPR
PV Low Low	PVLLPR
PV Rate-of-Change Positive	PVROCPPR
PV Rate-of-Change Negative	PVROCNP
PV Significant Change	PVSGCHPR
Bad PV	BADPVPR

PV alarm processing and related parameters are identical to those for AM-Regulatory data points (see 3.1.5.7 in this publication and 4.3 in *System Control Functions*); however, the AM-Counter data point doesn't have alarm-transition parameters.

### 3.3.5 AM Counter Point Deviation Alarming

If a PV target value is configured in parameter PVTV, deviation alarms can be monitored by an AM counter point. The deviation, DEV, is calculated as follows:

$$DEV = PVTV - PV.$$

The value in DEV is compared with the deviation-high trip point in DEVHITP and the deviation-low trip point in DEVLOTP. The alarm is generated if DEV equals or exceeds either of the trip points. The alarm priority parameter for the deviation-high trip point alarm is DEVHIPR. The alarm priority parameter for the deviation-low trip point alarm is DEVLOPR.

AM counter deviation-alarm functions are the same as for other AM regulatory points except that the AM counter doesn't have alarm-transition parameters.

If you do not configure a control algorithm, you may configure the target value for PV (PVTV). This parameter is shown on the Group and Detail displays as the setpoint.

### 3.3.6 Accumulation Processing in AM Counter Points

Accumulation processing in an AM counter point takes place only if the following conditions are true:

- The input connection-activity status for the input from the HG control counter, as indicated by parameter CIICTSTS, is Active and the HG counter-input data is processed.
- The access status for the HG control-counter input, as indicated by parameter CIACCSTS, is NoError. This parameter indicates any error that is detected when the input is prefetched for the AM counter.
- The HG control-counter point state, which is prefetched from the HG, is OK.
- The AM counter's accumulator state, as indicated by parameter STATE, is Running.

During accumulation processing, the total count read from the HG control counter, in the prefetch for the last sample, is subtracted from that read for the current sample. This difference is added to the AM counter's accumulation value, as counts. An engineering-unit conversion factor in parameter AVCONV is applied to yield the accumulation value, as follows:

$$AV = AVCOUNTS * AVCONV$$

where AVCOUNTS is a count value, AVCONV contains the engineering-units-per-counts factor, and AV is in engineering units.

The AV is stored as a 32-bit real number. In the AM, this value is internally maintained as a 48-bit binary representation to maintain accuracy and to expand the range of conversion factors. The HLPIU control counter maintains a 32-bit count value.

The value in AV can range from zero to 999,999. If the conversion results in a number greater than 999,999, the value in parameter ROLLOVER is incremented by one, the value in AV "rolls over" to zero, and begins accumulating again, on each processing pass. When ROLLOVER is incremented to 32,767 and AV exceeds 999,999, AV is set to NaN. In this case, use the three steps below to recover.

Parameter LASTAV maintains the last, good accumulation value, so if the value in AV goes to NaN, LASTAV contains the last value before AV became NaN.

If STATE contains Stopped, accumulation processing does not occur.

If the Point State received from the HG Control Counter is not OK, the HLPIU has failed between two prefetches by the AM. During this period, the value in AV is not valid so the value in AV is changed to NaN and a bad-control alarm is generated (BADCTLFL indicates this alarm condition). In this situation, an operator at a Universal Station must intervene to do the following:



1. Stop the accumulator (change STATE to Stopped).
2. Store a correct value in AV. AVCOUNTS is automatically recomputed (the last good accumulation value may be available in LASTAV).
3. Start the accumulator (change STATE to Running).

If CIACTSTS contains InActive, indicating that the counter input connection is not active, and STATE indicates that the accumulator is Running, an unrecoverable situation has occurred and a bad-control alarm is generated. The AM can't determine what the Point State in the HLPIU is, so it sets AV to NaN and generates a bad-control alarm. The operator can then try to recover, as in the three steps, above.

The rollover threshold in the HLPIU database is configured with a value of zero when the HG Control Counter data point is built and must always be zero for the AM Counter to properly operate (with this threshold set to zero, rollover in the HLPIU occurs at  $2^{32} - 1 = 4,294,967,295$  counts). The AM detects rollover when the difference in total count over two samples is a negative value, and it compensates accordingly. If the rollover threshold in the HLPIU is not zero, the HG Control-Counter Point State indicates Failed, so the AM places NaN in AV and a bad-control alarm is generated. Again, the operator can attempt recovery as described above.

### 3.3.7 AM Counter Point Accumulation Alarming

The accumulation value in AV is checked for the following alarm conditions:

Alarm Trip Point	Alarm Condition	Alarm Priority Parameter
Pre-Pre-Preset	AV equal to or greater than (AVTV - AVDEV2TP)	PPPRSTPR
Pre-Preset	AV equal to or greater than (AVTV - AVDEV1TP))	PRPRSTPR
Preset	AV equal to or greater than AVTV	PRESETPR

Where AVTV is the accumulation-target value,

AVDEV2TP is the second deviation-alarm trip point, and

AVDEV1TP is the first deviation-alarm trip point.

If any of these three parameters contains NaN, the alarm condition in which it is used is not checked.

AVTV, AVDEV2TP, and AVDEV1TP can have initial, configured values. AVTV can be changed by an Operator at a Universal Station and AVDEV2TP and AVDEV1TP can be changed at a Universal Station by someone with a Supervisor's or an Engineer's key. These parameters can also be changed by a user-written program.

Note that the first two alarms are defined as a deviation of AV from the target value in AVTV.

The AM Counter data point includes the following three parameters to indicate the presence of the corresponding alarm (On indicates an alarm, Off indicates no alarm):

AVDEV2FL

AVDEV1FL

AVTVFL

Alarm transition parameters are not provided.

### **3.3.8 Accumulator Control in AM Counter Points**

The AM Counter data point's accumulator is controlled through the parameters listed in Table 3-2. In most cases, the values of these parameters can be changed by someone at a Universal Station or by a user-written program. The table defines the effects of the parameter changes and all restrictions to those changes.

**Table 3-2 — Accumulator-Control Table**

Parameters	Values	Restrictions and Results
<b>STRTSTOP</b>  Normal Access Level: Operator	Start	Restrictions: Operator access not permitted when CNTLLOCK = NotPerm  Resultant parameter values: STRTSTOP = Start STATE = Running, if PTEXECST = Active STATE = Stopped, if PTEXECST = InActive  Special Notes: If PTEXECST = Active the counter point is immediately processed to acquire an initial total-counts value, which is used to calculate the difference in counts between two process-passes.
	Stop	Restrictions: Operator access not permitted when CNTLLOCK = NotPerm  Resultant parameter values: STATE = Stopped STRTSTOP = Stop  Special Notes: If PTEXECST = Active and the previous state was Running, the Counter point is immediately processed to compute the final AV, AVCOUNTS, and ROLLOVER values, and to check for accumulator alarms.
<b>RESETCMD</b>  Normal Access Level: Operator	Reset	Restrictions: Operator access not permitted when CNTLLOCK = NotPerm  Resultant parameter values: AV = 0 AVCOUNTS = 0 ROLLOVER = 0 RESETCMD = Reset, momentarily, then NotReset AVTVFL = Off AVDEV1FL = Off AVDEV2FL = Off
	NotReset	No restrictions

(Continued)

**Table 3-2 — Accumulator-Control Table (Continued)**

Parameters	Values	Restrictions and Results
<b>PTEXECST</b>  Normal Access Level: Supervisor	Active	No Restrictions  Resultant parameter values: PTEXECST = Active STATE = Running, if STARTSTOP = Start STATE = Stopped, if STARTSTOP = Stop All Accumulator alarms are cleared. RESTART = PtActvn
	InActive	No Restrictions  Resultant parameter values: PTEXECST = InActive PV = NaN (unless PVSOURCE = Man) PVSTS = Bad (Uncertain if PVSOURCE = Man) All PV alarms are cleared All accumulation alarms are cleared STATE = Stopped STRTSTOP = Stop AV = 0.0 ROLLOVER = 0 AVCOUNTS = 0.0
<b>AV or ROLLOVER</b>  Normal Access Level: Operator	A new accumulation value or ROLLOVER value	Restrictions: US or program cannot change AV or ROLLOVER when STATE = Running. US or program cannot store NaN to AV or to ROLLOVER. US or program cannot store a new value in AV that is > 999,999. US or program cannot store a new value in ROLLOVER that is > 32,767. New values for AV from other data point or user-written programs are clamped at 999,999.  Resultant parameter values: AV = the new accumulation value, or the clamped value, 999,999. ROLLOVER = the new rollover value.

(Continued)

**Table 3-2 — Accumulator-Control Table (Continued)**

Parameters	Values	Restrictions and Results
AVTV  Normal Access Level: Operator	A new accumulation target value	<p>Restrictions:</p> <p>If AVTVLOCK = NotPerm, an operator at a Universal Station can't change AVTV.</p> <p>If the new AVTV value is &gt; 999,999, new values from another data point or from a user-written program are clamped at 999,999.</p> <p>Resultant parameter values</p> <p>AVTV = the new target value, or the clamped value, 999,999.</p> <p>Special Notes:</p> <p>When the AVTV value is changed, the alarm-trip points are automatically readjusted; however, the alarm state doesn't change until the next processing pass.</p>

### 3.3.9 Other AM Counter Functions

When the control-lock parameter CNTLLOCK contains NotPerm, Universal Station operators can't change the STRTSTOP and RESETCMD parameter values, but someone at a Universal Station with a supervisor's or an engineer's key can change these values, as can user-written programs.

When the accumulation-target value-lock parameter AVTVLOCK contains NotPerm, Universal Station operators can't change the AVTV value, but someone at a Universal Station with a supervisor's or an engineer's key can change the AVTV value, as can user-written programs.

The AM Counter data point can be forced to be processed on demand by changing the value in the point processing-special parameter PPS to On. The special processing pass can be set up to occur sometime later, by first placing an integer representing the number of seconds (up to 86,400 seconds or 24 hours) in PPSCYCLE, and then changing PPS to On. The access level for PPS is Operator, and for PPSCYCLE it is Program.

Parameter PTINAL indicates if the counter point has an alarm (PTINAL = On) or not (PTINAL = Off).

### 3.3.10 Display- and History-Related Parameters in AM Counters

The following parameters contain information related to the representation of the AM Counter point of Universal Station displays and in the process-history files:

- AVFORMAT—Defines the decimal-point position in the AV value.
- DISPTYPE—Defines the format to be used on the Group and Detail displays for this data-point type (Counter).
- EUDESC—Defines the engineering-units description to appear with the PV and AV values.
- HIGHAL—Indicates the highest level alarm for this point, which is either BadCtl, BadPV, CComer, Cnferr, PVHi, PVHH, PVLo, PVLI, PVSgch, PVRocn, PVRocp, DevHi, DevLo, Preset, PrPreset, PPPreset or NoAlarm.
- HIGHALPR—Priority of the alarm value that is contained in HIGHAL.

### 3.3.11 AM Counter Point Restarts

When an AM Counter point is restarted using its database from a checkpoint file in an HM, the counter data is initialized, as described in the following paragraphs. This initialization takes place only if the PTEXECST value from the checkpoint file is Active.

For cold, warm, and hot restarts, if the accumulation option is configured (ACCUM = On),

- STATE is changed to Stop
- STRTSTOP is changed to Stop
- AV, LASTAV, AVCOUNTS, and ROLLOVER values are all changed to zero.
- Pre-Pre-Preset, Pre-Preset, and Preset alarms are cleared (Off).

In addition, for a cold restart, only,

- PV is changed to NaN
- PVSTS is changed to Bad
- All PV and deviation alarms are cleared (Off).

### 3.3.12 AM Counter Point Configuration

The configuration forms and Parameter Entry Displays for AM counter data points show several default values, which are entered when the point is built through the Data Entity Builder unless the default value is changed. Any parameters that don't have default values must be entered or the point can't be loaded into the AM.

The configuration forms and Parameter Entry Displays for AM counter data points offer both brief parameter sets and full-disclosure parameter sets. The brief set includes only the parameters needed to configure the PV- and AV-related functions. To configure general input connections, a contact cutout connection, or general connections, you must enter Full disclosure in parameter PTDISCL.

For more detailed information about the parameters listed in this section, refer to the *Application Module Parameter Reference Dictionary*.

The following are configurable parameters for an AM Counter data point:

NAME—The Tag Name for the Counter data point (no default value)

UNIT—The ID of the process unit that contains this point (no default value)

PTDESC—The description of this data point (default value = all blanks)

KEYWORD—The keyword for this data point (default value = all blanks)

PRIMMOD—The process module associated with this point (the default value represents a "null" tag name)

PTDISCL—Brief or full parameter-set disclosure (default = brief)

ACCUM—Selects the accumulator option (default = Off)

CISRC—Counter input connection-source point (default value represents a "null" tag name)

CIACTSTS—Counter input connection-activity status (default = Active)

PVEULO—PV engineering units low-range value (no default value)

PVEUHI—PV engineering units high-range value (no default value)

PVEXEULO—PV engineering units, extended low-range value (no default value)

PVEXEUHI—PV engineering units, extended high-range value (no default value)

PVCONV—PVAUTO conversion factor (default = 1.0)

PVTV—PV target value (default = NaN)

PVFORMAT—PV decimal point position (default = D1)

PVCLAMP—PV clamp option (default = NoClamp)

PVSRCOPT—PV source option (default = OnlyAuto)

### **3.3.12.1 AM Counter Alarm Configuration**

Each type of AM counter point alarm is enabled when the corresponding trip-point parameter contains a value other than NaN.

The PV low trip-point value is contained in PVLOTP. If PVLOTP contains a real number, a PV low-low trip point can be configured in PVLLTP.

The PV high trip-point value is contained in PVHITP. If PVHITP contains a real number, a PV high-high trip point can be configured in PVHHTP.

If PERIOD contains something other than NoPeriod and if PVLOTP and PVLLTP contain real numbers, rate-of-change values can be configured in PVROCPTP (positive trip point) and PVROCNTP (negative trip point). Likewise, if PERIOD = NoPeriod, and PVHITP and PVHHTP contain real numbers, rate-of-change values can be configured in PVROCPTP and PVROCNTP.

If the PV target value in PVTV contains a real number, deviation alarms can be configured in DEVHITP (deviation-high trip point) and DEVLOTP (deviation-low trip point).

A PV alarm-deadband value is configured in PVALDB. PVALDB contains one of the following enumerations: Half, One, Two, Three, Four, Five, or Eu (the default value is One). The values Half through Five specify the deadband in percent of the engineering-units range. The value Eu indicates that the value in PVALDBEU specifies the deadband value in engineering units.

If ACCUM = On, indicating that accumulation is to function, an accumulation-target value can be entered by an operator in AVTV. When AVTV contains a value other than NaN, the trip point configured in AVDEV1TP is monitored. If AVDEV1 contains a value other than NaN, a second deviation trip point can be configured in AVDEV2TP.

Each of the alarm conditions has its own alarm priority parameter. These parameters are:

BADCTLPR	Bad control alarm priority
BADPVPR	Bad PV alarm priority
CNFERRPR	Configuration error alarm priority
DEVHIPR	Deviation high alarm priority
DEVLOPR	Deviation low alarm parameter
PPPRSTPR	Pre-pre-preset alarm priority (see subsection 3.3.7)
PRESETPR	Preset alarm priority (see subsection 3.3.7)
PRPRSTPR	Pre-preset alarm priority (see subsection 3.3.7)
PVHHPR	PV high high alarm priority
PVHIPR	PV high alarm priority
PVLLPR	PV low low alarm prioity
PVLOPR	PV low alarm priority
PVROCNPR	PV negative rate of change alarm priority
PVROCPR	PV positive rate of change alarm priority
PVSGCHPR	Significant PV change alarm priority



### 3.3.12.2 AM Counter Accumulator Control and Operating Parameters

If ACCUMN = On, the following parameters can be configured:

AVCONV—the AV conversion factor (default = 1.0)

AVFORMAT—the AV decimal-point position (default = D0)

AVTV—the AV target value (default = NaN)

CNTLLOCK—the accumulator state-control lockout (default = Permit)

AVTVLOCK—the accumulator target-value lockout (default = Permit)

### 3.3.12.3 General Input, General Output, and Contact Cutout Configuration

Refer to the *Application Module Counter Configuration Forms, AM88-440*, and to the *Application Module Parameter Reference Dictionary*, to configure these connections. PTDISCL must contain Full.

### 3.3.12.4 AM Counter Configuring the Point-Processing Schedule

Refer to 2.2 in this publication and to the *Application Module Parameter Reference Dictionary*, to configure the processing schedule for the AM Counter data point.

## 3.3.13 AM Counter Error and Warning Messages

One of the following warnings may be returned when an attempt is made to change an AM Counter data-point parameter:

- **Value Clamped Error**—the new value exceeded one of the PV or AV limits and was clamped at that limit.
- **String Truncated**—the new value was a string with too many characters, so only the maximum number of characters was accepted. Characters at the right end of the string were rejected.

One of the following error messages may be returned when an attempt is made to change an AM Counter data-point parameter:

- **Parameter Invalid**—the parameter references is not configured on this data point.
- **Parameter Cannot Be Changed**—the value in the referenced parameter can't be changed.
- **Illegal Value**—the new value is the wrong type for the parameter.
- **PV Source Disallows PV Change**—PVSOURCE currently has a value that does not permit the requested PV value change.
- **Limit or Range Crossover**—a new value for a limit or a range parameter crosses over the opposite limit or range value.

- **Source of Request Invalid**—the requester had an improper access key for this parameter.
- **Mode Not Man or Point Active**—the point must be Inactive to allow a change of this parameter's value.
- **Engr Unit Span Too Small**—defined engineering-units span must be nonzero.
- **Limit or Range Exceeded**—the new value exceeded the range or a limit of this parameter.
- **Read Only Parameter**—this parameter can be read, only—its value cannot be changed.
- **Change Not Permitted By Operator**—a Universal Station operator cannot change the parameter.
- **Illegal Counter State**—the current state of this Counter point does not allow this value change.
- **Point Type Invalid**—the configured source point for the counter input connection is not an HG Control Counter.

### 3.4 AM FLAG DATA POINT

A Flag data point in an Application Module contains a PV with two discrete states. Flag data points can be used for any task that requires the retention of a discrete state indicator, such as synchronizing sequence programs that run concurrently or store recipe data.

The state of the PV is indicated on displays and printed logs and reports as one of the values in parameter PVSTATES. PVSTATES is an array of two values. The two values in PVSTATES are 8-character strings that can be configured and can be changed by someone with an engineer's key at a Universal Station; therefore, the PV value is said to be a "self-defining" enumeration. For example,

PVSTATES(0) = Full

and

PVSTATES(1) = Empty

The PV state can be changed by an operator at a Universal Station or by a user-written program. When the PV is one state its value is Full and when in the other state its value is Empty.

The character strings are configured in the two PVSTATES values. The configured values can be changed by someone with an engineer's key at a Universal Station, or by a user-written program. STATE1 contains the same character string as PVSTATES(0) and STATE2 contains the same character string as PVSTATES(1).

#### NOTE

Refer to the *Application Module Parameter Reference Dictionary*, for further information on the parameters mentioned in this section.

### 3.4.1 Scheduling of AM Flag Points

AM Flag data points are not scheduled and are not processed at regular intervals. These points are active only when a new value is stored in one or more of its parameters by someone at a Universal Station or by a user-written program.

### 3.4.2 AM Flag Point Display- and History-Related Parameters

The following parameters contain information related to the representation of the AM Flag data point on Universal Station displays and in the process-history files:

- **UBOXCLR**—Defines the color of the upper box on Group and Detail displays that show this data point. The upper box is filled with that color when the PV is in the state represented by the box.
- **LBOXCLR**—Defines the color of the lower box on Group and Detail displays that show this data point. The lower box is filled with that color when the PV is in the state represented by the box.
- **DISPTYPE**—Defines the format to be used on the Group and Detail displays for this data point type (Flag).
- **PTDESC**—The Point Descriptor, a configured string of up to 24 characters.
- **KEYWORD**—This point's keyword, configured as a string of up to eight characters.
- **OVERVAL**—If this parameter contains On, off-normal alarms for this point are shown on the Overview display.
- **HIGHAL**—Indicates the highest alarm value for this point, which is either NoAlarm or OffNormal.
- **HIGHALPR**—Indicates the priority of the alarm value that is in HIGHAL.

### 3.4.3 Off-Normal Alarms in AM Flag Points

If parameter OFFNORMAL contains On, off-normal alarms are enabled, so PVNORMAL should contain the PVSTATES character string that represents the normal state of this Flag data point. Parameter OFFNRMFL indicates whether the alarm exists or not (On = alarm, Off = no alarm). The following chart shows an example of the alarm conditions, assuming PVSTATES(0) contains Full and PVSTATES(1) contains Empty:

OFFNRMAL	PVNORMAL	PV	OFFNRMFL
Off	Full or Empty	Full	Off
Off	Empty or Full	Empty	Off
On	Full	Full	Off
On	Empty	Full	On
On	Full	Empty	On
On	Empty	Empty	Off

The off-normal alarm check is made when one of the following parameters is changed from a Universal Station or by a user-written program:

PV  
PVNORMAL  
OFFNRMAL  
ALENBST

And when OFFNRMPR is changed from NoAction.

If OFFNRMAL contains On, OFFNRMPR contains the alarm priority, which can be configured and can be changed by someone with an engineer's key at a Universal Station. The possible priorities are Emergency, High, Low, Journal, NoAction, Printer, and Jnlprint (see 4.3.1.3 in *System Control Functions*).

An operator at a Universal Station can enable, disable, or inhibit the alarm by changing the value in ALENBST.

A user-written program can cut out this alarm by storing On in CONTCUT. Contact cutout connections are not applicable.

### 3.4.4 AM Flag Point Restarts

The database for an AM Flag data point is restored from its checkpoint file in a History Module during cold, fast, and hot restarts.

### 3.4.5 AM Flag Point Configuration

The configuration forms and Parameter Entry displays for AM Flag data points show several default values, which are entered when the point is built through the Data Entity Builder unless the default value is changed. Any parameters that don't have default values must be entered or the point can't be loaded into the AM.

The following are the configurable parameters for an AM Flag data point (see the *Application Module Parameter Reference Dictionary* for further parameter details):

NAME—The Tag Name for the Numeric data point (no default value)

UNIT—The ID of the process unit that contains this point (no default value)

PTDESC—The description of this data point (default value = all blanks)

KEYWORD—The keyword for this data point (default value = all blanks)

PRIMMOD—The process module associated with this point (the default value represents a "null" tag name)

PVSTATES—Contains the two strings, of up to eight characters, that define the PV states and appear on Universal Station displays, in process history files, and on printed logs and reports (default for both values is all blanks)

UBOXCLR—The upper-box color on Group and Detail displays (default = red)

LBOXCLR—The lower-box color on Group and Detail displays (default = red)

OFFNRMAL—Off-normal alarm enable (default = Off)

If OFFNRMAL is configured as On, the following parameters can be configured:

PVNORMAL—The normal state of the PV (default is all blanks)

ALENBST—The initial value for the alarm-enable status (default = Enable)

OFFNRMPR—Off-normal alarm priority (default = Low)

OVERVAL—Overview-display indication of off-normal alarm (default = Off)

### **3.4.6 AM Flag Point Error Messages**

The following error indications are returned when an improper attempt is made to change a parameter value:

- Read Only Parameter—this parameter is a read only parameter.
- Illegal Value—the new value is illegal for this parameter.

## **3.5 AM TIMER DATA POINTS**

Timer data points in an Application Module are used to measure time intervals and to initiate events when the prescribed time has elapsed.

Figure 3-6 shows the functional structure of an AM Timer data point.

#### NOTE

Refer to the *Application Module Parameter Reference Dictionary* for further information on the parameters mentioned in this section.

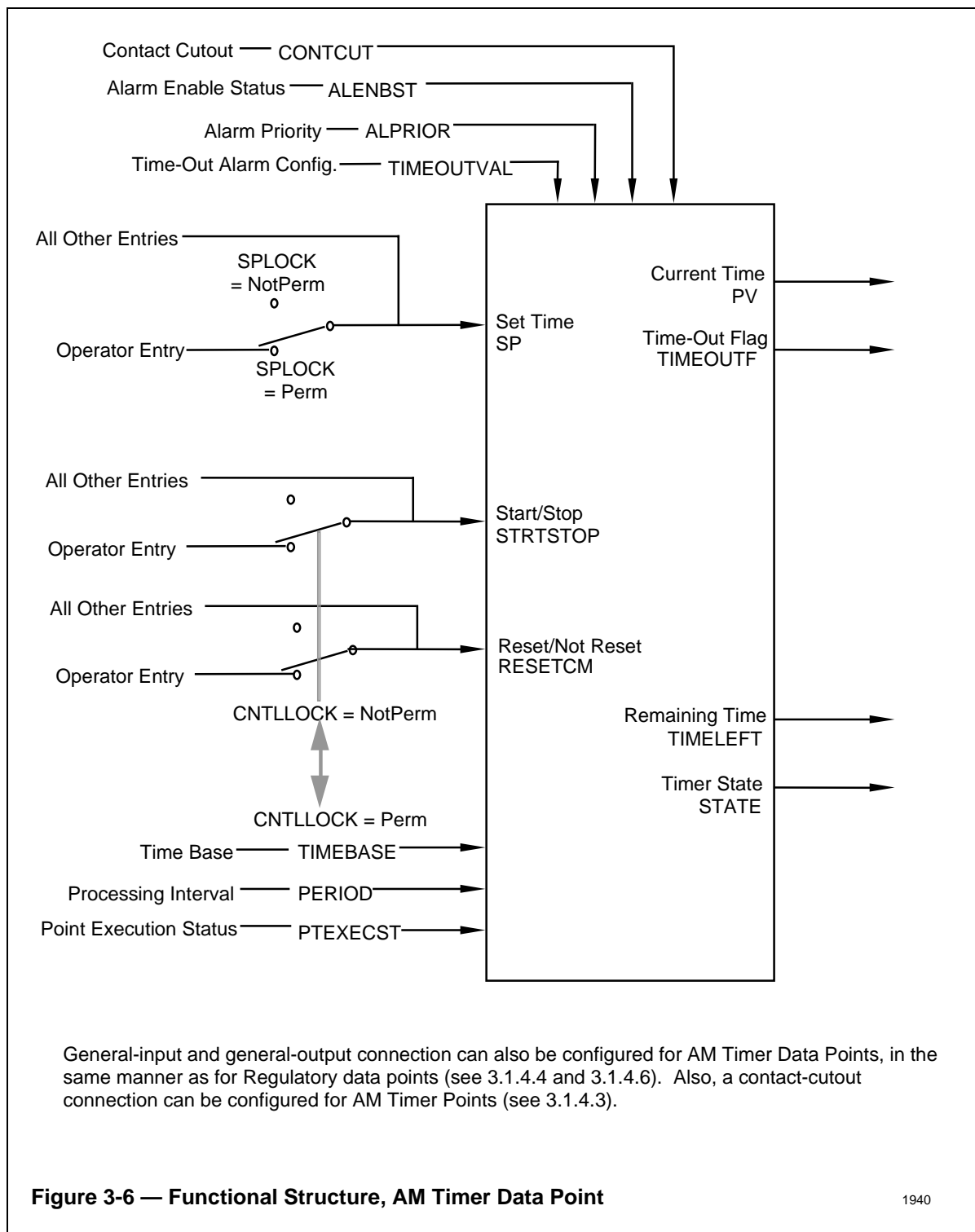
### 3.5.1 AM Timer Point Functional Description

When parameter STATE contains Running, the timer is running and

$$PV = PV + \text{Period}$$

where

PV is the current time and Period is the processing interval configured in PERIOD, expressed in the units specified by TIMEBASE.



For example, if PERIOD contains 30 seconds and TIMEBASE contains Minutes, Period in the equation above is 0.5 minutes.

Normally, an operator at a Universal Station can start and stop the timer and can reset the time by changing values in the STRTSTOP and RESET parameters; however, if CNTLLOCK contains NotPerm, operators can't change these parameters. CNTLLOCK can be changed from Perm to NotPerm and vice versa at a Universal Station by someone with an engineer's key.

Similarly, an operator can normally change the set-time value in SP, but if SPLOCK contains NotPerm, an operator can't make such changes. SPLOCK can be changed from Perm to NotPerm and vice versa at a Universal Station by someone with an engineer's key.

If the set-time value in SP is not NaN (not a number) and PV is equal to or greater than the value in SP, the timer state in STATE is Stopped and STRTSTOP contains Stop. Also, if the time-out alarm is enabled (TIMOUTAL contains On), the time-out flag, TIMOUTFL, contains On which generates an alarm.

If SP contains NaN, the PV is updated at each processing pass until it contains 1,000,000 time-base units, then it rolls over to zero and resumes updating.

Parameter TIMELEFT contains the difference between PV and SP ( $PV - SP$ ). If SP contains NaN, TIMELEFT also contains NaN.

If STATE contains Stopped, the AM Timer point is not processed.

#### **3.5.1.1 Controlling the Timer**

The AM Timer is controlled through the parameters listed in Table 3-3. In most cases, the values of these parameters can be changed by someone at a Universal Station or by a user-written program. The table defines the effects of the parameter changes and all restrictions to those changes.

#### **3.5.2 AM Timer Point Time-Out Alarm**

If TIMOUTAL contains On, the time-out alarm is enabled and the following alarm-related parameters apply to the AM Timer data point:

TIMOUTFL—The time-out alarm flag

TIMOUTPR—Which specifies the alarm priority as No/action, Journal, Low, High, Emergency, Printer, or Jnlprint

ALENBST—The alarm-enable status that can be Enable, Disable, or Inhibit.

It is also possible to configure contact cutout for this alarm. See 4.3.1.7 in *System Control Functions*.

If the set-time value in SP is NaN, the time-out alarm cannot occur.



**Table 3-3 — Timer Control Table**

Parameters	Values	Restrictions and Results
<b>STRTSTOP</b>  Normal Access Level: Operator	Start	Restrictions: No Start when SP is not NaN and PV is equal to or greater than SP Operator access not permitted when CNTLLOCK = NotPerm  Resultant parameter values: STRTSTOP = Start STATE = Running, if PTEXECST = Active STATE = Stopped, if PTEXECST = InActive
	Stop	Restrictions: Operator access not permitted when CNTLLOCK = NotPerm  Resultant parameter values: STATE = Stopped STRTSTOP = Stop
<b>RESETCMD</b>  Normal Access Level: Operator	Reset	Restrictions: Operator access not permitted when CNTLLOCK = NotPerm  Resultant parameter values: PV = 0.0 RESETCMD = Reset (momentarily), then NotReset TIMOUTFL = Off
	NotReset	No restrictions
<b>PTEXECST</b>  Normal Access Level: Superv.	Active	No restrictions  Resultant parameter values: PTEXECST = Active STATE = Running, if STRTSTOP = Start STATE = Stopped, if STRTSTOP = Stop TIMOUTFL = Off (any existing alarm is cleared) RESTART = PtActvn
	InActive	No restrictions  Resultant parameter values: PTEXECST = InActive STATE = Stopped STRTSTOP = Stop PV = 0.0 TIMOUTFL = Off

(Continued)

**Table 3-3 — Timer Control Table (Continued)**

Parameters	Values	Restrictions and Results
PV  Normal Access Level: Oper.	A new current time	<p>Restrictions:</p> <p>No change in PV when:</p> <p>STATE = Running</p> <p>The new value is NaN</p> <p>The new value is greater than SP</p> <p>SP <math>\neq</math> NaN</p> <p>If the new time is <math>&lt; 0</math> or if the new time <math>&gt; 1,000,000</math>, new values from a Universal Station are rejected and new values from another data point or from a user-written program are clamped to the limit (0 or 999,999).</p> <p>Resultant parameter values:</p> <p>PV = the new current time in TIMEBASE units, or the clamped value.</p>
SP  Normal Access Level: Superv.	A new set time	<p>Restrictions:</p> <p>No change by an operator when SPLOCK = NotPerm</p> <p>If the new time is <math>&lt; 0</math>, or if the new time <math>&gt; 1,000,000</math>, new values from a Universal Station are rejected and new values from another data point or from a user-written program are clamped to the limit (0 or 999,999).</p> <p>Resultant parameter values:</p> <p>SP = the new current time in TIMEBASE units, or the clamped value.</p>

### 3.5.3 AM Timer Point Scheduling

AM Timer data points are scheduled as described under 2.2.1 in this publication.

If PTEXECST contains Inactive, the AM Timer point is not processed. If PTEXECST contains Active, the point is processed in the following order at each processing interval:

- **General input processing**, if any general input connections are configured (see 3.1.4.4 in this publication).
- **Timer Processing** as described under 3.4.1 in this publication.

- **General output processing**, if any general output connections are configured (see 3.1.4.6 in this publication).

**NOTE: Do not connect the General Output of a Timer point to the PPS of another point. See the warning at the end of subsection 3.1.4.6 for additional details.**

### 3.5.4 AM Timer Point Restarts

When an AM Timer data point is restarted by using its data from a checkpoint file in an HM, its data is initialized as follows:

- STATE is set to Stopped
- STRTSTOP is set to Stop
- The PV value is set to 0.0
- TIMOUTFL is set off and return-to-normal is indicated

### 3.5.5 AM Timer Point Configuration

The configuration forms and Parameter Entry displays for AM Timer data points show several default values, which are entered when the point is built through the Data Entity Builder unless the default value is changed. Any parameters that don't have default values must be entered or the point can't be loaded into the AM.

The configuration forms and Parameter Entry displays for AM Timer data points offer both brief parameter sets and full-disclosure parameter sets. The brief set includes only the parameters needed to configure the basic timer functions. To configure general input connections, a contact cutout connection, or general output connections, you must enter Full disclosure in parameter PTDISCL.

The following are the configurable parameters for an AM Timer data point (see the *Application Module Parameter Reference Dictionary* for further parameter details):

NAME—The Tag Name for the Numeric data point (no default value)

UNIT—The ID of the process unit that contains this point (no default value)

PTDESC—The description of this data point (default value = all blanks)

KEYWORD—The keyword for this data point (default value = all blanks)

PRIMMOD—The process module associated with this point (the default value represents a "null" tag name)

PTDISCL—Brief or full parameter-set disclosure (default = brief)

TIMEBASE—Timebase units (default = Minutes)

SP—Set Time (default = NaN)

CNTLLOCK—Operator timer-control lock (default = Permit)

SPLOCK—Operator set-time lock (default = Permit)

TIMOUTAL—Time-out alarm enable (default = Off)

ALENBST—Time-out alarm-enable status (default = Enable)

BADCTLPR—Bad control value alarm priority parameter

CNFERRPR—Configuration error alarm priority (default = Low)

TIMOUTPR—Time-out alarm priority (default = Low)

PERIOD—Processing interval (no default value. See 2.2 in this publication)

BEFAFT—Processing before or after relationship (default = No)

If BEFAFT is configured with a value of Cycle, NORMCYCL must be configured with an appropriate cycle number.

Refer to the *Application Module Forms*, and to the *Application Module Parameter Reference Dictionary*, to configure general input, contact cutout, and general output connections.

### 3.5.6 AM Timer Point Error and Warning Messages

One of the following error messages may be returned when an attempt is made to change an AM Timer data point parameter:

- **Change Not Permitted By Operator**—A Universal Station operator cannot change the parameter.
- **Illegal Value**—The new value is the wrong type for the parameter.
- **Parameter Cannot Be Changed**—The value in the parameter can't be changed.
- **Illegal Timer State**—The value can't be changed because of the current state of the Timer.
- **Read Only Parameter**—Only Read access is allowed for the parameter.
- **Parameter Invalid**—The parameter referenced is not configured on this data point.

The following warning may be returned when an attempt is made to change an AM Timer data-point parameter:

- **Value Clamped Error**—The new value (for SP or PV) was greater than 999,999 or less than zero; therefore, the value was clamped at the appropriate one of those limits.

### **3.6 AM CUSTOM DATA POINTS**

Often, there is a need for custom CL logic that does not have a single, specific regulatory data point associated with it. That is, CL logic that conceptually stands by itself, perhaps monitoring or controlling the activities of several regulatory (or other) data points. What is needed for this is a data point that has the standard parameters relating to identification, scheduling, custom data-segment attachment, and CL block status, a data point that has no other data except custom data segments, and one that has no functions other than those added by custom CL blocks. Custom Data Points are provided to meet this need.

The chart below shows the standard user-visible parameters in a Custom Data Point.

Significance —	CONFIGURATION	OPERATION	STATUS/INFO
Identification —	NAME (B)		DISTYPE (V)
	PTDESC (EB)		
	KEYWORD (EB)		
	UNIT (B)		
	PRIMMOD (B)		
Schedule —	PERIOD (B)	PPS (O)	PPSTYPE (V)
	BEFAFT (B)		
	BEFAFTID (B)		
	NORMCYCL (B)		
General Operation —		PTEXECST (S)	RESTART (V)
Alarms —	BCLEALPR (EB)	ALENBST (O)	PTINAL (V)
	BCLFALPR (EB)	CONTCUT (P)	HIGHAL (V)
	CLEALMPR (EB)		HIGHALPR (V)
	CLFALMPR (EB)		CLEALMFL (V)
	CNFERRPR (EB)		CLFALMFL (V)
			\$BCLEAFL (V)
			\$BCLFAFL (V)
CDS —	NOPKG (B)		NOCDS (V)
	PKGNAME (B)		
	(1..NOPKG)		
	(.....Others.....Per.....CL.....)		
CL Blocks —	CLSLOTS (B)	CLACTIVE (EP)	BLKTIME (L)
	BLKNAME (L)	(1..CLSLOTS)	(1..CLSLOTS)
	(1..CLSLOTS)	\$CLCMPST (EP)	CLERRSUM (V)
	INSPPOINT (L)	(1..CLSLOTS)	CLBLKERR (V)
	(1..CLSLOTS)	\$BKGSTS (V)	(1..CLSLOTS)
	INSORDER (L)	(1..CLSLOTS)	CLERRLOC (V)
	(1..CLSLOTS)		(1...CLSLOTS)
			BRANCHES (V)
			(1...CLSLOTS)
			CLVERSIN (V)
			(1...CLSLOTS)
			CLREVISN (V)
			(1...CLSLOTS)
			CLUSECNT (V)
			(1...CLSLOTS)
			\$BKGABRT (V)
			\$BKGQFUL (V)
			\$BKGTIME (V)
			\$BKGPRTY (V)
			(1...CLSLOTS)

Parameter-access locks:

- (V)—View only
- (B)—DEB only
- (EB)—DEB or Engineer key only
- (P)—Program (or continuous control) or DEB
- (E)—Engineer, DEB or Program
- (S)—Supervisor, Engineer, DEB, or Program
- (O)—Operator Supervisor, Engineer, DEB, or Program
- (L)—Entered by CL Linker
- (EP)—Engineer or Program

### 3.6.1 Custom Data Point Identification Data

The identification data shown in the chart on the previous page is a subset of that used on all regulatory data points.

### 3.6.2 Custom Data Point Schedule and General Operation

Custom Data Points are scheduled just as Regulatory data points are, including period, before/after, and cycle options. This means that a Custom Data Point can be processed on a schedule, or it can be given no schedule at all and processed entirely with process specials. The PPSREQ parameter can be used by CL code to determine whether a PPS is used to process a given point execution.

The point-execution status, PTEXECST, has the states "active" and "inactive," which apply as in Regulatory data points. That is, the point doesn't execute in the inactive state. As with Regulatory points, the RESTART parameter provides initialization and startup information. Specifically, it indicates the first execution since point activation, if this is the first execution since a cold, warm, or hot restart.

### 3.6.3 Custom Data Point Alarms

The alarm-related parameters on the chart, above, are also the same as the corresponding functions of Regulatory data points. These functions include the following;

- Alarm priority
- Alarm enable
- Contact cutout, where the cutout flag is driven exclusively by CL logic
- Point in alarm (PTINAL) and the highest-level current alarm (HIGHAL) flags
- CL error and failure alarms

### 3.6.4 Custom Data Points and CL Segment Data

The number of packages and each package file name are specified as for other AM points. The CL segment data is defined under 4.1.5 in this publication.

### 3.6.5 CL Block Insertion Points in Custom Data Points

There are two insertion points on a Custom Data Point. These are the "general" insertion point (which is processed first) and the "backgrnd" insertion point. Multiple CL Blocks can be linked to either insertion point. The order of execution of blocks on the same insertion point can be established by information located in the Block Headers of the CL programs. See the *Control Language/Application Module Reference Manual* for more detail.

### 3.7 AM SWITCH DATA POINTS

A Switch data point is a Custom Data Point in an AM, with two additional functions: a CL Switch and some custom alarming features.

The following chart defines the user-visible parameters of a Switch data point.

	CONFIGURATION	OPERATION	STATUS/INFO
IDENTIFICATION	NAME (B) PTDESC (EB) KEYWORD (EB) UNIT (B) PRIMMOD (B)		DISTYPE (V)
SCHEDULE	PERIOD (B) BEFAFT (B) BEFAFTID (B) NORMCYCL (B)	PPS (O) PPSREQ (P)	PPSTYPE (V)
GENERAL OPERATION	NUMSWTCH (B) S1NSTATE (B) S1STATES (B) (0..x)* S1ACCLVL (EB) (0..x)*  S2NSTATE (B) S2STATES (B) (0..x)* S2ACCLVL (EB) (0..x)*	PTEXECST (S)  S1REQSTS (O) S1CURSTS (P)  S2REQSTS (O) S2CURSTS (P)	RESTART (V)
*x = 1 less than SnNSTATE			
ALARMS	BCLEALPR (EB) BCLFALPR (EB) CLEALMPR (EB) CLFALMPR (EB) S1ACCLVL (EB) CNFERRPR (EB) SWALM1PR (EB) SWALM2PR (EB) SWALM3PR (EB)  SALMDSC1 (E) SALMDSC2 (E) SALMDSC3 (E)	ALENBST (O) CONTCUT (P)        SALMFL1 (P) SALMFL2 (P) SALMFL3 (P)	PTINAL (V) HIGHAL (V) CLEALMFL (V) CLFALMFL (V)  \$BCLEAFL (V) \$BCLFAFL (V) HIGHALPR (V)  SALMTR1 (V) SALMTR2 (V) SALMTR3 (V)

#### Parameter-Access Locks:

- (V)—View only
- (B)—DEB only
- (EB)—DEB or Engineer key only
- (P)—Program, connection from a continuous-control data point or DEB.
- (O)—Operator, Supervisor, Engineer, DEB, or program
- (E)—Engineer, DEB, or program

(continued)



## Switch Data-Point Parameter Chart (Continued)

	CONFIGURATION	OPERATION	STATUS/INFO
CDS		NOPKG (B)	NOCDS (V)
	PKGNAME (B) (1..NOPKG)		
	(.....Others.....Per.....CL.....)		
CL BLOCKS	CLSLOTS (B) BLKNAME (L) (1..CLSLOTS) INSPOINT (L) (1..CLSLOTS) INSORDER (L) (1..CLSLOTS)	CLACTIVE (EP) (1..CLSLOTS) \$CLCMPST (EP) (1..CLSLOTS) \$BKGSTS (V) (1..CLSLOTS)	BLKTIME (E) (1..CLSLOTS) CLERRSUM (V) CLBLKERR (V) (1..CLSLOTS) CLERRLOC (V) (1..CLSLOTS) BRANCHES (V) (1..CLSLOTS) CLVERSIN (V) (1..CLSLOTS) CLREVISN (V) (1..CLSLOTS) CLUSECNT (V) (1..CLSLOTS) \$BKGABRT (V) \$BKGQFUL (V) \$BKGTIME (V)

### Parameter-Access Locks:

- (V)—View only
- (L)—Entered by CL Linker
- (B)—Data Entity Builder, only
- (E)—Engineer, Program, or DEB
- (EP)—Engineer or Program

The principal reasons to use a Switch data point are:

- Switch 1 is displayable on the point's Group Display at a Universal Station, therefore, an operator can use it to monitor and to control the CL logic. Switch 2 and switch 1 are displayed and can be controlled through a page of the point's Detail Display.
- The alarming features allow custom alarms to be generated by CL logic and to be displayed on the Process Alarm Summary display.

### 3.7.1 General Parameters

Except for the switch parameters and the added alarm information, the identification, schedule, CDS, and CL-Block data are as they would be for a Regulatory Data Point. The related functions, including scheduling, special point processing (PPS), and CL execution are also the same.

### 3.7.2 Switch Parameters

CL Switch parameters are as described for CL Switches on AM Regulatory Points, under 4.2. Here, the default value for NUMSWTCH is 1 rather than 0, and the legitimate options are 1 and 2, rather than 0, 1, and 2; otherwise, functions of the parameters and expected use by CL and operators are the same.

### 3.7.3 Switch Alarming Functions

In addition to the standard alarming functions that are supplied on Custom Data Points, a Switch Data Point includes nine standard parameters that can be used by CL logic to create three independent, custom, process-alarm conditions.

These parameters have this form:

SALMFLn    SALMDSCn    SALMTRn

where n ranges from 1 to 3 for the three alarm conditions. Each alarm flag parameter (SALMFLn) has a logical value—Off or On. A CL Block bound to the point can set an alarm condition by setting one of these alarm flags to On. The standard point processing logic generates an "off-normal" process alarm, using the SALMDSCn parameter as the alarm-description text. This alarm is displayed on the Alarm Summary Display at the appropriate Universal Stations, along with other process alarms. When the CL logic has determined that the alarm condition has gone away, it can set the alarm flag in SALMFLn to Off. This causes the standard point processing logic to generate a return-to-normal for that process alarm.

Alarm-description parameters contain strings of up to eight characters, and allow Engineer access. It is expected that they will be entered through point building through the DEB, but they can be changed at a Universal Station by someone with an Engineer's key.

The alarm transition flags (SALMTRn) are provided mainly for symmetry—all other alarm flags in the AM have transition flags for use in entering CL logic only when alarm transitions occur. The flag states are Alarm, Rtn, and NoChange.

These alarms have an internal level, which is SALMFL3, SALMFL2, and SALMFL1 (decreasing order), behind the higher level CL Error (CLEALMFL) and CL Failure (CLFALMFL). When alarms have been assigned the same priorities in their individual alarm priority parameters, the internal level is used as a tie-breaker to determine which alarm to show on the Detail and Group Displays (for more information, see *System Control Functions*, subsection 4.3.1.4).

### 3.7.4 CL Block Insertion Points in Switch Data Points

There are two insertion points on a Switch Data Point. These are the "general" insertion point (which is processed first) and the "backgrnd" insertion point. Multiple CL Blocks can be linked to either insertion point. The order of execution of blocks on the same insertion point can be established by information located in the Block Headers of the CL programs. See the *Control Language/Application Module Reference Manual* for more detail.



## CONTROL-LANGUAGE SUPPORT

### Section 4

#### 4.1 CL EXECUTION IN THE AM

##### 4.1.1 CL's Place in Point Processing

The Control Language provides a means for a process engineer to customize control strategies.

Where standard point processing is used without CL programs, continuous-control data points are processed according to a standard, fixed scheme, as configured by the process engineer. Because it would be impossible to implement every conceivable control scheme and algorithm as a standard option, three methods to vary the standard point-processing scheme are provided. They are:

- **When normal point processing is satisfactory**, except that some **nonstandard actions are required**, the customer can insert CL Blocks in the standard point-processing sequence in an AM, at one or more predefined insertion points.
- **When normal point processing is satisfactory**, except that a **nonstandard PV algorithm** or a **nonstandard control algorithm** is required, the customer can choose the CL PV Algorithm or the CL Control Algorithm. If so, a user-written CL Block executes in the PV-processing interval or in the control-processing interval in the standard point-processing sequence.
- **When normal point processing is satisfactory** and some **nonstandard actions are required** and **the inserting of logic into normal point processing would be cumbersome**, Background CL Blocks can be added to be initiated during the point processing cycle.

#### NOTE

Though the descriptions in this section on "Access Rights Checks" and on "Range and Limit Violations" apply to CL executing in Multifunction Controllers and in AMs, this section primarily covers the execution of CL in an AM.

##### 4.1.1.1 Foreground and Background CL Blocks

CL blocks are user-written routines (see the *Control Language Reference Manual* ) that are "inserted" into the point processing structure at defined places. Figure 3-2 shows the places where a CL block can be inserted into the processing of a regulatory data point.

CL blocks inserted at any place except the “BACKGRND” insertion point are referred to as "Foreground CL blocks." The Foreground CL blocks run as subroutines, each block completing before returning to process the remaining control functions on that point. This is a powerful and flexible structure, but it does require that the Foreground CL blocks complete quickly so that the normal point processing of other points is not delayed.

A final insertion point called BACKGRND is added at the end of point processing for the regulatory, custom, and switch point types. Blocks linked to this insertion point are called "Background CL blocks" and are given special treatment. Instead of running as subroutines that must complete before the processing of the next point can begin, Background CL blocks are queued to run under a separate "background" task.

Background CL blocks run in AM "free" time, after the "foreground" operations such as point processing (including foreground CL blocks) and alarm distribution for all points have completed. Therefore, background CL blocks can run for longer periods of time (over many processing cycles) without disrupting the timing of point processing. This leads to two important differences between Foreground and Background CL blocks: there is no backward branch count limit for Background CL blocks, and Background CL blocks can be aborted (or made inactive) by the operator.

The number of background blocks that a point can have is limited only by the configured number of insertion blocks per point (up to 255). Each time that a point is processed, it is checked for active Background CL blocks. If it has active background blocks and none are running, the point is queued for background processing. Once queued, a point's active Background CL blocks run to completion one at a time in order. After all have completed, the point can again be queued for background processing.

The Background CL blocks from up to 10 points can run at a time (per AM), sharing 50 millisecond slices of the available free time between processing of points. (The number of simultaneously executing background tasks is customer configurable and consumes user point and CL program data space.

Operational differences of Background CL blocks are covered in detail at heading 4.1.6, Background CL Operation. Many of the execution characteristics, such as data access checks and CL error conditions, are common to both Foreground and Background CL blocks, and are discussed together in the paragraphs that follow.

#### **4.1.2 CL's Connection to Data Points**

These are the steps required to connect CL/AM Blocks to data points.

- a) Source code is entered through the Engineering Personality (EP) Command Processor EDit function. Source code file names use the '.CL' extension, and can contain:
  - CL 'Block' programming statements and local data definitions.
  - Custom Data Segments (CDS) containing user-defined parameters.
  - Custom enumerations and parameter lists.

- b) When the source is compiled through the EP Command Processor's CL overlay, object files and custom data segment files are produced.
  - Object files (<filename>.AO) contain the program to execute in the AM. This is linked to an AM Custom, Regulatory, or Switch Point through the EP Command Processor's CL Link function. The point that the program is loaded to is referred to as the Bound Data Point or BDP.
  - Custom data segments (<filename>.GD) are attached to AM points through the EP Application Module Point Building function.
- c) The attaching of CL code to an AM point requires, in order:
  - 1) Any Custom parameters used must be compiled.
  - 2) The AM point is built with the quantity of CL objects (blocks) that can be linked to it specified (CLSLOTS) and, if Custom Data Segments are used, the quantity and names of the CDS source files are entered (PKGNAME(n)). Optionally, the values for the Custom parameters can be specified at point build time. The Custom parameters must be compiled before the package name can be entered, and the point must be built with CL slots before the CL object can be linked.
  - 3) The source code is compiled and the object is linked. The Bound Data Point must be inactive while linking.

There are two basic ways in which CL Blocks can be attached to data points: specific, and generic—as specified in the Header of the CL Block:

1. When a specific data-point name is specified in the CL header, the CL Block can be connected to only that point.
2. When the GENERIC clause is used in the CL header, the actual point to which the block is linked can be any point that is consistent with the applicable Parameter Lists and PARAMETER statements in the CL code.

### 4.1.3 Indirect Reference through Point Identifiers

CL/AM programs can read or write values of point.parameters through both direct and indirect reference formats.

**Direct Reference**—Both standard and custom parameters of the Bound Data Point can be directly referenced by the parameter name (without the requirement of the preceding point name). The parameters of other points in the AM and points that reside in other LCN nodes can be directly referenced from a CL/AM program by declaring the points as EXTERNAL and preceding the parameter references with the external point identifiers (names). These direct reference points must exist when the CL/AM program is compiled and linked.

**Indirect Reference**—You can use the indirect reference capability to write 'generic' CL/AM code that references external points through a Custom Data Segment (CDS) parameter whose type is point identifier (the names of the external point identifiers are not declared in the CL/AM program). This permits the program to be developed and compiled without knowing the external identifiers of points used in the program. Also, the point identifier can be changed without having to re-edit the source. The identifiers of the indirectly addressed external points are specified when the AM point is loaded (through the values in Custom Data Segment parameters), and bound to this CL/AM program when it is linked to the point.

A dynamic indirection capability allows the CDS parameter value that contains the external point identifier to be changed without having to inactivate, reload, or relink the point. The CDS parameters containing point identifiers can be modified through the DEB, Detail Display–Custom Page, Custom Schematics, through CG program stores, and built-in CL Functions and Subroutines.

CL indirection supports:

- a) references to arrays of point ids
- b) multiple levels of point id indirection
- c) no point identifier (Null Point ID) being specified in an array as the CDS parameter id value
- d) a store of the same point id value to an on-point CDS parameter relinks all CL references to this parameter when they start on this Bound Data Point.

The capabilities and restrictions of indirection are explained further in the *Control Language/Application Module Reference Manual* under headings 2.3.8 (Indirect Reference to Point Identifiers) and 2.3.4 (Data Points Data Type).

#### **4.1.4 Data Access and CL Runtime Errors**

Data access from both Foreground and Background CL Blocks is subject to several system-enforced restrictions. These include access rights checks, range and limit checks, and data-transport errors.

##### **4.1.4.1 Foreground CL Block Data Access**

Foreground CL references to data-point parameters proceed as follows:

- Each reference to a point in another node uses a prefetch/poststore mechanism that fetches all parameters that might be needed for a processing cycle, before the processing cycle.
- References to data points in the same node, including those to the bound data point, are made, in place, as the CL Block is executed.

The result is that a "fetch" from a parameter that received a new value during the current cycle returns a value whose "freshness" varies in different situations, as follows:



Situation	Value Returned on Fetch
Destination parameter is in the same node.	Store is done in place including Data Access checks. The Value that results from the store is returned on the subsequent fetch.
Destination parameter is another node and the fetch is from the same point from which the store was made.	The stored value is internally held (in the point). This value is returned on the fetch. Data Access checks of the stored value have not yet been made. It is assumed (but not assured) that the value will pass the checks.
Destination parameter is in another node and the fetch is from a point other than the one that caused the store.	The original prefetched value is returned. One point has no implicit knowledge of another point's off-node stores.
Destination parameter point identifier (indirection) is changed during the point execution cycle (between prefetch and end-of-point execution).	The change in point identifier does not take effect until the end of the point execution cycle (i.e., prefetch and poststores are done based on the old point identifier).

#### 4.1.4.2 Background CL Block Data Access

All Background CL accesses to data points, whether in the same node or in another node on the LCN, are made in place as the CL block is executed. There is no prefetch or poststore of data for Background CL blocks.

This results in a separate LCN data fetch or store for each Background CL reference to an off-node data point. Thus a large number of LCN data fetches will significantly lengthen the execution time of a Background CL block. You should consider alternatives such as having a process-special point fetch the required data in Foreground CL blocks and save the data in Custom Data Segment parameters.

#### 4.1.4.3 Access Rights Checks

There is an "Access" Clause in the Header of a CL Block that governs the rights and restrictions for CL stores to MODE, SP, OP, RATIO, and BIAS parameters of other points (the same rules apply to stores of these parameters in the bound data point, but they are seldom used). The Access clause can specify PROGRAM or CONT CONTROL access. CONT CONTROL is the default for continuous CL programs, but PROGRAM can be selected as an option. PROGRAM is the only access type used for sequence programs in the Multifunction Controller—it need not be explicitly selected.

PROGRAM access requires that the Mode Attribute for the destination point be "Program" and then follows Operator-type Mode rules, as follows:

- Stores to the mode parameter require the Mode Attribute of the destination point to be "Program."
- Stores to SP require the Mode Attribute to be "Program" and the Mode to be Auto (or, for PID points, Manual w/o PV tracking).

- Stores to OP require the Mode Attribute to be "Program" and the Mode to be Manual.

#### NOTE

Stores from on-process Universal Stations require the Mode Attribute to be "Operator" for changes to Mode, SP, and OP.

CONT CONTROL access obeys rules that are similar to those followed by normal Regulatory Data Point Output Processing, as follows:

- **Stores to MODE** are not affected by the Mode Attribute of the destination point.
- **Stores to SP** require CAS Mode and are unaffected by the Attribute. In addition, for HG points, RCASENB must contain On and the algorithm must be an SPC algorithm or a DDC algorithm without PV tracking.
- **Stores to OP** (if legal at the destination point) require CAS Mode and are unaffected by the Mode Attribute. Stores to OP are not permitted on AM points. For HG points, the algorithm must be a DDC algorithm and RCASENB must contain On.

In general SP and OP processing are accomplished by stores to the CV parameter (see Section 26 in *Application Module Algorithm Engineering Data*). To bypass the standard control-output processing, in addition to handling windup and initialization, the CL program must be involved in assuring that an operator's request to transfer to CAScade mode occurs. Refer to 4.4.2.1 in *System Control Functions* for a discussion of cascade-mode requests.

We recommend (but don't require) that CL programs that store to SP and OP with CONT CONTROL access, check the point's CASREQ parameter before storing in MODE. If CASREQ contains Request, the program should store Cas in MODE and exit (note that the store to MODE will occur even if CASREQ doesn't contain Request). The next time the program executes, it can store in SP or OP. The reason the program should exit and not store in SP or OP until the next processing pass is to delay to give the box enough time to change the mode. This is an example of such a program:

```
BLOCK storesp (generic; at pst ctag)
  EXTERNAL hg05 -- HG point
  If hg05.casreq = request then (set hg05.mode = cas; exit)
  If hg05.mode = cas then set hg05.sp = hg05.sp + 1
END storesp
```

Regardless of the Access selection, stores to Ratio or Bias, of the PID Ratio/Bias Algorithms, require the Attribute to be Program for CL stores.

AM/CL stores of MODE and MODATTR parameters to the same Multifunction Controller slot require that the values be stored on separate executions of the CL. The example below describes a method to change both parameters with multiple execution cycles of the CL.

```
-- THIS IS A CL THAT WILL CHANGE AN MC SLOT TO P-AUTO, THEN CONTINUE
BLOCK mdpauto (GENERIC; AT GENERAL)
  EXTERNAL hg0501
  IF hg0501.MODATTR <> PROGRAM THEN
```

```

&      (SET hg0501.MODATTR = PROGRAM; EXIT)
-- ONCE THE MODATTR IS PROGRAM, THEN CHANGE THE MODE
   IF hg0501.MODE <> AUTO THEN SET hg0501.MODE = AUTO
-- CONTINUE ON
....
END mdpauto

```

#### NOTE

Stores from on-process Universal Stations require the Mode Attribute "Operator" for changes to Ratio and Bias.

Storage (write) access by CL blocks bound to another data point is governed by restrictions defined for the particular parameter, as it is configured.

#### 4.1.4.4 Range and Limit Violations

Stores to SP are normally clamped so that they stay within the SP Limits (if they exist) for the destination point (points in Basic Controllers don't have SP Limits). This clamping applies for normal output processing logic, as well as for direct CL stores over interpoint connections within boxes and within AMs; however, when the connection is from an AM to a process-connected box, the rules for output processing allow the higher-level continuous control by the AM to violate the limits in the box.

For connections between AMs and Boxes, the following rules apply:

- For access where the attribute is PROGRAM, SP clamping of the store occurs.
- For CONT CONTROL access, the rules used by output processing for AM-to-Box cascades are followed. Values outside the SP limits (extended range in the case of CB and MC points) are clamped at the SP limits. More restrictive checks, if necessary, must be directly supplied by the CL code.
- Direct stores to the OP parameter in Boxes (or UCN Nodes) from CL under PROGRAM access are clamped to stay within the Box OP limits.

The values directly stored in OP of other data points are not clamped to stay within the OP limits. These limits apply to only the value of OP that are calculated within the point. That is, they are applied when the point creates an OP from a CV value, whether the CV is computed by a standard algorithm or by a CL Block. Direct stores to the OP parameter from Universal Stations, Output Processing of another point, or CL under CONT CONTROL access are not clamped unless the value is outside the -6.9% to 106.9% range, in which case it is clamped at the corresponding range limit for output processing and CL, and an error indication appears at the Universal Station(s).

For other parameters (other than SP & OP) that have configurable ranges (for example, PV), stores from CL that exceed the defined range of the destination parameter are clamped at the end of the range.

If the parameter has a system-fixed range (for example, a control-tuning parameter that must always be positive), attempts to store a value outside the range are rejected.

#### 4.1.4.5 Data-Transport Errors

There are two types of data-transport errors:

- **Nonexistence Errors**—An attempt to read or write a parameter of any data point cannot be performed if that point or parameter does not exist. Such an error is called a "nonexistence" error. These errors are normally discovered when the CL program is compiled or linked, but they can be detected when the program is running (at "runtime"), if the database has been modified. A nonexistence error at runtime causes the Configuration Error condition of the CL Block.
- **Communication Errors**—An attempt to read or write to a data point in another device results in a communication error if the destination box or module is inactive, or if the network connection to it is not operating.

#### 4.1.4.6 Bad Values

Parameters whose type is number can have a "bad" code for the value that indicates that a legitimate number cannot be determined. This happens under any of the following conditions:

- A PV is bad because its range has been exceeded (transmitter failure) and the clamping option was not selected.
- A limit or alarm trip point has been set Bad to bypass limiting or alarming action (this applies to only AM points).
- A communication error or device failure has prevented acquisition (fetching) of the value.
- A value is set to Bad, using the SETBAD CL Subroutine.

#### 4.1.4.7 CL Actions on Bad Values and Communication Errors

If no extra consideration is given by the customer engineer, a communication error (for example, box failure or node failure), or the use of a bad value that is a result of any of the above functions, results in the aborting of the CL Block. The effect of the CL abort is described below under "Execution Errors." This is accomplished as follows:

- For parameter types other than number, the abort occurs immediately as the CL block attempts to access the parameter.
- For a parameter whose type is number, a bad value code is used as the value, and expression evaluation (arithmetic) is allowed. The bad value code propagates through the expressions, always with a bad value result. The CL block is aborted on any attempt to store the bad value or on any attempt to use the bad value to make a decision (this could result in incorrectly storing in another parameter).

If a CL abort is not the desired action in these conditions, the Customer Engineer can use one of the following courses of action:

- In normal (Foreground) CL, the **Communication Error** (COMM\_ERR) Function is provided to allow a CL Block to check for transmission error on a prefetched parameter. This reference to a parameter with a communication error will not cause an abort. The Comm\_Err function operates on all types of parameters.
- In both Foreground and Background CL, the **MOVE\_PARAMETER** Subroutine can be used to fetch (or store to) off-node values. Communication or configuration errors encountered by a Move\_Parameter call result in no value move and an appropriate status return, but will not cause an abort.
- Additionally, the **BADVAL** function is provided to check a parameter whose type is number for a bad value. This function allows the convenience of checking only the result of a complex expression, rather than checking each of several inputs to it—if that is the desired action. This check is generally inserted before storing the result, so that alternate logic, can be executed to avoid storing a bad value.
- Finally, it may be desirable to **allow the storing of a bad value** so that other logic in the system may continue its propagation, or simply to detect its presence. This function is provided by the ALLOW BAD and SET BAD subroutines. ALLOW BAD(x,y) stores "y" into "x" even if "y" is bad, and does not abort the CL Block. (Both "x" and "y" may be direct or indirect point-parameter references. SET BAD (x) generates a "bad value" (NaN) code and stores it into "x" without aborting the CL Block.

#### 4.1.4.8 Infinities

Certain operations on values whose type is Number cause a positive or negative infinity to be created as the result. These cases include the following:

- **Dividing by zero**—the result is infinity with the sign of the numerator.
- **Arithmetic overflow**—(for example, adding one to the largest finite value that the internal number system can hold.) The result is an appropriately signed infinity.

#### NOTE

**These are NOT error conditions.** Infinite values are not Bad Values; they are handled in calculations and comparisons according to normal mathematical rules. For example, adding anything finite to a positive infinity yields a positive infinity; in a comparison, a positive infinity is greater than any finite value, and a negative infinity is less than any finite value. If this handling of infinite values is not deemed appropriate, the CL function, FINITE, can be used to test for finite/not finite (not finite includes bad values and infinities), and take the desired action.

#### 4.1.4.9 Execution-Time ("Runtime") Errors

There are three classes of execution-time errors: (1) **CL Error conditions**, (2) **CL failures**, and (3) **minor errors**. If any of these conditions occurs, a code that defines the reason for the error is stored in the CL segment. This code can be displayed at a Universal Station or can be accessed by CL logic, including CL Blocks bound to the same data point. If multiple errors have occurred, the reason code in the CL segment indicates the most severe error.

##### CL Error Conditions:

- **Configuration error (CNFERR)**

- 1) A fetch or store couldn't take place because the parameter or point doesn't exist, or because the parameter's current data type does not match the data type of the parameter when the CL block was linked.
- 2) An indirect reference is attempted through a null data-point ID within a point ID array.
- 3) A CL program is linked to a certain array data-type definition, and the data type changes so that the array becomes larger, and there is a subsequent attempt to access that parameter by a CL program (if this happens the CL block must be unlinked, recompiled, and relinked).
- 4) A CL program attempts to fetch an array parameter in an HG. (Such a program compiles and links properly. To avoid this problem, refer to the array parameter as an individual point. This is the case for MC numeric, flag, and timer points.)
- 5) The second or a subsequent store to a value-array-size parameter of reserved data point \$AMIF (only once can a value be stored in these parameters).
- 6) An attempt to fetch or to store to the string array size, value revision, value, or status parameter of reserved data point \$AMIF, before the \$AMIF configuration has been established.

7) An attempt to fetch or store a \$AMIF parameter that is legal but has not been established (created) by a user-written CL on AM startup.

- **Key-level Access Error (KEYLEVEL)**—An attempted store could not take place because of a key-level restriction. That is, a parameter store was attempted that is not legal from continuous control or program access.

- **Program Error (PROGERR)**—CL execution reached some illogical condition that indicates an error in the CL program (a "bug"). These are the situations that cause a program error:

- 1) The label that is the target of a GOTO statement is in the middle of a REPEAT loop.
- 2) None of the conditions in a conditional SET statement are true.
- 3) The CL block cannot get more memory while it is executing. When this happens, it is typically on a subroutine call or function call. This can be corrected by increasing the stack size for a CL running at a Background insertion point, or by reducing the quantity of CL local variables defined in the block.
- 4) Use as a loop index, a variable that has not been stored to, or a parameter whose value is "bad."

In cases 1), 2) and 4), the location of the error in the CL block is placed in parameter CLERRLOC(n) in the CL segment of the bound data point. In the third case, a zero is placed in CLERRLOC. More information about CLERRLOC is under 6.3 in *Control Language/Application Module Data Entry*.

- 5) A CL block attempted to call a subroutine or function in an external load module that is not loaded. (See the *Control Language/Application Module Reference Manual* for CL Runtime Extension information.)

- **Range Error (RANGE)**—An attempt was made to store a value outside the fixed range of a parameter that has a fixed value range, rather than a configurable value range.

- **Array Limit (ARRAYLIM)**—An attempt was made to access outside the bounds of an array.

- **Backward Branch Violation (BRANCHV)**—The legal number of backward branches has been exceeded. When a GOTO or REPEAT statement is executed that results in execution of a preceding portion of the program, a backward-branch counter is decremented by one. If the counter reaches zero, BRANCHV violation has occurred. There is no backward branch limit for Background CL blocks.

There are two backward branch counters: CLBACKF (fast processor) and CLBACKS (slow processor). These are processor status data-point parameters and they can be accessed and changed through a custom display (see Section 22 in the *Engineer's Reference Manual*). They can store a maximum count of 32,767. The default value in CLBACKF is 100 and the default value in CLBACKS is 500. Note that setting the value in one of these parameters too high can delay point processing if a CL block gets into a long loop. This can cause the AM to fail.

If the point containing the CL block executes due to a process special request (PPS), the backward branch counter associated with the point's assigned processor is used. If an unscheduled point (PERIOD is defined as NOPERIOD) executes due to a PPS request, the CLBACKS counter is used. Note that a CL block executing in background does not check for backward branch violations.

- **Arithmetic Error (ARITHERR)**—An attempt was made to store an infinite number of seconds into a time variable or time parameter. Note that an attempt to store a bad value number of seconds into a time variable or time parameter results in a BADVALST error.

#### CL Failure Conditions:

- **ABORT statement (ABORT)**—Execution of an ABORT statement, or for Background CL blocks only, operator request or program store to \$CLCMPST.
- **Communication error abort (COMABORT)**—Attempted use of a parameter, other than one whose type is Number, that had a communication error on the attempt to fetch the value.
- **Bad value store (BADVALST)**—Attempt to store a Bad value without using the ALLOW\_BAD or the SET\_BAD subroutine, or the use of a Bad value in making a decision.

#### Minor Errors:

- **Communication error (COMMERR)**— (1) A network fetch or store couldn't take place because of a communication error or because the device accessed wasn't on-line. (2) A failed attempt to obtain memory for the \$AMIF reserved data point initiated via a store to an array-size string or a string-length parameter.
- **Rights error (RIGHTS)**—A temporary restriction, removable by the operator, prevents the storing of a value. Examples are Mode and Mode Attribute restrictions on stores to SP and PVSOURCE restrictions on stores to PV.



- **Limit violation (LIMVIOL)**—An attempt to store a value that exceeded SP Limits, OP Limits, or Range Limits. This error is not generated for stores to this point's PVCALC or OP parameters, even if the stored value exceeds the range limits or OP limits. The store is completed and the value is clamped at the limit.

CL Error Conditions are expected to result from CL programming errors ("bugs") or other errors that require CL coding changes for successful recovery. This is similar to the ERROR state of a sequence program in an MC. The following actions are taken:

- The location count for the CL instruction where the error occurred is placed in CLERRLOC in the CL segment (for poststore errors, the value in CLERRLOC is zero). There is one CLERRLOC parameter for each CL slot in the segment. Refer to 6.3 in *Control Language/Application Module Data Entry* in the *Implementation/Application Module - 2* binder to learn how to determine which statement has the error.
- The CL Error Alarm condition (CLE for Foreground CL and BCLE for Background CL) is set, causing a process alarm to be generated on this point.
- A CL Abort occurs on the point and point processing (including CL execution) is attempted on the next scheduled processing pass of the point. There is an exception: The Abort cannot occur if the error is a "store" configuration error to another node.
- The CL Error Alarm condition is an "off-normal" condition that remains true until point processing for the point again normally completes with no CL Aborts, at which time the "return-to-normal" for the alarm is registered.

CL Failure Conditions are expected to result from equipment failures, such as a sensor failure, a box, module, or gateway failure, or a communication failure. Also, the explicit abort from execution of a CL ABORT statement is included in this class. This is similar to the Fail state of a sequence program, in that the system, rather than the CL code is expected to be at fault.

The actions taken are similar to those taken for the CL Error Condition:

- The Alarm condition is set, in this case, the CL Failure Alarm (CLF for Foreground CL and BCLF for Background CL). This causes a process alarm to be generated on this point.
- A CL Abort occurs on the point and point processing, including CL execution, is attempted on the next scheduled processing pass of the point.
- The CL Fail Alarm condition is an "off normal" condition that remains true until point processing for the point is completed normally with no CL Aborts. At that time the "return-to-normal" for the alarm is registered.

Minor errors are expected to result from conditions in the process or in the control strategies, such as modes that preclude stores or limits clamping the stored values. No alarms or aborts occur.

Note that in sequence programs, rights errors and communication errors are considered to be failures and the sequence is stopped to wait for operator intervention. For continuous CL programs, however, they are assumed to be acceptable operation overrides.

Any necessary checks should be made by explicit CL code before attempting to store values.

If a CL Abort caused by an Abort statement or some other error has occurred, the action taken is to skip all remaining processing of the point, except for distribution of the alarm conditions already detected on the present processing pass.

Additionally, if the abort occurs at an insertion point up to and including the post-PV processing insertion point, the following occurs:

- PVCALC and PVAUTO are set Bad.
- If PVSOURCE = Auto, the PV status goes Bad and a Bad-PV alarm is requested.
- The PV initialization condition is set.
- The Control initialization condition is set.

If the abort occurs at any insertion point after the calculation of the PV, the following occurs:

- The PV is left as is,
- The Control initialization condition is set.

If no errors have occurred in the last pass, a state of NORMAL is reported.

## 4.1.5 CL Segments

Each data point to which CL Blocks can be linked has a CL Segment, which holds data that relates the CL Blocks to the data point. The CL segment is written to and read from by the Linker (and its loader) and by the standard control software. Some of the CL Segment's parameters can be accessed by other data points and by user-written programs.

### 4.1.5.1 CL Segment Structure

The following chart lists user-visible parameters in CL segments related to CL blocks that are bound to AM points:

Parameter Significance-->	CONFIGURATION	OPERATION	STATUS/INFO
Parameter Names & ----> Access Locks	CLSLOTS (B) BLKNAME (L) (1..CLSLOTS) INSPOINT (L) (1..CLSLOTS) INSORDER (L) (1..CLSLOTS)	CLACTIVE (EP) (1..CLSLOTS) \$CLCMPST (EP) (1..CLSLOTS) \$BKGSTS (V) (1..CLSLOTS)	BLKTIME (V) (1..CLSLOTS) CLERRSUM (V) CLBLKERR (V) (1..CLSLOTS) CLERRLOC (V) (1..CLSLOTS) BRANCHES (V) (1..CLSLOTS) CLVERSIN (V) (1..CLSLOTS) CLREVISN (V) (1..CLSLOTS) CLUSECNT (V) (1..CLSLOTS) \$BKGABRT (V) \$BKGQFUL (V) \$BKGTIME (V) \$BKGPRTY (V) (1..CLSLOTS)
Parameter-Access Locks: (V)—View only (L)—Entered by CL Linker (B)—Data Entity Builder, only (E)—Engineer, Program, or DEB (EP)—Engineer or Program			

The values in BLKNAME, INSPOINT, INSORDER, \$CLCMPST, BLKTIME, CLERRLOC, CLERRSUM, and CLBLKERR appear on the CL page of the point's Detail Display. CLSLOTS, CLVERSIN, CLREVISN, CLACTIVE, \$BKGSTS, BRANCHES, CLUSECNT, \$BKGABRT, \$BKGTIME, \$BKGQFUL, and \$BKGPRTY can be included in a custom display.

#### 4.1.5.2 CL Slots

A CL segment contains one slot for each block that can be bound (linked) to the data point. The CL Linker ensures that all occupied slots are contiguous and begin with the first slot in the segment. All unoccupied, or "not configured" slots are grouped at the end of the segment.

Slots are maintained in order, first by insertion point, then by their specified order within the insertion point. Blocks that do not specify an execution order are executed after all those that are so specified.

#### 4.1.5.3 CL Slot Parameter Access

The fields of the slots are accessed as though they were elements of an array. They are indexed by the slot number. A block can determine its own slot number to use as the index, when the block is executing, by using the SELF function in the CL code. For example:

`A101.CLACTIVE(x)` —Active flag of Slot x

`A101.BLKNAME(Self)` —Block name of self

Note that a CL Block should never use a fixed, predetermined number for the slot index because this can change as other blocks are inserted or deleted on the data point.

#### 4.1.5.4 CL Segment Configuration Parameters

**CLSLOTS**—The "number of slots" field of the CL segment (CLSLOTS) is set up when the data point is built. It can be changed by rebuilding the data point through the Data Entity Builder (DEB). This allows CLSLOTS to be increased, making space for more CL blocks without affecting the blocks already linked to the point. The CLSLOTS parameter value cannot be decreased on a rebuild operation.

**BLKNAME**—A nonvisible parameter in the data point points to the block's entry in the CL Object Catalog in the AM. This gives the name (BLKNAME) and a time stamp (BLKTIME) of the block (corresponding to the Block name and time stamp on the CL listing) and allows the control system to locate the object code.

**INSPOINT & INSORDER**—The Insertion Point Number (INSPOINT) and Insertion Point Order (INSORDER) fields specify the insertion point at which the block is to be executed and provide an optional specification for the order of execution within the insertion point. The content of these fields are originally specified by the CL user in the AT clause of the block heading. For further information, see Figure 3-2 (under 3.1.3 in this publication) and refer to the *Control Language/Application Module Reference Manual*.

#### 4.1.5.5 CL Segment Operation Parameters

**CLACTIVE**—The CL Active parameter (CLACTIVE) has three states: NOTCONFIG, ACTIVE, and INACTIVE. These indicate whether the slot is vacant or has a CL Block linked to it that is either active or inactive. A block that is inactive cannot execute. A block that is active executes whenever its specified execution condition in the CL Block Heading statement is true. If no execution condition is specified, the block is unconditionally executed.

**\$CLCMPST**—For operating and display convenience, the operating state of the slot (\$CLCMPST(I)) is derived from the values of \$BKGSTS(I) and CLACTIVE(I). For Foreground blocks the values are identical to CLACTIVE(I). For Background blocks, the following table shows the possible \$CLCMPST values.

\$CLMPST(I)		CLACTIVE(I)	\$BKGSTS(I)
NOTCONFIG	(R)	NOTCONFIG	-----
INACTIVE	(R/W)	INACTIVE	OFF
INACTIVE	(R/W)	INACTIVE	QUEUED
ACTIVE	(R/W)	ACTIVE	OFF
QUEUED	(R)	ACTIVE	QUEUED
RUNNING	(R)	ACTIVE	RUNNING
DELAYED	(R)	ACTIVE	DELAYED
WAITING	(R)	ACTIVE	WAITING
INA_RUN	(R)	INACTIVE	RUNNING
INA_DELAY	(R)	INACTIVE	DELAYED
INA_WAIT	(R)	INACTIVE	WAITING

The operator can change the status of \$CLMPST between active and inactive from the Detail Display. (Note that when this parameter is selected the "pick" includes ABORT which is not a state but a command which will stop execution of a slot's Background CL Block until the point's next processing cycle. See heading 4.1.6.2.)

INA\_RUN, INA\_DELAY, and INA\_WAIT states occur when the CL Block associated with an inactive CL slot is still executing (see heading 4.1.6.3).

**\$BKGSTS**—The Background Status parameter (\$BKGSTS)—which applies only to slots containing Background CL Blocks—has five possible states:

OFF—The CL Block is available for execution.

QUEUED—The CL Block is in the queue waiting to be executed.

RUNNING—The CL Block has started execution and has not yet completed. It may not actually be running because of its priority level.

DELAYED—The CL Block has called a delay and is waiting for the delay to complete.

WAITING—The CL Block has called an off-node function and is waiting for completion of that function.

#### 4.1.5.6 CL Segment Status and Information Parameters

**BLKTIME**—See the description of the CL Segment configuration parameter BLKNAME.

**CLERRSUM**—The Error Status Summary field (CLERRSUM) reflects the most severe error that occurred for any block bound to the data point, as a result of the previous processing pass. This field is updated by the standard control software to reflect the results of a poststore error. The value in CLERRSUM is changed to Noerror when the point's execution state is set to Active, and its value is then redetermined each time the point is processed.

**CLBLKERR**—The Error Status field (CLBLKERR) indicates the most severe error encountered the last time this block was executed. If there were no errors detected the last time the CL block was executed, or if the CL Block's activity status changed to Inactive, the value in CLBLKERR is set to Noerror. The values in the CLERRSUM and CLBLKERR parameters are those in parentheses under 4.1.4.9 in this publication.

**CLERRLOC**—The listing generated when a program is compiled shows the location-counter value for the beginning of each CL statement (see 6.3 in *Control Language/Application Module Data Entry*). This location-counter value will be found in the parameter CLERRLOC for a CL block when one of the following occurs

- a) an error occurred while executing that statement
- b) a delay was called from that statement (Background CL only)
- c) a wait (caused by an off node data access, a file I/O call, etc.) occurred in that statement (Background CL only)

CLERRLOC contains zeros if the CL block executed to completion on its last pass. Note that an error on a post-store from Foreground CL does not abort the CL block execution, so that CLERRLOC will contain zero in this case.

**BRANCHES**—contains the number of backward branches taken the last time the CL block was executed. This parameter is not used for Background CL blocks.

**CLVERSIN**—contains the version number of the CL Compiler that was used to compile the CL block.

**CLREVISN**—contains the revision number of the CL Compiler that was used to compile the CL block.

**CLUSECNT**—contains the number of uses of the CL block for a specific unit.

**\$BKGABRT**— Equals TRUE if the Background CL blocks on the point were aborted the last time one was being executed.

**\$BKGQFUL**— Equals TRUE if the queue was full the last time that Background CL was scheduled to be queued.

**\$BKGTIME**—Contains the time (including queue time) in seconds that Background CL has been running on the point.

**\$BKGPRTY**—Background CL Blocks execute at three different priority levels, controlled by the parameter \$BKGPRTY. Its values are: LOW, MEDIUM, and HIGH. Change of priority is controlled by the CL block itself through the subroutine BKG\_CHANGE\_PRIORITY (see the *Control Language/Application Module Reference Manual* for details). All Background CL Blocks start execution at HIGH priority.

## 4.1.6 Background CL Operation

### CAUTION

Background CLs restart from the beginning after failover of a redundant AM. Users of background CL on redundant AMs must be careful to program for redundancy. Refer to subsection 7.4 in *Application Module Implementation Guidelines*.

#### 4.1.6.1 Initiating Background CL Blocks

Background CL blocks are initiated (queued) at the end of a point's execution. If none of the point's background blocks are running (\$BKGSTS for all ACTIVE background blocks is OFF), the point is queued for background processing.

When a point's background CLs cannot be queued because the queue limit of 50 points has been reached, the queue status is reported as follows:

- 1) The point parameter \$BKGQFUL is set TRUE.
- 2) The AM's PSDP parameter BKGQFUL is set TRUE.
- 3) An Auxiliary Status alarm "Background CL queue full" is output, causing notification on the Node Status display.

The system continues trying to queue the point's background CLs each time it is processed. When a background CL again can be queued the following happens:

- 1) The PSDP parameter BKGQFUL is set FALSE.
- 2) An Auxiliary Status message "Background CL queue available" is output.
- 3) The point parameter \$BKGQFUL is set to FALSE (but not until after successful completion of the next point processing cycle for Background CL blocks).

Be cautious in setting a Background CL Block to either low or medium priority. A block at low or medium priority can run only when there are no blocks of higher priority that are ready to run. For example, you have a number of high priority blocks that run frequently, a low or medium priority block can take an excessive length of time to complete.

#### 4.1.6.2 Aborting Background CL Blocks

The operator can abort a running Background CL Block from the CL page of the point's detail display by selecting the status displayed in the Activity column and then selecting the ABORT\_CL pick. A program can abort a running Background CL Block by storing ABORT\_CL to the \$CLCMPST parameter of the CL slot.

Aborting the running Background CL Block also prevents execution on this pass of any other active Background CL blocks for that point. Aborting a Background CL Block does not affect background CL execution the next time that the point is processed. When the point is next processed, it begins execution of the active Background CL Blocks in their normal sequence.

A Background CL Block can detect that an abort occurred during the latest execution of the point's background CL block(s) by testing the point's parameter \$BKGABRT. When background for a point has been aborted, this parameter remains TRUE until all active Background CL Blocks on the point have subsequently run to normal completion.

#### 4.1.6.3 Deactivating Background CL Blocks

An operator can set a point inactive from the first page of its Detail Display by changing the point's execution state to INACTIVE. A program sets a point inactive by storing INACTIVE to the parameter PTEXECST.

Setting a **point** Inactive stops the execution of all Background CL Blocks on that point. It causes the abort of a running Background CL Block or prevents a queued block from running.

An operator can set a Background CL Block inactive from the CL page of the Detail Display. A program sets a Background CL Block inactive by storing INACTIVE to the slot's \$CLCMPST parameter.

Setting a Background CL **Block** Inactive does not abort its execution if it is already queued or running. Inactivating a block takes effect the next time that the point's background blocks are to be queued.

#### 4.1.6.4 Interruption By Other CL Blocks

Since Background CL blocks run at low priority, they will be interrupted by other AM functions including Foreground CL blocks and Background CL blocks of equal or higher priority. This interruption can occur between two background block data accesses. CL programs that share data must take this probability into account.

#### 4.1.6.5 Foreground Processing Parameters

Several parameters on a point convey meaning only while the point is in Foreground CL execution and should not be used by Background CL blocks. These include RESTART, PATHIND, and alarm transition flags.

#### 4.1.7 CL Runtime Extensions

Applications oriented subroutines and functions that are callable by CL/AM programs can be purchased from Honeywell and added to your system. These subroutines and functions are known as CL Runtime Extensions and are packaged in files called Include Sets. These optional files are loaded into the AM during system startup and configuration.

Information on available Include Sets and the subroutines and functions that they contain is available with the optional product or package.



## 4.2 CL SWITCHES ON AM REGULATORY DATA POINTS

CL Switches can be added to Regulatory Data Points in an AM to communicate the state of custom CL strategies to Universal Station operators, and to provide a standard way for the operator to make strategy-change requests to the CL logic. The most visible parameters of a CL switch are the following:

- Current state—SnCURSTS that indicates the current state of the switch, as determined by a CL Block.
- Requested state—SnREQSTS indicates the requested position of the switch. Usually, the state request is made by an operator.

A store to the requested-state parameter initiates special processing of the data point on which the CL switch is configured. Therefore, programmers should use care when writing programs that write to these parameters, so as to avoid unnecessary processing. It is expected that CL Blocks bound to the data point will see each change in the requested state, check to see if the change is valid, and act accordingly. These CL Blocks can be configured to execute either when the CL switch position changes (implemented by a WHEN clause in the CL Block's header) or each time the point is processed (no WHEN clause specified). Switch CL Blocks that implement switch-strategy logic can be linked to any valid insertion point on the data point.

CL switches can have up-to-five states. The names of these states are configured through the Data Entity Builder. The main advantage of CL switches over simple enumerations in custom data segments is that CL switches have some supporting information on the points' detail displays. Examples of CL switch use include switching between alternate inputs, as the validity of inputs change, changing cascade-control schemes, and changing algorithm-related constants as required by process conditions (limit violations, etc.).

### 4.2.1 CL Switch Data Point Parameters

A CL Switch is added to a Regulatory Data Point by configuring something other than zero for the data-point parameter NUMSWTCH. Up to two switches can be added to an AM Regulatory data point.

Configuring a switch on the data point adds the parameters shown on the following chart:

	CONFIGURATION	OPERATION	STATUS/INFO
GENERAL OPERATION	NUMSWTCH -B S1NSTATE -B S1STATES -B (0..X)* S1ACCLVL -EB (0..X)*  S2NSTATE -B S2STATES -B (0..X)* S2ACCLVL -EB (0..X)*	 S1REQSTS -O S1CURSTS -P  S2REQSTS -O S2CURSTS -P	
Parameter access locks: B—DEB only EB—DEB or Engineer key only P—Program, connection from a continuous-control data point or DEB. O—Operator, Supervisor, Engineer, DEB, or program *X = 1 less than SnNSTATE			

Parameter Definitions (n = switch number; 1 or 2):

NUMSWTCH	Number of switches on the point: 0, 1, or 2.
SnNSTATES	Number of valid states of the switch. An integer in a range from 2 through 5.
SnSTATES	Character-string representation of each valid current and requested state, where n is 1 or 2. An array of up to five 8-character strings. The size of the array is determined by the value in SnNSTATES.
SnACCLVL	Access-level choice for SnCURSTS parameter, where n is 1 or 2. Value is one of the following enumerations: Operator 1, Operator 2, Engineer, and Program.
SnCURSTS	Current state of the switch, where n is 1 or 2. Self-defining enumeration. The parameter that holds the state definition is SnSTATES.
SnREQSTS	Requested state of the switch, where n is 1 or 2. A self-defining enumeration. The parameter that holds the state definition is SnSTATES. The access level for this parameter defaults to operator and is further qualified by the SnACCLVL parameter.

You can configure additional parameters required for switch operation and the switch/operator interface by configuring Custom Data Segments (see 3.6).

### 4.2.2 Changing State Representations in CL Switch Points

When a self-defining enumeration is declared, the assignment of values to each state follows the declared order of the states. When a CL Block is compiled, code generated for the switch states is based on the specific point in the database or it is based on the type assertions in the CL source code. When the block is linked, these assertions are rechecked for validity.

#### **WARNING**

A process engineer can change the state representations of self-defining enumerations (parameter states) any time; however, CL operates on the basis of the old states until the CL Block is relinked.



---

# Index

---

Topic	Section Heading
Access Rights Checks (CL)	4.1.4.3
Access Status, Regulatory Point Interconnections	3.1.4.8
Accumulation Alarming, Counter Points	3.3.7
Accumulation Processing, Counter Points	3.3.6
Accumulator Control, Counter Points	3.3.8
Activity Status, Regulatory Point Interconnections	3.1.4.7
Advisory Deviation Alarming, Regulatory Points	3.1.6.3
Advisory Deviation Alarms, defined	2.3.4
After and Before Assignments	2.2.1.4
Alarm Checks, PV – Regulatory Points	3.1.5.7
Alarm Configuration, Counter Points	3.3.12.2
Alarms	
Accumulation – Counter Points	3.3.7
Advisory Deviation, defined	2.3.4
Advisory Deviation – Regulatory Points	3.1.6.3
Bad Configuration	2.3.6
Bad Control	2.3.5
Bad PV	2.3.7
CL Error	2.3.8
Custom Data Points	3.6.3
Custom Process	2.3.11
Deadbands	2.3.3
Detection	2.3.1
Deviation – Counter Points	3.3.5
Functions - Switch Data Points	3.7.3
Off Normal, Flag Points	3.4.3
Operations (Unit)	2.3.12
Pre-Preset and Pre-Pre-Preset	2.3.10
Process	2.3
PV, Counter Points	3.3.4
Regulatory Points	3.1.8
Trip Points	2.3.2
Algorithm Processing, PV	3.1.5.2
AM	
Detection of Failed Nodes	2.1.4
Schedule Dumper:	2.2.1.7
Switch Data Points	3.7
Associated Display	2.2.2
Automatic Cycle Loading	2.2.1.3
Auxiliary Unit	2.3.16
Background CL	4.1.1, 4.1.6, 4.1.4.2
Bad Configuration Alarm	2.3.6
Bad Control Alarm	2.3.5
Bad PV Alarm	2.3.7
Bad Value Handling at Point Interconnections	2.1.3.7
Bad Values and Communication Errors, CL	4.1.4.7
Bad Values, CL	4.1.4.6
Before and After Assignments	2.2.1.4
Bias and Ratio Options, Regulatory Points	3.1.6.4
Bias Limits	3.1.9.3
CAScade Mode, Limiting	3.1.9.7
Changing State Representations, CL Switches	4.2.2
Checkpointing the AM Database	2.7.1

# Index

Topic	Section Heading
CL	
Connection to Data Points	4.1.2
Error Alarm	2.3.8
Execution in the AM	4.1
Fatal-Error Alarm	2.3.9
Insertion Points	3.1.3, 3.6.5, 3.7.4
Runtime Extensions	4.1.7
Segment Data, Custom Data Points	3.6.4
Segments	4.1.5
Slots	4.1.6
Switches on AM Regulatory Points	4.2
Cold Restart	2.7.2
Communication Errors	2.1.1
Communication Errors and Bad Values, CL	4.1.4.7
Comparison, Functions	
AM and Process-Connected Boxes	3.1.13
Configuration	
Alarm	2.3.6
Counter Points	3.3.12
Errors	2.1.3.2
Flag Points	3.4.5
Numeric Points	3.2.5
Timer Points	3.5.5
Connection	
Counter Points	3.3.12.4
to Data Points, CL	4.1.2
Connections	
Contact Cutout	2.3.13, 3.1.4.3
Control Input	3.1.4.2
Control Output	3.1.4.5
General Input	3.1.4.4
General Output	3.1.4.6
PV Input	3.1.4.1
Contact Cutout Connections	2.3.13, 3.1.4.3
Control	
Initialization	3.1.10.2
Input Connections	3.1.4.2
Language	See CL Ref. Man.
Output Connections	3.1.4.5
Output Processing, Regulatory Points	3.1.13
Controlling Timer Points	3.5.1.1
Counter Input and PVAUTO Calculation	3.3.3
Counter Points	3.3
Custom Data Points	3.6
Custom Process Alarms	2.3.11
Cycle Assignment by Users	2.2.1.5
Cycle Loading, Automatic	2.2.1.3
Data Access. CL	4.1.4
Database Checkpointing	2.7.1
Data Point Descriptions	3
Data Point Processing	2.2
Data Transport Errors, CL	4.1.4.5
Deadbands, Alarm	2.3.3

# Index

Topic	Section Heading
Detection of Failed Nodes	2.1.4
Detection of Process Alarms	2.3.1
Deviation Alarming, Counter Points	3.3.5
Deviation Alarms, Advisory	2.3.4
Display-and History-Related Parameters	
Counter Points	3.3.10
Flag Points	3.4.2
Distribution, Messages	2.4.2
Dumper, AM Schedule	2.2.1.7
Dynamic Indirection	4.1.3
Error	
Handling	2.1
Key Level	2.1.3.6
Messages, Flag Points	3.4.6
Range	2.1.3.5
Value Storage	2.1.3.4
Error Alarm, CL, Fatal	2.3.9
Error and Warning Messages	
Counter Points	3.3.13
Numeric Points	3.2.6
Timer Points	3.5.6
Errors	
Bad Values and Communication, CL	4.1.4.7
Communication	2.1.1
Configuration	2.1.3.2
Data Transport, CL	4.1.4.5
Execution-Time, CL	4.1.4.9
Point Interconnection	2.1.3
Software	2.1.3.1
Execution States, Regulatory Points	3.1.2
Execution-Time Errors, CL	4.1.4.9
External Mode Switching	2.5
Failed Nodes, Detection of	2.1.4
Failure, Value Storage	2.1.3.3
Fast Point Processor (FPP)	2.2.1.1
Fatal-Error Alarm, CL	2.3.9
Filtering, PV - Regulatory Points	3.1.5.3
Flag Points	3.4
Functional Summary	
AM and Process-Connected Boxes	3.1.13
General Input Connections	3.1.4.4
General Output Connections	3.1.4.6
General Parameters, Switch Data Points	3.7.1
Guidelines, Override Control	3.1.11.1
History-and Display-Related Parameters	
Counter Points	3.3.10
Flag Points	3.4.2
Hold Your Breath	2.1.4
Hot Restart	2.7.4
Identification Data, Custom Points	3.6.1
Indirect References, CL	4.1.3
Infinites, CL	4.1.4.8
Indirection, Dynamic	4.1.3

# Index

Topic	Section Heading
Internetwork Point Processor (IPP)	2.2.1
Initialization	
Control	3.1.10.2
PV	3.1.10.1
Regulatory Points	3.1.9
Insertion Points, CL	3.1.3, 3.6.5, 3.7.4
Interconnection	
Errors	2.1.3
Regulatory Points	3.1.4
Interval Assignments	2.2.1.2
Key-Level Error	2.1.3.6
Level Error	2.1.3.6
Limit and Range Violations, CL	4.1.4.4
Limiting	
In CAScade Mode	3.1.9.7
in MANual Mode	3.1.9.8
PV Range in Numeric Points	3.2.2
Limits	
Bias, Regulatory Points	3.1.9.3
in Regulatory Points	3.1.8
Minimum Output Change, Regulatory Points	3.1.9.6
Output Rate-of-Change, Regulatory Points	3.1.9.5
Ratio, Regulatory Points	3.1.9.2
Setpoint	3.1.9.1
Loading, Automatic Cycles	2.2.1.3
Loading of Cycles by Users	2.2.1.5
MANual Mode Limiting	3.1.9.8
Message	
Distribution	2.4.2
Generation	2.4.1
Messages	2.4
Minimum Output Change Limits, Regulatory Points	3.1.9.6
Mode	
Permissives	2.5.2
Switching, External	2.5
Multiple Primmods Alarming Option	2.3.15
Numeric Points	3.2
Off-Normal Alarms, Flag Points	3.4.3
Output Limits	
Rate-of-Change, Regulatory Points	3.1.9.6
Regulatory Points	3.1.9.4
Override Control	
Guidelines	3.1.12.1
Regulatory Points	3.1.12
Parameter Access, CL	4.1.5.3
Parameters	
CL Switches	4.2.1
General, Switch Data Points	3.7.1
Period Assignments	2.2.1.2
Permissives, Mode	2.5.2
Point Interconnection Errors	2.1.3
Point Interconnections, Bad Value Handling	2.1.3.7
Point Processing, CL's Place in	4.1.1



# Index

Topic	Section Heading
Point Processing Special	
AM Counter Functions	3.3.9
AM Switch Data Points	3.7.1
Special Processing Requests	2.2.1.2
Pre-Pre-Preset and Pre-Preset Alarms	2.3.10
Primary Module Points	2.3.14
Process Alarm Detection	2.3.1
Process Alarms	2.3
Process Alarms, Custom	2.3.11
Processing Order, Regulatory Points	3.1.3
Processing	
Control Output, Regulatory Points	3.1.12
Counter Points	3.3.2, 3.3.12.4
Custom Data Points	3.6.2
Data Point	2.2
Flag Points	3.4.1
Numeric Points	3.2.3
Point - CL's Place in	4.1.1
Point-processing special	2.2.1.2
PV, Regulatory Points	3.1.5
Timer Points	3.5.3
Processors, Point	2.2.1
PV Alarm	
Bad	2.3.7
Checks, Regulatory Points	3.1.5.7
Counter Points	3.3.4
Deadbands	2.3.3
PV Algorithm Processing	3.1.5.2
PV Filtering, Regulatory Points	3.1.5.3
PV Initialization	3.1.10.1
PV Input Connections	3.1.4.1
PV Processing, Regulatory Points	3.1.5
PV Range Checks, Regulatory Points	3.1.5.5
PV Range Limiting, Numeric Points	3.2.2
PV Source Selection, Regulatory Points	3.1.5.4
PV Target Value - PVTv	3.3.5
PV Value Status, Regulatory Points	3.1.5.6
PVAUTO Calculation, Counter Points	3.3.3
Range and Limit Violations, CL	4.1.4.4
Range Checks, PV, Regulatory Points	3.1.5.5
Range Error	2.1.3.5
Range Limiting, PV, Numeric Points	3.2.2
Ratio and Bias Options, Regulatory Points	3.1.6.4
Ratio Limits, Regulatory	3.1.9.2
References, Indirect - CL	4.1.3
Regulatory Data Points	3.1
Regulatory Point	
Alarms	3.1.8
Initialization	3.1.10
Limits	3.1.8
Reset Windup, Regulatory Points	3.1.11
Restart	
Cold	2.7.2
Hot	2.7.4
Warm	2.7.3

# Index

Topic	Section Heading
Restarting AMs	2.7
Restarts	
Counter Points	3.3.11
Flag Points	3.4.4
Numeric Points	3.2.4
Timer Points	3.5.4
Runtime Errors	4.1.4.9
Runtime Extensions to CL	4.1.7
Schedule, Custom Data Points	3.6.2
Schedule Dumper	2.2.1.7
Scheduling	
Counter Points	3.3.2, 3.3.12.5
Data Points	2.2
Flag Points	3.4.1
Timer Points	3.5.3
Segment Data, CL Custom Data Points	3.6.4
Segments, CL	4.1.5
Setpoint Handling, Regulatory Points	3.1.6
Setpoint Limits	3.1.9.1
SP/OP Tolerance Check	3.1.7
Slots, CL	4.1.6
Slow Point Processor (SPP)	2.2.1
Software Errors	2.1.3.1
Source Selection, PV – Regulatory Points	3.1.5.4
Special point processing	2.2.1.2
SP Target-Value Processing, Regulatory Points	3.1.6.2
State Representations, Changing, CL Switches	4.2.2
Status, Access – Regulatory Point Interconnections	3.1.4.8
Status, Activity – Regulatory Point Interconnections	3.1.4.7
Storage Failure	2.1.3.3
Switch	
Alarm Functions, Switch Data Points	3.7.3
Data Points	3.7
Switches, CL on AM Regulatory Points	4.2
Target-Value Processing, Regulatory Points	3.1.6.2
Time-Out Alarm, Timer Points	3.5.2
Timer Points	3.5
Transport Errors, Data – CL	4.1.4.5
Trip Points, Alarm	2.3.2
Unit Alarm Operations	2.3.12
Value Status, PV – Regulatory Points	3.1.5.6
Value Storage Failure	2.1.3.3
Value Storage with an Error	2.1.3.4
Warm Restart	2.7.3
Warning and Error Messages	
Counter Points	3.3.13
Numeric Points	3.2.6
Timer Points	3.5.6
Windup Protection, Regulatory Points	3.1.11.1

Fax Transmittal		Fax No.: (602) 313-4212	
Reader Comments			
To: Information Development			
From:	Name		Date:
	Title:		
	Company:		
	Address:		
	City:	State:	Zip:
	Telephone:	Fax:	
Application Module Control Functions, AM09-602			
Release UIS R620, 12/00			
Comments:			
<div></div>			
<div></div>			
<div></div>			
<div></div>			
<div></div>			
<div></div>			
<div></div>			
<div></div>			
<div></div>			
<div></div>			
<div></div>			
You may also call 800-343-0228 (available in the 48 contiguous states), or write to:			
<div></div>			
Honeywell		Knowledge Building Tools	
Industrial Automation and Control			
16404 North Black Canyon Highway			
Phoenix, AZ 85053-3033		TotalPlant	

**Honeywell**

---

Honeywell  
Industrial Automation and Control  
16404 North Black Canyon Highway  
Phoenix, AZ 85053