Mark* VIe Controller
Legacy Block Library

These instructions do not purport to cover all details or variations in equipment, nor to provide for every possible contingency to be met during installation, operation, and maintenance. The information is supplied for informational purposes only, and GE makes no warranty as to the accuracy of the information included herein. Changes, modifications, and/or improvements to equipment and specifications are made periodically and these changes may or may not be reflected herein. It is understood that GE may make changes, modifications, or improvements to the equipment referenced herein or to the document itself at any time. This document is intended for trained personnel familiar with the GE products referenced herein.

Public Information – This document contains non-sensitive information approved for public disclosure.

GE may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not provide any license whatsoever to any of these patents.

GE provides the following document and the information included therein as is and without warranty of any kind, expressed or implied, including but not limited to any implied statutory warranty of merchantability or fitness for particular purpose.

For further assistance or technical information, contact the nearest GE Sales or Service Office, or an authorized GE Sales Representative.

Revised: Nov 2020
Issued: June 2006


* Indicates a trademark of General Electric Company and/or its subsidiaries. All other trademarks are the property of their respective owners.

We would appreciate your feedback about our documentation. Please send comments or suggestions to controls.doc@ge.com
## Document Updates

<table>
<thead>
<tr>
<th>Revision</th>
<th>Update</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Throughout document</td>
<td>Fixed formatting error</td>
</tr>
<tr>
<td>M</td>
<td><em>Combine SLR (COMBINE_SLR)</em></td>
<td>New blocks</td>
</tr>
<tr>
<td>M</td>
<td><em>Combine SSD (COMBINE_SSD)</em></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td><em>Split LRS (SPLIT_LRS)</em></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td><em>Split SDS (SPLIT_SDS)</em></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td><em>Integer to Word Conversion (I_TO_WD)</em></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td><em>Word to Integer Conversion (WD_TO_J)</em></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td><em>Combine SD (COMBINE_SD)</em></td>
<td>Changed Legacy status and replaced content</td>
</tr>
<tr>
<td>M</td>
<td><em>Combine SR (COMBINE_SR)</em></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td><em>Split DS (SPLIT_DS)</em></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td><em>Split RS (SPLIT_RS)</em></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>Ramp (<em>RAMP</em>®)</td>
<td>Updates for clarification on configuration settings</td>
</tr>
<tr>
<td>K</td>
<td><em>Replacement Blocks</em></td>
<td>Changed the OUTXFER block Reason for No Replacement column content to reference the MOVE block with similar functionality</td>
</tr>
<tr>
<td>J</td>
<td>DALIP00</td>
<td>Updated with a better description of the <em>FTBL</em> input array</td>
</tr>
<tr>
<td>H</td>
<td><em>Replacement Blocks</em></td>
<td>Updated Legacy Status to explain that these blocks are supported but not recommended for use in any new controller applications. They are legacy because no new development is to be done to add any new functionality to these blocks. In many cases there is a newer block that is preferred for use as a replacement for these blocks, as listed in the table. Updated this table to explain why there are or are not replacement blocks.</td>
</tr>
</tbody>
</table>
Introduction

Blocks are software subroutines that are included in the tasks to be run in the controller. Within the ToolboxST* application, the connections are accomplished graphically using pins and wires to form a data flow diagram. The blocks run in the order listed in the task. The connections between the blocks are RAM memory elements through which data is passed. These elements are referred to as variables. Each variable has a defined data type and must be connected to pins of the same data type. Some block pins accept multiple data types and others are capable of converting from one data type to another.

Each block entry in the library consists of the following:

- Block name and category
- Block description
- Block diagram
- Pin definitions
- Application notes (optional)

Each block has a full name and a simple name. The full name is descriptive and too long to display in a block diagram. For this reason, all blocks have a simple name. The simple name is displayed in the ToolboxST application.

Floating point (REAL and LREAL) values conform to IEEE® 754. The standard defines certain computational exceptions that may produce non-standard results, such as divide by zero. These non-standard values include infinities and Not a Numbers (NaNs), which, if used in subsequent calculations, produce additional non-standard results. The software has been designed to prevent the production and proliferation of such values. Function block descriptions may include additional details concerning floating point exception handling where needed.

Block Libraries also contain rubber blocks, which are sized according to the number of pins used. Within the library, the blocks are grouped functionally to simplify the task of converting ideas into blocks. For example, the Controller Monitor (CTRLR_MON) block is a member of the System category.

**Note** Pin names are given modifiers that provide additional information about the pin.

A graphical representation of the software for each block is provided in the form of a block diagram that includes the simple block name and block configuration pins and variable pins, which transfer data in and out of the block. The pins are located on the block diagram according to their usage. All input pins (pins read by the block) and block configuration pins are listed on the left. All output pins (pins written by the block) and state pins (pins read and written by the block) are listed on the right. The pin name is listed beside each pin inside the block diagram border.

<table>
<thead>
<tr>
<th>Pin Name Modifiers</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>^ prefixed with a carat (^ONREQ)</td>
<td>Pin is rising-edge sensitive</td>
</tr>
<tr>
<td>[ ] suffixed with square brackets (STATE[4])</td>
<td>Pin expects a variable that is an array of the size indicated within the brackets</td>
</tr>
</tbody>
</table>
**Block Data Type**

The pin data type identifier is listed beside the pin outside of the block diagram border. The basic data types used in the block libraries are BOOL, INT, UINT, REAL, LREAL, DINT and UDINT. Blocks with data types ANY_NUM and ANY must have the Block Data Type set to match the intended data type of the block output.

**Note** The default value of Boolean inputs is *False* unless the input is a block Enable, in which case the default is *True* and the default value of inputs with data types INT, UINT, DINT, UDINT, REAL and LREAL is 0, until otherwise specified.

The block content provides tables to define the input and output pins. Each entry includes the block name for the variable, basic data type, typical scaling units, and a description of the variable. Typical scaling units are supplied for reference only; specific applications may use other scaling units as appropriate.

<table>
<thead>
<tr>
<th>Data Type Identifier</th>
<th>Basic Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Any (ANY)</td>
<td>Any data type</td>
</tr>
<tr>
<td>B</td>
<td>Boolean (BOOL)</td>
<td>8-bit Boolean</td>
</tr>
<tr>
<td>I</td>
<td>Integer (INT)</td>
<td>16-bit (Short) Signed Integer</td>
</tr>
<tr>
<td>DI</td>
<td>Double Integer (DINT)</td>
<td>32-bit (Long) Signed Integer</td>
</tr>
<tr>
<td>R</td>
<td>Real (REAL)</td>
<td>32-bit Floating Point</td>
</tr>
<tr>
<td>LR</td>
<td>Long Real (LREAL)</td>
<td>64-bit (Long) Floating Point</td>
</tr>
<tr>
<td>UI</td>
<td>Unsigned Integer (UINT)</td>
<td>16-bit (Short) Unsigned Integer</td>
</tr>
<tr>
<td>UDI</td>
<td>Unsigned Double Integer (UDINT)</td>
<td>32-bit (Long) Unsigned Integer</td>
</tr>
<tr>
<td>N</td>
<td>Numeric (ANY_NUM)</td>
<td>Any data type except Boolean</td>
</tr>
<tr>
<td>S</td>
<td>Analog or Boolean (Simple)</td>
<td>Any simple data type</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data Type Identifier</th>
<th>Type Definition Modifier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>m</td>
<td>Build time constant</td>
</tr>
</tbody>
</table>

**Note** The Block Data Type is set in the block Property Editor.
Changing Variant Block Data Types

There are few blocks that support more than one data type as an input and/or output.

➢ To change the block data type

1. Select the block for which to change the data type.
2. From the Block Properties window, expand the General section.
3. From the Property Editor, select the Block Data Type.
4. From the Block Data Type drop-down box, select a valid data type for the block.
**Replacement Blocks**

These blocks are supported but not recommended for use in any new controller applications (ControlST V05.02 or later). They are legacy because no new development is to be done to add any new functionality to these blocks. In many cases there is a newer block that is preferred for use as a replacement for these blocks, as listed in the following table.

### Legacy Block Rationale

<table>
<thead>
<tr>
<th>Legacy Block</th>
<th>Replacement Block</th>
<th>Description</th>
<th>Reason for No Replacement</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANA_TO_WD</td>
<td>None</td>
<td>Real, Double, Long, etc. to 16-bit integer</td>
<td>This block was intended to process special I/O types. I/O packs perform this function if needed.</td>
</tr>
<tr>
<td>BFILT</td>
<td>ON_OFF_DELAY</td>
<td>Boolean Filter, On and Off Delays</td>
<td></td>
</tr>
<tr>
<td>BIT_CNTR_UDI</td>
<td>BIT_CNTR</td>
<td>Counts the number of input booleans that are True, Outputs unsigned double integer</td>
<td></td>
</tr>
<tr>
<td>CNT_TO_BOOL</td>
<td>None</td>
<td>Unsigned integer input decoded to one of 32 booleans</td>
<td>This block was intended to process special I/O types. I/O packs perform this function if needed.</td>
</tr>
<tr>
<td>COMPHYS</td>
<td>COMPARE</td>
<td>Compare with hysteresis</td>
<td></td>
</tr>
<tr>
<td>COUNTER</td>
<td>None</td>
<td>Count UP to a maximum and set at count</td>
<td>The CTU (Count Up) and CTD (Count Down) blocks perform this function but can only preset to 0.</td>
</tr>
<tr>
<td>DALIP00</td>
<td>None</td>
<td>Double analog linear interpolation</td>
<td>INTERP_DL performs a similar function.</td>
</tr>
<tr>
<td>DPYSTAT1_UDI</td>
<td>DPYSTAT1</td>
<td>Display State Generator, input # of 1st True boolean is unsigned double integer out</td>
<td></td>
</tr>
<tr>
<td>EX_MAC</td>
<td>None</td>
<td>Exit macro</td>
<td>It exists (for upgrades only) in the Mark VIe controller. The block cannot be added from the ToolboxST application after V03.06.09C. The functionality is provided by Enable pins.</td>
</tr>
<tr>
<td>EX_TSK</td>
<td>None</td>
<td>Exit task</td>
<td></td>
</tr>
<tr>
<td>IN_CVRT</td>
<td>None</td>
<td>Convert coded (BCD,GRAY,BINARY) decimal booleans to a long integer</td>
<td>This block was intended to process special I/O types. I/O packs perform this function if needed.</td>
</tr>
<tr>
<td>MKVIE_MON</td>
<td>CTRLR_MON</td>
<td>Mark VIe Monitor, power_up, sig_forced, overtemp. More in Controller Monitor.</td>
<td></td>
</tr>
<tr>
<td>OUT_CVRT</td>
<td>None</td>
<td>Convert coded (BCD,GRAY,BINARY) long integer into booleans</td>
<td>This block was intended to process special I/O types. I/O packs perform this function if needed.</td>
</tr>
<tr>
<td>OUTXFER</td>
<td>None</td>
<td>Moves a value into an input point of this controller for simulation</td>
<td>MOVE block performs a similar function.</td>
</tr>
<tr>
<td>OVR_ST</td>
<td>OVR_ST_ENH_V2</td>
<td>Override Station for DCS/BOP PID controllers. Enhanced controller up to 3.</td>
<td></td>
</tr>
<tr>
<td>PE_TOGGLE</td>
<td>None</td>
<td>Permissive Engine toggle action</td>
<td>Used only for Exciter upgrade efficiency. Not planned for future.</td>
</tr>
<tr>
<td>Legacy Block</td>
<td>Replacement Block</td>
<td>Description</td>
<td>Reason for No Replacement</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------</td>
<td>-------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>PID_MA</td>
<td>PID_MA_ENH_V2</td>
<td>Operator station for PID control for DCS/BOP</td>
<td></td>
</tr>
<tr>
<td>PRESET_CNTR</td>
<td>None</td>
<td>Counts input pulses with ability to preset the current count</td>
<td>The CTU (Count Up) and CTD (Count Down) blocks perform this function but can only preset to 0.</td>
</tr>
<tr>
<td>RAMP</td>
<td>FUNGEN</td>
<td>Ramp Up/Down to a final value</td>
<td></td>
</tr>
<tr>
<td>RAMPH</td>
<td>FUNGEN</td>
<td>Ramp Up/Down to a final value with Hold feature</td>
<td></td>
</tr>
<tr>
<td>TIMER_UI</td>
<td>TIMER</td>
<td>Timer unsigned short</td>
<td></td>
</tr>
<tr>
<td>WD_TO_ANA</td>
<td>None</td>
<td>Converts up to 32 unsigned integer inputs to analog type outputs</td>
<td>This block was intended to process special I/O types. I/O packs perform this function if needed.</td>
</tr>
</tbody>
</table>
Analog to Word Conversion (ANA_TO_WD)

Block Category: ANA_TO_WD, Type Conversion

Legacy Status — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information.

The Analog to Word Conversion (ANA_TO_WD) block is a rubber block that transfers the values on 1 to 32 input pins to corresponding output pins. The inputs are analog data types and the outputs are unsigned integer types. Long Real, Real, Double Integer, Integer input values are clamped to 65535 and 0 (if negative). Integer and unsigned integer inputs are transferred without modification.

### ANA_TO_WD Block

#### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>Boolean</td>
<td>Enables block execution</td>
</tr>
<tr>
<td>IN1</td>
<td>Numeric</td>
<td>Analog input value1</td>
</tr>
<tr>
<td>↓</td>
<td>↓</td>
<td></td>
</tr>
<tr>
<td>IN32</td>
<td>Numeric</td>
<td>Analog input value2</td>
</tr>
</tbody>
</table>

#### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT1</td>
<td>Unsigned integer</td>
<td>Converted output variable</td>
</tr>
<tr>
<td>↓</td>
<td>↓</td>
<td></td>
</tr>
<tr>
<td>OUT32</td>
<td>Unsigned integer</td>
<td>Converted output variable</td>
</tr>
</tbody>
</table>
**Bit Counter (BIT_CNTR_UDI)**

**Block Category:** BIT_CNTR_UDI, **Legacy** BIT_CNTR

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section *Replacement Blocks* for additional information.

The Bit Counter (BIT_CNTR_UDI) block is a *rubber block* that counts the number of input bits that are currently set to *True*. The block can handle from 1 to 32 input Boolean signals. The resulting count is in placed in the Unsigned Double Integer `COUNT`.

### BIT_CNTR_UDI Block

**Inputs**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN1</td>
<td>Boolean</td>
<td>Boolean signal 1</td>
</tr>
<tr>
<td>IN32</td>
<td>Boolean</td>
<td>Boolean signal 32</td>
</tr>
</tbody>
</table>

**Output**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT</td>
<td>Unsigned Double Integer</td>
<td>Number of Boolean bits set to <em>True</em></td>
</tr>
</tbody>
</table>
**Boolean Filter (BFILT)**

**Block Category:** BFILT, Boolean Operations

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section *Replacement Blocks* for additional information.

The Boolean Filter (BFILT) block passes a rising True value on the IN pin to the OUT pin if IN remains True for PU_DEL time (in milliseconds). The block passes a falling False value on the IN pin to the OUT pin if IN remains False for DO_DEL time (in milliseconds).

![BFILT Block Diagram](image)

### BFILT Block

**Inputs**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PU_DEL</td>
<td>Unsigned Double Integer</td>
<td>Pick-up delay in milliseconds</td>
</tr>
<tr>
<td>IN</td>
<td>Boolean</td>
<td>Identifies the variable to filter</td>
</tr>
<tr>
<td>DO_DEL</td>
<td>Unsigned Double Integer</td>
<td>Drop-out delay in milliseconds</td>
</tr>
</tbody>
</table>

**Output**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT</td>
<td>Boolean</td>
<td>The filtered output</td>
</tr>
</tbody>
</table>

**State**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREVOUT</td>
<td>Boolean</td>
<td>This is the last filtered output, a hidden pin. It can be viewed or changed by right-clicking the block and selecting <em>Edit Block Pins</em>. Writing to this variable is not recommended, but can be used to preset block states.</td>
</tr>
</tbody>
</table>
**Combine SD (COMBINE_SD)**

**Block Category:** Type Conversion

**Legacy Status** — Not a legacy block.

The Combine blocks concatenate smaller variables into larger variables. They perform the reverse function of the Split blocks. The COMBINE_SD block is used to combine two 16-bit WORD values to a single unsigned double integer (32-bit).

The DATA_SWAP input on the block is used to decide whether to do a byte swap on the input WORD values while calculating the output.

The reverse of the COMBINE_SD block is the SPLIT_DS block.

### COMBINE_SD Block

#### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>BOOL</td>
<td>Block enable (default is True)</td>
</tr>
<tr>
<td>LOWORD</td>
<td>UINT</td>
<td>Low word input</td>
</tr>
<tr>
<td>HIWORD</td>
<td>UINT</td>
<td>High word input</td>
</tr>
<tr>
<td>DATA_SWAP</td>
<td>DataSwap_T</td>
<td>Byte swap input for the individual outputs: NO_SWAP (0) or BYTE_SWAP (1) (default is NO_SWAP)</td>
</tr>
</tbody>
</table>

#### Output

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>UDINT</td>
<td>The combined unsigned double integer output</td>
</tr>
</tbody>
</table>
**Combine SLR (COMBINE_SLR)**

**Block Category:** Type Conversion

**Legacy Status** — Not a legacy block.

The Combine blocks concatenate smaller variables into larger variables. They perform the reverse function of the Split blocks. The COMBINE_SLR block is used to combine four 16-bit WORD values to a single long real value (64-bit). This block performs NaN protection by setting OUTPUT to the respective minimum or maximum value if the combined input produces a NaN.

The DATA_SWAP input on the block is used to decide whether to do a byte swap on the input WORD values while calculating the output.

The reverse of the COMBINE_SLR block is the SPLIT_LRS block.

---

**COMBINE_SLR Block**

**Inputs**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>BOOL</td>
<td>Block enable (default is True)</td>
</tr>
<tr>
<td>WORD0</td>
<td>UINT</td>
<td>Input to represent bits 0-15 of OUTPUT</td>
</tr>
<tr>
<td>WORD1</td>
<td>UINT</td>
<td>Input to represent bits 16-31 of OUTPUT</td>
</tr>
<tr>
<td>WORD2</td>
<td>UINT</td>
<td>Input to represent bits 32-47 of OUTPUT</td>
</tr>
<tr>
<td>WORD3</td>
<td>UINT</td>
<td>Input to represent bits 48-63 of OUTPUT</td>
</tr>
<tr>
<td>DATA_SWAP</td>
<td>DataSwap_T</td>
<td>Byte swap input for the individual outputs: NO_SWAP (0) or BYTE_SWAP (1) (default is NO_SWAP)</td>
</tr>
</tbody>
</table>

**Output**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>LREAL</td>
<td>The combined long real output</td>
</tr>
</tbody>
</table>
Combine SR (COMBINE_SR)

Block Category: Type Conversion

Legacy Status — Not a legacy block.

The Combine blocks concatenate smaller variables into larger variables. They perform the reverse function of the Split blocks. The COMBINE_SR block is used to combine two 16-bit WORD values to a single real value (32-bit). This block performs NaN protection by setting OUTPUT to the respective minimum or maximum value if the combined input produces a NaN.

The DATA_SWAP input on the block is used to decide whether to do a byte swap on the input WORD values while calculating the output.

The reverse of the COMBINE_SR block is the SPLIT_RS block.

![COMBINE_SR Block](image)

**Inputs**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>BOOL</td>
<td>Block enable (default is True)</td>
</tr>
<tr>
<td>LOWORD</td>
<td>UINT</td>
<td>Low word input</td>
</tr>
<tr>
<td>HIWORD</td>
<td>UINT</td>
<td>High word input</td>
</tr>
<tr>
<td>DATA_SWAP</td>
<td>DataSwap_T</td>
<td>Byte swap input for the individual outputs: NO_SWAP (0) or BYTE_SWAP (1)</td>
</tr>
</tbody>
</table>
  
  (default is NO_SWAP)

**Output**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>REAL</td>
<td>The combined real output</td>
</tr>
</tbody>
</table>
**Combine SSD (COMBINE_SSD)**

**Block Category: Type Conversion**

**Legacy Status** — Not a legacy block.

The Combine blocks concatenate smaller variables into larger variables. They perform the reverse function of the Split blocks. The COMBINE_SSD block is used to combine two 16-bit WORD values to a signed double integer (32-bit).

The DATA_SWAP input on the block is used to decide whether to do a byte swap on the input WORD values while calculating the output.

The reverse of the COMBINE_SSD block is the SPLIT_SSD block.

---

**COMBINE_SSD Block**

### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>BOOL</td>
<td>Block enable (default is True)</td>
</tr>
<tr>
<td>LOWORD</td>
<td>UINT</td>
<td>Low word input</td>
</tr>
<tr>
<td>HIWORD</td>
<td>UINT</td>
<td>High word input</td>
</tr>
<tr>
<td>DATA_SWAP</td>
<td>DataSwap_T</td>
<td>Byte swap input for the individual outputs: NO_SWAP (0) or BYTE_SWAP (1) (default is NO_SWAP)</td>
</tr>
</tbody>
</table>

### Output

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>DINT</td>
<td>The combined signed long integer output</td>
</tr>
</tbody>
</table>
**Compare with Hysteresis (COMPHYS)**

**Block Category:** COMPHYS, Comparison

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section *Replacement Blocks* for additional information.

The Compare with Hysteresis (COMPHYS) block compares two floating point values and sets an output Boolean reflecting the result. Hysteresis. Sensitivity settings are provided to prevent toggling around a boundary condition. The supported functions include: greater than (gt), greater than or equal to (ge), equal to (eq), less than (lt), or less than or equal to (le).

Sensitivity refers to the thickness of the equality relationship. That is, IN1 is equal to IN2 if it is numerically within the band defined by IN2 ± SENS. Hysteresis applies for all relationships. Once any relationship evaluates to True it must exceed the normal specification by the HYST amount before it evaluates to False.

**Floating Point Exception Handling**

If a NaN is present on an input pin, the COMPHYS block replaces it internally with the most positive or most negative representable number based on its sign, for the purpose of the comparison.

*Note* This block is a variant block that supports the following block data types: Real and Long Real. To modify the data type, refer to the section, *Changing Variant Block Data Types.*

---

**COMPHYS Block**

**Inputs**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN1</td>
<td>REAL, LREAL</td>
<td>The input to compare</td>
</tr>
<tr>
<td>FUNC</td>
<td>Constant CompFnL_T</td>
<td>The comparison function (gt, lt, ge, le, eq). The default value is gt.</td>
</tr>
<tr>
<td>IN2</td>
<td>REAL, LREAL</td>
<td>The value to compare against</td>
</tr>
<tr>
<td>HYST</td>
<td>REAL, LREAL</td>
<td>The hysteresis value</td>
</tr>
<tr>
<td>SENS</td>
<td>REAL, LREAL</td>
<td>The sensitivity value</td>
</tr>
</tbody>
</table>

**Output**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT</td>
<td>BOOL</td>
<td>The result of the comparison</td>
</tr>
</tbody>
</table>
## Count to Boolean (CNT_TO_BOOL)

**Block Category:** CNT_TO_BOOL, Boolean Operations

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section *Replacement Blocks* for additional information.

The Count to Boolean (CNT_TO_BOOL) block uses the value of an integer input, IN, to select one of N Boolean outputs. It is a *rubber* that provides up to 32 outputs. The selected output is driven *True* while all others are driven *False*. An input value of 0 for IN selects the first output and an input value of N-1 selects the last output, OUT (N). If the integer input contains a value that is not within the range of 0 to N-1 then all outputs are driven *False*.

![CNT_TO_BOOL Diagram](attachment:image.png)

### CNT_TO_BOOL Block

#### Input

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN</td>
<td>Unsigned Integer</td>
<td>Selects a particular output Boolean</td>
</tr>
</tbody>
</table>

#### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT1</td>
<td>Boolean</td>
<td>Contains the results of the selection process</td>
</tr>
<tr>
<td>↓</td>
<td>↓</td>
<td>↓</td>
</tr>
<tr>
<td>OUT32</td>
<td>Boolean</td>
<td>Contains the results of the selection process</td>
</tr>
</tbody>
</table>
Counter (COUNTER)

Block Category: COUNTER, Timers and Counters

Legacy Status — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information.

The Counter (COUNTER) block accumulates rising edges of the INC pin into CUR_CNT while the RUN is True and CUR_CNT is less than MAX_CNT. When CUR_CNT is equal to MAX_CNT, AT_CNT transitions to True. If RUN is False then counting is suspended, but CUR_CNT holds its value. If RESET is True then CUR_CNT is set to zero and counting is suspended. If MAX_CNT is less than one, each rising edge of INC sets AT_CNT True and increments CUR_CNT.

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INC</td>
<td>Boolean</td>
<td>Triggers another count on its rising edge</td>
</tr>
<tr>
<td>MAX_CNT</td>
<td>Double Integer</td>
<td>The maximum count value</td>
</tr>
<tr>
<td>RESET</td>
<td>Boolean</td>
<td>Zeros the counter</td>
</tr>
<tr>
<td>RUN</td>
<td>Boolean</td>
<td>Enables the counter</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT_CNT</td>
<td>Boolean</td>
<td>Indicates the counter has reached the maximum count value</td>
</tr>
<tr>
<td>CUR_CNT</td>
<td>Double Integer</td>
<td>The current count value (always ≤ MAX_CNT)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAST_CURCNT†</td>
<td>Double Integer</td>
<td>This is the variable that holds the current count. It is a hidden pin that can be viewed or changed by right-clicking the block and selecting Edit Block Pins. Writing to this variable is not recommended, but can be used to preset block states.</td>
</tr>
<tr>
<td>LAST_ATCNT</td>
<td>Boolean</td>
<td>This is the variable that holds the state, indicating the counter has reached the maximum count value. This is a hidden pin. It can be viewed or changed by right-clicking the block and selecting Edit Block Pins. Writing to this variable is not recommended, but can be used to preset block states.</td>
</tr>
</tbody>
</table>

† Connect the LAST_CURCNT state variable to a NOVRAM variable to retain the counter value through a reboot.
Display State Generator for Unsigned Long (DPYSTAT1_UDI)

Block Category: DPYSTAT1_UDI, Legacy

Legacy Status — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information.

The Display State Generator for Unsigned Long (DPYSTAT1_UDI) block is a rubber block that generates an enumerated state variable by interrogating a list of logic variables and setting the enumerated state value to the number of the first True logic entry. It supports up to 32 inputs.

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT1</td>
<td>Boolean</td>
<td>Input Logical 1</td>
</tr>
<tr>
<td>INPUT32</td>
<td>Boolean</td>
<td>Input Logical 32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>Unsigned Double Integer</td>
<td>Integer Output Value</td>
</tr>
</tbody>
</table>
Double Analog Linear Interpolation (DALIP00)

Block Category: DALIP00, Legacy

Legacy Status — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information.

The Double Analog Linear Interpolation (DALIP00) block estimates a missing functional value \( f(x,y) \) from four known functional values at neighboring points \( f(X0,Y0), f(X0,Y1), f(X1,Y0), f(X1, Y1) \), provided \( X0 < x < X1 \) and \( Y0 < Y1 \) as follows. Expansion checks are done to ensure that the declared size of the function array \( f \) is equal to the size of the input array \( x \) times the size of the input array \( y \).

First, \( f(x, Y0) \) is linearly interpolated between \( f(X0, Y0) \) and \( f(X1, Y0) \):

\[
f(x, Y0) = f(X0, Y0) + \frac{x - X0}{X1 - X0} \times [f(X1, Y0) - f(X0, Y0)]
\]

Second, \( f(x, Y1) \) is linearly interpolated between \( f(X0, Y1) \) and \( f(X1, Y1) \):

\[
f(x, Y1) = f(X0, Y1) + \frac{x - X0}{X1 - X0} \times [f(X1, Y1) - f(X0, Y1)]
\]

Finally, the linear interpolation between \( f(x, Y0) \) and \( f(x, Y1) \) gives \( f(x,y) \).

\[
f(x, y) - f(x, Y0) = f(x, Y1) - f(x, Y0) + \frac{y - Y0}{Y1 - Y0} \times [f(x, Y1) - f(x, Y0)]
\]

\[
= f(X0, Y0) + \left[\frac{x - X0}{X1 - X0} \times [f(X1, Y0) - f(X0, Y0)] + \left[\frac{y - Y0}{Y1 - Y0} \times \frac{x - X0}{X1 - X0} \times [f(X1, Y1) - f(X0, Y1)] - f(X0, Y0)\right]\right] + \frac{y - Y0}{Y1 - Y0} \times \frac{x - X0}{X1 - X0} \times [f(X1, Y1) - f(X0, Y1) - f(X1, Y0) + f(X0, Y0)]
\]
### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Real</td>
<td>Input X</td>
</tr>
<tr>
<td>Y</td>
<td>Real</td>
<td>Input Y</td>
</tr>
<tr>
<td>XTBL</td>
<td>Real</td>
<td>Beginning address of X table</td>
</tr>
<tr>
<td>YTBL</td>
<td>Real</td>
<td>Beginning address of Y table</td>
</tr>
</tbody>
</table>
| FTBL | Real      | Beginning address of F table  
  \[FTBL[0] = f(XTBL[0], YTBL[0]), \quad FTBL[1] = f(XTBL[0], YTBL[1]), \quad FTBL[2] = f(XTBL[0], YTBL[2]), \ldots, \quad FTBL[nYTBL - 1] = f(XTBL[0], YTBL[nYTBL - 1]), \quad FTBL[nYTBL] = f(XTBL[1], YTBL[0]), \ldots,\]  
  where \(nYTBL = \) the number of elements in YTBL. |
| FMIN | Real      | Minimum function table value |
| FMAX | Real      | Maximum function table value |

### Output

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Real</td>
<td>Function output</td>
</tr>
</tbody>
</table>

### Exit Macro (EX_MAC)

**Block Category:** EX_MAC, Program Control

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section **Replacement Blocks** for additional information. The block cannot be added from the ToolboxST application after V03.06.09C. The functionality is provided by Enable pins.

When the Exit Macro (EX_MAC) block is enabled, the user block containing this block will exit immediately, skipping execution of rest of the blocks. This can be used as a break statement in a user block.

![EX_MAC Block Diagram](image)

**EX_MAC Block**

**Input**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>Boolean</td>
<td>Enable the block to exit.</td>
</tr>
</tbody>
</table>
Exit Task (EX_TSK)

Block Category: EX_TSK, Program Control

Legacy Status — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information. The block cannot be added from the ToolboxST application after V03.06.09C. The functionality is provided by Enable pins.

When the Exit Task (EX_TSK) block is enabled, the task containing this block will exit immediately, skipping execution of the remaining blocks. This can be used as a break statement in a task.

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>Boolean</td>
<td>Enable the block to exit.</td>
</tr>
</tbody>
</table>

![EX_TSK Block Diagram]
Input Conversion (IN_CVRT)

Block Category: IN_CVRT, Type Conversion

Legacy Status — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information.

The Input Conversion (IN_CVRT) block converts Boolean inputs into a long integer based on a conversion function. This rubber block supports up to 32 inputs and may be specified in groups of four. If the inputs contain an invalid value, the error pin will be set to True and the last valid conversion will remain on the output pin. Binary Coded Decimal (BCD), BINARY, and GRAY coded Booleans are decoded with this block.
### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>Boolean</td>
<td>Enable block execution</td>
</tr>
<tr>
<td>FCN</td>
<td>Constant InOutCvrt_T</td>
<td>Conversion to perform. The default value is BCD.</td>
</tr>
<tr>
<td>SIGN</td>
<td>Boolean</td>
<td>Sign - The sign of the Coded value (<em>False</em> = positive, <em>True</em> = negative). The default value is <em>False</em>.</td>
</tr>
<tr>
<td>IN11</td>
<td>Boolean</td>
<td>The first (lsb) input Boolean in the first set of four Booleans</td>
</tr>
<tr>
<td>IN12</td>
<td>Boolean</td>
<td>The second input Boolean in the first set of four Booleans</td>
</tr>
<tr>
<td>IN13</td>
<td>Boolean</td>
<td>The third input Boolean in the first set of four Booleans</td>
</tr>
<tr>
<td>IN14</td>
<td>Boolean</td>
<td>The fourth (msb) input Boolean in the first set of four Booleans</td>
</tr>
<tr>
<td>↓</td>
<td>↓</td>
<td>↓</td>
</tr>
<tr>
<td>IN81</td>
<td>Boolean</td>
<td>The first (lsb) input Boolean in the eighth set of four Booleans</td>
</tr>
<tr>
<td>IN82</td>
<td>Boolean</td>
<td>The second input Boolean in the eighth set of four Booleans</td>
</tr>
<tr>
<td>IN83</td>
<td>Boolean</td>
<td>The third input Boolean in the eighth set of four Booleans</td>
</tr>
<tr>
<td>IN84</td>
<td>Boolean</td>
<td>The fourth (msb) input Boolean in the eighth set of four Booleans</td>
</tr>
</tbody>
</table>

### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR</td>
<td>Boolean</td>
<td><em>True</em> if an invalid coding is detected or if unknown conversion is output</td>
</tr>
<tr>
<td>OUT</td>
<td>Double Integer</td>
<td>The numerical value of the coded Booleans</td>
</tr>
</tbody>
</table>
**Integer to Word Conversion (I_TO_WD)**

**Block Category:** Type Conversion

**Legacy Status** — Not a legacy block.

The Integer to Word Conversion (I_TO_WD) block moves a signed integer (16-bit) to a 16-bit WORD. The conversion is strictly mechanical. That is, there is no attempt to interpret the data, or to perform any clamping, scaling, data type conversion, or sign extension.

### I_TO_WD Block

#### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>BOOL</td>
<td>Block enable (default is True)</td>
</tr>
<tr>
<td>INPUT</td>
<td>INT</td>
<td>Integer input</td>
</tr>
</tbody>
</table>

#### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>UINT</td>
<td>Unsigned integer (16-bit WORD) output</td>
</tr>
</tbody>
</table>
Mark VIe Monitor (MKVIE_MON)

Block Category: MKVIE_MON, System

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information.

Only one MKVIE_MON block can be used per controller.

The Mark VIe Monitor (MKVIE_MON) block is used for various purposes. The block has a POWER_UP output that transitions to True 5 sec after the controller is in control mode. SIG_FORCED turns on if there are any forced variables in the controller, SIG_FORCED is set to False for 2 sec every time the number of forced variables are increased. SIG_FORCED_NUM indicates the number of forced variables in the controller.

*Note* In the Virtual Controller, the OVER TEMP_R, OVER TEMP_S and OVER TEMP_T are all False because there is no temperature measurement functionality.

The block detects the current controller configuration (Simplex, Dual, or TMR). According to this configuration, over temperature alarms OVER TEMP_R, OVER TEMP_S, OVER TEMP_T and controller online variables ONLINE_R, ONLINE_S, and ONLINE_T are generated for R, S and T respectively. The input ONLINE_FL_T is used to delay the controller online variables change from True to False in the event the controller heartbeat stops. MKVIE_MON creates global variables that can be used throughout the controller application software.

### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Type</th>
<th>Initial Value</th>
<th>Visibility</th>
<th>Usage</th>
<th>Interface Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ONLINE_FL_T</td>
<td>Online Fail Time</td>
<td>UDINT</td>
<td>5000</td>
<td>Parameter</td>
<td>Input</td>
<td>Value only</td>
</tr>
</tbody>
</table>

### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Type</th>
<th>Initial Value</th>
<th>Visibility</th>
<th>Usage</th>
<th>Interface Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>POWER_UP</td>
<td>Controller Powered Up</td>
<td>BOOL</td>
<td>0</td>
<td>Always</td>
<td>Output</td>
<td>Value Only</td>
</tr>
<tr>
<td>SIG_FORCED</td>
<td>Signal Forced In Controller</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Output</td>
<td>Value Only</td>
</tr>
<tr>
<td>SIG_FORCED_NUM</td>
<td>Number of Forced Signals</td>
<td>INT</td>
<td>0</td>
<td>Always</td>
<td>Output</td>
<td>Value Only</td>
</tr>
<tr>
<td>OVERTEMP_R</td>
<td>R Controller Over Temperature</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Output</td>
<td>Value Only</td>
</tr>
<tr>
<td>OVERTEMP_S†</td>
<td>S Controller Over Temperature</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Output</td>
<td>Value Only</td>
</tr>
<tr>
<td>OVERTEMP_T††</td>
<td>T Controller Over Temperature</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Output</td>
<td>Value Only</td>
</tr>
</tbody>
</table>

† Exists only when the controller’s redundancy is DUAL or TMR.
†† Exists only when the controller’s redundancy is TMR.

### Global Variables

<table>
<thead>
<tr>
<th>Global Pin Name</th>
<th>Description</th>
<th>Alarm</th>
<th>Alarm Class</th>
<th>Event</th>
<th>EGD Page</th>
<th>External Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>POWER_UP</td>
<td>Controller Powered Up</td>
<td>Not Alarmed</td>
<td></td>
<td>False</td>
<td></td>
<td>Read Only</td>
</tr>
<tr>
<td>SIG_FORCED</td>
<td>Signal Forced In Controller</td>
<td>Alarmed</td>
<td>Diag</td>
<td>False</td>
<td></td>
<td>Read Only</td>
</tr>
</tbody>
</table>
### Global Variables (continued)

<table>
<thead>
<tr>
<th>Global Pin Name</th>
<th>Description</th>
<th>Alarm</th>
<th>Event</th>
<th>EGD Page</th>
<th>External Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIG_FORCED_NUM</td>
<td>Number of Forced Signals</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>Read Only</td>
</tr>
<tr>
<td>OVERTEMP_R</td>
<td>R Controller Over Temperature</td>
<td>Alarmed</td>
<td>Diag</td>
<td>$Default</td>
<td>Read Only</td>
</tr>
<tr>
<td>OVERTEMP_S†</td>
<td>S Controller Over Temperature</td>
<td>Alarmed</td>
<td>Diag</td>
<td>$Default</td>
<td>Read Only</td>
</tr>
<tr>
<td>OVERTEMP_T ††</td>
<td>T Controller Over Temperature</td>
<td>Alarmed</td>
<td>Diag</td>
<td>$Default</td>
<td>Read Only</td>
</tr>
</tbody>
</table>

† Exists only when the controller’s redundancy is DUAL or TMR.

†† Exists only when the controller’s redundancy is TMR.
Output Conversion (OUT_CVRT)

Block Category: OUT_CVRT, Type Conversion

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information.

The Output Conversion (OUT_CVRT) block converts a Long Integer into coded booleans. Binary Coded Decimal (BCD), BINARY, and GRAY coded booleans are decoded by this block. This *rubber block* supports up to 32 outputs in groups of four. If the input cannot be represented by the specified number of booleans the nearest representable encoding will be generated and the ERR pin is set True.

---

**Inputs**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>Boolean</td>
<td>Enable block execution</td>
</tr>
<tr>
<td>FCN</td>
<td>Constant InOutCvrt_T</td>
<td>Conversion to perform</td>
</tr>
<tr>
<td>IN</td>
<td>Double Integer</td>
<td>The numerical value to be encoded</td>
</tr>
<tr>
<td>Name</td>
<td>Data Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SIGN</td>
<td>Boolean</td>
<td>The sign of the coded decimal value (<em>False</em> = positive, <em>True</em> = negative).</td>
</tr>
<tr>
<td>ERR</td>
<td>Boolean</td>
<td><em>True</em> if the number cannot be exactly represented, otherwise <em>False</em></td>
</tr>
<tr>
<td>OUT11</td>
<td>Boolean</td>
<td>The first (lsb) output boolean in the first set of four booleans</td>
</tr>
<tr>
<td>OUT12</td>
<td>Boolean</td>
<td>The second output boolean in the first set of four booleans</td>
</tr>
<tr>
<td>OUT13</td>
<td>Boolean</td>
<td>The third output boolean in the first set of four booleans</td>
</tr>
<tr>
<td>OUT14</td>
<td>Boolean</td>
<td>The fourth (msb) output boolean in the first set of four booleans</td>
</tr>
<tr>
<td>↓</td>
<td>↓</td>
<td>↓</td>
</tr>
<tr>
<td>OUT81</td>
<td>Boolean</td>
<td>The first (lsb) output boolean in the eighth set of four booleans</td>
</tr>
<tr>
<td>OUT82</td>
<td>Boolean</td>
<td>The second output boolean in the eighth set of four booleans</td>
</tr>
<tr>
<td>OUT83</td>
<td>Boolean</td>
<td>The third output boolean in the eighth set of four booleans</td>
</tr>
<tr>
<td>OUT84</td>
<td>Boolean</td>
<td>The fourth (msb) output boolean in the eighth set of four booleans</td>
</tr>
</tbody>
</table>
Output Transfer (OUTXFER)

Block Category: OUTXFER, Legacy

Legacy Status — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information.

The Output Transfer (OUTXFER) block conditionally transfers data from a local variable into an input point owned by the device.

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRCVAR</td>
<td>Simple Input variable to be transferred</td>
</tr>
<tr>
<td>ENABLE</td>
<td>Boolean Block enable pin</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESTVAR</td>
<td>Output value</td>
</tr>
</tbody>
</table>
Override Station (OVR_ST)

Block Category: OVR_ST, Controls (DCS)

Legacy Status — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information.

Note This block is transparent to the HMI operator.

The Override Station (OVR_ST) block controls the output function of PID controllers used in an override control scheme. It takes inputs from two PID controllers and sets and produces CVO as a control variable output. CVO is determined by the selection type and/or value of control variable inputs in automatic mode and by the HMI operator in manual mode. Automatic and manual modes are superseded by override, priority, and inhibit commands with overrides having the highest priority, then priority commands, followed by inhibit commands. The value of CVO cannot exceed the user specified limits of high (H) and low (L).

ToolboxST Configuration

OBQ is True when either AO_BQ is True, status of CVI1 is in the BAD range, or status of CVI2 is in the BAD range.

This block propagates quality status. Status option cannot be disabled on this block.

CVO status is based upon the following in order of priority:

- When OBQ is True, the quality status of CVO is NOT_LIMITED-BAD [0]
- The default quality status of CVO is NOT_LIMITED-GOODC, 192.
- These values are modified by US (add 1) and LS (add 2)

For further details on Relational blocks, refer to GEI-100682, Mark VI Controller Standard Block Library, the section, Status Monitoring (STATUS_MONITORING).

Modes of Operation (MODE_OPT) enumerations control the mode of operation of the OVR_ST block when it is in auto mode.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELECT</td>
<td>Application code select</td>
</tr>
<tr>
<td>MIN</td>
<td>Minimum</td>
</tr>
<tr>
<td>MAX</td>
<td>Maximum</td>
</tr>
</tbody>
</table>

SELECT mode — CVI1 or CVI2 inputs are selected by user-defined logic using SEL1 and SEL2. If CVI1 is selected, NSEL2 becomes True, and if CVI2 is selected, NSEL1 becomes True. If SEL1 and SEL2 are both False or both True, then the value of CVI1 is written to CVO, NSEL2 becomes True, and NSEL1 becomes False.

MIN mode — the minimum value of CVI1 and CVI2 is written to CVO. If CVI1 is selected, NSEL2 becomes True, and if CVI2 is selected, NSEL1 becomes True. If CVI1 and CVI2 are equal, then the value of CVI1 is written to CVO, NSEL2 becomes True, and NSEL1 becomes False.

MAX mode — the maximum value of CVI1 and CVI2 is written to CVO. If CVI1 is selected, NSEL2 becomes True, and if CVI2 is selected, NSEL1 becomes True. If CVI1 and CVI2 are equal, the value of CVI1 is written to CVO, NSEL2 becomes True, and NSEL1 becomes False.
Block Options (BLOCK_OPT) enumerations select the options of the OVR_ST block.

**BLOCK_OPT Enumerations**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>None</td>
</tr>
<tr>
<td>LOCK</td>
<td>Lock mode</td>
</tr>
<tr>
<td>POS</td>
<td>Position feedback</td>
</tr>
<tr>
<td>LOCK-POS</td>
<td>Lock mode and position feedback</td>
</tr>
</tbody>
</table>

NONE — the valve operates normally by the HMI operator. The options to not effect the normal operation of the block.

**Note** The lockout indication is provided for indication only. This action will not lockout the equipment, it will only indicate lockout status. The Owner's lockout procedures must be followed to safely lock equipment out.

LOCK — provides software lock-out capability for the valve. Selecting the LOCK mode property will set LOCK_P to True, and enable the LOCK_PB and UNLOCK_PB to control the lockout mode functionality. The valve output CVO must be equal to LOCK_SC (safe condition) before the block can be set to lockout mode. Feedback to the HMI operator of this mode is provided by LOCK. The HMI operator is prevented from energizing the solenoid operated valve from the control faceplate when LOCK is True, and auto mode is disabled.

**Note** The Lock button can also be set up to use the WorkStationST Lock function. Refer to GEI-100697, WorkstationST/CIMPLICITY Advanced Viewer Integration, the section, HMI Tagout.

POS — select for valves that provide an analog position feedback, which is used by the HMI objects and faceplates to display the valve open percentage. When the POS enumeration is selected, the output POSFB_P is True.

**Override Control**

The block provides an override input enumeration, CMD_OVR and a manual reject input enumeration, MJ. When CMD_OVR indicates Override Active, the value of CVO will be the value assigned to the OV input. The OVR_ST is rejected to manual when either CMD_OVR or MJ indicate Override Active and the manual reject alarm, MN_REJ_A, will be activated (delayed by 2 sec).

CMD_OVR and MJ (if used) are driven from an OVERRIDE block. Each connected input of the OVERRIDE block has the ability (when enabled) to be blocked by the HMI operator, which is accessed from associated tab of the permit HMI screen display. When CMD_OVR or MJ enumeration contains BLOCK, the HMI operator has blocked at least one override. OVR_ST does not use the block information, but passes it to the HMI for display. The possible enumerations values are provided in the following table.

**Manual Reject and Command Override (MJ and CMD_OVR) Enumeration**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Override Option Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO_OVR-NO-BLOCK</td>
<td>No override and no override is blocked</td>
</tr>
<tr>
<td>OVR-NO BLOCK</td>
<td>Override active and no override is blocked</td>
</tr>
<tr>
<td>NO_OVR-BLOCK</td>
<td>No override and a override is blocked</td>
</tr>
<tr>
<td>OVR-BLOCK</td>
<td>Override active and a override is blocked</td>
</tr>
</tbody>
</table>

**Force Control**

The block provides a force enumeration force command input, CMD_FRC. When CMD_FRC indicates Force Active, the value of CVO will be the value assigned to the FV input except when CMD_OVR indicate Override Active. The OVR_ST does not reject to manual if a force occurs. CMD_FRC (if used) is driven from a FORCE block. Each connected input of the FORCE block has the ability (when enabled) to be blocked by the HMI operator, which is accessed from associated tab of the permit HMI screen display. When CMD_FRC enumeration contains BLOCK, the HMI operator has blocked at least one force. OVR_ST does not use the block information, but passes it to the HMI for display. The enumeration has the possible values provided in the following table.
Command Force (CMD_FRC) Enumeration

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Force Option Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO_FORCE-NO-BLOCK</td>
<td>No force and no force is blocked</td>
</tr>
<tr>
<td>FORCE-NO_BLOCK</td>
<td>Force active and no force is blocked</td>
</tr>
<tr>
<td>NO_FORCE-BLOCK</td>
<td>No force and a force is blocked</td>
</tr>
<tr>
<td>FORCE-BLOCK</td>
<td>Force active and a force is blocked</td>
</tr>
</tbody>
</table>

**OVERRIDE, PRIORITY and INHIBIT Commands**

When OVR is *True*, CVO is set to the value of OV as long as OVR is *True*. When priority raise PR_INC or priority lower PR_DEC signal is *True*, CVO goes toward H or L value at a rate set by PR_INC_RT and PR_DEC_RT inputs as long as the priority signal is *True*. In addition, the block also provides inhibit function using INH_INC and INH_DEC inputs. When either of these inputs is *True*, they restrict any increase or decrease of CVO. Upon the reset of the override, priority, and inhibit commands, the selected CVI input will be transferred to CVO at a rate defined by MR, provided the block is in auto mode. In manual mode, when they reset CVO remains at last value.

**Alarms**

When OBQ, CVI1_BQ, CVI2_BQ, OVR_CMD, LOCK, or MJ becomes *True* and AUTO is *True*, the manual reject alarm MN_REJ_A is activated for two seconds and the block reverts to manual mode.

When position feedback is used (the POS mode option selected – refer to the section, *BLOCK_OPT Enumerations*), an alarm, POS_DH, is generated for deviations between the position feedback (signal attached to POS input pin) and the controller output CVO. Use the parameters POS_DH_SP to enter the maximum acceptable deviation and POS_DH_T to set a delay time before generating an alarm. Note that if POS_DH_T is set to 0, no alarm will be generated. Also, the alarm POS_BQ is generated when the position feedback has bad status. Note that POS_BQ and POS_DH will only be generated if POSFB_P is *True*.

**Track**

The track command TK is *True* when a priority command, override, inhibiting condition, output bad quality, or manual mode occurs. This output is used on all PIDs connected to the OVR_ST block as a track input.

**Tuning Mode**

**Note** For tuning interface details, refer to the section, *PID (PID_MA)*.

When the PID_MA blocks are configured with OVR_ST, the user should connect TUN_OUT of PID_MA1 and PID_MA2 to TUN1 and TUN2 of OVR_ST, respectively. It forces the OVR_ST to select the corresponding PID_MA/CVI input while it is in tuning mode.
**ToolboxST Configuration**

Inserting an OVR_ST block into the application code, for example, displays the following window.

![Image of Enter the Attribute Instance Values window](image)

**Default Attribute Values**

When the user clicks OK, the block and pin connections are configured automatically and generates global pins. For this example, the pin names are in the form \( Device\ Value.\ Input\_Name \), where \( Device\ Value \) is the attribute value \( 00OVR\_ST1000 \) and \( Input\_Name \) is the block input/output names.

---

**Note** The description (OVR_ST Description in this example) becomes the prefix of the block input/output descriptions.

![Diagram of OVR_ST Block](image)
The OVR_ST block is connected to the PID_MA block in an external reset override scheme. The outputs of the PID_MA blocks are connected to the inputs CVI1 and CVI2 of the OVR_ST block. The output of the OVR_ST block connects to the PID_MA input EV. NSEL1 and NSEL2 are connected to the ER inputs of the PID_MA blocks.
The OVR_ST block is connected to the PID_MA block for a track override scheme. The outputs of the PID_MA blocks, CVO, are connected to the inputs CVI1 and CVI2 of the OVR_ST block. CVO of the OVR_ST block connects to the PID_MA input TV and NSEL1, NSEL2 and TK of the OVR_ST block connects to the TK inputs of the PID_MA blocks.

**OVR_ST and PID_MA Track-if Not Selected Configuration**
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data Type</th>
<th>Initial Value</th>
<th>Visibility</th>
<th>Interface Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AO_BQ</td>
<td>OUTPUT BAD QUALITY</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>AU_SEL</td>
<td>AUTO MODE SELECT</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>BLOCK_OPT</td>
<td>BLOCK OPTIONS</td>
<td>UINT ENUM</td>
<td>NONE</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>CMD_FRC</td>
<td>FORCE COMMAND</td>
<td>UINT ENUM</td>
<td>NO_FORCE-NO_BLOCK</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>CMD_OVR</td>
<td>OVERRIDE COMMAND</td>
<td>UINT ENUM</td>
<td>NO_OVR-NO_BLOCK</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>CVI1</td>
<td>CONTROL VARIABLE INPUT 1</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value with Status</td>
</tr>
<tr>
<td>CVI2</td>
<td>CONTROL VARIABLE INPUT 2</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value with Status</td>
</tr>
<tr>
<td>FV</td>
<td>FORCE VALUE</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>H</td>
<td>(Desc) OUTPUT UPR LIMIT</td>
<td>REAL</td>
<td>100</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>INH_DEC</td>
<td>(Desc) INHIBIT DEC</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>INH_INC</td>
<td>(Desc) INHIBIT INC</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>L</td>
<td>(Desc) OUTPUT LWR LIMIT</td>
<td>REAL</td>
<td>0</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>LOCK_PB</td>
<td>(Desc) LOCK PUSHPUTTON</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>LOCK_SC</td>
<td>(Desc) LOCK SAFE CONDITION</td>
<td>REAL</td>
<td>0</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>MC</td>
<td>(Desc) MANUAL SETPOINT</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>MJ</td>
<td>MANUAL REJECT</td>
<td>UINT ENUM</td>
<td>NO_OVR-NO_BLOCK</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>MODE_OPT</td>
<td>MODE SELECTION INPUT</td>
<td>UINT ENUM</td>
<td>SELECT</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>MR</td>
<td>MANUAL ADJUSTMENT RATE</td>
<td>REAL</td>
<td>10</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>MS</td>
<td>(Desc) MANUAL MODE</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>OV</td>
<td>OVERRIDE VALUE</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>POS</td>
<td>(Desc) POSITION FDBK</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value with Status</td>
</tr>
<tr>
<td>POS_DH_SP</td>
<td>(Desc) POSITION DEV HI SP</td>
<td>REAL</td>
<td>10</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>POS_DH_T</td>
<td>(Desc) POSITION DEV HI DELAY</td>
<td>UDINT</td>
<td>5000</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>PR_DEC</td>
<td>(Desc) PRIORITY DEC</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>PR_DEC_RT</td>
<td>PRIORITY DEC RATE</td>
<td>REAL</td>
<td>1</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>PR_INC</td>
<td>(Desc) PRIORITY INC</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>PR_INC_RT</td>
<td>PRIORITY INC RATE</td>
<td>REAL</td>
<td>1</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>SEL1</td>
<td>SELECT CVI1 IN SELECTION MODE</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>SEL2</td>
<td>SELECT CVI2 IN SELECTION MODE</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>TUN1</td>
<td>(Desc) PID 1 TUNING MODE</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>TUN2</td>
<td>(Desc) PID 2 TUNING MODE</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>UNLOCK_PB</td>
<td>(Desc) UNLOCK PUSHPUTTON</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
</tbody>
</table>
### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data Type</th>
<th>Initial Value</th>
<th>Visibility</th>
<th>Interface Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Device)</td>
<td>(Desc)</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>AUTO</td>
<td>AUTO</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>CVI1_BQ</td>
<td>CONTROL VARIABLE INPUT 1 BQ</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>CVI2_BQ</td>
<td>CONTROL VARIABLE INPUT 2 BQ</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>CVO</td>
<td>CONTROL VARIABLE OUTPUT</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value with Status</td>
</tr>
<tr>
<td>FRC_CMD</td>
<td>FORCE COMMAND</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>FRC_CMDB</td>
<td>FORCE COMMAND BLOCK STS</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>LOCK</td>
<td>LOCKED</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>LOCK_P</td>
<td>LOCK PROPERTY</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>MN_REJ_A</td>
<td>MANUAL REJECT</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>NSEL1</td>
<td>CVI1 NOT SELECTED</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>NSEL2</td>
<td>CVI2 NOT SELECTED</td>
<td>BOOL</td>
<td>True</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>OBQ</td>
<td>OUTPUT BQ</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>OVR_CMD</td>
<td>OVERRIDE COMMAND</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>OVR_CMDB</td>
<td>OVERRIDE COMMAND BLOCK STS</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>POS_BQ</td>
<td>POSITION FDBK BAD QUALITY</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>POS_DH</td>
<td>POSITION DEV HI ALARM</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>POSFB_P</td>
<td>POSITION FDBK PROPERTY</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>REJ_MN</td>
<td>MANUAL REJECT</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>REJ_MNB</td>
<td>MANUAL REJECT BYPASS STS</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>TK</td>
<td>TRACK COMMAND</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
</tbody>
</table>

### Global Pins

<table>
<thead>
<tr>
<th>Global Pin Name</th>
<th>Description</th>
<th>Alarm</th>
<th>Alarm Class</th>
<th>Event</th>
<th>EGD Page</th>
<th>External Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>00OVR_ST1000</td>
<td>OVR_ST Description</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.AUTO</td>
<td>OVR ST Description AUTO</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.CVI1_BQ</td>
<td>CONTROL VARIABLE INPUT 1 BQ</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.CVI2_BQ</td>
<td>CONTROL VARIABLE INPUT 2 BQ</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.CVO</td>
<td>CONTROL VARIABLE OUTPUT</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.CVO.Status</td>
<td>CONTROL VARIABLE OUTPUT STATUS</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.FRC_CMD</td>
<td>FORCE COMMAND</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.FRC_CMDB</td>
<td>FORCE COMMAND BLOCK STS</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.H</td>
<td>OUTPUT UPR LIMIT</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.INH_DEC</td>
<td>INHIBIT DEC</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.INH_INC</td>
<td>INHIBIT INC</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>Global Pin Name</td>
<td>Description</td>
<td>Alarm</td>
<td>Alarm Class</td>
<td>Event</td>
<td>EGD Page</td>
<td>External Access</td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------------------</td>
<td>-------</td>
<td>-------------</td>
<td>-------</td>
<td>----------</td>
<td>-----------------</td>
</tr>
<tr>
<td>00OVR_ST1000.L</td>
<td>OVR_ST Description OUTPUT LWR LIMIT</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.LOCK</td>
<td>OVR_ST Description LOCKED</td>
<td>Not Alarmed</td>
<td>True</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.LOCK_P</td>
<td>OVR_ST Description LOCK PROPERTY</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.LOCK_PB</td>
<td>OVR_ST Description LOCK PUSHBUTTON</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadWrite</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.LOCK_SC</td>
<td>OVR_ST Description LOCK SAFE CONDITION</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.MC</td>
<td>OVR_ST Description LOCKED</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadWrite</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.MN_REJ_A</td>
<td>OVR_ST Description MANUAL REJECT</td>
<td>Alarmed</td>
<td>LVL_4</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00OVR_ST1000.MS</td>
<td>OVR_ST Description MANUAL MODE SELECT</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadWrite</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.NSEL1</td>
<td>OVR_ST Description CVI1 NOT SELECTED</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.NSEL2</td>
<td>OVR_ST Description CVI2 NOT SELECTED</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.OBQ</td>
<td>OVR_ST Description OUTPUT BQ</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.OVR_CMD</td>
<td>OVR_ST Description OVERRIDE COMMAND</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.OVR_CMDB</td>
<td>OVR_ST Description OVERRIDE COMMAND BLOCK STS</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.POS</td>
<td>OVR_ST Description POSITION FDBK</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.POS.Status</td>
<td>OVR_ST Description POSITION FDBK STATUS</td>
<td>Not Alarmed</td>
<td>False</td>
<td>ReadOnly</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.POS_BQ</td>
<td>OVR_ST Description POSITION FDBK BAD QUALITY</td>
<td>Alarmed</td>
<td>LVL_4</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00OVR_ST1000.POS_DH</td>
<td>OVR_ST Description POSITION DEV HI ALARM</td>
<td>Alarmed</td>
<td>LVL_4</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00OVR_ST1000.POSFB_P</td>
<td>OVR_ST Description POSITION FDBK PROPERTY</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.PR_DEC</td>
<td>OVR_ST Description PRIORITY DEC</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.PR_INC</td>
<td>OVR_ST Description PRIORITY INC</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.REJ_MN</td>
<td>OVR_ST Description MANUAL REJECT</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.REJ_MNB</td>
<td>OVR_ST Description MANUAL REJECT BYPASS STS</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.TK</td>
<td>OVR_ST Description TRACK COMMAND</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00OVR_ST1000.TUN1</td>
<td>OVR_ST Description PID 1 TUNING MODE</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadWrite</td>
<td></td>
</tr>
</tbody>
</table>
### Global Pins (continued)

<table>
<thead>
<tr>
<th>Global Pin Name</th>
<th>Description</th>
<th>Alarm</th>
<th>Alarm Class</th>
<th>Event</th>
<th>EGD Page</th>
<th>External Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>00OVR_ST1000.TUN2</td>
<td>OVR_ST Description PID 2 TUNING MODE</td>
<td>Not Alarmed</td>
<td></td>
<td>False</td>
<td>$Default</td>
<td>ReadWrite</td>
</tr>
<tr>
<td>00OVR_ST1000.UNLOCK_PB</td>
<td>OVR_ST Description UNLOCK PUSHBUTTON</td>
<td>Not Alarmed</td>
<td></td>
<td>False</td>
<td>$Default</td>
<td>ReadWrite</td>
</tr>
</tbody>
</table>

### HMI Configuration (CIMPLICITY)

This block is transparent to the HMI operator.
**Permissive Toggle Action Engine (PE_Toggle)**

**Block Category:** PE_Toggle, Sequencing

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information.

The Permissive Toggle Action Engine (PE_Toggle) block sets its ON output to True when either the On Request (ONREQ) input or the Toggle Request (TOGREQ) input goes True, as long as the start and run permissive inputs are True. The next time the TOGREQ input goes high, the PE_Toggle is turned off. The PE_Toggle can also be turned on through the ONREQ input, and turned off through the OFFREQ input.

The PE_Toggle block will start (ON input goes high) when either the TOGREQ or ONREQ inputs go from 0 to 1 while both SPRM and RPRM inputs are True and the OFFREQ input is False. Once started, the SPRM input can go low. Additionally, the PE_Toggle block will start (ON input goes high) when the FDBK input goes from 0 to 1 as long as the RPRM inputs is True and the OFFREQ input is False. When the PE_Toggle block is started in this manner, the ONMSGP output will not pulse.

The PE_Toggle will also turn on if the FDBK input becomes True, as long as the run permissive input is True and the OFFREQ input is False. The FDBK input indicates to the PE_Toggle block that the actions that are supposed to happen when the PE_Toggle turns on actually happened. If the FDBK does not become True within the time defined by the FBTIME input then the PE_Toggle will turn off. The State variable CTIME contains the time since the PE_Toggle turned on. If the FBTIME is set to negative, the PE_Toggle block will consider the FBTIME to be infinite, in which case the FDBK input never has to become True and CTIME remains 0.

Provided the OFFPRM is high, the PE_Toggle block is turned off by setting the OFFREQ to high or by the TOGGLE input going high after the engine has turned on. When the PE_Toggle block turns off, the ON output goes low. The PE_Toggle is also turned off through the FDBK input going low or the RPRM signal going low. If the OFFPRM permissive is low, the PE_Toggle block will not turn off except by loss of the run permissive, RPRPM.

### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>Boolean</td>
<td>Enable block execution input</td>
</tr>
<tr>
<td>ONREQ</td>
<td>Boolean</td>
<td>Edge sensitive input used to turn engine on</td>
</tr>
<tr>
<td>TOGREQ</td>
<td>Boolean</td>
<td>Edge sensitive input used to turn engine on or off</td>
</tr>
<tr>
<td>OFFREQ</td>
<td>Boolean</td>
<td>Edge sensitive input used to turn engine off</td>
</tr>
<tr>
<td>SPRM</td>
<td>Boolean</td>
<td>Start permissive input</td>
</tr>
</tbody>
</table>
### Inputs (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPRM</td>
<td>Boolean</td>
<td>Run permissive input</td>
</tr>
<tr>
<td>OFFPRM</td>
<td>Boolean</td>
<td>Off permissive input</td>
</tr>
<tr>
<td>FDBK</td>
<td>Boolean</td>
<td>Start feedback (indicates we actually are running)</td>
</tr>
<tr>
<td>FBTIME</td>
<td>Long Integer</td>
<td>Time, milliseconds, until start feedback expected (a negative number disables the feedback requirement)</td>
</tr>
<tr>
<td>ON</td>
<td>Boolean</td>
<td>On output: True = PE_Toggle block is ON</td>
</tr>
<tr>
<td>CTIME</td>
<td>Long Integer</td>
<td>Current feedback time in milliseconds</td>
</tr>
<tr>
<td>STATUS</td>
<td>Unsigned Long Integer</td>
<td>Bit encoded status output (see below)</td>
</tr>
<tr>
<td>STATE</td>
<td>Unsigned Long Integer</td>
<td>Bit-encoded state output (see below)</td>
</tr>
<tr>
<td>LAST_FRAME</td>
<td>Unsigned Long Integer</td>
<td>Last frame in microseconds (internal)</td>
</tr>
<tr>
<td>NOT 1ST PASS</td>
<td>Boolean</td>
<td>Always True except first pass (internal)</td>
</tr>
</tbody>
</table>

### States

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Permission to turn off is present</td>
</tr>
<tr>
<td>2</td>
<td>Request to turn off is received</td>
</tr>
<tr>
<td>1</td>
<td>Request to turn on is received</td>
</tr>
<tr>
<td>0</td>
<td>Request to toggle on state is received</td>
</tr>
</tbody>
</table>

### Status

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>Feedback has been received before FBTIME</td>
</tr>
<tr>
<td>11</td>
<td>Feedback has been received</td>
</tr>
<tr>
<td>10</td>
<td>Run permissive received</td>
</tr>
<tr>
<td>9</td>
<td>Start permissive received</td>
</tr>
<tr>
<td>8</td>
<td>PE_Toggle block is ON</td>
</tr>
<tr>
<td>7–0</td>
<td>The lower 8 bits specify a code representing the current state of the PE_Toggle block</td>
</tr>
<tr>
<td></td>
<td>6 = ON and OK for off state</td>
</tr>
<tr>
<td></td>
<td>4 = ON with feedback received</td>
</tr>
<tr>
<td></td>
<td>3 = ON without feedback received</td>
</tr>
<tr>
<td></td>
<td>2 = Ready to start</td>
</tr>
<tr>
<td></td>
<td>1 = Not ready to start</td>
</tr>
<tr>
<td></td>
<td>0 = Undefined</td>
</tr>
</tbody>
</table>
**PID (PID_MA)**

**Block Category:** PID_MA, Controls (DCS)

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section *Replacement Blocks* for additional information.

The PID (PID_MA) block is a combination of a PID controller and MA (Manual/Auto) station. The block may be configured as a series or parallel controller with direct or reverse action. Additionally, all output calculations are superseded by override, priority and inhibit commands. Override commands have the highest priority, followed by priority commands. Inhibit commands have the lowest priority of the three types. The value of the output, CVO, can not exceed the user specified limits of high (H) and low (L).

**ToolboxST Configuration**

OBQ is *True* when either AO_BQ is *True* or status of TV is in the BAD range.

This block propagates quality status. Status option cannot be disabled on this block.

CVO status is based upon the following in order of priority:

- When OBQ is *True*, the quality status of CVO is *NOT_LIMITED-BAD [0]*
- If one of CVI_BQ, EV_BQ, PV_BQ, or RSP_BQ is *True*, the value of CVO status is *NOT_LIMITED-LOCAL_OVERRIDE-GOODC [216]*
- The default quality status of CVO is *NOT_LIMITED–GOODC [192]*
- These values are modified by US (add 1) and LS (add 2)

For further details on relational blocks, refer to *GEI-100682, Mark VIe Controller Standard Block Library*, the section *Status Monitoring (STATUS_MONITORING)*.
Modes of Operation (MODE_OPT) determine in which mode the PID_MA block is of the six modes available. Modes that do not contain PID are not controller enabled.

**MODE_OPT Enumerations**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PID</td>
<td>PID only</td>
</tr>
<tr>
<td>MA</td>
<td>M/A Station only</td>
</tr>
<tr>
<td>PID_MA</td>
<td>PID with M/A station</td>
</tr>
<tr>
<td>PID_MA_EXT</td>
<td>PID with M/A station with external setpoint</td>
</tr>
<tr>
<td>PID_MA_EXT_CASC</td>
<td>PID with M/A station with external setpoint from cascade station</td>
</tr>
<tr>
<td>PID_MA_REM</td>
<td>PID with M/A station with remote setpoint</td>
</tr>
<tr>
<td>PID_MA_REM_CASC</td>
<td>PID with M/A station with remote setpoint from cascade station</td>
</tr>
<tr>
<td>MA_EXT</td>
<td>MA Station without PID with Remote SP (RSP) with HMI SP disabled</td>
</tr>
<tr>
<td>MA_REM</td>
<td>MA Station without PID with HMI SP enabled (using Remote/Local)</td>
</tr>
<tr>
<td>MA_MAN</td>
<td>MA Station without PID with HMI Manual SP enabled</td>
</tr>
</tbody>
</table>

Block Options (BLOCK_OPT) enumerations selects the options of the OVR_ST block.

**BLOCK_OPT Enumerations**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>None</td>
</tr>
<tr>
<td>LOCK</td>
<td>Lock mode</td>
</tr>
<tr>
<td>POS</td>
<td>Position feedback</td>
</tr>
<tr>
<td>LOCK-POS</td>
<td>Lock mode and position feedback</td>
</tr>
</tbody>
</table>

NONE — the valve operates normally by the HMI operator. The options to not effect the normal operation of the block.

**Note** The lockout indication is provided for indication only. This action will not lockout the equipment, it will only indicate lockout status. The Owner's lockout procedures must be followed to safely lock equipment out.

LOCK — provides software lock-out capability for the valve. Selecting the LOCK mode property will set LOCK_P to True, and enable the LOCK_PB and UNLOCK_PB to control the lockout mode functionality. The valve output CVO must be equal to LOCK_SC (safe condition) before the block can be set to lockout mode. Feedback to the HMI operator of this mode is provided by LOCK. The HMI operator is prevented from energizing the solenoid operated valve from the control faceplate when LOCK is True, and auto mode is disabled.

**Note** The Lock button can also be set up to use the Workstation Lock function. Refer to GEI-100697, WorkstationST/CIMPLICITY Advanced Viewer Integration, the section, HMI Tagout.

POS — select for valves that provide an analog position feedback, which is used by the HMI objects and faceplates to display the valve open percentage. When the POS enumeration is selected, the output POSFB_P is True.
Override Control

The block provides an override input enumeration, CMD_OVR and a manual reject input enumeration, MJ. The enumerations are provided in the following table.

When CMD_OVR indicates Override Active, the value of CVO will be the value assigned to the OV input. The PID_MA is rejected to manual when either CMD_OVR or MJ indicate Override Active and the manual reject alarm, MN_REJ_A, will be activated, (delayed by 2 seconds). Refer to the section, for further details about MN_REJ_A.

CMD_OVR and MJ (if used) are driven from an OVERRIDE block. Each connected input of the OVERRIDE block has the ability (when enabled) to be blocked by the HMI operator, which is accessed from associated tab of the permit HMI screen display. When CMD_OVR or MJ enumeration contains BLOCK, the HMI operator has blocked at least one override. PID_MA does not use the block information, but passes it to the HMI for display.

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Override Option Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO_OVR-NO-BLOCK</td>
<td>No override and no override is blocked</td>
</tr>
<tr>
<td>OVR-NO_BLOCK</td>
<td>Override active and no override is blocked</td>
</tr>
<tr>
<td>NO_OVR-BLOCK</td>
<td>No override and a override is blocked</td>
</tr>
<tr>
<td>OVR-BLOCK</td>
<td>Override active and a override is blocked</td>
</tr>
</tbody>
</table>
Force Control

The block provides a force enumeration force command input, CMD_FRC. When CMD_FRC indicates Force Active, the value of CVO will be the value assigned to the FV input except when CMD_OVR indicate Override Active. The PID_MA does not reject to manual if a force occurs. CMD_FRC (if used) is driven from a FORCE block. Each connected input of the FORCE block has the ability (when enabled) to be blocked by the HMI operator, which is accessed from associated tab of the permit HMI screen display. When CMD_FRC enumeration contains BLOCK, the HMI operator has blocked at least one force. PID_MA does not use the block information, but passes it to the HMI for display. The enumeration is provided in the following table.

### Command Force (CMD_FRC) Enumerations

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Force Option Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO_FORCE-NO-BLOCK</td>
<td>No force and no force is blocked</td>
</tr>
<tr>
<td>FORCE-NO_BLOCK</td>
<td>Force active and no force is blocked</td>
</tr>
<tr>
<td>NO_FORCE-BLOCK</td>
<td>No force and a force is blocked</td>
</tr>
<tr>
<td>FORCE-BLOCK</td>
<td>Force active and a force is blocked</td>
</tr>
</tbody>
</table>

Priority and Inhibit Commands

**Note** These commands apply to all modes of PID_MA.

When priority increase PR_INC or priority decrease PR_DEC signal is True, CVO goes toward H or L value at a rate set by PR_INC_RT and PR_DEC_RT as long as the priority signal is True. In addition, the block also provides inhibit function using INH_INC and INH_DEC inputs. When either of these inputs is True, they restrict any increase or decrease of CVO. Upon the reset of the priority and inhibit commands, the block is released to control, provided the block is in auto mode. In manual mode, when these commands reset, CVO remains at last value.

Alarms

When OBQ, CVI_BQ, EV_BQ, PV_BQ, RSP_BQ, TV_BQ, OVR_CMD, LOCK, RSP_OV, or MJ becomes True and AUTO is True, the manual reject alarm MN_REJ_A is activated for two seconds and the block reverts to manual mode.

**Note** Refer to the table, BLOCK_OPT Enumerations.

When position feedback is used (POS mode option selected), a POS_DH alarm is generated for deviations between the position feedback (signal attached to POS input pin) and the controller output CVO. Use the parameter POS_DH_SP to enter the maximum acceptable deviation and POS_DH_T to set a delay time before generating an alarm.

**Note** If POS_DH_T is set to 0, no alarm will be generated. If the position feedback has a bad status, the alarm POS_BQ is generated. POS_BQ and POS_DH will only be generated if POSFB_P is True.
Controller Action (CTRL_ACT) can be selected using this enumeration as follows. This enumeration is used with all block modes except MA (M/A Station Only).

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIRECT</td>
<td>Direct action (PV-SP)</td>
</tr>
<tr>
<td>REVERSE</td>
<td>Reverse action (SP-PV)</td>
</tr>
</tbody>
</table>

Controller Type (CTRL_TYPE) provides a selection of either series or parallel type of PID control implementation. This enumeration is used with all block modes except MA (M/A station only).

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SERIES</td>
<td>Series control</td>
</tr>
<tr>
<td>PARALLEL</td>
<td>Parallel control</td>
</tr>
</tbody>
</table>

The following equations are used to calculate controller's output.

Parallel:

\[ OUT = K_p \left[ e(t) + \frac{1}{\tau} \int e(t) \, dt + D \frac{d}{dt} e(t) \right] \]

Series:

\[ OUT = K_p \left[ e(t) + \frac{1}{\tau} \int e(t) \, dt \right] + D \frac{d}{dt} e(t) \]

Where:
- \( K_p \) = Proportional Gain (PG)
- \( e(t) \) = Error (or Error x Error, if SQR_ERR input is enabled)
- \( \tau \) = Reset time (IG)
- \( D \) = Derivative Gain (DG)
Additional Controller Enabled Functions

- HMI operator output command in manual mode (MC) is only allowed to change at a maximum rate specified by MR. The manual output is limited by H and L parameters as well as override, priority and inhibit commands.
- All controller enabled functions require the setting of the scale factor (SF). SF is used to scale the PV and SP into output units. For example, a PV and SP in RPM of 0 – 3600 would require an SF of 0.0277 to convert the output to 0 – 100%.
- External tracking is enabled by setting TK to True.
- The block output follows the track value TV while TK is True. Normal control functions resume with TV as the starting point upon the reset of TK.
- The block output follows the track value TV while TK is True. Normal control functions resume with TV as the starting point upon the reset of TK.
- The block provides a feed forward input (FF), which is added directly to the output in a controller enabled mode.
- The PID_MA block uses an anti-reset windup feature that is transparent to the user. When the block output reaches its high or low limit (user defined H or L) or when a inhibit signal (INH_DEC, INH_INC) is present, the integral action is prevented to move further in the same direction. When the direction of error signal changes or inhibit signal goes away, the integral action is resumed for normal operation. With anti-reset windup, PID_MA block provides improved control by decreasing delays due to reset winding.
- Setpoint control is accomplished using a few different parameters. There are two sources for the setpoint, SC (HMI Operator Setpoint) and RSP (Logic Driven Setpoint when RS is True). SH and SL establish the high and low limits of the setpoint regardless of source. SR sets the maximum allowed rate of change of the setpoint inputs. This rate limiter is bypassed when RS is True and when RSP_SR_DSBL is True. The rate and value limited setpoint that is used in the PID calculation is SP. This setpoint monitoring is disabled when the block is accepting the remote setpoint, RS is True. In addition, SP_TRACK is used force the operator setpoint, SC, to track PV in manual mode and RSP in remote setpoint mode.
- When set to True, the parameter INH_PDG will inhibit proportional and derivative action on setpoint change.

**Note** When using any modes that accept a Remote Setpoint, care should be taken in the setting of the INH_PDG parameter (default initial value of True) to obtain the desired controller response. When INH_PDG is set to True, continuously changing setpoints will make the controller an integral only controller. Conversely, when INH_PDG is set to False, continuously changing setpoints will have the proportional, integral and derivative action applied to the output based on the gains and the error generated between the moving setpoint and the process variable. This feature only applies to parallel controllers! INH_PDG has no effect on series type controllers. The user must account for proportional and derivative action of both the series type controller and the parallel type controller with the INH_PDG set to False by limiting the ramp rate of the setpoint.

- PG (%/%), IG (repeats/min), (minutes) are the proportional, integral and derivative gains that are used in the PID calculations.
- SQR_ERR allows the user to square the error which creates a faster controller response when the actual error is greater than one and a slower controller response when the actual error is less than one.
- ERR_C enables the use of error deadband equality processing. When the absolute value of the error is less than the controller error deadband (ERR_CDB) assumes the error is zero.
- PG_C and PG_CDB inputs to the PID_MA block are provided for a bumpless transition whenever a change to the proportional gain is made. When PG_C is True, rate of change of PG is compared to the value set by PG_CDB (maximum change allowed in one scan). If this value is exceeded, the controller aligns the values for that scan to remain constant and begin controller action on the next scan without a bump in the output. PG_CDB must be set so that normal changes in PG are ignored (variable gain) and this parameter only initiates during abrupt changes in PG. Similarly, changes made to IG are governed by IG_CDB when IG_C is True.
- Output Bad Quality (OBQ) is designed to reject the block to manual mode when output bad quality is detected and keep it in manual mode as long as OBQ is True.
- External Reset (ER) is used in an override control scheme. When ER is True, the PID calculation uses EV as a reset value to begin the next calculation. This is typically used when you have two PIDs with different PVs controlling the same output based on which one has the minimum or maximum output.
- When the block is in PID_MA_EXT_CASC or PID_MA_REM_CASC mode, RSP status is monitored to detect an override condition in the upstream PID_MA block. If an override condition is detected, RSP_OV is set to True and the block is rejected to manual mode.
- If PV_SP_LIM_TRK is True when the controller output reaches either the high or low limit, H or L, the controller will track the limit value until PV crosses SP. Once PV crosses SP, the controller will resume the controlling action. The following graph displays PV, SP, and CVO for a PID_MA block with PV_SP_LIM_TRK both True and False. Notice that with PV_SP_LIM_TRK set to True (left of the black vertical line) CVO remains at the limit until PV crosses SP.
With PV\_SP\_LIM\_TRK set to \textit{False} (right of the black vertical line) CVO only tracks the limit until PV changes slope. The graph displays PV (blue sine wave), \textit{SP} (green horizontal line), and \textit{CVO} (brown sine wave, capped). With \texttt{PV\_SP\_LIM\_TRK} set to \textit{True} the controller behaves as shown by the data to the left of the vertical black line. With \texttt{PV\_SP\_LIM\_TRK} set to \textit{False} the controller behaves as shown by the data to the right of the vertical black line.

\textbf{PV\_SP\_LIM\_TRK Slope Graph}

\textit{Note} Refer to the table, \textit{PID\_MA Block Pins Usage by Mode}.

\textit{MA mode} is a MA (Manual/Auto) station (no PID action). There is no PID action. In this mode, the HMI operator is allowed to adjust both the automatic mode setpoint and output. A remote setpoint is not allowed in this configuration. Refer to the section, \textit{Additional Controller Enabled Functions}.

\textit{Note} The PV bar graph visibility can be changed with an option on the graphic object.

\textbf{Functional Details for PID\_MA in MA Mode}
**Note** Refer to the table, *PID MA Block Pins Usage by Mode*.

**MA_MAN mode** is a MA (Manual/Auto) station (no PID action). There is no PID action. In this mode, the HMI operator is only allowed to adjust the output. Refer to the section, *Additional Controller Enabled Functions*.

**Note** The PV bar graph visibility can be changed with an option on the graphic object.

---

**Functional Details for PID MA in MA Mode**

**Note** Refer to the table, *PID MA Block Pins Usage by Mode*.
**MA_EXT mode** is a MA (Manual/Auto) station (no PID action). There is no PID action. In this mode, the HMI operator is not allowed to adjust the automatic mode setpoint. The setpoint is controlled by the remote setpoint (RSP) in this configuration. Refer to the section, *Additional Controller Enabled Functions*.

**Note** The PV bar graph visibility can be changed with an option on the graphic object.

---

**Note** Refer to the table, *PID_MA Block Pins Usage by Mode*.
MA_REM mode is a MA (Manual/Auto) station (no PID action). There is no PID action. In this mode, the HMI operator is allowed to adjust the setpoint in local mode, but not in remote mode. The remote setpoint is RSP. Refer to the section, Additional Controller Enabled Functions.

Note  The PV bar graph visibility can be changed with an option on the graphic object.

---

Note  Refer to the table, PID_MA Block Pins Usage by Mode.

---

![Diagram of PID_MA block in MA_REM mode](image)
**PID_MA mode** is a combination of a PID controller and a MA (Manual/Auto) station. The block can be configured as a series or parallel controller with direct or reverse action when used in this mode. In this mode, the HMI operator is allowed to adjust both the setpoint and output. A remote setpoint is not allowed in this configuration. Refer to the section, *Additional Controller Enabled Functions*.

**Note**  The PV bar graph visibility can be changed with an option on the graphic object.
**PID_MA_EXT Mode** is functionally equivalent to PID_MA mode except that it only accepts a remote setpoint, RSP. In this mode, the HMI Operator is not allowed to adjust the setpoint or view the source of the setpoint. Refer to the section, *Additional Controller Enabled Functions*.

---

**Note** The PV bar graph visibility can be changed with an option on the graphic object.
**PID_MA_EXT_CASC mode** is functionally equivalent to PID_MA mode except that it only accepts a remote setpoint, RSP. In this mode, the HMI Operator is not allowed to adjust the setpoint, but they are allowed to view the source of the setpoint using the CASC SP button. When used in this mode, the HMI faceplate is automatically configured to display a CASC SP button. This button allows the operator to pop-up another window to interface with upstream PID_MA block (typically in PID mode).

Additionally if the outer loop PID_MA block’s MODE_OPT is set to PID, the user can opt to show a single faceplate with the process variable from the outer loop, the setpoint from the outer loop, and the output from the inner loop. The option to perform this option is selected on the graphic object. Refer to the section, *Additional Controller Enabled Functions*.

---

**Note** The PV bar graph visibility can be changed with an option on the graphic object.
**PID_MA_REM mode** is functionally equivalent to PID_MA mode except that it accepts a remote setpoint, RSP and an HMI Operator setpoint, SC. In this mode, the HMI Operator is allowed to adjust the setpoint in local mode, but not in remote mode and does not have access to the source of the remote setpoint. The REM and LOCAL buttons are used by the HMI Operator to control the source of the setpoint. Refer to the section, *Additional Controller Enabled Functions*.

**Note**  The PV bar graph visibility can be changed with an option on the graphic object.
**PID_MA_REM_CASC mode** is functionally equivalent to PID_MA mode except that it accepts a remote setpoint, RSP and an HMI Operator setpoint, SC. In this mode, the HMI Operator is allowed to adjust the setpoint in local mode and can access the source of the remote setpoint to change its value using the CASC SP button. When used in this mode, HMI faceplate is automatically configured to display a CASC SP button. This button allows the operator to pop-up another window to interface with upstream PID_MA block (typically in PID_MA mode). The REM and LOCAL buttons are used by the HMI Operator to control the source of the setpoint. Refer to the section, *Additional Controller Enabled Functions*.

Additionally, the output of first PID_MA block is connected to RSP input for a secondary block for cascade action.

---

**Note**  The PV bar graph visibility can be changed with an option on the graphic object.
NOTE:
PRIMARY (PID_MA_1) CONTROLLERS TRACK VALUE (TV)
INPUT MUST BE USED FOR SMOOTH ACTION WHEN
SECONDARY (PID_MA_2) CONTROLLER IS IN MANUAL,
RECEIVED A PRIORITY OR INHIBIT INPUT.

PID_MA Block Typical Cascade Configuration
PID_MA with Proportional Gain (PROP_GAIN) Block

*Note* Refer to GEI-100679, Mark VIe Controller DCS Block Library, the section, Proportional Gain (PROP_GAIN).

The PROP_GAIN block may be used with the PID_MA block to automatically adjust a gain value on the PID_MA block. The following figure is a typical configuration of a PROP_GAIN block with the PID_MA block.

*Note* For this example, the PROP_GAIN output is driving the proportional gain (PG), but it may also be used to drive the integral gain (IG).
The PID_MA block has feedforward capability using the FF pin. The following figure is a typical configuration of the PID_MA block with an interpolation block providing the feedforward value. A separate PID controller block could also be used instead of a derivative block.
Tuning Mode

**Note** The user has full control of the output in this mode and must take care to know the upstream and downstream effects of their actions.

Each PID MA faceplate has a TUNE button that opens a screen with an embedded trend containing all variables necessary to perform tuning functions. Tuning is enabled and disabled from this screen. Once enabled, the user can change tuning parameters as well as control the modes of the station to allow output and setpoint changes to aid in tuning. This button is only made visible when the proper security login is entered. The TUNE button is not visible when the block is in MA mode.

![PID MA Description](image)

**Tune Button Enabled**

The mode of the block is not effected, if it is PID MA, PID MA REM or PID REM CASC. The following mode changes take place when TUN is *True* to allow output and setpoint changes while in tuning mode:

- PID mode changes to PID MA REM mode.
- PID MA EXT mode changes to PID MA REM mode.
- PID MA EXT CASC mode changes to PID MA REM CASC mode.

The block returns to its original mode when TUN is set to *False*. When the PID MA block is used in an override control scheme, the OVR ST block ensures that the PID that is in Tune mode is selected to drive the output to facilitate tuning.
Override Control Configuration

When PID.Ma is used with an Override Station (OVR.ST) block, the override control input, OVR_CTRL, should be set to True. This input keeps the block locked in automatic mode. MJ and OBQ are also ignored when OVR_CTRL is enabled. Note if OVR_CTRL is not enabled while using an OVR.ST, the faceplates will not function correctly. The TUN_OUT parameter of each PID.Ma should be connected to the corresponding TUN input of the OVR.ST.

In an external reset override scheme, the PID.Ma block is connected to the OVR.ST block. The outputs of the PID.Ma blocks are connected to the inputs CVI1 and CVI2 of the OVR.ST block. The output of the OVR.ST block connects to the PID.Ma input EV. NSEL1 and NSEL2 are connected to the ER inputs of the PID.Ma blocks.

**OVR.ST and PID.Ma Reset-if Not Selected Configuration**
The OVR_ST block is connected to the PID_MA block as displayed in the following figure for a tracking override scheme. The outputs of the PID_MA blocks, CVO, are connected to the inputs CVI1 and CVI2 of the OVR_ST block. CVO of the OVR_ST block connects to the PID_MA input TV and NSEL1, NSEL2 and TK of the OVR_ST block connects to the TK inputs of the PID_MA blocks.
PID_MA Block MA Station Usage

M/A Setpoint Modes MA, MA_EXT, and MA_REM
Output Station

Traditional PID_MA with M/A Station Configuration
ToolboxST Configuration for the PID_MA block

**Note** Each device name used in a controller must be unique.

When a PID_MA block is inserted into application code, it generates a pop-up window prompting user to enter Device Name, Device Description, Format Specification and HMI Screen Name. The block automatically creates the variables associated with the block and provides the appropriate attributes for each variable (for example Type, Format Spec, variables that need to be on EGD will be automatically placed on the $Default EGD page, and so forth). The HMI screen name is used as an Alarm attribute.

![Attribute Instance Values](image)

**Default Attribute Values**

When the user clicks OK, the block and pin connections will be configured automatically and generates *global pins*. In this example, the pin names are in the form `Device Value.pin Name`, where `Device Value` has the attribute value `00PID_MA1000` and `Input_Name` is the block input/output names.
PID_MA Block
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data Type</th>
<th>Initial Value</th>
<th>Visibility</th>
<th>Interface Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AO_BQ</td>
<td>OUTPUT BAD QUALITY</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>AU_SEL</td>
<td>AUTO MODE SELECT</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>BLOCK_OPT</td>
<td>BLOCK OPTIONS</td>
<td>UINT (ENUM)</td>
<td>NONE</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>CMD_FRC</td>
<td>FORCE COMMAND</td>
<td>UINT (ENUM)</td>
<td>NO_FORCE-NO_BLOCK</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>CMD_OVR</td>
<td>OVERRIDE COMMAND</td>
<td>UINT (ENUM)</td>
<td>NO_OVR-NO_BLOCK</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>CTRL_ACT</td>
<td>CONTROLLER ACTION</td>
<td>UINT (ENUM)</td>
<td>DIRECT</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>CTRL_TYPE</td>
<td>CONTROLLER TYPE</td>
<td>UINT (ENUM)</td>
<td>PARALLEL</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>CVI</td>
<td>[Desc] CONTROL VARIABLE INPUT</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value with Status</td>
</tr>
<tr>
<td>DG</td>
<td>(Desc) PID DERIVATIVE GAIN</td>
<td>REAL</td>
<td>0</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>ER</td>
<td>(Desc) PID EXTERNAL RESET ENABLED</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>ERR_C</td>
<td>ENABLE ERROR DEADBAND EQUALITY</td>
<td>BOOL</td>
<td>False</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>ERR_CDB</td>
<td>ERROR DEADBAND FOR PV_SP EQUALITY</td>
<td>REAL</td>
<td>0</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>EV</td>
<td>PID EXTERNAL RESET VALUE</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value with Status</td>
</tr>
<tr>
<td>FF</td>
<td>PID FEEDFORWARD VALUE</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>FV</td>
<td>FORCE VALUE</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>H</td>
<td>(Desc) PID UPR LIMIT</td>
<td>REAL</td>
<td>100</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>IG</td>
<td>(Desc) PID INTEGRAL GAIN</td>
<td>REAL</td>
<td>3</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>IG_C</td>
<td>ENABLE BUMPLESS IG CHANGES</td>
<td>BOOL</td>
<td>True</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>IG_CDB</td>
<td>IG BUMPLESS CHANGE DEADBAND</td>
<td>REAL</td>
<td>0.1</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>INH_DEC</td>
<td>(Desc) INHIBIT DEC</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>INH_INC</td>
<td>(Desc) INHIBIT INC</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>INH_PDG</td>
<td>INHIBIT PROPORTIONAL AND DERIVATIVE ACTION ON SETPOINT CHANGE</td>
<td>BOOL</td>
<td>True</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>L</td>
<td>(Desc) PID LWR LIMIT</td>
<td>REAL</td>
<td>0</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>LOCK_PB</td>
<td>(Desc) LOCK PUSHPUSHBUTTON</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>LOCK_SC</td>
<td>(Desc) LOCK SAFE CONDITION</td>
<td>REAL</td>
<td>0</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>MC</td>
<td>(Desc) MANUAL SAFE CONDITION</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>MJ</td>
<td>MANUAL REJECT</td>
<td>UINT (ENUM)</td>
<td>NO_OVR-NO_BLOCK</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>MODE_OPT</td>
<td>PID_MA MODE CONTROL</td>
<td>UINT (ENUM)</td>
<td>PID_MA</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>MR</td>
<td>MANUAL ADJUSTMENT RATE</td>
<td>REAL</td>
<td>10</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>MS</td>
<td>(Desc) MANUAL MODE</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>OV</td>
<td>OVERRIDE VALUE</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td>Data Type</td>
<td>Initial Value</td>
<td>Visibility</td>
<td>Interface Type</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------</td>
<td>-----------</td>
<td>---------------</td>
<td>-------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>OVR_CTRL</td>
<td>PID IS PART OF OVERRIDE CONTROL SCHEME</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>PG</td>
<td>(Desc) PID PROPORTIONAL GAIN</td>
<td>REAL</td>
<td>1</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>PG_C</td>
<td>ENABLE BUMPLESS PG CHANGES</td>
<td>BOOL</td>
<td>True</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>PG_CDB</td>
<td>PG BUMPLESS CHANGE DEADBAND</td>
<td>REAL</td>
<td>1</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>POS</td>
<td>(Desc) POSITION FDBK</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value with Status</td>
</tr>
<tr>
<td>POS_DH_SP</td>
<td>(Desc) POSITION DEV HI SP</td>
<td>REAL</td>
<td>5</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>POS_DH_T</td>
<td>(Desc) POSITION DEV HI DELAY</td>
<td>UDINT</td>
<td>2000</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>PR_DEC</td>
<td>(Desc) PRIORITY DEC</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>PR_DEC_RT</td>
<td>PRIORITY DEC RATE</td>
<td>REAL</td>
<td>1</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>PR_INC</td>
<td>(Desc) PRIORITY INC</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>PR_INC_RT</td>
<td>PRIORITY INC RATE</td>
<td>REAL</td>
<td>1</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>PV</td>
<td>(Desc) PID PROCESS VARIABLE</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value with Status</td>
</tr>
<tr>
<td>PV_SP_LIM_TRK</td>
<td>PID TRACK AT LIMIT UNTIL PV_SP CROSS</td>
<td>BOOL</td>
<td>True</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>RS</td>
<td>(Desc) REMOTE SETPOINT MODE</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>RSP</td>
<td>(Desc) REMOTE SETPOINT</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value with Status</td>
</tr>
<tr>
<td>RSP_SR_DSBL</td>
<td>REMOTE SETPOINT RATE DISABLE</td>
<td>BOOL</td>
<td>True</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>SC</td>
<td>(Desc) SETPOINT</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>SF</td>
<td>PID SCALE FACTOR</td>
<td>REAL</td>
<td>1</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>SH</td>
<td>(Desc) SETPOINT UPPER LIMIT</td>
<td>REAL</td>
<td>100</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>SL</td>
<td>(Desc) SETPOINT LOWER LIMIT</td>
<td>REAL</td>
<td>0</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>SP_TRACK</td>
<td>(Desc) SETPOINT TRACKS PV IN MANUAL MODE</td>
<td>BOOL</td>
<td>True</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>SQR.ERR</td>
<td>SQUARE ERROR USED IN PROPORTIONAL TERM</td>
<td>BOOL</td>
<td>False</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>SR</td>
<td>SETPOINT ADJUSTMENT RATE</td>
<td>REAL</td>
<td>10</td>
<td>Parameter</td>
<td>Value Only</td>
</tr>
<tr>
<td>TK</td>
<td>(Desc) TRACKING ACTIVE</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>TUN</td>
<td>(Desc) TUNING MODE</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>TV</td>
<td>PID TRACK VALUE</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value with Status</td>
</tr>
<tr>
<td>UNLOCK_PB</td>
<td>(Desc) UNLOCK PUSHBUTTON</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td>Data Type</td>
<td>Initial Value</td>
<td>Visibility</td>
<td>Interface Type</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------------</td>
<td>-----------</td>
<td>---------------</td>
<td>------------</td>
<td>----------------</td>
</tr>
<tr>
<td>AUTO</td>
<td>(Desc) AUTO</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>CVI_BQ</td>
<td>(Desc) CONTROL VARIABLE INPUT BQ</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>CVO</td>
<td>(Desc) CONTROL VARIABLE OUTPUT</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value with Status</td>
</tr>
<tr>
<td>EV_BQ</td>
<td>(Desc) PID EXTERNAL RESET VALUE BQ</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>FRC_CMD</td>
<td>(Desc) FORCE COMMAND</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>FRC_CMDB</td>
<td>(Desc) FORCE COMMAND BLOCK STS</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>LOCK</td>
<td>(Desc) LOCKED</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>LOCK_P</td>
<td>(Desc) LOCK PROPERTY</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>LS</td>
<td>(Desc) PID AT LWR LIMIT</td>
<td>BOOL</td>
<td>True</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>MA_EXT_P</td>
<td>(Desc) MA_ST — EXTERNAL SP</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>MA_MAN_P</td>
<td>(Desc) MA_ST — MANUAL SP</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>MA_P</td>
<td>(Desc) MA_ST — AUTO SP and MANUAL SP</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>MA_REM_P</td>
<td>(Desc) MA_ST — REMOTE SP and LOCAL SP</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>MN_REJ_A</td>
<td>(Desc) MANUAL REJECT</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>OBQ</td>
<td>(Desc) OUTPUT BQ</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>OVR_CMD</td>
<td>(Desc) OVERRIDE COMMAND</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>OVR_CMDB</td>
<td>(Desc) OVERRIDE COMMAND BLOCK STS</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>OVR_CTRL_P</td>
<td>(Desc) OVERRIDE CONTROL PROPERTY</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>PID_MA_EXT_C_P</td>
<td>(Desc) PID and MA_ST — EXTERNAL SP FROM CASC PID</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>PID_MA_EXT_P</td>
<td>(Desc) PID and MA_ST — EXTERNAL SP</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>PID_MA_P</td>
<td>(Desc) PID and MA_ST — AUTO SP and MANUAL SP</td>
<td>BOOL</td>
<td>True</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>PID_MA_REM_C_P</td>
<td>(Desc) PID and MA_ST — REMOTE SP FROM CASC PID and LOCAL SP</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>PID_MA_REM_P</td>
<td>(Desc) PID and MA_ST — REMOTE SP and LOCAL SP</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>PID_P</td>
<td>(Desc) PID — WITHOUT MA_ST</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>POS_BQ</td>
<td>(Desc) POSITION FDBK BAD QUALITY</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>POS_DH</td>
<td>(Desc) POSITION DEV HI ALARM</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>POSFB_P</td>
<td>(Desc) POSITION FDBK PROPERTY</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>PV_BQ</td>
<td>(Desc) PID PROCESS VARIABLE BQ</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>REJ_MN</td>
<td>(Desc) MANUAL REJECT</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>REJ_MNB</td>
<td>(Desc) MANUAL REJECT BYPASS STS</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>RSP_BQ</td>
<td>(Desc) REMOTE SETPOINT BQ</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>RSP_OV</td>
<td>(Desc) REMOTE SETPOINT OVERRIDE</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>SP</td>
<td>(Desc) SETPOINT</td>
<td>REAL</td>
<td>0</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>TK_OUT</td>
<td>PID INTERNAL TRACKING (MAN CNT) ENABLED</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
</tbody>
</table>
### Outputs (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data Type</th>
<th>Initial Value</th>
<th>Visibility</th>
<th>Interface Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>TUN_OUT</td>
<td>PID IN TUNING MODE ENABLED</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
<tr>
<td>TV_BQ</td>
<td>(Desc) PID TRACK VALUE BQ</td>
<td>BOOL</td>
<td>False</td>
<td>Internal</td>
<td>Value Only</td>
</tr>
<tr>
<td>US</td>
<td>(Desc) PID AT UPR LIMIT</td>
<td>BOOL</td>
<td>False</td>
<td>Always</td>
<td>Value Only</td>
</tr>
</tbody>
</table>

### Global Pins

<table>
<thead>
<tr>
<th>Global Pin Name</th>
<th>Description</th>
<th>Alarm</th>
<th>Alarm Class</th>
<th>Event</th>
<th>EGD Page</th>
<th>External Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>00PID_MA1000</td>
<td>PID_MA Description</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.AUTO</td>
<td>PID_MA Description AUTO</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.CVI</td>
<td>PID_MA Description CONTROL VARIABLE INPUT</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.CVI_BQ</td>
<td>PID_MA Description CONTROL VARIABLE INPUT BQ</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.CVO</td>
<td>PID_MA Description CONTROL VARIABLE OUTPUT</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.CVO_Status</td>
<td>PID_MA Description CONTROL VARIABLE OUTPUT STATUS</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.DG</td>
<td>PID DERIVATIVE GAIN (MINUTES)</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadWrite</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.EV_BQ</td>
<td>PID_MA Description PID EXTERNAL RESET VALUE BQ</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.FRC_CMD</td>
<td>PID_MA Description FORCE COMMAND</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.FRC_CMDB</td>
<td>PID_MA Description FORCE COMMAND BLOCK STS</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.H</td>
<td>PID_MA Description PID UPR LIMIT</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.IG</td>
<td>PID INTEGRAL GAIN (REPEATS/MINUTE)</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadWrite</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.INH_DEC</td>
<td>PID_MA Description INHIBIT DEC</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.INH_INC</td>
<td>PID_MA Description INHIBIT INC</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.L</td>
<td>PID_MA Description PID LWR LIMIT</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.LOCKED</td>
<td>PID_MA Description LOCKED</td>
<td>Not Alarmed</td>
<td>True</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.LOCKPROPERTY</td>
<td>PID_MA Description LOCK PROPERTY</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.LOCKPUSHBUTTON</td>
<td>PID_MA Description LOCK PUSHDOWN</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadWrite</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.LOCKSAFECONDITION</td>
<td>PID_MA Description LOCK SAFE CONDITION</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.LS</td>
<td>PID_MA Description IN LOW LIMIT</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>Global Pin Name</td>
<td>Description</td>
<td>Alarm</td>
<td>Alarm Class</td>
<td>Event</td>
<td>EGD Page</td>
<td>External Access</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
<td>-------</td>
<td>-------------</td>
<td>-------</td>
<td>----------</td>
<td>-----------------</td>
</tr>
<tr>
<td>00PID_MA1000.MA_P</td>
<td>MA_STATION ONLY MODE</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.MA_EXT_P</td>
<td>MA_STATION WITH EXTERNAL/REMOTE SP ONLY</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.MA_MAN_P</td>
<td>MA_STATION WITH MANUAL SP</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.MA_REM_P</td>
<td>MA_STATION WITH REMOTE SP AND LOCAL SP</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.MC</td>
<td>PID_MA Description MANUAL SETPOINT</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadWrite</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.MN_REJ_A</td>
<td>PID_MA Description MANUAL REJECT</td>
<td>Alarm</td>
<td>LVL_4</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.MS</td>
<td>PID_MA Description MANUAL MODE SELECT</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadWrite</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.OBQ</td>
<td>PID_MA Description OUTPUT BQ</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.OVR_CMD</td>
<td>PID_MA Description OVERRIDE COMMAND</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.OVR_CMDB</td>
<td>PID_MA Description OVERRIDE COMMAND BLOCK STS</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.OVR_CTRL_P</td>
<td>PID PART OF OVERRIDE CONTROL SCHEME PROPERTY</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.PG</td>
<td>PID PROPORTIONAL GAIN</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadWrite</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.PID_MA_EXT_C_P</td>
<td>PID CTRL STATION WITH EXTERNAL/REMOTE SP ONLY FROM CASC PID</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.PID_MA_EXT_P</td>
<td>PID CTRL STATION WITH EXTERNAL/REMOTE SP ONLY</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.PID_MA_P</td>
<td>PID CTRL STATION MODE</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.PID_MA_REM_C_P</td>
<td>PID CTRL STATION WITH REMOTE SP FROM CASC PID AND LOCAL SP</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.PID_MA_REM_P</td>
<td>PID CTRL STATION WITH REMOTE SP AND LOCAL SP</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.PID_P</td>
<td>PID ONLY MODE</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.POS</td>
<td>PID_MA Description POSITION FDBK</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.POS_Status</td>
<td>PID_MA Description POSITION FDBK STATUS</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.POS_BQ</td>
<td>PID_MA Description POSITION FDBK BAD QUALITY</td>
<td>Alarmed</td>
<td>LVL_4</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.POS_DH</td>
<td>PID_MA Description POSITION DEV HI ALARM</td>
<td>Alarmed</td>
<td>LVL_4</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.POSFB_P</td>
<td>PID_MA Description POSITION FDBK PROPERTY</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>00PID_MA1000.PR_DEC</td>
<td>PID_MA Description PRIORITY DEC</td>
<td>Not Alarmed</td>
<td>False</td>
<td>$Default</td>
<td>ReadOnly</td>
<td></td>
</tr>
<tr>
<td>Global Pin Name</td>
<td>Description</td>
<td>Alarm</td>
<td>Alarm Class</td>
<td>Event</td>
<td>EGD Page</td>
<td>External Access</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------------------</td>
<td>----------</td>
<td>-------------</td>
<td>-------</td>
<td>----------</td>
<td>-----------------</td>
</tr>
<tr>
<td>00PID_MA1000.PR_INC</td>
<td>PID_MA Description PRIORITY INC</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.PV</td>
<td>PID_MA Description PID PROCESS VARIABLE</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.PV._Status</td>
<td>PID_MA Description PID PROCESS VARIABLE STATUS</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.PV_BQ</td>
<td>PID_MA Description PID PROCESS VARIABLE BQ</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.REJ_MN</td>
<td>PID_MA Description MANUAL REJECT</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.REJ_MNB</td>
<td>PID_MA Description MANUAL REJECT BYPASS STS</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.RS</td>
<td>PID_MA Description REMOTE SETPOINT MODE</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadWrite</td>
</tr>
<tr>
<td>00PID_MA1000.RSP</td>
<td>PID_MA Description REMOTE SETPOINT</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.RSP._Status</td>
<td>PID_MA Description REMOTE SETPOINT STATUS</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.RSP._BQ</td>
<td>PID_MA Description REMOTE SETPOINT BQ</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.RSP._OV</td>
<td>PID_MA Description REMOTE SETPOINT OVERRIDE</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.SC</td>
<td>PID_MA Description SETPOINT</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadWrite</td>
</tr>
<tr>
<td>00PID_MA1000.SH</td>
<td>PID_MA Description SETPOINT UPPER LIMIT</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.SL</td>
<td>PID_MA Description SETPOINT LOWER LIMIT</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.SP</td>
<td>PID_MA Description SETPOINT</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.SP._TRACK</td>
<td>PID_MA Description SETPOINT TRACKING PARAMETER</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.TK</td>
<td>PID_MA Description TRACKING ACTIVE</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>00PID_MA1000.TUN</td>
<td>PID_MA Description TUNING ENABLE</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadWrite</td>
</tr>
<tr>
<td>00PID_MA1000.TV_BQ</td>
<td>PID_MA Description PID TRACK VALUE BQ</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadWrite</td>
</tr>
<tr>
<td>00PID_MA1000.UNLOCK_PB</td>
<td>PID_MA Description UNLOCK PUSHPUSHBUTTON</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadWrite</td>
</tr>
<tr>
<td>00PID_MA1000.US</td>
<td>PID_MA Description IN HIGH LIMIT</td>
<td>Not Alarmed</td>
<td>False</td>
<td></td>
<td>$Default</td>
<td>ReadOnly</td>
</tr>
<tr>
<td>Block_Mode</td>
<td>PID</td>
<td>MA</td>
<td>MA_EXT</td>
<td>MA_MAN</td>
<td>MA Rem</td>
<td>PID_MA</td>
</tr>
<tr>
<td>------------</td>
<td>-----</td>
<td>----</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>AU_SEL</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>BLOCK_MODE</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CTRL_ACT</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>CTRL_TYPE</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>CVI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PV</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RS</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSP</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SC</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SR</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SH</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SL</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERR_C</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERR_CDB</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP_TRACK</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQR_ERR</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PG</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PG_C</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PG_CDB</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IG</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IG_C</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IG_CDB</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INH_PDG</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PV_SP_LIM_TRK</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DG</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SF</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TK</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TV</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ER</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EV</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FF</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INH_L</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INH_R</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PR_L</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PR_R</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PR_RT_L</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PR_RT_R</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Block Mode Visible Pins

### Inputs

<table>
<thead>
<tr>
<th>Block_Mode</th>
<th>PID</th>
<th>MA</th>
<th>MA_EXT</th>
<th>MA_MAN</th>
<th>MA_REM</th>
<th>PID_MA</th>
<th>PID_MA_EXT</th>
<th>PID_MA_EXT_CASC</th>
<th>PID_MA_REM</th>
<th>PID_MA_REM_CASC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| MC         | X   | X   | X       | X       | X       | X       | X          | X               | X          |
| MJ         | X   | X   |         |         |         |         | X          |                 |            |
| MR         | X   | X   | X       | X       | X       | X       | X          |                 | X          |
| MS         | X   | X   | X       | X       | X       | X       | X          |                 | X          |
| OBQ        | X   | X   | X       |         | X       | X       | X          |                 | X          |
| OV         | X   | X   | X       | X       | X       | X       | X          |                 | X          |
| OVR_CTRL   | X   |     |         |         |         |         | X          |                 | X          |
| TUN        | X   |     |         |         |         |         | X          |                 | X          |

### Outputs

<table>
<thead>
<tr>
<th>Block_Mode</th>
<th>PID</th>
<th>MA</th>
<th>MA_EXT</th>
<th>MA_MAN</th>
<th>MA_REM</th>
<th>PID_MA</th>
<th>PID_MA_EXT</th>
<th>PID_MA_EXT_CASC</th>
<th>PID_MA_REM</th>
<th>PID_MA_REM_CASC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| AUTO       | X   | X   | X       | X       | X       | X       | X          | X               | X          |
| SP         | X   | X   | X       | X       | X       | X       | X          |                 | X          |
| LS         | X   | X   | X       | X       | X       | X       | X          |                 | X          |
| LS         | X   | X   | X       | X       | X       | X       | X          |                 | X          |
| MN_REJ_A   | X   | X   | X       | X       | X       | X       | X          |                 | X          |
| TUN_OUT    | X   |     |         |         |         |         | X          |                 | X          |
| TK_OUT     | X   |     |         |         |         |         | X          |                 | X          |
| PID_P      |     | X   |         |         |         |         |            |                 |            |
| MA_EXT_P   |     |     |         |         |         |         |            |                 |            |
| MA_MAN_P   |     | X   |         |         |         |         |            |                 |            |
| MA_P       |     |     |         |         |         |         |            |                 |            |
| MA_REM_P   | X   |     |         |         |         |         |            |                 |            |
| PID_MA_P   |     | X   |         |         |         |         |            |                 |            |
| PID_MA_EXT_P|     |     |         |         |         |         |            |                 |            |
| PID_MA_EXT_C_P|    |   |         |         |         |         |            |                 |            |
| PID_MA_REM_P|     |     |         |         |         |         |            |                 |            |
| PID_MA_REM_C_P|    |   |         |         |         |         |            |                 |            |
| OVR_CTRL_P | X   |     |         |         |         |         | X          |                 | X          |

**Instruction Guide GEI-100681N 75**

**Public Information**
HMI Configuration (CIMPLICITY)

The Object name for the HMI is the same as the Device name specified in the ToolboxST configuration.

Typical HMI Object for a PID_MA

PID_MA Faceplate
Preset Counter (PRESET_CNTR)

**Block Category:** PRESET_CNTR, Legacy

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section *Replacement Blocks* for additional information.

The Preset Counter (PRESET_CNTR) block presets a given pulse counter value to a specified value. When the preset enable input is *True*, the block's output counts value (CNT_OUT) is set to the given preset value. After the preset enable input is released, the output value CNT_OUT either decrease or increase from that point depending on the difference between the PRESET value and CNT_IN value.

### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNT_IN</td>
<td>Double Integer</td>
<td>Signal attached to device counter input point</td>
</tr>
<tr>
<td>PRESET</td>
<td>Double Integer</td>
<td>Preset value in counts</td>
</tr>
<tr>
<td>PRE_EN**</td>
<td>Boolean</td>
<td>Level activating signal enabling a preset operation</td>
</tr>
</tbody>
</table>

** The preset enable (PRE_EN) input is active as a level (as opposed to a rising edge) signal. Therefore, as long as it is *True*, the preset value is maintained on the output counts.

### Output

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNT_OUT</td>
<td>Double Integer</td>
<td>Preset output counts (Raw input counts 'normalized' by the PRESET pin's value)</td>
</tr>
</tbody>
</table>
**Ramp (RAMP)**

Block Category: RAMP, Regulators

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section **Replacement Blocks** for additional information.

The Ramp (RAMP) block produces a current ramp output (CURRAMP) that increments and decrements until the final value (FINAL) is reached. The current value is modified each time by the value specified in the accel_rate (ACCEL) or decel_rate (DECEL). ACCEL and DECEL are in units per second and both must be positive. The current ramp value is compared against the final value and the ACCEL value will be added or the DECEL value will be subtracted to/from the current ramp value depending on the results of the comparison. Once the final value is reached, the logical output (AFTINAL) will be set as *True* and current ramp is clamped to the final value. For example, if the output is desired to ramp to 500 in 25 seconds, set accel_rate at \( \frac{500}{25} = 20 \).

The Ramp block can also operate in Preset mode. When PRESET is set to *True*, the current ramp value is set to the value specified by PR_VAL, and the AFTINAL logical output value is set to *True* to indicate that the value of CURRAMP is at its commanded value. When PRESET is set back to *False*, AFTINAL output is set to *False* and CURRAMP begins ramping from the PRESET value to the FINAL value. AFTINAL output is set to *True* when the FINAL value is reached.

The Ramp block can also operate in Bypass mode. When Quick Pass (QUICKPS) is set to *True*, the current ramp value is set to the value specified by FINAL, and the AFTINAL logical output value is set to *True* to indicate that the value of CURRAMP is at its commanded (final) value.

![RAMP Block Diagram](image-url)
### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>Boolean</td>
<td>Enable the block to run</td>
</tr>
<tr>
<td>ACCEL</td>
<td>Real</td>
<td>Ramp acceleration rate in units/sec</td>
</tr>
<tr>
<td>FINAL</td>
<td>Real</td>
<td>Ramp Final (target) Value</td>
</tr>
<tr>
<td>PR_VAL</td>
<td>Real</td>
<td>Preset Value of Ramp output</td>
</tr>
<tr>
<td>PRESET</td>
<td>Boolean</td>
<td>Preset Ramp command</td>
</tr>
<tr>
<td>QUIKPS</td>
<td>Boolean</td>
<td>Bypass Ramp command</td>
</tr>
<tr>
<td>DECEL</td>
<td>Real</td>
<td>Ramp deceleration rate in units/sec</td>
</tr>
</tbody>
</table>

**Note** Resolution errors can occur when adding large and small REAL numbers. The RAMP block uses REAL variables, which are IEEE single-precision floating-point numbers. Values should be scaled properly to avoid resolution issues. For example, ramping over a range of 180 in 2 minutes is done with ACCEL set to 1.5, and this results in an increment of 0.015 per 10 ms frame. Ramping from 0 to 180 has less errors than ramping from 10000 to 10180 due to the floating point co-processor.

### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CURRAMP</td>
<td>Real</td>
<td>Current output, ramped</td>
</tr>
<tr>
<td>ATFINAL</td>
<td>Boolean</td>
<td>Ramp status (set to True when CURRAMP = FINAL)</td>
</tr>
</tbody>
</table>
**Ramp with Hold (RAMPH)**

**Block Category:** RAMPH, Regulators

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information.

The Ramp with Hold (RAMPH) block produces a current ramp output that increments and decrements until the final value is reached. The current value is modified each time by the value specified in the accel_rate (ACCRATE) or decel_rate (DECRATE). The modified value is returned by this procedure. The return value is set to PR_VAL when the preset logical is True. The logical output at_final (FINAL) will be set True when the final value is reached.

Both the ascending rate and the descending rate must be positive. The current ramp value is compared against the final value and the ascending rate value is added to, or the descending rate value is subtracted from the current ramp value, depending on the results of the comparison. Once the final value is reached, the ramp status (ATFINAL) is set to True and the current ramp is clamped to the final value. The current ramp value is set to the preset value (PR_VAL) and the ramp status is set to True when the preset ramp command (PRESET) is True. The current ramp is held at its current value when the hold ramp command (HOLD) is True. The current ramp is set to the final value and the ramp status is set True when the bypass command (PASS) is True. The ascending rate and the descending rate are in units per second. For example, if the output is desired to ramp up to 500 in 25 sec, set the ascending rate to 20 (500/25).
### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>Boolean</td>
<td>Enable the block to run</td>
</tr>
<tr>
<td>ACCRATE</td>
<td>Real</td>
<td>Ramp ascending rate in units/sec</td>
</tr>
<tr>
<td>FINAL</td>
<td>Real</td>
<td>Ramp final (target) Value</td>
</tr>
<tr>
<td>PR_VAL</td>
<td>Real</td>
<td>Preset value of ramp output</td>
</tr>
<tr>
<td>PRESET</td>
<td>Boolean</td>
<td>Preset ramp command</td>
</tr>
<tr>
<td>HOLD</td>
<td>Boolean</td>
<td>Hold ramp output command</td>
</tr>
<tr>
<td>PASS</td>
<td>Boolean</td>
<td>Bypass ramp command</td>
</tr>
<tr>
<td>DECRATE</td>
<td>Real</td>
<td>Ramp descending rate in units/sec</td>
</tr>
</tbody>
</table>

### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CURRAMP</td>
<td>Real</td>
<td>Current output, ramped</td>
</tr>
<tr>
<td>ATFINAL</td>
<td>Boolean</td>
<td>Ramp status, True when CURRAMP=FINAL or when CURRAMP = PR_VAL and PRESET is True</td>
</tr>
</tbody>
</table>
**Split DS (SPLIT_DS)**

**Block Category:** Type Conversion

**Legacy Status** — Not a legacy block.

The Split blocks break apart large variables into smaller variables. They perform the reverse function of the Combine blocks. The SPLIT_DS block is used to split an unsigned double integer (32-bit) to two 16-bit WORD values. The splitting of the signals is strictly mechanical. That is, there is no attempt to interpret the data, or to perform any clamping, scaling, data type conversion, or sign extension.

The DATA_SWAP input on the block is used to decide whether a byte swap is required on the output words. If the DATA_SWAP input is set, then the individual bytes in a single WORD output is swapped.

The reverse of the SPLIT_DS block is the COMBINE_SD block.

---

**SPLIT_DS Block**

**Inputs**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>BOOL</td>
<td>Block enable (default is True)</td>
</tr>
<tr>
<td>INPUT</td>
<td>UDINT</td>
<td>Input value to split</td>
</tr>
<tr>
<td>DATA_SWAP</td>
<td>DataSwap_T</td>
<td>Byte swap input for the individual outputs: NO_SWAP (0) or BYTE_SWAP (1) (default is NO_SWAP)</td>
</tr>
</tbody>
</table>

**Outputs**

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOWORD</td>
<td>UINT</td>
<td>Lower order 16 bits of the input variable</td>
</tr>
<tr>
<td>HIWORD</td>
<td>UINT</td>
<td>Higher order 16 bits of the input variable</td>
</tr>
</tbody>
</table>
Split LRS (SPLIT_LRS)

Block Category: Type Conversion

Legacy Status — Not a legacy block.

The Split blocks break apart large variables into smaller variables. They perform the reverse function of the Combine blocks. The SPLIT_LRS block is used to split a long real (64-bit) to four 16-bit WORD values. The splitting of the signals is strictly mechanical. That is, there is no attempt to interpret the data, or to perform any clamping, scaling, data type conversion, or sign extension.

The DATA_SWAP input on the block is used to decide whether a byte swap is required on the output words. If the DATA_SWAP input is set, then the individual bytes in a single WORD output is swapped.

The reverse of the SPLIT_LRS block is the COMBINE_SLR block.

### SPLIT_LRS Block

#### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>BOOL</td>
<td>Block enable (default is True)</td>
</tr>
<tr>
<td>INPUT</td>
<td>LREAL</td>
<td>Input value to split</td>
</tr>
<tr>
<td>DATA_SWAP</td>
<td>DataSwap_T</td>
<td>Byte swap input for the individual outputs: NO_SWAP (0) or BYTE_SWAP (1) (default is NO_SWAP)</td>
</tr>
</tbody>
</table>

#### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORD0</td>
<td>UINT</td>
<td>Lowest order 16 bits of the input variable</td>
</tr>
<tr>
<td>WORD1</td>
<td>UINT</td>
<td>Bits 16 to 31 of the input variable</td>
</tr>
<tr>
<td>WORD2</td>
<td>UINT</td>
<td>Bits 32 to 47 of the input variable</td>
</tr>
<tr>
<td>WORD3</td>
<td>UINT</td>
<td>Higher order 16 bits of the input variable</td>
</tr>
</tbody>
</table>
Split RS (SPLIT_RS)

Block Category: Type Conversion

Legacy Status — Not a legacy block.

The Split blocks break apart large variables into smaller variables. They perform the reverse function of the Combine blocks. The SPLIT_RS block is used to split a real (32-bit) to two 16-bit WORD values. The splitting of the signals is strictly mechanical. That is, there is no attempt to interpret the data, or to perform any clamping, scaling, data type conversion, or sign extension.

The DATA_SWAP input on the block is used to decide whether a byte swap is required on the output words. If the DATA_SWAP input is set, then the individual bytes in a single WORD output is swapped.

The reverse of the SPLIT_RS block is the COMBINE_SR block.

### SPLIT_RS Block

#### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>BOOL</td>
<td>Block enable (default is True)</td>
</tr>
<tr>
<td>INPUT</td>
<td>REAL</td>
<td>Input value to split</td>
</tr>
<tr>
<td>DATA_SWAP</td>
<td>DataSwap_T</td>
<td>Byte swap input for the individual outputs: NO_SWAP (0) or BYTE_SWAP (1) (default is NO_SWAP)</td>
</tr>
</tbody>
</table>

#### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOWORD</td>
<td>UINT</td>
<td>Lower order 16 bits of the input variable</td>
</tr>
<tr>
<td>HIWORD</td>
<td>UINT</td>
<td>Higher order 16 bits of the input variable</td>
</tr>
</tbody>
</table>
Split SDS (SPLIT_SDS)

Block Category: Type Conversion

Legacy Status — Not a legacy block.

The Split blocks break apart large variables into smaller variables. They perform the reverse function of the Combine blocks. The SPLIT_SDS block is used to split a signed double integer (32-bit) to two 16-bit WORD values. The splitting of the signals is strictly mechanical. That is, there is no attempt to interpret the data, or to perform any clamping, scaling, data type conversion, or sign extension.

The DATA_SWAP input on the block is used to decide whether a byte swap is required on the output words. If the DATA_SWAP input is set, then the individual bytes in a single WORD output is swapped.

The reverse of the SPLIT_SDS block is the COMBINE_SSD block.

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>BOOL</td>
<td>Block enable (default is True)</td>
</tr>
<tr>
<td>INPUT</td>
<td>DINT</td>
<td>Input value to split</td>
</tr>
<tr>
<td>DATA_SWAP</td>
<td>DataSwap_T</td>
<td>Byte swap input for the individual outputs: NO_SWAP (0) or BYTE_SWAP (1) (default is NO_SWAP)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOWORD</td>
<td>UINT</td>
<td>Lower order 16 bits of the input variable</td>
</tr>
<tr>
<td>HIWORD</td>
<td>UINT</td>
<td>Higher order 16 bits of the input variable</td>
</tr>
</tbody>
</table>
**Timer Unsigned Short (TIMER_UI)**

**Block Category:** TIMER_UI, Timers and Counters

**Legacy Status** — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section *Replacement Blocks* for additional information.

The Timer Unsigned Short (TIMER_UI) block accumulates incremental time into CURTIME while RUN is True. When CURTIME is equal to MAXTIME, AT_TIME transitions to True. If RUN is False then timing is suspended, but the value in CURTIME remains. If RESET is True then CURTIME is set to zero and counting is suspended. The block continuously counts to the MAXTIME value and reset if the AUTO_RS flag is set. The maximum value that MAXTIME parameter can take is 65535 in milliseconds.

The block diagram shows the inputs and outputs of the TIMER_UI block.

### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXTIME</td>
<td>Unsigned Integer</td>
<td>Maximum time value in milliseconds</td>
</tr>
<tr>
<td>RESET</td>
<td>Boolean</td>
<td>Re-zeros the timer</td>
</tr>
<tr>
<td>AUTO_RS</td>
<td>Boolean</td>
<td>Automatically zeros the timer when AT_TIME is set. Maintains residual count. The default value is False</td>
</tr>
<tr>
<td>RUN</td>
<td>Boolean</td>
<td>Enables the timer</td>
</tr>
</tbody>
</table>

### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT_TIME</td>
<td>Boolean</td>
<td>Indicates whether the timer has timed out</td>
</tr>
<tr>
<td>CURTIME</td>
<td>Unsigned Integer</td>
<td>The current time accumulated</td>
</tr>
</tbody>
</table>
Word to Analog Conversion (WD_TO_ANA)

Block Category: WD_TO_ANA, Type Conversion

Legacy Status — this block is not recommended for use in any new controller applications (ControlST V05.02 or later). Refer to the section Replacement Blocks for additional information.

The Word to Analog Conversion (WD_TO_ANA) block is a **rubber block** that transfers the values on 1 to 32 input pins to corresponding output pins, where the inputs are unsigned short integer data types and the outputs are analog types.

---

### WD_TO_ANA Block

#### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>Boolean</td>
<td>Enables block execution</td>
</tr>
<tr>
<td>IN1</td>
<td>Unsigned Integer</td>
<td>Unsigned input value 1</td>
</tr>
<tr>
<td>↓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IN32</td>
<td>Unsigned Integer</td>
<td>Unsigned input value 32</td>
</tr>
</tbody>
</table>

#### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT1</td>
<td>Numeric</td>
<td>Converted output value 1</td>
</tr>
<tr>
<td>↓</td>
<td>↓</td>
<td>↓</td>
</tr>
<tr>
<td>OUT32</td>
<td>Numeric</td>
<td>Converted output value 32</td>
</tr>
</tbody>
</table>

---

*Public Information*
**Word to Integer Conversion (WD_TO_I)**

Block Category: Type Conversion

**Legacy Status** — Not a legacy block.

The Word to Integer Conversion (WD_TO_I) block moves a 16-bit WORD to a signed integer (16-bit). The conversion is strictly mechanical. That is, there is no attempt to interpret the data, or to perform any clamping, scaling, data type conversion, or sign extension.

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>BOOL</td>
<td>Block enable (default is True)</td>
</tr>
<tr>
<td>INPUT</td>
<td>UINT</td>
<td>16-bit Word input</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>INT</td>
<td>Integer output</td>
</tr>
</tbody>
</table>