

You are here: [IoT Gateway User Guide](#) > [Projects and triggers](#) > [Trigger actions reference](#) > Enhanced Demand Write

Topic updated on September 21, 2023

Enhanced Demand Write

The **Enhanced Demand Write** action writes a group of device variables at that point in a trigger's execution, bypassing the internal trigger buffer used for device variables that is normally written when the trigger ends execution.

When a trigger starts its execution, all variables referenced by the trigger's actions are read into internal buffers. The variables' values are read from or written to this internal buffer when the variables are referenced by any of the trigger's actions. The internal buffer is written to the devices when the trigger completes its execution.

To bypass this internal buffering, the **Enhanced Demand Write** action forces a write of the device variables through the device manager and device driver to the device.

4.Enhanced Demand Write

Rules In JSON:

```

1  {
2    "TestStringVar1" : { "type":"STRING", "count":1, "length":10 },
3    "TestCountVar"   : { "type":"INT4", "count":5 },
4    "TestTemp"       : { "type":"FLOAT8", "count":1 }
5  }

```

Input

Routing

Details

Input

Name	Logical	Count	Value	Type
Device Name	STRING(128)	1	Logix_1_67	CONSTANT
TestStringVar 1 Name	STRING(128)	1	StrArray2048[0]	CONSTANT
TestStringVar 1	STRING(10)	1	LocalVariables.StringVar 1	STRING(10)
TestCountVar Name	STRING(128)	1	H1500_CycleData.NumPassed	CONSTANT
TestCountVar	INT4	5	LocalVariables.CountVar [0]	INT4
TestTemp Name	STRING(128)	1	REALscalar 1	CONSTANT
TestTemp	FLOAT8	1	LocalVariables.Temp	FLOAT8

Example

```


{
  "TestStringVar1": {"type":"STRING", "count":1, "length": 10},
  "TestCountVar": {"type":"INT4", "count":5},
  "TestTemp": {"type":"FLOAT8", "count":1}
}

```

Tip

The Enhanced Demand Write action is part of the Advanced Features package. For more information on obtaining and installing the package, see [Packages](#).

Parameter descriptions

Parameter	Description
Rules in JSON	<p>The description of the device variables to write. Each variable is specified as shown in the example with:</p> <ul style="list-style-type: none"> Map variable name - The name of the map variable, which will be added as two rows on the Input tab. <ul style="list-style-type: none"> The first row on the Input tab with the "Name" suffix is used to specify the destination device variable. The second row on the Input tab is used to specify the source variable who's value will be written to the destination variable. type - The data type of the variable to write. This type must match the data type of the destination variable. The supported types are: BOOL, INT1, INT2, INT4, INT8, UINT1, UINT2, UINT4, UINT8, FLOAT4, FLOAT8, STRING, TIMESTAMP, and BINARY. For the Mitsubishi MESInterface IT product: BIT, BYTE, WORD, DWORD, LWORD, UBYTE, UWORD, UDWORD, ULWORD, REAL, LREAL, STRING, TIMESTAMP and BINARY. count - The number of variable elements. Specify a 1 for a single (scalar) element or a value greater than 1 for an array. length - For types STRING and BINARY, the length of the variable element. The multi-line input icon  can be used to display a larger input area for the JSON variable description.

Input tab

The **Input** tab has the device name parameter and two rows for each variable identified in the JSON variable description.

Parameter	Description
Device Name	The name of the device containing the variables whose values you want to write.
Variables	<p>Each variable in the JSON variable description will be added as two rows on the Input tab.</p> <ul style="list-style-type: none"> The first row with the "Name" suffix is used to specify the destination device variable.

Parameter**Description**

- The second row is used to specify the source variable whose value will be written to the destination variable.

The source variables can be trigger variables (local or static) or device variables.

In the example input tab above, the destination device variables names are specified with constants.

The source variables are trigger local variables.

Enhanced Demand Write considerations

When a trigger action writes to a device variable, the updated value is held in an internal buffer until the trigger completes its execution.

When a trigger starts its execution, all variables referenced by the trigger's actions are read into internal buffers. The variables' values are read from or written to this internal buffer when the variables are referenced by any of the trigger's actions. The internal buffer is written to the devices when the trigger completes its execution. This process maximizes performance of the trigger execution.

In cases where the device variables need to be written directly to the device at a certain point in the trigger's execution and not just to the internal trigger buffers, the **Enhanced Demand Write** action is used. These cases need to take into account the overall application logic in the triggers and in the devices to understand the dynamic interaction of all parts of the application. The **Enhanced Demand Write** action does take additional resources in terms of processing time (compared to for example a **Set** action), since the trigger engine and device driver must communicate down to the physical device and get a response. This includes the communication across the network to the device.

The data type specified for each variable must match the data type of the destination device variable. If it does not match, that variable will not be successfully written.

The action will fail if there is a problem with the device specified (not defined or not started). The action will succeed if there are individual problems with the variables specified. The variables that had individual problems (variable not defined, data type mismatch, data overflow, data cast problem) will not update the corresponding device variable. The variables that did not have a problem will update the corresponding device variables.

[Related Topics](#)

[About Telit](#) | [Contact Us](#) | [Legal Notices](#) | [Terms of Service](#) | [Privacy Policy](#)

Copyright © 2025, Telit IoT Solutions Holding Ltd.. All rights reserved.