



Hyper Historian

Performance Calculations Value Array Functions



APPLICATION NOTE

September 2013

Description: A quick guide to start using the Hyper Historian Array Performance Calculation Functions

OS Requirement: Vista x64/ Windows 7 x64/ Windows 8 x64/ Windows Server 2008 x64/ Windows Server 2008 R2 x64/ Windows Server 2012

General Requirement: Installation of GENESIS64 v10.80 and basic knowledge of Hyper Historian and GENESIS64

Introduction

ICONICS Hyper Historian is a powerful, 64-bit, high performance plant historian. The Hyper Historian provides many features and functions for most mission critical applications demanding the highest level of availability and provides the highest performance possible by taking advantage of 64-bit hardware and software architectures, allowing it to access more CPU power and memory than traditional 32-bit-based historians.

One of the key features of the Hyper Historian is to support extensive data calculation on live and historical data using the new Performance Calculation engine.

This application note will show how to configure Hyper Historian performance calculations using the new Array Calculation Functions introduced in v10.80.

ATagNowX Functions

The Performance Calculations are executed against raw data that is logged by the Hyper Historian. So for anything that needs a performance calculation the standard logging tag(s) need to be configured first.

In Hyper Historian v10.80 a number of atagnowX functions were introduced which creates an array of the most recent tag values for between 3-6 tags depending on the value of X.

1. In the Workbench open the Hyper Historian provider, add a new Calculation Function by right-clicking on **Calculation Functions** and selecting "Calculation Function"
2. Set the name of the newly created function to be "CreateArray3"
3. Set the **CreateArray3** function returns to "A value array"
4. Add three parameters of type "variable" with unique names (e.g. Variable1...3)
5. On the Expression tab add the **atagnow3()** function by selecting Functions > Array Calculation Functions > atagnow3(variable, variable, variable)
6. The three instances of the "variable" text in red will need to be replaced with previously created parameters, by

highlighting the text "variable", clicking the **Variables...** button and browsing to a different variable each time

7. Apply the changes to allow it to be used in future Calculation Tags

This completes the configuration of the ATagNow3() calculation functions.

Statistical Array Functions

In Hyper Historian v10.80 a number of statistical array functions were added to allow the returning of a single sample from an array based on a specific function.

Average Function

When using an array of numerical values (integer or floating point values) it is possible to use the aavg() function to return the average of the contents.

1. After completing the atagnow3() section create a new Calculated Tag, by right-clicking on **Data Collections** and selecting "Calculated Tag"
2. Set the name of the newly created tag to be "AverageTag"
3. Configure the **AverageTag** to use the default Period 10s trigger created with the samples database, by selecting the ... button in the "Triggers", browsing and selecting the Calculation Triggers > Periodic > Period 10s trigger
4. On the Expression tab browse to the previously created function, by selecting Functions > Calculation Functions > CreateArray3
5. Replace all three instances of the red text with a different Hyper Historian tag, by highlighting it and selecting the **Variables...** button, from the variable browser select a different tag each time
6. Wrap the entire function within the aavg() function, by selecting the entire function and cutting it (CTRL+X), then add the Array Average function by selecting Functions > Array Calculation Functions > aavg(array)
7. Select the red "array" text and paste in the previously cut text, to leave the function as shown in

```
aavg(CreateArray3(((data:LogGrp.OPCUA.Ramp)), ((data:LogGrp.OPCUA.Random)), ((data:LogGrp.OPCUA.Sine))))
```

Figure 1 - Array Average

This completes the Array Average function, starting the Hyper Historian engine and leaving it running will generate a new average value every 10 seconds providing the underlying data is changing.