Hyper Historian – Performance Calculations

APPLICATION NOTE

July 2012

Description: A quick guide to start using Hyper Historian Performance Calculation.
OS Requirement: Windows Server 2003 x64/Vista x64/ Server 2008 x64/Windows 7 x64/ Server 2008 R2 x64
General Requirement: Installation of GENESIS64 v10.70 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 the configure Hyper Historian Performance Calculations for a SPC style $\bar{x}$ and R-chart.

The Raw Data Tag

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. To create the calculation for the $\bar{x}$ and R-chart we need a single logging value (the x-value).

The Hyper Historian will automatically make any value that is being logged available for data replay. For this purpose all historical logging tags are also exposed as OPC Data Access tags.

1. In the Workbench open the Hyper Historian provider, add a new Logging Group by right-clicking on **Groups** and choosing "Logging Group"
2. Set the name of the new Logging Group to "PerfCalc" and choose one of the available File Loggers on your system.
3. Under the newly created Logging Group create a new Collector Group by right-clicking on the **PerfCalc** Logging Group and choosing "Collector Group"
4. Set the name of the new Collector Group to "Data" and choose one of the available Collectors on your system.
5. Create new data collections tag by right clicking on the **Data Collections** and choosing Tag
6. Set the name of the new tag to "Source" and the data type to Float32
7. Click on the ellipsis button next to the Signal Name field to open the Data Browser and select:
   Home > OPC Classic Server > My Server > Simulations > Float > Random, then click in the signal name part of the data browser and change the period to 30, the minimum to 30, the maximum to 50, and the phase to 0 as shown in Figure 1.

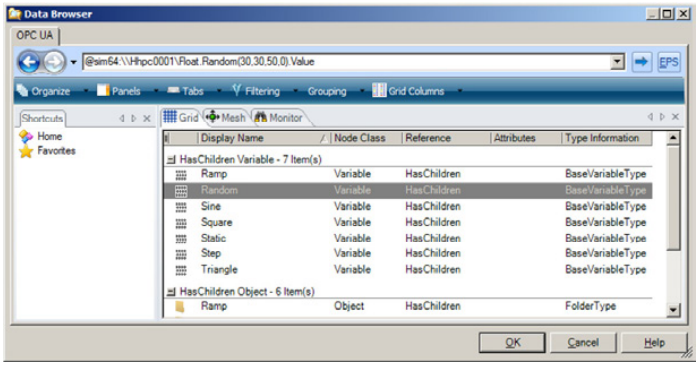

Figure 1 Data Browser

8. Change the **In Group** property of the tag to PerfCalc \ Data.

This completes the configuration of the base (raw data) tag that can then be used in the calculation tags.

The Counters

In SPC calculations it is important to know what samples and measurements you want to include in the calculations.
For that purpose we will create two calculated tags, one to count measurements, i.e. every change of the source value increments the counter and which will reset after a set number of measurements have been received in this case the reset will occur after 5 measurements. The other counter will count the number of samples, which will be incremented every time the 5 measurements have been taken.
As calculation tags are (re-)calculated on a trigger as part of the counter configuration the triggers will also be defined.

Measurement Data Trigger

1. Right click on the **Calculation Triggers** folder within the Hyper Historian Provider and choose "Data Trigger"
2. Change the name to Measurement and set the trigger type to "Any Data Point Change"
3. Click on the ellipsis button next to the Data Point field to open the tag browser as shown in Figure 2.
4. Select the Source tag

Copyright 2012 ICONICS, Inc.

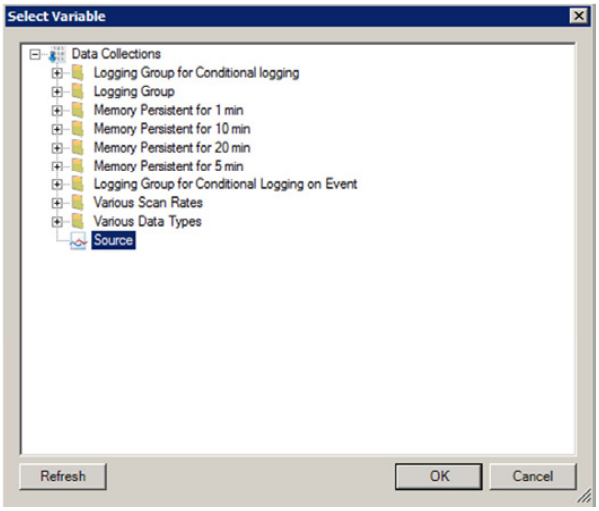

Figure 2 Tag Browser

Measurement Counter

With the trigger created the measurement counter calculation tag can be created

1. Right click on the **Data Collections** tree item and choose Calculated Tag
2. Set the name to "MeasurementCounter"
3. Click in the empty box under the Name column in the Triggers section, and type the letter "D", this will be the name of the trigger, the name of the trigger is useful when different parts of the expression will have to run depending on which trigger caused the calculated tag to recalculate
4. Click on the button for the Trigger path selection to open the trigger browser as shown in Figure 3.

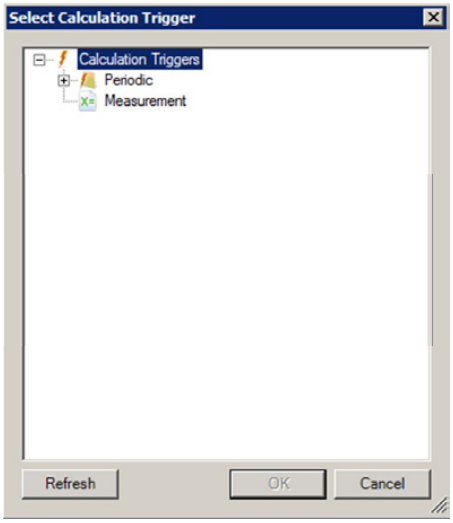

Figure 3 Select the Measurement Trigger

5. Select the previously created Measurement trigger.
6. Change the **In Group** property of the tag to PerfCalc.
7. Click on the Expression tab and enter the following expression (only start a newline after the ↵)

```
IF isgood(trgcur())↵
THEN↵
   IF isgood(tagprev({{calc:MeasurementCounter}}, now()))↵
   THEN (val(tagprev({{calc:MeasurementCounter}}, now()))
+ 1) % 5↵
   ELSE 1↵
ELSE nologval()↵
```

The first IF statement checks if the received update is of good quality, if not no value will be logged. The second IF statement (only used for good values) will check if the previously logged value was good, and if so, it will add 1 to the running total and by using the modulus function (%) it will reset every 5 values, if no previous value can be found it will start with a value of 1.
The *trgcur*() function returns an data object with the name, value, and quality of the trigger.
The *isgood*() function returns true or false depending on the quality of the specified data object.
The *tagprev*() function takes two parameters, the first is the name of the tag, and the second is a time value, the function will find the previous value before the specified time.
The *nologval*() function is a special instruction to Hyper Historian to ensure it does not log a new value.

Sample Data Trigger

With the measurement counter in place, the trigger for the sample counter can be implemented. The trigger will execute only when new measurements are received

1. Right click on the **Calculation Triggers** folder within the Hyper Historian Provider and choose "Data Trigger"
2. Change the name to Sample and set the trigger type to "Any Data Point Change"
3. Click on the ellipsis button next to the Data Point field to open the tag browser as shown in Figure 4.
4. Select the MeasurementCounter tag from the tree

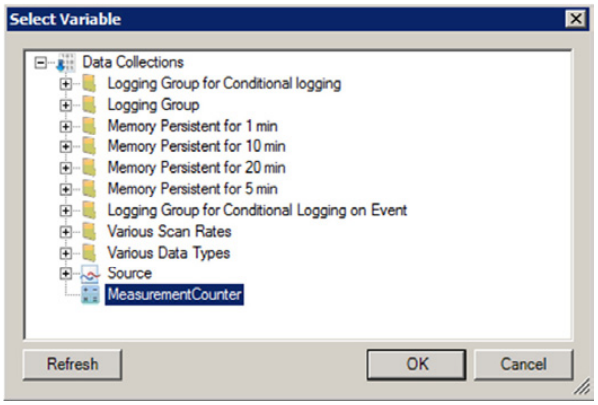

Figure 4 Select the Measurement Counter

Sample Counter

With the trigger created the sample counter calculation tag can be created

1. Right click on the **Data Collections** tree item and choose Calculated Tag
2. Set the name to "SampleCounter"
3. Click in the empty box under the Name column in the Triggers section, and type the letter "D"
4. Click on the button for the Trigger selection to open the trigger browser as shown in Figure 5.

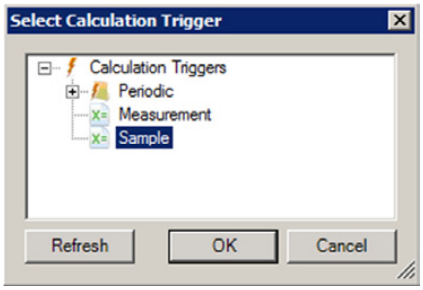

Figure 5 Select the Sample Trigger

5. Select the previously created Sample trigger.
6. Change the **In Group** property of the tag to PerfCalc.
7. Click on the Expression tab and enter the following expression (only start a newline after the ⏎)

```
IF isgood(trgcur()) && (val(trgcur()) == 0) ⏎
THEN⏎
   IF⏎ isgood(tagprev({{calc:SampleCounter}}, now()))⏎
   THEN⏎ (val(tagprev({{calc:SampleCounter}}, now())) + 1)
⏎
   ELSE 1⏎
ELSE nologval()⏎
```

The first IF statement checks if the received update is of good quality and the value is zero, which means that 5 measurements have been taken, if not no value will be logged. The second IF statement (only used for good values) will check if the previously logged value was good, and if so, it will add 1 to the running total, alternatively if no previous value can be found it will start with a value of 1.

SPC Data Calculation

With the measurement and sample counters in place, the SPC calculation can be implemented.

1. Right click on the **Data Collections** tree item and choose Calculated Tag
2. Set the name to "xbar"
3. Click in the empty box under the Name column in the Triggers section, and type the letter "D"
4. Click on the button for the Trigger selection to open the trigger browser as shown in Figure 6.

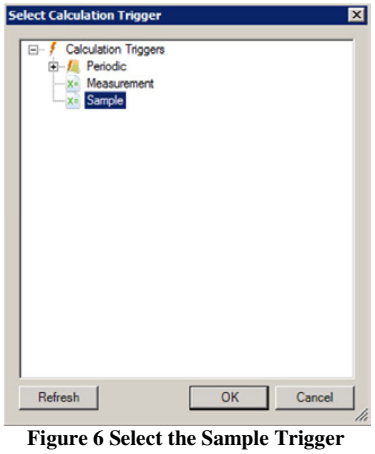

Figure 6 Select the Sample Trigger

5. Select the previously created Sample trigger.
6. Change the **In Group** property of the tag to PerfCalc.
7. Click on the Expression tab and enter the following expression (only start a newline after the ⏎)

```
IF isgood(trgcur()) && (val(trgcur()) == 0) ⏎
THEN⏎
   tagavg({{data:source}}, time(tagprev({{calc:xbar}},
now())), now(), 80) ⏎
ELSE nologval()⏎
```

The IF statement checks if the received update is of good quality and the value is zero, which means that 5 measurements have been taken, if not no value will be logged. If a successful value
The *tagavg*() function calculates the average value over a given time range, it requires four parameters, a tag name, the start time, the end time, and a confidence value. In this example the start time is determined by using the time of the last logged sample. The end time is set to the current time, and the confidence value is set to 80, which means that at least 80% of the values in the time period must be good quality for the average value to have a good quality.
The *time*() function returns the actual time of the specified data object.