Formula registers and remote devices

Physical registers represent any single data point recorded by an eGauge meter. These can come from locally-obtained raw measurements (eg, amperage or voltage), locally obtained calculations (eg, real power and other power register subtypes), and even registers imported from remote devices (remote eGauge or third party devices connected via Modbus TCP/RTU).

Formula registers are a type of physical register which can be used for a variety of calculations. Typically, a formula register will read data from one or more physical registers on a device, perform some calculation, and then store a new value. This is commonly used for advanced calculations such as power factor or reactive power.

However, when using a formula register with a remote register, some additional consideration must be taken to avoid incorrect values when the remote register is not available (eg, the remote device has been disconnected from the network, dropped offline, etc). When this happens, the remote register may return a "NaN" value (Not a Number) instead of a numerical value, which can "break" the formula register. Consider the simple example below:

In this example, a formula register is used to add together the physical registers "Local_Power" and "Remote_Power". "Local_Power" is a power register measured by the meter itself, while "Remote_Power" is a power register imported via Modbus from an external source.

Normally, this is simple addition: if "Local_Power" = 10 and "Remote_Power" = 20, the value of "Remote and Local Power" would be 30. However, if the remote device which provides a value for "Remote_Power" goes offline, the eGauge is now reading 10 + NaN, which returns NaN. This behavior isn't desirable - it would be much better to at least return the value of "Local_Power" (10).

To achieve this, we can use a **conditional** and a **function** to convert any returned "NaN" into a 0.

A **conditional** is expressed as X?Y:Z, where **X** is something that evaluates to True or False, **Y** is the value returned if **X** is true, and **Z** is the value returned if **X** is false. For example, (3=4)?0:1 would return 1, because 3=4 is false (3 does not equal 4).

The **function** isnan() takes one value in the parenthesis and returns a "True" if the value is NaN and "False" if the value is a number. For example, isnan(1) would return "False", because 1 is **not** a NaN. isnan(sqrt(-1)) would return "True", because sqrt(-1) (square root of -1) isn't a number (and is

therefore NaN).

As you can see, isnan() will return either a "True" or "False". We can feed that into our conditional and dictate which value is returned. The basic form for this is:

```
isnan($"registername")?0:$"registername"
```

If the value of the register "registername" is a NaN, isnan() returns "True" and the conditional returns 0. If the value of the register "registername" is **not** a NaN, isnan() returns "False" and the conditional returns the value of "registername".

Using our original example:

Please visit kb.egauge.net for the most up-to-date documentation.