Echo Metrics Requirements

# High Level Functions

* Conditional
  + Nested If / else
* Logical
  + AND (&&)
  + OR (||)
* Comparisons
  + ==
  + !=
  + <
  + >
  + <=
  + >=
* Bit operations
  + &
  + |
  + ^
  + ~
  + <<
  + >>
* Bit Packing
  + Bit Packing
* Math operations
  + +
  + –
  + \*
  + /
* Aggregations
  + Sum
  + Average
  + Min
  + Max

# Use ­­Cases

## Inverting flags

{

    "uri": "/components/dio/door\_open",

    "inputs": {

        "hw\_signal": {"uri": "/components/ess\_1", "id": "echo\_1" }

    },

    "expression": "~[hw\_signal]",

    "outputTypeCast": "bool",

    "initialValue": 0

}

## Averaging Values

{

    "uri": "/metrics/ess/soh",

    "inputs": {

        "ess\_01": {"uri": "/components/ess\_1", "id": "soh"},

        "ess\_02": {"uri": "/components/ess\_2", "id": "soh"},

        "ess\_03": {"uri": "/components/ess\_3", "id": "soh"},

        "ess\_04": {"uri": "/components/ess\_4", "id": "soh"}

    },

    "expression": "average([ess\_01], [ess\_02], [ess\_03], [ess\_04])",

    "outputTypeCast": "float",

    "initialValue": 0

}

## Bucketing Alarms

{

    "uri": "/metrics/ess/bms\_fault",

    "input\_template": {

        "ess\_num": {"start": 1, "stop": 4, "format": "%2d"}

    },

    "inputs": {

        "ess\_{ess\_num}\_fault": {"uri": "/components/ess\_{ess\_num}", "id": "bms\_faults"},

        "ess\_{ess\_num}\_other\_fault": {"uri": "/components/ess\_{ess\_num}", "id": "bms\_other\_faults"}

    },

    "expression": "max([ess\_\*.]) > 0",

    "outputTypeCast": "bool",

    "initialValue": false

}

\* Regular expression matching is only allowed in aggregate functions. Matching inputs should be expanded for aggregates and evaluated.

## Bit Packing

{

    "uri": "/metrics/ess/custom\_alarm",

    "inputs": {

        "alarm\_1" : {"uri": "/components/ess\_01", "id": "alarm\_1" },

        "alarm\_2" : {"uri": "/components/ess\_01", "id": "alarm\_2" }

    },

    "expression": "bit\_pack(0:[alarm\_1]:true, 6:[alarm\_2]:false)",

    "outputTypeCast": "Integer",

    "initialValue": 0

}

\* bit\_pack returns an integer packed with the parameters passed to it with format <shift:value:invert>

## Conditionals

{

    "uri": "/metrics/ess/on",

    "inputs": {

        "control\_1" : {"uri": "/components/ess\_01", "id": "c1" },

        "control\_2" : {"uri": "/components/ess\_01", "id": "c2" }

    },

    "expression": "if [control\_1] AND [control\_2] { 10 } else { 20 }",

    "outputTypeCast": "Integer",

    "initialValue": 0

}

## Debug Functions

The generated URI must have an attribute called “expression” that provides transparency to the expression evaluation by providing the expression with each parameter/function evaluated in a cell to its right, such as:

"expression": "~[hw\_signal:true]"

"expression": "average([ess\_01:25], [ess\_02:23], [ess\_03:27], [ess\_04:25]):25"

"expression": "bit\_pack(0:[alarm\_1:false]:true, 6:[alarm\_2:true]:false):65"

"expression": "if [control\_1:true] AND [control\_2:true] { 10 } else { 20 }",