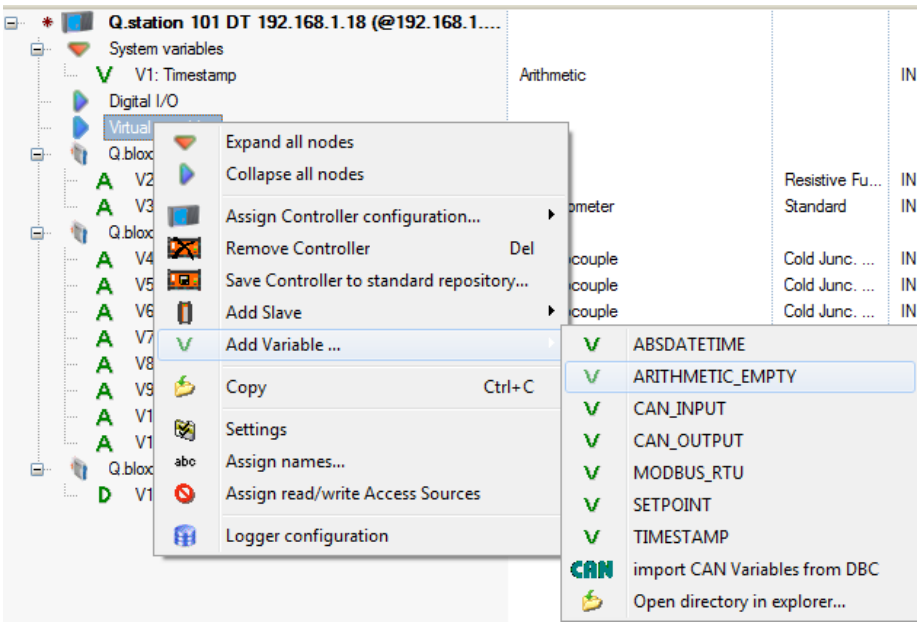
Q.uick Start Guide: Q.station Arithmetic Channels – Functions and Operators Library

Purpose: This guide is essentially a library of all the available functions and operations that can be created as an arithmetic channel under the virtual variables section of a Q.station.

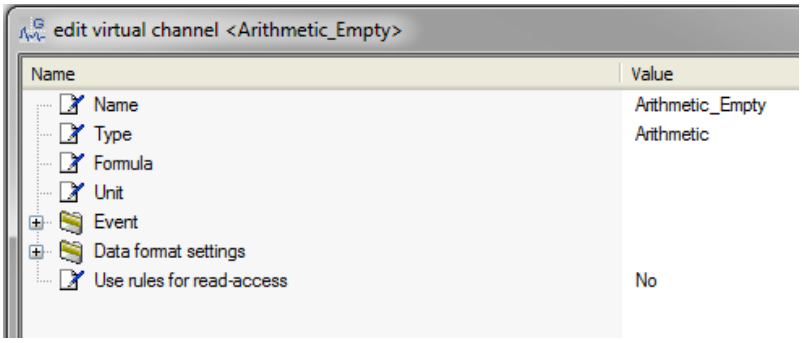
Procedure:

1. Right-click on the Virtual Variables section of a Q.station. Navigate to Add Variable and select Arithmetic Empty.

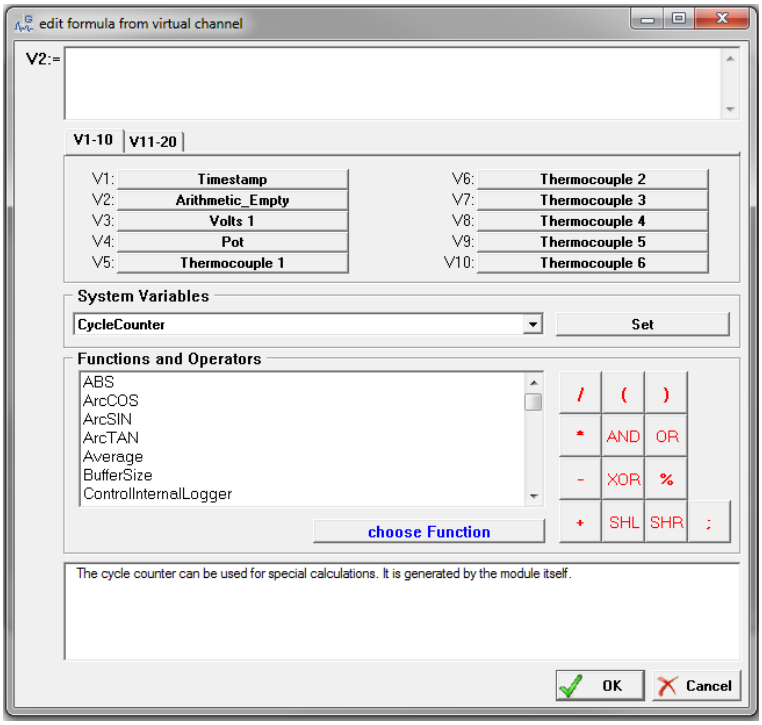


2. Double click on the new Arithmetic_Empty channel.

3. This will open the edit window for the channel. Specifications of the channel can be modified here including the name, type, unit, data format, etc. Click on the Formula section.



4. The formula edit window will appear. Use the variables available in the system and the standard operations to create the function. The list of functions and operators are shown here.



List of Functions and Operators:

1. ABS – Absolute Value:

   ABS(value)

   Instead of a value, a variable within the same system can be selected.

   Example: ABS(-243) = 243

2. ArcCOS – Arc Cosine:

   ArcCOS(value)

   Instead of a value, a variable within the same system can be selected.

3. ArcSIN – Arc Sine:

   ArcSIN(value)

   Instead of a value, a variable within the same system can be selected.

4. ArcTAN – Arc Tangents:

   ArcTAN(value)

   Instead of a value, a variable within the same system can be selected.

5. Average:

   Average(Variable;Type[;AdditionalParameter])

   Variable: Reference variable

   Type[;AdditionalParameter]: Type and correspondent additional parameter

Parameter	Description
0	Lowpass filter. AdditionalParameter defines filter frequency in [Hz].
1	Sliding average. AdditionalParameter defines number of values.
2	Event driven average. No AdditionalParameter.
3	North step average. No AdditionalParameter.
4	Arithmetic average. AdditionalParameter defines number of values.

6. BufferSize:

   This function gets the buffer size in percentage (100 % if the buffer is full).

   BufferSize(BufferIndex)

   BufferIndex: This index defines which buffer has to be monitored. (Note: if PAC-functionality is activated, only the DataLoggers in e.con can be indexed, not the Circlebuffer)

7. ControlInternalLogger

   This function is used to control the behavior of the internal DataLogger (Q.Station, Q.pac DL and e.pac DL).

   ControlInternalLogger(CommandSelector;LoggerIndex0[;Loggerindex1…LoggerIndex n-1])

   CommandSelector: 0 : Enable/Disable

   Event triggered function!

   If the event condition is true, logging is enabled, otherwise disabled.
   If the function is not used, the logger is, corresponding to the test.commander configuration, activated or deactivated.

8. COS – Cosine:

   COS(value)

   Instead of a value, a variable within the same system can be selected.

9. DateTime2OLE – Convert Date/Time to OLE2:

   DateTime2OLE(Year;Month;Day;Hours;Minutes;Seconds;Millisec)

   Example: DateTime2OLE(2006;09;01;16;30;25;10)

10. DateTime2TimeDC – Convert Date/Time to DC Time:

    DateTime2TimeDC(Year;Month;Day;Hours;Minutes;Seconds;Millisec)

    Example: DateTime2TimeDC(2006;09;01;16;30;25;10)

11. Deadband:

    Deadband(Variable;RangeMin;RangeMax;Type;AdditionalParameters)

    This function is used to suppress a defined range of a channels measurement range.
    The parameter “Type” defines what happens if (Variable >= RangeMin) && (Variable < RangeMax).

Type:	Additional Parameter:	
0:	Time [s]	duration the last valid value will be kept, then actual variable value
1:	Time [s]; Value	duration the last valid value will be kept, then fixed “Value”
2:	Time [s]; Value	“Value” for duration of “Time”, then actual variable value

    Example:
    Valid values are 0-360. All Values above 360 should be ignored for a short period of time.

    Deadband(Variable;361;999999;0;1)

    The function will stay at the last valid value for 1 s. Not until Variable is >360 for MORE than 1 s, then again the actual value of Variable will be taken.

12. Equal:

    The Equal function compares 2 arguments and if Arg1 = Arg2 the result is 1, otherwise it is 0.

    Equal(Arg1;Arg2)

The argument can be a variable in the system or a certain value.

Example:
Equal(17;17) = 1
Equal(17;12) = 0

13. FTPSendStoredBufferFile - A stored buffer file will be sent to an FTP server. This function is a combination of opening, sending, and closing the FTP connection:

    FTPSendStoredBufferFile(ServerIndex;BufferIndex;DataDrive;FileIdent;FileRepeatIndex;AppDevID;AppDateTime;Overwrite;DelScrAfter;IsBlocking)

    This function is event triggered.

    The files always have the following structure: ^xy_z.dat
    x = FileIdent character, y = BufferIndex, z = FileRepeatIndex

    ServerIndex: This is the index of the server being defined in the controller settings and indicates which server has to be connected to or where data have to be sent to. Up to 10 connections are possible, the index counts from 0..9, according to Connection#1 .. Connection #10

    BufferIndex: This index defines, which buffer file has to be sended (first buffer index = 0).

    DatadriveIndex: 0: internal (data only being stored in Flash)
    1: USB stick

    FileIdent: These are characters which can be used to identify different files. 0 = a, 1 = b,… e.g. 10 = k

    FileRepeatIndex: File Index. 0000 .. 9999 are valid repeat indices. This is being used in case very special files have to be sent. Using -1 the oldest data file will be sent, using -2 the latest file will be sent.

    AppDevID: 0: No device ID will be added
    1: The device ID (= location) is being added to the file name at the FTP server.

    AppDateTime: 0: Time and date will not be added
    1: Time and date of the beginning of the transfer will be added to the file name at the FTP server. e.g. ^xy_z_20060911.dat

    Overwrite: 0: If a file already exists at the FTP server, an error message will follow
    1: An existing file at the FTP server will be overwritten

    DeleteScrAfter: 0: The source file will not be deleted
    1: The source file will be deleted

    IsBlocking: 0: The thread will be handled without interruption of the main program (recommended!)

1: The main program will continue after the function had been finished

The result of this function is a certain state. The following states are possible:

0	Ready, OK
1	ConnectError
2	AddressIndexError
3	CannotCreateSubDirectoryError
4	CannotChangeToSubDirectoryError
5	TransferError
100	Connected
101	DisConnectedError
102	SendDataError
103	ScrFileNotFoundError
104	ScrFileDeleteError
105	DestFileDeleteError
501	ConnectInitInProgress
502	ConnectInProgress
503	SendBufferInitInProgress
504	SendBufferInProgress
505	DisConnectInitInProgress
506	DisConnectInProgress
1001	NotRunning

14. GetFreeMemoryOfDataDrivenInPercent

    GetFreeMemoryOfDataDriveInPercent(datadrive)

    This functionality provides the free memory of the data drive in percent.

15. GetPositioningData – Read values from NMEA devices or GarminGPS-USB:

    GetPositioningData(Selector;InfoSelector)

    Selector: defines the port where the device is connected to
    (e.g. GPS-Module connected via USB->RS232 converter and protocol NMEA-0183)

    100 = USB0 (left port)
    101 = Port0 on USB Hub on USB0 (left port)
    102 = Port1 on USB Hub on USB0 (left port)
    103 = Port2 on USB Hub on USB0 (left port
    104 = Port3 on USB Hub on USB0 (left port)
    200 = USB1 (right port)
    201 = Port0 on USB Hub on USB1 (right port)
    202 = Port1 on USB Hub on USB1 (right port)
    203 = Port2 on USB Hub on USB1 (right port)

204 = Port3 on USB Hub on USB1 (right port)

InfoSelector	Name	Description	Protocol
0	Time	TimeOLE2 (days since 01.01.1900)	GarminGPS, GGL, GGA, RMC
1	Latitude		GarminGPS, GGL, GGA, RMC
2	Longitude		GarminGPS, GGL, GGA, RMC
3	Speed	meter/second	GarminGPS, RMC
4	Heading	0=N, 90=E, 180=S, 270=W	GarminGPS, RMC
5	Sat Count	Number of Satellites Seen	GarminGPS, GSV, GGA
6	Altitude	Altitude above NN	GarminGPS, GGA
7	Quality	0=invalid, 1=GPS, 2=DGPS, 6=Estimated	GGA
8	Precision Dilution	Horizontal Dilution of Precision	GGA
9	Rotate of Turn		ROT
10	LongitudialWaterSpeed		VBW
11	TraversWaterSpeed		VBW
12	LongitudialGroundSpeed		VBW
13	TraversGroundSpeed		VBW
14	TrackDegreesTrue		VTG
15	TrackDegreesMagnetic		VTG
16	DepthBelowTransFeet	Depth below transducer (feet)	DBT
17	DepthBelowTransMeters	Depth below transducer (meter)	DBT
18	DepthBelowTransFathoms	Depth below transducer (fathoms)	DBT
19	DepthOfWater	Meter	DPT
20	OffsetFromTransducer		DPT
21	WaterAngle	0 to 360 degrees	MWV
22	WaterSpeed		MWV
23	WaterTemperature	Celsius	MTW
24	Heading	Own ship data	OSD
25	VesselCourse	Own ship data	OSD
26	VesselSpeed	Own ship data	OSD
27	VesselSet	Own ship data	OSD
28	VesselDrift	Own ship data	OSD
29	CursorRange	RADAR system data	RSD
30	CursorBearing	RADAR system data	RSD
31	RangeScale	RADAR system data	RSD
32	Heading	Heading Degrees, true	HDT
33	Speed Knots		VTG
34	Speed km/h		VTG
100	Error States		0=invalid char format 1=invalid baud rate 2=invalid port config 3=invalid sent. form.

Conversion of Lat/Long:
XXYY.ZZZZ ⇒ XX° + (YY.ZZZZ / 60)°

or

XXYY.ZZZZ ⇒ XX° YY' (0.ZZZZ * 60)''

16. GetSystemHealth(Selector)

    Selector:
    0: Actual System Health in %
    1: Actual Real-time Health in %
    2: Average System Health in %
    3: Average Real-time Health in %

17. High – The High function searches for the highest one of several arguments:

    High(Arg1;Arg2[;Arg3][;Arg4])

    The argument can be a variable or a certain value.

    Example: High(17;12;43;8) = 43

18. Higher - The Higher function compares 2 arguments and if Arg1 > Arg2 the result is 1, otherwise it is 0.

    Higher(Arg1;Arg2)

    The argument can be a variable or a certain value.

    Example:
    Higher(35;42) = result: 0
    Higher(35;23) = result: 1

19. HigherEqual - The HigherEqual function compares 2 arguments and if Arg1 is higher or equal than Arg2 the result is 1, otherwise it is 0.

    HigherEqual(Arg1;Arg2)

    The argument can be a variable or a certain value.

    Example:
    HigherEqual(35;35) à result: 1
    HigherEqual(17;35) à result: 0

20. LN – Logarithmic Base e:

    LN(value)

    Instead of a value a variable of the measurement system can be selected.

21. LOG – Logarithmic Base 10:

    LOG(value)

    Instead of a value a variable of the measurement system can be selected.

22. Loggerstate – This function can be used to get an actual state of a specific data logger. Depending on the result another action might follow, such as an email will be sent in case of an error. The result is one of the values according to the selector.

    Loggerstate(LoggerIndex; Selector)

    The following table indicates the type of the Selector and its description:

Selector	State	Bit	Description	Value
0	Error State	0	Config Error	1
		1	Buffer Overrun	2
		2	Data limit reached	4
		3	Renaming file failed	8
		4	Creating file failed	16
		5	No storage destination available	32
		6	Sending mail failed	64
		7	Sending file via FTP failed	128
1	Enabled		If the logger is enabled	0/1
2	Logging		If the logger is writing to a file	0/1
3	Start Trigger		If start triggering in progress	0/1
4	Stop Trigger		If stop triggering in progress	0/1
5	Files Stored		Number of logged files since	Int
6	File Progress		Progress of the actual file	0-100%
7	Trigger Progress		Progress of the actual trigger	0-100%
8	MailsSent		Number of eMails sent since start	Int
9	FTPSent		Number of files sent via FTP	Int
10	Destination		Index of the actual data storage	Int
11	Dest. Size		Size of the actual data storage	Byte
12	Dest. Remaining		Available size on the actual data storage	Byte
13	Dest. Load		Load of the actual data storage	0-100%
14	DataSource Overruns		Number of data source overruns	Int
15	DataSource Act. Size		Actual size of source data buffer	Byte
16	DataSource Capacity		Capacity size of source data buffer	Byte
17	DataSource Max Size		Maximum size of source data buffer since start	Byte
18	PostProcessorOverruns		Number of PostProcessor source overruns	Int

19	PostProcessoract Size		Actual size of PostProcessor source data buffer	Byte
20	PostProcessorCapacity		Capacity of PostProcessor source data buffer	Byte
21	PostProcessorMax Size		Max. size of PostProcessor source data buffer since start	Byte

23. Low - The Low function searches for the lowest one of several arguments:

    low(Arg1;Arg2[;Arg3][;Arg4])

    The argument can be a variable or a certain value.

    Example: low(35;21;46) = result: 21

24. Lower - The Lower function compares 2 arguments and if Arg1 < Arg2 the result is 1, otherwise it is 0.

    Lower(Arg1;Arg2)

    The argument can be a variable or a certain value.

    Example:
    Lower(17;12) = 1
    Lower(17;23) = 0

25. LowerEqual - The LowerEqual function compares 2 arguments and if Arg1 is lower or equal Arg2 the result is 1, otherwise it is 0.

    LowerEqual(Arg1;Arg2)

    The argument can be a variable or a certain value.

    Example:
    LowerEqual(17;17) = 1
    LowerEqual(17;12) = 0

26. MailSend:

    MailSend(AddressIndex;SubjectIndex;BodyIndex;WithDataFile;BufferIndex;DataDriveIndex;FileIdent;FileRepeatIndex;DeleteAfterSend;IsBlocking)

    This function is event triggered (the function is triggered as long as the event condition is active. Take care, that the event is only active for one cycle, so that only one email per event is sent)

    AddressIndex: Up to 10 e-mail addresses can be defined in the controller via “Host Settings, e-Mail”. The destination address index is 0 .. 9 according to “Email address #1” .. “Email address #10”, in case -1 is defined, the message will be sent to all e-mail addresses being defined.

SubjectIndex: Up to 10 subjects can be defined via “Host Settings, e-Mail”. The index is 0 .. 9 according to “Email subject #1” .. “Email subject #10”

BodyIndex: This is the text being defined via “Host Settings, e-Mail”. The index is 0 .. 9 according to “Email body text #1” .. “Email body text #10”

WithDataFile: 0: no
1: yes
(Only a file being stored in the Flash can be sent. Files in the RAM cannot be sent!)
The files always have the following structure: ^xy_z.dat or !xy_z.dat (file on USB-Stick)
x = FileIdent character, y = BufferIndex, z = FileRepeatIndex

BufferIndex: This index defines which buffer file has to be sent.

DatadriveIndex: 0: internal (data only being stored in Flash)
1: USB stick

FileIdent: These are characters which can be used to define different files. 0 = a, 1 = b,… e.g. 10 = k

FileRepeatIndex: Each file gets its own index. 0000 .. 9999 are valid repeat indices. This is being used in case very special files have to be sent. Using -1 the oldest data file will be sent, using -2 the latest file will be sent.

DeleteAfterSend: 0: The source file will not be deleted
1: The source file will be deleted

IsBlocking: 0: The thread will be handled without interruption of the main program (recommended!)
1: The main program will continue after the function had been finished

The result of this function is a certain state.
The following states are possible:

0	Ready, OK
1	DestinationAddressIndexError
2	SubjectIndexError
3	BodyIndexError
4	CreateError
5	SetDomainError
6	AuthenticateSendServerError
7	AuthenticatReceiveServerError
8	PrepareNewMediaError

9	SetDestinationAddressError
10	SetAttachmentError
11	SetSignatureError
12	SrcFileNotFoundError
13	ScrFileDeleteError
>=200	Internal Error
<=250	Internal Error
500	Busy
501	SendMailInitInProgress
502	SendMailInProgress
503	ReceiveMailInitInProgress
504	ReceiveMailInProgress
1001	NotRunning

27. MAX – Maximum Value:

    MAX(Value)

    The maximum value of a variable will be “stored” and has to be reset if required.
    To rest this variable, the settings have to be defined in “Event”

    These are the following possibilities to reset the value:

    - by host
    - on the logging interval
    - if a certain variable is > 0.5
    - on a digital input

28. MIN – Minimum Value:

    MIN(Value)

    The minimum value of a variable will be “stored” and has to be reset if required.
    To rest this variable, the settings have to be defined in “Event”

    These are the following possibilities to reset the value:

    - by host
    - on the logging interval
    - if a certain variable is > 0.5
    - on a digital input

29. NOT:

    NOT(Variable)

    With this function a value can be inverted.

30. OLE2DateTime - Convert OLE2 to Date/Time Part:

    OLE2DateTime(OLE2Time;PartSelector)

    This function converts OLE to Date/Time, the result is the value depending on the PartSelector:

    PartSelector: 0: year
    1: month
    2: day
    3: hour
    4: minute
    5: second
    6: millisecond

31. PIDController:

    PIDController(ReferenceValue;ActualValue;ProportionalPart;IntegralTime;DerivativeTime;Mode;TimeBase;OutputVarNumber[;Type])

    No events supported! (Turning control on/off is only possible with the “mode” parameter. If event “Host” is activated, the output value can be set in open loop mode.)

    ReferenceValue: - also named as setpoint or W
    - any variable like measured variable, setpoint or calculated variable

    AcutalValue: - also named as instantaneous value or X
    - any variable, usually a measured variable like Temperature, calculated variable also possible

    ProportionalPart: - also named as P-contribution
    - any variable or setpoint

    IntegralTime: - also named as integral part or I-contribution
    - any variable or setpoint
    Unit: if Type-ISM112 [min] else [s]

    DerivativeTime: - also named as differential part or D-contribution
    - any variable or setpoint
    Unit: if Type-ISM112 [min] else [s]

    Mode: Selectable is automatic (control) or hand mode, with or without limitation. Decimal coded bit-set:

Parameter	Bitset	Output	Loop
0	00	Output limitation on	Control loop open
1	01	Output limitation on	Control loop closed
2	10	Output limitation off	Control loop open
3	11	Output limitation off	Control loop closed

Output limitation: with Type-ISM112 to +/-100.0 else to +/-1.0
Standard value: 3

TimeBase [s]: Min = 1.0/sample frequency, selectable according the required control loop speed (slower control speed save CPU power)

OutputVarNumber:

Select the analog output variable that should be controlled by the PID (Y). The counting is for variables only with an OUTPUT-part.
For example:

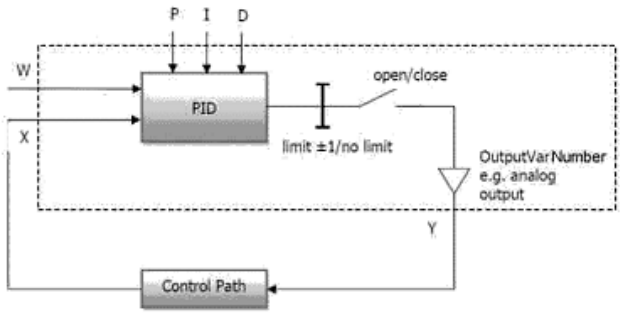
Channel	**DataDirection**	**OutputVarNumber**
V1 – Timestamp	INPUT	-
V2 – Setpoint1	INPUT	-
V3 – Setpoint2	INPUT/OUTPUT	1
V4 – Setpoint3	INPUT	-
V5 – Setpoint4	OUTPUT	2
V6 – WriteOutputVar	INPUT	-
V7 – OutputVar	INPUT/OUTPUT	3

Type: Optional parameter to define type of controller:

Parameter	Name	Description
0	e_q_gate_pac	D-Part based on derivative of error value (W - X)
1	q_station	D-Part based on derivative of process value (X)
2	ISM112	Like Type-q_station, but I- and D-Times are in [min] and the controller has an internal gain of 100

If this parameter is not set, with e./q.bloxx/pac-controllers Type-e_q_gate_pac and with q.station-controllers Type-q_station is used.
Remark: If same behaviour as with ISM112 is needed, also Mode-Bit1 = 0 (=Output limitation on) need to be set.

Functional Diagram:



32. Power:

    Power(x;y)

    The result of this function is x^y.
    Example: 2^3 = 8

33. RandomValue:

    RandomValue(Value)

    The paremeter "Value" needs to be an integer value > 1. As result the function generates random values between 1 and "Value".

34. Round to Value:

    RoundToValue(Value;decimal places)

    This function will be used to round a certain value.

    Example:
    RoundToValue(5,0537;1) = 5,1
    RoundToValue(5,0537;3) = 5,054

35. Select:

    Select Value:

    Select(Selector Variable;Variable1[;Variable2]…[;Variable8])

    The selector variable indicates which variable has to be choosen. This selector varaible can be defined via a setpoint channel but it also can be a result of some other arithmetics. A maximum of 8 variables are possible, the value of the SelectorVariable can be 0..7. For all other values the last variable will be used as a default value..

    Example: if you have 3 variables and the SelectorVariable is 7 = Variable 3 will be the default value.

36. SIN – Sine:

    SIN(value)

    Instead of a value a variable of the measurement system can be selected.

37. Sqrt – Square Root:

    Sqrt(value)

The argument can be a variable or a certain value.
Example: Sqrt(25) = 5

38. Square:

    Square(value)

    The argument can be a variable or a certain value.
    e.g. Square(4) = 16

39. StandDeviation – Standard Deviation:

    StandDeviation(value)

    The argument can be a variable or a certain value.

40. TAN – Tangents:

    TAN(value)

    Instead of a value a variable of the measurement system can be selected.

41. TrueRMS:

    TrueRMS(Variable;Type[;AdditionalParameter])

    Variable: Reference variable

    Type[;AdditionalParameter]: Type and correspondent additional parameter

Parameter	Description
0	Lowpass filter. AdditionalParameter defines time constant Tau in [s].
1	Sliding average. AdditionalParameter defines number of values.
2	Arithmetic average. AdditionalParameter defines number of values.

    Note: If only 2 parameters are set, type Lowpass filter is activated and 2nd parameter is taken as time constant Tau.

42. Trunc – Truncate Variable:

    Trunc(Value)

    The result of this function are just the digits left of the comma.
    Example: Trunc(17,689) à 17

43. ValueChanged:

    Value has changed:

    Two possible configurations:

    1) ValueChanged(Variable;RoundToValue)
       or
    2) ValueChanged(Variable;Type;HandleValue)

    Variable: defines the source value to be controlled

    ad 1)
    RoundToValue: with this parameter the accuracy can be defined, e.g. 0,1 è if the value of Variable changes at 0,1, this will indicate an event. The Value of the function will result in 1 for 1 cycle.
    The rounding will be done in the following way: 0-4 will not change anything; 5-9 will increase the value to the next higher one.

    Examples:
    ValueChanged(V3;0,02)
    If the value V3 changes with 0,02 this will indicate an event (1)

    ValueChanged(V1;0,5)
    If the value V1 changes from 0,24 to 0,25 this will indicate an event. (Values will be rounded to 0 or 0,5)

    ad 2)
    Type: 0: Same functionality as in configuration 1)
    1: Step since last cycle is greater than HandleValue. E.g. HandleValue = +0,5 = rising edges can be detected. E.g. value changes from 0 to 1. The value of the function will result in 1 for 1 cycle
    2: Step since last cycle is lower or equal than HandleValue. E.g. HandleValue = -0,5 = falling edges can be detected. E.g. value changes from 1 to 0. The value of the function will result in 1 for 1 cycle.

    (WARNING, when using this function in the Q.Station post-processed buffer (since firmware V1.08):

    use as type number 3, 4 and 5 instead of 0, 1 and 2!!)

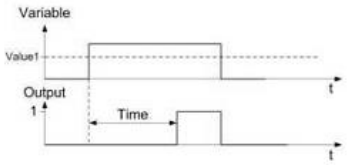
44. ValueEvaluation:

    ValueEvaluation(Variable;Type;Time;Value1[;Value2])

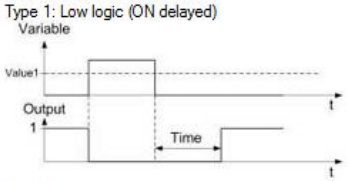
    ON or OFF delay, if a controlled Variable exceeds a threshold or gets in a defined range (see parameter “Type”). Also hysteresis definitions are possible with this function.

Variable:	defines the source value to be controlled (e.g. V1)
Type:	see below
Time:	delays state ON (0…5) or OFF (10…15) (only if condition is TRUE for t > Time)
Value 1, 2:	Thresholds

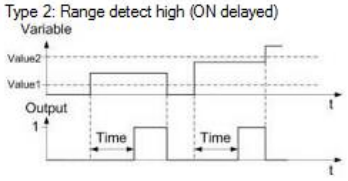
Type 0: High logic (ON delayed)



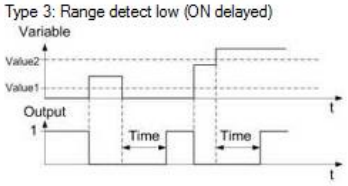
Type 1: Low logic (ON delayed)



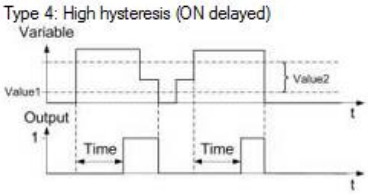
Type 2: Range detect high (ON delayed)



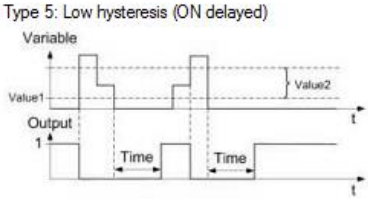
Type 3: Range detect low (ON delayed)



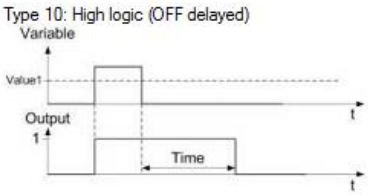
Type 4: High hysteresis (ON delayed)



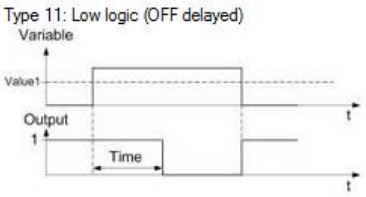
Type 5: Low hysteresis (ON delayed)



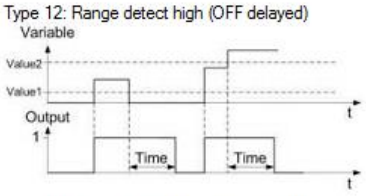
Type 10: High logic (OFF delayed)



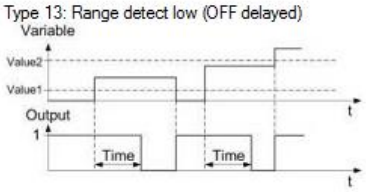
Type 11: Low logic (OFF delayed)



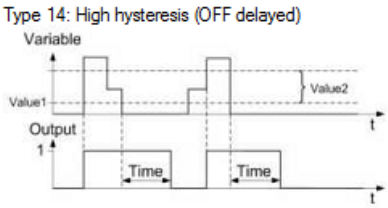
Type 12: Range detect high (OFF delayed)



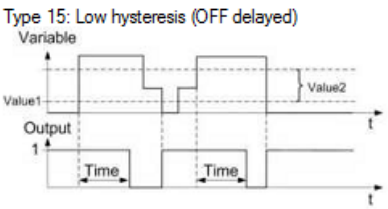
Type 13: Range detect low (OFF delayed)



Type 14: High hysteresis (OFF delayed)



Type 15: Low hysteresis (OFF delayed)



Gantner Instruments Incorporated
9835 Carroll Centre Road, Suite 100
San Diego, CA 92126 USA

■ **www.gantnerinstruments.com**
Toll Free: (877) 725-6997 (877 QBLOXXS)
Direct: (858) 537-2060

■ **info@gantnerinstruments.com**
Toll Free Fax: (800) 303-9381
Direct Fax: (858) 537-2064

45. WriteOutputVariable:

WriteOutputVariable(OutputVariableNumber;SourceValue)

OutputVariableNumber: defines to which output channel a certain value has to be written to. (Note: only variables with output-part as datadirection (OUTPUT or INPUT/OUTPUT) count.)

SourceValue: defines the source value (for example: V3). This source value can be a constant, a setpoint or another variable.

Example:
The channel setup looks like:

Channel	**DataDirection**	**OutputVarIndex**
V1 – Timestamp	INPUT	-
V2 – Setpoint 1	INPUT	-
V3 – Setpoint 2	INPUT/OUTPUT	1
V4 – Setpoint 3	INPUT	-
V5 – Setpoint 4	OUTPUT	2
V6 – WriteOutputVar	INPUT	-
V7 – SourceValue	INPUT/OUTPUT	3

WriteOutputVariable(1;V7) - value of "V7 - SourceValue" is written to "V3 - Setpoint2".
WriteOutputVariable(2;V7) - value of "V7 - SourceValue" is written to "V5 - Setpoint4".