#### Sedona 1.2 Component Descriptions



## Sedona 1.2 Component Descriptions

Developed by Tridium Inc., Sedona Framework<sup>™</sup> is a software environment designed to make it easy to build smart, networked, embedded devices which are well suited for implementing control applications. The system integrator's role is to create an application by assembling components onto a wire sheet using graphical programming tools such as Niagara Workbench or a third-party Sedona Tool. Applications are then executed by a Sedona Virtual Machine (SVM) resident in Contemporary Controls' BASremote or BAScontrol family of controllers.

Components are deployed in kits which are available from Tridium, Contemporary Controls and other members of the Sedona community. Kits without a company name are from Tridium. Kits with a company name and no product name are from a Sedona community member and these components can be used with other Sedona devices. Kits with both a company name and product name are hardware dependent thereby limiting portability. What follows are both standard and custom components compliant with Sedona release 1.2.28. These components are organized by kit name.

When studying these components keep the following in mind. Boolean variables are assumed if there is a false/ true state indication. Integers (32-bit signed integers) are shown as whole numbers while floats (32-bit floating point) are shown with a decimal point. Many of the following components may have been expanded in order to show all component slots in order to display configuration detail. The default view of these components on a wire sheet may not show the same level of detail. The standard Tridium components are shown first and it is Contemporary Controls' policy not to modify Tridium released components.

		Index	of Kits		
Basic Schedule	2	Math	12	<b>BASremote Platform</b>	20
Date Time STD	3	Priority	15	BAScontrol20/22 IO	21
Function	4	Timing	16	BAScontrol20/22 Platform	n 23
HVAC	8	Types	17	BAScontrol20 Web	24
Logic	10	BASremote Se	ervice 19	<b>BAScontrol Function</b>	25

#### Basic Schedule Kit (basicSchedule)

*DailySchedule* represents a simple daily schedule with up to two active periods. Each active period is defined by a start time and duration. If the duration is zero, the period is disabled. If the periods overlap, then the first period (defined by *Start1* and *Dur1*) takes precedence. If the duration extends past midnight, then the active period will span two separate calendar days. There are two components in the kit — one for Boolean outputs and the other for floats. Both kits rely upon the time being set in the target hardware.

Duration periods — *Dur1* and *Dur2* — are configured in minutes from zero to 1439 minutes.

DailySc basicSchedule::DailyScheduleBool	$\odot$
Start1	0 min
Dur1	0 min
Start2	0 min
Dur2	0 min
Val1	false
Val2	false
Def Val	false
Out	false

#### Daily Schedule Boolean — two-period Boolean scheduler.

Configure *Def Val* to the intended output value if there are no active periods. Configure *Val1* and *Val2* for the desired output values during period 1 and period 2 respectively.

Out = Def Val if no active periods Out = Val1 if period 1 is active Out = Val2 if period 2 is active

DailyS1 basicSchedule::DailyScheduleFloat		$\bigcirc$
Start1	0	min
Dur1	0	min
Start2	0	min
Dur2	0	min
Val1	(	0.00
Val2	(	0.00
Def Val	(	0.00
Out	(	0.00

#### Daily Schedule Float — two-period float scheduler.

Configure *Def Val* to the intended output value if there are no active periods. Configure *Val1* and *Val2* for the desired output values during period 1 and period 2 respectively.

CONTEMPORARY ONTROL

Out = Def Val if no active periods

*Out* = *Val1 if period 1 is active* 

Out = Val2 if period 2 is active

#### Date Time STD Kit (datetimeStd)

DateTim datetimeStd::Da	ateTimeServiceStd
Nanos	1378541000000000 ns
Hour	22
Minute	55
Second	41
Year	2000
Month	1
Day	16
Day Of Week	0
Utc Offset	0 s
Os Utc Offset	false

The DateTim component is the only component in the Date Time STD Kit. This component relies upon a properly functioning realtime clock implemented in hardware. Once date and time are configured, this component can be dragged onto a worksheet allowing individual integer outputs to be wired to logic if so desired. However, it is not necessary to have the component on the wiresheet at all. If the DailySchedule components are to be used, they will function properly without the presence of the DateTim component. The start and stop times in the DailySchedule key on the daily time generated by the DatTime component regardless if this component is on the wiresheet.

#### Please Note

By double clicking the DateTim component, you will see the setup screen below. When using Contemporary Controls' controllers, make sure that the Use System Offset option is selected as shown. To avoid confusing time settings, do not set the time with this component. Set the time using the Set Time web page on the controller which provides more flexibility. You can set time zone, daylight saving time and in some instances Network Time Protocol support using just the web page. These settings will then set this Sedona component properly.

DateTime Manage syste	Service em clock for device	
	Current	Desired
Current Time	27-May-2015 17:34:41 Wed	27-May-2015 05:34 PM CDT 🐳 📰
Time Zone		America/Chicago
UTC Offset	-5 hr	-5 hr
UTC Offset Mode	Using System Offset	$\diamondsuit$ Use System Offset $\diamondsuit$ Use Configured Offset
		Second Time



(func)

#### Function Kit

Cmpr func::Cmpr	
Xgy	false
Xey	true
Xly	false
Х	0.00
Y	0.00

Comparison math — comparison (<=>) of two floats.

If X > Y then Xgy is true If X = Y then Xey is true

# Count<br/>func::CountImage: CountOut0InfalsePreset0DirupEnablefalseRfalse

#### Integer counter — up/down counter with integer output.

Counts on the false to true transition of *In*. If Dir = true the counter counts up to the maximum value of the integer. If Dir = false the counter counts down but not below zero. For counting to occur, *Enable* must be *true*. The counter can be preset. If R = true and *Enable* = *true*, then Out equals the preset value and will not count.

Freq func::Fr	eq 🔲
Pps	0.000 /s
Ppm	0.000 /min
In	false

Hystere func::Hysteresis	
In	0.00
Out	false
Rising Edge	50.00
Falling Edge	50.00

IRamp func::IRamp	
Out	77
Min	0
Max	100
Delta	1
Secs	1 s

#### Pulse frequency — calculates the input pulse frequency.

*Pps* = number of pulses per second of In *Ppm* = number of pulses per minute of In

#### Hysteresis — setting on/off trip points to an input variable.

There are two internal floats called *Rising Edge* and *Falling Edge* which are configurable. If *Rising Edge* is greater than *Falling Edge*, then the following is true.

If In > Rising Edge then Out = true and will remain in that state until In < Falling Edge If Rising Edge is less than Falling Edge then the action is inverted.

#### IRamp — generates a repeating triangular wave with an integer output.

There are four configurable float parameters — *Min, Max, Delta and Secs.* For every scan cycle, the output increments by *Delta* units until the output equals the *Max* value at which time it decrements until *Min* is reached. The result is a triangular wave with limits of *Max* and *Min* and an incremental rate of *Secs* units.

#### Limiter — Restricts output within upper and lower bounds.

Limiter func::Limiter	
Out	0.00
In	0.00
Low Lmt	0.00
High Lmt	0.00

High Lmt and Low Lmt are configurable floats. If In > High Lmt then Out = High Lmt If In < Low Lmt then Out = Low Lmt If In < High Lmt and > Low Lmt then Out = In

Lineari func::Linearize	8.
Out	nan
In	0.00
X0	0.00
YO	0.00
X1	0.00
Y1	0.00
X2	0.00
Y2	0.00
X3	0.00
Y3	0.00
X4	0.00
Y4	0.00
X5	0.00
Y5	0.00
X6	0.00
Y6	0.00
X7	0.00
Y7	0.00
X8	0.00
Y8	0.00
Х9	0.00
Y9	0.00

LP func::LP	۲
Enable	true
Sp	0.00
Cv	0.000
Out	0.00
Кр	1.000000
Ki 0.00	0000 /min
Kd 0	.000000 s
Max 10	00.000000
Min	0.000000
Bias	0.000000
Max Delta	a 0.000000
Direct	true
Ex Time	1000 ms

#### Linearize — piecewise linearization of a float.

For piecewise linearization of a nonlinear input, there are ten pairs of x,y parameters that must be configured into this component. The x,y pairs indicate points along the input curve. For an x value of the input, there should be a corresponding y value of the output. For input values between these points, the component will estimate the output based upon the linear equation:

$$Out = y = y_0 + (y_1 - y_0) \frac{x - x_0}{x_1 - x_0}$$

where y is the value for input value x between coordinates  $x_0$ ,  $y_0$  and  $x_1$ ,  $y_1$ 

#### LP — proportional, integral, derivative (PID) loop controller.

The LP component is much more complex component requiring an explanation of the numerous configurable parameters. *Sp* is the *setpoint* or the desired outcome. *Cv* is the *controlled variable* which we are trying to make equal to the setpoint. The difference between *Cv* and *Sp* is the *error signal* (*e*) that drives the *output variable Out* used to manipulate the *controlled variable*. There are three gain factors *Kp*, *Ki*, *Kd* — called *tuning parameters* — for each of the three modes of the controller: *proportional, integral and derivative*. Setting a gain factor to zero effectively disables that particular mode. Setting *Kp* to zero would completely disable the controller. Typical controller operation is either:

Proportional-only (P)

Proportional plus reset (integral) (PI) Proportional plus reset plus rate (PID)

In HVAC applications, P and PI are the most common. PID is seldom used.

*Enable* must be set true if loop action is to occur. If *Enable* is set to false, control action ceases and the output will remain at its last state. However, if *Ki* or *Kd* are non-zero, internal calculations will continue.

If *Direct* is equal to *true,* then the output will increase if the *Cv* becomes greater than *Sp*. If this was a temperature loop, this would be considered being in *cooling mode*. If *Direct* is equal to *false*, then the output will decrease if the *Cv* becomes greater than the *Sp*. If this was a temperature loop, this would be considered being in *heating mode*. Notice that by entering negative gain factors, the action of the controller is reversed.

*Max* and *Min* are limits on the output's swing and are considered the absolute boundaries to the controller's throttling range (proportional control range). Basically, the *LP* component includes *Limiter* functionality.

*Bias* sets the output's offset. Sometimes *bias* is called manual reset to correct an output error with a large proportional band. It is usually only used with proportional-only control. The amount of bias is not influenced by the proportional gain *Kp*. Bias is also used on split-range control systems that will be discussed shortly.

*Ex Time* is the amount of time in milliseconds that the control loop is solved. Typical times are from 100–1000 ms with a default of 1000. Most HVAC loops are slow acting and therefore solving loops faster brings no benefit.

In the following discussion on setting the gain factors, assume we need a temperature controller enabled for direct action and that the output can swing from -50% to +50%. When the output ranges from 0 to 50%, a proportional cooling valve is modulated. When the output ranges from 0 to -50%, a proportional heating valve is modulated. At 0% output no valve is open. This is called a split range control system. *Max* and *Min* are set to -50 and +50 respectively. When we force the controller output from maximum heat to maximum cooling (100% output change), we notice that we can effect a change in our process temperature of 20°. This becomes our throttling range. In the real world, conducting this test might be difficult.

Now we need to set the three tuning parameters. We first begin by setting *Ki* and *Kd* to zero, thereby creating a proportional-only controller. The controller equation therefore becomes:

#### Out = Kp(e) + Bias where e = Cv - Sp and Bias equals zero

Our first guess at Kp is 5 because we know that a 100% change in output yielded a 20° change in process temperature. This assumes that we can cool with the same efficiency as we can heat which may not be the case. By having a Kp of 5, the output will remain linear over this wide range. Notice that if there is no error signal (Cv-Sp is equal to zero), the output will then equal the bias, but in this case the bias is zero. The value 5 is entered into Kp and a disturbance is introduced into the process such as a step change in the setpoint. If the process continues to oscillate between heating and cooling and never settles down, then we must reduce our proportional gain Kp which increases our proportional band (1/Kp times 100% is the proportional band). Assume we achieve a stable system with Kp at 5 (proportional band at 20%) but based on the load on the system we notice that the output reached 70%. Our setpoint is at 70°, but our controlled temperature is 74°. Temperature is stable, but we have a 4° offset. This is the inherent difficulty with proportional-only control, there is an offset depending upon the value of the output. We have two choices. We can increase the proportional gain to 10 because we do not need a 20° range in input, but we risk oscillation. The second approach is to "reset the output manually" by increasing the bias. Approach one will never solve the problem but will minimize it, and there is a better method to approach two and that is called *automatic reset* — or adding reset action by adding a *Ki* term. The new controller equation becomes:

Out = Kp(e + Ki e dt) (Bias is disabled when Ki is non-zero.)

If there remains an error signal ( $e \neq 0$ ), then the integral of the error over time will continue to drive the output until the error is driven to zero. The amount of action is determined by the *Ki* term. Notice that the integral term in the equation is also multiplied by the proportional gain before being applied to the output. The *Ki* coefficient is defined in units of repeats per minute. Too large a value can cause overshoot while too small a value will make the control system sluggish. The final setting *Kp* and *Ki* is done in the field based upon system response.

The third parameter is the rate parameter *Kd* which acts upon the rate of change of the error signal. Adding this term changes the controller equation as follows:

#### Out = Kp(e + Ki e dt + Kd de/dt)

For processes with extremely long reaction times, derivative control could be helpful in reducing overshoot. *Kd* is entered in seconds. As mentioned before, it is seldom used because tuning a control loop with three parameters can be challenging.

There is one more parameter called *Max Delta*. This value limits the output slew rate by restricting the output change each time the control loop is recalculated by the amount entered. This parameter will dramatically reduce the response time of the control loop.

Ramp func::Ramp	
Out	87.48
Min	0.00
Max	100.00
Period	10 s
Ramp Type	triangle

SRLatch func::SRLatch	
Out	false
S	false
R	false

TickToc func::TickTock	
Out	false
Ticks Per Sec	1 /s

UpDn func::UpDn	
Out	0.00
Ovr	false
In	false
Rst	false
C Dwn	false
Limit	0.00
Hold At Limit	false

## Ramp — generates a repeating triangular or sawtooth wave with a float output.

There are four configurable float parameters — *Min, Max, Period* and *Ramp Type*. For every scan cycle, the output increments by one unit until the output equals the *Max* value at which time it decrements until *Min* is reached. The result is a triangular wave with limits of *Max* and *Min* if *Ramp Type* is set for triangle. If *Ramp Type* is set for sawtooth, then the output will immediately drop to *Min* when *Max* is reached. The *Period* of the ramp is adjustable.

#### Set/Reset Latch — single-bit edge-triggered data storage.

The following logic applies on the false-to-true transition of S or R:

If S goes true and R does not change, then Out = true and remains true.

If R goes true and S does not change, then Out = false and remains false.

If both S and R go true on the same scan, then Out = false and remains false.

#### Ticking clock — an astable oscillator used as a time base.

There is one configurable float parameter — *Ticks Per Sec* — which can range from a low of 1 to a high of 10 pulses per sec.

Out = a periodic wave between 1 and 10 Hz

#### Float counter — up/down counter with float output.

The counter range is between zero and a value that can be set with configurable parameter *Limit*. To cease counting at the limit set the configurable parameter *Hold at Limit* to true. To count down instead of up, set *C Dwn* to true. To reset the counter to zero set *Rst* to true. *Ovr* is the overflow indicator. *In* is the Boolean count input.

Out = the current count If Out  $\geq$  Limit then Ovr is true

#### HVAC Kit (hvac)

LSeq	
hvac::LSeq	
In	0.00
In Min	0.00
In Max	100.00
Num Outs	16
Delta	5.88
D On	0
Out1	false
Out2	false
Out3	false
Out4	false
Out5	false
Out6	false
Out7	false
Out8	false
Out9	false
Out10	false
Out11	false
Out12	false
Out13	false
Out14	false
Out15	false
Out16	false
Ovfl	false

#### Linear Sequencer — bar graph representation of input value.

There are two internally configurable floats called *In Min* and *In Max* that set the range of input values. An internal configurable integer — called *Num Outs* — specifies the intended number of active outputs. By dividing the input range by one more than the number of active outputs, the *Delta* between outputs is determined. Outputs will turn on sequentially from *Out1* to *Out16* within the input range as a function of increasing input value.

For example: *In Min* is set to 0, *In Max* to 100, and *Num Outs* is set to 9. This would give a *Delta* of 10. The following is true for increasing values of the input:

If In = 9 then Out1-16 are false and D On is zero.

If In = 70 then Out1–7 are true and Out8–16 are false. D On is 7.

If In = 101 then Out1–9 are true and Out10–16 are false. D On is 9 and Ovfl is true.

Note that for decreasing values of the input, the outputs will change by a value of Delta/2 below the input values stated above.

ReheatS	
Out1	false
Out2	false
Out3	false
Out4	false
In	0.00
Enable	false
D On	0
Hysteresis	0.00
Threshold1	0.00
Threshold2	0.00
Threshold3	0.00
Threshold4	0.00

#### Reheat Sequence — linear sequence up to four outputs.

There are four configurable threshold points — *Threshold1* through *Threshold4* — that determine when a corresponding output will become true as follows:

 $Out1 = true when In \ge Threshold1$ 

 $Out2 = true when In \ge Threshold2$ 

 $Out3 = true when In \ge Threshold3$ 

 $Out4 = true when In \ge Threshold4$ 

These outputs will remain true until the input value falls below the corresponding threshold value by an amount greater than the configurable parameter *Hysteresis*. Output signal *D On* indicates how many outputs are true. Configurable parameter *Enable* must be true otherwise all outputs will be false.

Reset hvac::Rese	et 🛛
Out	0.00
In	0.00
In Min	0.00
In Max	4095.00
Out Min	0.00
Out Max	100.00

#### Reset — output scales an input range between two limits.

There are four configurable float parameters — *In Max, In Min, Out Max* and *Out Min* — which determine the input and output ranges respectively of the input and output. The output of this component will scale linearly with the value of the input if the input is within the input range. The input range (IR) is determined by *In Max-In Min* while the output range (OR) is determined by *Out Max-Out Min.* If the input is within the input range then the following is true:

*Out* = (*In* + *In Min*)(*OR*/*IR*) + *Out Min* If the input exceeds, *In Max* then *Out* = *Out Max.* If the input is less than, *In Min* then *Out* = *Out Min* 

Tstat hvac::Tstat	E
Diff	0.00
Is Heating	false
Sp	0.00
Cv	0.00
Out	false
Raise	false
Lower	false
l	

#### Thermostat — on/off temperature controller.

The configurable float parameter — *Diff* — provides hysteresis and deadband. Another configurable parameter — *Is Heating* — indicates a heating application. *Sp* is the *setpoint* input and *Cv* is the *controlled variable* input. *Raise* and *lower* are outputs.

If Cv > (Sp + Diff/2) then Lower is true and will remain true until Cv < SpIf Cv < (Sp - Diff/2) then Raise is true and will remain true until Cv > SpIf Is Heating is true then Out = LowerIf is Heating is false then Out = Raise



#### Logic Kit (logic)

ADemux2 logic::ADemux2	
Out1	0.00
Out2	0.00
In	0.00
S1	false

#### Analog Demux — Single-input, two-output analog de-multiplexer.

If S1 is false then Out1 = In while Out2 = the last value of In just before S1 changed.

If S1 is true then Out2 = In while Out1 = the last value of In just before S1 changed.

And2 logic::And2	٤
Out	false
Ini	false
In2	false

Two-input Boolean product — two-input AND gate.

And4 logic::And4	٤
Out	false
In1	false
In2	false
In3	false
In4	false

Four-input Boolean product — four-input AND gate.

 $Out = In1 \cdot In2 \cdot In3 \cdot In4$ 

 $Out = In1 \cdot In2$ 

ASW logic::ASW	E
Out	0.00
In1	0.00
In2	0.00
S1	false

Analog switch — selection between two float variables.

If S1 is false then Out = In1 If S1 is true then Out = In2

ASW4 logic::ASW4	
Out	0.00
In1	0.00
In2	0.00
In3	0.00
In4	0.00
Starts At	0
Sel	0

Analog switch — selection between four floats. Configurable integer parameter *Starts At* sets the base selection. *If integer Sel <= Starts At then Out = In1 If integer Sel = Starts At + 1 then Out = In2 If integer Sel* = *Starts At* + 2 *then Out* = *In3 If integer Sel = Starts At + 3 then Out = In4* For all values of Sel that are 4 greater than Starts At then Out = In4

B2P logic::B2P	
Out	false
In	false

Binary to pulse — simple mono-stable oscillator (single-shot). Out = true for one scan on the raising edge of In

BSW logic::BSW	
Out	false
In1	false
In2	false
S1	false

DemuxI2 logic::DemuxI2B4	
In	0
Out1	true
Out2	false
Out3	false
Out4	false
Starts At	0

Boolean Switch — selection between two Boolean variables.

If S1 is false then Out = In1If S1 is true then Out = In2

If S1 is false then Out = In1

#### Four-output Demux — integer to Boolean de-multiplexer. *If In = StartAt + 0 then Out1 is true, else false If In = StartAt + 1 then Out2 is true, else false* If In = StartAt + 2 then Out3 is true, else false

Integer switch — selection between two integer variables.

ISW logic::ISW	E
Out	0
In1	0
In2	0
S1	false

Not logic::Not	
Out	true
In	false

Or2 logic::Or2	U
Out	false
In1	false
In2	false

Or4 logic::Or4	
Out	false
In1	false
In2	false
In3	false
In4	false
1	

Xor logic::Xor	
Out	false
In1	false
In2	false

If In = StartAt + 3 then Out4 is true, else false

# If S1 is true then Out = In2

Not — inverts the state of a Boolean.  $O_{ijt} = In$ 

Two-input Boolean sum — two-input OR gate. *Out* = *In1* | *In2* 

Four-input Boolean sum — four-input OR gate. Out = In1 | In2 | In3 | In4

Two-input exclusive Boolean sum — two-input XOR gate.  $Out = On1 + In2 = In1 \cdot In2 | In1 \cdot In2$ 

CONTEMPORARY

Add2 math::Add2	+
Out	0.00
In1	0.00
In2	0.00

(math)

Add4 math::Add4	E
Out	0.00
In1	0.00
In2	0.00
In3	0.00
In4	0.00
l	J

Avg10 math::Avg10	R
Out	nan
In	0.00
Max Time	0 ms

AvgN	
math::Avgiv	
Out	0.00
In	0.00
Num Samples To	o Avg 5
Reset	false

Div2 math::Div2	
Out	0.00
In1	0.00
In2	0.00
Div0	true

**Two-input addition** — results in the addition of two floats. Out = In1 + In2

#### Four-input addition — results in the addition of four floats.

Out = In1 + In2 + In3 + In4

# Average of 10 — sums the last ten floats while dividing by ten thereby providing a running average.

Out = (sum of the last ten values)/ten

The float input *In* is sampled once every scan and stored. If the input does not change in value on the next scan, it is not sampled again — unless sufficient time passes that exceeds the internal integer *Max Time* with units of milliseconds. In this instance the input is sampled and treated as another value. Once ten readings occur, the average reading is outputted.

## Average of N — sums the last N floats while dividing by N thereby providing a running average.

Out = (sum of the last N values)/N

The float input *In* is sampled once every scan and stored regardless whether or not the value changes. Once *Num Samples to Avg* readings occur, the average reading is outputted.

#### Divide two — results in the division of two floats.

Out = In1/In2 Div0 = true if In2 is equal to zero

FloatOf math::FloatOffset	Ŧ
Out	0.00
In	0.00
Offset	0.00

#### Float offset — float shifted by a fixed amount.

*Out* = *In* + *Offset Offset* is a configurable float.



Max math::Max	
Out	0.00
In1	0.00
In2	0.00

Maximum selector — selects the greater of two inputs. Out = Max [In1, In2] where Out, In1 and In2 are floats

Min math::Min	
Out	0.00
In1	0.00
In2	0.00

Minimum selector — selects the lesser of two inputs. Out = Min [In1, In2] where Out, In1 and In2 are floats

#### Min/Max detector — records both the maximum and minimum values of a float.

MinMax math::MinMax	
Min Out	0.00
Max Out	0.00
In	0.00
R	false

Min Out = Max [In] if R is false Max Out = Min [In] if R is false If R is true then Min Out and Max Out = In Both Min Out and Max Out are floats — as is In. It may be necessary to reset the component after connecting links to the component.

Multiply two — results in the multiplication of two floats.

Mul2 math::Mul2	
Out	0.00
In1	0.00
In2	0.00
[	

Out = In1 \* In2

Mul4 math::Mul4	
Out	0.00
In1	0.00
In2	0.00
In3	0.00
In4	0.00
1114	0.00

Multiply four — results in the multiplication of four floats.

#### Out = In1 \* In2 \* In3 \* In4

Neg math::Neg	
Out	0.00
In	0.00

Negate — changes the sign of a float. Out = - ln

Round math::Round	۲
Out	0.0
In	0.000
Decimal Place	s 0

**Round — rounds a float to the nearest N places.** For N = -1, Out = In rounded to the nearest tens For N = 0, Out = In rounded to the nearest units For N = 1, Out = In rounded to the nearest tenth's For N = 2, Out = In rounded to the nearest hundredths For N = 3, Out = In rounded to the nearest thousandths For positive input values, the output will round up (more positive). For negative input values, the output will round down (more negative).

Sub2 math::Sub2	
Out	0.00
In1	0.00
In2	0.00

Subtract two — results in the subtraction of two floats. Out = ln1 - ln2

Sub4 math::Sub4	
Out	0.00
In1	0.00
In2	0.00
In3	0.00
In4	0.00

Subtract four — results in the subtraction of four floats. Out = In1 - In2 - In3 - In4

TimeAvg math::Tim	eAvg
Out	0.00
In	0.00
Time	10000 ms

**Time Average — the average of a float over a determined time.** *Out = Avg[In] over the integer time in milliseconds.* 



## Priority Kit

#### (pricomp)

Priorit	
pricomp::PrioritizedBool	9
In2	false
In3	false
In4	false
In5	false
In7	false
In9	false
In10	false
In11	false
In12	false
In13	false
In14	false
In15	false
In16	false
Fallback	false
Out	false

Priority array (Priorit) components exist for Boolean, float and integer variables. Up to 16 levels of priority from In1 to In16 can be assigned. In1 has the highest priority and In16 the lowest. With few exceptions, all can be pinned out. If a priority level is not assigned, it is marked as a Null and therefore ignored. If a Null is inputted to the priority array, the priority array will ignore it and choose the next input in line. The Boolean version of the array has two timer settings — one for minimum active time and one for minimum inactive time. If the highest priority device changes from false to true and then back to false, the priority component will maintain the event for the configured times.

There is a Fallback setting in each array that can be specified. If no valid priority input exists, the Fallback value is transferred to the output.

Priori1 pricomp::PrioritizedFloat	$\odot$
In2	0.00
In3	0.00
In4	0.00
In5	0.00
In6	0.00
In7	0.00
In9	0.00
In10	0.00
In11	0.00
In12	0.00
In13	0.00
In14	0.00
In15	0.00
In16	0.00
Fallback	0.00
Out	0.00

Priori2	$\bigcirc$
pricomp::Prioritizeain	
In2	0
In3	0
In4	0
In5	0
In6	0
In7	0
In9	0
In10	0
In11	0
In12	0
In13	0
In14	0
In15	0
In16	0
Fallback	0
Out	0

#### Timing Kit (timing)

DlyOff timing::DlyC	off 🔲
Out	false
In	false
Delay Time	0.00 s
Hold	0 ms

#### Off delay timer — time delay from a true to false transition of the input.

For input transitions from false to true, Out = true.

For input transitions from true to false that exceed the Delay Time, Out = false after the delay time.

On delay timer — time delay from a false to true transition of the input.

Hold is a read-only integer that counts down the time. Delay time is in seconds.

DlyOn timing::DlyO	n
Out	false
In	false
Delay Time	0.00 s
Hold	0 ms

For input transitions from true to false, Out = false.

For input transitions from false to true that exceed the Delay Time, Out = true after the delay time.

Hold is a read-only integer that counts down the time. Delay Time is in seconds.

OneShot timing::OneSh	ot 🔲
Out	false
In	false
Pulse Width	0.00 s
Can Retrig	false

#### Single Shot — provides an adjustable pulse width to an input transition.

Upon the input transitioning to true, the output will pulse true for the amount of time set in the configurable parameter *Pulse Width*. Time is in seconds. If the configurable parameter *Can Retrig* is set to true, the component will repeat its action on every positive transition of the input. For example in retrigger mode, a one-second *TickToc* connected to a *OneShot* with a 2 second pulse width setting will have the *OneShot* output in a continuous true state due to constant retriggering at a rate faster than the *OneShot* pulse width.

Timer timing::Timer	٧
Out	false
Run	stop
Time	0 s
Left	0 s

#### Timed pulse — predefined pulse output.

Out becomes true for a predetermined time when Run transitions from false to true. If Run returns to false, then Out becomes false.

*Time* determines the amount of time the output will be on in seconds.

CONTEMPORARY

-ONTRO

Types	Kit	(types)
-------	-----	---------

B2F	
Out	0.00
Count	0.00
In1	false
In2	false
In3 In4	false
104 In5	false
In6	false
In7	false
In8	false
In9	false
In10	false
Inii Ini2	falce
In12 In13	false
In14	false
In15	false
In16	false
l	

#### Binary to float encoder — 16-bit binary to float conversion.

*Out = encoded value of binary input with In16 being the MSB and In1 being the LSB* 

*Count* = *sum of the number of active inputs* 

ConstBo types::ConstBoo	۲	
Out	false	

**Boolean Constant — a predefined Boolean value.** *Out = a Boolean value that is internally configurable* 

ConstFl types::ConstFloat	
Out	0.00

Float Constant — a predefined float value. Out = a float value that is internally configurable

ConstIn types::ConstInt	0
Out	0

**Integer Constant — a predefined integer value.** *Out = an integer value that is internally configurable* 



F2B types::F2B	
In	0.00
Out1	false
Out2	false
Out3	false
Out4	false
Out5	false
Out6	false
Out7	false
Out8	false
Out9	false
Out10	false
Out11	false
Out12	false
Out13	false
Out14	false
Out15	false
Out16	false
Ovrf	false
l	

F2I types::F2I	
In	0.00
Out	0

12F types::I2F	E
In	0
Out	0.00

L2F types::L2F	»
In	1
Out	1.00

WriteBo types::WriteBool	
In	null
Out	null

WriteFl types::WriteFloat	$\bigcirc$
In	nan
Out	nan

WriteIn types::WriteInt	
In	min
Out	min

#### Float to binary decoder — float to 16-bit binary conversion.

Out1 to Out16 = the 16-bit decoded value of In — with Out16 representing the MSB and Out1 representing the LSB

Ovrf = true when In > 65535

Although the input requires a float, fractional amounts are ignored during the conversion.

#### Float to integer — float to integer conversion.

*Out = In except that the output will be a whole number* The fractional amount of the float input will be truncated at the output.

#### Integer to Float — integer to float conversion.

Out = In except that the output will become a float

Long to Float — 64-bit signed integer to float conversion. Out = In except that the output will become a float from a 64-bit signed integer

#### Write Boolean — setting a writable Boolean value.

#### Out = In

Unlike *ConstBo*, this component has an input. Could be helpful when transferring a variable between two wire sheets.

#### Write Float — setting a writable float value.

#### Out = In

Unlike *ConstFI*, this component has an input. Could be helpful when transferring a variable between two wire sheets.

#### Write Integer — setting an integer value.

#### Out = In

Unlike *Constln*, this component has an input. Could be helpful when transferring a variable between two wire sheets.

#### BASremote Service Kit (CControls\_BASR8M\_Services)

The BASremote service kit allows Sedona application to tie into real world inputs and outputs after object instance configuration. For the BASremote master, object instance assignments must match the I/O channel assignment. For configuring expansion module and virtual points, consult the BASremote User Manual for details. For the online status to revert to true, the point must be properly configured, must be actively scanned by the hardware and not be in a forced state.

InpBool CControls	BASR8M	Services::InpBool	$\odot$
Out			false
Online			false

Input I	Boolean –	– BASremote	binary	input.

Out = value of the real world binary input

InpFloa CControls	BASR8M	Services::InpFloat	$\odot$
Out			0.00
Online			false

OutBool CControls	BASR8M	Services::OutBool	$\bigcirc$
In			fals
Online			fals

OutFloa CControls	BASR8M	Services::OutFloat	$\bigcirc$
In			0.00
Online			false

OutFlo1 CControls BASR8M Services::OutFloatCond	$\bigcirc$
In	0.00
Out	0.00
Enable	false
Online	false

**Input Float — BASremote analog input or value.** *Out = value of the real world analog input* 

#### Output Boolean — BASremote binary output.

In = Boolean variable to be written to a real world output

#### Output Float — BASremote analog output.

In = Float variable to be written to a real world output

## Output Float Conditional — BASremote conditional analog output.

*In* = *Float* variable to be written to a real world output.

*Out = Float value currently written to real world output.* 

Enable = Boolean which indicates whether a write should occur. True will allow the write to occur and False will inhibit any writes.

Sedona will, normally, write the outputs from its logic every cycle. This can be an issue for some Modbus registers controlled by the BASremote. The writes to these registers can be controlled via the enable signal. If enable is false the Modbus register associated with this component will not be written.

SendEma CControls_BASR8M_Services::SendEmail	$\bigcirc$
Email Number	5
In	0.00
Enable	false

#### Send Email — BASremote email alert.

In = Float value to be included in email. Enable = Boolean used to indicate when to send an email. Email Number = which email to send (it must match the web configuration).

The BASremote can send an email using this component when the *Enable* signal is true. The email must be configured in the configuration webpage of the BASremote. When the email is sent, the text of the email will contain the current input float value. One Email will be sent on the false-to-true transition of the *Enable* signal.



#### BASremote Platform Kit (CControls\_BASR8M\_Platform)

 plat

 CControls
 BASR8M
 Platform::BASremotePlatformService

 Mem
 Available
 13520896

The BASremote platform kit has one component that advises the programmer how much usable memory is available for application programming. With a Linux platform, memory is seldom an issue.

The platform kit is found in the service folder.



#### BAScontrol20/22 I/O Kit (CControls\_BASC20\_IO) (CControls\_BASC22\_IO)

The BAScontrol20 IO kit provides several components necessary to interface Sedona logic to real world inputs and outputs on the BAScontrol20. In addition to 20 real I/O points, the BAScontrol20 accommodates 24 virtual points that can be treated as either inputs or outputs. Universal inputs and virtual points require configuration via a web browser. Other components are included in this kit that are BAScontrol20 hardware dependent.

AO1 – AO4	Analog Output	analog voltage output points
BI1 – BI4	Binary Input	binary input points
BO1 – BO6	Binary Output	binary output points (B01-B04 with the CControls_BASC20_IO)
UI1 – UI4	Universal Input	binary, analog voltage, thermistor, resistance or accumulator
UI5 – UI8	Universal Input	binary, analog voltage, thermistor or resistance
UC1 – UC4	Retentive Counters	up/down retentive universal counters
VT01 – VT24	Virtual Points	share data with BACnet/IP clients - first eight componenets are retentive
ScanTim	Scan Timer	monitors the time to execute Sedona logic

AO1 CControls BASC20 IO::AO1	$\bigcirc$
Inp F	0.00
Enable	false

#### AO1 – AO4 Analog Output — analog voltage output point.

Inp F = float value from 0–10 of respective point which translates to 0–10VDC output if Enable is true. If Enable is false, then output is controlled by a BACnet client.

BI1 CControls BASC20 IO:	:BI1 🔘
Out B	false

BO1 CControls BAS	C20	IO::BO1	
Inp B			false
Enable			false

UI4 CControls_BASC22_IO::UI4	0
Chn Type	Pulse
Out F	0.00
Out B	false
Out I	0
Reset	false

#### BI1 – BI4 Binary Input – binary input point.

Out B is true if input point is asserted to common; otherwise Out B is false.

#### BO1 – BO6 Binary Output — binary output point. (BO1-BO4 on BASC20)

Inp B = Boolean value of respective point which will translate to either a contact closure or triac output (on triac models).

If Inp B and Enable are true, the contact closure is made or the triac is turned on. If Enable is false, then output is controlled by a BACnet client.

## UI1 – UI8 Universal Input — binary, analog voltage, thermistor, resistance or accumulator point (UI1-UI4 can be accumulators).

*Out F* = float value of respective point if configured for analog input, thermistor, resistance or pulse accumulator.

If point is configured as a thermistor, or resistance, and an out-of-range condition is detected, Out F = the configured Out of Bounds value and Out B = true (thermistor or resistance fault)

Out B = Boolean value if configured for binary input. Out B is true if input point is asserted to common; otherwise Out B is false. If in Pulse mode and Reset =true, then Out F = 0. Out I = the integer representation of the float value.

VT01 – VT24 Virtual Points — wire sheet read or wire sheet write.

VT01 CControls_BASC20	IO::VT01	0
Chn Type	FloatIn	put
Reset	f	alse
Float V	(	0.00
Binary V	f	alse

VT02 CControls_BASC20	_IO::VT02
Chn Type	FloatOutput
Reset	false
Float V	0.00
Binary V	false

Virtual points are used to share wire sheet data with a BACnet/IP client. A BACnet/IP client can "read" wire sheet data such as a calculated value or it can "write" to the wire sheet with a set-point or command. Virtual points are first configured from a web page to be a BACnet binary value (BV) or BACnet analog value (AV). The BACnet description field and units of measure can be set as well as the BACnet name which must be unique within the device. Next go to Workbench to change the wire sheet Read or Write directions. The title of the virtual point on the web page will change to Wire Sheet Write or Wire Sheet Read accordingly. The four possibilities are shown on the left labelled as VT01 through VT04.

**VT01** is configured as analog variable, wire sheet write, which results in the component being a *FloatInput*.

VT03 CControls BASC20	0 IO::VT03
Chn Type	BinaryInput
Reset	false
Float V	0.00
Binary V	false

VT02 is configured as analog variable, wire sheet read, which results	3
in the component being a <i>FloatOutput</i> .	

**VT03** is configured as binary variable, wire sheet write, which results in the component being a *BinaryInput*.

VT04		0
CControls	BASC20_IO::VT04	9
Chn Type	BinaryO	utput
Reset		false
Float V		0.00
Binary V		false

**VT04** is configured as binary variable, wire sheet read, which results in the component being a *BinaryOutput*.

If configured as a *FloatInput,* then *Float V* represents the value written by the BACnet/IP client which can be used by other wire sheet components

If configured as a *FloatOutput,* then *Float V* represents a value from a wire sheet component that can be read by the BACnet/IP client

If configured as a *BinaryInput*, then *Binary V* represents the value written by the BACnet/IP client which can be used by other wire sheet components

If configured as a *BinaryOut,* then *Binary V* represents a value from a wire sheet component that can be read by the BACnet/IP client

Asserting *Reset* will clear the component. It is usually kept in the *False* state.

ScanTim CControls_BASC20_IO::ScanTim	0
Time Ms	73
Minimum Ms	71
Maximum Ms	77
Average Ms	71

#### ScanTimer – monitors the execution time of Sedona logic.

The scan timer monitors the current, minimum, maximum and average time it takes to execute a single scan of Sedona logic. All outputs are integers. The average time is based upon the last ten samples. The result of which becomes the first value in the next ten samples. The component can be reset by right-clicking the component and invoking an Action.

UC1 CControls_BASC20_IO::UC1	$\bigcirc$
Count	179
Count F	179.00
Ovf	false
Clk	true
Enable	true
Rst	false
C Dwn	false
Hold At Limit	true

#### UC1 – UC4 — retentive up/down universal counters.

Counts on the false to true transition of *Clk* if *Enable* is *true*. If *C Dwn* is *true*, counting is down until zero is reached. If *C Dwn is false*, counting is up to the limit of the counter (2147483647) before it rolls over to zero. If *Hold At Limit* is set to true, the counter will stop counting at the value set in the *Limit* slot on the property page. The *Ovf* flag is set *true* when the value of status equals or exceeds the limit value. The output *count* value can be displayed as an integer (*Count*) or a float (*Count F*). *Rst* set to *true* clears the counter and prevents further counting.

#### BAScontrol20/22 Platform Kit (CControls\_BASC20\_Platform) (CControls\_BASC22\_Platform)

plat CControls BASC20 Platform::BASC20PlatformService	
Mem Available	9352
plat	
CControls BASC22 Platform::BAS22PlatformService	
Mem Available	9352

The BAScontrol20/22 platform kit has only one component that advises the programmer how much usable memory is available for application programming. It is recommended that the usable memory not fall below 8,192 bytes. It can be found in the services folder and can be copied onto the wire sheet. The output type of this component is a *Long*.



#### BAScontrol20 Web Kit (CControls\_BASC20\_Web)

#### WC01 – WC48 Web Components — share data with BAScontrol20 web pages.

Web components provide a convenient method of sharing data between web pages and the wire sheet without the need of the Workbench tool. In this kit there are 48 web components that must be first configured via web pages. Web components can be configured to read wire sheet data or can write wire sheet data. The four possibilities are shown on the left labeled as WC01 through WC04.

WC01	V
CControls_BASC20_Web::WC01	
Wc Type Input	0
Fit Val 0.00	
Int Val 0	
Bin Val false	

**WC01** is configured as an input which results in the component being an *Input*.

WC02 CControls_BASC20	Web::WC02
Wc Type	FloatOutput
Flt Val	0.00
Int Val	0
Bin Val	false

**WC02** is configured as an output float which results in the component being a *FloatOutput*.

WC03 CControls_BASC2	20_Web::WC03
Wc Type	IntegerOutput
Fit Val	0.00
Int Val	0
Bin Val	false

**WC03** is configured as output integer which results in the component being an *IntegerOutput*.

WC04 CControls_BASC2	0_Web::WC04
Wc Type	BinaryOutput
Fit Val	0.00
Int Val	0
Bin Val	false

**WC04** is configured as an output binary which results in the component being a *BinaryOutput*.

If configured as an Input then Flt Val, Int Val, and BinVal represents the value written by a web page which can be used by other wire sheet components

If configured as a FloatOutput, then Flt Val represents a value from a wire sheet component that can be read by a web page

*If configured as an IntegerOutput, then Int Val represents a value from a wire sheet component that can be read by a web page* 

If configured as a BinaryOutput, then Bin Val represents a value from a wire sheet component that can be read by a web page

#### Contemporary Controls Function Kit (CControls\_Function)

These components apply to any Sedona Virtual Machine (SVM).

Cand2 CControls_Function::Cand2	0
Inp1	false
Inp2	false
Out	false
Out Not	true

**Two-input Boolean product – two-input AND/NAND gate.**   $Out = In1 \cdot In2$  $Out Not = \overline{Out}$ 

Cand4 CControls_Function::Cand4	0
Inp1	false
Inp2	false
Inp3	false
Inp4	false
Out	false
Out Not	true

Four-input Boolean product – four-input AND/NAND gate.
Out = In1 • In2 • In3 • In4
$Out Not = \overline{Out}$

Cand6 CControls_Function::Cand6	0
Inp1	false
Inp2	false
Inp3	false
Inp4	false
Inp5	false
Inp6	false
Out	false
Out Not	true

Six-input Boolean product – six-input AND/NAND gate.  $Out = In1 \cdot In2 \cdot In3 \cdot In4 \cdot In5 \cdot In6$  $Out Not = \overline{Out}$ 

	0	Eight-input Boolean product – eight-input AND/NAND gate.
d8	false	Out = In1 • In2 • In3 • In4 • In5 • In6 • In7 • In8
	false	$Out Not = \overline{Out}$

Cand8 CControls_Function::Cand8	0
Inp1	false
Inp2	false
Inp3	false
Inp4	false
Inp5	false
Inp6	false
Inp7	false
Inp8	false
Out	false
Out Not	true
	$\overline{}$

Cor2 CControls_Function::Cor2	0
Inp1	false
Inp2	false
Out	false
Out Not	true

#### **Two-input Boolean sum – two-input OR/NOR gate** *Out = In1 | In2 Out Not = Out*

Cand4 CControls_Function::Cand4	0
Inp1	false
Inp2	false
Inp3	false
Inp4	false
Out	false
Out Not	true

#### Four-input Boolean sum – four-input OR/NOR gate

Out = In1 | In2 | In3 | In4 $Out Not = \overline{Out}$ 

Cand6 CControls_Function::Cand6	0
Inp1	false
Inp2	false
Inp3	false
Inp4	false
Inp5	false
Inp6	false
Out	false
Out Not	true

#### Six-input Boolean sum – six-input OR/NOR gate

Out = In1 | In2 | In3 | In4 | In5 | In6 $Out Not = \overline{Out}$ 

Cor8 CControls_Function::Cor8	0
Inp1	false
Inp2	false
Inp3	false
Inp4	false
Inp5	false
Inp6	false
Inp7	false
Inp8	false
Out	false
Out Not	true

#### **Eight-input Boolean sum – eight-input OR/NOR gate** Out = ln1 + ln2 + ln3 + ln4 + ln5 + ln6 + ln7 + ln8

Out = In1 | In2 | In3 | In4 | In5 | In6 | In7 | In8 $Out Not = \overline{Out}$ 

Cmt	C
CControls_Function::Cmt	A
Comment The Comment component allows up to 64 characters to be displaye	d 1.

#### **Comment** A comment field from 1-64 characters used for documentation purposes.

*Out* = 9/5 \* *In* + 32

Dff CControls_Function::Dff	0
Preset	false
Reset	false
D	false
Clk	false
Out	false
Out Not	true

"D" Flip-Flop – D-style Edge-triggered Single-bit Storage If Preset = True and Reset = False then Out = True If Reset = True then Out = False regardless of all other inputs On the rising edge of Clk with Preset = False and Reset = False; If D = false then Out = false If D= true then Out = true Out Not = Out

FtoC CControls_Function::FtoC	0
In Temp Deg F	32.00
Out Temp Deg C	0.00

CtoF CControls_Function::CtoF	0
In Temp Deg C	100.00
Out Temp Deg F	212.00

HLpre CControls_Function::HLpre	0
Out	true
Out Not	false

PsychrE CControls_Function::PsychrE	$\bigcirc$
In Temp Deg F	70.00
In Relative Humidity Pct	50.00
Out Dew Point Deg F	50.56
Out Enthalpy Btu _per _lb	25.29
Out Sat Pressure _psi	0.36
Out Vapor Pressure _psi	0.18
Out Wet Bulb Temp Deg F	58.75
	1

PsychrS CControls_Function::PsychrS	0
In Temp Deg C	21.11
In Relative Humidity Pct	50.00
Out Dew Point Deg C	10.31
Out Enthalpy _k J _per _kg	40.99
Out Sat Pressure _k Pa	2.50
Out Vapor Pressure _k Pa	1.25
Out Wet Bulb Temp Deg C	14.86

°C to °F – Celsius to Fahrenheit Temperature Conversion Out = 5/9 \* (In - 32)

°F to °C – Fahrenheit to Celsius Temperature Conversion

**High – Low Preset – defined logical true and false states** *Out = true Out Not = false* 

#### **Psychrometric Calculator – English Units**

Inputs are Dry-bulb temperature (°F) and Relative Humidity (%) Outputs are Dew Point (°F), Enthalpy (Btu/lb), Saturation Pressure (psi), Vapor Pressure (psi) and Wet-bulb temperature (°F)

*Input temperature range* 32-120°*F; Input relative humidity range* 10-100%

#### **Psychrometric Calculator – SI Units**

Inputs are Dry-bulb temperature (°C) and Relative Humidity (%) Outputs are Dew Point (°C), Enthalpy (kJ/kg), Saturation Pressure (kPa), Vapor Pressure (kPa) and Wet-bulb temperature (°C)

Input temperature range 0-48.8 °C; Input relative humidity range 10-100%



**Simplified Psychrometric Chart** 

A simplified psychrometric chart greatly removes the detail of a professional chart. On the X-axis is the dry-bulb temperature with a typical range from 32°F to 120°F. This is the same temperature you measure with a thermometer or wallmounted thermostat. Lines of constant dry-bulb temperature are for all practical purposes vertical. On the Y-axis is the humidity ratio (lbw/lba) in lbs-water vapor to lbs-air ranging from zero to over 0.028. Lines of constant humidity ratio are horizontal. The left curved heavy line is called the saturation line indicating 100% saturation of water vapor or 100% relative humidity. Curves of lesser relative humidity would exist to the right of the saturation line. Along the saturation line you can

SCLatch CControls_Function::SCLatch	0
Set	false
Clear	false
Out	false
Out Not	true

determine both dew point temperature and wetbulb temperature although their lines of constant temperature are different. For dew point, the lines are horizontal while the lines of constant wetbulb are diagonal and almost parallel with lines of constant enthalpy.

Looking at the PsychrE component and the simplified chart we can study one example. Notice in the component that the two inputs are 70°F dry-bulb and 50% relative humidity. With these two values a single point on the psychrometic chart can be located. If you follow the horizontal line to the left you can determine the dew point temperature and to the right the humidity ratio. If you follow the diagonal line to the upper-left you can learn the wet-bulb and enthalpy values. We still have not determined the saturation pressure or the vapor pressure but these values can be derived with help from the humidity ratio. The PsychrE can make the calculations in the English system and the PsychrS can make the calculations in the SI system. Although simple conversions can be made between the two systems or to reflect the output values in different units of measure. be careful when working with enthalpy. With the English system, the change in enthalpy is referenced to a 0°F while in the SI system the reference is 0°C so a straight forward conversion between the two systems is not possible. Also note the limited range of the two psychrometric components. Both components are limited to an equivalent input range of 0-120°F dry-bulb and 10-100% relative humidity.

Set/Clear Latch – single-bit level-triggered single-bit data storage The following logic applies to the state of Set or Clear: If Set is true and Clear is false, then Out = trueIf Clear is true, then Out = false regardless of the state of Set Out Not = Out

## **Component–Kit Association**

Add2mathAdd4mathAdd4logicAnd2logicAnd4logicAnd4logicAnd4logicAnd4logicAO1-AO4CControls_BASC20_IO, CControls_BASC22_IOASW4logicAsymamathAvgNmathB2FlogicB1-B14CControls_BASC20_IO, CControls_BASC22_IOBO1-B06 (BO1-B04 on BASC20)CControls_BASC20_IO, CControls_BASC22_IOBASC2PlatformServiceCControls_BASC20_PlatformBASC2PlatformServiceCControls_BASC20_PlatformBASC2PlatformServiceCControls_BASC20_PlatformBASC2PlatformServiceCControls_BASC20_PlatformBASC2PlatformServiceCControls_BASC20_PlatformBASC2PlatformServiceCControls_FASC30_PlatformBASC2PlatformServiceCControls_FUNCTIONCand2CControls_FunctionCand3CControls_FunctionCand4CControls_FunctionCand5CControls_FunctionConstFicatypesCord2CControls_FunctionCord3CControls_FunctionCord4CControls_FunctionCord5CControls_FunctionCord6CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls	Component	Sedona Palette Folder
Add4mathADemux2logicAnd2logicAnd4logicAnd4logicAN1-AO4Controls_BASC2_IO, Controls_BASC22_IOASWlogicASW4logicASW4logicASW4logicASW4logicASW4logicASW4logicASW4logicASW4logicASW4logicB2FlogicB11-B14Controls_BASC20_IO, Controls_BASC22_IOB01-B06 (B01-B04 on BASC20)CControls_BASC20_IPlatformBASC22PlatformServiceCOntrols_BASC20_PlatformBASC22PlatformServiceCControls_BASC20_PlatformBASC22PlatformServiceCControls_BASC20_PlatformBASC22PlatformServiceCControls_BASC30_PlatformBASC22PlatformServiceCControls_FunctionCand4CControls_FunctionCand4CControls_FunctionCand4CControls_FunctionCand6CControls_FunctionCand7CControls_FunctionConstFloatypesConstFloatypesConstFloatypesConstFloatypesCordCControls_FunctionCordCControls_FunctionCordCControls_FunctionCordCControls_FunctionCordCControls_FunctionCordCControls_FunctionCordCControls_FunctionCordCControls_FunctionCordCControls_FunctionCordCContr	Add2	math
ADemux2logicAnd2logicAnd4logicAnd4logicAO1-AO4CControls_BASC20_IO, CControls_BASC22_IOASWlogicASW4logicAsyn4logicArg10mathAvgNmathB2FlogicB1-B14CControls_BASC20_IO, CControls_BASC22_IOB01-B06 (B01-B04 on BASC20)CControls_BASC20_IO, CControls_BASC22_IOBASC22PlatformServiceCControls_BASC20_IO, CControls_BASC22_IOBASC22PlatformServiceCControls_BASC22_IOBASC22PlatformServiceCControls_BASC22_IOBASC22PlatformServiceCControls_BASC22_IOBASC22PlatformServiceCControls_BASC22_IOBASC22PlatformServiceCControls_FARC22_PlatformBASC22PlatformServiceCControls_FunctionCand2CControls_FunctionCand3CControls_FunctionCand4CControls_FunctionCand5CControls_FunctionCand6CControls_FunctionConstBooltypesConstBooltypesConstBooltypesControls_FunctionControls_FunctionConstBooltypesControls_FunctionControls_FunctionConstBooltypesControls_FunctionControls_FunctionControls_FunctionControls_FunctionControls_FunctionControls_FunctionDailyScheduleFloatbasicScheduleDailyScheduleFloatbasicScheduleDailyScheduleFloattimingF21 <td>Add4</td> <td>math</td>	Add4	math
And2logicAnd4logicAO1-AO4Controls_BASC20_IO, CControls_BASC22_IOASWlogicASW4logicASW4logicASW4logicASW4logicASW4logicASW4logicASW4logicArg10mathAvg10mathB2FlypesB2FlogicB11-B14CControls_BASC20_IO, CControls_BASC22_IOBO1-BO6 (B01-BO4 on BASC20)CControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC20_PlatformBASC20PlatformServiceCControls_FunctionBASC20PlatformServiceCControls_FunctionCand2CControls_FunctionCand4CControls_FunctionCand5CControls_FunctionCand6CControls_FunctionConstFloatypesCord2CControls_FunctionCord3CControls_FunctionCord4CControls_FunctionCord5CControls_FunctionCord6CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionDailyScheduleFloatbasicScheduleDailyScheduleFloatbasicScheduleDailyScheduleFloatlogicDifdCControls	ADemux2	logic
And4logicAO1-AO4CControls_BASC20_IO, CControls_BASC22_IOASWlogicASW4logicAvg10mathB2FtypesB2PlogicB11-B14CControls_BASC20_IO, CControls_BASC22_IOBO1-BO6 (B01-B04 on BASC20)CControls_BASC20_IO, CControls_BASC22_IOBASc2DilatformServiceCControls_BASC20_IO, CControls_BASC22_IOBASc2DilatformServiceCControls_BASC20_ID, CControls_BASC22_IOBASc2DilatformServiceCControls_BASC20_ID, CControls_BASC22_IOBAScapitatomServiceCControls_BASC22_PlatformBASremotePlatformServiceCControls_BASRM_PlatformBASremotePlatformServiceCControls_FunctionCand4CControls_FunctionCand5CControls_FunctionCand6CControls_FunctionCand6CControls_FunctionConstBooltypesConstBooltypesCors2CControls_FunctionCors4CControls_FunctionCord5Controls_FunctionCord6CControls_FunctionCord7CControls_FunctionCord8CControls_FunctionCord7CControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleBoolbasicScheduleDailyScheduleBoolbasicScheduleDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDailyScheduleBoolbasicScheduleDiv2mathFigtimingFigtypesFiladDffset	And2	logic
AO1-AO4CControls_BASC20_IO, CControls_BASC22_IOASWlogicASW4logicAvg10mathArg10mathArg10mathArg10mathArg10mathArg10mathArg10mathArg10mathArg10mathB2FlogicB1H=B14CControls_BASC20_IO, CControls_BASC22_IOBO1-BO6 (B01-B04 on BASC20)CControls_BASC20_IO, CControls_BASC22_IOBASC20PlatformServiceCControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC20_PlatformBSWlogicCand2CControls_FunctionCand4CControls_FunctionCand6CControls_FunctionCand6CControls_FunctionCand7Crontrols_FunctionControls_FunctionfuncCord2CControls_FunctionCord3CControls_FunctionCord4CControls_FunctionDailyScheduleFloatbasicScheduleDailyScheduleFloatbasicScheduleDailyScheduleFloatbasicScheduleDailyScheduleFloatbasicScheduleDailyScheduleFloatbasicScheduleDailyScheduleFloatbasicScheduleDailyScheduleFloatbasicScheduleDig1CfuncFreqfuncFloatCoffsetmath <td>And4</td> <td>logic</td>	And4	logic
ASWlogicASW4logicASV4logicAvg10mathAvgNmathB2FtypesB2PlogicB11-B14CControls_BASC20_IO, CControls_BASC22_IOB01-B06 (B01-B04 on BASC20)CControls_BASC20_IO, CControls_BASC22_IOBASC2PlatformServiceCControls_BASC20_IPLatformBASC22PlatformServiceCControls_BASC20_PlatformBASC22PlatformServiceCControls_BASC20_PlatformBASC22PlatformServiceCControls_BASC20_PlatformBASC22PlatformServiceCControls_BASC8M_PlatformBSWlogicCand2CControls_FunctionCand4CControls_FunctionCand6CControls_FunctionCand6CControls_FunctionConstFloatypesConstFloatypesCord7Controls_FunctionCord8CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord8CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionCord7CControls_FunctionDailyScheduleBoolbaisCScheduleDailyScheduleFloatbaisCScheduleDailyScheduleFloatbaisCScheduleDailyScheduleFloatbaisCScheduleDiv2mathFieldOffset<	AO1–AO4	CControls BASC20 IO, CControls BASC22 IO
ASW4logicArg10mathB2FtypesB2FlogicB1-B14Controls_BASC20_IO, CControls_BASC22_IOB01-B06 (B01-B04 on BASC20)CControls_BASC20_IO, CControls_BASC22_IOBASC20PlatformServiceCControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC22_PlatformBASC20PlatformServiceCControls_BASC22_PlatformBASC20PlatformServiceCControls_BASC22_PlatformBASC20PlatformServiceCControls_BASC22_PlatformBASC20PlatformServiceCControls_BASC22_PlatformBASC20PlatformServiceCControls_BASC22_PlatformBASC20PlatformServiceCControls_BASC22_PlatformBASC20PlatformServiceCControls_BASC22_PlatformBASC20PlatformServiceCControls_BASC22_PlatformBASC20PlatformServiceCControls_BASC22_PlatformBASC20PlatformServiceCControls_FunctionCand4CControls_FunctionCand5CControls_FunctionContR0TypesCor2CControls_FunctionCor4CControls_FunctionCor4CControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDailyScheduleFloatbasicScheduleDailyScheduleFloatbasicScheduleDi/SCmathDi/SCmathDi/SCmathDi/SCMathDi/SCMathDi/SCMathDi/SCMathDi/SC <t< td=""><td>ASW</td><td>logic</td></t<>	ASW	logic
Avg10mathAvgNmathAvgNmathB2FtypesB2PlogicB11-B14CControls_BASC20_IO, CControls_BASC22_IOBO1-B06 (B01-B04 on BASC20)CControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC20_PlatformBAScentePlatformServiceCControls_BASC20_PlatformBAScentePlatformServiceCControls_BASC20_PlatformBASremotePlatformServiceCControls_FunctionCand2CControls_FunctionCand4CControls_FunctionCand8CControls_FunctionCand8CControls_FunctionConstBooltypesConstBooltypesCordControls_FunctionCord2CControls_FunctionCord3CControls_FunctionCord4CControls_FunctionCord5Controls_FunctionCord6Controls_FunctionCord7Controls_FunctionCord8CControls_FunctionCord7Controls_FunctionCountfuneCtoFControls_FunctionDailyScheduleFloatbasicScheduleDailyScheduleFloatbasicScheduleDailyScheduleFloatbasicScheduleDiv2mathDiv2mathDiv3mathFi2BtypesFloatOffsetmathFreqfuncFreqfuncFreqfuncFloatOffsetmathFreqfunc	ASW4	logic
AvgN         math           B2F         types           B2P         logic           B11-B14         CControls_BASC20_IO, CControls_BASC22_IO           BO1-BO6 (B01-B04 on BASC20)         CControls_BASC20_Platform           BASC20PlatformService         CControls_BASC20_Platform           BASC20PlatformService         CControls_BASC22_Platform           BASC20PlatformService         CControls_BASC22_Platform           BASC20         CControls_BASC20_Platform           Cand2         CControls_Function           Cand3         CControls_Function           ConstFloa         Vpges           Cord         CControls_Function           Cord         CControls_Function           Cord         CControls_Function           DaliyS	Avg10	math
B2FtypesB2PlogicB11-B14CControls_BASC20_IO, CControls_BASC22_IOB01-B06 (B01-B04 on BASC20)CControls_BASC20_PlatformBASC20PlatformServiceCControls_BASC22_PlatformBASC22PlatformServiceCControls_BASC22_PlatformBASC22PlatformServiceCControls_BASC22_PlatformBASC20PlatformServiceCControls_BASC8M_PlatformBASremotePlatformServiceCControls_FunctionCand2CControls_FunctionCand4CControls_FunctionCand6CControls_FunctionCand7funcCmtCControls_FunctionCand8CControls_FunctionConstBooltypesConstBooltypesCors10Controls_FunctionConstBooltypesCors2CControls_FunctionCors8CControls_FunctionCountfuncCtoFCControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDailyScheduleFloatlogicDiv2mathDiyOfftimingDiyOntimingF2BtypesF2ItypesF2ItypesF2ItypesF2ItypesF1actOffsetmathFreqfuncFreqfuncF1actOffsetmathFreqfuncF1actOffsetmathF1actOffsetfuncF1actOffsetfuncF1actOffsetfunc <tr< td=""><td>AvgN</td><td>math</td></tr<>	AvgN	math
B2P     logic       B11-B14     CControls_BASC20_IO, CControls_BASC22_IO       B01-B06 (B01-B04 on BASC20)     CControls_BASC20_IPlatform       BASC20PlatformService     CControls_BASC20_Platform       BASC22PlatformService     CControls_BASC22_Platform       BASC20PlatformService     CControls_BASC22_Platform       BAScanotePlatformService     CControls_BASC22_Platform       BAScanotePlatformService     CControls_BASC22_Platform       BASCanotePlatformService     CControls_Function       Cand2     CControls_Function       Cand4     CControls_Function       Cand6     CControls_Function       Cand7     Controls_Function       Cand8     CControls_Function       ConstBool     types       Cortrols_Controls_Function     Controls_Co	B2F	types
BI1-BI4CControls_BASC20_IO, CControls_BASC22_IOBO1-BO6 (BO1-BO4 on BASC20)CControls_BASC20_IO, CControls_BASC22_IOBASC20PlatformServiceCControls_BASC22_PlatformBASC22PlatformServiceCControls_BASC22_PlatformBASC22PlatformServiceCControls_BASC22_PlatformBASC22PlatformServiceCControls_FunctionCand2CControls_FunctionCand4CControls_FunctionCand6CControls_FunctionCand8CControls_FunctionCand8CControls_FunctionConstBooltypesConstBooltypesControls_Controls_FunctionControls_ControlDiv2math </td <td>B2P</td> <td>logic</td>	B2P	logic
BO1-BO6 (B01-BO4 on BASC20)CControls_BASC22_IO, CControls_BASC22_IOBASC20PlatformServiceCControls_BASC22_PlatformBASremotePlatformServiceCControls_BASRM_PlatformBSWlogicCand2CControls_FunctionCand4CControls_FunctionCand8CControls_FunctionCand8CControls_FunctionCand8CControls_FunctionConstBooltypesConstFloatypesControls_FunctionControls_FunctionConstBooltypesConstFloaCControls_FunctionCord6CControls_FunctionCord7CControls_FunctionConstBooltypesCord7CControls_FunctionCord8CControls_FunctionCord8CControls_FunctionCord1funcCtoFCControls_FunctionCotrols_FunctionDemux1284DataTimeServicedatetimeDemux1284logicDffCControls_FunctionDiv2mathDiv2mathFagtypesF21typesF22typesF23typesF24funcFreqfuncFreqfuncFloaCofficationControls_FunctionHyperCControls_FunctionHyperFinalDiv2mathFinaltypesF24typesF25funcFinaltypesF26CControls_FunctionHyper <t< td=""><td>BI1–BI4</td><td>CControls BASC20 IO, CControls BASC22 IO</td></t<>	BI1–BI4	CControls BASC20 IO, CControls BASC22 IO
BASC20PlatformServiceC Controls_BASC22_PlatformBASC22PlatformServiceC Controls_BASR8M_PlatformBASremotePlatformServiceC Controls_BASR8M_PlatformBSWlogicCand2C Controls_FunctionCand4C Controls_FunctionCand6C Controls_FunctionCand7funcControls_FunctionControls_FunctionCand8C Controls_FunctionControls_FunctionControls_FunctionControls_FunctionControls_FunctionControls_FunctionControls_FunctionConstBooltypesCors4C Controls_FunctionCord7C Controls_FunctionCord8C Controls_FunctionCountfuncCtoFC Controls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDattimeServicedatetimeDemux12B4logicDlyOfftimingDlyOfftimingDlyOfftimingF2BtypesF2ItypesF2ItypesF2IfuncFitaCCControls_FunctionHupeControls_FunctionHupreControls_FunctionHupreControls_FunctionHuprefuncFitaCControls_FunctionHupreInnoHupreControls_FunctionHupreInnoHypesFunctionHupreControls_FunctionHupreInnoHup PolIn	BO1–BO6 (BO1-BO4 on BASC20)	CControls BASC20 IO, CControls BASC22 IO
BASC22PlatformServiceCControls_BASC22_PlatformBASremotePlatformServiceCControls_BASR8M_PlatformBSWlogicCand2CControls_FunctionCand4CControls_FunctionCand6CControls_FunctionCand7funcCand8CControls_FunctionCand8CControls_FunctionConstBooltypesConstBooltypesCorstCoresCControls_FunctionCoresCControls_FunctionCoresCControls_FunctionCoresCControls_FunctionCoresCControls_FunctionCoresCControls_FunctionCoresCControls_FunctionCoresCControls_FunctionCoresCControls_FunctionDateTimeServicedatetimeDateTimeServicedatetimeDemux12B4logicDlyOfftimingDlyOfftimingDlyOfftimingF2BtypesF2IfuncFreqfuncFreqfuncFreqfuncFreqfuncFloCCControls_FunctionHypesfuncFloatOffsetmathFreqfuncFloatOffsetfuncFloatOffsetfuncFloatOffsetfuncFloatOffsetfuncFloatOffsetfuncHypesfuncFloatOffsetfuncHypesfuncFloatOffsetfuncHypesfunc	BASC20PlatformService	CControls BASC20 Platform
BASremotePlatformServiceCControls_BASR8M_PlatformBSWlogicCand2CControls_FunctionCand4CControls_FunctionCand6CControls_FunctionCand8CControls_FunctionCmprfuncCmtCControls_FunctionConstBooltypesCord72CControls_FunctionCord8CControls_FunctionCord8CControls_FunctionCord9CControls_FunctionCord9CControls_FunctionCord9CControls_FunctionCord9CControls_FunctionCord9CControls_FunctionCord9CControls_FunctionCord9CControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleBoolbasicScheduleDateTimeServicedatetimeDemux12B4logicDffCControls_FunctionDiv2mathDlyOnftimingDlyOntimingF2BtypesFloatOffsetmathFreqfuncFreqfuncFreqfuncFreqfuncFreqfuncHyperesisfuncIpBoolCControls_BASR8M_Services	BASC22PlatformService	CControls BASC22 Platform
BSWlogicCand2CControls_FunctionCand4CControls_FunctionCand6CControls_FunctionCand8CControls_FunctionCmrfuncCmtCControls_FunctionConstBooltypesCostBooltypesCord2CControls_FunctionCord3CControls_FunctionCord4Controls_FunctionCord5CControls_FunctionCord5CControls_FunctionCord6CControls_FunctionCord7CControls_FunctionCord8CControls_FunctionCord7CControls_FunctionCountfuncCtoFCControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasiCScheduleDateTimeServicedatetimeDemuxl2B4logicDlyOfftimingDlyOfftimingDlyOntimingF2BtypesF2ItypesF10atOffsetmathFreqfuncFreqfuncFreqfuncFreqfuncFreqfuncHyperesisfuncIpBoolCControls_BASR8M_Services	BASremotePlatformService	CControls BASR8M Platform
Cand2CControls_FunctionCand4CControls_FunctionCand6CControls_FunctionCand8CControls_FunctionCmprfuncCmtCControls_FunctionConstBooltypesCor2CControls_FunctionCor8CControls_FunctionCor8CControls_FunctionCor8CControls_FunctionCor8CControls_FunctionCor8CControls_FunctionCor8CControls_FunctionCor8CControls_FunctionCountfuncCtoFCControls_FunctionDailyScheduleFloatbasicScheduleDateTimeServicedatetimeDemuxl2B4logicDiv2mathDiyOntimingF28typesF21typesFloatOffsetmathFreqfuncFreqfuncFreqfuncFreqfuncFreqfuncFreqfuncFreqfuncFreqfuncFreqfuncFreqfuncFreqfuncFreqfuncFreqfuncFreqfuncFloatOffsetfuncHysteresisfuncIpBoolCControls_BASR8M_Services	BSW	
Cand4CControls_FunctionCand6CControls_FunctionCand8CControls_FunctionCmprfuncConstBooltypesConstFloatypesCor2CControls_FunctionCor3CControls_FunctionCor4CControls_FunctionCor4CControls_FunctionCor5CControls_FunctionCor6CControls_FunctionCor7Controls_FunctionCountfuncCtoFCControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDateTimeServicedatetimeDemux12B4logicDiffCControls_FunctionDiv2mathDiv2mathF2BtypesF2ItypesF2ItypesF2AfuncFreqfuncFreqfuncFloatOffsetmathFreqfuncFloatOffsetfuncFloatOffsetfuncFloatOffsetmathFreqfuncFloatOffsetfuncFloatOffsetfuncFloatOffsetfuncFloatOffsetfuncFloatOffsetfuncHysteresisfuncIzFtypesIpBoolCControls_BASR8M_Services	Cand2	CControls Function
Cand6CControls_FunctionCand8CControls_FunctionCmprfuncCmtCControls_FunctionConstBooltypesConstBooltypesConstFloaCControls_FunctionCora2CControls_FunctionCora4CControls_FunctionCora5CControls_FunctionCountfuncCtoFCControls_FunctionDailyScheduleBoolbasicScheduleDateTimeServicedatetimeDemuxl2B4logicDlyOfftimingDlyOfftimingDlyOfftimingF2BtypesF21typesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionItpBoolCControls_Function	Cand4	CControls Function
Cand8CControls_FunctionCmprfuncCmtCControls_FunctionConstBooltypesConstFloatypesCor2CControls_FunctionCor3CControls_FunctionCor4CControls_FunctionCountfuncCtoFCControls_FunctionDailyScheduleBoolbasicScheduleDaityScheduleFloatbasicScheduleDateTimeServicedatetimeDemuxl284logicDlyOfftimingDlyOfftimingF2BtypesF21typesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncIpBoolCControls_Function	Cand6	CControls Function
CmprfuncCmtCControls_FunctionConstBooltypesConstFloatypesCor2CControls_FunctionCor8CControls_FunctionCountfuncCtoFCControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDateTimeServicedatetimeDemuxl2B4logicDlyOfftimingDlyOfftimingDlyOfftimingF2BtypesF2ItypesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHustersisfuncF2FtypesIpBoolCControls_FunctionHpBoolCControls_Function	Cand8	CControls Function
CmtCControls_FunctionConstBooltypesConstFloatypesCor2CControls_FunctionCor8CControls_FunctionCountfuncCtoFCControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDateTimeServicedatetimeDemuxl2B4logicDiyOfftimingDiyOfftimingDiyOfftimingF2BtypesF2ItypesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHLprefuncFtoSfuncFtoGCControls_FunctionHLprefuncFtoGCControls_FunctionHLprefuncFtoGCControls_FunctionHLprefuncIpBoolCControls_BASR8M_Services	Cmpr	func
ConstBooltypesConstFloatypesCor2CControls_FunctionCor8CControls_FunctionCountfuncCtoFCControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDemuxI2B4logicDiyOfftimingDiyOfftimingF2BtypesF2ItypesFloatOffsetmathFreqfuncFreqfuncFreqfuncFloatOffsetmathHupreCControls_FunctionItpsfuncFloatOffsetmathFloatOffsetfuncFloatOffsetfuncFloatOffsetfuncFloatOffsetfuncFloatOffsetfuncIppBoolCControls_FARSMS_Services	Cmt	CControls Function
ConstFloatypesConstFloaCControls_FunctionCor2CControls_FunctionCor8CControls_FunctionCountfuncCtoFCControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDateTimeServicedatetimeDemuxI2B4logicDi/2mathDi/OfftimingDi/OfftimingF2BtypesF2ItypesFloatOffsetmathFreqfuncFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHlprefuncFtoCCControls_FunctionHlprefuncInpBoolCControls_Function	ConstBool	types
Cor2CControls_FunctionCor3CControls_FunctionCountfuncCtoFCControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDateTimeServicedatetimeDemuxl2B4logicDffCControls_FunctionDiv2mathDlyOfftimingDlyOntimingF2BtypesFloatOffsetmathFreqfuncFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncInpBoolCControls_BASR8M_Services	ConstFloa	types
Cor8CControls_FunctionCountfuncCtoFCControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDateTimeServicedatetimeDemuxl2B4logicDffCControls_FunctionDiv2mathDlyOfftimingDlyOfftimingF2BtypesF2ItypesFloatOffsetfuncFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHLprefuncIpBoolfuncIpBoolfuncLpsondfunc </td <td>Cor2</td> <td>CControls Function</td>	Cor2	CControls Function
CountfuncCtoFCControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDateTimeServicedatetimeDemuxl2B4logicDffCControls_FunctionDiv2mathDlyOfftimingDlyOntimingF2BtypesF2ItypesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHzprefuncFtoSfuncFloatOffsetfuncFtoCCControls_FunctionHLprefuncHpBoolCControls_BASR8M_Services	Cor8	CControls Function
CtoFCControls_FunctionDailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDateTimeServicedatetimeDemuxl2B4logicDffCControls_FunctionDiv2mathDlyOfftimingDlyOntimingF2BtypesF2ItypesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHLprefuncFloatOffsetfunc </td <td>Count</td> <td>func</td>	Count	func
DailyScheduleBoolbasicScheduleDailyScheduleFloatbasicScheduleDateTimeServicedatetimeDemuxl2B4logicDffCControls_FunctionDiv2mathDlyOfftimingDlyOntimingF2BtypesF2ItypesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHLprefuncHysteresisfuncIpBoolCControls_SASR8M_Services	CtoF	CControls Function
DailyScheduleFloatbasicScheduleDateTimeServicedatetimeDemuxl2B4logicDffCControls_FunctionDiv2mathDlyOfftimingDlyOntimingF2BtypesF2ItypesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHLprefuncFloatOffsetfuncHlpreCControls_FunctionHLprefuncFloatOffsetfuncHlpreCControls_FunctionHLprefuncFloatOffsetfuncHlpreCControls_FunctionHlprefuncFloatOffsetfuncHlpreCControls_FunctionHlprefuncHoteresisfuncIpBoolCControls_BASR8M_Services	DailvScheduleBool	basicSchedule
Date Time ServicedatetimeDemuxl2B4logicDffCControls_FunctionDiv2mathDlyOfftimingDlyOntimingF2BtypesF2ItypesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncIpBoolCControls_BASR8M_Services	DailyScheduleFloat	basicSchedule
Demuxl2B4logicDffCControls_FunctionDiv2mathDlyOfftimingDlyOntimingF2BtypesF2ItypesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncIzFtypesInpBoolCControls_BASR8M_Services	DateTimeService	datetime
DffCControls_FunctionDiv2mathDlyOfftimingDlyOntimingF2BtypesF21typesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncIpBoolCControls_BASR8M_Services	Demuxl2B4	logic
Div2mathDlyOfftimingDlyOntimingF2BtypesF2ItypesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncIpBoolCControls_BASR8M_Services	Dff	CControls Function
DlyOfftimingDlyOntimingF2BtypesF21typesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncI2FtypesInpBoolCControls_BASR8M_Services	Div2	math
DiyOntimingF2BtypesF2ItypesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncI2FtypesInpBoolCControls_BASR8M_Services	DIvOff	timing
F2BtypesF2ItypesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncI2FtypesInpBoolCControls_BASR8M_Services	DivOn	timing
F2ItypesFloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncI2FtypesInpBoolCControls_BASR8M_Services	F2B	types
FloatOffsetmathFreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncI2FtypesInpBoolCControls_BASR8M_Services	F2	types
FreqfuncFtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncI2FtypesInpBoolCControls_BASR8M_Services	FloatOffset	math
FtoCCControls_FunctionHLpreCControls_FunctionHysteresisfuncI2FtypesInpBoolCControls_BASR8M_Services	Freq	func
HLpre     CControls_Function       Hysteresis     func       I2F     types       InpBool     CControls_BASR8M_Services	FtoC	CControls Function
Hysteresis     func       I2F     types       InpBool     CControls_BASR8M_Services	HLpre	CControls Function
I2F types InpBool CControls_BASR8M_Services	Hysteresis	func
InpBool CControls_BASR8M_Services	12F	types
	InpBool	CControls BASR8M Services
InpFloat CControls BASR8M Services	InpFloat	CControls BASR8M Services



## **Component–Kit Association**

Comp	onent
------	-------

Sedona Palette Folder

ISW	logic
IRamp	func
L2F	types
Limiter	func
Linearize	func
LP	func
LSeq	hvac
Max	math
Min	math
MinMax	math
Mul2	math
Mul4	math
Neg	math
Not	logic
OneShot	timing
Or2	logic
Or4	logic
OutBool	CControls BASR8M Services
OutFloat	CControls BASR8M Services
OutFloatCond	CControls BASR8M Services
PrioritizedBool	pricomp
PrioritizedFloat	pricomp
PrioritizedInt	pricomp
PsychrE	CControls Function
PsychrS	CControls_Function
Ramp	func
ReheatSeq	hvac
Reset	hvac
Round	math
ScanTim	CControls_BASC20_IO, CControls_BASC22_IO
SCLatch	CControls_Function
SendEmail	CControls_BASR8M_Services
SRLatch	func
Sub2	math
Sub4	math
TickTock	func
TimeAvg	math
Timer	timing
Tstat	hvac
UC1–UC4	CControls BASC20 IO, CControls BASC22 IO
UI1–UI8	CControls BASC20 IO, CControls BASC22 IO
UpDn	func
VT0–VT24	CControls BASC20 IO. CControls BASC22 IO
WC01–WC48	CControls BASC20 Web, CControls BASC22 Web
WriteBool	types
WriteFloat	types
WriteInt	types
Xor	logic
	5



United States Contemporary Control Systems, Inc. 2431 Curtiss Street Downers Grove, IL 60515 USA	China Contemporary Controls (Suzhou) Co. Ltd 11 Huoju Road Science & Technology Industrial Park New District, Suzhou PR China 215009	United Kingdom Contemporary Controls Ltd 14 Bow Court Fletchworth Gate Coventry CV5 6SP United Kingdom	Germany Contemporary Controls GmbH Fuggerstraße 1 B 04158 Leipzig Germany
Tel: +1 630 963 7070	Tel: +86 512 68095866	Tel: +44 (0)24 7641 3786	Tel: +49 341 520359 0
Fax:+1 630 963 0109	Fax: +86 512 68093760	Fax:+44 (0)24 7641 3923	Fax: +49 341 520359 16
info@ccontrols.com	info@ccontrols.com.cn	ccl.info@ccontrols.com	ccg.info@ccontrols.com
www.ccontrols.com	www.ccontrols.asia	www.ccontrols.eu	www.ccontrols.eu

