инженерный центр
УРАЛТЕХЭНЕРГО

InfoTask

Программный комплекс
для реализации информационно-вычислительных
и аналитических задач

Язык автоматизации расчетов Tablik

InfoTask-Tablik

Екатеринбург
2025

Содержание

1. Введение 6

2. Общие сведения 6
2.1. Принципы функционирования 6
2.2. Соединения 6
2.3. Расчеты и ведомости 7

3. Описание языка 8
3.1. Состав языка и структура выражений 8
3.2. Условные обозначения 9
3.3. Типы данных 9
Список типов данных языка 9
Диаграмма сводимости типов данных 11
3.4. Структура расчетного значения 11
Мгновенные значения 11
Виды расчетных значений 12
Приписанные параметры 12
Период расчета 13
Недостоверность и ошибки 14
3.5. Пример простого расчета 14
3.6. Переменные. Управляющее выражение 16
Переменные 16
Управляющее выражение 17
Область видимости переменных 17
3.7. Использование значений сигналов 18
3.8. Накопление значений параметров 18
3.9. Порядок расчета параметров 21
Параметры, подпараметры, входы 21
Использование сигналов в качестве входов. 22
Порядок расчета параметров. Использование параметров в формулах.
Формирование архивных параметров 23
Пример использования параметров 24
3.10. Передача в приемники 25
3.11. Графики 25
3.12. Параметры ручного ввода 26

4. Встроенные функции 28
4.1. Общее описание 28
Скалярные функции 28
4.2. Операции 31
Общие сведения 31
Унарные операции 31
Бинарные операции 31
Арифметические операции 31
Логические операции 33
Операции сравнения 34
4.3. Логические функции 36
Функция Sl 36
Функция Sr 36
Функция Бит 36
Функция БитИ 37
Функция БитИли 37

Константа Ложь 37
Константа Правда 37
4.4. Математические функции 38
Функция Arcch 38
Функция Arccos 38
Функция Arcsh 38
Функция Arcsin 38
Функция Arctan 38
Функция Arcth 39
Функция Ch 39
Функция Cos 39
Функция Exp 39
Функция Ln 40
Функция Log 40
Функция Log10 40
Функция Max 40
Функция Min 41
Константа Pi 41
Функция Sh 41
Функция Sin 41
Функция Sqr 41
Функция Tan 42
Функция Th 42
Функция Знак 42
Функция Модуль 42
Функция Округлить 42
Функция Случайное 43
4.5. Функции работы с недостоверностью и ошибками 43
Функция Недост 43
Функция Ошибка 43
Функция ЗаписатьНедост 44
Функция ЗаписатьОшибку 44
Функция Достовер 45
Функция ДостоверН 45
Функция ДостоверНП 46
Функция ДостоверП 47
4.6. Функции работы с мгновенными значениями 48
Функция Апертура 48
Константа БезТочек 49
Функция ВремяМеждуТочками 49
Функция ВремяТочки 49
Функция ВыбратьТочки 50
Функция ЗаписатьВремя 51
Функция ЗначениеПоВремени 51
Функция ЗначениеПоТочкам 51
Функция Зона 52
Функция ЗонаСКонца 53
Функция МаксЗона 55
Функция НарастающийИтог 55
Функция НомерТочки 56
Функция Объединить 56
Функция РавномерныеТочки 57
Функция СдвигПоВремени 58

Функция СдвигПоНомеру 59
Функция Скорость 59
Функция Событие 60
Функция Точка 61
Функция УдалитьПовторы 62
4.7. Статистические функции 63
Общее описание 63
Функция Время 63
Функция Дисперсия 64
Функция ЗначениеКонец 65
Функция ЗначениеНачало 66
Функция ЗначениеПоНомеру 67
Функция ЗначениеПоНомеруСКонца 68
Функция Интеграл 69
Функция Количество 70
Функция КоличествоТочек 71
Функция Максимум 72
Функция Минимум 73
Функция Среднее 74
Функция СреднееВЗоне 75
Функция Сумма 76
4.8. Функции работы с сегментами 77
Функция РавномерныеСегменты 78
Функция Сегменты 79
Функция СегментыОтТочек 79
Функция КоличествоСегментов 80
Функция ОбъединитьСегменты 80
Функция СегментПоНомеру 81
4.9. Функции работы с таблицами 81
Функция ВзятьТабл 82
Функция ВзятьТаблИмя 82
Функция ТаблСодержит 82
Функция ВзятьПодТабл 83
Функция ВзятьПодТаблИмя 83
Функция ПодТаблСодержит 84
4.10. Функции типов данных 84
Функция Величина 85
Функция Дата 85
Функция Действ 85
Функция Логич 86
Функция Строка 86
Функция Целое 86
Функция ЯвляетсяВременем 87
Функция ЯвляетсяЦелым 87
Функция ЯвляетсяЧислом 87
4.11. Функции работы со временем 87
Функция НачалоПериода 87
Функция КонецПериода 88
Функция Сейчас 88
Функция ДобавитьКоВремени 88
Функция РазностьВремен 88
Функция СобратьВремя 89
Функция ЧастьВремени 89

Константы интервалов времени 89
4.12. Строковые функции 90
Строковые константы 90
Функция StrFind 90
Функция StrFindLast 90
Функция StrInsert 90
Функция StrLCase 91
Функция StrLeft 91
Функция StrLen 91
Функция StrLTrim 91
Функция StrMid 91
Функция StrRemove 92
Функция StrReplace 92
Функция StrReplaceReg 92
Функция StrRight 93
Функция StrRTrim 93
Функция StrTrim 93
Функция StrUCase 93
4.13. Ключевые слова 93
Константа Пустой 93
Переменная Расчет 94
Переменная Ручной 94
4.14. Функции работы с сигналами и параметрами 94
Функция Владелец 94
Функция Вызывающий 94
Функция ПолучитьЕдиницы 94
Функция ПолучитьЗадачу 95
Функция ПолучитьИмя 95
Функция ПолучитьИмяСигнала 95
Функция ПолучитьКод 95
Функция ПолучитьКодСигнала 96
Функция ПолучитьКомментарий 96
Функция Сигнал 96
Функция СигналДейств 97
Функция СигналЛогич 98
Функция СигналСтрока 98
Функция СигналЦелое 98
4.15. Условия и циклы 99
Функция Если 99
Функция ЕслиТочки 99
Пример вычисления условий 100
Функция Пока 101
Функция ПокаТочки 102
Пример вычисления циклов 102
5. Список принятых сокращений 104

1. ВВЕДЕНИЕ

Язык автоматизации расчетов Tablik представляет собой специализированный технологически ориентированный язык для описания расчетных выражений, использующихся при выполнении расчетов программным комплексом для реализации информационно-вычислительных и аналитических задач InfoTask.

В настоящем документе описывается структура расчетных выражений и возможности языка Tablik по использованию операторов, математических, статистических и технологических функций.

Функционирование InfoTask в целом и составляющих его компонентов описано в других документах, входящих в состав документации InfoTask.

Полный состав документации InfoTask приведен в документе InfoTask-Base «Общее описание».

2. ОБЩИЕ СВЕДЕНИЯ

2.1. Принципы функционирования

Входными данными для InfoTask являются список сигналов и значения этих сигналов программно-технического комплекса (далее – ПТК) автоматизированной системы управления технологическим процессом (далее – АСУТП).

Список сигналов необходим при проектировании расчетных формул, формировании наборов сигналов, а также при организации запросов для получения значений этих сигналов из архива. Список сигналов в зависимости от типа ПТК может быть получен из архива мгновенных значений ПТК, из клона архива или из базы данных ПТК.

Значения сигналов считываются из архива мгновенных значений ПТК или клона архива для использования в расчетах. Далее выполняются запроектированные пользователем расчеты и комплекс функций, связанных с ретроспективным анализом информации. Затем результаты отображаются и при необходимости распечатываются.

В состав компонентов InfoTask входят программные модули и приложения, имеющие графический интерфейс и предоставляющие пользователю возможность управления функционалом комплекса.

2.2. Соединения

Обмен информацией InfoTask с ПТК АСУТП и клоном архива производится через соединения, реализуемые при помощи провайдеров. Провайдерами в терминологии InfoTask называются программные модули, обеспечивающие функционирование соединений. Каждый провайдер имеет набор настроек, описывающих подключение к источникам данных, а также дополнительные настройки, необходимые для его функционирования.

Провайдеры, с помощью которых создаются соединения для обмена информацией с ПТК АСУТП, разрабатываются для определённого типа ПТК – того, на базе которого построена АСУТП. Для работы с каждым конкретным типом ПТК вместе с InfoTask поставляется специальный провайдер (комплект провайдеров), ориентированный на данный тип ПТК.

Соединения для обмена информацией бывают следующих типов:

- Источник – осуществляет подключение к источнику данных (архив мгновенных значений ПТК АСУТП или файл специального формата – клон архива) и считывание из него исходной информации (мгновенные значения сигналов технологического процесса);

- Коммуникатор – осуществляет подключение к базе данных ПТК, либо к архиву мгновенных значений ПТК, либо к клону архива, чтобы произвести считывание списка сигналов и характеристик сигналов. Характеристиками сигналов (точек технологического процесса) являются, например: идентификаторы, обозначения, наименования, аварийные и предупредительные уставки, единицы измерения, шкалы и т. п.

В зависимости от типа ПТК и типа соединения применяются разные провайдеры. Как правило, провайдеры используют штатные каналы доступа к информации ПТК, программное обеспечение для реализации которых обычно поставляется комплектно с ПТК. Работа с конкретным провайдером (комплектом провайдеров) описана в соответствующем документе на провайдер.

Соединения для подключения к клону архива называются встроенными соединениями, а провайдеры, с помощью которых они реализуются, называются Встроенными провайдерами. Встроенные провайдеры являются неотъемлемой частью InfoTask.

Управление соединениями, установка и конфигурирование связей для обмена информацией как с внешними системами – ПТК АСУТП, так и между компонентами комплекса, производится в приложении «Менеджер соединений».

2.3. Расчеты и ведомости

В отдельном файле формата базы данных SQLite находится проект InfoTask с расчетами. В основе проекта описания расчетных параметров, значения которых являются результатами расчета. В описаниях для каждого параметра задается формула его вычисления (формирования). Описание формул расчетных параметров вводится при помощи операторов специального языка автоматизации расчетов Tablik.

Расчеты выполняются в программном модуле Контроллер расчетов. Контроллер расчетов вызывается в приложении Анализатор архивных данных, в котором описывается (проектируется) расчет. По результатам расчета формируется файл ведомости, содержащий данные для отображения результатов расчета.

3. ОПИСАНИЕ ЯЗЫКА

Каждый расчетный параметр InfoTask имеет характеристики, которые задаются при создании параметра. Основной характеристикой является «Расчетная формула», которая содержит расчетную формулу (выражение), записанную на языке Tablik. В данном разделе описываются основные элементы выражений и синтаксис использования этих элементов.

3.1. Состав языка и структура выражений

В выражении можно использовать:

- числовые константы;
- строковые константы;
- константы даты;
- полные коды сигналов и объектов из базы данных ПТК;
- унарные и бинарные операции;
- встроенные функции;
- присваивания;
- функции условий и циклов;
- операторы работы с типами и значениями;
- входы параметра и его владельца;
- внутренние переменные параметра и его владельца;
- другие расчетные параметры, в том числе с заданием входов;
- подпараметры ранее указанных параметров, в том числе с заданием входов;
- функции получения предыдущих значений архивных параметров;
- функции получения сигналов и характеристик сигналов и параметров;
- обозначения графиков;
- круглые и квадратные скобки, разделители, пробелы;
- комментарии.

Элементы выражения отделяются друг от друга пробелами или переводами строки. Пробелами можно не отделять операции, состоящие из символов «+-*/^><=», сигналы ПТК, заключенные в фигурные скобки «{}» и разделители «():;».

В любом месте выражения допускается использование комментария. Комментарий начинается символами «*», заканчивается символами «*\», может содержать любой текст. При компиляции и расчете комментарии пропускаются.

Для обозначения расчетных параметров, переменных, встроенных функций, типов данных и графиков используются идентификаторы. Длина идентификатора не должна превышать 50 (пятьдесят) символов. Идентификатор может содержать заглавные и строчные русские и английские буквы, цифры и символы подчеркивания, причем он должен содержать хотя бы один символ отличный от цифры. Идентификатор не может содержать пробелов, скобок, точек и т. п. Коды объектов и сигналов идентификаторами не являются и могут содержать любые символы, кроме фигурных скобок.

Язык Tablik не чувствителен к регистру символов. При компиляции заглавные и соответствующие строчные буквы, содержащиеся в идентификаторах, или записи сигналов, считаются одинаковыми. Например, переменная с кодом «*var»* может также быть

записана как «Var» и как «VAR», все три варианта будут считаться кодом одной и той же переменной.

3.2. Условные обозначения

При описании синтаксиса какой-либо конструкции языка в настоящем документе используются следующие правила обозначения:

- если некоторое название написано в угловых скобках, то в формуле вместо него должно использоваться некоторое выражение, определение которого очевидно или описано в данном документе;
- если название в угловых скобках написано в начале строки, и после него идут символы «:=», то вся оставшаяся строка является описанием выражения с таким названием;
- если некоторая последовательность символов написана без угловых скобок, то она должна обязательно быть использована в выражении на соответствующем месте;
- если некоторая часть описания заключена в квадратные скобки, то ее использование в выражении допустимо, но не обязательно;
- если некоторые части описания перечислены через вертикальную черту, то из них нужно выбрать какое-нибудь одно.

3.3. Типы данных

В языке выражений пять основных встроенных типов данных: логический, целый, действительный, строковый и время. Есть также дополнительный встроенный тип данных – сегменты, который не имеет своих констант, не сводится к другим типам, но значение такого типа может быть результатом функции и использоваться в функциях в качестве параметров. Кроме того в языке есть тип, означающий отсутствие значения, а также и другие вспомогательные типы данных.

Каждый тип имеет обозначение и его английский синоним, полностью взаимозаменяемый с именем.

Типы могут приводиться друг к другу. Например, логический тип приводится к целому, и это значит, что значение логического типа можно использовать везде, где требуется значение целого типа (например, значение формулы «3+True» определено и равно 4). Для большинства встроенных типов в языке могут использоваться константы. Недостоверность любой константы равна 0, ошибки у любой константы нет. При накоплении результатов константе приписывается время значения, равное началу периода расчета.

Список типов данных языка

Далее приведено подробное описание всех типов данных, встроенных в язык Tablik.

- **Логический тип**. Обозначение: **логич**. Английское обозначение: **bool**.

Логический тип имеет два значения **0** и **1**. Вместо них можно использовать константы **False** и **True**. Логический тип сводится к целому, действительному (число остается без изменения) и строковому (число переводится в его строчную запись).

- **Целый тип**. Обозначение: **целое**. Английское обозначение: **int**.

Значения целого типа – это 4-х байтные целые числа от -2^{31} до 2^{31}. Целый тип сводится к действительному (число остается без изменения) и строковому (число переводится в его строчную запись).

- **Действительный тип**. Обозначение: **действ**. Английское обозначение: **real**.

Значения действительного типа – это вещественные числа с плавающей точкой размером в 8 байт. Константы действительного типа могут иметь один из следующих видов:

<**Целая часть**> например, 356;

<**Целая часть**>.<**Дробная часть**> например, 234.908 или -64.45;

<**Целая часть**>,<**Дробная часть**> например, 93,45 или -23,567;

<**Целая часть**>.<**Дробная часть**> **e** <**Экспонента**> например, -2.238e23 или 3.3e-34;

<**Целая часть**>,<**Дробная часть**> **e** <**Экспонента**> например, -2,238e23 или 3,3e-34.

Точка «**.**» и запятая «**,**» полностью эквивалентны. Длина целой и дробной частей, вместе взятых, не должна превышать 14 (четырнадцать) символов. Целая часть и экспонента могут быть отрицательными. Действительный тип сводится к строковому, число переводится в его строчную запись.

- **Строковый тип**. Обозначение: **строка**. Английское обозначение: **string**.

Значения строкового типа – это строки, содержащие произвольную последовательность символов. Константы строкового типа имеют следующий вид:

‘<**Строка**>’, например ‘ффф1’.

Строковый тип не приводится к другим типам данных.

- Временной тип. Обозначение: **время**. Английское обозначение: **time**.

Значение временного типа – это момент времени с точностью выше миллисекунды, с указанием полной даты. Константы временного типа имеют следующий вид:

 # <Год>.<Месяц>.<День> <Час>:<Минута>:<Секунда>#

Временной тип сводится к строковому, время переводится в строчную запись в формате <Год>.<Месяц>.<День> <Час>:<Минута>:<Секунда>.

- **Тип сегменты**. Обозначение: **сегменты**. Английское обозначение: **segments**.

Значения типа сегменты формируются при помощи функций работы с сегментами и используются статистическими функциями (см. разделы 4.7 и 4.8). Тип сегменты не имеет констант и не сводится к другим типам данных.

- **Отсутствие значения**. Обозначение: **пустой**. Английское обозначение: **void**.

Значения типа «пустой» возвращают операция присваивания, циклы и функция «Пустой». Тип не сводится к другим типам данных.

- **Ошибка**. Обозначение: **ошибка**. Английское обозначение: **error**.

Элемент расчета при компиляции получает тип «ошибка», если его тип данных не удалось определить корректно. Для удобства считается, что тип «ошибка» может сводиться к любому типу данных.

- **Величина без значения**. Вспомогательный тип данных. Обозначение: **величина**. Английское обозначение: **value**.

Задает величину без значения, но с заданным временем, недостоверностью и ошибкой. Возникает как тип неопределенных переменных и т. п. Сводится к любому из типов данных, кроме пустого и сегментов.

- **Величина любого типа**. Вспомогательный тип данных. Обозначение: **вариант**. Английское обозначение: **variant**.

В выражениях не используется. Участвует в описаниях возможных типов аргументов встроенных функций и означает, что аргумент может иметь любой тип, кроме пустого и сегментов.

Диаграмма сводимости типов данных

Диаграмма сводимости типов языка Tablik приведена на рисунке .3.1. Если на диаграмме изображена стрелка от типа T1 к типу T2, то тип T1 сводится к типу T2.

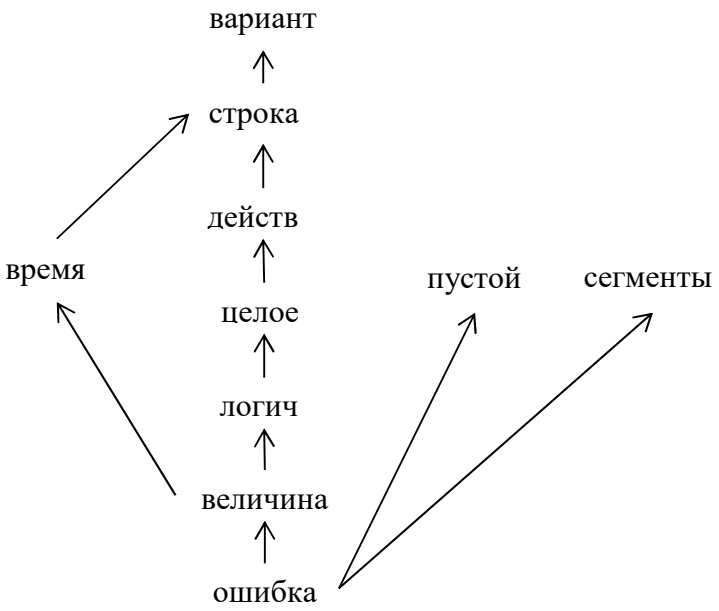


Рис. 3.1. Сводимость типов данных

3.4. Структура расчетного значения

Элементами расчета называются сигналы, параметры, графики, переменные, а также любые элементы выражений, имеющие выходное значение, такие как функции, операции, константы и т. д.

Описание встроенных функций, включая операции, выделено в раздел 4, остальные элементы языка описаны в разделе 3.

Каждый элемент расчета имеет определенный тип данных. Значение расчетного элемента называется **расчетным значением** или просто **значением**.

Мгновенные значения

Расчетные значения чаще всего (но не всегда) состоят из одного или нескольких **мгновенных значений**, которые также часто могут называться **точками**. Одно мгновенное значение содержит в себе следующую информацию:

- **Тип данных**: логич, целое, действ, время, строка или величина. Описание типов данных в разделе 3.3;
- **Величина** – величина мгновенного значения, которая имеет разный вид в зависимости от типа данных: логическое значение 0 или 1, целое число, вещественное число, время или строка;
- **Время значения**. Каждому мгновенному значению приписывается некоторый момент времени, который чаще всего (но не обязательно) лежит в пределах периода

расчета. Константным значениям при расчете автоматически присваивается время равное началу периода расчета;

- **Недостоверность**. Каждому мгновенному значению приписывается некоторое целочисленное значение недостоверности. Недостоверность достоверного значения равна 0;
- **Ошибка**. Если в процессе вычисления мгновенного значения произошла ошибка, или мгновенное значение было получено с использованием ошибочного значения, то мгновенному значению приписывается ошибка. Ошибка представляет собой строку с описанием. Ошибка безошибочного мгновенного значения равна пустой строке.

Виды расчетных значений

Одно расчетное значение в системе InfoTask, в зависимости от типа данных и количества мгновенных значений, может представлять собой структуру одного из следующих видов:

- **Одиночное значение**. Представляет собой одно мгновенное значение определенного типа данных. Обычно получается как значение константы, как результат статистической функции, или как результат применения функций к другим одиночным значениям;
- **Список мгновенных значений**. Представляет собой список различающихся временем мгновенных значений одинакового типа данных. Недостоверности и ошибки разных значений из списка могут различаться. Мгновенные значения в списке упорядочены по времени. Времена значений из списка могут не попадать в период расчета. Список мгновенных значений может быть пустым;
- **Список сегментов**. Представляет собой список сегментов – особую структуру данных для расчета статистических функций. Такое расчетное значение всегда имеет тип данных **сегменты**. Списки сегментов формируются при помощи функций работы с сегментами (см. раздел 4.8) и используются статистическими функциями (см. раздел 4.7). Список сегментов не содержит мгновенных значений;
- **Пустое значение**. Значение типа данных **пустой** означает отсутствие значения. Возникает как результат операции присвоения, функции цикла или функции Пустой.

Приписанные параметры

Кроме основной структуры расчетного значения, описанной в предыдущем подразделе, расчетное значение ссылается на расчетный параметр, при расчете которого оно получено. Каждому расчетному значению может быть приписан экземпляр расчетного параметра. Если расчетному значению приписан параметр «P1», то от этого значения через точку можно получать значения подпараметров параметра «P1» и предыдущие архивные значения, формируемые параметром «P1». Также если расчетному значению параметра «P2» приписан параметр «P1», то в архив будут накапливаться не только значения подпараметров. Подробнее о вызове расчетных параметров и о накоплении в архив см. разделы 4.1–4.4.

Расчетный параметр приписывается расчетному значению следующим образом:

- если в расчетном выражении используется параметр или подпараметр с входами или без, то он приписывается соответствующему расчетному значению;
- константы, сигналы и значения графиков не имеют приписанных параметров;

- результаты операций и встроенных функций, кроме условий и ключевых слов, не имеют приписанных параметров, даже если входные аргументы этих операций или функций приписанные параметры имели;
- расчетное значение присваивается переменной всегда вместе с приписанным параметром;
- если при вызове расчетного параметра-функции «Р1» с входами в выражении другого параметра, некоторый входной аргумент «Х» параметра «Р1» имел приписанный параметр «Р2», то и при вычислении значения параметра «Р1» любое расчетное значение входа «Х» будет иметь приписанный параметр «Р2»;
- пусть расчетному значению приписан параметр «Р1». Расчетному значению параметра «Р1» в свою очередь может быть приписан параметр «Р2» и т. д. Таким образом, каждому расчетному значению фактически может быть приписана целая цепочка расчетных параметров.

Подробнее о вызове расчетных параметров, подпараметров и о накоплении в архив в разделах 3.8–3.11.

Период расчета

Вычисление значений расчетных параметров каждый раз производится за определенный промежуток времени, называемый **периодом расчета**. Период расчета задается временем начала и временем конца. Для расчета значений параметров используются данные из источников за период расчета (это обычно срез значений на начало периода и изменения значений за период по сигналам ПТК).

Перед вычислением значений расчетных параметров для каждого используемого в выражениях сигнала ПТК из провайдера-источника считываются мгновенные значения за период расчета: список моментов времени со значениями сигналов в эти моменты. Значение сигнала ПТК (также как и расчетного параметра) на промежутке времени между двумя последовательными мгновенными значениями принимается равным мгновенному значению в начале промежутка. Значение сигнала (также как и расчетного параметра) в конце периода обработки принимается равным последнему мгновенному значению.

Если в выражении расчетного параметра используются сигналы ПТК или другие расчетные параметры, то мгновенное значение расчетного параметра вычисляется в каждый момент, соответствующий мгновенному значению одного из сигналов ПТК или одного из расчетных параметров, используемых в выражении этого параметра. Мгновенные значения могут быть накоплены в архиве значений расчетных параметров, если для параметра указан тип накопления «Мгновенные».

Если в выражении используется статистическая функция, то в результате чаще всего получится одно итоговое значение. Оно может являться итоговым значением расчетного параметра. Если значение расчетного параметра представляет собой список мгновенных значений, то итоговое значение расчетного параметра вычисляется как его мгновенное значение в конце периода обработки. Итоговые значения расчетных параметров могут быть записаны в архив результатов. При периодическом расчете по итоговым значениям базовых интервалов могут формироваться часовые, суточные и абсолютные значения для записи в архив.

Подробнее о том, как преобразуются расчетные значения при применении встроенных в язык операций и функций в разделе 4 – общие сведения, другие разделы главы 4 – применение конкретных функций.

Недостоверность и ошибки

Для каждого мгновенного значения сигнала ПТК из архива также считывается его **недостоверность**. Мгновенные и итоговые недостоверности вычисляемых выражений рассчитываются исходя из мгновенных недостоверностей сигналов ПТК.

Недостоверность достоверного сигнала ПТК равна 0, недостоверного – 1 (не важно по какой причине возникла недостоверность). Если в выражение входит сигнал ПТК, мгновенное значение которого недостоверно, и мгновенное значение выражения зависит от этого сигнала ПТК, то соответствующее мгновенное значение выражения тоже будет недостоверным. Если аргумент статистической функции расчетного параметра хотя бы в одной точке периода расчета был недостоверен, то итоговая недостоверность этого параметра равна 1, иначе равна 0.

Мгновенная недостоверность может быть использована при расчете (через функцию «Недост»). Мгновенное значение недостоверности может быть искусственно присвоено выражению, исходя из его значения (через функцию «ЗаписатьНедост»).

Мгновенные и итоговые значения выражений могут содержать ошибку. Например, результат выражения 1/0 будет содержать ошибку. Однако при возникновении такой ситуации расчет не остановится, а будет продолжаться, и в результате ошибки может и не быть. Например, выражение Если(1;2;1/0) всегда возвращает 2 без ошибки, хотя внутри себя содержит ошибочную часть. Если ошибка так и не исправилась к концу расчета, то при расчете выдается соответствующее предупреждение, но расчет не останавливается, а ошибка заносится в поле «Ошибка» списка расчетных параметров.

При операциях с мгновенными значениями чаще всего (исключения всегда указаны особо) результат, полученный по некоторым мгновенным значениям, имеет недостоверность равную максимуму недостоверностей этих значений. Аналогичным образом результат, полученный по некоторым мгновенным значениям, имеет ошибку, если хотя бы одно из этих значений имеет ошибку. При этом из исходных ошибок выбирается первая попавшаяся. В этом случае в данном документе часто формулируется, что ошибка результата равна максимуму из ошибок исходных значений.

Подробнее о присвоении расчетным значениям недостоверностей и ошибок при вычислении значений стандартных функций в разделе 4.1.

3.5. Пример простого расчета

Этот раздел содержит простой пример проекта с формулами, а также детальное описание расчета в соответствии с этим примером. Используемые языковые конструкции и функции будут подробно описаны в других разделах.

Допустим, наш проект содержит два объекта с кодами «Ob1» и «Ob2». Пусть объект Ob1 содержит единственный целочисленный сигнал Sig (тип «целое») заданный по умолчанию, а объект Ob2 содержит два аналоговых сигнала SigA и SigB (тип «действ», числа с плавающей точкой), причем SigA является сигналом по умолчанию. Пусть проект содержит 6 (шесть) расчетных параметров Par1, …, Par6 со следующими расчетными выражениями:

Параметр	Формула	Тип данных
Par1	Максимум({Ob1})	целое
Par2	Среднее({Ob2})+Минимум({Ob2.SigB})	действ
Par3	Par1*Par2	действ
Par4	{Ob1.Sig}+{Ob2.SigA}+{Ob2.SigB}	действ
Par5	A=Par4:B=Par1:A+B	действ
Par6	Если({Ob1}>2;{Ob1};2)	целое

Пусть управляющие выражения во всех параметрах проекта не используют никаких вычислений и заполнены как «Расчет».

Сигналы в формулах используются в фигурных скобках. Если сигнал является сигналом по умолчанию для объекта, то в фигурных скобках достаточно указать код объекта, иначе нужно указывать код объекта и через точку код сигнала. Таким образом, в формуле параметров Par1 и Par6 реально используется сигнал Ob1.Sig, в формуле параметра Par2 используются сигналы Ob2.SigA и Ob2.SigB, а в формуле параметра Par4 используются все три сигнала.

Для того чтобы использовать значение одного параметра при расчете другого в формуле достаточно написать его код. Таким образом, в расчете параметра Par3 используются параметры Par1 и Par2, в расчете параметра Par5 – параметры Par1 и Par4. Если один параметр используется при расчете другого, то его значение будет вычислено раньше.

Объект Ob1 имеет тип целое. Функция «Максимум» вычисляет максимальное значение своего аргумента за период расчета и возвращает одно значение, которому приписывается время, в которое аргумент достиг своего минимума. Таким образом, значение параметра Par1 равно максимуму сигнала Ob1.Sig за период расчета и является целым.

Функция «Среднее» вычисляет среднее интегральное за период расчета. Функция «Максимум» возвращает максимум за период расчета. Сигналы Ob2.SigA и Ob2.SigB имеют тип «дейст» (числа с плавающей точкой), поэтому и значение параметра Par2 будет иметь тип «дейст». Параметр Par3 будет иметь тип дейст как произведение целого числа и числа с плавающей точкой.

Значения параметров Par4, Par5 и Par6 будут представлять собой списки мгновенных значений, изменяющихся с течением времени. Значения параметров Par4 и Par5 имеют тип «дейст» как суммы значений типа «дейст». В формуле параметра Par5 используются две внутренних переменных A и B. Вначале переменным присваиваются значения, затем, в качестве значения расчетного параметра выводится сумма A+B.

Функция «Если» имеет 3 аргумента: первый – логическое условие; второй – значение, возвращаемое, когда условие выполняется; третий – значение, возвращаемое когда условие не выполняется. Оба возвращаемых значения имеют тип «целое», поэтому значение сигнала Ob1 имеет тип «целое». Условие может выполняться на одних временных промежутках и не выполняться на других, поэтому среди мгновенных значений параметра могут быть как значения второго аргумента функции «Если», так и третьего.

Допустим, производится расчет за период от 00:00:00 до 01:00:00, и сигналы имеют следующие мгновенные значения. Если в клетке таблицы не указано значение, то значение сигнала в этот момент равно значению, ближайшему вверх по столбцу, если выше по столбцу нет значений, то значение равно ближайшему вниз по столбцу значению.

Время	Ob1=Ob1.Sig	Ob2=Ob2.SigA	Ob2.SigB
00:00:00	3,0	0,5	3,5
00:10:00	1,0		
00:20:00		2,0	
00:35:00			2,5
00:40:00	4,0	3,5	
00:50:00	2,0		

При таких исходных данных расчетные параметры будут иметь следующие мгновенные значения:

Время	Par1	Par2	Par3	Par4	Par5	Par6
00:00:00		4,5	18,0	7,0	10,0	3,0
00:10:00				5,0	8,0	2,0
00:20:00				6,5	9,5	

Время	Par1	Par2	Par3	Par4	Par5	Par6
00:35:00				5,5	8,5	
00:40:00	4,0			10,0	13,0	4,0
00:50:00				8,0	11,0	2,0

Значением параметра Par1 является 3, то есть максимум объекта Ob1. Значению приписано время минимального значения Ob1, то есть 00:40:00.

Значение параметра Par4 получается как сумма трех сигналов и имеет мгновенные значения во всех точках, в которых мгновенное значение имеет хотя бы один из сигналов. Значение параметра Par5 получается как сумма параметра Par4 имеющего много значений, и параметра Par1, имеющего одно значение, причем не в начале периода. Однако, в подобном случае считается, что параметр Par1 имел значение 4 весь период расчета, а не только начиная с 00:40:00.

Расчет значений параметров Par2 и Par3 производится следующим образом. Среднее интегральное значение от сигнала Ob2 за период расчета «=»

= Интеграл({Ob2})/Длина периода = (0,5*20+2*20+3,5*20)/60=2.

Среднему значению всегда приписывается время, равное началу периода. Минимум сигнала {Ob2.SigB} достигается в 00:35:00 и равен 2,5. После сложения результат имеет два мгновенных значения, но оба значения равны, поэтому второе из них откидывается. Точно так же значение параметра Par3 состоит из трех одинаковых мгновенных значений, но два из них откидываются.

Результатом вычисления параметра Par6 является результат вычисления функции «Если». Условие {Ob1}>2 выполняется на промежутках от 00:00:00 до 00:10:00 и от 00:40:00 до 00:50:00. На этих промежутках времени значение второго аргумента – сигнала Ob1 равно 3 и 4 соответственно. Втечение всего оставшегося времени значение сигнала Par6 равно 2.

3.6. Переменные. Управляющее выражение

Переменные

В выражениях можно использовать внутренние **переменные**. Каждая переменная имеет свой код (имя). Код переменной является идентификатором языка Tablik, то есть может содержать только русские и английские буквы, цифры и символы подчеркивания, но при этом должен содержать хотя бы один символ отличный от цифры.

Переменной сначала присваивается значение (выражение), а затем эта переменная используется в вычисляемом выражении. Переменную не нужно специально объявлять, она считается объявленной после того, как ей в первый раз было присвоено значение. Присвоения значений переменным отделяются друг от друга и от итоговых значений двоеточием «**:**».

В целом выражение расчетного параметра выглядит следующим образом:

<**Выражение расчетного параметра**>:=[<**Описание переменных**>**:**]<**Выражение**>

<**Описание переменных**>

Присвоение значений переменным;

<**Код переменной**>=<**Выражение переменной**> [**:** <**Описание переменных**>]

Присвоение значения одной переменной выглядит так:

<**Код переменной**>=<**Выражение переменной**>.

Для того чтобы потом использовать значение переменной в выражении, достаточно указать ее код. В выражении переменная должна быть присвоена, перед тем как будет использована. Значение может быть присвоено переменной неоднократно.

Например, выражение может выглядеть так: A=2: B=A+1: B

В этом случае параметр содержит две внутренние переменные A и B. После вычисления выражения переменной A будет присвоено значение 2, переменной B будет присвоено значение 3, расчетное же значение всего параметра будет равно значению переменной B, то есть 3.

При присвоении значения переменной в нее обычно полностью копируется расчетное значение, будь то одиночное значение, список мгновенных значений, список сегментов или пустое значение. Мгновенные значения копируются вместе с недостоверностью и ошибкой.

Переменные также можно присваивать внутри всех, кроме первого, аргументов условных функций («Если», «ЕслиТочки») и функций циклов («Пока», «ПокаТочки»). В этом случае значение переменной присваивается не полностью, а только на тех временных промежутках, на которых выполняется условие условной функции или условие текущей итерации цикла. На остальных временных промежутках значение переменной остается прежним. Подробнее о присвоениях внутри условных функций и функций циклов в разделе 4.18.

Управляющее выражение

Выражение расчетного параметра состоит из двух частей: **управляющего выражения** и **расчетного выражения**. В управляющем значении можно использовать итоговое значение расчетного параметра, для этого служит слово «**Расчет»**. Фактически «**Расчет**» является переменной, которую можно использовать в управляющем выражении. При вычислении значения расчетного параметра сначала вычисляется значение расчетного выражения, потом это значение подставляется в управляющее выражение. Расчетным значением расчетного параметра является расчетное значение, полученное в результате вычисления значения его управляющего выражения.

Например, если параметр задан следующими выражениями:

Расчетное выражение	Управляющее выражение
1+2	Расчет+3

Тогда его итоговое значение будет равно 6.

Управляющее выражение служит для удобства, в него хорошо выносить части формул, общие для большого количества параметров. По умолчанию управляющее выражение задается равным переменной «Расчет».

Область видимости переменных

В расчетном параметре (включая расчетное и управляющее выражение) может быть использована только одна переменная с конкретным кодом.

В область видимости переменной входит весь параметр. То есть, переменная, впервые присвоенная в расчетном выражении (даже внутри условной функции или функции цикла), может быть неоднократно использована как в расчетном, так и в управляющем выражении.

В расчетных и управляющих выражениях подпараметров кроме собственных внутренних переменных можно использовать переменные параметра-владельца. Расчет параметра-владельца всегда производится раньше, чем расчет подпараметра, поэтому при использовании в подпараметре переменная будет иметь значение, которое у нее было

на момент окончания вычисления параметра-владельца. Если в подпараметре и его параметре владельце присваивается значение переменным с одинаковым кодом, то эти переменные считаются разными переменными. Если значение присваивается во владельце, а в подпараметре только используется, то такие переменные считаются одной переменной.

Код переменной не должен совпадать с кодом встроенной функции или графика.

Если «Ind» – некоторый идентификатор, в проекте есть параметр с кодом равным этому идентификатору и в одном из параметров проекта (обозначим его «Par») есть переменная с кодом, также равным этому идентификатору, то любое использование в выражениях параметра Par идентификатора Ind будет означать переменную, а не параметр. Аналогично в пользу переменной разрешается конфликт, в случае совпадения кодов переменной и подпараметра или подпараметра владельца. Более полно о разрешении конфликтов имен идентификаторов в соответствующем разделе.

3.7. Использование значений сигналов

В выражении можно использовать мгновенные значения любого сигнала, получаемого из источника данных. Сигнал в выражении указывается следующим образом:

<**Сигнал из ПТК**>:= **{** <**Код Объекта**>[**.**<**Код Сигнала**>] **},** где

<**Код Объекта**> – код объекта из источника;

<**Код Сигнала**> – код сигнала внутри объекта, который можно опускать, если сигнал является сигналом по умолчанию.

Например:

{4CU10E013.1_Пар} – значение сигнала 1_Пар объекта с кодом 4CU10E013;

{4CU10E013} – значение сигнала по умолчанию объекта с кодом 4CU10E013.

Код объекта может содержать в себе любые символы кроме фигурных скобок, в том числе и точку. Код сигнала может содержать любые символы кроме фигурных скобок и точек. Указанный код сигнала должен совпадать либо с полным кодом одного из сигналов, либо с кодом одного из объектов проекта, иначе при компиляции будет выдана ошибка.

При расчете значение сигнала чаще всего представляет собой список мгновенных значений из архива. Однако значение сигнала может быть константой для сигнала-константы. Подробнее о характеристиках объектов и сигналов в разделе 2.3.

3.8. Накопление значений параметров

Итоговые значения расчетных параметров, формируемые исходя из выражений, помещаются в архив результатов. Архив результатов в числе прочих содержит следующие таблицы:

- Таблица параметров;
- Таблица базовых значений;
- Таблица часовых значений;
- Таблица суточных значений;
- Таблица абсолютных значений;
- Таблица мгновенных значений;
- Таблица значений разовых интервалов;
- Таблица значений именованных интервалов;
- Таблица соответствующих интервалов для каждой таблицы значений.

В один архив могут записываться результаты расчетов по нескольким проектам InfoTask, в том числе обсчитываемые в разных потоках. Таблица параметров архива содержит копии списков архивных параметров для каждого проекта. Таблица параметров архива содержит следующие поля:

Поле	Описание поля
Код	Полный код расчетного параметра вместе со всей цепочкой кодов подпараметров
Код параметра	Код расчетного параметра без кодов подпараметров
Код подпараметра	Код последнего подпараметра в цепочке полного кода
Имя	Имя расчетного параметра
Имя подпараметра	Имя подпараметра
Задача	Задача в проекте, которой принадлежит параметр
Тип данных	Возвращаемый параметром тип данных
Комментарий	Комментарий
Единицы измерения	Единицы измерения
Тип накопления	Тип накопления
Min	Минимум шкалы
Max	Максимум шкалы
Active	Логическое, false, если параметр уже не участвует в расчете
ParamId	Уникальный номер для связи с другими таблицами

Таблицы интервалов архива содержат следующие поля:

Поле	Описание поля
Начало	Время начала интервала
Конец	Время конца интервала
Имя	Имя интервала
IntervalId	Уникальный номер для связи с другими таблицами
TimeAdd	Время добавления интервала
TimeChange	Время последнего изменения интервала

Каждый параметр на каждом интервале может иметь список значений. Значения хранятся в таблицах значений. Каждому типу интервалов соответствует своя таблица значений. Каждая из таблиц значений содержит следующие поля:

Поле	Описание поля
Величина	Величина мгновенного значения, число с плавающей точкой
Время	Время значения
Недостоверность	Недостоверность значения
ParamId	Номер параметра
IntervalId	Номер интервала

Сохранение значений расчетного параметра производится, если в соответствующей записи списка расчетных параметров заполнено поле «**Тип накопления»**. Если расчет разовый, то значения независимо от типа накопления всегда попадают в список значений разовых интервалов. После каждого разового расчета в архив добавляется разовый интервал. Список значений, попадающий в таблицу разовых значений, полностью соответствует списку мгновенных значений расчетного параметра.

При периодическом расчете значения распределяются по таблицам архива, в зависимости от типа накопления параметра. Если указан тип накопления «**Мгновенные»,** то значения сохраняются в таблицу мгновенных значений. При этом в таблице интервалов для каждого проекта присутствует всего один мгновенный интервал за все время периодического расчета.

Если указан тип накопления с буквами «**П»** или «**ПА»** в скобках (например, Среднее(П), Максимум(ПА)), то итоговое значение по расчетному параметру записывается

в таблицу базовых значений. Итоговое значение параметра за час вычисляется по итоговым значениям за базовые периоды, составляющие этот час, в соответствии с типом накопления и попадает в таблицу часовых значений. Итоговое значение параметра за сутки вычисляется по итоговым значениям за часы, составляющие эти сутки, в соответствии с типом накопления и попадает в таблицу суточных значений. Соответствующим образом, по мере необходимости в таблицу интервалов добавляются базовые, часовые и суточные интервалы.

Если указан тип накопления с буквами «**А**» или «**ПА**» в скобках (например, Среднее(А), Максимум(ПА)), то в соответствии с типом накопления формируется абсолютное значение параметра (например, среднее) за все время расчета. Абсолютное значение помещается в таблицу абсолютных значений и заменяет значение, которое было туда записано до текущего цикла расчета.

Итоговые значения параметров на интервале по значениям интервалов меньшего размера (часовые по базовым, суточные по часовым, абсолютные по базовым), составляющим этот интервал, вычисляются следующим образом, исходя из типа обработки:

- **Сумма** – значение параметра за интервал вычисляется как сумма значений параметра за интервалы его составляющие;
- **Среднее** – значение параметра за интервал вычисляется как сумма значений параметра за интервалы его составляющие, деленная на количество этих интервалов;
- **Максимум** – значение параметра за интервал вычисляется как максимум из значений параметра за интервалы его составляющие;
- **Минимум** – значение параметра за интервал вычисляется как минимум из значений параметра за интервалы его составляющие;
- **Первое** – значение параметра за интервал вычисляется как значение параметра за первый интервал из его составляющих;
- **Последнее** – значение параметра за интервал вычисляется как значение параметра за последний интервал из его составляющих;
- **Мгновенные** – значения записываются в таблицу мгновенных значений.

Если конец последнего периода обработки раньше чем конец архива расчетных параметров, то это означает, что ранее производился за расчет за более поздние периоды обработки. В этом случае значения параметров в списке базовых, часовых и суточных значений заменяются на новые. Часовые и суточные заменяются только в случае, если есть все базовые или часовые значения за соответствующий период.

В таблицах архива расчетных параметров недостоверность заносится в соответствующее поле. Недостоверность за составной интервал равна максимуму недостоверностей его составляющих.

3.9. Порядок расчета параметров

Параметры, подпараметры, входы

Основой проекта InfoTask является таблица расчетных параметров. Каждый параметр из таблицы может иметь несколько подчиненных параметров – подпараметров.

Каждый параметр или подпараметр имеет код, по которому к нему можно обращаться. Код представляет собой идентификатор, то есть строку из русских и латинских букв, цифр и символов подчеркивания, содержащую хотя бы одну не цифру. Длина кода не должна превышать 50 (пятьдесят) символов. Язык Tablik не зависит от регистра, поэтому используемые в кодах заглавные и соответствующие строчные буквы считаются одинаковыми. Например, коды par, Par и PAR будут считаться одинаковыми. В проекте нельзя одновременно использовать два параметра с одинаковыми кодами. Параметр также не должен иметь два подпараметра с одинаковыми кодами.

Некоторые расчетные параметры и подпараметры могут иметь **входы** (входные значения, используемые в расчете), такие параметры называются **параметрами-функциями**. Чтобы задать параметр-функцию нужно для него в таблице расчетных параметров заполнить поле **Входы**. Поле Входы параметра-функции заполняется следующим образом:

<**Вход**>;<**Вход**>;…;<**Вход**>,где

<**Вход**>:=[< **Тип данных**>]<**Код входа**> [= <**Значение по умолчанию**>] или

<**Вход**>:=[< **Тип данных**>]<**Код входа**> **Сигнал (**<**Вход**>;<**Вход**>;…;<**Вход**>**),**

< **Тип данных**>:=<**Встроенный тип**> | <**Код параметра-функции**>.

Код входа должен представлять собой идентификатор длинной не более 50 (пятьдесят) символов. Таким образом, задается список входных параметров, разделенных точкой с запятой «;». Причем для каждого параметра можно указать тип данных.

Тип данных может представлять собой либо встроенный тип данных (логич, целое и т. п.), либо расчетный параметр-функцию. Если типом является параметр, то считается, что значению входа приписан этот параметр, и в выражении можно получать от него характеристики или подпараметры через точку. Если тип данных не указан, то считается, что входной параметр имеет действительный тип (действ).

Указанный в списке входов вход можно использовать в выражении как переменную. При расчете, эта переменная будет иметь значение, которое будет передано в функцию при вызове или значение по умолчанию, если значение не будет передано. Применение входов, описанных с использованием ключевого слова **Сигнал**, описано в следующем подразделе.

В формулах одних расчетных параметров можно использовать другие расчетные параметры. Для использования параметра без входов, в формуле достаточно указать его код. Синтаксис использования в формуле параметра-функции следующий:

<**Код параметра**>(<**Значение входа**>;…;<**Значение входа**>**),**

то есть параметр-функция вызывается как обычная функция, с указанием входных значений. Тип каждого передаваемого в параметр-функцию значения должен соответствовать типу соответствующего входа этого параметра-функции. Значение входа по умолчанию можно в параметр-функцию не передавать, в этом случае будет использовано значение входа по умолчанию.

Подпараметры и параметры в выражениях разделяются точкой. Для того чтобы использовать в формуле подпараметр нужно использовать следующую запись:

<**Вызов параметра**>.<**Вызов подпараметра**>,

при этом как параметр, так и подпараметр может вызываться с указанием значений входов или без, в зависимости от того нужны они или нет. Выражение вызова подпараметра SB параметра B в выражении может в разных случаях выглядеть как B.SB, B(1;2).SB, B.SB(1) или даже как B(1).SB(2;3).

Использование сигналов в качестве входов.

Можно объявить входом параметра-функции объект, содержащий указанный набор сигналов. В этом случае, в расчетном выражении можно использовать сигналы из перечисленного набора, используя функцию Сигнал. Такой параметр-функцию можно вызывать от различных объектов, содержащих перечисленные во входе сигналы.

Описание входа, задающего объект, выглядит следующим образом:

<**Вход>**:=[< **Тип данных**>]<**Код входа**> **Сигнал (**<**Сигнал**>**;**<**Сигнал**>**;**…**;**<**Сигнал**>**)**,

где <**Сигнал>**:=[< **Тип данных**>]<**Код входа**>.

При вызове такого параметра в качестве входа должен передаваться сигнал, тип данных которого совпадает с указанным типом данных, объект которого содержит все сигналы, перечисленные в скобках, и типы данных сигналов объекта подходят под типы перечисленных в скобках сигналов.

Например, пусть в проекте есть следующие сигналы с указанием, какие из них являются сигналами по умолчанию:

Код объекта	Код сигнала	Тип данных	По умолчанию
Ob1	SigA	целое	да
Ob1	SigB	действ	
Ob2	SigA	логич	да
Ob2	SigB	действ	
Ob2	SigC	целое	
Ob3	SigA	действ	да

В этом случае допустимо использование следующего набора расчетных параметров:

Код	Входы	Расчетное выражение
Fun	целое X Сигнал (целое SigA; действ SigB)	Сигнал(X; SigA) + Сигнал(X; SigB)
Par1		Fun({Ob1.SigA})
Par2		Fun({Ob2})

Параметр Fun имеет один вход X, в качестве которого может передаваться сигнал, объект которого содержит сигналы SigA и SigB. Причем тип самого сигнала должен сводиться к целому, тип сигнала SigA должен сводиться к целому, а тип сигнала SigB, должен сводиться к действительному. В формуле параметра Fun при помощи функции Сигнал производится выделение из переданного объекта сигналов SigA и SigB, а затем производится их сложение.

Параметр Par1 в своей формуле вызывает Fun от сигнала Ob1.SigA. В результате получится сумма сигналов Ob1.SigA и Ob1.SibB. Параметр Par2 в своей формуле вызывает Fun от объекта Ob2. В результате получится сумма сигналов Ob2.SigA и Ob2.SibB. В то же время вызов Fun({Ob3}) вызвал бы ошибку, так как объект Ob3 не имеет сигнала SigB.

Порядок расчета параметров. Использование параметров в формулах. Формирование архивных параметров

Порядок расчета параметров определяется автоматически при компиляции. Также по списку параметров при компиляции формируется список архивных параметров, записываемых в архив.

Ниже описаны различные допустимые способы вызова одного параметра из другого, а также принципы формирования архивных параметров из расчетных в различных случаях. Конкретные примеры описанных ниже случаев приведены в следующем подразделе.

- Один расчет по проекту заключается в том, что последовательно вычисляются значения всех расчетных параметров без входов и значения всех, принадлежащих им подпараметров без входов. Параметры с входами не обсчитываются, если они не используются в формулах других параметров. Параметры без входов также могут являться параметрами ручного ввода (см. раздел 3.12). Также результаты вычисления параметров без входов могут быть переданы в приемник (см. раздел 3.10).
- Если параметр «A» не имеет входов, то значение по нему будет посчитано. Если при этом для него задан тип накопления, то по нему формируется архивный параметр с кодом «A». Если подпараметр «SA» параметра «A» не имеет входов, то значение по нему также будет посчитано. Если при этом для него задан тип накопления, то по нему будет сформирован архивный параметр с кодом «A.SA».
- Пусть параметр «B» не имеет входов и используется в параметре «A». Если параметр «A» не имеет входов, то значение параметра «B» вычисляется после параметра «A».
- Пусть параметр «B» не имеет входов и используется в параметре-функции «C». В этом случае значение первого вызова «C» как параметра функции вычисляется позже вычисления значения параметра «B».
- Параметр-функция может быть вызван в формулах нескольких параметров. При расчете для него будет создано несколько экземпляров, каждый из которых вычисляется в зависимости от своих входных параметров. Если параметр-функция «C» используется в формуле параметра «D» (неважно с входами он или без), то сначала вычисляется часть формулы до вызова «C», затем вызывается «C» и полностью вычисляется его значение, затем завершается вычисление значения «D», уже с использованием результата вычисления «C».
- Если «SE» и «SD» подпараметры параметра «E», то они могут быть использованы в выражении параметра «E», а также в выражениях друг друга. Для использования достаточно указать код подпараметра без кода параметра. Если при этом подпараметр является подпараметром-функцией, то при его вызове нужно как обычно задать значения входов.
- Подпараметр «SF» параметра «F» может быть использован в другом параметре или подпараметре. Для этого в выражении нужно вызвать параметр «F» с заданием значений входов, если они нужны, а затем после точки вызвать подпараметр «SF» также с заданием входов, если они нужны.
- Каждое расчетное значение кроме самого значения может иметь так называемый **приписанный расчетный параметр**. Значению приписывается расчетный параметр, если оно является результатом расчета по нему. Приписан может быть параметр как без входов, так и с входами. Значение сохраняет приписанный параметр при присвоении в переменную. Приписанный параметр теряется при

расчете функции от него. Подробнее в подразделе Приписанные параметры раздела 3.4.

- Пусть параметр-функция «H» с одним входом используется в параметрах «F» и «G» следующим образом:

I	H(1)
J	V=H(2):V+1

Тогда значению параметра «I» и значению переменной «V» параметра «J» приписан параметр «H», а значению параметра «J» он не приписан.

- Если значению параметра «I» приписан параметр-функция «H», и для «H» заполнен тип накопления, а для «I» не заполнен, то по параметру «I» все равно будет сформирован архивный параметр с кодом «I» с типом накопления взятым от «H». Если для «I» заполнен тип накопления, то накопление производится в соответствии с типом накопления параметра «I».

- Если значению параметра «I» приписан параметр-функция «H» и у «H» есть подпараметр «SH» без входов, то после расчета параметра «I» будет произведен расчет подпараметра «SH», использующий данные параметра «H» приписанного «I». Если для параметра «SH» заполнен тип накопления, то по нему будет сформирован архивный параметр «I.SH».

- Параметр-функция с подпараметрами представляет собой целый расчетный блок. Если после вызова такого блока выражение заканчивается, то он оказывается приписанным вызывающему параметру, и производится расчет и накопление всех его подпараметров. Используя такие блоки, можно значительно сокращать список расчетных параметров, так как один вызов такого блока приводит к расчету и накоплению сразу нескольких подпараметров.

- Параметры-функции как блоки также можно использовать и в подпараметрах. Если параметр-функция «H» с подпараметром без входов «SH» и заданным типом накопления приписана подпараметру «SL» без входов параметра «L», то по значение «SH» будет вычислено после значения «SL», и по подпараметру «SH» будет сформирован архивный параметр «L.SL.SH».

- Если параметру «A» приписан параметр без входов «B», то в отличие от параметра-функции, тип накопления «B» не передается к «A», и подпараметры параметра «B» после расчета по параметру «A» не расчитываются и не накапливаются.

- Если элементу расчета приписан накапливаемый параметр без выходов, то от него можно вызвать архивную функцию для получения предыдущего значения. Например, если «A» – расчетный параметр с заполненным типом накопления, то допустим вызов «Пред(A;0)». Подробнее об архивных функциях в разделе 4.17.

Пример использования параметров

Следующая таблица описывает проект, иллюстрирующий все варианты вызова параметров, описанные в предыдущем подразделе. Для каждого параметра указан код, входы (если они есть), расчетное выражение, тип накопления (если задан), а также подпараметры, если они есть. Кроме того, для каждого непосредственно вычисляемого при расчете параметра указано значение. Управляющие выражения всех перечисленных параметров заполнены как «Расчет» и ни на что не влияют.

Код	Входы	Расчетное выражение	Тип накопления	Значение
A		B	Среднее(П)	1
SA		2	Среднее(ПА)	2

Код	Входы	Расчетное выражение	Тип накопления	Значение
B		1	Сумма(А)	1
SB		A	Сумма(ПА)	1
C	X	B+X		
D		1+C(1)+1	Максимум(ПА)	4
E		SE(1)+SD	Минимум(ПА)	5
SD		SE(2)	Минимум(П)	3
SE	X	X+1		
F	Z	Z		
SF		Z	Первое(П)	
SG	T	Z+T	Последнее(П)	
G		F(1).SF+F(2).SG(3)	Сумма(ПА)	6
H	Y	10*Y	Среднее(П)	
SH		20+Y	Минимум(П)	
SK		30	Максимум(П)	
I		H(1)		10
J		V=H(2):V+1	Мгновенные	21
L		1		1
SL		H(0)	Мгновенные	0

В следующей таблице приведены архивные параметры, которые будут сформированы на основе расчетных параметров из предыдущей таблицы, с указанием итоговых накапливаемых значений.

Код	Тип накопления	Значение	Код	Тип накопления	Значение
A	Среднее(П)	1	I	Среднее(П)	10
A.SA	Среднее(ПА)	2	I.SH	Минимум(П)	30
B	Сумма(А)	1	I.SK	Максимум(П)	30
B.SB	Сумма(ПА)	1	J	Мгновенные	21
D	Максимум(ПА)	4	L.SL	Мгновенные	0
E	Минимум(ПА)	5	L.SL.SH	Минимум(П)	20
E.SD	Минимум(П)	3	L.SL.SK	Максимум(П)	30
G	Сумма(ПА)	6			

3.10. Передача в приемники

Вычисленное значение расчетного параметра можно передать в объект приемника, для последующей передачи обратно в ПТК АСУТП. Приемником чаще всего является OPC сервер DA. Код объекта-приемника указывается в поле «**Получатель»** списка расчетных параметров. Поле «Получатель» может быть указано только для параметра, но не для подпараметра.

Если результатом вычисления параметра передаваемого в приемник является одно мгновенное значение, то оно и передается в приемник. Если же возвращается список из нескольких мгновенных значений, то в источник передается последнее из них.

3.11. Графики

Графики, которые могут быть использованы в расчете, хранятся в отдельной таблице. По своей сути график представляет собой функцию многих переменных, заданную поточечно.

Использование графика в расчетном выражении ничем не отличается от использования встроенной функции. Используется следующий синтаксис:

<Код Графика>([<Z6>;][<Z5>;][<Z4>;][<Z3>;][<Z2>;][<Z1>;][<X>]),

где <**Код Графика**> задает код используемого графика, а параметры в скобках координаты, от которых вычисляется значение. Размерность графика на единицу больше количества переменных в списке «*X, Z1, Z2* …».

Значение графика вычисляется при помощи интерполяции по набору заданных точек. Если тип графика задан как «**График»**, то используется линейная интерполяция. Если тип графика задан как «**График0»**, то используется ступенчатая интерполяция. Если тип графика задан как «**Диаграмма»**, то значение вычисляется только в случае, если значения входных параметров в точности совпадают с координатами одной из точек, иначе возвращается 0.

- Если используется **ступенчатая интерполяция** (тип графика «График0»), тогда:

– Для двумерного графика значение Y в точке X определяется по точкам (X1,Y1) и (X2,Y2) по формуле:

Y=Y1;

– Для n-мерного графика значение Y в точке (X,Z1,…,Zn-2) сначала вычисляются значения Y1 и Y2 n-1 мерных графиков при фиксированной координате Zn-2=Z1n-2 и Z2n-2, а затем вычисляется значение графика по формуле:

Y=Y1.

- Если используется **линейная интерполяция** (тип графика «График»), тогда:

– Для двумерного графика значение Y в точке X определяется по точкам (X1,Y1) и (X2,Y2) по формуле:

Y=Y1 + (Y2 - Y1)*(X - X1)/(X2 - X1);

– Для n-мерного графика значение Y в точке (X,Z1,…,Zn-2) сначала вычисляются значения Y1 и Y2 n-1 мерных графиков при фиксированной координате Zn-2=Z1n-2 и Z2n-2, а затем вычисляется значение графика по формуле:

Y=Y1 + (Y2 - Y1)*(Zn-2 - Z1n-2)/(Z2n-2 - Z1n-2).

Недостоверность (ошибка) значения графика равна максимуму из недостоверностей (ошибок) аргументов. Таким образом, значение графика корректно вычисляется, только если все аргументы определены.

Пример: Пусть выражение имеет вид: G(1;1),

где G – 3-х мерный график, задаваемый следующим образом:

Z1=0			Z1=2		
X	0	2	X	0	2
Y	0	1	Y	1	2

В этом случае G(1;1)= (G(1;0) + G(2;1))/(2-0),

G(0;1)=(G(0;0) + G(0;2))/(2-0),

G(2;1)=(G(2;0) + G(2;2))/(2-0).

Поэтому G(1;1)=((0+1)/(2-0)+(1+2)/(2-0))/(2-0)=1.

3.12. Параметры ручного ввода

В качестве исходных данных для некоторых вычислительных задач используются отдельные параметры, значения которых не могут быть получены из ПТК АСУТП, поскольку их автоматизированное измерение в реальном режиме времени затруднено или вообще невозможно. В таком случае при реализации этих задач используется ручной ввод значений данных параметров.

Для того чтобы в проекте InfoTask была возможность реализации ручного ввода значений, в состав провайдеров этого проекта должен быть добавлен встроенный провайдер – источник ручного ввода значений (тип провайдера – **Источник**, код провайдера **HandInputSource** или **HandInputSqlSource**). Данный провайдер производит чтение и передачу в расчет значений ручного ввода.

Для ввода ручных значений используются параметры ручного ввода. Параметр из таблицы расчетных параметров является параметром ручного ввода, если у него заполнено поле «**Тип параметра**» одним из следующих значений:

- **ВводЛогич** – ввод значений типа логич, 0 или 1;
- **ВводЦелый** – ввод целочисленных значений;
- **ВводДейств** – ввод вещественных значений;
- **ВводВремя** – ввод времени;
- **ВводСтрока** – ввод строк.

Подпараметры не могут быть параметрами ручного ввода.

Поле «Тип параметра» ограничивает тип значений, которые могут быть введены вручную. Для каждого параметра ручного ввода обязательно должно быть указано значение в поле «**По умолчанию**». Тип значения по умолчанию должен соответствовать типу, указанному в поле «**Тип параметра**».

Для того чтобы использовать введенное значение в расчетном или управляющем выражении можно использовать ключевое слово-переменную **Ручной**. Таким образом, значение параметра ручного ввода может не совпадать с введенным значением, а вычисляться на основе него. Если значение не было введено, то используется значение из поля «**По умолчанию**».

4. ВСТРОЕННЫЕ ФУНКЦИИ

4.1. Общее описание

В языке выражений есть некоторое количество встроенных функций и операций. При вычислении выражений значение функции вычисляется в зависимости от значений ее параметров. Вызов функции осуществляется следующим образом:

<**Имя функции**>(<**Параметр**>;…;<**Параметр**>).

В роли параметров могут выступать любые выражения.

Функция может не иметь параметров. В этом случае для ее вызова используется следующий синтаксис: <**Имя функции**>.

Скалярные функции

Скалярные функции – это более половины всех встроенных функций, а также все операции языка. По сути, скалярные функции – самые простые, они оперируют не списками мгновенных значений, а отдельными мгновенными значениями. Например, скалярными являются сложение, умножение, cos, sin и т.п. Скалярные функции работают только с входными параметрами простых типов данных, то есть «логич», «целое», «действ», «время» и «строка». Входными параметрами скалярных функций могут являться отдельные мгновенные значения или списки мгновенных значений.

При их вычислении для каждого момента времени определяется конкретное мгновенное значение каждого входного параметра, и над этими значениями выполняется простое действие. Таким образом, на выходе получается список мгновенных значений, вычисленных в моменты времени мгновенных значений входных параметров функции.

Значения входных параметров в конкретные моменты времени определяются с использованием интерполяции указанной для текущего расчетного параметра (ступенчатой или линейной). Если для расчетного параметра не указан тип интерполяции, то используется тип интерполяции проекта.

Для всех скалярных функций, кроме операций «И» и «Или», результат, полученный по некоторым мгновенным значениям, имеет недостоверность равную максимуму недостоверностей этих значений. Аналогичным образом результат, полученный по некоторым мгновенным значениям, имеет ошибку, если хотя бы одно из этих значений имеет ошибку. При этом из исходных ошибок выбирается первая попавшаяся. В этом случае в данном документе часто формулируется, что ошибка результата равна максимуму из ошибок исходных значений.

Рассмотрим порядок вычисления скалярных функций на примере функции получения модуля числа и операции сложения. Пусть в проекте есть два сигнала S1 и S2, и расчет производится за период от 0:00:00 до 1:00:00. Пусть в проекте используется ступенчатая интерполяция. В следующей таблице приведен пример мгновенных значений сигналов S1 и S2 вместе с недостоверностями, а также мгновенные значения результатов вычисления выражений Модуль({S1}) и {S1}+{S2} вместе с недостоверностями.

Время	{S1}		{S2}		Модуль({S1})		{S1}+{S2}	
	Значение	Недост.	Значение	Недост.	Значение	Недост.	Значение	Недост.
0:00:00	2	0	2	0	2	0	4	0
0:10:00	-2	0			2	0	0	0
0:15:00			3	0			1	0
0:25:00	3	1			3	1	6	1
0:30:00			4	0			7	1
0:40:00	-4	0	5	1	4	0	1	1
0:45:00			3	0			-1	0
0:55:00	1	1		1	1	1	4	1

Для каждого мгновенного значения сигнала S1 результат вычисления функции Модуль ({S1}) содержит также одно мгновенное значение с таким же временем, недостоверностью и величиной равной модулю величины исходного значения.

Моменты времени мгновенных значений выражения {S1}+{S2} соответствуют мгновенным значениям сигналов S1 и S2. Для каждого момента времени вычисляются значения исходных сигналов. Если на какой-то момент времени в исходном списке сигнала нет значения, то значение берется равным предыдущему значению, так как интерполяция задана ступенчатая. Для каждого из полученных моментов времени значение результата вычисляется как сумма исходных значений, недостоверность вычисляется как максимум исходных недостоверностей.

Если какой-то из входных параметров функции имеет константное значение, то это значение приписывается ему на все моменты времени мгновенных значений других входных параметров. Если все входные значения представляют собой константные значения, то на выходе также получается одно мгновенное значение.

Таким образом, если про функцию известно, что она скалярная (это обычно сказано в начале раздела, описывающего группу функций), то для нее описывается только действие на отдельных мгновенных значениях, потому что этого достаточно, чтобы определить действия также и на списках значений.

Группы функций

Встроенные функции языка Tablik разбиты на группы. В следующей таблице приведены все группы функций с указанием следующей информации:

- номер;
- имя;
- описание;
- вид (указан, если группа состоит из скалярных функций);
- раздел данного документа, в котором подробно описываются функции этой группы.

№	Имя группы	Описание	Вид	Раздел
1.	Операции	Операции	скалярные	4.2
2.	Логические	Функции работы с битами, логические константы	скалярные	4.3
3.	Математические	Математические функции	скалярные	4.4
4.	Недостоверность	Работа с недостоверностью и ошибками, а также достоверизация	скалярные	4.5
5.	Точки	Функции работы с мгновенными значениями		4.6
6.	Статистические	Статистические функции		4.7
7.	Сегменты	Функции формирования сегментов		4.8
8.	Таблицы	Получение данных из дополнительных таблиц проекта		4.9
9.	Типы данных	Проверка и преобразование типов данных	скалярные	4.10
10.	Время	Функции для работы со временем	скалярные	4.11
11.	Строковая	Функции для работы со строками	скалярные	4.12
15.	Смесь газов	Термодинамические функции и параметры для их вызова	скалярные	4.13
16.	Вода и пар		скалярные	4.14
17.	WSP параметры		скалярные	4.15
19.	Ключевые слова	Ключевые слова		4.16
20.	Условия и циклы	Функции условий и циклов		4.19
21.	Архивные	Получение значений предыдущих расчетов из архива	предыдущие расчеты	4.18
22.	Сигналы и параметры	Обращение с сигналам, параметрам и характеристикам		4.17

Операции и функции могут по-разному выполняться и возвращать разный тип результата для разных типов входных аргументов. Разные способы вызова в зависимости от входных типов аргумента называются **сигнатурами** вызова функций или операций. Например, операция сложения «+» может применяться к целым числам и возвращать целое число, а может применяться к действительным числам и возвращать действительное число. То есть, среди прочих у операции «+» есть сигнатуры **целое = целое + целое** и **действ = действ + действ.** При этом если типы аргументов операции не подходят ни под одну сигнатуру, но могут быть сведены к типам одной из сигнатур, то будет выбрана сигнатура с наменьшими в смысле сводимости возможными типами аргументов. Например, если один из аргументов операции «+» будет иметь тип «логич», а другой тип «целое», то для расчета будет выбрана сигнатура **целое = целое + целое**, а не **действ = действ + действ.**

Недостоверность (ошибка) значения любой операции, кроме логических «Or» и «And», равна максимуму недостоверностей (ошибок) операндов. Значение недостоверности логической операции «Or» вычисляется как максимум недостоверностей аргументов, если все они равны 0 («Ложь») и как минимум недостоверностей аргументов равных 1 («Правда») иначе. Значение недостоверности логической операции «And» вычисляется как максимум недостоверностей аргументов, если все они равны 1 («Правда») и как минимум недостоверностей аргументов равных 0 («Ложь») иначе.

4.2. Операции

Общие сведения

Любую операцию (арифметическую, логическую, сравнения) можно рассматривать как функцию, только записанную по-другому. Поэтому операции в языке Tablik рассматриваются вместе с функциями. Все операции языка содержатся в группе функций «Операции». Все операции являются скалярными функциями (см. раздел 4.1).

Операции в языке бывают унарные и бинарные. Унарная операция одному числу сопоставляет одно число, бинарная сопоставляет двум числам одно число. В целом использование операций в языке выражений стандартно и ничем не отличается от большинства других языков программирования.

Операции имеют следующие приоритеты:

Приоритет	6	5	4	3	2	1	0
Операции	- (унарный), Not	And, Or, Xor	^	*, /, Div, Mod	+, -	<, >, >=, <=, <>, Like	==

Сначала в выражении выполняются операции с более высоким приоритетом, затем с более низким. Для задания порядка операций можно использовать круглые скобки.

Например, в выражении (7>2*(3+4^2)) And (1<=3) операции выполняются в следующем порядке: ^, +, *, >, <=, And.

Унарные операции

Унарные операции языка используют следующий синтаксис:

<**Результат операции**>:=<**Операция**><**Операнд**>.

Язык содержит следующие унарные операции:

- **Минус** (-) – стандартный арифметический минус, изменяет знак целого или вещественного числа на противоположный.

Возможные сигнатуры:

целое возвращает **целое;**

действ возвращает **действ.**

- **Отрицание** (**Not**) – логическое отрицание: Not(0)=1, Not(1)=0.

Сигнатура: **Not логич,** возвращает **логич**.

Бинарные операции

Бинарные операции используют следующий синтаксис:

<**Результат операции**>:=<**Операнд**><**Операция**><**Операнд**>,

например: 2+3; {4RM03E002.3.0} Or {4RM03E002.3.1}; 10 > {4RM03E002}.

В языке Tablik представлены стандартные арифметические операции, логические операции и операции сравнения.

Арифметические операции

- **Сложение** (+). Имеет различный смысл в зависимости от сигнатуры.

Возможные сигнатуры:

целое + **целое** возвращает **целое**, арифметическое сложение целых чисел;

действ + **действ** возвращает **действ**, арифметическое сложение вещественных чисел;

действ + **время** возвращает **время**, прибавляет ко времени заданное число секунд;

время + **действ** возвращает **время**, прибавляет ко времени заданное число секунд;

строка + **строка** возвращает **строку**, приписывает вторую строку к первой.

Примеры вычисления:

2 + 3 = 5;

2,5 + 3,2 = 5,7;

01.01.2014 00:00:00 + 45 = 01.01.2014 00:00:45;

‘xxx’ + ‘yyy’ = ‘xxxyyy’.

- **Вычитание** (-). Имеет различный смысл в зависимости от сигнатуры.

Возможные сигнатуры:

целое - целое возвращает **целое**, арифметическое вычитание целого числа из другого целого числа;

действ - действ возвращает **действ**, арифметическое вычитание вещественного числа из другого вещественного числа;

время - действ возвращает **время**, вычитает из времени заданное число секунд;

время - время возвращает **действ**, вычитает из одного времени другое и возвращает результат в секундах.

Примеры вычисления:

2 - 7 = -5;

5,5 - 3,2 = 2,3;

01.01.2014 00:00:45 - 45 = 01.01.2014 00:00:00;

01.01.2014 00:00:45 - 01.01.2014 00:00:00 = 45.

- **Умножение** (*). Имеет различный смысл в зависимости от сигнатуры.

Возможные сигнатуры:

целое * целое возвращает **целое**, арифметическое умножение целых чисел;

действ * действ возвращает **действ**, арифметическое умножение вещественных чисел.

Примеры вычисления: 2 * 3 = 6; 1,5 * 1,5 = 2,25.

- **Деление** (/). Деление одного вещественного числа на другое. Если второй аргумент равен 0, то возвращает ошибку.

Сигнатура: **действ** / **действ** возвращает **действ**.

Пример: 6 / 1,5 = 4.

- **Целочисленное деление** (**Div**). Деление одного целого числа на другое с отбрасыванием дробной части. Если второй аргумент равен 0, то возвращает ошибку.

Сигнатура: **целое Div целое** возвращает **целое**.

Пример: 5 Div 2 = 2.

- **Остаток от деления** (**Mod**). Нахождение остатка деления одного целого числа на другое. Если второй аргумент меньше либо равен 0, то возвращает ошибку.

Сигнатура: **целое Mod целое** возвращает **целое**.

Пример: 5 Div 2 = 1.

- **Возведение в степень** (^). Возведение вещественного числа (первый аргумент) в вещественную степень (второй аргумент). Если первый аргумент отрицательный и не целый, то возвращает ошибку.

Сигнатура: **действ ^ действ** возвращает **действ**.

Примеры: 2 ^ 3 = 8; 16 ^ 0,5 = 4.

Логические операции

- **И**, английский вариант – **And.** «И» над логическими значениями или побитовое «И» над целочисленными значениями. Логическое «И» имеет следующую таблицу значений (0-Ложь, 1-Истина):

И	0	1
0	0	0
1	0	1

Возможные сигнатуры:

логич И логич возвращает **логич**, логическое «И»;

целое И целое возвращает **целое**, побитовое «И».

Примеры: 1 И 0 = 0; 3 И 6 = 2.

- **Или**, английский вариант – **Or.** «ИЛИ» над логическими значениями или побитовое «ИЛИ» над целочисленными значениями. Логическое «ИЛИ» имеет следующую таблицу значений (0-Ложь, 1-Истина):

Или	0	1
0	0	1
1	1	1

Возможные сигнатуры:

логич Или логич возвращает **логич**, логическое «ИЛИ»;

целое Или целое возвращает **целое**, побитовое «ИЛИ».

Примеры: 1 Или 0 = 0; 3 Или 6 = 7.

- **ИсклИли**, английский вариант – **Xor.** Исключающее «ИЛИ» над логическими значениями или побитовое над целочисленными значениями. Логическое исключающее «ИЛИ» имеет следующую таблицу значений (0-Ложь, 1-Истина):

ИсклИли	0	1
0	0	1
1	1	0

Возможные сигнатуры:

логич ИсклИли логич возвращает **логич**, логическое исключающее «ИЛИ»;

целое ИсклИли целое возвращает **целое**, побитовое исключающее «ИЛИ».

Примеры: 1 ИсклИли 0 = 1; 3 ИсклИли 6 = 5.

Операции сравнения

- **Равно** (==). Сравнивает два значения. Возвращает «1» (True), если значения равны, и «0» (False), если значения не равны.

Возможные сигнатуры:

логич == **логич** возвращает **логич**, сравнение логических значений;

целое == **целое** возвращает **целое**, сравнение целочисленных значений;

действ == **действ** возвращает **действ**, сравнение вещественных значений;

время == **время** возвращает **время**, сравнение времен, времена сравниваются вместе с дробными частями секунд;

строка == **строка** возвращает **строка**, сравнение строк.

Примеры: 1 == 0 = 0; 3.4 == 3.4 = 0; ‘sss’ == ’sss’ = 0.

- **Не равно** (<>). Сравнивает два значения. Возвращает «0» (False), если значения равны, и «1» (True), если значения не равны.

Возможные сигнатуры:

логич <> **логич** возвращает **логич**, сравнение логических значений;

целое <> **целое** возвращает **целое**, сравнение целочисленных значений;

действ <> действ возвращает **действ**, сравнение вещественных значений;

время <> **время** возвращает **время**, сравнение времен, времена сравниваются вместе с дробными частями секунд;

строка <> **строка** возвращает **строка**, сравнение строк.

Примеры: 1 <> 0 = 1; 3.4 <> 3.4 = 0; ‘sss’ <> ’ttt’ = 1.

- **Больше** (>). Сравнивает два значения. Возвращает «1» (True), если первое значение больше второго, и «0» (False) иначе.

Возможные сигнатуры:

целое > **целое** возвращает **целое**, сравнение целочисленных значений;

действ > действ возвращает **действ**, сравнение вещественных значений;

время > **время** возвращает **время**, сравнение времен, времена сравниваются вместе с дробными частями секунд;

строка > строка возвращает **строка**, сравнение строк, строки считаются упорядоченными в лексикографическом порядке.

Примеры: 3 > 3 = 0; 3.4> 3.3 = 1; ‘aab’ > ’aaa’ = 1.

- **Меньше** (<). Сравнивает два значения. Возвращает «1» (True), если первое значение меньше второго, и «0» (False) иначе.

Возможные сигнатуры:

целое < целое возвращает **целое**, сравнение целочисленных значений;

действ < действ возвращает **действ**, сравнение вещественных значений;

время < **время** возвращает **время**, сравнение времен, времена сравниваются вместе с дробными частями секунд;

строка < строка возвращает **строка**, сравнение строк, строки считаются упорядоченными в лексикографическом порядке.

Примеры: 3 < 3 = 0; 3.3< 3.4 = 1; ‘aaa’ < ’aab’ = 1.

- **Больше или равно** (**>=**). Сравнивает два значения. Возвращает «1» (True), если первое значение больше или равно второму, и «0» (False) иначе.

Возможные сигнатуры:

целое >= **целое** возвращает **целое**, сравнение целочисленных значений;

действ >= действ возвращает **действ**, сравнение вещественных значений;

время >= **время** возвращает **время**, сравнение времен, времена сравниваются вместе с дробными частями секунд;

строка >= строка возвращает **строка**, сравнение строк, строки считаются упорядоченными в лексикографическом порядке.

Примеры: 3 >= 3 = 1; 3.4>= 3.3 = 1; ‘aaa’ >= ’aab’ = 0.

- **Меньше или равно** (**<=**). Сравнивает два значения. Возвращает «1» (True), если первое значение меньше или равно второму, и «0» (False) иначе.

Возможные сигнатуры:

целое <= целое возвращает **целое**, сравнение целочисленных значений;

действ <= действ возвращает **действ**, сравнение вещественных значений;

время <= **время** возвращает **время**, сравнение времен, времена сравниваются вместе с дробными частями секунд;

строка <= строка возвращает **строка**, сравнение строк, строки считаются упорядоченными в лексикографическом порядке.

Примеры: 3 <= 3 = 1; 3.3<= 3.4 = 1; ‘aab’ <= ’aaa’ = 0.

- **Подобно**, английский вариант – **Like**. Проверяет, удовлетворяет ли строка заданному шаблону. В качестве шаблона может выступать строка с использованием специальных символов. Символ «*» означает произвольную строку, символ «?» означает произвольный символ.

Сигнатура: **строка Подобно строка**, возвращает **логич**.

Примеры:

‘abbc’ Подобно ‘*bc’ = True, так как символ ‘*’ заменяет строку ‘ab’;

‘abbc’ Подобно ‘a*b?’ = True, так как символ ‘*’ заменяет строку ‘b’, а символ ‘?’ заменяет символ ‘c’;

‘abbc’ Подобно ‘b*’ = False.

4.3. Логические функции

Данный раздел описывает группу функций «Логические». Все логические функции являются скалярными (см. раздел 4.1). Функции работы с битами оперируют с 32-битными представлениями целых чисел. Биты в записи числа нумеруются с 0 и записываются слева направо по убыванию номера.

Целые числа в языке являются 32-битными знаковыми, то есть могут иметь значения от -2^{31} до $2^{31}-1$. Первые 31 бит записи такого числа задают значение, старший бит задает знак. По сути, работа с битами и побитовыми сдвигами ничем не отличается от безнаковых 32-битных чисел, поэтому язык позволяет полнеценно обрабатывать 32-битные значения, полученные из ПТК АСУТП. Например, можно свободно получать биты из 32-битных слов состояния, недостоверности и т.п.

Функция Sl

Назначение: Побитный сдвиг целого числа влево.

Синтаксис: **Sl(Выражение;КоличествоБит).**

Сигнатура: **целое** Sl (**целое** Выражение; **целое** *КоличествоБит*).

Параметры:

Выражение – выражение, от значения которого производится сдвиг;

КоличествоБит – на сколько бит производится сдвиг влево.

Пример: Sl(3; 2)=12.

Функция Sr

Назначение: Побитный сдвиг целого числа вправо.

Синтаксис: **Sr(Выражение;КоличествоБит).**

Сигнатура: **целое** Sr (**целое** Выражение; **целое** *КоличествоБит*).

Параметры:

Выражение – выражение, от значения которого производится сдвиг;

КоличествоБит – на сколько бит производится сдвиг вправо.

Пример: Sr(14; 2)=3.

Функция Бит

Английское имя: Bit.

Назначение: Получить от числа бит с указанным номером.

Синтаксис: **Бит(Выражение;НомерБита).**

Сигнатура: **логич** Бит (**целое** Выражение; **целое** *НомерБита*).

Параметры:

Выражение – выражение, от значения которого берется бит;

НомерБита – номер получаемого бита, должен быть в диапазоне от 0 до 31.

Пример: Бит(12;2)=1, Бит(12;1)=0.

Функция БитИ

Английское имя: BitAnd.

Назначение: Взять из целого числа указанные биты, сложенные по «И».

Синтаксис: **БитИ(Выражение;НомерБита1;…;НомерБитаN).**

Сигнатура: **логич** БитИ (**целое** *Выражение*; **целое** *НомерБита1*;…; **целое** *НомерБитаN*).

Параметры:

Выражение – выражение, от значения которого берутся биты;

НомерБита1, ..., НомерБитаN – номера используемых битов, должны быть в диапазоне от 0 до 31. Можно перечислить сколько угодно битов.

Пример: БитИ(12;2;3)=1.

Функция БитИли

Английское имя: BitOr.

Назначение: Взять из целого числа указанные биты, сложенные по «ИЛИ».

Синтаксис: **БитИли(Выражение;НомерБита1;…;НомерБитаN).**

Сигнатура: **логич** БитИли (**целое** *Выражение*; **целое** *НомерБита1*;… ; **целое** *НомерБитаN*).

Параметры:

Выражение – выражение, от значения которого берутся биты;

НомерБита1, ..., НомерБитаN – номера используемых битов, должны быть в диапазоне от 0 до 31. Можно перечислить сколько угодно битов.

Пример: БитИли(12;1;2)=1.

Константа Ложь

Английское имя: False.

Назначение: Константа, задающая логическое значение «Правда». Эквивалентна константе 0.

Синтаксис: **Ложь.**

Сигнатура: **логич** «Ложь».

Константа Правда

Английское имя: True.

Назначение: Константа, задающая логическое значение «Правда». Эквивалентна константе 1.

Синтаксис: **Правда.**

Сигнатура: **логич** «Правда».

4.4. Математические функции

Данный раздел описывает группу функций «Математические». Все математические функции являются скалярными.

Функция Arcch

Назначение: Вычисление гиперболического арккосинуса выражения. Результат измеряется в радианах.

Синтаксис: **Arcch(X).**

Сигнатура: **действ** Arcch (**действ** *X*).

Параметр: *X* – выражение, гиперболический арккосинус которого вычисляется.

Пример: Arcch(1)=0.

Функция Arccos

Назначение: Вычисление арккосинуса выражения. Результат измеряется в радианах.

Синтаксис: **Arccos(X).**

Сигнатура: **действ** Arccos(**действ** *X*).

Параметр: *X* – выражение, арккосинус которого вычисляется.

Пример: Arccos(1)=0.

Функция Arcsh

Назначение: Вычисление гиперболического арксинуса выражения. Результат измеряется в радианах.

Синтаксис: **Arcsh(X).**

Сигнатура: **действ** Arcsh (**действ** *X*).

Параметр: *X* – выражение, гиперболический арксинус которого вычисляется.

Пример: Arcsh(0)=0.

Функция Arcsin

Назначение: Вычисление арксинуса выражения. Результат измеряется в радианах.

Синтаксис: **Arcsin(X).**

Сигнатура: **действ** Arcsin (**действ** *X*).

Параметр: *X* – выражение, арксинус которого вычисляется.

Пример: Arcsin(0)=0.

Функция Arctan

Назначение: Вычисление арктангенса выражения. Результат измеряется в радианах.

Синтаксис: **Arctan(X).**

Сигнатура: **действ** Arctan (**действ** *X*).

Параметр: *X* – выражение, арктангенс которого вычисляется.

Пример: Arctan(0)=0.

Функция Arcth

Назначение: Вычисление гиперболического арктангенса выражения. Результат измеряется в радианах.

Синтаксис: **Arcth(X).**

Сигнатура: **действ** Arcth (**действ** *X*).

Параметр: *X* – выражение, гиперболический арктангенс которого вычисляется.

Пример: Arcth(0)=0.

Функция Ch

Назначение: Вычисление гиперболического косинуса угла. Угол задается в радианах.

Синтаксис: **Ch(X).**

Сигнатура: **действ** Ch (**действ** *X*).

Параметр: *X* – выражение, величина угла в радианах, гиперболический косинус которого вычисляется.

Пример: Ch(0)=1.

Функция Cos

Назначение: Вычисление косинуса угла.

Синтаксис: **Cos(X).**

Сигнатура: **действ** Cos (**действ** *X*).

Параметр: *X* – выражение, величина угла в радианах, косинус которого вычисляется.

Пример: Cos(0)=1, Cos(3,14159265358979)=-1.

Функция Ctan

Назначение: Вычисление котангенса угла.

Синтаксис: **Ctan(X).**

Сигнатура: **действ** Ctan (**действ** *X*).

Параметр: *X* – выражение, величина угла в радианах котангенс которого вычисляется.

Пример: Ctan(1,570796326794895)=0.

Функция Exp

Назначение: Вычисление экспоненты от выражения (возведение числа *e* в степень равную аргументу).

Синтаксис: **Exp(X).**

Сигнатура: **действ** Exp (**действ** *X*).

Параметр: *X* – выражение, степень в которую возводится число *e*.

Пример: Exp(0)=1.

Функция Ln

Назначение: Взятие натурального (по основанию *e*) логарифма от выражения.

Синтаксис: **Ln(X).**

Сигнатура: **действ** Ln (**действ** *X*).

Параметр: *X* – выражение, от которого берется логарифм. Значение выражения *X* должно быть положительным.

Пример: Ln(1)=0.

Функция Log

Назначение: Взятие логарифма от выражения по произвольному основанию.

Синтаксис: **Log(X;A).**

Сигнатура: **действ** Log (**действ** *X*; **действ** *A*).

Параметры:

X – выражение, от которого берется логарифм;

A – основание, по которому берется логарифм. *A* и *X* должны быть положительными.

Пример: Log(8;2)=3.

Функция Log10

Назначение: Взятие десятичного (по основанию 10) логарифма от выражения.

Синтаксис: **Log10(X).**

Сигнатура: **действ** Log10 (**действ** *X*).

Параметры: *X* – выражение, от которого берется логарифм. Значение выражения *X* должно быть положительным.

Пример: Log10(100)=2.

Функция Max

Назначение: Выбор максимального числа из значений набора выражений.

Синтаксис: **Max(X1;X2;…;Xn).**

Возможные сигнатуры:

логич Max (**логич** *X1*;…; **логич** *Xn*);

целое Max (**целое** *X1*;…; **целое** *Xn*);

действ Max (**действ** *X1*;…; **действ** *Xn*);

время Max (**время** *X1*;…; **время** *Xn*);

строка Max (**строка** *X1*;…; **строка** *Xn*).

Параметры: *X1, X2,…, Xn* – набор выражений, из значений которых выбирается максимальное. Тип данных результата получается как максимум типов данных аргументов.

Пример: Max(3;4;1)=4.

Функция Min

Назначение: Выбор минимального числа из значений набора выражений.

Синтаксис: **Min(X1;X2;…;Xn).**

Возможные сигнатуры:

логич Min (**логич** *X1*;…; **логич** *Xn*);

целое Min (**целое** *X1*;…; **целое** *Xn*);

действ Min (**действ** *X1*;…; **действ** *Xn*);

время Min (**время** *X1*;…; **время** *Xn*);

строка Min (**строка** *X1*;…; **строка** *Xn*).

Параметры: *X1, X2,..., Xn* – набор выражений, из значений которых выбирается минимальное. Тип данных результата получается как максимум типов данных аргументов.

Пример: Min(3;4;1)=1.

Константа Pi

Назначение: Число Пи.

Синтаксис: **Pi.**

Сигнатура: **действ** Pi.

Pi=3,14159265358979.

Функция Sh

Назначение: Вычисление гиперболического синуса угла. Угол измеряется в радианах.

Синтаксис: **Sh(X).**

Сигнатура: **действ** Sh (**действ** *X*).

Параметр: *X* – выражение, величина угла в радианах синус которого вычисляется.

Пример: Sh(0)=0.

Функция Sin

Назначение: Вычисление синуса угла.

Синтаксис: **Sin(X).**

Сигнатура: **действ** Sin (**действ** *X*).

Параметр: *X* – выражение, величина угла в радианах синус которого вычисляется.

Пример: Sin(0)=0.

Функция Sqr

Назначение: Извлечение квадратного корня из выражения.

Синтаксис: **Sqr(X).**

Сигнатура: **действ** Sqr (**действ** *X*).

Параметр: *X* – выражение, из которого извлекается корень. Значение выражения *X* должно быть неотрицательным.

Пример: Sqr(4)=2.

Функция Tan

Назначение: Вычисление тангенса угла.

Синтаксис: **Tan(X).**

Сигнатура**: действ** Tan **(действ** *X***).**

Параметр: *X* – выражение, величина угла в радианах тангенс которого вычисляется.

Пример: Tan(0)=0.

Функция Th

Назначение: Вычисление гиперболического тангенса угла.

Синтаксис: **Th(X).**

Сигнатура**: действ** Th **(действ** *X***).**

Параметр: *X* – выражение, величина угла в радианах гиперболический тангенс которого вычисляется.

Пример: Th(0)=0.

Функция Знак

Английское имя: Sign.

Назначение: Вычисление знака выражения. От положительных чисел возвращает «1», от отрицательных возвращает «-1», от нуля возвращает «0».

Синтаксис: **Знак(X).**

Сигнатура**: целое** Знак **(действ** *X***).**

Параметр: *X* – выражение, знак которого определяется.

Примеры: Знак(2)=1, Знак(-2)=-1.

Функция Модуль

Английское имя: Abs.

Назначение: Вычисление абсолютного значения (модуля) выражения.

Синтаксис: **Модуль(X).**

Возможные сигнатуры**:**

целое Модуль **(целое** *X***);**

действ Модуль **(действ** *X***).**

Параметр: *X* – выражение, от которого берется модуль.

Примеры: Модуль(2)=2, Модуль(-2)=2.

Функция Округлить

Английское имя: Round.

Назначение: Округление значения выражения до определенного количества знаков после запятой.

Синтаксис: **Округлить(X[;N]).**

Сигнатура**: действ** Округлить **(действ** *X*[; **целое** *N*]**).**

Параметры:

X – округляемое выражение;

N – количество знаков до которого округляется (по умолчанию 0).

Пример: Округлить(3,472;2)=3,47, Округлить(-5,757)=-6.

Функция Случайное

Назначение: Возвращает случайное число. В зависимости от параметров возвращается либо случайное действительное число от 0 до 1, либо случайное целое число из указанного диапазона.

Синтаксис: **Случайное([От; До]).**

Возможные сигнатуры**:**

действ Случайное;

целое Случайное(**целое** *От*; **целое** *До*).

Параметры: Если параметров нет, то возвращает случайное действительное число от 0 до 1.

Если параметры заданы, то возвращается целое число от значения «*От»* до значения «*До»*.

Примеры: Случайное, Случайное(1;100).

4.5. Функции работы с недостоверностью и ошибками

Данный раздел описывает группу функций «Недостоверность». Все функции работы с недостоверностью и ошибками являются скалярными.

Функция Недост

Английское имя: Uncertain.

Назначение: Возвращает недостоверность выражения.

Синтаксис: **Недост(Выражение).**

Сигнатура: **целое** Недост **(вариант** *Выражение***).**

Пример: Недост({KKS1}).

Функция Ошибка

Английское имя: Error.

Назначение: Возвращает текст ошибки выражения. Если выражение не имеет ошибки, то возвращает пустую строку.

Синтаксис: **Ошибка(Выражение).**

Сигнатура: **строка** Ошибка **(вариант** *Выражение***).**

Пример: Ошибка({KKS1}).

Функция ЗаписатьНедост

Английское имя: MakeCertain.

Назначение: Присваивает выражению заданную недостоверность. Значение выражения при этом не изменяется.

Синтаксис: **ЗаписатьНедост(Выражение; Недостоверность).**

Возможные сигнатуры:

логич ЗаписатьНедост **(логич** *Выражение*; **целое** *Недостоверность*);

целое ЗаписатьНедост **(целое** *Выражение*; **целое** *Недостоверность*);

действ ЗаписатьНедост **(действ** *Выражение*; **целое** *Недостоверность*);

время ЗаписатьНедост **(время** *Выражение*; **целое** *Недостоверность*);

строка ЗаписатьНедост **(строка** *Выражение*; **целое** *Недостоверность*).

Параметры:

Выражение – выражение, недостоверность которого устанавливается;

Недостоверность – значение устанавливаемой недостоверности.

Функция возвращает значение, равное значению *Выражения* и имеющее тот же тип данных, с недостоверностью, равной значению параметра *Недостоверность*.

Пример: ЗаписатьНедост({KKS};1).

Функция ЗаписатьОшибку

Английское имя: MakeError.

Назначение: Присваивает выражению заданную ошибку. Значение выражения при этом не изменяется.

Синтаксис: **ЗаписатьОшибку(Выражение;Ошибка;[Заменять]).**

Возможные сигнатуры:

логич ЗаписатьОшибку **(логич** *Выражение*; **строка** *Ошибка* [;**логич** *Заменять*]);

целое ЗаписатьОшибку **(целое** *Выражение*; **строка** *Ошибка* [;**логич** *Заменять*]);

действ ЗаписатьОшибку **(действ** *Выражение*; **строка** *Ошибка* [;**логич** *Заменять*]);

время ЗаписатьОшибку **(время** *Выражение*; **строка** *Ошибка* [;**логич** *Заменять*]);

строка ЗаписатьОшибку **(строка** *Выражение*; **строка** *Ошибка* [;**логич** *Заменять*]).

Параметры:

Выражение – выражение, которому приписывается ошибка;

Ошибка – текст, приписываемой ошибки;

Заменять – нужно ли заменять ошибку на новую, если *Выражение* уже имело ошибку; по умолчанию True.

Функция возвращает значение, равное значению *Выражения* и имеющее тот же тип данных, с ошибкой, равной значению параметра *Ошибка.* Если *Выражение* не имело ошибки или *Заменять*=«True» и с исходной ошибкой, то иначе.

Пример: ЗаписатьОшибку({KKS};’Приписанная ошибка’).

Функция Достовер

Английское имя: Certain.

Назначение: Достоверный результат по набору входных значений. Вычисляется как среднее всех достоверных значений из списка или как нормативное значение, если все параметры недостоверны.

Синтаксис: **Достовер(Рнорм;P1;P2;P3;…;Pn).**

Сигнатура: **дейстр** Достовер **(дейстр** *Рнорм*; **дейстр** *P1*; **…; дейстр** *Pn*).

Параметры:

P1, …, *Pn* – значения, исходя из которых вычисляется достоверное;

Рнорм – нормативное значение.

Значение функции вычисляется следующим образом:

- Из значений *P1*, …, *Pn* выбираются только те значения *R1, ..., Rm*, которые не имеют ошибки и достоверность которых равна 0.
- Если *m=0*, то есть значений *R1, ..., Rm* не нашлось, то функция возвращает *Рнорм*.
- Если *m>0*, то функция возвращает $(R1 + \cdots + Rm)/m$.

Пример: Пусть в выражении используется вызов функции

Достовер(100;{A};{B};{C};{D};{E}),

где A, B, C, D и E – сигналы со следующими значениями и недостоверностями:

A		B		C		D		E	
Знач.	Недост.	Знач.	Недост.	Знач.	Недост.	Знач.	Недост.	Знач.	Недост.
90	0	93	0	145	0	120	0	-50	1

В этом случае в качестве значений $R1, \ldots, R4$ будут взяты значения сигналов «A», «B», «C» и «D». Значение сигнала «E» будет отброшено, как недостоверное. Результатом функции будет достоверное значение (90+93+145+120)/4=112.

Функция ДостоверН

Английское имя: CertainN.

Назначение: Достоверный результат по набору входных значений. Вычисляется как среднее всех значений из списка, которые достоверны и отклоняются от нормативного значения не более, чем на указанную величину. Если таких значений в списке не нашлось, то возвращается нормативное значение.

Синтаксис: **ДостоверН(Рнорм;dн;P1;P2;P3;…;Pn).**

Сигнатура: **дейстр** ДостоверН **(дейстр** *Рнорм*; **дейстр** *днорм*; **дейстр** *P1*; **…; дейстр** *Pn*).

Параметры:

P1, …, *Pn* – значения, исходя из которых вычисляется достоверное;

Рнорм – нормативное значение;

днорм – допустимое отклонение от нормативного значения.

Значение функции вычисляется следующим образом:

- Из значений *P1*, …, *Pn* выбираются только те значения *R1, ..., Rm*, которые не имеют ошибки, достоверность которых равна 0 и отклонение которых от нормативного значения *Рнорм* по модулю не превосходит *днорм.*

- Если *m=0* , то есть значений *R1, …, Rm* не нашлось, то функция возвращает *Рнорм*.
- Если *m>0*, то функция возвращает $(R1 + \cdots + Rm)/m$.

Пример: Пусть в выражении используется вызов функции

ДостоверН(100;25;{A};{B};{C};{D};{E}),

где A, B, C, D и E – сигналы со следующими значениями и недостоверностями:

A		B		C		D		E	
Знач.	Недост.	Знач.	Недост.	Знач.	Недост.	Знач.	Недост.	Знач.	Недост.
90	0	93	0	155	0	120	0	-50	1

В этом случае в качестве значений *R1, R2* и *R3* будут взяты значения сигналов «A», «B» и «D». Значение сигнала «E» будет отброшено, как недостоверное. Значение сигнала «C» будет отброшено, как отклоняющееся от нормативного значения на 150-100=50, то есть больше чем на 25. Результатом функции будет достоверное значение (90+93+120)/3=101.

Функция ДостоверНП

Английское имя: CertainNP.

Назначение: Достоверный результат по набору входных значений. Вычисляется как среднее всех значений из списка, которые достоверны, отклоняются от нормативного значения не более чем на указанную величину и отклоняются от среднего между ними не больше чем на другую заданную величину. Если таких значений в списке не нашлось, то возвращается нормативное значение.

Синтаксис: **ДостоверНП(Рнорм;dнорм;dсред;P1;P2;P3;…;Pn).**

Сигнатура:

действ ДостоверНП **(действ** *Рнорм*; **действ** *dнорм*; **действ** *dсред*; **действ** *P1***; …; действ** *Pn***).**

Параметры:

P1, …, *Pn* – значения, исходя из которых вычисляется достоверное;

Рнорм – нормативное значение;

dнорм – допустимое отклонение от нормативного значения;

dсред – допустимое отклонение от среднего из входных значений.

Значение функции вычисляется следующим образом:

- Из значений *P1*, …, *Pn* выбираются только те значения *R1 ,…, Rm*, которые не имеют ошибки, достоверность которых равна 0 и отклонение которых от нормативного значения *Рнорм* по модулю не превосходит *dнорм.*
- Если *m=0,* то есть значений *R1, …, Rm* не нашлось, то функция возвращает *Рнорм*.
- Если *m>0*, то повторяется следующая процедура:
  – вычисляется среднее $M = (R1 + \cdots + Rm)/m$;
  – из списка R1, …, Rm выбрасываются значения, которые отличаются по модулю от M более чем на *dсред*;
  – если было выброшено хоть одно значение и в списке осталось хотя бы одно значение, то процедура повторяется.
- Пусть после выполнения процедуры в списке остались значения *Q1, …, Qs.*
- Если *s=0*, то есть значений *Q1, …, Qs* не нашлось, то функция возвращает *Рнорм*.
- Если *s*>0, то функция возвращает $(Q1 + \cdots + Qs)/s$.

Пример: Пусть в выражении используется вызов функции

ДостоверНП(100; 25; 15; {A};{B};{C};{D};{E}),

где A, B, C, D и E – сигналы со следующими значениями и недостоверностями:

A		B		C		D		E	
Знач.	Недост.	Знач.	Недост.	Знач.	Недост.	Знач.	Недост.	Знач.	Недост.
90	0	93	0	155	0	120	0	-50	1

В этом случае в качестве значений *R1, R2* и *R3* будут взяты значения сигналов «A», «B» и «D». Значение сигнала «E» будет отброшено, как недостоверное. Значение сигнала «C» будет отброшено, как отклоняющееся от нормативного значения на 150-100=50, то есть больше чем на 25. Среднее значение оставшихся значений M=(90+93+120)/3=101. Значение сигнала «D» будет отброшено, как отклоняющееся от среднего на 120-101=19, то есть больше чем на 15. Результатом функции будет достоверное значение (90+93)/2=91,5.

Функция ДостоверП

Английское имя: CertainP.

Назначение: Достоверный результат по набору входных значений. Вычисляется как среднее всех значений из списка, которые достоверны и отклоняются от среднего между ними не больше чем на другую заданную величину. Если таких значений в списке не нашлось, то возвращается нормативное значение.

Синтаксис: **ДостоверП(Рнорм;dcред;P1;P2;P3;…;Pn).**

Сигнатура: **действ** ДостоверНП **(действ** *Рнорм*; **действ** *dcред*; **действ** *P1***; …; действ** *Pn***).**

Параметры:

P1, …, *Pn* – значения, исходя из которых вычисляется достоверное,

Рнорм – нормативное значение;

dcред – допустимое отклонение от среднего из входных значений.

Значение функции вычисляется следующим образом:

- Из значений *P1*, …, *Pn* выбираются только те значения *R1, …, Rm*, которые не имеют ошибки, достоверность которых равна 0.
- Если *m=0,* то есть значений *R1, …, Rm* не нашлось, то функция возвращает *Рнорм*.
- Если *m>0*, то повторяется следующая процедура:
  – вычисляется среднее $M = (R1 + \cdots + Rm)/m$;
  – из списка R1, …, Rm выбрасываются значения, которые отличаются по модулю от M более чем на *dcред*;
  – если было выброшено хоть одно значение и в списке осталось хотя бы одно значение, то процедура повторяется.
- Пусть после выполнения процедуры в списке остались значения *Q1, …, Qs.*
- Если *s=0*, то есть значений *Q1, …, Qs* не нашлось, то функция возвращает *Рнорм*.
- Если *s*>0, то функция возвращает $(Q1 + \cdots + Qs)/s$.

Пример: Пусть в выражении используется вызов функции

ДостоверП(100;15;{A};{B};{C};{D};{E}),

где A, B, C, D и E – сигналы со следующими значениями и недостоверностями:

A		B		C		D		E	
Знач.	Недост.	Знач.	Недост.	Знач.	Недост.	Знач.	Недост.	Знач.	Недост.
90	0	93	0	155	0	120	0	-50	1

В этом случае в качестве значений *R1, R2, R3* и *R4* будут взяты значения сигналов «A», «B», «C» и «D». Значение сигнала «E» будет отброшено как недостоверное. Среднее значение оставшихся значений M=(90+93+155+120)/4=115. Значение сигналов «A» и «D» будут отброшены, как отклоняющееся от среднего на 115-90=25 и 155-115=40, то есть больше чем на 15. Результатом функции будет достоверное значение (120+93)/2=106,5.

4.6. Функции работы с мгновенными значениями

Данный раздел описывает группу функций «Точки». Функции этой группы не являются скалярными и служат для расширенной работы со списками мгновенных значений.

Функция Апертура

Английское имя: Aperture.

Назначение: Прореживает список исходных значений. Из исходного списка берется первое значение и затем из списка выбирается значение только в том случае, если оно отличается по модулю от предыдущего выбранного значения не меньше чем на заданную величину.

Синтаксис: **Апертура(Выражение;ВеличинаАпертуры).**

Возможные сигнатуры:

действ Апертура(действ *Выражение*; **действ** *ВеличинаАпертуры***);**

целое Апертура(целое *Выражение*; **Действ** *ВеличинаАпертуры***).**

Параметры:

Выражение – выражение, на которое устанавливается апертура;

ВеличинаАпертуры – величина, на которую должны различаться соседние мгновенные значения результата функции.

В начале периода обработки мгновенное значение функции равно мгновенному значению *Выражения*. Пусть в некоторый момент времени мгновенное значение функции было равно X. В следующие моменты времени значение функции будет по прежнему равно X, пока значение *Выражения* будет отличаться от X меньше чем на *ВеличинуАпертуры*. Как только значение *Выражения* будет отличаться от X больше чем на *ВеличинуАпертуры*, мгновенное значение функции станет равным значению выражения.

Пример: Пусть запись функции в формуле имеет вид: Апертура({S};10) и расчет производится за период с 0:00:00 до 0:15:00. В таблице ниже приведен пример мгновенных значений сигнала «S» и соответствующие мгновенные значения функции Апертура({S};10):

Время	S	Апертура({S};10)
20.11.2013 0:00:00	5	5
20.11.2013 0:02:00	9	5
20.11.2013 0:04:00	16	16
20.11.2013 0:08:00	22	16
20.11.2013 0:10:00	28	28
20.11.2013 0:12:00	13	13
20.11.2013 0:15:00	8	13

Константа БезТочек

Английское имя: WithoutPoints.

Назначение: Задает список мгновенных значений не содержащий ни одной точки.

Синтаксис: **БезТочек.**

Сигнатура: **величина** БезТочек.

Пример: БезТочек.

Функция ВремяМеждуТочками

Английское имя: TimeBetweenPoints.

Назначение: Для каждой точки из списка мгновенных значений возвращает длительность интервала от предыдущей точки в секундах.

Синтаксис: ВремяМеждуТочками**(Выражение).**

Сигнатура: **действ** ВремяМеждуТочками(**вариант** *Выражение*).

Параметр: *Выражение* – выражение, между точками которого вычисляется время.

Если значение *Выражения* представляет собой одно мгновенное значение, то возвращается одно мгновенное значение равное 0. Если значение *Выражения* представляет собой список мгновенных значений, то первое из них будет равно времени от начала периода расчета до первого мгновенного значения, если оно лежит после начала периода расчета, и 0, если оно лежит до начала периода расчета. Далее i-е мгновенное значение равно длительности интервала между i-м и i-1-м значениями *Выражения*. Недостоверность и ошибка i-ого мгновенного значения результата равны недостоверности и ошибке i-ого мгновенного значения *Выражения*.

Пример: ВремяМеждуТочками({S}).

Пусть расчет производится за период от 00:00:00 до 00:30:00, и сигнал {S} имеет в течение периода следующие мгновенные значения:

Время	Значение	Недостоверность
00:05:00	1	0
00:10:00	2	1
00:20:00	3	0

тогда результат вычисления выражения ВремяМеждуТочками({S}) будет иметь следующие мгновенные значения:

Время	Значение	Недостоверность
00:05:00	300	0
00:10:00	300	1
00:20:00	600	0

Функция ВремяТочки

Английское имя: PointTime.

Назначение: Возвращает время мгновенного значения или список времен мгновенных значений заданного выражения.

Синтаксис: **ВремяТочки(Выражение).**

Сигнатура: **время** ВремяТочки(**вариант** *Выражение*).

Параметр: *Выражение* – выражение, от которого берется время или список времен.

Если значение *Выражения* представляет собой одно мгновенное значение, то возвращается одно мгновенное значение, содержащее время исходного мгновенного значения. Если значение *Выражения* представляет собой список мгновенных значений, то результатом будет список мгновенных значений, величины которых соответствуют временам исходных мгновенных значений. Время, недостоверность и ошибка мгновенных значений результата совпадают со временем, недостоверностью и ошибкой исходных мгновенных значений.

Пример: ВремяТочки({S}).

Пусть сигнал {S} имеет в течение периода следующие мгновенные значения:

Время	Значение	Недостоверность
00:00:00	1	0
00:10:00	2	1
00:20:00	3	0

тогда результат вычисления выражения ВремяТочки({S}) будет иметь следующие мгновенные значения:

Время	Значение	Недостоверность
00:00:00	00:00:00	0
00:10:00	00:10:00	1
00:20:00	00:20:00	0

Функция ВыбратьТочки

Английское имя: SelectPoints.

Назначение: Выбирает из списка мгновенных значений выражения, удовлетворяющие заданному условию.

Синтаксис: **ВыбратьТочки(Выражение;Условие).**

Возможные сигнатуры:

логич ВыбратьТочки(**логич** *Выражение*; **логич** *Условие*);

целое ВыбратьТочки(**целое** *Выражение*; **логич** *Условие*);

действ ВыбратьТочки(**действ** *Выражение*; **логич** *Условие*);

время ВыбратьТочки(**время** *Выражение*; **логич** *Условие*);

строка ВыбратьТочки(**строка** *Выражение*; **логич** *Условие*).

Параметры:

Выражение – выражение, из значения которого выбираются точки;

Условие – логическое условие, в соответствии с которым точки выбираются.

Пример: ВыбратьТочки({S};{S}>0).

Пусть сигнал {S} имеет в течение периода следующие мгновенные значения:

Время	Значение	Недостоверность
00:00:00	1	0
00:10:00	-2	1
00:15:00	3	1
00:20:00	-4	0
00:25:00	5	0

тогда результат вычисления выражения ВыбратьТочки({S};{S}>0) будет иметь следующие мгновенные значения:

Время	Значение	Недостоверность
00:00:00	1	0
00:15:00	3	1
00:25:00	5	0

Функция ЗаписатьВремя

Английское имя: SetTime.

Назначение: Приписывает выражению указанное время.

Синтаксис: **ЗаписатьВремя(Выражение;Время).**

Возможные сигнатуры:

логич ЗаписатьВремя (**логич** *Выражение*; **время** *Время*);

целое ЗаписатьВремя (**целое** *Выражение*; **время** *Время*);

действ ЗаписатьВремя (**действ** *Выражение*; **время** *Время*);

время ЗаписатьВремя (**время** *Выражение*; **время** *Время*);

строка ЗаписатьВремя (**строка** *Выражение*; **время** *Время*).

Параметры:

Выражение – исходное выражение;

Время – время, которое приписывается выражению.

Пример: ЗаписатьВремя({S};#10.10.2013 10:10:10#).

Функция ЗначениеПоВремени

Английское имя: ValueAtTime.

Назначение: Возвращает значение выражения в указанный момент времени или список мгновенных значений на величины мгновенных значений–времен, заданных списком значений.

Синтаксис: **ЗначениеПоВремени(Выражение;Время).**

Возможные сигнатуры:

логич ЗначениеПоВремени (**логич** *Выражение*; **время** *Время*);

целое ЗначениеПоВремени (**целое** *Выражение*; **время** *Время*);

действ ЗначениеПоВремени (**действ** *Выражение*; **время** *Время*);

время ЗначениеПоВремени (**время** *Выражение*; **время** *Время*);

строка ЗначениеПоВремени (**строка** *Выражение*; **время** *Время*).

Параметры:

Выражение – выражение, значение которого возвращается на моменты времени;

Время – моменты время, в которые возвращается значение выражения.

Пример: ЗначениеПоВремени({A};#10.10.2013 10:10:10#).

Функция ЗначениеПоТочкам

Английское имя: ValueAtPoints.

Назначение: Возвращает значения (величину, недостоверность и ошибку) одного выражения на моменты мгновенных значений второго. Величина мгновенных значений результата вычисляется в соответствии с типом интерполяции расчетного параметра. Результат имеет тип данных первого выражения.

Синтаксис: **ЗначениеПоТочкам(Выражение1;Выражение2).**

Возможные сигнатуры:

логич ЗначениеПоТочкам (**логич** *Выражение1*; **вариант** *Выражение2*);

целое ЗначениеПоТочкам (**целое** *Выражение1*; **вариант** *Выражение2*);

действ ЗначениеПоТочкам (**действ** *Выражение1*; **вариант** *Выражение2*);

время ЗначениеПоТочкам (**время** *Выражение1*; **вариант** *Выражение2*);

строка ЗначениеПоТочкам (**строка** *Выражение1*; **вариант** *Выражение2*).

Параметр: *Выражение1* – выражение, значение которого вычисляется в моменты *Выражения2*.

Пример: Пусть в проекте есть сигналы {S1}, {S2} и расчетный параметр «A» с формулой ЗначениеПоТочкам({S1};{S2}) и ступенчатым типом интерполяции. Пусть расчет производится за период с 0:00:00 до 1:00:00. В следующей таблице приведен пример списка мгновенных значений сигналов {S1}, {S2} и параметра «A».

Время	{S1}	{S2}	A
0:00:00	1	1	1
0:05:00		0	1
0:10:00	2		
0:15:00		1	2
0:20:00		0	2
0:25:00	3		
0:30:00	4	1	4
0:35:00		0	4
0:40:00	5		
0:45:00	6	1	6
0:50:00	7		
0:55:00		0	7

Функция Зона

Английское имя: Zone.

Назначение: Формирует условие принадлежности определенной временной зоне. Зона представляет собой логическое условие, которое может быть истинным на нескольких интервалах внутри периода обработки. Период расчета просматривается, начиная с начала. Каждый интервал истинности зоны начинается с момента, когда заданное условие начала меняет свое значение с 0 на 1. Интервал истинности оканчивается в зависимости от входных параметров одним из следующих способов:

- Если задано только условие конца, то интервал истинности зоны оканчивается в момент изменения условия конца с 0 на 1.
- Если задана только длительность интервала зоны, то каждый интервал истинности зоны имеет эту длительность.
- Если заданы условие конец и длительность, то интервал истинности зоны заканчивается в момент начала выполнения условия конца или после указанной длительности интервала, если условие конца так и не выполнилось.

После завершения интервала истинности зоны выполняется поиск следующего перехода условия начала из 0 в 1, даже если в конце интервала зоны значение условия начала равно 1. И так далее до конца периода расчета.

Недостоверность и ошибка всех точек формируется как максимум недостоверностей и ошибок входных параметров в соответствующих точках.

Синтаксис: **Зона(УсловиеНачала;[УсловиеКонца];[Длительность]).**

Возможные сигнатуры:

логич Зона(**логич** *УсловиеНачала*; **логич** *УсловиеКонца*);

логич Зона(**логич** *УсловиеНачала*; **действ** *Длительность*);

логич Зона(**логич** *УсловиеНачала*; **логич** *УсловиеКонца*; **действ** *Длительность*).

Параметры:

УсловиеНачала – условие, при переключении которого из 0 в 1 начинается очередной интервал зоны;

УсловиеКонца – условие, при переключении которого из 0 в 1 оканчивается очередной интервал зоны;

Длительность – максимальная длительность интервалов истинности зоны.

Пример: Пусть в проекте есть сигналы {S1}, {S2} и три расчетных параметра «A», «B» и «C» со следующими формулами:

Параметр	Формула
A	Зона({S1}>0; {S2}>0)
B	Зона({S1}>0; 900)
C	Зона({S1}>0; {S2}>0; 900)

Пусть расчет производится за период с 0:00:00 до 1:00:00. В следующей таблице приведен пример списка мгновенных значений сигналов {S1}, {S2} и вычисленных мгновенных значений параметров «A», «B» и «C».

Время	{S1}	{S2}	A	B	C
0:00:00	-1	-1	0	0	0
0:05:00	1		1	1	1
0:15:00	-1	1	0		0
0:20:00				0	
0:21:00	1		1	1	1
0:25:00		-1			
0:36:00				0	0
0:37:00	-1	1	0		
0:40:00	1		1	1	1
0:55:00				0	0

Функция ЗонаСКонца

Английское имя: ZoneFromEnd.

Назначение: Формирует условие принадлежности определенной временной зоне. Зона представляет собой логическое условие, которое может быть истинным на нескольких интервалах внутри периода обработки. Период расчета просматривается с конца. Каждый интервал истинности зоны оканчивается в момент, когда заданное условие конца меняет свое значение с 0 на 1. Интервал истинности начинается в зависимости от входных параметров одним из следующих способов:

- Если задано только условие начала, то интервал истинности зоны начинается в момент изменения условия начала с 0 на 1.
- Если задана только длительность интервала зоны, то каждый интервал истинности зоны имеет эту длительность.
- Если заданы условие начала и длительность, то интервал истинности зоны начинается в момент начала выполнения условия начала, или после указанной длительности интервала, если условие начала так и не выполнилось (если двигаться с конца).

После начала интервала истинности зоны выполняется поиск следующего перехода условия конца из 0 в 1, даже если в начале интервала зоны значение условия конца равно 1. И так далее до начала периода расчета.

Недостоверность и ошибка всех точек формируется как максимум недостоверностей и ошибок входных параметров в соответствующих точках.

Синтаксис: **ЗонаСКонца([УсловиеНачала];УсловиеКонца;[Длительность]).**

Возможные сигнатуры:

логич ЗонаСКонца(**логич** *УсловиеНачала*; **логич** *УсловиеКонца*);

логич ЗонаСКонца(**логич** *УсловиеКонца*; **дейст** *Длительность*);

логич ЗонаСКонца(**логич** *УсловиеНачала*; **логич** *УсловиеКонца*; **дейст** *Длительность*).

Параметры:

УсловиеНачала – условие, при переключении которого из 0 в 1 начинается очередной интервал зоны;

УсловиеКонца – условие, при переключении которого из 0 в 1 оканчивается очередной интервал зоны;

Длительность – максимальная длительность интервалов истинности зоны.

Пример: Пусть в проекте есть сигналы {S1}, {S2} и три расчетных параметра «A», «B» и «C» со следующими формулами:

Параметр	Формула
A	ЗонаСКонца({S1}>0; {S2}>0)
B	ЗонаСКонца({S2}>0; 900)
C	ЗонаСКонца({S1}>0; {S2}>0; 900)

Пусть расчет производится за период с 0:00:00 до 1:00:00. В следующей таблице приведен пример списка мгновенных значений сигналов {S1}, {S2} и вычисленных мгновенных значений параметров «A», «B» и «C».

Время	{S1}	{S2}	A	B	C
0:00:00	-1		0	0	0
0:05:00	1		1	1	1
0:15:00		-1			
0:20:00		1	0	0	0
0:21:00					
0:25:00	-1			1	
0:30:00	1		1		1
0:37:00		-1			
0:40:00		1		0	0
0:45:00				1	1
0:55:00		-1			
1:00:00		1	0	0	0

Функция МаксЗона

Английское имя: MaxZone.

Пример: Возвращает условие, задающее самый длинный из непрерывных интервалов выполнения исходного условия. Интервал может лежать как посередине, так и в начале или конце периода обработки. Остальной части периода расчета результирующее значение присваивается равным 0. Недостоверность и ошибка всех точек переносится из исходного выражения.

Синтаксис: **МаксЗона(Условие).**

Сигнатура: **логич** МаксЗона(**логич** *Условие*).

Параметр: *Условие* – условие, из которого выбирается максимальный интервал непрерывного выполнения.

Пример: Пусть в проекте есть сигнал {S} и три расчетных параметра «A» и «B» со следующими формулами:

Параметр	**Формула**
A	МаксЗона({S}>0)
B	МаксЗона({S}<0)

Пусть расчет производится за период с 0:00:00 до 1:00:00. В следующей таблице приведен пример списка мгновенных значений сигнала {S} и вычисленных мгновенных значений параметров «A» и «B». Для параметра «A» максимальный интервал выполнения условия – от 0:10:00 до 0:30:00, для параметра «B» – от 0:30:00 до конца периода расчета.

Время	{S}	A	B
0:00:00	-1	0	0
0:10:00	2	1	
0:20:00	3		
0:30:00	-2	0	1
0:50:00	-4		

Функция НарастающийИтог

Английское имя: Accumulation.

Назначение: Для каждого мгновенного значения входного выражения вычисляет нарастающий итог как интеграл от параметра с начала периода расчета до времени этого значения. Кроме того, в конец результирующего списка мгновенных значений добавляется значения интеграла за весь период со временем, равным концу периода расчета.

Интеграл вычисляется методом левых прямоугольников, если для параметра задана ступенчатая интерполяция, и методом трапеций, если для параметра задана линейная интерполяция. Каждому мгновенному значению из результата присваивается недостоверность и ошибка равными максимуму недостоверностей и ошибок мгновенных значений, по которым мгновенное значение вычисляется.

Синтаксис: **НарастающийИтог(Выражение).**

Сигнатура: **действ** НарастающийИтог(**действ** *Выражение*).

Параметры: *Выражение* – выражение, от которого вычисляется нарастающий итог.

Пример: Пусть в проекте есть сигнал {S} и расчетные параметры «A» и «B» со следующими формулами и типами интерполяции:

Параметр	Формула	Интерполяция
A	НарастающийИтог({S})	Ступенчатая
B	НарастающийИтог({S})	Линейная

Пусть расчет производится за период с 0:00:00 до 1:00:00. В следующей таблице приведен пример списка мгновенных значений сигнала {S} и вычисленных мгновенных значений параметров «A» и «B». Также указаны недостоверности всех значений.

Время	{S}	Недост({S})	A	Недост(A)	B	Недост(B)
0:00:00	1	0	0	0	0	0
0:10:00	2	0	600	0	900	0
0:20:00	1	0	1800	0	1800	0
0:30:00	2	0	2400	0	2700	0
0:40:00	1	1	3600	0	3600	1
0:50:00	2	0	4200	1	4500	1
1:00:00			5400	1	5700	1

Функция НомерТочки

Английское имя: PointNumber.

Назначение: Для каждого мгновенного значения заданного выражения возвращает мгновенное значение с тем же временем и величиной, равной номеру исходного мгновенного значения в исходном списке. Значения нумеруются, начиная с 1. Недостоверность и ошибка мгновенного значения результата равны недостоверности и ошибке исходного мгновенного значения.

Синтаксис: **НомерТочки(Выражение).**

Сигнатура: **целое** НомерТочки(**вариант** *Выражение*).

Параметры: *Выражение* – список мгновенных значений, от которых берутся номера.

Пример: Пусть в проекте есть сигнал {S} и расчетный параметр «A» с формулой: НомерТочки({S}). Пусть расчет производится за период с 0:00:00 до 1:00:00. В таблице ниже приведены некоторые значения сигнала {S} и соответствующие значения параметра «A».

Время	{S}	A
0:00:00	0,23	1
0:10:00	4,34	2
0:20:00	1,35	3
0:30:00	-4,21	4
0:40:00	6,34	5
0:50:00	-1,35	6

Функция Объединить

Английское имя: Union.

Назначение: Объединяет мгновенные значения нескольких выражений в один список. Значения попадают в итоговый список вместе с недостоверностью, ошибкой и временем. В результирующем списке значения сортируются по времени. Если времена значений повторяются, то в итоговый список попадает только одно, из первого входного параметра (по порядку вхождения в формулу).

Синтаксис: **Объединить(Выражение1;Выражение2;…).**

Сигнатура: **вариант** Объединить (**вариант** *Выражение1*; **вариант** *Выражение2*; …).

Параметры: *Выражение1, Выражение2* – объединяемые списки мгновенных значений.

Пример: Пусть в проекте есть сигналы {S1}, {S2} и {S3} и расчетный параметр A с формулой Объединить({S1};{S2};{S3}. Пусть расчет производится за период с 0:00:00

до 1:00:00. В следующей таблице приведен пример списка мгновенных значений сигналов {S1}, {S2}, {S3} и вычисленных мгновенных значений параметра «A».

Время	{S1}	{S2}	{S3}	A
0:00:00	1		8	1
0:05:00		5		5
0:10:00	2			2
0:25:00			9	9
0:35:00	3	6		3
0:45:00	4	7	10	4
0:55:00			11	11

Функция РавномерныеТочки

Английское имя: UniformPoints.

Назначение: Возвращает значения заданного выражения, полученные через равные промежутки времени. Значения, недостоверность и ошибка формируются в соответствии с типом интерполяции расчетного параметра по одному или двум значениям исходного выражения.

Синтаксис: **РавномерныеТочки(Выражение;Длина;[Сдвиг]).**

Возможные сигнатуры:

логич РавномерныеТочки (**логич** *Выражение*; **действ** *Длина*; [**действ** *Сдвиг*]);

целое РавномерныеТочки (**целое** *Выражение*; **действ** *Длина*; [**действ** *Сдвиг*]);

действ РавномерныеТочки (**действ** *Выражение*; **действ** *Длина*; [**действ** *Сдвиг*]);

время РавномерныеТочки (**время** *Выражение*; **действ** *Длина*; [**действ** *Сдвиг*]);

строка РавномерныеТочки (**строка** *Выражение*; **действ** *Длина*; [**действ** *Сдвиг*]).

Параметры:

Выражение – исходное значения, по которому формируются равномерно распределенные точки;

Длина – длина интервала между последовательными точками в секундах;

Сдвиг – число от 0 до 1, определяет время, на которое будет сдвинуто значение, относительно начала каждого интервала. 0 соответствует началу, а 1 – концу интервала.

Пример: Есть условие, при переключении которого из 0 в 1 начинается очередной интервал зоны, и типы интерполяции:

Параметр	Формула	Интерполяция
A	РавномерныеТочки({S};10;0)	Ступенчатая
B	РавномерныеТочки({S};20;0,5)	Ступенчатая
C	РавномерныеТочки({S};15;1)	Линейная

Пусть расчет производится за период с 0:00:00 до 1:00:00. В таблице ниже приведены некоторые значения сигнала {S} и соответствующие значения параметров «A», «B» и «C».

Время	{S}	A	B	C
0:00:00	1	1		
0:05:00	2			
0:10:00		2	2	
0:15:00				4
0:20:00		2		
0:25:00	6			

Время	{S}	A	B	C
0:30:00	4	4	4	4
0:35:00	5			
0:40:00		5		
0:45:00				6
0:50:00	8	8	8	
1:00:00				8

Функция СдвигПоВремени

Английское имя: MoveByTime.

Назначение: Возвращает список мгновенных значений, сдвинутых на заданный в секундах промежуток времени от мгновенных значений заданного выражения.

Синтаксис: **СдвигПоВремени(Выражение;Сдвиг).**

Возможные сигнатуры:

логич СдвигПоВремени (**логич** *Выражение*; **действ** *Сдвиг*);

целое СдвигПоВремени (**целое** *Выражение*; **действ** *Сдвиг*);

действ СдвигПоВремени (**действ** *Выражение*; **действ** *Сдвиг*);

время СдвигПоВремени (**время** *Выражение*; **действ** *Сдвиг*);

строка СдвигПоВремени (**строка** *Выражение*; **действ** *Сдвиг*).

Параметры:

Выражение – выражение, мгновенные значения которого сдвигаются;

Сдвиг – время в секундах, на которое сдвигаются мгновенные значения. Значения сдвигаются вперед по ходу времени.

Пример: Пусть в проекте есть сигнал {S} и параметры «A» и «B» со следующими формулами:

Параметр	Формула
A	СдвигПоВремени({S};900)
B	СдвигПоВремени({S};-600)

Пусть расчет производится за период с 1:00:00 до 2:00:00. В следующей таблице приведен пример списка мгновенных значений сигнала {S} и вычисленных мгновенных значений параметров «A» и «B».

Время	{S}	A	B
0:50:00			1
1:00:00	1		1
1:10:00	2		2
1:15:00		2	
1:20:00	3		3
1:25:00		3	
1:30:00	4		4
1:35:00		4	
1:40:00	5		5
1:45:00		5	
1:50:00	6		
1:55:00		6	
2:05:00		6	

Функция СдвигПоНомеру

Английское имя: MoveByNumber.

Назначение: Получить список мгновенных значений, сдвинутых на заданное количество номеров в списке мгновенных значений заданного выражения. Если значения сдвигаются на *n* номеров вперед, то первые *n* значений выкидываются, а мгновенному значению с номером *i* приписывается величина и недостоверность мгновенного значения с номером *n+i* и время значения с номером *i*. Если значения сдвигаются на *n* номеров назад, то последние *n* значений выкидываются, а мгновенному значению с номером *i* приписывается величина и недостоверность мгновенного значения с номером *i-n* и время значения с номером *i*.

Синтаксис: **СдвигПоНомеру(Выражение;Сдвиг)**.

Возможные сигнатуры:

логич СдвигПоНомеру(**логич** *Выражение*; **целое** *Сдвиг*);

целое СдвигПоНомеру(**целое** *Выражение*; **целое** *Сдвиг*);

действ СдвигПоНомеру(**действ** *Выражение*; **целое** *Сдвиг*);

время СдвигПоНомеру(**время** *Выражение*; **целое** *Сдвиг*);

строка СдвигПоНомеру(**строка** *Выражение*; **целое** *Сдвиг*).

Параметры:

Выражение – выражение, мгновенные значения сдвигаются;

Сдвиг – на сколько номеров сдвигаются мгновенные значения.

Пример: Пусть в проекте есть сигнал {S} и параметры A и B со следующими формулами:

Параметр	Формула
A	СдвигПоНомеру({S};3)
B	СдвигПоНомеру({S};-2)

Пусть расчет производится за период с 0:00:00 до 1:00:00. В следующей таблице приведен пример списка мгновенных значений сигнала {S} и вычисленных мгновенных значений параметров «A» и «B».

Время	{S}	A	B
0:00:00	1	4	
0:10:00	2	5	
0:20:00	3	6	1
0:30:00	4		2
0:40:00	5		3
0:50:00	6		4

Функция Скорость

Английское имя: Speed.

Назначение: Вычисление скорости изменения выражения. Скорость измеряется в единицах измерения выражения в секунду.

Синтаксис: **Скорость(Выражение;[Количество];[Положение]).**

Сигнатура: **действ** Скорость(**действ** *Выражение*; **целое** *Количество***; действ** *Положение***).**

Параметры:

Выражение – выражение, скорость изменения которого вычисляется;

Количество – количество последовательных точек *Выражения*, по которым вычисляется каждое мгновенное значение скорости. Необязательный параметр, по умолчанию равен 8;

Положение – число от 0 до 1, задающее положение времени мгновенного значения скорости в отрезке между первой и последней из точек, по которым оно считается. Необязательный параметр, по умолчанию равен 1.

Для вычисления мгновенного значения скорости используется метод наименьших квадратов, при помощи которого строится значение линейной функции по количеству точек равному параметру *Количество*.

Пусть N =*Количестово,* $T_1, \dots, T_N$ – моменты времени мгновенных значений *Выражения* , $X_1, \dots, X_N$ – мгновенные значения *Выражения*. Пусть

$$MT = \frac{T_1 + \cdots + T_N}{N}, MX = \frac{(X_1 + \cdots + X_N)}{N}.$$

Тогда скорость V по мгновенным значениям $X_1, \dots, X_N$ вычисляется по формуле:

$$V = \frac{((X_1 - MX)(T_1 - MT) + \cdots + (X_N - MX)(T_N - MT)}{((T_1 - MT)(T_1 - MT) + \cdots + (T_N - MT)(T_N - MT))}.$$

Пусть P=*Положение*, тогда время мгновенного значения скорости V будет вычисляться по формуле:

$$T = T_1 + (T_N - T_1) * P.$$

Недостоверность (ошибка) значения функции равна максимуму (ошибок) недостоверностей значений $X_1, \dots, X_N$.

Если для выражения есть длинные промежутки времение без значенний, то в значения вычисленной скорости будут добавлены дополнительные промежуточные точки, повторяющие предыдущие значения.

Пример: Пусть выражение имеет вид:

Скорость({A};4;1), где A – код некоторого сигнала.

Выражение вычисляется за период с 0:00:00 до 0:15:00. В таблице ниже приведен пример мгновенных значений сигнала «A» и соответствующие значения функции «Скорость».

Время	A	Скорость([A];4;1)
0:00:00	0	
0:03:00	180	
0:06:00	360	
0:09:00	540	1
0:12:00	0	-12
0:15:00	60	-48

Функция Событие

Английское имя: Event.

Назначение: Возвращает значения выражения на моменты возникновения событий, то есть на моменты, когда заданное условие меняется с 0 на 1. В результат также можно добавить значения другого выражения на моменты завершения событий, то есть, когда условие меняется с 1 на 0.

Синтаксис: **Событие(Выражение1;[Выражение2];Условие).**

Возможные сигнатуры:

вариант Событие(**вариант** *Выражение1*; **логич** *Условие*);

вариант Событие(**вариант** *Выражение1*; **вариант** *Выражение2*; **логич** *Условие*).

Параметры:

Выражение1 – выражение, значения которого на моменты изменения условия с 0 на 1 добавляются в результат;

Выражение2 – выражение, значения которого на моменты изменения условия с 1 на 0 добавляются в результат;

Условие – логическое условие, изменение значения которого задает моменты событий.

Недостоверность и ошибка в моменты начала события вычисляется как максимум недостоверностей и ошибок Выражения1 и условия, в моменты окончания события как максимумы недостоверностей и ошибок обоих выражений и условия.

Пример: Пусть в проекте есть сигналы {S1}, {S2} и {S3} и расчетные параметры «A» и «B» со следующими формулами:

Параметр	Формула
A	Событие({S1};{S3}>0)
B	Событие({S1};{S2};{S3}>0)

Пусть расчет производится за период с 0:00:00 до 1:00:00. В следующей таблице приведен пример списка мгновенных значений сигналов {S1}, {S2}, {S3} и вычисленных мгновенных значений параметров «A» и «B».

Время	{S1}	{S2}	{S3}	A	B
0:00:00	1	5	-1		
0:10:00	2	5	2	2	2
0:20:00	3	5	-2		5
0:30:00	4	5	1	4	4
0:40:00	5	5	-2		5
0:50:00	6	5	-3		

Функция Точка

Английское имя: Point.

Назначение: Возвращает одно мгновенное значение с указанным значением, временем и недостоверностью.

Синтаксис: **Точка(Значение;Время;[Недостоверность]).**

Возможные сигнатуры:

логич ЗаписатьВремя (**логич** *Значение*; **время** *Время;* **целое** *Недостоверность*);

целое ЗаписатьВремя (**целое** *Значение*; **время** *Время;* **целое** *Недостоверность*);

действ ЗаписатьВремя (**действ** *Значение*; **время** *Время;* **целое** *Недостоверность*);

время ЗаписатьВремя (**время** *Значение*; **время** *Время;* **целое** *Недостоверность*);

строка ЗаписатьВремя (**строка** *Значение*; **время** *Время;* **целое** *Недостоверность*).

Параметры:

Значение – значение точки;

Время – время точки;

Недостоверность – недостоверность точки.

Пример: Точка(24;#10.10.2013 10:10:10#).

Функция УдалитьПовторы

Английское имя: DeleteRepetitions

Назначение: Удаляет из списка мгновенных значений все значения, величина, недостоверность и ошибка которых полностью повторяет величину, недостоверность и ошибку предыдущего значения.

Синтаксис: **УдалитьПовторы(Выражение;[УбиратьСписок]).**

Возможные сигнатуры:

логич УдалитьПовторы(**логич** *Выражение*; **логич** *УбратьСписок*);

целое УдалитьПовторы(**целое** *Выражение*; **логич** *УбратьСписок*);

действ УдалитьПовторы(**действ** *Выражение*; **логич** *УбратьСписок*);

время УдалитьПовторы(**время** *Выражение*; **логич** *УбратьСписок*);

строка УдалитьПовторы(**строка** *Выражение*; **логич** *УбратьСписок*).

Параметры:

Выражение – список мгновенных значений, из которого удаляются повторы;

УбратьСписок – если равно «True», то в случае, если после выкидывания значений в списке остается всего одно, то список превращается в константу.

Пример: Пусть в проекте есть сигнал {S} и параметр «A» с формулой УдалитьПовторы({S}). Пусть расчет производится за период с 0:00:00 до 1:00:00. В следующей таблице приведен пример списка мгновенных значений сигнала {S} и вычисленных значений параметра «A».

Время	{S}	A
0:00:00	1	1
0:10:25	1	
0:25:34	2	2
0:30:02	3	3
0:42:58	3	
0:57:11	4	4

4.7. Статистические функции

Общее описание

Данный раздел описывает группу функций «Статистические». Статистические функции не являются скалярными. Каждая статистическая функция может вычисляться двумя способами:

- Если функция просто вызывается от списка мгновенных значений (например, от сигнала) без указания сегментов, то итоговое значение вычисляется за весь период расчета и является единственным мгновенным значением результата.
- Если функция вызывается от списка мгновенных значений и списка сегментов, то на каждом из сегментов вычисляется свое итоговое значение, и результат представляет собой список мгновенных значений, полученных на разных сегментах. Описание функций формирования списка сегментов в разделе 5.7.

Для каждой статистической функции в описании приведен пример вызова без указания сегментов, а также с указанием сегментов как результата вычисления функций РавномерныеСегменты и Сегменты.

Функция Время

Английское имя: Time.

Назначение: Возвращает продолжительность времени выполнения заданного условия за период обработки или продолжительность времени выполнения условия на каждом из списка указанных сегментов. Результат возвращается в секундах.

Синтаксис: **Время(Условие;[Сегменты]).**

Возможные сигнатуры:

действ Время(**логич** *Условие*);

действ Время(**логич** *Условие*; **сегменты** *Сегменты*).

Параметры:

Условие – условие, время выполнения которого вычисляется;

Сегменты – набор сегментов, на каждом из которых производится вычисление результата функции.

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным началу периода. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени начала сегмента. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «Время». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00. Значения функции измеряются в секундах. В скобках указаны периоды выполнения условия, время выполнения которого вычисляется.

Время	A	Время(A>2)	Время(A>2; РавномерныеСегменты(300))	Время(A>2; Сегменты(A>1))
0:00:00	4	540 (00:00-04:00, 08:00-11:00, 12:00-14:00)	240 (00:00-04:00)	240 (00:00-04:00)
0:02:00	6			
0:04:00	0			
0:05:00			120 (08:00-10:00)	180 (08:00-11:00)
0:06:00	2			
0:08:00	3			
0:10:00			180 (10:00-11:00, 12:00-14:00)	
0:11:00	0			
0:12:00	3			120 (12:00-14:00)
0:14:00	1			
0:15:00				

Функция Дисперсия

Английское имя: Dispersion.

Назначение: Возвращает интегральную дисперсию заданного выражения за период расчета или список интегральных дисперсий по указанному списку сегментов. Результат возвращается в единицах измерения исходного значения в квадрате.

Синтаксис: **Среднее(Выражение;[Сегменты]).**

Возможные сигнатуры:

действ Дисперсия(**действ** *Выражение*);

действ Дисперсия(**действ** *Выражение*; **сегменты** *Сегменты*).

Параметры:

Выражение – выражение, от которого вычисляется дисперсия;

Сегменты – набор сегментов, на каждом производится вычисление результата функции.

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным началу периода. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если для выражения, в котором используется функция, задана ступенчатая интерполяция, то интегралы вычисляются методом левых прямоугольников, если задана линейная интерполяция, то интегралы вычисляются методом трапеций.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени начала сегмента. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «Дисперсия». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00. Интерполяция для выражения задана ступенчатая.

Время	A	Дисперсия (A)	Дисперсия(A; РавномерныеСегменты(300))	Дисперсия(A; Сегменты(A>1))
0:00:00	4	3,73333333 среднее 3, (1*1*2+3*3*2+ 3*3*2+1*1*2+ 0*0*3+2*2*1+ 2*2*2+2*2*1) /15	4,8 среднее 4, (0*0*2+2*2*2+4*4*1)/5	4 среднее 5, 2*2*4/4
0:02:00	6			
0:04:00	0			
0:05:00			1,2 среднее 2 (2*2*1+0*0*2+1*1*2)/5	
0:06:00	2			0,24 среднее 2,6 (0,6*0,6*2+0,4*0,4*3)/5
0:08:00	3			
0:10:00			3,2 среднее 3 (0*0*1+2*2*1+2*2*2+2*2*1)/ 5	
0:11:00	1			
0:12:00	5			0
0:14:00	1			
0:15:00				

Функция ЗначениеКонец

Английское имя: ValueEnd.

Назначение: Возвращает значение заданного выражения на момент конца периода расчета или на момент конца каждого из указанных сегментов.

Синтаксис: **ЗначениеКонец(Выражение;[Сегменты]).**

Возможные сигнатуры:

логич ЗначениеКонец(**логич** *Выражение*);

логич ЗначениеКонец(**логич** *Выражение*; **сегменты** *Сегменты*);

целое ЗначениеКонец(**целое** *Выражение*);

целое ЗначениеКонец(**целое** *Выражение*; **сегменты** *Сегменты*);

действ ЗначениеКонец(**действ** *Выражение*);

действ ЗначениеКонец(**действ** *Выражение*; **сегменты** *Сегменты*);

строка ЗначениеКонец(**строка** *Выражение*);

строка ЗначениеКонец(**строка** *Выражение*; **сегменты** *Сегменты*).

Параметры:

Выражение – выражение, конечное значение которого вычисляется;

Сегменты – набор сегментов, на каждом производится вычисление результата функции:

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным концу периода. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени конца сегмента. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «ЗначениеНачало». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «A», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00.

Время	А	Значение Конец(А)	ЗначениеКонец(А; РавномерныеСегменты(300))	ЗначениеКонец(А; Сегменты(А>1))
0:00:00	4			
0:02:00	6			0
0:04:00	0		0	
0:05:00				
0:06:00	2			
0:08:00	3		3	
0:10:00				1
0:11:00	1			
0:12:00	5			
0:13:00	5			1
0:14:00	1			
0:15:00		1	1	

Функция ЗначениеНачало

Английское имя: ValueBegin.

Назначение: Возвращает значение заданного выражения на момент начала периода расчета или на момент начала каждого из указанных сегментов.

Синтаксис: **ЗначениеНачало(Выражение[;Сегменты]).**

Возможные сигнатуры:

логич ЗначениеНачало(**логич** *Выражение*);

логич ЗначениеНачало(**логич** *Выражение*; **сегменты** *Сегменты*);

целое ЗначениеНачало(**целое** *Выражение*);

целое ЗначениеНачало(**целое** *Выражение*; **сегменты** *Сегменты*);

действ ЗначениеНачало(**действ** *Выражение*);

действ ЗначениеНачало(**действ** *Выражение*; **сегменты** *Сегменты*);

строка ЗначениеНачало(**строка** *Выражение*);

строка ЗначениеНачало(**строка** *Выражение*; **сегменты** *Сегменты*).

Параметры:

Выражение – выражение, начальное значение которого вычисляется;

Сегменты – набор сегментов, на каждом производится вычисление результата функции.

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным началу периода. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени начала сегмента. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «ЗначениеНачало». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного

параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00.

Время	А	Значение Начало(А)	ЗначениеНачало(А; РавномерныеСегменты(300))	ЗначениеНачало(А; Сегменты(А>1))
0:00:00	4	4	4	4
0:02:00	6			
0:04:00	0			
0:05:00			0	
0:06:00	2			2
0:08:00	3			
0:10:00			3	
0:11:00	1			
0:12:00	5			5
0:13:00	5			
0:14:00	1			
0:15:00				

Функция ЗначениеПоНомеру

Английское имя: ValueAtNumber.

Назначение: Возвращает мгновенное значение выражения с заданным номером, если отсчитывать от начала периода расчета, или список мгновенных значений, если отсчитывать от начала каждого из указанных сегментов. Значения нумеруются с 1.

Синтаксис: **ЗначениеПоНомеру(Номер;Выражение[;Сегменты]).**

Возможные сигнатуры:

логич ЗначениеПоНомеру(**целое** *Номер*; **логич** *Выражение*);

логич ЗначениеПоНомеру(**целое** *Номер*; **логич** *Выражение*; **сегменты** *Сегменты*);

целое ЗначениеПоНомеру(**целое** *Номер*; **целое** *Выражение*);

целое ЗначениеПоНомеру(**целое** *Номер*; **целое** *Выражение*; **сегменты** *Сегменты*);

действ ЗначениеПоНомеру(**целое** *Номер*; **действ** *Выражение*);

действ ЗначениеПоНомеру(**целое** *Номер*; **действ** *Выражение*; **сегменты** *Сегменты*);

строка ЗначениеПоНомеру(**целое** *Номер*; **строка** *Выражение*);

строка ЗначениеПоНомеру(**целое** *Номер*; **строка** *Выражение*; **сегменты** *Сегменты*).

Параметры:

Выражение – выражение, от которого берется мгновенное значение с указанным номером;

Номер – указанный номер мгновенного значения;

Сегменты – набор сегментов, на каждом производится вычисление результата функции.

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным времени выбранного значения. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени

выбранного значения внутри этого сегмента. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «ЗначениеПоНомеру». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00.

<table>
<tr><th>Время</th><th>А</th><th>ЗначениеПоНомеру(2;А)</th><th>ЗначениеПоНомеру(2;А; РавномерныеСегменты(300))</th><th>ЗначениеПоНомеру (2;А;Сегменты(А>1))</th></tr>
<tr><td>0:00:00</td><td>4</td><td rowspan="12">6</td><td rowspan="3">6</td><td rowspan="2">6</td></tr>
<tr><td>0:02:00</td><td>6</td></tr>
<tr><td>0:04:00</td><td>0</td><td rowspan="2"></td></tr>
<tr><td>0:05:00</td><td></td><td rowspan="3">3</td></tr>
<tr><td>0:06:00</td><td>2</td><td rowspan="2">3</td></tr>
<tr><td>0:08:00</td><td>3</td></tr>
<tr><td>0:10:00</td><td></td><td rowspan="5">5</td><td></td></tr>
<tr><td>0:11:00</td><td>1</td><td rowspan="2">5</td></tr>
<tr><td>0:12:00</td><td>5</td></tr>
<tr><td>0:13:00</td><td>5</td><td rowspan="2"></td></tr>
<tr><td>0:14:00</td><td>1</td></tr>
<tr><td>0:15:00</td><td></td><td></td><td></td></tr>
</table>

Функция ЗначениеПоНомеруСКонца

Английское имя: ValueAtNumberEnd.

Назначение: Возвращает мгновенное значение выражения с заданным номером, если отсчитывать с конца периода расчета, или список мгновенных значений, если отсчитывать с конца каждого из указанных сегментов. Значения нумеруются с 1, если считать с конца. Если время мгновенного значения совпадает с концом интервала, то отсчет ведется от него.

Синтаксис: **ЗначениеПоНомеруСКонца(Номер;Выражение[;Сегменты]).**

Возможные сигнатуры:

логич ЗначениеПоНомеруСКонца(**целое** *Номер*; **логич** *Выражение*);

логич ЗначениеПоНомеруСКонца(**целое** *Номер*; **логич** *Выражение*; **сегменты** *Сегменты*);

целое ЗначениеПоНомеруСКонца(**целое** *Номер*;**целое** *Выражение*);

целое ЗначениеПоНомеруСКонца(**целое** *Номер*;**целое** *Выражение*; **сегменты** *Сегменты*);

действ ЗначениеПоНомеруСКонца(**целое** *Номер*;**действ** *Выражение*);

действ ЗначениеПоНомеруСКонца(**целое** *Номер*;**действ** *Выражение*;**сегменты** *Сегменты*);

строка ЗначениеПоНомеруСКонца(**целое** *Номер*;**строка** *Выражение*);

строка ЗначениеПоНомеруСКонца(**целое***Номер*;**строка***Выражение*;**сегменты** *Сегменты*).

Параметры:

Выражение – выражение, от которого берется мгновенное значение с указанным номером;

Номер – указанный номер мгновенного значения с конца;

Сегменты – набор сегментов, на каждом производится вычисление результата функции:

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным времени выбранного значения. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени выбранного значения внутри этого сегмента. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «ЗначениеПоНомеруСКонца». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00.

<table>
<tr><th>Время</th><th>А</th><th>ЗначениеПо НомеруСКонца (2;А)</th><th>ЗначениеПоНомеруСКонца (2;А;РавномерныеСегменты (300))</th><th>ЗначениеПоНомеруС Конца (2;А;Сегменты(А>1))</th></tr>
<tr><td>0:00:00</td><td>4</td><td rowspan="13">5</td><td rowspan="3">6</td><td>4</td></tr>
<tr><td>0:02:00</td><td>6</td><td rowspan="3"></td></tr>
<tr><td>0:04:00</td><td>0</td></tr>
<tr><td>0:05:00</td><td></td><td rowspan="3">2</td></tr>
<tr><td>0:06:00</td><td>2</td><td rowspan="3">2</td></tr>
<tr><td>0:08:00</td><td>3</td></tr>
<tr><td>0:10:00</td><td></td><td rowspan="5">5</td></tr>
<tr><td>0:11:00</td><td>1</td><td></td></tr>
<tr><td>0:12:00</td><td>5</td><td rowspan="2">5</td></tr>
<tr><td>0:13:00</td><td>5</td></tr>
<tr><td>0:14:00</td><td>1</td><td rowspan="2"></td></tr>
<tr><td>0:15:00</td><td></td></tr>
</table>

Функция Интеграл

Английское имя: Integral.

Назначение: Возвращает интеграл от заданного выражения за период расчета или по указанным сегментам. Результат возвращается в единицах измерения исходного значения умноженных на секунды.

Синтаксис: **Интеграл(Выражение[;Сегменты]).**

Возможные сигнатуры:

действ Интеграл(**действ** *Выражение*);

действ Интеграл(**действ** *Выражение*; **сегменты** *Сегменты*).

Параметры:

Выражение – выражение, от которого вычисляется интеграл;

Сегменты – набор сегментов, на каждом из которых производится вычисление результата функции.

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным началу периода. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если для выражения, в котором используется функция, задана ступенчатая интерполяция, то интегралы вычисляются методом левых прямоугольников, если задана линейная интерполяция, то интегралы вычисляются методом трапеций.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени начала сегмента. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «Интеграл». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00. Интерполяция для выражения задана ступенчатая. Значения функции измеряются в исходных единицах, умноженных на секунду. В скобках описано, как реально производится расчет.

<table>
<tr><th>Время</th><th>А</th><th>Интеграл(А)</th><th>Интеграл(А;РавномерныеСегменты(300))</th><th>Интеграл(А; Сегменты(А>1))</th></tr>
<tr><td>0:00:00</td><td>4</td><td rowspan="12">2700
(4*120+6*120
+0*120+2*120
+3*180+0*60
+3*120+1*60)</td><td rowspan="3">1200
(4*120+6*120)</td><td>1200
(4*120+6*120)</td></tr>
<tr><td>0:02:00</td><td>6</td><td rowspan="3"></td></tr>
<tr><td>0:04:00</td><td>0</td></tr>
<tr><td>0:05:00</td><td></td><td rowspan="3">600
(0*60+2*120+3*120)</td><td rowspan="3">780
(2*120+3*180)</td></tr>
<tr><td>0:06:00</td><td>2</td></tr>
<tr><td>0:08:00</td><td>3</td></tr>
<tr><td>0:10:00</td><td></td><td rowspan="6">900
(3*60+1*60+5*120+1*60)</td><td></td></tr>
<tr><td>0:11:00</td><td>1</td><td rowspan="3">600
(5*120)</td></tr>
<tr><td>0:12:00</td><td>5</td></tr>
<tr><td>0:13:00</td><td>5</td></tr>
<tr><td>0:14:00</td><td>1</td><td rowspan="2"></td></tr>
<tr><td>0:15:00</td><td></td></tr>
</table>

Функция Количество

Английское имя: Count.

Назначение: Возвращает количество переходов заданного условия из 0 в 1 за период расчета или количество переходов заданного условия из 0 в 1 на каждом из указанных сегментов.

Синтаксис: **Количество(Условие[;Сегменты]).**

Возможные сигнатуры:

целое Количество(**логич** *Условие*);

целое Количество(**логич** *Условие*; **сегменты** *Сегменты*).

Параметры:

Условие – условие, количество переходов которого из 0 в 1 определяется;

Сегменты – набор сегментов, на каждом производится вычисление результата функции.

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным началу периода расчета. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным началу сегмента. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «Количество». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00. Для каждого значения, указаны моменты перехода из 0 в 1, по которым это значение вычислено.

<table>
<tr><th>Время</th><th>А</th><th>Количество(А>2)</th><th>Количество(А<3;РавномерныеСегменты(300))</th><th>Количество(А<3; Сегменты(А>1))</th></tr>
<tr><td>0:00:00</td><td>4</td><td rowspan="12">2
(08:00,12:00)</td><td rowspan="3">0</td><td rowspan="2">0</td></tr>
<tr><td>0:02:00</td><td>6</td></tr>
<tr><td>0:04:00</td><td>0</td><td rowspan="2"></td></tr>
<tr><td>0:05:00</td><td></td><td rowspan="3">1
(08:00)</td></tr>
<tr><td>0:06:00</td><td>2</td><td rowspan="3">1
(08:00)</td></tr>
<tr><td>0:08:00</td><td>3</td></tr>
<tr><td>0:10:00</td><td></td><td rowspan="6">1
(12:00)</td></tr>
<tr><td>0:11:00</td><td>1</td><td></td></tr>
<tr><td>0:12:00</td><td>5</td><td rowspan="2">0</td></tr>
<tr><td>0:13:00</td><td>5</td></tr>
<tr><td>0:14:00</td><td>1</td><td></td></tr>
<tr><td>0:15:00</td><td></td><td></td></tr>
</table>

Функция КоличествоТочек

Английское имя: CountPoints.

Назначение: Возвращает количество мгновенных значений заданного выражения за период расчета или количество значений на каждом из указанных сегментов.

Синтаксис: **КоличествоТочек(Выражение[;Сегменты]).**

Возможные сигнатуры:

целое КоличествоТочек(**вариант** *Выражение*);

целое КоличествоТочек(**вариант** *Выражение*; **сегменты** *Сегменты*).

Параметры:

Выражение – выражение, количество значений которого определяется;

Сегменты – набор сегментов, на каждом из которых производится вычисление результата функции.

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным началу периода. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени

начала сегмента. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «КоличествоТочек». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00.

<table>
<tr><th>Время</th><th>А</th><th>Количество Точек(А)</th><th>КоличествоТочек(А; РавномерныеСегменты(300))</th><th>КоличествоТочек (А;Сегменты(А>1))</th></tr>
<tr><td>0:00:00</td><td>4</td><td rowspan="12">9</td><td rowspan="3">3</td><td rowspan="2">2</td></tr>
<tr><td>0:02:00</td><td>6</td></tr>
<tr><td>0:04:00</td><td>0</td><td rowspan="3"></td></tr>
<tr><td>0:05:00</td><td></td><td rowspan="3">2</td></tr>
<tr><td>0:06:00</td><td>2</td></tr>
<tr><td>0:08:00</td><td>3</td><td rowspan="2">2</td></tr>
<tr><td>0:10:00</td><td></td><td rowspan="5">4</td></tr>
<tr><td>0:11:00</td><td>1</td><td></td></tr>
<tr><td>0:12:00</td><td>5</td><td rowspan="2">2</td></tr>
<tr><td>0:13:00</td><td>5</td></tr>
<tr><td>0:14:00</td><td>1</td><td rowspan="2"></td></tr>
<tr><td>0:15:00</td><td></td></tr>
</table>

Функция Максимум

Английское имя: Maximum.

Назначение: Возвращает максимальное из мгновенных значений заданного выражения за период расчета или максимальное значение на каждом из указанных сегментов. Результат возвращается в единицах измерения исходного значения.

Синтаксис: **Максимум(Выражение[;Сегменты]).**

Возможные сигнатуры:

логич Максимум(**логич** *Выражение*);

логич Максимум(**логич** *Выражение*; **сегменты** *Сегменты*);

целое Максимум(**целое** *Выражение*);

целое Максимум(**целое** *Выражение*; **сегменты** *Сегменты*);

действ Максимум(**действ** *Выражение*);

действ Максимум(**действ** *Выражение*; **сегменты** *Сегменты*);

строка Максимум(**строка** *Выражение*);

строка Максимум(**строка** *Выражение*; **сегменты** *Сегменты*).

Параметры:

Выражение – выражение, для которого вычисляется максимум;

Сегменты – набор сегментов, на каждом из которых производится вычисление результата функции.

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным времени максимального значения за период расчета.

Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени максимального значения на сегменте. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «Максимум». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00.

<table>
<tr><th>Время</th><th>А</th><th>Максимум(А)</th><th>Максимум(А;Равномерные Сегменты(300))</th><th>Максимум(А; Сегменты(А>1))</th></tr>
<tr><td>0:00:00</td><td>4</td><td rowspan="12">6</td><td rowspan="3">6</td><td rowspan="2">6</td></tr>
<tr><td>0:02:00</td><td>6</td></tr>
<tr><td>0:04:00</td><td>0</td><td rowspan="2"></td></tr>
<tr><td>0:05:00</td><td></td><td rowspan="3">3</td></tr>
<tr><td>0:06:00</td><td>2</td><td rowspan="2">3</td></tr>
<tr><td>0:08:00</td><td>3</td></tr>
<tr><td>0:10:00</td><td></td><td rowspan="5">5</td><td></td></tr>
<tr><td>0:11:00</td><td>1</td><td></td></tr>
<tr><td>0:12:00</td><td>5</td><td rowspan="2">5</td></tr>
<tr><td>0:13:00</td><td>5</td></tr>
<tr><td>0:14:00</td><td>1</td><td rowspan="2"></td></tr>
<tr><td>0:15:00</td><td></td></tr>
</table>

Функция Минимум

Английское имя: Minimum.

Назначение: Возвращает минимальное из мгновенных значений заданного выражения за период расчета или минимальное значение на каждом из указанных сегментов. Результат возвращается в единицах измерения исходного значения.

Синтаксис: **Минимум(Выражение[;Сегменты]).**

Возможные сигнатуры:

логич Минимум(**логич** *Выражение*);

логич Минимум(**логич** *Выражение*; **сегменты** *Сегменты*);

целое Минимум(**целое** *Выражение*);

целое Минимум(**целое** *Выражение*; **сегменты** *Сегменты*);

действ Минимум(**действ** *Выражение*);

действ Минимум(**действ** *Выражение*; **сегменты** *Сегменты*);

строка Минимум(**строка** *Выражение*);

строка Минимум(**строка** *Выражение*; **сегменты** *Сегменты*).

Параметры:

Выражение – выражение, для которого вычисляется минимум;

Сегменты – набор сегментов, на каждом из которых производится вычисление результата функции.

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным времени минимального значения за период расчета. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени минимального значения на сегменте. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «Минимум». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00.

<table>
<tr><th>Время</th><th>А</th><th>Минимум(А)</th><th>Минимум(А;РавномерныеСегменты(300))</th><th>Минимум(А; Сегменты(А>1))</th></tr>
<tr><td>0:00:00</td><td>4</td><td rowspan="12">0</td><td rowspan="3">0</td><td rowspan="2">0</td></tr>
<tr><td>0:02:00</td><td>6</td></tr>
<tr><td>0:04:00</td><td>0</td><td rowspan="2"></td></tr>
<tr><td>0:05:00</td><td></td><td rowspan="3">0</td></tr>
<tr><td>0:06:00</td><td>2</td><td rowspan="2"></td></tr>
<tr><td>0:08:00</td><td>3</td><td rowspan="2">1</td></tr>
<tr><td>0:10:00</td><td></td><td rowspan="5">1</td></tr>
<tr><td>0:11:00</td><td>1</td><td></td></tr>
<tr><td>0:12:00</td><td>5</td><td rowspan="2">1</td></tr>
<tr><td>0:13:00</td><td>5</td></tr>
<tr><td>0:14:00</td><td>1</td><td></td></tr>
<tr><td>0:15:00</td><td></td><td></td></tr>
</table>

Функция Среднее

Английское имя: Average.

Назначение: Возвращает интегральное среднее от заданного выражения за период расчета или список средних значений по указанным сегментам. Результат возвращается в единицах измерения исходного значения.

Синтаксис: **Среднее(Выражение[;Сегменты]).**

Возможные сигнатуры:

действ Среднее(**действ** *Выражение*);

действ Среднее(**действ** *Выражение*; **сегменты** *Сегменты*).

Параметры:

Выражение– выражение, от которого вычисляется среднее;

Сегменты – набор сегментов, на каждом из которых производится вычисление результата функции.

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным началу периода. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если для выражения, в котором используется функция, задана ступенчатая интерполяция, то интегралы вычисляются методом левых прямоугольников. Если задана линейная интерполяция, то интегралы вычисляются методом трапеций.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени начала сегмента. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «Среднее». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00. Интерполяция для выражения задана ступенчатая. Значения функции измеряются в исходных единицах. Также описано, как реально производится расчет.

<table>
<tr><th>Время</th><th>А</th><th>Среднее(А)</th><th>Среднее(А;Равномерные Сегменты(300))</th><th>Среднее(А; Сегменты(А>1))</th></tr>
<tr><td>0:00:00</td><td>4</td><td rowspan="12">3
(4*120+6*120
+0*120+2*120
+3*180+1*60
+5*120+1*60)
/900</td><td rowspan="3">4
(4*120+6*120) / 300</td><td rowspan="2">5
(4*120+6*120) / 240</td></tr>
<tr><td>0:02:00</td><td>6</td></tr>
<tr><td>0:04:00</td><td>0</td><td rowspan="2"></td></tr>
<tr><td>0:05:00</td><td></td><td rowspan="3">2
(0*60+2*120+3*120) /
300</td></tr>
<tr><td>0:06:00</td><td>2</td><td rowspan="2">2,6
(2*120+3*180) / 300</td></tr>
<tr><td>0:08:00</td><td>3</td></tr>
<tr><td>0:10:00</td><td></td><td rowspan="5">3
(3*60+1*60+5*120+1*60)
/300</td><td></td></tr>
<tr><td>0:11:00</td><td>1</td><td></td></tr>
<tr><td>0:12:00</td><td>5</td><td>5 (5*120) / 300</td></tr>
<tr><td>0:14:00</td><td>1</td><td></td></tr>
<tr><td>0:15:00</td><td></td><td></td></tr>
</table>

Функция СреднееВЗоне

Английское имя: AverageInZone.

Назначение: Возвращает интегральное среднее от заданного выражения за все время, в течение которого выполнялось заданное условие, внутри периода расчета или по каждому из указанных сегментов. Результат возвращается в единицах измерения исходного значения.

Синтаксис: **СреднееВЗоне(Выражение[;Сегменты];Условие;ПоУмолчанию).**

Возможные сигнатуры:

действ Среднее(**действ** *Выражение*; **логич** *Условие*; **действ** *ПоУмолчанию*);

действ Среднее(**действ** *Выражение*; **сегменты** *Сегменты*;

логич *Условие*; **действ** *ПоУмолчанию*).

Параметры:

Выражение– выражение, от которого вычисляется среднее;

Условие – условие зоны, в которой вычисляется среднее;

ПоУмолчанию – значение, которое возвращает функция, если за период расчета или за сегмент условие ни разу не выполнилось;

Сегменты – набор сегментов, на каждом из которых производится вычисление результата функции.

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным началу периода. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если для выражения, в котором используется функция, задана ступенчатая интерполяция, то интегралы вычисляются методом левых прямоугольников. Если задана линейная интерполяция, то интегралы вычисляются методом трапеций.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени начала сегмента. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «СреднееВЗоне». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00. Интерполяция для выражения задана ступенчатая. Значения функции измеряются в исходных единицах.

<table>
<tr><th>Время</th><th>А</th><th>СреднееВЗоне (А;А>3;7)</th><th>СреднееВЗоне(А; РавномерныеСегменты (300); А>3; 7)</th><th>Среднее(А; Сегменты(А>1))</th></tr>
<tr><td>0:00:00</td><td>4</td><td rowspan="12">5
(4*120+6*120 +5*120) / 360</td><td rowspan="3">5
(4*120+6*120) / 240</td><td rowspan="2">5
(4*120+6*120) / 240</td></tr>
<tr><td>0:02:00</td><td>6</td></tr>
<tr><td>0:04:00</td><td>0</td><td rowspan="3"></td></tr>
<tr><td>0:05:00</td><td></td><td rowspan="3">7
(нет значений в зоне)</td></tr>
<tr><td>0:06:00</td><td>2</td><td rowspan="3">7
(нет значений в зоне)</td></tr>
<tr><td>0:08:00</td><td>3</td></tr>
<tr><td>0:10:00</td><td></td><td rowspan="5">5</td></tr>
<tr><td>0:11:00</td><td>1</td><td></td></tr>
<tr><td>0:12:00</td><td>5</td><td>5 (5*120/120)</td></tr>
<tr><td>0:14:00</td><td>1</td><td rowspan="2"></td></tr>
<tr><td>0:15:00</td><td></td></tr>
</table>

Функция Сумма

Английское имя: Summ.

Назначение: Возвращает сумму всех мгновенных значений заданного выражения за период расчета или суммы мгновенных значений на каждом из указанных сегментов.

Синтаксис: **Сумма(Выражение[;Сегменты]).**

Возможные сигнатуры:

действ Сумма(**действ** *Выражение*);

действ Сумма(**действ** *Выражение*; **сегменты** *Сегменты*);

целое Сумма(**целое** *Выражение*);

целое Сумма(**целое** *Выражение*; **сегменты** *Сегменты*).

Параметры:

Выражение– выражение, для которого вычисляется сумма значений;

Сегменты – набор сегментов, на каждом из которых производится вычисление результата функции.

Если *Сегменты* не заданы, то функция возвращает одно значение за период расчета. Время значения задается равным началу периода. Недостоверность (ошибка) итогового значения за период равна максимуму недостоверностей (ошибок) исходных мгновенных значений.

Если *Сегменты* заданы, то функция возвращает по одному значению для каждого сегмента. Если для сегмента время значения не задано, то время значения задается равным времени начала сегмента. Недостоверность (ошибка) итогового значения за сегмент равна максимуму недостоверностей (ошибок) исходных мгновенных значений, принадлежащих сегменту.

Пример: В таблице приведен пример результатов вычисления трех выражений, использующих функцию «Сумма». Результатом каждого выражения является мгновенное значение или список мгновенных значений. Функции берутся от исходного параметра «А», мгновенные значения которого также приведены. Расчет производится за период от 0:00:00 до 0:15:00. В скобках описано, как реально производится расчет.

<table>
<tr><th>Время</th><th>А</th><th>Сумма(А)</th><th>Сумма(А;Равномерные Сегменты(300))</th><th>Сумма(А; Сегменты(А>1))</th></tr>
<tr><td>0:00:00</td><td>4</td><td rowspan="12">27
(4+6+0+2+3+1
+5+5+1)</td><td rowspan="3">10
(4+6+0)</td><td rowspan="2">10
(4+6)</td></tr>
<tr><td>0:02:00</td><td>6</td></tr>
<tr><td>0:04:00</td><td>0</td><td rowspan="2"></td></tr>
<tr><td>0:05:00</td><td></td><td rowspan="2">5
(2+3)</td></tr>
<tr><td>0:06:00</td><td>2</td><td rowspan="2">5
(2+3)</td></tr>
<tr><td>0:08:00</td><td>3</td></tr>
<tr><td>0:10:00</td><td></td><td rowspan="5">12
(1+5+5+1)</td><td></td></tr>
<tr><td>0:11:00</td><td>1</td><td rowspan="2">10
(5+5)</td></tr>
<tr><td>0:12:00</td><td>5</td></tr>
<tr><td>0:13:00</td><td>5</td><td rowspan="2"></td></tr>
<tr><td>0:14:00</td><td>1</td></tr>
<tr><td>0:15:00</td><td></td><td></td></tr>
</table>

4.8. Функции работы с сегментами

Расчетные значения в языке Tablik не обязательно являются мгновенными значениями или списками мгновенных значений. В языке еще есть особый тип данных – **сегменты**. Одно значение такого типа представляет собой набор сегментов. Именно для формирования таких значений предназначены функции этого раздела. Сегменты можно использовать в качестве входных параметров статистических функций (см. раздел 4.7). Задание сегментов позволяет вычислить не одно значение статистической функции за весь период расчета, а много значений, каждое на своем временном интервале, то есть сегменте.

Один сегмент представляет собой небольшую структуру, содержащую следующие данные:

- Начало сегмента – момент времени, с которого начинается сегмент;
- Конец сегмента – момент времени, которым оканчивается сегмент, конец не должен лежать раньше начала;
- Положение результата – число от 0 до 1, определяет момент времени, который должен быть приписан мгновенному значению, полученному в результате вычисления статистической функции по данному сегменту. Момент времени результата определяется по формуле:

Начало сегмента + (Начало сегмента – Конец сегмента) * Положение;

- Недостоверность – формируется по недостоверности исходных значений функции, порождающей сегмент;
- Ошибка - формируется по ошибкам исходным значениям порождающей сегмент функции или возникает непосредственно при формировании сегмента.

Сегменты используются в тесной связи со статистическими функциями, которые описаны в разделе 4.7. В статистическую функцию («Среднее», «Максимум», «Интеграл» и т.п.) обычно передается исходное значение (список мгновенных значений) для статистической обработки, а также может передаваться набор сегментов. Если набор сегментов задан, то на каждом сегменте вычисляются свои значения (например, среднее на сегменте), которые затем объединяются в результирующий список мгновенных значений. В разделе 4.7 для каждой статистической функции приведен пример вызова без указания сегментов, а также с указанием сегментов как результата вычисления функций «РавномерныеСегменты» и «Сегменты».

Время итогового мгновенного значения на сегменте формируется исходя из начала и конца сегмента, а также из положения результата. Сегменты в наборе сегментов могут между собой пересекаться, например, такие сегменты получаются при использовании функции «СегментыОтТочек».

Функция РавномерныеСегменты

Английское имя: UniformSegments.

Назначение: Возвращает набор сегментов одинаковой заданной длительности, на которые разбивается период расчета. Дополнительно может быть указано относительное положение результата, одинаковое для всех возвращаемых сегментов. Начало первого сегмента совпадает с началом периода расчета. Начало каждого следующего сегмента совпадает с концом предыдущего. Если сегмент не вмещается в период расчета, то он не добавляется в результат.

Недостоверность и ошибка каждого из сегментов равны максимуму недостоверности и ошибки входных параметров.

Синтаксис: **РавномерныеСегменты(ДлинаСегмента[;Положение]).**

Сигнатура:

сегменты РавномерныеСегменты (**действ** *ДлинаСегмента* [; **действ** *Положение*]).

Параметры:

ДлинаСегмента – длина одного сегмента в секундах;

Положение – число от 0 до 1 – положение результата статистической функции относительно каждого из возвращаемых сегментов.

Пример: Пусть производится расчет за период от 0:00:00 до 1:00:00. Выражение РавномерныеСегменты(600;1) вернет следующий список сегментов (каждый по 10 минут):

Начало	Конец	Положение	Время положения
0:00:00	0:10:00	1	0:10:00
0:10:00	0:20:00	1	0:20:00
0:20:00	0:30:00	1	0:30:00
0:30:00	0:40:00	1	0:40:00
0:40:00	0:50:00	1	0:50:00
0:50:00	1:00:00	1	1:00:00

Выражение РавномерныеСегменты(1500;0.5) вернет следующий список сегментов (каждый по 25 минут, третий не вмещается в период расчета):

Начало	Конец	Положение	Время положения
0:00:00	0:25:00	0,5	0:12:30
0:25:00	0:50:00	0,5	0:37:30

Функция Сегменты

Английское имя: Segments.

Назначение: Формирует сегменты по интервалам истинности заданного условия. Каждый из сегментов начинается с момента изменения исходного условия с 0 на 1 и продолжается до тех пор, пока значение условия не изменится обратно на 0. Если в начале периода расчета значение условия равно 1, то первый сегмент начинается с начала периода расчета. Если в конце периода расчета значение условия равно 1, то конец последнего сегмента совпадает с концом периода расчета. Дополнительно может быть указано относительное положение результата, одинаковое для всех возвращаемых сегментов.

Недостоверность и ошибка каждого из возвращаемых сегментов равны максимуму недостоверностей и ошибок входных параметров на временном интервале этого сегмента.

Синтаксис: **Сегменты(Условие[;Положение]).**

Сигнатура: **сегменты** Сегменты (**логич** *Условие* [; **действ** *Положение*]).

Параметры:

Условие – логическое условие, задающее сегменты;

Положение – число от 0 до 1 – положение результата статистической функции относительно каждого из возвращаемых сегментов.

Пример: Пусть в проекте есть сигнал {S} и расчетный параметр «А» с формулой Сегменты({S}>0;0). Пусть производится расчет за период от 0:00:00 до 1:00:00. В следующих таблицах приведены мгновенные значения сигнала {S} и результирующий список сегментов – значение параметра «А».

{S}				Сегменты({S}>0;0)			
Время	Значение	Недост.		Начало	Конец	Время положения	Недост.
0:00:00	2	0		0:00:00	0:10:00	0:00:00	0
0:10:00	-1	0		0:20:00	0:30:00	0:20:00	1
0:20:00	1	0		0:40:00	0:50:00	0:40:00	0
0:25:00	2	1					
0:30:00	-2	0					
0:40:00	1	0					
0:50:00	-2	0					

Функция СегментыОтТочек

Английское имя: SegmentsFromPoints.

Назначение: Возвращает набор сегментов, по одному на каждое мгновенное значение указанного выражения. Кроме исходного выражения задается длина сегмента. Если задана положительная длина сегмента, то сегмент начинается с момента одного из мгновенных значений выражения и продолжается в течение указанной временной длительности. Если задана отрицательная длина сегмента, то сегмент продолжается в течение указанной временной длительности и заканчивается одним из мгновенных значений выражения. Дополнительно может быть указано относительное положение результата, одинаковое для всех возвращаемых сегментов. Сегменты из возвращаемого списка могут пересекаться между собой. Сегменты могут выходить за период расчета.

Недостоверность и ошибка сегмента вычисляются как максимум недостоверностей и ошибок входных параметров на момент, от которого отсчитывается сегмент.

Синтаксис: **СегментыОтТочек(Выражение;ДлинаСегмента[;Положение]).**

Сигнатура: **сегменты** СегментыОтТочек(**вариант** Выражение; **действ** *ДлинаСегмента*

[;**действ** *Положение*]).

Пример: Пусть в проекте есть сигнал {S} и два расчетных параметра «A» и «B» со следующими формулами:

Параметр	Формула
A	СегментыОтТочек({S};600;0.5)
B	СегментыОтТочек({S};-900;1)

Пусть расчет производится за период от 1:00:00 до 2:00:00 и сигнал {S} имеет значения:

{S}		
Время	Значение	Недост.
1:00:00	0	0
1:05:00	1	0
1:25:00	2	1
1:30:00	3	0
1:55:00	4	0

Тогда параметр «A» вернет следующий список сегментов:

СегментыОтТочек({S};600;0.5)			
Начало	Конец	Время положения	Недост.
1:00:00	1:10:00	1:05:00	0
1:05:00	1:15:00	1:10:00	0
1:25:00	1:35:00	1:30:00	1
1:30:00	1:40:00	1:35:00	0
1:55:00	2:05:00	2:00:00	0

Тогда параметр «B» вернет следующий список сегментов:

СегментыОтТочек({S};-900;1)			
Начало	Конец	Время положения	Недост.
0:45:00	1:00:00	1:00:00	0
0:50:00	1:05:00	1:05:00	0
1:10:00	1:25:00	1:25:00	1
1:15:00	1:30:00	1:30:00	0
1:40:00	1:55:00	1:55:00	0

Функция КоличествоСегментов

Английское имя: SegmentsCount.

Назначение: Возвращает количество сегментов в указанном списке сегментов.

Синтаксис: **КоличествоСегментов(Сегменты).**

Сигнатура: **целое** КоличествоСегментов(**сегменты** *Сегменты*).

Параметры: *Сегменты* – сегменты, количество которых вычисляется.

Пример: КоличествоСегментов(S).

Функция ОбъединитьСегменты

Английское имя: UnionSegments.

Назначение: Объединяет указанные наборы сегментов.

Синтаксис: **ОбъединитьСегменты(Сегменты1;Сегменты2;…).**

Сигнатура:

сегменты ОбъединитьСегменты(**сегменты** *Сегменты1*; **сегменты** *Сегменты2;…*).

Параметры: *Сегменты1, Сегменты2* и т.д. – наборы сегментов, которые объединяются в один набор сегментов.

Пример: ОбъединитьСегменты(S1;S2).

Функция СегментПоНомеру

Английское имя: SegmentByNumber.

Назначение: Выделение одного сегмента по номеру с начала из набора сегментов. Сегменты считаются упорядоченными по времени начала, при равном времени начала упорядочиваются по времени конца. В результате получается список сегментов, состоящий из одного сегмента.

Синтаксис: **СегментПоНомеру(Сегменты;Номер).**

Сигнатура: **сегменты** СегментПоНомеру(**сегменты** *Сегменты*; **целое** *Номер*).

Параметры:

Сегменты – исходный список сегментов;

Номер – нос сегмента, который выбирается из списка.

Пример: СегментПоНомеру(S;1).

4.9. Функции работы с таблицами

Данный раздел описывает группу функций «Таблицы». Функции данной группы служат для получения записи из дополнительных таблиц проекта.

Каждая дополнительная таблица проекта содержит поля:

- **Num**. Порядковый номер записи. Должен быть уникальным. Используется для идентификации записи при вызове функций работы с таблицами;
- **Code**. Код записи. Должен быть уникальным. Заполняется пользователем. Используется для идентификации записи при вызове функций работы с таблицами;
- **Name**. Наименование (описание) записи. Может быть получено использованием соответствующей функции;
- **Val_0 … Val_20.** Поля значений для записи. Содержат информацию, возвращаемую функциями работы с таблицами. Идентифицируются своим номером.

Каждая дополнительная таблица может иметь подчиненную подтаблицу. Каждая запись подтаблицы дополнительных данных имеет следующие поля:

- **SubNum**. Порядковый номер. Должен быть уникальным в пределах одной записи родительской таблицы. Используется для идентификации записи подтаблицы при вызове функций работы с таблицами;
- **SubCode**. Код записи. Должен быть уникальным в пределах одной записи родительской таблицы. Используется для идентификации записи подтаблицы при вызове функций работы с таблицами;

- **SubName**. Наименование (описание) записи подтаблицы. Может быть получено использованием соответствующей функции;
- **SubVal_0 … SubVal_20.** Поля значений для записи подтаблицы. Создаются и заполняются пользователем. Содержат информацию, возвращаемую функциями работы с таблицами. Идентифицируются своим номером.

Функция ВзятьТабл

Английское имя: GetTabl.

Назначение: Получение значения ячейки из справочной таблицы по номеру записи или по коду записи.

Синтаксис: **ВзятьТабл(НомерТаблицы[;Код][;Номер];НомерПоля).**

Возможные сигнатуры:

строка ВзятьТабл(**целое** *НомерТаблицы*; **строка** *Код*; **целое** *НомерПоля*);

строка ВзятьТабл(**целое** *НомерТаблицы*; **целое** *Номер*; **целое** *НомерПоля*).

Параметры:

НомерТаблицы – номер таблицы, значение ячейки из которой нужно получить;

Код – строковый код, задающий запись в таблице;

Номер – числовой код, задающий номер записи в таблице;

Номер поля – номер поля, значение которого нужно получить.

Примеры: ВзятьТабл(1;'Code';2), ВзятьТабл(2;100;3).

Функция ВзятьТаблИмя

Английское имя: GetTablName.

Назначение: Получение значения поля «Имя» из справочной таблицы по номеру записи или по коду записи.

Синтаксис: **ВзятьТаблИмя(НомерТаблицы[;Код][;Номер]).**

Возможные сигнатуры:

строка ВзятьТабл(**целое** *НомерТаблицы*; **строка** *Код*);

строка ВзятьТабл(**целое** *НомерТаблицы*; **целое** *Номер*).

Параметры:

НомерТаблицы – номер таблицы, имя записи из которой нужно получить;

Код – строковый код, задающий запись в таблице;

Номер – числовой код, задающий номер записи в таблице.

Примеры: ВзятьТаблИмя(1;'Code'), ВзятьТаблИмя(2;100).

Функция ТаблСодержит

Английское имя: TablContains.

Назначение: Возвращает, содержит ли справочная таблица запись с заданным номером или кодом и поле с указанным номером.

Синтаксис: **ТаблСодержит(НомерТаблицы[;Код][;Номер];НомерПоля).**

Возможные сигнатуры:

логич ТаблСодержит (**целое** *НомерТаблицы*; **строка** *Код*; **целое** *НомерПоля*);

логич ТаблСодержит (**целое** *НомерТаблицы*; **целое** *Номер*; **целое** *НомерПоля*).

Параметры:

НомерТаблицы – номер таблицы, в которой выполняется поиск указанной записи;

Код – строковый код записи, которую нужно найти;

Номер – числовой код, задающий номер записи, которую нужно найти;

Номер поля – номер поля, которое нужно найти.

Примеры: ВзятьТабл(1;'Code';2), ВзятьТабл(2;100;3).

Функция ВзятьПодТабл

Английское имя: GetSubTabl.

Назначение: Получение значения ячейки из справочной подтаблицы по номеру или коду записи в основной таблице и номеру или коду записи в подтаблице.

Синтаксис:

ВзятьПодТабл(НомерТаблицы[;Код][;Номер][;КодПод][;НомерПод];НомерПоля).

Возможные сигнатуры:

строка ВзятьПодТабл(**целое** *НомерТаблицы*; **строка** *Код*; **строка** *КодПод*; **целое** *НомерПоля*);

строка ВзятьПодТабл(**целое** *НомерТаблицы*; **целое** *Номер*; **строка** *КодПод*; **целое** *НомерПоля*);

строка ВзятьПодТабл(**целое** *НомерТаблицы*; **строка** *Код*; **целое** *НомерПод*; **целое** *НомерПоля*);

строка ВзятьПодТабл(**целое** *НомерТаблицы*; **целое** *Номер*; **целое** *НомерПод*; **целое** *НомерПоля*).

Параметры:

НомерТаблицы – номер таблицы, значение ячейки из подтаблицы которой нужно получить;

Код – строковый код, задающий запись в таблице;

Номер – числовой код, задающий номер записи в таблице;

КодПод – строковый код, задающий запись в подтаблице;

НомерПод – числовой код, задающий номер записи в подтаблице;

НомерПоля – номер поля подтаблицы, значение которого нужно получить.

Примеры: ВзятьПодТабл(1;'Code';'SubCode';2), ВзятьПодТабл(2;100;'SubCode';3).

Функция ВзятьПодТаблИмя

Английское имя: GetSubTablName.

Назначение: Получение значения поля «Имя» из справочной подтаблицы по номеру или коду записи в основной таблице и номеру или коду записи в подтаблице

Синтаксис: **ВзятьПодТаблИмя(НомерТаблицы[;Код][;Номер][;КодПод][;НомерПод]).**

Возможные сигнатуры:

строка ВзятьПодТаблИмя(**целое** *НомерТаблицы*; **строка** *Код*; **строка** *КодПод*);

строка ВзятьПодТаблИмя(**целое** *НомерТаблицы*; **целое** *Номер*; **строка** *КодПод*);

строка ВзятьПодТаблИмя(**целое** *НомерТаблицы*; **строка** *Код*; **целое** *НомерПод*);

строка ВзятьПодТаблИмя(**целое** *НомерТаблицы*; **целое** *Номер*; **целое** *НомерПод*).

Параметры:

НомерТаблицы – номер таблицы, имя записи из подтаблицы которой нужно получить;

Код – строковый код, задающий запись в таблице;

Номер – числовой код, задающий номер записи в таблице;

КодПод – строковый код, задающий запись в подтаблице;

НомерПод – числовой код, задающий номер записи в подтаблице.

Примеры: ВзятьПодТаблИмя(1;'Code';'SubCode'), ВзятьПодТаблИмя(2;100;'SubCode').

Функция ПодТаблСодержит

Английское имя: SubTablContains.

Назначение: Возвращает, содержит ли справочная подтаблица заданную ячейку.

Ячейка задается номером или кодом записи в основной таблице, номером или кодом записи в подтаблицы, а также номером поля в подтаблице.

Синтаксис:

ПодТаблСодержит(НомерТаблицы[;Код][;Номер][;КодПод][;НомерПод];НомерПоля).

Возможные сигнатуры:

логич ПодТаблСодержит(**целое** *НомерТаблицы*; **строка** *Код*; **строка** *КодПод*; **целое** *НомерПоля*);

логич ПодТаблСодержит(**целое** *НомерТаблицы*; **целое** *Номер*; **строка** *КодПод*; **целое** *НомерПоля*);

логич ПодТаблСодержит(**целое** *НомерТаблицы*; **строка** *Код*; **целое** *НомерПод*; **целое** *НомерПоля*);

логич ПодТаблСодержит(**целое** *НомерТаблицы*; **целое** *Номер*; **целое** *НомерПод*; **целое** *НомерПоля*).

Параметры:

НомерТаблицы – номер таблицы, в подтаблице которой выполняется поиск записи;

Код – строковый код, задающий искомую запись в таблице;

Номер – числовой код, задающий номер искомой записи в таблице;

КодПод – строковый код, задающий искомую запись в подтаблице;

НомерПод – числовой код, задающий искомый номер записи в подтаблице;

НомерПоля – номер поля подтаблицы, которое нужно найти.

Пример: ПодТаблСодержит(1;'Code';'SubCode';2), ПодТаблСодержит(2;100;'SubCode';3).

4.10. Функции типов данных

Данный раздел описывает группу функций «Типы данных». Все функции этой группы являются скалярными. Функции этой группы служат либо для преобразования значений из одного типа данных в другой, либо для проверки, имеет ли заданное выражение заданный тип данных.

Функция Величина

Английское имя: Value.

Назначение: Перевод выражения в тип «величина». От исходного значения остаются только время, недостоверность и ошибка.

Синтаксис: **Величина(Выражение).**

Сигнатура: **величина** Величина(**вариант** *Выражение*).

Параметр: *Выражение* – выражение, от значения которого берется величина.

Пример: Величина(1)

Функция Дата

Английское имя: Date.

Назначение: Перевод значения в дату. Если значение аргумента – строка, не являющаяся записью даты, то функция возвращает ошибку.

Синтаксис: **Дата(Выражение).**

Возможные сигнатуры:

время Время(**время** *Выражение*);

время Время(**строка** *Выражение*).

Параметр: *Выражение* – строка, из которой получается дата.

Пример: Дата('10.10.2010 10:10:10') = #10.10.2010 10:10:10#.

Функция Действ

Английское имя: Real.

Назначение: Перевод значений разных типов в значение типа «действ». Если аргумент имеет тип «логич», «целое» или «действ», то возвращается число равное исходному. Если аргумент имеет тип «строка», то запись в строчном виде преобразуется в «число». Если значение аргумента не является записью «число», то функция возвращает ошибку.

Синтаксис: **Действ(Выражение).**

Возможные сигнатуры:

действ Действ(**логич** *Выражение*);

действ Действ(**целое** *Выражение*);

действ Действ(**действ** *Выражение*);

действ Действ(**строка** *Выражение*).

Параметр: *Выражение* – строка, из которой получается действительное число.

Пример: Действ('10,2') = 10,2.

Функция Логич

Английское имя: Bool.

Назначение: Перевод выражения в логическое значение. Числовое значение 1 и строковое ‘1’ переводятся в «Правду», 0 и “0” в «Ложь», в остальных случаях функция возвращает значение с ошибкой.

Синтаксис: **Логич(Выражение).**

Возможные сигнатуры:

логич Логич(**логич** *Выражение*);

логич Логич(**целое** *Выражение*);

логич Логич(**действ** *Выражение*);

логич Логич(**строка** *Выражение*).

Параметр: *Выражение* – выражение, из которого получается логическое значение.

Пример: Логич(1)=Правда.

Функция Строка

Английское имя: String.

Назначение: Перевод числового значения или времени в строчную запись.

Синтаксис: **Строка(Выражение).**

Возможные сигнатуры:

строка Строка(**логич** *Выражение*);

строка Строка(**целое** *Выражение*);

строка Строка(**действ** *Выражение*);

строка Строка(**время** *Выражение*);

строка Строка(**целое** *Выражение*).

Параметры: *Выражение* – выражение, из которого получается строка.

Пример: Строка(3)=’3’, Строка(#10.10.2010 10:10:10#)=’10.10.2010 10:10:10’.

Функция Целое

Английское имя: Int.

Назначение: Преобразует значения разных типов в целые числа. Из действительных чисел целое получается отбрасыванием дробной части. Результат операции – это максимальное целое число, среди чисел меньше данного значения. Из строки значение получается преобразованием строчной записи в число. Если входная строка не является записью целого числа, то возвращается значение с ошибкой.

Синтаксис: **Целое(Выражение)**

Возможные сигнатуры:

целое Целое(**логич** *Выражение*);

целое Целое(**целое** *Выражение*);

целое Целое(**действ** *Выражение*);

целое Целое(**строка** *Выражение*).

Параметры: *Выражение* – выражение, из которого получается целое число.

Пример: Int(4,91734)=4, Int(-2.39847)=-3, Int('33')=33.

Функция ЯвляетсяВременем

Английское имя: IsTime.

Назначение: Возвращает логическое значение, которое истинно, если входное значение является строковой записью константы типа время.

Синтаксис: ЯвляетсяВременем(Выражение).

Сигнатура: **логич** ЯвляетсяВременем (**строка** *Выражение*).

Параметр: *Выражение* – строка, для которой проверяется, является ли она временем.

Пример: ЯвляетсяВременем('10.10.2010 10:10:10') = True.

Функция ЯвляетсяЦелым

Английское имя: IsTime.

Назначение: Возвращает логическое значение, которое истинно, если входное значение является целым числом или строчной записью целого числа.

Синтаксис: ЯвляетсяЦелым (Выражение).

Возможные сигнатуры:

логич ЯвляетсяЦелым (**действ** *Выражение*);

логич ЯвляетсяЦелым (**строка** *Выражение*).

Параметр: *Выражение* – выражение, целочисленность которого проверяется.

Пример: ЯвляетсяЦелым(3)=True, ЯвляетсяЦелым(3,2)=False, ЯвляетсяЦелым('ss')=False.

Функция ЯвляетсяЧислом

Английское имя: IsReal.

Назначение: Возвращает логическое значение, которое истинно, если входное выражение является строчной записью вещественного числа.

Синтаксис: ЯвляетсяЧислом(Выражение).

Сигнатура: **логич** ЯвляетсяЧислом (**строка** *Выражение*).

Параметр: *Выражение* – выражение, для которого проверяется, является ли оно действительным числом.

Пример: ЯвляетсяЧислом(3)=True, ЯвляетсяЧислом(3,2)=True, ЯвляетсяЧислом('ss')=False.

4.11. Функции работы со временем

Данный раздел описывает группу функций «Время». Все функции этой группы являются скалярными.

Функция НачалоПериода

Английское имя: BeginPeriod.

Назначение: Возвращает время начала текущего периода расчета.

Сигнатура: **время** НачалоПериода.

Пример: НачалоПериода.

Функция КонецПериода

Английское имя: EndPeriod.

Назначение: Возвращает время конца текущего периода расчета.

Сигнатура: **время** КонецПериода.

Пример: КонецПериода.

Функция Сейчас

Английское имя: Now.

Назначение: Возвращает текущее время на момент запуска данной функции.

Сигнатура: **время** Сейчас.

Пример: Сейчас.

Функция ДобавитьКоВремени

Английское имя: TimeAdd.

Назначение: Добавить ко времени заданное количество заданных временных интервалов.

Синтаксис: **ДобавитьКоВремени(Интервал;Время;Количество).**

Сигнатура:

время ДобавитьКоВремени(**строка** *Интервал*; **время** *Время*; **целое** *Количество*).

Параметры:

Интервал – временной интервал, задающий размер временных интервалов, которые прибавляются к исходному времени. Может принимать значение одной из временных констант: кГод – год, кМесяц – месяц, кСутки – сутки, кЧас – час, кМинута – минута, кСекунда – секунда, кмс – милисекунда.

Время – время, к которому прибавляются интервалы.

Количество – количество интервалов типа *Интервал*, которые прибавляются ко *Времени*.

Пример: ДобавитьКоВремени(кЧас; #10.10.2013 10:10:10#; 3) = #10.10.2013 13:10:10#.

Функция РазностьВремен

Английское имя: TimeDiff.

Назначение: Разность между заданными временами, измеряемая в указанных временных интервалах.

Синтаксис: **РазностьВремен(Интервал;Уменьшаемое;Вычитаемое).**

Сигнатура:

Действ Разность времен(**строка** *Интервал*; **время** *Уменьшаемое*; **время** *Вычитаемое*).

Параметры: *Интервал* – временной интервал, задающий размер временных интервалов, в которых исчисляется разность времен. Может принимать значение одной из временных констант: кГод - год, кМесяц - месяц, кСутки – сутки, кЧас – час, кМинута – минута, кСекунда – секунда, кмс – милисекунда.

Уменьшаемлое – время, из которого вычитается *Вычитаемое*.

Пример: РазностьВремен(кЧас; #10.10.2013 10:10:10#; #10.10.2013 07:40:10#) = 2,5.

Функция СобратьВремя

Английское имя: TimeSerial.

Назначение: Формирует время из указанного количества интервалов разных типов.

Функция: **СобратьВремя(Годы;[Месяцы];[Дни];[Часы];[Минуты];[Секунды];[мс]).**

Сигнатура: Время **СобратьВремя(целое** *Годы*; **[целое** *Месяцы***]; [целое** *Дни***]; [целое** *Часы***]; [целое** *Минуты***]; [целое** *Секунды***]; [целое** *мс***]).**

Параметры: *Годы* – номер года, *Месяцы* – номер месяца, *Сутки* – номер дня, *Часы* – количество часов, *Минуты* – количество минут, *Секунды* – количество секунд, *мс* – количество миллисекунд

Пример: СобратьВремя(2013;10;10;12;0;0) = #10.10.2013 12:00:00#.

Функция ЧастьВремени

Английское имя: TimePart.

Назначение: Количество интервалов указанного типа, составляющее часть времени.

Функция: **ЧастьВремени(Интервал;Время).**

Сигнатура: **целое** ЧастьВремени(**строка** *Интервал*; **время** *Время*).

Параметры:

Интервал – временной интервал, задающий размер временных интервалов, количество которых извлекается из указанного времени. Может принимать значение одной из временных констант: кГод – год, кМесяц – месяц, кСутки – сутки, кЧас – час, кМинута – минута, кСекунда – секунда, кмс – милисекунда;

Время – время, количество интервалов из которого вычисляется.

Пример: ЧастьВремени(кЧас;#10.10.2013 10:10:10#) = 10.

Константы интервалов времени

- Константа кГод. Английское имя: cYear. Константа, задающая период – год;
- Константа кМесяц. Английское имя: cMonth. Константа, задающая период – месяц;
- Константа кСутки. Английское имя: cDay. Константа, задающая период – сутки;
- Константа кЧас. Английское имя: cHour. Константа, задающая период – час;
- Константа кМинута. Английское имя: cMinute. Константа, задающая период – минута;
- Константа кСекунда. Английское имя: cSecond. Константа, задающая период – секунда;
- Константа кмс. Английское имя: cms. Константа, задающая период – миллисекунда.

4.12. Строковые функции

Данный раздел описывает группу функций «Строковые». Все функции этой группы являются скалярными.

Строковые константы

- Константа NewLine

Назначение: Символ перевода строки.

Пример: ‘ABC’ + NewLine.

- Константа Tab

Назначение: Символ табуляции.

Пример: Tab + ‘ABC’.

Функция StrFind

Назначение: Возвращает положение первого вхождения одной строки в другую.

Синтаксис: **StrFind(Искомая;Просматриваемая;[Начало]).**

Сигнатура: **целое** StrFind(**строка** *Искомая*; **строка** *Просматриваемая*; [**целое** *Начало*]).

Параметры:

Искомая – строка, которая ищется;

Просматриваемая – строка, в которой ищется искомая строка;

Начало – номер позиции в просматриваемой строке, начиная с которой ищутся вхождения искомой строки, по умолчанию 1.

Пример: StrFind('aaa';'baaabaaab';1) = 2.

Функция StrFindLast

Назначение: Возвращает положение последнего вхождения одной строки в другую.

Синтаксис: **StrFindLast(Искомая;Просматриваемая;[Начало]).**

Сигнатура: **целое** StrFindLast(**строка** *Искомая*; **строка** *Просматриваемая*; [**целое** *Начало*]).

Параметры:

Искомая – строка, которая ищется;

Просматриваемая – строка, в которой ищется искомая строка;

Начало – номер позиции в просматриваемой строке, начиная с которой, если просматривать с конца, ищутся вхождения искомой строки. Если параметр не задан, то искомая строка ищется, начиная с конца просматриваемой строки.

Пример: StrFindLast('aaa';'baaabaaab';4) = 6.

Функция StrInsert

Назначение: Вставка одной строки в другую, начиная с заданной позиции.

Синтаксис: **StrInsert(Куда;Что;Позиция).**

Сигнатура: **строка StrInsert**(**строка** *Куда*; **строка** *Что*; **целое** *Позиция*).

Параметры: *Куда* – строка, в которую вставляется строка *Что*. Строка вставляется до позиции с номером *Позиция*.

Пример: StrInsert('aaa';'bbb';2) = ’abbbaa’.

Функция StrLCase

Назначение: Перевод всех символов строки в нижний регистр (заглавные буквы переводятся в строчные).

Синтаксис: **StrLCase(Строка).**

Сигнатура: **строка** StrLCase(**строка** *Строка*).

Параметры: *Строка* – исходная строка.

Пример: StrLCase('AAA') = ’aaa’.

Функция StrLeft

Назначение: Выделение подстроки заданной длины, с которой начинается заданная строка.

Синтаксис: **StrLeft(Строка;Длина).**

Сигнатура: **строка** StrLeft(**строка** *Строка;* **целое** *Длина*).

Параметры:

Строка – исходная строка;

Длина – длина выделяемой подстроки.

Пример: StrLeft('aaa';2) = ’aa’.

Функция StrLen

Назначение: Возвращает длину заданной строки.

Синтаксис: **StrLen(Строка).**

Сигнатура: **целое** StrLen(**строка** *Строка*).

Параметр: *Строка* – строка, длина которой вычисляется.

Пример: StrLen('aaa') = 3.

Функция StrLTrim

Назначение: Удаляет все пробелы с начала заданной строки.

Синтаксис: **StrLTrim(Строка).**

Сигнатура: **строка** *StrLTrim***(строка** *Строка***).**

Параметр: *Строка* – строка, из которой удаляются пробелы.

Пример: StrLTrim(' aaa') = ’aaa’.

Функция StrMid

Назначение: Выделение из заданной строки подстроки указанной длины и начинающейся с указанной позиции.

Синтаксис: **StrMid(Строка;Начало;Длина).**

Сигнатура: **строка** *StrMid*(**строка** *Строка;* **целое** *Начало;* **целое** *Длина*)**.**

Параметры:

Строка – исходная строка;

Начало – позиция в исходной строке, начиная с которой выделяется подстрока;

Длина – количество символов в выделяемой подстроке.

Пример: StrMid('aaba';2;2) = ’ab’.

Функция StrRemove

Назначение: Удаление из заданной строки подстроки указанной длины, начинающейся с указанной позиции.

Синтаксис: **StrRemove(Строка;Позиция;Длина).**

Сигнатура: **строка** *StrRemove*(**строка** *Строка;* **целое** *Позиция;* **целое** *Длина*)**.**

Параметры:

Строка – исходная строка;

Начало – позиция в исходной строке, начиная с которой удаляется подстрока;

Длина – количество символов в удаляемой подстроке.

Пример: StrRemove('abbaa';3;2) = ’aba’.

Функция StrReplace

Назначение: Заменяет все вхождения в строку одной подстроки на другую.

Синтаксис: **StrReplace(Где;Что;НаЧто).**

Сигнатура: **строка** *StrReplace*(**строка** *Где;* **строка** *Что;* **строка** *НаЧто*)**.**

Параметры:

Где – исходная строка;

Что – строка, все вхождения которой в исходную строку нужно заменить;

НаЧто – строка, на которою нужно заменить все вхождения строки *Что*.

Пример: StrReplace('aabaacaa';'aa';'tt') = ’ttbttctt’.

Функция StrReplaceReg

Назначение: Заменяет все вхождения в строку подстроки заданной регулярным выражением на другую строку. Функция использует стандартный синтаксис языка регулярных выражений (как в Perl).

Синтаксис: **StrReplaceReg(Где;Что;НаЧто).**

Сигнатура: **строка** *StrReplaceReg*(**строка** *Где;* **строка** *Что;* **строка** *НаЧто*)**.**

Параметры:

Где – исходная строка;

Что – регулярный шаблон строки, все вхождения которого нужно заменить;

НаЧто – строка, на которою нужно заменить все вхождения шаблона *Что*.

Пример: StrReplaceReg('aabaacaa';'a*';'tt') = ’ttbttctt’.

Функция StrRight

Назначение: Выделение подстроки заданной длины, которой заканчивается заданная строка.

Синтаксис: **StrRight(Строка;Длина).**

Сигнатура: **строка** StrRight(**строка** *Строка;* **целое** *Длина*).

Параметры:

Строка – исходная строка;

Длина – длина выделяемой подстроки.

Пример: StrRight('aabab';2) = ’ab’.

Функция StrRTrim

Назначение: Удаляет все пробелы с конца заданной строки.

Синтаксис: **StrRTrim(Строка).**

Сигнатура: **строка** *StrRTrim***(строка** *Строка***).**

Параметр: *Строка* – строка, из которой удаляются пробелы.

Пример: StrRTrim('aaa ') = ’aaa’.

Функция StrTrim

Назначение: Удаляет все пробелы с начала и с конца заданной строки.

Синтаксис: **StrTrim(Строка)**

Сигнатура: **строка** *StrTrim***(строка** *Строка***)**

Параметры: *Строка* – строка, из которой удаляются пробелы.

Пример: StrTrim(' aaa ') = ’aaa’

Функция StrUCase

Назначение: Перевод всех символов строки в верхний регистр (строчные буквы переводятся в заглавные).

Синтаксис: **StrUCase(Строка).**

Сигнатура: **строка** StrUCase(**строка** *Строка*).

Параметр: *Строка* – исходная строка.

Пример: StrUCase('aaa') = ’AAA’.

4.13. Ключевые слова

При разработке проекта все ключевые слова можно найти в списке функций в группе «Ключевые слова». Все ключевые слова вызываются без параметров.

Константа Пустой

Английское имя: Void.

Назначение: Задает пустое значение типа «пустой» (void).

Переменная Расчет

Английское имя: Calc.

Назначение: Служит для использования в управляющем выражении значения расчетного выражения. Подробнее в разделе 3.9.

Переменная Ручной

Английское имя: Hand.

Назначение: Переменная может быть использована только в параметрах ручного ввода. Имеет значение, совпадающее со значением ручного ввода для данного параметра. Возвращаемое значение имеет тип, указанный в поле «Тип параметра». Подробнее в разделе 4.8.

4.14. Функции работы с сигналами и параметрами

Функция Владелец

Английское имя: Owner.

Назначение: Используется только в подпараметрах. Возвращает значение владельца текущего рассчитываемого подпараметра. От владельца также могут быть получены характеристики при помощи функций ПолучитьКод, ПолучитьИмя и т.д.

Функция Вызывающий

Английское имя: Caller.

Назначение: Используется только в расчетных параметрах-функциях со входами. Возвращает расчетный параметр, из которого был вызван расчет текущего параметра. Значение вызывающего параметра не может быть использовано, однако от вызывающего параметра могут быть получены характеристики при помощи функций ПолучитьКод, ПолучитьИмя и т. д.

Функция ПолучитьЕдиницы

Английское имя: TakeUnits.

Назначение: Возвращает идиницы измерения указанного сигнала или расчетного параметра.

Синтаксис: **ПолучитьЕдиницы([СигналИлиПараметр])**

Возможные сигнатуры:

строка ПолучитьЕдиницы(**вариант** *СигналИлиПараметр*);

строка ПолучитьЕдиницы.

Параметр: *СигналИлиПараметр* – сигнал или расчетный параметр, или переменная, содержащая значение сигнала или расчетного параметра. Также возможно использование сигналов и параметров, переданных в текущий параметр-функцию в качестве входов, и параметров, полученных применением функций Владелец и Вызывающий. Если *СигналИлиПараметр* не указан, то функция возвращает единицы измерения текущего вычисляемого параметра.

Пример: A={Сигнал}:ПолучитьЕдиницы(A).

Функция ПолучитьЗадачу

Английское имя: TakeTask.

Назначение: Возвращает задачу, в которой содержится указанный расчетный параметр.

Синтаксис: **ПолучитьЗадачу([Параметр]).**

Возможные сигнатуры:

строка ПолучитьЗадачу(**вариант** *Параметр*);

строка ПолучитьЗадачу.

Параметр: *Параметр* – расчетный параметр или переменная, содержащая значение расчетного параметра. Также возможно использование параметров переданных в текущий параметр-функцию в качестве входов, и параметров полученных применением функций Владелец и Вызывающий. Если *Параметр* не указан, то функция возвращает единицы измерения текущего вычисляемого параметра.

Пример: ПолучитьЕдиницы(Параметр).

Функция ПолучитьИмя

Английское имя: TakeName.

Назначение: Возвращает имя объекта для указанного сигнала или имя указанного расчетного параметра.

Синтаксис: **ПолучитьИмя([СигналИлиПараметр])**

Возможные сигнатуры:

строка ПолучитьИмя(**вариант** *СигналИлиПараметр*);

строка ПолучитьИмя.

Параметр: *СигналИлиПараметр* – сигнал или расчетный параметр, или переменная, содержащая значение сигнала или параметра. Также возможно использование сигналов и параметров, переданных в текущий параметр-функцию в качестве входов, и параметров полученных применением функций Владелец и Вызывающий. Если *СигналИлиПараметр* не указан, то функция возвращает имя текущего вычисляемого параметра.

Пример: А={Сигнал}:ПолучитьИмя(А).

Функция ПолучитьИмяСигнала

Английское имя: TakeNameSignal.

Назначение: Возвращает для указанного сигнала имя, не влючающее имя объекта.

Синтаксис: **ПолучитьИмяСигнала(Сигнал).**

Возможная сигнатура: **строка** ПолучитьИмяСигнала(**вариант** *Сигнал*).

Параметр: *Сигнал* – сигнал или переменная, содержащая значение сигнала. Также возможно использование сигналов, переданных в текущий параметр-функцию в качестве входов.

Пример: А={Сигнал}:ПолучитьИмяСигнала(А).

Функция ПолучитьКод

Английское имя: TakeCode.

Назначение: Возвращает код объекта для указанного сигнала или код указанного расчетного параметра.

Синтаксис: **ПолучитьКод([СигналИлиПараметр])**

Возможные сигнатуры:

строка ПолучитьКод(**вариант** *СигналИлиПараметр*);

строка ПолучитьКод.

Параметр: *СигналИлиПараметр* – сигнал или расчетный параметр, или переменная, содержащая значение сигнала или параметра. Также возможно использование сигналов и параметров, переданных в текущий параметр-функцию в качестве входов, и параметров полученных применением функций Владелец и Вызывающий. Если *СигналИлиПараметр* не указан, то функция возвращает имя текущего вычисляемого параметра.

Пример: А={Сигнал}:ПолучитьКод(А).

Функция ПолучитьКодСигнала

Английское имя: TakeCodeSignal.

Назначение: Возвращает для указанного сигнала код, не влючающий код объекта.

Синтаксис: **ПолучитьКодСигнала(Сигнал).**

Возможная сигнатура: **строка** ПолучитьКодСигнала(**вариант** *Сигнал*).

Параметр: *Сигнал* – сигнал или переменная, содержащая значение сигнала. Также возможно использование сигналов, переданных в текущий параметр-функцию в качестве входов.

Пример: А={Сигнал}:ПолучитьКодСигнала(А).

Функция ПолучитьКомментарий

Английское имя: TakeComment.

Назначение: Возвращает комментарий к указанному расчетному параметру.

Синтаксис: **ПолучитьКомментарий([Параметр])**

Возможные сигнатуры:

строка ПолучитьКомментарий (**вариант** *Параметр*);

строка ПолучитьКомментарий.

Параметр: *Параметр* – расчетный параметр или переменная, содержащая значение расчетного параметра. Также возможно использование параметров переданных в текущий параметр-функцию в качестве входов, и параметров полученных применением функций Владелец и Вызывающий. Если *Параметр* не указан, то функция возвращает комментарий к текущему вычисляемому параметру.

Функция Сигнал

Английское имя: Signal.

Назначение: Возваращает сигнал с указанным именем от указанного объекта.

Синтаксис: **Сигнал(Объект;КодСигнала).**

Сигнатура: **вариант** Сигнал(**вариант** *Объект*; **строка** *КодСигнала*).

Параметры:

Объект – сигнал или переменная, содержащая значение сигнала, объект которого будет использован в данной функции;

КодСигнала – строка, задающая код сигнала, который будет возвращен функцией. В результате функция возвращает сигнал с кодом *КодСигнала* от объекта сигнала , заданного праметром *Объект*.

Пример: Сигнал({KKS};'Пар') – возвращает сигнал с кодом «Пар» от объекта с кодом «KKS».

Можно объявить входом параметра-функции объект, содержащий указанный набор сигналов. В этом случае, в расчетном выражении можно использовать сигналы из перечисленного набора, используя функцию Сигнал. Такой параметр-функцию можно вызывать от различных объектов, содержащих перечисленные во входе сигналы.

Описание входа, задающего объект, выглядит следующим образом:

<**Вход>**:=[< **Тип данных**>]<**Код входа**> **Сигнал (<Сигнал>;<Сигнал>;…;<Сигнал>)**,

где <**Сигнал>**:=[< **Тип данных**>]<**Код входа**>.

При вызове такого параметра, в качестве входа должен передаваться сигнал, тип данных которого совпадает с указанным типом данных, объект которого содержит все сигналы, перечисленные в скобках, и типы данных сигналов объекта подходят под типы перечисленных в скобках сигналов.

Например, пусть в проекте есть следующие сигналы с указанием, какие из них являются сигналами по-умолчанию:

Код объекта	Код сигнала	Тип данных	По умолчанию
Ob1	SigA	целое	да
Ob1	SigB	действ	
Ob2	SigA	логич	да
Ob2	SigB	действ	
Ob2	SigC	целое	
Ob3	SigA	действ	да

В этом случае допустимо использование следующего набора расчетных параметров:

Код	Входы	Расчетное выражение
Fun	целое X Сигнал (целое SigA; действ SigB)	Сигнал(X; SigA) + Сигнал(X; SigB)
Par1		Fun({Ob1.SigA})
Par2		Fun({Ob2})

Параметр Fun имеет один вход X, в качестве которого может передаваться сигнал, объект которого содержит сигналы SigA и SigB. Причем тип самого сигнала должен сводиться к целому, тип сигнала SigA должен сводиться к целому, а тип сигнала SigB, должен сводиться к действительному. В формуле параметра Fun при помощи функции Сигнал производится выделение из переданного объекта сигналов SigA и SigB, а затем производится их сложение.

Параметр Par1 в своей формуле вызывает Fun от сигнала Ob1.SigA. В результате получится сумма сигналов Ob1.SigA и Ob1.SibB. Параметр Par2 в своей формуле вызывает Fun от объекта Ob2. В результате получится сумма сигналов Ob2.SigA и Ob2.SibB. В то же время, вызов Fun({Ob3}) вызвал бы ошибку, так как объект Ob3 не имеет сигнала SigB.

Функция СигналДейств

Английское имя: SignalReal.

Назначение: Возвращает сигнал по указанному полному коду сигнала или коду объекта. Указанный сигнал должен иметь тип данных, сводимый к действительному типу. Чтобы функция сработала, указанный сигнал уже должен использоваться в расчете в другом месте.

Синтаксис: **СигналДейств(КодСигнала).**

Сигнатура: **действ** СигналДейств(**строка** *КодСигнала*).

Параметр: *КодСигнала* – полный код сигнала или код объекта. Если указан код объета, то функция вернет сигнал по умолчанию для этого объекта.

Пример: СигналДейств(‘KKS’).

Функция СигналЛогич

Английское имя: SignalBool.

Назначение: Возвращает сигнал по указанному полному коду сигнала или коду объекта. Указанный сигнал должен иметь логический тип данных. Чтобы функция сработала, указанный сигнал уже должен использоваться в расчете в другом месте.

Синтаксис: **СигналЛогич(КодСигнала).**

Сигнатура: **логич** СигналЛогич(**строка** *КодСигнала*).

Параметр: *КодСигнала* – полный код сигнала или код объекта. Если указан код объета, то функция вернет сигнал по умолчанию для этого объекта.

Пример: СигналЛогич(‘KKS’).

Функция СигналСтрока

Английское имя: SignalString.

Назначение: Возвращает сигнал по указанному полному коду сигнала или коду объекта. Указанный сигнал должен иметь тип данных, сводимый к строковому типу. Чтобы функция сработала, указанный сигнал уже должен использоваться в расчете в другом месте.

Синтаксис: **СигналСтрока(КодСигнала).**

Сигнатура: **действ** СигналСтрока(**строка** *КодСигнала*).

Параметры: *КодСигнала* – полный код сигнала или код объекта. Если указан код объета, то функция вернет сигнал по умолчанию для этого объекта.

Пример: СигналСтрока(‘KKS’).

Функция СигналЦелое

Английское имя: SignalInt.

Назначение: Возвращает сигнал по указанному полному коду сигнала или коду объекта. Указанный сигнал должен иметь тип данных, сводимый к целому типу. Чтобы функция сработала, указанный сигнал уже должен использоваться в расчете в другом месте.

Синтаксис: **СигналЦелое(КодСигнала).**

Сигнатура: **действ** СигналЦелое(**строка** *КодСигнала*).

Параметр: *КодСигнала* – полный код сигнала или код объекта. Если указан код объета, то функция вернет сигнал по умолчанию для этого объекта.

Пример: СигналЦелое(‘KKS’).

4.15. Условия и циклы

Данный раздел описывает группу функций «Условия и циклы». В отличие от обычных функций, в аргументах функций условий и циклов можно использовать присваивание переменных и т.п.

Функция Если

Английское имя: If.

Назначение: Вычисляет одно из указанных выражений или выполняет одно из указанных действий в зависисмости от заданных условий. Условия могут иметь как постоянные значения на периоде расчета, так и изменяться со временем. Соответственно, результат функции на разных временных промежутках может принимать значения разных аргументов.

Синтаксис: **Если(Условие;Значение;[Условие];[Значения];…;[ЗначениеИначе]).**

Возможные сигнатуры:

вариант Если(**логич** *Условие1*; **вариант** *Значение1*; **логич** *Условие2*; **вариант** *Значение2*;..., [**вариант** *ЗначениеИначе*]);

пустой Если(**логич** *Условие1*; **пустой** *Действие1*; **логич** *Условие2*; **пустой** *Действие2*;..., [**пустой** *ДействиеИначе*]).

Параметры:

Условие1, Условие2 – условия, от которых зависит результат;

Значение1 – задает мгновенные значения результата на промежутках выполнения Условия1;

Значение2 – задает мгновенные значения результата на промежутках выполнения Условия2 и т. д.

ЗначениеИначе – задает мгновенные значения результата на промежутках, на которых не выполняется ни одно из условий;

Действие1 – задает действия, которые нужно выполнить на промежутках выполнения Условия1. Действие представляет собой произвольное выражение и может включать в себя присвоение переменных, а также использование операторов условий и циклов;

Действие2 – задает действия, которые нужно выполнить на промежутках выполнения Условия2 и т. д.;

ДействиеИначе – задает действия, которые нужно выполнить на промежутках, на которых не выполняется ни одно из условий.

Примеры:

Если({S}>0;1;2);

Если({S}<0;0;{S}<1;1;{S}<2;2;10)+1;

Если({S}<0;x=3:y=2;{S}>=0;x=4:y=3).

Примеры конкретных вычислений значений функции в подразделе «Пример вычисления условий».

Функция ЕслиТочки

Английское имя: IfPoints.

Назначение: Вычисляет одно из указанных выражений или выполняет одно из указанных действий в зависимости от заданных условий. Условия накладываются только по временам мгновенных значений указанных выражений, мгновенные значения на моменты изменения истинности условий не добавляются в список мгновенных значений результата.

Функцию целесообразно использовать для фильтрации мгновенных значений, составляющих списки некоторых событий. В этом случае функция накладывающая условие не должна добавлять своих точек в результат наложения условия, как это сделала бы функция «Если».

Синтаксис: **ЕслиТочки(Условие;Значение;[Условие];[Значения];…;[ЗначениеИначе]).**

Возможные сигнатуры:

Вариант ЕслиТочки(**Логич** *Условие1*; **Вариант** *Значение1*; **Логич** *Условие2*; **Вариант** *Значение2*;...; [**Вариант** *ЗначениеИначе*]);

Пустой ЕслиТочки(**Логич** *Условие1*; **Пустой** *Действие1*; **Логич** *Условие2*; **Пустой** *Действие2*;...; [**Пустой** *ДействиеИначе*]).

Параметры:

Условие1, Условие2 – условия, от которых зависит результат;

Значение1 – задает мгновенные значения результата, на промежутках выполнения Условия1;

Значение2 – задает мгновенные значения результата, на промежутках выполнения Условия2 и т. д.

ЗначениеИначе – задает мгновенные значения результата, на промежутках, на которых не выполняется ни одно из условий;

Действие1 – задает действия, которые нужно выполнить на промежутках выполнения Условия1;

Действие2 – задает действия, которые нужно выполнить на промежутках выполнения Условия2 и т. д;

ДействиеИначе – задает действия, которые нужно выполнить на промежутках, на которых не выполняется ни одно из условий.

Примеры:

ЕслиТочки({S}<0;-{T};{S}<10;{T};2*{T});

ЕслиТочки({S}<10;x={T};{S}>=10;x=2*{T}).

Пример вычисления условий

Пусть в проекте есть объекты «S» и «T», сигналы по умолчанию для которых являются целочисленными. Пусть в проекте есть следующие расчетные параметры:

Код	Формула
A	Если(2>3;1;2)
B	Если({S}>0;1;2)+1
C	Если({S}<0;0;{S}<10;{T};{S}<20;2*{T};0)
D	Если({S}<10; x=1:y={T};{S}>=10; x=0:y=2*{T}) : x+y
E	ЕслиТочки({S}<0;-{T};{S}<10;{T};{S}<20;2*{T};3*{T})
F	ЕслиТочки({S}<10; x={T};{S}>=10; x=2*{T}): x

Пусть расчет производится за период от 00:00:00 до 01:00:00, и на этом промежутке времени значения сигналов {S} и {T} изменялись следующим образом:

Время	{S}	{T}
00:00:00	5	1
00:10:00	-13	
00:15:00	24	2
00:20:00	18	
00:25:00		3
00:30:00	7	
00:40:00	-3	
00:45:00		4
00:55:00	19	

В этом случае параметры «A», «B», «C», «D», «E» и «F», а также переменные x и y параметра «D» будут иметь следующие мгновенные значения:

Время	A	B	C	D.x	D.y	D	E	F
00:00:00	2	2	1	1	1	2	1	1
00:10:00		3	0					
00:15:00		2		0	4	4	6	4
00:20:00			4					
00:25:00			6		6	6	6	6
00:30:00		2	3	1	3	4		
00:40:00		3	0					
00:45:00					4	5	-4	4
00:55:00		2	8	0	8	8		

Условие в параметре «A» не выполняется на всем промежутке расчета, второй и третий аргументы функции «Если» также постоянны, поэтому результатом является число, не зависящее от времени, которому для накопления в архив приписывается время, равное началу периода.

Истинность условия в параметре «B» изменяется со временем, поэтому на выходе функции «Если» будет список мгновенных значений, состоящий из точек, времена которых совпадают с временами мгновенных значений условия (которые в свою очередь совпадают с временами мгновенных значений сигнала {S}), а значения равны то второму, то третьему аргументу функции «Если», в зависимости от истинности условия.

Параметр «C» отличается от параметра «B» тем, что формирующие результата аргументы функции «Если» также меняются со временем. Поэтому мгновенное значение результата изменяется как в моменты изменения истинности условий, так и в моменты изменения значений аргументов.

Функция «Если» в параметре «D» не возвращает значения, результат имеет тип «пустой». Зато в зависимости от истинности условий, переменным x и y присваиваются разные значения на разных временных промежутках. Результатом вычисления всего параметра является сумма переменных x и y.

В параметре «E» используется функция «ЕслиТочки», поэтому в результат не добавляются точки перемены истинности условий, а добавляются только точки соответствующие мгновенным значениям других аргументов. Конечно, при этом точки отбираются не все, а только попадающие во временной промежуток истинности соответствующих условий.

Параметр «F» вычисляется аналогично, только в нем функция «ЕслиТочки» сама не возвращает значение, зато присваивает значение переменной x, которая потом и возвращается как результат вычисления всего параметра.

Функция Пока

Английское имя: While.

Назначение: Многократно выполняет некоторые действия (присвоения, другие циклы и т. п.), пока выполняется заданное условие.

По аналогии с функцией «Если», условие функции «Пока» может быть постоянным на всем периоде расчета, а может и не быть. В общем случае функция выполняется до тех пор, пока условие выполняется хотя бы на одном временном промежутке внутри периода расчета. После каждого цикла истинность условия проверяется повторно. При этом на каждой итерации цикла действия выполняются только для тех временных промежутков, для которых на этой итерации выполняется условие.

Синтаксис: **Пока(Условие;Действия).**

Сигнатура: **Пустой** Пока(**Логич** *Условие*; **Пустой** *Действия*).

Параметры:

Условие – условие цикла;

Действия – действия, которые нужно выполнить в цикле.

Примеры:

i=1:x=0:Пока(i<=10;x=x+i:i=i+1):x – сумма чисел от 1 до 10;

x={S}:s=x:Пока({S}>0; Если(x>1;x=x-1):s=s*x) – вычисление факториала для всех мгновенных значений сигнала {S}.

Функция ПокаТочки

Английское имя: WhilePoints.

Назначение: Многократно выполняет некоторые действия (присвоения, другие циклы и т.п.), пока выполняется заданное условие. Условия накладываются только по временам мгновенных значений указанных выражений.

Функцию целесообразно использовать для работы с мгновенными значениями, составляющими списки некоторых событий. В этом случае при проверке условия в списке событий не должно добавляться новых точек, как это произошло бы при использовании функции «Пока».

Синтаксис: **ПокаТочки(Условие;Действия).**

Сигнатура: **Пустой** ПокаТочки(**Логич** *Условие*; **Пустой** *Действия*).

Параметры:

Условие – условие цикла;

Действия – действия, которые нужно выполнить в цикле.

Пример: ПокаТочки(A<x; x=x+1).

Пример вычисления циклов

Пусть в проекте есть объекты «S» и «T», сигналы по умолчанию для которых являются целочисленными. Пусть в проекте есть следующие расчетные параметры:

Код	Формула	Смысл
A	i=1:x=0: Пока(i<=10; x=x+i: i=i+1): x	Сумма чисел от 1 до 10
B	f=1:i=1: Пока(i<=4; f=f*{S}: i=i+1): f	Возведение всех мгновенных значений сигнала в 4-ю степень

Код	Формула	Смысл
C	x={S}:f=1: Пока(x>1; f=f*x: Если(x>1; x=x-1)): f	Вычисление факториала для всех мгновенных значений сигнала {S}
D	x={T}: f={S}: ПокаТочки(x>0: f=f+1: x=x-1): f	Прибавляет {S} к {T}, оставляя точки значений только на моменты значений {S}

Пусть расчет производится за период от 00:00:00 до 01:00:00 и на этом промежутке времени значения сигналов {S} и {T} изменялись следующим образом:

Время	{S}	{T}
00:00:00	1	0
00:05:00		1
00:10:00	2	
00:15:00		2
00:20:00	3	
00:25:00		3
00:30:00	4	
00:35:00		4
00:40:00	3	
00:45:00		3
00:50:00	2	
00:55:00		2

В этом случае параметры «A», «B», «C» и «D» будут иметь следующие мгновенные значения:

Время	A	B	C	D
00:00:00	55	1	1	1
00:10:00		16	2	3
00:20:00		81	6	5
00:30:00		256	24	7
00:40:00		81	6	7
00:50:00		16	2	5

Цикл в параметре «A» не зависит от времени, поэтому в результате получается одно число, которому для накопления приписывается время начала периода расчета.

В параметре «B» вычисляется цикл, условие которого не зависит от времени, он независимо от значений сигнала {S} выполнится 4 раза и тем самым возведет каждое мгновенное значение {S} в четвертую степень.

В параметре «C» условие цикла зависит от времени. На первой итерации цикла условие выполняется на всем периоде расчета, на второй только с 00:10:00, на третьей с 00:20:00 до 00:50:00, на четвертой с 00:30:00 до 00:40:00, на пятой не выполняется нигде, поэтому цикл на пятой итерации прерывается. При этом на каждой итерации значение переменной f умножается на очередное число, которое меньше предыдущего на 1 и так пока не станет равным 1. Поэтому результатов вычисления параметра станет список значений, каждое значение из которого является факториалом соответствующего мгновенного значения сигнала {S}.

В параметре «D» используется функция «ПокаТочки». В условии x>0 используются мгновенные значения со временами мгновенных значений сигнала {T}, однако они не будут добавлены при присвоении значений переменной f. В то же время, значение переменной x постепенно станет меньше либо равным 0 и цикл завершит выполнение. При этом начальный список значений переменной f равен списку значений сигнала {S} и к каждому мгновенному значению переменной f единица будет добавлена ровно столько раз, сколько составляет величина предыдущего значения сигнала {T}.

5. СПИСОК ПРИНЯТЫХ СОКРАЩЕНИЙ

№	Сокращение	Пояснение
1.	АСУТП	автоматизированная система управления технологическими процессами
2.	ПТК	программно-технический комплекс