А1. Описание применения

УТВЕРЖДЕН

А.В.00001-01 31 01-ЛУ

ПРОГРАММА «СРЕДА СОЗДАНИЯ И МОДЕЛИРОВАНИЯ ОБЪЕКТНО-АТРИБУТНОЙ СУПЕРКОМПЬЮТЕРНОЙ СИСТЕМЫ С УПРАВЛЕНИЕМ ПОТОКОМ ДАННЫХ»

А.В.00001-01 31 01

Листов 9

2012

Подпись и дата	
Инв. № дубл.	
Взам. инв. №	
Подпись и дата	
Инв. № подл.	

АННОТАЦИЯ

В данном программном документе приведено описание применения программы «Среда программирования и имитационного моделирования объектно-атрибутной суперкомпьютерной системы с управлением потоком данных» (далее Программа), предназначенной для создания и управления работой виртуальных функциональных устройств.

В данном программном документе, в разделе «Назначение программы» приведено описание назначения программы, возможности данной программы, а также ее основные характеристики и ограничения, накладываемые на область применения программы.

В разделе «Условия применения» указаны условия, необходимые для выполнения программы (требования к необходимым для данной программы техническим средствам, и другим программам, общие характеристики входной и выходной информации, а также требования и условия организационного, технического и технологического характера).

В данном программном документе, в разделе «Описание задачи» указаны определения задачи и методы ее решения.

В разделе «Входные и выходные данные» указаны сведения о входных и выходных данных.

Оформление программного документа «Описание программы» произведено по требованиям ЕСПД ГОСТ 19.502-78[1].

[1] ГОСТ 19.502-78 ЕСПД. Описание применения. Требования к содержанию и оформлению

СОДЕРЖАНИЕ

ГЛОССАРИЙ

Информационная пара (ИП) (атрибутированные данные) – совокупность нагрузки (данных или ссылки на данные), и ярлыка (атрибута/уникального идентификатора), идентифицирующего нагрузку. В нагрузке ИП может храниться не только константа, но и указатель на ячейку памяти. Указатель, хранящийся в нагрузке ИП, может ссылаться на информационные конструкции любой сложности (переменные, массивы, списки, другие ИП и т.д.).

Виртуальное функциональное устройство (ВФУ) – это виртуальное (реализованное программным способом) функциональное устройство, осуществляющее обработку данных. ВФУ имеет внутренние виртуальные регистры (набор регистров будем именовать контекстом ВФУ), используемые для хранения промежуточных данных, и может исполнять некий набор милликоманд, описываемый алгоритмом функционирования этого ВФУ. ВФУ состоит из контекста (набора виртуальных регистров: как правило, контекст реализуется с помощью структуры (записи) в языке высокого уровня) и процедуры реализации логики работы ВФУ, у которой имеется универсальный интерфейс.

Милликоманда (мК) – это ИП, где ярлык указывает ВФУ-ву, каким образом следует обрабатывать прикрепленную к нему нагрузку, и задает тип данных, хранящихся в нагрузке.

Капсула – это обособленный набор информационных пар, служащих для описания определенного объекта (с помощью капсулы обеспечивается абстракция данных). Каждая ИП, входящая в капсулу, задает один из критериев описываемого объекта.

Шина данных-атрибута (ШДА) – виртуальная шина, по которой ФУ обмениваются милликомандами между собой. ШДА в виртуальной реализации представляет собой ВФУ, специализирующийся на передаче данных между ВФУ.

ОА-платформа — совокупность программ реализации логики работы ВФУ, загруженных на вычислительный узел. ОА-платформа зависима от аппаратной архитектуры вычислительного узла.

ОА-образ — состояние контекстов ВФУ и синтезированные в данный момент ОА-информационные конструкции (переменные, капсулы, ОА-деревья). ОА-образ независим от аппаратной реализации вычислительных узлов ОА-системы.

1 НАЗНАЧЕНИЕ ПРОГРАММЫ

1.1 Назначение программы

Программа «СРЕДА СОЗДАНИЯ И МОДЕЛИРОВАНИЯ ОБЪЕКТНО-АТРИБУТНОЙ СУПЕРКОМПЬЮТЕРНОЙ СИСТЕМЫ С УПРАВЛЕНИЕМ ПОТОКОМ ДАННЫХ» предназначена для моделирования виртуальной модели суперкомпьютера объектно-атрибутной (ОА) архитектуры, с целью проверки работоспособности заданных с помощью специального языка программирования алгоритмов и проверки параметров выполняемого на суперкомпьютере вычислительного процесса (время выполнения программы, требуемые аппаратные ресурсы, нагрузку на линии передачи данных и т.д.).

Программа состоит из нескольких функциональных блоков:

- среда программирования (интерфейс пользователя): текстовые консоли для ввода программы и ввода/вывода данных для модели суперкомпьютера, инструментальные средства, облегчающие процесс программирования.

- компилятор языка программирования для суперкомпьютера (осуществляет перевод программы во внутреннее представление для модели суперкомпьютера и запуск программы на модели суперкомпьютера);

- объектно-атрибутная платформа: совокупность виртуальных устройств, эмулирующих работу реальных устройств суперкомпьютера;

- средства моделирования вычислительного процесса: виртуальные устройства, позволяющие отслеживать параметры моделируемого вычислительного процесса.

1.2 Возможности программы

Программа позволяет:

- создавать и уничтожать ВФУ;

- решать вычислительные задачи с помощью описания обмена данными между ВФУ (обмен между ВФУ описывается с помощью специального языка программирования);

- программировать ВФУ и задавать ОА-информационные конструкции с помощь специального ОА-языка программирования;

- запускать программы, написанные на ОА-языке программирования;

- запускать тестовые примеры (маленькие ОА-программы и наборы данных);

- считывать/записывать ОА-программы и тестовые примеры в/из файла;

- производить моделирование вычислительного процесса, запущенного на программной модели суперкомпьютера ОА-архитектуры.

1.3 Основные характеристики программы

В программе реализованы боле 70-ти типов ВФУ, предназначенных для организации вычислительного процесса обеспечения работы компилятора моделирования вычислительного процесса, и эмуляции работы блоков суперкомпьютера ОА-архитектуры.

1.4. Ограничения, накладываемые на область применения программы

Программа может использоваться для создания и исполнения программ и моделирования вычислительной системы ОА-архитектуры и не предназначена для моделирования вычислительных систем других архитектур. Программа создана для работы под управлением операционной системы Windows (Windows-2000, Windows XP, Windows Vista, Windows 7), работа под управлением других операционных систем не предусмотрена.

2 УСЛОВИЯ ПРИМЕНЕНИЯ

2.1 Минимальный состав технических средств

В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ), включающий в себя:

- процессор с тактовой частотой, ГГц - 1, не менее;
- операционную систему Windows XP, Windows Vista, Windows 7
- оперативную память объемом, Мб-512, не менее;

2.2 Минимальный состав программных средств

Системные программные средства, используемые программой, должны быть представлены операционной системой Windows XP (Windows Vista, Windows).

Также для реализации всех возможностей программы требуется предустановленный модуль OpenGL.

3 ОПИСАНИЕ ЗАДАЧИ

3.1. Определение задачи

Основная задача, решаемая программой – моделирование вычислительного процесса, запущенного на суперкомпьютерной вычислительной системе объектно-атрибутной архитектуры. Моделирование вычислительного процесса позволяет найти наиболее удачную архитектуру будущей системы и наиболее оптимальные ее параметры, что позволит улучшить характеристики будущей вычислительной системы, выбрать наиболее оптимальные режимы работы вычислительной системы.

3.2. Методы решения задачи

Среда создания и запуска объектно-атрибутных образов представляет собой ПО, реализующее принципы ОА-архитектуры и обеспечивает следующие возможности:

- создание ОА-образа;

- запуск ОА-образа;

- самостоятельное создание ВФУ;

- управление ОА-системой в реальном времени (т.е. изменение параметров ВФУ и ОА-образа без перезагрузки системы);

- просмотр ОА-деревьев, синтезированных в процессе выполнения ОА-образа;

- имитационное моделирование параллельного вычислительного процесса.

Данное ПО состоит из:

- ОА-платформы (процедуры, реализующие логику работы для применяемых типов ВФУ);

- компилятора ОА-языка (реализованного на разработанной ОА-платформе);

- системы автоматической загрузки ОА-образа по узлам распределенной вычислительной системы;

- инструментальных средств разработки ОА-образа: рабочая панель проектирования ОА-образа (окна с перечнем участвующих в вычислительном процессе ВФУ, указателей, переменных, констант и атрибутов); панель инструментов (окно вывода результатов выполнения программы, окно служебных сообщений, окно редактора ОА-образа и тестовых примеров).

4 ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

4.1 Сведения о входных данных

Входные данные оформляются в виде языковых конструкций на объектно-атрибутном языке.

Компилятор языка среды создания и запуска ОА-образа выполнен на базе ОА-архитектуры и реализует следующие языковые конструкции:

1. Константы

В ОА-языке могут использоваться константы следующих типов:

- символ (обозначается с помощью знака «’», например, ’a’);

- строка (обозначается с помощью знака «"», например, "abc");

- логическая константа («истина», «true», «ложь», «false»);

- целое число;

- дробное число (чтобы компилятор воспринял число как дробное, в нем обязательно должно присутствовать обозначение дробной части: например, 234.0).

2. Атрибуты

Атрибут можно задать двумя способами. Во-первых, можно в качестве атрибута задать конкретное число (задается с помощью знака "*" после мнемоники атрибута): Mnemo*2. Во-вторых, автоматически присвоить значение атрибуту может компилятор, для этого в текста ОА-программы должна присутствовать отдельная мнемоника: Mnemo.

3. Переменные

Переменные объявляются с помощью знака «=», например, выражение Variable=10 является объявлением переменной Variable и присвоением ей начального значения равного 10.

Переменные могут быть двух видов:

- числовые/символьные (могут принимать один из пяти вышеперечисленных для констант типов), переменная считается числовой/символьной в том случае, если при ее объявлении в качестве начального значения выступает константа;

- указатели (ссылки), переменная считается указателем, если в качестве начального значения указывается ссылка, например, Variable2=Variable; для обозначения нулевой ссылки в ОА языке применяются мнемоники «nil» или «нуль»: Variable2=nil.

4. Информационная пара

ИП описывается с помощью знака «=»: перед «=» стоит атрибут ИП, после – нагрузка: Mnemo="Variable". В качестве нагрузки ИП могут выступать как константы, так и ссылки: Var=Variable.

5. Милликоманда

Милликоманда указывается в качестве атрибута ИП и состоит из двух частей: мнемоника ВФУ, которому милликоманда должна быть передана; мнемоника милликоманды. Эти две части отделаются одна от другой с помощью знака «.».

6. Информационная капсула

ИП группируются в капсулу с помощью знаков «{» и «}»: Caplsule{Mnemo="abc" Mnemo="xyz"} (перед знаком «{» стоит мнемоника указателя на капсулу, ИП разделяются между собой пробелом или знаком «,»).

8. Комментарии

Текст комментариев в ОА-программировании оформляется с помощью знаков * Комментарии *\, также знаком комментария являются символы \\ - действие этого комментария распространяется до конца строки.

4.2 Сведения о выходных данных

Выходными данными является текстовый файл, формируемый в процессе выполнения программы. Кроме того, для вывода могут быть использованы стандартные VCL-компоненты,

входящие в состав среды Borland Delphi (оформлены в виде функциональных устройств).

Лист регистрации изменений

Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Входящий № сопроводит. докум. и дата	Подп.	Дата
измененных	замененных	новых	аннулированных					

А2 Руководство системного программиста

УТВЕРЖДЕН

А.В.00001-01 32 01-ЛУ

ПРОГРАММА «СРЕДА СОЗДАНИЯ И МОДЕЛИРОВАНИЯ ОБЪЕКТНО-АТРИБУТНОЙ СУПЕРКОМПЬЮТЕРНОЙ СИСТЕМЫ С УПРАВЛЕНИЕМ ПОТОКОМ ДАННЫХ»

А.В.00001-01 32 01

Листов 17

2012

Инв. № подл.	Подпись и дата	Взам. инв. №	Инв. № дубл.	Подпись и дата

АННОТАЦИЯ

В данном программном документе приведено руководство системного программиста по настройке и использованию программы «Среда программирования и имитационного моделирования объектно-атрибутной суперкомпьютерной системы с управлением потоком данных» (далее Программа), предназначенной для создания и управления работой виртуальных функциональных устройств.

В данном программном документе, в разделе «Общие сведения о программе» указаны назначение и функции программы и сведения о технических и программных средствах, обеспечивающих выполнение данной программы, а также требования к персоналу.

В разделе «Структура программы» приведены сведения о структуре программы, ее составных частях и о связях между составными частями.

В данном программном документе, в разделе «Настройка программы» приведено описание действий по настройке программы на условия конкретного применения.

В разделе «Проверка программы» приведено описание способов проверки, позволяющих дать общее заключение о работоспособности программы (контрольные примеры, методы прогона, результаты).

В данном программном документе, в разделе «Сообщения системному программисту» указаны тексты сообщений, выдаваемых в ходе выполнения настройки, проверки программы, а также в ходе выполнения программы, описание их содержания и действий, которые необходимо предпринять по этим сообщениям.

Оформление программного документа «Руководство системного программиста» произведено по требованиям ЕСПД ГОСТ 19.503-79[2].

[2] ГОСТ 19.503-79 ЕСПД. Руководство системного программиста. Требования к содержанию и оформлению

СОДЕРЖАНИЕ

ГЛОССАРИЙ

Информационная пара (ИП) (атрибутированные данные) – совокупность нагрузки (данных или ссылки на данные), и ярлыка (атрибута/уникального идентификатора), идентифицирующего нагрузку. В нагрузке ИП может храниться не только константа, но и указатель на ячейку памяти. Указатель, хранящийся в нагрузке ИП, может ссылаться на информационные конструкции любой сложности (переменные, массивы, списки, другие ИП и т.д.).

Виртуальное функциональное устройство (ВФУ) – это виртуальное (реализованное программным способом) функциональное устройство, осуществляющее обработку данных. ВФУ имеет внутренние виртуальные регистры (набор регистров будем именовать контекстом ВФУ), используемые для хранения промежуточных данных, и может исполнять некий набор милликоманд, описываемый алгоритмом функционирования этого ВФУ. ВФУ состоит из контекста (набора виртуальных регистров: как правило, контекст реализуется с помощью структуры (записи) в языке высокого уровня) и процедуры реализации логики работы ВФУ, у которой имеется универсальный интерфейс.

Милликоманда (мК) – это ИП, где ярлык указывает ВФУ-ву, каким образом следует обрабатывать прикрепленную к нему нагрузку, и задает тип данных, хранящихся в нагрузке.

Капсула – это обособленный набор информационных пар, служащих для описания определенного объекта (с помощью капсулы обеспечивается абстракция данных). Каждая ИП, входящая в капсулу, задает один из критериев описываемого объекта.

Шина данных-атрибута (ШДА) – виртуальная шина, по которой ФУ обмениваются милликомандами между собой. ШДА в виртуальной реализации представляет собой ВФУ, специализирующийся на передаче данных между ВФУ.

ОА-платформа — совокупность программ реализации логики работы ВФУ, загруженных на вычислительный узел. ОА-платформа зависима от аппаратной архитектуры вычислительного узла.

ОА-образ — состояние контекстов ВФУ и синтезированные в данный момент ОА-информационные конструкции (переменные, капсулы, ОА-деревья). ОА-образ независим от аппаратной реализации вычислительных узлов ОА-системы.

Контекст ВФУ – это запись (структура), полями которой являются переменные, выполняющие роль виртуальных регистров.

1 ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ

1.1 Назначение программы

Программа предназначена для создания и управления работой виртуальных функциональных устройств. Программа может использоваться для создания и исполнения программ и моделирования вычислительной системы ОА- архитектуры.

1.2 Функции программы

1) Создание виртуальных вычислительных систем ОА-архитектуры
2) Реализация алгоритмов для систем с управлением потоком данных
3) Управление функциональными устройствами
4) Создание виртуальных вычислительных устройств
5) Формирование индексного файла (предварительная компиляция ОА-программы для последующего запуска)

1.3 Минимальный состав технических средств

В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ), включающий в себя:

- процессор с тактовой частотой, ГГц - 1, не менее;
- оперативную память объемом, Мб-512, не менее;

1.4 Минимальный состав программных средств

Системные программные средства, используемые средой создания и выполнения ОА-образов, должны быть представлены операционной системой Windows XP (Windows Vista, Windows).

Также для реализации всех возможностей программы требуется предустановленный модуль OpenGL.

1.5 Требования к персоналу (системному программисту)

Системный программист должен иметь минимум среднее техническое образование и опыт программирования на языках высокого уровня.

В перечень задач, выполняемых системным программистом, должны входить:

а) задача поддержания работоспособности технических средств;

б) задача установки (инсталляции) и поддержания работоспособности системных программных средств – операционной системы;

в) задача установки (инсталляции), настройки и поддержания работоспособности среды создания и выполнения ОА-образов;

г) создание программ с помощью ОА-среды.

2 СТРУКТУРА ПРОГРАММЫ

2.1 Сведения о структуре программы

Программа состоит из:

- ОА-платформы (часть ОА-системы, реализующая логику работы виртуальных ВФУ) состоит из описания контекстов ВФУ и подпрограмм реализации логики работы для каждого типа ВФУ;

- компилятора ОА-языка (реализованного на разработанной ОА-платформе);

- системы автоматической загрузки ОА-образа по узлам распределенной вычислительной системы;

- инструментальных средств разработки ОА-образа: рабочая панель проектирования ОА-образа (окна с перечнем участвующих в вычислительном процессе ВФУ, указателей, переменных, констант и атрибутов); панель инструментов (окно вывода результатов выполнения программы, окно служебных сообщений, окно редактора ОА-образа и тестовых примеров).

2.2 Сведения о взаимодействии между функциональными устройствами программы

ОА-платформа состоит из описания контекстов ВФУ и подпрограмм реализации логики работы для каждого типа ВФУ.

ВФУ может передавать операнды двумя способами:

1. Напрямую ВФУ-адресату (такой вариант возможен только в едином адресном пространстве). В этом случае ВФУ-отправитель должен содержать в своем контексте ссылку на контекст ВФУ-приемника. Передача осуществляется с помощью вызова процедуры реализации логики работы ВФУ-приемника. Подпрограмма реализации логики имеет стандартный интерфейс (Рисунок 1):

```
Procedure FUProgName(context: Pointer; millicomand: int64; Load: Pointer);
```

Рисунок 1 - Подпрограмма реализации логики (миллипрограмма)

где context – ссылка на контекст ВФУ;

millicomand – индекс милликоманды, предназначенной для ВФУ;

Load – ссылка на нагрузку милликоманды;

2. Через ВФУ Шина с помощью расширенной милликоманды (возможна работа в распределенной вычислительной системе). В контекст всех ВФУ, алгоритм работы которых предполагает самостоятельную передачу милликоманд другим ВФУ, входит виртуальный регистр, где хранится ссылка на контекст Шины: ВФУ для передачи информации вызывает функцию реализации логики работы Шины и в качестве параметров передает ей милликоманду и ссылку на нагрузку милликоманды. Милликоманда, что передается для Шины, содержит расширенной индекс, который формируется по следующему правилу:

ExtendedMillicom=NFU*MilliRange+MillicomIndex,

где NFU — номер созданного ВФУ;

MilliRange — диапазон адресов милликоманд (данная величина входит в контекст Шины);

MillicomIndex — индекс милликоманды для ВФУ, которому адресуется милликоманда.

Передача милликоманд между ВФУ осуществляется Шиной следующим образом:

- Шина получает от какого-либо ВФУ милликоманду;

- для определения индекса атрибут делится на величину Диапазона милликоманд (MilliDap). Целая часть от деления будет являться индексом ВФУ;

- Шина в массиве описания созданных ВФУ находит ссылку на контекст ВФУ, индекс которого совпадает с полученным индексом ВФУ;

- Шина вызывает процедуру реализации логики работы ВФУ, (ссылка которую находится в контексте ВФУ) и в качестве параметров предаются: контекст (context) находится в массиве описания созданных ВФУ, милликоманда (millcomand) - это остаток от деления атрибута на величину диапазона милликоманд (MilliDiap), в качестве нагрузки (Load) передается нагрузка милликоманды.

2.2.1 Взаимодействие ВФУ в распределенных и параллельных вычислений

ОА-платформа способна эффективно обеспечить распределенные и параллельные вычисления, благодаря тому что разработанный программный продукт обеспечивает:

- передачу информационных ОА-конструкций по линиям связи;

- маршрутизацию сообщений;

2.2.2 Индексный режим

Для загрузки ОА-образа на удаленные вычислительные узлы и для записи ОА-образа в файл используется индексный режим (в данном режиме работают ВФУ «Шина», «Диспетчер капсул» и «Диспетчер переменных»). В этом режиме создаваемые информационные капсулы помещаются в так называемый индексный вектор, каждый элемент которого представляет собой запись (структуру), состоящую из следующих полей:

- индекс атрибута;
- нагрузка/индекс ячейки памяти;
- флаг режима нагрузки (указывает является ли значение, хранимое в нагрузке константой или ссылкой);
- индекс предыдущей ИП;
- индекс следующей ИП.

Индекс который помещается в качестве нагрузки формируется следующим образом: Index=IndexCell*NFields + Offset,

Где IndexCell — индекс ячейки индексного вектора;

NFields — число полей в описании ИП;

Offset — смещение, которое задает смещение индекса (для ссылки на ИП он равен 0, для ссылки на атрибут — 1, для ссылки на нагрузку — 2 и т.д.).

Когда индексный вектор поступает на вычислительный узел, ВФУ «Шлюз» (отвечает за прием пересылку данных по линиям связи) создает в ОП узла массив описаний ИП и производит конвертацию данных из индексного вектора в динамические ОА-конструкции: индексы заменяются ссылками на ячейки ОП.

2.2.3 Шлюзование и маршрутизация

ВФУ Шлюз осуществляет следующие функции:

- преобразование предназначенной для передачи по линии связи ОА-информационной конструкции (индексный вектор) в формат для передачи по линии связи в соответствии с определенным протоколом;

- передачу информации по линии связи;

- прием данных по линии связи;

- преобразование полученных по линии связи пакетов данных в информационные конструкции;

- передачу принятой из канала связи милликоманды ВФУ-потребителю информации.

Процедура настройки шлюза включает объявления и настройку каналов. Каналы бывают двух видов: внешние (предназначенные для обмена с другими вычислительными узлами) и внутренние (для обмена информацией с Шинами, расположенными в ОП данного

вычислительного узла). Настройка внешнего канала включает в себя установку верхнего и нижнего порогов диапазона милликоманд и запись для внешнего канала ссылки на подпрограмму реализации процедуры передачи сообщения, а для внутреннего канала — ссылки на контекст Шины. Алгоритм работы маршрутизатора следующий: получив милликоманду с какого-либо канала связи маршрутизатор проверяет в диапазон милликоманд какого из каналов попадает индекс пришедшей милликоманды вместе с нагрузкой и осуществляет отправку милликоманды с нагрузкой по этому каналу.

2.2.4 Загрузка ОА-образа по распределенным вычислительным узлам

ОА-среда обеспечивает создание ОА-образа сразу на нескольких вычислительных узлов (распределенная ОА-система).

Для разделения ОА-образа по нескольким вычислительным узлам в ОА-язык будет входить конструкция "Seg(...) … EndSeg", где в скобках указывается описание шлюзов, через которые будет происходить обмен информацией с соседними узлами; в описание шлюза входит указание протокола обмена информацией и основные параметры протокола (например, IP-адрес и IP-порт).

```
Segment{Sluice={Mnemo="SegTCP" Protocol="TCP" Address ="92.168.0.1" Port="198"}
  Sluice={Mnemo="Seg2TCP" Protocol="TCP" Address ="192.168.0.2" Port="20"}}
….....
EndSeg
Segment{Sluice={Mnemo="Seg2TCP" Protocol="TCP" Address ="192.168.0.1"
Port="198"}}
….....
EndSeg
…..
```

Рисунок 2 – Подпрограмма создания и настройки сегментов (ядер) ОА-вычислительной системы

Процесс подготовки к автоматической загрузке происходит следующим образом:

- на все узлы распределенной ВС загружается ОА-платформа;

- на каждом узле запускается ВФУ Шина;

- на каждом узле производится предварительная настройка шлюзов и маршрутизаторов, для того, чтобы обеспечить коммуникацию по линиям связи;

- компилятор на головном вычислительном узле производит компиляцию ОА-программы, во время которой формируются ОА-образы для каждого вычислительного узла распределенной системы;

- головной вычислительный узел рассылает для каждого узла системы его ОА-образ, который прикрепляется к милликоманде "выполнить ОА-образ" для ВФУ Шина.

- ВФУ Шина на каждом узле запускает переданной с головной машины ОА-образ на выполнение; в ОА-образ входят милликоманды для Шины на создание и инициализацию необходимых для работы узла ВФУ.

- после создания и инициализации необходимых ВФУ на каждом узле распределенная вычислительная система готова к работе.

2.3 Сведения о функциональных устройствах программы

2.3.1 Добавление новых ВФУ в ОА-платформу

Для работы ВФУ определенного типа программисту необходимо создать следующие процедуры и функции:

- функция инициализации ВФУ;
- процедура уничтожения (выгрузки) ВФУ;
- процедура реализации логики работы ВФУ;

Функция инициализации ВФУ

Необходима для того, чтобы ВФУ Шина (диспетчер ВФУ) или другое ВФУ могли произвести создание и инициализацию нового ВФУ. Например, процедура инициализации ВФУ Целочисленное АЛУ будет выглядеть следующим образом (Рисунок 3):

```
function IntALUIni(Parent: Pointer): Pointer;
 var Context: PIntALU;
begin
 New(Context);
 Context^.FUProgram:=@IntALU; // Установить ссылку на подпрограмму
                              // реализации логики работы ВФУ
 Context^.FUKill:=@IntALUKill;  // Установить ссылку на подпрограмму
                              // уничтожения контекста ВФУ
 Context^.accumulator:=0; // Инициализация значения в аккумуляторе
 IntALUIni:=Context;        // Возврат указателя на контекст созданного ВФУ
end;
```

Рисунок 3 - Процедура инициализации ВФУ Целочисленное АЛУ (миллипрограмма)

Процедура уничтожения (выгрузки) ВФУ

Используется для выгрузки контекста ВФУ из оперативной памяти компьютера. Параметром процедуры является ссылка на контекст ВФУ, который необходимо выгрузить. Так, для ВФУ целочисленное АЛУ данная процедура выглядит так (рис. 4):

```
procedure IntALUKill(Context: Pointer);
begin
  dispose(PIntALU(Context));
end;
```

Рисунок 4 - Процедура выгрузки ВФУ Целочисленное АЛУ (миллипрограмма)

Процедура реализации логики работы ВФУ

Процедура реализует общую логику работы ВФУ: прием милликоманд с ШДА, запись промежуточных данных в контекст ВФУ, выполнение операции по приходе всех необходимых для нее данных и т.д. Например, процедура реализации логики работы ВФУ целочисленного АЛУ будет следующей (Рисунок 5):

```
Procedure IntALU(context: Pointer; millicomand: int64; Load: Pointer);
 var IntALUContext: PIntALU;
begin
IntALUContext:=context;
 with IntALUContext^ do
  begin
   case millicomand of
    0: // Сброс
      accumulator:=0;
    1: // Записать значение в аккумулятор (если на входе nil аккумулятор=0) (Уст)
……
    2: // Выдать значение из аккумулятора
……
    3: // Выдать остаток от деления
……
    4: // Изменить знак числа в аккумуляторе и выдать его значение
……
   end;
 end
end;
```

Рисунок 5 - Процедура реализации логики ВФУ Целочисленное АЛУ (миллипрограмма)

где context – ссылка на контекст ВФУ;

millicomand – индекс милликоманды, предназначенной для ВФУ;

Load – ссылка на нагрузку милликоманды;

accumulator — аккумулятор, где хранится один из операндов и куда записывается результат выполнения операции.

2.3.2 Описание типов основных ВФУ

Описание основных типов ВФУ дается в Приложении 1 к Описанию программы.

2.4 Язык программирования для описания структур данных, работы ВФУ и их обмена информацией между собой

Компилятор языка среды создания и запуска ОА-образа, который переводит удобную для восприятия человеком символьную программу в индексы для ВФУ и формирует информационные ОА-конструкции выполнен на базе ОА-архитектуры и реализует следующие языковые конструкции:

1. Константы

В ОА-языке могут использоваться константы следующих типов:

- символ (обозначается с помощью знака «’», например, ’a’);
- строка (обозначается с помощью знака «"», например, "abc");
- логическая константа («истина», «true», «ложь», «false»);
- целое число;
- дробное число (чтобы компилятор воспринял число как дробное, в нем обязательно должно присутствовать обозначение дробной части: например, 234.0).

Компилятор ОА-языка предоставляет возможность задавать именованные константы: для этого используется знак «#», например: PI#3.1425962 (теперь вместо мнемоники PI компилятор будет подставлять число 3.1425962).

2. Атрибуты

Атрибут можно задать двумя способами. Во-первых, можно в качестве атрибута задать конкретное число (задается с помощью знака "*" после мнемоники атрибута): Mnemo*2. Во-вторых, автоматически присвоить значение атрибуту может компилятор, для этого в текста ОА-программы должна присутствовать отдельная мнемоника: Mnemo.

3. Переменные

Переменные обозначаются с помощью знака «=», например, выражение Variable=10 является объявлением переменной Variable и присвоением ей начального значения равного 10.

Переменные могут быть двух видов:

- числовые/символьные (могут принимать один из пяти вышеперечисленных для констант типов), переменная считается числовой/символьной в том случае, если при ее объявлении в качестве начального значения выступает константа;
- указатели (ссылки), переменная считается указателем, если в качестве начального значения указывается ссылка, например, Variable2=Variable; для обозначения нулевой ссылки

в ОА языке применяются мнемоники «nil» или «нуль»: Variable2=nil.

4. Информационная пара

ИП описывается с помощью знака «=»: перед «=» стоит атрибут ИП, после – нагрузка: Mnemo="Variable". В качестве на грузки ИП могут выступать как константы, так и ссылки: Var=Variable.

Причем, имеется возможность задать ссылки на атрибут, нагрузку и на ИП в целом: ICLink[AtrLink(Mnemo)=PayloadLink(Variable)], где ICLink – мнемоника ссылки на ИП, AtrLink – мнемоника ссылки на атрибут ИП, PayloadLink – мнемоника ссылки на нагрузку. Ссылка на ИП оформляется с помощью знаков «[» «]».

5. Милликоманда

Милликоманда указывается в качестве атрибута ИП и состоит из двух частей: мнемоника ВФУ, которому милликоманда должна быть передана; мнемоника милликоманды. Эти две части отделаются одна от другой с помощью знака «.». По мнемонике ОА-компилятор производит синтез индекса расширенной милликоманды и помещает его в атрибут создаваемой ИП. Например, ALU.Set=0. Существуют три стандартные мнемоники милликоманд: Reset (Сброс) — сбросить ВФУ в начальное состояние, «Set» («Уст») – установить значение, «Pop» («Выд») – выдать значение. В том случае, когда в ОА=языке после мнемоники расширенной милликоманды на указана нагрузка, то в качестве нагрузки компилятор подставляет nil (нулевой указатель), если передается указатель; или 0, если передается константа).

К тому же имеется возможность осуществлять автопрограммирование ВФУ (когда ФУ выдает милликоманды самому себе). Итак, для автопрограммирования необходимо в мнемонике атрибута милликоманды перед знаком «.» указать только тип ВФУ: например, FuIntAlu.Add=2.

6. Информационная капсула

ИП группируются в капсулу с помощью знаков «{» и «}»: Caplsule{Mnemo="abc" Mnemo="xyz"} (перед знаком «{» стоит мнемоника указателя на капсулу, ИП разделяются между собой пробелом или знаком «,»). Информационная капсула может быть указана в качестве нагрузки ИП: Capsule{Mnemonics={Mnemo="abc" Mnemo="xyz"}} (в данном случае нагрузка ИП с атрибутом Mnemonics будет хранить указатель на капсулу {Mnemo="abc" Mnemo="xyz"}).

Во время формирования капсулы существует возможность вставки копии другой капсулы. Например, в результате следующего описания (Рисунок 6).

```
Caplsule2{Digit=123 Digit=456}
Capsule{Mnemo="abc" Caplsule2 Mnemo="xyz"}
```

Рисунок 6 - Процедура формирования капсулы, содержащей другую капсулу (миллипрограмма)

получается капсула Caplsule, которая будет содержать следующую информацию (Рисунок 7).

```
Capsule{Mnemo="abc" Digit=123 Digit=456 Mnemo="xyz"}
```

Рисунок 7 – Результат капсулы, содержащей другую капсулу (миллипрограмма)

Имеется и возможность сцепления капсул. Так, в результате такого описания (Рисунок 8) ИП капсулы Caplsule2 прицепляются к «хвосту» Caplsule (рис. 9).

```
Caplsule{Digit=123 Digit=456}
Caplsule2{Mnemo="abc" Mnemo="xyz" + Caplsule}
Caplsule3{Digit=789 Digit=012 + Caplsule2}
```

Рисунок 8 - Сцепление капсул (миллипрограмма)

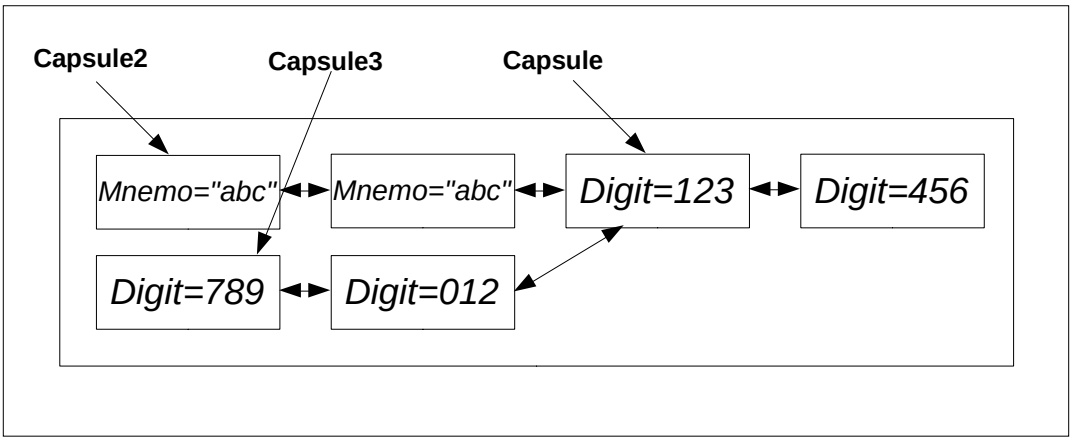


Рисунок 9 - Сцепление капсул

7. Знак уничтожения переменной

Для того, чтобы удалить какую-либо мнемонику из таблицы мнемоник ОА-компилятра, используется знак "!". Например. {ALU.Pop=temp OutKonsole.VarOut=temp(0)!} - переменная temp сначала объявляется, а затем уничтожается, т.к. она не нужна для дальнейших вычислений.

8. Комментарии

Текст комментариев в ОА-программировании оформляется с помощью знаков * Комментарии *\, также знаком комментария являются символы \\ - действие этого комментария распространяется до конца строки.

3 НАСТРОЙКА ПРОГРАММЫ

3.1. Настройка на состав технических средств

Программа не требует каких либо настроек на состав технических средств.

3.2. Настройка на состав программных средств

Программа должна быть установлена в каталог. Для установки данной программы достаточно скопировать перечисленные ниже файлы в указанную папку на компьютере пользователя. Каких-либо настроек после копирования программы не требуется.

Список необходимых файлов программы:

4. ПРОВЕРКА ПРОГРАММЫ

4.1. Описание способов проверки

Для проверки работоспособности программы можно воспользоваться несколькими способами:

- Проверить работоспособность программы можно открыв после запуска панель «ФУ»: во вкладке должено появиться дерево виртуальных устройств, содержащее перечень служебных устройств системы.
- Для проверки работоспособности можно использовать запуск следующей миллипрограммы: Console.Out="Hello". Если в выводной консоли появится надпись “Hello”, то программа исправна и готова к работе.

4.2. Методы прогона

4.2.1. Проверка работоспособности программы

Для проверки работоспособности программы, написанной на ОА-языке, рекомендуется в конце программы ставить милликоманду вывода сообщения об окончании программы, например, Console.LnOut="Program is finished". В случае зацикливания компилятор не сможет произвести запуск всех милликоманд, описанных программе и надпись "Program is finished" не будет выведена в выводную консоль.

4.2.2. Проверка на сообщение об ошибке

Проверить системы на сообщение об ошибке можно введя в консоль программы миллипрограмму (программу на специальном языке программирования для ОА-вычислительной системы) с синтаксической ошибкой. Например ввести миллипрограмму

«Console.OutErr» - в среде дожно открыться окно для служебных сообщений, где будет выведено сообщение «Wrong millicomand description!!!» («Неправильной описание милликоманды».

5 СООБЩЕНИЯ СИСТЕМНОМУ ПРОГРАММИСТУ

В ходе выполнения миллипрограммы могут выдаваться следующие сообщения об ошибках:

«Wrong millicomand description!!!» - ошибка в описании милликоманды

«Wrong attribute description!!!» - ошибка в описании атрибута милликоманды

«Wrong mnemo description!!!» - ошибка в описании мнемоники милликоманды

«Wrong millicomand load description!!!» - ошибка в описании нагрузки милликоманды

«Mnemo not founded!!!» - мнемоника не найдена

«Wrong Ic load description!!!» - ошибка в описании нагрузки информационной пары

«Wrong Ic description !!!» - ошибка в описании информационной пары

«Wrong capsule description!!!» - ошибка в описании информационной капсулы

«Wrong list description!!!» - ошибка в описании ОА-списка

«Bracket '(' not founded!!!» - знак «…» не найден

«Error!!!» - ошибка неопределенного типа

При загрузке программа может выдавать сообщение, показанное на Рисунке 10:

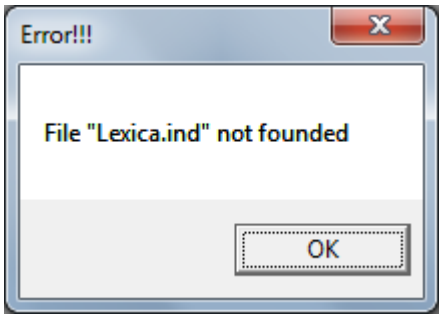


Рисунок 10 – Сообщение об ошибке

ПРИЧИНА. Отсутствие файла Lexica.ind в директории с программой.

ДЕЙСТВИЯ ПРОГРАММЫ. Остановка выполнения программы.

ДЕЙСТВИЯ ПРОГРАМИСТА. Записать файл Lexica.ind в директорию с программой.

Лист регистрации изменений								
Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Входящий № сопроводит. докум. и дата	Подп.	Дата
изменен-ных	заменен-ных	новых	аннули-рованных					

А3. Руководство оператора

УТВЕРЖДЕН

А.В.00001-01 34 01-ЛУ

ПРОГРАММА «СРЕДА СОЗДАНИЯ И МОДЕЛИРОВАНИЯ ОБЪЕКТНО-АТРИБУТНОЙ СУПЕРКОМПЬЮТЕРНОЙ СИСТЕМЫ С УПРАВЛЕНИЕМ ПОТОКОМ ДАННЫХ»

А.В.00001-01 34 01

Листов 15

2012

АННОТАЦИЯ

В данном программном документе приведено руководство оператора по применению и эксплуатации программы «Среда программирования и имитационного моделирования объектно-атрибутной суперкомпьютерной системы с управлением потоком данных» (далее Программа).

В данном программном документе, в разделе «Назначение программы» указаны сведения о назначении программы и информация, достаточная для понимания функций программы и ее эксплуатации.

В разделе «Условия выполнения программы» указаны условия, необходимые для выполнения программы (минимальный состав аппаратных и программных средств и т.п.).

В данном программном документе, в разделе «Выполнение программы» указана последовательность действий оператора, обеспечивающих загрузку, запуск, выполнение и завершение программы, приведено описание функций, формата и возможных вариантов команд, с помощью которых оператор осуществляет загрузку и управляет выполнением программы, а также ответы программы на эти команды.

В разделе «Сообщения оператору» приведены тексты сообщений, выдаваемых в ходе выполнения программы, описание их содержания и соответствующие действия оператора (действия оператора в случае сбоя, возможности повторного запуска программы и т.п.).

Оформление программного документа «Руководство оператора» произведено по требованиям ЕСПД ГОСТ 19.505-79[3].

[3] ГОСТ 19.505-79 ЕСПД. Руководство оператора. Требования к содержанию и оформлению

СОДЕРЖАНИЕ

1 НАЗНАЧЕНИЕ ПРОГРАММЫ

1.1 Функциональное назначение

Программа предназначена для создания и управления работой виртуальных функциональных устройств. Программа может использоваться для создания и исполнения программ и моделирования вычислительной системы ОА- архитектуры.

1.2 Эксплуатационное назначение

Программа должна применяться для создания вычислительных систем, управляемых потоком данных.

1.3 Интерфейс программы

1. Консоль ввода-вывода (программная консоль, консоль вывода, консоль служебных сообщений);
2. Инструментальная панель (дерево функциональных устройств, список атрибутов, список переменных, список указателей, список констант)
3. Главное меню.

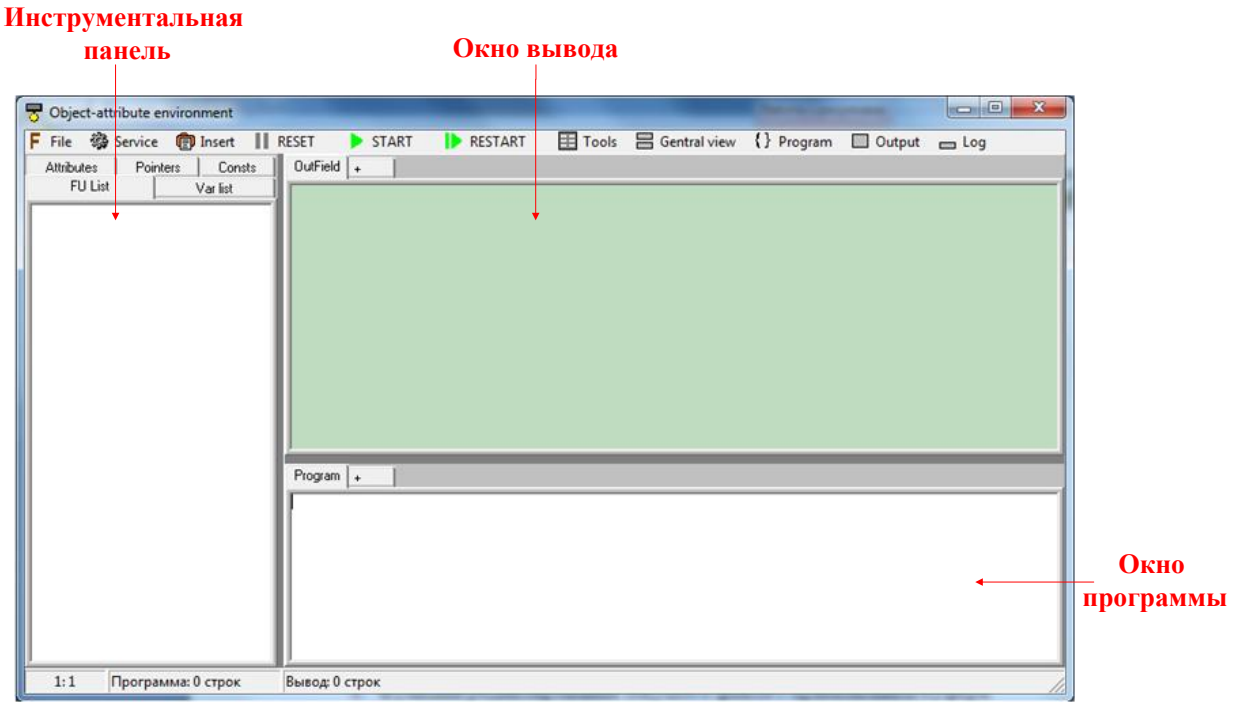


Рисунок 1- Интерфейс программы

Функции основных элементов программы:

1. Консоль ввода-вывода

- Функции работы с файлами (создание, открытие, редактирование, сохранение);
- Функции работы с текстом;

- Вывод результата работы программы;
- Запуск и перезапуск программы;

2. Инструментальная панель
   - Просмотр созданных в среде функциональных устройств и автоматический ввод милликоманд в программную панель
   - Просмотр атрибутов, констант, переменных, указателей;

3. Главное меню.
   - Создание индексного файла;
   - Выполнение индексного файла;
   - Очистка окна вывода;
   - Смена языка интерфейса среды;
   - Автоматическое построение списка функциональных устройств;
   - Автозагрузка;
   - Вставка наиболее применяемых конструкций языка;
   - Управление программой;
   - Настройка вида среды.

2 УСЛОВИЯ ВЫПОЛНЕНИЯ

2.1 Минимальный состав технических средств

В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ), включающий в себя:

- процессор с тактовой частотой, ГГц - 1, не менее;
- оперативную память объемом, Мб-512, не менее;

2.2 Минимальный состав программных средств

Системные программные средства, используемые средой создания и выполнения ОА-образов, должны быть представлены операционной системой Windows XP (Windows Vista, Windows).

Также для реализации всех возможностей программы требуется предустановленный модуль OpenGL.

3 ВЫПОЛНЕНИЕ ПРОГРАММЫ

3.1 Выполнение функций консоли ввода-вывода

Работа с файлами

Создание

Выполнение указанной функции возможно с помощью открытия вкладки «плюс» («+») панели ввода-вывода (Рисунок 2).

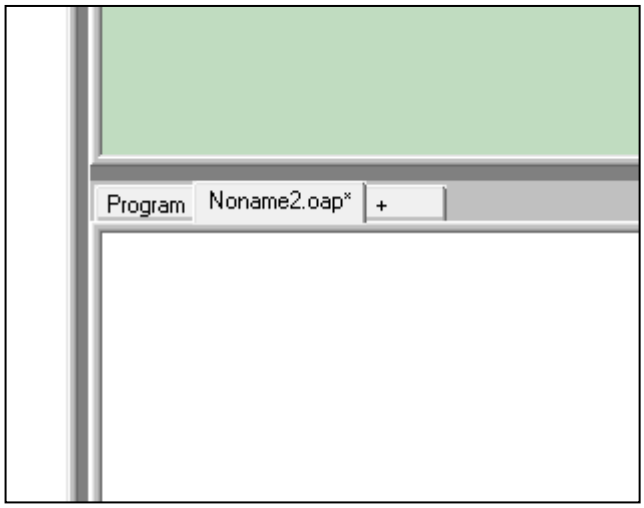


Рисунок 2- Создание файла в новой вкладке

Примечание - При успешном завершении загрузки и запуска программа автоматически создаст новый (безымянный) файл.

Открытие

Выполнение указанной функции возможно любым из перечисленных ниже способов:

последовательным выбором пунктов меню Файл-Открыть;

последовательным нажатием клавиш Ctrl и O (сочетанием клавиш Ctrl+O).

В результате на рабочем столе будет отображено окно Открыть.

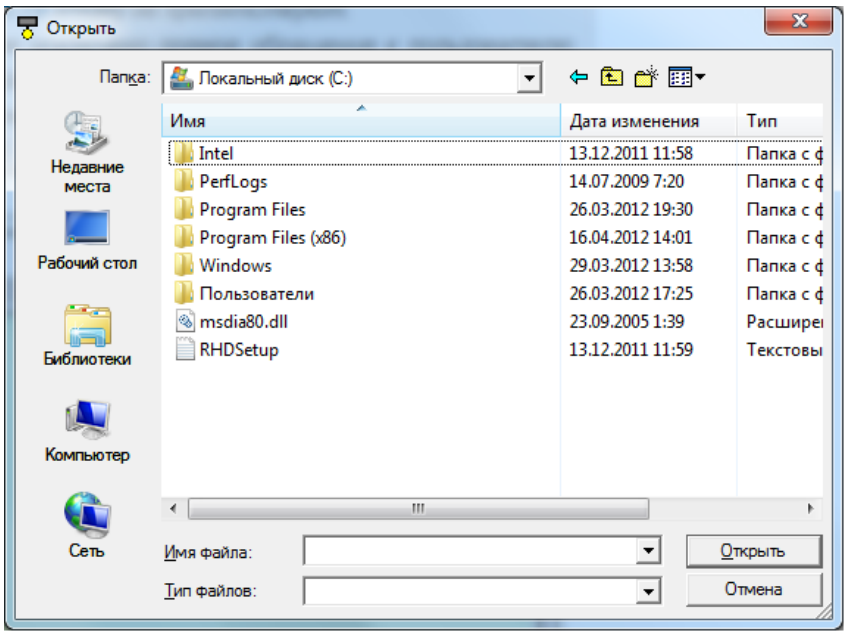


Рисунок 3 – Открытие файла

Примечание - Программа обеспечивает возможность загрузки файлов только с расширением *.oap и *.ind.

Завершается выполнение функции нажатием кнопки Открыть.

В случае успешного (выполнения программой функции) открытия файла на рабочем столе будет отображено окно с содержимым открытого (текущего) файла.

Редактирование

Осуществляется стандартными средствами и подразумевает следующие функции: Выделение текста Ctrl+A; Копирование Ctrl+C; Вырезать Ctrl+X; Вставка Ctrl+V.

Сохранение

Выполнение указанной функции возможно любым из перечисленных ниже способов:

- последовательным выбором пунктов меню Файл-Сохранить;
- последовательным нажатием клавиш Ctrl и S (сочетанием клавиш Ctrl+S).

Работа с текстом

Поиск

Вызов вкладки Find (Поиск) дополнительного окна управления осуществляется с использованием сочетания клавиш Ctrl+F.

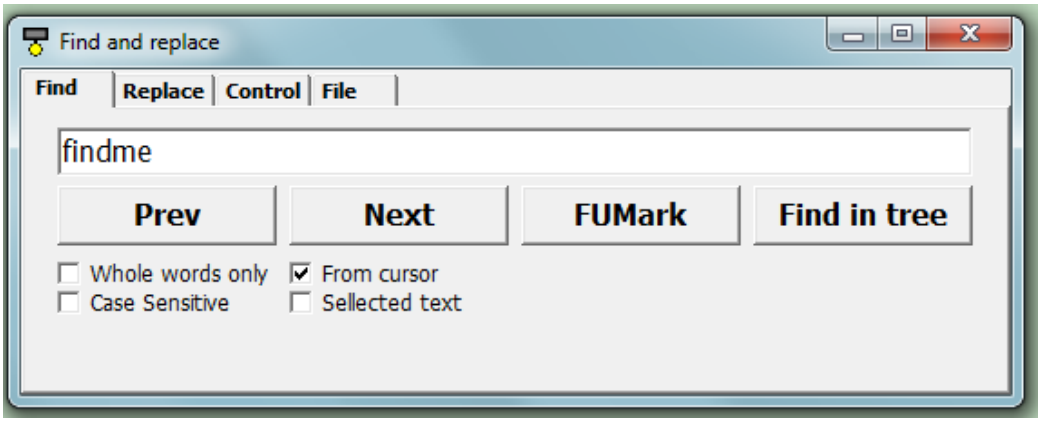


Рисунок 4 – Поиск по тексту

Основные элементы вкладки Find (Поиск)

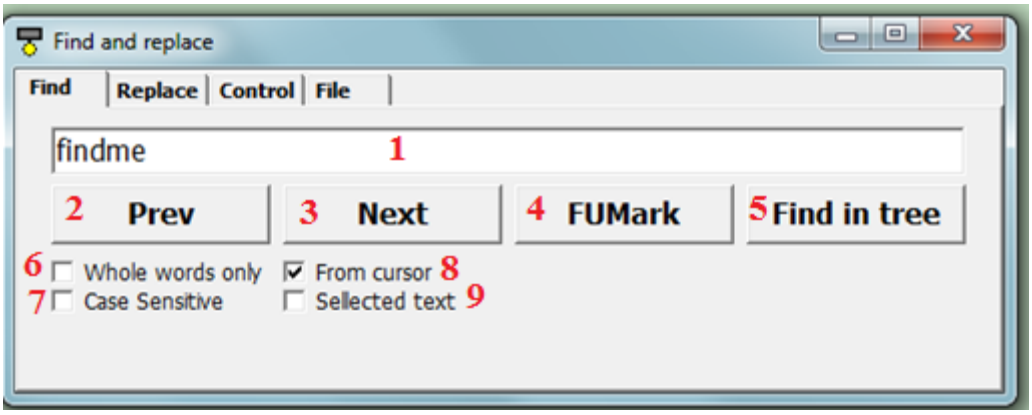


Рисунок 5 – Основные элементы вкладки Find (Поиск)

1. Текстовое поле для ввода поисковой строки
2. Кнопка «Previous» - Поиск предыдущего вхождения поисковой строки в тексте программы
3. Кнопка «Next» - Поиск следующего вхождения поисковой строки в тексте программы
4. Кнопка «FUMark» - Вставка специальной «строки-маски» в текстовое поле поиска [1]
5. Кнопка «Find in tree» - Поиск по инструментальному ОА-дереву
6. Флаг «Whole words only». Если выбран, то в результаты поиска будут включены только те вхождения, которые удовлетворяют условию «слово целиком»
7. Флаг «Case Sensitive» - Если выбран, то осуществляется регистрозависимый поиск, иначе – регистронезависимый
8. Флаг «From Cursor» - Если выбран, то поиск осуществляется с позиции курсора, иначе – с начала текста программы
9. Флаг «Sellected text» - Если выбран, то поиск осуществляется только по выделенному фрагменту текста программы, иначе – по всему тексту программы

Замена

Вызов вкладки Replace (Замена) дополнительного окна управления осуществляется с использованием сочетания клавиш Ctrl+R.

Основные элементы вкладки Replace (Замена)

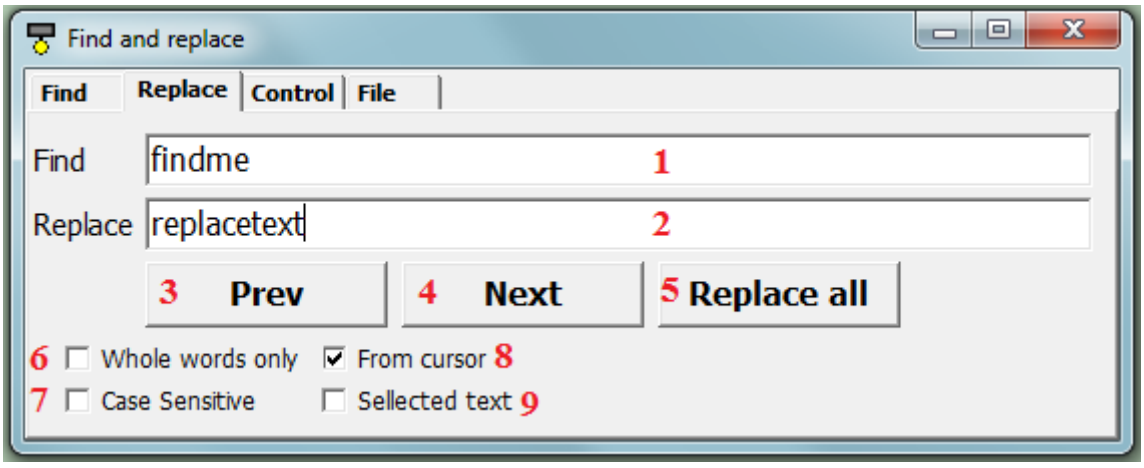


Рисунок 6 – Основные элементы вкладки Replace (Замена)

1. Текстовое поле для ввода поисковой строки
2. Текстовое поле для ввода заменяемой строки
3. Кнопка «Previous» - Поиск и замена предыдущего вхождения поисковой строки в тексте программы
4. Кнопка «Next» - Поиск и замена следующего вхождения поисковой строки в тексте программы
5. Кнопка «Replace All» - Поиск и замена всех вхождений поисковой строки в тексте программы
6. Флаг «Whole words only». Если выбран, то в поиск и замена будут осуществляться только с теми вхождениями, которые удовлетворяют условию «слово целиком»
7. Флаг «Case Sensitive» - Если выбран, то осуществляются регистрозависимый поиск и замена, иначе – регистронезависимый поиск и замена
8. Флаг «From Cursor» - Если выбран, то поиск и замена осуществляются с позиции курсора, иначе – с начала текста программы
9. Флаг «Sellected text» - Если выбран, то поиск и замена осуществляются только внутри выделенного фрагмента текста программы, иначе – по всему тексту программы

Вывод результата работы программы

Вывод результата работы запущенной программы в программной панели осуществляется в текущей вкладке панели вывода.

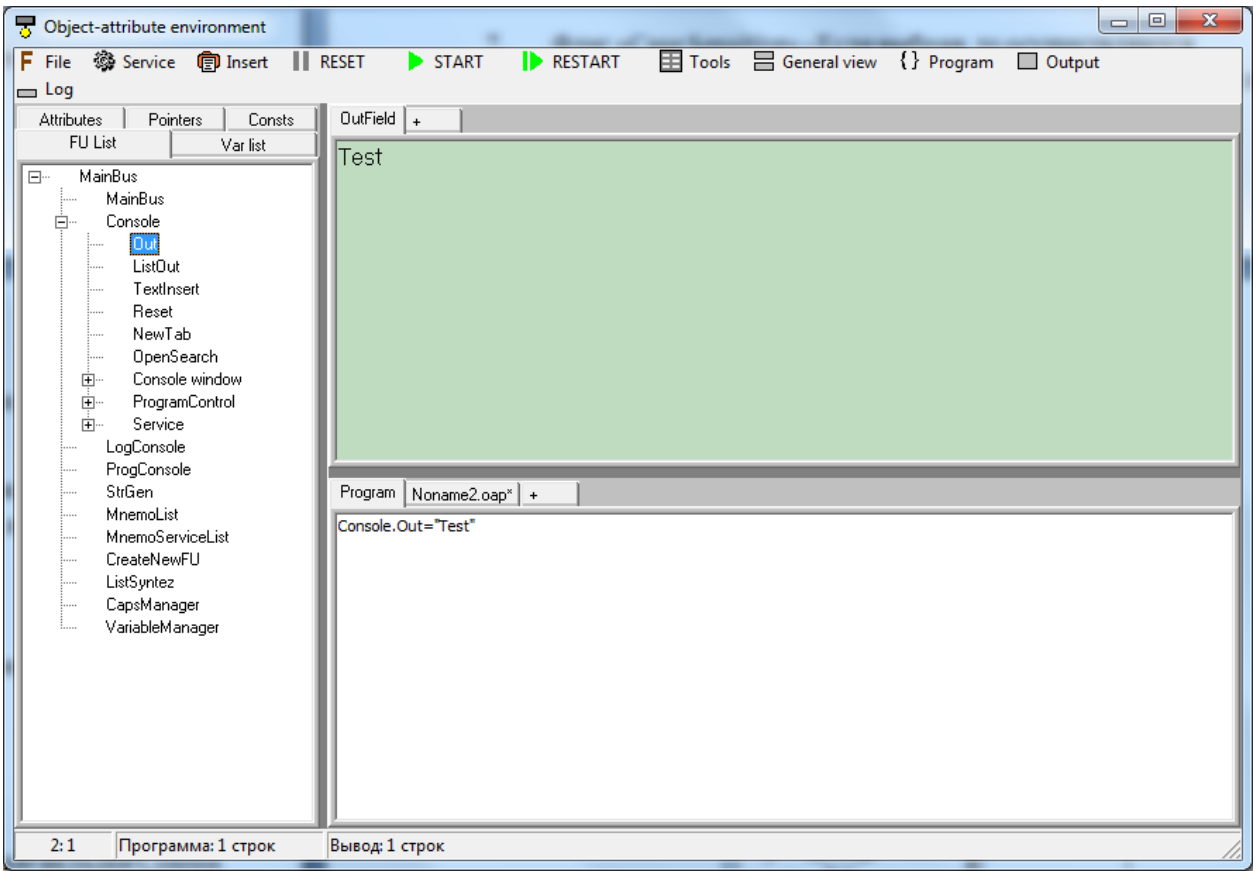


Рисунок 7 – Вывод результата работы программы

Запуск и перезапуск программы

Выполнение указанных функций осуществляется с помощью вкладки Control (Управление) дополнительного окна управления.

Основные элементы вкладки Control (Управление)

1. Кнопка «Restart program» - Перезапуск программы
2. Кнопка «Start program» - Запуск программы
3. Кнопка «Reset program» - Сброс программы
4. Кнопка «Clear text page» - Очистка текста программы текущей вкладки консоли ввода/вывода
5. Кнопка «Clear all text pages» - Очистка всех текстов программ на всех вкладках консоли ввода/вывода

3.2 Выполнение функций инструментальной панели

Просмотр созданных в среде функциональных устройств и автоматический ввод милликоманд в программную панель

Просмотр возможен в инструментальной панели во вкладке ФУ. Она представляет собой дерево компонент, где расположен список функциональных устройств, созданных в среде программирования.

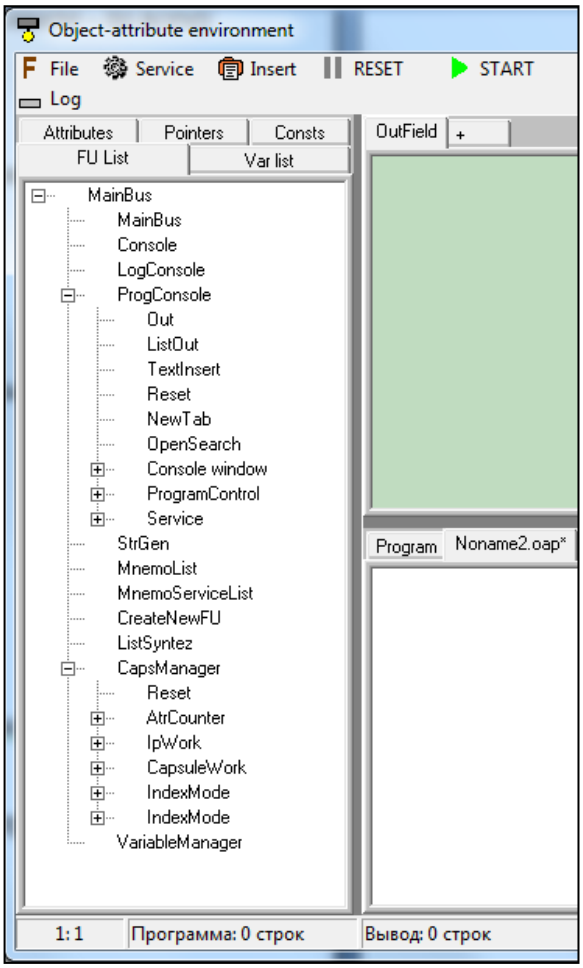


Рисунок 8 – Просмотр созданных в среде функциональный устройств

Для автоматического ввода милликоманды в программное поле, необходимо выполнить следующие действия:

1. Обозначить место курсора для вставки милликоманды.
2. Нажатием на название развернуть дерево списка милликоманд функционального устройства.
3. Выбрать милликоманду двойным нажатием.

Просмотр атрибутов, констант, переменных, указателей

Просмотр возможен в инструментальной панели среды в соответствующих вкладках («Атрибуты», «Константы», «Переменные», «Указатели»). Каждая вкладка содержит список мнемоник.

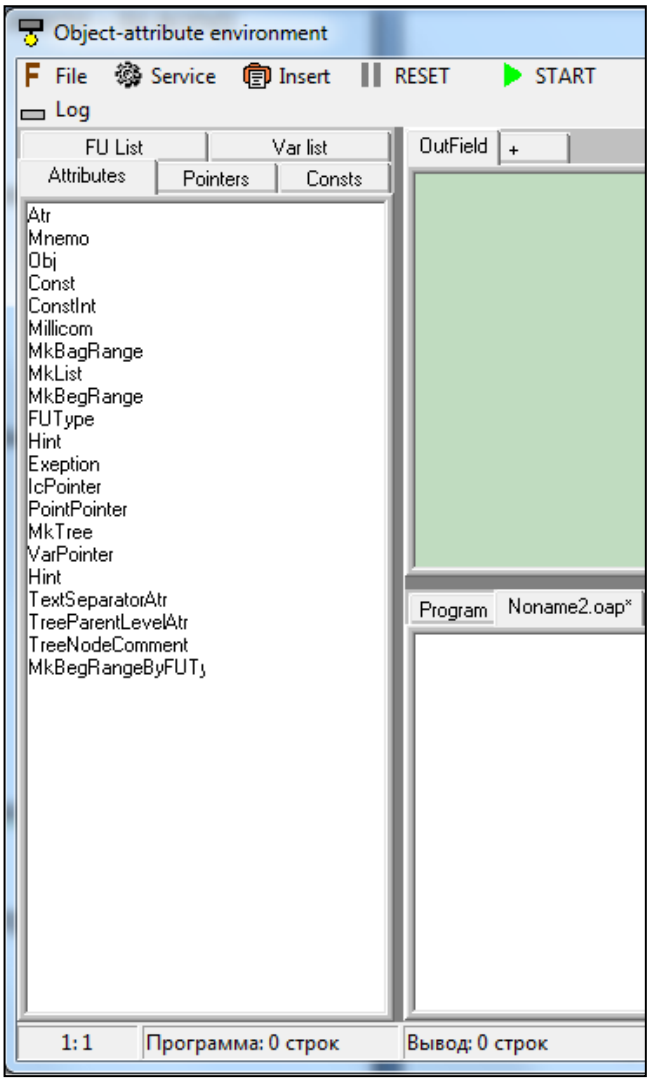


Рисунок 9 – Просмотр атрибутов, констант, переменных, указателей

Для выбора мнемоники необходимо выполнить следующие действия:

1. Обозначить место курсора для вставки мнемоники.
2. Выбрать мнемонику двойным нажатием.

3.3 Выполнение функций главного меню

Создание индексного файла

Для создания индексного файла (файл, запускаемый без компиляции) необходимо выполнить следующие действия:

1. Запустить программу.
2. В меню «Файл» выбрать соответствующий пункт.
3. Создать имя для индексного файла.

Выполнение индексного файла

Для запуска индексного файла необходимо выполнить следующие действия:

1. В меню «Файл» выбрать соответствующий пункт.
2. В диалоговом окне выбрать требуемый для исполнения файл.

Очистка окна вывода

Для очистки окна вывода необходимо в меню «Файл» выбрать пункт «Очистить окно вывода» («Clear Out Window»).

Смена языка интерфейса среды

Для смены языка интерфейса среды необходимо в меню «Сервис» («Service») выбрать пункт «Англ. интерфейс» («English interface»).

Автоматическое построение списка функциональных устройств

Для автоматического формирования списка функциональных устройств в инструментальной панели необходимо в меню «Сервис» («Service») выбрать пункт «Автоматический список ФУ». При выборе данной опции, все запущенные в среде функциональные устройства будут выведены в инструментальную панель в виде дерева компонент.

Автозагрузка

Для автоматической загрузки исполняемой программы в окно программы среды необходимо в меню «Сервис» («Service») выбрать пункт «Автозагрузка» («Autoload»).

Вставка наиболее применяемых конструкций языка

В среде имеется возможность вставки в текст программы наиболее применяемых конструкций языка. Для этого в меню «Вставка» («Insert») необходимо выбрать требуемую конструкцию.

Кнопки управления программой

Функции управления программой осуществляют следующие кнопки (дублируются с программной консоли):

«Сброс» («Reset») - Сброс программы;

«Старт» («Start») - Запуск программы;

«Рестарт» («Restart») - Перезапуск программы.

Настройка вида среды

Модификация вида среды осуществляется с помощью следующих кнопок:

«Инструментальная панель» («Tools») - Скрыть/показать инструментальную панель;

«Общий вид» («General view») - Показать все окна среду по умолчанию;

«Программа» («Program») - Скрыть/показать окно программы;

«Выводная консоль» («Output») - Скрыть/показать окно выводной консоли;

«Служебные сообщения» («Log») - Скрыть/показать окно служебных сообщений. Окно

служебных сообщений используется для вывода сообщений о синтаксических ошибка в программе. При обнаружении ошибок, данное окно появляется автоматически.

4 ЗАВЕРШЕНИЕ РАБОТЫ ПРОГРАММЫ

Выполнение указанной функции возможно любым из перечисленных ниже способов:

1. Последовательным выбором пунктов меню Файл-Выход (File-Exit);
2. Нажатием кнопки  .

5 СООБЩЕНИЯ ОПЕРАТОРУ

Все служебные сообщения оператору выводятся в служебную консоль.

Лист регистрации изменений

Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Входящий № сопроводит. докум. и дата	Подп.	Дата
изменен-ных	заменен-ных	новых	аннули-рованных					

А4. Описание программы

УТВЕРЖДЕН

А.В.00001-01 13 01-ЛУ

ПРОГРАММА «СРЕДА СОЗДАНИЯ И МОДЕЛИРОВАНИЯ ОБЪЕКТНО-АТРИБУТНОЙ СУПЕРКОМПЬЮТЕРНОЙ СИСТЕМЫ С УПРАВЛЕНИЕМ ПОТОКОМ ДАННЫХ»

А.В.00001-01 13 01

Листов 103

2012

Инв. № подл.	Подпись и дата	Взам. инв. №	Инв. № дубл.	Подпись и дата

АННОТАЦИЯ

В данном программном документе приведено описание программы создания и выполнения ОА – образов, предназначенной для создания и управления работой виртуальных функциональных устройств. Программа может использоваться для создания и исполнения программ и моделирования вычислительной системы ОА- архитектуры.

Исходным языком программы «millicom.exe» является Delphi.

Программа реализует следующие функции:

6) Создание виртуальных вычислительных систем ОА-архитектуры
7) Реализация алгоритмов для систем с управлением потоком данных
8) Управление функциональными устройствами
9) Создание виртуальных вычислительных устройств
10) Формирование индексного файла (предварительная компиляция ОА-программы для последующего запуска)

Оформление программного документа «Описание программы» произведено по требованиям ЕСПД ГОСТ 19.402-78[4].

[4] ГОСТ 19.402-78 ЕСПД. Описание программы

СОДЕРЖАНИЕ

1 ОБЩИЕ СВЕДЕНИЯ

1.1 Обозначение и наименование программы

Программа «Среда программирования и имитационного моделирования объектно-атрибутной суперкомпьютерной системы с управлением потоком данных» имеет следующие атрибуты:

- Наименование исполняемого файла - millicom.exe
- Размер исполняемого файла - 44 Мб
- «Иконка» исполняемого файла - 
- Версия файла - 1.0
- Версия продукта - 1.0
- Внутреннее имя - millicom
- Исходное имя файла - millicom.exe
- Название продукта - millicom
- Производитель - МИЭМ (ТУ)
- Язык интерфейса - Английский/Русский

1.2 Программное обеспечение, необходимое для функционирования программы

Системные программные средства, используемые средой создания и выполнения ОА-образов, должны быть представлены операционной системой Windows XP (Windows Vista, Windows).

Также для реализации всех возможностей программы требуется предустановленный модуль OpenGL.

1.3 Языки программирования, на которых написана программа

Программа «Среда программирования и имитационного моделирования объектно-атрибутной суперкомпьютерной системы с управлением потоком данных» была реализована на языке высокого уровня Delphi. Компилятор ОА-языка реализован на ОА-языке.

2 ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ

2.1 Классы решаемых задач

Программа предназначена для решения нескольких классов задач:

- создание и запуск на выполнение программ для суперкомпьютерной системы объектно-атрибутной архитектуры;

- имитационное моделирование вычислительного процесса на распределенной вычислительной системе объектно-атрибутной архитектуры.

2.2 Назначение программы

Программа предназначена для создания и управления работой виртуальных функциональных устройств. Программа может использоваться для создания и исполнения программ и моделирования вычислительной системы ОА- архитектуры.

Программа реализует следующие функции:

1) Создание виртуальных вычислительных систем ОА-архитектуры
2) Реализация алгоритмов для систем с управлением потоком данных
3) Управление функциональными устройствами
4) Создание виртуальных вычислительных устройств
5) Формирование индексного файла (предварительная компиляция ОА-программы для последующего запуска)

2.3 Сведения о функциональных ограничениях на применение

Системные программные средства, используемые средой создания и выполнения ОА-образов, должны быть представлены операционной системой Windows XP (Windows Vista, Windows).

Также для реализации всех возможностей программы требуется предустановленный модуль OpenGL.

3 ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ

3.1 Описание виртуального функционального устройства

Виртуальное функциональное устройство (ВФУ) составляет основным функциональным блоком программы. ВФУ – это подпрограмма с универсальным интерфейсом (набором входных параметров), выполняющая определенную обработку данных и выдачу результатов этой обработки. ВФУ состоит из контекста (набора

виртуальных регистров: как правило, контекст реализуется с помощью структуры (записи) в языке высокого уровня) и процедуры реализации логики работы ВФУ, у которой имеется универсальный интерфейс. Например, на языке высокого уровня Delphi интерфейс ВФУ будет выглядеть следующим образом:

```
Procedure IntALU(context: Pointer; millicomand: int64; Obj1: TPointIndex);
```

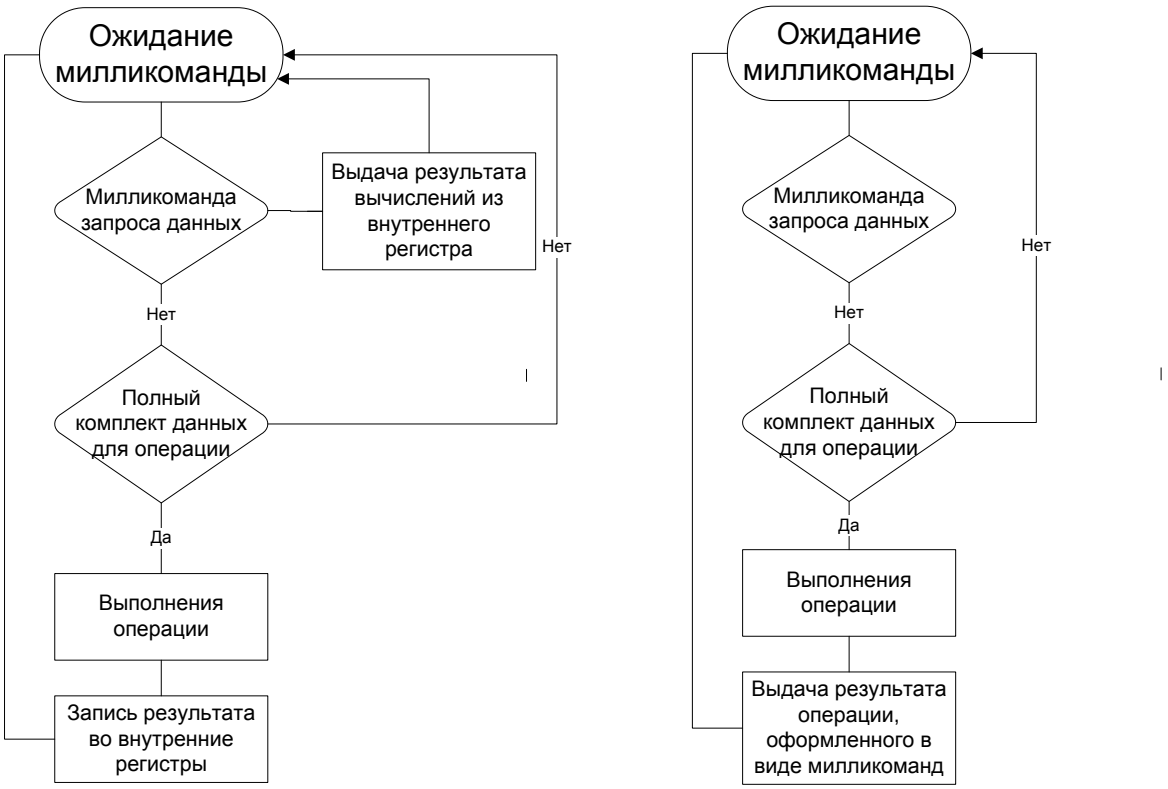
где Context – ссылка на контекст виртуального устройства;

millicomand – индекс милликоманды (каждая милликоманда имеет свой уникальный идентификатор);

Load – ссылка на нагрузку милликоманды (ссылка на информационную конструкцию, которая выступает в качестве нагрузки к милликоманде).

3.2 Алгоритм работы виртуального функционального устройства

Программа, разбита на множество функциональных блоков, оформленных в виде виртуальных функциональных устройств (ВФУ). ВФУ создают вычислительную среду, на которой запускается вычислительный процесс. Алгоритм задается с помощью описания обмена данными между ВФУ (данные оформляются в виде милликоманд (совокупность атрибута (универсального идентификатора данных) и нагрузки (указателя на переменную или информационную конструкцию). Атрибут однозначно идентифицирует нагрузку (по атрибуту ВФУ распознают данные, находящиеся в нагрузке) и исходя из атрибутов накапливают в своих внутренних регистрах комплект данных для выполнения вычислений. Как только полный комплект данных оказывается во внутренних регистрах ВФУ, начинается процесс обработки данных. Далее существует два варианта работы ВФУ: 1) ВФУ записывает результат во внутренние виртуальные регистры и ждет запроса данных (запрос также оформляется в виде милликоманды): в этом случае в качестве нагрузки передается адрес ячейки памяти, куда следует поместить результат вычислкний; 2) самостоятельная выдача результата (в этом случае в контексте ВФУ находится ссылка на ВФУ, которому следует передавать результат и атрибут, которым снабжаются передаваемые данные).



а) Модель работы ВФУ с запросом результата вычислений

б) Модель работы ВФУ с самостоятельной выдачей результата вычислений

Рисунок 1 – Модели работы ВФУ

3.3 Формат милликоманды

ВФУ управляются с помощью милликоманд. Атрибут милликоманды бывает локальным и расширенным. Локальный атрибут идентифицирует данные, находящиеся в нагрузке милликоманды. Расширенный атрибут, кроме атрибута данных указывает и ВФУ, которому адресуется милликоманда. Расширенный атрибут формируется по следующему правилу (1):

ExtendedMillicom=NFU*MilliRange+MillicomIndex, (1)

где NFU — номер созданного ВФУ;

MilliRange — диапазон адресов милликоманд (данная величина входит в контекст Шины);

MillicomIndex — индекс милликоманды для ВФУ, которому адресуется милликоманда.

3.4 Коммуникация между ВФУ

Передачу милликоманд ВФУ могут производить как напрямую (когда одно ВФУ вызывает программу реализации логики работы другого ВФУ и в качестве параметров

передает локальную милликоманду и нагрузку), так и через ВФУ-коммутатор. Коммутатор по расширенной милликоманде определяет номер ВФУ-приемника по формуле:

NFU=ExtendedMillicom div MilliRange , (2)

где div – операция целочисленного деления

В контекст коммутатора входит массив указателей на программы реализации логики работы всех ВФУ, между которыми он осуществляет передачу милликоманд. Индекс локальной милликоманды, передаваемой на ВФУ-приемник, определяется по формуле (3):

NFU=ExtendedMillicom mod MilliRange , (3)

где mod – операция нахождения остатка от целочисленного деления.

Описание основных типов ВФУ дается в **Приложении А**[5]**.**

3.5. Структура программы

Программа состоит из:

- ОА-платформы (часть ОА-системы, реализующая логику работы виртуальных ВФУ) состоит из описания контекстов ВФУ и подпрограмм реализации логики работы для каждого типа ВФУ;

- компилятора ОА-языка (реализованного на разработанной ОА-платформе);

- инструментальных средств разработки ОА-образа: рабочая панель проектирования ОА-образа (окна с перечнем участвующих в вычислительном процессе ВФУ, указателей, переменных, констант и атрибутов); панель инструментов (окно вывода результатов выполнения программы, окно служебных сообщений, окно редактора ОА-образа и тестовых примеров).

Все функциональные части программы оформлены в виде специализированных ВФУ.

4 ИСПОЛЬЗУЕМЫЕ ТЕХНИЧЕСКИЕ СРЕДСТВА

В состав используемых технических средств входит: IBM PC совместимый с процессором 80386 и выше, ОЗУ не менее 32 Мбайт, 16 МБ видеопамяти, наличие свободного места на жестком диске 100 Мбайт.

В состав технических средств входит IBM-совместимый персональный компьютер (ПЭВМ), включающий в себя процессор с тактовой частотой не менее 1 ГГц - 1; оперативную память объемом не менее Мб-512.

[5] Описание основных типов ВФУ См. Приложение 1

5 ВЫЗОВ И ЗАГРУЗКА

Загрузка и запуск программы осуществляется способами, детальные сведения о которых изложены в Руководстве пользователя.

6 ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

6.1 Сведения о входных данных

6.1.1. ОА-программа в текстовом виде

Входные данные оформляются в виде языковых конструкций на объектно-атрибутном языке.

Компилятор языка среды создания и запуска ОА-образа выполнен на базе ОА-архитектуры и реализует следующие языковые конструкции:

1. Константы

В ОА-языке могут использоваться константы следующих типов:

- символ (обозначается с помощью знака «’», например, ’a’);

- строка (обозначается с помощью знака «"», например, "abc");

- логическая константа («истина», «true», «ложь», «false»);

- целое число;

- дробное число (чтобы компилятор воспринял число как дробное, в нем обязательно должно присутствовать обозначение дробной части: например, 234.0).

2. Атрибуты

Атрибут можно задать двумя способами. Во-первых, можно в качестве атрибута задать конкретное число (задается с помощью знака "*" после мнемоники атрибута): Mnemo*2. Во-вторых, автоматически присвоить значение атрибуту может компилятор, для этого в текста ОА-программы должна присутствовать отдельная мнемоника: Mnemo.

3. Переменные

Переменные объявляются с помощью знака «=», например, выражение Variable=10 является объявлением переменной Variable и присвоением ей начального значения равного 10.

Переменные могут быть двух видов:

- числовые/символьные (могут принимать один из пяти вышеперечисленных для констант типов), переменная считается числовой/символьной в том случае, если при ее объявлении в качестве начального значения выступает константа;

- указатели (ссылки), переменная считается указателем, если в качестве начального значения указывается ссылка, например, Variable2=Variable; для обозначения нулевой

ссылки в ОА языке применяются мнемоники «nil» или «нуль»: Variable2=nil.

4. Информационная пара

ИП описывается с помощью знака «=»: перед «=» стоит атрибут ИП, после – нагрузка: Mnemo="Variable". В качестве нагрузки ИП могут выступать как константы, так и ссылки: Var=Variable.

5. Милликоманда

Милликоманда указывается в качестве атрибута ИП и состоит из двух частей: мнемоника ВФУ, которому милликоманда должна быть передана; мнемоника милликоманды. Эти две части отделаются одна от другой с помощью знака «.».

6. Информационная капсула

ИП группируются в капсулу с помощью знаков «{» и «}»: Caplsule{Мнемо="abc" Mnemo="xyz"} (перед знаком «{» стоит мнемоника указателя на капсулу, ИП разделяются между собой пробелом или знаком «,»).

8. Комментарии

Текст комментариев в ОА-программировании оформляется с помощью знаков * Комментарии *\, также знаком комментария являются символы \\ - действие этого комментария распространяется до конца строки.

6.1.2. ОА-программа в виде индексного массива

Также ОА-программа может быть представлена в виде индексного массива (ОА-программа, представленная в индексном виде (наподобие байт-кода JAVA-машины)). Для запуска индексного массива на ОА-платформе не требуется, как для текстового представления, компиляции программы, что существенно ускоряет запуск миллипрограммы. Для запуска индексного массива следует воспользоваться пунктом «Файл – выполнить индексный файл».

6.2 Сведения о выходных данных

Выходными данными является текстовый файл, формируемый в процессе выполнения программы. Кроме того, для вывода могут быть использованы стандартные VCL-компоненты, входящие в состав среды Borland Delphi (оформлены в виде функциональных устройств). Также для формирования выходных данных может быть использовано ВФУ Файловый шлюз (GatewayFile). ОА-программа может быть записана в файл. Также в файл может быть записан индексный массив (ОА-программа, представленная в индексном виде (наподобие байт-кода JAVA-машины)). Индексный массив. Для формирования индексного массива следует воспользоваться пунктом меню «Файл – выполнить индексный файл».

Лист регистрации изменений

Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Входящий № сопроводит. докум. и дата	Подп.	Дата
измененных	замененных	новых	аннулированных					

Приложение 1. Описание основных типов ВФУ

1 Диспетчер ВФУ (Bus)

1.1 Функциональное назначение

1. Осуществляет диспетчирование ВФУ: создание и уничтожение ВФУ;
2. Осуществляет коммуникацию милликоманд между ВФУ;
3. Частично осуществляет маршрутизацию милликоманд (если милликоманда адресована другому ядру, то она передается на маршрутизатор для дельнейшей отправки к ВФУ-адресату по наиболее оптимальному маршруту)

1.2 Контекст

Наименование поля контекста	Тип	Описание
ProgramPointers	TFUProgramArray	Указатели на процедуры реализации ФУ
KillPointers	TFUKillArray	Указатели на процедуры уничтожения ФУ
ContextPointers	Array of Pointer	Массив указателей на контексты ВФУ
UnitCount	integer	счетчик ФУ
urrentFU, LastFU	Variant	Номер текущего ФУ и номер последнего созданного ФУ
HieAdr, LowAdr,SelfAdr	int64	Верхняя и нижняя грани адресов Шины; верхняя граница адреса передачи милликоманды самому себе
MilliDiap, MilliDiapHalfIndex	int64	Миллидиапазон и миллидиапазон для полуиндексного режима работы Шины
RouterUK	Pointer	сылка на маршрутизатор (милликоманда передаётся шлюзу, если его её адрес не попадает в диапазон
RouterMk, RouterMkRerase	variant	Милликоманда, передаваемая на Роутер и милликоманда передачи на роутер и уничтожения передаваемой милликоманды
IndexMode	boolean	Флаг индексного режима
HalfIndexMode	boolean	Флаг полуиндексного режима (Когда команда создания нового ФУ, записывается в индексный массив)

IndexVector	TIndexVector	Указатель на массив индексов
IndexCounter	PVariant	Указатель на контекст ФУ счетчика Индексов (для индексного режима)
LastBusProgIP	PAtrData	Ссылка на последнюю ИП программы, выдаваемой на шину
PredMillicomand	int64	Предыдущая милликоманда (нужно для корректной работы полуиндексного режима
SluseFileBus	TSluseFileContext	Файловый шлюз для запуска индексного файла
NFUToReset	Variant	Количество ФУ для неполного сброса

1.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
1	Create	Создание нового ФУ
3	LastMkRangePop	Выдать миллидиапазон последнего созданного ФУ
12	ProgExec	Выполнить миллипрограмму
112	MkExec	Выполнить одну милликоманду
212	ServiceProgExec	Выполнить сервисную миллипрограмму (выполняется при включенном индексном режиме)
300	FileExec	Загрузить и выполнить миллипрограмму из файла
40	IndexVectorCreate	Создать индексный вектор (на входе - размерность индексного массива)
42	IndexVectorSet	Установить указатель на индексный вектор
44	IndexCounterPointerSet	Установить указатель на переменную индексного счетчика
46	IndexVectorPop	Выдать указатель на индексный вектор
48	IndexCounterPointerPop	Выдать указатель на индексный счетчик
50	ModeSet	Установить индексный режим
0	Reset	Сброс ФУ
100	PartialResetSet	Установить количество ФУ для частичного

		сброса
105	PartialReset	Частичный сброс ФУ
11	ContextPop	Выдать контекст данного ФУ
20	ContextPopMk	Выдать милликоманду с контекстом данного ФУ
68	Halt	Прекращение работы всей программы
80	SelfMkRangeSet	Установить миллидиапазон для адресании ФУ самому себе
200	RouterSet	Установить контекст роутера
14	TopRangeSet	Установить верхний диапазон адресов милликоманд
15	BottomRangeSet	Установить верхний диапазон адресов милликоманд

1.4 Примеры программирования

\\ Установить ссылку на контекст Шины для ВФУ Автомат

Bus.ContextPopMk=Automat.ContextSet

2 Служебная консоль (LogCon)

2.1 Функциональное назначение

Ввод текстовой информации на консоль, запись текста в файл, чтение текста из файла

2.2 Контекст

Наименование поля контекста	Тип	Описание
F	TextFile	Файловая переменная для вывода в файл
SetCharAtribyte, SetStringAtribyte, IsOpenFile	boolean	Флаги установки атрибутов симнола и строки и открытия файла
OutMemo	TMemo	Ссылка на компонент Memo
OutWindow	TForm	Окно для вывода
StrBuf	String	Строковой буфер (в нём можно составить строку, которую можно будет вывести)
Mk	int64	Милликоманда, прикрепляемая к генерируемым

		строкам
PrefixStrGen, PostFixStrGen	Pointer	Префиксная и постфиксная программы при генерации строк

2.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
10	CapsuleOut	Вывести капсулу
1	FileSet	Открыть файл
9	IcOut	Вывести запись (атрибут и объект)
11	ListOut	Вывести список
12	ListSeparatorOut	Вывести список с разделителями линий
40	MatrOut	Вывести матрицу
8	MemoClear	Очистить поле вывода
210	MemoPop	Выдать ссылку на компонент Memo
220	MemoPopMk	Выдать милликоманду со ссылкой на компонент Memo
7	MemoSet	Задать ссылку на Memo
0	Reset	Сброс ФУ
6	SpaceOut	Вывести символ перевода строки
230	StirngsPop	Выдать ссылку на компонент TStrings
240	StringsPopMk	Выдать милликоманду со ссылкой на компонент TStrings
100	TextInsert	Вставить текстовую строку
17	VarBuf	Поместить переменную в строковой буфер
4	VarOut	Вывести символ или строку (переменную)

2.4 Примеры программирования

ALU.PopMk=Con.VarOut \\ Вывод величины, хранимой в аккумулятора АЛУ на консоль

List.PopMk=Con.ListOut \\ Вывод ОА-списка на консоль

3 Автомат (Automat)

3.1 Функциональное назначение

Выдача последовательности милликоманд на Шину, выполнение безусловных и

условных переходов в программе, вызов подпрограмм, рекурсивный вызов подпрограмм.

3.2 Контекст

Наименование поля контекста	Тип	Описание
PC, StartProg: PAtrData	PAtrData	Программный счетчик и адрес начала программы
Mk	Int64	Индекс милликоманды, выдаваемой для приёмника милликоманд (по умолч 112)
InputObject	TPointIndex	Указатель на входной объект
DelInputObjAutomaticly	boolean	Флаг автоматического удаления входного объекта
FWork	boolean	Флаг рабочего режима автомата
IfGoToPoint	PAtrData	Адрес условного перехода
SubStack	PAtrData	Стек адресов возврата из подпрограммы, стек объектов
InObjAdress	Pointer	Адрес, куда следует записывать входной объект
InObjPlase	PAtrData	Ячейка, куда следует помещать голову входного объекта

3.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	Set	Установить ссылку на миллипрограмму
111	SetRun	Установить ссылку на миллипрограмму и запустить ее на выполнение
2	InObpPop	Выдать ссылку на входной объект
3	InObpPopMk	Выдать милликоманду со ссылкой на входной объект
19	InObjDel	Удалить входной объект
Goto (безусловный переход)		
6	Goto	Перейти по адресу, хранящемуся в нагрузке милликоманды
7	GotoStop	Перейти по адресу, хранящемуся в нагрузке

		милликоманды и останов
4	SubEnter	Вход в подпрограмму
5	SubOut	Выход из подпрограммы
24	BusSet	Установить контекст шины
Branch (условный переход)		
8	BranchSet	Установить адрес предполагаемого условного перехода
9	GoTrue	Переход по адресу условного перехода, если в нагрузке хранится true
16	GoTrueStop	Переход по адресу условного перехода, если в нагрузке хранится true и останов
10	GoFalse	Переход по адресу условного перехода, если в нагрузке хранится false
17	GoFalseStop	Переход по адресу условного перехода, если в нагрузке хранится false и оствнов
11	SubTrue	Вызов подпрограммы,, если в нагрузке хранится true
12	SubFalse	Вызов подпрограммы,, если в нагрузке хранится false
80	FlagPointerSet	Установить ссылку для флага перехода
85	FlagPointerPop	Выдать указатель на флаг перехода
86	FlagPointerPopMk	Выдать МК с адресом флага перехода
90	FlagSet	Записать значение во флаг перехода
95	GoTrueAdr	Переход при установленном флаге по адресу
96	GoTrueAdrStop	Переход при установленном флаге по адресу и остановка
100	GoFalseAdr	Переход при сброшенном флаге по адресу
101	GoFalseAdrStop	Переход при сброшенном флаге по адресу и остановка
Prog (управление программой)		
13	Run	Запуск программы
14	Stop	Останов программы
23	ProgReset	Запустить программу с начала
25	ProgResetStop	Установить усказатель на текущую

		милликоманду на и останов
40	RunTrue	Запустить программу, если в нагрузке милликоманды храниться true
42	RunFalse	Запустить программу, если в нагрузке милликоманды храниться false
51	ClearStaticContext	Очистка статического контекста
52	AddLocalVar	Добавление локальной переменной
53	ClearLocalVars	Очистка массива локальных переменных
55	CopyIterationContext	Копирование контекста заданной итерации в статический контекст (либо контекста последней итерации)
Recursion (рекурсивный вызов программы)		
60	Recursion	Рекурсивный вызов (в нагрузке передается указатель на рекурсивную миллиограмму, при nil происходит рекурсивный вызов с начала основной программы)
61	RecursionStop	Рекурсивный вызов и останов (в нагрузке передается указатель на рекурсивную миллиограмму, при nil происходит рекурсивный вызов с начала основной программы)
65	CopyLocalVars	Копирование (восстановление) локальных переменных для текущей итерации
66	RecursionExit	Восстановление контекста и перменных
70	IterationNumberPop	Восстановление контекста и перменных
71	IterationNumberPopMk	Восстановление контекста и перменных

3.4 Примеры программирования

NewFU={Mnemo="Automat" FUType=FUAutomat}

Automat.SetRun={Console.OutLn="Hellow world !!!" \\ Запуск миллипрограммы

Console.OutLn="Hellow world !!!"

Automat.GotoStop=EndOfProg} \\ Переход с остановом

EndOfProg={Console.OutLn="End of program"} \\ миллипрограмма

Automat.Run \\ Милликоманда продолжения выполнения миллипрограммы

4 Диспетчер объектов (Object Manager)

4.1 Функциональное назначение

Создание и уничтожение информационных пар, запись и чтение значений из полей информационных пар (атрибут, данные, указатель). Формирование, модификация и уничтожение информационных капсул и ОА-деревьев.

4.2 Контекст

Наименование поля контекста	Тип	Описание
ID1	TPointIndex	Указатель на информационную пару, в которой осуществляется поиск
UkCoincidence	TPointIndex	Указатель на найденную информационную пару
UkCoincidenceSource	TPointIndex	Указатель на найденную информационную пару в источнике
NCoincidence	int64	Счетчик числа совпадающих информационных пар в результате поиска
IncludePointRange	int64	Поиск числовых интервалов (точечный интервал)
ExcludePointRange	int64	Поиск числовых интервалов (выколотая точка)
BegRange	int64	Поиск числовых интервалов (начало интервала)
FinRange	int64	Поиск числовых интервалов (конец интервала)
BusProgFlag	boolean	Флаг выдачи последовательности команд на Шину
ObjectAtr	int64	Атрибут для поиска ссылки на объект
ExeptAtr	int64	Атрибут для поиска ссылки на исключение
SuccessProg	Pointer	Указатель на подпрограмму, запускаемую при выполнении условия поиска
FailProg	Pointer	Указатель на подпрограмму, запускаемую при не выполнении условия поиска
InObj	TPointIndex	Входной объект

4.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
Find (поиск)		
1	Set	Принять капсулу для поиска
2	FindIc	Принять запись для сравнения
3	FindAnd	Поиск по правилу «И»
4	FindOr	Поиск по правилу «ИЛИ»
20	FindRange	Поиск интервалов
FindResult (результат поиска)		
5	Result	Выдать результат сравнения
6	ResultAnd	Выдать результат ""И"" (выдача логического результата сравнения)
7	ResultOr	Выдать результат ""ИЛИ"" (выдача логического результата сравнения)
8	IcPointerPop	Выдать указатель на первую найденную пару
9	AtrPointPop	Выдать указатель на поле ""атрибут"" первой найденной записи
209	AtrPointPopMk	Выдать милликоманду с указателем на поле "атрибут" первой найденной записи
10	DataPointerPop	Выдать указатель на поле ""данные"" первой найденной записи
210	DataPointerPopMk	Выдать милликоманду с указателем на поле ""данные"" первой найденной записи
11	PointPointerPop	Выдать указатель на поле ""Point"" первой найденной записи
211	PointPointerPopMk	Выдать милликоманду с указателем на поле ""Point"" первой найденной записи
12	PointPop	Выдать значение поля ""Point"" первой найденной записи
212	PointPopMk	Выдать милликоманду со значением поля ""Point"" первой найденной записи

14	AtrPop	Выдать атрибут из найденной записи
214	AtrPopMk	Выдать милликоманду с атрибутом из найденной записи
15	DataPop	Выдать данные из первой найденной записи
215	DataPopMk	Выдать милликоманду с данными из первой найденной записи
21	IcSourcePointerPop	Выдать указатель на найденную запись в источнике
50	Pop	Выдать указатель на капсулу, в которой производится поиск
54	PopMk	Выдать милликоманду с указателем на капсулу, в которой производится поиск
55	SourceIcPop	Выдать указатель на найденную ИП в источнике
59	SourceIcPopMk	Выдать милликоманду с найденной ИП в источнике
60	SourcePointerPop	Выдать указатель из нагрузки найденной ИП
64	SourcePointerPopMk	Выдать милликоманду с указателем из нагрузки найденной ИП
65	SourceDataPop	Выдать данные из найденной ИП в источнике
Switch (ветвление)		
104	ProgBusFlagSet	Установить флаг Выдачи последовательности милликоманд на Шину
108	AtrObjSet	Установить атрибут объекта
122	AtrExeptionSet	Установить атрибут исключения
35	SuccessProgSet	Установить указатель на программу, выполняемую в случае успешного поиска
40	FailProgSet	Установить указатель на программу, выполняемую в случае неуспешного поиска

104	ProgBusFlagSet	Установить флаг Выдачи последовательности милликоманд на Шину
Service (служебные)		
100	BusSet	Установить ссылку на Шину
450	PrefixProgSet	Установка ссылки на программу, выполняемую перед выполнением программы ФУ
455	PostfixProgSet	Установка ссылки на программу, выполняемую после выполнения программы ФУ
InObj (работа с входным объектом)		
45	InObjPop	Выдать указатель на входной объект
49	InObjPopMk	Выдать милликоманду с указателем на входной объект
165	InObjAtrPop	Выдать атрибут входного объекта
150	InObjAtrPopMk	Выдать милликоманду с атрибутом входного объекта
170	InObjDataPop	Выдать данные входного объекта
155	InObjDataPopMk	Выдать милликоманду с указателем на данные входного объекта
175	InObjPointPop	Выдать указатель на входной объект
160	InObjPointPopMk	Выдать указатель из нагрузки входной ИП

4.4 Примеры программирования:

\\ Создание ФУ

NewFU={Mnemo="FindBracket" FUType=FUFind}

FindBracket.Set={Separator="{"} \\Установить капсулу для поиска

FindBracket.SuccessProgSet={FU1.Mk=FU2.Mk} \\Установить указатель на

\\программу, выполняемую в случае совпадения с условием поиска

FindBracket.FailProgSet={Console.Out="'{' not founded!!!"} \\Указатель на подпрограмму?

\\выполняемую при несовпадении с условием поиска

\\Найти информационную пару и вывести ее нагрузку на консоль

FindFU.Set={Separator=";" Mnemo="and"}

FindFU. SuccessProgSet={FindFU.DataPopMk=Console.Out}

FindBracket.FailProgSet={Console.Out= "Mnemo not founded!!!"}

FindFU.FindIc={Mnemo=nil}

5 Счетчик (Counter)

5.1 Функциональное назначение

Изменение значения переменной от заданного начального значения до конечного значения с некоторым шагом. Запуск миллипрограмм при достижении переменной определенных значений.

5.2 Контекст

Наименование поля контекста	Тип	Описание
Counter	Variant	Счетчик
Hight, Bottom, Value	Variant	Верхняя и нижняя грани интервала и заданное значение
HightProg, BottomProg, ValueProg	Pointer	Указатели на программы, вызываемые по событиям больше или равно максимальному значению, меньше или равно меньшего значения, равно заданному значению
Step	Variant	Шаг счетчика

5.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс
1	Set	Установить значение
2	Pop	Выдать значение
10	PopMk	Выдать милликоманду со значением
4	Inc	Увеличить на шаг счетчика
5	Dec	Уменьшить на шаг счетчика
20	Add	Прибавить к значению
25	Sub	Вычесть из значения
30	StepSet	Установить шаг счетчика
55	ValueSet	Установить значение для запуска миллипрограммы
60	TopSet	Верхний уровень значений для запуска миллипрограммы

65	BottomSet	Нижний уровень значений для запуска миллипрограммы
70	ValueProgSet	Миллипрограмма, запускаемая при достижении заданной величины
75	TopProgSet	Миллипрограмма, запускаемая при привышении заданного верхнего порога
80	BottomProgSet	Миллипрограмма, запускаемая при достижении значения, меньшго нижней границы
926	BusSet	Установить контекст шины
995	ContextPop	Выдать контекст устройства
999	ContextPopMk	Выдать милликоманду с контекстом устройства

5.4 Примеры программирования

NewFU={Mnemo="Counter" FUType=FUCounter}

Counter.Set=1 \\ Установить значение счетчика

\\ Ссылка на программу, запускаемую при достижении счетчика определенного значения

Counter.ValueProgSet={Console.OutLn="Zero"}

Counter.ValueSet=0 \\ Задать значение, при котором запускается программа

Counter.Dec=1 \\ Уменьшить значение счетчика на 1

\\(т.к. счестчик будет равен 1, запустится программы и на консоль выведется "Zero"

6 Поиск (FindFU)

6.1 Функциональное назначение

Осуществляет поиск информационных пар в информационных капсулах. Также функциональное устройство вызывает миллипрограммы, соответствующие исходу поиска, выдает указатель на найденную информационную пару.

6.2 Контекст

Наименование поля контекста	Тип	Описание
ID1	TPointIndex	Указатель на информационную пару, в которой осуществляется поиск

UkCoincidence	TPointIndex	Указатель на найденную информационную пару
UkCoincidenceSource	TPointIndex	Указатель на найденную информационную пару в источнике
NCoincidence	int64	Счетчик числа совпадающих информационных пар в результате поиска
IncludePointRange	int64	Поиск числовых интервалов (точечный интервал)
ExcludePointRange	int64	Поиск числовых интервалов (выколотая точка)
BegRange	int64	Поиск числовых интервалов (начало интервала)
FinRange	int64	Поиск числовых интервалов (конец интервала)
BusProgFlag	boolean	Флаг выдачи последовательности команд на Шину
ObjectAtr	int64	Атрибут для поиска ссылки на объект
ExeptAtr	int64	Атрибут для поиска ссылки на исключение
SuccessProg	Pointer	Указатель на подпрограмму, запускаемую при выполнении условия поиска
FailProg	Pointer	Указатель на подпрограмму, запускаемую при не выполнении условия поиска
InObj	TPointIndex	Входной объект

6.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
Find (поиск)		
1	Set	Принять капсулу для поиска
2	FindIc	Принять запись для сравнения
3	FindAnd	Поиск по правилу «И»
4	FindOr	Поиск по правилу «ИЛИ»

20	FindRange	Поиск интервалов
FindResult (результат поиска)		
5	Result	Выдать результат сравнения
6	ResultAnd	Выдать результат ""И"" (выдача логического результата сравнения)
7	ResultOr	Выдать результат ""ИЛИ"" (выдача логического результата сравнения)
8	IcPointerPop	Выдать указатель на первую найденную пару
9	AtrPointPop	Выдать указатель на поле ""атрибут"" первой найденной записи
209	AtrPointPopMk	Выдать милликоманду с указателем на поле "атрибут" первой найденной записи
10	DataPointerPop	Выдать указатель на поле ""данные"" первой найденной записи
210	DataPointerPopMk	Выдать милликоманду с указателем на поле ""данные"" первой найденной записи
11	PointPointerPop	Выдать указатель на поле ""Point"" первой найденной записи
211	PointPointerPopMk	Выдать милликоманду с указателем на поле ""Point"" первой найденной записи
12	PointPop	Выдать значение поля ""Point"" первой найденной записи

212	PointPopMk	Выдать милликоманду со значением поля ""Point"" первой найденной записи
14	AtrPop	Выдать атрибут из найденной записи
214	AtrPopMk	Выдать милликоманду с атрибутом из найденной записи
15	DataPop	Выдать данные из первой найденной записи
215	DataPopMk	Выдать милликоманду с данными из первой найденной записи
21	IcSourcePointerPop	Выдать указатель на найденную запись в источнике
50	Pop	Выдать указатель на капсулу, в которой производится поиск
54	PopMk	Выдать милликоманду с указателем на капсулу, в которой производится поиск
55	SourceIcPop	Выдать указатель на найденную ИП в источнике
59	SourceIcPopMk	Выдать милликоманду с найденной ИП в источнике
60	SourcePointerPop	Выдать указатель из нагрузки найденной ИП
64	SourcePointerPopMk	Выдать милликоманду с указателем из нагрузки найденной ИП
65	SourceDataPop	Выдать данные из найденной ИП в источнике
Switch (ветвление программы)		

104	ProgBusFlagSet	Установить флаг Выдачи последовательности милликоманд на Шину
108	AtrObjSet	Установить атрибут объекта
122	AtrExeptionSet	Установить атрибут исключения
35	SuccessProgSet	Установить указатель на программу, выполняемую в случае успешного поиска
40	FailProgSet	Установить указатель на программу, выполняемую в случае неуспешного поиска
104	ProgBusFlagSet	Установить флаг Выдачи последовательности милликоманд на Шину
Service (служебные)		
100	BusSet	Установить ссылку на Шину
450	PrefixProgSet	Установка ссылки на программу, выполняемую перед выполнением программы ФУ
455	PostfixProgSet	Установка ссылки на программу, выполняемую после выполнения программы ФУ
InObj (работа с входным объектом)		
45	InObjPop	Выдать указатель на входной объект
49	InObjPopMk	Выдать милликоманду с указателем на входной объект
165	InObjAtrPop	Выдать атрибут входного объекта
150	InObjAtrPopMk	Выдать милликоманду с

		атрибутом входного объекта
170	InObjDataPop	Выдать данные входного объекта
155	InObjDataPopMk	Выдать милликоманду с указателем на данные входного объекта
175	InObjPointPop	Выдать указатель на входной объект
160	InObjPointPopMk	Выдать указатель из нагрузки входной ИП

6.4 Примеры программирования;

\\ Создание ФУ

NewFU={Mnemo="FindBracket" FUType=FUFind}

FindBracket.Set={Separator="{"} \\Установить капсулу для поиска

FindBracket.SuccessProgSet={FU1.Mk=FU2.Mk} \\Установить указатель на

\\программу, выполняемую в случае совпадения с условием поиска

FindBracket.FailProgSet={Console.Out="'{' not founded!!!"} \\Указатель на подпрограмму?

\\выполняемую при несовпадении с условием поиска

\\Найти информационную пару и вывести ее нагрузку на консоль

FindFU.Set={Separator=";" Mnemo="and"}

FindFU. SuccessProgSet={FindFU.DataPopMk=Console.Out}

FindBracket.FailProgSet={Console.Out= "Mnemo not founded!!!"}

FindFU.FindIc={Mnemo=nil}

7 Список (ListFU)

7.1 Функциональное назначение

Формирование и модернизация списка капсул. Поиск по списку капсул и вызов миллипрограмм в зависимости от результата поиска.

7.2 Контекст

Наименование поля контекста	Тип	Описание
ListHead, ListTail	PAtrData	Начало списка, конец списка
ObjectRec,ExeptRec	TAtrData	Запись для поиска объекта и ссылки на ошибку в найденной капсуле
RecUk, UkCoincidenceSource, LineUk	TPointIndex	Указатели на найденную строку/запись
NCoincidenceLines, NCoincidenceRecs, NFirstFindLine	int64	Количество найденных строк/записей; номер первой найденной строки
FindFU	PFindContext	Собственное поисковое устройство
ListCounter	Variant	Счетчик строк в списке / Номер первой найденной строки
ErrorMessageBuf	string	Сообщение об ошибке
FailLink	TPointIndex	Ссылка на объект, которая выдается, если не найдена строка при поиске
FindLineFlag	boolean	Флаг сравнения номера найденной строки
BegRange, FinRange, IncludePointRange, ExcludePointRange	int64	Атрибуты начала, конца диапазона и отдельной точки, исключенной точки (для поиск интервалов)
IncludeF	boolean	Флаг вхождения точки в интервал
FindMode	int64	Режим поиска (0 - ищется только строка, 1 - Поиск с выдачей на Шину // 2 - поиск интервалов по ИП, 3 - поиск интервалов с выдачей на Шину по ИП // 4 - поиск Интервалов по значению, 5 - поиск интервалов по значению и выдача на Шину
FindQuantityMode	int64	Режим поиска количества строк (0 – поиск одной записи, 1 – поиск всех возможных записей)
DownUpFind	int64	Направление поиска (0 -сверху вниз, 1 - снизу вверх)

EnterObj	TPointIndex	Ссылка на входной объект
BusMode	Variant	Флаг выдачи программы на шины
ListSetMode	Variant	Режим установки ссылки на список (0- проверка хвоста списка, 1 - без проверки конца списка
RoutObj	TPointIndex	Ссылка на объект для маршрутизации
BusMk	TAtrAndPoint	Милликоманда, выдаваемая при совпадении условия поиска (выдаваемое сообщение помещается в нагрузку этой милликоманды)
FailProg, SuccessProg	Pointer	Ссылка на программу, выполняемую при неудачном и удачном поиске
NextFU	TContextAndMk	Миллидиапазон следующего ФУ для обработки
IndexCounter	PVariant	Индексный счётчик
IndexVector	TIndexVector	Указатель на массив индексов
IndexMode	boolean	Флаг индексного режима
PartialReset	Pointer	Метка для частичного сброса

7.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
List (список)		
2	Set	Задать ссылку на список
1	Del	Удалить список и входящие в него капсулы
4	LineAdd	Добавить новую строку в список
144	LineCopyAdd	Добавить новую строку в список
4	LineAdd	Добавить новую строку в список
28	FailObjSet	Назначить ссылку на объект, которая выдается в случае неудовлетворения условиям поиска
13	LineNextPop	Выдать указатель на следующую

		строку после первой найденной строки
17	ObjPop	Выдать количество найденных строк
18	LineExclude	Исключить из списка строку, на которую указывает метка в Obj
19	LineDel	Удалить из списка строку, на которую указывает метка в Obj
65	LineEndDel	Удалить из списка последнюю строку
220	Pop	Выдать указатель на начало списка
43	PopMk	Выдать милликоманду с указателем на начало списка
31	LineLastPop	Выдать указатель на конец списка
151	LineLastPopMk	Выдать милликоманду на последнюю строку списка
33	LineLastMarkPop	Выдать указатель на метку последней строки списка
36	LineLastPopExclude	Выдать последнюю строку и удалить её из списка
136	LineLastPopMkExclude	Выдать указатель на последнюю строку списка и исключить из списка
200	LineLastPopMkDel	Выдать указатель на последнюю строку списка и удалить ее
66	CurrentLineSet	Выдать флаг пустого списка
67	LineNext	Перевести указатель на текущей строки на следующую строку списка
68	LinePrev	Перевести указатель текущей строки на предыдущую строку
66	LineSet	Установить указатель на текущую строку
Find (поиск по списку)		
5	FindIc	Поиск одной ИП
6	FindAnd	Поиск по правилу "И"
7	FindOr	Поиск по правилу "ИЛИ"
50	FindRange	Поиск интервалов
60	FindIcInLastLine	Поиск ИП в последней строке

61	FindAndInLastLine	Поиск "И" в последней строке
62	FindOrInLastLine	Поиск "ИЛИ" в последней строке
73	FindIcInLine	Поиск ИП в найденной/текущей строке
74	FindAndInLine	Поиск"И" в найденной/текущей строке
75	FindOrInLine	Поиск "ИЛИ" в найденной/текущей строке
76	FinIntervInLine	Поиск интервалов в найденной/текущей строке
180	FindCapsIc	Поиск в капсуле по нескольким ИП
FindResult (результат поиска)		
8	Result	Найдено ли (выдается "true", если найдено
9	ResultAnd	Выдать логический результат поиска по правилу "И"
10	ResultOr	Выдать логический результат поиска по правилу "ИЛИ"
81	IcPointSourcePop	Выдать ссылку на найденную пару в источнике
93	IcPointSourcePopMk	Выдать ссылку на найденную пару в источнике
11	IcPointPop	Выдать указатель на первую найденную пару
12	LinePop	Выдать указатель на первую найденную строку
14	LineNumPop	Выдать номер первой найденной строки
21	AtrPointPop	Выдать указатель на атрибут найденной записи
22	DataPointPop	Выдать указатель на данные найденной записи
23	PointerPointPop	Выдать указатель на данные найденной записи
25	AtrPop	Выдать атрибут из первой найденной записи

26	DataPop	Выдать данные из первой найденной записи
87	SourseDataPop	Выдать/записать данные найденной записи в источнике
170	DelFindIcLines	Удалить строки с заданной ИП (удаляется все дерево)
InObj(работа с входным объектом)		
102	InObjPop	Выдать входной объект
103	InObjPopMk	Выдать милликоманду с указателем на входной объект
202	InObjAtrPop	Выдать атрибут входного объекта
204	InObjAtrPopMk	Выдать милликоманду с атрибутом входной ИП
206	InObjDataPop	Выдать данные входного объекта
208	InObjDataPopMk	Выдать милликоманду с указателем на данные входного объекта
210	InObjPointPop	Выдать указатель на входной объект
212	InObjPointPopMk	Выдать указатель из нагрузки входной ИП
102	InObjPop	Выдать входной объект
Service (служебные)		
70	ModeFindSet	Установить режим поиска
72	DirectionFindSet	Установить направление поиска
51	BusSet	Установить контекст шины
3	AtrObjSet	Задать атрибут ссылки на объект
106	BusModeSet	Установить режим выдачи сообщения на шину
430	NextFUMkRangeSet	Установить миллидапазон для дублирования милликоманды для следующего ФУ
435	NextFUSet	Установить контекст следующего ФУ
112	Rout	Маршрутизация
116	RoutMkPointPop	Выдать ссылку на милликоманду для маршрутизации

130	IndexdVectSet	Задать ссылку на индексный массив
132	IndexVectPop	Выдать ссылку на индексный массив
135	IndexCountSet	Задать ссылку на индексный счётчик
137	IndexVectPop	Выдать ссылку на индексный счётчик
140	IndexVectCreate	Сформировать индексный массив
400	BusMkSet	Установить милликоманду, выдаваемую на шину
410	BusMkPointerSet	Установить указатель на милликоманду, выдаваемую на шину
240	PartialResetSet	Установить позицию для частичного сброса
245	PartialReset	Частичный сброс (из списка удаляются все капсулы послед капсулы, установленной милликомандой "PartialResetSet" и до конца списка)
450	PrefixProgSet	Установить префиксную программу
455	PostfixProgSet	Установить постификсную программу
Common (общие)		
995	ContextPop	Выдать контекст
999	ContextPopMk	Выдать милликоманду с контекстом

7.4 Примеры программирования

Bus{AfterIc.BusModeSet=2} \\ Установка режима работы ФУ Список

AfterIc.Set= \\ Установить ссылку на список (далее идет описание списка)

\\ знак «>» обозначает начало новой капсулы, входящей в список

>{Separator="}" SyntaxStack.LineLastPopMkExclude=AfterCapsule.FindIc} *Нет уничтожения капсулы, выданной из стека*\

>{Separator=";" CapsuleManager.IcNewAttach Lexica.ReceiverMkSet=IcAnalysis.FindIc}

>{Separator="+" IcPlus.LineSet=nil Lexica.ReceiverMkSet=IcPlus.FindIcInLine}

>{0=nil CapsuleManager.IcNewAttach AfterIc.InObjPopMk=IcAnalysis.FindIc}

AfterIc. FindIc={ Separator=";" } // Поиск одной ИП в списке

8 Логическое АЛУ (BoolALU)

8.1 Функциональное назначение

1. Осуществление логических операций
2. Запуск подпрограмм в зависимости от результата логической операции

8.2 Контекст

Наименование поля контекста	Тип	Описание
accumulator	Variant	Аккумулятор (содержит один из операндов). В аккумулятор помещается результат операции.
TrueProg, FalseProg	Pointer	Указатели на подпрограммы, выполняемые при полученее true, false в аккумуляторе

8.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	Set	Записать значение в аккумулятор
3	InvSet	Записать в аккумулятор инвертированную величину
7	Pop	Выдать значение из аккумулятора
8	PopMk	Выдать милликоманду со значением из аккумулятора
9	PopInv	Выдать инвертированное значение из аккумулятора
11	Inv	Отрицать значение аккумулятора и выдать его содержимое
13	And	Логическое И
15	Or	Логическое ИЛИ
17	ExeptOr	Логическое ИсключающееИЛИ
19	AndNot	Лоигеское И-НЕ
21	OrNot	Логическое ИЛИ-НЕ
50	TruePogSet	Установить ссылку на программу, выполняемую по true
53	FalseProgSet	Установить ссылку на программу,

		выполняему по false
55	TrueExec	Выполнить программу по true
57	FalseExec	Выполнить программу по false

8.4 Примеры программирования

NewFU={Mnemo="BoolALU" FUType=FUBoolALU}

\\ Установить ссылку на программу, выполняемую при значении True в аккумуляторе
BoolALU.TruePogSet={Console.OutLn="True"}
BoolALU.Set=false \\ Установить значение аккумулятора
BoolALU.AndNot=true \\ Операция И-НЕ (в аккумулятор будет записано true,
\\ Запуститься программа и на консоль выдется надпись "True"

9 Дробное АЛУ (FloatALU)

9.1 Функциональное назначение

1. Осуществление арифметических операций с дробными числами
2. Выдача флага результата операции
3. Вызов подпрограмм в зависимости от результата операции

9.2 Контекст

Наименование поля контекста	Тип	Описание
accumulator	Variant	Аккумулятор (содержит один из операндов). В аккумулятор помещается результат операции.
ZProg,NZProg,BProg, BZProg,SProg,SZProg	Pointer	Миллипрограммы, запускаемые в зависимости от результата вычислений
FZero, FLager	boolean	Флаги нуля и больше нуля

9.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	Set	Записать значение в аккумулятор
2	Pop	Выдать текущий указатель

4	Opposite	Обратная величина
5	Round	Округлить значение
6	Int	Целая часть значения в аккумуляторе
7	Sqrt	Извлечь квадратный корень из величины в аккумуляторе
8	Sqr	Возвести значение в аккумуляторе в квадрат
9	Add	Сложить с аккумулятром
50	SqrAdd	Возвести входную величину в квадрат и прибавить к значению в аккумуляторе
10	Sub	Вычесть текущее значение из значения в аккумуляторе
51	SqrSub	Возвести входную величину в квадрат и вычесть из значения в аккумуляторе
11	Mul	Умножить
12	Div	Разделить
21	Cmp	Сравнить (вычесть из аккумулятора значение на входе и установить флаги)
13	FlagZero	Выдать флаг нуля
31	FlagNotZero	Выдать флаг «не нуль»
30	Abs	Абсолютное значение
40	IntPop	Выдать целое число
Flags		
14	FlagSign	Выдать флаг знака числа
15	FlagGreaterOrEqual	Флаг больше или равно
16	FlagLess	Флаг меньше
17	FlagLessOrEqual	Флаг меньше или равно
18	FlagGreater	Флаг больше
ProgByFlags		
200	ZeroProgSet	Программа, запускаемая по флагу нуля
205	NotZeroProgSet	Программа, запускаемая по флагу "не нуль"
210	GreaterProgSet	Программа, запускаемая по флагу "больше"

215	GreaterZeroProgSet	Программа, запускаемая по флагу "больше или нуль"
220	LessProgSet	Программа, запускаемая по флагу "меньше"
225	LessZeroProgSet	Программа, запускаемая по флагу "меньше или нуль"
ProgExec		
230	ZeroProgExec	Выполнить программу по флагу нуля
235	NotZeroProgExec	Выполнить программу по флагу "не нуль"
240	GreaterProgExec	Выполнить программа по флагу "больше"
245	GreaterZeroProgExec	Выполнить программа по флагу "больше или нуль"
250	LessProgExec	Выполнить программу по флагу "меньше"
255	LessZeroProgExec	Выполнить программу по флагу "меньше или нуль"

9.4 Примеры программирования

NewFU={Mnemo="FloatALU" FUType=FUFloatALU}

FloatALU.Set=1 \\ Установить значение

FloatALU.Sub=1 \\ Вычесть

FloatALU.ZeroProgExec={Console.OutLn="Zero"} \\ Запуск программы по флагу нуля

10 Целочисленное АЛУ (IntALU)

10.1 Функциональное назначение

1. Осуществление арифметических операций с целыми числами
2. Выдача флага результата операции
3. Вызов подпрограмм в зависимости от результата операции

10.2 Контекст

Наименование поля контекста	Тип	Описание
accumulator	variant	Аккумулятор (содержит один из операндов). В аккумулятор помещается результат операции.
remainder	variant	Остаток от целочисленного деления
ZProg,NZProg,BProg, BZProg,SProg,SZProg	Pointer	Миллипрограммы, запускаемые в зависимости от результата вычислений
FZero, FLager	boolean	Флаги нуля и больше нуля

10.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	Set	Записать значение в аккумулятор
2	Pop	Выдать текущий указатель
42	PopMk	Выдать милликоманду со значением из аккумулятора
6	Add	Сложить
7	Sub	Вычесть
8	Mul	Умножиить
9	Div	Разделить
18	Cmp	Сравнить (вычесть из аккумулятора значение на входе и установить флаги)
Flags		
10	FlagZero	Выдать флаг нуля
22	FlagNotZero	Выдать флаг «не нуль»
11	SignFlag	Выдать флаг знака числа
21	FlagGreaterOrEqual	Флаг больше или равно
15	FlagGreater	Флаг больше
13	FlagLess	Флаг меньше
14	FlagLessOrEqual	Флаг меньше или равно

ProgByFlags		
200	ZeroProgSet	Программа, запускаемая по флагу нуля
205	NotZeroProgSet	Программа, запускаемая по флагу "не нуль"
210	BiggerProgSet	Программа, запускаемая по флагу "больше"
215	BiggerZeroProgSet	Программа, запускаемая по флагу "больше или нуль"
220	LessProgSet	Программа, запускаемая по флагу "меньше"
225	LessZeroProgSet	Программа, запускаемая по флагу "меньше или нуль"
ProgExec		
230	ZeroProgExec	Выполнить программу по флагу нуля
235	NotZeroProgExec	Выполнить программу по флагу "не нуль"
240	BiggerProgExec	Выполнить программа по флагу "больше"
245	BiggerZeroProgExec	Выполнить программа по флагу "больше или нуль"
250	LessProgExec	Выполнить программу по флагу "меньше"
255	LessZeroProgExec	Выполнить программу по флагу "меньше или нуль"
4	SignInvers	Изменить знак числа в аккумуляторе
5	Sqr	Возвести значение в аккумуляторе в квадрат
16	AtrPop	Выдать значение в формате int64
17	ModPop	Вычислить остаток от деления
30	Abs	Абсолютное значение

10.4 Примеры программирования

NewFU={Mnemo="IntALU" FUType=FUIntALU}

IntALU.Set=10

\\ подпрограмма, запускаемая при установленном флаге "не нуль"

IntALU.NotZeroProgSet={IntALU.PopMk=Console.OutLn IntALU.Sub=1}

IntALU.Sub=0 \\ Операция для установки флагов и запуска подпрограммы

\\ В результате выполнения программы на консоль выведутся числа от 10 до 1

11 Диспетчер переменных (VarFU)

11.1 Функциональное назначение

1. Создание и уничтожение переменных (констант, указателей)
2. Запись значения в переменную
3. Чтение значения из переменных

11.2 Контекст

Наименование поля контекста	Тип	Описание
VarPoint	TPointIndex	Указатель
Flag	boolean	Флаг, устанавливаемый милликомандой сравнения указателей
IndexMode	boolean	Флаг индексного режима
IndexVector	TIndexVector	Массив индексных пар (нужен для работы в индексном режиме)
IndexCounter	PVariant	Указатель на контекст ФУ счетчика Индексов (для индексного режима)

11.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
3	Set	Установить текущий указатель (адрес)
53	Read	Прочить текущий указатель
4	Pop	Выдать текущий указатель

54	PopMk	Выдать милликоманду с текущим указателем
55	Write	Записать текущий указатель
VarWork		
1	VarNewSet	Создать новую милликоманду и установить ее значение
2	VarDel	Удалить переменную с текущим адресом
5	VarSet	Установить переменную, с текущим адресом
65	VarRead	Прочитать переменную в ячейку память с текущим адресом
6	VarPop	Выдать переменную, находящуюся по текущему адресу
66	VarWrite	Записать переменную, находящуюся по текущему адресу
PointWork		
7	PointNewSet	Создать новый указатель и записать в него значение
8	PointDel	Удалить указатель по текущему адресу
13	PointCompare	Сравнить указатель с текущим указателем
15	FlagPop	Выдать флаг сравнения текущего адреса, произведенного по милликоманде "PointCompare"
14	PointCompareNil	Проверить текущий указатель на nil и выдать результат сравнения
16	PointSet	Установить значение указателя, расположенному по текущему адресу
22	PointPopPointerSet	Прочитать адрес ячейки памяти и записать указатель по прочитанному адресу
23	PointPopObjSet	Прочитать адрес ячейки памяти и записать указатель по адресу,

		хранящемуся в нагрузке милликоманды
AtrWork		
18	AtrSet	Установить атрибут по адресу, который хранится в текущем указателе
19	AtrPop	Выдать атриубут, хранящийся по текущему адресу
20	AtrPopAtrSet	Выдать атриубут, хранящейся по текущему адресу и записать его по адресу, хранящемуся в нагрузке милликоманды
21	AtrPopVarSet	Выдать атриубут, хранящейся по текущему адресу и записать его в Variant-пременную по адресу, хранящемуся в нагрузке милликоманды

11.4 Примеры программирования

NewFU={Mnemo="Var" FUType=FUVarManager}

VarPoint(nil) \\ Объявление указателя
Var.VarNewSet=0 \\ Создать новую переменную и установить ее значение
Var.Pop=VarPoint \\ Запись указателя в VarPoint

12 Матрица указателей (PointMatr)

12.1 Функциональное назначение

Создание, уничтожение, модификация указателей, записанных в матрицу.

12.2 Контекст

Наименование поля контекста	Тип	Описание
Pointers	array of TPointIndex	Вектор указателей
PointMass	PPointMatrix	Матрица указателей
Xindex, Yindex	int64	Индекс текущего элемента массива
MoveType	int64	Тип автоматического сдвига индекса (0-без сдвига; 1-сдвиг

		по X; 2-сдвиг по Y)
SizeX,SizeY	int64	Размерность массива по X, по Y
BegFlagLine, EndFlagLine, BegFlagMass, EndFlagMass	boolean	Флаги индекса массива (флаг, если индекс в начале массива; флаг, если индекс находится в конце массива)
IndexMode	Variant	Индексный режим

12.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
100	Reset	Сброс ФУ
111	Create	Создать массив указателей (в нагрузке милликоманды передается адрес, который записывается во все ячейки матрицы)
112	Ini	Инициализировать массив (в нагрузке милликоманды передается адрес, который записывается во все ячейки матрицы)
205	Atribyting	Атрибутировать матрицу (в нагрузке передается указатель на матрицу с атрибутами милликоманд)
300	ProgToNilRun	Выполнять программу до тех пор, пока не в текущей ячейке на встретиться nil (после выполнения программы, индекс изменяется автоматически)
75	FindInCol	Поиск по строке
80	FindInRow	Поиск по столбцу
103	RowSet	Установить номер текущей строки
107	RowPop	Выдать номер текущей строки
104	ColSet	Установить номер текущего столбца
108	ColPop	Выдать номер текущего столбца
105	IndexModeSet	Установить режим автоматического

		изменения индексов (0 – без сдвига, 1 – сдвиг вправо по строке, 2- сдвиг вниз по столбцу)
119	RowCountPop	Выдать число строк в матрице
20	ColCountPop	Выдать число столбцов в матрице
113	CellPop	Выдать значение текущей ячейки
270	CellPopMk	Выдать милликоманду со значением текущей ячейки
214	CellSet	Установить значение в текущую ячейку
115	MatrDel	Удалить матрицу
116	Copy	Копировать матрицу
101	MatrSet	Установить указатель на матрицу
117	MatrPop	Выдать указатель на матрицу
170	RowMatrPop	Выдать указатель на строку матрицы

12.4 Примеры программирования

FuTypeMkTrees.RowSet=0 \\ Установить текущий номер строки

FuTypeMkTrees.IndexModeSet=2 \\ Установить индексный режим

// Установить значение в ячейку матрицы

FuTypeMkTrees.CellSet=ZeroFUMkList

13 Диспетчер матрицы констант (ConstMatr)

13.1 Функциональное назначение

Создание, редактирование матрицы, содержащей константные значения.

13.2 Контекст

Наименование поля контекста	Тип	Описание
accumulator	variant	Аккумулятор (содержит один из операндов). В аккумулятор помещается результат операции.
ZProg,NZProg,BProg,BZProg,SProg,SZProg	Pointer	Миллипрограммы, запускаемые в зависимости от результата вычислений
FZero, FLager	boolean	Флаги нуля и больше нуля

13.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
105	Atribyting	Атрибутирование матрицы (в нагрузке милликоманды передается указатель на числовую матрицу с индексами милликоманд)
85	ColOffsetSet	Установить смещение индекса колонки для чтения данных из ячейки массива
90	RowOffsetSet	Установить смещение индекса строки для чтения данных из ячейки массива
72	SearchModeSet	Установить режим поиска (0-обычный поиск, 1 – бинарный поиск)
75	SearchInCol	Поиск по колонке
80	SearchInRow	Поиск по строке
200	SearchSuccessProgSet	Установить ссылку на подпрограмму, запускаемую при удачном исходе поиска
205	SearchFailProgSet	Установить ссылку на подпрограмму, запускаемую при неудачном исходе поиска
130	SortIncreaceByRow	Сортировать по возрастанию по строке
132	SortDecreaceByRow	Сортировать по убыванию по строке
134	SortIncreaceByCol	Сортировать по возрастанию по столбцу
136	SortDecreaceByCol	Сортировать по убыванию по столбцу
1	MatrSet	Установить указатель на матрицу
6	Matr2Set	Установить указатель на вторую матрицу
2	RezSet	Установить указатель на результирующую матрицу (в матрицу помещается результат операции)

17	MatrPop	Выдать указатель на матрицу
160	MatrPopMk	Выдать милликоманду с указателем на матрицу
27	Matr2Pop	Выдать указатель на вторую матрицу
28	RezPop	Выдать указатель на результирующую матрицу
165	RezPopMk	Выдать милликоманду с указателем на результирующую матрицу
15	MatrDel	Удалить матрицу
11	Create	Создать матрицу (в нагрузке передается значение, которое присваивается всем ячейкам матрицы)
32	CreateUnitary	Создать единичную матрицу
12	Ini	Инициализация массива (в нагрузке передается значение, которое присваивается всем ячейкам матрицы)
16	Copy	Копирование матрицы
50	FileLoad	Считать матрицу из файла
55	FileSave	Сохранить матрицу в файл
19	RowCountPop	Выдать число строк в матрице
20	ColCountPop	Выдать число столбцов в матрице
26	UnitryMatr	Сделать матрицу единичной
70	RowMatrPop	Выдать строку матрицы
74	LineRezPop	Выдать строку результирующей матрицы
3	RowSet	Установить текущий номер строки
7	RowPop	Выдать текущий номер строки
37	RowPopMk	Выдать милликоманду ч текущим номером строки
4	ColSet	Установить текущий номер колонки
8	ColPop	Выдать текущий номер колонки
38	ColPopMk	Выдать милликоманду с текущим номером колонки
5	IndexModeSet	Установить режим автоматического

		сдвига текущего индекса (0 – без сдвига, 1 – сдвиг вправо по строке, 2- сдвиг вниз по столбцу)
140	SizeModeSet	Установить решим автоматического изменения размера матрицы (0 – без изменения, 1 – размер матрицы подстраивается под текущий индекс)
13	CellPop	Выдать содержимое ячейки матрицы с текущими координатами
170	CellPopMk	Выдать милликоманду с содержимым ячейки матрицы с текущими координатами
31	CellResPop	Выдать содержимое ячейки результирующей матрицы с текущими координатами
175	CellResPopMk	Выдать милликоманду с содержимым ячейки результирующей матрицы с текущими координатами
60	CellAdd	Прибавить к текущей ячейке матрицы
65	CellSub	Вычесть из текущей ячейки матрицы
14	CellSet	Установить значение в текущую ячейку матрицы

13.4 Примеры программирования

FuTypeIndexes.RowSet=0 \\ Установить текущий номер строки

FuTypeIndexes.CellSet=0 \\ Установить значение в текущую ячейку матрицы

14 OpenGL

14.1 Функциональное назначение

Реализация языка графического программирования OpenGL

14.2 Контекст

Наименование поля контекста	Тип	Описание
Qtype	(None, Sphere, Cylinder, Disk, PartialDisk)	Тип 3-мерного графического объекта

radius1, radius2	GLfloat	Первый и второй радиусы для 3-мерных графических объектов
height	GLfloat	
Angle1, Angle2	GLint	Первый и второй углы для 3-мерных графических объектов

14.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	FormCreate	Создать форму
2	StageClear	Очистить сцену
10	StageDraw	Вывод картинки
52	NormalSet	Установить нормаль для освещения
Perspective (перспектива)		
30	PerspLeft	Задать левую границу перспективы
31	PerspRight	Задать правую границу перспективы
32	PerspBottom	Задать нижнюю границу перспективы
33	PerspTop	Задать верхнюю границу перспективы
34	PerspClose	Задать ближнюю грань перспективы
35	PerspDist	Задать дальнюю грань перспективы
36	PerspProjSet	Создать перспективную проекцию
43	OrthoView	Задать ортогональную проекцию
Turn (поворот изображения)		
30	TurnX	Поворот по X
38	TurnY	Поворот по Y
39	TurnZ	Поворот по Z
Move (передвижение изображения)		
40	MoveX	Перенос по X
41	MoveY	Перенос по Y
42	MoveZ	Перенос по Z
3DPrimitive (3D-примитивы)		
60	New3DPrimitive	Новый 3Д примитив
61	Choose3DPrimitive	Выбор активного 3Д примитива

62	Primitive3DTypeSet	Задать тип 3Д примитива
63	Radius1	Радиус 1
64	Radius2	Радиус 2
65	Corner1	Угол 1
66	Corner2	Угол 2
67	QuantitySegmentHoriz	Количество разбиений поперёк оси Z
68	QuantitySegmentVert	Количество разбиений вдоль оси Z
69	HeightSet	Высота
70	Reflect3D	Отображение 3Д примитива
Light (освещение)		
71	LightPositionSet	Позиция света
72	LightDirectionSet	Направление света
50	TurnOnLight	Включить свет
51	TurnOffLight	Выключить свет
Color (цвет)		
11	Red	Красный
12	Green	Зеленый
13	Blue	Синий
14	ColorSet	Установить цвет
14	BackgroundColorSet	Установить цвет фона
Massive (работа с массивами)		
17	MassiveSet	
44	MatrIndexSet	Установить матричный индекс
45	MatrCounterSet	Установить матричный счетчик
8	MassiveDraw	Вывести матрицу (если на входе указатель на матрицу, то выводится вся матрица)
15	IndexSet	Установить индекс
16	CounterSet	Установить счетчик
Vertex (работа с точками)		
5	XSet	Задать X
6	YSet	Задать Y
7	ZSet	Задать Z
9	VertexSet	Задать вершину

Primitive (работа с графическими примитивами)		
3	Fug2DDraw	Рисовить 2мертый примитив
4	EndDraw	Выйти из режима рисования 2Д примитива

14.4 Примеры программирования

NewFU={Mnemo="OpenGL" FUType=FUOpenGL}

OpenGL.FormCreate \\ Создать форму для вывода изображения
OpenGL.StageClear \\ Очистить сцену
OpenGL.BackgroundColorSet=clLightGrey \\ Установить цвет фона

OpenGL.Fug2DDraw=4 \\ Начало рисования 2-мерного примитива
OpenGL.XSet=0 \\ Координаты первой точки
OpenGL.YSet=0
OpenGL.ZSet=0
OpenGL.VertexSet

OpenGL.XSet=10 \\ Координаты второй точки
OpenGL.YSet=20
OpenGL.ZSet=30
OpenGL.VertexSet

OpenGL.EndDraw \\ конец рисования 2-мерного примитива

15 График (Plot)

15.1 Функциональное назначение

Построение графика

15.2 Контекст

Наименование поля контекста	Тип	Описание
Color	TColor	Цвет поля графика
Graph	TChartSeries	Класс, отображающий на экране одну функцию

TChartSeries	int64	Количество выводимых на экран точек
NotInit	boolean	Флаг, обозначающий, что не поступало ни одной точки для вывода
X	double	Текущая координата, в которой выводится точка графика
M	array of array of Variant	Массив выводимых точек графика
StepX	double	Шаг значений по оси X
Xtemp	double	Текущее значение X (для вывода по шагу)
LastIndex	integer	Индекс последней точки (нужно для удаления точек

15.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	SeriaCreat	Создать Серию (на входе номер вида серии)
2	SeriaPointerPop	Выдать ссылку на серию
4	SeriaPointerPopMk	Выдать милликоманду со ссылкой на серию
3	ParentSet	Задать ссылку на родителя
5	SeriaDel	Уничтожить График
6	NPointersSet	Задать максимальное количество выводимых точек
10	ColorSet	Задать Цвет
50	XSet	Задать координату X
64	YSet	Задать координату Y и вывести точку на график
70	StepSet	Задать шаг
74	XCurrentSet	Установить текущую координату X

78	YByStepSet	Вывести значение Y соотвественно шагу
80	PenWidthSet	Установить ширину линии графика
84	PenColorSet	Установить Цвет линии

15.4 Примеры программирования

NewFU={Mnemo="Chart" FUType=FUChart}

NewFU={Mnemo="Plot" FUType=FUPlot}

Plot.SeriaCreat \\ Создание серии графиков

Chart.ParentSet \\ Создание отдельного окна для вывода графика

Chart.Caption="Parallel factor" \\ Установка заголовка окна диаграммы

Chart.Focus \\ Дать компоненту фокус ввода

Plot.SeriaCreat \\ Создать серию графика

Chart.ChartPopMk=Plot.ParentSet \\ Установка родительского компонента для графика

Plot.ColorSet=clRed \\ Установить цвет графика

Plot.PenWidthSet=3 \\ Установить ширину линии графика

Chart.TitleYSet="Parallel factor" \\ Текстовая метка оси Y графика

Chart.TitleXSet="Model time" \\ Текстовая метка оси X графика

Chart.TitleSet="Grep parallel factor" \\ Название графика

16 Файловый шлюз (GatewayFile)

16.1 Функциональное назначение

Запись и чтение данных из файла

16.2 Контекст

Наименование поля контекста	Тип	Описание
FileName	String	Имя файла
OpenFlag	boolean	Флаг успешности открытия

		файла
SaveLoadFlag	boolean	Флаг успешности записи/чтения индексного массива
millicomand	int64	Милликоманда, прикрепляемая к извлеченному из файла индексному массиву
IndexMassLong	int64	Длина записываемого в файл индексного массива
IndexCounter	TPointIndex	Ссылка на индексный счётчик
IndexVector	TIndexVector	Ссылка на индексный массив

16.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
5	FileNameSet	Задать имя файла
9	MkSet	Задать милликоманду, прикрепляемую к считанному из файла индексному массиву
13	IndexVectRead	Считать капсулу и выдать милликоманду с капсулой на Шину
17	IndexVectLength	Установить длину записываемого в файл индексного массива
21	IndexVectWrite	Записать индексный массив в файл
25	IndexCounterPointerPop	Выдать ссылку на индексный счётчик
29	IndexCounterPop	Выдать значение индексного счётчика
33	IndexCounterPointerSet	Установить ссылку на индексный счётчик
37	IndexVectPointPop	Выдать ссылку на индексный файл

16.4 Примеры программирования

```
NewFU={Mnemo="FileGateway" FUType=FUGatewayFile}
NewFU={Mnemo="Automat" FUType=FUAutomat}
```

\\ Задать милликоманду для запуска миллипрограммы на ФУ Автомат

FileGateway.MkSet=Automat.SetRun

FileGateway.FileNameSet="Program.ind" \\ Задать имя файла

\\ Считать индексный файл (считанная из файла программа запустится на ФУ Automat)

FileGateway.IndexVectRead

17 ОА-инструментальное дерево (OATreeView)

17.1 Функциональное назначение

ФУ ОА-инструментальное дерево предназначено для отображения и просмотра всех функциональных устройств, их милликоманд. Также, с помощью ФУ TreeView можно просматривать ОА-дерево.

17.2 Контекст

Наименование поля контекста	Тип	Описание
OATree	PAtrData	Ссылка на отображаемое ОА-дерево
WindowCaption	String	Заголовок выводимого окна просмотра
ImageList	TImageList	Переменная для хранения набора изображений к компоненту TTreeView
TopLevelAtrs	Int64	Атрибут указателя на следующий уровень
NodeName	Int64	Атрибут имени узла
IcoAtr	Int64	Атрибут иконки
MkTree	Int64	Атрибут указателя на дерево списка милликоманд
Hint	Int64	Атрибут всплывающей подсказки
ClickProg	Int64	Атрибут указателя на миллипрограмму, выполняемую по нажатию мыши
DoublebClickProg	Int64	Атрибут указателя на миллипрограмму, выполняемую

		по двойному нажатию мыши
KeydownProg	Int64	Атрибут указателя на миллипрограмму, выполняемую по нажатию клавиатуры
MkTextClickBusAtr	Int64	Атрибут строки, выдаваемой при нажатии мыши
MkTextDoubleClickBusAtr	Int64	Атрибут строки, выдаваемой при двойном нажатии мыши
MkTextKeyDownBusAtr	Int64	Атрибут строки, выдаваемой при нажатии клавиатуры
TextSeparatorAtr	Int64	Атрибут строки разделителя, вставляемого между названиями узлов при выдаче
TreeParentLevelAtr	Int64	Атрибут числа строковых меток при выдаче на шину
TreeNodeComment	Int64	Атрибут комментария (не участвует при формировании строки, выбрасываемой на шину
Events	TOACapsul TreeViewEvents	Переменная для хранения экзмепляра класса TOACapsul TreeViewEvents
CurrentNode	TTreeNode	Указатель на текущий узел
BaseNode	TTreeNode	Номер текущего базового узла в массиве указателей на базовые узлы
LeafTextMk	Int64	Милликоманда, прикрепляемая ко всем листьям при инициализации инструментального дерева
LeafTextLavel	Int64	Число строковых меток при выдаче на шину, прикрепляемое ко всем листьям при инициализации

		инструментального дерева
LeafTextSeparator	Variant	Строка разделителя, вставляемая между названиями узлов при выдаче на шину, прикрепляемая ко всем листьям при инициализации инструментального дерева
SearchForm	TTreeSearch	Переменная экземпляра класса TTreeSearch (формы поиска)
SelectedItem	TTreeNode	Текущий (выбранный) узел инструментального дерева
LastFound	TTreeNode	Последний найденный узел инструментального дерева
StdSearchDirection	Boolean	Флаг. Если равен True, то направление поиска – «сверху-вниз», иначе – «снизу-вверх»
Found	Boolean	Флаг результата поиска по инструментальному дереву
SearchFromSelection	Boolean	Флаг. Если равен True, то поиск производится от текущего выделенного узла инструментального дерева, иначе – с коневого узла дерева

Описание класса TOACapsulTreeViewNode

Наименование поля контекста	Тип	Описание
Hint	String	Всплывающая подсказка узла
OnclickProg	Pointer	Нагрузка для милликоманды, выбрасываемой на шину при нажатии мыши на узле
LoadClick, LoadDoublrClick, LoadKeyDown	TLoad	Нагрузки, выбрасываемые по событиям Onclick, OnDoubleClick, OnKeyDown

		узла
MkClickBus, MkDoubleClickBus, MkKeyDownBus	Variant	Миликоманды, прикрепляемые к нагрузке, выдаются на шину. Соответствуют событиям узла Onclick, OnDoubleClick, OnKeyDown
BusContext	Pointer	Указатель на контекст шины
MkTextClickBus, MkTextDoubleClickBus, MkTextKeyDownBus	Variant	Текст для выдачи на консоль при событиях узлах OnClick, OnDoubleClick, OnDoubleClick
MkDataClickBus, MkDataDoubleClickBus, MkDataKeyDownBus	Variant	Данные для выдачи для другого ФУ при событиях OnClick, OnDoubleClick, OnDoubleClick
TextSeparator	Variant	Разделитель между названиями узлов, выдаваемыми на шину
TreeParentLevel	Variant	Число узлов для выдачи на шину
Reference	TPointIndex	
MkTree	PAtrData	Атрибут указателя на дерево списка милликоманд
Comment	Boolean	Флаг. Если равен True, то узел игнорируется при построении названий при выдаче на шину
TextSeparatorForLeaf	Variant	Строка разделителя, вставляемая между названиями узлов при выдаче на шину

17.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	Set	Назначить ссылку на отображаемую капсулу

3	SetImageList	Назначить ссылку на контейнер с картинками
7	SetTreeView	Установить ссылку на элемент TreeView
19	SetWindowCaption	Установить заголовок ОкнаВыводаОА-Дерева
23	SetIcoAtr	Установить атрибут иконки
27	SetMkList	Установить атрибут СписокМилликоманд
60	ClearField	Очистить поле TreeView
70	HintAtrSet	Установить атрибут всплывающей подсказки
80	SetBusPointer	Установить ссылку на ШИну
90	DelCurrentNode	Удалить Узел с потомками
95	CurrentNodeNumberSet	Установить текущий узел по номеру
100	CurrentNodeSet	Установить текущий узел по ссылке
130	CurrentNodeTextSet	Установить текст в текущий узел
135	CurrentNoteTextAdd	Добавить текст в текущий узел
200	NodesCountSet	Принять количество узлов для вывода контекста
210	BaseRegViewNodeSet	Установить базовый узел для просмотра контекста
215	CurrentNodeSet	Установить текущий узел для просмотра регистра через смещение (на входе номер просматриваемого регистра)
220	RegViewNodeRecDel	Удалить узел просмотра контекста (На входе номер узла)
230	NodeSearch	Поиск элемента дерева по имени
240	ContinueAtrAdd	Добавить атрибут в список Continue
250	MkTreeAtrSet	Установить атрибут дерева милликоманд
300	RootNodeTextSet	Установить название корневого узла
320	LeafTextLavelSet	Установить текстовый уровень для всех

		листьев дерева
330	LeafTextMkSet	Установить милликоманду для текста всех листьев дерева
340	LeafTextSeparatorSet	Установить текстовый разделитель всех листьев дерева
400	SearchOpen	Открыть поисковую панель
401	StartSearch	Начать поиск (вперед)
402	StartSearchBackwards	Начать поиск (назад)
430	Search	Выполнить милликоманду поиска по консоли (Search)
431	SearchMkSet	Установить милликоманду для внешнего поиска
500	ImagesSet	Установить стандартный набор картинок
501	ImageAdd	Добавить изображение в список ImageList
502	ImageLastIndexPop	Получить индекс последней добавленной картинки (через ссылку)
503	ImageLastIndexPopMk	Получить индекс последней добавленной картинки (через милликоманду)
505	ImageNodeLastIndexSet	Установить индекс последней добавленной картинки к нужному узлу (по индексу)

17.4 Примеры программирования

```
\\Создание ФУ ОА-ДЕРЕВО
NewFU={Mnemo="ОАДЕРЕВО" FUType=ФУОАДеревоПросмотра}
\\Создание ФУ «Целочисленное АЛУ»
NewFU={Mnemo="ALU" FUType=ФУЦелАлу}
\\Установка элементов дерева
```

```
ОАДЕРЕВО.Set=
        {
                \\Установка Mnemoники первого элемента дерева
                Mnemo="Сумматор"
\\Установка всплывающей подсказки первого элемента дерева
                Hint="ФУ Сумматор"
                \\Установка «мили-диапазона» первого элемента дерева
                MkBegRange =ALU.Сброс
                \\Установка миллипрограммы на событие «Нажатие мыши» на 1-й эл-т
дерева
                MouseClickProg=
                {
                        OutFU.VarOut="Привет!"
                }
\\Установка миллипрограммы на событие «Двойное нажатие мыши» на первый элемент
дерева
                DblClickProg=
                {
                        OutFU.VarOut="Привет2!"
                }
\\Установка миллипрограммы на событие «Нажатие клавиши» на первый элемент дерева
                KeyDownProg=
                {
                        OutFU.VarOut="Привет3!"
                }
                \\Установка вложенного элемента
                Obj=
                {
\\Установка Мнемоники вложенного элемента
                        Mnemo="Узел3"
\\Установка всплывающей подсказки вложенного элемента
                        Hint="Проба3"
\\Установка милликоманды на событие «Двойное нажатие мыши» вложенного элемента
                        MkTextDoubleClickBusAtr=OutFU.VarOut
\\Установка поля «TreeParentLevelAtr» для вложенного элемента
```

```
                TreeParentLevelAtr=2
                \\Установка нагрузки для события «Двойное нажатие мыши» для
вложенного элемента
                MkDataDoubleClickBusAtr="тест7654"
        \\Установка вложенного элемента
                Obj=
                {
\\Установка Мнемоники вложенного элемента
                        Mnemo="Узел4"
                }
                \\Установка флага «TreeNodeComment» для вложенного элемента
                TreeNodeComment
        }
\\Установка вложенного элемента
        Obj=
        {
                Mnemo="Узел2"
        }
\\Установка вложенного элемента
        Obj=
        {
                Mnemo="Узел4"
        }
    }
```

18 Генератор строк (SringsSource)

18.1 Функциональное назначение

Чтение строк из различных источников (файл, TMemo, TRichEdit) и выдача строк, снабженных соответствующим атрибутом для ФУ-приемника лексем.

18.2 Контекст

Наименование поля контекста	Тип	Описание
Strings	TStrings	Ссылка на строки для анализа

FromIndex, ToIndex	Variant	Номера начальной и конечной строк (если равны 0, то генерируется весь файл)
Atr	Variant	Атрибут генерируемой строки (если меньше 0, то в нагрузку генерируемой милликоманды помещается только строка)
Mk,Mk2,MkLog	Variant	Миликоманда, прикрепляемая к генерируемой строке
FinProg	Pointer	Указатель на программу, запускаемую по окончании генерации строк
work	Variant	Флаг указателя рабочего состояния ФУ
StringCounter	Variant	Номер текущей генерируемой строки
StringsOutMemo	Tmemo	Ссылка на Memo для вывода текущей строки
Col	Variant	
reStartProg, StopProg	Pointer	Указатели на подпрограмму, запускаемую перед генерацией строк и по выполнении милликоманды Стоп

18.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	FileSet	Установить ссылку на текстовый файл
5	MemoSet	Установить указатель на компонент TMemo
80	MkSet	Установить первую милликоманду для генерируемой строки
84	Mk2Set	Установить вторую милликоманду для

		генерируемой строки
88	MkLogSet	Установить служебную милликоманду для генерируемой строки
100	StartLineSet	Номер первой генерируемой строки
105	FinLineSet	Номер последней генерируемой строки
115	Start	Начать генерацию строк
120	Stop	Закончить генерацию строк
160	LineNumPop	Выдать номер текущей генерируемой строки
150	FinProgSet	Установить ссылку на подпрограмму, выполняему при успешном завершении генерации строк
200	PreStartProgSet	Установить ссылку на подпрограмму, выполняему перед генерацией символов
205	StopProgSet	Установить ссылку на подпрограмму, выполняему по приходе милликоманды Stop

18.4 Примеры программирования

\\ Установить милликоманду, прикрепляемую к генерируемой строке

StrSource.MkSet=Lexica.Lexing

\\ Установить милликоманду для вывода строки на служебную консоль

StrSource.MkLogSet=LogCon.VarOut

StrSource.FileSet=”Prog.oap” \\ Установить имя файла

StrSource.Start \\ Начать генерацию строк

19 Лексика (LexicaFU)

19.1 Функциональное назначение

Разделение строки символов на лексемы и передача лексем, оформленных в виде милликоманд ФУ-приемнику для дальнейшего синтаксического разбора.

На Рисунок 1 представлена синтаксическая диаграмма работы ФУ лексического разбора

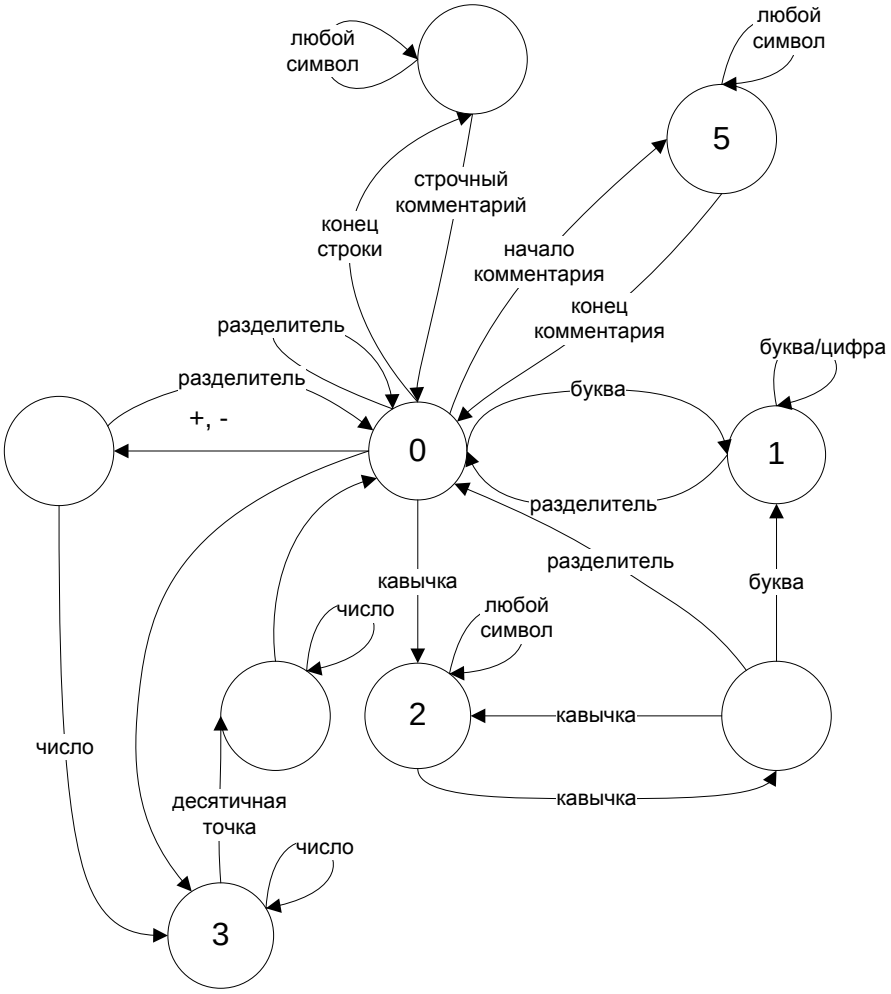


Рисунок 1 - Синтаксическая диаграмма работы ВФУ лексика

19.2 Контекст

Наименование поля контекста	Тип	Описание
Stop	Variant	Флаг останова генерации лексем
LexReceiver	TContextAndMkStask	Контекст и Мк для приемника лексем
MnemoReceiver	TContextAndMk	Контекст и Мк для ФУ разбора мнемоник
LastLexema	TPointIndex	Предыдущая и Последняя синтезированная лексема
LexemsListHead, LexemsListTail	PPointersList	Указатели на голову и хвост списка лексем

LexemsListCounter	integer	Счетчик сохраненных лексем
MessageMk	int64	Милликоманда для вывода сообщения
StrSourseStopMk	int64	Милликоманда останова генератора строк
StringBuffer	string	Служебный буфер
TrueStr	array[0..5] of string	Строки, обозначающие true
FalseStr	array[0..5] of string	Строки, обозначающие false
iCh	int64	Хранит позицию текущего распознаваемого символа

19.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
5	ReceiverSet	Установить указатель на контекст приемника лексем
10	ReceiverMkSet	Установить милликоманду для приемника лексем
15	MnemoReceiverSet	Установить указатель на контекст приемника мнемоник
20	MnemoReceiverMkSet	Установить милликоманду для приемника мнемоник
30	MessageMkSet	Установить сообщение, выдаваемое при обнаружении лексической ошибки
35	StopStrSourceMkSet	Установить милликоманду, выдаваемую при обнаружении лексической ошибки
240	LexemaToReceiver	Лексему, переданную в нагрузке милликоманды, потребителю лексем
245	LastLexemaToReceiver	Выдать последнюю лексему приемнику лексем (лексема выдается на приемник лексем

		даже, если это мнемоника)
248	LastLexemaPop	Выдать последнюю лексему
249	LastLexemaPopMk	Выдать милликоманду с последней лексемой
270	PrevLexemaPop	Выдать предыдущую лексему (если предыдущей лексемы не было, то выдается ИП с ярлыком разделителя и пустой строкой
275	PrevLexemaPopMk	Выдать милликоманду с предыдущей лексемой (если предыдущей лексемы не было, то выдается ИП с ярлыком разделителя и пустой строкой)
250	ReceiverMkPointerPop	Выдать указатель на милликоманду, прикрепляемую к лексеме
254	ReceiverMkPointerPopMk	Выдать на Шину милликоманду с указателем на милликоманду, прикрепляемоу к лексеме
255	ReceiverPointerPop	Выдать указатель на контекст приемника лексем
259	ReceiverPointerPopMk	Выдать на Шину милликоманду с указателем на контекст приемника лексем
225	ReceiverToStackMkSet	Положить милликоманду, прикрепляемую к лексеме, в стек и установить милликоманду к лексеме
230	ReceiverFromStackLexemaPop	Извлечь милликоманду премника и выдать лексекму
235	ReceiverFromStackLastLexemaPop	Извлечь милликоманду премника и выдать последнюю лексекму
50	Lexing	Разбор строки
190	NLexemsSet	Установить количество

		сохраняемых в контексте лексем (сохраненные лексемы доступны для анализа другим ФУ)
300	StopGenMkSet	Установить милликоманду, выполняемую при обраружении лексической ошибки
305	Stop	
276	CharCountPop	Выдать номер текущего распознаваемого символа
279	CharCountPopMk	Выдать милликоманду с номером текущего распознаваемого символа
280	CharMarkPop	Выдать маркер текущего распознаваемого символа
284	CharMarkPopMk	Выдать милликоманду с маркером текущего распознаваемого символа
320	IniProgSet	Установить программу начальной инициализации ФУ лексического разбора
330	Ini	Перейти в стартовое состояние (запускается программа инициализации и сбрасывается внутренний флаг комментария)

19.4 Примеры программирования

\\ Установить милликоманду для приемника лексем

Lexica.ReceiverMkSet=MkMnemoWait.FindIc

\\ Установить милликоманду для приемника мнемоник

Lexica.MnemoReceiverMkSet=MkMnemoWait.FindIc

\\ Лексический разбор одной строкм

Lexica.Lexing="ConstList.SearchMkSet=ProgPanel.SearchOpen"

19.5 Примечания

В том случае, если для разбора (милликоманда Lexing) приходит пустая строка, то для ФУ лексического разбора передает для приемника лексем ИП вида {Separator=""}, где

Separator – атрибут символа-разделителя.

20 Анализатор ОА-дерева (OATreeAnalyzer)

20.1 Функциональное назначение

Анализ ОА-дерева абстракций: подсчет вершин графа, содержащих определенный атрибут, составление матрицы по результатам анализа ОА-дерева.

20.2 Контекст

Наименование поля контекста	Тип	Описание
VerticalOrientation	Variant	Ориентация матирицы результатов (вертикальная или горизонтальная)
Atrs	array of TAtrsArray	Массив анализируемых в ОА-дереве атрибутов
FindArray	PMatrix	Матрица с результатами поиска в ОА-дереве
Tree	PAtrData	Ссылка на анализируемое ОА-дерево
OperationFlags	Byte	Флаги операций
Row	Integer	
NodesCount	Variant	Счетчик вершин анализируемого ОА-дерева
LeavesCount	Variant	Счетчик уровней (ярусов) анализируемого ОА-дерева
SortAsc	Boolean	Флаг сортировки массива с результатами поиска по ОА-дереву
SortIndex	Variant	Индекс типа сортировки массива с результатами поиска по ОА-дереву
NTableColomns	Variant	Количество столбцов результирующей таблице

20.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс
1	Set	Установить ссылку на анализируемое ОА-дерево
5	AtrSet	Установить атрибут для поиска в ОА-дереве
8	AtrAdd	Добавить атрибут для поиска в ОА-дереве
10	AtrClear	Очистить список атрибутов для поиска в ОА-дереве
15	VerticalOrientation	Установить ориентацию матрицы с результатами анализа ОА-дерева (true - вертикальная false – горизонтальная ориентация)
20	MatrOut	Выдать ссылку на матрицу с результатами анализа ОА-дерева
23	MatrOutMk	Выдать милликоманду со ссылкой на матрицу с результатами анализа ОА-дерева
25	NodesCount	Выдать количество вершина в анализируемом ОА-дереве
30	LeavesCount	Выдать количество уровней (ярусов) в анализируемом ОА-дереве
35	SortTypeSet	Установить тип сортировки результирующей матрицы (True - прямая, False - обратная)
40	SortIndexSet	Установить индекс строки/столбца, по которому производится сортировка
55	ColomnsSet	Установить количество столбцов в таблице (значение остальных атрибутов в таблицу не попадают)

20.4 Примеры программирования

ToolsPanelMaker.AtrClear \\ Очистить список атрибутов для анализа

ToolsPanelMaker.AtrSet=Mnemo \\ Установить первый атрибут для анализа

ToolsPanelMaker.AtrSet=Atr \\ Установить втрой атрибут для анализа

MnemoTable.PopMk=ToolsPanelMaker.Set \\ Установить ссылку на ОА-дерево

\\ Выдать ссылку на матрицу с результатами анализа ОА-дерева

ToolsPanelMaker.MatrOutMk=AtrList.ConstMatrOut

21 Многооконная текстовая консоль (OAConsole)

21.1 Функциональное назначение

Ввод и вывод текстовой информации на экран. Запуск программы из консоли. Поиск информации в тексте. Запись и чтение текста из/в файл. Поиск и замена в тексте.

21.2 Контекст

Наименование поля контекста	Тип	Описание
FUListSearchMK	Int64	Милликоманда поиска по FUList
FUListGetResultsMK	Int64	Милликоманда получения результата из FUList
TreeSearchMK	Variant	Милликоманда для поиска по дереву
StringsMK	Int64	Милликоманда, прикрепляемая к выдаваемым строкам
CurTab	Variant	Номер текущей вкладки
TabsCount	Variant	Количество вкладок
WholeWord	Boolean	Поиск слова целиком
PageControl	TPageControl	Регистрозависимый поиск
Edits	array of TRichEdit	Массив ссылок на объекты TRichEdit, расположенных на вкладках консоли
FileNames	array of string	Массив имен файлов, открытых на вкладках консоли
TabStatuses	array of Integer	массив состояний вкладок (0=корневая вкладка,

		1=тестовая вкладка, 2=Выводная вкладка)
Window	TForm	Собственное окно консоли
Events	TConsoleEvents	Объект, задающий реакции консоли на события
FindForm	TForm2	Форма (диалоговое окно) для осуществления поиска текста в консоли
StdSearchDirection	Boolean	Направление поиска врагмента текст (вперед/назад)
CurRichEdit	Integer	Текущий (активный) компонент RichEdit
FindFromCursor	Boolean	Флаг поиска от курсора
FindInSelection	Boolean	Флаг поиска в выделенном фрагменте текста
FUMark	Variant	Текстовая метка описания ФУ в программе (по умолчанию **\ - символ комментария)
ReplaceFlag	Boolean	Флаг замены при поиске
StartProgramMk	Int64	Милликоманда запуска компиляции текста программы, находящегося в текущей вкладке консоли
StartProgramMp	PAtrData	Указатель на миллипрограмму, служащую для компиляции текста программы, находящегося в текущей вкладке консоли
RestartProgramMk	Int64	Милликоманда сброса рестарта ОА-программы
RestartProgramMp		Указатель на миллипрограмму рестарта ОА-программы
ResetProgramMk	Int64	Милликоманда сброса ОА-компилятора

ResetProgramMp	PAtrData	Указатель на миллипрограмму сброса ОА-компилятора
OpenDialog	TOpendialog	Диалоговое окно чтения из файла текста в текущую текстовую вкладку
SaveDialog	TSavedialog	Диалоговое окно записи текста с текущей вкладки панели в файл
LastFileName	string	Имя последнего записываемого или открываемого файла
maxNonameIndex	Integer	Максимальный индекс, который дается имени вкладки новой открытой вкладки (“noname1”)
MatrSeparator	Variant	Символ-разделитель для вывода матрицы
RootTabProg, TestTabProg, OutTabProg	Pointer	Указатели на миллипрограммы, запускаемый при выборе корневой, тестовой и выводной вкладки консоли

21.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
130	LnOut	Вывести переменную или константу в отдельную строку
160	Out	Вывести переменную или константу в последнюю строку
165	OutLn	Вывести переменную или константу и перевести строку
131	ListOut	Вывести список капсул
100	TextInsert	Вставить фрагмент теста
0	Reset	Сброс ФУ
3	NewTab	Открыть новую вкладку консоли
7	SearchOpen	Открить диалог поиска

135	ClearPage	Очистить текстовую вкладку (на входе номер вкладки, если на входе nil, то очищается текущая вкладка
200	Focus	Дать фокус окну консоли
205	CaptionSet	Установить заголовок окна консоли
210	VisibleSet	Установить видимость окна консоли
215	StateSet	Установить статус окна консоли
60	StartMkSet	Установить милликоманду для запуска миллипрограммы
61	StartProgSet	Установить миллипрограмму для запуска миллипрограммы
62	RestartMkSet	Установить милликоманду для перезапуска миллипрограммы
63	RestartProgSet	Установить миллипрограмму для перезапуска миллипрограммы
64	ResetMkSet	Установить милликоманду для сброса миллипрограммы
65	ResetProgSet	Установить миллипрограмму для сброса миллипрограммы
66	Start	Компиляция и запуск миллипрограммы, текст которой находится в текущей вкладке консоли
67	Restart	Рестарт миллипрограммы, текст которой находится в текущей вкладке консоли
250	RootTabProgSet	Установить миллипрограмму, запускаемую при выборе корневой вкладки консоли
251	TestTabProgSet	Установить миллипрограмму, запускаемую при выборе тестовой вкладки консоли

252	OutTabProgSet	Установить миллипрограмму, запускаемую при выборе выводной вкладки консоли
1	ParentSet	Установить родительский графический элемент
2	Close	Закрыть текущую текстовую кладку
10	TreeSearchMkSet	Установить милликоманду для выдачи поиска фрагмента текста в ОА-инструментальном дереве
40	FUMarkSet	Установить текст, используемый в качестве метки описания ФУ в текста миллипрограммы
110	StringsPop	Выдать ссылку на текст текущей вкладки (TStrings)
115	StringsPopMk	Выдать милликоманду со ссылкой на текст текущей вкладки (TStrings)

21.4 Примеры программирования

ALU.PopMk=Console.Out \\ Вывод величины, хранимой в аккумулятора АЛУ на консоль

List.PopMk= Console.ListOut \\ Вывод ОА-списка на консоль

Console.ClearPage \\ Очитить текущую текстовую вкладку

Console.ClearPage=2 \\ Очистить вторую текстовую вкладку

Console.NewTab \\ Открыть новую вкладку

\\ Открыть окно поиска и поместить в поисковую строку "Console"

Console.SearchOpen="Console"

22 Графический элемент вывод текстового списка (ListBox)

22.1Функциональное назначение

Вывод таблицы текстовых строк на экран.

22.2 Контекст

Наименование поля контекста	Тип	Описание
ListBox	TListBox	Ссылка на компонент TListBox, который

		используется для вывода информации
Form	TForm	Ссылка на собственное графическое окно для компонента TListBox
OnClickProg	Pointer	Указатель на миллипрограмму, запускаемую по нажатию кнопки
OnDblClickProg	Pointer	Указатель на миллипрограмму, запускаемую при двойном нажатии кнопки мыши
OnKeyDownProg	Pointer	Указатель на миллипрограмму, запускаемую при нажатии кнопки клавиатуры
IndexForOut	Variant	Индекс строки (или столбца) матрицы на экран
Orientation	Variant	Ориентация (строка или столбец) вывода матрицы на экран
TextMk	Int64	Милликоманда, прикрепляемая к выбранной текстовой строке при двойном нажатии кнопки
FindMk	Int64	Милликоманда, прикрепляемая к текстовой строке при нажатии горячей клавиши Alt-F

22.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	Create	Создать компонент ListBox (на входе может быть ссылка на родительсткий компонент)
5	Set	Установить ссылку на компонент ListBox
10	ParentSet	Установить родительский компоент для ListBox
15	Del	Уничтожить компонент ListBox
45	Clear	Очистить поле компонента ListBox
50	VisibleSet	Установить свойство видимость
55	AlignSet	Установить свойство Align
60	ColomnsSet	Установить количество столбцов в

		таблице
65	EnabledSet	Установить свойство Enabled
70	MultiSelectSet	Установить свойство MultiSelect
75	ShowHintSet	Установить свойство ShowHint
80	StyleSet	Установить свойство Style
150	ConstMatrOut	Отразить массив в списке
155	IndexSet	Установить индекс для вывода массива
160	OrientationSet	Установить ориентацию для вывода
200	TextMkSet	Установить милликоманду, прикрепляемую к тексту из таблицы по нажанию кнопки

22.4 Примеры программирования

ListBox.ParentSet=nil \\ Создание собственного окна для компонента

ListBox.Clear \\ Очистка списка

ListBox.ConstMatrOut=VarList \\ Вывод списка из матрицы констант

\\ Установка милликоманды, для вывода мнемоники из списка на программную консоль

ListBox.TextMkSet=Console.TextInsert

23 Планировщик (Scheduler)

23.1 Функциональное назначение

Виртуальный планировщик- планирование вычислений в программной модели суперкомпьютерной системы.

23.2 Контекст

Наименование поля контекста	Тип	Описание
NCores	Int64	Количество ядер
CoresCount	Variant	Количество работающих ядер
ScheduleTime	Variant	
MkCount,CoreReguestCount	Variant	Счетчик Мк, стоящих в очереди к ФУ ; Счетчик запросов на ядро процесора

ContextLoadFactor, ContextSaveFactor, CPU_Factor	Double	Коэффициент времени на загрузку и выгрузку контекста ФУ; объем памяти контекста перемножается на коэффициет и получается время загрузки или выгрузки контекста; коэффицент времени в зависимости от количества тактов на выполнение программы
RamSize, RamSizeInWork, RamSizeInMkQueue	Int64	Объем оперативной памяти вычислительного узла; оперативная память, занятая под задачи; оперативная память, занятая под очередь милликоманд
CurrentTimeUk	PVariant	Указатель на переменную с текущим модельным временем
CapsuleCreator	TCapsuleCreator	Объект для создания капсул
EventserContext	Pointer	
EventserMk, EventserWaitMk	Int64	Милликоманда для Eventser-а добавления нового события; милликоманда для Eventser-а ожидания события
QueueHead, QueueTail	PPointersList	Указатели на голову и хвост списка ожидания ресурсов
CoreRequestCount	int64	Счетчик запросов на освобождение ядра
ContextChengeCount	int64	Счетчик событий смены контекста
CoreRequestHead, CoreRequestail,CoreRequestTemp	PObjQeueue	Список ожидания запросов на ядро
EventsCount	Int64	Счетчик событий

23.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
10	FUContextLoadFactorSet	Установить коэффициент времени загрузки образа ФУ
15	FUContextSaveFactorSet	Установить коэффициент времени выгрузки образа ФУ
20	RamSizeSett	Установить объем памяти вычислительного узла
25	TimePointerSet	Установить ссылку на переменную с текущим модельным временем
30	EventserContextSet	Установить контекст Eventser
35	EventserMkSet	Усновить милликоманду добавления события для Eventser
40	CoreCountPop	Выдать количество занятых ядер
44	CoreCountPopMk	Выдать милликоманду с количеством занятых ядер
45	MkCountPop	Выдать количество милликоманд, находящихся в очереди к исполнению на ФУ
49	MkCountPopMk	Выдать милликоманду с количеством милликоманд, находящихся в очереди к исполнению на ФУ
50	CoreRequestCountPop	Выдать количество запросов на ресурсы
54	CoreRequestCountPopMk	Выдать милликоманду с количеством запросов на ресурсы
55	RamPop	Выдать объем задайствованной в вычислительном процессе
59	RamPopMk	Выдать милликомаду с объемом задйствованной в вычислительном процессе памяти

64	EventsCountPopMk	Выдать милликоманду с числом зафиксированных событий
69	ContextChengeCountPopMk	Выдать милликоманду с числом зафиксированных событий смены контекста у исполнительного устройства
100	CoreRequest	Запрос ресурсов
105	CoreContinue	Запрос ФУ на продолжение работы на занятом ядре
110	CoreFree	Освободить ядро
115	MkQueue	Поставить/удалить милликоманду в/из очередь/и к ФУ
995	ProgExec	Выполнить миллипрограмму
995	ContextPop	Выдать указатель на контекст ФУ
999	ContextPopMk	Выдать милликоманду с указателем на контекст ФУ
920	FUContunueWork	Продолжить работу ФУ (в режиме ручного управления
918	SchedulerContextSet	Установить ссылку на контекст ФУ-планировщика
916	ManualModeSet	Установить режим ручного управления ФУ
914	MkQueueCountPop	Выдать количество милликоманд в очереди на выполнение к ФУ
913	MkQueueCountPopMk	Выдать милликоманду с количеством милликоманд в очереди на выполнение к ФУ
912	MkQueueCapsPop	Выдать указатель на капсулу с милликомандами с очереди на выполнение к ФУ
911	MkQueueCapsPopMk	Выдать милликоманду с указателем на капсулу с милликомандами с очереди на выполнение к ФУ

910	ModelingReset	Сбросить настройки моделирования у ФУ
924	PrefixProgSet	Установить ссылку на подпрограмму, запускаемую до обработки пришедшей милликоманды
922	PostfixProgSet	Установить ссылку на подпрограмму, запускаемую после обработки пришедшей милликоманды
926	BusSet	Установить контекст Шины
990	ProgExec	Выполнить миллипрограмму

23.4 Примеры программирования

\\ Задать ссылку на балансир (планировщик) для шины

\\ (шина при создании нового ФУ, будет автоматически прописывать

\\ в его контекст ссылку на данный планировщик

Scheduler.ContextPopMk=MainBus.SchedulerContextSet

Scheduler.NCoresSet=30 \\ Установить число ИУ в ядре

24 Исполнитель программ (ProgExec)

24.1 Функциональное назначение

Запуск миллипрограммы.

24.2 Контекст

Наименование поля контекста	Тип	Описание
TAutomatIteration	record	Контекст для каждой итеррации
PC, PC_Old	PAtrData	Программный счетчик
InputObject	TPointIndex	Указатель на входной объект
FWork	boolean	Флаг рабочего режима автомата
IfGoToPoint	PAtrData	Адрес условного перехода
SubStack	PAtrData	Стек адресов возврата из подпрограммы, стек объектов

InObjAdress	Pointer	Адрес, куда следует записывать входной объект
InObjPlase	PAtrData	Ячейка, куда следует помещать голову входного объекта
LocalVars	array of TLoad	
Flag	TLoad	флаг перехода

24.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	
1	Set	Установить указатель на начало программы
5	EnabledSet	Установить активный/пассивный режим
10	Exec	Выполнить программу
15	ExecTrue	Выполнить программу, если в нагрузке милликоманды true
20	ExecFalse	Выполнить программу, если в нагрузке милликоманды false

24.4 Примеры программирования

NewFU={Mnemo="ProgExec" FUType=FUProgExec}

\\ Задать указатель на выполняемую программу

ProgExec.Set={Console.OutLn="Hello World !!!"} ProgExec.Exec \\ Запуск программы

ProgExec.ExecTrue=True \\ Запуск программы, если в нагрузке милликоманды True

25 Главный менеджер сетки (IsingTopManager)

25.1 Функциональное назначение

Координация работы всего теста Изинга.

25.2 Контекст

Наименование поля контекста	Тип	Описание
StartEnergy, Temperatyre	Variant	Температура, стартовая энергия
N_Neighbors	Variant	Число соседей
IterationNumber	Integer	Номер итерации
Length, Height, Weigth	Variant	Длина, высота и ширина решётки
array of Variant	array of Variant	Энергии по итерациям
EnergyCount	array of Variant	Счётчик для массива энергий оп итерациям

25.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	NeiborsQuantitySet	Установить число соседей у узла вычислительной сетки (для 2-мерной сетки – 4, для 3-мерной – 6)
20	EnergySet	Установить текущую энергию
25	TemperatureSet	Установить текущую температуру
30	WidhtSet	Задать длину сегмента
35	HiegthSet	Задать высоту сегмента
40	LongSet	Задать ширину сегмента
45	EnergyIntgrate	суммирование общей энергии по блоку кристалла
100	NetCreate	Создание решётки
101	ToTopGateway	Приём волны по верхнему краю для передачи на шлюз
102	ToLeftGataway	Приём волны по левому краю для передачи на шлюз
105	TopMkTypeSet	Установка типа передаваемой миликоманды для верхнего края

106	LeftMkTypeSet	Установка типа передаваемой миликоманды для левого края
110	TopIsingFUTogataway	Присоединение изингов верхней границы к шлюзу
111	BottonIsingFUTogataway	Присоединение изингов нижней границы к шлюзу
112	LeftIsingFUTogataway	Присоединение изингов левой границы к шлюзу
113	RightIsingFUTogataway	Присоединение изингов правой границы к шлюзу
120	BorderLengthSet	Задать количество граничных изингов присоединённых к шлюзу
121	TopIsingToGateway	Присоединение изингов верхней границы к шлюзу
122	BottonIsingFUTogataway	Присоединение изингов нижней границы к шлюзу
123	LeftIsingFUTogataway	Присоединение изингов левой границы к шлюзу
124	RightIsingFUTogataway	Присоединение изингов правой границы к шлюзу
200	EnergyIntegrate	Подсчёт общей энергии блока

26 Менеджер сетки (IsingManager)

26.1 Функциональное назначение

Координация работы сегмента сетки ФУ Изинг на одном вычислительном узле

26.2 Контекст

Наименование поля контекста	Тип	Описание
Energy	array of Variant	Массив энергий по итерациям
EnergyCount	array of Variant	Указатель на последний элемент массива энергий
Temperature	variant	Температура
N_Neighbors	Variant	Число соседей для одного

		IsingFU
PIsingFirst, PIsingLast, PIsingCur, PIsingColBegin	Pointer	Указатель на первый IsingFU при создании сегмента решётки; указатель на последний IsingFU при создании сегмента решётки; указатель на текущий IsingFU при создании сегмента решётки; указатель на первый в столбце IsingFU
PLeftBorder, PRightBorder, PTopBorder, PBottomBorder	Pointer	Указатель на текущий IsingFU при присоединении нескольких шлюзов к левой границе; указатель на текущий IsingFU при присоединении нескольких шлюзов к правой границе; указатель на текущий IsingFU при присоединении нескольких шлюзов к верхней границе; указатель на текущий IsingFU при присоединении нескольких шлюзов к нижней границе
LeftBorderCount, RightBorderCount, TopBorderCount, BottomBorderCount	Variant	Задание диапазона номеров ФУ Изинг для подсоединения к шлюзу
IterationNumber	Integer	Номер итерации
Length, Height, Weigth	Variant	Длина, высота, ширина
BorderLength	Variant	Количество IsingFU, присоединяемых к шлюзу
PRightSpecialPoint, PBottomSpecialPoint	Pointer	Указатели на специальные крайние точки
CountIFU	Variant	Количество IsingFU
TopMkType, LeftMkType	Variant	

Manager	Pointer	Указатель на IsingTopManager

26.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	NeiborsQuantitySet	Установить число соседей у узла вычислительной сетки (для 2-мерной сетки – 4, для 3-мерной – 6)
20	EnergySet	Установить текущую энергию
25	TemperatureSet	Установить текущую температуру
30	WidhtSet	Задать длину сегмента
35	HiegthSet	Задать высоту сегмента
40	LongSet	Задать ширину сегмента
45	EnergyIntgrate	суммирование общей энергии по блоку кристалла
100	NetCreate	Создание решётки
101	ToTopGateway	Приём волны по верхнему краю для передачи на шлюз
102	ToLeftGataway	Приём волны по левому краю для передачи на шлюз
105	TopMkTypeSet	Установка типа передаваемой миликоманды для верхнего края
106	LeftMkTypeSet	Установка типа передаваемой миликоманды для левого края
110	TopIsingFUTogataway	Присоединение изингов верхней границы к шлюзу
111	BottonIsingFUTogataway	Присоединение изингов нижней границы к шлюзу
112	LeftIsingFUTogataway	Присоединение изингов левой границы к шлюзу
113	RightIsingFUTogataway	Присоединение изингов правой границы к шлюзу

120	BorderLengthSet	Задать количество граничных изингов присоединённых к шлюзу
121	TopIsingToGateway	Присоединение изингов верхней границы к шлюзу
122	BottonIsingFUTogataway	Присоединение изингов нижней границы к шлюзу
123	LeftIsingFUTogataway	Присоединение изингов левой границы к шлюзу
124	RightIsingFUTogataway	Присоединение изингов правой границы к шлюзу
200	EnergyIntegrate	Подсчёт общей энергии блока

26.4 Примеры программирования

\\ Создание 2-мерной сетки для моделирования размером 10x10

Задать ссылку на топ-менеджер ФУ для расчета по модели Изинга

Maneger.NeiborsQuantitySet=4

Maneger. WidhtSet=10

Maneger. HiegthSet=10

Maneger. NetCreate

27 ФУ для вычислений по модели Изинга (IsingFU)

27.1 Функциональное назначение

Моделирование двумерных и трехмерных спиновых эффектов по модели Изинга.

27.2 Контекст

Наименование поля контекста	Тип	Описание
Energy, Temperature	variant	энергия и температура
Neighborspins	array [0..10] of variant	спины у соседей
Neighbors	array [0..5] of Pointer	указатели на соседние IsingFU
N_Neighbors	Variant	число соседей
IsingProg	Pointer	
Spin	Variant	спин
Manager	Pointer	указатель на менеджера
Counter	Variant	счётчик

IterationNumber	Variant	номер итерации

27.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	NeiborsQuantitySet	Установить число соседей у узла вычислительной сетки (для 2-мерной сетки – 4, для 3-мерной – 6)
20	EnergySet	Установить текущую энергию
25	TemperatureSet	Установить текущую температуру
30	SpinPop	Выдать текущий спин
35	SpinPopMk	Выдать милликоманду с текущим спином
40	SpinProgSet	Усновить указатель на миллипрограмму, выполняемую при вычислении спина
45	EnergyPop	Выдать энергию
50	EnergyPopMk	Выдать милликоманду с энергией
90..99	Neibor1Set…Neibor9Set	Запись в массив указателей на контексты соседей
100	WaveStart	Начало вычислительной волны в кристалле
101	WaveTop	Волна по верхнему краю
102	WaveLeft	Волна по левому краю
103	WaveInside	Волна внутри грани
104	WaveLeftStart	Начало волны в блоках по левому краю кристалла
105	WaveSide	Волна внутри грани между блоками
106	SpinTopSet	Получение спина от прохода волны верхним соседом
107	SpinLeftSet	Получение спина от прохода волны левым соседом
110…119	Spin1Set… Spin9Set	Получение спина от соседей

28 Регулярные выражения (Regexp)

28.1 Функциональное назначение

Выделение подстрок из текста по регулярным выражениям.

28.2 Контекст

Наименование поля контекста	Тип	Описание
reg	TRegExpr	Объект, выполяющий поиск по регулярному выражению
expr	string	Поисковое регулярное выражение (шаблон)
InputString	string	Входная строка
ResultMk	Int64	Милликоманда для отправки результата (первого найденного вхождения)
ResultMatrixMk	Int64	Милликоманда для отправки результатат в формате массива/матрицы
ResultProg	PAtrData	Миллипрограмма для отправки результата
m	PMatrix	Массив/матрица результатов
IndexOffset	Integer	Смещение для выброса МК на коллектор
ResultFUContext	Pointer	Указатель на контекст устройства-получателя, по умолчанию - BusContext

28.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
20	PatternSet	Установка поискового шаблона (регулярное выражение)
21	StringSet	Установка входной строки для поиска
22	PatternSetStart	Установка поискового шаблона (регулярное выражение), затем запуск поиска
23	StringSetStart	Установка входной строки для поиска, затем запуск поиска
25	ResultMkSet	Установка милликоманды для отправки результата (первого найденного вхождения)

26	ResultMatrMkSet	Установка милликоманды для отправки результата в формате массива/матрицы (все внайденные вхождения)
27	ResultMatrPop	Выполнить милликоманду отправки результата в формате массива/матрицы
28	ResultMatrPopMk	Установить и выполнить милликоманду отправки результата в формате массива/матрицы
29	ResultProgSet	Установить миллипрограмму для отправки результата
30	Start	Запуск поиска
41	IndexOffsetSet	Установка стартового значения индекса при индексации входных строк. Используется в качестве милликоманды при приеме входной строки. (по умолчанию - 1000)
42	ResultFUContextSet	Установка контекста ФУ, которое будет принимать результаты поиска

28.4 Примеры программирования

\\Создание ФУ

NewFU={Mnemo="Reg" FUType=FURegexp}

…

\\Установка регулярного выражения поиска

Reg.PatternSet="[A-Za-z]+"

\\Установка контекста ФУ Regexp устройству RegexpManager в качестве эталонного устройства

Reg.ContextPopMk=RegManager.RegexpFUSet

29 Менеджер разбора по регулярным выражениям (RegexpManager)

29.1 Функциональное назначение

1. Прием и распределение строк между несколькими Regexp;
2. Клонирование и уничтожение ВФУ Regexp.

29.2 Контекст

Наименование поля контекста	Тип	Описание
Contexts	array of PRegexpContext	Массив указателей на контексты ФУ Regexp
Count	int64	Длина массива указателей
Index	Integer	Индекс текущего элемента массива указателей
Template	Pointer	Указатель на эталон ФУ Regexp
IndexOffset	Integer	Смещение для выброса МК на коллектор
CollectorContext	Pointer	Контекст ФУ Коллектора
StringIndex	Int64	Индекс текущей строки (заявки)
StringGenTime	Variant	Задержка генерации строки

29.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
2	RegexpFUSet	Установить ссылку на эталонный ФУ Regexp
3	Clone	Слонировать эталонных ФУ Regexps
9	StringPut	Принять строку для распределения на одно из ФУ Regexp

29.4 Примеры программирования

\\Создание ФУ

NewFU={Mnemo="RegManager" FUType=FURegexpManager}

…

\\Установка контекста ФУ Regexp устройству RegexpManager в качестве эталонного

устройства

Reg.ContextPopMk=RegManager.RegexpFUSet

\\Создание («клонирование») десяти устройств Regexp (Worker)

RegManager.Clone=10

\\Установка милликоманды вывода строки – приём строки на ФУ RegexpManager

StringSource.MkSet=RegManager.StringPut

30 Сборщик результатов разбора от ВФУ Regexp (RegexpCollector)

30.1 Функциональное назначение

Сбор поступающих с нескольких RegexpFU результатов анализа строк по регулярным выражением, восстановление первоначального порядка следования результатов и отправка результатов ФУ-адресату.

30.2 Контекст

Наименование поля контекста	Тип	Описание
Results	array of Tinfo	Массив результатов
ReceiverFUContext	Pointer	Контекст устройства, принимающего результаты
ReceiverMk	int64	Милликоманда, к которой прикрепляются результаты
LastSentResultIndex	Int64	Индекс последнего высланного результата
LastElement	Int64	Индекс последнего полученного элемента

30.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	ReceiverFUContextSet	Установить контекст ФУ, которому будут пересылаться результаты
2	ReceiverMkSet	Установить милликоманду, к которой будут прикрепляться

		результаты

30.4 Примеры программирования

\\Создание ФУ

NewFU={Mnemo="RegCollector" FUType=FURegexpCollector}

…

\\Установка контекста ФУ RegexpCollector устройству Regexp (Worker)

RegCollector.ContextPopMk=Reg.ResultFUContextSet

\\Установка милликоманды вывода результата на выводную консоль (ФУ Console)

RegCollector.ReceiverMkSet=Console.LnOut

31 Диаграмма (Chart)

31.1 Функциональное назначение

Построение диаграмм и графиков (ВФУ работает в связке с ВФУ Plot)

31.2 Контекст

Наименование поля контекста	Тип	Описание
Color	TColor	Цвет поля диаграммы
Graph	TChartSeries	Компонент Delphi, служащий для отображения одной функции
PointsRange	int64	Количество выводимых на экран точек
NotInit	boolean	Флаг, обозначающий, что не поступало ни одной точки для вывода
X	double	Текущая координата, в которой выводится точка графика
M	array of array of Variant	Массив выводимых точек графика
StepX	double	Шаг значений по оси X
Xtemp	double	Текущее значение X (для вывода по шагу)
LastIndex	integer	Индекс последней точки

31.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	SeriaCreat	Создать Серию (на входе номер вида серии)
2	SeriaPointerPop	Выдать ссылку на серию
4	SeriaPointerPopMk	Выдать милликоманду со ссылкой на серию
3	ParentSet	Задать ссылку на родителя
5	SeriaDel	Уничтожить График
6	NPointersSet	Задать максимальное количество выводимых точек
10	ColorSet	Задать Цвет
50	XSet	Задать координату X
64	YSet	Задать координату Y и вывести точку на график
70	StepSet	Задать шаг
74	XCurrentSet	Установить текущую координату X
78	YByStepSet	Вывести значение Y соотвественно шагу
80	PenWidthSet	Установить ширину линии графика
84	PenColorSet	Установить Цвет линии
0	Reset	Сброс ФУ
1	ParentSet	Установить родительский графический объект
4	VisibleSet	Установить свойство видимость
6	HeigthSet	Установить высоту
8	WigthSet	Установить ширину
10	TopSet	Установить верхнюю грань окна
12	LeftSet	Установить координату элемента

		по горизонтали
14	AlegnSet	Установить свойство Alegh
18	ChartPop	Выдать ссылку на Chart
22	ChartPopMk	Выдать милликоманду со ссылкой на Chart
30	SeriaIndexSet	Установить Индекс диаграммы
34	SeriaIndexSet	Установить Индекс Серии графика
40	ChartCreat	Создать диаграмму и выдать ссылку на неё
42	ChartSet	Задать ссылку на диаграмму
43	ChartDel	Уничтожить диаграмму
47	ChartPop	Выдать ссылку на диаграмму
49	ChartPopMk	Выдать милликоманду со ссылкой на диаграмму
54	SeriaLastIndexPop	Выдать индекс последней созданной серии
56	SeriesCountPop	Выдать количество созданных серий
58	SeriaDel	Удалить серию по индексу
60	SeriaColorSet	Установить Цвет серии
65	TitleSet	Установить заголовок
70	TitleYSet	Установить метку оси Y
75	TitleXSet	Установить метку оси X
100	Focus	Дать фокус окну с графиком
105	Caption	Установить заголовок индивидуального окна для графика

32.3 Примеры программирования

NewFU={Mnemo="Chart" FUType=FUChart}

NewFU={Mnemo="Plot" FUType=FUPlot}

Plot.SeriaCreat \\ Создание серии графиков

Chart.ParentSet \\ Создание отдельного окна для вывода графика

Chart.Caption="Parallel factor" \\ Установка заголовка окна диаграммы

Chart.Focus \\ Дать компоненту фокус ввода

Plot.SeriaCreat \\ Создать серию графика

Chart.ChartPopMk=Plot.ParentSet \\ Установка родительского компонента для графика

Plot.ColorSet=clRed \\ Установить цвет графика

Plot.PenWidthSet=3 \\ Установить ширину линии графика

Chart.TitleYSet="Parallel factor" \\ Текстовая метка оси Y графика

Chart.TitleXSet="Model time" \\ Текстовая метка оси X графика

Chart.TitleSet="Grep parallel factor" \\ Название графика

32 Менеджер ВФУ Graph500 (Graph500Manager)

32.1 Функциональное назначение

1. Создание ФУ для анализа графа
2. Синтез графа
3. Запуск итераций теста Graph500

32.2 Контекст

Наименование поля контекста	Тип	Описание
Scale,N,M, NFU	Int64	Диапазон номеров вершин графа для каждого ФУ анализа графа
probability1, probability2,probability3, probability4	Variant	Вероятности генерации вершины в 1,2,3 или 4-й четверти матрицы смежности
Graph500FUVector	array of PGraph500FU	Матрица указателей на ФУ для обработки дуг графа
EdgeGeneratorsVector	array of PEdgeGenerator	Массив ссылок на клонированнные ФУ
PointIndexRange	Int64	Диапазон номеров вершин графа для каждого ФУ анализа графа
Collector	PGraph500Collector	Ссылка на сборщике результатов

NIterations	Int64	Число итераций теста

32.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	ScaleSet	Установить scale (корень от числа вершин)
5	EdgeSet	Установить число ребер в графе
10	EdgeFactorSet	Установить число ребер с помощью коэффицента от числа вершин графа
20	CollectorSet	Установить ссылку на сборщика результатов
25	FUFieldRangeSet	Установить размерность поля для анализа графа (число ФУ на одной грани вычислительного поля)
100	Run	Начать тест (генерацию FU для анализа графа) (на входе количество шагов моделирования - по уморчанию прогонятеся 64 итарации)

32.4 Примеры программирования

\\ Настройка ФУ вычислительного поля

GraphManajer.ContextPopMk=GraphCollector.GraphGenSet \\Задать ссылку на коллектор

GraphManajer.ScaleSet=10\\задать число вершин графа

GraphManajer.EdgeFactorSet=16 \\ задать число ребер

GraphManajer.FUFieldRangeSet=7 \\ задать число ФУ, анализирующих граф.

33 Сборщик результатов работы по тесту Graph500 (Graph500Collector)

33.1 Функциональное назначение

Сбор данных с ВФУ Graph500, обрабатывающих граф (номера вершин, входящих в

путь по графу, средняя длина пути в графе и т.д.).

33.2 Контекст

Наименование поля контекста	Тип	Описание
VertexList	TInt64List	Список найденных вершин для каждой итерации
count	Int64	Счетчик найденных вершин
EdgeGenerator	PEdgeGenerator	Ссылка на генератор вершин

33.3 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
0	Reset	Сброс ФУ
1	GraphGenSet	Устновить ссылку на генератор графа
50	PointSet	Принять номер вершины с итераций
35	CountPopMk	Выдать милликоманду со значением счетчика вершин
40	TraceLongPopMk	Выдать милликоманду со средней длиной пути в графе
20	IterationSet	Установить номер текущей итерации
25	TraceLongIterationPopMk	Выдать длину пути для итерации
30	TraceLongIterationPopMkList	Выдать длину пути для итерации

34 Общая часть ВФУ

У ФУ любого типа есть стандартный набор полей в контексте и набор милликоманд дополняется стандартными милликомандами.

34.1 Контекст

Наименование поля контекста	Тип	Описание
FUProgram	TFUProgram	Ссылка на программу реализации

		логики работы ФУ
FUKill	TFUkill	Ссылка на подпрограмму уничтожения ФУ
FUIniProg	TContextViewIni	Ссылка на ОА-программу инициализации ФУ
BusContext	Pointer	Указатель на шину
FUMillidiap	Variant	Миллидиапазон
ManualMode	Variant	Флаг "ручного" управления ФУ
MkStage	int64	фаза выполнения милликоманды (если в ручном режиме милликоманда выполняется за несколько тактов)
FUType	Variant	Идентификатор типа ФУ
FUName	Variant	Имя ФУ
SchedulerContext	Pointer	Контекст планировщика
ContextSize	Variant	Размер контекста
qmks,qtail	PAtrData	Указатели на голову и хвост очереди милликоманд при ручном режиме управления (используется при в режиме моделирования и отладки)
getResource	function(context: Pointer; millicomand: Int64;Obj: TPointIndex):TPointIndex	Ссылка на функцию вычисления ресурсов компьютера для выполнения милликоманды (во время моделирования)

34.2 Входные и выходные данные (милликоманды)

Индекс милликоманды	Мнемоника милликоманды	Описание
995	ProgExec	Выполнить миллипрограмму
995	ContextPop	Выдать указатель на контекст ФУ
999	ContextPopMk	Выдать милликоманду с указателем на контекст ФУ
920	FUContunueWork	Продолжить работу ФУ (в режиме ручного управления
918	SchedulerContextSet	Установить ссылку на контекст

		ФУ-планировщика
924	PrefixProgSet	Установить ссылку на подпрограмму, запускаемую до обработки пришедшей милликоманды
922	PostfixProgSet	Установить ссылку на подпрограмму, запускаемую после обработки пришедшей милликоманды
990	ProgExec	Выполнить миллипрограмму
910	ModelingReset	Сброс настроек моделирования
916	ManualModeSet	Установить режим ручного управления ФУ (режим используется при моделировании и отладке)
914	MkQueueCountPop	Выдать количество ИП, находящихся в очереди на выполнение к ФУ
913	MkQueueCountPopMk	Выдать милликоманду с количеством ИП, находящихся в очереди на выполнение к ФУ
912	MkQueueCapsPop	Выдать указатель на капсулу с очередью милликоманды на исполнение
911	MkQueueCapsPopMk	Выдать милликоманду с указателем на капсулу с очередью милликоманды на исполнение

35 Потоковое АЛУ с плавающей точкой (StreamFloatALU)

35.1 Функциональное назначение

Производит арифметические операции над числами в потоковом режиме (вычисления начинаются в тот момент, когда на ФУ приходят все необходимые данные для вычислений, милликоманда, прикрепляемая к результату вычислений хранится в контексте ФУ)

35.2 Контекст

Наименование поля контекста	**Тип**	**Описание**
Operand	Variant	Поле для хранения одного из пришедших операндов
Operation	Integer	Код текущей операции (Операция 0 - нет операции)
ResultReceivers	array of TContextAndMk	Массив указателей на контексты и Милликоманды для ФУ-потребителей результата
Result	Variant	значение результата вычисления последней операции (null - нет результата)
SumTime, SubTime, MulTime, DivTime, SQRTime, SQRTTime, SinTime, CosTime, TgTime, CtgTime, AsinTime, AcosTime, AtgTime	Variant	Модельное время выполнения арифметических и тригономерических операций
CornerMode	int64	Режим измерения углов (0 - радианы, 1 – градусы)

35.3 Входные и выходные данные (милликоманды)

Мнемоника милликоманды	**Описание**
OperandsReset	Сброс операндов
ReceiverContextSet	Установить контекст ФУ-потребителя
ReceiverMkSet	Установить милликоманду для потребителя (автоматически создается новая запись описания потребителя
ReseaversReset	Сбросить список потребителей
CornerModeSet	Установить режим измерения угла (0-радианы, 1 – градусы)
Arifmetical	**Арифметичечкие операции**
Sum1	Первое слагаемое
Sum2	Второе слагаемое
Sub1	Вычитаемое
Sub2	Вычитатель
Mul1	Умножаемое

Mul2	Умножитель
Div1	Делимое
Div2	Делитель
Sqr	Квадрат
Sqrt	Квадратный корень
Trigonometrical	Тригонометрические операции
Cos	Синус
Sin	Косинус
Tg	Тангенс
Ctg	Котангенс
Acos	Арккосинус
Asin	Арксинус
Atg	Арктангенс
Modeling	Настройки моделирования
Arifmetical	Настройки моделирования арифметич. операций
SumTimeSet	Установить время суммирования
SubTimeSet	Установить время вычитания
MulTimeSet	Установить время умножения
DivTimeSet	Установить время деления
SqrTimeSet	Установить время вычисления квадрата
SqrtTimeSet	Установить время вычисления квадратного корня
Trigonometrical	Настройки моделированя тригонометр. операций
SinTimeSet	Установить время вычисления синуса
CosTimeSet	Установить время вычисления косинуса
TgTimeSet	Установить время вычисления тангенса
CtgTimeSet	Установить время ычисления котангенса
AcosTimeSet	Установить время вычисления арккосинуса
AsinTimeSet	Установить время вычисления арксинуса
AtgTimeSet	Установить время вычисления квадратного арктангенса

35.4 Примеры программирования

\\ Установить ссылку на контекст Шины для ВФУ Автомат

Bus.ContextPopMk=Automat.ContextSet

Лист регистрации изменений								
Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Входящий № сопроводит. докум. и дата	Подп.	Дата
измененных	замененных	новых	аннулированных					