

УТВЕРЖДЕН

29257777.509000.162.ЭД-ЛУ

ПРОГРАММА «Арифметико-логическое устройство»

Описание программы

...А.В.00001-01 32 01

Листов

2012

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

АННОТАЦИЯ

В данном программном документе приведено описание программы Арифметико-логическое устройство. 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:

- 1) Прием и накопление операндов
- 2) Выполнение арифметико-логических операций
- 3) Автоматическая рассылка результата вычисления
- 4) Выдача результата по запросу
- 5) Обработка ошибок при вычислениях
- 6) Вызов подпрограмм по флагам результата вычисления

Оформление программного документа «Описание программы» произведено по требованиям ЕСПД ГОСТ 19.402-78¹.

¹ ГОСТ 19.402-78 ЕСПД. Описание программы

СОДЕРЖАНИЕ

АННОТАЦИЯ.....	2
СОДЕРЖАНИЕ.....	3
1 ОБЩИЕ СВЕДЕНИЯ	4
1.1 Обозначение и наименование программы.....	4
1.2 Программное обеспечение, необходимое для функционирования программы	4
1.3 Языки программирования, на которых написана программа.....	4
2 ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ	5
2.1 Классы решаемых задач	5
2.2 Назначение программы	5
2.3 Сведения о функциональных ограничениях на применение.....	5
3 ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ.....	5
3.1 Описание виртуального функционального устройства	5
3.2 Алгоритм работы виртуального функционального устройства.....	6
3.3 Формат милликоманды	7
3.4 Коммуникация между ВФУ	8
3.5. Структура программы	8
3.6. Работа с ВФУ АЛУ	8
4 ИСПОЛЬЗУЕМЫЕ ТЕХНИЧЕСКИЕ СРЕДСТВА	10
5 ВЫЗОВ И ЗАГРУЗКА.....	10
6 ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ	10
6.1 Сведения о входных данных	10
6.1.1. ОА-программа в текстовом виде.....	10
6.1.2. ОА-программа в виде индексного массива.....	11
6.2 Сведения о выходных данных	12
<i>Лист регистрации изменений</i>	<i>13</i>



ПРИЛОЖЕНИЕ А

ПРИЛОЖЕНИЕ Б

1 ОБЩИЕ СВЕДЕНИЯ

1.1 Обозначение и наименование программы

Программа «Арифметико-логическое устройство» имеет следующие атрибуты:

- | | |
|-----------------------------------|---|
| • Наименование исполняемого файла | - millicom.exe |
| • Размер исполняемого файла | -  |
| • «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 C++.

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;
- 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 виртуальными 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:

<i>Procedure IntALU(context: Pointer; millicomand: int64; Obj1: TPointIndex);</i>

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 вычислений; 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. 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):

$$\text{ExtendedMillicom} = \text{NFU} * \text{MilliRange} + \text{MillicomIndex}, \quad (1)$$

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:

$$NFU = \text{ExtendedMillicom} \div \text{MilliRange}, \quad (2)$$

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):

$$NFU = \text{ExtendedMillicom} \bmod \text{MilliRange}, \quad (3)$$

где \bmod – операция нахождения остатка от целочисленного деления.

Описание основных типов ВФУ дается в **Приложении А²**.

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.

3.6. Работа с ВФУ АЛУ

Создание АЛУ производится с помощью милликоманды создания ВФУ с указанием типа ВФУ. Для целочисленного потокового АЛУ создание осуществляется с помощью конструкции ОА-языка `NewFU={Mnemo="ALU" FUType=FUStreamIntALU}`, дробного

² Описание основных типов ВФУ См. Приложение А

АЛУ - NewFU={Mnemo="ALU" FUType=FUStreamFloatALU}, потокового менеджера - NewFU={Mnemo="Manager" FUType=FUStreamManager}. Т.е. используются следующие типы ВФУ: потоковый менеджер – FUStreamManager, целочисленное АЛУ – FUStreamIntALU, дробного АЛУ – FUStreamFloatALU.

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 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): Мнемо*2. Во-вторых, автоматически присвоить значение атрибуту может компилятор, для этого в текста ОА-программы должна присутствовать отдельная мнемоника: Мнемо.

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 отделяются 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».

[illegible]

