

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ имени академика С. П. КОРОЛЕВА»
(САМАРСКИЙ УНИВЕРСИТЕТ)**

Е.В.Симонова

ОПИСАНИЕ ЯЗЫКА GPSS WORLD

Учебное пособие

**по дисциплине «Моделирование
информационно-вычислительных систем»**

САМАРА 2010

УДК 004.9 (075)
ББК 32.97

Симонова Е.В. Описание языка GPSS World // Учебное пособие.
Самара: Самарский государственный аэрокосмический университет
имени академика С.П. Королева, 2010. – 80 с.

Компьютерное моделирование – один из наиболее мощных и универсальных методов исследования и оценки сложных систем, поведение которых зависит от воздействия случайных факторов.

Одним из наиболее эффективных и самых распространенных является язык моделирования GPSS (General Purpose Simulation System), который используется для построения дискретных имитационных моделей и проведения компьютерных экспериментов.

Учебное пособие предназначено для студентов, обучающихся по специальности 230102 – “Автоматизированные системы обработки информации и управления”.

Разделы учебного пособия последовательно раскрывают структуру и состав объектов языка GPSS, предназначенных для реализации имитационных моделей. Учебное пособие содержит описание методики и примеры составления моделей на языке GPSS, что имеет большое учебно-методическое значение и необходимо при самостоятельной работе студентов во время выполнения ими лабораторных работ и курсового проекта по указанным дисциплинам.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	5
1 GPSS – универсальный язык моделирования динамических систем с дискретными событиями	6
1.1 Общая характеристика языка GPSS.....	6
1.2 Краткое описание объектов GPSS	7
1.3 Стандартные числовые атрибуты GPSS	8
1.4 Часы модельного времени в GPSS.....	12
1.5 Формат программы на языке GPSS	12
2 Группы элементов GPSS	14
2.1 Группа основных элементов	14
2.1.1 Транзакт	14
2.1.2 Блок	14
2.1.2.1 Блок генерации транзакта GENERATE	15
2.1.2.2 Блок уничтожения транзакта TERMINATE	16
2.1.2.3 Блок задержки движения транзакта ADVANCE	16
2.1.3 Управление продолжительностью моделирования. Организация таймеров.....	17
2.2 Группа элементов, имитирующих оборудование	17
2.2.1 Одноканальное устройство	18
2.2.1.1 Блок занятия устройства SEIZE.....	18
2.2.1.2 Блок освобождения устройства RELEASE.....	19
2.2.1.3 Организация обслуживания с прерыванием. Блоки PREEMPT и RETURN	19
2.2.2 Многоканальное устройство	21
2.2.3 Логический переключатель	23
2.3 Группа статистических элементов.....	23
2.3.1 Сбор статистики об ожидании транзакта. Блоки QUEUE и DEPART.....	24
2.3.2 Таблицы.....	26
2.4 Группа вычислительных элементов	27
2.4.1 Генератор случайных величин	28
2.4.2 Переменные.....	28
2.4.2.1 Арифметические переменные.....	29
2.4.2.2 Булевы переменные	31
2.4.3 Функции	33
2.5 Группа ссылочных элементов	37
2.5.1 Ячейки сохраняемых величин.....	37
2.5.2 Матрицы ячеек сохраняемых величин.....	39
3 Транзактно-ориентированные блоки GPSS	40
3.1 Работа с параметрами транзакта	40
3.1.1 Установка значений параметров транзакта. Блок ASSIGN.....	40
3.1.2 Отметка времени транзакта. Блок MARK.....	41
3.2 Установка приоритета транзакта. Блок PRIORITY	42
3.3 Изменение направления движения транзакта	43
3.3.1 Переход транзакта в блок, отличный от последующего. Блок TRANSFER.....	43
3.3.2 Изменение направления движения транзакта в зависимости от состояния	

оборудования. Блок GATE	45
3.3.3 Изменение направления движения транзакта в зависимости от выполнения логических условий, определенных на множестве СЧА. Блок TEST	47
<u>3.4 Организация циклов. Блок LOOP</u>	<u>48</u>
<u>3.5 Обработка транзактов, принадлежащих одному семейству</u>	<u>49</u>
3.5.1 Создание копий транзактов. Блок SPLIT	49
3.5.2 Синхронизация движения транзактов. Блоки MATCH, ASSEMBLE, GATHER	50
<u>4 Управление процессом моделирования в GPSS World</u>	<u>52</u>
4.1 Списки пользователя. Блоки LINK и UNLINK	54
4.2 Команда просмотра списка текущих событий. Блок BUFFER	57
<u>5 Команды GPSS World</u>	<u>58</u>
5.2 Оператор INITIAL	58
5.3. Команда RESET	59
5.4 Команда CLEAR	60
5.5 Команда RMULT	61
5.6 Оператор EQU	62
5.7 Операторы описания объектов	63
5.8 Команда EXIT	63
<u>6 ЯЗЫК PLUS</u>	<u>64</u>
6.1 Краткая характеристика языка PLUS	64
6.2 Пример использования языка PLUS	4
<u>7 Диалоговые возможности GPSS World</u>	<u>4</u>
7.1 Диалоговые окна	4
7.2 Стандартная выходная статистика. Описание элементов файла статистики	8
<u>ЗАКЛЮЧЕНИЕ</u>	<u>14</u>
<u>Библиографический список</u>	<u>15</u>
<u>Приложения</u>	<u>16</u>
Приложение А Операторы описания блоков GPSS World	16
Приложение Б Операторы описания данных и контроля управления GPSS World	18
Приложение В Сообщения GPSS World об ошибках	19

ВВЕДЕНИЕ

GPSS World (General Purpose System Simulation World) – глобальная общецелевая система моделирования – мощная универсальная среда моделирования, предназначенная для профессионального моделирования разнообразных процессов и систем.

В основу системы GPSS World положен язык имитационного моделирования GPSS, основными достоинствами которого являются следующие:

- наиболее важные классы объектов и их свойства широко используются в реальных вычислительных сетях, производственных и коммерческих системах и т.п.;
- язык прост в изучении и использовании, постоянно совершенствуется;
- расширение создаваемых моделей легко осуществимо.

В GPSS World появились дополнительные возможности:

- по всем классам объектов и переменных реализованы динамические графические окна, в которых в реальном времени представляется промежуточная выходная статистика;
- гибкий процедурный язык PLUS, который может быть использован для построения моделей и в процедурах проведения экспериментов;
- введены средства проведения факторного анализа, дисперсионного и регрессионного анализа, оптимизации на основе методологии оптимального планирования эксперимента.

Система имеет большой набор команд для управления процессом моделирования, которые можно использовать в интерактивном режиме и включать в модель. Обеспечена возможность проведения экспериментов, сгенерированных системой, пользовательских и оптимизационных.

В GPSS World используется большое число окон, упрощающих просмотр и анализ объектов модели. Можно открыть любое число графических окон для просмотра динамики изменения объектов модели. В GPSS World имеется библиотека распределения вероятностей, что облегчает моделирование стохастических факторов. Система имеет встроенный текстовый редактор.

Использование GPSS World значительно ускоряет процесс исследования и оптимизации технических систем, а также позволяет определять параметры функционирования сложных систем, реальный эксперимент с которыми невозможен либо очень дорог.

1 GPSS – УНИВЕРСАЛЬНЫЙ ЯЗЫК МОДЕЛИРОВАНИЯ ДИНАМИЧЕСКИХ СИСТЕМ С ДИСКРЕТНЫМИ СОБЫТИЯМИ

1.1 Общая характеристика языка GPSS

Система GPSS (General Purpose Simulation System) предназначена для имитационного моделирования сложных дискретных систем. Имитационное моделирование обеспечивает возможность испытания, оценки и проведения экспериментов с исследуемой системой без каких-либо непосредственных воздействий на нее. Система моделирования GPSS предоставляет пользователю достаточно краткий и, в то же время, применимый к широкому классу систем язык моделирования.

Функциональное описание моделируемой системы строится согласно следующим принципам:

1. Статика. Моделируемая система может быть описана в терминах конечного набора абстрактных элементов – «объектов». Элемент системы однозначно идентифицируется своим типом и номером (или именем). Совокупность типов элементов образует группу элементов. С каждым типом элемента связано множество стандартных числовых атрибутов (СЧА), являющихся характеристиками элемента. Для каждого СЧА определено множество его возможных значений. Совокупность значений СЧА элемента полностью определяет состояние этого элемента. Над каждым типом элемента определены возможные действия. На рисунке 1 представлена систематизированная схема языка GPSS.

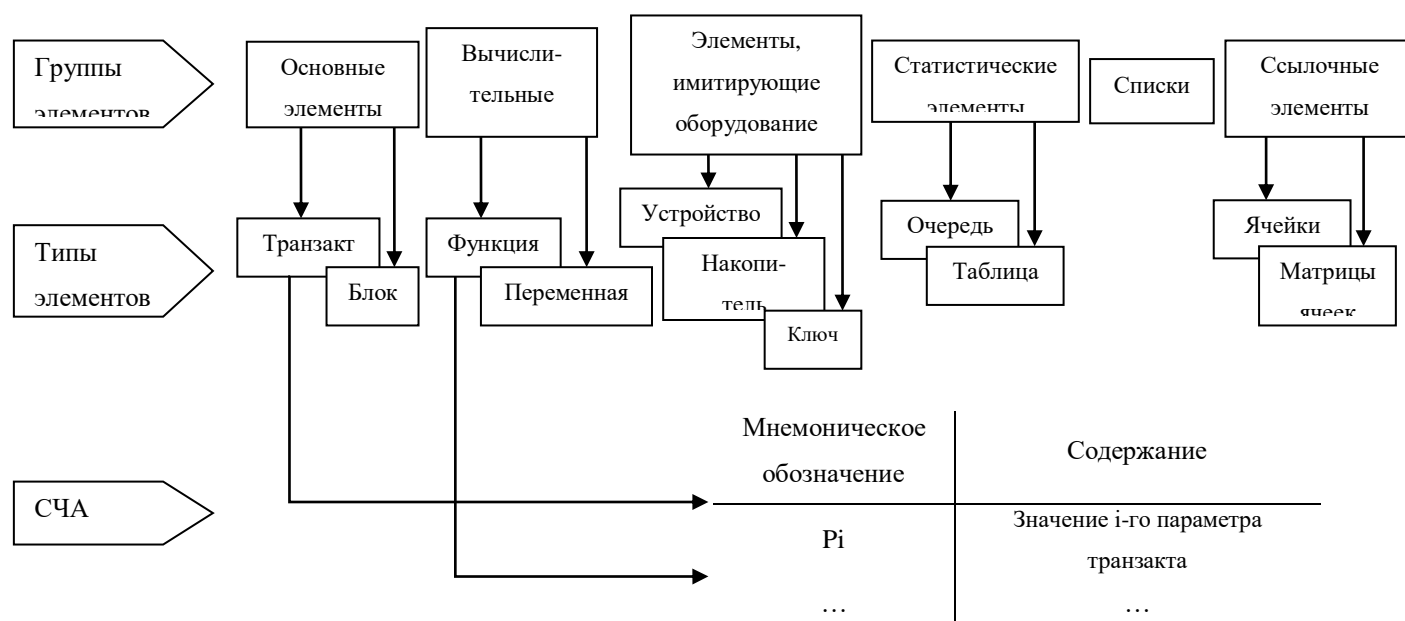


Рисунок 1 – Систематизированная схема языка GPSS

2. Динамика. Моделируемая система функционирует во времени. Время – системный атрибут языка. Любое изменение состояния элемента является событием. Последовательность событий в элементе является динамическим процессом функционирования элемента. Последовательность событий в системе является динамическим процессом функционирования системы.

1.2 Краткое описание объектов GPSS

Объект типа «Транзакт». Транзакты – это динамические объекты GPSS. Они создаются в определенных точках модели, продвигаются интерпретатором через блоки, а затем уничтожаются. Транзакты являются аналогами единиц потоков в реальной системе.

Объект типа «Блок». Блоки языка GPSS выполняют соответствующие операции в модели. Блоки реализуют операций четырех основных типов; создание или уничтожение транзактов, изменение числового атрибута объекта, задержка транзактов на определенный период времени, изменение маршрута транзакта в модели.

Объект типа «Одноканальное устройство». Объект этого типа представляет собой оборудование, которое в данный момент времени может быть занято только одним транзактом.

Объект типа «Многоканальное устройство». Объекты этого типа представляют собой оборудование для параллельной обработки, которое может быть использовано несколькими транзактами одновременно,

Объект типа «Логический переключатель». Объект типа «ключ» может находиться в состоянии «включен», «выключен», «инвертирован». Состояние ключа может быть изменено некоторым транзактом, любой транзакт может использовать состояние ключа для выбора одного из двух возможных путей или ожидать момента изменения состояния ключа.

Объект типа «Арифметическая переменная». Арифметические переменные позволяют вычислять арифметические выражения, состоящие из стандартных числовых атрибутов (СЧА), которые описаны ниже.

Объект типа «Булева переменная». Булевы переменные позволяют проверять в одном блоке одновременно несколько условий, исходя из состояния объектов и значений их атрибутов.

Объект типа «Функция». Используя функции, программист может производить вычисление непрерывных или дискретных функциональных

зависимостей между аргументом и значением функции.

Объект типа «Очередь». Объекты типа «Очередь» предназначены для сбора статистики об использовании оборудования транзактами.

Объект типа «Таблица». Таблица – объект, который определяет форму вывода статистической информации.

1.3 Стандартные числовые атрибуты GPSS

Стандартными числовыми атрибутами (СЧА) называются атрибуты объектов, значения которых может использовать и изменять программист.

СЧА определяется следующим образом:

СЧА ::= <прямая адресация> | <косвенная адресация>

прямая адресация ::=

<мнемоническое обозначение типа элемента> <номер элемента> |

< мнемоническое обозначение типа элемента> \$ <имя элемента>

косвенная адресация ::= <мнемоническое обозначение типа элемента> * <номер параметра транзакта>

мнемоническое обозначение типа элемента ::= F | Q | S | FN | ...

номер параметра транзакта ::= 1 | 2 | 3 | ... | 100

Если объект идентифицирован с помощью номера, то ссылка на его СЧА записывается как СЧА_j, где j – номер объекта (целое число). При символической идентификации объекта ссылка на его СЧА записывается как СЧА\$<имя>.

Пример использования СЧА:

- *FN1* – прямая адресация объекта «функция» с помощью номера, обращение к данному СЧА позволяет получить значение функции с номером 1;
- *FN\$PUASS* – прямая адресация объекта «функция» с помощью имени, обращение к данному СЧА позволяет получить значение функции с именем PUASS;
- *FN*1* – косвенная адресация объекта «функция», обращение к данному СЧА позволяет получить значение функции, номер которой определяется содержимым 1-го параметра транзакта, поступающего в блок, в поле операндов которого используется обращение к функции.

Каждый объект GPSS имеет свой набор СЧА (Таблица 2). Кроме СЧА объектов, существуют системные СЧА (Таблица 3), к которым программист может обращаться в модели, но не может изменять их значение.

Таблица 2 – Стандартные числовые атрибуты объектов GPSS

Группы объектов	Типы объектов	СЧА	
		Мнемоническое обозначение	Содержание
Основные объекты	Транзакт	Pj	Значение параметра j текущего транзакта
		MPj	Значение времени, равное разности абсолютного модельного времени и содержимого j-го параметра текущего транзакта
	Блок	Nj	Общее число входов транзактов в j-й блок
		Wj	Текущее число транзактов, которые находятся в блоке j
Объекты, имитирующие оборудование	Многоканальное устройство (накопитель)	Sj	Текущее содержимое многоканального устройства j
		Rj	Число свободных единиц многоканального устройства j
		SRj	Коэффициент использования многоканального устройства j в тысячных долях
		SAj	Среднее содержимое многоканального устройства j (целая часть)
		SMj	Максимальное содержимое многоканального устройства j
		SCj	Общее число входов в многоканальное устройство j
		STj	Среднее время пребывания транзактов в многоканальном устройстве j
		SEj	Признак пустоты многоканального устройства j: 1 – пусто
		SFj	Признак заполнения

	Одноканальное устройство		памяти j: 1- заполнена
		SVj	Признак доступности памяти j: 1- доступна
		Fj	Текущее состояние устройства j: 0 – устройство свободно
		FIj	Признак прерывания устройства j: 1 – устройство находится в состоянии прерывания
		FVj	Признак доступности устройства j: 1 – доступно
		FRj	Коэффициент использования устройства j в тысячных долях
		FCj	Общее число входов в устройство j
		FTj	Среднее время использования устройства одним транзактом
Вычислительные объекты	Логический ключ	LSj	Состояние логического ключа: 1 - установлен
	Функция	FNj	Вычисленное значение функции j. От значения функции берется целая часть, за исключением использования в качестве модификатора в блоках GENERATE, ADVANCE или ASSIGN или в качестве аргумента другой функции
	Переменная	Vj	Вычисленное значение переменной j. При вычислении значения переменной с фиксированной точкой получается целое число. При вычислении значения переменной с плавающей точкой дробная часть результата отбрасывается

		BVj	Вычисленное значение (1 или 0) булевой переменной
Статистические объекты	Очередь	Qj	Длина очереди j
		QAj	Средняя длина очереди j
		QMj	Максимальная длина очереди j
		QCj	Общее число входов в очередь j
		QZj	Число нулевых входов в очередь j
		QTj	Среднее время пребывания транзакта в очереди j (включая нулевые входы)
		QXj	Среднее время пребывания транзакта в очереди j (без нулевых входов)
	Таблица	TBj	Вычисленное среднее значение таблицы j
		TCj	Общее число включений в таблицу j
		TDj	Содержимое ячейки j
Сохраняемые объекты	Ячейка сохраняемых величин	Xj	Содержимое ячейки j
	Матрица ячеек сохраняемых величин	MXj (a, b)	Содержимое элемента матрицы ячеек j, расположенного в строке a, столбце b
Списковые объекты	Списки пользователя	CNj	Текущее число транзактов в j-м списке пользователя
		CAj	Среднее число транзактов в j-м списке пользователя
		CNj	Максимальное число транзактов в j-м списке пользователя
		CCj	Общее число транзактов в j-м списке пользователя
		CTj	Среднее время пребывания транзакта в j-м списке пользователя

Таблица 3 – Системные стандартные числовые атрибуты

Мнемоническое обозначение	Содержание
RNj, j=1.. ∞	Значение, вычисляемое j-м датчиком случайных чисел
C1	Текущее значение относительного времени
AC1	Текущее значение абсолютного времени
TG1	Число, равное текущему значению счетчика завершений
XN1	Номер активного транзакта
M1	Время пребывания в модели транзакта, обрабатываемого программой в данный момент

1.4 Часы модельного времени в GPSS

События в модели происходят в модельном времени. Часы модельного времени в GPSS регистрируют как целочисленные, так и вещественные значения. Но с целью ускорения процесса моделирования предпочтительным является использование целочисленных значений, т.к. операции целочисленной арифметики выполняются процессором ЭВМ быстрее.

Единица модельного времени определяется программистом. Значение принятой единицы модельного времени выражают в неявном виде в форме временных данных модели. Если все данные выражены в минутах, то единицей модельного времени будет минута, т.е. масштаб времени в модели будет следующий: одна единица модельного времени равна одной минуте реального времени. Программист задает такую единицу модельного времени, которая ему удобна для того, чтобы с необходимой степенью детализации отобразить в модели события, происходящие в реальной системе.

1.5 Формат программы на языке GPSS

Разработка GPSS-модели состоит из двух этапов:

1. В соответствии с принципом функционального соответствия между элементами моделируемой системы (реальной) и элементами модели устанавливается взаимно однозначное соответствие, которое оформляется в виде таблицы. Единица модельного времени (MB), соответствующая единичному изменению показаний часов реального времени, определяется программистом, который должен задавать все интервалы времени в

выбранных им единицах. Пример таблицы функционального соответствия приведен в Таблица 1.

Таблица 1 – Таблица функционального соответствия

Эл-т моделируемой системы (реальной)	Элемент модели
1 сек	10 единиц MB
...	...

2. Функциональное описание моделируемой системы на GPSS состоит из двух частей, которые являются отдельными фрагментами программы:

- 2.1. Описание вычислительных, статистических, ссылочных элементов и элементов, имитирующих оборудование (т.е., описание статики). В простых программах эта часть может отсутствовать.
- 2.2. Описание процессов, развивающихся в системе, в форме абстрактных действий над элементами языка (т.е., описание динамики).

GPSS – язык интерпретируемого типа, он связан с пошаговым выполнением инструкций каждого отдельно взятого оператора. Программа на языке GPSS имеет блочную структуру, т.е. управление осуществляется с помощью инструкций, называемых блоками.

Формат инструкции GPSS

[<метка>] <операция> <операнды> [<;комментарий>]

Каждая инструкция языка записывается в отдельной строке и содержит следующие поля:

- поле метки: в поле метки записывается либо номер, либо идентификатор блока, который представляет собой алфавитно-цифровую последовательность длиной до 5 символов, начинающуюся с буквы;
- поле операций, в котором указывается наименование действия над элементами языка, т.е. обозначение соответствующего блока;
- поле операндов, в котором записываются СЧА элементов; поле операндов состоит из подполей <A>, , <C>, <D>, <E>, <F>, G>, содержимое которых отделяется друг от друга запятыми; если одно из подполей операндов необходимо опустить, пробел не ставится, вместо него ставится запятая.

2 ГРУППЫ ЭЛЕМЕНТОВ GPSS

2.1 Группа основных элементов

Группу основных элементов GPSS образуют транзакт и блок.

2.1.1 Транзакт

Транзакт – основное понятие динамического процесса функционирования модели системы, единственный тип элемента, который существует в модели временно. Любой транзакт характеризуется набором параметров, которые являются множеством СЧА, принадлежащих транзакту. Параметры транзакта не имеют никаких ограничений на их физический смысл. Конкретный физический смысл транзакта и его параметров определяется предметной областью решаемой задачи. Транзактом может быть заявка, пришедшая на обслуживание в систему массового обслуживания (СМО), автомобиль при моделировании работы автостоянки, отказ системы при моделировании надежности системы и т.п. Параметры транзакта – единственная разновидность атрибутов, значения которых устанавливаются и модифицируются только в соответствии с логикой программы модели, заданной программистом. СЧА всех остальных элементов изменяются как под влиянием программы, так и под влиянием симулятора, т.к. их семантика заранее однозначно определена в языке. Параметры транзакта – единственный вид СЧА, существующих в модели временно.

2.1.2 Блок

Блок – тип элемента, который определяет действия над всеми остальными элементами GPSS. Блок ассоциируется с названием действия, которое определяет тип блока, например, блок «задержать транзакт», блок «занять устройство». С точки зрения кодировки программы модели алгоритм определяется последовательностью блоков, т.к. блок – основная инструкция языка GPSS. Действие, связанное с любым блоком, выполняется, когда этому блоку передается управление, т.е. когда в этот блок входит транзакт. Т.к. в модели одновременно могут находиться несколько транзактов, одновременно в модельном времени могут выполняться действия, связанные с различными блоками, но в реальном времени в каждый момент может обрабатываться только один транзакт. Перемещение транзакта от блока к блоку называется движением транзакта по программе модели. Это движение разворачивается во времени. Множество движущихся транзактов определяет протекание в системе

параллельных процессов.

В самом начале моделирования в GPSS-модели нет ни одного транзакта. В процессе моделирования транзакты входят в модель в определенные моменты времени, обрабатываются в модели и, в конечном итоге, покидают модель. В связи с этим над транзактами возможны следующие действия, которые реализуют соответствующие блоки:

- ввести транзакт в модель – блок GENERATE,
- вывести транзакт из модели – блок TERMINATE,
- задержать транзакт – блок ADVANCE.

2.1.2.1 Блок генерации транзакта GENERATE

Блок генерации GENERATE генерирует транзакт и направляет его к следующему блоку программы, поэтому следующим за GENERATE должен быть блок, который всегда может принять транзакт.

Формат блока:

GENERATE <A>,,<C>,<D>,<E>

Интервалы времени между транзактами, поступающими в модель, определяются содержимым полей операндов:

- <A> – среднее время между поступлениями транзактов в модель, по умолчанию, среднее время равно 0;
- – модификатор времени (описан ниже);
- <C> – начальная задержка, т.е. момент времени появления в модели первого транзакта, по умолчанию начальная задержка равна 0;
- <D> – общее число транзактов, которые должны быть введены в модель, по умолчанию, в модель вводится неограниченное число транзактов;
- <E> – приоритет транзакта (0..127), чем больше значение, тем выше приоритет транзакта, по умолчанию приоритет транзакта равен 0.

В поле может быть задан *модификатор времени* одного из двух типов:

- *Модификатор-интервал* – определяет целочисленную случайную величину T , принимающую равновероятные значения в диапазоне $[\langle A \rangle - \langle B \rangle .. \langle A \rangle + \langle B \rangle]$. Значение $\langle B \rangle$ должно быть меньше $\langle A \rangle$. Например, блок GENERATE 9,2 определяет, что интервал времени между поступлением транзактов в модель, с равной вероятностью принимает значения 7, 8, 9, 10, 11.
- *Модификатор-функция* – определяет интервал времени между поступлением транзактов в модель как целочисленную случайную величину $T = \langle A \rangle * \langle B \rangle$. В поле операнда $\langle B \rangle$ записывается СЧА –

значение функции. Например, блок GENERATE 2, FN\$EXPON определяет, что интервал времени между поступлениями транзактов в модель вычисляется как значение функции с именем EXPON, умноженное на 2.

В модели может быть несколько блоков GENERATE в соответствии с тем, какое количество параллельных процессов имитирует модель.

2.1.2.2 Блок уничтожения транзакта TERMINATE

Формат блока:

TERMINATE <A>

Транзакты, попадающие в этот блок, выводятся из модели и больше не участвуют в процессе моделирования. В поле операнда <A> записывается либо целое число, либо ничего. Каждый раз, когда транзакт входит в блок TERMINATE, целое число, стоящее в поле операнда <A> вычитается из счетчика завершений, который устанавливается управляющей командой START. Как только значение счетчика завершений обнулится, моделирование закончится. Например, конструкция

TERMINATE 1

START 100

обеспечивает такую длительность моделирования, при которой через программу модели пропускается 100 транзактов.

Если в поле операнда <A> ничего не указано, счетчик завершений не уменьшается и моделирование продолжается бесконечно.

2.1.2.3 Блок задержки движения транзакта ADVANCE

Формат блока:

ADVANCE <A>,

Задержка движения транзакта во времени имитируется при попадании транзакта в блок ADVANCE, для которого в полях операндов <A> и указываются соответственно среднее время задержки и модификатор времени, использование которого аналогично блоку GENERATE. Например,

ADVANCE 9, 2

ADVANCE 2, FN\$EXPON

2.1.3 Управление продолжительностью моделирования. Организация таймеров

Длительность моделирования в программе на GPSS можно задать двумя способами.

1. Определить в управляющей команде START количество транзактов, которые необходимо обработать в модели (этот способ используется для простых моделей, содержащих единственный процесс):

GENERATE...

<программа модели>

TERMINATE 1

START 100

2. С помощью *процесса-таймера* определить отрезок модельного времени, в течение которого должно осуществляться моделирование. Процесс-таймер должен быть единственным на всю модель, поэтому только в нем в блоке TERMINATE задается непустое поле операнда <A>. Во всех остальных процессах поле операнда <A> в блоке TERMINATE должно быть пустым:

; 1-й процесс

GENERATE...

<программа модели>

TERMINATE

...

; n-й процесс

GENERATE...

<программа модели>

TERMINATE

; процесс-таймер

GENERATE ,,1

ADVANCE 100

TERMINATE 1

START 1

Процесс-таймер реализуется транзактом, который вводится в модель в начале моделирования, задерживается на 100 единиц модельного времени, затем выводится из модели, что приводит к завершению всех процессов модели.

2.2 Группа элементов, имитирующих оборудование

В группу элементов, имитирующих оборудование, входят одноканальное устройство, многоканальное устройство, логический переключатель.

2.2.1 Одноканальное устройство

Одноканальное устройство (далее: устройство) используется для имитации любого элемента системы, функционирование которого во времени можно представить сменой двух состояний: «свободно» и «занято». Устройство характеризуется следующими свойствами:

Каждое устройство в любой момент времени может обслуживать только один транзакт. Если в процессе обслуживания появляется новый транзакт, его поведение определяется одним из трех вариантов:

- новый транзакт должен подождать своей очереди;
- новый транзакт должен направиться в другое место;
- если вновь пришедший транзакт имеет больший приоритет, устройство прерывает текущее обслуживание и начинает обслуживать новый транзакт.

Для работы с одноканальными устройствами используются следующие действия, которые реализуют соответствующие блоки:

- занять устройство – блок SEIZE,
- освободить устройство – блок RELEASE,
- прервать обслуживание на устройстве – блок PREEMPT,
- снять прерывание – блок RETURN.

2.2.1.1 Блок занятия устройства SEIZE

Формат блока:

SEIZE <A>

Операнд <A> определяет номер или имя устройства. Если в текущий момент времени устройство используется, транзакт не может войти в блок и должен ожидать своей очереди. Таким образом, перед блоком SEIZE может образоваться очередь транзактов.

Если устройство свободно, транзакт может войти в блок. При этом состояние устройства изменяется со «свободно» на «занято».

Предварительного объявления устройства в модели не требуется.

Пример использования блока (транзакт занимает устройство с именем CHAN):

SEIZE CHAN

Устройство находится в состоянии занятости до тех пор, пока не окончится обслуживание.

2.2.1.2 Блок освобождения устройства RELEASE

Формат блока:
RELEASE <A>

Операнд <A> определяет номер или имя устройства. Предназначением блока RELEASE является изменение состояния ранее занятого устройства с «занято» на «свободно». Блок RELEASE никогда не запрещает вход транзакта. Транзакт не может освободить устройство, которое он не занимал.

2.2.1.3 Организация обслуживания с прерыванием. Блоки PREEMPT и RETURN

Ситуацию обслуживания транзакта в устройстве с прерываниями можно смоделировать, считая, что отказ оборудования представляет собой транзакт, приоритет которого выше, чем у транзакта, обрабатываемого устройством. В этом случае более приоритетный транзакт должен прервать обслуживание менее приоритетного транзакта, т.е. выгрузить его из устройства. Дословный перевод с английского слова *preempt* – выгрузить, но с точки зрения работы одноканальной СМО принято использовать термин ЗАХВАТИТЬ устройство. Для организации обслуживания в устройстве с прерываниями используют пару блоков PREEMPT (ЗАХВАТИТЬ) – RETURN (ВЕРНУТЬ).

Формат блока:
PREEMPT <A>,,<C>,<D>,<E>

Поля операндов имеют следующий смысл:

- <A> – номер или имя устройства,
- – возможность захвата по приоритету,
- <C> – имя блока, в который переходит прерванный транзакт,
- <D> – номер параметра у прерванного транзакта,
- <E> – возможность снятия транзакта с обслуживания.

Операнд <A> определяет номер или имя устройства, на котором генерируется прерывание.

Операнд задает приоритетный режим (если =PR) или режим

прерывания (если этот операнд опущен). При работе в приоритетном режиме транзакт, уже занимающий устройство или генерирующий на нем прерывание, может быть прерван только транзактом, приоритет которого выше приоритета данного транзакта. Прерванные транзакты претендуют на дополнительное использование устройства, когда прервавший их транзакт войдет в соответствующий блок RETURN. Прерванные транзакты помещаются в список задержки в порядке приоритета.

Операнд <C> задает номер или имя блока, в который в этот же момент времени должен попытаться войти прерванный транзакт. Прерванный транзакт теряет управление устройством, но претендует на право его использования, если только не задан аргумент операнда <E>. В приоритетном режиме работы желательно задавать операнд <C>, если прерывающий транзакт имеет более высокий приоритет, чем прерываемый.

Операнд <D> задает номер параметра, связанного с прерванным транзактом. Если прерываемый транзакт в момент прерывания направляется в список будущих событий, тогда остаток времени записывается в заданный параметр. Если такой параметр не существует, то он создается. В приоритетном режиме работы операнд <D> задают только в том случае, если прерывающий транзакт имеет более высокий приоритет, чем прерываемый транзакт.

Операнд <E> задает либо не задает режим удаления (RE). В режиме удаления RE прерванный транзакт более не претендует на использование устройства и пытается войти в блок, заданный операндом <C> (если в операнде <E> стоит RE, то должен быть определен и операнд <C>). В приоритетном режиме работы режим RE используется только в том случае, если приоритет прерывающего транзакта больше приоритета прерываемого транзакта. При использовании RE прерванный транзакт не должен входить в блоки RELEASE или RETURN, связанные с устройством, в котором обслуживался прерванный транзакт. Если режим RE не задан (операнд <E> опущен), то прерванный транзакт по возвращении в список текущих событий будет вновь пытаться занять устройство.

Блок RETURN предназначен для освобождения ранее захваченного устройства. Формат блока:

RETURN <A>

В операнде <A> задается номер устройства, с которого снимается прерывание. Прерывание может быть снято в блоке RETURN только тем транзактом, которым оно было сгенерировано.

ПРИМЕР 1. В качестве примера использования блоков PREEMPT и RETURN смоделируем обработку отказа оборудования.

```

; 1-й процесс – нормальная работа оборудования
    GENERATE ...
    ...
    SEIZE      CHAN
    ...
    RELEASE   CHAN
    TERMINATE
; 2-й процесс – поток отказов оборудования
    GENERATE .....,1
    PREEMPT   CHAN,PR,ERR,5,RE
    ADVANCE ...
    RETURN    CHAN
    TERMINATE
; обработка события отказа оборудования
ERR ...

```

Во 2-м процессе блок GENERATE в определенный момент времени генерирует транзакт, имитирующий отказ оборудования, приоритет которого равен 1. Он выше, чем приоритет транзакта, моделирующего нормальную работу оборудования. Этот факт отмечается наличием значения PR в операнде транзакта, моделирующего отказ оборудования. Значение RE в операнде <E> означает, что прерванный транзакт снимается с устройства CHAN и переходит на метку ERR, по которой находится группа операторов, моделирующих обработку отказа оборудования. Прерванный транзакт более не претендует на занятие устройства CHAN после того, как прерывающий транзакт освободит его.

2.2.2 Многоканальное устройство

Два или более обслуживающих устройства, работающих параллельно, могут моделироваться в GPSS двумя или более одноканальными устройствами. Обычно это необходимо, когда отдельные устройства являются разнородными, например, имеют различную интенсивность обслуживания.

Однако очень часто параллельно работающие устройства являются одинаковыми, и GPSS предоставляет для их моделирования объект, называемый многоканальным устройством (накопителем или памятью).

Количество устройств, которое моделируется каждым из накопителей, определяется пользователем. В этом смысле употребляют термин «емкость накопителя». Эта емкость заранее должна быть определена пользователем, чтобы интерпретатор знал, сколько устройств использует данный накопитель.

Все используемые в модели накопителя должны быть заранее описаны, т.е. должно быть определено количество однотипных устройств, входящих в накопитель. Для этого используется оператор STORAGE, определяющий емкость накопителя.

Формат оператора задания емкости накопителя STORAGE:

<имя> STORAGE <A>

- поле метки – имя накопителя,
- поле операции – STORAGE,
- поле операнда <A> – емкость накопителя.

MEM STORAGE 100

Блоки ENTER (ВОЙТИ) и LEAVE (ВЫЙТИ). Использование МКУ аналогично использованию одиночного устройства. Элементом, который занимает и использует накопитель, является транзакт. При моделировании накопителя события происходят в следующем порядке:

- 1) транзакт ожидает своей очереди, если это необходимо;
- 2) транзакт занимает устройство;
- 3) устройство осуществляет обслуживание на протяжении некоторого интервала времени;
- 4) транзакт освобождает устройство.

Формат блоков ENTER и LEAVE:

ENTER <A>[,]

LEAVE <A>[,]

Поля операндов имеют следующий смысл:

- <A> – номер или имя накопителя,
- – количество занимаемых одновременно единиц емкости накопителя.

Когда транзакт входит в блок ENTER, интерпретатор выполняет следующие действия:

- увеличивает счетчик входов накопителя на значение операнда ;
- увеличивает текущее содержимое накопителя на значение операнда ;
- уменьшает доступную емкость накопителя на значение операнда .

Когда транзакт входит в блок LEAVE, интерпретатор выполняет обратные действия:

- уменьшает текущее содержимое накопителя на значение операнда ;
- увеличивает доступную емкость накопителя на значение операнда .

Операнду можно присвоить значение, отличное от единицы. По

умолчанию значение операнда равно 1. Освобождение накопителя может быть произведено не только тем транзактом, который его занял, а любым другим транзактом. Перед многоканальным устройством так же, как и перед одноканальным устройством, может возникнуть очередь транзактов.

Пример использования операндов ENTER и LEAVE:

```
ENTER    MEM
LEAVE    MEM
```

2.2.3 Логический переключатель

Логический переключатель (ключ) используется для моделирования объектов, имеющих всего два положения: «включен» (set или 1) и «выключен» (reset или 0).

Блок LOGIC используется для включения, выключения или инвертирования положения ключа. Положение ключа можно проверить любым транзактом в любой части модели.

Формат блока LOGIC:

```
LOGIC <X>    <A>
```

В поле операнда <A> указывается номер или имя ключа.

Когда транзакт входит в блок LOGIC, положение ключа, номер которого задан в операнде <A>, изменяется в зависимости от значения вспомогательного оператора <X> следующим образом:

- S – ключ устанавливается в положение «включен»;
- R – ключ устанавливается в положение «выключен»;
- I – ключ инвертируется, то есть положение его изменяется на противоположное.

Пример использования блока LOGIC:

```
LOGIC S    KEY
```

По умолчанию перед началом моделирования ключи находятся в положении «выключен». Установить ключ в состояние «включен» перед началом моделирования можно с помощью инструкции

```
INITIAL LS    KEY
```

2.3 Группа статистических элементов

2.3.1 Сбор статистики об ожидании транзакта. Блоки QUEUE и DEPART

Эти блоки обеспечивают в GPSS возможность автоматического сбора статистических данных, описывающих вынужденное ожидание, которое может происходить время от времени в различных точках модели.

Система моделирования GPSS обеспечивает возможность сбора статистики с помощью такого средства, как регистратор очереди.

При использовании регистратора очереди в тех точках модели, где число ресурсов ограничено, интерпретатор автоматически начинает собирать различную информацию об ожидании с помощью следующих СЧА:

- 1) число входов транзактов в очередь;
- 2) количество транзактов, которые фактически присоединились к очереди и сразу ее покинули, т.е. имели время ожидания равное нулю;
- 3) максимальная длина очереди;
- 4) среднее число ожидавших транзактов;
- 5) среднее время ожидания тех транзактов, которым пришлось ждать.

В модели может быть несколько регистраторов очередей, различающихся именами. Правила присвоения имен те же, что и для устройств. Разработчик вносит регистратор очереди в модель с помощью пары взаимодополняющих блоков:

```
QUEUE    <A>[,<B>]  
DEPART   <A>[,<B>]
```

При входе транзакта в блок QUEUE (СТАТЬ В ОЧЕРЕДЬ) выполняются четыре действия:

- 1) счетчик входов для данной очереди увеличивается на содержимое операнда ;
- 2) длина очереди (счетчик текущего содержимого) для данной очереди увеличивается на содержимое операнда ;
- 3) транзакт присоединяется к очереди с запоминаем ее имени и значения текущего модельного времени.

Когда транзакт входит в блок QUEUE, то выполняется поиск очереди с именем, определенным операндом А. При необходимости очередь создается.

Блок QUEUE не поддерживает список членов очереди, он только добавляет единицы к длине очереди.

Использование регистратора очереди необязательно. С его помощью интерпретатор собирает лишь статистику об ожидании. Если же регистратор не используется, то статистика не собирается, но везде, где должна возникать

очередь, она возникает. Ожидание является следствием состояния устройства, а не следствием использования регистратора. Если программист не планирует обработку статистических данных об очередях, то лучше не собирать статистику – это позволит уменьшить продолжительность моделирования.

Один и тот же транзакт может одновременно увеличить длину нескольких очередей.

Транзакт перестает быть элементом очереди только после того, как он переходит в блок DEPART (ПОКИНУТЬ ОЧЕРЕДЬ) соответствующей очереди. Когда это происходит, интерпретатор выполняет такие операции:

- 1) длина очереди соответствующей очереди уменьшается на содержимое операнда <V>;
- 2) используя привязку к значению времени, определяет: является ли время, проведенное транзактом в очереди, нулевым; если да, то такой транзакт по определению является транзактом с нулевым пребыванием в очереди и одновременно изменяется счетчик нулевых вхождений,
- 3) ликвидируется «привязка» транзакта к очереди.

ПРИМЕР 2. Смоделировать работу одноканальной системы массового обслуживания, на вход которой поступает поток заявок, в котором интервалы времени между поступлениями заявок распределены равномерно в диапазоне $[4.5 \pm 1.5]$ минуты. Время обслуживания заявки 1 мин. Смоделировать прохождение через модель 100 заявок.

Таблица функционального соответствия приведена в Таблица 2.

Таблица 2 – Таблица функционального соответствия для ПРИМЕР 2

Эл-т моделируемой системы (реальной)	Элемент модели
1 мин	10 единиц MB
Заявка на обслуживание	Транзакт
Очередь	QUE
Одноканальное устройство	CHAN

GENERATE	45,15
QUEUE	QUE
SEIZE	CHAN
DEPART	QUE
ADVANCE	10
RELEASE	CHAN
TERMINATE	1
START	100

2.3.2 Таблицы

GPSS-таблицы служат для сбора статистической информации о любом параметре модели, для накопления выборочных значений случайных величин, для статистической обработки этих выборок и для построения гистограммы распределения вероятностей значений указанного параметра. Графическим аналогом GPSS-таблицы является гистограмма выборочных значений случайной величины, которую можно просмотреть в окне таблицы. Прежде чем использовать таблицу, ее нужно определить, а затем указать, для какого параметра модели следует собрать статистику.

Таблица определяется оператором TABLE. Формат оператора TABLE:

<имя> TABLE <A>,,<C>,<D>

Поля операндов имеют следующий смысл:

- <A> – СЧА параметра, для которого выполняется сбор статистики,
- – верхняя граница самого левого интервала таблицы,
- <C> – ширина интервалов таблицы, за исключением самого левого и самого правого,
- <D> – общее число интервалов таблицы, включая самый левый и самый правый интервалы.

На рисунке 2 показана ось значений некоторого параметра и ее разбиение на ряд интервалов таблицы.

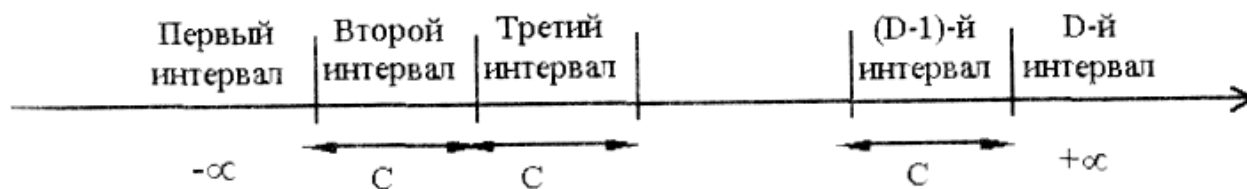


Рисунок 2 – Интервалы таблицы GPSS

Для сбора статистических данных об очередях используется оператор QTABLE. Его формат совпадает с форматом оператора TABLE, за исключением того, что операнд <A> задает имя очереди.

Механизм сбора статистики в таблицу включается, когда транзакт входит в блок TABULATE (ТАБУЛИРОВАТЬ), у которого в поле операнда <A> указано имя или номер таблицы, которая описана оператором TABLE.

Формат блока TABULATE:
 TABULATE <A>

Одну и ту же таблицу можно использовать в нескольких блоках TABULATE модели.

ПРИМЕР 3. Пример использования таблицы для сбора статистической информации о значениях, которые принимает в модели величина, хранящаяся в 1-м параметре транзактов.

TAB TABLE P1,4,3,5
 ...
 TABULATE TAB

В таблице 3 приведены интервалы регистрации значений P1.

Таблица 3 – Интервалы регистрации значений P1

Номер интервала	Граница интервала
1-й	$]-\infty, 4[$
2-й	$[4, 7[$
3-й	$[7, 10[$
4-й	$[10, 13[$
5-й	$[13, \infty[$

Пусть в результате статистического анализа значений 1-х параметров 100 транзактов получены следующие данные, которые приведены в таблице 4.

Таблица 4 – Результаты статистического анализа

Граница интервала	Число попаданий транзактов в интервал	Кумулятивная частота попаданий
4	20	0.20
7	35	0.55
10	15	0.70
13	20	0.90
>13	10	1.00

Гистограммы распределения вероятностей значений параметра P1 строится на основании границ интервалов и кумулятивной частоты попаданий транзактов в указанные интервалы.

2.4 Группа вычислительных элементов

Прежде чем рассматривать вычислительные элементы GPSS, необходимо рассмотреть генератор случайных величин в GPSS.

2.4.1 Генератор случайных величин

В GPSS World количество генераторов случайных величин неограничено. В них реализуется конгруэнтный мультипликативный метод. Генераторы выдают целочисленные значения, равномерно распределенные в диапазоне 0..999999. Число, стоящее в начале генерируемой последовательности, равно номеру генератора. Если генератор используется в качестве аргумента функции или при вычислении переменной, генерируемые им значения находятся в диапазоне 0..0.999999.

2.4.2 Переменные

При построении модели сложных систем возникает необходимость задать сложные математические или логические соотношения между атрибутами системы. Для этой цели в программе используются *переменные*.

В GPSS имеется три типа переменных:

- 1) арифметические переменные;
- 2) арифметические переменные с «плавающей точкой»;
- 3) булевы переменные.

Значение арифметических переменных может использоваться как:

1) операнд блока; в этом случае значение арифметической переменной может представлять собой:

- номер объекта (устройства, накопителя, очереди и т. п.);
- номер параметра транзакта;
- значение стандартного числового атрибута;

3) операнд <A> функции;

4) операнд <A> таблицы;

5) операнд выражения другой переменной.

В *выражениях* арифметические переменные используют следующие арифметические операции:

+ алгебраическое сложение;

– алгебраическое вычитание;

алгебраическое умножение (привычно используемый для умножения во многих языках знак «*», зарезервирован в GPSS для обозначения косвенной адресации);

/ алгебраическое деление (результатом операции является целая часть

частного);

@ деление по модулю;

^ возведение в степень;

\ деление без остатка (перед делением у обоих операндов отбрасываются дробные части, результатом операции является целая часть частного).

В выражениях может быть задано любое число приведенных операций в различных комбинациях. Знак результата вычисляется по обычным алгебраическим правилам. Допускаются отрицательные значения переменных. Выражения анализируются слева направо. Возведение в степень, умножение, деление и деление по модулю выполняются раньше, чем сложение и вычитание.

Вычисленное значение переменной является ее стандартным числовым атрибутом.

2.4.2.1 Арифметические переменные

Арифметические переменные аналогичны арифметическим выражениям в алгоритмических языках.

Арифметическая переменная с фиксированной точкой задается оператором VARIABLE, называемым оператором описания переменной, который содержит арифметическое выражение.

Формат оператора VARIABLE:

<имя> VARIABLE <A>

В поле операнда <A> записывается выражение, которое используется для вычисления значения переменной.

При обращении к переменной используется СЧА V<номер переменной> или V\$<имя переменной>.

ПРИМЕР 5.

Оператор описания VARIABLE определяет арифметическую переменную RSL:

RSL VARIABLE QT\$WAITL+3-FN\$DSTRB#P7

При любом обращении к переменной RSL (т.е. к СЧА V\$RSL) ее значение вычисляется как текущая длина очереди WAITL (QT\$WAITL – СЧА регистратора очереди) плюс константа 3 и минус произведение значения функции DSTRB (FN\$DSTRB – СЧА функции) на значение параметра 7-го транзакта, обрабатываемого в данный момент (Pj – СЧА транзакта).

Перед выполнением любой арифметической операции определяется значение каждого операнда и выделяется его целая часть. Константы без знака

считаются положительными числами. Пробелы между символами в выражениях не допускаются.

В выражении, заданном с помощью арифметической переменной, могут быть использованы любые СЧА, функции и другие арифметические переменные. Запрещается использование самой вычисляемой переменной, а также переменных со знаком, т.к. знаки в данном случае рассматриваются как арифметические операции.

Система моделирования GPSS допускает использование скобок в выражениях, задающих арифметические переменные (для группировки членов или для обозначения операции умножения).

В GPSS World выражения, записанные в круглых скобках, обрабатываются вычислительной процедурой встроенного алгоритмического языка PLUS. Поэтому их можно использовать в качестве операндов блоков и операторов языка GPSS. Например, выражение, описанное в примере 5, может быть использовано таким образом:

ADVANCE (QT\$WAITL+3-FN\$DSTRB#P7)

Арифметические переменные с плавающей точкой аналогичны рассмотренным арифметическим переменным, за исключением того, что все операции над операндами выражений переменных с плавающей точкой выполняются без преобразования операндов и промежуточных результатов в целые значения. Лишь окончательный результат вычисления преобразуется в целое число.

Формат операторов описания арифметических переменных с плавающей точкой идентичен рассмотренному выше формату операндов описания арифметических переменных за исключением того, что в поле операции записывается слово FVARIABLE. Правила написания операторов те же, что и для арифметических переменных. Арифметическая переменная и переменная с плавающей точкой не могут иметь одинаковые номера. Если они имеют одинаковые номера, то при вычислении используется более позднее из двух описаний.

Различие результатов, полученных при вычислении арифметических переменных с плавающей точкой и фиксированной, демонстрирует пример 6.

ПРИМЕР 6.

FLOAT FVARIABLE 10#(11/3)

FIXED VARIABLE 10#(11/3)

Значение переменной FLOAT равно 36, так как константа 10 умножается на 3,67 и от результата 36,7 взята целая часть. Переменная FIXED равна 30, так как результат промежуточной операции деления округляется до 3.

Для переменных с плавающей точкой не допускается операция деления по

модулю.

Использование дробных констант допускается только при описании переменных с плавающей точкой.

Стандартный числовой атрибут V\$<имя переменной> используется для обращения к значениям как арифметических переменных, так и переменных с плавающей точкой. Способ вычисления переменной определяется оператором описания этой переменной.

2.4.2.2 Булевы переменные

Булевы переменные позволяют принимать решения в зависимости от значений СЧА и состояния объектов GPSS, используя для этого только одно выражение.

Булевы переменные – это логические выражения, состоящие из различных СЧА и (или) других булевых переменных. В булевой переменной проверяется одно или несколько логических условий. Результатом проверки является единица (истина), если условия выполняются, и ноль (ложь) – в противном случае.

Стандартный числовой атрибут BV\$<имя переменной> используется для обращения к значениям булевых переменных.

При описании булевых переменных используются три типа операторов: логические, булевы и операторы отношения (условные операторы).

Логические операторы связаны с такими ресурсами, как устройства, накопители и логические ключи. Они используются для определения состояния данных объектов. Логические операторы, используемые в GPSS, представлены в таблице 5

Таблица 5 – Логические операторы, используемые в GPSS

Логические операторы	Значение оператора, отражающее состояние ресурса
FUj или Fj	Равно 1, если устройство j занято или обслуживает прерывание, иначе – 0
FNUj	Равно 1, если устройство j не занято и не обслуживает прерывание, иначе – 0
Ij	Равно 1, если устройство j обслуживает прерывание, иначе – 0
NIj	Равно 1, если устройство j не обслуживает прерывание, иначе – 0
NUj	Равно 1, если устройство j не используется, в противном

	случае – 0
UJ	Равно 1, если устройство j используется, иначе – 0
SFJ	Равно 1, если накопитель j заполнен, иначе – 0
SNFj	Равно 1, если накопитель j не заполнен, иначе – 0
SEj	Равно 1, если накопитель j пуст, иначе – 0
SNEj	Равно 1, если накопитель j не пуст, иначе – 0
SVj	Равно 1, если накопитель j находится в состоянии использования, иначе – 0
SNVJ	Равно 1, если накопитель j не используется, иначе – 0
LRj	Равно 1, если логический ключ j выключен, иначе – 0
LSj	Равно 1, если логический ключ j включен, иначе – 0

ПРИМЕР 7.

VAR1 BARIABLE FNU\$CHAN

Переменная VAR1 принимает значение «истина», если устройство с именем CHAN не занято и не обслуживает прерывание.

Условные операторы (операторы отношения) выполняют алгебраическое сравнение операндов. Операндами могут быть константы или стандартные числовые атрибуты. Операторы отношения записываются в кавычках:

‘G’ (Greater) – больше;

‘L’ (Less) – меньше;

‘E’ (Equal) – равно;

‘NE’ (Not Equal) – не равно;

‘LE’ (Less than or Equal) – меньше или равно;

‘GE’ (Greater than or Equal) – больше или равно.

ПРИМЕР 8.

VAR2 BARIABLE V\$FIXED’G’10

Переменная VAR2 принимает значение «истина», если значение переменной с именем FIXED больше 10.

Булевы операторы: ‘OR’ – оператор «или», и ‘AND’ – оператор «и». Оператор «или» проверяет, выполняется ли хотя бы одно из проверяемых условий. Оператор «и» требует выполнения обоих условий. В поле операндов используются комбинации логических, условных и булевых операторов.

ПРИМЕР 9.

VAR3 BARIABLE (V\$FIXED’G’10)’AND’ FNU\$CHAN

Переменная VAR3 принимает значение «истина», если значение переменной с именем FIXED больше 10 и одновременно устройство с именем

CHAN не занято и не обслуживает прерывание.

2.4.3 Функции

В GPSS рассматриваются пять типов функций:

- 1) дискретная числовая (D),
- 2) непрерывная числовая (C),
- 3) табличная числовая (L),
- 4) дискретная атрибутивная (E),
- 5) табличная атрибутивная (M).

Дискретная функция представляет собой кусочно-постоянную функцию, которая состоит из горизонтальных ступеней (Рисунок 3). *Непрерывная функция* представляет собой кусочно-непрерывную функцию. Непрерывная функция в GPSS состоит из соединенных между собой прямых отрезков и представляет собой ломаную линию (рисунок 4). Чтобы задать дискретную функцию, необходимо задать координаты крайних правых точек горизонтальных отрезков. Для непрерывной функции необходимо задать координаты всех точек, которые являются концами отрезков.

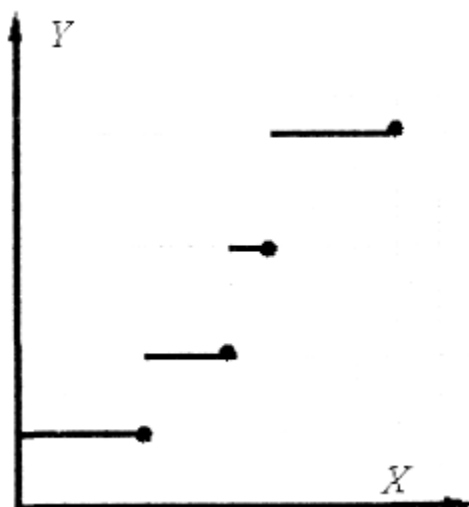


Рисунок 3 – Дискретная функция

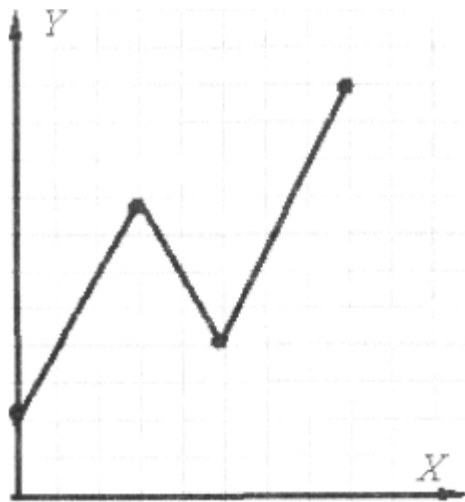


Рисунок 4 – Непрерывная функция

Действия, необходимые для определения GPSS-функции:

1. Присвоить функции имя. Имя может быть числовым либо символьным.
2. Задать аргумент функции. Аргументом могут быть:
 - ссылка на генератор случайных чисел, используемый для розыгрыша в соответствии с распределением, заданным функцией;
 - стандартный числовой атрибут;
 - ссылка на любую другую функцию.
3. Задать тип функции и число пар аргумент/значение функции.
4. Задать значения пар аргумент/значение функции.

Для задания GPSS-функции используются два оператора:

- оператор определения функции,
- оператор описания координат функции.

Формат оператора определения функции:

<имя> FUNCTION <A>,

Поля операндов имеют следующий смысл:

- <A> – датчик случайных чисел или СЧА,
- – Dn либо Cn, где D определяет дискретную функцию, C определяет непрерывную функцию; n – для дискретной функции – число различных значений, получаемых функцией (количество горизонтальных отрезков), для непрерывной функции – число, на единицу большее числа отрезков, составляющих функцию (количество точек).

Формат оператора описания координат функции:

X1, Y1/X2, Y2/.../Xn, Yn

где X_i и Y_i – координаты i -й точки функции (в случае моделирования случайной величины X_i является i -й кумулятивной частотой, Y_i – соответствующим значением случайной величины).

Особенности оператора описания координат функции:

- основной единицей информации оператора описания координат функции является пара значений X_i, Y_i (координаты точки i);
- значения координат X_i и Y_i – одной точки функции разделяются запятой;
- последовательные наборы координат разделяются знаком «/»;
- координаты X_i и Y_i – относящиеся к одной точке, задаются одним оператором, т.е. пара координат одной точки не должна разрываться;
- все строки описания координат функции должны начинаться с первой позиции;
- во всех случаях значения аргумента должны удовлетворять следующим неравенствам: $X_1 < X_2 < \dots < X_i < \dots < X_n$.

Значение функции является ее стандартным числовым атрибутом. Способ ссылки на этот атрибут зависит от того, как задано имя функции: в символьном или числовом виде. Если имя числовое, то к значению функции можно обратиться через FN_j (где j – номер функции), если имя символьное – через $FN\$<имя функции>$.

ПРИМЕР 10.

Необходимо смоделировать дискретную случайную величину, заданную следующей таблицей (Таблица 6).

Таблица 6 – Дискретная случайная величина

Интервал	Относительная частота	Кумулятивная частота	Диапазон	Значение случайной переменной
1	0.20	0.20	[0.0-0.20]	2
2	0.20	0.40]0.20-0.40]	5
3	0.20	0.60]0.40-0.60]	8
4	0.22	0.82]0.60-0.82]	9
5	0.18	1.00]0.82-1.00]	12

GPSS-функцию можно определить таким образом:

```
SERV      FUNCTION      RN4,D5
0.20,2/0.40,5/0.60,8/0.82,9/1,12
```

Графическая интерпретация функции показана на рисунке 5.

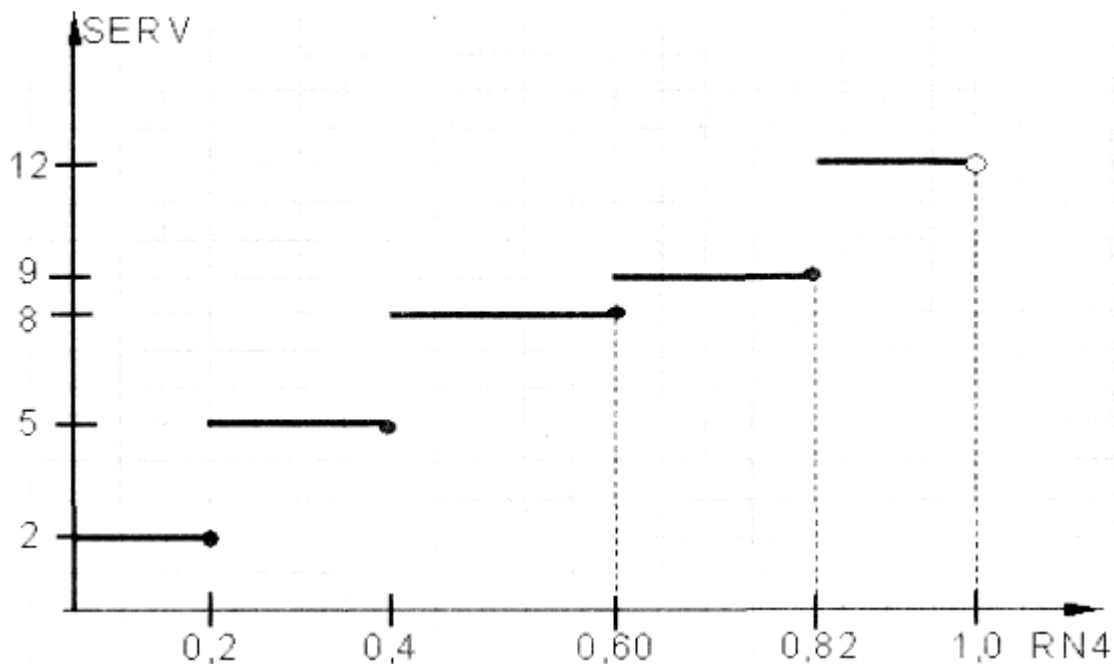


Рисунок 5 – Графическая интерпретация дискретной функции

Особенности вычисления дискретных и непрерывных GPSS-функций:

1. В начальной фазе выполняемые действия при вычислении дискретной и непрерывной функции одинаковы. При обращении к функции определяется значение ее аргумента. Потом просматривается упорядоченный ряд значений $X_1 < X_2 < \dots < X_i < \dots < X_n$ для определения интервала, в который попало значение аргумента (пусть это будет интервал между точками $i-1$ и i).

2. Если функция дискретная, то второй элемент соответствующей пары X_i, Y_j является значением функции. Если функция непрерывная, выполняется линейная интерполяция для пары точек $i-1$ и i , находящихся на концах интервала значений функции, в который попало значение аргумента. Целая часть результата интерполяции и является значением функции.

3. Если значение аргумента функции больше значения координаты X_n последней точки, то в обоих случаях (дискретной и непрерывной функции) значениями функции являются значения Y_n .

Табличная числовая функция определяется так, что аргументы функции образуют непрерывный ряд натуральных чисел. Аргумент функции используется для прямого обращения к массиву значений функции.

ПРИМЕР 11.

```
FUN1      FUNCTION      P1,L5
1,10/2,12/3,5/4,20/5,25
```

Дискретная атрибутивная функция подобна дискретной числовой функции с тем отличием, что значением функции может быть любой СЧА.

ПРИМЕР 12.

FUN2 FUNCTION X\$CAR,E4
21,V\$RED/25,X35/30,10/36,S\$RED

Значение функции FUN2 вычисляется по правилу вычисления значения дискретной числовой функции:

X\$CAR<=21	FN\$FUN2=V\$RED
22<=X\$CAR<=25	FN\$FUN2=X35
26<=X\$CAR<=30	FN\$FUN2=10
31<=X\$CAR<=36	FN\$FUN2= S\$RED

Табличная атрибутивная функция подобна табличной числовой функции с тем отличием, что значением функции может быть любой СЧА.

ПРИМЕР 13.

FUN3 FUNCTION P1,M5
1,V1/2,P12/3,X35/4,Q\$QUICK/5,99

2.5 Группа ссылочных элементов

Группу ссылочных элементов GPSS образуют ячейки сохраняемых величин и матрицы ячеек.

2.5.1 Ячейки сохраняемых величин

В GPSS пользователю предоставляется возможность определить глобальные переменные, начальные значения которых могут быть заданы перед моделированием и к которым можно обратиться из любого места модели в любой момент времени. Эти переменные называют *сохраняемыми величинами (ячейками)*. Совокупность логически связанных между собой ячеек образует *матрицу* (аналог массива).

В отличие от параметров транзакта, которые теряются в момент выхода транзакта из модели, ячейки доступны на протяжении всего процесса моделирования и после его завершения. Значения сохраняемых величин не подсчитываются интерпретатором автоматически (как СЧА устройств, очередей, накопителей и т.п.), а задаются и изменяются программистом.

Сохраняемые величины могут принимать положительные и

отрицательные значения. Стандартный числовой атрибут Xj (X\$<имя ячейки>) дает значение соответствующей сохраняемой величины. Например, X2 – значение ячейки 2; X\$DAY – значение ячейки DAY.

Если в процессе моделирования происходит обращение к сохраняемой величине, которая не была задана, то интерпретатор выдает ошибку в процессе выполнения программы. Поэтому перед началом моделирования все сохраняемые величины должны быть инициализированы с помощью оператора INITIAL.

Формат оператора инициализации ячейки:

INITIAL <A>,

Поля операндов имеют следующий смысл:

- <A> – имя ячейки,
- – начальное значение.

ПРИМЕР 14.

INITIAL X\$TIMER,100000

Значение сохраняемой величины изменяется при входе транзакта в блок SAVEVALUE (СОХРАНИТЬ ВЕЛИЧИНУ).

Формат блока SAVEVALUE:

SAVEVALUE <A>[±],

Поля операндов имеют следующий смысл:

- <A> – имя ячейки,
- – величина, используемая для модификации.

Блок SAVEVALUE может быть использован как в режиме замещения величины, так и в режиме увеличения или уменьшения. В режиме увеличения предыдущее значение сохраняемой величины увеличивается на значение, заданное операндом , а в режиме уменьшения – уменьшается на это значение. Режимы увеличения и уменьшения определяются введением соответственно знака «плюс» или «минус» перед запятой, разделяющей операнды <A> и .

ПРИМЕР 15.

SAVEVALUE 5+,100

При входе транзакта в блок величина ячейки номер 5 (X5) увеличивается на 100.

SAVEVALUE PROFIT-,FN\$COSTS

При входе транзакта в блок SAVEVALUE величина ячейки PROFIT (X\$PROFIT) уменьшается на значение функции FN\$COSTS.

SAVEVALUE P5,V\$ALPHA

При входе транзакта в блок SAVEVALUE значение переменной ALPHA записывается в ячейку, номер которой записан в 5-м параметре транзакта.

2.5.2 Матрицы ячеек сохраняемых величин

С матрицами связан стандартный числовой атрибут $MX_j(m, n)$ – значение, записанное в строке m и в столбце n матрицы j или $MX\langle\text{имя матрицы}\rangle(m, n)$, если матрица имеет символьное имя.

Оператор описания матрицы MATRIX. В GPSS World максимальная размерность матрицы равна 6. Каждая матрица должна быть объявлена до ее использования, т.е. должна иметь оператор описания MATRIX.

Формат оператора MATRIX:

$\langle\text{имя}\rangle \text{ MATRIX } \langle B \rangle, \langle C \rangle, \langle D \rangle, \langle E \rangle, \langle F \rangle, \langle G \rangle$

Поля операндов имеют следующий смысл:

- $\langle A \rangle$ – не используется,
- $\langle B \rangle$ – максимальное значение индекса для первой размерности,
- $\langle C \rangle$ – максимальное значение индекса для второй размерности,
- $\langle D \rangle$ – максимальное значение индекса для третьей размерности,
- $\langle E \rangle$ – максимальное значение индекса для четвертой размерности,
- $\langle F \rangle$ – максимальное значение индекса для пятой размерности,
- $\langle G \rangle$ – максимальное значение индекса для шестой размерности.

ПРИМЕР 16.

MATR MATRIX ,10,5

Блок MSAVEVALUE используется для записи значений в матрицы, а также для увеличения или уменьшения значений элементов матриц.

Формат блока MSAVEVALUE:

MSAVEVALUE $\langle A \rangle [\pm], \langle B \rangle, \langle C \rangle, \langle D \rangle$

Поля операндов имеют следующий смысл:

- $\langle A \rangle$ – имя матрицы,
- $\langle B \rangle$ – имя строки матрицы,
- $\langle C \rangle$ – имя столбца матрицы,
- $\langle D \rangle$ – величина, используемая для модификации.

Подобно блоку SAVEVALUE блок MSAVEVALUE может быть использован как в режиме замещения величины, так и в режиме увеличения или уменьшения.

Когда транзакт входит в блок MSAVEVALUE, анализируется операнд <A> и выполняется поиск матрицы с указанным именем. Если матрица не найдена, возникает ошибка. Соответствующий элемент матрицы определяется содержимым операндов и <C>. Если такого элемента не существует, то также возникает ошибка.

ПРИМЕР 17.

MSAVEVALUE MATR-,8,4,FN\$COSTS

При входе транзакта в блок MSAVEVALUE величина ячейки матрицы MATR на пересечении 8-й строки и 4-го столбца (MX\$MATR(8,4)) уменьшается на значение функции FN\$COSTS.

3 ТРАНЗАКТНО-ОРИЕНТИРОВАННЫЕ БЛОКИ GPSS

К этой группе относятся блоки, которые определяют действия над транзактами:

- 1) работа с параметрами транзакта;
- 2) установка приоритета транзакта;
- 3) изменение направления движения транзакта;
- 4) организация циклов;
- 5) обработка транзактов, принадлежащих одному семейству.

3.1 Работа с параметрами транзакта

3.1.1 Установка значений параметров транзакта. Блок ASSIGN

При входе транзакта в блок ASSIGN (НАЗНАЧИТЬ) значения параметров могут задаваться или изменяться.

Формат блока ASSIGN:

ASSIGN <A>[±],,[<C>]

Поля операндов имеют следующий смысл:

- <A> – номер или имя модифицируемого или задаваемого параметра,

- – величина, используемая для модификации (число или СЧА),
- <C> – имя функции.

Блок ASSIGN может быть использован как в режиме замещения значения параметра (начальное значение всех параметров транзактов равно 0), так и в режиме увеличения и уменьшения. В режиме увеличения предшествующее значение параметра увеличивается на значение, стоящее в операнде . В режиме уменьшения оно уменьшается на величину, стоящую в операнде . Режимы увеличения и уменьшения определяются введением соответственно знаков «плюс» и «минус» перед запятой, которая разделяет операнды <A> и .

При использовании операнда <C> значение операнда умножается на значение функции, указанной в операнде <C>. Параметр, заданный в операнде <A>, изменяется на величину полученного произведения (в режиме увеличения и уменьшения) или приобретает значение результата (в режиме замещения).

ПРИМЕР 18.

ASSIGN 3,25

Параметру P3 присваивается значение 25.

ASSIGN P4,FR\$BB

Параметру транзакта с номером, записанным в параметре P4, присваивается значение величины загрузки устройства BB (оба операнда заданы косвенным образом).

Блок ASSIGN в режимах накопления и уменьшения:

ASSIGN 4+,Q5

Параметр 4 увеличивается на значение, равное текущей длине очереди 5.

ASSIGN P2-,7

От значения параметра, номер которого задан параметром P2, вычитается

7.

3.1.2 Отметка времени транзакта. Блок MARK

При каждом входе транзакта в модель интерпретатор фиксирует для него текущее значение времени. Это значение времени называется *отметкой времени*. Она может быть интерпретирована как время «рождения» транзакта или время входа транзакта в модель. В явном виде отметка времени недоступна. Однако существует СЧА, связанный со значением времени входа транзакта в модель. Его имя M1, а значение определяется так:

$$M1 = \left(\begin{array}{c} \text{Текущее значение} \\ \text{таймера абсолютного} \\ \text{времени} \end{array} \right) - \left(\begin{array}{c} \text{Значение времени} \\ \text{входа транзакта} \\ \text{в модель} \end{array} \right)$$

Значение M1 для каждого транзакта изменяется в процессе моделирования. Сразу после входа транзакта в модель M1=0, через 10 единиц модельного времени M1=10 и т.д.

Стандартный числовой атрибут M1 измеряет время, которое прошло с момента входа транзакта в модель. Однако очень часто требуется знать время, затраченное на перемещение транзакта между двумя произвольными точками модели (т.е. транзитное время). Для этого используется блок MARK. При входе транзакта в блок MARK значение таймера абсолютного времени записывается в качестве одного из его параметров. Такую запись называют отметкой транзакта.

Формат блока MARK:

MARK <A>

Поля операндов имеют следующий смысл:

- <A> – номер параметра, в который записывается значение абсолютного времени (целое число или СЧА).

Пусть необходимо определить интервал времени, на протяжении которого транзакт проходит от точки T1 к точке T2. Для этого нужно выполнить два действия:

- 1) в точку T1 поместить блок MARK j, где j – номер параметра, в который записывается значение абсолютного времени в момент записи;
- 2) в точке T2 обратиться к СЧА с именем MPj, где j – номер параметра, в котором сделана отметка времени транзакта; СЧА MPj будет иметь такое значение:

$$MPj = \left(\begin{array}{c} \text{Текущее значение} \\ \text{таймера абсолютного} \\ \text{времени} \end{array} \right) - \left(\begin{array}{c} \text{Значение} \\ j\text{-го} \\ \text{параметра} \end{array} \right)$$

3.2 Установка приоритета транзакта. Блок PRIORITY

Блок PRIORITY (НАЗНАЧИТЬ ПРИОРИТЕТ) присваивает или изменяет приоритет транзакта, если он был задан блоком GENERATE (по умолчанию приоритет транзакта равен нулю).

Формат блока PRIORITY
PRIORITY <A>

Поле операндов <A> задает новое значение приоритета (число или СЧА).

Новое значение приоритета может быть меньше, больше или равно текущему значению приоритета транзакта. Приоритет влияет на порядок выбора транзакта для обслуживания устройствами и на порядок просмотра транзактов в списке текущих событий.

Стандартный числовой атрибут этого блока – PR. Поскольку уровень приоритета транзакта может изменяться от 0 до 127, то PR будет выдавать значение в диапазоне 0-127.

ПРИМЕР 19.

PRIORITY 100

Вошедшему в этот блок транзакту присваивается приоритет 100.

DELAY FUNCTION PR,D3

1,4/2,7/3,10

...

ADVANCE FN\$DELAY

...

Задержка в блоке ADVANCE зависит от приоритета транзакта. Транзакт с наиболее низким приоритетом (1) задерживается на 4 единицы модельного времени, транзакт с наиболее высоким приоритетом (3) задерживается на 10 единиц модельного времени.

3.3 Изменение направления движения транзакта

Направление движения транзактов в программе изменяют блоки TRANSFER, GATE, TEST.

3.3.1 Переход транзакта в блок, отличный от последующего. Блок TRANSFER

В GPSS блок TRANSFER (ПЕРЕДАТЬ) может быть использован в девяти

разных режимах. Рассмотрим три основных режима.

Режим безусловной передачи.

Формат блока TRANSFER в режиме безусловной передачи:

TRANSFER ,B

Поля операндов имеют следующий смысл:

- <A> – не используется,
- – позиция блока, в который должен перейти транзакт.

Позиция блока – это номер или метка блока. Так как операнд <A> не используется, то перед операндом должна стоять запятая. В режиме безусловной передачи блок TRANSFER не может отказывать транзакту во входе. Кстати, если транзакт входит в блок, то он сразу же пытается войти в блок .

Статистический режим. В этом режиме осуществляется передача транзакта в один из двух блоков случайным образом.

Формат блока TRANSFER в режиме статистической передачи:

TRANSFER <A>,[],<C>

Поля операндов имеют следующий смысл:

- <A> – вероятность передачи транзакта в блок <C>, задаваемая в долях тысячи,
- – позиция блока, в который должен перейти транзакт (с вероятностью 1-<A>),
- <C> – позиция блока, в который должен перейти транзакт (с вероятностью <A>).

При задании вероятности (операнд <A>) используется не более трех цифр, первый символ записи частоты «.» (десятичная точка), если используется действительное число, которое должно быть в пределах от 0 до 1,0 (например, 0,235).

ПРИМЕР 20.

TRANSFER .333,MET1,MET2

...

MET1 SEIZE PR1

...

MET2 SEIZE PR2

С вероятностью .333 транзакт переходит в блок с меткой MET2, а с

вероятностью .667 – в блок с меткой MET1.

Если с вероятностью .667 транзакт должен перейти к следующему блоку, в блоке TRANSFER можно опустить операнд .

	TRANSFER	.333,,MET2
MET1	SEIZE	PR1
...		
MET2	SEIZE	PR2

Режим BOTH. Если в операнде <A> указано зарезервированное слово BOTH, блок TRANSFER работает в режиме BOTH.

В этом режиме входящий транзакт сначала пытается перейти к блоку, указанному в операнде . Если это сделать не удастся, транзакт пытается перейти в блок, указанный в операнде <C>. Если транзакт не сможет перейти ни к тому, ни к другому блоку, то он остается в блоке TRANSFER и при каждом просмотре списка текущих событий будет в том же порядке повторять попытки перехода до тех пор, пока не сможет выйти из блока TRANSFER.

ПРИМЕР 21.

	TRANSFER	BOTH,MET1,MET2
...		
MET1	SEIZE	PR1
...		
MET2	SEIZE	PR2

Транзакт сначала пытается перейти в блок с меткой MET1. Если устройство PRI1 занято, транзакт пытается войти в блок с меткой MET2. Если транзакт не может войти и в этот блок (устройство PRI2 также занято), он остается в списке текущих событий и повторяет эти попытки при каждом просмотре списка до тех пор, пока не выйдет из блока TRANSFER.

3.3.2 Изменение направления движения транзакта в зависимости от состояния оборудования. Блок GATE

Блок GATE управляет потоком транзактов с помощью логических операторов. Блок GATE, как и блок TEST, не изменяет никаких атрибутов транзактов. Он определяет номер следующего блока, к которому должен перейти транзакт из блока GATE. Блок GATE может задержать транзакт на входе, если не задан альтернативный выход.

Формат блока GATE:

GATE <X> <A>,[]

Поля операндов имеют следующий смысл:

- <A> – имя или номер объекта, для которого производится проверка, операнд <A> может быть именем, положительным целым числом или СЧА,
- – номер следующего блока для входящего транзакта, если логический оператор имеет значение «ложь». Операнд может быть именем, положительным целым числом или СЧА. Если операнд определен, то он должен содержать номер блока, допустимый для текущей модели.

В дополнительном операторе <X> задается один из следующих логических операторов:

1. Логические операторы, связанные с устройствами:

NU – устройство j, заданное в операнде <A>, свободно;

U – устройство j, заданное в операнде <A>, занято (в результате выполнения транзактом блока SEIZE или PREEMPT);

NI – устройство j, заданное в операнде <A>, не прервано;

I – устройство j, заданное в операнде <A>, обслуживает прерывания;

FV – устройство j, заданное в операнде <A>, доступно;

FNV – устройство j, заданное в операнде <A>, не доступно.

2. Логические операторы, связанные с многоканальными устройствами:

SE – накопитель j, заданный в операнде <A>, пуст ($S[j]=0$);

SNE – накопитель j, заданный в операнде <A>, не пуст ($S[j] \neq 0$);

SF – накопитель j, заданный в операнде <A>, заполнен ($R[j]=0$);

SNF – накопитель j, заданный в операнде <A>, не заполнен ($R[j] \neq 0$);

SV – накопитель j, заданный в операнде <A>, доступен;

SNV – накопитель j, заданный в операнде <A>, не доступен.

3. Логические операторы, связанные с транзактами:

M – в блоке j, заданном в операнде <A> блока GATE, находится в состоянии синхронизации транзакт, принадлежащий тому же семейству, что и транзакт, который находится в блоке GATE или пытается войти в этот блок;

NM – в блоке j, заданном в операнде <A> блока GATE, в состоянии синхронизации нет ни одного транзакта, принадлежащего тому же семейству, что и транзакт, который пытается войти в блок GATE.

4. Логические операторы, связанные с логическими ключами:

LS – логический ключ j, заданный в операнде <A>, включен;

LR – логический ключ j, заданный в операнде <A>, выключен.

Блок GATE может работать в режиме отказа и условного перехода.

В *режиме условного перехода*, если заданный логический оператор имеет значение «истина», транзакт пытается перейти к следующему по номеру блоку. Если логический оператор имеет значение «ложь», то транзакт будет пытаться перейти к блоку, номер которого задан в операнде блока GATE. Выбор

следующего блока производится один раз в момент вхождения транзакта в блок GATE.

В *режиме отказа*, если операнд блока GATE пустой (альтернативный выход не задан), транзакты не смогут войти в блок GATE до тех пор, пока указанный в этом блоке логический оператор не будет иметь значение «истина». Интерпретатор не проверяет значение логических операторов, за исключением операторов M и NM. В режиме условного перехода задержанные транзакты находятся в списках задержки и, таким образом, исключаются из числа транзактов, обрабатываемых интерпретатором до тех пор, пока соответствующий логический оператор не примет значение «истина».

ПРИМЕР 22.

```
QUEUE    CHAN
GATE SNF MEM
DEPART    CHAN
ENTER     MEM
```

В данном примере транзакт помещается в список задержки, если многоканальное устройство MEM заполнено в тот момент, когда транзакт пытается войти в блок GATE. Когда накопитель MEM становится незаполненным, все транзакты выводятся из списка и делают попытку занять накопитель MEM.

3.3.3 Изменение направления движения транзакта в зависимости от выполнения логических условий, определенных на множестве СЧА. Блок TEST

Формат блока TEST:
TEST <X> <A>,,[<C>]

Поля операндов имеют следующий смысл:

- <A> – СЧА-левый операнд проверяемого отношения,
- – СЧА-правый операнд проверяемого отношения,
- <C> – имя блока, в который переходит транзакт, если проверяемое отношение имеет значение «ложь».

В дополнительном операторе <X> задается один из следующих операторов отношения (операторы отношения записываются без кавычек):

G (Greater) – больше;
L (Less) – меньше;
E (Equal) – равно;

NE (Not Equal) – не равно;
LE (Less than or Equal) – меньше или равно;
GE (Greater than or Equal) – больше или равно.

ПРИМЕР 23.

; режим отказа

TEST LE Q1,Q2

Проверяющий транзакт будет задержан в предыдущем блоке до тех пор, пока длина первой очереди не станет меньше или равна длине второй очереди.

; режим условного перехода

TEST LE Q1,Q2,MET

Проверяющий транзакт перейдет в следующий по порядку блок, если содержимое первой очереди меньше или равно содержимому второй очереди. Если это условие не выполняется, транзакт перейдет в блок с меткой MET.

3.4 Организация циклов. Блок LOOP

С помощью параметров транзактов в программе можно организовать циклы. Для этого используется блок LOOP. Он управляет количеством повторных прохождений транзактом определенной последовательности блоков модели.

Формат блока LOOP:

LOOP <A>,[]

Поля операндов имеют следующий смысл:

- <A> – параметр транзакта, используемый для организации цикла (переменная цикла). Он может быть именем, положительным целым числом, СЧА,
- – метка (имя блока) начального блока цикла.

Когда транзакт входит в блок LOOP, параметр, указанный в операнде <A>, уменьшается на единицу, а затем его значение проверяется на равенство нулю. Если значение не равно нулю, то транзакт переходит в блок, указанный в операнде . Если значение параметра равно нулю, транзакт переходит в следующий блок.

ПРИМЕР 24.

ASSIGN 1,3


```
MET SEIZE      CHAN
...
RELEASE CHAN
LOOP          1,MET
```

Цикл организован по первому параметру транзакта. Его начальное значение равно 3. После освобождения устройства проверяется значение первого параметра. Если оно не равно нулю, то транзакт возвращается к блоку, помеченному меткой MET, т.е. занимает устройство с именем CHAN. Всего каждый транзакт будет занимать это устройство три раза.

3.5 Обработка транзактов, принадлежащих одному семейству

К этой группе относятся блоки, выполняющие следующие действия:

- создание копий транзактов,
- синхронизация движения транзактов.

3.5.1 Создание копий транзактов. Блок SPLIT

Кроме блока GENERATE, для создания транзактов может использоваться блок SPLIT (РАЗДЕЛИТЬ), который выполняет функцию копирования транзакта, входящего в него. Этот транзакт называется начальным или порождающим. Все копии формируются в момент входа начального транзакта в блок SPLIT. Каждая новая копия становится членом семейства (ансамбля) транзактов, порожденных одним начальным транзактом, который был создан блоком GENERATE.

Формат блока SPLIT:
SPLIT <A>[,][,<C>]

Поля операндов имеют следующий смысл:

- <A> – число создаваемых копий транзакта,
- – метка блока, к которому направляются копии,
- <C> – параметр, в котором запоминаются номера копий транзактов.

Операнд <A> может быть положительным целым, СЧА. Если вычисленное значение операнда <A> равно нулю, блок SPLIT не выполняет никаких операций. После создания копий начальный транзакт пытается перейти к очередному блоку.

Операнд <В> задает блок, в который переходят копии начального транзакта. Операнд может быть именем (меткой), положительным целым, СЧА (в двух последних случаях операнд <В> задает номер блока). Значение операнда <В> вычисляется для каждой копии отдельно.

Операнд <С> задает параметр транзакта, который используется для присвоения копиям последовательных номеров. Операнд <С> может быть именем, положительным целым, СЧА.

Транзакты, принадлежащие одному семейству, объединяются интерпретатором в список. По связям внутри семейства транзактов невозможно установить, какой из транзактов семейства является начальным. Если копия транзакта входит в блок SPLIT, то повторная копия становится членом того же семейства, что и первичная копия. Таким образом, каждый транзакт является членом одного и только одного семейства. Семейство может состоять из произвольного числа транзактов. Когда транзакт уничтожается, интерпретатор автоматически исключает его из членов соответствующего семейства. Таким образом, семейство существует до тех пор, пока из модели не удалится последний из его членов.

В модели одновременно может присутствовать произвольное число семейств, оно все время меняется, поскольку каждый транзакт, генерируемый блоком GENERATE, может создать свое семейство.

ПРИМЕР 25.

```
SPLIT      1,MET
ADVANCE 10
...
MET SEIZE   CHAN
...
```

Основной транзакт, порождающий копию, переходит в блок ADVANCE, а транзакт-копия переходит к блоку с меткой MET.

3.5.2 Синхронизация движения транзактов. Блоки MATCH, ASSEMBLE, GATHER

Для синхронизации движения транзактов, принадлежащих одному семейству, используются блоки MATCH (СОГЛАСОВАТЬ), ASSEMBLE (СОБРАТЬ), GATHER (СОЕДИНИТЬ).

Блок MATCH синхронизирует движение транзактов с другим блоком MATCH.

Формат блока MATCH:

MATCH <A>

Операнд <A> указывает имя сопряженного блока. Сопряженным блоком является также блок MATCH.

ПРИМЕР 26.

В локальной сети рабочая станция опрашивается каждые 30 мс. Если на рабочей станции есть сообщение для передачи, то оно занимает канал.

```
MET1      MATCH  MET2
          SEIZE   CHAN
```

```
...
MET2      MATCH  MET1
          ADVANCE 30
```

При входе транзакта-сообщения в блок MATCH с меткой MET1 он будет ждать (в списке синхронизации) момента, когда другой опрашиваемый транзакт, принадлежащий тому же семейству, не войдет в сопряженный блок MATCH с меткой MET2. Только после этого сообщение займет канал CHAN, а опрашиваемый транзакт перейдет в блок ADVANCE.

Блок *ASSEMBLE* собирает начальный транзакт и все транзакты-копии из одного семейства, удаляет копии и выдает один начальный транзакт. После сборки из блока *ASSEMBLE* выходит только один транзакт, который переходит в следующий по номеру блок.

Формат блока *ASSEMBLE*:
ASSEMBLE <A>

Операнд <A> задает счетчик сборки, указывающий сколько членов одного семейства должны быть объединены. Операнд <A> может быть именем, положительным целым, СЧА. Первоначальное значение операнда <A> должно быть больше единицы.

Блок *GATHER* скапливает заданное количество транзактов, принадлежащих одному семейству. Он задерживает их до тех пор, пока не соберется необходимое число, указанное операндом <A>. Затем накопленные транзакты одновременно попытаются войти в следующий по номеру блок.

Формат блока *GATHER*:
GATHER <A>

Операнд <A> задает число транзактов, принадлежащих к одному семейству, которое нужно накопить. Операнд <A> может быть именем, положительным целым, СЧА.

4 УПРАВЛЕНИЕ ПРОЦЕССОМ МОДЕЛИРОВАНИЯ В GPSS WORLD

В системе GPSS интерпретатор (программа управления моделированием, монитор моделирования) поддерживает сложные структуры организации списков (рисунок 3). С целью уменьшения затрат компьютерного времени на просмотр списков система GPSS ведет два основных списка событий. Первым является *список текущих событий* (СТС), куда входят все события, запланированные на текущий момент модельного времени независимо от того, условные они или безусловные. Программа управления моделированием просматривает в первую очередь этот список и пытается переместить по модели те транзакты, для которых выполнены условия. Если в этом списке таких транзактов нет, то монитор моделирования обращается к другому списку – *списку будущих событий* (СБС). Монитор переносит все события, которые запланированы на ближайший момент модельного времени, из этого списка в СТС и повторяет его просмотр. Такой перенос осуществляется также в случае совпадения текущего времени моделирования со временем первого события в списке будущих событий. В СТС транзакты размещены в порядке уменьшения приоритета (то есть транзакты с более высоким приоритетом размещены ближе к началу списка). Транзакты с одинаковыми приоритетами размещаются в соответствии с последовательностью поступления в список. Каждый транзакт в СТС может находиться или в активном состоянии (то есть просматриваться в данный момент модельного времени), или в состоянии задержки.

В начальный момент (при выполнении оператора управления START, который начинает фазу интерпретации GPSS-модели) монитор моделирования обращается ко всем блокам GENERATE модели. Каждый из этих блоков планирует момент появления транзактов и заносит их в СБС, после чего монитор моделирования обращается к СТС. Так как в этом списке пока что отсутствуют транзакты, монитор просматривает СБС и выбирает из него все транзакты, запланированные на ближайший момент времени и переносит их в СТС, после чего пытается продвинуть первый транзакт этого списка по блокам модели. Если перемещение транзакта было задержано по какой-либо причине, не связанной с блоком ADVANCE, то он остается в СТС и монитор пробует перемещать такой транзакт из этого списка далее по блокам. Если транзакт вошел в блок ADVANCE, то планируется его выход из этого блока и транзакт переносится в СБС.

Если транзакты находятся в активном состоянии, то процедура просмотра пытается переместить их к следующим блокам. Если перемещение транзакта

блокируется каким-нибудь ресурсом ввиду его занятости, то вхождение в следующий блок невозможно и транзакт переводится в состояние задержки. Такие транзакты не просматриваются и размещаются в соответствующем списке задержки.

Если при обслуживании текущего активного транзакта произошло изменение состояния ресурса, пересмотр начинается сначала, и опять обслуживаются все транзакты из СТС, которые находятся в активном состоянии. Если изменение списков ресурсов не произошло, монитор моделирования опять обращается к СТС и проверяет, не остались ли в нем транзакты, которые необходимо обработать.

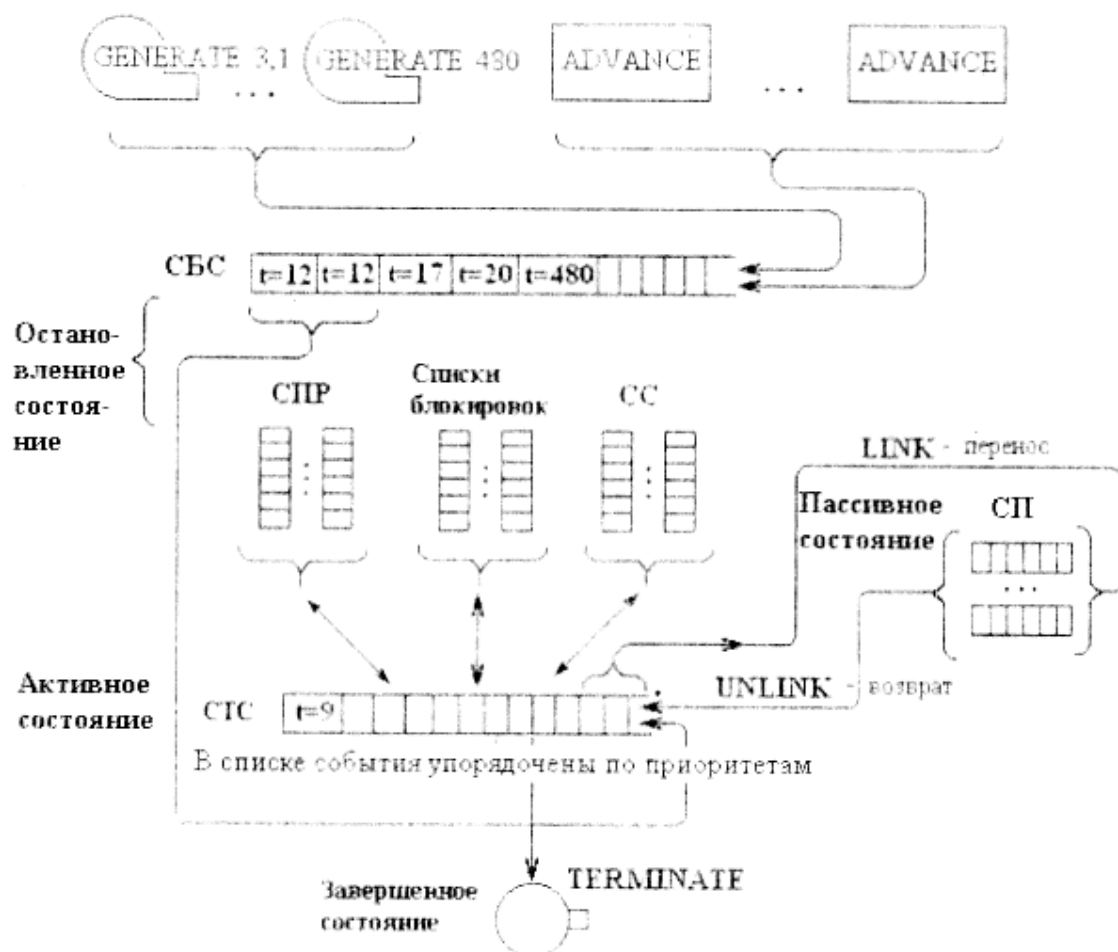


Рисунок 3 – Списки в GPSS

Список блокировок (СБ) – это список транзактов, которые ожидают изменения состояния ресурса. Существует шесть видов таких списков, связанных с устройствами; семь видов, связанных с многоканальными устройствами, и два вида, связанных с логическими ключами. С устройствами используются списки для занятых и незанятых, доступных и недоступных устройств и устройств, работающих без прерываний и с прерываниями. С

многоканальными устройствами используются списки для заполненного, незаполненного, пустого, непустого, доступного, недоступного многоканального устройства и транзактов, которые могут войти в многоканальное устройство. С логическими ключами связаны списки для включенных и выключенных ключей.

Список прерываний (СПР) содержит прерванные во время обслуживания транзакты, а также транзакты, вызвавшие прерывание. Список прерываний используется для организации обслуживания одноканальных устройств по абсолютным приоритетам. Это дает возможность организовать приоритетные дисциплины обслуживания транзактов в устройствах.

Список синхронизации (СС) содержит транзакты, которые на данный момент времени сравниваются. Этот список работает с транзактами, полученными с помощью блока SPLIT, который создает транзакты-копии, принадлежащие одному семейству или ансамблю. Синхронизацию движения транзактов одного семейства выполняют следующие блоки: MATCH (синхронизирует движение транзактов с другим блоком), ASSEMBLE (собирает все транзакты-копии и выдает один начальный транзакт), GATHER (собирает заданное количество транзактов и задерживает копии до тех пор, пока не соберется необходимое количество копий транзактов). Блок SPLIT можно использовать многократно.

Остановленные процессы находятся в СБС, СС и списках блокировок.

Список пользователя (СП) содержит транзакты, выведенные пользователем из СТС с помощью блока LINK и помещенные в СП как временно неактивные (переведенные пользователем в пассивное состояние). При работе монитора моделирования они ему недоступны до тех пор, пока не будут возвращены пользователем в СТС с помощью блока UNLINK.

Моделирование заканчивается тогда, когда счетчик завершения, инициализированный оператором управления START, будет сброшен в ноль или когда в СТС и СБС не будет ни одного транзакта.

4.1 Списки пользователя. Блоки LINK и UNLINK

Блок LINK (ВНЕСТИ В СПИСОК) собирает транзакты из СТС и помещает их в СП. Таким образом, интерпретатор их не просматривает и не перемещает по блокам модели до тех пор, пока пользователь не возвратит их в модель.

Формат блока LINK:

LINK <A>,,[<C>]

Поля операндов имеют следующий смысл:

- <A> – операнд <A> задает номер или имя СП, в который будет помещен транзакт. Операнд <A> может быть положительным целым, именем, СЧА;
- – задает алгоритм упорядочивания СП, операнд может быть LIFO, FIFO, целым, СЧА. Допустимые значения операнда :
 - FIFO – вошедший транзакт помещается в конец СП,
 - LIFO – вошедший транзакт помещается в начало СП,
 - номер параметра – входящие в СП транзакты располагаются в соответствии со значением указанного параметра,
 - PR – приоритет транзакта (транзакт помещается в список в соответствии с приоритетом),
 - M1 – время нахождения транзакта в модели (транзакт помещается в список в соответствии с временем нахождения транзакта в модели);
- <C> – параметр, в котором запоминаются номера копий транзактов.

Операнд <C> указывает альтернативный выход, который используется при описании разных ситуаций, возникающих в очередях. Операнд <C> может быть именем, положительным целым, СЧА.

Если операнд <C> не задан, индикатор, связанный с заданным СП, устанавливается в положение «1». Это приводит к тому, что все транзакты, безусловно входящие в блок, заносятся в СП, определенный операндом <A>, в порядке, который задан операндом .

Если операнд <C> задан, проверяется индикатор СП. Если индикатор списка установлен в положение «1», вошедший транзакт заносится в СП в порядке, заданном операндом . Если же индикатор списка установлен в положение «0», он переводится в положение «1», и вошедший транзакт перемещается к блоку, заданному в операнде <C>.

ПРИМЕР 27.

LINK LIST,FIFO

Транзакт, вошедший в блок, помещается в конец списка с именем LIST.

Блок UNLINK (ВЫВЕСТИ ИЗ СПИСКА). Блок UNLINK удаляет транзакты из СП. После этого интерпретатор GPSS возобновляет их движение по модели.

Формат блока UNLINK:

UNLINK [<X>] <A>,[,<C>][,<D>][,<E>][,<F>]

Операнд <A> задает список пользователя (СП), из которого удаляются один или несколько транзактов. Операнд <A> может быть именем, положительным целым, СЧА.

В операнде указывается номер блока, к которому переходят

удаляемые из списка транзакты. Операнд может быть именем, положительным целым, СЧА.

Операнд <C> задает число транзактов, удаляемых из СП (счетчик удалений). Операнд <C> может быть именем, положительным целым, СЧА или «ALL» (означает удаление всех транзактов).

Операнд <D> может быть именем, целым, СЧА или «BACK». Действия, выполняемые при вхождении транзакта в блок UNLINK, зависят от того, на что ссылается операнд <D>. В операнде <D> могут быть указаны номер параметра, булева переменная или слово «BACK».

Номер параметра. Если операнд <E> пропущен, значение заданного параметра вошедшего транзакта сравнивается со значением этого же параметра транзактов СП. Если <E> не пропущен, значение заданного параметра транзактов СП сравнивается со значением СЧА из операнда <E>. В обоих случаях транзакты, удовлетворяющие заданному отношению, будут удалены из списка и направлены в блок, указанный в операнде .

Булева переменная BVj вычисляется отдельно для каждого транзакта из СП. Если для транзакта значение $BV_j=1$, он удаляется из СП (количество удаляемых транзактов не может превышать значения операнда <C>). Если $BV_j=0$ для всех транзактов списка, вошедший транзакт пытается переместиться в блок, заданный в операнде <F>. Если операнд <F> пропущен, транзакт пытается перейти в следующий по номеру блок.

Если в операнде <D> задана булева переменная, операнд <E> должен быть пустым. Если булева переменная BV_j имеет ссылку на какой-либо параметр, то эта ссылка относится к параметрам транзактов из списка, а не к входящему в блок UNLINK транзакту.

Слово «BACK». Из указанного списка, начиная с его конца, будет исключено столько транзактов, сколько задано операндом <C>. Операнд <E> в этом случае должен быть пустым.

Операнд <E> содержит СЧА, значение которого сравнивается со значением параметра транзактов СП (номер параметра указан в операнде <D>). Операнд <E> может быть именем, целым, СЧА.

Операнд <F> задает номер следующего блока для того транзакта, который входит в блок UNLINK в случаях, когда соответствующий СП пустой или не выполнено заданное отношение, или же указанная в операнде <D> булева переменная равна нулю для всех транзактов списка (т.е. в случае, когда из СП нельзя ничего удалить). Операнд <F> может быть именем, положительным целым, СЧА.

Операторы отношения, которые записываются во вспомогательном операнде X, определяют, какое условие (отношение) будет рассматриваться. Если этот оператор не задан, предполагается отношение равенства E.

Операторы отношения могут быть следующими:

G (больше) – отношение истинно, если значение параметра, заданного в операнде <D>, больше значения, заданного в операнде <E>;

GE (больше или равно) – отношение истинно, если значение параметра, заданного в операнде <D>, больше значения, заданного в операнде <E>, или равно ему;

L (меньше) – отношение истинно, если значение параметра, заданного в операнде <D>, меньше значения, заданного в операнде <E>;

LE (меньше или равно) – отношение истинно, если значение параметра, заданного в операнде <D>, меньше значения, заданного в операнде <E>, или равно ему;

E (равно) – отношение истинно, если значение параметра, заданного в операнде <D>, равно значению, заданному в операнде <E>;

NE (не равно) – отношение истинно, если значение параметра, заданного в операнде <D>, не равно значению, заданному в операнде <E>.

ПРИМЕР 28.

UNLINK LIST,MET,1

Первый транзакт из СП с именем LIST помещается в блок с меткой MET. Он заносится в CTC после транзактов с таким же приоритетом. Транзакт, вошедший в блок UNLINK, переходит в следующий блок.

4.2 Команда просмотра списка текущих событий. Блок BUFFER

При определенных условиях в фазе просмотра интерпретатор GPSS World возобновляет просмотр CTC с начала. Возобновление просмотра не происходит до тех пор, пока активный транзакт не остановится в своем движении. Транзакт не прекратит движение до тех пор, пока не войдет или в блок ADVANCE (в котором указано ненулевое время задержки), или в блок TERMINATE, или в один из следующих блоков: SEIZE, RELEASE, ENTER, LEAVE, LOGIC, PRIORITY.

Но иногда пользователю бывает необходимо, чтобы просмотр CTC немедленно возобновился по достижении транзактом заданной точки модели. Для получения такого эффекта используется блок BUFFER.

Формат блока BUFFER:

BUFFER

Когда транзакт входит в блок BUFFER, интерпретатор GPSS World прекращает его движение и тут же начинает заново просматривать CTC.

Транзакт, помещенный в буфер (т.е. вошедший в блок BUFFER), остается в СТС, сохраняя в нем то же положение, что и при входе в блок BUFFER. По мере продолжения нового просмотра, транзакт, помещенный в буфер, будет вновь обработан интерпретатором и его движение по программе модели возобновится.

5 КОМАНДЫ GPSS WORLD

Команды необходимы как для построения программы модели, так и для интерактивного взаимодействия с моделью. Они включают операторы описания данных и команды управления.

5.1 Команда START

Команда START используется для инициации начала моделирования.

Формат команды START:

START <A>[,][,<C>][,<D>]

Поля операндов имеют следующий смысл:

- <A> – значение счетчика завершений, определяющего момент окончания прогона модели (положительное целое число),
- – операнд вывода статистики. Этот операнд может принимать значение «NP» или быть опущенным; задание «NP» в операнде приводит к блокированию вывода статистики; по умолчанию выводится стандартная статистика,
- <C> – не используется,
- <D> – задает необходимость вывода содержания СТС и СБС; операнд <D> может быть положительным целым, если операнд <D> не равен 0, СТС и СБС выводятся.

Моделирование продолжается до тех пор, пока счетчик завершения, определенный операндом <A>, не достигнет нулевого значения. Для уменьшения значения счетчика используется блок TERMINATE (п. 2.1.2.2).

Стандартный числовой атрибут, который связан с этой командой, TG1 – текущее значение счетчика завершения.

5.2 Оператор INITIAL

Оператор INITIAL позволяет задавать начальные значения сохраняемых

величин, элементов матриц и логических ключей.

Формат оператора INITIAL:

INITIAL <A>[,]

Поля операндов имеют следующий смысл:

- <A> – СЧА сохраняемых величин, элементов матриц или логических ключей; в операнде <A> могут быть указаны:
 - LS<положительное целое>, LS\$<имя> – имя логического ключа,
 - X<положительное целое>, X\$<имя> – имя сохраняемой величины,
 - MX<положительное целое>() или MX\$<имя>() – имя элемента матрицы,
- – устанавливаемое значение, по умолчанию равно единице, может быть числом, строкой, именем или UNSPECIFIED.

При выполнении оператора INITIAL значение, заданное операндом , назначается логическому ключу, сохраняемой величине или элементу матрицы, определенному в операнде <A>.

Если операнд <A> определен как логический ключ, то операнд может быть только нулем или единицей.

Если в операнде использовано ключевое слово UNSPECIFIED, то сохраняемая величина, матрица или элемент матрицы устанавливается в «неопределенное» состояние. Обычно это ключевое слово используется, чтобы указать на отсутствие данных в матрице результатов, которая должна быть в дальнейшем проанализирована библиотечной процедурой ANOVA.

С оператором INITIAL связаны блоки LOGIC (п.2.2.3), SAVEVALUE (п.2.5.1) и MSAVEVALUE (п.2.5.2).

5.3. Команда RESET

Команда RESET сбрасывает в ноль статистику и СЧА системы, но не удаляет гранзакты из модели. Она используется для повторных экспериментов с моделью и сброса статистических данных переходного периода имитационного процесса.

Формат команды RESET:

RESET

Действия команды RESET:

- 1) значение относительного модельного времени (C1) устанавливается в

ноль;

2) значение абсолютного модельного времени (AC1) остается без изменений;

3) все датчики псевдослучайных чисел остаются неизменными;

4) значения сохраняемых величин и матриц, а также состояния логических ключей не изменяются;

5) счетчики числа входов в блоки (Nj) сбрасываются в ноль;

6) времена занятости устройств устанавливаются в ноль;

7) счетчики числа входов в многоканальные устройства (SCj) и максимального содержимого многоканальных устройств (SMj) остаются неизменными;

8) счетчики вхождений в очередь (QCj) и максимального содержимого очереди (QMj) устанавливаются равными текущей длине очереди;

9) в таблицах стираются накопленные статистические данные;

10) счетчики числа вхождений в списки (CCj) и максимального содержимого списков (CMj) устанавливаются равными текущей длине списка.

5.4 Команда CLEAR

Команда CLEAR сбрасывает всю накопленную статистику, удаляет все транзакты из модели и устанавливает отсчет (нумерацию) транзактов, сгенерированных блоками GENERATE, начиная с единицы.

Формат команды CLEAR:

CLEAR [<A>]

Действия команды CLEAR:

1) все транзакты удаляются из модели;

2) содержимое всех блоков устанавливается в ноль;

3) текущие счетчики блоков (Wj) сбрасываются в ноль;

4) общие счетчики блоков (Nj) сбрасываются в ноль;

5) системное время (C1 и AC1) устанавливается в ноль;

6) устройства становятся незанятыми и доступными;

7) многоканальные устройства становятся свободными и доступными;

8) времена занятости устройств, накопителей, очередей и СП устанавливаются в ноль;

9) максимальные значения содержимого очередей, СП и многоканальных устройств устанавливаются равными их текущему значению;

10) состояние датчиков псевдослучайных чисел не изменяется;

11) внутренний счетчик транзактов, генерируемых в блоках GENERATE,

устанавливается в ноль;

12) содержимое всех сохраняемых величин и матриц принимает нулевое значение;

13) логические ключи сбрасываются.

Если используется CLEAR OFF, выполняется все перечисленные выше действия за исключением последних двух пунктов. То есть, если операнд A=OFF, то логические ключи, сохраняемые величины и матрицы остаются без изменений.

После выполнения всех перечисленных операций команды CLEAR GPSS-модель просматривается интерпретатором в поиске блоков GENERATE. В каждом выявленном блоке GENERATE создается новый транзакт так же, как при первой интерпретации блока GENERATE. Заново вычисляется время начальной задержки и максимальное число транзактов, которые будут образованы в блоках GENERATE.

5.5 Команда RMULT

Моделирование часто требует нескольких различных последовательностей случайных чисел. Эти последовательности выдаются генераторами случайных чисел, которые действуют независимо друг от друга. При каждом запуске системы генераторы выдают одну и ту же последовательность чисел. Команда RMULT позволяет изменять такую последовательность путем изменения начальных множителей, являющихся параметрами генераторов.

В системе GPSS World генераторы случайных чисел создаются по мере необходимости, их явное определение необязательно.

Формат команды RMULT:

RMULT [**<A>**][****][**<C>**][**<D>**][**<E>**][**<F>**][**<G>**]

Поля операндов имеют следующий смысл:

- **<A>** – начальный множитель для первого генератора случайных чисел RN1,
- **** – начальный множитель для второго генератора случайных чисел RN2,
- **<C>** – начальный множитель для третьего генератора случайных чисел RN3,
- **<D>** – начальный множитель для четвертого генератора случайных чисел RN4,
- **<E>** – начальный множитель для пятого генератора случайных чисел RN5,
- **<F>** – начальный множитель для шестого генератора случайных чисел RN6,
- **<G>** – начальный множитель для седьмого генератора случайных чисел RN7.

Стандартный числовой атрибут, связанный с этой командой, – RN<номер генератора>. Он возвращает случайное целое число из интервала от 0 до 999.

Операнды должны быть положительными целыми числами. В этом операторе должен быть задан хотя бы один операнд.

ПРИМЕР 29.

RMULT 875,1237,,,319

Устанавливаются начальные состояния множителей генераторов случайных чисел 1, 2 и 5. Остальные значения остаются без изменений.

5.6 Оператор EQU

Оператор EQU предназначен для присвоения числовых значений именам, которые используются в модели.

Формат оператора EQU:

<имя> EQU <A>

Поля оператора имеют следующий смысл:

- <имя> – имя переменной, которой присваивается числовое значение,
- <A> – выражение.

Когда интерпретатор обрабатывает оператор EQU, он вычисляет выражение, заданное операндом <A>, после чего создает или переопределяет имя переменной. Имени присваивается результат вычисленного выражения. Полученное значение заменяет ссылки на это имя в операндах или выражениях, используемых в модели.

Значения имен могут использоваться как внутренние значения переменных пользователя, или они могут определять объекты, такие как метка. Именам, используемым как метки объектов, значения обычно не назначаются. Интерпретатор автоматически назначает индивидуальные значения именам, если они еще не появились в операторе EQU, в выражениях или операндах. Имена могут использоваться для определения объекта в СЧА.

Выражения, содержащиеся в операторе EQU, могут использовать любые из арифметических и логических операторов. Если в выражении используются параметры, они вычисляются для активного транзакта.

Имена, которым не были явно назначены значения, не могут использоваться в выражении. Необходимо назначить значение для имени прежде, чем будет вычислено выражение. Переменные пользователя могут быть заданы операторами EQU.

Если значение имени определено, то оно сохраняет свое значение на

протяжении всего прогона модели.

Переменные FVARIABLE и BVARIABLE используют одну и ту же область имен.

Если необходимо использовать числовое имя для объекта, то оно должно быть назначено оператором EQU до определения объекта.

ПРИМЕР 30.

TZA EQU 10

GENERATE TZA,FN\$EXPON

Переменной пользователя TZA назначено значение 10. Далее в программе можно использовать имя этой переменной.

5.7 Операторы описания объектов

К операторам описания объектов относятся:

- BVARIABLE – определяет булеву переменную;
- FUNCTION – определяет функцию;
- FVARIABLE – определяет действительную переменную с фиксированной точкой;
- MATRIX – определяет матрицу;
- QTABLE – определяет таблицу для очереди;
- STORAGE – определяет MKY;
- TABLE – определяет таблицу;
- VARIABLE – определяет переменную.

Все эти операторы были рассмотрены ранее.

5.8 Команда EXIT

Команда EXIT (ВЫХОД) предназначена для завершения работы с системой GPSS World.

Формат команды EXIT:

EXIT [<A>]

В результате выполнения команды EXIT система немедленно завершает работу. Операнд <A> является необязательным и используется для управления сохранением модели и результатов моделирования. Он может принимать следующие значения:

- 0 или не указан – в окне сообщений появляется запрос на сохранение для каждого вновь созданного или измененного в сеансе

- моделирования файла,
- 1 – все файлы сохраняются без запроса,
- – 1 – не сохраняются никакие файлы.

6 ЯЗЫК PLUS

6.1 Краткая характеристика языка PLUS

Язык GPSS можно отнести к языкам высокого уровня. В силу этого он имеет довольно слабые алгоритмические возможности. Для устранения этого недостатка в систему GPSS World добавлен PLUS – язык низкого уровня. Выражения, процедуры и эксперименты PLUS можно использовать в GPSS-моделях.

Рассмотрим основные элементы языка PLUS.

Алфавит языка PLUS (GPSS World) содержит алфавитно-цифровые и специальные символы. Для задания имен используются алфавитно-цифровые символы (прописные буквы A-Z, строчные буквы a-z, цифры 0-9 и знак подчеркивания «_»). Для обозначения операторов и пунктуации используются специальные символы («#», «*», «&», «+», «-», «/», «\», «,», «;»). В комментариях допускается использование символов русского алфавита «А-Я».

Имена – это созданные пользователем последовательности символов, используемые для обозначения объектов, переменных и процедур. Имя должно начинаться с буквы, в нем можно использовать от 1 до 250 алфавитно-цифровых символов. При этом имена не должны совпадать с ключевыми словами GPSS и с СЧА. Следует отметить, что GPSS World не различает регистр алфавита.

PLUS-выражение – это комбинация одного или нескольких элементов, называемых факторами. Выражения строятся с использованием операторов и вызовов процедур, обрабатывающих факторы. Выражения могут использоваться в PLUS-процедурах и в операндах операторов GPSS. Если выражение используется в операндах GPSS-блоков, то оно должно записываться в круглых скобках.

Типы данных. Переменные пользователя, элементы матриц, ячейки, параметры транзактов могут иметь значения различных типов данных. В GPSS World используются три основных типа данных:

- целочисленный (*Integer*) – это 32-х битовые двоичные числа. Если при выполнении арифметической операции происходит переполнение целочисленного значения, производится автоматическое преобразование его в вещественное значение;

- вещественный (*Real*) – это числа с плавающей точкой двойной точности. Для представления мантиссы используется 15 десятичных разрядов, А для порядка – значения в интервале от – 306 до 306;
- строковый (*String*) – это последовательность символов ASCII; строки не ограничены в размерах (максимальная их длина определяется параметрами настройки Edit/Setting.../Simulation).

Выражения могут быть вычисляемыми, численно вычисляемыми, вычисляемыми в виде строки.

Факторы – это основные элементы выражений, которые в свою очередь используются в операндах операторов GPSS и PLUS-процедурах.

Факторами выражений GPSS могут быть:

- строковые константы;
- вещественные константы;
- целочисленные константы;
- имена;
- элементы PLUS-матрицы;
- обращения к процедурам (значения, возвращаемые процедурами);
- стандартные числовые атрибуты.

Для объединения факторов в выражение используются операторы и вызовы процедур.

Операторы. В таблице 7 представлены арифметические операторы, используемые в выражениях GPSS World, перечисленные в порядке убывания их приоритетов.

Таблица 7 – Арифметические операторы языка PLUS

Опера тор	Действие	Результат	Унарный/бинарный (количество операндов)
–	Отрицание	Аддитивная инверсия	Унарный
^	Возведение в степень	Арифметический показатель степени	Бинарный
NOT или	Инверсия	1 (TRUE) или 0 (FALSE)	Унарный
AND	Логическое И	1 (TRUE) или 0 (FALSE)	Бинарный
OR	Логическое ИЛИ	1 (TRUE) или 0 (FALSE)	Бинарный
G или >	Больше	1 (TRUE) или 0 (FALSE)	Бинарный
L или <	Меньше	1 (TRUE) или 0 (FALSE)	Бинарный
E или =	Равно	1 (TRUE) или 0 (FALSE)	Бинарный

NE или /=	Не равно	1 (TRUE) или 0 (FALSE)	Бинарный
LE или <=	Меньше или равно	1 (TRUE) или 0 (FALSE)	Бинарный
GE или =>	Больше или равно	1 (TRUE) или 0 (FALSE)	Бинарный
# или *	Умножение	Арифметическое произведение	Бинарный
/	Деление	Арифметическое частное	Бинарный
\	Целочисленное деление	Целочисленное частное	Бинарный
@	Деление по модулю	Целочисленный остаток	Бинарный
+	Сложение	Арифметическая сумма	Бинарный
–	Вычитание	Арифметическая разность	Бинарный

Все арифметические операторы автоматически преобразуют строковые операнды в числовые значения.

К операторам языка PLUS также относятся:

- PROCEDURE – определяет PLUS-процедуру;
- EXPERIMENT – определяет PLUS-эксперимент;
- TEMPORARY – определяет и ограничивает область действия локальных переменных пользователя и локальных матриц (существующих только во время вызова конкретной процедуры);
- BEGIN/END составной оператор, создает блок PLUS-операторов;
- присваивание – предназначен для изменения значений переменных;
- вызов процедуры – вызывает библиотечную процедуру;
- помеченный оператор – класс операторов, начинающихся с метки;
- IF/THEN – условный оператор, проверяет выражение и, если результат равен «TRUE» (истина), выполняет действие;
- IF/THEN/ELSE – условный оператор проверяет выражение и в зависимости от результата производит то или иное действие;
- WHILE/DO – оператор цикла, несколько раз выполняет действие;
- GOTO – оператор безусловного перехода, передает управление к метке внутри процедуры;
- RETURN – останавливает выполнение процедуры и возвращает результат ее выполнения (после чего память, используемая процедурой, освобождается).

Процедуры языка PLUS могут записываться в любом месте модели (кроме тела другой процедуры). Остальные операторы PLUS могут появляться только внутри оператора PROCEDURE.

Различают процедуры *пользовательские* и *встроенные (библиотечные)*. Пользовательские процедуры обычно используются для изменения значений глобальных переменных и поименованных величин или для вычисления выражения и выдачи результата. В первом случае результат вычисления не требуется. Во втором – требуется, в этом случае обязательно наличие оператора возврата RETURN. Результат, выдаваемый PLUS-процедурой, используется в операндах или других PLUS-выражениях. Если в операторе RETURN не задано какое-либо выражение или в процедуре отсутствует оператор возврата RETURN, то возвращается значение 0.

Для определения PLUS-процедуры необходимо поместить оператор PROCEDURE в файл модели и выполнить транслирование оператора вместе с моделью или передать оператор PROCEDURE процессу моделирования в интерактивном режиме. После этого уже определенную PLUS-процедуру можно вызывать при вычислении выражений или в PLUS-операторах присваивания.

Если процедура используется более, чем в одной модели, то ее можно сохранить в исходном файле, называемом библиотекой процедур пользователя. Командой INCLUDE можно включать эту библиотеку в каждую модель, в которой будет использоваться данная процедура.

Эксперимент – это особая разновидность PLUS-процедуры пользователя. Для ее определения используется оператор EXPERIMENT. Эта процедура применяется для управления несколькими повторяющимися имитациями.

Эксперименты обычно используются совместно с библиотечной процедурой DoCommand() для управления процессом моделирования, А также с библиотечной процедурой дисперсионного анализа ANOVA для анализа результатов моделирования. Обычно результаты эксперимента записываются в глобальную матрицу, которая передается процедуре ANOVA. Существует возможность автоматически создавать отсеивающие и оптимизирующие эксперименты при помощи генераторов экспериментов.

Эксперимент вызывается только командой CONDUCT, которую можно загрузить с помощью функциональной клавиши.

Эксперименты и обыкновенные процедуры, непосредственно или косвенно вызываемые экспериментом, могут использовать библиотечную процедуру DoCommand() для выполнения операторов GPSS, включая команды и
блоки.

Библиотека процедур – это множество PLUS-процедур. Существует два типа библиотек: библиотека пользователя, представляющая собой совокупность процедур пользователя, и встроенная библиотека GPSS World, содержащая готовые к использованию строковые и математические процедуры. Для того чтобы процедуру можно было использовать в PLUS-выражении, она должна находиться в библиотеке процедур.

Встроенная часть библиотеки содержит процедуры, которые могут быть разбиты на следующие группы:

1. *Обслуживающие процедуры* – процедуры, используемые для управления прогонами имитаций и анализа экспериментов:

- Процедура DoCommand. Транслирует строку аргумента и передает результат трансляции выполняющейся модели. Эта процедура может вызываться только из экспериментов и процедур, вызванных из экспериментов.
- Процедура ANOVA. Проводит многофакторный дисперсионный анализ, создает таблицу ANOVA, таблицу описательных статистик и выводит эти данные в журнал сессии.
- Процедура Exit() завершает сеанс работы с GPSS World – останавливает процесс моделирования и закрывает все окна GPSS.

2. *Файловые процедуры (процедуры потоков данных)* – процедуры управления потоками данных внутри PLUS-процедур. Потоки данных используются для чтения и записи файлов или для хранения и доступа больших объемов данных в памяти. Файловые процедуры выполняют следующие операции:

- Open(DataStream, FileNameString) инициализирует поток данных;
- Close(DataStream) – закрывает поток данных и извлекает код ошибки;
- Read(DataStream) – считывает текстовую строку из потока данных;
- Write(DataStream, String) – передает потоку данных текстовую строку;
- Seek(DataStream, NewLinePosition) – устанавливает позицию текущей строки потока данных и извлекает предыдущую позицию строки.

3. *Процедуры динамического вызова* используются для вызова функций, хранящихся во внешних исполняемых файлах, включая динамически подключаемые библиотеки DLL. Соответствующие процедуры можно использовать для запуска функций, предоставляемых другими фирмами в отдельных исполняемых файлах и поддерживающих протокол CDECL.

4. *Математические процедуры*. К ним относятся:

- ABS(Expression) – абсолютное значение;
- ATN(Expression) – арктангенс (в радианах);
- COS(Expression) – косинус;
- EXP(Expression) – число e , возведенное в степень аргумента;

- INT(Expression) – выделение целой части с отбрасыванием дробной части;
- LOG(Expression) – натуральный логарифм;
- SIN(Expression) – синус;
- SQR(Expression) – квадратный корень;
- TAN(Expression) – тангенс.

5. *Вероятностные распределения процедуры теоретических вероятностных распределений.* В GPSS World в библиотеку процедур включено 24 вероятностных распределений. При вызове вероятностного распределения требуется определить аргумент *Stream* (может быть выражением), который определяет номер генератора случайных чисел. При моделировании генераторы случайных чисел создаются по мере необходимости и их явное определение не обязательно. Большинство вероятностных распределений имеют некоторые параметры. Аргументы процедур, называемые обычно *Locate*, *Scale* и *Shape*, часто используются для этих целей. Аргумент *Locate* используется после построения применяемого распределения и прибавляется к нему. Это позволяет горизонтально перемещать функцию распределения по оси X. Аргумент *Scale* обычно меняет масштаб функции распределения, а *Shape* – ее форму.

Встроенная библиотека процедур содержит следующие вероятностные распределения:

- 1) бета (Beta);
- 2) биномиальное (Binomial);
- 3) Вейбулла (Weibull);
- 4) дискретно-равномерное (Discrete Uniform);
- 5) гамма (Gamma);
- 6) геометрическое (Geometric);
- 7) Лапласа (Laplace);
- 8) логистическое (Logistic);
- 9) логлапласово (LogLaplace);
- 10) логлогистическое (LogLogistic);
- 11) логнормальное (LogNormal);
- 12) нормальное (Normal);
- 13) обратное Вейбулла (Inverse Weibull);
- 14) обратное Гаусса (Inverse Gaussian);
- 15) отрицательное биномиальное (Negative Binomial);
- 16) Парето (Pareto);
- 17) Пирсона типа V (Pearson Type V);
- 18) Пирсона типа VI (Pearson Type VI);
- 19) Пуассона (Poisson);
- 20) равномерное (Uniform);
- 21) треугольное (Triangular);
- 22) экспоненциальное (Exponential);
- 23) экстремального значения A (Extreme Value A);
- 24) экстремального значения B (Extreme Value B).

ПРИМЕР 31.

Для генерации потока транзактов можно использовать библиотечную процедуру экспоненциального распределения с параметром $\lambda = 0,25$ и вызовом генератора случайных чисел RN1:

GENERATE (Exponential(1,0,(1/0.25)))

6. *Строковые процедуры* выполняют операции со строками:

- Align(InsertString, SourceString, Offset) – вставляет одну строку в другую с выравниванием по правому краю;
- Catenate(String1, String2) – производит объединение строк;
- Copies(SourceString, Count) – создает одну строку из нескольких копий исходной;
- Datatype(Datum) – возвращает строку, содержащую тип данных аргумента;
- Find(TestString, SourceString) – вычисляет смещение одной строки, содержащейся в другой строке;
- Left(SourceString, MaxCount) – возвращает левую часть заданной строки;
- Length(SourceString) – возвращает количество символов в строке;
- Lowercase(SourceString) – переводит все прописные буквы в строке в строчные;
- Place(InsertString, SourceString, Offset) – вставляет одну строку в другую с выравниванием по левому краю;
- PolyCatenate(String1, String1,...) – производит объединение двух или более строк;
- Right(SourceString, MaxCount) – возвращает правую часть заданной строки;
- String(Datum) – преобразует данные в их строковый эквивалент;
- StringCompare(String1, String2) – сравнивает две строки;
- Substring(SourceString, Offset, MaxCount) – возвращает подстроку заданной строки;
- Trim(SourceString) – удаляет начальные и конечные пробелы;
- Uppercase(SourceString) – переводит все строчные буквы в строке в прописные;
- Value(Datum) – возвращает числовой эквивалент строки;
- Word(SourceString, WordNumber) – возвращает заданное слово строки.
- Процедуры запроса возвращают информацию о состоянии транзакта, находящегося в модели. К ним относятся такие процедуры:
 - QueryXNExist(TransactionNumber) – возвращает 1, если транзакт существует в модели, иначе – 0;
 - QueryXNParameter(TransactionNumber, Parameter) – возвращает значение параметра транзакта. Если искомого параметра не существует, происходит останов по ошибке;
 - QueryXNAssemblySet(TransactionNumber) – возвращает номер семейства, к которому принадлежит транзакт;

- QueryXNPriority(TransactionNumber) – возвращает приоритет транзакта в виде целого числа;
- QueryXNMI(TransactionNumber) – возвращает время входа транзакта в систему.

6.2 Пример использования языка PLUS

Пусть требуется для заданного числа вычислить значение факториала.

Присвоим сохраняемой величине с именем N_DAN начальное значение 10. Создадим один транзакт для последующего моделирования. Для обращения к процедуре вычисления факториала, получения результата вычисления и присвоения его искомой переменной используем блок SAVEVALUE. В нем производится вычисление и запись результата вычисления процедуры N_FACT в сохраняемую переменную REZ_FACT. Вызов процедуры вычисления представляется в круглых скобках с указанием фактического значения аргумента. Затем транзакт удаляется из модели.

```
INITIAL    X$N_DAN,10
GENERATE    ,,1
SAVEVALUE  REZ_FACT,(N_FACT(X$N_DAN))
TERMINATE  1

PROCEDURE  N_FACT(N_X)
BEGIN
    IF ((N_X=0)OR(N_X=1)) THEN RETURN 1;
    ELSE RETURN(N_XN#(N_FACT(N_X-1)));
END;
```

7 ДИАЛОГОВЫЕ ВОЗМОЖНОСТИ GPSS WORLD

Взаимодействие пользователя с системой GPSS World осуществляется с помощью оконного интерфейса в режиме активного диалога.

7.1 Диалоговые окна

В системе предусмотрены диалоговые окна, которые позволяют отображать информацию о состоянии отдельных объектов на экране дисплея. Эта информация может быть как статической, так и динамической. Главное окно, появляющееся при запуске системы, показано на рисунке 4.

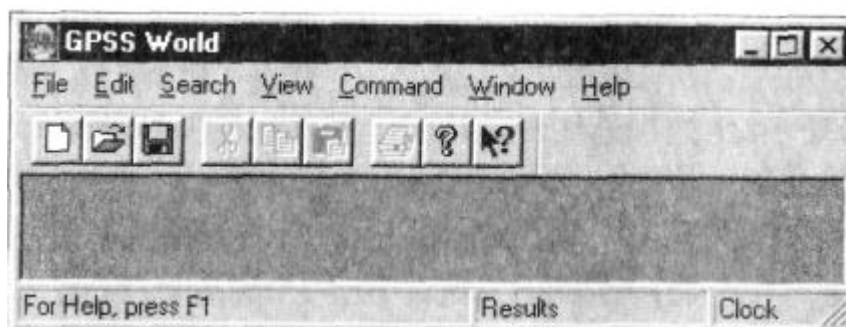


Рисунок 4 – Главное окно GPSS World

Пункт *File* главного меню содержит стандартные команды создания/сохранения GPSS-моделей или текстовых файлов.

Пункт *Edit* главного меню содержит команды редактирования GPSS-моделей.

Для удобства создания модели можно воспользоваться пунктом меню *Edit/InsertGPSSBlock...*, позволяющим выбрать из специального окна блоков и вставить в модель любой GPSS-блок (рис. 5). При выборе блока открывается окно с его параметрами (рис. 6). Использование этого средства GPSS World гарантирует правильность формирования строки модели с выбранным блоком.

Insert GPSS Block into Model Object		
ADOPT	ASSEMBLE	ALTER
ADVANCE	CLOSE	COUNT
ASSIGN	GATE	DISPLACE
BUFFER	JOIN	EXAMINE
DEPART	LINK	EXECUTE
ENTER	LOGIC	FAVAIL
GENERATE	LOOP	FUNAVAIL
LEAVE	MATCH	GATHER
MARK	OPEN	INDEX
MSAVEVALUE	PREEMPT	INTEGRATION
PLUS	PRIORITY	SAVAIL
QUEUE	READ	SCAN
RELEASE	REMOVE	SELECT
SAVEVALUE	RETURN	SUNAVAIL
SEIZE	SEEK	TABULATE
SPLIT	TEST	TRACE
TERMINATE	UNLINK	UNTRACE
TRANSFER	WRITE	

Рисунок 5 – Окно вставки GPSS-блока в модель

Пункт меню *Edit/Insert Experiment* позволяет вставить в модель эксперимент.

Пункт меню *Edit/Expression Window...* предназначен для редактирования информации в окне выражений, если это окно использовалось в модели.

Пункт меню *Edit/Plot Window...* предназначен для редактирования информации в окне графиков.

Пункт меню *Edit/Settings...* позволяет задавать параметры имитации, отчетов, генераторов случайных чисел, функциональных клавиш и выражений.

Пункт *Search* главного меню помогает передвигаться внутри текстового объекта. Первый его пункт *Find/Replace* (Найти/Заменить) открывает обычный диалог для поиска и замены текстовой информации.

Последующий набор пунктов меню используется для работы с закладками, позволяя размещать невидимые маркировочные знаки, которые сохраняются с объектом. Они составляют циклический список, который можно просматривать с помощью соответствующих команд меню или с помощью клавиш.

Enter Block Information

GENERATE

GENERATE - Create XN for future entry.

A: Intergeneration time.

B: Halfrange or Function Modifier.

C: Start delay time.

D: Creation limit.

E: Priority.

F:

G:

H:

Label:

Comment:

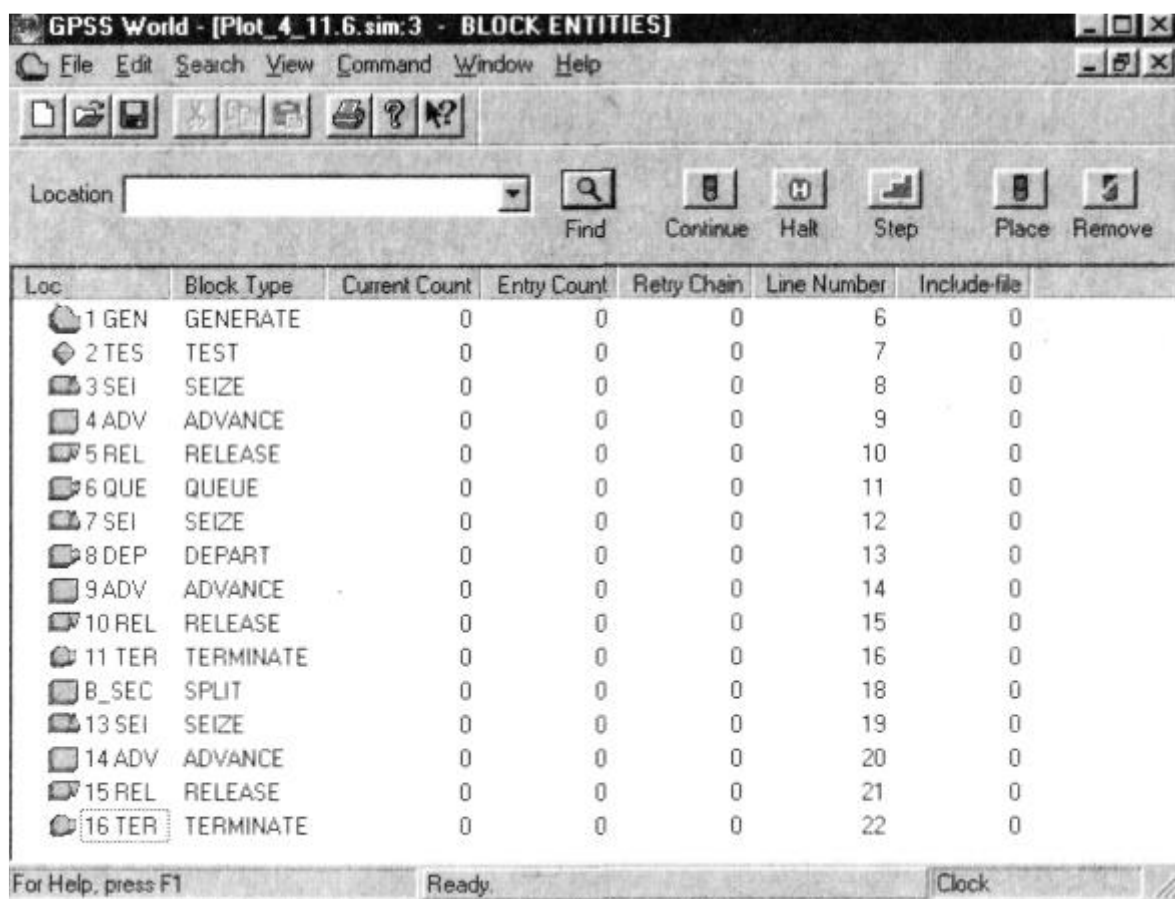
OK Cancel Help

Рисунок 6 – Окно с параметрами блока

Пункт меню *Search/NextBookmark* переводит к позиции следующей закладки в тексте. Пункт меню *Search/Mark* водит закладку в текущую позицию курсора, *Search/Unmark* – снимает выделение, удаляя текущую закладку, а *Search/Unmark All* снимает все закладки. Пункт меню *Search/Select to Bookmark* выделяет текст от текущей позиции курсора до текущей позиции закладки. Последние два пункта меню *Search* имеют дело с сообщениями об ошибках, которые возникают при трансляции GPSS-модели.

Ошибки трансляции заносятся в циклический список. Этот список хранится вместе с GPSS-моделью и модифицируется при повторной трансляции. Для поиска ошибок используется пункты меню *Search/Next Error* (следующая ошибка) и *Search/Previous Error* (предыдущая ошибка). Для быстрого поиска с помощью клавиатуры используются комбинации клавиш [b+a+N] и [b+a+P] соответственно. Курсор останавливается перед ошибкой.

Пункт *View* главного меню управляет отображением информации в окнах. Пункт меню *View/Notices* (заметки) выводит информацию о текущей версии GPSS World и ее особенностях. Пункт меню *View/Toolbar* позволяет отображать или не отображать панель инструментов в главном окне. Пункт меню *View/Entity Details* управляет выдачей детальной информации для некоторых динамических окон. Например, в окне блоков может быть показана детальная информация по всем блокам модели (рис.7) или отображаться только их графическое представление. Пункт меню *View/Simulation Clock* позволяет отображать часы модельного времени в нижнем правом углу главного окна.



Loc	Block Type	Current Count	Entry Count	Retry Chain	Line Number	Include-file
1 GEN	GENERATE	0	0	0	6	0
2 TES	TEST	0	0	0	7	0
3 SEI	SEIZE	0	0	0	8	0
4 ADV	ADVANCE	0	0	0	9	0
5 REL	RELEASE	0	0	0	10	0
6 QUE	QUEUE	0	0	0	11	0
7 SEI	SEIZE	0	0	0	12	0
8 DEP	DEPART	0	0	0	13	0
9 ADV	ADVANCE	0	0	0	14	0
10 REL	RELEASE	0	0	0	15	0
11 TER	TERMINATE	0	0	0	16	0
B_SEC	SPLIT	0	0	0	18	0
13 SEI	SEIZE	0	0	0	19	0
14 ADV	ADVANCE	0	0	0	20	0
15 REL	RELEASE	0	0	0	21	0
16 TER	TERMINATE	0	0	0	22	0

Рисунок 7 – Детальная информация по блокам модели

Пункт *Command* главного меню используется для создания и управления объектами имитации. Пункт меню *Command/Create Simulation* (создать имитацию) вызывает транслятор для создания объекта имитации, который включает кроме GPSS-модели, также файлы, связанные с ней. Команда меню *Command/Retranslate* доступна для выполнения повторной

трансляции после исправления ошибок. Остальные пункты меню *Command* выполняют команды

Пункт *Windows* главного меню GPSS World предоставляет пользователю возможность использования различных окон для наблюдения и взаимодействия с моделью в процессе имитации. Окна, отображающие визуальное состояние имитации, могут быть сохранены и распечатаны. Некоторые окна делают как бы мгновенный снимок состояния различных объектов имитации в некоторый момент модельного времени. Изображения в окнах изменяются динамически в интерактивном режиме взаимодействия с моделью. Следует отметить, что открытые динамические окна существенно замедляют скорость прогона модели.

При моделировании может быть открыто любое число динамических окон для следующих объектов: блоков, устройств, накопителей, очередей, логических ключей, сохраняемых величин, матриц, таблиц.

Кроме того, можно воспользоваться окнами для графиков и выражений, что позволяет проследить изменения значений переменных во время имитации.

7.2 Стандартная выходная статистика. Описание элементов файла статистики

Отформатированный файл статистики состоит из подразделов, содержащих стандартную статистику об объектах GPSS, используемых в данной модели (FACILITY, QUEUE, STORAGE и т.д.).

Начинается файл статистики с заголовка, который берется из поля комментария, расположенного перед началом программы. Заголовок появляется на каждой странице файла статистики. Например,

GPSS Report file TEST (V2) 06-24-1989 21:57:38

Далее следует строка, содержащая основную информацию о результатах работы модели. Например,

START TIME	END TIME	BLOCKS	FACILITIES	STORAGES	FREE MEMORY
0	10850	24	1	1	15850

Элементы статистики, представленные в этой строке имеют следующее содержание:

- **START TIME** – абсолютное модельное время в момент начала моделирования. Оно эквивалентно абсолютному модельному времени после последнего применения операторов RESET или CLEAR;
- **END TIME** – абсолютное время, когда счетчик завершений принял значение 0;
- **BLOCKS** – количество блоков, использованных в текущей модели к моменту завершения моделирования;
- **FACILITIES** – количество устройств, использованных в модели к моменту завершения моделирования;

- STORAGES – количество многоканальных устройств, использованных в модели к моменту завершения моделирования;
- FREE MEMORY – количество байтов памяти, доступной для дальнейшего использования.

Затем в файле статистики следует информация об именах, которые просматривает GPSS в ходе моделирования. Информация об именах имеет следующий вид:

NAME	VALUE	TYPE
MOTOR	10001	2

Поле NAME отмечает имена, содержащиеся в программе модели.

Поле VALUE определяет числовое значение (номер), соответствующее имени. Система устанавливает начальный номер равным 10000. Поле TYPE равно 0, если значение имени устанавливает пользователь, равно 2, если значение имени устанавливает система, 3, если имя является именем блока.

Далее описываются блоки текущей модели в виде:

LINE	LOC	BLOCK	TYPE	ENTRY	COUNT	CURRENT	COUNT	RETRY
90	1	GENERATE		383		0		0

Поле LINE определяет номер строки в рабочей модели, связанный с блоком GPSS. Поле LOC определяет имя или номер этого блока, Поле BLOCK TYPE определяет тип блока GPSS. Поле ENTRY COUNT определяет количество транзактов, вошедших в данный блок после последнего выполнения блоков RESET или CLEAR или с начала работы программы модели. Поле CURRENT COUNT определяет количество транзактов, находящихся в данном блоке в конце моделирования. Поле RETRY определяет количество транзактов, ожидающих специальных условий, зависящих от состояния данного блока.

Если в модели используются *объекты типа «устройство»*, то далее в файле статистики идет информация об этих объектах.

FACILITY	ENTRIES	UTIL	AVE.TIME	AVAILABLE	OWNER
TELLER	254	0.996	395.67	1	291

PEND	INTER	RETRY	DELAY
0	0	0	78

Поле FACILITY определяет номер или имя объекта типа «устройство». Поле ENTRIES определяет количество раз, когда устройство было занято или прервано после последнего выполнения блоков RESET или CLEAR или с начала работы программы. Поле UTIL. определяет часть периода моделирования, в течение которого устройство было занято. Поле AVE.TIME определяет среднее время занятости устройства одним транзактом в течение периода моделирования после последнего выполнения операторов RESET или CLEAR. Поле AVAILABLE определяет состояние готовности устройства в конце периода моделирования. Оно равно 1, если устройство готово и 0 - если не готово. Поле OWNER определяет номер последнего транзакта, занимавшего устройство: 0 означает, что устройство не занималось. Поле PEND определяет количество транзактов, ожидающих устройство, находящееся в режиме прерывания. Поле INTER определяет

количество транзактов, обработка которых прервана на устройстве в данный момент модельного времени. Поле RETRY определяет количество транзактов, ожидающих специальных условий, зависящих от состояния объекта типа «устройство». Поле DELAY определяет количество транзактов, ожидающих занятие устройства, включая транзакты, ожидающие выхода устройства из режима прерывания.

В случае использования в модели *объектов типа «очередь»*, далее следует информация об этих объектах.

QUEUE	MAX	CONT.	ENTRIES	ENTRIES (0)	AVE.CONT
TELLER	78	10	332	1	36.24
AVE.TIME	AVE. (-0)	RETRY			
11613.51	118 4 8. 6	0			

Поле QUEUE определяет имя или номер объекта типа «очередь». Поле MAX определяет максимальное содержимое объекта типа «очередь» в течение периода моделирования, который начинается с начала работы модели или с последнего оператора RESET или CLEAR. Поле CONT. определяет текущее содержимое объекта типа «очередь» в конце периода моделирования. Поле ENTRIES определяет общее количество входов в очередь в течение периода моделирования (счетчик входов). Поле ENTRIES(0) определяет общее количество входов в очередь с нулевым временем ожидания (счетчик «нулевых» входов). Поле AVE.CONT определяет среднее значение содержимого очереди. Поле AVE.TIME определяет среднее время, проведенное транзактом в очереди с учетом всех входов в очередь. Поле AVE.(-0) определяет среднее время, проведенное транзактом в очереди без учета «нулевых» входов в очередь. Поле RETRY определяет количество транзактов, ожидающих специальных условий, зависящих от состояния объекта типа «очередь».

Если в модели использовались *объекты типа «многоканальное устройство»*, то далее в файле статистики идет информация об этих объектах.

STORAGE	CAP.	REMAIN	MIN	MAX	ENTRIES	AVE.	AVE.C.
POOL	3	3	0	1	50	1	0.99
UTIL.	RETRY	DELAY					
0.33	10	0					

Поле STORAGE определяет имя или номер объекта типа «многоканальное устройство». Поле CAP определяет емкость многоканального устройства, заданную оператором STORAGE. Поле REMAIN определяет число единиц свободной емкости многоканального устройства в конце периода моделирования. Поле MIN определяет минимальное количество используемой емкости многоканального устройства за период моделирования. Поле MAX определяет максимальное количество используемой емкости многоканального устройства за период моделирования. Поле ENTRIES определяет количество входов в многоканальное устройство за период моделирования. Поле AVE определяет

состояние готовности многоканального устройства в конце периода моделирования: 1 - означает, что многоканальное устройство готово, 0 - не готово. Поле AVE.C определяет среднее значение занятой емкости за период моделирования. Поле UTIL определяет часть периода моделирования, в течение которого многоканальное устройство использовалось. Поле RETRY определяет количество транзактов, ожидающих специальных условий, зависящих от состояния многоканального устройства. Поле DELAY определяет количество транзактов, ожидающих возможности входа в блок ENTER.

Если в модели используются блоки TABLE, в файле стандартной статистики будет представлена информация о таблицах.

TABLE	MEAN	STD.DEV.	RETRY	RANGE	FREQUENCY	%CUM
TAB	38.90	22.22	0	– 2	10	3.95
				2 – 4	6	6.32
				4 – 6	5	8.30
				6 – 8	3	9.49
				8 – 10	6	11.86
				10 – 12	6	14.23
				12 – 14	9	17.79
				14 – 16	8	20.95
				16 – 18	3	22.13
				18 –	197	100.00

Поле TABLE определяет имя или номер объекта типа «таблица». Поле MEAN определяет среднее взвешенное значение табулируемого аргумента. Значение преобразуется в формат двойной точности при выводе в файл статистики. Поле STD.DEV. определяет взвешенное среднеквадратичное отклонение.

$STD.DEV. = \sqrt{SOS / (COUNT - 1) - (SUM / (COUNT) (COUNT - 1))}$, где SOS – сумма квадратов значений аргумента, COUNT - число входов в таблицу, SUM – квадрат суммы значений аргументов таблицы, STD.DEV. преобразуется в формат двойной точности при выводе в файл статистики. Поле RETRY определяет количество транзактов, ожидающих выполнения специальных условий, зависящих от состояния объекта типа «таблица». Поле RANGE определяет нижний и верхний пределы частотных классов. При попадании табулируемого аргумента в интервал, который имеет значение, большее нижней границы частотного класса или меньшее или равное верхней границе, изменяется значение частоты (FREQUENCY). Операнд блока TABULATE может быть использован для определения величины, которая добавляется в частотный класс при попадании табулируемого значения в этот частотный класс. Частотные классы, суммарное значение которых равно 0, в файл статистики не выводятся. Значения частотных классов не уменьшаются при их изменении. Поле FREQUENCY определяет суммарную величину, которая формируется при попадании табулируемого аргумента в указанные границы. Суммируются значения операнда блоков TABULATE. Поле %CUM. определяет величину частоты в процентах к общему количеству значений табулируемого аргумента.

Далее в файле статистики выводятся *списки пользователя*, если они использовались в модели.

USER CHAIN	CHAINSIZE	RETRY	AVE.CONT	ENTRIES	MAX	AVE.TIME
TAXILINE	202	0	100.70	252	203	40289.50

Поле USER CHAIN определяет номер или имя объекта типа «список пользователя», поле CHAIN SIZE определяет количество сообщений в окне пользователя в конце периода моделирования. Поле RETRY определяет количество транзактов, ожидающих наступления специальных условий, связанных с состоянием объекта типа «список пользователя». Поле AVE.CONT определяет среднее содержимое списка пользователя в течение периода моделирования. Поле ENTRIES стреляет общее количество транзактов, помещаемых в список пользователя в течение периода моделирования. Поле MAX определяет максимальное количество транзактов в списке пользователя за период моделирования. Поле AVE.TIME определяет среднее время пребывания транзакта в списке пользователя.

Далее в файле выходной статистики следует *информация о логических переключателях*, если они использовались в модели.

LOGICSWITCH	VALUE	RETRY
SWITCH1	1	0

Поле LOGICSWITCH определяет имя или номер объекта типа «логический переключатель». Поле VALUE определяет значение логического переключателя в конце моделирования: 1 - означает «установлен» или («истина»), 0 - означает «сброшен» или («ложь»). Поле RETRY определяет количество транзактов, ожидающих наступления специальных условий, зависящих от состояния логического переключателя.

Далее в файле статистики следует *информация о сохраняемых величинах (ячейках)*, если они использовались в модели.

SAVEVALUE	VALUE	RETRY
CLOCKSAVE	+100571	0

Поле SAVEVALUE определяет имя или номер объекта, типа «сохраняемая величина». Поле VALUE определяет значение сохраняемой величины в конце моделирования. Поле RETRY определяет количество транзактов, ожидающих наступления специальных условий, зависящих от состояния сохраняемой величины.

Далее следует описание матричных сохраняемых величин, если они использовались в модели.

MATRIX	RETRY	ROW	COLUMN	VALUE
ARRAY1	0	1	1	+0
		2	2	-15
		3	3	+10

Поле MATRIX определяет имя или номер матричной сохраняемой величины. Поле RETRY определяет количество транзактов, ожидающих наступлений специальных условий, связанных с состоянием матричной сохраняемой величины. Поле ROW и COLUMN определяют номер строки и номер столбца матричной сохраняемой величины. Поле VALUE определяет

значение элемента матричной сохраняемой величины в конце моделирования. Элементы, равные 0, также выводятся.

Далее следует статистика о *списке текущих событий*.

CEC XACT NUMBER	PRI	MI	CURRENT	NEXT	PARAMETER	VALUE
290	3	765	14	15	LINESIZE	18

Список текущих событий (CEC – current events chain) выводится в файл статистики, если в команде START значения операнда <D> равно 1. Поле CEC XACT NUMBER определяет номер каждого транзакта в списке текущих событий. Поле PRI определяет приоритет транзакта. Поле MI определяет время транзакта или время транзакта, породившего данный транзакт. Поле CURRENT определяет номер блока, в котором находится транзакт в конце моделирования. Поле NEXT определяет номер следующего блока, в который должен был войти транзакт. Поле PARAMETER определяет имена или номера параметров транзактов: 0 означает, что транзакт не имеет параметров. Поле VALUE определяет значение этого параметра.

Далее выводится информация о *списке будущих событий*.

FEC XACT NUMBER	PRI	BDT	CURRENT	NEXT	PARAMETER	VALUE
291	3	100873	5	6	0	0
382	3	100976	0	1	0	0

Список будущих событий (FEC – future events chain) будет выводиться в файл статистики, если операнд <D> блока START равен 1. Поле FEC XACT NUMBER определяет номер сообщения в списке будущих событий. Поле PRI определяет приоритет транзакта. Поле BDT определяет момент абсолютного модельного времени, когда сообщение покинет список будущих событий. Поле CURRENT определяет номер блока, в котором транзакт находится в конце моделирования. Поле NEXT определяет номер следующего блока, в который должен войти транзакт. Поле PARAMETER определяет имена или номера параметров транзактов: 0 означает, что у транзакта нет параметров. Поле VALUE определяет значение параметра.

ЗАКЛЮЧЕНИЕ

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Боев В. Моделирование систем. Инструментальные средства GPSS World. – Спб.: БХВ-Петербург, 2004.
2. Кудрявцев Е.М. GPSS World. Основы имитационного моделирования различных систем. – М.: ДМК Пресс, 2004.
3. Кудрявцев Е.М., Добровольский А. Основы работы с универсальной системой моделирования GPSS World. – М.: Изд-во ассоциации строительных вузов, 2005.
4. Рыжиков Ю.И. Имитационное моделирование. Теория и технологии. – Спб.: КОРОНА принт; М.: Альтекс-А, 2004.
5. Советов Б.Я., Яковлев С.А. Моделирование систем: Учеб. для вузов – 6-е изд., стереотипное. – М.: Высш. шк., 2009.
6. Советов Б.Я. Моделирование систем. Практикум: Учеб. пособие для вузов / Б.Я. Советов, С.А. Яковлев. – 4-е изд., стереотипное. – М.: Высш. шк., 2009.
7. Томашевский В., Жданова Е. Имитационное моделирование в среде GPSS. – М.: Бестселлер, 2003.
8. Шрайбер Т. Моделирование на GPSS. – М.:Машиностроение, 1980.
9. GPSS World Tutorial Manual.
<http://www.minutemansoftware.com/tutorial/t1.htm#Chapter%201>
10. GPSS World Reference Manual.
<http://www.minutemansoftware.com/reference/r4.htm>

ПРИЛОЖЕНИЯ

Приложение А Операторы описания блоков GPSS World

В поле операции пользователь должен записать обозначение блока. Задание исходных данных, необходимых для выполнения операций, соответствующих блоку, производится в поле операндов. Всего может быть 7 операндов, значения которых определяются типом блока. Операнды обозначаются буквами A, B, C, D и т.д. Если у блока несколько операндов, они разделяются запятыми, вместо пропущенного операнда ставится запятая.

Ниже приводится список блоков GPSS и дается их краткая характеристика:

ADVANCE – задержка сообщения на определенное время, с включением его в список будущих событий;

ALTER – проверка и модификация сообщения в группе;

ASSEMBLE – вывод из модели одного или нескольких сообщений;

ASSIGN – модификация параметров сообщений;

BUFFER – размещение сообщения в списке текущих событий последним в своем приоритетном уровне;

COUNT – подсчет числа элементов заданного множества, удовлетворяющих указанному условию;

DEPART – вывод сообщения из очереди;

ENTER – захват сообщением всего или части многоканального устройства;

EXECUTE – принудительная обработка сообщения некоторым блоком модели;

FAVAIL – смена состояния устройства на состояние «готово к использованию»;

FUNAVAIL – смена состояния устройства на состояние "не готово к использованию";

GATE – проверка состояния объекта модели и изменение направления движения потока сообщений;

GATHER – накопление нескольких связанных друг с другом сообщений (ансамбль сообщений);

GENERATE – создание сообщения и размещение его в списке будущих событий;

INDEX – изменение параметров сообщения;

JOIN – размещение членов в числовой группе или в группе сообщений;

LEAVE – освобождение всего или части многоканального устройства, пересылка сообщения в список пользователя;

LOGIC – модификация логического ключа;

LOOP – организация цикла с уменьшением значения параметра

сообщения;

MARK – сохранение значения системного времени в параметре сообщения;

MATCH – ожидание сообщением связанных с ним сообщений в других блоках модели С синхронизация сообщений;

MSAVEVALUE – присваивание значений элементам матриц;

PREEMPT – прерывание обработки обрабатываемого устройством сообщения и захват устройства активным сообщением;

PRIORITY – изменение приоритета сообщения;

QUEUE – включение сообщения в очередь;

RELEASE – освобождение устройства сообщением;

REMOVE – удаление члена из числовой группы или группы сообщений;

RETURN – освобождение захваченного устройства;

SAVAIL – изменение состояния многоканального устройства на состояние «готово к использованию»;

SAVEVALUE – присваивание значений ячейке сохраняемых величин;

SCAN – просмотр содержимого группы до выполнения некоторого условия;

SEIZE – занятие устройства или ожидание его освобождения для последующего занятия;

SELECT – выбор блока для дальнейшего продвижения сообщения с использованием значения параметра сообщения;

SPLIT – создание копии сообщения с адресацией ее в некоторый блок;

SUNAVAIL – изменение состояния многоканального устройства на состояние «не готово к использованию»;

TABULATE – обновление данных таблицы;

TERMINATE – уничтожение сообщения, уменьшение счетчика завершенных сообщений;

TEST – проверка условий и модификация направления движения потока сообщений;

TRACE – установка флага трассировки для активного сообщения;

TRANSFER – пересылка сообщения на указанный блок;

UNLINK – вывод сообщения из списка пользователя;

UNTRACE – сброс флага трассировки для активного сообщения.

Приложение Б Операторы описания данных и контроля управления GPSS World

Для описания многоканальных устройств, переменных, таблиц и т.д. требуются операторы описания объектов. Информация, необходимая для контроля за процессом моделирования, задается с помощью управляющих операторов GPSS. Ниже приводится список операторов описания данных и контроля управления GPSS с краткими комментариями по их назначению:

- FVARIABLE** – определение булевских переменных;
- CLEAR** – приведение модели к исходному состоянию;
- END** – завершение работы GPSS;
- EQU** – присвоение целых значений именам;
- FUNCTION** – определение функций;
- FVARIABLE** – определение переменных с плавающей точкой;
- INITIAL** – присвоение или модификация значений ячеек охраняемых величин или элементов матриц ячеек сохраняемых величин;
- MATRIX** – определение матриц ячеек сохраняемых величин;
- QTABLE** – определение Q-таблиц;
- RESET** – сброс статистики без сброса датчиков случайных чисел и таймера абсолютного времени;
- RMULT** – установка начальных значений одного или более генераторов случайных чисел;
- SIMULATE** – установка предела времени моделирования для последующей имитации;
- START** – установка значения счетчика завершения и запуск процесса моделирования;
- STORAGE** – определение многоканальных устройств;
- TABLE** – определение таблиц;
- VARIABLE** – определение целых переменных.

Приложение В Сообщения GPSS World об ошибках

- **A transaction is blocked on an impossible condition** – сообщение заблокировано при невыполнимых условиях.
- **A transaction tried to SEIZE or PREEMPT its own facility** – сообщение пытается занять (SEIZE) или захватить (PREEMPT) уже занятое им устройство.
- **A transaction which owns a facility is attempting to terminate** – сообщение, которое владеет устройством, пытается выйти из системы.
- **A transaction which was preempted at a facility tried to SEIZE or PREEMPT it** – сообщение, обработка которого устройством была прервана, пытается занять или захватить его.
- **Arithmetic overflow** – арифметическое переполнение.
- **Assembly count was not positive** – счетчик объединения положительный.
- **Attempt to release an idle facility** – попытка освободить незанятое устройство.
- **Attempt to release an unwonted facility** – попытка освободить не свое устройство.
- **Attempt to release more storage than existed** – попытка освободить большую емкость многоканального устройства, чем определено.
- **Block Index is not positive** – индекс блока неположительный.
- **Block index is too big** – индекс блока слишком большой.
- **Block limit error out of blocks** – ошибка границы блока, выход из блоков.
- **Block limit Is too small** – граница блока слишком маленькая.
- **Circular reference in expressions** – циклическая зависимость в выражении.
- **Conjugate block is not a MATCH block** – парные блоки не являются MATCH блоками.
- **Count block ran out of entities** – счетчик блоков вышел за допустимые границы.
- **Edit from a program file is not allowed** – редактирование из программного файла не разрешается.
- **Entity number Is too high** – номер объекта слишком большой.
- **Error priming this or another GENERATE block** – ошибка определения блока GENERATE.
- **Entity is not a B VARIABLE** – объект не B VARIABLE.
- **Entity is not a VARIABLE** – объект не VARIABLE.
- **Error during evalution of PLOT argument** – ошибка при оценке аргумента PLOT.
- **Error in expression** – ошибка в выражении.
- **EXECUTE block cannot act on another EXECUTE block** – блок EXECUTE не может оказывать действия на другой EXECUTE блок.
- **Expression too long** – выражение слишком длинное.

- **First cumulative probability must be 0** – первая координата аргумента непрерывной случайной функции должна быть 0.

- **Illegal attempt to make queue entity content negative** – незаконная попытка создать очередь с отрицательным содержимым.

- **Illegal combination of operands in REMOVE block** – незаконная комбинация операндов в блоке REMOVE.

- **Illegal combination of operands in UNLINK block** – незаконная комбинация операндов в блоке UNLINK.

- **Illegal SNA class in COUNT or SELECT block** – неправильный тип СЧА в блоках COUNT или SELECT.

- **Internal error** – внутренняя ошибка.

- **Internal string error at location xxxx** – системная ошибка размещения строки xxxx.

- **Invalid action after savable program has been deleted** – недопустимое действие после удаления рабочей программы.

- **Invalid arithmetic expression** – недопустимое арифметическое выражение.

- **Invalid character** – недопустимый символ.

- **Invalid character in program file** – недопустимый символ в программном файле.

- **Invalid cumulative probability distribution** – недопустимое распределение вероятности.

- **Invalid discrete function argument** – недопустимый аргумент дискретной функции.

- **Invalid file specification** – недопустимая спецификация файла.

- **Invalid function follower statement** – недопустимое описание значений функции.

- **Invalid function statement** – недопустимый оператор функции.

- **Invalid (nonincreasing) X values** – недопустимое (невозрастающее значение x).

- **Invalid 0 or negative argument** – недопустимый нулевой или отрицательный аргумент.

- **L or M type functions cannot be random** – функции типа L или M не могут быть случайными.

- **Labels may not be keywords** – метки не могут быть СЧА или ключевыми словами.

- **Line number overflow. Please renumber** – превышен максимально допустимый номер строки программы. Пожалуйста, перенумеруйте.

- **List function argument is too large** – список аргументов функции слишком большой.

- **Matrix is too large (8K entries)** – матрица слишком большая.

- **Memory request is too big** – требуемая память слишком большая.

- **More than 50 items in a function** – более чем 50 элементов данных в функции.

- **Name has not been given a value** – имени не было дано значение.

- **Negative A operand in SPLIT block** – отрицательный операнд A в блоке SPLIT.

- **Negative storage request** – отрицательная запрашиваемая емкость многоканального устройства.
- **Negative time Increment** – отрицательный прирост времени.
- **Non-positive list function argument** – неположительный аргумент табличной функции.
- **Not saved** – не сохранено.
- **Not enough memory to store statement** – не достаточно памяти для записи оператора.
- **Operand B of QUEUE or DEPART is negative** – операнд В блоков QUEUE или DEPART отрицателен.
- **Out of memory. Press <SPACE> to stop, any other key uses program listing space** – выход из памяти. Нажмите <SPACE> для останова, любая другая клавиша использует программу листания пространства.
- **Overflow** – переполнение.
- **Parenthesis error in expressions** – ошибка в выражении, в скобках.
- **Program line not found** – строка программы не найдена.
- **Reference to a non-existent block** – ссылка на несуществующий блок.
- **Reference to a non-existent function** – ссылка на несуществующую функцию.
- **Reference to a non-existent matrix savevalue** – ссылка к несуществующей матрице сохраняемых величин.
- **Reference to a non-existent parameter** – ссылка на несуществующий параметр.
- **Reference to a non-existent savevalue** – ссылка к несуществующей сохраняемой величине.
- **Reference to a non-existent storage** – ссылка к несуществующему устройству.
- **Reference to a non-existent table** – ссылка к несуществующей таблице.
- **Reference to a non-existent variable entity** – ссылка на несуществующую переменную.
- **Remove option was not specified** – Remove - опция не определена.
- **Remove option was used with no destination** – опция использована не по назначению.
- **Result savevalue entity number must be positive** – номер сохраняемой величины должен быть положительным.
- **Storage request exceeds total capacity** – запрос емкости многоканального устройства превышает полную емкость.
- **System error. Code xxH** – системная ошибка. Код xxH.
- **The disk is full** – диск заполнен.
- **The file directory is full** – каталог файлов полон.
- **The matrix row number is too large** – номер строки матрицы слишком большой.
- **The matrix row number is not positive** – номер строки матрицы не положительный.
- **The matrix column number is too large** – номер столбца матрицы слишком большой.

- **The matrix column number is not positive** – номер столбца матрицы не положительный.
- **The matrix offset exceeds 8192 elements** – содержимое матрицы превышает 8192 элемента.
- **The program line is too long** – строка программы слишком длинная.
- **The required label is missing** – требуемая метка пропущена.
- **The simulation has been perturbed by an interaction. Use RESET and then rerun** – при моделировании нарушилось содержимое файла результатов. Используйте RESET и затем перезапустите.
- **The table has more then 8191 frequency classes** – таблица не может иметь более чем 8191 частотных классов.
- **The upper count limit is too low** – верхний предел счетчика слишком низкий.
- **There are no group members** – нет элементов групп.
- **There are no transactions on userchain entities** – нет сообщений в цепочках пользователя.
- **There are no transactions** – нет сообщений.
- **There is no positive treatment level** – обработка неположительного уровня.
- **There is no transaction for the parameter refence** – нет сообщения для упоминаемого параметра.
- **There is no transaction for this SNA evaluation** – нет сообщений для вычисления СЧА.
- **Use of nonpositive entity number** – использование неположительного номера объекта.
- **You cannot change a block location value in an EQU statement** – вы не можете изменить значение метки блока в операторе EQU.
- **You must specify an SNA** – вы должны определить СЧА.
- *******Formatting overflow******* – переполнение формата.
- *******Stack overflow******* – переполнение стека.
- **Other Error Codes** – другие коды ошибок.

Учебное издание

Симонова Елена Витальевна

Описание языка GPSS World

Учебное пособие

Печатается в авторской редакции

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ имени академика С. П. КОРОЛЕВА»
(Самарский университет)
443086, САМАРА, МОСКОВСКОЕ ШОССЕ, 34.

Изд-во Самарского университета.
443086 Самара, Московское шоссе, 34.