

Классификация моделей и видов моделирования

Математическое моделирование существенно расширяет возможности экономического анализа, позволяет сформулировать новые постановки задач, повышает качество принимаемых решений.

Мат. модели экономических производств называются эк.-математическими моделями (ЭММ).

1. По целевому назначению модели могут разделяться на:

- теоретико-аналитические – используются при анализе общих свойств и закономерностей экономических процессов (например, модель спроса и предложения);

- прикладные модели – используются для анализа конкретных экономических проблем.

2. По уровню исследуемых процессов:

- модели производственно-технологического уровня;

- модели социально-экономического уровня.

3. По характеру причинно-следственных связей:

- детерминированные;
- вероятностные (стахостические) – учитывается фактор случайности.

4. По характеру отражения фактора времени:

- статически – исследуют объект в какой-то момент времени;

- динамические.

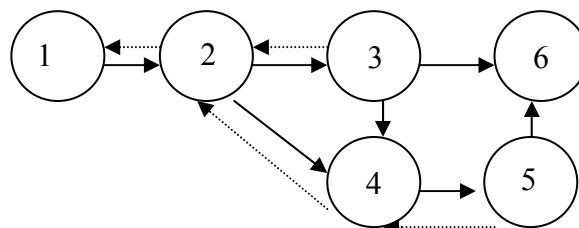
5. По виду математических зависимостей:

- линейные;
- нелинейные.

6. По степени детализации:

- агрегированные (макромодели);
- детализированные (микромодели).

Общая характеристика и этапы имитационного моделирования



Этапы экономико – математического моделирования:

1) Постановка экономической проблемы, ее качественный анализ. Главное на этом этапе – сформулировать сущность проблемы и определить те вопросы, на которые требуется дать ответы; сформулировать гипотезы, предварительно объясняющих поведение объекта.

2) Построение ММ – этап формализации экономической ситуации, ее выражение в виде математических соотношений. Обычно на этом этапе сначала определяются тип модели, а затем подбираются конкретные отдельные детали, параметры.

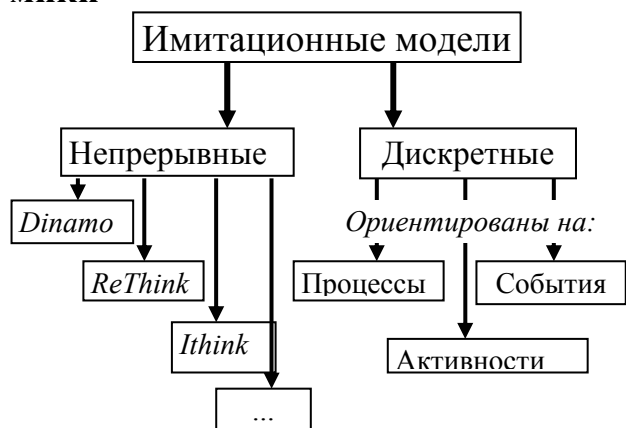
3) осуществляется выбор языка и средств программирования, типы компьютеров, расчет затрат ресурсов на составление и отладку программы.

4) осуществляется планирование и подготовка вычислительного эксперимента и выполняются многовариантные расчеты.

5) осуществляется сравнение результатов доступных экспериментов на объекте моделирования и вычислительных экспериментов.

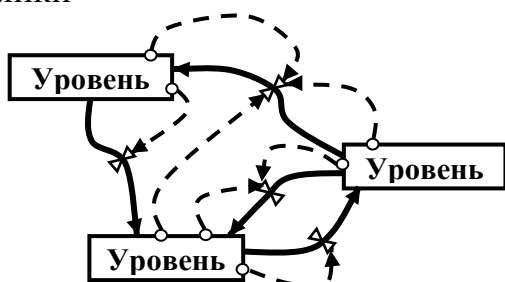
6) Анализ результатов и их применения. На этом этапе проверяется правильность, полнота и степень практической применимости полученных результатов.

Инструментальные средства моделирования: модели системной динамики



Информационная система с обратной связью существует там, где окружающая среда приводит к принятию решения, вызывающего действие, которое само влияет на окружающую среду и, значит, на дальнейшие решения.

Базовая структура модели системной динамики



⊗ *Функция решения* — *Канал потока*
 - - - *Источник информации*

Модель системной динамики включает четыре составляющие:

- уровни;
- потоки, перемещающие содержимое от одного уровня к другому;
- функции решений (изображены в виде вентилей), которые регулируют темпы потока между уровнями;
- каналы информации, соединяющие пункты решений с уровнями.

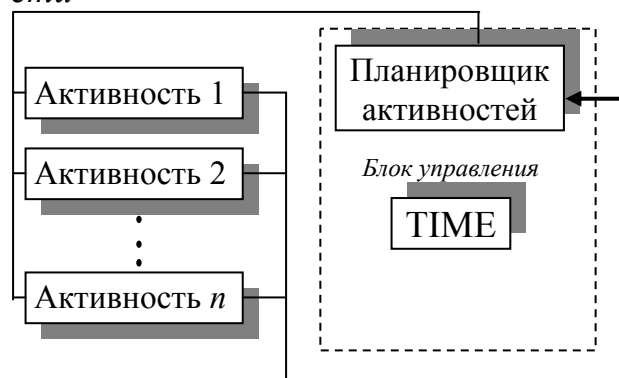
Дискретные системы имитационного моделирования ориентированы на:

- *активности*

- *процессы*
- *события*

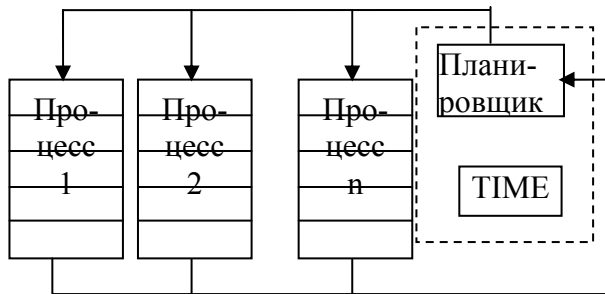
Первый вид дискретных имитационных моделей – системы, ориентированные на активности (CSL), **второй вид** – системы, ориентированные на процессы (GPSS), **третий** – системы, ориентированные на события (SMPL).

Системы, ориентированные на активности



Инструментальные средства моделирования: системы, ориентированные на процессы (GPSS).

Расписываются все типовые процессы в виде активностей, условий их возникновения, пассивностей и т.д. Каждый процесс представляет собой независимый программный сегмент, который управляется системой управления. Обслуживание этих программных модулей осуществляется в квазипараллельном режиме, через условные кванты времени. В одном процессе одновременно на обслуживании может находиться целый ряд заявок.



Инструментальные средства моделирования: системы, ориентированные на события (SMPL).

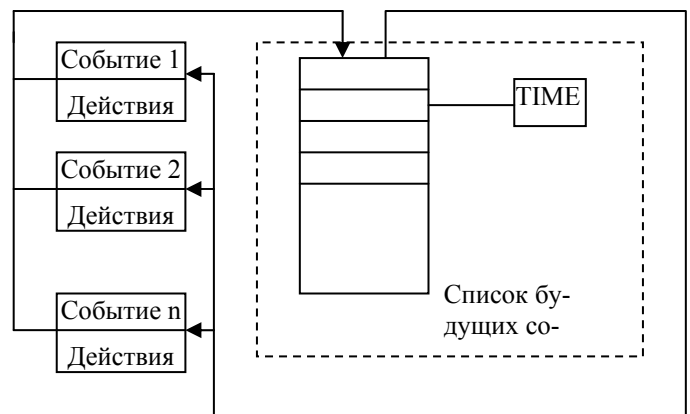
Система описывается как совокупность событий, которая изменяет ее состояние (поступление заявки в систему, вход, выход, начало обслуживания и т.д.).

Каждое событие должно иметь ряд характеристик:

- 1) действия, связанные с этими событиями;
- 2) логические условия взаимосвязи событий;
- 3) время поступления событий и т.д.

Имитационная модель представляет собой набор программных модулей, каждый из которых состоит из целого набора операторов, отражающих выполняемые действия.

В каждом событии предусмотрено планирование будущих событий. В результате планирования событий образуются списки будущих событий, элементы которого упорядочены во времени. В вершине списка находится ближайшее по времени событие.



Основные и вспомогательные события

События характеризуют изменение состояния системы под действием заявок.

В период времени от предыдущего до последующего события состояние системы считается неизменным.

В рассматриваемой системе событиями являются: прибытие заявки, начало и завершение обслуживания.

События, возникающие в системе, подразделяются на основные и вспомогательные.

Основными являются события, которые можно запланировать заранее, независимо от текущего состояния системы.

Вспомогательными называют события, не являющиеся основными.

Рассмотрим процесс планирования основного события – поступление заявки в систему.

Как правило, планирование прихода последующей заявки осуществляется в момент поступления предыдущей.

Временная диаграмма



Основное событие – приход 1-й заявки и окончание обслуживания.

Вспомогательное – начало обслуживания, которое зависит от текущего состояния очереди.

Отдельно необходимо запланировать момент поступления первой заявки в систему.

Время свершения основного события «завершение обслуживания» определяется в момент «начала обслуживания» в соответствии с заданной длительностью обслуживания.

Предварительное планирование времени поступления на обслуживание отсутствует и определяется в процессе функционирования модели.

Таким образом:

1) приход заявки вызывает следующие действия:

- планирование следующего прихода (следующей заявки);

- проверка состояния обслуживающего прибора. Если прибор свободен, то: необходимо перевести обслуживающий прибор из свободного состояния в занятое; запланировать событие окончания обслуживания.

Если прибор занят, то заявка помещается в очередь.

2) окончание обслуживания вызывает следующие действия:

проверка состояния очереди. Если в очереди есть заявка, ожидающая обслуживания, то удаляем заявку из вершины очереди и планируем окончание обслуживания. Если очереди нет, то перевод прибора из занятого состояния в свободное.

Основным событием является также событие завершения моделирования. Оно планируется заранее и с его помощью исследователь ограничивает длительность функционирования исследуемой системы. События в модели предполагают выполнение 2 действий:

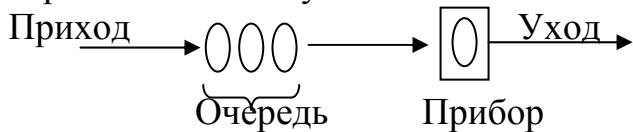
1) Планирование событий.

Для каждого события необходимо указать номер события, номер заявки, время поступления этого события. Благодаря операции планирования, мы формируем список будущих событий (динамический список).

2) **Выполнение событий**, т.е. переход к соответствующему программному модулю, который описывает действия, связанные с данным событием.

Основные характеристики простейшей СМО

Она характеризуется тем, что существует некоторый прибор и заявки, которые на нем обслуживаются.



Система с одним прибором и очередью к нему

В качестве процесса обслуживания могут быть представлены различные по своей физической природе процессы функционирования экономических, производственных, технических и других схем (поток поставок продукции в магазины, сырья предприятию и т.д.).

Характерным для работы таких объектов являются случайные появления требований (заявок) на обслуживание и завершение обслуживания в случайные моменты времени, т.е. стохастический характер процессов их функционирования.

ПСМО с одним прибором и очередью представлена на рисунке. Эта система характеризуется двумя независимыми случайными переменными:

- интервалы времени между последовательными моментами прибытия заявок в систему (интервал прибытия);
- время, требуемое прибору для выполнения обслуживания.

Распределение системных величин, зависящими от этих двух, является предметом исследования. Зависимые переменные:

1. Число заявок, прибывших на обслуживание за заданный промежуток времени;
2. Число заявок, попавших на обслуживание сразу же по прибытии;
3. Среднее время пребывания заявок в очереди;
4. Нагрузка прибора;
5. Средняя длина очереди;
6. Максимальная длина очереди.

Рассмотрим основные предпосылки построения простейшей имитационной модели:

- 1) Случайная переменная – интервал прибытия – является равномерно распределенной и принимает только целые значения;
- 2) Время обслуживания также предполагается равномерно распределенной случайной величиной с целыми значениями;
- 3) Есть возможность обращения к генератору равномерно распределенных случайных чисел, т.е. при обращении он выдает значения с 6-ю знаками после ‘,’;
- 4) Все заявки остаются в системе независимо от длины очереди;
- 5) В начале моделирования система свободна;
- 6) Моделирование завершается в заранее определенный момент модельного времени;
- 7) Фиксируется некоторая статистическая информация.

Составляющие имитационной модели: заявки, события, активность, процесс

Заявки – некоторые объекты, которые вызывают ряд действий в имитационной модели. Заявки появляются из внешней среды и являются ее основным атрибутом. **Источники заявок** находится во внешней среде и представляется в виде *генераторов* заявок (мы можем задать простой (жесткий) интервал поступления или задать любой закон).

Заявки характеризуются **типом**. Заявки одного типа вызывают аналогичные действия в системе.

Другой важной характеристикой является **приоритет**, который характеризует степень ее важности. Приоритеты бывают относительные, абсолютные и динамические. При относительном приоритете при поступлении заявки обслуживание текущей заявки не прерывается, а поступившая заявка с более высоким приоритетом помещаются в вершину очереди. При абсолютном приоритете возможно прерывание в обслуживании текущей заявки. При динамическом (меняющее во времени) – заявка с течением времени имеет различный приоритет.

Действия с заявками:

- 1) Генерирование заявок, при этом, как правило, должны быть указаны следующие атрибуты: номер или имя, тип заявки, количество заявок данного типа, приоритет заявки, время появления первой заявки и интервал поступления заявок.
- 2) Изменение параметров заявки в ходе моделирования;
- 3) Уничтожение заявки.

Движение заявки в системе представляет собой процесс чередования активностей, т.е. периодов времени, когда заявки обслуживаются приборами. В отличие от активностей пассивность

приостанавливает процесс движения заявок и, как правило, не имеет заранее заданного времени выполнения.

Процесс – движение заявки в виде появления, чередования событий и активностей, пассивностей или событий и уничтожения заявки.

Составляющие имитационной модели: средства и очереди

Средство (приборы, устройство) – условная единица, которая связана с активностью, т.е. некоторый станок, прибор, оператор, кассир и т.д.

Средство характеризуется рядом параметров, временем реализации активности и своим типом, т.е. может является простым и сложным многоканальным и параллельно обрабатывать несколько событий.

Действия, выполняемые средствами:

1. захват или включение (резервирование) средства
2. проверка состояния средства (занятое или свободное)
3. освобождение средства.

Очередь – следующая составляющая имитационной модели.

Очереди возникают из-за конфликтов при параллельной поступлений требований на один и тот же ресурс, при этом заявки помещаются в очереди, где они находятся в пассивном состоянии.

Очереди характеризуются входом и выходом, т.е. существуют потоки входа и необходимо предусмотреть способы организации очереди; и потоки выхода, т.е. необходимо предусмотреть организацию обслуживания заявок.

Теория массового обслуживания доказывает, что среднее время в очереди будет минимальным, если в первую очередь обслуживаться будут заявки с минимальным временем обслуживания. (Shortest input output (SIO)).

Кроме FIFO и LIFO, существуют другие системы приоритетов – прибывшая заявка помещается последней в группу заявок с тем же приоритетом.

Действия с очередями:

- поступившие заявки в очередь;
- удаление заявки из очереди;
- проверка состояния очереди.

Составляющие имитационной модели: время

Время – еще одна составляющая имитационного периода.

Как правило, значение времени является целочисленным, поэтому они должны быть достаточно малыми, по сравнению с реальными промежутками времени, протекающими в моделируемой системе:

$$TIME_{i+1} = TIME_i + \Delta$$

Два способа приращения времени: с постоянным и переменным шагом:

1) если $\Delta = \text{const}$ при увеличении таймера на единицу необходимо проверить есть ли в системе события, которые происходят в данный момент времени. Если 'да', то осуществляется выполнение действий, связанных с этими событиями. Если 'нет', снова увеличением значением таймера на единицу и т.д.

2) если выбрана стратегия с переменным шагом, то время в модели меняется скачком от предыдущего к последующему.

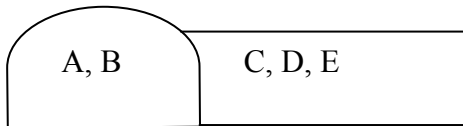
GPRS: генерация и уничтожение заявки

Модели систем на GPSS могут быть записаны в виде блок-схем или программ, эквивалентных блок-схемам. Каждый блок соответствует одному оператору, характеризуется глаголом, отражающим основное назначение блока.

Блоки содержат различные операнды, которые задают информацию, характеризующую этот блок. Максимальное число операндов равно 7, они обозначаются буквами A, B, C, D, E, F, G.

Каждому блоку может быть присвоено символическое имя, которое содержит не более 5 символов, первые три из которых должны быть алфавитными.

GENERATE A, B, C, D, E



A – средний интервал времени поступления заявки, по умолчанию 0.

B – половина поля допуска равномерно распределенного интервала, по умолчанию 0.

GENERATE 10, 2 {заявки приходят в систему от 8 до 12}

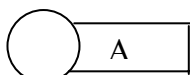
C - первоначальное смещение интервалов времени прихода первой заявки, по умолчанию отсутствует.

D – ограничитель транзакций, поступающих в модель из этого блока, по умолчанию ∞.

E – уровни приоритета заявок поступающих через заданный блок, по умолчанию 0, а всего максимально возможный приоритет 127.

Например, GENERATE 8, 4, 10, 2.

Блок TERMINATE A



Блок удаления транзакты из модели. Транзакты, попадающие в этот блок,

покидают модель.....

Обычно используются без операнда, т.е. GENERATE...

....

TERMINATE

На значение операнда A уменьшается счетчик завершений. Количество прогонов данной программы (или завершений) задаются оператором START при ее запуске.

START [A]

например, 5

если поставим в TERMINATE 1, значение START уменьшается на 1.

Для того, чтобы завершить моделирование:

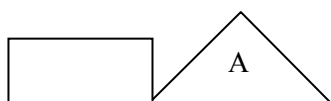
GANERATE 48 00

TERMINATE 1

а START 1 из него «-» 1 и моделирование прервется.

Блоки GPSS, связанные с приборами

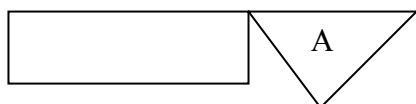
Блок SEIZE A



Блок SEIZE A (занять), имеет один операнд. В данном блоке моделируется занятие прибора, имя которого указано в операнде А.

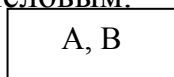
Если прибор уже используется, то транзакт не может войти в данный блок. Если же прибор свободен, блок SEIZE моделирует занятие прибора, т.е. вызывается подпрограмма, которая изменяет статус прибора и переводит его из свободного в состояние занятое.

Окончание - блок RELEASE A



Освобождает прибор, изменяя его статус из занятого переводит в свободное. Блок RELEASE никогда не запрещает вход транзактам. Имя прибора может быть символьным или числовым.

Блок ADVANCE A, B



- задержать

Предназначен для задержки продвижения транзактов в течение некоторого времени, операнды А и В аналогичны в GENERATE:

А – среднее значение интервала

В – половина поля допуска равномерно распределенного интервала

Модель работы одного станка:

1 сегмент { GENERATE 10, 2
SEIZE STAN
ADVANCE 9, 3
RELEASE STAN
TERMINATE

2 сегмент { GENERATE 4800
TERMINATE 1

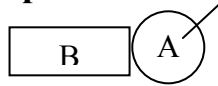
В явном виде очередь не описана

В процессе моделирования транзакты входят в модель в определенные моменты времени через блок GENERATE. И транзакт начинает движение от блока к блоку в соответствии с описанием своего процесса. В момент входа в блок выполняется соответствующая подпрограмма и транзакт пытается войти в следующий блок. Такое продвижение происходит до тех пор, пока:

1. Транзакт входит в блок, функцией которого является задержка движения.
2. Транзакт попадает в блок, удаляющий его из модели.
3. Транзакт пытается войти в следующий блок, который отказывается его принять. Тогда транзакт остается в текущем блоке и попытается войти в следующий блок при изменении условий функционирования моделей.

Блоки GPSS, обеспечивающие сбор статистики об ожидании

Блоки, связанные с очередями ЛОК QUEUE



Операнд А – имя очереди, может быть цифра, символ;

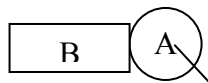
В – по умолчанию 1. Указывает, сколько единиц поступило в очередь.

При поступлении в очередь:

При входе транзакта в блок QUEUE соответствующей подпрограммой выполняется 4 действия:

1. Счетчик входов для данной очереди увеличивается на 1.
2. Счетчик текущего содержимого очереди увеличивается на 1.
3. Для данного транзакта запоминается (? Катя) имя очереди, в которой он находится .
4. Запоминается момент модельного времени постановки в очередь данного транзакта.

БЛОК DEPART



Покинуть очередь, А и В те же.

Вызывается подпрограмма, которая выполняет следующие действия:

1. Текущее содержимое увеличивается на 1.
2. Определить время пребывания транзакта в очереди.
3. Если время пребывания в очереди нулевое, то увеличивается счетчик нулевых входов.
4. Транзакт исключается из очереди.

Пример,

```
с  
е  
г  
м  
е  
н  
т  
(!) {  
    GENERATE 10,4  
        QUEUE NAK  
  
    TERMINATE 1  
    SEIZE STAN  
    DEPART NAK  
    ADVANCE 12,6  
    RELESE STAN  
    GENERATE 4800  
    TERMINATE
```

В процессе моделирования собирается следующая статистическая информация об ожидании:

- номер или имя очереди (QUEUE);
- максимальная длина очереди (MAX);
- текущая длина очереди (CONT);
- общее количество входов (ENTRY);
- количество «нулевых» входов (ENTRY (0));
- средняя длина очереди (AVE. CONT.);
- среднее время пребывания транзактов в очереди (AVE. TIME);
- среднее время пребывания транзактов в очереди без учета «нулевых» входов (AVE (-0));
- количество транзактов, ожидающих специальных условий (RETRY).

Блоки, связанные с устройствами

Статистику работы устройства в процессе моделирования можно наблюдать в окне устройств для GPSS/PC (перейти в это окно можно с помощью клавиш [ALT+TAB]) или в окне Facilities Window для GPSS World.

В процессе моделирования в файле стандартной статистики собирается следующая информация об используемых устройствах:

- номер или имя устройства (FACILITY);
- количество входов (ENTRIES);
- коэффициент использования (UTIL);
- среднее время пребывания транзакта в устройстве (AVE. TIME);
- состояние готовности (AVAIL.);
- номер последнего транзакта, занявшего устройство (OWNER);
- количество прерванных в устройстве транзактов (PEND);
- количество прерывающих устройство транзактов (INTER);
- количество транзактов, ожидающих специальных условий (RETRY);
- количество транзактов, ожидающих занятия устройства (DELAY).

Блоки, связанные с многоканальными устройствами

В процессе моделирования в файл стандартной статистики собирается следующая информация о МКУ:

- номер или имя МКУ (STORAGE);
- емкость МКУ (CAP);
- количество единиц свободной емкости МКУ в конце моделирования (REMAIN);
- минимальное количество свободной емкости МКУ за период моделирования (MIN);
- максимальное количество свободной емкости МКУ за период моделирования (MAX);

- количество входов (ENTRIES);
- состояние готовности (AVAIL.);
- среднее значение занятой емкости за период моделирования (AVE.C);
- коэффициент использования (UTIL);
- количество транзактов, ожидающих специальных условий (RETRY);
- количество транзактов, ожидающих занятия устройства (DELAY).

Статистику работы МКУ в процессе моделирования можно наблюдать в окне МКУ для GPSS/PC (перейти в это окно можно с помощью клавиш [ALT+S]) или в окне Storage Window для GPSS World.

Логика работы GPSS

GPSS рассматривает каждый транзакт как элемент (звено) одной или нескольких цепей. Цепи являются открытыми и имеют начало и конец. Каждый транзакт занимает определенное положение относительно начала цепи.

Существуют цепи:

- Текущих событий;
- Будущих событий;
- Пользователя;
- Прерываний;
- Парности.

Последних трех цепей может быть несколько.

Цепь текущих событий состоит из тех транзактов, для которых планируется их продвижение в одном или нескольких блоках в течение текущего значения модельного времени.

Кроме того, в неё входят транзакты, движение которых заблокировано из-за выполнения определенных условий.

Цепь будущих событий состоит из транзактов, движение которых не планируется до наступления некоторого времени:

- Транзакт попал в блок ADVANCE.
- Транзакт должен войти в более поздний момент времени через блок GENERATE.

GPSS изменяет состояние модели при просмотре цепи текущих событий от начала к концу, транзакт за транзактом.

При анализе каждого транзакта GPSS выбирает его и двигает в модели до тех пор, пока:

Транзакт входит в блок ADVANCE, где он должен пробыть некоторое время.

1) В этом случае транзакт отправляется в цепь будущих событий и помещается в ту позицию, которая соответствует времени выхода транзакта из цепи в следующий блок.

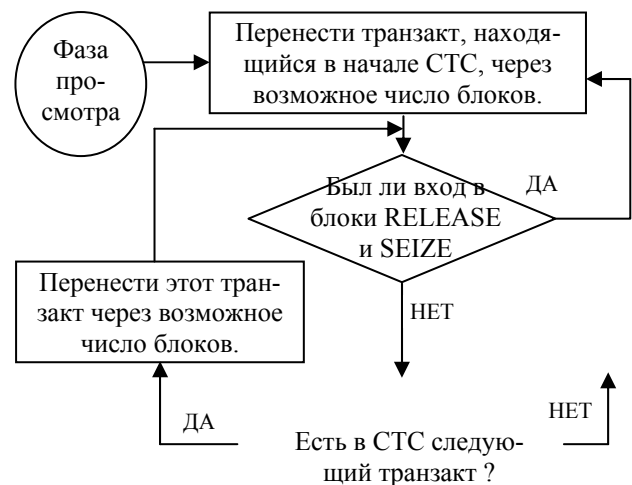
2) Возникает условие блокировки,

препятствующее входу транзакта в следующий блок. Транзакт остается в цепи текущих событий в этом же блоке.

3) Транзакт входит в блок TERMINATE и удаляется из модели.

Когда транзакт прекращает движение, выполняется одно из следующих действий:

- Продолжается просмотр цепи текущих событий и выбирается следующий транзакт, который пытается продвинуться в модели;
- Без продвижения таймера начинается просмотр цепи текущих событий от начала, т.е. осуществляется возврат к началу цепи, выбор первого транзакта и движение его по модели.



Просмотр от начала цепи начинается при выполнении определенных условий, чаще всего при прохождении блоков RELEASE, SEIZE.

В первом случае в модели могут быть заблокированы транзакты в ожидании освобождения прибора, во втором – в ожидании занятия прибора.

Пусть обработан последний транзакт, находящийся в цепи текущих событий, тогда осуществляется переход к цепи будущих событий.

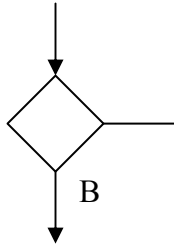
Таймер модельного времени принимает значение, равное времени, запланированному для движения следующего транзакта. Этот транзакт переносится из цепи будущих событий вместе с другими, движение которых может быть возобновлено в дан-

ный момент модельного времени.
При наличии временных узлов каждый входящий транзакт располагается как последний элемент внутри своего приоритетного класса. После окончания переноса начинается новый просмотр цепи текущих событий.



Система GPSS: режимы использования блока TRANSFER

1. Режим безусловной передачи. В этом случае не используется операнд А, используется только операнд В, который представляет собой имя блока, в который передается транзакт.

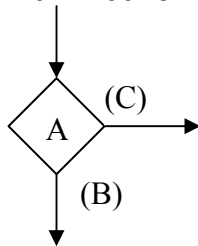


Блок TRANSFER не запрещает вход транзакта.

TRANSFER, ALT

Если блок, имя которого указано в операнде В, запрещает вход транзакта, то транзакт относится в блок TRANSFER.

2. Режим статистической передачи.



А – частота передачи транзакта в блок С.

В – имя блока В.

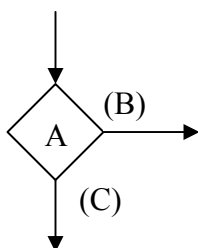
С – имя блока С.

TRANSFER .25, WORK, PLAY

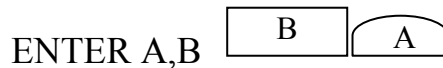
25% транзактов попадают (входят) в блок PLAY, а 75% - в блок с именем WORK.

3) использование блока TRANSFER BOTH

TRANSFER BOTH, В, С



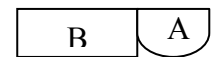
Блоки GPSS, связанные с моделированием работы многоканальных устройств



ENTER A, В Резервирование В - каналов (по умолчанию 1) многоканального устройства с именем А.

При входе в блок ENTER выполняются следующие действия:

1. Счетчик входов увеличивается на В.
2. Текущее содержимое увеличивается на В.
3. Доступная емкость увеличивается на В. Освободить многоканальное устройство – блок LEAVE.



Освободить В - каналов устройства с именем А.

Действия: 1. Текущее содержимое уменьшается на В.

2. Доступная емкость увеличивается на В единиц.

Пример, имеется одна единица оборудования, но заявки поступают с различными приоритетами. Если очередь общая, то в ее вершине будут находиться заявки с более высоким приоритетом.

```
GENERATE 10,2,,,1    GENERATE 8,4
QUEUE NAK           QUEUE NAK
SEIZE STAN          SEIZE STAN
DEPART NAK          DEPART NAK
ADVANCE 12,6        ADVANCE 10,4
RELEASE STAN        RELEASE STAN
TERMINATE           TERMINATE
```

```
GENERATE 4800
TERMINATE 1
```

Определение емкости многоканальных устройств

STORAGE \$\$ имя1, C1/\$\$ имя2, C2...

\$ - ставится, если имя начинается с буквы, если имя цифровое, то \$ не ставится.

C1, C2 – емкость многоканального устройства.

Логические особенности генератора случайных чисел

Пусть целью является генерация четырехзначных случайных чисел в интервале 0,0000 ... 0,9999. Для этого используется два нечетных целых числа, каждое из которых содержит до четырех цифр. Одно из них – ядро – оно остается неизменным, а значения второго числа – множителя – меняется. допустим, ядро – число 5167, а начальное значение множителя 3729.

Множитель	Произведение	Случайное число
3729	19267743	0,2677
8081	40008081	0,0080
4527	41754527	0,7545

Числа будут нечетными, чтобы мы не пришли к 0, иначе придется остановиться

Количество случайных чисел, полученное до того, как они начнут повторяться, называется периодом или длиной генератора.

Для увеличения периода следует увеличить больше знаков ядра и множителя.

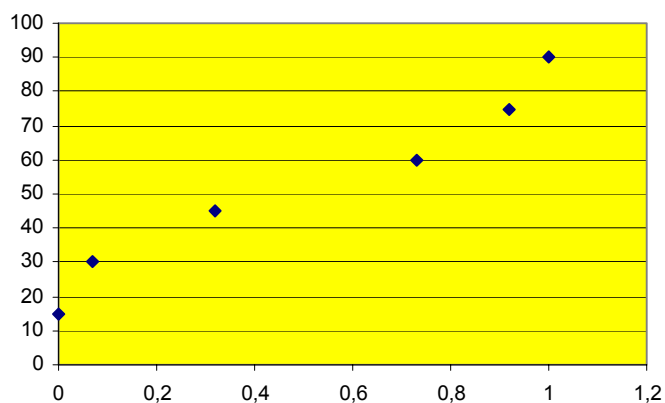
Непрерывные распределения

При обращении к непрерывной функции также разыгрывается случайное число, которое далее используется как аргумент функции.

Пусть, например, некоторая случайная переменная X распределена равномерно и является непрерывной в интервале от 2 до 6 (6 не входит в интервал). Тогда:

$$P(X < 2) = 0, P(X < 6) = 1.$$

Её график имеет вид:



Если, например, значения интервалов времени распределены равномерно в диапазоне от 200 до 375 включительно, то соответствующая функция будет иметь вид:

FAST FUNCTION RN5,C2

0,200/1,376

Статистические данные могут быть использованы в непрерывных функциях для аппроксимации. Пусть в результате обследования работы банка получены:

Интервал времени обслуживания, с	Относительная частота	Суммарная частота
Менее 15	0,00	0,00
15 ÷ 30 -	0,07	0,07
30 ÷ 45 -	0,25	0,32
45 ÷ 60 -	0,41	0,73
60 ÷ 75 -	0,19	0,92
75 ÷ 90 -	0,08	1,00

Дискретные неравномерные распределения

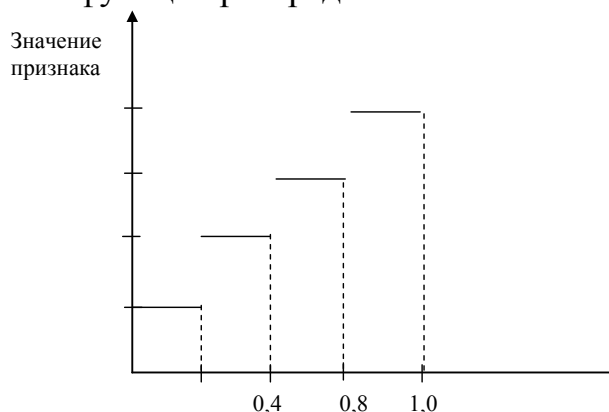
Случайная переменная является дискретной, если множество её значений конечно.

Дискретная случайная переменная может принимать как целые (число точек на игральной кости), так и дробные значения (цена товара).

Предположим, что случайная переменная может получать значения, представленные в таблице

Интервалы, час	Относительная частота	Суммарная частота
2	0,15	0,15
5	0,20	0,35
8	0,25	0,60
9	0,22	0,92
12	0,12	1

Графическая интерпретация интегральной функции распределения



Для определения дискретной функции распределения ей должно быть присвоено имя, а также необходимо задать:

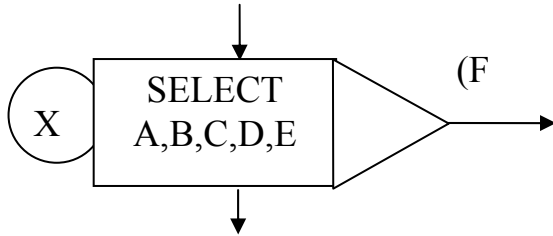
- аргумент функции, который определяет источник случайных чисел, используемых для розыгрыша в соответствии с распределением, заданным функцией;
- число различных значений, которые может принимать случайная переменная;
- сами значения переменной и соответствующие суммарные частоты.

Функция должна быть определена в блоке инициализации и может иметь символьное или цифровое имя:

```
5 FUNCTION RN4,D5  
.05,6/.3,7/.7,8/.95,9/1,10
```


Система GPSS: выбор элементов модели по их состоянию

Блок SELECT позволяет оперативно определить состояние указанных объектов и выбрать тот объект, который удовлетворяет некоторому условию.



Е – групповое имя стандартного числового атрибута;

С и В – наибольший и наименьший номера из множества членов просматриваемой группы;

Д – значение, с которым сравнивается значение, указанное в операнде Е;

Х – оператор сравнения, принимающий следующие значения: Е, G, L, GE, LE;

А – номер параметра, в котором записывается номер члена группы, отвечающего установленному условию;

Ф – необязательный операнд, указывающий имя блока, в который передается транзакт, если ни один член группы не отвечает установленному условию.

Например, проверяем состояние трех устройств, имеющих в системе.

Если все устройства заняты, то транзакт встанет в очередь с минимальной длиной:

Если есть свободное устройство, оператору присваивается номер устройства, и оно будет занято транзактом.

SELECT E 7,1,3,0,F

Просматриваются приборы 1,2, и 3, для определения, существует ли среди них свободный, для которого F=0.

Если просмотр закончен успешно, то транзакт выйдет из блока со значением параметра 7, равным номеру прибора (1, 2, 3).

Если поиск был безуспешным, то 7 па-

раметр равен 0 и транзакт попытается войти в следующий блок

SELECT L P3,5,8,250,SR

Просматриваются многоканальные устройства 5,6,7,8 для определения, который из них имеет нагрузку менее .250.

Номер соответствующего многоканального устройства записывается в P3 (или 0).

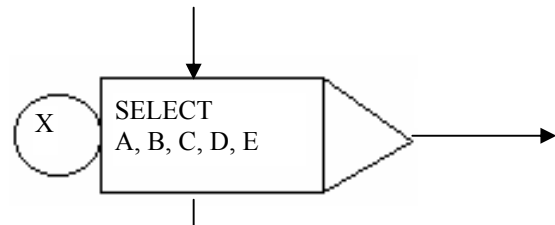
SELECT GE 5,13,17,3,QT,EXIT2

Просматриваются очереди с номерами с 13 по 17 для определения той, в которой среднее время пребывания 3.

Если такая очередь есть, её номер записывается в пятый параметр транзакта, который переходит в следующий блок

Если нет, то транзакт переходит в блок EXIT2.

Блок SELECT в режимах MIN и MAX



Оператор сравнения X имеет два значения - MIN и MAX, операнды А,В,С,,Е аналогичны рассмотренным.

SELECT MIN 7,9,13,,Q

Арифметические переменные и сохраняемые величины в GPSS

Арифметические переменные

Это стандартные числовые атрибуты, ссылкой на которые является либо Vj, либо V\$имя.

Над сохраняемыми величинами выполняются следующие арифметические операции:

*, / и @ - действия, имеющие высший приоритет.

+ и - - действия, имеющие низший приоритет.

Символ @ обозначает деление по модулю (остаток): $16@5=1$

Оператор определения арифметической переменной имеет вид:

Имя VARIABLE операнд

LQF VARIABLE R5+S5

Все данные в арифметических выражениях являются целыми, для действительной переменной определяется:

Имя FVARIABLE операнд

Сохраняемые величины

Они могут быть полусловными (от -32768 до +32768) и полнословными (от -2147483648 до +2147483648).

Ссылки на полусловные величины имеют вид: XHj (XH\$ имя), на полнословные - Xj (X\$имя).

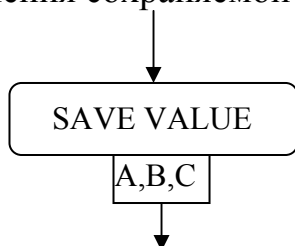
Ненулевые значения сохраняемых величин задаются оператором:

INITIAL имя1, величина1/.../имя n, величина n

INITIAL X\$T, 100/XH2,3

Блок SAVEVALUE

При входе транзакта в блок изменяются значения сохраняемой величины.



A - номер или сохраняемое имя изменяемой сохраняемой величины;

B - величина, используемая в процессе

модификации;

C - для полусловной величины - H, по умолчанию - полнословная величина.

SAVEVALUE P5,V\$ALFA

Вычисляем значение ALFA, а полученный результат назначается полнословной сохраняемой величине, номер которой записан в P5.

SAVEVALUE DAV-,FN\$HOLD,H

Величина XH\$DAV будет уменьшена на значение функции HOLD.

Логические переключатели

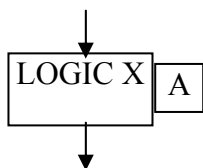
Это элементы GPSS, модифицирующие управление. Есть 2 положения: «установлено» и «сброшено».

Обычно перед началом моделирования логические переключатели сброшены.

Если необходимо установить переключатель используется оператор INITIAL LSj или

INITIAL LS\$имя_переключателя.

Блок LOGIC используется для изменения значения логических переключателей.

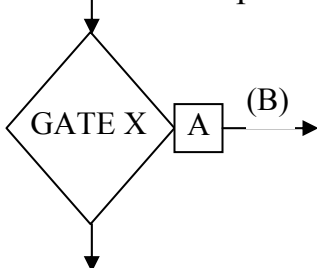


A - имя логического переключателя;

X - имеет 3 значения

$$X = \begin{cases} R - \text{сброс,} \\ S - \text{установка,} \\ I - \text{инвертировать.} \end{cases}$$

Блок GATE позволяет проверить состояние переключателя A.



A - имя переключателя;

X - оператор сравнения.

$$X = \begin{cases} LS - \text{проверка истинна, если} \\ \text{переключатель установлен;} \\ LR - \text{проверка истинна, если} \\ \text{переключатель сброшен.} \end{cases}$$

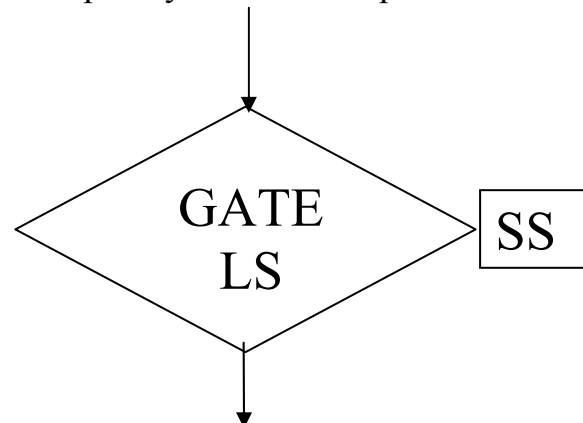
GATE LR SS

B - это необязательный операнд. Если B не используется, то проверка осуществляется в режиме отказа: если проверка не будет истинной, то клапан закрыт и транзакт удерживается в блоке, предшествующем блоку GATE. Этот транзакт будет считаться пассивным,

его внутренний флаг включен и он не будет продвигаться при последующих просмотрах цепи текущих событий.

При продолжении моделирования и изменении состояния флага логического переключателя транзакт переводится в активное состояние (его флаг «выключается»).

Если B используется, то проверка производится в режиме условной передачи: если результат проверки - истина, то транзакт переходит в следующий блок, или «ложь», то транзакт переходит в блок, имя которого указано в операнде B.



Блок закрыт, если переключатель SS «сброшен»

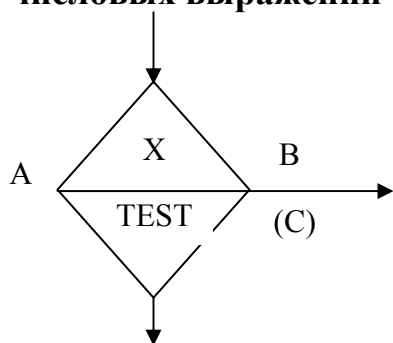
Логические операторы (булевы переменные) в GPSS

1. LS - логический переключатель установлен;
2. LR - логический переключатель сброшен;
3. FU (F) - прибор используется;
4. FNU - прибор не используется;
5. SF - многоканальное устройство заполнено;
6. SNF - многоканальное устройство не заполнено;
7. SE - многоканальное устройство пусто;
8. SNE - многоканальное устройство не пусто.

SNE \$ SUN - истина, если многоканальное устройство SUN не пусто.

Блок TEST X A,B,C

Система GPSS: изменение уровня приоритета транзактов, проверка числовых выражений



А – имя первого стандартного числового атрибута (СЧА);

В – имя второго СЧА;

Х – оператор сравнения (G, GE, L, LE, E);

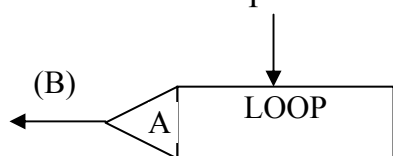
С – необязательный операнд, содержит имя блока, куда приходят транзакты, если результат сравнения ложь.

Если операнд С отсутствует, то блок работает в режиме отказа.

Например, TEST G Q1,Q2, PAS.

Если длина очереди₁ > длины очереди₂, то проходим дальше, иначе – идем в PAS.

Блок LOOP – организовать цикл.



А - № параметра,

В – символическое имя.

Выполняются следующие действия:

1) Величина указанного параметра уменьшается на 1.

2) Вычисляемая величина проверяется на =0.

3) Если значение не равно 0, то переход на блок В, иначе – на следующий блок.

ASSIGN 1, FN\$HOLD

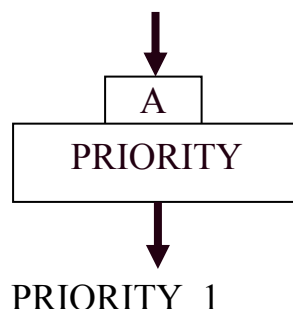
ADVANCE 4,3 NEXT

LOOP 1, NEXT

Блок PRIORITY

При выполнении параллельных процессов в системе возникают конфликты из-за общих ресурсов. Простейший способ их разрешения – назначить приоритеты транзактов.

Начальный приоритет может быть задан в блоке GENERATE с помощью операнда Е. В дальнейшем, если необходимо изменить приоритет транзакта, используется блок PRIORITY



PRIORITY 1

Использование таблиц в GPSS

Наблюдая в процессе моделирования за случайными величинами, мы получаем некоторую совокупность значений или выборку.

Основными характеристиками выборки являются:

- Среднее значение;
- Стандартное отклонение;
- Относительная частота, с которой элементы выборки попадают в тот или иной диапазон значений или частотный интервал.

Для получения этих характеристик используются таблицы.

Оператор TABLE. В нем задаются имя и характеристики таблицы.

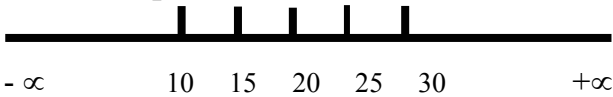
Имя_таблицы TABLE A,B,C,D

A – имя переменной, значения которой должно учитываться в таблице;

B – первое граничное значение;

C – ширина промежуточного интервала;

D – общее число интервалов, включая левый и правый.



ТУМ TABLE P3,10,5,6

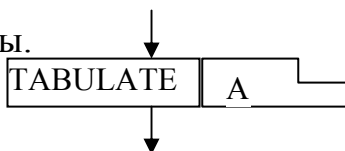
MAD TABLE X\$MAD,0,1,10

Блок TABULATE.

Значения выборки попадают в таблицу, когда транзакты входят в блок TABULATE.

TABULATE ТУМ

таблицы. A – имя



QTABLE – режим.

Этот режим используется для оценки распределения времени пребывания в очереди.

имя_таблицы QTABLE имя_очереди, B,C,D.

B – величина интервала...

Для очереди с именем LINE:

INQUE QTABLE LINE, 300, 200, 7.

Стандартные числовые атрибуты таблиц. Некоторые из компонентов таблиц являются СЧА. Это означает, что часть информации из таблиц может быть использована для управления логикой работы модели.

Имя	Значение
ТВ _j или ТВ\$	Среднее значение элементов таблицы
ТС _j или ТС\$	Число учтенных в таблице элементов
ТD _j или TD\$	Стандартное отклонение элементов таблицы

Модель автобусной остановки

Постановка задачи. По расписанию автобус должен приходить на остановку каждые 30 минут, однако возможное опоздание составляет 1,5 1,5 мин. В автобусе в момент прибытия может находиться от 20 до 50 пассажиров.

Приход людей подчиняется закону Пуассона с интенсивностью 12 человек каждые 30 мин. После того, как от 3 до 7 человек выйдут (равномерное распределение), в автобус входят столько людей, сколько возможно. Те, кто не смог уехать, уходят и не возвращаются.

Для высадки пассажиров требуется 4 3с, для посадки – 8 4с. Посадка выполняется после высадки и по правилу FIFO. Если пассажир приходит в момент прибытия автобуса (временной узел), то он может сесть в автобус.

Необходимо:

1. Собрать статистику очереди, включая распределение времени пребывания в очереди.
2. Найти распределение случайной переменной «число необслуженных пассажиров на один прибывший автобус».

Смоделировать 25 прибытий и уходов автобуса.

Метод построения модели

Модель состоит из 2х сегментов: первый моделирует пассажиров, входящих на остановку; второй – приход автобуса и высадку пассажиров. Единица времени – 1 с.

1-й сегмент

Каждый транзакт-пассажир, поступивший в модель, входит в блок “QUEUE LINE”, где остается до тех пор, пока не придет автобус, и пока не подойдет очередь этого пассажира сесть в автобус.

Если эти условия выполнены, то двери автобуса (BUS) открыты для пассажира. Если мест нет, то транзакт-

пассажир модифицирует счетчик отказов и уходит из модели.

Если в автобусе есть место, транзакт-пассажир переходит в блок, следующий за блоком TEST, и закрывает вход в автобус для следующего пассажира.

После задержки в 8 4 с подсчитывается количество пассажиров в автобусе.

Далее транзакт-пассажир открывает автобус для следующего транзакта и уходит из модели.

Транзакт выходит из очереди и в блоке TEST, работающем в режиме передачи, осуществляется проверка, есть ли свободное место в автобусе (NOWON меньше 50).

2-й сегмент

Транзакт-автобус вводится в модель каждые 30 мин.

Затем он переводится в блок ADVANCE, где моделируется опоздание автобуса.

После блока ADVANCE, транзакт-автобус разыгрывает число своих пассажиров NOWON.

Далее разыгрывается число пассажиров, желающих выйти, и это число заносится в параметр 1.

Используя затем параметр 1 в качестве параметра цикла, транзакт-автобус выпускает всех желающих выйти.

При этом моделируется соответствующая задержка и корректируется величина NOWON.

После выхода из цикла транзакт-автобус открывает BUS и ждет до тех пор, пока последний пассажир из тех, кто может сесть в автобус, не закончит посадку.

Далее транзакт-автобус записывает в таблицу число необслуженных людей и обнуляет сохраняемую величину MAD для следующего автобуса.

Наконец, транзакт-автобус закрывает BUS и выходит из модели.

Таблица определений

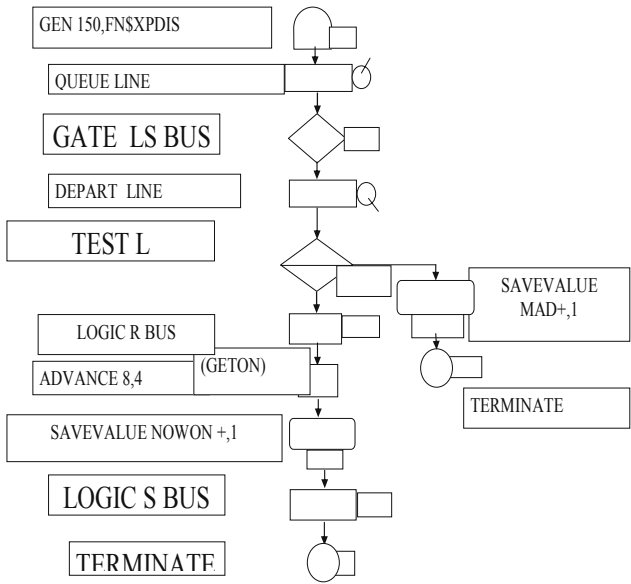
Элементы GPSS	Интерпретация
<i>Транзакты</i>	
1й сегмент	Пассажир
2й сегмент	Автобус. В параметре P1 первоначально фиксируется число людей, желающих выйти из автобуса, а затем число людей, которым ещё необходимо выйти.
<i>Таблицы</i>	
INQUE	Для оценки (1)
MAD	Для оценки (2)
<i>Логические переключатели</i> BUS	В положении «установлено» указывает на одновременное выполнение следующих условий: 1. Автобус находится на остановке 2. Пассажир может попытаться сесть в автобус
<i>Очереди</i> LINE	Очередь пассажиров на автобус
<i>Сохраняемые величины</i>	
MAD	Счетчик не попавших в автобус
NOWON	Счетчик числа пассажиров в прибывшем автобусе
<i>Функции</i>	
XPDIS	Экспоненциальная функция распределения
ONBUS	Функция распределения числа пассажиров в автобусе в момент прибытия на остановку
OFF	Функция распределения числа выходящих из автобуса пассажиров

В первой секции:

- Приход человека на остановку;
- Встать в очередь;
- Подождать открытия дверей автобуса;
- Выйти из очереди;
- • Если в автобусе нет места, то необходимо модифицировать счетчик отказов, затем уход с остановки; Если есть свободное место, то закрыть автобус перед следующим пассажиром;
- Войти в автобус;
- Подсчитать число людей в автобусе;
- Открыть автобус перед следующим пассажиром;
- Удалить пассажира из модели.

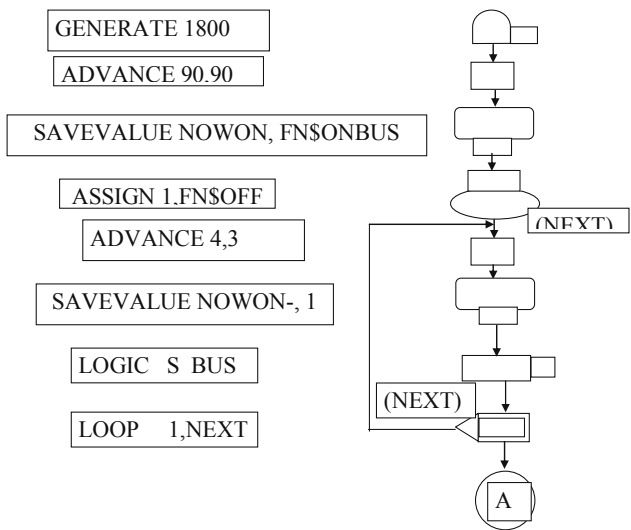
Во второй секции:

- Прибытие автобуса каждые 30 минут.;
- Предусмотреть возможное опоздание автобуса;
- Определить число пассажиров в автобусе;
- Определить число выходящих пассажиров;
- Организовать выход пассажиров с использованием оператора цикла;
- Открыть автобус для входа;
- Ожидание входа пассажиров из очереди;
- • Убедиться, что последний пассажир совершил посадку; Зафиксировать число необслуженных пассажиров;
- Отъезд автобуса.



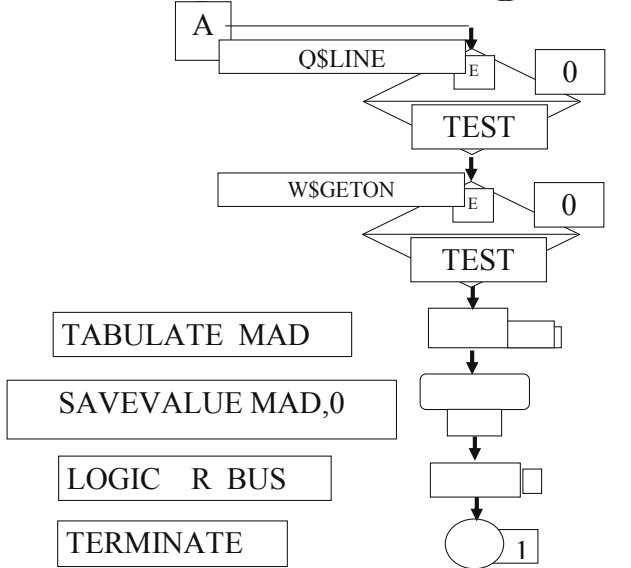
ONBUS FUNCTION RN1,C2
 0,20/1,51
 OFF1 FUNCTION RN1,C2
 0,3 /1,8
 XPDIS FUNCTION RN1,C24
 0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.91
 5/.7,1.2/.75,1.38/.8,1.6/.84,1.83/.88,2.12/.9,2.
 3/.92,2.52/.94,2.81/
 .95,2.99/.96,3.2/.97,3.5/.98,3.9/.99,4.6/.995,5.
 3/.998,6.2/.999,7/.9998,8

mad2 table x\$mad,0,1,10
 initial x\$mad,0
 initial x\$nowon,0
 inque qtable line,300,300,7



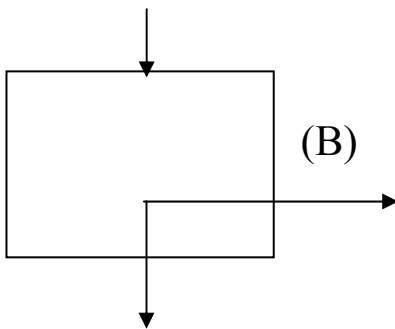
generate 150,Fn\$XPDIS
 queue Line
 Gate Ls Bus
 Depart Line
 Test L X\$NOWON,50,MAD1
 Logic R Bus
 Advance 8,4
 Savevalue Nowon+,1
 Logic S Bus
 Terminate

Madl Savevalue mad+,1
 terminate
 Generate 1800
 advance 90,90
 savevalue
 NOWON,(Int(FN\$OnBus))
 Plus(Int(FN\$OFF1)),4
 Next Advance 4,3
 SaveValue NOWON-,1
 Loop 4,Next
 Logic S BUS
 Test E Q\$Line,0
 TEST E W\$Geton,0
 Tabulate MAD2
 Logic R BUS
 Terminate 1



Система GPSS: расщепление транзактов и вывод членов ансамбля из модели.

В GPSS предусмотрена возможность расщепления транзактов.



A – число дополнительных транзактов, вводимых в модель.

B – имя блока, куда будут направлены дополнительные транзакты.

Входной транзакт называется *родителем*. Его *потомки*:

1. Имеют тот же уровень приоритета.
2. Имеют то же число, тип, значения параметров.
3. Отметка времени потомков совпадает с отметкой времени родителя.

Родитель, выходя из блока SPLIT безусловно переходит в следующий по порядку блок.

Для задания различия между транзактом-родителем и его потомками предусмотрена возможность упорядочения. При использовании операнда C его значение понимается как номер параметра, в котором родитель и его потомки будут упорядочены по номерам.

Предположим, что при входе транзакта-родителя в блок SPLIT значением операнда C будет j , а величина j -го параметра будет v . Тогда у транзакта-родителя P_j будет увеличен на 1 ($v+1$), первый транзакт-потомок получит номер ($v+2$) и т.д.

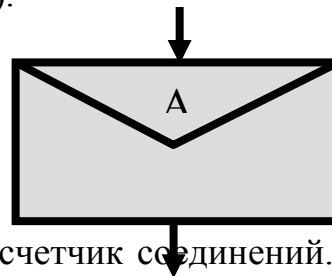
Пример. Предположим, что транзакт с величиной P7, равной 0, входит в блок SPLIT 3,RUT9,7

В результате в модель будут введены три потомка, которые перейдут в блок с именем RUT9. Родитель и его потомки будут иметь упорядоченные номера в седьмом параметре. Значение P7 у родителя будет равно 1 ($0+1$), у потомков – соответственно 2,3 и 4.

Необязательный операнд D в блоке SPLIT определяет число параметров, которое должно быть у каждого потомка. Если транзакт-потомок имеет больше параметров, чем родитель, то дополнительным параметрам присваивается начальное нулевое значение.

2 Блок ASSEMBLE

Целью блока является выход из модели одного или более членов ансамбля (семейства).



A – счетчик соединений. Его величина на единицу превышает число транзактов, которое должно быть удалено из модели.

Первый член ансамбля, войдя в блок, задерживается до прихода других членов ансамбля.

Когда число вошедших транзактов сравняется со значением счетчика соединений, то:

- транзакт, прибывший первым, продолжает движение в модели;
- остальные члены ансамбля выводятся из модели.

Свойства блока ASSEMBLE1. Для каждого ансамбля в одном блоке ASSEMBLE может выполняться только одна операция соединения.

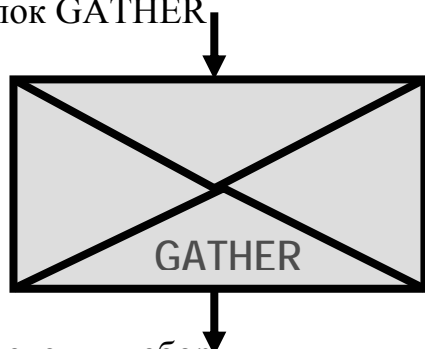
2. В данном блоке ASSEMBLE может параллельно выполняться операция соединения для двух и более ансамблей.

3. Для каждого ансамбля операции соединения могут параллельно выполняться в двух и более блоках.

4. После завершения одной операции соединения в данном блоке может начаться другая операция для того же ансамбля.

С точки зрения цепей, остающийся член ансамбля при входе в блок **ASSEMBLE** выводится из цепи текущих событий и помещается в цепь парности. Цепь парности можно трактовать как единственную для данного ансамбля и данного блока **ASSEMBLE**. При достижении заданного значения счетчика соединения, остающийся транзакт выводится из цепи парности и возвращается в цепь текущих событий в качестве последнего члена в своем классе приоритета.

3 Блок **GATHER**



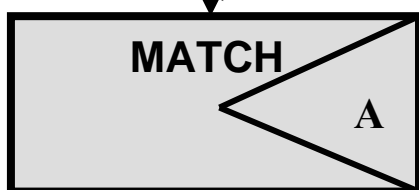
A – счетчик сборки

Целью блока является накопление членов ансамбля, пока в нем одновременно не окажется заданное число членов этого ансамбля.

Свойства блока **GATHER**:

1. Для каждого ансамбля в одном блоке **GATHER** может выполняться только одна операция сборки.
2. В данном блоке **GATHER** может параллельно выполняться операция сборки для двух и более ансамблей.
3. Для каждого ансамбля операции сборки могут параллельно выполняться в двух и более блоках.
4. После завершения одной операции сборки в данном блоке может начаться другая операция для того же ансамбля.

4 Блок **MATCH** (СОГЛАСОВАТЬ)



A – адрес сопряженного блока.

Блок **MATCH** (СОГЛАСОВАТЬ) синхронизирует движение транзактов с другим блоком **MATCH**.

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

```
LABEL1 MATCH LABEL 2
SEIZE CHANNEL
```

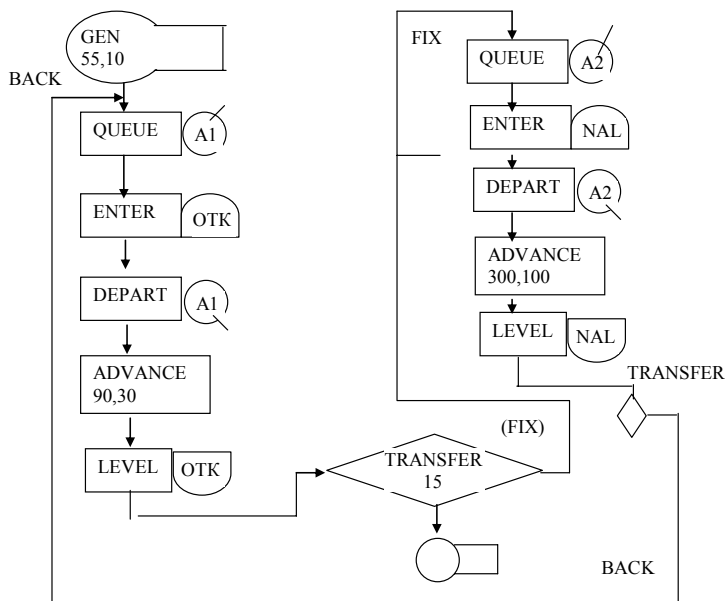
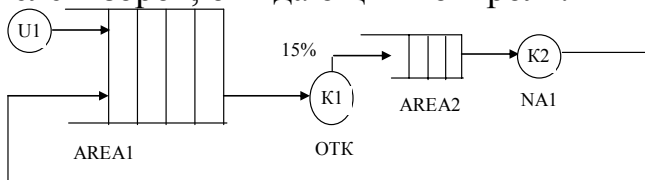
...

```
LABEL2 MATCH LABEL 1
ADVANCE 30
```

При входе транзакта-сообщения в блок **MATCH** с меткой **LABEL1** он будет ждать (в списке синхронизации) момента, когда другой опросный транзакт, принадлежащий тому же семейству, не войдет в сопряженный блок **MATCH** с меткой **LABEL2**. Только после этого сообщение займет канал **CHANNEL**, а опросное сообщение перейдет в блок **ADVANCE**.

Модель участка ОТК

Участок производства телевизоров. На участок технического контроля с интервалом 55 ± 10 (десятых минуты) поступают собранные телевизоры. В среднем 15% из них имеют неисправности, которые устраняются наладчиками. Затем телевизоры возвращаются в ОТК и вновь проверяются. Длительность контроля 90 ± 30 , длительность наладки от 300 ± 100 . Телевизоры, прошедшие ОТК, отправляются на склад. Надо определить число контролеров и наладчиков, таким образом, чтобы была минимальная длительность очереди телевизоров, ожидающих контроля.



```

OTK STORAGE 2
    GENERATE 55,10    BACK
QUEUE A1
    ENTER OTK
    DEPART A1
ADVANCE 90,30
    LEAVE OTK
    TRANSFER .15,,FIX
    TERMINATE
FIX QUEUE A2
    ENTER NAL

```

```

DEPART A2
ADVANCE 300,100
LEAVE NL
TRANSFER ,BACK
GENERATE 4800
TERMINATE 1

```

SMPL: список событий

SMPL – *Simulation Modeling Program Language* – это система моделирования, ориентированная на события.

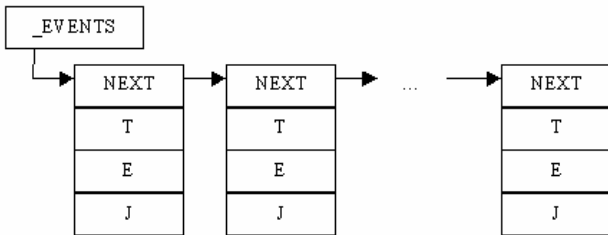
SMPL представляет собой набор процедур и функций, реализованных в среде Borland Pascal.

Для хранения текущего времени моделирования используется глобальная переменная `_TIME`.

Внутренняя организация языка SMPL В SMPL основой языка являются списки, отображающие события, устройства и очереди.

Для описания списков используется тип *запись*, а связь между элементами списка осуществляется через систему указателей.

Список событий (рис. 1) пополняется при обращении к функции `_SCHEDL`, которая помещает в список событий новый элемент и автоматически упорядочивает этот список.



Переменная `_EVENTS` указывает на начало списка событий. Для связей отдельных элементов используется указатель `NEXT`, который указывает на следующее по времени событие. Для последнего события значение элемента `NEXT` равно `nil`. Для удаления события из списка событий используется функция `_CAUSE` или `_CANCEL`. Поля элемента списка обозначают следующее:

- T – время свершения события;
- E – номер события, который указывается в качестве аргумента при обращении к функции `_SCHEDL`;
- J – номер транзакта, для которого планируется данное событие.

SMPL: список средств

Список средств (рис. 2) формируется при обращении к функции `_DEVICE` – функция описания средств. Для связи средств используется поле `NEXT`. Другие поля обозначают:

- NAME – имя устройства;
- B – время последнего обращения к функции `_RESERVE`;
- J – номер транзакта за которым зарезервировано средство, если прибор свободен, то $J = 0$;
- Z – число завершенных операций резервирования и освобождения;
- SB – сумма периодов занятого времени. Это поле подсчитывается при обращении к функции `_RELEASE`.

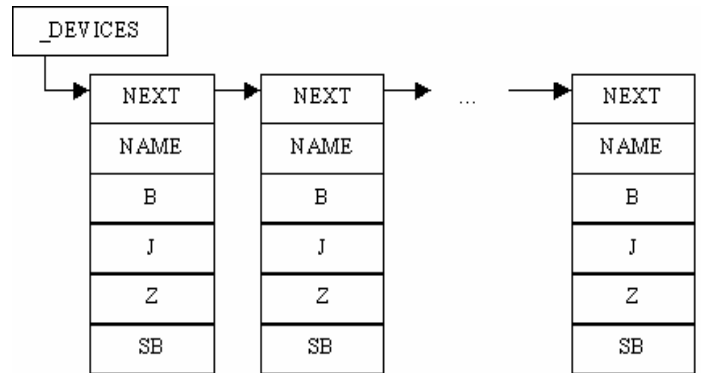


Рис. 2. Структура списка средств

SMPL: список очередей

В SMPL с помощью специальных инструментов можно организовать очередь, которая состоит из заголовка и собственно очереди (рис. 3). Каждому элементу очереди соответствует один транзакт в состоянии ожидания. Заголовок очереди формируется с помощью функции `_QUEUE`.

На начало списка очередей указывает переменная `QUEUES`. Связь между элементами списка устанавливается полем `NEXT`. Другие поля обозначают: `LIST` – указатель на первый элемент очереди;

`NEXT 1` – указатель, связывающий элементы одной очереди;

`NAME` – имя очереди;

`MAX` – максимальное содержимое очереди;

`STQ` – сумма произведений длины очереди на время, в течение которого эта сумма была постоянной;

`SW` – сумма времени ожидания, которая вычисляется при удалении транзакта из очереди (текущее время – время постановки транзакта в очередь);

`SW2` – сумма квадратов времени ожидания;

`TLAST` – время последнего изменения длины очереди;

`LEN` – текущее содержимое очереди;

`COUNT` – число элементов, удаленных из очереди;

`STAD` – стадия обработки транзакта;

`I` – значение приоритета;

`T` – время постановки транзакта в очередь;

`J` – номер транзакта.

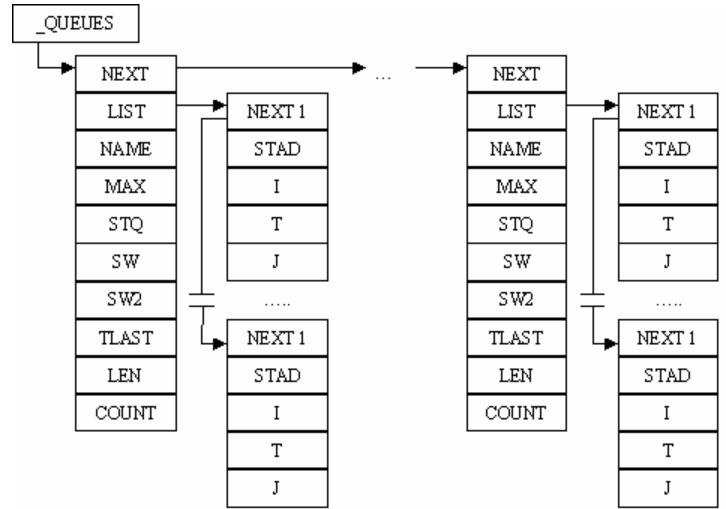


Рис. 3. Структура списка очередей

Элементы записываются в очередь функцией `_EnQUEUE`, а функция `_HEAD` позволяет извлекать элемент из очереди.

Операции инициализации языка SMPL

Все операции языка SMPL, реализованные в виде набора функций и процедур, можно разделить на несколько групп:

1. Операции инициализации: `_DEVICE`, `_QUEUE`, `_REINIT`.
2. Операции для работы со списком событий: `_SCHEDL`, `_CAUSE`, `_CANCEL`.
3. Операции для работы с устройствами: `_RESERVE`, `_RELEASE`, `_STATUS`.
4. Операции для работы с очередями: `_ENQUEUE`, `_HEAD`, `_LENGTH`.
5. Статистические функции: `IRANDOM`, `NEGEXP`.
6. Операции вывода результатов моделирования: `_M_EVENTS`, `_M_QUEUES`, `_M_DEVICES`, `_MONITOR`, `_REPORT`.
7. Функции обнаружения ошибок.

Операции инициализации

Функция `_DEVICE`. Она создает заголовок средства и включает его в список средств. Она возвращает указатель на заголовок средства, используемый при работе со средствами. Обращение к функции следующее:

`D := _DEVICE (DNAME);`

где `DNAME: string [12]` – задает имя устройства, которое используется при выводе информации;

`D: P_DEVICE` – содержит указатель на заголовок средства.

Функция `_QUEUE`. Обращение к функции следующее:

`Q := _QUEUE (QNAME);`

Эта функция создает заголовок очереди и возвращает указатель на заголовок `Q`.

`Q: P_QUEUE`

`QNAME: string [12]` – имя очереди.

Операция `_REINIT`. Она используется при завершении моделирования для освобождения динамической памяти.

Операции над списком событий языка SMPL

Функция `_SCHEDL`. Она включает в список новый элемент, соответствующий планируемому событию. Возвращаемое значение не используется. Вызов функции аналогичен вызову процедуры: `_SCHEDL (AE, AT, AJ)`; где `AE` – номер или тип события; `AT` – промежуток времени, через который событие должно произойти; `AJ` – номер транзакта.

Время свершения данного события определяется как сумма текущего времени моделирования и значения параметра `AT`:

$T_{\text{TIME}} + AT = T$. Время свершения используется при упорядочивании списка событий. Полученное время заносится в параметр `T` соответствующего члена списка событий.

Функция `_CAUSE`. Она удаляет событие из вершины списка. При этом текущее время моделирования становится равным времени свершения события. Обращение к функции следующее: `_CAUSE (AE, AJ)`;

Значение, возвращаемое самой функцией, не используется, а переменные используются для возврата следующих значений:

`AE` – номер или тип события; `AJ` – номер транзакта.

Функция `_CANCEL`. Она отыскивает в списке ближайшее по времени событие с указанным номером транзакта. Найденное событие удаляется из списка, а функция возвращает разность между временем, когда событие должно было произойти, и текущим временем моделирования.

Эта функция может быть использована для временной приостановки обслуживания транзакта с целью освобождения устройства, требуемого для продвижения заявки с абсолютным приоритетом (при приходе такой заявки любой прибор немедленно должен быть освобожден). Обращение к функции следующее:

`T := _CANCEL (AE, AJ)`; где `AE` – номер события; `AJ` – номер транзакта; `T` – разность между текущим временем и временем, когда событие должно произойти.

Операции над средствами языка SMPL

Функция `_RESERVE`. При обращении к ней указанное устройство резервируется за заданным транзактом. До освобождения устройства оно считается занятым и не может быть использовано другим транзактом. Обращение к функции следующее:

`_RESERVE (D, AJ);`

где `D` – указатель на заголовок средства, возвращаемый функцией `_DEVICE`; `AJ` – номер транзакта.

При обращении к этой функции в заголовке указанного средства модифицируются поля `B` и `J`:

`B := _TIME;`

`J := номер транзакта AJ;`

Функция `_RELEASES`. Освобождает указанное средство, занятое ранее функцией `_RESERVE`. Обращение к функции следующее:

`_RELEASES (D);`

При обращении к функции модифицируются поля `B` и `J` устройства:

`Z := Z + 1;`

`J := 0;`

`SB := SB + _TIME - B;`

Функция `_STATUS`. Возвращает значение 0, если устройство свободно, или номер транзакта, занимающего устройство в данный момент времени. Обращение к функции следующее:

`J := _STATUS (D);`

Операции над очередями языка SMPL

Операция `_EnQUEUE`. Эта функция помещает транзакт в указанную очередь. Результатом обращения к ней является новый элемент, который включается в очередь в соответствии с указанным приоритетом. Очередь упорядочивается по убыванию приоритета. Обращение к функции следующее:

`_EnQUEUE (Q, AJ, AI, ASS);`

где `Q` – указатель на заголовок очереди, возвращаемый функцией `_QUEUE`;

`AJ` – номер транзакта; `AI` – приоритет;

`ASS` – стадия обработки транзакта.

При обращении к функции корректируются поля:

`STQ := STQ + LEN (_TIME - TLAST);`

`LEN := LEN + 1; TLAST := _TIME;`

Иногда может быть модифицировано поле `MAX` (в том случае, когда текущая длина очереди превышает максимальную длину).

Функция `_HEAD`. Удаляет транзакт из вершины очереди и возвращает номер этого транзакта. Обращение к функции следующее:

`J := _HEAD(Q, ASS);`

где `Q` – указатель на заголовок очереди;

`ASS` – стадия обработки транзакта;

`J` – номер транзакта.

При обращении к данной функции модифицируются следующие статистические поля в заголовке очереди:

`STQ := STQ + LEN (_TIME - TLAST);`

`SW := SW + (_TIME - T);`

где `T` – время постановки транзакта в очередь.

`SW2 := SW2 + (_TIME - T)2; LIST := NEXT1;`

`TLAST := _TIME; LEN := LEN - 1; COUNT := COUNT + 1;`

Функция `_LENGTH`. Возвращает текущее содержимое соответствующей очереди. Обращение к функции следующее:

`L := _LENGTH(Q);`

где `Q` – заголовок очереди.