Министерство образования Республики Беларусь

Учреждение образования
«Белорусский государственный университет информатики
и радиоэлектроники»

Кафедра систем управления

С. В. Лукьянец

МОДЕЛИРОВАНИЕ В ПРОЕКТИРОВАНИИ ПРОМЫШЛЕННЫХ СИСТЕМ

Конспект лекций
для студентов специальности
1–53 01 07 «Информационные технологии и управление
в технических системах»

Минск

Содержание

Стр.

Введение 4

1. Модели массового обслуживания 6

2. Общие сведения о языке моделирования GPSS 11
   2.1. Сущность языка моделирования GPSS World 11
   2.2. Способы представления моделей 13

3. Создание моделей систем с одноканальными и многоканальными устройствами 15
   3.1 Операторы создания, уничтожения и задержки транзактов 15
   3.2. Операторы занятия и освобождения одноканальных устройств 17
   3.3. Регистраторы очередей 18
   3.4. Внутренняя логика работы пакета 20
   3.5. Реализация дисциплины обслуживания «первым пришел – первым обслужен внутри приоритетного класса» 21
   3.6. Моделирование многоканальных устройств 22
   3.7. Изменение маршрутов движения транзактов 24
   3.8. Управляющие операторы очистки и сброса статистики 26
   3.9. Стандартные числовые атрибуты (СЧА) 26
   3.10. Переменные пользователя 27
   3.11. Сохраняемые величины 28
   3.12. Примеры 30

4. Создание моделей систем с использованием объектов вычислительной категории 37
   4.1. Генераторы случайных чисел 37
   4.2. Использование дискретных равномерных распределений 37
   4.3. Использование дискретных неравномерных распределений (функции типа D) 38
   4.4. Использование непрерывных распределений (функции типа C) 39
   4.5. Функции типа E, L, M 40
   4.6. Моделирование пуассоновских потоков 41
   4.7. Арифметические переменные 42
   4.8. Моделирование нормального распределения 42
   4.9. Булевы переменные 43
   4.10. Проверка числовых выражений 44
   4.11. Примеры 45

5. Использование средств рационального построения моделей 47
   5.1. Параметры транзактов, изменение их значений и уровня приоритетов 47
   5.2. Системы с параллельно работающими идентичными устройствами и раздельными очередями 48
   5.3. Моделирование таблиц 49
   5.4. Примеры 50

6. Моделирование систем с устройствами в режимах прерывания и недоступности 53
   6.1. Моделирование захвата устройств 53
   6.2. Моделирование недоступности устройств 54
   6.3. Применение списков пользователя 55
   6.4. Примеры 57

7. Моделирование сложных производственных систем 59
   7.1. Операторы создания копий транзактов и обеспечения синхронизации их движения 59
   7.2. Моделирование цикла 61

7.3. Моделирование логического управления 61
7.4. Проверка состояний логического ключа и других объектов модели 62
7.5. Примеры 63
Заключение 70
Литература 71

ВВЕДЕНИЕ

Дисциплина «Моделирование в проектировании промышленных систем» рассматривает вопросы применения имитационного моделирования при проектировании сложных технических систем, к которым относятся гибкие производственные системы (ГПС), их подсистемы и другие объекты дискретного производства, в основе которых с точки зрения моделей лежат системы массового обслуживания.

Сложная техническая система состоит из множества взаимосвязанных элементов и подвержена влиянию внешних воздействий. При ее изучении осуществляют структуризацию, связанную с выявлением наиболее существенных элементов и установлением связей между ними, и алгоритмизацию, заключающуюся в рассмотрении алгоритмов поведения системы и ее подсистем в процессе функционирования. Под последним понимают изменение состояния системы во времени при достижении ею поставленной цели. Адекватную замену реальной системы моделью, облегчающую, упрощающую, ускоряющую и удешевляющую изучение свойств системы, называют моделированием.

Так как модель реального объекта сама является сложной системой, то ее разработка представляет собой трудоемкий процесс, к основным этапам которого относятся: формулирование цели и подготовка исходных данных; разработка концептуальной модели (связанная со структуризацией и алгоритмизацией объекта); выбор метода и средств моделирования (исходя из характера системы, достоинств и недостатков методов, наличия средств их реализации); разработка математической и программной модели (составление математического описания элементов системы, их взаимосвязи и написание программы на универсальном языке, языке моделирования общего назначения или специализированном языке); проверка адекватности модели объекту и ее корректировка; планирование и проведение расчетов на ЭВМ; анализ полученных результатов.

К настоящему времени разработано большое количество методов моделирования. С точки зрения подобия реальному объекту различают полные и приближенные модели. В зависимости от характера процессов в системе они подразделяются на детерминированные и стохастические, статические и динамические, дискретные, непрерывные и дискретно–непрерывные. По форме представления объекта модели делят на физические и математические. Математические модели в свою очередь делятся на аналитические и имитационные. Аналитические модели, позволяющие получить решения в явном виде, часто не являются структурно подобными объектам моделирования. При имитационном моделировании модель структурно подобна объекту, воспроизводит процесс функционирования системы во времени с сохранением составляющих его элементарных явлений и событий. При этом есть возможность управлять масштабом времени, пересчет которого осуществляют либо с постоянным шагом (обычно при моделировании систем непрерывного производства), либо в

'движении от события к событию (для систем дискретного производства), что позволяет существенно экономить машинное время.

В процессе проектирования промышленных систем используется ряд моделей. В самом общем случае все модели, отражающие динамику происходящих в таких системах процессов, можно подразделить на синхронные, в которых есть привязка ко времени, и асинхронные, учитывающие параллельные процессы и причинно-следственные связи. К первой группе относятся дискретные детерминированные и дискретно–стохастические модели, при построении которых используется теория автоматов. Ко второй группе относят модели, основанные на теории систем массового обслуживания (СМО). Наряду с ними получили широкое распространение сетевые модели: сети Петри и их модификации. Значительная роль при создании сложных систем отводится компьютерному расчету, реализующему методы имитационного моделирования, основанные на интерпретации СМО.

Сети Петри целесообразно применять для проверки согласованности работы оборудования и выявления конфликтных ситуаций в системе. Применение временных сетей Петри позволяет находить условия выполнения в сети циклических процессов в системах конвейерного типа.

В данном пособии будет уделено внимание имитационным моделям на языке GPSS World.

1. МОДЕЛИ МАССОВОГО ОБСЛУЖИВАНИЯ

Основными понятиями систем массового обслуживания являются: входящий поток требований (заявок), поступающих на обслуживание; правило постановки требований в очередь и выхода из нее; дисциплина обслуживания требований в устройстве; выходящий поток требований.

Входящий поток требований задается описанием моментов поступления требований в систему каким–либо законом: детерминированным или вероятностным. В СМО поток требований принимается простейшим – пуассоновским.

Различают следующие правила постановки требований в очередь и выхода из нее: первым пришел – первым обслужен; последним пришел – первым обслужен (стек); случайный выбор из очереди; выбор из очереди по определенным признакам (рангу, типу требований и т.п.).

Дисциплины обслуживания делят на бесприоритетные и приоритетные, последние подразделяют на дисциплины без прерывания и с прерыванием.

Обслуживание в СМО может осуществляться в одном устройстве (канале) или в нескольких, первые СМО называют одноканальными, другие – многоканальными. Если в СМО один этап обслуживания, то система называется однофазной, если последовательно используются несколько каналов обслуживания, то многофазной.

Выходящий поток требований одной СМО может быть входящим для другой СМО. Если выходящий поток в данной СМО не поступает на ее вход, то систему считают разомкнутой, если поступает, – то замкнутой.

Основные аналитические характеристики разомкнутых СМО. Если интенсивность входящего потока $\lambda = 1/t_n$, где t_n – среднее значение длительностей интервалов поступления требований, а интенсивность обслуживания требований в одноканальном устройстве $\mu = 1/t_0$, где t_0 – среднее время обслуживания одного требования, то загрузка оборудования

$$\rho = \lambda/(K\mu), \qquad (1.1)$$

где K – число каналов в СМО.

Применительно к ГПС, в случаях, когда $\rho \le 1$, система справляется с обслуживанием требований (деталей, партий деталей). Это возможно, если для каждой группы оборудования загрузка не превышает 1. При $\rho = 1$ наступает установившийся режим, который достижим теоретически (при детерминированных входящих потоках и интервалах обработки). Практически $\rho < 1$, а $1 - \rho$ – время простаивания СМО. Если же $\rho > 1$, то ГПС или ее отдельные единицы оборудования не справляются с обслуживанием деталей и очереди к оборудованию растут бесконечно.

В одноканальной СМО средняя длина очереди с пуассоновским входящим потоком и произвольно распределенным временем обслуживания по формуле Поллачека–Хинчина:

$$\ell = \frac{\rho^2(1+c^2)}{2(1-\rho)}, \tag{1.2}$$

где $c = \sigma / t_0$, здесь σ – среднее квадратичное отклонение времени обслуживания.

Так как при детерминированном обслуживании $c = 0$, а при экспоненциальном – единице, то формула (1.2) при детерминированном обслуживании имеет вид

$$\ell = \frac{\rho^2}{2(1-\rho)}, \tag{1.3}$$

а при экспоненциальном –

$$\ell = \frac{\rho^2}{1-\rho}. \tag{1.4}$$

Из формулы (1.2) следует, что длина очереди в сильной степени зависит от коэффициента c. Поэтому нестабильности в ГПС влияют на объем незавершенного производства. Кроме того, из этого выражения следует, что ℓ зависит от отношения интенсивностей поступления λ и обслуживания μ, а не от их абсолютных значений. Это значит, что если выдерживать загрузку оборудования одинаковой, а изменять интенсивность поступления партий деталей в систему, то относительно партий деталей длина очереди, а значит, и объем незавершенного производства останутся неизменными. Если же при этом увеличить размер партии, то объем незавершенного производства возрастет.

Если обозначить через $t_{ож}$ среднее время ожидания обслуживания, то к моменту начала обслуживания в системе образуется очередь:

$$\ell = \lambda t_{ож}. \tag{1.5}$$

Аналогично будет определяться среднее число заявок в СМО через среднее время их пребывания в системе $t_{пр}$:

$$n = \lambda t_{пр}. \tag{1.6}$$

Для СМО справедливо также соотношение

$$n = K\rho + \ell, \tag{1.7}$$

где n – среднее число требований в системе. Это число необходимо знать при расчете емкостей складов ГПС и накопителей в гибких производственных модулях (ГПМ).

Наконец, для СМО с простейшим входящим потоком и экспоненциальным обслуживанием вероятность того, что в системе нет требований, составляет

$$P_0 = \left[\sum_{i=0}^{K-1} \frac{(K\rho)^i}{i!} + \frac{(K\rho)^K}{K!(1-\rho)} \right]^{-1}, \tag{1.8}$$

а вероятность того, что число требований в системе в установившемся режиме равно k, составляет

$$P_K = \begin{cases} P_0 \dfrac{(K\rho)^k}{k!}, & k \le K; \\ P_0 \dfrac{K^K \rho^k}{K!}, & k > K. \end{cases} \tag{1.9}$$

Средняя длина очереди при отсутствии в системе требований равна

$$\ell = \frac{\rho(K\rho)^K}{K!(1-\rho)^2} P_0. \tag{1.10}$$

Стохастические сети. Стохастическая сеть состоит из связанных между собой узлов, в которых находятся СМО, соответствующие группам оборудования, гибким производственным модулям, имеющим свои накопители деталей. Общий склад ГПС представляется в стохастической сети источником–приемником. Иногда при моделировании ГПС могут применяться замкнутые сети. Это делается для изучения работы транспорта, который циркулирует в ГПС (в модели – постоянное число требований).

В стохастической сети в установившемся режиме интенсивности входящего в узел потока требований равны интенсивностям выходящего из этого узла потока. Если требования при переходе от узла к узлу не меняют приоритета, то сети называют однородными.

Рассмотрим разомкнутую стохастическую сеть с пуассоновским потоком требований из источника и экспоненциальным распределением интервалов обслуживания. Такие сети часто называют экспоненциальными.

Пусть структура ГПМ имеет вид, изображенный на рис. 1.1.

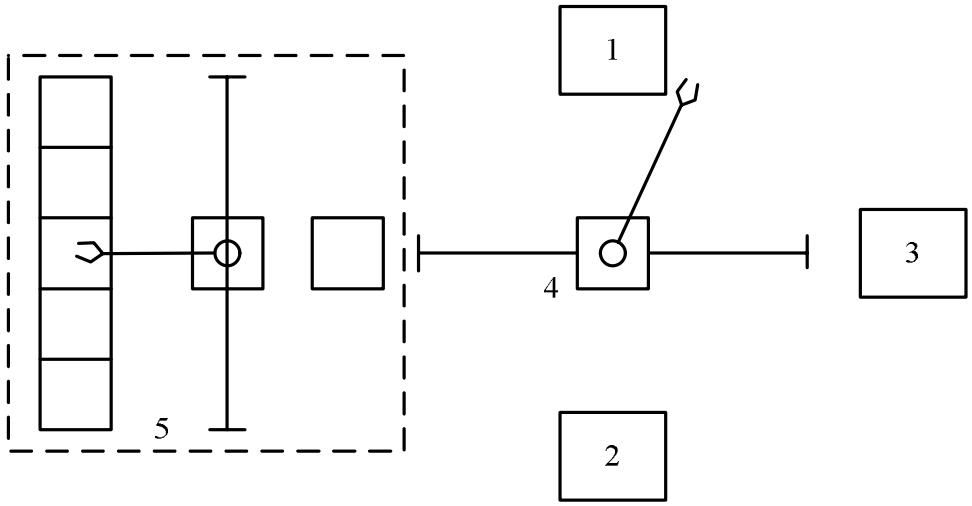


Рис. 1.1. Схема ГПМ: 1 – 3 – станки; 4 – транспортный робот; 5 – автоматизированный склад

Сеть, соответствующая этому ГПМ, приведена на рис. 1.2.

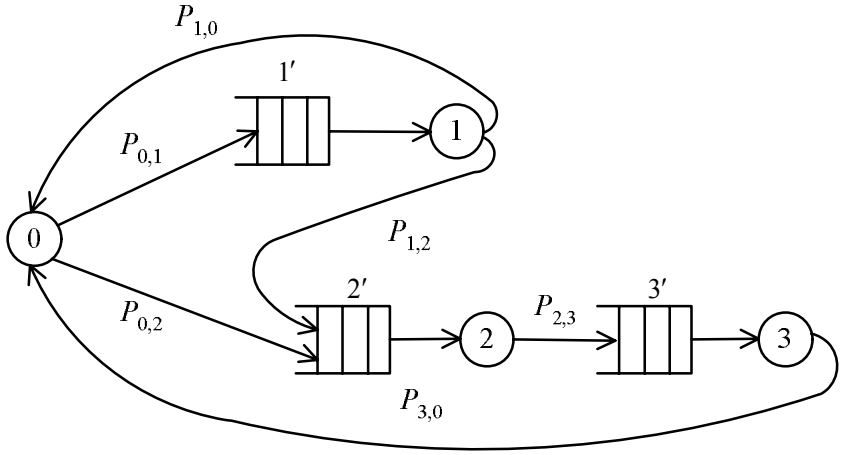


Рис. 1.2. Стохастическая сеть ГПМ: 0 – источник–приемник требований; 1–3 – станки; 1′–3′ – очереди к станкам

Особенность ГПМ состоит в том, что после обработки на станке 1 часть деталей уходит из системы с вероятностью $P_{1,0}$, а часть с вероятностью $P_{1,2}$ – на обработку к станку 2. В общем случае вероятность $P_{i,j}$ – это часть всех деталей, поступающих с узла i на узел j.

Так как в сетях без потерь интенсивность потока поступающих в i–й узел требований равна интенсивности потоков, выходящих из i–го узла, то справедливо выражение

$$\lambda_i = \sum_{j=1}^{M} \lambda_{i,j} = \sum_{j=1}^{M} P_{i,j}\lambda_i , \qquad (1.11)$$

где M – число узлов сети.

Интенсивность входящего потока деталей в i–й узел за время T:

$$\lambda_i = \frac{1}{T}\sum_{k=1}^{I} N_k q_{i,k},\ i = 1,...,M, \qquad (1.12)$$

где I – количество наименований деталей, обрабатываемых на i–м оборудовании; N_k – число деталей k–го наименования; $q_{i,k}$ – число операций над деталью k–го наименования на i–м оборудовании.

Интенсивность источника требований:

$$\lambda_0 = \frac{1}{T}\sum_{k=1}^{\mathrm{Q}} N_k , \qquad (1.13)$$

где Q – полное число наименований деталей в ГПС.

Интенсивность потока деталей из i–го узла в j–й:

$$\lambda_{i,j} = \frac{1}{T}\sum_{k=1}^{I+J} N_k q_{i,k} \,. \tag{1.14}$$

Вероятность передачи деталей из i–го в j–й узел (на станок):

$$P_{i,j} = \frac{\lambda_{i,j}}{\lambda_i}. \tag{1.15}$$

Среднее время обслуживания в i–м узле:

$$\tau_{0i} = \frac{\sum_{k=1}^{I} \rho_{i,k}}{\lambda_i}, \tag{1.16}$$

где $\rho_{i,k}$ – загрузка i–го узла деталями k–го наименования.

Средний объем незавершенного производства ГПС:

$$N = \sum_{i=1}^{M} n_i \tag{1.17}$$

и среднее время цикла ГПС:

$$T_{пр} = \sum_{i=1}^{M} \frac{\lambda_i}{\lambda_0} t_{пр\,i}, \tag{1.18}$$

где $t_{пр\,i}$ – среднее время пребывания требования в i–м узле.

Стохастическая сеть, как и СМО, сбалансирована, если загрузка всех узлов одинакова. Если сеть несбалансирована, то в ней есть узкое место, т.е. узел, для которого загрузка максимальна. Такой узел становится еще одним источником требований в сети.

2. ОБЩИЕ СВЕДЕНИЯ О ЯЗЫКЕ МОДЕЛИРОВАНИЯ GPSS

GPSS (General Purpose Simulation System) – общецелевая система моделирования, представляет собой мощное средство для исследования процессов, подчиняющихся законам СМО и выходящих за их рамки. Язык GPSS был разработан профессором Гордоном более 40 лет назад. Развитием этой системы явилось создание в 1984 году специалистами фирмы Minuteman Software (США) системы GPSS/PC, ориентированной на DOC. Система GPSS World – мировая общецелевая система моделирования, с 1994 года была ориентирована на OS/2 фирмы IBM, а с 2000 года – на ОС Windows фирмы Microsoft. Последняя версия позволяет моделировать как дискретные, так и непрерывные процессы. Как отмечают специалисты [1–3], язык GPSS в настоящее время переживает второе рождение благодаря его достоинствам: доступности для изучения, дружественному интерфейсу при использовании, автоматической подготовке обширной информации, возможности постоянного совершенствования.

Система GPSS World позволяет исследовать как сложные производственные системы и их подсистемы (обрабатывающие, складские, транспортные), так и непроизводственные системы (телекоммуникационные, торговые, банковские и др.).

2.1. Сущность языка моделирования GPSS World

Рассматриваемый язык моделирования основан на концепции, суть которой в том, что моделью сложной системы является адекватное описание ее абстрактных элементов (объектов) и правил их взаимодействия.

Все объекты делятся на семь категорий: динамические, операционные, аппаратные, статистические, вычислительные, запоминающие, группирующие.

К ***динамическим объектам*** относятся ***транзакты***. Они создаются, передвигаются, останавливаются, задерживаются, уничтожаются. С каждым транзактом связан ряд их конкретных свойств (номер, уровень приоритета, другие данные). Физический смысл транзактов устанавливает пользователь. Это могут быть детали, транспортные средства, операторы и т.п.

Объекты операционной категории – блоки описывают правила взаимодействия элементов системы и определяют пути движения транзактов в модели. Эти объекты реализуют подпрограммы, написанные на макроассемблере. В рассматриваемой версии языка около 50 типов блоков.

К ***объектам аппаратной категории*** относятся одноканальные устройства, многоканальные устройства и логические ключи («семафоры»). В реальных системах – это ресурсы (станки, транспортные средства, сборочные машины, работники и т.п.).

К ***статическим объектам*** относятся регистраторы очередей и средства табулирования количественных характеристик модели.

Вычислительная категория включает объекты, моделирующие функции, арифметические и булевы переменные.

Запоминающая категория объектов представляет собой линейные и матричные сохраняемые величины, предназначенные для сохранения и использования числовой информации.

Группирующая категория объектов объединяет числовые группы, группы транзактов и списки событий. Особо выделим ***списки***, которые определяют внутреннюю логику работы пакета.

Существуют шесть видов списков: текущих событий; будущих событий; прерываний; повторений; задержки; пользователя.

Список текущих событий (СТС) – Current Events Chain, включает транзакты, соответствующие событиям, время наступления которых равно или меньше (для заблокированных транзактов) текущего.

В ***список будущих событий*** (СБС) – Future Events Chain, входят транзакты, события с которыми произойдут в будущем.

Список прерываний (Interrupt Chain) включает транзакты, обслуживание которых прервано.

Список повторений (Retry Chain) включает транзакты, которые ожидают изменения состояния канала обслуживания.

Список задержки (Delay Chain) содержит транзакты, пытающиеся войти в занятый канал.

Список пользователя (User Chain) включает транзакты, которые пользователь переводит в определенных случаях из списка текущих событий в качестве пассивных.

В модели может быть до шести списков, но обязательными являются СТС и СБС (каждый по одному).

В языке GPSS организовано автоматическое обслуживание системных часов – таймера модельного времени, которое продвигается дискретно – от события к событию. Системные часы состоят из двух частей: таймера абсолютного времени (от начала до конца моделирования) и таймера относительного времени (от определенного момента до конца моделирования).

Поясним работу таймера. Пусть работает гибкий производственный модуль (ГПМ). На обработку поступают детали после включения ГПМ и в дальнейшем через каждые 20 минут. Время обработки детали в модуле 15 минут.

Составим таблицу.

Номер события	Событие	Фактическое время	Модельное время
1	Включение ГПМ	8 ч. 00 мин.	0
2	Первая деталь поступает на обработку	8 ч. 20 мин.	20
3	Первая деталь обработана	8 ч. 35 мин.	35
4	Вторая деталь поступает на обработку	8 ч. 40 мин.	40
	⋮		

В самом начале моделирования таймер в положении «0».

Когда начинается моделирование, планируется приход первого транзакта, после чего таймер устанавливается в значение времени, соответствующее моменту появления этого транзакта (20 единиц). Транзакт продвигается по модели, а таймер – к моменту следующего события (конец обработки детали) – 35 единиц и т.д.

Выделим особенности таймера.

1. Единица модельного времени определяется разработчиком. Она должна быть целой вещественной и единой для всех модулей программы.

2. Так как пакет GPSS ориентирован на следующее событие, то таймер продвигается к событию, если оно наступило, если нет – то пропускает его и идет дальше. При этом время прогона не зависит от масштаба единицы модельного времени (в этом смысл «сжатия» или «растягивания» времени).

2.2. Способы представления моделей

Модели на языке GPSS могут быть представлены в виде блок–диаграмм или в виде текстов программ.

На основе разработанного алгоритма функционирования моделируемой системы создают ***блок–диаграмму*** модели. Она содержит набор фигур, соответствующих определенным блокам, соединенным между собой линиями. Например, для рассмотренного ГПМ, фрагмент блок–диаграммы имеет вид, приведенный на рис. 2.1, где блоки обозначены тремя первыми буквами, остальные надписи соответствуют исходным данным. Стрелки указывают направле–ния движения транзактов. Из блок–диаграммы следует, что транзакт–деталь входит через блок GENERATE в ГПМ через 20 единиц модельного времени после начала работы модуля, занимает модуль с именем GPM, задерживается в нем (блок ADVANCE) на 15 единиц, освобождает модуль (блок RELEASE) и удаляется из системы (блок TERMINATE).

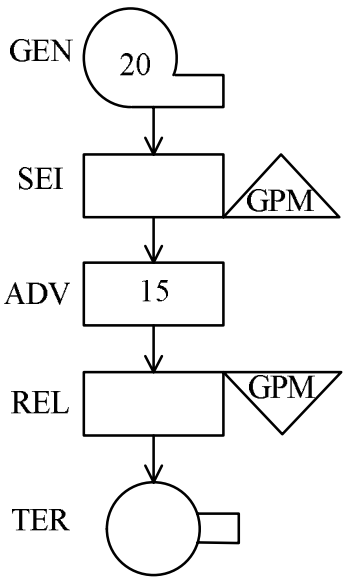


Рис. 2.1. Блок–диаграмма фрагмента модели работы ГПМ

Блок–диаграммы целесообразно создавать в случаях сложных моделей (с обратными связями, при параллельных процессах и т.п.).

При написании ***текстов программ*** используют операторы. Все операторы делятся на два типа: блоки; операторы описания данных (команды). Операторы GPSS в соответствии с форматом записываются в определенных полях. Набор полей в GPSS World: метка – оператор – операнды – комментарий.

Для идентификации объектов (например, устройств, очередей, ключей и т.п.) им присваивают имена, которые начинаются с буквы, могут включать до 200 букв и цифр, а также символы подчеркивания (например, SEIZE A28).

Метки образуются также, как и имена операторов. Так как транслятор GPSS присваивает меткам номера, начиная с 10000, то пользователь при желании записать вместо символической метки цифру может это сделать через команду EQU, предусмотрев номер, не превышающий 9999.

В поле операций записывается глагол (ключевое слово оператора).

В поле операндов задаются исходные данные (имена операторов, приоритеты транзактов, временные интервалы и др.). Операнды будем обозначать символами A, B, C, D, E, F, G, H. Операнды разделяются запятыми. Особенности описания данных в командах для переменных, функций и других объектов будут рассмотрены позднее.

Комментарии отделяются от поля операндов точкой с запятой. Модули программы могут иметь наименования, которые предваряются точкой с запятой в начале строки.

Для избежания ошибок не следует использовать в качестве имен и меток начальные символы объектов GPSS.

Поясним сказанное на примере программы фрагмента модели, блок–диаграмма которого изображена на рис. 2.1.

```
GENERATE    20      ; Интервалы поступления деталей
SEIZE       GPM     ; Занятие ГПМ
ADVANCE     15      ; Обработка детали
RELEASE     GPM     ; Освобождение ГПМ
TERMINATE           ; Вывод транзактов из модели
```

В этом фрагменте программы метки не нужны.

3. СОЗДАНИЕ МОДЕЛЕЙ СИСТЕМ С ОДНОКАНАЛЬНЫМИ И МНОГОКАНАЛЬНЫМИ УСТРОЙСТВАМИ

При моделировании промышленных систем возникает необходимость рассмотрения взаимодействия операционных и динамических объектов с аппаратными объектами. Эти ситуации невозможно отразить без соответствующих статистических объектов.

В связи с этим необходимо изучить ряд операторов, которые позволяют решать эти задачи.

3.1 Операторы создания, уничтожения и задержки транзактов

GENERATE – генерировать. Это блок, через который транзакты входят в модель. Его формат:

GENERATE [A], [B], [C], [D], [E]

Квадратные скобки здесь и в дальнейшем указывают, что данный операнд является необязательным.

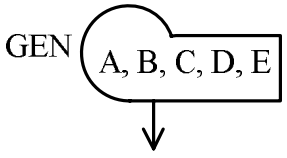
Операнды A и B задают интервалы времени поступления транзактов в модель: A определяет среднее значение, B – половину поля допуска интервалов при равномерном законе распределения или модификатор–функцию при неравномерном законе распределения. В первом случае интервал определяется как A±B, во втором – как A#B (здесь и в дальнейшем # – знак умножения). Если, например, задан оператор GENERATE 3,1, то интервалы прихода транзактов будут от 2 до 4. Значения A и B могут быть как целыми, так и дробными числами (например, GENERATE 4.2, 1.1). По умолчанию A= 0, B=0. При B=0 интервал детерминирован и равен значению операнда A. Использование модификатор–функции, а также библиотечных генераторов случайных чисел будет рассмотрено позднее.

Операнд C задает смещение, т.е. время прихода первого транзакта; по умолчанию это время определяется операндами A и B.

Операнд D – ограничитель. Он задает общее число транзактов, которые могут войти в модель в период моделирования; по умолчанию ограничения нет.

Операнд E определяет уровень приоритета транзактов. Чем большее целое число задано, тем выше уровень приоритета. По умолчанию – уровень нулевой.

На блок–диаграммах оператор GENERATE изображается следующим образом:

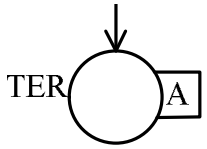


В блоке GENERATE обязателен или операнд A или D, остальные в соответствии с моделируемой ситуацией могут отсутствовать.

TERMINATE – завершать. Это блок, через который транзакты удаляют из модели. Его формат:

```
TERMINATE        [A]
```

и графическое изображение:



Операнд A является указателем уменьшения счетчика завершения моделирования. Счетчик завершения – это ячейка памяти, в которую заносится целое положительное число, записанное в начале моделирования в операнде A команды START. Оператор START имеет формат:

```
START       A, [B], C, [D]
```

Операнд B задает разрешение (по умолчанию) или запрет (NP) на вывод статистики.

Операнд C в этом пакете не используется (в GPSS–PC он задает «снимки»).

Оператор START располагают либо в конце текста программы (после его ввода сразу начнется выполнение программы), либо в командной строке (в интерактивном режиме).

В модели может быть несколько блоков TERMINATE, а счетчик завершений один. Поэтому надо следить за тем, какие блоки TERMINATE уменьшают значение счетчика завершений, а какие – нет.

Если моделируемая система работает определенное время, то для окончания моделирования в программе организуется модуль таймера. Если, например, моделирование длится 1000 единиц, то этот модуль выглядит так:

```
GENERATE      1000
TERMINATE     1
START         1
```

Во всех других точках модели операнды A блоков TERMINATE должны быть заданы по умолчанию.

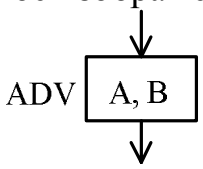
Если же необходимо закончить процесс моделирования после обработки определенного количества транзактов, например, деталей, то в операнд A команды START заносят это число, а в операнды A блоков TERMINATE, которые удаляют из модели соответствующие транзакты – детали, по единице. Например, на участке на трех станках обрабатываются детали соответственно первого, второго и третьего типов. Необходимо закончить моделирование после обработки 500 деталей первых двух типов. Это может быть реализовано так:

```
        ⋮
TERMINATE    1; Удаление деталей первого типа
        ⋮
TERMINATE    1; Удаление деталей второго типа
        ⋮
TERMINATE     ; Удаление деталей третьего типа
        ⋮
START        500
```

ADVANCE – задержать. Этот блок реализует задержку транзакта на время, указанное в операндах A и B. Формат блока:

```
ADVANCE    A, [B]
```

В операнде A указывается среднее время задержки на обслуживание, в B – половина интервала при равномерном законе или модификатор–функция при других законах распределения. Время задержки вычисляется соответственно как A±B и A#B. Значения A и B по умолчанию равны нулю. Если оба операнда заданы по умолчанию, то задержка равна нулю и транзакт переходит в следующий оператор. Графическое изображение блока ADVANCE:



3.2. Операторы занятия и освобождения одноканальных устройств

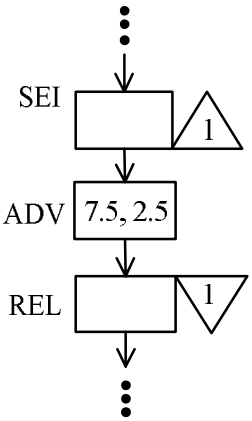
Блок ***SEIZE*** (занять) моделирует занятие устройства, блок ***RELEASE*** (освободить) имитирует освобождение устройства.

Форматы этих блоков:

```
SEIZE      A
RELEASE    A
```

В операнде A указывается цифровое или символическое имя устройства.

При работе модели автоматически формируется информация о работе устройств (о загрузке, числе входов, среднем интервале занятости и т.п.). Часто в модели используется последовательность блоков: SEIZE – ADVANCE – RELEASE, как в следующем фрагменте:



Однако не следует делать вывод, что эта последовательность обязательна.

3.3. Регистраторы очередей

В качестве регистратора очереди используются блоки ***QUEUE*** (стать в очередь) и ***DEPART*** (выйти из очереди).

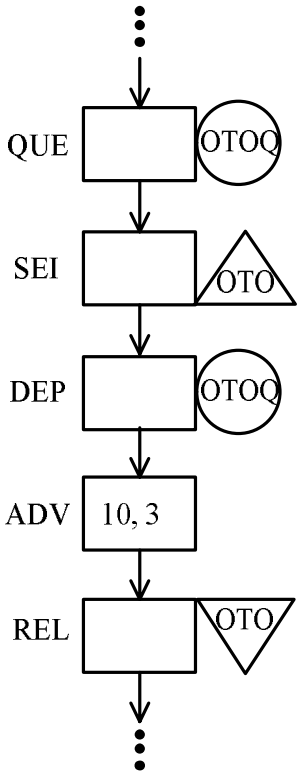
Форматы этих блоков:

QUEUE A, [B]
DEPART A, [B]

В операндах A указывается символическое или числовое имя очереди, в B – число, на которое увеличивается (в первом блоке) и уменьшается (во втором блоке) длина очереди; по умолчанию это число равно единице.

В результате моделирования формируется статистика о количестве вхождений транзактов в очередь, о числе нулевых вхождений в очередь, о максимальной и средней длине очереди, о среднем времени пребывания в очереди и т.п.

Рассмотрим фрагмент блок–диаграммы модели с блоками занятия, задержки, освобождения устройства и регистрации очереди.



Примечание: Не следует считать, что регистратор очереди нужен всегда, где в системе есть очередь. Программа работает и без блоков QUEUE и DEPART, но в этом случае не формируется статистика об очереди.

Пример 3.1. На рабочее место одного рабочего поступают заготовки через каждые (8 ± 4) минут, время на изготовление одной детали – (7 ± 1) минут. Время работы – 1 смена. Дисциплина обслуживания: первый пришел – первым

обслужен. Составить модель работы рабочего, сформировать информацию о загрузке рабочего и об очереди.

За единицу модельного времени принимаем 1 минуту, тогда время моделирования будет 480 единиц.

Программа модели и выходная статистика имеют вид:

```
; Программа
        GENERATE        8,4
        QUEUE           QMASTER
        SEIZE           MASTER
        DEPART          QMASTER
        ADVANCE         7,1
        RELEASE         MASTER
        TERMINATE
;
        GENERATE        480
        TERMINATE       1
        START           1
```

Выходная статистика

START TIME	END TIME	BLOCKS	FACILITIES	STORAGES
0.000	480.000	9	1	0

LABEL	LOC	BLOCK TYPE	ENTRY COUNT	CURRENT COUNT	RETRY
	1	GENERATE	59	0	0
	2	QUEUE	59	0	0
	3	SEIZE	59	0	0
	4	DEPART	59	0	0
	5	ADVANCE	59	1	0
	6	RELEASE	58	0	0
	7	TERMINATE	58	0	0
	8	GENERATE	1	0	0
	9	TERMINATE	1	0	0

FACILITY	ENTRIES	UTIL.	AVE. TIME	AVAIL.	OWNER	PEND	INTER	RETRY	DELAY
MASTER	59	0.848	6.898	1	60	0	0	0	0

QUEUE	MAX	CONT.	ENTRY	ENTRY (0)	AVE.CONT.	AVE.TIME	AVE.(–0)	RETRY
QMASTER	2	0	59	25	0.238	1.933	3.354	0

Анализ статистики показывает, что рабочий изготовил за смену 58 деталей, при этом его загрузка составила 84,8 %, а среднее время изготовления одной детали – 6,898 минут. Средняя длина очереди 0,238 деталей, среднее время ожидания заготовки в очереди 1,933 минуты; максимальная длина очереди 2 детали; 25 заготовок сразу попали к рабочему (нулевое вхождение в очередь); без учёта этих заготовок среднее время нахождения детали в очереди составило 3,354 минуты.

3.4. Внутренняя логика работы пакета

Как отмечено ранее, из возможных шести списков событий обязательны СТС и СБС. При работе модели транслятор продвигает транзакты в модели, изменяя их положение в списках и блоках.

Рассмотрим внутреннюю логику пакета на предыдущем примере.

Первым действием является ***ввод модели***. До этой фазы проверяются все операторы и в первую очередь отыскивается блок GENERATE (блок № 1 в выходной статистике). Так как операнды A и B этого блока соответственно равны 8 и 4, то определяется время прихода первого транзакта в блок. Допустим это время равно 10. Транзакту для дальнейшего движения присваивается номер 1 и он помещается в список будущих событий.

Далее для блока GENERATE № 8 следующему транзакту присваивается номер 2 и он также помещается в СБС со временем 480 единиц.

Читая команду START, транслятор заносит в счетчик завершения 1. На этом фаза ввода заканчивается.

Если в карте START задан операнд D (например, START 1, , , 1), то в выходной статистике для этого момента СТС (CEC) пуст, а в СБС (FEC) будет информация, которая содержится в общем случае в полях:

CEC
(CEC)XN PRI M1 ASSEM CURRENT NEXT PARAMETER VALUE

Здесь XN – номер транзакта; PRI – приоритет транзакта; M1 – время ввода в модель; ASSEM – номер семейства транзакта; CURRENT – номер блока, в котором находится транзакт в данный момент; NEXT – номер блока, куда войдет транзакт; PARAMETER и VALUE – имя и значение параметра транзакта.

В данном рассмотрении будем учитывать только 5 полей, исключив поля ASSEM, PARAMETER и VALUE.

В нашем случае:

СТС – пусто;
СБС – 1 0 10 0 1
2 0 480 0 8

Поскольку СТС пуст, происходит ***коррекция таймера***. Таймер устанавливается в значение 10 (время движения первого транзакта) и транзакт № 1 перемещается в СТС. Информация о списках следующая:

СТС – 1 0 0 0 1

СБС – 2 0 480 0 8

Выполнение фазы просмотра. Транзакт № 1 из СТС продвигается в блок 1 (GENERATE). Так как транзакт может беспрепятственно пойти в блок 2 (QUEUE), то движение транзакта приостанавливается и создается его последователь, который будет двигаться через разыгранное время из интервала 8 ± 4, допустим через 5 единиц, т.е. в 10 + 5 = 15 единиц. Следующему транзакту

присваивается № 3 и с этим временем он помещается в СБС. Транзакт же № 1 продвигается в блоки QUEUE, SEIZE, DEPART и ADVANCE. Так как в блоке ADVANCE он задерживается на (7 ± 1) единиц (допустим на 6 единиц), то он выводится из СТС и помещается в СБС со временем 10 + 6 = 16 единиц.

Так как транзакт № 1 прошел через блок SEIZE, транслятор просматривает СТС. Но он пуст. На этом фаза просмотра закончена. Содержимое списков событий:

```
СТС – пусто;
СБС – 3     0     15     0     1
      1     0     16     5     6
      2     0     480    0     8
```

Далее выполняется вторая фаза коррекции таймера и процесс продвижения транзактов продолжается по рассмотренному алгоритму.

3.5. Реализация дисциплины обслуживания «первым пришел – первым обслужен внутри приоритетного класса»

В предыдущем примере реализована дисциплина обслуживания: первым пришел – первым обслужен. В реальных системах часто возникают задачи, когда на одном оборудовании обрабатываются партии деталей различных типов (основные в данный плановый период и второстепенные – «фоновые», которые будут нужны в более поздние сроки). Рассмотрим этот случай на конкретном примере.

Пусть на одном обрабатывающем центре обрабатываются две группы деталей, причем детали первого типа имеют более низких приоритет, чем детали второго типа. Для деталей первого типа интервалы поступления (420 ± 360) с, интервалы обслуживания (300 ± 90) с; для деталей второго типа интервалы поступления (360 ± 240) с, интервалы обслуживания (100 ± 30) с. Время работы – 1 смена, т.е. 28800 с.

В данном случае реализуется дисциплина обслуживания: первым пришел – первым обслужен внутри приоритетного класса (рис. 3.1).

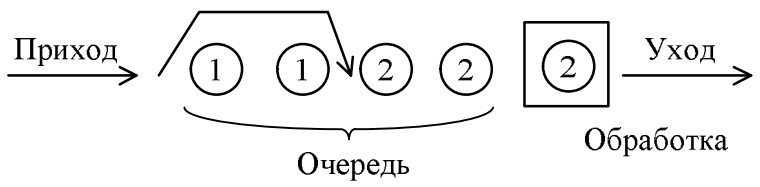


Рис. 3.1

Как бы ни поступали детали, они будут выстраиваться в очередь на обработку по приоритетам: в начале – с более высоким (2), в конце – с более низким (1).

При моделировании этой ситуации необходимы кроме модуля таймера два модуля, каждый для своего типа деталей. При единице модельного времени в 1 с и принятых обозначениях имен (OTO – обрабатывающего центра и OTOQ – очереди к нему) программа имеет вид:

```
; Модуль обработки первого типа деталей
        GENERATE    420, 360,,, 1
        QUEUE       OTOQ
        SEIZE       OTO
        DEPART      OTOQ
        ADVANCE     300, 90
        RELEASE     OTO
        TERMINATE
; Модуль обработки второго типа деталей
        GENERATE    360, 240,,, 2
        QUEUE       OTOQ
        SEIZE       OTO
        DEPART      OTOQ
        ADVANCE     100, 30
        RELEASE     OTO
        TERMINATE
; Модуль таймера
        GENERATE    28800
        TERMINATE   1
        START       1
```

Анализ выходной статистики показывает, что количество обработанных деталей равно 140, коэффициент загрузки обрабатывающего центра равен 0,932, максимальная очередь равна 3, средняя очередь равна 0,77.

Если же в этой программе установить одинаковый приоритет для обоих типов деталей, например, задав операнды E в первых двух блоках GENERATE по умолчанию, то при незначительном увеличении количества деталей (142) и коэффициента загрузки оборудования (0,959) резко возрастут максимальная очередь (до 7), что приведет к необходимости увеличения размеров накопителя деталей, и среднее содержимое очереди (до 2,731), которое вызовет дополнительные потери от пролеживания деталей.

Другие дисциплины обслуживания можно реализовать с помощью средств, которые будут рассмотрены при изучении различных режимов блока TRANSFER и блоков списка пользователя.

3.6. Моделирование многоканальных устройств

Комплекс однородных параллельно работающих одноканальных устройств с общим входом и общим выходом называют ***многоканальным устройством*** (МКУ). Примерами МКУ может быть бригада наладчиков, модуль из нескольких сборочных машин, участок станков с ЧПУ и т.п. Число одноканальных устройств, входящих в МКУ, называют его емкостью.

Работа МКУ в режиме его занятия и освобождения моделируется тремя операторами: командой STORAGE и блоками ENTER и LEAVE.

Команда STORAGE (хранилище) устанавливает емкость МКУ. Формат команды:

```
<метка>     STORAGE     A
```

В поле метки указывается символическое имя МКУ, в операнде A – целое положительное число, определяющее емкость МКУ.

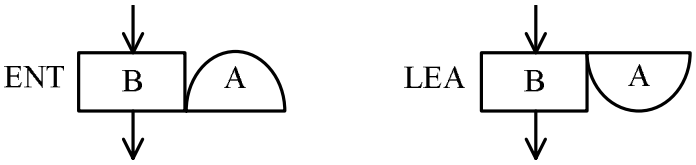
Блок ***ENTER*** (войти) моделирует занятие канала МКУ, а блок ***LEAVE*** (выйти) – освобождение канала МКУ.

Их форматы:

```
ENTER    A, [B]
LEAVE    A, [B]
```

В операнде A обоих блоков указывается имя МКУ, определенное в поле метки команды STORAGE. В операнде B указывается количество каналов, которые занимаются (блок ENTER) или освобождаются (блок LEAVE), по умолчанию B = 1.

Графическое изображение блоков:



Когда транзакт входит в блок ENTER, транслятор увеличивает на значение операнда B этого блока счетчик входов и текущее содержимое МКУ и уменьшает на это число его доступную емкость.

При входе транзакта в блок LEAVE уменьшается на значение операнда B текущее содержимое и увеличивается на это же число доступная емкость.

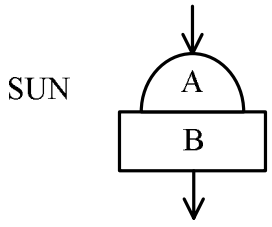
При моделировании автоматически формируется статистика о работе МКУ относительно счетчика входов, текущего содержимого, максимального содержимого, среднего значения занимаемых каналов и др. При этом надо иметь в виду, что вся информация формируется не по количеству транзактов, а по количеству каналов.

При исследовании систем с МКУ возможны различные дисциплины обслуживания: первым пришел – первым обслужен; первым пришел – первым обслужен внутри приоритетного класса и др.

Иногда возникает ситуация, когда МКУ становится недоступным (например, аварийное состояние устройства на его входе).

Недоступность МКУ моделируется блоком ***SUNAVAIL*** (S обозначает МКУ, AVAIL – доступно, UN – отрицание). Формат блока и изображение:

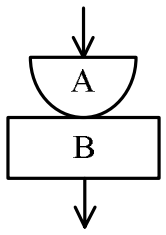
```
SUNAVAIL    A
```



Доступность МКУ восстанавливается после прохождения транзактом блока ***SAVAIL***. Его формат и изображение:

SAVAIL A

SAV



В операндах A этих блоков указывается имя МКУ (метка команды STORAGE).

Если в МКУ при его недоступности находились транзакты, то они будут обслуживаться, пока текущее содержимое МКУ не станет равным нулю.

Прежде чем рассматривать примеры с одноканальными и многоканальными устройствами изучим основные режимы оператора, позволяющего изменять маршруты движения транзактов в модели.

3.7. Изменение маршрутов движения транзактов

Для изменения маршрутов движения транзактов в модели используется блок ***TRANSFER*** (передать). Его полный формат:

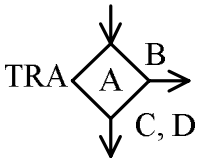
TRANSFER [A], [B], [C], [D]

Операнд A задает режим работы этого блока. Всего существует 9 режимов: **,** (по умолчанию) – безусловный; **·** (статистический) – случайный выбор одного из двух операторов; BOTH (последовательный) – выбор одного из двух операторов; ALL (последовательный) – выбор одного из нескольких операторов; PICK (случайный) – выбор одного из нескольких операторов; FN – функциональный; P – параметрический; SBR – подпрограммный; SIM – одновременный.

Операнды B и C задают адреса последующих по требуемому маршруту операторов.

Операнд D используется в режиме ALL для указания шага изменения положения последующих операторов.

Графическое изображение блока имеет вид:



Рассмотрим первые три режима блока TRANSFER, остальные режимы изложены в [1] и в лабораторном практикуме данного ЭУМК.

Режим безусловной передачи. В этом режиме операнд A задается по умолчанию, в операнде B указывается метка оператора, куда направляется транзакт. Например, запись TRANSFER ,ALFA означает, что транзакт, вошедший в данный блок, направляется в оператор с меткой ALFA.

Режим статистической передачи. В этом режиме в операнде A после точки задается число (до трех знаков), показывающее долю (от тысячи) транзактов, входящих в данный блок, которые направляются по адресу, указанному в операнде C, остальные пойдут по адресу, указанному в операнде B. Если B задан по умолчанию, то эта часть транзактов пойдет в следующий за блоком TRANSFER оператор. Например, TRANSFER . 200,, MET1 означает, что 20 % всех транзактов, вошедших в данный блок, пойдут по MET1, остальные 80 % – в следующий оператор.

Режим BOTH. В этом режиме в операнде A указывается слово BOTH (оба). Вошедший в данный блок транзакт последовательно пытается войти в операторы, адреса которых указаны в их полях метками, записанными в операндах B и C данного блока TRANSFER, пока попытка не будет успешной. Операнд B, как и в предыдущем режиме, может быть задан по умолчанию.

Пример 3.2. Изделия с интервалами (5 ± 2) мин. поступают в пункт технического контроля – ожидают в очереди – параллельно обслуживаются двумя контролерами по (9 ± 3) мин. – после этого 92 % уходят на упаковку – 8 % в очередь к наладчику – обслуживаются наладчиком в течение (30 ± 10) мин. и снова поступают в пункт технического контроля. Промоделировать работу участка в течение 5 недель при пятидневной неделе и двухсменном режиме работы. При единице модельного времени в 1 минуту время моделирования будет $5 \times 5 \times 2 \times 8 \times 6 = 24000$ мин.

Особенность модели: одно МКУ (2 контролера) и одно одноканальное устройство (1 наладчик).

Программа модели:

```
CON       STORAGE    2
          GENERATE   5, 2
BETTA     QUEUE      CONTR
          ENTER      CON
          DEPART     CONTR
          ADVANCE    9, 3
          LEAVE      CON
          TRANSFER   · 008, , ALFA
          TERMINATE
ALFA      QUEUE      NALAD
          SEIZE      NAL
          DEPART     NALAD
          ADVANCE    30,    10
          RELEASE    NAL
          TRANSFER   , BETTA
          GENERATE   24000
          TERMINATE  1
          START      1
```

3.8. Управляющие операторы очистки и сброса статистики

Команда управления CLEAR (очистить) используется для подготовки модели к повторному прогону после переопределения операндов одного или нескольких операторов. Формат команды:

CLEAR [A]

В операнде A может быть указано OFF, тогда обнуляется вся статистика, кроме сохраняемых величин и логических ключей. Если операнд A задан по умолчанию (соответствует символу ON), то обнуляется вся статистика. При действии команды CLEAR все транзакты выводятся из модели. В обоих случаях значения генераторов случайных чисел не сбрасываются.

Команда управления RESET (сбросить) применяется для многократных прогонов модели при изучении работы моделируемой системы в течение длительного периода времени, включающего как переходный, так и стационарный режимы. Этот оператор не имеет операндов. Команда RESET сбрасывает в ноль статистику и относительное модельное время. При этом абсолютное модельное время не сбрасывается. Транзакты не выводятся из модели, а счетчик текущих значений каждого блока устанавливается равным числу транзактов, находящихся в блоке. Не сбрасываются в исходное состояние генераторы случайных чисел.

Рассмотрим некоторые объекты, необходимые для построения более сложных моделей систем с рассматриваемыми одноканальными и многоканальными устройствами.

3.9. Стандартные числовые атрибуты (СЧА)

СЧА – это условные обозначения объектов модели, к которым могут обращаться программы GPSS в процессе моделирования. Необходимость в СЧА обусловлена стремлением к минимизации программ. Все СЧА можно подразделить на два класса: системные СЧА, к которым пользователь может обратиться, но не может изменить (например C1 – относительное модельное время, AC1 – абсолютное модельное время, RNj – случайное число, генерируемое датчиком случайных чисел и другие), и СЧА объектов модели, которые могут изменяться пользователем (СЧА устройств, МКУ, очередей, таблиц, блоков, транзактов, объектов вычислительной категории, сохраняемых величин и др.).

Для обозначения СЧА используют одну–две буквы, определяющие групповое имя объекта (например, Q – длина очереди, FN – функция) и идентификатор в зависимости от способа адресации. При прямой адресации задают объект константой, например, SEIZE 10, или СЧАj, где j – номер объекта или $ имя, а имя – символическое имя объекта, например: SEIZE P1 (имя объекта содержится в первом параметре активного транзакта), SEIZE P$ABC (имя объекта содержится в параметре с именем ABC активного транзакта). При косвенной адресации СЧА определяют так: СЧА*СЧА объекта, например SEIZE P*X10

означает, что номер устройства содержится в параметре (P), номер которого определяется значением сохраняемой ячейки (X10).

Рассмотрим СЧА наиболее распространенных объектов.

СЧА одноканальных устройств

Код	Значение
Fj	Занятость: 1 – занято, 0 – не занято
FIj	Прерванность: 1 – прервано, 0 – не прервано
FVj	Доступность: 1 – доступно, 0 – не доступно
FRj	Коэффициент использования в долях тысячи
FTj	Среднее время использования устройств одним транзактом
FCj	Количество занятий устройства

СЧА многоканальных устройств

Код	Значение
Rj	Доступная емкость
Sj	Текущее содержимое
SAj	Среднее значение содержимого за определенное время
SRj	Коэффициент использования в долях тысячи
SMj	Максимально занятое (одновременно) количество каналов
SCj	Счетчик числа входов
STj	Среднее время использования одного канала
SEj	Занятость: 1 – занято, 0 – не занято
SFj	Заполненность: 1 – заполнено, 0 – не заполнено
SVj	Доступность: 1 – доступно, 0 – не доступно

СЧА очередей

Код	Значение
Qj	Текущее содержимое
QAj	Средняя длина очереди
QMj	Максимальная длина очереди
QCj	Число входов в очередь
QZj	Число нулевых входов в очередь
QTj	Среднее время пребывания транзакта в очереди для всех транзактов
QXj	Среднее время пребывания транзакта в очереди без нулевых входов

СЧА блоков

Код	Значение
Nj	Счетчик входов
Wj	Текущее содержимое

СЧА других объектов будут рассматриваться при изучении соответствующих операторов.

3.10. Переменные пользователя

Переменные пользователя создаются командой ***EQU*** (эквивалентировать) или PLUS–процедурами.

Рассмотрим оператор EQU. Он имеет формат:

<метка> EQU A

Эта команда присваивает имени переменной, указанному в поле метки, или переменной пользователя, указанной в поле метки этого операнда, числовое значение.

Например, запись ALFA EQU 1 означает, что имя ALFA (допустим в операторе SEIZE ALFA) переименуется в имя цифровое (1) и при этом в выходной статистике будет FACILITY…, а не FACILITY… .

1 ALFA

Запись ABC EQU 32.5 означает, что переменной пользователя ABC присвоено числовое значение 32.5.

При исследовании влияния вариации какого–либо фактора на показатели системы необходимо использовать переменные пользователя.

3.11. Сохраняемые величины

В GPSS есть возможность использовать специальные ячейки памяти с начальными значениями, устанавливаемыми пользователем. Такие ячейки называют ***сохраняемыми величинами*** (сохраняемыми ячейками).

Рассмотрим ***линейные сохраняемые величины***. Их СЧА – Xj, где j – целое положительное число или $ имя, например, X1 или X$ABC. Сохраняемые величины могут использоваться в качестве операндов операторов, аргументов функций и таблиц. В начале моделирования все сохраняемые величины устанавливаются в нуль.

Для задания отдельным сохраняемым величинам ненулевых значений используют команду ***INITIAL*** (установить начальное значение). Ее формат:

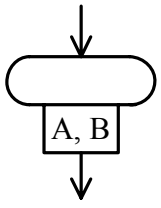
INITIAL A, [B]

В операнде A указывается Xj; в операнде B – первоначальное значение: по умолчанию «1»; может быть СЧА, число; может UNSPECIFIED (не определено). Например:

```
INITIAL     X10
INITIAL     X$OMEGA, Q1
INITIAL     41, UNSPECIFIED
```

Для изменения значения сохраняемой величины при работе модели используется блок ***SAVEVALUE*** (сохранить величину). Формат и графическое изображение этого блока:

SAVEVALUE A, B SAV



Операнд A задает цифровое или символическое имя (без указания символа группового имени). Так как блок может работать в трех режимах (замещения, приращения и уменьшения), то в первом случае в операнде A указывается

только имя сохраняемой величины, а в B – устанавливаемое значение (СЧА, число, строка в круглых скобках); во втором случае правым крайним знаком содержимого операнда A является знак «+» , тогда значение сохраняемой величины увеличивается на величину, указанную в операнде B; в третьем случае операнд A заканчивается знаком «–» и значение сохраняемой величины уменьшается на величину операнда B.

Примеры этих трех режимов:

```
SAVEVALUE     10, X$ALFA
SAVEVALUE     MET+, 286
SAVEVALUE     BCA–, (Q2 + Q3)
```

Рассмотрим ***матричные сохраняемые величины.***

Для задания исходной матрицы применяют команды: описания матрицы ***MATRIX*** и задания первоначальных значений ее элементов INITIAL.

Формат команды MATRIX:

```
<имя>    MATRIX     A, B, C, [D], [E], [F], [G]
```

В поле метки указывается символическое имя матрицы. Операнд A не используется (оставлен для совместимости с прежними версиями). Операнды B – G могут быть только целыми положительными числами, так как они задают соответственно: B – число строк, C – число столбцов матрицы во втором измерении, D, E, F, G – количество элементов в третьем, четвертом, пятом и шестом измерениях.

Наиболее распространен формат:

```
<имя>      MATRIX       , B, C
```

СЧА матриц MXj, где j – это $ имя. Если j – целое число, то оно должно быть определено предварительно через переменную пользователя. Например: матрица задана командой

```
ALFA          MATRIX      , 3, 3
```

Тогда ссылка на ее элемент, находящийся на пересечении первой строки и третьего столбца, имеет вид:

```
MX$ALFA (1, 3)
```

Если же необходимо изменить имя матрицы на числовое (например, 2), то справедлива запись:

```
ALFA     EQU          2
ALFA      MATRIX      , 3, 3
```

Ссылка на тот же элемент будет:

```
MX2 (1, 3)
```

Команда INITIAL устанавливает по желанию пользователя первоначальные значения ***элементов*** матриц, отличные от нуля. Формат команды тот же, что и для линейных сохраняемых величин, только в операнде A этой команды указывается СЧА элемента матрицы. Смысл операнда B такой же, но уже для элемента матрицы. Когда всем элементам матрицы необходимо присвоить значение, равное «1», в операнде A указывается MXj, а операнд B задается по умолчанию; если необходимо установить значение – «не определено», то в A указывается MXj, а в B – UNSPECIFIED. Примеры:

```
INITIAL     MX$ABC (2, 3), –328
INITIAL     MX10 (1, 6),  Q8
INITIAL     MX$MTD
INITIAL     MX$GAMMA, UNSPECIFIED
```

При этом запись во втором примере предполагает предварительное присвоение матрице цифрового имени с помощью оператора EQU.

В процессе моделирования значения элементов матрицы изменяются с помощью блока ***MSAVEVALUE***.

Его формат:

```
MSAVEVALUE      A, B, C, D
```

В операнде A задается имя матрицы, крайним правым знаком может быть «+» (режим приращения), знак «–» (режим уменьшения); операнд B задает номер строки; операнд C – номер столбца; в операнде D указывается заносимое, добавляемое или вычитаемое значение. Все операнды обязательны, могут быть заданы непосредственно или косвенно. Примеры:

```
MSAVEVALUE      1, 3, P5, 28.5
MSAVEVALUE      ALFA+, Q1, Q5, M1
MSAVEVALUE      P1, 1, (V$10 + Q2), P$OMEGA
MSAVEVALUE      8, STROKA, STOLBEZ, X2
```

Последняя строка означает, что значение, определенное сохраняемой величиной номер 2, записывается в элемент матрицы номер 8, находящийся на пересечении строки, определяемой переменной пользователя STROKA, и столбца, определенного переменной пользователя STOLBEZ. При этом номер матрицы также определен предварительно с помощью оператора EQU.

Матрицы более высоких порядков (3 и более) создаются с помощью PLUS–процедур [1].

3.12. Примеры

Пример 3.3. Рассмотрим участок сборки деталей с обжигом при 4, 5 и 6 транзактах–сборщиках. Пусть модельное время равно 5 дням по 8 часов в день и при единице модельного времени, равной 1 мин., составляет 2400 единиц.

Программа с использованием переменных пользователя, сохраняемых величин и команды CLEAR представлена ниже.

```
KSB     EQU       4; Число сборщиков равно 4
        GENERATE  ,,, KSB
SBOR    ADVANCE   30,5; Сборка деталей
        SEIZE     OTO
        ADVANCE   8,2; Обжиг деталей
        RELEASE   OTO
        TRANSFER  ,SBOR
        GENERATE  2400
        TERMINATE 1
        START     1
KSB     EQU       5; Число сборщиков равно 5
        GLEAR
        START     1
KSB     EQU       6; Число сборщиков равно 6
        GLEAR
        START     1
```

Выходная статистика относительно количества готовых деталей и коэффициента использования печи при четырех, пяти, шести сборщиках соответственно имеет вид:

```
FACILITY   ENTRIES    UTIL.
OTO          239      0.798

FACILITY   ENTRIES    UTIL.
 OTO         287      0.950

FACILITY    ENTRIES   UTIL.
OTO          297      0.989
```

Из этой информации видно, что с увеличением числа сборщиков количество готовых деталей (ENTRIES) и коэффициент использования печи (UTIL.) растут.

Окончательный вывод об эффективности участка можно сделать после расчета других технико–экономических показателей (зарплата сборщиков, стоимость оборудования, стоимость изделия, затраты на материалы и др.).

Пример 3.4. В цехе 50 станков с ЧПУ работают 50 недель по 5 дней в неделю при односменном режиме. Наработка на отказ станка (150 ± 25) час. Время на ремонт одного станка равно (7 ± 3) час. Станки, вышедшие из строя, ремонтируются несколькими наладчиками. После этого отремонтированные станки пополняют резерв и включаются в производство. Схема работы цеха изображена на рис. 3.2.

Необходимо составить модель, предусмотрев информацию о загрузке станков с ЧПУ и наладчиков при различном числе наладчиков (3, 4, 5) и различном количестве резервных машин (3, 4, 5).

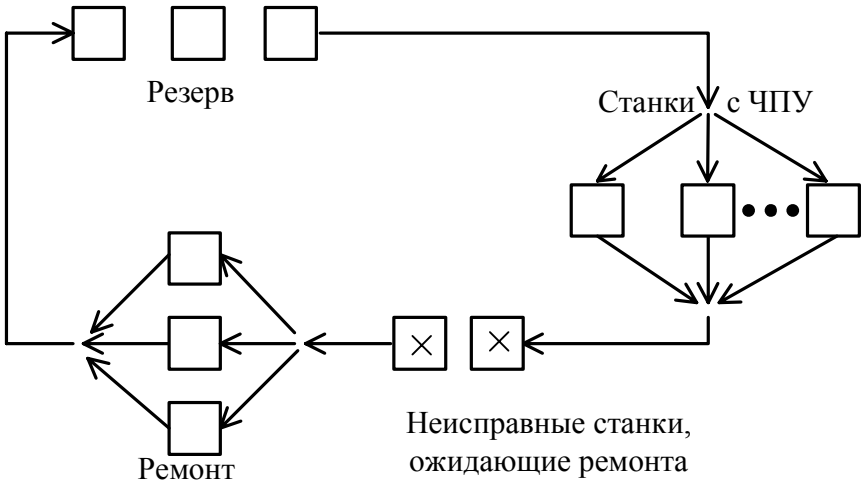


Рис. 3.2. Схема работы цеха

При единице модельного времени в 1 час модельное время будет 50 × 5 × 8 = 2000 единиц. Особенности модели: два многоканальных устройства (наладчики и станки); в процессе моделирования переопределяются операнды, связанные с числом наладчиков и количеством резервных машин; информация о работе станков с ЧПУ заносится в матрицу.

Программа модели имеет вид:

```
; Задание исходных данных
KRS       EQU                 3  ;  Количество резервных станков = 3
STROK     EQU                 1  ;  Номер строки матрицы
STOLB     EQU 1  ;            Номер столбца матрицы
ZAGRS     MATRIX              , 3, 3;       Матрица загрузки станков
ZAGRN     MATRIX              , 3, 3;       Матрица загрузки наладчиков
STAN      STORAGE             50;  Количество станков = 50
NAL       STORAGE             3  ;Число наладчиков = 3

; Моделирование работы цеха
          GENERATE            , , , (50 + KRS)
MET1      ENTER               STAN
          ADVANCE             150, 25
          LEAVE               STAN
          ENTER               NAL
          ADVANCE             7, 3
          LEAVE               NAL
          TRANSFER            , MET1

; Моделирование завершения работы цеха и формирования результатов
          GENERATE            2000
          MSAVEVALUE          ZAGRS, STROK, STOLB, (SR $ STAN/1000)
          MSAVEVALUE          ZAGRN, STROK, STOLB, (SR $ NAL/1000)
          TERMINATE           1
          START               1, NP; Резервных станков 3, наладчиков 3
KRS       EQU                 4
STOLB     EQU                 2
          CLEAR               OFF
          START               1, NP;  Резервных станков 4, наладчиков 3
```

```
KRS       EQU              5
STOLB     EQU              3
          CLEAR            OFF
          START            1, NP;  Резервных станков 5, наладчиков 3
NAL       STORAGE          4
KRS       EQU              3
STROK     EQU              2
STOLB     EQU              1
          CLEAR            OFF
          START            1, NP;  Резервных станков 3, наладчиков 4
KRS       EQU              4
STOLB     EQU              2
          CLEAR            OFF
          START            1, NP;  Резервных станков 4, наладчиков 4
KRS       EQU              5
STOLB     EQU              3
          CLEAR            OFF
          START            1, NP;  Резервных станков 5, наладчиков 4
NAL       STORAGE          5
KRS       EQU              3
STROK     EQU              3
STOLB     EQU              1
          CLEAR            OFF
          START            1, NP;  Резервных станков 3, наладчиков 5
KRS       EQU              4
STOLB     EQU              2
          CLEAR            OFF
          START            1, NP;  Резервных станков 4, наладчиков 5
KRS       EQU              5
STOLB     EQU              3
          CLEAR            OFF
          START            1;   Резервных станков 5, наладчиков 5
```

В этой модели количество резервных машин переопределялось с помощью переменной пользователя KRS EQU, а число наладчиков через команду NAL STORAGE.

Коэффициенты использования МКУ станков и наладчиков через СЧА соответственно SR $ STAN и SR $ NAL для всех девяти вариантов занесены в матрицы:

```
MATRIX          RETRY          INDICES          VALUE
  ZAGRS           0
                                 1   1           . 971
                                 1   2           . 981
                                 1   3           . 981
                                 2   1           . 982
                                 2   2           . 986
                                 2   3           . 987
                                 3   1           . 984
                                 3   2           . 989
                                 3   3           . 990
```

```
ZAGRN          0
                                    1   1          . 728
                                    1   2          . 721
                                    1   3          . 743
                                    2   1          . 558
                                    2   2          . 547
                                    2   3          . 563
                                    3   1          . 445
                                    3   2          . 434
                                    3   3          . 447
```

Пример 3.5. В предыдущем примере в самом начале работы цеха (до 150 – 25 = 125 час.) все станки будут исправны. Следовательно статистика будет не достоверной. Для оценки переходного режима промоделируем работу цеха при четырех наладчиках и четырех резервных станках в течение десяти декад со сбросом статистики с помощью команды RESET через каждую декаду, т.е. через 80 час.

Программа модели имеет вид:

```
KRS       EQU       4;          Количество резервных станков = 4
STROK     EQU       1;          Номер строки матрицы
STOLB     EQU       1;          Номер столбца матрицы
ZAGRS     MATRIX    , 10, 1;    Матрица загрузки станков
ZAGRN     MATRIX    , 10, 1;    Матрица загрузки наладчиков
STAN      STORAGE   50;         Количество станков = 50
NAL       STORAGE   4;          Число наладчиков = 4

          GENERATE  ,,,(50+KRS)
MET1      ENTER     STAN
          ADVANCE   150,25
          LEAVE     STAN
          ENTER     NAL
          ADVANCE   7,3
          LEAVE     NAL
          TRANSFER  ,MET1

          GENERATE  80
          MSAVEVALUE ZAGRS, STROK, STOLB, (SR$STAN/1000)
          MSAVEVALUE ZAGRN, STROK, STOLB, (SR$NAL/1000)
          TERMINATE 1
          START     1,NP
STROK     EQU       2
          RESET
          START     1,NP
STROK     EQU       3
          RESET
          START     1,NP
STROK     EQU       4
          RESET
          START     1,NP
STROK     EQU       5
          RESET
```

```
        START   1,NP
STROK   EQU     6
        RESET
        START   1,NP
STROK   EQU     7
        RESET
        START   1,NP
STROK   EQU     8
        RESET
        START   1,NP
STROK   EQU     9
        RESET
        START   1,NP
STROK   EQU     10
        RESET
        START   1
```

Результаты моделирования занесены в следующие матрицы:

MATRIX	RETRY	INDICES	VALUE
ZAGRS	0		
		1 1	1
		2 1	. 869
		3 1	. 836
		4 1	. 988
		5 1	. 996
		6 1	. 996
		7 1	. 997
		8 1	. 974
		9 1	. 998
		10 1	. 989
ZAGRN	0		
		1 1	0
		2 1	. 972
		3 1	. 713
		4 1	. 694
		5 1	. 629
		6 1	. 591
		7 1	. 481
		8 1	. 742
		9 1	. 458
		10 1	. 793

Анализ этой информации показывает, что в течение десяти декад режим переходный (загрузка станков – неустановившаяся). В связи с этим для получения достоверных результатов рекомендуется в команде START в операнде A заносить не единицу, а число 10. В этом случае итоги моделирования следующие:

MATRIX	RETRY	INDICES	VALUE
ZAGRS	0		
		1 1	. 972
		2 1	. 997
		3 1	. 998
		4 1	. 996
		5 1	. 996
		6 1	. 998
		7 1	. 996
		8 1	. 995
		9 1	. 997
		10 1	. 995
ZAGRN	0		
		1 1	. 529
		2 1	. 556
		3 1	. 593
		4 1	. 576
		5 1	. 588
		6 1	. 563
		7 1	. 571
		8 1	. 607
		9 1	. 581
		10 1	. 589

Из этих результатов видно, что после десяти декад процесс относительно работы основного оборудования установился (загрузка комплекса станков более 99 %).

4. СОЗДАНИЕ МОДЕЛЕЙ СИСТЕМ С ИСПОЛЬЗОВАНИЕМ ОБЪЕКТОВ ВЫЧИСЛИТЕЛЬНОЙ КАТЕГОРИИ

В этом разделе рассматриваются генераторы случайных величин, функции и переменные, а также ряд операторов, необходимых для применения перечисленных объектов.

4.1. Генераторы случайных чисел

Источником случайности в моделях являются генераторы случайных чисел. Их СЧА RNn, где n – номер генератора (совпадающий с его начальным числом). Ограничения на количество генераторов нет. Первые семь генераторов (RN1 – RN7) являются управляемыми. Пользователь может в необходимых случаях установить их начальные значения. При этом используется команда ***PMULT*** (установить значение генератора). Ее формат:

PPMULT [A], [B], [C], [D], [E], [F], [G]

В операндах, заданных явно целыми положительными числами, устанавливаются необходимые начальные значения генераторов, остальные остаются без изменений. Например,

PMULT , , 32, , 40, , 72

устанавливает новые значения генераторов с номерами 3, 5, 7.

Все генераторы выдают равномерно распределенные числа в интервале от 0 до 0,999999 при вычислении функций и от 0 до 999 включительно – в остальных случаях.

4.2. Использование дискретных равномерных распределений

Рассмотренные до сих пор интервалы генерации и обработки транзактов подчинялись равномерному закону распределения. Они задавались с помощью среднего значения и «полуразмаха» интервала, например:

```
GENERATE    20, 5
ADVANCE     30, 2
```

Закон равномерного распределения можно задать, используя библиотечный генератор дискретно-равномерного распределения ***DUNIFORM***. Его формат:

DUNIFORM (n, min, max)

Здесь n – номер генератора, min – левое крайнее, max – правое крайнее значения диапазона.

Для приведенных примеров в этой записи

```
GENERATE    (DUNIFORM (1, 15, 25))
ADVANCE     (DUNIFORM (1, 28, 32))
```

Перечень библиотечных генераторов других распределений приведен в [1].

При использовании равномерно–распределенных чисел в переменных и других объектах рекомендуется вычислять равномерное распределение по алгоритму

$$A - B + (2 \# B \# RNn) / 999, \qquad (4.1)$$

где A и B – соответственно среднее значение и полуразмах диапазона.

4.3. Использование дискретных неравномерных распределений (функции типа D)

Предположим, что случайная переменная принимает значения 2, 5, 6, 8, 10 с относительной частотой 0,1; 0,2; 0,25; 0,12; 0,33 соответственно, как изображено в таблице:

Значение случайной величины	Относительная частота	Суммарная частота	Диапазон	Номер интервала
2	0,1	0,1	0 … 0,1	1
5	0,2	0,3	0,1+ … 0,3	2
6	0,25	0,55	0,3+ … 0,55	3
8	0,12	0,67	0,55+ … 0,67	4
10	0,33	1,00	0,67+ … 1,00	5

Пусть разыгранное число из интервала [0, 1] будет 0,328153. Оно попало в интервал № 3 и значение случайной величины равно 6.

Таким образом, для розыгрыша случайного числа в соответствии с распределением, определенным таблицей, необходимо иметь источник случайных чисел в интервале [0, 1] и суммарную частоту появления значений переменной. Эта информация задается с помощью дискретной функции ***FUNCTION*** (определить функцию). Ее формат:

```
<имя >          FUNCTION        RNn, Dm
X1, Y1 /…/ Xm, Ym
```

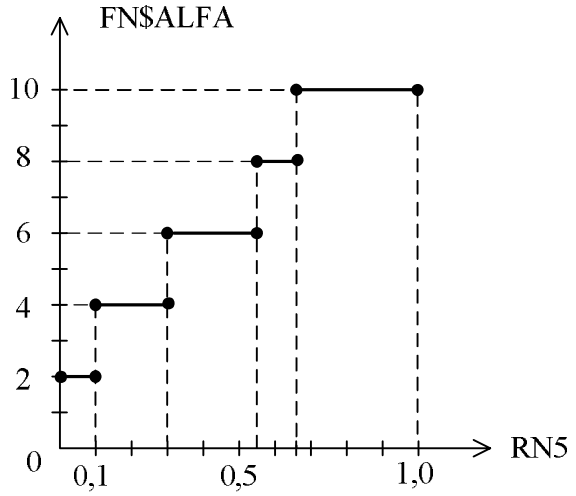
В поле метки записывается символическое имя функции, в операнде A – генератор случайных чисел, в B – символ D и количество пар значений аргумента и функции, во второй строке – значения суммарной частоты (аргумента) и случайной величины (функции). Эти пары разделяются наклонной чер-

той, причем значения суммарной частоты должны возрастать. Функция записывается в самом начале программы (до первого оператора GENERATE).

Для значений, отраженных в рассматриваемой таблице, функция, например, с именем ALFA, имеет вид:

```
ALFA      FUNCTION          RN5,  D5
. 1, 2  /. 3, 5 /. 55, 6 /. 67, 8 / 1, 10
```

График этого распределения будет следующим:



При использовании дискретных неравномерных распределений в качестве аргумента можно задавать не только СЧА генераторов, а и других объектов, например, параметров транзактов (Pj), блоков (Wj или Nj) и т.п. Например,

```
BETTA        FUNCTION   P1, D4
 2, 10 / 3, 12 / 4, 15 / 5, 20
```

С помощью функции типа D можно задать и равномерное распределение. Например, значения переменной 2, 5, 6 и 8 выпадают с равной частотой, равной 0,25. Тогда дискретная функция будет:

```
ABC   FUNCTION       RN1, D4
. 25, 2 / . 5, 5 / . 75, 6 / 1, 8
```

4.4. Использование непрерывных распределений (функции типа C)

Пусть следующая таблица задает значения частоты и соответствующий диапазон случайной величины:

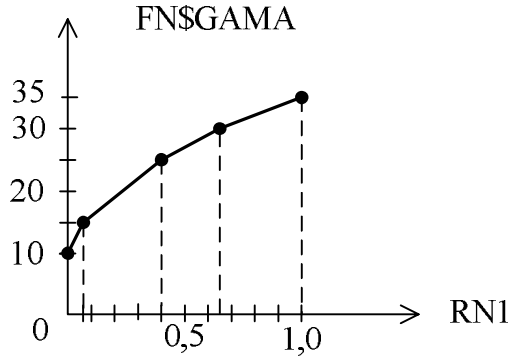
Относительная частота	Суммарная частота	Диапазон случайной величины
0	0	< 10
0,05	0,05	10 … 15 –
0,35	0,40	15 … 25 –
0,25	0,65	25 … 30 –
0,35	1,00	30 … 35 –

При моделировании непрерывных распределений выполняется линейная интерполяция для пары точек, находящихся по краям того интервала значений суммарной величины, на который выпало значение аргумента.

Непрерывная неравномерная функция будет иметь вид:

```
GAMA        FUNCTION      RN1, C5
0, 10 / . 05, 15 / . 4, 25 / . 65, 30 / 1, 35
```

График этой зависимости следующий:



Функция типа C может применяться и для моделирования дискретных распределений. Если случайная переменная распределена равномерно и непрерывно на интервале, например, от 3 до 10, то ее можно задать так:

```
ACD     FUNCTION        RN5, C2
0, 3 / 1, 10
```

Так как наибольшее значение RN5 есть 0,999999, то значение переменной ACD никогда не достигнет величины 10, т.е. наибольшее целое число будет 9.

Функцию типа C целесообразно применять для задания равномерного распределения целых чисел в широком диапазоне. Например, если надо задать такие значения от 100 до 850 включительно, то можно записать:

```
KKK     FUNCTION    RN25, C2
0, 100 / 1, 851
```

4.5. Функции типа E, L, M

Если в дискретной функции в качестве значений переменной используются и СЧА, то при определении такой функции в операнде B указывается символ E. Такую функцию называют атрибутивно–значимой. Например,

```
MET    FUNCTION      P1, E3
2, V$ALFA / 5, V$BETTA / 8, V$GAMA
```

В этом случае при соответствующем значении аргумента (допустим P1=3) вычисляется значение случайной переменной функции MET через арифметическую переменную ALFA.

Если в дискретной функции типа D аргумент принимает последовательно целые значения 1, 2, 3,…, то в операнде B указывается символ L, а функцию называют списковой. Например:

```
AAA     FUNCTION        N$ALFA, L4
1,25 / 2, 40 / 3, 60 / 4, 80
```

Если же в списковой функции в качестве значений переменной используются и СЧА, то в операнде B указывается символ M, а функция называется списковой атрибутивно–значимой. Например:

```
KAP       FUNCTION     P5, M4
1, Q1 / 2, Q2 / 3, Q3 / 4, 12
```

4.6. Моделирование пуассоновских потоков

Часто в СМО интервалы времени поступления и обслуживания требований (транзактов) подчиняются пуассоновскому закону. При их моделировании можно использовать в блоках GENERATE и ADVANCE в операнде A – среднее значение, а в операнде B – экспоненциальную функцию XPDIS с единичным средним значением. При этом значение времени поступления или задержки транзакта вычисляется так:

```
A#FN$XPDIS,                                                    (4.2)
```

где функция типа C с именем XPDIS задается 24 парами чисел:

```
XPDIS   FUNCTION          RN1, C24
0,0 /.1,.104 /.2,.222 /.3,.355 /.4,.509 /.5,.69 /.6,.915 /.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
```

Пусть, например, распределение поступления транзактов экспоненциальное, среднее значение равно 80 единицам. Это моделируется блоком:

```
GENERATE       80, FN $ XPDIS
```

Если распределение времени обработки транзактов также экспоненциальное при среднем значении 42, то эта ситуация моделируется блоком:

```
ADVANCE        42, FN $XPDIS
```

В общих случаях предварительно (в начале программы) должна быть задана функция XPDIS.

В GPSS World экспоненциальное распределение можно задать с помощью библиотечного генератора оператором ***EXPONENTIAL***, формат которого

```
EXPONENTIAL   (A, B, C)
```

Здесь A – номер генератора случайных чисел, B – величина сдвига, C – среднее значение, причем B и C – вещественные значения.

Например, вместо приведенных выше операторов можно записать:

```
GENERATE    (EXPONENTIAL (1, 0, 80))
ADVANCE     (EXPONENTIAL (5, 0, 42))
```

4.7. Арифметические переменные

Арифметическая переменная предназначена для вычисления арифметических выражений. Она определяется командой ***VARIABLE*** или ***FLARIABLE*** (переменная). В GPSS World различий между этими командами нет. В GPSS PC первая соответствует переменной с плавающей точкой (при этом отбрасывается дробная часть как в окончательном выражении, так и в промежуточных вычислениях), вторая – переменной с фиксированной точкой (дробная часть отбрасывается только в окончательном результате).

Формат переменной:

<имя> VARIABLE <выражение>

В поле метки указывается символическое имя, в поле операндов – выражение. Выражение – это набор данных, связанных арифметическими операциями. В GPSS World допустимы операции:

^ – возведение в степень (x^y соответствует x^y),

– умножение,

/ – деление,

\ – целочисленное деление (результат – целое число),

@ – деление по модулю (13 @ 3 = 1, результат остаток),

+ – сложение.

Порядок вычисления: ^, #, /, \, @, +, –. Допускаются круглые скобки, при этом вычисляется выражение в этих скобках.

Примеры:

```
ABC     VARIABLE        (P2 – P4) / 2
KAP     VARIABLE        (Q$OTO + Q$ALK) # 28.5
```

Арифметическая переменная имеет СЧА V$ имя.

4.8. Моделирование нормального распределения

Случайная величина с нормальным распределением однозначно описывается заданием математического ожидания (среднего) и стандартного отклонения. При моделировании такого распределения применяют выражение:

$$GNORM = GNORM_{СТ.ОТ.} \# SNORM + GNORM_{М.ОЖ.}, \quad (4.3)$$

где GNORM – выборка из нормального распределения; SNORM – выборка из нормированного нормального распределения, имеющего нулевое матожидание и единичное стандартное отклонение; $GNORM_{СТ.ОТ.}$ и $GNORM_{М.ОЖ.}$ – реальные стандартное отклонение и матожидание соответственно.

Нормированная выборка задается функцией типа C с именем SNORM с использованием 25 пар чисел:

```
 SNORM        FUNCTION      RN1,  C25
0, –5 /.00003, –4 /.00135, –3 /.00621, –2.5 /.02275, –2 /.06681, –1,5
.11507, –1.2 /.15866, –1 /.21186, –.8 /.27425, –.6 /.34458, –.4 /.42074,–.2
.5, 0 /.57926, .2 /.65542, .4 /.72575, .6 /.78814, .8 /.84134, 1/.88493, 1.2
.93319, 1.5 /.97725, 2 /.99379, 2.5 /.99865, 3 /.99997, 4 / 1,5
```

При использовании этого закона распределения определяется арифметическая переменная GNORM, например, при стандартном отклонении $GNORM_{СТ.ОТ.} = 5$ и матожидании $GNORM_{М.ОЖ.} = 20$ следующим образом:

```
GNORM        FVARIABLE    5#FN$ SNORM + 20
```

Далее в программе делается ссылка на эту переменную, например:

```
GENERATE   V $ GNORM
```

В связи с тем, что часто реальные случайные величины (временные интервалы и т.п.) положительны, необходимо следить за тем, чтобы матожидание по меньшей мере в 5 раз превышало стандартное отклонение.

Нормальное распределение можно промоделировать с помощью библиотечного генератора с использованием оператора NORMAL. Его формат:

```
NORMAL (A, B, C)
```

В A указывается номер генератора случайных чисел, в B – среднее значение, в C – стандартное отклонение. Например, для рассмотренного случая:

```
GENERATE       (NORMAL (1, 20, 5))
```

4.9. Булевы переменные

Булева переменная предназначена для вычисления логических выражений, которые могут принимать одно из двух значений: «истина» или «ложь» (величина которого 1 или 0 соответственно). Эта переменная определяется командой ***BVARIABLE*** (булева переменная). Ее формат:

```
<имя>          BVARIABLE          <выражение>
```

В поле метки указывается символическое имя, в поле операндов – выражение. В выражении могут использоваться операторы трех типов: логические, отношения и булевы.

Логические операторы используются для ссылок на логическое состояние устройств, МКУ и ключей. Наиболее распространенные операторы:

FVj – устройство используется (1), в противном случае – 0;
FIj – устройство прервано (1), в противном случае – 0;
SFj – МКУ заполнено (1), в противном случае – 0;
SEj – МКУ пусто (1), в противном случае – 0;
LSj – ключ включен (1), в противном случае – 0;

Операторы отношения сравнивают численные величины:
'G' – больше,
'GE' – больше или равно,
'E' – равно,
'NE' – не равно,
'LE' – меньше или равно,
'L' – меньше.

Булевы операторы выполняют условия: 'AND' – И, 'OR' – ИЛИ.

Порядок выполнения: логические операторы и операторы отношения, затем – булевы операторы. Если есть круглые скобки, то сначала производятся вычисления в них. Например, выражение

```
KAPPA        BVARIABLE        C1 'GE' 500 ' AND' LS $ ALFA
```

истинно, если относительное модельное время больше или равно 500 и логический ключ ALFA включен.

В случае, если булева переменная задана через арифметическую переменную, то значение булевой переменной 0, если арифметическая переменная равна 0, в остальных случаях значение булевой переменной равно 1.

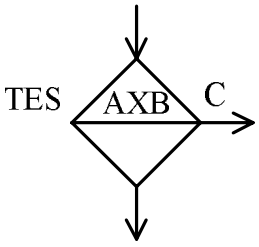
Булева переменная имеет СЧА: BVj, для приведенного примера это BV$KAPPA.

4.10. Проверка числовых выражений

Блок ***TEST*** проверяет соотношение между операндами A и B через дополнительный операнд X, который может принимать такие же значения, как и операторы отношении (только апострофы не ставятся). Формат блока:

```
TEST     X              A, B, [C]
```

Графическое изображение:



Этот блок может работать в двух режимах: отказа (клапана) и условного перехода. В первом режиме операнд C не используется. Транзакт пойдет в следующий оператор только при выполнении условия, заданного операндом X. Например, при записи TEST E Q1, 10 транзакт пойдет дальше, когда содержимое очереди № 1 будет равно 10. Во втором режиме в случае выполнения условия транзакт пойдет в следующий оператор, а в случае невыполнения – по метке, указанной в операнде C. Например, TEST G P5, V$AAA, MET1 означает, что при условии, когда значение пятого параметра вошедшего тракзакта

больше значения арифметической переменной AAA, транзакт пойдет в следующий оператор, если нет – то по адресу MET1.

Блок TEST может использоваться и для проверки булевых переменных. Например, TEST E BV $ DDD, 1 означает, что транзакт пойдет в следующий оператор, только тогда, когда значение булевой переменной DDD станет равным единице.

4.11. Примеры

Пример 4.1. Пусть производственный участок работает 1 смену (28800 с). Поступающие детали подчиняются пуассоновскому закону с интенсивностью 12 поступлений в час (т.е. 300 с на 1 поступление). Обработка на одном станке с экспоненциальным временем, но среднее время t зависит от количества деталей в очереди ℓ: при отсутствии очереди t=330 с; при ℓ = 1 и 2 время t=300 с; при ℓ = 3; 4; 5 t=270 с; при $\ell \geq 6$ t=240 с.

Программа модели при единице модельного времени, равной 1 с, будет иметь вид:

```
QVEF          FUNCTION        Q$ABC, D4
0,330 / 2, 300 / 5, 270 / 6, 240
              GENERATE        (EXPONENTIAL (1, 0, 300))
              QUEUE           ABC
              SEIZE           OTO
              DEPART          ABC
              ADVANCE         (EXPONENTIAL (1,0, FN$QVEF))
              RELEASE         OTO
              TERMINATE
              GENERATE        28800
              TERMINATE       1
              START           1
```

Эта программа чувствительна к длине очереди ℓ.

Пример 4.2. Рассмотрим процесс сборки деталей с обжигом (см. пример 3.3). Пусть время сборки (30 ± 5) мин, время обжига (8 ± 2) мин, зарплата сборщика 30 единиц в день, стоимость печи 80 единиц в день, стоимость детали – 5 единиц. Рассчитать среднюю дневную прибыль, в одном прогоне модели исследовать случаи с четырьмя, пятью и шестью сборщиками. Модельное время – 1 неделя.

Определим операнд D в блоке GENERATE косвенно через сохраняемую величину с именем RAB.

Средняя дневная прибыль = 5 # число ежедневно изготовленных деталей – дневные расходы.

Если в поле метки блока RELEASE OTO записать имя PLAN, то N $ PLAN есть число готовых деталей за весь плановый период, т.е. за 5 дней, а N $ PLAN/5 – за 1 день.

Затраты за 1 день = стоимость печи (80) + зарплата рабочих (30 # X$RAB).

Следовательно, дневная прибыль, выраженная арифметической переменной с именем PRIB будет:

```
PRIB    VARIABLE    5 # N$PLAN / 5–80–30 # X$RAB
```

Для подсчета средней дневной выручки в модуль таймера поместим блок SAVEVALUE с операндами A = INDEX и B = V$PRIB.

```
        INITIAL     X$RAB, 4
PRIB    VARIABLE    5#N$PLAN/5–80–30#X$RAB
        GENERATE    ,,, X$RAB
SBOR    ADVANCE     30, 5
        SEIZE       OTO
        ADVANCE     8, 2
PLAN    RELEASE     OTO
        TRANSFER    , SBOR
        GENERATE    2400
        SAVEVALUE   INDEX, V$PRIB
        TERMINATE   1
        START       1
        INITIAL     X$RAB, 5
        CLEAR       OFF
        START       1
        INITIAL     X$RAB, 6
        CLEAR       OFF
        START       1
```

Выходные данные, касающиеся прибыли:

SAVEVALUE	VALUE	SAVEVALUE	VALUE
RAB	4	INDEX	35
RAB	5	INDEX	50
RAB	6	INDEX	35

Эта статистика показывает, что при пяти сборщиках будет наибольшая дневная прибыль.

5. ИСПОЛЬЗОВАНИЕ СРЕДСТВ РАЦИОНАЛЬНОГО ПОСТРОЕНИЯ МОДЕЛЕЙ

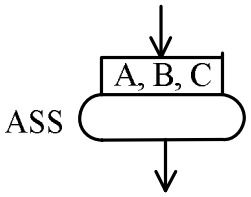
В этом разделе будут рассмотрены некоторые средства, позволяющие упростить модели при их большой размерности, сложных дисциплинах обслуживания транзактов, необходимости компактного представления результатов моделирования.

5.1. Параметры транзактов, изменение их значений и уровня приоритетов

Каждый транзакт может иметь по желанию пользователя любое число параметров. При появлении в модели транзакты имеют нулевые значения параметров.

Для задания значений параметров используют блок ***ASSIGN*** (назначить). Его формат и графическое изображение:

ASSIGN A, B, [C]



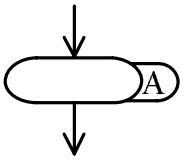
В операнде A указывается номер параметра (число, СЧА, СЧА* параметр и может быть знак «+» или «–» в зависимости от режимов: замещения, приращения, уменьшения). В операнде B указывается новое значение, в C – модификатор–функция. Например:

```
ASSIGN      1, 285
ASSIGN      ABC+, Q10
```

В первом случае значение параметра № 1 вошедшего транзакта заменяется на число 285, во втором – к значению вошедшего транзакта с именем ABC прибавляется значение текущего содержимого очереди № 10.

Для задания в транзакт или его параметр абсолютного модельного времени используют блок ***MARK*** (отметить). Формат и графическое изображение блока:

MARK [A]

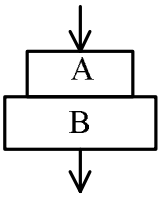


В операнде A указывается номер параметра, в который заносится значение абсолютного модельного времени, по умолчанию это время устанавливается вошедшему транзакту.

Есть и другие блоки для изменения значения параметров транзакта [1].

Для изменения уровня приоритета транзакта применяют блок ***PRIORITY*** (установить приоритет). Формат и графическое изображение:

PRIORITY A, [B]



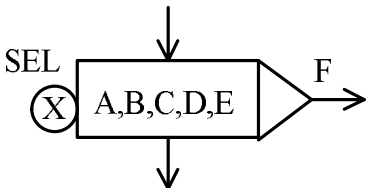
В операнде A указывается новый уровень приоритета (целое число, СЧА, СЧА*СЧА).

В операнде B может быть указан режим BU (т.е. BUFFER), когда транзакты переводятся в список текущих событий для возобновления просмотра этого списка.

5.2. Системы с параллельно работающими идентичными устройствами и раздельными очередями

Часто возникают ситуации, когда в системе с несколькими очередями и устройствами реализуется такая дисциплина обслуживания: если одно из устройств свободно, то транзакт занимает его, если все устройства заняты, то транзакт присоединяется к самой короткой очереди. Такую стратегию можно промоделировать с помощью блока ***SELECT*** (выбрать) в условном режиме. Формат блока и графическое изображение:

SELECT X A, B, C, [D], [E], [F]



В условном режиме операнд X может принимать значения: G, GE, E, L, LE, NE, MIN, MAX, NE.MAX, NE.MIN. Здесь первые шесть значений аналогичны значениям операнда X блока TEST. Символы MAX и MIN означают наибольшее и наименьшее значения, NE.MAX и NE.MIN – ненаибольшее и ненаименьшее значения.

При использовании блока SELECT в логическом режиме операнд X принимает значения, определяющие логические условия работы устройств и МКУ (с ними можно ознакомиться в [1]).

Операнды задают: E – групповое имя (СЧА) проверяемых объектов; B и C – наименьший и наибольший номера из проверяемых объектов; D – значение, с которым сравнивается атрибут (в случае, когда X = MIN (MAX) и X = NE.MIN (NE.MAX), D – не задается); A – номер параметра, в который заносится номер найденного типа группы; F – метка оператора, куда пойдет транзакт, если искомый объект не обнаружен.

Пусть, например, на участке работают восемь идентичных станков с приставочными столами для накопления заготовок в случае занятости станков. Заготовка попадает на свободный станок; если таких нет, то присоединяется к наименьшей очереди. Эту ситуацию можно промоделировать так:

```
                ⋮
            SELECT E        2, 1, 8, 0, F, ALFA
ABC         QUEUE           P2
            SEIZE           P2
                ⋮
ALFA        SELECT MIN      2, 1, 8, ,Q
            TRANSFER        , ABC
                ⋮
```

В этом фрагменте устройства (с 1 по 8) проверяются на предмет свободы (F→E→O), если такое устройство есть, то его номер записывается во второй параметр активного транзакта, он проходит дальше (с нулевым вхождением в очередь) и попадает в свободное устройство с именем, определенным в P2, и т.д.

Если свободного устройства нет, то транзакт идет по метке ALFA к блоку SELECT в режиме MIN и после проверки содержимого очередей (Q) с первой по восьмую, присоединяется к наименьшей через оператор TRANSFER , ABC (номер минимальной очереди фиксируется также во втором параметре активного транзакта). При этом будет сформирована статистика относительно всех восьми устройств и очередей к ним, хотя в программе используется одна последовательность операторов.

5.3. Моделирование таблиц

При исследовании систем возникает необходимость определения для какой–либо выборки среднего значения, стандартного отклонения, количества элементов, попадающих в установленные интервалы и т.п. информации. Такая табуляция осуществляется с использованием операторов ***TABLE*** (таблица) и ***TABULATE*** (табулировать).

Команда TABLE имеет формат:

```
<имя>          TABLE          A, B, C, D
```

В поле метки записывается символическое имя таблицы. В операнде A задается СЧА табулируемого элемента, в B – верхний предел первого интервала, в C – ширина промежуточных интервалов, в D – общее число интервалов. Например, MET1 TABLE M1, 10, 100, 8 задает таблицу с именем MET1 для табулирования времени пребывания в модели активного транзакта M1 при верхней границе первого значения интервала, равного 10, ширине интервалов в 100 единиц и общем числе интервалов таблицы, равном 8. Эта команда описывается в начале программы.

Блок TABULATE имеет формат:

TABULATE A, [B]

В операнде A указывается имя таблицы, в которую заносятся табулируемые значения. В операнде B указывается весовой коэффициент, т.е. количество единиц, заносимых в интервал таблицы, по умолчанию – это значение равно единице.

Для использования описанной ранее таблицы этот блок будет:

TABULATE MET1

Блок TABULATE помещается в точку модели, для которой табулируется выбранный объект (в нашем случае – время «жизни» транзактов, например, перед удалением их из модели).

В данном пакете есть возможность табулировать время пребывания в очереди с помощью одной команды ***QTABLE***. Ее формат:

<имя> QTABLE A, B, C, D

Здесь в операнде A указывается имя очереди, назначение остальных операндов такое же, как и в общем случае. Например:

```
                ⋮
MET     QTABLE  ALFA , 8, 10, 5
                ⋮
        QUEUE   ALFA
                ⋮
        DEPART  ALFA
                ⋮
```

В таблицу с именем MET будет автоматически занесена информация об очереди ALFA.

5.4. Примеры

Пример 5.1. В одном обрабатывающем центре обрабатываются детали трех типов. Детали типа 1 поступают с интервалами от 5 до 10 мин, типа 2 – от 8 до 12 мин, типа 3 – от 10 до 14 мин. Обработка детали типа 1 занимает 7,5 мин, типа 2 – 10 мин, типа 3 – 12 мин. Создать модель работы обрабатывающего центра в течение 8 часов.

Программа имеет вид:

```
GENERATE    7.5, 2.5
ASSIGN      1, 1
TRANSFER    , OBR
GENERATE    10, 2
ASSIGN      1, 2
TRANSFER    , OBR
GENERATE    12, 2
ASSIGN      1, 3
```

```
OBR       QUEUE      QOBR
          SEIZE      OBRC
          DEPART     QOBR
          TEST  E    P1, 1, TIP2
          ADVANCE    7.5
          TRANSFER   , KONEC
TIP2      TEST  E    P1, 2, TIP3
          ADVANCE    10
          TRANSFER   , KONEC
TIP3      ADVANCE    12
KONEC     RELEASE    OBRC
          TERMINATE
          GENERATE   480
          TERMINATE  1
          START      1
```

В этой программе благодаря применению блока ASSIGN для определения типа деталей через значение первого параметра транзактов удалось осуществить их соответствующую задержку на обработку в одном обрабатывающем центре.

Пример 5.2. На участок контроля поступает поток изделий с интервалами от 5 до 7 мин. На участке три идентичных стенда. Изделие занимает свободный стенд, если такого нет, то направляется к стенду с наименьшей очередью. Время контроля подчиняется нормальному распределению с матожиданием 18 мин и стандартным отклонением 3 мин. После контроля изделия уходят с участка контроля для проведения других операций. Промоделировать работу участка при программе контроля 5000 изделий. С целью улучшения организации работы участка протабулировать информацию о времени нахождения изделий в очередях к стендам.

Программа модели:

```
OCH1      QTABLE     1,10,10,8
OCH2      QTABLE     2,10,10,8
OCH3      QTABLE     3,10,10,8
          GENERATE   6,1
          SELECT E   1,1,3,0,F,ALFA
BETTA     QUEUE      P1
          SEIZE      P1
          DEPART     P1
          ADVANCE    (NORMAL(2,18,3))
          RELEASE    P1
          TERMINATE  1
ALFA      SELECT MIN 1,1,3,,Q
          TRANSFER   ,BETTA
          START      5000
```

Таблицы по результатам моделирования будут иметь вид:

TABLE	MEAN	STD.DEV.	RANGE	RETRY	FREQUENCY	CUM.%
OCH1	43.410	16.957		0		
	–	–	10.000	17	1.02	
	10.000	–	20.000	161	10.69	
	20.000	–	30.000	198	22.58	
	30.000	–	40.000	344	43.24	
	40.000	–	50.000	313	62.04	
	50.000	–	60.000	362	83.78	
	60.000	–	70.000	165	93.69	
	70.000	–	—	105	100.00	
OCH2	37.139	16.827		0		
	–	–	10.000	77	4.59	
	10.000	–	20.000	248	19.39	
	20.000	–	30.000	256	34.67	
	30.000	–	40.000	352	55.67	
	40.000	–	50.000	329	75.30	
	50.000	–	60.000	278	91.89	
	60.000	–	70.000	97	97.67	
	70.000	–	—	39	100.00	
OCH3	31.023	16.275		0		
	–	–	10.000	154	9.27	
	10.000	–	20.000	330	29.14	
	20.000	–	30.000	261	44.85	
	30.000	–	40.000	428	70.62	
	40.000	–	50.000	276	87.24	
	50.000	–	60.000	155	96.57	
	60.000	–	70.000	45	99.28	
	70.000	–	–	12	100.00	

Анализ этих результатов показывает, что величины среднего значения и стандартного отклонения уменьшаются по мере возрастания номера стенда.

6. МОДЕЛИРОВАНИЕ СИСТЕМ С УСТРОЙСТВАМИ В РЕЖИМАХ ПРЕРЫВАНИЯ И НЕДОСТУПНОСТИ

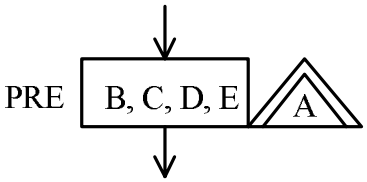
В этом разделе рассматриваются операторы, моделирующие ситуации, когда прерывается обслуживание транзактов в устройствах, а также средства, позволяющие сэкономить машинное время при поиске заблокированных транзактов.

6.1. Моделирование захвата устройств

Для моделирования захвата и освобождения устройств используются блоки ***PREEMPT*** (захватить) и ***RETURN*** (возвратить).

Формат и графическое изображение блока PREEMPT:

PREEMPT A, [B], [C], [D], [E]



В операнде A указывается имя устройства, подлежащего захвату.

Блок PREEMPT может работать или в приоритетном режиме, или в режиме прерывания. В первом случае в операнде B указывается PR, во втором этот операнд задается по умолчанию.

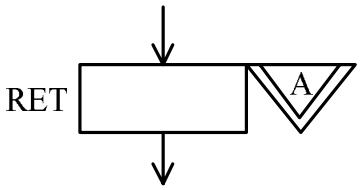
В операнде C указывается метка, куда направляется прерванный транзакт.

В операнде D указывается номер параметра прерванного транзакта, в котором записывается оставшееся время обслуживания.

Операнд E определяет право на дообслуживание: по умолчанию – сохраняется право, при указании RE – не сохраняется.

Блок RETURN фиксирует факт освобождения устройства от захвата. Его формат и изображение:

RETURN A



В операнде A указывается имя освобожденного устройства.

В модели устройство может быть захвачено любое количество раз различными транзактами, но не два раза подряд одним транзактом. Транзакт не может войти в блок, если в приоритетном режиме устройство захвачено транзактом с равным или более высоким приоритетом, чем активный транзакт.

Рассмотрим примеры использования блока PREEMPT:

```
PREEMPT         OTO
PREEMPT         OTO, PR
PREEMPT         OTO, PR, MET, , RE
```

В первом случае блок работает в режиме прерывания. Прерванный транзакт никуда не направляется, после обслуживания захватчика будет дообслужен.

В последнем случае прерванный транзакт теряет право на дообслуживание, поэтому по метке MET его можно направить, например, в блок TERMINATE, т.е. на удаление из модели. Этого нельзя было сделать, если в последнем случае операнд E задать по умолчанию. Более подробно различные варианты моделирования захвата изложены в [1].

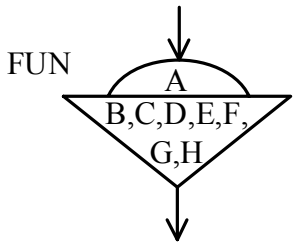
6.2. Моделирование недоступности устройств

При моделировании неисправностей устройств и реализации различных дисциплин обслуживания могут быть использованы блоки FUNAVAIL и FAVAIL.

Блок ***FUNAVAIL*** (F обозначает устройство, UNAVAIL – недоступно) моделирует недоступность одноканального устройства.

Формат и изображение блока:

FUNAVAIL A, [B], [C], [D], [E], [F], [G], [H]



В операнде A указывается имя устройства, которое становится недоступным.

Назначение остальных операндов зависит от характера транзактов, занимавших устройство до перевода его в недоступное состояние.

Операнды B, C, D соответствуют транзактам, занимавшим устройство после входа в него через SEIZE и PREEMPT .

Операнды E, F соответствуют прерванным транзактам.

Операнды G, H соответствуют транзактам, находящимся в списке задержки.

В операнде B задаются режимы работы с транзактами в период недоступности:

- CO (continue – продолжение) – обслуживание продолжается;
- RE (remove – удаление) – транзакт удаляется по адресу, указанному в операнде C;
- по умолчанию – обработка прерывается, будет продолжена, когда устройство станет доступным.

В операнде D задается номер параметра транзакта, занимавшего устройство, в который записывается оставшееся время обслуживания после того, как устройство стало недоступным.

В операнде E задаются режимы работы с ранее прерванными транзактами:

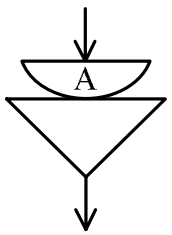
- CO – продолжение обслуживания;
- RE – удаление по адресу, указанному в операнде F;
- по умолчанию – прерванные ранее транзакты остаются в списке прерываний и не обслуживаются в устройстве в период его недоступности.

Операнд G определяет режимы работы с транзактами, находящимися в списке задержки в период недоступности устройства:

- CO – продолжение обслуживания;
- RE – удаление по адресу, указанному в операнде H;
- по умолчанию – оставление транзактов в списке задержки до момента, когда устройство станет доступным.

Блок ***FAVAIL*** (устройство доступно) моделирует доступность устройства. Формат и изображение блока:

FAVAIL A



В операнде A указывается имя устройства, которое становится доступным.

Пусть, например, модуль моделирования аварийной ситуации имеет вид:

```
                GENERATE        (NORMAL  (1, 100, 10))
                FUNAVAIL        OTO, CO , , , RE , MET1 , RE , MET1
                ADVANCE         5, 2
                FAVAIL          OTO
MET1            TERMINATE
```

При единице модельного времени, равной 1 часу, распределение интервалов наступления аварийных ситуаций подчиняется нормальному закону при среднем значении 100 часов и стандартном отклонении 10 часов.

При переводе устройства OTO в недоступное состояние продолжится обработка транзакта (если он находился в устройстве); транзакты, ранее прерванные, и находившиеся в списке задержки, будут удалены из модели (через MET1 TERMINATE). После восстановления устройства в течение (5 ± 2) часов оно становится доступным.

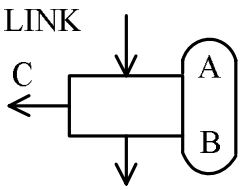
6.3. Применение списков пользователя

Списки пользователя (СП) организуются как с целью экономии машинного времени (из–за устранения потерь на просмотр СТС для поиска заблокированных транзактов), так и для реализации различных дисциплин обслуживания.

Ввод транзактов в список пользователя осуществляется блоком ***LINK*** (внести в список), вывод – блоком ***UNLINK*** (вывести из списка).

Формат и изображение блока LINK:

<имя> LINK A, B, [C]



В операнде A указывается символическое или цифровое имя списка пользователя.

Операнд B задает место списка, куда направляется транзакт.

Допустимые значения операнда B:

FIFO – в конец СП;

LIFO – в начало СП;

PR – в порядке убывания приоритета;

P – в порядке возрастания значения параметра;

M1 – в порядке возрастания относительного времени.

Операнд C указывает альтернативный выход. Если операнд C не задан (безусловный режим), то индикатор СП устанавливается в «1». Все транзакты заносятся в СП в порядке, определенном операндом B. Если операнд C задан, то проверяется индикатор СП. Если при этом индикатор в положении «1», то вошедший транзакт заносится в СП, в порядке определенном операндом B. Ели же индикатор окажется в положении «0», то он переводится в «1», а транзакт направляется по адресу, указанному в операнде C.

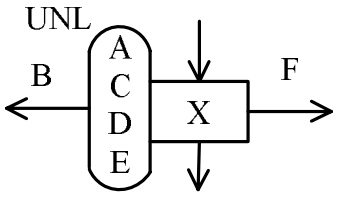
Например:

```
LINK        FAC, FIFO
LINK        DDD, M1, MET1
```

В первом случае транзакты присоединяются к СП по правилу FIFO. Во втором случае, если индикатор СП будет в положении «0», транзакт пойдет на метку MET1, а не в СП.

Рассмотрим блок UNILINK. Его формат и изображение:

<имя> UNLINK X A, B, C, [D], [E], [F]



В операнде A указывается символическое или цифровое имя СП.

В операнде B записывается метка, по которой направляются выведенные из СП транзакты.

В операнде C записывается число выводимых из СП транзактов или слово ALL (все), по умолчанию – ALL.

Операнды D и E вместе с операндом X задают порядок вывода транзактов из СП.

В дополнительном операнде X указываются операторы отношения (G, GE, L, LE, NE), по умолчанию E (равно). При этом сравнивается значение операнда D со значением операнда E.

В операнде D могут быть указаны булева переменная, номер параметра, слово BACK (обратно).

Если в D указана булева переменная, то операнды X и E пусты. Булева переменная вычисляется относительно транзакта из СП. Если BVj = 1, транзакт удаляется из СП, если BVj = 0 для всех транзактов СП, то вошедший транзакт пытается пройти по адресу, указанному в операнде F, если последний опущен, то в следующий оператор.

Если в D указан параметр, а операнд E отсутствует, то значение параметра вошедшего транзакта сравнивается со значением такого же параметра транзактов из СП, если E не пропущен, то со значением, указанным в операнде E. Удаляемые транзакты направляются по адресу, указанному в операнде B.

При использовании в операнде D ключевого слова BACK (операнд E отсутствует) транзакты выводятся из конца СП.

Операнд F указывает метку, по которой направляется транзакт, вошедший в блок UNLINK, если ни один транзакт не выводится из СП.

Например, блок UNLINK ABC, MET, ALL выводит все транзакты из списка пользователя с именем ABC по правилу LIFO и направляет их по адресу MET. Блок UNLINK BBB, ALFA, 1, BACK выводит из конца списка BBB один транзакт и направляет его по адресу ALFA.

Всего из–за сочетания значений операндов X, D и E может быть восемь вариантов вывода транзактов. Они рассмотрены в [3].

6.4. Примеры

Пример 6.1. На участке контроля и наладки изделий один стенд для одного изделия. Режим работы участка – трехсменный по 24 часа в сутки. Обычные (так называемые «фоновые») изделия поступают на участок по пуассоновскому закону со средним временем 48 часов, на их обслуживание уходит (3 ± 0,5) часа. «Плановые» изделия, необходимые для дальнейшего использования в производстве в данный плановый период, поступают по пуассоновскому закону со средним временем 24 часа, их обслуживание – экспоненциальное со средним временем 2 часа. Плановые изделия могут захватить стенд, после их обслуживания фоновые изделия дообслуживаются. Ежедневно через 8 часов после начала работы на участок поступают более приоритетные изделия и занимают стенд на 2 часа.

Составить модель работы участка в течение 25 суток.

Структура примера заимствована из источника [4] и адаптирована к условиям производства.

Особенности модели следующие. В модуле обычных изделий транзакты занимают и освобождают стенд с помощью блоков SEIZE STEND и RELEASE STEND; в модуле плановых изделий транзакты захватывают и освобождают стенд с помощью блоков PREEMPT STEND (режим прерывания) и RETURN

STEND; в модуле более приоритетных изделий захват и освобождение стенда происходят с использованием блока PREEMPT STEND, PR (режим приоритетный) и RETURN STEND.

При единице модельного времени в одну минуту, модельное время составляет 25 × 3 × 8 × 60 = 36000 единиц.

Программа модели имеет вид:

```
                GENERATE        (EXPONENTIAL (1, 0, 2880))
                SEIZE           STEND
                ADVANCE         180, 30
                RELEASE         STEND
                TERMINATE
                GENERATE        (EXPONENTIAL (2, 0, 1440))
                PREEMPT         STEND
                ADVANCE         (EXPONENTIAL (3, 0, 120))
                RETURN          STEND
                TERMINATE
                GENERATE        (EXPONENTIAL (4, 0, 1440)) , , 480, , 1
                PREEMPT         STEND, PR
                ADVANCE         120
                RETURN          STEND
                TERMINATE
                GENERATE        36000
                TERMINATE       1
                START           1
```

Результаты моделирования показывают, что за 25 недель прошло контроль и наладку 13 обычных, 25 плановых и 25 более приоритетных изделий (по числу входов в первый, второй и третий блоки TERMINATE).

Пример 6.2. Составить модель работы системы с реализацией дисциплины обслуживания FIFO, если транзакты генерируются с интервалами от 17,5 до 22,5 единиц, распределенными по равномерному закону, а задерживаются при обработке в одноканальном устройстве по экспоненциальному закону со средним значением 10 единиц.

Программа обработки 10000 транзактов с использованием блоков LINK и UNLINK при выводе из СП по одному транзакту имеет вид:

```
                GENERATE        20, 2.5
                LINK            ALFA, FIFO, MET1
MET1            SEIZE           ROBOT
                ADVANCE         (EXPONENTIAL (1, 0, 10))
                RELEASE         ROBOT
                UNLINK          ALFA, MET1, 1
                TERMINATE       1
                START           10000
```

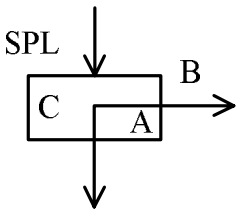
7. МОДЕЛИРОВАНИЕ СЛОЖНЫХ ПРОИЗВОДСТВЕННЫХ СИСТЕМ

При моделировании производственных систем (ГПС механообработки, штамповки, сборки и других дискретных технологических процессов) возникают задачи ветвления путей и синхронизации движения объектов, создания копий транзактов, их объединения и сбора, разрешения конфликтных ситуаций из-за ограниченности ресурсов основного и вспомогательного оборудования. В связи с этим рассмотрим операторы, предназначенные для моделирования ансамблей (семейств) транзактов, организации циклических процессов и логического управления.

7.1. Операторы создания копий транзактов и обеспечения синхронизации их движения

Наряду с блоком GENERATE для ввода в модель транзактов применяется блок ***SPLIT*** (расщепить). Этот блок порождает копии входящего в него транзакта. Формат и изображение оператора:

```
SPLIT      A, [B], [C]
```



В операнде A указывается количество копий (в виде константы или СЧА).

В операнде B указывается метка, по которой направляются копии транзактов. Порождающий транзакт проходит в следующий оператор. Если операнд B задан по умолчанию, то копии также проходят в следующий оператор.

В операнде C указывается номер параметра, в котором упорядочиваются номера копий транзакта. Если номер порождающего транзакта k, то после блока SPLIT его номер будет $k + 1$, а номера копий $k + 2$, $k + 3$ и т.д.

Различные номера копий могут быть использованы для задания адресов, по которым они могут пойти. Например:

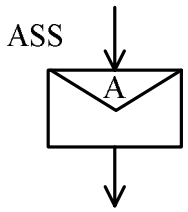
```
                    ⋮
ABC          FUNCTION   P5, D3
2, BLOK A / 3, BLOK B / 4, BLOK C
                    ⋮
             SPLIT      3, FN$ABC, 5
                    ⋮
```

Копии транзактов будут направлены по меткам (адресам) BLOK A или BLOK B или BLOK C.

Транзакты, порожденные с помощью блока SPLIT, образуют ансамбль. Для работы с транзактами, принадлежащими ансамблю, используются блоки ASSEMBLE, GATHER, MATCH.

Блок ***ASSEMBLE*** (соединить) применяется для объединения нескольких транзактов одного семейство в один и вывода его из модели. Формат и изображение блока:

ASSEMBLE A



В операнде A указывается количество объединяемых транзактов. Например:

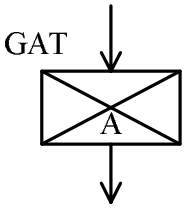
ASSEMBLE 10

объединяет 10 транзактов, при этом 9 уничтожаются, а 1 уходит в следующий оператор.

Блок ***GATHER*** (собрать) применяется для накопления транзактов до количества, указанного в операнде A.

Формат и изображение блока:

GATHER A

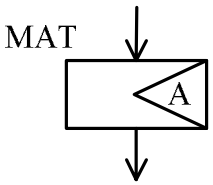


Когда количество вошедших в блок транзактов достигнет значения, указанного в операнде A, все они пойдут в следующий оператор.

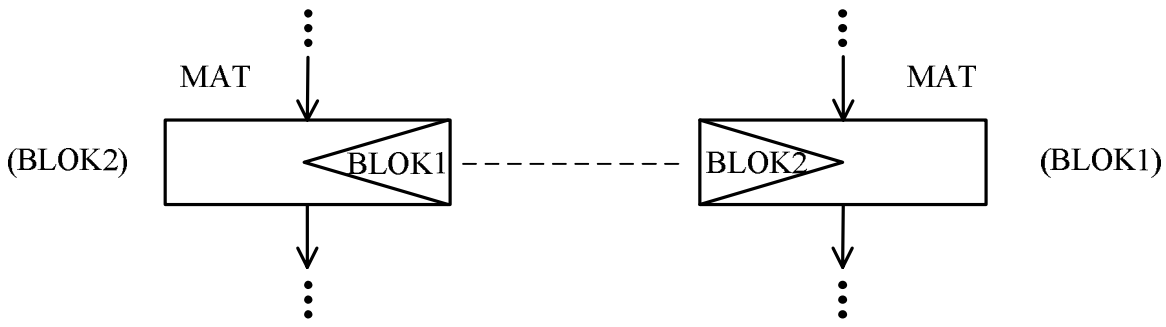
Блок ***MATCH*** (синхронизировать) осуществляет синхронизацию движения транзактов по двум направлениям.

Формат и изображение блока:

<метка> MATCH A



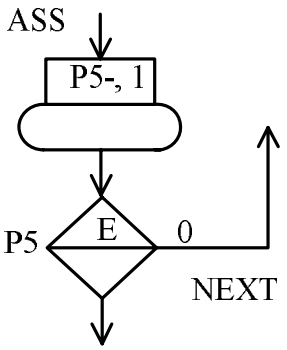
При этом используются два блока, как показано ниже:



Здесь в качестве операнда A левого блока используется метка правого (сопряженного) блока и наоборот.

7.2. Моделирование цикла

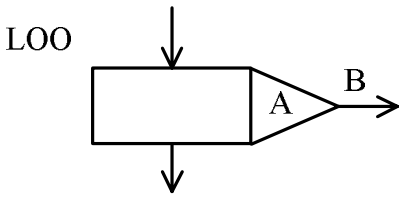
Для организации цикла можно применить рассмотренные ранее блоки ASSIGN и TEST с проверкой равенства нулю уменьшенного значения, как показано на следующем рисунке:



Эта схема соответствует ситуации, когда, например, детали в паллете убывают (транзакты идут по метке NEXT) до тех пор, пока не обнулится содержимое паллеты (тогда транзакт пойдет дальше по программе).

Для моделирования такого цикла можно использовать специальный блок ***LOOP*** (организовать цикл). Его формат и изображение:

LOOP A, B



В операнде A указывается параметр, содержимое которого после уменьшения на единицу сравнивается с нулем. В операнде B указывается метка начального в цикле оператора. Для рассмотренного фрагмента в этом случае будет запись:

LOOP P5, NEXT

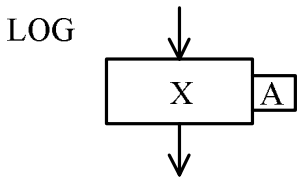
7.3. Моделирование логического управления

Для моделирования логического управления используются логические ключи, представляющие собой объекты, которые могут принимать одно из двух значений: включено (1), выключено (0).

Логический ключ моделируется блоком ***LOGIC*** (воздействовать на логический ключ).

Его формат и изображение:

LOGIC X A



Этот блок изменяет состояние логического ключа, имя которого указано в операнде A, в соответствии с вспомогательным операндом X. Операнд X мо-

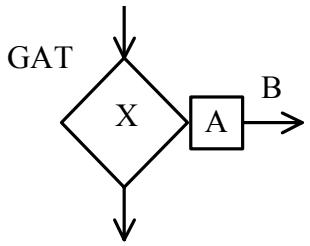
жет принимать значения: S – установить (включить); R – сбросить (выключить); I – инвертировать (изменить состояние на противоположное).

Например: LOGIC R 12 переводит логический ключ в состояние «выключено», а LOGIC I ALFA изменяет состояние ключа с именем ALFA на противоположное.

7.4. Проверка состояний логического ключа и других объектов модели

Для проверки состояния логического ключа используется блок ***GATE*** (впустить). Его формат и изображение:

GATE X A, [B]



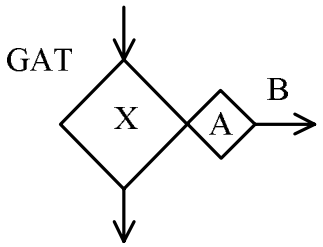
Этот блок может использоваться в режиме отказа (операнд B не задается), когда транзакт пройдет в следующий оператор лишь при успешном результате проверки ключа, имя которого указано в операнде A, или в режиме условной передачи (в операнде B указывается метка, по которой направляется транзакт при безуспешном результате проверки).

В дополнительном операнде X указываются символы:

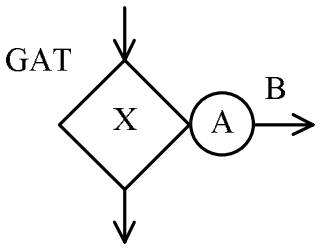
LS – равен 1, если ключ включен; 0 – если выключен;

LR – равен 1, если ключ выключен; 0 – если включен.

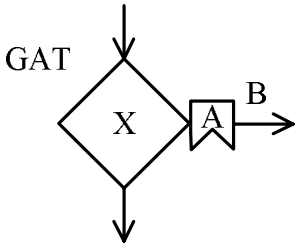
При использовании блока GATE для проверки состояния других объектов модели его формат и режимы такие же.



Для проверки состояния одноканальных устройств в операнде A указывается имя или номер устройства; в операнде X: U (занято), NU (не занято); I (захвачено), NI (не захвачено); FV (доступно), FNV (не доступно).



При проверке состояния многоканальных устройств в операнде A указывается имя или номер МКУ; в операнде X: SE (пусто), SNE (не пусто); SF (заполнено), SNF (не заполнено); SV (доступно), SNV (не доступно).



При проверке состояния синхронизации исследуемого транзакта в операнде A указывается метка блока MATCH; в операнде X: M (есть ли в блоке MATCH ожидающий синхронизации транзакт), NM (нет такого транзакта).

7.5. Примеры

Пример 7.1. Моделирование гибкой автоматической линии (ГАЛ) механообработки. Рассмотрим ГАЛ механообработки, состоящую из склада SKL, обрабатывающего модуля ОМО, транспортного робота TRM (рис. 7.1).

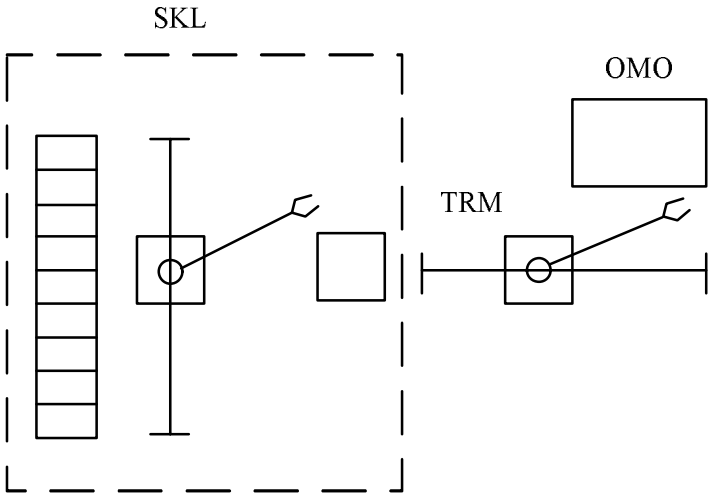


Рис. 7.1

Работа линии соответствует диаграмме (рис. 7.2), на которой приняты обозначения: ПСК и РСК – соответственно получение заготовок и размещение деталей на складе; ОбМ – обработка в модуле; ТРМ и ТРС – соответственно транспортировка к модулю и складу.

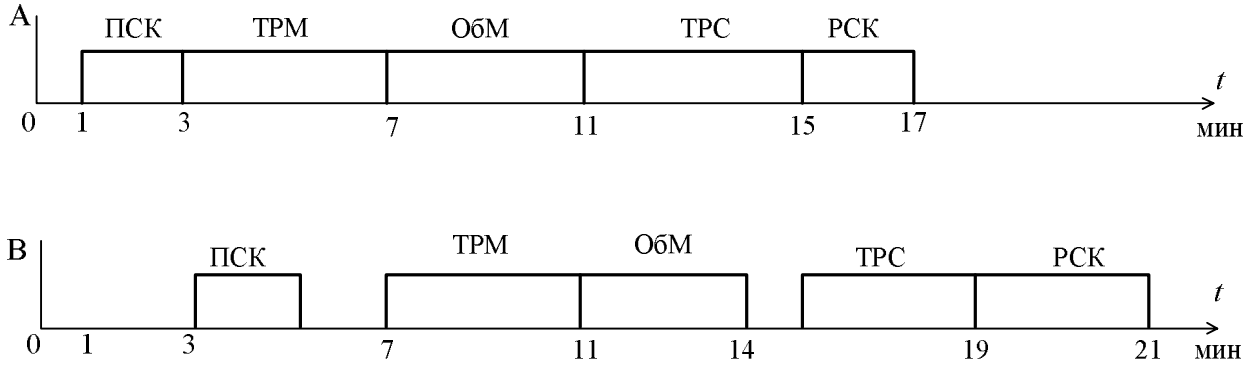


Рис. 7.2

Промоделируем работу линии в течение смены при коэффициенте использования оборудования 0,9. Учтем равномерный закон распределения продолжительности интервалов получения заготовок и размещения деталей на складе, пусть он будет (2 ± 1) мин (на диаграмме указано среднее значение – 2 мин).

Алгоритм моделирования гибкой линии приведен на рис. 7.3. Его особенность заключается в том, что для проверки незанятости транспортного робота используются блоки 5, 6, 12, а для проверки незанятости склада – блок 14. В случае невыполнения проверяемых условий блокируется дальнейшее продвижение по алгоритму.

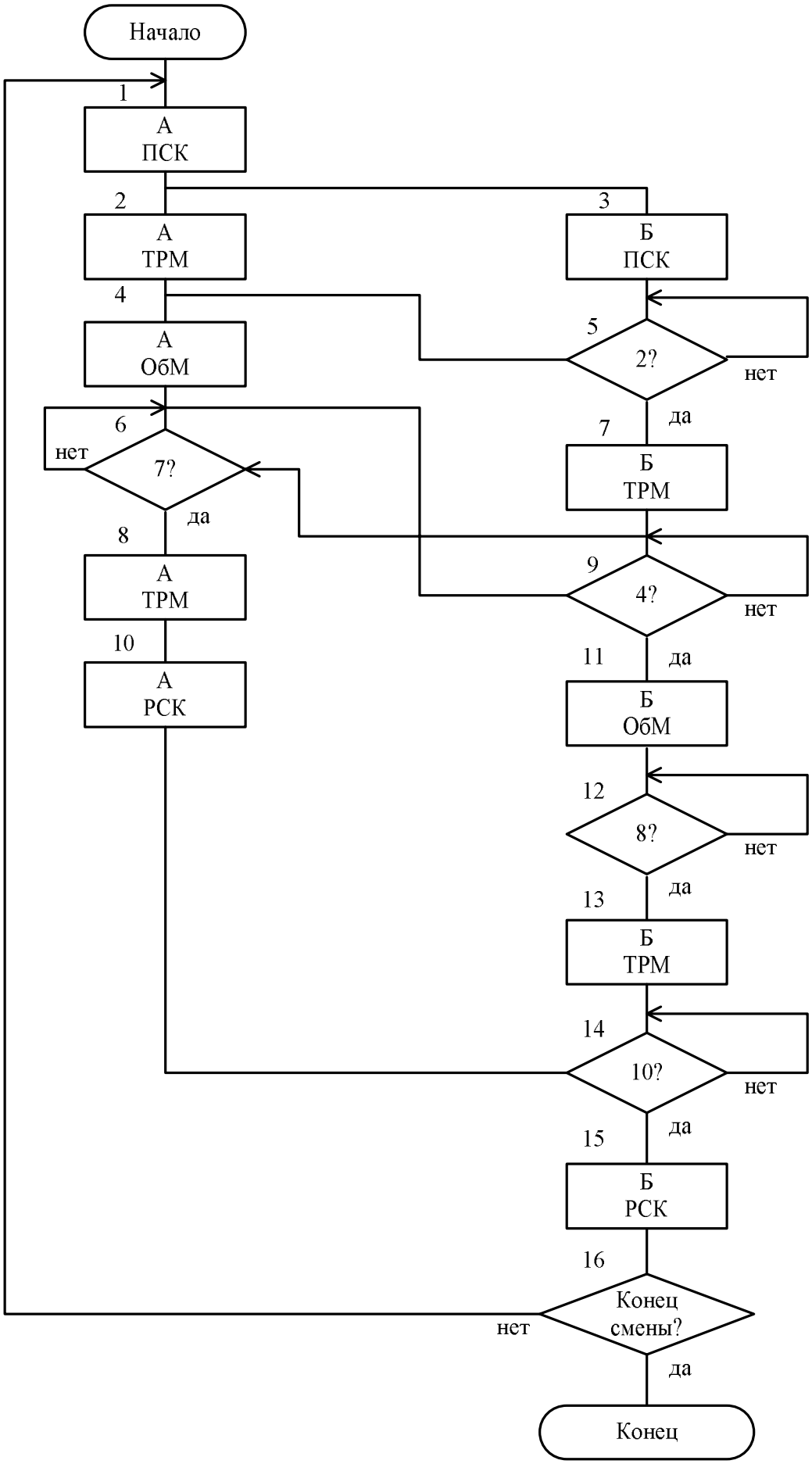

Начало
1
А
ПСК
2
А
ТРМ
3
Б
ПСК
4
А
ОбМ
5
2?
нет
да
6
нет
7?
да
7
Б
ТРМ
8
А
ТРМ
9
4?
нет
10
А
РСК
11
да
Б
ОбМ
12
8?
нет
да
13
Б
ТРМ
14
10?
нет
да
15
Б
РСК
16
Конец
смены?
нет
да
Конец

Рис. 7.3

При разработке программы используем для моделирования разветвлений оператор SPLIT, а для моделирования проверки отмеченных конфликтных ситуаций – операторы LOGIC и GATE. За единицу модельного времени примем 1 мин. Продолжительность времени моделирования будет 480 × 0,9 = 432 единицы. Для наглядного сопоставления программы и алгоритма в качестве цифровой части меток операторов программы используем номера блоков алгоритма.

Программа модели представлена ниже.

```
        GENERATE    , , , 1
MET1    SEIZE       SKL
        ADVANCE     2, 1
        RELEASE     SKL
        SPLIT       1, MET3
        SEIZE       TRM
        ADVANCE     4
        RELEASE     TRM
        SPLIT       1, MET5
        SEIZE       OMO
        ADVANCE     4
        RELEASE     OMO
        SPLIT       1, MET9
        TRANSFER    , MET6
MET3    SEIZE       SKL
        ADVANCE     2, 1
        RELEASE     SKL
MET5    LOGIC I     PER1
        GATE LS     PER1, MET7
        TERMINATE
MET6    LOGIC I     PER2
        GATE LS     PER2, MET8
        TERMINATE
MET7    SEIZE       TRM
        ADVANSE     4
        RELEASE     TRM
        SPLIT       1, MET6
        TRANSFER    , MET9
MET8    SEIZE       TRM
        ADVANCE     4
        RELEASE     TRM
        SPLIT       1, MET12
        SEIZE       SKL
        ADVANCE     2,1
        RELEASE     SKL
        TRANSFER    , MET14
MET9    LOGIC I     PER3
        GATE LS     PER3, MET11
        TERMINATE
MET11   SEIZE       OMO
        ADVANCE     3
        RELEASE     OMO
```

```
MET12     LOGIC I          PER4
          GATE LS          PER4, MET13
          TERMINATE
MET13     SEIZE            TRM
          ADVANCE          4
          RELEASE          TRM
MET14     LOGIC I          PER5
          GATE LS          PER5, MET15
          TERMINATE
MET15     SEIZE            SKL
          ADVANCE          2,1
          RELEASE          SKL
          TEST LE          M1, 432, KONEC
          TRANSFER         , MET1
KONEC     TERMINATE        1
          START            1
```

Результаты моделирования: за смену обработано 44 детали двух типов; при этом загрузка обрабатывающего модуля составила 34,6 %; загрузка склада – 40,2 %; загрузка транспортного робота – 79,1 %.

Для повышения производительности ГАЛ целесообразно использовать более быстродействующий транспортный робот. Как показало моделирование, целесообразно применение в этой линии транспортного робота со временем перемещения 1 мин. В этом случае возрастает загрузка основного оборудования и количество обработанных деталей (табл. 7.1).

Таблица 7.1

Зависимость показателей ГАЛ механообработки от времени перемещения транспортного робота

Время перемещения TRM, мин	**Коэффициент загрузки, %**			**Количество обработанных деталей, шт.**
	OMO	**SKL**	**TRM**	
4	34,6	40,2	79,1	44
3	40,4	49,8	69,2	50
2	46,6	55,0	52,9	58
1	53,5	62,2	30,6	68

Пример 7.2. Моделирование ГАУ штамповки. Рассмотрим ГАУ штамповки деталей из штучных заготовок. Участок содержит пресс PRESS, четырехпозиционное поворотное загрузочное устройство ZNU, приемное устройство PRU, промежуточный приемный стол PRS (рис. 7.4).

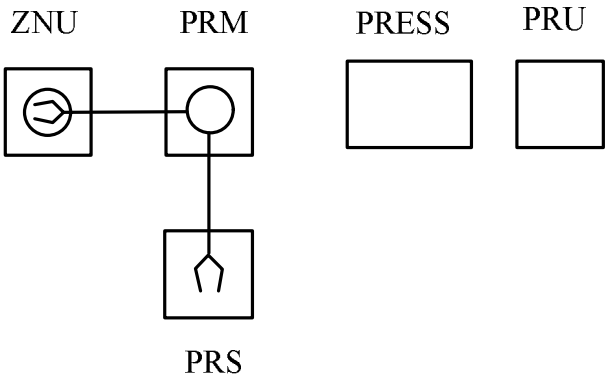


Рис. 7.4

Движение заготовок, полуфабрикатов и деталей осуществляется слева направо. Продолжительность цикла работы манипулятора (опустить руку, взять заготовку, поднять руку, повернуться на 90°, опустить руку, положить заготовку, поднять руку, возвратиться в исходное положение) составляет 3,7 с. Кассета вмещает 300 заготовок, тара под отштампованные детали – 2100 деталей. Продолжительность поворота загрузочного устройства на 90° – 10 с, перегрузка приемного устройства после его заполнения – 180 с, рабочего цикла прессования детали – 60/63 с.

Промоделируем работу участка в течение двух смен при коэффициенте использования рабочего времени равном 0,8. Оценим производительность участка и загрузку оборудования.

При составлении алгоритма моделирования введем следующие обозначения: K – количество деталей, n – счетчик тары под готовые детали (так как емкость тары под детали – 2100, а кассеты – 300 деталей, то n = 1, 2,..., 7). С учетом коэффициента использования участка и сменности его работы модельное время будет $0{,}8 \cdot 2 \cdot 28\,800 = 46\,080$ с.

Алгоритм моделирования участка представлен на рис. 7.5.

Принимаем за единицу модельного времени 0,01 с. При разработке программы для организации цикла используем оператор TRANSFER в режиме безусловной передачи. При моделировании счетчиков используются операторы SAVEVALUE в режимах замещения и приращения, а также оператор TEST.

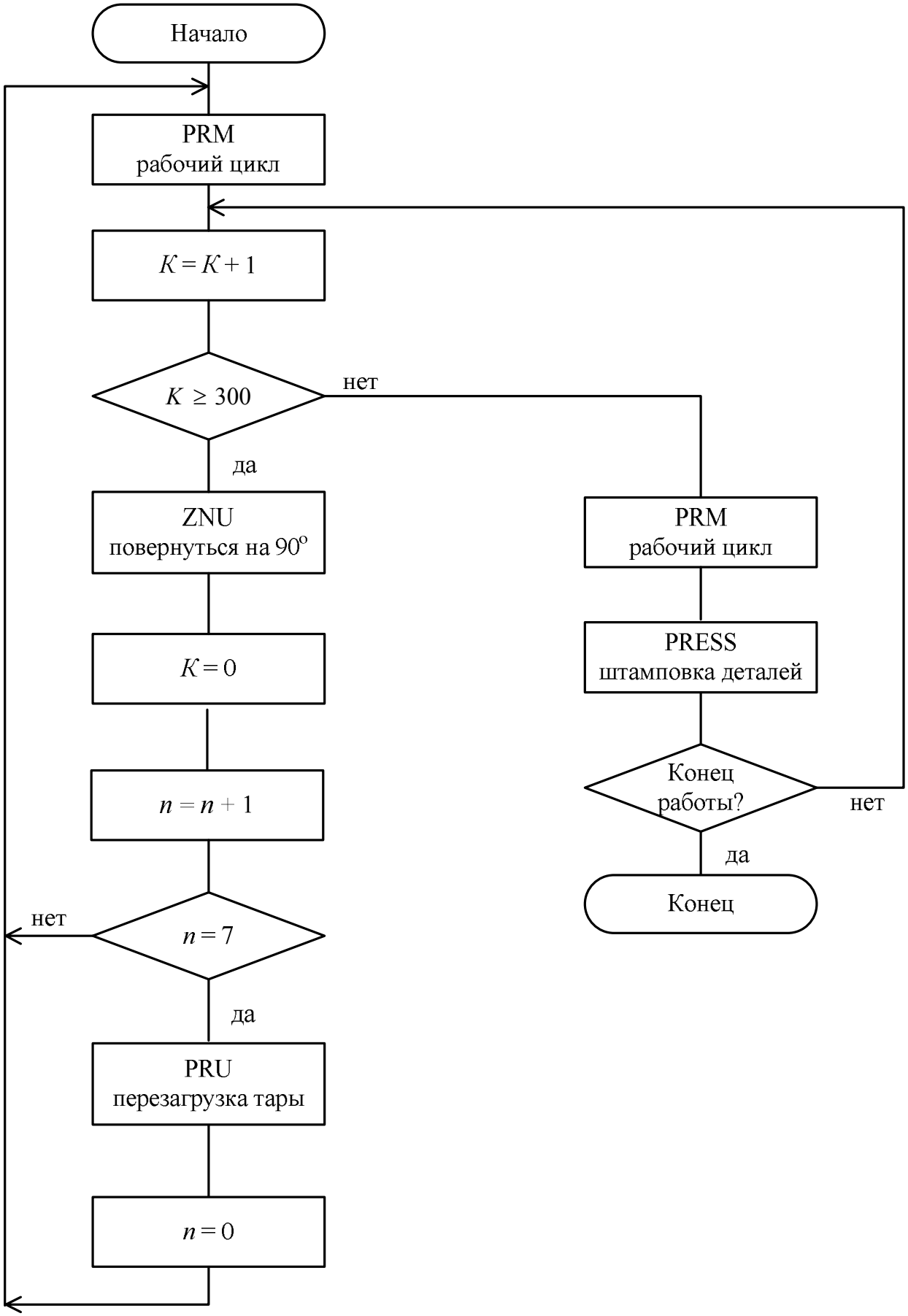

Начало
PRM
рабочий цикл
K = K + 1
K ≥ 300
нет
да
ZNU
повернуться на 90°
K = 0
n = n + 1
n = 7
нет
да
PRU
перезагрузка тары
n = 0
PRM
рабочий цикл
PRESS
штамповка деталей
Конец
работы?
нет
да
Конец

Рис. 7.5

Программа имеет вид:

```
VRA         VARIABLE        6000/63
            GENERATE        , , , 1
            ASSIGN          1, 4608000
MET1        SEIZE           PRM
            ADVANCE         370
            RELEASE         PRM
MET2        SAVEVALUE       1+, 1
            TEST  GE        X1, 300, MET 3
            SEIZE           ZNU
            ADVANCE         1000
            RELEASE         ZNU
            SAVEVALUE       1, 0
            SAVEVALUE       2+, 1
            TEST E          X2, 7, MET 1
            SEIZE           PRU
            ADVANCE         18000
            RELEASE         PRU
            SAVEVALUE       2, 0
            TRANSFER        , MET 1
MET3        SEIZE           PRM
            ADVANCE         370
            RELEASE         PRM
            SEIZE           PRESS
            ADVANCE         V$VRA
            RELEASE         PRESS
            TEST LE         M1, P1, MET 4
            TRANSFER        , MET 2
MET4        TERMINATE       1
            START           1
```

В результате моделирования установлено, что за время 46080 с отштамповано 9765 деталей; коэффициенты загрузки оборудования составили: пресса 19 %, манипулятора 78,6 %.

ЗАКЛЮЧЕНИЕ

В данном конспекте лекций рассмотрены основы построения языка имитационного моделирования GPSS, изложены инструментальные средства пакета GPSS World применительно к проектированию дискретных промышленных систем.

Поскольку язык GPSS предназначен для моделирования систем массового обслуживания, в первом разделе конспекта приведены краткие сведения о моделях массового обслуживания, необходимые для понимания последующего материала.

Второй раздел посвящен изложению концептуальной сущности языка GPSS и представлению на нем моделей.

Наиболее объемным является третий раздел, в котором раскрывается базовая часть языка GPSS, дающая представления о внутренней логике работы пакета, основных операторах языка, приемах разработки программ при создании моделей систем с одноканальными и многоканальными устройствами.

Четвертый раздел посвящен изложению особенностей применения объектов вычислительной категории.

В пятом разделе рассмотрены наиболее часто встречающиеся средства, применяемые для упрощения моделей при их большой размерности, сложных дисциплинах обслуживания, необходимости рационального представления обширной информации о результатах моделирования.

Последние разделы посвящены изложению материала, связанного с возможностями пакета при моделировании реальных промышленных систем различного назначения как в обычных режимах, так и в нештатных ситуациях.

Рассмотренные примеры иллюстрируют возможности языка при проектировании сложных систем дискретного производства. Особую значимость этот подход приобретает при различных законах поступления, обработки транзактов и реализации дисциплин обслуживания. В таких случаях имитационное моделирование является наиболее приемлемым методом определения характеристик систем.

ЛИТЕРАТУРА

1. Боев В. Д. Моделирование систем. Инструментальные средства GPSS World: Учеб. пособие. – СПб.: БХВ–Петербург, 2004. – 368 с.

2. Кудрявцев Е. М. GPSS World. Основы имитационного моделирования различных систем. – М.: ДМК Пресс, 2004. – 320 с.

3. Томашевский В. И., Жданова Е. Г. Имитационное моделирование в среде GPSS. – М.: Бестселлер, 2003. – 416 с.

4. Шрайбер Т. Дж. Моделирование на GPSS.–М., 1980.–592 с.

5. Лукьянец С. В., Пашкевич А. П. Моделирование гибких производственных систем и роботизированных комплексов. – Мн.: БГУИР, 2005. – 232 с.