Ресурсы, конфликты на ресурсах

Процессы Zi в системе Q развиваются параллельно. Это значит, что они изменяют значения параметров системы в течение одного и того же интервала времени. Достаточно типичны ситуации, когда по логике функционирования системы накладываются ограничения на изменение некоторых параметров несколькими процессами одновременно в течение заданного либо обусловленного интервала времени. Это возможно лишь для пересекающихся объектов. Ограничения развития процесса, накладываемые другими процессами, назовем конфликтными ситуациями, или конфликтами.

Общую область параметров для пересекающихся процессов Ok и Om назовём ресурсом. Таким образом, ресурс $R_{k,m}$ определяется, как

$$R_{k,m} = O_k \cap O_m$$

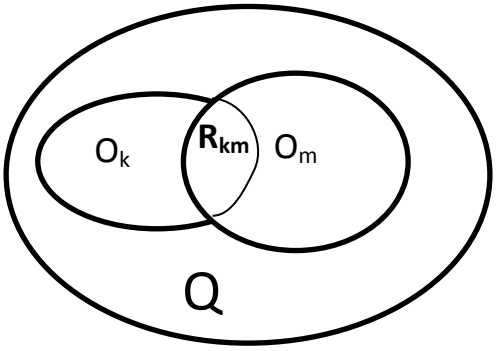


Рисунок 1. Определение ресурса

Конфликт возможен только при наличии ресурса. Таким образом, необходимо добиться согласования процессов в этой области. В основе способов разрешения конфликтов лежит утверждение об обязательном разделении доступа процессов к ресурсу во времени. Рассмотрим следующие способы разрешения конфликтных ситуаций.

А. Явное разделение по времени (синхронизация). При этом способе разрешения конфликта разнесение во времени производится явным указанием интервалов времени, определенных для каждого процесса. На рисунке 2 показан пример выделения таких интервалов для случая конфликта трех процессов.

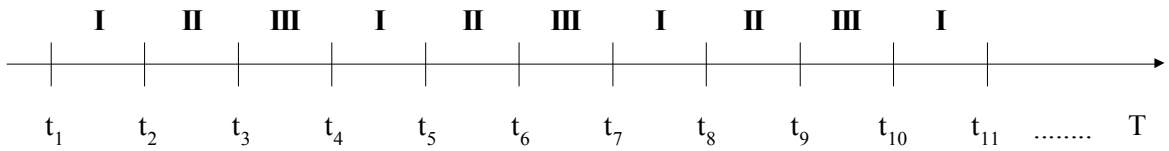


Рисунок 2. Синхронизация процессов

Допустим, что есть сигналы, поступающие на ресурс, который может работать лишь с одним сигналом. Решим задачу с помощью метода синхронизации.

Синхросерии строятся из некоторой базовой серии. На примере, приведенном на рисунке 3, показано расщепление базовой серии на три частных серии. И каждая серия подсоединена к своему компаратору. Кроме этого, к каждому компаратору подходят свои сигналы A, B, C.

В качества ресурса выступает блок R.

Из схемы, показанной на рисунке 2, видно, что на ресурс R попадает в любой момент времени только один сигнал.

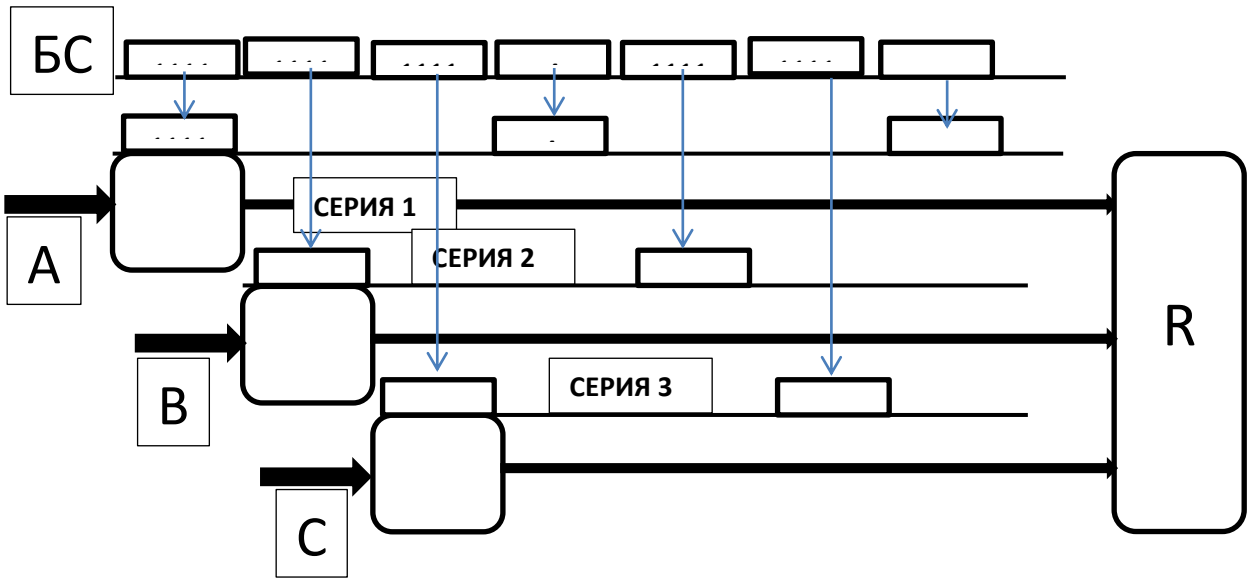


Рисунок 3. Разрешение конфликта доступа методом синхронизации

Б. Логический способ. Если условие захвата ресурса не ограничивает время использования этого ресурса захватившим его процессом, то в этом случае удобно использовать семафоры. Семафор относится к логическим способам разрешения конфликтов. Семафор есть простая логическая переменная, однозначно соответствующая ресурсу. Значение семафора ‘0’ означает, что ресурс может быть захвачен процессом, значение семафора ‘1’ блокирует захват ресурса. На рисунке 4 показан пример использования семафора C при захвате ресурса R двумя процессами Z1 и Z2.

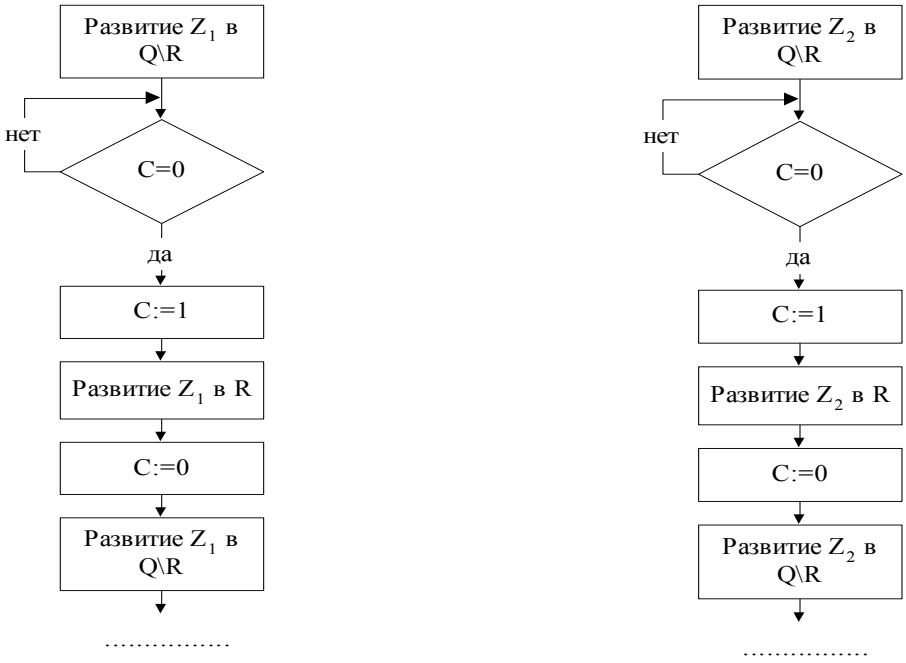


Рисунок 4. Применение семафора

Следует заметить, что в качества семафора может выступать любая логическая функция.

Примеры процессов с наличием ресурсов

Постановка. Рассмотрим перекрёсток двух автомобильных трасс (рисунок 5):

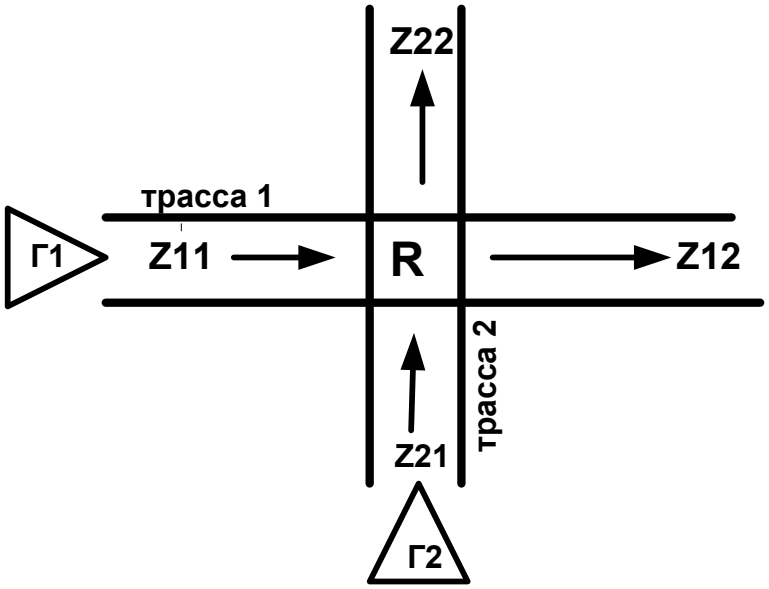


Рисунок 5. Схема перекрестка

Перекресток R есть ресурс, т.к. $Z1 \cap Z2 = R$ (R является общим параметром обоих процессов);

Г1 и Г2 генераторы; генерируют инициаторы в свои процессоры;

трасса1 и трасса2, процессоры реализующие процессы Z1 и Z2 соответственно.

Реализация семафором. Этот вариант захвата ресурса годится для случая слабой загрузки трасс, когда на трассах нет очередей. Поэтому мы можем использовать принцип «кто первый захватил».

Описание процессоров

```
блок процессор  ТРАССА 1;
описания;
r – скаляр;  // отражает состояние ресурса R
ТПЕР – скаляр;  // время переезда через перекресток
TKR – скаляр;  // момент времени покидания перекрестка
…….
все  описания

алгоритм
ВХОД :  ждать  r =0;
        r := 1;
        TKR := ВРЕМЯ+ТПЕР;
        ждать ВРЕМЯ = TKR;
        r := 0;
        направить инициатор на ,,,,,(продолжение процесса  Z1)
все алгоритм;
все блок;


блок процессор  ТРАССА 2;
описания;
r – скаляр;  // отражает состояние ресурса R
ТПЕР – скаляр;  // время переезда через перекресток
TKR – скаляр; // момент времени покидания перекрестка
…….
все  описания
алгоритм
ВХОД :  ждать  r =0;
        r := 1;
        TKR := ВРЕМЯ+ТПЕР;
        ждать ВРЕМЯ = TKR;
        r := 0;
        направить инициатор на ,,,,,(продолжение процесса  Z2)
все алгоритм;
все блок;
```

Алгоритмический способ

Наиболее общий способ управления процессами при захвате ресурсов состоит в использовании блоков - агрегатов, в том числе и контроллеров. Пример использования этих блоков для разрешения конфликтов приведено ниже.

В случае интенсивной загрузки трасс и появления очередей предыдущий вариант не пройдет, он приведет к авариям. Поэтому рассмотрим вариант алгоритмического управления доступа к ресурсу.

Реализация логического управления агрегатом (управление через параметры, рисунок 6)

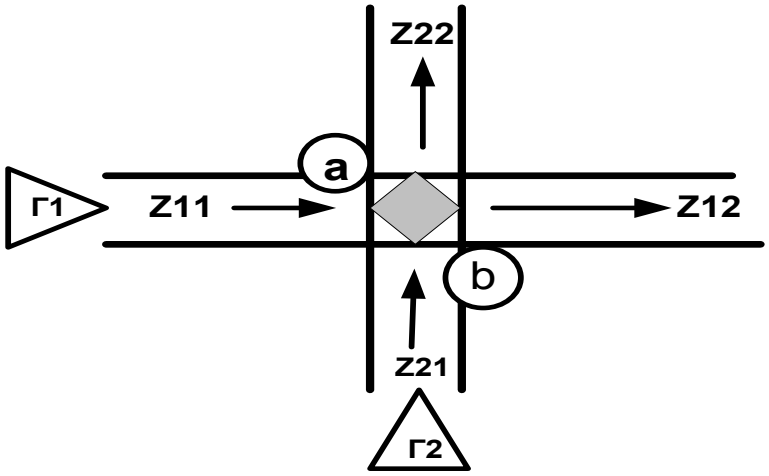


Рисунок 6. Алгоритмический вариант

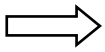
Управление осуществляется введением параметров a и b, которые отражают состояние ресурса – перекресток.

Описание процессоров

блок процессор ТРАССА1;
описания;
a**– скаляр;** // внешний параметр**,** отражает состояние ресурса R
ТПЕР – **скаляр**; // время переезда через перекресток
TKR – **скаляр**; // момент времени покидания перекрестка
…….
все описания
алгоритм
ВХОД : **ждать** a = «зеленый»;
TKR := ВРЕМЯ+ТПЕР;
ждать ВРЕМЯ = TKR;
направить инициатор на ,,,,,(продолжение процесса Z1)
все алгоритм;
все блок;

блок процессор ТРАССА2;
описания;
b **– скаляр;** // внешний параметр, отражает состояние ресурса R
ТПЕР – **скаляр**; // время переезда через перекресток
TKR – **скаляр**; // момент времени покидания перекрестка
…….
все описания

алгоритм



ВХОД : **ждать** b = «зеленый»;
TKR := ВРЕМЯ+ТПЕР;
ждать ВРЕМЯ = TKR;
направить инициатор на ,,,,,(продолжение процесса Z2)
все алгоритм;
все блок;

блок агрегат СВЕТОФОР;
описания
a,b – **скаляры**; // параметры состояния агрегата
ТПЕР – **скаляр**; // момент вреения переезда переключения светофора
T1, T2 – **скаляры**; // длительность состояний сваетофора
…….
все описания
алгоритм
НАЧ : a := «зеленый»;
b := «красный»;
ТПЕР := ВРЕМЯ+T1;
ждать ВРЕМЯ = ТПЕР;
a := «красный»;
b := «зеленый»;
ТПЕР := ВРЕМЯ+T2;
ждать ВРЕМЯ = ТПЕР;
направить инициатор на НАЧ;
все алгоритм;
все блок;

Схемы описаний функционирования систем

Агрегативная схема. Агрегативная схема описания функционирования системы предполагает использование только А-блоков (рисунок 7). Как показано выше, в такой схеме взаимодействие может осуществляться лишь через параметры. В результате формируется информационная сеть агрегатов: агрегаты - вершины сети, дуги - информационные связи. Для агрегативных схем показано, что каждый А-блок в такой модели может представляться некоторым конечным автоматом. Если функционирование системы может быть описано совокупностью конечных автоматов, взаимодействующих между собой через множество входных и выходных параметров, то применение агрегативных схем представляется наиболее рациональным.

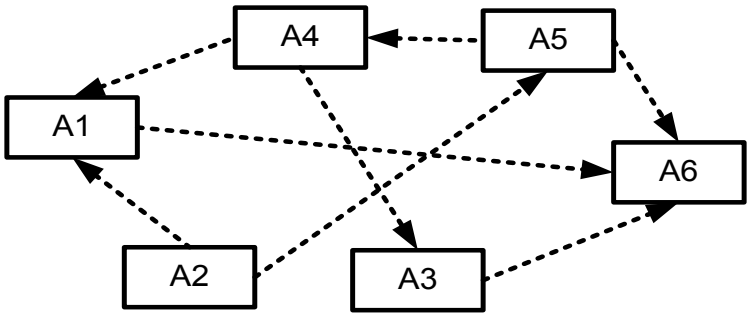


Рисунок 7. Пример агрегативной схемы

Процессная схема. Если в основе описания функционирования системы лежит использование П-блоков, то такое описание назовем процессным. Взаимодействие в процессных схемах осуществляется как через параметры, так и через локальные среды процессов. Процессный подход наиболее эффективен, когда имеем дело с множеством явно выраженных локальных процессов, например, при описании информационных, биологических, экономических, социальных и т.п. систем. Процессный подход реализован в языках Simula, GPSS и др. Пример процессной схемы приведен на рисунке 8.

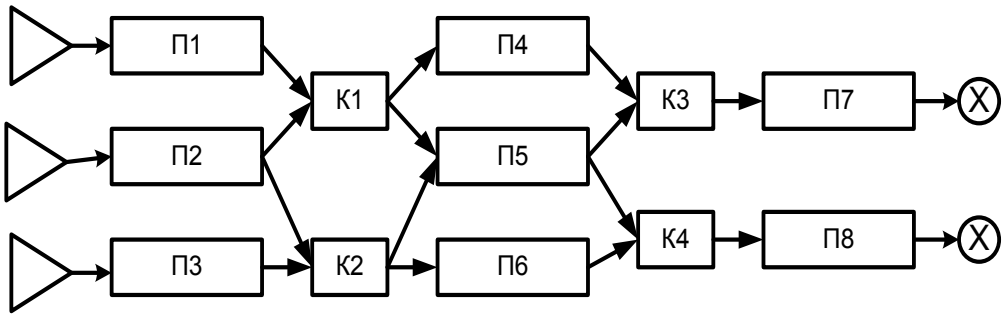


Рисунок 8. Пример процессной схемы

Потоковая схема. Потоковая схема возникает из агрегативной, когда среди связей агрегатов появляются связи с управляющей информацией. Когда эта информация поступает в агрегат, она производит инициацию процессов внутри агрегата. Если на графе информационных связей агрегативной схемы выделить лишь связи с управляющей информацией, то получим потоковую сеть агрегатов. Управляющая информация в этом случае может рассматриваться, как *инициаторы сетевого уровня*.

Таким образом, в потоковых схемах существуют два вида инициаторов: сетевые инициаторы, инициирующие запуск агрегата, и блочные инициаторы, реализующие процесс функционирования блока. Пример потоковой схемы приведен на рисунке 9.

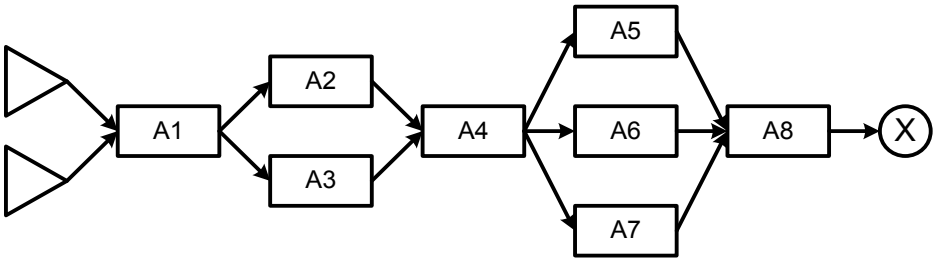


Рисунок 9. Пример потоковой схемы

На рисунке стрелками показаны пути движения сетевых инициаторов. Блочные (внутренние инициаторы) находятся внутри агрегатов. Например, сетевой инициатор, поступающий на агрегат А4 от агрегата А2, вызывает выполнение некоторого процесса в А4. Сам инициатор после этого может быть уничтожен или перемещен блоком А4 дальше по сети.

К потоковым схемам относятся сети Наура, Е-сети, сети Петри. Наибольшее распространение среди потоковых схем получили сети массового обслуживания (СеМО). Эти сети возникают из процессных схем, когда процессоры очень просты, либо отсутствуют вообще. Тогда основным элементом сети становятся контроллеры. Объединение контроллера с простым процессором или агрегатом формируют объект, называемый *системой массового обслуживания* (СМО). Таким образом, возникает сеть из СМО. Согласно определению потоковой схемы, по этой сети перемещаются инициаторы сетевого уровня, которые в теории СеМО называются сообщениями или требованиями.

Агрегативно-процессные схемы. Наиболее распространен подход при описании функционирования сложной системы с применением всех видов блоков - А, П и К. Примерами могут быть системы моделирования GPSS, SOL, СЛЕНГ, ДИС и другие. В каждой из них существуют свои ограничения на алгоритмы этих блоков. Так, в языке GPSS существуют операторы-агрегаты (GENERATE, TERMINATE), операторы

навигационного типа (TRANSFER), объекты-контроллеры (STORAGE). Большинство остальных операторов имеют тип процессоров. В языке GPSS введены инициаторы в явном виде (TRANSACT), доступные пользователю и способные создавать локальные среды (параметры транзактов). Однако, в языке GPSS пользователь лишен возможности задавать алгоритмы блоков и использует лишь библиотечные конструкции.

Сети массового обслуживания. Сети массового обслуживания являются особым видом потоковых схем, когда все процессы пересечены на ресурсах. Разрешение конфликтных ситуаций в этих схемах выполняется с помощью контроллеров, которые называются системами массового обслуживания (СМО). Система массового обслуживания есть объединение К-блока и одного или нескольких А-блоков: К-блок реализует дисциплину обслуживания требований, а А-блоки - процесс изменения параметров ресурса.

При исследовании систем массового обслуживания аналитическими методами выбираются достаточно простые алгоритмы К-блоков, а алгоритмы А-блоков сводятся, как правило, к задержке инициатора на некоторое время в ресурсе. Такие системы могут содержать единственный К-блок, совмещающий вышеуказанные действия.

Таким образом, формализация функционирования систем в виде сети массового обслуживания возникает в том случае, если в схемах общего вида отсутствуют независимые процессоры, а все процессы пересечены на ресурсах. Поток требований на каждый контроллер есть не что иное, как поток инициаторов процессов, а каждое отдельное требование, будучи инициатором, определяет процесс.