УДК 681.3

А.К. ТУМАНЯН, Э.В. ВИРАБЯН

АРБИТРЫ В СОВРЕМЕННЫХ КОМПЬЮТЕРНЫХ СИСТЕМАХ

В большинстве современных цифровых систем имеются разделяемые ресурсы. Они представлены функциональными блоками и узлами, а также их межсоединениями, используемыми поочередно разными устройствами. Как правило, запросы поступают от устройств системы, которые осуществляют параллельную обработку и требуют в определенные моменты доступа к разделяемым ресурсам.

Ключевые слова: разделяемые ресурсы, арбитры, round-robin арбитр, приоритетный арбитр.

Разделяемые ресурсы в современных вычислительных системах есть на всех уровнях взаимодействия. На аппаратном уровне разделяемыми ресурсами могут быть: оперативная память при доступе к данным от разных обработчиков, центральный процессор в случае поступления нескольких запросов аппаратных прерываний или магистральная системная шина, используемая для обмена данными между процессорами, оперативной памятью и адаптерами периферийных устройств.

Рассмотрим решение задачи доступа к разделяемым ресурсам на аппаратном уровне. Кратко такая задача называется арбитражем, а реализующий ее аппаратный узел — арбитром.

Примеры применяемых в компьютерных системах арбитров:

- арбитры шины;
- арбитры памяти.

Арбитры шины используются в многопроцессорных системах, в которых несколько процессоров разделяют между собой системную шину. При этом схемы арбитража должны разрешать конфликтные ситуации, возникающие при одновременной попытке доступа к системной шине со стороны нескольких процессоров.

Некоторые системы, такие как обычная PCI шина, имеют одно централизованное устройство арбитража, которое является арбитром шины. Другие системы используют децентрализованный арбитраж шины, когда все устройства взаимодействуют, чтобы решить, кто будет следующим.

Арбитр памяти - это устройство, используемое в системе с общей памятью. Арбитр принимает решение, какому процессору в каждом цикле памяти будет предоставлен доступ. Арбитр требуется и при использовании многопортовой памяти.

Арбитр памяти обычно интегрирован в контроллер памяти и в контроллер прямого доступа к памяти (DMA).

При выполнении атомарных (неделимых) инструкций арбитр должен предотвратить чтение памяти другими процессорами на полпути (например, при выполнении инструкции “чтение-изменение-запись” - read-modify-write) [1].

Когда каждый процессор, подключенный к арбитру памяти, имеет синхронизированные циклы доступа к памяти, арбитр памяти может быть спроектирован как синхронный арбитр. В противном случае, арбитр памяти должен быть спроектирован как асинхронный арбитр.

Общая структура системы с использованием арбитра представлена на рис.1.

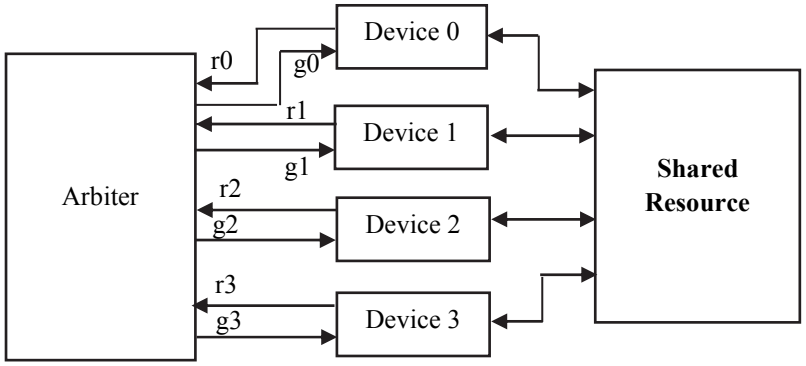


Рис. 1. Структура системы с арбитром

Задача арбитра: управление доступом к общему ресурсу по запросам от четырех устройств. r0, r1, r2, r3 – запросы на обслуживание со стороны четырех устройств: device0, device1, device2, device3. g0, g1, g2, g3 – сигналы разрешения обслуживания, предоставляемые арбитром.

Общим ресурсом может быть память, switch, специализированная FSM, или какой-либо другой сложный вычислительный элемент.

Существует несколько вариантов обслуживания устройств, использующих общий ресурс: по приоритетам и по круговому циклу (round-robin).

Обслуживание по приоритетам может осуществляться двумя способами: с фиксированными приоритетами и с изменяемыми в процессе вычислений приоритетами.

Структура схемы обслуживания round-robin приведена на рис. 2 [2].

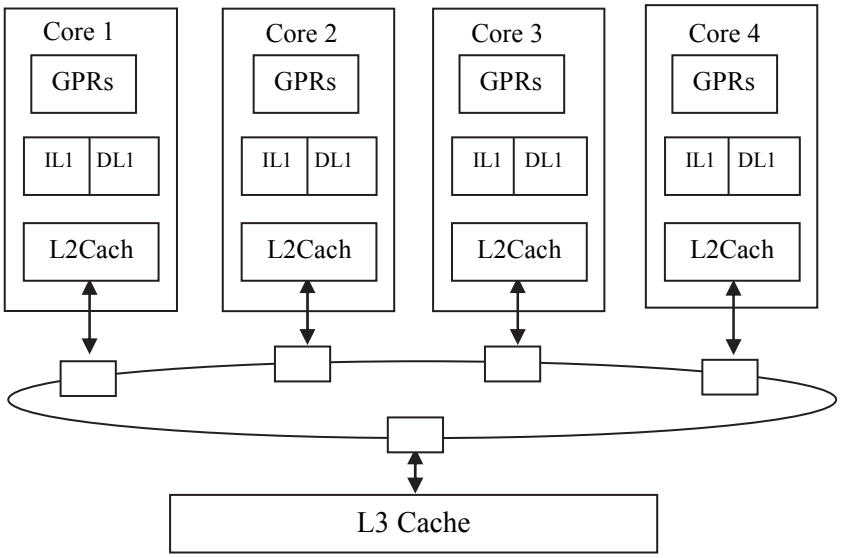


Рис. 2. Иерархия памяти современного процессора

Каждое ядро процессора имеет собственные кэши команд (IL1) и данных (DL1), а также объединенный кэш L2. Процессоры связаны с собственными кэшами по выделенным каналам. На следующем уровне иерархии находится кэш L3.

Кэши L2 связываются с общим кэшем L3 по сети типа “round-robin”. Запрос, входящий в такую сеть, передается следующему узлу. Каждый узел проверяет, не достиг ли запрос своего целевого узла.

Процесс передачи происходит до тех пор, пока не будет найден целевой узел, то есть источник запроса. Основное преимущество кольцевой сети – недорогая реализация при высокой пропускной способности.

Арбитр решает задачу распределения общего ресурса между запрашивающими устройствами. При разработке арбитра требуется:

- разработать интерфейс;
- разработать схему управления (автомат или FSM).

Простейшая схема round-robin арбитра представлена на рис.3.

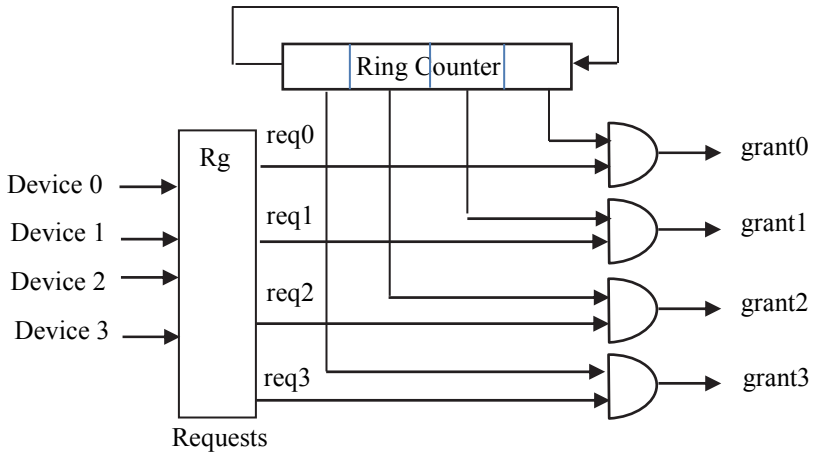


Рис. 3. Простейшая схема round-robin арбитра

Запросы от устройств фиксируются в регистре запросов. Управляет порядком обслуживания запросов кольцевой счетчик (ring counter).

Пример. Пусть в некоторый момент времени содержимое кольцевого счетчика будет 0001, а в регистре запросов находится код 1001, т.е. обслуживания требуют устройства – device 0 и device 3. От устройств 1 и 2 запросов нет. Тогда запрос от устройства 3 (req3) будет обслужен только через два такта после обслуживания запроса req0. Это основной недостаток такой схемы. Современные процессоры содержат большое количество ядер, и этот недостаток становится очень существенным при обращении к L3 кэшу.

Усложненная схема арбитра приведена на рис.4 и работает следующим образом [3]. В состав данной схемы входят 4 приоритетные схемы: priority_circuit0, priority_circuit1, priority_circuit2, priority_circuit3. Выходы кольцевого счетчика t0, t1, t2, t3 управляют входами разрешения соответствующих приоритетных схем. Рассмотрим предыдущий пример. В этом случае запрос от устройства 3 будет обслужен в следующем такте после обслуживания запроса r0. Следовательно, если какой-либо запрос ожидает своей очереди, он будет обслужен вовремя.

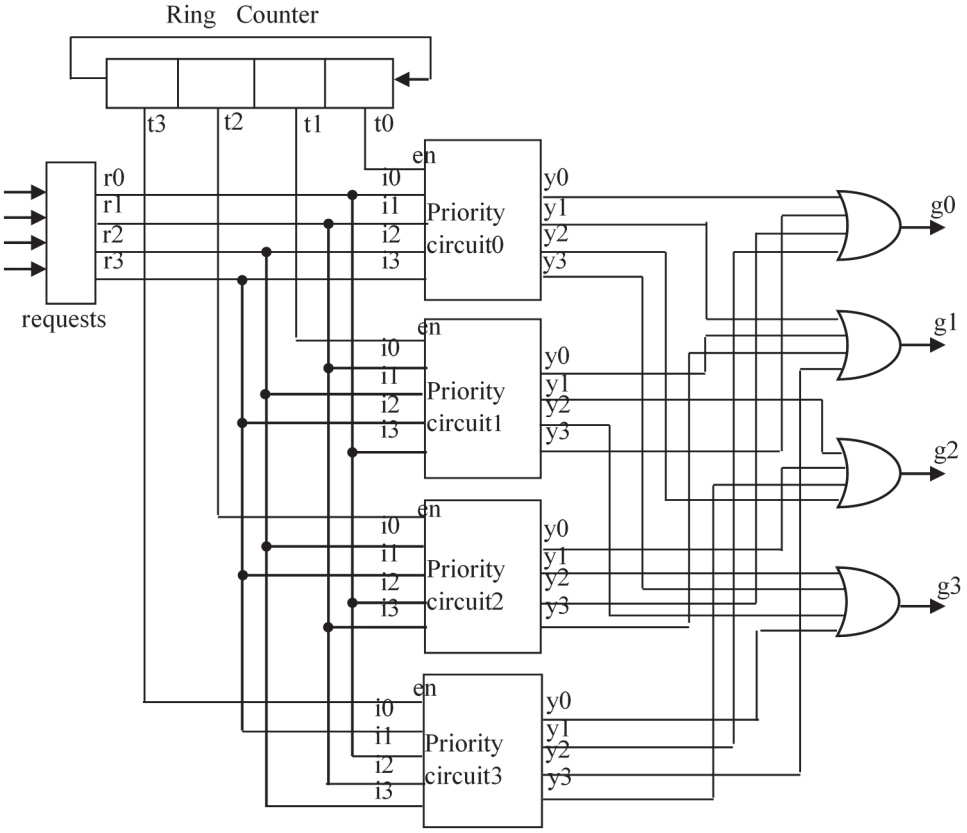


Рис. 4. Усложненная схема round-robin арбитра

Недостатком round-robin арбитра является то, что к каждому запрашивающему устройству применяется одинаковое отношение. Если нам нужно какому-то определенному устройству предоставлять общий ресурс чаще, мы не можем этого сделать. Единственный выход - предоставление нескольких линий запросов данному устройству.

Построение арбитра с фиксированным приоритетом

Пример обслуживания по приоритету представлен временной диаграммой, приведенной на рис. 5. Приоритеты устройств: r0>r1>r2>r3.

Устройство 1 вырабатывает запрос на обслуживание первым. Так как нет более приоритетного запроса r0, арбитр предоставляет устройству 1 обслуживание (grant), устанавливая на выходе сигнал g1. После завершения операции передачи данных запрос r1 сбрасывается в 0, что приводит к установке в 0 и сигнала g1. К этому моменту обслуживания требуют устройство 0, устройство 2 и устройство 3. Арбитр предоставляет грант (g0) устройству 0, так как у

него более высокий приоритет, чем у остальных устройств. После окончания операции запрос r0 устанавливается в 0. Предоставляется грант устройству 2, а затем устройству 3.

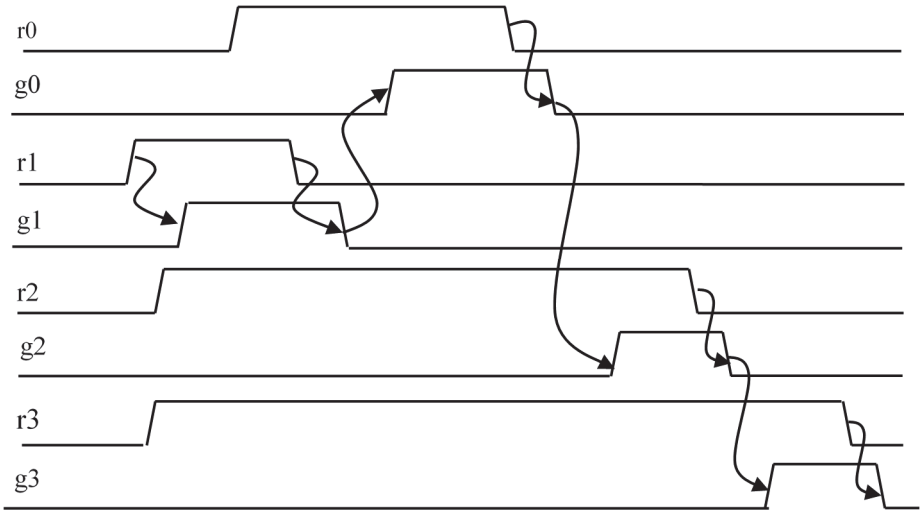


Рис. 5. Предоставление обслуживания устройствам на основе приоритетов

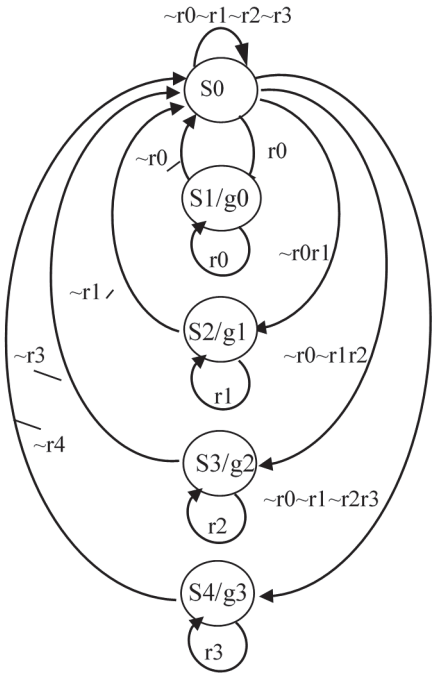


Рис. 6. Граф переходов арбитра

Недостатком данной схемы является то, что устройства с меньшими приоритетами могут быть обслужены с большой задержкой.

Граф автомата арбитра с фиксированными приоритетами представлен на рис. 6.

Автомат задается с помощью пяти множеств A={X,Y,S, δ,λ}.
X={r0,r1,r2,r3}; Y={g0,g1,g2,g3}.
S={S0,S1,S2,S3,S4}.

Функции переходов и выходов представлены графом автомата на рис.6.

Изменение приоритетов в такой схеме невозможно.

Построение арбитра с изменяемыми в процессе вычислений приоритетами

Рассмотрим пример для четырех запросов. Число возможных вариантов приоритетов будет 16. Ограничиваемся 4-мя вариантами. Обозначим эти варианты переменными p0, p1, p2, p3 (назовем их векторами):

p0=r0>r1>r2>r3, p1=r1>r2>r3>r0, p2=r2>r3>r1>r0, p3=r3>r2>r1>r0.

Формирование сигналов p0,p1, p2,p3

В схеме интерфейса между устройствами, которые используют общий ресурс, предусматриваем специальный регистр номеров позиций устройств (рис.7). Регистр состоит из четырех двухбитных полей, содержащих номера устройств. Позиции с номером N0 соответствует самый высокий приоритет, позиции с номером N3 – самый низкий приоритет (уровень приоритета 3).

Приоритет убывает в соответствии со стрелкой (знак >), т.е. верхние два бита номера устройства соответствуют устройству с наивысшим приоритетом 0, а нижние два бита – устройству с самым низким приоритетом 3.

При изменении приоритета каждому устройству ставится в соответствие номер позиции в регистре.

На изменение приоритетов автомат реагирует только тогда, когда он находится в начальном состоянии.

Значения приоритетов в зависимости от порядков номеров приведены в таблице.

Таблица

Значения приоритетов в зависимости от порядков номеров

p0	p1	p2	p3
r0 (N0)	r3(N0)	r2(N0)	r1(N0)
r1(N1)	r0(N1)	r3(N1)	r3(N1)
r2(N2)	r1(N2)	r0(N2)	r2(N2)
r3(N3)	r2(N3)	r1(N3)	r0(N3)

Схема формирования порядков приоритетов при заданных вариантах приоритетов приведена на рис. 7.

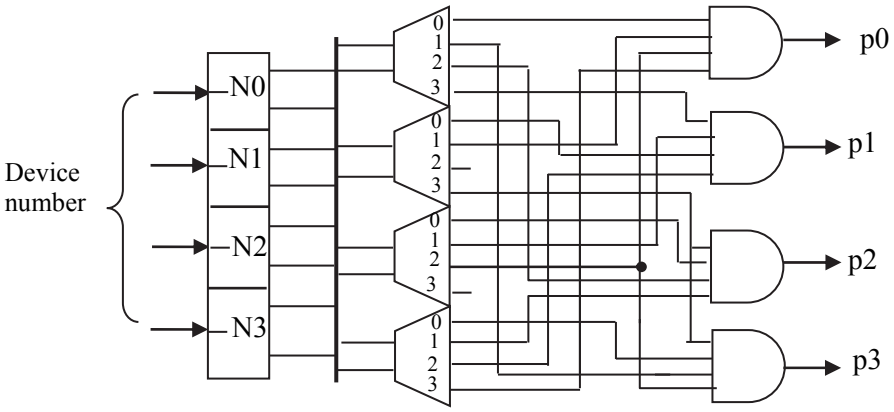
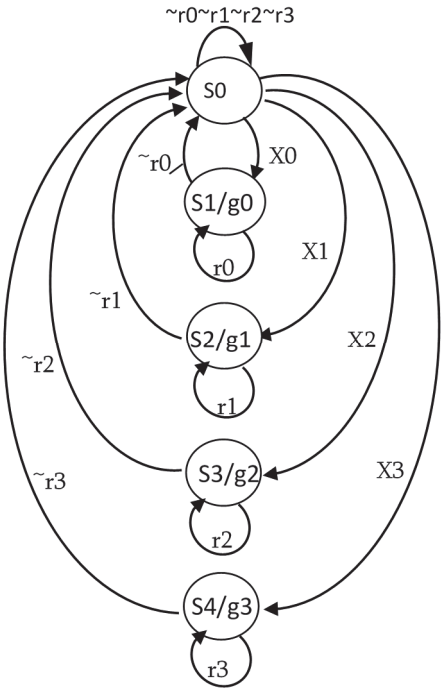


Рис. 7. Формирование сигналов p0, p1, p2, p3

Построение графа автомата

Автомат задается с помощью следующих пяти множеств: A={X,Y,S,δ,λ}. X={p0,p1,p2,p3,r0,r1r2,r3}; Y={g0,g1g2,g3}. Функции переходов и выходов представлены графом автомата на рис.8. Как и в предыдущем примере, это автомат Мура.



X0,X1,X2,X3 – функции перехода из состояния S0 в состояния S1,S2,S3,S4 соответственно. X0 →S1, X1→S2, X2→S3, X3→S4.
X0=p0r0+p1˜r1˜r2˜r3r0+p2˜r2˜r3˜r1r0+
+p3˜r3˜r2˜r1r0;
X1=p0˜r0r1+p1˜r3˜r0r1+p2˜r2˜r3˜r0r1+
+p3r1;
X2=p0˜ro˜r1r2+p1˜r3˜r0˜r1r2+p2r2+
+p3˜r1˜r3˜r2 +p3˜r1˜r3r2;
X3=p0˜r0˜r1˜r2r3+p1r3+p2˜r2r3 +
+p3˜r1r3

Рис. 8. Граф переходов арбитра с изменяемыми приоритетами

Анализ полученных выражений для функций переходов приводит к следующему выводу: комбинационная логика функций переходов автомата может быть представлена с использованием 4-ех приоритетных схем (рис. 9).

Сигналы p0,p1,p2,p3 являются сигналами разрешения приоритетных схем priority circuit 0, priority circuit 1, priority circuit 2, priority circuit 3.

Основное преимущество приоритетного арбитра перед round-robin арбитром заключается в том, что мы можем отдать предпочтение любому конкретному запрашивающему обслуживания устройству, изменив вектор приоритетов.

Число приоритетных схем определяется количеством реализованных вариантов приоритетов. В рассмотренном выше примере их всего 4.

Недостатком схемы является ее усложнение, но возможность изменения приоритетов при обслуживании в зависимости от решаемых задач увеличивает гибкость системы.

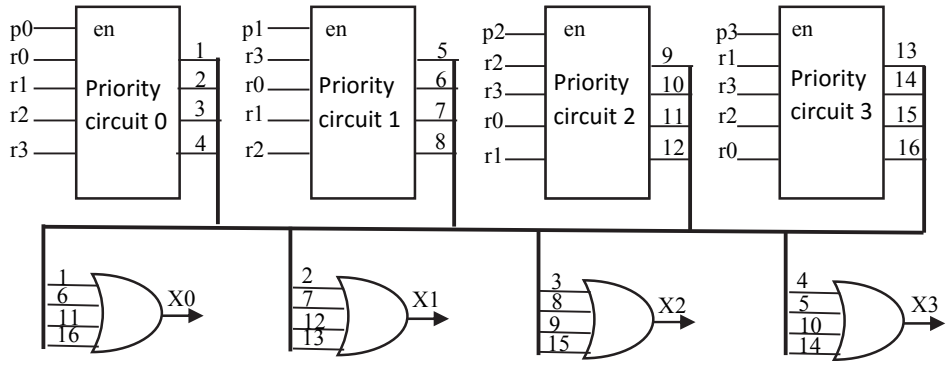


Рис. 9. Схема формирования входных сигналов X0, X1, X2, X3 арбитра

Заключение. Произведен анализ двух вариантов схем арбитров, используемых в современных компьютерных системах.

Предложен вариант схемы арбитра с изменяемыми приоритетами обслуживаемых устройств. Разработаны схема формирования векторов приоритетов и граф переходов арбитра.

Изменение вектора приоритетов приводит к изменению порядка обслуживания запросов от устройств.

СПИСОК ЛИТЕРАТУРЫ

1. **Patterson D., Hennessy J.** Computer Organization and Design. - 6-th edition.- 2021.
2. **Таненбаум Э., Остин Т.** Архитектура компьютера. - 6-е издание. – СПб.: Питер, 2014. – 816с.

3. **TharunTeja K., PonnadaVelkataSai Siva Tarun.** Implementation of Round Robin Arbiter Using Verilog //International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineearing.- October, 2015.

Ա.Կ. ԹՈՒՄԱՆՅԱՆ, Է.Վ. ՎԻՐԱԲՅԱՆ

ԱՐԲԻՏՐՆԵՐԸ ԺԱՄԱՆԱԿԱԿԻՑ ՔՈՄՓՅՈՒԹԵՐԱՅԻՆ ՀԱՄԱԿԱՐԳԵՐՈՒՄ

Ժամանակակից թվային սարքերի մեծ մասն ունի բաշխված ռեսուրսներ: Դրանք ներկայացված են ֆունկցիոնալ բլոկներով և հանգույցներով, ինչպես նաև դրանց միջմիացումներով, որոնք իրենց հերթին օգտագործվում են տարբեր սարքերում: Որպես կանոն, հայցերը տրվում են համակարգի սարքերից, որոնք իրականացնում են զուգահեռ մշակում և որոշակի պահերին պահանջում են բաշխված ռեսուրսներին հասանելիություն:

Առանցքային բառեր. բաշխված ռեսուրսներ, արբիտրներ, round-robin արբիտր, նախընտրելի արբիտր:

A.K. TUMANYAN, E.V. VIRABYAN

ARBITERS IN MODERN COMPUTER SYSTEMS

Most modern digital systems have shared resources. They are presented by functional blocks and nodes, as well as their interconnections used in turn by different devices. As a rule, requests come from system devices that perform parallel processing and require access to shared resources at certain times.

Keywords: shared resources, arbiters, round-robin arbiter, priority arbiter.