УДК 004.072.4

Оптимизация подсистемы памяти вычислительной системы с помощью предоставления гарантированной полосы пропускания канала памяти

А.В. Корниленко, О.И. Эсула

ФГУ ФНЦ НИИСИ РАН, akorn@cs.niisi.ras.ru

***Аннотация* — В статье представлено описание алгоритмов арбитража потоков запросов к памяти, которые были реализованы в процессорах 1890ВМ5, 1890ВМ6 и 1890ВМ8. Приведено обоснование выбора алгоритма арбитража для процессора 1890ВМ9.**

***Ключевые слова* — система на кристалле, СнК, QoS, подсистема памяти.**

I. Введение

В современных микропроцессорных системах контроллер внешней памяти принимает запросы на запись и чтение от ряда устройств (микропроцессорное ядро, а также SATA, USB, Ethernet и другие периферийные устройства). Хотя задача распределения приоритетов между несколькими абонентами-запросчиками, обращающимися к одному абоненту-ответчику, имеет известные решения [1], в конечном итоге выбор механизма арбитража потоков запросов в память зависит в первую очередь от технических требований. Проблема выбора механизма арбитража потоков запросов в память стояла при разработке микропроцессоров с архитектурой КОМДИВ64, таких как 1890ВМ5, 1890ВМ6, 1890ВМ8 и 1890ВМ9 (в дальнейшем упоминается только номер процессора, без серии). При этом каждая новая версия процессора являлась развитием предыдущей. Наряду с изменениями, касающимися микропроцессорного ядра и производительности системного контроллера, увеличивался набор периферийных устройств. Однако пропускная способность подсистемы динамической памяти с переходом на более высокоскоростные микросхемы типа DDR2 и DDR3 возрастала не прямо пропорционально частоте работы контроллера памяти по причине увеличения количества тактов в циклах чтения-записи. Поэтому возникала необходимость настраиваемого использования пропускной способности подсистемы динамической памяти. В статье описаны реализованные в микропроцессорах ВМ5, ВМ6 и ВМ8 алгоритмы арбитража потоков запросов к памяти. Представлено обоснование выбора алгоритма арбитража потоков запросов к памяти для ВМ9, позволяющего обеспечить требуемую ширину пропускания канала для всех устройств.

II. Алгоритмы арбитража потоков запросов к памяти в проектах ВМ5, ВМ6 и ВМ8

В контроллере памяти каждого из рассматриваемых в статье проектов находится арбитр запросов от устройств. Арбитр необходим для определения порядка доступа к общему ресурсу памяти. В арбитре используется алгоритм упорядочивания, с помощью которого раздаются приоритеты устройствам, одновременно выставившим запросы в память. Арбитр гарантирует доступ к памяти только одного устройства.

В микропроцессоре ВМ5, изготовленном по проектным нормам 0,18 мкм, использовался тип внешней динамической памяти DDR. ВМ5 включает в себя процессорное ядро, контроллер интерфейса PCI, контроллер Ethernet и контроллер памяти, содержащий арбитр запросов. На рис. 1 представлена схема обращения устройств к памяти.

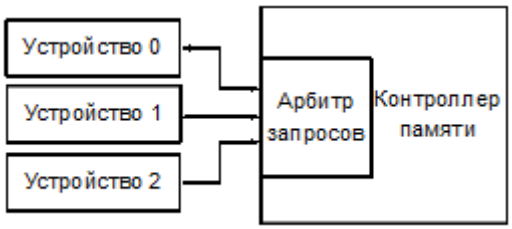

Рис. 1. Схема обращения устройств к памяти

Устройств, обменивающихся данными с памятью, всего три. При таком количестве устройств для поддержания необходимого уровня производительности достаточным является использование фиксированных приоритетов при арбитрации.

Микропроцессор ВМ6 является развитием архитектуры процессора ВМ5. Внесены изменения в процессорное ядро (например, стал поддерживаться пакетный механизм Write Back), разработан сопроцессор СР2 (векторный сопроцессор для быстрого преобразования Фурье и комплексной арифметики) и добавлены контроллеры интерфейсов RapidIO и USB. Тип внешней динамической памяти – DDR.

Количество устройств, обращающихся к памяти, увеличено до семи. Благодаря аппаратной реализации

алгоритма распределения приоритетов Round-Robin [2] гарантирован доступ к памяти всем устройствам (отсутствие ситуации, когда устройство с наивысшим приоритетом занимает всю полосу пропускания), стал возможен расчет времени ожидания (оно пропорционально количеству устройств минус один), обеспечен уровень производительности, отвечающий требованиям применения [3]. Для контроля производительности и набора статистики в систему добавлены 64-разрядные счётчики количества обработанных запросов от устройств.

Микропроцессор ВМ8 изготавливался с использованием технологического процесса 65 нм. Тип внешней динамической памяти – DDR, DDR2 или DDR3. Добавлены отражённые в техническом задании интерфейсы, такие как SATA, DVI, звуковой интерфейс, интерфейс гигабитного Ethernet, а также встроенный ускоритель 2D-графики. Осуществлён переход с параллельной микросхемы ПЗУ NOR на последовательную SPI.

Количество устройств, обращающихся в память возросло, но количество каналов контроллера памяти осталось семь, поэтому добавлен еще один уровень арбитража с переключаемыми по алгоритму Round-Robin приоритетами (расположение блоков коммутатора представлено на рис. 2). Для арбитрации запросов внутри контроллера памяти реализован алгоритм распределения приоритетов Least Recently Used (LRU) [4], отличающийся тем, что его реализация не требует сложной логики с большим количеством элементов и не влияет в конечном итоге на максимальную частоту памяти. Также этот алгоритм не требует долгого RTL-моделирования для верификации [5], для прохождения тестов достаточно нескольких часов.

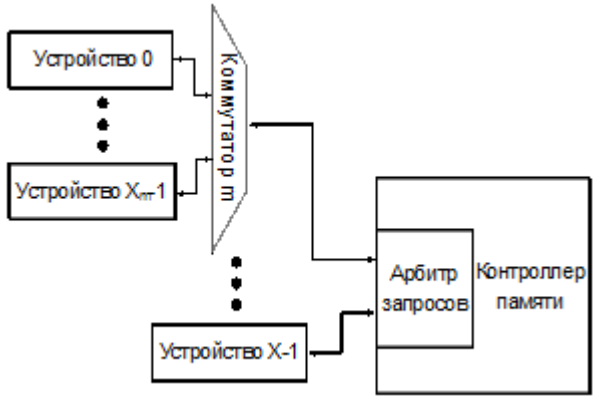

Рис. 2. Схема обращения устройств к памяти с использованием дополнительных коммутаторов; X_m – количество устройств, обращающихся к памяти через коммутатор m; X – общее количество устройств

III. Особенности доступа в память в микропроцессоре ВМ9

Микропроцессор ВМ9 является дальнейшим развитием линейки процессоров КОМДИВ64, начатой ВМ5. Алгоритмы арбитрации запросов к памяти, используемые в предыдущих версиях контроллеров памяти, не позволяют обеспечить необходимый уровень качества обслуживания устройств (Quality of service, QoS) [6] во всех режимах, требуемых в техническом задании. Так, на ПЛИС-прототипе при разрешении 1024x168 и частоте памяти DDR3 75 МГц при интенсивном обращении других устройств к памяти на экране наблюдается дрожание изображения из-за недостаточной пропускной способности канала памяти, выделенного под вывод на экран.

По этим причинам усложнён механизм LRU в области предоставляемой устройству ширины полосы пропускания. Каждому каналу контроллера памяти присваивается некоторое число (обозначается далее n_i, где i – номер канала), определяющее количество запросов, имеющих неизменный приоритет.

Блок-схема усложнённого механизма LRU для одного запроса показана на рис. 3. В первую очередь проверяется наличие запроса от устройства. Далее проверяется наличие другого запроса с более высоким приоритетом. Если есть такой, то значение приоритета увеличивается на единицу. Иначе предоставляется доступ к памяти. Приоритет становится минимальным при $n_i = 0$. Но если n_i больше 0, то приоритет не изменяется, а $n_i = n_i - 1$.

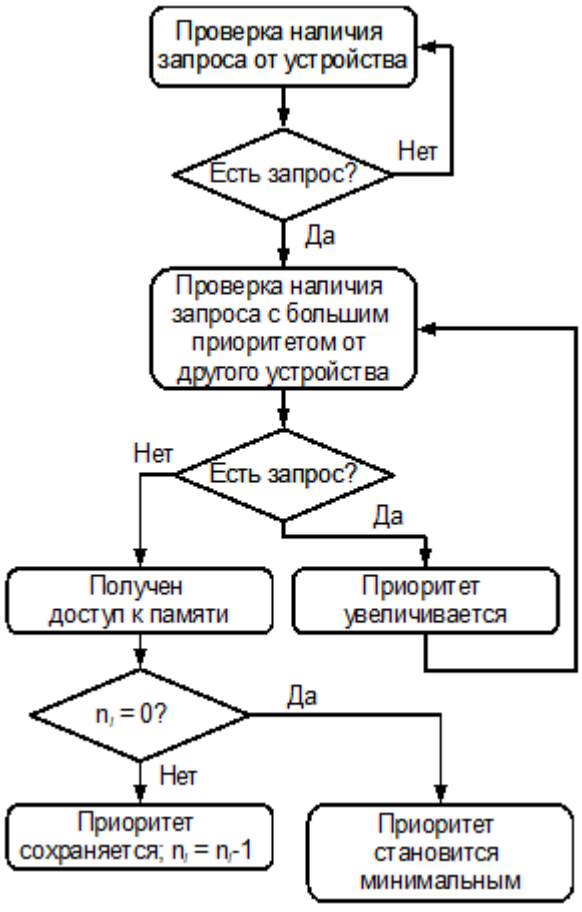

Рис. 3. Блок-схема усложнённого механизма LRU для одного запроса

Введенные меры позволяют гарантировать для канала получение или передачу количества данных, определяемого формулой (1):

$$M_K = \frac{n_K + 1}{\sum_{i=0}^{N-1} n_i + 7} \times M_B \,, \qquad (1)$$

где:
K – номер выбранного канала контроллера памяти,
M_K – гарантированная пропускная способность канала,
M_B – максимальная пропускная способность шины памяти,
N – общее количество каналов, равное 7.

IV. Обоснование выбранного решения

Выбор разрядности числа n_i обусловлен соотношением между затрачиваемыми ресурсами и относительной пропускной способности канала (M_K/M_B). Для анализа использован график, изображенный на рис. 4, сделанный по формуле (1) для $n_i = 0$ и для всех $i = 0, .., N\text{-}1$.

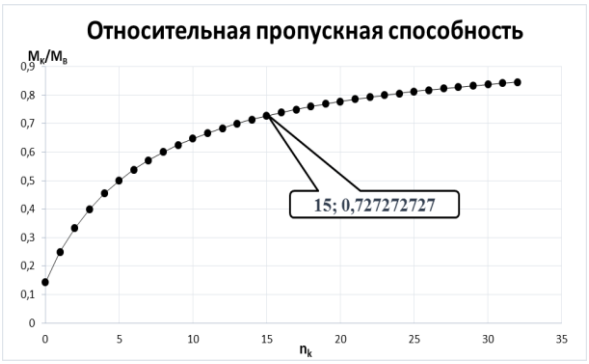

Рис. 4. График относительной пропускной способности канала K в зависимости от n_k

Как видно по графику, прирост относительной пропускной способности канала при $n_k = 8$ превышает 7% ($M_K/M_B = 0{,}57$ при $n_k = 7$, $M_K/M_B = 0{,}6$ при $n_k = 8$), однако становится уже незначительным, начиная с $n_k = 16$ ($M_K/M_B = 0{,}73$ при $n_k = 15$, $M_K/M_B = 0{,}74$ при $n_k = 16$, прирост меньше 2%). Следующие ступени (от 17 до 31) дают ещё меньший прирост, таким образом, повышение величины n_k не приводит к значимому росту пропускной способности канала. На практике для конечного пользователя рост производительности останется незаметным на фоне обращений от других устройств. Следовательно, оптимальным диапазоном значений будет n_k = [0 ... 15]. На основании изложенного выше для процессора ВМ9 решено использовать 4 бита для числа n_k, кодирующих 16 значений.

Таким образом, динамическое переключение позволяет изменять ширину полосы конкретного канала в зависимости от выполняемой вычислительной системой задачи в процессе работы. В итоге, если использовать только алгоритм арбитрации LRU, то при равной интенсивности запросов на всех каналах пропускная способность любого канала равна 1/7 от максимальной пропускной способности шины памяти. При этом алгоритм, реализованный в ВМ9, позволяет при необходимости гарантировать выбранному каналу памяти до 0,7 от максимальной пропускной способности шины памяти.

Рассматривая приведённый ранее пример с выводом изображения на экран, вычислим пропускную способность канала памяти для контроллеров памяти ВМ8 и ВМ9. На ПЛИС-прототипе полная пропускная способность шины памяти составляет 1,2 Гбайт/сек без учёта времени на переключение страниц памяти и регенерацию. Вывод изображения 1024x768 с частотой 60 Гц глубиной цвета 32 бита на точку требует 180 Мбайт/сек. На ВМ8 при алгоритме LRU на один канал памяти при наличии запросов на других шести каналах может быть отдано только 1200 / 7 = 170 Мбайт/сек. Этой пропускной способности недостаточно, что проявляется в виде непериодического дрожания изображения. На ВМ9 при максимальном значении n_k = 15 пропускная способность канала памяти может составить 850 Мбайт/сек, что более чем достаточно для вывода на экран при выбранном разрешении.

V. Оценка эффективности алгоритмов арбитража

Рассмотрим повышение эффективности алгоритма доступа к памяти с развитием линейки процессоров. В соответствии с техническими требованиями в проектах ВМ5, ВМ6, ВМ8 и ВМ9 использовались разные алгоритмы арбитража: алгоритм с фиксированными приоритетами, Round-Robin, LRU и LRU с механизмом гарантированного предоставления ширины полосы пропускания (далее модернизированный алгоритм LRU).

Алгоритм с фиксированными приоритетами гарантированно предоставляет доступ к памяти только одному устройству с наибольшим приоритетом. Если устройство с наибольшим приоритетом будет постоянно выставлять запросы, то время ожидания доступа в память запросов от других устройств может стать бесконечным.

Эффективность других алгоритмов зависит от того, в какой момент устройство выставит запрос. Рассмотрим две ситуации: устройство выставляет запрос сразу после предыдущего получения доступа в память (ситуация 1) и устройство выставляет запрос после получения доступа в память всеми остальными устройствами (ситуация 2).

Максимальное время ожидания доступа в память t_{max} при использовании алгоритма Round-Robin в ситуации 1 описывается формулой (2):

$$t_{\max} = t_{mem} + t_{burst} \times (N-1) \,, \qquad (2)$$

где t_{mem} – время получения доступа в память, t_{burst} – время получения данных, N – общее количество

каналов, равное 7. В ситуации 2 максимальное время ожидания доступа в память останется таким же.

Алгоритм LRU имеет преимущество перед предыдущим алгоритмом в том, что имея в ситуации 1 максимальное время ожидания доступа в память, описываемое формулой (2), в ситуации 2 имеет нулевое время ожидания.

Модернизированный алгоритм LRU в ситуации 1 будет иметь время t_{max}, описываемое формулой (2), при n_k = 0 (параметр описан в формуле (1)) и нулевое время при n_k не равном 0. В ситуации 2 максимальное время ожидания доступа в память $t`_{max}$ описывается формулой (3):

$$t`_{\max} = n_L \times (t_{mem} + t_{burst}) , \quad (3)$$

где n_L – параметр, описанный в формуле (1), присвоенный каналу, который последним получил доступ в память. При n_L = 0 время ожидания также равно 0.

Таблица 1

Максимальное время ожидания доступа в память при использовании различных алгоритмов арбитража запросов

Название алгоритма	Максимальное время ожидания доступа в память	
	В ситуации 1	В ситуации 2
Алгоритм Round-Robin	tmax	tmax
Алгоритм LRU	tmax	0
Модернизированный алгоритм LRU	tmax, при nk = 0; 0, при nk > 0	t`max

Таким образом, из таблицы 1 видно, что модернизированный алгоритм LRU имеет преимущество перед предыдущими использованными алгоритмами в том, что позволяет настроить время ожидания прихода данных для выбранного канала в соответствии с требованиями технического задания и условий работы.

VI. Заключение

Рассмотрены алгоритмы арбитража потоков запросов, соответствующие различным требованиям подключенных устройств к пропускной способности памяти. Аппаратная реализация выбранных в зависимости от технических требований алгоритмов настройки приоритетов потоков запросов в память гарантирует всем устройствам необходимые ресурсы при условии, что их суммарные требования не превышают общей пропускной способности шины памяти. Алгоритм арбитража, реализованный в BM9, предоставляет более качественное и предсказуемое обслуживание по сравнению с предыдущими версиями процессора. Эффективность использованных методов обеспечения качества обслуживания для процессоров BM5, BM6, BM8 и BM9 подтверждена результатами тестов производительности (LMbench [7]).

Литература

[1] Mohan S., Joseph A. A Dynamic Priority Based Arbitration Algorithm. //International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278 - 3075, Volume - 3, Issue - 1, June 2013. P. 232-233.

[2] Nosrati M., Karimi R., Hariri M.. Task Scheduling Algorithms Introduction. //World Applied Programming, Vol (2), Issue (6), June 2012. P. 394-398.

[3] Аряшев С.И., Корниленко А.В. Оптимизация работы с памятью на уровне системного контроллера // Электроника, микро- и наноэлектроника. Сб. научн. трудов. - М.: МИФИ, 2011. С. 176-179.

[4] Aravind A.A.. An arbitration algorithm for multiport memory systems. //IEICE Electronics Express, Vol.2, No.19, 1–7.

[5] Аряшев С.И., Зубковская Н.В., Корниленко А.В., Рогаткин Б.Ю. Имитационное моделирование подсистемы памяти микропроцессора со встроенным системным контроллером // Электроника, микро- и наноэлектроника. Сб. научн. трудов. - М.: МИФИ, 2013. С. 118-125.

[6] Grot B., Keckler S.W., Mutlu O.. Preemptive Virtual Clock: A Flexible, Efficient, and Cost-effective QOS Scheme for Networks-on-Chip. //MICRO'09, December 12–16, 2009, New York, NY, USA. P. 1-2.

[7] Ровинский Е.В., Чибисов П.А. Запуск ОС Linux как этап функционального тестирования микропроцессоров // Проблемы разработки перспективных микро- и наноэлектронных систем - 2012. Сборник трудов / под общ. ред. академика РАН А.Л. Стемпковского. М.: ИППМ РАН, 2012. С. 125-128.

Computer memory subsystem optimization by providing guaranteed memory bandwidth

A.V. Kornilenko, O.I. Esula

Federal State Institution "Scientific Research Institute for System Analysis of the Russian Academy of Sciences" (SRISA), akorn@cs.niisi.ras.ru

Keywords **— system-on-a-chip, SoC, QoS, memory subsystem.**

EXTENDED ABSTRACT

In modern microprocessors memory controller receives requests to write or read from a number of system components (microprocessor core, SATA, USB, Ethernet controllers and other peripheral devices). Arbitration challenge has well known decisions [1], which depend on technical requirements. The article describes various algorithms of memory requests arbitration, that were implemented in processors 1890VM5, 1890VM6, 1890VM8 and 1890VM9.

Each memory controller of the given processors has its own arbiter for ordering requests from different devices. However, scheduling algorithms are different in different processors. 1890VM5 has few components, only three, so it was enough to use fixed priority algorithm for arbitration. 1890VM6 has seven system components. Round-Robin [2] algorithm was used to guarantee access to the memory for all the devices and to provide required performance [3]. In 1890VM8 the number of system components increased, but the memory channels number wasn't changed. There is additional commutator level of arbitration with Round-Robin algorithm before the request reaches memory controller. In the memory controller Least Recently Used (LRU) [4] is used. This algorithm does not require long RTL simulation for verification [5], it takes several hours to pass the tests.

Arbitration algorithms, used in previous versions of memory controllers do not provide the necessary level of quality of service (QoS) [6] for all required modes for 1890VM9. Thus, algorithm LRU was complicated. Each channel of the memory controller was assigned a number to indicate the number of requests that have constant priority.

We analyzed how the relative bandwidth of memory channel depends on this assigned number. Based on the results of the analysis for the processor 1890VM9, four bits that encode 16 values are used for the number. Thus, dynamic switching allows us to change the bandwidth of the memory channel based on the tasks in the process.

We evaluated the efficiency of the arbitration algorithms of the given projects. Thus, it is evident that LRU with the mechanism of providing guaranteed bandwidth has the advantage over the other used algorithms. This algorithm allows configuring the time of data waiting for memory channel in accordance with the technical requirements and task conditions.

The effectiveness of the arbitration algorithms used in processors 1890VM5, 1890VM6, 1890VM8 and 1890VM9 was confirmed by the results of performance tests (LMbench [7]).

REFERENCES

[1] Mohan S., Joseph A. A Dynamic Priority Based Arbitration Algorithm. //International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278 - 3075, Volume - 3, Issue - 1, June 2013. P. 232-233.

[2] Nosrati M., Karimi R., Hariri M.. Task Scheduling Algorithms Introduction. //World Applied Programming, Vol (2), Issue (6), June 2012. P. 394-398.

[3] Arjashev S.I., Kornilenko A.V. Optimizacija raboty s pamjat'ju na urovne sistemnogo kontrollera // Jelektronika, mikro- i nanojelektronika. Sb. nauchn. trudov. - Moscow: MEPhI, 2011. Pp. 176-179 (in Russian).

[4] Aravind A.A.. An arbitration algorithm for multiport memory systems. //IEICE Electronics Express, Vol.2, No.19, 1–7.

[5] Arjashev S.I., Zubkovskaja N.V., Kornilenko A.V., Rogatkin B.Ju. Imitacionnoe modelirovanie podsistemy pamjati mikroprocessora so vstroennym sistemnym kontrollerom // Jelektronika, mikro- i nanojelektronika. Sb. nauchn. trudov. - Moscow: MEPhI, 2013. Pp. 118-125 (in Russian).

[6] Grot B., Keckler S.W., Mutlu O.. Preemptive Virtual Clock: A Flexible, Efficient, and Cost-effective QOS Scheme for Networks-on-Chip. //MICRO'09, December 12–16, 2009, New York, NY, USA. P. 1-2.

[7] Rovinskij E.V., Chibisov P.A. Zapusk OS Linux kak jetap funkcional'nogo testirovanija mikroprocessorov // Problemy razrabotki perspektivnyh mikro- i nanojelektronnyh sistem - 2012. Sbornik trudov / pod obshhej redakciej akademika RAN A.L. Stempkovskogo. Moscow: IPPM RAS, 2012. Pp. 125-128 (in Russian).