УДК 004.272.43

ЖАБИН В.И.,
ЖАБИНА В.В.

ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ ОБМЕНА ДАННЫМИ В МУЛЬТИПРОЦЕССОРНЫХ СИСТЕМАХ С НЕОДНОРОДНЫМ ДОСТУПОМ К ПАМЯТИ

Исследуются методы межпроцессорного обмена данными в системах с общей шиной и различными способами доступа к памяти. Рассматривается возможность синхронизации процедур обмена по шине данных и шине управления. Предлагается метод ускорения обмена данными в централизованных системах, основанный на непосредственном доступе каждого процессора к части адресного пространства локальной памяти других процессоров.

Methods of inter-processor data exchange in the common bus systems with different data access schemes are analyzed. Possibility to synchronize the procedure of exchange through data bus and control bus is considered. Method for data exchange acceleration based on each processor direct access to other processors local memory addressing space is proposed for the host systems.

Ключевые слова: мультипроцессорные системы, организация памяти, обмен данными.

Введение

Среди вычислительных систем различного назначения можно выделить класс систем, функционирующих в режиме реального времени. Системы реального времени (real-time system) можно определить как программно-аппаратные комплексы, на реакцию которых накладываются ограничения со стороны внешних факторов. Реакция системы на конкретное внешнее событие оценивается промежутком времени от момента получения информации о внешнем событии до момента завершения его обслуживания. Указанный класс систем является весьма многочисленным, в него входят системы, работающие в контуре автоматического управления техническими объектами и технологическими процессами.

Непрерывное расширение сферы применения систем управления приводит к необходимости решения все более сложных задач. В ряде случаев (например, при управлении быстрыми процессами, при большом числе координат управления) для обеспечения необходимой реакции системы на внешние события необходимо использование параллельных систем [1-4].

Выбор архитектуры систем должен осуществляться с учетом класса решаемых задач. Исследования, выполненные при разработке многокоординатных систем числового программного управления [5] показывают, что алгоритмы управления технологическими процессами имеют преимущественно последовательно-параллельную структуру (содержат параллельные и последовательные участки). Кроме того, для рассматриваемого класса алгоритмов характер обмена данными является преимущественно трансляционным и дифференциальным. Для этого коммуникационная среда вычислительных систем должна обеспечивать полный граф межпроцессорных связей.

Для одновременной обработки параллельных ветвей алгоритмов должна быть обеспечена возможность одновременного выполнения в разных процессорах различных команд над различными данными. С точки зрения функциональной классификации Флинна [6] системы, которые позволяют выполнять одновременно разные команды в разных процессорах, относятся к классу MIMD (Multiple Instruction, Multiple Data – множественный поток команд, множественный поток данных). В данный класс входят системы, существенно отличающиеся структурной организацией и способами межпроцессорного обмена информацией. В соответствии с механизмом обмена информацией между процессорами можно выделить две основные разновидности систем: мультипроцессорные вычислительные системы (МВС) и распределенные вычислительные системы (РВС). В МВС обмен данными осуществляется словами через общую память, а в РВС процессоры обмениваются пакетами данных через коммуникационную сеть [3, 4].

Применение технологии RSoC (Reconfigurable System on Chip – реконфигурируемая система на кристалле) позволяет с использованием

ПЛИС создавать компактные МВС, которые легко адаптируются к реальным условиям эксплуатации [7, 8]. В дальнейшем рассматриваются системы такого типа.

Выбор архитектуры МВС должен осуществляться с учетом не только минимизации времени параллельной обработки данных в процессорах, но и минимизации затрат на обмен данными между процессорами. Эффективность реализации параллельных ветвей в основном зависит от производительности и количества работающих параллельно процессоров, а эффективность обмена информацией – от организации коммуникационной среды и способов синхронизации процессов обмена. В соответствии с организацией коммуникационной среды обмена данными между процессорами и модулями памяти (внешними устройствами) можно выделить три основные разновидности систем: с многопортовой памятью, с коммутатором и шинной топологией [9].

В устройствах первых двух типов возникает меньше конфликтов при пересылке данных, что обеспечивает большую скорость обмена информацией между компонентами системы. Однако, в состав таких систем должны входить сложные и дорогие устройства коммутации. Кроме того, увеличение числа процессоров в системе требует изменения интерфейсов устройств, что не позволяет обеспечить простоту масштабирования системы.

Следовательно, важной задачей ускорения параллельной обработки данных в мультипроцессорных системах с общей шиной является создание методов и средства ускорения обмена данными между процессорами

Симметричные мультипроцессоры с однородным доступом к памяти

Наиболее простым способом организации коммуникационной среды в МВС является общая шина (магистраль), которая обеспечивает соединение между устройствами только на время передачи информации между ними, то есть осуществляет временное разделение процессов передачи данных. Преимуществом общей шины является низкая стоимость, большие возможности по стандартизации интерфейсов, более простая адаптация программного обеспечения, разработанного для однопроцессорных систем. В таких системах значительно проще решаются вопросы обеспечения отказоустойчивости за счет скользящего резервирования процессорных модулей [10]. Благодаря этому МВС с общей магистралью образуют наиболее многочисленный класс систем автоматического управления.

Наиболее простыми с точки зрения технической реализации являются симметричные мультипроцессоры (symmetric multiprocessor) или SMP-системы, содержащие одинаковые процессоры (процессорные ядра) [3, 4, 11]. Все процессоры имеют одно общее адресное пространство, через которое осуществляется доступ к общему системному ресурсу (к памяти и внешним устройствам).

Общая память используется не только для хранения системной информации и обмена данными между процессорами, но и для хранения программ процессоров. При выполнении своих программ процессоры обращаются к памяти за командами, что приводит к большим затратам времени на ожидание доступа и разрешение конфликтов. Наличие в каждом процессорном модуле кэш-памяти уменьшает число конфликтов при обращении к общей памяти. Однако в этом случае возникают трудности с обеспечением когерентности данных, что усложняет взаимодействие процессоров.

Обмен данными между процессорами осуществляется также через общую шину. Для передачи одного слова от источника к приемнику осуществляется минимум два обращения к шине (для записи и чтения). Кроме этого, присутствуют затраты времени на синхронизацию обмена.

Указанные недостатки приводят к тому, что в таких системах используется небольшое число процессоров (обычно, не более 32), что ограничивает их производительность.

Обмен данными в системах с неоднородным доступом к памяти

С целью ускорения обмена данными между компонентами системы применяют более сложную организацию коммуникационной среды системы, причем, с различными процедурами доступа к различным частям памяти. Такие системы можно отнести к определенному классу NUMA-систем (Non-Uniform Memory Access — «неоднородный доступ к памяти») [4].

Уменьшение числа конфликтных ситуаций при параллельном выполнении программ в процессорах может быть достигнуто за счет использования в каждом процессорном модуле собственной локальной памяти, в которой хра-

нится программа данного процессора [2, 4, 11]. Процессор взаимодействует со своей памятью через локальную шину, а с общей памятью – через общую (системную) шину.

Такой подход уменьшает число обращения к общей шине, то есть уменьшает число конфликтов. Следовательно, при этом уменьшатся продолжительность решения задачи, но на время обмена данными между процессорами это практически не сказывается, так как обмен (как и в системах с однородным доступом к памяти) осуществляется через общую память (минимум два обращения к общей шине).

Для ускорения обмена данными между процессорами в системах может быть предусмотрен непосредственный доступ каждого процессора к определенной области адресного пространства локальной памяти других процессоров, называемой коммуникационной памятью [10]. Для каждого процессора коммуникационная память других процессоров является частью адресного пространства общей памяти системы, доступ к которой осуществляется через общую шину. За счет организации интерфейса доступ к такой памяти возможен как со стороны локальной, так и со стороны общей шины. Арбитраж осуществляется на аппаратном уровне.

Интенсивность обращений процессоров к системной магистрали при обмене данными можно оценить с помощью коэффициента эффективности использования системной магистрали [10], который определяется по формуле

$$K_{\text{СМ}} = \frac{N_S + N_D}{M}, \qquad (1)$$

где N_S – число обращений к системной магистрали для инициализации и синхронизации процедур обмена (непроизводительные затраты времени); N_D – число обращений к системной магистрали для передачи непосредственно данных; M – число передаваемых слов.

Коэффициент $K_{\text{СМ}}$ определяет среднее число обращений к системной магистрали для передачи одного слова между компонентами системы. При равной скорости обработки параллельных ветвей алгоритмов система с меньшим значением $K_{\text{СМ}}$ будет затрачивать меньше времени на решение задач за счет более быстрого межпроцессорного обмена.

Определение $K_{\text{СМ}}$ в общем случае является нетривиальной задачей и требует разработки программ (если известна система команд) или, по крайней мере, алгоритмов синхронизации и пересылки данных на этапе проектирования системы.

Рассмотрим случай, когда в системах с коммуникационной памятью используется централизованный принцип управления. Функции управления процессом обработки данных возлагаются на управляющий (master) процессор, который организует работу подчиненных (slave) процессоров.

В работе [10] исследованы процедуры обмена данными между процессорами для случая синхронизации обмена как по шине данных, так и по шине управления. В первом случае синхронизация обмена осуществляется с помощью примитивов низкого уровня (флажки, семафоры), а во втором – через сигналы прерываний.

Согласно формуле (1) получены следующие зависимости для коэффициента эффективности использования системной магистрали соответственно при синхронизации обмена по шине данных и по шине управления [10]:

$$K_{\text{СМ}} = 2 + 4 / M, \qquad (2)$$

$$K_{\text{СМ}} = 2 + 2 / M. \qquad (3)$$

В обоих случаях для передачи массива необходимо $N_D = 2M$ обращений к общей магистрали. Синхронизация процессов по шине данных требует дополнительно четырех обращений к магистрали ($N_S = 4$). Два обращения требуются для установки в памяти управляющего процессора двух флажков подчиненными процессорами, готовыми к обмену, а еще два – для установки управляющим процессором флажков в памяти подчиненных после завершения передачи массива.

При синхронизации по шине управления подчиненные процессоры могут при готовности к обмену сообщать об этом управляющему процессору сигналом прерывания, что в два раза сокращает число обращений к магистрали ($N_S = 2$). Считается, что сигналы прерывания поступают к управляющему процессору только от подчиненных процессоров. В противном случае при увеличении количества процессоров в системе шина управления существенно усложняется. Процессор обращается к своему блоку формирования сигналов прерывания на локальной шине, выполняя стандартный цикл вывода. Управляющий процессор воспринимает сигнал как требование внешнего прерывания

и переходит на соответствующую подпрограмму обслуживания прерывания.

Как следует из выражений (2) и (3), при увеличении размерности массивов передаваемых данных среднее число обращений к системной магистрали стремится к двум. Рассмотрим метод, позволяющий уменьшить число обращений к магистрали при пересылке массивов данных.

Метод ускорения обмена данными в централизованных системах

В централизованных системах функции управления возлагаются на один управляющий процессор. К основным функциям управляющего процессора можно отнести: планирование и распределение работ между подчиненными процессорами; синхронизация доступа к системному ресурсу; организация обмена данными между подчиненными процессорами. Вычислительный процесс может быть организован таким образом, чтобы управляющий процессор взаимодействовал с подчиненными путем передачи им команд по шине данных. Команды могут передаваться в ячейку памяти или в специальный регистр команд, который включен в адресное пространство, доступное для управляющего процессора. Работа подчиненных процессоров сводится к выполнению команд, которые дает им управляющий процессор. Очередная команда передается процессору после выполнения предыдущей команды, о чем подчиненный процессор сообщает управляющему по шине данных или управления.

Уменьшения числа обращений к системной магистрали в процессе пересылки данных между процессорами в централизованных системах можно достигнуть за счет передачи управления непосредственно пересылкой данных процессору-приемнику информации в соответствии с командой управляющего процессора.

Возможный вариант организации архитектуры системы показан на рис. 1. Система представлена в виде модели на графическом языке [12], который является модификацией языка, описанного в работе [13].

Система рассматривается на функциональном уровне, на котором в качестве объектов видны устройства (процессоры, память, устройства ввода-вывода, системы коммутации и т.д.). Объекты взаимодействуют на уровне протоколов обмена информацией. На функциональном уровне видна архитектура систем, что позволяет при дальнейшей детализации модулей перейти к логическому уровню описания структуры объектов.

Модель представлена двумя компонентами:

-графическим описанием с использованием унифицированных функциональных обозначений;

-картой распределения адресного пространства.

Графическое описание архитектуры является графом, который отображает основные компоненты систем и потоки управляющей информации, которые определяют способ их взаимодействия. Дуги графа ориентированы в направлении от активного элемента, который является инициатором действия, к элементу, с которым это действие должно осуществляться. При этом направление передачи данных может не совпадать с ориентацией дуг.

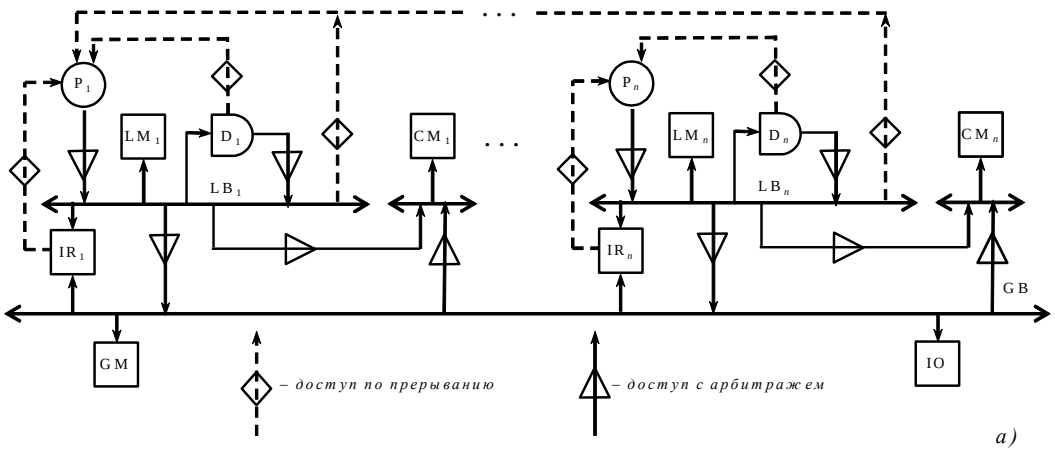
Взаимодействие между процессорами осуществляется следующим образом. Управляющий процессор P_1 пересылает команды, соответствующие определенным заданиям, в регистры IR_i подчиненных процессоров P_i. Запись слова в регистр команд автоматически вызывает прерывание, по которому процессор P_i переходит на подпрограмму выполнения задания (обслуживания прерывания). После выполнения задания подчиненный процессор P_i сообщает об этом управляющему процессору P_1 пересылкой в IR_1 определенного слова, что вызывает соответствующее прерывание. Для ускорения доступа к регистру команд со стороны системной и локальной магистралей он выполнен в виде двухпортового регистра. При одновременном обращении процессоров к магистралям LB_i и GB арбитраж в соответствии с приоритетами может выполняться известными аппаратными средствами [14].

Пересылка массива из процессорного модуля P_i в модуль P_j требует четыре обращения к системной магистрали (к регистрам IR_i, IR_j и два раза к IR_1) для синхронизации процесса и M обращений процессора P_j для пересылки массива. Процессор P_i формирует массив в своей коммуникационной памяти, которая доступна P_j. Процессор P_j пересылает массив в свою локальную память. Следовательно, коэффициент эффективности использования магистрали имеет вид

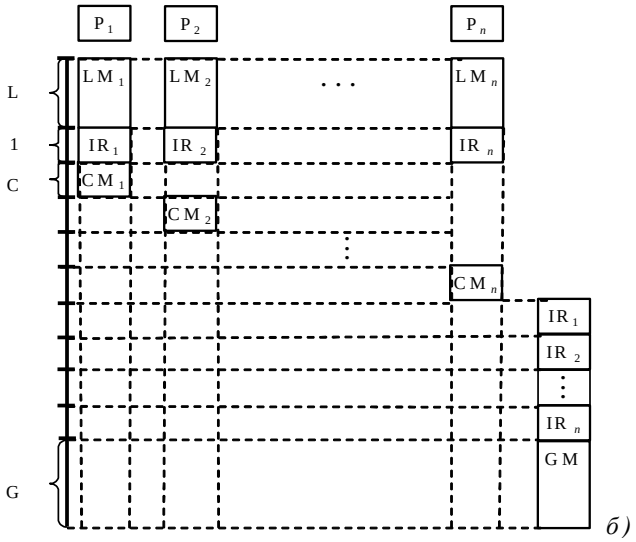
$$K_{\text{СМ}} = 1 + 4 / M \ . \qquad (4)$$

На рис.1а показаны средства прерывания управляющего процессора подчиненными процессорами со своих локальных шин, что не требует двух обращений к системной магистрали для подтверждения выполнения заданий. При таком подходе к организации обмена получим

$$K_{CM} = 1 + 2 / M \ . \qquad (5)$$



а)



б)

Рис. 1. Модель системы: а – граф управляющих потоков; б – карта распределения адресного пространства; P_i ***– процессор;*** D_i ***– контроллер прямого доступа к памяти;*** LM_i ***– локальная память;*** CM_i ***– коммуникационная память;*** GM_i ***– общая память;*** LB_i ***– локальная шина;*** GB ***– общая системная шина;*** IR_i ***– регистр команд;*** IO ***– внешние устройства;*** **C, I, L, G** ***– обозначения объемов памяти***

Указанные средства формирования сигналов прерывания по шине управления не являются обязательными. Они приводят к дополнительным затратам оборудования, но ускоряют пересылку одиночных или малого числа слов.

Сравнивая формулы (2) и (3) с формулами (4) и (5) становится очевидным, что предложенный метод обмена данными примерно вдвое уменьшает число обращений к системной магистрали при пересылке массивов. С увеличением длины массива число обращений к магистрали для пересылки одного слова стремится к минимально возможному значению – одному обращению. Для ускорения пересылки массивов данных в состав процессорных модулей вводят контроллеры прямого доступа к памяти (рис.1). Контроллеры D_i подключены к локальным магистралям процессоров P_i и разделяют с ними часть адресного пространства. Контроллер является комбинированным функциональным элементом, то есть выполняет функции как активного, так и пассивного элемента. В адресном пространстве локальной памяти каждого процессора выделяется область

адресов для обращения к внутренним регистрам контроллера. Для пересылки массива данных процессор настраивает контроллер (передает в регистры начальные адреса, длину массива, указывает режим работы). После запуска контроллер самостоятельно управляет процессом пересылки данных. Об окончании задания он сообщает процессору сигналом требования прерывания.

Использование контроллеров прямого доступа не улучшает коэффициент $K_{СМ}$, поскольку количество обращений к системной магистрали при пересылке массива одинаково как с использованием контроллера, так и без него. Однако контроллер позволяет уменьшить интервалы обращения к магистрали при пересылке данных. Это объясняется тем, что ему не требуется обращаться к памяти за своими командами (что необходимо процессору), поскольку он является устройством с микропрограммным или схемным управлением. Следовательно, к адресному пространству он обращается для пересылки данных с большей частотой, чем процессор. Кроме того, контроллер и процессор могут в определенной мере совмещать во времени свою работу, что ускоряет обработку данных в системе.

Заключение

В работе проведен анализ архитектур мультипроцессорных систем с однородным и неоднородным доступом к памяти с точки зрения эффективности использования коммуникационной среды систем. Показана необходимость уменьшения загрузки коммуникационной среды для ускорения обработки данных.

Предложенный метод пересылки массивов данных обеспечивает уменьшение числа обращений к системной магистрали при обмене данными между процессорами. Уменьшение загруженности коммуникационной среды при пересылке данных позволяет увеличить число процессорных модулей и, следовательно, повысить производительность систем. Все это создает предпосылки для уменьшения времени реализации алгоритмов и, как следствие, расширения области применения систем реального времени.

Список литературы

1. Белецкий В.Н. Многопроцессорные и параллельные структуры с организацией асинхронных вычислений / В.Н. Белецкий. – К.: Наукова думка, 1988. – 240 с.
2. Головкин Б.А. Параллельные вычислительные системы / Б.А.Головкин. – М: Наука, 1980. – 519 с.
3. Корнеев В.В. Параллельные вычислительные системы / В.В.Корнеев. – М.: "Нолидж", 1999. – 320 с.
4. Воеводин В.В. Параллельные вычисления / В.В Воеводин, Вл.В.Воеводин. – СПб.: БХВ, 2002. – 608 с.
5. Жабин В.И. Архитектура вычислительных систем реального времени / В.И.Жабин. – К.: ВЕК+, 2003. – 176 с.
6. Flynn M. Some Computer Organisations and Their Effectiveness / M.Flynn // IEEE Trans. Computers. – 1972. – Vol. 21, N 9. – P. 948-960.
7. DeHon A. Reconfigurable Computing Architectures / A.DeHon, R.Tessier, K.Pocek // Proceedings of the IEEE. – 2015. – Vol. 103, N 3. – P. 332-354.
8. Koch D. Partial Reconfiguration on FPGAs. Architectures, Tools and Applications / D.Koch // – New York: Springer. – 2013. – 295 p.
9. Мультипроцессорные системы и параллельные вычисления / Под. ред. Ф.Г.Энслоу. – М.: Мир, 1976. – 383 с.
10. Жабин В.И. Повышение эффективности параллельной обработки данных и обеспечение отказоустойчивости мультипроцессорных вычислительных систем реального времени / В.И.Жабин // Искусственный интеллект. – 2015. – №1-2. – С. 87-97.
11. Wilson A.B. More Power to You: Symmetrical Multiprocessing Gives Large-Scale Computer Power at a Lower Cost with Higher Availability / A.B.Wilson // Datamation. –1980. – N 6. – P. 216-223.
12. Жабин В.И. Графическое описание архитектуры вычислительных систем / В.И.Жабин // Вісник Національного технічного університету України "КПІ". Інформатика, управління та обчислювальна техніка. – К: "ВЕК+", 2001. – № 36. – С. 80–88.
13. Conte G. Multimicroprocessors systems for real-time applications / G.Conte, D. Del Corco – D. Reidel Publ. Co, 1985. – 298 p.
14. Валях Е. Последовательно-параллельные вычисления / Е.Валях. – М.: Мир, 1985. – 456 с.