УДК 004.272.43

Богуславский О.В.

ОСОБЕННОСТИ МОДЕЛИРОВАНИЯ ВНУТРЕННИХ ШИН ОДНОКРИСТАЛЬНЫХ МНОГОПРОЦЕССОРНЫХ СИСТЕМ

Национальный технический университет Украины
«Киевский политехнический институт»

В статье описываются особенности внутренних шин однокристальных многопроцессорных систем, которые необходимо учитывать при аналитической оценке эффективности реализации параллельных алгоритмов на таких системах. В качестве примера рассматривается двухпроцессорная система с шиной АМБА АНВ, приводится модель этой системы на языке моделирования PAMELA, которая учитывает основные особенности архитектуры системы. Предложены способы оценки средней задержки при передаче данных по шине.

Введение

Однокристальные многопроцессорные системы (ОМС) являются особенным классом параллельных вычислительных систем, поэтому технология производства таких систем отличается от технологий производства классических многопроцессорных систем. В ОМС снимаются многие ограничения, характерные для обычных многопроцессорных систем [1]:

- между блоками, расположенными на одном кристалле, могут присутствовать линии связи большой разрядности (на порядок выше, чем количество выводов микросхем);
- скорость передачи по линиям связи внутри кристалла существенно выше, чем на плате;
- в то время как на платах, как правило, используются шины с мультиплексированием адреса и данных, выходы на которые организуются по схеме с тремя состояниями, на кристаллах, вследствие особенностей технологии, в основном используются шины с раздельными линиями адреса и данных с мультиплексированием выходов на шину;
- в ОМС существенно отличается время передачи по коротким и длинным линиям связи.

Производительность ОМС зависит как от модулей, входящих в состав системы, так и от используемой внутренней шины. В настоящее время существует большое количество стандартов высокопроизводительных шин, использующихся в ОМС [2], из-за чего возникает проблема выбора наилучшего архитектурного решения при построении ОМС для решения определенного набора задач. Необходимо отметить, что зачастую именно характеристики шины ограничивают максимальную производительность ОМС. В связи с этим вычислительные возможности модулей такой системы могут использоваться не полностью. Поэтому при аналитической оценке производительности ОМС с использованием средств языка моделирования ***PAMELA (PerformAnce ModEling LAnguage)*** [3, 4] необходимо решить задачу описания шины ОМС с помощью вышеуказанного языка моделирования. Имея детальное описание всех узлов ОМС, можно получить максимально точную оценку времени выполнения параллельных алгоритмов на таких системах, не прибегая к динамическим методам анализа [5], требующим наличия целевой ОМС и программной реализации алгоритмов.

На данный момент создано много языков описания цифровых устройств, методов их эффективного тестирования и моделирования; прослеживается повсеместный переход на методику ***Transaction Level Modeling {TEM),*** позволяющую работать со сверхсложными однокристальными системами на более высоком уровне абстракции. В отличие от ***RTL***-моделирования (***Register Transfer Level),*** традиционно использующегося для реализации аппаратных средств и описывающего систему на уровне вентилей и регистров, моделирование уровня транза-

кций является более высокоуровневым, т.к. основное внимание при таком моделировании уделяется взаимодействию между компонентами системы (передача по шине, прерывания, сигналы) [6]. В работах [7] и [8] были приведены результаты моделирования систем, построенных на основе шин *AMBA* [9] и *CoreConnect* [10]. При этом для описания функционирования шин использовались конечные автоматы, которые затем записывались с помощью средств языка моделирования *NuSMV* [11].

Основной целью данной статьи является разработка метода описания шин ОМС с помощью языка *PAMELA* для получения детальной модели ОМС, необходимой для аналитического определения времени выполнения параллельных алгоритмов.

Описание модели шины ОМС с помощью языка PAMELA

Язык *PAMELA* [3] позволяет описать как модель вычислительной системы, так и исследуемый алгоритм. В результате подстановки описаний модели архитектуры вычислительной системы в модель алгоритма можно получить аналитическую модель, позволяющую вычислить временную верхнюю и/или нижнюю оценку времени выполнения алгоритма. Возможности этого языка позволяют получать оценки времени выполнения программ, как на последовательных, так и на параллельных вычислительных системах.

Модель на языке *PAMELA* представляет собой последовательность выражений, представленных с помощью примитив *use* и *delay*, разделенных операторами последовательности ";", параллельности "||" либо условным оператором "*if...else*". Примитив *use(Res, t)* указывает на эксклюзивное использование ресурса *Res* на протяжении времени *t*; примитив *delay(t)*– на то, что программа будет выполняться на протяжении времени *t*, и при этом общие ресурсы не будут использоваться.

Для записи параллельных и последовательных наборов одинаковых операций используются следующие формулы приведения:

$$seq(i=m,n)L_i \Leftrightarrow L_m;...;L_n$$
$$par(i=m,n)L_i \Leftrightarrow L_m \| ... \| L_n$$

Рассмотрим абстрактную ОМС, построенную на базе шины *AMBA AHB* (рис. 1), состоящую из 2-х процессоров, являющихся ведущими устройствами, и модулем памяти, являющимся ведомым устройством. Кэш данных каждого процессора работает в режиме прямой записи с модификацией данных в кэше [12].

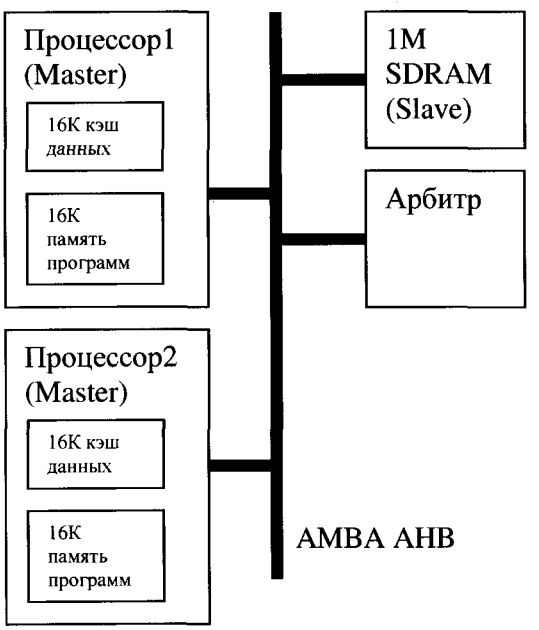


Рис. 1. Структура исследуемой ОМС

Для простоты будем считать, что на обоих процессорах нет кэш-памяти программ, а во внутреннюю память каждого процессора загружены программы *L1* и *L2*, описанные с помощью средств языка *PAMELA* следующим образом:

L = L1 || L2
L1 = seq (i=0..1)
{ read ; read ; add ; mul } ; write
L2 = seq (i=0..2) {
read ; add } ; write

Модель ОМС представлена следующим образом:

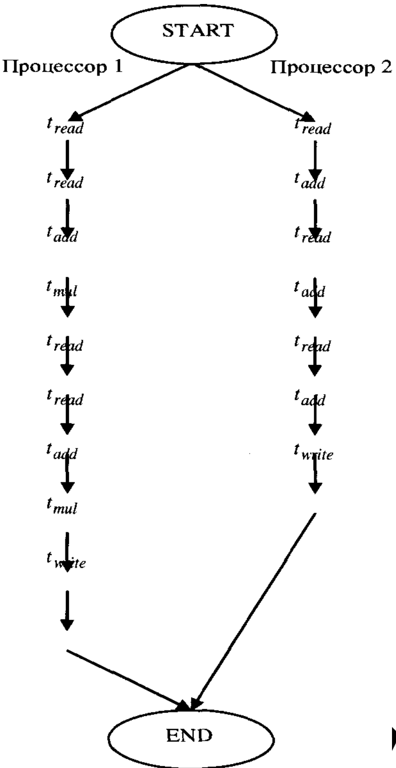
read = use (mem, t_{read})
add = delay (t_{add})
mul = delay (t_{mul})
write = use (mem, t_{write})

Построим граф выполнения данного алгоритма (рис.2).

Очевидно, что в случае, если нет зависимостей по данным, в худшем случае процессор 1 закончит вычисления позднее процессора 2, причем затратит на это время:

$$T = 4t_{read} + 2(t_{add} + t_{mul}) + t_{write} + 3t_{read} + t_{write}.$$

В результате выигрыш за счет второго процессора сводится к тому, что во время некоторых операций чтения/записи второй процессор сможет параллельно выполнять вычисления. Однако при наличии арбитра, который, например, выставит процессору 1 высший приоритет, время вычислений может сократиться за счет осуществления загрузки данных процессором 2 во время выполнения операций сложения и умножения, а не чередуя доступ к шине между процессорами. Тогда, если $t_{add} + t_{mul} \ge t_{load}$, то вычисления будут выполнены за время



р р

ро е о о ес и использо з
мо ость па тной передачи н
то время за ру да
внешне па яти будет меньше

Таким образом, даже в данном простом примере видно, что отсутствие точной модели шины ОМС приводит к существенным искажениям результатов моделирования. Если детализировать модель исследуемой абстрактной ОМС с учетом особенностей работы кэша [4], принимая во внимание тот факт, что обращение к внешней памяти, являющейся ведомым устройством по отношению к процессорам 1 и 2, происходит через шину, то в модели ОМС операции чтения из памяти и записи в память можно представить следующим образом:

а) для кэшируемых сегментов памяти

$$read\_cb(dev, nBytes) =$$
$$read\_cache(nBytes);$$
$$read\_mem\_cb(nBytes);$$
$$read\_bus(dev, nBytes)$$
$$read\_cache(dev, nBytes) = use(cache(dev), \left(\left\lceil \frac{nBytes-1}{I_{dev}^{cache}} \right\rceil + 1\right) \cdot t_{dev}^{cache} \cdot h_{dev}^{RCache})$$
$$read\_mem\_cb(nBytes) = use(MEM, (1-h_{dev}^{RCache}) \times \left(\left\lceil \frac{nBytes-1}{B} \right\rceil + 1\right)(t_{dev}^{cache} + t_{mem}\frac{B}{I_{bus}}))$$
$$read\_bus(dev, nBytes) = use(BUS, (1-h_{dev}^{RCache}) h_{dev}^{RBus} \left(\left\lceil \frac{nBytes-1}{I_{bus}} \right\rceil + 1\right))$$

$$write\_cb(dev, nBytes) =$$
$$write\_cache(nBytes);$$
$$write\_mem\_cb(nBytes);$$
$$write\_bus(dev, nBytes)$$
$$write\_cache(dev, nBytes) = use(cache(dev), \left(\left\lceil \frac{nBytes-1}{I_{dev}^{cache}} \right\rceil + 1\right) \cdot t_{dev}^{cache} \cdot h_{dev}^{WCache})$$
$$write\_mem\_cb(nBytes) = use(MEM, (1-h_{dev}^{WCache}) \times \left(\left\lceil \frac{nBytes-1}{B} \right\rceil + 1\right)(t_{mem} + t_{mem}\frac{B}{I_{bus}}))$$
$$write\_bus(dev, nBytes) = use(BUS,$$
)h B nBytes −

б не уши емых се ментов па яти

e n b ev Bytes
re ytes
e b e nBytes
re e n yte use
tes
b s ev nB tes use
ytes

$write\_ncb(dev, nBytes) =$
$write\_mem\_ncb(nBytes);$
$write\_bus(dev, nBytes)$
$write\_mem\_ncb(nBytes) = use(MEM,$
$(\left[\frac{nBytes-1}{I_{bus}}\right]+1)\cdot t_{mem})$
$write\_bus(dev, nBytes) = use(BUS,$
$h_{dev}^{WBus}\left(\left[\frac{nBytes-1}{I_{bus}}\right]+1\right)),$

где: $nBytes$ – количество передаваемых данных;

dev – ведущее устройство (в данном примере – процессор 1 или 2);

$h_{dev}^{RCache} / h_{dev}^{WCache}$ – вероятность успешного чтения/записи для кэша устройства dev;

I_{dev}^{cache} – разрядность кэша;

B – размер блока кэша;

t_{dev}^{cache} – время чтения единицы информации, разрядностью I_{dev}^{cache}, из кэша;

t_{mem} – время доступа к внешней памяти;

I_{bus} – разрядность шины;

$h_{dev}^{RBus}(n) / h_{dev}^{WBus}(n)$ – средняя задержка при передаче/приеме (по отношению к ведущему устройству) n посылок на шине.

Все величины, кроме de d de - остоянными ля конкретной { С у ествует большое количество особов определения значений 2, 1]. становимся более подробно на определении величин (поскольку, в отличие от методов, описанн в , , , при моделировании с испо ьзованием языка дл аких лементов системы, кэш ил н р яя шина, методы коляю у лят делиро о

$h_{dev}^{RBus}(n) / h_{dev}^{WBus}(n)$.

Перечислим основные характеристики, которые необходимо учитывать при вычислении этих параметров [14]:

а) способ синхронизации (синхронный, асинхронный);

б) способ арбитража [14, 15];

в) частота шины;

г) способ передачи данных по шине (возможность пакетной, разделенной, широковещательной, многоадресной передачи и т.д. [14]).

Важной характеристикой шины ОМС является также разрядность шины данных. Этот параметр уже учтен в выражении $\left[\frac{nBytes-1}{I_{bus}}\right]+1$, которое, как будет показано далее, будет использоваться во всех формулах расчета $h_{dev}^{RBus}(n) / h_{dev}^{WBus}(n)$.

В некоторых случаях можно сразу уменьшить количество параметров, от которых зависит средняя задержка при передаче/приеме на шине. Если в спецификации шины не разрешено одновременно выполнять и запись, и чтение (т.е. когда и запись, и чтение осуществляется по одному каналу), то $h_{dev}^{RBus}(n) = h_{dev}^{WBus}(n)$. Если при передаче данных по шине транзакция может быт прервана в любой момент, то значения г (не зависят от ро е ого, в бо ьшинстве вн ренних шин используется конвейеризация, поэтому задер ками, связанными с функ и онированием арбитр , можно пренебречь

редположим, что в системе, представленной на рис используется метод ар итража *D ii e Divis n u ti e cces*] Во примерах будем с тать, что апис ем осуществляется по од у ка

где k_i - количество пересылок, которое может осуществить i-е устройство за период t_p, то:

$$h_{dev}^{R(W)Bus}(n)=\frac{\left(\left[\frac{n-1}{I_{bus}}\right]+1\right)(t_p-k_{dev}t_{mem})}{k_{dev}}.$$

Данная формула справедлива в том случае, если нет выигрыша в производительности при пакетной передаче. Если же пакетная передача допускается, то формула будет иметь следующий вид:

$$h_{dev}^{R(W)Bus}(n)=\frac{\left(\left[\frac{n-1}{I_{bus}}\right]+1\right)(1-t_{gain})(t_p-k_{dev}t_{mem})}{k_{dev}}$$

где t_{gain} – относительное ускорение при передаче каждой посылки пакетным способом.

В случае, если используется циклический метод арбитража [15], предыдущие формулы будут иметь следующий вид:

а) при отсутствии пакетной передачи:

$$h_{dev}^{R(W)Bus}(n)=\frac{\left(\left[\frac{n-1}{I_{bus}}\right]+1\right)}{k_{dev}}\times$$

$$\times\left(\sum_i(k_i-\overline{k_i^{stall}})-k_{dev}\right)t_{mem}$$

б) при наличии пакетной передачи:

$$h_{dev}^{R(W)Bus}(n)=\frac{\left(\left[\frac{n-1}{I_{bus}}\right]+1\right)}{k_{dev}}\times$$

$$\times(1-t_{gain})\left(\sum_i(k_i-\overline{k_i^{stall}})-k_{dev}\right)t_{mem},$$

где $\overline{k_i^{stall}}$ – среднее время простоя i-го устройства за интервал времени t_p.

Рассмотрим теперь способ арбитража, при котором устройствам не выделяется определенный промежуток времени; вместо этого каждому устройству назначается определенный вес (приоритет) $P_{dev}, \sum_i P_i = 1$. Если передача по шине может быть прервана в любой момент, а пакетная передача отсутствует, то $h_{dev}^{R(W)Bus}$ можно вычислить по формуле:

$$h_{dev}^{R(W)Bus}=\left(\left[\frac{n-1}{I_{bus}}\right]+1\right)(1-P_{dev})t_{mem}.$$

В более сложных случаях необходимо использовать комплексные методы расчета, и при этом возникнет потребность в оценке общего количества вычислений, не требующих обращений к шине. Тогда моделирование с помощью языка *PAMELA* не обходимо выполнять в две итерации. На первой итерации определяется время выполнения вычислений, не требующих обращения к памяти, с целью нахождения значений параметров, которые необходимы для расчета $h_{dev}^{R(W)Bus}(n)$, а на второй итерации проводится подстановка неизвестных параметров в выражения $h_{dev}^{R(W)Bus}(n)$ и вычисляется окончательная оценка времени выполнения параллельного алгоритма.

Выводы

Рассмотрены особенности и предложена методика моделирования внутренних шин ОМС с помощью языка *PAMELA*. Показано, что дальнейшая детализация архитектуры ОМС (описание шины, кэш-памяти) позволяет получать более точные оценки времени выполнения параллельных алгоритмов. Выделены характеристики шин ОМС, которые необходимо учитывать при моделировании, приведены примеры вычисления средней задержки при передаче/приеме на шине. В дальнейшем планируется формализовать метод вычисления средней задержки при передаче/приеме на шине при арбитраже, базирующемся на статическом или динамическом присвоении приоритетов.

Список литературы

1. *Cordan, B.* An Efficient Bug Architecture for System-On-Chip Design // IEEE Custom Integrated Circuit Conference, Jan 2001. – P. 29 – 35.

2. *Ryu K., Shin E., Mooney V.* A Comparison of Five Different Multiprocessor SoC Bus Architectures // Proc. of the EUROMICRO Symposium on Digital Sys-

tems Design. – Washington (USA), 2001. – P. 202 – 209.

3. *Germund A.* Symbolic Performance Modeling of Parallel Systems. IEEE Transactions on Parallel and Distributed systems, 2003 – V.14. – №2. – P. 154 – 165.

4. *Марченко А.И., Богуславский О.В.* Аналитическая оценка времени выполнения параллельных алгоритмов на однокристальных многопроцессорных системах // Вісник Хмельницького національного університету. – Хмельницький, 2007 – №2. – Т.2. – С. 17 – 21.

5. *Воеводин В.В., Филамофитский М.П.* Анализ динамических характеристик параллельных программ // Высокопроизводительные вычисления и их приложения: Тр. Всерос. науч. конф. 30.10-02.11.2000г. – М.: 2000. – С. 203 – 207.

6. *Ariyamparambath, M, et al.* A highly efficient modeling style for heterogeneous bus architectures // Proceedings of International Symposium on System-on-Chip - Tampere, 2003. – P. 83 – 87.

7. *Madl G. et al.* Formal Performance Evaluation of AMBA-based System-on-Chip Designs // Proc. of EMSOFT. – Seoul (Korea), 2006. – P. 311 – 320.

8. *Goel, A., Lee, W.R.* Formal verification of an IBM CoreConnect processor local bus arbiter care // Proceedings of 37th Design Automation Conference. – Los-Angeles, 2000. – P. 196 – 200.

9. AMBA Specification. Revision 2.0 // http://www.arm.com/products/solutions/AMBA_Spec.htm

10. *IBM.* The CoreConnect Bus Architecture. http://www.chips.ibm.com/product/coreconnect/docs/crcon_wp.pdf

11. *A. Cimatti et al.* NuSMV 2: An OpenSource Tool for Symbolic Model Checking // Proceedings of the 14th International Conference on Computer-Aided Verification. – Copenhagen, 2002. – P. 359 –364.

12. *Sebek F.* The State Of Art in Cache Memories and Real-Time Systems. MRTC Technical Report 01/37, Sept 2001. // http://www.idt.mdh.se/fsk/docs/sota.pdf

13. *Сигнаевский В.А., Коган Я.А.* Методы оценки быстродействия вычислительных систем. – М.: Наука, 1991. – 256 с.

14. *Марченко А.И., Богуславский О.В.* Обзор возможностей внутренних шин однокристальных многопроцессорных систем // Современные телекоммуникационные и информационные технологии: Материалы научно-практического семинара, УНИИС, 25-26 декабря 2006 г. – К.: 2006. – С. 58 – 64.

15. *Jerraya A., Wolf W.* Multiprocessor Systems-on-Chips. – San Francisco (USA), 2004. – 582 p.