

УДК 681.3

АРХИТЕКТУРА МАРШРУТИЗАТОРА ANTNNM ДЛЯ СЕТИ НА КРИСТАЛЛЕ

В.А. Мирецкая, Ю.В. Ладыженский
Донецкий национальный технический университет

Представлена архитектура маршрутизатора для комбинированного алгоритма маршрутизации AntNNM в сетях на кристалле. Предложены новая схема передачи и формат флитов пакета маршрутизации в алгоритме AntNNM.

Введение

Потребность в создании высокопроизводительных, гибких, масштабируемых, программируемых микропроцессорных схем и платформ для многократного использования побудила к разработке сетей на кристалле (NoC) [1].

Актуальной является задача разработки эффективных алгоритмов маршрутизации для сетей на кристалле. Описана новая архитектура маршрутизатора для сети на кристалле, использующая новый алгоритм AntNNM [1], основанный на муравьином подходе к решению задачи маршрутизации, и являющийся развитием алгоритма AntNet [2, 3]. Предложены новый формат и схема отправки флитов [1] F-муравья, позволяющая избавиться от буферов сборки муравьев в маршрутизаторе сети.

Алгоритм AntNNM является комбинированным и позволяет находить маршруты в обход загруженных участков сети при помощи специальных пакетов-муравьев, исследующих состояние сети. В незагруженном состоянии сети, все маршруты статичны и не изменяются. Муравьи начинают поиск маршрутов только при обнаружении маршрутизаторами загруженных участков.

Использование комбинированной схемы маршрутизации, по сравнению с предложенной в [3], позволяет снизить среднюю задержку пакетов в маршрутизаторах, т.к. в неадаптивном режиме маршрутизаторы не исследуют сеть. В адаптивном режиме маршрутизаторы ведут поиск маршрутов в обход загруженных участков и, тем самым, не допускают значительного снижения пропускной способности сети.

Архитектура маршрутизатора алгоритма AntNNM

Схема маршрутизатора для алгоритма AntNNM основана на схеме VCR-маршрутизатора с мультиплексированием после вхождения в очередь [4]. Архитектура AntNNM маршрутизатора

представлена на рис. 1. В каждом входном порте выполняется разделение всех флитов по двум очередям: очередь пакетов маршрутизации и очередь данных. Флиты из этих входных портов обрабатываются и маршрутизируются параллельно.

Представленная архитектура реализует маршрутизацию с коммутацией каналов [1]. Для этого в портах имеются специальные регистры: CRT – номер выходного порта, в который должен быть отправлен флит; CSIP – текущий обслуживаемый входной порт и NBS – состояние соседнего входного порта.

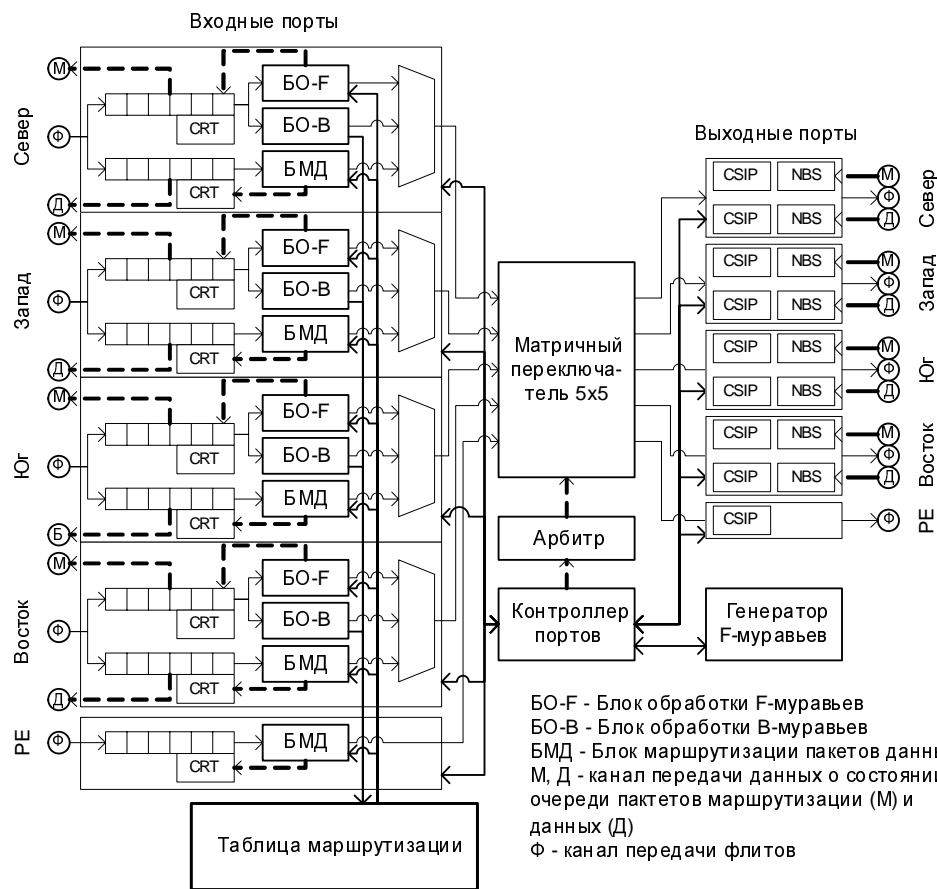


Рис. 1. Архитектура маршрутизатора AntNNM

Муравьи обрабатываются блоками БО-F и БО-V. Решение о том, куда направлять очередной флит, принимается на основе содержимого флита, либо на основе таблицы маршрутизации и состояния выходных портов маршрутизатора.

Обычные пакеты данных обрабатываются блоком БМД. Этот блок читает содержимое таблицы маршрутизации и меняет значение CRT. Контроллер портов опрашивает содержимое регистров CRT, CSIP и NBS и принимает решение о том, из каких входных портов можно отправлять флиты и в каком направлении. Муравьи имеют

наивысший приоритет и всегда обслуживаются первыми. На основании своего решения, контроллер отправляет запрос к арбитру, который меняет состояние матричного переключателя. При помощи контроллера выполняется поиск заблокированных выходных портов и генерация новых F-муравьев.

Схема отправки и формат пакетов

В алгоритме AntNNM используются два типа пакетов маршрутизации (муравьев) [1]: F-муравей и В-муравей. В [1] при достижении F-муравьем узла назначения, происходила сборка всех его флитов в отдельном буфере и создание из него В-муравья с последующей отправкой этих флитов в обратном порядке для обновления таблиц маршрутизации. Сборка пакетов маршрутизации в буфере приводит к усложнению аппаратной реализации маршрутизатора, так как появляется необходимость в расширении ее дополнительными регистрами и логическими схемами, управляющими регистрами. Для того чтобы избавиться от дополнительных буферов в маршрутизаторах, разработана новая схема передачи муравьев, в которой взамен буферов пакет муравья расширяется одним дополнительным флитом, дублирующим информацию о целевом узле F-муравья. Новый формат пакета маршрутизации (типы флитов, на которые разбивается пакет) представлен в таблице 1.

Таблица 1 - Формат пакета маршрутизации алгоритма AntNNM

Флит	Описание	Используется муравьем
DSTINFO	Информация об узле назначения	F-муравей
STACK	Стек муравья	F-муравей, В-муравей
BANTINFO	Информация о состоянии предыдущего узла	В-муравей

Подробное описание полей флитов DSTINFO, STACK, BANTINFO представлено в таблицах 2,3,4 соответственно.

Таблица 2 - Флит DSTINFO

Поле	Занимаемые биты	Описание
ANTTYPE	1 бит	0 – F-муравей; 1 – В-муравей
DSTID	16 бит	Идентификатор узла назначения
RESERVED	15 бит	Зарезервированные биты

Таблица 3 - Флит STACK

Поле	Занимаемые биты	Описание
ANTTYPE	1 бит	0 – F-муравей; 1 – В-муравей
RESERVED	1 бит	Зарезервированный бит
INDEX	4 бит	Количество заполненных ячеек стека (значения – от 0 до 13)
CELLS	26 бит (13 по 2 бит)	Ячейки стека

Таблица 4 - Флит BANTINFO

Поле	Занимаемые биты	Описание
ANTTYPE	1 бит	0 – F-муравей; 1 – В-муравей
DSTID	16 бит	Идентификатор узла назначения
CS	2 бит	CS – статус перегрузки предыдущего узла
RESERVED	13 бит	Зарезервированные биты

Рассмотрим более подробно порядок отправки флитов муравья:

1. Из узла источника в направлении узла назначения отправляются по порядку 3 флита: DSTINFO, STACK, BANTINFO.
2. На основе информации из DSTINFO узлы вычисляют направление перемещения муравья. Обратное направление записывается в STACK (записывается в ячейку с индексом INDEX, а само поле INDEX увеличивается на 1). Если в STACK закончилось место, то создается новый пустой флит STACK, с которым выполняется та же операция.
3. При достижении флитом DSTINFO целевого узла, флит уничтожается, во входном порте устанавливается флаг, показывающий необходимость превращения всех последующих флитов пакета в В-муравья.
4. При получении флита STACK, из него вытаскивается направление перемещения (в узле-назначения – это обратное направление, т.е. если муравей пришел с севера, то на север и будет отправлен). Если флит STACK стал пустым, то он удаляется, не дожидаясь отправки.
5. При получении флита BANTINFO: если идентификатор маршрутизатора \neq DSTID, то флаг CS используется для обновления таблицы маршрутизации. Затем для любого DSTID в флаг CS помещается информация о состоянии маршрутизатора.
6. Если в муравье остается только один флит BANTINFO, т.е. все флиты STACK уничтожены другими маршрутизаторами, то считается,

что муравей достиг своего узла назначения. Флит BANTINFO используется для обновления таблицы маршрутизации, после чего уничтожается.

Заключение

Разработанная архитектура маршрутизатора и реализуемый ею алгоритм маршрутизации позволяют выявлять загруженные участки сети и своевременно находить новые маршруты, идущие в обход этих загруженных участков, и распространять информацию о них по сети. Это выполняется за счет того, что каждый маршрутизатор следит за состоянием своих каналов и занимается поиском обходных маршрутов, в случае обнаружения блокировок, путем отправки муравьев по менее загруженным каналам. Муравьи двигаются по приоритетным каналам, что обеспечивает быструю доставку новой маршрутной информации.

Новый формат и схема отправки муравья позволяет избавиться от дополнительных буферов в маршрутизаторе, которые усложняли аппаратную реализацию алгоритма AntNNM.

Литература

1. Ладьженский Ю.В., Мирецкая В.А. Моделирование алгоритмов маршрутизации в сетях на кристалле. Научные труды Донецкого национального технического университета. Серия «Информатика, кибернетика и вычислительная техника» (ИКВТ-2008). Выпуск 9 (132) – Донецк: ДонНТУ. – 2008. – с. 79-86.
2. Di Caro G., Dorigo M. AntNet: Distributed Stigmergetic Control for Communications Networks: Journal of Artificial Intelligence Research. 1998. №9. p. 317-365.
3. Daneshtalab M., Sobhani A., Mottaghi M. D., Kusha A.A., Navabi Z., Fatemi O. Ant Colony Based Routing Architecture for Minimizing Hot Spots in NOCs // Proceedings of the 19th annual symposium on Integrated circuits and systems design. 2006. - p. 56-61.
4. Kavaljdjev N., Smit G.J.M., Jansen P.G.: A Virtual Channel Router for Onchip Networks. Proceedings, IEEE International SOC Conference, 12–15 September 2004. – p. 289–293.

Получено 29.05.2009