УДК 004.318

Ю. А. Недбайло

ЗАО «МЦСТ», ПАО «ИНЭУМ им. И. С. Брука»

Разработка сети на кристалле для перспективных многоядерных микропроцессоров

Описана проблема эффективной организации соединений на кристалле многоядерного микропроцессора с распределённым общим кэшем. От сети соединений требуются высокая частота передачи пакетов, низкие задержки, поддержка качества обслуживания и сохранение порядка пакетов. Предложено решение на основе топологии сетка, деления сети на строки и столбцы и экспресс-виртуальных каналов с доработками их механизма предотвращения «голодания». Для размеров сети от 4×4 до 16×16 оценены производительность и размер роутеров предложенной схемы и нескольких традиционных. Только предложенная схема продемонстрировала все требуемые качества в сочетании с хорошей масштабируемостью.

Ключевые слова: сеть на кристалле, Эльбрус, архитектура, многоядерность, подсистема памяти, общий кэш.

Yu. A. Nedbailo

JSC «MCST», PJSC «Brook INEUM»

On-chip network design for prospective chip multiprocessors

The paper is dedicated to the problem of the efficient on-chip interconnection of a high performance many-core microprocessor with distributed shared cache. Packet injection rate, latency, Quality of Service and in-order delivery are the main concerns. A solution is proposed employing mesh topology, row-column decoupling, and Express Virtual Channels with starvation avoidance mechanism improvements. Throughput, fairness, latency and silicon area of the proposed scheme and a few traditional ones are evaluated for network sizes from 4×4 to 16×16. The proposed scheme is the only one to demonstrate all of the required qualities in conjunction with good scalability.

Key words: on-chip interconnect, NoC, Elbrus, architecture, many-core, memory subsystem, shared cache, NUCA.

1. Введение

Развитие полупроводниковой технологии позволяет размещать всё большее количество процессорных ядер на одном кристалле. Уже выпускаются микропроцессоры с десятками и сотнями ядер, появляются модели с тысячами. Как правило, они включают в себя распределённый общий кэш для уменьшения нагрузки на внешнюю память, которая развивается не так быстро. Одной из главных проблем реализации таких процессоров является эффективная организация соединений между ядрами и банками общего кэша (рис. 1).

Чтобы не стать узким местом производительности процессора в каких-либо задачах, сеть соединений должна обладать рядом качеств:

- **достаточная пропускная способность** для одновременных обменов каждого ядра со всеми банками кэша;
- **минимальные задержки** между входом пакета в сеть и выходом, чтобы время доступа в кэш оставалось достаточно низким;

- **качество обслуживания** абонентов сети, то есть ядер, должно удерживаться в некоторых рамках, чтобы работа одних ядер не вызывала простой других;
- **соблюдение порядка пакетов** между каждой парой узлов (для тех типов пакетов, которые этого требуют), иначе некоторые транзакции приходится задерживать до завершения предыдущих.

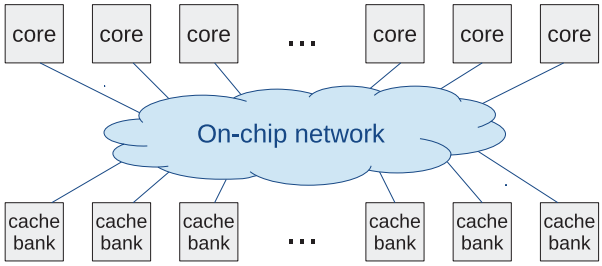

Рис. 1. Концепция распределённого общего кэша

Пропускная способность сети в основном определяется её топологией – порядком, в котором соединены каналами её узлы (роутеры), – и шириной этих каналов. Соблюдение порядка пакетов достигается применением детерминированной маршрутизации, то есть использованием только одного маршрута между каждой парой узлов, и FIFO-реализацией всех буферов. Сложными проблемами пока остаются минимизация задержек и реализация качества обслуживания, причём задержки становятся проблемой всё более острой [1].

В данной статье предлагается архитектура сети, названная FEMIDA[1], сочетающая перечисленные качества и хорошо масштабируемая как минимум до сотен узлов. Она, в частности, рассчитана на применение в процессорах серии «Эльбрус» с 16–64 ядрами.

2. Существующие методы

Из всего многообразия методов построения сетей на кристалле наиболее интересны те, которые хорошо масштабируются и не требуют избыточного количества проводов или специальных возможностей физического проектирования, таких как оптические шины или трёхмерная компоновка.

Традиционный подход заключается в равномерном распределении ядер, банков кэша и сетевых роутеров по кристаллу и присоединении одного или нескольких ядер и одного или нескольких банков кэша к каждому роутеру. В этом случае задача сужается до выбора топологии сети (рис. 2) и внутреннего устройства роутеров.

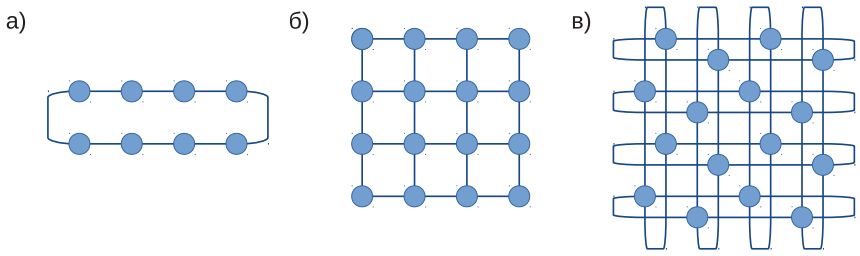

Рис. 2. Топологии: а) кольцо, б) сетка, в) сложенное двумерное кольцо

Самыми распространёнными топологиями являются кольцо (ring, или 1D torus) и двумерная сетка (2D mesh). Менее известным, но тоже хорошо подходящим для современных чипов вариантом является сложенное двумерное кольцо (folded 2D torus). Наилучшей масштабируемостью и наименьшей длиной путей обладает двумерная сетка. [2]

Внутреннее устройство роутеров будет обсуждаться в разделе 3.

[1] **F**air **E**VC-based **M**esh network with **I**n-order **D**elivery of **A**ll packets.

2.1. Методы минимизации задержек

Простейший вариант реализации сети на кристалле, ориентированный на хорошую масштабируемость и минимальные задержки, – плиточная (tiled) организация – заключается в использовании топологии двумерная сетка с подключением одного ядра и одного банка кэша к каждому роутеру (рис. 3а). Однако, если учитывать, что прохождение пакета через каждый роутер включает задержки на его маршрутизацию, арбитраж и прочее, предпочтительнее может оказаться сеть с более крупным шагом и несколькими ядрами и банками на один роутер (рис. 3б). В любом случае задержки получаются существенно выше идеальных – таких, как если бы все абоненты сети были соединены напрямую.

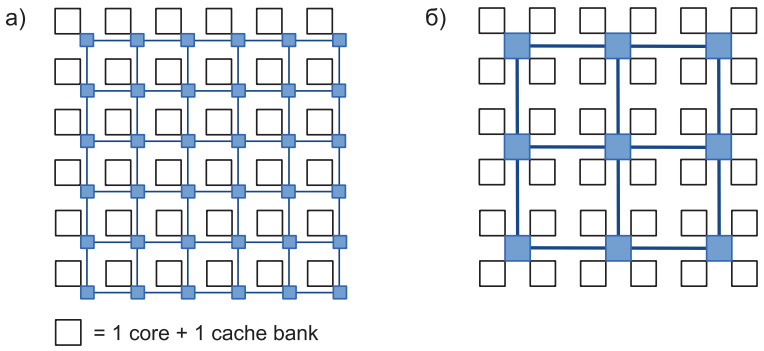

Рис. 3. Организация сети: а) плиточная, б) концентрированная

В борьбе за приближение задержек к идеальным можно выделить два подхода. Первый заключается в оптимизации всех стадий конвейера обработки пакета для либо их ускорения, либо выполнения параллельно друг с другом. При этом в каждом роутере всё равно будет некоторая задержка и может увеличиться энергопотребление.

Второй подход заключается в реализации экспресс-каналов или их производных. Экспресс-каналы – дополнительные каналы, соединяющие удалённые узлы. Их использование уменьшает среднее количество роутеров, посещаемых пакетами, и таким образом суммарные задержки в роутерах. В больших сетях они могут быть объединены в дополнительную крупную сетку (рис. 4а) [2].

Производной от обычных экспресс-каналов является MECS – многоточечные (multidrop) экспресс-каналы, проходящие мимо нескольких узлов так, что пакет может передаваться из первого узла в любой другой (рис. 4б). Нескольких таких каналов на каждую строку и столбец сети достаточно, чтобы все пакеты передавались только по ним и максимум в два шага, что приближает задержки к минимально возможным [3].

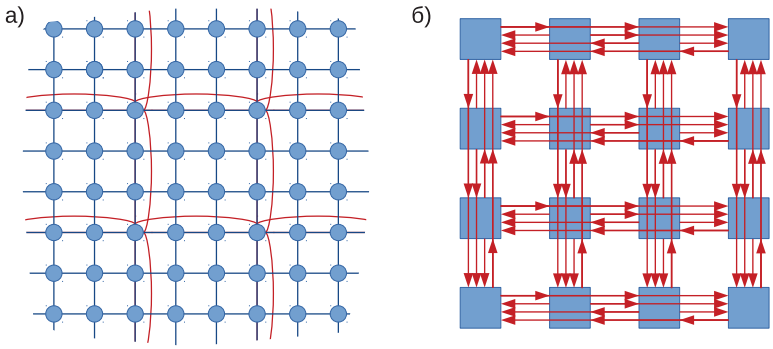

Рис. 4. Экспресс-каналы: а) обычные, б) MECS

Экспресс-каналы в упомянутом виде подразумевают прокладку на кристалле дополнительных проводов и усложнение структуры роутеров, что дорого. Избавлены от этих проблем экспресс-виртуальные каналы (EVC). Если цепочку обычных каналов соединить бай-

пасами (например, из левого входа роутера в правый выход), включаемыми специальным заранее посылаемым сигналом, эта цепочка будет работать почти как обычный экспресс-канал (рис. 5). Если соединить такими цепочками попарно все узлы в каждой строке и каждом столбце сети, все пакеты можно будет передавать максимум в два логических шага, как и при использовании MECS, хотя физически сеть останется обычной двумерной сеткой и роутеры усложнятся незначительно [4].

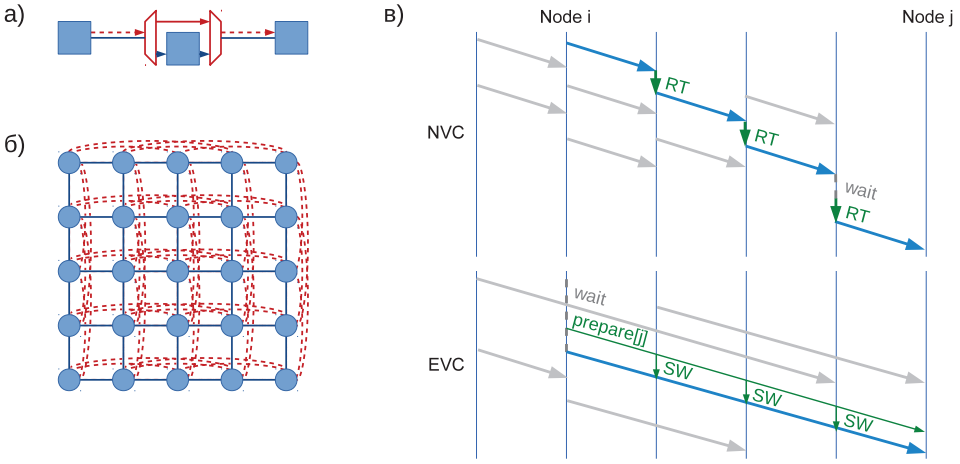

Рис. 5. Экспресс-виртуальные каналы: а) принцип, б) сеть с ними, в) пространственно-временная диаграмма передачи пакета в сравнении с обычными (NVC) каналами

В данной работе мы используем экспресс-виртуальные каналы.

2.2. Методы реализации качества обслуживания

Применительно к сетям качество обслуживания может означать следующее:

- **гарантированный сервис** – определённую пропускную способность или/и задержки для конкретных абонентов;
- **приоритеты** между абонентами в случае нехватки пропускной способности на всех;
- **честность** в распределении пропускной способности при её нехватке;
- некоторые более специфические вещи [2].

Первые две разновидности бывают нужны задачам реального времени и критически важным процессам. Обычным задачам важна честность сети. Программист вправе ожидать, что одинаковые участки кода, запущенные на разных ядрах, выполнятся за примерно одинаковое время. Если это будет нарушаться, распределение и синхронизация работы между тредами могут быть сопряжены со значительной долей простоев некоторых ядер, соответственно общая производительность снизится. Кроме того, когда одновременно выполняются задачи с разной интенсивностью обменов с памятью, суммарная производительность будет выше, если приоритет будет отдаваться менее интенсивным.

Такая честность, или оптимальность распределения пропускной способности, обычно описывается понятием max-min fairness. В простых случаях её можно реализовать Round Robin арбитрами. Для сложной сети традиционное решение – Fair Queuing – предполагает перед каждым выходом сети отдельные очереди для каждого потока и арбитр между ними (рис. 6). К сожалению, этот вариант плохо масштабируется.

Одним из подходов к реализации честности сети является динамическое определение приоритетов между потоками и учёт их арбитрами каждого роутера. Так, схема Preemptive Virtual Clock (PVC) [5] включает счётчики пакетов для каждого потока в каждом роутере

и использует их в качестве приоритетов. Она используется даже в процессорах с 1000 ядрами [3] и потому представляет основной интерес. Этот подход, как мы увидим далее, не вполне сочетается с нашими целями, в первую очередь с масштабируемостью.

Другим подходом может быть Source Throttling – ограничение темпа приёма в сеть пакетов от тех абонентов, которые, согласно собранной статистике, используют сеть слишком активно. На эту тему было несколько публикаций [6], но в них нет деталей, необходимых для применения метода к нечестной сети.

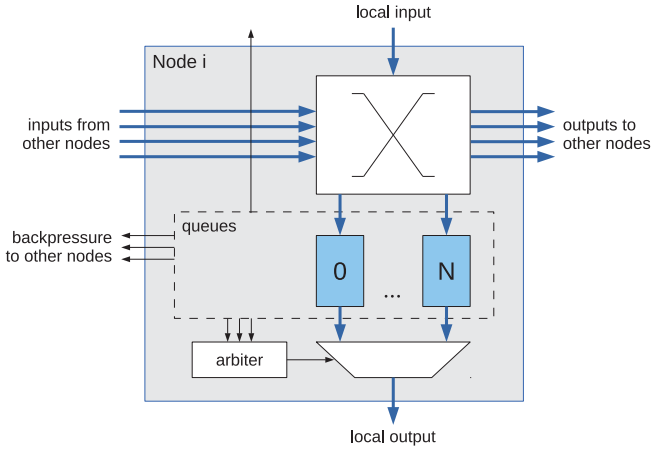

Рис. 6. Fair Queuing в роутере сети на кристалле

3. Предлагаемая схема

Итак, мы считаем оптимальной сеть топологии двумерная сетка, к каждому роутеру которой подключены одно или несколько ядер и один или несколько банков общего кэша, и хотим, чтобы она обладала качествами, перечисленными в разделе 1.

Остаётся продумать внутреннее устройство роутеров.

3.1. Общая компоновка

Традиционное устройство роутера подразумевает FIFO-буфер на каждом из пяти входов (от соседних роутеров плюс один от абонентов), коммутатор 5×5 между выходами этих буферов и выходами роутера и управляющую логику (рис. 7а). В рассматриваевом случае при соблюдении порядка пакетов и соответственно детерминированной маршрутизации логика сводится к отдельному арбитру у каждого выхода, выбирающему пакет из пяти буферов, и тривиальным схемам маршрутизации и управления потоком. Схема качества обслуживания Preemptive Virtual Clock добавляет к каждому выходу счётчики пакетов, учитываемые арбитрами. Экспресс-виртуальные каналы добавляют байпасы между противоположными каналами, почти не меняя внутреннее устройство роутера.

Как покажут в разделе 4 наши эксперименты – конфигурации NVC-Q, EVC-Q и EVC-QI, – такая схема, во-первых, не вполне справляется с задачей честного распределения пропускной способности между абонентами. Проблема кроется в использовании разными потоками общих буферов, что не позволяет арбитрам тормозить одни потоки, не тормозя другие. Исходный вариант схемы PVC включает вытеснение и переповтор низкоприоритетных пакетов для решения этой проблемы, но это нарушает порядок пакетов и потому нам не подходит; в конфигурации EVC-QI мы не используем этот механизм. Во-вторых, эта схема не очень хорошо масштабируется. Хотя счётчики PVC требуют гораздо меньше площади, чем буферы Fair Queuing, порядок роста у этой схемы такой же – пропорциональный числу абонентов сети. В упомянутом 1000-ядерном процессоре эта проблема смягчается объединением абонентов по четыре ядра на роутер, что не всегда оптимально.

Мы предлагаем другой подход к реализации роутера, а именно:

- разделить роутер на X и Y части (рис. 7б), то есть сеть – на строки и столбцы;
- в каждой части сделать общий буфер с Fair Queuing в пределах строки/столбца;
- доработать механизм предотвращения «голодания» экспресс-виртуальных каналов.

При правильной реализации это позволяет добиться требуемых качеств и масштабируемости порядка квадратного корня из количества абонентов.

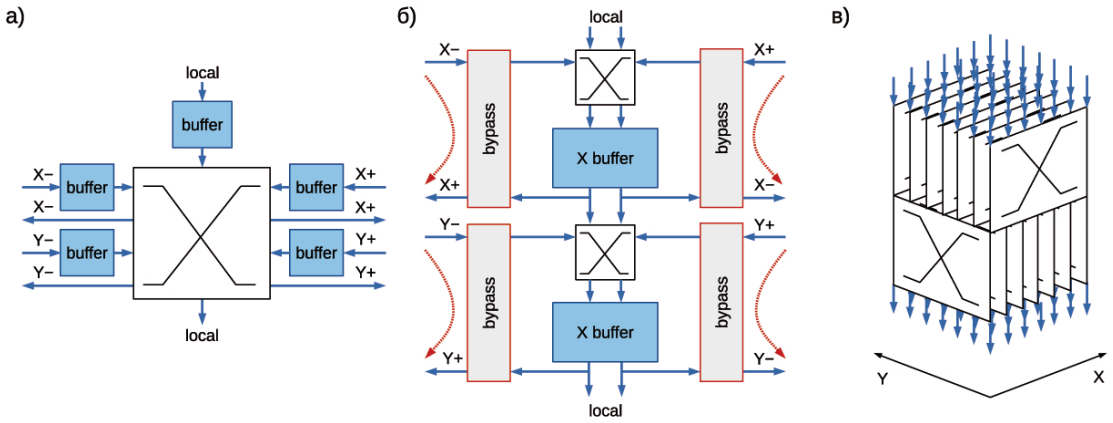

Рис. 7. Структура роутеров: а) традиционная, б) предлагаемая, в) предлагаемая иерархия сети

Идея разделить роутер на две части не новая и в основном применяется для упрощения его логики [7]. Маршрутизация пакетов сначала по X координате и потом по Y часто применяется в сетях как дешёвое средство предотвращения дедлоков [2], но нам оно также позволяет удешевить реализацию качества обслуживания, поскольку на каждом из двух полученных таким образом уровней иерархии сети, по сути распределённых коммутаторов (рис. 7в), будет только корень от общего числа узлов.

Реализация буферов и экспресс-виртуальных каналов будет обсуждаться далее.

3.2. Организация буферов

Рассмотрим одну половину роутера. У неё два входа от соседних роутеров, с которых нужно уметь одновременно принимать пакеты, и локальный вход или входы, приём с которых можно блокировать.

Традиционный подход с отдельным буфером на каждом входе в нашем случае не совсем оптимален. Fair Queuing подразумевает отдельную очередь для каждого абонента, а для узлов, близких к краю сети, большинство абонентов расположены по одну сторону от роутера. Если делать все роутеры одинаковыми и выделить каждому абоненту одинаковое место в буферах, при таком подходе потребуется двукратный запас места.

Более логичное решение – общий буфер с двумя входами и коммутатором (рис. 7б). Такой буфер проще всего сделать на основе двухпортовой статической памяти (DPSRAM). Хотя двухпортовые памяти имеют примерно вдвое меньшую плотность по сравнению с однопортовыми, поэтому выигрыша площади такое решение не даст, их использование позволит несколько повысить производительность, поскольку оба порта памяти можно использовать для выдачи любых пакетов, когда в буфер не производятся записи. При размерах сети меньше 8x8, однако, очереди с локальных входов нужно делать отдельными, иначе двухпортовая память ограничит пропускную способность сети[2], поскольку пакет в нашей схеме проходит буферы четыре раза. Вообще же общий буфер можно собрать из нескольких однопортовых памятей, и тогда у такой схемы будут все преимущества [8].

[2]Максимальная ПС сетей 4x4, 8x8 и 16x16 при равномерном трафике – 16, 32 и 64 передачи за такт.

Важным вопросом является реализация очередей. Для соблюдения порядка пакетов между каждой парой абонентов буфер должен хранить порядок прихода пакетов от каждого источника, то есть каждая очередь Fair Queuing должна быть FIFO.

Мы видим два подходящих варианта такого хранения порядка. Первый, используемый в конфигурациях EVC-PF и FEMIDA-F в разделе 4, подразумевает FIFO-очереди с указателями на буфер (рис. 8а). То есть вся информация, хранимая в буфере, кроме самих пакетов, статически разделена между источниками. У этого варианта есть два недостатка. Во-первых, статическое разбиение буфера не всегда оптимально, особенно когда от одного источника могут приходить пакеты разных типов, которые тоже нужно хранить раздельно из-за разного их приоритета или для предотвращения протокольных дедлоков[3]. А для динамического разбиения потребуется увеличить размеры очередей. Во-вторых, в одной очереди могут храниться пакеты, предназначенные разным получателям, но доступен для выборки всегда только самый старый, что снижает пропускную способность сети.

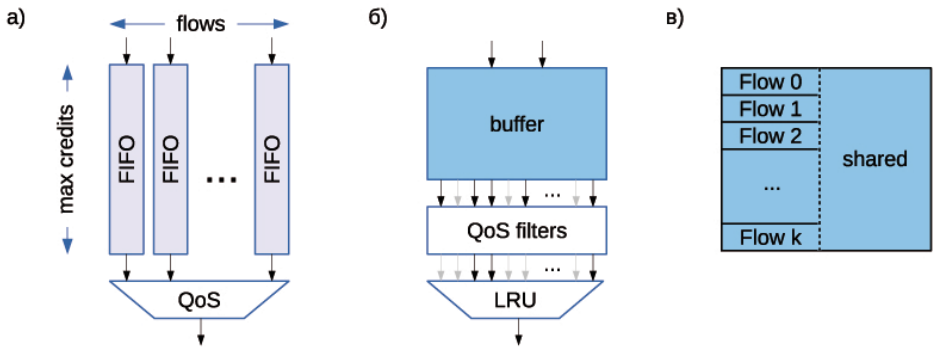

Рис. 8. Организация буфера: а) FIFO-очереди, б) фильтры + LRU-арбитр, в) разбиение

Более гибкое и производительное решение (FEMIDA-L) – отказ от FIFO-очередей и соблюдение порядка при помощи LRU-арбитров (рис. 8б). Качество обслуживания тогда можно реализовать фильтрами, выбирающими пакеты одной или нескольких пар источник–получатель, а LRU-арбитр выберет из них самый старый. Простейший фильтр содержит Round Robin арбитр, выбирающий один поток среди тех, чьи пакеты есть в буфере, и подсвечивает все пакеты этого потока. Такой фильтр нужен для честности сети. Похожий фильтр может выбирать пакеты с самым высоким приоритетом. Достаточно быстрый и хорошо масштабируемый LRU-арбитр можно сделать на основе [9]. Учитывая, что пакеты от других узлов идут только в локальный выход, арбитры на выходах в EVC-байпасы можно сделать «видящими» только часть буфера, используемую локальным входом.

Последний серьёзный вопрос при реализации буфера: как распределять его объём между абонентами? С одной стороны, Fair Queuing требует статически выделить некоторое место каждому абоненту. С другой стороны, при сплошном потоке пакетов между двумя абонентами через экспресс-виртуальный канал одному абоненту может потребоваться столько места, сколько нужно для покрытия задержки до второго, то есть пропорционально расстоянию между ними, иначе пропускная способность сети будет снижена. Следовательно, статическое разбиение буферов, рассчитанное на любые такие потоки пакетов, плохо масштабируется. Подходящим решением будет распределить, например, половину места статически, а остальную часть сделать общей (рис. 8в). Необходимый размер буфера – обеих частей – в этом случае будет пропорционален количеству абонентов в строке или столбце, то есть квадратному корню от общего их количества, как мы и хотели.

[3]Пример протокольного дедлока: сеть заполнена запросами, потому что кэш не может принять запрос от ядра, потому что не может выполнить уже принятые запросы, потому что не может передать в ядро свой, потому что сеть заполнена. Запросы от ядер и от кэша лучше хранить в отдельных очередях.

3.3. Реализация экспресс-виртуальных каналов

Чтобы пакет мог проходить по обычным каналам, не проходя через конвейер роутера, между каналами и конвейером можно поставить мультиплексоры, как показано на рис. 9а. Если переключать их заранее, они почти не добавляют задержек, и пакет передаётся как по обычному экспресс-каналу. Сигнал переключения можно формировать сравнением координат роутера с адресом назначения пакета, передаваемым на такт раньше. Такая схема проста в реализации и не ограничивает длину экспресс-виртуальных каналов. Передача адреса и значимости пакета, конечно, включает некоторую задержку, но это всего лишь несколько сигналов, которые можно ускорить просто сделав более мощными.

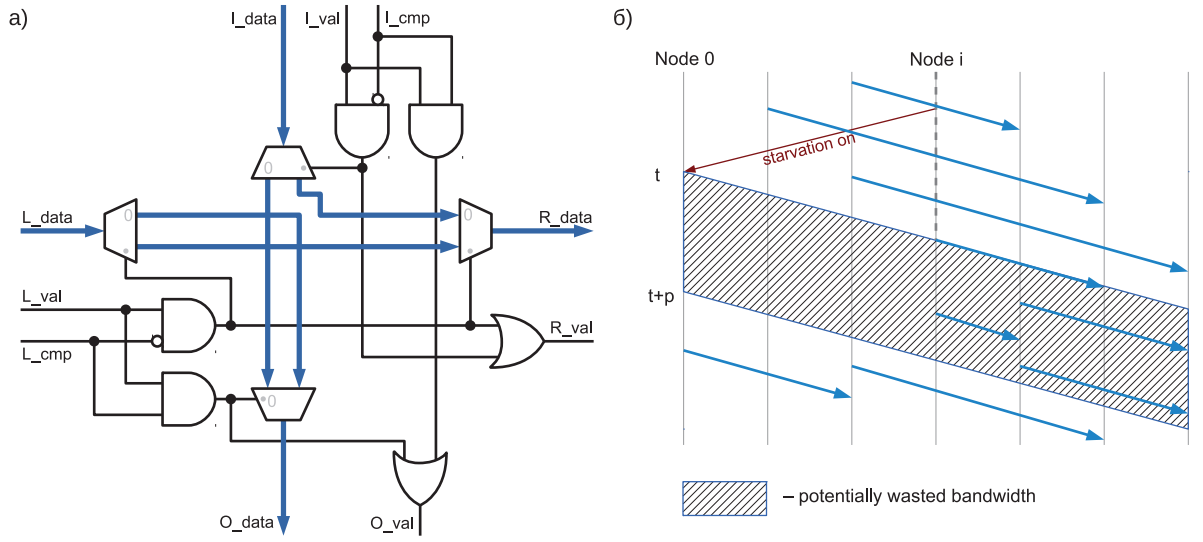

Рис. 9. Экспресс-виртуальные каналы: а) схема байпаса, б) сообщение о «голодании»

Первоначально длину экспресс-виртуальных каналов предлагалось делать небольшой [4], потому что иначе понадобятся большие буферы, что мы обсуждали выше. Но в нашем случае – при реализации Fair Queuing – требуемый размер буферов пропорционален количеству узлов в измерении независимо от длины экспресс-виртуальных каналов, поэтому её ограничение имеет мало смысла.

Для минимизации задержек пакет лучше всего передавать вдоль строки или столбца без остановок. Однако это создаёт проблему при интенсивном трафике. Поскольку обсуждаемая механика не предполагает остановки пакета на полпути, крайние узлы-источники получают приоритет над остальными, из-за чего честность сети может нарушаться. Изначальная схема [4] в такой ситуации предполагает использование специальных сообщений о «голодании» (starvation token). Если роутер в течение определённого времени не может отправить пакет из-за потока пакетов от других узлов сети, он посылает им это сообщение, и они на некоторое время перестают выдавать свои пакеты (рис. 9б).

Указанная схема предотвращения «голодания» гарантирует прогресс каждому потоку, но полноценную честность сети обеспечивать не предназначена. Кроме того, если её настроить на частое срабатывание, она снизит пропускную способность сети, а в противном случае её влияние на честность сети незначительно. Мы внесли в эту схему две доработки, устраняющие эти недостатки. FEMIDA-F и FEMIDA-L в разделе 4 их содержат.

Первая доработка заключается в ослаблении эффекта от сообщения о «голодании»: теперь оно освобождает определённый канал сети на один такт. Когда оно доходит до некоторого узла (в нашей реализации – крайнего), то превращается в другое сообщение, параллельно с которым передаются только те пакеты, которые не достигнут исходного узла (рис. 10а). Таким образом уменьшаются потери пропускной способности.

Вторая доработка – реализация честности в использовании этих сообщений. Узлы периодически обмениваются статистикой, сколько пакетов им удалось отправить за послед-

ний интервал времени. Собранные значения суммируются с несколькими предыдущими интервалами (в экспериментах мы используем один), и на основе этих чисел оценивается количество пакетов, которые каждый узел может отправить в следующем интервале, чтобы суммарное число пакетов выровнялось (рис. 10б). Эту оценку для себя каждый узел использует для ограничения количества отправляемых сообщений о «голодании».

Полученный механизм можно считать разновидностью Source Throttling.

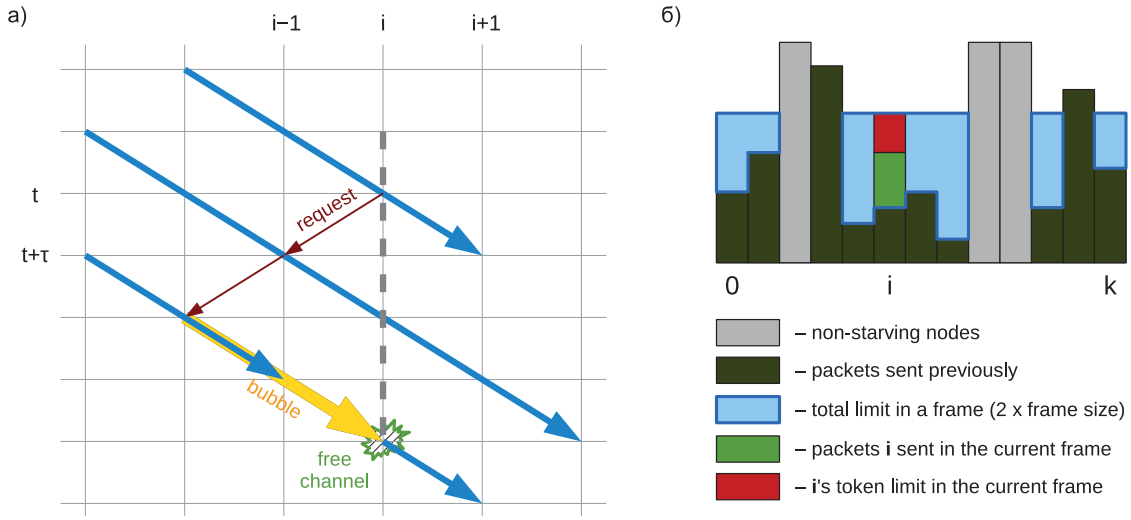

Рис. 10. Доработки ЭВК: а) новое сообщение о «голодании», б) ограничение частоты их отправки

4. Результаты экспериментов

Несколько вариантов реализации роутеров, приведённые в табл. 1, были описаны на языке Verilog и протестированы при размерах сети от 4×4 до 16×16. С помощью ПО Synopsys® и 28 нм технологических библиотек получены оценки их площади.

Таблица 1

Конфигурации роутеров, используемые в экспериментах

Название	Длина EVC	Сохранение порядка	Глубина буферов	Длина конвейера	Описание
NVC	–	+	8	1	Простейший традиционный роутер
NVC-Q	–	–	8	1	–//– с Preemptive Virtual Clock [5]
EVC	3	–	32	2	Традиционный EVC-роутер [4]
EVC-Q	3	–	32	2	–//– с Preemptive Virtual Clock
EVC-QI	3	+	32	2	–//– –//– и сохранением порядка
EVC-PF	полная	+	64	3	EVC-роутер с предложенной структурой буферов
FEMIDA-F	полная	+	64	3	Предлагаемая схема (FIFO-очереди)
FEMIDA-L	полная	+	64	3	Предлагаемая схема (фильтры + LRU-арбитр)

Конвейеры роутеров снабжены байпасами, при слабом трафике проходимыми за один такт. Буферы реализованы используя регистры в простейших роутерах, 2PSRAM в традиционных EVC-роутерах и DPSRAM при предложенной структуре; для локальных входов во всех схемах используются регистры. Глубины 8 и 32 в традиционных схемах выбраны для покрытия задержек в обычных и экспресс-виртуальных каналах; в роутерах с предложенной структурой буферов их глубины выбраны для такой же суммарной ёмкости, как в обычных EVC-роутерах. Разбиение буферов статическое, управление потоком – кредитное. Счётчики пакетов Preemptive Virtual Clock 10-битные, сбрасываются раз в 1000 тактов, все 10 бит используются в качестве приоритета.

Пакеты не делятся на посылки и передаются за один такт, каналы имеют задержку в два такта. Это соответствует процессорам с большими ядрами и относительно высоки-

ми частотами, которые станет возможным делать с рассматриваемым числом ядер при дальнейшем повышении плотности полупроводниковой технологии.

4.1. Равномерный трафик

Типовой сценарий максимальной нагрузки на сеть соединений – одновременная работа каждого ядра с распределённым кэшем. Наиболее интересен трафик запросов от ядер, где каждый узел является независимым источником пакетов, направляемых по случайному адресу. Для этого случая мы измерили пропускную способность сети (рис. 11), среднюю задержку (рис. 12) и, ограничивая пропускную способность выходов сети, честность её распределения между абонентами как относительное стандартное отклонение (рис. 13).

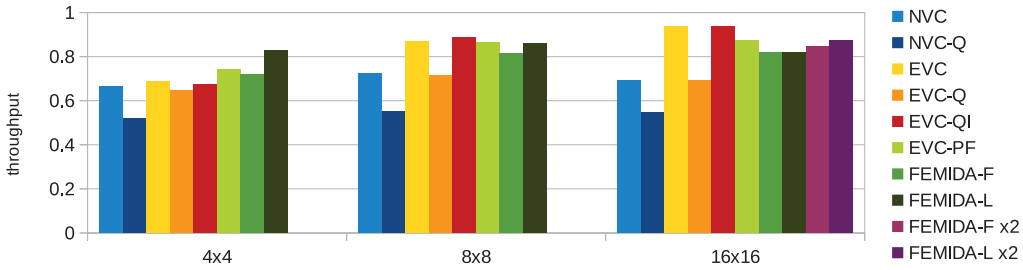

Рис. 11. Пропускная способность сети при равномерном трафике

Пропускная способность предлагаемой схемы составляет от 72 до 87% от идеальной, что сравнимо с результатами традиционных EVC-роутеров. Вследствие повышенной потребности в размере буферов, обсуждённой в разделе 3.2, при фиксированном их размере наша схема менее производительна в сети 16×16, чем в 8×8; результаты с вдвое увеличенным размером буферов для этого случая также приведены (x2).

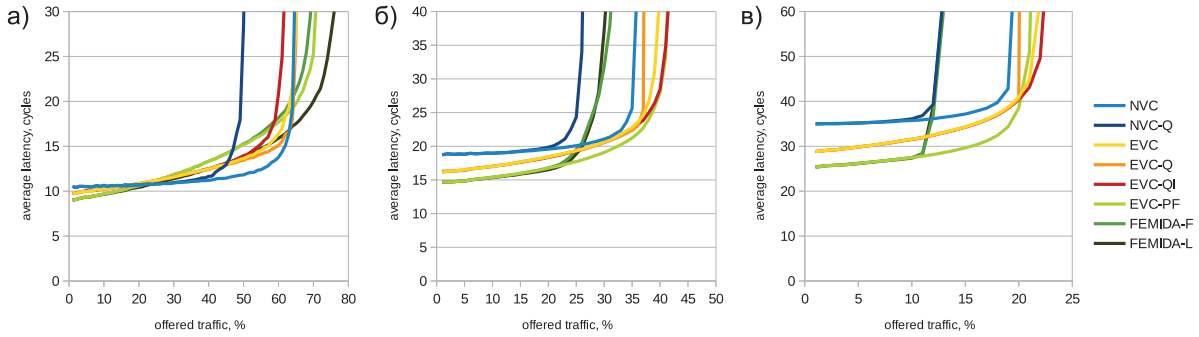

Рис. 12. Среднее время передачи пакета в сети а) 4×4, б) 8×8, в) 16×16

Задержки при минимальном трафике у предлагаемой схемы и EVC-PF ожидаемо самые низкие и превышают идеальные на четыре такта в соответствии с числом проходимых пакетом конвейеров – по одному такту на байпас одного конвейера. При возрастании нагрузки байпасы срабатывают реже, и задержки возрастают ещё на несколько тактов. При приближении к пропускной способности задержки стремятся к бесконечности, как и должно быть [2]. Некоторые схемы показывают более медленный рост задержек из-за нечестности с большей долей трафика из центральных узлов по сравнению с периферийными.

Честность сети при максимальном трафике, что соответствует случаю, когда сама сеть является узким местом системы, достигается как в предлагаемой схеме, так и в комбинациях традиционной EVC-схемы и Preemptive Virtual Clock (EVC-Q и EVC-QI). Однако при низком трафике, когда узкое место – кэш или память, эффективность PVC снижается, поскольку приоритеты недостаточно быстро достигают оптимальных значений после обнуления счётчиков, производимого периодически; только схемы с Fair Queuing – EVC-PF

и FEMIDA – хорошо справляются с задачей. Таким образом, только предлагаемая схема обеспечивает максимальную честность сети во всём диапазоне нагрузок.

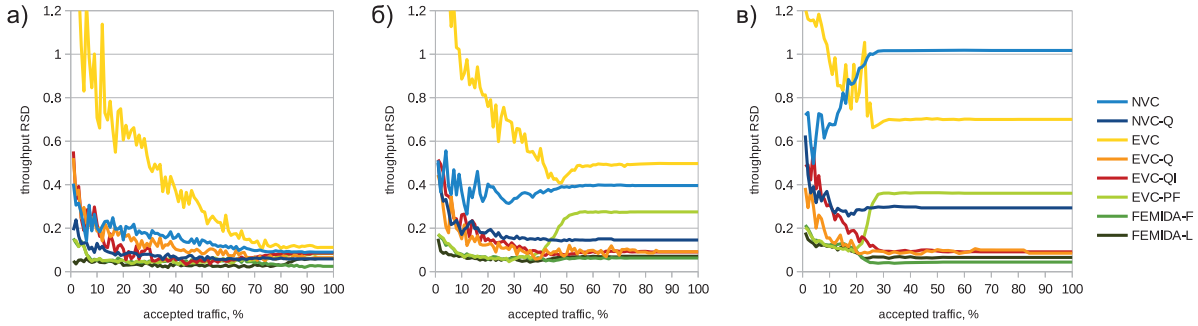

Рис. 13. Разброс пропускной способности абонентов сети 4×4, б) 8×8, в) 16×16

4.2. Hot-spot трафик

Другой сценарий тяжёлой нагрузки на сеть, встречающийся в процессорах с распределённым общим кэшем, подразумевает одновременную отправку пакетов из всех узлов в один, например при синхронизации. Это называется hot-spot трафиком.

Пропускная способность сети при таком сценарии составляет одну передачу за такт, задержки стремятся к бесконечности. Единственное отличие между разными реализациями сети заключается в честности (рис. 14); здесь она измерялась по формуле Джейна [10]:

$$\mathcal{J}(x_1, x_2, \ldots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \cdot \sum_{i=1}^{n} {x_i}^2},$$

где x_i – пропускная способность i-го абонента, n – их число.

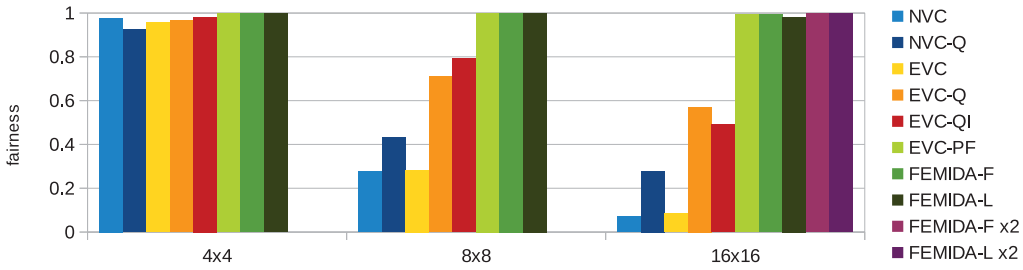

Рис. 14. Честность сети при hot-spot трафике

Все сети размера 4×4 достаточно честны, но при большем размере только Fair Queuing обеспечивает высокую честность. Результаты Preemptive Virtual Clock зависят от размера счётчиков и интервала их сброса; для приближения к результатам предлагаемой схемы в сети 16×16 EVC-QI требуется не менее 13 бит и 8000 тактов соответственно. При этом размер счётчиков значительно влияет на площадь, занимаемую роутерами. И само то, что на выравнивание качества обслуживания абонентов требуются тысячи тактов, является существенным недостатком – предлагаемой схеме для этого достаточно сотен.

4.3. Занимаемая площадь

Размер каждого роутера, оценённый путём технологического покрытия для частоты 1.5 ГГц, приведён на рис. 15. Мы установили размер пакетов в 9 байт, имитируя пакеты запросов, которым наиболее важны качества сети, перечисленные в разделе 1.

Preemptive Virtual Clock в оригинальной статье [5] стоила увеличения площади в 1.8 и 3.6 раз для сетей 8×8 и 16×16; в наших EVC-роутерах её цена получилась около 2.4 и 3.9

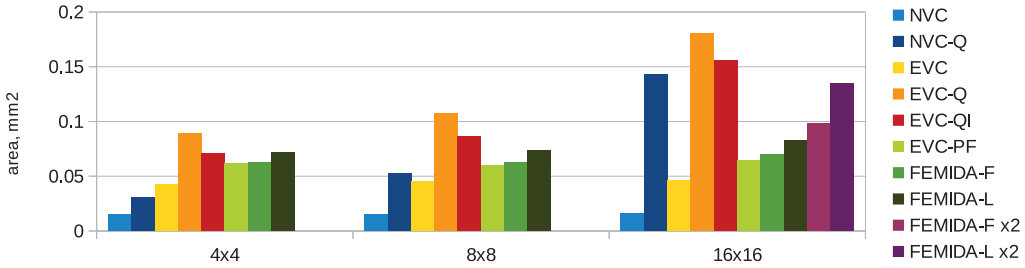

Рис. 15. Площадь роутера при технологии 28 нм

раз, а при упрощении схемы для сохранения порядка пакетов – 1.9 и 3.4 раз. Предлагаемая схема масштабируется лучше, требуя 1.4 и 1.5 в простом варианте и 1.6 и 1.8 в версии с LRU-арбитром, если при этом не менять размер буферов. Если в сети 16×16 их удвоить, это повышает цену до 2.1 и 2.9 раз, что всё равно дешевле, чем PVC.

5. Заключение

В данной работе мы предложили архитектуру сети на кристалле, сочетающаю высокую пропускную способность, близкие к идеальным задержки, честность в распределении пропускной способности между абонентами и сохранение порядка пакетов.

Эксперименты подтвердили эти качества и хорошую масштабируемость сети при размерах сети от 4×4 до 16×16, чего не демонстрируют существующие решения.

Пропускная способность сети предлагаемой схемы в варианте с фильтрами и LRU-арбитрами и с буферами глубиной 64 (FEMIDA-L) при всех размерах сети от 4×4 до 16×16 составляет более 80% от теоретической, при этом площадь роутера для всех размеров сети сравнима с традиционной EVC-схемой дополненной QoS-механизмом Preemptive Virtual Clock (EVC-Q или EVC-QI) для размера 4×4 и примерно вдвое меньше таковой для 16×16. Другие рассмотренные схемы значительно нечестны в большинстве сценариев, и EVC-Q и EVC-QI также демонстрируют худшую честность в некоторых случаях. Минимальные базовые задержки достигаются только предлагаемыми схемами (FEMIDA-F и FEMIDA-L) и EVC-роутером с Fair Queuing по строкам и столбцам (EVC-PF), но последний значительно нечестен при высоком равномерном трафике и размере сети 8×8 и более.

Результаты позволяют сделать вывод, что предложенная схема может использоваться в микропроцессорах с распределённым общим кэшем и десятками или сотнями ядер.

Литература

1. *De Micheli G., Benini L.* Networks on Chips: Technology and Tools. Morgan Kaufmann Publishers Inc., 2006.

2. *Dally W., Towles B.* Principles and Practices of Interconnection Networks. Morgan Kaufmann Publishers Inc., 2003.

3. *Grot B., Hestness J., Keckler S.W., Mutlu O.* Kilo-NOC: A Heterogeneous Network-on-Chip Architecture for Scalability and Service Guarantees // Proc. 38th Int'l Symp. Computer Architecture (ISCA '11), IEEE CS. 2011. P. 268–279.

4. *Kumar A., Peh L.-H., Kundu P., Jha N.K.* Express Virtual Channels: Towards the Ideal Interconnection Fabric // Proc. Int'l Symp. Computer Architecture (ISCA '07), ACM Press. 2007. P. 150–161.

5. *Grot B., Keckler S.W., Mutlu O.* Preemptive Virtual Clock: a Flexible, Efficient, and Cost-effective QOS Scheme for Networks-on-Chip // Proc. 42nd Annual IEEE/ACM Int'l Symp. Microarchitecture (MICRO 42), ACM. 2009. P. 268–279.

6. *Ebrahimi E., Lee C.J., Mutlu O., Patt Y.N.* Fairness via Source Throttling: A Configurable and High-Performance Fairness Substrate for Multicore Memory Systems // ACM Trans. Comput. Syst. 2012. V. 30, I. 2. P. 1–35.

7. *Kim J., Nicopoulos C., Park D., Narayanan V., Yousif M.S., Das C.R.* A Gracefully Degrading and Energy-Efficient Modular Router Architecture for On-Chip Networks // Proc. 33rd Int'l Symp. Computer Architecture (ISCA '06), IEEE. 2006. P. 4–15.

8. *Ramanujam R.S., Soteriou V., Lin B., Peh L.-S.* Design of a High-Throughput Distributed Shared-Buffer NoC Router // Proc. 2010 4th ACM/IEEE Int'l Symp. Networks-on-Chip (NOCS '10), IEEE. 2010. P. 69–78.

9. *Harteros K., Katevenis M.* Fast Parallel Comparison Circuits for Scheduling. Technical Report TR-304, FORTH-ICS, 2002.

10. *Jain R.K., Chiu D.-M.W., Hawe W.R.* A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems — DEC Research Report TR-301, 1984.

References

1. *De Micheli G., Benini L.* Networks on Chips: Technology and Tools. Morgan Kaufmann Publishers Inc., 2006.

2. *Dally W., Towles B.* Principles and Practices of Interconnection Networks. Morgan Kaufmann Publishers Inc., 2003.

3. *Grot B., Hestness J., Keckler S.W., Mutlu O.* Kilo-NOC: A Heterogeneous Network-on-Chip Architecture for Scalability and Service Guarantees. Proc. 38th Int'l Symp. Computer Architecture (ISCA '11), IEEE CS. 2011. P. 268–279.

4. *Kumar A., Peh L.-H., Kundu P., Jha N.K.* Express Virtual Channels: Towards the Ideal Interconnection Fabric. Proc. Int'l Symp. Computer Architecture (ISCA '07), ACM Press. 2007. P. 150–161.

5. *Grot B., Keckler S.W., Mutlu O.* Preemptive Virtual Clock: a Flexible, Efficient, and Cost-effective QOS Scheme for Networks-on-Chip. Proc. 42nd Annual IEEE/ACM Int'l Symp. Microarchitecture (MICRO 42), ACM. 2009. P. 268–279.

6. *Ebrahimi E., Lee C.J., Mutlu O., Patt Y.N.* Fairness via Source Throttling: A Configurable and High-Performance Fairness Substrate for Multicore Memory Systems. ACM Trans. Comput. Syst. 2012. V. 30, I. 2. P. 1–35.

7. *Kim J., Nicopoulos C., Park D., Narayanan V., Yousif M.S., Das C.R.* A Gracefully Degrading and Energy-Efficient Modular Router Architecture for On-Chip Networks. Proc. 33rd Int'l Symp. Computer Architecture (ISCA '06), IEEE. 2006. P. 4–15.

8. *Ramanujam R.S., Soteriou V., Lin B., Peh L.-S.* Design of a High-Throughput Distributed Shared-Buffer NoC Router. Proc. 2010 4th ACM/IEEE Int'l Symp. Networks-on-Chip (NOCS '10), IEEE. 2010. P. 69–78.

9. *Harteros K., Katevenis M.* Fast Parallel Comparison Circuits for Scheduling. Technical Report TR-304, FORTH-ICS, 2002.

10. *Jain R.K., Chiu D.-M.W., Hawe W.R.* A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems — DEC Research Report TR-301, 1984.

Поступила в редакцию 29.05.2017