`УДК 004.27

Архитектура планировщика процессора сопоставления ППВС «Буран»

Д.Н. Змеев, Н.Н. Левченко, А.С. Окунев, А.В. Климов

Институт проблем проектирования в микроэлектронике РАН, zmejevdn@ippm.ru, nick@ippm.ru, oku@ippm.ru, klimov@ippm.ru

***Аннотация* — В данной статье описываются аппаратно-программные средства планировщика процессора сопоставления ППВС «Буран», включение которого в состав вычислительного ядра позволит эффективно управлять работой процессора сопоставления при неравномерной загрузке исполнительного устройства и предотвращать блокировки вычислительного процесса при переполнении памяти ключей и других аппаратных ресурсов вычислительной системы.**

***Ключевые слова* — планировщик процессора сопоставления, модель вычислений с управлением потоком данных.**

I. Введение

Параллельная потоковая вычислительная система (ППВС) «Буран» [1] представляет собой многоядерную масштабируемую вычислительную систему. В состав вычислительного ядра входят исполнительное устройство, процессор сопоставления, коммутатор токенов, блок хэширования. Между ядрами в системе передаются единицы информации в виде токенов. Коммутация между ядрами осуществляется по номеру вычислительного ядра, который вырабатывается в блоке хэширования на основе полей токена и настраиваемой программистом функции распределения.

Процессор сопоставления (ПС) в ППВС обеспечивает взаимодействие токенов различных типов в зависимости от содержимого поля токена «код операции», сравнивая поля ключей токенов с учетом кратности и оценивая состояние других полей токенов. Ключи токенов сравниваются в ассоциативной памяти ключей (АПК), а сами токены хранятся в памяти токенов (ПТ).

Памяти, в которых хранятся ключи токенов и сами токены, не могут быть бесконечного размера, причем память ключей токена должна быть ассоциативной. Вообще, согласно принципам программирования параллельной потоковой вычислительной системы для программиста отсутствует представление о памяти в традиционном понимании. В частности, при написании программ нет понятия записи в память или чтения из нее, однако в аппаратной реализации архитектуры физическая память существует, а ее размеры – вынужденно ограничены. Тем не менее, в нашей модели вычислений введено понятие виртуального адресного пространства токенов большой размерности, связанное с полем «Ключ» токена.

Для того чтобы обеспечить работу программ в ППВС, необходимо эффективно управлять заполнением локальных памятей устройств сопоставления. Для этого были разработаны методы автоматической откачки/подкачки токенов в зависимости от необходимости «освобождать» память от пока невостребованных токенов, и возвращать эти токены в работу при наличии свободного места в памяти устройства сопоставления и снижении активности вычислительных процессов.

II. Структура и функции планировщика процессора сопоставления

Основными функциями планировщика ПС являются:

- установка начальных физических адресов областей откачиваемых и остановленных токенов в многоуровневой внешней памяти токенов (ВПТ), причем для каждого временно*го* этапа (подмножества) откачиваемых и останавливаемых токенов имеется своя область ячеек в ВПТ;
- распределение свободных страниц ВПТ;
- поддержка операционной системы с точки зрения распределение ресурсов ВПТ и связи ВПТ с дисковой памятью;
- сбор статистики по выполняемым задачам:
- анализ и фиксация количества свободных ячеек в АПК;
- мониторинг интенсивности формирования пакетов по каждому этапу;
- мониторинг интенсивности прихода токенов на вход АПК по каждому этапу;
- анализ загрузки буфера готовых пакетов по каждому этапу;
- выполнение ротации задач и подзадач;
- проверка активности задач, реализация функции останова задач.

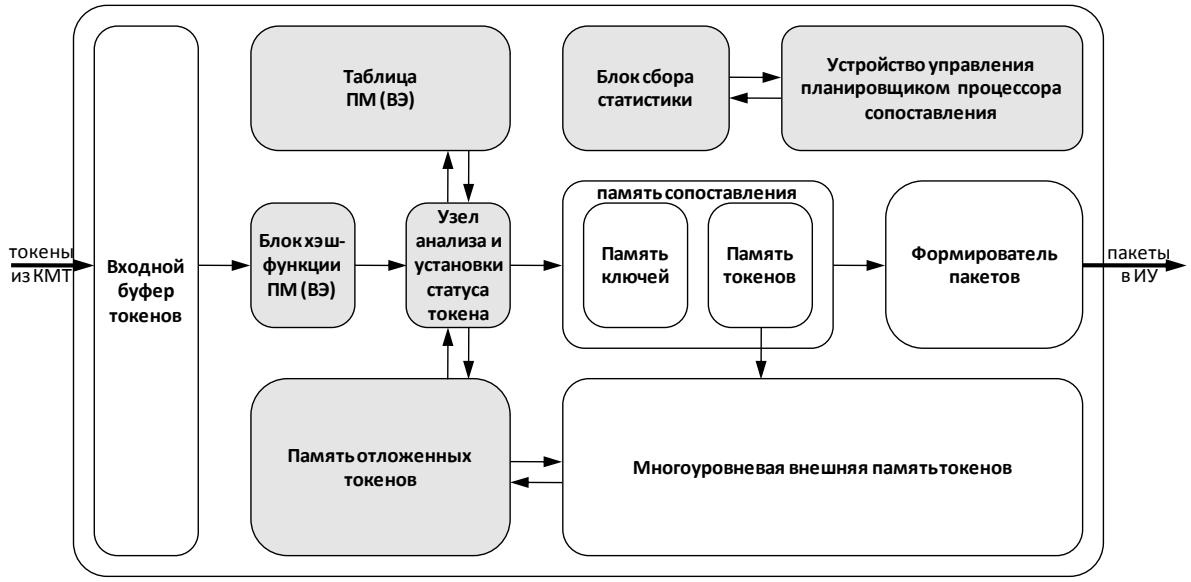

Рис. 1. Архитектура планировщика процессора сопоставления

Временн***ы***м этапом (или подмножеством) мы называем группу токенов, которая необходима для выполнения определенного интервала вычислений. Как правило, она связана с более или менее независимым действием (например, итерации в программе могут быть оформлены как временн***ы***е этапы). Задача разбивается на этапы программистом при помощи имеющихся средств хэширования, которые можно настроить непосредственно под выполняемую задачу. Эти средства в процессе выполнения задачи присваивают этапам определенные номера, с которыми в дальнейшем и работает аппаратура планировщика, размещая эти этапы во времени прохождения задачи.

Функционально планировщик ПС состоит из следующих основных блоков (на рис. 1 они выделены темным фоном):

- блока хэш-функций подмножеств (ПМ);
- узла анализа и установки статуса токена;
- таблицы подмножеств (или временн***ы***х этапов – ВЭ);
- блока сбора статистики;
- устройства управления планировщиком процессора сопоставления;
- памяти отложенных токенов.

Блок хэш-функций подмножеств предназначен для формирования номера подмножества (временн***о***го этапа) на основании заранее заданной функции преобразования ключа токена.

Таблица подмножеств предназначена для хранения статусного состояния подмножеств, а также дополнительной статистической информации. Каждая строка памяти таблицы подмножеств состоит как минимум из трех полей: поля «№ этапа», поля «Статус этапа» и поля «Число токенов этапа».

Узел анализа и установки статуса токена оценивает приходящий токен и устанавливает необходимый статус токена, в соответствии с номером подмножества в таблице подмножеств.

Блок сбора статистики собирает следующую информацию:

- количество токенов задачи, приходящих во все вычислительные ядра (ВЯ) за единицу времени;
- уровень загрузки исполнительных устройств;
- количество хранящихся в процессоре сопоставления токенов задачи в конкретном ВЯ;
- количество откачанных токенов;
- количество свободных ячеек в ассоциативной памяти ключей;
- интенсивность формирования пакетов по каждому подмножеству;
- интенсивность прихода токенов на вход ассоциативной памяти ключей по каждому подмножеству;
- загрузку буфера готовых пакетов по каждому подмножеству;
- загруженность памятей процессора сопоставления и исполнительного устройства (ИУ), различных буферов вычислительного модуля и загрузка ИУ (процент использования функциональных устройств ИУ).

В частности, для сбора статистики в таблице подмножеств имеется поле «Число токенов этапа». Если

токен принадлежит к «отложенному» или «откачанному» этапу, то сразу же происходит изменение поля «Число токенов этапа». В противном случае, когда токен принадлежит к «активному» этапу, изменение данного поля происходит только по результатам обработки ключа токена в АПК, поскольку в случае взаимодействия пришедшего токена («верхнего») с уже имеющимися в АПК («нижними») может произойти уменьшение количества токенов этапа. Подмножество считается пустым при счетчике равном нулю, то есть в области хранения «отложенных» токенов этого подмножества нет.

Память отложенных токенов предназначена для хранения откладываемых в процессе решения задачи токенов соответствующих подмножеств. Она представляет собой прямоадресуемую память, в которой токены упорядочены по номерам этапов.

Устройство управления планировщиком координирует работу всех узлов и блоков планировщика.

III. Статусы данных (токенов) в памяти

Во внешнюю память токенов могут быть записаны следующие данные:

- отложенные токены, которые находятся в переходном состоянии к статусу «откачанных»;
- «откачанные» токены соответствующих этапов;
- откачанные пакеты соответствующих этапов;
- спецтокены для загрузки памяти ИУ.

Таким образом, вводятся два дополнительных пула (или области в памяти) токенов: пул задержки и пул откачки. В пул задержки помещаются токены, не поданные на обработку в память сопоставления, поскольку они относятся к пассивному этапу (они просто как бы задерживаются перед входом в неё). Такие токены называются отложенными. В пул откачки перемещаются (откачиваются) токены из памяти сопоставления в связи с её переполнением. Эти токены называются откачанными. В обоих пулах токены сгруппированы по этапам.

В таблице подмножеств каждому этапу сопоставлен признак (статус) "активный/ отложенный/ откачанный/ остановленный". Статусы «отложенный/ откачанный/ остановленный» относятся к так называемому «пассивному» этапу. Если пришедший в ПС токен относится к активному этапу, он направляется в память сопоставления на поиск. Если – к пассивному, он направляется в пул задержки. Соответственно, и токены делятся на активные и пассивные.

Каждая задача (подзадача) должна иметь в ВПТ несколько типов областей (сегментов) для вышеописанных токенов. Поэтому можно иерархически структурировать содержимое ВПТ, например, для подзадач - верхняя ступень структурирования - по номеру подзадачи, нижняя – по номеру этапа.

В каждый момент времени один из этапов может быть объявлен переходным. Он может быть в одном из двух состояний: откачки или подкачки. Если приходящий токен относится к переходному этапу (активному или пассивному), то он откладывается так же, как и в случае просто пассивного этапа.

Переходный этап может быть работающим или приостановленным. В первом случае откачка или подкачка выполняется по одному токену за такт: один токен этого этапа переписывается из памяти сопоставления в пул откачки или наоборот. Во втором случае ничего не делается. Решение о приостановке или возобновлении работы переходного этапа принимается на основе сравнения текущего заполнения памяти сопоставления с соответствующим порогом. Всего должны быть заданы четыре порога: для приостановки или возобновления, подкачки или откачки. По завершению откачки (когда токенов данного этапа нет в памяти) переходный пассивный этап автоматически становится просто пассивным. По завершению подкачки (когда токенов данного этапа нет в пуле откачки) переходный активный этап становится просто активным.

IV. Работа планировщика процессора сопоставления

В процессе создания программы программист самостоятельно определяет или выбирает из уже имеющегося набора (и настраивает под свою задачу) функцию, формирующую номер подмножества (этапа), которая зависит, в основном, от ключа токена. При этом этапы с меньшими номерами не должны зависеть (по возможности) от этапов с большими номерами. Это означает, что не должно быть интенсивного обмена между этими этапами (обмена в двух направлениях), то есть подмножества должны обладать локальностью во времени. Однако, если такая зависимость существует, нужно исключить обмен токенами хотя бы с этапами, имеющими намного больший номер.

Планировщик ПС начинает функционировать с поступлением во «входной буфер токенов» токенов из внутреннего коммутатора вычислительного модуля. Далее, в узле анализа и установки статуса токенов происходит определение номера этапа, к которому принадлежит токен, для чего вычисляется хэш-функция от ключа пришедшего токена по заранее определенному алгоритму. На следующем шаге, по номеру этапа, в таблице подмножеств выявляется статус этапа, к которому относится пришедший токен. На основе извлеченного из АТП статуса пришедшему токену присваивается выработанный по определенному алгоритму статус. Следует отметить, статусы этапов и статусы токенов не идентичны, в частности для переходных этапов.

В случае, если токен по результатам поиска относится к «активному» этапу, он направляется на поиск в АПК, если к «отложенному» этапу – направляется в область хранения «отложенных» токенов, а если к «откачанному» этапу – записывается область хранения «откачанных» токенов.

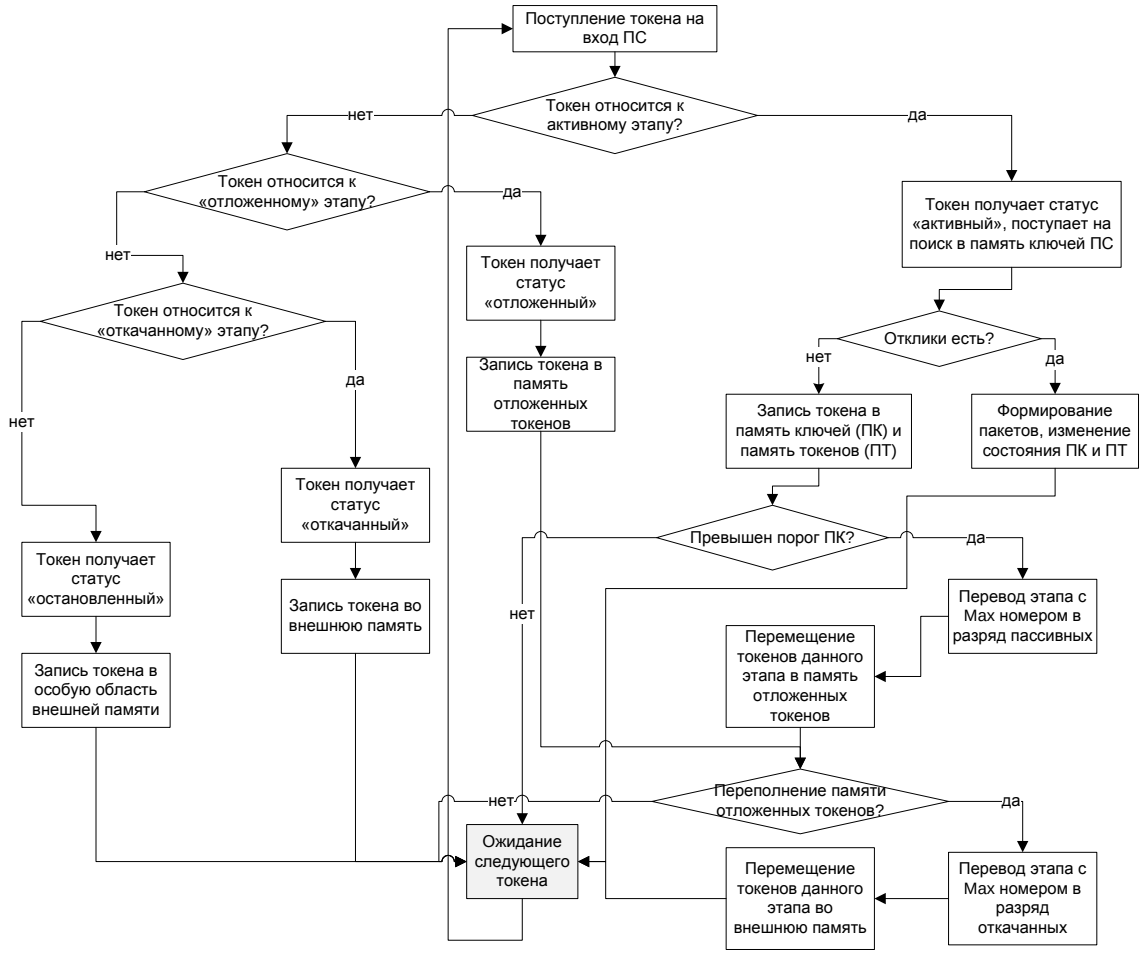

Рис. 2. Общий алгоритм работы планировщика ПС (основные режимы)

Работа планировщика ПС непосредственно связана с информацией, которую поставляет блок сбора статистики (БСС). БСС собирает статистику по токенам, пакетам и интенсивности их формирования, поступления для задач, подзадач и этапов. Кроме того собирается статистика по загруженности памятей ПС и ИУ, различных буферов вычислительного модуля и загрузке ИУ (процент использования функциональных устройств ИУ).

Устройство управления планировщика ПС выполняет работу по активации/деактивации этапов, установке порогов (в АПК) срабатывания различных режимов работы при откачке/подкачке токенов, обеспечивает также работу с задачами и подзадачами, выполняя различные команды управления, поступающие извне.

Основные режимы алгоритма работы планировщика ПС представлены на рис. 2.

V. Заключение

В статье представлена архитектура планировщика ПС ППВС, введение которого в состав аппаратуры вычислительного ядра системы позволяет решить проблемы переполнения памяти сопоставления, обеспечивает эффективную загрузку исполнительных устройств и исключает блокировку вычислительного процесса при прохождении задач. Приведена структура и функциональные особенности планировщика ПС, описана работа отдельных узлов и блоков.

Основные алгоритмы функционирования планировщика были проверены на программной модели ППВС. Получены данные по использованию предложенных решений, свидетельствующие о сокращении для некоторых задач необходимого размера ассоциативной памяти ключей (в среднем – на порядок).

Литература

[1] Стемпковский А.Л., Климов А.В., Левченко Н.Н., Окунев А.С. Методы адаптации параллельной потоковой вычислительной системы под задачи отдельных классов // Информационные технологии и вычислительные системы. 2009. №3. С. 12-21.

[2] Климов А.В., Левченко Н.Н., Окунев А.С. Преимущества потоковой модели вычислений в условиях неоднородных сетей // Информационные технологии и вычислительные системы. 2012. № 2. С. 36-45.