КОНЦЕПЦИЯ ГЕТЕРОМОРФНОЙ МУЛЬТИЯДЕРНОЙ ОПЕРАЦИОННОЙ СИСТЕМЫ

Е. И. Клименков

Белорусский университет информатики и радиоэлектроники,
кафедра программного обеспечения информационных технологий
ул. Гикало, 9, г. Минск, Республика Беларусь
телефон(ы): + (37533) 6045147; e-mail: ZarathustrA@np.by

Бурное развитие процессоров, массовый переход на многоядерные процессоры и большой опыт знаний накопленных со времен UNIX позволяют по-новому взглянуть на архитектуру операционной системы и выдвинуть набор требований к такой архитектуре, а также осветить некоторые решения и проблемы.

Ключевые слова — операционная система, многоядерность.

1 КОНЦЕПЦИЯ ГЕТЕРОМОРФНОЙ МУЛЬТИЯДЕРНОЙ ОПЕРАЦИОННОЙ СИСТЕМЫ

Архитектура компьютерных систем претерпевает изменения значительно быстрее, чем архитектура операционных систем. Внутренняя архитектура современного компьютера напоминает распределенную сетевую систему, состоящую из смеси процессорных ядер, кэшей, внутренних связей, устройств ввода/вывода и карт расширения. Современный компьютер похож на ранние параллельные системы или на многопроцессорные системы конца прошлого века. Многоядерные компьютерные системы, которые, по сути, являются теми же многопроцессорными системами, локализованными на процессорной подложке [3], занимают все более и более прочные позиции в большинстве сегментов компьютерного рынка. Им не покорился пока только сегмент встроенных систем. Более того, За многоядерными процессорами будущее, так как производители процессоров уже вплотную подошли к пределу возможностей транзисторной микроэлектроники и дальнейший рост производительности в этом направлении уже не возможен. Дальнейший рост производительности процессоров может быть обеспечен только интенсивными мерами, заключающимися в развитии архитектуры ядра, и экстенсивными мерами, заключающимися в наращивании количества ядер процессора. Исследования и разработки в области многоядерности процессоров были признаны главным направлением развития процессоров двумя титанами процессорного рынка, компаниями Intel и AMD.

Основы архитектуры всех современных операционных систем общего назначения, таких как Windows, Linux, BSD и MacOS, были заложены операционной системой UNIX, которая в свое время явилась настоящей революцией в сфере управления компьютерных систем. Напомним, что она появилась в 1969 году благодаря сотрудникам AT&T Labs на базе более раннего проекта Multics [8]. За более чем 45 лет представление об архитектуре компьютерных систем значительно изменились. Современные процессоры обладают очень развитыми ядрами с большим количеством возможностей. О росте сложности процессоров и их возможностей можно судить по объему спецификаций процессора предназначенных для программистов. Например, если объем спецификации процессора Intel i386, выпущенного в 1985 году составляет 421 страницу, то объем спецификации процессора Intel Pentium, выпущенного в 1993 году составляет уже 1032 страницы [1][2]. Традиция наследования архитектурных принципов UNIX при разработке операционных систем приводила к тому, что аппаратные возможности процессора лишь учитывались, но не являлись отправной точкой архитектуры. Такой подход имел свой запас возможностей, который практически исчерпал себя к моменту массового появления многоядерных процессоров. Возможности современных процессоров не в состоянии полностью раскрыться под управлением операционных систем прошлого века.

Существуют следующие проблемы при использовании традиционных ОС на современных процессорах:

- Операционная система на возможно более низком уровне старается эмулировать одноядерный процессор для всего вышележащего ПО. С одной стороны это упрощает структуру системы, но в тоже время лишает и многих возможностей, которые бы можно было бы реализовать в многоядерной системе. Разработчики операционных систем упорно стараются рассматривать процессор как одноядерный.
- Поскольку ядра процессора под управлением традиционных ОС работают с общей памятью, то вопрос межзадачной синхронизации приводит к чрезмерно частому сбросу кэша процессора и блокированию его работы, что весьма негативно отражается на общей производительности системы.
- Не полностью использует возможности процессора.

Ко всему этому списку в ближайшем будущем добавиться еще одна большая проблема. В ближайшем будущем на рынок выйдут гетероморфные процессоры, такие как AMD Fusion, в которых на одной процессорной подложке будут располагаться ядра с разной архитектурой и набором команд. Эффективное использование таких процессоров в традиционных операционных системах будет более чем проблематичным.

Данная проблема уже была осознанна научной общественностью, и усилиями Швейцарской высшей технической школы Цюриха и корпорацией Microsoft был разработан первый исследовательский проект в этой области. Этот проект носит название Barellfish [6].

Концепция гетерогенной многоядерной операционной системы включает в себя следующие принципы:

- Операционная система должна быть действительно многоядерной. То есть каждое процессорное ядро должно работать под управлением своего ядра операционной системы. Следовательно, аварийная ситуация на одном ядре не должна приводить к отказу всей операционной системы.
- Операционная система должна быть гетерогенной. На одном компьютере должны иметь возможность работать несколько ядер с различными параметрами и архитектурой.
- Должен существовать механизм взаимодействия между ядрами операционной системы, обеспеченный универсальным и стабильным интерфейсом.
- Все ядра имеют в системе одинаковую значимость и работают друг с другом по системе Pear To Pear, а не по системе клиент-сервер.
- Каждое ядро операционной системы должно обладать своим собственным набором ресурсов.
- Ядра операционной системы должны быть защищены от несанкционированного вмешательства в свою деятельность другого ядра.
- Операционная система должна обладать межъядерным блоком параметров служащим для учета ресурсов компьютерной системы.

Ко всем этим требованиям присущим непосредственно многоядерным и гетерогенным операционным системам следует добавить принципы уже давно созревшие в концепции микроядра [4]:

- Ядро должно быть минималистичным. Большинство задач, решающихся в традиционных системах в ядре операционной системы должны решаться специальными ядерными службами, работающими вне ядра.
- Ядро должно являть собою нечто подобное драйверу процессора или HAL. При миграции операционной системы на другую аппаратную платформу должен меняться только код ядра.
- Только ядро обладает наивысшими привилегиями. Никакой разработчик программного обеспечения, за исключением ядра не может рассчитывать на полный набор привилегий.
- Главные задачи ядра — это управление обработкой прерываний, многозадачность и межъядерное взаимодействие.

Также, в целях повышения гибкости платформы гетерогенной многоядерной операционной системы в нее необходимо включить набор требований, полученный при разработке теории операционных систем реального времени [5], которые Квинг Ли называл операционными системами следующего поколения, эры пост-PC вычислений. Эти требования определяют приоритеты в разработке ядра:

- Жесткая система обработки прерываний, способная работать в режиме реального времени.
- Гибкость
- Предсказуемость
- Производительность
- Компактность
- Масштабируемость

Платформа гетерогенной многоядерной операционной системы предъявляет особые требования к загрузчику операционной системы. В отличие от загрузчиков традиционных операционных систем в задачи загрузчика гетерогенной многоядерной операционной системы входят следующие задачи:

- Более полное определение аппаратной конфигурации компьютерной системы. В том числе типа многопроцессорности (SMP, NUMA и т.д.), количество и типы процессорных ядер, архитектура контроллеров прерываний [3].
- Загрузка на все процессорные ядра системы ядер операционной системы в соответствии с определенной аппаратной конфигурацией и политикой инициализации системы.
- Первоначальное разделение аппаратных ресурсов компьютера между всеми ядрами операционной системы.

В целях повышения гибкости операционной системы для каждой задачи предлагается ввести дополнительный атрибут — вес задачи, которым определяется значимость задачи для операционной системы. Этот параметр используется при конкурентном разделении ресурсов между задачами в целях разрешения конфликтов. Более значимая задача имеет больше прав на получение ресурсов системы. Следует отметить, что вес задачи и ее приоритет не являются одним и тем же понятием. Приоритет задачи определяет долю процессорного времени реально выделяемого задаче. Вес же задачи, определяет какую долю ресурса и в какую очередь может получить задача. Следовательно, задача с большим весом может выполняться с маленьким приоритетом. Задача же с маленьким весом не может выполняться с большим приоритетом. Вес задачи используется для управления обработкой прерываний, разделения процессорного времен и в борьбе за шлюзы вызова. При этом может проводиться политика отбора ресурсов у менее значимой задачи или даже ее полного вытеснения из системы.

Описанная платформа гетероморфной многоядерной системы обладает высокой масштабируемостью и гибкостью. Загрузчик ОС во время своей работы манипулирует предоставленными ему ядрами в соответствии с выбранной для него политикой загрузки ОС. Ядра представляют собой минимизированные HAL-модули разрабатываемые исходя из специфики и возможностей конкретного процессора, они предоставляют как можно более универсальный интерфейс ядерным службам. Ядерные службы могут работать в любом количественном и качественном составе в зависимости от нужд и задач операционной системы. Они же формируют набор API для прикладных задач. Служащих для решения непосредственных пользовательских задач. На базе данной платформы можно будет построить любую операционную систему, максимально приспособленную для решения конкретных задач, начиная от операционных систем для встраиваемых систем и роботов и заканчивая операционными системами общего назначения, серверными системами и системами для суперкомпьютеров.

Бурное развитие процессоров и накопленные знания в области теории операционных систем делают необходимым и возможным создание новой архитектуры операционной системы наиболее полно использующей ресурсы современных и будущих компьютерных систем и обладающей новым уровнем универсальности и масштабируемости, открывающей возможности ее применения практически во всех видах компьютерных систем. Нами была поставлена задача и выдвинут набор требований к этой архитектуре, однако еще необходим большой комплекс исследований позволяющих взглянуть на вопросы управления обработкой прерываний, управление памятью и многозадачность через призму гетероморфной мультиядерности.

ЛИТЕРАТУРА

[1] Intel 80386 Programmer's reference manual / Intel corporation // Intel corporation – 1986.

[2] Pentium Processor Family Developer's Manual / Intel Corporation // Intel Corporation – 1995. – T.3: Architecture and Programming Manual / Intel Corporation – 1995.

[3] MultiProcessor Specification / Intel Corporation // Intel Corporation – 1997.

[4] Операционные системы. Разработка и реализация / Э. Таненбаум, А. Вудхалл // Питер – 2007.

[5] Real-Time Concepts for Embedded Systems / Qing Li // CMP Books – 2003.

[6] The Multikernel: A new OS architecture for scalable multicore systems / A. Baumann [и др.], [Электрон. ресурс] // Режим доступа: http://www.barrelfish.org/barrelfisk_sosp09.pdf.

[7] The Indispensable PC Hardware Book / H. Messmer // Addison-wesley - 2001

[8] Архитектура операционной системы UNIX / М.Д. Бах, [Электрон. ресурс] // Режим доступа: http://www.citforum.ru/operating_systems/bach/contents.shtml