ОБЕСПЕЧЕНИЕ ПОДДЕРЖКИ ПРИЛОЖЕНИЙ РЕАЛЬНОГО ВРЕМЕНИ ОПЕРАЦИОННЫМИ СИСТЕМАМИ ОБЩЕГО НАЗНАЧЕНИЯ

М. В. Данилов

Санкт-Петербургский институт информатики и автоматизации РАН
199178, Санкт-Петербург, 14-я линия В.О., д.39
rt_mvd@mail.ru

УДК 681.3.06
М. В. Данилов. **Обеспечение поддержки приложений реального времени операционными системами общего назначения** // Труды СПИИРАН. Вып. 1, т. 3. — СПб: СПИИРАН, 2003.
Аннотация. *Рассматривается проблема расширения состава сервисов таких ОС общего назначения как Linux в направлении поддержки задач жесткого реального времени. Приводится вариант решения этой проблемы за счет включения в состав ОС дополнительного ядра реального времени. Предлагается подход, обеспечивающий улучшение реактивности и эффективности планирования задач в рамках двухядерной архитектуры ОС.* — Библ.7 назв.

UDC 681.3.06
M. V. Danilov. **Real-time application support by general-purpose operating systems** // SPIIRAS Proceedings. Issue 1, v.3. — SPb: SPIIRAS, 2003.
Abstract. *The problem under consideration is an extension of the set of services of Linux-like OS in a direction of hard real-time tasks support. The case of such problem solution is regarded that consists of including into the OS an additional real-time kernel. An approach is suggested for improving of the OS reactivity and task scheduling efficiency for dual-kernel OS architecture framework.* — Bibl.7 items.

Рост производительности средств вычислительной техники сопровождается качественным изменением состава сервисов, предоставляемых современными Операционными Системами Общего Назначения (ОСОН) прикладным программам. В последнее время все большее число разработчиков ОС для персональных компьютеров и рабочих станций декларирует поддержку прикладных задач жесткого реального времени [6]. Однако совмещение свойств ОСОН и ОС реального времени в рамках однопроцессорной системы сопряжено с рядом проблем, причина которых скрыта в различии, а зачастую и противоречии, требований, предъявляемых к вычислительным системам общего назначения и реального времени [3]. Как следствие, попытки приспособить ОСОН к поддержке задач реального времени нередко приводят лишь к частичному достижению желаемых результатов. Данная статья посвящена рассмотрению вопросов адаптации ОСОН к потребностям прикладных систем, содержащих задачи жесткого реального времени. Рассмотрение ведется на примере ОС *Linux*.

1. Системы реального времени

Отличительной особенностью Систем Реального Времени (СРВ) является наличие ограничений на время выполнения задач. Необходимость получения результатов вычислений в заданные сроки выдвигает на передний план такое свойство системы как предсказуемость поведения во времени, качественная СРВ не может быть построена без тщательного анализа временных характеристик всех компонент системы.

В зависимости от характера требований ко времени выполнения различа-

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

ют мягкие и жесткие СРВ. СРВ принадлежит к классу жестких, если нарушение установленных сроков выполнения задач недопустимо и приводит систему в некорректное состояние. СРВ называют мягкой, если получение результата выполнения задачи после установленного срока нежелательно, но допустимо.

В общем случае СРВ не обязательно должна создаваться на базе ОС. Однако, при конструировании сложного программного комплекса, состоящего из множества взаимодействующих задач, использование ОС в значительной мере сокращает сроки разработки системы, а также способствует повышению ее надежности. В качестве программной платформы для приложений реального времени, как правило, используются специализированные ОС Реального Времени (ОСРВ).

2. Операционные системы реального времени

Для обеспечения своевременного выполнения прикладных задач операционные системы РВ используют специфические механизмы управления ресурсами системы. В отличие от ОСОН, распределяющих ресурсы так, чтобы обеспечить максимальную производительность, ОСРВ производит планирование порядка доступа задач к разделяемым ресурсам (в том числе к ресурсу процессора) с учетом временных характеристик задач.

Механизмы, предоставляемые ОСРВ, должны быть предсказуемыми как по результатам, так и по времени выполнения. Данное требование означает, что прикладному программисту, использующему в качестве программной платформы ОСРВ, должна быть предоставлена исчерпывающая информация о принципах действия и временных характеристиках работы сервисов ОС.

Одним из основных показателей качества ОСРВ является реактивность. Время реакции СРВ на внешнее событие определяется максимальной продолжительностью временного интервала, на протяжении которого запрещены внешние прерывания, временем создания задачи, временем переключения контекста и т.п. Очевидно, что ОСРВ с большим временем реакции на внешнее событие находит ограниченное применение.

3. Оценка применимости ОС *Linux* в качестве ОСРВ

В ОС *Linux* реализованы два варианта дисциплины планирования исполнения задач. Первый вариант, ориентированный на приложения общего назначения, реализует дисциплину разделения времени [7]. Второй вариант, реализующий механизм планирования с фиксированными приоритетами (100 уровней приоритета задач), ориентирован на приложения реального времени. Результаты измерительных экспериментов показывают, что реализация механизмов управления задачами реального времени имеет особенности, ограничивающие использование ОС *Linux* в системах жесткого реального времени [1].

Контекст процесса в ОС *Linux* включает большое количество регистров процессора: регистры общего назначения, регистры математического сопроцессора, регистры блока управления памятью и управляющие регистры. Большой объем контекста задач приводит к большому времени его переключения. При увеличении числа задач реального времени значительно возрастает время выполнения операции перепланирования, что говорит о низкой эффективности планировщика ОС *Linux* в терминах СРВ.

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

Измерения задержек обработки внешних событий для ОС *Linux* также показывают ограниченную пригодность этой ОС для обслуживания приложений РВ. Многие подсистемы ядра *Linux* запрещают обработку внешних прерываний для защиты критических секций. При работе с такими устройствами как жесткий диск или сетевая карта подобный подход приводит к задержкам начала обработки внешних событий до нескольких сотен микросекунд. Есть еще одно обстоятельство, препятствующее своевременной реакции на внешние события, оно состоит в том, что процесс в *Linux* может выполняться либо в режиме пользователя, либо в режиме ядра, при этом высокоприоритетный процесс не может захватить процессор до выхода низкоприоритетного из режима ядра.

Приведенный перечень особенностей ОС *Linux* объясняет необходимость ее доработки в случае возникновения требований поддержки ею приложений жесткого реального времени.

4. Принцип фоновой задачи

В [5] представлен один из возможных подходов к использованию ОС *Linux* в системах жесткого реального времени (*RTLinux*). Суть подхода заключается в том, что *Linux* работает в качестве фоновой задачи, задачи с нулевым приоритетом, под управлением небольшого ядра ОСРВ (система переключается на решение задач *Linux* только тогда, когда нет ни одной задачи РВ, требующей ресурса процессора). При этом коды *Linux* адаптируются к работе на виртуальной машине. Компоненты *Linux* не имеют доступа к аппаратным флагам запрещения/разрешения прерываний, все реальные прерывания перехватываются ОСРВ и, при необходимости, отображаются на состоянии виртуальной машины (рис. 1).

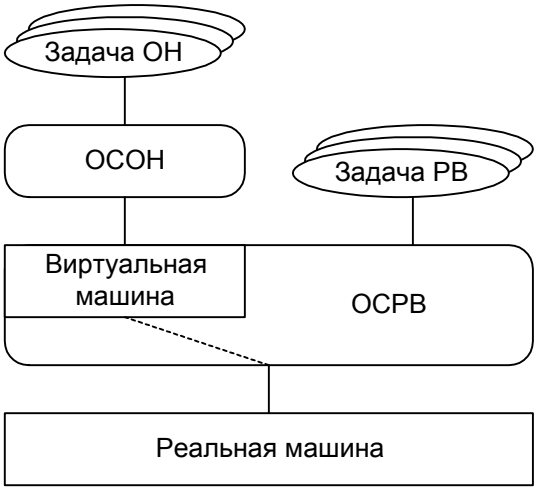

Рис. 1. Принцип фоновой задачи

Все задачи с временными ограничениями выполняются под управлением ядра ОСРВ, под управлением ядра *Linux* решаются только задачи общего назначения. В распоряжении ядра *Linux* остаются такие периферийные устройства, как дисковые накопители, монитор и т.п. Ядром ОСРВ контролируются устройства, функционирующие в режиме реального времени.

Взаимодействие подсистем *Linux* и РВ осуществляется посредством программной эмуляции работы подсистем на разных машинах. Для обмена информацией предусмотрены разделяемая память и механизм очередей элемен-

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

тарных сообщений, для передачи которых не требуется синхронизация.

Основное достоинство систем, реализуемых на базе принципа фоновой задачи — высокое качество работы подсистем РВ. Однако такая модель имеет три существенных недостатка.

Во-первых, ОСРВ должна приостанавливаться для обработки каждого из прерываний ядра *Linux*.

Во-вторых, существует ряд приложений РВ, требующих от ОС исчерпывающего набора сервисов (включающего сервисы для работы с жестким диском, сетевым интерфейсом и т.п.) и обладающих временными характеристиками порядка десяти миллисекунд. Примером такого приложения могут служить мультимедийные пакеты программ. Приложения данного класса ранее успешно выполнялись под управлением ОС *Linux*. ОС, реализующие принцип фоновой задачи, не поддерживают такие приложения, т.к. под управлением ядра *Linux* могут эффективно выполняться только задачи общего назначения, а набор сервисов ядра ОСРВ недостаточно богат.

В-третьих, реализация взаимодействия задач *Linux* и ОСРВ неэффективна и неудобна в использовании. В частности, вопросы синхронизации задач РВ и задач, обслуживаемых ядром *Linux*, при доступе к разделяемым данным приходится решать на прикладном уровне. Автором найдены конструктивные решения, позволяющие устранить эти недостатки.

5. Распределение между ядрами приоритетных уровней прерываний

Первый из отмеченных недостатков эффективно устраняется в случае, если целевая аппаратная платформа имеет два или более уровней прерываний. Примером такой целевой системы является 32-разрядный микропроцессор *M-Core* фирмы *Motorola*. Ключевой для предлагаемого решения особенностью архитектуры микропроцессора *M-Core* является то, что на уровне процессора осуществляется поддержка двух типов прерываний: быстрых (более приоритетных) и обычных (менее приоритетных). Суть предлагаемого решения изображена графически на рис. 2.

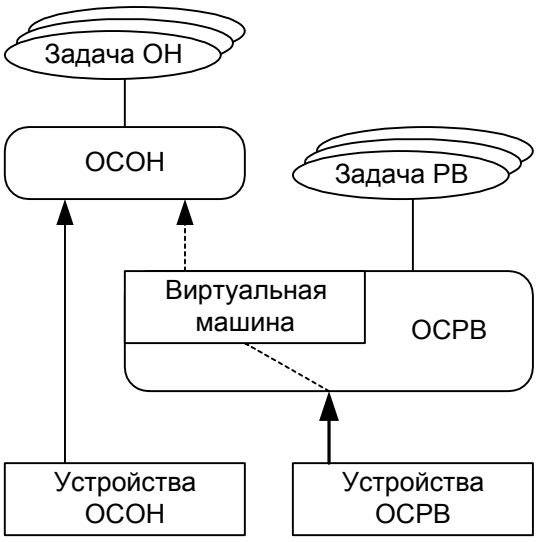

Рис. 2. Сопряжение ядер *Linux* и ОСРВ в системе с несколькими уровнями приоритетов прерываний

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

Внешние устройства делятся на две группы: одну группу составляют устройства, работающие в режиме реального времени (устройства ОСРВ), другую группу составляют устройства, контролируемые подсистемами *Linux* (устройства *Linux*).

Устройства *Linux* могут генерировать только обычные прерывания (*INT*), на рисунке прерывания типа *INT* обозначены тонкими стрелками. Устройства ОСРВ генерируют более приоритетные, быстрые прерывания (*FINT*), на рисунке прерывания типа *FINT* обозначены жирными стрелками.

Выделяются два режима работы системы — режим *Linux* и режим ОСРВ. В режиме работы *Linux* все действия выполняются только с обычными прерываниями, в режиме ОСРВ — только с быстрыми прерываниями.

Таким образом, *Linux*, разрешая или запрещая прерывания, производит соответствующие операции только с флагом обычных прерываний. Во все время работы системы в режиме *Linux* быстрые прерывания разрешены, причем при поступлении быстрого прерывания система безотлагательно переключается на решение задач РВ.

В режиме ОСРВ работа ведется только с быстрыми прерываниями. Т.е. модули ОСРВ, разрешая или запрещая прерывания, производят соответствующие действия только с флагом быстрых прерываний. В течение всего времени работы системы в режиме ОСРВ обычные прерывания запрещены. Переход программного комплекса в режим *Linux* происходит в момент, когда задачи РВ и модули ОСРВ не нуждаются в использовании процессора, при этом флаг разрешения/запрещения обычных прерываний приводится в состояние, в котором он находился на момент переключения системы в режим ОСРВ.

Как видно из рисунка 2, часть быстрых прерываний транслируется ядру *Linux* (см. пунктирную линию, проходящую через ядро ОСРВ). Необходимость в таком решении возникает в случае, когда некоторое устройство совместно используется обеими подсистемами. Примером такого устройства может служить системный таймер. Подобную периферию следует считать устройствами ОСРВ, обработчики, поступающих от них прерываний, должны отображать состояние соответствующего устройства на виртуальной машине ОС *Linux*. Т.о. *Linux* по-прежнему работает на виртуальной машине, управляемой ОСРВ.

Использование решения о распределении ядер ОС по приоритетным уровням процессора возможно также для аппаратных платформ имеющих только один приоритетный уровень прерываний (рис. 3). В этом случае, логика обслуживания запросов на прерывания с разными приоритетами реализуется программно.

Аппаратные действия по обработке прерываний от устройств *Linux* запрещены. Специальная задача РВ — менеджер виртуальной машины — периодически проверяет, есть ли установленные флаги запросов на прерывания от устройств *Linux*, и если есть, то выставляет соответствующие биты в ячейке памяти виртуальной машины, выполняющей функции регистра флагов. Таким образом, подсистема *Linux* работает с периферийными устройствами по опросу.

Следует обратить особое внимание на выбор временных параметров Менеджера Виртуальной Машины (МВМ). Период активации задачи необходимо выбирать с таким расчетом, чтобы не происходило потери запросов на прерывания. В случае ОС *Linux* оценкой периода МВМ может служить максимальная продолжительность работы ядра при запрещенных прерываниях (около 0,5 мс). МВМ является обязательной частью приложения реального времени. Наравне с остальными задачами, МВМ должен назначаться приоритет в соответствии с

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

его временными параметрами. Для менеджера виртуальной машины также должен выполняться анализ выполнимости [2].

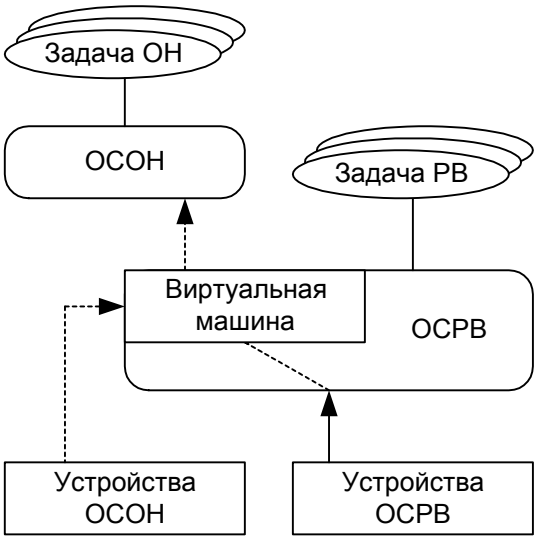

Рис. 3. Сопряжение ядер *Linux* и ОСРВ в системе с одним уровнем приоритетов прерываний

6. Интегрированная схема планирования задач *Linux* и ОСРВ

Согласно принципу фоновой задачи, ядро *Linux* имеет нулевой приоритет в подсистеме реального времени. Это означает, что все приоритетные уровни планировщика Linux отображаются на нулевой приоритетный уровень планировщика ОСРВ. Задачи РВ, выполняющиеся под управлением ядра *Linux*, в общем случае не могут планироваться эффективно, т.к. они дискриминируются задачами ОСРВ.

Значительное повышение эффективности планирования приложений реального времени *Linux* обеспечивает переход к единой шкале приоритетов, образованной пересекающимися приоритетными шкалами ядер *Linux* и ОСРВ (рис. 4). Нулевой приоритет принимается наименьшим в системе. Значению *p1* соответствует первый приоритетный уровень, разделяемый задачами *Linux* и ОСРВ, значению *p2* — последний. Приоритетные уровни от *p2* до максимального (*max*) занимают только задачи, выполняющиеся под управлением ядра ОСРВ.

Распределение всех задач в системе по уровням единой приоритетной шкалы производится по общему правилу, и, как видно из рисунка 4, часть задач *Linux* может на равных конкурировать с задачами ОСРВ. Со стороны ОСРВ на уровнях от *p1* до *p2* располагаются задачи РВ с невысокими требованиями на время отклика, имеющие тесные связи с аппаратурой или другими задачами ОСРВ.

Для организации единой шкалы приоритетов в рамках двухуровневого планирования (ядро ОСРВ планирует ядро *Linux*, планирующее свои задачи) необходимо, чтобы в каждый момент времени приоритет *Linux* в подсистеме реального времени был равен приоритету текущей задачи *Linux*. Ядро *Linux* при переключении контекста или изменении приоритета текущей задачи посредством системного вызова ядра ОСРВ назначает себе приоритет в соответствии с приоритетом текущей задачи. Для повышения эффективности ядру *Linux* сле-

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

дует обрабатывать прерывания на приоритетном уровне, расположенном выше, чем максимально возможный уровень задачи *Linux*.

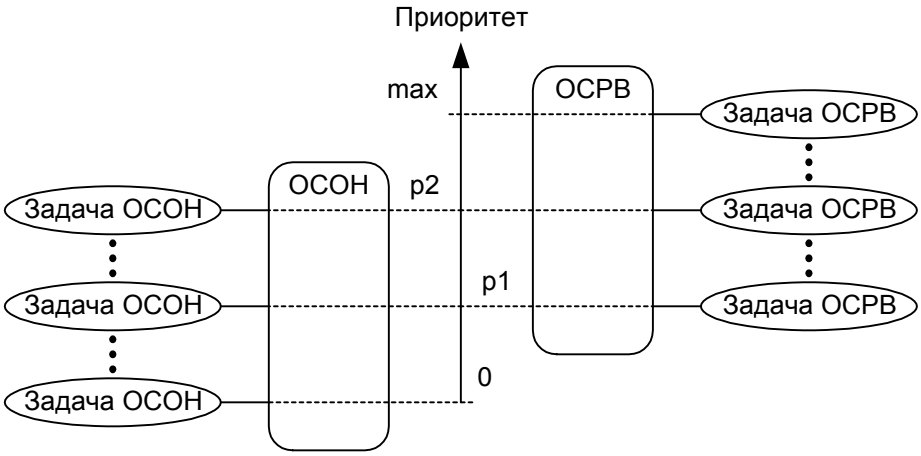

Рис. 4. Интегрированная схема планирования задач *Linux* и ОСРВ

Использование интегрированной схемы планирования задач *Linux* и ОСРВ также позволяет частично устранить третий недостаток подхода с фоновой задачей (ускорить и упростить взаимодействие задач *Linux* и ОСРВ). Благодаря единой приоритетной шкале задачи *Linux* могут на общих основаниях осуществлять синхронизацию с задачами ОСРВ при доступе к разделяемым данным с использованием протоколов, основанных на понятии порога [2, 4]. Доступ задач *Linux* к соответствующим сервисам ОСРВ должен предоставляться ядром *Linux*.

Трудоемкость реализации предлагаемых решений невысока. Вместе с тем, они позволяют эффективно объединить задачи ОС *Linux* с задачами жесткого реального времени.

7. Показатели эффективности сопряжения задач *Linux* и ОСРВ

При реализации рассмотренных принципов сопряжения задач жесткого реального времени с программами ОС *Linux* эффективность обслуживания задач ОСРВ никак не зависит от особенностей организации *Linux* — качество сервисов, предоставляемых задачам жесткого реального времени зависит только от характеристик ядра ОСРВ. Независимость разработки ядер ОС позволяет использовать накопленный опыт в разработке ОСРВ с повышенными требованиями к предсказуемости работы. Экспериментальный вариант системы с предлагаемой архитектурой, имеет фиксированное, не зависящее от количества задач в системе, время создания и уничтожения задач РВ: 12 и 5 мкс, соответственно. Задержка начала обработки внешнего события составляет несколько микросекунд. Следует заметить, что эти данные были получены на аппаратной платформе *M-Core*, значительно уступающей по производительности современным персональным компьютерам и рабочим станциям.

Основным недостатком известной реализации *RTLinux* [5] является высокая сложность и низкая эффективность взаимодействия между задачами *Linux* и задачами ОСРВ. Предлагаемый вариант сопряжения *Linux* с ОСРВ позволяет организовать более эффективную координацию действий и информационных потоков. Для сравнения двух механизмов взаимодействия был проведен ряд измерительных экспериментов (рис. 5).

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

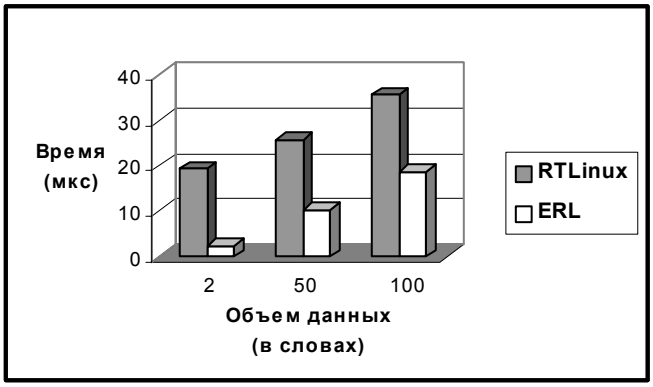

Рис. 5. Продолжительность операций передачи данных

В ходе экспериментов измерялось время необходимое для копирования блока данных из адресного пространства *Linux* в адресное пространство ОСРВ (выполнение этой операции с блоком, размер которого превышает слово процессора, требует синхронизации). Как видно из рисунка использование предложенного авторами механизма взаимодействия (*ERL — Enhanced Real-time Linux*) сокращает время выполнения операции на 17 мкс.

8. Заключение

Сервисы свободно-распространяемой ОС *Linux* не рассчитаны на поддержку задач жесткого реального времени. Одним из вариантов добавления к ОС *Linux* возможности работы в режиме жесткого РВ является использование двухядерной архитектуры, в которой доступ *Linux* к ресурсу процессора контролируется ядром ОСРВ. Переход к интегрированной схеме планирования задач ядер *Linux* и ОСРВ позволяет значительно повысить эффективность планирования и синхронизации смешанных приложений реального времени. Распределение ядер ОС по приоритетным уровням процессора позволяет сократить издержки связанные с виртуализацией аппаратуры.

Литература

[1] *В. В. Никифоров, М. В. Данилов, М. В. Осипов*. Сопряжение пакетов программ общего назначения с задачами жесткого реального времени // Программные продукты и системы. — 2000. — №4. — с. 19-24.

[2] *В. В. Никифоров, М. В. Данилов*. Статическая обработка спецификаций программных систем реального времени // Программные продукты и системы. — 2000. — №4. — с 13-18.

[3] *В. В. Никифоров, В. А. Павлов, В. В. Сидельников*. Интеграция систем реального времени с программными средствами общего назначения // Программные продукты и системы. — 1999. — №4. — с. 15-20.

[4] *М. В. Данилов*. Асимметричный протокол приоритетных порогов // Программные продукты и системы. — 2000. —№4. —с. 33-36.

[5] *V. Yodaiken*. The RTLinux Maniffesto. New Mexico Institute of Technology. `http://www.rtlinux.org`.

[6] RTOS Evaluation Program. Can NT 4 be used as an RTOS? Real-Time Consult, 1998. `http://www.realtime-info.be`.

[7] Linux MAN. `http://www.linux.org`.

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru