УДК 681.325.5–181.4

ЭКСПЕРИМЕНТАЛЬНАЯ ОЦЕНКА БЫСТРОДЕЙСТВИЯ ВЕРСИЙ ОС LINUX

Н. В. Гуцалов,
аспирант
Санкт-Петербургский институт информатики и автоматизации РАН (СПИИРАН)

Параметры быстродействия относятся к ряду наиболее важных характеристик операционных систем реального времени (ОСРВ). В статье рассматриваются методы экспериментальной оценки быстродейсвия ОСРВ, приводятся результаты применения этих методов для оценки быстродействия версий ОС Linux, ориентированных на поддержку приложений реального времени. Рассматриваются принципы организации и систематизации измерений. Рассмотрение ведется с учетом рекомендаций стандарта POSIX, относящихся к экспериментальной оценке быстродействия ОС.

Системы реального времени (СРВ) имеют жесткие ограничения сроков выполнения заданий, реализуемых аппаратно-программными комплексами [1]. Отсюда вытекают специфические требования, предъявляемые к операционным системам реального времени (ОСРВ). Стандарт POSIX [ISO/IEC 9945-1: 1996] определяет ключевое свойство ОСРВ как способность операционной системы предоставить необходимый уровень сервиса при условии ограниченного, в смысле конкретного значения, времени отклика.

Операционные системы общего назначения (ОСОН) и, в частности, ОСОН Linux изначально разрабатывались без учета требований, предъявляемых к СРВ. Вместе с тем, в последнее время предпринимаются усилия по адаптации ОСОН к условиям работы в системах реального времени. Особое внимание привлекает появление версий ОС Linux, ориентированных на использование в СРВ. Это обусловлено следующими факторами:

— ОСОН Linux является свободнораспространяемой, что существенно сказывается на стоимости конечных устройств при использовании в них модификаций ОС Linux, поддерживающих задачи РВ;

— коды ядра ОС Linux открыты для модификации, что существенно упрощает адаптацию системы к конкретным нуждам;

— модификации ОС Linux для поддержки задач РВ совмещают в себе свойства ОСОН и ОСРВ, т. е. наряду с обеспечением временных ограничений, предъявляемых к системе, предоставляют широкий спектр библиотек и приложений, которые доступны для использования как в ходе разработки, так и в действующей системе;

— ОС Linux соответствует стандарту POSIX, что позволяет разрабатывать высокомобильные приложения.

Наиболее перспективным методом адаптации ОС Linux к требованиям СРВ является оснащение базовой версии ОС Linux дополнительным ядром реального времени. Разработка такой двухъядерной ОС выполнена, в частности, в Миланском политехническом институте: специальное ядро реального времени RTAI (Real-Time Application Interface) интегрировано с ядром Linux [2].

Для определения возможностей использования ОС Linux в качестве ОСРВ необходимо располагать оценками значений параметров быстродействия ОС. Наиболее надежным способом оценки быстродействия ОСРВ является проведение измерительных экспериментов. Ниже изложены методы экспериментальной оценки быстродействия, применявшиеся автором для получения параметров быстродействия RTAI, приводятся результаты измерений, сравниваются параметры быстродействия RTAI с аналогичными параметрами базовой версии ОС Linux.

Основные параметры быстродействия ОСРВ

Параметры быстродействия ОСРВ можно разделить на две группы:

1) характеристики реактивности;
2) характеристики производительности.

Параметры первой группы характеризуют скорость реакции системы на внутренние и внешние события. Параметры второй группы характеризуют эффективность реализации сервисных функций ОСРВ.

Реактивность ОСРВ характеризуется длительностью интервалов задержки реакции системы на возникновение внешних и внутрисистемных событий [3]. Внешние события регистрируются внешними устройствами и приводят к генерации сигналов внешних прерываний. Внутренние события фиксируются программно и могут привести к изменению системных состояний выполняемых задач.

Задержкой обработки прерывания называется время между появлением сигнала прерывания t_s на входе контроллера прерываний процессора и моментом t_a начала исполнения обработчика события, вызвавшего это прерывание (рис. 1) [4]. Ин-

тервал задержки обработки прерывания содержит четыре составляющих:

1) (t_s, t_i) — завершение действий ОС при запрещенных прерываниях;

2) (t_i, t_p) — аппаратные действия системы по сохранению контекста прерываемой программы и передаче управления обработчику прерывания;

3) (t_p, t_j) — регистрация прерывания и выполнение вспомогательных действий ОС;

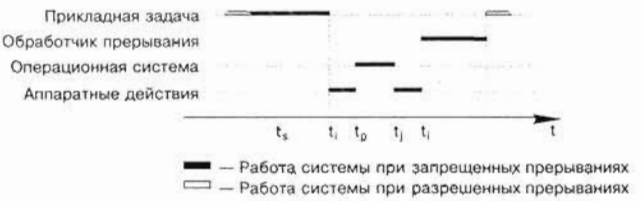
4) (t_j, t_a) — аппаратные действия по передаче управления пользовательскому обработчику прерываний.

Составляющая (t_s, t_i) может отсутствовать в структуре задержки, если прерывание произошло во время работы системы при разблокированных прерываниях. Отметим однако, что величина максимальной задержки обработки прерывания в значительной мере зависит от максимально возможной продолжительности интервала (t_s, t_i).

Сохранение контекста процессора и передача управления аппаратно зарегистрированному обработчику прерываний реализованы на уровне микрокоманд процессора, поэтому продолжительность этих действий невелика и варьируется несущественно, но во время выполнения этих действий может иметь место промах в кэш. Практически все современные платформы имеют в своем составе несколько уровней кэш. Использование кэш значительно увеличивает среднее быстродействие системы, но когда в кэш возникает промах, время выполнения операции может быть значительно больше, чем в случае, когда кэш не используется. Это вызвано тем, что перед загрузкой в кэш требуемых данных необходимо сохранить в обычной памяти блок, вытесняемый из кэша. При этом, если в системе используется многоуровневый кэш, то в худшем случае может произойти цепочка промахов на всех уровнях кэш, что приведет к значительным задержкам. Для устранения этого эффекта применяются специальные приемы [5], например, часть кода ОС, использующаяся при обработке прерываний, блокируется в кэш-памяти. Но зависимость задержки от размера кода, блокированного в кэш, достаточно сложная, а алгоритм выбора участков кода четко не регламентирован.

Регистрация прерывания операционной системой и передача управления пользовательскому обработчику прерываний имеет место в том случае, когда обработка прерываний производится операционной системой, и только после этого управление передается прикладному обработчику. Более подробно структура интервала задержки рассмотрена в [3].

Производительность системы характеризуется продолжительностью выполнения сервисных функций ОС. При подготовке измерений производительности системы целесообразно учитывать рекомендации стандарта POSIX, относящиеся к экспериментальной оценке ОС. В стандарте указано, что метрики, характеризующие быстродействие системы, должны содержать среднее значение и худший случай с условиями, в которых это значение получено. Набор условий определяется как использованная методология, аппаратная и программная конфигурация, параметры вызова, загрузка системы, начальные условия и другая информация, с помощью которой можно продублировать измерительный эксперимент. Для функций, время выполнения которых зависит от параметров вызова, необходимо указывать набор результатов с соответствующими параметрами вызова. Если значение может отклоняться от приведенного в документации, необходимо указывать условия, при которых будут наблюдаться эти отклонения.



■ ***Рис. 1.*** *Структура интервала задержки обработки прерывания*

В стандарте описываются метрики основных групп сервисов: обмен сигналами, синхронный ввод-вывод, асинхронный ввод-вывод, операции с семафорами, операции с мютексами и условными переменными, доступ к виртуальной памяти процесса, блокированной в ОЗУ, доступ к разделяемой памяти, планирование задач, часы и таймеры, передача сообщений, управление потоками, отмена потоков. Из этих групп сервисов особый интерес представляют операции с семафорами и планирование задач.

Операции с семафорами. В разделе содержится семь метрик: *освобождение семафора* (для условий: *нет ожидающих этого семафора, только низкоприоритетные задачи ожидают освобождения этого семафора, высокоприоритетная задача ожидает освобождения семафора*), *захват семафора* (для условий: *семафор свободен, семафор занят*) и *условный захват семафора* (для условий: *семафор свободен, семафор занят*).

Ниже приведены данные, характеризующие быстродействие ОС Linux, для трех типов параметров из рекомендованных стандартом: продолжительность захвата свободного семафора, продолжительность освобождения семафора при отсутствии ожидающих его освобождения, продолжительность захвата занятого семафора. Измерение продолжительности захвата занятого семафора в исследовании производится при различном количестве задач в очереди ожидания, что вызвано рекомендацией стандарта, относящейся к оценке производительности планировщика. В дополнении к рекомендованным метрикам указываются продолжительность создания и удаления семафора.

Планирование. Все метрики, характеризующие управление планированием, должны быть приведены для двух дисциплин планирования: SCHED_FIFO и SCHED_RR. При максимально возможном количестве задач в системе больше 129 для однопроцессорной платформы измерительные эксперименты должны строиться для следующих количеств балластных процессов в дополнение к двум измерительным: 0, 1, 33, 65, 129. В качестве параметра производительности планировщика задач в исследовании используется время переключения контекста задачи.

Методика измерений

Существует несколько способов организации измерений параметров быстродействия операционных систем. Часто для этих целей используется специальное оборудование, которое подключается к исследуемой платформе и протоколирует события, протекающие в системе, для последующей обработки. Основное преимущество данного способа состоит в том, что подобное оборудование является внешним по отношению к системе и не оказывает влияния на ее работу. К недостаткам использования специальной аппаратуры относятся высокая стоимость и трудоемкость освоения. Ниже рассматриваются программные методы измерений.

Производительность системы. Все сервисы ОС можно разделить на две большие группы: а) сервисы, выполнение которых не приводит к переключению контекста; б) сервисы, выполнение которых сопровождается переключением контекста. Способы измерения времени выполнения этих сервисов несколько отличаются друг от друга (рис. 2).

Метод измерения времени выполнения сервиса ОС состоит в следующем. Перед выполнением оцениваемого сервиса фиксируется момент времени t_s, а после завершения его выполнения — момент t_f. Тогда разность $(t_f - t_s)$ и будет определять время выполнения данного сервиса ОС. Отличие метода измерения состоит в том, что при оценке сервисов, не приводящих к переключению контекста, моменты начала и конца выполнения сервиса фиксируются одной измерительной задачей; в то время как при оценке сервисов, приводящих к переключению контекста, момент начала выполнения сервиса ОС фиксируется задачей, владеющей процессором к началу эксперимента, а момент окончания выполнения сервиса — задачей, получающей управление в результате его выполнения.

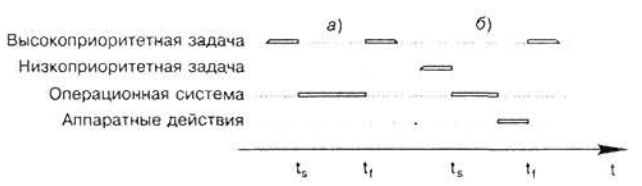
Так как в каждом отдельно взятом эксперименте условия проведения будут отличаться от других (состояние системы постоянно изменяется и воспроизвести его точно не представляется возможным), результаты различных экспериментов в общем случае будут отличаться друг от друга. Для получения достаточно объективной статистики необходимо провести серию экспериментов с тем, чтобы после проведения статистической обработки результатов получить представление о параметрах их распределения. Минимальное значение, полученное в ходе серии экспериментов, дает оценку времени выполнения сервиса в наиболее благоприятных условиях; среднее значение можно оценивать как наиболее вероятный случай, а максимальное время оценивает продолжительность выполнения сервиса в том случае, когда на результат оказывают влияние такие факторы, как поступление запроса на прерывание во время измерительного эксперимента, промах или цепочка промахов в кэш, конфликты доступа к памяти или шине, другие конфликты в системе. Максимальная продолжительность выполнения сервиса может определяться как каждым из этих факторов в отдельности, так и их сочетанием. При оценке производительности основной интерес представляет среднее значение как наиболее характерное.

Накладные расходы на регистрацию текущего времени. Рис. 2 соответствует случаю, когда измерительное приложение имеет возможность моментально получить значение текущего времени. В действительности операция получения текущего времени имеет определенную продолжительность (рис. 3).

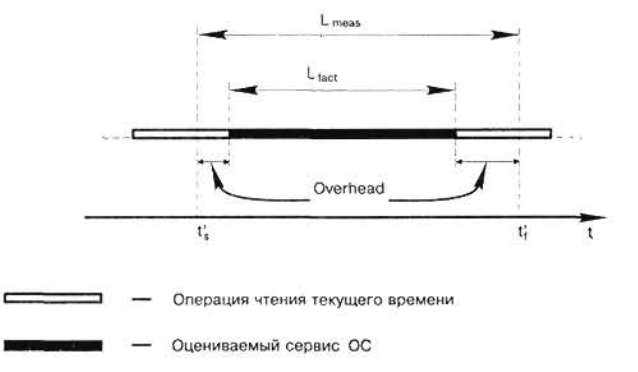
Полученное в ходе измерительного эксперимента значение разности $(t_f' - t_s')$ превышает действительное время выполнения сервиса ОС. Значение, возвращаемое функцией *дать_время*, соответствует неизвестной фазе операции чтения текущего времени. Поэтому в экспериментальных данных присутствует составляющая, равная сумме накладных расходов на регистрацию момента начала и конца эксперимента. Но необходимости вычислять эти накладные расходы по отдельности нет. Из рисунка видно, что сумма накладных расходов начала и конца эксперимента составляет продолжительность выполнения операции регистрации текущего времени. Для того чтобы оценить величину накладных расходов, достаточно провести следующий эксперемент (рис. 4).

Если двумя последовательными вызовами функции *дать_время* зафиксировать моменты времени t_s'' и t_f'', их разность $(t_f'' - t_s'')$ будет составлять суммарные накладные расходы на регистрацию текущего времени. Так как при проведении эксперимента в этот интервал могут вмешиваться посторонние события, за оценку издержки измерений принимается минимум $min\{(t_f'' - t_s'')\}$.

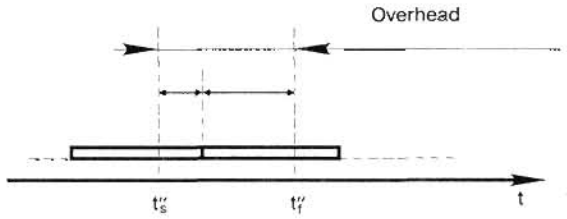
Оценка задержки реакции на внешние прерывания. Для оценки величины задержки обработки прерывания необходимо, чтобы были известны два момента времени: момент поступления запроса на обработку прерывания и момент активизации обработчика прерывания. Если момент активизации обработчика прерываний фиксируется первой операцией, выполняемой измерительным обработчиком, то истинный момент поступления запроса на обработку прерывания без применения специальных аппаратных средств точно зафик-



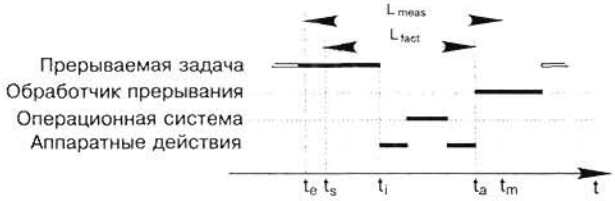
■ ***Рис. 2.*** *Измерение времени выполнения сервисов ОС*



■ ***Рис. 3.*** *Ошибка измерений*



■ ***Рис. 4.*** *Измерение издержки*



■ ***Рис. 5.*** *Измерение задержки обработки прерывания*

сировать невозможно. Но этот момент можно предсказать с достаточно высокой точностью, если в качестве источника измерительных прерываний использовать независимый периодический таймер, обеспечивающий генерацию запросов прерывания с заданной частотой. Независимость измерительного таймера понимается в том смысле, что он не должен использоваться операционной системой во избежание нежелательной синхронизации действий операционной системы и измерительного приложения. Такой независимый таймер имеется в составе большинства аппаратных платформ, используемых в СРВ. Тогда измерительный эксперимент будет выглядеть так, как показано на рис. 5.

Специфическими являются моменты времени:

— t_e — ожидаемое время прихода запроса на прерывание;

— t_m — измеренное значение времени активизации обработчика.

Если запрос на прерывание поступает в момент, когда прерывания разрешены, отрезок (t_s t_i) является вырожденным.

Ошибкой измерений является сумма отрезков (t_e t_s) и (t_a t_m) с учетом знаков (прерывание может произойти как раньше предполагаемого момента времени, так и позже). Различие между истинным моментом t_a активизации обработчика прерываний и величиной t_m, полученной в результате измерения, обусловлено затратами на считывание времени. Разность $|t_s - t_e|$ обусловлена в том числе относительным дрейфом частот генератора измерительных прерываний и генератора тактов процессора. Если стабильность генераторов не хуже, чем 10^{-4}, то для того, чтобы эта разность не превышала десятой доли микросекунды, достаточно один раз в миллисекунду уточнять привязку фазы периодических измерительных прерываний к источнику измерительной информации. Вместе с тем, существует способ оценки максимальной задержки, не требующий таких уточнений. Этот способ наряду с другими особенностями измерений задержки обработки прерываний подробно описан в [3].

Значения параметров быстродействия версий ОС Linux

Описанные выше методы были применены для оценки быстродействия версий ОС Linux, ориентированных на поддержку задач реального времени, а также (для сравнения) оценки быстродействия базовой версии ОС Linux. В качестве ОС реального времени были выбраны разработки, опирающиеся на упоминавшиеся выше ядра реального времени RTAI. Эти разработки представляют собой совокупность ядра реального времени и «заплатки» для ядра базовой версии ОС Linux. Модифицированное ядро базовой версии ОС Linux вместе с ядром реального времени составляет двухъядерную операционную систему [2]. Ядро Linux в такой системе работает как задача ядра РВ с наименьшим приоритетом — когда в подсистеме РВ нет других, готовых к исполнению задач, управление получает ядро Linux. Взаимодействие подсистем осуществляется посредством механизмов очередей сообщений и разделяемой памяти. Такая архитектура позволяет конструировать системы взаимодействующих задач, часть из которых, имеющая ограничения на сроки выполнения, работает в подсистеме РВ, а задачи, не имеющие ограничений на сроки выполнения и использующие сервисы ядра базовой версии ОС Linux, работают под управлением Linux.

Для сравнения взяты две версии данной системы: RTAI-22.2.4 и RTAI-24.1.8. Первая версия была разработана в 2000 г. и предназначалась для ядра Linux 2.2.16. Вторая (вышла полтора года спустя) предназначена для работы с ядром Linux 2.4.17. За это время архитектура системы была усовершенствована, и сравнение значений параметров быстродействия показывает, как эволюция архитектуры RTAI улучшила временные характеристики системы.

Условия проведения экспериментов. Измерения производились на персональном компьютере Compaq Deskpro EN на базе процессора Intel Pentium II 266 MHz под управлением операционной системы RedHat Linux 7.2 с ядрами 2.2.16 и 2.4.17 соответственно. В качестве источника информации о текущем времени использовался регистр TSC (Time Stamp Counter), который присутствует во всех процессорах Pentium и представляет собой счетчик тактов процессора. Источником измерительных прерываний является RTC (Real-Time Clock) — внешний по отношению к процессору таймер, настроенный на генерацию прерываний с частотой 8192 Гц. Для измерения задержки обработки прерываний использовалась техника измерений с неизвестной фазой измерительных прерываний, описанная выше. Измерения задержки обработки прерываний выполнялись для трех вариантов загрузки системы:

1) минимальная нагрузка (задача ядра реального времени, исполняющая пустой цикл, блокирует активность Linux);
2) драйверы Linux (поиском утилитой badblocks поврежденных секторов на флоппи-диске; измерительное приложение запускается спустя 5 с после начала работы badblocks);
3) задачи реального времени (200 активно взаимодействующих между собой задач ядра реального времени).

■ ***Таблица 1.*** *Результаты измерения задержки обработки прерываний*

Нагрузка	Ядро		
	RTAI-22.2.4	RTAI-24.1.8	Linux-2.2.15
Минимальная нагрузка	6,0	5,1	12,6
Драйверы Linux	16,5	9,7	481,5
Задачи реального времени	30,5	9,2	105,5

■ ***Таблица 2.*** *Затраты процессорного времени на переключение контекста задачи*

Количество балластных задач	Ядро		
	RTAI-22.2.4	RTAI-24.1.8	Linux-2.2.15
0	0,4	0,6	05,0
1	0,4	0,6	05,1
33	0,9	0,6	12,4
65	1,4	0,6	26,5
129	5,3	0,6	61,5

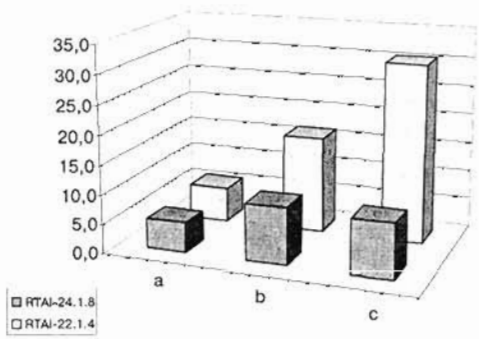
Для оценки параметров быстродействия рассматриваемых версий ОС использовались идентичные измерительные приложения.

Задержка обработки прерываний. В табл. 1 представлены значения задержек обработки прерываний для обеих версий RTAI с тремя видами нагрузок.

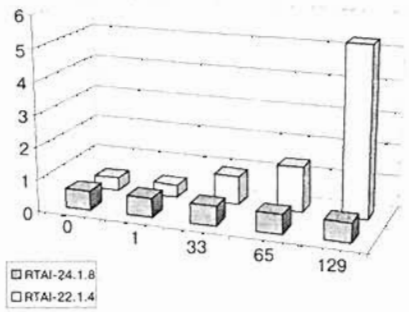
Из приведенных данных можно сделать следующие выводы. Во-первых, несмотря на то, что принцип взаимодействия ядра Linux с ядром реального времени остался без изменений, из-за модификации самого ядра Linux максимальная задержка при нагрузке Linux значительно снизилась. Во-вторых, максимальная задержка обработки прерывания при нагрузке в виде задач реального времени снизилась более чем в три раза. Это объясняется тем, что планировщик RTAI претерпел серьезную модификацию, что привело к значительному сокращению продолжительности промежутков работы системы при запрещенных прерываниях. В версии RTAI-22.2.4 планировщик реального времени хранил все задачи, независимо от их состояния, в едином неупорядоченном по приоритетам списке. Каждый раз при выборе задачи, которой должно быть передано управление, планировщик просматривал весь список задач. Поэтому время перепланирования тем больше, чем больше задач находится в системе. А так как перепланирование происходит при запрещенных прерываниях, подобная организация влияет не только на время переключения контекста (см. ниже), но и на задержку обработки прерываний.

Эффект от усовершенствования архитектуры планировщика задач реального времени наглядно представлен на рис. 6.

Переключение контекста. Особенности архитектуры планировщика непосредственно влияют на длительность переключения контекста задачи. По



■ ***Рис. 6.*** *Максимальные значения задержки обработки прерываний*



■ ***Рис. 7.*** *Зависимость среднего времени переключения контекста от количества задач в системе*

тем же причинам, что и в случае с обработкой прерываний, среднее время переключения контекста в версии RTAI-22.2.4 тем больше, чем больше задач находится в системе (табл. 2 и рис. 7).

Из таблицы и диаграммы видно, что в версии RTAI-24.1.8 зависимость времени переключения контекста от количества задач в системе устранена.

Семафорные операции. Затраты на выполнение семафорных операций для версий RTAI-22.2.4 и RTAI-24.1.8 приведены в табл. 3.

Полученные результаты закономерны: при том, что издержки на выполнение сервисов ОС, которые не вызывают переключения контекста, в оцениваемых версиях RTAI практически не различаются, издержки на исполнение функций ОС, приводящих к переключению контекста (к которым и относится захват занятого семафора, так как задача в этом случае блокируется в очереди ожидания

■ ***Таблица 3.*** *Затраты процессорного времени на выполнение семафорных операций*

Операция			Ядро		
			RTAI-22.2.4	RTAI-24.1.8	Linux-2.2.15
Создание семафора			0,1	0,2	2,8
Удаление семафора			0,2	0,2	2,4
Захват свободного семафора			0,3	0,3	2,4
Освобождение семафора (без ожидающих)			0,2	0,3	2,4
Захват занятого семафора	Длина очереди	0	0,8	0,7	3,0
		1	0,9	0,7	3,1
		33	1,5	0,7	6,3
		65	1,9	0,7	6,3
		129	3,4	0,7	7,5

семафора), в версии RTAI-22.2.4 зависят от числа балластных задач.

Примечательно, что одно и то же свойство ОС (отсутствие упорядоченного списка готовых задач) проявляется в трех разных типах характеристик быстродействия системы.

Заключение

Применение описанных методов измерения параметров быстродействия ОС позволяет оценивать временные характеристики ОСРВ, которые предоставляют возможность проводить сравнительную оценку быстродействия различных модификаций ОС Linux с целью поддержки задач РВ. Как показано выше, построение двухъядерной ОСРВ на базе ОС Linux позволяет добиться значительного улучшения характеристик системы по сравнению с базовой версией ОС Linux. Заметим, однако, что базовая версия ОС Linux разрабатывалась для решения задач, не связанных с требованиями реального времени, поэтому ее сравнительно невысокое быстродействие не следует расценивать как показатель низкого качества системы. Вместе с тем, приведенные выше результаты показывают, что использование дополнительного ядра реального времени позволяет добиться значительного улучшения характеристик быстродействия ОС для задач, которые не нуждаются в полном наборе сервисов, предоставляемых ядром Linux, но жестко ограничены по срокам выполнения. Сочетая в себе эти качества, модификации ОС Linux позволяют строить программные системы, требующие функциональности полноценной ОС, с одной стороны, и отвечающие требованиям реального времени, с другой. Такое сочетание открывает возможность использования ОС Linux в системах, которые ранее могли работать только под управлением специализированных ОСРВ.

Литература

1. **Kopetz H.** Real-Time Systems. Design Principles for Distributed Embedded Applications. — Dordrecht: Kluwer Academic Publishers, 1997.
2. **Cloutier P., Mantegazza P., Papacharalambous S., Soanes I., Hughes S.** DIAPM-RTAI position paper. (*http://www.rtai.org*).
3. **Никифоров В. В., Гуцалов Н. В.** Методы измерения реактивности систем реального времени // Программные продукты и системы. — 2001. — № 4.
4. **Straumann T.**, Open Source Real Time Operating Systems Overview. (*http://www.slac.stanford.edu/econf/C011127/WEBI001.pdf*).
5. **Dejan Bucar** Reducing Interrupt Latency using the Cache, Royal Institute of Technology. — Sweden. — 2001.

УДК 681.325.5

ПРИМЕНЕНИЕ ШИНЫ CAN–BUS В РАСПРЕДЕЛЕННЫХ СИСТЕМАХ СБОРА И ОБРАБОТКИ ИНФОРМАЦИИ В РЕАЛЬНОМ МАСШТАБЕ ВРЕМЕНИ

С. Т. Хвощ,
д-р техн. наук
А. В. Луковкин,
инженер
А. Г. Лютов,
инженер
ЗАО «Электронная компания «ЭЛКУС»

Рассматриваются характеристики типовых, серийно выпускаемых CAN-узлов для реализации систем реального времени, в том числе бортовых с распределенным энергоснабжением, а также встраиваемых. Рассматриваются и описываются аппаратные средства сопряжения интерфейса CAN с MIL-STD-1553B и другими специализированными шинами и магистралями информационно-управляющих систем.