ИССЛЕДОВАНИЕ АЛГОРИТМОВ И МЕТОДОВ БАЛАНСИРОВКИ НАГРУЗКИ И ПОСТРОЕНИЕ МОДЕЛЕЙ ДЛЯ СЕТЕЙ МАССОВОГО ОБСЛУЖИВАНИЯ

К. А. Айдаров[1], Г. Т. Балакаева[1]

[1] *Казахский Национальный университет им. аль-Фараби, 050043, Алматы, Казахстан*

УДК 519.872

Данная работа описывает алгоритмы балансировки для внешних сервисов с неспециализированными клиентами, используемых в настоящих промышленных центрах обработки данных. Простейший пример такого сервиса и клиента — это веб-сервер и браузер. Исследован алгоритм балансировки нагрузки через приоритетное планирование. По результатам реализации модели для кластерной системы получены оценки уменьшения эффекта осцилляции между ее перегруженным и нормальными состояниями.

Ключевые слова: балансировка нагрузки, дискретно-событийное моделирование, модельно-ориентированное проектирование

Введение

При планировании многоядерного ресурса, такого как микроконтроллер встроенной системы, желаемым преимуществом является эффективное использование ресурсов, а также контроль над управлением, вкупе со строгими ограничениями по времени для достижения желаемого уровня производительности. Чтобы оптимизировать весь проект системы, во-многих случая необходим анализ компромиссов, который может быть сделан на основе предположения, что функциональность планирования на совместных используемых ресурсах может быть удовлетворена с целью повышения производительности управления.

Работа основана на использовании набора инструментов модельно-ориентированного проектирования, который основывается на фреймворке Simulink [1] от компании Mathworks. Суть модельно-ориентированного подхода описана в [2]. Интеграция данных инструментов позволяет проводить непосредственный анализ влияния параметров планировщика на производительность управления в рамках системы управления целиком. Данный анализ включает прямую оценку соответствия производительности системным требованиям. Кроме того, при необходимости, лучшая производительность управления может быть достигнута, изменением управляющей функциональности или проектированием системы в оригинальном представлении.

Решение основывается на вышеупомянутой среде моделирования дискретных событий с соответствующим механизмом исполнения для динамического планирования событий. Работа, представленная здесь иллюстрирует поддерживаемые методы анализа производительности для различных стратегий управления, которые могут быть реализованы по-разному.

Предыдущие решения в этой области на основе модельно-ориентированного проектирования и Simulink включают TRES [3] и TrueTime [4]. Оба обеспечивают функциональность, достаточную для моделирования многозадачного планирования и сетей во встроенных системах управления.

TRES является модульным фреймворком который добавляет задачи реального времени и сетевые модели к существующим моделям Simulink. Это налагает дополнительные трудности при моделировании отказоустойчивых и производительных решений. Кроме того, для существующей модели Simulink, где контроллеры определены как графические блоки с выполнением и расчетами спецификации, этот подход требует существенного перепроектирования и часто не практичен.

В TrueTime блок Kernel моделирует одноядерный компьютерный узел с универсальным ядром в реальном времени, A/D и цифро-аналоговые преобразователи, внешние вводы и сетевые интерфейсы. Ограничением TrueTime является моделирование задачи управления, так как в режиме реального времени основные блоки

должны быть записаны на языках MATLAB или C++. В итоге это приводит к ограниченному доступу к набору блоков Simulink.

1 Модель

Платформа в среде Simulink предложена для графического моделирования многоядерного планировщика с ресурсами. Платформа реализована при помощи блока планировщика, являющейся частью модуля SimEvents для Simulink с дискретно-событийной семантикой [5]. Данная платформа включает моделирование планирования эффектов (например, задержки, время ожидания), и исследуют пространство исполнения задач, через анализ влияния планирования выбора на производительность управления. Платформа наложена на алгоритм управления и позволяет плавно перейти к реализации алгоритма через автоматические технологии генерации кода.

Рисунок 1а показывает модель Simulink, использующую эту платформу, состоящую из блока Scheduler (Планировщик) и связанный Контроллер алгоритмических компоненты (Controller1 и Controller2). Здесь, Controller1 имеет простой пропорциональный алгоритм управления формы, однако он реализован в как три функции: t1_read, t1_run, и t1_write. Это разложение помогает моделировать неявное чтение датчика и запись привода, которая присутствовала бы в фактическом контроллере реализации.

Сам планировщик моделирует гомогенную многоядерную систему и определена следующими свойствами, которые являются представленный как простые параметры блока как показано на рисунке 1б, такие как количество ядер, планирование политики, взаимоисключающие ресурсы. Блок планировщика SimEvents позволяет пользователям определять задачи как блочные параметры также как показано в рисунке 1б.

Данные параметры определяют гомогенную многоядерную систему с двумя ядрами. Планировщик настроен на выполнение политики на основе приоритетов. У системы есть две задачи управления:

Задача 1 для блока Controller1 сконфигурирована с периодом 0,5 секунд, приоритет 200 и 3 сегментов, чьи функции и продолжительности выполнения (t1_read, t1_run, t1_write) и (0.1, 0.2, 0.1) второй соответственно.

Задача 2 для блока Controller2 сконфигурирована с периодом 0,5 секунды, приоритет 50, 2 сегмента, чьи функции и продолжительности выполнения (t2_run, t2_write) и (0.25, 0.1) второй соответственно.

2 Результаты

На рис. 1 представлены результаты моделирования.

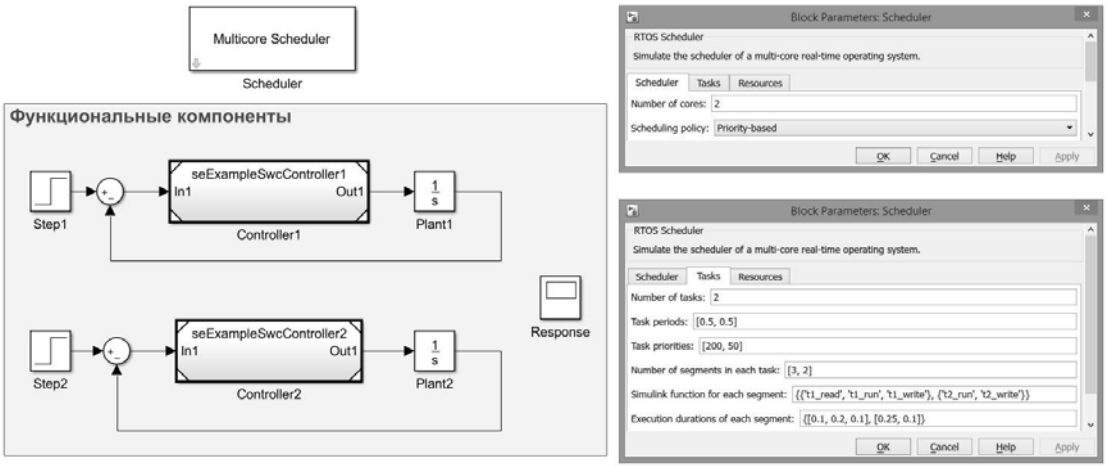

Рис. 1: Simulink модель многоядерной системы: (а) блок планировщика (б) конфигурация параметров планировщика

Реализация блока планировщика основывается на высокоуровневом дискретном событийно-ориентированном языке, который поддерживается Simulink и включает реализацию дополнительного планирования функци-

ональности (например, различные политики). Платформа моделирования более универсальна и не имеет строгой привязки к использованию этого блока планировщика.

Блок планировщика позволяет присваивать произвольный номер из ядер и показывает их влияние на системную производительность. В первом сценарии двум ядра были назначены две задачи управления. Рисунок 1б иллюстрирует параметры блока планировщика для этой конфигурации.

Обе системы управления с обратной связью работают достаточно эффективно, когда точка установления (setpoint) изменяется в диапазоне [0;1) (см. рисунок 2). Каждый из рисунков 2–6 включает диаграмму ответа обработчика (на левой стороне) и временную диаграмму (на правой стороне). Диаграмма ответа обработчика показывает ответ от двух блоков, Controller1 в главном графике, и блока Controller2 в нижнем графике. Временная диаграмма показывает использование ядер и ресурсов. Черно-белый столбик показывает выполнение сегмента задачи. Позиция панели на горизонтальной оси указывает время запуска и завершения сегмента. Позиция панели на вертикальной оси указывает нормализованное завершение задачи перед и после выполнения сегмента.

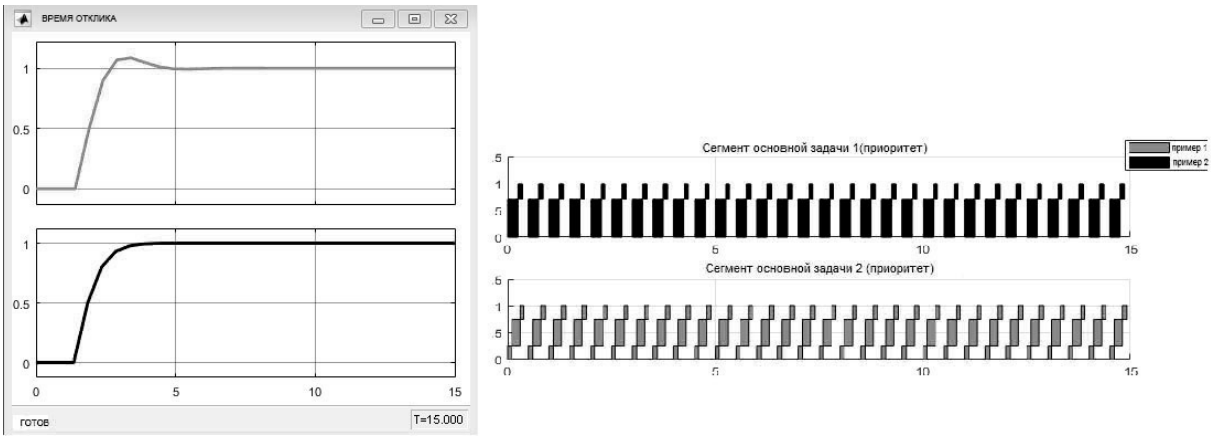

Рис. 2: Последовательность выполнения задач и производительность контроллера (два ядра, основанное на приоритетном планировании)

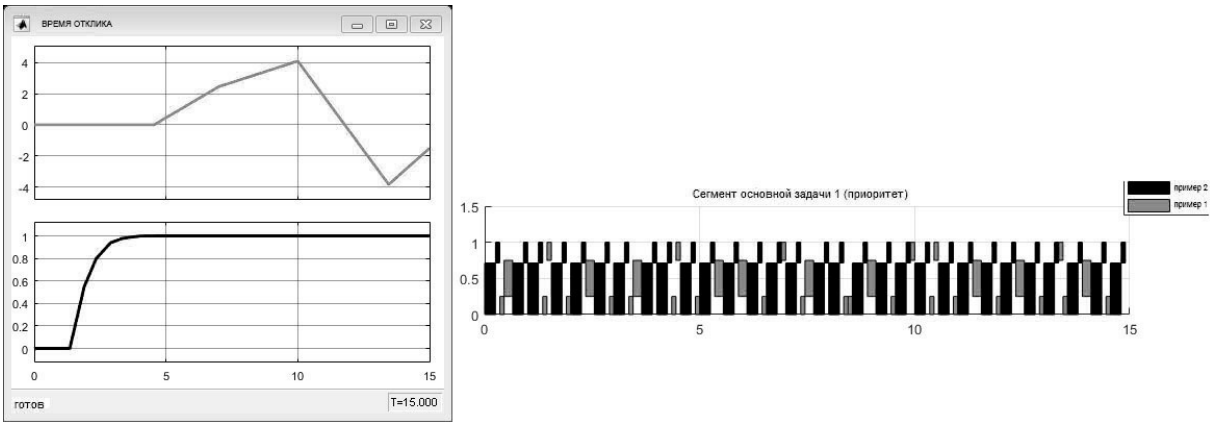

Рис. 3: Последовательность выполнения задач и производительность контроллера (одно базовое, приоритетное планирование)

Временная диаграмма рисунка 6 указывает, что задачи, обработанные одновременно двумя ядрами и имели среднюю и сбалансированную нагрузку использования. Здесь Task2 присвоен Core1 потому что имеет более высокий приоритет и, поэтому, запущен перед Task1.

В рисунке 3, когда только одно ядро доступно, производительность Controller1 (отображенный на низкоприоритетной задаче) ухудшается из-за переполнений задачи. Для примера, были запланированы 10 экземпляров. Первый экземпляр Task1 не завершается через 1,45 сек, потому что его выполнение было вытеснено в

двумя экземплярами первоочередной задачи (Task2). Причиной переполнения является 3 экземпляра Task1, чтобы завершить выполнение между временем 0 и 5. При этом, производительность задачи 2 управления остается неизменной. Это вызвано тем, что планировщик применяет политику на приоритетах, где все мощности отдаются высокоприоритетным задачам.

Если планировщик переключается, чтобы использовать алгоритм Round Robin, то система управления выполняется по-другому (см. рисунок 4). По сравнению с предыдущим случаем, где скорость обработки остается одинаковой и управление Plant1 поддерживается стабильным, за счет ухудшения производительности управления Plant2. Это происходит, потому что алгоритм Round Robin политики равномерно распределяет ресурсы среди всех задачи.

Если добавить ресурс, который должен быть совместно использован задачами взаимоисключающим способом, такими как дескриптор файлов (см. рисунок 5), то, как обозначено временной диаграммой, несмотря на то, что параллельное выполнение можно выполнить двумя ядрами, задачи были обработаны последовательным способом. Только одно ядро было задействовано. Это вызвано тем, что задача должна ожидать требуемый ресурс, прежде чем он сможет начать обработку.

Такая ситуация может быть устранена, добавлением большего количества ресурсов. Таким образом планировщик сконфигурирован, так, чтобы использовать 2 ресурса и каждой задаче позволяется иметь свой ресурс (см. рисунок 6). С каждой задачей, имеющей его собственный ресурс, задачи теперь могут быть обработаны одновременно.

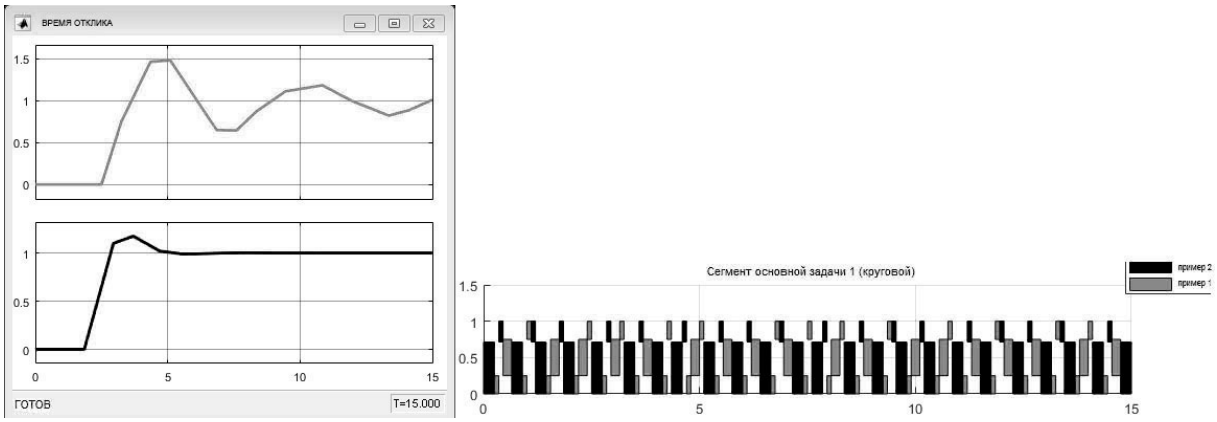

Рис. 4: Последовательность выполнения задач и производительность контроллера (одно ядро, циклическое планирование)

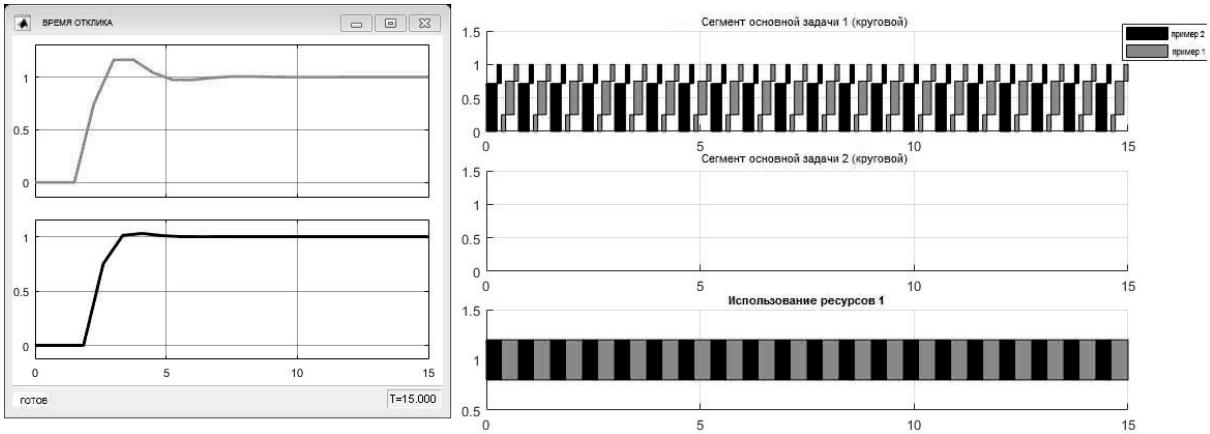

Рис. 5: Последовательность выполнения задач и производительность контроллера (два ядра, задачи совместно используют ресурс, приоритетное планирование)

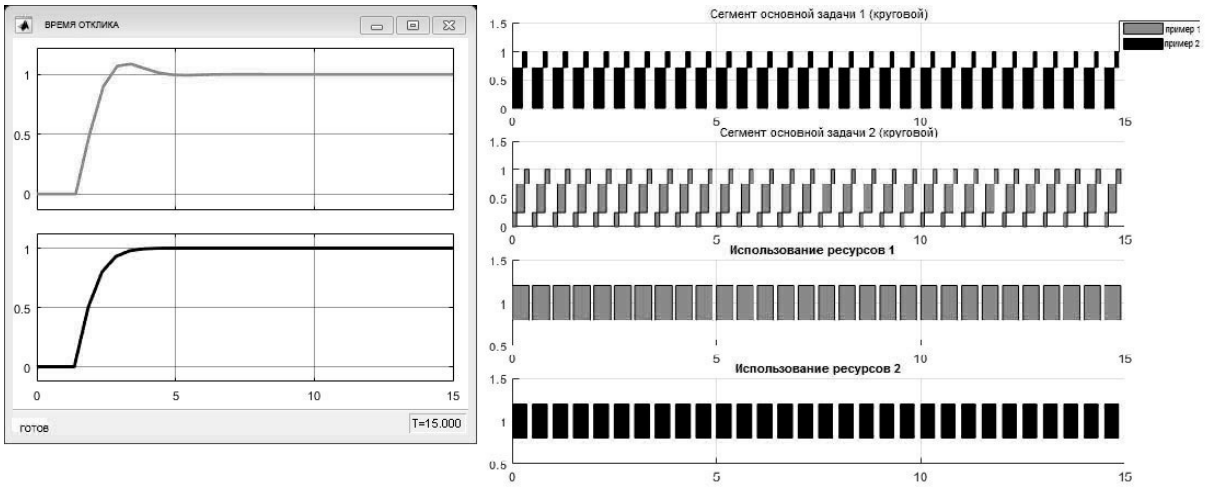

Рис. 6: Последовательность выполнения задач и производительность контроллера (два ядра, без разделения ресурсов, приоритетное планирование)

Заключение

Эта работа описывает использование методологии модельно-ориентированного проектирования для моделирования архитектурных компонентов многоядерных систем управления в реальном времени на основе среды моделирования Mathworks Simulink, включающие в себя планировщик ресурсов для моделирования задач планирования (например, задержка, переполнение ресурсов и другие непредвиденные обстоятельства) для компонентов многоядерных систем управления. Был применен подход к задаче распределения программных задач на ядра многоядерной системы на основе итеративного модельно-ориентированного подхода. Данный подход хорошо подходит для программных встроенных систем реального времени с множеством обрабатывающих узлов и позволяет моделировать с применением собственных реализации известных алгоритмов для распределения ресурсов, при этом показывая приближенную к реальности производительность данных алгоритмов. На нескольких конфигурациях проиллюстрированы подходы к распределению ресурсов с акцентом на два параметра производительности таких систем, а именно, временные показатели и эффективность распределения вычислительной нагрузки по доступным ядрам многоядерной системы.

Список литературы

[1] Simulink MathWorks, Simulink User's Guide. MathWorks, Natick, MA. March, 2016.

[2] Aarenstrup, R. Managing Model-Based Design. 1st edition. CreateSpace Independent Publishing Platform. 2015. — 100 p.

[3] Cremona, F., Morelli, M., and Natale, M.D. TRES: A Modular Representation of Schedulers, Tasks, and Messages to Control Simulations in Simulink // In Proceedings of the 30th Annual ACM Symposium on Applied Computing, ACM Press. 2015. — P. 1940–1947.

[4] Cervin, A., Arzen, K.E. TrueTime // Model-Based Design for Embedded Systems Computational Analysis, Synthesis, & Design Dynamic Systems. 2009. — P. 145–76.

[5] MathWorks, SimEvents User's Guide, MathWorks, Natick, MA, March, 2016.

Айдаров Канат Алхожаевич — ст. преподаватель кафедры информатики механико-математического факультета Казахского Национального университета им. аль-Фараби;
e-mail: kanat.aydarov@kaznu.kz;
Балакаева Гульнар Тултаевна — д.ф.-м.н., профессор кафедры информатики механико-математического факультета Казахского Национального университета им. аль-Фараби;
e-mail: gulnardtsa@gmail.com.
Дата поступления — 31 мая 2017 г.