ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ И ИХ ЭЛЕМЕНТЫ / COMPUTING SYSTEMS AND THEIR ELEMENTS

DOI: https://doi.org/10.23670/IRJ.2023.131.18

ОПЕРАЦИОННЫЕ СИСТЕМЫ ИНТЕРНЕТА ВЕЩЕЙ: ВОЗМОЖНОСТИ, ПРОБЛЕМЫ И РЕШЕНИЯ

Научная статья

Калхиташвили Д.Ш.[1, *]

[1] Российская академия народного хозяйства и государственной службы при Президенте РФ, Москва, Российская Федерация

* Корреспондирующий автор (fullmetaltraumer[at]gmail.com)

Аннотация

В статье рассматриваются операционные системы Интернета вещей (IoT). Обоснована актуальность разработки IoT. Отмечено, что необходимость в разработке операционной системы для IoT обусловлена потребностью в улучшении эксплуатации оборудования IoT и поддержке стандартов и методов для всех уровней связи. Представлена сравнительная характеристика ОС IoT. Выявлено, что Интернет вещей сложен в плане разработки подходящей ОС, которая должна обладать способностью удовлетворять потребности разнородных аппаратных платформ и сценариев приложений, предоставлять адаптивный сетевой стек IP и оптимальный API, на базе которого возможны дальнейшие разработки. Сделан вывод о том, что из проанализированных ОС только RIOT отвечает данным требованиям, позволяя создать полноценный программный продукт и легко использовать имеющиеся библиотеки для задач, связанных с разнородным оборудованием IoT.

Ключевые слова: Интернет вещей, операционная система IoT, совместимость, энергоэффективность, TinyOS, 6LoWPAN.

IOT OPERATING SYSTEMS: OPPORTUNITIES, PROBLEMS AND SOLUTIONS

Research article

Kalkhitashvili D.S.[1, *]

[1] Russian Academy of National Economy and Public Administration under the President of the Russian Federation, Moscow, Russian Federation

* Corresponding author (fullmetaltraumer[at]gmail.com)

Abstract

The article examines Internet of Things (IoT) operating systems. The relevance of IoT development is substantiated. It is noted that the necessity to develop an operating system for IoT is due to the need to improve the operation of IoT equipment and to support standards and methods for all levels of communication. A comparative characterization of IoT operating systems is presented. It is shown that the Internet of Things is challenging in terms of developing a suitable OS, which must have the ability to meet the needs of heterogeneous hardware platforms and application scenarios, provide an adaptive IP network stack and an optimal API on which further development is possible. It is concluded that among the OSs analysed, only RIOT meets these requirements, allowing the creation of a complete software product and easy use of existing libraries for tasks related to heterogeneous IoT hardware.

Keywords: Internet of things, IoT operating system, compatibility, energy efficiency, TinyOS, 6LoWPAN.

Введение

Устройства Интернета вещей (Internet of Things, IoT) имеют ограниченные функциональные возможности и минимальную занимаемую площадь, которые потребляют меньше памяти и вычислительной мощности, чем обычные устройства. Благодаря революции IoT, функциональные возможности этих устройств расширяются. Следовательно, устройство или контроллер должны быть более интеллектуальными для мониторинга нескольких входов, обновления событий для шлюзов или устройств, а также для получения команд от шлюзов или других устройств.

Устройства IoT считаются разнородными и используются в разных приложениях. Различные типы приложений требуют различной архитектуры и аппаратной поддержки, которые могут работать от маломощных источников. Следовательно, иметь операционную систему (ОС), удовлетворяющую всем требованиям к устройствам IoT в различных областях, практически невозможно. Интернет вещей реализуется как высокопроизводительными, так и недорогими устройствами, для которых требуются соответствующие ОС. Устройства высокого класса могут работать с использованием традиционных ОС, тогда как устройства низкого уровня работают с ОС с ограниченными возможностями, которые не могут выполнять те же задачи, что и традиционные ОС. Кроме того, поддержка ОС для IoT играет ключевую роль в развертывании надежных и масштабируемых крупномасштабных развертываний IoT.

Устройствам IoT требуются вычислительные возможности для их операций обнаружения. Эти ограниченные датчики не предлагают большой объем памяти и возможности обработки. Из-за этого ограничения устройства IoT должны эффективно управлять своими ресурсами. Кроме того, плотность, случайность и неопределенность делают управление ресурсами устройств IoT сложной задачей. ОС действует как менеджер ресурсов для этой сложной системы IoT. Чтобы справиться с ограниченной вычислительной мощностью и памятью, ОС требуется эффективный механизм управления процессами и памятью.

Таким образом, появление Интернета вещей позволяет крошечным устройствам с сенсорными и коммуникационными возможностями быть взаимосвязанными и взаимодействовать с киберфизическим миром.

Однако эти крошечные устройства обычно питаются от батарей и имеют ограниченный объем памяти, поэтому они не могут работать с обычными операционными системами, разработанными для компьютеров общего назначения, таких как Windows и Linux. Встроенные операционные системы решили эту проблему и создали для разработчиков основу для написания приложений на этих крошечных устройствах.

Вместе с тем проведенный контент-анализ публикаций отечественных авторов, выявил недостаточное внимание к таким важным вопросам как исследование и разработка операционных систем платформы Интернета вещей, обеспечивающих повышение эффективности системы IoT.

Актуальность проблемы, её недостаточная научная разработка определили важность данного исследования.

Интернет вещей (Internet of Things, IoT) или IoT представляет собой платформу, на которой данные собираются, анализируются и передаются пользователю в удобной для него форме. Платформы могут быть установлены локально или в облаке [4]. IoT позволяет оцифровывать физические объекты и бизнес-операции посредством преобразования аналоговой информации в цифровые данные и добавления возможности подключения программных приложений (выполняемого кода) за счет встраивания датчиков, микроконтроллеров, исполнительных механизмов и других компонентов в объекты (вещи). Выбор платформы зависит от требований конкретного проекта IoT и таких факторов, как архитектура и стек технологии, надежность, параметры настройки, используемые протоколы, аппаратная независимость, безопасность, эффективность, наличие ресурсов [10]. В этой связи важным является рассмотреть возможности операционных систем IoT, задачи и пути их решения.

Методами исследования послужили анализ, синтез, обобщение и систематизация информации по проблеме исследования. Дана характеристика и оценка операционных систем IoT.

Основные результаты

Операционные системы Интернета вещей встраиваются в устройства IoT, объединяя программные компоненты, которые позволяют программировать функции, а также возможности обработки**,** необходимые для обеспечения высокой производительности. Операционные системы предназначены для работы в условиях строгих ограничений устройств IoT с ограниченной памятью и вычислительной мощностью [3].

Операционные системы реального времени IoT (Real-Time Operating Systems, RTOS) запускаются и выполняют функции в устройствах IoT, которые являются ключевыми для сбора и обработки данных при принятии бизнес-решений. Они управляют аппаратным и программным обеспечением устройства, обновлениями драйверов, реконфигурацией и другими ресурсами (обработкой, памятью и хранилищем) [2]. Кроме того, данные системы позволяют подключать устройства к облачным сервисам, приложениям машинного обучения и другим устройствам. Встроенные приложения имеют аппаратное и программное обеспечение специального назначения, котороее взаимодействует с внешними событиями и требует ответов в режиме реального времени (рисунок) [1], [2], [3], [4].

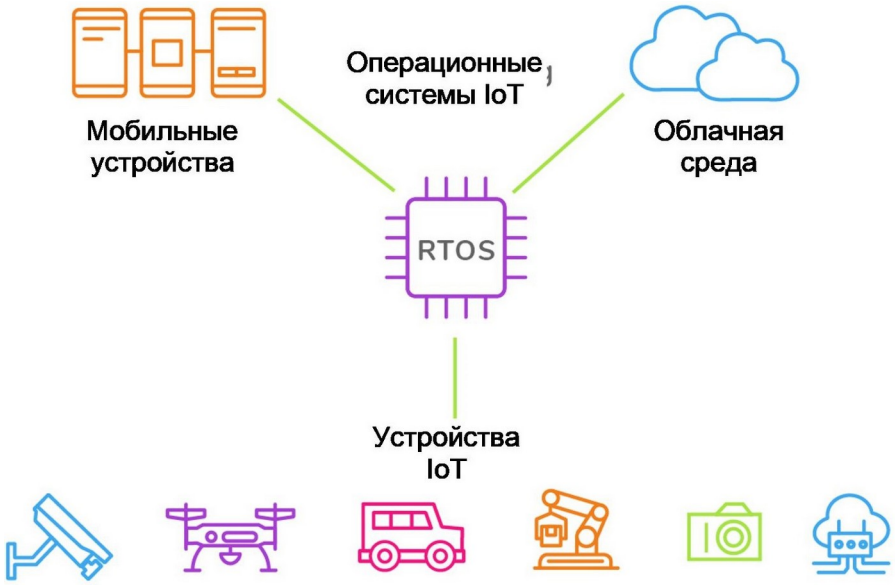

Рисунок 1 - Архитектура операционной системы Интернета вещей
DOI: https://doi.org/10.23670/IRJ.2023.131.18.1

Датчики, радиоидентификаторы (Radio-Frequency Identification, RFID), приводы в сочетании с сенсорными сетями позволяют создавать интеллектуальные объекты при взаимодействии различных устройств (интеллектуальных объектов), которые в последующем будут сформированы в IoT. Интернет вещей – это взаимосвязанные интеллектуальные устройства, формирующие инфраструктуру связи и позволяющие использовать услуги с добавленной стоимостью. Архитектура IoT включает в себя миллиарды устройств, таких как RFID, приводы, компьютеры, мобильные устройства, умные часы, гаджеты и т. д. [7]. Данные устройства генерируют достаточно объемные данные как в целом, так и с целью связи с другими устройствами в инфраструктуре и в разных форматах, таких как аудио, видео, текст, изображения и т. д. Достижения в области прикладных наук IoT способствуют подключению гаджетов удаленно. Следовательно, адаптация требований имеет решающее значение для обеспечения

взаимодействия между разнородными сетями в IoT. Связность отраслевых факторов и использование централизованных или выделенных способов для повышения производительности и эффективности необходимо для промышленного IoT (IIoT).

Непрерывная работа разработчиков ОС IoT имеет основополагающее значение для создания интеллектуальной платформы IoT, поддерживающей современные протоколы. Синхронизация неоднородных устройств IoT, а также управление архитеркурой являются довольно сложной задачей [4].

Существует достаточно большое количество ОС IoT, в частности, ContikiOS, RIOT и Zephyr, использование которых способствует повышению качества программного продукта**.** Блоки IoT отличаются ограниченностью аппаратных ресурсов и емкости батареи. Следовательно, адекватно считающаяся зрелой ОС не может быть запущена на данных устройствах. Код ОС IoT должен быть оптимизирован с использованием типичных возможностей протокола управления передачей/протокола Интернета (TCP/IP) для интеграции с глобальным Интернетом. В связи с этим используемая ОС IoT должна обладать высокой эффективностью для управления всеми компонентами на разных уровнях. Большинство ОС IoT предоставляют полный стек IP-сетей с широко распространенными протоколами пользовательских дейтаграмм (UDP), TCP и протоколом передачи гипертекста (HTTP). Кроме того, ОС дополнительно поддерживает актуальные стандарты, такие как модель Интернет-протокола 6 (IPv6) в маломощных беспроводных персональных сетях (6LoWPAN), маршрутизация в сетях с низким энергопотреблением и потерями (ROLL), а также протокол ограниченных приложений (CoAP).

Рассмотрим основные задачи Интернета вещей и возможные пути их решения.

1. Возможности в реальном времени

Одной из основных задач в ОС IoT является обеспечение реального времени, необходимое для выполнения критических задач, что требует жесткого соблюдения времени выполнения той или иной процецуры [6]. Примером может служить операционная система реального времени (RTOS), которая специально разработана с гарантией выполнения задачи в определенные сроки, что для ОС IoT является достаточно важной характеристикой.

2. Энергоэффективность

Энергоэффективность является фундаментальной задачей Интернета вещей. Большинство устройств IoT по своей природе ограничены в ресурсах [8]. В связи с этим для его работы используются батареи или другие источники энергии. Возможности развертывания IoT разнообразны, сложны и зачастую рассчитаны на автономную работу без подключения к источнику питания. В связи с этим IoT использует RDC для достижения эффективности, надежности работы и точной синхронизации узлов.

3. Сетевое подключение

Для загрузки и выгрузки данных трафика важную роль играет сетевое подключение. Наличие мультиинтерфейса позволяет обеспечить множественную адресацию. Непрерывное развитие и доступность разнородных общих отраслевых протоколов на разных уровнях приводит к плавной интеграции и подключению узлов для формирования сетей.

4. Надежность и безопасность

Надежность и безопасность имеют первостепенное значение для создания единой автоматизированной экосистемы управления всеми устройствами, например, умный дом, умный город и т. д. В целом ОС IoT должна обеспечивать безопасность и конфиденциальность стандартной сети IoT. Открытые вызовы охватывают целостность фактов, аутентификацию и доступ к механизмам. Оптимизированное решение, в первую очередь, основанное на блокчейне, является одним из многообещающих подходов к обеспечению конфиденциальности и безопасности IoT. Кроме того, развернутые сетевые решения должны постоянно пересматриваться для устранения ошибок. Быстрая разработка, развертывание, испытание и адаптация к новейшим предлагаемым стандартам безопасности необходимы для обеспечения максимальной безопасности функционирования сети IoT.

5. Небольшой объем памяти

Небольшой объем памяти ОС IoT с доступностью всего стека TCP/IP для работы на гаджетах с жесткими ограничениями является неотъемлемой частью бесшовной интеграции с международным Интернетом. Оптимизация модулей в памяти эффективным способом, исключающим потерю производительности, является тривиальной задачей [8]. Для ее достижения необходимо соблюдать принципы кодирования, простоту, конфигурирования и модульность.

6. Поддержка гетерогенных устройств

В ОС IoT необходимой характеристикой является поддержка гетерогенных устройств [2]. Быстрое развитие Интернета вещей с разнообразными вариантами использования приводит к увеличению количества больших разнородных устройств. Следовательно, ОС IoT требуют постоянного внедрения новых платформ, обладающих функциональной совместимостью с разнородными устройствами, что позволяет решить вопрос интеграции сценариев развертывания с несколькими экземплярами использования, которые несут разнородные устройства, запускающие разные ОС IoT.

7. Интеллектуальный Интернет вещей

Интеллектуальный Интернет вещей (I-IoT) постепенно становится ключевым фактором масштабной адаптации Интернета вещей к повседневной жизни. В последние годы исследователи начинают изучать и применять искусственный интеллект в сценариях использования IoT. Машинное обучение должным образом изучается и исследуется совместно с нейронными сетями и обработкой изображений. Тем не менее его интеллектуальное и программное обеспечение еще не полностью адаптировано для IoT. Методы машинного обучения необходимо оптимизировать для работы на устройствах с ограниченным доступом к Интернету вещей.

8. Интернет вещей и большие данные

Интернет вещей и большие данные тесно связаны друг с другом, поскольку гаджеты генерируют огромное количество неструктурированных данных. В связи с этим хранение, обработка и проверка статистических данных

являются неотъемлемой частью создания важных отчетов при принятии решений. Новые эффективные и точные методы анализа информации необходимы для улучшения будущих инновационных решений [9].

Проведем сравнительный анализ операционных систем для IoT.

Существует множество операционных систем для IoT. Однако стоит рассмотреть только наиболее используемые разработчиками.

TinyOS предварительно разработана для беспроводных сенсорных сетей (Wireless Sensor Networks, WSN). Однако в настоящее время она мало используется из-за отсутствия активных разработок. TinyOS использует сложный язык программирования C, называемый nesC [5]. Он следует монолитной архитектуре и предоставляет совместный планировщик задач.

Tinythread можно использовать для достижения многопоточности. Он также предоставляет стек IPv6 на основе 6LoWPAN.

TinyOS Low-Power Listening (LPL) реализует функцию Radio Duty Cycling (RDC) для обеспечения энергоэффективности и, следовательно, увеличивает срок службы сети. TinyOS предоставляет симулятор дискретных событий под названием TOSSIM. Следовательно, пользователи могут запускать и отлаживать приложение в системе, а не на mote (беспроводной однокристальный датчик типа «интеллектуальная пыль»).

Таблица 1 - Обзор операционных систем IoT

DOI: https://doi.org/10.23670/IRJ.2023.131.18.2

OS	Min RAM	Min ROM	C Support	C++ Support	Multi Treading	Архитектура	Планировщик
TinyOS	< 1 kB	< 4 kB	χ	χ	~	Монолитная	Совместный
Contiki	< 2 kB	<30 kB	~	χ	~	Монолитная	Совместный, превентивный
RIOT	⟳1,5 kB	⟳5 kB	∨	∨	∨	Микроядро	Tickless, Preempetive, Priority based
Zephyr	⟳2-8 kB	⟳50 kB	∨	∨	∨	Наноядро Микроядро	Preempetive, Priority based
MbedOS	⟳5 kB	⟳15 kB	∨	∨	∨	Монолитная	Preemptive
brillo	⟳32 MB	⟳128Mb	∨	∨	∨	Монолитная	Completely Fair

Примечание: ~ - частичная поддержка; ∨ - поддержка; χ - нет поддержки

В приведенной выше таблице представлены наиболее популярные операционные системы.

Contiki активно используется для устройств IoT. Низкие требования к объему памяти делают Contiki подходящим для устройств с низким уровнем фиксации. Она написана на языке C, обеспечивает многопоточность при использовании протопотока.

Contiki использует совместное или упреждающее планирование процессов. Contiki поддерживает многочисленные богатые сетевые стеки, которые предлагают полный набор точек, таких как IPv6, 6LoWPAN, RPL и CoAP [1]. Кроме того, данная ОС также обеспечивает более одного стандартного управления доступом к среде передачи (Medium Access Control, MAC), такой как множественный доступ с контролем несущей (Carrier Sense Multiple Access, CSMA) и переключение каналов с временными интервалами (Time Slotted Channel Hopping, TSCH). ContikiMAC и Contiki X-MAC RDC используются для повышения энергоэффективности узлов. Симулятор или эмулятор Cooja используется для быстрого написания, просмотра и отладки кода перед его фактическим развертыванием. Он поддерживает несколько устройств IoT, таких как WI smote, Sky и Z1.

ОС Cooja написана на Java и реализована как один поток моделирования. Следовательно, она не может использовать преимущества многоядерных процессоров и занимает много времени для завершения моделирования сценариев. Кроме того, данная ОС требует увеличения скорости новых доступных аппаратных платформ IoT.

RIOT разработана на базе микроядра FireKernel. Основные характеристики включают энергоэффективность, малый объем памяти, модульность и единый доступ к API, который обеспечивает независимую аппаратную абстракцию. RIOT поддерживает языки программирования C и C++. Она также обеспечивает многопоточность с бестактовым, вытесняющим и приоритетным планировщиком. Многопоточность предназначена для уменьшения

внутренних недостатков, таких как затраты на управление потоками, использование стека кода и обмен сообщениями между процессами. Native представляет собой эмулятор или аппаратный виртуализатор, который разрешает пользователю запускать код RIOT в качестве процессов Linux. Следовательно, усовершенствовать программное обеспечение IoT проще, за исключением отсутствия реального оборудования [6].

ОС Zephyr изначально разрабатывалась с помощью Wind River, дочерней компании Intel. Она предоставляет микроядро для менее ограниченных устройств IoT и наноядро для ограниченных устройств, поддерживает многопоточность с совместным, приоритетным, ранним крайним сроком (Earliest Deadline First, EDF), неупреждающим и упреждающим планированием. Программы могут быть написаны на языках программирования C и C++. Zephyr представляет поддержку стека сообщества с несколькими протоколами, поддерживает Bluetooth Low Energy (BLE) 5.0. Приложения можно разрабатывать, создавать и проверять, используя собственный порт POSIX.

MbedOS, операционная система реального времени (RTOS), разработана с помощью Advanced RISC Machine (ARM) для ограниченных устройств IoT. Она специально разработана для 32-битной архитектуры ARM, основана на монолитном ядре и представляет собой вытесняющий планировщик, что помогает развитию C и C+. MbedOS поддерживает многопоточность, 6LoWPAN, BLE, WiFi, subGHz, связь ближнего радиуса действия (NFC), радиочастотную идентификацию (RFID) и глобальную сеть с низким энергопотреблением большой дальности (LoRaLPWAN). Низкие потребности в памяти и ряд аппаратных средств mbedOS делают ее подходящей для применения в сфере Интернета вещей.

Заключение

Таким образом, Интернет вещей сложен в плане разработки подходящей ОС, которая должна обладать способностью удовлетворять потребности разнородных аппаратных платформ и сценариев приложений, предоставлять адаптивный сетевой стек IP и оптимальный API, на базе которого возможны дальнейшие разработки. Из проанализированных ОС RIOT отвечает данным требованиям. Данная ОС предлагает надежную микроядерную архитектуру, в которой для самого ядра требуется всего несколько сотен байт ПЗУ и ОЗУ, включает адаптивный стек сообщества с учетом требований IETF для подключения ограниченных систем к Интернету, таких как 6LoWPAN и RPL. Благодаря удобству API, которое частично совместимо с POSIX, и поддержке различных платформ, RIOT позволяет создать полный программный продукт и легко использовать имеющиеся библиотеки для задач, связанных с разнородным оборудованием IoT.

Конфликт интересов

Не указан.

Рецензия

Все статьи проходят рецензирование. Но рецензент или автор статьи предпочли не публиковать рецензию к этой статье в открытом доступе. Рецензия может быть предоставлена компетентным органам по запросу.

Conflict of Interest

None declared.

Review

All articles are peer-reviewed. But the reviewer or the author of the article chose not to publish a review of this article in the public domain. The review can be provided to the competent authorities upon request.

Список литературы на английском языке / References in English

1. Abid M.A. Evolution towards Smart and Software-Defined Internet of Things / M.A. Abid [et al.] // AI. — 2022. — Vol. 3. — p. 100–123.

2. Baccelli E. OS for the IoT – Goals, Challenges, and Solutions / E. Baccelli [et al.] // Workshop Interdisciplinaire sur la Sécurité Globale (WISG2013). — Troyes, 2013.

3. El-Hasan T.S. Internet of Thing (IoT) Based Remote Labs in Engineering / T.S. El-Hasan // 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT). — Paris, 2019. — p. 976-982.

4. Elsaadany A. Experimental Evaluation of Internet of Things in the Educational Environment / A. Elsaadany, M. Soliman // International Journal of Engineering Pedagogy. — 2017. — Vol. 7. — 3. — p. 50-60.

5. Gomes L. Current Trends in Remote Laboratories / L. Gomes, S. Bogosyan // IEEE Transactions on Industrial Electronics. — 2009. — Vol. 56. — 12. — p. 4744-4756.

6. Gubbi J. Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions / J. Gubbi [et al.] // Future Generation Computer Systems. — 2013. — Vol. 29(7). — p. 1645–1660.

7. Heradio R. Virtual and Remote Labs in Control Education: A survey / R. Heradio, L. de la Torre, S. Dormido // Annual Reviews in Control. — 2016. — Vol. 42. — p. 1-10.

8. Ray P.P. A Survey on Internet of Things Architectures / P.P. Ray // Journal of King Saud University – Computer and Information Sciences. — 2018. — Vol. 30. — 3. — p. 291-319.

9. SerdarAsan S. Engineering Education Trends in the Digital Era / S. SerdarAsan, E. Isikli // IGI Global. — 2020.

10. Toyoda Y. An FPGA-based Remote Laboratory: Implementing Semi-automatic Experiments in the Hybrid Cloud / Y. Toyoda, N. Koike,Y. Li // 2016 13th International Conference on Remote Engineering and Virtual Instrumentation (REV). — Madrid, 2016. — p. 24-29.