Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 23.Б15-мм

Улучшение поддержки операционной системы Embox на отечественном микроконтроллере MIK32 АМУР (K1948BK018)

Наздрюхин Матвей Евгеньевич

Отчёт по учебной практике
в форме «Решение»

Научный руководитель:
доцент кафедры системного программирования Луцив Д. В.

Консультант:
Генеральный директор, ООО «Embox» Бондарев А. В.

Санкт-Петербург
2025

Оглавление

Введение **3**

1. Постановка задачи **5**

2. Обзор **6**

2.1. Обзор существующих решений 6

2.2. Обзор используемых технологий 7

2.3. Выводы 8

3. Описание решения **9**

3.1. Изучение архитектурной и платформенной части операционной системы Embox 9

3.2. Добавить протоколы взаимодействия SPI, I^2C 11

3.3. Перепрошивка микроконтроллера МІК32 АМУР операционной системой Embox для проверки работоспособности реализации 11

Заключение **13**

Список литературы **15**

Введение

Современные информационные технологии развиваются с невероятной скоростью, и одной из тенденций в этой области является разработка и внедрение новых архитектур процессоров, которые обеспечивают высокую производительность, энергоэффективность и гибкость. Одной из таких архитектур является RISC-V, которая набирает популярность благодаря своей открытой и свободной лицензии. В то же время, микроконтроллеры, как важнейшие элементы электронных устройств, также претерпевают значительные изменения, связанные с переходом на новые архитектуры и повышением функциональности. Рассмотрим использование архитектуры RISC-V в микроконтроллерах, а также применение отечественных решений, что является актуальным для развития российской электронной промышленности.

Одной из ключевых задач при разработке встраиваемых систем является выбор операционной системы, которая обеспечивает необходимый баланс между производительностью, функциональностью и минимальными затратами ресурсов. Embox — это легковесная операционная система, разработанная специально для встраиваемых систем и микроконтроллеров. Она предоставляет базовые функции ОС, такие как управление процессами, памятью и устройствами ввода-вывода, при этом занимая минимальный объем памяти. Embox поддерживает различные архитектуры, включая RISC-V, что делает её привлекательным выбором для тестирования и работы на микроконтроллерах.

Архитектура RISC-V представляет собой современную реализацию принципов Reduced Instruction Set Computing (RISC), которая отличается простотой, открытостью и гибкостью. В отличие от закрытых архитектур, таких как ARM, RISC-V не требует лицензионных отчислений, что делает её привлекательной для разработчиков, особенно в условиях глобальных санкций и ограничений. ARM, несмотря на свою популярность, имеет более сложную лицензионную политику и высокие затраты на использование. RISC-V, с другой стороны, предоставляет возможность гибкой настройки и оптимизации под конкретные задачи,

что делает её перспективной для встраиваемых систем и микроконтроллеров. Однако, на данный момент ARM по-прежнему доминирует на рынке благодаря своей зрелости, широкой экосистеме, большому количеству готовых решений и привычке потребителя.

В последние годы в России наблюдается активная разработка отечественных микроконтроллеров, которые базируются на современных архитектурах, включая RISC-V. Эти решения направлены на замещение импортных микроконтроллеров и обеспечение технологической независимости. Например, компания «Микрон» разрабатывает микроконтроллеры на базе RISC-V, которые уже находят применение в различных отраслях. Однако для достижения конкурентоспособности на мировом рынке необходимо не только создание собственных решений, но и развитие экосистемы, включая программное обеспечение, инструменты разработки и поддержку сообщества.

1. Постановка задачи

Цель заключается в улучшении поддержки операционной системы реального времени Embox на микроконтроллере MIK32 АМУР (K1948BK018) Для того чтобы достичь этого нужно выполнить следующие задачи:

1. Изучить ОС Embox, в частности её архитектуру и возможности для встраеваемых систем
2. Рассмотреть специфику микроконтроллера MIK32 АМУР
3. Улучшить поддержку операционной системы Embox на микроконтроллере MIK32 АМУР, в частности реализовать протоколы связи I^2C и SPI
4. Провести апробацию полученного решения

2. Обзор

Микроконтроллер — микросхема для программного управления электронными устройствами. Обычно изготавливается в виде единого кристалла с функциями ядра микропроцессора, шин команд и данных, периферийных устройств, ОЗУ и ПЗУ.

Существует большое количество семейств микроконтроллеров, в качестве примера рассмотрим семейство STM32 [3].

STM32 - Семейство на базе архитектуры ARM Cortex (ARMv7-M, ARMv6-M). Закрытая архитектура с широким спектром уже оптимизированных решений, требующей лицензирования от ARM. У него зрелая экосистема с обширной документацией, библиотеками и инструментами, и, как следствие, широкий выбор готовых модулей и периферии.

Исходя из небольшого обзора, можно сказать, что STM32 является популярным семейством, но в него труднее добавить что-то новое, в отличие от микроконтроллеров на RISC-V [1], для которых не нужно получать лицензию от ARM. Так же для российских производителей будет выгоднее и безопаснее использовать RISC-V, так как не придется платить за лицензирование (которое еще могут не дать), да и не смогут отозвать его

2.1. Обзор существующих решений

Микроконтроллеры на архитектуре RISC-V активно развиваются как в научной, так и в промышленной среде, благодаря своей открытости, которая позволяет адаптировать их под конкретные задачи. На российском рынке уже представлены такие решения, как МІК32 АМУР (К1948ВК018) [2] от компании «Микрон» и К1921ВГ015 от АО «НИИЭТ» [5]. Эти микроконтроллеры сочетают высокую производительность с низким энергопотреблением, что делает их привлекательными для использования в встраиваемых системах реального времени. Однако их интеграция с современными операционными системами, такими как Embox [6] и Zephyr [8], сопряжена с рядом сложностей.

2.2. Обзор используемых технологий

Архитектура RISC-V открывает перед разработчиками широкие возможности благодаря своей гибкости и открытости. Однако её эффективное использование в сочетании с операционными системами требует значительных усилий. В случае с отечественными микроконтроллерами, такими как МІК32 АМУР (К1948ВК018) и К1921ВГ015, ключевой задачей становится разработка специализированных драйверов и адаптация системных компонентов. Это необходимо для обеспечения полноценной поддержки уникальных особенностей этих устройств и их периферийных модулей.

2.2.1. сравнение систем Embox и Zephyr

Embox выделяется своей легковесностью и высокой настраиваемостью. Её архитектура позволяет разработчикам включать только необходимые компоненты, что минимизирует объём занимаемой памяти и повышает производительность в условиях ограниченных ресурсов. Для поддержки платформ требуется разработка драйверов и глубокая оптимизация системных компонентов.

Zephyr предлагает более широкий набор функций «из коробки». Эта операционная система поддерживает модульность, что позволяет создавать решения различной сложности. Zephyr активно развивается при поддержке крупных корпораций и имеет обширную пользовательскую базу, что обеспечивает доступ к подробной документации и множеству примеров.

Для успешного использования отечественных микроконтроллеров на RISC-V в обеих операционных системах необходима дальнейшая разработка и оптимизация, включая создание драйверов и адаптацию системных компонентов. В ходе работы была выбрана операционная система Embox, так как её разработка ведётся в том числе и на МатМехе СПБГУ.

2.3. Выводы

Исходя из проведённого выше анализа, можно сформулировать следующие выводы:

1. Архитектура RISC-V демонстрирует активное развитие, однако поддержка отечественных микроконтроллеров, таких как МІК32 и К1921ВГ015, остаётся на недостаточном уровне.

2. Для успешной интеграции этих микроконтроллеров в современные системы требуется разработка специализированных драйверов и адаптация ключевых системных компонентов.

3. Операционные системы Embox и Zephyr представляют собой перспективные платформы для встраиваемых устройств, однако их поддержка RISC-V нуждается в доработке.

4. Выбор между Embox и Zephyr зависит от задачи. Для поставленных целей подходит больше Embox, так как обычно у встраиваемых систем очень ограниченное количество аппаратных ресурсов.

3. Описание решения

В ходе работы была изучена архитектурная часть реализации операционной системы Embox для микроконтроллера МІК32 АМУР. Этот процесс включал анализ структуры ядра, механизмов управления ресурсами, а также взаимодействия с периферийными устройствами. Также рассмотрена адаптация Embox под специфические особенности архитектуры МІК32, что позволило глубже понять, как операционная система взаимодействует с аппаратной частью.

3.1. Изучение архитектурной и платформенной части операционной системы Embox

В ходе работы выполнено портирование Embox на платформу Syntacore SCR3. Реализована поддержка многозадачности (5 потоков), командной оболочки tish, а также драйверов UART и таймера. Конфигурация системы включает архитектуру RV32IMC, 2 MB ROM и 128 KB RAM. Необходимые данные были взяты из пакета разработчика компании Syntacore [7]. Успешный запуск подтверждён в QEMU.

Так же в рамках изучения Embox выполнена доработка поддержки STM32:

- добавлены конфигурации для чипов stm32F100xb/xe и STM32F10x;
- обновлены конфиги для плат stm32f103_bluepill и stm32f103rc16.

В архитектурной части для платформы МІК32 АМУР было сделано многое, несмотря на то, что для RISC-V уже была реализация, но из-за особенностей микроконтроллера были некоторые нюансы. Например, базовый адрес вектора прерываний (регистр TRAP_VECTOR_REG) его нижний байт зафиксирован в 0xC0. Потому было добавлено это смещение к собственному «вектору прерываний» [4]

Листинг 1: Программная адаптация под эту особенность

```
#if OPTION_GET(BOOLEAN,mik32_fixup)
.section .mik32_fixup, "ax"

.globl _mik32_fixup_interrupt
//.org _start + 0xC0
.org 0x40
_mik32_fixup_interrupt:
.cfi_startproc
        j riscv_trap_handler
.cfi_endproc
#endif
```

В платформенной части уже реализованы UART, GPIO, PIC и так далее, поговорим о них.

UART - (Universal asynchronous receiver/transmitter) преобразует передаваемые данные в последовательный вид таким образом, чтобы было возможно передать их по одной физической цифровой линии другому аналогичному устройству. Он представляет собой логическую схему, с одной стороны подключённую к шине вычислительного устройства, а с другой стороны имеющую два или более выводов для внешнего соединения.

PIC (Programmable Interrupt Controller) – по-русски, контроллер прерываний - микросхема или встроенный блок процессора, отвечающий за возможность последовательной обработки запросов на прерывание от различных устройств.

GPIO (Интерфейс ввода/вывода общего назначения) — это универсальный интерфейс, позволяющий микроконтроллерам взаимодействовать с внешними устройствами. Контакты GPIO могут работать как в режиме входа для считывания сигналов (например, от датчиков или кнопок), так и в режиме выхода для управления устройствами (например, светодиодами или реле). Они часто объединяются в порты для удобства управления и настраиваются программно.

3.2. Добавить протоколы взаимодействия SPI, I^2C

SPI (Serial Peripheral Interface) — это протокол связи, представляющий собой последовательный синхронный стандарт передачи данных. Он используется для обмена информацией между микроконтроллерами и периферийными устройствами, такими как датчики, дисплеи, SD-карты и другие компоненты. SPI был разработан компанией Motorola в 1980-х годах, и благодаря своей простоте и высокой скорости передачи данных получил широкое распространение в электронике. Однако на данный момент его реализация в Embox для платформы MIK32 АМУР отсутствует.

I^2C (Inter-Integrated Circuit) — это интерфейс связи, представляющий собой последовательный синхронный протокол передачи данных, разработанный компанией Philips (ныне NXP Semiconductors) в 1980-х годах. I^2C активно применяется для взаимодействия между микроконтроллерами и периферийными устройствами, такими как датчики, EEPROM, дисплеи и другие компоненты. Основное преимущество I^2C заключается в использовании всего двух линий для передачи данных, что делает его удобным для систем с ограниченным количеством выводов.

На текущем этапе работы реализация обоих протоколов на платформе MIK32 АМУР ещё не завершена и находится в стадии разработки.

3.3. Перепрошивка микроконтроллера MIK32 АМУР операционной системой Embox для проверки работоспособности реализации

На текущем этапе операционная система Embox была успешно собрана и установлена на платформу MIK32 АМУР. Хотя реализация протоколов связи ещё не завершена, данная процедура была выполнена для практического ознакомления с работой системы и проверки её работоспособности. В рамках этого этапа были проведены настройка конфигурации системы, сборка проекта и тестирование на реальном устройстве. В результате удалось подтвердить корректное функциони-

рование операционной системы на данном микроконтроллере.

Заключение

Целью данной работы является улучшение поддержки операционной системы реального времени Embox на микроконтроллере МІК32 АМУР (К1948ВК018). Для достижения этой цели были поставлены и выполнены следующие задачи:

1. проведено изучение операционной системы Embox, включая её архитектуру и возможности для встраиваемых систем. Это позволило понять основные принципы работы ОС и её потенциал для использования на микроконтроллерах;

2. выполнена доработка поддержки платформы STM32 (ARM Cortex-M):

   - добавлены конфигурации для чипов stm32F100xb/xe и stm32F10x;
   - обновлены настройки для плат stm32f103_bluepill и stm32f103rc16;
   - изменения приняты в основную ветку Embox (Pull Request #3544).

3. реализовано портирование Embox на платформу Syntacore SCR3 (RISC-V):

   - поддержка архитектуры RV32IMC с настройкой памяти (2MB ROM, 128KB RAM);
   - разработка драйверов для таймера, UART и контроллера прерываний;
   - успешный запуск в QEMU с поддержкой многозадачности и shell tish;
   - изменения приняты в основную ветку (Pull Request #3535).

4. рассмотрена специфика микроконтроллера МІК32 АМУР, включая его аппаратные особенности и возможности. Это стало важным этапом для адаптации Embox к данной платформе;

5. начата работа по улучшению поддержки Embox на МІК32 АМУР;
6. проведена апробация текущего решения, включая сборку и установку Embox на микроконтроллер. Это позволило убедиться в корректности работы системы на данной платформе.

В результате проделанной работы удалось подтвердить возможность запуска и функционирования Embox на микроконтроллере МІК32 АМУР. Однако для полного достижения цели требуется завершение реализации протоколов I^2C и SPI, что станет следующим этапом работы. Полученные результаты демонстрируют перспективность использования Embox на данной платформе и открывают возможности для дальнейшего развития поддержки встраиваемых систем.

Список литературы

[1] About RISC-V International. — URL: https://riscv.org/about/.

[2] RISC-V микроконтроллер МІК32 АМУР. — URL: https://mikron.ru/products/mikrokontrollery/mk32-amur/#!/tab/672102497-1.

[3] STM32 32-bit Arm Cortex MCUs. — URL: https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html.

[4] Бондарев А. В. Путешествие Embox на AMUR. — URL: https://habr.com/ru/articles/844288/ (дата обращения: 18 сентября 2024 г.).

[5] Микроконтроллер К1921ВГ015 от компании АО НИИЭТ. — URL: https://niiet.ru/product/к1921вг015/.

[6] Официальная википедия Embox. — URL: https://github.com/embox/embox/wiki.

[7] Официальный сайт Syntacore. — URL: https://syntacore.com/tools/development-tools.

[8] Официальный сайт Zephyr. — URL: https://zephyrproject.org/.