Министерство образования и науки Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра дискретной математики и
информационных технологий

РЕАЛИЗАЦИЯ ОТДЕЛЬНЫХ ФУНКЦИЙ ОПЕРАЦИОННОЙ СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ ДЛЯ ПРОЦЕССОРА ARM CORTEX M3 НА БАЗЕ МИКРОКОНТРОЛЛЕРА STM32 DISCOVERY

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студента 4 курса 421 группы
направления (специальности) 09.03.01 — «Информатика и вычислительная техника»
факультета КНиИТ
Кизимова Андрея Дмитриевича

Научный руководитель
к.э.н., доцент _____ Чернышова Г.Ю.
дата, подпись

Заведующий кафедрой
к.ф.-м.н., доцент _____ Тяпаев Л.Б.
дата, подпись

Саратов 2018

Введение. Встраиваемые системы служат для обработки и хранения данных, управления устройствами, контроля и мониторинга. Для таких систем предъявляются повышенные требования к работе: необходимость работы в промышленных условиях, перепадах температур и т.д.

Операционные системы для встраиваемых систем называют операционные системы реального времени (ОСРВ). Потребность в такой операционной системе возникает, когда необходимо максимально использовать все ресурсы процессора, обеспечить многозадачный режим работы устройства, а также способы синхронизации задач.

Разработка программного обеспечения, предоставляющего дополнительные возможности при использовании микроконтроллеров, представляет интерес с практической и методической точки зрения.

Целью выпускной квалификационной работы является реализация отдельных функций операционной системы реального времени для встроенных устройств.

Задачами работы являются:

– сравнение существующих операционных систем реального времени;
– исследование функциональных возможностей ОСРВ;
– исследование архитектуры ARM;
– анализ аппаратной части плат серии STM32F0;
– разработка модулей ОСРВ (диспетчера задач и менеджера глобальных переменных).

Выпускная квалификационная работа состоит из введения, трех разделов, заключения, списка используемых источников и трех приложений. Объем работы составляет 42 страницы, объем библиографии – 25 источников, объем приложений – 10 страниц.

В первом разделе рассматриваются системы реального времени, особенности встраиваемых устройств, их применение, проблемы при проектировании, а также архитектура процессоров ARM Cortex-M3. Во втором разделе представлен обзор операционных систем реального времени

для встраиваемых устройств, их место в иерархии операционных систем, рассматриваются принципы и особенности их работы. В третьем разделе описывается реализация модуля диспетчера задач и менеджера глобальных переменных, возможности использования данных модулей в качестве части системы реального времени, справочные материалы для разработанных модулей.

1 Встраиваемые устройства. Встраиваемая система (embedded system) — вычислительная система, которая является модулем, блоком устройства (частью устройства), и встраивается непосредственно в него или входит в состав устройств, объединенных в единую сеть [1].

Наибольшее применение встраиваемых устройств находит в системах промышленного контроля и автоматизации – 36%. 25% встраиваемых устройств применяются для создания продуктов для массового рынка. Наконец, третье место занимают устройства для Интернета вещей – 24% [2]. Наиболее перспективными направлениями в Интернете вещей являютя система контроля движения на дороге, концепция «умного дома», беспилотные транспортные средства.

Система реального времени — это система, которая должна своевременно выдать ответ на внешнее событие. Обработка входной информации должна производиться за заданный промежуток времени, чтобы обеспечить актуальность информации или корректность дальнейшей работы самой системы [3]. Системы реального времени отличает то, что они призваны работать в условиях жестких временных ограничений. Программно-аппаратный комплекс должен гарантированно реагировать на внешние события с заданным интервалом времени [4].

Системы реального времени можно разделить на два типа: жесткие и мягкие. Если задержка или опоздание реакции системы на внешнее событие может вызвать нарушение корректной работы самой системы, или окружающих ее систем, то, говорят, что это система жесткого реального времени (hard). Если же задержка реакции системы допускается и не приведет к серьезным

последствиям, то такую систему называют мягкой системой реального времени (soft).

Можно выделить несколько особенностей встраиваемых систем: использование специальных процессоров с низким энергопотреблением, сокращенным набором команд; ограничения на размер памяти устройства; необходимость оптимизации программного кода; работа от автономного источника тока.

Часть встроенных систем предназначена для управления относительно дешевыми устройствами. Поэтому успешность использования той или иной встроенной системы зависит от ее стоимости. Стоимость встроенной системы включает в себя стоимость аппаратной части (процессор, периферия, плата), оплату работы проектировщиков и разработчиков, а также стоимость коммерческой операционной системы, если она используется в проекте. Дороговизна коммерческих операционных систем — один из ключевых факторов, почему некоторые разработчики отказываются от их использования в своей системе.

Большинство встраиваемых систем должны обслуживать в режиме реального времени сразу несколько подключенных к ней устройств. В этом случае возникает потребность в использовании операционной системы или написание собственного диспетчера задач.

2 Сравнение операционных систем реального времени. В данной работе рассматриваются основные функциональные возможности трех операционных систем реального времени.

FreeRTOS. Свободно-распространяемая ОСРВ, которая занимает первое место в рейтинге использования операционных систем реального времени. Объем основного ядра системы занимает несколько килобайт, может использоваться в жестких системах реального времени, алгоритм планирования - вытесняющая многозадачность. Из основных достоинств системы можно выделить открытый исходный код и адаптированность системы на многие популярные аппаратные платформ.

QNX. Коммерческая операционная система для мягких и жестких систем реального времени. Система построена на концепции микроядер - каждая важная функция системы оформлена как отдельное ядро, а не монолитное, как, например, у FreeRTOS. Содержит несколько доступных алгоритмов планирования: FIFO, Round-Robin и спорадическое планирования. В бакалаврской работе отдельное внимание уделено работе спорадического планирования в данной операционной системе.

VxWorks. Коммерческая операционная система адаптирована для жестких систем. Система также построена на концепции микроядер, каждый компонент можно подключить в случае необходимости его использования, таким образом, можно сконфигурировать VxWorks под каждую конкретную разработку. Алгоритмы планирования - вытесняющая многозадачность и Round-Robin. Размер ядра - от 100 Кбайт. Портирована на большое кол-во аппаратных архитектур.

Основной недостаток использования коммерческих систем, по исследованию компании AspenCore и специализированного издания EETimes, заключается в их большой стоимости. В большом количестве систем реального времени необходимый функционал реализовывается разработчиками самостоятельно.

В работе рассматривается 32-битный процессор ARM Cortex-M3, который построен на архитектуре ARMv7-M. Процессоры этого семейства распространены на массовом рынке, используются во многих устройствах, в том числе и во встраиваемых системах реального времени. STM32 Discovery представляет собой доступное комплексное решение для оценки возможностей 32-разрядных микроконтроллеров с ARM Cortex.

3 Разработка модулей ОСРВ. Необходимость в разработке отдельных модулей ОСРВ заключается в том, чтобы обеспечить разработчикам минимальный функционал для работы системы в режиме многозадачности и синхронизации задач. Каждый модуль должен быть максимально независим друг от друга и подключаться лишь в случае необходимости.

Модуль диспетчера задач. Многозадачность – одна из ключевых функциональных возможностей операционных систем. В данной работе реализован модуль операционной системы, получивший название диспетчер задач, обеспечивающий работу системы в режиме вытесняющей многозадачности. Суть вытесняющей многозадачности состоит в том, что операционная система разделяет процессорное время между несколькими задачами и принимает решение о переключении по истечении выделенного задачи процессорного времени [5].

В случае использования вытесняющей многозадачности операционные системы разделяют задачи на несколько групп в зависимости от их приоритета. Задача с большим приоритетом получает большее процессорное время.

Задача – это набор операций, предназначенной для выполнения логически законченной функции системы [6]. Свойства (атрибуты) задач: приоритет, контекст и состояние (статус). Каждая задача может иметь приоритет. В разработанном модуле диспетчера задач, приоритет может принимать значения от 0 до 2, где 0 – наименьший приоритет, а 2 – наибольший. Контекст задачи – это набор данных, содержащий всю необходимую информацию для возобновления выполнения задачи с того момента, где она была ранее прервана. Это данные, находящиеся в регистрах процессора, значение программного счетчика в определенный момент времени и указатель стека. Сохранение контекста – запись в память контекста задачи для последующего восстановления через некоторый промежуток времени. Восстановление контекста – присвоение значений регистрам процессора, программному счетчику и указателю стека данных из памяти ранее сохраненного контекста. Переключение контекста – это сохранение контекста текущей выполняемой процессором задачи и восстановление контекста другой задачи.

Задача может находиться в нескольких состояниях:

- активная – задача, которая выполняется в данный момент времени;
- готовая – задача, находящаяся в очереди на выполнение;

– заблокированная – задача, которая находится в ожидании до наступления некоторых событий (например, освобождения необходимого ресурса).

Программная реализация приведена в работе.

Модуль менеджера глобальных переменных. Несмотря на то, что задачи, работающие в режиме многозадачности выполняют разные функции, могут возникнуть ситуации, требующие синхронизации задач. Задачи могут быть связаны, т.е. одна задача подготавливает данные, которые затем будут обработаны другой задачей. Также может возникнуть ситуация, когда одной задаче нужно дождаться выполнения некоторых вычислений в другой задаче [7].

Один из способов синхронизации задач – использование общих ресурсов. В данном случае под общим ресурсом будем понимать область памяти, к которой могут обращаться несколько задач. Использование общих ресурсов в режиме многозадачности может вызвать сложности в реализации. Например, в системе имеется две задачи – одна записывает данные в некоторый общий ресурс, а вторая – считывает и использует для дальнейших вычислений. В режиме многозадачности может возникнуть ситуация, когда первая задача будет производить запись новых значений в ресурс, но в момент записи, будет прервана операционной системой, так как выделенное ей процессорное время истекло. Операционная система передаст управление второй задаче, которая прочитает значение переменной, которое первая задача не успела изменить, и продолжит свои дальнейшие вычисления с этим значением. Это одна из проблем использования общего ресурса, которая имеет название “состояние гонки” – ошибка в проектировании приложений, при котором работа системы зависит от того, в каком порядке выполняются участки кода [8].

Для упрощения работы с задачами, которые должны обмениваться информацией, был разработан модуль ОСРВ – менеджер глобальных переменных. В работе приведена программная реализация менеджера глобальных переменных на языке С.

Задачи не могут напрямую изменять значения переменных, так как они не находятся в их области видимости. Задачи сообщают менеджеру глобальных переменных о необходимости изменить или прочитать значение переменной путем вызова функций getter и setter. Прямой запрет на чтение и модификацию переменной позволит свести до минимума состязательные ситуации и обеспечить упрощение разработки системы, а также позволит отследить изменения и подписаться на них.

Подписка на изменения переменной – это одна из функций, предоставляемой менеджером переменных, которая заключается в следующем. При каждом изменении глобальной переменной, менеджер оповещает систему об этом изменении, путем вызова так называемого наблюдателя - функции, которой в качестве входных аргументов передается новое значение переменной.

Общий ресурс, предлагаемый менеджером переменных, имеет следующие свойства: имя, тип, значение, функция-подписчик. Имя – это название общего ресурса, необходимое для упорядочивания и обращения к нему. Имя может быть произвольным, формата char *. Менеджер памяти поддерживает работу с типами int, float, double. Функция-подписчик – ссылка на функцию, которая будет вызвана после изменения значения ресурса. Предлагается подробное описание разработанных модулей, которые могут применяться в процессе изучения операционных систем реального времени.

Заключение. Основной задачей выпускной квалификационной работы была разработка отдельных модулей ОСРВ с поддержкой архитектуры ARM.

В процессе работы над ВКР, были разработаны модули диспетчера задач и менеджера глобальных переменных. Диспетчер задач предоставляет возможность работы системы в режиме вытесняющей многозадачности. Менеджер глобальных переменных, в свою очередь, предоставляет в область видимости задач функции для взаимодействия с глобальными переменными, что позволяет упростить работу системы в многозадачном режиме.

Разработанные модули ОСРВ могут использоваться как часть системы, где необходимо обеспечить базовый набор для полноценной работы системы в режиме вытесняющей многозадачности. Также модули можно использовать в методических целях для изучения принципов работы операционных систем реального времени.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Программирование микроконтроллеров с использованием OCPB OSA [Электронный ресурс] : (на 1 июня 2018 года) // PIC24 [Электронный ресурс] : [сайт]. URL: http://www.pic24.ru/doku.php/osa/articles/rtos_usage
2. 2017 Embedded Markets Study [Электронный ресурс] : (на 1 июня 2018 года) // Eetimes/embedded.com, 2017 [Электронный ресурс] : [сайт]. URL: \url {https://m.eet.com/media/1246048/2017-embedded-market-study.pdf}
3. Встраиваемые системы [Электронный ресурс] : (на 1 июня 2018 года) // Промышленные компьютеры [Электронный ресурс] : [сайт]. URL: https://ipc2u.ru/catalog/vstraivaemye-sistemy/
4. Федосеев А. Системы реального времени: от Linux к Java / Федосеев А. // Открытые системы. СУБД // 2009. №03.
5. Henderson H. Encyclopedia of Computer Science. 4-е изд. NY: Facts On File. 2009. 580 p.
6. Сорокин С. Системы реального времени / С. Сорокин // Современные технологии автоматизации // 1997. №2. С. 22-29
7. Сулейманова А. Системы реального времени. Учебное пособие. Уфа: Уфимский государственный авиационный технический университет.
8. Реймонд Э. Искусство программирования для Unix М.: Издательский дом «Вильямс», 2005. 544 с.