САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Математико-механический факультет

Кафедра Системного Программирования

Сарманова Снежана Геннадьевна

Проектирование и реализация библиотеки времени исполнения виртуальной Java машины CLDC HI для кибернетического контроллера ТРИК

Бакалаврская работа

Допущена к защите.
Зав. кафедрой:
профессор Терехов А. Н.

Научный руководитель:
ст. преп. Полозов В. С.

Рецензент:
ведущий инженер Oracle Development SPB Трошин С. Л.

Санкт-Петербург
2014

SAINT-PETERSBURG STATE UNIVERSITY
Mathematics & Mechanics Faculty

Chair of Software Engineering

Snezhana Sarmanova

Design and implementation of Java virtual machine CLDC HI runtime library for TRIK cybernetic controller

Graduation Thesis

Admitted for defence.
Head of the chair:
professor Andrey Terehov

Scientific supervisor:
sen.lect. Victor Polozov

Reviewer:
staff engineer at Oracle Development SPB Sergey Troshin

Saint-Petersburg
2014

Оглавление

Введение

Робототехника является одним из важнейших направлений научно-технического прогресса, в котором проблемы механики и новых технологий соприкасаются с проблемами искусственного интеллекта. К этой сфере проявляется большой интерес как со стороны опытных программистов, так и со стороны школьников, поскольку для последних это возможность наглядного изучения математики и информатики.

Информатика является одним из сложных школьных предметов, так как детям приходится учиться создавать абстракции, принципиально незримые и неосязаемые. Для облегчения понимания данного предмета в школах активно внедряются робототехнические конструкторы [7]. Примером такого конструктора является проект ТРИК, созданный командой опытных инженеров Математико-механического факультета СПбГУ, который включает в себя кибернетический контроллер и специальный конструктор, с помощью которых можно создавать и программировать роботов [4, 5].

Контроллер — это основа робота. Он способен одновременно решать множество задач: обработка аудио- и видеоданных, синтез речи, навигация; управление сервоприводами и моторами; сбор показаний с аналоговых и цифровых датчиков; обмен информацией по беспроводной связи.

В IT индустрии существует множество языков программирования, которые в ознакомительных целях изучаются в школах. Поскольку проект ТРИК внедряется в общеобразовательные процесс, становится актуальна проблема управления роботами на разных языках [7]. Одним из вариантов рассматривался объектно-ориентированный язык Java. К преимуществам данного языка относится наличие сборщика мусора, благодаря которому разработчик не тратит время на явное выделение и освобождение памяти, и возможности использования JIT-компилятора, который компилирует байт-код в машинный во время работы программы. Существует предположение, что это позволит в некоторых ситуациях работать программе на контроллере быстрее по сравнению с "нативной" реализацией. По этой причине возникла необходимость внедрения объектно-ориентированного языка Java, одного из популярных и активно развивающихся языков на сегодняшний день.

На момент начала работы над проектом контроллер не поддерживал платформу Java. Одним из вариантов решения этой проблемы является использование виртуальной Java машины Connected Limited Device Configuration HotSpot Implementation (aka CLDC HI) для встраиваемых систем, разработка компании Sun Microsystems.

Использование данной виртуальной машины позволит решить проблему отсутствия платформы Java на контроллере ТРИК. Целью данной работы является разработка удобного API с поддержкой требуемой периферии контроллера ТРИК для решения задач по программированию роботов на языке Java.

Постановка задачи

Виртуальная машина CLDC HI позволяет Java-приложению работать на устройствах, ограниченных в ресурсах. Если использовать CLDC HI на контроллере ТРИК, который не поддерживает Java платформу, то это позволит создавать приложения для управления роботами.

Целью данной работы является разработка удобного API с поддержкой требуемой периферии контроллера ТРИК для программирования роботов.

Для достижения поставленной цели были определены следующие задачи:

- анализ предметной области и существующих решений;
- сборка виртуальной машины Java CLDC HI средствами ТРИК;
- проектирование и реализация библиотеки времени исполнения виртуальной Java машины CLDC HI для контроллера ТРИК;
- апробация разработанной библиотеки на контроллере.

1. Обзор

1.1. Существующие решения

Перед решением поставленной задачи был произведен анализ предметной области и проанализированы существующие решения.

1.1.1. Oracle Java ME Embedded

Oracle Java ME Embedded — это полноценный Java runtime, оптимизированный для устройств с ARM архитектурой и систем с ограниченными аппаратными возможностями. Технология обеспечивает специализированную встраиваемую функциональность и предназначена для маломощных, ограниченных в ресурсах запоминающих устройств, работающих с различными сетевыми сервисами и интерфейсами ввода/вывода [1].

Java ME Embedded является первой реализацией платформы, оптимизированной для Small Embedded.

Oracle Java ME Embedded включает следующие возможности для встраиваемых систем:

- система управления жизненным циклом;
- проводное (Ethernet) или беспроводное TCP/IP соединение;
- прямой доступ к периферии из Java (Device Access API) и пр.

Полностью совместим с CLDC 1.1 (JSR 139) и IMP-NG (JSR-228), а также поддержка нескольких дополнительных JSRs: File I/O API's (JSR-75), Wireless Messaging API's (JSR-120), Web Services (JSR-172), Security and Trust Services subset (JSR-177), Location API's (JSR-179), XML API's (JSR-280).

Текущая версия релиза Oracle Java ME 3.3—3.4.

Платформа портирована на Cortex-M3/RTX (KEIL Evaluation Board), ARM11/Linux (Raspberry Pi Model B), ARM9/BREW MP (Qualcomm IoE development), X86/Windows (эмулятор).

Ниже перечислены недостатки этой технологии.

1. Не open source проект, что не позволяет при необходимости внедрить дополнительный функционал.

2. Java ME Embedded 3.3-3.4 может быть самостоятельно установлена на RPi, Keil borad, etc., т.е. на тех платформах, на которые она была портирована. Однако, для этого могут требоваться лицензии и коммерческие инструменты. Для установки на новую платформу типа ТРИК требуется сначала портировать Java ME Embedded, что невозможно без исходного кода.

3. Отсутствие графического интерфейса в Java Embedded не позволит использовать дисплей контроллера ТРИК.

1.1.2. JSR 256 Sensor

Библиотека Sensor API (JSR 256 для Java Platform, Micro Edition [Java ME]) входит в состав платформы Java ME, предоставляет возможность Java мидлетам управлять физическими и виртуальными датчиками, доступными на устройстве. Это позволяет обеспечить мидлету средства для оценки состояния устройства, а также внешней физической среды, что позволяет создавать функциональные приложения. Конечно, как и все API-интерфейсы, реализация этого API ппозволяет определить фактические датчики, которые доступны на мобильном устройстве.

Эта спецификация определяет общий интерфейс для датчиков при работе J2ME приложения, который поставляется в качестве дополнительного пакета. Данный API предполоагет унифицированный способ управления датчиками, подключенными к мобильным устройствам, а также легкий доступ к данным с датчиков. С помощью данного API возможно управлять датчиками с помощью базовых протоколов подключения, например включения, выключения датчика, его обнаружение, получение данных и т.д.

Ниже перечислены недостатки этой технологии.

1. Данная библиотека позволяет только считывать данные с датчиков, но не записывать.
2. Sensor API ориентирован исключительно на мобильные устройства и, как следствие, отсутствует поддержка необходимой периферии контроллера ТРИК.

1.2. Используемые технологии

При решении поставленной задачи были использованы следующие технологии.

1.2.1. Контроллер ТРИК

ТРИК — это возможность прототипировать робототехнические комплексы от радиоуправляемых моделей до сложных кибернетических систем. Современные возможности электроники позволяют быстро моделировать и недорого реализовать даже группы автономно взаимодействующих роботов. ТРИК — исключительно Российская разработка, создаваемая как унифицированная платформа для сервисной робототехники. Этот проект создан инженерами Математико-механического факультета СПбГУ, который включает в себя все необходимое для сборки робототехнических комплексов: контроллер, набор прототипирования и среду разработки TRIK Studio.

Контроллер — это основа робота. Он совместим с широким спектром периферийных устройств, имеет в своем составе все необходимое оборудование для управления двигателями постоянного тока и сервоприводами, а также для приема и обработки информации от цифровых и аналоговых датчиков, микрофонов, видеомодулей. Контроллер снабжён цветным сенсорным дисплеем, программируемыми кнопками, есть поддержка WiFi, Bluetooth 4.0 (включая LE) и ANT. В контроллере установлены встроенные защиты от перегрузки по току и от глубокой разрядки аккумулятора.

Контроллер работает под управлением операционной системы linux, ядро которой было специально уменьшено и настроено под данный контроллер. ТРИК работает на основе трех процессоров: DSP, ARM, MIPS. Центральным процессором является OMAP-L138 C6-Integra™ DSP+ARM, разработанный компанией Texas Instruments. Доступная оперативная память составляет 256 Mb, тактовая частота — 375 МГц.

Контроллер поддерживает следующие периферийные устройства (рис. 1):

- **I2C** (Inter-Integrated Circuit) — шина для безопасной передачи данных между периферийными устройствами и контроллером, использующая две двунаправленные линии связи (SDA и SCL). По данной шине обмениваются данными силовые моторы, энкодеры, аналоговые сенсоры.
- **Энкодеры**. Контроллер положения вращающегося объекта или по-другому энкодер - это электромеханическое устройство, с помощью которого можно определить положение вращающейся оси (вала). В данном устройстве механическое движение преобразовывается в электрические сигналы, определяющие положение объекта, дают информацию об угле поворота вала, его положении и направлении вращения. С помощью энкодера также можно измерить длину и расстояние или установить перемещение инструмента. Разъемы на схеме: JM1 - JM4
- **Сервопривод** — привод, который в отличие от мотора постоянного тока не просто крутится, пока подаётся напряжение, а стремится повернуться к заданному углу и удержаться в этом положении. Угол устанавливается с помощью ШИМ (PWM)-сигнала. Разъемы на схеме: JE1 - JE4, JC1 - JC3
- **Датчики положения в пространстве**. К таким устройствам относятся гироскоп и акселерометр, которые позволяют с высокой точностью определить положение устройства в пространстве. Результатом работы являются три проекции вектора ускорения на оси x, y и z. Диапазон значений от -32767 до 32767. В спокойном состоянии показывает проекции вектора g.
- **Цифровые и аналоговые сенсоры**. Широкоий спектр устройств, таких как видеокамера, микрофон, работа с аудио и пр. Разъемы на схеме: JA1 - JA6, JD1 - JD2

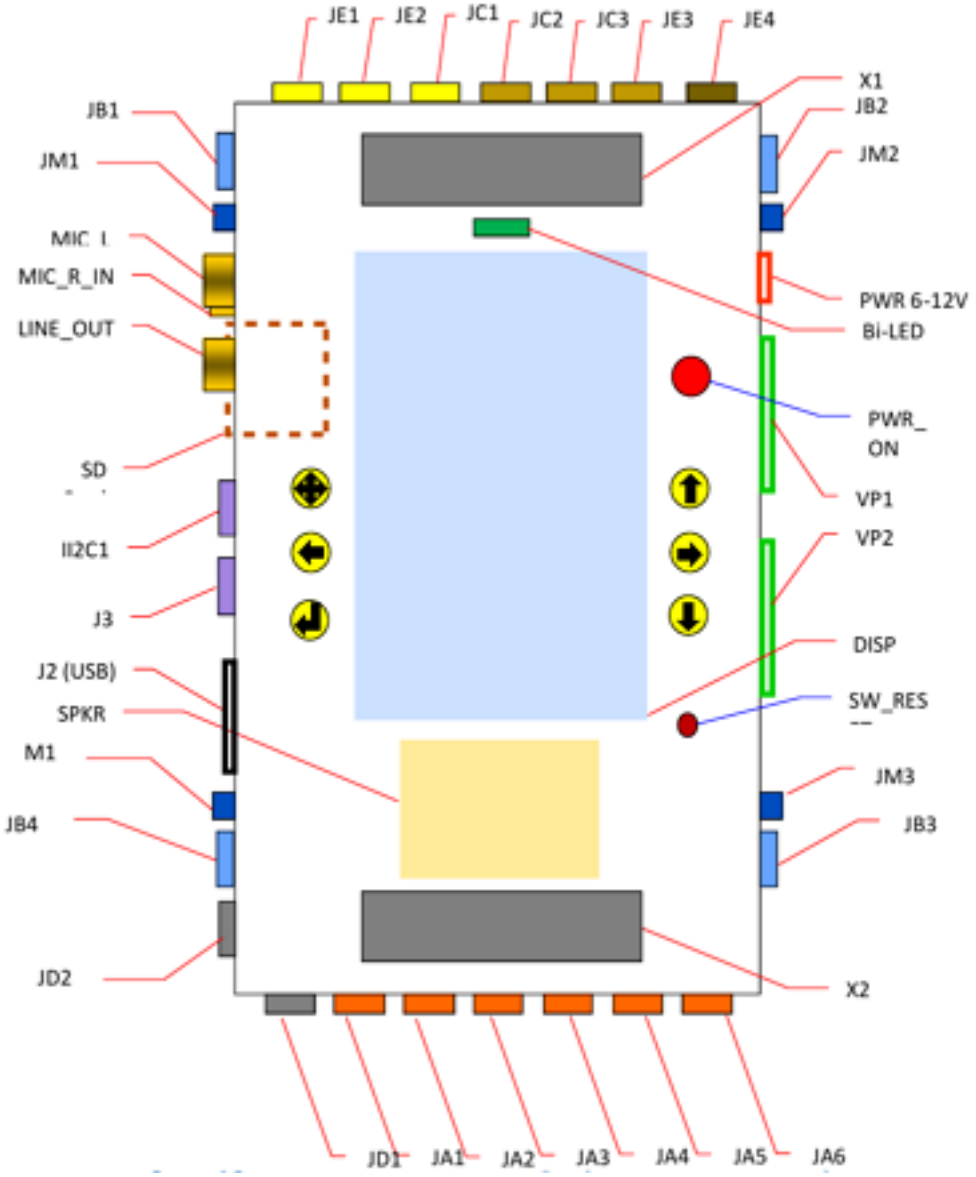

Рис. 1: Расположение элементов управления и разъемов контроллера

- **Геймпад**. Представляет из себя приложение, работающее на Android-устройстве и позволяющее удаленно управлять роботом через tcp-соединение.
- **Прочее:** кнопки, светодиоды, сенсорный дисплей ...

С подробной спецификацией кибернетического контроллера ТРИК можно ознакомиться в документации [5].

Стоит отметить, что обычно большая часть программирования для ARM-систем ведется на рабочих компьютерах с процессорами, отличными от ARM, с помощью инструментов, выполняющих кросс-компиляцию для платформы ARM. Одной из таких сред разработки является пакет инструментов trikSDK, стандартный набор GCC инструментов, адаптированный под проект ТРИК, позволяющий разрабатывать при-

ложения для ARM-систем.

Пакет инструментов trikSDK доступен для бесплатной загрузки и использования на сайте проекта ТРИК [5].

1.2.2. Java 2 Micro Edition

Java 2 Micro Edition (J2ME) – подмножество платформы Java для устройств, ограниченных в ресурсах. J2ME разработана под руководством Sun Microsystems и является заменой похожей технологии — PersonalJava.

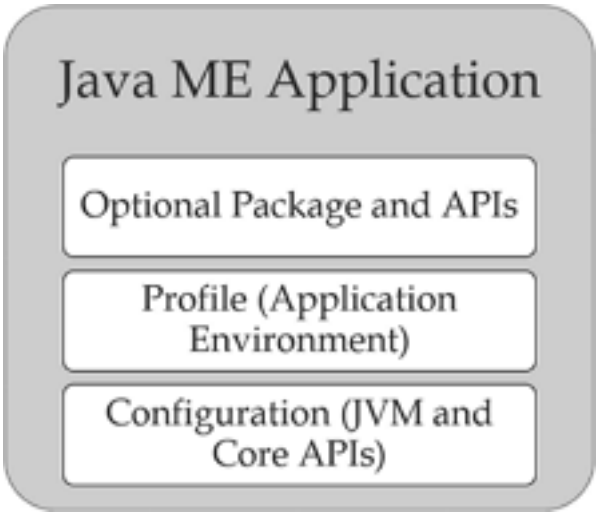

Рис. 2: Архитектура J2ME

Устройства, на которых может работать J2ME-приложение, определяются поддерживаемой конфигурацией и профилем платформы (рис. 2).

Конфигурация описывает только низкоуровневую часть платформы: возможности языка Java, его виртуальной машины, и базовые классы. Конфигурация призвана объединять все устройства со сходными вычислительными возможностями, независимо от их назначения.

В настоящее время существует две конфигурации J2ME — CDC (Connected Device Configuration) и CLDC (Connected Limited Device Configuration), которые различаются вычислительными способностями. Конфигурация CLDC предназначена для устройств с урезанными вычислительными возможностями по сравнению с CDC. Она содержит базовый набор библиотек и функций виртуальной машины, необходимый для реализации J2ME на устройствах с существенно ограниченными возможностями. CLDC ориентировано на устройства с медленными сетевыми соединениями, ограниченным электропитанием, 128 Кб энергонезависимой защищенной от записи памяти и 32 Кб энергозависимой памяти для исполнения приложений. CLDC использует энергонезависимую память для хранения библиотек и виртуальной машины.

Более высокоуровневой частью платформы является профиль. Предполагается,

что профиль будет задаваться для каждого крупного класса устройств (мобильные телефоны, игровые автоматы, бытовые приборы). Т.е. профиль определяет тип устройств, поддерживаемых приложением. Профиль дополняет конфигурацию специфическими классами, определяющими область применения устройств. В J2ME определено три профиля, построенных на основе CLDC: Information Module Profile, Embedded Profile и Mobile Information Device Profile (MIDP). Для CDC доступен шаблонный профиль — Foundation Profile.

MIDP – общеиндустриальный стандартный профиль для мобильных устройств, который не зависит от разработчика и производителя устройства. Этот профиль построен на базе CLDC и обеспечивает стандартное окружение и динамическую передачу приложений на пользовательские устройства. MIDP содержит пакеты для работы с графикой, звуком, взаимодействия с пользователем (клавиатура и экран), базовый набор классов для отображения на стандартных экранах.

Сочетание конфигурации CLDC и профиля MIDP являются распространенным на рынке мобильных систем.

1.2.3. Виртуальная Java машина CLDC HI

Connected Limited Device Configuration HotSpot™ Implementation Virtual Machine (CLDC HI VM) — высокопроизводительная виртуальная Java машина для устройств, ограниченных в ресурсах, разрабатываемая компанией Oracle (бывшей Sun Microsystems). Это одна из виртуальных машин для «малых» устройств, позволяющая запускать J2ME-приложения на устройствах с ограниченным объемом памяти и вычислительной мощностью, например, на мобильных телефонах, КПК, платежных терминалах и т.д.

CLDC HI JVM является оптимизированной виртуальной машиной, которая обеспечивает более быстрое исполнение байт-кода и более эффективное использование ресурсов по сравнению с другими доступными виртуальными машинами, такими как Squawk, KVM, Maxine и пр. Стоит отметить, что данная Java-машина ориентирована главным образом на ARM процессоры.

Как отдельный продукт она не поставляется, но входит в несколько решений для производителей соответствующей аппаратуры. Одна из версий этой виртуальной машины разрабатывалась компанией Sun под лицензией GPLv2, т.е. с открытым исходным кодом (также проект носит имя PhoneME Feature). Однако документации по сборке и запуску этой виртуальной машины в публичном доступе практически нет, и её разработка и поддержка практически были остановлены Oracle после поглощения компании Sun.

CLDC HI построена на конфигурации CLDC 1.1(JSR 118) и профиле MIDP 2.1 (JSR 139).

Приложения J2ME, которые должны работать на CLDC HI, ориентированы на устройства со следующими характеристиками:

- минимальный объём ПЗУ: 128 Кб для CLDC 1.0, 160 Кб для CLDC 1.1;
- минимальный объём ОЗУ — 32 Кб;
- процессор: 16- или 32-битный;
- низкое энергопотребление.

С более подробной спецификацией этой виртуальной машины можно ознакомиться в документации [3].

2. Сборка CLDC HI средствами ТРИК

После поглощения Oracle компании Sun разработка и поддержка CLDC HI, как проекта с открытым исходным кодом, были прекращены, поэтому на данный момент документации по сборке и запуску этой виртуальной машины в публичном доступе практически нет.

В рамках данной работы была создана методика по сборке Java машины CLDC HI [6], включающая:

- минимальный набор продуктов, необходимых для сборки и кросс-компиляции машины в исполняемый модуль;
- настройка среды кросс-компиляции на основе trikSDK (набор инструментального ПО для кросс-компиляции исходного кода с одной платформы на другую);
- локализованные и исправленные ошибки в исходном коде, связанные с неполной обратной совместимостью инструментов сборки;
- ключи компиляции для запуска на платформе с ARM-архитектурой.

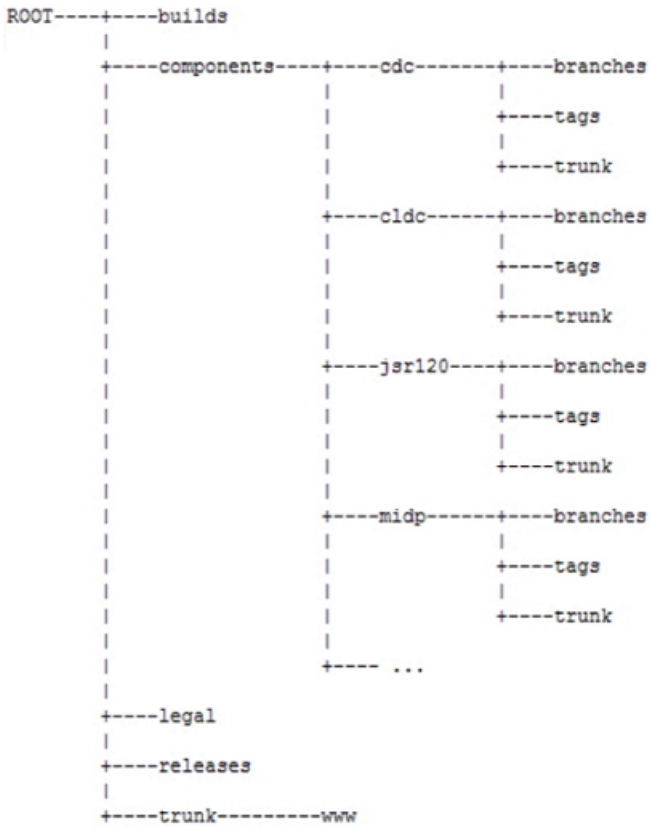

Рис. 3: www.java.net/: Структура репозитория CLDC HI

Для сборки виртуальной машины в репозитории проекта CLDC HI, структура которого отображена на рис. 3, необходимы базовые компоненты:

- cldc – конфигурация платформы Java;
- midp – профиль платформы Java;
- pcsl – набор вспомогательных библиотек;
- tools – вспомогательные инструменты для сборки.

Если нужно расширить функционал работы Java машины, то понадобятся соответствующие JSR (Java Specification Request) пакеты, находящиеся в репозитории проекта. Они должны собираться вместе с midp.

Далее приведены основные этапы построения CLDC HI:

- настройки среды кросс-компиляции (инструментарий, переменные окружения);
- сборка PCSL библиотек;
- сборка конфигурации CLDC;
- сборка профиля MIDP, а также при необходимости сборка дополнительных API (JSRs).

С более детальной инструкцией по сборке виртуальной машины CLDC HI на Linux под целевую платформу ARM можно ознакомиться в статье [6].

3. Архитектура библиотеки

CLDC — это спецификация конфигурации виртуальной машины и базовых классов. CLDC HI VM — это конкретная реализация спецификации CLDC от Sun Microsystems. MIDP — спецификация профайла, состоящего из набора библиотек. Профиль платформы включает набор дополнительных библиотек, таких как работа с сетью, безопасность и др, а также функциональные JSR пакеты, которые дополняют виртуальную машину специфическим функционалом. Сочетание CLDC HI VM с реализацией MIDP дает одну из реализаций платформы Java ME (рис. 4).

Разрабатываемая библиотека com.trikset.control подобна JSR, так как она представляет собой специфичный набор классов по управлению периферийными устройствами контроллера. Соответственно, при кросс-компиляции CLDC HI под целевую платформу com.trikset.control собирается вместе с профилем MIDP.

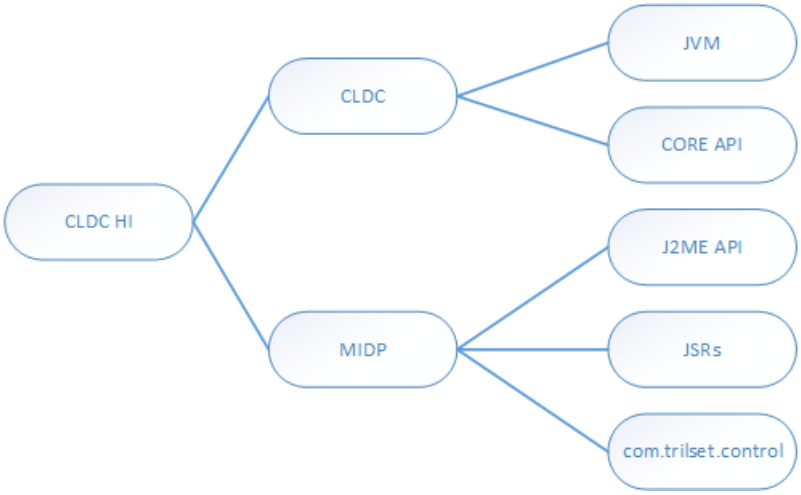

Рис. 4: Архитектура виртуальной машины CLDC HI с библиотекой com.trikset.control

Все периферийные устройства контроллера можно классифицировать 2 способами: те, которые общаются с контроллером по I2C шине, и те, которые записывают и вычитывают данные из файла устройства на плате (рис. 5).

Рассмотрим первый случай на примере силового мотора. На плате имеется микроконтроллер, генерирующий сигнал по принципу широтно-импульсной модуляции(PWM), который приводит в действие периферийное устройство. Для того, чтобы задать скорость вращения силового мотора, необходимо по I2C шине передать данные на микроконтроллер; тот, в свою очередь, сгенерирует сигнал, который позволит изменить значение скорости движения мотора.

Во втором случае, запись/чтение данных на устройство осуществляется путем общения с файлом устройства, который хранится в окружении системы Linux на контроллере.

Далее приведено краткое описание некоторых классов разрабатываемой билиб-

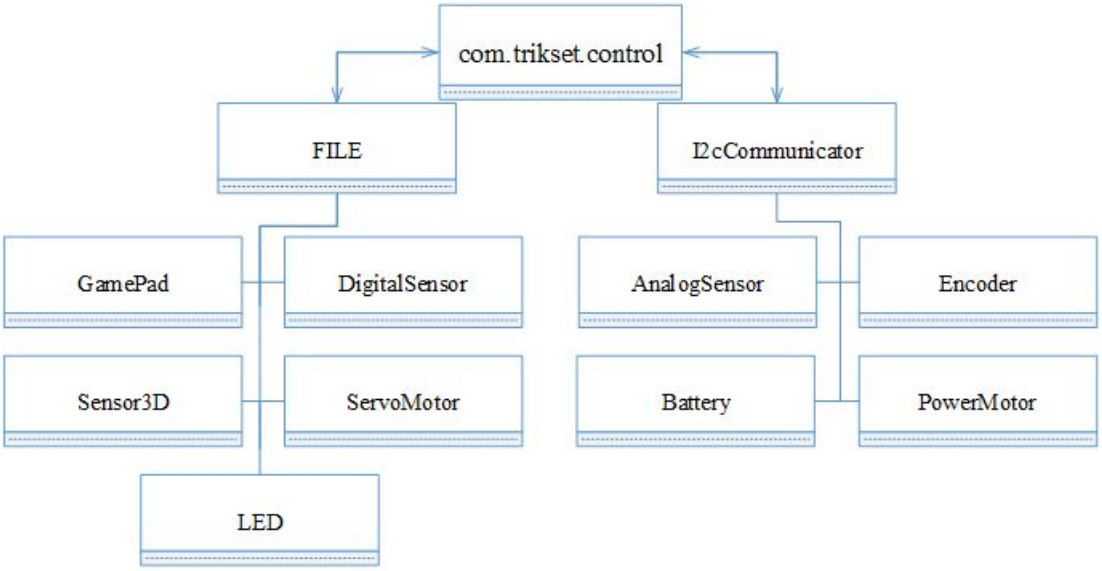

Рис. 5: Архитектура библиотеки com.trikset.control

лиотеки.

- Sensor3D – управление датчиками положения – читает текущие показания с датчика;
- PowerMotor – силовой мотор, позволяет выставить или получить текущую мощность;
- GamePad – обеспечивает управление роботом по tcp-соединению.

С более детальной инструкцией можно ознакомиться на сайте проекта [2].

4. Особенности реализации

При решении поставленной задачи для сборки виртуальной машины необходимы были три модуля: конфигурация cldc, профиль midp и реализованная библиотека com.trikset.control (рис. 6).

Изначально предполагалось, что сборка Java-машины будет происходить только с модулями cldc и com.trikset.control, поскольку они предоставляют базовый набор классов языка Java для управления роботом. Но так как проект ТРИК имеет возможность удаленного управления роботом по tcp-соединению с помощью android-устройства, появляется необходимость использования модуля midp при сборке CLDC HI, поскольку он предоставляет интерфейсы обмена данными через сокеты.

Её использование также обосновывается тем, что контроллер имеет в своем составе все необходимое оборудование для приема и обработки информации с видеомодулей, микрофона, цветного сенсорного дисплея и пр. Модуль midp предоставляет широкий спектр функций для работы с данными, получаемыми с этих устройств.

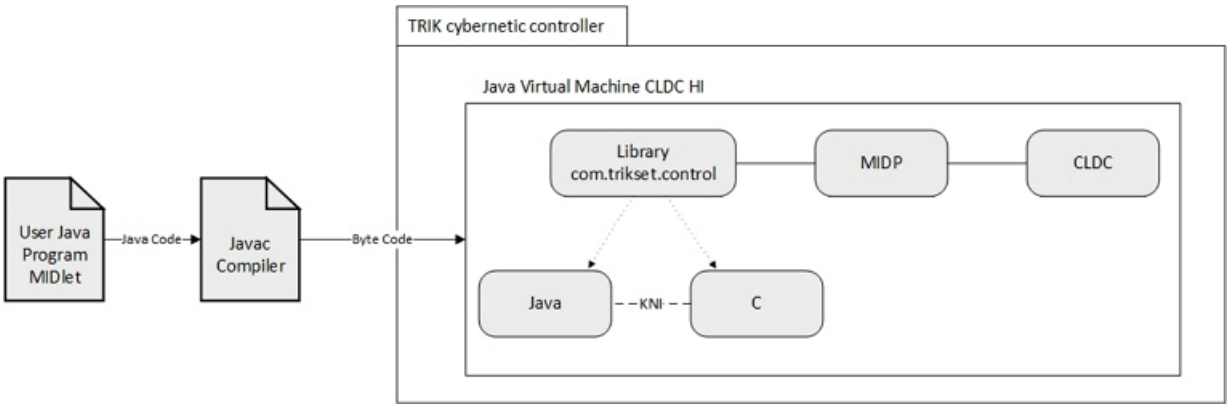

Рис. 6: Схема взаимодействия компонет

При сборке всех компонент в один исполняемый модуль появилась необходимость модифицировать компоненту midp, а именно – удалить блокировку приложения системой безопасности Java(Security). Это связано с тем, что в некоторых случаях CLDC HI не предоставляет приложению свободу действий. Например, при обмене данными по сети приложение запрашивает разрешение на выполнение действия. Приминительно к случаю удаленного управления роботом излишние проверки могут оказаться затруднительными, поэтому, не смотря на возможные угрозы, риск которых минимален, пришлось отказаться от проверок безопасности Java и дать пользователю контроль над безопасностью своего приложения.

Как происходит работа с библиотекой com.trikset.control?

Пользователь пишет программу на языке Java, которая должна иметь структуру MIDlet приложения, используя члены библиотеки com.trikset.control. С помощью компилятора javac программа компилируется в байт-код под версией Java 1.4, это

означает, что будут задействованы только классы Java Micro Edition. Виртуальная машина CLDC HI, которая собрана и установлена на контроллере ТРИК, компилирует на лету (DAC Dynamic Adaptive Compiler) и интерпретирует байт-код. Если байт-код содержит классы библиотеки com.trikset.control, то JVM обращается к Java реализации библиотеки, которая при необходимости может иметь низкоуровневую реализацию на языке C (native функции).

”Нативные” функции разрабатываемой библиотеки реализуются с помощью специального интерфейса KNI (K Native Interface), реализация которого является частью данной виртуальной машины. Этот механизм дает возможность вызова функций C/C++ из программы на Java [3].

5. Апробация на контроллере

В данной главе рассказывается об использовании виртуальной машины CLDC HI и библиотеки com.trikset.control на контроллере ТРИК.

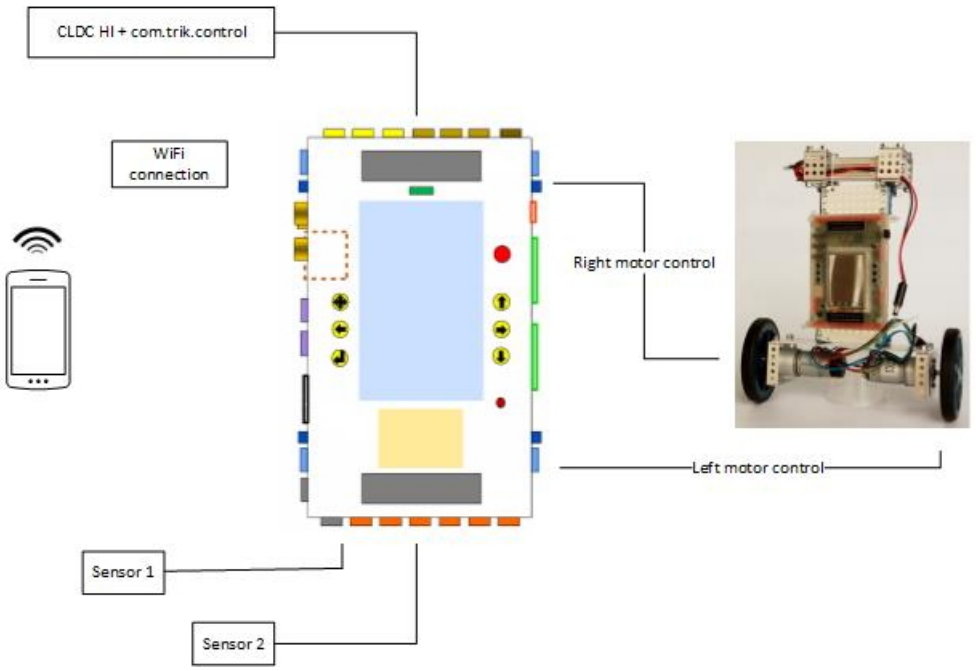

Рис. 7: Апробация на контроллере

На рис. 7 представлена концептуальная модель использования Java-машины CLDC HI. На изображении видно, что геймпад есть мобильное устройство, которое передает команды контроллеру по Wi-fi соединению. Контроллер принимает данные, обрабатывает и посылает на периферийные устройства, а именно датчики, моторы и др., сигналы, приводящие их в действие. Каждый выход на плате отвечает за определенное устройство (подробнее см. [5]).

На плату устанавливается собранная вместе с реализованной библиотекой виртуальная машина CLDC HI. С помощью специального конструктора, разработанного проектом ТРИК, собирается модель робота, а также программируется его логика на Java языке с использованием классов разработанного API на контроллере. Далее на плату должен быть перенесем Midlet, например, с помощью ssh-соединения.

Java-машина CLDC HI, собранная с модулем midp, может интерпретировать только Midlet-приложения. Класс, который будет являться мидлетом, должен расширять (extends) класс MIDlet. Этот класс должен иметь конструктор без параметров. Класс MIDlet имеет методы, предназначенные для управления жизненным циклом мидлета (рис. 8). Так для того, чтобы сообщить виртуальной машине о том, что мидлет завершается, используется метод notifyDestroyed(), а чтобы сообщить мидлету о том, что он будет завершен, виртуальная машина вызывает метод destroyApp(bolean uconditional). Мидлет может находится в состоянии паузы (paused - например, ко-

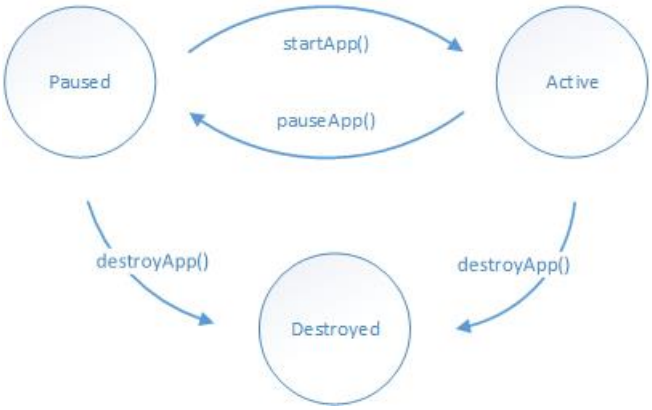

Рис. 8: Состояния, в которых может находиться мидлет

гда дисплей занят каким-нибудь сообщением и т.п.). Чтобы сообщить мидлету о том, что он переходит в состояние паузы, виртуальная машина вызывает метод мидлета pauseApp(), а чтобы войти в состояние паузы, мидлет использует метод notifyPaused(). Когда мидлет входит в активное состояние (выход из паузы и начало работы мидлета), вызывается его метод startApp(). Этот метод может вызываться несколько раз за время выполнения мидлета.

Для того, чтобы запустить мидлет на контроллере, необходимо выполнить следующие действия:

1. Получить уникальный номер для приложения с помощью утилиты installMidlet, сгенерированной при сборке виртуальной машины.

2. Начать выполнение приложения с помощью исполняемого файла runMidlet, принимающего аргументом уникальный номер Midlet-приложения.

Подробнее описано в статье [6].

Пример написания мидлета с помощью com.trikset.control

Для того, чтобы управлять, например, силовыми моторами на роботе, пользователю необходимо создать мидлет, в котором нужно создать экземпляр класса I2cCommunicator, т.к. силовые моторы общаются с контроллером по I2C шине, а затем создать экземпляр класса PowerMotor.

Код на Java:

```
public void startApp(){
    I2cCommunicator i2c = new I2cCommunicator("/dev/i2c-2", 0x48);
    i2c.connect();
    PowerMotor pw = new PowerMotor(i2c, 0x16, false);
    pw.setPower(40);
}
```

Объект i2c принимает при инициализации два параметра – имя файла I2C шины и адрес порта на плате. Объект pw принимает при инициализации три параметра – экземпляр класса I2cCommunicator для обмена данными, адрес порта на плате для мотора, булевский параметр, отвечающий за тип мотора (подробнее см. [5]).

Заключение

В ходе работы были получены результаты, представленные ниже.

- Произведен анализ предметной области и выявлено, что существующие решения — Java ME Embedded и JSR 256 Sensor — не предоставляют необходимый функционал для взаимодействия периферийных устройств с контроллером ТРИК.
- Была собрана виртуальная машина CLDC HI инструментарием контроллера ТРИК.
- С помощью языков программирования C и Java была разработана библиотека времени исполнения виртуальной Java машины CLDC HI для контроллера ТРИК. Библиотека является open source проектом и распространяется под GPLv2 лицензией.
- Разработанная библиотека апробирована на контроллере ТРИК.

В качестве дальнейшей работы над виртуальной машиной и разработанным API рассматривается вариант обобщения интерфейсов взаимодействия с периферийными устройствами и апробация на других системах, ограниченных в ресурсах.

Список литературы

[1] Oracle America Inc. — URL: http://www.oracle.com/ (дата обращения: 12.04.2014).

[2] Sarmanova S. Implementation of Java virtual machine CLDC HI runtime library for TRIK cybernetic controller // GitHub. — URL: https://github.com/Snezhana-Sarmanova/JavaTRIKControl/ (дата обращения: 27.05.2014).

[3] Sun Microsystems Inc. CLDC HotSpot™ Implementation Architecture Guide CLDC HotSpot Implementation. Version 1.1.3 Java™ ME Platform. — July 2006.

[4] Terekhov A.N., Luchin R.M., Filippov S.A. Educational cybernetical construction set for schools and universities. Advances in Control Education, Vol. 9, pp. 430–435. — 2012.

[5] Проект ТРИК // Всё о ТРИК. — URL: http://www.trikset.com/ (дата обращения: 12.04.2014).

[6] Сарманова С.Г. Сборка CLDC HotSpot Implementation для ARM // Habrahabr. — 2013. — URL: http://habrahabr.ru/post/184952/ (дата обращения: 12.04.2014).

[7] Терехов А.Н., Литвинов Ю.В., Брыксин Т.А. Среда для обучения информатике и робототехнике QReal:Robots. Девятая независимая научно-практическая конференция «Разработка ПО 2013» (CEE SEC(R)-2013). — Москва, 24.10.2013.