Московский государственный университет имени М.В.Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра системного программирования

Доледенок Максим Вадимович

Разработка средств активного наблюдения за состоянием бортовой операционной системы реального времени

Tools for active monitoring of airborne real-time operating system state

Выпускная квалификационная работа

Научный руководитель:
к.ф.-м.н.
Камкин Александр Сергеевич

Научный консультант:
Чепцов Виталий Юрьевич

Москва, 2023

Аннотация

Разработка средств активного наблюдения за состоянием
бортовой операционной системы реального времени

Доледенок Максим Вадимович

При эксплуатации бортовой операционной системы, предназначенной для применения на космических аппаратах, в программно-аппаратном комплексе должно возникать как можно меньше непредвиденных ситуаций. Одна из возможностей минимизировать количество таких ситуаций - это предварительная отработка полетного задания на программно-аппаратном комплексе в более контролируемых условиях на земле. Для большего контроля над системой во время отработки полётного задания требуется регулярно получать телеметрическую информацию изнутри системы. То есть получать данные из памяти системы в реальном времени. Также для моделирования исключительных ситуаций необходима возможность в реальном времени изменять состояние программного обеспечения.

В данной работе рассматривается метод разработки инструмента для доступа к памяти системы на примере операционной системы реального времени JetOS, соответствующей стандарту ARINC-653.

Abstract

Tools for active monitoring of airborne real-time operating system state

Max Doledenok

When operating an on-board operating system intended for use on spacecraft, as few unforeseen situations as possible should occur in the hardware and software complex. One of the ways to minimize the number of such situations is to pre-test a flight task on a hardware and software complex in more controlled conditions on the ground. For greater control over the system during the flight task, it is required to regularly receive telemetry information from inside the system. That is, to receive data from the system memory in real time. Also, to simulate exceptional situations, it is necessary to be able to change the state of the software in real time.

This paper discusses the method of developing a tool for accessing system memory using the example of the JetOS real-time operating system conforming to the ARINC-653 standard.

Содержание

1 Введение

1.1 ОСРВ

Операционные системы реального времени (ОСРВ) – это операционные системы, одним из важнейших требований к которым является выполнение поставленных задач за заранее определенное время. Это отличает их от известных пользовательских операционных систем, где фактор времени не так важен. ОСРВ используются там, где небольшая временная задержка может привести к существенным проблемам. Например, в авиакосмической отрасли, на некоторых производствах, в системах аварийной защиты.

1.2 ОСРВ JetOS

В данной работе рассматривается операционная система реального времени(ОСРВ) JetOS, разрабатываемая в Институте Системного Программирования Российской Академии Наук. ОСРВ JetOS предназначена для использования в авионике, в частности, в гражданских самолетах и спутниках. JetOS разрабатывается в соответствии со стандартом ARINC 653, который регламентирует временное и пространственное разделение ресурсов авиационной ЭВМ и определяет программный интерфейс, которым должно пользоваться прикладное ПО для доступа к ресурсам ЭВМ. Единицей планирования ресурсов является раздел, аналог пользовательской программы. Каждый раздел получает как процессорное время, так и некоторую часть оперативной памяти.

1.3 Конфигурация памяти в JetOS

В JetOS конфигурация памяти является статической, то есть раскладка памяти происходит на этапе загрузки операционной системы, исходя из заранее описанной пользователем конфигурации. При динамической конфигурации памяти пользовательские приложения могут заимствовать области оперативной памяти для временного использования. Но для ОСРВ это не всегда подходит, потому что сложно обеспечить детерминированность при работе с памятью. А для ОСРВ JetOS детерминированность является очень важным фактором, повышающим безопасность и отказоустойчивость операцион-

ной системы. Поэтому в JetOS применяется строго статическая конфигурация памяти. Причем, согласно стандарту ARINC 653, каждый раздел имеет своё адресное пространство, такое, что никакой другой раздел не имеет доступа к памяти этого адресного пространства. А ядро ОСРВ JetOS имеет доступ к памяти всех разделов с такими же правами, что и сами разделы.

1.4 Инструмент для активного наблюдения за состоянием ОСРВ

<Краткое описание инструмента, что он может, ещё раз зачем он(как в аннотации), к какой памяти имеет доступ, доступ к значениям переменных для удобства пользователя>

2 Цель работы, постановка задач

Целью данной работы является разработка и интеграция в ОСРВ JetOS инструмента для активного наблюдения за состоянием ОСРВ. Необходимо отслеживать значения заданных переменных.

На основе поставленной цели можно выделить набор задач, который необходимо выполнить в ходе работы:

- Необходимо изучить принцип доступа к памяти в ОСРВ JetOS.
- Придумать схему работы инструмента.
- Реализовать доступ на чтение памяти разделов и ядра ОС. Причем этот доступ должен осуществляться как по адресу, так и по имени переменной в разделе.
- Реализовать возможность изменения памяти разделов и ядра ОС. Причем этот доступ должен осуществляться как по адресу, так и по имени переменной.
- Реализовать возможность читать и изменять переменные сложных типов данных, таких как структуры,
- Реализовать интеграционный проект для тестирования .

3 Обзор существующих решений

3.1 JTAG

3.2 VxWorks

<Вставить скрин из документации.>

3.3 Иные

<Проприетарный инструмент РКК «Энергия» для ОСРВ RTEMS.>

4 Исследование и простроение решения задачи

4.1 Устройство памяти в ОСРВ JetOS

Основные принципы устройства памяти в ОСРВ JetOS:

- платформа имеет одно или несколько ядер процессора;
- набор регионов физической памяти общий для всех ядер;
- на одной платформе может быть запущено несколько модулей, здесь модуль – сконфигурированная операционная система, предназначенная для выполнения на одном или нескольких ядрах процессора на платформе;
- память разделяется на виртуальную и физическую;
- виртуальная память своя для каждого модуля, физическая – общая для всех модулей;
- виртуальная память разделяется на привилегированную (доступную только ядру) и непривилегированную (доступную и ядру, и разделам);
- физическая память разделяется на оперативную память и память физических устройств;
- и виртуальная, и физическая память делятся на страницы;
- размеры страниц одинаковы для виртуальной и физической памяти;
- адрес страницы пропорционален ее размеру.

4.2 Схема работы инструмента

<Картинка из презентации, и её подробное описание>

4.3 Задание отслеживания переменных

4.3.1 Задание статически в файле YAML

<Скрин примера с комментариями>

4.3.2 Задание динамически через консоль

<Формат, пример>

4.4 Протокол взаимодействия целевой системы и бортового вычислителя

4.4.1 Протокол посылки команд бортовому вычислителю

<Взять из документации, расписать поподробнее>

4.4.2 Протокол посылки данных целевой системе

<Взять из документации, расписать поподробнее>

4.5 Получение адреса переменной по её названию

4.5.1 Переменные простых типов

<Краткое описание ELF>

4.5.2 Переменные сложных типов

<Краткое описание DWARF>

5 Результаты работы

<Описание сделанного как в задачах> <Скрины успешного вывода программы>

6 Заключение

<Формальные слова>

Список литературы

[1] ARINC Avionics Standards // SAE ITC [Электронный ресурс]. — URL: https://www.aviation-ia.com/product-categories/600-series/ (дата обращения: 30.05.2022).

[2] Address Sanitizer // Clang Team [Электронный ресурс]. — URL: https://clang.llvm.org/docs/AddressSanitizer.html (дата обращения: 30.05.2022).

[3] SCons 4.3.0 // The SCons Foundation [Электронный ресурс]. — URL: https://scons.org/doc/production/HTML/scons-user.html (дата обращения: 30.05.2022).

[4] JetOS // Redmine [Электронный ресурс]. — URL: https://forge.ispras.ru/projects/jetos/repository/main/revisions/master/entry/README.md (дата обращения: 30.05.2022).

[5] JetOS allocator // Redmine [Электронный ресурс]. — URL: https://forge.ispras.ru/projects/jetos/repository/allocator/revisions/master/entry/README.txt (дата обращения: 30.05.2022).