УДК 004.451

Метод обеспечения устойчивости функционирования бортовой операционной системы реального времени

И. А. Дорошенко[1,2], В. Ю. Чепцов[2]

[1]*Московский физико-технический институт (национальный исследовательский университет)*
[2]*Институт системного программирования им. В. П. Иванникова Российской академии наук*

Современные операционные системы, используемые в аэрокосмической отрасли, должны обеспечивать высокую надежность и разрабатываются в соответствии со специальными стандартами. Одним из стандартов на программное обеспечение, используемое в бортовом оборудовании, является ARINC 653. В Институте Системного Программирования им. П. Иванникова РАН разрабатывается система, нацеленная на соответствие этому стандарту [1].

ARINC 653 регламентирует разделение ресурсов вычислительной системы в соответствии с принципами интегрированной модульной авионики и определяет необходимые сервисы, которые операционная система предоставляет прикладному программному обеспечению. Такой подход позволяет объединять на одном модуле несколько функциональных узлов, предоставляя каждому свой *раздел*. Распределение ресурсов, в т.ч. времени, в ARINC 653, происходит не динамически, а статически. В соответствии с этим, планирование происходит по заранее заданным *расписаниям*, которые распределяют время по *окнам* между разделами в рамках модуля. Окна расписания повторяются каждый *основной временной кадр*. Стандарт также определяет единую точку обработки отказов, называемую *монитором состояния*. Монитор состояния позволяет обрабатывать синхронные ошибки, то есть происходящие в ходе выполнения какой-то конкретной операции, но не предоставляет возможности для обнаружения асинхронных ошибок. [2]

В подобных операционных системах часто присутствуют подсистемы, называемые *BITE* (built-in test equipment) — встроенная система самопроверок, роль которой заключается в своевременном обнаружении асинхронных ошибок функционирования системы. Одним из обособленных компонентов этой подсистемы может выступать *watchdog-таймер* (WDT) — устройство, которое предназначено для обнаружения зависаний. Система, однажды запустив такой таймер, периодически взводит его заново. Если взвод не был произведен вовремя, WDT генерирует прерывание, позволяющее системе выйти из состояния зависания, или перезагружает систему. Использование таких таймеров широко распространено при проектировании встраиваемых систем, в том числе авиационных [3]. Тем не менее, существующие подходы к их использованию нацелены на защиту в основном от зависаний и взаимных блокировок *в прикладных программах* и основаны на использовании этого таймера прикладными программами.[4]

В рамках этого исследования разработан метод обеспечения устойчивости функционирования ОСРВ, основывающийся на использовании Watchdog-таймеров. Метод нацелен на защиту функций операционной системы, таких как системный таймер и системные разделы (то есть реализующие функции драйверов). Если системный таймер перестаёт генерировать прерывания или системные разделы не получают

управление, то система функционирует некорректно и должна быть выведена из этого состояния путем перезагрузки. Для того, чтобы отследить ошибки, связанные с непредоставлением системным разделам процессорного времени, предлагается в таких разделах периодически совершать специальный системный вызов, сообщающий о получении разделом управления.

Специальный компонент ядра *менеджер WDT*, ответственный за настройку и сброс таймера, располагается в ядре. Менеджер хранит битовую маску, в которую при совершении системного вызова устанавливаются биты. В конце каждого основного временного кадра менеджер проверяет, что все биты в этой маске установлены, и в таком случае производит сброс и обнуляет маску.
Системный интегратор определяет, какие биты может устанавливать тот или иной системный раздел. Раздел, совершая системный вызов, указывает, какой бит он просит установить, а менеджер проверяет, разрешено ли этому разделу устанавливать этот бит. Конфигурация доступных битов, времени таймаута WDT и других параметров определяется в XML-файлах конфигурации и не изменяется в течение работы модуля.

В данный момент идет разработка прототипа этого метода для ОСРВ ИСП РАН.

Список литературы

[1] Mallachiev K.M., Pakulin N.V., Khoroshilov A.V. Design and architecture of real-time operating system. Trudy ISP RAN/Proc. ISP RAS, vol.28, issue 2, 2016

[2] Goldberg, A., Horvath, G. Software Fault Protection with ARINC 653. 2007 IEEE Aerospace Conference.

[3] Tomayko, James E. Computers in spaceflight: The NASA experience. 1988

[4] MicroSin.net: Руководство по сторожевым таймерам для встраиваемых систем