ПРИНЦИПЫ РАБОТЫ ОПЕРАЦИОННЫХ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

Тыманович Н.А.

Белорусский государственный университет информатики и радиоэлектроники
Институт информационных технологий,
г. Минск, Республика Беларусь

Скудняков Ю.А. - к.т.н., доцент

Операционная система реального времени (*rtos*) – тип операционной системы, основное назначение которой – предоставление необходимого и достаточного набора функций для проектирования, разработки и функционирования систем реального времени на конкретном аппаратном оборудовании.

В настоящее время актуальной проблемой является решение задачи расширения больших проектов с алгоритмами, реализующими многопоточные вычисления. Многопоточность обычно

реализует планировщик операционной системы. Но под встраиваемые системы портирование полноценных операционных систем типа Linux является переизбыточным и влечет к огромным накладным расходам. Дело усугубляет то, что планировщик придется модернизировать под требования реального времени для своевременного ответа на внешние события и прочие факторы, которые нужно обработать с минимальной задержкой.

Данная работа посвящена разработке и изучению принципов работы операционных систем реального времени.

В качестве микроконтроллера был выбрал nrf52832 – микроконтроллер с современной arm-M4-архитектурой и богатой периферией, включая bluetooth low energy (ble).

В качестве языка программирования был выбран rust – быстрый язык программирования с эффективной работой с памятью: нет рантайма или сборщика мусора. Это позволяет использовать его во встраиваемых системах, где раннее использовались только C и C++ [1].

Перед тем, как начать разработку rtos, следует ознакомиться с целевой архитектурой, под которую необходимо ее создавать. В частности, особое внимание стоит уделить регистрам процессора.

Регистры arm-m3/m4-архитектуры представлены на рисунке 1.

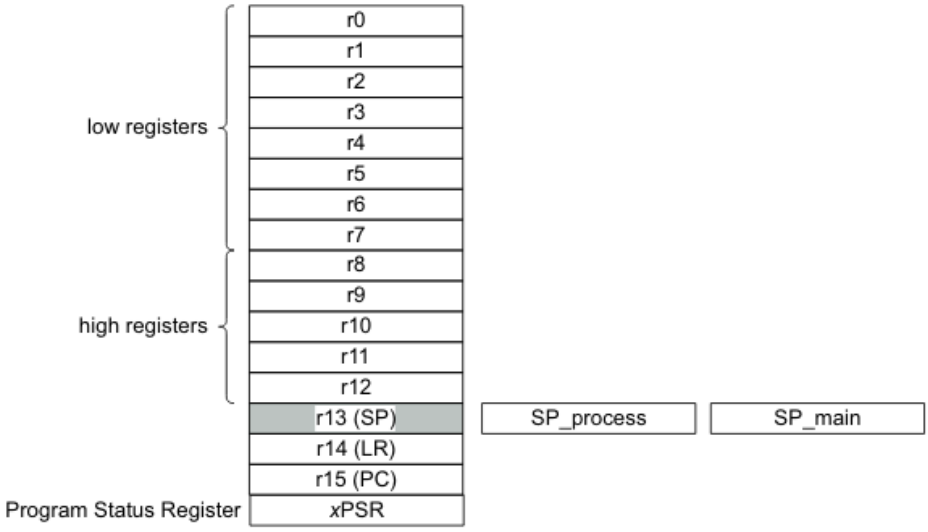

Рисунок 1 – Регистры arm-m3/m4-процессора

На рисунке 1 обозначены: 1) r0-r12 – регистры общего назначения. В них выполняется арифметика, хранятся временные значения и т.п.; 2) r13 – указатель на вершину стека; 3) r14 – указатель на адрес возврата, например, из функции; 4) R15 – указатель на выполняемую инструкцию; 5) XPSR – составной флаговый регистр, включающий в себя 3 флаговых регистра: регистр приложения (флаги арифметики), регистр прерываний (флаги прерываний) и регистр выполнения (флаги условного перехода и промежуточных выполнений во время прерывания).

Контекст потока, он же сам поток, есть набор значений регистров процессора. Чтобы переключить контекст/ поток, необходимо выгрузить в память содержимое всех регистров, а затем заполнить эти же регистры значениями другого потока. Если контекст/поток только начал свою работу, имеет смысл использовать только указатель стека (SP) и указатель выполняющейся инструкции (PC).

Проблема в том, что PC и xPSR постоянно изменяются по ходу работы, пусть даже если они явно не перезаписываются, что существенно усложняет схему их сохранения в нужный момент времени. Также сложность возрастает из-за того, что далеко не все регистры доступны для записи. Выход из такой ситуации – прерывание. Процессор во время прерывания сохраняет часть содержимого регистров, включая такие “проблемные” регистры как PC и xPSR, самостоятельно в стеке. При этом остается лишь сохранить оставшуюся часть содержимого самостоятельно. При выходе из прерывания процессор сам выгрузит содержимое всех тех регистров из стека, которое он сохранил в нем ранее. Лишь остается выгрузить ранее сохраненное содержимое регистров вручную из памяти, которое было сохранено самостоятельно до этого [2].

Процесс переключения потока изображен на рисунке 2.

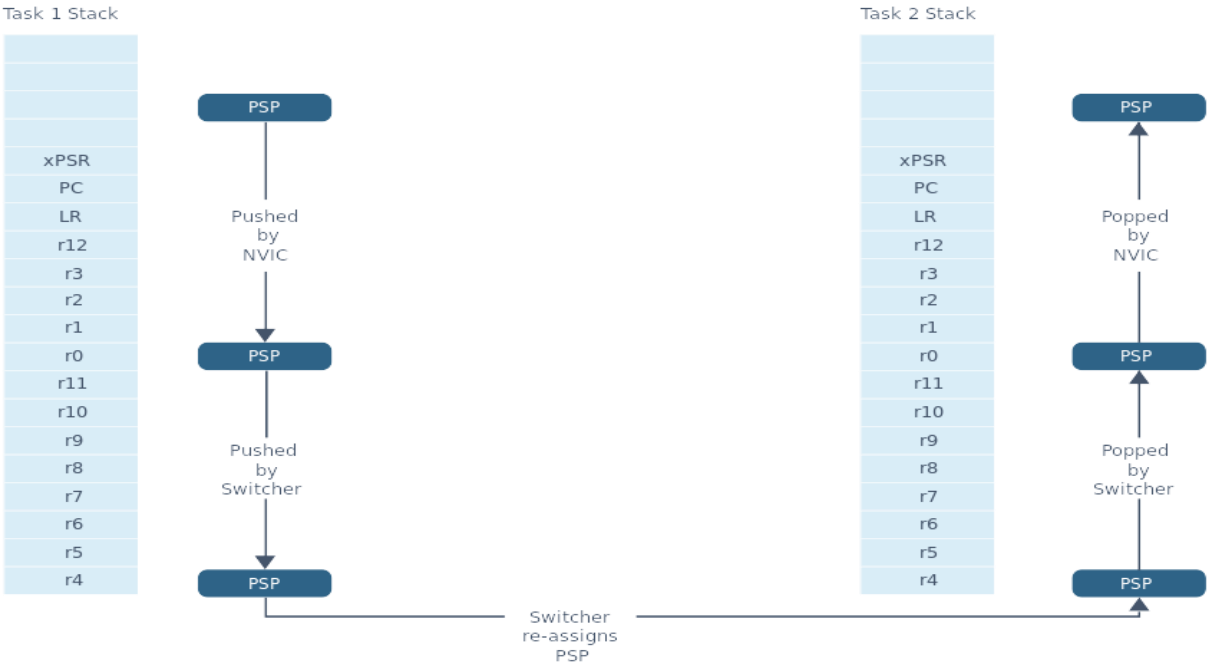

Рисунок 2 – Иллюстрация процесса переключения потока для arm-m3/4-архитектуры

Хоть общая схема переключения приблизительно одинакова для всех rtos и os, но конкретные релализации могут сильно отличаться в зависимости от конкретной архитектуры процессора, приложения и даже выбранного языка программирования.

Таким образом, были рассмотрены базовые принципы работы rtos на примере arm-m-архитектуры.

Список использованных источников:

1.Rust-lang describtion [Электронный ресурс]. – Режим доступа: https://www.rust-lang.org/ – Дата доступа: 13.02.2021.

2.Context Switching on the Cortex-M3 [Электронный ресурс]. – Режим доступа: https://blog.stratifylabs.co/device/ 2013-10-09-Context-Switching-on-the-Cortex-M3/ – Дата доступа: 12.11.2020.