УДК 621.375.018.78:681.3

ИСПОЛЬЗОВАНИЕ СТРУКТУР ДАННЫХ В ПЛАНИРОВЩИКЕ ОПЕРАЦИОННОЙ СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ

Зайцев С.А., к.т.н., доцент Корниенко С.К.
Запорожский национальный технический университет
zaitsev.serge@gmail.com

Планировщик – одна из наиболее критических и сложных составляющих операционных систем реального времени (Realtime operating system, RTOS). От архитектуры планировщика зависит быстродействие, надежность и эффективность работы системы. Из множества подходов к реализации планировщиков RTOS выделяют планировщики с событийно-ориентированным подходом. Этот вид планировщиков позволяет сократить количество процессов и при этом обеспечивает быстродействие, пропорциональное приоритетам поступающих событий, т.е. важные события обрабатываются в первую очередь, а события с менее высоким приоритетом продолжают оставаться в списке. Событийно-ориентированный подход гарантирует то, что обработчик событий не будет прерван в контексте процесса, а это позволяет избежать использования примитивов синхронизации, таких как семафоры. Кроме того, событийно-ориентированная парадигма обеспечивает то, что все поступившие события будут обработаны (в случае вложенных прерываний возможно переполнение стека в контексте прерывания).

Это реализуется за счет хранения очереди событий в динамической структуре данных. В большинстве существующих RTOS для таких целей используют двусвязный список. Таким образом, на один элемент структуры, описывающий событие, отводятся дополнительные указатели на предыдущий и последующий элементы. Двусвязные списки отличаются хорошей скоростью удаления и вставки элемента, однако поиск в них производится, как правило, последовательно, а это значит, что в случае поступления событий с убывающими приоритетами приходится пройти весь список для определения места вставки поступившего события [1].

Из существующих структур данных с этой проблемой намного лучше справляются бинарные деревья, а именно самобалансирующиеся деревья [2]. Основной принцип работы заключается в том, что если после вставки очередного элемента в дерево нарушается его баланс (заранее заданный некоторыми ограничениями), то производится реорганизация дерева – перестановки узлов, изменение их атрибутов и т.д. Сбалансированность красно-черных деревьев достигается за счет введения дополнительного свойства узла дерева — «цвет». Красно-чёрное дерево обладает следующими свойствами: корень и листья окрашены в чёрный цвет; любой красный узел имеет только чёрных потомков; все пути от корня к листьям имеют одинаковое число чёрных узлов. При этом для удобства листьями красно-чёрного дерева считаются фиктивные «нулевые» узлы, не содержащие данных.

В ходе исследования данной проблемы предпочтение было отдано красно-черным деревьям [3]. В большинстве реализаций они требуют дополнительные три указателя — на левого и правого потомка и на родительский узел. Кроме того, у

каждого узла имеется однобитный атрибут — цвет (красный или черный). Однако существуют реализации, которым достаточно двух указателей на дочерние узлы. В таком случае бит цвета можно хранить различными способами. Если в системе имеет место выравнивание памяти, то младшие биты адреса, как правило, равны нулю, а значит в них можно хранить указанный бит. Помимо этого, встраиваемые системы не обладают физически тем объемом памяти, который предусмотрен размером указателя. Например, очень малая доля 32-битных встраиваемых систем содержит 4 Гб памяти. Значит существуют адреса памяти, которым не соответствуют ее физические участки. Их и можно использовать для определения цвета. Также бит цвета может храниться в старших битах значения приоритета события (как правило, размером 1 байт) – для большинства задач достаточно 128 уровней приоритетов.

Для оценки эффективности работы этих алгоритмов был разработан программно-аппаратный комплекс, основанный на 8-битном контроллере ATMega8. Система генерирует события в обработчиках прерываний аппаратного таймера. Искусственно создаются некоторые случайные временные отклонения, чтобы события поступали неравномерно.

Система испытывалась в двух режимах. В режиме нормальной нагрузки события поступают достаточно редко, чтобы обрабатываться по мере поступления (таким образом, очередь событий в среднем содержит один элемент). В режиме кратковременной перегрузки поступает большое число событий с малым временным интервалом. При проведении экспериментов учитывалось время задержки обработки события, т.е. время, прошедшее с момента установки события в очередь до вызова обработчика этого события. Результаты группировались по приоритетам событий.

Данный эксперимент послужил наглядной демонстрацией использования очередей событий в архитектуре встраиваемых устройств. Красно-черные деревья показали в несколько раз меньшее время задержки (для высокоприоритетных событий время оказалось пренебрежительно малым – десятки машинных тактов), хотя их реализация содержит намного больше кода.

Таким образом, существуют предпосылки для использования красно-черных деревьев в событийно-ориентированных операционных системах реального времени.

Библиографический список

1. Альфред В. Ахо. Структуры данных и алгоритмы / Альфред В. Ахо, Джон Хопкрофт, Джеффри Д. Ульман. – М.: «Вильямс», 2000. – 384 с.

2. Дональд Кнут. Искусство программирования, том 3. Сортировка и поиск / Дональд Кнут. – М.: «Вильямс», 2007. – 824с.

3 Томас Х. Кормен. Алгоритмы: построение и анализ / Томас Х. Кормен и др. – М.: Издательский дом «Вильямс», 2007. – 1296с.