УДК 001.1

ИССЛЕДОВАНИЕ ТОЧНОСТИ ОТСЧЁТА ВРЕМЕННЫХ ИНТЕРВАЛОВ В СИСТЕМЕ WINDOWS

Братуха М.А., Шевченко О.Г.
ГВУЗ «Донецкий национальный технический университет», Украина
astr0n@ukr.net

Исследование разрешающей способности системного таймера Windows и его стойкости при нагрузках системы. В основе исследования лежат WinAPI-функции, которые позволяют программировать аппаратный таймер. В качестве объекта исследования выступают «ожидаемые таймеры».

Введение

Управление процессом предоставления ресурсов системы задачам, нитям, процедурам обработки прерываний и т.д. является одной из основных функций любой операционной системы и осуществляется при помощи такого механизма, как планирование. В системах реального времени планирование должно также гарантировать предсказуемое поведение, безопасность, возможность длительной, безотказной работы, выполнение задач в установленных временных рамках. От метода планирования во многом зависит успешная работа системы в целом.

Характерной особенностью операционной системы реального времени является управление ресурсами компьютера таким образом, чтобы определенные действия выполнялись периодически с одинаковым интервалом времени. Например, несвоевременное перемещение детали механизма, лишь по той причине, что это позволяют ресурсы системы, может привести к катастрофическим результатам, так же, как и невозможность осуществления перемещения этой детали вследствие занятости системы.

В данной работе исследуется возможность использования ОС Windows, как модельной среды при разработке и отладке ПО систем реального времени.

1 Способы измерения временных интервалов

Документированный набор функций Windows предоставляет несколько инструментов для задания и измерения временных интервалов:

1) *timeGetTime()* – возвращает время в миллисекундах с момента старта операционной системы («мультимедийный таймер»). Возвращаемое значение обнуляется каждые 49,71 дней. Точность измерения около 1-5 мс.
2) *GetTickCount()* – возвращает время в миллисекундах с момента старта операционной системы (обнуление каждые 49,71 дней). Является достаточно быстрой функцией, однако, точность измерения порядка 55 мс.
3) *QueryPerformanceCounter()* – «таймер высокого разрешения». Возвращает

«набежавшее» количество тиков. Частота работы этого таймера не меняется во время работы системы (порядка 1 МГц и более). Частота определяется с помощью функции QueryPerformanceFrequency().

4) *SetTimer()* – задаёт временной интервал и обладает небольшой точностью. Позволяет выполнять пользовательскую функцию при каждом срабатывании, однако нет возможности организовывать отложенные вызовы.
5) *WaitableTimer()* – «ожидаемый таймер». Обладает высокой точностью (порядка 1 мс). При срабатывании таймера возможен вызов пользовательской функции.

Из перечисленных функций интерес представляю те, которые позволяют задать желаемый интервал времени, по истечении которого необходимо выполнить определенные действия. В связи с этим в качестве объекта исследования был выбран, так называемые «ожидаемый таймер» (waitable timer). Программный интерфейс таймера предоставляет возможность указать пользовательскую функцию, которая по каждому тику должна запускаться на исполнение.

Ожидаемый таймер являются объектом ядра, что немаловажно для его точности и адекватности поведения. Данный объект может находиться в одном из двух состояний – быть занятым или свободным, при этом в первое из указанных состояний объект переходит сразу же после своего создания. Для сообщения таймеру момента перехода в свободное состояние используется функция SetWaitableTimer(). Переход таймера в свободное состояние означает завершение временного интервала. Согласно интерфейсу соответствующих API-функций, при программировании ожидаемого таймера может быть предусмотрена передача управление пользовательской (так называемой callback) функции. Вызываемая callback-функция ставиться в APC (asynchronous procedure call) очередь, а это означает, что каждый запрос на ее вызов будет обслужен системой. Однако, если длительность выполнения APC функции превышает величину временного интервала, отсчитываемую ожидаемым таймером, то следующий вызов этой функции будет выполнен только по завершению работы текущей. Для систем, выполнение действий в которых зависит от временных факторов, это может быть серьёзной проблемой. В Windows данная ситуация разрешается путем привлечения стандартных механизмов организации многопоточных приложения – ресурсоемкая часть callback функции локализуется в рамках одного или нескольких потоков.

По-умолчанию разрешающая способность системного таймера в Windows на большинстве рабочих станций составляет около 15,6 мс. Это означает, что даже высокоточный «ожидаемый таймер» не сможет измерять с точностью в 1мс на таких машинах, поскольку он также реализован на базе системного таймера. Обойти это ограничение можно с помощью специальных WinAPI-функций timePeriodBegin() и timerPeriodEnd(), которые позволяют задать точность системного таймера вплоть до 1 мс.

Поскольку системы класса Windows (исключая CE) изначально не являются системами реального времени, поэтому задачей исследования было выяснить – можно ли реализовать высокоточные измерения интервалов времени в Windows и что это за собой повлечёт, а также, какие условия должны при этом выполнятся.

2 Программа тестирования таймера

Для проведения полномасштабного тестирования возможностей системного таймера, была разработана программа, которая использовала WinAPI-функцию WaitableTimer в качестве связующего звена с системным таймером и позволяла тестировать таймер в различных условиях с учетом степени нагрузки и приоритетов приложений. Разработанная программа предоставляет следующие возможности:

– задание разрешающей способности таймера;
– установку временных интервалов между тиками таймера;
– указание количества тиков таймера (т.е. задание длительности выполнения программы);
– регулировку приоритетов нагрузочных потоков;
– тестирование с учётом приоритета текущего потока;
– моделирование степени нагрузки системы.

Программа разработана с консольным интерфейсом на языке C++. Тестирование проводилось в ОС Windows 7 SP1 и Windows XP SP3.

В ОС Windows предусмотрено 32 уровня приоритета потоков, которые составляют 6-ть классов приоритетов процессов, для тестирования таймера было выбрано три из них: низкий, высокий и нормальный.

Моделирование степени загрузки системы («легкая», «средняя», «тяжелая») осуществлялось путем увеличения количества функционирующих потоков – 0, 100, 400 соответственно. В качестве легкой степени загрузки принимается состояние системы на момент запуска однопоточного теста.

Для наглядности анализа результаты экспериментов представлены в виде графиков. Они позволяют оценить:

– количество ошибочных срабатываний таймера (то есть таких, при которых последующий тик таймера приходит позже установленного интервала);
– среднее время величины интервала.

На рис. 1 видно три линии, которые показывают зависимость количества ошибочных значений от количества выполненных тиков, каждая линия соответствует

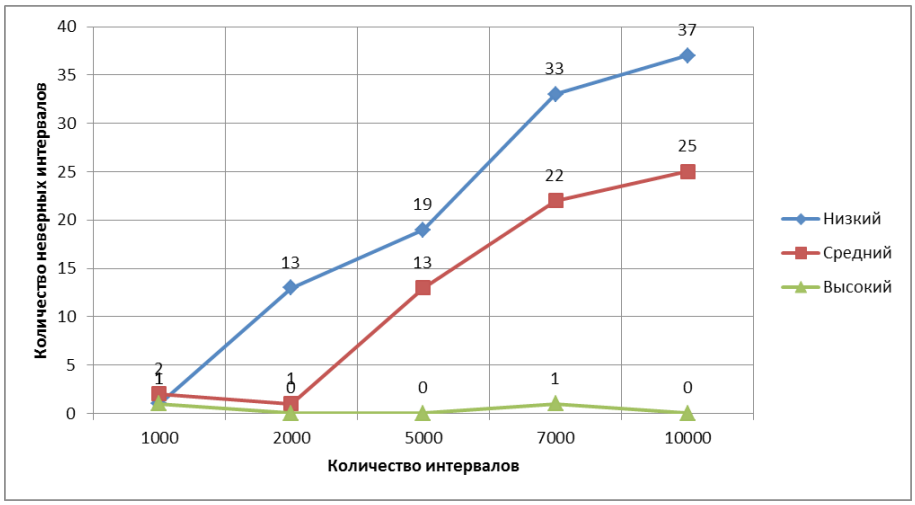

Рисунок 1. График зависимости ошибочных срабатываний таймера от их общего количества

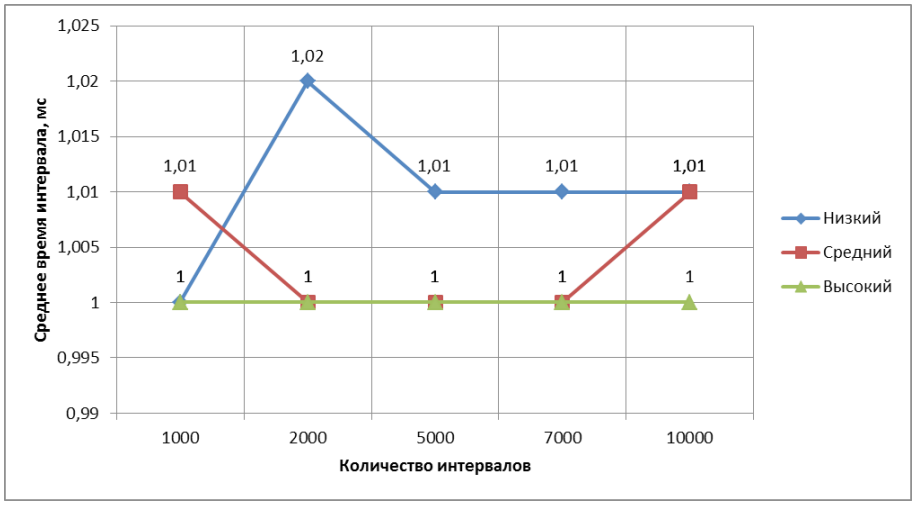

Рисунок 2. График зависимоти среднего времени интервала от общего количества таких интервалов

процессу с определённым приоритетом, в данном случае верхняя линия соответствует процессу с низким приоритетом, средняя – со средним приоритетом и нижняя – с высоким приоритетом. Представленные результаты позволяют сделать вывод о том, что качественные характеристики таймера зависят от приоритета выполняемой программы.

Рис. 2 демонстрирует зависимость среднего времени интервала от количества таких интервалов.

В ходе проведения экспериментов на системе Windows 7 было обнаружено, что если система сама устанавливает разрешающую способность таймера ниже, чем 15,6 мс, то увеличить ее не представляется возможным, пока сама ОС не сделает этого.

Выводы

Анализируя полученные данные, можно с определенной уверенностью сказать, что точность таймера зависит от приоритета главного потока, приоритетов нагрузочных потоков, а так же от количества таких потоков, то есть от общей нагрузки системы.

Поскольку на Windows XP не удается установить желаемую разрешающую способность таймера (даже используя функцию timePeriodBegin), то рекомендуется задавать интервал между тиками кратным времени, которое устанавливает система, в таком случае точность таймера будет высокой и количество ошибочных интервалов значительно уменьшиться.

Не смотря на то, что во время тестирования некоторые тесты приводили к эффекту «зависания» системы, а другие выполнялись слишком долго (в 10 раз больше, чем было задано), результаты тестов говорят о том, что на базе систем Windows можно реализовать достаточно высокоточные измерения и использовать ОС Windows, как модельную среду при разработке и отладке ПО для систем реального времени.

Список источников

[1] Руссинович М. и Соломон Д. Внутреннее устройство Microsoft Windows: Windows Server 2003, Windows XP и Windows 2000. Мастер-класс. / Пер. с англ. – 4-е изд. – М.: Издательство «Русская Редакция»; СПб.: Питер, 2008. – 992 стр.: ил.

[2] Рихтер Дж. Windows для профессионалов: создание эффективных Win32-приложений с учётом специфики 64-разрядной версии Windows/Пер. с англ. – 4-е изд. – СПб.: Питер; М.: Издательско-торговый дом «Русская Редакция», 2004. – 749 с.: ил.

[3] Примеры практического применение WinAPI-функций [электронный ресурс]. – Режим доступа: http://www.rsdn.ru.

[4] Документация по WinAPI-функциям [электронный ресурс]. – Режим доступа: http://www.msdn.microsoft.com.