Федеральное Агенство по Образованию

НИЯУ МИФИ

Обнинский Институт Атомной Энергетики

Факультет Кибернетики

*Кафедра АСУ*

Отчет по лабораторной работе №1

**«Объекты синхронизации WIN32»**

по дисциплине «Операционные системы»

**(Вариант №9)**

Выполнил:

Студент гр. ИС-С10

Смирнов А.А.

Проверил:

Пляскин А.В.

г. Обнинск, 2013

**Лабораторная работа №1.**

**Объекты синхронизации WIN32.**

**Задание:**

Реализовать в зависимости от варианта один из механизмов синхронизации задач с несколькими потоками выполнения.

**Требования:**

1. Использование только вызовов API

2. Желательно описание внешних вызовов ОС производить непосредственно в своем коде (разрешено пользоваться описанием windows.pas)

3. Обработка ошибок и их анализ по каждому вызову функций API

**Конечный результат:**

1. Получить требуемую в варианте величину.

2. Пакетный файл компиляции.

3. Отчет с выводами и анализом результатов. В анализ должны входить оценки вероятностных показателей измеренной в варианте величины.

**Вариант №9:** Организовать совместное использование мьютекса и события для защиты добавления элемента в очереди. Очередь произвольного формата. Событие сигнализирует чтение из очереди, а мьютекс - запись. Требуется снять статистику по времени использования объектов ядра.

**Описание используемых функций и процедур:**

Функция **WaitForMultipleObject** возвращается, когда происходит одно из следующего:

* любой или все указанные объекты в сигнальном состоянии
* время истекло

DWORD WaitForMultipleObjects(

DWORD nCount, // число дескрипторов в массиве

CONST HANDLE \*lpHandles, // указатель на массив дескрипторов

BOOL bWaitAll, // флаг ожидания

DWORD dwMilliseconds // временной интервал в миллисекундах

);

**Возвращаемые значения: Успех – событие; сбой – WAIT\_FAILED**

**Работа с объектами синхронизации**

Чтобы создать тот или иной объект синхронизации, производится вызов специальной функции WinAPI типа Create... (напр. CreateMutex). Этот вызов возвращает дескриптор объекта (HANDLE), который может использоваться всеми потоками, принадлежащими данному процессу. Есть возможность получить доступ к объекту синхронизации из другого процесса - либо унаследовав дескриптор этого объекта, либо, что предпочтительнее, воспользовавшись вызовом функции открытия объекта (Open...). После этого вызова процесс получит дескриптор, который в дальнейшем можно использовать для работы с объектом. Объекту, если только он не предназначен для использования внутри одного процесса, обязательно присваивается имя. Имена всех объектов должны быть различны (даже если они разного типа). Нельзя, например, создать событие и семафор с одним и тем же именем.

По имеющемуся дескриптору объекта можно определить его текущее состояние. Это делается с помощью т.н. ожидающих функций. Чаще всего используется функция WaitForSingleObject. Эта функция принимает два параметра, первый из которых - дескриптор объекта, второй - время ожидания в мсек. Функция возвращает WAIT\_OBJECT\_0, если объект находится в сигнальном состоянии, WAIT\_TIMEOUT - если истекло время ожидания, и WAIT\_ABANDONED, если объект-взаимоисключение не был освобожден до того, как владеющий им поток завершился. Если время ожидания указано равным нулю, функция возвращает результат немедленно, в противном случае она ждет в течение указанного промежутка времени. В случае, если состояние объекта станет сигнальным до истечения этого времени, функция вернет WAIT\_OBJECT\_0, в противном случае функция вернет WAIT\_TIMEOUT.

Если в качестве времени указана символическая константа INFINITE, то функция будет ждать неограниченно долго, пока состояние объекта не станет сигнальным.

Если необходимо узнавать о состоянии сразу нескольких объектов, следует воспользоваться функцией WaitForMultipleObjects.

Чтобы закончить работу с объектом и освободить дескриптор вызывается функция CloseHandle.

Очень важен тот факт, что обращение к ожидающей функции блокирует текущий поток, т.е. пока поток находится в состоянии ожидания, ему не выделяется процессорного времени.

**Вывод программы:**



**Вывод по лабораторной работе:** Мьютексы и события позволяют синхронизировать доступ к одной информации несколькими потоками (или процессами для мьютексов) – т.е. на примере данной работы один из двух потоков ждет сигнала освобождения либо от события, либо от мьютекса, сам занимает его и изменяет информацию.