Windows操作系统

C/C++ 程序实验

首都师范大学 信息工程学院

系统分析与管理实验室

## 实验五 Windows 内存结构

### 一、背景知识

Windows 的32位系统中，计算机CPU可以用32位地址对32位内存块进行操作。内存中的每一个字节都可以用一个32位的指针来寻址。这样，最大的存储空间就是232字节或4000兆字节 (4GB) 。这样，在Windows下运行的每一个应用程序都认为能独占可能的4GB大小的空间。

而另一方面，实际上没有几台机器的RAM能达到4GB，更不必说让每个进程都独享4GB内存了。Windows在幕后将虚拟内存 (virtual memory，VM) 地址映射到了各进程的物理内存地址上。而所谓物理内存是指计算机的RAM和由Windows分配到用户驱动器根目录上的换页文件。物理内存完全由系统管理。

### 二、实验目的

1) 通过实验了解Windows 内存的使用，学习如何在应用程序中管理内存，体会Windows应用程序内存的简单性和自我防护能力。

2) 了解Windows 的内存结构和虚拟内存的管理，进而了解进程堆和Windows为使用内存而提供的一些扩展功能。

### 三、工具/准备工作

在开始本实验之前，请回顾教科书的相关内容。

您需要做以下准备：

1) 一台运行Windows 操作系统的计算机。

2) 计算机中需安装Visual Studio Code。

### 四、实验内容与步骤

Windows提供了一个API即GetSystemInfo() ，以便用户能检查系统中虚拟内存的一些特性。清单5-1显示了如何调用该函数以及显示系统中当前内存的参数。

**步骤1**：登录进入Windows 。

**步骤2**：在“开始”菜单中单击“程序- Microsoft Visual Studio Code”。

**步骤3：**新建项目名为“5-1”，并且新建项“5-1.cpp”。

**清单5-1 获取有关系统的内存设置的信息**

// 工程vmeminfo

# include <windows.h>

# include <iostream>

# include <shlwapi.h>

# include <iomanip>

# pragma comment(lib, "shlwapi.lib")

void main()

{

// 首先获得系统信息

SYSTEM\_INFO si;

:: ZeroMemory(&si, sizeof(si) ) ;

:: GetSystemInfo(&si) ;

// 使用外壳辅助程序对一些尺寸进行格式化

TCHAR szPageSize [MAX\_PATH] ;

:: StrFormatByteSize(si.dwPageSize, szPageSize, MAX\_PATH) ;

DWORD dwMemSize = (DWORD) si.lpMaximumApplicationAddress -

(DWORD) si.lpMinimumApplicationAddress;

TCHAR szMemSize [MAX\_PATH] ;

:: StrFormatByteSize(dwMemSize, szMemSize, MAX\_PATH) ;

// 将内存信息显示出来

std :: cout << "Virtual memory page size: " << szPageSize

<< std :: endl;

std :: cout.fill ('0') ;

std :: cout << "Minimum application address: 0x"

<< std :: hex << std :: setw(8)

<< (DWORD) si.lpMinimumApplicationAddress

<< std :: endl;

std :: cout << "Maximum application address: 0x"

<< std :: hex << std :: setw(8)

<< (DWORD) si.lpMaximumApplicationAddress

<< std :: endl;

std :: cout << "Total available virtual memory: "

<< szMemSize << std :: endl ;

system("pause");

}

**步骤4：**按“F5”开始调试，注意路径里不要含有中文。

**步骤5：**按暂停按钮可暂停程序的执行，按终止按钮可终止程序的执行。



操作能否正常进行？如果不行，则可能的原因是什么？

操作能够正常进行，如果不行可能是文件路径包含中文字符，或者文件中有错误的字符格式。

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

运行结果 (如果运行不成功，则可能的原因是什么？) ：

1) 虚拟内存每页容量为：4.00KB\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2) 最小应用地址：0x00010000\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3) 最大应用地址：0xfffeffff\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4) 当前可供应用程序使用的内存空间为：3.99GB\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5) 当前计算机的实际内存大小为：15.8GB

徽标, 公司名称

描述已自动生成

图片 1 任务管理器中查看物理内存大小

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

阅读和分析程序5-1，请回答问题：

1. 理论上每个Windows应用程序可以独占的最大存储空间是：

理论上，在32位系统中可以独占的最大空间为4GB。

\_\_\_\_\_\_\_\_\_\_\_\_\_

1. 在清单5-1程序中，用于检查系统中虚拟内存特性的API函数是：

si.lpMinimumApplicationAddress 给出每个进程的最小应用地址0x00010000，在其开头空出64KB的保护区。

si.lpMaximumApplicationAddress 给出每个进程的最大应用地址0xfffeffff，在其结尾余下64KB的保护区。

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |
| --- |
| **提示：**可供应用程序使用的内存空间实际上已经减去了开头与结尾两个64KB的保护区。虚拟内存空间中的64KB保护区是防止编程错误的一种Windows方式。任何对内存中这一区域的访问 (读、写、执行) 都将引发一个错误陷阱，从而导致错误并终止程序的执行。也就是说，假如用户有一个NULL指针 (地址为0) ，但仍试图在此之前很近的地址处使用另一个指针，这将因为试图从更低的保留区域读写数据，从而产生意外错误并终止程序的执行。 |

选作：对于64位系统，说明如何修改程序以便显示相关信息。

文本

描述已自动生成

图片 2 修改代码查看64位系统的虚拟内存

文本

描述已自动生成

图片 3 代码输出结果

文本

描述已自动生成

图片 4 查看任务管理器的内存使用情况



图片 5 Windows官方学习文档提供资料

所以在64位系统中可用的最大虚拟内存为128TB，在我的电脑中，约有19GB的分页文件，剩余可利用的分页文件大小约为6GB，根据任务管理器查看可以验证该结论。