**山东师范大学**

**实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 院系部所 | 信息科学与工程学院 | | 课程名称 | | 计算机操作系统 |
| 所在班级 | 计工本1702 | 学号 | 201711010202 | 姓名 | 王汝芸 |
| 实验编号 | 09 | 题目 | 采用异步方式实现文件读写 | | |

一、实验目的和要求

了解Windows系统异步文件读/写的概念

熟悉Windows系统文件读/写相关API函数

掌握采用异步方式实现文件读/写的相关参数设置

二、实验环境

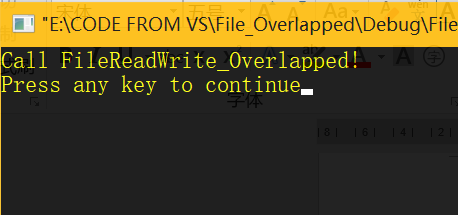
VC++6.0

Windows 10

三、实验内容及实施

建立一个函数，使用该函数将源文件source.txt中的内容读出，再写到目标文件overlapped.txt中去。

四、实验结果





五、实验分析

该实验完成异步方式的文件读/写操作。先创建两个文件，即soure.txt和overlapped.txt，然后反复从文件source.txt中读出数据块，并写到文件overlapped.txt中去，直到文件尾为止。

六、实验讨论（可选）

无。

七、附录 （可选）

1 *// File\_Overlapped.cpp : Defines the entry point for the console application.*

2 *//*

3

4 #include "stdafx.h"

5 #include "File\_Overlapped.h"

6

7 #ifdef \_DEBUG

8 #define new DEBUG\_NEW

9 #undef THIS\_FILE

10 **static char** THIS\_FILE[] = \_\_FILE\_\_;

11 #endif

12

13 DWORD BufferSize = 1024;

14 **char** buf[1024];

15

16

17 */////////////////////////////////////////////////////////////////////////////*

18 *// The one and only application object*

19

20 CWinApp theApp;

21

22 **using namespace** std;

23

24 **void** **FileReadWrite\_Overlapped**(**char**\*source, **char**\*destination);

25

26 **int** **\_tmain**(**int** argc, TCHAR\*argv[], TCHAR\*envp[])

27 {

28 **int** nRetCode = 0;

29 **printf**("Call FileReadWrite\_Overlapped!\n");

30 **FileReadWrite\_Overlapped**("source.txt", "overlapped.txt");

31 **return** nRetCode;

32 }

33

34 **void** **FileReadWrite\_Overlapped**(**char** \* source, **char** \* destination)

35 {

36 HANDLE handle\_src, handle\_dst;

37 DWORD NumberOfByteRead, NumberOfByteWrite, Error;

38 BOOL cycle;

39 **char**\*buffer;

40 buffer = buf;

41 OVERLAPPED overlapped;

42 *//CREATE FILE*

43 handle\_src = **CreateFile**(source,

44 GENERIC\_READ,

45 0,

46 NULL,

47 OPEN\_EXISTING,

48 FILE\_FLAG\_NO\_BUFFERING | FILE\_FLAG\_OVERLAPPED,

49 NULL);

50 handle\_dst = **CreateFile**(destination,

51 GENERIC\_WRITE,

52 NULL,

53 NULL,

54 CREATE\_ALWAYS,

55 NULL,

56 NULL);

57 **if** (handle\_src == INVALID\_HANDLE\_VALUE || handle\_dst == INVALID\_HANDLE\_VALUE)

58 {

59 **printf**("File Create Fail!\n");

60 **exit**(1);

61 }

62

63 cycle = TRUE;

64 overlapped.hEvent = NULL;

65 overlapped.Offset = -BufferSize;

66 overlapped.OffsetHigh = 0;

67

68 **while** (cycle)

69 {

70 overlapped.Offset = overlapped.Offset + BufferSize;

71 NumberOfByteRead = BufferSize;

72 *//READ FILE*

73 **if** (!**ReadFile**(handle\_src,

74 buffer,

75 NumberOfByteRead,

76 &NumberOfByteRead,

77 &overlapped))

78 {

79 **switch** (Error = **GetLastError**())

80 {

81 **case** ERROR\_HANDLE\_EOF:

82 cycle = FALSE;

83 **break**;

84 **case** ERROR\_IO\_PENDING:

85 **if** (!**GetOverlappedResult**(handle\_src,

86 &overlapped,

87 &NumberOfByteRead,

88 TRUE))

89 {

90 **printf**("GetOverlappedResult!%d\n", **GetLastError**());

91 **exit**(1);

92 }

93 **break**;

94 **default**:

95 **break**;

96 }

97 }

98

99 **if** (NumberOfByteRead < BufferSize)cycle = FALSE;

100 *//WRITE FILE*

101 **if** (!**WriteFile**(handle\_dst,

102 buffer,

103 NumberOfByteRead,

104 &NumberOfByteWrite,

105 NULL

106 ))

107 {

108 **printf**("Write File Error!%d\n", **GetLastError**());

109 **exit**(1);

110 }

111 }

112

113 *//CLOSE FILE HANDLE*

114 **CloseHandle**(handle\_src);

115 **CloseHandle**(handle\_dst);

116 }