**Lab Session # 4**

**IPC Using Named Pipes**

In this lab session, you will learn named pipes for windows for inter process communication. More specifically, we will focus on bidirectional communication. The following code creates two processes and communication between them. Visual C++ 6.0 IDE is recommended for editing/compiling the program(s).

**Here is the code for creating named pipe**#include <windows.h>

#include <stdio.h>

#include <string.h>

#define THE\_PIPE "\\\\.\\pipe\\testpipe"

void main()

{

HANDLE hIn;

DWORD dwBytesRead, dwWritten,len;

char buf[100];

hIn = CreateNamedPipe(THE\_PIPE, // Name

PIPE\_ACCESS\_DUPLEX, // OpenMode

PIPE\_TYPE\_BYTE|PIPE\_READMODE\_BYTE|PIPE\_WAIT, // PipeMode

PIPE\_UNLIMITED\_INSTANCES, // MaxInstances

4096, // OutBufferSize

4096, // InBuffersize

1000L, // TimeOut

NULL); // Security

if (hIn == INVALID\_HANDLE\_VALUE)

{

printf("Could not create the pipe\n");

exit(1);

}

printf("hPipe=%p\n", hIn);

printf("connect...\n");

ConnectNamedPipe(hIn, NULL);

printf("...connected\n");

for (int i=0;i<5;i++)

{

if (!ReadFile(hIn, buf, sizeof(buf), &dwBytesRead, NULL))

{

printf("ReadFile failed -- probably EOF\n");

break;

}

buf[dwBytesRead] = '\0';

printf("read [%s]\n", buf);

}

sprintf(buf, "Bye");

len = lstrlen(buf);

printf("Sending [%s]\n", buf);

if (!WriteFile(hIn, buf, sizeof(buf), &dwWritten, NULL))

{

printf("WriteFile failed -- probably EOF\n");

}

DisconnectNamedPipe(hIn);

CloseHandle(hIn);

}

**Code for Other Process**

#include <windows.h>

#include <stdio.h>

#define THE\_PIPE "\\\\.\\pipe\\testpipe"

void main()

{

HANDLE hOut;

char buf[1024];

DWORD len;

DWORD dwWritten,dwRead;

printf("pwrite: waiting for the pipe...\n");

if (WaitNamedPipe(THE\_PIPE, NMPWAIT\_WAIT\_FOREVER) == 0)

{

printf("WaitNamedPipe failed. error=%d\n", GetLastError());

return;

}

printf("pwrite: the pipe is ready\n");

hOut = CreateFile(THE\_PIPE,

GENERIC\_WRITE |GENERIC\_READ,

0,

NULL, OPEN\_EXISTING,

0,//FILE\_ATTRIBUTE\_NORMAL,

NULL);

if (hOut == INVALID\_HANDLE\_VALUE)

{

printf("CreateFile failed with error %d\n", GetLastError());

return;

}

printf("Opened the pipe\n");

for (int i = 0; i < 5; i++)

{

sprintf(buf, "This is test line %d so there.", i);

len = lstrlen(buf);

printf("Sending [%s]\n", buf);

if (!WriteFile(hOut, buf, len, &dwWritten, NULL))

{

printf("WriteFile failed\n");

break;

}

}

if (!ReadFile(hOut, buf, sizeof(buf), &dwRead, NULL))

printf("ReadFile failed -- probably EOF\n");

buf[dwRead] = '\0';

printf("%s\n", buf);

CloseHandle(hOut);

}