UserPort Documentation

1 Summary

`UserPort.SYS` is a kernel mode driver for Windows NT/2000 that gives usermode programs access to I/O Ports. This makes it possible to access hardware directly from a normal executable in the same way as under Windows 95/98/ME. This driver does not work on Windows 95/98/ME and there is really no need to run it anyway because I/O ports are always granted to usermode programs on these operating systems.

The driver can be used for the following purposes:

- To run software on Windows NT/2000 that normally only runs on Windows 95/98/ME.
- To easily access hardware like the parallel port and other I/O ports.

So what's the drawbacks with this wonderful software? Microsoft has for security reasons prohibited usermode access to I/O ports. Opening up I/O ports creates a big security hole in your system. You should therefore carefully set the grant lists to only give usermode access to the specific I/O ports you need. The default values opens up a wide range of I/O ports and you should narrow it down.

If you are writing your own software you should only grant access through the file "\\.\UserPort". Access is then given to your program when you open the file "\\.\UserPort". Other programs that don't open "\\.\UserPort" will not have access to these I/O ports.

2 Installation

The driver can be installed in the following two ways:

- Copy `UserPort.SYS` to `%WINDIR%\SYSTEM32\DRIVERS`
  Start `UserPort.EXE` and add the addresses you want and remove the others and click on start.
- Run `UserPort.EXE` with the driver filename and path as an argument
  i.e. run `UserPort.EXE X:\YOURDIR\UserPort.SYS`
  Add the addresses you want and remove the others and click on start.

You should now have usermode access to the addresses you have chosen.

3 Examples

Port instructions are not included in development environments (such as Visual C++ and Delphi) because direct I/O access isn't allowed by the operating system. You will therefore need to include a portion of assembler code into your software in order to access your hardware, see Figure 2, 3 and 4.

```
BYTE inportb(UINT portid)
{
  unsigned char value;

  __asm mov edx,portid
  __asm in al,dx
  __asm mov value,al
  return value;
}
```

Figure 2: Read I/O port

```
void outportb(UINT portid,
BYTE value)
{
  __asm mov edx,portid
  __asm mov al,value
  __asm out dx,al
}
```

Figure 3: Write I/O port

```
if (inportb(0x379) & 0x10) {            // Check "Select" pin
  outportb(0x378,'A');                  // Write character 'A' to printer
  outportb(0x37a,inportb(0x37a) | 0x01); // Set "strobe"
  Sleep(1);                             // Wait 1ms
  outportb(0x37a,inportb(0x37a) & 0xfe); // Clear "Strobe" pin
}
```

Figure 4: Print 'A' example using direct I/O access

Figure 4 shows how simple it now is to access hardware from a usermode program. The UserPort package should contain the files IOPort.c, IOPort.h and IOPort.pas to be when developing C, C++ and Delphi programs.

4 Technical Description

The driver gives user mode program access to selected ports by changing the x86-processors IOPM (I/O Permission Map). Figure 1 shows how the driver works. For a detailed description on the TSS see Intel proccessor handbooks.

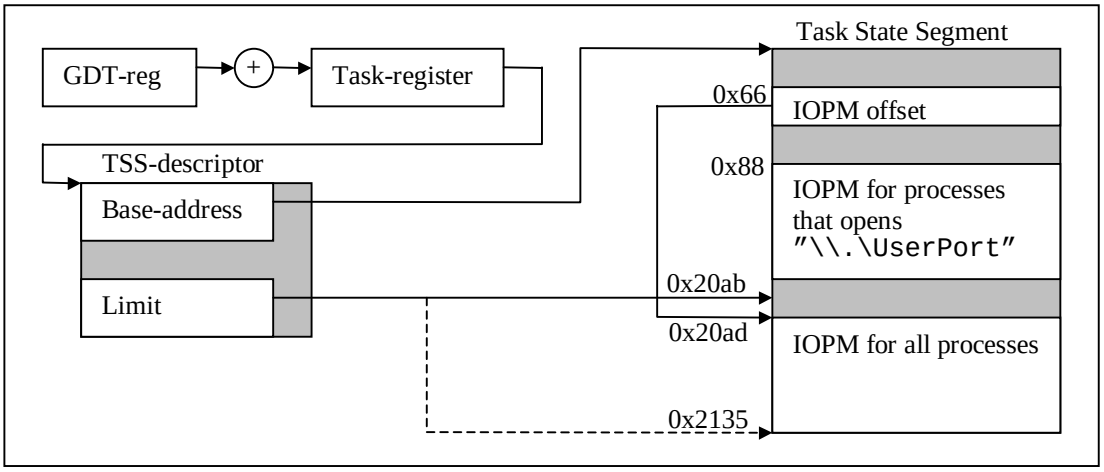

Figure 1: 80x86 TSS description

The original size of the TSS is 0x20ab and the driver extends it to 0x2135. The default IOPM offset is 0x20ad and this value is rewritten by the OS on every task switch. The IOPM offset must therefore be changed with the undocumented function `Ke386IoSetAccessProcess`, which sets the IOPM offset to 0x88. The AllProcessesIOPM is written to 0x20ad because this is the default IOPM offset for all processes and the ThroughCreateFileIOPM is written to 0x88 because the `Ke386IoSetAccessProcess` function sets the IOPM offset to 0x88. The `Ke386IoSetAccessProcess` function is called when a user mode program opens the file "\\.\UserPort". The driver loads the two IOPM:s from:
`HKEY_LOCAL_MACHINE\Software\UserPort\AllProcessesIOPM`
`HKEY_LOCAL_MACHINE\Software\UserPort\ThroughCreateFileIOPM`
It will use default values below if these doesn't exist.

This driver is influenced and inspired by an article written by Dale Roberts 8/30/95, published in May 96 Dr Dobbs Journal, see www.ddj.com.

Tomas Franzon
tomas_franzon@hotmail.com