Serial port support in Windows CE / Pocket PC: Bugs and caveats

Jelmer Vernooij

2003

Abstract

1 Introduction

Getting serial port support under Windows CE to work is a job not easily done. It took me several weeks to get it to work properly.

The information provided here is just a collection of random things I ran into.

I'm not sure what causes these bugs or flaws — for some of them I'm sure they're caused by the operating system (Windows CE), but they might also be caused by the hardware I was using (Compaq iPAQ).

The information in this document applies to (at least) eMbedded Visual C++ 3.0 in combination with the Pocket PC, Pocket PC 2002 and Pocket PC 2003.

2 API

The COM port interface under Windows CE is supposed to imitate the interface available under Windows 9x and NT.

COM ports are like 'virtual files' under Windows.

Consult MSDN for information about this API. Also, there are plenty of good tutorials on serial port programming under Windows, so I won't bother to write another one.

3 Bugs

3.1 cbInQue

The element cbInQue of the COMSTAT structure stays 0 after it has been retrieved using ClearCommError(), even when data is available!

This basically means you cannot use ClearCommError() to check whether there is any data waiting in the buffer.

3.2 Multithreading and the emulator

While serial communication is supposed to work from multiple threads, this is not supported on the emulator. If you're using WaitCommEvent() from one thread (for example) and WriteFile() or ReadFile() from another, the other thread will block until WaitCommEvent() is done.

This makes working with WaitCommEvent() practically impossible from the emulator. However, code like the example below can be used to not use WaitCommEvent() on the emulator. Instead of WaitCommEvent(), just call ReadFile() in a loop on the emulator. The emulator is fast enough (on most PC's at least) to not slow down the computer considerably.

```
#ifndef _WIN32_WCE_EMULATION
unsigned long event;
if(m_hComm) {
SetCommMask(m_hComm, EV_ERR | EV_RXCHAR);
WaitCommEvent(m_hComm, &event, NULL);
}
#endif
```

4 Other special points of attention

4.1 Frequently read the buffer

Read the buffer often. The internal buffer isn't very big and if you get a buffer overrun, you'll lose data.

4.2 SetupComm() is only used for "advices"

Buffer sizes as set by using SetupComm() are, as the MSDN documentation describes, only used as an 'advice'. It looks like the buffers are not really resized

4.3 Too frequent read operations

If you try to read too often (e.g. doing a ReadFile() in a while loop in a seperate thread), you'll notice that the thread doing the reading will slow down your pocket PC considerably.

4.4 COM port names

'Special' port names as supported by Windows 9x, 2000, etc are not supported. This has a reason, though. The pocket PC does not have different 'drives'. It has one general root, "\", and this makes "\\.\$COMx$" a valid file name. This would cause naming conflicts (files and com ports with the same name).

The following code can be used to circumvent this problem:

```
int i;
#ifndef _WIN32_WCE
        portname.Format(TEXT("\\\\.\\COM%d"), i);
#else
        portname.Format(TEXT("COM%d:"), i);
#endif
```

5 Comments

If you have comments or if you're aware of other bugs, please let me know .