GO Software Pty Limited
Map: 27 Tacoma Blvd, Pasadena SA 5042
Phn: 0403-063-991 Fax: none
ABN: 54-008-044-906 ACN: 008-044-906
Eml: support@gosoftware.com.au Web: www.gosoftware.com.au

General Protection Fault in module WIN87EM.DLL at 0001:02C6

Recently ran into an issue with a Windows 7 32-bit VM running in VMware Fusion on Mac OSX 10.9 (Mavericks) when running a Point of Sale application would receive a Application has caused a general protection fault in module WIN87EM.DLL. Starting doing some digging and found this thread discussing the issue at http://social.technet.microsoft.com/Forums/en-US/e0922437-0dc7-4091-a8f7-fadc7fb5bbb5/general-protection-fault-in-module-win87emdll-while-running-windows-xp-mode?forum=w7itprovirt.

Basically the issue is that when you try to run a legacy 16-bit application in Windows XP and above you will receive the GPF (General Protection Fault).

The Windows 80×87 emulator library, WIN87EM.DLL, works at the 16-bit-Windows level to virtualize the coprocessor among multiple Windows-based applications that run inside the system VM.

It appears to me that, with the Virtual Machine using a Virtualized Processor (not related to the above reference to "virtualized coprocessor") as opposed to an emulated processor, the Virtual Math Coprocessor Device (VMCPD) and/or WIN87EM.DLL generate an error from the virtualized processor.

So how do we fix it. Simple enough you just have to to "hiding" the math coprocessor from the Virtual Machine, so it doesn't go to the processor. This apparently can be achieved by using a program called WinFloat which includes a tool called HIDE87, which is suppose to hide the math coprocessor from the kernel.

Lucky for us you can download winfloat from

http://www.conradshome.com/win31/archive/ or
www.gosoftware.com.au/download/winfloat.exe.

- Click on **winfloat.exe** to download to a directory. Once downloaded double-click to extract the files.
- Copy **HIDE87.com** to **c:\windows\system32** directory
- Add **lh c:\windows\system32\HIDE87.com** as the first line to the **c:\windows\system32\autoexec.nt** file.
- Reboot virtual machine

The error should now be gone. This should work with virtual machines running on Virtualbox and VMWare.

Another Solution

1. I created a R:\VB\Test folder
2. I copied R:\VB\Rentmstr files to it.
3. I copied R:\VB\Standard files to it.
4. I wrote a program (VB_TEST.exe) to read all of the *.FRM and *.BAS files anmd add an extra line into every function and subroutine to debug.print "{modulname}: {FunctionName/SubName)"
   Note that to make the program simpler I hard coded the file list (*.FRM{*.Bas) into VB_Test and exported the *.frm files to *.FRM2 and the *.Bas files to *.BAS2
   Then I copied *.frm2 over the top of *.frm and *.Bas2 files over the top of *.bas files once I was happy that VB_Test had made the correct changes.
5. I mapped drive R: on the machine with the problem to \\Diskstation\GlynR
6. I copied the R:\Lic files to C:\Windows folder on the machine with the problem
7. I copied the R:\Lic files to C:\Windows\System folder on the machine with the problem
8. I copied the R:\Lic files to C:\Windows\System32 folder on the machine with the problem
9. I ran VB3, loaded RentMstr.VBP and ran it in interactive mode
10. I waited until the GPF while watching the debug window scrolling the progress
11. The last line in the debug list when the GPF occurred was the routine which ased the GPF

VMWare Solution

There are a few applications which may fail with this error when executed in a VM. The problem is that the library tries to read the instruction bytes of the last FPU instruction executed based onhe code segment and instruction pointer saved in the FPU environment. Unfortunately, the code segment saved in the FPU environment may be NULL.

When VMWare switches between the virtual machine monitor and the host operating system, VMWare uses the primitive hardware instructions to save and restore the FPU state. These instructions were not really designed for an environment that mixes 32-bit and 64-bit execution. When the instructions are executed in 64-bit mode, they don't keep track of the code segment, since segments have little meaning in 64-bit mode. The virtual machine monitor, which saves and restores the FPU state for the guest VM, executes in 64-bit mode. Hence, the FPU code segment is lost.

Fortunately, there is a workaround for this issue. If you add the following option to your configuration file, VMWare will make sure that the FPU code segment is properly saved and restored when switching between the virtual machine monitor and the host operating system (at some small performance penalty):

```
monitor_control.enable_rigorous_fpu_save_restore = TRUE
```

Video Driver win87em.dll

This is the step by step resolution to the problem we had with the "win87em.dll" issue.

1. Left-Click the START button in the bottom left corner of the screen.
2. Right-Click My Computer and left click Properties.
3. Left-Click the tab at the top that says Hardware
4. Left-Click the button that says Device Manager.
5. Left-Click the + sign next to Display Adapters near the top of the list.
6. Right-Click the items shown in the expanded list under Display Adapters and left-click Disable.
7. Left-Click the Yes button that shows when windows asks if you are sure you want to disable it.
8. Left-Click the No button when windows asks if you want to reboot.
9. Repeat the disable process for each item listed under Display Adapters (usually only one or two)
10. Reboot the PC and the win87em.dll General Protection Fault errors should go away.

This is only applicable for users on Windows XP. Most likely the display adapters listed will be shown as an Intel G41 internal display adapter, but it may be nother Intel device. If this does not fix the issue then it is likely a bad printer driver causing the problem.

Disabling the video adapter will not hurt windows. It will make their computer unable to watch videos or play 3D games, but windows will still run and look fine. (They will probably need to change their screen resolution after rebooting.)

Another Fix

The WW0548. EXE file contains a revised Windows WIN87EM. dll file. , you must use Windows version 3.1 to use this file Dynamic Link Library (DLL). Do you have a version of Windows than 3.1, you do not need to replace the current WIN87EM. dll file. , you also do not need to replace the WIN87EM. dll file when you use the math coprocessor Intel 80387. 80486DX Intel processors and 80486DX2 is not the new file is required. Installation The following file is available for download from the Microsoft Download Center: DOWNLOAD Ww0548.exe download now For more information about how to download Microsoft support files, click the following article number to view the article in the Microsoft Knowledge Base: 119591 How to obtain Microsoft support files from online services Microsoft has tested this file for viruses. , Microsoft has used the currently available at the time of publication of file virus detection software. The file is stored on security-enhanced servers that help to prevent any unauthorized changes to the file. Download the WW0548. exe file from the Microsoft Download Center. Double-click the WW0548. exe file to extract the contents. Exit Windows. WIN87EM The. dll file that is used by Windows and not updated to be replaced while Windows is running. Copy the WIN87EM. DLL file in the Windows \ System folder. Restart the computer. The new WIN87EM. dll file has now been installed and will be used when you restart Windows.
