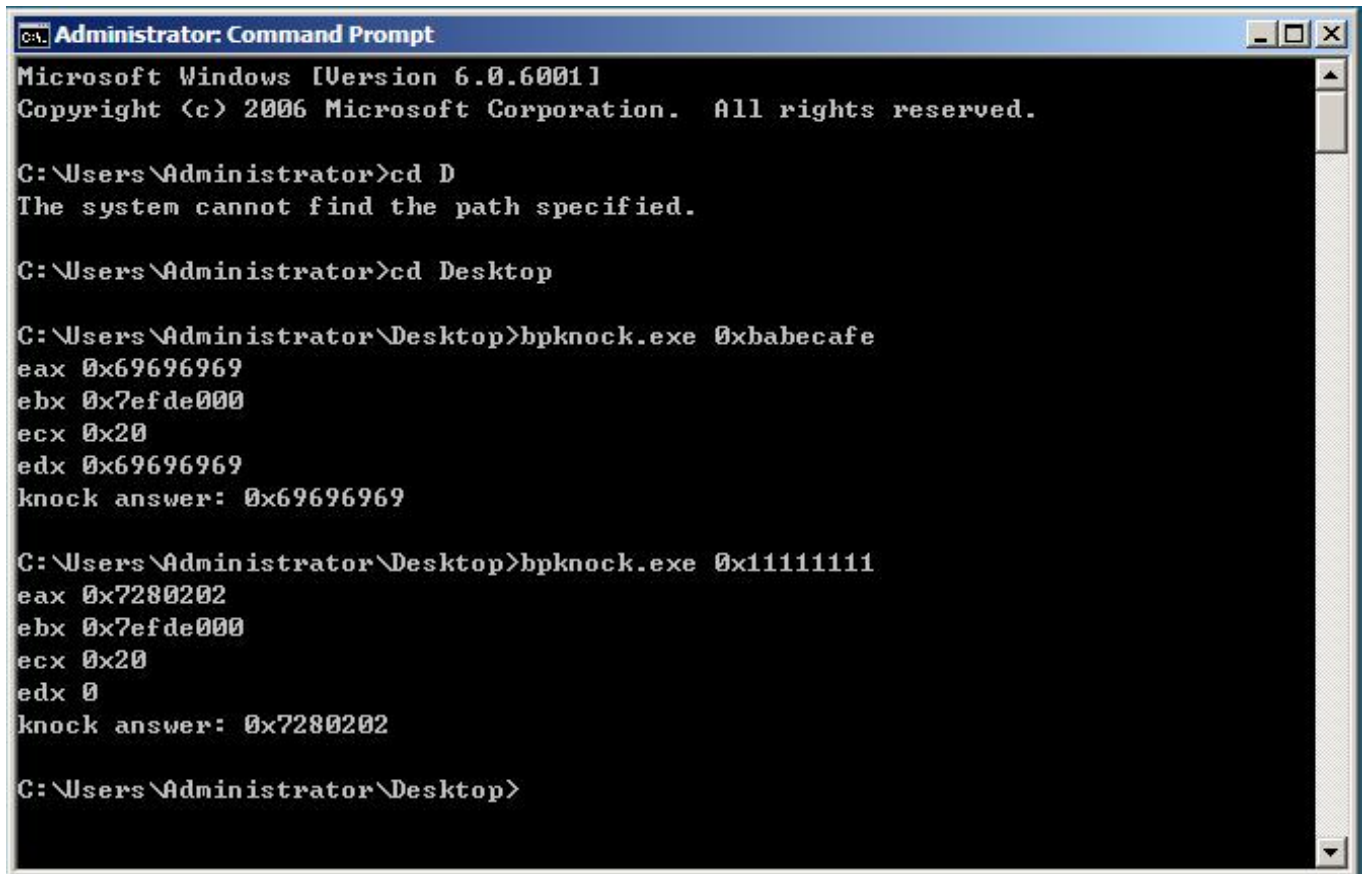


两个测试的例子：

1.修改了 CPUID 的寄存器输出值，只需通过 GuestRegs->rax =...来设置执行 CPUID 后 rax 寄存器的值就可以了。以下图运行效果为例：



```
Administrator: Command Prompt
Microsoft Windows [Version 6.0.6001]
Copyright (c) 2006 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>cd D
The system cannot find the path specified.

C:\Users\Administrator>cd Desktop

C:\Users\Administrator\Desktop>bpknock.exe 0xbabecafe
eax 0x69696969
ebx 0x7efde000
ecx 0x20
edx 0x69696969
knock answer: 0x69696969

C:\Users\Administrator\Desktop>bpknock.exe 0x11111111
eax 0x7280202
ebx 0x7efde000
ecx 0x20
edx 0
knock answer: 0x7280202

C:\Users\Administrator\Desktop>
```

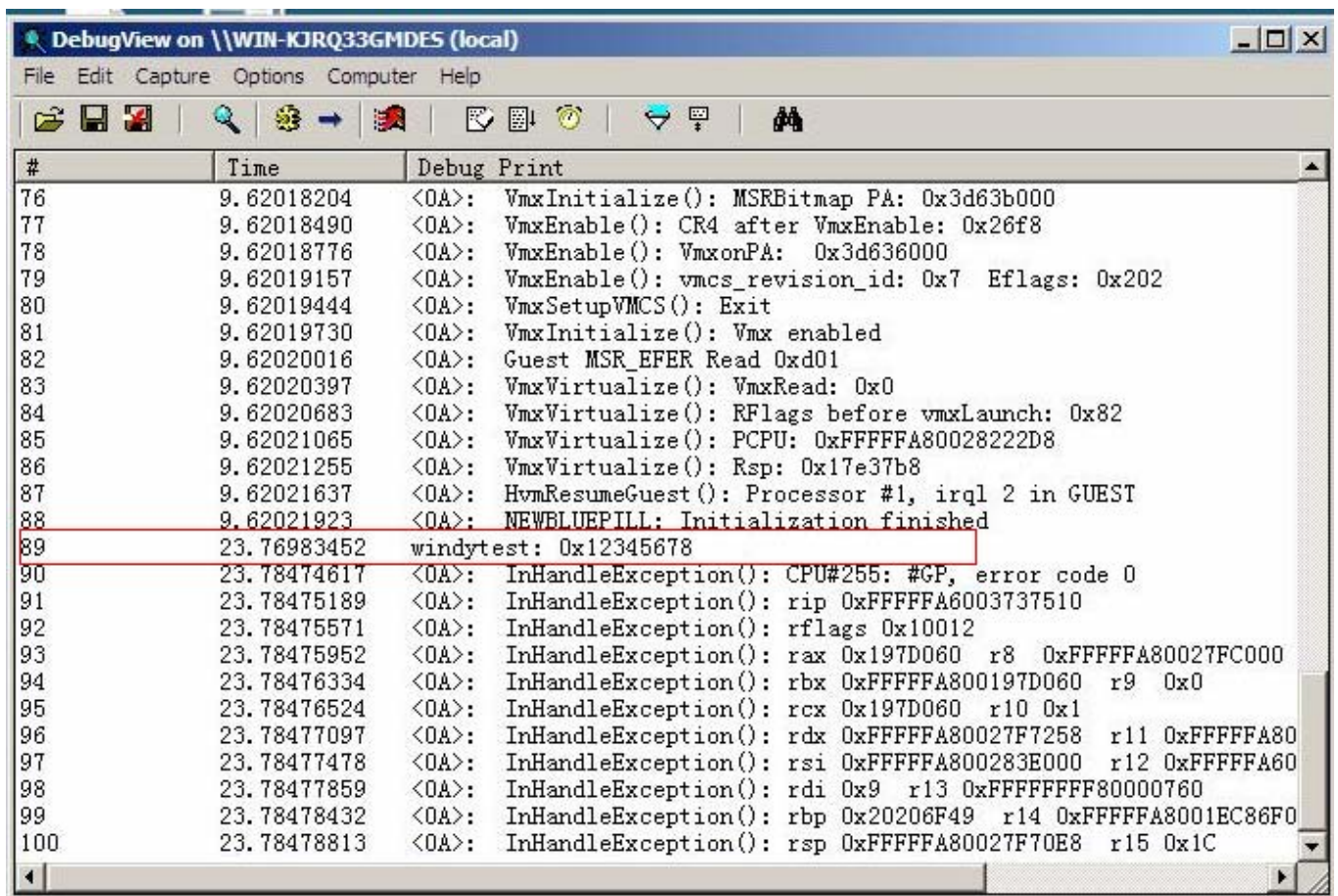
主要代码

```
#ifdef BP_KNOCK
    if (fn == BP_KNOCK_EAX) {
# if DEBUG_LEVEL>3
        _KdPrint (("Magic knock received: %p\n", BP_KNOCK_EAX));
# endif
        GuestRegs->rax = BP_KNOCK_EAX_ANSWER;
        GuestRegs->rdx = BP_KNOCK_EAX_ANSWER;//这句是添加的
        return TRUE;
    }
#endif
```

然后执行 bpKnock.exe，通过打印的寄存器值可以看出效果。

执行 bpknock.exe 0xbabecafe 时，执行到了添加的那句代码，rdx 寄存器的值就被修改成与 rax 一样的值。执行 bpknock.exe 0x11111111 时，由于不会执行到添加的代码，寄存器的值默认为调用 CPUID 后的寄存器的值的情况。

2.测试 MsrRead 的 trap，由于 rdmsr 只能在 ring0 层执行，因此通过写一个驱动程序，在驱动中调用 rdmsr 即可触发 vmexit。



#	Time	Debug Print
76	9.62018204	<OA>: VmxInitialize(): MSRBitmap PA: 0x3d63b000
77	9.62018490	<OA>: VmxEnable(): CR4 after VmxEnable: 0x26f8
78	9.62018776	<OA>: VmxEnable(): VmxonPA: 0x3d636000
79	9.62019157	<OA>: VmxEnable(): vmcs_revision_id: 0x7 Eflags: 0x202
80	9.62019444	<OA>: VmxSetupVMCS(): Exit
81	9.62019730	<OA>: VmxInitialize(): Vmx enabled
82	9.62020016	<OA>: Guest MSR_EFER Read 0xd01
83	9.62020397	<OA>: VmxVirtualize(): VmxRead: 0x0
84	9.62020683	<OA>: VmxVirtualize(): RFlags before vmxLaunch: 0x82
85	9.62021065	<OA>: VmxVirtualize(): PCPU: 0xFFFFFA80028222D8
86	9.62021255	<OA>: VmxVirtualize(): Rsp: 0x17e37b8
87	9.62021637	<OA>: HvmResumeGuest(): Processor #1, irq1 2 in GUEST
88	9.62021923	<OA>: NEWBLUEPILL: Initialization finished
89	23.76983452	windytest: 0x12345678
90	23.78474617	<OA>: InHandleException(): CPU#255: #GP, error code 0
91	23.78475189	<OA>: InHandleException(): rip 0xFFFFFA6003737510
92	23.78475571	<OA>: InHandleException(): rflags 0x10012
93	23.78475952	<OA>: InHandleException(): rax 0x197D060 r8 0xFFFFFA80027FC000
94	23.78476334	<OA>: InHandleException(): rbx 0xFFFFFA800197D060 r9 0x0
95	23.78476524	<OA>: InHandleException(): rcx 0x197D060 r10 0x1
96	23.78477097	<OA>: InHandleException(): rdx 0xFFFFFA80027F7258 r11 0xFFFFFA80
97	23.78477478	<OA>: InHandleException(): rsi 0xFFFFFA800283E000 r12 0xFFFFFA60
98	23.78477859	<OA>: InHandleException(): rdi 0x9 r13 0xFFFFFFFF80000760
99	23.78478432	<OA>: InHandleException(): rbp 0x20206F49 r14 0xFFFFFA8001EC86F0
100	23.78478813	<OA>: InHandleException(): rsp 0xFFFFFA80027F70E8 r15 0x1C

主要调用代码必须用汇编：

```
PrintReg PROC
```

```
    rdmsr
```

```
    ret
```

```
PrintReg ENDP
```

通过驱动打印的信息可以在 debugView 中看到。为了看到效果，对原代码进行了以下修改后，由于 rdmsr 返回值会写入到 rdx:rax 中，根据定义的函数返回值大小，就可将 rax 的值打印出来。结果如图，修改代码如下：

```
//GuestRegs->rax = MsrValue.LowPart;
```

```
GuestRegs->rax = 0x12345678;
```

```
GuestRegs->rdx = MsrValue.HighPart;
```

在图中一个引人注意的问题就是，在执行完 rdmsr 的 Trap 后，引发了 VM 自定义的 Exception,调用函数为 InHandleException。这个异常是如何引发的，尚待研究。