1、概述

包括两个部分

VMM

Guest software

VMX

VMX是对应的虚拟化操作的名称，有两种，root和non-root，正如想的那样，VMM通常在root，而Guest在non-root，从VMM进入non-root的VMX称为VM entries，而从non-root进入root则称为VM exits。

除了VM陷入的那些指令，其他没有任何指令或者比特位可以标志着他处于一个VMM中，哪怕是CPL0级别的指令。

VM软件的生命周期

VMXON->VMXLAUNCH->VMXRESUME->VMXOFF

在resume之后，就进入了Guest环境下，并通过一些VM exit的指令来VM exit，VMXOFF来离开虚拟机。

VMCS结构体

这是用来管理虚拟机的结构体，后面会讲到

对于该结构体，使用一个寄存器存放该指针，使用VMPTRST和VMPTRLD来读和写。

而使用VMREAD，VMWRITE和VMCLEAR来进行结构体的configure

如何判断是否支持VT-x

使用CPUID命令，然后传入的参数中，EAX设为1，返回的值放在ECX中，其中bit5如果为1，标识了支持VT-x。

Enable VT-x

通过在CR4中的bit13使能它，然后VMXON，如果CR4没有置位，那么会报一个例外。如果已经VMXON了，那么不能够再清除掉CR4中的bit13。

同样的过程，发生在结束VM，先VMXOFF，再CR4中的bit13清零。

除此之外，想要Enable VT-x，软件还需要分配一个4k对齐的页，用于作为VMXON的参数。软件不应当在该内存块的内容中写任何东西，在VMXON和VMXOFF之间，不应当用任何方式去访问该内存块。

一些其他标志位的约束

CR0的PE，NE和PG，以及CR4的13bit都需要置位。表明VT-x指令必须开启硬件分页机制。如果想要在裸板子上面不带内存管理运行，那么VMM必须支持这件事情。

2、VMCS结构体

VMCS结构体和那个64位的4k对齐的地址相联系起来。当使用VMPTRLD之后，那么对应的VMCS结构体就active了，该操作会将对应的那个加载到当前的VMCS指针里面去。

直到使用VMCLEAR把相同的这个地址清理掉之后。才算结束。

如果VMCLEAR清理的地址就是当前VMCS结构体，那么他就失效了。

同一时刻可能存在多个VMCS是active的，而current VMCS指针则是最近的那个VMPTRLD执行的，并且仍然active的那个。

如果不是active的，那么就没有current VMCS。