内核版本：linux-5.8

作者：彭东林

邮箱：[pengdonglin137@163.com](mailto:pengdonglin137@163.com)

内核配置：

CONFIG\_ARM64\_VA\_BITS\_48

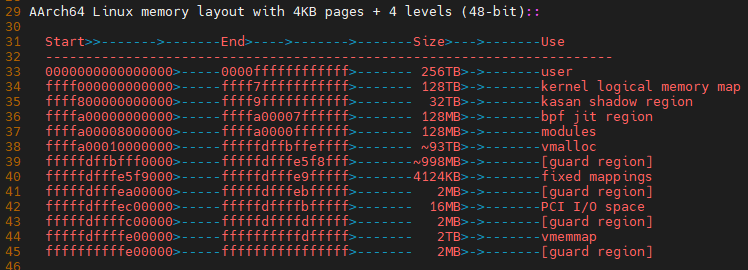
CONFIG\_ARM64\_PA\_BITS=48

CONFIG\_ARM64\_4K\_PAGES

CONFIG\_PGTABLE\_LEVELS=4

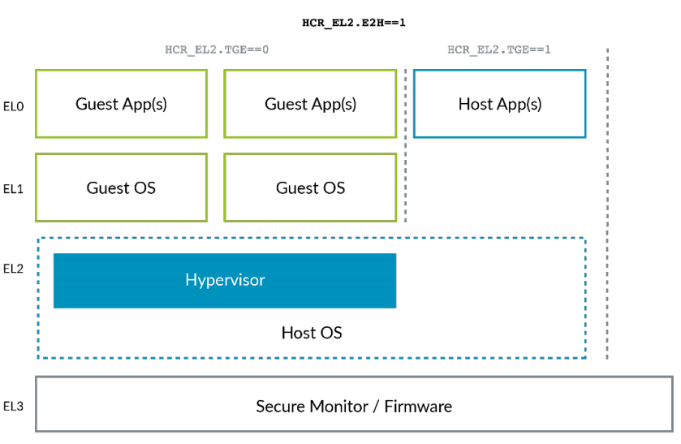
CONFIG\_CPU\_LITTLE\_ENDIAN

内存布局：Documentation/arm64/memory.rst

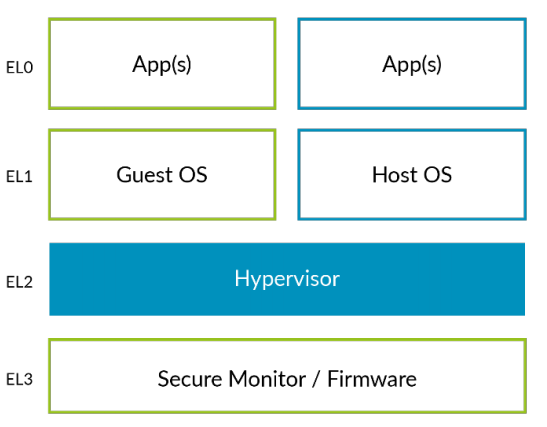


# 初始化

在支持VHE的情况下，HOST的内核和应用程序分别运行在EL2和EL0，而GUEST的内核和应用程序则运行在EL1和EL0.



在不支持VHE的情况下，GUEST的内核和应用分别运行在EL1和EL0，HOST的内核和应用分别运行在EL1和EL0，HOST的内核有一部分运行在EL2.



下面分析的是不支持VHE的情况。

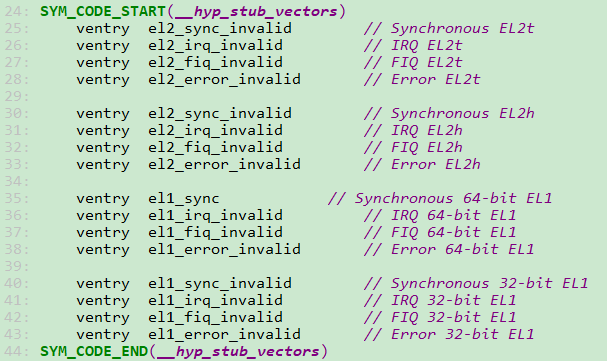
要支持虚拟化，bootloader在跳转到linux kernel时，需要处于EL2.

在arch\arm64\kernel\head.S：

\_head -> primary\_entry -> el2\_setup:

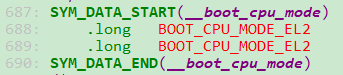
1. 配置在EL2下数据和页表访问的大小端，当前设置为小端（如果没有定义CONFIG\_CPU\_BIG\_ENDIAN的话）
2. 配置HCR\_EL2: EL1运行在Aarch64
3. 配置cnthctl\_el2：允许EL0和EL1访问物理timer和物理counter
4. 将cntvoff\_el2清零，使物理counter和虚拟counter的读数相同，不存在时间差
5. 如果支持以系统寄存器（SRE）的方式访问GICv3和GICv4的cpu interface的话，需要使能EL2\_SRE
6. 设置ID寄存器：将vpidr\_el2和vmpidr\_el2设置为同midr\_el1和mpidr\_el1一样的值
7. 将vttbr\_el2清零，这个寄存器存放的使stage2页表基地址
8. 设置sctrl\_el1，设置EL0和EL1的数据和页表访问时的大小端
9. 设置vbar\_el2为**\_\_hyp\_stub\_vectors**
10. 设置spsr\_el2，然后eret，将cpu切入el1模式

上面提到的\_\_hyp\_stub\_vectors定义在arch\arm64\kernel\hyp-stub.S：

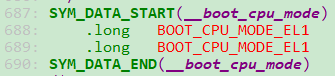


可以看到，只有一个el1\_sync入口，host会调用\_\_hyp\_set\_vectors陷入这里，设置新的vbar\_el2。

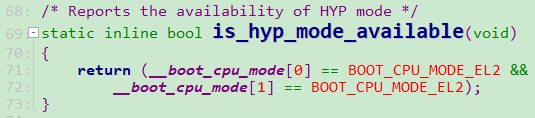
如果进入kernel的时候，cpu处于EL2，那么会通过set\_cpu\_boot\_mode\_flag将\_\_boot\_cpu\_mode设置为：



如果进入kernel时，cpu处于EL1，那么会设置为



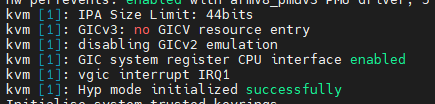
将来会通过函数is\_hyp\_mode\_available来判断EL2是否可用，对于guest返回false，对于host返回true，在子系统初始化时会利用这个函数的返回值走不同的分支。



在guest的启动log里可以看到：



在host的启动log里可以看到：



接下来看kvm模块的初始化。

在arch\arm64\kvm\arm.c中用module\_init定义了arm\_init函数，这个函数会在系统启动的后期被回调：

start\_kernel-> arch\_call\_rest\_init -> rest\_init -> kernel\_init -> kernel\_init\_freeable -> do\_basic\_setup -> do\_initcalls:

static int arm\_init(void)

{

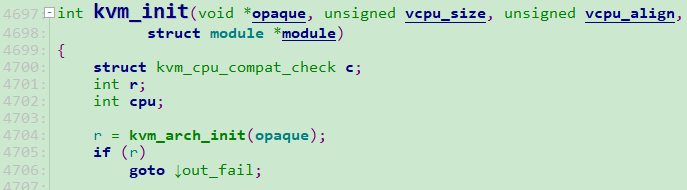
int rc = kvm\_init(NULL, sizeof(struct kvm\_vcpu), 0, THIS\_MODULE);

return rc;

}

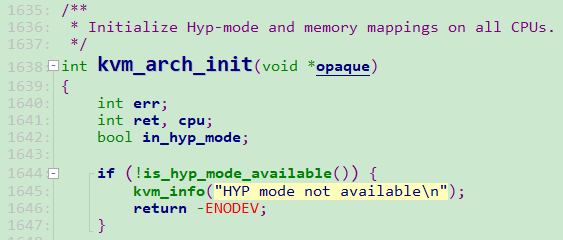
kvm\_init定义在virt\kvm\kvm\_main.c中：

arm\_init -> kvm\_init:

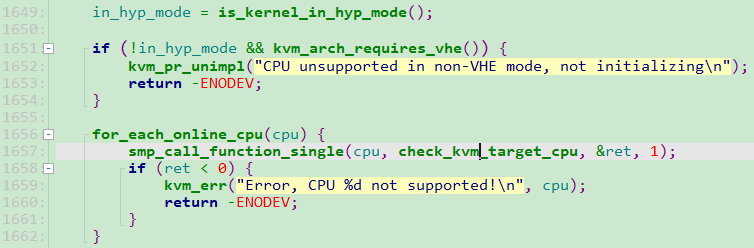


第4704行，调用kvm\_arch\_init，这个函数定义在arch\arm64\kvm\arm.c中：

arm\_init -> kvm\_init -> kvm\_arch\_init:



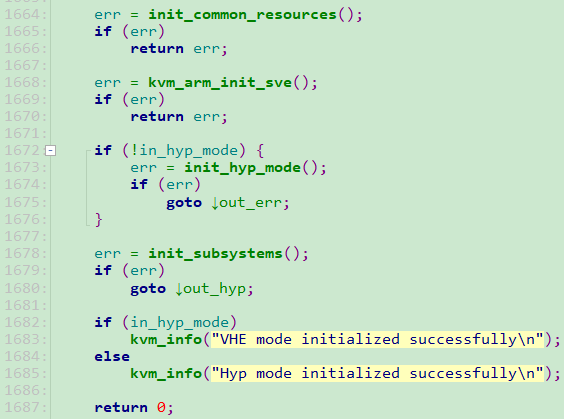
第1644，判断hyp是否可用，对于host来说返回true，对于guest返回false。



第1649行，判断当前host是否运行在EL2，对于NVHE来说，运行在EL1，所以返回false。

第1651行，如果不支持VHE，函数kvm\_arch\_requires\_vhe返回false

第1657行，在每host的每个cpu上运行check\_kvm\_target\_cpu，获取cpu的part id。



第1664行，根据cpu的特性，设置kvm\_ipa\_limit的值，目前这个值是44

第1668行，暂不考虑SVE的情况

第1673行，由于当前不是EL2，所以会调用init\_hyp\_mode。

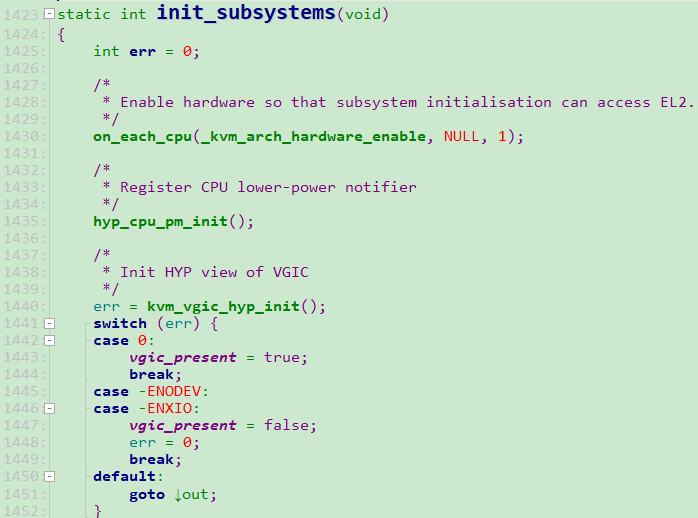


第1678行，调用init\_subsystems初始化vgic和timer

下面是struct kvm\_host\_data数据结构的内容，每个cpu一个，用于保存host的cpu寄存器的值。



arm\_init -> kvm\_init -> kvm\_arch\_init -> init\_subsystems:



第1430行，在每个cpu上执行函数\_kvm\_arch\_hardware\_enable