【ARM】Thumb2指令集中函数的地址不对齐?

简介ARM指令集、Thumb指令集和Thumb2指令集：

ARM指令集：

编代码全部是 32bits 的，每条指令能承载更多的信息，因此使用最少的指令完成功能， 所以在相同频率下运行速度也是最快的， 但也因为每条指令是32bits 的而占用了最多的程序空间。

Thumb指令集：

编代码全部是 16bits 的，每条指令所能承载的信息少，因此它需要使用更多的指令才能完成功能， 因此运行速度慢， 但它也占用了最少的程序空间

Thumb-2指令集：
在前面两者之间取了一个平衡， 兼有二者的优势， 当一个 操作可以使用一条 32bits指令完成时就使用 32bits 的指令， 加快运行速度， 而当一次操作只需要一条16bits 指令完成时就使用16bits 的指令，节约存储空间。

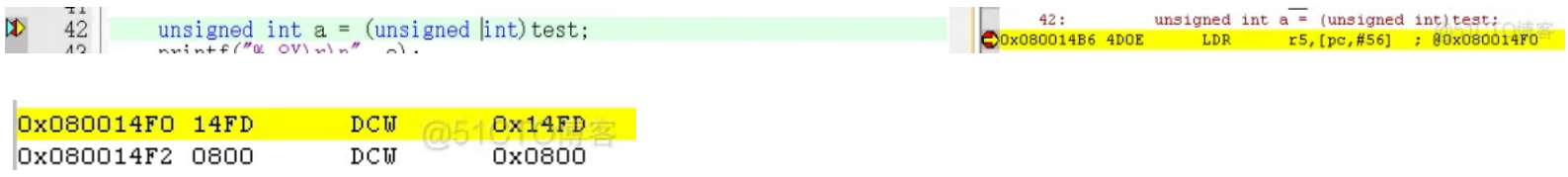
Cortex-M核的单片机使用的是Thumb-2指令集。STM32F1单片机上的实验程序如下：

```
1.  void test(void (*p)(USART_TypeDef*,u8))
2.  {
3.      ...
4.  }
5.
6.  int main(void)
7.  {
8.      Initialization();
9.
10.     printf("%.8X\r\n", (unsigned int)main);
11.
12.     unsigned int a = (unsigned int)test;
13.     printf("%.8X\r\n", a);
14.
15.     while(1);
16. }
```

ARM指令是4字节对齐的，Thumb-2指令是2字节对齐的，所以这里函数test的地址应该是2字节对齐的，但是打印出来的值却是：080014A9和080014FD。main函数的地址和test函数的地址都成了奇数地址了。

这个问题在于有些ARM处理器即能使用ARM指令，又能兼容Thumb指令，同一个应用程序中可能同时存在ARM指令和Thumb指令，这两者的处理方式肯定是大不相同的，所以为了切换ARM状态和Thumb状态，在跳转到Thumb指令编写的代码块的时候，将程序地址的最低位置1（因为不管是ARM指令还是Thumb指令，都至少是2字节对齐的，所以最低位一定是0，所以最低位可以拿来用于区分ARM状态和Thumb状态），这样处理器识别到最低位为1的话就会切换到Thumb状态，否则则是ARM状态。Thumb2指令集也是为了兼容以前的ARM状态和Thumb状态这样做的。

所以编译器编译STM32F1的程序的时候，会把函数的真实地址 加上1 作为常量放在ROM空间（如果这个函数的地址有被用到的话），获取函数的指针的时候就会获取到最低位被置1的一个地址。如下图，获取test的地址的时候，到0x080014F0地址处读取到了0x080014FD的值，这其实就是test的真实地址0x080014FC + 1得到的。



另外做了一个实验，强行跳转到 0x080014FC 地址处执行函数，发现单步运行的时候没有问题，但是正常运行的话就会报HardFault。