PA1-1 report

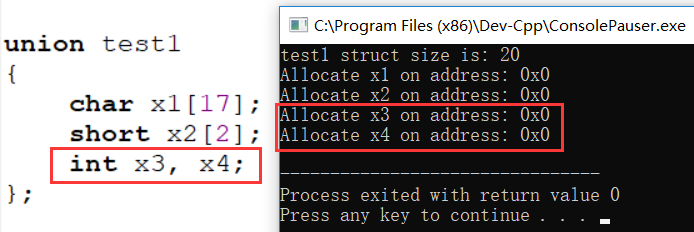
17300180070

在PA1第一阶段中，我完成了5项内容：阅读框架代码、完成寄存器结构体、添加单步执行功能、添加打印寄存器功能、添加扫描内存功能。

阅读框架代码没什么难度，虽然一开始感觉无从下手，但是PA手册中提示：一个程序从哪里开始执行呢？这样一来，就可以从main.c入手，再逐个阅读相关函数的定义。值得思考的是PA手册思考题提到的opcode\_table数组，我一开始没有反应过来它的类型，搜索后发现这是一个函数指针类型的数组，巩固了我的程序设计基础知识。

完成寄存器结构体有一定的难度，我断断续续在两天内完成。题意还是非常清晰的；首先很容易想到要把struct {\_32, \_16, \_8[2]}改成union；但这之后就需要灵机一动才能想出来了。核心问题就是：怎样确保eax, ecx, edx…和上面的union数组从同一地址开始一一对应呢？

我首先尝试了在union数组和int eax, ecx…外面再套一层union，但是发现，即使是同一语句内逗号声明的不同变量，仍然是开始于同一起始地址的，而并不是顺序排布。此路不通。

然后我尝试了把八个int变量改成int&（int引用）类型，然而非常遗憾的是，C语言里并没有引用类型，这是C++独有的。此路不通。

之后我又尝试在结构体内声明同名函数，然而甚至无法完成编译：C语言是不能在结构体内声明函数的，这又是C++独有的；如果想要声明函数，只能通过函数指针完成。那天剩下的时间里，我没有再得出任何新的想法。

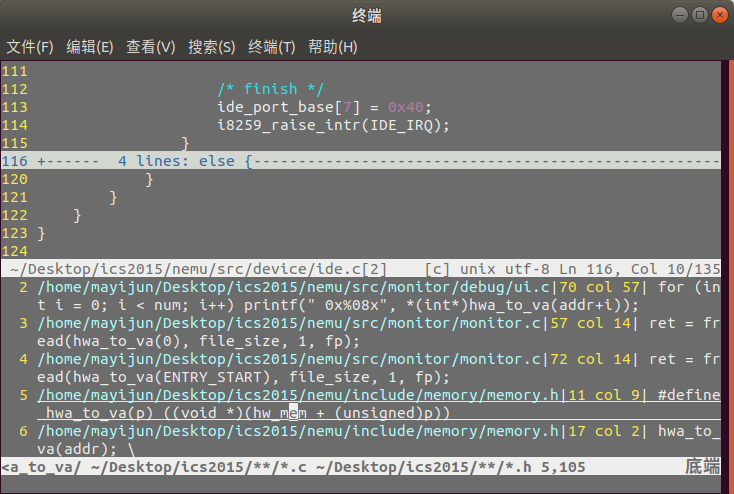
第二天，我开始思考，之前尝试的第一种方法——再嵌套一层union应该是正确的方向，唯有这样我们才能确保eax这列数从正确的地址开始。那么如果eax这列数用一个数组表示的话，问题将直接得到解决。此时我已经非常接近正确答案了——使得一些无关的对象拥有像数组一样的整体性的方法就是struct。但非常遗憾的是我接下来又跑偏了，思考了半天如何将一列int变量和一个int数组建立一一对应关系，自然是无计可施。最后我不得不求助做出来的同学；当我知道答案的时候，用一句歌词来说，“最后知道真相的我眼泪掉下来”。

单步执行功能和打印寄存器功能毫无思维难度，只要看得懂框架代码就非常容易写出来。然而在写单步执行的时候有一个小插曲：如果键入si 10或者更大的数，就不会打印执行结果。在群里提问之后，很快有同学提醒我，在cpu\_exec.c中定义了一个常数MAX\_INSTR\_TO\_PRINT，如果一次调用函数的指令条数达到这个数，将不会被打印。（在此也向维扬（网名）同学和张政屹同学致谢）他们还说，只要通过for循环执行同样次数的cpu\_exec(1)就可以打印出结果。但我选择尊重原作者的设计，因此仍然保留了调用cpu\_exec(n)，并在n>=10时输出提示信息。

值得一提的是，cpu\_exec.c中定义的宏在ui.c中无法使用，这样只有在ui.c中也定义一个一模一样的宏了，会带来数据冗余的问题；但为了尊重原作者的设计，我没有动手改原来的程序，因为把一个不需要的宏引进无关的源程序中更危险。

扫描内存功能比前两项功能相对复杂，需要通过阅读的框架代码自行找出访问内存的方式（hwa\_to\_va()函数），且这一命令有两个参数，需要自行解析。

因为PA手册中并没有提到hwa\_to\_va()的定义在什么地方，但是我们又必须阅读这个函数的定义，来使它更好地为我们所用，所以我一开始在Windows中建立了一份代码拷贝，并将其建设成Visual Studio项目，这样就能通过VS的右侧列表快速找到函数定义。然而因为网络不通畅，这项计划没有付诸实践。

但我后来又想，vim号称是最强大的编辑器，应该不会没有全局查找的功能吧？于是我在网上自行学习了vim的全局查找，终于揪出了这个藏得很深的函数。另外，得益于PA手册的教导，我决定把我的虚拟机桌面也设置成vim键位图，方便随时查找快捷键。

除了通过hwa\_to\_va()访问内存地址，还需要在参数解析中加入特判。利用sscanf()函数的返回值判断成功读取的参数个数；若args的格式不符合预期需要输出错误信息“未知参数'%s'”，若args为NULL需要提示用户输入参数。

最后，还有另一个插曲，是我不熟悉git的使用导致的：PA手册说，如果git log中有自己手动提交的、合理的git记录，将可以获得加分。然而，在我做完一项子任务后准备提交时，键入git commit命令，却返回“无文件要提交”。后来我重新翻了一遍PA手册里的git教程，得知，正确的处理方式应该是：git commit --allow-empty。至此问题得到解决。