PA2.5

1. 理解mainargs参数是如何传递给hello程序的:
   1. 首先要明确，这个hello程序是要被nemu加载的程序，因此在hello被编译的时候，就需要把这个参数传递给hello里面的main函数了。因此我们可以定位这个参数是在编译的时候。但是目前只能找到makefile中关于把这个参数作为预定义的宏传给了hello.c函数。那么hello是如何根据这个宏来得到参数的呢？？？这个地方不理解！
   2. 补充a) 找到了这个宏在哪里被引用了！！！在abstract-machine/am/src/platform/nemu/trm.c这个函数下面！这个trm.c里面的内容还是非常重要的。他直接定义了halt、heap、以及trm的启动函数\_trm\_init。在\_trm\_init函数中，调用了宏mainargs并传递给了main函数！

另外，在start.S汇编文件中，定义了程序的入口\_start，并且从\_start启动进入\_trm\_init。

1. 在始终update这一块的代码似乎是有BUG的！只有把高低位都拼接起来才能正确让us读出时间，但是只读地位却不行，us一直是0.这就非常奇怪了，因为地位肯定有数的呀！！所以可能是我的低32位数据的指令有BUG？？
   1. 进一步探索发现，只要把RTC\_ADDR的高位和地位都读出来就没有这个BUG。。。。。莫不着头脑～～。
   2. !!!原来这就是框架代码埋下的大坑！！！！他是通过寄存器来模拟MMIO的，所以他需要通过回调函数来更新，而回调函数只有在你读时钟外设高地址的时候才会更新。所以读取顺序要从高到低。。。。。。太难了！！！

进入到ysyx基础阶段 B线

1. 又遇到了ALU的问题。重新思考了以下关于无符号数比较的问题。觉得可以类比一个12小时制的时钟。利用减法来比较时，对一个点取反相当于他在时钟上对称了一下，即时钟针到0/12点的距离在之前和之后都是一样的。这就像一个头尾连在一起的一个结构。因此无符号数利用减法比较时，当结果的carry有进位的时候，那么就说明他们相加溢出了！溢出就说明被减的那个时钟针在减数的后面或者相等！！非常巧妙好理解，尝试在纸上画一个12小时制的时钟就明白了。
2. 对于补码运算的符号溢出问题。 加法减法都转换为加法运算，而且数据以补码的形式进行运算，得出的结果就是补码形式，不论以有符号数还是无符号数的角度看，计算结果都是正确的！ 对于加法来说，只有正数+正数，负数+负数这两种情况会发生溢出问题，因此只需要特殊处理这两类。
3. 比较运算这个地方感觉实验讲义上有点麻烦。在不溢出的情况下，直接看符号为判断；在溢出的情况下，说明A，B肯定不同号，这个时候直接判断A、B的符号位不就行了吗？ 但是判断结果的符号位确实更省逻辑资源一些： 那么这样只剩两种情况： 正-负 ->正+正溢出，那么结果符号位肯定是1,这时候A为正， A>B;另一种情形就是负-正 ->负+负溢出，那么结果符号位肯定是0,这时候A<B；

B线

1. PA3.1
2. nanos的代码和NEMU的代码是分离的，互不影响。Nanos运行在AM之上
3. AM给的栈空间有多大呢？ 这一点应该根据连接器的脚本可以查询到。在AM目录下的link.ld文件查询，他在链接的时候通过-T被引用，可以看到栈空间的大小是0x8000，并且是0x1000字节对齐的。