

PA2 平时作业

学号：2011743 姓名：高祎珂

问题一：为何是jbe指令？

If-else语句举例

```
int get_cont( int *p1, int *p2 )
{
    if ( p1 > p2 )
        return *p2;
    else
        return *p1;
}
```

p1和p2对应实参的存储地址分别为R[ebp]+8、R[ebp]+12，EBP指向当前栈帧底部，结果存放在EAX。

为何这里是“jbe”指令？

```
movl 8(%ebp), %eax    //R[eax] ← M[R[ebp]+8], 即 R[eax]=p1
movl 12(%ebp), %edx   //R[edx] ← M[R[ebp]+12], 即 R[edx]=p2
cmpl %edx, %eax       //比较 p1 和 p2, 即根据 p1-p2 结果置标志
jbe .L1               //若 p1 ≤ p2, 则转 L1 处执行
movl (%edx), %eax     //R[eax] ← M[R[edx]], 即 R[eax]=M[p2]
jmp .L2               //无条件跳转到 L2 执行

.L1:
    movl (%eax), %eax  // R[eax] ← M[R[eax]], 即 R[eax]=M[p1]
.L2:
```

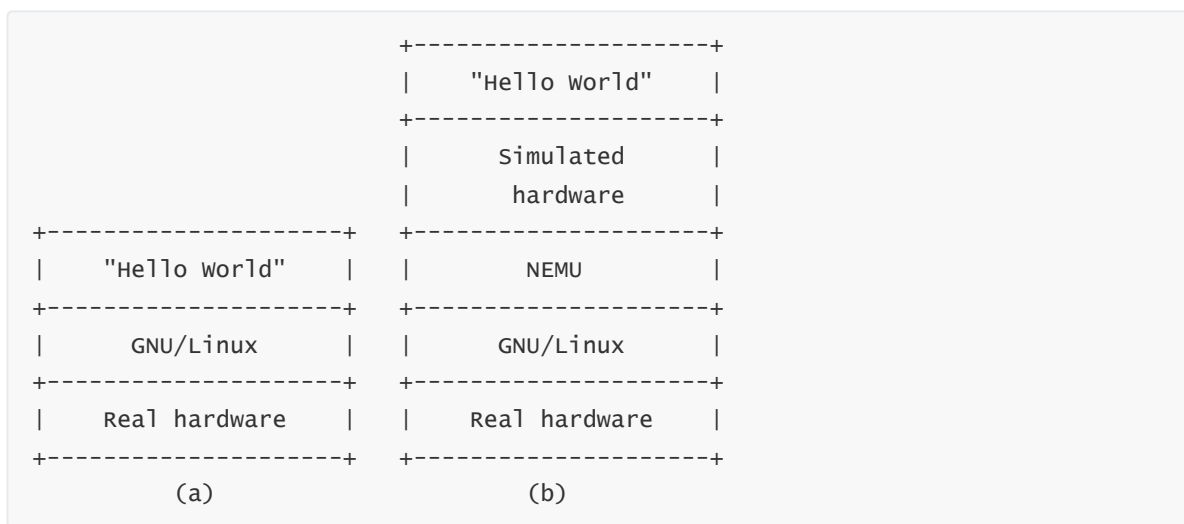
75

jbe的上一条cmpl指令，该指令的格式为：cmpl op1,op2，它比较op1和op2的大小，根据比较结果更新处理器标志寄存器的标志位。如果op1<op2，则SF标志被置1，CF和ZF标志被置0；如果op1>op2，则SF和ZF标志被置0，CF标志被置1；如果op1==op2，则ZF标志被置1，SF和CF标志被置0。在上述代码中，由于edx存储p2的值，eax存储p1的值，故：

- 如果p1>p2，则op1<op2，SF=1，CF=0，ZF=0
- 如果p1<p2，则op1>op2，SF=0，CF=1，ZF=0
- 如果p1==p2，则src==dest，SF=0，CF=0，ZF=1

因此，对于p1<=p2的情况，应该进入else分支(即L1语句)。此时CF或ZF==1，而jbe指令表示，当CF或ZF为1时跳转。因此，使用jbe指令后，程序逻辑为：若p1<=p2则执行else语句，否则执行if语句，与高级语言程序一致。

问题二：nemu输出的helloworld和程序中输出的helloworld有什么区别？



上图(a)展示了"在GNU/Linux中运行Hello World"的情况. GNU/Linux操作系统直接运行在真实的计算机硬件上, 对计算机底层硬件进行了抽象, 同时向上层的用户程序提供接口和服务. Hello World程序输出信息的时候, 需要用到操作系统提供的接口, 因此Hello World程序并不是直接运行在真实的计算机硬件上, 而是运行在操作系统(在这里是GNU/Linux)上.

上图(b)展示了"在GNU/Linux中通过NEMU执行Hello World"的情况. 在GNU/Linux看来, 运行在其上的NEMU和上面提到的Hello World程序一样, 都只不过是一个用户程序而已. 但NEMU的功能是负责模拟出一套计算机硬件, 让程序可以在其上运行.

综上所述, nemu输出的helloworld和程序中输出的helloworld的区别在于: nemu输出的helloworld是在程序输出helloworld的真是硬件基础上, 实现了一个软件nemu, 这个软件可以模拟硬件的各种行为, 在这一套模拟出的硬件上输出的helloworld. nemu 输出 hello world 的程序直接运行在裸机上, 在AM 的抽象下直接输出到设备(串口); 而编写的输出 hello world 的程序位于操作系统之上, 不能直接操作设备, 只能通过操作系统提供的服务进行输出, 输出的数据要经过很多层抽象才能到达设备层.