

3月28日

9. 1)  $\pm 1\text{MB}$

2)  $\pm 4\text{KB}$

3) 可以, 一条 `LUI` 指令将目标地址的高20位加载至 `rs1`, `JALR` 可以加上低12位, 然后便可访问32位空间

10. 条件: ① 立即数或者偏移量补码

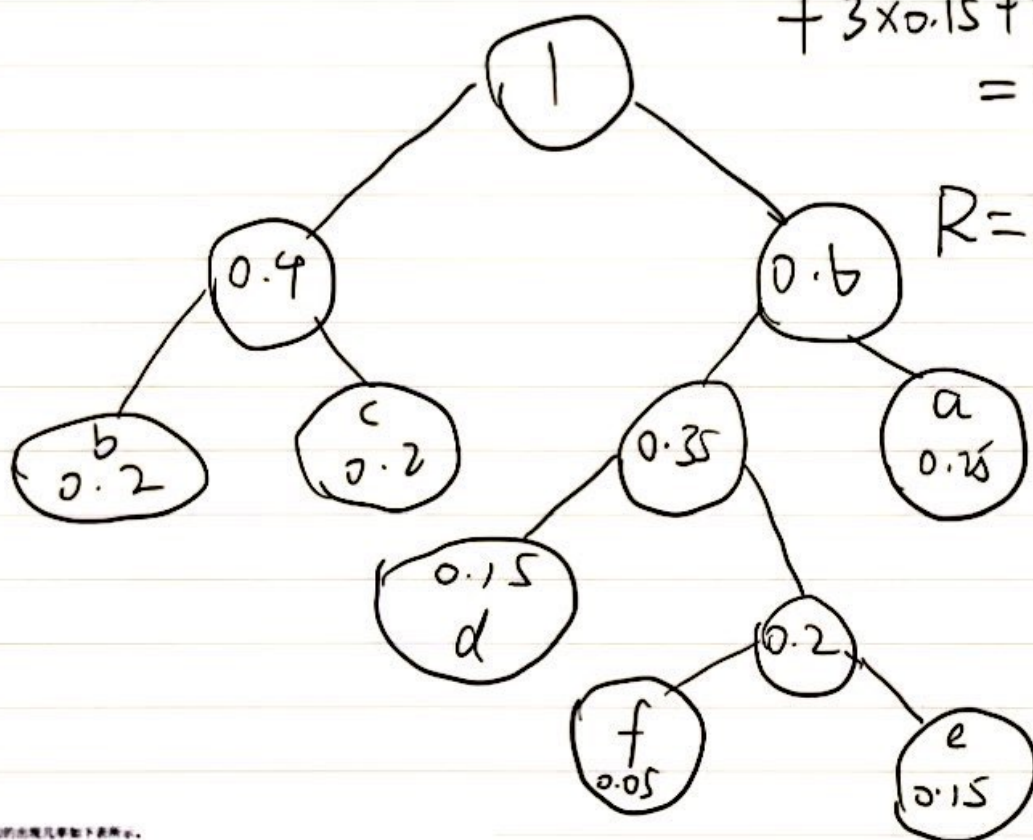
② 其中一个寄存器是 `x0`, `x1` 或 `x2`

③ 目标寄存器和第一个源寄存器相同

④ 最常见情况下使用了8个寄存器

但是, 比如 `I` 指令格式仅有 20 位立即数, 不足以使用完整的 32 位寄存器

18.  $H = -0.25 \times \log_2 0.25 - 0.2 \times \log_2 0.2 - 0.2 \log_2 0.2 - 0.15 \log_2 0.15 - 0.05 \log_2 0.05 - 0.15 \log_2 0.15 = 2.466$ ,  $L = 2 \times 0.25 + 2 \times 0.2 + 2 \times 0.2 + 3 \times 0.15 + 4 \times 0.05 + 4 \times 0.15 = 2.55$



$R = 1 - \frac{H}{L} = 0.033$

18. 有一组操作码，它们的出现几率如下表所示。

a	0.25
b	0.20

c	0.20
d	0.15
e	0.15
f	0.05

请按前缀大数编码对这些操作码进行编码，计算操作码的平均长度和信息冗余度。

19. (1) 原理: 在计算机中, 函数调用是通过栈数据结构实现的。每当程序进入一个程序调用, 栈就会加一层栈帧。由于栈的大小不是无限的, 所以递归调用的次数过多, 导致栈溢出。函数参数是通过栈传递的, 在调用中会占用线程的栈资源。如果递归调用层数过多, 就会导致占用的栈资源超过线程中最大值, 从而导致栈溢出。



19. (2) ① 用栈把递归换成非递归  
 ② 使用 static 对象替代 nonstatic 局部对象  
 ③ 增大堆栈大小值  
 ④ 尾调用优化

20.

ra (F <sub>1</sub> ) ← sp 返回地址 )
F <sub>1</sub> 输入参数 (F <sub>1</sub> )
to (F <sub>1</sub> )
so (F <sub>1</sub> )
ra (F <sub>2</sub> )
F <sub>2</sub> 入参 <sub>1</sub> (F <sub>2</sub> )
F <sub>2</sub> 入参 <sub>2</sub> (F <sub>2</sub> )
to (F <sub>2</sub> )
t <sub>1</sub> (F <sub>2</sub> )
so (F <sub>2</sub> )
s <sub>1</sub> (F <sub>2</sub> )