

9. 解: (1) jal指令格式为J类型, 20位立即数进行符号拓展, 并左移1位, 形成一个32位符号数。该数与PC相加产生目标地址。

因此相较于当前PC, 可跳转的地址空间范围为 $\pm 1\text{MiB}$ 。

(2) 条件分支指令含有12位有符号立即数编码, 相较于当前PC可以跳转的地址空间范围为 $\pm 4\text{KiB}$

(3) 可以。lui指令可以先将高20位地址加载到rs1中, 然后jalr补足低位的地址。

10. 解: (1) 条件: ① 立即数或地址偏移量较小;

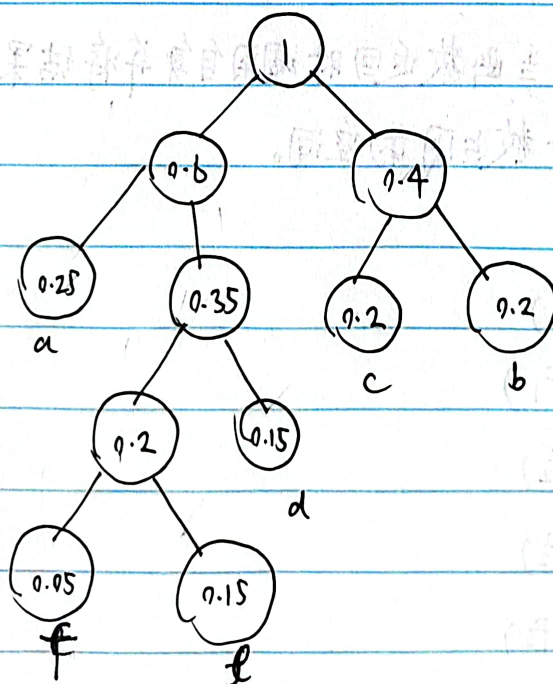
② 其中一个寄存器是零寄存器(x_0)或ABI链接寄存器(x_1)或ABI栈寄存器(x_2);

③ 目标寄存器和第一个源寄存器相同;

④ 使用的寄存器是8个最常用的寄存器。

(2) CR、CL和CSS可以使用32个通用整型寄存器, 但是CLW、CL、CS和CB只能使用8个通用整型寄存器, 即 $x_8 \sim x_{15}$

18. 解:



可得

a_i	p_i	l_i	$c(a_i)$
a	0.25	2	00
b	0.20	2	11
c	0.20	2	10
d	0.15	3	011
e	0.15	4	0101
f	0.05	4	0100

操作码平均长度 $\sum_{i=1}^6 p_i l_i = 2.55$

$$\begin{aligned} \text{信息冗余量 } R &= 1 - \frac{-\sum_{i=1}^n p_i \cdot \log_2 p_i}{\lceil \log_2 n \rceil} \\ &= 1 - \frac{-\sum_{i=1}^6 p_i \log_2 p_i}{\lceil \log_2 6 \rceil} \\ &= 0.178 \end{aligned}$$

19. 解: ① 函数调用时通过堆栈来保存返回地址、局部变量和参数等信息。当函数嵌套层数过多时, 堆栈长度超过系统能提供的内存范围, 就会造成栈溢出;

② ①使用循环替代递归;

② 使用尾递归优化, 当函数返回时调用自身并将结果返回上一级调用者;

③ 减少局部变量、参数占用的空间。

20. 解:

$t_0(F_1)$

$s_0(F_2)$

$t_0(F_2)$

$t_1(F_2)$

$r_a(F_2)$