

9.

- (1) JAL 指令使用U类格式，立即数J编码一个2的倍数的有符号偏移量，偏移量符号扩展后加到PC上，跳转的范围达到 $\pm 1MB$
- (2) bne 指令使用B类格式，12位立即数编码了一个以2字节倍数的有符号偏移量，并被加到当前PC上，条件分支范围是 $\pm 4KB$
- (3) 可以

LU1指令将目标地址的高20位加载到寄存器中，然后JALR指令可以加上低12位，所以可以跳转到32位绝对地址空间的任何地方。

10.

32位指令被压缩为16位RVC指令的条件：

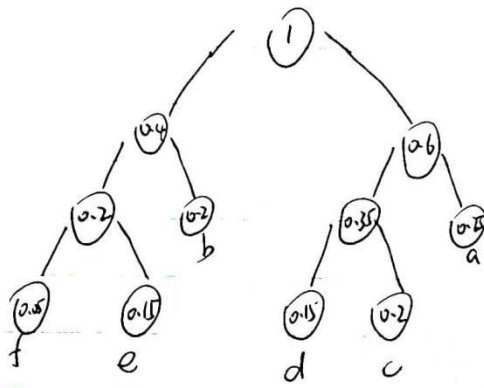
- ① 立即数或地址偏移量较小
- ② 其中一个寄存器是零寄存器(X0)、ABI链接寄存器(X1)或ABI栈寄存器(X2)
- ③ 目标寄存器和第一个源寄存器相同

并不是所有指令都可以使用所有32个RVI寄存器

CR、CI、CS 格式的压缩指令可以使用任何32个RVI寄存器

而CLW、CL、CS、CB 格式的指令被限制为只能使用32个寄存器中的8个寄存器。

18.



P_i 编码:

P_i	字母	二进制编码	l_i
0.25	a	11	2
0.2	b	01	2
0.2	c	101	3
0.15	d	100	3
0.15	e	001	3
0.05	f	000	3

$$L^* = \sum_{i=1}^n P_i l_i = 2.55$$

$$H = - \sum_{i=1}^n P_i \log_2(P_i) = 2.47$$

$$\therefore R = 1 - \frac{H}{L^*} = 0.03 = 3\%$$

∴ 操作码平均长度为 2.55
信息冗余度为 3%.

19.

(1) 在计算机中, 函数调用是通过栈这种数据结构实现的, 每当程序执行进入一个函数调用, 栈就会加一层栈帧, 每当函数返回, 栈就会减一层栈帧。由于栈的大小不是无限的, 所以, 递归调用的次数过多就会导致栈溢出。函数参数是通过栈来传递的, 在调用中会占用线程的栈资源。递归调用在到达最后的结束点后函数才能依次退栈清栈, 如果递归调用层数过多, 就可能导致占用的栈资源超过线程的最大值, 从而导致栈溢出, 程序异常退出。

(2) 尾递归优化可以避免栈溢出

尾递归优化, 无论调用多少次, 都只占用一丁点栈, 不会出现溢出

20. $\gamma_a(\bar{F}_1)$

$\gamma_a(\bar{F}_2)$

$t_0(\bar{F}_1)$

$s_0(\bar{F}_2)$

$\gamma_a(\bar{F}_3)$

$t_0(\bar{F}_2)$

$t_1(\bar{F}_2)$

$s_0(\bar{F}_3)$

$s_1(\bar{F}_3)$