LabH3 report

PB21000039 陈骆鑫

实验目的与内容

- 1. 提供对要求的18条指令的逐指令测试程序;
- 2. 使用提供指令完成排序程序,以进行进一步测试。

逻辑设计

- 1. 逐指令测试:
 - 。 PPT中已经给出了测试逻辑,按其完成即可。
 - 。 结果返回方式如下: 依最后ra寄存器的值而定, 1代表成功, 0代表失败。
- 2. 排序程序:
 - 。 由于已经上升到了软件层面,使用更高效的排序算法。这里选择堆排序。
 - 。 算法参考严蔚敏《数据结构》一书。C语言代码如下:

```
1 int *a;
 void heap_adjust(int s, int n){
       int j = 2 * s, t = a[s];
 4
       while (j \le n) {
 5
           if (j < n){
              if (a[j + 1] > a[j]) {
 7
                  j++;
               }
 9
           }
          if (t > a[j]) break;
10
11
          a[s] = a[j];
          s = j;
12
           j *= 2;
       }
14
       a[s] = t;
15
16
       return;
17 }
19 int main(){
      *a = \{7, 5, 3, 1, 4, 7, 2, 6\};
20
21
       int n = a[0];
22
       i = n / 2;
      while (i > 0){
24
          heap_adjust(i, n);
           i--;
25
26
       }
27
       i = n;
```

```
28  while (i > 1){
29     swap(a[1], a[i]);
30     i -= 1;
31     heap_adjust(1, i);
32   }
33 }
```

将其转写为汇编语句即可。注意数据长度为32,长为4个字节;注意可以使用的指令有限,需要改变部分位置的边界判断方式。

测试结果与分析

• 测试程序在RARS环境下,最终返回值为1:

	Name	Number	Value		
zero		0	0x0000000		
ra		1	0x0000000		
sp		2	0x00003ff		
gp		3	0x0000180		
tp		4	0x0000000		
t0		5	0x0000000		
t1		6	0x0000019		
t2		7	0x0000019		
s0		8	0x0000000		
s1		9	0x0000000		
a0		10	0x0000000		
a1		11	0x0000000		
a2		12	0x0000000		
a3		13	0x0000000		
a4		14	0x0000000		
a5		15	0x0000000		
a6		16	0x0000000		
a 7		17	0x0000000		
s2		18	0x00000x0		
s3		19	0x00000x0		
s4		20	0x00000x0		
s5		21	0x0000000		
s6		22	0x00000x0		
s7		23	0x0000000		
s8		24	0x0000000		
s9		25	0x0000000		
s10		26	0x0000000		
s11		27	0x0000000		
t3		28	0x0000019		
t4		29	0x44417a9		
t5		30	0x186a000		
t6		31	0x186a003		
рс			0x000001b		

• 排序程序将预置的数据成功排序。

п									
	Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
	0x00002000	0x00000040	0x00000001	0x00000002	0x00000003	0x00000004	0x00000005	0x00000006	0x00000007
	0x00002020	80000000x0	0x00000009	0x0000000a	d0000000x0	0x0000000c	b0000000x0	0x0000000e	0x0000000f
	0x00002040	0x00000010	0x00000011	0x00000012	0x00000013	0x00000014	0x00000015	0x00000016	0x00000017
· II	0x00002060	0x00000018	0x00000019	0x0000001a	0x0000001b	0x0000001c	0x0000001d	0x0000001e	0x0000001f
	0x00002080	0x00000020	0x00000021	0x00000022	0x00000023	0x00000024	0x00000025	0x00000026	0x00000027
	0x000020a0	0x00000028	0x00000029	0x0000002a	0x0000002b	0x0000002c	0x0000002d	0x0000002e	0x0000002f
	0x000020c0	0x0000030	0x00000031	0x00000032	0x00000033	0x00000034	0x00000035	0x00000036	0x00000037
	0x000020e0	0x00000038	0x00000039	0x0000003a	0x0000003b	0x0000003c	0x0000003d	0x0000003e	0x0000003f
	0x00002100	0x00000040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000