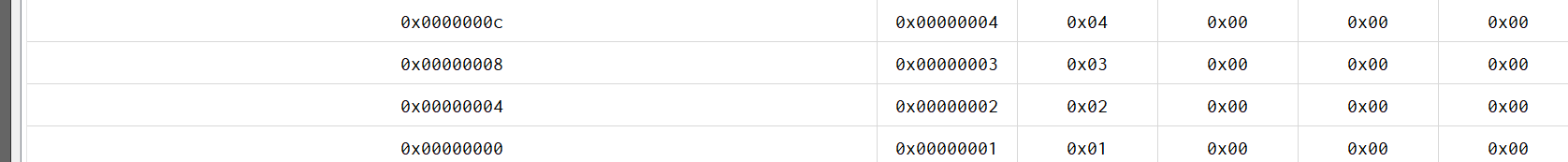
521030910009 王奕权

表1 测试用代码段功能描述表

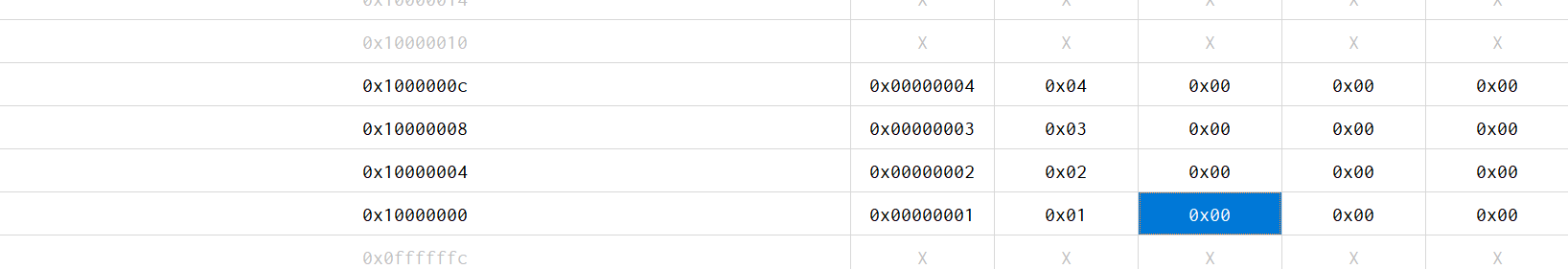
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Address(HEX) | 标号 | 仿真平台2输入代码（这样输入才不会出错） | 仿真平台2显示代码 | 机器码（BIN） | 机器码（HEX） | 指令类型 | 描述 |
| 0 |  | lui x10, 0 | lui a0, 0 | 00000000000000000000010100110111 | 00000537 | U | #initialize x10 =base address 0 |
| 4 |  | ori x4, x10, 0 | oritp, a0, 0 | 00000000000001010110001000010011 | 00056213 | I | #x4<- base address x10 + offset 0 =0 |
| 8 |  | addi x25, x0, 1 | addi s9, x0, 1 | 00000000000100000000110010010011 | 00100c93 | I | #initialize x25 = 1 |
| c |  | addi x26, x0, 2 | addi s10, x0, 2 | 00000000001000000000110100010011 | 00200d13 | I | #X26=2 |
| 10 |  | addi x27, x0, 3 | addi s11, x0, 3 | 00000000001100000000110110010011 | 00300d93 | I | #X27=3 |
| 14 |  | addi x28, x0, 4 | addi t3, x0, 4 | 00000000010000000000111000010011 | 00400e13 | I | #X28=4 |
| 18 |  | sw x25, 0(x4) | sw s9, 0(tp) | 00000001100100100010000000100011 | 01922023 | S | #[0] = 1 |
| 1c |  | sw x26, 4(x4) | sw s10, 4(tp) | 00000001101000100010001000100011 | 01a22223 | S | # [4] = 2 |
| 20 |  | sw x27, 8(x4) | sw s11, 8(tp) | 00000001101100100010010000100011 | 01b22423 | S | # [8] = 3 |
| 24 |  | sw x28, 12(x4) | sw t3, 12(tp) | 00000001110000100010011000100011 | 01c22623 | S | #[12]=4 |
| 28 |  | addi x5, x0, 4 | addi t0, x0, 4 | 00000000010000000000001010010011 | 00400293 | I | # x5 = 4，循环次数为4 |
| 2c | Call： | Call: jal sum | jalra, 128 | 00000101010000000000000011101111 | 054000ef | UJ | # call function sum  跳转到 pc = 80 |
| 30 |  | sw x12, 0(x4) | sw a2, 0(tp) | 00000000110000100010000000100011 | 00c22023 | S | #[16] <- 0x0000000a  (x12=0x0000000a) |
| 34 |  | lw x19, 0(x4) | lw s3, 0(tp) | 00000000000000100010100110000011 | 00022983 | I | #x19<- [16] （0x10）  ([16]=0x0000000a) |
| 38 |  | sub x18, x19, x12 | sub s2, s3, a2 | 01000000110010011000100100110011 | 40c98933 | RR | #x18= 0 |
| 3c |  | addi x5, x0, 3 | addi t0, x0, 3 | 00000000001100000000001010010011 | 00300293 | I | #x5=3 |
| 40 | loop2: | loop2:addi x5, x5, -1 | addi t0, t0, -1 | 11111111111100101000001010010011 | fff28293 | I | #x5=2 |
| 44 |  | ori x18, x5, -1 | ori s2, t0, -1 | 11111111111100101110100100010011 | fff2e913 | I | #x18= 0xffffffff ，（x18 = x5 or 12bit 立即数有符号扩展0xffffffff） |
| 48 |  | xori x18, x18, 1365 | xori s2, s2, 1365 | 01010101010110010100100100010011 | 55594913 | I | #X18=0xfffffaaa |
| 4c |  | addi x19, x0, -1 | addi s3, x0, -1 | 11111111111100000000100110010011 | fff00993 | I | #X19=0xffffffff |
| 50 |  | andi x20, x19, -1 | andi s4, s3, -1 | 11111111111110011111101000010011 | fff9fa13 | I | #X20=0xffffffff，（X20=0xffffffff and 0xffffffff） |
| 54 |  | or x16, x20, x19 | or a6, s4, s3 | 00000001001110100110100000110011 | 013a6833 | R | #X16=0xffffffff |
| 58 |  | xor x18, x20, x19 | xor s2, s4, s3 | 00000001001110100100100100110011 | 013a4933 | R | #X18=0x00000000 |
| 5c |  | and x17, x20, x16 | and a7, s4, a6 | 00000001000010100111100010110011 | 010a78b3 | R | #X17=0xffffffff |
| 60 |  | beq x5, x0, shift | beq t0, x0, 104 | 00000000000000101000010001100011 | 00028463 | SB | #Ifx5 = 0  Goto shift after finished loop2 4 times，goto pc= 68 |
| 64 |  | j loop2 | jal x0, 64 | 11111101110111111111000001101111 | fddff06f | UJ | #Loop Loop2 for 4 times，goto pc=40 |
| 68 | shift: | shift:addi x5, x0, -1 | addi t0, x0, -1 | 11111111111100000000001010010011 | fff00293 | I | #X5=0xffffffff |
| 6c |  | slli x18, x5, 15 | slli s2, t0, 15 | 00000000111100101001100100010011 | 00f29913 | I | #X18=0xffff8000 |
| 70 |  | slli x18, x18, 16 | slli s2, s2, 16 | 00000001000010010001100100010011 | 01091913 | I | #X18=0x80000000 |
| 74 |  | srai x18, x18, 16 | srai s2, s2, 16 | 01000001000010010101100100010011 | 41095913 | I | #X18=0xffff8000 |
| 78 |  | srli x18, x18, 15 | srli s2, s2, 15 | 00000000111110010101100100010011 | 00f95913 | I | #X18=0x0001ffff |
| 7c | finish: | finish:j finish | jal x0, 124 | 00000000000000000000000001101111 | 0000006f | UJ | #Endhere |
| 80 | sum: | sum:add x18, x0, x0 | add s2, x0, x0 | 00000000000000000000100100110011 | 00000933 | R | #X18 = 0x0001ffff |
| 84 | loop: | loop:lw x19, 0(x4) | lw s3, 0(tp) | 00000000000000100010100110000011 | 00022983 | I | #X19 <- [x4] |
| 88 |  | addi x4, x4,4 | additp, tp,4 | 00000000010000100000001000010011 | 00420213 | I | #x4 <-x4+4 |
| 8c |  | add x18, x18, x19 | add s2, s2, s3 | 00000000010000100000001000010011 | 01390933 | R | #X18= x18 + [x4],  x18=x18+x19 |
| 90 |  | addi x5, x5, -1 | addi t0, t0, -1 | 11111111111100101000001010010011 | fff28293 | I | #x5 <- (x5-1)，循环次数-1 |
| 94 |  | bne x5, x0, loop | bne t0, x0, 132 | 11111110000000101001100011100011 | fe0298e3 | SB | #loop循环累加 次，结果存于：x5 |
| 98 |  | slli x12, x18, 0 | slli a2, s2, 0 | 00000000000010010001011000010011 | 00091613 | I | #X12<- x18 ，X12 = 0x0000000a，函数调用结果存于：x12 |
| 9c |  | Jr ra | jalr x0, 0(ra) | 00000000000000001000000001100111 | 00008067 | I | #函数sum调用返回，回到pc = ra |

**思考**

以上代码段最开始将数据1，2，3，4写到了哪些存储器地址单元?可以查看memory页面，找到被更新的单元，作截图。



Ripes软件中是将.data段起始地址默认设定为0x10000000的。如果用Ripes来仿真，为了便于在 .Data段直接观测，如何修改测试程序的前两句，才可以将1，2，3，4写到0x10000000为基址的 .Data段起始位置? 修改后完成仿真，查看memory页面，找到.data段的起始位置，看看是否被改写?从0x10000000开始的几个单元是否被更新?理解对应的语句功能含义。可将前两句改为lui x10, 0x10000

ori x4, x10, 0 。后测试程序，可以看到0x10000000开始的几个单元更新。

如果使用仿真平台2 完成以上仿真的话直接用该代码段是否可以? 提示：程序加载与仿真界面截图，如图2和图3。如果在仿真平台2上进行仿真，第2句可改为ori x4, x10, 1024，其他不变。因为该平台数据存储器（data段）设置的偏移量为地址1024处开始。为便于在界面直接观察数据存储器内的变化结果，可以做此修改。不可以，会报错。