**关于测试多周期CPU的简单方法**

**（特别说明：本表每个同学都必须建立，检查实验时，必须提供！）。**

1. 测试程序段

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **地址** | **汇编程序** | **指令代码** | | | | | |
| **op（6）** | **rs(5)** | **rt(5)** | **rd(5)/immediate (16)** | **16进制数代码** | |
| **0x00000000** | addiu $1,$0,8 | 001001 | **00000** | **00001** | **0000 0000 0000 1000** | **=** | 24010008 |
| **0x00000004** | ori $2,$0,2 | 001101 | 00000 | 00010 | 0000 0000 0000 0010 | **=** | 34020002 |
| **0x00000008** | xori $3,**$2**,8 | 001110 | 00010 | 00011 | **0000 0000 0000 1000** | **=** | 38430008 |
| **0x0000000C** | sub $4,**$3**,$1 | 000000 | 00011 | 00001 | 0010000000100010 | **=** | 00612022 |
| **0x00000010** | and $5,**$4**,$2 | 000000 | 00100 | 00010 | 0010100000100100 | **=** | 00822824 |
| **0x00000014** | **sll $5,$5,2** | 000000 | 00000 | 00101 | **00101 00010 000000** | **=** | 01252880 |
| **0x00000018** | **beq $5,$1,-2(=,转14)**  **offset左移两位加上当前PC的值，作为分支跳转地址** | 000100 | 00101 | 00001 | 1111111111111110 | **=** | c0a1fffe |
| **0x0000001C** | **jal 0x0000050** | 000011 | 00000 | 00000 | **0000 0000 0001 0100** | **=** | 0c0000014 |
| **0x00000020** | **slt $8,$13,$1** | 000000 | 01101 | 00001 | 0100000000101010 | **=** | 01a1402a |
| **0x00000024** | **addiu $14,$0,-2** | 001001 | 00000 | 01110 | 1111111111111110 | **=** | 240efffe |
| **0x00000028** | **slt $9,$8,$14** | 000000 | 01000 | 01110 | 0100100000101010 | **=** | 010e482a |
| **0x0000002C** | **slti $10,$9,2** | 001010 | 01001 | 01010 | **0000000000000010** | **=** | 292a0002 |
| **0x00000030** | **slti $11,$10,0** | 001010 | 01 010 | 0 1011 | **0000000000000000** | **=** | 294b0000 |
| **0x00000034** | add $11,**$11**,$10 | 000000 | 01011 | 01010 | 01011 00000 100000 | **=** | 016a5820 |
| **0x00000038** | **bne $11,$2,-2 (≠,转34)** | 000101 | 00010 | 01011 | 1111111111111110 | **=** | 144bfffe |
| **0x0000003C** | addiu $12,$0,-2 | 001001 | 00000 | 01100 | 1111111111111110 | **=** | 240cfffe |
| **0x00000040** | addiu $12,$12,1 | 001001 | 01100 | 01100 | **0000000000000001** | **=** | 258c0001 |
| **0x00000044** | **bltz $12,-2 (<0,转40)** | 000001 | 01100 | 00000 | 1111111111111110 | **=** | 0580fffe |
| **0x00000048** | andi $12,$2,2 | 001000 | 00010 | 01100 | **0000000000000010** | **=** | 204c0002 |
| **0x0000004C** | **j 0x000005C** | 000010 | 00000 | 00000 | **0000000000010111** | **=** | 08000017 |
| **0x00000050** | sw $2,4($1) | 101011 | 00001 | 00010 | 0000 0000 0000 0100 | **=** | ac220004 |
| **0x00000054** | lw $13,4($1) | 100011 | 00001 | 01101 | 0000000000000100 |  |  |
| **0x00000058** | **jr $31** | 000000 | 11111 | 00000 | 0000000000001000 |  |  |
| **0x0000005C** | **halt** | 111111 | 00000 | 00000 | 0000000000000000 | **=** | fc000000 |
|  |  |  |  |  |  |  |  |

1. 将**指令代码初始化到指令存储器**中，直接写入。
2. 初始化PC的值，也就是以上程序段首地址PC=**0x00000000**，以上程序段从**0x00000000**地址开始存放。
3. 运行Xilinx Vivado进行仿真，看波形。

PC地址以16进制表示