

第五周作业

(2) 否

3. 1) nop: addi x0, x0, 0

ARM 可配置是否异常

2) ret: jalr x0, x1, 0

3) call offset: auipc x6, offset[31, 12]

jalr x1, x6, offset[11:0] 12. Linux kernel 3

4) mv: addi rd, rs, 0

Boot ROM 1

5) rdcycle rd: csrcs rd, ~~ms~~ cycle, x0

BootLoader 1

6) sext.w: addiw rd, rs, 0

USB Driver 1

Vim

0

7. 1) sli t3, t2, 2

sli t4, t2, t1

13. addi sp, sp, -32

(2) add t0, t1, t2

sd ra, 24(sp)

bltu t0, t1, overflow

sd s0, 16(sp)

3) ARM 中通过 CPSR 的状态

addi s0, sp, 32

寄存器反映当前指令的溢出状态

li t3, 0

MIPS 溢出时通过指令触

li t4, 100

发中断的方式产生溢出信号

beq t3, t4, end

x86 x86 的 ALU 中会给出溢出信号

mv t5, t0

mv t6, t1

8. 1) $2^{LEN} - 1$, x -1 x

part 1:

beq t3, t4, end

2) IV: Invalid operation

mul (t5), (t6), t2

DZ: Divided by Zero

addi t5, t5, 4

OF: Overflow

addi t6, t6, 4

UF: Underflow

NX: Inexact

addi t3, t3, 1

j part1.
End:

ld ra, 24(sp)

ld s0, 16(sp)

addi sp, sp, 32

ret

int a1 = 1;

17. for (i=0; i<30; i++) {

a1 = a1 * 2

}

#14

```
start:  addi    sp, sp, -32
        sd     ra, 24(sp)
        sd     s0, 16(sp)
        addi    s0, sp, 32

        blt    a1, a0, part1
        add    a2, a0, a1
        j     end
```

part1:

```
        sub    a2, a0, a1
```

end:

```
        ld     ra, 24(sp)
        ld     s0, 16(sp)
        addi    sp, sp, 32
        ret
```

#15

```
        sw     t0, (t0)
        li     t1, 3
        sw     t1, 4(t0)
        mv     t2, t0
        mv     t3, t1
        slli   t3, t1, 2
        add    t2, t2, t3
        sw     t1, (t2)
```

#16

#void swap(int *a, int *b)

```
        addi    sp, sp, -32
        sw      ra, 28(sp)
        sw      s0, 24(sp)
        addi    s0, sp, 32
        sw      t0, -12(s0)
        sw      t1, -16(s0)
        lw      t0, -12(s0)
        lw      t0, 0(t0)
        sw      t0, -20(s0)
        lw      t0, -16(s0)
        lw      t0, 0(t0)
        lw      t1, -12(s0)
        sw      t0, 0(t1)
        lw      t0, -20(s0)
        lw      t1, -16(s0)
        sw      t0, 0(t1)
        lw      ra, 28(sp)
        lw      s0, 24(sp)
        addi    sp, sp, 32
        ret
```