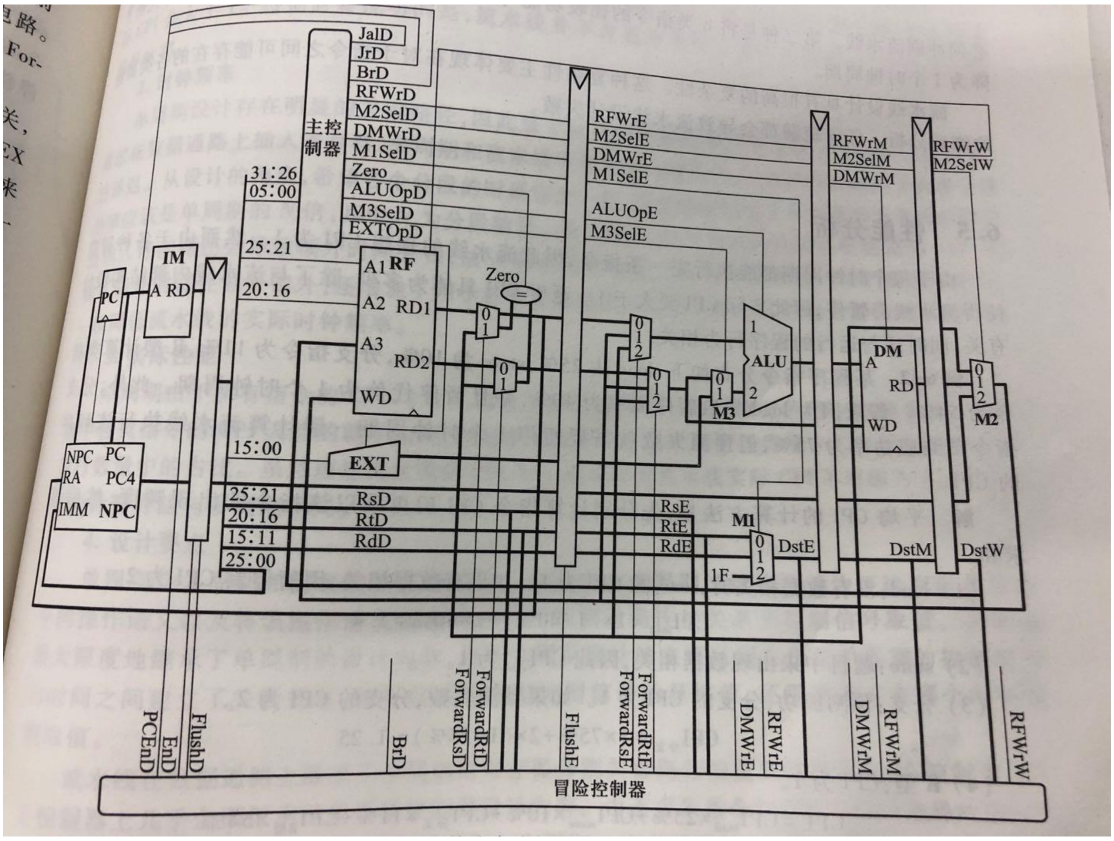
P6 实验报告

一．整体结构：

流水线处理器包括流水寄存器、各级组合逻辑、各级控制器和冲突处理模块四大部分

支持的指令集为：addu,subu, ori, lw, sw, beq, lui, j,jal, jr,nop



二、模块设计

1. PC（程序计数器）

用于输出当前指令的 PC 值，和保存下一条指令的 PC 值。该模块由一个 32 位寄存器构成。

模块接口

信号名	方向	描述
npc[31:0]	I	下一条指令的 PC 值
clk	I	时钟信号

reset	I	复位信号 1：有效 0：无效
pc[31:0]	O	当前指令 PC 值
pcend	I	暂停信号 1：暂停 0：继续

功能定义

序号	功能名称	功能描述
1	输出 PC	在 clk 上跳沿输出当前指令的 PC 值
2	复位	当复位信号有效时，PC 被设置为起始地址 0x00003000
3	暂停	暂停更新 PC 的值

2. NPC

用于计算下一条指令的 PC 值

模块接口

信号名	方向	描述
PC [31:0]	I	当前指令 PC 值
Imme[25:0]	I	26 位立即数
ra [31:0]	I	当前 RD1
brd	I	Beq 的跳转
jald	I	Jal 的跳转
jrd	I	Jr 的跳转
pc4[31:0]	O	PC + 4
npc	O	下一条指令的 PC 值

序号	功能名称	功能描述
1	输出 PC	输出下一条指令的 PC 值
2	PC + 4	输出当前指令的 PC + 4

3. IM（指令存储器）

IM 容量为 32bit×4096，实际地址宽度为 12 位，从而将地址的 2~13 位连接到 IM 选择地址端口。

模块接口

信号名	方向	描述
clk	I	时钟信号
reset	I	复位信号
pc	I	Pc 的值
Instr	O	指令

功能定义

序号	功能名称	功能描述
1	输出指令	根据当前 PC 值输出所对应的指令

4. GRF（通用寄存器组）

由 32 个 32 位寄存器构成，其中$0 始终保持为 0

模块接口

信号名	方向	描述
clk	I	时钟信号

reset	I	复位信号 1：有效 0：无效
PC [31:0]	I	当前指令的 PC 值
wr	I	读写控制信号 1：写操作 0：读操作
a1 [4:0]	I	读寄存器 1 的地址
a2 [4:0]	I	读寄存器 2 的地址
a3 [4:0]	I	写寄存器的地址
wd [31:0]	I	向写寄存器中写入的值
rd1e [31:0]	O	a1 所对应的寄存器的值
rd2e [31:0]	O	a2 所对应的寄存器的值

功能定义

序号	功能名称	功能描述
1	复位	当复位信号有效时，所有寄存器的值被设置为 0
2	写寄存器	根据输入的写寄存器地址，把输入的数据写入写寄存器中
3	读寄存器	根据输入的读寄存器地址，将数据读出

5. ALU（算术逻辑单元）

提供 32 位加、减、或运算

可以不支持溢出

模块接口

信号名	方向	描述
A [31:0]	I	ALU32 位输入数据 A
B [31:0]	I	ALU32 位输入数据 B

ALUop[1:0]	I	ALU 功能选择信号 00:加法 01:减法 10:或运算
Result[31:0]	O	32 位计算结果
Zero	O	A == B

功能定义

序号	功能名称	功能描述
1	加	A + B
2	减	A - B
3	或	A | B

6. DM（数据存储器）

DM 容量为 32bit×4096，其起始地址为 0x0000_0000

模块接口

信号名	方向	描述
clk	I	时钟信号
reset	I	复位信号 1：有效 0：无效
wr	I	读写控制信号 1：写操作 0:读操作
pc	I	对应指令的 PC 值
result[31:0]	I	所要进行操作的地址
A	I	读或写的地址
wd [31:0]	I	写入数据的输入

rd [31:0]	O	读取数据的输出

功能定义

序号	功能名称	功能描述
1	复位	当复位信号有效时，所有数据被设置为 0x00000000
2	写操作	根据输入 A，把输入的数据写入
3	读操作	根据输入 A，将其中的数据读出

7. EXT（位扩展器）

模块接口

信号名	方向	描述
imm [15:0]	I	16 位立即数
EOp[1:0]	I	位扩展选择信号 00：无符号扩展 01：有符号扩展 10：扩展至[31:16]位，低 16 位补 0
ext [31:0]	O	位扩展后的 32 位输出

功能定义

序号	功能名称	功能描述
1	高位补 0	高 16 位补 0
2	低位补 0	低 16 位补 0
3	符号扩展	若符号位为 0，则高位补 0 若符号位为 1，则高位补 1

8. 控制器（Controller）

模块接口

信号名	方向	描述
opcode[5:0]	I	Opcode
func[5:0]	I	Func
zero	I	cmp 的模块的等于判断信号
brd	I	Beq 信号（分支？）
jald	I	Jal 信号（类 jal 的跳转信号）
jrd	I	Jrd
m1sele[2:0]	O	GRF 输入地址的选择信号
m2sele[2:0]	O	GRF 写入数据的选择信号
rfwrd	O	GRF 写使能
Dmwre	O	DM 写使能
m3sele[2:0]	O	ALU 的 B 输入的控制信号
extope[1:0]	O	位扩展控制信号
aluope[1:0]	O	ALU 行为控制信号

9. chengchuqi（乘除槽）

模块接口

信号名	方向	描述
Clk	I	时钟信号
Reset	I	复位信号
op	I	信号选择信号
D1	I	被运算数
D2	I	运算数
HI-out	O	HI 寄存器的输出
LO-out	O	LO 寄存器的输出

Start	O	乘除槽开始运算标志
Buzy	O	乘除槽正在运算标志

10.　BE（字节使能模块）

模块接口

信号名	方向	描述
A1_0	I	地址信号的最后两位
BE	O	字节使能
sw	I	Sw 的信号流水
sh	I	Sh 的信号流水
sb	I	Sb 的信号流水

图 3 给出了增加 BE 扩展的数据通路局部参考设计。显然，BE 扩展功能部件还需要有来自控制器的控制信号。注意：由于 DM 容量有限，因此ALU 计算出来的 32 位地址没有必要也不可能都用上。

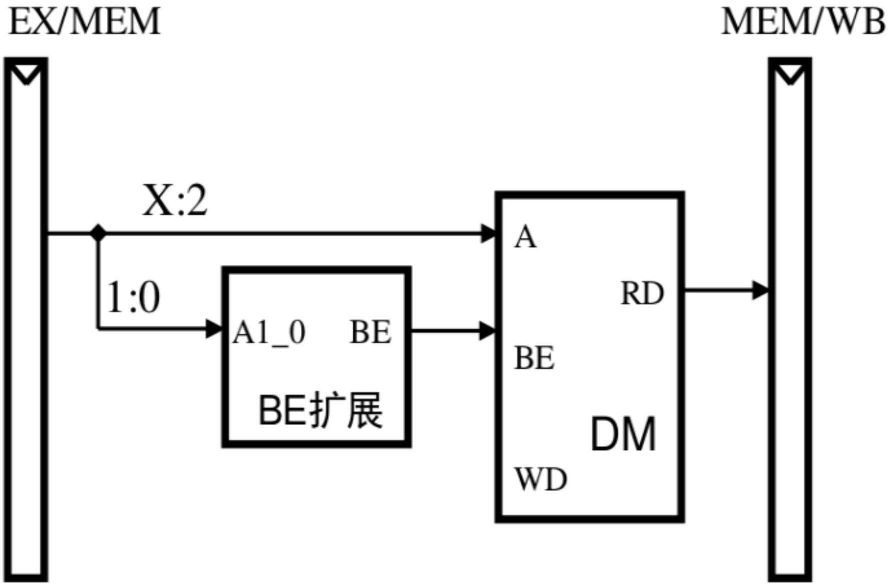
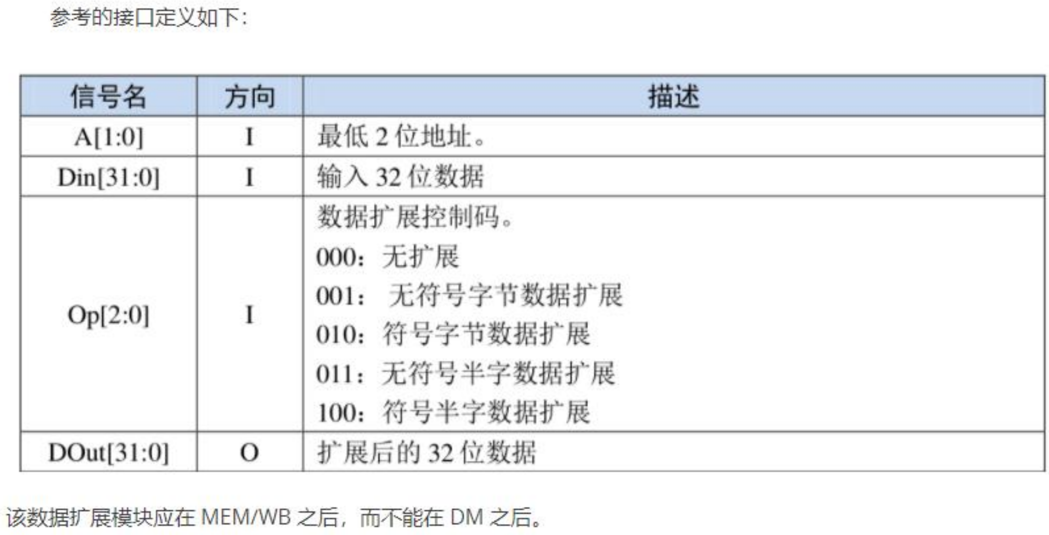


图 3 BE 扩展

对于 lb、lbu、lh、lhu 来说，你必须额外增加一个数据扩展模块。这个模块把从DM 读出的数据做符号或无符号扩展。

11.　RDKUOZHAN（DM_RD 位扩展模块）

模块接口



参考的接口定义如下:

信号名	方向	描述
A[1:0]	I	最低 2 位地址。
Din[31:0]	I	输入 32 位数据
Op[2:0]	I	数据扩展控制码。 000：无扩展 001：无符号字节数据扩展 010：符号字节数据扩展 011：无符号半字数据扩展 100：符号半字数据扩展
DOut[31:0]	O	扩展后的 32 位数据

该数据扩展模块应在 MEM/WB 之后，而不能在 DM 之后。

三、冲突

1.RR 冲突

2.LR 冲突，暂停一周期转发

3.RB，LB，LXB 冲突

```
init:
   addu $28,$0,$0
   addu $29,$0,$0
   addu $30,$0,$0
   addu $31,$0,$0
   ori $28,$28,4
block_0:
   lui $1,0xffff
   ori $1,$1,0xaa99
   lui $2,0x0000
   ori $2,$2,0x787f
   addu $26,$1,$2
   sw $26,0($29)
   addu $29,$29,$28
   j block_1
   nop

block_1:
   lui $4,0xffff
   ori $4,$4,0x85dc
   lui $5,0xffff
   ori $5,$5,0x83d3
   addu $0,$4,$5
   sw $0,0($29)
   addu $29,$29,$28
   j block_2
   nop

block_2:
   lui $16,0xffff
   ori $16,$16,0xd9ef
   lui $17,0xffff
   ori $17,$17,0x8dcd
   subu $27,$16,$17
   sw $27,0($29)
   addu $29,$29,$28
   j block_15
   nop



block_15:
   lui $2,0xffff
   ori $2,$2,0xf12e
   ori $19,$2,0xf0ad
   sw $19,0($29)
   addu $29,$29,$28
   j block_16
   nop

block_16:
   lui $26,0xffff
   ori $26,$26,0xe2b7
   ori $0,$26,0xd416
   sw $0,0($29)
   addu $29,$29,$28
   j block_17
   nop

block_17:
   lui $14,0xffff
   ori $14,$14,0xe504
   ori $15,$0,2141
   lb $14,-2117($15)
   sw $14,0($29)
   addu $29,$29,$28
   sw $15,0($29)
```

```
    addu $29,$29,$28
    j block_18
    nop

block_18:
    lui $19,0xffff
    ori $19,$19,0xd257
    ori $8,$0,30066
    subu $8,$0,$8
    lb $19,30127($8)
    sw $19,0($29)
    addu $29,$29,$28
    sw $8,0($29)
    addu $29,$29,$28
    j block_19
    nop

block_19:
    lui $23,0xffff
    ori $23,$23,0x8395
    ori $24,$0,13769
    subu $24,$0,$24
    lb $23,13799($24)
    sw $23,0($29)
    addu $29,$29,$28
    sw $24,0($29)
    addu $29,$29,$28
    j block_20
    nop

block_20:
    lui $12,0x0000
    ori $12,$12,0x1733
    ori $4,$0,24393
    lb $12,-24326($4)
    sw $12,0($29)
    addu $29,$29,$28
    sw $4,0($29)
    addu $29,$29,$28
    j block_21
    nop

block_21:
    lui $27,0xffff
    ori $27,$27,0x98f4
    ori $11,$0,16440
    subu $11,$0,$11
    lbu $27,16468($11)
    sw $27,0($29)
    addu $29,$29,$28
    sw $11,0($29)
    addu $29,$29,$28
    j block_28
    nop


block_28:
    lui $2,0x0000
    ori $2,$2,0x2e29
    ori $21,$0,22215
    lhu $2,-22129($21)
    sw $2,0($29)
    addu $29,$29,$28
    sw $21,0($29)
    addu $29,$29,$28
    j block_29
    nop

block_29:
    lui $22,0xffff
    ori $22,$22,0xdb7e
    ori $5,$0,713
    subu $5,$0,$5
    sb $22,757($5)
    sw $22,0($29)
    addu $29,$29,$28
    sw $5,0($29)
    addu $29,$29,$28
    j block_30
    nop

block_30:
    lui $25,0x0000
    ori $25,$25,0x1bac
    ori $17,$0,27745
    subu $17,$0,$17
    sb $25,27866($17)
    sw $25,0($29)
    addu $29,$29,$28
    sw $17,0($29)
    addu $29,$29,$28
    j block_31
    nop

block_31:
    lui $7,0x0000
    ori $7,$7,0x2290
    ori $3,$0,6465
    sb $7,-6299($3)
    sw $7,0($29)
    addu $29,$29,$28
    sw $3,0($29)
    addu $29,$29,$28
    j block_32
    nop

block_32:
    lui $17,0x0000
    ori $17,$17,0x0f93
    ori $20,$0,4826
    sb $17,-4659($20)
    sw $17,0($29)
    addu $29,$29,$28
    sw $20,0($29)
    addu $29,$29,$28
    j block_33
    nop

block_33:
    lui $10,0x0000
    ori $10,$10,0x4cb4
    ori $13,$0,3571
    sh $10,-3447($13)
    sw $10,0($29)
    addu $29,$29,$28
    sw $13,0($29)
    addu $29,$29,$28
    j block_34
    nop

block_34:
    lui $9,0x0000
    ori $9,$9,0x64ed
    ori $11,$0,14172
    sh $9,-13970($11)
    sw $9,0($29)
    addu $29,$29,$28
    sw $11,0($29)
    addu $29,$29,$28
    j block_35
    nop

block_35:
    lui $5,0x0000
    ori $5,$5,0x2a25
    lui $22,0xffff
    ori $22,$22,0x9df5
    add $15,$5,$22
    sw $15,0($29)
    addu $29,$29,$28
    j block_36
    nop

block_36:
    lui $19,0x0000
    ori $19,$19,0x5ab2
    lui $4,0x0000
    ori $4,$4,0x5fe4
    add $0,$19,$4
    sw $0,0($29)
    addu $29,$29,$28
    j block_37
    nop

block_37:
    lui $3,0x0000
    ori $3,$3,0x0595
    lui $18,0xffff
    ori $18,$18,0xd204
    sub $2,$3,$18
    sw $2,0($29)
    addu $29,$29,$28
    j block_38
    nop

block_38:
    lui $15,0x0000
    ori $15,$15,0x07ce
    lui $6,0x0000
    ori $6,$6,0x6c89
    sub $0,$15,$6
    sw $0,0($29)
    addu $29,$29,$28
    j block_39
    nop

block_39:
    lui $21,0xffff
    ori $21,$21,0x8675
    lui $8,0xffff
    ori $8,$8,0xaa15
    sll $1,$8,1
    sw $1,0($29)
    addu $29,$29,$28
    j block_40
    nop

block_40:
    lui $20,0xffff
    ori $20,$20,0x8b2c
    lui $10,0xffff
    ori $10,$10,0x952b
    sll $0,$10,28
    sw $0,0($29)
    addu $29,$29,$28
    j block_41
    nop

block_41:
    lui $24,0xffff
    ori $24,$24,0xc528
    lui $27,0xffff
    ori $27,$27,0x8a3e
    srl $3,$27,25
    sw $3,0($29)
    addu $29,$29,$28
    j block_50
    nop
```

```
block_50:
    lui $21,0x0000
    ori $21,$21,0x0b42
    lui $22,0x0000
    ori $22,$22,0x0656
    srav $0,$21,$22
    sw $0,0($29)
    addu $29,$29,$28
    j block_51
    nop

block_51:
    lui $27,0x0000
    ori $27,$27,0x1c84
    lui $20,0x0000
    ori $20,$20,0x1652
    and $23,$27,$20
    sw $23,0($29)
    addu $29,$29,$28
    j block_52
    nop

block_52:
    lui $5,0xffff
    ori $5,$5,0x91d1
    lui $7,0x0000
    ori $7,$7,0x14a2
    and $0,$5,$7
    sw $0,0($29)
    addu $29,$29,$28
    j block_53
    nop

block_53:
    lui $6,0x0000
    ori $6,$6,0x2195
    lui $3,0x0000
    ori $3,$3,0x3204
    or $19,$6,$3
    sw $19,0($29)
    addu $29,$29,$28
    j block_63
    nop
```

```
block_63:
   lui $21,0x0000
   ori $21,$21,0x3bcd
   addiu $15,$21,-0x1db9
   sw $15,0($29)
   addu $29,$29,$28
   j block_64
   nop

block_64:
   lui $1,0xffff
   ori $1,$1,0xdcaa
   addiu $0,$1,-0x1d20
   sw $0,0($29)
   addu $29,$29,$28
   j block_65
   nop

block_65:
   lui $27,0xffff
   ori $27,$27,0xe46b
   andi $1,$27,0x702d
   sw $1,0($29)
   addu $29,$29,$28
   j block_66
   nop

block_66:
   lui $16,0xffff
   ori $16,$16,0xa2ef
   andi $27,$16,0x809d
   sw $27,0($29)
   addu $29,$29,$28
   j block_67
   nop

block_67:
   lui $2,0xffff
   ori $2,$2,0x9c8e
   andi $0,$2,0xebda
   sw $0,0($29)
   addu $29,$29,$28
   j block_68
   nop
```

```
block_68:
    lui $11,0x0000
    ori $11,$11,0x3226
    xori $1,$11,0x4913
    sw $1,0($29)
    addu $29,$29,$28
    j block_72
    nop


block_72:
    lui $14,0x0000
    ori $14,$14,0x2cb5
    lui $5,0xffff
    ori $5,$5,0xafef
    slt $0,$14,$5
    sw $0,0($29)
    addu $29,$29,$28
    j block_73
    nop

block_73:
    lui $16,0xffff
    ori $16,$16,0x92f5
    slti $17,$16,0x3100
    sw $17,0($29)
    addu $29,$29,$28
    j block_76
    nop

block_76:
    lui $6,0xffff
    ori $6,$6,0x997e
    sltiu $15,$6,0x7cd5
    sw $15,0($29)
    addu $29,$29,$28
    j block_77
    nop

block_77:
    lui $9,0xffff
    ori $9,$9,0x9783
    sltiu $3,$9,-0x2c64
    sw $3,0($29)
    addu $29,$29,$28
    j block_78
    nop

block_78:
    lui $21,0x0000
    ori $21,$21,0x0c7a
    sltiu $0,$21,-0x211d
    sw $0,0($29)
    addu $29,$29,$28
    j block_79
    nop

block_79:
    lui $5,0xffff
    ori $5,$5,0xe441
    lui $14,0x0000
    ori $14,$14,0x1876
    sltu $21,$5,$14
    sw $21,0($29)
    addu $29,$29,$28
    j block_80
    nop

block_80:
    lui $13,0xffff
    ori $13,$13,0x97eb
    lui $7,0x0000
    ori $7,$7,0x0adb
    sltu $0,$13,$7
    sw $0,0($29)
    addu $29,$29,$28
    j block_81
    nop

block_81:
    lui $9,0x0000
    ori $9,$9,0x1227
    lui $24,0x0000
    ori $24,$24,0x6e19
    bne $9,$24,jump_block_5
    nop
    jump_back_5:
```

```
    sw $9,0($29)
    addu $29,$29,$28
    sw $24,0($29)
    addu $29,$29,$28
    j block_82
    nop

jump_block_5:
    nop
    j jump_back_5
    nop

block_82:
    lui $11,0x0000
    ori $11,$11,0x10e0
    lui $11,0xffff
    ori $11,$11,0xaf57
    bne $11,$11,jump_block_6
    nop
    jump_back_6:
    sw $11,0($29)
    addu $29,$29,$28
    sw $11,0($29)
    addu $29,$29,$28
    j block_83
    nop

jump_block_6:
    ori $30,$0,8
    j jump_back_6
    nop

block_83:
    lui $27,0x0000
    ori $27,$27,0x5a2e
    blez $27,jump_block_7
    nop
    jump_back_7:
    sw $27,0($29)
    addu $29,$29,$28
    j block_84
    nop

jump_block_7:
    nop
    j jump_back_7
    nop

block_84:
    lui $27,0xffff
    ori $27,$27,0x992f
    blez $27,jump_block_8
    nop
    jump_back_8:
    sw $27,0($29)
    addu $29,$29,$28
    j block_85
    nop

jump_block_8:
    ori $30,$0,8
    j jump_back_8
    nop

block_85:
    lui $0,0x0000
    ori $0,$0,0x7f6e
    blez $0,jump_block_9
    nop
    jump_back_9:
    sw $0,0($29)
    addu $29,$29,$28
    j block_86
    nop

jump_block_9:
    ori $30,$0,8
    j jump_back_9
    nop

block_86:
    lui $20,0x0000
    ori $20,$20,0x4b0d
    bgtz $20,jump_block_10
    nop
    jump_back_10:
    sw $20,0($29)
    addu $29,$29,$28
    j block_87
    nop

jump_block_10:
    nop
    j jump_back_10
    nop

block_87:
    lui $20,0xffff
    ori $20,$20,0x9907
    bgtz $20,jump_block_11
    nop
    jump_back_11:
    sw $20,0($29)
    addu $29,$29,$28
    j block_88
    nop

jump_block_11:
    ori $30,$0,8
    j jump_back_11
    nop

block_88:
    lui $0,0x0000
    ori $0,$0,0x5e4d
    bgtz $0,jump_block_12
    nop
    jump_back_12:
    sw $0,0($29)
    addu $29,$29,$28
    j block_89
    nop

jump_block_12:
    nop
    j jump_back_12
    nop

block_89:
    lui $23,0x0000
    ori $23,$23,0x6bda
    bltz $23,jump_block_13
    nop
    jump_back_13:
    sw $23,0($29)
    addu $29,$29,$28
    j block_90
    nop

jump_block_13:
    nop
    j jump_back_13
    nop

block_90:
    lui $23,0xffff
    ori $23,$23,0xeb69
    bltz $23,jump_block_14
    nop
    jump_back_14:

    sw $23,0($29)
    addu $29,$29,$28
    j block_91
    nop

jump_block_14:
    ori $30,$0,8
    j jump_back_14
    nop

block_91:
    lui $0,0x0000
    ori $0,$0,0x784a
    bltz $0,jump_block_15
    nop
    jump_back_15:

    sw $0,0($29)
    addu $29,$29,$28
    j block_92
    nop

jump_block_15:
    ori $30,$0,8
    j jump_back_15
    nop

block_92:
    lui $26,0x0000
    ori $26,$26,0x2c79
    bgez $26,jump_block_16
    nop
    jump_back_16:

    sw $26,0($29)
    addu $29,$29,$28
    j block_93
    nop

jump_block_16:
    ori $30,$0,8
    j jump_back_16
    nop

block_93:
    lui $26,0xffff
    ori $26,$26,0xf4aa
    bgez $26,jump_block_17
    nop
    jump_back_17:
    sw $26,0($29)
    addu $29,$29,$28
    j block_94
    nop

jump_block_17:
    nop
    j jump_back_17
    nop

block_94:
    lui $0,0x0000
    ori $0,$0,0x77a5
    bgez $0,jump_block_18
    nop
    jump_back_18:

    sw $0,0($29)
    addu $29,$29,$28
```

```
   j block_95
   nop

jump_block_18:
   ori $30,$0,8
   j jump_back_18
   nop

block_95:
   j jump_block_19
   nop
   jr_back_19:
      addu $13,$0,$31
      jalr $12,$13
      nop
   jump_back_19:
      nop

   j block_96
   nop

jump_block_19:
   jal jr_back_19
   ori $30,$0,8
   j jump_back_19
   nop

block_96:
   lui $1,0x24f6
   ori $1,$1,0xc67f
   lui $27,0x01d5
   ori $27,$27,0xd30e
   mult $1,$27
   j block_97
   nop

block_97:
   lui $15,0x08bd
   ori $15,$15,0xb6ce
   lui $25,0xdea9
   ori $25,$25,0x89cc
   multu $15,$25
   j block_102
   nop
```

```
block_102:
    lui $2,0xa524
    ori $2,$2,0xabdb
    lui $0,0x86c9
    ori $0,$0,0x941e
    mthi $2
    j block_103
    nop

block_103:
    lui $10,0x9730
    ori $10,$10,0xc54b
    lui $0,0x5f99
    ori $0,$0,0xc14a
    mtlo $10
    j block_104
    nop

block_104:
    beq $0,$0,block_104
    Nop
```

1. 随机测试

```
.data
.space 64
arr1: .space 64
arr2: .space 64
.space 64


.text

N0: slti $t2, $t0, 9032
N1: xor $zero, $t0, $t2
N2: mthi $t0
N3: lui $t2, 1155
N4: or $zero, $t1, $t0
N5: sra $t1, $zero, 3
```

```
N6: mflo $zero
N7: addi $t1, $t2, 21953
N8: mthi $t2
N9: mult $t2, $t0
N10: lui $zero, 40695
N11: lui $zero, 27425
N12: add $t0, $t0, $t1
N13: ori $t2, $t2, 1
div $t1, $t2
N14: and $t2, $t2, $t0
N15: addu $t0, $t0, $zero
N16: mflo $t0
N17: andi $t0, $t1, 60370
N18: xor $zero, $t0, $zero
N19: xor $zero, $t1, $t0
N20: addi $t1, $t0, 12625
N21: subu $t2, $t0, $t0
N22: mflo $t0
N23: mflo $zero
N24: xori $t0, $t0, 23836
N25: sra $zero, $zero, 19
N26: sra $t1, $t1, 11
N27: nor $zero, $t0, $t1
N28: or $t1, $zero, $t0
N29: mult $zero, $t2
N30: or $zero, $t0, $t1
N31: mult $t1, $t1
N32: add $t1, $t1, $zero
N33: mthi $t2
N34: mflo $t2
N35: mfhi $t0
N36: xor $zero, $t1, $t1
N37: mflo $t1
N38: srlv $t0, $t2, $zero
N39: sllv $t2, $t2, $t0
N40: add $zero, $t0, $t0
N41: nor $zero, $t1, $zero
N42: addu $t2, $t1, $t1
N43: xori $t0, $zero, 52965
N44: addiu $zero, $t0, 46776
N45: nor $t2, $t2, $t0
N46: addu $t2, $t1, $zero
N47: ori $t2, $t2, 1
divu $zero, $t2
```

```
N48: srl $t0, $t1, 3
N49: srl $zero, $t2, 28
N50: sltiu $t2, $zero, 5932
N51: multu $t2, $t2
N52: addiu $t2, $t1, 33416
N53: add $t0, $t1, $t0
N54: sltu $t1, $zero, $t2
N55: sub $zero, $zero, $t1
N56: ori $zero, $zero, 55192
N57: sltu $t2, $zero, $t0
N58: subu $t0, $t2, $t1
N59: add $t0, $t2, $t2
N60: srlv $t1, $t2, $t1
N61: mult $t2, $t1
N62: sltu $t2, $t1, $t1
N63: srlv $t1, $t1, $zero
N64: andi $t1, $t0, 19156
N65: addu $t1, $t0, $t1
N66: nop
N67: addi $t0, $t0, -32339
N68: ori $t0, $t0, 1
divu $zero, $t0
N69: ori $t2, $t2, 1
div $t0, $t2
N70: slt $t2, $t0, $t1
N71: sub $t1, $t2, $zero
N72: mtlo $t2
N73: and $t0, $t1, $t2
N74: addiu $t2, $t1, 56635
N75: srlv $zero, $t0, $t2
N76: mflo $zero
N77: sltu $t0, $t0, $t2
N78: ori $t2, $t2, 1
div $zero, $t2
N79: subu $t0, $zero, $t0
N80: subu $t2, $t1, $t1
N81: mult $t2, $t2
N82: addu $t1, $t1, $zero
N83: slt $t1, $zero, $zero
N84: xor $t1, $t0, $zero
N85: nop
N86: mult $t2, $t2
N87: or $t0, $zero, $zero
N88: srl $zero, $t2, 30
```

```
N89: ori $t2, $t2, 1
div $zero, $t2
N90: sra $zero, $t1, 6
N91: xor $t1, $t0, $t1
N92: slti $zero, $t2, -16324
N93: sll $t1, $t2, 10
N94: slt $zero, $t1, $t2
N95: nop
N96: ori $t0, $t2, 27082
N97: addu $t0, $t0, $t1
N98: lui $t1, 58199
N99: sub $t1, $zero, $t1
N100: or $zero, $t0, $t1
N101: andi $zero, $t1, 26236
N102: mflo $t2
N103: ori $t0, $t0, 1
div $t2, $t0
N104: nop
N105: ori $t1, $t1, 1
div $t1, $t1
N106: mflo $zero
N107: and $t2, $t1, $t0
N108: ori $t1, $t1, 1
div $t2, $t1
N109: mthi $t1
N110: ori $t2, $t2, 1
divu $t2, $t2
N111: sltiu $t0, $t2, 14040
N112: sltiu $zero, $t1, 2666
N113: mult $t2, $t0
N114: ori $t1, $t1, 1
divu $t0, $t1
N115: slt $zero, $t2, $t2
N116: andi $t0, $t1, 29622
N117: addiu $t0, $t2, 9602
N118: and $t2, $t1, $t0
N119: slti $zero, $t0, -14316
N120: srlv $t2, $t1, $t0
N121: slti $t0, $t2, -10097
N122: slt $t0, $t1, $t2
N123: multu $zero, $t1
N124: sltiu $t1, $t1, -29742
N125: addiu $t2, $zero, 36072
N126: ori $t1, $t1, 1
```

```
div $t2, $t1
N127: sll $t0, $t2, 14
N128: ori $t0, $t0, 1
div $t0, $t0
N129: or $t0, $t1, $zero
N130: srlv $t0, $t0, $zero
N131: mthi $t1
N132: andi $zero, $t0, 25162
N133: ori $t1, $t1, 1
divu $t1, $t1
N134: xor $zero, $t0, $t2
N135: multu $zero, $t2
N136: multu $zero, $t1
N137: ori $t1, $t1, 1
div $t0, $t1
N138: slt $t0, $zero, $t0
N139: xor $t1, $t2, $t2
N140: sra $t0, $t0, 1
N141: xori $t0, $t1, 55611
N142: addu $t0, $t1, $zero
N143: sll $t0, $zero, 17
N144: addu $t0, $t1, $t0
N145: sltiu $zero, $t0, -1869
N146: nor $t0, $t2, $t1
N147: srav $t0, $t2, $t0
N148: xor $t0, $t0, $t2
N149: or $t0, $t0, $zero
N150: addu $zero, $t0, $t0
N151: ori $t0, $t0, 1
divu $t1, $t0
N152: srl $zero, $zero, 29
N153: multu $zero, $zero
N154: sltiu $zero, $t1, -24791
N155: sllv $t2, $t2, $t0
N156: mthi $t2
N157: ori $zero, $t1, 21922
N158: subu $t2, $t0, $t1
N159: addiu $t0, $zero, 9637
N160: ori $t0, $t0, 1
div $t1, $t0
N161: or $t0, $t2, $t2
N162: sra $t1, $t2, 10
N163: nor $t2, $zero, $t1
N164: sltu $t1, $t0, $zero
```

```
N165: sub $t2, $t0, $t0
N166: andi $t2, $t2, 59451
N167: ori $t0, $t0, 1
div $zero, $t0
N168: mflo $t0
N169: nop
N170: sltiu $t1, $zero, 2477
N171: subu $zero, $t1, $zero
N172: addiu $t2, $t2, 45943
N173: andi $t0, $t2, 51775
N174: mult $zero, $zero
N175: slti $t1, $t0, -29124
N176: add $t2, $t0, $zero
N177: nop
N178: ori $t2, $t2, 1
divu $t0, $t2
N179: nor $t0, $t2, $t1
N180: subu $zero, $t2, $t0
N181: subu $t1, $t0, $t0
N182: sltiu $t2, $t0, 26365
N183: sltu $t2, $zero, $zero
N184: addu $t0, $t2, $t1
N185: addi $t1, $t0, 24960
N186: srlv $t2, $t2, $t0
N187: xor $t0, $t2, $t2
N188: mthi $t1
N189: srlv $t0, $t0, $t1
N190: add $t0, $t2, $zero
N191: ori $t0, $t0, 1
div $t0, $t0
N192: ori $t0, $t0, 1
divu $t1, $t0
N193: sltu $t2, $t0, $t0
N194: ori $t0, $t0, 1
divu $t2, $t0
N195: sllv $t0, $t0, $t1
N196: mtlo $zero
N197: addi $t2, $t1, -28515
N198: slti $t0, $t0, -27724
N199: srlv $t0, $t0, $t1
N200: ori $t2, $t0, 16720
N201: ori $t2, $t2, 5869
N202: or $zero, $t0, $t2
N203: mfhi $t2
```

```
N204: addiu $t0, $t1, 61750

N205: slti $t0, $t0, 3862

N206: mtlo $zero

N207: nop

N208: mflo $t2

N209: srlv $t2, $t0, $zero

N210: ori $t1, $t0, 7708

N211: xori $zero, $t0, 25111

N212: mthi $t1

N213: sltiu $t2, $t2, -22730

N214: ori $t2, $t1, 42465

N215: slti $zero, $zero, -4915

N216: slt $t1, $t0, $t0

N217: mtlo $t1

N218: mtlo $t1

N219: ori $t1, $t2, 3678

N220: mult $zero, $t1

N221: ori $t1, $t1, 1

divu $t2, $t1

N222: ori $zero, $t2, 24034

N223: nor $t2, $t0, $t1

N224: srlv $t0, $t2, $t0

N225: and $t1, $t2, $t1

N226: slt $t2, $t0, $t0

N227: mthi $zero

N228: srlv $t2, $t2, $zero

N229: srlv $zero, $t2, $zero

N230: subu $t0, $zero, $t1

N231: slti $t2, $zero, -30256

N232: addiu $t1, $t2, 15807

N233: xor $t2, $t0, $t0

N234: sllv $t0, $zero, $t2

N235: mflo $t2

N236: lui $t2, 13541

N237: addiu $zero, $t1, 10414

N238: nop

N239: or $t2, $t0, $t2

N240: or $t1, $zero, $t2

N241: slt $t0, $t1, $t1

N242: ori $t0, $t0, 1

divu $t2, $t0

N243: mtlo $t0

N244: srlv $zero, $t2, $t0

N245: srl $t0, $t1, 30
```

```
N246: slt $t2, $t0, $t1
N247: srl $t1, $t0, 17
N248: slti $t1, $zero, 6427
N249: sub $t0, $t2, $zero
N250: mfhi $t0
N251: sltiu $t1, $t1, -21540
N252: ori $t2, $t2, 1
divu $zero, $t2
N253: srav $t2, $t2, $t0
N254: subu $zero, $t0, $t2
N255: lui $t1, 13917
N256: srl $t2, $zero, 9
N257: sra $zero, $zero, 30
N258: addiu $t2, $t0, 10605
N259: slti $t2, $t1, 7393
N260: mfhi $t2
N261: subu $t2, $t1, $zero
N262: subu $zero, $t2, $t0
N263: ori $t2, $t2, 1
div $t1, $t2
N264: mflo $t1
N265: srlv $t1, $t1, $zero
N266: add $zero, $t1, $t1
N267: mthi $t1
N268: addiu $zero, $t0, 50878
N269: sub $t1, $t0, $t2
N270: mtlo $t2
N271: sub $t1, $t0, $t0
N272: mflo $t1
N273: or $t1, $t2, $zero
N274: srl $t0, $zero, 29
N275: addiu $t0, $zero, 62478
N276: andi $zero, $t0, 36293
N277: andi $t2, $zero, 45780
N278: sll $t0, $t1, 19
N279: srl $t1, $t2, 30
N280: mflo $t2
N281: srl $t2, $t2, 20
N282: sllv $t0, $t0, $t1
N283: lui $t1, 14496
N284: and $t2, $t1, $zero
N285: subu $t2, $t0, $t2
N286: lui $t2, 9106
N287: mthi $t2
```

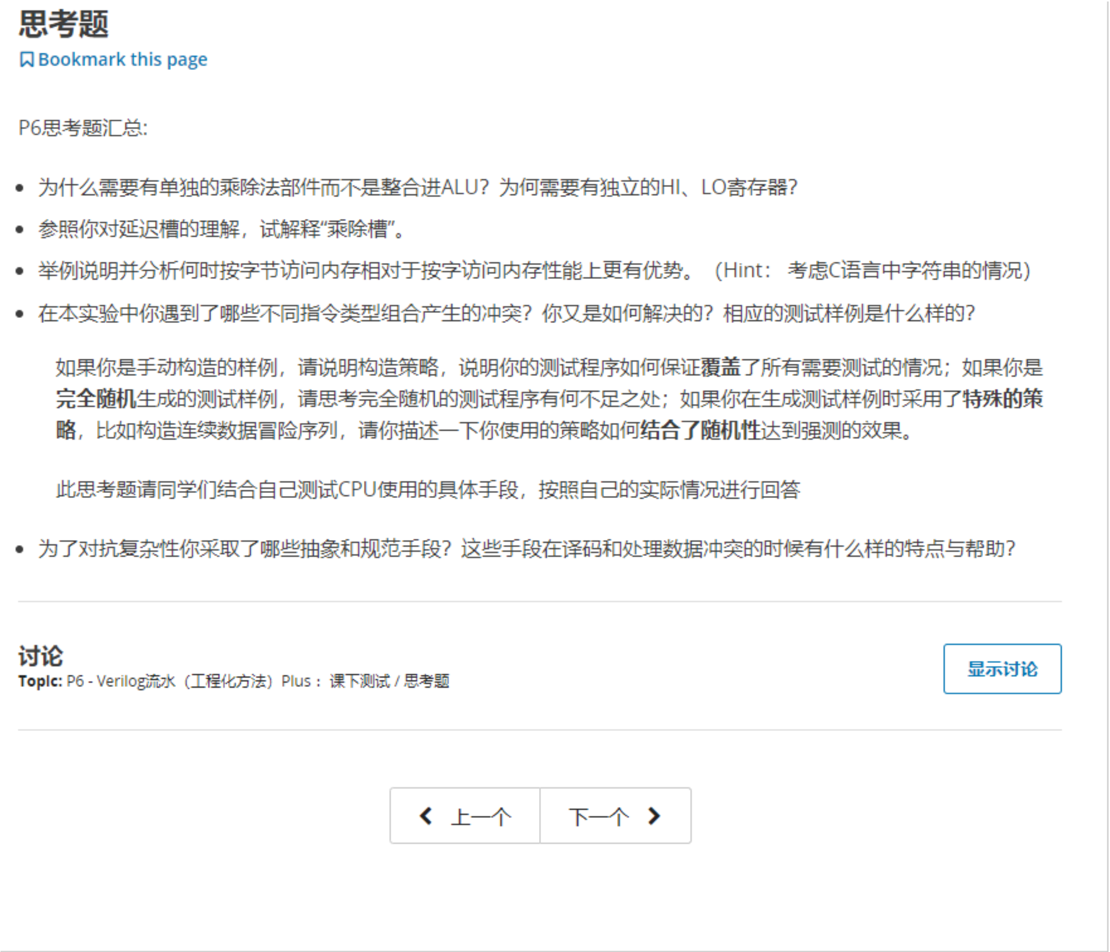
```
N288: addi $t0, $t1, 31426
N289: srav $t1, $t1, $t1
N290: slt $t0, $zero, $t0
N291: addiu $t0, $t2, 39063
N292: xori $t2, $t1, 64091
N293: ori $t0, $t0, 1
divu $t0, $t0
N294: srav $zero, $t2, $t0
N295: addiu $zero, $t1, 32372
N296: nop
N297: srl $t0, $t0, 19
N298: xori $zero, $zero, 12350
N299: sra $t0, $t0, 25
N300: mult $zero, $t1
N301: sllv $zero, $t1, $t2
N302: sltu $t2, $t1, $t2
N303: sltu $t1, $t2, $zero
N304: mfhi $t2
N305: srlv $zero, $t0, $t2
N306: slti $t0, $zero, 27349
N307: ori $t1, $t1, 1
divu $t2, $t1
N308: add $t2, $zero, $zero
N309: sllv $t0, $t0, $zero
N310: addi $t1, $t1, 3701
N311: ori $t1, $t1, 1
divu $t0, $t1
N312: sltu $zero, $t2, $zero
N313: sltu $t0, $zero, $t0
N314: srlv $t2, $t0, $t0
N315: slt $t2, $zero, $t1
N316: nop
N317: subu $zero, $t1, $t0
N318: ori $t2, $zero, 59747
N319: mfhi $zero
N320: lui $t2, 38498
N321: srlv $t2, $t0, $zero
N322: nop
N323: sltiu $t1, $t2, -8328
N324: sllv $t0, $t2, $t2
N325: srlv $t2, $zero, $t1
N326: srav $zero, $zero, $t1
N327: lui $zero, 43288
N328: and $zero, $t2, $zero
```

```
N329: or $t1, $zero, $t1

N330: multu $t1, $zero

N331: xor $t2, $t1, $zero

N332: subu $t0, $t0, $t0

N333: sltiu $t2, $zero, -27838

N334: xor $t2, $t2, $t0

N335: add $t1, $t2, $zero

N336: xori $t1, $zero, 49411

N337: srlv $t1, $zero, $t0

N338: sra $zero, $zero, 23

N339: sltu $t0, $t0, $t2

N340: sltu $t2, $t1, $t2

N341: srav $t1, $t1, $zero

N342: xori $t0, $zero, 2142

N343: xori $t1, $t2, 31452

N344: and $t0, $t2, $t0

N345: ori $t2, $t2, 1

div $t0, $t2

N346: multu $zero, $t2

N347: mflo $t1

N348: sub $t0, $t0, $t1

N349: srlv $t1, $t0, $t2

N350: addu $t1, $zero, $zero

N351: srl $t2, $zero, 2

N352: slti $zero, $t1, -6515

N353: slti $t2, $t1, -25065

N354: multu $t0, $t2

N355: addu $t2, $t2, $t1

N356: or $t2, $t2, $t2

N357: sub $t0, $t0, $t2

N358: ori $t1, $t1, 1

divu $zero, $t1

N359: or $t2, $t1, $zero

N360: addu $t0, $t1, $t2

N361: srl $t0, $t0, 3

N362: ori $t2, $t1, 49348

N363: sllv $t0, $t1, $t1

N364: srl $zero, $t0, 31

N365: mflo $zero

N366: mfhi $zero

N367: mthi $t0

N368: nop

N369: lui $t2, 720

N370: addiu $t1, $t2, 46117
```

```
N371: addiu $zero, $zero, 32219
N372: subu $t1, $zero, $t2
N373: ori $t0, $t0, 1
div $t2, $t0
N374: srav $t2, $t2, $t0
N375: sub $t1, $t1, $t1
N376: mult $t2, $t1
N377: mfhi $t1
N378: addiu $t2, $t1, 33508
N379: slt $t0, $zero, $t2
N380: sra $t1, $zero, 1
N381: srl $t2, $t2, 4
N382: or $zero, $t2, $t2
N383: or $t2, $t0, $zero
N384: sltiu $zero, $t1, 9303
N385: nop
N386: ori $t1, $t0, 30496
N387: mfhi $zero
N388: addiu $t1, $t2, 16466
N389: andi $t1, $t0, 18492
N390: sllv $t0, $t1, $t0
N391: mfhi $t1
N392: mult $t1, $t1
N393: ori $t2, $t1, 30243
N394: subu $zero, $t1, $zero
N395: mthi $zero
N396: slt $t0, $t0, $zero
N397: nor $zero, $t2, $zero
N398: subu $t0, $t1, $t0
N399: srlv $t0, $t2, $t2
N400: ori $t2, $t2, 1
div $t1, $t2
N401: srav $zero, $t0, $zero
N402: sltiu $t2, $zero, 32670
N403: sll $zero, $zero, 7
N404: multu $t2, $t0
N405: andi $zero, $t0, 14003
N406: srlv $t0, $t2, $t0
N407: slti $zero, $t0, -1389
N408: or $t1, $t2, $t1
N409: sll $t2, $t2, 23
N410: mflo $t2
N411: sra $t0, $t0, 24
N412: xori $zero, $t1, 1009
```

```
N413: addu $t2, $zero, $t0
N414: ori $t2, $t0, 41628
N415: srlv $zero, $t0, $t2
N416: andi $t2, $zero, 45886
N417: subu $t1, $t2, $t0
N418: slti $t1, $t2, -20700
N419: sub $zero, $t1, $zero
N420: lui $zero, 43915
N421: or $t0, $t1, $zero
N422: sltu $zero, $t0, $t2
N423: subu $t2, $t1, $t0
N424: ori $t1, $t1, 1
div $t2, $t1
N425: multu $t1, $t2
N426: addiu $zero, $t1, 41230
N427: multu $t0, $zero
N428: mflo $t1
N429: and $t2, $t2, $t0
N430: srl $t1, $t2, 7
N431: sll $t2, $t2, 15
N432: mthi $t1
N433: lui $t2, 25124
N434: mult $t2, $t0
N435: nor $zero, $zero, $zero
N436: addu $t1, $t2, $t2
N437: sub $t1, $zero, $t0
N438: addi $t0, $t0, -28319
N439: add $t0, $t1, $t2
N440: mflo $zero
N441: nor $t0, $t1, $t0
N442: slti $t2, $t2, 1844
N443: mfhi $zero
N444: lui $t2, 28145
N445: mfhi $t1
N446: srl $t0, $t0, 18
N447: or $t0, $t1, $zero
N448: subu $zero, $t1, $t1
N449: addi $t2, $zero, 195
N450: nop
EXIT:
beq $zero, $zero, EXIT
Nop
```

四、思考题



思考题

P6思考题汇总:

- 为什么需要有单独的乘除法部件而不是整合进ALU？为何需要有独立的HI、LO寄存器？
- 参照你对延迟槽的理解，试解释"乘除槽"。
- 举例说明并分析何时按字节访问内存相对于按字访问内存性能上更有优势。（Hint： 考虑C语言中字符串的情况）
- 在本实验中你遇到了哪些不同指令类型组合产生的冲突？你又是如何解决的？相应的测试样例是什么样的？

  如果你是手动构造的样例，请说明构造策略，说明你的测试程序如何保证**覆盖**了所有需要测试的情况；如果你是**完全随机**生成的测试样例，请思考完全随机的测试程序有何不足之处；如果你在生成测试样例时采用了**特殊的策略**，比如构造连续数据冒险序列，请你描述一下你使用的策略如何**结合了随机性**达到强测的效果。

  此思考题请同学们结合自己测试CPU使用的具体手段，按照自己的实际情况进行回答

- 为了对抗复杂性你采取了哪些抽象和规范手段？这些手段在译码和处理数据冲突的时候有什么样的特点与帮助？

讨论

Topic: P6 - Verilog流水（工程化方法）Plus： 课下测试 / 思考题

Q: 为什么需要有单独的乘除法部件而不是整合进 ALU？为何需要有独立的 HI、LO 寄存器？ A：1，ALU 进行乘除法在实际中会导致此级延迟数倍增大，大大降低流水线的效率；在模拟延迟中，ALU 无法实现时序逻 辑的周期延迟，2，乘除法的结果无法通过一个寄存器回写，3，实际需求中，需要存储乘除结果； 独立的 HI，LO 寄存器与实际需求相匹配，同时因为需要置于 EX 阶段，无法通过 ID 阶段的寄存器堆扩展来处理。

Q：参照你对延迟槽的理解，试解释"乘除槽"。 A：表面上，延迟槽就是分支和跳转指令后一条指令会执行（不会是分支和跳转），是通过编译来优化的一种方式，对于 CPU 而言，只是顺序执行指令；而乘除槽是执行乘除法运算时会产生的延迟，当 busy 有效时，后面乘除相关指令均会阻塞，所 以二者没有什么关系。

Q：举例说明并分析何时按字节访问内存相对于按字访问内存性能上更有优势。(Hint： 考虑 C 语言中字符串的情况） A：当访问一连串连续的字节，如字符串结构，按字访问每个字节均需要通过 32 位地址进行索引；而按字节，可每四次通 过高 30 位索引一个字，再低 2 位索引所需字节即可，低 2 位寻址比 32 位寻址效率上快的多。

Q：在本实验中你遇到了哪些不同指令类型组合产生的冲突？你又是如何解决的？相应的测试样例是什么样的？ A：本方案是在 p5 的基础上进行扩充，对于冲突而言，无新类型的产生（其中乘除结果通过多选器与 alu 输出结果多选， 相当于并入 alu 类），故在 p5 方案基础上，增加对扩充指令的功能测试即可。

Q：为了对抗复杂性你采取了哪些抽象和规范手段？这些手段在译码和处理数据冲突的时候有什么样的特点与帮助？ A：进行分类，p5 所支持指令集本身具有代表性，将新增指令加入其分类即可，更新的是乘除指令，通过新增乘除单元单 独处理即可，故复杂性的提升仅在量上，故控制信号驱动型 control 的构建方式优势就大大体现出来了，对于指令数倍的增 加，control 仅在 assign 赋值指令式 wire 需要增加代码量（方式很简易），以及对控制信号的 assign 并入或门即可。 译码上大大简化，处理冲突上通过特异性信号的流水，使结构目的清晰，代码可读性强。