流水线 MIPS_CPU 实验报告

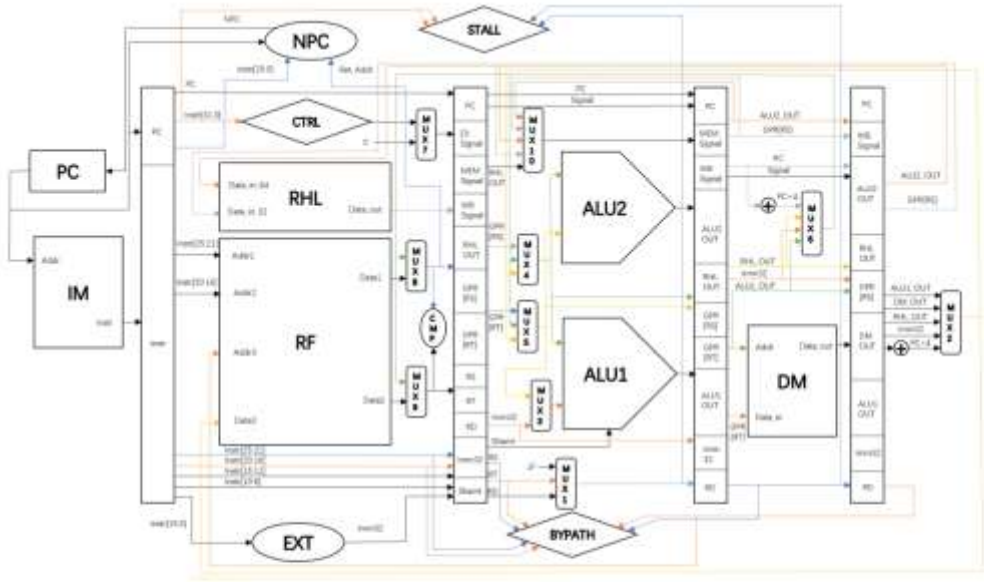
完成人员

1、姓名：杨士欣　　学号：2019302820
2、姓名：吴奇　　学号：2019302817
3、姓名：王翰墨　　学号：2019302683
4、姓名：王玉佳　　学号：2019302685

一、 任务概述

使用流水线技术，设计一个 57 条指令的 CPU。

二、 系统模块设计



ID signals	EX signals	MEM signals	WB signals	BYPATH signals	STALL signals	OUTPUT signals
EXTOp	ALU1Op	DMWr	RFWr	MUX4Sel	PCWr	CMPOut1
NPCOp	ALU2Op	DMSel	RHLWr	MUX5Sel	IF_IDWr	CMPOut2
IF_Flush	MUX1Sel	MUX6Sel	RHLSel_Wr	MUX8Sel	MUX7Sel	Overflow
RHLSel_Rd	MUX3Sel	DMRd	MUX2Sel	MUX9Sel		B_JOp
	ALU1Sel					

三、 详细设计

3.1 PC（程序计数器）

3.1.1 基本描述

PC 的主要功能是完成输出当前指令地址并保存下一条指令地址。复位后，PC 指向 0x0000_3000，此处为第一条指令的地址。

3.1.2 模块接口

信号名	方向	描述
clk	I	时钟信号（上升沿有效）

rst	I	复位信号（上升沿有效）
PCWr	I	PC 写使能信号 1：允许 NPC 写入 PC 内部寄存器 0：不允许 NPC 写入 PC 内部寄存器
NPC[31:0]	I	下一条指令的地址
PC[31:0]	O	30 位指令存储器地址

3.2 NPC（Next PC 计算单元）

3.2.1 基本描述

计算出下一条指令的地址并将其传给 PC 寄存器

3.2.2 模块接口

信号名	方向	描述
PC[31:0]	I	30 位指令存储器地址
Imm[25:0]	I	立即数
ret_addr[31:0]	I	针对 jr 和 jalr 的返回地址（即 GPR[rs]）
NPCOp[1:0]	I	00：NPC = PC + 4 01：NPC = PC + 4 + offset||00 10：NPC = PC[31:28]||imm[25:0]||00 11：NPC = ret_addr
PC[31:0]	O	30 位指令存储器地址
IF_Flush	O	清空流水线信号，若 NPC 不是 PC+4 并且当前流水线未被阻塞，则意味着之前取出 PC+4 的那条指令要被丢弃，详细清空操作见 IF/ID 模块

3.3 IM（指令存储器）

3.3.1 基本描述

IM 为指令存储器，容量为 8KB。

3.3.2 模块接口

信号名	方向	描述
addr[11:2]	I	访问地址

dout[31:0]	O	读出的指令

3.4 DM（数据存储器）

3.4.1 基本描述

DM 为数据存储器，容量为 8KB。这里需要注意的是，由于 SB 和 SH 不满 32 位，所以地址的末两位不一定是 0，因此还要特别判断地址末两位选择存在 32 位中的[7:0]或者[15:8]或者[23:16]或者[31:24]（SH 则是[15:0]或[31:16]）。

对于取字指令在内部已进行符号扩展。

3.4.2 模块接口

信号名	方向	描述
clk	I	时钟信号
din [31:0]	I	需要写回的数据
DMWr	I	读写操作的写使能端 0：禁止写 1：允许写
DMSel[2:0]	I	数据存储器访问控制信号，判断是存字或取字 000：SB 001：SH 010：SW 011：LBU 100：LB 101：LHU 110：LH 111：LW
addr[12:0]	I	访问地址
dout[31:0]	O	读出的数据

3.5 RF（通用寄存器组）

3.5.1 基本描述

32 个 32 位通用寄存器，r0~r31。其中有两个被赋予了特殊含义：r0，0 号通用寄存器，值永远为 0；r31，31 号通用寄存器，被 JAL，BLTZAL 和 BGEZAL 指令隐式的用作目标寄存器，存放返回地址。

3.5.2 模块接口

信号名	方向	描述
Addr1 [4:0]	I	需要读的寄存器 1 的地址
Addr2 [4:0]	I	需要读的寄存器 2 的地址
Addr3 [4:0]	I	需要写的寄存器的地址
WD [31:0]	I	需要写的寄存器的数据
RFWr	I	寄存器写使能端 0：寄存器不写 1：寄存器写
clk	I	时钟信号
RD1 [31:0]	O	需要读的寄存器 1 的数据
RD2 [31:0]	O	需要读的寄存器 2 的数据

3.6 ALU1（算术逻辑单元）

3.6.1 基本描述

完成加、减、或、与、或非、异或、算术移位（左移和右移）和逻辑移位（左移和右移），完成有符号数和无符号数的大小比较。

3.6.2 模块接口

信号名	方向	描述
A [31:0]	I	操作数 A
B [31:0]	I	操作数 B
Shamt[4:0]	I	Instr[10:6]
ALU1Op[3:0]	I	需要进行的运算 0000：加法 0001：减法 0010：或 0011：与 0100：或非 0101：异或 0110：逻辑左移/算术左移 0111：逻辑右移 1000：算术右移 1001：有符号比较 1010：无符号比较
ALU1Sel	I	针对移位信号：

		0：移位量为 A[4:0] 1：移位量为 shamt
Overflow	O	判断有符号数加减是否溢出 1：溢出 0：溢出
C [31:0]	O	运算结果

3.7 ALU2 （算术逻辑单元）

3.7.1 基本描述

完成有符号数和无符号数的乘法与除法，其中乘法和除法运算使用了 verilog 内置的乘除法器，有符号乘法直接对操作数进行符号扩展得到结果，有符号除法先将操作数转化成原码再进行除法操作，最后将结果再转化为补码。

3.7.2 模块接口

信号名	方向	描述
A [31:0]	I	操作数 A
B [31:0]	I	操作数 B
ALU2Op[1:0]	I	需要进行的运算 00：无符号乘法 01：有符号乘法 10：无符号除法 11：有符号除法
C [63:0]	O	运算结果。对于除法，余数放在高 32 位，结果放在低 32 位

3.8 EXT（扩展单元）

3.8.1 基本描述

将 16 位的数据通过不同方式扩展成为 32 位数据

3.8.2 模块接口

信号名	方向	描述
Imm16 [15:0]	I	需要进行扩展的数据
EXT1Op[1:0]	I	扩展方式的控制信号 00：0 扩展 01：符号扩展 10：将立即数扩展到高位
Imm32 [31:0]	O	扩展结果

3.9 RHL（Hi 和 Lo 寄存器）

3.9.1 基本描述

RHL 寄存器的高 32 位用作 Hi 寄存器，低 32 位用作 Lo 寄存器。Hi 寄存器存放乘法指令结果的高半部分或是除法指令结果的余数，LO 寄存器存放乘法指令结果的低半部分或是除法指令结果的商。

3.9.2 模块接口

信号名	方向	描述
Din_64[63:0]	I	乘除法得到的结果
Din_32[31:0]	I	GPR[rs]（针对 MTHI 和 MTLO 两条指令）
RHLWr	I	1：允许 RHL 寄存器被写入 0：不允许 RHL 寄存器被写入
RHLSel_Wr[1:0]	I	00：写 LO（低 32 位） 01：写 HI（高 32 位） 10：写 HI 和 LO
RHLSel_Rd	I	0：读 LO 1：读 HI
RHLOut[31:0]	O	RHL 寄存器中高 32 位或低 32 位的内容（由 RHLSel_Rd 决定）

3.10 MUX1

3.10.1 基本描述

3 选 1 选择器，生成 RF 寄存器组需要写的寄存器的地址

3.10.2 模块接口

信号名	方向	描述
Instr[31:0]	I	当前指令的内容
MUX1Sel[1:0]	I	00： Instr[20:16]（rt） 01： Instr[15:11] (rd) 10： 5'h1f
1FH	I	常量 1FH
Addr3	O	需要写的寄存器的地址

3.11 MUX2

3.11.1 基本描述

4 选 1 选择器，生成 RF 寄存器组需要写的寄存器的内容

3.11.2 模块接口

信号名	方向	描述
ALU1Out[31:0]	I	ALU1 进行运算得出的结果
RHLOut[31:0]	I	RHL 寄存器中高 32 位或低 32 位的内容（由 RHLSel_Rd 决定）
DMOut[31:0]	I	数据寄存器中的内容
PC[31:0]	I	存储指令地址，主要针对 AL 后缀指令，保存返回地址
Imm32[31:0]	I	EXT 扩展器扩展后的结果
MUX2Sel[2:0]	I	000： RHLOut 001： Imm32 010： ALU1Out 011： PC+4 100： DMOut
MUX2Out[31:0]	O	RF 寄存器组需要写的寄存器的内容，ALU1 和 ALU2 和 RHL 旁路所需要的数据

3.12 MUX3

3.12.1 基本描述

2 选 1 选择器，用来生成 ALU1 的第二个操作数。如果指令是立即数加法等带有立即数的操作，则选择 Imm32，反之选择 MUX5Out

3.12.2 模块接口

信号名	方向	描述
MUX5Out	I	MUX5 多选器的输出
Imm32	I	经过 EXT 扩展后的 32 位数据
MUX3Sel	I	0： MUX5Out 1： Imm32
B[31:0]	O	ALU1 的第二个操作数

3.13 MUX4

3.13.1 基本描述

旁路选择器，用来生成 ALU1 和 ALU2 的第一个操作数，旁路机制具体见旁路模块

3.13.2 模块接口

信号名	方向	描述
GPR_RS[31:0]	I	操作数来自寄存器堆
MUX6Out[31:0]	I	操作数由上一个 ALU（ALU1 或者 ALU2）运算结果或立即数（针对 LUI 指令）或者 PC+4（针对 AL 后缀指令）旁路获得
MUX2Out[31:0]	I	操作数从数据存储器或者前面的 ALU 结果或立即数（针对 LUI 指令）中旁路获得
MUX4Sel[1:0]	I	00： GPR[rs](RD1[31:0]) 01： MUX6Out 10： DMOut
A[31:0]	O	ALU1 和 ALU2 的第一个操作数

3.14 MUX5

3.14.1 基本描述

旁路选择器，用来生成 ALU1（未和立即数参与选择前）和 ALU2 的第二个操作数

3.14.2 模块接口

信号名	方向	描述
GPR_RT[31:0]	I	操作数来自寄存器堆
MUX6Out[31:0]	I	操作数由上一个 ALU（ALU1 或者 ALU2）运算结果或立即数（针对 LUI 指令）或者 PC+4（针对 AL 后缀指令）旁路获得
MUX2Out[31:0]	I	操作数从数据存储器或者前面的 ALU 结果或立即数（针对 LUI 指令）中旁路获得
MUX5Sel[1:0]	I	00： GPR[rt](RD2[31:0]) 01： MUX6Out 10： DMOut
B[31:0]	O	ALU2 的第二个操作数

3.15 MUX6

3.15.1 基本描述

4 选 1 选择器，用来生成 ALU1 和 ALU2 旁路时所需的数据，注意到 MUX6Sel 就是 MUX2Sel 的低 2 位，故控制信号不重复生成，只需生成 MUX2Sel

3.15.2 模块接口

信号名	方向	描述
ALU1Out[31:0]	I	ALU1 进行运算得出的结果
RHLOut[31:0]	I	RHL 寄存器中高 32 位或低 32 位的内容（由 RHLSel_Rd 决定）
DMOut[31:0]	I	数据寄存器中的内容
PC[31:0]	I	存储指令地址，主要针对 AL 后缀指令，保存返回地址

Imm32[31:0]	I	EXT 扩展器扩展后的结果
MUX6Sel[1:0]	I	00： RHLOut 01： Imm32 10： ALU1Out 11： PC+4
MUX6Out[31:0]	O	ALU1 和 ALU2 和 RHL 旁路时所需的数据

3.16 MUX7

3.16.1 基本描述

阻塞时，将控制信号中的写信号全部赋值为 0，阻塞机制详见阻塞模块

3.16.2 模块接口

信号名	方向	描述
WRSign[2:0]	I	DMWr、RFWr、RHLWr
0	I	常量
MUX7Sel	I	0：WRSign 1：0
MUX7Out[2:0]	O	流水线当前 ID/EX 级的写信号

3.17 MUX8

3.17.1 基本描述

旁路选择器，主要针对分支指令，由于分支指令可能用到 GPR[rs]和 GPR[rt]的值，故这里也要旁路

3.17.2 模块接口

信号名	方向	描述
MUX6Out[31:0]	I	旁路源结果来自 MUX6Out，详见 MUX6 模块
GPR_RS[31:0]	I	旁路源结果来自 RF 寄存器传来的 GPR[rs]
MUX8Sel	I	0：GPR_RS 1：MUX6Out
ID_EX_GPR_RS[31:0]	O	ID/EX 流水线寄存器中的 GPR[rs]

3.18 MUX9

3.18.1 基本描述

同 MUX8

3.18.2 模块接口

信号名	方向	描述
MUX6Out[31:0]	I	旁路源结果来自 MUX6Out，详见 MUX6 模块
GPR_RT[31:0]	I	旁路源结果来自 RF 寄存器传来的 GPR[rt]
MUX9Sel	I	0：GPR_RT 1：MUX6Out
ID_EX_GPR_RT[31:0]	O	ID/EX 流水线寄存器中的 GPR[rt]

3.19 MUX10

3.19.1 基本描述

旁路选择器，用来生成 EX/MEM 级流水线寄存器的 RHLOut 字段

3.19.2 模块接口

信号名	方向	描述
EX_MEM_ALU2Out[63:0]	I	旁路源结果来自上一个 ALU2 的输出（高 32 位或低 32 位）
MEM_WB_ALU2Out[63:0]	I	旁路源结果来自之前 ALU2 的输出（高 32 位或低 32 位）
EX_MEM_GPR_RS[31:0]	I	旁路源来自 EX/MEM 级的 GPR[rs]（针对 MTHI 和 MTLO 指令）
MEM_WB_GPR_RS[31:0]	I	旁路源来自 MEM/WB 级的 GPR[rs]（针对 MTHI 和 MTLO 指令）
RHLOut[31:0]	I	ID/EX 级的 RHLOut
MUX10Sel[2:0]	I	001：EX_MEM_ALU2Out[63:32] 010：EX_MEM_ALU2Out[31:0] 011：EX_MEM_GPR_RS[31:0] 100：MEM_WB_ALU2Out[63:32] 101：MEM_WB_ALU2Out[31:0]

		110：MEM_WB_GPR_RS[31:0] 111：RHLOut[31:0]
Out[31:0]	O	EX/MEM 级流水线的 RHLOut

3.20 CMP

3.20.1 基本描述

比较器，比较 GPR[RS]和 GPR[RT]的大小或者 GPR[RS]和 0 的大小

3.20.2 模块接口

信号名	方向	描述
GPR_RS[31:0]	I	GPR[rs]
GPR_RT[31:0]	I	GPR[rt]
CMPOut1	O	0：GPR[rs] = GPR[rt] 1：GPR[rs] != GPR[rt]
CMPOut2[1:0]	O	00：GPR[rs] = 0 01：GPR[rs] > 0 10：GPR[rs] < 0

3.21 BYPATH

3.21.1 基本描述

旁路模块，根据不同的冒险来生成相应的旁路信号。

3.21.2 旁路情况分析

- 当前指令需要读取寄存器的值要送入 ALU 单元，并且该值在之前与当前指令相邻的那条指令中被改变，则需要将 EX/MEM 寄存器中的值旁路到 ALU 的输入端口。
  例如：
  sub $2, $1, $3
  and $12, $2, $5
- 当前指令需要读取寄存器的值要送入 ALU 单元，并且该值在之前与当前指令相隔一条的那条指令中被改变，则需要将 MEM/WB 寄存器中的值旁路到 ALU 的输入端口。
  例如：

sub $2, $1, $3
and $12, $2, $5
or $13, $6, $2

- 当前指令需要读取寄存器的值要送入 ALU 单元，并且该值在之前与当前指令相隔两条的那条指令中被改变，在读操作中有一个判断，如果要读取的寄存器，是在下一个时钟上升沿要写入的寄存器，那么就将要写入的数据直接作为结果输出。如此就解决了相隔 2 条指令存在数据相关的情况。

  例如：

  sub $2, $1, $3
  and $12, $2, $5
  or $13, $6, $2
  add $14, $2, $2

  解决方法：

  assign RD1 = ((Addr1 == Addr3) && RFWr) ? WD : register[Addr1];

  其中 Addr1 是读取寄存器的地址，Addr3 是写入寄存器的地址，若两者相等，则直接将写入数据输出到结果，否则将读取寄存器的值输出到结果，两者相差一个时钟周期

- 当前指令需要读取寄存器的值要送入 CMP 单元或者当前指令是一条跳转（返回指令）并需要读取寄存器的值，并且该值在之前与当前指令相隔一条的那条指令中被改变（相邻的话需要阻塞一个周期再旁路），则需要将 EX/MEM 寄存器中的值旁路到 CMP 单元（ID/EX.RS 或者 ID/EX.RT）的输入端口。

  例 1：

  sub $2, $1, $3
  beq $2, $1, 7
  （先阻塞再旁路）
  例 2：
  sub $2, $1, $3
  jr $2
  (先阻塞再旁路)
  例 3：
  sub $2, $1, $3
  sub $5, $4, $3
  beq $2, $1, 7
  直接旁路

- 进行乘除法后或者数据移动指令（MTHI、MTLO）对 RHL 寄存器进行访问（MFHI、MFLO）也需要进行旁路，并判断是对 hi 部分或者 lo 部分进行旁路。该情况同样分为相邻指令数据相关情况和相隔一条指令数据相关情况，分别需要将 EX/MEM 或者 MEM/WB 寄存器中的值旁路到 ID/EX.RHLOut 的输入端口。

  例如：

  mult $2, $1, $3
  mfhi $4

```
mflo $5
```

3.21.3 模块接口

信号名	方向	描述
ID_EX_RS[4:0]	I	ID/EX 寄存器的 rs 字段
ID_EX_RT[4:0]	I	ID/EX 寄存器的 rt 字段
IF_ID_RS[4:0]	I	IF/ID 寄存器的 rs 字段
IF_ID_RT[4:0]	I	IF/ID 寄存器的 rt 字段
EX_MEM_RD[4:0]	I	EX/MEM 寄存器的 rd 字段
MEM_WB_RD[4:0]	I	MEM/WB 寄存器的 rd 字段
EX_MEM_RFWr	I	EX/MEM 寄存器的 RFWr 信号
MEM_WB_RFWr	I	MEM/WB 寄存器的 RFWr 信号
EX_MEM_RHLWr	I	EX/MEM 寄存器的 RHLWr 信号
ID_EX_RHLSelRd	I	ID/EX 寄存器的 RHLSelRd 信号
EX_MEM_RHLSelWr	I	EX/MEM 寄存器的 RHLSelWr 信号
MEM_WB_RHLWr	I	MEM/WB 寄存器的 RHLWr 信号
MEM_WB_RHLSelWr	I	MEM/WB 寄存器的 RHLSelWr 信号
BJOp	I	1：该指令是一条分支或者返回（JR、JALR）指令 0：该指令是其余指令
MUX4Sel[1:0]	O	MUX4 控制信号
MUX5Sel[1:0]	O	MUX5 控制信号
MUX9Sel[1:0]	O	MUX9 控制信号
MUX10Sel[1:0]	O	MUX10 控制信号

3.22 STALL

3.22.1 基本描述

阻塞模块，当指令为装载指令时阻塞流水线，将 PCWr 和 IF_IDWr 置为 0，并将 ID/EX 流水线寄存器的写信号都置为 0，这相当于插入了一条空指令

3.22.2 阻塞情况分析

- 装载指令后紧跟着一个需要读取它的结果的指令，则至少需要阻塞一个周期
  例如：
  lw $2, 20($1)
  and $4, $2, $5
- 当前指令为分支指令或者为跳转返回指令（JR 和 JALR）（这些指令类型

定义为 BJ 型），且需要读取的寄存器的值与上一条指令相关，则需要阻塞一个周期，若上一条指令为 load 指令，则需要阻塞两个周期

例 1：

```
sub $1, $2, $3
beq $1, $2, 7
```

例 2：

```
lw $2, 20($1)
beq $1, $2, 7
```

- 当前指令为分支指令或者为跳转返回指令，且之前相隔一条的指令为 load 指令并和 load 指令存在数据相关，则需要阻塞一个周期

  例如：

```
lw $2, 20($1)
add $3, $4, $5
beq $1, $2, 7
```

编码时，先通过判断情况 1，来决定是否阻塞一个周期，再判断情况 3，再决定是否阻塞一个周期，情况 1 阻塞了一个周期后就变成了情况 3，这样同时解决了 load 指令和 BJ 型指令相邻时要连续阻塞两个周期的问题，剩下那种情况就是当前指令为 BJ 型指令且上一条指令不是 load 指令。

3.22.3 模块接口

信号名	方向	描述
ID_EX_RT[4:0]	I	ID/EX 寄存器的 rt 字段
EX_MEM_RT[4:0]	I	EX/MEM 寄存器的 rt 字段
IF_ID_RS[4:0]	I	IF/ID 寄存器的 rs 字段
IF_ID_RT[4:0]	I	IF/ID 寄存器的 rt 字段
ID_EX_DMRd	I	ID/EX 寄存器的 DMRd 信号
EX_MEM_DMRd	I	EX/MEM 寄存器的 DMRd 信号
ID_EX_RFWr	I	ID/EX 寄存器的 RFWr 信号
BJOp	I	1：该指令是一条分支或者返回（JR、JALR）指令 0：该指令是其余指令
PCWr	O	PC 写使能信号 Stall：0 Not stall：1
IF_IDWr	O	IF/ID 寄存器写使能信号 Stall：0 Not stall：1
MUX9Sel	O	Stall：1 Not stall：0

3.23 CTRL

3.23.1 基本描述

ctrl 主要功能是产生各种控制信号

3.23.2 模块接口

信号名	方向	描述
clk	I	时钟信号
rst	I	复位信号
OP[5:0]	I	指令格式中的 OPCODE 域
Funct[5:0]	I	指令格式中的 FUNCT 域
rt[4:0]	I	指令格式中的 RT 域
Zero	I	ALU1 输出信号 0：ALU 两操作数不等 1：ALU 两操作数相等
Less	I	ALU1 输出信号 0：A ≥ B 1：A < B
CMPOut1	I	比较器输出信号（主要用于分支指令） 0：GPR[rs] = GPR[rt] 1：GPR[rs] != GPR[rt]
CMPOut2[1:0]	I	比较器输出信号（主要用于和 0 比较的分支指令） 00：GPR[rs] = 0 01：GPR[rs] > 0 10：GPR[rs] < 0
MUX1Sel[1:0]	O	MUX1 的片选信号 00： Instr[20:16] 01： Instr[15:11] 10： 5’h1f（$31 号寄存器）
MUX2Sel[2:0]	O	MUX2 的片选信号 000： RHLOut 001： Imm32 010： ALU1Out 011： PC+4 100： DMOut
MUX3Sel	O	MUX3 的片选信号 0： MUX7Out 1： Imm32
MUX8Sel[1:0]	O	MUX8 的片选信号

		00： RHLOut 01： Imm32 10： ALU1Out 11： PC+4
RFWr	O	RF 的写使能信号 0：禁止写 1：允许写
RHLWr	O	0：禁止写 1：允许写
DMWr	O	数据存储器写使能信号 0：禁止写 1：允许写
DMRd	O	0：该指令不读取 DM 1：该指令读取 DM
NPCOp[1:0]	O	NPC 的片选信号 00：NPC = PC + 4 01：NPC = PC + 4 + offset||00 10：NPC = PC[31:28]||imm[25:0]||00 11：NPC = ret_addr
EXTOp[1:0]	O	数据扩展模式选择信号 00：零扩展 01：符号扩展 10：将立即数扩展至高位
ALU1Op[3:0]	O	传送给 ALU1 的运算控制信号。 0000：加法 0001：减法 0010：或 0011：与 0100：或非 0101：异或 0110：逻辑左移/算术左移 0111：逻辑右移 1000：算术右移 1001：有符号比较 1010：无符号比较
ALU1Sel	O	针对移位信号： 0：移位量为 A[4:0] 1：移位量为 shamt
ALU2Op[1:0]	O	传送给 ALU2 的运算控制信号 00：无符号乘法 01：有符号乘法 10：无符号除法 11：有符号除法
RHLSel_Wr[1:0]	O	00：写 LO（低 32 位）

		01：写 HI（高 32 位） 10：写 HI 和 LO
RHLSel_Rd	O	0：读 LO（低 32 位） 1：读 HI（高 32 位）
DMSel[2:0]	O	000：SB 001：SH 010：SW 011：LBU 100：LB 101：LHU 110：LH 111：LW
B_JOp	O	1：该指令是一条分支或者返回（JR、JALR）指令 0：该指令是其余指令

3.23 IF_ID

3.23.1 基本描述

流水线寄存器，将取指令和指令译码阶段分开

3.23.2 模块接口

信号名	方向	描述
PC[31:0] （IF_ID_Out[63:32]）	I	存储指令地址
Instr[31:0] (IF_ID_Out[31:0])	I	指令存储器中的内容
IF_IDWr	I	1：正常工作 0：流水线阻塞
clk	I	时钟信号，上升沿有效
IF_FLUSH	I	1：将 IF/ID 流水线寄存器的指令字段置为 0 0：不进行操作
out[63:0]	O	pc 值和指令存储器中的内容

3.24 ID_EX

3.24.1 基本描述

流水线寄存器，将指令译码阶段和执行阶段分开

3.24.2 模块接口

信号名	方向	描述
PC[31:0] (ID_EX_Out[201:170])	I	存储指令地址
ALU1Op[3:0] (ID_EX_Out[169:166])	I	传送给 ALU1 的运算控制信号，详见上面几节内容
ALU1Sel (ID_EX_Out[160])	I	针对移位信号，详见上面几节内容
ALU2Op[1:0] (ID_EX_Out[165:164])	I	传送给 ALU2 的运算控制信号，详见上面几节内容
clk	I	时钟信号，上升沿有效
rst	I	复位信号，上升沿时将写信号都置为 0
MUX1Sel[1:0] (ID_EX_Out[163:162])	I	MUX1 的片选信号 00： Instr[20:16] 01： Instr[15:11] 10： 5'h1f（$31 号寄存器）
MUX3Sel (ID_EX_Out[161])	I	MUX3 的片选信号 0： MUX7Out 1： Imm32
DMWr (ID_EX_Out[159])	I	数据存储器写使能信号 0：禁止写 1：允许写
DMRd (ID_EX_Out[155])	I	0：该指令不读取 DM 1：该指令读取 DM
DMSel[2:0] (ID_EX_Out[158:156])	I	000：SB 001：SH 010：SW 011：LBU 100：LB 101：LHU 110：LH 111：LW
RFWr (ID_EX_Out[154])	I	RF 的写使能信号 0：禁止写

		1：允许写
RHLWr (ID_EX_Out[153])	I	RHL 的写使能信号 0：禁止写 1：允许写
RHLSel_Wr[1:0] (ID_EX_Out[152:151])	I	00：写 LO（低 32 位） 01：写 HI（高 32 位） 10：写 HI 和 LO
RHLSel_Rd (ID_EX_Out[202])	I	0：读 LO（低 32 位） 1：读 HI（高 32 位）
MUX2Sel[2:0] (ID_EX_Out[150:148])	I	MUX2 的片选信号 000：ALU1OUT 001：Imm32 010：RHL_OUT 011：PC+4 100：DM_OUT
RHLOut[31:0] (ID_EX_Out[147:116])	I	RHL 寄存器的输出
GPR_RS[31:0] (ID_EX_Out[115:84])	I	GPR[rs]的值
GPR_RT[31:0] (ID_EX_Out[83:52])	I	GPR[rt]的值
RS[4:0] (ID_EX_Out[51:47])	I	寄存器 rs 的地址
RT[4:0] (ID_EX_Out[46:42])	I	寄存器 rt 的地址
RD[4:0] (ID_EX_Out[41:37])	I	寄存器 rd 的地址
Imm32[31:0] (ID_EX_Out[36:5])	I	Instr[15:0]经过 EXT 扩展后的结果
shamt[4:0]	I	Instr[10:6]，由立即数指定的移位量
out[201:0]	O	除时钟信号和复位信号的输入端口

3.25 EX/MEM

3.25.1 基本描述

流水线寄存器，将执行阶段和存储器访问阶段分开

3.25.2 模块接口

信号名	方向	描述
PC[31:0] (EX_MEM_Out[240:209])	I	存储指令地址
clk	I	时钟信号，上升沿有效
rst	I	复位信号，上升沿时将写信号都置为 0
DMWr (EX_MEM_Out[208])	I	数据存储器写使能信号 0：禁止写 1：允许写
DMRd (EX_MEM_Out[204])	I	0：该指令不读取 DM 1：该指令读取 DM
DMSel[2:0] (EX_MEM_Out[207:205])	I	000：SB 001：SH 010：SW 011：LBU 100：LB 101：LHU 110：LH 111：LW
RFWr (EX_MEM_Out[203])	I	RF 的写使能信号 0：禁止写 1：允许写
RHLWr (EX_MEM_Out[202])	I	RHL 的写使能信号 0：禁止写 1：允许写
RHLSel_Wr[1:0] (EX_MEM_Out[201:200])	I	00：写 LO（低 32 位） 01：写 HI（高 32 位） 10：写 HI 和 LO
MUX2Sel[2:0] (EX_MEM_Out[199:197])	I	MUX2 的片选信号 000：ALU1OUT 001：Imm32 010：RHL_OUT 011：PC+4 100：DM_OUT
ALU2Out[63:0] (EX_MEM_Out[196:193])	I	ALU2（乘除法）的输出
ALU1Out[31:0] (EX_MEM_Out[68:37])	I	ALU1 的输出
RHLOut[31:0] (EX_MEM_Out[132:101])	I	RHL 寄存器的输出
GPR_RS[31:0] (EX_MEM_Out[100:69])	I	GPR[rs]的值

GPR_RT[31:0] (EX_MEM_Out[36:5])	I	GPR[rt]的值
Imm32[31:0] (EX_MEM_Out[272:241])	I	32 位立即数
RD[4:0] (EX_MEM_Out[4:0])	I	寄存器 rd 的地址
out[272:0]	O	除时钟信号和复位信号的输入端口

3.26 MEM/WB

3.26.1 基本描述

流水线寄存器，将存储器访问阶段和数据写回寄存器阶段分开

3.26.2 模块接口

信号名	方向	描述
PC[31:0] (MEM_WB_Out[267:236])	I	存储指令地址
clk	I	时钟信号，上升沿有效
rst	I	复位信号，上升沿时将写信号都置为 0
RFWr (MEM_WB_Out[235])	I	RF 的写使能信号 0：禁止写 1：允许写
RHLWr (MEM_WB_Out[234])	I	RHL 的写使能信号 0：禁止写 1：允许写
RHLSel_Wr[1:0]	I	00：写 LO（低 32 位） 01：写 HI（高 32 位） 10：写 HI 和 LO
MUX2Sel[2:0] (MEM_WB_Out[231:229])	I	MUX2 的片选信号 000：ALU1OUT 001：Imm32 010：RHL_OUT 011：PC+4 100：DM_OUT
ALU2Out[63:0] (MEM_WB_Out[228:165])	I	ALU2（乘除法）的输出
ALU1Out[31:0] (MEM_WB_Out[68:37])	I	ALU1 的输出
RHLOut[31:0]	I	RHL 寄存器的输出

(MEM_WB_Out[164:133])		
GPR_RS[31:0] (MEM_WB_Out[132:101])	I	GPR[rs]的值
DMOut[31:0] (MEM_WB_Out[100:69])	I	数据存储器的输出
Imm32[31:0] (MEM_WB_Out[36:5])	I	16 位立即数经扩展后的数据
RD[4:0] (MEM_WB_Out[4:0])	I	寄存器 rd 的地址
out[267:0]	O	除时钟信号和复位信号的输入端口

四、 对编码的说明

五、 功能测试

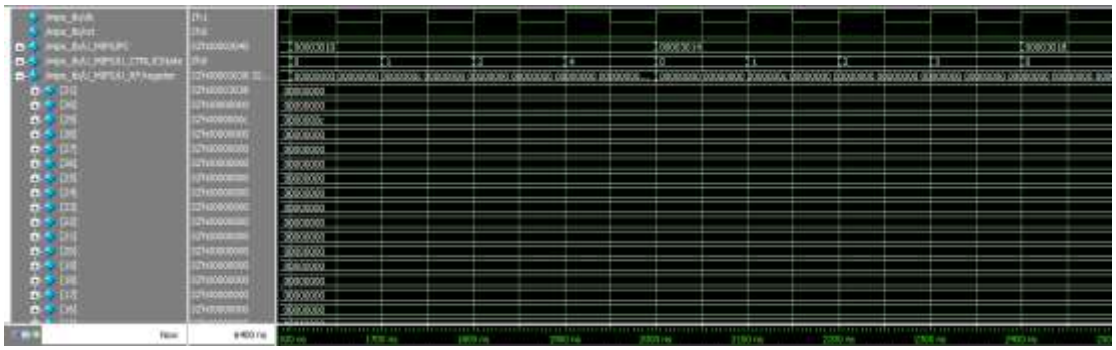
5.1.开发思路

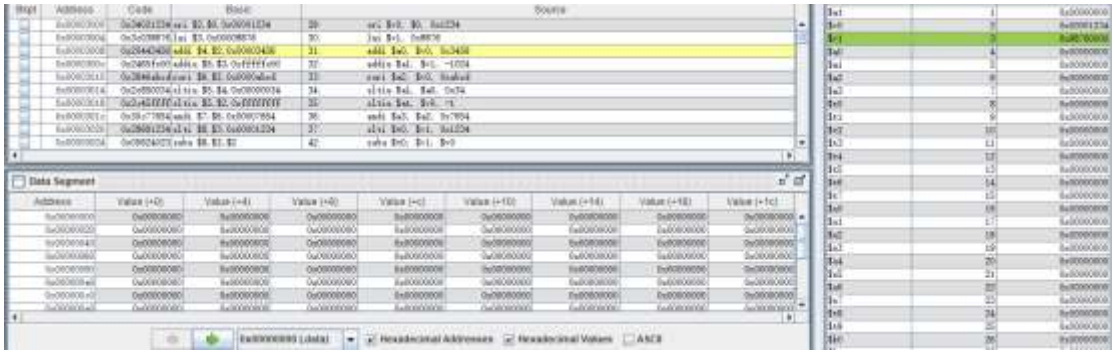
写出包含 57 条（本次设计暂时只实现了 53 条）指令的汇编代码，通过 MARS 软件将其转化成机器码并保存为“code.txt”，在 tb 部分将其读入到指令寄存器。利用 MARS 软件一步步模拟指令执行的结果，并和波形图进行相应对照

5.2. Testbench 说明

```
1  `timescale 1ns/1ns
2  module mips_tb();
3
4     reg clk, rst;
5
6     mips U_MIPS(
7        .clk(clk), .rst(rst)
8     );
9
10    initial begin
11       $readmemh( "code.txt" , U_MIPS.U_IM.imem ) ;
12       $monitor("PC = 0x%8X, IR = 0x%8X", U_MIPS.PC, U_MIPS.IR );
13       clk = 1 ;
14       rst = 0 ;
15       #5 ;
16       rst = 1 ;
17       #20 ;
18       rst = 0 ;
19    end
20
21    always
22        #(50) clk = ~clk;
23
24 endmodule
25
```

六、 仿真结果





由于指令过多，不一一贴出相应的测试段，所以指令均被测试且能正确执行