MIPS CPU 设计文档

杜家驹[1] 袁星驰[2] 王硕[3]

2017年1月

[1]电子邮件：dujj14@mails.tsinghua.edu.cn，学号：2014010257
[2]电子邮件：yuanxc13@mails.tsinghua.edu.cn，学号：2013011665
[3]电子邮件：w-s14@mails.tsinghua.edu.cn，学号：2014012276

目录

1 总体设计

1.1 流水线

本实验的CPU采用5级流水线结构，5个阶段分别为取指，译码，运算，访存，写回。 在同一时刻，最多可以有5条指令在流水线上执行。CPU的流水线部分由以下9个部分组成。

PC 表示当前要取的指令的地址。正常情况下，每个时钟周期地址增加4。 同时PC还要正确的对跳转和异常作出正确的反应。如果PC指向的地址需要TLB映射并且TLB中找不到对应的虚拟地址，则报告发生tlb miss on load异常。

IF/ID 取指阶段和译码阶段之间的流水线寄存器。

ID 译码。根据指令，判断指令的类型，决定是否要读寄存器，是否要 读/写内存，指令的特权等级。并检测syscall异常，reserved instruction异常。判断跳转指令是否进行跳转。然后把所有后续需要的信息传给下一阶段。

ID/EX 译码阶段和运算阶段之间的流水线寄存器。

EX 运算。计算算术运算指令的结果和访存指令的内存地址。把后续需要的 信息传给下一阶段。

EX/MEM 运算阶段和访存阶段之间的流水线寄存器。

MEM 访存。检测 interrupt/tlb miss on load/tlb miss on store/address error on load/address error on store/debug异常。

MEM/WB 访存阶段和写回阶段之间的流水线寄存器。

WB 写回。写回的寄存器目标包括通用寄存器/hi/lo/cp0。

1.2 流水线冲突的处理

数据冲突 我们采用数据旁路的方法来解决。读寄存器的操作全部发生在译码阶段。 把运算阶段和访存阶段的指令要修改的寄存器和要写入的值传给译码阶段，就可以解决冲 突。对于译码阶段和写回指令的冲突，我们在寄存器模块解决，即一个寄存器如果要同时 读或写，读出的值就是要写入的值。但如果一条load指令读取的值在紧邻的指令中要用到 ，就不能用数据旁路的方法来解决。这时让流水线暂停一周期。以上的方法针对的寄存器 包括通用寄存器，Hi/Lo寄存器和cp0寄存器。

结构冲突 在访存和取指同时发生时，暂停一周期。

控制冲突 在跳转指令的译码阶段得出是否跳转，并传给PC。

1.3 流水线暂停

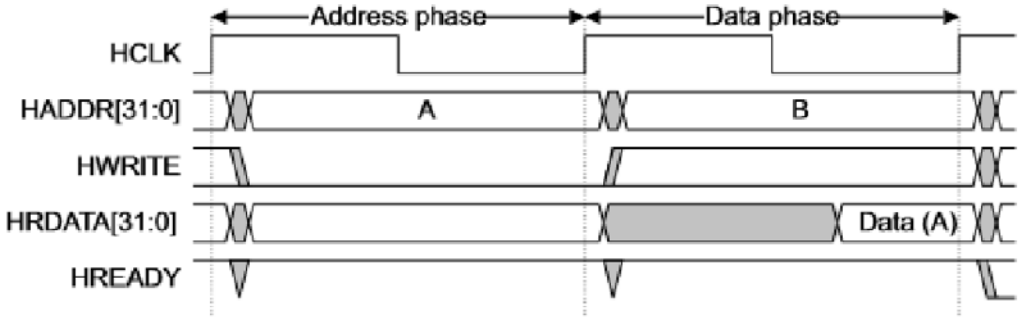
在需要流水线暂停时，我们保持PC模块，IF/ID模块的值不变，ID/EX模块的输出置为nop 指令的输出，EX/MEM模块和MEM/WB模块的输出和流水线正常运行时的输出相同。这样等效 与在EX阶段插入了一个空指令。为了使异常处理能够正确处理插入的空指令，这一空指令的 PC应该等于译码阶段的PC，是否在延迟槽中也应该和译码阶段的指令一致。为实现这一功能， 我们增加了一个新的模块control。

1.4 异常处理

为了进行异常处理，在流水线的各个阶段都要收集必要的异常信息，然后统一在MEM阶段 汇总到control模块。control模块根据收集到的信息决定是否要引发异常。如果要引发异常，则发出flush信号。除MEM/WB和寄存器模块之外，其它的流水线寄存器都要清空（输出变为空指 令的输出）。同时为了防止在刚刚flush流水线后又发生异常时找不到要返回的地址，需要把流水线前4阶段的PC均设置为异常处理程序的入口地址。

1.5 总线

为了设计的简洁，我们实现了AHB-Lite总线协议。其主要原理如图所示。



图中共有两个时钟周期，在第一个时钟周期开始时，CPU把要读写的地址以及是否要读写放到总线上。在第二个时钟周期开始时，外设得到要读写的地址。在第三个时钟周期开始时，CPU得到读写的数据。

总线的上的信号有：

Hclock：总线时钟，25M

Hreset：复位信号

Hsize：数据大小，1表示Word，0表示Byte

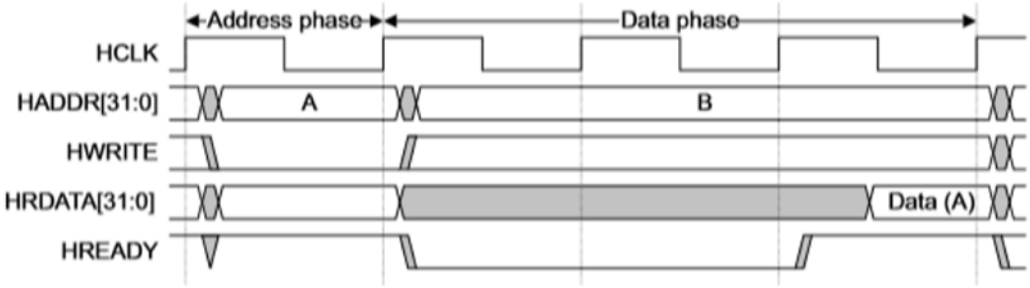
Hwrite：是否进行写操作

Hwritedata：需要写入的数据

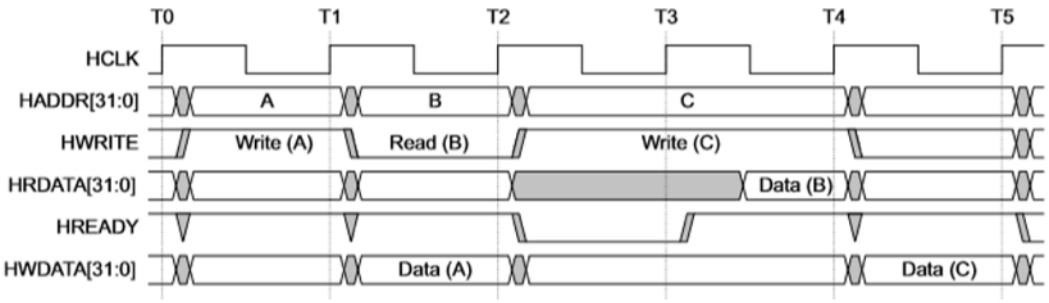
Hreaddata：读到的数据

Hresponse：总线的读写状态

Hready：总线是否已经读写完成



对于慢速设备，在读写没有完成时，可以把Hready信号置为低，这样CPU就会保持 状态不变，直到Hready回到高为止。读写完成时把Hready信号置为高。



上述读写方式每次需要两个总线周期。我们采用了一种流水线的方式来让CPU和总线同步。在上图中，在第一个时钟周期CPU发出写A的命令。在第二个时钟周期外设写A，同时CPU又可以发出读B的命令，这样就实现了CPU和总线的频率同步。

1.6 外设

ROM 直接用片上的寄存器实现。

SRAM 每次读需要一个周期，每次写需要三个周期。在写Byte时，由于只能以32bit为单位读写，所以需要先读取，再写回。 串口和键盘共用一个缓冲区，每次接收到一个字符后就发出串口中断。在写串口并且没完成时需要把Hready置为0。

VGA 仅实现写Byte的功能。

Flash 实现了读取的功能，每次读取需要较长的实现，需要妥善处理Hready信号。

1.7 GDB

为了实现调试功能，我们增加了两个cp0寄存器。当PC等于这两个寄存器之一时触发23号调试异常。读写这两个寄存器可以通过mfc0和mtc0来实现。操作系统增加了一个新的系统调用设置断点，编号为45号。查看寄存器的值是通过查看异常栈帧来实现的。设断点直接使用mtc0。修改寄存器内容可通过修改异常栈帧来实现。单步运行等效于把断点设置为下一条要执行的指令的地址。

1.8 自检工具

为了能够检测SRAM和Flash的正确性，以及把操作系统写入Flash，需要完成一个自检工具，功能为通过串口读写SRAM和Flash。自检工具由硬件和软件两部分组成。硬件部分为一个简单的状态机，不断轮询串口处是否有数据输入，有输入时读入数据并进行相应的操作。软件部分使用python实现，支持的命令有

```
$writeRom #写ROM
$readRom #读ROM
$writeFlash #写Flash
$eraseFlash #擦除Flash
$readFlash #读Flash
$writeSram #写Sram
$readSram #读Sram
$writeOs #写入OS
```

2 模块设计

2.1 top

top模块是本项目的顶层模块，主要功能是把CPU和外设连接在一起，并提供25M时钟。

表 1: top

输入/输出	信号类型	名称	描述
input	wire	clock	50M时钟
input	wire	reset	复位0有效
output	wire	Ram1OE	SRAM读使能0有效
output	wire	Ram1WE	SRAM写使能0有效
output	wire	Ram1EN	SRAM使能0有效
output	wire[19:0]	Ram1Address	SRAM的地址
inout	wire[31:0]	Ram1data	SRAM的数据
input	wire	RxD	串口接收信号
output	wire	TxD	串口发送信号
output	wire	CE0	Flash使能0有效
output	wire	BYTE	Flash操作模式，始终为1（字模式）
output	wire	VPEN	Flash写保护，始终为1
output	wire	RP	Flash工作，始终为1
output	wire	OE	Flash读使能，0有效
output	wire	WE	Flash写使能，0有效
output	wire[22:0]	flashAddress	Flash的地址
inout	wire[15:0]	flashdata	Flash的数据
output	wire	vs	VGA场同步信号
output	wire	hs	VGA行同步信号
output	wire[2:0]	r	VGA红信号
output	wire[2:0]	g	VGA绿信号
output	wire[2:0]	b	VGA蓝信号
input	wire	ps2clk	PS/2时钟
input	wire	ps2data	PS/2数据

2.2 cpu

CPU的顶层模块

表 2: cpu

输入/输出	信号类型	名称	描述
input	wire	clock	时钟
input	wire	reset	复位0有效
input	wire[31:0]	CPURomData	从总线上获取的数据
input	wire	CPUBusResponse	总线的回复
input	wire	CPUBusReady	总线是否已完成工作
input	wire	CPUSerialInterrupt	是否发生串口中断
output	reg[31:0]	CPUAddress	需要读写的地址
output	reg	CPUWriteEnable	是否进行写
output	reg	CPUDataSize	读写的单位是否为word
output	reg[31:0]	CPUWriteData	需要写的数据

2.3 PCRegister

PC寄存器在每个周期增加4。在flush信号为1时，值设置为flush的目标。在需要跳转时，值设置为跳转的目标。该模块还需要把取指是否发生TLB缺失这一信息传递给下一阶段。

表 3: PCRegister

输入/输出	信号类型	名称	描述
input	wire	clock	时钟
input	wire	reset	复位
input	wire	ready	外设已就绪
input	wire	PauseSignal	暂停信号
input	wire	BranchFlag	是否发生转移
input	wire[31:0]	BranchTarget	转移的目标地址
input	wire	PCTLBMiss	取指时是否发生TLB缺失
input	wire	flush	是否需要清空流水线
input	wire[31:0]	flushTarget	清空流水线之后的PC
output	reg[31:0]	PC	PC
output	reg[31:0]	PCPlus4	下一个PC
output	reg	PCTLBMissOut	取指时是否发生TLB缺失

2.4 if_id

表 4: if_id

输入/输出	信号类型	名称	描述
input	wire	clock	时钟
input	wire	reset	复位
input	wire	ready	外设已就绪
input	wire	flush	是否需要清空流水线
input	wire[31:0]	flushTarget	清空流水线之后的PC
input	wire	PauseSignal	暂停信号
input	wire[31:0]	PC	PC
input	wire[31:0]	Instruction	当前指令
input	wire	PCTLBMiss	取指时是否发生TLB缺失
input	wire	IsInDelaySlot	当前指令处于延迟槽中
output	reg[31:0]	IdPC	PC
output	reg[31:0]	IdInstruction	当前指令
output	reg	PCTLBMissOut	取指时是否发生TLB缺失
output	reg	IsInDelaySlotOut	当前指令处于延迟槽中

2.5 id

判断指令的种类，具体类型，读取寄存器操作数和立即数操作数。在此阶段需要解决除load相关之外的数据冲突。在译码之后，对于跳转指令，判断是否需要跳转。然后把要写回的寄存器编号传给下一阶段。此外，还需要检测syscall，reserved instructions异常，并把相关的信息传给下一阶段。

表 5: id

输入/输出	信号类型	名称	描述
input	wire	reset	复位
input	wire[31:0]	IdPC	PC
input	wire[31:0]	IdInstruction	当前指令
input	wire[31:0]	RegisterData1	读到的第一个寄存器数据
input	wire[31:0]	RegisterData2	读到的第二个寄存器数据
input	wire[31:0]	CPData	读到的CP0的数据
input	wire	IsInDelaySlotIn	当前指令处于延迟槽中

表 5：id

输入/输出	信号类型	名称	描述
input	wire[31:0]	hi	hi寄存器
input	wire[31:0]	lo	lo寄存器
input	wire	ExWriteRegisterIn	EX阶段是否要写寄存器
input	wire[31:0]	ExWriteDataIn	EX阶段要写入寄存器的数据
input	wire[4:0]	ExWriteAddressIn	EX阶段要写入寄存器的编号
input	wire	ExWriteHiIn	EX阶段是否要写hi
input	wire[31:0]	ExWriteHiDataIn	EX阶段要写入hi的数据
input	wire	ExWriteLoIn	EX阶段是否要写lo
input	wire[31:0]	ExWriteLoDataIn	EX阶段要写入lo的数据
input	wire	ExWriteCPIn	EX阶段是否要写CP0
input	wire[4:0]	ExWriteCPAddress	EX阶段要写入CP0的数据
input	wire[31:0]	ExWriteCPData	EX阶段要写入CP0的编号
input	wire	MemWriteRegisterIn	MEM阶段是否要写寄存器
input	wire[31:0]	MemWriteDataIn	MEM阶段要写入寄存器的数据
input	wire[4:0]	MemWriteAddressIn	MEM阶段要写入寄存器的编号
input	wire	MemWriteHiIn	MEM阶段是否要写hi
input	wire[31:0]	MemWriteHiDataIn	MEM阶段要写入hi的数据
input	wire	MemWriteLoIn	MEM阶段是否要写lo
input	wire[31:0]	MemWriteLoDataIn	MEM阶段要写入lo的数据
input	wire	MemWriteCPIn	MEM阶段是否要写CP0
input	wire[4:0]	MemWriteCPAddress	MEM阶段要写入CP0的数据
input	wire[31:0]	MemWriteCPData	MEM阶段要写入CP0的编号
input	wire	ExIsLoad	EX阶段的指令是load指令
input	wire	PCTLBMiss	取指时是否发生TLB缺失
output	reg[31:0]	currentPC	当前PC
output	reg	tlbwi	指令是tlbwi
output	reg	syscall	指令是syscall
output	reg	eret	指令是eret
output	reg	privilege	指令是特权指令
output	reg	ValidInstruction	指令是合法的
output	reg	RegisterReadEnable1	需要读第一个寄存器
output	reg	RegisterReadEnable2	需要读第二个寄存器
output	reg[4:0]	RegisterAddress1	第一个寄存器的地址
output	reg[4:0]	RegisterAddress2	第二个寄存器的地址
output	reg[4:0]	CPAddress	CP0寄存器的地址

表 5：id

输入/输出	信号类型	名称	描述
output	reg[4:0]	WriteCPAddress	需要写的CP0寄存器地址
output	reg[7:0]	ALUOperation	ALU运算子类型
output	reg[2:0]	ALUSel	ALU运算类型
output	reg[31:0]	Register1	读到的第一个寄存器的值
output	reg[31:0]	Register2	读到的第二个寄存器的值
output	reg[4:0]	WriteAddressOut	需要写的寄存器地址
output	reg	WriteRegisterOut	是否需要写寄存器
output	reg	WriteHiOut	是否需要写hi
output	reg	WriteLoOut	是否需要写lo
output	reg	NextInstructionInDelaySlot	下一条指令在延迟槽中
output	reg	BranchFlag	是否需要跳转
output	reg[31:0]	BranchTarget	跳转的目标地址
output	reg[31:0]	LinkAddress	跳转的连接地址
output	reg	PauseRequest	是否需要暂停
output	reg[15:0]	IdInstructionOut	当前指令
output	reg	IsInDelaySlotOut	当前指令在延迟槽中
output	reg	PCTLBMissOut	当前指令取指时发生了TLB缺失

2.6 id_ex

表 6: id_ex

输入/输出	信号类型	名称	描述
input	wire	clock	时钟
input	wire	reset	复位
input	wire	ready	外设已就绪
input	wire	flush	是否需要清除流水线
input	wire[31:0]	flushTarget	清除流水线后的PC
input	wire	PauseSignal	暂停信号
input	wire[7:0]	IdALUOperation	ALU子运算类型
input	wire[2:0]	IdALUSel	ALU运算类型
input	wire[31:0]	IdRegister1	第一个寄存器的值

表 6：id_ex

输入/输出	信号类型	名称	描述
input	wire[31:0]	IdRegister2	第二个寄存器的值
input	wire[4:0]	IdWriteAddress	要写的寄存器的地址
input	wire	IdWriteRegister	是否要写寄存器
input	wire	IdWriteHi	是否要写hi
input	wire	IdWriteLo	是否要写lo
input	wire[4:0]	IdWriteCPAddress	要写的CP0寄存器的地址
input	wire[31:0]	IdLinkAddress	跳转指令链接的地址
input	wire	IdIsInDelaySlot	当前指令在延迟槽中
input	wire	Idtlbwi	指令是tlbwi
input	wire	Idsyscall	指令是syscall
input	wire	Ideret	指令是eret
input	wire	Idprivilege	指令是特权指令
input	wire	IdValidInstruction	指令是合法的
input	wire[15:0]	IdInstruction	当前指令
input	wire	PCTLBMiss	当前指令取指时发生TLB缺失
input	wire[31:0]	IdPC	当前PC
output	reg[7:0]	ExALUOperation	ALU子运算类型
output	reg[2:0]	ExALUSel	ALU运算类型
output	reg[31:0]	ExRegister1	第一个寄存器的值
output	reg[31:0]	ExRegister2	第二个寄存器的值
output	reg[4:0]	ExWriteAddress	要写的寄存器的地址
output	reg	ExWriteRegister	是否要写寄存器
output	reg	ExWriteHi	是否要写hi
output	reg	ExWriteLo	是否要写lo
output	reg[4:0]	ExWriteCPAddress	要写的CP0寄存器的地址
output	reg[31:0]	ExLinkAddress	跳转指令链接的地址
output	reg	ExIsInDelaySlot	当前指令在延迟槽中
output	reg	Extlbwi	指令是tlbwi
output	reg	Exsyscall	指令是syscall
output	reg	Exeret	指令是eret
output	reg	Exprivilege	指令是特权指令
output	reg	ExValidInstruction	指令是合法的
output	reg[15:0]	ExInstruction	当前指令
output	reg	PCTLBMissOut	当前指令取指时发生TLB缺失
output	reg[31:0]	ExPC	当前PC

2.7 ex

计算运算的结果。对于访存指令，计算结果为内存地址。

表 7: ex

输入/输出	信号类型	名称	描述
input	wire	reset	复位
input	wire[7:0]	ALUOperation	子运算类型
input	wire[2:0]	ALUSel	运算类型
input	wire[31:0]	Register1	操作数1
input	wire[31:0]	Register2	操作数2
input	wire[4:0]	WriteAddressIn	要写入的寄存器
input	wire	WriteRegisterIn	是否要写寄存器
input	wire	WriteHiIn	是否要写hi
input	wire	WriteLoIn	是否要写lo
input	wire[31:0]	LinkAddress	跳转指令的链接地址
input	wire	IsInDelaySlotIn	当前指令在延迟槽中
input	wire[15:0]	Instruction	当前指令
input	wire[4:0]	WriteCPAddress	要写入的CP0寄存器
input	wire	tlbwi	指令是tlbwi
input	wire	syscall	指令是syscall
input	wire	eret	指令是eret
input	wire	privilege	指令是特权指令
input	wire	ValidInstruction	指令是合法的
input	wire	PCTLBMiss	当前指令取指时发生TLB缺失
input	wire[31:0]	currentPC	当前PC
output	reg[31:0]	WriteDataOut	要写的寄存器数据
output	reg[4:0]	WriteAddressOut	要写入的寄存器
output	reg	WriteRegisterOut	是否要写寄存器
output	reg[31:0]	WriteHiDataOut	要写的hi数据
output	reg	WriteHiOut	是否要写hi
output	reg[31:0]	WriteLoDataOut	要写的lo数据
output	reg	WriteLoOut	是否要写lo
output	reg	IsLoad	当前指令是load指令

表 7：ex

输入/输出	信号类型	名称	描述
output	reg	SignExtend	load的结果是否要做符号扩展
output	reg[31:0]	RAMAddress	内存虚拟地址
output	reg	RAMWriteEnable	是否要写内存
output	reg[31:0]	RAMData	要写入内存的数据
output	reg	RAMDataSize	数据是否以word为单位
output	reg	RAMReadEnable	是否要读内存
output	reg	WriteCPOut	是否要写CP0
output	reg[4:0]	WriteCPAddressOut	要写入CP0的地址
output	reg[31:0]	WriteCPDataOut	要写入CP0的数据
output	reg	IsInDelaySlotOut	当前指令在延迟槽中
output	reg	tlbwiOut	指令是tlbwi
output	reg	syscallOut	指令是syscall
output	reg	eretOut	指令是eret
output	reg	privilegeOut	指令是特权指令
output	reg	ValidInstructionOut	指令是合法的
output	reg	PCTLBMissOut	当前指令取指时发生TLB缺失
output	reg[31:0]	currentPCOut	当前PC
output	reg	ExAddressReadPrivilege	读内存是否需要权限
output	reg	ExAddressWritePrivilege	写内存是否需要权限

2.8 ex_mem

表 8: ex_mem

输入/输出	信号类型	名称	描述
input	wire	clock	时钟
input	wire	reset	复位
input	wire	ready	外设已就绪
input	wire	flush	是否需要清除流水线
input	wire[31:0]	flushTarget	清除流水线之后的PC
input	wire[4:0]	ExWriteAddress	要写入寄存器的地址
input	wire	ExWriteRegister	是否需要写寄存器

表 8：ex_mem

输入/输出	信号类型	名称	描述
input	wire[31:0]	ExWriteData	要写入寄存器的数据
input	wire[31:0]	ExWriteHiData	要写入hi的数据
input	wire[31:0]	ExWriteLoData	要写入lo的数据
input	wire	ExWriteHi	是否要写hi
input	wire	ExWriteLo	是否要写lo
input	wire	ExIsInDelaySlot	当前指令在延迟槽中
input	wire	ExSignExtend	load是否进行符号扩展
input	wire	ExRAMReadEnable	是否读内存
input	wire	ExWriteCP	是否写cp0
input	wire[4:0]	ExWriteCPAddress	写入CP0的地址
input	wire[31:0]	ExWriteCPData	写入CP0的数据
input	wire	Extlbwi	指令是tlbwi
input	wire	Exsyscall	指令是syscall
input	wire	Exeret	指令是eret
input	wire	Exprivilege	指令是特权指令
input	wire	ExValidInstruction	指令是合法的
input	wire	PCTLBMiss	当前指令取指时发生TLB缺失
input	wire	ExReadTLBMiss	发生TLB读缺失
input	wire	ExWriteTLBMiss	发生TLB写缺失
input	wire	ExReadError	发生读错误
input	wire	ExWriteError	发生写错误
input	wire[31:0]	ExPC	当前PC
input	wire	ExAddressReadPrivilege	读操作需要权限
input	wire	ExAddressWritePrivilege	写操作需要权限
input	wire[31:0]	ExBadAddress	错误的地址
output	reg[4:0]	MemWriteAddress	要写入寄存器的地址
output	reg	MemWriteRegister	是否需要写寄存器
output	reg[31:0]	MemWriteData	要写入寄存器的数据
output	reg[31:0]	MemWriteHiData	要写入hi的数据
output	reg[31:0]	MemWriteLoData	要写入lo的数据
output	reg	MemWriteHi	是否要写hi
output	reg	MemWriteLo	是否要写lo
output	reg	MemIsInDelaySlot	当前指令在延迟槽中
output	reg	MemSignExtend	load是否进行符号扩展
output	reg	MemRAMReadEnable	是否读内存

表 8：ex_mem

输入/输出	信号类型	名称	描述
output	reg	MemWriteCP	是否写cp0
output	reg[4:0]	MemWriteCPAddress	写入CP0的地址
output	reg[31:0]	MemWriteCPData	写入CP0的数据
output	reg	Memtlbwi	指令是tlbwi
output	reg	Memsyscall	指令是syscall
output	reg	Memeret	指令是eret
output	reg	Memprivilege	指令是特权指令
output	reg	TLBMissRead	发生TLB读缺失
output	reg	TLBMissWrite	发生TLB写缺失
output	reg	ReadError	发生读错误
output	reg	WriteError	发生写错误
output	reg	MemValidInstruction	指令是合法的
output	reg[31:0]	MemPC	当前PC
output	reg	MemAddressReadPrivilege	读操作需要权限
output	reg	MemAddressWritePrivilege	写操作需要权限
output	reg[31:0]	MemBadAddress	错误的地址

2.9 mem

访存模块。我们把访存模块的大部分控制逻辑移动到了control模块。本模块仅用于传递流水线信息。

表 9: mem

输入/输出	信号类型	名称	描述
input	wire	reset	复位
input	wire[4:0]	WriteAddressIn	要写入的寄存器地址
input	wire	WriteRegisterIn	是否要写寄存器
input	wire[31:0]	WriteDataIn	要写入寄存器的数据
input	wire[31:0]	HiIn	要写入hi的数据
input	wire[31:0]	LoIn	要写入lo的数据
input	wire	WriteHiIn	是否要写hi
input	wire	WriteLoIn	是否要写lo

表 9：mem

输入/输出	信号类型	名称	描述
input	wire	SignExtend	是否进行符号扩展
input	wire	RAMReadEnable	是否要读内存
input	wire[31:0]	RAMData	读内存的结果
input	wire	WriteCP	是否要写CP0
input	wire[4:0]	WriteCPAddress	写入CP0的地址
input	wire[31:0]	WriteCPData	写入CP0的数据
output	reg[4:0]	WriteAddressOut	要写入寄存器的地址
output	reg	WriteRegisterOut	是否要写入寄存器
output	reg[31:0]	WriteDataOut	要写入寄存器的地址
output	reg[31:0]	HiOut	要写入hi的数据
output	reg[31:0]	LoOut	要写入lo的数据
output	reg	WriteHiOut	是否要写hi
output	reg	WriteLoOut	是否要写lo
output	reg	WriteCPOut	是否要写CP0
output	reg[4:0]	WriteCPAddressOut	写入CP0的地址
output	reg[31:0]	WriteCPDataOut	写入CP0的数据

2.10 mem_wb

表 10: mem_wb

输入/输出	信号类型	名称	描述
input	wire	clock	时钟
input	wire	reset	复位
input	wire	ready	外设已就绪
input	wire	flush	需要清空流水线
input	wire[4:0]	MemWriteAddress	要写入寄存器的地址
input	wire	MemWriteRegister	是否要写入寄存器
input	wire[31:0]	MemWriteData	写入寄存器的数据
input	wire[31:0]	MemHi	要写入hi的数据
input	wire[31:0]	MemLo	要写入lo的数据
input	wire	MemWriteHi	是否要写hi

表 10：mem_wb

输入/输出	信号类型	名称	描述
input	wire	MemWriteLo	是否要写lo
input	wire	MemWriteCP	是否要写CP0
input	wire[4:0]	MemWriteCPAddress	写入CP0的地址
input	wire[31:0]	MemWriteCPData	写入CP0的数据
input	wire	MemWriteepc	是否要写epc
input	wire[31:0]	MemWriteepcData	要写入epc的数据
input	wire	MemWritestatus	是否要写status
input	wire[31:0]	MemWritestatusData	要写入status的数据
input	wire	MemWritecause	是否要写cause
input	wire[31:0]	MemWritecauseData	要写入cause的数据
input	wire	MemWritebadaddr	是否要写badaddr
input	wire[31:0]	MemWritebadaddrData	要写入badaddr的数据
output	reg[4:0]	WbWriteAddress	要写入寄存器的地址
output	reg	WbWriteRegister	是否要写入寄存器
output	reg[31:0]	WbWriteData	写入寄存器的数据
output	reg[31:0]	WbHi	要写入hi的数据
output	reg[31:0]	WbLo	要写入lo的数据
output	reg	WbWriteHiOut	是否要写hi
output	reg	WbWriteLoOut	是否要写lo
output	reg	Write0	是否要写CP0的0号寄存器
output	reg	Write2	是否要写CP0的2号寄存器
output	reg	Write3	是否要写CP0的3号寄存器
output	reg	Write8	是否要写CP0的8号寄存器
output	reg	Write10	是否要写CP0的10号寄存器
output	reg	Write11	是否要写CP0的11号寄存器
output	reg	Write12	是否要写CP0的12号寄存器
output	reg	Write13	是否要写CP0的13号寄存器
output	reg	Write14	是否要写CP0的14号寄存器
output	reg	Write15	是否要写CP0的15号寄存器
output	reg	Write18	是否要写CP0的18号寄存器
output	reg	Write19	是否要写CP0的19号寄存器
output	reg[31:0]	Write0Data	要写入CP0的0号寄存器的数据
output	reg[31:0]	Write2Data	要写入CP0的2号寄存器的数据
output	reg[31:0]	Write3Data	要写入CP0的3号寄存器的数据
output	reg[31:0]	Write8Data	要写入CP0的8号寄存器的数据

表 10：mem_wb

输入/输出	信号类型	名称	描述
output	reg[31:0]	Write10Data	要写入CP0的10号寄存器的数据
output	reg[31:0]	Write11Data	要写入CP0的11号寄存器的数据
output	reg[31:0]	Write12Data	要写入CP0的12号寄存器的数据
output	reg[31:0]	Write13Data	要写入CP0的13号寄存器的数据
output	reg[31:0]	Write14Data	要写入CP0的14号寄存器的数据
output	reg[31:0]	Write15Data	要写入CP0的15号寄存器的数据
output	reg[31:0]	Write18Data	要写入CP0的18号寄存器的数据
output	reg[31:0]	Write19Data	要写入CP0的19号寄存器的数据

2.11 hilo_reg

Hi/Lo寄存器。解决了ID/WB的数据冲突。

表 11: hilo_reg

输入/输出	信号类型	名称	描述
input	wire	clock	时钟
input	wire	reset	复位
input	wire	ready	外设已就绪
input	wire	WriteHiEnable	是否要写hi
input	wire	WriteLoEnable	是否要写lo
input	wire[31:0]	HiIn	要写入hi的数据
input	wire[31:0]	LoIn	要写入lo的数据
output	reg[31:0]	HiOut	hi寄存器的数据
output	reg[31:0]	LoOut	lo寄存器的数据

2.12 regfile

通用寄存器。解决了ID/WB的数据冲突。

表 12: regfile

输入/输出	信号类型	名称	描述
input	wire	clock	时钟
input	wire	reset	复位
input	wire	ready	外设已就绪
input	wire	WriteEnable	是否要写寄存器
input	wire[4:0]	WriteAddress	要写入寄存器的地址
input	wire[31:0]	WriteData	要写入寄存器的数据
input	wire	ReadEnable1	是否要读第一个操作数
input	wire[4:0]	ReadAddress1	第一个操作数的地址
output	reg[31:0]	ReadData1	第一个操作数
input	wire	ReadEnable2	是否要读第二个操作数
input	wire[4:0]	ReadAddress2	第二个操作数的地址
output	reg[31:0]	ReadData2	第二个操作数

2.13 cp

cp寄存器。解决了ID/WB的数据冲突。其中的9号寄存器每个工作周期值加1，与11号寄存器的值相同时发出时钟中断。

表 13: cp

输入/输出	信号类型	名称	描述
input	wire	clock	时钟
input	wire	reset	复位
input	wire	ready	外设已就绪
input	wire[4:0]	address	要读的地址
input	wire	write0	是否要写0号寄存器
input	wire	write2	是否要写2号寄存器
input	wire	write3	是否要写3号寄存器
input	wire	write8	是否要写8号寄存器
input	wire	write10	是否要写10号寄存器
input	wire	write11	是否要写11号寄存器
input	wire	write12	是否要写12号寄存器

表 13：cp

输入/输出	信号类型	名称	描述
input	wire	write13	是否要写13号寄存器
input	wire	write14	是否要写14号寄存器
input	wire	write15	是否要写15号寄存器
input	wire	write18	是否要写18号寄存器
input	wire	write19	是否要写19号寄存器
input	wire[31:0]	write0data	要写入0号寄存器的数据
input	wire[31:0]	write2data	要写入2号寄存器的数据
input	wire[31:0]	write3data	要写入3号寄存器的数据
input	wire[31:0]	write8data	要写入8号寄存器的数据
input	wire[31:0]	write10data	要写入10号寄存器的数据
input	wire[31:0]	write11data	要写入11号寄存器的数据
input	wire[31:0]	write12data	要写入12号寄存器的数据
input	wire[31:0]	write13data	要写入13号寄存器的数据
input	wire[31:0]	write14data	要写入14号寄存器的数据
input	wire[31:0]	write15data	要写入15号寄存器的数据
input	wire[31:0]	write18data	要写入18号寄存器的数据
input	wire[31:0]	write19data	要写入19号寄存器的数据
output	reg	clockInterrupt	时钟中断
output	reg[31:0]	value	读取的寄存器的值
output	reg[31:0]	index0Out	0号寄存器的数据
output	reg[31:0]	entryLo02Out	2号寄存器的数据
output	reg[31:0]	entryLo13Out	3号寄存器的数据
output	reg[31:0]	entryHi10Out	10号寄存器的数据
output	reg[31:0]	status12Out	12号寄存器的数据
output	reg[31:0]	cause13Out	13号寄存器的数据
output	reg[31:0]	epc14Out	14号寄存器的数据
output	reg[31:0]	ebase15Out	15号寄存器的数据
output	reg[31:0]	watchLo18Out	18号寄存器的数据
output	reg[31:0]	watchHi19Out	19号寄存器的数据

2.14 control

control模块主要用于cpu流水线的控制。包含的主要功能有：

结构冲突的处理　当一条访存指令执行到访存阶段时，暂停取指一周期。

访存控制　接收各个模块的访存请求，把虚拟地址转换为物理地址，对于不合法的指令给出响应的错误信息。由于总线在一个周期只能进行一次访存，本模块每个周期挑选出一个合法的访存请求进行访存，对于其它的访存请求，暂停流水线的相应阶段。对于总线的访问结果，本模块把结果发送给对应的请求模块。

暂停控制　根据访存情况和各个流水线模块的请求发出暂停信号。

异常处理　根据访存阶段的异常信号决定是否触发异常。如果触发异常，设置需要写到各个寄存器的值。

表 14: control

输入/输出	信号类型	名称	描述
input	wire	reset	复位
input	wire	clock	时钟
input	wire	ready	外设已就绪
input	wire	flushin	是否需要清除流水线
input	wire	Response	外设的回复
input	wire	SerialInterrupt	串口中断
input	wire	clockInterrupt	时钟中断
input	wire	BranchFlag	是否需要跳转
input	wire[31:0]	BranchTarget	跳转的目标地址
input	wire[31:0]	PCPlus4	需要从内存中取的指令位置
input	wire	PauseRequest	是否需要暂停流水线
input	wire[31:0]	RAMAddress	需要从内存中读写的位置
input	wire	RAMWriteEnable	是否写内存
input	wire[31:0]	RAMData	需要写到内存中的位置
input	wire	RAMDataSize	数据的大小
input	wire	RAMReadEnable	是否读内存
input	wire[31:0]	ReadResult	读内存的结果
input	wire[31:0]	TLBPhysicalAddress	经过TLB映射后的地址
input	wire	ValidAddress	地址是否合法
input	wire	isMiss	TLB缺失/权限错误
input	wire[31:0]	currentPC	当前PC

表 14：control

输入/输出	信号类型	名称	描述
input	wire	isInDelaySlot	当前指令在延迟槽中
input	wire[31:0]	excBadAddress	发生地址错误的地址
input	wire[31:0]	cp0status	cp0/status的值
input	wire[31:0]	cp0cause	cp0/cause的值
input	wire[31:0]	cp0epc	cp0/epc的值
input	wire[31:0]	cp0base	cp0/base的值
input	wire[31:0]	cp0watchLo	cp0/watchLo的值
input	wire[31:0]	cp0watchHi	cp0/watchHi的值
input	wire	TLBMissRead	发生TLB读缺失
input	wire	TLBMissWrite	发生TLB写缺失
input	wire	ReadError	读错误
input	wire	WriteError	写错误
input	wire	ValidInstruction	指令不合法
input	wire	syscall	系统调用
input	wire	InstructionPrivilege	指令需要权限
input	wire	AddressReadPrivilege	读操作需要权限
input	wire	AddressWritePrivilege	写操作需要权限
input	wire	eret	eret指令
output	reg	Writeepc	是否要写epc
output	reg[31:0]	WriteepcData	要写入epc的数据
output	reg	Writestatus	是否要写status
output	reg[31:0]	WritestatusData	要写入status的数据
output	reg	Writecause	是否要写cause
output	reg[31:0]	WritecauseData	要写入cause的数据
output	reg	Writebadaddr	是否要写badaddr
output	reg[31:0]	WritebadaddrData	要写入badaddr的数据
output	reg	flush	是否需要清空流水线
output	reg[31:0]	targetAddress	清空流水线之后的执行地址
output	reg[31:0]	TLBVirtualAddress	传给TLB的虚拟地址
output	reg	TLBWriteEnable	是否进行写操作
output	reg[31:0]	Address	传给内存的物理地址
output	reg	WriteEnable	是否进行写
output	reg	DataSize	数据大小
output	reg[31:0]	WriteData	需要写的数据
output	reg	PauseSignal	暂停信号

表 14：control

输入/输出	信号类型	名称	描述
output	reg	ExReadTLBMiss	发生TLB读缺失
output	reg	ExWriteTLBMiss	发生TLB写缺失
output	reg	ExReadError	读错误
output	reg	ExWriteError	写错误
output	reg	PCReadTLBMiss	取指时发生TLB缺失
output	reg[31:0]	InstructionResult	取指的结果
output	reg[31:0]	LoadResult	访存的结果

2.15 tlb

页表的缓存，同时可以设置其中的任意一项

表 15: tlb

输入/输出	信号类型	名称	描述
input	wire	clock	时钟
input	wire	reset	复位
input	wire[31:0]	VirtualAddress	虚拟地址
input	wire	WriteEnable	是否进行写
input	wire	WriteTLB	是否写TLB
input	wire[31:0]	index	写的序号
input	wire[31:0]	entryLo0	与物理页0有关的信息
input	wire[31:0]	entryLo1	与物理页1有关的信息
input	wire[31:0]	entryHi	与虚拟页有关的信息
output	reg	ValidAddress	地址是否合法
output	reg	isMiss	TLB缺失/权限错误
output	reg[31:0]	PhysicalAddress	物理地址

2.16 bus

总线模块。把CPU的访存请求分发到各个外设。

表 16: bus

输入/输出	信号类型	名称	描述
input	wire	Hclock	时钟
input	wire	Hreset	复位
input	wire	Hsize	数据大小
input	wire	Hwrite	是否写数据
input	wire[31:0]	Hwritedata	要写的数据
input	wire[31:0]	Haddress	地址
output	reg[31:0]	Hreaddata	读到的数据
output	reg	Hresponse	读的结果
output	reg	Hready	外设是否就绪
output	wire	Ram1OE	SRAM读使能0有效
output	wire	Ram1WE	SRAM写使能0有效
output	wire	Ram1EN	SRAM使能0有效
output	wire[19:0]	Ram1Address	SRAM的地址
inout	wire[31:0]	Ram1data	SRAM的数据
input	wire	RxD	串口接收信号
output	wire	TxD	串口发送信号
output	wire	break	串口中断
output	wire	CE0	Flash使能0有效
output	wire	BYTE	Flash操作模式，始终为1（字模式）
output	wire	VPEN	Flash写保护，始终为1
output	wire	RP	Flash工作，始终为1
output	wire	OE	Flash读使能，0有效
output	wire	WE	Flash写使能，0有效
output	wire[22:0]	flashAddress	Flash的地址
inout	wire[15:0]	flashdata	Flash的数据
output	wire	vs	VGA场同步信号
output	wire	hs	VGA行同步信号
output	wire[2:0]	r	VGA红信号
output	wire[2:0]	g	VGA绿信号
output	wire[2:0]	b	VGA蓝信号
input	wire	ps2clk	PS/2时钟
input	wire	ps2data	PS/2数据

2.17 rom

表 17: rom

输入/输出	信号类型	名称	描述
input	wire	Hclock	时钟
input	wire	Hreset	复位
input	wire	Hsize	数据大小
input	wire	Hwrite	是否写数据
input	wire[31:0]	Hwritedata	要写的数据
input	wire[8:0]	Haddress	地址
input	wire	Hselect	是否激活外设
input	wire	ready	bus是否就绪
output	reg[31:0]	Hreaddata	读到的数据
output	reg	Hready	读的结果
output	reg	Hresponse	外设是否就绪

2.18 sram

表 18: sram

输入/输出	信号类型	名称	描述
input	wire	Hclock	时钟
input	wire	Hreset	复位
output	reg	Ram1OE	SRAM读使能0有效
output	reg	Ram1WE	SRAM写使能0有效
output	reg	Ram1EN	SRAM使能0有效
output	reg[19:0]	Ram1Address	SRAM的地址
inout	wire[31:0]	Ram1data	SRAM的数据
input	wire	Hselect	是否激活外设
input	wire	Hwrite	是否写数据
input	wire	Hsize	数据的大小
input	wire	ready	bus是否就绪
input	wire[31:0]	Hwritedata	要写的数据
input	wire[21:0]	Haddress	地址

表 18：sram

输入/输出	信号类型	名称	描述
output	reg[31:0]	Hreaddata	读到的数据
output	wire	Hready	外设是否就绪
output	reg	Hresponse	访存的结果

2.19 flash

表 19: flash

输入/输出	信号类型	名称	描述
input	wire	Hclock	时钟
input	wire	Hreset	复位
input	wire[22:0]	Haddress	地址
input	wire	Hselect	是否激活外设
input	wire	ready	bus是否就绪
output	reg	CE0	Flash使能0有效
output	reg	BYTE	Flash操作模式，始终为1（字模式）
output	reg	VPEN	Flash写保护，始终为1
output	reg	RP	Flash工作，始终为1
output	reg	OE	Flash读使能，0有效
output	reg	WE	Flash写使能，0有效
output	reg[22:0]	addr	Flash的地址
inout	wire[15:0]	data	Flash的数据
output	reg[15:0]	Hreaddata	读到的数据
output	reg	Hready	外设是否就绪
output	reg	Hresponse	访存的结果

2.20 serial_port

表 20: serial_port

输入/输出	信号类型	名称	描述
input	wire	Hclock	时钟
input	wire	Hreset	复位
input	wire	ready	bus是否就绪
input	wire[7:0]	Hwritedata	要写的数据
input	wire	Hselect	是否激活外设
input	wire	Hwrite	是否写
input	wire[2:0]	Haddress	读写的地址
output	reg[7:0]	receiveData	读到的数据
output	reg	break	串口中断
output	wire	Hready	外设是否就绪
output	reg	Hresponse	读写的结果
output	wire	TxD	串口接收信号
input	wire	RxD	串口发送信号
input	wire	ps2clk	PS/2时钟
input	wire	ps2data	PS/2数据

2.21 async_transmitter

串口发送模块

表 21: async_transmitter

输入/输出	信号类型	名称	描述
input	wire	clk	时钟
input	wire	TxD_start	是否开始发送
input	wire[7:0]	TxD_data	要发送的数据
output	wire	TxD	发送的数据
output	wire	TxD_busy	是否正在发送

2.22 async_receiver

串口接收模块

表 22: async_receiver

输入/输出	信号类型	名称	描述
input	wire	clk	时钟
input	wire	RxD	收到的数据
output	reg	RxD_data_ready	是否收到数据
output	reg[7:0]	RxD_data	接收数据线

2.23 BaudTickgen

表 23: BaudTickgen

输入/输出	信号类型	名称	描述
input	wire	clk	时钟
input	wire	enable	是否可用
output	wire	tick	产生的串口时钟

2.24 clkChanger

把时钟频率降低一半

表 24: clkChanger

输入/输出	信号类型	名称	描述
input	wire	clk1	高频时钟
output	reg	clk2	低频时钟

2.25 keyboardControl

表 25: keyboardControl

输入/输出	信号类型	名称	描述
input	wire	Hclock	时钟
input	wire	Hreset	复位
input	wire	ps2data	ps2数据
input	wire	ps2clk	ps2时钟
output	wire	Hready	是否有数据到达
output	reg[7:0]	data	到达的数据

2.26 keyboard2ascii

把键盘扫描码转换为ascii码

表 26: keyboard2ascii

输入/输出	信号类型	名称	描述
input	wire	shift_flag	是否按下shift
input	wire[7:0]	key	扫描码
output	wire[7:0]	ascii	ascii码

2.27 ps2

表 27: ps2

输入/输出	信号类型	名称	描述
input	wire	Hclock	时钟
input	wire	Hreset	复位
output	reg[7:0]	receiveData	收到的数据

表 27：ps2

输入/输出	信号类型	名称	描述
output	reg	Hready	是否就绪
input	wire	ps2data	ps2数据
input	wire	ps2clk	ps2时钟

2.28 vga

表 28: vga

输入/输出	信号类型	名称	描述
input	wire	Hclock	时钟
input	wire	Hreset	复位
input	wire	Hselect	是否激活外设
input	wire	Hwrite	是否写
input	wire	Hsize	数据的大小
input	wire	ready	bus是否就绪
input	wire[31:0]	Hwritedata	要写的数据
input	wire[11:0]	Haddress	要写的地址
output	reg[31:0]	Hreaddata	读到的数据
output	reg	Hready	外设是否就绪
output	reg	Hresponse	访存的结果
output	reg	vs	VGA场同步信号
output	reg	hs	VGA行同步信号
output	reg[2:0]	r	VGA红信号
output	reg[2:0]	g	VGA绿信号
output	reg[2:0]	b	VGA蓝信号

2.29 vga_rom

把ascii码转换为字体

表 29: vga_rom

输入/输出	信号类型	名称	描述
input	wire[6:0]	ch	ascii码
input	wire[6:0]	pos	当前需要的像素在字体中的位置
output	reg	mask	当前需要的像素是否为亮