CPU设计文档v0.2

（一）架构设计

1.设计说明

- 复位时入口点：0xBFC0_0000
- 异常处理程序入口点：0xBFC0_0380
- 采用固定地址映射的MMU。
- 处理器永远处于核心态模式，可以执行所有指令，访问32位全地址空间。

（二）支持的指令

算术运算	逻辑运算	移位指令	分支跳转	数据移动	访存	自陷
ADD	AND	SLL	BEQ	MFHI	LB	BREAK
ADDI	ANDI	SLLV	BNE	MFLO	LBU	SYSCALL
ADDU	LUI	SRA	BGEZ	MTHI	LH	ERET
ADDIU	NOR	SRAV	BGTZ	MTLO	LHU	MFC0
SUB	OR	SRL	BLEZ		LW	MTC0
SUBU	ORI	SRLV	BLTZ		SB	
SLT	XOR		BLTZAL		SH	
SLTI	XORI		BGEZAL		SW	
SLTU			J			
SLTIU			JAL			
DIV			JR			
DIVU			JALR			
MULT						
MULTU						

（三）模块概述

1.mycpu_top模块

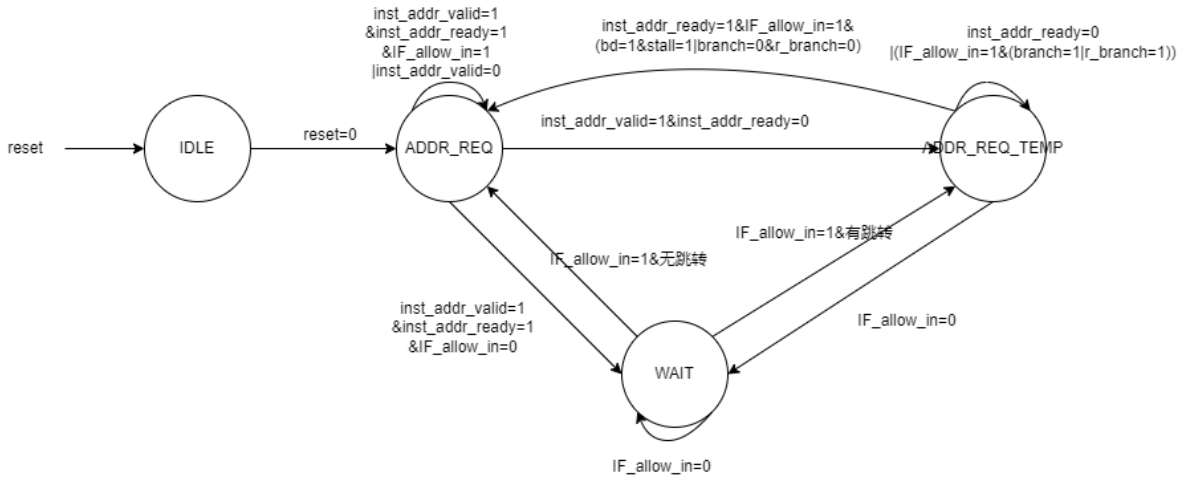
作用是将mycpu封装为AXI接口，内部是取指端与数据端的仲裁逻辑。

2.mycpu模块

端口行为描述

取指端口

- 取指端通过一个状态机控制与外界的交互行为,地址与数据均具有双向握手信号。
- 取指状态机



访存端口

- 访存端口与外界的交互类型为双向握手。

端口定义

名称	位宽	方向	描述
时钟/复位与中断			
clk	1	input	时钟，来自clk_pll的输出时钟
resetn	1	input	复位信号，低电平同步复位
ext_int	6	input	硬件中断，高电平有效
取指端访存接口			
inst_sram_addr	32	output	ram读地址
inst_sram_rdata	32	input	ram读数据
inst_addr_valid	1	output	当前是有效的地址请求
inst_addr_ready	1	input	slave端接收到了读请求
inst_data_valid	1	input	当前读数据有效，一定在地址请求之后
inst_data_ready	1	output	当前可以接收读数据
数据端访存接口			
data_sram_wen	4	output	ram字节写使能信号，高电平有效
data_sram_addr	32	output	ram读写地址，字节寻址
data_sram_wdata	32	output	ram写数据
data_sram_rdata	32	input	ram读数据
data_addr_valid	1	output	地址信号与控制信号在通道中有效（读请求，写请求，写数据共用一个）
data_addr_ready	1	input	slave端接收到了地址信号（读请求，写请求，写数据共用一个）
data_rvalid	1	input	读数据在通道中有效
data_rready	1	output	master端接受读数据
data_bvalid	1	input	写反馈信号
data_bready	1	output	master端可以接收写反馈信号
debug信号，验证平台使用			
debug_wb_pc	32	output	写回级PC
debug_wb_rf_wen	4	output	写回级RF字节写使能
debug_wb_rf_wnum	5	output	写回级写 regfiles 的目的寄存器号
debug_wb_rf_wdata	32	output	写回级写 regfiles 的写数据

3.功能模块

PC

模块功能定义

- 一个存储当前指令地址的32位寄存器

模块端口定义

名称	位宽	方向	描述
clk	1	I	时钟
reset	1	I	复位信号，复位至0xBFC0_0000
en	1	I	使能信号
pc_i	32	I	下一个PC值
pc_o	32	O	当前PC值

IM

模块功能定义

外置，指令存储器

模块端口定义

- (相对于CPU)

名称	位宽	方向	描述
inst_sram_en	1	output	ram使能信号，使能条件：`D_allow_in|req`
inst_sram_wen	4	output	ram字节写使能信号，高电平有效
inst_sram_addr	32	output	ram读写地址，字节寻址
inst_sram_wdata	32	output	ram写数据
inst_sram_rdata	32	input	ram读数据

NPC

模块功能定义

- 次地址计算单元，**处于D级寄存器前面**，驱动PC。
- 非跳转指令时，NPC每次自增4，
    - 对于16位寄存器跳转指令，如果判断条件为真，NPC将Imm26[15:0]左移两位并符号扩展后与当前pc提供的pc值相加，输出。
    - 对于26位跳转指令，**NPC将Imm26左移两位后与pc给出的PC[31:28]拼接后输出。**
    - 对于寄存器跳转指令，PC直接输出rs_reg。

- 对于PC8端口，NPC需要将当前PC值加8后输出。
- 支持跳转到异常处理程序
- 跳转优先级：IntReq>NPCOp

模块规格定义

模块端口定义

名称	位宽	方向	描述
pc	32	I	PC传入的当前PC值
imm26	26	I	26位指令立即数
rs_reg	32	I	要跳转的寄存器中**正确值**,在转发后
npc_op	4	I	4'b0000:pc+4 4'b0001:equal为真则进行16位立即数跳转(beq) 4'b0010:26位立即数跳转(J,JAL) 4'b0011:rs_reg寄存器跳转(jr,jalr) 4'b0100:equal为假则进行16位立即数跳转（bne） 4'b0101:rs寄存器>=0则进行16位立即数跳转(bgez,bgezal) 4'b0110:rs寄存器>0则进行16位立即数跳转(bgtz,bgtzal) 4'b0111:rs寄存器<=0则进行16位立即数跳转(blez) 4'b1000:rs寄存器<0则进行16位立即数跳转(bltz)
equal	1	I	1：相等，0：不相等
int_req	1	I	中断请求，若int_req=1,则跳转到0xBFC0_0380
eret	1	I	若D级为eret指令，则该信号为高电平
epc	32	I	中断恢复入口点
npc	32	O	32位下一个地址输出
branch	1	O	branch为1当且仅当当前D级为跳转指令并且需要跳转

GRF

模块功能定义

- 可以同时读出两个32位数据或者在时钟上升沿且写信号有效情况下写入一个32位数据。
- **支持内部转发**
- 如果A1=A3并且写使能为1，将WD转发至RD1，如果A2=A3并且写使能为1，将WD转发至RD2。与此同时将WD写入目标寄存器。(注意如果目标寄存器为$0则不转发)

模块规格定义

- 32个寄存器
- 同步复位
- **0 号寄存器**的值始终保持为 0。其他寄存器**初始值（复位后）均为 0**，无需专门设置。

模块端口定义

名称	位宽	方向	描述
reset	1	I	同步复位信号
clk	1	I	时钟
en	1	I	写使能信号，高电平有效
raddr1	5	I	5位寄存器地址值，将指向寄存器的数据读出到RD1
raddr2	5	I	5位寄存器地址值，将指向寄存器的数据读出到RD2
waddr	5	I	5位寄存器地址值，为WD指定写入的寄存器
wdata	32	I	要写入的数据
rdata1	32	O	A1数据输出
rdata2	32	O	A2数据输出

EXT

模块功能定义

符号扩展或者无符号扩展

模块规格定义

模块端口定义

名称	位宽	方向	描述
ext_op	1	I	EXTOp=1:符号扩展 EXTOp=0:无符号扩展
ext_in	16	I	位扩展输入16位数据
ext_out	32	O	位扩展输出32位数据

CMP

模块功能定义

比较两个32位数据大小

模块规格定义

模块端口定义

名称	位宽	方向	描述
data1	32	I	第一个数据
data2	32	I	第二个数据
equal	1	O	Equal = C1==C2?1:0;

ALU

模块功能定义

- 非乘除类计算
- 支持输出溢出信号

模块端口定义

名称	位宽	方向	描述
srca	32	I	第一个运算数
srcb	32	I	第二个运算数
shamt	5	I	移位位数
alu_op	5	I	5'b00000:a+b 5'b00001:a-b 5'b00010:a OR b 5'b00011:a NOR b 5'b00100:a XOR b 5'b00101:a AND b 5'b00110:b << 16 (LUI) 5'b00111:b << shamt 5'b01000:逻辑右移：b>>shamt 5'b01001:算术右移：b>>shamt 5'b01010:SLLV 5'b01011:SRLV 5'b01100:SRAV 5'b01101:a<b?1:0(有符号) 5'b01110:a<b?1:0(无符号)
alu_result	32	O	运算结果
overflow	1	O	溢出

MD

模块功能定义

- 乘除计算单元
- 乘法计算花费5周期，除法计算花费10周期
- 自 Start 信号有效后的第 1 个 clock 上升沿开始，乘除部件开始执行运算，同时 Busy 置位为 1。
- 在运算结果保存到 HI 和 LO 后，Busy 位清除为 0。

- 当 Busy 或 Start 信号为 1 时，mfhi、mflo、mthi、mtlo、mult、multu、div、divu 均被阻塞，即被阻塞在 IF/ID。
- 数据写入 HI 或 LO，均只需 1 个 cycle。
- 如果E或M级流水线存在中断，则当前新进入MD的指令无效

模块端口定义

名称	位宽	方向	描述
clk	1	I	时钟
reset	1	I	复位
start	1	I	计算开始信号
md_op	3	I	3'b000:mult 3'b001:multu 3'b010:div 3'b011:divu 3'b100:MTHI 3'b101:MTLO
srca	32	I	第一个源操作数
srcb	32	I	第二个源操作数
int_req	1	I	E或M级流水线存在中断
busy	1	O	置1表示尚在计算中
hi	32	O	结果
lo	32	O	结果

BE

模块功能定义

- 位于E级，根据m_data_addr[1:0]与当前指令产生m_data_byteen
- 根据指令与地址重新生成输入数据
- BE根据外部输入与自身数据判断当前是否需要将四字节写使能置0

模块端口定义

名称	位宽	方向	描述
be_op	2	I	2'b00:none 2'b01:sw 2'b10:sh 2'b11:sb
p_data_addr	32	I	待写入的数据存储器相应地址
p_data_wdata	32	I	未处理的存储器输入数据
int_req	1	I	部分异常中断信号
m_data_byteen	4	O	四字节使能,对应字节写使能
m_data_addr	32	O	待写入的数据存储器相应地址
m_data_wdata	32	O	处理后的存储器输入数据

DM

模块功能定义

外置数据存储器

模块端口定义

名称	位宽	方向	描述
data_sram_en	1	I	ram使能信号，高电平有效（将字节使能或起来）
data_sram_addr	32	I	待写入/读出的数据存储器相应地址
data_sram_wdata	32	I	待写入数据存储器相应数据
data_sram_wen	4	I	四位字节使能
data_sram_rdata	32	O	数据存储器存储的相应数据

DMEXT

模块功能定义

扩展从DM输出的数据

模块端口定义

名称	位宽	方向	描述
dmext_op	3	I	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
m_data_addr	32	I	读出数据的地址
p_data_rdata	32	I	输入 32 位数据
m_data_rdata	32	O	扩展后的32位数据

PIPREG

流水线寄存器跟随前级命名，模块的输入的基本数据通路中的多路选择器放在流水线寄存器之后。

流水线寄存器一共有三部分

- 数据通路中的流水线寄存器：流水数据通路与控制信号，包括PC
- EP中的流水线寄存器：流水异常编码

流水线寄存器的清空与赋值操作

- 异常产生时（Req==1'b1）：清空**所有**流水线寄存器
    - 复位时（reset==1'b1）：清空**所有**流水线寄存器
- 暂停时(Stall==1'b1)：清空除E级PC，BD外的所有E级流水线寄存器，并冻结**所有**D级流水线寄存器
- D级为eret时：如果下一拍不是暂停，则清空除D级pc外的所有D级流水线寄存器

流水线寄存器的操作

- 异常产生或复位时(req|reset=1)：将所有流水级的有效位置为无效
- 暂停时(stall=1)：D级流水线被阻塞(D_ready_go=!stall)
- D级允许进入并且为eret时：只流水PC，其它值置为0
- 注意：bd与pc信号在所有流水级中均有效，与valid无关。
- 前级有异常时流水前级异常，否则流水本级异常。

流水线寄存器的控制逻辑

- 每一级流水线包含一下信号：
    - X_valid：当前级流水线的数据是否有效
    - X_ready_go:当前级流水线对数据的处理已经完成，可以将数据移动到下一个流水级
    - X_allow_in:当前流水级可以接收数据 `X_allow_in  = !X_valid || X_ready_go && X+1_allow_in`
    - X_to_X+1valid:X级有数据需要传递给下一级 `X_to_X+1_valid   = X_valid && X_ready_go`

信号流水

信号/流水级	D	E	M	W
控制信号流水				
alu_op[4:0]		**alu_op_E**		
md_op[2:0]		**md_op_E**		
start		**start_E**		
srcbm_sel		**srcbm_sel_E**		
mdm_sel		**mdm_sel_E**		
be_op[1:0]		be_op_E		
dmext_op[2:0]		dmext_op_E	**dmext_op_M**	
grfwdm_sel[2:0]		grfwdm_sel_E	grfwdm_sel_M	**grfwdm_sel_W**
regwrite		regwrite_E	regwrite_M	**regwrite_W**
a1[4:0]		a1_E	a1_M	a1_W
a2[4:0]		a2_E	a2_M	a2_W
a3[4:0]		a3_E	a3_M	**a3_W**
tnew[1:0]		tnew_E	tnew_M（特殊判断）	
instr_code[INSTR_CODE_WIDTH-1:0]		**instr_code_E**		
cp0_write		cp0_write_E	**cp0_write_M**	
cp0_addr[4:0]		cp0_addr_E	**cp0_addr_M**	
cp0_sel[2:0]		cp0_sel_E	cp0_sel_M	
cache_req		cache_req_E	cache_req_M	
cache_op[4:0]		cache_op_E	cache_op_M	
eret		eret_E	**eret_M**	
mtc0		mtc0_E	mtc0_M	
bd	bd_D	bd_E	bd_M	
数据通路流水				
instr[31:0]	instr_D			
pc8[31:0]	pc8_D	pc8_E	pc8_M	pc8_W
pc[31:0]	pc_D	pc_E	pc_M	pc_W
rd1[31:0]		rd1_E		
rd2[31:0]		rd2_E	rd2_M	
ext32[31:0]		ext32_E		
shamt[4:0]		shamt_E		
alu_result[31:0]			alu_result_M	alu_result_W
md_result[31:0]			md_result_M	md_result_W
m_data_rdata[31:0]				m_data_rdata_W
cp0_out[31:0]				cp0_out_W
异常流水				
exccode	exccode_D	exccode_E	exccode_M	

MUX

	000	001	010	011	100	5	6	7	out	Sel
功能 MUX:										
GRFWDM	alu_result_W_o	md_result_W_o	m_data_rdata_W_o	pc8_W_o	cp0_W_o				grfwdm_o	grfwdm_sel
GRFA3M	instr_D_o[15:11](rd)	instr_D_o[20:16](rt)	0x1F						grfa3m_o	grfa3m_sel
SRCBM	rd2mfe_o	ext32_E_o							srcbm_o	srcbm_sel
MDM	md.hi	md.lo							mdm_o	mdm_sel
转发MUX:										
RD1MFD	grf.rdata1	alu_result_M_o	md_result_M_o	pc8_M_o	pc8_E_o				RD1MFD_O	RD1MFD_Sel
RD2MFD	grf.rdata2	alu_result_M_o	md_result_M_o	pc8_M_o	pc8_E_o				RD2MFD_O	RD2MFD_Sel
RD1MFE	rd1_E_o	grfwdm_o	alu_result_M_o	md_result_M_o	pc8_M_o				RD1MFE_O	RD1MFE_Sel
RD2MFE	rd2_E_o	grfwdm_o	alu_result_M_o	md_result_M_o	pc8_M_o				RD2MFE_O	RD2MFE_Sel
RD2MFM	rd2_M_o	grfwdm_o							RD2MFM_O	RD2MFM_Sel

3.控制与异常处理模块

- Main_Controller主控制器：指令译码，功能部件控制信号生成与AT生成
- Stall_Controller暂停控制器：生成暂停信号
- Forward_Controller转发控制器：生成转发信号

Main_Controller

模块端口定义

名称	位宽	方向	描述
instr_i	32	I	D级流水线寄存器输出的指令
clr_instr	1	I	是否清空D级指令，高电平有效
a1	5	O	GRF读地址1
a2	5	O	GRF读地址2
a3	5	O	GRF写地址
npc_op	4	O	NPC操作码
ext_op	1	O	EXT操作码
alu_op	5	O	ALU操作码
md_op	3	O	MD操作码
start	1	O	乘除启动控制
srcbm_sel	1	O	SRCBM控制
mdm_sel	1	O	MDM控制
be_op	2	O	BE控制
dmext_op	3	O	DMEXT控制
regwrite	1	O	RF写使能
grfwdm_sel	3	O	GRFWDM控制
instr_code	宏定义	O	指令总线输出
cp0_write	1	O	CP0写使能
cp0_addr	5	O	CP0地址
cp0_sel	3	O	CP0寄存器选择信号
cache_req	1	O	cache控制信号
cache_op	5	O	Instr的rt域
eret	1	O	ERET译码
mtc0	1	O	MTC0译码
bd	1	O	当前是否为延迟槽指令，高电平有效

Stall_Controller

模块端口定义

名称	位宽	方向	描述
instr_code	宏定义	I	D级指令总线
tnew	2	O	
tnew_E	2	I	E级Tnew
tnew_M	2	I	M级Tnew
a1	5	I	D级GRF读地址1
a2	5	I	D级GRF读地址2
a3_E	5	I	E级A3
a3_M	5	I	M级A3
regwrite_E	1	I	E级GRF写使能
regwrite_M	1	I	M级GRF写使能
busy	1	I	MD输出信号
start_E	1	I	E级Start
mtc0_E	1	I	E级MTC0译码
mtc0_M	1	I	M级MTC0译码
cp0_addr_E	5	I	E级CP0地址
cp0_addr_M	5	I	M级CP0地址
stall	1	O	暂停请求

Forward_Controller

模块端口定义

名称	位宽	方向	描述
A1	5	I	
A1_E	5	I	
A2	5	I	
A2_E	5	I	
A2_M	5	I	
A3_E	5	I	
A3_M	5	I	
A3_W	5	I	
Tnew_E	2	I	
Tnew_M	2	I	
RegWrite_E	1	I	
RegWrite_M	1	I	
RegWrite_W	1	I	
GRFWDM_Sel_M	3	I	
RD1MFD_Sel	3	O	
RD2MFD_Sel	3	O	
RD1MFE_Sel	3	O	
RD2MFE_Sel	3	O	
RD2MFM_Sel	1	O	

功能部件控制信号

- EXCEL中完成

Tuse信号：

- Tuse_RS0：

```
BEQ|BNE|BGEZ|BGTZ|BLEZ|BLTZ|JR|JALR|BGEZAL|BLTZAL
```

- Tuse_RS1：

```
ADDI|ADDIU|SLTI|SLTIU|ANDI|ORI|XORI|ADD|ADDU|SUB|SUBU|SLT|SLTU|AND|OR|NOR|XOR|SLLV
|SRLV|SRAV|SW|SH|SB|LW|LH|LHU|LB|LBU|MTHI|MTLO|MULT|MULTU|DIV|DIVU
```

- Tuse_RT0:

```
BEQ|BNE
```

- Tuse_RT1:

```
SLL|SRL|SRA|ADD|ADDU|SUB|SUBU|SLT|SLTU|AND|OR|NOR|XOR|SLLV|SRLV|SRAV|MULT|MU
LTU
|DIV|DIVU|SW|SH|SB
```

- Tuse_RT2:

```
MTC0
```

- Tuse_MD:

```
MFHI|MFLO|MTHI|MTLO|MULT|MULTU|DIV|DIVU
```

Tnew类型：

ALUType（TnewE=1）：

```
ADDI|ADDIU|SLTI|SLTIU|ANDI|ORI|XORI|SLL|SRL|SRA|ADD|ADDU|SUB|SUBU|SLT|SLTU|AND|O
R|NOR
|XOR|SLLV|SRLV|SRAV|MFHI|MFLO|LUI
```

DMType（TnewE=2）：

```
LW|LH|LB|LHU|LBU|MFC0
```

PCType（TnewE=0）：

```
JAL|JALR|BGEZAL|BLTZAL
```

BD:

```
bd = `JAL|`JR|`JALR|`J|`BEQ|`BNE|`BGEZ|`BGTZ|`BLEZ|`BLTZ|`BGEZAL|`BLTZAL
```

暂停与转发策略

- 对于当前指令，计算Tuse_RS0,Tuse_RS1,Tuse_RT0,Tuse_RT1,Tuse_RT2与Tnew.
- Tuse:处于D级的指令再过多少周期需要使用寄存器输出值
- Tnew：位于当前级(E或M)的指令再过多少周期产生**写入寄存器中的值**
- 暂停条件：
  - &当前指令Tuse小于某级指令Tnew
  - &当前Tuse的读取寄存器地址与后级Tnew对应的寄存器写地址相同
  - &后级写使能为1
  - &读取寄存器地址不为0
  - | D级为MD相关指令，E级Start或Busy为1
  - |D级为eret且E级或M级为mtc0指令且CP0写地址为14
- **暂停操作：**
  - 冻结D级流水线寄存器（D.EN = ~Stall）

- PC写使能为0(PC.EN = ~Stall)
  - 问题：暂停时到来外部中断，PC无法写入异常处理程序入口值
  - 解决办法：PC使能：(~Stall|Req)
- 清空E级所有流水线寄存器(E.reset = Stall)包括CU中所有E级控制信号，Tnew_E

- **转发选择信号生成规则：**
  - 当前位点的读取寄存器地址与某转发输入来源的写入寄存器地址相等
  - 当前位点读取寄存器地址不为0
  - **转发源Tnew为0（已经产生了新值）**
  - 根据GRFWDM_Sel判断M级转发PC8还是AO还是MDO
  - 有多个转发源时，选择流水级靠前的
  - 转发源写使能为1

RS策略矩阵

Tuse/Tnew	ALU_E	DM_E	PC_E	ALU_M	DM_M	PC_M	ALU_W	DM_W	PC_W
	1	2	0	0	1	0	0	0	0
0	**S**	**S**	F	F	**S**	F	F	F	F
1	F	**S**	F	F	F	F	F	F	F

RT策略矩阵

Tuse/Tnew	ALU_E	DM_E	PC_E	ALU_M	DM_M	PC_M	ALU_W	DM_W	PC_W
	1	2	0	0	1	0	0	0	0
0	**S**	**S**	F	F	**S**	F	F	F	F
1	F	**S**	F	F	F	F	F	F	F
2	F	F	F	F	F	F	F	F	F

CP0（协处理器0）

模块接口定义

信号名	方向	用途	产生来源及机制
clk	I	时钟信号	
reset	I	复位信号	
addr[4:0]	I	读写 CP0 寄存器编号	执行 `mfc0`，`mtc0` 指令时产生
cp0_sel[2:0]	I	cp0寄存器选择信号	main_controller产生
wdata [31:0]	I	CP0 寄存器的写入数据	执行 `mtc0` 指令时产生
en	I	CP0 寄存器写使能，优先级低于中断	执行 `mtc0` 指令时产生
pc_M[31:0]	I	中断/异常时的 PC	M级流水线输出的PC
alu_result_M[31:0]	I	中断/异常时的M级ALU输出	用于BadVAddr
ExcCode[4:0]	I	流水线中发生的异常的类型	EP
HWInt[5:0]	I	6 个设备中断	外部设备
EXLClr	I	置 0 SR 的EXL 位	执行 `eret` 指令时产生
bd	I	判断M级指令是否为分支延迟槽指令	由D级产生
ext_req	O	中断请求	由 CP0 模块确认响应中断
epc[31:0]	O	EXC 寄存器输出至 NPC	输出值为字对齐
rdata[31:0]	O	CP0 寄存器的输出数据	执行 `mfc0` 指令时产生

CP0寄存器定义

- 位于流水线M级

BadVAddr寄存器 地址：8，select0

- 只读，记录最近一次导致发生地址错例外的虚地址，在实现中保存出错指令的PC或错误访存地址

Count寄存器 地址：9，select0

- 内部计数器，每隔一个CPUclock（每两个CPU周期）累加1
- 可读可写，无复位值

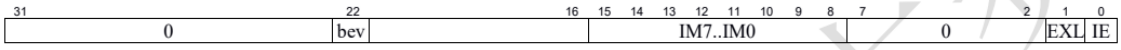
Compare寄存器 地址：11，select0

- 可读可写

SR （Status）状态寄存器 地址：12,select0

- 可读可写寄存器，包含有处理器操作模式，中断使能以及处理器状态诊断信息。
- SR寄存器中EXL位由CP0自行更改，其它位只能由外部输入信号更改。

图 6-3 Status 寄存器格式

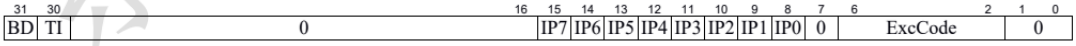


域名称	对应位	功能	读/写	复位值
0	31至23	只读，恒为0	0	0
Bev	22	恒为1	R	1
0	21至16	只读恒为0	0	0
IM7..IM0	SR[15:8]	为 8 位中断屏蔽位，分别对应 8 个中断。相应位置 1 表示允许中断，置 0 表示禁止中断。	R/W	无
0	7至2	只读恒为0	0	0
ERL	2	Error Level,0：正常，1：异常。当Reset,Soft Reset,NMI,或者Cache异常发生时置位。为1时处理器运行在核心态模式，软件与硬件中断被屏蔽，eret指令会使用ErrorEPC作为返回地址。kuseg地址映射为unmapped。	R/W	1
EXL	SR[1]	为异常级。该位置 1 表示已进入异常，不再允许中断，置 0 表示允许中断。当除了Reset,Soft Reset,NMI,或者Cache异常以外的异常或中断发生时置位。当EXL为1时，处理器自动处于核心态，**所有硬件与软件中断被屏蔽**，EPC，Cause的BD在发生新的例外时不做更新	R/W	0
IE	SR[0]	为全局中断使能。该位置 1 表示允许中断，置 0 表示**屏蔽所有硬件和软件中断。**	R/W	0

Cause 原因寄存器 地址：13,select0

- Cause寄存器主要用于描述最近一次例外的原因，除此之外还对软件中断进行了控制。
- 除了IP1,IP0域外，Cause寄存器的其它域对于软件均只读。

图 6-4 Cause 寄存器格式



域名称	对应位	功能	读/写	复位值
BD	31	标识最近发生例外的指令是否处于分支延迟槽，高电平有效。**发生异常时才更新**	R	0
TI	30	恒为0（release1版本）实现为当Compare=Count时置1	R	0
0	29..16	只读恒为0	0	0
IP7..IP2	15..10	**待处理**硬件中断标识，每一位对应一个中断线，由高至低分别对应硬件中断5~0（每周期均更新）(就是Latch)	R	0
IP1..IP0	9..8	**待处理**软件中断标识，每一位对应一个软件中断，由高至低分别对应软件中断1,0.软件中断标识位可由软件设置和清除	R/W	0
0	7	只读恒为0	0	0
ExcCode	6..2	异常编码，记录当前发生的是什么异常。**产生异常时才更新。**若为外部异常，则该域为0，若为内部异常，则如实记录		
0	1..0	只读恒为0	0	0

ExcCode编码及其对应例外类型

EPC 寄存器 地址：14，select0

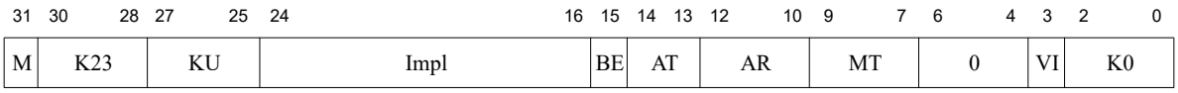
- EPC 寄存器负责保存中断/异常时的 PC 值。
- 响应同步（精确）例外时，处理器向EPC写入直接触发例外的指令的PC（或PC-4）
- 响应异步（非精确）例外时，处理器向EPC寄存器中写入例外处理完成后继续执行指令的PC
- 当Status寄存器的EXL位是1时，发生例外时不更新EPC

ErrorEPC 地址：30，select0

- 与EPC一样（？）

config寄存器 地址：16，select0

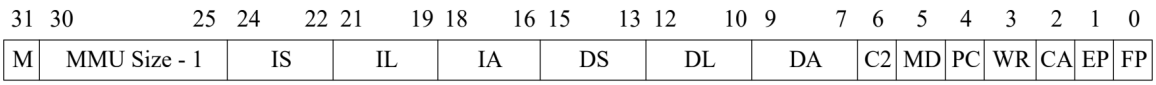
- config寄存器保存cpu的资源信息和配置。config寄存器的K23,KU,K0域必须由软件初始化。



域名称	bits	描述	Read/Write	复位值
M	31	为1表示实现了Config1	R	1
K23	30:28	对实现了固定地址映射的MMU，该域定义kseg2与kseg3的可缓存性。2：uncached，3：Cached。对于其它类型MMU，该域为0	R/W	未定义或0
KU	27:25	对实现了固定地址映射的MMU，该域定义kuseg的可缓存性。对于其它类型MMU，该域为0	R/W	未定义或0
Impl	24:16	reserved		未定义
BE	15	大小端信息。0：小端。1：大端	R	0
AT	14:13	处理器架构信息。	R	0
AR	12:10	MIPS32 架构版本信息。	R	0
MT	9:7	MMU类型。0：无。1：标准TLB。3：固定地址映射	R	3
0	6:4	读出值必须为0	0	0
VI	3	是否使用了虚拟index与标签的指令Cache。0：未使用。1：使用。	R	0
K0	2:0	kseg0的可缓存性。2：uncached。3：cached	R/W	未定义

config1寄存器 地址：16 select1

- cache size = associativity * line size * sets per way
- 所有域均为只读



域名称	bits	描述	Read/Write	复位值
M	31	是否有Config2寄存器。1：存在。0：不存在。	R	0
MMU-Size-1	30:25	TLB数组的大小减一	R	0
IS	24:22	I-Cache每路的行数	R	1（128行）
IL	21:19	I-Cache行大小	R	4（32字节）
IA	18:16	I-Cache相联度	R	1（2路）
DS	15:13	D-Cache每路的行数	R	1（128行）
DL	12:10	D-Cache行大小	R	4（32字节）
DA	9:7	D-Cache相联度	R	1（2路）
C2	6	是否实现了协处理器2，0：没实现。1：实现。	R	0
MD	5	MIPS32中没用	R	0
PC	4	是否实现了性能计数器。0：没实现。1：实现。	R	0
WR	3	是否实现了监视寄存器。0：没实现。1：实现。	R	0
CA	2	是否实现了MIPS16e。0：没实现。1：实现。	R	0
EP	1	是否实现了EJTAG。0：没实现。1：实现。	R	0
FP	0	是否实现了FPU。0：没实现。1：实现且可以访问。	R	0

异常处理操作

- 例外入口向量位置：**0xBFC0_0380**

内部异常的流水

- 由EP完成

检测中断

中断包括硬件中断，软件中断，计时器中断

流水线响应中断的条件：

- &8个中断请求，至少有1个有效且未被屏蔽`(|(Cause[15:8]&SR[15:8]))`
- &全局中断使能有效（SR的IE=1）
- &当前不处于中断服务程序中（SR的EXL=0且SR的ERL=0）

流水线响应异常条件：

- &当前不响应外部中断
- &异常码有效
- &当前不处于中断服务程序中（SR的EXL=0且SR的ERL=0）

支持的异常：

异常与中断码	助记符与名称	指令与指令类型	描述
0	`Int`(外部中断)	所有指令	中断请求
4	`AdEL`(取指异常)	所有指令	PC地址未字对齐
10	`RI`(未知指令)	-	未知的指令码，仅考虑OP,FUNCT?
12	`Ov`(溢出异常)	`add`，`addi`，`sub`或有算术溢出的指令	算术溢出
8	Sys(系统调用)	执行SYSCALL触发	
9	Bp（断点例外）	执行BREAK触发	
4	`AdEL`(取数异常)	lw	取数地址未与 4 字节对齐
4		lh，lhu	取数地址未与 2 字节对齐
5	`AdES`(存数异常)	`sw`	存数地址未 4 字节对齐
5		`sh`	存数地址未 2 字节对齐

- **表 5-1 例外优先级**

例外	类型
中断	异步
地址错例外—取指	同步
保留指令例外	同步
整型溢出例外、陷阱例外、系统调用例外	同步
地址错例外—数据访问	同步

- 发生取指异常后视为`nop`直至提交到 CP0。（由EP控制，在CU中判断是否将Instr置为0）

- 发生 `RI` 异常后视为 `nop` 直至提交到 CP0。 （什么都不做，此时没有指令线被驱动，所有信号均为0，A1,A2，A3可能不是0，但因为写使能为0，所以不会有转发，因为Tuse没有被驱动，所以不会有暂停，最后的结果和nop等效）
- 发生地址错例外（AdEL,AdES）时BadVAddr记录例外触发的虚地址

中断模式

- 处理器支持两个软件中断（SW0,SW1），6个硬件中断(HW0~HW5)和一个计时器中断，计时器中断复用HW5硬件中断
- 软件中断中断源为Cause.IP[1:0],该域仅可以通过软件进行更改
- 计时器中断中断源记录在Cause.TI位，在Count=Compare的情况下由硬件置1，软件可通过写Compare寄存器间接清除Cause.TI位记录的中断
-

表 5-2 各中断请求生成条件

中断类型	中断源	中断请求生成
硬件中断 5 号、计时器中断	HW5	Cause.IP7 & Status.IM7
硬件中断 4~0 号	HW4	Cause.IP6 & Status.IM6
	HW3	Cause.IP5 & Status.IM5
	HW2	Cause.IP4 & Status.IM4
	HW1	Cause.IP3 & Status.IM3
	HW0	Cause.IP2 & Status.IM2
软件中断	SW1	Cause.IP1 & Status.IM1
	SW0	Cause.IP0 & Status.IM0

CP0响应异常中断的操作

同一个周期内完成：

中断操作：

- 保存：将M级指令的PC值写入EPC（若当前中断M级指令为延迟槽指令，则需保存PC-4）
- 跳转：产生中断处理程序入口地址并写入PC
- 关中断：根据情况将EXL或ERL置位，防止再次进入
- 清空所有流水线寄存器

异常操作：

- 保存：将M级指令的PC值写入EPC（若产生异常指令为延迟槽指令，则需保存PC-4），将ExcCode写入Cause
- 跳转：产生中断处理程序入口地址并写入PC
- 关中断：根据情况将EXL或ERL置位，防止再次进入
- 清空所有流水线寄存器

产生异常中断时避免M级PC为0：暂停时不清空E级PC的流水线寄存器与流水线寄存器中的BD_E

由ERET指令结束中断

- eret在D级完成译码，并执行。eret流水到M级，输出，令EXL复位。eret在D级执行跳转操作。
- eret在D级的暂停条件：D级为eret指令并且流水线中存在mtc0指令并且mtc0的写地址为14

note:将EPC接入NPC并删除PCM，可以缩短关键路径，不过需要当eret在D级时，如果下一拍不是暂停，则清空D级寄存器，同时将NPC在此上升沿写入PC的值也写入D级PC（避免产生PC的空泡）（可行的方法）

eret触发的操作：

- 恢复PC：将EPC写入PC
- 开中断：清除EXL与ERL，允许中断再次发生
- 清除ERL

EP

模块功能定义

- 产生F,D,E级的异常
- 系统调用例外，陷阱例外，地址错例外-数据访问在E级产生

模块端口定义

名称	位宽	方向	描述
pc_F	32	I	F级PC
instr_code_D	宏定义	I	D级指令
instr_code_E	宏定义	I	E级指令
alu_result_E	32	I	E级ALU计算结果
overflow_E	1	I	ALU溢出
exc_F	5	O	F级异常码
exc_D			D级异常码
exc_E			E级异常码

异常与中断码	助记符与名称	指令与指令类型	描述
0	`Int`(外部中断)	所有指令	中断请求
4	`AdEL`(取指异常)	所有指令	PC地址未字对齐
10	`RI`(未知指令)	-	未知的指令码，仅考虑OP,FUNCT?
12	`Ov`(溢出异常)	`add`，`addi`，`sub`	算术溢出
8	Sys(系统调用)	执行SYSCALL触发	
9	Bp（断点例外）	执行BREAK触发	
4	`AdEL`(取数异常)	lw	取数地址未与 4 字节对齐
4		lh，lhu	取数地址未与 2 字节对齐
5	`AdES`(存数异常)	`sw`	存数地址未 4 字节对齐
5		`sh`	存数地址未 2 字节对齐

表 5-1 例外优先级

例外	类型
中断	异步
地址错例外—取指	同步
保留指令例外	同步
整型溢出例外、陷阱例外、系统调用例外	同步
地址错例外—数据访问	同步