|  |  |  |
| --- | --- | --- |
| 阶段 | 异常（按优先级） | 触发 |
| IF阶段 | IF Cache错误 |  |
| IF地址错误 | 地址未对齐：  If (Pc[0]|pc[1]==1)  IF\_addr\_error = Enable;  Else  IF\_addr\_error = Disable;  越界：  If ( ((UM==1)&&(pc>text\_upper\_limit))  || ((UM==0)&&(pc>mainmem\_upper\_limit))  || ((UM==0)&&(ERL==ENABLE)&&(pc  > uncached\_upper\_limit ||  (pc <uncached\_down\_limit))  IF\_addr\_error = Enable;  Else  IF\_addr\_error = Disable; |
| IF总线错误 |  |
| ID阶段 | 未定义的系统调用 | If((opcode == syscall)&&(系统调用编号未定义))  Undefined\_syscall\_num = Enable ;  Else  Undefined\_syscall\_num = Disable ; |
| 系统调用 | If(opcode == syscall)  Syscall = Enable ;  Else  Syscall = Disable; |
| 断点 | If(opcode ==break)  break= Enable ;  Else  break = Disable; |
| CP0不可用 | 1. 用户模式下cp0不可用时用了特权指令   If((UM==1)&&(cu0==0)&&(opcode ==eret/mfc0/mtc0/DI/EI))  Cp0\_unusable = Enable ;  Else  Cp0\_unusable = Disable; |
| 未知指令 | If(opcode == 未定义的操作码)  Undefined\_ins = Enable;  Else  Undefined\_ins = Disable; |
| EXE阶段 | CP0不可用 | 2. mtc0、mfc0等要访问的寄存器不存在  ((UM==0)&&(cp0[reg] ==未实现)) )  Cp0\_unusable = Enable ;  Else  Cp0\_unusable = Disable; |
| 算术溢出 | Result = a1 op a2  有符号数溢出：  If (a1[31]⊕ a2[31] ==0)  { if ((funct == add/addi)&& (a1[31]⊕result[31] ==1 ))  Signed\_overflow = Enable;  Else  Signed\_overflow = Disable;  }  else if (a1[31]⊕a2[31] ==1)  {if ((funct == sub) && (a1[31]⊕result[31] == 1))  Signed\_overflow = Enable;  Else  Signed\_overflow = Disable;  }  无符号数溢出(对于计算地址的指令)：  注：此处overflow信号的产生借鉴副本B-35  If((opcode ==branch/load类/store类)&&(overflow == enable))  Unsigned\_overflow = Enable;  Else  Unsigned\_overflow = Disable;  最终产生的overflow信号：  Overflow=Signed\_overflow||Unsigned\_overflow |
| MEM阶段 | MEM cache 错误 |  |
|  | MEM地址错误 | if(word == `ENABLE && (addr[0]|addr[1] == 1'b1))  mem\_addr\_error <= `ADDR\_ERROR\_ENABLE;  else if(half\_word == `ENABLE && addr[0] == 1'b1)  mem\_addr\_error <= `ADDR\_ERROR\_ENABLE;  //地址越界  else if(rw == `RW\_WRITE && um == `UM\_ENABLE &&  (addr <= `TEXT\_UPPER\_LIMIT || addr > `USER\_UPPER\_LIMIT))  mem\_addr\_error <= `ADDR\_ERROR\_ENABLE;  else if (rw == `RW\_WRITE && um == `UM\_DISABLE &&  (addr <= `TEXT\_UPPER\_LIMIT | ( addr > `MEM\_UPPER\_LIMIT)|  (addr >= `UNMAPPED\_UNDER\_LIMIT && addr <= `UNMAPPED\_UPPER\_LIMIT) ) )  mem\_addr\_error <= `ADDR\_ERROR\_ENABLE;  else if (rw == `RW\_READ && um == `UM\_ENABLE &&  (addr > `USER\_UPPER\_LIMIT))  mem\_addr\_error <= `ADDR\_ERROR\_ENABLE;  else if (rw == `RW\_READ && um == `UM\_DISABLE &&  (addr > `MEM\_UPPER\_LIMIT))  mem\_addr\_error <= `ADDR\_ERROR\_ENABLE;  else  begin  mem\_addr\_error <= `ADDR\_ERROR\_DISABLE; |
|  | MEM总线错误 |  |
| WB阶段 | 无 |  |