# **功能概述**

乘法除法单元实现以下9个指令

DIV , DIVU , MUL , MULT , MULTU , MFHI , MFLO , MTHI , MTLO共9个指令。

在乘除法单元中，集成了HI寄存器和LO寄存器以及乘除法功能单元。

从乘除法操作码Md\_op中获得具体操作要求，根据操作要求执行乘除法操作或者MFLO , MFHI, MTLO , MTHI操作。将结果从Res\_out输出。

由于除法的运算会涉及到多个周期，所以在执行除法的时候，要停掉cpu中其他部件的执行，所以通过Md\_stall有效来告知cu，执行相关的停顿操作。

# **输入**

Md\_op[3:0] 乘除法操作码

4’b0001 DIV

4’b0010 DIVU

4’b0011 MFHI

4’b0100 MFLO

4’b0101 MTHI

4’b0110 MTLO

4’b0111 MUL

4’b1000 MULT

4’b1001 MULTU

Rs\_in[31:0] rs操作数输入

Rt\_in[31:0] rt操作数输入

# **输出**

Res\_out[31:0] 结果输出

Md\_stall 由乘除法单元送出的阻塞信号，和存储器输出的mem\_stall信号做或操作，输入到cu中

# **数据冒险相关**

1.

如果在指令1改变了HI寄存器的值， 紧接着的指令2要读取HI寄存器的值。因为在指令1中，HI寄存器的值已经写回，所以指令2读到的值就是最新的值，不存在RAW冒险。写读LO寄存器的情况是一样的。

2.

如果指令1和紧接着的指令2都用写HI寄存器，因为Hi寄存器发射是按此发射，按序到达乘除法单元，所以不存在WAW冒险。写LO寄存器的情况是一样的。

3.

如果指令1读HI寄存器，指令2写HI寄存器，因为指令时按序发射，按序到达乘除法单元，所以不存在WAR冒险。读写LO的情况是一样的。

在乘法除法器中涉及到通用寄存器的操作，其冒险处理由FU和CU处理。

# MFHI

将HI寄存器中的内容从Res\_out输出

# MFLO

将LO寄存器中的内容从Res\_out输出

# MTHI

将Rs\_in的输入存储到HI寄存器中

# MTLO

将Rs\_in的输入存储到LO寄存器中

# MUL

Rs\_in 与 Rt\_in相乘，结果低32位从Res\_out输出（符号扩展）

# MULT

Rs\_in 与Rt\_in相乘， 结果低32为存放在LO寄存器中，高32为存放在HI寄存器中(符号扩展)

# MULTU

Rs\_in 与Rt\_in相乘， 结果低32为存放在LO寄存器中，高32为存放在HI寄存器中(无符号扩展)

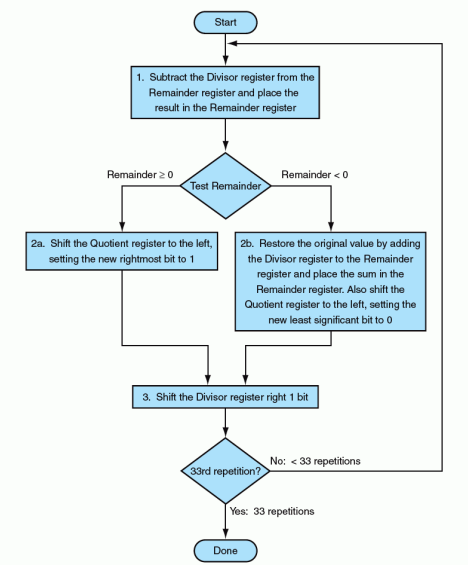
# DIV & DIVU

Rs\_in作为被除数， Rt\_in作为除数，执行除法计算， 计算结果存放在LO中，余数存放在Hi中。除数为0的情况暂不做处理。

DIV对源操作数做符号扩展， DIVU对源操作数做无符号扩展。

除法的执行要32个周期。在此期间，通过Md\_stall 有效阻塞其他部件的执行。

使用如下算法



# **参考资料**

Computer organization and design – the hardware / software interface