

课程项目报告(一)

课程名称: 硬件描述语言原理与应用

项目内容: 32 位除法器设计

姓 名: 周开宁

学 号: <u>3180101148</u>

院 系: 信息与电子工程学院

专业年级: 2018 级微电子科学与工程专业

指导教师: 沈海斌

完成时间: 2020年11月2日

目录

		1
课程项目报告(一)		
—、	设计目的及要求	2
	设计步骤	2
三、	设计内容	2
四、	仿真与验证	5
五、	结论	7
	附录:Verilog 编写规范	

一、设计目的及要求

实现 32 位除法器的 RTL 代码,可同时支持有符号除法和无符号除法。搭建 Testbench (测试平台),对 32 位除法器进行功能验证。运用 Verilog 语言进行建模、仿真和验证。

二、设计步骤

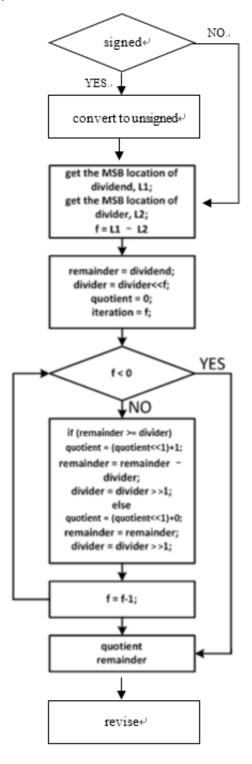
- 1、 根据乘法的"移位——加"性质确定除法的"移位——减"性质。
- 2、 针对除法运算的特点,对被除数、除数、商和余数的情况进行讨论。
- 3、 完成一个无符号数的除法。
- 4、 将有符号数转化为无符号数的除法。
- 5、 对各种输入输出的异常情况进行处理。
- 6、 写 Testbench 进行测试。

三、 设计内容

- 1、 总体框图: 见图 1.
- 2、 行为描述: 32 位除法器的功能实现主要分为四个步骤。
- (1) 第一步,数据预处理。如果有异常的输入,比如 NaN、Infinity、除数为 0 等等,那么输出异常 NaN 或 Infinity.
- (2)第二步,模式选择。如果模式输入为有符号除法,转化为无符号除法。如果无符号除法,则跳过。
- (3)第三步,对非负的被除数和除数进行整除运算,方法如框图所示,得出商和余数。
- (4) 第四步,根据相应的模式为对商和余数进行修正。余数的符号和被除数相同。
- 3、代码描述:

本项目由两个模块以及其两个. v 文件——32 为除法器模块 HD32. v 和测试模

块 testbench. v 组合而成, testbench 实例化除法模块 HD32. HD32 是主要的功能模块,其输入有被除数 dividend,除数 divider,余数 remainder 和商 quotient,以及除零错误的 zero_error。模块 HD32 是不含时钟的,因此它不是一个时序电路,而是一个组合电路。



图表 1 总体设计框图

为了增强可扩展性和通用性, 我将输入输出的位数写成了宏定义的形式。

[`]define InputWidth 32

[`]define OutputWidth 32

在进行实现的异常处理后,由于有符号数和无符号数之间的除法只有取补码的差别,因此为了增加模块的适用范围,降低设计的复杂度,我将有符号的除法转化为无符号数再进行输出。

```
reg_dividend=dividend[`InputWidth-1]==1?~dividend[`InputWidth-1:0]+1:dividend;
reg_divider=divider[`InputWidth-1]==1?~divider[`InputWidth-1:0]+1:divider;
```

由于后期有移位操作,所以我定义了一个名为"msb"的函数,来检测除数与被除数的"1"的最高位,以此来确定一开始除数需要移动的位数。

```
msb_dividend=msb(reg_dividend);
msb_divider=msb(reg_divider);
    //...
function [`InputWidth-1:0] msb;
```

之后就循环进行"移位——减"操作,直到被减数小于除数。此时的被减数就是余数,中间过程中存储的差就是商。

```
temp_divider=reg_divider<<(msb_dividend-msb_divider);
// variational subtrahend
    for(i=msb_dividend-msb_divider;i>=0;i=i-1) begin
        quotient=quotient<<1;
        if (remainder>=temp_divider) begin
            remainder=remainder-temp_divider;
            quotient[0]=1;
        end
        else
            quotient[0]=0;
        temp_divider=temp_divider>>1;
        end
```

最后根据模式位的输入(有/无符号除法)对得到的无符号的整除除法进行修正。其中需要特别注意的是,余数和被除数的符号是相同的,这就使得之前可以通过有符号数转化为无符号数进行运算。

4、 测试例: 测试例涵盖各种情况。

无符号模式下:

- (1) 正数除以某个比自己大的正数;
- (2) 正数除以某个和自己一样大的正数;
- (3) 正数整除另一个正数:
- (4) 正数除以另一个正数,带余数;
- (5) 正数除以 0;
- (6) 最大的正数除以某个正数:

- (7) 0 除以某个正数;
- (8) 0 除以 0:

有符号模式下:

- (9) 正数除以负数, 其绝对值大小参考 1-8 例;
- (10) 负数除以正数, 其绝对值大小参考 1-8 例;
- (11) 正数除以 0;
- (12) 负数除以 0:
- (13) 32 位范围内边界值的除法。

5、 调试技巧:

- (1)记录调试日志。设置 testbench,实例化被测模块,并且使用 monitor系统命令,将所有输出输入输出的变化都打印出来;同时设置延时时间间隔,每隔一段时间就变换输入,完成不同测试样例的测试。对于模块中的变量,可以在模块内部暂时使用 display 命令,显示内部输出。将以上时间戳、输出结合起来作为日志,进行调试。
- (2)生成波形文件观察波形。在代码中生成 vcd 和 vpd 波形文件,通过 DVE 打开 vpd 文件,观察波形进行调试。

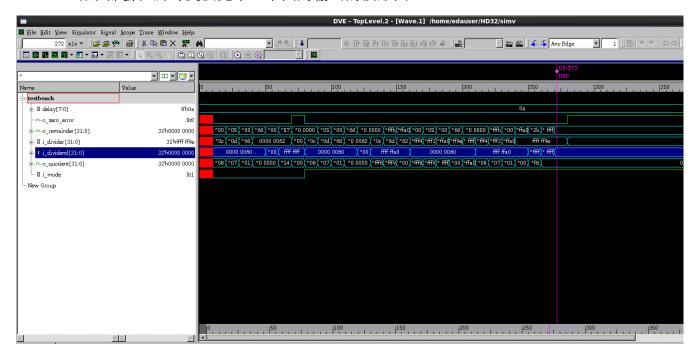
四、 仿真与验证

以下两张图为仿真时输出的日志信息。

```
Compiler version G-2012.09; Runtime version G-2012.09; Nov 2 01:11 2
At time=
mode=x
o zero error=x
At time=
              10
mode=0
dividend=00000000000000000000000001100000
o zero error=0
At time=
              20
mode=0
dividend=00000000000000000000000001100000
quotient=000000000000000000000000000000111
o zero error=0
At time=
              30
mode=0
dividend=00000000000000000000000001100000
divider=000000000000000000000000001100000
o zero error=0
At time=
              40
mode=0
dividend=000000000000000000000000001100000
divider=0000000000000000000000000001100010
remainder=000000000000000000000000001100000
```

```
At time=
       270
mode=1
divider=11111111111111111111111110011110
o zero error=0
At time=
       280
mode=1
o_zero_error=1
```

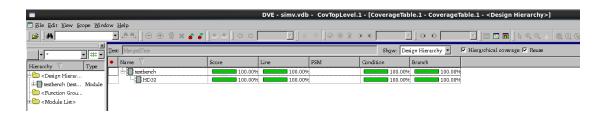
由于我的除法器是一个纯组合电路,所以各种输入切换以后,对应除法计算结果都会在瞬时变换完毕。下图为输出的波形图。



经过细致的测试,达到了100%的代码覆盖率。并且每个测试例的结果都是正确的。

查看代码覆盖率的方法:

- 1、输入 vcs 编译时,加上命令选项 -cm line+cond+fsm+branch+tgl。
- 2、运行 simv 时,也加上命令选项 -cm line+cond+fsm+branch+tgl。
- 3、调用 dve, 输入命令 dve -covdir simv. vdb。



五、 结论

HD32.v 和 testbench.v 能达到设计要求。

六、 附录: Verilog 编写规范

https://verilogcodingstyle.readthedocs.io/en/latest/