实验五 计数器和时钟

实验报告

181860085 汤昊

2019.10.3

一．加法计数器

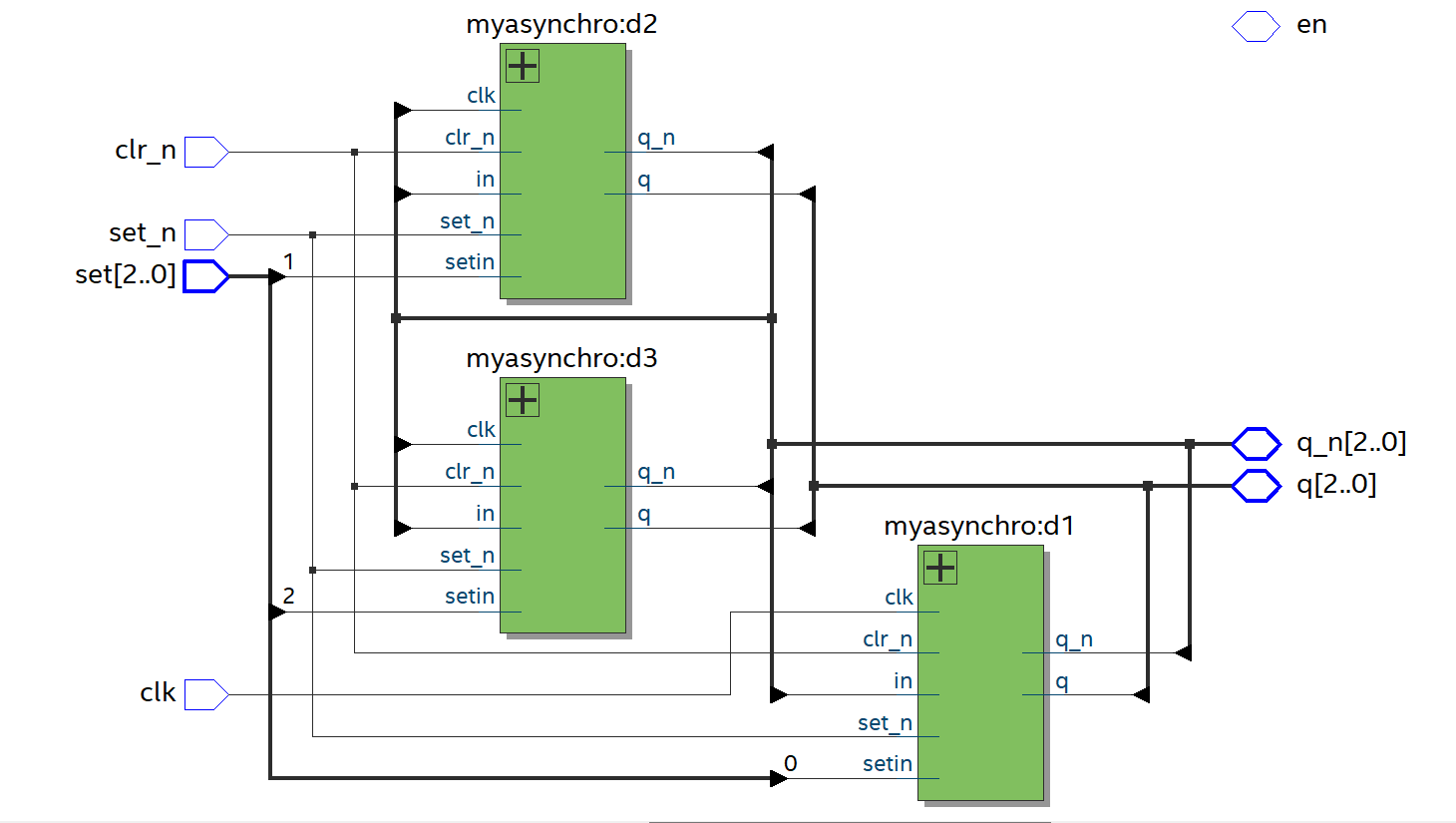
（1）清零端和置数端的设计

在二进制异步加数器的基础上，增加输入clr\_n,set和setin，clr\_n低电位有效，set高电位有效，setin是set的输入。

（2）计数器的设计

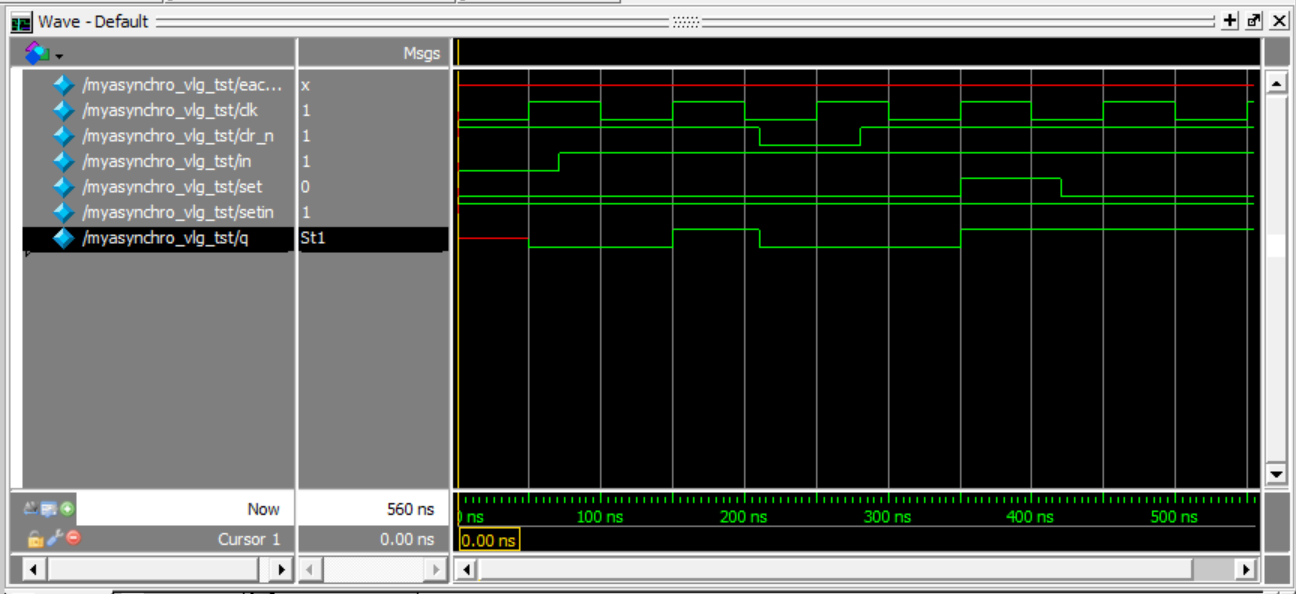
三个异步D触发器串联，增加输入的取反Q\*，前一个触发器的Q\*作为下一个触发器的时钟端，当后一个触发器从0到1，Q\*产生一个上升沿，让下一个触发器进位。同时Q\*也作为这个触发器的输入，如果当前状态是Q，下一个状态就是Q\*。

（3）netlistview



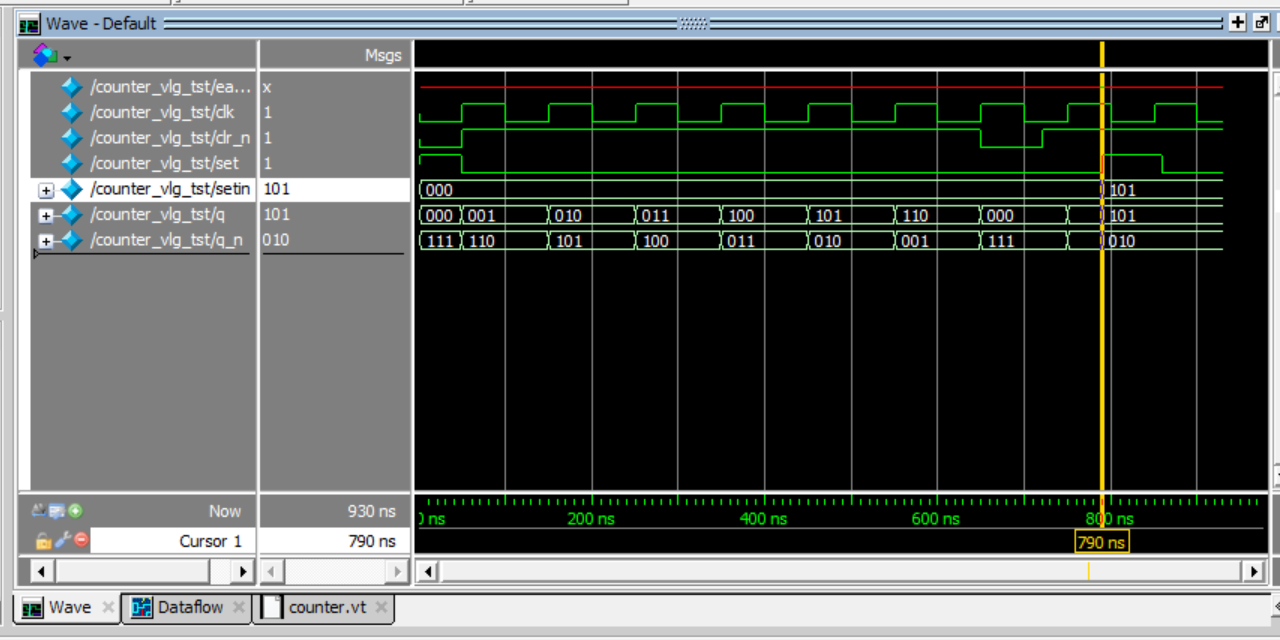
（4）仿真模拟

* 带清零和置数端的D触发器测试



Clr\_n有效时立即清零，set有效时立即转为setin的值

* 用上述三个D触发器实现的行波计数器测试



能够正常循环计数000~111，且在清零端时有效时清零（图中110—000），在置数端有效时置为输入(图中000—101)

二．定时器

1. 分频器

要求：输入50MHz的时钟，输出1Hz，周期为1s的信号，

设计：时钟每秒变化50M次，那么产生25M的上升沿，用一个变量每次上升沿计数，达到25M输出一次信号。由于224=16777216，225=33554432，224<25M<225，因此count\_clk的位宽应该设置为25位

(2) 计时功能

由于计数范围是0-99，所以用两个寄存器分别记录时间的个位和百位，用分频器输出的信号作为时钟，每次上升沿进行循环加操作。

(3) 开始，暂停和清零功能

开始键高电平有效，有效时计数，无效时数码管无显示

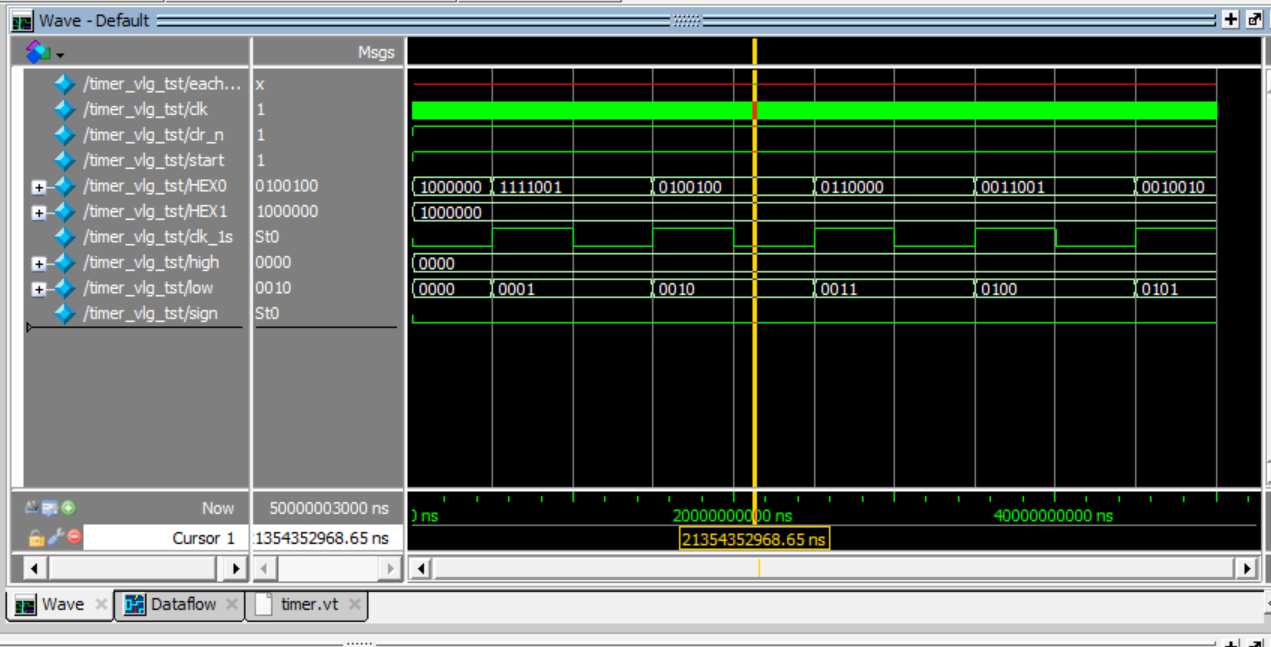
暂停键低电平有效，有效时时钟保持在当前状态

清零键低电平有效，有效时时钟计数归0

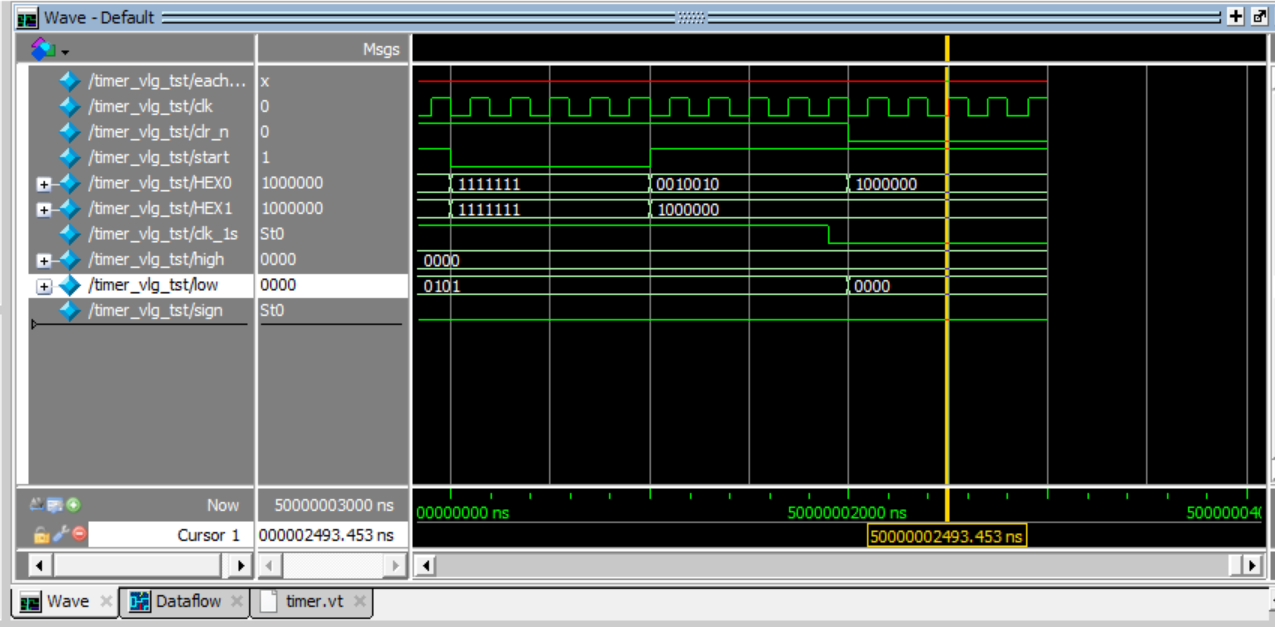
（4）计时结束指示

用一个变量指示，当达到99要+1时，变量为1，其余时间都赋值为0，这样指示的效果可以维持到下一次高电平.

(5)仿真模拟



仿真模拟了五个上升沿，可以看到在每个clk\_1s上升沿，low代表的个位数字递增，HEX0的编码也相应变化



在start无效时代表计数暂停，此时内部计时暂停，数码管全一不再显示，而在清零端clr\_n有效时，high和low存储的值都清零，数码管也为0的编码

sign的测试由于需要到99-0时体现，耗时较长，直接在开发板上验证，实际效果为99变成0时灯亮，持续到变为1时。

三．电子时钟

（1）模块划分

分为分频器（divider）：利用硬件原有的时钟提供1s的时钟输入

时钟（digitclock）：实现计时，秒表，设置时间等功能

显示（display）：实现时间的数码管显示和闹钟的指示

1. 模块设计

* 分频器：与上述计时器相同
* 时钟：用stopwatch和settime代表时钟的三种状态

Stopwatch和settime都为0：正常计时，每个上升沿用if-else语句对时间状态进行判断，得出+1s的结果

Stopwatch为1：秒表状态，进行0-59的循环计数，只显示秒数

Settime为1：设置时间，考虑到开发板的switch开关只有10个，使用两位的setselect指示要设置的时间类型，其他8位为时间输入

Setselect==2’b00：将秒数修改为输入

Setselect==2’b01：将分钟修改为输入

Setselect==2’b10：将小时修改为输入

Setselect==2’b11：暂停

* 显示：用case语句将输入的时间转换为相应的数码管编码，当输入符合程序设置的某个值时，闹钟指示灯亮

1. 仿真模拟

clk=0;settime=0;stopwatch=0; #250000000;//正常计时

set\_l=4'b1001;

set\_h=4'b0101;//设置输入为59

settime=1;

setselect=2'b00;#200000000;//设置秒

settime=0;#100000000;

settime=1;

setselect=2'b01;#150000000;//设置分

settime=0;#200000000;

settime=1;

set\_l=4'b0011;

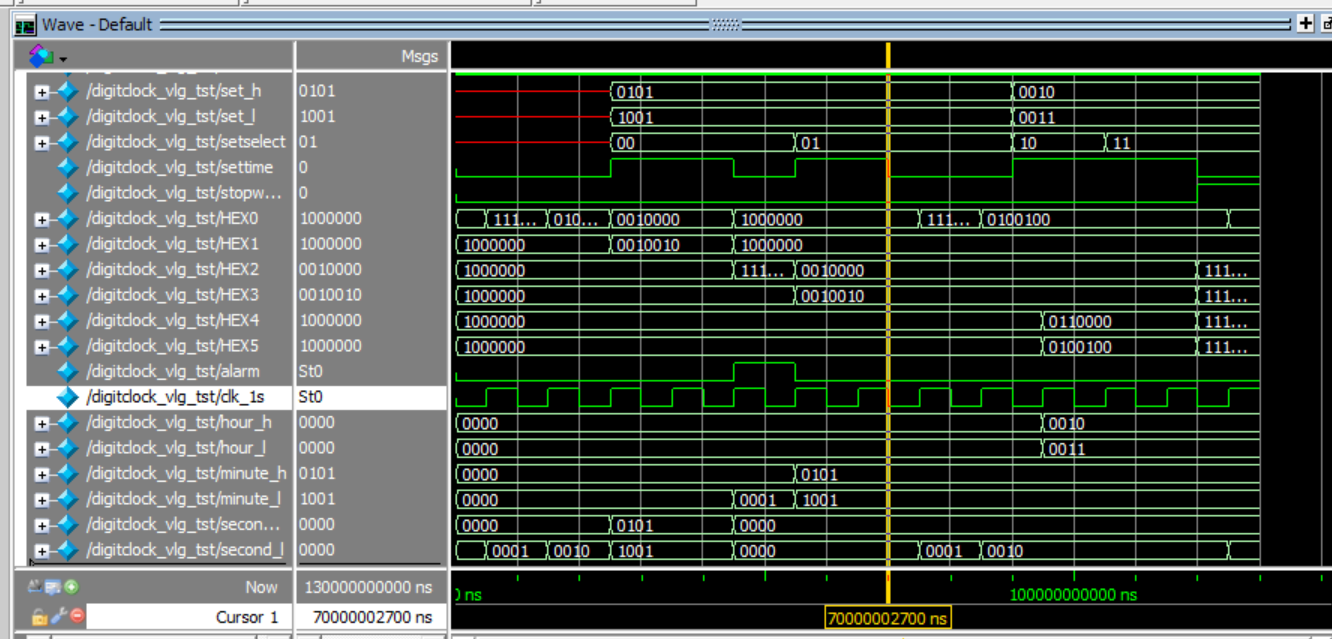
set\_h=4'b0010;

setselect=2'b10;#150000000;//设置小时

setselect=2'b11;#150000000;//暂停

settime=0;stopwatch=1;#100000000;//测试秒表

$stop;



开始正常计时，秒从0到2，之后settime有效，输入为59，秒数置为59，settime置0，恢复计时，下一个上升沿产生进位，秒数为0，分钟为1，接着settime有效，分钟被置为59，之后输入改为23，小时置为23，最后状态切换为秒表，只有秒数变化

四．实验反思与收获

* Always不可同时监测一个变量的上升沿和下降沿
* 可以对reg型变量赋初值，不可以对wire型变量赋初值
* 模块实例化的参数错误在对单个文件编译时无法发现