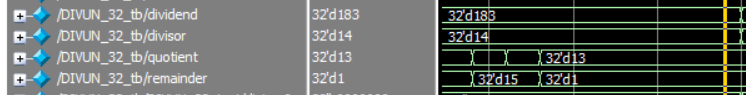
**Log**

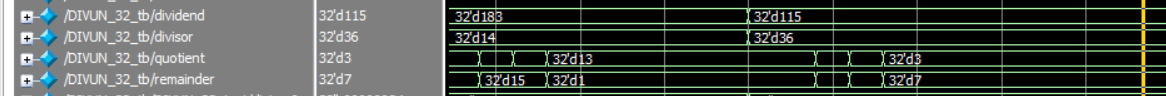
钱煜 3180103948

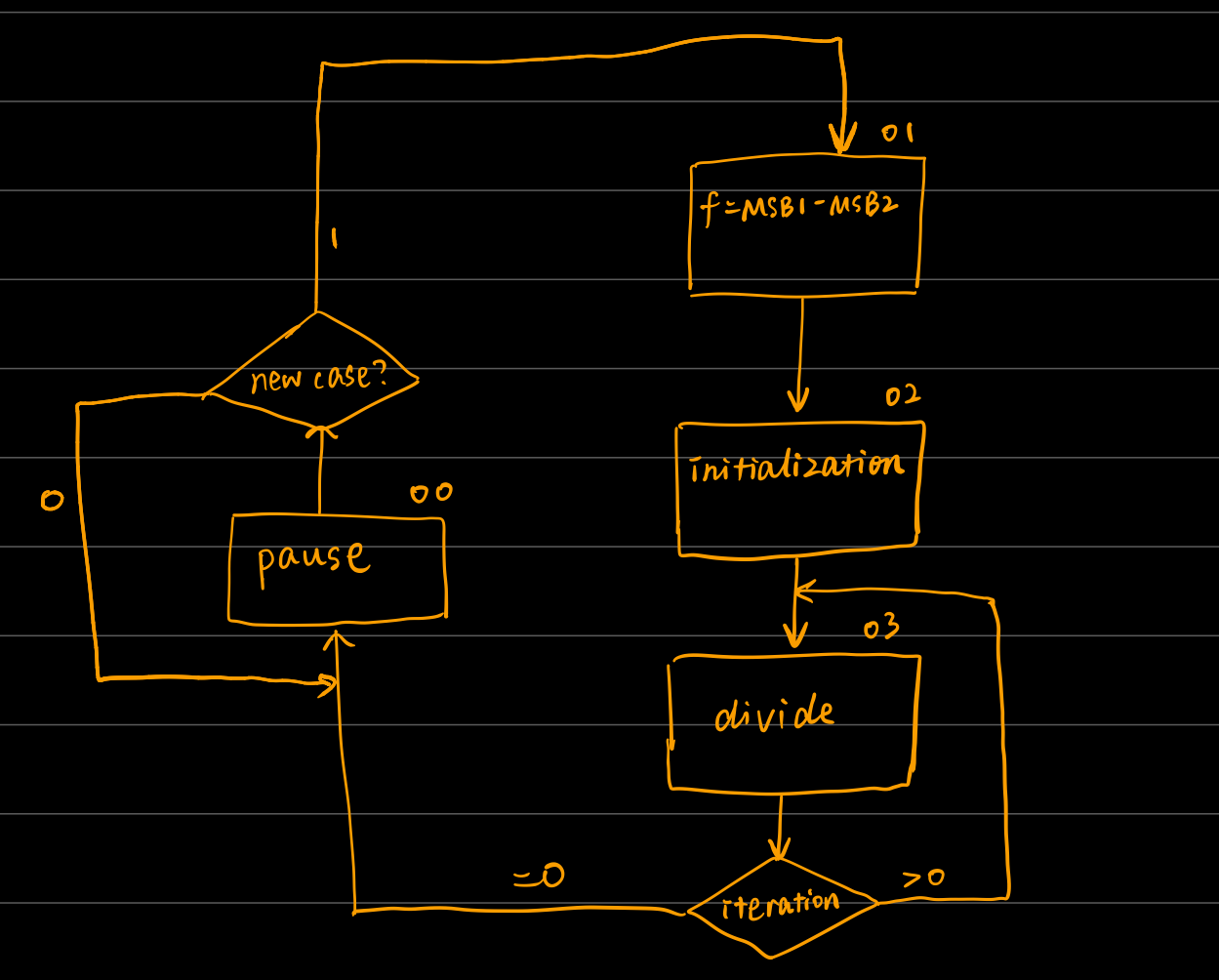
10.8

今天花了一天，写了主要的.v代码以及test bench，在modelsim上进行了仿真，结果基本正确。（只做成了unsigned）

先上结果图：







遇到的问题（花费时间较长的问题）：

1.我设置了一个“暂停态“，即当除法完成后，若发现输入的被除数和除数没有变化，则会在这个状态中循环。为此，我专门制造了两个reg，用来存放一开始的两个参数，然后每次都比较他们与当前的输入。但会出现，看似相等（即没有新的输入），但系统判定为有新输入的状况。

解决方法：经过反复的排查和检查，发现是因为我直接将除数divisor用作了比较对象，而divisor虽然在一开始被移位，且之后不断移位直到原先的状态，但在计算机运算过程中可能出现bug（也可能是软件本身的bug），我在结果图中，divisor和divisor0两者无论是十进制还是二进制都完全相等，但两者的差却不为0。虽然不明白具体原因，但通过添加cmp\_divisor（其中的数据不会变化），我解决了这个问题。

可以优化和需要进一步解决的问题：

1.寻找MSB的location是否有更好的办法？我网上没有找到，目前采用的是case语句（牺牲了电路规模，换得了计算速度）；

2.根据课件中的思路，如果被除数和除数的MSB的location相差太多，则有可能导致下一次运算来临的时候上一次还在state3中运算，这就会引起崩坏；

3.如何在虚拟机中完成以上运算？

4.如何改进下，成为能支持signed？

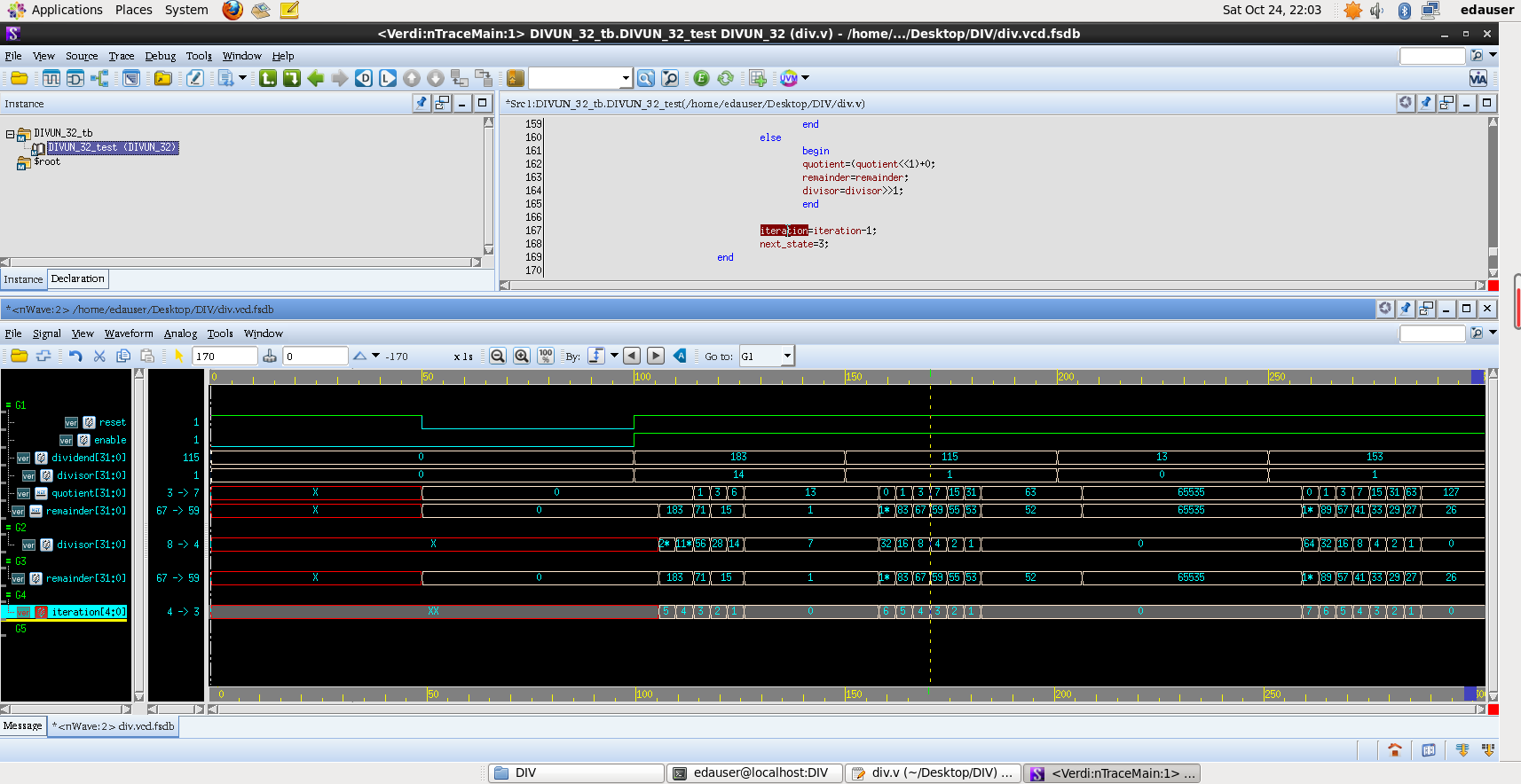
小结：

这是第一次在完全自行看懂课件的基础上，自己琢磨，编写出.v和tb文件，通过不断的debug，也算是有了个半成品，还是很有成就感的。虽然花费了整整一天的时间，但还是值得的。希望之后能不断完善，做得更好，也祝自己国庆最后一天的假期快乐。

10.8 夜 于玉泉教七302

10.24

之前有一些进展，但未log下来。今日将目前所得进行记录。



今天最大的收获是，很多天的学习后，我终于知道了如何在虚拟机上进行仿真除法器。

问题：

1.MSB的具体数位选择有待修改，不然就可能出现如图中115/1的时候出现的少除一次的情况；

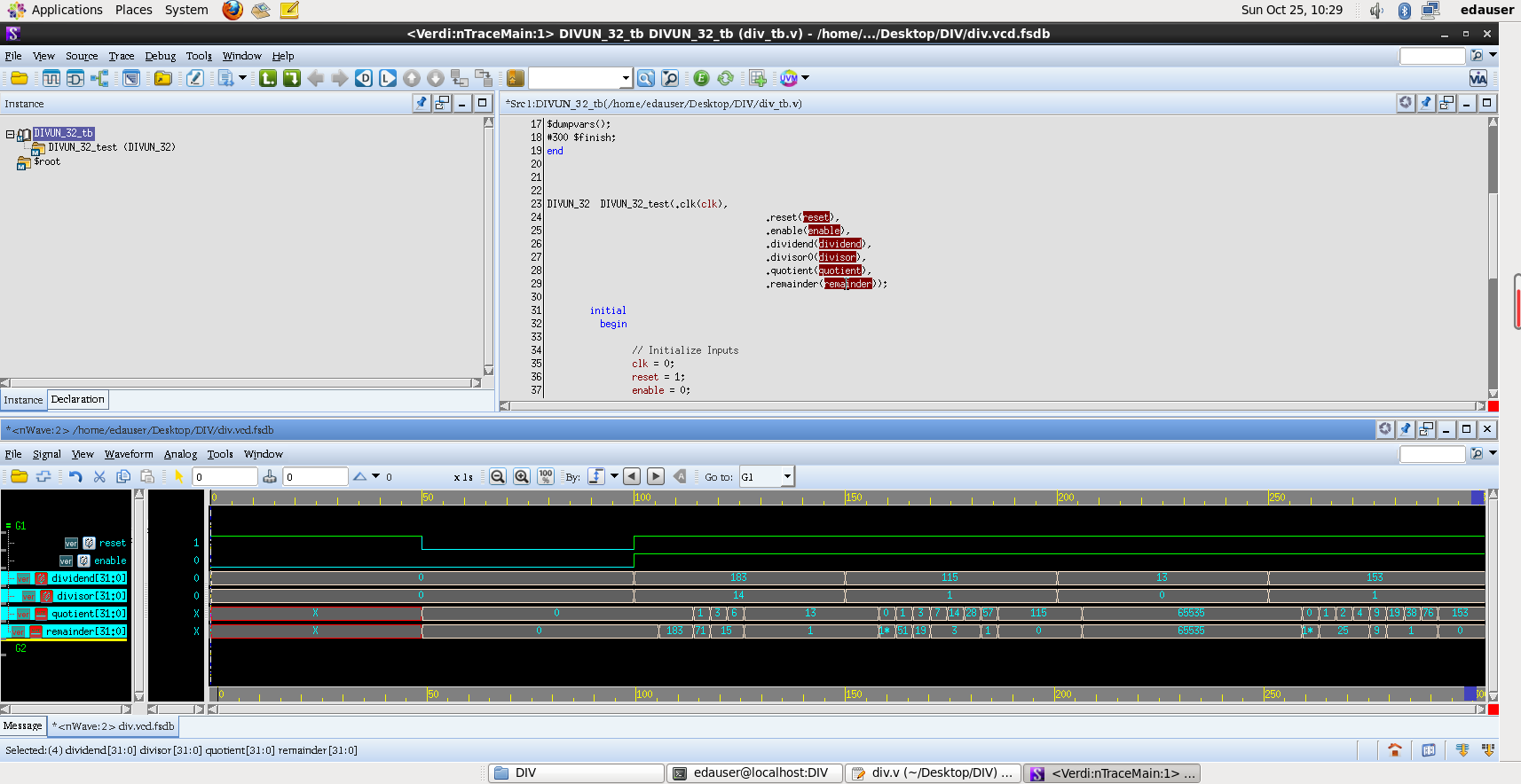
2.有符号除法尚未解决。

心得：

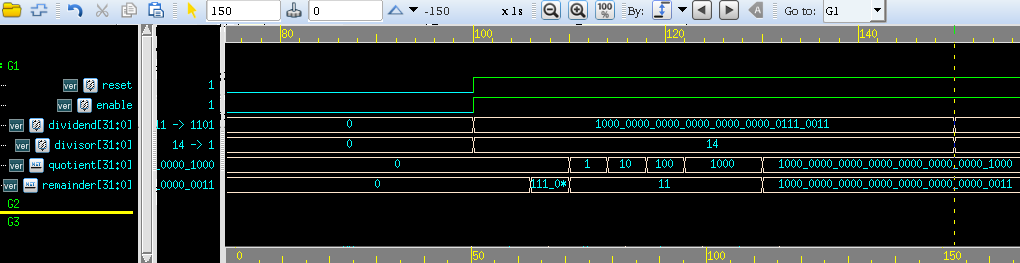
1.虽然从结果论来看，我在熟悉软件运用上多花费了很多很多的时间无用功（因为命令行里只需要很基础的命令就能解决），但过程中学习到的知识还是宝贵的，眼光不能只放在当前这个项目上；

2.添加了/0的case，使得结果为最大的商+最大的余数（表示正无穷）

10.25



完成了无符号除法器，通过重新设定MSB解决了昨天的问题，至此，无符号除法器基本完成。



完成了有符号除法器，将最高位设置为符号位，这样一来数据处理范围变小了。从结果来看，-115/14=-8…-3，符合要求。至此，有符号除法器基本完成。

通过查找MIPS指令集，知道了机器码的具体构成，以及如何识别各个命令。

问题：

1.知道命令后如何读取寄存器的值？是要写在processor里吗？

2.目前输入就是32位的机器码，testbench怎么写？

10.27

更新下。

目前完成：单独除法器，浮点单元里的加法器和乘法器（希望没错）

有待完成和解决：

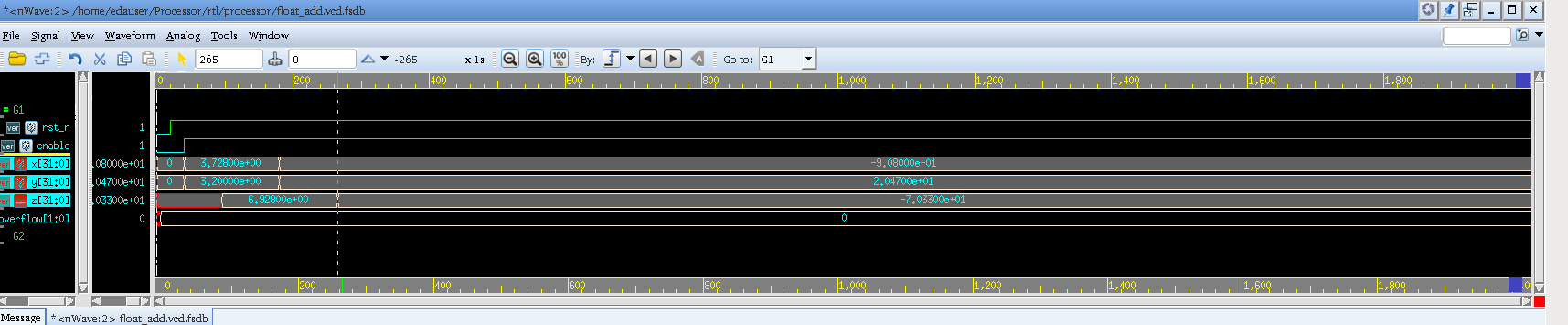
1.虚拟内存

2.processor到底是怎样的？

3.除法器和精度转换

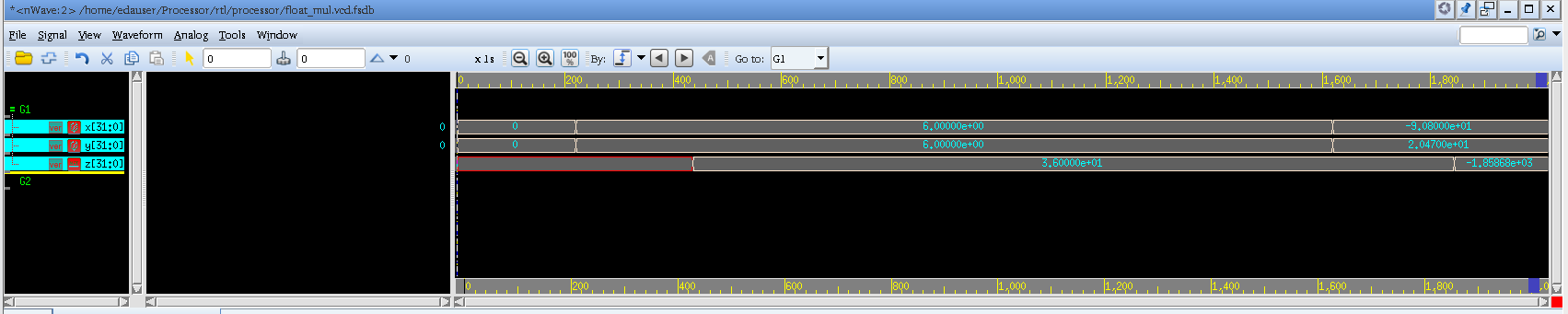
4.最终交什么？

10.28



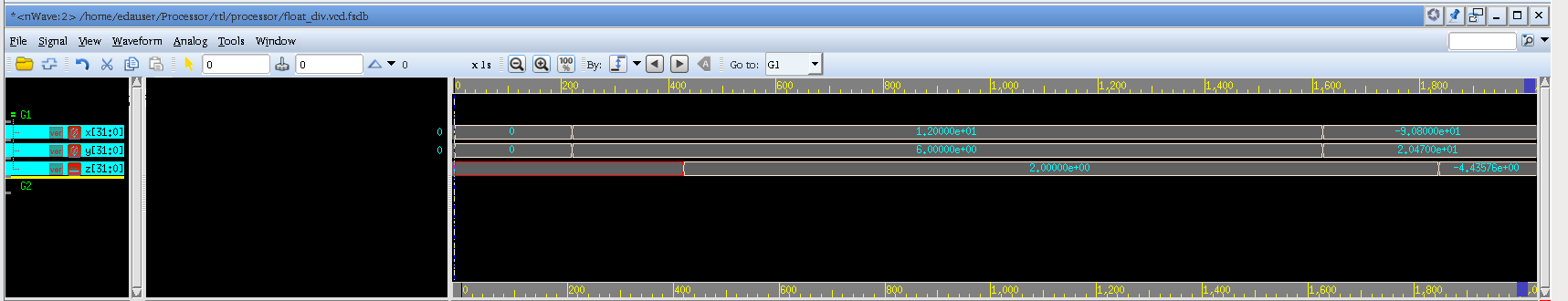
加法器完成！！！

10.29



单精度乘法器完成！

10.30



单精度除法器完成！

总体架构：

1.加法器

FPU-add：enable；

FPU-Memory：读写信号，两个读出地址信号，一个写入地址信号，一个立即数写入数值

Add-memory：memory的输出中，ft，fs与add的x，y连线，fd与add的z连线。

工作状态：

1.译码，进入addread状态；

2.读数据，利用fs和ft的地址，把memory中的数值放到 r\_data\_ft;

r\_data\_fs; 两个输出寄存器（其实因为连线，此时add里的x，y中已经有数据），进入addcal状态

3.运算，得到z，同理，此时memory中的wdata也有数据，进入addwrite状态

4.写入，在memory中利用addr\_fd和wdata把数据写入对应的内存，进入over状态

5.over状态负责所有enable归0，同时比较命令有无变化，如果有则跳转到decode状态，否则一直循环

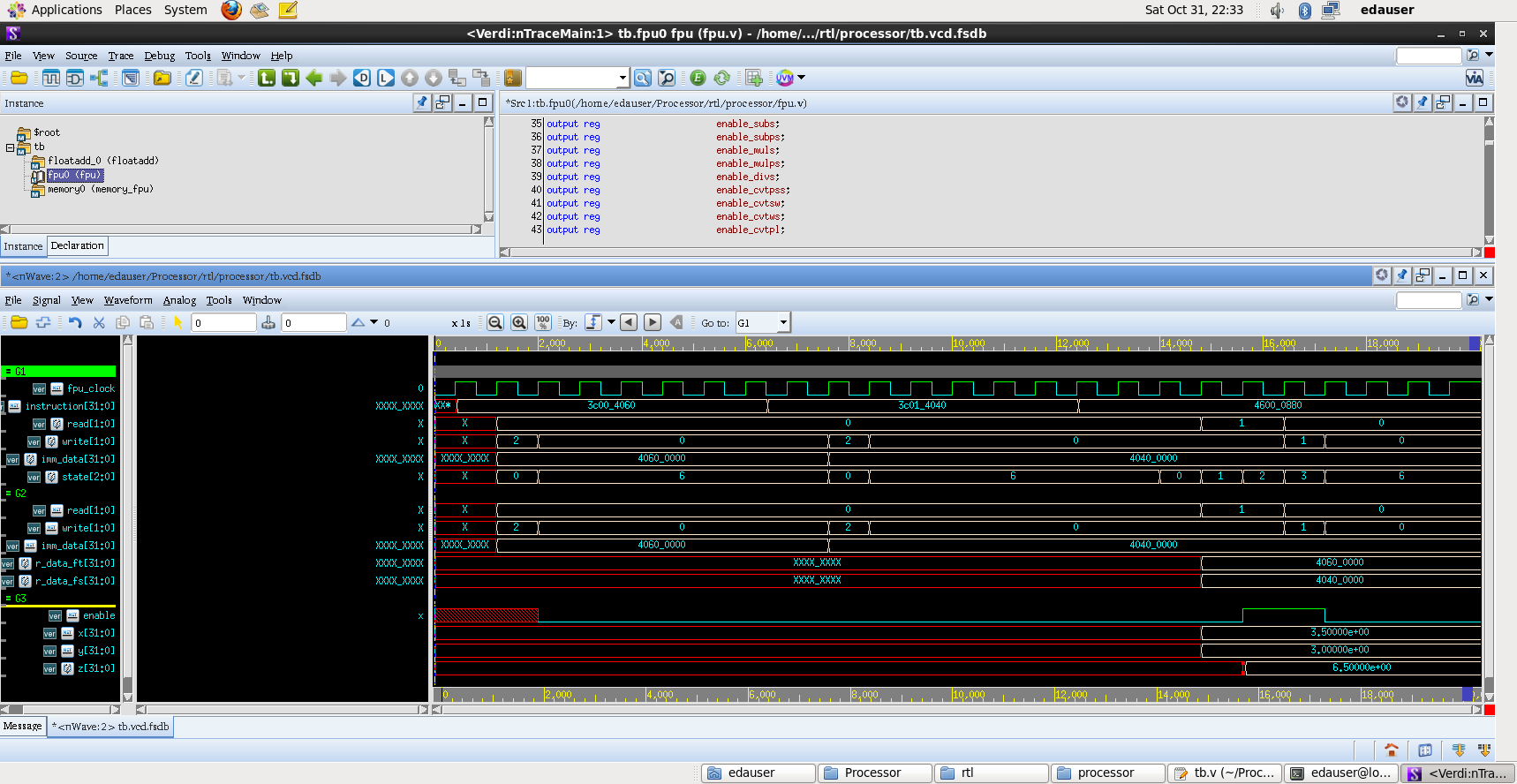
可能问题：

信号线之间的冲突

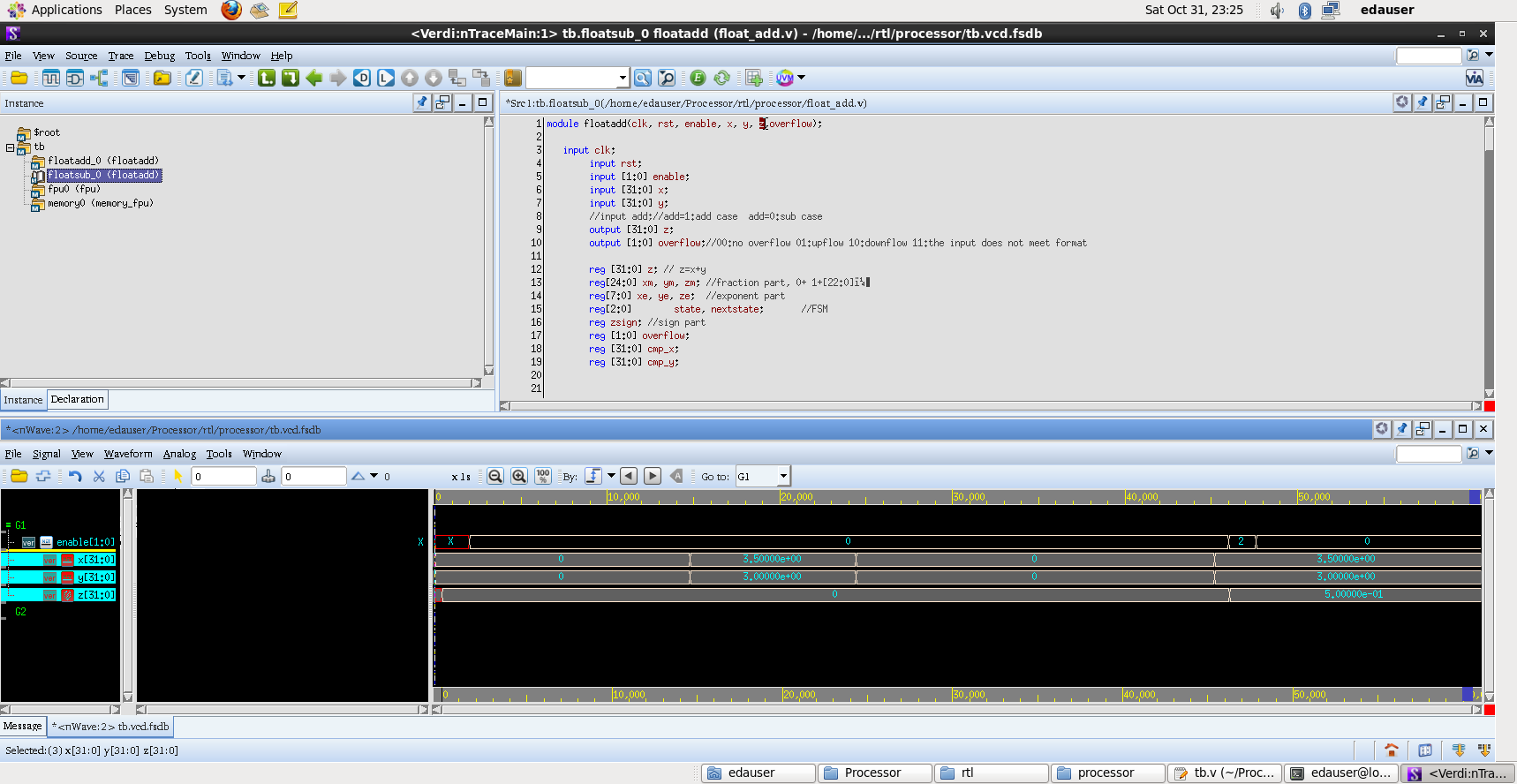
可能的解决办法：

每种case设置单独的线

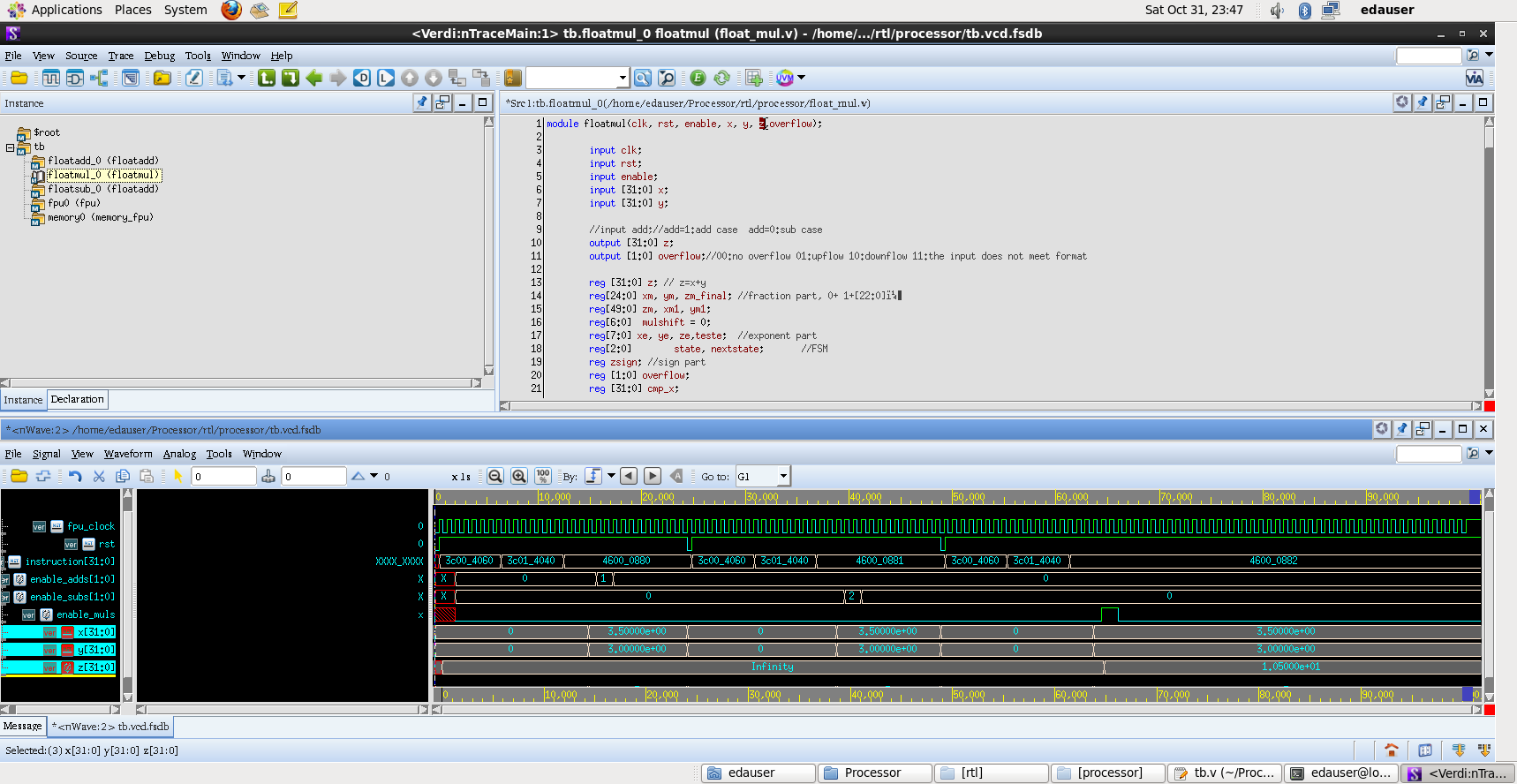
10.31



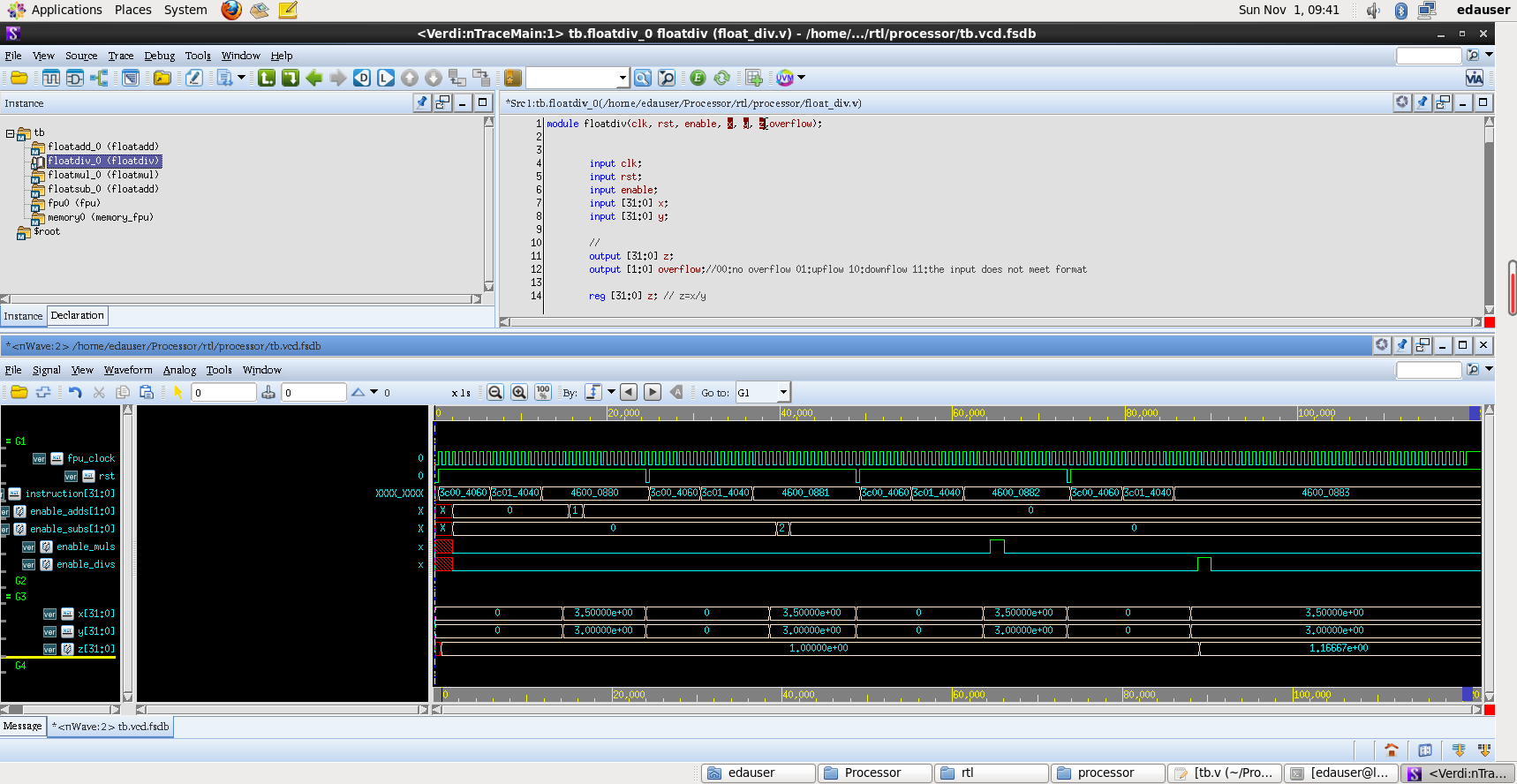
Fpu的add.s完成！



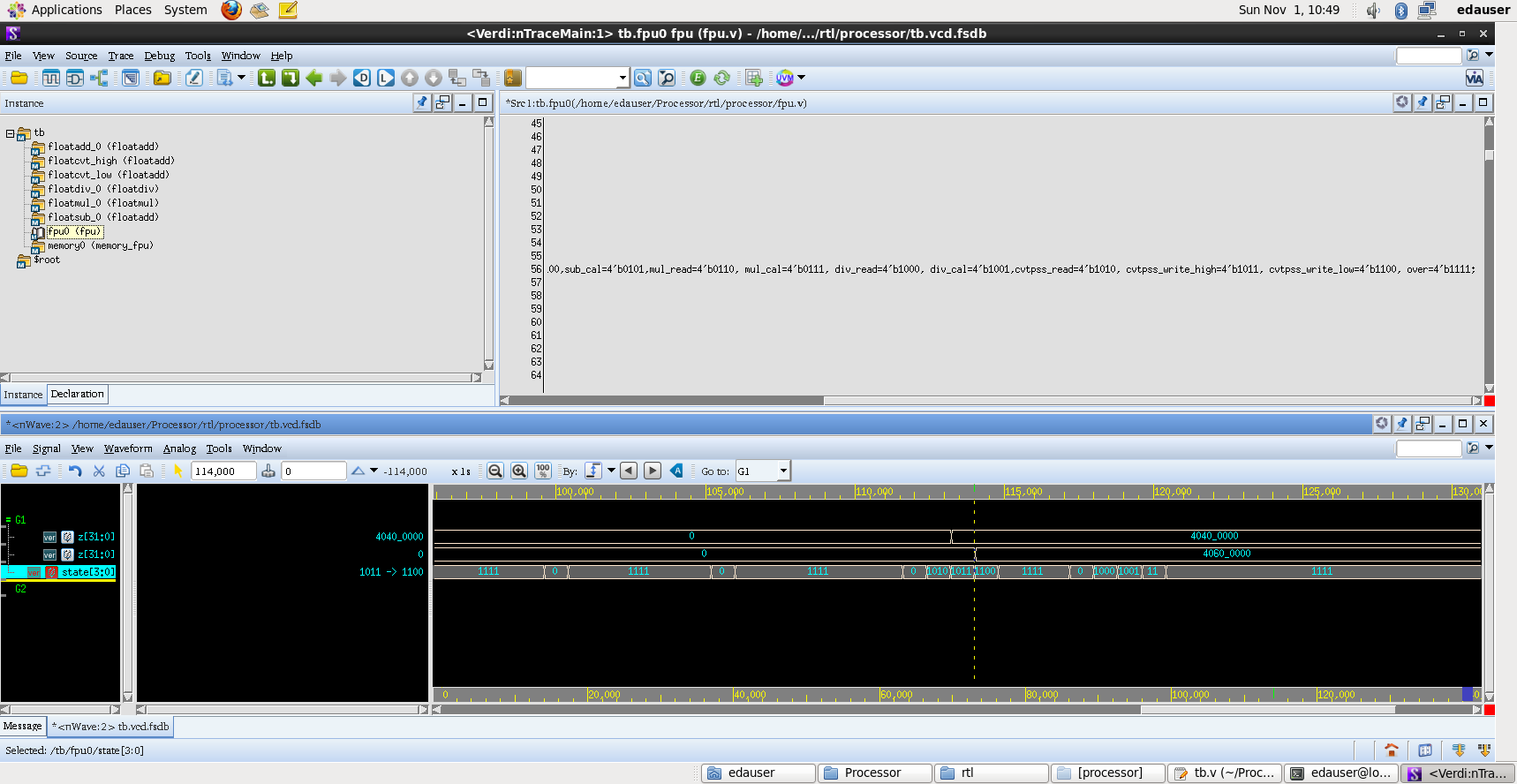
Fpu的sub.s完成！



Fpu的mul.s完成！



Fpu的divs完成！

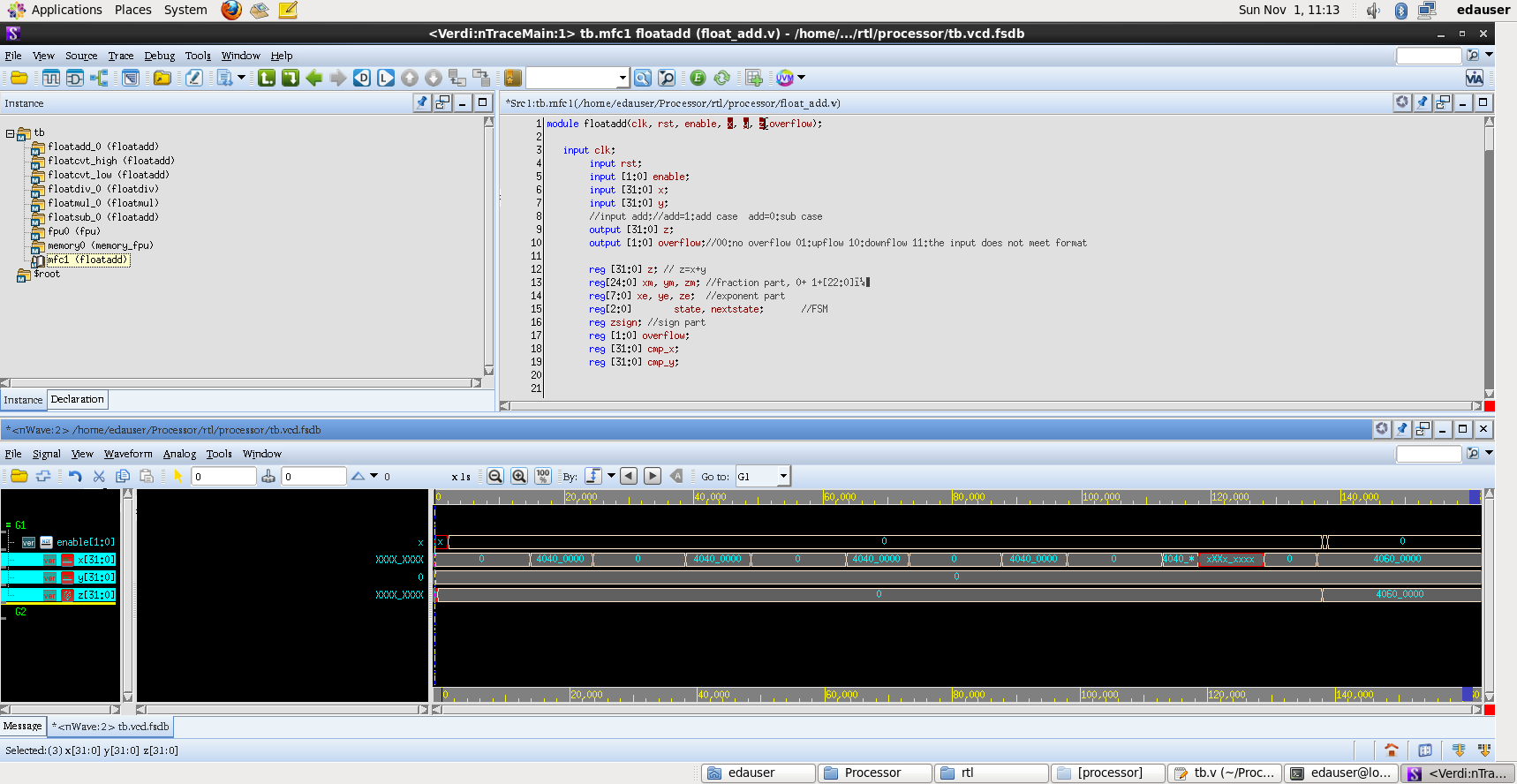


Fpu.cvtpss完成

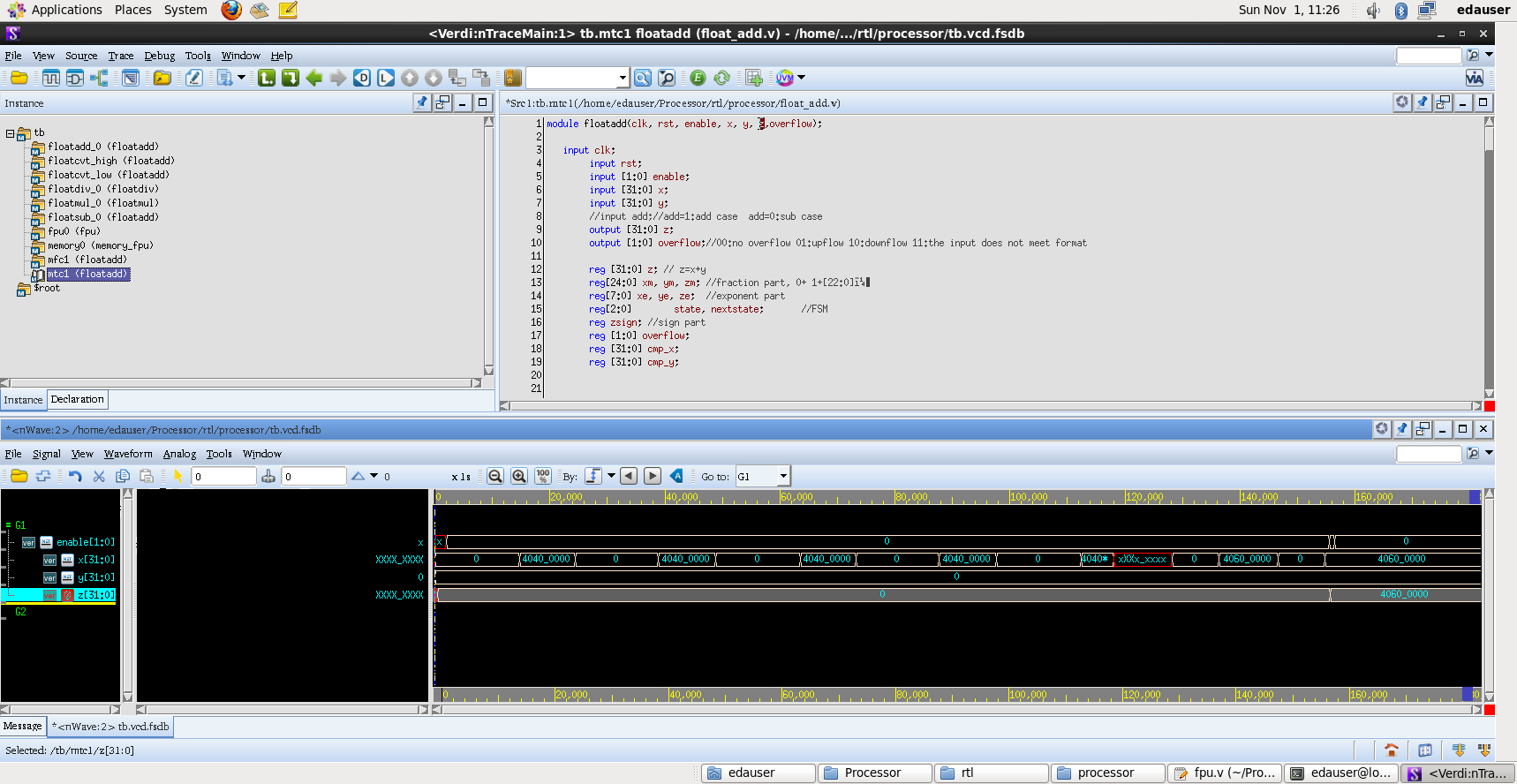
发现问题：内存的写入端口接了后面一堆单元的数据出口，会出现不定态

预计解决办法：或门+严格控制运算单元输出在不是自己工作时间内为0

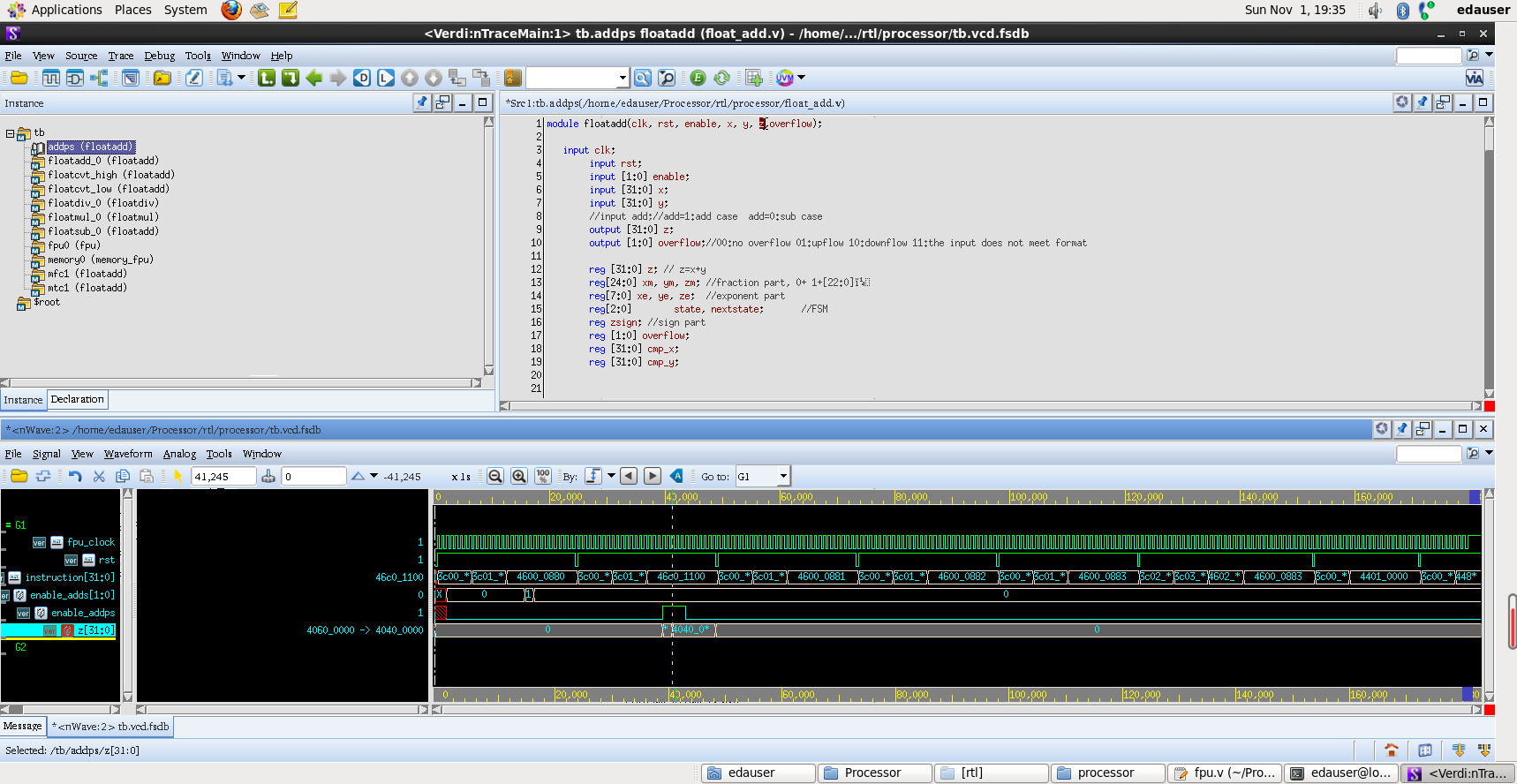
11.1



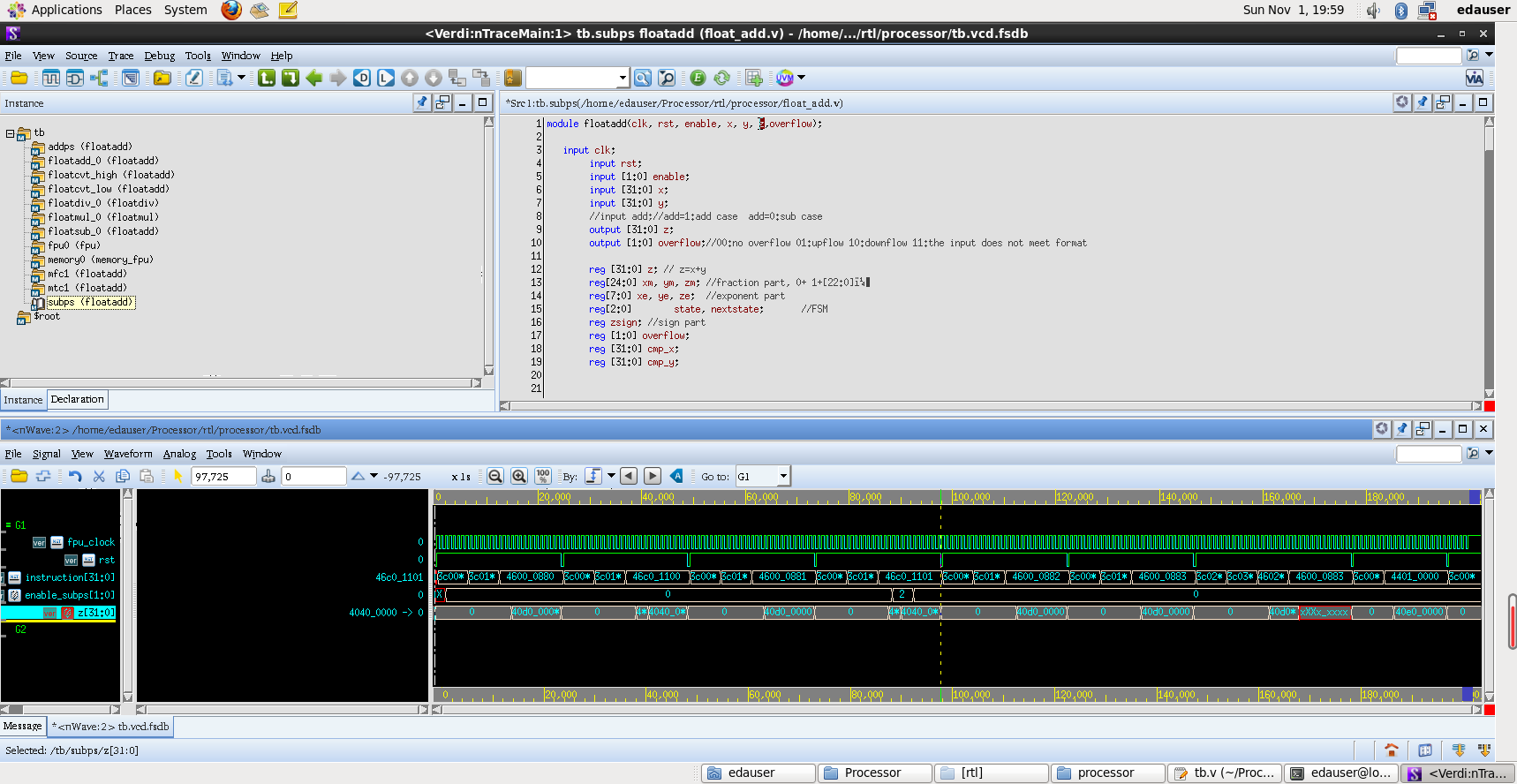
Mfc1完成



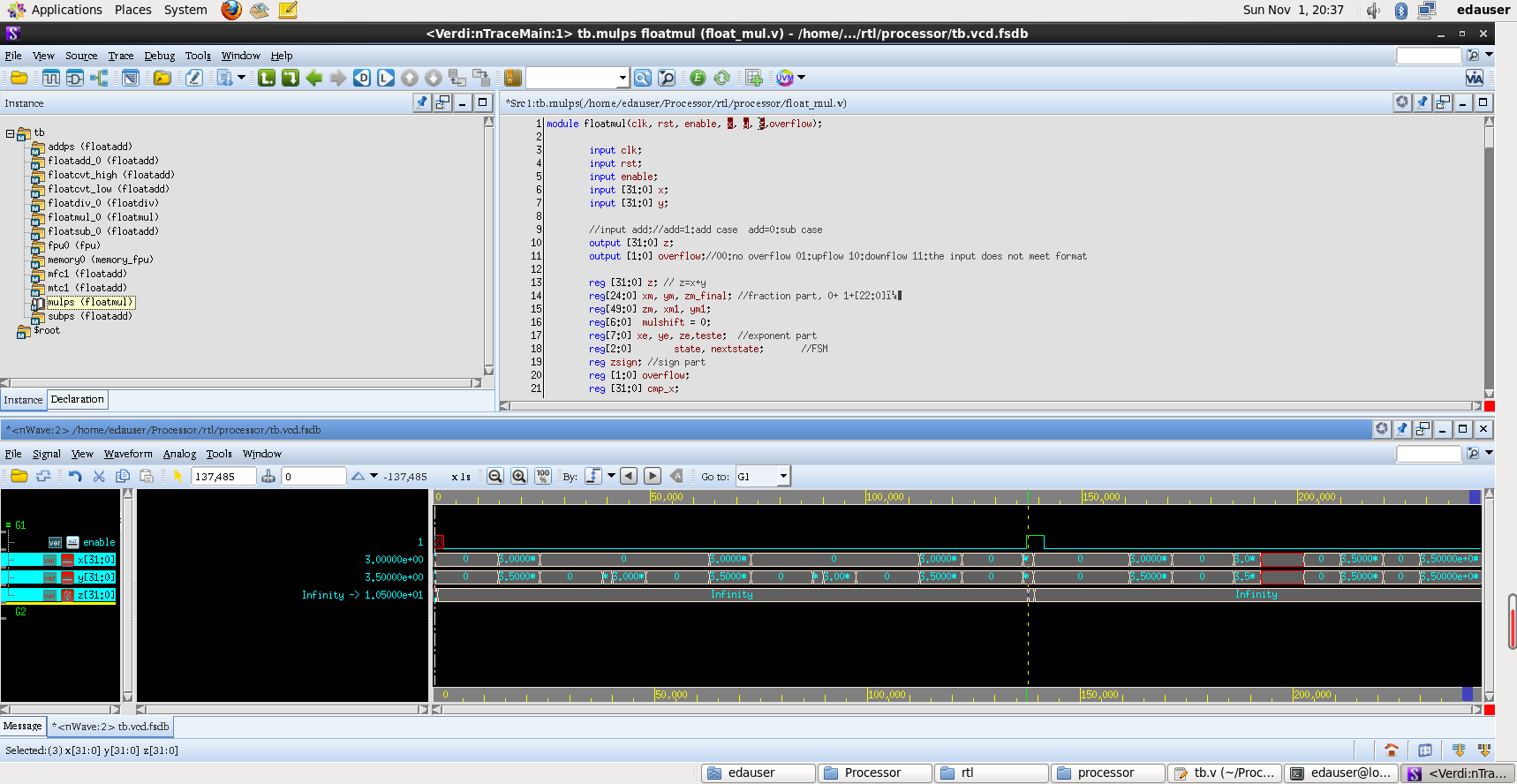
Mtc1完成



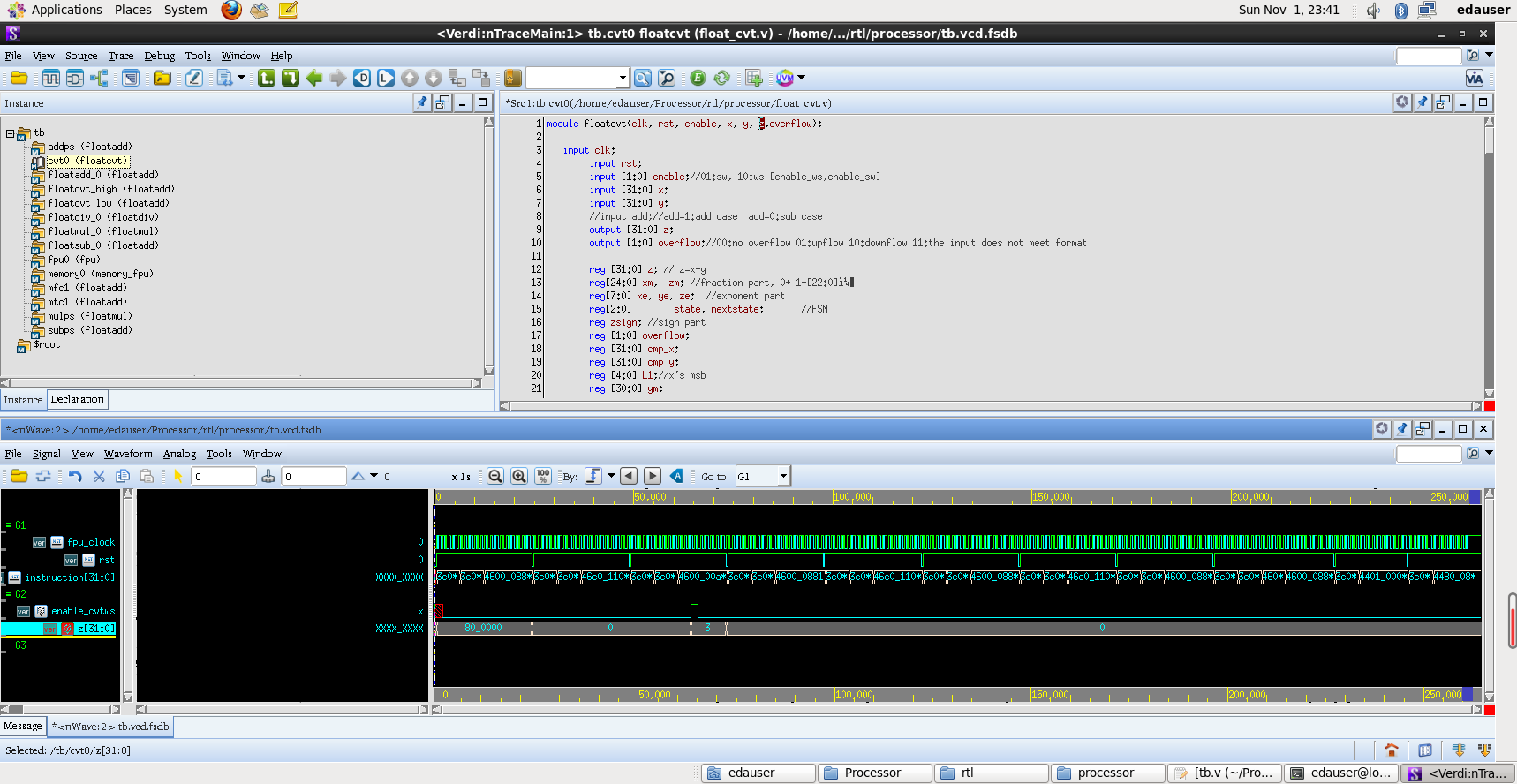
Addps完成！



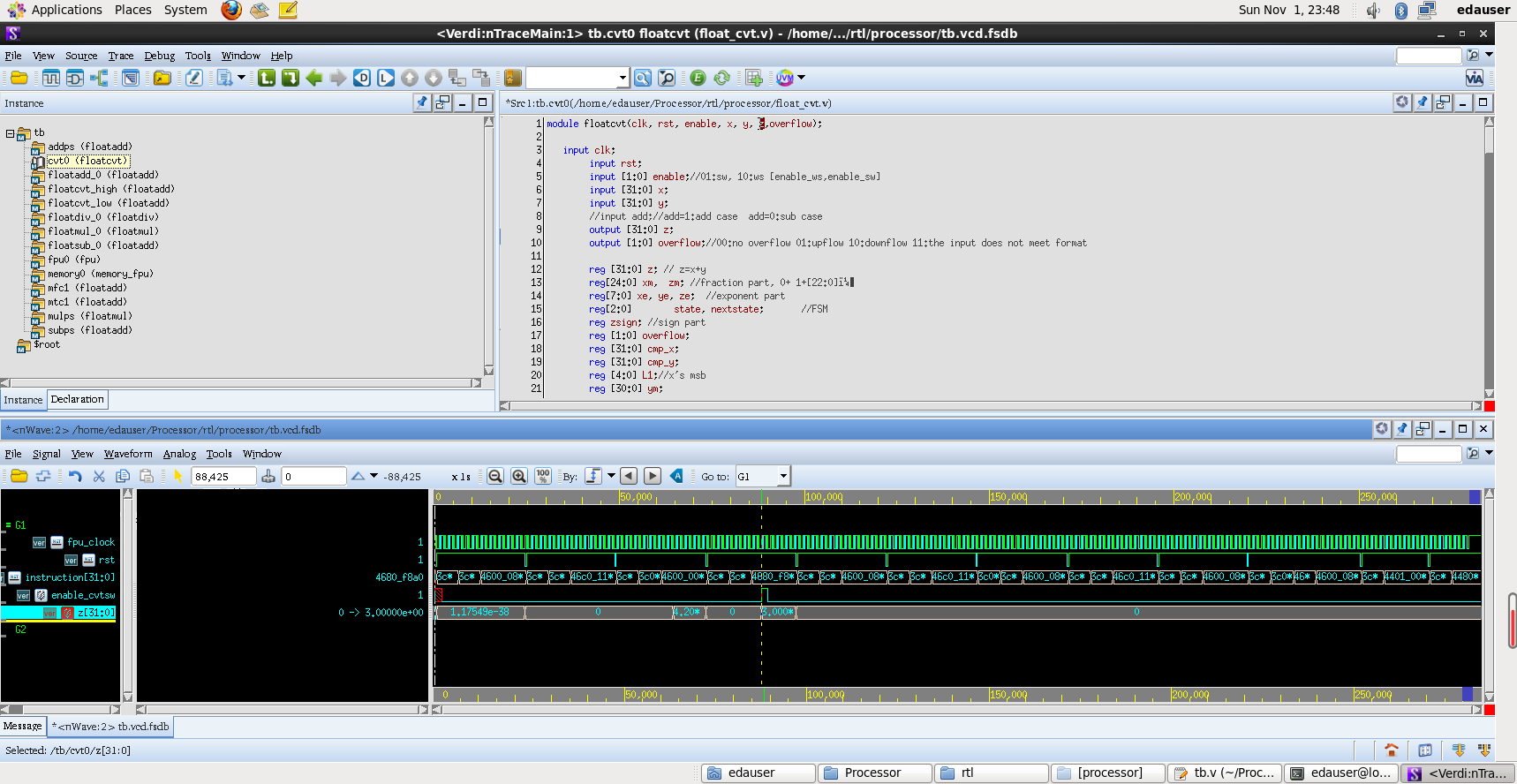
Subps完成！



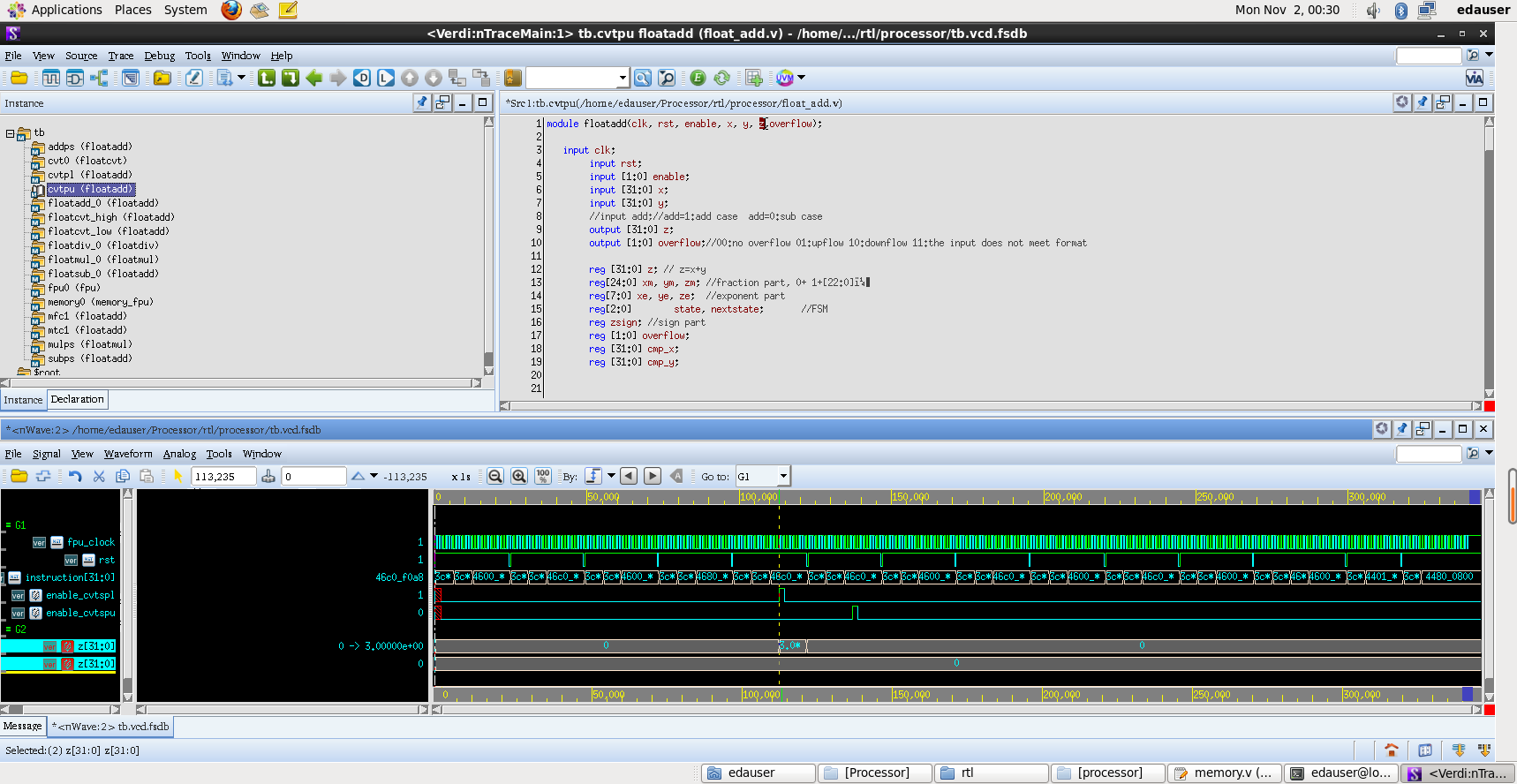
Mulps完成！



Cvtws完成！



Cvtsw完成！



Cvt.s.pl cvt.s.pu完成！

至此，指令都完成

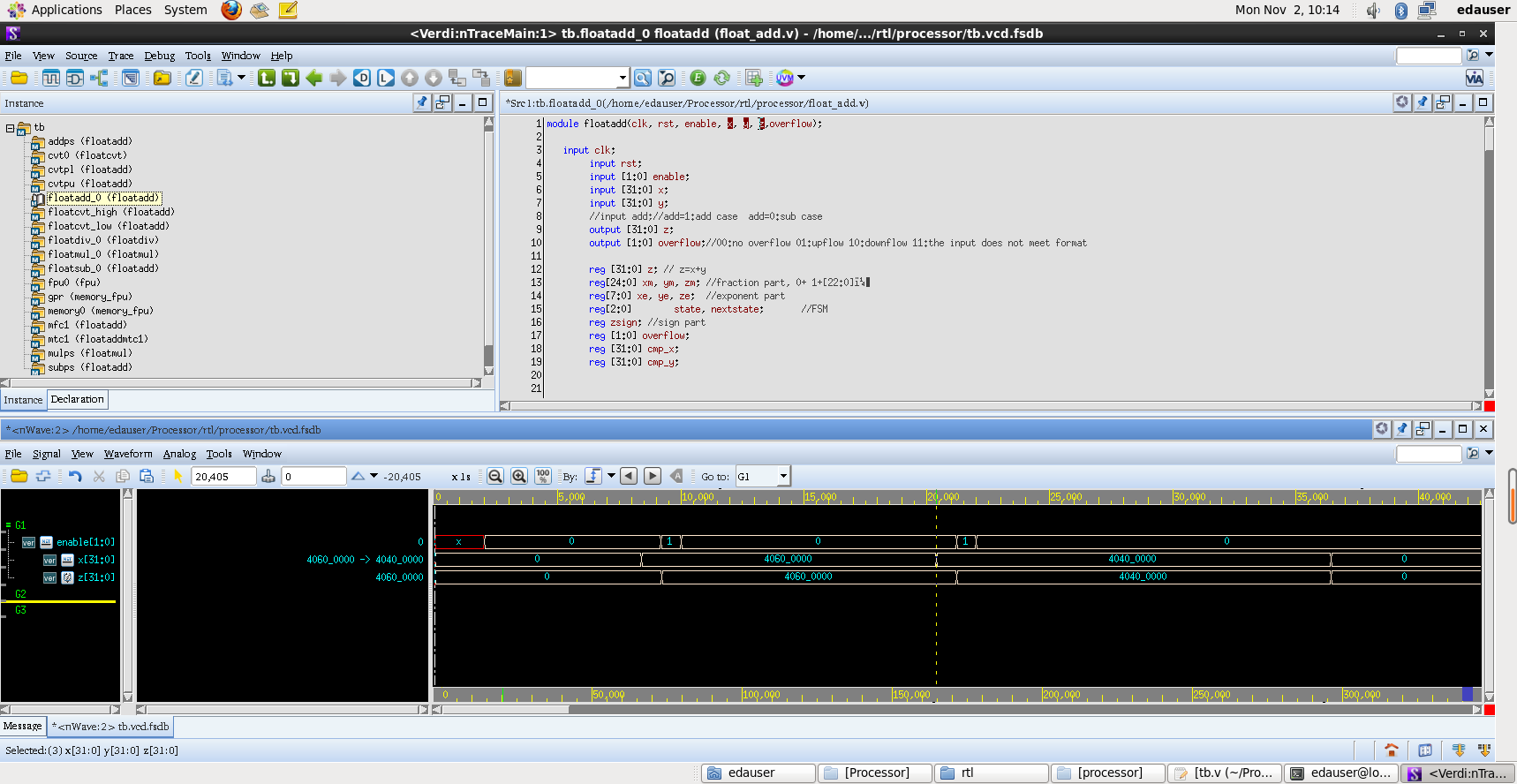
问题和解决办法：

1.mfc1和mft1再加一个gpr进行运算

2.将所有的输出写一个MUX，否则会出现信号冲突

明天最后一天，加油！

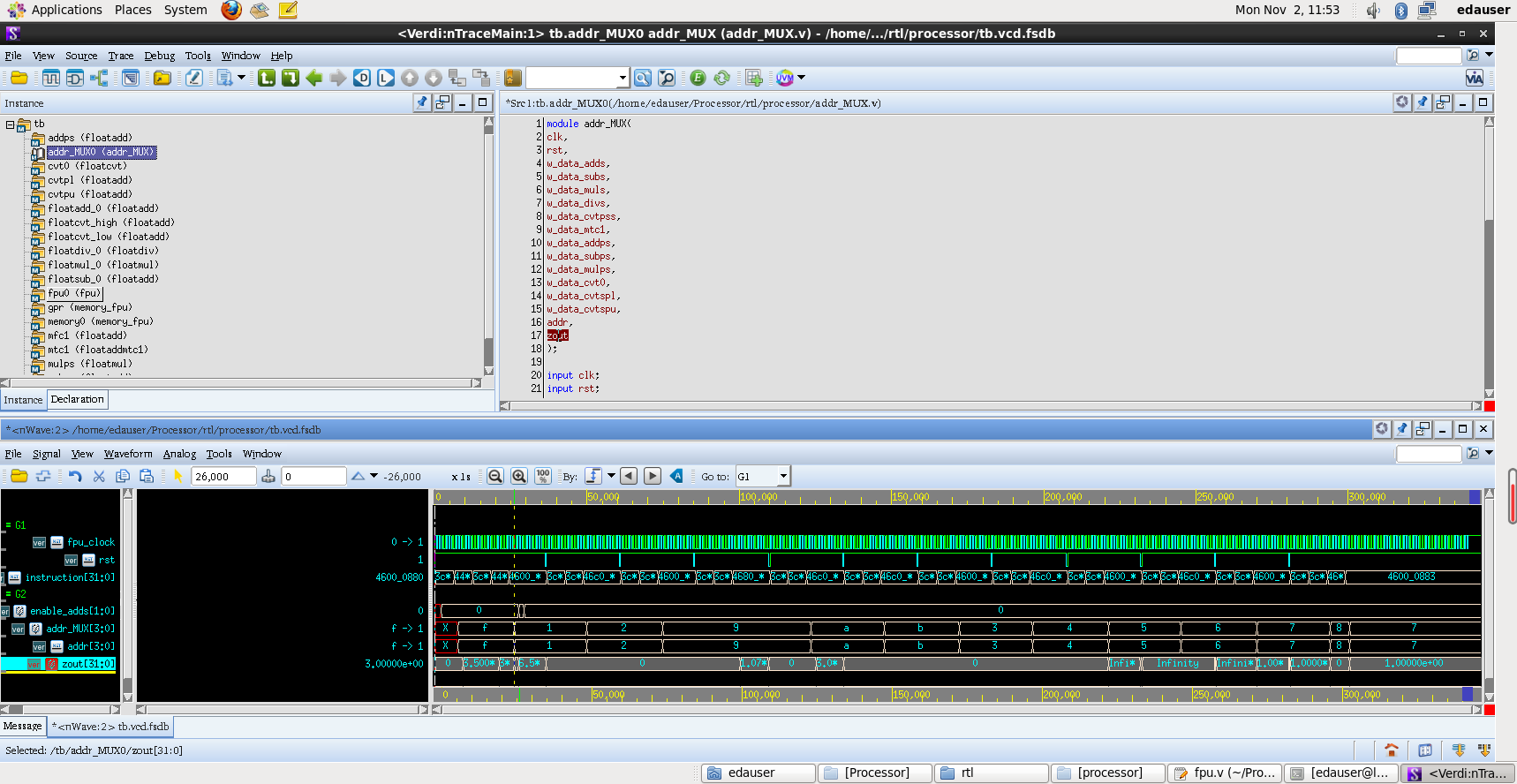
11.2



经过修改，gpr成功导入！

剩下问题：

输出选择



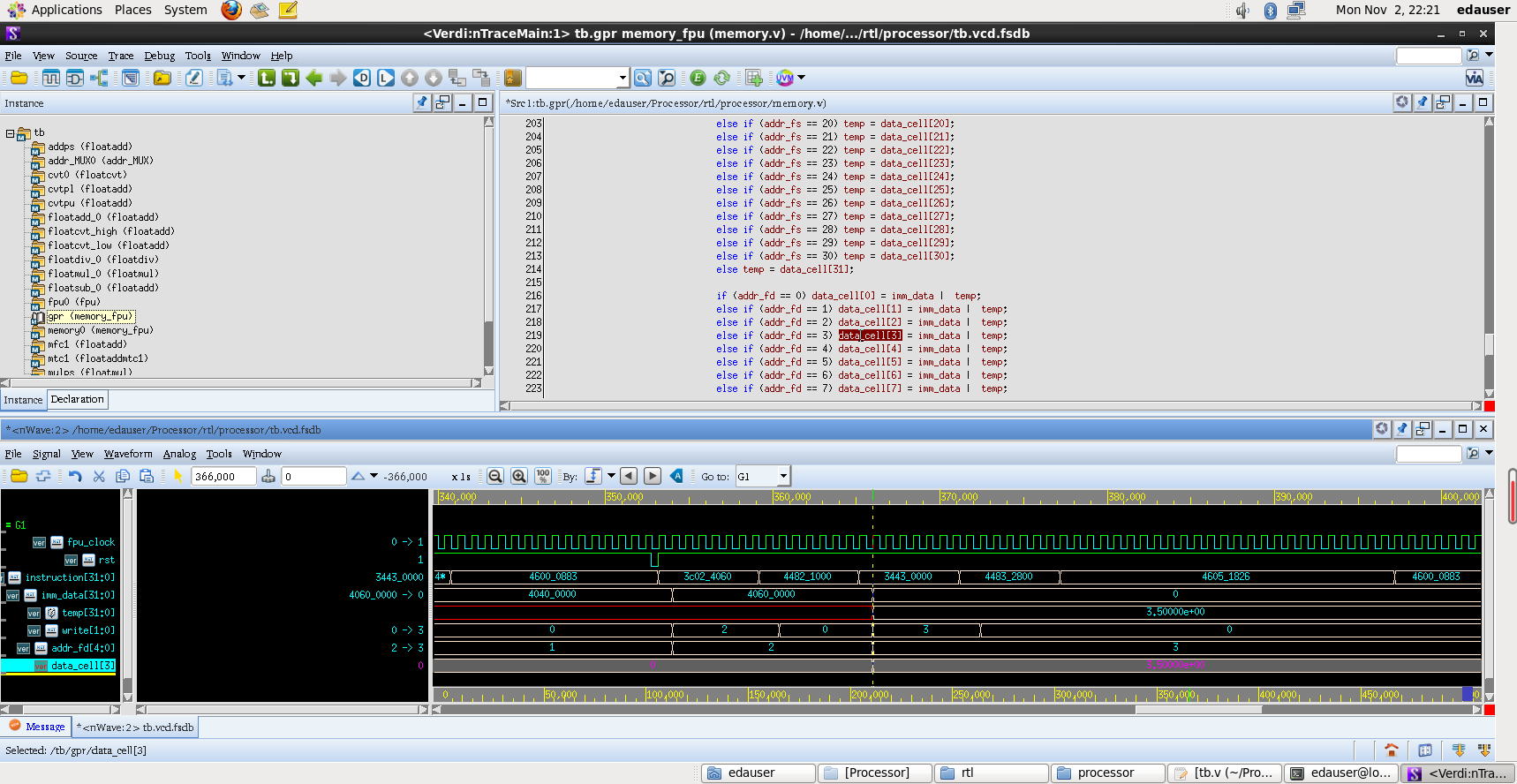
解决！！！

发现bug：

Cmp未重置，导致可能出现新一轮运算时系统判断指令未改变，故不做运算。

解决方法：

在！rst时重置为32‘b1（upflow）



调试技巧：查看内存波形