

（深圳）

实验报告

开课学期： 2021秋季

课程名称：数字逻辑设计（实验）

实验名称： 十六进制计算器设计

实验性质： 综合设计型

实验学时： 6 地点： T2615

学生班级： 2001105班

学生学号： 200110513

学生姓名： 宗晴

评阅教师：

报告成绩：

实验与创新实践教育中心制

2021年12月

注：本设计报告中各个部分如果页数不够，请大家自行扩页，原则是一定要把报告写详细，能说明设计的成果和特色。报告中应该叙述设计中的每个模块。设计报告将是评定每个人成绩的重要组成部分（**设计内容及报告写作**都作为评分依据）。

|  |
| --- |
| 设计的功能描述 |
| 概述基本功能、详细描述自行扩展的功能  基本功能：该实验设计了一个十六进制的计算器，能够支持加、减、乘、求商、求余、平方六种运算，同时连续运算的功能。当按下开关按键S1时，计算器可实现异步复位功能。操作数1由拨码开关SW15-SW8输入，操作数2由拨码开关SW7-SW0输入。运算功能由拨码开关SW23-SW21决定。对应功能如下图：  IMG_256  开关按键S2相当于等号功能，按下时会根据拨码开关所示数值进行对应运算并将结果按照十六进制输出到数码管上，按照高位补0的格式进行显示。第一次运算时，对输入的两个操作数进行运算，其中平方运算为第一个操作数的平方。连续运算时，以上一次的结果和操作数2作为输入数据进行运算，其中平方运算为上一次结果的平方。  若所要输出的数据大于可表示范围，则进行截断并显示。若所要输出的数据为负数，则采用补码进行表示。  自行扩展的功能：增加了按键消抖的功能。由于开关按键在实际操作中，在按下前会出现抖动，因此会有多次从0变为1的情况，这会使得运算被多次执行，最终出现错误的结果。因此，需增加按键消抖的功能来解决这种情况。在本实验中，我选择了增加计数器（即计时器）的方法来实现该功能，只有当按键被持续按下达到100000个周期（即10ms）时，才被记作按下一次。该方法可有效地解决按键抖动造成多次计算、输出错误结果的问题。 |
| 系统功能详细设计 |
| 用硬件框图描述系统主要功能及各模块之间的相互关系  硬件框图如下：  IMG_256  系统包括四个模块，分别为clk\_div、calculator\_hex、calculator\_display以及button\_vib，由顶层模块calculator\_top控制。整个系统输入端为clk、button、位宽为3的func、位宽为8的num1和num2、rst，输出端为位宽为8的led\_en、led\_cx（其中x为a,b,c,d,e,f,g）及led\_dp。输出端控制数码管显示计算结果。  首先模块clk\_div将原100MHz的时钟信号clk分频成10MHz的时钟信号clk\_g，提供给模块calculator\_hex、calculator\_display和button\_vib使用。而后模块button\_vib按照新的时钟信号将button消抖后生成新的按键信号button\_new，传给模块calculator\_hex和模块calculator\_display。模块calculator\_hex按照新的时钟信号和按键信号执行计算功能，将输入的num1和num2或者上轮数据按照func所对应的运算功能进行运算。最终将运算结果cal\_result输入模块calculator\_display。模块calculator\_display将接受到的运算结果按照十六进制显示到数码管上，输出led\_en及led\_cx来控制数码管的显示情况。 |
| 各模块描述 |
| 包括模块功能，输入、输出端口、变量含义及主要设计代码   1. 顶层模块calculator\_top：   IMG_256  功能：调用四个底层模块clk\_div、calculator\_hex、calculator\_display以及button\_vib，使其信号相互连通，通过复位信号、启动信号等控制这四个模块工作。  IMG_256  主要设计代码：实例化四个模块  IMG_256 IMG_256   1. 模块clk\_div：   IMG_256  功能：将100MHz的时钟信号clk分频成10MHz的时钟信号clk\_g并输出。  IMG_256   1. 模块button\_vib：   IMG_256  功能：在分频后的时钟信号控制下，将原始带有抖动的启动信号button消抖，生成新的启动信号button\_new并输出。防止出现因为启动按键抖动产生多次重复计算而输出错误结果的情况。通过运用计数器，规定只有当按键被按下满10ms时，才视为被真正按下，从而区分按键按下与抖动的区别。  IMG_256  主要设计代码：  首先实现计数器功能，只有当button被连续按下时，计数器才会进行累加，最终保留在最大值100000。按照10MHz的时钟周期，100000个周期即为10ms。  IMG_256  尽管按键有多次抖动，但计数器只可能有一次达到99999，可以认为达到此值时，按键确实被按下。由此实现按键消抖功能。  IMG_256   1. 模块calculator\_hex：   IMG_256  功能：在分频后的时钟信号控制下，根据消抖后的启动信号进行计算功能。运算包括加、减、乘、求商、求余、平方六种，同时支持连续运算的功能。第一轮运算时，两个操作数为输入的num1和num2，按照func所选择的运算功能进行运算，其中平方运算为第一个操作数的平方。连续运算时，两个操作数为上一轮结果和num2，按照func所选择的运算功能进行运算，其中平方运算为上一轮结果的平方。将运算结果cal\_result（用二进制表示）输出。  若所要输出的数据大于可表示范围，则进行截断并显示。若所要输出的数据为负数，则采用补码进行表示。  IMG_256  主要设计代码：  首先利用flag变量记录当前是否为第一轮运算（即两个操作数是num1和num2还是上一轮数据和num2）  IMG_256  若是第一轮运算，则对num1和num2进行运算。  IMG_256  否则，则对cal\_result和num2进行运算。  IMG_256   1. 模块calculator\_display：   IMG_256  功能：在分频后的时钟信号控制下，根据消抖后的启动信号进行显示功能。将接受到的二进制运算结果转换为十六进制显示到数码管上（每四位二进制数表示一位十六进制数），输出led\_en及led\_cx来控制数码管的显示情况。led\_en控制亮灯的位置，led\_cx控制亮灯位置显示的数字（即该位置数码管的亮暗方式）  IMG_256  中间变量：  IMG_256  主要设计代码：  首先，每四位不同的二进制数对应了一位十六进制数，而该十六进制数又对应了一种数码管亮暗方式，因此可以直接将每四位二进制数对应到一种数码管亮暗方式。  IMG_256  其次，由于数码管的八位显示位置每次仅点亮一个位置，因此需要有计数器控制位置的轮换，使肉眼看来八个位置每次都是同时点亮的。  IMG_256  还需控制复位时数码管全灭。并且按下button之后，直到复位前，数码管均需保持点亮状态，而button为按键开关，仅有一下为1，因此需设置变量ena记录此时数码管是否应该被点亮，即是否处在按下button后和按下rst前的区间。  IMG_256  然后当计数器达到上限时，数码管从后往前轮换一位，每位显示的数字对应原用二进制表示的计算结果的四位。该四位又通过上述模块对应到相应的数码管表示方式。从而实现控制数码管显示计算结果。  IMG_256 |
| 调试报告 |
| 仿真波形截图及仿真分析  IMG_256  如上图，首先从整体来看，初始时，复位按键被按下，此时计算结果保持00000000（16进制表示），且led\_en为ff表示没有数码管被点亮，符合预期。  而后，在7.195us前一个周期，button被按下，同时输入num1和num2分别为06和04（十六进制表示），此时func为000，表示加法运算。经过两个周期，计算出结果，cal\_result变为0000000a（十六进制表示），符合预期。  在8.795us前一个周期，button被第二次按下，由于过程中没有复位，因此此时应进行连续运算，即用上一轮的计算结果0000000a与输入的num2操作数0c（十六进制表示）进行运算，此时func为010，表示乘法运算。经过两个周期，计算出结果，cal\_result变为00000078（十六进制表示），符合预期。  在10.395us前一个周期，button被第三次按下，由于过程中没有复位，因此此时应进行连续运算，即用上一轮的计算结果00000078与输入的num2操作数05（十六进制表示）进行运算，此时func为101，表示平方运算，即对上一轮的计算结果00000078平方。经过两个周期，计算出结果，cal\_result变为00003840（十六进制表示），符合预期。  在11.995us前一个周期，button被第四次按下，由于过程中没有复位，因此此时应进行连续运算，即用上一轮的计算结果00003840与输入的num2操作数c8（十六进制表示）进行运算，此时func为001，表示减法运算。经过两个周期，计算出结果，cal\_result变为00003778（十六进制表示），符合预期。  在13.595us前一个周期，button被第五次按下，由于过程中没有复位，因此此时应进行连续运算，即用上一轮的计算结果00003778与输入的num2操作数08（十六进制表示）进行运算，此时func为011，表示求商运算。经过两个周期，计算出结果，cal\_result变为000006ef（十六进制表示），符合预期。  在15.195us前一个周期，button被第六次按下，由于过程中没有复位，因此此时应进行连续运算，即用上一轮的计算结果000006ef与输入的num2操作数00（十六进制表示）进行运算，此时func为101，表示平方运算，即对上一轮的计算结果000006ef平方。经过两个周期，计算出结果，cal\_result变为00301321（十六进制表示），符合预期。  IMG_256  将第一个周期放大分析：在7.195us前一个周期，button被按下，同时输入num1和num2分别为06和04（十六进制表示）。func为000，表示加法运算。此时cal\_result仍为00000000，led\_en仍为ff，表示数码管还未被点亮。经过两个周期，即7.395us时，计算出结果，cal\_result变为0000000a（十六进制表示），符合预期。同时led\_en变为fe，控制最后一个位置的数码管开始点亮，此时7个数码管中只有cg为高电平，显示数字为0，同时switch为0，可以看出数字比位置滞后一拍，符合预期。7.695us时，led\_en控制点亮位置开始前移，此时switch变为a，一周期后，7个数码管只剩cd为高电平，显示数字为a，符合预期。而后，7.995us时switch变回0，一周期后，7个数码管只剩cg为高电平，显示数字为0，符合预期。接着，8.295us、8.595us时，led\_en不断控制点亮位置前移，同时显示数字保持为0，符合预期。此周期内，数码管从最后一个位置开始依次向前轮换，表示数字为0000000a，符合预期。  IMG_256  将第二个周期放大分析：在8.795us前一个周期，button被第二次按下，同时输入num1和num2分别为00和0c（十六进制表示）。func为010，表示乘法运算。由于此时为连续运算，因此操作数为上一轮结果0000000a和num2表示的0c。经过两个周期，即8.995us时，计算出结果，cal\_result变为00000078（十六进制表示），符合预期。同时led\_en变为fe，控制最后一个位置的数码管开始点亮，此时7个数码管中只有cg为高电平，显示数字为0，同时switch为0，可以看出数字比位置滞后一拍，符合预期。9.295us时，led\_en控制点亮位置开始前移，此时switch变为8，一周期后，7个数码管全为低电平，显示数字为8，符合预期。而后，9.595us时switch变为7，一周期后，7个数码管中ca、cb、cc为低电平即被点亮，显示数字为7，符合预期。然后，9.895us时switch变回0，一周期后，7个数码管只剩cg为高电平，显示数字为0，符合预期。接着，10.195us时，led\_en不断控制点亮位置前移，同时显示数字保持为0，符合预期。此周期内，数码管从最后一个位置开始依次向前轮换，表示数字为00000078，符合预期。  由此可知，calculator\_hex模块实现了十六进制计算器的功能。 |
| 设计过程中遇到的问题及解决方法 |
| 在设计过程中，我遇到了一些问题：   1. 数码管显示乱码，且数码管亮度不一致：经研究，我发现导致该问题的原因是，在成功通过仿真之后上板时，未修改数码的管轮换周期，导致几个数字叠加显示在同一个位置，因而显示乱码，且部分数码管亮度不一。在延长轮换周期后，我成功解决了该问题，数码管显示正常。 2. 数码管显示数字错位，如本应显示0000000a，却显示了000000a0：经研究，我发现导致该问题的原因是，我的显示模块中将显示功能分为了两步完成，因此产生了滞后。将显示步骤合并之后，数码管显示正确。 3. 数码管输出结果错误，且每次输出值不确定：经研究，我发现这是由于按键抖动引起的，按键抖动导致模块接收到多个信号，从而进行了多次计算。在增加了按键消抖模块之后，数码管输出结果正确。 |
| 课程设计总结 |
| 包括设计的总结和还需改进的内容以及收获  在这次实验中，我实现了一个完整的十六进制计算器。在设计过程中，我运用了自顶向下的结构化设计方法，设计了顶层模块、四个底层模块，也亲自设计了四个模块之间的交互关系和接口情况。在设计过程中，我用到了之前所学过的时钟IP、计数器模块、数码管显示模块等，整合了之前所学的内容，也对数字逻辑设计有了一个整体的理解。这次实验让我体会到模块化设计的简洁清晰，实验的顺利完成也给我带来了很大的成就感。  就数字逻辑实验课程整体而言，前期确实比较难熬，虽然课堂上讲过一些基础知识，但由于当时没有实际操作的机会，因此理解不深。因而具体开始实验时，有一些无从下手，但很有幸实验课的老师和助教都非常耐心地解决我们的问题。最初的几节课，在课后都要花好几个小时自己进行摸索。在向同学请教了整整一天后，我才终于对verilog语言以及vivado的使用有了一个较为清晰的理解。所以后续的实验也因此进行的还算顺利。个人希望在最初的几节实验课上，老师能先带领我们读懂几份完整的代码，在此过程中跟我们讲解verilog的语法以及注意事项，或许会比在课堂上直接讲述verilog的语法更直观形象，当前课堂上对于verilog语法的讲解稍显零碎，也不易记住。同时，希望能增加实验课每次实验的课时，可以适当减少大家在课后的压力。  数字逻辑实验课程让我学会了verilog语言的使用，也收获了模块化设计的思想，学会了用vivado进行仿真、上板实验等，为后续课程的学习打下了基础。 |