第1部分：命令规则	每个文件只包含一个module，module名要小写，并且与文件名保持一致
	除parameter外，信号名全部小写，名字中的两个词之间用下划线连接
	由parameter定义的常量要求全部字母大写，自己定义的参数、类型用大写标识
	推荐用parameter来定义有实际意义的常数，包括单位延时、版本号、板类型、单板在位信息、LED亮灯状态、电源状态、电扇状态等
	信号名长度不超过20字符
	避免使用Verilog和VHDL保留字命令
	建议给信号名添加有意义的前缀或后缀，命名符合常用命名规范（_clk 或clk_表示时钟， n表示低电平有效， z表示三态信号， en表示使能控制，rst 表示复位
	保持缩写意义在模块中的一致性
	同一信号在不同层次应该保持一致性
第2部分：注释	每个文件有一个文件头，文件头中注明文件名、功能描述、引用模块、设计者、设计时间、修改信息及版权信息等
	对信号、参量、引脚、模块、函数及进程等加以说明，便于阅读与维护，如信号的作用、频率、占空比、高低电平宽度等
	用“//”做小于1行的注释，用“/*　*/”做多于1行的注释
	更新的内容要做注释，记录修改原因，修改日期和修改人
第3部分：模块	module例化名用u_xx_x标示
	建议每个模块加timescale
	不要书写空的模块，即一个模块至少要有一个输入和一个输出
	为了保持代码的清晰、美观和层次感，一条语句占用一行，每行限制在80个字符以内，如果较长（超出80个字符）则要换行
	采用基于名字（name_based)的调用而非基于顺序的（order_based)的调用
	模块的接口信号按输入、双向、输出顺序定义
	使用降序列定义向量有效位顺序，最低位是0
	管脚和信号说明部分，一个管脚和一组总线占用一行，说明清晰
	不要采用向量的方式定义一组始终信号
	逻辑内部不对input进行驱动，在module内不存在没有驱动源的信号，更不能在模块端口存在没有驱动的输出信号，避免在elabarate和compile时产生warning，干
	在顶层模块中，除了内部的互连和module的例化外，避免在做其他逻辑
	出于层次设计和同步设计的考虑，子模块输出信号建议用寄存器
	内部模块端口避免inout，尽量在最顶层模块处理双向总线
	三态逻辑可以在顶层模块使用，子模块中禁止使用三态。如果能确保该信号不会被其它子模块使用，而是直接通过顶层模块输出要I/O口，可以在子模块中使用三态
	没有未连接的端口
	为逻辑升级保留的无用端口以及信号要注释
	建议采用层次化设计，模块之间相对独立
	对于层次化设计的逻辑，在升级中采用增量编译
第4部分：线网和寄存器	不允许锁存器和触发器在不同的always块中赋值，造成多重驱动
	出于功能仿真考虑，非阻塞赋值应该增加单位延时，尤其是对于寄存器类型的变量赋值时；阻塞赋值不允许使用单位延时
	always语句实现时序逻辑采用非阻塞赋值；always语句实现的组合逻辑和assign语句块中使用阻塞赋值
	同一信号赋值不能同时使用阻塞和非阻塞方式
	不允许出现定义的parameter/wire/reg没有使用
	建议不使用integer类型寄存器
	寄存器类型的信号要初始化
	除移位寄存器外，每个always语句只对一个变量赋值，尽量避免在一个always语句出现多个变量进行运算或赋值
第5部分：表达式	在表达式内使用括号表示运算的优先级

	一行中不能出现多个表达式
	不要给信号赋“x”态，以免x值传递
	设计中使用到的0，1，z等常数采用基数表示法书写（即表示为1'b0,1'b1,1'bz或十六进制）
	端口申明、比较、赋值等操作时，数据位宽要匹配
第6部分：条件语句	if 都有else和它对应
	变量在if-else或case语句中所有变量在所有分支中都赋值。如果用到case语句，写上default项
	禁止使用casex
	case语句item必须使用常数
	不允许使用常数作为if语句的条件表达式
	条件表达式必须是1bit value。如异步复位，高电平有效使用“if(asynch_reset==1'b1)”，低电平有效用“if(asynch_reset==1'b0)”，不要写成“if(! asynch_reset)”或者“if(asynch_reset==0)”
	不推荐嵌套使用5级以上if…else if…结构
第7部分：可综合的	不要使用include语句
	不要使用disable、initial等综合工具不支持的电路，而应采用复位方式进行初时化。但在testbench电路中可以使用
	不要使用specify模块
	不要使用===、！==等不可综合的操作符
	除仿真外，不要使用fork-join语句
	除仿真外，不要使用while语句
	除仿真外，不要使用repeat语句
	除仿真外，不要使用forever语句
	除仿真外，不要使用系统任务($)
	除仿真外，不要使用deassign语句
	除仿真外，不要使用force,release语句
	除仿真外，不要使用named events语句
	不要在连续赋值语句中引入驱动强度和延时
	禁止使用trireg型线网
	制止使用tri1、tri0、triand和trior型的连接
	不要位驱动supply0和supply1型的线网赋值
	设计中不使用macro_module
	不要在RTL代码中实例门级单元，尤其下列单元：CMOS/RCOMS/NMOS/PMOS/RNMOS/RPMOS/trans/rtrans/tranif0/tranif1/rtranif0/rtranif1/pull_gate
第8部分：可重用的	考虑未使用的输入信号power_down，避免传入不稳定态
	接口信号尽量少，接口时序尽量简单
	将状态机（FSM）电路与其它电路分开，便于综合和后端约束
	将异步电路和同步电路区分开，便于综合和后端约束
	将相关的逻辑放在一个模块内
	合理划分设计的功能模块，保证模块功能的独立性
	合理划分模块的大小，避免模块过大
	在设计的顶层（top）模块，将I/O口、Boundary scan电路、以及设计逻辑（core logic)区分开
第9部分：同步设计	在一个module中，在时钟信号的同一个沿动作。如果必须使用时钟上升沿和时钟下降沿，要分两个module设计
	在顶层模块中，时钟信号必须可见，不要在模块内部生成时钟信号，而是使用DCM/PLL产生的时钟信号

	避免使用门控时钟和门控复位
	同步复位电路，建议在同一时钟域使用单一的全局同步复位电路；异步复位电路，建议使用单一的全局异步复位电路
	不要在时钟路径上添加任何buffer
	不要在复位路径上添加任何buffer
	避免使用latch
	寄存器的异步复位和异步置位信号不能同时有效
	避免使用组合反馈电路
	always有且仅有一个的敏感事件列表，敏感事件列表要完整，否则可能会造成前后仿真的结果不一致。异步复位情况下需要异步复位信号和时钟沿做敏感量，同步复位情况下只需要时钟沿做敏感量
	时钟事件的表达式要用“negedge<clk_name>”或“posedge<clock_name>”的形式
	复杂电路将组合逻辑和时序逻辑电路分成独立的always描述
第10部分：循环语句	在设计中不推荐使用循环语句。在非常有必要使用的循环语句时，可以使用for语句
第11部分：约束	对所有时钟频率和占空比都进行了约束
	对全局时钟skew进行了约束
	对于时序要求的路径需要针对特殊要求进行约束，如锁相环鉴相信号
	需要根据输出管脚驱动要求进行约束，包括驱动电流和信号边沿特性
	需要根据输入和输出信号的特性进行管脚上下拉约束
	针对关键I/O是否约束了输入信号和输入时钟的相位关系，控制输入信号在CLK信号之后或之前多少ns到达输入pad
	综合设置时，fanout建议设置为30
	需要使用输入输出模块中的寄存器，如Xinlinx公司的IOB，map properties选项pack I/O register/latches into IOBsactor需要设置成为“for input and output”，这样可以控制管脚到内部触发器的延时时间
	布局布线报告中IOB、LUTs、RAM等资源利用率应小于百分之八十
	对于逻辑芯片对外输入接口，进行tsu/th约束；对于逻辑芯片对外输出接口，进行
第12部分：PLL/DCM	如果使用FPGA内部DCM和PLL时，应该保证输入时钟的抖动小于300ps，防止DCM/PLL失锁；如果输入时钟瞬断后必须复位PLL/DCM
	对于所有厂家的FPGA，其片内锁相环只能使用同频率的时钟信号进行锁相。如果特殊情况下需要使用不同频率的信号进行锁相，需要得到厂家的认可，以避免出时钟
第13部分：代码编辑	由于不同编辑器处理不同，对齐代码使用空格，而不是tab键