本系统采用结构化的建模方式，依据需求文档将项目功能划分为手动模式与自动模式两大功能，分别设计对应的Top模块并统一接口，并使用Designed Top将两模块功能合并，整体的项目结构如下

图形用户界面

描述已自动生成

其中，对于手动模式功能的实现，我们将Manual Top模块设计为一个两状态状态机：状态一为轮询状态，状态二为发送状态，其状态切换关系如下

图示

描述已自动生成

Manual Top模块的结构示意如下

图示

描述已自动生成

在初始阶段，Manual Top模块将被置为状态一，此时该模块将会轮询其所例化的多个用于解析用户的不同操作的Encoder模块，如果用户执行了游戏操作，则对应的Encoder会将该操作依据通讯协议编码为对应指令传出，并产生一个时钟周期的激活脉冲信号，Manual模块将会在轮询中接收到这个脉冲并切换至状态二。

由于需要发送较长时钟周期的指令信号才能确保该信号被客户端接收与处理，因此在发送状态时，Manual Top模块将启动一个计数器用以向客户端发送指定时钟周期的指令信号，在计数结束后，该模块状态将会切换为状态一，以继续进行操作的轮询。

为实现对非法用户操作的过滤，Manual Top模块例化了Filter模块用以过滤非法操作，Filter模块将分析从客户端所接收的反馈信号以判断当前用户操作产生的指令信号是否合法，并对非法指令信号进行拦截，从而实现了对非法用户操作的过滤

对于脚本模式功能的实现，我们将Script Top模块设计为三状态状态机：状态一为读取下一条指令，状态二为分析当前脚本，状态三为发送指令，其状态切换关系如下

图示

描述已自动生成

Script Top模块的结构示意如下

日程表

描述已自动生成

在初始阶段，Script Top模块将被置为状态一，此时该模块将会依据处是否处在手动单步调试判断是否加载下一条脚本，如加载下一条脚本，则切换至状态二，否则维持当前状态，等待切换操作的发生。

对于状态二，该状态将依据feedback反馈与用户所处位置分析当前脚本所需执行的操作，如为指令操作则切换至发送状态，并且记录hasNext参数该脚本是否已经执行完成。如为非指令操作（如jump指令切换脚本地址）则返回状态一。

对于状态三，该状态将会发送指定时钟周期的传输信号，在发送结束后将依据上一条脚本是否执行完成切换状态。

基于上述设计，脚本模式实现了较为清晰的状态切换与需求划分，避免了在单一文件中分析所有脚本造成的状态切换复杂，具有一定的可扩展性。

Design Top与Demo Top模块为上述模块的顶层模块，其模块关系如下

图形用户界面, 应用程序

描述已自动生成

而对于Design Top模块，该模块例化了Manual Top 与Script Top模块，该模块将使用Target Register存储目标机器的所处位置，并依据外部反馈信号进行手动模式与自动模式的切换，发送对应模块的反馈信号。Demo Top模块则例化了Designed Top模块，使用所提供的URAT模块与Script模块实现脚本读取与 客户端-开发板之间的交互，同时例化了时钟分频与消抖模块以提供合适且稳定的时钟与按键信号。

-----------下面有Bonus

在Bonus部分的手动单步调试与自动脚本模式任意切换中，得益于Script Top模块较为清晰的状态设计，我们能够较为清晰方便地实现该功能，其实现为在切换脚本状态内部添加一个小状态机，用于判断是否为单步调试状态，如为单步调试状态则判断是否加载下一条指令以进行外部状态机的切换，如为自动运行状态则直接加载下一条指令，该实现避免了对外部状态机的直接改动，从而避免了对外部模块的代码改动，使该功能具备较好的稳定性与兼容性，其状态机设计如下

图示, 维恩图

描述已自动生成