数字逻辑项目文档：GenshinKitchen

黄政东
祝超
何俞均

1 团队分工

1.1 团队分工

- 贡献比 `1:1:1`
- 详细工作安排见下表。

黄政东	project 顶层模块的处理（manualTop, ScriptTop 以及 DemoTop）
	实现接收并用 LED 显示来自客户端的四个反馈信号
	所有模块的代码规范性检查
	实现在手动模式/自动 (脚本) 模式之间进行切换
	实现脚本单步调试以及手动模式/自动 (脚本) 模式之间的切换
	部分文档攥写工作
祝超	project 手动模式的合法性检查
	project 脚本模式中 jump，wait 语句相关模块执行的处理
	各种需要的测试脚本的准备，高效的脚本设计
	loadingLamp 模块的编写
	部分文档攥写工作
何俞均	project 手动模式的按键到机器码部分（buttonDecoder 模块）
	project 脚本模式中 Action、Game State Instruction 部分的处理
	使用七段数码管显示信息
	错误脚本状态自动处理
	部分文档攥写工作

表 1: 详细工作安排

1.2 开发计划日程安排和实施情况

开发计划安排

- 11 周讨论 `project` 实现方式，大致模块安排以及粗略设计实现方式。

- 12 周完成 manual 模块的所有子模块，一人进行 manualTop 模块的编写及调试，另外两人进行脚本模块的编写。
- 13 周脚本模块编写完成，一人进行 scriptTop 模块的编写及调试，一人负责准备脚本，另一人负责完成一些 bonus 中的任务。
- 14 周基本完成所有模块，解决一些发现的 bug，并对一些对用户不太友好的操作进行更加人性化的调整。
- 15 周测试所有模块，继续解决遇到的 bug，准备答辩。
- 15 周答辩。

实际开发情况

- 11.30: 开会讨论设计大致模块。决定设计 DesignedTop，ManualTop，ScriptTop，TargetRegister。并先完成 DesignedTop，ManualTop，TargetRegister。
- 12.2: DesignedTop，TargetRegister 完成。设计 ManualTop 中包含的 TargetStateMachine，TargetStateEncoder，GameStateEncoder，OperationEncoder，ManualFliter。
- 12.10: ManualTop 完成设计 ScriptTop 中的 GameStateScriptHandler，ActionScriptHandler，JumpScriptHandler，WaitScriptHandler。
- 12.17：ScriptTop 完成准备 Bonus 中的 ScriptFixer 以及快速脚本。
- 12.23：ScriptTop 中出现 bug，没修好。
- 12.26: ScriptTop 修好，快速脚本完成。ScriptFixer 拼不上去。
- 12.27: ScriptFixer 修不好，决定放弃。决定整理代码于 12.29 答辩。
- 12.28：加入走马灯，整理代码完成。

2 系统功能列表

2.1 手动模式

1. 通过按钮，玩家可以进行实现：

    - `game start/end`：分别对应了拨码 `0/1`。
    - `get`：右侧左边按钮。
    - `put`：右侧下边按钮。
    - `move`：右侧中间上面按钮。
    - `interact`：右侧中间按钮。
    - `throw`：右侧右边按钮。
    - `change target machine`：2 号拨码开关为顺时针移动一格，3 号为逆时针移动一格。

2. 对于开始和结束游戏：

    - 开始游戏：拨动拨码 0，可以开始游戏。玩家可以通过按钮进行各种操作。如果当前正处于脚本模式，则会直接进入手动模式。
    - 结束游戏：先拨动拨码 1，再回拨拨码 0，可以结束游戏。这是为了防止拨码接触不良以及玩游戏时误启动 `game end`，设计的“双保险”。

3. 可以自动过滤掉非法的操作：

    - 开发板能够阻止移动时 (玩家未在机器跟前时) 的非法交互。
    - 开发板能够阻止不合理的存取物品交互。
    - 开发板能够阻止不合理的投掷食材操作 (只可以投掷到桌子/垃圾桶)。
    - 开发板能保证操作信号是 `One-Hot` 编码的。

2.2 脚本模式

1. 通过向上波动拨码 0 可以实现手动模式与脚本模式的切换。
2. 通过输入脚本,可以实现 `get`,`put`,`interact`,`throw`,`wait`,`waituntil`,`jump`，`jumpif` 等操作。
3. 脚本模式中，4、5 拨码同时向上拨启动单步调试，随后下拨上拨一次拨码 5 执行下一条指令。

3 系统使用说明

使用说明已在功能列表中阐述。
在开发板上的实际按钮如下图所示：

4 系统结构说明

5 子模块功能说明

5.1 人工模式

5.2 脚本模式

6 bonus 的实现说明

6.1 高效脚本设计

设计理念：

1. 考虑到一些及其操作是全自动的，因此操作者实际上可以利用这段时间去做一些别的事情，以节省时间。
2. 由于操作者的移动事实上是比较耗时的，因此在脚本中尽量避免了操作者长距离的跑动，若是需要运输物品这样的操作，如果能丢到附近的桌子上，那么操作者就不会亲自跑过去一趟。

执行完成时间：12.6s 左右。

7 项目总结

在硬件开发方面，我们团队积累了丰富的经验。通过对不同的硬件模块进行研究和开发，我们深入了解了各种硬件的特性和使用方法，并能够根据需求进行相应的选择和调整。在实际的硬件开发过程中，我们熟练掌握了各种工具和技术，并能够快速解决遇到的问题。同时我们在调错中深入理解了硬件编程与软件编程的区别，培养了硬件编程的思维。

在团队合作方面，我们团队具备良好的沟通和协作能力。通过合理的任务分配和提供清晰的接口文档，我们能够有效地协调各个组员的工作，确保整个项目顺利进行。此外，我们定期召开团队会议，及时交流项目进展和存在的问题，并寻求共同解决方案。

在开发工作中，我们使用 github 平台进行版本控制和团队协作。通过合理的分支管理和代码审查，我们能够有效地协同完成项目的不同部分，并保证代码质量和可维护性。同时我们除了使用 Vivado 进行代码编辑，还安装了 VScode 中的插件对 verilog 进行编辑。

在测试工作中，我们采用了仿真测试的方法，对每个模块进行了全面的测试，并及时修复发现的问题。通过不断完善测试流程和测试用例，我们能够保证项目的质量和稳定性。同时我们在测试代码的过程中讲变量绑到 led 灯中进行查看，这帮助我们发现了不少问题的原因。例如我们在编写 Wait 模块时，我们等待的时长一直不稳定，通过仿真测试，我们发现我们的一个变量的预期返回结果只会持续一个时钟周期，我们才修复了 bug；在编写脚本执行时，我们通过 led 灯发现 ActionScriptHandler 模块传入的 feedback 因为硬件原因位移了一位，使得 Get 指令执行不了。这两种测试方法给予了我们很大的帮助。

8 其他的想法和建议

1. 首先，可以将几个之前的项目保留，作为祖传项目，每年做一点修改然后发布，这样不至于在项目上线后还有各种 bug 和问题，以后的同

学们做起来也会舒服一点。（但是这样可能就要加强一下作弊的管控，谨防抄袭往年作品）

2. 关于新题目，可以做俄罗斯方块等一些小游戏。这样可以有人实现 VGA 的接口，同时助教也可以提供 UART，（俄罗斯方块和这次厨房的 project 类似，有游戏结束和开始、旋转方块、处理不同类型方块、消除已经可以消除的方块等模块，还有对于不同方块降落下来的 feedback，以及方块下降时的等待语句等，同时还可以设计通过七段数码管显示分数、得到最高分数/以最快的时间执行完某种固定的下降模式等 bonus），这样在加强同学们对于硬件语言的理解时也不至于过于复杂。