电子电路基础软件板块设计方案

1. 项目概述

名称：ourCircuit Simulator（教学用电子电路仿真器）

目标：开发一款支持基础电路仿真与交互式学习的软件，通过项目实践培养团队协作、软件工程和电路知识应用能力。

二、技术架构

核心语言：Python 3.12

GUI框架：TKinter

仿真引擎：NumPy（数学计算）+ Matplotlib（可视化）

AI辅助：Deepseek R1（主要使用）/ChatGPT o3-mini/ Gemini 2.0 / Claude 3.7

版本控制：Git + GitHub

1. 功能模块与团队合作
2. 核心算法组

核心算法与功能，搭建基本框架，按照课程相应内容逐渐添加器件或功能。

1. GUI开发组

开发GUI部分，电路的可视化与具体操作，简洁美观。

1. 测试协调组

测试各板块的协调性与准确性，反馈程序可能出现的问题并提出优化建议。

1. 学生组织具体方案
2. 共同部分：

进行开发流程的规划讲解与基本工具的使用教程，确保简洁易懂，具有可操作性。

1. 分组部分：

适当对班上40余名同学进行分组，可进行大小组的分组方案（三个大组，每个大组分2-3个小组完成不同任务/完成相同任务然后总合择优）

1. 任务布置/完成奖励：

每节课余20-30分钟选取上次程序的贡献者进行简要讲解，同时提出相应新的编写任务，召集小组讨论，听取想法与提出建议。

每次任务完成提交至指定平台，由测试协调组的同学进行下载与测试，反馈相应模块运行情况。

程序贡献者可请求老师合理加平时分，学期结束赠送礼品（实体礼品/虚拟礼品）。

四、开发流程

第1周：项目启动与基础准备

目标：统一认知、分配任务、学习工具

周一：

项目启动会：讲解项目目标、功能模块与开发流程。

分组与角色分配：

核心算法组（15人，分3小组：基础元件/动态元件/分析算法）

GUI开发组（15人，分3小组：界面框架/元件绘制/交互逻辑）

测试协调组（10人，分2小组：功能测试/集成测试）

基础工具培训：Git基础、Python代码规范、tkinter基础模板。

周三：

核心算法组：学习电路分析算法理论（如节点电压法），设计元件数据结构（JSON/类结构）。

GUI组：绘制界面原型图（Figma/白板），确定元件图标风格。

测试组：编写测试计划模板，学习单元测试框架（如unittest）。

第2周：框架搭建与模块设计

目标：完成核心框架与界面原型

周一：

核心算法组：定义元件基类（电阻、电源等），实现基础电路数据结构（节点、支路）。

GUI组：用tkinter搭建主窗口（功能区+左右面板），实现工具栏按钮布局。

测试组：评审数据结构与界面原型，提出可测试性建议。

周三：

核心算法组：实现线性电阻的节点解析法（矩阵方程求解）。

GUI组：完成元件区的图标绘制（Canvas+Button），支持元件拖拽占位符。

测试组：编写线性电阻的单元测试用例。

第3-4周：核心功能迭代

目标：实现基础元件与基础分析功能

周一/周三（第3周）：

核心算法组：

扩展电源元件（直流电压源、电流源）

实现直流分析的节点电压法完整逻辑

GUI组：

实现画布元件放置与连线功能（Canvas绑定点击事件）

添加保存/打开功能（序列化电路数据为JSON）

测试组：

验证节点电压法计算结果与理论值一致性

测试画布元件连线的正确性

周一/周三（第4周）：

核心算法组：

添加动态元件（电容、电感），设计瞬态分析接口

实现正弦稳态分析（复数阻抗计算）

GUI组：

支持元件参数编辑（双击弹窗输入阻值/容值等）

添加分析结果展示面板（表格+Matplotlib绘图）

测试组：

测试正弦稳态分析的幅频特性曲线

第5-6周：高级功能扩展

目标：添加非线性元件与复杂分析

周一/周三（第5周）：

核心算法组：

实现二极管模型（分段线性近似）

搭建小信号分析框架（工作点计算+AC分析）

GUI组：

添加元件旋转/删除功能

实现分析参数配置面板（如AC分析频率范围）

测试组：

验证二极管IV特性曲线是否符合预期

周一/周三（第6周）：

核心算法组：

添加三极管模型（简化Ebers-Moll模型）

实现开关电路动态分析（状态方程数值求解）

GUI组：

支持多级撤销/重做功能

优化频响特性图交互（缩放/标注关键点）

测试组：

压力测试大规模电路（100+元件）的性能

第7-8周：集成测试与优化

目标：模块联调与用户体验提升

周一/周三（第7周）：

核心算法组：封装算法为API，提供异常处理（如电路不连通）。

GUI组：实现实时错误提示（如短路/开路警告）。

测试组：组织全班交叉测试，收集崩溃与性能问题。

周一/周三（第8周）：

全体：

Bug修复冲刺：优先处理关键路径问题（如矩阵求解失败）

性能优化：算法复杂度分析、画布渲染加速

GUI组：添加帮助文档与操作提示（Tooltip）。

第9周：交付与总结

目标：项目验收与经验复盘

周一：

最终演示：各组展示完整功能（如搭建运放电路并分析频响）。

用户手册编写：分小组撰写功能说明与操作指南。

周三：

代码归档：提交至GitHub，生成API文档（Sphinx）。

总结会：评选最佳贡献者，发放奖励（如定制U盘/虚拟勋章）。

关键协作机制

每日站会（异步）：各组使用在线文档每日更新进度/阻塞问题。

版本控制：强制使用Git分支开发，每周合并一次主分支。

AI辅助：鼓励使用Copilot生成模板代码（如tkinter事件绑定）。

激励制度：每周评选1名“代码之星”，额外奖励平时分。

五、AI应用指南

代码开发阶段：

使用ChatGPT进行：

/code 生成特定功能模板代码

/debug 解释错误信息并提供解决方案

/optimize 重构现有代码

测试阶段：

用AI生成边界测试用例

自动化生成测试报告

智能日志分析

六、教学评估方式

过程评估：

提交记录分析

成果评估：

核心功能完整度

代码可读性（AI静态分析）

用户测试反馈

创新性AI应用案例

建议在开发过程中建立：

AI提示词知识库（记录有效提示词）

代码质量看板（使用SonarQube）

持续集成报告（自动化生成）