# 项目进度安排

**阶段一：准备与环境配置 (2025.01 - 2025.02)**

**•目标**:

完成项目环境的搭建，包括虚拟化和编译环境。

确认龙芯环境下的基本工具链支持。

**•任务**:

1.在 QEMU 中配置龙芯 Loongarch64 虚拟机环境，并安装基本开发工具链（GCC、Clang 等）。

2.获取并准备 Nim 的源码（包括 Nim 主仓库和 csources_v2 仓库）。

3.阅读 Nim 编译器源码，特别是与架构相关的部分，了解现有 mips64le 的支持情况。

4.初步测试从 mips64le 配置编译的 Nim 生成的 C 代码在 Loongarch64 下的运行情况，确保能够执行基本命令（nim --version 等）。

**阶段二：伪交叉编译与初步移植 (2025.03 - 2025.05)**

**•目标**:

实现 Nim 编译器的伪交叉编译，生成 Loongarch64 版本的 Nim 编译器。

**•任务**

1.配置 Nim 编译器进行伪交叉编译，使用 mips64le 作为初始配置，编译为 C 代码。

2.在 Loongarch64 本地编译生成的 C 代码，获得 Nim 编译器。

3.针对初步移植过程中出现的编译错误，记录并分析错误原因，进行简单修正。

4.验证生成的编译器是否能在 Loongarch64 本地执行简单的 Nim 命令。

**阶段三：本地编译与深入源码修改 (2025.06 - 2025.08)**

**•目标**:

使用 Loongarch64 环境下生成的 Nim 编译器编译 Nim 源码，并解决出现的错误。

**•任务**:

1.使用 Loongarch64 环境下的 Nim 编译器重新编译 Nim 源码，生成本地架构版本的 Nim 编译器。

2.详细分析编译过程中涉及 Loongarch 架构的报错，定位并修改相关 C 代码与 Nim 源码。

3.提交初步的源码修改，记录每次修改后的测试结果。

4.与 Nim 社区保持沟通，了解其他架构移植过程中可能遇到的类似问题。

**阶段四：集成测试与优化 (2025.09 - 2025.10)**

**•目标**:

通过 Nim 语言的所有集成测试，并进一步优化源码以兼容 Loongarch64。

**•任务**:

1.使用 Loongarch64 版本的 Nim 编译器运行 Nim 的集成测试（nim testament all）。

2.收集并分析测试失败的用例，确定问题是否与 Loongarch 架构相关。

3.根据测试结果，修改对应的源码部分（可能涉及 Nim 编译器、C 代码和 Nim 标准库）。

4.在 Nim 社区中发布进度更新，并请求社区成员协助测试和提供反馈。

**阶段五：代码贡献与 Pull Request 提交 (2025.11 - 2025.12)**

**•目标**:

为 Nim 官方仓库提交代码贡献，并推动 Nim 对 Loongarch 架构的官方支持。

**•任务**:

1.整理所有修改过的源码文件和补丁，准备 Pull Request 提交给 Nim 的官方仓库。

2.在 GitHub 上提交 Pull Request，详细说明每个修改的原因和测试结果。

3.与 Nim 的维护者沟通，回答可能的代码审查问题，并根据反馈进行进一步调整。

4.在龙芯架构操作系统的包管理器中配置 Nim 语言包，并测试其安装和运行效果。

5.撰写项目总结报告，记录从伪交叉编译到本地编译、源码修改和测试的全过程。

**项目里程碑**

1. **2025.02**：完成 QEMU 虚拟环境搭建，并能够在 Loongarch64 上运行简单的 Nim 编译器命令。

2.**2025.05**：成功在 Loongarch64 环境下编译出初步可用的 Nim 编译器。

3.**2025.08**：通过 Nim 编译器的完整本地编译，并解决大部分架构兼容性问题。

4.**2025.10**：通过所有 Nim 语言集成测试，编译器性能与稳定性达到预期标准。

5.**2025.12**：完成代码提交，并获得 Nim 官方对 Loongarch 支持的认可；配置龙芯包管理器中的 Nim 安装包。