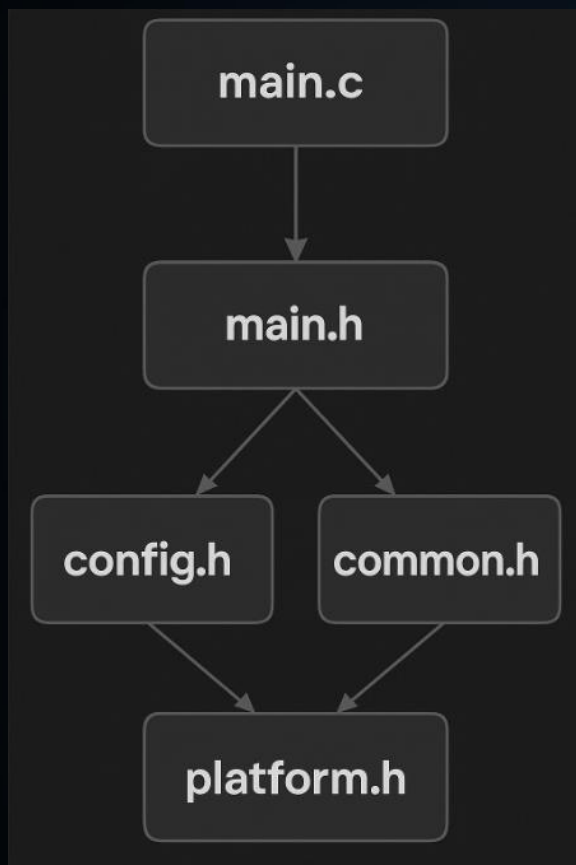


Makefile 增量构建之 头文件依赖链

自动化建立C代码头文件依赖链的原理

- 鱼丸ECU



头文件依赖链的前世今生

MAKEFILE GCC SCONS



GNU Make + GCC 解决方案

GNU Make GCC



构建演示

Demo

头文件依赖链的前世今生

✅ 一、早期阶段：手动依赖维护（Makefile 静态规则）

⌚ 特点：

- 使用 `Makefile` 明确写出依赖关系
- 程序员需手动管理 `.c` 与 `.h` 文件之间的关系

📦 示例：

```
make
```

[📄 复制](#) [✎ 编辑](#)

```
main.o: main.c main.h  
    gcc -c main.c
```

! 问题：

- 人工维护麻烦、易出错
- 无法追踪头文件间的间接依赖

头文件依赖链的前世今生

✅ 二、自动依赖生成：编译器辅助分析（GCC -M 系列）

🔧 背景：

随着项目变大，程序员不可能手动维护头文件依赖，于是 `gcc` 提供 `-M`、`-MM`、`-MD`、`-MMD` 等选项，自动生成依赖信息。

🎯 示例：

bash

📄 复制 ✎ 编辑

```
gcc -M main.c # 生成 Makefile 格式的依赖列表
```

✂️ 组合方式：

- `-MD`：自动生成 `.d` 文件，包含依赖信息
- `-MF file.d`：指定依赖输出文件名
- `-MP`：为每个依赖生成一个空规则，避免删除后make报错
- `-MT`：设置目标名称，配合模式规则使用

头文件依赖链的前世今生

起源与发展简要

初期背景

- 在早期 UNIX 编译系统中，Make 是主力构建工具，但它并不自动管理 `#include` 依赖。
- 用户需要手动写出所有 `.c` → `.h` 的依赖链，非常容易出错。

GNU 体系的解决方案

GNU 的 GCC 工具链 (GNU Make + GCC) 为了缓解这个问题，引入了：

技术	作用
<code>gcc -M / -MM</code>	输出 C/C++ 文件的头文件依赖
<code>-MD / -MMD</code>	编译时生成 <code>.d</code> 文件（依赖描述文件）
<code>make -include</code>	Makefile 中隐式包含 <code>.d</code> 文件，实现自动重编译

头文件依赖链的前世今生

✓ 三、结构化项目：工具链辅助构建（以 SCons 为例）

⌚ 特点：

- 构建逻辑用脚本（如 `SConstruct`）编写，声明即依赖
- 系统自动追踪 `#include`，不依赖 `.d` 文件
- 构建图显式建模，支持跨平台、增量、缓存
- 基于内容哈希追踪变更，不依赖时间戳

如何确保Scons系统解析的头文件和编译器实际一致？

1. Include path
2. 头文件中的 `#if` 宏解析

```
#ifdef xxx
#include <xxx.h>
#endif
```

手动依赖维护演示

```
#
# makefile 基本结构: rule(规则)
#
# target: prerequisite ..... 目标: 依赖
# → command(recipe) ..... 命令(配方)
#
# https://gcc.gnu.org/onlinedocs/gcc/Preprocessor-Options.html#Preprocessor-Options
#
# -M/-MM/-MD/-MMD/-MP/-MF [build/main.d] -MT [build/main.o]
```

```
# 手动管理头文件依赖 -- 方案1
build/main.o: src/main.c src/main.h
→ gcc -c src/main.c -o build/main.o
```

```
# 手动管理头文件依赖 -- 方案2
build/main.o: src/main.c
→ gcc -c src/main.c -o build/main.o

# 手动管理头文件依赖 -- 方案2
build/main.o: src/main.h
```

GNU Make + GCC 解决方案

```
#
# makefile 基本结构: rule(规则)
#
# target: prerequisite ..... 目标: 依赖
# → command(recipe) ..... 命令(配方)
#
# https://gcc.gnu.org/onlinedocs/gcc/Preprocessor-Options.html#Preprocessor-Options
#
# -M/-MM/-MD/-MMD/-MP/-MF [build/main.d] -MT [build/main.o]
```

```
# 自动管理头文件依赖 -- 方案3
build/main.o: src/main.c
→ gcc -c -MD -MP src/main.c -o build/main.o

# 加载 dependency -- 方案3
#include build/main.d
```

```
# 自动管理头文件依赖 -- 方案4
build/main.o: src/main.c
→ gcc -c -MD src/main.c -o build/main.o

# 加载 dependency -- 方案4
#include build/main.d

# 提升兼容性 -- 方案4
%.h: ;
```


构建演示

GNU Make + GCC 解决方案

头文件依赖演示

1. 首次构建
2. 修改头文件
3. 移动头文件
4. 删除头文件