

LT168 LVGL V95 SDK 编译教程

1. 简介

1.1 项目概述

- LT168 LVGL V95 SDK 是基于 LVGL v9.5 图形库的嵌入式开发套件，专为 LT168 系列 MCU 设计。
- 该 SDK 提供了完整的 LVGL 9.5.0 图形界面开发环境，包含多个演示程序和用户产品开发模板。

1.2 硬件平台

- **MCU 型号**: LT168B 系列
- **显示接口**: LCD (支持 RGB/RGB565 等格式)
- **触摸接口**: 电容触摸屏 (CTP) / 电阻触摸屏 (RTP)
- **存储介质**: eFlash、SPI NorFlash

1.3 文档目标读者

- 嵌入式软件工程师
-

2. 环境准备

2.1 开发环境要求

- **操作系统**: Windows
- **必要工具**: CCore IDE (包含交叉编译工具链)
- **下载地址**: https://www.levetop.cn/uploadfiles/2023/12/setup_CCore-20231227.zip

2.2 安装 CCore IDE

2.2.1 下载与安装

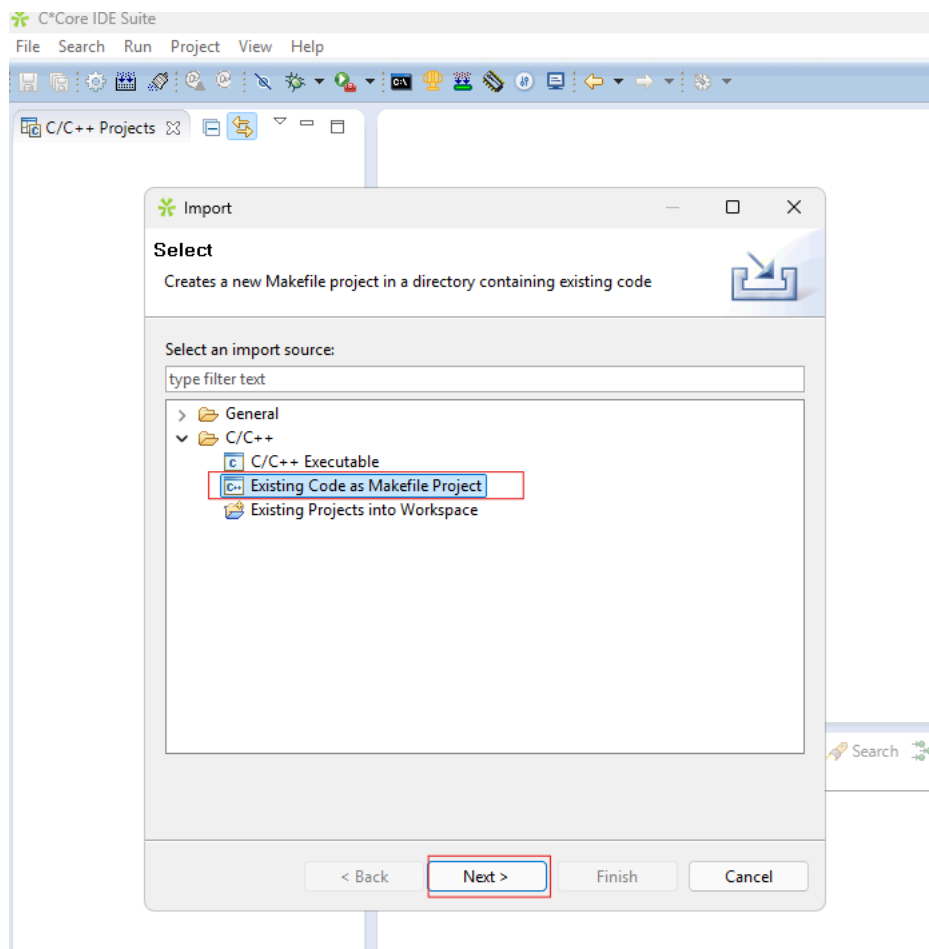
1. **下载**: 下载 CCore IDE 安装包。
2. **安装**: 解压并运行安装程序，按照提示完成安装。
3. **启动**: 安装完成后，启动 CCore IDE。

2.3 导入工程

步骤 1: 打开 CCore IDE

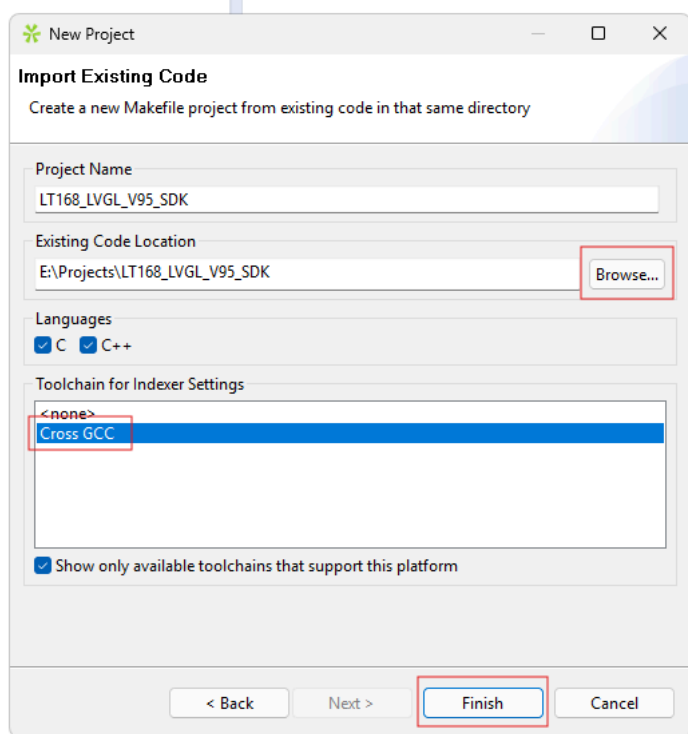
步骤 2: 导入现有项目

- 点击菜单: **Project** → **Import**
- 在弹出的对话框中，展开 **C/C++** 文件夹
- 选择 **Existing Code as Makefile Project**
- 点击 **Next**



步骤 3：选择项目根目录

- 点击 **Browse** 按钮
- 导航到项目根目录：例如 `E:\Projects\LT168_LVGL_V95_SDK`
- Toolchain for Indexer Settings 列表中 选择 [Cross GCC]
- 点击 **Finish**



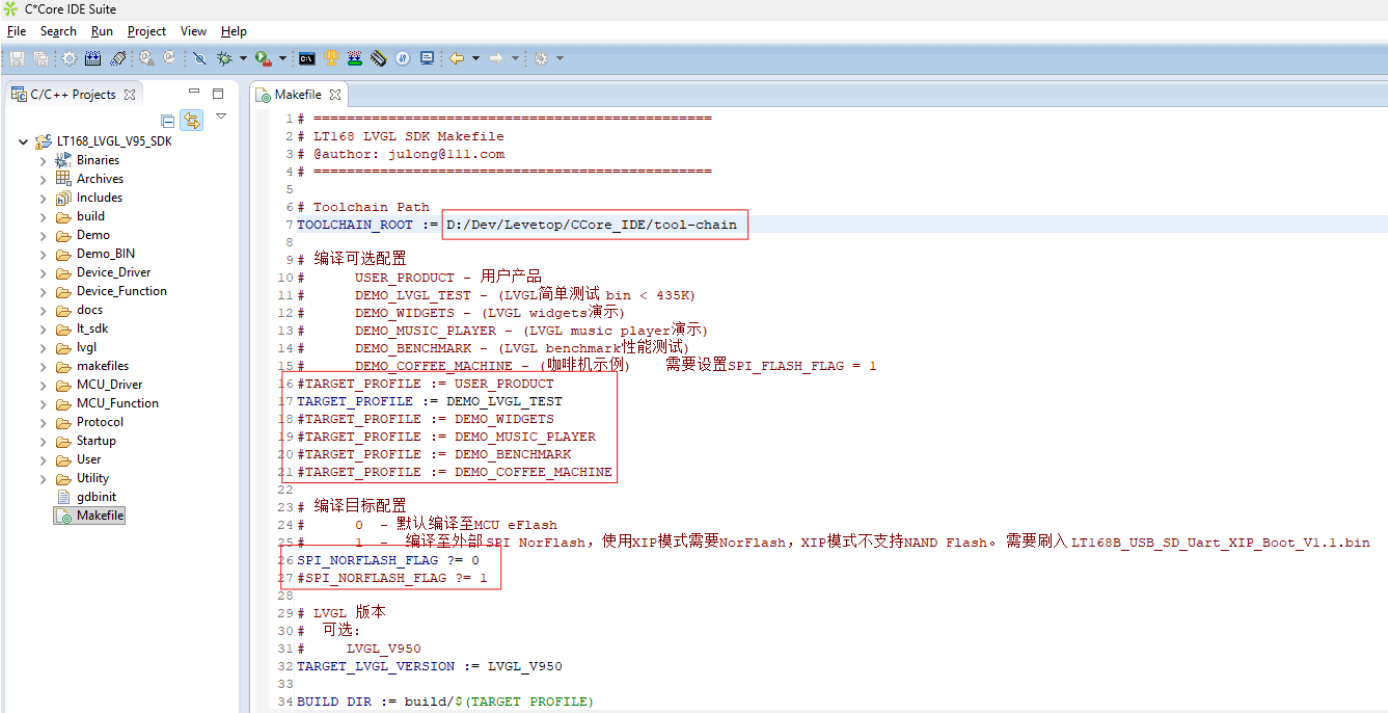
2.4 工具链配置

打开项目根目录下的 `Makefile` 文件，修改第7行 `TOOLCHAIN_ROOT` 的配置，指向您的 CCore IDE 工具链实际安装路径。

例如：

```
TOOLCHAIN_ROOT := D:/Dev/Levetop/CCore_IDE/tool-chain
```

如果您的 CCore IDE 安装在其他位置，请修改为实际路径。



3. 工程编译配置

3.1 概念介绍

- **Target Profile** 是编译配置的核心概念，用于指定当前编译的目标程序。
- 通过修改项目根目录下Makefile文件中的 **TARGET_PROFILE** 变量，可以切换不同的演示程序或用户产品开发模式。
- **SPI_NORFLASH_FLAG** 是编译编译目标位置的核心概念，用于指定当前编译的目标位置为芯片内部eFlash(2M)还是外部SPI NorFlash中。

3.2 可用 Profile 配置

系统支持以下 6 个 Profile：

Profile 名称	说明	生成的 Bin 文件	文件大小	需要的Bootloader	运行模式
USER_PRODUCT	用户产品开发模式	build\USER_PRODUCT\USER_PRODUCT.bin	379KB	芯片出厂自带的Bootloader	eFlas
DEMO_LVGL_TEST	LVGL 简单测试程序	build\DEMO_LVGL_TEST\lvgl_test.bin	372KB	芯片出厂自带的Bootloader	eFlas
DEMO_WIDGETS	LVGL Widgets 完整演示	build\DEMO_WIDGETS\demo_widgets.bin	634KB	Demo_BIN\LT168B_USB_SD_Uart_Boot_V1.51A-eFlash-2M.bin	eFlas
DEMO_MUSIC_PLAYER	LVGL 音乐播放器演示	build\DEMO_MUSIC_PLAYER\demo_music.bin	1214KB	Demo_BIN\LT168B_USB_SD_Uart_Boot_V1.51A-eFlash-2M.bin	eFlas
DEMO_BENCHMARK	LVGL 性能基准测试	build\DEMO_BENCHMARK\demo_benchmark.bin	822KB	Demo_BIN\LT168B_USB_SD_Uart_Boot_V1.51A-eFlash-2M.bin	eFlas
DEMO_COFFEE_MACHINE	咖啡机 Demo	build\DEMO_COFFEE_MACHINE\demo_coffee_machine.bin	6990KB	Demo_BIN\LT168B_USB_SD_Uart_XIP_Boot_V1.1-Fix.bin	SPI Flash XIP

3.3 可用 SPI_NORFLASH_FLAG 配置

系统支持编译至芯片内部Flash和外部SPI NorFlash。

值	目标	起始位置	可用空间大小
0	芯片内部Flash	0x6000A000	1960KB
1	外部SPI NorFlash	0x70000000	视使用的具体芯片容量大小，LT168B开发板自带128M

3.4 编译不同 Demo

在 Makefile中修改 TARGET_PROFILE 变量。取消注释即可使用，不可同时打开多个Profile

```
TARGET_PROFILE := USER_PRODUCT
#TARGET_PROFILE := DEMO_LVGL_TEST
#TARGET_PROFILE := DEMO_WIDGETS
#TARGET_PROFILE := DEMO_MUSIC_PLAYER
#TARGET_PROFILE := DEMO_BENCHMARK
#TARGET_PROFILE := DEMO_COFFEE_MACHINE
```

在 Makefile中修改 SPI_NORFLASH_FLAG 变量。取消注释即可使用。不可同时打开多个SPI_NORFLASH_FLAG

```
SPI_NORFLASH_FLAG ?= 0
#SPI_NORFLASH_FLAG ?= 1
```

操作步骤：

- 1. 找到要使用的 Profile，取消该行注释
- 2. 选择要编译的目标位置，SPI_NORFLASH_FLAG，取消该行注释。（默认2M之内的固件选择0。 咖啡机Demo超过了2M，必须编译至外部SPI NorFlash中，选择1。）
- 3. 保存文件
- 4. 项目名称上点击 鼠标右键 > Build Project。
- 5. 等待编译完成
- 6. 在build 目录下查看对应profile名称的编译结果

编译完成后，会在 build/\${TARGET_PROFILE}/ 目录下生成以下文件：

- *.elf - ELF 格式的可执行文件
- *.bin - 二进制固件文件（用于烧录）
- *.map - 链接映射文件（用于分析内存使用情况）

3.5 各 Profile 入口文件说明

USER_PRODUCT

- 主文件：User/user_product.c
- LVGL配置文件：User/user_product_lvgl_conf.h
- 用途： 用户产品开发的起点，可在此文件中编写自定义业务逻辑。详细信息参考[6.用户二次开发指南]

DEMO_LVGL_TEST

- 用途：简单的 LVGL 功能测试，快速验证显示，动画和触摸功能

DEMO_WIDGETS

- 用途：展示 LVGL 所有标准控件的使用方法

DEMO_MUSIC_PLAYER

- 用途：音乐播放器界面演示，包含动画和音频可视化效果

DEMO_BENCHMARK

- 用途：性能基准测试，评估 MCU 的图形处理能力

DEMO_COFFEE_MACHINE

- 用途：咖啡机界面示例，展示复杂 UI 布局和业务逻辑

4. 常见编译问题

问题 1：未知的 TARGET_PROFILE

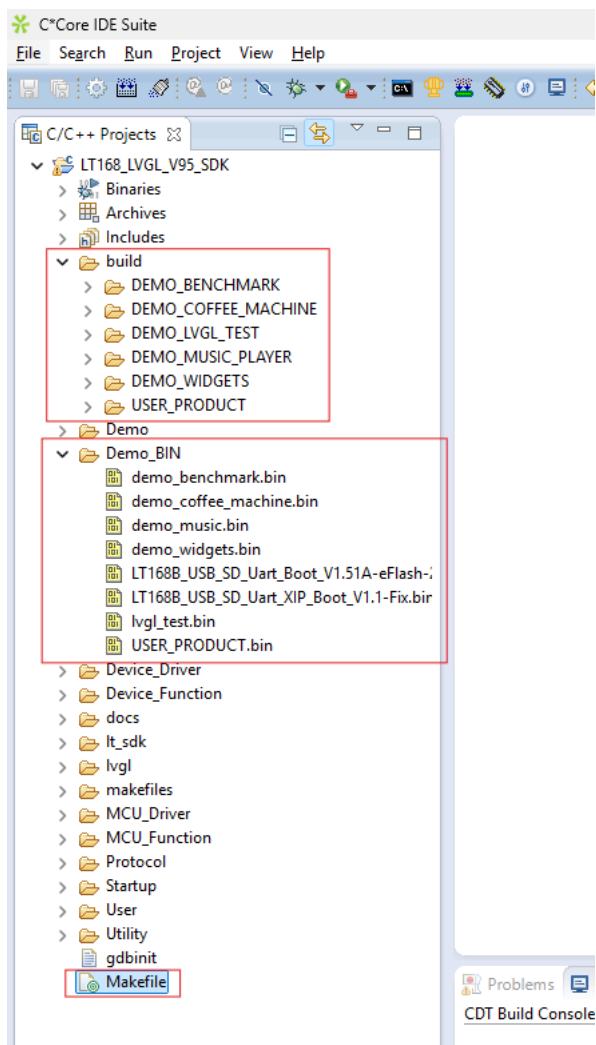
错误信息: `Unknown TARGET_PROFILE: XXX`

解决方法: 检查TARGET_PROFILE拼写是否正确，确保使用的是支持的 Profile 名称

5. Build 目录结构解析

编译输出按照 Profile名称 分类存储:

```
build/
├── USER_PRODUCT/
│   ├── USER_PRODUCT.elf
│   ├── USER_PRODUCT.bin
│   ├── USER_PRODUCT.map
│   └── ...
├── DEMO_LVGL_TEST/
│   ├── lvgl_test.elf
│   ├── lvgl_test.bin
│   ├── lvgl_test.map
│   └── ...
├── DEMO_WIDGETS/
│   ├── demo_widgets.elf
│   ├── demo_widgets.bin
│   ├── demo_widgets.map
│   └── ...
├── DEMO_MUSIC_PLAYER/
│   ├── demo_music.elf
│   ├── demo_music.bin
│   ├── demo_music.map
│   └── ...
├── DEMO_BENCHMARK/
│   ├── demo_benchmark.elf
│   ├── demo_benchmark.bin
│   ├── demo_benchmark.map
│   └── ...
└── DEMO_COFFEE_MACHINE/
    ├── demo_coffee_machine.elf
    ├── demo_coffee_machine.bin
    ├── demo_coffee_machine.map
    └── ...
```



6. 用户二次开发指南

User/user_product.c 是用户产品开发的入口文件。

6.1 user_product.c 基本结构

```
#include "lt_api.h" // MCU硬件驱动
#include "lvt_api.h" // LVGL 工具类

// 用户初始化函数
void user_product_init(void)
{
    // 在这里添加您的初始化代码
    // 例如：外设初始化、变量初始化等
}
```

6.2 user_product_lvgl_conf.h

User/user_product_lvgl_conf.h 是用户LVGL配置文件，用于覆盖 USER_PRODUCT的 LVGL模块 配置。

显示分辨率配置：

```
#define LV_HOR_RES_MAX    480
#define LV_VER_RES_MAX    272
```

颜色深度配置：

```
#define LV_COLOR_DEPTH    16
```

配置加载优先级

LVGL配置加载顺序：

1. LVGL 9.5 默认配置（`lt_sdk/lvgl/sw_profiles/profile_v950_full.h`）
2. Demo配置覆写（`lt_sdk/lvgl/lv_conf_dispatcher.h`）
3. 用户App覆写配置（`User/user_product_lvgl_conf.h`） ← 您可以修改这个文件

6.3 LVGL 测试

在 `user_product_init()` 中调用 LVGL API 创建界面：

```
#include "lt_api.h"
#include "lvt_api.h"
void user_product_init(void)
{
    // 串口打印信息
    printf("User App run.\r\n");

    // LVGL
    lvt_screen_clear(lv_color_white());

    lv_obj_t * ver_label = lv_label_create(lv_scr_act());
    lv_label_set_text_fmt(ver_label, "LVGL:%d.%d.%d", LVGL_VERSION_MAJOR, LVGL_VERSION_MINOR, LVGL_VERSION_PATCH);
    lv_obj_set_style_text_color(ver_label, lv_color_black(), 0);

    lv_obj_t * app_label = lv_label_create(lv_scr_act());
    lv_label_set_text(app_label, "User App");
    lv_obj_set_style_text_color(app_label, lv_color_black(), 0);
    lv_obj_center(app_label);
}
```

7. 快速烧录指南Demo

Demo_BIN/ 目录中包含以下文件：

文件名	对应 Demo	说明
lvgl_test.bin	DEMO_LVGL_TEST	LVGL 简单测试程序
demo_widgets.bin	DEMO_WIDGETS	Widgets 完整演示
demo_music.bin	DEMO_MUSIC_PLAYER	音乐播放器演示
demo_benchmark.bin	DEMO_BENCHMARK	性能基准测试
demo_coffee_machine.bin	DEMO_COFFEE_MACHINE	咖啡机 UI 示例
USER_PRODUCT.bin	USER_PRODUCT	用户App框架
LT168B_USB_SD_Uart_Boot_V1.51A-eFlash-2M.bin	Bootloader	bin大于512KB需要更新此Bootloader
LT168B_USB_SD_Uart_XIP_Boot_V1.1-Fix.bin	Bootloader	bin大于1960KB需要更新此Bootloader

8 Bootloader 更新

8.1 什么情况需要更新 Bootloader？

- 检查您的 Bin 文件大小,当生成的 Bin 文件**大于 512KB** 时，必须更新出厂自带的 Bootloader 才能正常烧录和运行。芯片出厂自带的 Bootloader 不支持超过512KB的固件。
- eFlash最大支持2M的bin文件。
- 大于2M的固件需要将固件写入外部的SPI NorFlash中。

8.2 Bootloader 文件

- 文件名： `LT168B_USB_SD_Uart_Boot_V1.51A-eFlash-2M.bin`
- 版本： V1.51A-eFlash-2M
- 支持容量： 最高 2MB Flash
- 烧录方式： USB
- 文件名： `LT168B_USB_SD_Uart_XIP_Boot_V1.1-Fix.bin`
- 版本： V1.1-Fix
- 支持容量： 取决于所用具体芯片容量
- 烧录方式： USB

8.3 Bootloader 烧录方法

USB 烧录方式

- 1. 取得 乐升烧录板 LT_SWD_ISP_PRogrammer_Lite_V1.3
- 2. 下载 LT_Programmer_V1.20A 烧录程序
- 3. 使用 USB 线连接 LT168 开发板到 PC
- 4. 运行 LT_Programmer_V1.20A
- 5. 选择 需要的Bootloader bin 文件
- 6. 烧录至Flash
- 7. 重启开发板

9. 常见问题与故障排除

问题 1：Windows 路径问题

症状：找不到文件或目录，路径中包含空格或特殊字符
解决：

- 避免在项目路径中使用空格
- 使用正斜杠 / 或双反斜杠 \\
- 避免使用中文路径

10. 编译优化

在 `makefiles/flags.mk` 中调整编译优化级别：

```
CFLAGS += -O2      # 平衡优化
# 或
CFLAGS += -Os      # 体积优化（推荐用于容量受限场景）
# 或
CFLAGS += -O3      # 速度优化
```

技术支持

- 官方文档：参考 CCore IDE 用户手册
- LVGL 文档：<https://docs.lvgl.io/>
- 问题反馈：联系乐升技术支持团队

文档版本：V1.0 适用 SDK 版本：LT168 LVGL V95 SDK 作者：julong@111.com