《LCD 字库 API 功能及用法详解》

在使用本字库前，请务必仔细阅读并严格遵守本使用手册。该字库基于stb库开发的文本框绘制功能、字体渲染功能，手册中均提供了详尽的解释与操作说明。本次测试设备型号为正点原子I.MX6ULL，屏幕分辨率1024×600，色彩模式RGB565，字库在此设备上运行表现稳定。若需将字库应用于其他ARM\_Linux设备，需手动修改lcd\_font.c文件中lcd设备初始化部分的分辨率参数，随后使用arm-linux-gnueabihf-gcc编译链重新编译生成静态库，以便在不同项目中顺利调用。

受限于作者技术水平，本字库部分代码由AI辅助生成，使用过程中可能出现未知错误，甚至导致设备异常。使用本字库即视为您已充分知悉并自愿承担相关风险，若因使用本字库造成任何损失，作者均不承担责任。

本字库以轻量化设计实现高效字体渲染，支持多类型小型嵌入式设备运行环境，是开发者值得信赖的优质资源。如需基于该字库开展二次开发，建议严格遵守开源协议，保留原作者版权声明，并在合理范围内进行功能扩展。因使用或修改字库产生的任何技术问题与法律纠纷，原作者将不承担相关责任，感谢各位开发者的理解与配合。

|  |
| --- |
| e45a168012ee92c6c1b7b25c9fdd26dc |

作者：梁垲东

联系方式：[226429965@qq.com](mailto:226429965@qq.com)

时间：2025.05.21

# 一、初始化与清理

## 1. lcd\_init

* **功能**：初始化 LCD 设备和字体库。
* **原型**：int lcd\_init(const char \*lcd\_path, const char \*font\_path);
* **参数**：
  + lcd\_path：LCD 设备文件的路径，如 /dev/fb0。
  + font\_path：字体文件的路径，如 simkai.ttf。
* **返回值**：成功返回 0，失败返回 -1。
* **用法示例**：

|  |
| --- |
| if (lcd\_init("/dev/fb0", "simkai.ttf") != 0) {  printf("初始化失败.\n");  return -1;  } |

## lcd\_cleanup

* **功能**：清理资源，释放 LCD 设备和字体缓冲区，防止内存泄漏。
* **原型**：void lcd\_cleanup(void);
* **用法示例**：

|  |
| --- |
| // 在程序结束前调用  lcd\_cleanup(); |

# 二、基本图形绘制

## 1. lcd\_draw\_pixel

* **功能**：在指定位置绘制一个像素点。
* **原型**：void lcd\_draw\_pixel(int x, int y, color\_t color);
* **参数**：
  + x、y：像素点的坐标。
  + color：像素点的颜色。
* **用法示例**：

|  |
| --- |
| lcd\_draw\_pixel(100, 200, COLOR\_RED); |

## lcd\_draw\_line

* **功能**：使用 Bresenham 算法在两点之间绘制一条直线。
* **原型**：void lcd\_draw\_line(int x1, int y1, int x2, int y2, color\_t color);
* **参数**：
  + x1、y1：直线起点坐标。
  + x2、y2：直线终点坐标。
  + color：直线颜色。
* **用法示例**：

|  |
| --- |
| lcd\_draw\_line(50, 50, 200, 200, COLOR\_GREEN); |

## lcd\_draw\_rectangle

* **功能**：绘制一个空心矩形。
* **原型**：void lcd\_draw\_rectangle(int x, int y, int width, int height, color\_t color);
* **参数**：
  + x、y：矩形左上角坐标。
  + width、height：矩形的宽度和高度。
  + color：矩形边框颜色。
* **用法示例**：

|  |
| --- |
| lcd\_draw\_rectangle(50, 350, 200, 60, COLOR\_GREEN); |

## lcd\_draw\_filled\_rectangle

* **功能**：绘制一个填充矩形。
* **原型**：void lcd\_draw\_filled\_rectangle(int x, int y, int width, int height, color\_t color);
* **参数**：
  + x、y：矩形左上角坐标。
  + width、height：矩形的宽度和高度。
  + color：矩形填充颜色。
* **用法示例**：

|  |
| --- |
| lcd\_draw\_filled\_rectangle(300, 350, 200, 60, COLOR\_GREEN); |

## lcd\_draw\_rounded\_rectangle

* **功能**：绘制一个空心圆角矩形。
* **原型**：void lcd\_draw\_rounded\_rectangle(int x, int y, int width, int height, int radius, color\_t color);
* **参数**：
  + x、y：矩形左上角坐标。
  + width、height：矩形的宽度和高度。
  + radius：圆角半径。
  + color：矩形边框颜色。
* **用法示例**：

|  |
| --- |
| lcd\_draw\_rounded\_rectangle(50, 420, 200, 60, 15, COLOR\_YELLOW); |

## lcd\_draw\_filled\_rounded\_rectangle

* **功能**：绘制一个填充圆角矩形。
* **原型**：void lcd\_draw\_filled\_rounded\_rectangle(int x, int y, int width, int height, int radius, color\_t color);
* **参数**：
  + x、y：矩形左上角坐标。
  + width、height：矩形的宽度和高度。
  + radius：圆角半径。
  + color：矩形填充颜色。
* **用法示例**：

|  |
| --- |
| lcd\_draw\_filled\_rounded\_rectangle(300, 420, 200, 60, 25, COLOR\_YELLOW); |

# 三、文本渲染

## 1. lcd\_set\_font\_size

* **功能**：设置字体大小。
* **原型**：void lcd\_set\_font\_size(int size);
* **参数**：
  + size：字体大小。
* **用法示例**：

|  |
| --- |
| lcd\_set\_font\_size(20); |

## 2. lcd\_get\_text\_width

* **功能**：计算指定文本的宽度。
* **原型**：int lcd\_get\_text\_width(const char \*text);
* **参数**：
  + text：要计算宽度的文本字符串。
* **返回值**：文本的宽度。
* **用法示例**：

|  |
| --- |
| int width = lcd\_get\_text\_width("Hello World"); |

## lcd\_get\_text\_height

* **功能**：计算文本的高度。
* **原型**：int lcd\_get\_text\_height(void);
* **返回值**：文本的高度。
* **用法示例**：

|  |
| --- |
| int height = lcd\_get\_text\_height(); |

## lcd\_render\_text

* **功能**：在指定位置渲染文本。
* **原型**：void lcd\_render\_text(const char \*text, int x, int y, color\_t text\_color, int font\_size);
* **参数**：
  + text：要渲染的文本字符串。
  + x、y：文本渲染起始位置的坐标。
  + text\_color：文本的颜色。
  + font\_size：文本的字体大小。
* **用法示例**：

|  |
| --- |
| lcd\_render\_text("普通文本渲染", 50, 50, COLOR\_WHITE, 45); |

## lcd\_render\_text\_with\_box

* **功能**：在带文本框的情况下渲染文本。
* **原型**：void lcd\_render\_text\_with\_box(const char \*text, int x, int y, color\_t text\_color, color\_t box\_color, int padding, BoxStyle style, int radius, int font\_size, int box\_width, int box\_height);
* **参数**：
  + text：要渲染的文本字符串。
  + x、y：文本渲染起始位置的坐标。
  + text\_color：文本的颜色。
  + box\_color：文本框的填充颜色。
  + padding：文本与文本框边缘的间距。
  + style：文本框的样式，如 BOX\_STYLE\_RECTANGLE 或 BOX\_STYLE\_ROUNDED。
  + radius：若 style 为圆角矩形，该参数指定圆角的半径。
  + font\_size：文本的字体大小。
  + box\_width：文本框的宽度，为 0 时，根据文本量与字体大小调整，文字居中对齐。
  + box\_height：文本框的高度，为 0 时，根据文本量与字体大小调整，文字居中对齐。
* **用法示例**：

|  |
| --- |
| lcd\_render\_text\_with\_box(  "标准文字自动居中文本框 (r=10)",  50, 170,  COLOR\_WHITE,  COLOR\_RED,  10,  BOX\_STYLE\_ROUNDED,  15,  25,  0,  0  ); |

# 四、其他辅助函数

## 1. decode\_utf8

* **功能**：将 UTF - 8 字符解码为 Unicode 码点。
* **原型**：static int decode\_utf8(const char \*str, int \*codepoint);
* **参数**：
  + str：指向 UTF - 8 编码字符串的指针。
  + codepoint：指向整数的指针，用于存储解码后的 Unicode 码点。
* **返回值**：当前字符的字节长度。
* **用法示例**：

|  |
| --- |
| int codepoint;  int len = decode\_utf8("你好", &codepoint); |

# 五、枚举类型

## 1. BoxStyle

* **定义**：

|  |
| --- |
| enum {  BOX\_STYLE\_RECTANGLE, /\* 矩形样式 \*/  BOX\_STYLE\_ROUNDED /\* 圆角矩形样式 \*/  }; |

* **用法**：在 lcd\_render\_text\_with\_box 函数中用于指定文本框的样式。

|  |
| --- |
| lcd\_render\_text\_with\_box(  "圆角文本框",  50, 170,  COLOR\_WHITE,  COLOR\_RED,  10,  BOX\_STYLE\_ROUNDED,  15,  25,  0,  0  ); |

# 六、 屏幕清空

## 1. lcd\_clear

* **功能**：将整个 LCD 屏幕填充为指定颜色。
* **原型**：void lcd\_clear(color\_t color);
* **参数**：
  + color：要填充的颜色。
* **用法示例**：

|  |
| --- |
| lcd\_clear(COLOR\_BLACK); |

# 其余事项

字库重新编译：arm-linux-gnueabihf-gcc -c -o lcd\_font.o lcd\_font.c -lm -std=gnu99

生成静态库：arm-linux-gnueabihf-ar rcs liblcd\_font.a lcd\_font.o

测试demo编译：arm-linux-gnueabihf-gcc -o font\_demo font\_demo.c -L. -llcd\_font -lm

传输命令：tftp -g -r font\_demo XXX.XXX.XXX.XXX

权限赋予：chmod 777 font\_demo

程序运行: ./font\_demo