QuecPython LCD 用户指导

LTE Standard 模块系列

版本：1.0.0

日期：2021-01-08

状态：临时文件

www.quectel.com

上海移远通信技术股份有限公司始终以为客户提供最及时、最全面的服务为宗旨。如需任何帮助，请随时联系我司上海总部，联系方式如下：

上海移远通信技术股份有限公司
上海市闵行区田林路 1016 号科技绿洲 3 期（B 区）5 号楼　邮编：200233
电话：+86 21 51086236　邮箱：info@quectel.com

或联系我司当地办事处，详情请登录：http://www.quectel.com/cn/support/sales.htm。

如需技术支持或反馈我司技术文档中的问题，可随时登陆如下网址：
http://www.quectel.com/cn/support/technical.htm 或发送邮件至：support@quectel.com。

前言

上海移远通信技术股份有限公司提供该文档内容用以支持其客户的产品设计。客户须按照文档中提供的规范、参数来设计其产品。因未能遵守有关操作或设计规范而造成的损害，上海移远通信技术股份有限公司不承担任何责任。在未声明前，上海移远通信技术股份有限公司有权对该文档进行更新。

免责声明

上海移远通信技术股份有限公司尽力确保开发中功能的完整性、准确性、及时性或效用，但不排除上述功能错误或遗漏的可能。除非其他有效协议另有规定，否则上海移远通信技术股份有限公司对开发中功能的使用不做任何暗示或明示的保证。在适用法律允许的最大范围内，上海移远通信技术股份有限公司不对任何因使用开发中功能而遭受的损失或损害承担责任，无论此类损失或损害是否可以预见。

保密义务

除非上海移远通信技术股份有限公司特别授权，否则我司所提供文档和信息的接收方须对接收的文档和信息保密，不得将其用于除本项目的实施与开展以外的任何其他目的。未经上海移远通信技术股份有限公司书面同意，不得获取、使用或向第三方泄露我司所提供的文档和信息。对于任何违反保密义务、未经授权使用或以其他非法形式恶意使用所述文档和信息的违法侵权行为，上海移远通信技术股份有限公司有权追究法律责任。

版权申明

本文档版权属于上海移远通信技术股份有限公司，任何人未经我司允许而复制转载该文档将承担法律责任。

文档历史

修订记录

版本	日期	作者	变更表述
-	2021-01-08	Felix YE	文档创建
1.0.0	2021-01-08	Felix YE	临时版本

目录

文档历史 2
目录 3
表格索引 4

1 引言 5

2 LCD 相关接口 6
2.1. 创建 LCD 对象 6
2.2. lcd.lcd_init 6
2.3. lcd.lcd_clear 7
2.4. lcd.lcd_write 8
2.5. lcd.lcd_brightness 8
2.6. lcd.lcd_display_on 9
2.7. lcd.lcd_display_off 9
2.8. lcd.lcd_write_cmd 10
2.9. lcd.lcd_write_data 10

3 LCD 配置流程 12
3.1. LCD 硬件接线 12
3.2. 编写屏幕初始化参数 12
3.3. 执行初始化接口 14
3.3.1. 创建 LCD 对象 14
3.3.2. 打开屏显 15
3.3.3. 关闭屏显 15
3.3.4. 编写写屏时区域值 15
3.3.5. 初始化配置 15
3.3.6. 清屏 16
3.3.7. 区域写屏 16

4 LCD 执行示例 18
4.1. LCD 流程脚本命令 18
4.2. LCD 流程脚本执行结果 22

5 附录 A 术语缩写 27

表格索引

表 1：引脚对应表 12
表 2：术语缩写 27

1 引言

本文档基于 QuecPython 介绍如何通过 LCD 模块实现 LCD 显示功能，包括接口、配置流程及示例。本文档仅支持 SPI LCD。

本文档适用于以下模块：

- EC100Y-CN
- EC600S-CN

2 LCD 相关接口

2.1. 创建 LCD 对象

首先，导入 machine 模块下的 LCD，然后创建 LCD 对象。有关示例代码，请参考以下代码清单：

```
from machine import LCD
lcd = LCD()
```

2.2. lcd.lcd_init

该方法用于初始化 LCD。

- **函数原型**

```
lcd.lcd_init(lcd_init_data, lcd_width, lcd_hight, lcd_clk, data_line, line_num, lcd_type, lcd_invalid,
lcd_display_on, lcd_display_off, lcd_set_brightness)
```

- **参数**

lcd_init_data：
传入 LCD 的配置命令

lcd_width：
LCD 屏幕的宽度。宽度不超过 500。

lcd_hight：
LCD 屏幕的高度。高度不超过 500。

lcd_clk：
LCD SPI 时钟。SPI 时钟为 6.5K/13K/26K/52K。

data_line：
数据线数。参数值为 1 和 2。
line_num：

线的数量。参数值为 3 和 4。

lcd_type：
屏幕类型。0：rgb；1：fstn

lcd_invalid：
LCD 写屏时 xy 的设置

lcd_display_on：
LCD 屏亮

lcd_display_off：
LCD 屏灭

lcd_set_brightness：
LCD 设置亮度值。设置为 none 表示由 LCD_BL_K 控制亮度（有些屏幕是由寄存器控制屏幕亮度，有些是通过 LCD_BL_K 控制屏幕亮度）

返回值

0	成功
-1	已经初始化
-2	屏初始化参数错误（为空或过大（大于 1000 像素点））
-3	初始化参数解析错误
-4	屏幕缓存申请失败
-5	配置参数错误

2.3. lcd.lcd_clear

该方法用于清除屏幕。

- **函数原型**

```
lcd.lcd_clear(color)
```

- **参数**

color：
需要刷屏的颜色值

- **返回值**

0	成功

-1	屏幕未初始化

2.4. lcd.lcd_write

该方法用于区域写屏。

- **函数原型**

```
lcd.lcd_write(color_buffer,start_x,start_y,end_x,end_y)
```

- **参数**

Color_buffer：
屏幕的颜色值缓存。

start_x：
起始 x 坐标

start_y：
起始 y 坐标

end_x：
结束 x 坐标

end_y：
结束 y 坐标

- **返回值**

0	成功
-1	屏幕未初始化
-2	宽度和高度设置错误
-3	数据缓存为空

2.5. lcd.lcd_brightness

该方法用于设置屏幕亮度。

- **函数原型**

```
lcd. lcd_brightness(level)
```

- **参数**

level：
亮度等级。此处会调用 *lcd.lcd_init()*中的 *lcd_set_brightness* 参数。若该参数为 None，亮度调节则由背光亮度调节引脚来控制。

- **返回值**

0	成功
-1	屏幕未初始化

2.6. lcd.lcd_display_on

该方法用于设置亮屏。调用此接口后调用 *lcd.lcd_init()*中的 *lcd_display_on* 回调。

- **函数原型**

```
lcd.lcd_display_on ()
```

- **参数**

无

- **返回值**

0	成功
-1	屏幕未初始化

2.7. lcd.lcd_display_off

该方法用于灭屏设置。调用此接口后调用 *lcd.lcd_init()*中的 *lcd_display_off* 回调。

- **函数原型**

```
lcd.lcd_display_off ()
```

- **参数**

无

- **返回值**

0	成功
-1	屏幕未初始化

2.8. lcd.lcd_write_cmd

该方法用于写入命令。

- **函数原型**

```
lcd.lcd_write_cmd (cmd_value, cmd_value_len)
```

- **参数**

cmd_value：
命令值

cmd_value_len：
命令值长度

- **返回值**

0	成功
其他值	失败

2.9. lcd.lcd_write_data

该方法用于写入数据。

- **函数原型**

```
lcd.lcd_write_data (data_value, data_value_len)
```

- **参数**

data_value：
数据值

data_value_len：
数据值长度

- **返回值**

0	成功
其他值	失败

3 LCD 配置流程

3.1. LCD 硬件接线

如下表所示，LCD 引脚对应模块使用的 LCD 模块引脚，如下表所示：

表 1：引脚对应表

LCD 引脚	LCD 模块引脚
LCD_SPI_CLK	gpio[20]
LCD_SPI_DOUT	gpio[24]
LCD_SPI_CS	gpio[22]
LCD_SPI_RS	gpio[21]
LCD_SPI_RST	gpio[26]
LCD_BL_K	/

3.2. 编写屏幕初始化参数

在交互式命令行窗口中键入以下内容，准备 LCD 屏幕初始化参数，参数格式为：类型+长度+参数值：

- 类型：0 表示命令；1 表示数据；2 表示延时
- 长度：若类型为 0，则长度表示命令后的数据数量；若类型为 1，则长度表示数据的长度
- 参数值：对应值

以下以 ili9225 为例：

```
Ili9225_init = (
0,1,0x02,                    #命令，后接一个 data, cmd 值为 0x02
1,2,0x01,0x00,               #数据，数据长度为 2，  data 值为 0x0100
0,1,0x01,
1,2,0x01,0x1C,
0,1,0x03,
1,2,0x10,0x30,
0,1,0x08,
1,2,0x08,0x08,
0,1,0x0B,
1,2,0x11,0x00,
0,1,0x0C,
1,2,0x00,0x00,
0,1,0x0F,
1,2,0x14,0x01,
0,1,0x15,
1,2,0x00,0x00,
0,1,0x20,
1,2,0x00,0x00,
0,1,0x21,
1,2,0x00,0x00,
0,1,0x10,
1,2,0x08,0x00,
0,1,0x11,
1,2,0x1F,0x3F,
0,1,0x12,
1,2,0x01,0x21,
0,1,0x13,
1,2,0x00,0x0F,
0,1,0x14,
1,2,0x43,0x49,
0,1,0x30,
1,2,0x00,0x00,
0,1,0x31,
1,2,0x00,0xDB,
0,1,0x32,
1,2,0x00,0x00,
0,1,0x33,
1,2,0x00,0x00,
0,1,0x34,
1,2,0x00,0xDB,
0,1,0x35,
1,2,0x00,0x00,
0,1,0x36,
1,2,0x00,0xAF,
0,1,0x37,
1,2,0x00,0x00,
0,1,0x38,
1,2,0x00,0xDB,
0,1,0x39,
1,2,0x00,0x00,
0,1,0x50,
1,2,0x00,0x01,
0,1,0x51,
1,2,0x20,0x0B,
0,1,0x52,
1,2,0x00,0x00,
0,1,0x53,
1,2,0x04,0x04,
0,1,0x54,
1,2,0x0C,0x0C,
0,1,0x55,
1,2,0x00,0x0C,
0,1,0x56,
1,2,0x01,0x01,
0,1,0x57,
1,2,0x04,0x00,
0,1,0x58,
1,2,0x11,0x08,
0,1,0x59,
1,2,0x05,0x0C,
0,1,0x07,
1,2,0x10,0x17,
0,1,0x22,
)
Ili9225_init_data = bytearray(Ili9225_init)
```

3.3. 执行初始化接口

本节以 ili9225 为例，演示如何驱动屏幕。

3.3.1. 创建 LCD 对象

在交互式命令行窗口中键入以下命令，创建 LCD 对象。

```
from machine import LCD
lcd = LCD()
```

3.3.2. 打开屏显

在交互式命令行窗口中键入以下命令，打开屏显：

```
def display_on(para):
print("display on")
lcd.lcd_write_cmd(0x07, 1)
lcd.lcd_write_data(0x1017, 2)
```

3.3.3. 关闭屏显

在交互式命令行窗口中键入以下命令，关闭屏显：

```
def display_off(para):
print("display off")
lcd.lcd_write_cmd(0x07, 1)
lcd.lcd_write_data(0x1004, 2)
```

3.3.4. 编写写屏时区域值

不同的 LCD 屏有不同的设置区域方式，故放置 python 层实现。在底层实现 *lcd_write* 时，会调用该函数在交互式命令行窗口中键入以下命令，实现 *lcd_invalid*：

```
def lcd_invalid(para):
print("invalid:",para[0], para[1], para[2], para[3])
lcd.lcd_write_cmd(0x36, 1)
lcd.lcd_write_data(para[2], 2)
lcd.lcd_write_cmd(0x37, 1)
lcd.lcd_write_data(para[0], 2)
lcd.lcd_write_cmd(0x38, 1)
lcd.lcd_write_data(para[3], 2)
lcd.lcd_write_cmd(0x39, 1)
lcd.lcd_write_data(para[1], 2)
lcd.lcd_write_cmd(0x20, 1)
lcd.lcd_write_data(para[0], 2)
lcd.lcd_write_cmd(0x21, 1)
lcd.lcd_write_data(para[1], 2)
lcd.lcd_write_cmd(0x22, 1)
lcd.lcd_write_cmd(0xff, 1)  #此 0xff 尤为重要，此值时配置区域完成的标志
```

3.3.5. 初始化配置

在交互式命令行窗口中键入以下命令，实现 LCD 配置：

```
lcd.lcd_init(Ili9225_init_data,176,220,13000,1,4,0,lcd_invalid,display_on,display_off,None))
Ili9225_init_data：2.1 配置的初始化参数
176：      lcd 宽度
220：      lcd 高度
13000：   spi clk
1：          1 根数据线
4：          4 根线
0：          type，  0 表示 rgb
lcd_invalid：  区域写屏，设置范围
display_on：  亮屏
display_off:         息屏
None:           表示 LCD 亮度由 IO 口控制
```

3.3.6. 清屏

在交互式命令行窗口中键入以下命令，实现清屏：

```
lcd.lcd_clear(0x001f)
```

3.3.7. 区域写屏

在交互式命令行窗口中键入以下命令，实现区域写屏：

```
test_buf = (
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
)
test_buf1 = bytearray(test_buf)
lcd.lcd_write(test_buf1,10,10,20,20)
```

备注

因设计原因，该示例目标显示为 0x001f（蓝色），当前为 0x1f00。

4 LCD 执行示例

本章节以 ili9225 为例，汇总 LCD 流程脚本命令以及执行结果。

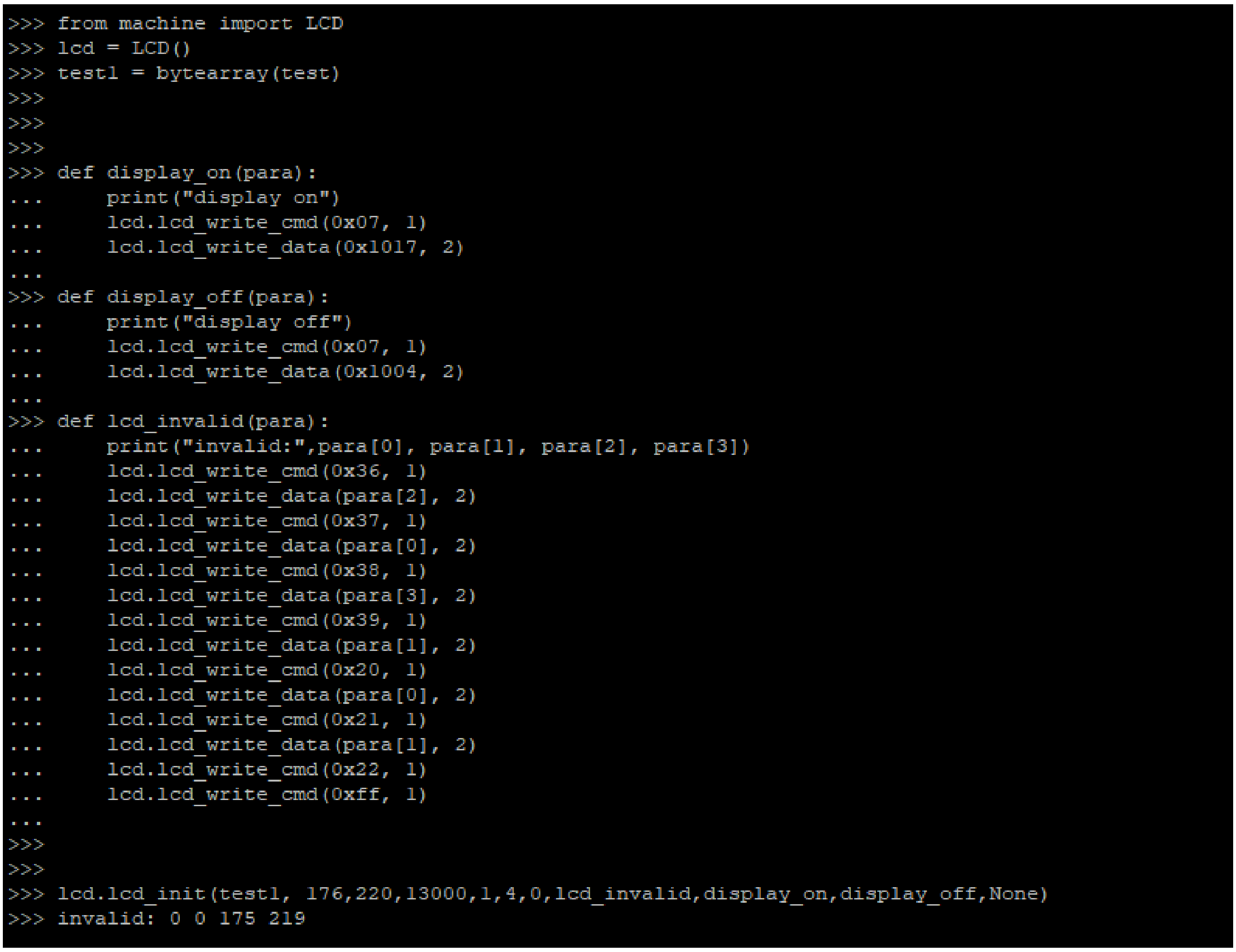
4.1. LCD 流程脚本命令

```
test=(
0,1,0x02,
1,2,0x01,0x00,
0,1,0x01,
1,2,0x01,0x1C,
0,1,0x03,
1,2,0x10,0x30,
0,1,0x08,
1,2,0x08,0x08,
0,1,0x0B,
1,2,0x11,0x00,
0,1,0x0C,
1,2,0x00,0x00,
0,1,0x0F,
1,2,0x14,0x01,
0,1,0x15,
1,2,0x00,0x00,
0,1,0x20,
1,2,0x00,0x00,
0,1,0x21,
1,2,0x00,0x00,
0,1,0x10,
1,2,0x08,0x00,
0,1,0x11,
1,2,0x1F,0x3F,
0,1,0x12,
1,2,0x01,0x21,
0,1,0x13,
1,2,0x00,0x0F,
0,1,0x14,
1,2,0x43,0x49,
0,1,0x30,
1,2,0x00,0x00,
0,1,0x31,
1,2,0x00,0xDB,
0,1,0x32,
1,2,0x00,0x00,
0,1,0x33,
1,2,0x00,0x00,
0,1,0x34,
1,2,0x00,0xDB,
0,1,0x35,
1,2,0x00,0x00,
0,1,0x36,
1,2,0x00,0xAF,
0,1,0x37,
1,2,0x00,0x00,
0,1,0x38,
1,2,0x00,0xDB,
0,1,0x39,
1,2,0x00,0x00,
0,1,0x50,
1,2,0x00,0x01,
0,1,0x51,
1,2,0x20,0x0B,
0,1,0x52,
1,2,0x00,0x00,
0,1,0x53,
1,2,0x04,0x04,
0,1,0x54,
1,2,0x0C,0x0C,
0,1,0x55,
1,2,0x00,0x0C,
0,1,0x56,
1,2,0x01,0x01,
0,1,0x57,
1,2,0x04,0x00,
0,1,0x58,
1,2,0x11,0x08,
0,1,0x59,
1,2,0x05,0x0C,
0,1,0x07,
1,2,0x10,0x17,
0,1,0x22,
)
```

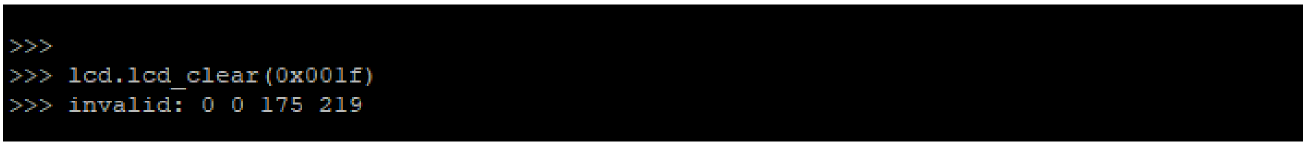
```
from machine import LCD
lcd = LCD()
test1 = bytearray(test)

def display_on(para):
print("display on")
lcd.lcd_write_cmd(0x07, 1)
lcd.lcd_write_data(0x1017, 2)

def display_off(para):
print("display off")
lcd.lcd_write_cmd(0x07, 1)
lcd.lcd_write_data(0x1004, 2)

def display_light(para):
print("display_light")
lcd.lcd_write_cmd(0x13, 1)
lcd.lcd_write_data(para, 2)

def lcd_invalid(para):
print("invalid:",para[0], para[1], para[2], para[3])
lcd.lcd_write_cmd(0x36, 1)
lcd.lcd_write_data(para[2], 2)
lcd.lcd_write_cmd(0x37, 1)
lcd.lcd_write_data(para[0], 2)
lcd.lcd_write_cmd(0x38, 1)
lcd.lcd_write_data(para[3], 2)
lcd.lcd_write_cmd(0x39, 1)
lcd.lcd_write_data(para[1], 2)
lcd.lcd_write_cmd(0x20, 1)
lcd.lcd_write_data(para[0], 2)
lcd.lcd_write_cmd(0x21, 1)
lcd.lcd_write_data(para[1], 2)
lcd.lcd_write_cmd(0x22, 1)
lcd.lcd_write_cmd(0xff, 1)

lcd.lcd_init(test1, 176,220,13000,1,4,0,lcd_invalid,display_on,display_off,display_light)

test_buf = (
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,
0x1f,
)
test_buf1 = bytearray(test_buf)

lcd.lcd_write(test_buf1,10,10,20,20)

lcd.lcd_clear(0xf800) //红色
```

4.2. LCD 流程脚本执行结果

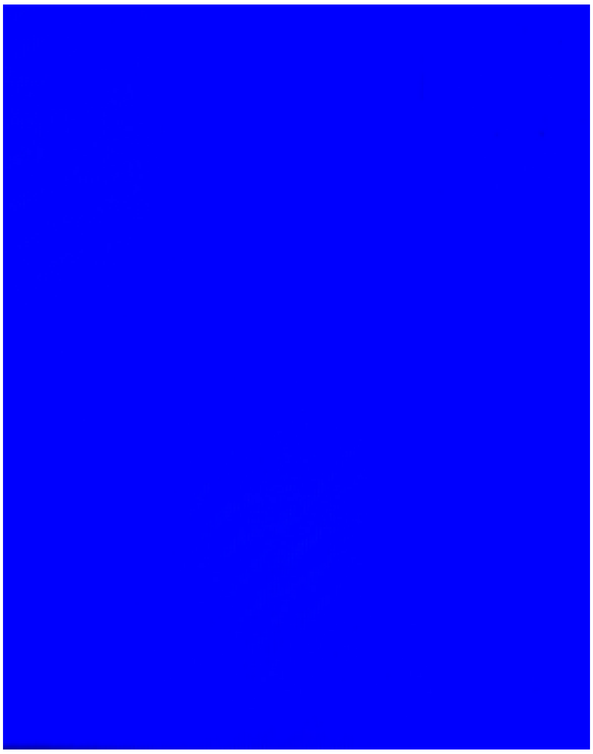
LCD 流程脚本执行结果分别如图所示：

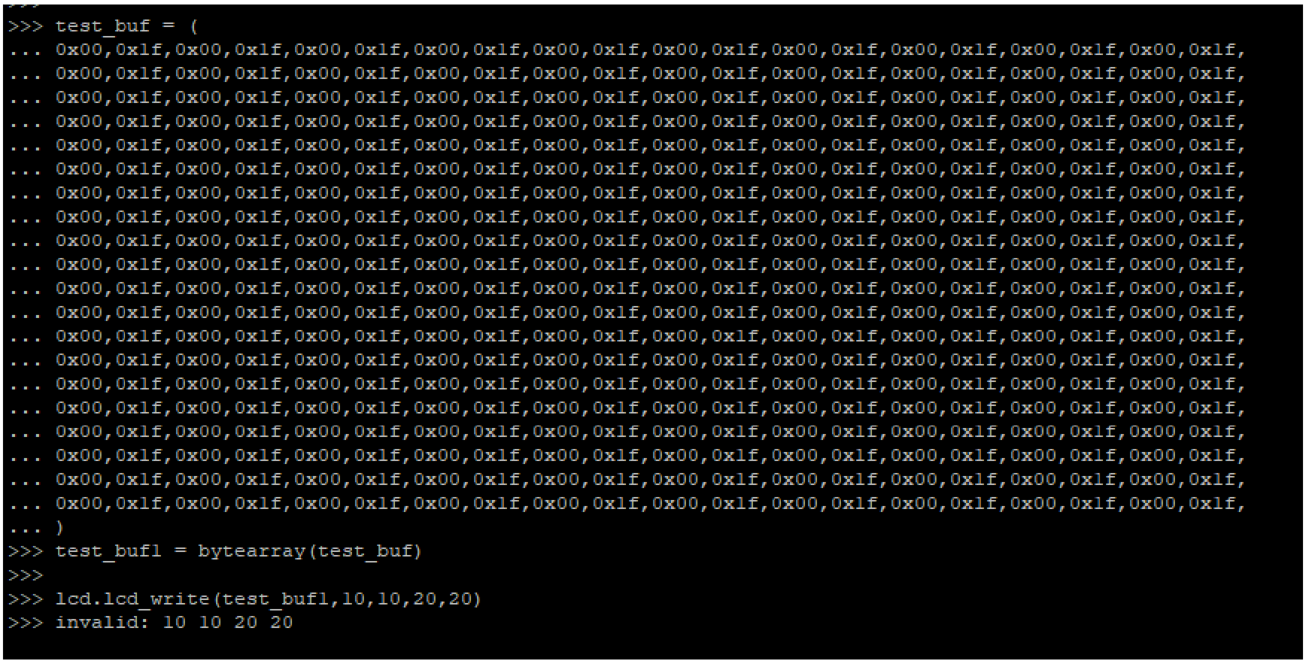


```
>>> from machine import LCD
>>> lcd = LCD()
>>> test1 = bytearray(test)
>>>
>>>
>>>
>>> def display_on(para):
...     print("display on")
...     lcd.lcd_write_cmd(0x07, 1)
...     lcd.lcd_write_data(0x1017, 2)
...
>>> def display_off(para):
...     print("display off")
...     lcd.lcd_write_cmd(0x07, 1)
...     lcd.lcd_write_data(0x1004, 2)
...
>>> def lcd_invalid(para):
...     print("invalid:",para[0], para[1], para[2], para[3])
...     lcd.lcd_write_cmd(0x36, 1)
...     lcd.lcd_write_data(para[2], 2)
...     lcd.lcd_write_cmd(0x37, 1)
...     lcd.lcd_write_data(para[0], 2)
...     lcd.lcd_write_cmd(0x38, 1)
...     lcd.lcd_write_data(para[3], 2)
...     lcd.lcd_write_cmd(0x39, 1)
...     lcd.lcd_write_data(para[1], 2)
...     lcd.lcd_write_cmd(0x20, 1)
...     lcd.lcd_write_data(para[0], 2)
...     lcd.lcd_write_cmd(0x21, 1)
...     lcd.lcd_write_data(para[1], 2)
...     lcd.lcd_write_cmd(0x22, 1)
...     lcd.lcd_write_cmd(0xff, 1)
...
>>>
>>>
>>> lcd.lcd_init(test1, 176,220,13000,1,4,0,lcd_invalid,display_on,display_off,None)
>>> invalid: 0 0 175 219
```

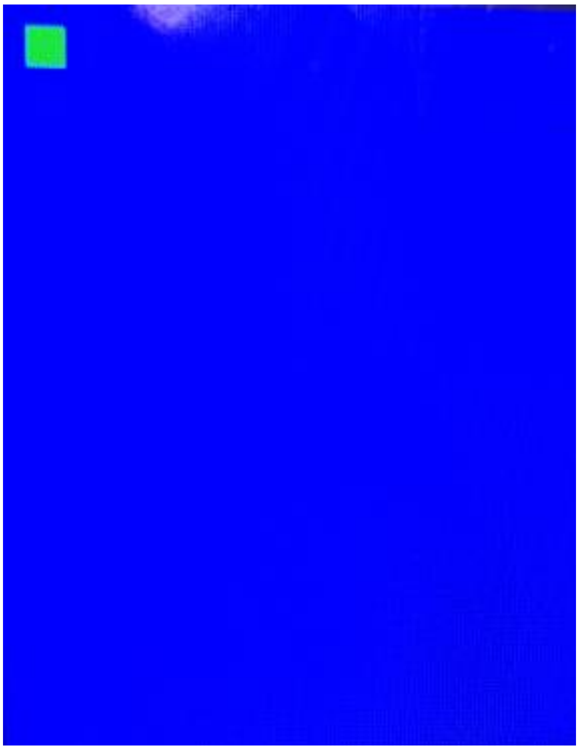


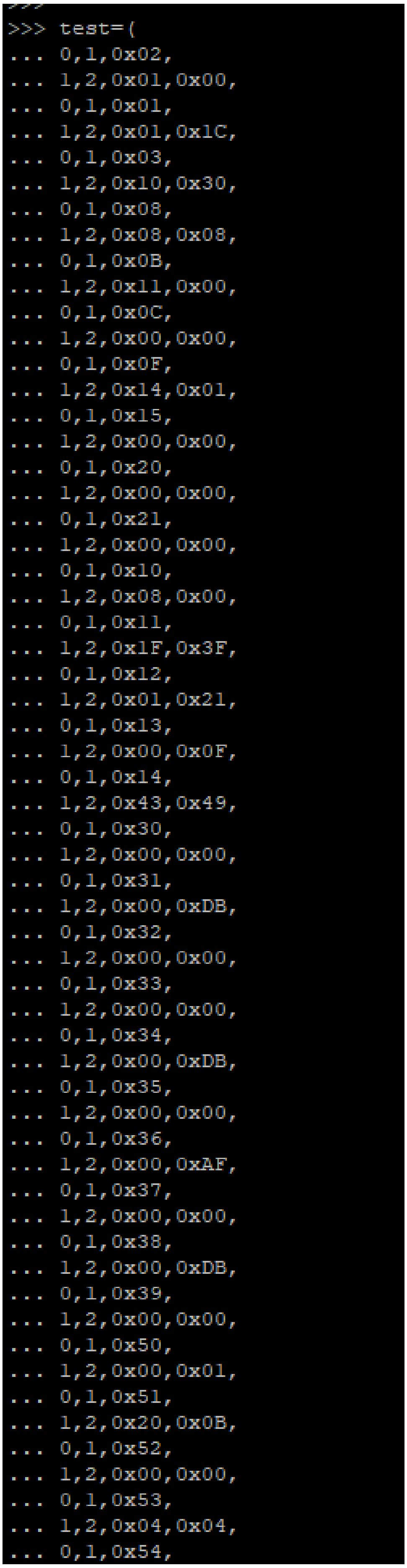
```
>>>
>>> lcd.lcd_clear(0x001f)
>>> invalid: 0 0 175 219
```





```
>>> test_buf = (
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... 0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,0x00,0x1f,
... )
>>> test_buf1 = bytearray(test_buf)
>>>
>>> lcd.lcd_write(test_buf1,10,10,20,20)
>>> invalid: 10 10 20 20
```





```
>>>
>>> test=(
... 0,1,0x02,
... 1,2,0x01,0x00,
... 0,1,0x01,
... 1,2,0x01,0x1C,
... 0,1,0x03,
... 1,2,0x10,0x30,
... 0,1,0x08,
... 1,2,0x08,0x08,
... 0,1,0x0B,
... 1,2,0x11,0x00,
... 0,1,0x0C,
... 1,2,0x00,0x00,
... 0,1,0x0F,
... 1,2,0x14,0x01,
... 0,1,0x15,
... 1,2,0x00,0x00,
... 0,1,0x20,
... 1,2,0x00,0x00,
... 0,1,0x21,
... 1,2,0x00,0x00,
... 0,1,0x10,
... 1,2,0x08,0x00,
... 0,1,0x11,
... 1,2,0x1F,0x3F,
... 0,1,0x12,
... 1,2,0x01,0x21,
... 0,1,0x13,
... 1,2,0x00,0x0F,
... 0,1,0x14,
... 1,2,0x43,0x49,
... 0,1,0x30,
... 1,2,0x00,0x00,
... 0,1,0x31,
... 1,2,0x00,0xDB,
... 0,1,0x32,
... 1,2,0x00,0x00,
... 0,1,0x33,
... 1,2,0x00,0x00,
... 0,1,0x34,
... 1,2,0x00,0xDB,
... 0,1,0x35,
... 1,2,0x00,0x00,
... 0,1,0x36,
... 1,2,0x00,0xAF,
... 0,1,0x37,
... 1,2,0x00,0x00,
... 0,1,0x38,
... 1,2,0x00,0xDB,
... 0,1,0x39,
... 1,2,0x00,0x00,
... 0,1,0x50,
... 1,2,0x00,0x01,
... 0,1,0x51,
... 1,2,0x20,0x0B,
... 0,1,0x52,
... 1,2,0x00,0x00,
... 0,1,0x53,
... 1,2,0x04,0x04,
... 0,1,0x54,
```

5 附录 A 术语缩写

表 2：术语缩写

缩写	英文全称	中文全称
LCD	Liquid Crystal Display	液晶显示器
SPI	Serial Peripheral Interface	串行外设接口