LUA 脚本 API V1.0

工程技术笔记

类别	内容
关键词	Lua 脚本
摘要	LUA 脚本 API 接口函数

深圳市恒域威电子有限公司

修订历史

版本	日期	原因	编制
V1.0	2022/11/29	创建文档	HYWOO7
V1.1	2023/04/10	新增MODBUS接口	HYWOO7

销售与服务

深圳市恒域威电子有限公司

电 话：0755-29100074

手机：18923790358

Email：hywdz@126.com

网 站：www.hywlcm.com

地 址：深圳市宝安区西乡固戍钜鑫科技产业园E栋5楼

官网零售淘宝店：https://shop257531722.taobao.com/

目录

1. **适用范围** 7
2. LUA **脚本介绍** 8
3. API **接口函数** 9
3.1 **控件属性类** 9
3.1.1 change_screen(screen) 9
3.1.2 change_child_screen (screen) 9
3.1.3 change_screen_effect(screen,effect) 9
3.1.4 set_slide_screen(slide,round,start_screen,end_screen) 9
3.1.5 set_button_notify_release () 9
3.1.6 get_current_screen() 9
3.1.7 set_value(screen,control,value) 9
3.1.8 get_value(screen,control) 9
3.1.9 set_visiable(screen,control,visiable) 9
3.1.10 set_enable(screen,control,enable) 9
3.1.11 set_fore_color(screen,control,color) 10
3.1.12 set_back_color(screen,control,color) 10
3.1.13 set_text(screen,control,text) 10
3.1.14 set_text_roll(screen,control, speed) 10
3.1.15 set_text_flicker (screen,control, cycle) 10
3.1.16 get_text(screen,control) 10
3.1.17 set_options (screen, control, options) 10
3.1.18 play_animation(screen，control) 10
3.1.19 stop_animation(screen，control) 10
3.1.20 set_history_graph_value(screen,control,channel1,..,channel8) 10
3.2 **记录控件** 10
3.2.1 record_set_event(screen,control,eventid) 10
3.2.2 record_reset_event(screen,control,eventid) 10
3.2.3 record_add(screen,control,record) 10
3.2.4 record_insert (screen,control,position,record) 11
3.2.5 record_clear(screen,control) 11
3.2.6 record_setoffset(screen,control,offset) 11
3.2.7 record_get_count(screen,control) 11
3.2.8 record_read(screen,control,position) 11
3.2.9 record_modify(screen,control, position ,record) 11
3.2.10 record_delete(screen,control,position) 11
3.2.11 record_select(screen,control,position) 11
3.2.12 record_export(screen,control) 11
3.3 **常用回调函数** 11
3.3.1 on_init() 11
3.3.2 on_systick() 11
3.3.3 on_control_notify(screen,control,value) 11
3.3.4 on_screen_change(screen) 11
3.3.5 on_press(state,x,y) 11
3.3.6 on_usb_inserted(driver) 12
3.3.7 on_usb_removed() 12
3.3.8 on_sd_inserted(dir) 12
3.3.9 on_sd_removed() 12
3.3.10 on_screen_sliding(screen, control) 12
3.4 **绘图函数** 12

3.4.1 on_draw(screen) 12
3.4.2 redraw() 12
3.4.3 set_pen_color(color) 12
3.4.4 draw_line(x0,y0,x1,y1,width) 12
3.4.5 draw_rect(x0,y0,x1,y1,fill) 13
3.4.6 draw_rect_alpha(x0,y0,x1,y1,alpha) 13
3.4.7 draw_circle(x,y,r,fill) 13
3.4.8 draw_ellipse(x0,y0,x1,y1,fill) 13
3.4.9 draw_image(image_id,frame_id,dstx,dsty,width,height,srcx,srcy) 13
3.4.10 draw_image_file(filename,dstx,dsty,width,height,srcx,srcy) 13
3.4.11 load_surface (filename) 14
3.4.13 destroy_surface (surface) 14
3.4.14 draw_surface (surface,dstx,dsty,width,height,srcx,srcy) 14
3.4.15 draw_text(text,x,y,w,h,font,color,align) 14
3.5 **网络相关** 14
3.5.1 get_wifi_cfg() 14
3.5.2 set_wifi_cfg(wifi_mode, secumode, ssid, password) 15
3.5.3 on_wifi_callback(state,reason) 15
3.5.4 get_network_state() 15
3.5.5 set_network_cfg(dhcp, ipaddr, netmask, gateway, dns) 15
3.5.6 get_network_cfg() 15
3.5.7 save_network_cfg() 15
3.5.8 set_network_service_cfg(wificom, mode, port, server_addr) 15
3.5.9 get_network_service_cfg() 15
3.5.10 scan_ap() 15
3.5.11 get_ap_info(index) 15
3.5.12 client_send_data(packet) 16
3.5.13 server_send_data(clinet_id，packet) 16
3.5.14 on_client_recv_data(packet) 16
3.5.15 on_server_recv_data(clinet_id，packet) 16
3.5.16 http_request(taskid,uri,method,content_type,postdata) 16
3.5.17 on_http_response(taskid,response) 17
3.5.18 http_download (taskid,uri,savepath) 17
3.5.19 http_download_bigfile(taskid,uri,,uri,savepath) 17
3.5.20 on_http_download (taskid, status) 17
3.5.21 udp_create(port) 17
3.5.22 udp_close(sockfd) 17
3.5.23 udp_recvfrom(sockfd) 17
3.5.24 udp_sendto(sockfd,ip,port,packet) 17
3.5.25 get_wifi_mac() 17
3.6 **定时器** 18
3.6.1 start_timer(timer_id, timeout, countdown, repeat) 18
3.6.2 stop_timer(timer_id) 18
3.6.3 on_timer(timer_id) 18
3.6.4 get_timer_value(timer_id) 18
3.7 **串口** 18
3.7.1 uart_send_data(packet) 18
3.7.2 uart_set_timeout(timeout, timeout_inter) 18
3.7.3 uart_set_baudrate(baudrate) 18
3.7.4 uart_get_baudrate() 18
3.7.5 on_uart_recv_data(packet) 18

3.7.6 uart_setup(baudrate,parity,stopbit,databits) 19
3.8 音视频 19
3.8.1 play_sound(filename) 19
3.8.2 stop_sound() 19
3.8.3 on_audio_callback (state) 19
3.8.4 set_volume(level) 19
3.8.5 get_volume() 19
3.8.6 play_video(pathname,repeat) 19
3.8.7 pause_video() 19
3.8.8 resume_video() 19
3.8.9 stop_video() 19
3.8.10 play_video(file,left,top,width,height) 19
3.8.11 on_video_notify(msg，v1, v2) 20
3.9 FLASH 存储器读写 20
3.9.1 write_flash(addr,data) 20
3.9.2 read_flash(addr,length) 20
3.9.3 write_flash_string(addr,str) 20
3.9.4 read_flash_string(addr) 20
3.9.5 flush_flash() 20
3.10 MODBUS 协议访问 20
3.10.1 get_variant(name) 20
3.10.2 set_variant(name,value) 20
3.10.3 mb_set_timeout (timeout) 20
3.10.4 mb_read_coil_01 (slave,addr,quantity) 21
3.10.5 mb_read_input_02(slave,addr,quantity) 21
3.10.6 mb_read_reg_03(slave,addr,quantity) 21
3.10.7 mb_read_input_reg_04(slave,addr,quantity) 21
3.10.8 mb_write_coil_05 (slave,addr,status) 21
3.10.9 mb_write_reg_06 (slave,addr,reg) 21
3.10.10 mb_write_coil_15 (slave,addr,quantity,coils) 21
3.10.11 mb_write_reg_16 (slave,addr,regs) 21
3.11 其他 21
3.11.1 set_backlight(level) 21
3.11.2 get_backlight() 21
3.11.3 set_language(lang) 21
3.11.4 get_language () 21
3.11.5 beep(time) 21
3.11.6 get_tick_count () 22
3.11.7 get_date_time () 22
3.11.8 set_date_time (time) 22
3.11.9 refresh_screen() 22
3.11.10 get_version() 22

1. 适用范围

文档仅适合F 系列 串口屏产品。

2. LUA 脚本介绍

LUA 脚本初学者可以通过下面链接进行学习。
http://www.runoob.com/lua/lua-arrays.html

3. API 接口函数

3.1 控件属性类

3.1.1 change_screen(screen)

切换到指定画面
screen：目标画面 ID

3.1.2 change_child_screen (screen)

切换到子画面， 例如对话框
screen：目标子画面 ID

3.1.3 change_screen_effect(screen,effect)

切换到指定画面，使用动画效果
screen：目标画面 ID
effect：动画效果，0 无动画， 1 从左到右， 2 从右到左， 3 从上到下， 4 从下到上，5 渐隐渐消

3.1.4 set_slide_screen(slide,round,start_screen,end_screen)

设置窗口滑动模式
slide：0 禁止滑动， 1 左右滑动， 2 上下滑动
round: 1 环回模式，尾页可以滑到首页
start_screen: 开始画面
end_screen：结束画面

3.1.5 set_button_notify_release ()

设置按钮松开通知模式， 在触摸按钮上也可以滑动窗口

3.1.6 get_current_screen()

获取当前画面 ID

3.1.7 set_value(screen,control,value)

设置控件数值
按钮控件： value -0 按下，1 弹起
文本控件： value -整数或小数
也可以设置进度条、滑块、仪表等

3.1.8 get_value(screen,control)

获取控件数值， 按钮、文本、进度条、滑块、仪表等

3.1.9 set_visiable(screen,control,visiable)

设置控件是否可见
visiable 为 0 隐藏， 1 显示

3.1.10 set_enable(screen,control,enable)

设置控件是否可触摸

enable 为 0 禁止触摸， 1 启用触摸

3.1.11 set_fore_color(screen,control,color)

设置控件前景色，color 为 RGB565
例如文本控件文字颜色， 进度条显示颜色

3.1.12 set_back_color(screen,control,color)

设置控件背景色，color 为 RGB565
例如文本控件背景颜色， 进度条背景颜色

3.1.13 set_text(screen,control,text)

设置控件显示内容(字符串)，文本控件，二维码控件等

3.1.14 set_text_roll(screen,control, speed)

设置文本控件滚动速度， 每秒多少个像素。设置为 0 停止滚动

3.1.15 set_text_flicker (screen,control, cycle)

设置文本控件闪烁周期， 单位 10ms。设置为 0 停止闪烁

3.1.16 get_text(screen,control)

获取控件字符串内容(字符串)，文本控件， 二维码控件等

3.1.17 set_options (screen, control, options)

设置选择控件的内容
例如：set_options (screen,control, ‘选项 1;选项 2;选项 3;’)

3.1.18 play_animation(screen，control)

设置动画控件播放

3.1.19 stop_animation(screen，control)

设置动画控件停止

3.1.20 set_history_graph_value(screen,control,channel1,..,channel8)

设置历史曲线的通道值， 根据通道的个数填写参数
channe l1：通道 1 的值
…
channe8：通道 8 的值

3.2 记录控件

3.2.1 record_set_event(screen,control,eventid)

告警类型-触发告警

3.2.2 record_reset_event(screen,control,eventid)

告警类型-解除告警

3.2.3 record_add(screen,control,record)

在末尾添加一条记录，record 为字符串，例如 “item1;item2;item3;”

3.2.4 record_insert (screen,control,position,record)

在指定位置插入一条记录

3.2.5 record_clear(screen,control)

清除记录数据

3.2.6 record_setoffset(screen,control,offset)

设置滚动显示位置

3.2.7 record_get_count(screen,control)

获取记录条数

3.2.8 record_read(screen,control,position)

读取一条记录， 字符串

3.2.9 record_modify(screen,control, position ,record)

修改一条记录

3.2.10 record_delete(screen,control,position)

删除一条记录

3.2.11 record_select(screen,control,position)

选中一条记录

3.2.12 record_export(screen,control)

导出记录到 SD 卡/U 盘

3.3 常用回调函数

3.3.1 on_init()

系统加载 LUA 脚本文件之后，立即调用此回调函数，通常用于执行初始化操作。

3.3.2 on_systick()

系统每隔 1 秒钟自动调用此回调函数。

3.3.3 on_control_notify(screen,control,value)

用户触摸修改控件后，执行此回调函数。
点击按钮控件， 修改文本控件、修改滑动条都会触发此事件。
value-为数值类型，如果需要获取文本控件的字符串值， 使用 get_text(screen,control)

3.3.4 on_screen_change(screen)

当画面需要切换时， 执行此回调函数，screen 为目标画面。
注意， 此函数内部调用 change_screen,不会嵌套执行 on_screen_change。

3.3.5 on_press(state,x,y)

用户点击触摸屏时， 执行此回调函数。
state-0 松开， 1 按下， 2 持续按压
x,y-为触摸坐标

3.3.6 on_usb_inserted(driver)

U 盘插入时， 执行此回调函数， dirver 为 U 盘的盘符

3.3.7 on_usb_removed()

U 盘拔出时， 执行此回调函数

3.3.8 on_sd_inserted(dir)

SD 卡插入通知， dir 盘符路径

3.3.9 on_sd_removed()

SD 卡拔出通知

3.3.10 on_screen_sliding(screen, control)

用于确定手指滑动时的目标画面 ID

返回切换目标的画面的 ID，`left`、`right` 、up、down，4 个参数， 指定左、右、上、下的页面 `ID`，若为`-1`，该方向不允许滑动

```
-- 系统函数:用于确定手指滑动时的目标画面ID
-- 返回切换目标的画面ID, -1为静止滑动
function on_screen_sliding(screen, control)

    local left  = -1
    local right = -1
    local up    = -1
    local down  = -1
    local index = -1

    if screen == sc_Page0
    then
        left  = sc_Page9
        right = sc_Page1
    end

    return left,right,up,down
end
```

3.4 绘图函数

3.4.1 on_draw(screen)

当界面的显示内容需要更新时， 系统自动调用此函数， 用户在此函数中添加自定义的绘图操作。用户绘制的内容叠加在画面内容之上。

此函数为系统回调函数， 用户不要直接调用。

下面几种情况会触发此函数：

- 界面有动画播放、视频播放、 RTC 时间显示的动态刷新；
- 用户操作屏幕控件控件；
- 通过 LUA 脚本或串口指令更新控件；
- 通过执行 redraw；

总之， 界面上有任何变化，都会触发此回调函数。

3.4.2 redraw()

发送重绘请求， 触发 on_draw 的执行。

3.4.3 set_pen_color(color)

设置画笔的颜色，RGB565 ，用于指定线、矩形、圆等的颜色。

3.4.4 draw_line(x0,y0,x1,y1,width)

绘制直线
x0,y0 起始点坐标
x1,y1 结束点坐标
width 为线条的厚度， 1~10

3.4.5 draw_rect(x0,y0,x1,y1,fill)

绘制矩形
x0,y0 左上角坐标
x1,y1 右下角坐标
fill 为 0 不填充， 1 填充

3.4.6 draw_rect_alpha(x0,y0,x1,y1,alpha)

绘制实心的半透明矩形
x0,y0 左上角坐标
x1,y1 右下角坐标
alpha 透明度 0 全透明~255 不透明

3.4.7 draw_circle(x,y,r,fill)

绘制圆形
x,y 圆的中心坐标
r 圆的半径
fill 为 0 不填充， 1 填充

3.4.8 draw_ellipse(x0,y0,x1,y1,fill)

绘制椭圆
x0,y0 左上角坐标
x1,y1 右下角坐标
fill 为 0 不填充， 1 填充

3.4.9 draw_image(image_id,frame_id,dstx,dsty,width,height,srcx,srcy)

绘制图片
image_id 图片资源的 ID
frame_id 对应图标，可以设置帧 ID，其他图片固定为 0
dstx 图片显示 X 坐标
dsty 图片显示 Y 坐标
width 图片显示宽度
height 图片显示高度
srcx 图片裁剪 X 坐标
srcy 图片裁剪 Y 坐标

3.4.10 draw_image_file(filename,dstx,dsty,width,height,srcx,srcy)

绘制图片，此方法不对图片进行缓存，效率较低

filename 图片文件， 支持 JPEG/PNG
dstx 图片显示 X 坐标
dsty 图片显示 Y 坐标
width 图片显示宽度
height 图片显示高度

srcx 图片裁剪 X 坐标
srcy 图片裁剪 Y 坐标

3.4.11 load_surface (filename)

加载图片到图层
filename 图片文件， 支持 JPEG/PNG
例如： surface = load_surface (“/Images/0.jpg”)
图层不再使用时，需要调用 destroy_surface 进行销毁，否则会导致内存泄漏。

3.4.13 destroy_surface (surface)

销毁图层
surface 图层资源指针

3.4.14 draw_surface (surface,dstx,dsty,width,height,srcx,srcy)

绘制图层，相比于 draw_image_file ，此方法效率较高
surface 图层资源指针
dstx 图片显示 X 坐标
dsty 图片显示 Y 坐标
width 图片显示宽度[可选]
height 图片显示高度[可选]
srcx 图片裁剪 X 坐标[可选]
srcy 图片裁剪 Y 坐标[可选]
例如：
平铺显示 draw_surface(surface, dstx, dsty)
缩放显示 draw_surface(surface, dstx, dsty, width, height)
裁剪显示 draw_surface(surface, dstx, dsty, width, height ,srcx, srcy)

3.4.15 draw_text(text,x,y,w,h,font,color,align)

显示文字
text 字符串
x 显示 X 坐标
y 显示 Y 坐标
w 显示宽度
h 显示高度
font 字体编号
color 颜色 RGB565
align 对齐方式
bit0~bit1 水平对齐方式，0 左对齐，1 居中对齐，2 右对齐
bit2~bit3 垂直对齐方式，0 上对齐，1 居中对齐，3 下对齐

3.5 网络相关

3.5.1 get_wifi_cfg()

返回 4 个参数
wifi_mode, secumode, ssid, password = get_wifi_cfg()
wifi_mode 无线网络模式 0-禁用无线网络， 1-无线网卡模式，2-AP 热点模式
secumode 加密模式 0-AUTO(默认值) 1-WEP 2-WPAPSK 3-WPAPSK2

ssid 无线网络名称
password 无线网络密码

3.5.2 set_wifi_cfg(wifi_mode, secumode, ssid, password)

参数说明同上

3.5.3 on_wifi_callback(state,reason)

网络状态改变时，会被调用

3.5.4 get_network_state()

state = get_network_state()

状态位说明
bit0-无线网络连接
bit1-有线网络连接
bit2-是否连上服务器
bit3-是否有客户端连上

3.5.5 set_network_cfg(dhcp, ipaddr, netmask, gateway, dns)

dhcp-启用 DHCP，0 禁用 1 启用，禁用时后面的参数才有效
ipaddr-静态 IP
netmask-掩码
gateway-子网掩码
dns-域名服务器

3.5.6 get_network_cfg()

返回五个参数， 说明同上
dhcp, ipaddr, netmask, gateway, dns = get_network_cfg()

3.5.7 save_network_cfg()

保存网络设置， 并重连网络
注意： 修改网络配置、服务器参数等需要调用此函数进行保存生效。

3.5.8 set_network_service_cfg(wificom, mode, port, server_addr)

设置网络服务参数
wificom -默认为 0，为 1 时启用透传模式(即无线串口屏)
mode -0 禁用网络服务， 1 客户端模式，2 服务器模式
port -服务端口， 默认 5050
server_addr -服务器地址，(屏作客户端时)

3.5.9 get_network_service_cfg()

返回 4 个参数， 说明同上
wificom, mode, port, server_addr = get_network_service_cfg()

3.5.10 scan_ap()

扫描无线热点， 返回热点数目
ap_count = scan_ap()

3.5.11 get_ap_info(index)

获取指定热点的信息

ssid, security, quality = get_ap_info(index)

index 热点索引， 索引从 0 开始

ssid 热点名称

security 加密方式

quality 信号质量

3.5.12 client_send_data(packet)

通过客户端 SOCKET 发送报文

local packet = {} –定义数组

packet[0] = 0x01

packet[1] = 0x02

…

client_send_data(packet)，发送字节数组 packet，下标从 0 开始

3.5.13 server_send_data(clinet_id，packet)

通过服务端 SOCKET 发送报文

clinet_id：目标客户端 ID

packet：发送字节数组 packet，下标从 0 开始

3.5.14 on_client_recv_data(packet)

当客户端 SOCKET 接收到数据时，系统自动回调此函数。

接收的字节数组 packet，下标从 0 开始

```
function on_client_recv_data(packet)
    --打印消息
    print('on_client_recv_data:')
    for i=0,#(packet) do
      print(packet[i])
    end

    --处理消息，这里简单回送数据
    client_send_data(packet)

    --返回1时，消息不通过串口发送给用户MCU
    return 1
end
```

接收的字节数组 packet，下标从 0 开始

3.5.15 on_server_recv_data(clinet_id，packet)

当服务端 SOCKET 接收到数据时，系统自动回调此函数。

clinet_id：客户端 ID

packet：接收的字节数组 packet，下标从 0 开始

处理方法与 on_client_recv_data 类似。

3.5.16 http_request(taskid,uri,method,content_type,postdata)

发送 HTTP 请求到服务器

taskid：请求任务编号，任意设置

uri：资源路径
method：方法， 0GET，1POST

以下参数 POST 方法才需要
content_type ：数据类型例如 json,xml,text 等
postdata：POST 数据

3.5.17 on_http_response(taskid,response)

HTTP 响应
taskid：响应任务编号，与 http_request 匹配
response ：响应数据

3.5.18 http_download (taskid,uri,savepath)

使用 HTTP 协议下载文件
taskid：请求任务编号，任意设置
uri：资源路径
savepath：存放位置

3.5.19 http_download_bigfile(taskid,uri,,uri,savepath)

使用 HTTP 协议下载大文件
参数说明同上

3.5.20 on_http_download (taskid, status)

下载响应
taskid：响应任务编号，与 http_download 匹配
status：下载状态：0 下载失败， 1 下载成功但存储失败， 2 下载并存储成功

3.5.21 udp_create(port)

创建 UDP 套接字， 并绑定服务端口
例如： sockfd = udp_create(12345)

3.5.22 udp_close(sockfd)

关闭 UDP 套接字

3.5.23 udp_recvfrom(sockfd)

接收 UDP 数据报文
ret,ip,port,packet = udp_recvfrom(sockfd)
ret=- 1 表示发生错误，ret=0 表示无数据，其他值表示数据长度
ip,port 发送端的 IP 和端口
packet 为数据报文，table 类型

3.5.24 udp_sendto(sockfd,ip,port,packet)

发送 UDP 数据报文
sockfd：UDP 套接字
ip,port 接收端的 IP 和端口
packet 为数据报文，table 类型，下标从 0 开始

3.5.25 get_wifi_mac()

获取 mac 地址，返回字符串

3.6 定时器

3.6.1 start_timer(timer_id, timeout, countdown, repeat)

启动定时器，超时后系统自动调用 on_timer
timer_id-定时器 ID，0~31
timeout-超时时间，单位毫秒
countdown-0 顺计时， 1 倒计时
repeat-重复次数，0 表示无限重复

3.6.2 stop_timer(timer_id)

停止定时器

3.6.3 on_timer(timer_id)

定时器超时回调函数

3.6.4 get_timer_value(timer_id)

获取定时器当前计时时间，单位毫秒

3.7 串口

3.7.1 uart_send_data(packet)

通过串口发送数据， packet 为字节数组，下标从 0 开始

3.7.2 uart_set_timeout(timeout, timeout_inter)

设置串口接收超时时间
timeout-接收总超时
timeout_inter-字节间隔超时

3.7.3 uart_set_baudrate(baudrate)

设置波特率，如 uart_set_baudrate(9600)

3.7.4 uart_get_baudrate()

获取波特率

3.7.5 on_uart_recv_data(packet)

串口接收数据的回调函数，通过以下方式触发此函数执行：

- 使用自由串口协议： 在 LUA 脚本中定义全局变量 uart_free_protocol = 1

```
function on_uart_recv_data(packet)
    --打印消息
    print('on_uart_recv_data:')
    for i=0,#(packet) do
      print(packet[i])
    end
end
```

3.7.6 uart_setup(baudrate,parity,stopbit,databits)

串口参数设置
baudrate：波特率值
parity：-0 无校验，1-ODD 奇校验，2-EVEN 偶校验
stopbit：- 0- 1bit ， 1- 1.5bit
databits：数据位 5~8

3.8 音视频

3.8.1 play_sound(filename)

播放指定的声音文件，例如播放屏内音频文件：
play_sound(‘data/Sounds/welcome.mp3’)

3.8.2 stop_sound()

停止声音播放

3.8.3 on_audio_callback (state)

声音播放结束回调通知， state 保留未使用。

3.8.4 set_volume(level)

设置音量 0~100，0 静音， 100 最大音量

3.8.5 get_volume()

获取音量， 返回值 0- 100

3.8.6 play_video(pathname,repeat)

播放视频
pathname 为视频路径
repeat 为重复次数， 0 为无限制次，
播放屏内视频： play_ video(‘data/Video/ 1.mp4’ ， 0)

3.8.7 pause_video()

暂停视频播放

3.8.8 resume_video()

恢复视频播放

3.8.9 stop_video()

停止视频播放

3.8.10 play_video(file,left,top,width,height)

播放视频
file ： 文件路径
left：起始坐标 x
top ： 起始坐标 y
width：视频显示的宽度

height：视频显示的高度

3.8.11 on_video_notify(msg ， v1, v2)

视频播放回调函数
msg：1-播放中， 0-播放完毕
v1:当前播放进度，当前已播时长，单位 s
v2:播放总进度, 当前视频总时长，单位 s

3.9 FLASH 存储器读写

屏幕提供 128K 用户 FLASH，可用于存储配置参数。

3.9.1 write_flash(addr,data)

写用户 FLASH 数据，addr 写入地址，data 字节数组，下标从 0 开始。

3.9.2 read_flash(addr,length)

读用户 FLASH 数据，addr 写入地址，length 读取字节数，返回类型为字节数组，下标从 0 开始
data = read_flash(addr,length)

3.9.3 write_flash_string(addr,str)

写字符串到指定 FLASH 地址

3.9.4 read_flash_string(addr)

从指定 FLASH 地址读取字符串，成功返回字符串， 失败返回 nil
str = read_flash_string(addr)

3.9.5 flush_flash()

系统会对 FLASH 写入操作进行缓存优化，以提高写入效率。
flush_flash 操作会立即把数据写入 FLASH。

3.10 MODBUS 协议访问

LUA 中访问 MODBUS/PLC 等协议中定义的变量，需要通过下面的变量访问接口。
mb_前缀的接口，固件版本要求：W 系列>=3.0.590。0，F 系列>= 4.1.401.0，M 系列>= V6.1.59.00

3.10.1 get_variant(name)

获取协议变量的数值，get_variant(“Variable1”)

3.10.2 set_variant(name,value)

设置协议变量的数值，set_variant(“Variable1”,12345)

3.10.3 mb_set_timeout (timeout)

设置从机应答超时时间，范围 1~255，10 毫秒单位

3.10.4 mb_read_coil_01 (slave,addr,quantity)

01 功能码，读取线圈，成功时返回字节数组，一个字节 8 个 coil，失败返回 nil

3.10.5 mb_read_input_02(slave,addr,quantity)

02 功能码，读取离散输入，成功时返回字节数组，一个字节 8 个 input，失败返回 nil

3.10.6 mb_read_reg_03(slave,addr,quantity)

03 功能码，读取保持寄存器，成功时返回 WORD 数组，失败返回 nil

3.10.7 mb_read_input_reg_04(slave,addr,quantity)

04 功能码，读取输入寄存器，成功时返回 WORD 数组，失败返回 nil

3.10.8 mb_write_coil_05 (slave,addr,status)

05 功能码，写单个线圈，成功时返回 true，失败返回 false
写入 status 1 为 ON，0 为 OFF

3.10.9 mb_write_reg_06 (slave,addr,reg)

06 功能码，写入单个保持寄存器，成功时返回 true，失败返回 false
reg 为寄存器值

3.10.10 mb_write_coil_15 (slave,addr,quantity,coils)

15 功能码，写多个线圈，成功时返回 true，失败返回 false
coils 为字节数组，一个字节 8 个 coil

3.10.11 mb_write_reg_16 (slave,addr,regs)

16 功能码，写入多个保持寄存器，成功时返回 true，失败返回 false
regs 为寄存器 WORD 数组

3.11 其他

3.11.1 set_backlight(level)

设置背光亮度 0~100，0 最暗， 100 最亮

3.11.2 get_backlight()

level = get_backlight，level 为 0- 100

3.11.3 set_language(lang)

设置当前语言选项

3.11.4 get_language ()

获取当前语言选项

3.11.5 beep(time)

蜂鸣器叫， 单位毫秒

3.11.6 get_tick_count ()

获取上电以后运行时间， 单位 10ms。
32 位计数器， 大约 49 天后溢出归零重新计时。

3.11.7 get_date_time ()

获取当前日期时间
year,mon,day,hour,min,sec,week = get_date_time()

3.11.8 set_date_time (time)

设置当前日期时间
set_date_time(year,mon,day,hour,min,sec)

3.11.9 refresh_screen()

刷新画面

3.11.10 get_version()

获取固件版本号，返回字符串