AC54 usb从机

AC54的usb从机主要有三种模式：
存储器
PC摄像头
录像模式
统一由usb_app实现，数据结构在 `struct usb_app_handle` 中管理

```
struct usb_app_handle {
    u8 state;
    u8 mode;
    struct server *usb_slave;
    struct server *usb_host;
    struct server *ui;
    u8 *buf;
    u32 buf_size;
};
```

注册app

```
static const struct application_operation usb_app_ops = {
    .state_machine  = state_machine,
    .event_handler  = event_handler,
};

REGISTER_APPLICATION(app_usb) = {
    .name   = "usb_app",
    .action = ACTION_USB_SLAVE_MAIN,
    .ops    = &usb_app_ops,
    .state  = APP_STA_DESTROY,
};
```

状态机（state_machine）

APP_STA_CREATE – 初始化，打开ui服务；
APP_STA_START:

- **ACTION_USB_SLAVE_MAIN** – 打开菜单；
- **ACTION_USB_SLAVE_SET_CONFIG** – 菜单选择；
- **ACTION_USB_SLAVE_GET_CONFIG** – 获取配置（暂时无动作）。

APP_STA_STOP – 关闭菜单，关闭usb服务并释放资源；
APP_STA_DESTROY – 关闭ui服务。

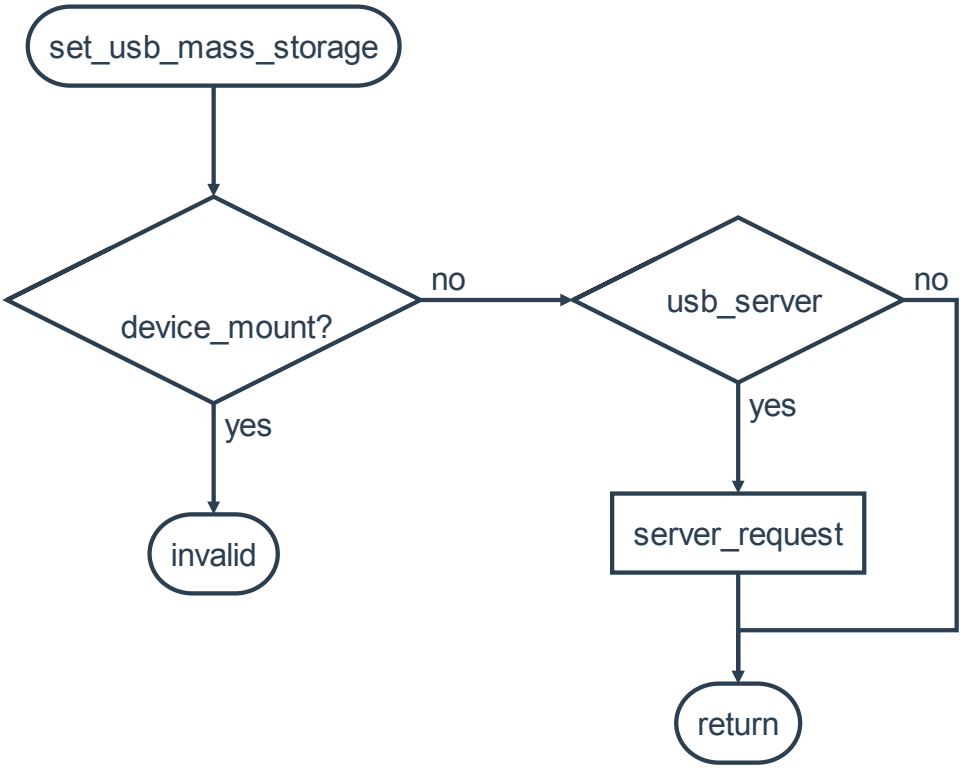
事件处理器（event_handler）

usb app下暂时没有需要处理的消息

存储器（mass storage）

菜单通过 `ACTION_USB_SLAVE_SET_CONFIG` 来选择的该功能，通过判断调用 `set_usb_mass_storage()` 函数。

set_usb_mass_storage有以下过程：



其中的主要部分为server_request的参数配置:

```
/*
 *设置请求参数：
 *类型：USB_MASS_STORAGE
 *挂载设备数量
 *挂载设备列表
 *状态：从机连接
 */
req.type = USB_MASS_STORAGE;
req.storage.dev_num = 1;
req.storage.dev = mass_storage_dev_list;
req.state = USB_STATE_SLAVE_CONNECT;
```

`dev_num` 表示mass storage挂载的设备数量
`dev` 表示设备的列表，支持1-8个设备数。

PC Camera

pc camera的设置流程与mass storage过程一致，请求的参数设置为：

```
        /*
         *设置请求参数
         *类型：USB_CAMERA (可增加isp工具，调屏工具选项)
         *设置camera的编码所需buffer、buffer长度
         *quality -- 设置图像质量
         */
        req.type = USB_CAMERA | USB_ISP_TOOL | USB_SCREEN_TOOL;
        req.camera.name = "video0";
        if (!__this->buf) {
            __this->buf = (u8 *)malloc(USB_CAMERA_BUF_SIZE);
            if (!__this->buf) {
                return -ENOMEM;
            }
        }
        __this->buf_size = USB_CAMERA_BUF_SIZE;
        req.camera.buf = __this->buf;
        req.camera.buf_size = __this->buf_size;
        req.camera.info = NULL;
        req.camera.quality = 1;

        req.state = USB_STATE_SLAVE_CONNECT;
```

`type` 表示请求的设备类型，除了USB_CAMERA之外还可以同时支持一些调试工具，例如ISP工具 `USB_ISP_TOOL` 、调屏工具 `USB_SCREEN_TOOL`
`req.camera.name = "video0"` 表示设备未前置前置摄像头
`buf` 、 `buf_size` 表示配置的视频流buffer
`info` 表示camera分辨率的列表，NULL默认为

```
    struct uvc_camera_info default_camera_info[] = {
        {0x500, 0x2D0}, // 1280x720
        {0x320, 0x258}, // 800x600
        {0x280, 0x1E0}, // 640x480
        {0x160, 0x120}, // 352x288
        {0x140, 0x0F0}, // 320x240
    };
```

`quality` 表示视频质量，目前分3个等级，0-低，1-中，2-高