# 基础配置

详见官方文档无线网络

# 配置 SDMC 接口和 WiFi 模组

Board options

[ * ] Using SDMC0 # 硬件连接哪个 SDMC 接口，选择哪个 SDMC

[ * ] Enable the interrupt of sdmc

[ * ] Using Wireless lan --->

[ * ] Using Realtek wlan driver --->

Select Realtek wlan0 modul (rtl8189)--- > # 以 RTL8189 为例

Realtek wlan0 paramete --->

(PC.7) realtek wlan0 power on gpio

(192.168.3.20) wlan0 ip addr

(192.168.3.1) wlan0 gateway addr

(255.255.255.0) wlan0 netmask

[*] Enable Realtek driver debug information

# 配置内核

Rt-Thread options --->

RT-Thread Kernel --->

(4) The priority level value of timer thread

(4096) The stack size of timer thread

RT-Thread Components --->

Device Drivers --->

(4096) The stack size for sdio irq thread

(5) The priority level value of sdio irq thread

# 使用第三方包中的lwIP

Local packages options --->

Third-party packages options --->

LwIP: light weight TCP/IP stack

[*] LwIP Examples --->

[*] Using net tools

[*] Enable DNS for name resolution

[*] Enable alloc ip address through DHCP

[*] Enable ping features

# ping代码完善

在`lwip\contrib\apps\ping\ping.c`,函数**cmd_ping**上有说明未实现DNS解析相关代码，将导致在msh控制台里只能ping通ip

地址、而不能ping通域名

解决方案：

- 在 `lwip/src/core/dns.c` 中修改DNS配置

*DNS*

```
void
dns_init(void)
{
  /* 手动设置DNS服务器地址 */
  ip_addr_t dnsserver1, dnsserver2;
  IP4_ADDR(&dnsserver1, 8, 8, 8, 8);          // 谷歌DNS
  IP4_ADDR(&dnsserver2, 114, 114, 114, 114); // 国内DNS
  dns_setserver(0, &dnsserver1);              // 设置为首选DNS
  dns_setserver(1, &dnsserver2);              // 设置为备用DNS

  /* 注释或删除原有的宏配置代码 */
  // #ifdef DNS_SERVER_ADDRESS
  //   ip_addr_t dnsserver;
  //   DNS_SERVER_ADDRESS(&dnsserver);
  //   dns_setserver(0, &dnsserver);
  // #endif
  /* 其余代码保持不变 */
 #if DNS_LOCAL_HOSTLIST
  dns_init_local();
#endif
```

```
}
```

- 在`lwip\contrib\apps\ping\ping.c`中增加DNS解析功能

*ping*

```
#include "lwip/dns.h"
static void ping_dns_found(const char *name, const ip_addr_t *ipaddr, void *arg)
{
  struct ping_msg *ctx = (struct ping_msg *)arg;
  if (ipaddr != NULL) {
    ctx->target_ip = *ipaddr;
    PING_PRINTF(("Resolved %s to ", name));
    ping_ip4_addr_print(&ctx->target_ip);
    PING_PRINTF(("\n"));
    ping_register(ctx);
  } else {
    PING_PRINTF(("DNS lookup failed for %s\n", name));
    aicos_free(MEM_DEFAULT, ctx);
  }
}

void cmd_ping(int argc, char **argv)
{
  static ip_addr_t remote_ip;
  int i;
  u32_t ping_count = DEFAULT_PING_TIMES;
  err_t err;
  struct ping_msg *dns_ctx = NULL;
```

```
  if (argc < 2) {
    PING_PRINTF(("Argument error: no destination address!\n"));
    goto ping_help;
  }
  else if (argc > 5) {
    PING_PRINTF(("Argument error: too many arguments\n"));
    goto ping_help;
  }
  if (!strcmp(argv[1], "help"))
    goto ping_help;

  /* Try to parse as IP address first */
  if ((remote_ip.addr = ipaddr_addr(argv[1])) == IPADDR_NONE) {
    /* Not an IP address, try DNS resolution */
    dns_ctx = aicos_malloc(MEM_DEFAULT, sizeof(struct ping_msg));
    if (!dns_ctx) {
      PING_PRINTF(("Memory allocation failed\n"));
      goto out;
    }
    memset(dns_ctx, 0, sizeof(struct ping_msg));
    PING_PRINTF(("Resolving %s...\n", argv[1]));
    err = dns_gethostbyname(argv[1], &dns_ctx->target_ip, ping_dns_found, dns_ctx);
    if (err == ERR_INPROGRESS) {
      /* Async DNS resolution started */
      dns_ctx->conut = (argc >=4 && !strcmp(argv[2], "-n")) ?
```

```
                        (u32_t)atoi(argv[3]) :
DEFAULT_PING_TIMES;
      goto out;
    }
    else if (err == ERR_OK) {
      /* Immediate resolution */
      PING_PRINTF(("Resolved %s to ", argv[1]));
      ping_ip4_addr_print(&dns_ctx->target_ip);
      PING_PRINTF(("\n"));
      remote_ip = dns_ctx->target_ip;
      aicos_free(MEM_DEFAULT, dns_ctx);
    }
    else {
      PING_PRINTF(("DNS lookup failed: %d\n",
err));
      aicos_free(MEM_DEFAULT, dns_ctx);
      goto out;
    }
  }
  /* 其余代码保持不变 */
ping_help:
  ping_usage();
}
```

# 使用RTT框架中的lwIP

# 配置

menuconfig配置如下：

- 在Rt-Thread options → RT-Thread Components → Network 下，使能**SAL**和**lwIP**。在lwIP配置中，**将版本设置为lwIP v2.1.2**，否则初始化wifi时会报错

```
[I] rtt_thread_enter()593 RTKTHREAD xmitThread
[I] rtt_thread_enter()593 RTKTHREAD RTW_CMD_THREAD
RTL871X: -871x_drv - dev_open, bup=1
[I] wifi_on()1124 WIFI initialized CPU Exception:
NO.4 x1: 400c3f62 x2: 401f32a0 x3: 401c6058 x4:
deadbeef x5: deadbeef x6: deadbeef x7: deadbeef
x8: 401d24ca x9: deadbeef x10: f528ffff x11:
401d24ca x12: 00000001 x13: 401f32a8 x14: 00000002
x15: ffffffff x16: 00000011 x17: deadbeef x18:
3004f888 x19: deadbeef x20: deadbeef x21: deadbeef
x22: deadbeef x23: deadbeef x24: deadbeef x25:
deadbeef x26: deadbeef x27: deadbeef x28: deadbeef
x29: deadbeef x30: deadbeef x31: deadbeef mcause :
0x38000004 mtval : 0x00000009 mepc : 0x400c3f70
mstatus: 0x80007880
```

- 配置RT_USING_NETDEV：`RT-Thread Components → Network → Network interface device`，否则在保存menuconfig配置时会出现警告`(RT_LWIP_USING_PING) selects NETDEV_USING_PING which has unmet direct dependencies (RT_USING_NETDEV)`
- 打开wifi debug：`Rt-Thread options-> RT-Thread Components -> Device Drivers-> Using Wi-Fi framework (RT_USING_WIFI [=y]) -> Enable WLAN Debugging Options (RT_WLAN_DEBUG [=y])`

- 打开pbuf设置：在wlan的配置选项里，打开`force pbuf transmission`。如果不打开，在wifi连接成功后总是会报错`CPU EXCEPTION:4`

# 使用

在msh控制台中，可以执行以下命令：

- 开启wifi：`wifi -d mode sta wlan0`
- 查看wifi相关命令：`wifi`
- `wifi help`
- `wifi scan [SSID]`
- `wifi join [SSID] [PASSWORD]`
- `wifi ap SSID [PASSWORD]`
- `wifi disc`
- `wifi ap_stop`
- `wifi status`
- `wifi smartconfig`

### 加入wifi时报错

日志显示WiFi连接成功，但在设置加密密钥后立即崩溃。具体是在发送EAPOL包、设置单播和组密钥之后触发的异常

问题排查`F:\Embedded\Artinchip\luban-lite\toolchain\bin> ./riscv64-unknown-elf-addr2line -e d13x.elf -afp 0x400c77c4`

**参考资料**

sockert组成部分_IOT-OS之RT-Thread（十六）--- WLAN管理框架 + AP6181(BCM43362) WiFi模块

# MQTT

LWIP的MQTT例程默认未启用DNS解析，在测试过程中，需将MQTT服务器提供的域名转换成IP地址

在lwIP中，通过以下步骤将DNS服务器地址传入到`dns.c`的代码段中：

---

## 方法1：通过`lwipopts.h`静态配置DNS地址

1. **在`lwipopts.h`中定义宏和DNS服务器地址**

   修改`lwipopts.h`，添加以下配置：

```
/* 启用DNS功能 */
#define LWIP_DNS 1

/* 定义DNS服务器地址（例如Google DNS） */
#define DNS_SERVER_ADDRESS(addr) IP4_ADDR(addr, 8, 8, 8, 8)
```

- `IP4_ADDR` 是lwIP提供的IPv4地址初始化宏，参数依次为地址指针和四个IPv4字节。

2. **验证代码逻辑**

   当启用 `DNS_SERVER_ADDRESS` 宏后， `dns.c` 中的代码段会自动生效，将DNS服务器设置为指定的地址（如 `8.8.8.8` ）。

---

## 方法2：在应用程序中动态设置DNS地址

如果需要在运行时动态设置DNS服务器（例如从配置文件读取），可以绕过宏定义，直接在代码中调用 `dns_setserver` 函数：

1. **在应用程序初始化代码中添加以下内容**

```
#include "lwip/dns.h"

void set_custom_dns_server(void) {
    ip_addr_t dns_server;
    IP4_ADDR(&dns_server, 8, 8, 8, 8);  // 设置DNS服务器地址
    dns_setserver(0, &dns_server);      // 设置第一个DNS服务器
}
```

   - 调用 `set_custom_dns_server()` 函数后，DNS服务器将被动态设置。

2. **禁用静态配置宏（可选）**

   如果不需要静态配置，确保 `lwipopts.h` 中**不定义** `DNS_SERVER_ADDRESS` 宏：

```
// #define DNS_SERVER_ADDRESS(addr)  // 注释或删除此行
```

---

## 关键解释

- `dns_setserver` **函数**：
  lwIP通过 `dns_setserver(uint8_t num, const ip_addr_t *dnsserver)` 设置DNS服务器， `num` 表示DNS服务器索引（通常0表示主DNS）。
- **IP地址初始化**：
  使用 `IP4_ADDR` 宏初始化IPv4地址，例如：

```
IP4_ADDR(&addr, 192, 168, 1, 1);  // 初始化为192.168.1.1
```

  对于IPv6，使用 `IP6_ADDR` 宏。

---

## 验证配置

1. **检查DNS服务器是否生效**

   在应用程序中调用以下代码打印DNS服务器地址：

   ```
   const ip_addr_t *dns = dns_getserver(0);
   printf("DNS Server: %s\n", ipaddr_ntoa(dns));
   ```

   输出应为配置的地址（如`8.8.8.8`）。

2. **确保DNS功能已启用**

   在`lwipopts.h`中确认以下宏已定义：

   ```
   #define LWIP_DNS 1
   ```

---

## 完整示例（静态配置）

```
// lwipopts.h
#define LWIP_DNS 1
#define DNS_SERVER_ADDRESS(addr) IP4_ADDR(addr, 8, 8, 8, 8)
```

通过上述任一方法，即可将DNS服务器地址正确传入到`dns.c`的代码段中。