Ethernet driver user guide

This document provides guideline to use Ethernet driver module in SDK.

Table of Contents

1 Introduction 3
2 Configuration for Ethernet mode 3
2.1 Set the Ethernet device state 3
2.2 Replace the example code 3
2.3 Check Interface information 4
2.4 Check Link state change 5
3 Commands description 6
3.1 “ATE1” set default interface 6
4 Ethernet API 7
4.1 ethernet_init 7
4.2 ethernet_free 7
4.3 ethernet_write 7
4.4 ethernet_send 8
4.5 ethernet_receive 8
4.6 ethernet_read 9
4.7 ethernet_address 9
4.8 ethernet_link 10
4.9 ethernet_set_link 10

1 Introduction

This document provides a guide to use Ethernet driver module provided in SDK. These API and related data structures are included in "**ethernet_api.h"** file.

2 Configuration for Ethernet mode

This sub-section lists the necessary configuration to use this Ethernet driver.

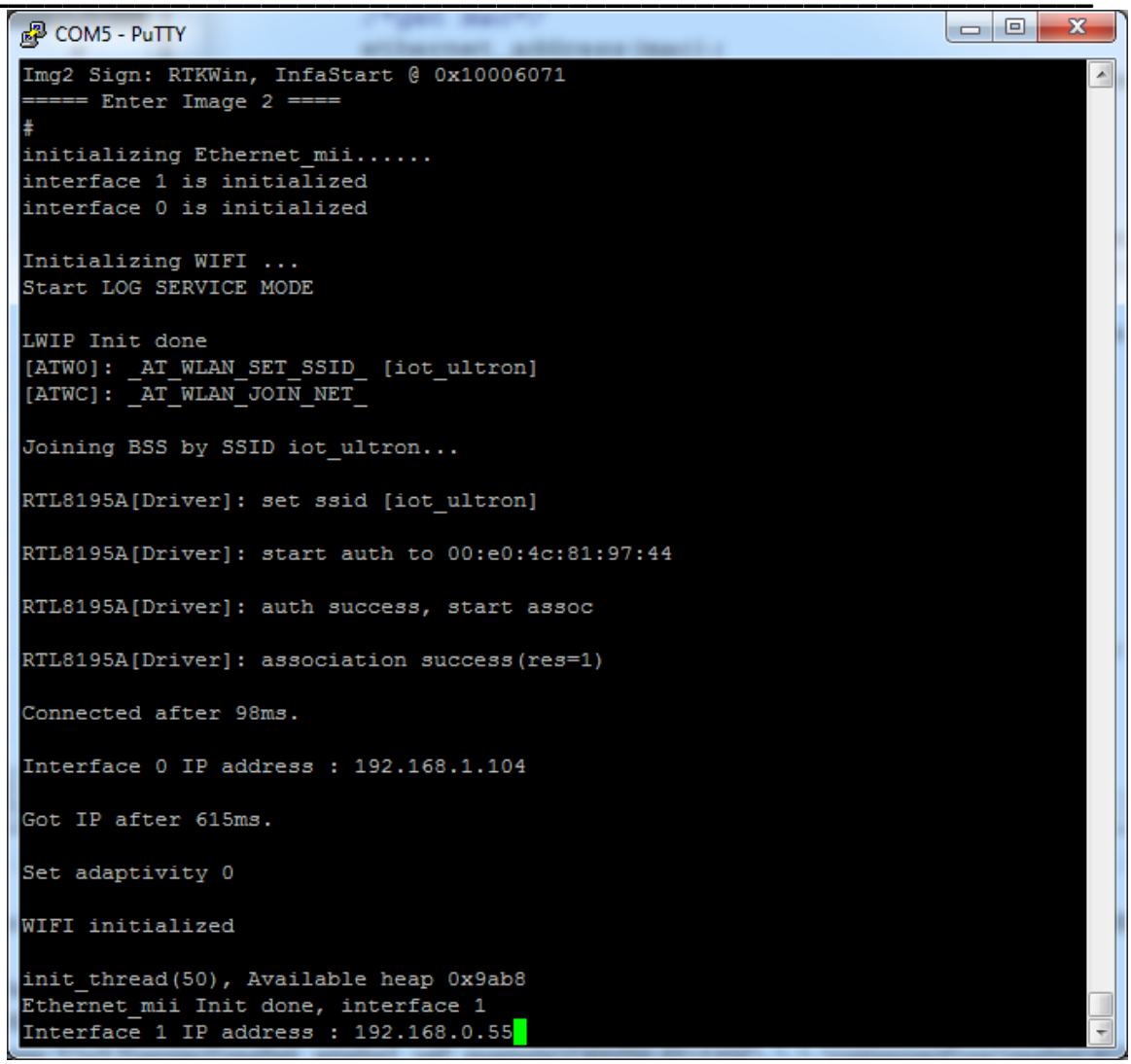
2.1 Set the Ethernet device state

```
#define CONFIG_ETHERNET   1
```

This macro is used to set the Ethernet device state. 1 indicates this is an Ethernet device and 0 means otherwise. It should be set to 1 to use this driver. It is defined in the following file: "project\realtek_ameba1_va0_example\inc\platform_opts.h"

2.2 Replace the example code

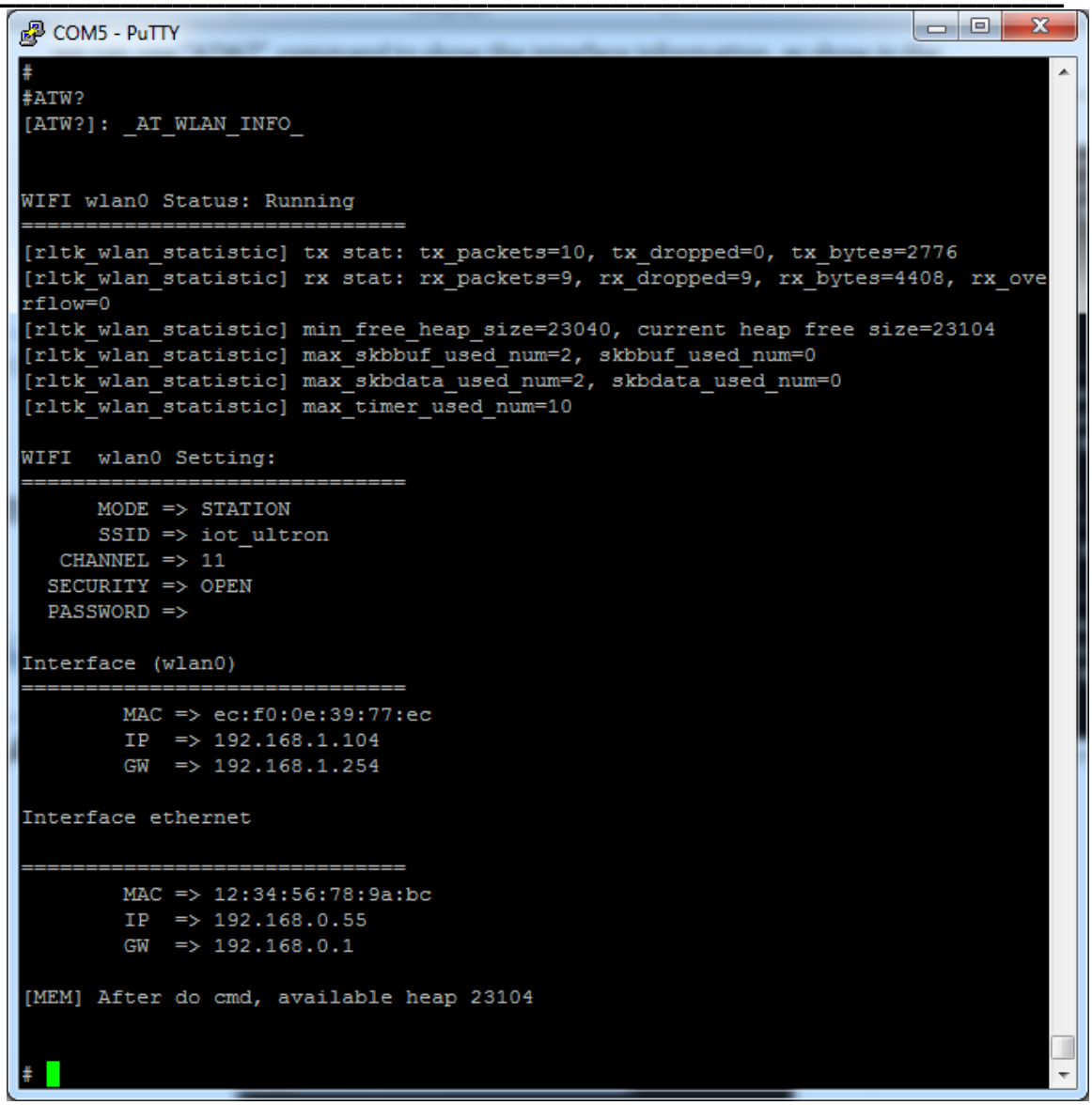
Copy the following file:
"project\realtek_ameba1_va0_example\example_sources\ethernet_mii\src\main.c" to "project\realtek_ameba1_va0_example\src" to replace the default main.c file. Then build and download the firmware, you could see the following information from terminal:



From the above log information, we could see that there are two interfaces initialized, the interface 0 is wlan with IP address “192.168.1.104”, and the interface 1 is Ethernet with IP address “192.168.0.55”

2.3 Check Interface information

We can use “ATW?” command to show the interface information, as show in the following figure:



2.4 Check Link state change

When the Ethernet cable is plugged out/in, the corresponding link change status is shown as follows:

```
Link Up

Interface 1 IP address : 192.168.0.55
#
#
```

3 Commands description

3.1 “ATE1” set default interface

Description: When wlan and Ethernet are both enabled, we can set and check the default interface.

Command Format: ATE1=NULL/0/1

Default Value: None

Response:

Set/check default interface	
Check default interface	NULL
Set wlan as default interface	0
Set ethernet as default interface	1

```
# ATE1
[ATE1]:Set/check the default interface
wlan is the default interface

[MEM] After do cmd, available heap 27304


# ATE1=1
[ATE1]:Set/check the default interface
ethernet is set to the default interface

[MEM] After do cmd, available heap 27304


# ATE1
[ATE1]:Set/check the default interface
Ethernet is the default interface

[MEM] After do cmd, available heap 27304


#
```

4 Ethernet API

This sub-section lists the provided API for Ethernet driver.

4.1 ethernet_init

This function initializes the Ethernet driver and should be called before using other related API.

Syntax

```
int ethernet_init(void);
```

Parameters

None.

Return Value

If the function succeeds, the return value is 0.

Remarks

None.

4.2 ethernet_free

This function is used to free the dynamic memory. It should be called when the Ethernet exits.

Syntax

```
void ethernet_free(void);
```

Parameters

None

Return Value

None

4.3 ethernet_write

This function writes a specified size to ethernet buffer.

Syntax

```
int ethernet_write(const char *data, int size);
```

Parameters

data

Pointer to the data to be written to Ethernet buffer.

size

Size of data to be written in byte

Return Value

Return the number bytes of data being written if succeed. Return -1 if size is too big.

Remarks

None

4.4 ethernet_send

This function sends out the data in ethernet buffer.

Syntax

```
int ethernet_send(void);
```

Parameters

None

Return Value

The size of data which has been sent

Remarks

None

4.5 ethernet_receive

This function receives data and stores in ethernet buffer .

Syntax

```
int ethernet_receive(void);
```

Parameters

None

Return Value

The size of data which has been received

Remarks

None

4.6 ethernet_read

This function reads data from ethernet buffer.

Syntax

```
int ethernet_read(char *data, int size);
```

Parameters

Data

Pointer to a buffer to store the data which has been read from ethernet buffer

Size

The size of data in byte to be read

Return Value

The actual size of data in byte which has been read

Remarks

ethernet_receive() should be called before this function to get the size of data to read.

4.7 ethernet_address

This function is used to get the ethernet address.

Syntax

```
void ethernet_address(char *mac);
```

Parameters

mac

Pointer to store the ethernet address

Return Value

None

4.8 ethernet_link

This function is used to check if link is up.

Syntax

```
int ethernet_link(void);
```

Parameters

None

Return Value

Value of 1 indicates the link is up. 0 indicates the link is down.

4.9 ethernet_set_link

This function is used to set the link state.

Syntax

```
void ethernet_set_link(int speed, int duplex);
```

Parameters

Speed

0: 10 Mbps; 1: 100 Mbps

Duplex:

0: half duplex; 1: full duplex

Return Value

None

Remarks

If the given parameter "**speed**" is not 0 or 1, or "**duplex**" is not 0 or 1, this function is used to initiate an auto negotiate state to determine the actual link state.