AN01677

RHF76-052AM Design Guide

V1.0

Document information

Info	Content
Keywords	*LoRaWAN, UART, Extremely Auto Low Power*
Abstract	RHF76-052AM design guide

Content

Figures

Tables

1 Preface

RHF76-052AM is a UART interface LoRaWAN AT Modem with auto ultra-low power feature. This document applies RHF76-052AM firmware equal to or higher than v2.0.8.
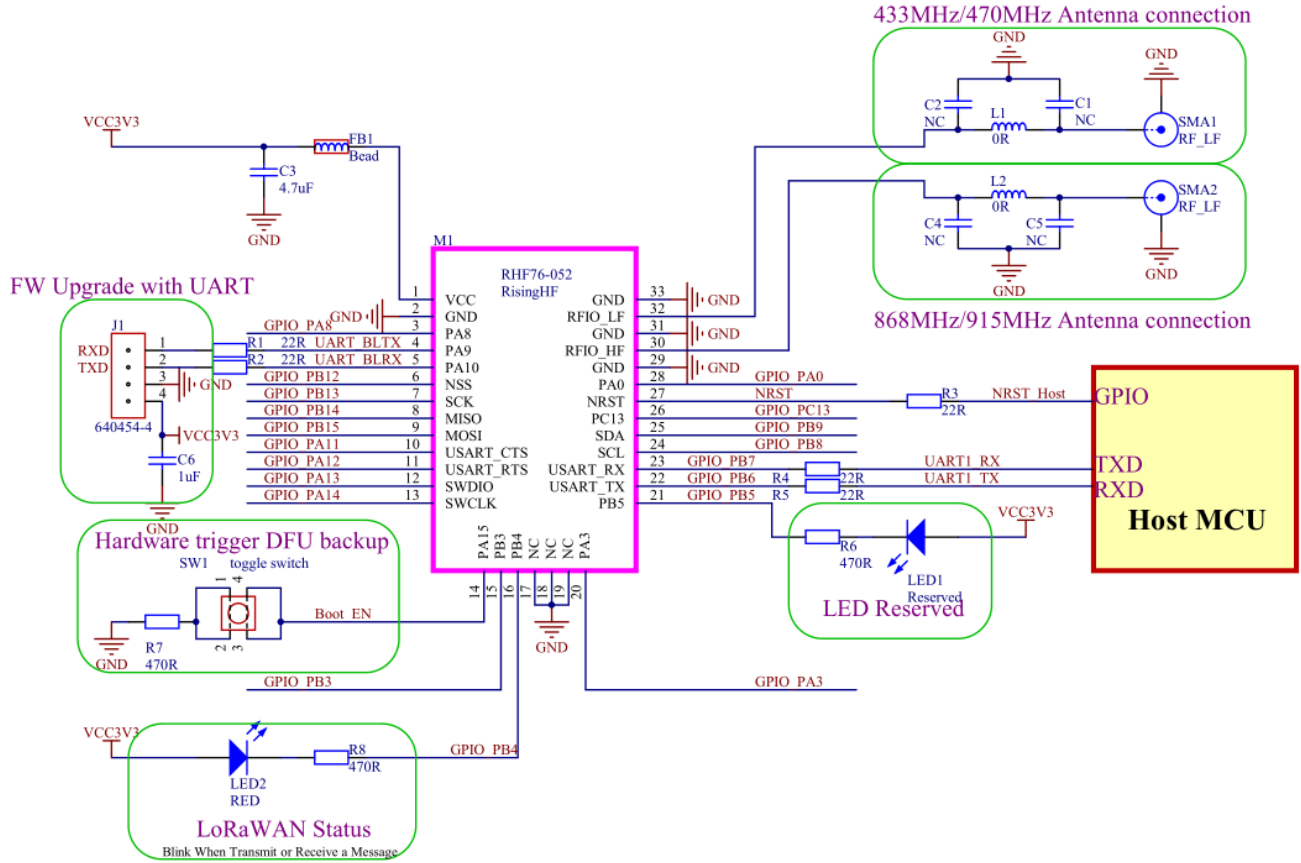

Figure 1 RHF76-052AM reference design

2 Application Information

2.1 Pin Definition

RHF76-052	STM32L0	RHF76-052AM	Comments
1	VCC	VCC	
2	GND	GND	
3	PA8		
4	PA9	UART_BLTX	Bootloader UART TX (115200,8,n,1)
5	PA10	UART_BLRX	Bootloader UART RX (115200,8,n,1)
6	PB12		
7	PB13		
8	PB14		
9	PB15		
10	PA11		
11	PA12		
12	PA13		
13	PA14		
14	PA15	BLEN	Low to enable bootloader during power up
15	PB3		
16	PB4	LED2	LoRaWAN RX/TX Status
17	-	-	
18	-	-	
19	-	-	
20	PA3		
21	PB5	LED1	Reserved LED
22	PB6	UART_TX	AT command interface UART TX (9600,8,n,1)
23	PB7	UART_RX	AT command interface UART RX (9600,8,n,1)
24	PB8		
25	PB9		
26	PC13		
27	NRST		
28	PA0		
29	GND	GND	
30	RFIO_HF	RFIO_HF	High frequency band RF port
31	GND	GND	
32	RFIO_LF	RFIO_LF	Low frequency band RF port
33	GND	GND	

Table 1 RHF76-052AM pin definition

2.1 UART Configuration

2.1.1 AT Mode

Interface:Pin22-PB6 (TX) / Pin23-PB7 (RX)
Baud Rate: 9600
Data Bits: 8
Stop Bits: 1
Parity: None

2.1.2 Bootloader Mode

Interface: Pin4-PA9 (TX) / Pin5-PA10 (RX)
Baud Rate: 115200
Data Bits: 8
Stop Bits: 1
Parity: None

For more information about how to upgrade firmware refer to <u>UM01518</u>

2.2 Reset

If the application requires the disconnection of VCC from the RHF76-052AM module or send
AT+RESET command, user should wait for 2s from of the end of the POR cycle before commencing
communications over the UART port. RHF76-052AM use this 2s to initial itself.

Example: (AT+RESET)
```
printf("AT+RESET");
delay_ms(2000);
```

Example: (Power On)
```
// Wait until RHF76-052AM is ready
delay_ms(2000);
```

2.2.1 NRST

Two millisecond (2ms) negative pulses on NRST pin will trigger a reset of RHF76-052 module, it
performs the same as "AT+RESET".
This pin is optional to connect with host MCU, if it is connected the pin of host MCU is expected to set
to floating after reset is triggered.

2.3 Sleep Mode

While RHF76-052AM is in sleep mode, RHF76-052AM TX/RX pins status will be changed to save
power. Details:

RHF76-052AM	Status
UART TX	Floating
UART RX	Input weak pull up

Table 2 RHF76-052AM UART pin sleep mode status

Host controller is expected to set HOST TX pin floating. As RHF76-052AM TX pin is set to floating, so HOST RX status is not mandatory, customer could decide it on demand.

Host MCU	Status
UART TX	Floating
UART RX	Depends on HOST MCU

Table 3 Expected Host MCU UART pin sleep mode status

For more information about how to use auto low power mode refer to PS01509 V3.1+.

2.4 LED

Event	LED Status
Send packet	LED2 on ~100ms
ACK or packet received	LED2 on ~300ms
Join request	LED2 on ~800ms
Joined	LED2 on ~600ms

Table 4 LED status

2.5 Input Timeout

RHF76-052AM AT engine expects to receive '\n' to identify end of a frame for each of the command, if the TIMEOUT value time passed after last received character, an input timeout event will occur.

```
+INFO: Input timeout
+AT: OK
```

Input timeout feature enable AT engine to flush input commands in a predefined time to make sure it will not concatenate two commands wrongly.

For terminal tool (PuTTY, Teraterm) users, there is way to disable this feature. It is RisingHF official recommend to try SSCOM to tool.
Disable input timeout:

```
AT+UART=TIMEOUT,0
```

NOTE: UART TIMEOUT status is not saved v2.0.8.

3 ATCTL API

RisingHF ATCTL API package is available for customers who integrate RHF76-52AM in their design.

To use ATCTL software package user need implement all function in hal-atctl-template.c. And there is detailed description in hal-atctl.h to explain usage of each function and macro.

3.1 DBG

```
/*!
 * main.c debug log information
 */
#define DBG(x...)                           printf(x)
```

3.2 MUTEX

```
/*!
 * Enable protection of critical code, usually be disable global interrupt
 */
#define HAL_ATCTL_ENTER_MUTEX()           __disable_irq()

/*!
 * End protection of critical code, recove the interrupt
 */
#define HAL_ATCTL_EXIT_MUTEX()            __enable_irq()
```

3.3 Callback Function Prototype

```
/*!
 * atctl driver callback function pointer type
 */
typedef void (*atctl_cb_t)(uint8_t);
```

3.4 UART TX

hal_atctl_printf and hal_atctl_putchar will be called by ATCTL to send data to RHF76-052AM.

```
/*!
 * atctl driver call this function to send data to UART
 * printf output must be redirected to UART, refer compiler manual about how
 * to redirect printf
 */
#define hal_atctl_printf(x...)            printf(x)

/*!
 * UART low level TX one byte function
 */
void hal_atctl_putchar(uint8_t c);
```

3.5 UART RX

User need implement hal_atctl_init function and supply interface for atctl driver to register atctl rx_byte function.

```
/*!
 * atctl driver HAL layer initialize, this function will register atctl callback
 * function to the UART driver, UART driver must receive all data returned by
 * RHF76-052AM module
 */
void hal_atctl_init(atctl_cb_t cb);
```

The way it is called by atctl driver:

```
void atctl_init(bool wakeup)
{
    hal_atctl_init(atctl_rx_byte);

    atctl_reset();

    atctl_rx_tmp_wr_index = 0;
    atctl_rx_tmp_rd_index = 0;
    atctl_wakeup = wakeup;
}
```

3.6 System Time

These low level functions are mandatory for ATCTL driver.

```
/*!
 * return system tick, if the HOST MCU tick is not 32bits,
 * user should make hal_get_timeout function to adapt it.
 */
uint32_t hal_get_tick(void);

/*!
 * return difference between current time and an elapsed time
 * old_time    - an elapsed time
 */
uint32_t hal_get_timeout(uint32_t etime);

/*!
 * delay some time, unit is millisecond (ms)
 * old_time    - an elapsed time
 */
void hal_delay_ms(uint32_t ms);
```

3.7 Optional

```
/*!
 * HOST mcu initialize, it is optional to implement this function
 */
void hal_sys_init(void);

/*!
 * HOST mcu lowpower event, need to be polled in main loop,
 * it is optional to implement this function, user could local lowpower
 * management APIs to replace this one
 */
void hal_sys_lowpower(void);

/*!
 * This function is used to generate a periodic event, main.c will detect
 * this event, to use this event to trigger OTAA JOIN or send message periodicly
 */
bool hal_evt_flag(void);
```

4 FAQ

1. How to check configuration?

Each configuration except the keys (AppKey/NWKSKEY/APPSKEY) could be queried by user through query format command. Like:

```
AT+ID
AT+CH
AT+RXWIN1
AT+RXWIn2
AT+DR
AT+LW=ULDL                // Check uplink and downlink counter
AT+UART=TIEMOUT
```

Revision

V1.0 2016-07-25
        + Creation

Please Read Carefully:

Information in this document is provided solely in connection with RisingHF products. RisingHF reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All RisingHF products are sold pursuant to RisingHF's terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the RisingHF products and services described herein, and RisingHF assumes no liability whatsoever relating to the choice, selection or use of the RisingHF products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by RisingHF for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN RISINGHF'S TERMS AND CONDITIONS OF SALE RisingHF DISCLAIMS ANY EXPRESS OR IMPLIEDWARRANTY WITH RESPECT TO THE USE AND/OR SALE OF RisingHF PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIEDWARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWSOF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

RISINGHF PRODUCTS ARE NOT DESIGNED OR AUTHORIZED FOR USE IN: (A) SAFETY CRITICAL APPLICATIONS SUCH AS LIFE SUPPORTING, ACTIVE IMPLANTED DEVICES OR SYSTEMS WITH PRODUCT FUNCTIONAL SAFETY REQUIREMENTS; (B) AERONAUTIC APPLICATIONS; (C) AUTOMOTIVE APPLICATIONS OR ENVIRONMENTS, AND/OR (D) AEROSPACE APPLICATIONS OR ENVIRONMENTS. WHERE RISINGHF PRODUCTS ARE NOT DESIGNED FOR SUCH USE, THE PURCHASER SHALL USE PRODUCTS AT PURCHASER'S SOLE RISK, EVEN IF RISINGHF HAS BEEN INFORMED IN WRITING OF SUCH USAGE, UNLESS A PRODUCT IS EXPRESSLY DESIGNATED BY RISINGHF AS BEING INTENDED FOR "AUTOMOTIVE, AUTOMOTIVE SAFETY OR MEDICAL" INDUSTRY DOMAINS ACCORDING TO RISINGHF PRODUCT DESIGN SPECIFICATIONS. PRODUCTS FORMALLY ESCC, QML OR JAN QUALIFIED ARE DEEMED SUITABLE FOR USE IN AEROSPACE BY THE CORRESPONDING GOVERNMENTAL AGENCY.

Resale of RisingHF products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by RisingHF for the RisingHF product or service described herein and shall not create or extend in any manner whatsoever, any liability of RisingHF.

http://www.risinghf.com