Fibocom
PERFECT WIRELESS EXPERIENCE

MTC

General Configurable Solution User Guide

V1.0

Disclaimer

Any action you take in the course of using this document is at your own risk, and Fibocom shall not be liable for any damages or losses under any circumstances. Due to product version upgrade or other reasons, Fibocom reserves the right to modify any information in this document at any time without prior notice and any responsibility. Unless otherwise agreed, all statements, information and suggestions in this document do not constitute any express or implied guarantee.

This document may include the third-party information covering products, services, software, data, and so on. Fibocom does not control and assumes no responsibility for the third-party content, including but not limited to the accuracy, compatibility, reliability, availability, legitimacy, appropriateness, performance, non-infringement, and status update, unless otherwise specified in this document. Fibocom does not provide any guarantee or authorization for the third-party content mentioned or referenced in this document. If you need a third-party license, obtain it in an authorized or legal way, unless otherwise specified in this document.

Copyright Notice

Unless specially authorized by Fibocom, the recipient of the documents shall keep the documents and information received confidential, and shall not use them for any purpose other than the implementation and development of this project. Without the written permission of Fibocom, no unit or individual shall extract or copy part or all of the contents of this document without authorization, or transmit them in any form. Fibocom has the right to investigate legal liabilities for any offense and tort in connection with violation of confidentiality obligations, or unauthorized use or malicious use of the said documents and information in other illegal forms.

Trademark Statement

Fibocom The trademark is registered and owned by Fibocom Wireless Inc.

Other trademarks, product names, service names and company names appearing in this document are owned by their respective owners.

Contact Information

Website: https://www.fibocom.com

Address: 10/F-14/F, Block A, Building 6, Shenzhen International Innovation Valley, Dashi First Road, Xili Community, Xili Subdistrict, Nanshan District, Shenzhen

Tel: 0755-26733555

Contents

Change History

V1.0 (2024-11-04)	Initial version.

Applicable Model

No.	Applicable Model	Description
1	FG132	MTC

1 Overview

This document introduces the workflow and usage method of the software general configurable plan (hereinafter referred to as FIBCFG), including configuration file modification, use of the fibocfg_tool, and validation logic of configuration items. This document aims to guide users to modify project NV and the initial values of other configuration items.

The following figure shows the flowchart of FIBCFG.

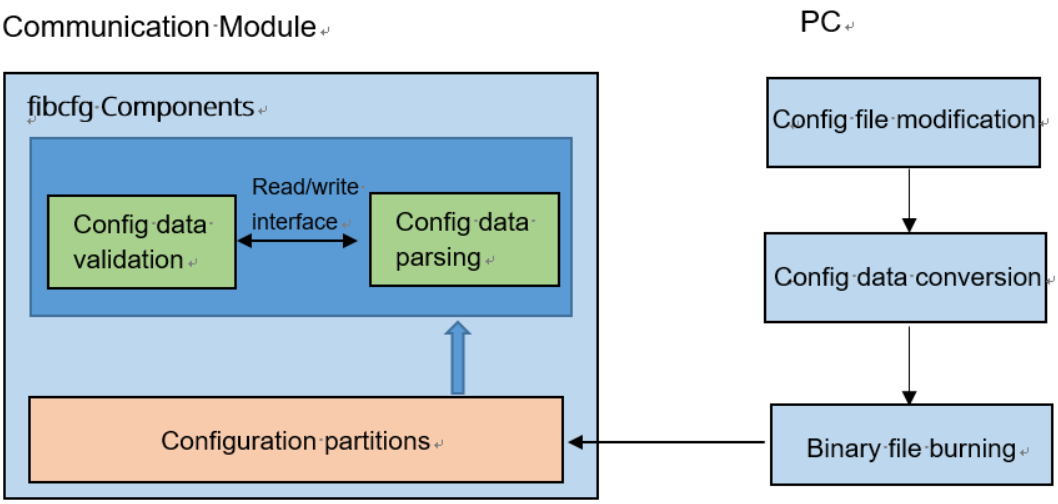

Figure 1. FIBCFG workflow

2 Configurable File

2.1 Configuration File Description

The configuration file is a text file in XML format. It is the input of the FIBCFG process. You can edit and modify this file according to your needs. The content of the configuration file is divided into two major elements: "**version**" and "**configuration**", which include configuration feature version number, configuration data version number, configuration item name, ID, and value. The following table provides the details.

Table 1. Configuration file elements

Element	Function	Remarks
magic	Magic number, used as FIBCFG internal information identification.	You do not need to pay attention to this element.
hash_algorithm	Hash algorithm of burning files to facilitate algorithm compatibility among platforms.	You do not need to pay attention to this element.
configrue_feature_ver	Configuration feature version number, FIBCFG solution iteration version	You do not need to pay attention to this element.
configrue_data_ver	Configuration data version number. When a configuration item is modified, you need to update this version number, which is used to determine the internal validity logic of the device.	--
configrue_binary_ver	Binary type version number	You do not need to pay attention to this element.
name/id	Configuration item name and ID, which have a one-to-one mapping.	You are not allowed to modify it.
type	Configuration item type, including int, string, and binary. Each configuration item has a fixed type.	You are not allowed to modify it.
min/max	Only applies to INT-type configuration items, indicating the settable range of a configuration item.	You are not allowed to modify it.
val	The value of the configuration item. You can modify it as needed. The value of INT-type configuration items cannot exceed **min/max**.	--

Following is an example of the configuration file, which contains three types of configuration items: int, string, and binary:

```
<?xml version="1.0" encoding="utf-8"?>
<fib_config>
    <version>
        <magic>FIBO</magic>
        <!-- supported algorithm options: MD5 SHA256 -->
        <hash_algorithm>MD5</hash_algorithm>
        <configrue_feature_ver>001.01.01.0001</configrue_feature_ver>
        <configrue_data_ver>0000.000.0000.000</configrue_data_ver>
        <configrue_binary_ver>0000</configrue_binary_ver>
    </version>
    <configurations>
        <setting name="USBMODE" id="00BB0000">
            <type>int</type>
            <min>20</min>
            <max>43</max>
            <val>36</val>
        </setting>
        <!--  only for example
        <setting name="DHCPDNS" id=00BB00D1">
            <type>string</type>
            <val>192.168.0.1</val>
        </setting>
        <setting name="README_FILE" id="00BB0002">
            <type>binary</type>
            <val>./README.md</val>
        </setting>
        -->
    </configurations>
</fib_config>
```

2.2 Modify Configuration File

2.2.1 Modify Parameters

The configuration file is generally released with the project SDK, and the file name is f_cust_devcfg.xml. The name may vary depending on projects. The name of the specific project prevails.

Procedure:

1. Use a text editor to open the configuration file. An XML editor is recommended.
2. Find the element that needs to be modified, modify the val value to the target value, and ensure that

the current element is not commented.

3. Save the file.

You can query configuration items supported by the current project by using the AT+GTCFGLIST command. For details, see Chapter 6.

2.2.2 Modify Version Number

The FIBCFG component determines whether the configuration parameters need to be validated by comparing the last three digits of configure_data_ver in the burning file. Therefore, after modifying the configuration items in the "configurations" element, you must modify the last three digits of confirue_data_ver in the "version" element. You can increment the digit by 1 to ensure it is different from the current configure_data_ver in the device. Do not modify other information in the "version" element.

Example:

Take the above configuration file as an example. If you need to modify the value of USBMODE to 37, the modified configuration file is as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<fib_config>
    <version>
        <magic>FIBO</magic>
        <!-- supported algorithm options: MD5 SHA256 -->
        <hash_algorithm>MD5</hash_algorithm>
        <configrue_feature_ver>001.01.01.0001</configrue_feature_ver>
        <configrue_data_ver>0000.000.0000.001</configrue_data_ver>
        <configrue_binary_ver>0000</configrue_binary_ver>
    </version>
    <configurations>
        <setting name="USBMODE" id="00BB0000">
            <type>int</type>
            <min>20</min>
            <max>43</max>
            <val>37</val>
        </setting>
    </configurations>
</fib_config>
```

- The value of INT-type parameters must be within the range of min/max, and that of other types must meet the product requirements.
- During the modification, you must ensure the XML format is correct (you are advised to use an XML editor for format detection after the modification). Otherwise, parsing may fail and binary burning files may fail to be generated.

3 Configurable Conversion Tool

3.1 Tool Introduction

fibocfg_tool is a lightweight tool developed by Fibocom. It is used to convert the configuration file into binary burning file. By downloading the burning file to a device that supports the FIBCFG function, the default value of the device can be modified. You can download fibocfg_tool from Fibocom Information Center or obtain it from Fibocom technical support.

3.2 Environment Requirements

fibocfg_tool is released as an executable binary file with no other special dependencies. It can run directly in Linux or Windows environment. The recommended OS version is Ubuntu 14.04/Windows 10 or later.

3.3 Usage

You can use a combination of multiple options to enable the fibocfg_tool to realize functions such as burning file generation, reverse parsing of burning file, query of specified configuration items in the burning file. During tool execution, key debugging information is printed. The following table describes options and functions. You can also use the **-h** option to view the user guide. The tool options and usage methods are the same for Linux and Windows.

Table 2. Tool options

Option	Function
-i <cfg xml file path>	Specifies the configuration file path, that is, the tool will convert the file in this path into the burning file.
-o <out file path>	Specifies the generation path of the burning file. If this option is empty, the burning file will be generated in the **fibocfg_tool** directory by default, and the default file name is **config_binary.bin**.
-p <config binary file>	Performs reverse parsing on the specified burning file and displays all configuration items and their values in it.
-d <ID>	Parses the specified configuration item. This option must be used in combination with the **-p** option.
-v	Queries fibocfg_tool version.
-h	User guide

Example:

1. Run the following command to convert the configuration file into a burning file.

```
./fibocfg_tool -i fibocom_config.xml -o binary_itlv.bin
```

```
input_xml_name: fibocom_config.xml
output_tlv_name: binary_itlv.bin

<------------------------------ setting infor ------------------------------>
Magic: FIBO
hash_algorithm name: MD5,index: 1
Feature ver: 001.01.01.0001
Data ver: 0000.000.0000.000
binary ver: 0000
<--------------------------------------------------------------------------->


Start parsing XML file
----------------------

-------------------- Index[1] ----------------------
name = USBMODE, id = 00BB0000
type = int
val = 36

ITLV raw data (16 bytes):
+--------------------------------------------------------------------------+
| Offset  : 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F   0123456789ABCDEF |
+--------------------------------------------------------------------------+
| 00000000: FD FD 00 BB 00 00 00 01 04 00 00 00 24 00 00 00   ............$... |
+--------------------------------------------------------------------------+

Start make binary file
----------------------

Make binary file successfully!!!
```

2. Run the following command to reversely parse the binary_itlv.bin file.

```
./fibocfg_tool -p binary_itlv.bin
```

```
input_binnary_name: binary_itlv.bin

<-------------------- binnary info --------------------->
make time: 2024-11-06 15:38:11
feature version: 001.01.01.0001
data version: 0000.000.0000.000
binnary version: 0000
<------------------------------------------------------->

items number: 1
---------------------- Index[1] ----------------------
id = 00BB0000
type = int
val = 36
```

3. Run the following command to query the value of the AABB0001 configuration item in the binary_itlv.bin file.

```
./fibocfg_tool -p binary_itlv.bin -d 00BB0000
```

```
input_binnary_name: binary_itlv.bin

---------------------- item info ----------------------
id = 00BB0000
type = int
val = 36
```

4. Run the following command to query the fibocfg_tool version.

```
./fibocfg_tool -v
```

```
fibocom configuration tool
==========================
Version 1.0.2
Built on May  9 2024
```

4 Burn Binary File

The generated binary burning file must be written into the device to trigger the validation logic inside the device. There are two configurable burning partitions reserved inside the device: One is the default partition and the other is the customer partition. You only need to pay attention to the customer partition. You can write the burning file into a single partition, or burn the entire package into the device.

Different device platforms have different writing methods. Please refer to the burning instruction document of the specific device. This document does not provide details.

5 Validate Configurable File

5.1 Description of Validation Logic

After the burning file is written to the device, restart the device to trigger the FIBCFG component logic in the device. After the burning file is verified, the FIBCFG component will first compare the configuration data version in the burning file with the configuration data version saved in the device. If the two match, the FIBCFG component will start the validation process, traverse and parse all the configuration items in the burning file and write them one by one. After writing is completed, the configuration data version inside the device will be updated to the configuration data version in the burning file. At this point, the entire FIBCFG function ends. After the device restarts next time, the validation process will not be triggered again because the configuration data version in the device is consistent with that in the burning file. The entire logic is shown in the figure below.

In the debugging phase, after writing the burning file, you can issue AT+GTFORCECFG through the AT port to trigger forced validation. This command can directly start the FIBCFG component for traversal to take effect without the need to compare the configuration data version or restart the device. It can save the configuration data version in the burning file to the device. For the specific command, see Chapter 6.

After the device validation logic is executed, the device will automatically restart to ensure that all configuration items take effect.

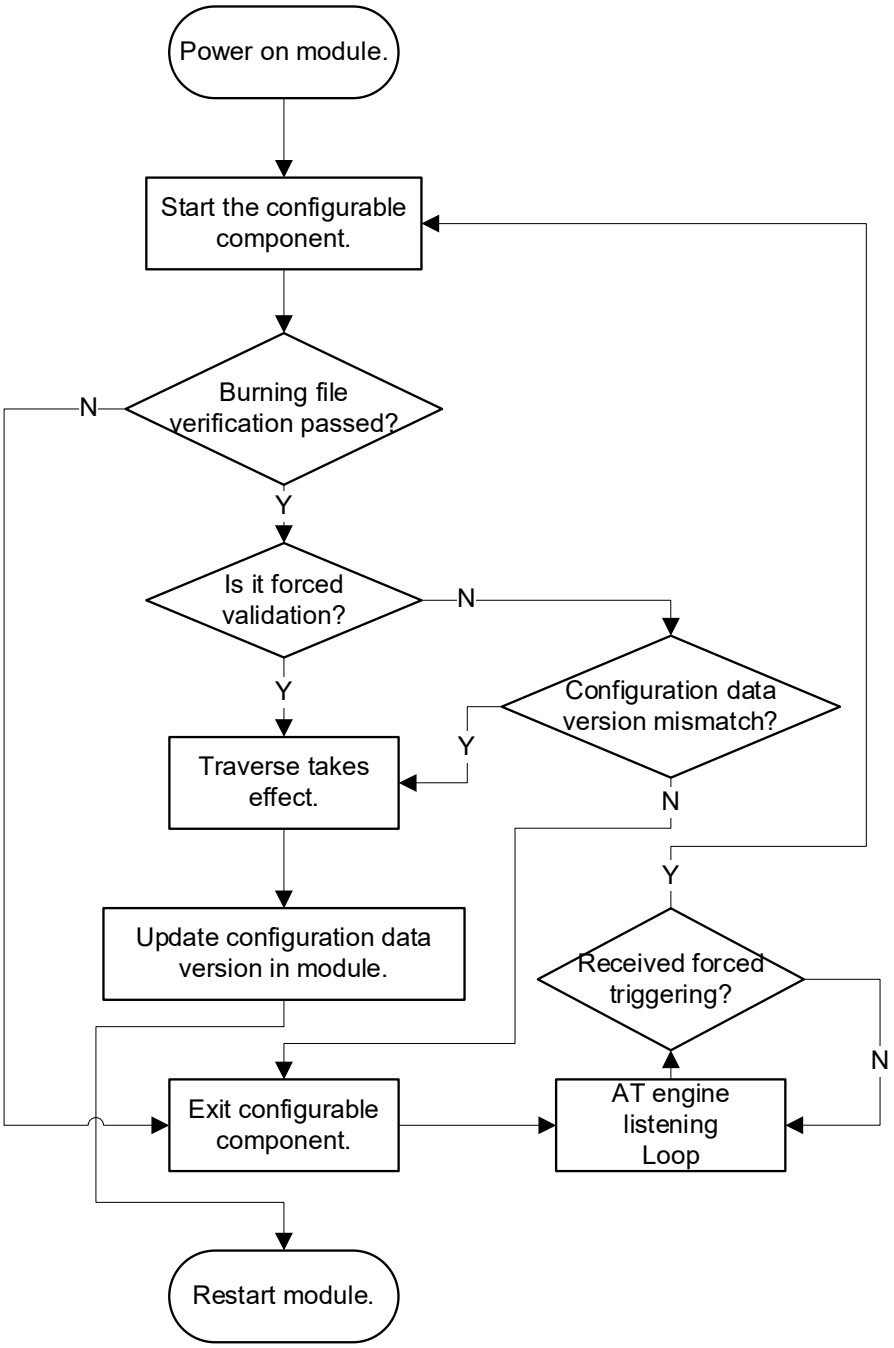

Figure 2. FIBCFG component validation logic

5.2 Query Validation Result

After the validation logic is completed, you can issue AT+GTCFGVER through the AT port to query the configuration data version in the device to confirm whether the validation is successful. If the configuration data version of the burning partition is consistent with that in the burning file, the configurations have taken effect. Otherwise, the configurations did not take effect. For details, see Chapter 6.

6 FIBCFG Debugging Commands

To facilitate debugging, the FIBCFG component provides a series of debugging commands, such as query of configuration data version, trigger of forced validation, and query of supported configuration items.

6.1 AT Commands for Querying Configuration Data Version

Type	Command	Response
Setting command	AT+GTCFGVER=<index>	+GTCFGVER: <partition>,"<version>" OK
Query command	AT+GTCFGVER?	+GTCFGVER: <partition>,"<version1>", "<version2>" OK

- Parameters

Parameter Name	Description	Value
index	Version number definition	Range: 0 to 1 (expandable) Unsigned type, decimal value 0: configuration feature version 1: configuration data version
partition	Partition	Range: 0 to 1, unsigned type, decimal value 0: default burning partition 1: customer burning partition
version1	Configuration feature version	A string containing version number information
version2	Configuration data version	A string containing version number information

- Example

```
AT+GTCFGVER=1
+GTCFGVER: 0,"SD35.001.0001.001"
+GTCFGVER: 1,"SD35.000.0000.000"

OK
```

```
AT+GTCFGVER?
+GTCFGVER: 0,"001.01.01.0001","SD35.001.0001.001"
+GTCFGVER: 1,"001.01.01.0001","SD35.000.0000.000"

OK
```

6.2 AT Commands for Triggering Forced Validation

Type	Command	Response
Setting command	AT+GTFORCECFG	Response 1: OK Response 2: ERROR

6.3 AT Commands for Querying Supported Configuration Items

Type	Command	Response
Query command	AT+GTCFGLIST?	<index>,<id>,”<name>” OK

- Parameters

Parameter Name	Description	Value
index	Configuration item index	Type: Integer
ID	Configuration item ID	Type: hexadecimal value
name	Configuration item name	Type: string

- Example

```
AT+GTCFGLIST?
0,00BB0000,"USBMODE"
1,00BB0080,"UARTBAUD"
2,00BB0020,"SIM0DETECT"

OK
```

7 FAQ

Table 3. FAQ and troubleshooting method

Problem	Troubleshooting Method
Default value does not take effect as expected.	• Check whether the configuration data version in the configuration file has been modified. Make sure that the last three digits are different from those in the device. You can query the configuration data version of the current device using the AT+GTCFGVER command. • Check whether the binary burning file is successfully burned into the device. • Check whether the partition for binary burning is the customer partition reserved for the current project. • For whole package burning, check whether the binary burning file is successfully burned into the whole package.
Configurations took effect successfully but the default values of some configuration items are not modified.	Check whether the values of these configuration items in the configuration file meet the range requirements.