BG95&BG77&BG600L Series
FILE Application Note

LPWA Module Series

Version: 1.1

Date: 2020-10-29

Status: Released

Our aim is to provide customers with timely and comprehensive service. For any assistance, please contact our company headquarters:

Quectel Wireless Solutions Co., Ltd.
Building 5, Shanghai Business Park Phase III (Area B), No.1016 Tianlin Road, Minhang District, Shanghai 200233, China
Tel: +86 21 5108 6236
Email: info@quectel.com

Or our local office. For more information, please visit:
http://www.quectel.com/support/sales.htm.

For technical support, or to report documentation errors, please visit:
http://www.quectel.com/support/technical.htm
Or email to support@quectel.com.

General Notes

Quectel offers the information as a service to its customers. The information provided is based upon customers' requirements. Quectel makes every effort to ensure the quality of the information it makes available. Quectel does not make any warranty as to the information contained herein, and does not accept any liability for any injury, loss or damage of any kind incurred by use of or reliance upon the information. All information supplied herein is subject to change without prior notice.

Disclaimer

While Quectel has made efforts to ensure that the functions and features under development are free from errors, it is possible that these functions and features could contain errors, inaccuracies and omissions. Unless otherwise provided by valid agreement, Quectel makes no warranties of any kind, implied or express, with respect to the use of features and functions under development. To the maximum extent permitted by law, Quectel excludes all liability for any loss or damage suffered in connection with the use of the functions and features under development, regardless of whether such loss or damage may have been foreseeable.

Duty of Confidentiality

The Receiving Party shall keep confidential all documentation and information provided by Quectel, except when the specific permission has been granted by Quectel. The Receiving Party shall not access or use Quectel's documentation and information for any purpose except as expressly provided herein. Furthermore, the Receiving Party shall not disclose any of the Quectel's documentation and information to any third party without the prior written consent by Quectel. For any noncompliance to the above requirements, unauthorized use, or other illegal or malicious use of the documentation and information, Quectel will reserve the right to take legal action.

Copyright

About the Document

Revision History

Version	Date	Author	Description
1.0	2019-09-02	Alfred LI	Initial
1.1	2020-10-29	Matt YE/ Egbert XU	1. Enabled EUFS (Extended User File Storage on application side) for the module. 2. Added the value 3 for <mode> in AT+QFOPEN. 3. Added AT+QFCRC and AT+QFCPY. 4. Added an example to illustrate how to write and read an EUFS file (Chapter 3.2.2).

Contents

Table Index

1 Introduction

Quectel BG95 series, BG77 and BG600L-M3 modules provide AT commands to operate files on different physical storage mediums. This document is a reference guide to these commands.

The supported storage mediums are as follows:

- **UFS**: User File Storage directory on modem side. It is a special directory on the flash file system.
- **EUFS**: Extended User File Storage on application side.

The file name indicates the storage location. When the file name begins with “EUFS:”, it means the file is located in EUFS. If the file name begins with “UFS:” or there are no prefix characters in the file name, the file is located in UFS.

1.1. Applicable Modules

Table 1: Applicable Modules

Module Series	Model	Description
BG95	BG95-M1	Cat M1 only
	BG95-M2	Cat M1/Cat NB2
	BG95-M3	Cat M1/Cat NB2/EGPRS
	BG95-M4	Cat M1/Cat NB2, 450 MHz Supported
	BG95-M5	Cat M1/Cat NB2/EGPRS, Power Class 3
	BG95-M6	Cat M1/Cat NB2, Power Class 3
	BG95-MF	Cat M1/Cat NB2, Wi-Fi Positioning
BG77	BG77	Cat M1/Cat NB2
BG600L	BG600L-M3	Cat M1/Cat NB2/EGPRS

NOTE

See the firmware release notes of corresponding module models to check whether the function has been supported.

1.2. The Process of Using FILE AT Commands

The following illustrates the general procedure to upload/download, open/create, read and write a file in the storage:

1) Upload a file to the storage with **AT+QFUPL**. If necessary, output/download the file with **AT+QFDWL** to check the file content, data integrity, etc.
2) Open the file with **AT+QFOPEN**, and then you can write or read the file at any time and any location until the file is closed by **AT+QFCLOSE**.
   - When opening a file with **AT+QFOPEN**, you can set the file into overwrite mode or read-only mode or others with **<mode>** (see ***Chapter 2.6*** for details). After the file is opened, a **<filehandle>** is assigned to it. Then the file can be operated via this **<filehandle>**.
   - After the file is opened, you can write it with **AT+QFWRITE** or read it with **AT+QFREAD** from the current file position.
   - You can set the file position with **AT+QFSEEK** and query the current position with **AT+QFPOSITION**.
   - **AT+QFTUCAT** will truncate the file from the current position to the end of the file.
3) Close the file with **AT+QFCLOSE**, after which the **<filehandle>** becomes meaningless any more.

The following are several commonly used commands to manage files in the storage:

1) **AT+QFLDS**: Get the space information of storage
2) **AT+QFLST**: List the file information in the specified storage.
3) **AT+QFDEL**: Delete the file(s) in the specified storage.

1.3. Description of Data Mode

The COM port of the modules has two working modes: AT command mode and data mode. In AT command mode, the inputted data via COM port is taken as AT command. While in data mode, it is taken as data.

Inputting **+++** or pulling up MAIN_DTR pin (**AT&D1** should be set first) can make the COM port exit from data mode. To prevent the **+++** from being misinterpreted as data, the following sequence should be

followed:

1) Do not input any character for at least 1 second before you input **+++**.
2) Input **+++** within 1 second, and no other characters can be inputted during the time.
3) Do not input any character for at least 1 second after **+++** has been inputted.

When **AT+QFUPL**, **AT+QFDWL**, **AT+QFREAD** and **AT+QFWRITE** are executed, the COM port enters data mode. If you use **+++** or MAIN_DTR to make the port exit from data mode, the executing procedure of these commands will be interrupted before the response is returned. In such a case, the COM port cannot reenter data mode by executing **ATO**.

2 Description of FILE AT Commands

2.1. AT Command Syntax

2.1.1. Definitions

- **<CR>** Carriage return character.
- **<LF>** Line feed character.
- **<...>** Parameter name. Angle brackets do not appear on the command line.
- **[...]** Optional parameter of a command or an optional part of TA information response. Square brackets do not appear on the command line. When an optional parameter is omitted, the new value equals to the previous value or the default settings, unless otherwise specified.
- **Underline** Default setting of a parameter.

2.1.2. AT Command Syntax

All command lines must start with "**AT**" or "**at**" and end with **"<CR>"**. Information responses and result codes always start and end with a carriage return character and a line feed character: **<CR><LF><response><CR><LF>**. Throughout this document, only the commands and responses are presented, while carriage return and line feed characters are deliberately omitted.

Table 2: Types of AT Commands and Responses

Command Type	Syntax	Description
Test Command	**AT+<cmd>=?**	Returns the list of parameters and value ranges set by the corresponding Write Command or internal processes.
Read Command	**AT+<cmd>?**	Returns the currently set value of a parameter or parameters.
Write Command	**AT+<cmd>=<p1>[,<p2>[,<p3>[...]]]**	Sets parameter values.
Execution Command	**AT+<cmd>**	Reads non-variable parameters affected by internal processes in the module.

2.2. Description of FILE AT Commands

2.2.1. AT+QFLDS Get the Space Information of Storage

This command gets the space information of the specified storage.

AT+QFLDS Get the Space Information of Storage	
Test Command **AT+QFLDS=?**	Response **OK**
Write Command **AT+QFLDS=<name_pattern>**	Response **+QFLDS: <freesize>,<total_size>** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**
Execution Command **AT+QFLDS**	Response **+QFLDS: <UFS_file_size>,<UFS_file_number>** **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	/

Parameter

<name_pattern>	String type. Storage pattern. "UFS" UFS on Modem side "EUFS" Extended UFS on AP side
<freesize>	Integer type. The free space size of **<name_pattern>**. Unit: byte.
<total_size>	Integer type. The total size of **<name_pattern>**. Unit: byte.
<UFS_file_size>	Integer type. The size of all files in UFS. Unit: byte
<UFS_file_number>	Integer type. The number of files in UFS.
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.

Example

```
AT+QFLDS="UFS"                    //Query the space information of UFS on Modem side.
+QFLDS: 1249984,1562624

OK
```

```
AT+QFLST="*"
+QFLST: "Test1.txt",10
+QFLST: "Test2.txt",16
+QFLST: "Test3.txt",150
+QFLST: "Test4.txt",2056
+QFLST: "Test5.txt",98

OK
AT+QFLDS                    //Query the size and number of all files in UFS.
+QFLDS: 2330,5

OK
```

2.2.2. AT+QFLST List the File Information

This command lists the information of a single file or all files in the specified storage.

AT+QFLST List the File Information	
Test Command **AT+QFLST=?**	Response **OK**
Write Command **AT+QFLST=<name_pattern>**	Response **+QFLST: <filename>,<file_size>** **[+QFLST: <filename>,<file_size>** **[…]]** **OK** If there is any error: **+CME ERROR: <err>**
Execution Command **AT+QFLST**	Response Return the information of UFS files: **+QFLST: <filename>,<file_size>** **[+QFLST: <filename>,<file_size>** **[…]]** **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	/

Parameter

<name_pattern>	String type. The file to be listed.	
	"*"	All the files in UFS
	"UFS:*"	All the files in UFS
	"EUFS:/ufs/*"	All the files in *ufs* directory of EUFS
	"EUFS:/datatx/*"	All the files in *datatx* directory of EUFS
	"EUFS:*"	All the files in *ufs* directory of EUFS
	"**<filename>**"	A specified file **<filename>** in UFS
	"UFS:**<filename>**"	A specified file **<filename>** in UFS
	"EUFS:**<filename>**"	A specified file **<filename>** in *ufs* directory of EUFS
	"EUFS:/ufs/**<filename>**"	A specified file **<filename>** in *ufs* directory of EUFS
	"EUFS:/datatx/**<filename>**"	A specified file **<filename>** in *datatx* directory of EUFS
<filename>	String type. Filename. The maximum length of **<filename>** is 80 bytes.	
<file_size>	Integer type. Size of the file. Unit: byte.	
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.	

Example

```
AT+QFLST="*"                          //List all the files in UFS.
+QFLST: "F_M12-1.bmp",562554
+QFLST: "F_M12-10.bmp",562554
+QFLST: "F_M12-11.bmp",562554

OK
AT+QFLST="Test1.txt"                  //List a specified file "Test1.txt" in UFS.
+QFLST: "Test1.txt",2

OK
AT+QFLST="EUFS:/ufs/*"                //List all the files in ufs directory of EUFS .
+QFLST: "EUFS:test.txt",10
+QFLST: "EUFS:test2.txt",20

OK
AT+QFLST="EUFS:test.txt"              //List a specified file "test.txt" in ufs directory of EUFS.
+QFLST: "EUFS:test.txt",10

OK
```

2.2.3. AT+QFDEL Delete File(s)

This command deletes a specified file or all the files in the storage.

AT+QFDEL Delete File(s)	
Test Command **AT+QFDEL=?**	Response **+QFDEL: <filename>** **OK**
Write Command **AT+QFDEL=<filename>**	Response **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	/

Parameter

<filename>	String type. Name of the file to be deleted. The maximum length of **<filename>** is 80 bytes.	
	"*"	All the files in UFS
	"UFS:*"	All the files in UFS
	"EUFS:/ufs/*"	Delete all the files in *ufs* directory of EUFS (*ufs* directory is remained)
	"EUFS:/datatx/*"	Delete all the files in *datatx* directory of EUFS (*datatx* directory is remained)
	"EUFS:*"	Delete all the files in *ufs* directory of EUFS (*ufs* directory is remained)
	"**<filename>**"	A specified file **<filename>** in UFS
	"UFS:**<filename>**"	A specified file **<filename>** in UFS
	"EUFS:**<filename>**"	Delete a specified file **<filename>** in *ufs* directory of EUFS
	"EUFS:/ufs/**<filename>**"	Delete a specified file **<filename>** in *ufs* directory of EUFS
	"EUFS:/datatx/**<filename>**"	Delete a specified file **<filename>** in *datatx* of EUFS
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.	

Example

```
AT+QFDEL="*"
OK

AT+QFDEL="Test1.txt"
OK
```

2.2.4. AT+QFUPL Upload a File

This command uploads a file to the storage. If any file in the storage has the same name with the file uploaded, an error is reported.

After the Write Command is executed and **CONNECT** is returned, the module switches to data mode. When the uploaded data reaches **<file_size>**, or there is no any data inputted when **<timeout>** reaches, the module exits from data mode automatically. During data transmission, you can use **+++** or MAIN_DTR to make the module exit from data mode, and more details are provided in ***Chapter 1.2***.

AT+QFUPL Upload a File	
Test Command **AT+QFUPL=?**	Response **+QFUPL: <filename>[,(1-<freesize>)[,(**range of supported **<timeout>**s)**[,(**list of supported **<ackmode>**s)**]]]** **OK**
Write Command **AT+QFUPL=<filename>[,<file_size>[,<timeout>[,<ackmode>]]]**	Response **CONNECT** TA switches to the data mode (transparent access mode), and the binary data of file can be inputted. When the total size of the inputted data reaches **<file_size>** (unit: byte), TA will return to command mode and reply the following codes: **+QFUPL: <upload_size>,<checksum>** **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	/

Parameter

<freesize>	Integer type. The free space size of **<name_pattern>**. See **AT+QFLDS** for more details of **<name_pattern>**.
<filename>	String type. Name of the file to be uploaded. The maximum length is 80 bytes. "**<filename>**" Upload the file to UFS "UFS:**<filename>**" Upload the file to UFS "EUFS:**<filename>**" Upload the file to *ufs* directory of EUFS "EUFS:/ufs/**<filename>**" Upload the file to *ufs* directory of EUFS "EUFS:/datatx/**<filename>**" Upload the file to *datatx* directory of EUFS
<file_size>	Integer type. The file size expected to be uploaded. Default value: 10240. Unit: byte. The maximum length is not greater than **<freesize>**.
<upload_size>	Integer type. The actual size of the uploaded data. Unit: byte.
<timeout>	Integer type. The time waiting for data to be inputted to USB/UART. Range: 1–65535. Default value: 5. Unit: s.
<ackmode>	Integer type. Whether to use ACK mode. 0 Turn off the ACK mode by default. 1 Turn on the ACK mode.

<checksum>	Integer type. The checksum of the uploaded data.
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.

NOTES

1. It is strongly recommended to use DOS 8.3 file name format for **<filename>**.
2. **<checksum>** is a 16-bit checksum based on bitwise XOR.
   If the number of the characters is odd, set the last character as the high 8 bit, and the low 8 bit as 0, and then use an XOR operator to calculate the checksum. **+++** sequence will make TA end the command and switch to command mode. However, the data previously uploaded will be preserved in the file.
3. When executing the command, the data must be entered after **CONNECT** is returned.
4. The ACK mode is provided to avoid the loss of data when uploading a large file, in case hardware flow control does not work. The ACK mode works as follows:
   1) Run **AT+QFUPL=<filename>,<file_size>,<timeout>,1** to enable the ACK mode.
   2) The module outputs **CONNECT**.
   3) MCU sends 1K bytes data, and then BG95 series, BG77 and BG600L-M3 modules will respond with an **A**.
   4) MCU receives this **A** and then sends the next 1K bytes data;
   5) Repeat step 3) and 4) until the transfer is completed.

2.2.5. AT+QFDWL Download a File

This command downloads a specified file from the storage.

AT+QFDWL Download a File	
Test Command **AT+QFDWL=?**	Response **+QFDWL: <filename>** **OK**
Write Command **AT+QFDWL=<filename>**	Response **CONNECT** TA switches to data mode, and the binary data of the file will be outputted. When the file is read over, TA will return to command mode and reply the following codes: **+QFDWL: <download_size>,<checksum>** **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	/

Parameter

<filename>	String type. Name of the file to be downloaded. The maximum length is 80 bytes. "**<filename>**" Download the file from UFS "UFS:**<filename>**" Download the file from UFS "EUFS:**<filename>**" Download the file from *ufs* directory of EUFS "EUFS:/ufs/**<filename>**" Download the file from *ufs* directory of EUFS "EUFS:/datatx/**<filename>**" Download the file from *datatx* directory of EUFS
<download_size>	Integer type. The size of the downloaded data. Unit: byte.
<checksum>	Integer type. The checksum of the downloaded data.
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.

NOTES

1. **+++** sequence causes TA to end the command and switches to command mode.
2. **<checksum>** is a 16-bit checksum based on bitwise XOR.

2.2.6. AT+QFOPEN Open a File

This command opens a file and gets the file handle to be used in commands such as **AT+QFREAD**, **AT+QFWRITE**, **AT+QFSEEK**, **AT+QFPOSITION**, **AT+QFTUCAT** and **AT+QFCLOSE**.

AT+QFOPEN Open a File	
Test Command **AT+QFOPEN=?**	Response **+QFOPEN: <filename>[,(**range of supported <**mode>**s**)]** **OK**
Read Command **AT+QFOPEN?**	Response **+QFOPEN: <filename>,<filehandle>,<mode>** **[+QFOPEN: <filename>,<filehandle>,<mode>** **[…]]** **OK**
Write Command **AT+QFOPEN=<filename>[,<mode>]**	Response **+QFOPEN: <filehandle>** **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	/

Parameter

<filename>	String type. Name of the file to be opened. The maximum length is 80 bytes. "**<filename>**" Open the file in UFS "UFS:**<filename>**" Open the file in UFS "EUFS:**<filename>**" Open the file of *ufs* directory of EUFS. "EUFS:/ufs/**<filename>**" Open the file of *ufs* directory of EUFS "EUFS:/datatx/**<filename>**" Open the file of *datatx* directory of EUFS
<filehandle>	Integer type. The handle of the file to be operated.
<mode>	Integer type. The open mode of the file. 0 If the file does not exist, it is created. If the file exists, it is opened directly. In any case, the file can be read and written. 1 If the file does not exist, it is created. If the file exists, it is overwritten. In any case, the file can be read and written. 2 If the file exists, it is opened directly and is read only. If the file does not exist, an error is returned. 3 If the file does not exist, it is created. If the file exists, write data to the file. In any case, the file can be read and written.
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.

NOTE

<filehandle> starts form 0 in UFS and starts from 20000 in EUFS.

2.2.7. AT+QFREAD Read a File

This command reads the data of a file which is specified by the file handle. The data starts from the current position of the file pointer which belongs to the file handle.

AT+QFREAD Read a File	
Test Command **AT+QFREAD=?**	Response **+QFREAD: <filehandle>[,<length>]** **OK**
Write Command **AT+QFREAD=<filehandle>[,<length>]**	Response **CONNECT <read_length>** TA switches to data mode. When the total size of the data reaches **<length>** (unit: byte), TA will return to command mode, display the result and then reply the following codes: **OK** If there is any error:

	+CME ERROR: <err>
Characteristics	/

Parameter

<filehandle>	Integer type. The handle of the file to be operated.
<length>	Integer type. The length of the file to be read out and the default is the file length. Unit: byte.
<read_length>	Integer type. The actual read length. Unit: byte.
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.

2.2.8. AT+QFWRITE Write a File

This command writes data into a file. The data starts from the current position of the file pointer which belongs to the file handle.

AT+QFWRITE Write a File	
Test Command **AT+QFWRITE=?**	Response **+QFWRITE: <filehandle>[,<length>[,<timeout>]]** **OK**
Write Command **AT+QFWRITE=<filehandle>[,<length>[,<timeout>]]**	Response **CONNECT** TA switches to data mode. When the total size of the written data reaches **<length>** (unit: byte) or the time reaches **<timeout>**, TA will return to command mode and reply the following codes: **+QFWRITE: <written_length>,<total_length>** **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	/

Parameter

<filehandle>	Integer type. The handle of the file to be operated.
<length>	Integer type. The length of the file to be written. The maximum value of this parameter is determined by **<freesize>** of **AT+QFUPL**.

	Default value: 10240. Unit: byte.
<timeout>	Integer type. The time waiting for data to be inputted to USB/UART. Range: 1–65535. Default value: 5. Unit: s.
<written_length>	Integer type. The actual written length. Unit: byte.
<total_length>	Integer type. The total length of the file. Unit: byte.
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.

2.2.9. AT+QFSEEK Set File Pointer to a Position

This command sets a file pointer to a specified position. This will decide the starting position of commands such as **AT+QFREAD**, **AT+QFWRITE**, **AT+QFPOSITION** and **AT+QFTUCAT**.

AT+QFSEEK Set File Pointer to a Position	
Test Command **AT+QFSEEK=?**	Response **+QFSEEK: <filehandle>,<offset>[,<position>]** **OK**
Write Command **AT+QFSEEK=<filehandle>,<offset>[,<position>]**	Response **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	/

Parameter

<filehandle>	Integer type. The handle of the file to be operated.
<offset>	Integer type. The number of bytes of the file pointer movement.
<position>	Integer type. Pointer movement mode. 0 The beginning of the file. 1 The current position of the pointer. 2 The end of the file.
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.

NOTES

1. If **<position>**=0 and **<offset>** exceeds the file size, the command returns **ERROR**.
2. If **<position>**=1 and the total size of **<offset>** and the current position of the pointer exceed the file size, the command returns **ERROR**.
3. If **<position>**=2, the handle moves forth.

2.2.10. AT+QFPOSITION Get the Offset of a File Pointer

This command gets the offset of a file pointer from the beginning of the file.

AT+QFPOSITION Get the Offset of a File Pointer	
Test Command **AT+QFPOSITION=?**	Response **+QFPOSITION: <filehandle>** **OK**
Write Command **AT+QFPOSITION=<filehandle>**	Response **+QFPOSITION: <offset>** **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	/

Parameter

<filehandle>	Integer type. The handle of the file to be operated.
<offset>	Integer type. The offset from the beginning of the file.
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.

2.2.11. AT+QFTUCAT Truncate a File from the File Pointer

This command truncates all the data behind the position that the file pointer indicates.

AT+QFTUCAT Truncate a File from the File Pointer	
Test Command **AT+QFTUCAT=?**	Response **+QFTUCAT: <filehandle>** **OK**
Write Command **AT+QFTUCAT=<filehandle>**	Response **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	/

Parameter

<filehandle>	Integer type. The handle of the file to be operated.
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.

2.2.12. AT+QFCLOSE Close a File

This command closes a file and ends all the operation to the file. After that, the file handle is released and should not be used again, unless open the file again with **AT+QFOPEN**.

AT+QFCLOSE Close a File	
Test Command **AT+QFCLOSE=?**	Response **+QFCLOSE: <filehandle>** **OK**
Write Command **AT+QFCLOSE=<filehandle>**	Response **OK** If there is any error **+CME ERROR: <err>**
Characteristics	/

Parameter

<filehandle>	Integer type. The handle of the file to be operated.
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.

2.2.13. AT+QFCRC Calculate the CRC of a Specified UFS File

This command calculates the CRC16, CRC16_CCITT and CRC32 checksum of a specified UFS file.

AT+QFCRC Calculate the CRC of a Specified UFS File	
Test Command **AT+QFCRC=?**	Response **+QFCRC: <filename>** **OK**
Write Command **AT+QFCRC=<filename>**	Response When the specified file exists, the command returns 32 bits CRC, 16 bits CRC CCITT and 16 bits CRC values: **+QFCRC: <crc32>,<crc16>,<crc16_ccitt>** **OK**

	When the file exists but has no byte: **+QFCRC: 0,0,0** If there is any error: **+CME ERROR: <err>**
Characteristics	/

Parameter

<filename>	String type. The input format is "UFS:filename" where "UFS:" can be omitted.
<crc32>	The 32 bits CRC value.
<crc16>	The 16 bits CRC value.
<crc16_ccitt>	The 16 bits CRC CCITT value.
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.

NOTE

This command is applicable to UFS files only.

2.2.14. AT+QFCPY Make a Copy of a Specified File

This command copies the whole content of a source file to a destination file.

- When the destination file has already existed, the command will return an **ERROR** unless **<enable>**=1.
- When the destination file has the same name as the source file, the command will return an **ERROR**.
- While the remaining space of the file system is smaller than the file size to be copied, the command also will return an **ERROR**.

AT+QFCPY Make a Copy of a Specified File	
Test Command **AT+QFCPY=?**	Response **+QFCPY: <src_file>,<dest_file>[,<enable>]** **OK**
Write Command **AT+QFCPY=<src_file>,<dest_file>[,<enable>]**	Response **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	/

Parameter

<src_file>	String type. The source file name. The input format is "UFS:filename" where "UFS:" can be omitted.
<dest_file>	String type. The destination file name. The file name input format is “UFS:filename” where “UFS:” can be omitted.
<enable>	Integer type. Determine whether to overwrite the existing file which has the same name as the destination file. 0 Disable 1 Enable
<err>	Integer type. Error code. See ***Chapter 4*** for possible **<err>** values.

NOTE

This command is applicable to UFS files only.

3 Examples

3.1. Upload and Download Files

3.1.1. Upload a File

3.1.1.1. Non ACK Mode

```
AT+QFUPL="test1.txt",10                  //Upload the text file "test1.txt" to UFS.
CONNECT
<Input file bin data>
+QFUPL: 10,3938

OK
AT+QFUPL="EUFS:test.txt",4               //Upload the text file "test.txt" to ufs directory of EUFS.
CONNECT
<Input file bin data>
+QFUPL: 4,6a05

OK
```

3.1.1.2. ACK Mode

The ACK mode can make the data transmission more reliable. When transmitting a large file without hardware flow control, the ACK mode is recommended to be used to prevent the data from being lost. For more details about ACK mode, please refer to **AT+QFUPL** command.

```
AT+QFUPL="test.txt",3000,5,1             //Upload the text file "test.txt" to UFS.
CONNECT
<input file bin data of 1024bytes>
A                                        //After receiving 1024 bytes data, the module returns A. And
                                         then the next 1024 bytes data can be inputted.
<input file bin data of 1024bytes>
A
<input the rest file bin data>
+QFUPL: 3000,B34A
```

```
OK
```

3.1.2. Download a File

```
AT+QFDWL="test.txt"                 //Download the text file "test.txt" from UFS.
CONNECT
<Output Data>
+QFDWL: 10,613e                     //Get the bytes of the downloaded data and the checksum.

OK
```

3.2. Write and Read Files

3.2.1. Write and Read a UFS File

```
AT+QFOPEN="test",0                  //Open the file to get the file handle.
+QFOPEN: 0

OK
AT+QFWRITE=0,10                     //Write 10 bytes to the file.
CONNECT
<Write Data>
+QFWRITE: 10,10                     //The actual bytes written and the size of the file are returned.

OK
AT+QFSEEK=0,0,0                     //Set the file pointer to the beginning of the file.
OK
AT+QFREAD=0,10                      //Read 10 bytes from the file.
CONNECT 10
<Read Data>

OK
AT+QFCLOSE=0                        //Close the file.
OK
```

3.2.2. Write and Read an EUFS File

```
AT+QFLDS="EUFS"                     //Query the space information of EUFS.
+QFLDS: 1388544,2435072

OK
```

```
AT+QFOPEN="EUFS:test",0          //Open the file to get the file handle.
+QFOPEN: 20000

OK
AT+QFWRITE=20000,10              //Write 10 bytes to the file.
CONNECT
<Write Data>
+QFWRITE: 10,10                  //The actual bytes written and the size of the file are returned.

OK
AT+QFSEEK=20000,0,0              //Set the file pointer to the beginning of the file.
OK
AT+QFREAD=20000,10               //Read 10 bytes from the file.
CONNECT 10
<Read Data>

OK
AT+QFCLOSE=20000                 //Close the file.
OK
```

4 Summary of Error Codes

The error code **<err>** indicates an error related to mobile equipment or network. The details about **<err>** are described in the following table, and these error codes are only related to file operation of the modules.

Table 3: Summary of Error Codes

<err>	Meaning
400	Invalid input value
401	Larger than the size of the file
402	Read zero byte
403	Drive full
405	File not found
406	Invalid file name
407	File already existed
409	Fail to write the file
410	Fail to open the file
411	Fail to read the file
413	Reach the max number of file allowed to be opened
414	The file read-only
416	Invalid file descriptor
417	Fail to list the file
418	Fail to delete the file
419	Fail to get disk info

420	No space
421	Time out
423	File too large
425	Invalid parameter
426	File already opened

5 Appendix A Reference

Table 4: Terms and Abbreviations

Abbreviation	Description
ACK	Acknowledgement
COM	Communication Port
CRC	Cyclic Redundancy Check
DOS	Disk Operating System
EUFS	Extended User File Storage
ME	Mobile Equipment
TA	Terminal Adapter
UART	Universal Asynchronous Receiver/Transmitter
UFS	User File Storage
USB	Universal Serial Bus
XOR	Exclusive OR