Multi-interface Controller

UC50x Series (LTE Version)

Communication Protocol

Revision History

Date	Doc Version	Description
March 15, 2024	V 1.0	Initial Version

Contents

1. Overview

UC50x supports transmission with AWS/MQTT/TCP/UDP server. This document is only for introduction and communication structure explanation.

Note: All explanations and examples in this document are based on HEX format.

2. AWS/MQTT Topics

When the device is connected to AWS/MQTT server, the bi-directional communication uses different default topics. MQTT topics support to change as required.

Topic	Content
uc/[SN]/uplink	Receive periodic reports, alarms, etc.
uc/[SN]/downlink	Send downlink commands
uc/[SN]/uplink/passthrough/serial	Receive the transparent replies from RS232/RS485 terminal devices
uc/[SN]/downlink/passthrough/serial	Send any type of messages to RS232/RS485 terminal devices
uc/[SN]/uplink/passthrough/SDI-12	Receive the transparent replies from SDI-12 terminal devices
uc/[SN]/downlink/passthrough/SDI-12	Send any type of messages to SDI-12 terminal devices

3. Uplink Payload

All uplink data are based on the following format (HEX):

Start	ID	Packet Length	FLAG	Frame Counter	Protocol Version	Software Version	Hardware Version
02	0001	2 Bytes	00	0000	01	4 Bytes	4 Bytes
SN	**IMEI**	**IMSI**	**ICCID**	**Signal**	**Data Length**	**Data1**	...
16 Bytes	15 Bytes	15 Bytes	20 Bytes	1 Byte	2 Bytes	N Bytes	...

Example:

02 0001 0060 00 0008 00 30313031 30313030
3637373244343132333531383030313 3836373130373036383733353430313436303038383333373630343237393838363038313331303233383039363032373911 000f
07ef 7bbe ee65 0300 0004 0000 0175 62

Type	Content
Start	02
ID	0001
Packet Length	00 60=96 bytes
FLAG	00
Frame Counter	0008
Protocol Version	00
Software Version	30 31 30 31 => 0101=V1.1
Hardware Version	30 31 30 30 => 0100=V1.0
SN	36 37 37 32 44 34 31 32 33 35 31 38 30 30 31 33 =>6772D41235180013
IMEI	38 36 37 31 30 37 30 36 38 37 33 35 34 30 31=>867107068735401
IMSI	34 36 30 30 38 38 33 33 37 36 30 34 32 37 39 =>460088337604279
ICCID	38 39 38 36 30 38 31 33 31 30 32 33 38 30 39 36 30 32 37 39 => 89860813102380960279
Network Signal	11=>17 asu
Data Length	00 0f=>15 Bytes
Data	See details below

Data part is based on Channel+Type+Data, the Data field should follow little-endian:

Channel1	Type1	Data1	Channel2	Type2	Data2	Channel 3	...
1 Byte	1 Byte	N Bytes	1 Byte	1 Byte	M Bytes	1 Byte	...

Note:

1) The frame counter will clear when the device reboots.

2) For all Milesight IoT decoder examples please find files on https://github.com/Milesight-IoT/SensorDecoders

3.1 Periodic Report

UC50x series reports data collected from sensors according to reporting interval (360min by default). UC50x series only report data which data interfaces are enabled.

Note: RS232/RS485/SDI-12 Transparent doesn't have its own channel or type.

Normal Report:

Item	Channel	Type	Description
Unix Timestamps	07	ef	4 Bytes, unit: s
Battery Level	01	75	UINT8, Unit: %
GPIO1-Digital Input	03	00	00=low, 01=high
GPIO1-Digital Output		01	00=low, 01=high
GPIO1-Counter		c8	UINT32
GPIO2-Digital Input	04	00	00=low, 01=high
GPIO2-Digital Output		01	00=low, 01=high

<table>
<tr><td>GPIO2-Counter</td><td></td><td>c8</td><td>UINT32</td></tr>
<tr><td>Analog Input1</td><td>05</td><td rowspan="2">f1</td><td rowspan="2">1. When no AI alarm rule:
AI Type (1B) + Current Value (2B)
AI Type:
04: 4-20 mA, 05: 0-10V

2. When adding the AI alarm rule:
AI Type (1B) + Current Value (2B)+ Min. Value (2B)+ Max. Value (2B)+ Average Value (2B)
AI Type:
06: 4-20 mA, 07: 0-10V

Current/Min./Max./Avg. Value: Float16</td></tr>
<tr><td>Analog Input2</td><td>06</td></tr>
<tr><td>SDI-12</td><td>08</td><td>f2</td><td>Byte 1: 00~0f (Channel 1 to 16)
Byte 2: Data length
Byte 2-37: SDI-12 data (ASCII characters)
Note: every channel only reports the first 36 characters and will not report the rest if the data length is more than 36.</td></tr>
<tr><td>RS485 Modbus Channel</td><td>09</td><td>f3</td><td>Channel ID(1B)+Data Type (1B)+Data (1~4 B)
Byte 1: 00~0f (Channel 1 to 16)
Byte 2: Data Type:
<table>
<tr><th>Code</th><th>Data Type</th></tr>
<tr><td>00</td><td>Coil</td></tr>
<tr><td>01</td><td>Discrete</td></tr>
<tr><td>02</td><td>Input16
Input_int32_with upper 16 bits
Input_int32_with lower 16 bits
(Unsigned)</td></tr>
<tr><td>12</td><td>Input16
Input_int32_with upper 16 bits
Input_int32_with lower 16 bits
(Signed)</td></tr>
<tr><td>03</td><td>Hold16
Hold_int32_with upper 16 bits
Hold_int32_with lower 16 bits
(Unsigned)</td></tr>
</table></td></tr>
</table>

				13	Hold16 Hold_int32_with upper 16 bits Hold_int32_with lower 16 bits (Signed)
				x4	04: Hold32-Unsigned 14: Hold32-Signed
				05	Hold_float
				x6	06: Input32-Unsigned 16: Input32-Signed
				07	Input_float

Abnormal Report:

Item	Channel	Type	Description
Analog Input1	b5	f1	00: read error 01: out of measuring range
Analog Input2	b6		
SDI-12 Collection Failure	b8	f2	Channel ID(1B) + 00
RS485 Modbus Channel Collection Failure	b9	f3	Channel ID(1B) + 00

Example:

07efe105f365 030001 04c801000000 05f1058f42 09f300030f00 b9f30500 08f2000e362b302e302b302b32332e370d0ab 017561		
Channel	**Type**	**Value**
07	ef	e1 05 f3 65 => 65 f3 05 e1 =1710425569s=2024-03-14 22:12
03	00	GPIO1-Digital Input 1: 01=High
04	c8	GPIO2-Counter: 01 00 00 00=>00 00 00 01=1
05	f1	AI Type: 05=0-10 V Current Value: 8f 42=>42 8f=3.28V
09	f3	00=Modbus Channel 1 03=Hold16 (Unsigned) Value: 0f 00=>00 0f=15
b9	f3	05=Modbus Channel 6 Collection Failure
08	f2	00=Channel 1 Data length: 0e=>15 Data: Hex to ASCII result is 6+0.0+0+23.7
01	75	Battery Level: 61 => 97%

Note: When data type is holding register or input register, ToolBox can set different byte order. Take the following Modbus register response from RS485 sensors as example:

Register Address	Value (Hex)
0	00 15

1	00 20

When using different byte orders,you can use ToolBox to fetch different results and the device will upload data with little endian order.

Data Type	Byte Order	Fetch Result	Uplink (HEX)
Holding/Input Register (INT16)	AB	21 (0x15)	15 00 (BA)
	BA	5376 (0x1500)	00 15 (AB)
Holding/Input Register (INT32)	ABCD	1376288 (0x00150020)	20 00 15 00 (DCBA)
	BADC	352329728 (0x15002000)	00 20 00 15 (CDAB)
	CDAB	2097173 (0x00150015)	15 00 20 00 (BADC)
	DCBA	536876288 (0x20001500)	00 15 00 20 (ABCD)
Holding/Input Register (INT32 with upper 16 bits)	/	21 (0x15)	15 00 00 00
Holding/Input Register (INT32 with lower 16 bits)	/	21 (0x15)	15 00 00 00

3.2 Alarm

DI Alarm:

When GPIO works as DI mode, the device will reports the DI status packet when DI status changes.

Example: when GPIO1-DI changes status from low to high.

07efe049f065 030001		
Channel	**Type**	**Value**
07	ef	e0 49 f0 65 => 65 f0 49 e0 =1710246368s=2024-03-12 20:26:08
03	00	GPIO1-Digital Input: 01=High

Analog/RS485 Alarm:

UC50x series supports to set and send alarms when the analog value or RS485 Modbus channel value reaches the preset conditions. Usually there are two alarm types:

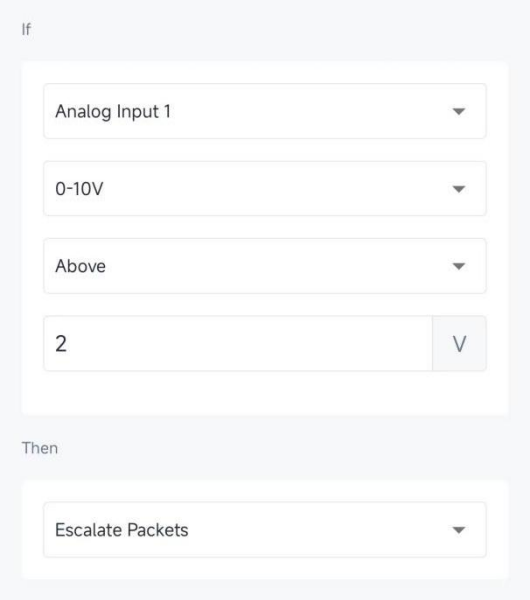
- Threshold alarm: when the value is above or below or within the range of the threshold;
- Change alarm: when the current collected value-last collected value > change value.

Item	Channel	Type	Description
Analog Input1-Threshold Alarm	85	f1	AI Type (1B) + Current Value (2B) + 01 AI Type: 04: 4-20 mA, 05: 0-10V Current Value: Float16
Analog Input2-Threshold Alarm	86		
Analog Input1-Change	95		AI Type (1B) + Current Value (2B) +

Alarm			Change Value (2B) + 01 AI Type: 04: 4-20 mA, 05: 0-10V Current Value: Float16
Analog Input2-Change Alarm	96		
Modbus Channel-Threshold Alarm	89	f3	Channel ID(1B)+Data Type (1B)+Data (1~4 B)+01
Modbus Channel-Change Alarm	99		Channel ID(1B)+Data Type (1B)+Data (1~4 B)+Change Value (1~4B) +00

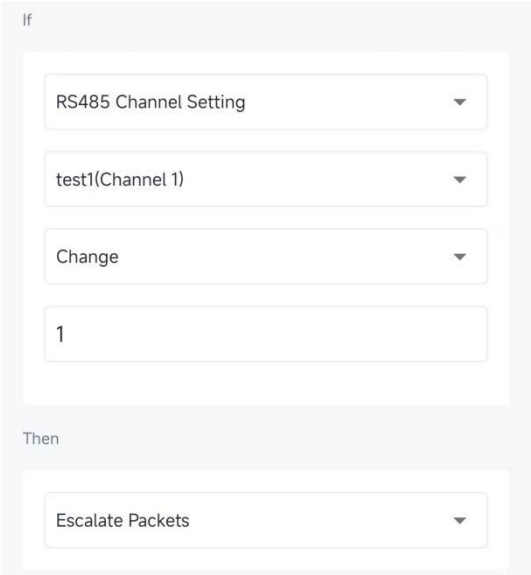
Examples:

1. Analog Input threshold alarm



07ef f301f365 85f1058f4201					
Channel	Type	Value	Channel	Type	Value
07	ef	f3 01 f3 65 => 65 f3 01 f3 =1710424563s=2024-03-14 21:56:03	85	f1	AI Type: 05=0-10 V Current Value: 8f 42 >42 8f=3.28V

2. RS485 change alarm



07ef7705f365 99f300030f00060000					
Channel	Type	Value	Channel	Type	Value
07	ef	77 05 f3 65 => 65 f3 05 77= 1710425463s=2024-03-14 22:11:03	99	f3	00=Modbus Channel 1 03=Hold16 (Unsigned) Value: 0f 00=>00 0f=15 Change value: 06 00=>00 06=6

4. Downlink Command

Downlink command is used for controlling the UC300 via server remotely. If you use MQTT/AWS server, please subscribe corresponding downlink topics to send commands.

Note:

1) The device can only receive downlink commands within the 8s after sending uplink packets or opening reception windows.

2) Downlink control is not supported when using UDP protocol.

3) The device will return “fe”+type+command if it executes the command in success.

Item	Channel	Type	Description
GPIO1-DO	03	/	00ffff: Low level 01ffff: High level
GPIO2-DO	04	/	
Collecting Interval	ff	02	UINT16, unit: s
Reporting Interval		03	UINT16, unit: s
Reboot		10	ff
UTC Time Zone		17	INT16/10
Enquire Current Data		28	ff
Data Storage		68	00: disable, 01: enable
Data Retransmission		69	00: disable, 01: enable

<table>
<tr><td>Data Retransmission Interval</td><td></td><td>6a</td><td>3 Bytes
Byte 1: 00
Byte 2-3: interval time, unit:s
range: 30~1200s (600s by default)</td></tr>
<tr><td>Modbus Channel Setting</td><td></td><td>ef</td><td>01+Channel ID (1B)+Slave ID (1B) + Address (2B) + Type (1B) + Sign (1B)
Sign: 11=signed, 01=unsigned
Type:
<table>
<tr><th>Code</th><th>Data Type</th></tr>
<tr><td>00</td><td>Coil</td></tr>
<tr><td>01</td><td>Discrete</td></tr>
<tr><td>08</td><td>Input32_AB</td></tr>
<tr><td>09</td><td>Input32_CD</td></tr>
<tr><td>0a</td><td>Hold32_AB</td></tr>
<tr><td>0b</td><td>Hold32_CD</td></tr>
<tr><td>0c</td><td>Input16_AB</td></tr>
<tr><td>0d</td><td>Input16_BA</td></tr>
<tr><td>0e</td><td>Input32_ABCD</td></tr>
<tr><td>0f</td><td>Input32_BADC</td></tr>
<tr><td>10</td><td>Input32_CDAB</td></tr>
<tr><td>11</td><td>Input32_DCBA</td></tr>
<tr><td>12</td><td>Input_float_ABCD</td></tr>
<tr><td>13</td><td>Input_float_BADC</td></tr>
<tr><td>14</td><td>Input_float_CDAB</td></tr>
<tr><td>15</td><td>Input_float_DCBA</td></tr>
<tr><td>16</td><td>Hold16_AB</td></tr>
<tr><td>17</td><td>Hold16_BA</td></tr>
<tr><td>18</td><td>Hold32_ABCD</td></tr>
<tr><td>19</td><td>Hold32_BADC</td></tr>
<tr><td>1a</td><td>Hold32_CDAB</td></tr>
<tr><td>1b</td><td>Hold32_DCBA</td></tr>
<tr><td>1c</td><td>Hold_float_ABCD</td></tr>
<tr><td>1d</td><td>Hold_float_BADC</td></tr>
<tr><td>1e</td><td>Hold_float_CDAB</td></tr>
<tr><td>1f</td><td>Hold_float_DCBA</td></tr>
</table></td></tr>
<tr><td>Delete Modbus Channel</td><td></td><td>ef</td><td>00+Channel ID (1B)</td></tr>
<tr><td>Mobus Channel Name</td><td></td><td>ef</td><td>02+Channel ID (1B) + Name Length (1B) + Name (Mutable)</td></tr>
</table>

Note: Channel ID in downlink commands is different from uplinks:

Channel ID	Description
01	RS485 (Modbus Master) Channel 1
02	RS485 (Modbus Master) Channel 2
03	RS485 (Modbus Master) Channel 3
...	...
10	RS485 (Modbus Master) Channel 16

Examples:

1. Change DO2 (GPIO2) status as high.

04 01ffff	
Channel	**Value**
04(GPIO2)	01ffff=High

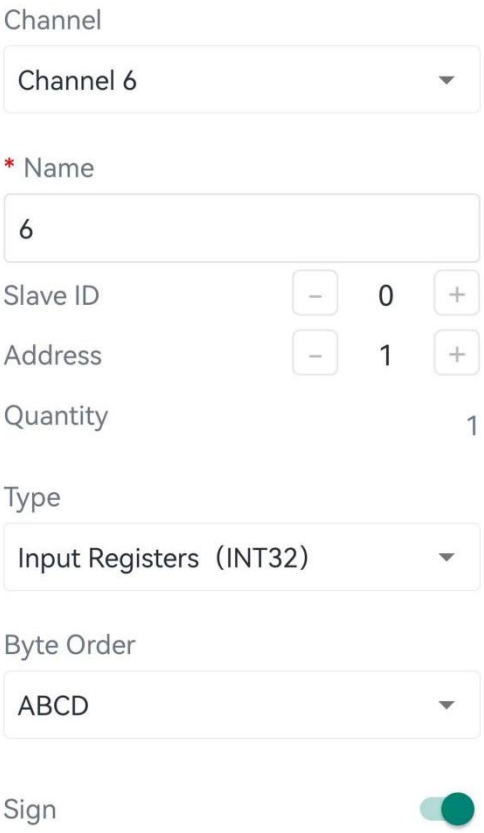
2. Set reporting interval as 20 minutes.

ff 03 b0 04		
Channel	**Type**	**Value**
ff	03	b0 04 => 04 b0 = 1200 s = 20 mins

3. Reboot the device

ff 10 ff		
Channel	**Type**	**Reserved**
ff	10 (Reboot)	ff

4. Add a Modbus channel as below:



ff ef 01 06 00 0100 0e 11		
Channel	Type	Value
ff	ef	Channel: 06=Channel 6 Slave ID: 00 Address: 01 00=>00 01=1 Type: 0e=Input32_ABCD Sign: 11=signed

5. Set name of Modbus channel6 as “test6”.

ff ef 02 06 05 7465737436		
Channel	Type	Value
ff	ef	Channel: 06=Channel 6 Name length: 05=5 Bytes Hex to ASCii: 74 65 73 74 36 => t e s t 6

6. Set the time zone as UTC-2.

ff17ecff		
Channel	Type	Value
ff	17	ec ff => ff ec = -20 the time zone is UTC-2

5. Historical Data Enquiry

UC50x supports sending downlink commands to enquire historical data for specified time point or time range. Before that, ensure the device time is correct and data storage feature has been enabled to store the data.

Command format:

Channel	Type	Description
fd	6b (Enquire data in time point)	4 Bytes, Unix timestamp
fd	6c (Enquire data in time range)	Start time (4 bytes) + End time (4 bytes), Unix timestamp
fd	6d (Stop query data report)	ff
ff	6a (Report Interval)	3 Bytes Byte 1: 01 Byte 2-3: interval time, unit:s range: 30~1200s (60s by default)

Reply format:

Channel	Type	Description
fc	6b/6c	00: data enquiry success 01: time point or time range invalid

		02: no data in this time or time range
20	ef	Data time stamp (4 Bytes) + Periodic Report (Mutable)

Note:

1. The device only uploads no more than 300 data records per range enquiry.
2. When enquiring the data in time point, it will upload the data which is closest to the search point within the reporting interval range. For example, if the device reporting interval is 10 minutes and users send command to search for 17:00's data, if the device finds there is data stored in 17:00, it will upload this data; if not, it will search for data between 16:50 to 17:10 and upload the data which is closest to 17:00.

Example:

1. Enquire historical data between 2024/03/14 22:10:00 to 2024/03/14 22:15:00.

fd6c 3805f365 6406f365		
Channel	**Type**	**Value**
fd	6c (Enquire data in time range)	Start time: 3805f365 => 65f30538 = 1710425400 =2024/03/14 22:10:00 End time: 6406f365 => 65f30664 = 1710425700 =2024/03/14 22:15:00

Reply:

fc6c00		
Channel	**Type**	**Value**
fc	6c (Enquire data in time range)	00: data enquiry success

20ef e105 f365 0300 01 20ef e105f365 04c801000000 20ef e105f365 05f105 8f42 20ef e105f365 09f300030f00 20ef e105f365 b9f30500 20ef e105f365 08f2000e362b302e302b302b32332e370d0a					
Channel	**Type**	**TimeStamp**	**Channel**	**Type**	**Value**
20	cf	e1 05 f3 65 => 65 f3 05 e1 =1710425569s =2024-03-14 22:12:49	03	00	GPIO1-Digital Input 1: 01=High
			04	c8	GPIO2-Counter: 01 00 00 00=>00 00 00 01=1
			05	f1	AI Type: 05=0-10 V Current Value: 8f 42=>42 8f=3.28V
			09	f3	00=Modbus Channel 1 03=Hold16 (Unsigned) Value: 0f 00=>00 0f=15
			b9	f3	05=Modbus Channel 6 Collection Failure
			08	f2	00=Channel 1 Data length: 0e=>15 Data: Hex to ASCII result is 6+0.0+0+23.7

-END-