UC300 IoT Controller

Communication Protocol
(for LoRaWAN® Version)

Revision History

Date	Doc Version	Description
Mar. 31, 2022	V 1.0	Initial version
June 30, 2023	V 2.0	Update based on hardware 2.x

Contents

1. Overview

UC300 uses the standard Milesight IoT payload format based on IPSO. All data are based on following format:

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

Note:

1) All explanations and examples in this document are based on HEX format.
2) For all Milesight IoT decoder examples please find files on https://github.com/Milesight-IoT/SensorDecoders

2. Uplink Payload

Uplink payloads of UC300 are made up of device information and sensor data.

2.1 Device Information

UC300 reports basic device information of device every time joining the network.

Channel	Type	Byte	Description
ff	01 (Protocol Version)	1	01 => V1
	09 (Hardware Version)	2	01 20 => V1.2
	0a (Software Version)	2	01 01 => V1.1
	0b (Power On)	1	Device is on
	16 (Device SN)	8	16 digits

Example:

ff0bff ff0101 ff166445b43411300001 ff090100 ff0a0101		
Channel	**Type**	**Value**
ff	0b(Power On)	ff(Reserved)
ff	01 (Protocol Version)	01 (V1)
ff	16 (Device SN)	64 45 B4 34 11 30 00 01
ff	09 (Hardware Version)	0100 (V1.0)
ff	0a (Software Version)	0101 (V1.1)

2.2 Sensor Data

UC300 reports sensor data according to reporting interval (20min by default). RS232 uplink doesn't have its own channel or type since it only forward RS232 terminal devices data to server transparently.

Channel	Type	Byte	Description
03 (DI 1)	00 (Digital Input)	1	00 = low, 01 = high
	c8 (Counter)	4	UINT32
04 (DI 2)	00 (Digital Input)	1	00 = low, 01 = high
	c8 (Counter)	4	UINT32
05 (DI 3)	00 (Digital Input)	1	00 = low, 01 = high
	c8 (Counter)	4	UINT32
06 (DI 4)	00 (Digital Input)	1	00 = low, 01 = high
	c8 (Counter)	4	UINT32
07 (DO 1)	01 (Digital Output)	1	00 = low, 01 = high
08 (DO 2)			
09 (PT100 1)	67 (Temperature)	2	INT16/10 , unit: °C
0a (PT100 2)			
09 (PT100 1)	e2 (Analog Input)	8	**Byte 1-2:** Current value, float 16 **Byte 3-4:** Min value, float 16 **Byte 5-6:** Max value, float 16 **Byte 7-8:** Average value, float 16
0a (PT100 2)			
0b (4-20mA 1)	02 (Analog Input)	4	UIN32/100, unit: mA
0c (4-20mA 2)			
0d (0-10V 1)			UIN32/100, unit: V

<table>
<tr><td>0e (0-10V 2)</td><td></td><td></td><td></td></tr>
<tr><td>0b (4-20mA 1)</td><td rowspan="3">e2 (Analog Input)</td><td rowspan="3">8</td><td>Byte 1-2: Current value, float 16
Byte 3-4: Min value, float 16
Byte 5-6: Max value, float 16
Byte 7-8: Average value, float 16
Unit: mA</td></tr>
<tr><td>0c (4-20mA 2)</td></tr>
<tr><td>0d (0-10V 1)</td><td rowspan="2">Byte 1-2: Current value, float 16
Byte 3-4: Min value, float 16
Byte 5-6: Max value, float 16
Byte 7-8: Average value, float 16
Unit: V</td></tr>
<tr><td>0e (0-10V 2)</td></tr>
<tr><td>ff</td><td>19 (RS485)</td><td>Mutable (4-7)</td><td>Total: Byte 1+Byte 2+Byte 3+Value
Byte 1: Channel ID
Byte 2: Data Size
Byte 3: 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</td></tr>
<tr><td>03</td><td>Hold16</td></tr>
<tr><td>04</td><td>Hold32</td></tr>
<tr><td>05</td><td>Hold_float</td></tr>
<tr><td>06</td><td>Input32</td></tr>
<tr><td>07</td><td>Input_float</td></tr>
<tr><td>08</td><td>Input_int32_with upper 16 bits</td></tr>
<tr><td>09</td><td>Input_int32_with lower 16 bits</td></tr>
<tr><td>0a</td><td>Hold_int32_with upper 16 bits</td></tr>
<tr><td>0b</td><td>Hold_int32_with lower 16 bits</td></tr>
</table></td></tr>
<tr><td>ff</td><td>15 (Modbus collecting exception)</td><td>1</td><td>Channel ID of failed Modbus collection.</td></tr>
</table>

Note: Channel ID can be configured in ToolBox.

Channel ID	Description
00	RS485 (Modbus Master) Channel 1
01	RS485 (Modbus Master) Channel 2
02	RS485 (Modbus Master) Channel 3
...	...
1f	RS485 (Modbus Master) Channel 31

Examples:

1. Periodic packet:

1) Digital input/counter/output:

03 c8 16 00 00 00 04 00 00 05 00 00 06 00 01 07 01 00 08 01 01		
Channel	**Type**	**Value**
03 (DI 1)	c8 (Pulse Counter)	16 00 00 00 => 00 00 00 16 = 22
04 (DI 2)	00 (Digital Input)	00 => Low
05 (DI 3)	00 (Digital Input)	00 => Low
06 (DI 4)	00 (Digital Input)	01 => High
07 (DO 1)	01 (Digital Output)	00 => Low
08 (DO 2)	01 (Digital Output)	01 => High

2) RS485 Modbus channels:

ff 19 07 02 03 15 00					
Channel	**Type**	**Channel ID**	**Data Size**	**Data Type**	**Value**
ff	19 (RS485)	07 => Channel 8	02 => 2 bytes	03 => Hold 16	15 00 => 00 15 = 21

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

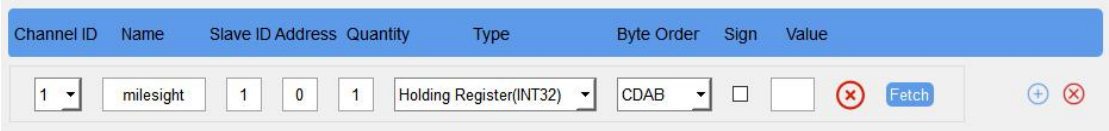
Register Address	Value (Hex)
0	00 15
1	00 20

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)
	CDAB	2097173 (0x00200015)	15 00 20 00 (BADC)
	BADC	352329728 (0x15002000)	00 20 00 15 (CDAB)
	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	/	32 (0x20)	20 00 00 00

with lower 16 bits)			

If UC300 fails to connect the Modbus data, it will an error message.



ff 15 00		
Channel	**Type**	**Value**
ff	15 (Poll Failed)	00 => Channel 1

3) Analog input (4-20mA):

0b 02 02 06 00 00 0c 02 00 00 00 00		
Channel	**Type**	**Avg Value**
0b (4-20mA 1)	02 (Analog Input)	02 06 00 00 => 00 00 06 02 = 1538/100 = 15.38 mA
0c (4-20mA 2)	02 (Analog Input)	00 00 00 00 = 0

4) Analog input (0-10V):

0d e2 8f42 8f42 8f42 8f42 0e e2 0000 0000 0000 0000		
Channel	**Type**	**Avg Value**
0d (0-10V 1)	e2 (Analog Input)	Current/Min./Max./Avg. Value: 8f 42=>42 8f=3.28V
0e (0-10V 2)		00 00 00 00 = 0

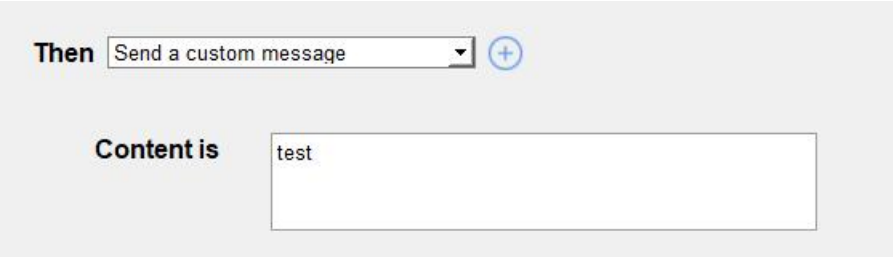
5) PT100 input:

09 67 17 01 0A 67 00 00		
Channel	**Type**	**Avg Value**
09 (PT100 1)	67 (PT100 Input)	17 01 => 01 17 = 279/10 = 27.9 °C
0a (PT100 2)		00 00 00 00 = 0 °C

2. DI/DO change packet: reports when DI or DO status changes.

08 01 01		
Channel	**Type**	**Value**
08 (DO 2)	01 (Digital Output)	01 => High

3. Custom message: take below example,



74 65 73 74
Value
Hex to Ascii: 74 65 73 74=> t e s t

3. Downlink Payload

Downlink is used for controlling the UC300 via network server remotely. Downlink port (Application port) is 85 by default and can be configured via ToolBox.

Note: only one downlink command can be sent to control device everytime.

Item	Channel	Type	Description
DO1	07	/	Open: 01ff, Close: 00ff
DO2	08	/	Open: 01ff, Close: 00ff
Reporting Interval	ff	03	UINT16, unit: s
Reboot		10	ff
UTC Time Zone		17	INT16/10
Data Storage		68	00: disable, 01: enable
Data Retransmission		69	00: disable, 01: enable
Data Retransmission Interval		6a	3 Bytes Byte 1: 00 Byte 2-3: interval time, unit:s range: 30~1200s (600s by default)
DI/DO Debounce Time		91	5 Bytes, **Byte 1:** 00=All, 01=DI1, 02=DI2, 03=DI3, 04=DI4, 05=DO1, 06=DO2 **Byte 2-5:** Debounce time, unit:ms, range: 0~6000ms
DO Duration Control		93	6 Bytes, **Byte 1:** 01=DO1, 02=DO2

			Byte 2: 00=close, 01=open **Byte 3-6:** Duration time, unit:ms, range: 0~86400000ms, 0 means permanent.
Enquiry Current Data		94	ff

Examples:

1. Control DO1 to open (takes effect only when DO is enabled)

0701ff	
Channel	**Value**
07	01ff=open

2. Control DO2 to open for 6s (takes effect only when DO is enabled)

ff93 02 01 70170000		
Channel	**Type**	**Value**
ff	93 (DO Duration Control)	02=DO2, 01=open 70 17 00 00=>00 00 17 70=6000ms=6s

3. Set the reporting interval as 20 minutes.

ff 03 b0 04		
Channel	**Type**	**Value**
ff	03 (Reporting Interval)	b0 04 => 04 b0 = 1200 s = 20 mins

4. Reboot the device.

ff10ff		
Channel	**Type**	**Reversed**
ff	10 (Reboot)	ff

5. Set time zone as UTC-2.

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

4. Historical Data Enquiry

UC300 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 was 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	dc(Basic Historical Data)	Data time stamp (4 Bytes) + Interface Mask (2 Bytes) + Data (Mutable)
20	dd (Modbus Channel Historical Data)	Data time stamp (4 Bytes) + Modbus Mask (4 Bytes) + Data (Mutable)

Interface Mask: when bit=1, the corresponding data interface is enabled.

Bit	7	6	5	4	3	2	1	0
	PT100_2	PT100_1	DO_2	DO_1	DI_4	DI_3	DI_2	DI_1
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
				Custom message	0-10V_2	0-10V_1	4-20mA_2	4-20mA_1

Basic data format:

Interface	Description
DI	DI Type (1 Byte) + Value (1 Byte or 4 Bytes) DI Type: 00=Digital input, 01=Pulse counter
DO	Value (1 Byte)
PT100/4-20mA/0-10V	Current (2 Bytes) +Min (2 Bytes) +Max (2 Bytes) +Average (2 Bytes)

Modbus Mask: when bit=1, the corresponding Modbus channel is configured.

Bit	31	...	15	...	1	0
	Chan32	...	Chan16	...	Chan2	Chan1

Modbus data format: data type (1 Byte) + Value (4 Bytes). The data type format is referred as below:

Bit 7	Bit 6-0
0=unsigned, 1=signed	000000=Coil, 000001=Discrete, 000010=Input 16, 000011=Hold16, 000100=Hold32, 000101=Hold_float, 000110=Input32, 000111=Input_float, 001000=Input_int32_with upper 16 bits, 001001=Input_int32_with upper 16 bits, 001010=Hold_int32_with upper 16 bits, 001011=Hold_int32_with lower 16 bits

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 find 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 2023/03/09 17:00:00 to 2023/03/09 17:10:40.

fd6c 10a00964 90a20964		
Channel	**Type**	**Value**
fd	6c (Enquire data in time range)	Start time: 10a00964 => 6409a010 = 1678352400 =2023/03/09 17:00:00 End time: 90a20964 => 6409a290 = 1678353040 =2023/03/09 17:10:40

Reply:

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

20ce 3fa10964 0098000000 20ce 3fa10964 1700000000			
Channel	**Type**	**Time Stamp**	**Value**
20	ce (Historical	3fa10964 =>	00: Valve 1 close

	Data)	2023/03/09 17:05:00	98 00 00 00 => 00 00 00 98 = pulse counter of GPIO1 is 152 17=10111=>Valve 2 open, DI status of GPIO2 is open

-END-