Lorawan payloads and commands

R56 FW version – H Hw.

Lorawan payload description

The below payloads are valid after setting the sensor in Lorawan mode, after the payload composition the cryptography is applied depending on the status flags.

Sensor to Server

Payload KEEPALIVE

#	Field	Length	Value or explanation
1	Header	1 byte	0x80
2	Sensor address	4 byte	Sensor address
3	Length	1 byte	Payload variable section length (the items from 4 to 12)
4	Fixed data	1 byte	0xF3
5	Fixed data	1 byte	0xF9
6	Status	1 byte	1 -> free, 2 -> busy
7	Magnitude	4 byte	Sensor magnitude value
8	Fixed data	1 byte	0x56
9	Fw version	1 byte	Sensor FW release # (hex number, i.e. 0x38-> 56 version)
10	RSSI	2 byte	Last transmission RSSI value (a positive one)
11	ProgTX	4 byte	Transmitted packet progressive #
12	Retries	1 byte	Transmission retry number #
13	Checksum	2 byte	

Payload MEASURE

#	Field	Length	Value or explanation
1	Header	1 byte	0x80
2	Sensor address	4 byte	Sensor address
3	Length	1 byte	Payload variable section length (the items from 4 to 12)
4	Fixed data	1 byte	0xF4
5	Fixed data	1 byte	0x00
6	Fixed data	1 byte	0x01
7	Time	4 byte	Sensor time
8	Status	1 byte	1 -> free, 2 -> busy
9	Version	1 byte	Sensor FW release # (hex number, i.e. 0x38-> 56 version)
10	Magnitude	4 byte	Sensor magnitude value
11	ProgTX	4 byte	Transmitted packet progressive #
12	Retries	1 byte	Transmission retry number #
13	Checksum	2 byte	

Payload ACK (ACK to received Command)

#	Field	Length	Value or explanation
1	Header	1 byte	0xC0
2	Sensor address	4 byte	Sensor address
3	Length	1 byte	Payload variable section length (the items from 4 to 6)
4	Command	1 byte	Received command to be acknowledged
5	FW version	1 byte	Sensor FW release # (hex number, i.e. 0x38-> 56 version)
6	ProgTX	4 byte	Transmitted packet progressive #
7	Checksum	2 byte	

Payload CALIBRATION DONE (CalDone - Reply to Calibration command)

#	Field	Length	Value or explanation
1	Header	1 byte	0x80
2	Sensor address	4 byte	Sensor address
3	Length	1 byte	Payload variable section length (from 4 to 5 fields)
4	Fixed data	1 byte	0xF5
5	ProgTX	4 byte	Transmitted packet progressive #
6	Checksum	2 byte	

Payload INFO@RESET

#	Field	Length	Value or explanation
1	Header	1 byte	0x80
2	Sensor address	4 byte	Sensor address
3	Length	1 byte	Payload variable section length (from 4 to 8 fields)
4	Fixed data	1 byte	0xFE
5	Fw version	1 byte	Sensor FW release # (hex number, i.e. 0x38-> 56 version)
6	Status	1 byte	1 -> free, 2 -> busy
7	Reset cause	2 byte	Reset type
8	Flags	2 byte	Sensor status flags
9	Checksum	2 byte	

Lorawan commands description

The below commands from network server to sensor are valid after setting the sensor in Lorawan mode.

Server to Sensor

Acknowledge – ACK (to Measure and Keepalive)

#	Field	Length	Value or explanation
1	Start message	1 byte	0xE0
2	Sensor address	4 byte	Sensor address
3	Length	1 byte	6 (referring to the below items 4, 5 and 6)
4	Header	1 byte	0xF4
5	Command	1 byte	0x40
6	Time	4 byte	Unix time stamp (date and time)
7	Checksum	2 byte	The checksum refers to the items from 1 to 6, see below Appendix A for further data.

Example

//For this example Sensor Address: 605 (0x000025D)

Unix timestamp 1561644202 (06/27/2019 @ 2:03pm (UTC)), i.d. 0x5D14CCAA

unsigned char CommandBuffer[50];

CommandBuffer [0] = 0xE0;

CommandBuffer [1] = 0x00;

CommandBuffer [2] = 0x00;

CommandBuffer [3] = 0x02;

CommandBuffer [4] = 0x5D;

CommandBuffer [5] = 0x06;

CommandBuffer [6] = 0xF4;

CommandBuffer [7] = 0x40;

CommandBuffer [8] = 0x5D;

CommandBuffer [9] = 0x14;

CommandBuffer [10] = 0xCC;

CommandBuffer [11] = 0xAA;

CommandBuffer [12] = 0xCA;

CommandBuffer [13] = 0x48;

Command **CALIBRATION**

#	Field	Length	Value or explanation
1	Start message	1 byte	0xE0
2	Sensor address	4 byte	Sensor address
3	Length	1 byte	0x02 (referring to the below items 4 and 5)
4	Header	1 byte	0xF3
5	Command	1 byte	0x01
6	Checksum	2 byte	The checksum refers to the items from 1 to 5, see below Appendix A for further data.

Example

//For this example Sensor Address: 605 (0x000025D)

unsigned char CommandBuffer[50];

CommandBuffer [0] = 0xE0;

CommandBuffer [1] = 0x00;

CommandBuffer [2] = 0x00;

CommandBuffer [3] = 0x02;

CommandBuffer [4] = 0x5D;

CommandBuffer [5] = 0x02;

CommandBuffer [6] = 0xF3;

CommandBuffer [7] = 0x01;

CommandBuffer [8] = 0xDE;

CommandBuffer [9] = 0x2F;

Command **SWITCH-TO-LORA ITC** (Maintenance Network)

#	Field	Length	Value or explanation
1	Start message	1 byte	0xE0
2	Sensor address	4 byte	Sensor address
3	Length	1 byte	4 (referring to the below items 4, 5 and 6)
4	Header	1 byte	0xF3
5	Command	1 byte	0x57
6	Fixed data	2 byte	0x0000
7	Checksum	2 byte	The checksum refers to the items from 1 to 6, see below Appendix A for further data.

Example

//For this example Sensor Address: 605 (0x000025D)

unsigned char CommandBuffer[50];

CommandBuffer [0] = 0xE0;

CommandBuffer [1] = 0x00;

CommandBuffer [2] = 0x00;

CommandBuffer [3] = 0x02;

CommandBuffer [4] = 0x5D;

CommandBuffer [5] = 0x04;

CommandBuffer [6] = 0xF3;

CommandBuffer [7] = 0x57;

CommandBuffer [8] = 0x00;

CommandBuffer [9] = 0x00;

CommandBuffer [8] = 0xF1;

CommandBuffer [9] = 0xA7;

APPENDIX A

Checksum calculation

The checksum calculation is done accordingly to the below function

```
#define                 POLYNOMIAL          0x8408u
#define                 PRESET_VALUE        0xFFFFu
unsigned int chks_calc(unsigned int startAddress, unsigned int endAddress, unsigned char *bytes)
{
  unsigned int i, j;
  unsigned int returnvalue = PRESET_VALUE;

  for (i=startAddress;i<endAddress;i++) {
        returnvalue = returnvalue ^ ((unsigned int)bytes[i] & 0x00ff);
        for (j=0;j<8;j++)
              if ((returnvalue & 0x0001) != 0)
                    returnvalue = (returnvalue >> 1) ^ POLYNOMIAL;
              else
                    returnvalue = returnvalue >> 1;
  }
  return returnvalue;
}
```

Example

```
unsigned char CommandBuffer[50];
unsigned int Cksum;

CommandBuffer [0] = 0xE0;
CommandBuffer [1] = 0x00;
CommandBuffer [2] = 0x00;
CommandBuffer [3] = 0x02;
CommandBuffer [4] = 0x5D;
CommandBuffer [5] = 0x02;
CommandBuffer [6] = 0xF3;
CommandBuffer [7] = 0x01;

Cksum = chks_calc(0, 8, CommandBuffer);
```

The result is Cksum = 0xDE2F