

# Interface / API Description for WS-23XX

This document is unofficial and contains information that owners of the WS-23XX have collected by experiments.

The purpose of this document is to help owner using the product under Linux to write their own interface program for the Weather station.

This site is not associated with the manufacturer in anyway. There is no guarantee of the correctness of the information in this document and the use of the information is at your own risk. The author cannot be held responsible for any damage use of this information may cause.

The weather station actually does not really have an API. It has a few commands to read and write to the memory of the weather station. All the data you collect from the weather station is simply raw data fetched from the memory.

This topic describes how the physical RS232 interface and how to read and write data to and from the weather station.

## Serial Port Settings.

Baudrate 2400 Bits 8 Parity None Stopbits 1

## Physical connection

<u>RS232</u>	<u>Wire Colour</u>	<u>Function</u>
Pin 2 RXD	Orange wire	Receive data from WS2300
Pin 3 TXD	Green wire	Transmit data to WS2300
Pin 4 DTR	White wire	Signal from PC to WS2300
Pin 7 RTS	Blue wire	Signal from PC to WS2300

Note:

There is no ground connection.

DTR and RTS are not used for handshake. They are steady DTR at negative voltage and RTS at positive voltage.

Since there is no ground it seems that the WS2300 uses the DTR and RTS to define high and low. DTR must be low and RTS must be high for the communication to work.

## Basic Protocol

The communication with the WS-2300 is very primitive. There is no real user interface. The only thing you can do is read and write from the memory of the device. The device seems to a memory area of around 5k. Memory area starts at 0000 and ends around 13C0.

Most data except alarm flags are in the range 200-6C5. History records starts from 6C6. Each record is 19 bytes.

Commands are normally sent 1 byte at a time and one byte is returned as a receipt.

Initial command is 0x06 which signals to start all over with same data as before.

Twice 0x06 without a read between seems to mean rewind to initial data

Except for the 0x06 commands are always 5 bytes.

## Errors

When reading and writing to the WS 2300 the communication often fails. The station simply does not see RS232 communication as first priority and refuses to answer when it does something else. Errors happen quite often.

When you get no answer or a wrong check byte returned, start again by sending 06. You should now get the answer 02 back. If not repeat sending 06 until you receive 02 and then the entire command all over again.

## Reading

- First 4 bytes is the first address to read from.
- The address is a 13 bit number which can be seen as 4 hex digits (4-bit nibbles) (First digit is either 0 or 1).
- The address is coded as  $\text{hexdigit} \times 4 + 0x82$ .

- WS2300 returns 0S, 1S, 2S and 3S where S is a check digit calculated as  $(\text{command} - 0x82)/4$ . I.e. S is the hex digits of the address.

A more graphical way to see it is like this.

1	0	0	0	0	0	1	0	0x82
+								
0	0	A3	A2	A1	A0	0	0	Address * 4
=								
1	0	A3	A2	A1	A0	1	0	Command

Here is a table that translate between hex digit and command.

Command	Hex Digit	Command	Hex Digit
82	0	A2	8
86	1	A6	9
8A	2	AA	A
8E	3	AE	B
92	4	B2	C
96	5	B6	D
9A	6	BA	E
9E	7	BE	F

When reading data the last byte is the number of databytes requested. It is coded as  $\text{number} * 4 + 0xC2$ . See table below. Max value is 15.

1	1	0	0	0	0	1	0	0xC2
+								
0	0	N3	N2	N1	N0	0	0	Number * 4
=								
1	1	N3	N2	N1	N0	1	0	Command

Here is a table that translate between number and command.

Command	Number		Command	Number
C6	1		E6	9
CA	2		EA	10
CE	3		EE	11
D2	4		F2	12
D6	5		F6	13
DA	6		FA	14
DE	7		FE	15
E2	8			

WS2300 returns 3X where X is the number of databytes to follow (excl checksum byte). The first byte 3 is some fixed header.

Data bytes are now all sent at once.

Last a checksum is sent which is the low byte of the sum of the databytes sent.

## Writing 4-bit Nibbles

When writing you set up the address the same way as when reading.

To write a nibble (4-bits) the data write command byte is calculated as  $(data\_nibble * 4) + 0x42$ .

0	1	0	0	0	0	1	0	0x42
+								
0	0	D3	D2	D1	D0	0	0	Data * 4
=								
0	1	D3	D2	D1	D0	1	0	Command

Here is a table that translate between number and command.

Command	Hex Digit		Command	Hex Digit
42	0		62	8
46	1		66	9
4A	2		6A	A
4E	3		6E	B

52	4		72	C
56	5		76	D
5A	6		7A	E
5E	7		7E	F

The station will acknowledge every written nibble write command by returning (0x10 + hexdigit).

0	0	0	1	D3	D2	D1	D0	Acknowledge Data
---	---	---	---	----	----	----	----	---------------------

After the last data byte is written you send can either send 06 init command (acknowledged by 0x02) or a new read/write address (0x82/0x83) which is then acknowledged by 0x00/0x01. It is probably a good idea to end the writing in some controlled way so that no more data is written to the station.

## Writing 1-bits

Same method as for writing nibbles except data command have the following format. Command for setting a bit (numbered from 0 to 3) is (bit\_no\*4) + 0x12 and the acknowledge is 0x04+bit\_no. Command for resetting a bit is (bit\_no\*4) + 0x32 and the acknowledge is 0x0C+bit\_no. Setting a bit

0	0	0	1	0	0	1	0	0x12
+								
0	0	0	0	B1	B0	0	0	Bit Number * 4
=								
0	0	0	1	B1	B0	1	0	Command

Acknowledge setting bit

0	0	0	0	0	1	B1	B0	Acknowledge Data
---	---	---	---	---	---	----	----	---------------------

Resetting a bit

0	0	1	1	0	0	1	0	0x32
+								

0	0	0	0	B1	B0	0	0	Bit Number * 4
=								
0	0	1	1	B1	B0	1	0	Command

Acknowledge resetting bit

0	0	0	0	1	1	B1	B0	Acknowledge Data
---	---	---	---	---	---	----	----	---------------------