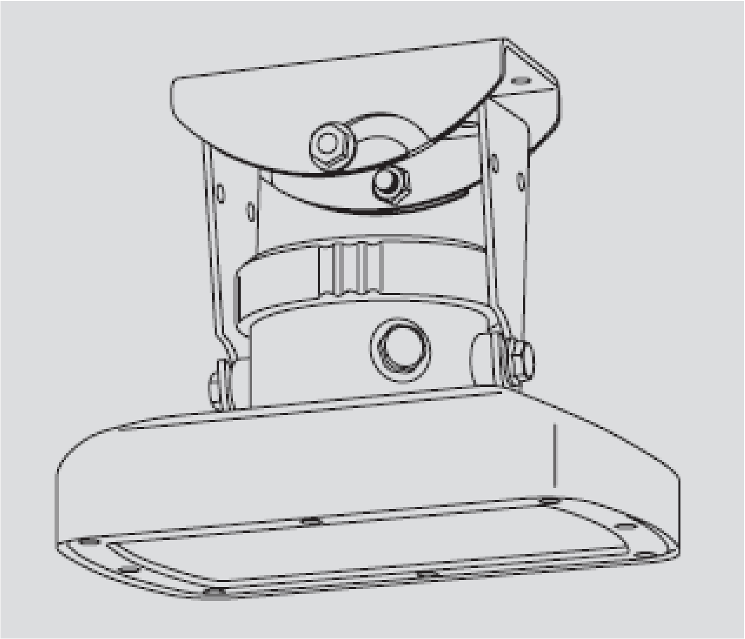
# OTT RLS

## RS 485 (SDI-12 Protocol)
## Basic Information for Data Logger Implementation



English

## 1 Scope of supply

_ **OTT RLS** – 1 Radar sensor OTT RLS,
incl. two part swivel mount (consisting of device and wall brackets)
– 1 Installation kit (4 x wood screws 6 x 40 mm; 4 x plastic plugs S8)
– 2 Double open-ended wrenches size 10 x 13
– 1 Operating instructions
– 1 Factory acceptance test certificate (FAT)

## 2 Order numbers

_ **OTT RLS Radar sensor OTT RLS** 63.105.001.9.2
– Version 4: RS-485 + 4 … 20 mA interface
– Version S: SDI-12 + 4 … 20 mA interface

_ **Accessories Connection cable** 97.000.039.9.5
– twisted-pair design
– PVC, black
– 2 x 2 x 0,5 $mm^2$
approx. AWG 22 (7 x AWG 30)

**Connection cable** 97.000.040.9.5
– twisted-pair design
– PVC, black
– 2 x 2 x 0,75 $mm^2$
AWG 20 (7 x AWG 28)

_ **Accessories: USB/SDI-12 Converter** 65.050.001.9.2
_ multipurpose interface box for RS485 / SDI-12 interfaces
_Connection cable for OTT RLS
_ USB cable
_ Driver CD
_ Terminal program

## 3 Introduction – OTT RLS

The OTT RLS radar sensor is used for contact less measurement of the levels of surface water.

Three standardized electrical interfaces are available for connecting the OTT RLS to a data logger or peripheral devices: 4 … 20 mA, SDI-12 and RS-485 (SDI-12 protocol).
The 4…20mA signal can be scaled via SDI-12 commands.

The OTT RLS is available respectively will be delivered in 2 versions:

Type VAR code "S" with SDI-12 interface and 4…20 mA
Type VAR code "4" with RS485(SDI-12 protocol) and 4…20 mA

**OTT RLS – factory settings - general remarks**
Independent which type is used, the 4…20 mA interface of the OTT RLS is preset "active"
Measurement modus: 4…20 mA
Scaling 4…20 mA equal to 0….35 m (complete measurement range)
SDI-12 sensor address: 0

**OTT RLS – Version RS 485 (SDI-12 protocol)**
The OTT RLS type VAR code "4" uses a RS 485 as digital output. The protocol which is used is the "SDI-12" protocol. Using the RS 485 it is possible to connect data cable of up to 1000m between sensor and datalogger. The reason for choosing SDI-12 protocol: It's a non propriety protocol, defined as standard for micro processor base sensors. Characteristic of the SDI-12 protocol – ASCII protocol with defined timing of communication

**To implement it is necessary to know about timing and protocol description.**

**General aspects will be shown in this document – the whole and detailed description of the protocol can be downloaded under www.sdi-12.org**

# 4 SDI-12 Communication Protocol – General information

Datalogger and sensors (OTT RLS) communicate by an exchange of ASCII characters on the data line. The data logger sends a break to wake up the sensor (OTT RLS) on the data line. A break is continuous spacing on the data line for at least 12 milliseconds. The datalogger then sends a command. The sensor, in turn, returns the appropriate response. Each command is for a specific sensor. The first character of each command is a unique sensor address that specifies with which sensor the datalogger wants to communicate. When a datalogger tells a sensor to start its measurement procedure, the datalogger does not communicate with any other sensor until the data collection from the sensor is complete. (During a concurrent measurement command, however, a datalogger can communicate with other sensors while one or more sensors are taking measurements.

A typical data logger /sensor (OTT RLS) measurement sequence proceeds as follows:

Step 1. The datalogger wakes the sensor on the RS 485  with a break.

Step 2. The data logger  transmits a command to a specific, addressed sensor, instructing it to make a measurement.

Step 3. The addressed sensor responds within 15.0 milliseconds returning the maximum time until the measurement data will be ready and the number of data values it will return.

Step 4. If the measurement is immediately available, the datalogger transmits a command to the sensor instructing it to return the measurement(s). If the measurement is not ready, the datalogger waits for the sensor to send a request to the datalogger, which indicates that the data are ready. The datalogger then transmits a command to get the data.

Step 5. The sensor responds, returning one or more measurements.

### Baud Rate and Byte Frame Format

The baud rate for SDI-12 is 1200.

1 start bit
7 data bits, least significant bit transmitted first
1 parity bit, even parity
1 stop bit

### Allowable Characters

All characters transmitted on the SDI-12 bus must be printable ASCII characters. Table 3 shows the printable characters.
space, 32 decimal
through
~, 126 decimal

There are two exceptions:

1) all responses from an SDI-12 sensor end with a carriage return (0D hex, 13 decimal) and a line feed (0A hex, 10 decimal) character, shown as <CR><LF> in this document;

2) in some cases the second and third character of a CRC code may not be printable ASCII characters.

**Device/sensor addresses**

The first character of every command must be a sensor address. Likewise, the first character of a response is also the address character. This lets an datalogger verify that the response has come from the correct sensor. (An address is a single character used to indicate which sensor is to respond to the command.) Table 4 shows the address codes.

| ASCII address (a single character) | Decimal | Hex | Description |
|---|---|---|---|
| "0" (zero) | 48 | 30 | Default address, all sensors are initially set to "0" (zero) by the manufacturer for use in single sensor systems |
| "1" to "9" | 49 to 57 | 31 to 39 | Addresses for additional sensors on the SDI-12 bus |

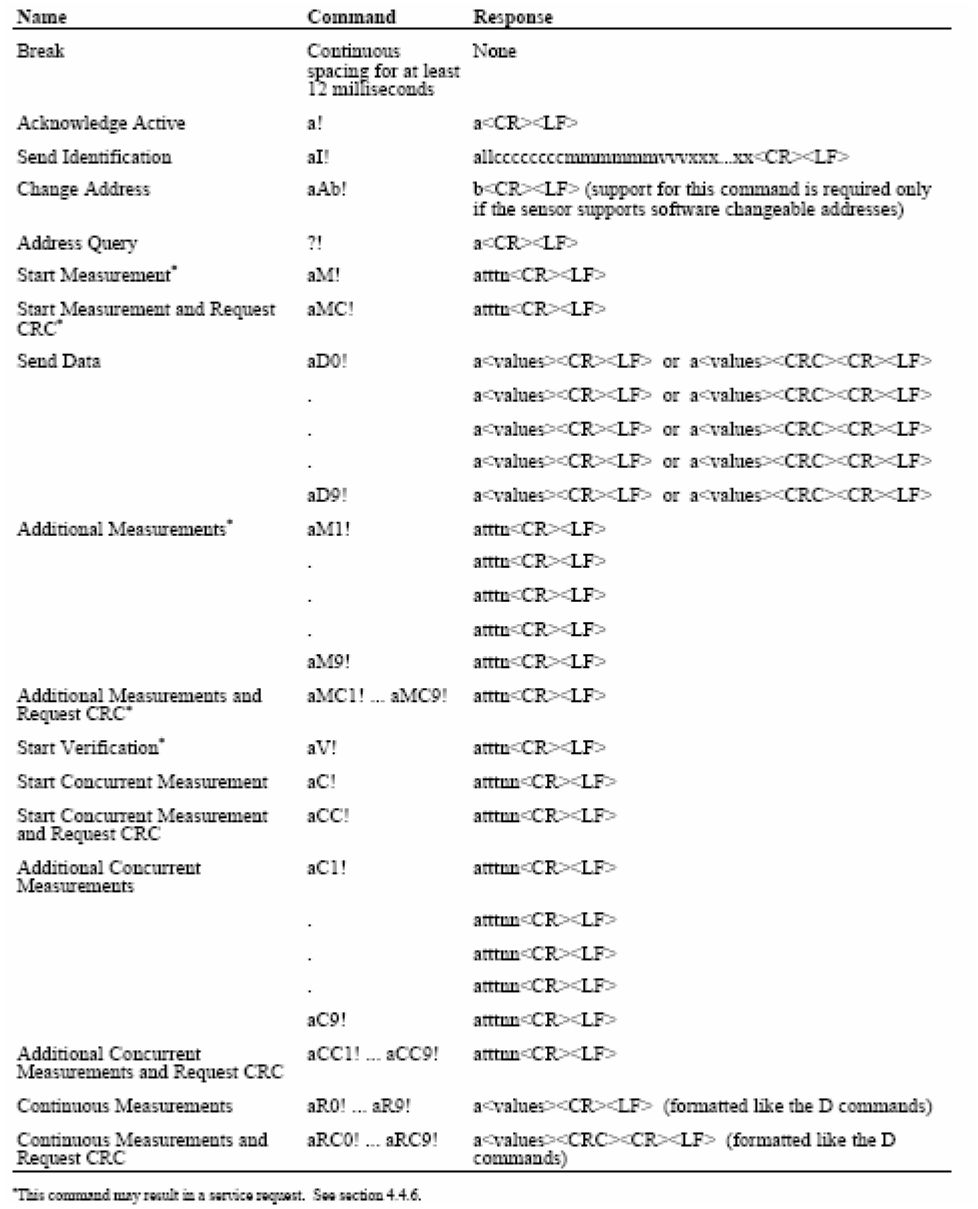
Table 4. Sensor address codes

ASCII '0' through ASCII '9' are the standard addresses which all sensors and datalogger  must support. Should there be a need for more than 10 sensors, use an address in the range ASCII 'A' through ASCII 'Z' (decimal 65 through 90) and ASCII 'a' through ASCII 'z' (decimal 97 through 122).

## 5 Commands and responses

Table 5 lists each basic SDI-12 command, its format, and the format of each response to a command.

Table 5



| Name | Command | Response |
|---|---|---|
| Break | Continuous spacing for at least 12 milliseconds | None |
| Acknowledge Active | a! | a<CR><LF> |
| Send Identification | aI! | allccccccccmmmmmmvvvxxx...xx<CR><LF> |
| Change Address | aAb! | b<CR><LF> (support for this command is required only if the sensor supports software changeable addresses) |
| Address Query | ?! | a<CR><LF> |
| Start Measurement* | aM! | atttn<CR><LF> |
| Start Measurement and Request CRC* | aMC! | atttn<CR><LF> |
| Send Data | aD0! | a<values><CR><LF> or a<values><CRC><CR><LF> |
| | . | a<values><CR><LF> or a<values><CRC><CR><LF> |
| | . | a<values><CR><LF> or a<values><CRC><CR><LF> |
| | . | a<values><CR><LF> or a<values><CRC><CR><LF> |
| | aD9! | a<values><CR><LF> or a<values><CRC><CR><LF> |
| Additional Measurements* | aM1! | atttn<CR><LF> |
| | . | atttn<CR><LF> |
| | . | atttn<CR><LF> |
| | . | atttn<CR><LF> |
| | aM9! | atttn<CR><LF> |
| Additional Measurements and Request CRC* | aMC1! ... aMC9! | atttn<CR><LF> |
| Start Verification* | aV! | atttn<CR><LF> |
| Start Concurrent Measurement | aC! | atttnn<CR><LF> |
| Start Concurrent Measurement and Request CRC | aCC! | atttnn<CR><LF> |
| Additional Concurrent Measurements | aC1! | atttnn<CR><LF> |
| | . | atttnn<CR><LF> |
| | . | atttnn<CR><LF> |
| | . | atttnn<CR><LF> |
| | aC9! | atttnn<CR><LF> |
| Additional Concurrent Measurements and Request CRC | aCC1! ... aCC9! | atttnn<CR><LF> |
| Continuous Measurements | aR0! ... aR9! | a<values><CR><LF> (formatted like the D commands) |
| Continuous Measurements and Request CRC | aRC0! ... aRC9! | a<values><CRC><CR><LF> (formatted like the D commands) |

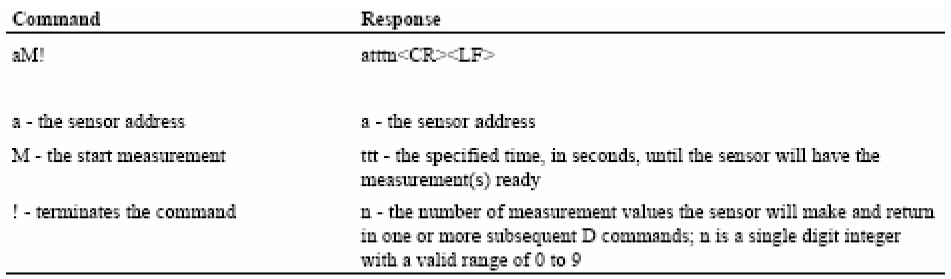
*This command may result in a service request. See section 4.4.6.

The first character of all commands and responses is always a device address. The last character of a command is the "!" character. The “!” can only be in a command as the command terminator. The last two bytes of a response are a carriage return and line feed (<CR><LF>).

The maximum number of characters that can be returned in the <values> part of the response to a D command is either 35 or 75. If the D command is issued to retrieve data in response to a concurrent measurement command, the maximum is 75. The maximum is also 75 in response to a continuous measurement command. Otherwise, the maximum is 35.

## Description of selected commands

**Start Measurement Command (aM!)**
This command tells the sensor to take a measurement. The sensor does not, however, return the measurement to the datalogger after this command. It returns the time until one or more measurements will be ready and the number of measurements that it will make. The send data (D0!) command must be issued to get the measurement(s).



| Command | Response |
|---|---|
| aM! | atttn<CR><LF> |
| a - the sensor address | a - the sensor address |
| M - the start measurement | ttt - the specified time, in seconds, until the sensor will have the measurement(s) ready |
| ! - terminates the command | n - the number of measurement values the sensor will make and return in one or more subsequent D commands; n is a single digit integer with a valid range of 0 to 9 |

If the sensor returns 000 (ttt), the measurement is immediately available for transfer to the datalogger. The datalogger should issue the D0 command to get the data.
I
f ttt is not equal to zero (000), the datalogger must wait for the specified time to elapse. The ttt time period begins upon completion of the transmission of the line feed character.

Then it wakes the sensor with a break and issues the D0 command. If, however, the sensor has the measurement ready before ttt seconds elapse, it will send a service request to the datalogger.

This tells the datalogger to stop marking time and issue the D0 command.
When a datalogger issues an M command, it must complete the command/response sequence with the sensor before it sends any command to any other sensor. For example, suppose that following the issuance of the M command, the sensor responds
0M!00101<CR><LF>

This response shows that one data value will be ready in 10 seconds. The datalogger must wait for either of two events to occur before issuing a D0 command:

1) receipt of a service request (a<CR><LF>) from the sensor
2) the specified time to elapse (10 seconds)

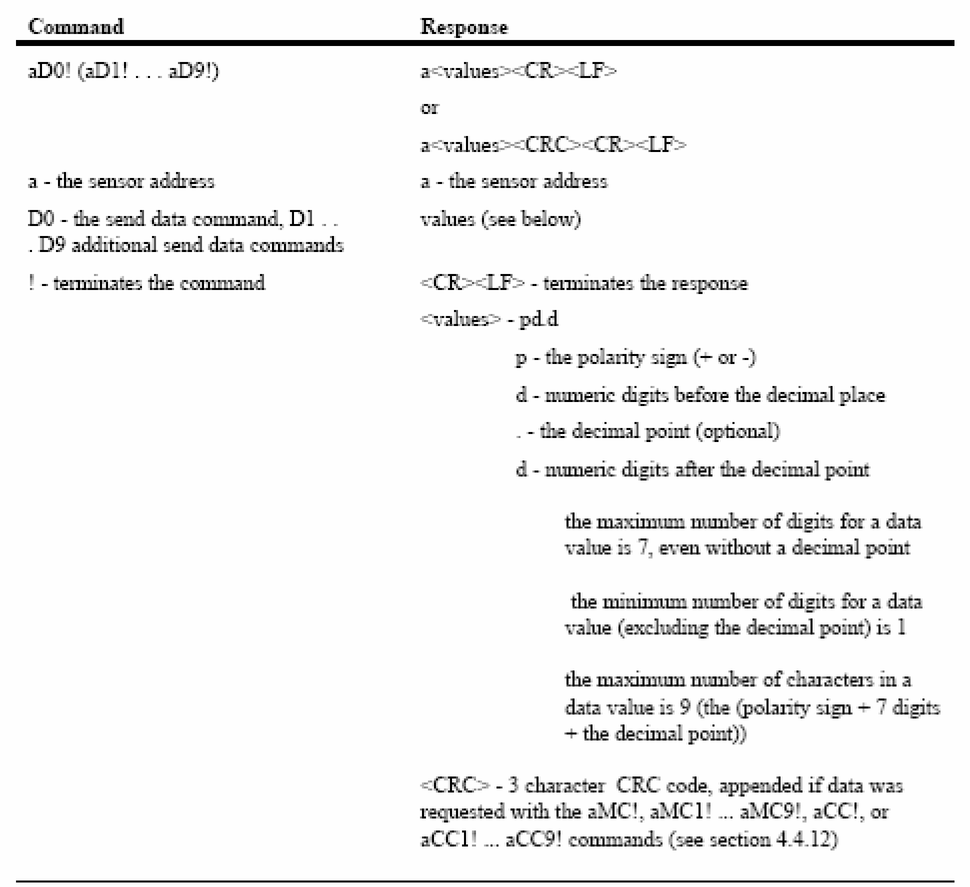
The datalogger then issues the D0 command to get the data. After the M command and before the sensor issues the service request, the datalogger will not communicate with any other sensor.
The sensor must not drive the data line until it sends a service request. The datalogger must not drive the data line while waiting for the service request.
A sensor should return a ttt value greater than the time it takes to make a measurement, to allow for timing tolerances and for the service request.
After a sensor finishes its measurement procedure, the sensor must retain the measured data in memory until it receives another M or V command, or another command that returns datavia the D0 command.

**Service Request (a<CR><LF>)**
A service request is a response from a sensor. It is not a command. It is sent, after an M command, to tell the datalogger that the sensor has finished its measurement(s) and the data are ready. A service request is issued by the sensor after an M, MC, or V command, when it has finished its measurement. The entire service request must be returned before ttt seconds (see Start Measurement Command) have elapsed. The time (ttt) is the maximum time that a sensor will take before it has data available.

If a datalogger does not detect a service request, the datalogger must wait for the specified time (ttt), given in response to the M or V command, and then issue the D0 command to get the data. The minimum time before a datalogger can issue the D0 command to get the data, in this case, is ttt seconds.

A sensor is required to issue a service request if it indicates that it will take one second or longer before data are ready, in response to an M or V command. If a sensor says it will take zero seconds before the data are ready, then a sensor must not issue a service request.

**Send Data Command (aD0! . . . aD9!)**
This command is used to get groups of data from the sensor. D0! is issued after an M, C, C, CC, or V command. The sensor responds by sending the data. If the expected number of measurements is not returned in response to a D0! command, the datalogger issues D1!, D2!, etc. until all measurement values are received. (The expected number of measurements is given in the response to an M, C, or V command.) Table shows the send data command.



| Command | Response |
|---|---|
| aD0! (aD1! . . . aD9!) | a<values><CR><LF> |
| | or |
| | a<values><CRC><CR><LF> |
| a - the sensor address | a - the sensor address |
| D0 - the send data command, D1 . . . D9 additional send data commands | values (see below) |
| ! - terminates the command | <CR><LF> - terminates the response |
| | <values> - pd.d |
| | p - the polarity sign (+ or -) |
| | d - numeric digits before the decimal place |
| | . - the decimal point (optional) |
| | d - numeric digits after the decimal point |
| | the maximum number of digits for a data value is 7, even without a decimal point |
| | the minimum number of digits for a data value (excluding the decimal point) is 1 |
| | the maximum number of characters in a data value is 9 (the (polarity sign + 7 digits + the decimal point)) |
| | <CRC> - 3 character CRC code, appended if data was requested with the aMC!, aMC1! ... aMC9!, aCC!, or aCC1! ... aCC9! commands (see section 4.4.12) |

If the response to a D command is valid, but no data are returned, the sensor has aborted the measurement. To obtain data the datalogger must issue another M, C, or V command.

The maximum number of characters that can be returned in the <values> part of the response to a D command is either 35 or 75. If the D command is issued to retrieve data in response to a concurrent measurement command, the maximum is 75. Otherwise, the maximum is 35.

**Examples of the Start Measurement Command (aM!) and the Send Data Commands**

a. One measurement is immediately available after the M command:

```
0M!00001<CR><LF>
0D0!0+3.14<CR><LF>
```

b. Three measurements will be ready 5 seconds after the M command, and the sensor issues a service request. All 3 measurements are returned in response to the D0 command:

```
0M!00053<CR><LF>
0<CR><LF>
0D0!0+3.14+2.718+1.414<CR><LF>
```

c. Nine measurements will be ready 35 seconds after the M command, and the sensor issues a service request. Because the number of characters in all 9 measurements exceeds the limit in the<values> field, a D1 command must be issued to get the second group of measurements:

```
0M!00359<CR><LF>
0<CR><LF>
0D0!0+1.11+2.22+3.33+4.44+5.55+6.66<CR><LF>
0D1!0+7.77+8.88+9.99<CR><LF>
```

d. Two measurements will be available in 1 second, and the sensor does not issue a service request. After 1 second, the datalogger sends a break to wake the sensor and issues the D0 command:

```
0M!00012<CR><LF>
0D0!0+3.14+2.718<CR><LF>
```

Note: this example shows the proper operation of the dataloggerr, but the sensor is out of compliance because it did not issue a service request.

e. Three measurements will be ready 5 seconds after the M command, and the sensor issues a service request. Upon receipt of the service request, the datalogger issues D0 to get the data. However, only 1 measurement is returned. The datalogger then issues the D1 command to get the next group of data. In response, the second measurement is returned. Then the datalogger issues D2 to get the next, and last, group of data. In the example, each group contains only one measurement.

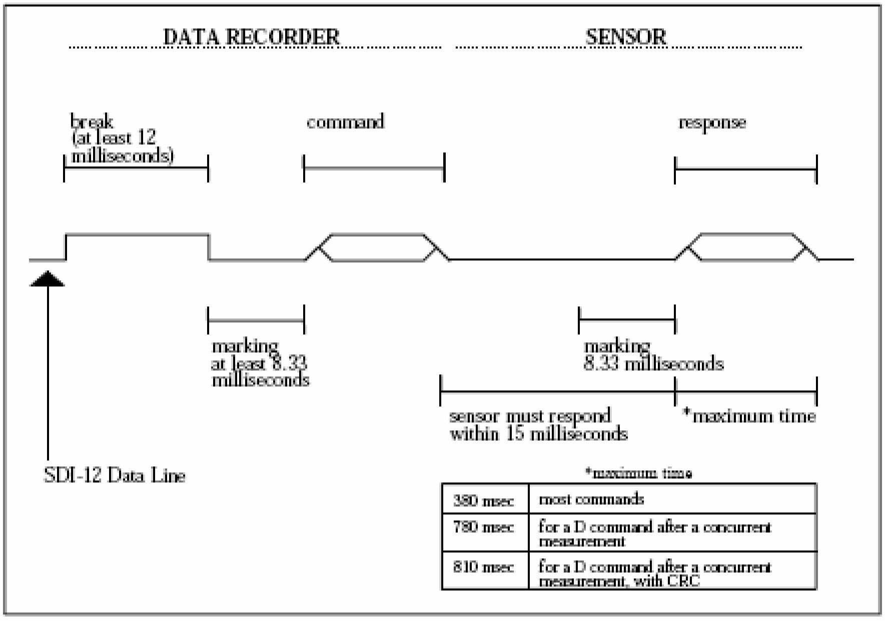
```
0M!00053<CR><LF>
0<CR><LF>
0D0!0+3.14<CR><LF>
0D1!0+2.718<CR><LF>
0D2!0+1.414<CR><LF>
```

Note: This is in compliance with the standard. As many measurements as possible, however, should be returned in response to each D command.

## 6 SDI-12 - timing

Figure 3 shows a timing diagram for an SDI-12 command and its response. The tolerance for all SDI-12 timing is ± 0.40 milliseconds. The only exception to this is the time between the stop bit of one character and the start bit of the next character. The maximum time for this is 1.66 milliseconds, with no tolerance.

Fig.3


DATA RECORDER
SENSOR
break (at least 12 milliseconds)
command
response
marking at least 8.33 milliseconds
marking 8.33 milliseconds
sensor must respond within 15 milliseconds
*maximum time
SDI-12 Data Line
*maximum time

| | |
|---|---|
| 380 msec | most commands |
| 780 msec | for a D command after a concurrent measurement |
| 810 msec | for a D command after a concurrent measurement, with CRC |

- A datalogger transmits a break by setting the data line to spacing for at least 12 milliseconds.
- The sensor will not recognize a break condition for a continuous spacing time of less than 6.5 milliseconds and will always recognize a break when the line is continuously spacing for more than 12 milliseconds.
- Upon receiving a break, a sensor must detect 8.33 milliseconds of marking on the data line before it looks for an address.
- A sensor must wake up from a low-power standby mode and be capable of detecting a start bit from a valid command within 100 milliseconds after detecting a break.
- After a datalogger transmits the last character of a command, it must relinquish control of the data line within 7.5 milliseconds following the end of the stop bit. (Tolerance: +0.40 milliseconds.)
- After receiving the break and the command, the addressed sensor sets the data line to marking for 8.33 milliseconds and then send the response. (Tolerance: -0.40 milliseconds.) The start bit of the first response byte must start within 15 milliseconds after the stop bit of the last byte of the command. (Tolerance: +0.40 milliseconds.)
- After a sensor transmits the last character of a response, it must relinquish control of the data line within 7.5 milliseconds. (Tolerance: +0.40 milliseconds.)
- No more than 1.66 milliseconds of marking are allowed between the end of the stop bit and the start bit (e.g., between characters) on any characters in the command or the mresponse. (No tolerance.) This permits a response to an M command to be sent within a 380 millisecond window.
- Sensors must return to a low-power standby mode after receiving an invalid address or after detecting a marking state on the data line for 100 milliseconds. (Tolerance: +0.40 milliseconds.)
- When a datalogger addresses a different sensor, or if the data line has been in the marking state for more than 87 milliseconds, the next command must be preceded by a break.

### Rules for the Break

The datalogger sends a break when it is necessary to wake a sensor from low-power standby mode. An SDI-12 sensor is required to return to low-power standby mode after receiving an invalid address or after detecting a marking state on the data line for 100 milliseconds. For this reason, a break must precede a command whenever a new sensor is addressed and after 87 milliseconds of marking on the data line.

When a datalogger receives a service request, it does not have to send a break if it issues the D0 command within 87 milliseconds after the service request. If, however, more than 87 milliseconds elapse, the D0 command must be preceded with a break.

### Retries

A datalogger must support retries. Sensors have up to 100 milliseconds to wake up after detecting a break and will not respond to any commands until they have awakened. Also, sensors will go to sleep after detecting 100 milliseconds of idle time (marking) on the SDI-12 bus.

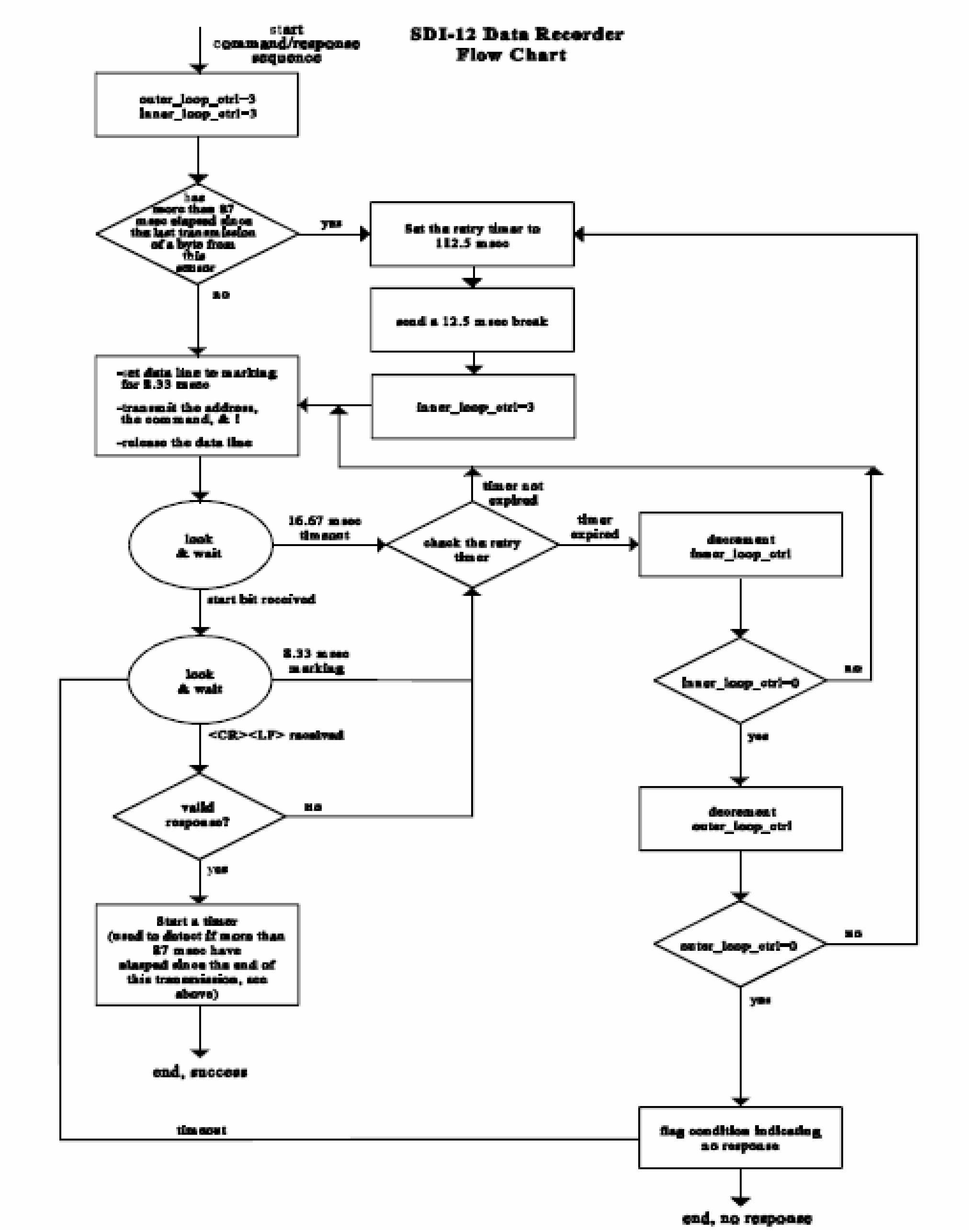
When issuing retries, if no response is received from a sensor, the datalogger must wait for at least 16.67 milliseconds after the last stop bit of the command, but no longer than 87 milliseconds, and then issue a retry (without a break). (This period of 87 milliseconds includes the 16.67 milliseconds spent waiting for a response from the sensor.) If a correct response is not received after re-transmitting the command at least two more times, with at least one of those retries more than 100 milliseconds after the end of the break, the entire sequence (including the break and the retries) should be repeated at least two more times. At least one of the retries must be issued after 100 milliseconds after the falling edge of the break to ensure that the sensor has been given the full 100 milliseconds to wake up after the break. A retry is needed if one of the following three conditions exist:

1) no response from the sensor;
2) 8.33 milliseconds of marking on the data line, after receiving the start bit of the response;
3) an invalid response.

Invalid responses include responses in a incorrect format, parity errors, framing errors, or bus contentions. The datalogger must wait for the response to complete before sending a retry. Because the datalogger is the master of the SDI-12 bus, it is not required to issue a retry. The requirement is for a datalogger to support retries. If one of the above three conditions exists the datalogger will issue retries. Under extraordinary conditions, not as normal operating procedure, the datalogger has the option of not issuing retries.

## 7 SDI-12 data logger – Flow Chart

This flow chart shows the program flow within the datalogger


SDI-12 Data Recorder Flow Chart
start command/response sequence
outer_loop_ctrl=3
inner_loop_ctrl=3
has more than 87 msec elapsed since the last transmission of a byte from this sensor
yes
no
Set the retry timer to 112.5 msec
send a 12.5 msec break
inner_loop_ctrl=3
-set data line to marking for 8.33 msec
-transmit the address, the command, & !
-release the data line
look & wait
16.67 msec timeout
check the retry timer
timer not expired
timer expired
decrement inner_loop_ctrl
start bit received
look & wait
8.33 msec marking
inner_loop_ctrl=0
no
yes
<CR><LF> received
valid response?
no
yes
decrement outer_loop_ctrl
Start a timer (used to detect if more than 87 msec have elapsed since the end of this transmission, see above)
outer_loop_ctrl=0
no
yes
end, success
timeout
flag condition indicating no response
end, no response

## 8 USB/SDI-12 interface converter

For the test of commands and to get know of all responses it is useful to use a USB/SDI-12 converter.
The converter will be connected to the PC and the OTT RLS sensors.

A terminal program is used for the command input.

USB/SDI-12 converter with cable + driver CD

**Steps to sep-up the OTT RLS with the USB/SDI-12 converter**

1. Install driver for USB/Converter
2. Install Terminal program
3. Connect USB/SDI-12 converter to RLS and PC
4. Start terminal program
5. Type commands

## Step 1 Install driver

*OTT USB to SDI12 Link Driver Installation (Windows XP/Vista)*

1. Connect the USB to SDI12 Link to the computers USB port.
2. Windows will automatically detect the USB to SDI12 Link.
3. Windows will ask to look for a driver in the internet. Please choose no.
4. The „Add New Hardware Wizard“ dialog box appears and searches for new drivers for USB to SDI12 Link.
5. From the listed box, choose "Install software from a known source…".
6. Click on Next.
7. On the next Screen choose: do not search for driver…
8. Click on Next.
9. If not done already, insert the driver CD into the CD-ROM drive.
10. Click on Disc…
11. Search the path on the CD ROM with the driver
12. Click on OK
13. Windows copies the files to your system. Two drivers will be installed.
14. The dialog shows "Windows has finished installing the software that your new hardware device requires". Click "Finish".

*OTT USB to SDI12 Link Driver Installation (Windows 2000)*

1. Connect the USB to SDI12 Link to the computers USB port.
2. Windows will automatically detect the USB to SDI12 Link.
3. The „Add New Hardware Wizard“ dialog box appears and searches for new drivers for USB to SDI12 Link.
4. From the listed box, choose "Install software from a known source…".
5. Click on Next.
6. On the next Screen choose: choose other source…
7. Click on Next.
8. If not done already, insert the driver CD into the CD-ROM drive.
9. Click on Disc…
10. Search the path on the CD ROM with the driver
11. Click on OK
12. Windows copies the files to your system. Two drivers will be installed.
13. The dialog shows "Windows has finished installing the software that your new hardware device requires". Click "Finish".

*Checking installation*

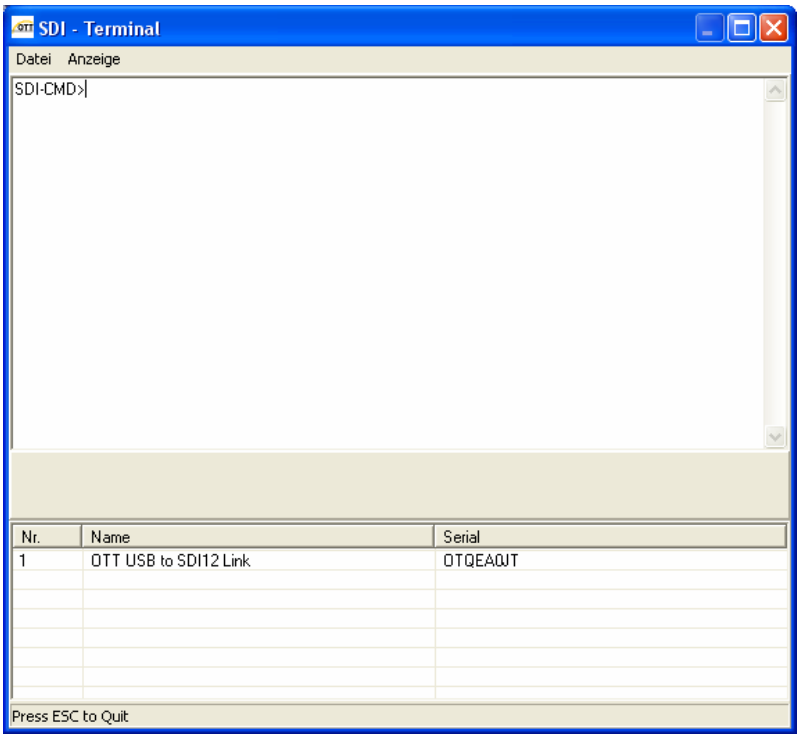
To verify the installation:

1. Open "Control panel".
2. Open "System" applet.
3. Click the "Device Manager" tab.
4. Click on the "+" sign next to "Ports (COM & LPT)".
5. The " USB to SDI12 Link (COMx)" is listed.
6. Double click on it, you can check the resources of the new COM port.

### Step 2 Install Terminal program

**Copy terminal program “SDITerminal.exe “  to PC**
**Connect USB/SDI-12 interface converter**

**Run program “SDITerminal.exe”**

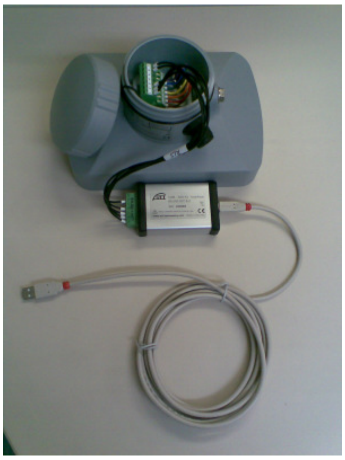


```
SDI-CMD>|
```

| Nr. | Name | Serial |
|---|---|---|
| 1 | OTT USB to SDI12 Link | OTQEA0JT |

Press ESC to Quit

### Step 3 Connect RLS to USB/SDI-12 converter



**Type in command:** **0V! <CR>** **(Verification if device is ready)**
**Response:**

### Step 4 Type in commands

**....**