SENSUS PRO

Developer's Guide

Published October 17, 2002

1.0 Introduction

This document is aimed at software developers who wish to communicate with ReefNet's SENSUS PRO dive data recorder. This device was designed specifically with third-party developers in mind: the communication protocol is extremely clean and simple, and can be implemented on nearly any platform or operating system.

ReefNet encourages all interested parties to read this document and produce unique new software solutions compatible with SENSUS PRO. Feel free to contact us with any questions or comments – we want to help you. When your product is ready, we'd be happy to announce it to all of our users, and will promote it on our web site.

We would like to see as many operating systems and platforms supported as possible!

NOTE:
This document only applies to communication with SENSUS PRO devices.
Communication with the original (silver) SENSUS model is done differently.
A separate Developer's Guide is available for SENSUS users.

1.1 Terminology

All references to the ***device*** refer to a SENSUS PRO recorder connected via the universal cradle to a ***host*** computer system. The ***host*** may be any of the following: PC, Mac, Palm handheld, Pocket PC, Unix workstation, etc.

Any host with a standard 9-pin RS232 serial port should be capable of communicating with the device.

2.0 Hardware Overview

This section describes the basic operating modes of the device's internal processor, as well as the physical interface between the device and the host.

2.1 Modes of Operation

SENSUS PRO is always in one of three basic operating modes. Depending on the mode, its response to outside activity will vary:

Table 1 -- Basic operating modes

Mode	Description
SLEEP	The normal operating mode. The processor is in deep hibernation. Once a second, the device wakes to advance the internal clock, check for the presence of a host, and check if the depth indicates that a dive has started.
DIVE	Same as SLEEP, except that after each sample interval, depth and temperature are written to memory. All host communication is suspended in this mode. When the dive is complete, the device returns to SLEEP mode.
WAIT	If host communication is detected during SLEEP, the device fully awakens, transmits its identity to the host, and waits for an instruction. If no instruction is received within 1 second, the device returns to SLEEP mode.

2.2 Interface

The device connects to any industry-standard 9-pin (DB9) RS-232 serial port. Some hosts do not have such a port, but can still be used with the device if an appropriate adapter is purchased:

- **PCs or Macs with USB only** require a USB-serial adapter such as the Keyspan High-Speed USB Serial Adapter (USA-19QW)
- **Handhelds with USB only** require a ***serial* cradle or cable**, or serial port add-on card (ask your PDA manufacturer)
- **Macs with MiniDIN8 serial port** require a MiniDIN8 to DB9 adapter (*contact ReefNet for details*)

The universal cradle does not contain any electronics. It is simply designed to expose the three serial port connections RX, TX, and GND to the device. The metal contacts on the device mate with those on the cradle to establish the necessary connections.

The contact layout is designed to prevent users from connecting the device backwards. The device has built-in protection against moderate static discharge, and will not be damaged by everyday handling. Nor does shorting between any of its contacts harm it.

For those interested: the cradle's PC/Mac and Palm ports differ as follows:

- The **PC/Mac port** is meant to connect to a **DTE** (*Data Terminal Equipment*) such as a PC, Mac, Unix workstation, or any other host with a male DB9 serial port.
- The **Palm port** is meant to connect to a **DCE** (*Data Communication Equipment*) such as a Palm handheld's cradle, or any other host with a female DB9 serial port.

The two cradle ports are wired together with the RX/TX lines swapped between them.

The software interface requires direct access to the serial port being used by the SENSUS PRO device. The only signals used are RX, TX, and GND. No flow control signals are implemented, and these should be disabled by the software interface to prevent erroneous transmissions.

3.0 Retrieving Data from a Device

This section defines the serial communication protocol used to send data to- and receive data from SENSUS PRO devices. Only rudimentary knowledge of serial communications and data processing is required.

3.1 Serial Port Settings

Communication with the device is done at **19.2 kbps, 8 data bits, No parity, 1 stop bit**.

NO flow control is implemented.

3.2 Byte Order, Data Types

All data transmitted by the device are Little Endian (Intel style), with the least significant byte(s) first. Three data types are used:

Table 2 -- Data Type Definitions

Type	Description
UInt8	8 bit unsigned integer
UInt16	16 bit unsigned integer
UInt32	32 bit unsigned integer

3.3 Error Detection

After transmitting each block of data, the device transmits a 16-bit CRC to enable your software to detect communication problems.

The CRC used is the standard CRC-CCITT style, with polynomial 0x1021 and initial value 0xFFFF.

If you are unfamiliar with CRC calculation, there are numerous online and offline resources to which you may refer. An excellent tutorial is Ross Williams' "*A Painless Guide to CRC Error Detection Algorithms*", which can be found at:

```
ftp.adelaide.edu.au/pub/rocksoft/crc_v3.txt
```

To verify the operation of your CRC routine (regardless of its source), you should compute the CRC of the following test string:

```
ReefNet     CRC = 0xEF03
```

3.4 Handshaking

Communicating with a SENSUS PRO device begins with the process of awakening it from its normal sleep mode. Once every second, the device briefly checks the condition of the incoming data line. If the line is idle, the device returns to sleep mode. If the line is active, the device remains awake and transmits a ***12-byte handshake packet*** to the host.

To properly wake a SENSUS PRO device, you must EITHER

- Assert a BREAK condition on the port
- Wait until data begin arriving from the device
- Immediately clear the BREAK condition on the port
- Read the 12-byte handshake packet

OR (if you cannot assert a BREAK condition)

- Transmit a byte (e.g. 0x00) to the device
- Check if any data are waiting in the receive buffer
- Repeat only until data begin arriving from the device
- Read the 12-byte handshake packet

Because the device only probes the incoming data line infrequently, it may take 1-2 seconds for the device to wake up. The handshake packet provides information about the product version, battery level, and other device parameters:

Table 3 -- Handshake data packet

Offset	Type	Description	Notes
0x00	UInt8	Product code	0x02 for SENSUS PRO
0x01	UInt8	Version code	Varies
0x02	UInt8	Battery voltage	0 to 255 = 0 to 5 V scale**
0x03	UInt8	Sample interval	In seconds
0x04	UInt16	Device ID	Imprinted serial number
0x06	UInt32	Current time	# seconds since power-up
0x0A	UInt16	16-bit CRC	For the preceding data

**The battery used is a long-life lithium cell. A full cell is about 3 V, and when drained is about 2.5-2.7 V. The discharge curve is NOT linear…you CANNOT compute a "% battery life remaining" using the above values!

After transmitting the handshake packet, the device waits for a *one-byte instruction code* from the host. If an instruction code is sent within 1 second, it is processed. Otherwise the device stops waiting and returns to sleep mode.

3.5 Downloading Dive Data

To download the contents of the device flash memory, you must

- Wake the device as outlined above
- Wait at least 10 ms after the last handshake byte is received
- **Send the instruction code 0xB4 (DUMP) to the device**
- Collect **56322 bytes** from the device

Transmission of the data takes just over 30 seconds, after which the device returns to sleep mode. The data consist of

56320 byte dive data block + 16-bit CRC

Use the CRC to detect any transmission errors that might have occurred.

The dive data block is ordered from oldest dive to newest dive. Each dive has the following structure:

Table 4 -- Structure of a single dive

Offset	Type	Description	Notes
0x0000	UInt32	Dive start	0x00000000
0x0004	UInt16	Sample interval	Interval (in seconds) between samples
0x0006	UInt32	Timestamp	Dive start time, in seconds since power-up
0x000A	UInt16	Sample #1	See Figure XXXXXX for structure
0x000C	UInt16	Sample #2	See Figure XXXXXX for structure
...	...	...	...
...	...	...	...
0x*NNNN*	UInt16	Dive end	0xFFFF

Figure 1 -- Sample point structure (UInt16)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Temperature (7 bits) in degrees F							Depth (9 bits) in feet of seawater, absolute (fswa)								

Note that there may be what appears to be 'garbage' before the first complete dive in the data block. It represents the tail of an old dive being pushed out of memory.

To tabulate the dives in the dive data block:

1) Start at the beginning of the block
2) Advance one byte at a time until you find a dive start flag (UInt32, 0x00000000)
3) Read the sample interval and timestamp
4) Read one sample at a time until you reach the dive end flag (UInt16, 0xFFFF)
5) **Repeat steps 2 - 4** until you reach the end of the dive data block

3.6 Programming the Sampling Interval

SENSUS PRO devices can be programmed to sample depth and temperature at any interval between 1 second and 127 seconds. Storage capacity is inversely proportional to sampling interval:

Table 5 -- Examples of sampling interval vs. capacity

Interval	Capacity
10 seconds	100 hours
1 second	10 hours
2 second	20 hours
127 seconds	1270 hours (53 DAYS!!)

To change the sampling interval of a device, you must

- Wake the device as outlined above
- Wait at least 10 ms after the last handshake byte is received
- **Send the instruction code 0xB5 (INTERVAL) to the device**
- Wait at least 10 ms
- **Send the new sampling interval as a single byte (e.g. 0x0A = 10 sec)**

The new sampling interval will immediately be stored, after which the device returns to sleep mode. You may verify that the interval was programmed correctly by subsequently waking the device and observing the sample interval in the handshake packet.

4.0 Frequently Asked Questions

Can my software damage the device by accident?

NO. As long as the device is connected to a standard serial port, nothing that your software sends to the port can cause damage. Unlike other dive computers, you can't accidentally reprogram or erase the firmware stored in the flash memory.

How do I figure out when a dive started?

The device contains a very accurate crystal clock that simply counts the number of seconds since the battery was installed and the processor powered-up.

When a dive starts, the internal clock value is stamped in the dive header. When you receive the handshake packet from the device, it includes the *current* clock value.

Hence you can compute the time that has elapsed since the dive occurred:

The dive occurred (handshake clock – dive header clock) seconds ago.

On your host machine, note the time at which the handshake was received. This establishes a reference between the real world time (host) and the device clock. Then simply count the current host date/time backward by the above number of seconds, and you have the true dive start date/time.

Though this may seem confusing, it is actually very simple, and quite clever. This method of timekeeping means that the internal clock in the device never needs to be 'set' to the proper time. It simply relies on your host clock. The catch? YOUR HOST CLOCK MUST BE CORRECT.

Are there other programmable features?

Yes and no. The Device ID is programmed after assembly, but the applicable instruction code only works once. After the Device ID has been set, the device disregards any future attempt to change it.

Other instructions exist, but are for internal testing purposes only. They are not of any use to the public. We don't suggest you search for them – you will not find anything interesting.

I know nothing about CRCs. Must I compute and verify the CRC?

You really should. The dive data block is over 50 kb in size – it is entirely possible that a byte or two of that data is incorrect due to a bad electrical contact or other problem. Hoping that the data are valid is not a good strategy.

5.0 DISCLAIMER

ReefNet cannot be held responsible for damage to a connected device caused by software/hardware other than ReefNet's SENSUS PRO MANAGER + cradle. Nor can ReefNet be held responsible for damage caused by tampering with either a SENSUS PRO device or its cradle.

6.0 Contacting Us

For technical questions, please contact:

Kris Wilk
ReefNet Inc.
3610 Walnut Grove
Mississauga, Ontario L5L2W8
Canada

Tel:	1-888-819-REEF or
	(905)608-9373
Fax:	(905)820-1927

wilk@reefnet.on.ca
www.reefnet.on.ca