Datex-Ohmeda

Datex-Ohmeda Division
Instrumentarium Corp.

AS/3, CS/3 Monitoring System Main Software
S/5 Monitor System Main Software
Computer Interface

Author:	Raimo Piskunen
File name:	Compint_99_S5.doc
Revision:	1.0
Date:	2001-01-17
Status:	

Table of contents

1 General

The AS/3, CS/3 and S/5 Monitors have a high-speed asynchronous serial interface for data acquisition purposes. The interface provides access to the physiological database of the monitor, which contains the most important measurement values and associated status information. A limited set of waveform data is available starting from software versions S-STD94 and S-ARK94.

The interface is in no way compatible with older Datex Whiteline monitors. Compatibility with future versions of Datex-Ohmeda (D-O) monitors is, however, the main objective of the interface. Data structures and constants presented here will be compatible with later versions.

Internal data representation format follows the D-O Record Interface definition. Relevant parts of the D-O Record Interface are published in this document.

It is recommended that the reader has basic knowledge about the C programming language. The notation of the C language has been used in many definitions.

2 Use of the computer interface

2.1 Line parameters

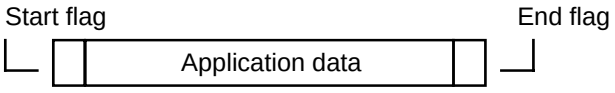
The interface uses the following serial communication line parameters:

19 200 bit/s transmission rate,
8 data bits with
even parity and
1 stop bit.
CTS/RTS hardware handshaking is used for communication control.

The line parameters cannot be changed.

2.2 Frame structure

All data from the monitor, and to the monitor, is transferred using flag-delimited frames. Each data frame starts and ends with a flag character. All application data is always located between these flags.



2.2.1 Data transparency

The value of the start and end flags is always **0x7E**. As the application data may contain an arbitrary number of bytes with the same value, following algorithm is used to detect the start and end of a frame correctly:

A control character (**0x7D**) is used to indicate the start of a control sequence. At the transmitting end, each application data byte with value 0x7E (flag) or 0x7D (control character) is replaced with a control character and the original byte with the 5th (of bits 0-7) bit cleared. Therefore, the following conversions are possible:

0x7E -> 0x7D, 0x5E
0x7D -> 0x7D, 0x5D

This method guarantees that there are no flag characters in the outgoing application data stream.

As a control character is received, it is not interpreted to be a part of the application data. The 5th bit of the **next** character must be set to restore the original value of the character. Therefore, the following conversions are possible:

0x7D, 0x5E -> 0x7E
0x7D, 0x5D -> 0x7D

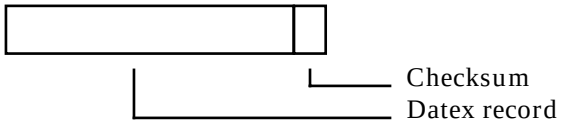
This conversion method is similar to that represented in the ISO3309 standard.

NOTE: Software version S-STD93 accepts the end flag of a frame to act as the start flag of the next frame. This undocumented feature is not present in later versions.

2.2.2 Application data structure

As the start and end flags are removed and necessary conversions done, the application data can be further processed.

The application data consists of a D-O record and a checksum.



A D-O record has variable length. Its internal structure is defined in following chapters.

NOTE: When doing the conversion for the application data, the checksum byte must be included in the conversion.

2.2.3 Checksum

When interfacing through the serial interface, each D-O record is followed by a checksum byte. The checksum is calculated by summing all bytes in the D-O Record using 8 bit unsigned arithmetic.

2.3 Connector

The physical computer interface is a PC/AT style 9-pin male D-connector at the rear panel of AS/3 or CS/3 monitor. Following pins are used:

Pin	Usage
2	Rx Data
3	Tx Data
5	GND
7	RTS
8	CTS

3 Data structures

3.1 General

All data is in binary '80x86' format. Lower byte of a 16-bit value is in lower address and higher byte in higher address. Lower word of a 32-bit value is in lower address and the higher word in higher address.

Sizes of integer and short integer are 32 bits and 16 bits, respectively.

Some commonly used data types are defined below:

```
#define  byte      unsigned char
#define  word      unsigned short
#define  dword     unsigned long
```

IMPORTANT !! Fields within structures are aligned to **1-byte boundaries.**

NOTE: When transmitting data to the D-O Monitor Computer Interface, all reserved fields must be set to 0 to ensure compatibility with future versions of monitors.

3.2 Structure of D-O record

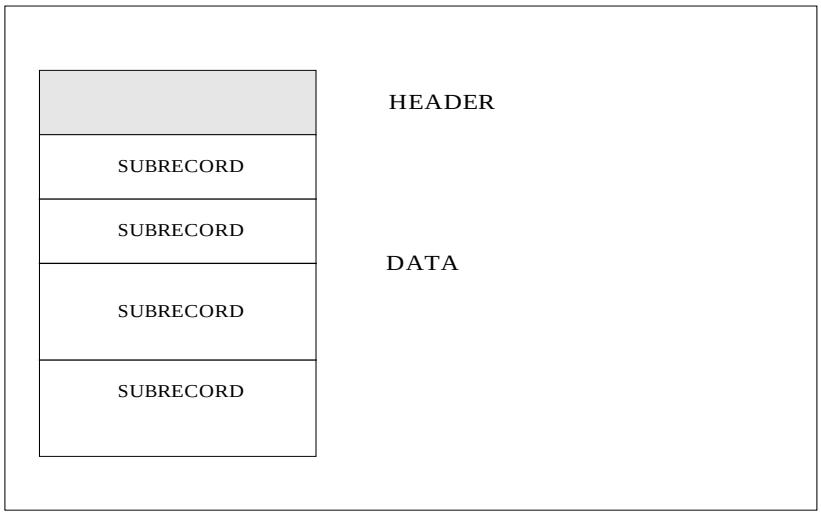
At the highest level, a D-O record consists of a D-O header and variable amount of data:

```
struct D-O_record
{
        struct D-O_hdr          hdr;
        union
        {
                union wf_srcrds         wf_rcrd;
                union ph_srcrds         ph_rcrd;
                byte                    data[DRI_MAX_RECSIZE];  (max 1450 bytes)
        } rcrd;

};
```

As the size of the header is 40 bytes, the maximum size of a D-O record is 1490 bytes.

The data area is divided into smaller blocks which are of variable size. These blocks are called **subrecords**. There can be up to 8 subrecords in one D-O record.
A subrecord contains a logical entity of D-O monitor application data.



3.2.1 Record header

The structure of the D-O header is

```
                struct D-O_hdr
                {
                        short           r_len;
                        byte            r_nbr;
                        byte            dri_level;
                        word            plug_id;
```

```
                dword       r_time;
                byte        n_subnet:
                byte        res;
                word        dest_plug_id;
                word        r_maintype;
                struct sr_desc   sr_desc[8];
        };
```

The contents of the header fields are:

r_len equals the total length of the record, including the D-O header.

r_nbr record number

dri_level indicates the D-O Record Interface level the monitor supports (see chapter “Monitor versions”). The monitor ignores this field when receiving data requests.

plug_id plug identifier number

r_time is the time when the record was transmitted. Time is defined as the number of seconds since 1.1.1970.

NOTE: Some compilers and libraries use 1.1.1900 as the start moment of time.

n_subnet subnet identifier

dest_plug_id destination address used by rs232

r_maintype is the main type of the record. Subrecord types are subtypes of the main type. The following main types are used:

DRI_MT_PHDB	0	for physiological data and related transmission requests.
DRI_MT_WAVE	1	for waveform data and related transmission requests.

sr_desc is an array which describes the data in the subrecords.

```
        struct sr_desc
        {
                short sr_offset;
                byte sr_type;
        };
```

sr_type contains the subrecord type. Valid subrecord types depend on the record main type. As the number of subrecords is not fixed, the value 0xFF as subrecord type indicates that there are no more subrecords in the record.

sr_offset is a relative pointer to the subrecord. The origin, from where the offsets are calculated, is the start of the data area. Thus, offset to the first subrecord is always 0.

res field is reserved for future extensions. When transmitting data to the monitor these fields must be zeroed.

3.3 Monitor versions

The field **r_dri_level** in the packet header indicates the D-O Record Interface the monitor supports. The interface level indicates which commands the D-E monitor accepts and which fields in structures are used and which are not. In practice the interface level goes hand in hand with the AS/3 and CS/3 software version as follows:

Level	Software version
0x00	S-STD93
0x01	S-STD94, S-ARK94
0x02	S-STD95, S-ARK95, S-STD96, S-ARK96
0x03	S-ANE97, S-ARK97, S-ICU97
0x04	S-ANE98, S-ARK98, S-ICU98
0x05	S-ANE99, S-ARK99, S-ICU99, S/L-00xxx..00

Differences between interface levels are included in this document.

NOTE: As there are some differences between the interface levels, you may need the interface level information before starting the actual data acquisition. The easiest way to do this is to send a single transmission request which is supported by all interface levels and determine the actual level from the r_dri_level field in the header.

4 D-O monitor physiological data structure

4.1 Subrecord types

Following values are used to describe the physiological database subrecords and related transmission request subrecords:

Value	Usage	Version info
0	Physiological data transmission requests	
1	Current (displayed) values of the physiological database	
2	10 s trended values of the phys. database	
3	60 s trended values of the phys. database	Interface level 1 ->
4	Auxiliary physiological information	
5	Trend data download control	
6	Trend download requests	

In addition to subrecord types there is another dimension of classification for physiological data records of type 1,2 or 3: the physiological subrecord class.

193 All the physiological subrecords of types 1,2 and 3 have the same length and some common
194 fields, like the time stamp of the record. However, the data contents of the record vary
195 depending on the record class.

196 The physiological subrecord classes are:

Name	Usage	Version info
Basic	Basic physiological parameters: ECG, blood pressures, temperatures, basic gas measurement data, SpO2, SvO2, ...	
Ext1	Arrhythmia analysis related data	Interface level 3 ->
Ext2	NMT related parameters	Interface level 3 ->
Ext3	Gas exchange and spirometry related parameters.	Interface level 3 ->

197

198 ## 4.2 Subrecord structures

199 ### 4.2.1 Transmission request (subtype 0)

200 The structure of a physiological data **transmission request subrecord** is

```
201 struct dri_phdb_req
202 {
203     byte        phdb_rcrd_type;
204     short       tx_interval;
205     long        phdb_class_bf;
206     short       reserved;
207 };
```

208 **phdb_rcrd_type** specifies the subrecord the interfacing device wishes to receive.
209 Valid values are listed in table in the previous chapter.

tx_interval specifies the transmission interval in seconds. Following values can be used:

Value	Use	Version info
-1	to request a single transmission of a subrecord	
Any positive value together with subrecord type DRI_PH_10S_TREND	to start automatic transmission of that subrecord type using 10 s interval. No further transmission requests are needed after this.	Interface level: 1 ->
Any positive value together with subrecord type DRI_PH_60S_TREND	to start automatic transmission of that subrecord type using 60 s interval. No further transmission requests are needed after this.	Interface level: 1 ->
A positive value greater or equal to 5 together with subrecord type DRI_PH_DISPL or DRI_PH_AUX_INFO	to start automatic transmission of that subrecord type using the specified transmission interval. No further transmission requests are needed after this.	Interface level: 1 ->
0	to cancel the automatic transmission of the specified subrecord	Interface level: 1 ->

NOTE: If -1 us used for transmission interval, the minimum interval for requesting data from the AS/3 or CS/3 monitors is 5 seconds. If the monitors receive two requests within 5 seconds, the latter request is ignored.

phdb_class_bf is a bit field specifying which classes of physiological data records should be sent to the requesting application.

DRI_PHDBCL_REQ_BASIC_MASK	0x0000
DRI_PHDBCL_DENY_BASIC_MASK	0x0001
DRI_PHDBCL_REQ_EXT1_MASK	0x0002
DRI_PHDBCL_REQ_EXT2_MASK	0x0004
DRI_PHDBCL_REQ_EXT3_MASK	0x0008

DRI_PHDBCL_BASIC	0
DRI_PHDBCL_EXT1	1
DRI_PHDBCL_EXT2	2
DRI_PHDBCL_EXT3	3

The field **reserved** must be zeroed.

4.2.2 Physiological database (subrecord types 1, 2, 3)

There are three subrecord types for the actual measurement data:
for the displayed values,
for the 10 s trended values and
for the 60 s trended values.

These subrecords have identical internal structure.

```
struct dri_phdb
{
        dword                   time;
        union
        {
        ////////////////////////////////////
        struct basic_phdb       basic;
        ////////////////////////////////////
        struct ext1_phdb        ext1;
        ////////////////////////////////////
        struct ext2_phdb        ext2;
        ////////////////////////////////////
        struct ext3_phdb        ext3;
        ////////////////////////////////////
        }               physdata;
        byte                    marker;
        byte                    pdm_ctrl_bf;
        word                    cl_drilvl_subt;
}
```

time field in physiological data record contains the time stamp of the record in unixtime, i.e., the number of seconds since 00:00:00 , 1.1.1970.

marker contains the number of latest entered mark.

Pdm_ctrl_bf field contains control information for patient data management functions, used internally by the monitor.

cl_drilvl_subt, the last word of the subrecord, contains

- the physiological data record class
- the current D-O Record Interface level and
- the subrecord type. The contents of the union field physdata depends on the subrecord class.

The subrecord type (DRI_PH_DISPL, DRI_PH_10S_TREND or DRI_PH_60S_TREND) is stored in bits 0...3 of the lower byte.

Bits 4...7 of the lower byte contain the D-O Record Interface level.

Bits 0...5 of the higher byte contain the record class, which is

0 for the class “basic”,

267 1 for the class “ext1”,

268 2 for the class “ext2” and

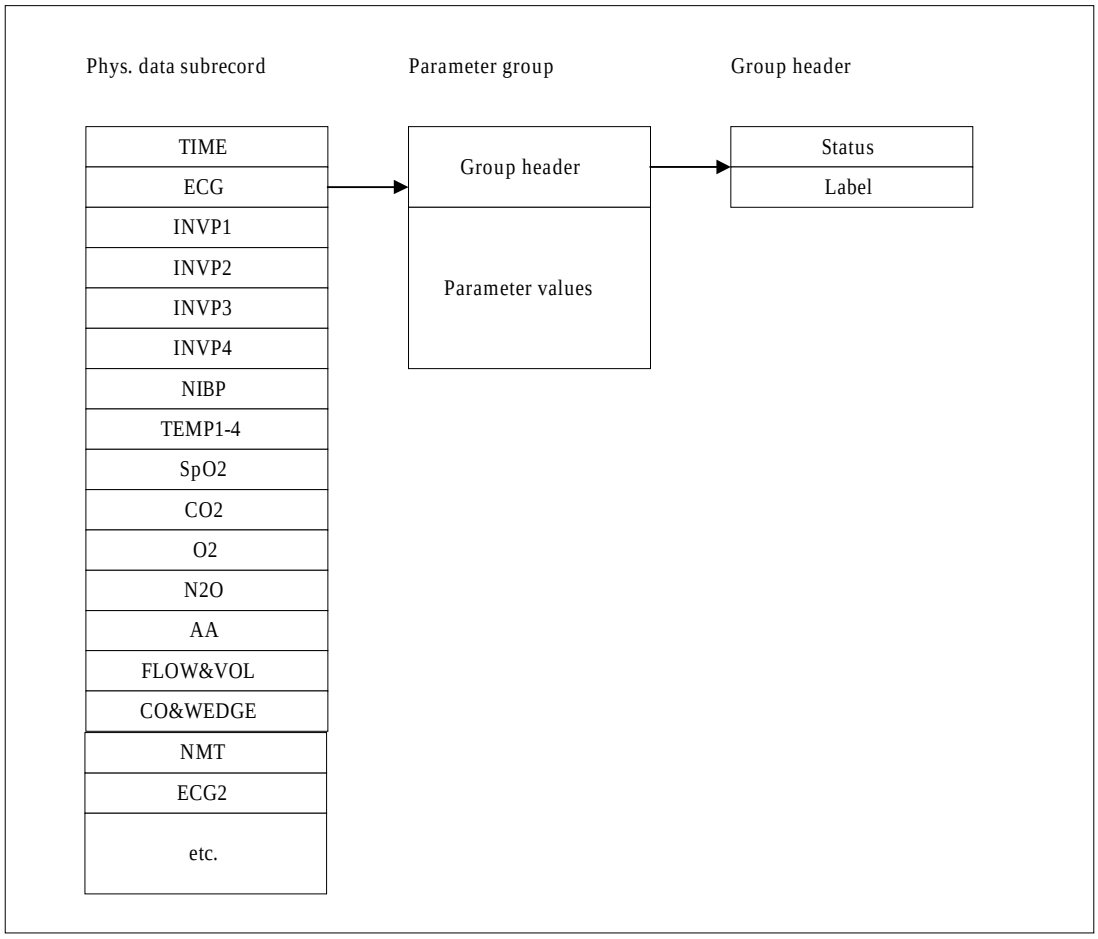
269 3 for the class “ext3”.

270 Rest of the values (4-31) are reserved for future extensions.

271 NOTE: Definition of struct dri_phdb looks different from the struct phdb definition of the
272 previous versions. However, struct dri_phdb is binary compatible with the old struct phdb
273 and the existing applications have no problems in accessing the data of 97-version monitors.

274 4.2.3 Structure of measurement data

275 The data areas (union physdata) of the physiological database subrecords have been divided
276 into smaller parameter specific groups. A parameter group consists of a group header and 1-8
277 measurement values.



278

The group header contains parameter or measurement specific information.

```
struct group_hdr
{
        union phdb_status   status;
        word                label;

};
```

label is a parameter specific field, and its contents are discussed together with the contents of the measurement data.

Status holds up to 32 common and parameter specific bits. Currently two bits are common to all parameters:

Bit	Usage
0	Measurement module existence: TRUE, if the module exists.
1	Measurement activity: TRUE, if the measurement is active.
2-31	Reserved for future and parameter specific use. See parameter definitions for details.

All measurement data is represented as **signed 16-bit** values. Some control information is embedded into the measurement data by assigning special meaning to certain values. As values with special meaning start from -32001 downwards, the smallest valid value is always -32000.

The most common special values are

```
        DATA_INVALID            -32767
        DATA_NOT_UPDATED        -32766

        DATA_UNDER_RANGE        -32764
        DATA_OVER_RANGE         -32763
```

4.2.4 Basic class (basic_phdb)

The data contents for subrecord class “basic” is specified below:

```
struct basic_phdb
{
        struct ecg_group        ecg;
        struct p_group          p1234[4];
        struct nibp_group       nibp;
        struct t_group          t[4];
        struct spo2_pl_group    spo2;
        struct co2_group        co2;
        struct o2_group         o2;
        struct n2o_group        n2o;
        struct aa_group         aa;
        struct flow_vol_group   flow_vol;
        struct co_wedge_group   co_wedge;
        struct nmt_group        nmt;
        struct ecg_extra_group  ecg_extra;
        struct svo2_group       svo2;
        struct p_group          p56[2];
        byte                    reserved[2];
    struct group_hdr
{
    union phdb_status   status;
    word                                    label;
};
}
```

4.2.4.1 ECG

```
struct ecg_group
{
        struct group_hdr        hdr;
        short                   hr;
        short                   st1;
        short                   st2;
        short                   st3;
        short                   imp_rr;
};
```

Status field

Bit	Usage	Version info
2	Asystole.	Interface level 1 ->
3 .. 6	These bits indicate the heart rate source. Possible values are listed below.	Interface level 1 ->
7	Noise	Interface level 3 ->
8	Artifact	Interface level 3 ->
9	Learning	Interface level 3 ->
10	Pacer on	Interface level 3 ->
11	Channel 1 off	Interface level 3 ->
12	Channel 2 off	Interface level 3 ->
13	Channel 3 off	Interface level 3 ->
14 .. 31	Reserved	

Heart rate sources

Value	Heart rate source
0	Not selected
1	ECG
2	Invasive pressure channel 1
3	Invasive pressure channel 2
4	Invasive pressure channel 3
5	Invasive pressure channel 4
6	SpO2
7	Invasive pressure channel 5
8	Invasive pressure channel 6
9 .. 15	Reserved

Label field

This field contains the lead configuration of each ECG channel. Four bits are used to describe each channel as follows:

Bits	ECG channel
0 .. 3	3
4 .. 7	2
8 .. 11	1
12 .. 15	Reserved

The value of the four bits used to describe the lead configuration are interpreted as follows:

Value	Lead
0	NOT_SELECTED
1	ECG_I
2	ECG_II
3	ECG_III
4	ECG_AVR
5	ECG_AVL
6	ECG_AVF
7	ECG_V
8 .. 15	RESERVED

Data fields

Field	Usage	Unit	Notes/Version info
hr	Heart rate	1/min	The heart rate value is not necessarily from the ECG measurement but is based on the monitor's heart rate source selection. See status bits for details.
st1 st2 st3	St-level	1/100 mm	St values are calculated from currently selected user leads Ecg1-3. Labels in hdr indicates selected label. Only if selected one of V1-V6 then label is set to V. (ext1 group includes 12-lead st values. See 5.2.5.2.)
imp_rr	Respiration rate	1/min	Based on measurement of ECG impedance. Interface level 2->.

4.2.4.2 Invasive pressures

```
struct p_group
{
        struct group_hdr        hdr;
        short                   sys;
        short                   dia;
        short                   mean;
        short                   hr;
};
```

Status field

Bit	Usage	Version info
2	Zeroing	Interface level 3 ->

Label field

The field contains the invasive pressure label, which is one of the following:

Value	Label
0	NOT DEFINED
1	ART
2	CVP
3	PA
4	RAP
5	RVP
6	LAP
7	ICP
8	ABP
9	P1
10	P2
11	P3
12	P4
13	P5
14	P6
15	Reserved

Data fields

Field	Usage	Unit	Notes/Version info
Sys dia mean	Invasive pressure	1/100 mmHg	
hr	Pulse rate	1/min	

4.2.4.3 Non-invasive blood pressure

```
struct nibp_group
{
        struct group_hdr        hdr;
        short                   sys;
        short                   dia;
        short                   mean;
        short                   hr;
};
```

NOTE: NIBP 10 s trended values are always equal to the displayed values.

Status field

No parameter specic bits are used.

Label field

Bit	Usage	Unit	Notes/Version info
0..2	cuff type	1	See cuff types. Interface level 1->
3	AUTO mode selected	1	Interface level 1->
4	STAT mode selected	1	Interface level 1->
5	measuring	1	Interface level 1->
6	STASIS ON	1	Interface level 1->
7	calibrating	1	Interface level 1->
8	data is older than 60 s	1	
9 .. 15	not used		

Cuff types

Value	Cuff type
0	NOT DEFINED
1	Infant
2	Reserved
3	Adult
4..7	Reserved

Data fields

Field	Usage	Unit	Notes/Version info
sys dia mean	pressure	1/100 mmHg	
hr	pulse rate	1/min	

4.2.4.4 Temperatures

```
struct t_group
{
        struct group_hdr        hdr;
        short                   temp;
};
```

Status field

No parameter specic bits are used.

Label field

This field contains the temperature label, which is one of the following:

Value	Label
0	NOT USED
1	ESO
2	NASO
3	TYMP
4	RECT
5	BLAD
6	AXIL
7	SKIN
8	AIRW
9	ROOM
10	MYO
11	T1
12	T2
13	T3
14	T4
15	CORE
16	SURF
17 .. 31	Reserved

Data fields

Field	Usage	Unit	Notes/Version info
temp	temperature	1/100 °C	

4.2.4.5 SpO2

```
struct SpO2_pl_group
{
        struct group_hdr        hdr;
        short                   SpO2;
        short                   pr;
        short                   ir_amp;
        short                   SvO2;
};
```

Status field

No parameter specic bits are used.

Label field

Bits	Usage	Version info
0 .. 1	2 bit value containing the SO2 label.	Interface level 1 ->
2 .. 15	Reserved	

SO2 labels

Value	Label
0	SO2
1	SaO2
2	SvO2
3	Not used

Data fields

Field	Usage	Unit	Notes/Version info
SpO2	oxygenation percentage	1/100 %	
pr	pulse rate	1 /min	
ir_amp	modulation	%	Plethysmograph amplitude
SO2	SO2, SvO2 or SaO2 value as specified by the label	1/100%	Interface level 1->

4.2.4.6 CO2

```
struct co2_group
{
        struct group_hdr        hdr;
        short                   et;
        short                   fi;
        short                   rr;
        short                   amb_press;
};
```

Status field

Bits	Usage	Version info
2	Apnea	Interface level 1->
3	Calibrating sensor	Interface level 1->
4	Zeroing sensor	Interface level 1->
5	Occlusion	Interface level 1->
6	Air leak	Interface level 1->
7 .. 31	Reserved	

Label field

Bits	Usage	Version info
0 .. 2	These bits indicate the respiration rate source. See Respiration rate sources.	Interface level 1->
3 .. 15	Reserved	

Respiration rate sources

Value	RR source
0	Not selected
1	CO2
2	ECG, Impedance Resp.

Data fields

Field	Usage	Unit	Notes
et	Expiratory concentration	1/100%	
fi	Inspiratory concentration	1/100%	
rr	Respiration rate	1/min	Based on measurement indicated in the label field.
amb_press	Ambient pressure	1/10 mmHg	

4.2.4.7 O2

```
struct o2_group
{
        struct group_hdr        hdr;
        short                   et;
        short                   fi;
};
```

Status field

Bits	Usage	Version info
2	Calibrating	Interface level 3->
3	Measurement off	Interface level 3->
4 .. 31	Reserved	

Label field

Not used.

Data fields

Field	Usage	Unit	Notes/Version info
et	Expiratory concentration	1/100%	
fi	Inspiratory concentration	1/100%	

4.2.4.8 N_2O

```
struct n2o_group
{
        struct group_hdr        hdr;
        short                   et;
        short                   fi;
};
```

Status field

Bits	Usage	Version info
2	Calibrating	Interface level 3->
3	Measurement off	Interface level 3->
4 .. 31	Reserved	

Label field

Not used.

Data fields

Field	Usage	Unit	Notes/Version info
et	Expiratory concentration	1/100%	
fi	Inspiratory concentration	1/100%	

4.2.4.9 Anesthesia agents

```
struct aa_group
{
        struct group_hdr        hdr;
        short                   et;
        short                   fi;
        short                   mac_sum;
};
```

Status field

Bits	Usage	Version info
2	Calibrating	Interface level 3->
3	Measurement off	Interface level 3->
4 .. 31	Reserved	

Label field

This field contains the anaesthesia agent label, which is one of the following:

Label	Agent
0	RESERVED (Unknown)
1	NONE
2	HAL
3	ENF
4	ISO
5	DES
6	SEV
7 .. 15	Reserved

Data fields

Field	Usage	Unit	Notes/Version info
Et	Expiratory concentration	1/100%	
Fi	Inspiratory concentration	1/100%	
mac_sum		1/100%	

4.2.4.10 Flow & volume measurement

```
struct flow_vol_group
{
    struct group_hdr    hdr;
    short               rr;
    short               ppeak;
    short               peep;
    short               pplat;
    short               tv_insp;
    short               tv_exp;
    short               compliance;
    short               mv_exp;
};
```

Status field

Bits	Usage	Version info
2	Disconnection	Interface level 1->
3	Calibrating	Interface level 1->
4	Zeroing	Interface level 1->
5	Obstruction.	Interface level 1->
6	Leak	Interface level 1->
7	Measurement off	Interface level 3->
8 .. 31	Reserved	

Label field

Not used.

Data fields

Field	Usage	Unit	Notes/Version info
rr	Respiration rate	1/min	Based on spirometry
ppeak	Peak pressure	1/100 cmH_2O	
peep	Positive end exp. pressure	1/100 cmH_2O	
pplat	Plateau pressure	1/100 cmH_2O	
tv_insp	Inspiratory tidal volume	1/10 ml	
tv_exp	Expiratory tidal volume	1/10 ml	
compliance	compliance	1/100 ml / cmH_2O	
mv_exp	Expiratory minute volume	1/100 l/min	

4.2.4.11 Cardiac output & wedge pressure

```
struct co_wedge_group
{
        struct group_hdr        hdr;
        short                   co;
        short                   blood_temp;
        short                   ref;
        short                   pcwp;
};
```

Status field

No parameter specic bits are used.

NOTE: Bits in status field reflect the status of the Cardiac Output module.

Label field

Bits	Usage
0	Age of CO reading is > 60 s
1	Age of PCWP reading is > 60 s
2 .. 15	Reserved

Data fields

Field	Usage	Unit	Notes/Version info
co	Cardiac output	ml/min	
blood_temp	blood temperature	1/100 °C	
ref	right heart ejection fraction	%	Interface level 2->
pcwp	wedge pressure	1/100 mmHg	

4.2.4.12 NMT

NOTE: This group is available starting from Interface level 2.

```
struct nmt_group
{
        struct group_hdr        hdr;
        short                   t1;
        short                   tratio;
        short                   ptc;
};
```

Status field

Bits	Usage
2..3	Stimulus mode: 0 : Train Of Four (TOF mode) 1 : Double Burst (DB mode) 2: Single Twitch (ST mode) 3: Post-tetanic count 4: Tetanic 5: Regional block
4..5	0: not used 1: 100 us 2: 200 us 3: 300 us
6	Supramax current found
7	Calibrated
8...31	Reserved

Label field

Not used

520 **Data fields**

Field	Usage	Unit	Notes/Version info
t1		1/10 %	Interface level 2->
tratio	t4/t1 in TOF mode t2/t1 in DB mode	1/10 %	Interface level 2->
ptc	split into a bit field, see below		Interface level 2->

521

522 **bits in ptc field**

Bits	Usage
0...4	Post tetanic count, max. value 21. Has value 31 if count not available.
5...8	TOF count (0...4) in TOF mode DB count (0...2) in DB mode ST count (0...1) in ST mode
9...15	Stimulus current, mA

523

524 ### 4.2.4.13 ECG extra group

525 NOTE: This group is available starting from Interface level 2.

526 struct ecg_extra_group
527 {
528 short hr_ecg;
529 short hr_max;
530 short hr_min;
531 };

532 NOTE: This group has no header. Status information is the same as in *ecg_group*.

Field	Usage	Unit	Notes/Version info
hr_ecg	heart rate as derived from the ecg signal	1	
hr_max	maximum heart rate	1	
hr_min	minimum heart rate	1	

533

534 #### 4.2.4.14 SvO2 group

535 NOTE: This group is available starting from Interface level 3.

536 ```
struct svo2_group
537 {
538   struct group_hdr   hdr;
539   short   svo2;
540 };
```

541 **Status field**

542 No parameter specific bits are used

543 **Label field**

544 Not used

545 **Data fields**

| Field | Usage | Unit | Notes/Version info |
|---|---|---|---|
| svo2 | SvO2 | 1/100% | |

546

547 ### 4.2.5 Ext 1 class (ext1_phdb)

548 The data contents for subrecord class “ext1” is specified in this chapter.

549 NOTE: Subrecord class “ext1” is available starting from interface level 3.

550 ```
struct ext1_phdb
551 {
552     struct arrh_ecg_group   ecg;
553     struct ecg_12_group     ecg12;
554     byte                    reserved[192];
555 };
```

4.2.5.1 ECG_12

```
struct ecg_12_group
{
        struct group_hdr        hdr;
        short                   stI;
        short                   stII;
        short                   stIII;
        short                   stAVL;
        short                   stAVR;
        short                   stAVF;
        short                   stV1;
        short                   stV2;
        short                   stV3;
        short                   stV4;
        short                   stV5;
        short                   stV6;
}
```

Status field

Bit	Usage	Version info
0..31	Reserved	

Label field

Bits	ECG channel
0..3	3
4..7	2
8..11	1
12..15	Reserved

577 **Data fields**

Field	Usage	Unit	Notes/Version info
StI StII StIII StAVL StAVR StAVF StV1 StV2 StV3 StV4 StV5 StV6	St-level	1/100 mm	

578

579 ### 4.2.6 Ext 2 class (ext2_phdb)

580 The data contents for subrecord class “ext2” is specified in this chapter.

581 NOTE: Subrecord class “ext2” is available starting from interface level 3.

```
582 struct ext2_phdb
583 {
584         struct nmt2_group       nmt2;
585         struct eeg_group        eeg;
586         byte                    reserved[174];
587 };
```

588 #### 4.2.6.1 NMT2 group

```
589 struct nmt2_group
590 {
591 struct group_hdr        hdr;
592 short                   count;
593 short                   nmt_t1;
594 short                   nmt_t2;
595 short                   nmt_t3;
596 short                   nmt_t4;
597 short                   nmt_resv1;
598 short                   nmt_resv2;
599 short                   nmt_resv3;
600 short                   nmt_resv4;
601 };
```

Status field

No parameter specific bits are used.

Label field

Not used.

Data fields

Field	Usage	Unit	Notes/Version info
nmt_t1	t1 absolute value		
nmt_t2	t2 absolute value		
nmt_t3	t3 absolute value		
nmt_t4	t4 absolute value		
nmt_resv1 .. nmt_resv4	Future extensions		

4.2.6.2 EEG group

```
struct eeg_group
{
        struct group_hdr        hdr;
        short                   femg;
        struct eeg_channel      eeg1;
        struct eeg_channel      eeg2;
        struct eeg_channel      eeg3;
        struct eeg_channel      eeg4;
};
struct eeg_channel
{
        short       ampl;
        short       sef;
        short       mf;
        short       delta_proc;
        short       theta_proc;
        short       alpha_proc;
        short       beta_proc;
        short       bsr;
 };
```

629 **Status field**

Bit	Usage	Version info
2	Measurement on	
3...6	Montage (in use: 0...7)	
7	Headbox off	
8	SSEP cable off	
9	Channel 1 leads off	
10	Channel 2 leads off	
11	Channel 3 leads off	
12	Channel 4 leads off	
13	Channel 1 artefact	
14	Channel 2 artefact	
15	Channel 3 artefact	
16	Channel 4 artefact	
17	Channel 1 noise	
18	Channel 2 noise	
19	Channel 3 noise	
20	Channel 4 noise	
21	EP selection (AEP = 0, SSEP = 1)	
22	Measurement type (referential = 0, bipolar = 1)	
23...31	Reserved	

630

631 **Label field**

632 Not used.

Data fields

Field	Usage	Unit	Notes/Version info
femg	Frontal electro-myography	1/10 uV	
ampl	RMS amplitude	1/10 uV	
sef	Spectral edge frequency	1/10 Hz	
mf	Median frequency	1/10 Hz	
delta_proc	Relative power spectral content in delta band	%	
theta_proc	Relative power spectral content in theta band	%	
alpha_proc	Relative power spectral content in alpha band	%	
beta_proc	Relative power spectral content in beta band	%	
bsr	Burst suppression ratio	%	

4.2.7 Ext 3 class (ext3_phdb)

The data contents for subrecord class “ext3” is specified in this chapter.

NOTE: Subrecord class “ext3” is available starting from interface level 3.

```
struct ext3_phdb
{
        struct gasex_group          gasex;
        struct flow_vol_group2      flow_vol2;
        struct bal_gas_group        bal;
        struct tono_group           tono;
        byte                        reserved[178];
};
```

4.2.7.1 Gas exchange measurements

```
struct gasex_group
{
    struct group_hdr    hdr;
    short               vo2;
    short               vco2;
    short               ee;
    short               rq;

};
```

Status field

No parameter specific bits are used.

Label field

Not used.

Data fields

Field	Usage	Unit	Notes/Version info
vo2	Oxygen consumption	0,1 ml/ min	
vco2	Carbon dioxide consumption	0,1 ml/ min	
ee	Energy expenditure	1 kcal/ 24h	
rq	Respiratory quotient		

4.2.7.2 Flow & volume group 2

```
struct flow_vol_group2
{
    struct group_hdr    hdr;
    short               ipeep;
    short               pmean;
    short               raw;
    short               mv_insp;
    short               epeep;
    short               mv_spont;
    short               ie_ratio;
    short               insp_time;
    short               exp_time;
    short               static_compliance;
    short               static_pplat;
    short               static_peepe;
    short               static_peepi;
    short               reserved(7);
};
```

Status field

No parameter specific bits are used.

Label field

Not used.

Data fields

Field	Usage	Unit	Notes/Version info
Ipeep	Intrinsic PEEP	0,01 cmH2O	
Pmean	Mean pressure	0,01 cmH2O	
Raw	Airway resistance	0,01 cmH2O	
mv_insp	Inspired minute volume	0,01 L/min	
Epeep	Extrinsic PEEP	0,01 cmH2O	
mv_spont	Spontaneous expired minute volume	0,01 L/min	
Ie_ratio			
Insp_time			
Exp_time			
static_compliance			
static_pplat			
static_peepe			
static_peepi			

4.2.7.3 Balance gas group

```
struct bal_gas_group
{
    struct group_hdr    hdr;
    short               et;
    short               fi;
};
```

Status field

Bit	Usage	Version info
0 .. 31	Reserved	

Label field

Not used.

Data fields

Field	Usage	Unit	Notes/Version info
et	Expiratory concentration	1/100%	
fi	Inspiratory concentration	1/100%	

4.2.7.4 Tonometry group

```
struct tono_group
{
    struct group_hdr    hdr;
    short               prco2;
    short               pr_et;
    short               pr_pa;
    short               pa_delay;
    short               phi;
    short               phi_delay;
    short               amb_press;
    short               cpma;
};
```

712 **Status field**

Bits	Usage	Version info
2	leak	
3	volume dropped in catheter	
4	technical failure	
5	unable to fill catheter	
6	PrCO2 over limit	
7 .. 31	Reserved	

713

714 **Label field**

715 **Data fields**

Field	Usage	Unit	Notes
prco2	PrCO2 concentration	1/100 kPa	
pr_et	P(r-Et)CO2 gap	1/100 kPa	
pr_pa	P(r-a)CO2 gap	1/100 kPa	
pa_delay	PaCO2 delay	min	
phi	pHi value	1/100	
phi_delay	pHa delay	min	
amb_press	Ambient pressure	1/10 mmHg	
Cpma	Research data	1	

716

4.2.8 Auxiliary physiological information (aux_phdb_info)

The auxiliary physiological subrecord contains the following fields

```
struct aux_phdb_info
{
        dword       nibp_time;
        short       cuff_press;
        dword       co_time;
        dword       pcwp_time;
        short       pat_bsa;
        short       ecg_display_size;
        short       spo2_display_size;
        short       invp1_display_size;
        short       invp2_display_size;
        short       invp3_display_size;
        short       invp4_display_size;
        short       resp_display_size;
        short       co2_scale;
        short       o2_scale;
        short       n2o_scale;
        short       aa_scale;
        short       reserved2_scale;
        short       flow_scale;
        short       reserved_scale;
        short       awp_scale;
        short       bp_unit;
        short       co2_unit;
        short       temp_unit;
        short       awp_unit;
        short       flow_unit;
        short       ie_unit;
        short       o2_display_offset;
        short       reserved_field;
        struct      misc_bits
        {
                    unsigned BFSHORT        tv_or_mv     : 1;
        } misc;
        short       invp5_display_size;
        short       invp6_display_size;
        word        old_or_new_gasmod;
        short       prco2_unit;
        long        prco2_age;
        dword       static_peep_time;
        short       eeg_scale;

        short       pvc_shown;

        byte        reserved[30];
};
```

nibp_time, co_time, pcwp_time	is the time of the latest NIBP, Cardiac Output and PCWP measurement. All times are defined as seconds since 1.1.1970. If a time is not known, the value is 0.
pat_bsa	is the patient's body surface area in 1/100 m^2 units.
ecg_display_size, spo2_display_size, invp1_display_size, invp2_display_size, invp3_display_size, invp4_display_size, resp_display_size, invbp5_display_size, invbp6_display_size	size of the waveform on the screen in vertical direction
co2_scale, o2_scale, n2o_scale, aa_scale, flow_scale, awp_scale, eeg_scale	TBD
bp_unit, co2_unit, temp_unit, awp_unit, flow_unit, ie_unit, prco2_unit	TBD
o2_display_offset	TBD
tv_or_mv	TBD
old_or_new_gasmod	TBD
prco2_age	TBD
static_peep_time	
Pvc_shown	PVC reading shown with HR in numberfield

5 Access to waveform data

The D-O Monitor Computer Interface provides limited access to the real-time waveform data produced by the monitor. Accessing the waveforms does not exclude access to the physiological data.

NOTE: The waveforms are available starting from Interface level 1 (software versions S-STD94 and S-ARK94).

5.1 Limitations

As a significant data transmission-rate is needed to produce a real-time waveform, this interface supports waveform transmission only up to a total of 600 samples (1200 bytes) per second. The waveforms can be freely selected within this limitation.

5.2 Record main types

Main type for all D-O records carrying waveform related data is

DRI_MT_WAVE (value 1)

5.3 DRI_MT_WAVE Record

5.3.1 Subrecord types

The following enumeration lists available subrecord types:

```
enum dri_wf_subtype
{
DRI_WF_CMD,
DRI_WF_ECG1,
DRI_WF_ECG2,
DRI_WF_ECG3,
DRI_WF_INVP1,
DRI_WF_INVP2,
DRI_WF_INVP3,
DRI_WF_INVP4,
DRI_WF_PLETH,
DRI_WF_CO2,
DRI_WF_O2,
DRI_WF_N2O,
DRI_WF_AA,
DRI_WF_AWP,
DRI_WF_FLOW,
DRI_WF_RESP,
DRI_WF_INVP5,                    ( Interface level 3 -> )
DRI_WF_INVP6,                    ( Interface level 3 -> )
DRI_WF_EEG1,                     ( Interface level 4 -> )
DRI_WF_EEG2,                     ( Interface level 4 -> )
DRI_WF_EEG3,                     ( Interface level 4 -> )
DRI_WF_EEG4,                     ( Interface level 4 -> )
};
```

5.3.2 Waveform request

A subrecord of type DRI_WF_CMD is used to carry transmission requests to the monitor. The subrecord has the following structure:

```
struct wf_req
{
short       req_type;                   // request type
short       secs;                       // duration of snapshot
byte        type[DRI_MAX_SUBRECS];      // waveform selectors
```

```
        byte    addl_type[2*DRI_MAX_SUBRECS];   // waveform selectors
        short   reserved[10];                   //
        };
```

req_type is the waveform request specifier. The following values are valid for this field:

0	to start continuous transmission of the specified waveform
1	To stop transmission of the waveform. In this case monitor ignores all other fields of this structure
2	Start timed continuos transmission

Waveform transmission request types:

```
enum dri_wf_req
{
    WF_REQ_CONT_START,          /* start continuous transmission */
    WF_REQ_CONT_STOP,           /* stop continuous transmission */
    WF_REQ_TIMED_START          /* start timed continuous   */
};
```

reserved is reserved for future use. This field must be set to 0 to ensure compatibility with future versions of the monitor.

type is an array of the requested waveform subrecords.

There is room for up to 8 waveforms, but the monitor sends only the waveforms that fit within the 600 samples/s limitation and ignores the rest.
The **type** array must be terminated using the DRI_EOL_SUBR_LIST constant (0xFF), unless there are 8 waveforms is the request.

The following example shows how fields are used as the CO_2 waveform is requested:

```
struct wf_req req;

req.req_type = WF_REQ_CONT_START;
req.type[0] = DRI_WF_CO2;
req.type[1] = DRI_EOL_SUBR_LIST; /* 0xFF */
req.secs = 0;
memset(req.reserved, 0, sizeof(req.reserved));
```

5.3.3 About the use of the waveform requests

As the monitor receives a valid waveform request, data transmission is started within one second. Before that the monitor checks for the 600 samples/s limitation and ignores all waveform subrecords in the record that exceed this limit.
For example, if the request contains DRI_WF_ECG1 (300 samples/s), DRI_WF_ECG2 (300 samples/s) and DRI_WF_INVP1 (100 samples/s), the invasive pressure waveform is ignored as the two ECGs fill up the bandwidth.
On the other hand, it is acceptable to request for all six invasive pressure waveforms (6 * 100

samples/s), or for 4 (4 * 100 samples/s) invasive pressures combined with 4 gas waveforms (4 * 25 samples/s), etc.
For details on sample rates, see chapter Structure of the waveform data.

The selected waveforms can be changed at any moment. If some other waveform is needed, the currently active transmission need not to be stopped.

Monitor keeps transmitting the waveform data as long as the serial line accepts it.

NOTE: If RTS/CTS handshaking disables data transmission for a time longer than 2 seconds, the monitor stops the transmission automatically.
A new waveform request is needed to restart transmission after this.

5.3.4 Structure of the waveform data

Each D-O waveform record contains one or more subrecords, the types of which match the types specified in the waveform request. Depending on the total number of samples/s the monitor sends a waveform packet every 1000 ms, 500 ms or 250 ms.

Waveform subrecords are of variable length. The actual length of the subrecord is included in the subrecord itself, but the length can be calculated using the subrecord offsets in the D-O header as well.

Each waveform subrecord has a 6 byte header, which has the following fields:

```
struct wf_hdr
{
        short act_len;
        word status;
        word label;
};
```

act_len is the actual number of 16-bit waveform samples in the subrecord following the waveform header.

status is a bit field:

```
WF_STATUS_GAP          0x0001   // bit 0, gap in sampling
WF_STATUS_MODULE_OFF 0x0002     // bit 1, module disconnected
WF_STATUS_PACER_DET    0x0004   // bit 2, pacer detected
WF_STATUS_LEAD_OFF     0x0008   // bit 3, ecg channel is off
```

If WF_STATUS_GAP bit is 1, there has been a sampling gap between the first sample of this subrecord and the last sample of the previous subrecord of same type.
The bit is normally set in the first subrecord following the reception of a waveform request.

The actual waveform data follows the waveform header. All samples are signed short integers (16 bits). Values less than or equal to -32000 are no true measurement data but control codes as specified in chapter “Structure of measurement data”.

Time between samples varies by subrecord type. Sampling speeds and units for various parameters are listed in the following table:

Subrecord	**Samples/s**	**Unit**
DRI_WF_ECGx	300	ECG x: µV
DRI_WF_INVPx	100	Invasive blood pressure x: 1/100 mmHg
DRI_WF_PLETH	100	Plethysmograph: modulation, 1/100%
DRI_WF_CO2	25	CO_2 concentration: 1/100%
DRI_WF_O2	25	O_2 concentration: 1/100%
DRI_WF_N2O	25	N2O concentration: 1/100%
DRI_WF_AA	25	Anaesthesia agent: 1/100%
DRI_WF_AWP	25	Airway pressure: 1/100 cmH_2O
DRI_WF_FLOW	25	Airway flow: 1/100 l/min
DRI_WF_VOL	25	Airway volume ?
DRI_WF_RESP	25	ECG impedance, 1/100 Ω
DRI_WF_EEGx	100	1/10 µV

--------- end of document ----------