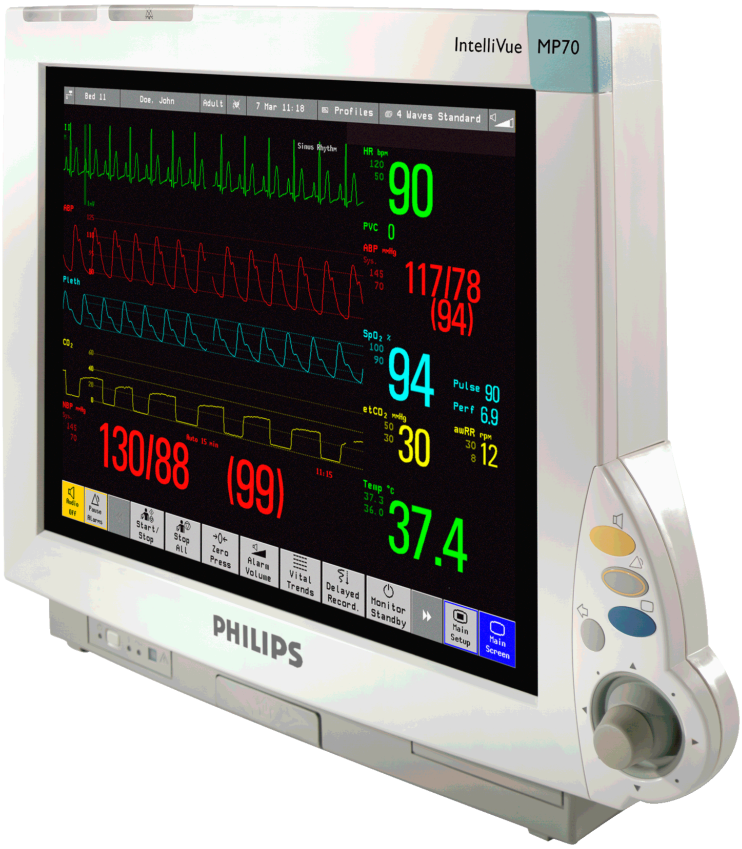
IntelliVue MP40/50/60/70/90



DATA EXPORT INTERFACE
PROGRAMMING GUIDE

IntelliVue Patient Monitor

MP40/50/60/70/90

Patient Monitoring

M8000-9305C

Reorder Part Number: 451261001426

PHILIPS

Table Of Contents

1

About this Guide

This Programming Guide is for use with the Philips IntelliVue MP40/MP50/MP60/MP70/MP90 (M8003A/M8004A/M8005A/M8007A/M8010A) patient monitors, hereafter referred to as the IntelliVue monitor. It describes the functionality in the monitor software version B.0x.xx and later.

The information in this Programming Guide describes the capability of the Data Export Interface. It is the responsibility of the user to create applications using the capability provided.

This device is not intended for home use.

In this guide

- A warning alerts you to a potential serious outcome, adverse event or safety hazard. Failure to observe a warning may result in death or serious injury to the user or patient.
- A caution alerts you where special care is necessary for the safe and effective use of the product. Failure to observe a caution may result in minor or moderate personal injury or damage to the product or other property, and possibly in a remote risk of more serious injury.

Who Should Use this Guide?

This programming guide is intended to be used by software professionals and biomedical engineers at medical research clinics or industrial institutions.

To successfully create an application, users should have a **good** working knowledge of:

- Advanced software application design.
- C and/or C++ Programming Language.
- General digital communications theory.
- Local Area Network configuration guidelines.and communication protocols.
- RS232 communication protocols and the IrDA protocol.

Given this background knowledge, this Programming Guide provides the information necessary to create your own applications.

Philips cannot provide any technical assistance for individual programming efforts.

About the Data Export Interface

This document describes the IntelliVue Data Export Interface. Using a communication interface protocol, data from the Philips IntelliVue Series Patient Monitor can be transferred via the Local Area Network (LAN) Interface or Medical Information Bus (MIB/RS232) Interface to an external Computer.

By creating basic applications using the IntelliVue Data Export Interface, the following data can be accessed from the IntelliVue monitor:

- All measurement numerics and alarm data (real-time update rates up to 1024 ms).
- Wave data (simultaneous access to 2 ECG and 5 other waves or 3 ECG and 2 other waves depending on monitor and client configuration)
- IntelliVue monitor system data.
- Patient demographic data entered by the user in the IntelliVue monitor.

The IntelliVue Data Export Interface cannot be accessed via the Local Area Network when the IntelliVue monitor is connected to the Philips LAN, e.g. to an Information Center (central station). Communication via the MIB/RS232 Interface is always possible.

Although alarm data can be accessed using the protocol, it must not be used as a real-time alarming system due to the delays in message transfer and the possibility of data loss.

Data Export Interface Features

- The IntelliVue Data Export Interface uses the Local Area Network (LAN) and MIB/RS232 interfaces.
- The LAN interface uses the standard UDP/IP transport protocol.
- The MIB/RS232 interface can be configured to use either a fixed or a variable baudrate protocol.
- The Data Export Protocol is a connection-oriented, message-based request/ response protocol on top of the transport protocol. The UDP and fixed baudrate transport protocols are connection-less, whereas the variable baudrate protocol is connection-oriented.
- The LAN interface supports automatic configuration of the network IP address with the standard BootP protocol.

Manufacturer’s Information

The information contained in this document is subject to change without notice. Philips makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

Philips shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

Philips assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Philips. Philips reserves the right to change the protocol described in this document without prior notice. Philips makes no warranty of any kind with regard to software applications that are created by the user. Philips assumes no responsibility for adverse interaction of an external Computer Interface communicant with the Philips IntelliVue Series Patient Monitor.

Philips Medizin Systeme Boeblingen Gmbh
Hewlett-Packard-Str. 2
71034 Böblingen
Germany

Trademark Information

Microsoft ® is a U.S. registered trademark of Microsoft Corp. Windows ® and Windows NT ® are U.S. registered trademarks of Microsoft Corp.

INFRARED DATA ASSOCIATION (IrDA) is a trademark of the Infrared Data Association in the USA and other countries.

2

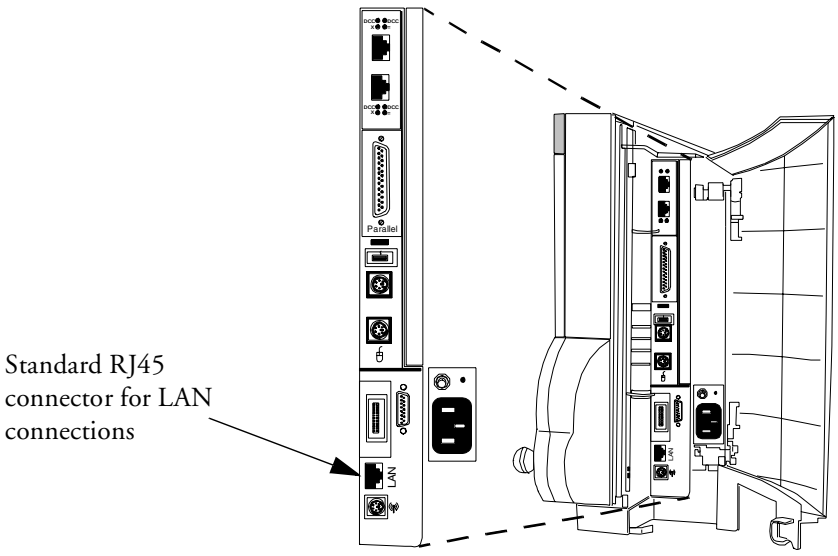
Connecting to the Network

The Philips IntelliVue Series Patient Monitor uses a standard IEEE802.3 10BaseT (10MBit/s) Local Area Network interface for the Data Export Capability.

The Data Export Interface via LAN is not available when the IntelliVue monitor is connected to the Philips LAN (e.g. to the Philips Information Center central station). Only devices approved for use with the Philips network may be connected to the Philips LAN.

Connecting to the Network via a LAN Interface

The IntelliVue monitor connects to the network using a standard unshielded LAN cable with an RJ45 connector. The network cable must be plugged to the orange-framed LAN connector on the left side of the IntelliVue monitor, as shown in the following diagram.



WARNING In order to maintain the galvanic isolation of the IntelliVue monitor, it is essential that UTP (Unshielded Twisted Pair) LAN cables must be used to connect the IntelliVue monitor to other devices.

The following LAN cables supplied by Philips can be used to connect the IntelliVue monitor:

- M3199AI #J10 - 3ft (0.91m), Part No. M3199-60103 (12NC: 453563337391)
- M3199AI #J11 - 7ft (2.1m), Part No. M3199-60104 (12NC: 453563337401)
- M3199AI #J12 - 12ft (3.6m), Part No. M3199-60105 (12NC: 453563337411)

The maximum cable length between the IntelliVue monitor and the Computer Client should never exceed 330ft (100m) in total.

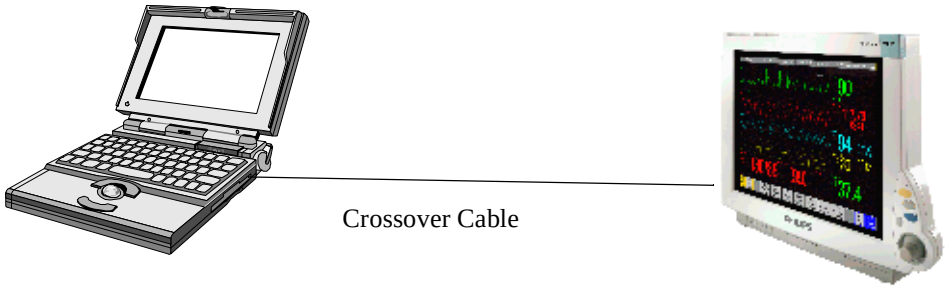
Connection via Hub/Switch

The IntelliVue monitor and the Computer Client are connected to a standard Ethernet switch or hub using UTP LAN cables.

NOTE In order to avoid high latency and data loss, it is strongly recommended to use a dedicated network that is exclusively used for patient data collection by IntelliVue monitor devices and Computer Clients.

Connection with Cross-over Cable

You can connect the IntelliVue monitor directly to the Computer Client, without a network hub or network switch, by using a UTP network crossover cable. In this case, the connection is a point-to-point connection only (one IntelliVue monitor connects to one Computer Client).



Avoiding Current Leakage

You must use Unshielded Twisted Pair (UTP) LAN cables to connect the IntelliVue monitor to other devices.

The Computer Client and network infrastructure devices typically are not classified as medical devices and must be located outside the patient vicinity. The patient vicinity is defined as an area within 6ft (1.85m) of the perimeter of the patient's bed or within 7.5ft (2.3m) of the floor.

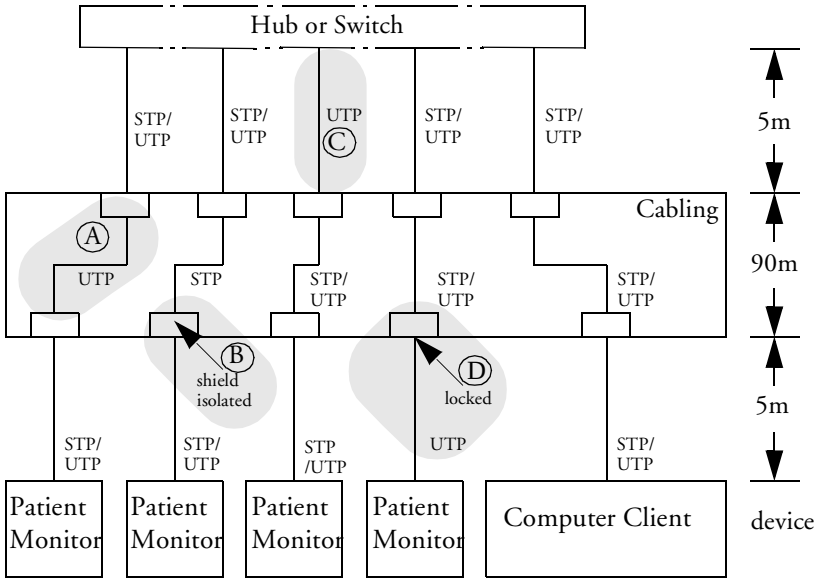
- If the Computer Client is installed in the patient vicinity and connected to the monitoring device, it must be correctly isolated from the mains power supply by an isolation transformer.
- If the Computer Client is installed in the patient vicinity and a network switch or hub is used to connect it to a monitoring device, it must be correctly isolated from the mains power supply by an isolation transformer.

WARNING All external devices in the patient vincinity must comply with IEC 60601-1:1988/A1:1991A2:1995 or EN 60601-1:1990/A1:1993/A2:1995. This applies also to all signal connections, entering the patient vincinity. Additional safety equipment, e.g. isolation transformers might be used.

The installation procedures e.g. for electrical connections as documented in the Instructions for Use must be strictly followed.

Using the Monitor with an Installed, Wired Network

The following diagram shows an overview of a possible LAN installation which provides galvanic isolation of the IntelliVue monitor:



If required by regulations valid in your hospital, the installation must comply to EN60601-1-1:1993/ A1:1996 or IEC 60601-1-1:1992/A1:1995.

The maximum cable length between the IntelliVue monitor and the Computer Client should never exceed 330ft (100m) in total.

WARNING In order to maintain the galvanic isolation of the IntelliVue monitor, it is essential that the shield is not connected from the IntelliVue monitor through to the hub or switch. At least one of the following precautions must be taken:

- UTP (Unshielded Twisted Pair) LAN cables are used in the wall.
- If STP (Shielded Twisted Pair) LAN cables are used in the wall, do not connect the shield of the cable from the IntelliVue monitor to the wall socket. Ensure that the shield of the STP cable in the wall is isolated from the other contacts. For a reference voltage of 250V, a clearance of at least 2.5 mm and a creepage distance of at least 4.0 mm is required. Cutting the shield back and covering it with a nonconducting shroud will fulfill this requirement.
- Ensure that only UTP cables are used in the wiring closet for connections to the hub or switch.
- Use only UTP cables such as M3199AI #J10/J11/J12 to connect the IntelliVue monitor to the wall socket. To avoid these cables being replaced by non-UTP cables, the connector which goes into the wall socket must be modified so that it cannot be removed without using tools. This can be done by cutting off the part of the plug lock which normally extends beyond the socket.

Configuring the LAN Interface

Configuring the Network Address

No explicit configuration of the network addresses (IP addresses etc.) is required. The IntelliVue monitor uses the standard BootP protocol to acquire an IP address and subnet mask from a BootP server in the network. If you are using a DHCP server, make sure the server supports BootP clients.

Without a working BootP server in the network, the IntelliVue monitor will show a technical alarm (INOP) “Unsupported LAN”, indicating that no (valid) IP address has been received.

Configuring the Network Setting

The Central Monitoring setting on the IntelliVue monitor determines whether the monitor requires a connection to the Philips Information Center (central station). If **`Central Monitoring`** is set to **`Mandatory`**, the monitor issues a technical alarm (INOP) if a network is detected without an Information Center (central station). If you are connecting the IntelliVue monitor to a Computer Client, **`Central Monitoring`** should be set to **`Optional`**.

To do this, in Configuration Mode,

1 Select **`Main Setup`** to enter the Main Setup menu.

2 Select **`Network`**

3 Select **`Central Monitoring`** and toggle to the appropriate setting:

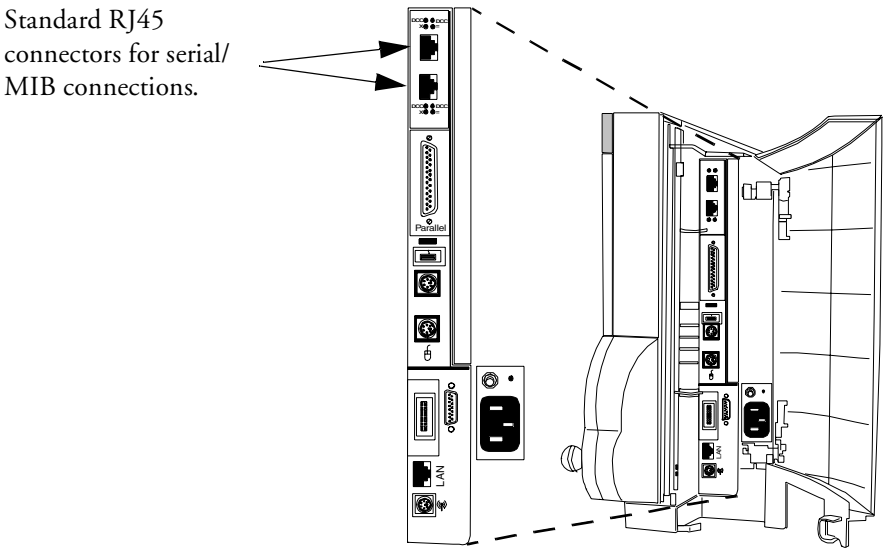
Mandatory	The IntelliVue monitor should be connected to an Information Center. An INOP is displayed if no connection is available.
Optional	The IntelliVue monitor can be connected to an Information Center. An INOP is only displayed if the connection to the Information Center is lost. No INOP is displayed if no connection is found at power on.

4 After the configuration, make sure you have stored all the active settings and leave Configuration Mode. You do not need a password to return to Monitoring Mode.

For further details on configuration, please refer to the IntelliVue configuration guide (M8000-9306X).

Connecting to the MIB/RS232 Interface

The IntelliVue monitor MIB/RS232 interface provides an eight-pin RJ-45 modular jack.



For the cable connection an eight conductor #24 American Wire Gage (AWG) unshielded twisted-pair (UTP) cable must be used. The cable must follow ANSI/TIA/EIA-568-A-1995 Category 5 (CAT-5). The cable length must not exceed 65ft (20m). Straight-through pinning must be used.

The physical specification of the MIB/RS232 Interface follows the standard IEEE 1073.3.2. Refer to the standard for more information on cables and pin assignment.

The MIB/RS232 interface provides a RS232 port with the following pin assignment. This table is valid when the MIB/RS232 Interface is in DCC mode (DCC LED on the MIB/RS232 board is on).

Computer Client	Pin and Signal Direction	IntelliVue monitor
	1<=	dDPWR
GND	4 <=>	GND
RxD	5 <=	TxD
TxD	7 =>	RxD

The pins of the RJ45 are counted from 1 for the lowest pin to 8 for the highest pin when looking at the RS232/MIB interface board.

NOTE This drawings and descriptions of the RS232/MIB board above apply to the IntelliVue MP60/70 monitors. Location and orientation of the board may vary, depending on the monitor purchased.

The TxD and RxD lines are the RS232 receive and transmit lines. The signals are referenced to the round (GND). The dDPWR can be used to power an external device with low power consumption. Refer to the Power Output specification in the table below.

Other applications in the IntelliVue monitor may be configured to use the MIB/RS232 Interface. These applications may use pins which are not used by the Data Export interface. Unused pins should not be connected. The IntelliVue monitor provides multiple RJ-45 connectors. Make sure, to use the correct connector with a port configured for Data Export.

The configuration of a specific MIB/RS232 port can be viewed in config mode and altered in service mode. To alter the configuration of an MIB port select **Main Setup** then **Hardware** then **Setup MIB/RS232.** This brings up the MIB/RS232 card configuration. The port that you are using must be set to **DtOut1** for the "Data Out" function. If the MIB/RS232 port is configured for data export the yellow arrow out LED will be lit.

NOTE Be aware that if you change a port assignment this assignment is not reset upon boot up. If the MIB/RS232 board is removed and replaced with a different type of board the settings are deleted. If the MIB/RS232 board is then refitted, you must reconfigure the MIB/RS232 port. The configuration of MIB/RS232 is not cloned between services.

Parameter	Limit
Driver (TxD)	
Driver load output voltage (3 kOhm to 7 kOhm load)	5 V <= |Vout| <= 15 V
Driver open-circuit voltage	|Vout| <= 25 V
Driver short-circuit current (to +/- 15 V)	|Iosv| <= 100 mA
Receiver (RxD)	
Receiver input resistance	3 kOhm to 7 kOhm
Maximum receiver input voltage	+/- 25 V
Receiver threshold	+/- 3V
Power output (dDPWR)	
Minimum output voltage	4.75 V
Maximum output voltage	5.25 V
Minimum guaranteed output current	100 mA
Maximum typical output current	150 mA

REPEATED INFORMATION: If the Computer Client is not classified as a medical device, it must be located outside the patient vicinity. The patient vicinity is defined as an area within 6ft (1.85m) of the perimeter of the patient's bed or within 7.5ft (2.3m) of the floor.

WARNING All external devices in the patient vincinity must comply with IEC 60601-1:1988/A1:1991A2:1995 or EN 60601-1:1990/A1:1993/A2:1995. This applies also to all signal connections, entering the patient vincinity. Additional safety equipment, e.g. isolation transformers might be used.

The installation procedures e.g. for electrical connections as documented in the User's Guide must be strictly followed.

If it is installed in patient vicinity, the Computer Client, connected to the instrument, must be correctly isolated from the mains power supply by an isolation transformer. The MIB/RS232 interface provides galvanic isolation of the IntelliVue monitor from a connected device.

Configuring the IntelliVue Monitor MIB/RS232 Interface

The MIB/RS232 interface supports different transport protocols.To change the MIB/RS232 interface configuration, in Configuration Mode,

1 Select **Main Setup**

2 Select **Setup Hardware**

3 Select **Data Export** and select the required setting:

AutoSpeed	Transport protocol with baudrate negotiation, based on the IrDA protocol.
Fix 19200	Transport protocol with a fixed baudrate of 19200 baud.
Fix 115200	Transport protocol with a fixed baudrate of 115200 baud.

4 Exit Configuration Mode. You do not need a password to return to Monitoring Mode.

For further details on configuration, please refer to the IntelliVue configuration guide (M8000-9306X).

3

Protocol Concept

The Protocol is based on a Client/Server Model. The Personal Computer *(Client)* maintains a logical connection with the Philips IntelliVue Series Patient Monitor (*Server)*. Communication occurs by sending and receiving Command messages.

Supported Transport Protocols

The Data Export functionality in the IntelliVue monitor can be accessed via the LAN interface or via the MIB/RS232 interface. While the Association Control and Data Export Protocol is the same for both interfaces, the underlying transport protocol varies.

- For the LAN interface the transport protocol is the standard UDP/IP protocol.
- For the MIB/RS232 interface, two transport protocols are supported:
  - a fixed baudrate protocol at 19200 or 115200 baud and
  - a protocol with baudrate negotiation (Auto Speed) based on the IrDA protocol with a baudrate from 9600 baud to 115200 baud.

Association Control and Data Export Protocol		
UDP/IP	RS232 Fixed Baudrate	RS232 Auto Speed
LAN Interface	MIB/RS232 Interface	

UDP/IP Protocol

The transport protocol uses the Universal Datagram Protocol/ Internet Protocol (UDP/IP).The protocol is based on the Request For Comment (RFC) internet standard. UDP is defined in RFC 768; IP is defined in RFC 760.

The UDP/IP transport protocol is part of the internet protocol suite. Drivers and necessary hardware are available for all relevant computing platforms. It provides for a simple exchange of messages (Datagrams) across a Local Area Network. The maximum size of user data in a protocol message can be negotiated at connection time between the IntelliVue monitor and the Computer Client.

Fixed Baudrate Protocol

The Fixed Baudrate Protocol provides a transport protocol with minimal overhead and complexity. It is intended for Computer Clients which cannot use the Auto Speed Protocol. The protocol operates at a fixed baudrate and can be used with standard RS232 concentrators. It provides packet-oriented data exchange and checksum protection on top of the RS232 protocol. For the specification of the Fixed Baudrate Protocol see “Transport Protocols for the MIB/RS232 Interface” on page 20.

Auto Speed Protocol

The Auto Speed Protocol is based on the IrDA protocol. It offers a reliable transport layer with checksum protection and a retry mechanism in the case of transmission problems. The baudrate can be negotiated in a range from 9600 baud to 115200 baud. For the specification of the AutoSpeed Protocol see “Transport Protocols for the MIB/RS232 Interface” on page 20.

Protocol Model

The protocol is based on an object-oriented modelling concept. All information available through the Data Export Protocol is modelled as attribute values of information objects.

The following information object classes are supported by the IntelliVue monitor:

- Medical Device System (MDS)

  The MDS object contains attributes representing dynamic state information (e.g. current operating mode) and static device specific identification information (e.g. Serial Numbers).

- Alert Monitor

  The Alert Monitor object contains attributes representing the current technical and patient alarms, as e.g. displayed on the IntelliVue monitor.

- Numeric

  Numeric objects contain attributes representing the state and value of numerical measurements (e.g. Heart Rate).

- Waves

  Realtime sample array objects contain attributes representing the state and value of wave data (e.g. ECG).

- Patient Demographics

  The Patient Demographics object contains attributes representing patient information stored in the IntelliVue monitor (e.g. Patient Name).

The object attributes can be accessed by a poll of the MDS object, which allows a query of the sets of attribute values from all objects of a specified class.

The method can be called by sending a command message from a Computer Client to the IntelliVue monitor.

Protocol Dialog

The following diagram shows the protocol dialog between the IntelliVue monitor Data Export server and a Computer Client:

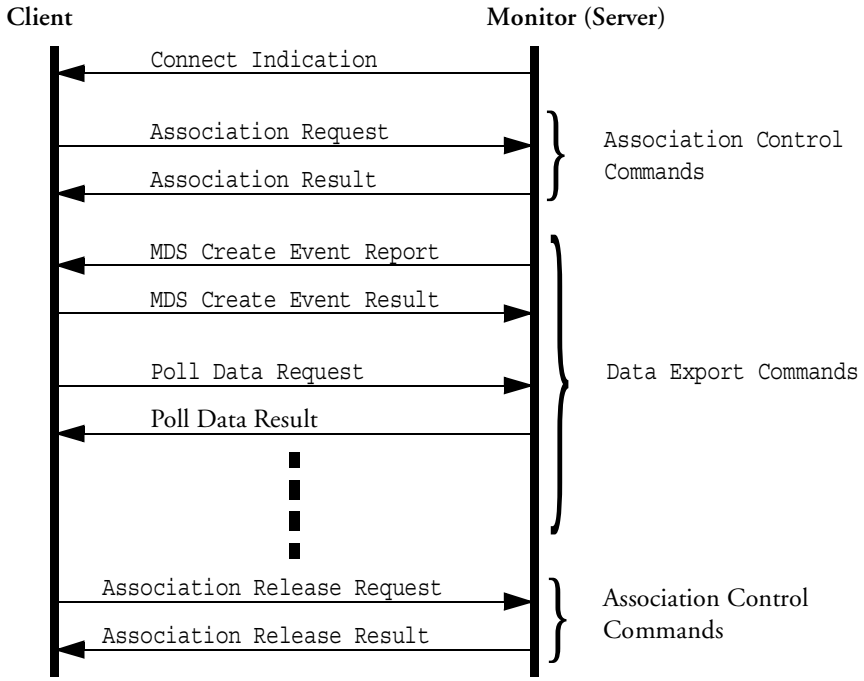


Figure 1 Protocol Dialog

Transport protocol-specific messages are not shown in the diagram. The Connect Indication message is only available on the LAN interface.

The Philips IntelliVue Series Patient Monitor processes global commands and sends response messages to the requests made by the client of the personal computer.

The messages shown in the diagram are explained in the following sections.

Connect Indication

The Connect Indication message is only sent on the LAN interface.

As soon as the IntelliVue monitor has received a valid IP address from the BootP server in the network, it sends out the Connect Indication message on its LAN interface. The message is a periodic subnet broadcast message that allows Computer Clients to find the IntelliVue monitor on the network. The message contains a set of device-related information, e.g., serial numbers, network addresses, internal states.

The IntelliVue monitor resends the Connect Indication message until a logical connection to a central station has been established. The IntelliVue monitor uses the retransmit strategy described in RFC 951. The resend period starts with 4 seconds and is doubled with each resend. The maximum resend period is about 64 seconds. The actual resend period contains a random component to avoid network congestion e.g. after a power failure.

Association Request

To establish a logical connection, the Computer Client sends the Association Request message to the IntelliVue monitor.

The Association Request can be used to set optional features of the logical connection between Computer Client and IntelliVue monitor.

Association Result

The IntelliVue monitor processes the Association Request and sends an Association Result. The result can be either a refuse message or an accept message.

The Computer Client must parse the Association Result to find out which protocol features can be used for this association.

MDS Create Event Report

If the IntelliVue monitor accepts the association, it sends a MDS Create Event Report after the positive Association Result message.

The MDS Create Event Report contains information about the system and its configuration.

MDS Create Event Result

The Computer Client must confirm the reception of the MDS Create Event Report. If the IntelliVue monitor does not receive a MDS Create Event Result message, the association is aborted.

Poll Data Request

After establishing an association, the Computer Client can send Poll Data Requests to access the data within the IntelliVue monitor.

The Poll Data Request contains a data-type parameter, which defines the specific type of requested data. The following data types are supported:

- Numeric Measurements
- Wave data
- Alerts (patient alarms and technical alarms)
- Patient Demographics
- System Attributes (e.g. dynamic state information, serial numbers, versions, etc.)

Only one type of data can be accessed per Poll Data Request.

Poll Result

Depending on the status of the IntelliVue monitor and the options set during the establishment of the logical connection (association phase), a Poll Data Request message can return:

- a single Poll Data Reply
- multiple, linked Poll Data Replies, if the size of the requested data exceeds the maximum size of a transport layer message
- a continuous number of periodic Poll Data Replies for a time period defined by the Computer Client (supported for Numeric Measurements, Waves and Alerts only).

Association Release Request

When the Computer Client wants to close an association, it can send a Release Request.

Association Release Result

The IntelliVue monitor parses the Release Request. If the Release Request is syntactically correct, the IntelliVue monitor sends an Association Release Result, indicating that the Association has been released.

Association Abort

In the case of communication problems, such as time-out, the IntelliVue monitor can send an Association Abort message. This message indicates that the association has been closed. A Computer Client should use the Association Release Request which provides a confirmation.

More Information

- For more details on the association control commands, such as Association Request, Association Result, Association Abort etc., please refer to the section “Definition of the Association Control Protocol” on page 55.
- For more detail on the data export commands, such as Poll Data Request, MDS Create Event Report, MDS Create Event Result, etc., please refer to the section “Definition of the Data Export Protocol” on page 25.

Connection Time-out Mechanism

The IntelliVue monitor automatically closes the connection if it detects a connection time-out condition. The connection time-out value is derived from the minimum poll period that is negotiated during the connection establishment phase.

A connection time-out period is 3 times the negotiated minimum poll period time. However, the minimum connection time-out is 10s, the maximum connection time-out period is 130s.

If the IntelliVue monitor does not receive a protocol message within the connection time-out period, the device closes the connection to the Computer Client by sending an Association Abort message. After that, a new connection can be established from the Computer Client to the IntelliVue monitor.

Network Load Consideration

Input Data

The IntelliVue monitor accepts a specific amount of input data per association. If the Computer Client sends more than the specified number of messages, the IntelliVue monitor will discard messages to avoid an unreasonably high system load. A Computer Client should be able to handle the loss of messages.

Message Type	Messages per Second
Association Control	1
Poll Request - Numerics (observed values)	1
Poll Request - Numerics (other attributes)	1
Poll Request - Waves	1
Poll Request - Alert Monitor	1
Poll Request - Patient Demographics	1
Poll Request - Medical Device System	1

The IntelliVue monitor will send a Remote Operation Error message if it receives a poll request for an object while it is still processing another poll request for the same object.

Output Data

The IntelliVue monitor processes the received message and sends the corresponding results. In rare cases, it can take up to several seconds until the response message is returned, and Poll Requests may be lost.

To avoid poll requests or poll responses getting lost, it is strongly recommended that the Computer Client uses the extended poll method to poll real-time numerics.

4

Definition of the Transport Protocols

Transport Protocols for the LAN Interface

UDP/IP

The Protocol uses the Universal Datagram Protocol/ Internet Protocol (UDP/IP) as the transport protocol. The protocol is based on the following internet standards (Request For Comment, RFC):

UDP is defined in RFC 768.

IP is defined in RFC 760.

The UDP/IP transport protocol is part of the internet protocol suite. Drivers and necessary hardware are available for all relevant computing platforms.

It provides for a simple exchange of messages (Datagrams) across a Local Area Network.

The maximum size of user data in a protocol message can be negotiated at connection time between the IntelliVue monitor and the Computer Client.

The upper limit for the negotiated user data size (MTU, Maximum Transport Unit) is 1364 bytes, the lower limit for the negotiated MTU is 500 Bytes. The maximum size of a UDP message sent by the IntelliVue monitor is 1380 bytes.

IP Address

The IP Address and the subnet mask necessary for communicating with the IP Protocol is set using the BootP protocol defined in the Internet RFC 951.

In order to communicate with the Philips IntelliVue Series Patient Monitor, a BootP server must exist in the network. The BootP server must be configured so that it answers BootP Request messages from the IntelliVue monitor.

UDP Port Number

The UDP Port Number used by the IntelliVue monitor for the Protocol can be extracted from the Connect Indication broadcast message used for Device Discovery (see “CONNECT INDICATION EVENT” on page 43). The current Protocol version uses the fixed UDP port 24105.

All messages sent from the Computer Client to the IntelliVue monitor must use this port number as the destination port number.

The Computer Client can chose any available source port for the communication. Once the Computer Client has chosen a source port, it must not use any other port. Protocol messages from another source port will be regarded as messages from a different Computer Client).

Any messages sent from the IntelliVue monitor back to the Computer Client use the source port number set by the Computer Client in first message (the Association Request message, see “Association Request Message” on page 57).

Transport Protocols for the MIB/RS232 Interface

The Fixed Baudrate Protocol, RS232 Port Settings

Each transmitted byte consists of one start bit, 8 data bits (no parity) and one stop bit. The baudrate can be set to 115kBit/s or 19.2kBit/s.

Flow control is not supported (same behavior as UDP). The monitor limits the number of Frames which will be processed in a given time. The monitor will process up to 4 frames within 128ms. If a client sends more frames, additional frames are ignored. (Implementation Note: the monitor allows 5 frames within 128ms, the additional frame is required because of possible jitter.)

A client system must be able to handle the loss of messages, because the Fixed Baudrate Protocol does not guarantee the reliable transmission of messages.

Framing

BOF	Hdr	User Data	FCS	EOF

The framing structure is the same as for AutoSpeed protocol. A frame starts with a single BOF.

BOF	Beginning Of Frame (0xC0)
Hdr	Header Information
User Data	Association Control or Data Export Command message
FCS	16 bit Frame Check Sequence using CRC-CCITT algorithm
EOF	End Of Frame (0xC1)

Header Information

The *Hdr* field is defined as follows:

```
typedef struct
{
    u_8             protocol_id;
    u_8             msg_type;
    u_16            length;
} FrameHdr;
```

The *protocol_id* field contains ID and version information. It can be used to define different service access points. Data Export uses the ID 0x11.

The *msg_type* field defines the type of message which is being sent. The value 0x01 indicates an Association Control or Data Export Command message, future message types could be used for flow control, lifetick, message confirmation etc.

The *length* field contains the length of the appended user data in bytes (without transparency characters).

If a client receives messages with an unknown *protocol_id* or *msg_type*, it should ignore the message.

Frame Check Sequence Field

The Frame Check Sequence Field can be used to detect transmission errors. The field contains a 16 bit CRC-CCITT cyclic redundancy check (not the popular XMODEM variation of CRC-CCITT). The CRC is computed from the *Hdr* and *User Data* field. Refer to "Serial Infrared Link Access Protocol (IrLAP)" Version 1.1 for the actual computation method of the CRC. A code snippet for the FCS algorithm can be found in the Network Working Group Request for Comment: 1171 (PPP protocol). The one's complement of the CRC is transmitted, rather than the CRC itself. The CRC is transmitted LSB first.

If the CRC is not correct, a client system should ignore the message.

Transparency

The contents of the *Hdr* and *User Data* fields is unrestricted. This can lead to problems if a BOF or EOF character appear in the *Hdr*, User Data, or FCS field. A Control Escape byte is defined as 0x7D. The sender must examine each byte in the User Data and FCS fields; for each byte with the value 0xC0, 0xC1, 0x7D it does the following:

- insert a 0x7D byte proceeding the byte
- complement bit 5 of the byte (XOR with 0x20).

Frame Abort

The sending station may abort the transmission of a frame by sending a control escape character followed by a EOF character (0x7D 0xC1) without sending the FCS field.

Examples The examples below do not include the *Hdr* field. For a correct message, the framing algorithm must be applied to the *Hdr* and *UserData* field of the message.

1 If a Computer Client wants to send the data:

"0x3a 0x71"

The CRC for this data would be:

"0xd9 0x64"

after building the one's-complement and byte-swapping, this results in:

"0x9b 0x26"

The whole frame would be:

"0xc0 0x3a 0x71 0x9b 0x26 0xc1"

2 If a Computer Client wants to send the data:

"0x3a 0x91"

The CRC for this data would be:

“0x3e 0x6a”

after building the one’s-complement and byte-swapping, this results in:

“0x95 0xc1”

The whole frame would be:

“0xc0 0x3a 0x91 0x95 0x7d 0xe1 0xc1”

Note that byte “0xc1” in the CRC is a reserved character and must be escape. This results in “0x7d 0xe1”.

The AutoSpeed Protocol

The AutoSpeed Protocol follows the definition of the Transport Protocol defined in the standard IEEE 1073.3.2: IEEE Standard for Medical Device Communications - Transport Profile - IrDA Based Cable Connection.

For a description of the IrDA Protocol refer to the specifications of the Infrared Data Association (www.irda.org):

- IrDA, Serial Infrared Link Access Protocol (IrLAP), Version 1.1, June 16, 1996
- IrDA, Link Management Protocol (IrLMP), Version 1.1, Oct. 20, 1996
- IrDA, Tiny TP: A Flow-Control Mechanism for use with IrLMP, Version 1.1, Oct. 20, 1996

Commercial IrDA stacks are available for most operating systems. Some operating systems, like Microsoft® Windows 2000® and Linux, come with an off-the-shelf IrDA stack.

The Data Export protocol resides as a packet oriented client on top of the IrDA TinyTP layer.

Establishing a Connection

A connection is created using the following steps:

- Discovery

The Computer Client sends an IrLAP discovery request to find out if a device is physically connected. The IntelliVue monitor answers with an discovery response message. The discovery procedure is done at a fixed baudrate of 9600 baud.

- Open an IrLAP connection

When the Computer Client finds a connected system, it can send an IrLAP Set Normal Response Mode message to establish a logical IrLAP connection. The IntelliVue monitor sends an response message. During this procedure parameters of the IrLAP connection, like baudrate, data size, etc. are negotiated.

- Open an IAS port

The Information Access Service (IAS) is provided by the IrLMP layer. It provides a database with device information which can be queried by the client. Before accessing the service, the client must connect to the special IrLMP service access point (SAP) 0.

- Perform an IAS query

The IrLMP layer does not specify a well-known SAP for the Data Export Protocol, hence the client should query the IAS database to find the SAP for the Data Export Protocol. The database contains the attribute “IrDA:TinyTP:LsapSel” under the object class “IEEE:1073:3:2:MDDL”. The attribute specifies the SAP for the Data Export Protocol on the IrDA TinyTP layer as an integer value.

- Close the IAS port

After performing the IAS query, the Computer Client should close the IAS port again with an IrLMP disconnect message.

- Open a Tiny TP connection

After retrieving the number for the TinyTP SAP, the client system can open a connection on this SAP. This is done with an IrLMP connect request message which contains a TinyTP connect in its user data.

- Send an Association Request

After the transport layer connection has been established, the Computer Client can send an Association Request message to start a Data Export session.

- Send a Release Request

When the client has no need for further communication, it can send a Release Request message to terminate the Data Export session.

- Close the IrLAP connection

After the Data Export session has been closed, theComputer Client should also close the TinyTP SAP. This can be done by sending an IrLMP disconnect message or by closing the whole IrLAP connection.

5

Definition of the Data Export Protocol

Definitions Shared by Protocols

Byte Order

The protocol data structures use the Network Byte Order. This means that bytes of a multi-byte data structure are transmitted on the network with the most significant byte first (as in big-endian data storage). This may or may not match the order in which numbers are normally stored in memory for a particular processor.

If the Computer Client is not using big-endian storage internally (many common Personal Computer Platforms use little-endian storage), protocol data structures (message structures) must be transformed before they are sent to a IntelliVue monitor or after they have been received from a IntelliVue monitor.

Byte Alignment

The Association Control and Data Export protocols assume no data alignment. However, most data types used in this guide have an even length for performance reasons. Many compilers use different alignment modes by default. Make sure that the compiler uses the right alignment when parsing and formatting protocol messages.

Bit Order

The index for bits starts with zero for the most significant bit.

MSB															LSB
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Common Data Types

Basic Data Types

The C data types defined here make use of the following basic types:

```
u_8     unsigned 8  bit wide integer
u_16    unsigned 16 bit wide integer
u_32    unsigned 32 bit wide integer
i_8     signed 8  bit wide integer
i_16    signed 16 bit wide integer
i_32    signed 32 bit wide integer
```

The mapping of these types to data types used in a Computer Client application is machine specific and compiler dependent.

Absolute Time

The Absolute Time data type is used whenever data is time stamped and a resolution of 1s is sufficient.

```
typedef struct {
        u_8             century;
        u_8             year;
        u_8             month;
        u_8             day;
        u_8             hour;
        u_8             minute;
        u_8             second;
        u_8             sec_fractions;
} AbsoluteTime;
```

The individual u_8 fields are BCD encoded, they are not encoded as regular integer values. E.g. the year 99 (decimal) is coded as 0x99.

Note that the time resolution in IntelliVue monitor with this format is 1 second. The *sec_fractions* element in the structure is not used.

Relative Time

The Relative Time is a high resolution time marker which defines a time relative to an event (e.g. power-on). It is used to position events (a particular event message) relative to each other with a higher resolution. It is defined as follows:

```
typedef u_32      RelativeTime;
```

The resolution of the *RelativeTime* is 1/8ms (125us). The IntelliVue monitor sets the Relative Time with a precision of 2 ms. The Computer Client can calculate the absolute time (wall clock) from a known relation between Absolute Time and Relative Time with a precision of about 1s. For more information on the time mapping refer to “MDS CREATE EVENT” on page 44.

OID Type

For the identification of all protocol elements (e.g. physiological meaning, alert codes, units of measure), the *OIDType* (Object Identifier Type) is used.

```
typedef u_16            OIDType;
```

Values for the *OIDType* (the nomenclature) are listed at the end of the section "Attribute Data Types and Constants Used" on page 65. Independent value ranges (partitions) exist, e.g. for physiological identifiers, alert condition identifiers, units of measurement etc.

Private OID

For the identification of private or manufacturer specific elements, a special type is used.

```
typedef u_16            PrivateOID;
```

Values for the *PrivateOIDs* are listed whenever a *PrivateOID* is used. Refer to the section "Attribute Data Types and Constants Used" on page 65 for a complete list of identifiers.

TYPE

Whenever it is not clear from the context, from which nomenclature value range the *OIDType* comes, the TYPE data type is used. Here, the nomenclature value range (the partition) is explicitly identified.

```
typedef u_16            NomPartition;
#define NOM_PART_OBJ          1
#define NOM_PART_SCADA        2
#define NOM_PART_EVT          3
#define NOM_PART_DIM          4
#define NOM_PART_PGRP         6
#define NOM_PART_INFRASTRUCT  8

typedef struct {
       NomPartition    partition;
       OIDType         code;
} TYPE;
```

The *code* values are grouped in the following partitions:

NOM_PART_OBJ: Object oriented element, device nomenclature

NOM_PART_SCADA: Types of measurement and place of the measurement

NOM_PART_EVT: Codes for alerts

NOM_PART_DIM: Units of measurement

NOM_PART_PGRP: Identification of parameter groups

NOM_PART_INFRASTRUCT: Infrastructure for Data Export applications

The *code* is only unique in a given partition. The values for the *OIDType* are defined in the section "Attribute Data Types and Constants Used" on page 65.

Handle

Object instances, e.g. Numeric object instances, are identified with a 16bit wide ID, the object Handle:

```
typedef u_16            Handle;
```

Global Handle

Handles are unique within the context of a particular system. The Protocol supports multiple measurement servers, where each measurement server assigns object handles independently. To assure handle uniqueness across system boundaries, the Global Handle contains an additional identifier for the source system, e.g., each measurement server has a distinct context id. The context id is assigned dynamically when a measurement server is connected.

```
typedef u_16            MdsContext;

typedef struct {
        MdsContext      context_id;
        Handle          handle;
} GlbHandle;
```

Managed Object Identifier

The Managed Object Identifier is a fully qualified object identifier which contains an identifier for the object class (e.g. Numeric object) together with a Global Handle.

```
typedef struct {
        OIDType         m_obj_class;
        GlbHandle       m_obj_inst;
} ManagedObjectId;
```

Attribute Value Assertion

Object attributes are represented in the form of data record structures which contain an identifier for the attribute, a length field for parsing and the actual value of the attribute.

The structure of such an attribute record is the Attribute Value Assertion, which is defined as follows:

```
typedef struct {
        OIDType         attribute_id;
        u_16            length;
        u_16            attribute_val;
} AVAType;
```

The *attribute_id* identifies the type of the attribute. The length field contains the size of the *attribute_val* field in bytes. The *attribute_val* field itself is only a placeholder in this structure. The parsing algorithm must assign the attribute value to the correct data structure based on the value of the *attribute_id*.

Attribute List

Typically, object instances have multiple attributes which are captured in a list with the following data type:

```
typedef struct{
        u_16            count;
        u_16            length;
        AVAType         value[1];
} AttributeList;
```

The count field contains the number of Attribute Value Assertion elements in the list.

The length field contains the size of the list (the value array) in bytes.

The value field itself again is only a placeholder data structure. A parser must be used to interpret the data structure. Refer to “Protocol Examples” on page 215 for an example of an *AttributeList*.

String

The text string is preceded by a length field, followed by the value. The *length* field denotes the number of octets in *value*. If the length is zero, no octets are appended. The *String* data type is used for UNICODE encoded texts.

```
typedef struct {
    u_16            length;
    u_16             value[1];
} String;
```

Where possible, the real string lengths have been included in this document. However, these string lengths may change in future releases, producing discrepancies between the actual string lengths and this document.

The *String* uses the same language as the IntelliVue monitor. The IntelliVue monitor uses UNICODE for the *String* data type (see “Connect Indication Attributes” on page 94). The *String* may contain code values from the UNICODE private use area (0xE000 to 0xF8FF). The Computer Client most likely will not support these characters. The following codes are frequently used:

```
#define SUBSCRIPT_CAPITAL_E_CHAR      0xE145
        /* SUBSCRIPT CAPITAL E                          */
#define SUBSCRIPT_CAPITAL_L_CHAR      0xE14C
        /* SUBSCRIPT CAPITAL L                          */
#define LITER_PER_CHAR                0xE400
        /* LITER PER - used in 4 char unit "l/min"      */
#define HYDROGEN_CHAR                 0xE401
        /* HYDROGEN - Used in 4 char unit "cmH2O"       */
#define ALARM_STAR_CHAR               0xE40D
        /* ALARM STAR "*"                               */
#define CAPITAL_V_WITH_DOT_ABOVE_CHAR 0xE425
        /* CAPITAL_V_WITH_DOT_ABOVE (V with dot)        */
#define ZERO_WIDTH_NO_BREAK_SPACE_CHAR0xFEFF
        /* The character 0xFFEF is used as FILL character.
        For each wide asian character, a FILL character is
        appended for size calculations. */
```

Variable Label

The string is preceded by a length field, followed by the value. If the length is zero, no octets are appended. The *VariableLabel* data type uses 8 bit ASCII encoding for the text. The *length* of a *VariableLabel* is always even.

```
typedef struct {
        u_16    length;
        u_8     value[1];
} VariableLabel
```

Where possible, the real string lengths have been included in this document. However, these string lengths may change in future releases, producing discrepancies between the actual string lengths and this document.

TextId

The *TextId* type is a 32bit wide private ID.

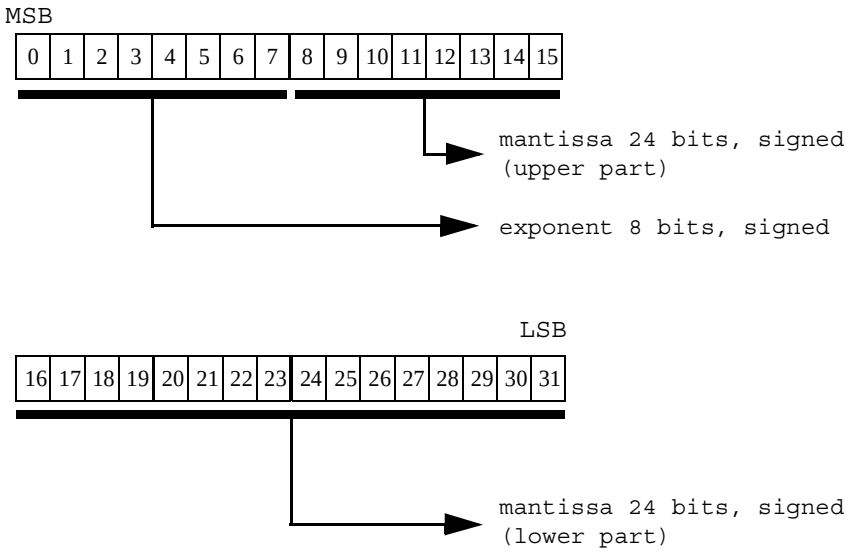
```
typedef u_32    TextId;
```

FLOAT-Type

For floating point numbers, a special 32bit wide format is used. For message parsing and for the definition of the message structures a 32bit wide placeholder structure is defined here.

```
typedef u_32    FLOATType;
```

The FLOAT-Type must be interpreted as follows:



The number represented is (mantissa)*($10^{exponent}$). Both the exponent and mantissa are in 2's complement form. The mantissa is not necessarily normalized.

There are four special values of the mantissa that can be represented:

NaN (Not a Number), which has a mantissa of $+(2^{23} -1)$ (0x7fffff)

NRes (Not at this resolution), which has a mantissa of $-(2^{23})$ (0x800000)

+/- INFINITY, which have mantissa of +/- $(2^{23}-2)$ (0x7ffffe, 0x800002).

The exponent is not important in these cases. This leaves the following ranges for normal number representation:

-128 <= exponent <= 127

$-(2^{23}-3)$ <= mantissa <= $+(2^{23}-3)$

Definition on the number of the valid digits for the presentation on the IntelliVue monitor's display:

1.) If the exponent < 0, then the integer value of the exponent shows the number of valid digits after the point:

Examples:

value = 0xfd007d00: exponent = -3, mantissa = 32000 ➡ 32.000

value = 0xff000140: exponent = -1, mantissa = 320 ➡ 32.0

2.) If the exponent >= 0, then the number of valid digits after the point is zero.

Examples:

value = 0x01000140: exponent = 1, mantissa = 320 ➡ 3200

value = 0x02000020: exponent = 2, mantissa = 32 ➡ 3200

Protocol Command Structure

Protocol Command messages, as defined in this section, are the data structures that are transported within the transport layer message (UDP datagram, IrDA message or Fixed Baudrate Protocol message). The generic structure is common for messages sent from the Computer Client to the IntelliVue monitor (e.g. Poll Request messages) and messages sent from the IntelliVue monitor to the Computer Client (e.g. Poll Result messages).

The Protocol Command messages represent the ISO/ OSI layers 5 - 7 (session layer, presentation layer, application layer). The message that transports a Protocol Command contains a checksum. Computer Clients should validate this checksum to detect corrupted messages.

The Protocol command messages used to establish the logical connection (association) between the IntelliVue monitor and a Computer Client follow the definitions of the ACSE Standard (ISO/IEC 8649 and ISO/IEC 8650).

For the Protocol Commands during the logical connection, the message structure is layered and has the following basic format:

Session/Presentation Header
Remote Operation Header
Command Header
Command- Specific Parameter Data

The Session Header and Presentation Header are small fields only which contain fixed values for the life time of the logical connection between the IntelliVue monitor and the Computer Client.

The Remote Operation Header allows to distinguish between the different types of command messages, command response messages and error messages.

The Command Header contains the common part of the Command data structure identified in the Remote Operation Header.

Command-specific parameters or data are appended to the generic message structure.

Session/Presentation Header

Each protocol message starts with a common data structure representing the session and presentation protocol, defined as follows:

```
typedef struct {
    u_16    session_id;    /* contains a fixed value 0xE100 */
    u_16    p_context_id;  /* negotiated in association phase */
} SPpdu;
```

- session_id

  This field identifies a Protocol message. The field contains a fixed value 0xE100. Conceptually, this field represents the session header.

- p_context_id

  The presentation context identifier is negotiated during the exchange of the association messages.

The Computer Client can use the first byte of the *session_id* to distinguish between Data Export protocol commands and Association Control protocol commands.

If a Computer Client encodes the Association Control protocol commands as suggested in “Definition of the Association Control Protocol” on page 55, the *context_id* for the Data Export protocol commands is 2.

Remote Operation Header

A protocol message is considered a remote operation. There are different types of operations as defined below. The different operations are described by a common operation header data structure:

```
typedef struct {
    u_16            ro_type;            /* ID for operation */
#       define      ROIV_APDU 1
#       define      RORS_APDU 2
#       define      ROER_APDU 3
#       define      ROLRS_APDU 5
    u_16            length;             /* bytes to follow */
} ROapdus;
```

- ro_type

  This field defines which type of remote operation is appended.

  The following remote operation types exist:

  Remote Operation Invoke (ROIV_APDU) invokes (calls) a remote operation.

  Remote Operation Result (RORS_APDU) returns the result of a remote operation

  Remote Operation Error (ROER_APDU) returns an error for a remote operation.

  Remote Operation Linked Result (ROLRS_APDU) returns parts of the result of a remote operation. It is used when the size of the complete result exceeds the maximum size of one message.

- length

  This field defines the remaining number of bytes in the message.

Remote Operation Invoke

A Remote Operation Invoke message is defined as follows:

```
typedef struct {
    u_16    invoke_id;     /* identifies the transaction */
    CMDType command_type;  /* identifies type of command */
    u_16    length;        /* no. of bytes in rest of message */
} ROIVapdu;
```

- invoke_id

  The invoke identifier is used to reference the specific operation while it is being processed. Result messages or error messages will use this identifier as a reference. Therefore, the invoke identifier should be unique while the operation transaction is in process.

- command_type

  The command type identifier defines what command data type is appended to this structure.

- length

  This field defines the remaining number of bytes in the message.

Remote Operation Result

A Remote Operation Result message is a response to an Operation Invoke message requiring confirmation.

The message is defined as follows:

```
typedef struct {
    u_16     invoke_id;    /* mirrored back from op. invoke */
    CMDType  command_type; /* identifies type of command */
    u_16     length;       /* no of bytes in rest of message */
} RORSapdu;
```

- invoke_id

  The invoke identifier is mirrored back from the related Remote Operation Invoke message that triggered this result. This field allows to relate the response message to the original request.

- command_type

  The command type identifier defines what command data type is appended to this structure.

- length

  This field defines the remaining number of bytes in the message. This length is not larger than the negotiated Maximum Transport Unit (MTU). For larger messages, the Remote Operation Linked Result mechanism will be used.

Remote Operation Linked Result

In some cases, the total data that must be returned as a result of a command may exceed the maximum message size. In these cases, multiple Remote Operation Linked Result messages are used.

These messages are defined as follows:

```
typedef struct {
    RorlsId  linked_id;    /* see below */
    u_16     invoke_id;    /* see below */
    CMDType  command_type; /* identifies type of command */
    u_16     length;       /* no of bytes in rest of message */
} ROLRSapdu;
```

- linked_id

  The linked identifier identifies each Remote Operation Linked Result message in a sequence of linked messages (see below).

- invoke_id

  The invoke identifier is mirrored back from the related Remote Operation Invoke message that triggered this result. This field allows to relate the response message to the original request.

- command_type

  The command type identifier defines what command data type is appended to this structure.

- length

  This field defines the remaining number of bytes in the message.

If the size of the result data exceeds the maximum message size, a combination of Remote Operation Linked Result Messages and Remote Operation Result messages is used, with the following rules:

- For all response messages except the very last one:
  - the ROLRS_APDU message type is used
  - the linked identifier is set by the responder to the *RorlsId* data type
  - the invoke identifier is the value of the invoke identifier of the associated Operation Invoke
- For the very last message:
  - The RORS_APDU message type is used

The invoke identifier in this response is the value of the invoke identifier of the associated Operation Invoke.

The following data type is used for the linked identifier:

```
typedef struct {
    u_8     state;
#     define RORLS_FIRST 1    /* set in the first message */
#     define RORLS_NOT_FIRST_NOT_LAST 2
#     define RORLS_LAST 3     /* last RORLSapdu, one RORSapdu
                                  to follow */
    u_8     count;             /* counter starts with 1 */
} RorlsId;
```

The first Remote Operation Linked Result message sets the state RORLS_FIRST.

The last Remote Operation Linked Result message sets the state RORLS_LAST. Note that there is one more Remote Operation Result message to follow.

All other Remote Operation Linked Result messages set the state RORLS_NOT_FIRST_NOT_LAST.

Examples:

- If a total of 3 messages are needed, the first message is a Remote Operation Linked Result with state RORLS_FIRST and count field 1.The second message is a Remote Operation Linked Result with state RORLS_LAST and count field 2.The third message is a Remote Operation Result message.
- If a total of 2 messages are needed, the first message is a Remote Operation Linked Result with state RORLS_LAST and count field 1. The second message is a Remote Operation Result message.

The *count* field starts with 1 for the first of the linked messages and is increased with each following message.

When a message is split, each message contains a full command data structure (see “Command Header” on page 37).

If the messages contain data from several objects, the Computer Client can not assume that all data belonging to one object is sent within one message. In some cases it can happen that the data belonging to one attribute of a given object must be sent in multiple messages (see the description of the available data in the section “Attribute Data Types and Constants Used” on page 65). This may only occur for attributes which are encoded in the form of a list (e.g Device T-Alarm List).

Object data which did not fit in one message is guaranteed to continue in the next linked message.

Remote Operation Error

If an error is detected at the Remote Operation level, an error message is returned:

```
typedef struct {
    u_16     invoke_id;
    u_16     error_value;
#       define      NO_SUCH_OBJECT_CLASS          0
#       define      NO_SUCH_OBJECT_INSTANCE       1
#       define      ACCESS_DENIED                 2
#       define      GET_LIST_ERROR                7
#       define      SET_LIST_ERROR                8
#       define      NO_SUCH_ACTION                9
#       define      PROCESSING_FAILURE            10
#       define      INVALID_ARGUMENT_VALUE        15
#       define      INVALID_SCOPE                 16
#       define      INVALID_OBJECT_INSTANCE       17
    u_16     length;
} ROERapdu;
```

- invoke_id

  The invoke identifier is mirrored back from the related Remote Operation Invoke message that triggered this result. This field allows to relate the response message to the original request.

- error_value

  The error values have the following meaning:

  GET_LIST_ERROR: Get operation failed. A *GetListError* is appended to the message.

  SET_LIST_ERROR: Set operation failed. A *SetListError* is appended to the message.

  NO_SUCH_ACTION: Unknown action type. The object class ID and action type are appended to the message.

  NO_SUCH_OBJECT_CLASS: There is no such object class in the system. An OIDType with the class ID is appended to the message.

  NO_SUCH_OBJECT_INSTANCE: The object instance does not exist. The *ManagedObjectId* of the instance is appended.

  ACCESS_DENIED: Computer Client has not required privileges to perform the operation. No data is appended.

  PROCESSING_FAILURE: Generic error indicating an invalid request. A *ProcessingFailure* is appended to the message.

  INVALID_ARGUMENT_VALUE: The argument of the ROSE message was not valid. An Action result is appended.

  INVALID_SCOPE: The scope is not valid for the operation. The value of the scope is appended.

  INVALID_OBJECT_INSTANCE: Wrong object instance. The *ManagedObjectId* of the instance is appended.

- length

  This field defines the remaining number of bytes in the message.

  The *GetListError* and *SetListError* structures are defined as follows:

```
typedef struct {
        ManagedObjectId         managed_object;
        struct {
                u_16                    count;
                u_16                    length;
                GetError                value[1];
        } getInfoList;
} GetListError;

typedef struct {
        ErrorStatus             errorStatus;
        OIDType                 attributeId;
} GetError;

typedef struct {
        ManagedObjectId         managed_object;
        struct {
                u_16                    count;
                u_16                    length;
                SetError                value[1];
        } setInfoList;
} SetListError;
typedef struct {
        ErrorStatus             errorStatus;
        ModifyOperator          modifyOperator;
        OIDType                 attributeId;
} SetError;

typedef u_16          ErrorStatus;
#define ATTR_ACCESS_DENIED            2
#define ATTR_NO_SUCH_ATTRIBUTE        5
#define ATTR_INVALID_ATTRIBUTE_VALUE  6
```

```
#define ATTR_INVALID_OPERATION        24
#define ATTR_INVALID_OPERATOR         25
```

The *ProcessingFailure* is defined as follows:

```
typedef struct {
    OIDType         error_id;
    u_16            length;
} ProcessingFailure;
```

Additional data with error information can be appended to the *ProcessingFailure*. The default *error_id* is 0 with no appended data.

Command Header

In each protocol message, a Command data structure is appended. The specific Command is identified by the value of the *CMDType* field in the Remote Operation Invoke/ Result/ Linked Result data structures.

The following Command types are used in the Protocol:

```
typedef u_16                CMDType;
#define                     CMD_EVENT_REPORT 0
#define                     CMD_CONFIRMED_EVENT_REPORT 1
#define                     CMD_GET 3
#define                     CMD_SET 4
#define                     CMD_CONFIRMED_SET 5
#define                     CMD_CONFIRMED_ACTION 7
```

The following command types are used:

CMD_EVENT_REPORT: An Event Report is used for an unsolicited event message.

CMD_CONFIRMED_EVENT_REPORT: The Confirmed Event Report is an unsolicited event message for which the receiver must send an Event Report Result message.

CMD_GET: The Get operation is used to request attribute values of managed objects. The receiver responds with a Get Result message.

CMD_SET: The Set operation is used to set values of managed objects.

CMD_CONFIRMED_SET: The Confirmed Set operation is used to set attribute values of managed objects. The receiver responds with a Set Result message.

CMD_CONFIRMED_ACTION: The Confirmed Action is a message to invoke an activity on the receiver side. The receiver must send an Action Result message.

For confirmed messages, the receiver must send the appropriate result message. For both the confirmed and unconfirmed Event Report, an *EventReportArgument* is appended.

If the result message is not received within 3 seconds, the IntelliVue monitor resends the message. If the message has not been confirmed after sending it 3 times (2 resend tries), the association is aborted by the IntelliVue monitor.

Event Report

The Event Report command (CMD_EVENT_REPORT) is used for unsolicited messages from the sending device to the receiving device. It is appended to the Remote Operation Invoke message. In the Protocol the Event Report may require a response from the receiver (if a response is required, the CMD_CONFIRMED_EVENT_REPORT Command identifier is used).

The Event Report message uses the following data structure:

```
typedef struct {
    ManagedObjectId managed_object; /* ident. of sender */
    RelativeTime    event_time;     /* event time stamp */
    OIDType         event_type;     /* identification of event */
    u_16            length;         /* size of appended data */
} EventReportArgument;
```

- managed_object

  Identifies the object that generates the unsolicited Event Report command.

- event_time

  The relative time (in 1/8ms time ticks) of the event.

- event_type

  Identifies the event type and thus the data structure that is appended.

- length

  This field defines the remaining number of bytes in the message (which is the size of the event specific data appended to this data structure).

Event-specific data is appended to the data type.

Event Report Result

The Event Report Result command is used as a response message to the Event Report message. It is appended to the Operation Result message with the *command_type* CMD_CONFIRMED_EVENT_REPORT.

The Event Report Result uses the following data structure:

```
typedef struct {
    ManagedObjectId managed_object; /* mirrored from EvRep */
    RelativeTime    current_time;   /* result time stamp */
    OIDType         event_type;     /* identification of event */
    u_16            length;         /* size of appended data */
} EventReportResult;
```

- managed_object

  Identifies the object to which the response is sent back. This field must be mirrored back from the Event Report message.

- event_time

  The relative time (in 1/8ms time ticks) of the event result.

- event_type

  Identifies the event type and thus the data structure that is appended. This field must contain the same value as the Event Report.

- length

  This field defines the remaining number of bytes in the message (which is the size of the event specific result data appended to this data structure).

Event-specific data is appended to the data type.

Action

The ACTION command (CMD_CONFIRMED_ACTION) is used to call a Protocol specific method in the receiver. The Protocol uses this command to call the *Data Poll* method which returns device data. The ACTION command is appended to the Operation Invoke message.

The Action command uses the following data structure:

```
typedef struct {
    ManagedObjectIdmanaged_object;/* addressed object */
    u_32           scope;         /* fixed value 0 */
    OIDType        action_type;   /* identification of method */
#define NOM_ACT_POLL_MDIB_DATA           3094
#define NOM_ACT_POLL_MDIB_DATA_EXT       61755
    u_16           length;        /* size of appended data */
} ActionArgument;
```

- managed_object

  Identifies the object to which the ACTION command is sent.

- scope

  Contains a fixed value 0 in this version of the protocol.

- action_type

  Identifies the specific method that should be called (and thus the data type that is appended to this data structure).

  NOM_ACT_POLL_MDIB_DATA is used for a Single Poll Data Request.

  NOM_ACT_POLL_MDIB_DATA_EXT is used for an Extended Poll Data Request

- length

  This field defines the remaining number of bytes in the message (which is the size of the method specific data appended to this data structure).

Method-specific data is appended to the data type.

Action Result

The Action Result command is used as a response message to the Action message. It is appended to the Operation Result message or an Operation Linked Result message (if the size of the returned data exceeds a maximum message size). The *command_type* is set to CMD_CONFIRMED_ACTION.

The Action Result uses the following data structure:

```
typedef struct {
    ManagedObjectId managed_object;
    OIDType        action_type;   /* identification of method */
    u_16           length;        /* size of appended data */
} ActionResult;
```

- managed_object

  Identifies the object that responds to the ACTION command (usually mirrored from ACTION command).

- action_type

  Identifies the specific method that was called (and thus the data type that is appended to this data structure).

- length

  This field defines the remaining number of bytes in the message (which is the size of the method specific result data appended to this data structure).

Method-specific data is appended to the data type.

Get

The Get command (CMD_GET) specifies attributes that should be returned. It is appended to an Operation Invoke message.

The Get command uses the following data structure:

```
typedef struct {
        ManagedObjectId         managed_object;
        u_32                    scope;
        AttributeIdList         attributeIdList;
} GetArgument;
```

- managed_object

  Identifies the object to which the Get command is sent.

- scope

  Contains a fixed value 0 in this version of the protocol.

- attributeIdList

  Contains the list of attribute identifiers.

```
typedef struct {
        u_16                    count;
        u_16                    length;
        OIDType                 value[1];
} AttributeIdList;
```

Get Result

The Get Result is returned in response to the Get command. It is appended to an Operation Result or Operation Linked Result message.

The Get Result uses the following data structure:

```
typedef struct {
        ManagedObjectId         managed_object;
        AttributeList           attributeList;
} GetResult;
```

- managed_object

  Identifies the object that responds to the Get command.

- attributeList

  Contains the requested attributes.

Set

The Set command (CMD_SET) or Confirmed Set command (CMD_CONFIRMED_SET) specifies attributes that should be added, replaced, or removed. It is appended to an Operation Invoke message.

The Set command uses the following data structures:

```
typedef struct {
        ManagedObjectId         managed_object;
        u_32                    scope;
```

```
        ModificationList        modificationList;
} SetArgument;
```

- managed_object

  Identifies the object to which the Get command is sent.

- scope

  Contains a fixed value 0 in this version of the protocol.

- modificationList

  Contains the attribute ids and values to be modified.

```
typedef struct {
        u_16                    count;
        u_16                    length;
        AttributeModEntry value[1];
} ModificationList;

typedef struct {
        ModifyOperator          modifyOperator;
        AVAType                 attribute;
} AttributeModEntry;

typedef u_16           ModifyOperator;
#define REPLACE                         0
#define ADD_VALUES                      1
#define REMOVE_VALUES                   2
#define SET_TO_DEFAULT                  3
```

Set Result

The Set Result is returned in response to the Confirmed Set command. It is appended to an Operation Result or Operation Linked Result message.

The Set Result uses the following data structure:

```
typedef struct {
        ManagedObjectId         managed_object;
        AttributeList           attributeList;
} SetResult;
```

- managed_object

  Identifies the object that responds to the Set command.

- attributeList

  Contains all modified attributes.

Command Structure Summary

The following diagram shows how the different generic Protocol Command command structures are built from the different data type definitions that were introduced in this section.

<table>
<tr><td colspan="10">SPpdu</td></tr>
<tr><td colspan="10">ROapdus</td></tr>
<tr><td colspan="3">ROIVapdu</td><td></td><td colspan="3">RORSapdu
ROLRSapdu</td><td></td><td>ROERapdu</td></tr>
<tr><td>Event Report Argument</td><td>Action Argument</td><td rowspan="2">Get Argument
Set Argument</td><td></td><td>Event Report Result</td><td>Action Result</td><td rowspan="2">Get Result
Set Result</td><td></td><td>Error Data</td></tr>
<tr><td>Event Data</td><td>Action Data</td><td></td><td>Event Result Data</td><td>Action Result Data</td><td></td><td></td></tr>
</table>

From this generic message structure the specific Protocol Command messages introduced in “Protocol Dialog” on page 15 are derived by:

- Defining identifier codes for the supported specific Event Report and Action types. These identifier codes are the values of the *event_type* and *action_type* fields.
- Defining the specific Event Data and Action Data data types for these Event Report and Action types.

Protocol Commands

This section describes the actual commands as constructed from the building blocks. Consult the “Command Structure Summary” on page 42 as a reference.

Notation

The Protocol Commands are constructed from the data types previously defined. A generic protocol machine must parse the individual elements of a command message separately, so in this chapter a special notation is used to define how the command messages are constructed (rather than defining composite C data type definitions).

Example:

```
MDSCreateEventReport ::=
 <SPpdu>
 <ROapdus (ro_type := ROIV_APDU)>
 <ROIVapdu (command_type := CMD_CONFIRMED_EVENT_REPORT) >
 <EventReportArgument (event_type := NOM_NOTI_MDS_CREAT)>
 <MDSCreateInfo>
```

This notation means that an MDS Create Event Report Command message is constructed from the individual data types listed in the < > brackets, which are C data types. Some elements of these data types have specific values. E.g. the *ro_type* field in the *ROapdus* data type has the value *ROIV_APDU*.

Additional data structures for appended event specific or method specific data are defined in the usual C type definition notation.

Most of the elements of the command messages contain length fields. You must take care to correctly set and parse these fields so that the message can be correctly parsed.

Device Discovery Messages

The Device Discovery messages lets the client locate new IntelliVue monitor devices in the network without prior knowledge of their IP address. The IntelliVue monitor only *sends a* Device Discovery on the LAN interface. This message is not available on the MIB/RS232 interface.

CONNECT INDICATION EVENT

The Connect Indication Event message is a sub-net-wide broadcast message in the normal Event Report format. It is sent to the port 24005.

The IntelliVue monitor resends the Connect Indication message as long as no logical connection to a central station has been established. The connection of a Data Export Computer Client does not stop the transmission of Connect Indication messages.

The IntelliVue monitor uses the retransmit strategy described in RFC 951. The initial resend period is 4 seconds, and this is doubled with each resend. The maximum resend period is approximately 64 seconds. The actual resend period contains a random component to avoid network congestion, e.g., after a power failure.

The UDP checksum in the Connect Indication message may be set to 0, indicating that no checksum has been calculated.

The Connect Indication message has the following structure:

```
ConnectIndication ::=
    <Nomenclature>
    <ROapdus (ro_type := ROIVapdu)>
    <ROIVapdu (command_type := CMD_EVENT_REPORT)>
    <EventReportArgument
            (managed_object:={NOM_MOC_MDS_COMPOS_SINGLE_BED,0,0},
            event_type := NOM_NOTI_MDS_CONNECT_INDIC)>
    <ConnectIndInfo>

typedef u_32 Nomenclature;
```

The nomenclature starts with two bytes 0x0, followed by one byte major and one byte minor version.

```
typedef AttributeList ConnectIndInfo;
```

See the section “Connect Indication Attributes” on page 94 for a list of attributes contained in the appended attribute list.

The Computer Client should parse the *ConnectIndInfo* to find out about the port for the Data Export protocol. The Computer Client must send requests to the port that is specified for the Data Export protocol.

The Computer Client application can run on any free local port, but must not change the port during the association (refer to “Definition of the Association Control Protocol” on page 55 for more information).

Connection Startup

After the logical connection has been established between the IntelliVue monitor and the Computer Client, the IntelliVue monitor sends the MDS Create Event message to announce version and status information.

MDS CREATE EVENT

The MDS Create Event describes the software and hardware configuration of the IntelliVue monitor. The Computer Client should parse this message to learn about the system configuration.

The MDS Create Event message has the following structure:

```
MDSCreateEventReport ::=
 <SPpdu>
 <ROapdus (ro_type := ROIV_APDU)>
 <ROIVapdu (command_type := CMD_CONFIRMED_EVENT_REPORT)>
 <EventReportArgument
    (managed_object := {NOM_MOC_VMS_MDS, 0, 0},
     event_type := NOM_NOTI_MDS_CREAT)>
 <MDSCreateInfo>
```

The MDS Create Information uses the following C type definition:

```
typedef struct {
    ManagedObjectId       managed_object;
    AttributeList         attribute_list;
} MdsCreateInfo;
```

- managed_object

  Identifies the MDS object. Contents is the same as in the *managed_object* field in the Event Report structure.

- attribute_list

  The attached *attribute_list* contains the IntelliVue monitor MDS attributes from the System Identification and from the System Application Attribute Group. See “Wave Objects” on page 71 for a list of all attributes

Depending on the protocol and the protocol options which were negotiated when the association was established, the IntelliVue monitor may map its internal data representation to a representation which is supported by the negotiated protocol. Hence, the Connect Indication message may describe the system differently from the MDS Create Event message. In the case of differences, the MDS Create Event is the relevant message.

The MDS Create Event message contains both the "Date and Time" and the "Relative Time" attributes. The Computer Client can use this data to make a mapping from the relative time to the absolute time of the IntelliVue monitor. The Computer Client should regularly check if the mapping is still valid by sending a Single Poll Data Request for the MDS attributes (“SINGLE POLL DATA REQUEST” on page 45).

If the size of the Event Report (Event Report Result + Event Result Data) exceeds the size of a maximum message (MTU - Maximum Transmit Unit), multiple messages are sent. Each of these messages is sent as a single Event Report.

The Computer Client must confirm the MDS CREATE EVENT with a MDS CREATE EVENT RESULT message, otherwise the association will be aborted by the IntelliVue monitor. The MDS CREATE EVENT message is resent with a period of about 3 seconds. The association is aborted if the Event message has been sent 3 times without receiving a confirmation.

When the MDS Create Event message is resent, it has the same invoke ID as the original message.

MDS CREATE EVENT RESULT

As the MDS Create Event Report is a confirmed operation, the Computer Client must send a MDS Create Event Result message to confirm it.

The reply message has the following structure:

```
MDSCreateEventResult ::=
 <SPpdu>
 <ROapdus (ro_type := RORS_APDU)>
 <RORSapdu
    (invoke_id := mirrored from event report,
     command_type := CMD_CONFIRMED_EVENT_REPORT)>
 <EventReportResult
    (managed_object := mirrored from event report,
     event_type := NOM_NOTI_MDS_CREAT)
     length := 0 >
```

As the MDS Create Event Result message does not contain any appended additional information, the length of the appended information is set to 0.

The result message must have the same *invoke_id* as the event message.

Specific Data Access Commands

The following protocol commands are used to access the different types of data in the IntelliVue monitor.

SINGLE POLL DATA REQUEST

This message can be sent as soon as the logical connection is established and the MDS Create Event/ Reply message sequence is finished. The message calls a method that returns IntelliVue monitor device data in a single response message.

The message has the following structure:

```
MDSPollAction ::=
 <SPpdu>
 <ROapdus (ro_type := ROIV_APDU)>
 <ROIVapdu (command_type := CMD_CONFIRMED_ACTION)>
 <ActionArgument
    (managed_object := {NOM_MOC_VMS_MDS, 0, 0},
     action_type := NOM_ACT_POLL_MDIB_DATA)>
 <PollMdibDataReq>
```

The *managed_object* must be the same as the *managed_object* in the MDS Create Event message. This is the top level object which actually implements the Data Export protocol.

The appended *PollMdibDataRequest* has the following data type:

```
typedef struct{
    u_16          poll_number;
    TYPE          polled_obj_type;
    OIDType       polled_attr_grp;
} PollMdibDataReq;
```

- poll_number

  This field will be sent back in the response message. It is recommended to use this field as a counter.

- polled_obj_type

  Defines which objects (Numerics or Alarms or MDS or Patient Demographics) is polled.

The following is a list of supported objects and their corresponding TYPE values:

```
NUMERICS:       partition:      0x0001
                code:           NOM_MOC_VMO_METRIC_NU
WAVES:          partition:      0x0001
                code:           NOM_MOC_VMO_METRIC_SA_RT
ALERTS:         partition:      0x0001
                code:           NOM_MOC_VMO_AL_MON
Pat.Demog:      partition:      0x0001
                code:           NOM_MOC_PT_DEMOG
MDS:            patition:       0x0001
                code:           NOM_MOC_VMS_MDS
```

The codes are taken from the Object Oriented Elements partition of the nomenclature (see “Object Classes” on page 98).

- polled_attr_grp

  Defines which set of attributes is polled. For more information on the supported attribute groups and their contents, please refer to the section “Attribute Data Types and Constants Used” on page 65.

The IntelliVue monitor specifies limits on the maximum frequency for incoming SINGLE POLL DATA REQUEST messages. If the Computer Client sends messages with a frequency above the limit, some of the messages will be ignored (no response is sent). Separate limits are calculated for each object.

The IntelliVue monitor will process a maximum of one POLL DATA REQUEST messages for each object type per second. An additional POLL DATA REQUEST for Numeric Observed Values is allowed.

SINGLE POLL DATA RESULT

This message is sent by the IntelliVue monitor in response to the Single Poll Data Request.

The message has the following structure:

```
MDSPollActionResult ::=
 <SPpdu>
 <ROapdus (ro_type := RORS_APDU)>
 <RORSapdu (invoke_id := "mirrored from request message"
            command_type := CMD_CONFIRMED_ACTION)>
 <ActionResult
    (managed_object := {NOM_MOC_VMS_MDS, 0, 0},
     action_type := NOM_ACT_POLL_MDIB_DATA)>
 <PollMdibDataReply>
```

The appended *PollMdibDataReply* is constructed from the following data types:

- The *PollMdibDataReply* structure is the top level data structure returned in the Single Poll Data Result message. It contains the following fields:

```
typedef struct {
    u_16                poll_number;
    RelativeTime        rel_time_stamp;
    AbsoluteTime        abs_time_stamp;
    TYPE                polled_obj_type;
    OIDType             polled_attr_grp;
    PollInfoList        poll_info_list;
} PollMdibDataReply;
```

- poll_number

  The poll number field contains the value of the same field in the Poll Request message.

- rel_time_stamp

  The Relative Time Stamp is a high resolution time stamp that represents the system time when the event message is sent by the IntelliVue monitor.

  For Numerics, the Relative Time Stamp denotes the time, when the Numeric measurement was generated. It may contain 0 if no measurement has been made yet.

- abs_time_stamp

  The IntelliVue monitor does not support Absolute Time Stamps in the Poll Data Result. All fields contain 0xff. If the Computer Client needs Absolute Time Stamps, it should use the corresponding MDS attributes ("Relative Time" and "Date and Time" to map the rel_time_stamp to an *abs_time_stamp*.

- polled_obj_type

  Defines for which objects (Numerics or Alarms or MDS or Patient Demographics) data is returned in the Poll Result message.

- polled_attr_grp

  Defines which set of attributes is returned in the Poll Result message.

- poll_info_list

  This structure contains the attribute values of the objects included in the poll.

  The Poll Info List is an array structure where each *SingleContextPoll* element contains the poll result data of one naming context.

```
typedef struct {
    u_16                count;
    u_16                length;
    SingleContextPoll   value[1];
} PollInfoList;
```

- count

  Number of Single Context Poll structures that are appended.

- length

  Size in bytes of the appended Single Context Poll structures.

- value

  This field is a placeholder field only. It represents the specified number of appended Single Context Poll structures.

The Single Context Poll structure contains polled data of all object instances within one unique naming context (IntelliVue monitor supports multiple naming contexts). It contains the following fields:

```
typedef struct {
    MdsContext            context_id;
    struct {
        u_16              count;
        u_16              length;
        ObservationPoll   value[1];
    } poll_info;
} SingleContextPoll;
```

- context_id

  The *context_id* field is used when the sourcing device represents multiple physical devices, so that the Handle attribute would not allow a unique identification of the object instance.

- poll_info.count

  This field contains number of appended Observation Poll structures.

- poll_info.length

  This field contains the length in bytes of the appended list of Observation Poll structures.

- poll_info.value

  This field is a placeholder field only. It represents the specified number of appended Observation Poll structures.

The *ObservationPoll* represents the polled data of one object instance. It contains the following fields:

```
typedef struct {
    Handle                obj_handle;
    AttributeList         attributes;
} ObservationPoll;
```

- obj_handle

  The handle identifies the object instance. It is used to identify the object in different Poll Reply Messages.

- attributes

  The attributes field is a list structured field containing the values of the polled object attributes. For a list of supported object attributes, see the chapter on “Attribute Data Types and Constants Used” on page 65.

If the size of data returned for a Poll Result (Action Result + Action Result Data) exceeds the size of a maximum message (MTU - Maximum Transmit Unit), multiple messages are returned. These messages use the Remote Operation Linked Result mechanism (“Remote Operation Linked Result” on page 34). This means that in all result messages except the last result message the *ROLRSapdu* is used instead of the *RORSapdu*).

When the Linked Result mechanism is used, the IntelliVue monitorm may send the terminating Remote Operation Result message with an empty *PollInfoList* (count and length fields of the *PollInfoList* set to 0). It also may send Linked Result messages with one empty SingleContextPoll (count and length field of the *SingleContxtPoll* set to 0).

EXTENDED POLL DATA REQUEST

The Extended Poll Data Request allows the following extensions of the Single Poll Data Request:

- Access 12 second, 1 minute and 5 minute averaged Numerics.
- Access wave data
- Request periodic Poll Replies without sending a Poll Request every time.
- Request that only a limited number of objects is encoded within a Poll Result

The Extended Poll Data Request message is only allowed, if the Poll Profile Extensions optional package has been negotiated during the association phase. For more information on the negotiation of optional packages see the sections “Association Request Message” on page 57 and “Association Response Message” on page 62.

The message has the following structure:

```
MDSPollAction ::=
 <SPpdu>
 <ROapdus (ro_type := ROIV_APDU)>
 <ROIVapdu (command_type := CMD_CONFIRMED_ACTION)>
 <ActionArgument
    (managed_object := {NOM_MOC_VMS_MDS, 0, 0},
     action_type := NOM_ACT_POLL_MDIB_DATA_EXT)>
 <PollMdibDataReqExt>
```

The appended *PollMdibDataRequestExt* has the following data type:

```
typedef struct{
    u_16           poll_number;
    TYPE           polled_obj_type;
    OIDType        polled_attr_grp;
    AttributeList  poll_ext_attr;
} PollMdibDataReqExt;
```

- poll_number

  This field will be sent back in the response message. It is recommended to use this field as a counter. See also the section “EXTENDED POLL DATA RESULT” on page 51 for more information about the handling of *poll_number*.

- polled_obj_type

  Defines for which objects data is returned in the Poll Result message. The Extended Poll Data Request message only allows the polling of Numerics, Waves and the Alert Monitor.

- polled_attr_grp

  Defines which set of attributes is returned in the Poll Result message.

- poll_ext_attr

  The appended *AttributeList* allows to define additional options.

Accessing 12 second, 1 minute and 5 minute averaged Numerics

Within the Poll Profile Extensions optional package, the Computer Client and the IntelliVue monitor have negotiated which data source (real-time or averaged) is used to obtain the Numeric data (refer to the chapter "Definition of the Association Control Protocol" on page 55 for more information on how to negotiate optional packages). Currently, the IntelliVue monitor allows the specification of one data source for Numeric data.

The IntelliVue monitor responds to an Extended Poll Data Request message with an Extended Poll Data Result message, which contains the Numeric data from the source specified in the Poll Profile Extensions optional package.

The normal Poll Data Request message always returns data from real-time measurements. If another data source has been negotiated in the Poll Profile Extensions optional package, the Poll Data Request message will fail, if no data from real-time measurements is available.

The *poll_ext_attr AttributeList* in the Extended Poll Data Request message allows to specify additional options. Currently, the following attributes are supported:

Attribute: Time Periodic Data Poll

The Time Periodic Data Poll attribute allows to request periodic Poll Replies for a given time.

```
Attribute ID:      NOM_ATTR_TIME_PD_POLL
Attribute Type:    PollDataReqPeriod
Attribute Groups:  -
Availability:      Optional
```

The PollDataReqPeriod is defined as follows:

```
typedef struct {
    RelativeTime    active_period;
} PollDataReqPeriod;
```

The *active_period* specifies the time for which the IntelliVue monitor will send periodic Poll Replies.

The *AttributeList* Structure may contain additional attributes, e.g. in future releases.

If the Computer Client adds the Time Periodic Data Poll attribute to the Extended Poll Data Request message, the IntelliVue monitor sends periodic Extended Poll Data Result messages for the time specified in the attribute.

Data Source	Result Period
real-time waves	256ms
real-time measurements	1s
12 secound averaged data	6s
1 minute averaged data	30s
5 minute averaged data	150s
alert data	1s

When the IntelliVue monitor receives an Extended Poll Data Request message, the first result message is sent immediately as a confirmation. It has the sequence number zero (see below). This allows the Computer Client to detect that its request was successful. The following messages are sent with the period specified in the table above.

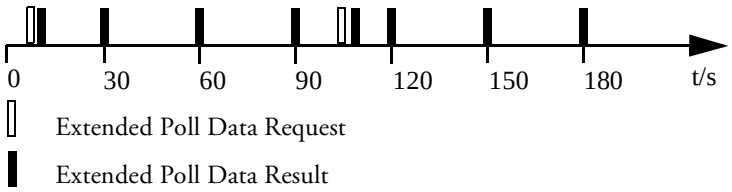


Figure 2 Period of Extended Poll Replies

The Computer Client should send a new Extended Poll Request before the time specified in the Time Periodic Data Poll attribute has expired. Each new Extended Poll Request is confirmed with an immediate Extended Poll Result message. However, the basic period of the replies is continued as illustrated in Figure 2 for 1 minute averaged data.

If the Computer Client uses the Extended Poll Request to access Realtime Numerics, it may happen that the IntelliVue monitor needs more than 1 second to encode all the data for the numerics (e.g. systems with a huge number of measurement modules). In this case the Poll Results will be sent at the highest possible frequency.

Limiting the Number of Objects in the Poll Result

In some cases, a Computer Client may want to limit the number of objects which are contained in a Poll Result. If the IntelliVue monitor is connected to a large number of measurement modules, a Poll Request for numerics will result in a large amount of data being sent from the IntelliVue monitor to the Computer Client.

Attribute: Number of Prioritized Objects

The attribute Number of Prioritized Objects specifies the maximum number of objects which will be encoded in the Poll Result.

```
Attribute ID:      NOM_ATTR_POLL_OBJ_PRIO_NUM
Attribute Type:    u_16
Attribute Groups:  -
Availability:      Optional
```

Based on an internal priority table, the IntelliVue monitor determines which objects will be added to the Poll Result. The priority table is constructed in the background, if the system configuration changes, it may take up to two minutes until the table has been updated. During this transition phase, the Poll Results sent by the monitor may contain less than the requested number of objects.

EXTENDED POLL DATA RESULT

When the IntelliVue monitor receives an Extended Poll Data Request message, it responds with a single or periodic Extended Poll Data Result messages.

The message has the following structure:

```
MDSPollActionResultExt ::=
 <SPpdu>
 <ROapdus (ro_type := RORS_APDU)>
 <RORSapdu (invoke_id := "mirrored from request message"
            command_type := CMD_CONFIRMED_ACTION)>
 <ActionResult
    (managed_object := {NOM_MOC_VMS_MDS, 0, 0},
     action_type := NOM_ACT_POLL_MDIB_DATA_EXT)>
 <PollMdibDataReplyExt>
```

The *PollMdibDataReplyExt* is defined as follows:

```
typedef struct PollMdibDataReplyExt {
    u_16                poll_number;
    u_16                sequence_no;
    RelativeTime        rel_time_stamp;
    AbsoluteTime        abs_time_stamp;
    TYPE                polled_obj_type;
    OIDType             polled_attr_grp;
    PollInfoList        poll_info_list;
} PollMdibDataReplyExt;
```

The *PollMdibDataReplyExt* structure is the top level data structure returned in the Extended Poll Data Result message. The appended data has the same structure as for the Single Poll Data Result.

The *PollMdibDataReplyExt* structure contains the following fields:

- poll_number

  The poll number field contains the value of the same field in the Extended Poll Request message.

- sequence_number

  The *sequence_number* is set to 0 when a new Extended Poll Data Request message is received. The IntelliVue monitor increases it with each periodic result message. This field allows the Computer Client to verify the sequence of the received result messages.

- rel_time_stamp

  The Relative Time Stamp is a high resolution time stamp that represents the system time when the event message is sent by the IntelliVue monitor.

  For Numerics, the Relative Time Stamp denotes the time when the Numeric measurement was generated. It may contain 0 if no measurement has been made yet.

  For Waves, the Relative Time Stamp denotes the beginning of the 256ms result period for real-time waves.

- abs_time_stamp

  The IntelliVue monitor does not support Absolute Time Stamps in the Poll Data Result. All fields contain 0xff. If the Computer Client needs Absolute Time Stamps, it should use the corresponding MDS attributes ("Relative Time" and "Date and Time" to map the rel_time_stamp to an abs_time_stamp).

- polled_obj_type

  Defines for which objects (Numerics or Alarms or MDS or Patient Demographic) data is returned in the Poll Result message.

- polled_attr_grp

  For more information on the supported attribute groups and their contents, please refer to the section “Attribute Data Types and Constants Used” on page 65.

- poll_info_list

  This structure contains the attribute values of the objects included in the poll.

Keep Alive Message

The IntelliVue monitor closes an association if it does not receive any protocol commands within a specified time (see “Definition of the Association Control Protocol” on page 55 to learn how the limit for a timeout is negotiated). If the Computer Client sends messages with a very low frequency (e.g. when using the extended poll mechanism) it must send a keep alive message to prevent the IntelliVue monitor from closing the association.

It is suggested that the Computer Client sends a Poll Data Request message for this purpose. This has the advantage that the message is confirmed and the Computer Client can detect a possible loss of the message. The Computer Client should chose a Poll Request which results in as little processing overhead as possible.

A suitable keep alive message would be a Poll Request for the Alert Monitor object, requesting the VMO Static Context Attribute group. The associated Poll Result sent by the IntelliVue monitor is a short message.

Specify Objects in the Poll Result

The Get and Set operations can be used to specify wave objects to be reported within the Poll Results.

There is a default priority list which depends on an internal priority table and the current system configuration. For wave objects, the default list can be replaced by a user defined priority list.

Due to the high amount of data it is always recommended to specify the required wave objects before requesting wave data.

GET PRIORITY LIST REQUEST

The message has the following structure:

```
MDSGetPriorityList ::=
        <SPpdu>
        <ROapdus (ro_type := ROIV_APDU)>
        <ROIVapdu (command_type := CMD_GET)>
        <GetArgument
                (managed_object := {NOM_MOC_VMS_MDS, 0, 0})>
```

The Get argument’s *AttributeIdList* specifies the attribute identifiers:

- NOM_ATTR_POLL_RTSA_PRIO_LIST

  Wave object priority list.

GET PRIORITY LIST RESULT

This message is sent in response to the Get Priority List Request.

The message has the following structure:

```
MDSGetPriorityListResult ::=
        <SPpdu>
        <ROapdus (ro_type := RORS_APDU)>
        <RORSapdu
                (invoke_id := “mirrored from request message”,
                command_type := CMD_GET)>
        <GetResult
                (managed_object := {NOM_MOC_VMS_MDS, 0, 0})>
```

The Get result's *AttributeList* contains the requested attribute identifiers and values. The *TextIdList* structure is used to define the wave object priority list:

```
typedef struct {
        u_16                    count;
        u_16                    length;
        TextId                  value[1];
} TextIdList;
```

The array of *TextId*s specifies the objects by their label, as returned in the dynamic context.

SET PRIORITY LIST REQUEST

The message has the following structure:

```
MDSSetPriorityList ::=
        <SPpdu>
        <ROapdus (ro_type := ROIV_APDU)>
        <ROIVapdu (command_type := CMD_CONFIRMED_SET)>
        <SetArgument
                (managed_object := {NOM_MOC_VMS_MDS, 0, 0})>
```

The Set argument's *ModificationList* specifies the modify operations, attribute identifiers, and new values (if needed).

For the REPLACE operation, a wave object priority list attribute with modified *TextIdList* structure is attached.

For the SET_TO_DEFAULT operation, there is an empty attribute (*length* is 0)attached.

The ADD_VALUES and REMOVE_VALUES operations are not supported.

SET PRIORITY LIST RESULT

This message is sent in response to the Set Priority List Request.

The message has the following structure:

```
MDSSetPriorityListResult ::=
        <SPpdu>
        <ROapdus (ro_type := RORS_APDU)>
        <RORSapdu
                (invoke_id := "mirrored from request message",
                command_type := CMD_CONFIRMED_SET)>
        <SetResult
                (managed_object := {NOM_MOC_VMS_MDS, 0, 0})>
```

The Set result returns the modified *AttributeList*, as defined above.

6

Definition of the Association Control Protocol

Protocol Command Structure

The Protocol messages to establish the logical connection (association) between the IntelliVue monitor and a Computer Client follow the definitions of the ACSE Standard (ISO/IEC 8649 and ISO/IEC 8650), with some proprietary extensions.

All Association Control Commands share a common structure as shown here:

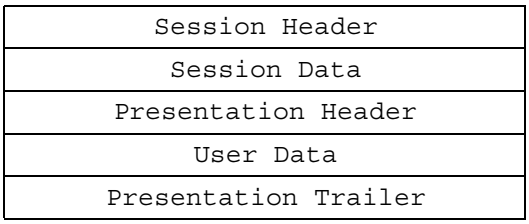


Figure 3 Protocol Commands for Association Control

For some messages, the Session Data and the User Data block may be empty.

A Computer Client can use the pre-defined building blocks for the Session Data, Presentation Header, and Presentation Trailer listed in the appendix to conveniently build valid messages (“Association Control Protocol Examples” on page 222 for a list of building blocks). Only the User Data block of the Association Request must be filled with Computer Client-specific data.

Protocol Commands

Protocol Command messages as defined in this section are the data structures that are transported within the transport layer messages.

The following commands are used to manage a logical connection between a Computer Client and a IntelliVue monitor:

- Association Request Message
- Association Response Message
- Refuse Message
- Release Request Message

- Release Response Message
- Abort Message

The Association Request message is sent from the Computer Client to the IntelliVue monitor when it wants to establish a new association. The *AssocReqUserData* contains information about the requested protocol and protocol options.

```
AssociationRequestMessage ::=
    <AssocReqSessionHeader>
    <AssocReqSessionData>
    <AssocReqPresentationHeader>
    <AssocReqUserData>
    <AssorReqPresentationTrailer>
```

The Association Response message is sent by the IntelliVue monitor if an Association Request message was parsed successfully and the association is accepted.

```
AssociationResponseMessage ::=
    <AssocRespSessionHeader>
    <AssocRespSessionData>
    <AssocRespPresentationHeader>
    <AssocRespUserData>
    <AssorRespPresentationTrailer>
```

If the Association Request message is corrupt, or if the association cannot be accepted (e.g. there is already another association), the IntelliVue monitor sends a Refuse message.

```
RefuseMessage ::=
    <RefuseSessionHeader>
    <RefuseSessionData>
    <RefusePresentationData>
    <RefuseUserData>
    <RefusePresentationTrailer>
```

When the Computer Client wants to terminate an association, it can send a Release Request message.

```
ReleaseRequestMessage ::=
    <ReleaseReqSessionHeader>
    <ReleaseReqSessionData>
    <ReleaseReqPresentationHeader>
    <ReleaseReqUserData>
    <ReleaseReqPresentationTrailer>
```

When the IntelliVue monitor receives a Release Request message, it sends a Release Response message as confirmation. The Release Response message indicates that the association has been terminated.

```
ReleaseRespMessage ::=
    <ReleaseRespSessionHeader>
    <ReleaseRespSessionData>
    <ReleaseRespPresentationHeader>
    <ReleaseRespUserData>
    <ReleaseRespPresentationTrailer>
```

The Abort message terminates an association without further confirmation. For example, the IntelliVue monitor sends an Abort message if an association is timed out (no communication from the Computer Client).

```
AbortMessage ::=
    <AbortSessionHeader>
    <AbortSessionData>
    <AbortPresentationHeader>
    <AbortUserData>
    <AbortPresentationTrailer>
```

Session Headers

The Session Headers can be used to identify the protocol commands. Each Session Header type maps to one protocol command.

The Session Header occupies the first bytes of the message. It is defined as follows:

```
typedef struct {
    u_8     type;
#   define CN_SPDU_SI              0x0D
#   define AC_SPDU_SI              0x0E
#   define RF_SPDU_SI              0x0C
#   define FN_SPDU_SI              0x09
#   define DN_SPDU_SI              0x0A
#   define AB_SPDU_SI              0x19
    LI      length;
} SessionHeader;
```

The *type* has the following meaning:

CN_SPDU_SI: A Session Connect header. The message contains an Association Request.

AC_SPDU_SI: A Session Accept header. The message contains an Association Response, indicating that the association has been established.

RF_SPDU_SI: A Session Refuse header. An association could not be established.

FN_SPDU_SI: A Session Finish header. The message contains a Release Request, indicating that the association should be terminated.

DN_SPDU_SI: A Session Disconnect header. The message contains a Release Response, indicating that the association has been terminated.

AB_SPDU_SI: A Session Abort header. The message contains an Abort message, indicating the immediate termination of the association.

If the first byte is 0xE1, the message is a Data Export Protocol command message (see “Definition of the Data Export Protocol” on page 25).

The *LI* field contains the length of the appended data (including all presentation data). The length encoding uses the following rules:

- If the length is smaller or equal 254 bytes, LI is one byte containing the actual length.
- If the length is greater than 254 bytes, LI is three bytes, the first being 0xff, the following two bytes containing the actual length.

Examples:

L = 15 is encoded as 0x0f

L = 256 is encoded as {0xff,0x01,0x00}

Message Encoding

The following section describes how a Computer Client can use the building blocks in the section “Association Control Protocol Examples” on page 222 to format correct Association Control messages.

Association Request Message

For the Association Request message, only the Session Header and the User Data must be filled out individually, as they contain variable data.

When using the building blocks, the presentation context ID for the Data Export Protocol is set to 2. This ID is sent in the SPpdu of all Data Export Protocol Commands.

The Session Header of the Association Request Message is defined as follows:

```
AssocReqSessionHeader ::=
    <SessionHead (type := CN_SPDU_SI)>
```

The length field in the Session Header must be set to the total length of the all appended data (including the presentation trailer).

Also the length field of the Presentation Header must be set to the total length of the appended message after this field. The field starts at the 2nd byte of the Presentation Header. It has the same format as the length field in the Session Header.

The User Data contains a specification of the requested protocol and protocol options. It is defined as follows:

```
AssocReqUserData ::=
    <ASNLength>
    <MDSEUserInfoStd>
```

The *ASNLength* contains the length of the *MDSEUserInfoStd*. It uses the following encoding rules:

- if the length is less or equal to 127, *ASNLength* is one byte, containing the actual length.
- if the length is greater than 127, *ASNLength* is several bytes long. The most significant bit (bit 0) of the first byte is set to 1, the bits 1 to 7 indicate the number of bytes which are appended to encode the actual length.

Examples:

L = 15 is encoded as 0x0f

L = 256 is encoded as {0x82,0x01,0x00}

The *MDSEUserInfoStd* is defined as follows:

```
typedef struct MDSEUserInfoStd {
    ProtocolVersion     protocol_version;
    NomenclatureVersion nomenclature_version;
    FunctionalUnits     functional_units;
    SystemType          system_type;
    StartupMode         startup_mode;
    AttributeList       option_list;
    AttributeList       supported_aprofiles;
} MDSEUserInfoStd;
```

The Computer Client must fill out the *MDSEUserInfoStd* data structure. It specifies the protocol versions and options the Computer Client supports. The IntelliVue monitor parses the *MDSEUserInfoStd* and constructs an Association Response message, which also contains a *MDSEUserInfoStd* data structure. The Association Response specifies which protocol versions and options will be used for the session.

The *ProtocolVersion* is a bit field containing the supported versions of the Data Export protocol. The Computer Client must set the bits for each version is supports. The IntelliVue monitor checks the supported versions and returns the bit for the highest commonly supported protocol version. If no matching version is found, the Association Request is refused.

```
typedef u_32       ProtocolVersion;
#define            MDDL_VERSION1  0x80000000
```

The *NomenclatureVersion* is a bit field containing the revision of the nomenclature which is used to name objects and their attributes. The Computer Client must set the bits for each version is supports. The IntelliVue monitor checks the supported versions and returns the bit for the highest commonly supported nomenclature version. If no matching version is found, the Association Request is refused.

```
typedef u_32 NomenclatureVersion;
#define            NOMEN_VERSION  0x40000000;
```

The *FunctionalUnits* is used to activate additional protocol functions. The Computer Client must set the bit for each functional unit it supports. The IntelliVue monitor checks the supported functional units and returns the bits for all commonly supported units (bitwise AND). No additional protocol functions have been defined yet.

```
typedef u_32       FunctionalUnits;
```

The *SystemType* is a bit field indicating whether the device is a Computer Client or a server. The Computer Client must set the SYST_CLIENT bit and the IntelliVue monitor will return the SYST_SERVER bit. If the SYST_CLIENT bit is not set in the Association Request, the association is refused.

```
typedef u_32 SystemType;
#define    SYST_CLIENT    0x80000000
#define    SYST_SERVER    0x00800000
```

The *StartupMode* is used to indicate the startup mode of the Computer Client and the IntelliVue monitor respectively. The IntelliVue monitor sets the bit for the startup mode which was used for the last reboot.

```
typedef u_32       StartupMode;
#define    HOT_START              0x80000000
#define    WARM_START             0x40000000
#define    COLD_START             0x20000000
```

If the IntelliVue monitor performs a COLD_START, all device settings are reset to the factory defaults. The configurations of the measurements might have changed and the patient data is lost.

The startup mode WARM_START and HOT_START indicate that configuration was not reset during the last restart.

The *option_list* can be used to negotiate additional protocol options in the form of an *AttributeList*. Currently, no further options are supported.

The *option_list* has a variable length. The offset of the *supported_aprofiles* field depends on the length of the *option_list.*

The *supported_aprofiles AttributeList* is used to define the available application profiles. An application profile specifies a set of protocol commands that is supported by the system. The Computer Client must add an entry for each supported profile to this list. The IntelliVue monitor parses the *supported_aprofiles* and returns the first profile in the list that is supported. If none of the profiles is supported, the Association Request is refused. The IntelliVue monitor supports the following profile:

Attribute: Poll Profile Support

The Poll Profile Support attribute contains the specification of the polling profile supported by the system.

```
Attribute ID:      NOM_POLL_PROFILE_SUPPORT
Attribute Type:    PollProfileSupport
Attribute Groups:  -
```

The *PollProfileSupport* is defined as follows:

```
typedef struct PollProfileSupport {
    PollProfileRevision    poll_profile_revision;
    RelativeTime           min_poll_period;
    u_32                   max_mtu_rx;
    u_32                   max_mtu_tx;
    u_32                   max_bw_tx;
    PollProfileOptions     options;
    AttributeList          optional_packages;
} PollProfileSupport;
```

The *PollProfileRevision* is a bit field containing the supported versions of the Polling Profile. The Computer Client must set the bits for each version it supports. The IntelliVue monitor checks the supported versions and returns the bit for the highest commonly supported profile version. If no matching version is found, the profile is not supported.

```
typedef u_32 PollProfileRevision;
#define POLL_PROFILE_REV_0        0x80000000
```

The *min_poll_period* specifies the minimum period with which the Computer Client wants to poll. If the IntelliVue monitor supports the requested poll period, it will return the value, otherwise it will return the minimum poll period it supports. The Computer Client should not send poll requests with a higher period than the negotiated value. For more information on poll periods, refer to the section “SINGLE POLL DATA REQUEST” on page 45.

The *min_poll_period* is also used to specify association time-outs. If the IntelliVue monitor does not receive any messages from the Computer Client within a given time, it sends an Abort message and terminates the association. The time-out periods depend on the negotiated *min_poll_period*, they are listed in the table below.

min_poll_period	**Association Time out**
< 3.3s	10s
3.3s ... 43s	3**min_poll_period*
> 43s	130s

The *max_mtu_rx* and *max_mtu_tx* fields contain the maximum size (MTU - Maximum Transport Unit) for protocol commands (the size of the protocol command is the size of the data appended after the Remote Operation Header).

The MTU negotiation uses the following procedure:

- The Computer Client determines the maximum size of a protocol command it can send and receive.
- The Computer Client sets *max_mtu_tx* to the maximum size it can transmit (i.e. the IntelliVue monitor should provide receive capabilities for messages of this size) and the *max_mtu_rx* to the maximum size it can receive (i.e. the IntelliVue monitor should not send larger commands).
- The IntelliVue monitor determines the maximum size of a protocol command it can send and receive.
- The IntelliVue monitor sets *max_mtu_tx* to the maximum size the Computer Client is allowed to transmit (this is the minimum of the *max_mtu_tx* the Computer Client requested and the message size the IntelliVue monitor *can receive).* The IntelliVue monitor sets *max_mtu_rx* to the maximum size the client must be able to receive (this is the minimum of the *max_mtu_rx the* Computer Client requested and the message size the IntelliVue monitor *can send).*

Example:

- The Computer Client can send 800 bytes and receive 500 bytes of user data in one message.
- The Computer Client sets *max_mtu_tx* to 800 and *max_mtu_rx* to 500.
- The IntelliVue monitor can send 700 bytes and receive 600 bytes in one message.
- The IntelliVue monitor sets *max_mtu_tx* to 600 bytes (the IntelliVue monitor can not receive larger messages) and *max_mtu_rx* to 500 bytes (the Computer Client can not receive more than 500 bytes in a message).

The IntelliVue monitor requires that the Computer Client can receive protocol commands of at least 300 bytes. Otherwise the profile is not supported. Smaller command sizes would lead to a considerable communication overhead. The largest negotiable MTU is 1364 bytes for the LAN interface and 1000 Bytes for the MIB/RS232 interface. The resulting size of the data packets may be larger than the MTU, because the MTU covers only the size of the Command Header and the Command Specific Data.

It is recommended that the Computer Client uses a large MTU. This reduces processing overhead and in most cases avoids splitting of messages.

For wave data export, the Computer Client needs to be able to receive observed values with 256 ms of wave data in one message. The MTU should be at least 500 bytes (700 bytes with multiplexed context).

The *max_bw_tx* contains the estimated maximum transmit bandwidth which will be used. The IntelliVue monitor fills in the maximum transmit bandwidth it uses, the value 0xffffffff indicates that no estimation is possible (this is the default). The current software does not support bandwidth estimation.

The *PollProfileOptions* bit field is used to set additional profile options. The IntelliVue monitor sets the P_OPT_DYN_CREATE_OBJECTS and P_OPT_DYN_DELETE_OBJECTS bits to indicate that the number of internal objects (e.g. the number of Numerics) may change dynamically. The *PollProfileOptions* is defined as follows:

```
typedef u_32 PollProfileOptions;
#define P_OPT_DYN_CREATE_OBJECTS          0x40000000
#define P_OPT_DYN_DELETE_OBJECTS          0x20000000
```

The *optional_packages AttributeList* allows the definition of additional options supported in the profile. The Computer Client must add an entry for each optional package it requests. The IntelliVue monitor checks the packages and adds an entry for each package it supports in the Association Response.

An attribute constitutes an optional package. The Poll Profile Extension is an optional package available for use.

Attribute: Poll Profile Extensions

The Poll Profile Extensions attribute specifies some extensions for the standard polling profile. For more information on how to use these extensions refer to the section “EXTENDED POLL DATA REQUEST” on page 49.

```
Attribute ID:      NOM_ATTR_POLL_PROFILE_EXT
Attribute Type:    PollProfileExt
Attribute Groups:  -
```

The *PollProfileExt* is defined as follows:

```
typedef struct {
    PollProfileExtOptions options;
    AttributeList         ext_attr;
} PollProfileExt;

typedef u_32 PollProfileExtOptions;
#define POLL_EXT_PERIOD_NU_1SEC           0x80000000
#define POLL_EXT_PERIOD_NU_AVG_12SEC      0x40000000
#define POLL_EXT_PERIOD_NU_AVG_60SEC      0x20000000
#define POLL_EXT_PERIOD_NU_AVG_300SEC     0x10000000
#define POLL_EXT_PERIOD_RTSA              0x08000000
```

The *PollProfileExtOptions* bit field defines available options for the Poll Profile Extensions package.

If the POLL_EXT_PERIOD_NU_1SEC bit is set, the Computer Client requests real-time measurements as source for Numeric data.

If the POLL_EXT_PERIOD_NU_AVG_12SEC bit is set, the Computer Client requests 12 second averaged data as source for Numeric data.

If the POLL_EXT_PERIOD_NU_AVG_60SEC bit is set, the Computer Client requests 1 minute averaged data as source for Numeric data.

If the POLL_EXT_PERIOD_NU_AVG_300SEC bit is set, the Computer Client requests 5 minute averaged data as source for Numeric data.

The Computer Client must set at least one of the bits, otherwise the optional package is ignored. Currently, the IntelliVue monitor supports only one source for an association. If more than one of the bits is set, the source with the smallest measurement period is selected. The IntelliVue monitor sets the corresponding bit in the Association Response message.

There may be only one active numeric source at a given time. If there is an active association on the LAN interface which has requested realtime numerics, it is not possible to establish another association on the MIB/RS232 interface which requests 1 minute averaged data. In this case, the association request would result in a refuse message.

If the POLL_EXT_PERIOD_RTSA bit is set, the computer client requests wave data. The patient monitor sets the corresponding bit in its response message to indicate wave data export capability.

The *ext_attr AttributeList* is reserved for future extensions.

The Computer Client must parse the Association Response message to find out whether the requested options have been accepted by the IntelliVue monitor.

Release Request Message

The Release Request message does not contain variable data. It is sufficient for the Computer Client to use the building blocks listed in the section “Association Control Protocol Examples” on page 222.

Abort Message

The Abort message does not contain variable data. It is sufficient for the Computer Client to use the building blocks listed in the section “Association Control Protocol Examples” on page 222.

Message Parsing

In most cases, it is sufficient for the Computer Client to check the first byte of the association control message. The first byte defines the Session Layer header, which can be mapped to an Association Control command.

Association Response Message

The IntelliVue monitor sends the Association Response message if an association has been established successfully. The Computer Client must parse the User Data within this message to find out which protocol options have been negotiated.

The Computer Client should not assume that the same Association Request message will always lead to the same Association Response message. The internal state of the IntelliVue monitor might lead to different responses.

The Association Response message is identified by its Session Header:

```
AssocRespSessionHeader ::=
    <SessionHead (type := AC_SPDU_SI)>
```

When parsing the Association Response message, the Computer Client must find the beginning of the User Data. This can be done by identifying the following byte sequence within the message;

```
0xBE 0x80 0x28 0x80 0x81
```

or

```
0xBE 0x80 0x28 0x80 0x02 0x01 0x02 0x81
```

The User Data is defined as follows;

```
AssocRespUserData ::=
    <ASNLength>
    <MDSEUserInfoStd>
```

The last byte of the User Data must be followed by 16 bytes 0x00.

The *MDSEUserInfo* follows the same definitions as described above for the Association Request Message.

Refuse

The IntelliVue monitor sends a Refuse message if an Association Request message was not accepted, because it was formatted incorrectly or because the requested protocol and protocol options are not supported by the IntelliVue monitor.

A Refuse message is also sent, if the maximum number of concurrent associations has been reached. Currently, the IntelliVue monitor only supports one active association.

The Refuse messages is identified by its Session Header:

```
RefuseSessionHeader ::=
    <SessionHead (type := RF_SPDU_SI)>
```

Release Response

It is sufficient to check the Session Header to detect a Release Response message. The Session Header is defined as follows:

```
ReleaseRespSessionHeader ::=
    <SessionHead (type := DN_SPDU_SI)>
```

7

Attribute Data Types and Constants Used

The data types in this chapter are based on the data types introduced in the chapter "Definition of the Data Export Protocol" on page 25. Refer to this chapter for more information about the base data types.

All data types used in this guide assume that elements of structures are aligned on 2 byte boundaries. Many compilers use different alignment modes by default. Make sure that the compiler uses the right alignments when parsing and formatting protocol messages.

The Poll Reply messages may contain attributes which are not documented here. A Computer Client should ignore all unknown attributes.

Numeric Objects

Numeric Object Attributes

This section defines the attributes of the Numeric object, together with the attribute identifier codes and attribute data types.

Attribute: Handle

The Handle attribute contains a unique identification of the Numeric object in the form of a numeric value. The actual value of the Handle attribute does not have a meaning. It is used for reference and relation purposes (e.g. Alert Monitor entries reference the Numeric object instance by means of the Handle).

```
Attribute ID:      NOM_ATTR_ID_HANDLE
Attribute Type:    Handle (see Definitions Shared by Protocols)
Attribute Groups:  VMO Static Context Group
Availability:      Mandatory
```

Attribute: Type

The Type attribute contains an identification of the object type.

```
Attribute ID:      NOM_ATTR_ID_TYPE
Attribute Type:    TYPE (see Definitions Shared by Protocols)
Attribute Groups:  VMO Static Context Group
Availability:      Mandatory
```

Attribute: Numeric Observed Value

The Numeric Observed Value attribute represents the (measured) value, along with state and identification data.

```
Attribute ID:      NOM_ATTR_NU_VAL_OBS
Attribute Type:    NuObsValue (see below)
Attribute Groups:  Metric Observed Value Group
Availability:      Conditional (either NuObsValue or
                   NuObsValueCmp must be present)
```

The *NuObsValue* data type is defined as follows:

```
typedef struct  {
    OIDType               physio_id;
    MeasurementState      state;
    OIDType               unit_code;
    FLOATType             value;
} NuObsValue;
```

The *physio_id* (physiological identifier) field contains a nomenclature code from the SCADA partition that identifies the represented value (typically a physiological measurement).

The *unit_code* field contains a nomenclature code from the dimension nomenclature partition. It identifies the units of measure.

The *value* field is a floating point number with the actual value. Before interpreting the numeric value, the *state* must be checked. Only if *state* indicates a valid measurement, should the *value* field be interpreted.

The *state* field is a bit field structure (multiple bits can be set simultaneously) defined as follows:

```
typedef u_16            MeasurementState;
#define INVALID                    0x8000
#define QUESTIONABLE               0x4000
#define UNAVAILABLE                0x2000
#define CALIBRATION_ONGOING        0x1000
#define TEST_DATA                  0x0800
#define DEMO_DATA                  0x0400
#define VALIDATED_DATA             0x0080
#define EARLY_INDICATION           0x0040
#define MSMT_ONGOING               0x0020
#define MSMT_STATE_IN_ALARM        0x0002
#define MSMT_STATE_AL_INHIBITED    0x0001
```

The bits have the following meaning:

INVALID: The source detects a sufficient degradation to render the data meaningless.

QUESTIONABLE: A problem exists, but it is still appropriate to present the data. This occurs when (1) either the degradation in the data is marginal or (2) the source cannot make a definite judgement on the reliability of the data.

UNAVAILABLE: The signal does not permit derivation of the numeric in question. This could be a transient state (e.g. first breath detected after an apnea -> no rate available), or a continuous state (no etCO_2 detection possible on a flat CO_2 wave).

CALIBRATION_ONGOING: Parameter is currently being calibrated.

TEST_DATA: The signal is an automatically generated test signal only and is not a valid patient signal. If this bit is set, the value is not suitable for patient diagnosis.

DEMO_DATA: The IntelliVue monitor runs in demonstration mode, the signal is automatically generated and is not a valid patient signal. If this bit is set, the value is not suitable for patient diagnosis.

VALIDATED_DATA: The value has been manually validated.

EARLY_INDICATION: The value represents an early estimate of the actual signal (the Non-Invasive Blood Pressure measurement e.g. sets this bit as soon as it has derived a systolic value, even if mean and diastolic values are still missing).

MSMT_ONGOING: A new aperiodic measurement is currently ongoing.

MSMT_STATE_IN_ALARM: Indicates that the numeric has an active alarm condition

MSMT_STATE_AL_INHIBITED: Alarms are switched off for the numeric (crossed bell)

The measurement is valid if the first octet of the state is all 0.

Attribute: Compound Numeric Observed Value

The Compound Numeric Observed Value attribute represents multiple (measured) values modelled in one Numeric object, along with state and identification data.

The Compound Numeric Observed Value is e.g. used to represent Blood Pressure measurements. For these measurements, systolic, diastolic and mean values are represented by a single Numeric object.

```
Attribute ID:      NOM_ATTR_NU_CMPD_OBS_VAL
Attribute Type:    NuObsValCmp (see below)
Attribute Groups:  Metric Observed Value Group
Availability:      Conditional (either NuObsValue or
                   NuObsValueCmp must be present)
```

The *NuObsValueCmp* data type is defined as follows:

```
typedef struct {
    u_16           count;
    u_16           length;
    NuObsValue     value[1];
} NuObsValueCmp;
```

The count field defines the number of *NuObsValue* elements in the structure. Note that the count field is variable, the number of elements may change over time. For a Blood Pressure measurement e.g there can be 3 values (systolic, diastolic, mean) or a single value only (mean only).

The length field defines the size of the array of *NuObsValue* structures in bytes.

The value field is a place holder for parsing.

Attribute: Absolute Time Stamp

The Absolute Time Stamp attribute is used to define a time tag for the current Numeric value. In the IntelliVue monitor, the attribute is used for aperiodic measurements only.

```
Attribute ID:      NOM_ATTR_TIME_STAMP_ABS
Attribute Type:    AbsoluteTime(see Definitions Shared by Protocols)
Attribute Groups:  Metric Observed Value Group
Availability:      Optional
```

Attribute: Relative Time Stamp

The Relative Time Stamp attribute is used to define a high resolution time tag for the current Numeric value.

```
Attribute ID:      NOM_ATTR_TIME_STAMP_REL
Attribute Type:    RelativeTime(see Definitions Shared by Protocols)
Attribute Groups:  Metric Observed Value Group
Availability:      Optional
```

Attribute: Label

The Label attribute is a 32 bit wide ID which represents the Numeric label string. The Label is unique for all numerics in the system.

```
Attribute ID:       NOM_ATTR_ID_LABEL
Attribute Type:     TextId
                    (see Protocol Common Definitions)
Attribute Group:    VMO Dynamic Context Group
Availability:       Optional
```

Attribute: Label String

The Label String attribute is a unicode string which contains the label string for a Numeric.

```
Attribute ID:       NOM_ATTR_ID_LABEL_STRING
Attribute Type:     String
                    (see Protocol Common Definitions)
Attribute Group:    VMO Dynamic Context Group
Availability:       Optional
```

Attribute: Display Resolution

The Display Resolution attribute is present if the resolution of the numeric shown on the display must be different from the resolution communicated in the Numeric Observed Value attribute. E.g. a Temperature is displayed with a resolution of 1/10, but the Observed Value is sent with a precision of 1/100 to get the necessary accuracy for differential temperatures. The Display Resolution attribute describes the format in which the value of a numeric is displayed on the screen.

```
Attribute ID:       NOM_ATTR_DISP_RES
Attribute Type:     DispResolution
Attribute Group:    VMO Dynamic Context Group
Availability:       Optional
```

The *DispResolution* is defined as follows:

```
typedef struct
{
    u_8     pre_point;
    u_8     post_point;
} DispResolution;
```

The value of *pre_point* denotes the number of digits before the decimal point. The value of *post_point* denotes the number of digits after the decimal point.

Attribute: Color

The Color attribute describes the color in which a numeric is displayed on the screen.

```
Attribute ID:       NOM_ATTR_COLOR
Attribute Type:     SimpleColour
Attribute Group:    VMO Dynamic Context Group
Availability:       Optional
```

The *SimpeColour* is defined as follows:

```
typedef u_16 SimpleColour;
#define COL_BLACK       0
#define COL_RED         1
#define COL_GREEN       2
#define COL_YELLOW      3
#define COL_BLUE        4
#define COL_MAGENTA     5
#define COL_CYAN        6
#define COL_WHITE       7
#define COL_PINK        20
#define COL_ORANGE      35
#define COL_LIGHT_GREEN 50
#define COL_LIGHT_RED   65
```

Attribute: Metric Specification

The Metric Specification attribute describes static properties of a numeric.

```
Attribute ID:       NOM_ATTR_METRIC_SPECN
Attribute Type:     MetricSpec
Attribute Group:    VMO Static Context Group
Availability:       Mandatory
```

The *MetricSpec* is defined as follows:

```
typedef struct
{
    RelativeTime        update_period;
    MetricCategory      category;
    MetricAccess        access;
    MetricStructure     structure;
    MetricRelevance     relevance;
} MetricSpec;
```

The *update_period* is the minimum time between changes of the observed value.

The *MetricCategory* is defined as follows:

```
typedef u_16            MetricCategory;
#define MCAT_UNSPEC                 0
#define AUTO_MEASUREMENT            1
#define MANUAL_MEASUREMENT          2
#define AUTO_SETTING                3
#define MANUAL_SETTING              4
#define AUTO_CALCULATION            5
#define MANUAL_CALCULATION          6
#define AUTO_ADJUST_PAT_TEMP        128
#define MANUAL_ADJUST_PAT_TEMP      129
#define AUTO_ALARM_LIMIT_SETTING    130
```

It allows to distinguish between measurements, calculations and settings. The values have the following meaning:

MCAT_UNSPEC: not specified

AUTO_MEASUREMENT: automatic measurement

MANUAL_MEASUREMENT: manual measurement

AUTO_SETTING: automatic setting

MANUAL_SETTING: manual setting

AUTO_CALCULATION: automatic calculation, e.g. differential temperature

MANUAL_CALCULATION: manual calculation

AUTO_ADJUST_PAT_TEMP: measurement is automatically adjusted for patient temperature

MANUAL_ADJUST_PAT_TEMP: measurement manually adjusted for patient temperature

AUTO_ALARM_LIMIT_SETTING: this is not a measurement, but an alarm limit setting

The *MetricAccess* bit field provides info on how the metric value can be accessed and when a measurement is available.

```
typedef u_16            MetricAccess;
#define AVAIL_INTERMITTEND      0x8000
#define UPD_PERIODIC            0x4000
#define UPD_EPISODIC            0x2000
#define MSMT_NONCONTINUOUS      0x1000
```

The values have the following meaning:

AVAIL_INTERMITTEND: The intermitted availability bit is set, if the observed values not always available (e.g. only if a measurement is explicitly started).

UPD_PERIODIC: observed value is updated periodically

UPD_EPISODIC: observed value is updated episodically (exactly one update mode (UPD_) must be set

MSMT_NONCONTINUOUS: indicates that the measurement is non continuous (this is different from the update mode)

The *MetricStructure* describes if the object represents a single measurement or multiple

related measurements (an invasive blood pressure could be compound when it represents

a pulsatile pressure like ABP and derives systolic, diastolic, mean values)

```
typedef struct MetricStructure {
    u_8             ms_struct;
    u_8             ms_comp_no;
} MetricStructure;
```

ms_struct describes the structure of the object, 0 means simple, 1 means compound object.

ms_comp_no contains the maximum number of components in the compound, it contains 0 for simple objects.

The MetricRelevance is a 16 bit wide field for internal use only.

```
typedef u_16 MetricRelevance;
```

Attribute Groups

The attributes of the Numeric object are arranged in the following attribute groups:

```
Attribute Group:   VMO Static Context Group
Group ID:          NOM_ATTR_GRP_VMO_STATIC
Description:       Static context of the object
Attributes:        Type, Handle, Metric Specification

Attribute Group:   VMO Dynamic Context Group
Group ID:          NOM_ATTR_GRP_VMO_DYN
Description:       Dynamic context of the object
Attributes:        Label, Label String, Color, Display Resolution

Attribute Group:   Metric Observed Value Group
Group ID:          NOM_ATTR_GRP_METRIC_VAL_OBS
Description:       Observed values of the object
Attributes:        Nu Observed Value,
                   Compound Nu Observed Value,
                   Absolute Time Stamp, Relative Time Stamp
```

Dynamic Context Changes

Internally, the IntelliVue monitor uses two different communication channels for attributes from the VMO Dynamic Context Group and the Metric Observed Value Group. This can lead to possible inconsistencies between these two attribute groups. Imagine that a Computer Client is polling all attribute groups. If the user changes the Label of a numeric (VMO Dynamic Context Group), the *physio_id* in the Nu Observed Value (Metric Observed Value Group) may be updated a short period later.

For real-time Numerics, this inconsistency is typically resolved after less than one second with the periodic update of the Observed Values. For averaged Numerics, the update of the Observed Values depends on the averaging period. It may be 12 seconds, 1 minute or 5 minutes.

Wave Objects

Wave Object Attributes

This section defines the attributes of the Wave object, together with the attribute identifier codes and attribute data types.

Attribute: Handle

The Handle attribute contains an identification of the wave object in the form of a numeric value. The actual value of the Handle attribute does not have a meaning. It is used for reference and relation purposes.

```
Attribute ID:          NOM_ATTR_ID_HANDLE
Attribute Type:        Handle (see Definitions Shared by Protocols)
Attribute Groups:      VMO Static Context Group
Availability:          Mandatory
```

Attribute: Type

The Type attribute contains an identification of the object type.

```
Attribute ID:          NOM_ATTR_ID_TYPE
Attribute Type:        TYPE (see Definitions Shared by Protocols)
Attribute Groups:      VMO Static Context Group
Availability:          Mandatory
```

Attribute: Metric Specification

The Metric Specification describes static properties of a metric object.

```
Attribute ID:      NOM_ATTR_METRIC_SPECN
Attribute Type:    MetricSpec
Attribute Groups:  VMO Static Context Group
Availability:      Mandatory
```

The *MetricSpec* is defined as follows:

```
typedef struct {
       RelativeTime          update_period;
       MetricCategory        category;
       MetricAccess          access;
       MetricStructure       structure;
       MetricRelevance       relevance;
} MetricSpec;
```

The *update_period* specifies the time between observed values.

MetricCategory, *MetricAccess*, *MetricStructure*, and *MetricRelevance* are already defined for the Numeric object.

Attribute: Sample Array Specification

The Sample Array Specification describes static properties of a wave object.

```
Attribute ID:          NOM_ATTR_SA_SPECN
Attribute Type:        SaSpec
Attribute Groups:      VMO Static Context Group
Availability:          Mandatory
```

The *SaSpec* is defined as follows:

```
typedef struct {
       u_16                  array_size;
       SampleType            sample_type;
       SaFlags               flags;
} SaSpec;
```

The *array_size* specifies the maximum number of samples in one observed value.

The *SampleType* is defined as follows:

```
typedef struct {
       u_8                   sample_size;
       u_8                   significant_bits;
} SampleType;
```

The *sample_size* specifies the number of bits used to encode one wave sample.

The number of *significant_bits* is less or equal *sample_size*. To get the actual sample value, non-significant bits must be masked if indicated in the flags value.

The *SaFlags* is defined as follows:

```
typedef u_16          SaFlags;
#define SMOOTH_CURVE                0x8000
#define DELAYED_CURVE               0x4000
#define STATIC_SCALE                0x2000
#define SA_EXT_VAL_RANGE            0x1000
```

The values have the following meaning:

SMOOTH_CURVE, DELAYED_CURVE : used for wave presentation
STATIC_SCALE: Scale and range specification does not change.
SA_EXT_VAL_RANGE: The non-significant bits in the sample value must be masked.

Attribute: Sample Array Fixed Value Specification

The Sample Array Fixed Value Specification defines a list of fixed sample values or bit masks that indicate specific conditions.

```
Attribute ID:          NOM_ATTR_SA_FIXED_VAL_SPECN
Attribute Type:        SaFixedValSpec16
Attribute Groups:      VMO Static Context Group
Availability:          Optional
```

The *SaFixedValSpec16* is a sequence of *SaFixedValSpecEntry16* elements:

```
typedef struct {
       u_16                    count;
       u_16                    length;
       SaFixedValSpecEntry16   value[1];
} SaFixedValSpec16;
typedef struct {
       SaFixedValId            sa_fixed_val_id;
       u_16                    sa_fixed_val;
} SaFixedValSpecEntry16;
```

The *SaFixedValId* is defined as follows:

```
typedef u_16           SaFixedValId;
#define SA_FIX_UNSPEC                  0
#define SA_FIX_INVALID_MASK            1
#define SA_FIX_PACER_MASK              2
#define SA_FIX_DEFIB_MARKER_MASK       3
#define SA_FIX_SATURATION              4
#define SA_FIX_QRS_MASK                5
```

The values have the following meaning:

SA_FIX_UNSPEC: Not specified.
SA_FIX_INVALID_MASK: Invalid sample mask.
SA_FIX_PACER_MASK: Pace pulse detected.
SA_FIX_DEFIB_MARKER_MASK: Defib marker in this sample.
SA_FIX_SATURATION: Indicates saturation condition in this sample.
(Note: despite the name, this is a mask as well.)
SA_FIX_QRS_MASK: Indicates QRS trigger around this sample.

The *sa_fixed_val* may be a value or a bit mask, as indicated in the *sa_fixed_val_id*.

Attribute: Sample Period

The Sample Period specifies the sample rate.

```
Attribute ID:          NOM_ATTR_TIME_PD_SAMP
Attribute Type:        RelativeTime (see Definitions Shared by Protocols)
Attribute Groups:      VMO Static Context Group
Availability:          Mandatory
```

Attribute: Label

The Label attribute contains a 32 bit wide ID which represents the wave label string. The Label is unique for all waves in the system.

```
Attribute ID:          NOM_ATTR_ID_LABEL
Attribute Type:        TextId (see Definitions Shared by Protocols)
Attribute Groups:      VMO Dynamic Context Group
Availability:          Optional
```

Attribute: Label String

The Label String is a unicode string which contains the label string for a wave.

```
Attribute ID:          NOM_ATTR_ID_LABEL_STRING
Attribute Type:        String (see Definitions Shared by Protocols)
Attribute Groups:      VMO Dynamic Context Group
Availability:          Optional
```

Attribute: Metric State

The Metric State attribute indicates metric on or off state.

```
Attribute ID:          NOM_ATTR_METRIC_STAT
Attribute Type:        MetricState
Attribute Groups:      VMO Dynamic Context Group
Availability:          Optional
```

The *MetricState* is a bit field defined as follows:

```
typedef u_16           MetricState;
#define METRIC_OFF                    0x8000
```

Attribute: Unit Code

The Unit Code attribute contains a nomenclature code from the dimension partition. It identifies the units of measure.

```
Attribute ID:          NOM_ATTR_UNIT_CODE
Attribute Type:        OIDType (see Definitions Shared by Protocols)
Attribute Groups:      VMO Dynamic Context Group
Availability:          Optional
```

Attribute: Color

The Color attribute describes the color in which a wave is displayed on the screen.

```
Attribute ID:          NOM_ATTR_COLOR
Attribute Type:        SimpleColour
Attribute Groups:      VMO Dynamic Context Group
Availability:          Optional
```

The *SimpleColour* is already defined for the Numeric object.

Attribute: Measure Mode

The Measure Mode attribute defines specific measurement modes.

```
Attribute ID:          NOM_ATTR_MODE_MSMT
Attribute Type:        MeasureMode
Attribute Groups:      VMO Dynamic Context Group
Availability:          Optional
```

For wave objects, the following *MeasureMode* bits are defined:

```
typedef u_16           MeasureMode;
#define CO2_SIDESTREAM                0x0400
#define ECG_PACED                     0x0200
#define ECG_NONPACED                  0x0100
#define ECG_DIAG                      0x0080
#define ECG_MONITOR                   0x0040
#define ECG_FILTER                    0x0020
#define ECG_MODE_EASI                 0x0008
#define ECG_LEAD_PRIMARY              0x0004
```

The values have the following meaning:

CO2_SIDESTREAM: CO_2 sidestream.
ECG_PACED, ECG_NONPACED: Paced mode setting.
ECG_DIAG, ECG_MONITOR, ECG_FILTER: ECG filter setting.
ECG_MODE_EASI: EASI derived lead.
ECG_LEAD_PRIMARY: ECG primary lead.

Attribute: Metric Info Label

The Metric Info Label allows to specify an additional dynamic text (32 bit ID).

```
Attribute ID:          NOM_ATTR_METRIC_INFO_LABEL
Attribute Type:        TextId (see Definitions Shared by Protocols)
Attribute Groups:      VMO Dynamic Context Group
Availability:          Optional
```

Attribute: Metric Info Label String

The Metric Info Label String allows to specify an additional dynamic text (unicode string).

```
Attribute ID:          NOM_ATTR_METRIC_INFO_LABEL_STR
Attribute Type:        String (see Definitions Shared by Protocols)
Attribute Groups:      VMO Dynamic Context Group
Availability:          Optional
```

Attribute: Scale and Range Specification

The Scale and Range Specification describes a relation between scaled values and absolute values and also defines the range of the measured values and samples.

```
Attribute ID:          NOM_ATTR_SCALE_SPECN_I16
Attribute Type:        ScaleRangeSpec16
Attribute Groups:      VMO Dynamic Context Group
Availability:          Mandatory
```

The *ScaleRangeSpec16* is defined as follows:

```
typedef struct {
       FLOATType             lower_absolute_value;
       FLOATType             upper_absolute_value;
       u_16                  lower_scaled_value;
       u_16                  upper_scaled_value;
} ScaleRangeSpec16;
```

The scaled values refer to the wave samples in the observed values.

If the wave does not represent any absolute value, the absolute value fields must be *NaN* (Not a Number).

Attribute: Sample Array Physiological Range

The Sample Array Physiological Range is used for display scaling.

```
Attribute ID:          NOM_ATTR_SA_RANGE_PHYS_I16
Attribute Type:        ScaledRange16
Attribute Groups:      VMO Dynamic Context Group
Availability:          Optional
```

The *ScaledRange16* is defined as follows:

```
typedef struct {
       u_16                  lower_scaled_value;
       u_16                  upper_scaled_value;
} ScaledRange16;
```

Attribute: Visual Grid

The Visual Grid attribute allows to define grid lines.

```
Attribute ID:          NOM_ATTR_GRID_VIS_I16
Attribute Type:        SaVisualGrid16
Attribute Groups:      VMO Dynamic Context Group
Availability:          Optional
```

The *SaVisualGrid16* is defined as follows:

```
typedef struct {
        u_16                    count;
        u_16                    length;
        SaGridEntry16           value[1];
} SaVisualGrid16;
typedef struct {
        FLOATType               absolute_value;
        u_16                    scaled_value;
        u_16                    level;
} SaGridEntry16;
```

Different *levels* define relative importance of grid lines. *0* is the first (most important) level.

Attribute: Sample Array Calibration Specification

The Sample Array Calibration Specification allows to define the presence of a calibration bar or calibration stair.

```
Attribute ID:          NOM_ATTR_SA_CALIB_I16
Attribute Type:        SaCalData16
Attribute Groups:      VMO Dynamic Context Group
Availability:          Optional
```

The *SaCalData16* is defined as follows:

```
typedef struct {
        FLOATType               lower_absolute_value;
        FLOATType               upper_absolute_value;
        u_16                    lower_scaled_value;
        u_16                    upper_scaled_value;
        u_16                    increment;
        u_16                    cal_type;
#define BAR                             0
#define STAIR                           1
} SaCalData16;
```

Attribute: Sample Array Observed Value

The Sample Array Observed Value attribute represents the wave samples, along with state and identification data.

```
Attribute ID:          NOM_ATTR_SA_VAL_OBS
Attribute Type:        SaObsValue
Attribute Groups:      Metric Observed Value Group
Availability:          Conditional (either SaObsValue or
                       SaObsValueCmp is present)
```

The *SaObsValue* data type is defined as follows:

```
typedef struct {
        OIDType                 physio_id;
        MeasurementState        state;
        struct {
                u_16                    length;
                u_8                     value[1];
```

```
        } array;
} SaObsValue;
```

The *physio_id* (physiological identifier) field contains a nomenclature code from the SCADA partition that identifies the represented wave (typically a physiological measurement).

The *state* indicates measurement validity. Refer to the Numeric object for a definition of the bit field. The measurement is valid if the first octet of the *state* is all *0*.

Attribute: Compound Sample Array Observed Value

The Compound Sample Array Observed Value attribute represents multiple waves modelled in one Wave object, along with state and identification data.

Compound Sample Array Observed Values are used to provide 250 samples/s ECG waves with common context.

```
Attribute ID:          NOM_ATTR_SA_CMPD_VAL_OBS
Attribute Type:        SaObsValueCmp
Attribute Groups:      Metric Observed Value Group
Availability:          Conditional (either SaObsValue or
                       SaObsValueCmp is present)
```

The *SaObsValueCmp* data type is defined as follows:

```
typedef struct {
        u_16                count;
        u_16                length;
        SaObsValue          value[1];
} SaObsValueCmp;
```

The *count* field defines the number of *SaObsValue* elements in the structure.

The *length* field defines the size of the array of *SaObsValue* structures in bytes.

The *SaObsValue* data type is defined above. The elements in a compound observed value can be identified by their *physio_id*.

Attributes Groups

The attributes of the Wave object are arranged in the following attribute groups:

```
Attribute Group:       VMO Static Context Group
Group ID:              NOM_ATTR_GRP_VMO_STATIC
Description:           Static context of the object
Attributes:            Handle, Type, Metric Specification, Sample Array
                       Specification, Sample Array Fixed Value Specification,
                       Sample Period
Attribute Group:       VMO Dynamic Context Group
Group ID:              NOM_ATTR_GRP_VMO_DYN
Description:           Dynamic context of the object
Attributes:            Label, Label String, Metric State, Unit Code,
                       Color, Measure Mode, Metric Info Label, Metric Info
                       Label String, Scale and Range Specification,
                       Sample Array Physiological Range, Visual Grid,
                       Sample Array Calibration Specification
Attribute Group:       Metric Observed Value Group
Group ID:              NOM_ATTR_GRP_METR_VAL_OBS
Description:           Observed values of the object
Attributes:            Sample Array Observed Value, Compound Sample Array
                       Observed Value
```

System Objects

System Objects Attributes

This section defines the attributes of the Medical Device System (MDS) object, together with the attribute identifier codes and attribute data types.

Attribute: Handle

The Handle attribute contains a unique identification of the MDS object in the form of a numeral value. The actual value of the Handle attribute does not have a meaning. It is used for reference and relation purposes.

```
Attribute ID:      NOM_ATTR_ID_HANDLE
Attribute Type:    Handle (see Definitions Shared by Protocols)
Attribute Groups:  -
Availability:      Mandatory
```

Attribute: System Type

The System Type attribute contains an identification of the device type identified with the MDS object (e.g. monitor)

```
Attribute ID:      NOM_ATTR_SYS_TYPE
Attribute Type:    TYPE (see Definitions Shared by Protocols)
Attribute Groups:  System Identification Attribute Group
Availability:      Mandatory
```

For the MDS object, the OBJ nomenclature partition is used. The code value is a static identification.

Attribute: System Model

The System Model attribute contains a manufacturer ID and a manufacturer-specific model number for the device.

```
Attribute ID:      NOM_ATTR_ID_MODEL
Attribute Type:    SystemModel
Attribute Groups:  System Identification Attribute Group
Availability:      Mandatory
```

The *SystemModel* is defined as follows:

```
typedef struct  {
    VariableLabel manufacturer;
    VariableLabel model_number;
} SystemModel;
```

The *manufacturer* field is of variable length, hence the offset of *model_number* depends on the length of *manufacturer*. Currently, the IntelliVue monitor uses 4 characters for the *manufacturer* and 6 characters for the *model_number* (including the terminating '\0').

Attribute: System ID

The Sytem ID attribute contains a unique identifier for the device.

```
Attribute ID:      NOM_ATTR_SYS_ID
Attribute Type:    VariableLabel
                   (see Definitions Shared by Protocols)
Attribute Groups:  System Identification Attribute Group
Availability:      Mandatory
```

The IntelliVue monitor uses the 6 byte MAC address as identifier. Future versions might use an 8 byte EUI-64 identifier.

Attribute: Nomenclature Version

The Nomenclature Version attribute contains the version of the nomenclature used by the device.

```
Attribute ID:      NOM_ATTR_NOM_VERS
Attribute Type:    u_32
Attribute Groups:  System Identification Attribute Group
Availability:      Mandatory
```

The Nomenclature Version is composed of 16 bit major and 16 bit minor version number. The IntelliVue monitor currently uses the Nomenclature Version 1.0.

Attribute: System Localization

The System Localization attribute contains information about the language version used by the device.

```
Attribute ID:      NOM_ATTR_LOCALIZN
Attribute Type:    SystemLocal
Attribute Groups:  System Identification Attribute Group
Availability:      Optional
```

The *SystemLocal* is defined as follows:

```
typedef struct {
    u_32            text_catalog_revision;
    Language        language;
    StringFormat    format;
} SystemLocal;
```

The *text_catalog_revision* contains revision information about the texts used by the monitor. The two most significant bytes contain the version of the text catalog (one byte major, one byte minor revision). The text catalog defines the possible values for Attributes of the type *TextId*. A client which depends on a *TextId* having a specific value can use this information for revision control.

The lower two bytes of the *text_catalog_revision* are used for a language revision (one byte major, one byte minor revision). The language revision denotes the mapping from a *TextId* to an actual string in the monitor language.

The *Language* describes the language used by the monitor. It is defined as follows:

```
typedef u_16 Language;
#define LANGUAGE_UNSPEC          0
#define ENGLISH                  1
#define GERMAN                   2
#define FRENCH                   3
#define ITALIAN                  4
#define SPANISH                  5
#define DUTCH                    6
#define SWEDISH                  7
#define FINNISH                  8
#define NORWEG                   9
#define DANISH                   10
#define JAPANESE                 11
#define REP_OF_CHINA             12
#define PEOPLE_REP_CHINA         13
#define PORTUGUESE               14
#define RUSSIAN                  15
#define BYELORUSSIAN             16
#define UKRAINIAN                17
#define CROATIAN                 18
#define SERBIAN                  19
#define MACEDONIAN               20
#define BULGARIAN                21
#define GREEK                    22
#define POLISH                   23
#define CZECH                    24
#define SLOVAK                   25
#define SLOVENIAN                26
#define HUNGARIAN                27
#define ROMANIAN                 28
#define TURKISH                  29
#define LATVIAN                  30
#define LITHUANIAN               31
#define ESTONIAN                 32
#define KOREAN                   33
```

The *StringFormat* describes how strings are encoded. The IntelliVue monitor uses unicode encoding.

```
typedef u_16 StringFormat;
#define STRFMT_UNICODE_NT        11
```

Attribute: System Specification

The System Specification attribute contains a set of functional components supported by the system.

```
Attribute ID:      NOM_ATTR_SYS_SPECN
Attribute Type:    SystemSpec
Attribute Groups:  System Application Attribute Group
Availability:      Optional
```

The *SystemSpec* is defined as follows:

```
typedef struct {
    u_16            count;
    u_16            length;
    SystemSpecEntry value[1];
} SystemSpec;

typedef struct {
    PrivateOid      component_capab_id;
    u_16            length;
    u_16            value[1];
} SystemSpecEntry;
```

The supported components are:

```
Component ID:      NOM_MDIB_OBJ_SUPPORT
Component Type:    MdibObjectSupport
Availability:      Mandatory
```

The *MdibObjectSupport* is defined as follows:

```
typedef struct {
    u_16    count;
    u_16    length;
    MdibObjectSupportEntry        value[1];
} MdibObjectSupport;

typedef struct {
    TYPE    object_type;
    u_32    max_inst;
} MdibObjectSupportEntry;
```

The *MdibObjextSupport* contains a list of all object classes supported by the system and the maximum number of instances per class. If *max_inst* contains 0xffffffff, it is not defined.

Attribute: Mds General System Info

The Mds General System Info attribute contains global information about the monitor and its configuration.

```
Attribute ID:       NOM_ATTR_MDS_GEN_INFO
Attribute Type:     MdsGenSystemInfo
Attribute Group:    System Application Attribute Group
Availability:       Optional
```

The *MdsGenSystemInfo* is defined as follows:

```
typedef struct
{
    u_16            count;
    u_16            length;
    MdsGenSystemInfoEntry value[1];
} MdsGenSystemInfo;
```

The *MdsGenSysemInfoEntry* allows to encode generic system information. It has the following structure:

```
typedef struct
{
    u_16            choice;
#define MDS_GEN_SYSTEM_INFO_SYSTEM_PULSE_CHOSEN         1
    u_16            length;
    u_8             value[1]; /* placeholder for appended data */
} MdsGenSystemInfoEntry;
```

One MdsGenSytemInfoEntry is used to encode the System Pulse information. The monitor can generate a pulse rate from several sources.

```
Choice:         MDS_GEN_SYSTEM_INFO_SYSTEM_PULSE_CHOSEN 1
Type:           SystemPulseInfo
Availability:   Optional
```

The *SystemPulseInfo* is defined as follows:

```
typedef struct
{
    ManagedObjectId system_pulse;
    ManagedObjectId alarm_source;
} SystemPulseInfo;
```

It enfolds the *ManagedObjectId*s of the object instances selected as system-pulse respectively alarm-source.

Attribute: Production Specification

The Production Specification attribute contains a list of component revisions and serial numbers within the system.

```
Attribute ID:      NOM_ATTR_ID_PROD_SPECN
Attribute Type:    ProductionSpec
Attribute Groups:  System Production Attribute Group
Availability:      Optional
```

The *ProductionSpec* is defined as follows:

```
typedef struct {
    u_16            count;
    u_16            length;
    ProdSpecEntry   value[1];
} ProductionSpec;

typedef struct {
    u_16           spec_type;
#define UNSPECIFIED       0
#define SERIAL_NUMBER     1
#define PART_NUMBER       2
#define HW_REVISION       3
#define SW_REVISION       4
#define FW_REVISION       5
#define PROTOCOL_REVISION 6
    PrivateOid     component_id;
    VariableLabel  prod_spec;
} ProdSpecEntry;
```

The current IntelliVue monitor uses 10 characters for a serial number, 14 characters for part numbers and 8 characters for revision strings. The strings are not null-terminated.

The supported components are:

```
Component ID:      ID_COMP_PRODUCT
Description:       Overall product specification

Component ID:      ID_COMP_CONFIG
Description:       Specific system configuration

Component ID:      ID_COMP_BOOT
Description:       Boot code specification

Component ID:      ID_COMP_MAIN_BD
Description:       Mainboard hardware specification

Component ID:      ID_COMP_APPL_SW
Description:       Application software specification
```

See the section “Component IDs” on page 7-170 for the values of the *component_id*. The *ProductionSpec* may contain additional private entries.

Attribute: MDS Status

The MDS Status attribute describes the device state.

```
Attribute ID:      NOM_ATTR_VMS_MDS_STAT
Attribute Type:    MDSStatus
Attribute Groups:  System Application Attribute Group
Availability:      Mandatory
```

The *MDSStatus* is defined as follows:

```
typedef u_16        MDSStatus;
#define    DISCONNECTED   0
#define    UNASSOCIATED   1
#define    OPERATING      6
```

The *MDSStatus* values have the following meaning:

DISCONNECTED: The IntelliVue monitor is not connected to the network.

UNASSOCIATED: The IntelliVue monitor is connected to the network, but no association is currently active.

OPERATING: The IntelliVue monitor has an association with a Computer Client.

Currently, a Computer Client will only see the MDS Status OPERATING, if the MDS has another Status, there is no association with a Computer Client.

Attribute: Bed Label

The Bed Label attribute contains a printable string identifying the system location.

```
Attribute ID:      NOM_ATTR_ID_BED_LABEL
Attribute Type:    String
                   (see Definitions Shared by Protocols)
Attribute Groups:  System Application Attribute Group
Availability:      Optional
```

The Bed Label can be entered in the Admit/Discharge dialog. It uses 16 bit unicode character encoding. Currently, the Bed Label is 17 characters (including terminating '\0'). If the actual label is shorter, the string is filled with '\0' characters.

Attribute: Operating Mode

The Operating Mode attribute identifies the current operating mode of the device.

```
Attribute ID:      NOM_ATTR_MODE_OP
Attribute Type:    PrivateOID
Attribute Groups:  System Application Attribute Group
Availability:      Optional
```

The Operating Mode is defined as a bit field. The following mode bits are defined:

```
#define    OPMODE_UNSPEC  0x8000
#define    MONITORING     0x4000
#define    DEMO           0x2000
#define    SERVICE        0x1000
#define    OPMODE_STANDBY 0x0002
#define    CONFIG         0x0001
```

The values have the following meaning:

OPMODE_UNSPEC: The Operating Mode is not specified.

MONITORING: Device is configured to monitor patient data (the default mode).

DEMO: Demonstration Mode with simulated patient data.

SERVICE: Device is in Service Mode.

STANDBY: Standby and Power Safe Mode.

CONFIG: Device is in Configuration Mode.

Exactly one of the bit out of the bits 0 - 4 must be set, bits 14 and 15 (the stand-by and config mode bits) can be set optionally.

Attribute: Application Area

The Application Area attribute describes the intended application area for the device.

```
Attribute ID:      NOM_ATTR_AREA_APPL
Attribute Type:    ApplicationArea
Attribute Groups:  System Application Attribute Group
Availability:      Optional
```

The *ApplicationArea* is defined as follows:

```
typedef u_16            ApplicationArea;
#define    AREA_UNSPEC            0
#define    AREA_OPERATING_ROOM    1
#define    AREA_INTENSIVE_CARE    2
#define    AREA_NEONATAL_INTENSIVE_CARE 3
#define    AREA_CARDIOLOGY_CARE   4
```

The values have the following meaning:

AREA_UNSPEC: The application area has not been specified.

AREA_OPERATING_ROOM: The application area has been specified as an operating room.

AREA_INTENSIVE_CARE: The application area has been specified as an intensive care unit.

AREA_NEONATAL_INTENSIVE_CARE: The application area has been specified as a neonatal intensive care unit.

AREA_CARDIOLOGY_CARE: The application area has been specified as a cardiology care unit.

Attribute: Date and Time

The Date and Time attribute contains the current device time.

```
Attribute ID:      NOM_ATTR_TIME_ABS
Attribute Type:    AbsoluteTime
                   (see Definitions Shared by Protocols)
Attribute Groups:  System Application Attribute Group
Availability:      Optional
```

Attribute: Relative Time

The Relative Time attribute contains the current device relative time.

```
Attribute ID:      NOM_ATTR_TIME_REL
Attribute Type:    RelativeTime
                   (see Definitions Shared by Protocols)
Attribute Groups:  System Application Attribute Group
Availability:      Optional
```

The Relative Time is set to zero after each power cycle.

Attribute: Altitude

The Altitude attribute contains the system altitude above or below sea level.

```
Attribute ID:      NOM_ATTR_ALTITUDE
Attribute Type:    i_16
Attribute Groups:  System Application Attribute Group
Availability:      Optional
```

Attribute: Line Frequency

The Line Frequency attribute describes the frequency of the main power supply in Hz.

```
Attribute ID:      NOM_ATTR_LINE_FREQ
Attribute Type:    LineFrequency
Attribute Groups:  System Application Attribute Group
Availability:      Optional
```

The *LineFrequency* is defined as follows:

```
typedef u_16            LineFrequency;
#define    LINE_F_UNSPEC  0
#define    LINE_F_50HZ    1
#define    LINE_F_60HZ    2
```

Attribute: Association Invoke ID

The Association Invoke ID attribute is a counter for the number of associations. It is incremented with each new association.

```
Attribute ID:      NOM_ATTR_ID_ASSOC_NO
Attribute Type:    u_16
Attribute Groups:  System Identification Attribute Group
Availability:      Optional
```

Attribute Groups

The attributes of the Medical Device System object are arranged in the following attribute groups:

```
Attribute Group:   System Identification Attribute Group
Group ID:          NOM_ATTR_GRP_SYS_ID
Description:       Identification of the system
Attributes:        System Type, System Model, System Id,
                   Nomenclature Version, System Localization, Association Invoke Id

Attribute Group:   System Application Attribute Group
Group ID:          NOM_ATTR_GRP_SYS_APPL
Description:       System Capabilities and Settings
Attributes:        System Specification, MDS Status, Bed Label,
                   Operating Mode, Application Area, Data and
                   Time, Relative Time, Altitude, Line
                   Frequency, Mds General System Info

Attribute Group:   System Production Attribute Group
Group ID:          NOM_ATTR_GRP_SYS_PROD
Description:       HW and SW configuration
Attributes:        Production Specification
```

Alert Monitor Object

Attributes of the Alert Monitor Object

This section defines the attributes of the Alert Monitor object, together with the attribute identifier codes and attribute data types.

The Alert Monitor object represents the overall device alert condition. It contains a global alert status and a list of active technical and patient alerts.

Attribute: Handle

The Handle attribute contains a unique identification of the Alert Monitor object in the form of a numeral value. The actual value of the Handle attribute does not have a meaning. It is used for reference and relation purposes.

```
Attribute ID:      NOM_ATTR_ID_HANDLE
Attribute Type:    Handle (see Definitions Shared by Protocols)
Attribute Groups:  VMO Static Context Group
Availability:      Mandatory
```

Attribute: Type

The Type attribute contains an identification of the object type represented by the Alert Monitor.

```
Attribute ID:      NOM_ATTR_ID_TYPE
Attribute Type:    TYPE (see Definitions Shared by Protocols)
Attribute Groups:  VMO Static Context Group
Availability:      Mandatory
```

Attribute: Device Alert Condition

The Device Alert Condition attribute contains global device alert status information.

```
Attribute ID:      NOM_ATTR_DEV_AL_COND
Attribute Type:    DeviceAlertCondition
Attribute Groups:  Alert Monitor Group
Availability:      Mandatory
```

The *DeviceAlertCondtion* is defined as follows:

```
typedef struct {
    AlertState     device_alert_state;
    u_16           al_stat_chg_cnt;
    AlertType      max_p_alarm;
    AlertType      max_t_alarm;
    AlertType      max_aud_alarm;
} DeviceAlertCondition;
```

The *AlertState* is a bit field defined as follows:

```
typedef u_16 AlertState;
#define AL_INHIBITED              0x8000
#define AL_SUSPENDED              0x4000
#define AL_LATCHED                0x2000
#define AL_SILENCED_RESET         0x1000
#define AL_DEV_IN_TEST_MODE       0x0400
#define AL_DEV_IN_STANDBY         0x0200
#define AL_DEV_IN_DEMO_MODE       0x0100
#define AL_NEW_ALERT              0x0008
```

The *AlertState* is used for the overall device alert state and for the specific state of each alert. The bits in *AlertState* have the following meaning:

AL_INHIBITED: Alert is switched off.

AL_SUSPENDED: Alert inactivated temporarily, alert condition is acknowledged.

AL_LATCHED: Alert condition is not active but latched, note that technical alarms are never latching.

AL_SILENCED_RESET: Alert condition stopped but alarming re-enabled (only for *DeviceAlertCondition*).

AL_DEV_IN_TEST_MODE: Device is in a temporary test mode.

AL_DEV_IN_STANDBY: Device is in standby mode.

AL_DEV_IN_DEMO_MODE: Indicates that the device is in demo mode.

AL_NEW_ALERT: Indicate a new alarm (not in *DeviceAlertCondition*). A Computer Client might not see this bit if it does not poll fast enough or other delays occur.

The *al_stat_chg_cnt* is an internal change counter. A Computer Client should not interpret this field, because it can not be guaranteed that no internal message is missed.

The *AlertType* is a bit field defined as follows:

```
typedef u_16 AlertType;
#define    NO_ALERT       0
#define    LOW_PRI_T_AL   1
#define    MED_PRI_T_AL   2
#define    HI_PRI_T_AL    4
#define    LOW_PRI_P_AL   256
#define    MED_PRI_P_AL   512
#define    HI_PRI_P_AL    1024
```

The bits have the following meaning:

NO_ALERT: No alert active.

LOW_PRI_T_AL: Low priority technical alarm (soft inop). These inops are generated after a signal analysis (e.g “Noisy ECG”).

MED_PRI_T_AL: Medium priority technical alarm (hard inop). These inops are generated during inoperable parameter measurement because of hardware faults or no transducer connected (e.g “Leads Off”, “ABP No Transducer”)

HI_PRI_T_AL: High priority technical alarm (severe inop).

LOW_PRI_P_AL: Awareness Condition (short yellow alarm): These alarms are marked with a “**” in the alarm string and a specific short yellow alarm sound is issued. Today short yellow alarms are generated only from arrhythmia computer.

MED_PRI_P_AL: Medium priority patient alarm (yellow alarm): These alarms are marked with a “**” in the alarm string. They indicate a less critical patient condition usually due to violation of user defined criteria (e.g. limit violation alarm).

HI_PRI_P_AL: High priority patient alarm (red alarm): These alarms are marked with a "***" in the alarm string. These alarms indicate a life threatening patient condition.

Attribute: Device T-Alarm List

The Device T-Alarm List attribute contains the active technical alarms (inops) in the system.

```
Attribute ID:      NOM_ATTR_AL_MON_T_AL_LIST
Attribute Type:    DevAlarmList
Attribute Groups:  Alert Monitor Group
Availability:      Mandatory
```

The *DevAlarmList* is defined as follows:

```
typedef struct {
    u_16           count;
    u_16           length;
    DevAlarmEntry  value[1];
} DevAlarmList;

typedef struct {
    OIDType                al_source;
    OIDType                al_code;
    AlertType              al_type;
    AlertState             al_state;
    ManagedObjectId        object;
    PrivateOid             alert_info_id;
#define GEN_ALMON_INFO 513
#define STR_ALMON_INFO 516
    u_16                   length;
} DevAlarmEntry;
```

The *al_source* is taken from the Object Oriented or the SCADA partition (depending on *al_code*). It identifies the origin of the alert (e.g. temperature).

The *al_code* is taken from the Events partition and describes the reason for the alert (e.g. high alarm). The least significant bit is used to define the nomenclature partition for *al_source*. Last bit 0 means SCADA partition, last bit 1 means Object Oriented partition.

The definitions for *AlertType* and *AlertState* can be found in the paragraph about the Device Alert Condition.

The *object* field contains a reference to the object which generated the alert. The object may not be known to the Computer Client, if the Data Export protocol does not allow accessing the specific object.

If the *alert_info_id* is set to GEN_ALMON_INFO, an *AlMonGenInfo* structure is appended:

```
typedef struct {
    u_16                   al_inst_no;
    TextId                 al_text;
    AlertPriority          priority;
    AlertFlags             flags;
} AlMonGenInfo;
```

If the *alert_info_id* is set to STR_ALMON_INFO, an *StrAlMonInfo* structure is appended:

```
typedef struct {
    u_16                   al_inst_no;
    TextId                 al_text;
    AlertPriority          priority;
    AlertFlags             flags;
    String                 string;
} StrAlMonInfo;
```

Currently, the IntelliVue monitor only supports the *StrAlMonInfo* data type.

The *al_inst_no* is a private ID.

The *al_text* is a private ID.

The *AlertPriority* is defined as follows:

```
typedef u_16       AlertPriority;
```

The *AlertPriority* only allows prioritization within a group of alarms. A Computer Client application should use the *AlertType* to distinguish low and high priority alarms.

The *AlertFlags* type is defined as follows:

```
typedef u_16            AlertFlags;
#define BEDSIDE_AUDIBLE          0x4000
#define CENTRAL_AUDIBLE          0x2000
#define VISUAL_LATCHING          0x1000
#define AUDIBLE_LATCHING         0x0800
#define SHORT_YELLOW_EXTENSION   0x0400
#define DERIVED                  0x0200
```

The bits in the *AlertFlag* have the following meaning:

BEDSIDE_AUDIBLE: Alert sound at the bedside

CENTRAL_AUDIBLE: Alert sound at the central station

VISUAL_LATCHING: Alert is visible after the alarm condition has ceased. The alarm indication will exist until a specific action is taken by a user (e.g. Silence/Reset).

AUDIBLE_LATCHING: Alert is sound issued after the alarm condition has ceased. The alarm indication will exist until a specific action is taken by a user (e.g. Silence/Reset).

SHORT_YELLOW_EXTENSION: Alarm is not active but artificially extended for short yellow behavior.

DERIVED: Derived alarm.

The *String* contains the a description of the alarm in the language supported by the IntelliVue monitor. *Strings* for patient alarms are prefixed with two "**" or three "***" alarm stars (see “Definitions Shared by Protocols” on page 5-25 for UNICODE character encoding). Currently, the String is 19 characters long, including the terminating ’\0’.

Attribute: Device P-Alarm List

The Device P-Alarm List attribute contains the active patient alarm in the system.

```
Attribute ID:      NOM_ATTR_AL_MON_P_AL_LIST
Attribute Type:    DevAlarmList
Attribute Groups:  Alert Monitor Group
Availability:      Mandatory
```

The *DevAlarmList* data type is the same as for the Device T-Alarm List.

The data in a Device T-Alarm List or Device P-Alarm List might be too large to fit in a single message. In this case the Remote Operation Linked Result message will be used (see “Remote Operation Linked Result” on page 5-34). In this case each message will contain a correctly formatted Alarm list and the Computer Client must merge the lists to get the complete Device T-Alarm List or Device P-Alarm List.

Attribute Groups

The attributes of the Alert Monitor object are arranged in the following attribute groups:

```
Attribute Group:   VMO Static Context Group
Group ID:          NOM_ATTR_GRP_VMO_STATIC
Description:       Static context of the object
Attributes:        TYPE, Handle

Attribute Group:   Alert Monitor Group
Group ID:          NOM_ATTR_GRP_AL_MON
Description:       Alarm related attributes
Attributes:        Device Alert Condition, Device P-Alarm List,
                   Device T-Alarm List
```

Patient Demographics Object

Attributes of the Patient Demographic Object

This section defines the attributes of the Patient Demographics object, together with the attribute identifier codes and attribute data types.

The Patient Demographics object contains the patient information present in the system.

Attribute: Handle

The Handle attribute contains a unique identification of the Patient Demographics object in the form of a numeral value. The actual value of the Handle attribute does not have a meaning. It is used for reference and relation purposes.

```
Attribute ID:      NOM_ATTR_ID_HANDLE
Attribute Type:    Handle (see Definitions Shared by Protocols)
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Mandatory
```

Attribute: Pat Demo State

The Pat Demo State attribute describes the current state of the Patient Demographics object.

```
Attribute ID:      NOM_ATTR_PT_DEMOG_ST
Attribute Type:    PatDemoState
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Mandatory
```

The *PatDemoState* is defined as follows:

```
typedef u_16       PatDmgState;
#define    EMPTY          0
#define    PRE_ADMITTED   1
#define    ADMITTED       2
#define    DISCHARGED     8
```

The values have the following meaning:

EMPTY: No patient information present.

PRE_ADMITTED: Currently not used.

ADMITTED: Patient information is present and valid.

DISCHARGED: Data is still available, but patient is no longer assigned to device.

Attribute: Patient Type

The Patient Type attribute describes the type of patient admitted to the system.

```
Attribute ID:      NOM_ATTR_PT_TYPE
Attribute Type:    PatientType
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

The Patient Type is defined as follows:

```
typedef u_16       PatientType;
#define    PAT_TYPE_UNSPECIFIED   0
#define    ADULT                  1
#define    PEDIATRIC              2
#define    NEONATAL               3
```

The Patient Type can be set by the user in the Admit/Discharge dialog (Patient Cat.).

Attribute: Patient Paced Mode

The Patient Paced Mode attribute indicates whether the patient is paced or not.

```
Attribute ID:      NOM_ATTR_PT_PACED_MOD
Attribute Type:    PatPacedMode
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

The *PatPacedMode* is defined as follows:

```
typedef u_16       PatPacedMode;
#define    PAT_NOT_PACED          0
#define    PAT_PACED_GEN          1
```

Values greater one are reserved to indicate special paced modes. The Computer Client should test for "== 0" or "!= 0".

Attribute: Given Name

The Given Name attribute contains the first name of the patient.

```
Attribute ID:      NOM_ATTR_PT_NAME_GIVEN
Attribute Type:    String
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

Currently, the Given Name can be up to 19 characters long, including the terminating '\0'.

Attribute: Family Name

The Family Name attribute contains the last name of the patient.

```
Attribute ID:      NOM_ATTR_PT_NAME_FAMILY
Attribute Type:    String
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

Currently, the Family Name can be up to 19 characters long, including terminating '\0'.

Attribute: Patient ID

The Patient ID attribute contains the ID of the patient.

```
Attribute ID:      NOM_ATTR_PT_ID
Attribute Type:    String
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

Currently, the Patient ID (Medical Record Number - MRN) can be up to 17 characters long, including the terminating '\0'.

Attribute: Patient Sex

The Patient Sex attribute contains the sex of the patient.

```
Attribute ID:      NOM_ATTR_PT_SEX
Attribute Type:    PatientSex
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

The *PatientSex* is described as follows:

```
typedef u_16       PatientSex;
#define    SEX_UNKNOWN           0
#define    MALE                  1
#define    FEMALE                2
#define    SEX_UNSPECIFIED       9
```

The values have the following meaning:

SEX_UNKNOWN: Patient sex is not known

MALE: Patient is male

FEMALE: Patient is female

SEX_UNSPECIFIED: Patient sex is not specified

Attribute: Date of Birth

The Date of Birth attribute contains the Date of Birth of the patient.

```
Attribute ID:      NOM_ATTR_PT_DOB
Attribute Type:    AbsoluteTime
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

Attribute: Patient Height

The Patient Height attribute contains the height of the patient.

```
Attribute ID:      NOM_ATTR_PT_HEIGHT
Attribute Type:    PatMeasure
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

The *PatMeasure* is defined as follows:

```
typedef struct {
    FLOATType        value;
    OIDType          m_unit;
} PatMeasure;
```

The *value* contains the actual value of the attribute and the *m_units* indicates the unit of measurement for the *value*.

Attribute: Patient Weight

The Patient Height attribute contains the weight of the patient.

```
Attribute ID:      NOM_ATTR_PT_WEIGHT
Attribute Type:    PatMeasure
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

Attribute: Patient Age

The Patient Age attribute contains the age of the patient.

```
Attribute ID:      NOM_ATTR_PT_AGE
Attribute Type:    PatMeasure
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

Attribute: Patient BSA

The Patient BSA attribute contains the body surface area of the patient.

```
Attribute ID:      NOM_ATTR_PT_BSA
Attribute Type:    PatMeasure
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

Attribute: Patient BSA Formula

The Patient BSA Formula attribute describes the formula which is used for the calculation of the patient body surface area.

```
Attribute ID:      NOM_ATTR_PT_BSA_FORMULA
Attribute Type:    PatBsaFormula
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

The *PatBsaFormula* is described as follows:

```
typedef u_16       PtBsaFormula;
#define    BSA_FORMULA_UNSPEC     0
#define    BSA_FORMULA_BOYD       1
#define    BSA_FORMULA_DUBOIS     2
```

The values have the following meaning:

BSA_FORMULA_UNSPEC: Formula not specified

BSA_FORMULA_BOYD: BSA calculation according to Boyd

BDA_FORMULA_DUBOIS:: BSA calculation according to Dubois

Attribute: Notes1

The Notes1 attribute provides additional information about the patient.

```
Attribute ID:      NOM_ATTR_PT_NOTES1
Attribute Type:    String
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

Currently, the Notes1 field can be up to 31 characters long, including the terminating '\0'.

Attribute: Notes2

The Notes2 attribute provides additional information about the patient.

```
Attribute ID:      NOM_ATTR_PT_NOTES2
Attribute Type:    String
Attribute Groups:  Patient Demographics Attribute Group
Availability:      Optional
```

Currently, the Notes2 field can be up to 31 characters long, including the terminating '\0'.

Attribute Groups

The attributes of the Patient Demographics object are arranged in the following attribute groups:

```
Attribute Group:   Patient Demographics Attribute Group
Group ID:          NOM_ATTR_GRP_PT_DEMOG
Description:       Attributes containing patient information
Attributes:        all attributes
```

Patient Conflict Handling

The patient information is stored in the monitor, the measurement server and the central station (if present). This can lead to patient conflicts when the patient information in these locations differ. If the IntelliVue monitor detects a patient conflict, it will display a “Patient Selection” window which allows the user to resolve the conflict.

In the case of a patient conflict, the behavior of the Data Export software is as follows:

- If the Patient Type or Patient Paced Mode attribute is different, the data from the measurement server is considered as relevant.
- If the patient is different (devices have been disconnected and a new patient has been admitted), the Patient Type and Patient Paced Mode information from the measurement server is exported. The other attributes are cleared and the Family Name attribute is set to “???”.

Connect Indication Attributes

This section describes the attributes contained in the Connect Indication Message.

Attribute: System Type

The System Type attribute describes the type of the system (e.g. Monitor).

```
Attribute ID:      NOM_ATTR_SYS_TYPE
Attribute Type:    TYPE (see Definitions Shared by Protocols)
Attribute Groups:  -
Availability:      -
```

Attribute: Protocol Support

The Protocol Support contains an entry for each protocol supported on the network interface.

```
Attribute ID:      NOM_ATTR_PCOL_SUPPORT
Attribute Type:    ProtoSupport
Attribute Groups:  -
Availability:      -
```

The *ProtoSupport* is defined as follows:

```
typedef struct {
    u_16            count;
    u_16            length;
    ProtoSupportEntry value[1];
} ProtoSupport;

typedef struct {
    ApplProtoId     appl_proto;
    TransProtoId    trans_proto;
    u_16            port_number;
    ProtoOptions    options;
} ProtoSupportEntry;

typedef u_16 ApplProtoId;
#define    AP_ID_ACSE           1
#define    AP_ID_DATA_OUT       5

typedef u_16 TransProtoId;
#define    TP_ID_UDP            1

typedef u_16            ProtoOptions;
#define    P_OPT_WIRELESS       0x8000
```

The Computer Client should parse the available protocols and search for the AP_ID_DATA_OUT. This entry specifies the port for the Data Export Protocol. The corresponding Association Control Protocol runs on the same port.

The Computer Client must only send requests to the port specified for the Data Export Protocol.

Attribute: System Localization

The System Localization attribute describes the handling of natural language items.

```
Attribute ID:      NOM_ATTR_LOCALIZN
Attribute Type:    SystemLocal
Attribute Groups:  -
Availability:      -
```

The *SystemLocal* is defined as follows:

```
typedef struct {
    u_32            syslocal_revision;
    Language        language;
    StringFormat    format;
} SystemLocal;
```

The *syslocal_revision* contains the revision of the text catalog used for internal texts.

The *Language* describes the language used in any String type. It is defined as follows:

```
typedef u_16            Language;
#define    LANGUAGE_UNSPEC      0
#define    ENGLISH              1
#define    GERMAN               2
#define    FRENCH               3
#define    ITALIAN              4
#define    SPANISH              5
#define    DUTCH                6
#define    SWEDISH              7
#define    FINNISH              8
#define    NORWEG               9
#define    DANISH               10
#define    JAPANESE             11
#define    REP_OF_CHINA         12
```

```
#define     PEOPLE_REP_CHINA        13
#define     PORTUGUESE              14
#define     RUSSIAN                 15
#define     BYELORUSSIAN            16
#define     UKRAINIAN               17
#define     CROATIAN                18
#define     SERBIAN                 19
#define     MACEDONIAN              20
#define     BULGARIAN               21
#define     GREEK                   22
#define     POLISH                  23
#define     CZECH                   24
#define     SLOVAK                  25
#define     SLOVENIAN               26
#define     HUNGARIAN               27
#define     ROMANIAN                28
#define     TURKISH                 29
#define     LATVIAN                 30
#define     LITHUANIAN              31
#define     ESTONIAN                32
#define     KOREAN                  33
```

The *StringFormat* defines the format used for the *String* data type. The IntelliVue monitor uses 16bit Unicode characters.

```
typedef u_16            StringFormat;
#define                     STRFMT_UNICODE_NT 11
```

Attribute: IP Address Information

The IP Address Information attribute identifies the network interface of the IntelliVue monitor.

```
Attribute ID:      NOM_ATTR_NET_ADDR_INFO
Attribute Type:    IpAddressInfo
Attribute Groups:  -
Availability:      -
```

The *IpAddressInfo* is defined as follows:

```
typedef struct IpAddressInfo {
    MACAddress      mac_address;
    IPAddress       ip_address;
    IPAddress       subnet_mask;
} IpAddressInfo;

typedef struct MACAddress {
    u_8             value[6];
} MACAddress;

typedef struct IPAddress {
    u_8             value[4];
} IPAddress;
```

Partition IDs

The following sections contain a list of identifiers which are used within the IntelliVue monitor. Each identifier is unique within a given partition.

```
#define NOM_PART_OBJ                            1
    /* Object Oriented Elements */
#define NOM_PART_SCADA                          2
    /* Physiological Measurements */
#define NOM_PART_EVT                            3
    /* Events for Alerts */
#define NOM_PART_DIM                            4
    /* Units of Measurement */
#define NOM_PART_PGRP                           6
    /* Identification of Parameter Groups */
#define NOM_PART_INFRASTRUCT                    8
    /* Infrastructure Elements */
```

Object Classes

The following IDs identify object types. They are taken from the Object Oriented Elements partition. These objects may be the source of alerts (see “Alert Monitor Object” on page 85).

```
#define NOM_MOC_VMO_VMD                         2
    /* Virtual Medical Device */
#define NOM_MOC_VMO_CHAN                        3
    /* Channel Object, used for grouping objects */
#define NOM_MOC_VMO_METRIC_ENUM                 5
    /* Enumeration Object, status/annotation information */
#define NOM_MOC_VMO_METRIC_NU                   6
    /* Numeric Object */
#define NOM_MOC_VMO_METRIC_SA_RT                9
    /* Real Time Sample Array */
#define NOM_MOC_VMS_MDS                         33
    /* Medical Device Service Object */
#define NOM_MOC_VMS_MDS_COMPOS_SINGLE_BED       35
    /* Composite Single Bed MDS (multiple MDS objects) */
#define NOM_MOC_VMS_MDS_HYD                     36
    /* Hydra Mds with multiple devices (VMDs) */
#define NOM_MOC_VMS_MDS_SIMP                    37
    /* Simple Mds with a single device (VMD) */
#define NOM_MOC_LOG_ERR                         39
    /* Status Log Object */
#define NOM_MOC_BATT                            41
    /* Battery Object */
#define NOM_MOC_PT_DEMOG                        42
    /* Patient Demographics Object */
#define NOM_MOC_VMO_AL_MON                      54
    /* Alert Monitor Object */
#define NOM_MOC_VMO_PMSTORE                     61
    /* Persistent Metric Store Object */
#define NOM_MOC_PRINTER                         74
    /* Printer Object */
#define NOM_MOC_PT_DEMOG_MGR                    75
    /* Patient Demographics Manager Object */
#define NOM_DEV_ANALY_SAT_O2_VMD                4106
    /* SpO2 Device */
#define NOM_DEV_ANALY_PRESS_BLD_VMD             4174
    /* Blood Pressure Device */
#define NOM_DEV_ANALY_RESP_RATE_VMD             4186
    /* Respiration Rate Device */
#define NOM_DEV_ECG_VMD                         4262
    /* ECG Device */
#define NOM_DEV_METER_TEMP_VMD                  4366
    /* Temperature Device */
#define NOM_DEV_MON_PHYSIO_MULTI_PARAM_MDS      4429
    /* Multi Parameter MDS */
#define NOM_DEV_ANALY_SAT_O2_CHAN               4107
    /* SpO2 Channel */
#define NOM_DEV_ANALY_AWAY_MULTI_PARAM_CHAN     4147
    /* Airway Gas Channel */
#define NOM_DEV_ANALY_RESP_RATE_CHAN            4187
    /* Resp Rate Channel */
#define NOM_DEV_ECG_CHAN                        4263
    /* ECG Channel */
#define NOM_DEV_METER_PRESS_BLD_CHAN            4319
    /* Blood Pressure Channel */
#define NOM_DEV_METER_TEMP_CHAN                 4367
    /* Temperature Channel */
#define NOM_DEV_GENERAL_VMD                     5122
    /* general Device */
#define NOM_DEV_GENERAL_CHAN                    5123
```

```
    /* General Channel */
#define NOM_DEV_AUX_VMD                         5126
    /* auxiliary VMD */
#define NOM_DEV_ECG_RESP_VMD                    5130
    /* ECG/Resp Device */
#define NOM_DEV_ARRHY_VMD                       5134
    /* Arrhythmia Analysis Device */
#define NOM_DEV_PULS_VMD                        5138
    /* Pulse Device */
#define NOM_DEV_ST_VMD                          5142
    /* ST Analysis Device */
#define NOM_DEV_CO2_VMD                         5146
    /* CO2 Device */
#define NOM_DEV_PRESS_BLD_NONINV_VMD            5150
    /* NBP Device */
#define NOM_DEV_TEMP_DIFF_VMD                   5178
    /* Temperature Difference Device */
#define NOM_DEV_CNTRL_VMD                       5182
    /* Control Device */
#define NOM_DEV_AL_VMD                          5186
    /* Alarming Device */
#define NOM_DEV_PMSTORE_VMD                     5198
    /* Persistent Metric Store Device */
#define NOM_DEV_ARRHY_CHAN                      5135
    /* Arrhythmia Channel */
#define NOM_DEV_PULS_CHAN                       5139
    /* Pulse Channel */
#define NOM_DEV_ST_CHAN                         5143
    /* ST Channel */
#define NOM_DEV_CO2_CHAN                        5147
    /* CO2 Channel */
#define NOM_DEV_TEMP_DIFF_CHAN                  5179
    /* Temperature Difference Channel */
#define NOM_DEV_PMSTORE_CHAN                    5199
    /* Persistent Metric Store Channel */
#define NOM_DEV_CARD_RATE_CHAN                  5203
    /* Heart Rate Channel */
#define NOM_DEV_SYS_VS_UNCONFIG_MDS             5213
    /* Unconfigurable Device System */
#define NOM_OBJ_DISP                            61616
    /* Display Object */
#define NOM_OBJ_SOUND_GEN                       61648
    /* Sound Generator Object */
#define NOM_OBJ_SETTING                         61649
    /* Device Settings Object */
#define NOM_OBJ_EVENT                           61683
    /* Event Object */
#define NOM_VMD_CONFIG                          61684
    /* Configuration Device */
#define NOM_OBJ_BATT_CHARGER                    61690
    /* Battery Charger Object */
#define NOM_OBJ_ECG_OUT                         61691
    /* ECG Output Object */
#define NOM_OBJ_INPUT_DEV                       61692
    /* Input Device Object */
#define NOM_OBJ_NETWORK                         61693
    /* Network Object */
#define NOM_OBJ_QUICKLINK                       61694
    /* Quicklink Bar Object */
#define NOM_OBJ_SPEAKER                         61695
    /* Speaker Object */
#define NOM_OBJ_AUTO_LIMIT                      61706
    /* Automatic Alarm Limits Object */
#define NOM_OBJ_RECORDING                       61709
    /* Recorder Object */
```

```
#define NOM_OBJ_PUMP                            61716
    /* NBP Pump Object */
#define NOM_OBJ_IR                              61717
    /* IR Transmitter Object */
```

Physiological Identifier

A Physiological Identifier denotes the origin of a physiological measurement. The identifiers are located in the SCADA partition. The Physiological Identifier is transmitted as part of the numeric or wave observed value. The Physiological Identifier may not be unique. However, it is guaranteed that the Label ID is unique. The Label ID is mapped to a Label String based on the text catalogue (see “Attribute: System Localization” on page 79).

The list below shows the numerics and waves which are supported by the monitor. The numerics and waves are sorted according to their internal priority, i.e. numerics or waves with a higher priority are listed first. This information depends heavily on the software revision of the monitor and the connected devices. Especially data coming from a VueLink module depends on the version of the VueLink driver and the specification of the connected external device.

For a given software revision, the IntelliVue monitor may not export all of the numerics specified below. The IntelliVue monitor may export numerics, which are not specified here. If a numeric is exported also depends on the configuration of the monitor. In general, a numeric will only be available if the required measurement module is connected and if the specific measurement is activated. Some measurements require the presents of more than one measurement module or special configuration steps may be necessary to activate the measurement.

Numerics

PVC	Premature Ventricular Contractions	
	Label:	
	NLS_NOM_ECG_V_P_C_CNT	0x00024261
	Observed Value:	
	NOM_ECG_V_P_C_CNT	0x4261
ST	ST generic label	
	Label:	
	NLS_NOM_ECG_AMPL_ST	0x00020300
	Compound Observed Value:	
	NOM_ECG_AMPL_ST_I	0x0301
	NOM_ECG_AMPL_ST_II	0x0302
	NOM_ECG_AMPL_ST_III	0x033D
	NOM_ECG_AMPL_ST_AVR	0x033E
	NOM_ECG_AMPL_ST_AVL	0x033F
	NOM_ECG_AMPL_ST_AVF	0x0340
	NOM_ECG_AMPL_ST_V	0x0343
	NOM_ECG_AMPL_ST_MCL	0x034B
	NOM_ECG_AMPL_ST_V1	0x0303
	NOM_ECG_AMPL_ST_V2	0x0304
	NOM_ECG_AMPL_ST_V3	0x0305
	NOM_ECG_AMPL_ST_V4	0x0306
	NOM_ECG_AMPL_ST_V5	0x0307
	NOM_ECG_AMPL_ST_V6	0x0308
STindx	ST Index	
	Label:	
	NLS_NOM_ECG_AMPL_ST_INDEX	0x0002f03d
	Observed Value:	
	NOM_ECG_AMPL_ST_INDEX	0xF03D
HR	Heart Rate	
	Label:	
	NLS_NOM_ECG_CARD_BEAT_RATE	0x00024182
	Observed Value:	

	NOM_ECG_CARD_BEAT_RATE	0x4182
Pulse	Pulse Rate	
	Label:	
	NLS_NOM_PULS_RATE	0x0002480a
	Observed Value:	
	NOM_PULS_RATE	0x480A
SpO2	Arterial Oxigen Saturation	
	Label:	
	NLS_NOM_PULS_OXIM_SAT_O2	0x00024bb8
	Observed Value:	
	NOM_PULS_OXIM_SAT_O2	0x4BB8
Pulse	Pulse Rate from Plethysmogram	
	Label:	
	NLS_NOM_PULS_OXIM_PULS_RATE	0x00024822
	Observed Value:	
	NOM_PLETH_PULS_RATE	0x4822
SpO2 L	Arterial Oxigen Saturation (left)	
	Label:	
	NLS_NOM_PULS_OXIM_SAT_O2_ART_LEFT	0x00024bc8
	Observed Value:	
	NOM_PULS_OXIM_SAT_O2_ART_LEFT	0x4BC8
Pulse	Pulse Rate from Plethysmogram (left)	
	Label:	
	NLS_SPO2_NAMES_PULS_OXIM_PULS_RATE_LEFT	0x80155401
	Observed Value:	
	NOM_PLETH_PULS_RATE	0x4822
SpO2 R	Arterial Oxigen Saturation (right)	
	Label:	
	NLS_NOM_PULS_OXIM_SAT_O2_ART_RIGHT	0x00024bcc
	Observed Value:	
	NOM_PULS_OXIM_SAT_O2_ART_RIGHT	0x4BCC
Pulse	Pulse Rate from Plethysmogram (right)	
	Label:	
	NLS_SPO2_NAMES_PULS_OXIM_PULS_RATE_RIGHT	0x80155402
	Observed Value:	
	NOM_PLETH_PULS_RATE	0x4822
ΔSpO2	Difference between two SpO2 Values (like Left - Right)	
	Label:	
	NLS_NOM_PULS_OXIM_SAT_O2_DIFF	0x00024bc4
	Observed Value:	
	NOM_PULS_OXIM_SAT_O2_DIFF	0x4BC4
PERF	Perfusion Indicator	
	Label:	
	NLS_NOM_PULS_OXIM_PERF_REL	0x00024bb0
	Observed Value:	
	NOM_PULS_OXIM_PERF_REL	0x4BB0
PERF L	Relative Perfusion Left	
	Label:	
	NLS_NOM_PULS_OXIM_PERF_REL_LEFT	0x0002f08a
	Observed Value:	
	NOM_PULS_OXIM_PERF_REL_LEFT	0xF08A
PERF R	Relative Perfusion Right label	
	Label:	
	NLS_NOM_PULS_OXIM_PERF_REL_RIGHT	0x0002f08b
	Observed Value:	
	NOM_PULS_OXIM_PERF_REL_RIGHT	0xF08B
NBP	non-invasive blood pressure	
	Label:	
	NLS_NOM_PRESS_BLD_NONINV	0x00024a04
	Compound Observed Value:	
	NOM_PRESS_BLD_NONINV_SYS	0x4A05
	NOM_PRESS_BLD_NONINV_DIA	0x4A06
	NOM_PRESS_BLD_NONINV_MEAN	0x4A07
ABP	Arterial Blood Pressure (ABP)	
	Label:	

	NLS_NOM_PRESS_BLD_ART_ABP	0x00024a14
	Compound Observed Value:	
	NOM_PRESS_BLD_ART_ABP_SYS	0x4A15
	NOM_PRESS_BLD_ART_ABP_DIA	0x4A16
	NOM_PRESS_BLD_ART_ABP_MEAN	0x4A17
Pulse	Pulse derived from ABP	
	Label:	
	NLS_PRESS_NAMES_PULSE_FROM_ABP	0x80035402
	Observed Value:	
	NOM_PULS_RATE	0x480A
ART	Arterial Blood Pressure (ART)	
	Label:	
	NLS_NOM_PRESS_BLD_ART	0x00024a10
	Compound Observed Value:	
	NOM_PRESS_BLD_ART_SYS	0x4A11
	NOM_PRESS_BLD_ART_DIA	0x4A12
	NOM_PRESS_BLD_ART_MEAN	0x4A13
Pulse	Pulse derived from ART	
	Label:	
	NLS_PRESS_NAMES_PULSE_FROM_ART	0x80035403
	Observed Value:	
	NOM_PULS_RATE	0x480A
Ao	Arterial Blood Pressure in the Aorta (Ao)	
	Label:	
	NLS_NOM_PRESS_BLD_AORT	0x00024a0c
	Compound Observed Value:	
	NOM_PRESS_BLD_AORT_SYS	0x4A0D
	NOM_PRESS_BLD_AORT_DIA	0x4A0E
	NOM_PRESS_BLD_AORT_MEAN	0x4A0F
Pulse	Pulse derived from Ao	
	Label:	
	NLS_PRESS_NAMES_PULSE_FROM_AO	0x80035404
	Observed Value:	
	NOM_PULS_RATE	0x480A
PAP	Pulmonary Arterial Pressure (PAP)	
	Label:	
	NLS_NOM_PRESS_BLD_ART_PULM	0x00024a1c
	Compound Observed Value:	
	NOM_PRESS_BLD_ART_PULM_SYS	0x4A1D
	NOM_PRESS_BLD_ART_PULM_DIA	0x4A1E
	NOM_PRESS_BLD_ART_PULM_MEAN	0x4A1F
Pulse	Pulse derived from PAP	
	Label:	
	NLS_PRESS_NAMES_PULSE_FROM_PAP	0x80035405
	Observed Value:	
	NOM_PULS_RATE	0x480A
CVP	Central Venous Pressure (CVP)	
	Label:	
	NLS_NOM_PRESS_BLD_VEN_CENT	0x00024a44
	Compound Observed Value:	
	NOM_PRESS_BLD_VEN_CENT_SYS	0x4A45
	NOM_PRESS_BLD_VEN_CENT_DIA	0x4A46
	NOM_PRESS_BLD_VEN_CENT_MEAN	0x4A47
Pulse	Pulse derived from CVP	
	Label:	
	NLS_PRESS_NAMES_PULSE_FROM_CVP	0x80035406
	Observed Value:	
	NOM_PULS_RATE	0x480A
RAP	Right Atrial Pressure (RAP)	
	Label:	
	NLS_NOM_PRESS_BLD_ATR_RIGHT	0x00024a34
	Compound Observed Value:	
	NOM_PRESS_BLD_ATR_RIGHT_SYS	0x4A35
	NOM_PRESS_BLD_ATR_RIGHT_DIA	0x4A36
	NOM_PRESS_BLD_ATR_RIGHT_MEAN	0x4A37

Pulse	Pulse derived from RAP	
	Label:	
	NLS_PRESS_NAMES_PULSE_FROM_RAP	0x80035407
	Observed Value:	
	NOM_PULS_RATE	0x480A
LAP	Left Atrial Pressure (LAP)	
	Label:	
	NLS_NOM_PRESS_BLD_ATR_LEFT	0x00024a30
	Compound Observed Value:	
	NOM_PRESS_BLD_ATR_LEFT_SYS	0x4A31
	NOM_PRESS_BLD_ATR_LEFT_DIA	0x4A32
	NOM_PRESS_BLD_ATR_LEFT_MEAN	0x4A33
Pulse	Pulse derived from LAP	
	Label:	
	NLS_PRESS_NAMES_PULSE_FROM_LAP	0x80035408
	Observed Value:	
	NOM_PULS_RATE	0x480A
ICP	Intra-cranial Pressure (ICP)	
	Label:	
	NLS_NOM_PRESS_INTRA_CRAN	0x00025808
	Compound Observed Value:	
	NOM_PRESS_INTRA_CRAN_SYS	0x5809
	NOM_PRESS_INTRA_CRAN_DIA	0x580A
	NOM_PRESS_INTRA_CRAN_MEAN	0x580B
Pulse	Pulse derived from ICP	
	Label:	
	NLS_PRESS_NAMES_PULSE_FROM_ICP	0x80035409
	Observed Value:	
	NOM_PULS_RATE	0x480A
UAP	Umbilical Arterial Pressure (UAP)	
	Label:	
	NLS_NOM_PRESS_BLD_ART_UMB	0x00024a28
	Compound Observed Value:	
	NOM_PRESS_BLD_ART_UMB_SYS	0x4A29
	NOM_PRESS_BLD_ART_UMB_DIA	0x4A2A
	NOM_PRESS_BLD_ART_UMB_MEAN	0x4A2B
Pulse	Pulse derived from UAP	
	Label:	
	NLS_PRESS_NAMES_PULSE_FROM_UAP	0x8003540a
	Observed Value:	
	NOM_PULS_RATE	0x480A
UVP	Umbilical Venous Pressure (UVP)	
	Label:	
	NLS_NOM_PRESS_BLD_VEN_UMB	0x00024a48
	Compound Observed Value:	
	NOM_PRESS_BLD_VEN_UMB_SYS	0x4A49
	NOM_PRESS_BLD_VEN_UMB_DIA	0x4A4A
	NOM_PRESS_BLD_VEN_UMB_MEAN	0x4A4B
Pulse	Pulse derived from UVP	
	Label:	
	NLS_PRESS_NAMES_PULSE_FROM_UVP	0x8003540b
	Observed Value:	
	NOM_PULS_RATE	0x480A
P	unspecific pressure	
	Label:	
	NLS_NOM_PRESS_BLD	0x00024a00
	Compound Observed Value:	
	NOM_PRESS_BLD_SYS	0x4A01
	NOM_PRESS_BLD_DIA	0x4A02
	NOM_PRESS_BLD_MEAN	0x4A03
Pulse	Pulse derived from unspecific Pressure	
	Label:	
	NLS_PRESS_NAMES_PULSE_FROM_P	0x80035401
	Observed Value:	
	NOM_PULS_RATE	0x480A

PAWP	Pulmonary Artery Wedge Pressure	
	Label:	
	NLS_NOM_PRESS_BLD_ART_PULM_WEDGE	0x00024a24
	Observed Value:	
	NOM_PRESS_BLD_ART_PULM_WEDGE	0x4A24
CPP	Cerebral Perfusion Pressure	
	Label:	
	NLS_NOM_PRESS_CEREB_PERF	0x00025804
	Observed Value:	
	NOM_PRESS_CEREB_PERF	0x5804
CCO	Continuous Cardiac Output	
	Label:	
	NLS_NOM_OUTPUT_CARD_CTS	0x00024bdc
	Observed Value:	
	NOM_OUTPUT_CARD_CTS	0x4BDC
CCI	Continuous Cardiac Output Index	
	Label:	
	NLS_NOM_OUTPUT_CARD_INDEX_CTS	0x0002f047
	Observed Value:	
	NOM_OUTPUT_CARD_INDEX_CTS	0xF047
SV	Stroke Volume	
	Label:	
	NLS_NOM_VOL_BLD_STROKE	0x00024b84
	Observed Value:	
	NOM_VOL_BLD_STROKE	0x4B84
SI	Stroke Index	
	Label:	
	NLS_NOM_VOL_BLD_STROKE_INDEX	0x0002f048
	Observed Value:	
	NOM_VOL_BLD_STROKE_INDEX	0xF048
SVV	Stroke Volume Variation	
	Label:	
	NLS_NOM_VOL_BLD_STROKE_VAR	0x0002f049
	Observed Value:	
	NOM_VOL_BLD_STROKE_VAR	0xF049
dPmax	Index of Left Ventricular Contractility	
	Label:	
	NLS_NOM_GRAD_PRESS_BLD_AORT_POS_MAX	0x00024c25
	Observed Value:	
	NOM_GRAD_PRESS_BLD_AORT_POS_MAX	0x4C25
C.O.	Cardiac Output	
	Label:	
	NLS_NOM_OUTPUT_CARD	0x00024b04
	Observed Value:	
	NOM_OUTPUT_CARD	0x4B04
C.I.	Cardiac Index	
	Label:	
	NLS_NOM_OUTPUT_CARD_INDEX	0x0002490c
	Observed Value:	
	NOM_OUTPUT_CARD_INDEX	0x490C
ITBV	Intrathoracic Blood Volume	
	Label:	
	NLS_NOM_VOL_BLD_INTRA_THOR	0x0002f040
	Observed Value:	
	NOM_VOL_BLD_INTRA_THOR	0xF040
ITBVI	Intrathoracic Blood Volume Index	
	Label:	
	NLS_NOM_VOL_BLD_INTRA_THOR_INDEX	0x0002f041
	Observed Value:	
	NOM_VOL_BLD_INTRA_THOR_INDEX	0xF041
EVLW	Extravascular Lung Water	
	Label:	
	NLS_NOM_VOL_LUNG_WATER_EXTRA_VASC	0x0002f042
	Observed Value:	
	NOM_VOL_LUNG_WATER_EXTRA_VASC	0xF042

EVLWI	Extravascular Lung Water Index	
	Label:	
	NLS_NOM_VOL_LUNG_WATER_EXTRA_VASC_INDEX	0x0002f043
	Observed Value:	
	NOM_VOL_LUNG_WATER_EXTRA_VASC_INDEX	0xF043
GEDV	Global End Diastolic Volume	
	Label:	
	NLS_NOM_VOL_GLOBAL_END_DIA	0x0002f044
	Observed Value:	
	NOM_VOL_GLOBAL_END_DIA	0xF044
GEDVI	Global End Diastolic Volume Index	
	Label:	
	NLS_NOM_VOL_GLOBAL_END_DIA_INDEX	0x0002f045
	Observed Value:	
	NOM_VOL_GLOBAL_END_DIA_INDEX	0xF045
CFI	Cardiac Function Index	
	Label:	
	NLS_NOM_CARD_FUNC_INDEX	0x0002f046
	Observed Value:	
	NOM_CARD_FUNC_INDEX	0xF046
SO2	Oxygen Saturation	
	Label:	
	NLS_NOM_SAT_O2_ART	0x00024b34
	Observed Value:	
	NOM_SAT_O2_ART	0x4B34
SvO2	Mixed Venous Oxygen Saturation	
	Label:	
	NLS_NOM_SAT_O2_VEN	0x00024b3c
	Observed Value:	
	NOM_SAT_O2_VEN	0x4B3C
AaDO2	Alveolar- Arterial Oxygen Difference	
	Label:	
	NLS_NOM_SAT_DIFF_O2_ART_ALV	0x00024b40
	Observed Value:	
	NOM_SAT_DIFF_O2_ART_ALV	0x4B40
Sp-VO2	Difference between Spo2 and SvO2	
	Label:	
	NLS_NOM_SAT_DIFF_O2_ART_VEN	0x0002f06c
	Observed Value:	
	NOM_SAT_DIFF_O2_ART_VEN	0xF06C
TcGas	Generic Term for the Transcutaneous Gases	
	Label:	
	NLS_NOM_GAS_TCUT	0x0002f051
	Observed Value:	
	NOM_GAS_TCUT	0xF051
TcpO2	Transcutaneous Oxygen Partial Pressure	
	Label:	
	NLS_NOM_O2_TCUT	0x000250d0
	Observed Value:	
	NOM_O2_TCUT	0x50D0
TcpCO2	Transcutaneous Carbon Dioxide Partial Pressure	
	Label:	
	NLS_NOM_CO2_TCUT	0x000250cc
	Observed Value:	
	NOM_CO2_TCUT	0x50CC
CO2	CO2 concentration	
	Label:	
	NLS_NOM_AWAY_CO2	0x000250ac
	Compound Observed Value:	
	NOM_AWAY_CO2_ET	0x50B0
	NOM_AWAY_CO2_INSP_MIN	0x50BA
AwRR	Airway Respiration Rate	
	Label:	
	NLS_NOM_AWAY_RESP_RATE	0x00025012
	Observed Value:	

	NOM_AWAY_RESP_RATE	0x5012
O2	Generic oxigen measurement label	
	Label:	
	NLS_NOM_CONC_AWAY_O2	0x00025164
	Compound Observed Value:	
	NOM_CONC_AWAY_O2_ET	0x5378
	NOM_CONC_AWAY_O2_INSP	0x5284
FIO2	Fractional Inspired Oxygen FIO2	
	Label:	
	NLS_NOM_VENT_CONC_AWAY_O2_INSP	0x00027498
	Observed Value:	
	NOM_VENT_CONC_AWAY_O2_INSP	0x7498
RR	Respiration Rate	
	Label:	
	NLS_NOM_RESP_RATE	0x0002500a
	Observed Value:	
	NOM_RESP_RATE	0x500A
T.I.	Transthoracic Impedance	
	Label:	
	NLS_NOM_IMPED_TTHOR	0x000250e4
	Observed Value:	
	NOM_IMPED_TTHOR	0x50E4
VCO2	CO2 Production	
	Label:	
	NLS_NOM_FLOW_CO2_PROD_RESP	0x000250e0
	Observed Value:	
	NOM_FLOW_CO2_PROD_RESP	0x50E0
Pplat	Plateau Pressure	
	Label:	
	NLS_NOM_PRESS_RESP_PLAT	0x000250e8
	Observed Value:	
	NOM_PRESS_RESP_PLAT	0x50E8
AWPmin	Airway Pressure Minimum	
	Label:	
	NLS_NOM_PRESS_AWAY_MIN	0x000250f2
	Observed Value:	
	NOM_PRESS_AWAY_MIN	0x50F2
CPAP	Continuous Positive Airway Pressure	
	Label:	
	NLS_NOM_PRESS_AWAY_CTS_POS	0x000250f4
	Observed Value:	
	NOM_PRESS_AWAY_CTS_POS	0x50F4
IPEEP	Intrinsic PEEP Breathing Pressure	
	Label:	
	NLS_NOM_PRESS_AWAY_END_EXP_POS_INTRINSIC	0x00025100
	Observed Value:	
	NOM_PRESS_AWAY_END_EXP_POS_INTRINSIC	0x5100
AWPin	Airway Pressure Wave - measured in the inspiratory path	
	Label:	
	NLS_NOM_PRESS_AWAY_INSP	0x00025108
	Observed Value:	
	NOM_PRESS_AWAY_INSP	0x5108
PIP	Positive Inspiratory ressure	
	Label:	
	NLS_NOM_PRESS_AWAY_INSP_MAX	0x00025109
	Observed Value:	
	NOM_PRESS_AWAY_INSP_MAX	0x5109
MnAwP	Mean Airway Pressure. Printer Context	
	Label:	
	NLS_NOM_PRESS_AWAY_INSP_MEAN	0x0002510b
	Observed Value:	
	NOM_PRESS_AWAY_INSP_MEAN	0x510B
I:E 1:	Inpired:Expired Ratio	
	Label:	
	NLS_NOM_RATIO_IE	0x00025118

	Observed Value:	
	NOM_RATIO_IE	0x5118
Vd/Vt	Ratio of Deadspace to Tidal Volume Vd/Vt	
	Label:	
	NLS_NOM_RATIO_AWAY_DEADSP_TIDAL	0x0002511c
	Observed Value:	
	NOM_RATIO_AWAY_DEADSP_TIDAL	0x511C
Rstat	Static Lung Resistance	
	Label:	
	NLS_NOM_RES_AWAY	0x00025120
	Observed Value:	
	NOM_RES_AWAY	0x5120
TV	Tidal Volume	
	Label:	
	NLS_NOM_VOL_AWAY_TIDAL	0x0002513c
	Observed Value:	
	NOM_VOL_AWAY_TIDAL	0x513C
MINVOL	Airway Minute Volum Inspiratory	
	Label:	
	NLS_NOM_VOL_MINUTE_AWAY	0x00025148
	Observed Value:	
	NOM_VOL_MINUTE_AWAY	0x5148
SpMV	Spontaneous Minute Volume	
	Label:	
	NLS_NOM_VENT_VOL_MINUTE_AWAY_SPONT	0x0002f091
	Observed Value:	
	NOM_VENT_VOL_MINUTE_AWAY_SPONT	0xF091
ΔO2	relative Dead Space	
	Label:	
	NLS_NOM_VENT_CONC_AWAY_O2_DELTA	0x00025168
	Observed Value:	
	NOM_VENT_CONC_AWAY_O2_DELTA	0x5168
PECO2	Partial O2 Venous	
	Label:	
	NLS_NOM_VENT_AWAY_CO2_EXP	0x0002517c
	Observed Value:	
	NOM_VENT_AWAY_CO2_EXP	0x517C
AWFin	Airway Flow Wave - measured in the inspiratory path	
	Label:	
	NLS_NOM_VENT_FLOW_INSP	0x0002518c
	Observed Value:	
	NOM_VENT_FLOW_INSP	0x518C
VQI	Ventilation Perfusion Index	
	Label:	
	NLS_NOM_VENT_FLOW_RATIO_PERF_ALV_INDEX	0x00025190
	Observed Value:	
	NOM_VENT_FLOW_RATIO_PERF_ALV_INDEX	0x5190
Poccl	Occlusion Pressure	
	Label:	
	NLS_NOM_VENT_PRESS_OCCL	0x0002519c
	Observed Value:	
	NOM_VENT_PRESS_OCCL	0x519C
PEEP	Positive End-Expiratory Pressure PEEP	
	Label:	
	NLS_NOM_VENT_PRESS_AWAY_END_EXP_POS	0x000251a8
	Observed Value:	
	NOM_VENT_PRESS_AWAY_END_EXP_POS	0x51A8
Vd	Dead Space Volume Vd	
	Label:	
	NLS_NOM_VENT_VOL_AWAY_DEADSP	0x000251b0
	Observed Value:	
	NOM_VENT_VOL_AWAY_DEADSP	0x51B0
RelVd	relative Dead Space	
	Label:	
	NLS_NOM_VENT_VOL_AWAY_DEADSP_REL	0x000251b4

	Observed Value:	
	NOM_VENT_VOL_AWAY_DEADSP_REL	0x51B4
TrpVol	Lung Volume Trapped	
	Label:	
	NLS_NOM_VENT_VOL_LUNG_TRAPD	0x000251b8
	Observed Value:	
	NOM_VENT_VOL_LUNG_TRAPD	0x51B8
Leak	Leakage	
	Label:	
	NLS_NOM_VENT_VOL_LEAK	0x00025370
	Observed Value:	
	NOM_VENT_VOL_LEAK	0x5370
ALVENT	Alveolar Ventilation ALVENT	
	Label:	
	NLS_NOM_VENT_VOL_LUNG_ALV	0x00025374
	Observed Value:	
	NOM_VENT_VOL_LUNG_ALV	0x5374
VC	Vital Lung Capacity	
	Label:	
	NLS_NOM_CAPAC_VITAL	0x00025080
	Observed Value:	
	NOM_CAPAC_VITAL	0x5080
COMP	generic label Lung Compliance	
	Label:	
	NLS_NOM_COMPL_LUNG	0x00025088
	Observed Value:	
	NOM_COMPL_LUNG	0x5088
Cdyn	Dynamic Lung Compliance	
	Label:	
	NLS_NOM_COMPL_LUNG_DYN	0x0002508c
	Observed Value:	
	NOM_COMPL_LUNG_DYN	0x508C
Cstat	Static Lung Compliance	
	Label:	
	NLS_NOM_COMPL_LUNG_STATIC	0x00025090
	Observed Value:	
	NOM_COMPL_LUNG_STATIC	0x5090
BIS	Bispectral Index	
	Label:	
	NLS_NOM_EEG_BISPECTRAL_INDEX	0x0002f04e
	Observed Value:	
	NOM_EEG_BISPECTRAL_INDEX	0xF04E
SQI	Signal Quality Index	
	Label:	
	NLS_NOM_EEG_BIS_SIG_QUAL_INDEX	0x0002f04d
	Observed Value:	
	NOM_EEG_BIS_SIG_QUAL_INDEX	0xF04D
EMG	Electromyography	
	Label:	
	NLS_NOM_EMG_ELEC_POTL_MUSCL	0x0002593c
	Observed Value:	
	NOM_EMG_ELEC_POTL_MUSCL	0x593C
TP	Total Power	
	Label:	
	NLS_NOM_EEG_PWR_SPEC_TOT	0x000259b8
	Observed Value:	
	NOM_EEG_PWR_SPEC_TOT	0x59B8
TP1	Total Power channel 1	
	Label:	
	NLS_EEG_NAMES_CHAN_TP1	0x800f5403
	Observed Value:	
	NOM_EEG_PWR_SPEC_TOT	0x59B8
TP2	Total Power channel 2	
	Label:	
	NLS_EEG_NAMES_CHAN_TP2	0x800f5404

	Observed Value:	
	NOM_EEG_PWR_SPEC_TOT	0x59B8
SR	Suppression Ratio	
	Label:	
	NLS_NOM_EEG_RATIO_SUPPRN	0x0002f04a
	Observed Value:	
	NOM_EEG_RATIO_SUPPRN	0xF04A
SEF	Spectral Edge Frequency	
	Label:	
	NLS_NOM_EEG_FREQ_PWR_SPEC_CRTX_SPECTRAL_EDGE	0x00025988
	Observed Value:	
	NOM_EEG_FREQ_PWR_SPEC_CRTX_SPECTRAL_EDGE	0x5988
MDF	Mean Dominant Frequency	
	Label:	
	NLS_NOM_EEG_FREQ_PWR_SPEC_CRTX_DOM_MEAN	0x0002597c
	Observed Value:	
		0x0000
PPF	Peak Power Frequency	
	Label:	
	NLS_NOM_EEG_FREQ_PWR_SPEC_CRTX_PEAK	0x00025984
	Observed Value:	
		0x0000
Frequ1	generic label for EEG channel 1	
	Label:	
	NLS_EEG_NAMES_CHAN_FREQ1	0x800f5413
	Compound Observed Value:	
	NOM_EEG_FREQ_PWR_SPEC_CRTX_SPECTRAL_EDGE	0x5988
	NOM_EEG_FREQ_PWR_SPEC_CRTX_DOM_MEAN	0x597C
	NOM_EEG_FREQ_PWR_SPEC_CRTX_PEAK	0x5984
Frequ2	generic label for EEG channel 2	
	Label:	
	NLS_EEG_NAMES_CHAN_FREQ2	0x800f5414
	Compound Observed Value:	
	NOM_EEG_FREQ_PWR_SPEC_CRTX_SPECTRAL_EDGE	0x5988
	NOM_EEG_FREQ_PWR_SPEC_CRTX_DOM_MEAN	0x597C
	NOM_EEG_FREQ_PWR_SPEC_CRTX_PEAK	0x5984
Prcnt1	generic label for EEG channel 1	
	Label:	
	NLS_EEG_NAMES_CHAN_PCNT1	0x800f5415
	Compound Observed Value:	
	NOM_EEG_PWR_SPEC_ALPHA_REL	0x59D4
	NOM_EEG_PWR_SPEC_BETA_REL	0x59D8
	NOM_EEG_PWR_SPEC_DELTA_REL	0x59DC
	NOM_EEG_PWR_SPEC_THETA_REL	0x59E0
Prcnt2	generic label for EEG channel 2	
	Label:	
	NLS_EEG_NAMES_CHAN_PCNT2	0x800f5416
	Compound Observed Value:	
	NOM_EEG_PWR_SPEC_ALPHA_REL	0x59D4
	NOM_EEG_PWR_SPEC_BETA_REL	0x59D8
	NOM_EEG_PWR_SPEC_DELTA_REL	0x59DC
	NOM_EEG_PWR_SPEC_THETA_REL	0x59E0
AAI	A-Line ARX Index	
	Label:	
	NLS_NOM_EMFC_AAI	0x04011194
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
BSI	Burst Suppression Indicator	
	Label:	
	NLS_NOM_EMFC_BSI	0x04011198
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
T	Unspecific Temperature	
	Label:	
	NLS_NOM_TEMP	0x00024b48

	Observed Value:	
	NOM_TEMP	0x4B48
Trect	Rectal Temperature	
	Label:	
	NLS_NOM_TEMP_RECT	0x0002e004
	Observed Value:	
	NOM_TEMP_RECT	0xE004
Tblood	Tblood	
	Label:	
	NLS_NOM_TEMP_BLD	0x0002e014
	Observed Value:	
	NOM_TEMP_BLD	0xE014
Tcore	Core (Body) Temperature	
	Label:	
	NLS_NOM_TEMP_CORE	0x00024b60
	Observed Value:	
	NOM_TEMP_CORE	0x4B60
Tskin	Skin Temperature	
	Label:	
	NLS_NOM_TEMP_SKIN	0x00024b74
	Observed Value:	
	NOM_TEMP_SKIN	0x4B74
Tesoph	Esophagial Temperature	
	Label:	
	NLS_NOM_TEMP_ESOPH	0x00024b64
	Observed Value:	
	NOM_TEMP_ESOPH	0x4B64
Tnaso	Naso pharyngial Temperature	
	Label:	
	NLS_NOM_TEMP_NASOPH	0x00024b6c
	Observed Value:	
	NOM_TEMP_NASOPH	0x4B6C
Tart	Areterial Temperature	
	Label:	
	NLS_NOM_TEMP_ART	0x00024b50
	Observed Value:	
	NOM_TEMP_ART	0x4B50
Tven	Venous Temperature	
	Label:	
	NLS_NOM_TEMP_VEN	0x00024b7c
	Observed Value:	
	NOM_TEMP_VEN	0x4B7C
Tairwy	Airway Temperature	
	Label:	
	NLS_NOM_TEMP_AWAY	0x00024b54
	Observed Value:	
	NOM_TEMP_AWAY	0x4B54
ΔT	Difference Temperature	
	Label:	
	NLS_NOM_TEMP_DIFF	0x0002e018
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
T1	Non-specific temperature 1	
	Label:	
	NLS_NOM_EMFC_T1	0x04010064
	Observed Value:	
	NOM_TEMP	0x4B48
T2	Non-specific temperature 2	
	Label:	
	NLS_NOM_EMFC_T2	0x04010068
	Observed Value:	
	NOM_TEMP	0x4B48
Tpat	Patient Temperature	
	Label:	
	NLS_NOM_EMFC_Tpat	0x04010a38

	Observed Value:	
	NOM_TEMP	0x4B48
T-Tre	Temperature difference between Temp and Trect	
	Label:	
	NLS_DIFFTEMP_NAMES_TEMP_TRECT	0x800b5402
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
T-Tbl	Temperature difference between Temp and Tblood	
	Label:	
	NLS_DIFFTEMP_NAMES_TEMP_TBLOOD	0x800b5409
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
T-Tco	Temperature difference between Temp and Tcore	
	Label:	
	NLS_DIFFTEMP_NAMES_TEMP_TCORE	0x800b5403
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
T-Tsk	Temperature difference between Temp and Tskin	
	Label:	
	NLS_DIFFTEMP_NAMES_TEMP_TSKIN	0x800b5404
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
T-Tes	Temperature difference between Temp and Tesoph	
	Label:	
	NLS_DIFFTEMP_NAMES_TEMP_TESOPH	0x800b5405
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
T-Tna	Temperature difference between Temp and Tnaso	
	Label:	
	NLS_DIFFTEMP_NAMES_TEMP_TNASO	0x800b5406
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
T-Tar	Temperature difference between Temp and Tart	
	Label:	
	NLS_DIFFTEMP_NAMES_TEMP_TART	0x800b5408
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
T-Tve	Temperature difference between Temp and Tven	
	Label:	
	NLS_DIFFTEMP_NAMES_TEMP_TVEN	0x800b5407
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
T-Taw	Temperature difference between Temp and Tairwy	
	Label:	
	NLS_DIFFTEMP_NAMES_TEMP_TAIRWY	0x800b540a
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tre-T	Temperature difference between Trect and Temp	
	Label:	
	NLS_DIFFTEMP_NAMES_TRECT_TEMP	0x800b5415
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tre-Tbl	Temperature difference between Trect and Tblood	
	Label:	
	NLS_DIFFTEMP_NAMES_TRECT_TBLOOD	0x800b541d
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tre-Tco	Temperature difference between Trect and Tcore	
	Label:	
	NLS_DIFFTEMP_NAMES_TRECT_TCORE	0x800b5417
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tre-Tsk	Temperature difference between Trect and Tskin	
	Label:	
	NLS_DIFFTEMP_NAMES_TRECT_TSKIN	0x800b5418

	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tre-Tes	Temperature difference between Trect and Tesoph	
	Label:	
	NLS_DIFFTEMP_NAMES_TRECT_TESOPH	0x800b5419
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tre-Tna	Temperature difference between Trect and Tnaso	
	Label:	
	NLS_DIFFTEMP_NAMES_TRECT_TNASO	0x800b541a
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tre-Tar	Temperature difference between Trect and Tart	
	Label:	
	NLS_DIFFTEMP_NAMES_TRECT_TART	0x800b541c
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tre-Tve	Temperature difference between Trect and Tven	
	Label:	
	NLS_DIFFTEMP_NAMES_TRECT_TVEN	0x800b541b
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tre-Taw	Temperature difference between Trect and Tairwy	
	Label:	
	NLS_DIFFTEMP_NAMES_TRECT_TAIRWY	0x800b541e
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tbl-T	Temperature difference between Tblood and Temp	
	Label:	
	NLS_DIFFTEMP_NAMES_TBLOOD_TEMP	0x800b54a1
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tbl-Tre	Temperature difference between Tblood and Trect	
	Label:	
	NLS_DIFFTEMP_NAMES_TBLOOD_TRECT	0x800b54a2
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tbl-Tco	Temperature difference between Tblood and Tcore	
	Label:	
	NLS_DIFFTEMP_NAMES_TBLOOD_TCORE	0x800b54a3
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tbl-Tsk	Temperature difference between Tblood and Tskin	
	Label:	
	NLS_DIFFTEMP_NAMES_TBLOOD_TSKIN	0x800b54a4
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tbl-Tes	Temperature difference between Tblood and Tesoph	
	Label:	
	NLS_DIFFTEMP_NAMES_TBLOOD_TESOPH	0x800b54a5
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tbl-Tna	Temperature difference between Tblood and Tnaso	
	Label:	
	NLS_DIFFTEMP_NAMES_TBLOOD_TNASO	0x800b54a6
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tbl-Tar	Temperature difference between Tblood and Tart	
	Label:	
	NLS_DIFFTEMP_NAMES_TBLOOD_TART	0x800b54a8
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tbl-Tve	Temperature difference between Tblood and Tven	
	Label:	
	NLS_DIFFTEMP_NAMES_TBLOOD_TVEN	0x800b54a7

	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tbl-Taw	Temperature difference between Tblood and Tairwy	
	Label:	
	NLS_DIFFTEMP_NAMES_TBLOOD_TAIRWY	0x800b54aa
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tco-T	Temperature difference between Tcore and Temp	
	Label:	
	NLS_DIFFTEMP_NAMES_TCORE_TEMP	0x800b5429
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tco-Tre	Temperature difference between Tcore and Trect	
	Label:	
	NLS_DIFFTEMP_NAMES_TCORE_TRECT	0x800b542a
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tco-Tbl	Temperature difference between Tcore and Tblood	
	Label:	
	NLS_DIFFTEMP_NAMES_TCORE_TBLOOD	0x800b5431
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tco-Tsk	Temperature difference between Tcore and Tskin	
	Label:	
	NLS_DIFFTEMP_NAMES_TCORE_TSKIN	0x800b542c
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tco-Tes	Temperature difference between Tcore and Tesoph	
	Label:	
	NLS_DIFFTEMP_NAMES_TCORE_TESOPH	0x800b542d
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tco-Tna	Temperature difference between Tcore and Tnaso	
	Label:	
	NLS_DIFFTEMP_NAMES_TCORE_TNASO	0x800b542e
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tco-Tar	Temperature difference between Tcore and Tart	
	Label:	
	NLS_DIFFTEMP_NAMES_TCORE_TART	0x800b5430
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tco-Tve	Temperature difference between Tcore and Tven	
	Label:	
	NLS_DIFFTEMP_NAMES_TCORE_TVEN	0x800b542f
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tco-Taw	Temperature difference between Tcore and Tairwy	
	Label:	
	NLS_DIFFTEMP_NAMES_TCORE_TAIRWY	0x800b5432
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tsk-T	Temperature difference between Tskin and Temp	
	Label:	
	NLS_DIFFTEMP_NAMES_TSKIN_TEMP	0x800b543d
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tsk-Tre	Temperature difference between Tskin and Trect	
	Label:	
	NLS_DIFFTEMP_NAMES_TSKIN_TRECT	0x800b543e
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tsk-Tbl	Temperature difference between Tskin and Tblood	
	Label:	
	NLS_DIFFTEMP_NAMES_TSKIN_TBLOOD	0x800b5445

	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tsk-Tco	Temperature difference between Tskin and Tcore	
	Label:	
	NLS_DIFFTEMP_NAMES_TSKIN_TCORE	0x800b543f
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tsk-Tes	Temperature difference between Tskin and Tesoph	
	Label:	
	NLS_DIFFTEMP_NAMES_TSKIN_TESOPH	0x800b5441
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tsk-Tna	Temperature difference between Tskin and Tnaso	
	Label:	
	NLS_DIFFTEMP_NAMES_TSKIN_TNASO	0x800b5442
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tsk-Tar	Temperature difference between Tskin and Tart	
	Label:	
	NLS_DIFFTEMP_NAMES_TSKIN_TART	0x800b5444
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tsk-Tve	Temperature difference between Tskin and Tven	
	Label:	
	NLS_DIFFTEMP_NAMES_TSKIN_TVEN	0x800b5443
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tsk-Taw	Temperature difference between Tskin and Tairwy	
	Label:	
	NLS_DIFFTEMP_NAMES_TSKIN_TAIRWY	0x800b5446
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tes-T	Temperature difference between Tesoph and Temp	
	Label:	
	NLS_DIFFTEMP_NAMES_TESOPH_TEMP	0x800b5451
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tes-Tre	Temperature difference between Tesoph and Trect	
	Label:	
	NLS_DIFFTEMP_NAMES_TESOPH_TRECT	0x800b5452
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tes-Tbl	Temperature difference between Tesoph and Tblood	
	Label:	
	NLS_DIFFTEMP_NAMES_TESOPH_TBLOOD	0x800b5459
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tes-Tco	Temperature difference between Tesoph and Tcore	
	Label:	
	NLS_DIFFTEMP_NAMES_TESOPH_TCORE	0x800b5453
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tes-Tsk	Temperature difference between Tesoph and Tskin	
	Label:	
	NLS_DIFFTEMP_NAMES_TESOPH_TSKIN	0x800b5454
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tes-Tna	Temperature difference between Tesoph and Tnaso	
	Label:	
	NLS_DIFFTEMP_NAMES_TESOPH_TNASO	0x800b5456
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tes-Tar	Temperature difference between Tesoph and Tart	
	Label:	
	NLS_DIFFTEMP_NAMES_TESOPH_TART	0x800b5458

	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tes-Tve	Temperature difference between Tesoph and Tven	
	Label:	
	NLS_DIFFTEMP_NAMES_TESOPH_TVEN	0x800b5457
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tes-Taw	Temperature difference between Tesoph and Tairwy	
	Label:	
	NLS_DIFFTEMP_NAMES_TESOPH_TAIRWY	0x800b545a
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tna-T	Temperature difference between Tnaso and Temp	
	Label:	
	NLS_DIFFTEMP_NAMES_TNASO_TEMP	0x800b5465
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tna-Tre	Temperature difference between Tnaso and Trect	
	Label:	
	NLS_DIFFTEMP_NAMES_TNASO_TRECT	0x800b5466
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tna-Tbl	Temperature difference between Tnaso and Tblood	
	Label:	
	NLS_DIFFTEMP_NAMES_TNASO_TBLOOD	0x800b546d
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tna-Tco	Temperature difference between Tnaso and Tcore	
	Label:	
	NLS_DIFFTEMP_NAMES_TNASO_TCORE	0x800b5467
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tna-Tsk	Temperature difference between Tnaso and Tskin	
	Label:	
	NLS_DIFFTEMP_NAMES_TNASO_TSKIN	0x800b5468
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tna-Tes	Temperature difference between Tnaso and Tesoph	
	Label:	
	NLS_DIFFTEMP_NAMES_TNASO_TESOPH	0x800b5469
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tna-Tar	Temperature difference between Tnaso and Tart	
	Label:	
	NLS_DIFFTEMP_NAMES_TNASO_TART	0x800b546c
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tna-Tve	Temperature difference between Tnaso and Tven	
	Label:	
	NLS_DIFFTEMP_NAMES_TNASO_TVEN	0x800b546b
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tna-Taw	Temperature difference between Tnaso and Tairwy	
	Label:	
	NLS_DIFFTEMP_NAMES_TNASO_TAIRWY	0x800b546e
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tar-T	Temperature difference between Tart and Temp	
	Label:	
	NLS_DIFFTEMP_NAMES_TART_TEMP	0x800b548d
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tar-Tre	Temperature difference between Tart and Trect	
	Label:	
	NLS_DIFFTEMP_NAMES_TART_TRECT	0x800b548e

	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tar-Tbl	Temperature difference between Tart and Tblood	
	Label:	
	NLS_DIFFTEMP_NAMES_TART_TBLOOD	0x800b5495
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tar-Tco	Temperature difference between Tart and Tcore	
	Label:	
	NLS_DIFFTEMP_NAMES_TART_TCORE	0x800b548f
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tar-Tsk	Temperature difference between Tart and Tskin	
	Label:	
	NLS_DIFFTEMP_NAMES_TART_TSKIN	0x800b5490
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tar-Tes	Temperature difference between Tart and Tesoph	
	Label:	
	NLS_DIFFTEMP_NAMES_TART_TESOPH	0x800b5491
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tar-Tna	Temperature difference between Tart and Tnaso	
	Label:	
	NLS_DIFFTEMP_NAMES_TART_TNASO	0x800b5492
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tar-Tve	Temperature difference between Tart and Tven	
	Label:	
	NLS_DIFFTEMP_NAMES_TART_TVEN	0x800b5493
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tar-Taw	Temperature difference between Tart and Tairwy	
	Label:	
	NLS_DIFFTEMP_NAMES_TART_TAIRWY	0x800b5496
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tve-T	Temperature difference between Tven and Temp	
	Label:	
	NLS_DIFFTEMP_NAMES_TVEN_TEMP	0x800b5479
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tve-Tre	Temperature difference between Tven and Trect	
	Label:	
	NLS_DIFFTEMP_NAMES_TVEN_TRECT	0x800b547a
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tve-Tbl	Temperature difference between Tven and Tblood	
	Label:	
	NLS_DIFFTEMP_NAMES_TVEN_TBLOOD	0x800b5481
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tve-Tco	Temperature difference between Tven and Tcore	
	Label:	
	NLS_DIFFTEMP_NAMES_TVEN_TCORE	0x800b547b
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tve-Tsk	Temperature difference between Tven and Tskin	
	Label:	
	NLS_DIFFTEMP_NAMES_TVEN_TSKIN	0x800b547c
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tve-Tes	Temperature difference between Tven and Tesoph	
	Label:	
	NLS_DIFFTEMP_NAMES_TVEN_TESOPH	0x800b547d

	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tve-Tna	Temperature difference between Tven and Tnaso	
	Label:	
	NLS_DIFFTEMP_NAMES_TVEN_TNASO	0x800b547e
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tve-Tar	Temperature difference between Tven and Tart	
	Label:	
	NLS_DIFFTEMP_NAMES_TVEN_TART	0x800b5480
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Tve-Taw	Temperature difference between Tven and Tairwy	
	Label:	
	NLS_DIFFTEMP_NAMES_TVEN_TAIRWY	0x800b5482
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Taw-T	Temperature difference between Tairwy and Temp	
	Label:	
	NLS_DIFFTEMP_NAMES_TAIRWY_TEMP	0x800b54b5
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Taw-Tre	Temperature difference between Tairwy and Trect	
	Label:	
	NLS_DIFFTEMP_NAMES_TAIRWY_TRECT	0x800b54b6
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Taw-Tbl	Temperature difference between Tairwy and Tblood	
	Label:	
	NLS_DIFFTEMP_NAMES_TAIRWY_TBLOOD	0x800b54bd
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Taw-Tco	Temperature difference between Tairwy and Tcore	
	Label:	
	NLS_DIFFTEMP_NAMES_TAIRWY_TCORE	0x800b54b7
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Taw-Tsk	Temperature difference between Tairwy and Tskin	
	Label:	
	NLS_DIFFTEMP_NAMES_TAIRWY_TSKIN	0x800b54b8
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Taw-Tes	Temperature difference between Tairwy and Tesoph	
	Label:	
	NLS_DIFFTEMP_NAMES_TAIRWY_TESOPH	0x800b54b9
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Taw-Tna	Temperature difference between Tairwy and Tnaso	
	Label:	
	NLS_DIFFTEMP_NAMES_TAIRWY_TNASO	0x800b54ba
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Taw-Tar	Temperature difference between Tairwy and Tart	
	Label:	
	NLS_DIFFTEMP_NAMES_TAIRWY_TART	0x800b54bc
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
Taw-Tve	Temperature difference between Tairwy and Tven	
	Label:	
	NLS_DIFFTEMP_NAMES_TAIRWY_TVEN	0x800b54bb
	Observed Value:	
	NOM_TEMP_DIFF	0xE018
N2	generic N2 label	
	Label:	
	NLS_NOM_CONC_AWAY_N2	0x0002537c

	Compound Observed Value:	
	NOM_CONC_AWAY_N2_ET	0x5380
	NOM_CONC_AWAY_N2_INSP	0x5384
N2O	generic Nitrous Oxide label	
	Label:	
	NLS_NOM_CONC_AWAY_N2O	0x000251f0
	Compound Observed Value:	
	NOM_CONC_AWAY_N2O_ET	0x522C
	NOM_CONC_AWAY_N2O_INSP	0x5280
ISO	generic Isoflurane label	
	Label:	
	NLS_NOM_CONC_AWAY_ISOFL	0x000251e8
	Compound Observed Value:	
	NOM_CONC_AWAY_ISOFL_ET	0x5224
	NOM_CONC_AWAY_ISOFL_INSP	0x5278
SEV	generic Sevoflurane label	
	Label:	
	NLS_NOM_CONC_AWAY_SEVOFL	0x000251e4
	Compound Observed Value:	
	NOM_CONC_AWAY_SEVOFL_ET	0x5220
	NOM_CONC_AWAY_SEVOFL_INSP	0x5274
ENF	generic Enflurane label	
	Label:	
	NLS_NOM_CONC_AWAY_ENFL	0x000251dc
	Compound Observed Value:	
	NOM_CONC_AWAY_ENFL_ET	0x5218
	NOM_CONC_AWAY_ENFL_INSP	0x526C
HAL	generic Halothane label	
	Label:	
	NLS_NOM_CONC_AWAY_HALOTH	0x000251e0
	Compound Observed Value:	
	NOM_CONC_AWAY_HALOTH_ET	0x521C
	NOM_CONC_AWAY_HALOTH_INSP	0x5270
DES	generic Desflurane label	
	Label:	
	NLS_NOM_CONC_AWAY_DESFL	0x000251d8
	Compound Observed Value:	
	NOM_CONC_AWAY_DESFL_ET	0x5214
	NOM_CONC_AWAY_DESFL_INSP	0x5268
AGT	generic Agent label	
	Label:	
	NLS_NOM_CONC_AWAY_AGENT	0x00025388
	Compound Observed Value:	
	NOM_CONC_AWAY_AGENT_ET	0x538C
	NOM_CONC_AWAY_AGENT_INSP	0x5390
AGT1	generic Agent1 label	
	Label:	
	NLS_GASES_NAMES_CONC_AWAY_AGENT1	0x805a5401
	Compound Observed Value:	
	NOM_CONC_AWAY_AGENT_ET	0x538C
	NOM_CONC_AWAY_AGENT_INSP	0x5390
AGT2	generic Agent2 label	
	Label:	
	NLS_GASES_NAMES_CONC_AWAY_AGENT2	0x805a5402
	Compound Observed Value:	
	NOM_CONC_AWAY_AGENT_ET	0x538C
	NOM_CONC_AWAY_AGENT_INSP	0x5390
MAC	Airway MAC Concentration	
	Label:	
	NLS_NOM_CONC_AWAY_SUM_MAC	0x0002f05d
	Compound Observed Value:	
	NOM_CONC_AWAY_SUM_MAC_ET	0xF05E
	NOM_CONC_AWAY_SUM_MAC_INSP	0xF05F
SVRI	Systemic Vascular Resistance Index	
	Label:	

	NLS_NOM_RES_VASC_SYS_INDEX	0x00024900
	Observed Value:	
	NOM_RES_VASC_SYS_INDEX	0x4900
LVSWI	Left Ventricular Stroke Volume Index	
	Label:	
	NLS_NOM_WK_LV_STROKE_INDEX	0x00024904
	Observed Value:	
	NOM_WK_LV_STROKE_INDEX	0x4904
RVSWI	Right Ventricular Stroke Work Index	
	Label:	
	NLS_NOM_WK_RV_STROKE_INDEX	0x00024908
	Observed Value:	
	NOM_WK_RV_STROKE_INDEX	0x4908
VO2	Oxygen Consumption VO2	
	Label:	
	NLS_NOM_SAT_O2_CONSUMP	0x00024b00
	Observed Value:	
	NOM_SAT_O2_CONSUMP	0x4B00
PVR	Pulmonary vascular Resistance	
	Label:	
	NLS_NOM_RES_VASC_PULM	0x00024b24
	Observed Value:	
	NOM_RES_VASC_PULM	0x4B24
SVR	Systemic Vascular Resistance	
	Label:	
	NLS_NOM_RES_VASC_SYS	0x00024b28
	Observed Value:	
	NOM_RES_VASC_SYS	0x4B28
LCW	Left Cardiac Work	
	Label:	
	NLS_NOM_WK_CARD_LEFT	0x00024b90
	Observed Value:	
	NOM_WK_CARD_LEFT	0x4B90
RCW	Right Cardiac Work	
	Label:	
	NLS_NOM_WK_CARD_RIGHT	0x00024b94
	Observed Value:	
	NOM_WK_CARD_RIGHT	0x4B94
LVSW	Left Ventricular Stroke Volume	
	Label:	
	NLS_NOM_WK_LV_STROKE	0x00024b9c
	Observed Value:	
	NOM_WK_LV_STROKE	0x4B9C
RVSW	Right Ventricular Stroke Volume	
	Label:	
	NLS_NOM_WK_RV_STROKE	0x00024ba4
	Observed Value:	
	NOM_WK_RV_STROKE	0x4BA4
GCS	Glasgow Coma Score	
	Label:	
	NLS_NOM_SCORE_GLAS_COMA	0x00025880
	Observed Value:	
	NOM_SCORE_GLAS_COMA	0x5880
EyeRsp	SubScore of the GCS: Eye Response	
	Label:	
	NLS_NOM_SCORE_EYE_SUBSC_GLAS_COMA	0x00025882
	Observed Value:	
	NOM_SCORE_EYE_SUBSC_GLAS_COMA	0x5882
MotRsp	SubScore of the GCS: Motoric Response	
	Label:	
	NLS_NOM_SCORE_MOTOR_SUBSC_GLAS_COMA	0x00025883
	Observed Value:	
	NOM_SCORE_MOTOR_SUBSC_GLAS_COMA	0x5883
VblRsp	SubScore of the GCS: Verbal Response	
	Label:	

	NLS_NOM_SCORE_SUBSC_VERBAL_GLAS_COMA	0x00025884
	Observed Value:	
	NOM_SCORE_SUBSC_VERBAL_GLAS_COMA	0x5884
HC	Head Circumferince	
	Label:	
	NLS_NOM_CIRCUM_HEAD	0x00025900
	Observed Value:	
	NOM_CIRCUM_HEAD	0x5900
PRL	Pupil Reaction Left eye - light reaction of left eye's pupil	
	Label:	
	NLS_NOM_TIME_PD_PUPIL_REACT_LEFT	0x00025924
	Observed Value:	
	NOM_TIME_PD_PUPIL_REACT_LEFT	0x5924
PRR	Pupil Reaction Righteye - light reaction of right eye's pupil	
	Label:	
	NLS_NOM_TIME_PD_PUPIL_REACT_RIGHT	0x00025928
	Observed Value:	
	NOM_TIME_PD_PUPIL_REACT_RIGHT	0x5928
PHa	pH in arterial Blood	
	Label:	
	NLS_NOM_CONC_PH_ART	0x00027004
	Observed Value:	
	NOM_CONC_PH_ART	0x7004
PaCO2	Partial Pressure of arterial Carbon Dioxide	
	Label:	
	NLS_NOM_CONC_PCO2_ART	0x00027008
	Observed Value:	
	NOM_CONC_PCO2_ART	0x7008
PaO2	Partial O2 arterial	
	Label:	
	NLS_NOM_CONC_PO2_ART	0x0002700c
	Observed Value:	
	NOM_CONC_PO2_ART	0x700C
Hb	Hemoglobin in arterial Blood	
	Label:	
	NLS_NOM_CONC_HB_ART	0x00027014
	Observed Value:	
	NOM_CONC_HB_ART	0x7014
CaO2	Arterial Oxygen Content CaO2	
	Label:	
	NLS_NOM_CONC_HB_O2_ART	0x00027018
	Observed Value:	
	NOM_CONC_HB_O2_ART	0x7018
PHv	pH in venous Blood	
	Label:	
	NLS_NOM_CONC_PH_VEN	0x00027034
	Observed Value:	
	NOM_CONC_PH_VEN	0x7034
PvCO2	Partial CO2 in the venous blood	
	Label:	
	NLS_NOM_CONC_PCO2_VEN	0x00027038
	Observed Value:	
	NOM_CONC_PCO2_VEN	0x7038
PvO2	Partial O2 Venous	
	Label:	
	NLS_NOM_CONC_PO2_VEN	0x0002703c
	Observed Value:	
	NOM_CONC_PO2_VEN	0x703C
CvO2	Venous Oxygen Content	
	Label:	
	NLS_NOM_CONC_HB_O2_VEN	0x00027048
	Observed Value:	
	NOM_CONC_HB_O2_VEN	0x7048
UrNa	Natrium in Urine	
	Label:	

	NLS_NOM_CONC_NA_URINE	0x0002706c
	Observed Value:	
	NOM_CONC_NA_URINE	0x706C
SerNa	Natrium in Serum	
	Label:	
	NLS_NOM_CONC_NA_SERUM	0x000270d8
	Observed Value:	
	NOM_CONC_NA_SERUM	0x70D8
PH	pH in the Blood Plasma	
	Label:	
	NLS_NOM_CONC_PH_GEN	0x00027104
	Observed Value:	
	NOM_CONC_PH_GEN	0x7104
HCO3	Hydrocarbon concentration in Blood Plasma	
	Label:	
	NLS_NOM_CONC_HCO3_GEN	0x00027108
	Observed Value:	
	NOM_CONC_HCO3_GEN	0x7108
Na	Natrium (Sodium)	
	Label:	
	NLS_NOM_CONC_NA_GEN	0x0002710c
	Observed Value:	
	NOM_CONC_NA_GEN	0x710C
K	Kalium (Potassium)	
	Label:	
	NLS_NOM_CONC_K_GEN	0x00027110
	Observed Value:	
	NOM_CONC_K_GEN	0x7110
Glu	Glucose	
	Label:	
	NLS_NOM_CONC_GLU_GEN	0x00027114
	Observed Value:	
	NOM_CONC_GLU_GEN	0x7114
PCO2	Partial CO2	
	Label:	
	NLS_NOM_CONC_PCO2_GEN	0x00027140
	Observed Value:	
	NOM_CONC_PCO2_GEN	0x7140
PO2	Partial O2.	
	Label:	
	NLS_NOM_CONC_PO2_GEN	0x00027174
	Observed Value:	
	NOM_CONC_PO2_GEN	0x7174
Hct	Haematocrit	
	Label:	
	NLS_NOM_CONC_HCT_GEN	0x00027184
	Observed Value:	
	NOM_CONC_HCT_GEN	0x7184
BE	Base Excess of Blood	
	Label:	
	NLS_NOM_BASE_EXCESS_BLD_ART	0x0002716c
	Observed Value:	
	NOM_BASE_EXCESS_BLD_ART	0x716C
PVRI	Pulmonary vascular Resistance PVRI	
	Label:	
	NLS_NOM_RES_VASC_PULM_INDEX	0x0002f067
	Observed Value:	
	NOM_RES_VASC_PULM_INDEX	0xF067
LCWI	Left Cardiac Work Index	
	Label:	
	NLS_NOM_WK_CARD_LEFT_INDEX	0x0002f068
	Observed Value:	
	NOM_WK_CARD_LEFT_INDEX	0xF068
RCWI	Right Cardiac Work Index	
	Label:	

	NLS_NOM_WK_CARD_RIGHT_INDEX	0x0002f069
	Observed Value:	
	NOM_WK_CARD_RIGHT_INDEX	0xF069
VO2I	Oxygen Consumption Index VO2I	
	Label:	
	NLS_NOM_SAT_O2_CONSUMP_INDEX	0x0002f06a
	Observed Value:	
	NOM_SAT_O2_CONSUMP_INDEX	0xF06A
PB	Barometric Pressure = Ambient Pressure	
	Label:	
	NLS_NOM_PRESS_AIR_AMBIENT	0x0002f06b
	Observed Value:	
	NOM_PRESS_AIR_AMBIENT	0xF06B
DO2	Oxygen Availability DO2	
	Label:	
	NLS_NOM_SAT_O2_DELIVER	0x0002f06d
	Observed Value:	
	NOM_SAT_O2_DELIVER	0xF06D
DO2I	Oxygen Availability Index	
	Label:	
	NLS_NOM_SAT_O2_DELIVER_INDEX	0x0002f06e
	Observed Value:	
	NOM_SAT_O2_DELIVER_INDEX	0xF06E
O2ER	Oxygen Extraction Ratio	
	Label:	
	NLS_NOM_RATIO_SAT_O2_CONSUMP_DELIVER	0x0002f06f
	Observed Value:	
	NOM_RATIO_SAT_O2_CONSUMP_DELIVER	0xF06F
Qs/Qt	Percent Alveolarvenous Shunt Qs/Qt	
	Label:	
	NLS_NOM_RATIO_ART_VEN_SHUNT	0x0002f070
	Observed Value:	
	NOM_RATIO_ART_VEN_SHUNT	0xF070
LI	Light Intenisty. SvO2	
	Label:	
	NLS_NOM_INTENS_LIGHT	0x0002f072
	Observed Value:	
	NOM_INTENS_LIGHT	0xF072
ETVI	ExtraVascular Thermo Volume Index. Cardiac Output.	
	Label:	
	NLS_NOM_VOL_THERMO_EXTRA_VASC_INDEX	0x0002f07a
	Observed Value:	
	NOM_VOL_THERMO_EXTRA_VASC_INDEX	0xF07A
Cl	Chloride	
	Label:	
	NLS_NOM_CONC_CHLORIDE_GEN	0x00027168
	Observed Value:	
	NOM_CONC_CHLORIDE_GEN	0x7168
BUN	Blood Urea Nitrogen	
	Label:	
	NLS_NOM_CONC_BLD_UREA_NITROGEN	0x0002f08f
	Observed Value:	
	NOM_CONC_BLD_UREA_NITROGEN	0xF08F
Ca-VO2	Arteriovenous Oxygen Difference Ca-vO2	
	Label:	
	NLS_NOM_CONC_DIFF_HB_O2_ATR_VEN	0x0002f092
	Observed Value:	
	NOM_CONC_DIFF_HB_O2_ATR_VEN	0xF092
BSA	Body Surface Area	
	Label:	
	NLS_NOM_AREA_BODY_SURFACE	0x0002f071
	Observed Value:	
	NOM_AREA_BODY_SURFACE	0xF071
Weight	Patient Weight	
	Label:	

```
            NLS_NOM_PAT_WEIGHT                          0x0002f093
            Observed Value:
            NOM_PAT_WEIGHT                              0xF093
Height      Patient Height
            Label:
            NLS_NOM_PAT_HEIGHT                          0x0002f094
            Observed Value:
            NOM_PAT_HEIGHT                              0xF094
P1          non-specific label for Pressure 1
            Label:
            NLS_NOM_EMFC_P1                             0x04010030
            Observed Value:
            NOM_PRESS_BLD                               0x4A00
P1s         non-specific label for Pressure 1 Systolic
            Label:
            NLS_NOM_EMFC_P1_SYS                         0x04010031
            Observed Value:
            NOM_PRESS_BLD                               0x4A00
P1d         non-specific label for Pressure 1 Diastolic
            Label:
            NLS_NOM_EMFC_P1_DIA                         0x04010032
            Observed Value:
            NOM_PRESS_BLD                               0x4A00
P1m         non-specific label for Pressure 1 Mean
            Label:
            NLS_NOM_EMFC_P1_MEAN                        0x04010033
            Observed Value:
            NOM_PRESS_BLD                               0x4A00
P2          non-specific label for Pressure 2
            Label:
            NLS_NOM_EMFC_P2                             0x04010034
            Observed Value:
            NOM_PRESS_BLD                               0x4A00
P2    s     non-specific label for Pressure 2 Systolic
            Label:
            NLS_NOM_EMFC_P2_SYS                         0x04010035
            Observed Value:
            NOM_PRESS_BLD                               0x4A00
P2    d     non-specific label for Pressure 2 Diastolic
            Label:
            NLS_NOM_EMFC_P2_DIA                         0x04010036
            Observed Value:
            NOM_PRESS_BLD                               0x4A00
P2    m     non-specific label for Pressure 2 Mean
            Label:
            NLS_NOM_EMFC_P2_MEAN                        0x04010037
            Observed Value:
            NOM_PRESS_BLD                               0x4A00
P3          non-specific label for Pressure 3
            Label:
            NLS_NOM_EMFC_P3                             0x04010038
            Observed Value:
            NOM_PRESS_BLD                               0x4A00
P3s         non-specific label for Pressure 3 Systolic
            Label:
            NLS_NOM_EMFC_P3_SYS                         0x04010039
            Observed Value:
            NOM_PRESS_BLD                               0x4A00
P3d         non-specific label for Pressure 3 Diastolic
            Label:
            NLS_NOM_EMFC_P3_DIA                         0x0401003a
            Observed Value:
            NOM_PRESS_BLD                               0x4A00
P3m         non-specific label for Pressure 3 Mean
            Label:
```

	NLS_NOM_EMFC_P3_MEAN	0x0401003b
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P4	non-specific label for Pressure 4	
	Label:	
	NLS_NOM_EMFC_P4	0x0401003c
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P4s	non-specific label for Pressure 4 Systolic	
	Label:	
	NLS_NOM_EMFC_P4_SYS	0x0401003d
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P4d	non-specific label for Pressure 4 Diastolic	
	Label:	
	NLS_NOM_EMFC_P4_DIA	0x0401003e
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P4m	non-specific label for Pressure 4 Mean	
	Label:	
	NLS_NOM_EMFC_P4_MEAN	0x0401003f
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P5	non-specific label for Pressure 5	
	Label:	
	NLS_NOM_EMFC_P5	0x04010400
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P5s	non-specific label for Pressure 5 Systolic	
	Label:	
	NLS_NOM_EMFC_P5_SYS	0x04010401
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P5d	non-specific label for Pressure 5 Diastolic	
	Label:	
	NLS_NOM_EMFC_P5_DIA	0x04010402
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P5m	non-specific label for Pressure 5 Mean	
	Label:	
	NLS_NOM_EMFC_P5_MEAN	0x04010403
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P6	non-specific label for Pressure 6	
	Label:	
	NLS_NOM_EMFC_P6	0x04010404
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P6s	non-specific label for Pressure 6 Systolic	
	Label:	
	NLS_NOM_EMFC_P6_SYS	0x04010405
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P6d	non-specific label for Pressure 6 Diastolic	
	Label:	
	NLS_NOM_EMFC_P6_DIA	0x04010406
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P6m	non-specific label for Pressure 6 Mean	
	Label:	
	NLS_NOM_EMFC_P6_MEAN	0x04010407
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P7	non-specific label for Pressure 7	
	Label:	

	NLS_NOM_EMFC_P7	0x04010408
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P7s	non-specific label for Pressure 7 Systolic	
	Label:	
	NLS_NOM_EMFC_P7_SYS	0x04010409
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P7d	non-specific label for Pressure 7 Diastolic	
	Label:	
	NLS_NOM_EMFC_P7_DIA	0x0401040a
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P7m	non-specific label for Pressure 7 Mean	
	Label:	
	NLS_NOM_EMFC_P7_MEAN	0x0401040b
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P8	non-specific label for Pressure 8	
	Label:	
	NLS_NOM_EMFC_P8	0x0401040c
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P8s	non-specific label for Pressure 8 Systolic	
	Label:	
	NLS_NOM_EMFC_P8_SYS	0x0401040d
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P8d	non-specific label for Pressure 8 Diastolic	
	Label:	
	NLS_NOM_EMFC_P8_DIA	0x0401040e
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
P8m	non-specific label for Pressure 8 Mean	
	Label:	
	NLS_NOM_EMFC_P8_MEAN	0x0401040f
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
IUP	Intra-Uterine Pressure	
	Label:	
	NLS_NOM_EMFC_IUP	0x04010054
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
IUPs	Intra-Uterine Pressure Systolic	
	Label:	
	NLS_NOM_EMFC_IUP_SYS	0x04010055
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
IUPd	Intra-Uterine Pressure Diastolic	
	Label:	
	NLS_NOM_EMFC_IUP_DIA	0x04010056
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
IUPm	Intra-Uterine Pressure Mean	
	Label:	
	NLS_NOM_EMFC_IUP_MEAN	0x04010057
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
VPB	VPB Rate	
	Label:	
	NLS_NOM_EMFC_VPB	0x04010088
	Observed Value:	
	NOM_ECG_V_P_C_FREQ	0x4268
AUX	Auxiliary Wave/Parameter	
	Label:	

	NLS_NOM_EMFC_AUX	0x040100b4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
ST1	ST parameter - value ST1	
	Label:	
	NLS_NOM_EMFC_ST1	0x040100b8
	Observed Value:	
	NOM_ECG_AMPL_ST	0x0300
ST2	ST parameter - value ST2	
	Label:	
	NLS_NOM_EMFC_ST2	0x040100c4
	Observed Value:	
	NOM_ECG_AMPL_ST	0x0300
ST3	ST parameter - value ST3	
	Label:	
	NLS_NOM_EMFC_ST3	0x040100c8
	Observed Value:	
	NOM_ECG_AMPL_ST	0x0300
BUN/cr	BUN Creatinine Ratio	
	Label:	
	NLS_NOM_EMFC_BUN_PER_cr	0x04010110
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
CH2O	Free Water Clearance	
	Label:	
	NLS_NOM_EMFC_CH2O	0x04010118
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
COsm	Osmolar Clearance	
	Label:	
	NLS_NOM_EMFC_COsm	0x04010120
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
CrCl	Creatinine Clearance	
	Label:	
	NLS_NOM_EMFC_CrCl	0x04010124
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
FeNa	Fractional Excretion of Sodium	
	Label:	
	NLS_NOM_EMFC_FeNa	0x0401012c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
IMV	Intermittent Mandatory Ventilation	
	Label:	
	NLS_NOM_EMFC_IMV	0x04010138
	Observed Value:	
	NOM_VENT_MODE_MAND_INTERMIT	0xD02A
PlOsm	Plasma Osmolarity	
	Label:	
	NLS_NOM_EMFC_PlOsm	0x04010164
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SCreat	Serum Creatinine	
	Label:	
	NLS_NOM_EMFC_SCreat	0x04010180
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
U/POsm	Urine Plasma Osmolarity Ratio	
	Label:	
	NLS_NOM_EMFC_U_PER_POsm	0x04010198
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
U/SCr	Urine Serum Creatinine Ratio	
	Label:	

	NLS_NOM_EMFC_U_PER_SCr	0x0401019c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UCreat	Urine Creatinine	
	Label:	
	NLS_NOM_EMFC_UCreat	0x040101a0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrK	Urine Potassium	
	Label:	
	NLS_NOM_EMFC_UrK	0x040101a4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrKEx	Urinary Potassium Excretion	
	Label:	
	NLS_NOM_EMFC_UrKEx	0x040101a8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrNa/K	Urine Sodium/Potassium Ratio	
	Label:	
	NLS_NOM_EMFC_UrNa_PER_K	0x040101b0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrNaEx	Urine Sodium Excretion	
	Label:	
	NLS_NOM_EMFC_UrNaEx	0x040101b4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrOsm	Urine Osmolarity	
	Label:	
	NLS_NOM_EMFC_UrOsm	0x040101b8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrVol	Urine Volume	
	Label:	
	NLS_NOM_EMFC_UrVol	0x040101bc
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
T3	Temperature non-specific label - third temperature	
	Label:	
	NLS_NOM_EMFC_T3	0x04010410
	Observed Value:	
	NOM_TEMP	0x4B48
T4	Temperature non-specific label - fourth temperature	
	Label:	
	NLS_NOM_EMFC_T4	0x04010414
	Observed Value:	
	NOM_TEMP	0x4B48
Length	Length for neonatal/pediatric	
	Label:	
	NLS_NOM_EMFC_Length	0x04010420
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
G.Age	Gestational age for neonatal	
	Label:	
	NLS_NOM_EMFC_G_Age	0x04010428
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Cal	Calories	
	Label:	
	NLS_NOM_EMFC_Cal	0x0401042c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
I/O	Input/Output totals	
	Label:	

	NLS_NOM_EMFC_I_O	0x04010430
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Apgar	Apgar scores for neonatal	
	Label:	
	NLS_NOM_EMFC_Apgar	0x04010434
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
BSA(B)	BSA formula: Boyd	
	Label:	
	NLS_NOM_EMFC_BSA_B	0x0401043c
	Observed Value:	
	NOM_AREA_BODY_SURFACE	0xF071
BSA(D)	BSA formula: Dubois	
	Label:	
	NLS_NOM_EMFC_BSA_D	0x04010440
	Observed Value:	
	NOM_AREA_BODY_SURFACE	0xF071
PVcP	Pressure Ventilation Control Pressure	
	Label:	
	NLS_NOM_EMFC_PVcP	0x0401046c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rdyn	Dynamic Lung Resistance	
	Label:	
	NLS_NOM_EMFC_Rdyn	0x04010480
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
NgInsP	Negative Inspiratory Pressure	
	Label:	
	NLS_NOM_EMFC_NgInsP	0x04010484
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SpPkFl	Spontaneous Peak Flow	
	Label:	
	NLS_NOM_EMFC_SpPkFl	0x0401048c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SpAWRR	Spontaneous Airway Respiration Rate	
	Label:	
	NLS_NOM_EMFC_SpAWRR	0x04010510
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
PlGain	Pleth Gain	
	Label:	
	NLS_NOM_EMFC_PlGain	0x04010514
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
AwN2O	Airway N2O concentration	
	Label:	
	NLS_NOM_EMFC_AwN2O	0x04010518
	Observed Value:	
	NOM_CONC_AWAY_N2O	0x51F0
fgAGT	Fresh gas Anesthetic Agent	
	Label:	
	NLS_NOM_EMFC_fgAGT	0x04010520
	Observed Value:	
	NOM_CONC_AWAY_AGENT	0x5388
O2EI	Oxygen Extraction Index	
	Label:	
	NLS_NOM_EMFC_O2EI	0x0401052c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
REF	Right Heart Ejection Fraction	
	Label:	

	NLS_NOM_EMFC_REF	0x04010530
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
EDV	End Diastolic Volume	
	Label:	
	NLS_NOM_EMFC_EDV	0x04010534
	Observed Value:	
	NOM_VOL_GLOBAL_END_DIA	0xF044
ESV	End Systolic Volume	
	Label:	
	NLS_NOM_EMFC_ESV	0x04010538
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
EDVI	End Diastolic Volume Index	
	Label:	
	NLS_NOM_EMFC_EDVI	0x0401053c
	Observed Value:	
	NOM_VOL_GLOBAL_END_DIA_INDEX	0xF045
ESVI	End Systolic Volume Index	
	Label:	
	NLS_NOM_EMFC_ESVI	0x04010540
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SaO2	Oxygen Saturation measured in the Artery	
	Label:	
	NLS_NOM_EMFC_SaO2	0x04010548
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
DABP	Duration Above Base Pressure	
	Label:	
	NLS_NOM_EMFC_DABP	0x0401054c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
RiseTi	Rise Time	
	Label:	
	NLS_NOM_EMFC_RiseTi	0x04010550
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
BasePr	Base Pressure	
	Label:	
	NLS_NOM_EMFC_BasePr	0x04010554
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
HFVAmp	High Frequency Ventilation Amplitude	
	Label:	
	NLS_NOM_EMFC_HFVAmp	0x0401055c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrUrea	Urine Urea	
	Label:	
	NLS_NOM_EMFC_UrUrea	0x04010580
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrpH	pH value in the Urine	
	Label:	
	NLS_NOM_EMFC_UrpH	0x04010584
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
tCO2	total of CO2 - result of Blood gas Analysis	
	Label:	
	NLS_NOM_EMFC_tCO2	0x04010588
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
tBili	total Bilirubin	
	Label:	

	NLS_NOM_EMFC_tBili	0x0401058c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SerGlc	Glucose in Serum	
	Label:	
	NLS_NOM_EMFC_SerGlc	0x04010590
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrGlc	Glucose in Urine	
	Label:	
	NLS_NOM_EMFC_UrGlc	0x04010594
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
dBili	direct Bilirubin	
	Label:	
	NLS_NOM_EMFC_dBili	0x04010598
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SerCa	Calcium in Serum	
	Label:	
	NLS_NOM_EMFC_SerCa	0x0401059c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
tSerCa	total of Calcium in Serum	
	Label:	
	NLS_NOM_EMFC_tSerCa	0x040105a0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SerMg	Magnesium in Serum	
	Label:	
	NLS_NOM_EMFC_SerMg	0x040105a4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SerPho	Phosphat in Serum	
	Label:	
	NLS_NOM_EMFC_SerPho	0x040105a8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SerK	Kalium (Potassium) in Serum	
	Label:	
	NLS_NOM_EMFC_SerK	0x040105ac
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SerCl	Clorid in Serum	
	Label:	
	NLS_NOM_EMFC_SerCl	0x040105b0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SerAlb	Albumine in Serum	
	Label:	
	NLS_NOM_EMFC_SerAlb	0x040105b4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrCl	Clorid in Urine	
	Label:	
	NLS_NOM_EMFC_UrCl	0x040105b8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SerGlo	Globulin in Serum	
	Label:	
	NLS_NOM_EMFC_SerGlo	0x040105bc
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SerPro	(Total) Protein in Serum	
	Label:	

```
            NLS_NOM_EMFC_SerPro                         0x040105c0
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
SrUrea      Serum Urea
            Label:
            NLS_NOM_EMFC_SrUrea                         0x040105c4
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
WBC         White Blood Count (leucocyte count)
            Label:
            NLS_NOM_EMFC_WBC                            0x040105c8
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
RBC         Red Blood Count (erithrocyte count)
            Label:
            NLS_NOM_EMFC_RBC                            0x040105cc
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
Plts        Platelets  (thrombocyte count)
            Label:
            NLS_NOM_EMFC_Plts                           0x040105d0
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
MCV         Mean Corpuscular Volume
            Label:
            NLS_NOM_EMFC_MCV                            0x040105d4
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
MCH         Mean Corpuscular Hemoglobin. Is the erithrocyte hemoglobin content
            Label:
            NLS_NOM_EMFC_MCH                            0x040105d8
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
MCHC        Mean Corpuscular Hemoglobin Concentration
            Label:
            NLS_NOM_EMFC_MCHC                           0x040105dc
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
PTT         Partial Thromboplastin Time
            Label:
            NLS_NOM_EMFC_PTT                            0x040105e0
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
PT          Prothrombin Time
            Label:
            NLS_NOM_EMFC_PT                             0x040105e4
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
TT          Thrombin Time
            Label:
            NLS_NOM_EMFC_TT                             0x040105e8
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
AP          Alkalische Phosphatase
            Label:
            NLS_NOM_EMFC_AP                             0x040105f0
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
alphaA      Alpha Amylase
            Label:
            NLS_NOM_EMFC_alphaA                         0x040105f4
            Observed Value:
            NOM_METRIC_NOS                              0xEFFF
CHE         Cholesterinesterase
            Label:
```

	NLS_NOM_EMFC_CHE	0x040105f8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SerCK	Creatinin Kinase	
	Label:	
	NLS_NOM_EMFC_SerCK	0x040105fc
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
CK-MB	Creatine Cinase of type "muscle-brain	
	Label:	
	NLS_NOM_EMFC_CK_MB	0x04010600
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
CK-MM	Creatine Cinase of type "muscle	"
	Label:	
	NLS_NOM_EMFC_CK_MM	0x04010604
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
GGT	Gamma GT = Gamma Glutamyltranspeptidase	
	Label:	
	NLS_NOM_EMFC_GGT	0x04010608
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
GOT	Glutamic Oxaloacetic Transaminase	
	Label:	
	NLS_NOM_EMFC_GOT	0x0401060c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
GPT	Glutamic-Pyruvic-Transaminase	
	Label:	
	NLS_NOM_EMFC_GPT	0x04010610
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Fe	Ferrum	
	Label:	
	NLS_NOM_EMFC_Fe	0x04010614
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Chol	Cholesterin	
	Label:	
	NLS_NOM_EMFC_Chol	0x04010618
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
TGL	Triglyzeride	
	Label:	
	NLS_NOM_EMFC_TGL	0x0401061c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrPro	(Total) Protein in Urine	
	Label:	
	NLS_NOM_EMFC_UrPro	0x04010620
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrCa	Calzium in Urine	
	Label:	
	NLS_NOM_EMFC_UrCa	0x04010624
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
CO-Hb	Carboxy Hemoglobin	
	Label:	
	NLS_NOM_EMFC_CO_Hb	0x04010628
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
HbF	Fetal Hemoglobin	
	Label:	

```
                NLS_NOM_EMFC_HbF                                0x0401062c
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
Met-Hb          MetHemoglobin
                Label:
                NLS_NOM_EMFC_Met_Hb                             0x04010630
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
tProt           Total Protein
                Label:
                NLS_NOM_EMFC_tProt                              0x04010634
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
LDH             Lactate Dehydrogenase
                Label:
                NLS_NOM_EMFC_LDH                                0x04010638
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
AST             Aspartin - Aminotransferase
                Label:
                NLS_NOM_EMFC_AST                                0x0401063c
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
ALP             Alveolarproteinose Rosen-Castleman-Liebow- Syndrom
                Label:
                NLS_NOM_EMFC_ALP                                0x04010640
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
RC              Reticulocyte Count
                Label:
                NLS_NOM_EMFC_RC                                 0x04010644
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
CT              Coagulation Time
                Label:
                NLS_NOM_EMFC_CT                                 0x04010648
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
ESR             Erithrocyte sedimentation rate
                Label:
                NLS_NOM_EMFC_ESR                                0x0401064c
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
PCV             Packed Cell Volume
                Label:
                NLS_NOM_EMFC_PCV                                0x04010650
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
KCT             Kaolin cephalin time
                Label:
                NLS_NOM_EMFC_KCT                                0x04010654
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
K+              Kalium (Potassium)
                Label:
                NLS_NOM_EMFC_KPLUS                              0x0401065c
                Observed Value:
                NOM_CONC_K_GEN                                  0x7110
Rexp            Expiratory Resistance
                Label:
                NLS_NOM_EMFC_Rexp                               0x04010664
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
AWV             Airway Volume Wave
                Label:
```

	NLS_NOM_EMFC_AWV	0x04010668
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
ExpTi	Expiratory Time	
	Label:	
	NLS_NOM_EMFC_ExpTi	0x0401066c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rinsp	Inspiratory Resistance	
	Label:	
	NLS_NOM_EMFC_Rinsp	0x04010670
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
inPkFl	Inspiratory Peak Flow	
	Label:	
	NLS_NOM_EMFC_inPkFl	0x04010674
	Observed Value:	
	NOM_PRESS_AWAY_INSP_MAX	0x5109
Pmax	Maximum Pressure during a breathing cycle	
	Label:	
	NLS_NOM_EMFC_Pmax	0x04010678
	Observed Value:	
	NOM_PRESS_AWAY_INSP_MAX	0x5109
Pmin	Minimum Pressure during a breathing cycle	
	Label:	
	NLS_NOM_EMFC_Pmin	0x0401067c
	Observed Value:	
	NOM_PRESS_AWAY_MIN	0x50F2
AccVol	Infusion Pump Accumulated volume. Measured value	
	Label:	
	NLS_NOM_EMFC_AccVol	0x04010680
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
i-eN2O	Inspired - EndTidal N2O	
	Label:	
	NLS_NOM_EMFC_i_eN2O	0x04010688
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
i-eHAL	Inspired - EndTidal Halothane	
	Label:	
	NLS_NOM_EMFC_i_eHAL	0x0401068c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
i-eENF	Inspired - EndTidal Enfluran	
	Label:	
	NLS_NOM_EMFC_i_eENF	0x04010690
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
i-eISO	Inspired - EndTidal Isofluran	
	Label:	
	NLS_NOM_EMFC_i_eISO	0x04010694
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
i-eSEV	Inspired - EndTidal Sevofluran	
	Label:	
	NLS_NOM_EMFC_i_eSEV	0x04010698
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
i-eDES	Inspired - EndTidal Desfluran	
	Label:	
	NLS_NOM_EMFC_i_eDES	0x0401069c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
i-eAGT	Inspired - EndTidal Agent	
	Label:	

	NLS_NOM_EMFC_i_eAGT	0x040106a0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
i-eO2	Inspired - EndTidal O2	
	Label:	
	NLS_NOM_EMFC_i_eO2	0x040106a4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
cktO2	O2 measured in the Patient Circuit	
	Label:	
	NLS_NOM_EMFC_cktO2	0x040106a8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Patm	Barometric pressure measured in the Sample Chamber	
	Label:	
	NLS_NOM_EMFC_Patm	0x040106ac
	Observed Value:	
	NOM_PRESS_AIR_AMBIENT	0xF06B
TVin	inspired Tidal Volume	
	Label:	
	NLS_NOM_EMFC_TVin	0x040106b0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
TVex	expired Tidal Volume	
	Label:	
	NLS_NOM_EMFC_TVex	0x040106b4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
MV	Minute Volume as delivered by the Ohmeda Ventilator	
	Label:	
	NLS_NOM_EMFC_MV	0x040106b8
	Observed Value:	
	NOM_VOL_MINUTE_AWAY	0x5148
Paw	Airway Pressure numeric value. Used by the Ohmeda ventilator.	
	Label:	
	NLS_NOM_EMFC_Paw	0x040106bc
	Observed Value:	
	NOM_PRESS_AWAY	0x50F0
Pmean	Mean Airway Pressure. Used by the Ohmeda Ventilator	
	Label:	
	NLS_NOM_EMFC_Pmean	0x040106c0
	Observed Value:	
	NOM_PRESS_AWAY_INSP_MEAN	0x510B
RRaw	Airway Respiration Rate. Used by the Ohmeda Ventilator.	
	Label:	
	NLS_NOM_EMFC_RRaw	0x040106c4
	Observed Value:	
	NOM_AWAY_RESP_RATE	0x5012
Ppeak	Peak pressure	
	Label:	
	NLS_NOM_EMFC_Ppeak	0x040106cc
	Observed Value:	
	NOM_PRESS_AWAY_INSP_MAX	0x5109
HFMVin	Inspired High Frequency Mandatory Minute Volume	
	Label:	
	NLS_NOM_EMFC_HFMVin	0x040106d8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
DCO2	High Frequency Gas Transport Coefficient value	
	Label:	
	NLS_NOM_EMFC_DCO2	0x040106dc
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SpTVex	Spontaenous Expired Tidal Volume	
	Label:	

	NLS_NOM_EMFC_SpTVex	0x040106e0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
HFTVin	Inspired High Frequency Tidal Volume	
	Label:	
	NLS_NOM_EMFC_HFTVin	0x040106e4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
HFVTV	High Frequency Fraction Ventilation Tidal Volume	
	Label:	
	NLS_NOM_EMFC_HFVTV	0x040106e8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
extHR	denotes a Heart Rate received from an external device	
	Label:	
	NLS_NOM_EMFC_extHR	0x04010700
	Observed Value:	
	NOM_ECG_CARD_BEAT_RATE	0x4182
Rf-I	ST Reference Value for Lead I	
	Label:	
	NLS_NOM_EMFC_Rf_I	0x04010734
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rf-II	ST Reference Value for Lead II	
	Label:	
	NLS_NOM_EMFC_Rf_II	0x04010738
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rf-III	ST Reference Value for Lead III	
	Label:	
	NLS_NOM_EMFC_Rf_III	0x0401073c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rf-aVR	ST Reference Value for Lead aVR	
	Label:	
	NLS_NOM_EMFC_Rf_aVR	0x04010740
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rf-aVL	ST Reference Value for Lead aVL	
	Label:	
	NLS_NOM_EMFC_Rf_aVL	0x04010744
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rf-aVF	ST Reference Value for Lead aVF	
	Label:	
	NLS_NOM_EMFC_Rf_aVF	0x04010748
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rf-V1	ST Reference Value for Lead V1	
	Label:	
	NLS_NOM_EMFC_Rf_V1	0x0401074c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rf-V2	ST Reference Value for Lead V2	
	Label:	
	NLS_NOM_EMFC_Rf_V2	0x04010750
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rf-V3	ST Reference Value for Lead V3	
	Label:	
	NLS_NOM_EMFC_Rf_V3	0x04010754
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rf-V4	ST Reference Value for Lead V4	
	Label:	

	NLS_NOM_EMFC_Rf_V4	0x04010758
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rf-V5	ST Reference Value for Lead V5	
	Label:	
	NLS_NOM_EMFC_Rf_V5	0x0401075c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Rf-V6	ST Reference Value for Lead V6	
	Label:	
	NLS_NOM_EMFC_Rf_V6	0x04010760
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SpO22	SpO2 from the second SpO2/PLETH module	
	Label:	
	NLS_NOM_EMFC_SpO2_2	0x040107a0
	Observed Value:	
	NOM_PULS_OXIM_SAT_O2	0x4BB8
PERF2	PERF from the second SpO2/PLETH module	
	Label:	
	NLS_NOM_EMFC_PERF2	0x040107a4
	Observed Value:	
	NOM_PULS_OXIM_PERF_REL	0x4BB0
LT%AL	Percent Alpha , Left (LT) Side	
	Label:	
	NLS_NOM_EMFC_LT_PCT_AL	0x040107d0
	Observed Value:	
	NOM_EEG_PWR_SPEC_ALPHA_REL	0x59D4
LT%BE	Percent Beta, Left Side	
	Label:	
	NLS_NOM_EMFC_LT_PCT_BE	0x040107d4
	Observed Value:	
	NOM_EEG_PWR_SPEC_BETA_REL	0x59D8
LT%DL	Percent Delta, Left Side	
	Label:	
	NLS_NOM_EMFC_LT_PCT_DL	0x040107d8
	Observed Value:	
	NOM_EEG_PWR_SPEC_DELTA_REL	0x59DC
LT%TH	Percent Theta, Left Side	
	Label:	
	NLS_NOM_EMFC_LT_PCT_TH	0x040107dc
	Observed Value:	
	NOM_EEG_PWR_SPEC_THETA_REL	0x59E0
LTAL	Absolute Alpha, Left Side	
	Label:	
	NLS_NOM_EMFC_LT_AL	0x040107e0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
LTBE	Absolute Beta, Left Side	
	Label:	
	NLS_NOM_EMFC_LT_BE	0x040107e4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
LTDL	Absolute Delta, Left Side	
	Label:	
	NLS_NOM_EMFC_LT_DL	0x040107e8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
LTTH	Absolute Theta, Left Side	
	Label:	
	NLS_NOM_EMFC_LT_TH	0x040107ec
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
LTMDF	Mean Dominant Frequency, Left Side	
	Label:	

	NLS_NOM_EMFC_LT_MDF	0x040107f4
	Observed Value:	
	NOM_EEG_FREQ_PWR_SPEC_CRTX_DOM_MEAN	0x597C
LTMPF	Median Power Frequency, Left Side	
	Label:	
	NLS_NOM_EMFC_LT_MPF	0x040107f8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
LTPPF	Peak Power Frequency, Left Side	
	Label:	
	NLS_NOM_EMFC_LT_PPF	0x040107fc
	Observed Value:	
	NOM_EEG_FREQ_PWR_SPEC_CRTX_PEAK	0x5984
LTSEF	Spectral Edge Frequency, Left Side	
	Label:	
	NLS_NOM_EMFC_LT_SEF	0x04010800
	Observed Value:	
	NOM_EEG_FREQ_PWR_SPEC_CRTX_SPECTRAL_EDGE	0x5988
LTTP	Total Power, Left Side	
	Label:	
	NLS_NOM_EMFC_LT_TP	0x04010804
	Observed Value:	
	NOM_EEG_PWR_SPEC_TOT	0x59B8
LSCALE	Scale of the Left Channel EEG wave	
	Label:	
	NLS_NOM_EMFC_LSCALE	0x04010808
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
RT%AL	Percent Alpha , Right (RT) Side	
	Label:	
	NLS_NOM_EMFC_RT_PCT_AL	0x0401080c
	Observed Value:	
	NOM_EEG_PWR_SPEC_ALPHA_REL	0x59D4
RT%BE	Percent Beta, Right Side	
	Label:	
	NLS_NOM_EMFC_RT_PCT_BE	0x04010810
	Observed Value:	
	NOM_EEG_PWR_SPEC_BETA_REL	0x59D8
RT%DL	Percent Delta, Right Side	
	Label:	
	NLS_NOM_EMFC_RT_PCT_DL	0x04010814
	Observed Value:	
	NOM_EEG_PWR_SPEC_DELTA_REL	0x59DC
RT%TH	Percent Theta, Right Side	
	Label:	
	NLS_NOM_EMFC_RT_PCT_TH	0x04010818
	Observed Value:	
	NOM_EEG_PWR_SPEC_THETA_REL	0x59E0
RTAL	Absolute Alpha, Right Side	
	Label:	
	NLS_NOM_EMFC_RT_AL	0x0401081c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
RTBE	Absolute Beta, Right Side	
	Label:	
	NLS_NOM_EMFC_RT_BE	0x04010820
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
RTDL	Absolute Delta, Right Side	
	Label:	
	NLS_NOM_EMFC_RT_DL	0x04010824
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
RTTH	Absolute Theta, Right Side	
	Label:	

	NLS_NOM_EMFC_RT_TH	0x04010828
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
RTMDF	Mean Dominant Frequency, Right Side	
	Label:	
	NLS_NOM_EMFC_RT_MDF	0x04010830
	Observed Value:	
	NOM_EEG_FREQ_PWR_SPEC_CRTX_DOM_MEAN	0x597C
RTMPF	Median Power Frequency, Right Side	
	Label:	
	NLS_NOM_EMFC_RT_MPF	0x04010834
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
RTPPF	Peak Power Frequency, Right Side	
	Label:	
	NLS_NOM_EMFC_RT_PPF	0x04010838
	Observed Value:	
	NOM_EEG_FREQ_PWR_SPEC_CRTX_PEAK	0x5984
RTSEF	Spectral Edge Frequency, Right Side	
	Label:	
	NLS_NOM_EMFC_RT_SEF	0x0401083c
	Observed Value:	
	NOM_EEG_FREQ_PWR_SPEC_CRTX_SPECTRAL_EDGE	0x5988
RTTP	Total Power, Right Side	
	Label:	
	NLS_NOM_EMFC_RT_TP	0x04010840
	Observed Value:	
	NOM_EEG_PWR_SPEC_TOT	0x59B8
RSCALE	Scale of the Right Channel EEG wave	
	Label:	
	NLS_NOM_EMFC_RSCALE	0x04010844
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
DPosP	Duration of Positive Pressure	
	Label:	
	NLS_NOM_EMFC_DPosP	0x04010848
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
RRsync	Sync Breath Rate	
	Label:	
	NLS_NOM_EMFC_RRsync	0x0401084c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
RRmech	Mechanical Breath Rate	
	Label:	
	NLS_NOM_EMFC_RRmech	0x04010850
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
fgDES	fresh gas agent for DESflurane	
	Label:	
	NLS_NOM_EMFC_fgDES	0x04010854
	Observed Value:	
	NOM_CONC_AWAY_DESFL	0x51D8
fgSEV	fresh gas agent for SEVoflurane	
	Label:	
	NLS_NOM_EMFC_fgSEV	0x04010858
	Observed Value:	
	NOM_CONC_AWAY_SEVOFL	0x51E4
fgHAL	fresh gas agent for HALothane	
	Label:	
	NLS_NOM_EMFC_fgHAL	0x0401085c
	Observed Value:	
	NOM_CONC_AWAY_HALOTH	0x51E0
fgENF	fresh gas agent for ENFlurane	
	Label:	

	NLS_NOM_EMFC_fgENF	0x04010860
	Observed Value:	
	NOM_CONC_AWAY_ENFL	0x51DC
fgISO	fresh gas agent for ISOflurane	
	Label:	
	NLS_NOM_EMFC_fgISO	0x04010864
	Observed Value:	
	NOM_CONC_AWAY_ISOFL	0x51E8
fgN2O	N2O concentration in the fresh gas line	
	Label:	
	NLS_NOM_EMFC_fgN2O	0x04010868
	Observed Value:	
	NOM_CONC_AWAY_N2O	0x51F0
fgO2	Oxygen concentration in the fresh gas line	
	Label:	
	NLS_NOM_EMFC_fgO2	0x0401086c
	Observed Value:	
	NOM_CONC_AWAY_O2	0x5164
AGTLev	Liquid level in the anesthetic agent bottle	
	Label:	
	NLS_NOM_EMFC_AGTLev	0x04010870
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
ISOLev	Liquid level in the ISOflurane bottle	
	Label:	
	NLS_NOM_EMFC_ISOLev	0x04010874
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
ENFLev	Liquid level in the ENFlurane bottle	
	Label:	
	NLS_NOM_EMFC_ENFLev	0x04010878
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
HALLev	Liquid level in the HALothane bottle	
	Label:	
	NLS_NOM_EMFC_HALLev	0x0401087c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
DESLev	Liquid level in the DESflurane bottle	
	Label:	
	NLS_NOM_EMFC_DESLev	0x04010880
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SEVLev	Liquid level in the SEVoflurane bottle	
	Label:	
	NLS_NOM_EMFC_SEVLev	0x04010884
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
BP	Unspecified Blood Pressure	
	Label:	
	NLS_NOM_EMFC_BP	0x04010888
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
BPs	Unspecified Blood Pressure Systolic	
	Label:	
	NLS_NOM_EMFC_BP_SYS	0x04010889
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
BPd	Unspecified Blood Pressure Diastolic	
	Label:	
	NLS_NOM_EMFC_BP_DIA	0x0401088a
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
BPm	Unspecified Blood Pressure Mean	
	Label:	

	NLS_NOM_EMFC_BP_MEAN	0x0401088b
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
UrVSht	Urimeter - Urine Shift Volume.	
	Label:	
	NLS_NOM_EMFC_UrVSht	0x0401088c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrFl	Urimeter - Urine Flow.	
	Label:	
	NLS_NOM_EMFC_UrFl	0x04010890
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
???V@	The "V" stands for Volume	
	Label:	
	NLS_NOM_EMFC_XXXVYY	0x04010898
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
???R@	The "R" stands for Rate	
	Label:	
	NLS_NOM_EMFC_XXXRYY	0x0401089c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
iCa	ionized Calcium	
	Label:	
	NLS_NOM_EMFC_iCa	0x04010a2c
	Observed Value:	
	NOM_CONC_CA_GEN	0x7118
Hb	Calculated Hemoglobin	
	Label:	
	NLS_NOM_EMFC_HGB_CALC	0x04010a34
	Observed Value:	
	NOM_CONC_HB_ART	0x7014
pHc	pH value in the capillaries	
	Label:	
	NLS_NOM_EMFC_pHc	0x04010a44
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
&pH	Adjusted pH at &Patient Temperature	
	Label:	
	NLS_NOM_EMFC_pH_ADJ	0x04010a48
	Observed Value:	
	NOM_CONC_PH_GEN	0x7104
&pHa	Adjusted pH in the arterial Blood	
	Label:	
	NLS_NOM_EMFC_pHa_ADJ	0x04010a4c
	Observed Value:	
	NOM_CONC_PH_ART	0x7004
&pHv	Adjusted pH value in the venous Blood	
	Label:	
	NLS_NOM_EMFC_pHv_ADJ	0x04010a50
	Observed Value:	
	NOM_CONC_PH_VEN	0x7034
&pHc	Adjusted pH value in the capillaries	
	Label:	
	NLS_NOM_EMFC_pHc_ADJ	0x04010a54
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
PcO2	Partial O2 in the capillaries	
	Label:	
	NLS_NOM_EMFC_PcO2	0x04010a5c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
&PO2	Adjusted PO2 at Patient Temperature	
	Label:	

	NLS_NOM_EMFC_PO2_ADJ	0x04010a60
	Observed Value:	
	NOM_CONC_PO2_GEN	0x7174
&PaO2	Adjusted PaO2 at Patient Temperature on the arterial blood	
	Label:	
	NLS_NOM_EMFC_PaO2_ADJ	0x04010a64
	Observed Value:	
	NOM_CONC_PO2_ART	0x700C
&PvO2	Adjusted PvO2 at Patient Temperature	
	Label:	
	NLS_NOM_EMFC_PvO2_ADJ	0x04010a68
	Observed Value:	
	NOM_CONC_PO2_VEN	0x703C
&PcO2	Adjusted PcO2 at Patient Temperature	
	Label:	
	NLS_NOM_EMFC_PcO2_ADJ	0x04010a6c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
PcCO2	Partial CO2 in the capillaries	
	Label:	
	NLS_NOM_EMFC_PcCO2	0x04010a78
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
&PCO2	Computed PCO2 at Patient Temperature	
	Label:	
	NLS_NOM_EMFC_PCO2_ADJ	0x04010a7c
	Observed Value:	
	NOM_CONC_PCO2_GEN	0x7140
&PaCO2	Computed PaCO2 at Patient Temperature on the arterial blood	
	Label:	
	NLS_NOM_EMFC_PaCO2_ADJ	0x04010a80
	Observed Value:	
	NOM_CONC_PCO2_ART	0x7008
&PvCO2	Computed PvCO2 at Patient Temperature	
	Label:	
	NLS_NOM_EMFC_PvCO2_ADJ	0x04010a84
	Observed Value:	
	NOM_CONC_PCO2_VEN	0x7038
&PcCO2	Computed PcO2 at Patient Temperature	
	Label:	
	NLS_NOM_EMFC_PcCO2_ADJ	0x04010a88
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
tCO2	Calculated total CO2	
	Label:	
	NLS_NOM_EMFC_tCO2_CALC	0x04010a8c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SO2	Calculated SO2	
	Label:	
	NLS_NOM_EMFC_SO2_CALC	0x04010a90
	Observed Value:	
	NOM_SAT_O2_ART	0x4B34
SaO2	Calculated SaO2	
	Label:	
	NLS_NOM_EMFC_SaO2_CALC	0x04010a94
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SvO2	Calculated SvO2	
	Label:	
	NLS_NOM_EMFC_SvO2_CALC	0x04010a98
	Observed Value:	
	NOM_SAT_O2_VEN	0x4B3C
ScO2	Calculated ScO2	
	Label:	

	NLS_NOM_EMFC_ScO2_CALC	0x04010a9c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
HCO3	Calculated HCO3	
	Label:	
	NLS_NOM_EMFC_HCO3_CALC	0x04010aa0
	Observed Value:	
	NOM_CONC_HCO3_GEN	0x7108
BEecf	Calculated Base Excess	
	Label:	
	NLS_NOM_EMFC_BEecf_CALC	0x04010aa4
	Observed Value:	
	NOM_CONC_BASE_EXCESS_ECF	0xF090
AnGap	Calculated AnionGap	
	Label:	
	NLS_NOM_EMFC_AnGap_CALC	0x04010aa8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Urea	Urea used by the i-Stat	
	Label:	
	NLS_NOM_EMFC_Urea	0x04010ab8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
FIO2*	Manually entered FIO2 in the Blood Review window	
	Label:	
	NLS_NOM_EMFC_FIO2_MANUAL	0x04010abc
	Observed Value:	
	NOM_VENT_CONC_AWAY_O2_INSP	0x7498
BE,B	Calculated Base Excess in Blood	
	Label:	
	NLS_NOM_EMFC_BE_B_CALC	0x04010ac0
	Observed Value:	
	NOM_BASE_EXCESS_BLD_ART	0x716C
iMg	ionized Magnesium	
	Label:	
	NLS_NOM_EMFC_iMg	0x04010ac4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
O2*	Manually entered value. Can be either FIO2 or Flow.	
	Label:	
	NLS_NOM_EMFC_O2_MANUAL	0x04010ad4
	Observed Value:	
	NOM_CONC_AWAY_O2	0x5164
SetTmp	Setup Temperature. It's the selected/adjusted Temp in i-Stat Window	
	Label:	
	NLS_NOM_EMFC_SetTmp	0x04010ad8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Crea	Creatinine - Measured Value by the i-Stat Module	
	Label:	
	NLS_NOM_EMFC_Crea	0x04010adc
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
B/Cre	Ratio BUN/Creatinine. Calculated value by the i-Stat module	
	Label:	
	NLS_NOM_EMFC_B_PER_Cre_CALC	0x04010ae0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
U/Cre	Ratio Urea/Creatinine. Calculated value by the i-Stat module	
	Label:	
	NLS_NOM_EMFC_U_PER_Cre_CALC	0x04010ae4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Lact	Lactate. SMeasured value by the i-Stat module	
	Label:	

	NLS_NOM_EMFC_Lact	0x04010ae8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Sim	Simulator Result. Device data with the values:"Passed" and "Failed"	
	Label:	
	NLS_NOM_EMFC_Sim	0x04010af0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Press1	Pressure value of sensor 1. Device specific data	
	Label:	
	NLS_NOM_EMFC_Press1	0x04010b10
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Press2	Pressure value of sensor 2. Device specific data	
	Label:	
	NLS_NOM_EMFC_Press2	0x04010b14
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
ModTmp	Module Temperature. Device specific data	
	Label:	
	NLS_NOM_EMFC_ModTmp	0x04010b28
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
PatT	Patient Temperature	
	Label:	
	NLS_NOM_EMFC_Pat_T	0x04010b54
	Observed Value:	
	NOM_TEMP	0x4B48
AirT	Air temperature.	
	Label:	
	NLS_NOM_EMFC_Air_T	0x04010b58
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Power	Power requ'd to set the Air&Pat Temp in the incubator	
	Label:	
	NLS_NOM_EMFC_Power	0x04010b5c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
DET	Desired Environmental Temperature	
	Label:	
	NLS_NOM_EMFC_DET	0x04010b60
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
BagWgt	Weight of the Urine Disposable Bag	
	Label:	
	NLS_NOM_EMFC_BagWgt	0x04010bb8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
tUrVol	Total Urine Volume of the current measurement period	
	Label:	
	NLS_NOM_EMFC_tUrVol	0x04010bbc
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
UrDens	Density of the Urine fluid	
	Label:	
	NLS_NOM_EMFC_UrDens	0x04010bc0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Turine	Temperature of the Urine fluid	
	Label:	
	NLS_NOM_EMFC_Turine	0x04010bc4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Age	actual patient age. measured in years	
	Label:	

	NLS_NOM_EMFC_Age	0x04010bc8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Urine	Daily Urine output	
	Label:	
	NLS_NOM_EMFC_Urine	0x04010bd8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
PaFIO2	PaO2 to FIO2 ratio. Expressed in mmHg to % ratio	
	Label:	
	NLS_NOM_EMFC_PaFIO2	0x04010be0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SAPS	Simplified Acuity Physiologic Score	
	Label:	
	NLS_NOM_EMFC_SAPS	0x04010be8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
VNrmal	Missing parameters in SAPS parameter set which are considered in normal range	
	Label:	
	NLS_NOM_EMFC_VNrmal	0x04010bec
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
ALT	Means aspartate aminotransferase	
	Label:	
	NLS_NOM_EMFC_ALT	0x04010bf0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
SpRR	Spontaneous Respiration Rate	
	Label:	
	NLS_NOM_EMFC_SpRR	0x04010bf4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
???P@	The "P" stands for Pressure.	
	Label:	
	NLS_NOM_EMFC_XXXPYY	0x04010c80
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
inAGTs	Inspired secondary Anesthetic Agent	
	Label:	
	NLS_NOM_EMFC_inAGTs	0x04010cec
	Observed Value:	
	NOM_CONC_AWAY_AGENT_INSP	0x5390
etAGTs	EndTidal secondary Anesthetic Agent	
	Label:	
	NLS_NOM_EMFC_etAGTs	0x04010cf0
	Observed Value:	
	NOM_CONC_AWAY_AGENT_ET	0x538C
BagVol	Current fluid (Urine) in the Urine Bag	
	Label:	
	NLS_NOM_EMFC_BagVol	0x04010cfc
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
TOFcnt	Train Of Four (TOF) count - Number of TOF responses.	
	Label:	
	NLS_NOM_EMFC_TOFcnt	0x04010dac
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
TOFrat	Train Of Four (TOF) ratio	
	Label:	
	NLS_NOM_EMFC_TOFrat	0x04010db0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
Twitch	Twitch height of the 1Hz/0.1Hz stimulation response	

	Label:	
	NLS_NOM_EMFC_Twitch	0x04010db4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
PTC	Post Tetatic Count stimulation	
	Label:	
	NLS_NOM_EMFC_PTC	0x04010db8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
RemTi	Remaining Time until next stimulation	
	Label:	
	NLS_NOM_EMFC_RemTi	0x04010dbc
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
TOF1	TrainOf Four (TOF) first response value TOF1	
	Label:	
	NLS_NOM_EMFC_TOF1	0x04010dc0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
TOF2	TrainOf Four (TOF) first response value TOF2	
	Label:	
	NLS_NOM_EMFC_TOF2	0x04010dc4
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
TOF3	TrainOf Four (TOF) first response value TOF3	
	Label:	
	NLS_NOM_EMFC_TOF3	0x04010dc8
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
TOF4	TrainOf Four (TOF) first response value TOF4	
	Label:	
	NLS_NOM_EMFC_TOF4	0x04010dcc
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
ACT	Activated Clotting Time. Measured value by the i-Stat module	
	Label:	
	NLS_NOM_EMFC_ACT	0x04010e10
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
InsTi	Spontaneous Inspiration Time	
	Label:	
	NLS_NOM_EMFC_InsTi	0x04010e74
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
C20/C	Overdistension Index	
	Label:	
	NLS_NOM_EMFC_C20_PER_C	0x04010e78
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
TC	Time Constant	
	Label:	
	NLS_NOM_EMFC_TC	0x04010e7c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
r	Correlation Coefficient	
	Label:	
	NLS_NOM_EMFC_r	0x04010e80
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
RVrat	Rate Volume Ratio	
	Label:	
	NLS_NOM_EMFC_RVrat	0x04010e84
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
iCa(N)	ionized Calcium Normalized	

	Label:	
	NLS_NOM_EMFC_iCa_N	0x04010e88
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
TVPSV	Tidal Volume (TV) in "Pressure Support Ventilation" mode	
	Label:	
	NLS_NOM_EMFC_TVPSV	0x04010e98
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
NIF	Negative Inspiratory Force Index	
	Label:	
	NLS_NOM_EMFC_NIF	0x04010e9c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
RSBI	Rapid Shallow Breathing Index	
	Label:	
	NLS_NOM_EMFC_RSBI	0x04010ea0
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
iCa-N	Ionized Calcium Normalized	
	Label:	
	NLS_NOM_EMFC_iCa_N_CALC	0x04011114
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
sVMode	Setting: Ventilation Mode, e.g.AUTO for the Sechrist Ventilator	
	Label:	
	NLS_NOM_EMFC_sVMode	0x04018000
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
sAWRR	Setting: Airway Respiratory Rate	
	Label:	
	NLS_NOM_EMFC_sAWRR	0x04018004
	Observed Value:	
	NOM_AWAY_RESP_RATE	0x5012
sTV	Setting: Tidal Volume	
	Label:	
	NLS_NOM_EMFC_sTV	0x04018008
	Observed Value:	
	NOM_VOL_AWAY_TIDAL	0x513C
sPkFl	Setting: Peak Inspiratory Flow	
	Label:	
	NLS_NOM_EMFC_sPkFl	0x0401800c
	Observed Value:	
	NOM_PRESS_AWAY_INSP_MAX	0x5109
sFIO2	Setting: Inspired Oxygen Concentration	
	Label:	
	NLS_NOM_EMFC_sFIO2	0x04018010
	Observed Value:	
	NOM_VENT_CONC_AWAY_O2_INSP	0x7498
sTrig	Setting: Trigger Sensitivity	
	Label:	
	NLS_NOM_EMFC_sTrig	0x04018014
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
sPltTi	Setting: Plateau Time	
	Label:	
	NLS_NOM_EMFC_sPltTi	0x04018018
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
sSghR	Setting: Sigh Rate	
	Label:	
	NLS_NOM_EMFC_sSghR	0x0401801c
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
sSghTV	Setting: Sigh Tidal Volume	

```
                Label:
                NLS_NOM_EMFC_sSghTV                         0x04018020
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sSghNr          Setting: Multiple Sigh Number
                Label:
                NLS_NOM_EMFC_sSghNr                         0x04018024
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sATV            Setting: Apnea Tidal Volume
                Label:
                NLS_NOM_EMFC_sATV                           0x04018028
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sARR            Setting: Apnea Respiration Rate
                Label:
                NLS_NOM_EMFC_sARR                           0x0401802c
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sAPkFl          Setting: Apnea Peak Flow
                Label:
                NLS_NOM_EMFC_sAPkFl                         0x04018030
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sAFIO2          Setting: Apnea Inspired O2 Concentration
                Label:
                NLS_NOM_EMFC_sAFIO2                         0x04018034
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sPSV            Setting: Pressure Support Ventilation
                Label:
                NLS_NOM_EMFC_sPSV                           0x04018038
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sPWave          Setting: Pressure Waveform
                Label:
                NLS_NOM_EMFC_sPWave                         0x0401803c
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sEnSgh          Setting: Enable Sigh
                Label:
                NLS_NOM_EMFC_sEnSgh                         0x04018040
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sNeblr          Setting: Nebulizer
                Label:
                NLS_NOM_EMFC_sNeblr                         0x04018044
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sO2Suc          Setting: Suction Oxygen Concentration
                Label:
                NLS_NOM_EMFC_sO2Suc                         0x04018048
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sDtaIv          Setting: Data Averaging Interval
                Label:
                NLS_NOM_EMFC_sDtaIv                         0x04018050
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sChrIv          Setting: Chart Interval
                Label:
                NLS_NOM_EMFC_sChrIv                         0x04018054
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sBasFl          Setting: Flow-by Base Flow
```

```
                Label:
                NLS_NOM_EMFC_sBasFl                         0x04018058
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sSenFl          Setting: Flow-by Sensitivity Flow
                Label:
                NLS_NOM_EMFC_sSenFl                         0x0401805c
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sOxiIv          Setting: Oximeter Averaging Interval
                Label:
                NLS_NOM_EMFC_sOxiIv                         0x04018060
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sPVcP           Setting: Pressure Ventilation Control Pressure
                Label:
                NLS_NOM_EMFC_sPVcP                          0x04018064
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sPVinT          Setting: Pressure Ventilation Inspiratory Time
                Label:
                NLS_NOM_EMFC_sPVinT                         0x04018068
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sAPVcP          Setting: Apnea Pressure Ventilation Control Pressure
                Label:
                NLS_NOM_EMFC_sAPVcP                         0x0401806c
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sAPVRR          Setting: Apnea Pressure Ventilation Respiration Rate
                Label:
                NLS_NOM_EMFC_sAPVRR                         0x04018070
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sAPVTi          Setting: Apnea Pressure Ventilation Inspiratory Time
                Label:
                NLS_NOM_EMFC_sAPVTi                         0x04018074
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sAPVO2          Setting: Apnea Pressure Ventilation Oxygen Concentration
                Label:
                NLS_NOM_EMFC_sAPVO2                         0x04018078
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sAPVhP          Setting: Apnea Pressure Ventilation High Airway Pressure
                Label:
                NLS_NOM_EMFC_sAPVhP                         0x0401807c
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sSilnc          Setting: Alarm Silence Status
                Label:
                NLS_NOM_EMFC_sSilnc                         0x04018080
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sPVI            Setting: Pressure Ventilation I component of I:E Ratio
                Label:
                NLS_NOM_EMFC_sPVI                           0x04018084
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sPVE            Setting: Pressure Ventilation E component of I:E Ratio
                Label:
                NLS_NOM_EMFC_sPVE                           0x04018088
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sAPVI           Setting: Apnea Pressure Ventilation I component of I:E Ratio
```

```
                Label:
                NLS_NOM_EMFC_sAPVI                              0x0401808c
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sAPVE           Setting: Apnea Pressure Ventilation E component of I:E Ratio
                Label:
                NLS_NOM_EMFC_sAPVE                              0x04018090
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sPVCtl          Setting: Pressure Ventilation Control
                Label:
                NLS_NOM_EMFC_sPVCtl                             0x04018094
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sMode           Setting: Mode. Example include the ART/VEN from Oximterix Abbott.
                Label:
                NLS_NOM_EMFC_sMode                              0x04018098
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sCycTi          Setting: Cycle Time
                Label:
                NLS_NOM_EMFC_sCycTi                             0x0401809c
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sIPPV           Setting: Ventilation Frequency in IPPV Mode
                Label:
                NLS_NOM_EMFC_sIPPV                              0x040180a0
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sIMV            Setting: Ventilation Frequency in IMV Mode
                Label:
                NLS_NOM_EMFC_sIMV                               0x040180a4
                Observed Value:
                NOM_VENT_MODE_MAND_INTERMIT                     0xD02A
sPEEP           Setting: PEEP/CPAP
                Label:
                NLS_NOM_EMFC_sPEEP                              0x040180a8
                Observed Value:
                NOM_VENT_PRESS_AWAY_END_EXP_POS                 0x51A8
sSPEEP          Setting: Pressure Support PEEP
                Label:
                NLS_NOM_EMFC_sSPEEP                             0x040180ac
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sMV             Setting: Minute Volume
                Label:
                NLS_NOM_EMFC_sMV                                0x040180b0
                Observed Value:
                NOM_VOL_MINUTE_AWAY                             0x5148
sEnTrg          Setting: Enable Trigger
                Label:
                NLS_NOM_EMFC_sEnTrg                             0x040180b4
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sEnTP           Setting: Tachyapnea warning
                Label:
                NLS_NOM_EMFC_sEnTP                              0x040180b8
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
BPAPPL          Setting: BIPAP Pressure Level 1 - Low
                Label:
                NLS_NOM_EMFC_BPAPPL                             0x040180bc
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
BPAPPH          Setting: BIPAP Pressure Level 2  - High
```

```
                Label:
                NLS_NOM_EMFC_BPAPPH                          0x040180c0
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
BPAPTL          Setting: BIPAP Pressure Time 1 - Low Time
                Label:
                NLS_NOM_EMFC_BPAPTL                          0x040180c4
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
BPAPTH          Setting: BIPAP Pressure Time 2  - High Time
                Label:
                NLS_NOM_EMFC_BPAPTH                          0x040180c8
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
sAVDel          Setting: Apnea Ventilation Delay
                Label:
                NLS_NOM_EMFC_sAVDel                          0x040180cc
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
sTPDel          Setting: Tachyapnea warning time alarm
                Label:
                NLS_NOM_EMFC_sTPDel                          0x040180d0
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
sO2Mon          Setting: O2 Monitoring
                Label:
                NLS_NOM_EMFC_sO2Mon                          0x040180d4
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
sO2Cal          Setting: O2 Calibration
                Label:
                NLS_NOM_EMFC_sO2Cal                          0x040180d8
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
sMVAl           Setting: Minute Volume Alarms On/Off state
                Label:
                NLS_NOM_EMFC_sMVAl                           0x040180dc
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
sPmax           Setting: Maximum Pressure
                Label:
                NLS_NOM_EMFC_sPmax                           0x040180e0
                Observed Value:
                NOM_PRESS_AWAY_INSP_MAX                      0x5109
sInsTi          Setting: Inspiratory Time
                Label:
                NLS_NOM_EMFC_sInsTi                          0x040180e4
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
sExpTi          Setting: Exhaled Time
                Label:
                NLS_NOM_EMFC_sExpTi                          0x040180e8
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
sIE1:           Setting: Inspiration to Expiration Ratio.
                Label:
                NLS_NOM_EMFC_sIE_1                           0x040180ec
                Observed Value:
                NOM_RATIO_IE                                 0x5118
sALMRT          Setting: Alarm Percentage on Rise Time.
                Label:
                NLS_NOM_EMFC_sALMRT                          0x040180f0
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
sCPAP           Setting: Continuous Positive Airway Pressure Value
```

```
                Label:
                NLS_NOM_EMFC_sCPAP                              0x040180f4
                Observed Value:
                NOM_PRESS_AWAY_CTS_POS                          0x50F4
sFlow           Setting: Flow
                Label:
                NLS_NOM_EMFC_sFlow                              0x040180f8
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sPIP            Setting: Positive Inspiratory Pressure
                Label:
                NLS_NOM_EMFC_sPIP                               0x040180fc
                Observed Value:
                NOM_PRESS_AWAY_INSP_MAX                         0x5109
sLInPr          Setting: Low Inspiratory Pressure
                Label:
                NLS_NOM_EMFC_sLInPr                             0x04018100
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sHFVFl          Setting: High Freqyency Ventilation Flow
                Label:
                NLS_NOM_EMFC_sHFVFl                             0x04018104
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sHFVRR          Setting: High Frequency Ventilation Respiration Rate
                Label:
                NLS_NOM_EMFC_sHFVRR                             0x04018108
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
Model           Setting: Oxygen Concentration in %
                Label:
                NLS_NOM_EMFC_Model                              0x04018110
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sO2             Enumeration Type - denotes type of Instrument.
                Label:
                NLS_NOM_EMFC_sO2                                0x0401810c
                Observed Value:
                NOM_CONC_AWAY_O2                                0x5164
sCMV            Setting: Controlled mechanical ventilation
                Label:
                NLS_NOM_EMFC_sCMV                               0x04018114
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sSIMV           Setting: Synchronized intermittent mandatory ventilation
                Label:
                NLS_NOM_EMFC_sSIMV                              0x04018118
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sMMV            Setting: Mandatory Minute Volume
                Label:
                NLS_NOM_EMFC_sMMV                               0x0401811c
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sFWave          Setting: Flow Pattern (Enumeration type )
                Label:
                NLS_NOM_EMFC_sFWave                             0x04018120
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sDRate          Setting: Infusion Pump Delivery Rate
                Label:
                NLS_NOM_EMFC_sDRate                             0x04018124
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
sPin            Setting: Inspiratory Pressure. Used by the Ohmeda Ventilator.
```

```
                Label:
                NLS_NOM_EMFC_sPin                           0x04018128
                Observed Value:
                NOM_PRESS_AWAY_INSP_MAX                     0x5109
sRRaw           Setting: Airway Respiration Rate. Used by the Ohmeda Ventilator.
                Label:
                NLS_NOM_EMFC_sRRaw                          0x0401812c
                Observed Value:
                NOM_AWAY_RESP_RATE                          0x5012
sInsFl          Setting: Inspiratory Flow.
                Label:
                NLS_NOM_EMFC_sInsFl                         0x04018130
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sExpFl          Setting: Expiratory Flow
                Label:
                NLS_NOM_EMFC_sExpFl                         0x04018134
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sTrVol          Setting: Trigger Flow/Volume
                Label:
                NLS_NOM_EMFC_sTrVol                         0x04018138
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sAADel          Setting: Apnea Alarm Delay
                Label:
                NLS_NOM_EMFC_sAADel                         0x0401813c
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sHFVAm          Setting: HFV Amplitude (Peak to Peak Pressure)
                Label:
                NLS_NOM_EMFC_sHFVAm                         0x04018140
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sMVDel          Setting: Minute Volume Alarm Delay
                Label:
                NLS_NOM_EMFC_sMVDel                         0x04018144
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sTrgFl          Setting: Flow Trigger - delivered by the Evita 2 Vuelink Driver
                Label:
                NLS_NOM_EMFC_sTrgFl                         0x04018148
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sPincR          Setting: Pressure Increase Rate
                Label:
                NLS_NOM_EMFC_sPincR                         0x0401814c
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sVmax           Setting: Volume Warning - delivered by the Evita 2 Vuelink Driver
                Label:
                NLS_NOM_EMFC_sVmax                          0x04018150
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sFlCal          Enumeration Setting: Flow Calibration
                Label:
                NLS_NOM_EMFC_sFlCal                         0x04018154
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sVolAl          Enumeration Setting: Volume Alarm
                Label:
                NLS_NOM_EMFC_sVolAl                         0x04018158
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sCO2Wm          Enumeration Setting: CO2 Monitoring in Low Accuracy (WARM UP) mode
```

```
                Label:
                NLS_NOM_EMFC_sCO2Wm                         0x0401815c
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sCO2Al          Enumeration Setting: CO2 Alarm
                Label:
                NLS_NOM_EMFC_sCO2Al                         0x04018160
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sPtCat          Enumeration Setting: Patient Category (Patient Type)
                Label:
                NLS_NOM_EMFC_sPtCat                         0x04018164
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sOxiAl          Enumeration Setting: Oximeter Alarm
                Label:
                NLS_NOM_EMFC_sOxiAl                         0x04018168
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sAGTWm          Enumeration Setting: Agent Monitoring in Low Accuracy (WARM UP) mode
                Label:
                NLS_NOM_EMFC_sAGTWm                         0x0401816c
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
Wave            Enumeration: Used to label the wave delay (Mortara ELI 100/STM).
                Label:
                NLS_NOM_EMFC_Wave                           0x04018170
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
loPmax          Setting: Low Maximum Airway Pressure Alarm Setting.
                Label:
                NLS_NOM_EMFC_loPmax                         0x04018174
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sAgent          Enumeration Setting: Shows current selected Agent
                Label:
                NLS_NOM_EMFC_sAgent                         0x04018178
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sADel           Setting used for the Apnea Alarm Delay (Apnea Time).
                Label:
                NLS_NOM_EMFC_sADel                          0x0401817c
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sPSVrp          Setting: PSV Ramp (ASB Ramp). PSV = Pressure Support Ventilation
                Label:
                NLS_NOM_EMFC_sPSVrp                         0x04018180
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sTVap           Setting: Applied Tidal Volume.
                Label:
                NLS_NOM_EMFC_sTVap                          0x04018184
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sSens           Setting: Assist Sensitivity. Used by the Bear 1000 ventilator.
                Label:
                NLS_NOM_EMFC_sSens                          0x04018188
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sHInPr          Setting: High Inspiratory Pressure Setting.
                Label:
                NLS_NOM_EMFC_sHInPr                         0x0401818c
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sBkgFl          Setting: Background Flow Setting. Range is 2 - 30 l/min
```

```
            Label:
            NLS_NOM_EMFC_sBkgFl                          0x04018190
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
sfgAGT      Setting: Vaporizer concentration.
            Label:
            NLS_NOM_EMFC_sfgAGT                          0x04018198
            Observed Value:
            NOM_CONC_AWAY_AGENT                          0x5388
sfgISO      Setting: Vaporizer concentration for ISOflurane
            Label:
            NLS_NOM_EMFC_sfgISO                          0x0401819c
            Observed Value:
            NOM_CONC_AWAY_ISOFL                          0x51E8
sfgENF      Setting: Vaporizer concentration for ENFlurane
            Label:
            NLS_NOM_EMFC_sfgENF                          0x040181a0
            Observed Value:
            NOM_CONC_AWAY_ENFL                           0x51DC
sfgHAL      Setting: Vaporizer concentration for HALothane
            Label:
            NLS_NOM_EMFC_sfgHAL                          0x040181a4
            Observed Value:
            NOM_CONC_AWAY_HALOTH                         0x51E0
sfgDES      Setting: Vaporizer concentration for DESflurane
            Label:
            NLS_NOM_EMFC_sfgDES                          0x040181a8
            Observed Value:
            NOM_CONC_AWAY_DESFL                          0x51D8
sfgSEV      Setting: Vaporizer concentration for SEVoflurane
            Label:
            NLS_NOM_EMFC_sfgSEV                          0x040181ac
            Observed Value:
            NOM_CONC_AWAY_SEVOFL                         0x51E4
sfgAir      Setting: Total fresh gas Air flow on the mixer
            Label:
            NLS_NOM_EMFC_sfgAir                          0x040181b0
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
sfgO2       Setting: Fresh gas oxygen Flow on the mixer
            Label:
            NLS_NOM_EMFC_sfgO2                           0x040181b4
            Observed Value:
            NOM_CONC_AWAY_O2                             0x5164
sfgFl       Setting: Total fresh gas Flow on the mixer
            Label:
            NLS_NOM_EMFC_sfgFl                           0x040181b8
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
sfgN2O      Setting: fresh gas N2O flow on the mixer
            Label:
            NLS_NOM_EMFC_sfgN2O                          0x040181bc
            Observed Value:
            NOM_CONC_AWAY_N2O                            0x51F0
sGasPr      Setting: Gas Sample point for the oxygen measurement
            Label:
            NLS_NOM_EMFC_sGasPr                          0x040181c0
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
sO2Pr       Setting: Gas sample point for oxygen measurement
            Label:
            NLS_NOM_EMFC_sO2Pr                           0x040181c4
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
sCircl      Setting: Type of circle in the patient system
```

```
                Label:
                NLS_NOM_EMFC_sCircl                         0x040181c8
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sTVin           Setting: inspired Tidal Volume
                Label:
                NLS_NOM_EMFC_sTVin                          0x040181cc
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
setT            Setting: Desired Temperature for the Incubator/Warmer
                Label:
                NLS_NOM_EMFC_set_T                          0x040181d0
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sUrTi           Setting: Preset period of time for the UrVol numeric
                Label:
                NLS_NOM_EMFC_sUrTi                          0x040181d4
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sBrSys          Settting: Breathing System. Used by the Ohmeda Anesthesia Machine
                Label:
                NLS_NOM_EMFC_sBrSys                         0x040181d8
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
GasCar          Enumeration: Selected Gas Carrier. Used by the Julian Draeger I/F
                Label:
                NLS_NOM_EMFC_GasCar                         0x040181dc
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
CO2Cal          Enumeration or CO2 Calibration active
                Label:
                NLS_NOM_EMFC_CO2Cal                         0x040181e0
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sTlow           Setting: part of the Evita 4 Airway Pressure Release Ventilation Mode
                Label:
                NLS_NOM_EMFC_sTlow                          0x040181e4
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sThigh          Setting: part of the Evita 4 Airway Pressure Release Ventilation Mode
                Label:
                NLS_NOM_EMFC_sThigh                         0x040181e8
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sPlow           Setting: part of the Evita 4 Airway Pressure Release Ventilation Mode
                Label:
                NLS_NOM_EMFC_sPlow                          0x040181ec
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sPhigh          Setting: part of the Evita 4 Airway Pressure Release Ventilation Mode
                Label:
                NLS_NOM_EMFC_sPhigh                         0x040181f0
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sVolas          Setting: Volume Assist level for the CPAP mode
                Label:
                NLS_NOM_EMFC_sVolas                         0x040181f4
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sFlas           Setting: Flow Assist level for the CPAP mode
                Label:
                NLS_NOM_EMFC_sFlas                          0x040181f8
                Observed Value:
                NOM_METRIC_NOS                              0xEFFF
sCurnt          Setting: Preset stimulation current
```

```
            Label:
            NLS_NOM_EMFC_sCurnt                          0x040181fc
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
sChrge      Setting: Preset stimulation charge
            Label:
            NLS_NOM_EMFC_sChrge                          0x04018200
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
sPulsD      Setting: Preset stimulation impulse duration
            Label:
            NLS_NOM_EMFC_sPulsD                          0x04018204
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
sRepTi      Setting: Preset Train Of Four (Slow TOF) repetition time
            Label:
            NLS_NOM_EMFC_sRepTi                          0x04018208
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
sfmax       Setting: Panting Limit
            Label:
            NLS_NOM_EMFC_sfmax                           0x0401820c
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
BO_ABG      Setting: Measured Parameter reported at Blood Temp adjusted to ABG
            Label:
            NLS_NOM_EMFC_BO_ABG                          0x04018274
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
BO_PAR      Setting: Parameter reported at Blood Temp adjusted to TrendCare Cal
            Label:
            NLS_NOM_EMFC_BO_PAR                          0x04018278
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
37_ABG      Setting: Measured Parameter reported at 37 Grad C  adjusted to ABG
            Label:
            NLS_NOM_EMFC_37_ABG                          0x0401827c
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
37_PAR      Setting: Parameter reported at 37 Grad C  adjusted to TrendCare Cal
            Label:
            NLS_NOM_EMFC_37_PAR                          0x04018280
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
highP       Alarm Limit: High Pressure
            Label:
            NLS_NOM_EMFC_highP                           0x0401a000
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
loPEEP      Alarm Limit: Low PEEP/CPAP
            Label:
            NLS_NOM_EMFC_loPEEP                          0x0401a004
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
hiSghP      Alarm Limit: Sigh High Pressure
            Label:
            NLS_NOM_EMFC_hiSghP                          0x0401a008
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
liATi       Alarm Limit: Apnea Time Interval
            Label:
            NLS_NOM_EMFC_liATi                           0x0401a00c
            Observed Value:
            NOM_METRIC_NOS                               0xEFFF
liPVAT      Alarm Limit: Pressure Ventilation Apnea Time Interval
```

```
                Label:
                NLS_NOM_EMFC_liPVAT                          0x0401a010
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
sustP           Alarm Limit: Sustained Pressure Alarm Limit.
                Label:
                NLS_NOM_EMFC_sustP                           0x0401a014
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
lowMV           Alarm Limit: Low Minute Volume Alarm Limit
                Label:
                NLS_NOM_EMFC_lowMV                           0x0401a018
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
lowO2           Alarm Limit: Low Oxygen (O2) Alarm Limit
                Label:
                NLS_NOM_EMFC_lowO2                           0x0401a01c
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
highO2          Alarm Limit. High Oxygen (O2) Alarm Limit
                Label:
                NLS_NOM_EMFC_highO2                          0x0401a020
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
sBPAl           Blood Pressure Alarm state (ON/OFF) - Enumeration type
                Label:
                NLS_NOM_EMFC_sBPAl                           0x0401a024
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
sPStat          General status of Infusion Pump - Enumeration type
                Label:
                NLS_NOM_EMFC_sPStat                          0x0401a028
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
highMV          Alarm Limit: High Minute Volume Alarm Limit
                Label:
                NLS_NOM_EMFC_highMV                          0x0401a02c
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
lowTV           Alarm Limit: Low Tidal Volume Alarm Limit
                Label:
                NLS_NOM_EMFC_lowTV                           0x0401a030
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
highTV          Alarm Limit: High Tidal Volume Alarm Limit
                Label:
                NLS_NOM_EMFC_highTV                          0x0401a034
                Observed Value:
                NOM_METRIC_NOS                               0xEFFF
Num1            Placeholder for a Vuelink Flex Text: Numeric Label
                Label:
                NLS_VUELINK_FLX1_NPS_TEXT_NUM1               0x80aaf064
                Observed Value:
                depends on configuration
Num2            Placeholder for a Vuelink Flex Text: Numeric Label
                Label:
                NLS_VUELINK_FLX1_NPS_TEXT_NUM2               0x80aaf066
                Observed Value:
                depends on configuration
Num3            Placeholder for a Vuelink Flex Text: Numeric Label
                Label:
                NLS_VUELINK_FLX1_NPS_TEXT_NUM3               0x80aaf068
                Observed Value:
                depends on configuration
Num4            Placeholder for a Vuelink Flex Text: Numeric Label
```

	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM4	0x80aaf06a
	Observed Value:	
	depends on configuration	
Num5	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM5	0x80aaf06c
	Observed Value:	
	depends on configuration	
Num6	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM6	0x80aaf06e
	Observed Value:	
	depends on configuration	
Num7	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM7	0x80aaf070
	Observed Value:	
	depends on configuration	
Num8	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM8	0x80aaf072
	Observed Value:	
	depends on configuration	
Num9	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM9	0x80aaf074
	Observed Value:	
	depends on configuration	
Num10	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM10	0x80aaf076
	Observed Value:	
	depends on configuration	
Num11	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM11	0x80aaf078
	Observed Value:	
	depends on configuration	
Num12	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM12	0x80aaf07a
	Observed Value:	
	depends on configuration	
Num13	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM13	0x80aaf07c
	Observed Value:	
	depends on configuration	
Num14	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM14	0x80aaf07e
	Observed Value:	
	depends on configuration	
Num15	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM15	0x80aaf080
	Observed Value:	
	depends on configuration	
Num16	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM16	0x80aaf082
	Observed Value:	
	depends on configuration	
Num17	Placeholder for a Vuelink Flex Text: Numeric Label	

	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM17	0x80aaf084
	Observed Value:	
	depends on configuration	
Num18	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM18	0x80aaf086
	Observed Value:	
	depends on configuration	
Num19	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM19	0x80aaf088
	Observed Value:	
	depends on configuration	
Num20	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM20	0x80aaf08a
	Observed Value:	
	depends on configuration	
Num21	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM21	0x80aaf08c
	Observed Value:	
	depends on configuration	
Num22	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM22	0x80aaf08e
	Observed Value:	
	depends on configuration	
Num23	Placeholder for a Vuelink Flex Text: Numeric Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_NUM23	0x80aaf090
	Observed Value:	
	depends on configuration	
Gases	Label for the Service Information output	
	Label:	
	NLS_GASES_NAMES_DIAG_INFO_LBL	0x805a5403
	Observed Value:	
	depends on configuration	
	Enumeration Pump label: Name of drug (blank)	
	Label:	
	NLS_NOM_EMFC_BLANK	0x04010960
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF

Waves

ECG	Unspecific ECG wave	
	Label:	
	NLS_NOM_ECG_ELEC_POTL	0x00020100
	Observed Value:	
	depends on configuration	
I	ECG lead I	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_I	0x00020101
	Observed Value:	
	NOM_ECG_ELEC_POTL_I	0x0101
II	ECG lead II	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_II	0x00020102
	Observed Value:	
	NOM_ECG_ELEC_POTL_II	0x0102
III	ECG lead III	

	Label:	
	NLS_NOM_ECG_ELEC_POTL_III	0x0002013d
	Observed Value:	
	NOM_ECG_ELEC_POTL_III	0x013D
AVR	ECG lead aVR	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_AVR	0x0002013e
	Observed Value:	
	NOM_ECG_ELEC_POTL_AVR	0x013E
AVL	ECG lead aVL	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_AVL	0x0002013f
	Observed Value:	
	NOM_ECG_ELEC_POTL_AVL	0x013F
AVF	ECG lead aVF	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_AVF	0x00020140
	Observed Value:	
	NOM_ECG_ELEC_POTL_AVF	0x0140
V	ECG lead V	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_V	0x00020143
	Observed Value:	
	NOM_ECG_ELEC_POTL_V	0x0143
MCL	ECG lead MCL	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_MCL	0x0002014b
	Observed Value:	
	NOM_ECG_ELEC_POTL_MCL	0x014B
V1	ECG lead V1	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_V1	0x00020103
	Observed Value:	
	NOM_ECG_ELEC_POTL_V1	0x0103
V2	ECG lead V2	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_V2	0x00020104
	Observed Value:	
	NOM_ECG_ELEC_POTL_V2	0x0104
V3	ECG lead V3	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_V3	0x00020105
	Observed Value:	
	NOM_ECG_ELEC_POTL_V3	0x0105
V4	ECG lead V4	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_V4	0x00020106
	Observed Value:	
	NOM_ECG_ELEC_POTL_V4	0x0106
V5	ECG lead V5	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_V5	0x00020107
	Observed Value:	
	NOM_ECG_ELEC_POTL_V5	0x0107
V6	ECG lead V6	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_V6	0x00020108
	Observed Value:	
	NOM_ECG_ELEC_POTL_V6	0x0108
MCL1	ECG lead MCL1	
	Label:	
	NLS_NOM_ECG_ELEC_POTL_MCL1	0x0002014c
	Observed Value:	
	NOM_ECG_ELEC_POTL_MCL1	0x014C
Pleth	PLETH wave label	

	Label:	
	NLS_NOM_PULS_OXIM_PLETH	0x00024bb4
	Observed Value:	
	NOM_PULS_OXIM_PLETH	0x4BB4
PLETHl	PLETH wave (left)	
	Label:	
	NLS_NOM_PULS_OXIM_PLETH_LEFT	0x0002f08d
	Observed Value:	
	NOM_PULS_OXIM_PLETH_LEFT	0xF08D
PLETHr	PLETH wave (right)	
	Label:	
	NLS_NOM_PULS_OXIM_PLETH_RIGHT	0x0002f08c
	Observed Value:	
	NOM_PULS_OXIM_PLETH_RIGHT	0xF08C
ABP	Arterial Blood Pressure (ABP)	
	Label:	
	NLS_NOM_PRESS_BLD_ART_ABP	0x00024a14
	Observed Value:	
	NOM_PRESS_BLD_ART_ABP	0x4A14
ART	Arterial Blood Pressure (ART)	
	Label:	
	NLS_NOM_PRESS_BLD_ART	0x00024a10
	Observed Value:	
	NOM_PRESS_BLD_ART	0x4A10
Ao	Arterial Blood Pressure in the Aorta (Ao)	
	Label:	
	NLS_NOM_PRESS_BLD_AORT	0x00024a0c
	Observed Value:	
	NOM_PRESS_BLD_AORT	0x4A0C
PAP	Pulmonary Arterial Pressure (PAP)	
	Label:	
	NLS_NOM_PRESS_BLD_ART_PULM	0x00024a1c
	Observed Value:	
	NOM_PRESS_BLD_ART_PULM	0x4A1C
CVP	Central Venous Pressure (CVP)	
	Label:	
	NLS_NOM_PRESS_BLD_VEN_CENT	0x00024a44
	Observed Value:	
	NOM_PRESS_BLD_VEN_CENT	0x4A44
RAP	Right Atrial Pressure (RAP)	
	Label:	
	NLS_NOM_PRESS_BLD_ATR_RIGHT	0x00024a34
	Observed Value:	
	NOM_PRESS_BLD_ATR_RIGHT	0x4A34
LAP	Left Atrial Pressure (LAP)	
	Label:	
	NLS_NOM_PRESS_BLD_ATR_LEFT	0x00024a30
	Observed Value:	
	NOM_PRESS_BLD_ATR_LEFT	0x4A30
ICP	Intra-cranial Pressure (ICP)	
	Label:	
	NLS_NOM_PRESS_INTRA_CRAN	0x00025808
	Observed Value:	
	NOM_PRESS_INTRA_CRAN	0x5808
UAP	Umbilical Arterial Pressure (UAP)	
	Label:	
	NLS_NOM_PRESS_BLD_ART_UMB	0x00024a28
	Observed Value:	
	NOM_PRESS_BLD_ART_UMB	0x4A28
UVP	Umbilical Venous Pressure (UVP)	
	Label:	
	NLS_NOM_PRESS_BLD_VEN_UMB	0x00024a48
	Observed Value:	
	NOM_PRESS_BLD_VEN_UMB	0x4A48
P	unspecific pressure	

	Label:	
	NLS_NOM_PRESS_BLD	0x00024a00
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
CO2	CO2 concentration	
	Label:	
	NLS_NOM_AWAY_CO2	0x000250ac
	Observed Value:	
	NOM_AWAY_CO2	0x50AC
O2	Generic oxigen measurement label	
	Label:	
	NLS_NOM_CONC_AWAY_O2	0x00025164
	Observed Value:	
	NOM_CONC_AWAY_O2	0x5164
Resp	Impedance RESP wave	
	Label:	
	NLS_NOM_RESP	0x00025000
	Observed Value:	
	NOM_RESP	0x5000
AWF	Airway Flow Wave	
	Label:	
	NLS_NOM_FLOW_AWAY	0x000250d4
	Observed Value:	
	NOM_FLOW_AWAY	0x50D4
AWP	Airway Pressure Wave	
	Label:	
	NLS_NOM_PRESS_AWAY	0x000250f0
	Observed Value:	
	NOM_PRESS_AWAY	0x50F0
AWPin	Airway Pressure Wave - measured in the inspiratory path	
	Label:	
	NLS_NOM_PRESS_AWAY_INSP	0x00025108
	Observed Value:	
	NOM_PRESS_AWAY_INSP	0x5108
AWFin	Airway Flow Wave - measured in the inspiratory path	
	Label:	
	NLS_NOM_VENT_FLOW_INSP	0x0002518c
	Observed Value:	
	NOM_VENT_FLOW_INSP	0x518C
EEG	generic EEG and BIS label	
	Label:	
	NLS_NOM_EEG_ELEC_POTL_CRTX	0x0002592c
	Observed Value:	
	NOM_EEG_ELEC_POTL_CRTX	0x592C
EEG1	EEG wave channel 1	
	Label:	
	NLS_EEG_NAMES_EEG_CHAN1_LBL	0x800f5401
	Observed Value:	
	NOM_EEG_ELEC_POTL_CRTX	0x592C
EEG2	EEG wave channel 2	
	Label:	
	NLS_EEG_NAMES_EEG_CHAN2_LBL	0x800f5402
	Observed Value:	
	NOM_EEG_ELEC_POTL_CRTX	0x592C
Tblood	Tblood	
	Label:	
	NLS_NOM_TEMP_BLD	0x0002e014
	Observed Value:	
	NOM_TEMP_BLD	0xE014
N2	generic N2 label	
	Label:	
	NLS_NOM_CONC_AWAY_N2	0x0002537c
	Observed Value:	
	NOM_CONC_AWAY_N2	0x537C
N2O	generic Nitrous Oxide label	

```
                Label:
                NLS_NOM_CONC_AWAY_N2O                   0x000251f0
                Observed Value:
                NOM_CONC_AWAY_N2O                       0x51F0
ISO             generic Isoflurane label
                Label:
                NLS_NOM_CONC_AWAY_ISOFL                 0x000251e8
                Observed Value:
                NOM_CONC_AWAY_ISOFL                     0x51E8
SEV             generic Sevoflurane label
                Label:
                NLS_NOM_CONC_AWAY_SEVOFL                0x000251e4
                Observed Value:
                NOM_CONC_AWAY_SEVOFL                    0x51E4
ENF             generic Enflurane label
                Label:
                NLS_NOM_CONC_AWAY_ENFL                  0x000251dc
                Observed Value:
                NOM_CONC_AWAY_ENFL                      0x51DC
HAL             generic Halothane label
                Label:
                NLS_NOM_CONC_AWAY_HALOTH                0x000251e0
                Observed Value:
                NOM_CONC_AWAY_HALOTH                    0x51E0
DES             generic Desflurane label
                Label:
                NLS_NOM_CONC_AWAY_DESFL                 0x000251d8
                Observed Value:
                NOM_CONC_AWAY_DESFL                     0x51D8
AGT             generic Agent label
                Label:
                NLS_NOM_CONC_AWAY_AGENT                 0x00025388
                Observed Value:
                NOM_CONC_AWAY_AGENT                     0x5388
AGT1            generic Agent1 label
                Label:
                NLS_GASES_NAMES_CONC_AWAY_AGENT1        0x805a5401
                Observed Value:
                NOM_CONC_AWAY_AGENT                     0x5388
AGT2            generic Agent2 label
                Label:
                NLS_GASES_NAMES_CONC_AWAY_AGENT2        0x805a5402
                Observed Value:
                NOM_CONC_AWAY_AGENT                     0x5388
P1              non-specific label for Pressure 1
                Label:
                NLS_NOM_EMFC_P1                         0x04010030
                Observed Value:
                NOM_PRESS_BLD                           0x4A00
P2              non-specific label for Pressure 2
                Label:
                NLS_NOM_EMFC_P2                         0x04010034
                Observed Value:
                NOM_PRESS_BLD                           0x4A00
P3              non-specific label for Pressure 3
                Label:
                NLS_NOM_EMFC_P3                         0x04010038
                Observed Value:
                NOM_PRESS_BLD                           0x4A00
P4              non-specific label for Pressure 4
                Label:
                NLS_NOM_EMFC_P4                         0x0401003c
                Observed Value:
                NOM_PRESS_BLD                           0x4A00
P5              non-specific label for Pressure 5
```

```
                Label:
                NLS_NOM_EMFC_P5                                 0x04010400
                Observed Value:
                NOM_PRESS_BLD                                   0x4A00
P6              non-specific label for Pressure 6
                Label:
                NLS_NOM_EMFC_P6                                 0x04010404
                Observed Value:
                NOM_PRESS_BLD                                   0x4A00
P7              non-specific label for Pressure 7
                Label:
                NLS_NOM_EMFC_P7                                 0x04010408
                Observed Value:
                NOM_PRESS_BLD                                   0x4A00
P8              non-specific label for Pressure 8
                Label:
                NLS_NOM_EMFC_P8                                 0x0401040c
                Observed Value:
                NOM_PRESS_BLD                                   0x4A00
IUP             Intra-Uterine Pressure
                Label:
                NLS_NOM_EMFC_IUP                                0x04010054
                Observed Value:
                NOM_PRESS_BLD                                   0x4A00
AUX             Auxiliary Wave/Parameter
                Label:
                NLS_NOM_EMFC_AUX                                0x040100b4
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
AWV             Airway Volume Wave
                Label:
                NLS_NOM_EMFC_AWV                                0x04010668
                Observed Value:
                NOM_METRIC_NOS                                  0xEFFF
LV1             Lead V1 - ECG wave label
                Label:
                NLS_NOM_EMFC_L_V1                               0x04010764
                Observed Value:
                NOM_ECG_ELEC_POTL_V1                            0x0103
LV2             Lead V2 - ECG wave label
                Label:
                NLS_NOM_EMFC_L_V2                               0x04010768
                Observed Value:
                NOM_ECG_ELEC_POTL_V2                            0x0104
LV3             Lead V3 - ECG wave label
                Label:
                NLS_NOM_EMFC_L_V3                               0x0401076c
                Observed Value:
                NOM_ECG_ELEC_POTL_V3                            0x0105
LV4             Lead V4 - ECG wave label
                Label:
                NLS_NOM_EMFC_L_V4                               0x04010770
                Observed Value:
                NOM_ECG_ELEC_POTL_V4                            0x0106
LV5             Lead V5 - ECG wave label
                Label:
                NLS_NOM_EMFC_L_V5                               0x04010774
                Observed Value:
                NOM_ECG_ELEC_POTL_V5                            0x0107
LV6             Lead V6 - ECG wave label
                Label:
                NLS_NOM_EMFC_L_V6                               0x04010778
                Observed Value:
                NOM_ECG_ELEC_POTL_V6                            0x0108
L_I             Lead I - ECG wave label
```

	Label:	
	NLS_NOM_EMFC_L_I	0x0401077c
	Observed Value:	
	NOM_ECG_ELEC_POTL_I	0x0101
L_II	Lead II - ECG wave label	
	Label:	
	NLS_NOM_EMFC_L_II	0x04010780
	Observed Value:	
	NOM_ECG_ELEC_POTL_II	0x0102
L_III	Lead III - ECG wave label	
	Label:	
	NLS_NOM_EMFC_L_III	0x04010784
	Observed Value:	
	NOM_ECG_ELEC_POTL_III	0x013D
LaVR	Lead aVR - ECG wave label	
	Label:	
	NLS_NOM_EMFC_L_aVR	0x04010788
	Observed Value:	
	NOM_ECG_ELEC_POTL_AVR	0x013E
LaVL	Lead aVL - ECG wave label	
	Label:	
	NLS_NOM_EMFC_L_aVL	0x0401078c
	Observed Value:	
	NOM_ECG_ELEC_POTL_AVL	0x013F
LaVF	Lead aVF - ECG wave label	
	Label:	
	NLS_NOM_EMFC_L_aVF	0x04010790
	Observed Value:	
	NOM_ECG_ELEC_POTL_AVF	0x0140
AWVex	Expiratory Airway Volume Wave. Measured in l.	
	Label:	
	NLS_NOM_EMFC_AWVex	0x04010794
	Observed Value:	
	NOM_METRIC_NOS	0xEFFF
PLETH2	PLETH from the second SpO2/PLETH module	
	Label:	
	NLS_NOM_EMFC_PLETH2	0x0401079c
	Observed Value:	
	NOM_PULS_OXIM_PLETH	0x4BB4
LTEEG	Left channel EEG wave	
	Label:	
	NLS_NOM_EMFC_LT_EEG	0x040107f0
	Observed Value:	
	NOM_EEG_ELEC_POTL_CRTX	0x592C
RTEEG	Right channel EEG wave	
	Label:	
	NLS_NOM_EMFC_RT_EEG	0x0401082c
	Observed Value:	
	NOM_EEG_ELEC_POTL_CRTX	0x592C
BP	Unspecified Blood Pressure	
	Label:	
	NLS_NOM_EMFC_BP	0x04010888
	Observed Value:	
	NOM_PRESS_BLD	0x4A00
AGTs	Anesthetic Agent - secondary agent	
	Label:	
	NLS_NOM_EMFC_AGTs	0x04010ce4
	Observed Value:	
	NOM_CONC_AWAY_AGENT	0x5388
Wave1	Placeholder for a Vuelink Flex Text: Wave Label	
	Label:	
	NLS_VUELINK_FLX1_NPS_TEXT_WAVE1	0x80aaf001
	Observed Value:	
	depends on configuration	
Wave2	Placeholder for a Vuelink Flex Text: Wave Label	

```
            Label:
            NLS_VUELINK_FLX1_NPS_TEXT_WAVE2                0x80aaf003
            Observed Value:
            depends on configuration
Wave3       Placeholder for a Vuelink Flex Text: Wave Label
            Label:
            NLS_VUELINK_FLX1_NPS_TEXT_WAVE3                0x80aaf005
            Observed Value:
            depends on configuration
Wave4       Placeholder for a Vuelink Flex Text: Wave Label
            Label:
            NLS_VUELINK_FLX1_NPS_TEXT_WAVE4                0x80aaf007
            Observed Value:
            depends on configuration
Wave5       Placeholder for a Vuelink Flex Text: Wave Label
            Label:
            NLS_VUELINK_FLX1_NPS_TEXT_WAVE5                0x80aaf009
            Observed Value:
            depends on configuration
Wave6       Placeholder for a Vuelink Flex Text: Wave Label
            Label:
            NLS_VUELINK_FLX1_NPS_TEXT_WAVE6                0x80aaf00b
            Observed Value:
            depends on configuration
Wave7       Placeholder for a Vuelink Flex Text: Wave Label
            Label:
            NLS_VUELINK_FLX1_NPS_TEXT_WAVE7                0x80aaf00d
            Observed Value:
            depends on configuration
Wave8       Placeholder for a Vuelink Flex Text: Wave Label
            Label:
            NLS_VUELINK_FLX1_NPS_TEXT_WAVE8                0x80aaf00f
            Observed Value:
            depends on configuration
```

Attribute IDs

The Attribute ID specifies the type of an attribute in the AttributeList. The IDs are taken from the Object Oriented Elements partition. Unknown attributes should be ignored.

Device P-Alarm List		
	NOM_ATTR_AL_MON_P_AL_LIST	0x0902
Device T-Alarm List		
	NOM_ATTR_AL_MON_T_AL_LIST	0x0904
Altitude		
	NOM_ATTR_ALTITUDE	0x090C
Application Area		
	NOM_ATTR_AREA_APPL	0x090D
Color		
	NOM_ATTR_COLOR	0x0911
Device Alert Condition		
	NOM_ATTR_DEV_AL_COND	0x0916
Display Resolution		
	NOM_ATTR_DISP_RES	0x0917
Visual Grid		
	NOM_ATTR_GRID_VIS_I16	0x091A
Association Invoke Id		
	NOM_ATTR_ID_ASSOC_NO	0x091D

Bed Label	NOM_ATTR_ID_BED_LABEL	0x091E
Object Handle	NOM_ATTR_ID_HANDLE	0x0921
Label	NOM_ATTR_ID_LABEL	0x0924
Label String	NOM_ATTR_ID_LABEL_STRING	0x0927
System Model	NOM_ATTR_ID_MODEL	0x0928
Product Specification	NOM_ATTR_ID_PROD_SPECN	0x092D
Object Type	NOM_ATTR_ID_TYPE	0x092F
Line Frequency	NOM_ATTR_LINE_FREQ	0x0935
System Localization	NOM_ATTR_LOCALIZN	0x0937
Metric Info Label	NOM_ATTR_METRIC_INFO_LABEL	0x093C
Metric Info Label String	NOM_ATTR_METRIC_INFO_LABEL_STR	0x093D
Metric Specification	NOM_ATTR_METRIC_SPECN	0x093F
Metric State	NOM_ATTR_METRIC _STAT	0x0940
Measure Mode	NOM_ATTR_MODE_MSMT	0x0945
Operating Mode	NOM_ATTR_MODE_OP	0x0946
Nomenclature Version	NOM_ATTR_NOM_VERS	0x0948
Compound Numeric Observed Value	NOM_ATTR_NU_CMPD_VAL_OBS	0x094B
Numeric Observed Value	NOM_ATTR_NU_VAL_OBS	0x0950
Patient BSA	NOM_ATTR_PT_BSA	0x0956
Pat Demo State	NOM_ATTR_PT_DEMOG_ST	0x0957
Patient Date of Birth	NOM_ATTR_PT_DOB	0x0958
Patient ID	NOM_ATTR_PT_ID	0x095A
Family Name	NOM_ATTR_PT_NAME_FAMILY	0x095C
Given Name	NOM_ATTR_PT_NAME_GIVEN	0x095D
Patient Sex	NOM_ATTR_PT_SEX	0x0961
Patient Type	NOM_ATTR_PT_TYPE	0x0962
Sample Array Calibration Specification	NOM_ATTR_SA_CALIB_I16	0x0964
Compound Sample Array Observed Value	NOM_ATTR_SA_CMPD_VAL_OBS	0x0967

Sample Array Physiological Range		
	NOM_ATTR_SA_RANGE_PHYS_I16	0x096A
Sample Array Specification		
	NOM_ATTR_SA_SPECN	0x096D
Sample Array Observed Value		
	NOM_ATTR_SA_VAL_OBS	0x096E
Scale and Range Specification		
	NOM_ATTR_SCALE_SPECN_I16	0x096F
Safety Standard		
	NOM_ATTR_STD_SAFETY	0x0982
System ID		
	NOM_ATTR_SYS_ID	0x0984
System Specification		
	NOM_ATTR_SYS_SPECN	0x0985
System Type		
	NOM_ATTR_SYS_TYPE	0x0986
Date and Time		
	NOM_ATTR_TIME_ABS	0x0987
Sample Period		
	NOM_ATTR_TIME_PD_SAMP	0x098D
Relative Time		
	NOM_ATTR_TIME_REL	0x098F
Absolute Time Stamp		
	NOM_ATTR_TIME_STAMP_ABS	0x0990
Relative Time Stamp		
	NOM_ATTR_TIME_STAMP_REL	0x0991
Unit Code		
	NOM_ATTR_UNIT_CODE	0x0996
MDS Status		
	NOM_ATTR_VMS_MDS_STAT	0x09A7
Patient Age		
	NOM_ATTR_PT_AGE	0x09D8
Patient Height		
	NOM_ATTR_PT_HEIGHT	0x09DC
Patient Weight		
	NOM_ATTR_PT_WEIGHT	0x09DF
Sample Array Fixed Values Specification		
	NOM_ATTR_SA_FIXED_VAL_SPECN	0x0A16
Patient Paced Mode		
	NOM_ATTR_PT_PACED_MODE	0x0A1E
Internal Patient ID		
	NOM_ATTR_PT_ID_INT	0xF001
Private Attribute		
	NOM_SAT_O2_TONE_FREQ	0xF008
Private Attribute		
	NOM_ATTR_CMPD_REF_LIST	0xF009
IP Address Information		
	NOM_ATTR_NET_ADDR_INFO	0xF100
Protocol Support		
	NOM_ATTR_PCOL_SUPPORT	0xF101
Notes1		
	NOM_ATTR_PT_NOTES1	0xF129
Notes2		
	NOM_ATTR_PT_NOTES2	0xF12A
Time for Periodic Polling		
	NOM_ATTR_TIME_PD_POLL	0xF13E

Patient BSA Formula		
	NOM_ATTR_PT_BSA_FORMULA	0xF1EC
Mds General System Info		
	NOM_ATTR_MDS_GEN_INFO	0xF1FA
no of prioritized objects for poll request		
	NOM_ATTR_POLL_OBJ_PRIO_NUM	0xF228
Numeric Object Priority List		
	NOM_ATTR_POLL_NU_PRIO_LIST	0xF239
Wave Object Priority List		
	NOM_ATTR_POLL_RTSA_PRIO_LIST	0xF23A

The attributes are arranged in the following attribute groups:

Alert Monitor Group		
	NOM_ATTR_GRP_AL_MON	0x0801
Metric Observed Value Group		
	NOM_ATTR_GRP_METRIC_VAL_OBS	0x0803
Patient Demographics Attribute Group		
	NOM_ATTR_GRP_PT_DEMOG	0x0807
System Application Attribute Group		
	NOM_ATTR_GRP_SYS_APPL	0x080A
System Identification Attribute Group		
	NOM_ATTR_GRP_SYS_ID	0x080B
System Production Attribute Group		
	NOM_ATTR_GRP_SYS_PROD	0x080C
VMO Dynamic Attribute Group		
	NOM_ATTR_GRP_VMO_DYN	0x0810
VMO Static Attribute Group		
	NOM_ATTR_GRP_VMO_STATIC	0x0811

Component IDs

The Component IDs specify system components such as the entries in the Production Specification attribute of the Medical Device Service object. A Component ID is a PrivateOid and is not assigned to any nomenclature partition.

for the overall product		
	ID_COMP_PRODUCT	0x0008
for the specific bundle		
	ID_COMP_CONFIG	0x0010
for the boot code		
	ID_COMP_BOOT	0x0018
mainboard component		
	ID_COMP_MAIN_BD	0x0050
application software component		
	ID_COMP_APPL_SW	0x0058

Unit Codes

The Unit Codes describe the dimension of a physiological measurement. They are grouped in the Units partition.

NOS	(no dimension)

	NOM_DIM_NOS	0x0000
/		
	NOM_DIM_DIV	0x0002
-	(no dimension)	
	NOM_DIM_DIMLESS	0x0200
%	(percentage)	
	NOM_DIM_PERCENT	0x0220
ppth	(parts per thousand)	
	NOM_DIM_PARTS_PER_THOUSAND	0x0240
ppm	(parts per million)	
	NOM_DIM_PARTS_PER_MILLION	0x0260
mol/mol	(mole per mole)	
	NOM_DIM_X_MOLE_PER_MOLE	0x0360
ppb	(parts per billion)	
	NOM_DIM_PARTS_PER_BILLION	0x02A0
ppt	(parts per trillion)	
	NOM_DIM_PARTS_PER_TRILLION	0x02C0
pH		
	NOM_DIM_PH	0x03E0
drop	(vital signs count drop)	
	NOM_DIM_DROP	0x0400
rbc	(vital signs count red blood cells)	
	NOM_DIM_RBC	0x0420
beat	(vital signs count beat)	
	NOM_DIM_BEAT	0x0440
breath	(vital signs count breath)	
	NOM_DIM_BREATH	0x0460
cell	(vital signs count cells)	
	NOM_DIM_CELL	0x0480
cough	(vital signs count cough)	
	NOM_DIM_COUGH	0x04A0
sigh	(vital signs count sigh)	
	NOM_DIM_SIGH	0x04C0
%PCV	(percent of packed cell volume)	
	NOM_DIM_PCT_PCV	0x04E0
m	(meter)	
	NOM_DIM_X_M	0x0500
cm	(centimeter)	
	NOM_DIM_CENTI_M	0x0511
mm	(millimeter)	
	NOM_DIM_MILLI_M	0x0512
µm	(micro-meter)	
	NOM_DIM_MICRO_M	0x0513
in	(inch)	
	NOM_DIM_X_INCH	0x0560
ml/m2	(used e.g. for SI and ITBVI)	
	NOM_DIM_MILLI_L_PER_M_SQ	0x0592
/m	(per meter)	
	NOM_DIM_PER_X_M	0x05A0
/mm	(per millimeter)	
	NOM_DIM_PER_MILLI_M	0x05B2
m2	(used e.g. for BSA calculation)	
	NOM_DIM_SQ_X_M	0x05C0
in2	(used e.g. for BSA calculation)	
	NOM_DIM_SQ_X_INCH	0x05E0
m3	(cubic meter)	

	NOM_DIM_CUBIC_X_M	0x0620
cm3	(cubic centimeter)	
	NOM_DIM_CUBIC_CENTI_M	0x0631
l	(liter)	
	NOM_DIM_X_L	0x0640
ml	(milli-liters, used e.g. for EVLW, ITBV, SV)	
	NOM_DIM_MILLI_L	0x0652
ml/breath	(milli-liter per breath)	
	NOM_DIM_MILLI_L_PER_BREATH	0x0672
/cm3	(per cubic centimeter)	
	NOM_DIM_PER_CUBIC_CENTI_M	0x0691
/l	(per liter)	
	NOM_DIM_PER_X_L	0x06A0
1/nl	(per nano-liter)	
	NOM_DIM_PER_NANO_LITER	0x06B4
g	(gram)	
	NOM_DIM_X_G	0x06C0
kg	(kilo-gram)	
	NOM_DIM_KILO_G	0x06C3
mg	(milli-gram)	
	NOM_DIM_MILLI_G	0x06D2
µg	(micro-gram)	
	NOM_DIM_MICRO_G	0x06D3
ng	(nono-gram)	
	NOM_DIM_NANO_G	0x06D4
lb	(pound)	
	NOM_DIM_X_LB	0x06E0
oz	(ounce)	
	NOM_DIM_X_OZ	0x0700
/g	(per gram)	
	NOM_DIM_PER_X_G	0x0720
g-m	(used e.g. for LVSW, RVSW)	
	NOM_DIM_X_G_M	0x0740
kg-m	(used e.g. for RCW, LCW)	
	NOM_DIM_KILO_G_M	0x0743
g-m/m2	(used e.g. for LVSWI and RVSWI)	
	NOM_DIM_X_G_M_PER_M_SQ	0x0760
kg-m/m2	(used e.g. for LCWI and RCWI)	
	NOM_DIM_KILO_G_M_PER_M_SQ	0x0763
kg-m2	(kilo-gram meter squared)	
	NOM_DIM_KILO_G_M_SQ	0x0783
kg/m2	(kilo-gram per square meter)	
	NOM_DIM_KG_PER_M_SQ	0x07A3
kg/m3	(kilo-gram per cubic meter)	
	NOM_DIM_KILO_G_PER_M_CUBE	0x07C3
g/cm3	(gram per cubic meter)	
	NOM_DIM_X_G_PER_CM_CUBE	0x07E0
mg/cm3	(milli-gram per cubic centimeter)	
	NOM_DIM_MILLI_G_PER_CM_CUBE	0x07F2
µg/cm3	(micro-gram per cubic centimeter)	
	NOM_DIM_MICRO_G_PER_CM_CUBE	0x07F3
ng/cm3	(nano-gram per cubic centimeter)	
	NOM_DIM_NANO_G_PER_CM_CUBE	0x07F4
g/l	(gram per liter)	
	NOM_DIM_X_G_PER_L	0x0800
g/dl	(used e.g. for Hb)	

	NOM_DIM_X_G_PER_DL	0x0840
mg/dl	(milli-gram per deciliter)	
	NOM_DIM_MILLI_G_PER_DL	0x0852
g/ml	(gram per milli-liter)	
	NOM_DIM_X_G_PER_ML	0x0860
mg/ml	(milli-gram per milli-liter)	
	NOM_DIM_MILLI_G_PER_ML	0x0872
µg/ml	(micro-gram per milli-liter)	
	NOM_DIM_MICRO_G_PER_ML	0x0873
ng/ml	(nano-gram per milli-liter)	
	NOM_DIM_NANO_G_PER_ML	0x0874
sec	(seconds)	
	NOM_DIM_SEC	0x0880
msec	(milli-seconds)	
	NOM_DIM_MILLI_SEC	0x0892
µsec	(micro-seconds)	
	NOM_DIM_MICRO_SEC	0x0893
min	(minutes)	
	NOM_DIM_MIN	0x08A0
hrs	(hours)	
	NOM_DIM_HR	0x08C0
days	(days)	
	NOM_DIM_DAY	0x08E0
weeks	(weeks)	
	NOM_DIM_WEEKS	0x0900
months	(months)	
	NOM_DIM_MON	0x0920
years	(years)	
	NOM_DIM_YR	0x0940
TOD	(time of day)	
	NOM_DIM_TOD	0x0960
date	(date)	
	NOM_DIM_DATE	0x0980
/sec	(per second)	
	NOM_DIM_PER_X_SEC	0x09A0
Hz	(hertz)	
	NOM_DIM_HZ	0x09C0
/min	(per minute, used e.g. for the PVC count numerical value)	
	NOM_DIM_PER_MIN	0x09E0
/hour	(per hour)	
	NOM_DIM_PER_HR	0x0A00
/day	(per day)	
	NOM_DIM_PER_DAY	0x0A20
/week	(per week)	
	NOM_DIM_PER_WK	0x0A40
/month	(per month)	
	NOM_DIM_PER_MO	0x0A60
/year	(per year)	
	NOM_DIM_PER_YR	0x0A80
bpm	(beats per minute, used e.g. for HR/PULSE)	
	NOM_DIM_BEAT_PER_MIN	0x0AA0
puls/min	(puls per minute)	
	NOM_DIM_PULS_PER_MIN	0x0AC0
rpm	(respiration breathes per minute)	
	NOM_DIM_RESP_PER_MIN	0x0AE0
m/sec	(meter per second)	

	NOM_DIM_X_M_PER_SEC	0x0B00
mm/sec	(speed for recordings)	
	NOM_DIM_MILLI_M_PER_SEC	0x0B12
l/min/m2	(used for CI)	
	NOM_DIM_X_L_PER_MIN_PER_M_SQ	0x0B20
ml/min/m2	(used for DO2I, VO2I, O2AVI)	
	NOM_DIM_MILLI_L_PER_MIN_PER_M_SQ	0x0B32
m2/sec	(square meter per second)	
	NOM_DIM_SQ_X_M_PER_SEC	0x0B40
cm2/sec	(square centimeter per second)	
	NOM_DIM_SQ_CENTI_M_PER_SEC	0x0B51
m3/sec	(cubic meter per second)	
	NOM_DIM_CUBIC_X_M_PER_SEC	0x0B60
cm3/sec	(cubic centimeter per second)	
	NOM_DIM_CUBIC_CENTI_M_PER_SEC	0x0B71
l/sec	(liter per second)	
	NOM_DIM_X_L_PER_SEC	0x0BE0
l/min	(liter per minutes, used e.g. for MINVOL)	
	NOM_DIM_X_L_PER_MIN	0x0C00
dl/min	(deciliter per second)	
	NOM_DIM_DECI_L_PER_MIN	0x0C10
ml/min	(used for DO2, VO2, ALVENT)	
	NOM_DIM_MILLI_L_PER_MIN	0x0C12
l/hour	(liter per hour)	
	NOM_DIM_X_L_PER_HR	0x0C20
ml/hour	(milli-liter per hour)	
	NOM_DIM_MILLI_L_PER_HR	0x0C32
l/day	(liter per day)	
	NOM_DIM_X_L_PER_DAY	0x0C40
ml/day	(milli-liter per day)	
	NOM_DIM_MILLI_L_PER_DAY	0x0C52
ml/kg	(used e.g. for EVLWI)	
	NOM_DIM_MILLI_L_PER_KG	0x0C72
kg/sec	(kilo-gram per second)	
	NOM_DIM_KILO_G_PER_SEC	0x0CE3
g/min	(gram per minute)	
	NOM_DIM_X_G_PER_MIN	0x0D00
kg/min	(kilo-gram per minute)	
	NOM_DIM_KILO_G_PER_MIN	0x0D03
mg/min	(milli-gram per minute)	
	NOM_DIM_MILLI_G_PER_MIN	0x0D12
µg/min	(micro-gram per minute)	
	NOM_DIM_MICRO_G_PER_MIN	0x0D13
ng/min	(nano-gram per minute)	
	NOM_DIM_NANO_G_PER_MIN	0x0D14
g/hour	(gram per hour)	
	NOM_DIM_X_G_PER_HR	0x0D20
kg/hour	(kilo-gram per hour)	
	NOM_DIM_KILO_G_PER_HR	0x0D23
mg/hour	(milli-gram per hour)	
	NOM_DIM_MILLI_G_PER_HR	0x0D32
µg/hour	(micro-gram per hour)	
	NOM_DIM_MICRO_G_PER_HR	0x0D33
ng/hr	(nano-gram per hour)	
	NOM_DIM_NANO_G_PER_HR	0x0D34
kg/day	(kilo-gram per day)	

	NOM_DIM_KILO_G_PER_DAY	0x0D43
g/kg/min	(gram per kilo-gram per minute)	
	NOM_DIM_X_G_PER_KG_PER_MIN	0x0D80
mg/kg/min	(milli-gram per kilo-gram per minute)	
	NOM_DIM_MILLI_G_PER_KG_PER_MIN	0x0D92
µg/kg/min	(micro-gram per kilo-gram per minute)	
	NOM_DIM_MICRO_G_PER_KG_PER_MIN	0x0D93
ng/kg/min	(nano-gram per kilo-gram per minute)	
	NOM_DIM_NANO_G_PER_KG_PER_MIN	0x0D94
g/kg/hour	(gram per kilo-gram per hour)	
	NOM_DIM_X_G_PER_KG_PER_HR	0x0DA0
mg/kg/hour	(mili-gram per kilo-gram per hour)	
	NOM_DIM_MILLI_G_PER_KG_PER_HR	0x0DB2
µg/kg/hour	(micro-gram per kilo-gram per hour)	
	NOM_DIM_MICRO_G_PER_KG_PER_HR	0x0DB3
ng/kg/hour	(nano-gram per kilo-gram per hour)	
	NOM_DIM_NANO_G_PER_KG_PER_HR	0x0DB4
kg/l/sec	(kilo-gram per liter per second)	
	NOM_DIM_KILO_G_PER_L_SEC	0x0DE3
kg/m/sec	(kilo-gram per meter per second)	
	NOM_DIM_KILO_G_PER_M_PER_SEC	0x0E63
kg-m/sec	(kilo-gram meter per second)	
	NOM_DIM_KILO_G_M_PER_SEC	0x0E83
N-s	(newton seconds)	
	NOM_DIM_X_NEWTON_SEC	0x0EA0
N	(newton)	
	NOM_DIM_X_NEWTON	0x0EC0
Pa	(pascal)	
	NOM_DIM_X_PASCAL	0x0F00
hPa	(hekto-pascal)	
	NOM_DIM_HECTO_PASCAL	0x0F02
kPa	(kilo-pascal)	
	NOM_DIM_KILO_PASCAL	0x0F03
mmHg	(mm mercury)	
	NOM_DIM_MMHG	0x0F20
cmH2O	(centimeter H20)	
	NOM_DIM_CM_H2O	0x0F40
mBar	(milli-bar)	
	NOM_DIM_MILLI_BAR	0x0F72
J	(Joules)	
	NOM_DIM_X_JOULES	0x0F80
eV	(electronvolts)	
	NOM_DIM_EVOLT	0x0FA0
W	(watt)	
	NOM_DIM_X_WATT	0x0FC0
mW	(milli-watt)	
	NOM_DIM_MILLI_WATT	0x0FD2
nW	(nano-watt)	
	NOM_DIM_NANO_WATT	0x0FD4
pW	(pico-watt)	
	NOM_DIM_PICO_WATT	0x0FD5
Dyn-sec/cm^5	(dyne second per cm^5)	
	NOM_DIM_X_DYNE_PER_SEC_PER_CM5	0x1020
A	(ampere)	
	NOM_DIM_X_AMPS	0x1040
mA	(milli-ampere, used e.g. for the battery indications)	

	NOM_DIM_MILLI_AMPS	0x1052
C	(coulomb)	
	NOM_DIM_X_COULOMB	0x1060
µC	(micro-coulomb)	
	NOM_DIM_MICRO_COULOMB	0x1073
V	(volts)	
	NOM_DIM_X_VOLT	0x10A0
mV	(milli-volt)	
	NOM_DIM_MILLI_VOLT	0x10B2
µV	(micro-volt)	
	NOM_DIM_MICRO_VOLT	0x10B3
Ohm		
	NOM_DIM_X_OHM	0x10C0
kOhm	(kilo-ohm)	
	NOM_DIM_OHM_K	0x10C3
F	(farad)	
	NOM_DIM_X_FARAD	0x1100
°K	(kelvin)	
	NOM_DIM_KELVIN	0x1120
°F	(degree-fahrenheit)	
	NOM_DIM_FAHR	0x1140
cd	(candela)	
	NOM_DIM_X_CANDELA	0x1180
mOsm	(milli-osmole)	
	NOM_DIM_MILLI_OSM	0x11B2
mol	(mole)	
	NOM_DIM_X_MOLE	0x11C0
mmol	(milli-mole)	
	NOM_DIM_MILLI_MOLE	0x11D2
mEq	(milli-equivalents)	
	NOM_DIM_MILLI_EQUIV	0x11F2
mOsm/l	(milli-osmole per liter)	
	NOM_DIM_MILLI_OSM_PER_L	0x1212
mmol/l	(used for HB)	
	NOM_DIM_MILLI_MOLE_PER_L	0x1272
µmol/l	(micro-mol per liter)	
	NOM_DIM_MICRO_MOLE_PER_L	0x1273
mEq/l	(milli-equivalents per liter)	
	NOM_DIM_MILLI_EQUIV_PER_L	0x12F2
mEq/day	(milli-equivalents per day)	
	NOM_DIM_MILLI_EQUIV_PER_DAY	0x1452
i.u.	(international unit)	
	NOM_DIM_X_INTL_UNIT	0x1560
mi.u.	(mili-international unit)	
	NOM_DIM_MILLI_INTL_UNIT	0x1572
i.u./cm3	(international unit per cubic centimeter)	
	NOM_DIM_X_INTL_UNIT_PER_CM_CUBE	0x1580
mi.u./cm3	(mili-international unit per cubic centimeter)	
	NOM_DIM_MILLI_INTL_UNIT_PER_CM_CUBE	0x1592
i.u./ml	(international unit per milli-liter)	
	NOM_DIM_X_INTL_UNIT_PER_ML	0x15E0
i.u./min	(international unit per minute)	
	NOM_DIM_X_INTL_UNIT_PER_MIN	0x1620
mi.u./ml	(milli-international unit per milli-liter)	
	NOM_DIM_MILLI_INTL_UNIT_PER_ML	0x15F2
mi.u./min	(milli-international unit per minute)	

	NOM_DIM_MILLI_INTL_UNIT_PER_MIN	0x1632
i.u./hour	(international unit per hour)	
	NOM_DIM_X_INTL_UNIT_PER_HR	0x1640
mi.u./hour	(milli-international unit per hour)	
	NOM_DIM_MILLI_INTL_UNIT_PER_HR	0x1652
i.u./kg/min	(international unit per kilo-gram per minute)	
	NOM_DIM_X_INTL_UNIT_PER_KG_PER_MIN	0x16A0
mi.u./kg/min	(milli-international unit per kilo-gram per minute)	
	NOM_DIM_MILLI_INTL_UNIT_PER_KG_PER_MIN	0x16B2
i.u./kg/hour	(international unit per kilo-gram per hour)	
	NOM_DIM_X_INTL_UNIT_PER_KG_PER_HR	0x16C0
mi.u./kg/hour	(milli-international unit per kilo-gram per hour)	
	NOM_DIM_MILLI_INTL_UNIT_PER_KG_PER_HR	0x16D2
ml/cmH2O	(milli-liter per centimeter H2O)	
	NOM_DIM_MILLI_L_PER_CM_H2O	0x1712
cmH2O/l/sec	(centimeter H2O per second)	
	NOM_DIM_CM_H2O_PER_L_PER_SEC	0x1720
ml2/sec	(milli-liter per second)	
	NOM_DIM_MILLI_L_SQ_PER_SEC	0x1752
cmH2O/%	(centimeter H2O per percent)	
	NOM_DIM_CM_H2O_PER_PERCENT	0x1760
DS*m2/cm5	(used for SVRI and PVRI)	
	NOM_DIM_DYNE_SEC_PER_M_SQ_PER_CM_5	0x1780
°C	(degree-celsius)	
	NOM_DIM_DEGC	0x17A0
cmH2O/l	(centimeter H2O per liter)	
	NOM_DIM_CM_H2O_PER_L	0x1800
mmHg/%	(milli-meter mercury per percent)	
	NOM_DIM_MM_HG_PER_PERCENT	0x1820
kPa/%	(kilo-pascal per percent)	
	NOM_DIM_KILO_PA_PER_PERCENT	0x1843
mAh	(milli-ampere per hour, used e.g. for the battery indications)	
	NOM_DIM_MILLI_AMP_HR	0x17D2
ml/dl	(used for CaO2, CvO2, Ca-vO2)	
	NOM_DIM_MILLI_L_PER_DL	0x1912
dB	(decibel)	
	NOM_DIM_DECIBEL	0x1920
g/mg	(gram per milli-gram)	
	NOM_DIM_X_G_PER_MILLI_G	0x1940
mg/mg	(milli-gram per milli-gram)	
	NOM_DIM_MILLI_G_PER_MILLI_G	0x1952
bpm/l	(beats per minute per liter)	
	NOM_DIM_BEAT_PER_MIN_PER_X_L	0x1960
bpm/ml	(beats per minute per milli-liter)	
	NOM_DIM_BEAT_PER_MIN_PER_MILLI_L	0x1972
1/(min*l)	(per minute per liter)	
	NOM_DIM_PER_X_L_PER_MIN	0x1980
m/min	(meter per minute)	
	NOM_DIM_X_M_PER_MIN	0x19A0
cm/min	(speed for recordings)	
	NOM_DIM_CENTI_M_PER_MIN	0x19B1
complx		
	NOM_DIM_COMPLEX	0xF000
count	(count as a dimension)	
	NOM_DIM_COUNT	0xF001
part	(part)	

	NOM_DIM_PART	0xF002
puls	(puls)	
	NOM_DIM_PULS	0xF003
µV p-p	(micro-volt peak to peak)	
	NOM_DIM_UV_PP	0xF004
µV2	(micro-volt square)	
	NOM_DIM_UV_SQ	0xF005
lumen	(lumen)	
	NOM_DIM_LUMEN	0xF007
lb/in2	(pound per square inch)	
	NOM_DIM_LB_PER_INCH_SQ	0xF008

Alert Codes

The first column in the tables below shows the alert source, the second column shows the associated alert code and the third column contains the alert text which would be displayed by the monitor. The XXX in the alert text is a placeholder for the actual alert source. It is filled depending on the alert source. Note that the alert text depends on the localization of your monitor.

The least significant bit of the alert codes listed below is used to identify the source of an alert (refer to “Alert Monitor Object” on page 85). If the alert code is marked with a (*), the associated alert source is from the object oriented nomenclature partition and hence the least significant bit of the alert code is set to 1.

ECG/HR/Arrhy

NOM_ECG_ELEC_POTL	NOM_EVT_EQUIP_MALF	XXX EQUIP MALF
	NOM_EVT_LEADS_OFF	XXX LEADS OFF
	NOM_EVT_NOISY	XXX NOISY SIGN.
NOM_ECG_LEAD_C	NOM_EVT_LEAD_DISCONN	XXX LEAD OFF
NOM_ECG_LEAD_RA	NOM_EVT_NOISY	ECG EL. NOISY XXX
NOM_ECG_LEAD_LA		
NOM_ECG_LEAD_LL		
NOM_ECG_LEAD_RL		
NOM_ECG_LEAD_C1FR		
NOM_ECG_LEAD_C2FR		
NOM_ECG_LEAD_C3FR		
NOM_ECG_LEAD_C4FR		
NOM_ECG_LEAD_C5FR		
NOM_ECG_LEAD_C6FR		
NOM_ECG_LEAD_AS		
NOM_ECG_LEAD_AI		
NOM_ECG_LEAD_ES		
NOM_ECG_ELEC_POTL	NOM_EVT_SIG_UNANALYZEABLE	CANNOT ANALYZE XXX
NOM_ECG_ELEC_POTL	NOM_EVT_UNDEF	XXX UNKN. ALERT

NOM_ECG_CARD_BEAT_RATE	NOM_EVT_ECG_ASYSTOLE	ASYSTOLE
	NOM_EVT_ECG_V_FIB_TACHY	VENT FIB/TACH
	NOM_EVT_ECG_BRADY_EXTREME	EXTREME BRADY
	NOM_EVT_ECG_TACHY_EXTREME	EXTREME TACHY
	NOM_EVT_LO	XXX LOW
	NOM_EVT_HI	XXX HIGH
	NOM_EVT_ECG_PACER_NOT_PACING	PACER NT PACING
	NOM_EVT_ECG_PACING_NON_CAPT	PACER NOT CAPT
	NOM_EVT_ECG_SV_TACHY	SVT
	NOM_EVT_ECG_BEAT_MISSED	MISSED BEAT
	NOM_EVT_ECG_PAUSE	PAUSE
	NOM_EVT_ECG_CARD_BEAT_RATE_IRREG	IRREGULAR HR
NOM_ECG_V_P_C_CNT	NOM_EVT_STAT_ECG_AL_SOME_OFF	SOME ECG ALRMS OFF
	NOM_EVT_STAT_ECG_AL_ALL_OFF	ALL ECG ALARMS OFF
	NOM_EVT_ECG_V_TACHY	VTACH
	NOM_EVT_ECG_V_P_C_RATE	PVCs/min HIGH
	NOM_EVT_ECG_V_RHY	VENT RHYTHM
	NOM_EVT_ECG_V_P_C_RUN	RUN PVCs HIGH
	NOM_EVT_ECG_V_P_C_PAIR	PAIR PVCs
	NOM_EVT_ECG_V_P_C_RonT	R-ON-T PVCs
	NOM_EVT_ECG_BIGEM	VENT BIGEMINY
	NOM_EVT_ECG_V_TRIGEM	VENT TRIGEMINY
	NOM_EVT_ECG_V_TACHY_NON_SUST	NON-SUSTAIN VT
	NOM_EVT_ECG_V_P_C_MULTIFORM	MULTIFORM PVCs

ST

NOM_ECG_AMPL_ST	NOM_EVT_SIG_UNANALYZEABLE	CANNOT ANALYZE XXX

NOM_ECG_AMPL_ST_I	NOM_EVT_LO	XXX LOW
NOM_ECG_AMPL_ST_II	NOM_EVT_HI	XXX HIGH
NOM_ECG_AMPL_ST_III		
NOM_ECG_AMPL_ST_AVF		
NOM_ECG_AMPL_ST_AVL		
NOM_ECG_AMPL_ST_AVR		
NOM_ECG_AMPL_ST_V		
NOM_ECG_AMPL_ST_MCL		
NOM_ECG_AMPL_ST_V1		
NOM_ECG_AMPL_ST_V2		
NOM_ECG_AMPL_ST_V3		
NOM_ECG_AMPL_ST_V4		
NOM_ECG_AMPL_ST_V5		
NOM_ECG_AMPL_ST_V6		
NOM_ECG_AMPL_ST	NOM_EVT_ST_MULTI	ST MULTI XXX,XXX

Resp

NOM_RESP	NOM_EVT_LEADS_OFF	XXX LEADS OFF
	NOM_EVT_ERRATIC	XXX ERRATIC
NOM_RESP_RATE	NOM_EVT_APNEA	APNEA
	NOM_EVT_LO	XXX LOW
	NOM_EVT_HI	XXX HIGH

Derived Measurements

NOM_PRESS_CEREB_PERF	NOM_EVT_ADVIS_SRC_CHK	XXX CHK SOURCES
NOM_RES_VASC_SYS		
NOM_RES_VASC_SYS_INDEX		
NOM_TEMP_DIFF		
NOM_SAT_DIFF_O2_ART_VEN		
NOM_PULS_OXIM_SAT_O2_DIFF		
NOM_RATE_DIFF_CARD_BEAT_PULSE		
NOM_PRESS_CEREB_PERF	NOM_EVT_ADVIS_UNIT_CHK	XXX CHK UNITS
NOM_RES_VASC_SYS		
NOM_RES_VASC_SYS_INDEX		
NOM_TEMP_DIFF		
NOM_RES_VASC_SYS	NOM_EVT_ADVIS_PRESUMED_CVP	XXX SET CVP USED
NOM_RES_VASC_SYS_INDEX		
NOM_PRESS_CEREB_PERF	NOM_EVT_HI	XXX HIGH
NOM_PRESS_CEREB_PERF	NOM_EVT_LO	XXX LOW

C.O./CCO

NOM_DEV_ANALY_CARD_OUTPUT_VMD	NOM_EVT_EQUIP_MALF*	XXX EQUIP MALF
NOM_OUTPUT_CARD_CTS	NOM_EVT_XDUCR_DISCONN	CCO/Tbl NO TRANSD.
NOM_OUTPUT_CARD		XXX NO TRANSDUC
NOM_TEMP_BLD	NOM_EVT_RANGE_ERR	XXX OVERRANGE
	NOM_EVT_MSMT_RANGE_OVER	XXX OVERRANGE
NOM_OUTPUT_CARD_CTS	NOM_EVT_UNSUPPORTED	CCO NOT SUPPORTED
	NOM_EVT_ADVIS_SRC_CHK	CCO NO XXX
	NOM_EVT_ADVIS_SRC_CHK	CCO XXX INVALID
	NOM_EVT_STAT_PULSE_SRC_RANGE_OVER	CCO PULSE OVERRANG
	NOM_EVT_ADVIS_CALIB_REQD	CCO NO CALIBRATION
	NOM_EVT_STAT_PRESS_SRC_RANGE_OVER	CCO PRESS OVERRANG
	NOM_EVT_SIG_UNANALYZEABLE	CCO BAD PRESS SIGN
	NOM_EVT_MSMT_RANGE_OVER	XXX OVERRANGE
	NOM_EVT_ADVIS_CALIB_AND_ZERO_CHK	CCO RECALIBRATE
NOM_OUTPUT_CARD_INDEX_CTS	NOM_EVT_ADVIS_BSA_REQD	CCI NO BSA
	NOM_EVT_MSMT_RANGE_OVER	XXX OVERRANGE
NOM_TEMP_BLD	NOM_EVT_HI	XXX HIGH
NOM_OUTPUT_CARD_CTS	NOM_EVT_LO	XXX LOW
NOM_OUTPUT_CARD_INDEX_CTS		

EEG

NOM_EEG_ELEC_POTL_CRTX	NOM_EVT_EQUIP_MALF	XXXEQUIP MALF
	NOM_EVT_XDUCR_DISCONN	XXX NO TRANSDUC
	NOM_EVT_LEADS_OFF	XXX LEADS OFF
	NOM_EVT_MSMT_RANGE_OVER	XXX OVERRANGE
	NOM_EVT_IMPED_HI	EEG IMPEDANCE HIGH
	NOM_EVT_MUSCLE_NOISE	EEG MUSCLE NOISE
	NOM_EVT_LINE_NOISE	EEG LINE NOISE
NOM_OBJ_CHAN_1	NOM_EVT_LEAD_DISCONN*	EEG2 LEAD OFF XXX
NOM_OBJ_CHAN_2	NOM_EVT_LEADS_OFF*	XXX LEADS OFF
	NOM_EVT_MSMT_RANGE_OVER*	XXX OVERRANGE
	NOM_EVT_IMPED_HI*	EEGX IMPED. HIGH
	NOM_EVT_IMPEDS_HI*	EEGX IMPED. HIGH
NOM_OBJ_CHAN_1	NOM_EVT_MUSCLE_NOISE*	XXX MUSCLE NOISE
NOM_OBJ_CHAN_2	NOM_EVT_LINE_NOISE*	XXX LINE NOISE

BIS

NOM_DEV_ANALY_BISPECTRAL_INDEX_VMD	NOM_EVT_EQUIP_MALF*	XXX EQUIP MALF
	NOM_EVT_DISCONN*	BIS ENGINE DISCONN
	NOM_EVT_VOLTAGE_OUT_OF_RANGE*	BIS OVERCURRENT
NOM_EEG_BISPECTRAL_INDEX	NOM_EVT_INCOMPAT	BIS ENGINE INCOMPT
NOM_DEV_ANALY_BISPECTRAL_INDEX_VMD	NOM_EVT_MALF*	BIS ENGINE MALFUNC
	NOM_EVT_XDUCR_DISCONN*	BIS DSC DISCONN
	NOM_EVT_STAT_FW_UPDATE_IN_PROGRESS*	BIS DSC UPDATE
NOM_OBJ_XDUCR	NOM_EVT_INCOMPAT*	BIS DSC INCOMPT
NOM_DEV_ANALY_BISPECTRAL_INDEX_VMD	NOM_EVT_XDUCR_MALF*	BIS DSC MALFUNC
	NOM_EVT_SENSOR_DISCONN*	BIS SENSOR DISCONN
	NOM_EVT_SENSOR_MALF*	BIS SENSOR MALFUNC
NOM_OBJ_SENSOR	NOM_EVT_INCOMPAT*	BIS SENSR INCOMPAT
	NOM_EVT_EXH*	BIS SENSOR USAGE
NOM_ELECTRODE_IMPED	NOM_EVT_ADVIS_CHK	BIS IMPEDANCE CHCK
NOM_EEG_BISPECTRAL_INDEX	NOM_EVT_LEAD_DISCONN	BIS LEAD OFF
	NOM_EVT_IMPED_HI	BIS HIGH IMPEDANCE
NOM_EEG_BIS_SIG_QUAL_INDEX	NOM_EVT_SIG_LO	BIS SQI < 15%
NOM_EEG_BISPECTRAL_INDEX	NOM_EVT_ADVIS_CHK	BIS IMPEDANCE CHCK
	NOM_EVT_LEAD_DISCONN	BIS LEAD OFF
	NOM_EVT_IMPED_HI	BIS HIGH IMPEDANCE
NOM_EEG_BIS_SIG_QUAL_INDEX	NOM_EVT_LO	BIS SQI < 50%
NOM_EEG_ELEC_POTL_CRTX	NOM_EVT_ABSENT	BIS ISOELECTRC EEG
NOM_EEG_BISPECTRAL_INDEX	NOM_EVT_HI	XXX HIGH
	NOM_EVT_LO	XXX LOW

Temp

NOM_TEMP	NOM_EVT_EQUIP_MALF	XXX EQUIP MALF
NOM_TEMP_RECT	NOM_EVT_XDUCR_DISCONN	XXX NO TRANSDUC
NOM_TEMP_BLD	NOM_EVT_MSMT_RANGE_OVER	XXX OVERRANGE
NOM_TEMP_CORE	NOM_EVT_HI	XXX HIGH
NOM_TEMP_SKIN	NOM_EVT_LO	XXX LOW
NOM_TEMP_ESOPH		
NOM_TEMP_NASOPH		
NOM_TEMP_ART		
NOM_TEMP_VEN		

Invasive Pressure

NOM_PRESS_BLD_ART_ABP	NOM_EVT_EQUIP_MALF	XXX EQUIP MALF
NOM_PRESS_BLD_ART_ABP_SYS	NOM_EVT_XDUCR_DISCONN	XXX NO TRANSDUCER
NOM_PRESS_BLD_ART_ABP_DIA	NOM_EVT_XDUCR_MALF	XXX TRANSDUC MALF
NOM_PRESS_BLD_ART_ABP_MEAN	NOM_EVT_MSMT_RANGE_OVER	XXX OVERRANGE
NOM_PRESS_BLD_ART	NOM_EVT_WAVE_ARTIF_ERR	XXX ARTIFACT
NOM_PRESS_BLD_ART_SYS	NOM_EVT_ADVIS_GAIN_DECR	XXX REDUCE SIZE
NOM_PRESS_BLD_ART_DIA	NOM_EVT_WAVE_OSCIL_ABSENT	XXX NON-PULSATILE
NOM_PRESS_BLD_ART_MEAN	NOM_EVT_NOISY	XXX NOISY SIGNAL
NOM_PRESS_BLD_AORT	NOM_EVT_HI	XXX HIGH
NOM_PRESS_BLD_AORT_SYS	NOM_EVT_LO	XXX LOW
NOM_PRESS_BLD_AORT_DIA	NOM_EVT_MSMT_DISCONN	XXX DISCONNECT
NOM_PRESS_BLD_AORT_MEAN	NOM_EVT_ADVIS_CALIB_AND_ZERO_CHK	XXX ZERO+CHECK CAL
NOM_PRESS_BLD_ART_PULM		
NOM_PRESS_BLD_ART_PULM_SYS		
NOM_PRESS_BLD_ART_PULM_DIA		
NOM_PRESS_BLD_ART_PULM_MEAN		
NOM_PRESS_BLD_VEN_CENT		
NOM_PRESS_BLD_VEN_CENT_SYS		
NOM_PRESS_BLD_VEN_CENT_DIA		
NOM_PRESS_BLD_VEN_CENT_MEAN		
NOM_PRESS_BLD_ATR_RIGHT		
NOM_PRESS_BLD_ATR_RIGHT_SYS		
NOM_PRESS_BLD_ATR_RIGHT_DIA		
NOM_PRESS_BLD_ATR_RIGHT_MEAN		

NOM_PRESS_BLD_ATR_LEFT	NOM_EVT_EQUIP_MALF	XXX EQUIP MALF
NOM_PRESS_BLD_ATR_LEFT_SYS	NOM_EVT_XDUCR_DISCONN	XXX NO TRANSDUCER
NOM_PRESS_BLD_ATR_LEFT_DIA	NOM_EVT_XDUCR_MALF	XXX TRANSDUC MALF
NOM_PRESS_BLD_ATR_LEFT_MEAN	NOM_EVT_MSMT_RANGE_OVER	XXX OVERRANGE
NOM_PRESS_INTRA_CRAN	NOM_EVT_WAVE_ARTIF_ERR	XXX ARTIFACT
NOM_PRESS_INTRA_CRAN_SYS	NOM_EVT_ADVIS_GAIN_DECR	XXX REDUCE SIZE
NOM_PRESS_INTRA_CRAN_DIA	NOM_EVT_WAVE_OSCIL_ABSENT	XXX NON-PULSATILE
NOM_PRESS_INTRA_CRAN_MEAN	NOM_EVT_NOISY	XXX NOISY SIGNAL
NOM_PRESS_BLD_ART_UMB	NOM_EVT_HI	XXX HIGH
NOM_PRESS_BLD_ART_UMB_SYS	NOM_EVT_LO	XXX LOW
NOM_PRESS_BLD_ART_UMB_DIA	NOM_EVT_MSMT_DISCONN	XXX DISCONNECT
NOM_PRESS_BLD_ART_UMB_MEAN	NOM_EVT_ADVIS_CALIB_AND_ZERO_CHK	XXX ZERO+CHECK CAL
NOM_PRESS_BLD_VEN_UMB		
NOM_PRESS_BLD_VEN_UMB_SYS		
NOM_PRESS_BLD_VEN_UMB_DIA		
NOM_PRESS_BLD_VEN_UMB_MEAN		
NOM_PRESS_BLD		
NOM_PRESS_BLD_SYS		
NOM_PRESS_BLD_DIA		
NOM_PRESS_BLD_MEAN		
NOM_PULS_RATE	NOM_EVT_HI	XXX HIGH
	NOM_EVT_LO	XXX LOW
	NOM_EVT_BRADY	XXX BRADY (Pulse)
	NOM_EVT_TACHY	XXX TACHY (Pulse)

SpO_2

NOM_PULS_OXIM_SAT_O2	NOM_EVT_EQUIP_MALF	XXX EQUIP MALF
NOM_PULS_OXIM_SAT_O2_ART_LEFT	NOM_EVT_SENSOR_MALF	XXX SENSOR MALF

NOM_PULS_OXIM_SAT_O2_ART_RIGHT	NOM_EVT_XDUCR_DISCONN	XXX NO SENSOR
	NOM_EVT_MSMT_INTERF_ERR	XXX INTERFERNCE
	NOM_EVT_NOISY	XXX NOISY SIGN.
	NOM_EVT_WAVE_OSCIL_ABSENT	XXX NON-PULSAT.
	NOM_EVT_ERRATIC	XXX ERRATIC
	NOM_EVT_SUST	XXX EXTD.UPDATE
	NOM_EVT_SIG_LO	XXX LOW PERF
	NOM_EVT_HI	XXX HIGH
	NOM_EVT_LO	XXX LOW
	NOM_EVT_DESAT	XXX DESAT
	NOM_EVT_ADVIS_SENSOR_CHK	XXX UNKN. SENSOR
	NOM_EVT_STAT_FW_UPDATE_IN_PROGRESS	XXX UPGRADE
	NOM_EVT_STAT_LEARN	XXX SEARCHING
	NOM_EVT_MSMT_RANGE_UNDER	XXX PULSE?
	NOM_EVT_SENSOR_DISCONN	XXX SENSOR OFF
	NOM_EVT_SIG_QUALITY	XXX POOR SIGNAL
NOM_PULS_OXIM_PULS_RATE	NOM_EVT_LO	XXX LOW
	NOM_EVT_HI	XXX HIGH
	NOM_EVT_BRADY	XXX BRADY (Pulse)
	NOM_EVT_TACHY	XXX TACHY (Pulse)

SvO_2

NOM_SAT_O2_VEN	NOM_EVT_EQUIP_MALF	XXX EQUIP MALF
	NOM_EVT_CONFIG_ERR	SvO2 CONFIGURATION
	NOM_EVT_STAT_OPT_MOD_SENSOR_CONN	SvO2 CONNCT OPTMOD
	NOM_EVT_OPTIC_MODULE_ABSENT	SvO2 NO OPTMOD
	NOM_EVT_STAT_CALIB_PREINS_RUNNING	SvO2 PRE-INS CALIB
	NOM_EVT_CALIB_FAIL	SvO2 CAL FAILED
	NOM_EVT_ADVIS_CALIB_REQD	SVO2 CAL REQUIRED
	NOM_EVT_STAT_CALIB_MODE	SvO2 CAL MODE
	NOM_EVT_SIG_LO	SvO2 LOW LIGHT
	NOM_EVT_MSMT_ERR	SvO2 UNABL TO MEAS
	NOM_EVT_INTENS_LIGHT_ERR	SvO2 LIGHT INTENS
	NOM_EVT_STAT_CALIB_LIGHT_RUNNING	SvO2 LIGHT CALIB
	NOM_EVT_STAT_CALIB_INVIVO_RUNNING	SvO2 IN-VIVO CALIB
	NOM_EVT_STAT_OPT_MOD_SENSOR_WARMING	SvO2 OPTMOD WARMUP
	NOM_EVT_OPTIC_MODULE_DEFECT	SvO2 OPTMOD DEFECT
	NOM_EVT_HI	XXX HIGH
	NOM_EVT_LO	XXX LOW

CO_2

NOM_AWAY_CO2	NOM_EVT_EQUIP_MALF	CO2 EQUIP MALF
	NOM_EVT_EQUIP_MALF	CO2 EQUIP MALF
	NOM_EVT_XDUCR_DISCONN	XXX NO TRANSDUC
	NOM_EVT_CALIB_FAIL	CO2 FAILED CAL
	NOM_EVT_WAIT_CAL	CO2 WAIT CAL2
	NOM_EVT_STAT_CALIB_RUNNING	CO2 CAL RUNNING
	NOM_EVT_STAT_CALIB_MODE	CO2 CAL MODE
	NOM_EVT_ADVIS_CALIB_AND_ZERO_CHK	CO2 CHECK CAL
	NOM_EVT_STAT_SENSOR_WARMING	CO2 SENSOR WARMUP
	NOM_EVT_ADVIS_CHANGE_SCALE	XXX CHANGE SCALE
	NOM_EVT_SW_VER_UNK	CO2 UPDATE FW
	NOM_EVT_TUBE_DISCONN	CO2 NO TUBING
	NOM_EVT_TUBE_OCCL	CO2 OCCLUSION
	NOM_EVT_MSMT_RANGE_OVER	CO2 OVERRANGE
	NOM_EVT_TUBE_OBSTRUC	CO2 PURGING
	NOM_EVT_STAT_ZERO_RUNNING	CO2 AUTO ZERO

NOM_AWAY_CO2_ET	NOM_EVT_HI	XXX HIGH
	NOM_EVT_LO	XXX LOW
NOM_AWAY_CO2_INSP_MIN	NOM_EVT_HI	XXX HIGH
NOM_AWAY_RESP_RATE	NOM_EVT_APNEA	XXX APNEA
	NOM_EVT_LO	XXX LOW
	NOM_EVT_HI	XXX HIGH

AGM

NOM_DEV_ANALY_CONC_GAS_MULTI_PARAM_VMD	NOM_EVT_INCOMPAT*	GA INCOMPATIBLE
	NOM_EVT_MALF*	GA EQUIP MALFUNCT.
	NOM_EVT_STAT_STANDBY*	GA STANDBY
	NOM_EVT_STAT_DISCONN*	GA NOT AVAILABLE
	NOM_EVT_STAT_SELFTEST_RUNNING*	GA SELFTEST
	NOM_EVT_OBSTRUC*	GA OCCLUSION
	NOM_EVT_MSMT_INOP*	GA UNABLE TO MEAS
	NOM_EVT_MSMT_RANGE_OVER*	XXX OVERRANGE
	NOM_EVT_STAT_CALIB_RUNNING*	GA ZERO RUNNING
	NOM_EVT_WARMING*	GA WARMUP
	NOM_EVT_CALIB_FAIL*	GA ZERO FAILED
	NOM_EVT_MSMT_ERR*	GA ACCURACY?
	NOM_EVT_STAT_AL_OFF*	GA ALARMS SUPPRESS
	NOM_EVT_BREATH_ABSENT*	GA NO BREATH
NOM_AWAY_CO2	NOM_EVT_MSMT_INOP	XXX UNABLE TO MEAS
	NOM_EVT_DISTURB	XXX MEAS DISTURBED
	NOM_EVT_ADVIS_CHANGE_SCALE	XXX CHANGE SCALE
NOM_CONC_AWAY_O2	NOM_EVT_MALF	O2 EQUIP MALF
	NOM_EVT_CALIB_FAIL	O2 ZERO FAILED
	NOM_EVT_MSMT_INOP	XXX UNABLE TO MEAS
	NOM_EVT_DISTURB	XXX MEAS DISTURBED
	NOM_EVT_ADVIS_CHANGE_SCALE	XXX CHANGE SCALE
NOM_CONC_AWAY_N2O	NOM_EVT_MSMT_INOP	XXX UNABLE TO MEAS
	NOM_EVT_DISTURB	XXX MEAS DISTURBED
	NOM_EVT_ADVIS_CHANGE_SCALE	XXX CHANGE SCALE

NOM_CONC_AWAY_AGENT NOM_CONC_AWAY_DESFL NOM_CONC_AWAY_ENFL NOM_CONC_AWAY_HALOTH NOM_CONC_AWAY_SEVOFL NOM_CONC_AWAY_ISOFL	NOM_EVT_GAS_AGENT_IDENT_MALF	AGT ID MALFUNCTION
	NOM_EVT_CALIB_FAIL	AGT ID ZERO FAILED
	NOM_EVT_ADVIS_GAS_AGENT_CHK	CHECK AGENT
	NOM_EVT_MSMT_INOP	XXX UNABLE TO MEAS
	NOM_EVT_MSMT_RESTART	AGT MEAS RESTARTNG
	NOM_EVT_DISTURB	XXX MEAS DISTURBED
	NOM_EVT_CONTAM	GAS CONTAMINANT
	NOM_EVT_TOO_MANY_AGENTS	GA TOO MANY AGENTS
	NOM_EVT_ADVIS_CHANGE_SCALE	XXX CHANGE SCALE
	NOM_EVT_AGENT_MIX	AGENT MIXTURE
NOM_CONC_AWAY_N2	NOM_EVT_ADVIS_CHANGE_SCALE	XXX CHANGE SCALE
NOM_AWAY_CO2_ET	NOM_EVT_LO	XXX LOW
NOM_AWAY_RESP_RATE		
NOM_CONC_AWAY_O2_INSP		
NOM_CONC_AWAY_AGENT_ET		
NOM_CONC_AWAY_AGENT_INSP		
NOM_CONC_AWAY_HALOTH_ET		
NOM_CONC_AWAY_HALOTH_INSP		
NOM_CONC_AWAY_ENFL_ET		
NOM_CONC_AWAY_ENFL_INSP		
NOM_CONC_AWAY_ISOFL_ET		
NOM_CONC_AWAY_ISOFL_INSP		
NOM_CONC_AWAY_SEVOFL_ET		
NOM_CONC_AWAY_SEVOFL_INSP		
NOM_CONC_AWAY_DESFL_ET		
NOM_CONC_AWAY_DESFL_INSP		

NOM_AWAY_CO2_ET	NOM_EVT_HI	XXX HIGH
NOM_AWAY_CO2_INSP_MIN		
NOM_AWAY_RESP_RATE		
NOM_CONC_AWAY_O2_INSP		
NOM_CONC_AWAY_N2O_INSP		
NOM_CONC_AWAY_AGENT_ET		
NOM_CONC_AWAY_AGENT_INSP		
NOM_CONC_AWAY_HALOTH_ET		
NOM_CONC_AWAY_HALOTH_INSP		
NOM_CONC_AWAY_ENFL_ET		
NOM_CONC_AWAY_ENFL_INSP		
NOM_CONC_AWAY_ISOFL_ET		
NOM_CONC_AWAY_ISOFL_INSP		
NOM_CONC_AWAY_SEVOFL_ET		
NOM_CONC_AWAY_SEVOFL_INSP		
NOM_CONC_AWAY_DESFL_ET		
NOM_CONC_AWAY_DESFL_INSP		
NOM_AWAY_RESP_RATE	NOM_EVT_APNEA	XXX APNEA
NOM_CONC_AWAY_O2_INSP	NOM_EVT_O2_SUPPLY_LO	XXX inO2 LOW OXYGEN

System

NOM_OBJ_NETWORK	NOM_EVT_STAT_DISCONN*	Unsupported LAN
	NOM_EVT_MALF*	No Central Monit.
NOM_OBJ_QUICKLINK	NOM_EVT_IRREG*	Bad ServerLink
	NOM_EVT_UNSUPPORTED*	MeasSrv Unsupported
NOM_OBJ_SPEAKER	NOM_EVT_MALF*	Speaker Malfunct.
NOM_OBJ_INPUT_DEV		User I/F Malfunct.
NOM_OBJ_HIF_KEY		Check Keyboard
NOM_OBJ_HIF_MOUSE		.Check Mouse Device
NOM_OBJ_HIF_TOUCH		Check Touch Input
NOM_OBJ_HIF_SPEEDPOINT		Check Speedpoint
NOM_OBJ_HIF_ALARMBOX		Rem.AlarmDev.Malf
NOM_OBJ_QUICKLINK	NOM_EVT_ADVIS_PWR_HI*	MSL Power High
	NOM_EVT_ADVIS_PWR_OFF*	MSL Power Off
	NOM_EVT_ADVIS_PWR_OVER*	MSL Power Overload
NOM_OBJ_BUS_I2C	NOM_EVT_MALF*	Internal.Comm.Malf

NOM_MOC_VMS_MDS	NOM_EVT_VOLTAGE_OUT_OF_RANGE*	CheckInternVoltage
NOM_OBJ_QUICKLINK		Check MSL Voltage
NOM_MOC_VMS_MDS	NOM_EVT_TEMP_HI_GT_LIM*	Check Monitor Temp
NOM_OBJ_SETTING	NOM_EVT_MALF*	Check Settings
NOM_OBJ_CPU_SEC		Check Main Board 2
NOM_OBJ_LED		Check Alarm Lamps
NOM_OBJ_RELAY		Check Nurse Relay

Configuration

actual SourceId of the parameter.	NOM_EVT_STAT_DISCONN	XXX UNPLUGGED
	NOM_EVT_ADVIS_DEACT	XXX DEACTIVATED

NBP

NOM_PRESS_BLD_NONINV	NOM_EVT_CUFF_NOT_DEFLATED	CUFF NOT DEFLATED
	NOM_EVT_CUFF_INFLAT_OVER	NBP CUFF OVERPRESS
	NOM_EVT_EQUIP_MALF	XXX EQUIP MALF
	NOM_EVT_MSMT_INTERRUP	NBP INTERRUPTED
	NOM_EVT_MSMT_FAIL	NBP MEASURE FAILED
NOM_PRESS_BLD_NONINV_MEAN	NOM_EVT_HI	XXX HIGH
	NOM_EVT_LO	XXX LOW
NOM_PRESS_BLD_NONINV_SYS	NOM_EVT_HI	XXX HIGH
	NOM_EVT_LO	XXX LOW
NOM_PRESS_BLD_NONINV_DIA	NOM_EVT_HI	XXX HIGH
	NOM_EVT_LO	XXX LOW

TcGas

NOM_O2_TCUT	NOM_EVT_EQUIP_MALF	XXX EQUIP MALF
NOM_CO2_TCUT		
NOM_GAS_TCUT		
NOM_O2_TCUT	NOM_EVT_SENSOR_DISCONN	XXX NO TRANSDUC
NOM_CO2_TCUT		
NOM_GAS_TCUT		
NOM_O2_TCUT	NOM_EVT_STAT_CALIB_RUNNING	XXX CAL RUNNING
NOM_CO2_TCUT		
NOM_GAS_TCUT		
NOM_O2_TCUT	NOM_EVT_CALIB_FAIL	XXX CAL FAILED
NOM_CO2_TCUT		
NOM_GAS_TCUT		

NOM_O2_TCUT	NOM_EVT_ADVIS_CALIB_REQD	XXX CAL REQUIRD
NOM_CO2_TCUT		
NOM_GAS_TCUT		
NOM_O2_TCUT	NOM_EVT_ADVIS_CHANGE_SITE	XXX CHANGE SITE
NOM_CO2_TCUT		
NOM_GAS_TCUT		
NOM_O2_TCUT	NOM_EVT_STAT_SENSOR_WARMING	XXX STABILIZING
NOM_CO2_TCUT		
NOM_GAS_TCUT		
NOM_O2_TCUT	NOM_EVT_ADVIS_CHECK_SITE_TIME	XXX CHECK TIME
NOM_CO2_TCUT		
NOM_GAS_TCUT		
NOM_O2_TCUT	NOM_EVT_LO	XXX LOW
	NOM_EVT_HI	XXX HIGH
NOM_CO2_TCUT	NOM_EVT_LO	XXX LOW
	NOM_EVT_HI	XXX HIGH

VueLink

NOM_DEV_SYS_MULTI_MODAL_MDS	NOM_EVT_EQUIP_MALF	XXX EQUIP MALF
	NOM_EVT_CONFIG_ERR*	XXX NO CONFIG
	NOM_EVT_ADVIS_SETUP_CHK	XXX CHECK SETUP
	NOM_EVT_ADVIS_CONFIG_CHK	XXX CHK CONF.
	NOM_EVT_ADVIS_CABLE_CHECK	XXX CHK CABLE

VueLink-specific ID	NOM_EVT_EXT_DEV_AL_CODE_1*	VueLink specific
	NOM_EVT_EXT_DEV_AL_CODE_2*	
	NOM_EVT_EXT_DEV_AL_CODE_3*	
	NOM_EVT_EXT_DEV_AL_CODE_4*	
	NOM_EVT_EXT_DEV_AL_CODE_5*	
	NOM_EVT_EXT_DEV_AL_CODE_6*	
	NOM_EVT_EXT_DEV_AL_CODE_7*	
	NOM_EVT_EXT_DEV_AL_CODE_8*	
	NOM_EVT_EXT_DEV_AL_CODE_9*	
	NOM_EVT_EXT_DEV_AL_CODE_10*	
	NOM_EVT_EXT_DEV_AL_CODE_11*	
	NOM_EVT_EXT_DEV_AL_CODE_12*	
	NOM_EVT_EXT_DEV_AL_CODE_13*	
	NOM_EVT_EXT_DEV_AL_CODE_14*	
	NOM_EVT_EXT_DEV_AL_CODE_15*	
	NOM_EVT_EXT_DEV_AL_CODE_16*	
	NOM_EVT_EXT_DEV_AL_CODE_17*	
	NOM_EVT_EXT_DEV_AL_CODE_18*	
	NOM_EVT_EXT_DEV_AL_CODE_19*	
	NOM_EVT_EXT_DEV_AL_CODE_20*	
	NOM_EVT_EXT_DEV_AL_CODE_21*	
	NOM_EVT_EXT_DEV_AL_CODE_22*	
	NOM_EVT_EXT_DEV_AL_CODE_23*	
	NOM_EVT_EXT_DEV_AL_CODE_24*	
	NOM_EVT_EXT_DEV_AL_CODE_25*	
	NOM_EVT_EXT_DEV_AL_CODE_26*	
	NOM_EVT_EXT_DEV_AL_CODE_27*	
	NOM_EVT_EXT_DEV_AL_CODE_28*	
	NOM_EVT_EXT_DEV_AL_CODE_29*	
	NOM_EVT_EXT_DEV_AL_CODE_30*	
	NOM_EVT_EXT_DEV_AL_CODE_31*	
	NOM_EVT_EXT_DEV_AL_CODE_32*	
	NOM_EVT_EXT_DEV_AL_CODE_33*	

`VueLink-specific ID`	NOM_EVT_EXT_DEV_AL_CODE_1	VueLink specific
	NOM_EVT_EXT_DEV_AL_CODE_2	
	NOM_EVT_EXT_DEV_AL_CODE_3	
	NOM_EVT_EXT_DEV_AL_CODE_4	
	NOM_EVT_EXT_DEV_AL_CODE_5	
	NOM_EVT_EXT_DEV_AL_CODE_6	
	NOM_EVT_EXT_DEV_AL_CODE_7	
	NOM_EVT_EXT_DEV_AL_CODE_8	
	NOM_EVT_EXT_DEV_AL_CODE_9	
	NOM_EVT_EXT_DEV_AL_CODE_10	
	NOM_EVT_EXT_DEV_AL_CODE_11	
	NOM_EVT_EXT_DEV_AL_CODE_12	
	NOM_EVT_EXT_DEV_AL_CODE_13	
	NOM_EVT_EXT_DEV_AL_CODE_14	
	NOM_EVT_EXT_DEV_AL_CODE_15	
	NOM_EVT_EXT_DEV_AL_CODE_16	
	NOM_EVT_EXT_DEV_AL_CODE_17	
	NOM_EVT_EXT_DEV_AL_CODE_18	
	NOM_EVT_EXT_DEV_AL_CODE_19	
	NOM_EVT_EXT_DEV_AL_CODE_20	
	NOM_EVT_EXT_DEV_AL_CODE_21	
	NOM_EVT_EXT_DEV_AL_CODE_22	
	NOM_EVT_EXT_DEV_AL_CODE_23	
	NOM_EVT_EXT_DEV_AL_CODE_24	
	NOM_EVT_EXT_DEV_AL_CODE_25	
	NOM_EVT_EXT_DEV_AL_CODE_26	
	NOM_EVT_EXT_DEV_AL_CODE_27	
	NOM_EVT_EXT_DEV_AL_CODE_28	
	NOM_EVT_EXT_DEV_AL_CODE_29	
	NOM_EVT_EXT_DEV_AL_CODE_30	
	NOM_EVT_EXT_DEV_AL_CODE_31	
	NOM_EVT_EXT_DEV_AL_CODE_32	
	NOM_EVT_EXT_DEV_AL_CODE_33	

The following code from the SCADA partition are used for the alert source:

NOM_ECG_LEAD_LA	21
NOM_ECG_LEAD_RA	22
NOM_ECG_LEAD_LL	23
NOM_ECG_LEAD_C	66
NOM_ECG_LEAD_C1FR	82
NOM_ECG_LEAD_C2FR	83
NOM_ECG_LEAD_C3FR	84
NOM_ECG_LEAD_C4FR	85
NOM_ECG_LEAD_C5FR	87
NOM_ECG_LEAD_C6FR	88
NOM_ECG_LEAD_ES	100
NOM_ECG_LEAD_AS	101
NOM_ECG_LEAD_AI	102
NOM_ECG_LEAD_RL	115
NOM_ECG_ELEC_POTL	256
NOM_ECG_AMPL_ST	768
NOM_ECG_AMPL_ST_I	769
NOM_ECG_AMPL_ST_II	770
NOM_ECG_AMPL_ST_V1	771
NOM_ECG_AMPL_ST_V2	772
NOM_ECG_AMPL_ST_V3	773
NOM_ECG_AMPL_ST_V4	774
NOM_ECG_AMPL_ST_V5	775
NOM_ECG_AMPL_ST_V6	776
NOM_ECG_AMPL_ST_III	829
NOM_ECG_AMPL_ST_AVR	830
NOM_ECG_AMPL_ST_AVL	831
NOM_ECG_AMPL_ST_AVF	832
NOM_ECG_AMPL_ST_V	835
NOM_ECG_AMPL_ST_MCL	843
NOM_ECG_CARD_BEAT_RATE	16770
NOM_ECG_V_P_C_CNT	16993
NOM_PULS_RATE	18442
NOM_PLETH_PULS_RATE	18466
NOM_RES_VASC_SYS_INDEX	18688
NOM_PRESS_BLD	18944
NOM_PRESS_BLD_SYS	18945
NOM_PRESS_BLD_DIA	18946
NOM_PRESS_BLD_MEAN	18947
NOM_PRESS_BLD_NONINV	18948
NOM_PRESS_BLD_NONINV_SYS	18949
NOM_PRESS_BLD_NONINV_DIA	18950
NOM_PRESS_BLD_NONINV_MEAN	18951
NOM_PRESS_BLD_AORT	18956
NOM_PRESS_BLD_AORT_SYS	18957
NOM_PRESS_BLD_AORT_DIA	18958
NOM_PRESS_BLD_AORT_MEAN	18959
NOM_PRESS_BLD_ART	18960
NOM_PRESS_BLD_ART_SYS	18961
NOM_PRESS_BLD_ART_DIA	18962
NOM_PRESS_BLD_ART_MEAN	18963
NOM_PRESS_BLD_ART_ABP	18964
NOM_PRESS_BLD_ART_ABP_SYS	18965
NOM_PRESS_BLD_ART_ABP_DIA	18966
NOM_PRESS_BLD_ART_ABP_MEAN	18967

NOM_PRESS_BLD_ART_PULM	18972
NOM_PRESS_BLD_ART_PULM_SYS	18973
NOM_PRESS_BLD_ART_PULM_DIA	18974
NOM_PRESS_BLD_ART_PULM_MEAN	18975
NOM_PRESS_BLD_ART_UMB	18984
NOM_PRESS_BLD_ART_UMB_SYS	18985
NOM_PRESS_BLD_ART_UMB_DIA	18986
NOM_PRESS_BLD_ART_UMB_MEAN	18987
NOM_PRESS_BLD_ATR_LEFT	18992
NOM_PRESS_BLD_ATR_LEFT_SYS	18993
NOM_PRESS_BLD_ATR_LEFT_DIA	18994
NOM_PRESS_BLD_ATR_LEFT_MEAN	18995
NOM_PRESS_BLD_ATR_RIGHT	18996
NOM_PRESS_BLD_ATR_RIGHT_SYS	18997
NOM_PRESS_BLD_ATR_RIGHT_DIA	18998
NOM_PRESS_BLD_ATR_RIGHT_MEAN	18999
NOM_PRESS_BLD_VEN_CENT	19012
NOM_PRESS_BLD_VEN_CENT_SYS	19013
NOM_PRESS_BLD_VEN_CENT_DIA	19014
NOM_PRESS_BLD_VEN_CENT_MEAN	19015
NOM_PRESS_BLD_VEN_UMB	19016
NOM_PRESS_BLD_VEN_UMB_SYS	19017
NOM_PRESS_BLD_VEN_UMB_DIA	19018
NOM_PRESS_BLD_VEN_UMB_MEAN	19019
NOM_OUTPUT_CARD	19204
NOM_RES_VASC_SYS	19240
NOM_SAT_O2_VEN	19260
NOM_TEMP	19272
NOM_TEMP_ART	19280
NOM_TEMP_CORE	19296
NOM_TEMP_ESOPH	19300
NOM_TEMP_NASOPH	19308
NOM_TEMP_SKIN	19316
NOM_TEMP_VEN	19324
NOM_PULS_OXIM_SAT_O2	19384
NOM_PULS_OXIM_SAT_O2_DIFF	19396
NOM_PULS_OXIM_SAT_O2_ART_LEFT	19400
NOM_PULS_OXIM_SAT_O2_ART_RIGHT	19404
NOM_OUTPUT_CARD_CTS	19420
NOM_RESP	20480
NOM_RESP_RATE	20490
NOM_AWAY_RESP_RATE	20498
NOM_AWAY_CO2	20652
NOM_AWAY_CO2_ET	20656
NOM_AWAY_CO2_INSP_MIN	20666
NOM_CO2_TCUT	20684
NOM_O2_TCUT	20688
NOM_CONC_AWAY_O2	20836
NOM_CONC_AWAY_DESFL	20952
NOM_CONC_AWAY_ENFL	20956
NOM_CONC_AWAY_HALOTH	20960
NOM_CONC_AWAY_SEVOFL	20964
NOM_CONC_AWAY_ISOFL	20968
NOM_CONC_AWAY_N2O	20976
NOM_CONC_AWAY_DESFL_ET	21012
NOM_CONC_AWAY_ENFL_ET	21016

NOM_CONC_AWAY_HALOTH_ET	21020
NOM_CONC_AWAY_SEVOFL_ET	21024
NOM_CONC_AWAY_ISOFL_ET	21028
NOM_CONC_AWAY_DESFL_INSP	21096
NOM_CONC_AWAY_ENFL_INSP	21100
NOM_CONC_AWAY_HALOTH_INSP	21104
NOM_CONC_AWAY_SEVOFL_INSP	21108
NOM_CONC_AWAY_ISOFL_INSP	21112
NOM_CONC_AWAY_N2O_INSP	21120
NOM_CONC_AWAY_O2_INSP	21124
NOM_CONC_AWAY_N2	21372
NOM_CONC_AWAY_AGENT	21384
NOM_CONC_AWAY_AGENT_ET	21388
NOM_CONC_AWAY_AGENT_INSP	21392
NOM_PRESS_CEREB_PERF	22532
NOM_PRESS_INTRA_CRAN	22536
NOM_PRESS_INTRA_CRAN_SYS	22537
NOM_PRESS_INTRA_CRAN_DIA	22538
NOM_PRESS_INTRA_CRAN_MEAN	22539
NOM_EEG_ELEC_POTL_CRTX	22828
NOM_TEMP_RECT	57348
NOM_TEMP_BLD	57364
NOM_TEMP_DIFF	57368
NOM_OUTPUT_CARD_INDEX_CTS	61511
NOM_ELECTRODE_IMPED	61515
NOM_EEG_BIS_SIG_QUAL_INDEX	61517
NOM_EEG_BISPECTRAL_INDEX	61518
NOM_GAS_TCUT	61521
NOM_SAT_DIFF_O2_ART_VEN	61548
NOM_RATE_DIFF_CARD_BEAT_PULSE	61560

The following code from the object oriented partition are used for the alert source:

NOM_MOC_VMS_MDS	33
NOM_DEV_SYS_MULTI_MODAL_MDS	4493
NOM_DEV_ANALY_CONC_GAS_MULTI_PARAM_VMD	4114
NOM_DEV_ANALY_CARD_OUTPUT_VMD	4134
NOM_OBJ_HIF_KEY	61584
NOM_OBJ_SETTING	61649
NOM_OBJ_INPUT_DEV	61692
NOM_OBJ_NETWORK	61693
NOM_OBJ_QUICKLINK	61694
NOM_OBJ_SPEAKER	61695
NOM_DEV_ANALY_BISPECTRAL_INDEX_VMD	61806
NOM_OBJ_SENSOR	61902
NOM_OBJ_XDUCR	61903
NOM_OBJ_CHAN_1	61916
NOM_OBJ_CHAN_2	61917
NOM_OBJ_HIF_MOUSE	61983
NOM_OBJ_HIF_TOUCH	61984
NOM_OBJ_HIF_SPEEDPOINT	61985
NOM_OBJ_HIF_ALARMBOX	61986
NOM_OBJ_BUS_I2C	61987
NOM_OBJ_CPU_SEC	61988
NOM_OBJ_LED	61990
NOM_OBJ_RELAY	61991

The following codes from the event partition are used for the alert code:

NOM_EVT_ABSENT	4
NOM_EVT_CONTAM	14
NOM_EVT_DISCONN	22
NOM_EVT_DISTURB	24
NOM_EVT_ERRATIC	32
NOM_EVT_EXH	36
NOM_EVT_HI	40
NOM_EVT_IRREG	58
NOM_EVT_LO	62
NOM_EVT_MALF	70
NOM_EVT_NOISY	74
NOM_EVT_OBSTRUC	80
NOM_EVT_SUST	106
NOM_EVT_UNDEF	112
NOM_EVT_WARMING	124
NOM_EVT_BREATH_ABSENT	136
NOM_EVT_CALIB_FAIL	138
NOM_EVT_CONFIG_ERR	142
NOM_EVT_RANGE_ERR	164
NOM_EVT_CUFF_NOT_DEFLATED	230
NOM_EVT_CUFF_INFLAT_OVER	232
NOM_EVT_EQUIP_MALF	242
NOM_EVT_TUBE_OCCL	250
NOM_EVT_GAS_AGENT_IDENT_MALF	258
NOM_EVT_LEAD_DISCONN	268
NOM_EVT_LEADS_OFF	274
NOM_EVT_O2_SUPPLY_LO	296
NOM_EVT_OPTIC_MODULE_ABSENT	298
NOM_EVT_OPTIC_MODULE_DEFECT	300
NOM_EVT_SENSOR_DISCONN	308
NOM_EVT_SENSOR_MALF	310
NOM_EVT_SW_VER_UNK	322
NOM_EVT_TUBE_DISCONN	326
NOM_EVT_TUBE_OBSTRUC	330
NOM_EVT_XDUCR_DISCONN	336
NOM_EVT_XDUCR_MALF	338
NOM_EVT_INTENS_LIGHT_ERR	350
NOM_EVT_MSMT_DISCONN	352
NOM_EVT_MSMT_ERR	354
NOM_EVT_MSMT_FAIL	356
NOM_EVT_MSMT_INOP	358
NOM_EVT_MSMT_INTERRUP	362
NOM_EVT_MSMT_RANGE_OVER	364
NOM_EVT_MSMT_RANGE_UNDER	366
NOM_EVT_SIG_LO	380
NOM_EVT_SIG_UNANALYZEABLE	384
NOM_EVT_TEMP_HI_GT_LIM	394
NOM_EVT_UNSUPPORTED	400
NOM_EVT_WAVE_ARTIF_ERR	432
NOM_EVT_SIG_QUALITY	434
NOM_EVT_MSMT_INTERF_ERR	436
NOM_EVT_WAVE_OSCIL_ABSENT	442
NOM_EVT_VOLTAGE_OUT_OF_RANGE	460
NOM_EVT_INCOMPAT	600
NOM_EVT_APNEA	3072
NOM_EVT_ECG_ASYSTOLE	3076

NOM_EVT_ECG_BEAT_MISSED	3078
NOM_EVT_ECG_BIGEM	3082
NOM_EVT_ECG_BRADY_EXTREME	3086
NOM_EVT_ECG_PACING_NON_CAPT	3102
NOM_EVT_ECG_PAUSE	3108
NOM_EVT_ECG_TACHY_EXTREME	3122
NOM_EVT_ECG_CARD_BEAT_RATE_IRREG	3158
NOM_EVT_ECG_PACER_NOT_PACING	3182
NOM_EVT_ECG_SV_TACHY	3192
NOM_EVT_ECG_V_P_C_RonT	3206
NOM_EVT_ECG_V_P_C_MULTIFORM	3208
NOM_EVT_ECG_V_P_C_PAIR	3210
NOM_EVT_ECG_V_P_C_RUN	3212
NOM_EVT_ECG_V_RHY	3220
NOM_EVT_ECG_V_TACHY	3224
NOM_EVT_ECG_V_TACHY_NON_SUST	3226
NOM_EVT_ECG_V_TRIGEM	3236
NOM_EVT_DESAT	3246
NOM_EVT_ECG_V_P_C_RATE	3252
NOM_EVT_STAT_AL_OFF	6144
NOM_EVT_STAT_CALIB_MODE	6152
NOM_EVT_STAT_CALIB_RUNNING	6154
NOM_EVT_STAT_CALIB_INVIVO_RUNNING	6156
NOM_EVT_STAT_CALIB_LIGHT_RUNNING	6158
NOM_EVT_STAT_CALIB_PREINS_RUNNING	6160
NOM_EVT_STAT_SELFTEST_RUNNING	6164
NOM_EVT_STAT_ZERO_RUNNING	6170
NOM_EVT_STAT_OPT_MOD_SENSOR_CONN	6172
NOM_EVT_STAT_OPT_MOD_SENSOR_WARMING	6174
NOM_EVT_STAT_SENSOR_WARMING	6176
NOM_EVT_STAT_ECG_AL_ALL_OFF	6182
NOM_EVT_STAT_ECG_AL_SOME_OFF	6184
NOM_EVT_STAT_LEARN	6224
NOM_EVT_STAT_STANDBY	6228
NOM_EVT_STAT_DISCONN	6256
NOM_EVT_ADVIS_CHK	6658
NOM_EVT_ADVIS_CALIB_REQD	6662
NOM_EVT_ADVIS_CALIB_AND_ZERO_CHK	6664
NOM_EVT_ADVIS_CONFIG_CHK	6666
NOM_EVT_ADVIS_SETUP_CHK	6670
NOM_EVT_ADVIS_SRC_CHK	6672
NOM_EVT_ADVIS_CABLE_CHK	6680
NOM_EVT_ADVIS_GAS_AGENT_CHK	6688
NOM_EVT_ADVIS_SENSOR_CHK	6696
NOM_EVT_ADVIS_GAIN_DECR	6704
NOM_EVT_ADVIS_UNIT_CHK	6710
NOM_EVT_ECG_V_FIB_TACHY	61444
NOM_EVT_WAIT_CAL	61678
NOM_EVT_ADVIS_CHANGE_SITE	61682
NOM_EVT_ADVIS_CHECK_SITE_TIME	61684
NOM_EVT_STAT_FW_UPDATE_IN_PROGRESS	61688
NOM_EVT_EXT_DEV_AL_CODE_1	61690
NOM_EVT_EXT_DEV_AL_CODE_2	61692
NOM_EVT_EXT_DEV_AL_CODE_3	61694
NOM_EVT_EXT_DEV_AL_CODE_4	61696
NOM_EVT_EXT_DEV_AL_CODE_5	61698

NOM_EVT_EXT_DEV_AL_CODE_6	61700
NOM_EVT_EXT_DEV_AL_CODE_7	61702
NOM_EVT_EXT_DEV_AL_CODE_8	61704
NOM_EVT_EXT_DEV_AL_CODE_9	61706
NOM_EVT_EXT_DEV_AL_CODE_10	61708
NOM_EVT_EXT_DEV_AL_CODE_11	61710
NOM_EVT_EXT_DEV_AL_CODE_12	61712
NOM_EVT_EXT_DEV_AL_CODE_13	61714
NOM_EVT_EXT_DEV_AL_CODE_14	61716
NOM_EVT_EXT_DEV_AL_CODE_15	61718
NOM_EVT_EXT_DEV_AL_CODE_16	61720
NOM_EVT_EXT_DEV_AL_CODE_17	61722
NOM_EVT_EXT_DEV_AL_CODE_18	61724
NOM_EVT_EXT_DEV_AL_CODE_19	61726
NOM_EVT_EXT_DEV_AL_CODE_20	61728
NOM_EVT_EXT_DEV_AL_CODE_21	61730
NOM_EVT_EXT_DEV_AL_CODE_22	61732
NOM_EVT_EXT_DEV_AL_CODE_23	61734
NOM_EVT_EXT_DEV_AL_CODE_24	61736
NOM_EVT_EXT_DEV_AL_CODE_25	61738
NOM_EVT_EXT_DEV_AL_CODE_26	61740
NOM_EVT_EXT_DEV_AL_CODE_27	61742
NOM_EVT_EXT_DEV_AL_CODE_28	61744
NOM_EVT_EXT_DEV_AL_CODE_29	61746
NOM_EVT_EXT_DEV_AL_CODE_30	61748
NOM_EVT_EXT_DEV_AL_CODE_31	61750
NOM_EVT_EXT_DEV_AL_CODE_32	61752
NOM_EVT_EXT_DEV_AL_CODE_33	61754
NOM_EVT_ST_MULTI	61756
NOM_EVT_ADVIS_BSA_REQD	61760
NOM_EVT_ADVIS_PRESUMED_CVP	61762
NOM_EVT_BRADY	61766
NOM_EVT_TACHY	61768
NOM_EVT_ADVIS_CHANGE_SCALE	61770
NOM_EVT_MSMT_RESTART	61772
NOM_EVT_TOO_MANY_AGENTS	61774
NOM_EVT_STAT_PULSE_SRC_RANGE_OVER	61778
NOM_EVT_STAT_PRESS_SRC_RANGE_OVER	61780
NOM_EVT_MUSCLE_NOISE	61782
NOM_EVT_LINE_NOISE	61784
NOM_EVT_IMPED_HI	61786
NOM_EVT_AGENT_MIX	61788
NOM_EVT_IMPEDS_HI	61790
NOM_EVT_ADVIS_PWR_HI	61792
NOM_EVT_ADVIS_PWR_OFF	61794
NOM_EVT_ADVIS_PWR_OVER	61796
NOM_EVT_ADVIS_DEACT	61798

Private Unicode Characters

The IntelliVue monitor may use the following private codes for UNICODE characters:

```
#define SUBSCRIPT_CAPITAL_E_CHAR          0xE145
           /* SUBSCRIPT CAPITAL E                          */
#define SUBSCRIPT_CAPITAL_L_CHAR          0xE14C
           /* SUBSCRIPT CAPITAL L                          */
#define LITER_PER_CHAR                    0xE400
           /* LITER PER - used in 4 char unit "l/min"      */
#define HYDROGEN_CHAR                     0xE401
           /* HYDROGEN - Used in 4 char unit "cmH2O"       */
#define ALARM_STAR_CHAR                   0xE40D
           /* ALARM STAR                                   */
#define CAPITAL_V_WITH_DOT_ABOVE_CHAR     0xE425
           /* CAPITAL_V_WITH_DOT_ABOVE (V with dot)        */
#define ZERO_WIDTH_NO_BREAK_SPACE_CHAR    0xFEFF
           /* The character 0xFEFF is used as FILL character.
           For each wide asian character, a FILL character is
           appended for size calculations. */
```

List of Constants Used Within the Protocol Definition

RO Types

```
#define ROIV_APDU                1
#define RORS_APDU                2
#define ROER_APDU                3
#define ROLRS_APDU               5
```

ROLRS Identifier

```
#define RORLS_FIRST       1   /* set in the first message */
#define RORLS_NOT_FIRST_NOT_LAST 2
#define RORLS_LAST        3   /* last RORLSapdu, one RORSapdu
                                 to follow */
```

ROSE Commands

```
typedef u_16            CMDType;
#define CMD_EVENT_REPORT                  0
#define CMD_CONFIRMED_EVENT_REPORT        1
#define CMD_GET                           3
#define CMD_SET                           4
#define CMD_CONFIRMED_SET                 5
#define CMD_CONFIRMED_ACTION              7
```

ROER Error Values

```
#define NO_SUCH_OBJECT_CLASS            0
#define NO_SUCH_OBJECT_INSTANCE         1
#define ACCESS_DENIED                   2
#define GET_LIST_ERROR                  7
#define SET_LIST_ERROR                  8
#define NO_SUCH_ACTION                  9
#define PROCESSING_FAILURE              10
#define INVALID_ARGUMENT_VALUE          15
#define INVALID_SCOPE                   16
#define INVALID_OBJECT_INSTANCE         17
```

Action and Event Types

The Action and Event Types are defined in the Object Oriented Elements partition of the nomenclature.

```
#define NOM_NOTI_MDS_CREAT              3334
   /* MDS Create Notification */
#define NOM_NOTI_CONN_INDIC             3351
    /* connect indication event type */
#define NOM_ACT_POLL_MDIB_DATA          3094
    /* poll data action */
#define NOM_ACT_POLL_MDIB_DATA_EXT      61755
    /* extended poll data action */
```

Protocol Identification

The IDs for the protocol identification are from the Infrastructure nomenclature partition.

```
#define NOM_POLL_PROFILE_SUPPORT        1
    /* id for polling profile */
#define NOM_MDIB_OBJ_SUPPORT            258
    /* supported objects for the active profile */
#define NOM_ATTR_POLL_PROFILE_EXT       61441
    /* id for poll profile extensions opt. package */
```

Association Control

```
#define MDDL_VERSION1                       0x80000000
    /* Data Export Protocol Version */
#define NOMEN_VERSION                       0x40000000
    /* Nomenclature Version */
#define SYST_CLIENT                         0x80000000
    /* System Type Client */
#define SYST_SERVER                         0x00800000
    /* System Type Server */
#define HOT_START                           0x80000000
    /* Startup Mode Hotstart */
#define WARM_START                          0x40000000
    /* Startup Mode Warmstart */
#define COLD_START                          0x20000000
    /* Startup Mode Coldstart */
#define POLL_PROFILE_REV_0                  0x80000000
    /* Poll Profile Revision */
#define P_OPT_DYN_CREATE_OBJECTS            0x40000000
    /* option dynamic object creation */
#define P_OPT_DYN_DELETE_OBJECTS            0x20000000
    /* option dynamic object deletion */
#define POLL_EXT_PERIOD_NU_1SEC             0x80000000
    /* 1 sec Real-time Numerics */
#define POLL_EXT_PERIOD_NU_AVG_60SEC        0x20000000
    /* 1 min. averaged Numerics */
#define POLL_EXT_PERIOD_NU_AVG_300SEC       0x10000000
    /* 5 min. averaged Numerics */
#define POLL_EXT_PERIOD_RTSA                0x08000000
```

8

Building a Computer Client

Interfacing the LAN interface with UDP/IP

When setting up a Computer Client, a network traffic analyzing tool can be useful to verify the success of each step. Widely used tools are:

- Microsoft® Network Monitor
- Tcpdump (available under the GNU Public License from ftp://ftp.ee.lbl.gov/)

Setting Up the BootP Server

Step 1: Connect the Computer Client to the IntelliVue monitor.

The Computer Client and the IntelliVue monitor should be connected with a crossover LAN cable. If you need a dedicated system to run the BootP server, use a hub/switch to connect the devices. It is strongly recommended that a dedicated network is used for the data export. Do not connect any additional devices.

Step 2: Start the BootP server.

Please refer to the documentation of your BootP/DHCP server for installation guidelines. If you use a DHCP server, make sure that the server supports BootP clients.

Step 3: Verify that the IntelliVue monitor receives a valid IP address.

Use a network monitor to verify that the IntelliVue monitor receives the correct IP address. If the IntelliVue monitor shows an **`Unsupported LAN`** INOP, it has not received a valid IP address.

If the IntelliVue monitor does not receive an IP address,

- make sure that there is no IP address conflict on the network
- try to reboot the IntelliVue monitor.

Parsing the Connect Indication Message

Step1: Verify that the Connect Indication message is sent.

Use a networked monitor and check that the IntelliVue monitor sends a subnet broadcast message to the Connect Indication port (24005). If the IntelliVue monitor does not send the message, verify that the IntelliVue monitor received a valid IP address from the BootP server (see “Setting Up the BootP Server” on page 203).

Step 2: Receive the Connect Indication message on the Computer Client.

Open a socket on the Computer Client that receives the subnet broadcast message. If the Computer Client does not receive the Connect Indication message, verify the correct network connection, use an ICMP echo (ping) to check connectivity of the IntelliVue monitor.

Step 3: Parse the Data Export Protocol Command.

The Computer Client must parse the Connect Indication message to determine the port for the Data Export Protocol. The message also contains the IP address of the IntelliVue monitor.

The Computer Client should check that all length and type fields in the message are set correctly, otherwise the message must be discarded.

Then the Computer Client should parse the appended *AttributeList* and extract the IP address and port information (refer to “Connect Indication Attributes” on page 94 for the specification of these attributes).

Interfacing the MIB/RS232 Interface with the Fixed Baudrate Protocol

Step 1: Connect the Computer Client to the IntelliVue monitor.

It may be useful to try out the Association Request/Response mechanism on the LAN interface before working with the MIB/RS232 interface. This might help to find out whether an error is related to a ill-formatted Data Export message or if it is related to a transport layer problem.

Step 2: Implement the framing algorithm.

The section “The Fixed Baudrate Protocol, RS232 Port Settings” on page 20 contains some examples which can be used to check if your framing algorithm works correctly. Remember to apply the framing algorithm to both the *Hdr* and *User Data* part of the message.

If you have tried out the Association Request message on the LAN interface, you can try to send the message within the Fixed Baudrate protocol. Just add the *Hdr* information and apply the framing algorithm.

You should keep the following points in mind when implementing the the Fixed Baudrate protocol:

- Verify that the checksum algorithm works correctly for received messages, i.e., make sure that received messages with a corrupt checksum are discarded.
- Make sure that you implement an exception handling in case the received message grows larger than your receive buffer (e.g., if an end of frame character is lost somewhere during communication).
- The Fixed Baudrate Protocol is not connection oriented. After starting your application, there may be an existing Data Export Association (either from running your own application previously or from another system which has been connected to the MIB/RS232 Interface before). This may have some unexpected consequences for your application.

Interfacing the MIB/RS232 Interface with the AutoSpeed Protocol

Step 1: Connect the Computer Client and the IntelliVue monitor.

If your operating system comes with an IrDA stack, please refer to the documentation of your operating system. The operating system will cover most of the steps below automatically.

It may be useful to try out the Association Request/Response mechanism on the LAN interface before working with the MIB/RS232 interface. This may help you to find out whether an error is related to a wrongly-formatted Data Export message or if it is related to a transport layer problem.

Step 2: Establish an IrDA connection

The IrDA protocol supports a device detection procedure. If the detection is successful, it will return information about the detected device. This information contains a device nickname and a service hints field which indicates that the device supports the IEEE 1073 standard.

After this the Computer Client can establish an IrLAP connection with the device. This involves the negotiation of the baudrate and packet size for the lower layers. Refer to the Serial Infrared Link Access Protocol (IrLAP) specification (see page 22) for more information on this topic.

Step 3: Query the IAS database

The IAS database contains the object “IEEE:1073:3:2:MDDL” with the attribute “IrDA:TinyTP:LsapSel”. This attribute contains the number of the TinyTP Service Access point for the Data Export protocol. The value type of the attribute is an *integer*. The value should be equal to 1 if the MIB/RS232 Interface is used for Data Export.

The database also contains an object named “IEEE:1073:3:2” with the attribute “NodeType”. This attribute is of type integer and specifies the type of driver which resides on the interface. A value of 1 indicates that it is a data source, i.e. it is used to export data from the monitor.

After finishing the IAS query, the Computer Client should close the IAS connection before connecting to the TinyTP Service Access Point.

Step 4: Connect to the IEEE:1073:3:2:MDDL TinyTP Service Access Point

After connecting to the TinyTP Service Access Point, the connection can be used to send Association Control and Data Export Protocol messages within TinyTP data packets.

You should check the following points for your IrDA protocol stack:

- The connection may be interrupted or reset due to communication problems (e.g., if the cable is disconnected, or the monitor is rebooted). The Computer Client should be able to recover from such problems and initiate a new connection. Note: when a disconnect occurs on the IrDA protocol layer, an Association on the Data Export protocol layer will be terminated automatically.
- The Data Export protocol is packet oriented, this means that data is exchanged as a sequence of packets. Your IrDA stack may or may not provide a packet oriented interface to the TinyTP layer. The Data Export software requires that a received IrDA packet contains only one Data Export Protocol message.

Establishing an Association

Step 1: Send an Association Request message to the IntelliVue monitor.

Format an Association Request message as described in the section “Association Request Message” on page 57. Make sure that all length fields are set correctly, the right byte order is used, and the compiler does not insert extra bytes for structure alignment.

Step 2: Parse the Association Response message sent by the IntelliVue monitor.

Verify that the IntelliVue monitor sends an Association Response message.

If the IntelliVue monitor does not send a Response message, this can have the following reasons:

- The Association Request message has been sent to the wrong port.
- The IntelliVue monitor is connected to a central station or has been connected to one (reboot the IntelliVue monitor).
- The Association Request message was not formatted correctly.

If the IntelliVue monitor sends a Refuse message, this can have the following reasons:

- The Association Request message was not formatted correctly or requested a protocol that is not supported by the IntelliVue monitor.
- The IntelliVue monitor already has an association with a different Computer Client on the same interface.
- The IntelliVue monitor already has an association with a different Computer Client on another interface and the active association uses a different source for the numeric data (only one source for numeric data may be active at a time). Please refer to “Association Request Message” on page 57 for more information on the different sources for numeric data.

If the Computer Client has an association with the IntelliVue monitor and sends a second Association Request from the same source port, the message is discarded.

Look for the byte sequence described in “Association Response Message” on page 62 to find the beginning of the User Data. Parse the User Data and make sure that the IntelliVue monitor sets the protocol versions and options as expected. Check that the requested optional packages are present.

Step 3: Parse the MDS Create Event message.

The IntelliVue monitor will send the MDS Create Event message shortly after the Association Response message. The Computer Client should parse the message and extract all necessary information. Refer to the section “Wave Objects” on page 71 for a description of the available attributes.

Step 4: Send an MDS Create Result message.

The Computer Client must send an MDS Create Result message to confirm the MDS Create Event message. Refer to “MDS CREATE EVENT RESULT” on page 45 to see how the message is formatted.

Make sure that the message uses the correct presentation context ID.

It is important that the result message has the same invoke ID as the MDS Create Event message.

If the IntelliVue monitor receives a correct MDS Create Result message, it stops re-sending MDS Create Event messages. Use a network monitor to verify this.

Step 5: Send a Release Request message.

Use the building blocks from the section “RELEASE REQUEST” on page 225 to build a Release Request message and send it to the IntelliVue monitor.

The IntelliVue monitor identifies a Computer Client based on its IP address and the source port of the messages. The Computer Client must use the same source port as in the Association Request for all communication during the association. If a message is sent from another source port, it will be treated as a message from a different Computer Client.

Step 6: Parse the Release Response message

The IntelliVue monitor sends the Release Response message to confirm that the association has been terminated. For the Computer Client it is sufficient to check the session header of the response and verify that it is indeed a Release Response message (see “Release Response” on page 63).

If the Computer Client does not receive the response message, it should try to resend the Release Request message.

Accessing Data

Step 1: Establish an association as described above.

Step 2: Send a Poll Data Request message to the IntelliVue monitor.

Message Frequencies

If the Computer Client sends Protocol Messages with a high frequency, the IntelliVue monitor is not able to process all the requests. Some of the messages will be discarded. The Computer Client can detect discarded Poll Data Request messages by checking the poll number in the response. The Computer Client must set the poll number so that it will be able to detect loss of messages.

Single and Extended Polling

If the Computer Client needs to access real-time numeric or wave data, it should use Poll Profile Extensions (see “EXTENDED POLL DATA REQUEST” on page 49). This avoids sending poll requests with a high frequency and reduces the communication overhead.

The Computer Client can use an Extended Poll Request only to access Numerics, Waves and Alarms. It must use Single Poll Data Requests to access data from Patient Demographics or from the Medical Device System object.

Receive the Poll Data Response message and parse it.

The IntelliVue monitor sends a Single or Extended Poll Data Result message if the Poll Request message was parsed correctly.

Availability of Data

Not all of the data is available right after a new association has been established. The time span until all data is collected depends on the internal update frequency of the data. Typical times are listed in the table below.

Object Type	Max. Time
Numerics (real-time)	< 2 s
Numerics (12 second averaged)	< 18 s
Numerics (1 minute averaged)	< 70 s
Numerics (5 minute averaged)	< 310 s
Alarms	< 2 s
Patient Demographics	< 10 s
Medical Device System Object	< 1 s

During the startup phase, Poll Data Request messages on the object will result in Poll Data Response messages, which

- do not contain all the objects which are present in the IntelliVue monitor.
- do not contain all the available attributes of an object.

Numeric data is only available if a Measurement Server is connected to the IntelliVue monitor and if the system is not in stand-by mode. If a Measurement Server is connected to a running system, it may take several seconds until the data from the Measurement Server is available.

Parsing the Poll Result

The Poll Data Result message contains a checksum in the transport layer message. The Computer Client should verify that this checksum is correct. In the case of a corrupted checksum, the Computer Client must discard the message.

The Computer Client should check the poll number in the Poll Data Result message if it needs to detect lost messages. The Computer Client should check the *rel_time_stamp* which indicates the system time when the data was internally generated.

If the Computer Client needs to acquire a specific Numeric label (e.g., ABP), the preferred method is to use the *PhysioId* which is part of the Numeric Observed Value attribute (see “Numeric Objects” on page 65). The *physio_id* (physiological identifier) field contains a nomenclature code from the SCADA partition that identifies the represented value (typically a physiological measurement). It can be mapped to a label. However, for some numerics, the physio_id does not uniquely identify the measurement. E.g. all difference temperatures have the same physio_id, the numerics in the two channels of an EEG have the same physio_ids, the VueLink module may have numerics where the physio_id is not specified. However, if the label is derived by enumeration (e.g. the temperatures T1 and T2), the labels map to the same *PhysioId*. This ambiguity can be resolved if the user assigns other labels to the Numerics.

A Computer Client should not send Poll Requests for all attribute groups (polled_attr_grp = 0) when querying data with a high update frequency. Polling all attribute groups with a high frequency might lead to high system load and increased response latency. Future releases of the Data Export Protocol may support more attributes for each object.

If the IntelliVue monitor sends no response, check for the following causes:

- There is no association. Either the association was not established correctly or the IntelliVue monitor sent an Abort message (e.g., time-out) in the meantime.
- The Computer Client sent too many messages and messages were lost.
- The length of the transport layer message is corrupt.
- Length fields in the message are corrupt.

If the IntelliVue monitor sends a Remote Operation Error, this might have one of the following reasons:

- Wrong length field in the message.
- Wrong message type (*ro_type*, *command_type*, *action_type*).
- Wrong *managed_object* for the action (for Poll Requests, this must be the MDS object announced in the MDS Create Event).
- Wrong *polled_obj_type* (refer to “SINGLE POLL DATA REQUEST” on page 45 and “EXTENDED POLL DATA REQUEST” on page 49).
- Computer Client sent an Extended Poll Data Request, but the necessary optional package was not negotiated.
- Computer Client sent an Extended Poll Data Request with the wrong polled attribute group.
- Computer Client requested periodic Poll Data Result messages for too many objects. The Computer Client should at most send one request for Numerics (Metric Observed Value Attribute Group) and one for the AlertMonitor (Alert Monitor Attribute Group).

If the IntelliVue monitor sends a Poll Result message which does not contain all object/attributes check for the following problems:

- The Computer Client sent a Single Poll Data Request with the wrong polled attribute group. The Poll Result shows the objects with empty attribute lists (there are no attributes from the requested group).
- The association has been established and not all of the objects have been created. Wait until the objects are created.

Parsing AttributeLists

When parsing an *AttributeList*, the Computer Client should adhere to the following guidelines:

- Verify that the length fields in the *AttributeList* are consistent with other length fields in the message.
- Check both the count and length field of the *AttributeList* to detect the end of the list.
- Do not rely on the sequence of attributes in an *AttributeList*.
- Skip unknown attributes.
- Verify that the length field of each *AVAType* is consistent with its value.

If the Computer Client fails parsing the message, it is useful to compare the raw message (captured with a network monitor) with the Computer Client’s interpretation of the data. Common problems are:

- The Computer Client uses a different byte order. Wrong interpretation of length and count fields in particular can lead to problems.

- The Computer Client uses a different alignment for structures. The offset for members of a structure will be wrong, because the compiler for the Computer Client inserted bytes for alignment.
- Length fields denote the length of data appended, excluding the size of the length field.

Interpreting Data from Numerics

- Do not rely on the sequence of values within a Compound Numeric Observed Value attribute. The physiological identifiers must be interpreted.
- A triple valued pressure parameter can change to single valued (mean only), whenever the diastolic and systolic values are close together. This commonly happens when a pressure is being zeroed or when a transducer is left exposed to air. The parameter is still sent as a Compound Numeric Observed Value, even if only one value is available.
- The text in the label strings is localized. If you have a monitor with chinese localization, the strings will contain chinese UNICODE characters.

Interpreting Data from the Alert Monitor

- If the Computer Client wants to display Alarm messages, it should check the strings for UNICODE characters from the private use area (see “Definitions Shared by Protocols” on page 25).
- The text in the alarm strings is localized. If you have a monitor with chinese localization, the strings will contain chinese UNICODE characters.

Interpreting Wave Data

- The IntelliVue patient monitor supports the following wave types, which are defined by sample period, sample and array size (Sample Array Specification), and update period (Metric Specification) in the static context.

Wave Type	Sample Period	Sample Size	Array Size	Update Period	Bandwidth Requirement[1]
500 samples/s (ECG)	2 ms	16 bits	128 samples	256 ms	1064 bytes/s
250 samples/s (Compound ECG)	4 ms	16 bits	3*64 samples	256 ms	1640 bytes/s
125 samples/s	8 ms	16 bits	32 samples	256 ms	296 bytes/s
62.5 samples/s	16 ms	16 bits	16 samples	256 ms	168 bytes/s

1. Observed values, not including context data.

- The Computer Client can poll the dynamic context to determine the available waves. Because of the high amount of data, the client should specify the required wave objects before requesting wave observed values in a periodic data poll.
- Up to three ECG waves (500 samples/s) can be polled simultaneously by selecting the appropriate lead labels in the Wave object priority list. The object handle is the same for all ECG waves. Waves can be identified by their physiological identifier.

- In non-EASI mode, three ECG waves (250 samples/s, including the primary and secondary lead) can be polled by selecting the NLS_NOM_ECG_ELEC_POTL label in the Wave object priority list. The monitor sends poll results with a compound wave, containing three waves with common context. Waves can be identified by their physiological identifier.
- Up to eight non-ECG waves (125 or 62.5 samples/s) can be polled simultaneously by selecting the appropriate labels in the Wave object priority list.
- The Computer Client needs to keep track of the poll results time stamps to detect missing wave samples.
- Entries in the Wave object priority list are ignored if the label does not exist or the object is not available, or more than three ECG and/or more than eight non-ECG waves are specified.
- The wave context can be polled separately or multiplexed with the wave observed values. If the *polled_attr_grp* is *0* in a periodic data poll request, the monitor reports one object's static and dynamic context per 1024 ms. Context attributes are included in the observation poll.

9

Troubleshooting

This chapter will help you identify and locate faults that may occur when using the Protocol. The procedure to locate faults uses a troubleshooting matrix.

When the fault has been identified, check the Possible Causes and corresponding Corrective Actions. Perform the corrective actions. Re-check the fault after each corrective action is performed until the fault has been cleared.It is assumed that you have a functioning Computer Client.

Fault	Possible Causes	Corrective Actions
Computer Client doesn't receive LAN messages	Cable connection is broken or wrong cable used.	• Verify that the IntelliVue monitor is correctly connected to the network. • Verify that the Computer Client is correctly connected to the network. Try to use an ICMP echo (ping) to check the monitor and Computer Client connections.
	IntelliVue monitor failure	Re-boot the IntelliVue monitor and try to make a new connection. Refer to the Troubleshooting section in the *Service Guide* of your device.
IntelliVue monitor shows an **Unsupported LAN** INOP	BootP server does not send a valid IP address.	Check the configuration of the BootP server. Check that the BootP server is correctly connected to the network.
	Cable connection is broken or wrong cable used.	Check the connection between the IntelliVue monitor and the BootP Server.
IntelliVue monitor shows a **No Central Monitoring** INOP	**Central Monitoring Mandatory** is configured to **On** in the monitor	Data Export must not be used with a central station. Configure **Central Monitoring** to **Optional**.
	Central Monitoring Mandatory is configured to **On** in the monitor and the connection to the central station is interrupted	Data Export must not be used with a central station. Reboot the IntelliVue monitor and make sure it is not connected to a central station.

Fault	Possible Causes	Corrective Actions
Computer Client doesn't receive messages with the AutoSpeed protocol	Cable connection is broken or wrong cable used.	Check the connection between the IntelliVue monitor and the Computer Client.
	Wrong configuration of MIB/RS232 Interface	Check if the MIB/RS232 interface is configured for the desired protocol
	IntelliVue monitor failure	Re-boot the IntelliVue monitor and try to make a new connection. Disconnect the MIB/RS232 cable for more than 60s, this will most likely reset the IrDA stack of the client system too. Refer to the Troubleshooting section in the *Service Guide* of your device.
Computer Client does not establish an association.	Another Computer Client Application is already associated with the IntelliVue monitor.	Make sure no other Computer Client Application is trying to connect to the IntelliVue monitor. Reboot the IntelliVue monitor or wait until the association is timed out.
Computer Client does not report data.	Parameter is switched off.	If the Computer Client requires a specific measurement, the parameter must be switched on in the IntelliVue monitor.
	Wave label is not included in the Wave object priority list.	Specify the wave objects to be polled in the Set Priority List Request
Wave Samples are missing in a perioidic data poll	Too many Wave objects polled.	Reduce the number of entries in the Wave object priority list.

10

Protocol Examples

Data Export Protocol Examples

CONNECT INDICATION EVENT

The Connect Indication message contains the *ConnectIndInfo* which is of variable length. The length fields in the message depend on the length of the *ConnectIndInfo*.This message is only available on the LAN interface.

```
Nomenclature          u_32                 : 1.0
                          {0x00 0x00 0x01 0x00}
ROapdus               ro_type              : ROIV_APDU
                      length               : <xx>
                          {0x00 0x01 0xXX 0xXX}
ROIVapdu              invoke_id            : 0
                      command_type         : CMD_EVENT_REPORT
                      length               : <xx>
                          {0x00 0x00 0x00 0x00 0xXX 0xXX}
EventReportArg.
ManagedObjectId       m_obj_class       :NOM_MOC_VMS_MDS_COMPOS_SINGLE_BED
                      context_id           : 0
                      handle               : 0
RelativeTime          event_time           : 39424
OIDType               event_type           : NOM_NOTI_MDS_CONNECT_INDIC
u_16                  length               : <xx>
                          {0x00 0x23 0x00 0x00 0x00 0x00 0x00 0x00
                           0x9A 0x00 0x0D 0x17 0xXX 0xXX}
ConnectIndInfo        [...]
```

MDS CREATE EVENT

The MDS Create Event message contains an *AttributeList* which is of variable length. The length fields in the message depend on the length of the *AttributeList*.

```
SPpdu                 session_id           : 0xE100
                      p_context_id         : 2
                          {0xE1 0x00 0x00 0x02}
ROapdus               ro_type               : ROIV_APDU
                      length               : <xx>
                          {0x00 0x01 0xXX 0xXX}
ROIVapdu              invoke_id            : 1
                      command_type         : CMD_CONFIRMED_EVENT_REPORT
                      length               : <xx>
                          {0x00 0x01 0x00 0x01 0xXX 0xXX}
EventReportArg.
ManagedObjectId       m_obj_class          : NOM_MOC_VMS_MDS
                      context_id           : 0
                      handle               : 0
RelativeTime          event_time           : 126976
OIDType               event_type           : NOM_NOTI_MDS_CREAT
u_16                  length               : <xx>
                          {0x00 0x21 0x00 0x00 0x00 0x00 0x00 0x01
                           0xf0 0x00 0x0d0x06 0xXX 0xXX}
MDSCreateInfo
ManagedObjectId       m_obj_class          : NOM_MOC_VMS_MDS
```

```
                    context_id          : 0
                    handle              : 0
                        {0x00 0x21 0x00 0x00 0x00 0x00}
AttributeList       [...]
```

MDS CREATE EVENT RESULT

```
SPpdu               session_id          : 0xE100
                    p_context_id        : 2
                        {0xE1 0x00 0x00 0x02}
ROapdus             ro_type             : RORS_APDU
                    length              : 20
                        {0x00 0x02 0x00 0x14}
RORSapdu            invoke_id           : 1
                    command_type        : CMD_CONFIRMED_EVENT_REPORT
                    length              : 14
                        {0x00 0x01 0x00 0x01 0x00 0x0e}
EventReportRes.
ManagedObjectId     m_obj_class         : NOM_MOC_VMS_MDS
                    context_id          : 0
                    handle              : 0
RelativeTime        event_time          : 4736768
OIDType             event_type          : NOM_NOTI_MDS_CREAT
u_16                length              : 0
                       {0x00 0x21 0x00 0x00 0x00 0x00 0x00 0x48
                        0x47 0x00 0x0d 0x06 0x00 0x00}
```

SINGLE POLL DATA REQUEST

```
SPpdu               session_id          : 0xE100
                    p_context_id        : 2
                       {0xE1 0x00 0x00 0x02}
ROapdus             ro_type             : ROIV_APDU
                    length              : 28
                       {0x00 0x01 0x00 0x1c}
ROIVapdu            invoke_id           : 0
                    command_type        : CMD_CONFIRMED_ACTION
                    length              : 22
                       {0x00 0x01 0x00 0x07 0x00 0x16}
ActionArgument
ManagedObjectId     m_obj_class         : NOM_MOC_VMS_MDS
                    context_id          : 0
                    handle              : 0
u_32                scope               : 0
OIDType             action_type         : NOM_ACT_POLL_MDIB_DATA
u_16                length              : 8
                       {0x00 0x21 0x00 0x00 0x00 0x00 0x00 0x00
                        0x00 0x00 0x0c 0x16 0x00 0x08}
PollMdibDataReq
u_16                poll_number         : 1
TYPE                partition           : NOM_PART_OBJ
                    code                : NOM_MOC_VMO_METRIC_NU
OIDType             polled_attr_grp     : all attribute groups
                       {0x00 0x01 0x00 0x01 0x00 0x06 0x00 0x00}
```

SINGLE POLL DATA RESULT

The Single Poll Data Result message contains a *PollInfoList* which is of variable length. The length fields in the message depend on the length of the *PollInfoList*.

```
SPpdu               session_id          : 0xE100
                    p_context_id        : 2
                       {0xE1 0x00 0x00 0x02}
ROapdus             ro_type             : RORS_APDU
                    length              : <xx>
                       {0x00 0x02 0xXX 0xXX}
RORSapdu            invoke_id           : 0
                    command_type        : CMD_CONFIRMED_ACTION
                    length              : <xx>
                       {0x00 0x00 0x00 0x07 0xXX 0xXX}
ActionResult
ManagedObjectId     m_obj_class         : NOM_MOC_VMS_MDS
                    context_id          : 0
                    handle              : 0
OIDType             action_type         : NOM_ACT_POLL_MDIB_DATA
u_16                length              : <xx>
```

```
                    {0x00 0x21 0x00 0x00 0x00 0x00 0x0c 0x16
                     0xXX 0xXX}
PollMdibDataReply
u_16              poll_number           : 1
RelativeTime      rel_time_stamp        : 4766464
AbsoluteTime      abs_time_stamp        : 0xffffffff 0xffffffff
TYPE              partition             : NOM_PART_OBJ
                  code                  : NOM_MOC_VMO_METRIC_NU
OIDType           polled_attr_grp       : all attribute groups
                    {0x00 0x01 0x00 0x48 0xbb 0x00 0xff 0xff
                     0xff 0xff 0xff 0xff 0xff 0xff 0x00 0x01
                     0x00 0x06 0x00 0x00}
PollInfoList      [...]
```

SINGLE POLL DATA RESULT (LINKED)

It is assumed that the IntelliVue monitor needs two messages to encode all the data from a Poll Request.

The first message would have a linked result header:

```
SPpdu             session_id            : 0xE100
                  p_context_id          : 2
                    {0xE1 0x00 0x00 0x02}
ROapdus           ro_type               : ROLRS_APDU
                  length                : <xx>
                    {0x00 0x05 0xXX 0xXX}
ROLRSapdu
RorlsId           state                 : RORLS_FIRST
                  count                 : 1
u_16              invoke_id             : 0
CMDType           command_type          : CMD_CONFIRMED_ACTION
u_16              length                : <xx>
                    {0x01 0x01 0x00 0x00 0x00 0x07 0xXX 0xXX}
ActionResult[...]
```

The second message would contain the rest of the data:

```
SPpdu             session_id            : 0xE100
                  p_context_id          : 2
                    {0xE1 0x00 0x00 0x02}
ROapdus           ro_type               : ROLRS_APDU
                  length                : <xx>
                    {0x00 0x05 0xXX 0xXX}
ROLRSapdu
RorlsId           state                 : RORLS_LAST
                  count                 : 2
u_16              invoke_id             : 0
CMDType           command_type          : CMD_CONFIRMED_ACTION
u_16              length                : <xx>
                    {0x03 0x02 0x00 0x00 0x00 0x07 0xXX 0xXX}
ActionResult[...]
```

Finally, the monitor sends a Remote Operation Result message:

```
SPpdu             session_id            : 0xE100
                  p_context_id          : 2
                    {0xE1 0x00 0x00 0x02}
ROapdus           ro_type               : RORS_APDU
                  length                :<xx>
                    {0x00 0x02 0xXX 0xXX}
RORSapdu          invoke_id             : 0
                  command_type          : CMD_CONFIRMED_ACTION
                  length                : <xx>
                    {0x00 0x00 0x00 0x07 0xXX 0xXX}
ActionResult      [...]
```

Note that all messages contain a fully encoded *ActionResult* data structure. The last Remote Operation Result message, however, would contain a *PollInfoList* structure with the *count* and *length* field set to 0. A client system should not depend on the terminating Remote Operation Result to have an empty *PollInfoList*. The message should be parsed as any other message.

EXTENDED POLL DATA REQUEST

The next example shows a message which could be used to access averaged data. The message will only be accepted if the optional package for Poll Profile Extensions has been negotiated during the association phase.

```
SPpdu             session_id            : 0xE100
                  p_context_id          : 2
                     {0xE1 0x00 0x00 0x02}
ROapdus           ro_type               : ROIV_APDU
                  length                : 32
                     {0x00 0x01 0x00 0x20}
ROIVapdu          invoke_id             : 0
                  command_type          : CMD_CONFIRMED_ACTION
                  length                : 26
                     {0x00 0x01 0x00 0x07 0x00 0x1a}
ActionArgument
ManagedObjectId   m_obj_class           : NOM_MOC_VMS_MDS
                  context_id            : 0
                  handle                : 0
u_32              scope                 : 0
OIDType           action_type           : NOM_ACT_POLL_MDIB_DATA_EXT
u_16              length                : 12
                     {0x00 0x21 0x00 0x00 0x00 0x00 0x00 0x00
                      0x00 0x00 0xf1 0x3b 0x00 0x0c}
PollMdibDataReqExt
u_16              poll_number           : 1
TYPE              partition             : NOM_PART_OBJ
                  code                  : NOM_MOC_VMO_METRIC_NU
OIDType           polled_attr_grp       : all attribute groups
AttributeList
u_16              count                 : 0
u_16              length                : 0
                     {0x00 0x01 0x00 0x01 0x00 0x06 0x00 0x00
                      0x00 0x00 0x00 0x00}
```

EXTENDED POLL DATA RESULT

The Extended Poll Data Result message contains an additional *sequence_no*, which is used if the client requests periodic replies.

```
SPpdu             session_id            : 0xE100
                  p_context_id          : 2
                     {0xE1 0x00 0x00 0x02}
ROapdus           ro_type               : RORS_APDU
                  length                : <xx>
                     {0x00 0x02 0xXX 0xXX}
RORSapdu          invoke_id             : 0
                  command_type          : CMD_CONFIRMED_ACTION
                  length                : <xx>
                     {0x00 0x00 0x00 0x07 0xXX 0xXX}
ActionResult
ManagedObjectId   m_obj_class           : NOM_MOC_VMS_MDS
                  context_id            : 0
                  handle                : 0
OIDType           action_type           : NOM_ACT_POLL_MDIB_DATA_EXT
u_16              length                : <xx>
                     {0x00 0x21 0x00 0x00 0x00 0x00 0xf1 0x3b
                      0xXX 0xXX}
PollMdibDataReplyExt
u_16              poll_number           : 1
u_16              sequence_no           : 0
RelativeTime      rel_time_stamp        : 4766464
AbsoluteTime      abs_time_stamp        : 0xffffffff 0xffffffff
TYPE              partition             : NOM_PART_OBJ
                  code                  : NOM_MOC_VMO_METRIC_NU
OIDType           polled_attr_grp       : all attribute groups
                     {0x00 0x01 0x00 0x00 0x00 0x48 0xbb 0x00
                      0xff 0xff 0xff 0xff 0xff 0xff 0xff 0xff
                      0x00 0x01 0x00 0x06 0x00 0x00}
PollInfoList      [...]
```

GET PRIORITY LIST REQUEST

```
SPpdu             session_id            : 0xE100
```

```
                        p_context_id          : 2
                {0xE1 0x00 0x00 0x02}
ROapdus                 ro_type               : ROIV_APDU
                        length                : 22
                {0x00 0x01 0x00 0x16}
ROIVapdu                invoke_id             : 0
                        command_type          : CMD_GET
                        length                : 16
                {0x00 0x00 0x00 0x03 0x00 0x10}
GetArgument
ManagedObjectId         m_obj_class           : NOM_MOC_VMS_MDS
                        context_id            : 0
                        handle                : 0
u_32                    scope                 : 0
AttributeIdList         count                 : 1
                        length                : 2
                        OIDType               : NOM_ATTR_POLL_RTSA_PRIO_LIST
                {0x00 0x21 0x00 0x00 0x00 0x00 0x00 0x00
                 0x00 0x00 0x00 0x01 0x00 0x02 0xF2 0x3A}
```

GET PRIORITY LIST RESULT

```
SPpdu                   session_id            : 0xE100
                        p_context_id          : 2
                {0xE1 0x00 0x00 0x02}
ROapdus                 ro_type               : RORS_APDU
                        length                : <xx>
                {0x00 0x02 0xXX 0xXX}
RORSapdu                invoke_id             : 0
                        command_type          : CMD_GET
                        length                : <xx>
                {0x00 0x00 0x00 0x03 0xXX 0xXX}
GetResult
ManagedObjectId         m_obj_class           : NOM_MOC_VMS_MDS
                        context_id            : 0
                        handle                : 0
                {0x00 0x21 0x00 0x00 0x00 0x00}
AttributeList           count                 : 1
                        length                : <xx>
AvaType                 attribute_id          : NOM_ATTR_POLL_RTSA_PRIO_LIST
                        length                : <xx>
                {0x00 0x01 0xXX 0xXX 0xF2 0x3A 0xXX 0xXX}
TextIdList              [...]
```

SET PRIORITY LIST REQUEST

```
SPpdu                   session_id            : 0xE100
                        p_context_id          : 2
                {0xE1 0x00 0x00 0x02}
ROapdus                 ro_type               : ROIV_APDU
                        length                : <xx>
                {0x00 0x01 0xXX 0xXX}
ROIVapdu                invoke_id             : 0
                        command_type          : CMD_CONFIRMED_SET
                        length                : <xx>
                {0x00 0x00 0x00 0x05 0xXX 0xXX}
SetArgument
ManagedObjectId         m_obj_class           : NOM_MOC_VMS_MDS
                        context_id            : 0
                        handle                : 0
u_32                    scope                 : 0
                {0x00 0x21 0x00 0x00 0x00 0x00 0x00 0x00
                 0x00 0x00}
ModificationList        count                 : 1
```

```
                        length                  : <xx>
AttributeModEntry       modifyOperator          : REPLACE
AvaType                 attribute_id            : NOM_ATTR_POLL_RTSA_PRIO_LIST
                        length                  : <xx>
                    {0x00 0x01 0xXX 0xXX 0x00 0x00 0xF2 0x3A
                     0xXX 0xXX}
TextIdList              [...]
```

SET PRIORITY LIST RESULT

```
SPpdu                   session_id              : 0xE100
                        p_context_id            : 2
                    {0xE1 0x00 0x00 0x02}
ROapdus                 ro_type                 : RORS_APDU
                        length                  : <xx>
                    {0x00 0x02 0xXX 0xXX}
RORSapdu                invoke_id               : 0
                        command_type            : CMD_CONFIRMED_SET
                        length                  : <xx>
                    {0x00 0x00 0x00 0x05 0xXX 0xXX}
SetResult
ManagedObjectId         m_obj_class             : NOM_MOC_VMS_MDS
                        context_id              : 0
                        handle                  : 0
                    {0x00 0x21 0x00 0x00 0x00 0x00}
AttributeList           count                   : 1
                        length                  : <xx>
AvaType                 attribute_id            : NOM_ATTR_POLL_RTSA_PRIO_LIST
                        length                  : <xx>
                    {0x00 0x01 0xXX 0xXX 0xF2 0x3A 0xXX 0xXX}
TextIdList              [...]
```

AttributeList

This example shows an AttributeList which contains attributes from the Alert Monitor.

```
AttributeList    count                  : 5
                 length                 : 248
                    {0x00 0x05 0x00 0xf8}
AVAType          attribute_id           : NOM_ATTR_ID_HANDLE
                 length                 : 2
                 attribute_val          : 0x835d
                    {0x09 0x21 0x00 0x02 0x83 0x5d}
AVAType          attribute_id           : NOM_ATTR_ID_TYPE
                 length                 : 4
                 attribute_val          : 0x0001 0x0036
                    {0x09 0x2f 0x00 0x04 0x00 0x01 0x00 0x36}
AVAType          attribute_id           : NOM_ATTR_DEV_AL_COND
                 length                 : 10
                 attribute_val          : 0x1000 0x091a 0x0000 0x0002
                                            0x0000
                    {0x09 0x16 0x00 0x0a 0x10 0x00 0x09 0x1a
                     0x00 0x00 0x00 0x02 0x00 0x00}
AVAType          attribute_id           : NOM_ATTR_AL_MON_P_AL_LIST
                 length                 : 4
                 attribute_val          : 0x0000 0x0000
                    {0x09 0x02 0x00 0x04 0x00 0x00 0x00 0x00}
AVAType          attribute_id           : NOM_ATTR_AL_MON_T_AL_LIST
                 length                 : 208
attribute_val    : [...]
                    {0x09 0x04 0x00 0xd0 0x00 0x03 0x00 0xcc
                     0x4b 0xb8 0x01 0xba 0x00 0x02 0x10 0x00
                     0x00 0x02 0x00 0x00 0x83 0x3a 0x02 0x04
                     0x00 0x32 0x00 0x01 0x80 0x15 0x04 0x02
                     0x00 0x07 0x78 0x00 0x00 0x26 0x00 0x53
                     0x00 0x70 0x00 0x4f 0x20 0x82 0x00 0x20
                     0x00 0x4e 0x00 0x4f 0x00 0x4e 0x00 0x2d
                     0x00 0x50 0x00 0x55 0x00 0x4c 0x00 0x53
                     0x00 0x41 0x00 0x54 0x00 0x49 0x00 0x4c
                     0x00 0x45 0x00 0x00 0x50 0x00 0x01 0x12
                     0x00 0x02 0x10 0x00 0x00 0x09 0x00 0x00
                     0x02 0x91 0x02 0x04 0x00 0x32 0x00 0x01
```

```
0x00 0x03 0x01 0x0c 0x00 0x00 0x78 0x00
0x00 0x26 0x00 0x52 0x00 0x65 0x00 0x73
0x00 0x70 0x00 0x20 0x00 0x20 0x00 0x20
0x00 0x4c 0x00 0x45 0x00 0x41 0x00 0x44
0x00 0x53 0x00 0x20 0x00 0x4f 0x00 0x46
0x00 0x46 0x00 0x20 0x00 0x20 0x00 0x00
0x4a 0x04 0x00 0xf2 0x00 0x02 0x10 0x00
0x00 0x02 0x00 0x00 0x82 0x63 0x02 0x04
0x00 0x32 0x00 0x01 0x00 0x03 0x00 0x46
0x00 0x02 0x78 0x00 0x00 0x26 0x00 0x4e
0x00 0x42 0x00 0x50 0x00 0x20 0x00 0x20
0x00 0x20 0x00 0x20 0x00 0x45 0x00 0x51
0x00 0x55 0x00 0x49 0x00 0x50 0x00 0x20
0x00 0x4d 0x00 0x41 0x00 0x4c 0x00 0x46
0x00 0x20 0x00 0x00}
```

Association Control Protocol Examples

ASSOCIATION REQUEST

The following building blocks can be used to format an Association Request message:

AssocReqSessionHeader

```
0x0D <LI>
```

AssocReqSessionData

```
0x05 0x08 0x13 0x01 0x00 0x16 0x01 0x02
0x80 0x00 0x14 0x02 0x00 0x02
```

AssocReqPresentationHeader

```
0xC1 <LI> 0x31 0x80 0xA0 0x80 0x80 0x01
0x01 0x00 0x00 0xA2 0x80 0xA0 0x03 0x00
0x00 0x01 0xA4 0x80 0x30 0x80 0x02 0x01
0x01 0x06 0x04 0x52 0x01 0x00 0x01 0x30
0x80 0x06 0x02 0x51 0x01 0x00 0x00 0x00
0x00 0x30 0x80 0x02 0x01 0x02 0x06 0x0C
0x2A 0x86 0x48 0xCE 0x14 0x02 0x01 0x00
0x00 0x00 0x01 0x01 0x30 0x80 0x06 0x0C
0x2A 0x86 0x48 0xCE 0x14 0x02 0x01 0x00
0x00 0x00 0x02 0x01 0x00 0x00 0x00 0x00
0x00 0x00 0x61 0x80 0x30 0x80 0x02 0x01
0x01 0xA0 0x80 0x60 0x80 0xA1 0x80 0x06
0x0C 0x2A 0x86 0x48 0xCE 0x14 0x02 0x01
0x00 0x00 0x00 0x03 0x01 0x00 0x00 0xBE
0x80 0x28 0x80 0x06 0x0C 0x2A 0x86 0x48
0xCE 0x14 0x02 0x01 0x00 0x00 0x00 0x01
0x01 0x02 0x01 0x02 0x81
```

AssocReqUserData

The *AssocReqUserData* contains variable data, see see “Protocol Commands” on page 6-55.

AssocReqPresentationTrailer

```
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
```

ASSOCIATION RESPONSE

The following building blocks can be used to format an Association Response message:

AssocRespSessionHeader

```
0x0E <LI>
```

AssocRespSessionData

```
0x05 0x08 0x13 0x01 0x00 0x16 0x01 0x02
0x80 0x00 0x14 0x02 0x00 0x02
```

AssocRespPresentationHeader

```
0xC1 <LI> 0x31 0x80 0xA0 0x80 0x80 0x01
0x01 0x00 0x00 0xA2 0x80 0xA0 0x03 0x00
0x00 0x01 0xA5 0x80 0x30 0x80 0x80 0x01
0x00 0x81 0x02 0x51 0x01 0x00 0x00 0x30
0x80 0x80 0x01 0x00 0x81 0x0C 0x2A 0x86
0x48 0xCE 0x14 0x02 0x01 0x00 0x00 0x00
0x02 0x01 0x00 0x00 0x00 0x00 0x61 0x80
0x30 0x80 0x02 0x01 0x01 0xA0 0x80 0x61
0x80 0xA1 0x80 0x06 0x0C 0x2A 0x86 0x48
0xCE 0x14 0x02 0x01 0x00 0x00 0x00 0x03
0x01 0x00 0x00 0xA2 0x03 0x02 0x01 0x00
0xA3 0x05 0xA1 0x03 0x02 0x01 0x00 0xBE
0x80 0x28 0x80 0x02 0x01 0x02 0x81
```

AssocRespUserData

The *AssocRespUserData* contains variable data, see “Protocol Commands” on page 55.

AssocRespPresentationTrailer

```
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
```

REFUSE

The following building blocks can be used to format a Refuse message:

RefuseSessionHeader

```
0x0C 0x03
```

RefuseSessionData

```
0x32 0x01 0x00
```

RefusePresentationHeader

This block is empty in the Refuse message.

RefuseUserData

This block is empty in the Refuse message.

RefusePresentationTrailer

This block is empty in the Refuse message.

RELEASE REQUEST

The following building blocks can be used to format a Release Request message:

ReleaseReqSessionHeader

```
0x09 0x18
```

ReleaseReqSessionData

This block is empty in the Release Request message.

ReleaseReqPresentationHeader

```
0xC1 0x16 0x61 0x80 0x30 0x80 0x02 0x01
0x01 0xA0 0x80 0x62 0x80 0x80 0x01 0x00
0x00 0x00 0x00 0x00
```

ReleaseReqUserData

This block is empty in the Release Request message.

ReleaseReqPresentationTrailer

```
0x00 0x00 0x00 0x00
```

RELEASE RESPONSE

The following building blocks can be used to format a Release Response message:

ReleaseRespSessionHeader

```
0x0A 0x18
```

ReleaseRespSessionData

This block is empty in the Release Response message.

ReleaseRespPresentationHeader

```
0xC1 0x16 0x61 0x80 0x30 0x80 0x02 0x01
0x01 0xA0 0x80 0x63 0x80 0x80 0x01 0x00
0x00 0x00 0x00 0x00
```

ReleaseRespUserData

This block is empty in the Release Response message.

ReleaseRespPresentationTrailer

```
0x00 0x00 0x00 0x00
```

ASSOCIATION ABORT

The following building blocks can be used to format a Association Abort message:

AbortSessionHeader

```
0x19 0x2E
```

AbortSessionData

```
0x11 0x01 0x03
```

AbortPresentationHeader

```
0xC1 0x29 0xA0 0x80 0xA0 0x80 0x30 0x80
0x02 0x01 0x01 0x06 0x02 0x51 0x01 0x00
0x00 0x00 0x00 0x61 0x80 0x30 0x80 0x02
0x01 0x01 0xA0 0x80 0x64 0x80 0x80 0x01
0x01 0x00 0x00 0x00 0x00 0x00 0x00
```

AbortUserData

This block is empty in the Abort message.

AbortPresentationTrailer

```
0x00 0x00 0x00 0x00
```

User Data

The following section contains an example for the User Data which is contained in an Association Request message.

```
UserData
ASNLength         length                : 72
                     {0x48}
MDSEUserInfoStd
ProtocolVersion   protocol_version      : MDDL_VERSION1
NomenclatureVers.nomenclature_version   : NOMEN_VERSION
FunctionalUnits   functional_units      : 0
SystemType        system_type           : SYST_CLIENT
StartupMode        startup_mode         : COLD_START
                     {0x80 0x00 0x00 0x00 0x40 0x00 0x00 0x00
                      0x00 0x00 0x00 0x00 0x80 0x00 0x00 0x00
                      0x20 0x00 0x00 0x00}
Option List
AttributeList     count                 : 0
                  length                : 0
                     {0x00 0x00 0x00 0x00}
Supported Profiles
AttributeList     count                 : 1
                  length                : 44
                     {0x00 0x01 0x00 0x2c}
AVAType
OIDType           attribute_id          : NOM_POLL_PROFILE_SUPPORT
u_16              length                : 40
                     {0x00 0x01 0x00 0x28}
PollProfileSupport (attribute_val)
PollProfileRev.   poll_profile_revision : POLL_PROFILE_REV_0
RelativeTime      min_poll_period       : 800000
u_32               max_mtu_rx           : 1000
u_32               max_mtu_tx           : 1000
u_32              max_bw_tx             : 0xffff 0xffff
PollProfileOpt.   options               : 0x6000 0x0000
                     {0x80 0x00 0x00 0x00 0x00 0x00 0x09 0xc4
                      0x00 0x00 0x09 0xc4 0x00 0x00 0x03 0xe8
                      0xff 0xff 0xff 0xff 0x60 0x00 0x00 0x00}
Optional Packages
AttributeList     count                 : 1
                  length                : 12
                     {0x00 0x01 0x00 0x0c}
AVAType
OIDType           attribute_id          : NOM_ATTR_POLL_PROFILE_EXT
u_16              length                : 8
                      {0xf0 0x01 0x00 0x08}
PollProfileExt (attribute_val)
PollProfileExtOpt.options               : POLL_EXT_PERIOD_NU_AVG_60SEC
AttributeList     count                 : 0
                  length                : 0
                      {0x20 0x00 0x00 0x00 0x00 0x00 0x00 0x00}
```

With this User Data, the length field of the Presentation Header must be set to 220 (0xDC) and the length field of the Session Header must be set to 236 (0xEC).