BM800 Communication protocol Specification

Bl-1013-02
elektronisk kopia

Contents

CONTENTS **2**

1. SUMMARY **5**

1.1. FUTURE CHANGES 5

2. INTRODUCTION, AND USE OF XML IN BM800 ..... **6**

3. SAMPLE DATA FORMAT **7**

3.1. SAMPLE DATA BASIC STRUCTURE 7
3.2. COMMON PARAMETER FORMAT 8
3.3. <VER> - DATA FORMAT VERSION 8
3.4. <INSTRINFO> - INSTRUMENT INFORMATION 8
3.4.1. Static Instrument Parameters *9*
3.4.1.1. PRDI - Product Identifier 9
3.4.1.2. SNO - Serial Number 9
3.4.1.3. BRND – Instrument Brand 9
3.4.2. Instrument Configuration Parameters *9*
3.4.2.1. FIWV - Firmware Version 9
3.4.2.2. IAPL – Instrument Application 9
3.4.2.3. IID - User-Settable Instrument ID 9
3.5. <SMPINFO> SAMPLE INFORMATION PARAMETER VALUES 9
3.5.1. Sample Identification Parameters *10*
3.5.1.1. ID – Blood Sample ID 10
3.5.1.2. SEQ – Sample Run Sequence Number 11
3.5.1.3. DATE – Sample Run Date and Time 11
3.5.1.4. SORC – Sample or Control 11
3.5.1.5. STYP – Sample Type 12
3.5.2. Aspiration Parameters *12*
3.5.2.1. ASPM - Aspiration Mode 12
3.5.2.2. ASPS – Aspiration Status 12
3.5.2.3. BLMD - Background Mode Sample Run 12
3.5.2.4. BLNK - Blank Sample 12
3.5.2.5. ASWN - Autosampler Wheel Number 13
3.5.2.6. ASWP - Autosampler Wheel Position 13
3.5.3. Differentiation – Red Cells *13*
3.5.3.1. RPD - RBC/PLT Discriminator position 13
3.5.3.2. RPDS - RBC/PLT Floating Discriminator Status 14
3.5.3.3. RPDL - RBC/PLT Floating Discriminator Low Range Limit 14
3.5.3.4. RPDH - RBC/PLT Floating Discriminator High Range Limit 14
3.5.3.5. RPDF - RBC/PLT Fixed Fallback Discriminator 14
3.5.4. Differentiation – White Cells *14*
3.5.4.1. WDMS - WBC Diff Method Set 14
3.5.4.2. WDMA - WBC Diff Method Actually Used 15
3.5.4.3. WDFB - WBC Diff Method Fallback enabled 15
3.5.4.4. WDLL - WBC Floating Discriminator Low Range Limit (LYM-L) 15
3.5.4.5. WDLH - WBC Floating Discriminator High Range Limit (GRAN-H) 15
3.5.4.6. WDCL - WBC Calculated Floating Discriminator Low (LYM/MID) 16
3.5.4.7. WDCH - WBC Calculated Floating Discriminator High (MID/GRAN) 16
3.5.4.8. WLGL – WBC DM4 LYM/GRAN Limit 16
3.5.4.9. WDIL - WBC Fixed Discriminator Low (MID-L) 16
3.5.4.10. WDIH - WBC Fixed Discriminator High (MID-H) 17
3.5.4.11. WDDM - WBC Debris Discriminator Mode 17
3.5.4.12. WDDP - WBC Debris Discriminator Position 17
3.5.5. Control Blood with Reference Ranges *18*
3.5.5.1. CAPL – Control Blood Application 18
3.5.5.2. CLVL – Control Blood Level 18

3.5.5.3. CEXP - Expiry Date of a Control Blood with Reference Ranges ... 18
3.5.5.4. CEXT - Extra Control Blood Information for a Control Blood with Reference Ranges ... 18
3.5.6. Patient Information ... 19
3.5.7. Container Status ... 19
3.5.7.1. RGED - Diluent Container Empty ... 19
3.5.7.2. RGEL - Hemolyzer Container Empty ... 19
3.5.7.3. RGEC - Cleaner Container Empty ... 19
3.5.8. Settings ... 19
3.5.8.1. APNU – Analysis Profile Number ... 19
3.5.8.2. APNA – Analysis Profile Name / control blood level name ... 20
3.5.8.3. XLT - Extra Hemolyzing Time ... 20
3.5.9. Reagent statistics ... 21
3.5.9.1. RDLI Reagent Statistics Diluent Lot identifier ... 21
3.5.9.2. RDPN Reagent Statistics Diluent Package number ... 21
3.5.9.3. RDED Reagent Statistics Diluent Expiry date ... 21
3.5.9.4. RLLI Reagent Statistics Hemolyzer Lot identifier ... 21
3.5.9.5. RLPN Reagent Statistics Hemolyzer Package number ... 21
3.5.9.6. RLED Reagent Statistics Hemolyzer Expiry date ... 22
3.5.9.7. RCLI Reagent Statistics Cleaner Lot identifier ... 22
3.5.9.8. RCPN Reagent Statistics Cleaner Package number ... 22
3.5.9.9. RCED Reagent Statistics Cleaner Expiry date ... 22
3.6. <SMPRESULTS> SAMPLE RESULT PARAMETER VALUES ... 22
3.6.1. Result Parameter Format ... 23
3.6.1.1. <v> - Parameter Value ... 23
3.6.1.2. <r> - Parameter Out-of-Range ... 23
3.6.1.3. <f> - Parameter Error Flag ... 23
3.6.1.4. <l> - Parameter Normal/Reference Range Low ... 23
3.6.1.5. <h> - Parameter Normal/Reference Range High ... 24
3.6.1.6. Examples ... 24
3.6.2. Description of the Sample Result Parameters ... 24
3.7. <HGRAMS> HISTOGRAMS ... 25
3.7.1. <hgrams> - Histograms Section ... 27
3.7.2. <hgram> - Histogram Format ... 27
3.7.2.1. <n> – Histogram Name ... 27
3.7.2.2. <min> - Histogram Min Range ... 27
3.7.2.3. <m> - Histogram Max Range ... 27
3.7.2.4. <k> - Histogram Number of Bins ... 27
3.7.2.5. <w> - Histogram Filter Factor ... 28
3.7.2.6. <d> - Histogram Discriminator Index ... 28
3.7.3. <hgdata> - Histogram Data Vector ... 28
3.7.3.1. <n> - Histogram Data Vector Name ... 28
3.7.3.2. <v> - Histogram Data Vector Values ... 28
4. TRANSPORT PACKAGE ... 29
4.1. PACKAGE DELIMITERS ... 29
4.2. CHECKSUM ARMOUR ... 29
4.2.1. New Line handling ... 30
4.2.2. Checksum algorithm ... 30
4.2.2.1. “Easy calculation” checksum algorithm ... 30
4.3. MESSAGE HANDLING ... 32
4.3.1. Message ID assignment ... 32
4.3.2. Acknowledge messages ... 33
4.3.3. Retransmission of lost messages ... 34
4.3.4. Handling of duplicate message ID:s ... 34
4.3.5. Handling of missing message ID:s ... 34
4.3.6. Handling of incomplete packages ... 34
4.3.7. Handling of packages that fails the checksum test, or has transport package or message format errors 34
4.3.8. Handling of messages with undefined message content ... 34
4.3.9. Some comments about the message handling in BM800 ... 35
4.3.10. Final tips and tricks ... 35

5. REFERENCES 35

1. Summary

This specification describes both the BM800 XML formatted sample result format (see 3) and a rather generic full duplex transport package protocol which is formatted inside XML comments (see 4).

BM800 uses an eXtensible Markup Language (XML) based format for all in- and outgoing computer communications.

The normal BM800 communication consists of a BM800 transmitting sample results, and a receiving end storing and/or processing the sample results. The receiving end can also optionally acknowledge each sample results package.

The results of one sample are transmitted in one transport package.

The BM800 can either be set up to automatically transmit sample results every time one sample has been processed, or the operator can initiate a transmission of one or more sample results from the sample memory in the BM800. It the latter case each sample results is still transmitted in one transport package.

It is not possible to determine if one set of sample results comes from an automatic transmission, or from a send from the sample memory. They looks exactly the same.

The receiving end should be prepared to handle duplicate transmissions of the same sample. (Do no confuse this with the transport package retransmissions described in 4.3.3. Such retransmitted sample results are dropped on the transport package level.)

BM800 instruments equipped with special firmware can also receive sample results formatted according to this specification.

Future versions of the BM800 firmware may well include more kinds of in- and outgoing communications.

1.1. Future changes

This version of the specification describes the situation when it is released. There will probably be future versions of both this specification and of the BM800 firmware communications.

This specification states in some places that something “may change” or “may be added”. This is done when the author of the specification clearly foresees future changes that has a relatively good chance of being implemented. But other things could change as well.

These are some clearly foreseeable future changes:

- The number of characters in ID numbers (see 3.5.1.1).
- The number of analysis profiles (see 3.5.8.1)
- New kinds of data content, like “all settings”. These will have their own top-level node names.

And there are also some clearly foreseeable problems:

- Analysis profile types (species in vet instruments) do not have reserved names or numbers. Qualified operators can set the names and reprogram profiles. And the names might be translated to other languages as well. Applications receiving data from a BM800 may therefore require a specific analysis profile (species) setup to correctly identify different blood sample types.
- New features will most certainly be added to the BM800. New features may require changes and / or additions to future versions of the specification.
- BM800 outputs log texts, error texts and such in some cases. A receiving end connected to a BM800 must handle “garbage” outside transport packages. Or preferably log them somewhere. (Log texts and such will never end up inside a transport package. Unless the BM800 restarts in the middle om a transmission. Then that transmission will be cut short.)

2. Introduction, and use of XML in BM800

This is unfortunately a mix (mess?) of formal specification, implementation details about BM800 and XML, and rationale for some decisions.

Some knowledge of XML, character encoding and BM800 is assumed. And some C language programming too.

BM800 only uses the syntax and structure from XML. This gives it a lot of existing XML parsers to use instead of having to write custom tailored ones. The format needs to be as simple as possible, since the BM800 has to parse it as well. Producing XML, on the other hand, is easy for fixed data structures. For further description of XML, see [1].

Document Type Declarations (DTD:s) are not used, but could be added. The BM800 communication format is a very application-specific use of XML. But a DTD should preferably actually be possible to write and use.

The XML output from the BM800 is strictly speaking not correct XML. An XML document should have one top-level document node. BM800 outputs multiple sample result nodes after each other on the top level. But each sample result node in itself is a correct XML document.

BM800 does not use attributes. They might not be visible by a simple XML text viewer, and complicates parsing in the BM800.

BM800 will use the ISO 10646 ~ "Unicode" character set for non-7-bit-ASCII characters. These characters will be encoded in "UTF-8" or as numeric character entities. (Note: “UTF-8” is the default encoding in XML unless one is specified.)

BM800 does not use XML declarations '<?xml version="..." ... ?> ' or text declarations "<?xml ... ?>". They are not required since only "UTF-8" is used (or its 7-bit ASCII subset), which is the XML default character encoding.

BM800 only uses the predefined entities "&" (&), "<" (<), ">" (>), "'" (') and "&quote;" ('). ("'" and "&quote;" are only needed in attribute values, which are not used in this specification.)

The tag name syntax is limited to be like C identifiers:

```
[A-Za-z_][A-Za-z_0-9]*
```

(=An upper- or lowercase letter or an underscore followed by zero or more upper- or lowercase letters, underscores or digits)

Note: all tag names beginning with "xml" in any combination of upper- or lowercase letters are reserved for use by XML standards.

Case is significant in tag names and in parameter names.

BM800 uses single-character names for common element tag names to speed up serial transfers. Examples: "<n>" for name, "<v>" for value etc. But note that all element tag names in an XML document should be unique if it should be possible to add a "tight" DTD that separates PCDATA content and element content. (It can be argued that PCDATA elements could be reused in different contexts. F.ex. "<n>" could mean name in one context, and number of data points in another. This is mostly an issue for human readers, since the XML standard does not interpret PCDATA.)

The acronym PCDATA means Parsed Character Data and is for the purpose of this specification element free, free formatted text.

The BM800 XML formatted data is free-format. It is allowed to insert white space at any position allowed by XML standards. At least one white space character is required in some character data to separate data elements. White space characters is of course not allowed inside individual character data elements. Example. `"<X>3.14</X>"`, not `"<X>3 . 14</X>"`. But `"<X> 3.14 </X>"` is ok. (And increases the transfer time.)

3. Sample Data Format

A sample result is the actual output produced by the BM800. A sample result may differ in the number of parameters, the presence of histograms etc.

BM800 defines a lot of parameter names below. They should be unique within the sample definition. BM800 limits parameter names to at most 4 characters. This saves space in the BM800, and time during serial transfer. The syntax restriction of a parameter name is the same as for a C identifier.

BM800 has a simple XML structure, and use parameters for flexibility instead. This might minimize changes to the XML structure. External applications should be prepare to handle the addition and removal of parameters. And also be prepared for values outside of the ranges defined below.

The basic idea is that different BM800 instruments are allowed to dump different parameter lists, but that one particular instrument should dump the same parameter list for every sample. In the latter case, unused parameters should be output with a name but without a value. (This can make it easier to go through dump texts. At least for humans, or very simple text handling programs.)

BM800 splits information in many small, easily parsed pieces instead of combining more than one value as one parameter. This makes the XML text bigger, but simplifies external applications.

The exception is: "<v>" (list of values) for histograms and raw HGB data.

Note about storing samples in databases: there exist no 100% correct primary key among these parameter values. But a combination of "SEQ" and "DATE" (in the case of one instrument), or "SNO" (or "IID" maybe), "SEQ" and "DATE" (in the case of multiple instruments) is probably good enough. (Note: 3.4.1 gives PRDI and SNO as an instrument primary key, but this is probably overkill for a sample primary key.)

3.1. Sample Data Basic Structure

Overall structure.

```
<!--:Begin:Chksum:...-->
<!--:Begin:Msg:...-->
<sample>
  <ver>
    <!-- Version number of XML sample data format -->
  </ver>
  <instrinfo>
    <!-- Information about the instrument. -->
  </instrinfo>
  <smpinfo>
    <!—Sample Information parameter values -->
  </smpinfo>
  <smpresults>
    <!—Sample Result parameter values -->
  </smpresults>
  <tparams>
    <!-- Technical (troubleshooting) parameter values (optional) -->
  </tparams>
  <hgrams>
    <!-- Histograms (optional) -->
  </hgrams>
  <rawdata>
    <!-- Raw data (optional) -->
  </rawdata>
  <scatter>
    <!-- Scatter data (optional) -->
  </scatter>
</sample>
<!--:End:Msg:...-->
<!--:End:Chksum:...-->
```

Note: The ":Begin:...: " and ":End:...: " XML comments are present during serial transfer for error detection, flow control and retransmissions. See section 4 for further info.

3.2. Common Parameter Format

All parameters have the following common structure:

```
<p><n>NAME</n><!-- value and/or other stuff --></p>
```

The parameter name should follow C identifier syntax (see above). By convention, sample result parameter names use only upper case.

The "<n>" (name) tag must be the first one.

The "<v>" tag is the parameter value. It is present if a valid value of the parameter exists.

Parameters that could be present should be present with their names even if they have no value. Parameters are not present if:

1) The parameter is not included in the BM800 model, or
2) The parameter is blocked by the user.

Examples:

A parameter without any value etc.

```
<p><n>APNU</n></p>
```

A parameter with a value

```
<p><n>APNU</n><v>2</v></p>
```

The common parameter format is extended for result parameters. See section 3.6.1.

3.3. <ver> - Data Format Version

The "<ver>" tag identifies the version of the BM800 XML format used. It is not correlated to any version numbering of this specification. The used version denotation is *{major.minor}*, example "1.1". It is supposed that the major number is changed when incompatible changes have been implemented. The minor number is changed for backward-compatible changes like adding parameters and tags.

The "<ver>" tag is first to make it possible to change the interpretation of the rest of the sample dump depending on this version number.

Implementation note: This is an issue for "SAX type" parsers that handles one piece of the XML document at a time, but it is less of an issue for "DOM type" parser that builds up and returns a tree of nodes. But note that a DTD cannot depend on PCDATA content.

The current data format version is "1.1".

3.4. <instrinfo> - Instrument Information

The instrument information parameters are formatted according to the common parameter format described in section 3.2. The structure of the instrument information parameters is as follows:

```
<instrinfo>
  <p><n>PRDI</n><v> ... </v></p>
  <p><n>FIWV</n><v> ... </v></p>
  <p><n>SNO</n><v> ... </v></p>
  <p><n>SWI</n><v> ... </v></p>
  <p><n>BRND</n><v> ... </v></p>
  <p><n>IAPL</n><v> ... </v></p>
  <p><n>IID</n><v> ... </v></p>
</instrinfo>
```

3.4.1. Static Instrument Parameters

The static instrument parameters are parameters that are always the same during the instrument life cycle. The combination of the PRDI and SNO parameters can be used as a unique identifier (primary key) of an individual instrument.

3.4.1.1. PRDI - Product Identifier

PRDI identifies the instrument as for example a BM800. This parameter is always present with a value. Its format is not further defined here. (Other instruments could in theory use this specification.)

```
<p><n>PRDI</n><v>BM800</v></p>
```

3.4.1.2. SNO - Serial Number

SNO is the serial number of the instrument. This parameter is always present with a value. This number is supposed to be unique for a certain instrument chassis.

Value range: 1 – 65534.

```
<p><n>SNO</n><v>10001</v></p>
```

3.4.1.3. BRND – Instrument Brand

BRND indicates the marketing brand of the instrument. This parameter is always present with a value.

Value range: M - for Medonic, S - for Swelab. Other OEM brands may be added. All brand codes will be one uppercase letter.

```
<p><n>BRND</n><v>M</v></p>
```

3.4.2. Instrument Configuration Parameters

The instrument configuration parameters are "semi-static" instrument parameters. The parameters are probably the same during the instrument life cycle, but they may change, for example after a software upgrade.

3.4.2.1. FIWV - Firmware Version

FIWV identifies the software used in the instrument. This parameter is always present with a value. Its format is not further defined here.

```
<p><n>FIWV</n><v>2.1.3</v></p>
```

3.4.2.2. IAPL – Instrument Application

IAPL describes the application configuration of the instrument, i.e. if it is a human, veterinary or blood bank instrument. This parameter is always present with a value.

Value range: H – Human, V- Veterinary, B – Blood Bank. Other instrument applications may be added. All instrument application codes will be one uppercase letter.

```
<p><n>IAPL</n><v>H</v></p>
```

3.4.2.3. IID - User-Settable Instrument ID

IID identifies an individual BM800 instrument. It is a user-settable string of 1-15 characters. . This parameter is always present, but the value is only present if the instrument ID is set.

Note: be prepared for non-ASCII characters.

```
<p><n>IID</n><v>AA123</v></p>
```

3.5. <smpinfo> Sample Information Parameter Values

Sample information parameter values identify the sample run.

The parameter order below is not fixed. The parameters might be in another order. Be prepared for added and removed parameters.

```
<smpinfo>
  <p><n>ID</n>...</p>
  <p><n>SEQ</n>...</p>
  <p><n>DATE</n>...</p>
  <p><n>APNU</n>...</p>
  <p><n>APNA</n>...</p>
  <p><n>ASPM</n>...</p>
  <p><n>ASPS</n>...</p>
  <p><n>SORC</n>...</p>
  <p><n>BLMD</n>...</p>
  <p><n>BLNK</n>...</p>
  <p><n>STYP</n>...</p>
  <p><n>RGED</n>...</p>
  <p><n>RGEL</n>...</p>
  <p><n>RGEC</n>...</p><!-- Optional, may be present in vet instruments -->
  <p><n>RDLI</n>...</p>
  <p><n>RDPN</n>...</p>
  <p><n>RDED</n>...</p>
  <p><n>RLLI</n>...</p>
  <p><n>RLPN</n>...</p>
  <p><n>RLED</n>...</p>
  <p><n>RCLI</n>...</p><!-- Optional, may be present in vet instruments -->
  <p><n>RCPN</n>...</p><!-- Optional, may be present in vet instruments -->
  <p><n>RCED</n>...</p><!-- Optional, may be present in vet instruments -->
  <p><n>RPD</n>...</p>
  <p><n>RPDS</n>...</p>
  <p><n>RPDL</n>...</p>
  <p><n>RPDH</n>...</p>
  <p><n>RPDF</n>...</p>
  <p><n>WDDM</n>...</p>
  <p><n>WDDP</n>...</p>
  <p><n>WDMS</n>...</p>
  <p><n>WDMA</n>...</p>
  <p><n>WDFB</n>...</p>
  <p><n>WDLL</n>...</p>
  <p><n>WDLH</n>...</p>
  <p><n>WDCL</n>...</p>
  <p><n>WDCH</n>...</p>
  <p><n>WLGL</n>...</p><!-- Optional, may be present in vet instruments -->
  <p><n>WDIL</n>...</p>
  <p><n>WDIH</n>...</p>
  <p><n>XLT</n>...</p><!-- Optional, may be present in vet instruments -->
  <p><n>CAPL</n>...</p>
  <p><n>CLVL</n>...</p>
  <p><n>CEXP</n>...</p>
  <p><n>CEXT</n>...</p>
  <p><n>ASWN</n>...</p>
  <p><n>ASWP</n>...</p>
</smpinfo>
```

3.5.1. Sample Identification Parameters

3.5.1.1. ID – Blood Sample ID

ID is the identification of the blood sample. This parameter is always present, but the value is absent for "blank" ID:s.

Value range: 1 - 15 characters

Note 1: non-numeric characters are allowed in the ID. But leading or trailing white space will be filtered away.

Note 2: be prepared for non-ASCII characters.

Note 3: All control blood sample ID:s are a number followed by a trailing "+". (The number is normally the digits of the lot number of the control blood.)

Note 4: leading zero digits in ID numbers are significant.

Note 5: the maximum number of characters in ID numbers may change.

"Blank" ID

```
<p><n>ID</n></p>
```

Numeric ID

```
<p><n>ID</n><v>12356</v></p>
```

Non-numeric ID

```
<p><n>ID</n><v>Mrs. Smith</v></p>
```

Control blood ID

```
<p><n>ID</n><v>0606123+</v></p>
```

3.5.1.2. SEQ – Sample Run Sequence Number

SEQ is the sequence number of the sample run. This parameter is always present with a value.

Value range: 1 - 9999

Note 1: SEQ numbers are not unique in samples from one instrument. The operator can set the sequence number of the next sample, and sequence numbers wraps around.

Note 2: the upper limit of sequence numbers may change.

```
<p><n>SEQ</n><v>444</v></p>
```

3.5.1.3. DATE – Sample Run Date and Time

DATE is the date and time of the sample run. This parameter is always present with a value except when the date and time is not set in the instrument. The value is always formatted according to ISO 8601. See example below.

The date and time is always in the local time zone. There is never any time zone info.

```
<p><n>DATE</n><v>2004-05-06T07:08:09</v></p>
```

3.5.1.4. SORC – Sample or Control

SORC indicates if it is a normal sample run or a control sample run. This parameter is always present with a value.

Note 1: more *sample or control* types may be added. They will all be small positive numbers.

Note 2: the value range is in fact 0 and 2.

Value range:
0: normal sample
2: control sample with reference ranges

```
<p><n>SORC</n><v>0</v></p>
```

3.5.1.5. STYP – Sample Type

STYP indicates the sample type, i.e. if the run was a blood or a PLT concentrate sample. This parameter is always present with a value.

Note: more sample types may be added. They will all be small positive numbers.

Value range:
0: blood
1: PLT concentrate

```
<p><n>STYP</n><v>0</v></p>
```

3.5.2. Aspiration Parameters

3.5.2.1. ASPM - Aspiration Mode

ASPM indicates the aspiration mode for the run. This parameter is always present with a value. The value is one of the values listed in Table 1.

Value	Description
"OT"	Open Tube
"PD"	Prediluted sample
"CT"	Closed Tube = Cap piercer
"MC"	Micro capillary = MPA / MCI
"AS"	Auto sampler

Table 1 Listing of possible aspiration mode values.

```
<p><n>ASPM</n><v>OT</v></p>
```

3.5.2.2. ASPS – Aspiration Status

ASPS indicates the aspiration status for the run. This parameter is always present with a value.

The value is a status flag indicating if blood (or some other sample type) was detected during aspiration.

Value range:
0: no blood (or other) detected
1: blood (or other) detected
2: indeterminate due to e.g. timed aspiration

```
<p><n>ASPS</n><v>1</v></p>
```

3.5.2.3. BLMD - Background Mode Sample Run

BLMD indicates if the sample was run in background mode. Background mode is used for blank test runs. This parameter is always present with a value.

Note: the parameter BLNK indicates if the sample actually was a blank sample.

Value range:
0: not background mode
1: background mode

```
<p><n>BLMD</n><v>0</v></p>
```

3.5.2.4. BLNK - Blank Sample

BLNK indicates the sample run was a blank sample run or not, in the opinion of the instrument. This parameter is always present with a value.

Note: The definition of a blank is RBC < 0,50 and HGB < 2,0 (below c:a 15% of a normal human blood sample).

Value range:

0: not a blank sample

1: a blank sample

```
<p><n>BLNK</n><v>0</v></p>
```

3.5.2.5. ASWN - Autosampler Wheel Number

ASWN indicates the wheel number for an autosampler run. This parameter is always present. The value is only present for autosampler samples.

Note: more wheel numbers may be added.

Value range:

1 – 2

```
<p><n>ASWN</n><v>1</v></p>
```

3.5.2.6. ASWP - Autosampler Wheel Position

ASWP indicates the wheel position for an autosampler run. This parameter is always present. The value is only present for autosampler samples.

Value range:

1 - 20.

```
<p><n>ASWP</n><v>17</v></p>
```

3.5.3. Differentiation – Red Cells

The differentiation red cells and platelets is realised with a differentiation method that uses floating discriminators. A low limit and a high limit limit the floating range.

The floating discriminator is denoted RPD, the low and the high limits are denoted RPDL and RPDH. They are described in the sections 3.5.3.1, 3.5.3.3 and 3.5.3.4 respectively and illustrated in Figure 1.

If the floating discriminator fails, the fixed discriminator RPDF is used instead.

RPDS indicates if the discriminator is floating or fixed.

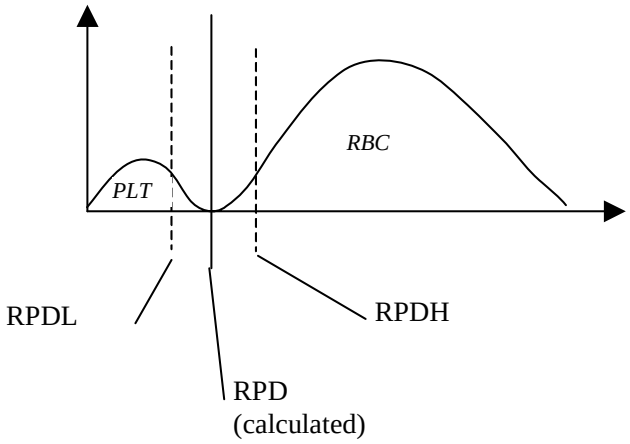

Figure 1 Illustration of the RPD floating discriminator and the range limits RPDL and RPDH.

3.5.3.1. RPD - RBC/PLT Discriminator position

RPD indicates the position of the RBC/PLT discriminator used during calculations. The position can either be a floating position in the range RPDL to RPDH (inclusive), or the fixed position RPDF. This parameter is always present with a value. The value is given in the unit *fl*.

Value range: 0 - 99

Note: the position of this discriminator is included in the PLT and RBC histograms too (see 3.7.2.6), but in that case as histogram bin indices.

```
<p><n>RPD</n><v>24</v></p>
```

3.5.3.2. RPDS - RBC/PLT Floating Discriminator Status

RPDS indicates if a suitable floating RBC/PLT discriminator was found. This parameter is always present with a value.

Value range:
0: a fixed RBC/PLT discriminator position was used (See 3.5.3.5) (PLT is "FD flagged")
1: a floating RBC/PLT discriminator position was used

```
<p><n>RPDS</n><v>1</v></p>
```

3.5.3.3. RPDL - RBC/PLT Floating Discriminator Low Range Limit

RPDL indicates the low range limit for the floating RBC/PLT discriminator. This parameter is always present with a value. The value is given in the unit *fl*.

A qualified operator can set the value of this parameter.

Value range: 0 - 99

```
<p><n>RPDL</n><v>12</v></p>
```

3.5.3.4. RPDH - RBC/PLT Floating Discriminator High Range Limit

RDPH indicates the high range limit for the floating RBC/PLT discriminator. This parameter is always present with a value. The value is given in the unit *fl*.

A qualified operator can set the value of this parameter.

Value range: 0 - 99

```
<p><n>RPDH</n><v>27</v></p>
```

3.5.3.5. RPDF - RBC/PLT Fixed Fallback Discriminator

RDPF indicates a fixed RBC/PLT discriminator when a suitable floating discriminator cannot be found. This parameter is always present with a value. The value is given in the unit *fl*.

A qualified operator can set the value of this parameter.

Value range: 0 - 99

```
<p><n>RPDF</n><v>30</v></p>
```

3.5.4. Differentiation – White Cells

The BM800 instrument has several different methods of separating the complete WBC population in 3 subpopulations: LYM, MID , GRAN (in increasing size order).

The instrument configuration allows or disallows selection of the different WBC diff methods.

3.5.4.1. WDMS - WBC Diff Method Set

WDMS indicates the WBC diff method selected in the instrument setup. This parameter is always present. The value is only present if the instrument actually tried a WBC diff.

This parameter is either fixed in the instrument configuration, or if not, a qualified operator can set the value of this parameter.

Note 1: WDMS and WDMA normally have the same value, except when a "fallback" from the selected WBC diff method to another one occurs. This happens when it isn't possible to calculate a WBC diff using

the selected method, but it is possible with the fallback method. Fallback is only used when the instrument configuration allows it, and only in veterinary instruments.

Note 2: more diff methods may be added. They will all be small positive numbers.

Note 3: the value range is 1, 2, 4.

Value range:
1: DM1 human floating
2: DM2 fixed
4: DM4 vet floating

```
<p><n>WDMS</n><v>1</v></p>
```

3.5.4.2. WDMA - WBC Diff Method Actually Used

WDMA indicates the floating WBC diff method actually used during the calculations. This parameter is always present. The value is only present if the instrument actually tried a WBC diff.

Note 1: more diff methods may be added. They will all be small positive numbers.

Note 2: the value range is 1, 2, 4.

Value range:
1: DM1 human floating
2: DM2 fixed
4: DM4 vet floating

```
<p><n>WDMA</n><v>2</v></p>
```

3.5.4.3. WDFB - WBC Diff Method Fallback enabled

WDFB indicates if the instrument should try a fixed WBC diff method if a floating WBC diff method fails during the calculations. This parameter is always present. The value is only present if the instrument has WBC diff method fallback enabled.

Value range:
0: WBC diff method fallback disabled
1: WBC diff method fallback enabled

```
<p><n>WDFB</n><v>0</v></p>
```

3.5.4.4. WDLL - WBC Floating Discriminator Low Range Limit (LYM-L)

WDLL indicates the low range limit for the floating WBC discriminator (LYM-L). The value is given in the unit *fl*. This parameter is always present. The value is present only if a floating differentiation method (DM1 or DM4) is used.

A qualified operator can set the value of this parameter.

Value range:
0 - 800

```
<p><n>WDLL</n><v>40</v></p>
```

3.5.4.5. WDLH - WBC Floating Discriminator High Range Limit (GRAN-H)

WDLH indicates the high range limit for the floating WBC discriminator (GRAN-H). The value is given in the unit *fl*. This parameter is always present. The value is present only if a floating differentiation method (DM1 or DM4) is used.

A qualified operator ca set the value of this parameter.

Value range:
0 - 800

```
<p><n>WDLH</n><v>360</v></p>
```

3.5.4.6. WDCL - WBC Calculated Floating Discriminator Low (LYM/MID)

WDCL indicates where a fixed LYM/MID discriminator should be positioned to report the same LYM result as was calculated using floating discriminators during this sample run. See Figure 2 in section 3.5.4.10 and substitute WDCL for WDIL. This position is visible in the WBC histogram when the *show as lines* mode is enabled. The value is given in the unit *fl*. This parameter is always present. The value is present only if a floating differentiation method (DM1 or DM4) is used.

Note: WDCL is also included in the WBC histogram as a discriminator line expressed as a bin number if the *show as lines* mode is enabled (see 3.7.2.6).

Value range:
0 - 800

```
<p><n>WDCL</n><v>111</v></p>
```

3.5.4.7. WDCH - WBC Calculated Floating Discriminator High (MID/GRAN)

WDCH indicates where a fixed MID/GRAN discriminator should be positioned to report the same GRAN result as was calculated using floating discriminators during this sample run. See Figure 2 in section 3.5.4.10 and substitute WDCH for WDIH. This position is visible in the WBC histogram when the *show as lines* mode is enabled. The value is given in the unit *fl*. This parameter is always present. The value is present only if a floating differentiation method (DM1 or DM4) is used.

Note: WDCH is also included in the WBC histogram as a discriminator line expressed as a bin number if the *show as lines* mode is enabled (see 3.7.2.6).

Value range:
0 - 800

```
<p><n>WDCH</n><v>129</v></p>
```

3.5.4.8. WLGL – WBC DM4 LYM/GRAN Limit

WLGL is the hard limit between LYM and GRAN populations for the floating WBC discriminator mode DM4. The hard limit is used when the floating differentiation method cannot find a suitable floating limit. The value is given in the unit *fl*. This parameter optional, and may be present in vet instruments. The value is present only if the DM4 floating differentiation method is used.

A qualified operator can set the value of this parameter.

Value range:
0 – 800

```
<p><n>WLGL</n><v>150</v></p>
```

3.5.4.9. WDIL - WBC Fixed Discriminator Low (MID-L)

WDIL indicates the low range limit for the fixed WBC discriminator (MID-L). See Figure 2 in section 3.5.4.10. The value is given in the unit *fl*. The parameter is always present. The value is present only if a differentiation method with fixed discriminators (DM2) is used, or if a fallback to such a method could have been used.

Note: WDIL is also included in the WBC histogram as a discriminator line expressed as a bin index (see 3.7.2.6).

A qualified operator can set the value of this parameter.

Value range:
0 - 800

```
<p><n>WDIL</n><v>110</v></p>
```

3.5.4.10. WDIH - WBC Fixed Discriminator High (MID-H)

WDIH indicates the high range limit for the fixed WBC discriminator (MID-L). See Figure 2 in section 3.5.4.10. The value is given in the unit *fl*. The parameter is always present. The value is present only if a differentiation method with fixed discriminators (DM2) is used, or if a fallback to such a method could have been used.

Note: WDIL is also included in the WBC histogram as a discriminator line expressed as a bin index (see 3.7.2.6).

A qualified operator can set the value of this parameter.

Value range:
0 - 800

```
<p><n>WDIH</n><v>130</v></p>
```

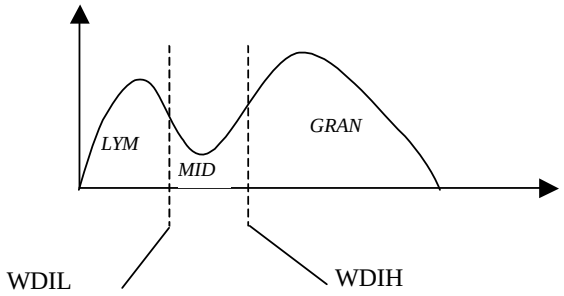

Figure 2 Illustration of the WBC fixed discriminators WDIL and WDIH.

3.5.4.11. WDDM - WBC Debris Discriminator Mode

WDDM indicates the used debris subtraction mode used during the WBC calculations. This parameter is always present with a value.

A qualified operator can set the value of this parameter.

Note 1: more WBC debris discriminator modes may be added. They will all be small positive numbers.

Note 2: Other WBC debris discriminator modes than 0 = fixed mode is only used in vet instruments.

Value range:
0: fixed mode
1: floating mode 1
2: floating mode 2

```
<p><n>WDDM</n><v>0</v></p>
```

3.5.4.12. WDDP - WBC Debris Discriminator Position

WDDP indicates the WBC fixed debris subtraction discriminator setting used during calculations The value is given in the unit *fl*. This parameter is always present with a value.

A qualified operator can set the value of this parameter.

Note: WDDP is also included in the WBC histogram as a discriminator line expressed as a bin index (see 3.7.2.6).

Value range:
0 – 99

```
<p><n>WDDP</n><v>45</v></p>
```

3.5.5. Control Blood with Reference Ranges

The operator inputs control blood assay information to specify reference ranges and other information about a control blood.

All control blood lots have a unique ID number that is used to separate different control blood lots.

Note 1: the unique ID number is normally the digits from the control blood lot number followed by a plus (+) sign. Example: lot number “0606-123” gets the ID number “0606123+”.

Note 2: leading zero digits in control blood ID numbers are significant.

3.5.5.1. CAPL – Control Blood Application

CAPL indicates the application (human/vet/blood bank) of a control blood with reference ranges. This parameter is always present. The value is only present if the blood was a control blood with reference ranges.

The value is a one character string: "H" for human, "V" for vet and "B" for blood bank. Other control blood applications may be added. All control blood application codes will be one uppercase letter.

Note: This information comes from the control blood reference range assay information.

```
<p><n>CAPL</n><v>H</v></p>
```

3.5.5.2. CLVL – Control Blood Level

CLVL identifies the level (normal/low/high/calibrator) of a control blood with reference ranges. This parameter is always present. The value is only present if the blood was a control blood with reference ranges.

The value is a one-character string: "N" for normal level, "L" for low level, "H" for high level, and "C" for calibrator. Other control blood levels may be added. All control blood level codes will be one uppercase letter.

Note 1: This information comes from the control blood reference range assay information.

Note 2: The control blood level is also present as a name in the parameter APNA (see 3.5.8.2)

```
<p><n>CLVL</n><v>N</v></p>
```

3.5.5.3. CEXP - Expiry Date of a Control Blood with Reference Ranges

CEXP is the expiry date of a control blood with reference ranges. This parameter is always present. The value is only present if the blood was a control blood with reference ranges.

The value is a date, always formatted according to ISO 8601. See example below.

Note: This information comes from the control blood reference range assay information.

```
<p><n>CEXP</n><v>2005-03-09</v></p>
```

3.5.5.4. CEXT - Extra Control Blood Information for a Control Blood with Reference Ranges

CEXT can identify extra control blood info for a control blood with reference ranges. This parameter is always present. The value is only present if the blood was a control blood with reference ranges, and there is some extra information.

Value range:
completely TBD

Note: this information might be included in the control blood reference range assay information.

No extra info:

```
<p><n>CEXT</n></p>
```

Some extra info (example only):

```
<p><n>CEXT</n><v>1</v></p>
```

3.5.6. Patient Information

Patient information may be added.

Note: The instrument will not manage a patient database, i.e. there is no way to relate samples to a certain patient record. The patient information is only textual information entered by the operator and hence its consistency cannot be guaranteed. Therefore no patient field or combination of fields can be considered as unique for a certain patient (i.e. no “primary key” exists).

3.5.7. Container Status

3.5.7.1. RGED - Diluent Container Empty

RGED indicates if the diluent reagent detector indicated that the diluent level in the container was too low before the sample run. This parameter is always present. The value is present if the diluent level was actually checked before the sample run.

Value range:

0: no diluent detected
1: diluent detected
2: indeterminate

```
<p><n>RGED</n><v>1</v></p>
```

3.5.7.2. RGEL - Hemolyzer Container Empty

RGEL indicates if the hemolyzer reagent detector indicated that the hemolyzer level in the container was too low before the sample run. This parameter is always present. The value is present if the hemolyzer level was actually checked before the sample run.

Value range:

0: no hemolyzer detected
1: hemolyzer detected
2: indeterminate

```
<p><n>RGEL</n><v>1</v></p>
```

3.5.7.3. RGEC - Cleaner Container Empty

RGEC indicates if the cleaner reagent detector indicated that the cleaner level in the container was too low before the sample run. This parameter is always present. The value is present if the cleaner level was actually checked before the sample run.

Note: cleaner reagent is only used in vet instruments.

Value range:

0: no cleaner detected
1: cleaner detected
2: indeterminate

```
<p><n>RGEC</n><v>1</v></p>
```

3.5.8. Settings

3.5.8.1. APNU – Analysis Profile Number

APNU is the number of the Analysis Profile (species in vet instruments) selected for the run. This parameter is always present. The value is present if the blood was not a control blood with reference ranges.

A qualified operator can set the value of this parameter.

Note 1: the upper limit may change.

Note 2: the initial instrument setup uses 1 for the “blood” analysis profile, and 2 for “background”.

Note 3: an analysis profile can have different numbers in different instruments, since a qualified operator can set up analysis profiles.

Value range: 1 - 15.

```
<p><n>APNU</n><v>1</v></p>
```

3.5.8.2. APNA – Analysis Profile Name / control blood level name

APNA is the name (if defined) of the Analysis Profile (species in vet instruments) selected for the run. This parameter is always present. The value is present if a name is defined for the selected Analysis Profile (species).

A qualified operator can set the value of this parameter.

The value is a string of 1 to 15 characters.

Note 1: be prepared for non-ASCII characters.

Note 2: the initial instrument setup uses "BLOOD" for the “blood” analysis profile, and "BACKGROUND" for “background”. The control bloods with reference ranges have "LOW", "NORMAL", "HIGH", and "CALIBRATOR".

Note 3: The predefined names in the previous note may be translated to other languages.

Note 4: A qualified operator can set the names of analysis profiles.

APNA without a name:

```
<p><n>APNA</n></p>
```

APNA with an initially defined name:

```
<p><n>APNA</n><v>BLOOD</v></p>
```

APNA with a user set name:

```
<p><n>APNA</n><v>CHILD</v></p>
```

APNA for a control blood with reference ranges:

```
<p><n>APNA</n><v>NORMAL</v></p>
```

3.5.8.3. XLT - Extra Hemolyzing Time

XLT indicates the use of a longer hemolyzing time than normal. Only used in vet models. The value is given in seconds with one decimal. This parameter is always present. Value only present if a longer hemolyzing time was used.

A qualified operator can set the value of this parameter.

Value range:
0.0 – 99,9.

No extra lyze time:

```
<p><n>XLT</n></p>
```

Extra lyze time:

```
<p><n>XLT</n><v>5.0</v></p>
```

3.5.9. Reagent statistics

The BM800 instrument has an optional reagent statistics system that keeps track of the reagents used in the instrument.

Note: The diluent and hemolyzing regent information is exactly the same if a reagent "combo" pack is used.

3.5.9.1. RDLI Reagent Statistics Diluent Lot identifier

RDLI is the lot identifier of the diluent reagent. This parameter is always present. The value is only present if the reagent statistics system is enabled.

The value is always formatted as "YYMM-NNN", where YY, MM and NNN are digits. (Normally YY = production year, MM =production month and NNN = lot sequence number.) See example below.

Note 1: This information comes from the reagent package information.

Note 2: leading zero digits in reagent lot identifiers are significant.

```
<p><n>RDLI</n><v>0606-123</v></p>
```

3.5.9.2. RDPN Reagent Statistics Diluent Package number

RDPN is the package number within one the lot of the diluent reagent. This parameter is always present. The value is only present if the reagent statistics system is enabled.

Value range:
1-9999.

Note: This information comes from the reagent package information.

```
<p><n>RDPN</n><v>1234</v></p>
```

3.5.9.3. RDED Reagent Statistics Diluent Expiry date

RDED is the expiry date of the diluent reagent. This parameter is always present. The value is only present if the reagent statistics system is enabled.

The value is a date, always formatted according to ISO 8601. See example below.

Note: This information comes from the reagent package information.

```
<p><n>RDED</n><v>2006-06-29</v></p>
```

3.5.9.4. RLLI Reagent Statistics Hemolyzer Lot identifier

RLLI is the lot identifier of the hemolyzing reagent. This parameter is always present. The value is only present if the reagent statistics system is enabled.

The value is always formatted as "YYMM-NNN", where YY, MM and NNN are digits. (Normally YY = production year, MM =production month and NNN = lot sequence number.) See example below.

Note 1: This information comes from the reagent package information.

Note 2: leading zero digits in reagent lot identifiers are significant.

```
<p><n>RLLI</n><v>0606-124</v></p>
```

3.5.9.5. RLPN Reagent Statistics Hemolyzer Package number

RLPN is the package number within one the lot of the hemolyzing reagent. This parameter is always present. The value is only present if the reagent statistics system is enabled.

Value range:
1-9999.

Note: This information comes from the reagent package information.

```
<p><n>RLPN</n><v>1234</v></p>
```

3.5.9.6. RLED Reagent Statistics Hemolyzer Expiry date

RLED is the expiry date of the hemolyzing reagent. This parameter is always present. The value is only present if the reagent statistics system is enabled.

The value is a date, always formatted according to ISO 8601. See example below.

Note: This information comes from the reagent package information.

```
<p><n>RLEL</n><v>2006-06-29</v></p>
```

3.5.9.7. RCLI Reagent Statistics Cleaner Lot identifier

RCLI is the lot identifier of the cleaning reagent. This parameter optional, and may be present in vet instruments. The value is only present if the reagent statistics system is enabled.

The value is always formatted as "YYMM-NNN", where YY, MM and NNN are digits. (Normally YY = production year, MM =production month and NNN = lot sequence number.) See example below.

Note 1: This information comes from the reagent package information.

Note 2: leading zero digits in reagent lot identifiers are significant.

```
<p><n>RLLI</n><v>0606-125</v></p>
```

3.5.9.8. RCPN Reagent Statistics Cleaner Package number

RLPN is the package number within one the lot of the cleaning reagent. This parameter optional, and may be present in vet instruments. The value is only present if the reagent statistics system is enabled.

Value range:
1-9999.

Note: This information comes from the reagent package information.

```
<p><n>RLPN</n><v>1234</v></p>
```

3.5.9.9. RCED Reagent Statistics Cleaner Expiry date

RLED is the expiry date of the cleaning reagent. This parameter optional, and may be present in vet instruments. The value is only present if the reagent statistics system is enabled.

The value is a date, always formatted according to ISO 8601. See example below.

Note: This information comes from the reagent package information.

```
<p><n>RLEL</n><v>2006-06-29</v></p>
```

3.6. <smpresults> Sample Result Parameter Values

Sample result parameter values are the measured and calculated hematology parameters from a sample run.

The parameter order below is not fixed. The parameters might be in another order. Be prepared for added and removed parameter definitions.

```
<smpresults >
  <p><n>RBC</n>...</p>
  <p><n>MCV</n>...</p>
  <p><n>HCT</n>...</p>
  <p><n>MCH</n>...</p>
  <p><n>MCHC</n>...</p>
  <p><n>RDWR</n>...</p> <!-- RDW% -->
  <p><n>RDWA</n>...</p> <!-- RDWa -->
  <p><n>PLT</n>...</p>
  <p><n>MPV</n>...</p>
  <p><n>PCT</n>...</p>
  <p><n>PDW</n>...</p>
  <p><n>LPCR</n>...</p>
```

```
  <p><n>HGB</n>...</p>
  <p><n>WBC</n>...</p>
  <p><n>LA</n>...</p> <!-- LYM# -->
  <p><n>MA</n>...</p> <!-- MID# -->
  <p><n>GA</n>...</p> <!-- GRAN# -->
  <p><n>LR</n>...</p> <!-- LYM% -->
  <p><n>MR</n>...</p> <!-- MID% -->
  <p><n>GR</n>...</p> <!-- GRAN% -->
</smpresults >
```

3.6.1. Result Parameter Format

A result parameter always has a name and a normal / reference range. It could also have a value or a out-of-range flag, and an error flag.

3.6.1.1. <v> - Parameter Value

The "<v>" tag is the normal parameter value. It is present if a valid value was calculated for the run.

The data type of the result parameters is either floating point or scaled integer.

For example, if the instrument counts RBC as 0.00, then there are no cells to calculate an MCV value from. The instrument reports the RBC value as 0.00, since it did actually not count any cells, but the MCV value is absent.

The value range and the number of decimals depend of the parameter. (*Implementation note:* The BM800 stores each parameter internally as a 16-bit scaled integer, so the "mantissa range" is 0 - 65535.)

The "<v>" and "<r>" tags are mutually exclusive. A value is either not calculated (neither "<v>" nor "<r>"), calculated ("<v>" only), or out-of-range ("<r>" only).

3.6.1.2. <r> - Parameter Out-of-Range

The "<r>" tag is only present if the parameter value is either completely out-of-range, or could not be determined for some unusual reason. For example, if the MCV value is absent due to a blank run, then there is no "<r>" tag. But if the RBC value is absent due to a ridiculously high count (above 14 in human models), then BM800 outputs a "<r>H</r>".

The "<r>" tag can take the values "H" (value too high) or "L" (value too low).

The "<r>" and "<v>" tags are mutually exclusive. A value is either not calculated (neither "<v>" nor "<r>"), calculated ("<v>" only), or out-of-range ("<r>" only).

3.6.1.3. <f> - Parameter Error Flag

The "<f>" tag is an optional parameter error flag. Some errors block the calculation of the corresponding parameter value, while others provide additional information.

The error flag is always two upper-case letters, or one upper-case letter followed by a digit. The list of possible flags is not included here. Be prepared for any combination of two uppercase letters, or one uppercase letter followed by one digit.

Only these parameters can have an error flag: RBC, MCV, PLT, HGB, WBC, LYM, MID, GRAN.

3.6.1.4. <l> - Parameter Normal/Reference Range Low

The "<l>" tag is always present. Its value is the low end of the normal / reference(*) range. If the parameter value is exactly equal to "<l>", then it is within the normal / reference range.

(* It is called the normal range for normal blood, and reference range for control blood with reference ranges.)

See the <v> tag above for information about values ranges etc.

3.6.1.5. <h> - Parameter Normal/Reference Range High

The "<h>" tag is always present. Its value is the high end of the normal / reference range. If the parameter value is exactly equal to "<h>", then it is within the normal / reference range.

See the <v> tag above for information about values ranges etc.

3.6.1.6. Examples

Examples:

A result parameter could be without any value, for example if the value could not be calculated:

```
<p><n>MCV</n><l>70.0</l><h>100.0</h></p>
```

Normally, a result parameter has a value:

```
<p><n>RBC</n><v>4.56</v><l>3.50</l><h>5.50</h></p>
```

Or it could have a value and an error flag:

```
<p><n>PLT</n><v>234</v><f>FD</f><l>100</l><h>400</h></p>
```

Or an error flag only:

```
<p><n>WBC</n><f>TU</f><l>5.5</l><h>8.5</h></p>
```

Or it could have an out-of-range flag instead of a value (possibly with an error flag too).

```
<p><n>HGB><r>H</r><l>12.5</l><h>16.5</h></p>
```

3.6.2. Description of the Sample Result Parameters

Result Parameter	Description	Unit	Existence Condition
RBC	Red blood cell concentration	10^{12}/l	Red side counting succeeded.
MCV	Mean Cell Volume	fl	Red side counting succeeded, and enough cells counted.
HCT	Hematocrit value	%	RBC and MCV values present.
MCH	Mean Cell Haemoglobin	pg	RBC and HGB values present.
MCHC	Mean Cell Haemoglobin Concentration	g/dl	RBC, HGB and MCV values present.
RDWR	Red blood cell distribution width, relative. Note: RDW relative is the commonly accepted RDW value.	%	Red side counting succeeded, and enough cells counted.
RDWA	Red blood cell distribution width, absolute.	fl	Red side counting succeeded, and enough cells counted.
PLT	Platelet concentration (trombocytes).	10^{9}/l	Red side counting succeeded.
MPV	Mean Platelet Volume.	fl	Red side counting succeeded, and enough cells counted.
PCT	Platelet Crit. As HCT but for platelets (PLT).	%	PLT and MPV values present.
PDW	Platelet distribution width.	fl	Red side counting succeeded, and enough cells counted.
LPCR	Large Platelet Concentration Ratio. ("PLT cells > 12 fl" / PLT)	%	Red side counting succeeded, and enough cells counted.
HGB	Haemoglobin concentration.	g/dl	HGB measuring succeeded.

WBC	White blood cell concentration	10^9/l	White side counting succeeded.
LA	Lymphocyte concentration	10^9/l	White side counting succeeded, enough cells counted, and WBC diff calculation succeeded.
MA	Mid concentration	10^9/l	White side counting succeeded, enough cells counted, and WBC diff calculation succeeded.
GA	Granulocyte concentration.	10^9/l	White side counting succeeded, enough cells counted, and WBC diff calculation succeeded.
LR	Lymphocyte concentration, relative total WBC. Given in percent (%).	%	White side counting succeeded, enough cells counted, and WBC diff calculation succeeded.
MR	Mid concentration, relative total WBC. Given in percent (%).	%	White side counting succeeded, enough cells counted, and WBC diff calculation succeeded.
GR	Granulocyte concentration, relative total WBC. Given in percent (%).	%	White side counting succeeded, enough cells counted, and WBC diff calculation succeeded.

3.7. <hgrams> Histograms

Note about terminology: This specification could call graphical results histograms or graphs (with plotted curves). Histograms are used in statistics to show the relative frequency of discrete "bins" of some underlying measurement. Graphs are used for example in mathematics to show continuous curves over function values. In a way, a graphical representation of the BM800 raw data would be a histogram, while a graphical representation of filtered data would be more like a curve in a graph. Messy. Let's call them histograms. (And let's separate out the actual graphical representation from the data representation here. The data here is just a vector of values.)

The whole "<hgrams>" part is optional. The control sample memory has no histograms, and the user might choose not to include histograms in the output data.

Note 1: when presenting histograms, set the y scale max of the graphical presentation around 50 to 60. If the maximum value in the histogram is above the y scale max, scale down all values in the histogram accordingly. If the maximum value in the histogram is below the y scale max, present them as they are. This way, samples with few cells will show low curves in the histograms, and the external histograms will look like the histograms from the BM800 instrument. (The BM800 instrument uses a y scale max of 48 for the display, and 64 for printouts.)

Note 2: when presenting 3 diff histograms (LYM, MID, GRA), the y scale max for all 3 of them must be the same!

There are two variants of histogram layouts. Variant 1 is normally used for successful floating WBC diff methods, execpt when the *show as lines* mode is enabled. Variant 2 is normally used in all other cases.

Variant 1: 3 diff histograms (LYM, MID, GRA), which should be overlaid to create one WBC histogram with 3 populations. BM800 uses the multiple <hgdata> nodes in the <hgram> construct to indicate this. In this case, the <hgdata> nodes also have a <n> tag that indicates the name of each histogram vector that is a part of the overlaid histogram.

```
<hgrams>
  <hgram>
    <n>PLT</n><m>30</m><k>80</k><w>7</w><d>64</d>
    <hgdata>
      <v>
        0 0 2 5 7 ... <!-- 80 numbers in total, separated by white space -->
      </v>
```

```
    </hgdata>
  </hgram>
  <hgram>
    <n>RBC</n><m>250</m><k>80</k><w>2</w><d>8</d>
    <hgdata>
       <v>
         0 1 3 1 ... <!-- 80 numbers in total, separated by white space -->
       </v>
    </hgdata>
  </hgram>
  <hgram>
    <n>WBC</n><m>450</m><k>80</k><w>4</w><d>8</d>
    <hgdata>
      <n>LYM</n>
      <v>
        0 0 1 3 5 ... <!-- 80 numbers in total, separated by white space -->
      </v>
    </hgdata>
    <hgdata>
      <n>MID</n>
      <v>
        0 0 0 0 0 ... <!-- 80 numbers in total, separated by white space -->
      </v>
    </hgdata>
    <hgdata>
      <n>GRA</n>
      <v>
        0 0 0 0 0 ... <!-- 80 numbers in total, separated by white space -->
      </v>
  </hgram>
</hgrams>
```

Variant 2, One WBC histogram with 2 (optional) discriminator lines.

```
<hgrams>
  <hgram>
    <n>PLT</n><m>30</m><k>80</k><w>7</w><d>64</d>
    <hgdata>
      <v>
        0 0 2 5 7 ... <!-- 80 numbers in total, separated by white space -->
      </v>
    </hgdata>
  </hgram>
  <hgram>
    <n>RBC</n><m>250</m><k>80</k><w>2</w><d>8</d>
    <hgdata>
      <v>
        0 1 3 1 ... <!-- 80 numbers in total, separated by white space -->
      </v>
    </hgdata>
  <hgram>
    <n>WBC</n><m>450</m><k>80</k><w>4</w><d>22</d><d>26</d><d>8</d>
    <hgdata>
      <v>
        0 0 1 3 5 ... <!-- 80 numbers in total, separated by white space -->
      </v>
    </hgdata>
  </hgram>
</hgrams>
```

3.7.1. <hgrams> - Histograms Section

The <hgrams> section collects all the histograms (<hgram>) in this sample.

The <hgrams> section is optional. The operator can enable and disable the inclusion om the <hgrams> section> in the sample data. The control memory has no histogram data.

3.7.2. <hgram> - Histogram Format

All tags must be present and in this order, exceptions for:

1) The discriminator tag "<d>" that may be present zero, one or more than one times. It depends on the instrument setup and the sample results.
2) The histogram data vector tag <hgdata> that may be present one or more than one times. It depends on the instrument setup and the sample results.
3) The histogram data vector tag <hgdata> can have an optional <n> tag as the first element, if it part of an overlaid histogram.

```
<hgram>
  <n>WBC</n>
  <min>5</min><!-- Optional. If not present the min range is 0. -->
  <m>450</m>
  <k>80</k>>
  <w>4</w>
  <d>8</d>
  <hgdata>
    <n>LYM</n><!-- Optional. Only present for overlaid histograms.-->
    <v>
      0 0 2 5 7 ...
    </v>
  </hgdata>
</hgram>
```

3.7.2.1. <n> – Histogram Name

The "<n>" tag specifies the name of histogram.

3.7.2.2. <min> - Histogram Min Range

The "<min>" tag specifies the cell volume of the cells that are represented by the bin at the left end of the histogram, i.e. the smallest cells. (Min value of x-axis).

This tag is optional. If it does not exist, the default minimum value is 0.

Unit: fl (femtoliter).

Range: 0 –99 (PLT), 0 – 400 (RBC) or 0 – 800 (WBC)

3.7.2.3. <m> - Histogram Max Range

The "<m>" tag specifies the cell volume of the cells that are represented by the bin at the right end of the histogram, i.e. the largest cells. (Max value of x-axis).

Qualified operators can set the value of this tag.

The initial values for this tag are 250 for the RBC histogram, 30 for PLT and 450 for WBC.

Unit: fl (femtoliter).

Range: 0 –99 (PLT), 0 – 400 (RBC) or 0 – 800 (WBC)

3.7.2.4. <k> - Histogram Number of Bins

The "<k>" tag specifies the number of data points (bins) in the histogram.

Note 1: It has always been 80 data points in all BM800 histograms, but this may change.

Note 2: Do not assume that all histograms in one sample have the same number of data points. This may change. Exception: the "LYM", "MID", "GRA" histograms must of course have the same number of data points, since they are overlaid to create a single three-population histogram.

3.7.2.5. <w> - Histogram Filter Factor

The "<w>" tag is specifies the filter factor and indicates how hard the instrument has filtered the histogram. The values 0 and 1 mean no filtering while 2, 3, 4, … mean increasingly harder filtering. (If a max filter factor is needed, set it to "<k>/2".)

Qualified operators can set the value of this tag.

3.7.2.6. <d> - Histogram Discriminator Index

The "<d>" tag(s) optionally specifies the indices of the bins where the discriminators are positioned in the histogram. This tag is optional. A histogram may have more than one discriminator, so the tag may be present more than once.

The histogram discriminator index is zero-based.

Unit: data point index. In the example above 64 for "PLT" gives a scale value of <d>*(<m>/<k>) = 64*(30/80) = 24.

Note 1: The exact definition of the scale values for the bins of the histogram is a bit fuzzy. And so is the exact definition of the discriminator position: is it supposed to be at the left edge, in the middle or at the right edge of the position range for a certain bin(?).(Assume for now that it is in the middle.)

Note 2: Do not assume that the maximum number of discriminator lines is 3. (At least not during development.) If a fixed maxis needed, use 8 (is possible) until further notice.

Note 3: do not assume that the discriminators are output with ascending positions.

Note 4: The discriminator for the PLT and RBC histograms is also included as RPD (see 3.5.3.1).

Note 5: The discriminators for the WBC histograms is also include as WDIL (see 3.5.4.9) / WDIL (see 3.5.4.10) or WDCL (see 3.5.4.6) / WDCH (see 3.5.4.7), plus WDDP (see 3.5.4.12).

3.7.3. <hgdata> - Histogram Data Vector

The <hgdata> element represents a histogram data vector, i.e. the values for each bin in the histogram

Each histogram, <hgram>, may have one ore more histogram data vectors. The latter case with more than one data vector represents an "overlay" histogram, i.e. a histogram with more than one data vectors drawn in the same histogram.

```
<hgdata>
  <n>LYM</n><!-- Optional. Only present for overlaid histograms.-->
  <v>
    0 0 2 5 7 ...
  </v>
</hgdata>
```

3.7.3.1. <n> - Histogram Data Vector Name

The "<n> tag specifies the name of the histogram data vector. The name is optional if there is only one data vector present in a single histogram, but it is mandatory of there is more than one data vector in a single histogram.

3.7.3.2. <v> - Histogram Data Vector Values

The "<v>" tag specifies the data points (bin values) of the histogram as a data vector. The vector values are represented is a list of numbers, separated by at least one character of white space.

Value range of a vector value:
0 - 255

Index range:
0 - "<k>-1"

Note: Try to apply some formatting to the data point numbers. For instance output 80 data points as 5 rows with 16 data points each. (This is mainly for those human readers who are able to "see" the histogram by looking at the data values.)

4. Transport Package

The data that belongs to a certain sample run and is represented by the XML structure described in section 3 is subject to be transferred via the communication port (today RS232) of the instrument. To ensure that the sample data is correctly transferred BM800 needs to add some mechanisms related to the transport level of a protocol stack. This specification therefore defines the term *package*. A package has the following logical structure:

```
{Checksum begin}
   {Message begin}
      {Message content/"Payload". Here Sample Data.}
   {Message end}
{Checksum end}
```

So what BM800 actually sends over the communication link are packages, one package per sample.

The Checksum part is described in section 4.2 while the Message part is described in section 4.3.

4.1. Package delimiters

When parsing a package it is important to know where a package starts and ends. Therefore, package begin and end delimiters should be used.

No explicit package delimiters are defined, so instead the static heading part of the checksum begin and end tokens (see section 4.2) are used. That gives us the following:

Package begin token:

```
<!--:Begin:Chksum:
```

Package end token:

```
<!--:End:Chksum:
```

4.2. Checksum armour

Since BM800 will use serial communication, at least in the first version, and serial communication is not 100% safe, this specification add a generic check summing mechanism to the BM800 XML communication. BM800 therefore introduces checksum armour.

The data that is subject for checksum control is delimited by a checksum begin token and a checksum end token and data according to the following:

Checksum begin token:

```
<!--:Begin:Chksum:{Algorithm ID}:-->
```

Checksum end token, generic:

```
<!--:End:Chksum:{Algorithm ID}:{Checksum...}:-->
```

`{Algorithm ID}`	Corresponds to an integer that identifies what checksum algorithm that has been used. The different check sum algorithms are described in section 4.2.2.
`{Checksum...}`	Is the optional checksum for the checksum with a certain algorithm ID. Disregard when not implementing an algorithm ID.

Checksum end token, algorithm 1:

```
<!--:End:Chksum:1:{Checksum byte 1}:{Checksum byte 2}:-->
```

`1` Algorithm ID for checksum algorithm 1.

`{Checksum byte 1}` Is the first byte of the calculated checksum. The value is in the range 0 – 255.

`{Checksum byte 2}` Is the second byte of the calculated checksum. The value is in the range 0 – 255.

Example:

```
<!--:Begin:Chksum:1:-->
   {Data to be check summed.}
<!--:End:Chksum:1:105:246:-->
```

The checksum tokens have the syntax of an XML comment (surrounded by the "`<!–`" and "`-->`" tokens) to not break the XML structure of the sample data described in section 3 and hence the data may be displayed in any XML viewer application.

The checksum is calculated for every character (byte) that exist after the ">" character in the checksum begin token and up to but not including the "<" character in the checksum end token. This means that white spaces and new lines also are included in the checksum calculation, including newlines at the beginning and at the end. Notice though that new lines are handled according to the XML standard, i.e. they are always replaced with the line feed character (ASCII 10) as described in section 4.2.1.

The data between checksum begin and end tokens should be considered as a "byte stream" and hence the format of the data is irrelevant. Note that no XML parsing of the data should be performed before the checksum calculation.

4.2.1. New Line handling

In the checksum calculation the transport package handling have to handle the different ways of representing a new line. It therefore adopts the line ending handling in the XML standard:

- a "\n" (NL, ASCII 10) not followed by a "\r" (CR, ASCII 13) is treated as it is,
- the character pair "\r\n" is treated as a single "\n",
- and a "\r" not followed by a "\n" is also treated as a single "\n".

4.2.2. Checksum algorithm

An ID of the used checksum algorithm is given in the checksum tokens. The table below describes the different checksum algorithms and their IDs.

Of course the checksum begin and checksum end tokens must have the same algorithm ID.

Currently there exists only one checksum algorithm.

Checksum algorithm ID	Description
0	No checksum algorithm used. Disregard any "checksum" in the checksum end token.
1	"Easy calculation" checksum as described in section 4.2.2.1.

4.2.2.1. "Easy calculation" checksum algorithm

Here is a checksum algorithm which is very easy to calculate, and even easier to verify.

C code sample:

The basic check calculation for a sequence of bytes looks like this in C. The actual checksum in placed in the two last bytes (`buf[ lim-2 ], buf[ lim-1 ]`). They should be extracted from *Checksum byte 1* and *Checksum byte 2* in the *Checksum end token, algorithm 1*, and appended as two bytes after the other package content in the buffer. See 4.2.

```
void
cs_calc( const unsigned char buf[], // Pointer to buffer
         unsigned start, // Index of first check summed byte
         unsigned lim, // Index of last+1 check summed byte
         unsigned char cs[2] ) // Output checksum
{
    unsigned char cs1 = 0, cs2 = 0;
    for( ; start < lim; ++start ){
        cs1 += buf[start];
        cs2 += cs1;
    }
    cs[0] = cs1;
    cs[1] = cs2;
}
```

The correct result when checking check summed data is "`cs[0]==0 && cs[1]==0`".

This is how to set the checksum in the two last bytes in a buffer, so it has a correct checksum:

```
void
cs_set( unsigned char buf[], // Pointer to buffer (2 bytes changed)
        unsigned start, // Index of first check summed byte
        unsigned lim ) // Index of last+1 check summed byte
{
    // precond: start+2 <= lim
    unsigned char cs[2];
    buf[lim-2] = buf[lim-1] = 0; // Init checksum bytes for calc
    cs_calc( buf, start, lim, cs );
    buf[lim-2] = cs[0] - cs[1];
    buf[lim-1] = cs[1] - 2*cs[0];
}
```

`buf[lim-2]` and `buf[lim-1]` shall be inserted as *Checksum byte 1* and *Checksum byte 2* in *Checksum end token, algorithm 1* during transfer. They shall not be included in the package content!

C# Code sample:

The data to be check summed is stored in the `msg` string variable. The two checksums are returned in the out parameters `checksum1` and `checksum2`.

```
private void CalcCheckSum(string msg, out byte checksum1, out byte
checksum2)
{
  byte cs1 = 0;
  byte cs2 = 0;

  // Always use "\n" as line break when calculating the checksum.
  msg = msg.Replace("\r\n", "\n"); // Find and replace CR LF with LF
  msg = msg.Replace("\r", "\n"); // Find and replace CR with LF.

  for (int i = 0; i < msg.Length; i++)
  {
    cs1 += (byte) msg[i];
    cs2 += cs1;
  }

  checksum1 = cs1;
  checksum2 = cs2;
}
```

Then add the checksums read from the transferred package to the calculated checksums in the following way:

```
checksum1 += transferredChecksum1;
checksum2 += checksum1;
checksum1 += transferredChecksum2;
checksum2 += checksum1;
```

The values of `checksum1` and `checksum2` should now be 0.

4.3. Message Handling

The checksum armour makes it possible to detect communication errors within the check-summed data. But what happens if one or more complete packages is lost? To handle this problem a Message part is added to the package. The message part is delimited by tokens according to the following:

Message begin:

```
<!--:Begin:Msg:{Message ID}:{Acknowledge flag}:-->
```

Message end:

```
<!--:End:Msg:{Message ID}:{Acknowledge flag}:-->
```

`{Message ID}` The message ID is an integer that identifies the message in a sufficiently unique way. The message ID is a sequential number that may be used to detect if any complete package has been lost. The algorithm for ID assignment to a message is described in section 4.3.1.

`{Acknowledge flag}` The acknowledge flag is set to 1 by the transmitting side to indicate that an acknowledge message is expected by the transmitting side and 0 if no acknowledge is expected. The acknowledge management is described in section 4.3.2.

Example:

```
<!--:Begin:Msg:3:0:-->
   {Message content}
<!--:End:Msg:3:0:-->
```

4.3.1. Message ID assignment

The message ID is a single digit number. The transmitting side sets the message ID in transmitted messages.

The transmitting side must have a variable containing the next message ID of the next numbered message. The transmitting side sets the message ID of the next numbered message to 1 during initialisation. Then the transmitting side updates the message ID of the next numbered message after every completed transmission according to the table below. (In the presence of acknowledges: completed = after receiving an acknowledge message, or giving up re-transmitting.)

The receiving side must have a variable containing the message ID of the last received numbered message. The receiving side sets the message ID of the last received numbered message during initialisation so it does not match any valid message ID (Suggestion: initialize it to 10). Every time the receiving side receives a numbered message, it sets the message ID of the last received numbered message to the message ID of the last numbered message received.

The BM800 sets the message ID of the next numbered message to 1 during “power up” and “exit standby”. BM800 may also sets the message ID of the next numbered message to 1 at other times.

The possible values for message ID:s are:

0 This message is unnumbered, and is not included in the sequence of numbered messages. (This should only be used during development, since it is impossible to detect duplicate messages by looking at the message ID alone.)

1 This message is the initial message of a sequence of numbered messages.

2 – 9 This message is one of the non-initial messages of a sequence of numbered messages.

If the message ID of a transmitted message is:	The message ID of the next numbered message should be:
0	Unchanged.
1	Set to 2.
2 – 8	Set to current message number + 1.
9	Set to 2.

Example:
The transmitted message ID:s of a normal sequence of numbered messages:
1 2 3 4 5 6 7 8 9 2 3 4 5 6 7 8 9 2...

Example:
The transmitted message ID:s of a sequence of numbered messages with two unnumbered messages inserted:
1 0 2 3 4 5 6 7 8 9 2 3 4 5 0 6 7 8 9 2...

Example:
The transmitted message ID:s of a sequence of numbered messages where the transmitting side re-initialise the sequence of numbered message:
1 2 3 4 5 6 7 8 9 2 3 1 2 3 4 5 6 7 8 9 2 3 4 5 6 7 8 9 2...

4.3.2. Acknowledge messages

The receiving side shall return an acknowledge message as soon as possible when it receives a message from the transmitting side with the acknowledge flag set to 1.

The message ID of the acknowledge message shall be set to the message ID of the message it acknowledges.

Message acknowledge:

```
<!--:Ack:Msg:{Message ID}:{Acknowledge type}:-->
```

`{Message ID}` The message ID of the acknowledge message shall be set to the message ID of the message it acknowledges.

`{Acknowledge type}` The receiving side sets to acknowledge type to indicate if the message content was accepted. The acknowledge type shall be set according to the following table.

Acknowledge type	Description
0	The message was received correctly, and has been accepted.
1	The message was received correctly, but was not accepted due to some temporary condition that will probably disappear in the near future. The BM800 instrument can not handle some messages during cycle runs, for example. The transmitting side can either report this as an error, or retransmit the same message at a later time (and with an updated message ID). Note: The BM800 instrument does not retransmit messages in this case. It reports an error instead.
2	The message was received correctly, but was not accepted due to some permanent condition. Example: the message had indeed a correct checksum, but contained malformed XML data. Note: The BM800 instrument reports an error in this case.

Example of an acknowledge message for an accepted message, with a correct checksum of type 1:

```
<!--:Begin:Chksum:1:--><!--:Ack:Msg:3:0:--><!--:End:Chksum:1:184:62:-->
```

4.3.3. Retransmission of lost messages

If the transmitting side transmits a message with the *acknowledge flag* set 1, the receiving side shall return an acknowledge message as soon as possible. If the transmitting side does not receive an acknowledge message with the right message ID within a suitable time, it shall retransmit exactly the same message, including the same message ID.

The transmitting side shall repeat this process a suitable number of times.

The BM800 instrument has an acknowledge timeout of 5 seconds from the transmission of the last character in the message

The BM800 instrument sends each message at most 3 times (=1 initial transmission and up to 2 retransmissions).

4.3.4. Handling of duplicate message ID:s

If the receiving side receives a message with the same message ID as the previous received message, and the message ID is 2 or higher, then it shall acknowledge the message but silently drop it contents, since the receiving side already has the message content. The acknowledge message shall have the same *acknowledge type* as the first time this message was acknowledged.

This situation happens when the message was transferred correctly, but the corresponding acknowledge message was lost.

The receiving side should implement a maximum time for the duplicate message ID handling. If the time between two messages is longer than some suitable maximum time, the second message shall be treated as a normal, out of sequence message.

The BM 800 instrument has a maximum duplicate message time of 120 seconds from the transmission of the last character in the message.

4.3.5. Handling of missing message ID:s

The receiving side may take some suitable action when it detects a missing message ID in sequence of numbered messages. (Suitable actions can be to log the event and trying to get the attention of the operator).

The receiving side shall then process the received message in the usual way. (Remember: the receiving side must receive a valid message with a new message ID to detect a missing message ID).

4.3.6. Handling of incomplete packages

If the receiving side detects a *package begin token* (see 4.1) inside another package, it shall drop the partly received package and begin receiving the new one.

The receiving side may take some suitable action (like logging the event) when this happens.

4.3.7. Handling of packages that fails the checksum test, or has transport package or message format errors

The receiving side shall drop all received packages that fails the checksum test (see 4.2).

The receiving side shall also drop all received packages which does not conform to the transport package format or message format described here.

The receiving side may take some suitable action (like logging the event) when any of this happens.

4.3.8. Handling of messages with undefined message content

The receiving side may drop all received messages that has a message content which does not conform to any format described in this specification.

If the transmitting side has set the *acknowledge flag* in the message, the receiving side may return an acknowledge message with the *acknowledge type* set to 2 (permanent error condition) in this case.

(Note: there may be receiving side applications which acknowledges all correctly received messages, but does not do any interpretation of the message content. Not even checking if it seems to be in XML format.)

Note: the BM800 instrument acknowledges badly formatted messages content with an acknowledge message with the *acknowledge type* set to 2.

4.3.9. Some comments about the message handling in BM800

It is possible to use a "dumb" terminal program (like Hyperterm) to capture and store the output data. But of course the BM800 cannot retransmit garbled messages in this case.

The message handling in BM800 is designed to fit within the XML syntax.

The checksum can be kept in place after reception to check the data integrity at later times.

The receiving side can detect missing messages, even if no acknowledges are used. It is also possible to detect missing messages (and checksum errors) in saved BM800 output data.

It is possible to re-initialize any side at any time. (Ok, a transmission in progress will probably be lost.)

This message handling handles the "lost acknowledge" problem in a correct way.

Both sides can actually transmit at the same time. The message ID sequences in each direction are independent of each other. But acknowledge messages may be delayed due to large outgoing transmissions.

4.3.10. Final tips and tricks

The BM800 might send informational log texts between correctly formatted messages. A sample receiving system shall disregard all characters received between a package end and a package begin token (See 4.1). (Or it might log them in a separate log file.)

A BM800 operator can manually send samples more than once. A sample receiving system connected to a BM800 that stores samples in some kind of database should be prepared to handle the same sample multiple times. Use the combination of the parameters SNO, DATE and SEQ (See 3.4.1.2, 3.5.1.2, and 3.5.1.3) to detect duplicate samples. Duplicate samples should be silently dropped. (But the message should of course be acknowledged first.)

Never disconnect one BM800 and reconnect another one without restarting the sample receiving system. (Risk of dropping messages due to duplicate message ID:s)

5. References

[1] DuCharme, Bob. *"XML- The Annotated Specification"*, 1999, ISBN 0-13-082676-6.