NOVITUS

Communication protocol XML

Version: 1.0.1
English

Warning !

Data of each packet can be optionally completed with check sum (CRC32 standard). General information about CRC32 check sum can be found at:

http://pl.wikipedia.org/wiki/Cykliczny_kod_nadmiarowy.

Using check sum makes sense wherever there may be noise and distortion and the transport protocol itself does not contain mechanisms of error correction. For fiscal printers check sums make sense in the case of RS232 and USB and there is no need to apply them in the case of TCP / IP.

If needed detailed information please contact technical support in NOVITUS.

Example oif packet without check sum:

```
<packet>
  <info action=”transaction”/>
</packet>
```

Example of the same packet but with check sum:

```
<packet crc="BB1E3EC8">
  <info action=”transaction”/>
</packet>
```

Communication buffer size is 5000 bytes. In the case where the data packet exceeds this value, the data must be divided into smaller data packets and send individually.

1. Read information about device readness for data processing (DLE).

In response for question:

```
<packet>
  <dle/>
</packet>
```

the printer sends back information in the following format:

```
<packet>
  <dle online="yes" papererror="no" printererror="no"/>
</packet>
```

where:

online – when printer is in online mode „yes”, otherwise „no”.
papererror – when there is no paper - „yes”, otherwise „no”.
printererror – when occured problem with printing mechanizs - „yes”, otherwise „no”.

2. Read information about logical status of device (ENQ).

In response for question:

```
<packet>
  <enq/>
</packet>
```

the printer sends back information in the following format:

```
<packet>
  <enq  fiscal="yes" lastcommanderror="no" intransaction="no" lasttransactioncorrect="yes"/>
</packet>
```

where:

Fiscal – when printer is in fiscal mode „yes", when in training mode „no".
Lastcommanderror – when the last command was executed correctly „yes" , otherwise „no".
Intransaction – when printer is in transaction mode „yes", otherwise „no".
Lasttransactcorrect – when the last transatcion was completed correctly „yes", otherwise „no".

3. Start the invoice.

After sending below packet, the printer receives order of starting invoice:

```
<packet>
 <invoice action="begin" number="120/2012"nip="" description="both"  paymentname="cash"
     paymentdate="dd-mm-yyyy" recipient="" issuer="" copies="1" margins="yes"
     signarea="yes" customernameoptions="info" sellernameoptions="none" paidlabel="paid by
     cash" selldate="dd-mm-yyyy">

  <customer>data of receiver 1</customer>
  <customer>data of receiver 2</customer>
  <customer>data of receiver 3</customer>
  <customer>data of receiver 4</customer>
  <customer>data of receiver 5</customer>
  <customer>data of receiver 6</customer>
  <option id="1"></option>
  <option id="2"></option>
  <option id="11"></option>

 </invoice>
</packet>
```

where:

action = "begin" – starting invoice.
number – invoice number (not required).
nip – buyer Nip number (not required).
description – takes the following values (not required):

1. description = „both" – original and copy (default value).
2. description = "orginal" – only original.

paymentname – form of payment (not required).
paymentdate – date of payment (format: day – month – year).
recipient – recipient _ (not required).
issuer – issuer_ (not required).
copies – number of copies – default 0 (not required).
margines – margineses_ - takes values „yes" or „no" (not required).
signarea – place for signature – takes values „yes" or „no" (not required).
customernameoptions – takes the following values:

1. customernameoptions = „info" – option of buyer's name (informative only).
2. customernameoptions = „all" – name and information block (default value).
3. customernameoptions = „none" – no information about buyer.

sellernameoptions – takes the following values:

1. sellernameoptions = „info" – options of seller's name (informative only).
2. sellernameoptions = „all" – namme and information block (default value).
3. sellernameoptions = „none" – no information about seller

paidlabel – informative inscription about payment form (not required).
selldate – date of sale (format: day – month – year, not required).

tag <customer> – additional invoice line for receiver data (not required).
tag <option> - id takes the following values (not required),

1 – skip the verbal description of the amount to pay,
2 – skip block of gross amounts in tax settlement,
3 – bold label "buyer".,
4 – bold label „seller",
5 – bold NIP number of buyer,
6 – bold NIP number of seller,

7 – print label "description / symbol" in the header of invoice before the invoice items,
8 – print an item number in the invoice item,
9 – label "to pay" settlement before block of taxes settlement,
10 – print amount of pennies in the verbal form,
11 – don't print date of sale if it is the same like the invoice date,
12 – don't print seller's data,
13 – not print descriptions for invoice items (information describing the contents of the fields in the invoice items),
14 – included payments service (such as a receipt),
15 – don't print receiver's data (doesn't concern fields with nip no. and others identification data),
16 – print inscription „Paid by cash",
17 – skip label „seller:",
18 – skip label „ORIGINAL" (works only in the absence of copies to print),
19 – print label „VAT INVOICE" (instead of label „INVOICE"),

4. Invoice cancellation.

After sending below packet, the printer receives order of invoice cancellation.

```
<packet>
  <invoice action="cancel"></invoice>
</packet>
```

5. Invoice closing.

After sending below packet, the printer receives order of invoice closing.

```
<packet>
  <invoice action="close" total="123.32" systemno="123" checkout="1" cashier="Jan">
  </invoice>
</packet>
```

where:

total – amount to pay,
systemno – system number (@ - if number is to be a QR – code, # - if number is to be a bar code),
checkout – checkout number,
cashier – cashier name.

Function of closing invoice allows to call additional options which should be placed in tag „<invoice>", which is responsible for invoice closing. The additional options are:

5.1. Discount / mark-up to the whole invoice.

```
<packet>
  <discount value="23%" name="Occasional" descid="1" action="discount"></discount>
</packet>
```

where:

value – amount of given discount / mark-up,
name – name of given discount / mark-up (in case when „name" is already completed, parameter „descid" is ignored, not required),
action – takes the following values (not required):
1. action = „discount" – discount_ (default value),
2. action = „markup" – mark-up_

5.2. Additional non-fiscal line.

```
<packet>
  <additional type="line" id="1" value="1234"></additional>
</packet>
```

where:

type – default value is „line” (not required),
value – value which is to be printed.

5.3. Definition of invoice’s footer.

```
<packet>
  <additional type="definition">
                <line>abc line 1</line>
                <line>abc line 2</line>
                <line>abc line 3</line>
      </additional>
</packet>
```

Warning !

There is possible one footer’s change between daily report.

Example:

```
<packet>
  <invoice action="close" systemno="123" checkout="1" cashier="Jan" total="123.32">

  <discount value="23%" name="Ocasional" descid="1" action="discount"></discount>

  <additional type="line" id="1" value="1234"></additional>

  <additional type="definition">
                <line>abc line 1</line>
                <line>abc line 2</line>
                <line>abc line 3</line>
 </additional>

 </invoice>
</packet>
```

As a result of closing invoice with all the options, there is given an occasional discount of 23% of the entire invoice. Another function is to add an additional line that displays the information (in this case points), whose value is: 1234. To invoice are also added three lines of the invoice’s footer.

6. Beginning a receipt.

After sending below packet, the printer receives order of beginning receipt:

```
<packet>
  <receipt action="begin" mode="online"></receipt>
</packet>
```

where:
mode – takes the following values (not required):
1. mode = „online” – receipt online mode (default value)
2. mode = „offline” – receipt offline mode.

7. Receipt cancelling.

After sending below packet, the printer receives order of receipt cancelling:

```
<packet>
  <receipt action="cancel"></receipt>
</packet>
```

8. Receipt closing.

After sending below packet, the printer receives order of closing receipt:

```
<packet>
  <receipt action="close" systemno="123" checkout="1" cashier="Jan" total="123.32">
  </receipt>
</packet>
```

where:

systemno – receipt number (@ - if number is to be a QR – code, # - if number is to be a bar code),
checkout – checkout number (not required),
cashier – cashier name (not required),
total – verifies receipt value calculated by the printer and the program (not required).

Function of closing receipt allows to call additional options which should be placed in tag „<receipt>”, which is responsible for receipt closing. The additional options are:

8.1. Discount / mark-up to the whole receipt.

```
<packet>
  <discount value="23%" name="Occasional" descid="1" action="discount"></discount>
</packet>
```

where:

value – amount of given discount / mark-up,
name – name of given discount / mark-up (in case when field „name” is already completed, the parameter „descid” is ignored, (not required),
action – takes the following values (not required):

1. action = „discount” – discount_ (default value),
2. action = „mark-up”.

8.2. Additional non-fiscal line.

```
<packet>
  <additional type="line" id="1" value="1234"></additional>
</packet>
```

where:
type – default value is „line” (not required),
value – value which is to be printed.

8.3. Definition of receipt’s footer.

```
<packet>
```

```
  <additional type="definition">
                 <line>abc line 1</line>
                 <line>abc line 2</line>
                 <line>abc line 3</line>
    </additional>
</packet>
```

Warning !

There is possible one footer's change between daily report.

Example:

```
<packet>
  <receipt action="close" systemno="123" checkout="1" cashier="Jan" total="123.32">

  <discount value="23%" name="Occasional" descid="1" action="discount"></discount>

  <additional type="line" id="1" value="1234"></additional>

  <additional type="definition">
                 <line>abc line 1</line>
                 <line>abc line 2</line>
                 <line>abc line 3</line>
 </additional>

 </receipt>
</packet>
```

As a result of closing receipt with all the options, there is given an occasional discount of 23% of the entire receipt's sum. Another function is to add an additional line that displays the information (in this case points), whose value is: 1234. To receipt are also added three lines of the receipt's footer.

9. Forms of payment.

After sending below packet, the printer receives order of choosing forms of payment.

```
<packet>
    <payment type="cash" action="add" value="1.23" rate="4.32345678" mode="rest"
     name="EUR">
    </payment>
</packet>
```

where:

type – takes the following values:
1. type = „card",
2. type = „cheque",
3. type = „voucher",
4. type = „credit",
5. type = „transfer",
6. type = „account",
7. type = „foreign",
8. type = „cash",

value – amount of payment,
rate – exchange rate in foreign currency (not required, dafault value is 1),
mode – takes the following values (not required),
1. mode = „payment" (default value),
2. mode = „rest",

name – currency name (not required).

Warning !

If the parameter rate varies from 1, then the parameter "value" is interpreted as the value of payments in foreign currency (the value in the current currency is calculated as the product of the parameter "value" and "rate").

10. Adding item to receipt / invoice.

After sending below packet, the printer receives order of adding item to receipt / invoice.

```
<packet>
  <item name="name" quantity="2" quantityunit="pcs" ptu="A" price="1.23" total="2.46"
   recipe="" charge="" plu="" description="" action="sale">
  </item>
</packet>
```

where:

name – name of added article,
quantity – article's amount,
ptu – VAT rate,
price – price per item,
total – calculated as product of parameter "quantity" and "price" (not required),
recipe – required only in pharmacies version,
charge – required only in pharmacies version,
plu – article's code (@ - if code is to be QR – code, # - if code is to be a bar code),
description – article's description (not required),
action – takes the following values (not required):

1. action = „Sale" – sale_ (default value),
2. action = „Storno" – storno_(reversal).

Function of adding receipt's / invoice's allows to call additional option, which is giving discount / mark-up to the concrete item. The option is defined in the following way:

```
<packet>
  <discount value="23%" name="Occasional" descid="1" action="discount"></discount>
</packet>
```

where:

value – amount of given discount / mark-up,
name – name of given discount / mark-up (in case when field „name" is already completed, the parameter „descid" is ignored, not required),
action – takes the following values (not required):

1. action = „discount" – discount_ (default value),
2. action = „markup" – mark-up_.

Warning !

Option of giving discount / mark-up to the concrete item must be placed in tag „<item>", which is responsible for item, to which discount / mark-p is to be given.

Example:

```
<packet>
  <item name="exemplary article" quantity="1" quantityunit="pcs" ptu="A" price="1.23"
   total="1.23" recipe="" charge="" plu="" description="" action="sale">

     <discount value="23%" name="Occasional" descid="1" action="discount"></discount>

   </item>
</packet>
```

As a result of sending this data packet to the receipt / invoice, there will be added article named “exemplary article” , whose price is 1.23. There was given an occasional discount to the concrete item of 23% of the article’s value.

11. Settlement of returnable containers.

After sending below packet, the printer receives order of returnable containers settlement.

```
<packet>
 <container action="sale" price="1.00" type="out" quantity="1"></container>
</packet>
```

where:

action – takes the following values (not required):

1. action = „sale” – sale_ (default value),
2. action = „storno” – storno_ (reversal),

price – container’s price,
type – takes the following values (not required):

1. type = „out” – container’s sale (default value),
2. type = „In” – container’s return,

quantity – number of containers (default 1, not required).

12. Giving discount / mark-up.

After sending below packet, the printer receives order of giving discount / mark-up to the sum of all items from receipt.

```
<packet>
 <discount value="23%" name="Occasional" descid="1" type="subtotal" total="2" ptu="A"
  action="discount"></discount>
</packet>
```

where:

value – amount of given discount / mark-up,
name – name of given discount / mark-up (in case when field „name” is already completed, the parameter „descid” is ignored, not required),
type – takes the following values (not required):

1. type = „subtotal” – discount to the subtotal of receipt (default value),
2. type = „intransaction” – discount during the transaction. (This type displays information about discount’s amount. (Discount isn’t printed on receipt),

total – sum of all item’s values on receipt before giving discount / mark-up,
ptu – VAT rates,
action – takes the following values (not required):

1. action = „discount” – discount_ (default value),
2. action = „markup” – mark-up_.

Warning !

1. When you set type = "subtotal", then it is possible to add field "total", thanks to which before giving discount to the subtotal, it is checked its value and the sum of all the items on receipt and an error is reported in case of difference between these values.
2. When you set type = „subtotal”, then it is possible to add field „ptu”. In such a case the discount will be given to the group of articles in the concrete rate.
3. When you set type = „intransaction”, the parameter „action” may take the additional value „none” – shortage of any discount / mark-up, cancelling earlier given discount / mark-up and all forms of payment.

13. Information about date and version of device.

In response to question:

```
<packet>
  <info action="date" version="" date=""></info>
</packet>
```

the printer sends back values in the following format:

```
<packet>
  <info action="date" version="" date="08-10-2013 11:13"/>
</packet>
```

where:

action – takes the following values:
1. action = „date” – sending back information about date and time,
2. action = „version” – sending back information about version of device.

14. Checkout information.

In response to question:

```
<packet>
  <info action="checkout" type="receipt" lasterror="?" isfiscal="?" receiptopen="?"
        lastreceipterror="?" resetcount="?" date="?" receiptcount="?" cash="?" uniqueno="?"
        lastreceipt="?" lastinvoice="?" lastprintout="?"></info>
</packet>
```

the printer sends back values in the following format:

```
<packet>
  <info action="checkout" type="receipt" lasterror="0" isfiscal="yes" receiptopen="no"
        lastreceipterror="no" resetcount="0" date="08-10-2013" receiptcount="11" cash="444.76"
        uniqueno="ABC12345678" lastreceipt="11" lastinvoice="1" lastprintout="47">
        <ptu name="A">12.50</ptu>
        <ptu name="B">0.00</ptu>
        <ptu name="C">0.00</ptu>
        <ptu name="D">0.00</ptu>
        <ptu name="G">0.00</ptu>
  </info>
</packet>
```

where:

type – takes the following values:
1. type = „receipt” – sending back receipt’s toatlizers since the last daily report (default value),
2. type = „invoice” – sending back invoice’s totalizers since the last daily report,

lasterror – number of the last error,
isfiscal – if printer is fiscalised,
receiptopen – if receipt is openy,
lastreceipterror – if there was an error during the last transaction,
resetcount – number of resettins in the memory,
date – current date,
receiptcount – number of receipts,
cash – cash_,
uniqueno – unique number,
lastreceipt – number of the last receipt,
lastinvoice – number of the lasr invoice,
lastprintout – number of the last printout,
name – rate name.

15. Information about fiscal memory.

In response to question:

```
<packet>
  <info action="fiscalmemory" ></info>
</packet>
```

the printer sends back values in the following form:

```
<packet>
  <info action="fiscalmemory" fiscalmemorysize="1048576" recordsize="464" fiscal="1"
    uniqueno="ABC12345678" nip="123-456-78-90" maxrecordscount="2144" recordscount="7"
    maxreportscount="1830" reportscount="4" resetmaxcount="200" resetcount="0"
    taxratesprglimit="30" taxratesprg="1" currencychangeprglimit="4" currencychangeprg="0"
    fiscalstartdate="dd-mm-yyyy hh:dd:ss" fiscalstopdate="dd-mm-yyyy hh:dd:ss"
    currencyname="PLN">
<ptu name="A">123.23</ptu>
<ptu name="B">123.23</ptu>
<sale>999.23</sale>
</packet>
```

where:

fiscalmemorysize – size of fiscal memory (in bytes),
recordsize – size of record (in bytes),
fiscal – work mode. Takes the following values:

1. fiscal = „0" – non-fiscal,
2. fiscal = „1" – fiscal,
3. fiscal = „2" – fiscal closed,

uniqueno – unique number,
nip – NIP number (number of tax identification),
maxrecordcount – maximum amount of records,
recordscount – number of records,
maxreportscount – maximum amount of daily reports,
reportscount – number of daily reports,
resetmaxcount – maximum amount of RAM resettings,
resetcount – number of RAM resettings,
taxratesprglimit – maximum amount of PTU rates changes,
taxratesprg – number of PTU rates changes,
currencychangeprglimit – maximum amount of currency changes,
currencychangeprg – number of currency changes,
fiscalstartdate – date of fiscalization,
fiscalstopdate – date of fiscal mode closing,
currencyname – name of the current currency.

16. Information about SD card.

In response to question:

```
<packet>
  <info action="sdcard" type="external"></info>
</packet>
```

the printer sends back values in the following format:

```
 <packet>
  <info action="sdcard" type="external" state="open" size="2145124352" free="2045"
    reportscount="3" lastreport="4" lastaccess="08-10-2013 00:00"/>
 </packet>
```

where:

type – takes the following values (not required):

1. type = „external" – external_ (default value),

state – card state. Takes the following values:

1. state = „open” – open_,
2. state = „close” – closed,
3. state = „unknow” – unknown,
4. state = „removed” – removed_ (no card),
5. state = „error” – error on card,
6. state = „process” – status during processing,

size – card size (in bytes),
free – how many free space,
reportscount – amount of reports files,
lastreport – number of the last daily report,
lastaccess – date of the last modification on the card.

17. Request about current transaction status.

In response to question:

```
<packet>
 <info action="transaction"></info>
</packet>
```

the printer sends back values in the following format:

```
<packet>
 <info action="transaction" nettotal="233" grosstotal="343" type="receipt" mode="online">
    <total name="A" tax="23" net="100" gross="123"></total>
 </info>
</packet>
```

where:

nettotal – sum of net totalizers,
grosstotal – sum of gross totalizers,
type – type of transaction. Takes the following values:

1. type = „none” – no transaction. No other data are sent back in this case,
2. type = „receipt” – open receipt. in this mode there is additionally sent back parameter mode = „online” or mode = „offline”,
3. type = „container” – container’s settlement,
4. type = „invoice” – open invoice,

name – rate name,
tax – tax value,
net – net totalizer’s value
gross – gross totalizer’s value.

18. Request about status of active or recent transaction.

In response to question:

```
<packet>
 <info action="lasttransaction"></info>
</packet>
```

the printer sends back values in the following format:

```
<packet>
 <info action="lasttransaction" total="343" type="receipt" state="open" number="23"
    date="dd-mm-yyyy hh:dd" printoutno="123" checkout="1" cashier="Jan"
    systemno="1234" discountcalc="1" euroexchangerate="4.344" rest="30" eurorest="3">

    <item name="name" code="1234" quantity="4" ptu="A" price="2.10" total="8.40"
      totaldiscount="12%" totalafterdiscount="7.20">
```

```
            <discount value="5" name="Occasional" total="2" type="discount"
            descid="1"></discount>
        </item>
        <container code="2345" quantity="1" price="0.40" total="0.40"></container>
        <payment name="Gift" type="card" value="12"></payment>
        <discount value="23%" name="Occasional" total="2" type="discount" ></discount>
  </info>
</packet>
```

where:

type – type_. Takes the following values:
1. type = „receipt" – receipt / invoice,
2. type = „container" – container's settlement,

state – status. Takes the following values:
1. state = „open" – open_,
2. state = „close" – closed,
3. state = „canceled" – canceled_,

number – receipt number,
date – date and time of receipt / invoice,
printoutno – printout number,
checkout – checkout number,
cashier – cashier name,
systemno – system number,
discountcalc – method of discount calculation,
euroexchangerate – foreign currency exchange rate,
rest – rest_ (change),
eurorest – rest in foreign currency,
name – discount name,
value – discount amount (numerical or percentage value),
totaldiscount – total percentage discount, which results from given discounts to the item,
totalafterdiscount – value of the item after given discounts,
total – value,
quantity – number of pieces of the concrete item (position).

19. Controlling instructions.

19.1. Beep signal.

```
<packet>
  <control action="beep"></control>
</packet>
```

19.2. Paper feeding of the number of lines.

```
<packet>
  <control action="paperfeed" value="6"></control>
</packet>
```

where:

value – number of lines, of which paper is to eject.

19.3. Opening drawer.

```
<packet>
  <control action="drawer"></control>
</packet>
```

19.4. Showing text on displayer.

```
<packet>
  <control action="display" line1="test1" line5="test5" line8="test8"></control>
</packet>
```

where:

Line1... 8 – text shown on displayer.

20. Daily report.

Afetr sending below packet, the printer receives order of printing a daily report.

```
<packet>
  <report type="daily" date="04-06-2004"></report>
</packet>
```

where:

date – date of daily report (not required).

21. Periodical report.

After sending below packet, the printer receives order of printing a periodical report.

```
<packet>
  <report type="periodical" from="04-06-2004" to="06-06-2006" checkout="1" cashier="Jan"
   kind=""></report>
</packet>
```

where:

from, to – range of report's dates or numbers,
checkout – checkout number (not required),
cashier– cashier name (not required),
kind – takes the following values:

1. kind = „full" – full report (fiscal document), range from given dates or numbers (default value),
2. kind = „salesummary" – sale summary _ (non-fiscal document), range from given dates or numbers,
3. kind = „monthllyfull" – full monthly report (fiscal document),
4. kind = „monthlysummary" – monthly sale summary (non-fiscal document),
5. kind = „billingfull" – full billing report, range from given dates,
6. kind = „billingsummary" – billing report summary, range from given dates.

22. Shift report.

After sending below packet, the printer receives order of printing a shift report.

```
<packet>
  <report type="cash" checkout="1" cashier="Jan" begin="" end="" income="10"
   expense="10" cash="23.23" payin="40" payout="10" balance="30" receiptcount="12"
   canceledreceiptcount="1" stornocount="0">
    <payment name="gift card" value="234.12" type="card"></payment>
    <container type="out" value="23"></container>
  </report>
```

```
</packet>
```

where:

checkout – checkout number (not required),
cashier – cashier name (not required),
begin – begin_,
end – end_,
income – income_,
expense – rexpense_,
cash – cash_,
payin – payin_,
payout – payout_y,
balance – cash balance,
receiptcount – amount of receipts,
canceledreceiptcount – number of cancelled receipts,
stornocount – amount of storno (reversal),
payment type – type of payment. Takes the following values:

1. type = „cash”,
2. type = „card”,
3. type = „cheque”,
4. type = „voucher”,
5. type = „credit”,
6. type = „transfer”,
7. type = „account”,
8. type = „foreign”,

container -> type – takes the following values:

1. type = „out” – taken (sale),
2. type = „in” – returned (container return)

container -> value – value of returned / taken deposit.

23. Data report from E copy.

In order to read data from E copy, you must send to the printer the following data packet:

```
<packet>
  <report type="ecopy" from="04-06-2004" to="06-06-2006" kind=""
   uniqueno="ABC11112222"></report>
</packet>
```

where:

from, to – dates / numbers range (day-month-year),
kind – takes the following values:

1. kind = „receipt” – receipts,
2. kind = „invoice” – invoices,
3. kind = „dailyreport” – daily reports,
4. kind = „nonfiscal” – non-fiscal printouts,
5. kind = „all” – all printouts,

uniqueno – unique number (not required),

24. Cashier’s log-in and log-off.

Cashier’s log-in is done by sending the following packet:

```
<packet>
  <cashier action="login" number="1" name="Jan"></cashier>
</packet>
```

where:

action – takes the following values:

1. action = „login” – cashier’s log-in,
2. action = „logoff” – cashier’s log-off,

number – checkout number (not required),
name – cashier name (not required).

25. Pay-in, pay-out and cash balance.

Pay-in to checkout is done by sending the following packet:

```
<packet>
 <cash action="payin" value="2.30" checkout="1" cashier="Jan"></cash>
</packet>
```

where:

action – takes the following values:

1. action = „payin” – pay-in to the checkout,
2. action = „payout” – pay-out from the checkout,
3. action = „read” – returns current account balance,

checkout – checkout number (not required),
cashier – cashier name (not required),

26. Error handling mode.

After sending below packet, in the printer would be set error handling mode.

```
<packet>
 <error action="set" value=""></error>
</packet>
```

where:

value – takes the following values:

1. value = „display” – error is shown on display (not required),
2. value = „silent” – silent mode. No messages in the printer.

In order to receive the last error number, you must send to the printer the following packet:

```
<packet>
 <error action="get" value=""></error>
</packet>
```

27. Control of goods base.

Executing control of goods base is done by sending the following packet:

```
 <packet>
 <dbcheck action="begin" mode="all" pluname="pluname" ptu="A" checkout="1"
  cashier="Jan"></dbcheck>
</packet>
```

where:

action – takes the following values:

1. action = „begin” – report begin,
2. action = „check” – checking / printing goods,
3. action = „end” – report end,

mode – takes the following values:

1. mode = „all” – printing all goods (default value),
2. mode = „lock” – printing only blocked goods,

pluname – article name,
ptu – VAT rate,
checkout – checkout number (not required),
cashier – cashier name (not required),

28. Configuration.

In order to set the parameters you must send packet in the folowing form:

```
<packet>
 <config action="set">
    <set id="0">200</set>
    <set id="1">125</set>
    <set id="discountcalculation">0</set>
 </config>
</packet>
```

where:

action – takes the following values:

1. action = „set" – setting parameters,
2. action = „get" – getting parameters. In response you receive only these parameters that you asked about,

id = „discountcalculation" – checking method of discount calculating.

Warning !

All configuration options and possible configuration parameters are presented in Description of communication protocol of NOVITUS.

29. Setting date and hour.

In order to set date and hour in the printer you must send packet in the following form:

```
<packet>
 <clock date="dd-mm-yyyy hh:dd"></clock>
</packet>
```

where:

DD – day,
MM – month,
YYYY – year,
HH – hour,
DD – minute.

30. Programming the header.

In order to program the header you must send to the printer packet in the following form:

```
<packet>
 <header action="set">
    <line bold="yes" align="center">This is a text header</line>
    <line bold="yes" >This is a text header, line 2</line>
    <line>This is a text header, line 3</line>
    <line></line>
    <line>Next header line</line>
 </header>
</packet>
```

where:

action – takes the following values:

1. action = „set” – programming header,
2. action = „get” – read header,

bold – bold_ (not required),
align – takes the following values (not required);

1. align = „center” – centered,
2. align = „left” – to the left (default value),
3. align = „right” – to the right,

31. Service blockade.

In order to set a service blockade you must send to the printer packet in the following form:

```
<packet>
  <service action="lock" date="dd-mm-yyyy" description="service blockade"
   password="1234"></service>
</packet>
```

where:

action – takes the following values:

1. action = „lock” – service blockade,
2. action = „unlock” – unlock service blockade (password required, if blockade was set with password),

date – date (day – month – year). If there is no date set, the review is canceled.
description – description_ (not required),
password – password_ (not required).

32. Service review.

To set date of a service review you must send to the printer packet in the following form:

```
<packet>
  <service action="review" date="dd-mm-yyyy" description="periodical review"></service>
</packet>
```

where:

date – date of review (day – month – year),
description – description_ (not required).

33. Getting data from E copy.

In order to get data from E copy you must send to the printere packet in the following form:

```
<packet>
  <ecopy action="begin" from="0" to="234" file="0001.txt" uniqueno="12123as"
source="sdcard" read="alldata"></ecopy>
</packet>
```

where:

action – takes the following values:

1. action = „begin” – beginning data collecting,
2. action = „next” – next data packet,
3. action = „repeat” – sending again recent sent back data,

from – takes values in format date (day-month-year) or number. It’s responsible for begin of data getting range (not requested).
to – takes value in format date (day-month-year) or number. It’s responsible for end of data gettong range (not requested).
file – file name (requested full path to the file),

uniqueno – unique number of the card,
source – read source. Takes the following values:

1. source = „sdcard” – SD card,
2. source = „cache” – cache memory,

read – read type. Takes the following values:

1. read = „alldata” – read all data,
2. read = „contentlist” – read table of contents,

In response to above data packet the printer sends back the following returnable packet:

```
<packet>
  <ecopy action="data" size="2048" file="0001.txt">E copy data</ecopy>
</packet>
```

where:

action – takes the following values:

1. action = „data” – information that in queue waits the next data packet,
2. action = „end” – information that the last data packet was read,

size – size of sent back data,
file – name of sent back file,

Warning !

The printer sends back individual data packets. Until Action = "date" the next packet should be sent to the printer with request of parameter Action = "next" to get the next data packet. Parameters "From" and "To" determine the scope of collected data and they are taken into account only when the parameter File is empty. Data are collected in hexadecimal format.

34. Data verification.

In order to verify data you must send to the printer the following data packet:

```
 <packet>
  <verification action="begin"  type="receipt" uniqueno="ABC012345678" verifycode =
   "ABAB...">Data to verification</verification>
</packet>
```

where:

action – this attribute is essential only while E copy verification. Takes the following values:

1. action = „begin” – begin of sending data to verification,
2. action = „next” – the next data packet to verification,
3. action = „end” – end of sending data to verification,

type – takes the following values:

1. type = „receipt” – receipt_,
2. type = „invoice” – invoice_,
3. type = „dailyreport” – daily report_,
4. type = „file” – E copy file,

uniqueno – unique number,
verifycode – verification code.

Warning !

The first data packet must be transferred with set mode action = "begin". Every other one must have mode action = "next", apart from the last packet. The last packet of data must include action = "end" and filled the box "verifycode". In case when all the data are in one package, two packages should be sent - initial with data and the other with empty data and mode action = "end". Data to E copy files verification are accepted in hexadecimal encoding, and size of a single data packet can’t exceed 2048 characters. In case of receipts, invoices and daily reports, verified data must be encoded in ASCI.

35. Programming currency change.

In order to program currency change you must send to the printer packet in the following form:

```
<packet>
  <currency action="change" date="dd-mm-yyyy hh:dd" name="EUR"
   exchangerate="4.00000001"></currency>
</packet>
```

where:

date – date and time of chanage (day – month – year   hour:minute),
name – three digits name of currency accordable with convention used by NBP,
exchangerate – exchange rate (from: 0:00000001 to: 9999:99999999),

36. Programming mode of currency conversion’s printout.

In order to program mode of currency conversion’s printout you must send to the printer packet in the following form:

```
<packet>
  <currency action="print" name="EUR" exchangerate="4.00000001"
   type="defined"></currency>
</packet>
```

where:

name – three-digits name of currency accordable with convention used by NBP (National Bank of Poland),
exchangerate – exchange rate_ (from: 0:00000001 to: 9999:99999999),
type – takes the following values:

1. type = „defined” – defined mode. Fields „Name” and „exchangerate” are required,
2. type = „none” – no conversion

37. Setting PTU rate.

In order to set PTU rate you must send to the printer packet in the following form:

```
<packet>
  <taxrates action="set" checkout="1" cashier="Jan" date="dd-mm-yyyy hh:dd">
  <ptu name="A">23%</ptu>
  </taxrates>
</packet>
```

where:

action – takes the following values:

1. action = „set” – setting rates,
2. action = „get” - reading rates,

checkout – checkout number (not requested),
cashier – cashier name (not requested),
date – date and hour of modification (day – month – year   hour : minutes),
name – PTU rate name.

Warning !

To make a tax-free rate you must in tag "ptu" enter value „free” instead of a percentage value.
Example:

```
<packet>
  <taxrates action="set" checkout="1" cashier="Jan" date="10-10-2013 10:00">
  <ptu name="A">23%</ptu>
  <ptu name="B">free</ptu>
  </taxrates>
</packet>
```

After executing this packet two VAT rates would be set. Rate A contains 23% of tax, rate B is free of tax.

38. Request of sending back data from fiscal memory.

In order to read data from fiscal memory you must send to the printer the following packet:

```
<packet>
  <fiscalmemory action="get" from="dd-mm-yyyy hh:dd"></fiscalmemory>
</packet>
```

then you must send the same data packet with parameter action=”next” in order to read the next data packets.

where:

action – takes the following values:

1. action = „get” – request of sending back data,
2. action = „next” – sending back the next record,

from – date and hour or record number.

Sent back records may take the following types:

38.1. Type = „report” – Record of daily report.

```
<packet>
  <fiscalmemory type="report" time="dd-mm-yyyy hh:dd" receiptcount="23" invoicecount="3"
   canceledreceiptcount="2" number="1" lastreceipt="4" lastinvoice="2" lastprintout="20"
   sale="12342.33" tax="20" totalsale="123213.12" totaltax="232.12" invoicesale="21.33"
   invoicetax="22.12" invoicetotalsale="1213.52" invoicetotaltax="3231.12"
   currencyname="PLN">

  <ptu name="A" rate="23%" sale="2.31" tax="23.12" totaltax="21142.23" invoicesale="2.20"
   invoicetax="0.23" invoicetotaltax="232.12"></ptu>

   </fiscalmemory>
</packet>
```

where:

time – time of record registering,
receiptcount – amount of receipts,
invoicecount – amount of invoices,
canceledreceiptcount – amount of canceled receipts,
number – report number,
lastreceipt – last receipt number,
lastinvoice – last invoice number,
sale – sum of gross sale (receipts),
tax – suma of PTU tax (receipts),
totalsale – sum of sale increasingly (receipts),
totaltax – sum of tax increasingly (receipts),
invoicesale – sum of sale (invoices),
invoicetax – sum of tax (invoices),
invoicetotalsale – sum of sale increasingly (invoices),
invoicetotaltax – sum of tax increasingly (invoices),
currencyname – currency name _,
name = "A".."G" – rate name,
rate – tax rate,

sale – sale in the rate (receipts),
tax – tax value in the rate (receipts),
totaltax – tax value increasingly in the rate (receipts),
invoicesale – sale in the rate (invoices),
invoicetax – tax value in the rate (invoices),
invoicetotaltax – tax value increasingly in the rate (invoices).

38.2. Type = „reset" – RAM memory resetting.

```
<packet>
  <fiscalmemory type="reset" time="dd-mm-yyyy hh:dd" number="4"
   reason="internal"></fiscalmemory>
</packet>
```

where:

time – time of record registering,
reason – reason of memory resetting (internal or external),

38.3. Type = „taxrates" – Programming VAT rates.

```
<packet>
  <fiscalmemory type="taxrates" time="dd-mm-yyyy hh:dd">
  <ptu name="A">23%</ptu>
  <ptu name="A">Free</ptu>
  </fiscalmemory>
</packet>
```

where:

time – time of record registering.

38.4. Type = „currencyset" – Record of currency change setting.

```
<packet>
  <fiscalmemory type="currencyset" time="dd-mm-yyyy hh:dd" name="EUR"
   changetime="dd-mm-yyyy hh:dd" exchangerate="4.3231"></fiscalmemory>
</packet>
```

where:

time – time of record registering,
name – currency name,
changetime – date and time of currency change,
exchangerate – exchange rate_

38.5. Type = „currencychange" – Record of currency change.

```
<packet>
  <fiscalmemory type="currencychange" time="dd-mm-yyyy hh:dd" name="EUR"
   changetime="dd-mm-yyyy hh:dd" exchangerate="4.3231"></fiscalmemory>
</packet>
```

where:

time – time of record registering,
name – currency name,
changetime – date and time of currency change,

exchangerate – exchange rate.

38.6. Type = „cardclose” – Record of card closing.

```
<packet>
  <fiscalmemory type="cardclose" time="dd-mm-yyyy hh:dd" number="23" closetime="dd-
   mm-yyyy hh:dd"></fiscalmemory>
</packet>
```

where:

time – time of record registering,
number – card number (1..999).

38.7. Type = „end” – Data end.

```
<packet>
  <fiscalmemory type="end"></fiscalmemory>
</packet>
```

39. Request of sending back logs from port.

In response to question:

```
<packet>
  <log action="read" port="com1"></log>
</packet>
```

the printer sends back information in the following format:

```
<packet>
  <log action="read" port="com1" date="dd-mm-yyyy hh:dd">data....</log>
  <log action="write" port="com1" date="dd-mm-yyyy hh:dd">data....</log>
</packet>
```

where:

Port – chosen port „com1”, „com2”,…,”comN” . If parameter port is absent, data from all ports are sent back,
Action – takes the following values:

1. Action = „read” – read data,
2. Action = „write” – sent data,

Date – date and time of collecting / sending data (day – month – year   hour : minutes).

40. Non-fiscal printout.

In order to make a non-fiscal printout you must send to the printer the following data packet:

```
<packet>
  <nonfiscalprintout systemno="123">
    <line type="line" bold="yes" inwers="yes" center="yes" fontid="1"
      fontattr="big">line1</line>
    <line type="underline"></line>
    <line>line 3</line>
  </nonfiscalprintout>
</packet>
```

where:

systemno – system number (not required),
type – takes the following values (parameter not requested):

1. type – „line” – text (default value),
2. type = „barcode” – bar code_,
3. type = „qrcode” – 2D code,
4. type = „underline” – underline_,

bold – bold_ (reduces of half number of characters in line, not requested),
fontid – font (1 – basic font, 2- the second font, parameter not requested),
fontattr – font attributes. Takes the following values (not requested):

1. fontattr = „big” – enlarged font (reduces of half number of characters in line. Enlargement contained in the font style),
2. fontattr = „high” – higher font,
3. fontattr = „bold” – bold font (reduces of half number of characters in line),
4. fontattr = „inwers” – inwers_,

Warning !

QR code (type = „qrcode”) takes specific characters and limitations that depend on device on which they are used. Detailed information concerning QR code operating are placed in Description of communication protocol of Novitus.

41. Programming a graphic.

In order to program a graphic header you must send to the printer the following data packet:

```
<packet>
  <graphic action="programm"  id="0" type="all" >graphic in hexadecimal form</graphic>
</packet>
```

where:

id – number of programmed graphic (0 – 50 where 0 is a graphic header),
type – takes the following values:

1. type =„begin” – begin of graphic data packet,
2. type = „next” – next packet of graphic data,
3. type = „end” – end of graphic,
4. type = „all” – all graphic in one data packet.

In order to delete graphic or get its check sum you must send the following data packet:

```
<packet>
  <graphic action="delete"  id="0"></graphic>
</packet>
```

where:

action – takes the following values:

1. action = „delete” – deleting a graphic header,
2. action = „readcrc” – reading a graphic check sum.

In order to read programmed graphic you must send the following data packet:

```
<packet>
  <graphic action="read"  id="0" type="begin" ></graphic>
</packet>
```

where:

action – takes the following values:

1. action = „begin” – begin of graphic reading,
2. action = „next” – next packet of graphic data,
3. action = „repeat” – sending back again the previous data packet.

In response to above packet, the printer sends back data in the following form:

```
<packet>
 <graphic action="read"  id="0" type="data" >graphic in hexadecimal form</graphic>
</packet>
```

where:

type – takes the following values:
1. type = „data” – the next data packet
2. type = „end” – end of graphic.

Warning !

In order to program a graphic you must send data in hexadecimal form. Graphic can be sent in one packet (provided that the packet doesn't exceed the size of memory buffer) or divided into smaller packets. In this case data in hexadecimal form must have an even number of characters.

42. Programming an animation.

To programm an animation you must send to the printer packets in the following form::

```
<packet>
 <animation action="programm"  name="test.png"  size="147682" type="begin"> data
  packet in encodeing base64</animation>
</packet>
```

where:

action = „programm” – programming an animation,
name – animation name with extension
type – takes the following values:
1. type = „begin” – the first data packet,
2. type = „next” – next data packets.

There are also possible other operations concerning an animation. In order to use one of them you must send to the printer the following data packet:

```
 <packet>
 <animation action="delete" name="test.png" > </animation>
</packet>
```

where:

action – takes the following values:
1. action = „delete” – deleting chosen animation,
2. action = „readcrc” – read check sum of chosen animation,
3. action = „setactive” – setting chosen animation as active one,
4. action = „hide” – switching off active animation (attribute 'name' not requested),
5. action = „deleteall” – deleting all animations placed in the printer (attribute 'name' not requested).

Warning !

To program the animation you must send data packet prepared in base64 encoding. Please note that the length of a single package doesn't exceed size of memory buffer, because the printer reports an error. Name must be unique and must be provided with file extension, while the animation size must be entered in bytes and it is a physical size of image on your hard drive. The first package is to be sent with attribute type = "begin", and the next packets with parameter type = "next". Sending ends when size of sent packets is equal to size given in animation attributes.

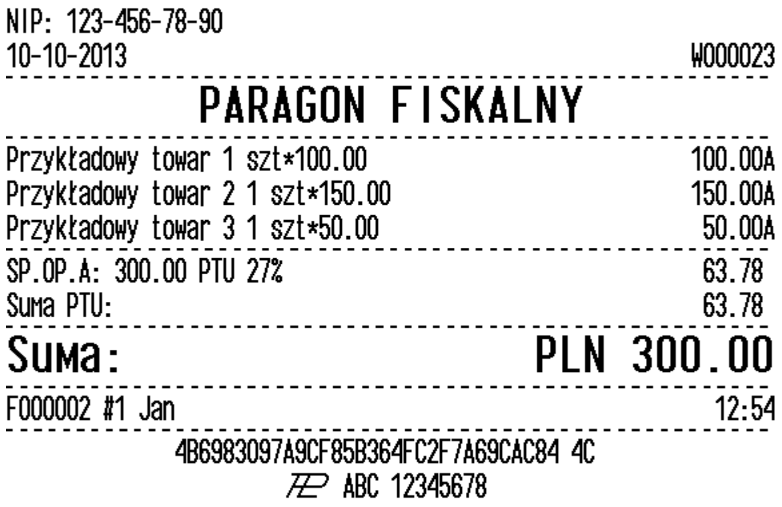
43. Exemplary printouts.

43.1. Fiscal receipt.

Below is presented packet printing receipt with 3 items (positions) on it:

```
<packet>
 <receipt action="begin" mode="online"></receipt>
 <item name="Exemplary article" quantity="1" quantityunit="pcs" ptu="A" price="100.00"
  recipe="" charge="" plu="" description="" action="sale"/>
 <item name="Exemplary article2" quantity="1" quantityunit="pcs" ptu="A" price="150.00"
  recipe="" charge="" plu="" description="" action="sale"/>
 <item name="Exemplary article3 " quantity="1" quantityunit="pcs" ptu="A" price="50.00"
  recipe="" charge="" plu="" description="" action="sale"/>
 <receipt action="close" systemno="123" checkout="1" cashier="Jan" total="300.00" />
</packet>
```

As result we receive:



```
NIP: 123-456-78-90
10-10-2013                                   W000023
----------------------------------------------------
               PARAGON FISKALNY
----------------------------------------------------
Przykładowy towar 1 szt*100.00               100.00A
Przykładowy towar 2 1 szt*150.00             150.00A
Przykładowy towar 3 1 szt*50.00               50.00A
----------------------------------------------------
SP.OP.A: 300.00 PTU 27%                       63.78
Suma PTU:                                     63.78
----------------------------------------------------
Suma:                                    PLN 300.00
----------------------------------------------------
F000002 #1 Jan                                 12:54
----------------------------------------------------
          4B6983097A9CF85B364FC2F7A69CAC84 4C
                 ABC 12345678
```

43.2. Invoice.

Below is presented packet printing invoice with 3 items (positions) on it:

```
<packet>
```

```
  <invoice action="begin" no="120/2012" nip="" description="both" paymentname="cash"
   paymentdate="10-10-2013" recipient="" issuer="" copies="1" margins="yes" signarea="yes"
   customernameoptions="info" sellernameoptions="none" paidlabel="paid by cash"
   selldate="10-10-2013">
  <customer>data of receiver1</customer>
  <option id="1"/>
  <option id="2"/>
  <option id="11"/>
   </invoice>
  <item name="Exemplary article" quantity="1" quantityunit="pcs" ptu="A" price="100.00"
   recipe="" charge="" plu="" description="" action="sale"/>
  <item name="Exemplary article2" quantity="1" quantityunit="pcs" ptu="A" price="150.00"
   recipe="" charge="" plu="" description="" action="sale"/>
  <item name="Exemplary article3" quantity="1" quantityunit="pcs" ptu="A" price="50.00"
   recipe="" charge="" plu="" description="" action="sale"/>
  <invoice action="close" total="300.00" systemno="123" checkout="1" cashier="Jan"/>
</packet>
```

As result we receive:

```
--------------------------------------------------------
                        FAKTURA
Numer:  120/2012                                 W000020
Data wystawienia: 10-10-2013
                        ORYGINAŁ
--------------------------------------------------------
Sprzedawca:

NIP: 123-456-78-90
--------------------------------------------------------
Nabywca:
dane odbiorcy 1

NIP:
--------------------------------------------------------
Forma płatności: gotówka
Data płatności: 10-10-2013
--------------------------------------------------------
Nazwa  Ilość*Cena Jedn.Brutto
  Wart.Netto  Stawka  Podatek                Wart.Brutto
--------------------------------------------------------
Przykładowy towar  1 szt*100.00
  78.74          27%:   21.26                     100.00
Przykładowy towar 2  1 szt*150.00
  118.11         27%:   31.89                     150.00
Przykładowy towar 3  1 szt*50.00
  39.37          27%:   10.63                      50.00
--------------------------------------------------------
Stawka  Wart. Netto  Wart. VAT
27%         236.22      63.78
--------------------------------------------------------
            236.22      63.78
--------------------------------------------------------
Suma:                                     PLN 300.00



.......................................
Imię, nazwisko, podpis osoby uprawnionej do
wystawienia faktury VAT


.......................................
Imię, nazwisko, podpis osoby uprawnionej do
odbioru faktury VAT
F000003 #1 Jan                                     12:53
            2054618430F77474D5643375779F43D1 24
                        ABC 12345678
```